Scraping FlashBlade Metrics Using a Prometheus Exporter

Prometheus is an open-source time-series database used for monitoring and alerting. The data is scraped in regular intervals from endpoints using client libraries called exporters that expose the metrics in a Prometheus format. A Prometheus server then collects those metrics via HTTP requests and saves them with timestamps in a database.

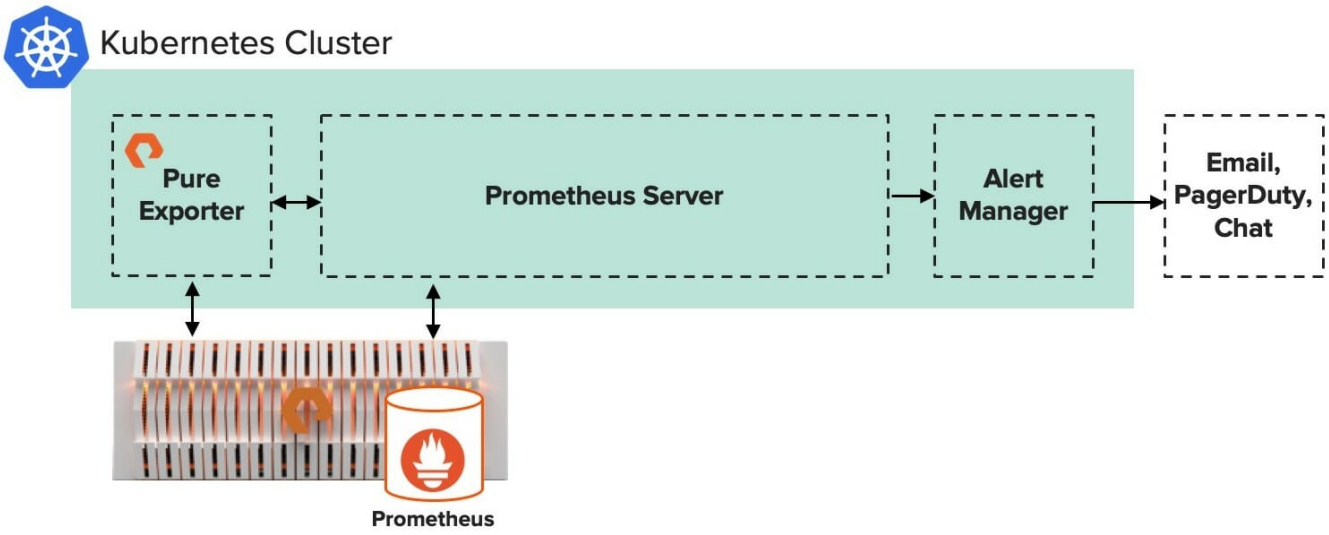
Prometheus can be used to collect metrics about the CPU, memory, and network utilization of an AI cluster. The short instructions for adding Prometheus to a Rancher-based Kubernetes cluster can be found here. By default, that Prometheus application will collect metrics from the exporters native to the cluster – such as usage stats for all the Kubernetes pods.

However, you might have additional infrastructure in your AI cluster beyond the compute servers that the Kubernetes pods are running on. For example, you might have shared storage that needs to be monitored as well.

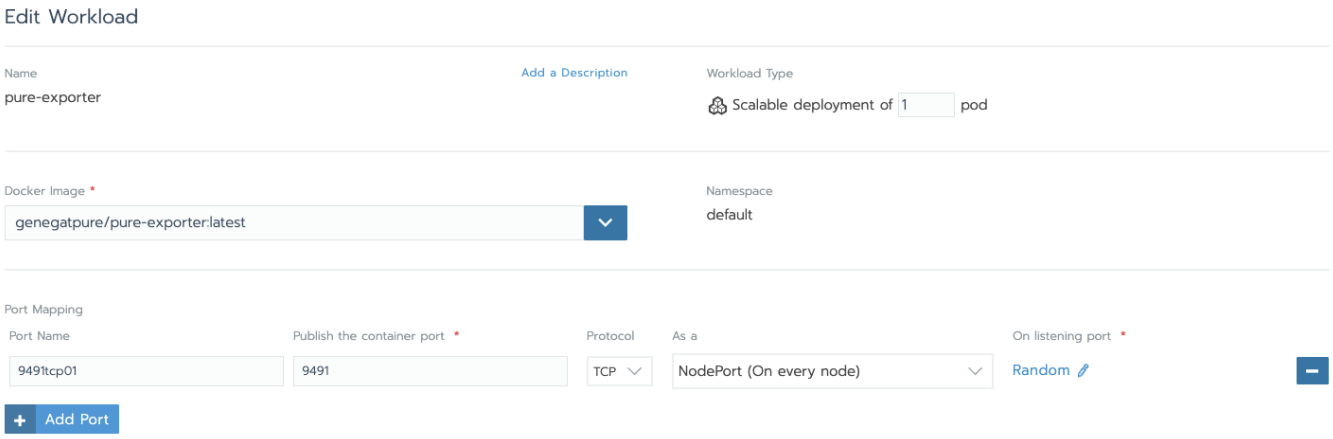
Training and inference pipelines often include a variety of applications like Kafka for message queuing and JupyterHub for exploring models. Having a shared data platform like Pure Storage FlashBlade™ is beneficial for these pipelines because it enables:

- easy access to shared datasets
- seamless collaboration among data scientists
- a cost-effective way to scale capacity and performance as needed

In this blog post, we'll cover the steps to use our Pure Exporter for FlashBlade so you can have a single pane of glass for monitoring your entire AI cluster.



