Container Insights Multi-tenant Managed Logging (Public Preview)

Introduction

This doc discusses the multi-tenant logging feature in Azure Monitor - Container Insights.

This feature supports the below scenarios …

1. Multi-tenancy: Allows the routing of container console (stdout & stderr) logs from one or more Kubernetes namespaces to their respective Azure Log Analytics Workspace destination.
2. Multi-homing: Same set of Logs from one or more Kubernetes namespaces to multiple Azure Log Analytics workspaces.

Multi-tenancy capability is a common request from customers who operate shared cluster platforms utilizing AKS. These customers need the ability to configure log collection in a way that segregates log by different teams so that each team has access to the logs of the containers running in k8s namespaces for which they own and ability to get the billing and management associated to the Azure Log analytics workspace.

For instance, logs from infrastructure k8s namespaces such as kube-system, can be directed to a specific Azure Log Analytics workspace for the Infrastructure team, while each application team's logs can be sent to their respective Azure Log Analytics workspaces.

This feature supports in 2 modes – Basic and Advanced mode.

Basic mode supports up to 50K logs/sec per node and whereas Advanced mode supports up to 100k logs/sec per node.

Advanced mode has dependency on the ama-logs k8s service, k8s deployment and ama-logs HPA.

Note: Multi-tenant Managed Logging is currently in Private preview. This Private preview documentation is not for publication. To be shared with internal users and NDA customers only.

How does it work?

Container Insights supports the default ContainerInsights Extension DCR, which is a singleton DCR.

As part of this feature, Container Insights has introduced support for the ***ContainerLogV2Extension*** DCR, primarily for container logs, with Kubernetes namespaces as data collection settings and transform KQL filter for advance filtering in the Azure Log Analytics Ingestion Pipeline. Multiple ContainerLogV2Extension DCRs can be created with different data

collection settings, ingestion transformation KQL filters and destination, and associated all these DCRs to same AKS cluster.

When the multi-tenancy feature is enabled through a config map, the Container Insights agent periodically fetches both the default ContainerInsights Extension DCR and the ContainerLogV2Extension DCR (during container startup and every 5 minutes). All configured streams in the ContainerInsights Extension DCR will be ingested into the Azure Log Analytics workspace destination defined in this default DCR as usual, except for container logs.

The routing and ingestion of container logs operate as follows:

- The Container Insights agent builds the Kubernetes namespaces to DCR map using the periodically (every 5 minutes) fetched ContainerLogV2Extension DCRs. This map gets periodically updated as new DCRs added or existing DCRs removed to keep the state of the DCRs.
- For each log entry:
    - If the Kubernetes namespace of the log entry is in the Kubernetes namespace DCR map, then this log entry will be ingested into all the Azure Log Analytics Workspace destinations specified in the corresponding DCRs. If the DCR has ingestion transformation KQL filter, then KQL filter is applied in the Azure Log Analytics ingestion pipeline.
    - If the k8s namespace is not in the Kubernetes namespace to DCR map, the log entry will be ingested into the Azure Log Analytics workspace defined in the default ContainerInsights Extension DCR which is default behavior, and this behavior can be disabled through below setting

```
[log_collection_settings]
  #  This feature in private preview and not ready for production use,please reach out us to try out this feature.
  #  [log_collection_settings.multi_tenancy]
  #    # High log scale MUST be enabled to use this feature. Refer to https://aka.ms/cihsmode for more details on high log scale mode
  #    enabled = true
  #    disable_fallback_ingestion = false # by default fallback ingestion to default container insights extension DCR is enabled
  ###   Below advanced settings MUST be provided when ama-logs-multitenancy service deployed through AKS Monitoring Addon
  ##    advanced_mode_enabled = false # Default value is false
  ##    namespaces = ["app-team-1","app-team-2"]
  ##    namespace_settings = ["app-team-1:throttle_rate=2000","app-team-2:throttle_rate=5000"]
```

    - If the Kubernetes Namespace DCR map has the ***_ALL_K8S_NAMESPACES_*** then all log entries from across all the namespaces will be ingested to the Azure Log Analytics Workspace destinations specified in the corresponding DCRs. This is multi-homing behavior.

