## Integration

- Instructions: https://apps.sematext.com/ui/howto/Docker/overview

## Autodiscovery for containerized application performance monitoring

Container orchestration tools like Docker Enterprise, Google Kubernetes or Red-Hat OpenShift are a perfect fit for agile DevOps teams. The orchestration tools make an automatic decision on which node in the cluster the application containers are deployed - no worries anymore to scale applications according to the current workload! The dynamic nature of container deployments make it more difficult to track where which application is running - and in case of performance issues, it is critical to have a full picture of cluster, server and application performance. How can you monitor application metrics in such dynamic environment?

## General Container Monitoring

A basic coverage for server and container monitoring provides Sematext Docker Agent, which collects general host- and container metrics, container events, and container logs.

Container metrics do not include application-specific metrics like index rate of Elasticsearch clusters or the number of Database operations or web server request rates. Such metrics must be monitored with application-specific monitoring agents, such as Sematext SPM-Client.

## Application monitoring for containerized applications

Monitoring of application-specific metrics like Queue size of message queues, Database query times, Elasticsearch indexing rate or web server request rates requires the setup of application-specific monitoring agents. Typically application monitoring agents connect to the application via TCP/IP to collect application metrics. In container environments, such setup can't be static because IP addresses and port numbers are changing dynamically. In addition, an application might run in an isolated virtual network and the monitoring agent must be able to connect to the application network.

Sematext SPM-Client Auto-Discovery solves the issues above by the automatic discovery of all relevant information for application monitoring: - SPM-Client container connects to Docker Remote API and Kuberntes API to discover new application containers and network parameters - SPM-Client connects to container networks and Kubernetes pods for monitoring - Application containers

1

can be tagged with the SPM_TOKEN, and SPM-Client uses the SPM_TOKEN label or environment variable to ship metrics to the right Monitoring App in Sematext Cloud (or Sematext Enterprise).

## How does SPM Client Autodiscovery work?

The SPM Client container gets distributed to every Kubernetes, Docker Swarm or Docker Enterprise node. Then it starts watching container events and compares the image name of started containers with a list of application images defined in a configuration template called `autoDiscovery.yml`. Then SPM-Client joins the container network and configures SPM-Client for the right IP-address and port and other settings according to the configuration template.

Monitoring with Sematext Cloud requires the SPM_TOKEN for each application type. The SPM_TOKEN is read from the application container environment or container labels.

Example: To monitor a containerized application like Elasticsearch with SPM-Client Autodiscovery we need only 3 steps:

1) Create a monitoring app for Elasticsearch in Sematext UI.
2) Deploy SPM-Client container as Kubernetes DaemonSet or global Swarm Service. Instructions are displayed in Sematext UI.

3) Add the displayed SPM_TOKEN to the environment of your Elasticsearch container.

## Customize autoDiscovery.yml

There are typically a few cases you might need to adjust settings in autoDiscovery.yml:

1) Using custom image names. SPM-Client uses the images names of official application images like `nginx`, `mongo`,`elasticsearch`. In case you use custom images with different names or you want to use JVM monitor for your own Java application the settings for application image names must be adjusted. A shortcut is to overwrite values in autoDiscovery.yml by environment variables `spm_templates__jvm__matchImageName=myJavaAppImage|jetty|spring`
2) Changing Java JMX settings globally. The current template assumes JMX on port 3000 with no authentication. Please not JMX ports don't need to be exposed as service because SPM-Client connects to the container network automatically.

## Autodiscovery configuration settings

All default settings of autoDiscovery.yml can be overwritten via environment variables. Property names are separated by two underscores `___` and the environment variables have the prefix `spm_templates`. TO change e.g. JMX port for JVM apps use `spm_templates__jvm__jmxPort=3001`. Overwriting a few values via environment variables is an easy way to change a few settings. Alternativly you could mount autoDiscovery.yml to `/usr/lib/node_modules/docker-spm-client/autoDiscovery.yml` into 'sematext/spm-client:autodiscovery' container.

