Discussion/Clarifications from Google/Cloud Teams

GKE Observability Questions

General Questions

SNo.	Issues/Inputs Required	Details	Comments
1	Data Loss Prevention in GKE Managed solutions	Can you provide detail on any mechanisms in place to prevent data loss (for your managed solutions) if Cloud Monitoring, Cloud Logging or Cloud Trace become temporarily unavailable?	Client libraries use buffering mechanisms to handle temporary unavailability of the services **H.W Are there any figures that can be attributed to this? i.e. volume of data that can be buffered whilst backends are unavailable?** Answering at a granular level is difficult as the service, service configuration (sizing and libraries) vary greatly. Moving the conversation to SLA/SLO's considers the product as a whole and gives us meaningful data and targets.

Logging Questions

SNo.	Issues/Inputs Required	Details	Comments
1	Use Single FluentBit for application logs and cos audit logs	We have seen in the Google documentation that when enabling logging of COS Audit logs in GKE, a separate FluentBit daemonset is spun up, which has to be self-managed. Is this able to be merged into the managed FluentBit?	Unfortunately, at this time, this dedicated Fluent Bit instance cannot be merged into the managed Fluent Bit setup that GKE provides for standard logging.
2	FluentBit throughput	Currently the FluentBit available can only support 10MB/s. This may not be enough for the XCS use case. Google mentioned they are working on a 20MB/s version. Is there an ETA for this and can we get early access?	logging V3 agent can support upto 20MB/s. We are discussing the timelines for availability and early access is possible

3	Google Provided OTEL Collector	For GKE logging, it was suggested we could use a Google provided (not-managed) OTEL collector instead of the current GKE managed FluentBit. This is because there is work to release a Google Managed OTEL solution for logs going forwards. If we are to configure this Google Provided OTEL, how do we get access to it, and would it be a daemonset configuration like the FluentBit setup?	The Google-built OpenTelemetry Collector should be available in March. You will be able to deploy it in the same way as the current OSS OTel collector: as DaemonSet or Sidecar. It is up to you to select the native Fluentbit or an OTel collector.

Metrics Questions

SNo.	Issues/Inputs Required	Details	Comments
1	GMP use of K8s Secrets	We have noticed that if we wish to use Google Managed Prometheus to scrape a workload with TLSConfig or behind an authentication layer, the PodMonitoring resource only supports supplying the certs/keys/credentials via K8s Secrets. K8s secrets are an insecure way of handling sensitive data, is there another framework that can be implemented using the managed Prometheus? Traditionally at Citi, we use the Vault Agent Injector to mount secrets on the pods via annotations. Can something similar be implemented?	Not possible currently. Secret Provider API to Prometheus feature is in works which will enable Prometheus developers the ability to add additional secret providers, including vault integration. **H.W Are there any timelines on this?** This has been entered by the PM as a Feature Request but has not been prioritized as of yet so no timeline as of now.
4	OTEL Sidecar Pattern	What is the rationale for recommending OTEL sidecars to write metric data to Google Managed Prometheus first, rather than having the OTEL sidecars write directly to Cloud Monitoring?	The regular Cloud Monitoring API is considered legacy and significantly more expensive than GMP
5	Custom Storage Location	Does the Cloud Monitoring service support the use of storing metric data in a custom backend? This would enable our teams greater flexibility to encrypt and store data as per Citi policy	Metrics data must be stored in a project. The Cloud Monitoring service will then store the data in Google's proprietary time-series database Monarch.

6	Disable alertmanager/rule-evaluator deployments	We do not wish to leverage the alertmanager and rule-evaluator pods as part of the GMP solution. Is there any way to disable the alertmanager statefulset creation and rule-evaluator deployment creation when enabling Google Managed Prometheus?	They are disabled by default, they will be enabled only if a rule, clusterRule or GobalRule resource is deployed. ref **H.W Currently the deployments are scaled to zero when no rules etc. are deployed. But can it be supported to disable the creation of the deployments altogether?** You can deploy without alertmanager, but rule-evaluator will need to be there even if unused.
7	HTTPS metrics endpoints	Is it possible for any Google/system workloads in GKE to have their metrics endpoints exposed over HTTPS. Citi policy dictates that no connections are authorized to use HTTP	The GKE metric endpoints are all exposed using HTTPS.
8	Cloud Monitoring Dashboard Features	Is it possible to enable the following two features when it comes to the filters on the Cloud Monitoring dashboards: ● Ability to create dependent filters ○ We would like a similar feature as on Grafana whereby the values presented by one filter/variable are dynamic depending on the value of another filter/variable. Say I set a project filter to projectA, can my cluster filter then only present options that are clusters within projectA, rather than still showing clusters from all projects? ○ Can a filter value support regex? Say I want to filter for all clusters containing the string 'mycluster', is this possible?	filter value can support regex as shown in this example

