BRK Ambiental

SAN5

Arquitetura de Infraestrutura

Versão 1.2

21/08/2020

Índice

[1. Controle do Documento 3](#_Toc48822388)

[2. Objetivo 4](#_Toc48822389)

[3. Arquitetura da Nuvem 4](#_Toc48822390)

[3.1 Visão Geral 4](#_Toc48822391)

[3.2 Avaliação de Arquitetura para o SAN5 4](#_Toc48822392)

[3.3 Arquitetura de Infraestrutura para o SAN5 6](#_Toc48822393)

[4. Detalhamento da Arquitetura do Ambiente de Nuvem 7](#_Toc48822394)

[4.1 Visão Geral 7](#_Toc48822395)

[4.2 Descrição da Vnet 8](#_Toc48822396)

[4.3 Descrição do AKS 9](#_Toc48822397)

[4.4 Descrição do SQL Server 13](#_Toc48822398)

[4.5 Descrição do Azure Container Instance 13](#_Toc48822399)

[4.6 Descrição do Log Analytics Workspace 14](#_Toc48822400)

[4.7 Descrição do API Manager 14](#_Toc48822401)

[5 Pré-requisitos para scriptação do ambiente 14](#_Toc48822402)

[6 Provisionamento do Ambiente de Nuvem 16](#_Toc48822403)

[6.1 Pré-requisitos 16](#_Toc48822404)

[6.2 Variáveis do Ambiente 16](#_Toc48822405)

[6.3 Estrutura Azure 17](#_Toc48822406)

[6.4 Estrutura Kubernetes 17](#_Toc48822407)

[6.5 Provisionamento de Componentes Kubernetes 19](#_Toc48822408)

[6.6 Ativar Cluster AKS 20](#_Toc48822409)

[6.7 Instalação Istio 21](#_Toc48822410)

[6.8 Instalação Elastic 21](#_Toc48822411)

[6.9 Instalação Fluentd 22](#_Toc48822412)

[6.10 Instalação Kafka 23](#_Toc48822413)

[6.11 Instalação SQL Manage Instance 23](#_Toc48822414)

[6.12 Instalação Dashboard Confluentd no Grafana 24](#_Toc48822415)

[6.12 Instalação do api gateway 25](#_Toc48822416)

[7 Throubleshooting 25](#_Toc48822417)

[7.1 Azure 25](#_Toc48822418)

[7.2 Kubernetes 26](#_Toc48822419)

[8 Melhores Práticas do Terraform 26](#_Toc48822420)

[9 Perguntas frequentes (FAQ) 28](#_Toc48822421)

[10 Anexos 29](#_Toc48822422)

# Controle do Documento

|  |  |  |  |
| --- | --- | --- | --- |
| Versão | Data | Responsável | Alteração |
| 1.0 | 17/08/2020 | André Ribas  Denis Gozzi  Paulo Teixeira | Versão Inicial |
| 1.1 | 18/08/2020 | André Pinheiro Ribas | Revisão dos passos do ansible |
| 1.2 | 21/08/2020 | Paulo Teixeira | Adição de decisões de arquitetura |

# Objetivo

O objetivo deste documento é descrever a infraestrutura do ambiente SAN5, que correponde ao sistema core da BRK, contemplando diversas funções de negócio.

Anteriormente ao projeto, foi realizada uma Prova de Arquitetura (POA – Proof of Architecture), visando demostrar o uso de uma arquitetura de microserviços para o SAN, denominada SAN5, visando prover o desacoplamento das funções realizadas pela versão atual do SAN.

Neste documento é descrita a arquitetura de infraestrutura, as decisões que levaram à esta arquitetura e o processo de automação do provisionamento da infraestrutura, com base em ferramentas Terraform e Ansible, em um modelo de Infrastructure as Code – IaC).

# Arquitetura da Nuvem

A arquitetura acordada para o ambiente SAN5 foi ade um modelo híbrido, no qual uma instância do SAN5 em produção possa contemplar uma ou mais unidades de negócio da BRK Ambiental, de forma a compartilhar recursos e otimizar os custos de TI, ao mesmo tempo e que atende às necessidades de negócio.

Para o ambiente desenvolvimento será considerada uma instalação compartilhada para todos os ambientes produtivos, assim como o ambiente de testes. Isto se deve ao fato de que o código será o mesmo, independente da instância do ambiente produtivo.

* 1. Visão Geral

Ao longo do projeto, a partir de apresentação das equipes de infraestrutura e arquitetura da BRK e discussões sobre possíveis cenários para a arquitetura de infraestrutura para o SAN5.

