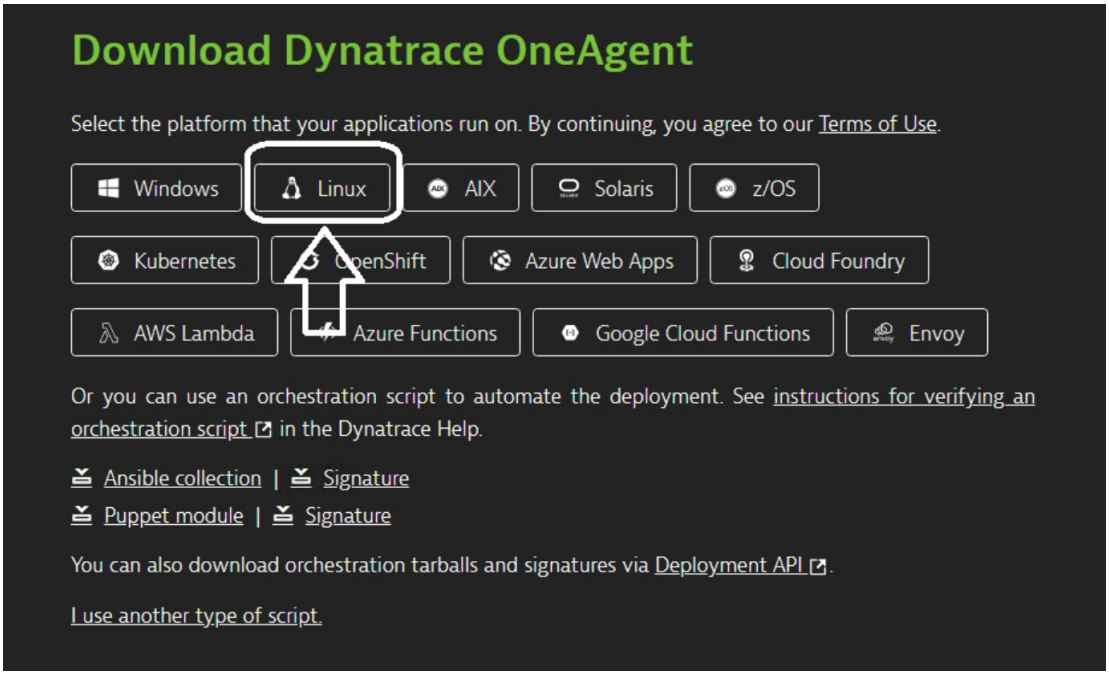
Instalação Dynatrace (APM - Kubernetes)

Ambiente Managed (on premise)

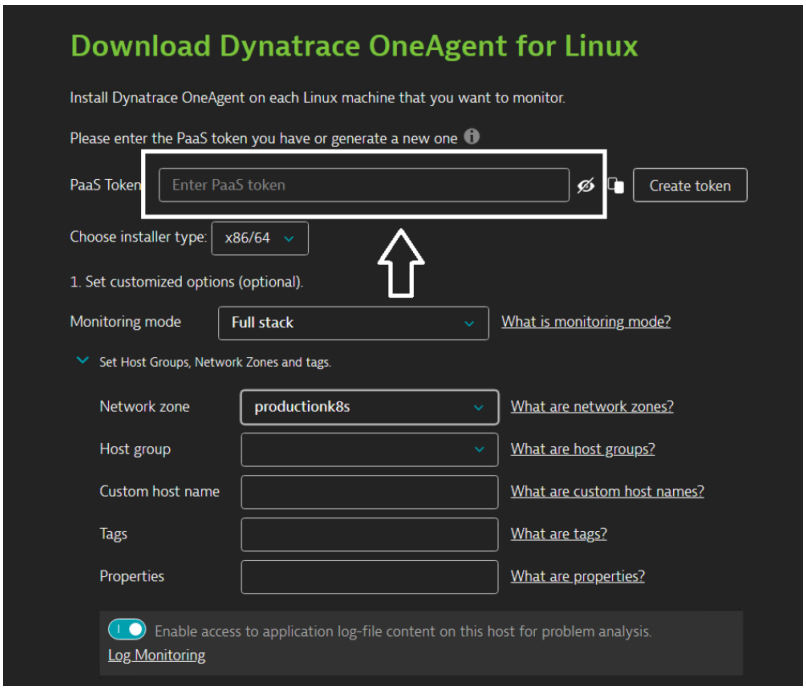
1) Valide se está liberado a porta 443 do lado dos nodes Kubernetes, e a porta 9999 no ActiveGate