Tracing Questions

SNo.	Issues/Inputs Required	Details	Comments

2	Sidecar OTEL Collector	Is the recommended approach for collecting trace data in GKE, to have a sidecar otel collector on each application pod, or to have a central otel collector?	The same Sidecar can be used for logs, metrics, and traces: https://cloud.google.com/stackdriver/docs/instrumentation/overview
3	Custom Storage Location	Does the Cloud Trace service support the use of storing trace data in a custom backend? This would enable our teams greater flexibility to encrypt and store data as per Citi policy	Trace data must be stored in Cloud Trace in a Google Cloud project.
4	Configuration Recommendations	Are there any recommendations for configuration values to set for the tracing solution, to ensure robustness/resiliency when dealing with extremely high volumes	Cloud Trace can handle high volumes.

Metrics Availability In Managed Exporters

H.W Is there a response from Google on whether the below metrics could be made available on managed KSM and cadvisor/kubelet scraping

Our SRE team would want the below metrics to be exposed by the managed kube-state-metrics and cadvisor/kubelet scraping. Let us know if the managed services can offer these.

Kube State Metrics

The listed metrics from your Kube State Metrics list are NOT directly exposed by kube-state-metrics for scraping:

- kube_deployment_status_replicas_unavailable:
  - kube-state-metrics doesn't provide this directly. You calculate it by subtracting kube_deployment_status_replicas_ready from kube_deployment_status_replicas_desired (which is provided).
- kube_daemonset_status_number_unavailable:
  - Similar to deployments, you'd calculate this using metrics like kube_daemonset_status_desired_number_scheduled and kube_daemonset_status_number_ready.
- kube_cronjob_next_schedule_time:
  - kube-state-metrics focuses on the current state of objects, not future schedules.

- kube_endpoint_address_available:
    - kube-state-metrics provides kube_endpoint_address_condition, which you need to process to determine availability.
- kube_endpoint_address_not_ready:
    - Again, derived from kube_endpoint_address_condition.
- kube_horizontalpodautoscaler_status_target_metric:
    - kube-state-metrics provides related metrics like kube_horizontalpodautoscaler_status_current_metric_value, but not the "target" in that exact form.
- kube_pod_completion_time:
    - Completion time is an event-driven piece of data, not a continuous state metric. kube-state-metrics focuses on state.
- kube_job_status_completion_time:
    - Similar to pods, completion time is event-driven.
- kube_ingress_created:
    - Creation timestamps are available via the Kubernetes API, but kube-state-metrics doesn't expose a dedicated metric for "created".
- kube_networkpolicy_created:
    - Same as Ingress - creation timestamps are API data, not a direct kube-state-metrics metric.
- kube_pod_start_time:
    - Pod start time is typically associated with events and is not a state metric.
- kube_resourcequota:
    - kube-state-metrics provides information about resource quotas, but you'll often need to filter or process the data to get specific resource usage.
- kube_volumeattachment_spec_source_persistentvolume:
    - This is more about the configuration of the volume attachment, not a runtime state metric.

Cadvisor/Kubelet

The Cadvisor/Kubelet list are generally NOT exposed as metrics you would scrape:

- It's important to understand that all of the container_* and kubelet_* metrics you provided are designed to be exposed for scraping by Prometheus (and thus are available in Cloud Monitoring).
    - cAdvisor and the Kubelet export these metrics in Prometheus format.
    - Prometheus is designed to scrape these endpoints.