Dado que o SAN não opera como um único sistema para todas as unidades de negócio da BRK, mas uma instância para cada unidade, discutiu-se qual dos modelos abaixo seria o mais apropriado para o SAN5:

1. Modelo Integrado, com uma única instância para todas as unidades de negócio;
2. Modelo Distribuído, com cada unidade de negócio com uma instância do SAN5, como acontece no modelo atual do SAN (SAN4)
3. Modelo Híbrido, com instâncias dedicadas do SAN5 para determinadas unidades de negócio e compartilhadas para outras unidades de negócio, mas mantendo independência dos dados

Os modelos Distribuído e Híbrido consideram o compartilhamento dos componentes de infraestrutura (ex. computação), mas consideram a separação dos dados, visando com isto manter, entre outros:

1. a segurança dos dados;
2. independência entre as unidades de negócio, mas com o compartilhamento de infraestrutura;
3. permitir o retorno dos dados ao poder concedente, ao término e/ou transferência de uma concessão.
   1. Avaliação de Arquitetura para o SAN5

Para se definir a arquitetura do SAN5, foram realizadas duas atividades para avaliar qual dos modelos seria o mais apropriado:

1. Avaliação através de aderência a requisitos técnicos
2. Comparação com softwares de mercado

**Avaliação através de aderência a requisitos técnicos**

Os times de arquitetura da BRK e da Accenture avaliaram os três modelos de arquitetura com base em uma avaliação de quesitos técnicos, com as seguintes regras:

1. Para cada quesito deveria ser atribuída uma nota de um a três para cada um dos três modelos de arquitetura (Integrado, Híbrido e Dedicado);
2. A maior nota (três) foi atribuída àquela arquitetura que melhor atende o quesito técnico;
3. Não poderia haver uma mesma nota para mais de uma arquitetura; sem empates;
4. Para um quesito que demonstrasse ser indiferente às arquitetura não haveria pontuação;
5. Não deveria ser inicialmente levado em consideração o quesito financeiro, para não influenciar a decisão técnica.

Os quesitos e as pontuações atribuídas a cada alternativa de arquitetura se encontram no item 10.

Seguindo esta avaliação, uma arquitetura integrada seria a melhor, com leve vantagem sobre a arquitetura híbrida.

**Comparação com softwares de mercado**

Foi realizada uma avaliação de alguns modelos de software como serviço (SaaS), de forma a tar uma base de referência para o SAN5, considerando alguns softwares de mercado, como Microsoft Dynamics, Salesforce, Service Now:

**Microsoft Dynamics & ERPs**

Alguns CRMs de mercado trabalham com arquiteturas compartilhadas, para pequenos clientes, e dedicadas para clientes maiores. Neste caso o modelo se aproxima do modelo híbrido.

Este [link](https://crmswitch.com/buying-crm/single-tenant-crm-vs-multi-tenant-crm/) contém uma referência sobre “tenancy” de ERPs, com alguns comentários sobre o Microsoft Dynamics, no qual a camada de aplicação é compartilhada e os bancos de dados segregados, em uma linha mais próxima do modelo híbrido.

**Service Now**

Basicamente o Service Now trabalha como uma solução multi-instance (“each customer gets its own database and its own front-end app tier”) com a infraestrutura compartilhada entre os clientes.

Este [link](https://mariadb.com/resources/customer-stories/servicenow/) do site do banco de dados MariaDB contém maiores informações sobre a arquitetura do Service Now.

Trata-se também de uma arquitetura híbrida.

**Salesforce**

Este [link](https://status.salesforce.com/products/Sales_Cloud) contém informações sobre o modelo de implementação do Salesforce. Ele trabalha com um arquitetura descrita como multi-tenant, com várias instâncias por tenant.

Assim sendo também se aproxima de um modelo híbrido.

* 1. Arquitetura de Infraestrutura para o SAN5

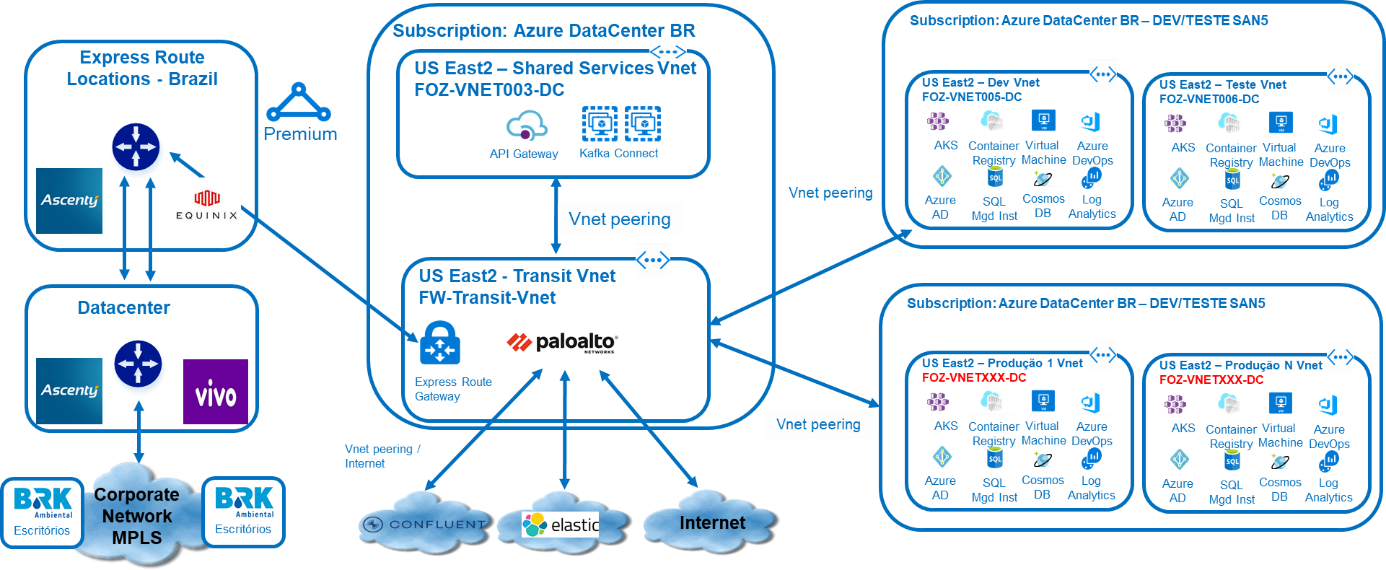
**Modelo Escolhido - Arquitetura híbrida**

