Configure Prometheus and Grafana for JFrog Artifactory

- **Need to create an endpoint that exposes Artifactory applicative metrics to prometheus**
- **The applicative metrics is expected to have some statistics, sizing, timing of processes that artifactory could share with users - to use them to create visualized dashboards and setup alerts based on those metrics.**

Run from Docker Hub:

docker run -p 9100:9100 -e ARTIFACTORY_URL=http://novartis.devops.altimetrik.io:8081/artifactory -e ARTIFACTORY_USER=admin -e ARTIFACTORY_PASSWORD=Password petrjurasek/artifactory-prometheus-exporter

Metrics

You can access the metric

here: http://novartis.devops.altimetrik.io:9100/metrics

Metrics are then available on listening port and examples

```
# HELP go_gc_duration_seconds A summary of the GC invocation durations.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 3.0623e-05
go_gc_duration_seconds{quantile="0.25"} 4.041e-05
go_gc_duration_seconds{quantile="0.5"} 4.4972e-05
go_gc_duration_seconds{quantile="0.75"} 5.5399e-05
go_gc_duration_seconds{quantile="1"} 0.000970338
go_gc_duration_seconds_sum 0.055092078
go_gc_duration_seconds_count 690
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 9
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.12.5"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in
use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 1.564256e+06
# HELP go_memstats_alloc_bytes_total Total number of bytes allocated,
even if freed.
# TYPE go_memstats_alloc_bytes_total counter
go_memstats_alloc_bytes_total 1.992339728e+09
# HELP go_memstats_buck_hash_sys_bytes Number of bytes used by the
profiling bucket hash table.
# TYPE go_memstats_buck_hash_sys_bytes gauge
go_memstats_buck_hash_sys_bytes 1.589889e+06
# HELP go_memstats_frees_total Total number of frees.
# TYPE go_memstats_frees_total counter
go_memstats_frees_total 1.712497e+07
# HELP go_memstats_gc_cpu_fraction The fraction of this program's
available CPU time used by the GC since the program started.
# TYPE go_memstats_gc_cpu_fraction gauge
go_memstats_gc_cpu_fraction 2.6031896897152185e-05
# HELP go_memstats_gc_sys_bytes Number of bytes used for garbage
```

```
collection system metadata.
# TYPE go_memstats_gc_sys_bytes gauge
go_memstats_gc_sys_bytes 2.377728e+06
# HELP go_memstats_heap_alloc_bytes Number of heap bytes allocated and
still in use.
# TYPE go_memstats_heap_alloc_bytes gauge
go_memstats_heap_alloc_bytes 1.564256e+06
# HELP go_memstats_heap_idle_bytes Number of heap bytes waiting to be
used.
# TYPE go_memstats_heap_idle_bytes gauge
go_memstats_heap_idle_bytes 6.3594496e+07
# HELP go_memstats_heap_inuse_bytes Number of heap bytes that are in
use.
# TYPE go_memstats_heap_inuse_bytes gauge
go_memstats_heap_inuse_bytes 2.957312e+06
# HELP go_memstats_heap_objects Number of allocated objects.
# TYPE go_memstats_heap_objects gauge
go_memstats_heap_objects 7516
# HELP go_memstats_heap_released_bytes Number of heap bytes released to
OS.
# TYPE go_memstats_heap_released_bytes gauge
go_memstats_heap_released_bytes 6.0416e+07
# HELP go_gc_duration_seconds A summary of the GC invocation durations.
# TYPE go_gc_duration_seconds summary
artifactory_artifacts_downloaded{key="example-repo-local",minutes_ago="1
"} 0.0
artifactory_artifacts_downloaded{key="example-repo-local",minutes_ago="6
0"} 0.0
artifactory_artifacts_downloaded{key="example-repo-local",minutes_ago="5
"} 0.0
# HELP artifactory_artifacts_created Created artifacts
# TYPE artifactory_artifacts_created gauge
artifactory_artifacts_created{key="example-repo-local",minutes_ago="1"}
0.0
artifactory_artifacts_created{key="example-repo-local",minutes_ago="60"}
0.0
artifactory_artifacts_created{key="example-repo-local",minutes_ago="5"}
0.0
# HELP artifactory_repository_files_count Artifactory repository file
count
# TYPE artifactory_repository_files_count gauge
artifactory_repository_files_count{key="example-repo-local",type="LOCAL"
} 0.0
# HELP artifactory_security_users Number of artifactory users
# TYPE artifactory_security_users gauge
artifactory_security_users{realm="internal"} 4.0
# HELP artifactory_security_groups Number of artifactory groups
# TYPE artifactory_security_groups gauge
artifactory_security_groups 1.0
# HELP artifactory_system_licence Licence information
```

```
# TYPE artifactory_system_licence gauge
artifactory_system_licence{expires="Dec 21, 2018"} 21.0
# HELP artifactory_system_revision Version information
# TYPE artifactory_system_revision gauge
artifactory_system_revision{version="6.5.8"} 60508900.0
# HELP node_cpu_guest_seconds_total Seconds the cpus spent in guests
(VMs) for each mode.
# TYPE node_cpu_guest_seconds_total counter
node_cpu_guest_seconds_total{cpu="0",mode="nice"} 0
node_cpu_guest_seconds_total{cpu="0",mode="user"} 0
node_cpu_guest_seconds_total{cpu="1",mode="nice"} 0
node_cpu_guest_seconds_total{cpu="1",mode="user"} 0
# HELP node_cpu_seconds_total Seconds the cpus spent in each mode.
# TYPE node_cpu_seconds_total counter
node_cpu_seconds_total{cpu="0",mode="idle"} 13829.22
node_cpu_seconds_total{cpu="0",mode="iowait"} 26.07
node_cpu_seconds_total{cpu="0",mode="irq"} 0
node_cpu_seconds_total{cpu="0",mode="nice"} 0.05
# HELP node_disk_io_now The number of I/Os currently in progress.
# TYPE node_disk_io_now gauge
node_disk_io_now{device="dm-0"} 0
node_disk_io_now{device="dm-1"} 0
node_disk_io_now{device="xvda"} 0
node_disk_io_now{device="xvdf"} 0
node_disk_io_now{device="xvdg"} 0
# HELP node_disk_io_time_seconds_total Total seconds spent doing I/Os.
# TYPE node_disk_io_time_seconds_total counter
node_disk_io_time_seconds_total{device="dm-0"} 31.155
node_disk_io_time_seconds_total{device="dm-1"} 18.628
node_disk_io_time_seconds_total{device="xvda"} 51.312
node_disk_io_time_seconds_total{device="xvdf"} 31.137
node_disk_io_time_seconds_total{device="xvdg"} 18.641000000000002
# HELP node_disk_io_time_weighted_seconds_total The weighted # of
seconds spent doing I/Os.
# TYPE node_disk_io_time_weighted_seconds_total counter
node_disk_io_time_weighted_seconds_total{device="dm-0"} 128.369
node_disk_io_time_weighted_seconds_total{device="dm-1"} 347.141
node_disk_io_time_weighted_seconds_total{device="xvda"} 491.624
node_disk_io_time_weighted_seconds_total{device="xvdf"} 113.473
node_disk_io_time_weighted_seconds_total{device="xvdg"} 308.887
# HELP node_disk_read_bytes_total The total number of bytes read
successfully.
# TYPE node_disk_read_bytes_total counter
node_disk_read_bytes_total{device="dm-0"} 7.43973888e+08
node_disk_read_bytes_total{device="dm-1"} 5.9234816e+08
node_disk_read_bytes_total{device="xvda"} 6.09327616e+08
node_disk_read_bytes_total{device="xvdf"} 7.44477696e+08
node_disk_read_bytes_total{device="xvdg"} 5.92851968e+08
# HELP node_disk_read_time_seconds_total The total number of seconds
spent by all reads.
```

```
# TYPE node_disk_read_time_seconds_total counter
node_disk_read_time_seconds_total{device="dm-0"} 90.194
node_disk_read_time_seconds_total{device="dm-1"} 262.573
node_disk_read_time_seconds_total{device="xvda"} 395.885
```

```
node_disk_read_time_seconds_total{device="xvdf"} 88.812
node_disk_read_time_seconds_total{device="xvdg"} 262.597
# HELP node_disk_reads_merged_total The total number of reads merged.
```