2) Faça a instalação do OneAgent direto nos nós

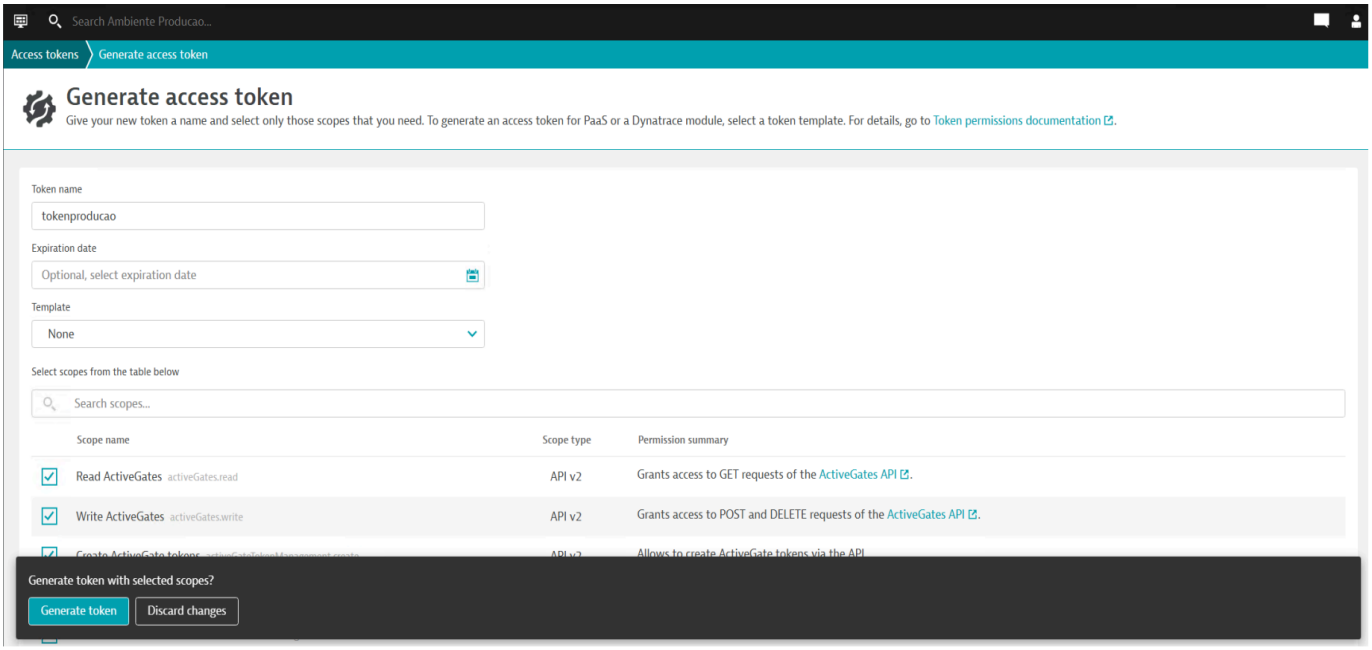
Para isso vamos gerar o instalador na console UI



⚠ **Aqui será necessário que você gere um token com as devidas permissões habilitadas**

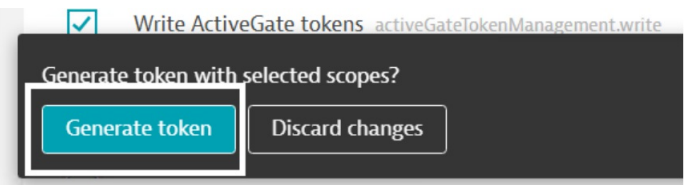


Em "Access Tokens" clique em "Generate access token"