Pre-requisites

1. An Azure CLI version of 2.63.0 or higher

2. The AKS-preview CLI extension version must be 7.0.0b4 or higher if an AKS-preview CLI extension is installed.
3. The Container Log schema version MUST be v2, i.e., ContainerLogV2 and must be enabled in High log scale mode
4. Your cluster meets the Firewall requirements

Migration steps

If Container Insights is already enabled on your cluster, then simply disable the monitoring

```
az aks disable-addons -a monitoring -g <clusterRGName> -n <clusterName>
```

And then proceed onto the onboarding steps.

If you prefer to use the ARM template disable monitoring, you can refer to this article https://learn.microsoft.com/en-us/azure/azure-monitor/containers/kubernetes-monitoring-disable#aks-cluster

Onboarding Steps

1. Download ConfigMap and add below settings to downloaded ConfigMap

1a. Add below setting under **agent-settings: |-**

```
[agent_settings.high_log_scale]
  enabled = true
```

1b. Add below setting under [log_collection_settings.multi_tenancy] Under **log-data-collection-settings: |-**

Basic mode

ConfigMap setting	*Default value*	*Description*
enabled	false	Boolean flag to enable or disable the feature
disable_fallback_ingestion	true	If there is no ContainerLogV2 extension DCR corresponding to log of the k8s namespaces, then those will be ingested to destination configured default ContainerInsights Extension DCR and configuring this true will disable this default behavior.
advanced_mode_enabled	false	If the scale of the logs per node is more than 50k logs/sec, enable this flag. This requires ama-logs-multi-tenancy deployment, reach out through support ticket to enable this in backend.
namespaces	“”	Specify the array of namespaces for which advanced mode MUST be enabled. In this configuration, for each namespace there will be separate fluent-bit plugins with configuration such as throttle limit and buffer config etc.
namespace_settings	Defaults used across all the namespace	Custom settings required for specific namespaces, configure through this.

Advanced Mode

Note > Advanced Mode requires the ama-logs service deployment, please reach out us if you are planning to enable Advanced Mode.

ConfigMap setting	*Default value*	*Description*
enabled	false	Boolean flag to enable or disable the feature
disable_fallback_ingestion	true	If there is no ContainerLogV2 extension DCR corresponding to log of the namespaces will be ingested to destination configured default ContainerInsights Extension DCR
advanced_mode_enabled	true	If the scale of the logs per node is more than 50k logs/sec, enable this flag. This requires ama-logs-multi-tenancy deployment, reach out through support ticket to enable this in backend.
namespaces	[“”]	The array of namespaces for which advanced_mode MUST be enabled. In this configuration, for each namespace there will be separate fluent-bit plugins with configuration such as throttle limit and buffer config etc.
namespace_settings	Defaults used across all the namespace	Custom settings required for specific namespaces, configure through this.

The above settings instruct the container insights ama-logs deamonset pods to run in multi-tenancy mode.

2. Enable Monitoring Add-on either through Azure CLI or ARM Template.

Monitoring Addon enablement with Azure CLI

Scenario	**Azure CLI Command**

Existing AKS Cluster	`az aks enable-addons -a monitoring \`
	`-g <clusterRGName> \`
	`-n <clusterName> \`
	`--enable-high-log-scale-mode`

New AKS cluster	`az aks create \`
	`-g <clusterRGName> \`
	`-n <clusterName> \`
	`enable-addons -a monitoring \`
	`--enable-high-log-scale-mode`

Existing Private AKS Cluster	`az aks enable-addons -a monitoring \`
	` -g <clusterRGName> -n <clusterName> \`
	` --enable-high-log-scale-mode`
	` --ampls-resource-id <Azure Monitor Private Link Resource Id>`

For the new Private AKS cluster	Refer to this documentation https://learn.microsoft.com/en-us/azure/aks/private-clusters?tabs=azure-portal for creating AKS Private cluster and use these additional

	parameters `--enable-high-scale-mode` and `--ampls-resource-id` to configure high log scale mode with Azure Monitor Private Link Scope Resource ID.