The complete list of properties expressed as evironment variable for `sematext/spm-client:autodiscovery` image:

| Env. variable | Description |
| --- | --- |
| *Nginx* | |
| `spm_templates__nginx__portInContainer` | Port in the container, default `80/tcp` |
| `spm_templates__nginx__matchImageName` | regexp expression to match image name. Default `nginx` |

3

| Env. variable | Description |
| --- | --- |

`spm_templates__nginx__matchContainerName` expression to match container name. Default `.*`

Apache Webserver

`spm_templates__httpd__portInContainer` in the container, default `80/tcp`

`spm_templates__httpd__matchImageName` expression to match image name. Default `httpd`

4

| Env. variable | Description |
| --- | --- |
| spm_templates__httpd__matchContainerName | Regular expression to match container name. Default `.*` |

HAProxy

| Env. variable | Description |
| --- | --- |
| spm_templates__haproxy__portInContainer | Port in the container, default `1936/tcp` |
| spm_templates__haproxy__matchImageName | Regular expression to match image name. Default `haproxy` |

| Env. variable | Description |
| --- | --- |
| spm_Templates__haproxy__matchContainerName | Template expression to match container name. Default .* |
| spm_Templates__haproxy__spmConfig__SPM_MONITOR_HAPROXY_STATS_U | Default http://${host}:${port}${containerEnv.SPM_MONITOR_HAPROXY_S |
| spm_Templates__haproxy__spmConfig__SPM_MONITOR_HAPROXY__USER | Default ${containerEnv.SPM_MONITOR_HAPROXY_USER} |
| spm_Templates__haproxy__spmConfig__SPM_MONITOR_HAPROXY_PASSWOR | Default ${containerEnv.SPM_MONITOR_HAPROXY_PASSWORD} |
| MongoDB | |
| spm_Templates__mysql__portInContainer | Port in the container, default 27017/tcp |

| Env. variable | Description |
| --- | --- |
| spm_templates__mysql__matchImageName | regexp expression to match image name. Default mongo |
| spm_templates__mysql__matchContainerName | regexp expression to match container name. Default .* |

MySQL

| | |
| --- | --- |
| spm_templates__mongodb__portInContainer | port in the container, default 27017/tcp |

| Env. variable | Description |
| --- | --- |
| spm_templates__mongodb__matchImageName | Regular expression to match image name. Default `mysql|mariadb|percona` |
| spm_templates__mongodb__matchContainerName | Regular expression to match container name. Default `.*` |
| spm_templates__mongodb__spmConfig__SPM_MONITOR_MYSQL_DB_USER | Default `${containerEnv.MYSQL_USER}` |
| spm_templates__mongodb__spmConfig__SPM_MONITOR_MYSQL_DB_PASSWORD | Default `${containerEnv.MYSQL_PASSWORD}` |

Java Virtual Machine

| Env. variable | Description |
| --- | --- |
| spm_templates__jvm__jmxPort | JMX port in the container, default `3000` |
| spm_templates__jvm__matchImageName | Regular expression to match image name. Default `java|spring|jetty` |
| spm_templates__jvm__matchContainerName | Regular expression to match container name. Default `.*` |
| spm_templates__jvm__spmConfig__SPM_MONITOR_JMX_PARAMS | Default `-Dspm.remote.jmx.url=${host}:${config.jmxPort}` |

Elasticsearch

| Env. variable | Description |
| --- | --- |
| spm_templates__elasticsearch__portInContainer | port in the container, default `9200/tcp` |
| spm_templates__elasticsearch__matchImageName | expression to match image name. Default `elasticsearch` |
| spm_templates__elasticsearch__matchContainerName | expression to match container name. Default `.*` |
| spm_templates__elasticsearch__spmConfig__SPM_MONITOR_ES_NODE_P | `http://${host}:${port}` |

Redis

| Env. variable | Description |
| --- | --- |
| spm_templates__redis__portInContainer | in the container, default `6379/tcp` |
| spm_templates__redis__matchImageName | expression to match image name. Default `redis` |
| spm_templates__redis__matchContainerName | expression to match container name. Default `.*` |
| spm_templates__redis__spmConfig__REDIS_PASSWORD | `null` |