A decisão foi de uso de uma arquitetura híbrida para os ambientes de produção do SAN5, considerando que:

1. Permite maior versatilidade e a um custo previsível;
2. A diferença de pontuação entre os modelos híbrido e integrado não foi relevante;
3. Permite otimizar os custos para concessões de variados tamanhos, sem ter uma proliferação grande de instâncias, como acontece com o SAN4.
4. Permite um isolamento do ambiente, visando atender requisitos especfícos de uma concessão;
5. A conversão de um modelo hibrido para dedicado pode ser atingida com relativa velocidade, considerando que foi concebido para operar em um modelo de Infraestructure as Code.

**Arquitetura Híbrida do SAN5**

Para cada instância SAN5 foi considerada uma vnet dedicada apenas para os componentes daquela instância. A figura 1 abaixo contém uma visão macro da arquitetura do SAN5



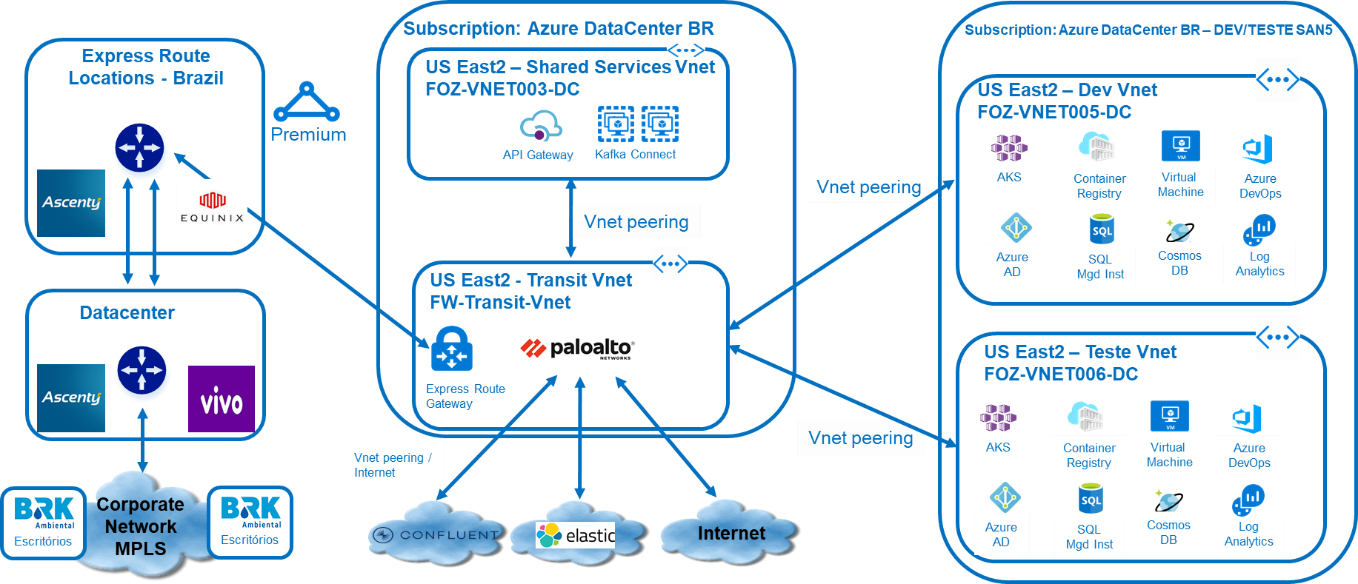
**Figura 1 – Arquitetura Macro do SAN5**

Os ambientes de desenvolvimento e teste foram criados em uma subscrição de Dev/Teste. Os ambientes de produção serão criados em uma subscrição específica para o SAN5, de forma a facilitar a separação de custos do SAN5 de outros consumos de nuvem da BRK.

# Detalhamento da Arquitetura do Ambiente de Nuvem

* 1. Visão Geral

A figura abaixo contém uma visão em alto nível da arquitetura e da conexão do ambiente de desenvolvimento e teste com a rede corporativa da BRK Ambiental. O ambiente de produção seguirá a mesma arquitetura macro.



**Figura 2 – Arquitetura do ambiente SAN5**

A BRK decidiu concentrar os seus ambientes de nuvem na região East US2 do Azure e seguindo o seu padrão de segurança, bem como recomendação da Accenture, que fosse adotada uma topologia Hub/Spoke com uma vnet de trânsito (hub) entre o ambiente corporativo e as aplicações (spokes).

Por definição da BRK, apenas o firewall Palo Alto, balanceadores de carga, e o gateway Express Route ficarão nesta vnet.