Azure Monitor Private Link Resource Id format would be /subscriptions/<subscriptionId>/resourceGroups/<resourceGroupName>/providers/microsoft.insights/privatelinkscopes/<resourceName>

Monitoring Addon enablement with ARM Templates

Download ARM Template and Parameter file

```
curl -LO  https://aka.ms/aks-enable-monitoring-msi-onboarding-template-file
curl -LO  https://aka.ms/aks-enable-monitoring-msi-onboarding-template-parameter-file
```

Configure the Parameter file, aks-enable-monitoring-msi-onboarding-template-parameter-file

Parameter Name	Description
`aksResourceId`	Azure Resource ID of the AKS cluster
`aksResourceLocation`	Azure Region of the AKS cluster
`workspaceResourceId`	Azure Resource ID of the Azure Log Analytics Workspace
`workspaceRegion`	Azure Region of the Azure Log Analytics Workspace
`enableContainerLogV2`	Flag to indicate whether to use ContainerLogV2 or not. This MUST be true
`enableSyslog`	Flag to indicate to enable Syslog collection or not.
`syslogLevels`	Log levels for Syslog collection
`syslogFacilities`	Facilities for Syslog collection
`resourceTagValues`	Azure Resource Tags to use on AKS, Azure Monitor Data Collection Rule, and Azure Monitor Data collection endpoint etc.
`dataCollectionInterval`	Data collection interval for applicable inventory and perf data collection. Default is 1m
`namespaceFilteringModeForDataCollection`	Data collection namespace filtering mode for applicable inventory and perf data collection. Default is off
`namespacesForDataCollection`	Namespaces for data collection for applicable for inventory and perf data collection.
`streams`	Streams for data collection. For high scale mode, instead of Microsoft-ContainerLogV2, use Microsoft-ContainerLogV2-HighScale. Please note, having both Microsoft-ContainerLogV2 and Microsoft-ContainerLogV2-HighScale will cause duplicate logs.

useAzureMonitorPrivateLinkScope	Flag to indicate whether to configure Azure Monitor Private Link Scope or not.
azureMonitorPrivateLinkScopeResourceId	Azure Resource ID of the Azure Monitor Private Link Scope.

Deploy the ARM template

```
az deployment group create \
--name AzureMonitorDeployment \
--resource-group <aksClusterResourceGroup> \
--template-file  aks-enable-monitoring-msi-onboarding-template-file \
--parameters aks-enable-monitoring-msi-onboarding-template-parameter-file
```

3. For ***each app or infra team, onboard*** the ContainerLogV2 extension DCR with list of k8s namespaces under Data Collection settings and Azure Log Analytics workspace as destination.

   Download ARM Template and Parameter file

   curl -LO https://aka.ms/aks-enable-monitoring-multitenancy-onboarding-template-file
   curl -LO https://aka.ms/aks-enable-monitoring-multitenancy-onboarding-template-parameter-file

Configure the Parameter file, aks-enable-monitoring-multitenancy-onboarding-template-parameter-file

Parameter Name	Description
aksResourceId	Azure Resource ID of the AKS cluster
aksResourceLocation	Azure Region of the AKS cluster
workspaceResourceId	Azure Resource ID of the Azure Log Analytics Workspace
workspaceRegion	Azure Region of the Azure Log Analytics Workspace
k8sNamespaces	List of k8s namespaces for which logs need to be ingested to the Azure Log Analytics Workspace defined in this parameter file
resourceTagValues	Azure Resource Tags to attach on Azure Monitor Data Collection Rule and Azure Monitor Data Collection Endpoint resources
transformKql	KQL filter for advance filtering using Ingestion transformation. For example, to exclude the logs for specific pod, the KQL would be "***source | where PodName != '<podName>'***" which will be applied in Azure Log Analytics Pipeline and this drops all the logs for specified pod.