Solr Cloud

| Env. variable | Description |
| --- | --- |
| spm_templates__solrCloud__jmxPort | JMX port in the container, default 3000 |
| spm_templates__solrCloud__matchImageName | Regexp expression to match image name. Default solr |
| spm_templates__solrCloud__matchContainerName | Regexp expression to match container name. Default .* |
| spm_templates__solrCloud__spmConfig__SPM_MONITOR_JMX_PARAMS | Default -Dspm.remote.jmx.url=${host}:${config.jmxPort} |

| Env. variable | Description |
| --- | --- |
| **Kafka Broker** | |
| `spm_templates__kafka-broker__jmxPort` | JMX port in the container, default `3000` |
| `spm_templates__kafka-broker__matchImageName` | Regular expression to match image name. Default `kafka` |
| `spm_templates__kafka-broker__matchContainerName` | Regular expression to match container name. Default `.*` |

| Env. variable | Description |
| --- | --- |
| `spm_templates__kafka-broker__spmConfig__SPM_MONITOR_JMX_PARAMS` | Default is `-Dspm.remote.jmx.url=${host}:${config.jmxPort}` |
| `spm_templates__kafka-producer__jmxPort` | Kafka Producer JMX port in the container, default 3000 |
| `spm_templates__kafka-producer__matchImageName` | Regexp expression to match image name. Default `kafka` |

| Env. variable | Description |
|---|---|
| `spm_templates__kafka-producer__matchContainerName` | Regular expression to match container name. Default `.*` |
| `spm_templates__kafka-producer__spmConfig__SPM_MONITOR_JMX_PARA` | Default `-Dspm.remote.jmx.url=${host}:${config.jmxPort}` |
| | Kafka Consumer |
| `spm_templates__kafka-consumer__jmxPort` | JMX port in the container, default `3000` |

| Env. variable | Description |
| --- | --- |
| spm_templates__kafka-consumer__matchImageName | expression to match image name. Default `kafka` |
| spm_templates__kafka-consumer__matchContainerName | expression to match container name. Default `.*` |
| spm_templates__kafka-consumer__spmConfig__SPM_MONITOR_JMX_PARA | `-Dspm.remote.jmx.url=${host}:${config.jmxPort}` |

Apache Tomcat

| Env. variable | Description |
| --- | --- |
| spm_templates__tomcat__jmxPort | JMX port in the container, default 3000 |
| spm_templates__tomcat__matchImageName | Reg exp expression to match image name. Default `tomcat` |
| spm_templates__tomcat__matchContainerName | Reg exp expression to match container name. Default `.*` |
| spm_templates__tomcat__spmConfig__SPM_MONITOR_JMX_PARAMS | Default `-Dspm.remote.jmx.url=${host}:${config.jmxPort}` |

| Env. variable | Description |
| --- | --- |
| **Apache Zookeeper** | |
| `spm_templates__zookeeper__jmxPort` | JMX port in the container, default **3000** |
| `spm_templates__zookeeper__matchImageName` | Regexp expression to match image name. Default `zookeeper` |
| `spm_templates__zookeeper__matchContainerName` | Regexp expression to match container name. Default `.*` |

| Env. variable | Description |
| --- | --- |
| `spm_templates__zookeeper__spmConfig__SPM_MONITOR_JMX_PARAMS` | `-Dspm.remote.jmx.url=${host}:${config.jmxPort}` |

Apache Cassandra

| `spm_templates__cassandra__jmxPort` | JMX port in the container, default `7199` |
| `spm_templates__cassandra__matchImageName` | Regexp expression to match image name. Default `cassandra` |

| Env. variable | Description |
| --- | --- |
| spm_templates__cassandra__matchContainerName | Template expression to match container name. Default .* |
| spm_templates__cassandra__spmConfig__SPM_MONITOR_JMX_PARAMS | Defaults to `-Dspm.remote.jmx.url=${host}:${config.jmxPort}` |