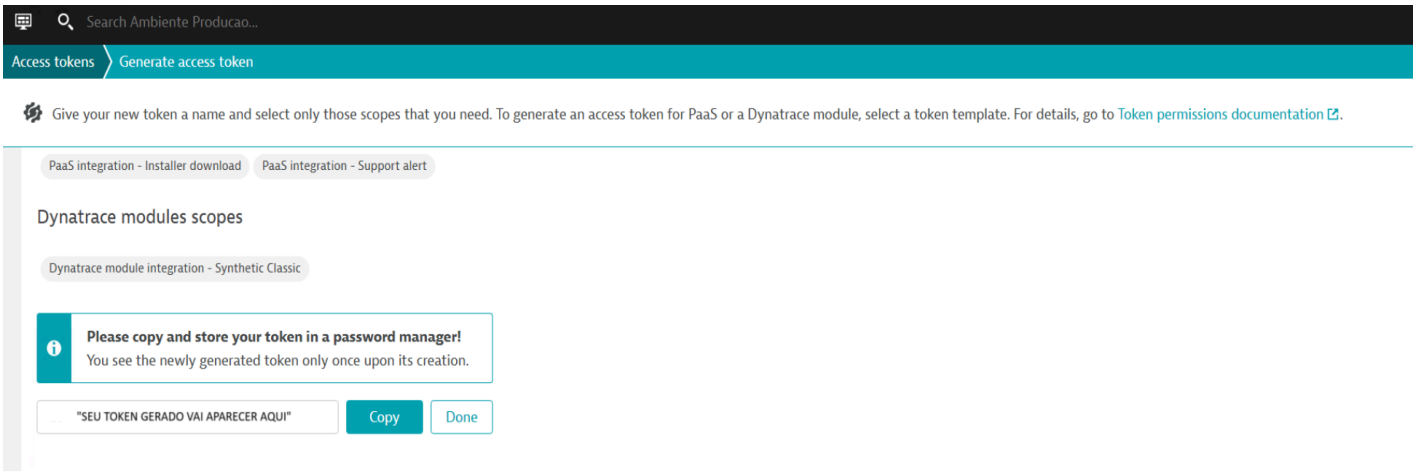


Você vai criar seu token, não coloque data de expiração para esse token expecífico. Marque todas as permissões necessárias para o OneAgent.

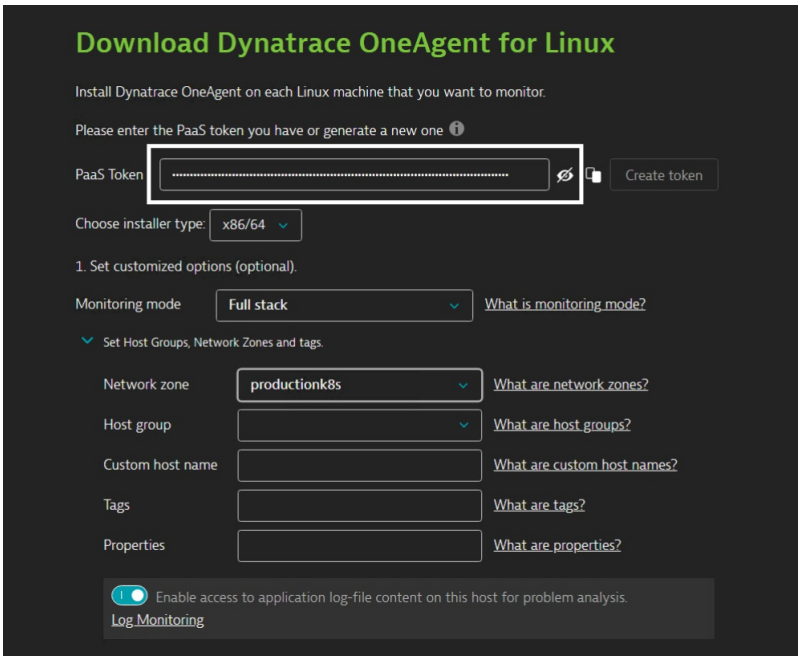
Depois de validar se está tudo certo, clique em "Generate Token".