	Refer to https://learn.microsoft.com/en-us/azure/azure-monitor/essentials/data-collection-transformations for additional details
`useAzureMonitorPrivateLinkScope`	Flag to indicate whether to configure Azure Monitor Private Link Scope or not.
`azureMonitorPrivateLinkScopeResourceId`	Azure Resource ID of the Azure Monitor Private Link Scope.

Deploy the ARM template

```
az deployment group create \
--name AzureMonitorDeployment \
--resource-group <aksClusterResourceGroup> \
--template-file  aks-enable-monitoring-multitenancy-onboarding-template-file \
--parameters aks-enable-monitoring-multitenancy-onboarding-template-parameter-file
```

QoS Grafana Dashboards

Pre-requisites

- Enable Azure Managed Prometheus by following the instructions https://learn.microsoft.com/en-us/azure/azure-monitor/containers/kubernetes-monitoring-enable?tabs=cli#enable-prometheus-and-grafana

Basic Mode

1. Download the ama-metrics-prometheus-config-node ConfigMap

```
curl -LO https://raw.githubusercontent.com/microsoft/Docker-Provider/refs/heads/ci_prod/Documentation/MultiTenancyLogging/BasicMode/ama-metrics-prometheus-config-node.yaml
```

2. Check if you already have an existing ama-metrics-prometheus-config-node ConfigMap via

```
kubectl get cm -n kube-system | grep ama-metrics-prometheus-config-node
```

If there is an existing ConfigMap, then you can add the ama-logs-daemonset scrape job to the existing ConfigMap else you can apply this ConfigMap through

```
kubectl apply -f ama-metrics-prometheus-config-node.yaml
```

3. Import Grafana dashboard JSON file to the Azure Managed Grafana Instance - https://raw.githubusercontent.com/microsoft/Docker-Provider/refs/heads/ci_prod/Documentation/MultiTenancyLogging/BasicMode/AzureMonitorContainers_BasicMode_Grafana.json

Advanced Mode

1. Download the ama-metrics-prometheus-config-node ConfigMap

```
curl -LO https://raw.githubusercontent.com/microsoft/Docker-Provider/refs/heads/ci_prod/Documentation/MultiTenancyLogging/BasicMode/ama-metrics-prometheus-config-node.yaml
```

2. Check if you already have an existing ama-metrics-prometheus-config-node ConfigMap via

```
kubectl get cm -n kube-system | grep ama-metrics-prometheus-config-node
```

If there is an existing ConfigMap, then you can add the ama-logs-daemonset scrape job to the existing ConfigMap

Else you can apply this ConfigMap through

```
kubectl apply -f ama-metrics-prometheus-config-node.yaml
```

3. Downlaod the ama-metrics-prometheus-config ConfigMap
   ```
   curl -LO https://raw.githubusercontent.com/microsoft/Docker-Provider/refs/heads/ci_prod/Documentation/MultiTenancyLogging/AdvancedMode/ama-metrics-prometheus-config.yaml
   ```
4. Check if you already have an existing ama-metrics-prometheus-config ConfigMap via

```
kubectl get cm -n kube-system | grep ama-metrics-prometheus-config
```

If there is an existing ConfigMap, then you can add the ama-logs-multitenancy scrape job to the existing ama-metrics-prometheus-config ConfigMap

Else you can apply this ConfigMap through

```
kubectl apply –f ama-metrics-prometheus-config.yaml
```

