# Cronapp APM

Ter uma visão do desempenho e verificar os problemas que podem ocorrer em tempo real são algumas das vantagens que o sistema de monitoramento oferecem. Para facilitar o monitoramento de sistemas, o Cronapp possui a ferramenta Cronapp APM, que permite exportar métricas da aplicação para serem usadas com o Prometheus. O Cronapp APM foi construído a partir da API Spring Boot Actuator e do mecanismo micrometer-prometeus, permitindo gerar recursos de monitoramento, auditoria e outras propriedades pertencentes do Spring Boot no formato que é utilizado pelo Prometheus.

## **Endpoints**

Um endpoint é uma URI que normalmente direciona para o serviço de uma API. Dessa forma, a API do Spring Boot disponibiliza alguns endpoints próprios e também utiliza o endpoint pertencente ao Prometheus: <URL do Servidor>/actuator/prometheus, que gera as métricas no formato que será lido pelo servidor do Prometheus.

O formato de utilização do endpoint é geral: actuator/id como, por exemplo, actuator/health, actuator /env.



Existem diversos endpoints gerados pelo Actuador, como <URL do Servidor>/actuator/health ou <URL do Servidor>/actuator/env. A lista com todos os endpoints pertencentes ao Spring boot actuador podem ser acessados no link.

### Permissão de segurança

Como a funcionalidade já vem habilitada ao criar o projeto, é necessário dar permissão ao endpoint (recomenda-se inicialmente configurar a permissão para authenticated), que fica no endereço /actuator/\*\* . Siga os passos abaixo para configurá-la.

â	Permissão de Segu	urança ×							0
6	Permissionáveis	📇 Grupos 💄 Usu							
Q,									Ċ
6									Ŧ
6		Permissionaveis - Editar					(Z		Ť
ô					3			R	Ť
					$\odot$				
			Visões - Inserin	do					
			Geral						
			Padrão	/actuator/**		4			
			Permitir GET						
			Permitir POST						
			Permitir PUT						
			Permitir HEAD						
			Permitir PATCH						
			Permitir DELETE						
			Permitir OPTIONS						
			Permitir TRACE						
						🔲 Sal	var X Canc	elar	

Figura 1.1 - Definir a permissão de segurança para acessar a página

- 1. Como o botão Modo Avançado habilitado, clique em Projeto no menu de sistemas e selecione Permissão de Segurança;
- 2. Após abrir a janela, na aba Permissionáveis clique no botão Editar de Authenticated;
- Clique na aba Visões e, em seguida, no ícone "+" para Adicionar;
   No campo Padrão escreva o caminho /actuator/\*\*, deixe todos os itens selecionados, clique em Salvar e clique em Salvar novamente na janela de edição.

### Métricas exportadas

Com a permissão devidamente configurada, rode o projeto e acesse a URL <URL do Servidor> /actuator em seu navegador. Remova a parte "#/home" e acrescrente "actuator" (Figura 1.2).

#### Nesta página

- Endpoints
- Permissão de segurança Métricas exportadas
- Obtendo as métricas no Prometheus
- Visualizando as métricas em gráficos no Grafana

<pre>#IELP ism_biffer_memory_used_bytes in estimate of the memory that the laws virtual machine is using for this buffer pool #IPE ["bu_biffer_memory_used_bytes gauge ism_biffer_memory_used_bytes [la"mapped"], 0.40 #IELP ism_memory_used_bytes [mass-formations], 1311872.0 %0.memory_used_bytes[mass-formations], 145"codeseg "non-method"], 1311872.0 %0.memory_used_bytes[mass-formations], 145"codeseg "non-method"], 1311872.0 %0.memory_used_bytes[mass-formations], 145"codeseg "non-method"], 1311872.0 %0.memory_used_bytes[mass-formations], 145"codeseg "non-methods"], 1311872.0 %0.memory_used_bytes[mass-formations], 145"codeseg "non-methods"], 1318872.0 %0.memory_used_bytes[mass-formations], 145"codeseg "non-methods"], 1318872.0 %0.memory_used_bytes[mass-formations], 145"codeseg "non-portilate methods"], 1.394807207 %0.memory_used_bytes[mass-formations], 145"codeseg "non-portilate methods"], 1.39480727 %0.memory_used_bytes[mass-formations], 145"codeseg "non-portilate methods"], 1.39480727 %0.memory_used_bytes[mass-formations], 145"codeseg "non-portilate methods"], 1.39480727 %0.memory_used_bytes[mass-formations], 145"codeseg "non-portilate methods"], 1.272230.0 %0.memory_used_bytes[mass-formations], 146"codeseg "non-portilate methods"], 1272230.0 %0.memory_used_bytes[mass-formations], 146"codeseg "non-portilate methods"], 146%000000000000000000000000000000000000</pre>