Why These Are Different

- kube-state-metrics focuses on the state of Kubernetes objects as they exist in the API server (e.g., deployments, pods, nodes).
- cAdvisor and the Kubelet provide performance and resource usage metrics (e.g., container CPU, memory, Kubelet operations).

Key Takeaway

- All the cAdvisor/Kubelet metrics are designed for scraping.
- kube-state-metrics provides a subset of Kubernetes information, and some metrics need to be calculated or derived from its output. Other information, especially time-based or configuration-focused details, are not provided.

Kube State Metrics

kube_deployment_status_replicas_ready
kube_deployment_status_replicas_unavailable
kube_deployment_status_condition

kube_daemonset_status_number_unavailable
kube_daemonset_status_observed_generation

kube_cronjob_status_active
kube_cronjob_next_schedule_time
kube_cronjob_status_last_successful_time
kube_endpoint_address_available
kube_endpoint_address_not_ready
kube_horizontalpodautoscaler_status_target_metric
kube_node_spec_unschedulable
kube_node_status_allocatable
kube_node_status_capacity
kube_node_status_condition
kube_pod_completion_time
kube_job_status_completion_time
kube_job_status_failed
kube_job_status_active
kube_ingress_created
kube_networkpolicy_created
kube_persistentvolume_status_phase
kube_persistentvolumeclaim_status_phase
kube_namespace_status_phase

kube_pod_container_status_restarts_total
kube_pod_container_status_running
kube_pod_container_status_terminated
kube_pod_container_status_terminated_reason
kube_pod_container_status_waiting_reason
kube_pod_created
kube_pod_deletion_timestamp
kube_pod_start_time
kube_pod_status_ready
kube_pod_status_reason
kube_pod_status_scheduled
kube_pod_container_resource_limits
kube_pod_container_resource_requests
kube_poddisruptionbudget_status_current_healthy
kube_poddisruptionbudget_status_desired_healthy
kube_replicaset_status_ready_replicas
kube_replicaset_status_replicas
kube_statefulset_status_replicas_available
kube_statefulset_status_replicas
kube_replicaset_metadata_generation
kube_resourcequota
kube_service_status_load_balancer_ingress
kube_volumeattachment_spec_source_persistentvolume
kube_volumeattachment_status_attached

All of the Cadvisor/Kubelets list below are able to be scraped by Prometheus

Cadvisor/Kubelet

container_cpu_cfs_throttled_seconds_total
container_cpu_load_average_10s
container_cpu_system_seconds_total
container_cpu_usage_seconds_total
container_cpu_user_seconds_total
container_memory_usage_bytes
container_memory_max_usage_bytes
container_memory_failcnt
container_memory_working_set_bytes
container_memory_rss
container_memory_cache

container_memory_swap
container_fs_usage_bytes
container_fs_limit_bytes
container_fs_io_time_seconds_total
container_fs_reads_bytes_total
container_fs_writes_bytes_total
container_network_receive_bytes_total
container_network_transmit_bytes_total
container_network_receive_errors_total
container_network_transmit_errors_total
container_network_receive_packets_total
container_network_transmit_packets_total
container_last_seen
container_oom_events_total
container_scrape_error
kubelet_graceful_shutdown_end_time_seconds
kubelet_graceful_shutdown_start_time_seconds
kubelet_desired_pods
kubelet_http_requests_total
container_processes
container_scrape_error
container_spec_cpu_quota
container_spec_cpu_shares
container_spec_memory_limit_bytes
container_spec_memory_reservation_limit_bytes
container_spec_memory_swap_limit_bytes
container_start_time_seconds
container_threads
kubelet_active_pods
kubelet_desired_pods
kubelet_http_requests_total
kubelet_http_requests_duration_seconds_sum
kubelet_image_pull_duration_seconds_count
kubelet_pod_start_duration_seconds
kubelet_pod_status_sync_duration_seconds
kubelet_restarted_pods_total
kubelet_started_containers_total
kubelet_started_containers_errors_total
kubelet_started_pods_total
kubelet_started_pods_errors_total

kubelet_runtime_operations_duration_seconds
kubelet_runtime_operations_errors_total
kubelet_runtime_operations_total
kubelet_volume_stats_available_bytes
kubelet_volume_stats_capacity_bytes
kubelet_volume_stats_used_bytes