As aplicações serão agrupadas em outras vnets, conectadas à vnet de trânsito via “vnet peering”. Todas as vnets deverão possuir uma rota default apontando para um load-balance em uma rede de trust do Palo Alto. Este load-balance irá distribuir a carga entre um ou mais instâncias do Palo Alto, visto que ele segue a política de “scale out”.

O link [topologia de rede hub-spoke no Azure](https://docs.microsoft.com/pt-br/azure/architecture/reference-architectures/hybrid-networking/hub-spoke) contém maiores detalhes da topologia Hub/Spoke.

Foi definida uma subscrição Dev/Teste do Azure para hospedar tanto o ambiente de desenvolvimento quanto de teste. Isto permitirá uma redução de custos de licenciamento com produtos Microsoft como o banco de dados SQL Server.

Cada ambiente será criado em Resource Groups especifícos em cada ambiente: para desenvolvimento, teste e produção..

Os Resource Groups devem seguir o padrão RG-<aplicação>-<ambiente>. Assim para o ambiente de desenvolvimento do SAN5 foram criados os resource groups:

1. RG-REDE-SAN5-DEVQA, no qual foi criada a vnet para o ambiente de desenvolvimento e a tabela de rotas associada ao ambiente;
2. RG-SAN5-DEV, no qual foram criados os componentes do SAN5.

Como o ambiente será scriptado nas ferramentas Terraform e Ansible, é mandatório que a separação dos nomes seja feita via hífen (“-“) e não via ponto (“.”), de forma a evitar problemas na execução das scripts de provisionamento.

Os componentes Azure dedicados para cada instância do SAN5 serão instalados na vnet dedicada para a instância. Basicamente serão utilizados em cada instância SAN5:

1. Um cluster AKS (Azure Kubernete Services), para hospedagem dos microserviços.
2. Um cluster SQL Service Managed Instance, para o banco de dados dos Microserviços.
3. Um Log Analytics Workspace para armazenamento de logs e monitoração de componentes do Azure, de forma a permitir uma granularidade por ambiente. Tal log analytics pode ser configurado como um data source de um dashboard corporativo, como o Grafana.
4. Um Container Registry com as imagens dos containers com os microserviços.
   1. Descrição da Vnet

Para a Vnet de uma instância SAN5, foi considerado um IP Address Space /23, que permite 512 endereços TCP/IP no ambiente. Visto que os nós do AKS e do SQL server podem escalar verticalmente, decidiu-se manter um tamanho padrão o Address Space que permita crescimento dos ambientes de produção maiores, podendo cada cluster do AKS chegar a sete nós.

Para os serviços de DNS foi considerado os servidores do Active Directory/DNS informados pela BRK: 10.157.16.4 e 10.157.16.5.

A tabela abaixo contém a descrição da estrutura de endereços IP e sua alocação nas subnets a serem utilizadas no projeto:

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome da Subnet** | **Endereço** | **Máscara** | **Função** |
| subnet-aks-san5 | 10.161.0.0 | /24 | Cluster AKS |
| subnet-mgm-san5 | 10.161.1.0 | /26 | Managed Instance SQL |
| <reservado> | 10.161.1.64 | /26 | Reservado para uso futuro |
| <reservado> | 10.161.1.128 | /25 | Reservado para uso futuro |

Estes IPs consideram o address space utilizado no ambiente de desenvolvimento do SAN5 (10.161.0.0/22). Para a criação de outros ambientes deverá ser seguido o padrão de tamanho e ordem especificados acima, isto considerando os primeiros 256 endereços para o AKS e os 64 seguintes para o SQL server.

A vnet deverá ter uma rota default (0.0.0.0) apontando para o load balancer do Palo Alto na sua rede de trust, cujo endereço IP é 10.161.8.21.

Para a subnet do banco de dados SQL Server foi criado um Network Sercurity Group (NSG) chamado **nsg-misa5dev**, com regras padrão para acesso privado ao SQL e gestão pela plataforma Azure, restringindo acessos pela Internet. Todo o controle de inspeção e permissão de acesso entre ambientes ficou delegado para o Palo Alto.

Na mesma subnet do SQL Server foi criada um tabela de rotas chamada **rt-misan5dev**.

A planilha no item 10 contém a configuração da vnet de trânsito quando de sua instalação inicial e pode ser usada como referência da arquitetura implantada na vnet de trânsito. A arquitetura vigente deve ser verificada com as equipes de operação e suporte de rede e de nuvem.

Todos os componentes do ambiente receberam as seguintes tags:

1. Ambiente: DEV
2. Aplicação: SAN5
3. Arquiteto Responsável: Rafael Faita
4. Fornecedor: Accenture
5. Responsável: Marcelo Castanho

Para os recursos provisionados automaticamente pelo Terraform/Ansible, foi adicionada a tag “Deployed by”, com valor atribuído de “Terraform”, sinalizando a forma como foi feito o provisionamento do componente.

Além dos componentes criados no ambiente de desenvolvimento na vnet FOZ-VNET005-DC, dois componentes da solução foram provisionados em uma vnet de serviços compartilhados, denominada FOZ-VNET003-DC:

1. Dois servidores CentOS, para instalação da aplicação Kafka Connect, nomeados srvsan5dcr01 (10.161.4.132) e srvsan5dcr02 (10.161.4.133);
2. Um API Manager para o ambiente de desenvolvimento (10.161.4.199), denominado san5dev.