Configuration

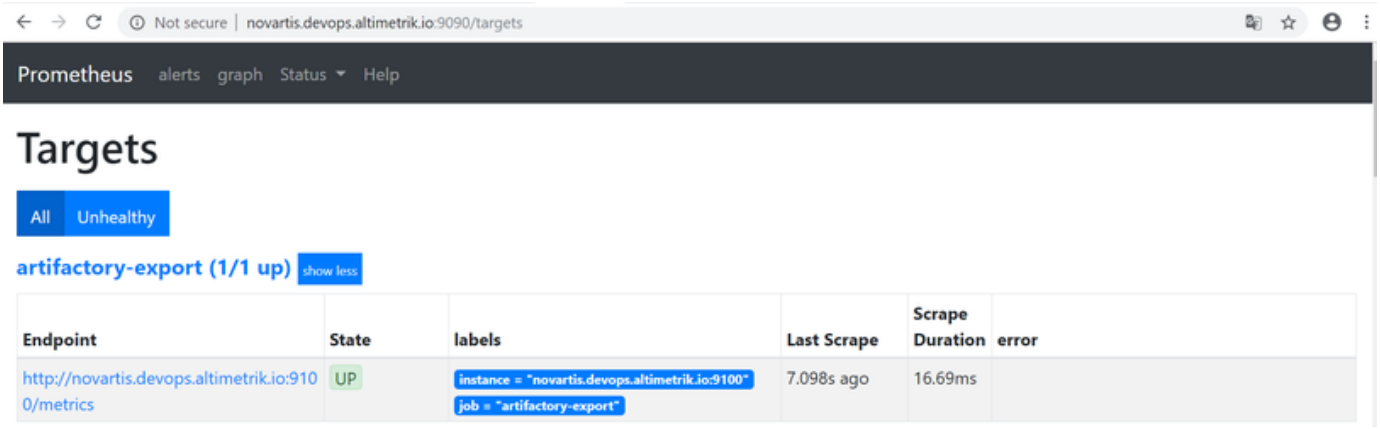
After plug-in is successfully installed the following link can be used to expose **metrics: http://novartis.devops.altimetrik.io:9100/metrics**