Figura 1.2 - Métricas exportadas pelo plugin

### Obtendo as métricas no Prometheus

Nessa etapa será mostrado como utilizar as métricas geradas pela API no Prometheus.

prometneus							
The Prometheus monitoring system and time series database. O prometheus/prometheus							
2.24.1 / 2021-01-20 Release notes							
File name	os	Arch	Size	SHA256 Checksum			
prometheus-2.24.1.darwin-amd64.tar.gz	darwin	amd64	62.53 MIB	73c27ba24f5b7beaf78a7bd46a6605066c6ad5f4db82866dc577aaf55b84458			
prometheus-2.24.1.linux-amd64.tar.gz	linux	amd64	62.47 MiB	Saec18296624449e83469eF647cb762b44de2ae12Fc91d2375c5e6be9Fd849c			
prometheus-2.24.1 windows-amd64.zin	windows	amd64	63.66 MIR	2ef42e064051381062de1db3e676f45091bc73b2b3bea8dc82d2dd0ce54c7b2			

Figura 2.1 - Versão utilizada

- 1. Primeiramente, faça o download do prometheus e extraia a pasta.
- 2. Finalizada a extração, abra o arquivo prometheus.yml (pode ser aberto com o Bloco de Notas do Windows ou similar em outro sistema operacional) e transcreva o código abaixo para esse arquivo, de modo que ele esteja da mesma forma que é mostrado (destaque em vermelho) na Figura 2.1.
- 3. Em seguida volte ao ambiente Cronapp, rode o projeto, abra no navegador web e copie a URL do Servidor. Cole-o dentro (destaque 1 da Figura 2.1) da linha *targets*.
- 4. Salve o arquivo.

#### Exemplo do scrape\_config para adicionar no arquivo

- job\_name: define um nome para que o Prometheus reconheça
- metrics\_path: informa o endereço em que as métricas são geradas e ele é fixo, não havendo necessidade de alteração.
- scheme: protocolo utilizado pela aplicação.
- static\_configs: parâmetro utilizado para configurar os campos estáticos do target.
  - targets: informa o endereço da aplicação e a porta que ela utiliza no formato ['endereç o:porta']

#### Observação

YAM, a linguagem utilizada no Prometheus, utiliza uma notação baseada em indentação. Verifique no seu arquivo se está como na Figura 2.1.

prometheus.yml - Bloco de Notas				- 0	×	
Arquivo Editar Formatar Exibir Ajuda						
# my global config					1	
<pre>groups: scrape_interval: 15s # Set the scrape interval evaluation_interval: 15s # Evaluate rules every 15 # scrape_timeout is set to the global default (10s)</pre>	to every 15 secon seconds. The defa	ds. De ult is	fault is every every 1 minut	1 minute. e.		
<pre># Alertmanager configuration alerting:     alertmanagers:     static_configs:     targets:     # - alertmanager:9093</pre>						
<pre># Load rules once and periodically evaluate them according to the global 'evaluation_interval'. rule_files: # "first_rules.yml" # - "second_rules.yml"</pre>						
<pre># A scrape configuration containing exactly one endpo # Here it's Prometheus itself. scrape_configs: # The job name is added as a label `job=<job_name>' _job_name: 'prometheus'</job_name></pre>	int to scrape: to any timeserie	s scra	ped from this	config.		
<pre># metrics_path defaults to '/metrics' # scheme defaults to 'http'.</pre>						
static_configs: - targets: ['localhost:9090']						
<ul> <li>job_name: 'spring' metrics_path: '/actuator/prometheus' scheme: 'https'</li> <li>static_configs:</li></ul>	3']					
¢.	Ln 35. Col 51	100%	Unix (LF)	UTF-8	>	

Figura 2.1 - Configurando arquivo prometheus.yml

Após salvar o arquivos, inicie a aplicação clicando duas vezes no arquivo *prometheus.exe*, abrirá a janela do terminal deixe-o aberta. Abra uma página no navegador e digite o endereço **localhost:9090**. Clique em **Status** (destaque 1 da Figura 2.2) e vá para a aba **Targets** (2) para visualizar se a aplicação está sendo reconhecida pelo Prometheus (3).