Esta vnet também se encontra na região US East2 do Azure, que deve operar pareada com a vnet de trânsito. Isto foi feito porque serão serviços compartilhados por mais de um ambiente (como o ambiente de teste) do SAN5 e eventualmente por outros projetos da BRK no futuro, como o Kafka Connect.

O projeto considera a publicação de APIs, inicialmente apenas internamente, sem exposição das mesmas para a Internet. O projeto pressupõe que o API manager deverá estar conectado à rede interna e possibilitar autenticação pelo Active Directory.

O API Manager para o ambiente de desenvolvimento foi instalado no tier **Developer**, visto os requerimentos acima. O mesmo deverá ser seguido para o ambiente de qualidade. Para o ambiente de produção deverá ser utilizado o tier **Premium**, adequado para ambientes produtivos com estes requerimentos.

Para uma validação do uso de “secrets” para o SAN5, foi criado um novo Key Vault para o projeto na subscrição “Azure Datacenter BR” na região East US2.

* 1. Descrição do AKS

Para a implementação do AKS no ambiente de desenvolvimento do SAN5 foi considerado o uso de nós B8ms (burstable, 8 vCPUs, 32GB RAM) com sistema operacional Linux, o mesmo modelo utilizado na prova de arquitetura.

Por se tratar de um ambiente de desenvolvimento, considerou-se que estas máquinas (B8ms) seriam suficientes para as primeiras iterações do projeto e mais baratas que as inclusas na proposta (D16v3).

Considerando o tamanho da subnet criada para o cluster AKS e a maneira como o mesmo foi configurado, cada cluster pode escalar até 7 nós no cluster. Dado às definições de arquitetura híbrida do SAN5 e escalabilidade vertical dos nós do cluster Kubernetes, considerando nós de 16-cores (série Dv3) pode-se chegar a um ambiente 4 vezes a capacidade inicial do cluster com três nós de 8-cores (série B), sem consideradar o aumento de desempenho entre as séries, visto a série B trabalhar com créditos de CPU (“bursts”).

Esta capacidade pode aumentar com máquinas mais novas, já lançadas ou que serão lançadas no futuro, como a Dsv4, que provê 50% capacidade adicional de processamento por vCPU pelo mesmo preço.

Para o início da operação do ambiente a seguinte configuração é recomendada:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ambiente** | **Nós** | **Configuração** | **Quantidade** | **Região** |
| Desenvolvimento | B8ms | 8 vCPU / 32GB RAM | 3 | East US2 |
| Qualidade | B8ms | 8 vCPU / 32GB RAM | 3 | East US2 |
| Produção | D8sv4 | 8 vCPU / 32GB RAM | 3 | East US2 |

Itens importantes levados em consideração:

1. Na configuração inicial do cluster foi considerada uma quantidade mínima de nós para suportar o ambiente com confiabilidade e redundância.
2. Foi considerado apenas um node pool, com sistema operacional Linux.
3. Não foi considerada a utilização de capacidade adicional via Azure Container Instance (ACI).
4. Foi utilizada a versão 1.16.10 do Kubernetes.
5. Foi considerado o uso de scale sets.
6. Foi considerada autenticação via Managed Identity
7. O cluster AKS foi provisionado em uma Vnet pre-existente e informada a subnet subnet-aks-san5 para uso como subnet do AKS.
8. Foi considerado um cluster privado para assegurar que o tráfego entre a API do AKS permaneça privado.

Para a implementação e gestão da arquitetura de Microserviços baseada em eventos, uma série de produtos foi instalada:

1. Istio: para gestão do service mesh e inclui outros produtos como Grafana (visualização e dashboard) e Prometheus (registro de métricas de microserviços)
2. Kiali: console de gerenciamento do service mesh (Istio)
3. Fluentd: coletor de dados para seu armazenamento em um agregador de logs
4. Jagger: para trace dos microserviços
5. ELK (Elastic, Logstash, Kibana): agregador de logs, instalado provisoriamente no AKS de desenvolvimento enquanto a licença definitiva (SaaS – Software as a Service) não é adquirida pela BRK.
6. Confluent Kafka: plataforma de streaming de eventos, instalada provisoriamente enquanto a licença definitiva não é adquitida pela BRK. Inclui uma série de produtos: Zookeeper, Ksql, Rest Proxy, Schema Registry, Kafka Connect

Cada produto foi instalado em um namespace diferente (“cluster virtual”), permitindo um maior isolamento dos componentes, e com isto facilitar a gestão dos mesmos.

O ambiente de desenvolvimento foi instalado com três nós:

kubectl get nodes

NAME STATUS ROLES AGE VERSION

aks-agentpool-42752616-vmss00000s Ready agent 12d v1.16.10

aks-agentpool-42752616-vmss00000u Ready agent 12d v1.16.10

aks-agentpool-42752616-vmss00000v Ready agent 12d v1.16.10

A instalação considera dois namespaces: **kube-system** e **kubernetes-dashboard**.