The *prometheus.yml* settings looks like:

/data/prometheus/config/prometheus.yaml

```
- job_name: artifactory-exporter
  honor_timestamps: true
  scrape_interval: 15s
  scrape_timeout: 10s
  metrics_path: /metrics
  scheme: http
  ec2_sd_configs:
  - endpoint: ""
    region: us-east-1
    access_key: AKIAXZXIY4Q4M3JKI4TO
    secret_key: <secret>
    profile: arn:aws:iam::536285340728:user/devplatarn
    refresh_interval: 1m
    port: 9100
    filters: []
  basic_auth:
    username: admin
    password: <secret>
  relabel_configs:
  - source_labels: [__meta_ec2_tag_Name]
    separator: ;
    regex: ^novartis-devops-artifactory$
    replacement: $1
    action: keep
  - source_labels: [__meta_ec2_public_ip]
    separator: ;
    regex: (.+)
    target_label: __address__
    replacement: novartis.devops.altimetrik.io:9100
    action: replace
```

Installation and setup is finished. You can collect metrics. Prometheus will be available at http://novartis.devops.altimetrik.io:9090/

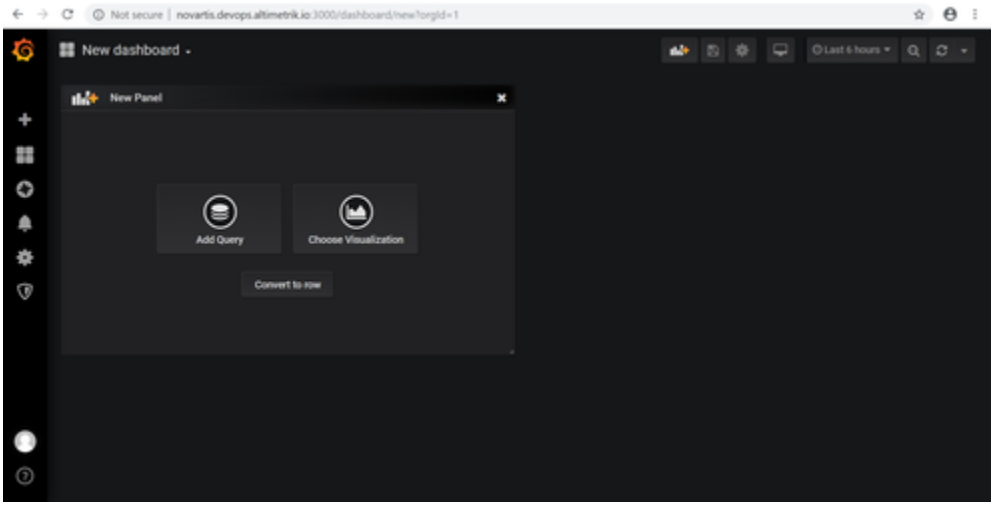


Grafana Configuration

We will use Grafana to visualize metrics stored in Prometheus. There are a couple of example dashboards in the official site https://grafana.com/grafana/dashboards

Download the latest release of Grafana for your platform, then extract it:

Here i am building manually using query.



Memory handler Request Total

Metrics -promhttp_metric_handler_requests_total{code="503",instance="novartis-devops-artifactory",job="node-exporter"}

Metrics -promhttp_metric_handler_requests_total{code="500",instance="novartis-devops-artifactory",job="node-exporter"}

Metrics -promhttp_metric_handler_requests_total{code="200",instance="novartis-devops-artifactory",job="node-exporter"}

Process Max File Discriptor

Metrics -process_max_fds{instance="novartis.devops.altimetrik.io:9100",job="artifactory-exporter"}

CPU Per Second Total

Metrics -process_cpu_seconds_total{instance="novartis.devops.altimetrik.io:9100",job="artifactory-exporter"}

Virtual Memory Max

Metrics -process_virtual_memory_max_bytes{instance="novartis.devops.altimetrik.io:9100",job="artifactory-exporter"}

MemStat Free Total

Metrics - go_memstats_frees_total{instance="novartis.devops.altimetrik.io:9100",job="artifactory-exporter"}

After a couple of minutes you will be able to view your metrics on the Dashboard. You can also add new panels to the Dashboard.