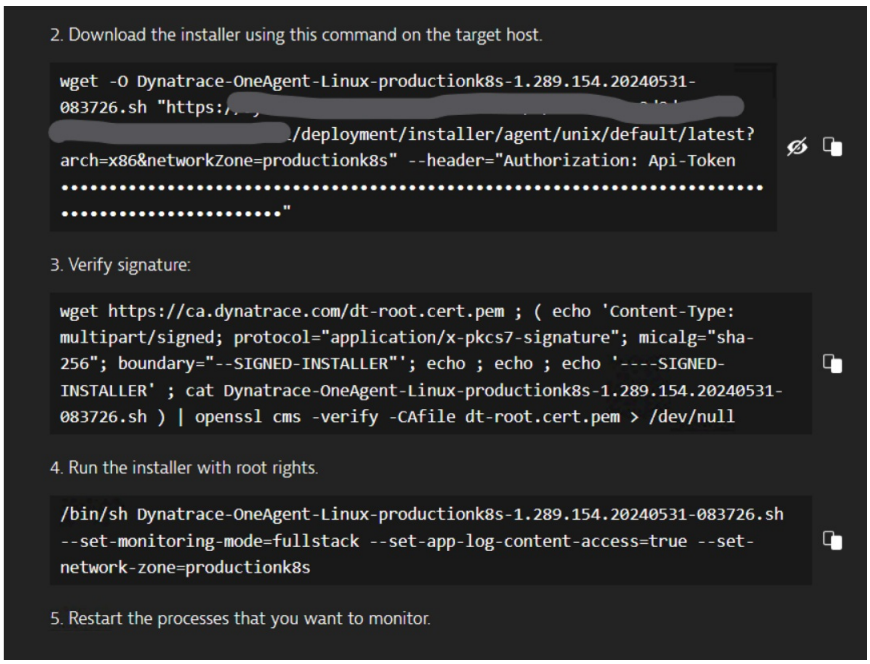
Copie o token gerado e cole em um local de segurança, vamos precisar dele para a próxima parte.



Volte a aba "Download Dynatrace OneAgent for Linux", vamos inserir o token que acabamos de criar.



Após inserir, scrole com o mouse para baixo, você vai ver três passos que precisam ser feito do lado do Sistema operacional lá nos nodes.



2. Download do instalador:

- Baixa o instalador do Dynatrace OneAgent com wget, utilizando um token de API e parâmetros como arquitetura e zona de rede.

3. Verificação da assinatura:

- Baixa o certificado raiz da Dynatrace e verifica a assinatura do instalador com openssl, garantindo sua autenticidade.

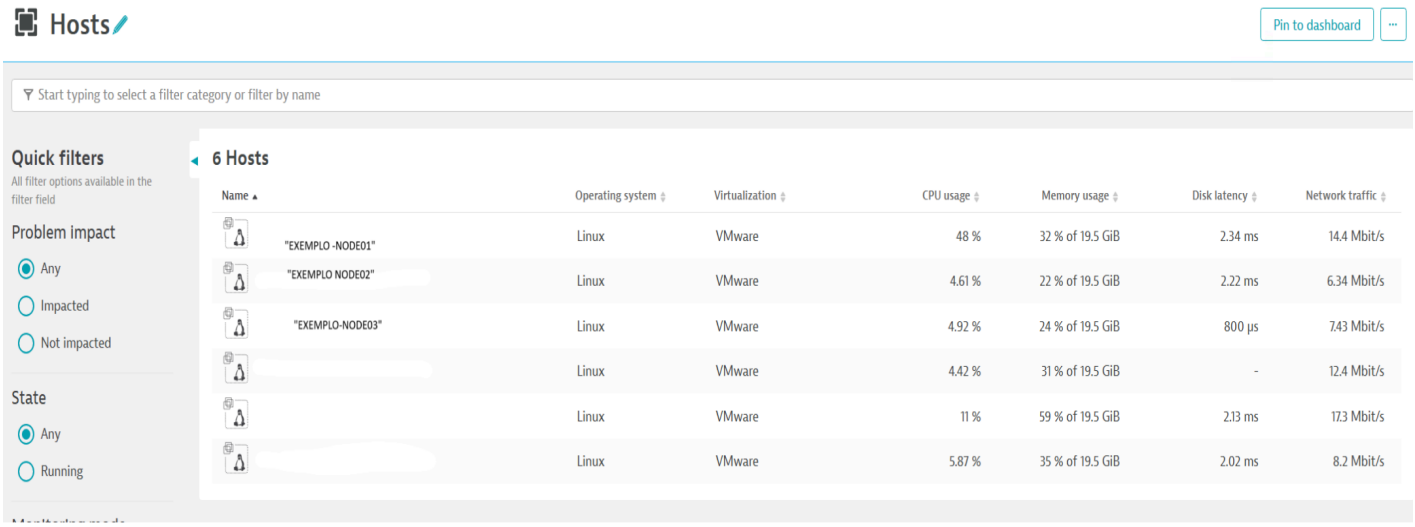
4. Execução do instalador como root:

- Executa o script de instalação com /bin/sh, configurando o modo de monitoramento como fullstack, habilitando acesso a logs de aplicativos e definindo a zona de rede.