$kubectl get pods --namespace=kube-system

NAME READY STATUS RESTARTS AGE

azure-cni-networkmonitor-kglcs 1/1 Running 0 12d

azure-cni-networkmonitor-vhsv6 1/1 Running 0 12d

azure-cni-networkmonitor-w7l26 1/1 Running 0 12d

azure-ip-masq-agent-2wzlp 1/1 Running 0 12d

azure-ip-masq-agent-s4s59 1/1 Running 0 12d

azure-ip-masq-agent-vpg7t 1/1 Running 0 12d

coredns-869cb84759-7smv9 1/1 Running 0 6d3h

coredns-869cb84759-hwt6n 1/1 Running 0 6d3h

coredns-autoscaler-5b867494f-gxrs5 1/1 Running 0 6d3h

dashboard-metrics-scraper-566c858889-gbwcl 1/1 Running 0 6d3h

kube-proxy-8mqvf 1/1 Running 0 12d

kube-proxy-p456d 1/1 Running 0 12d

kube-proxy-rv2rq 1/1 Running 0 12d

kubernetes-dashboard-7f7d6bbd7f-mlvmw 1/1 Running 0 6d3h

metrics-server-6cd7558856-wzb25 1/1 Running 0 6d3h

omsagent-rgjm9 1/1 Running 0 6d3h

omsagent-rs-655447648b-58hmk 1/1 Running 0 6d3h

omsagent-wr6sw 1/1 Running 0 6d3h

omsagent-z8gfc 1/1 Running 0 6d3h

tiller-deploy-6974685dbc-q95qt 1/1 Running 0 12d

tunnelfront-6c6f49c64b-pvstp 2/2 Running 0 6d3h

$ kubectl get pods --namespace=kubernetes-dashboard

NAME READY STATUS RESTARTS AGE

dashboard-metrics-scraper-c79c65bb7-spr9t 1/1 Running 0 5d4h

kubernetes-dashboard-56484d4c5-zms9t 1/1 Running 0 5d4h

**Istio**

Como todos os componentes da solução, o Istio foi instalado em um namespace específico, denominado **istio-system**. A instalação do Istio inclui também o Grafana, Jaeger, Kiali e Prometheus.

$ kubectl get pod --namespace=istio-system

NAME READY STATUS RESTARTS AGE

grafana-86897cb4f5-g7p8k 1/1 Running 0 2d1h

istio-ingressgateway-6d6589d6f9-8dpdp 1/1 Running 0 2d1h

istio-tracing-57d7cfd779-ntv8c 1/1 Running 0 2d1h

istiod-7697c784c4-mpx7h 1/1 Running 0 2d1h

kiali-7c974669b4-6zqtt 1/1 Running 0 2d1h

prometheus-5587d8757f-tlmw2 2/2 Running 0 2d1h

Os serviços destes vários componentes foram disponibilizados através de endereços internos da vnet de desenvolvimento do SAN5.

kubectl get svc --namespace=istio-system

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

grafana ClusterIP 10.0.240.129 <none> 3000/TCP 2d2h

istio-ingressgateway LoadBalancer 10.0.100.40 10.161.0.99 15021:32182/TCP,80:30683/TCP,443:31229/TCP,15443:30819/TCP 2d2h

istiod ClusterIP 10.0.175.163 <none> 15010/TCP,15012/TCP,443/TCP,15014/TCP,853/TCP 2d2h

jaeger-agent ClusterIP None <none> 5775/UDP,6831/UDP,6832/UDP 2d2h

jaeger-collector ClusterIP 10.0.58.6 <none> 14267/TCP,14268/TCP,14250/TCP 2d2h

jaeger-collector-headless ClusterIP None <none> 14250/TCP 2d2h

jaeger-query ClusterIP 10.0.250.49 <none> 16686/TCP 2d2h

kiali ClusterIP 10.0.197.174 <none> 20001/TCP 2d2h

prometheus ClusterIP 10.0.98.20 <none> 9090/TCP 2d2h

tracing ClusterIP 10.0.108.211 <none> 80/TCP 2d2h

zipkin ClusterIP 10.0.75.217 <none> 9411/TCP 2d2h

**Elastic Search**

A solução considera o Elastic Search como agregador de logs em solução SaaS. Ele foi instalado interinamente no AKS do ambiente de desenvolvimento do SAN5, visto que a licença definitiva (SaaS) ainda não havia sido adquirida, de forma a permitir o início do desenvolvimento. Sua instalação foi feita em um namespace denominado **elastic** e sua instalação também inclui o fluentd e o Kibana.

Foi instalada a versão Basic do Elastic Search, disponível sem custo de licenciamento mas com limitação em relação aos produtos licenciados, como autenticação de usuários do Kibana via Active Directory que não se encontra disponível na versão Basic (vide <https://www.elastic.co/subscriptions>).

Foi considerado um pod de cada serviço para cada um dos nós Kubernetes instalado.

kubectl get pod --namespace=elastic | more

NAME READY STATUS RESTARTS AGE

elasticsearch-bpgna-test 0/1 Error 0 4d22h

elasticsearch-master-0 1/1 Running 0 4d22h

elasticsearch-master-1 1/1 Running 0 4d22h

elasticsearch-master-2 1/1 Running 0 4d22h

fluentd-lf8t5 1/1 Running 0 4d22h

fluentd-pl6bp 1/1 Running 0 4d22h

fluentd-zpc2b 1/1 Running 0 4d22h

kibana-kibana-774fb445d6-2zr4c 1/1 Running 0 4d22h

kibana-kibana-774fb445d6-7vjvs 1/1 Running 0 4d22h

kibana-kibana-774fb445d6-vz5fn 1/1 Running 0 4d22h

kubectl get svc --namespace=elastic

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

elasticsearch-master LoadBalancer 10.0.70.202 10.161.0.97 9200:32119/TCP,9300:32259/TCP 4d23h

elasticsearch-master-headless ClusterIP None <none> 9200/TCP,9300/TCP 4d23h

kibana-kibana LoadBalancer 10.0.78.140 10.161.0.98 5601:30010/TCP 4d23h

Ele deverá ser descontinuado quando da finalização do licenciamento (ELK), e estiver disponível para acesso pela Internet através da vnet de trânsito. Por definição foi solicitado que todo o acesso Internet a partir do ambiente de teste seja liberado no Palo Alto, de forma a permitir este acesso.