As shown in the diagram above, the Pure Exporter runs in a Kubernetes pod. Kick off a workload with the Pure Exporter docker image, which can be downloaded here.



Once the Pure Exporter is running as a pod, note the cluster-IP and port values for the pure-exporter. You can find them by running “kubectl get svc”:

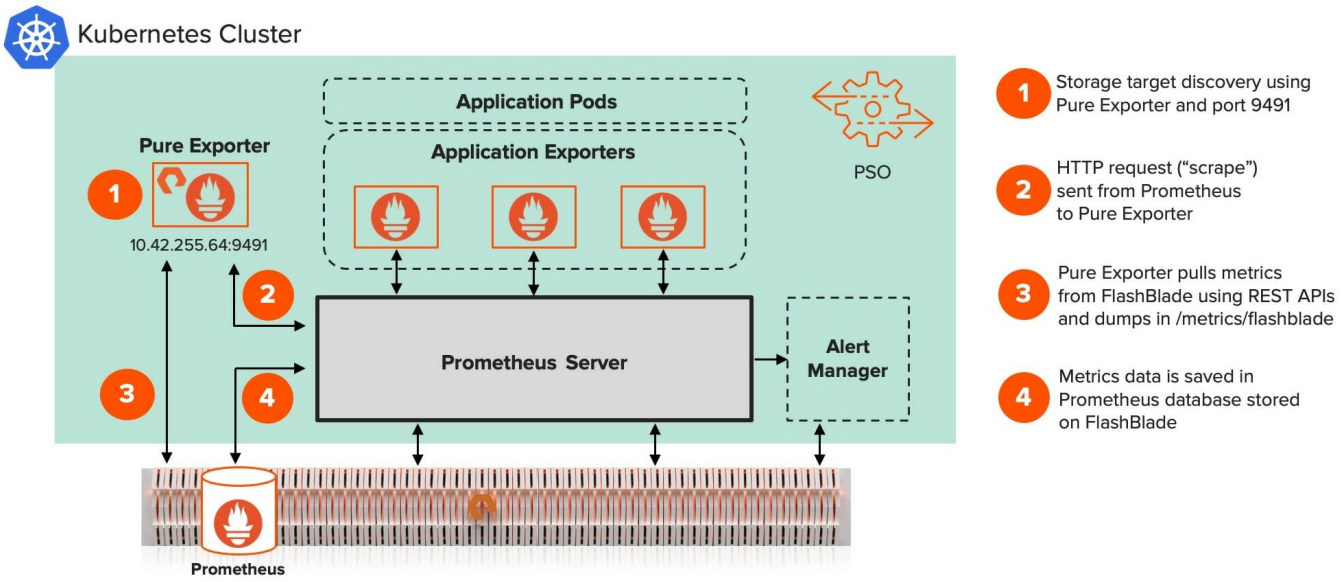
[crayon-681e52153a7ce402076559/]

Next, edit the Prometheus server configmap to include a new job to scrape the FlashBlade metrics.

[crayon-681e52153a7db436406448/]

FlashBlade will be now listed as a target in the Prometheus server and ready to scrape. Prometheus now knows where to find the “/metrics/flashblade” endpoint on the Pure Exporter pod.

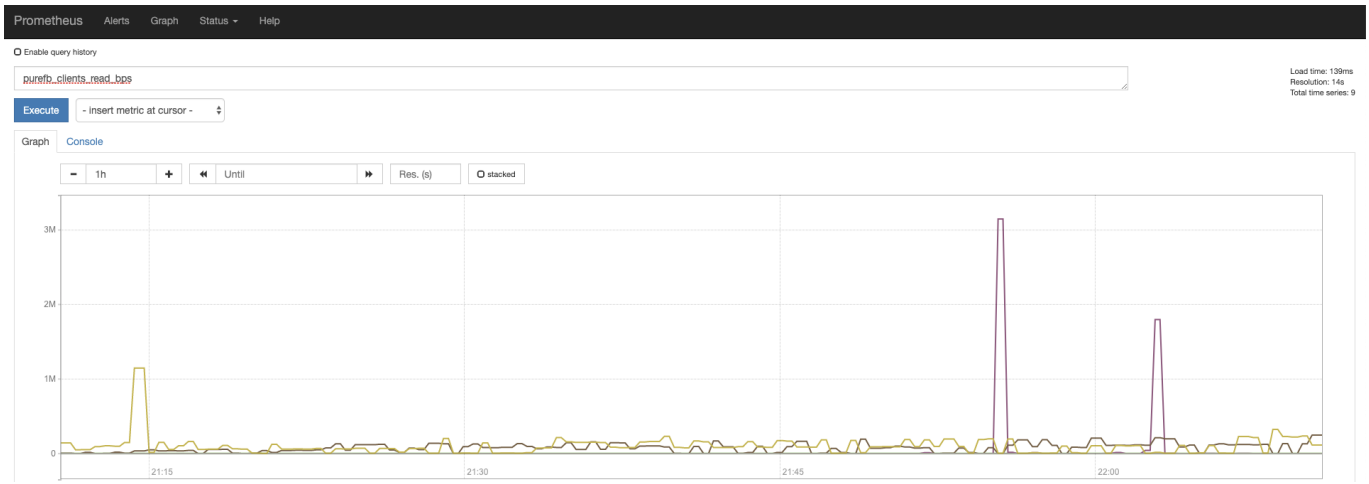
So, now the overall flow looks like this:



The Prometheus server discovers the FlashBlade through the Pure Exporter and port 9491 in the Kubernetes cluster. The Pure Exporter gathers the FlashBlade metrics data using RESTful APIs . The collected data is then stored in the Prometheus database.

(In fact, that database is stored in the FlashBlade itself. Storing the monitoring data on FlashBlade provides the ability to scale as AI workloads increase. FlashBlade also provides data reduction of ~2:1 for Prometheus database!)

Once the metrics are stored in the Prometheus database, it is easy to query.



Here, we’re visualizing a metric directly from Prometheus, but it’s easy to pull the

data forward into Grafana for beautiful, easy-to-create dashboards. We'll review Grafana integration into the AI Data Hub in our next post in this series.

To summarize, Pure Exporter allows Prometheus to scrape metrics from Pure Storage data platforms using RESTful APIs.

- No additional scripting needed to monitor storage along with the rest of the cluster.
- FlashBlade is a great place to save a cluster's Prometheus database since it provides the simplicity of centralized logs and performant scalability.

The next post will demonstrate how to configure and integrate a FlashBlade metrics dashboard into Grafana for the AI cluster. Visit the blog for our next posts in the series over the coming weeks:

- Automating an inference pipeline in a Kubernetes Cluster
- Tuning networking configuration of a Kubernetes-based AI Data Hub
- Integrating Pure RapidFile Toolkit into Jupyter notebooks

Updated Information

1. Transition to OpenMetrics Exporter:

- **Deprecation of Pure Exporter:** The previously utilized Pure Exporter has been deprecated in favor of the Pure Storage OpenMetrics Exporter. This new exporter offers enhanced performance, improved security, and better integration with modern monitoring stacks. Users are encouraged to transition to the OpenMetrics Exporter to take advantage of these improvements. github.com
- **Key Advantages:**

- **Standardization:** Aligns with the OpenMetrics standard, ensuring compatibility with a wide range of observability tools.
- **Enhanced Security:** Improved authentication mechanisms, including support for bearer tokens and configuration files for API tokens.
- **Modular Design:** Provides specific endpoints for different metrics categories, allowing for more granular monitoring.

2. Updated Deployment and Configuration Steps:

- **Deployment:** The OpenMetrics Exporter can be deployed using Docker. Build the Docker image from the provided Dockerfile:bashCopyEdit`docker build -t pure-fb-ome:<VERSION> .` Replace `<VERSION>` with the desired version tag.
- **Configuration:** Authentication is managed through a configuration file specifying the FlashBlade address and API token:yamlCopyEdit`array_id_1: address: <ip-address-or-hostname-1> api_token: <api-token-1> array_id_2: address: <ip-address-or-hostname-2> api_token: <api-token-2>` This file is passed to the exporter at runtime, enhancing security by avoiding hard-coded credentials.
- **Prometheus Integration:** Configure Prometheus to scrape metrics from the OpenMetrics Exporter by adding a new job in the `prometheus.yml` configuration file:yamlCopyEdit`- job_name: 'pure_flashblade' scrape_interval: 30s metrics_path: /metrics/flashblade static_configs: - targets: ['<exporter_host>:<port>']` Ensure that `<exporter_host>` and `<port>` correspond to the OpenMetrics Exporter's network settings.

3. Enhanced Visualization with Grafana:

- **Grafana Dashboards:** Leverage Grafana's capabilities to create comprehensive dashboards for visualizing FlashBlade metrics. The OpenMetrics Exporter provides detailed metrics that can be used to monitor performance, capacity, and other critical parameters.

Conclusion

Transitioning to the Pure Storage OpenMetrics Exporter offers improved performance, security, and compatibility with modern observability tools. By updating deployment and configuration practices, organizations can achieve more effective monitoring of FlashBlade systems within their Prometheus and Grafana environments.

See the previous posts in the series, Providing Data Science Environments with Kubernetes and FlashBlade and Storing a Private Docker Registry on FlashBlade S3.