Prometheus	Alerts	s Graph	Status Help Classic UI			
Targets			Runtime & Build Information TSDB Status Command-Line Flags Configuration	n		
Endpoint http://localhost 0/metrics	:909	up) show State UP	Rules Targets ② J <sup>m</sup> Service Discovery job="promethous"	Scrape Duration 5.985ms	Error	
spring (1/1 Endpoint	up) 💁 State	Labels	)	Last S Scrape I	Scrape Duration	Error
https://app-28 -171-12118.id e.cronapp.io/a ctuator/prome theus	UP	instance: job="spr	-*app-28-171-12118.ide.cronapp.ios443 ing*	<b>6.519s</b> 3	314.532ms	

Figura 2.2 - Targets reconhecidas pelo Prometheus

Agora, clique no menu **Graph** do Prometheus e escolha uma query para obter os dados que são mostrados em **elements**, como mostrado na Figura 2.3.

rometheus Alerts Graph Status * Help				
jvm_memory_used_bytes	Load time: 8ms Resolution: 14s Total time series:			
Execute - insert metric at cursor - +				
Graph Console				
← Moment				
Element	Value			
jvm_memory_used_bytes(area="heap",id="Eden Space",instance="app-15-250-13150.ide.cronapp.icx443",job="spring")	23002968			
jvm_memory_used_bytes(area="heap",id="Survivor Space",instance="app-15-250-13150,ide.cronapp.iox443",job="spring")	2416432			
jvm_memory_used_bytes(area="heap",id="Tenured Gen",instance="app-15-250-13150.ide.cronapp.io:443",job="spring"}				
	1311872			
jvm_memory_used_bytes(area="nonheap",id="CodeHeap 'non-nmethods",instance="app-15-250-13150.ide.cronapp.io:443".job="spring")				
jvm_memory_used_bytes{area="nonheap";id="CodeHeap 'non-nmethods";instance="app-15-250-13150.ide.cronapp.iox443";job="spring"; jvm_memory_used_bytes{area="nonheap";id="CodeHeap 'non-profiled nmethods";instance="app-15-250-13150.ide.cronapp.iox443";job="spring";	7773056			
jum_memory_used_bytes/tera="nonhaep".id="CodeHeap" tron-methods".intance="app-15-259-1130.idae.comapp.io.443".job="sping"] jum_memory_used_bytes/tera="nonhaep".id="CodeHeap" tron-profiled methods".intance="app-15-250-1130.idae.comapp.io.443".job="sping"] jum_memory_used_bytes/tera="nonhaep".id="CodeHeap" profiled methods".intance="app-15-250-1130.idae.comapp.io.443".job="sping"]	7773056 24249728			
jim memory used bytestream-inonheap'sid="CodeHeap from methods" instance="app-15-250-1150.decompapite43" job="spring"] jim memory used bytestream-inonheap'sid="CodeHeap from politied methods" instance="app-15-250-1150.decompapite43" job="spring"] jim memory used bytestream-inonheap'sid="CodeHeap politied methods" instance="app-15-250-1150.decompapite43" job="spring"] jim memory used bytestream-inonheap'sid="CodeHeap politied methods" instance="app-15-250-1150.decompapite43" job="spring"]	7773056 24249728 10178344			

Figura 2.3 - Dados retornados pela query escolhida

### Visualizando as métricas em gráficos no Grafana

A plataforma Grafana permite que as métricas disponibilizadas pelos softwares de monitoramento sejam visualizados através de gráficos. Com isso, faça o download do Grafana e instale-o em sua máquina. Após finalizar, abra o link default do Grafana, o **localhost:3000**, e entre com o usuário e senha padrão (admin/admin).

Ao logar no sistema, clique no ícone de configurações (1 da Figura 3.1) e clique em **data sources** (2 da Figura 3.1).



Figura 3.1 - Configurações > data sources

Clique em adicionar nova fonte de dados e selecione o Prometheus. No campo **url**, adicione o endereço do Prometheus (nesse caso, localhost:9090) (Figura 3.2) e salve.

Data Sources / Prometheus						
	🛫 Settings 👪 Dashboards					
Name 6	Prometheus	Default				
Name O Prometheus Default						
нттр						
URL ®	http://localhost:9090/					
Whitelisted Cookies		Add				

Figura 3.2 - Adicionar endereço do Prometheus

Por fim, crie um dashboard e clique em **add query**. Copie a query escolhida na página do prometheus no campo **metrics**.



Figura 3.3 - Adicionar query no campo

Com isso, o gráfico será gerado automaticamente.



Figura 3.4 - Gráfico gerado pela query