**Kafka**

De forma similar ao Elastic Search, o Kafka foi instalado provisoriamente no ambiente de desenvolvimento até a disponibilização da licença definitiva (SaaS).

Foi instalada a versão open source da Confluent que contém os seguintes componentes:

* Control Center (gestão do Kafka)
* Kafka
* Kafka Connect
* Kafka Rest Proxy
* KSQL
* Kafka Schema Registry
* Zookeper

O [link](https://docs.confluent.io/current/control-center/installation/licenses.html) descreve o licenciamento dos produtos instalados, sendo que o Control Center disponível por apenas trinta dias.

O Kafka foi instalado em um namespace dedicado, denominado **kafka**.

kubectl get pod --namespace=kafka

NAME READY STATUS RESTARTS AGE

confluent-oss-cp-control-center-76dc4c585d-z6khh 1/1 Running 5 5d6h

confluent-oss-cp-kafka-0 2/2 Running 0 5d6h

confluent-oss-cp-kafka-1 2/2 Running 0 5d6h

confluent-oss-cp-kafka-2 2/2 Running 0 5d6h

confluent-oss-cp-kafka-connect-58699887c4-mnxhx 2/2 Running 5 5d6h

confluent-oss-cp-kafka-rest-5654b6ddc6-6gv5m 2/2 Running 3 5d6h

confluent-oss-cp-ksql-server-57dfcfd9d6-86wwd 2/2 Running 5 5d6h

confluent-oss-cp-schema-registry-94786c874-vdzpx 2/2 Running 5 5d6h

confluent-oss-cp-zookeeper-0 2/2 Running 0 5d6h

confluent-oss-cp-zookeeper-1 2/2 Running 0 5d6h

confluent-oss-cp-zookeeper-2 2/2 Running 0 5d6h

A lista de serviços ativados no namespace do Kafka se encontra abaixo.

kubectl get svc --namespace=kafka

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

confluent-oss-cp-control-center ClusterIP 10.0.208.13 <none> 9021/TCP 5d6h

confluent-oss-cp-control-center-balancer LoadBalancer 10.0.165.7 52.247.35.208 9021:30532/TCP 5d4h

confluent-oss-cp-kafka ClusterIP 10.0.96.53 <none> 9092/TCP 5d6h

confluent-oss-cp-kafka-connect ClusterIP 10.0.156.26 <none> 8083/TCP 5d6h

confluent-oss-cp-kafka-headless ClusterIP None <none> 9092/TCP 5d6h

confluent-oss-cp-kafka-rest ClusterIP 10.0.45.105 <none> 8082/TCP 5d6h

confluent-oss-cp-ksql-server ClusterIP 10.0.8.231 <none> 8088/TCP 5d6h

confluent-oss-cp-schema-registry ClusterIP 10.0.14.55 <none> 8081/TCP 5d6h

confluent-oss-cp-zookeeper ClusterIP 10.0.68.153 <none> 2181/TCP 5d6h

confluent-oss-cp-zookeeper-headless ClusterIP None <none> 2888/TCP,3888/TCP 5d6h

* 1. Descrição do SQL Server

Para o banco de dados dos microserviços foi considerado o SQL server em modelo Paas (Platform as a Service), no modelo managed instance.

Para o ambiente de desenvolvimento foi provisionado a menor instândia disponível neste modelo (4 vCPUs e 32GB de RAM), com o nome de **misan5dev**. O administrador do banco de dados foi denominado **misan5adm**.

Para ambientes não produtivos, isto é desenvolvimento e qualidade, deve-se utilizar o modelo General Purpose. Para o ambiente de produção será utilizado o modelo Business Critical, por prover um SLA de 99.99% e prover maior desempenho que o modelo General Purpose.

Por questões de segurança foi desabilitado o acesso do banco de dados pela Internet e o uso de TLS na versão 1.2.

No projeto não foi considerado o uso da feature adicional de segurança do SQL na nuvem. Ela poderá ser adicionada, no futuro em conjunto com o Azure Security Center. Maiores detalhes sobre esta funcionalidade pode ser encontrada neste [link](https://docs.microsoft.com/en-us/azure/azure-sql/database/advanced-data-security).

* 1. Descrição do Azure Container Instance

Foi criado um Container Regitry denominado **crsan5dev** no resource group de desenvolvimento, no tier Standard, como considerado na proposta.