5. Import Grafana dashboard JSON file to the Azure Managed Grafana Instance - https://raw.githubusercontent.com/microsoft/Docker-Provider/refs/heads/ci_prod/Documentation/MultiTenancyLogging/AdvancedMode/AzureMonitorContainers_AdvancedMode_Grafana.json

Offboarding Steps

To disable this feature, follow these steps:

1. Remove the ***ContainerLogV2 Extension*** DCR association

   1a. list all the DCR associations

```
az monitor data-collection rule association list-by-resource --resource /subscriptions/<subId>/resourcegroups/<rgName>/providers/Microsoft.ContainerService/managedClusters/<clusterName>
```

1b. Delete all DCR-A's for ContainerLogV2 extension

```
az monitor data-collection rule association delete --association-name
<ContainerLogV2Extension DCR-A" --resource
/subscriptions/<subId>/resourcegroups/<rgName>/providers/Microsoft.ContainerService/m
anagedClusters/<clusterName>
```

2. Update from enabled = true to enabled = false under [log_collection_settings.multi_tenancy] in container-azm-ms-agentconfig

```
    kubectl edit cm container-azm-ms-agentconfig -n kube-system -o yaml
```

3. If you want to disable container insights completely, then you can disable through

```
az aks disable-addons -a monitoring -g  <clusterResourceGroup> -n <clusterName>
```

Network / Firewall Requirements

After configuring the network firewall requirements for monitoring a Kubernetes cluster, additional configurations are needed for this feature.

The following table lists the additional proxy and firewall configuration information:

Azure Public Cloud

Endpoint	Purpose	Port
Obtain the Logs Ingestion endpoint from Azure Monitor Data Collection Endpoint resource. The Resource name should be MSCI-<region>-<clusterName>. Endpoint should be in this format - <data-collection-endpoint>-<suffix>.<cluster-region-name>-<suffix>. .ingest.monitor.azure.com	Azure Monitor for containers - logs ingestion endpoint (DCE). Default Data Collection endpoint name is MSCI-<region>-<clusterName>	443

Microsoft Azure operated by 21Vianet cloud

Endpoint	Purpose	Port
Obtain the Logs Ingestion endpoint from Azure Monitor Data Collection Endpoint resource. The Resource name	Azure Monitor for containers - logs ingestion endpoint (DCE).	443

should be MSCI-<region>-<clusterName>. Endpoint should be in this format - <data-collection-endpoint>-<suffix>.<cluster-region-name>-<suffix>.ingest.monitor.azure.cn	Default Data Collection endpoint name is MSCI-<region>-<clusterName>	

Azure Government cloud

Endpoint	Purpose	Port
Obtain the Logs Ingestion endpoint from Azure Monitor Data Collection Endpoint resource. The Resource name should be MSCI-<region>-<clusterName>. Endpoint should be in this format - <data-collection-endpoint>-<suffix>.<cluster-region-name>-<suffix>.ingest.monitor.azure.us	Azure Monitor for containers - logs ingestion endpoint (DCE). Default Data Collection endpoint name is MSCI-<region>-<clusterName>	443

Known issues

Limitations

The following scenarios are not supported during the Preview release, and we have plans to address as part of the GA:

1. More than 30 ContainerLogV2 Extension DCR associations supported per cluster.
2. AKS Clusters with ARM64 nodes
3. Azure Arc Kubernetes
4. HTTP proxy with trusted cert
5. Onboarding through Azure Portal, Azure Policy, Terraform and Bicep
6. Configuring through Monitor Settings from AKS Insights
7. Automatic migration from Existing Container Insights Onboarding

Support

Azure Support is not available for private preview features. For queries and feedback, please reach out Suvarna Saraswat susaraswat@microsoft.com and/or Container Insights - Logs cilogs@microsoft.com.

Legal & Compliance

See the Supplemental Terms of Use for Microsoft Azure Previews for legal terms that apply to Azure features that are in beta, preview, or otherwise not yet released into general availability.