5. Reinício de processos:

- Orienta a reiniciar os processos que devem ser monitorados para que o agente comece a coletar dados.

Depois desse processo realizado em todos os nodes você já deve começar a visualizar os hosts na console UI



3) Parte Kubernetes

Primeiro vamos fazer a instalação do Dynatrace operator na ultima versão direto do repositório oficial

```
helm install dynatrace-operator oci://public.ecr.aws/dynatrace/dynatrace-operator \
  --set csidriver.enabled="true" \
  --create-namespace \
  --namespace dynatrace \
  --atomic
```

Explicação dos parâmetros

- `helm install dynatrace-operator oci://...`
  Instala o Dynatrace Operator a partir de um repositório OCI (container registry).
- `--set csidriver.enabled="false"`
  Desativa o driver CSI, usado para gerenciamento de volumes, caso não seja necessário.
- `--set csidriver.enabled="true"`
  Ativa o driver CSI, usado para gerenciamento de volumes, caso seja necessário.
- `--create-namespace`
  Cria automaticamente o namespace `dynatrace` se ele ainda não existir no cluster.
- `--namespace dynatrace`
  Define o namespace `dynatrace` como o local onde os recursos serão instalados.
- `--atomic`
  Garante que a instalação seja atômica: se algum passo falhar, tudo será revertido automaticamente.

Criar o secret

Depois de instalado, vamos criar o secret

```
kubectl create secret generic productionk8s \
  --from-literal="apiToken=SEU_API_TOKEN" \
  --from-literal="paasToken=SEU_PAAS_TOKEN" \
  -n dynatrace
```

Substitua **SEU_API_TOKEN** e **SEU_PAAS_TOKEN** pelos valores reais gerados na interface do Dynatrace.

Verificar versões suportadas pelo CRD

```
kubectl get crd dynakubes.dynatrace.com -o=jsonpath='{.spec.versions[*].name}'
```

Configuração do Dynakube

```
apiVersion: dynatrace.com/v1beta5
kind: DynaKube
metadata:
  name: productionk8s
  namespace: dynatrace
  annotations:
    feature.dynatrace.com/metadata-enrichment: "true"
    feature.dynatrace.com/automatic-kubernetes-api-monitoring: "true"
spec:
  apiUrl: https://SEU-AMBIENTE-DYNATRACE.NET/e/1004594-4d8d-4671d-86f4-3189005dd8da35/api
  skipCertCheck: true
  tokens: productionk8s
  networkZone: productionk8s

  activeGate:
    capabilities:
      - routing
      - kubernetes-monitoring
      - dynatrace-api
    group: productionk8s
    resources:
      requests:
        cpu: 500m
        memory: 512Mi
      limits:
        cpu: 1000m
        memory: 1.5Gi
```

DynaKube Manifest - Ambiente `productionk8s`

Este manifesto define os componentes necessários para integrar o cluster Kubernetes ao Dynatrace usando o operador oficial.

Estrutura do Manifesto

- **apiVersion**: `dynatrace.com/v1beta1`
  Versão da API do CRD (Custom Resource Definition) do Dynatrace.
- **kind**: `DynaKube`
  Tipo de recurso customizado que representa a configuração da integração Dynatrace no cluster.
- **metadata.name**: `productionk8s`
  Nome da instância do DynaKube.
- **metadata.namespace**: `dynatrace`
  Namespace onde os recursos Dynatrace serão instalados.

Annotations

- `feature.dynatrace.com/metadata-enrichment: "true"`
  Habilita enriquecimento automático de metadados das entidades monitoradas.
- `feature.dynatrace.com/automatic-kubernetes-api-monitoring: "true"`
  Ativa a monitoração automática da API do Kubernetes.

Spec

- **apiUrl**
  URL do ambiente Dynatrace usado para se conectar à API.
- **skipCertCheck: true**
  Ignora verificação de certificado SSL (útil para ambientes internos ou certificados customizados).
- **tokens: productionk8s**
  Nome do `Secret` contendo os tokens `apiToken` e `paasToken`.
- **networkZone: productionk8s**
  Zona de rede lógica usada para segmentação de comunicação dentro do Dynatrace.

activeGate

- **capabilities**
  Lista de funcionalidades ativadas:
  - `routing`: roteamento de tráfego para o Dynatrace
  - `kubernetes-monitoring`: monitoramento da API do Kubernetes
  - `dynatrace-api`: acesso à API Dynatrace
- **resources**
  Requisições e limites de CPU e memória para o ActiveGate.

> **Importante**: certifique-se de criar o `Secret` chamado `productionk8s` no namespace `dynatrace` contendo os tokens `apiToken` e `paasToken`.

Aplique a configuração

```
kubectl apply -f dynakube.yaml
```

OU:

```
.\kubectl.exe --kubeconfig=.\kubeconfig-homolog.yaml apply -f C:\Users\ANDESI3\Downloads\kubectl\dynakube.yaml
```

Instrumentação (auto-injection)

Para facilitar essa parte, foi realizado a criação de um script para auxiliar a criação das labels

```
# Caminho do kubectl e do kubeconfig
$kubectl = ".\kubectl.exe"
$kubeconfig = "--kubeconfig=.\kubeconfig-homolog.yaml"

# Lista de namespaces do sistema a ignorar
$excludedNamespaces = @(
    "kube-system", "kube-public", "kube-node-lease",
    "cattle-system", "cattle-monitoring-system",
    "cattle-fleet-system", "cattle-impersonation-system",
    "calico-system", "tigera-operator",
    "longhorn-system", "cert-manager",
    "local-path-storage", "system-upgrade",
    "default"
)

# Obter todos os namespaces
$namespaces = & $kubectl $kubeconfig get ns -o jsonpath="{.items[*].metadata.name}" | Out-String
$namespaces = $namespaces.Trim().Split()

foreach ($ns in $namespaces) {
    if ($excludedNamespaces -notcontains $ns) {
        Write-Host "Aplicando label em namespace: $ns"
        & $kubectl $kubeconfig label ns $ns dynatrace.com/injection=true --overwrite
    } else {
        Write-Host "Ignorando namespace de sistema: $ns"
    }
}
```

Para validar se se o label (dynatrace.com/injection=true) foi realmente aplicado aos namespaces corretos, você pode seguir alguns dos passos abaixo:

```
kubectl get ns --show-labels
```

ou:

```
.\kubectl.exe --kubeconfig=.\kubeconfig-homolog.yaml get ns --show-labels | findstr "dynatrace.com/inject=true"
```

```
dynatrace.com/inject=true,dynatrace.com/injection=true,field.cattle.io/projectId=p-2617c,kubernetes.io/metadata.name=alert
dynatrace.com/inject=true,dynatrace.com/injection=true,field.cattle.io/projectId=p-t52dg,kubernetes.io/metadata.name=apicur
dynatrace.com/inject=true,dynatrace.com/injection=true,field.cattle.io/projectId=p-2617c,kubernetes.io/metadata.name=argocd
app.kubernetes.io/instance=rancher-monitoring,app.kubernetes.io/managed-by=Helm,app.kubernetes.io/part-of=rancher-monitorin
dynatrace.com/inject=true,dynatrace.com/injection=true,field.cattle.io/projectId=p-fg4fn,heritage=Helm,kubernetes.io/metadat

dynatrace.com/inject=true,dynatrace.com/injection=true,kubernetes.io/metadata.name=dynatrace
dynatrace.com/inject=true,dynatrace.com/injection=true,field.cattle.io/projectId=p-2617c,kubernetes.io/metadata.name=elasti
dynatrace.com/inject=true,dynatrace.com/injection=true,field.cattle.io/projectId=p-2617c,kubernetes.io/metadata.name=elasti
dynatrace.com/inject=true,dynatrace.com/injection=true,field.cattle.io/projectId=p-fg4fn,kubernetes.io/metadata.name=ingres
dynatrace.com/inject=true,dynatrace.com/injection=true,field.cattle.io/projectId=p-t52dg,kubernetes.io/metadata.name=kafka,
dynatrace.com/inject=true,dynatrace.com/injection=true,field.cattle.io/projectId=p-fg4fn,kubernetes.io/metadata.name=local
```

Deep Monitoring

⚠ **Um ponto importante não esqueça de habilitar o "Deep Monitoring" na console UI, se faz necessário para que funcione os traces distribuídos**