* 1. Descrição do Log Analytics Workspace

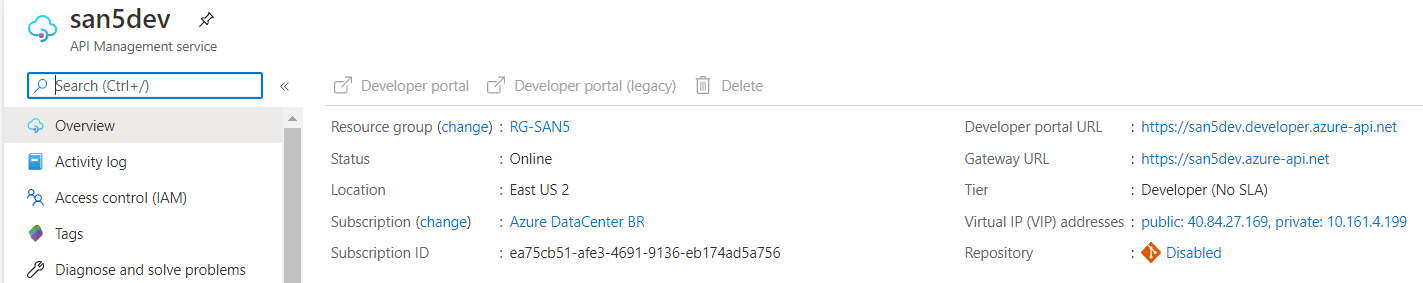
Foi criado um workspace do Log Analytics chamado **la-san5-dev** para receber os logs dos recursos Azure do ambiente de desenvolvimento.

Este log analytics pode ser adicionado manualmente como um datasource do Grafana instalado na vnet FOZ-VNET003-DC, de serviços compartilhados.

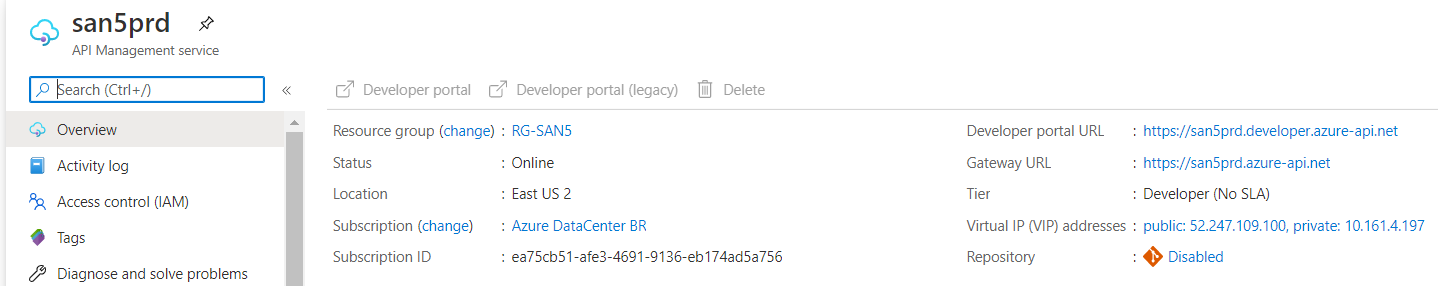
* 1. Descrição do API Manager

Foram criados dois API Manager na subscrição de Produção. O primeiro deles é para Desenvolvimento e o segundo para produção. Eles tem as seguintes caracteristicas:

Desenvolvimento:



Produção:



Ambos foram conectados na subnet “SUBNET-APIM-HUB-USEAST2" devido a sua intensa comunicação com o ambiente da BRK.

Para os ambientes de desenvolvimento e teste deve ser utilizado o tier Developer; para produção o tier Premium, dado o requerimento de conexão do API Manager à vnet e autenticação pelo Active Directory.

# Pré-requisitos para scriptação do ambiente

Para a correta operação do ambiente, uma série de pré-requisitos deve ser atendido nas máquinas do time de operação e suporte de nuvem da BRK.

1. Instalar a interaface de linha de comando do Azure, [Azure CLI](https://docs.microsoft.com/en-us/cli/azure/install-azure-cli?view=azure-cli-latest)
2. Instalar o comando [kubectl](https://kubernetes.io/docs/tasks/tools/install-kubectl/), para gestão e operação do Kubernetes;
3. Instalar o [Terraform](https://terraform.io/downloads.html) (versão mínima 0.12), para permitir a criação de scripts para provisionamento de infraestrutura, no modelo IaaC (Infrastructure as a Code);
4. Utilizar o [Azure Provider](https://www.terraform.io/docs/providers/azurerm/index.html),(versão mínima: 1.36.1) para permitir o provisionamento de recursos no Azure pelo Terraform.
5. Instalar o Ansible (versão 2.8) para permitir o provisionamento de componentes Azure e do AKS.
6. Instalar o Helm (versão 3.0) para instalação de componentes no AKS (<https://helm.sh/docs/intro/install/>).
7. Instalar o Istioctl (versão 1.6.5) para instalação do Istio e seus componentes (<https://github.com/istio/istio/releases/tag/1.6.5>).
8. Possuir um usuário com perfil de **Owner** na subscrição/resource groups do ambiente SAN5 e na vnet de trânsito em US East2.
9. Para este modelo, recomendamos o uso do WSL Ubuntu (<https://docs.microsoft.com/pt-br/windows/wsl/install-win10>)

# Provisionamento do Ambiente de Nuvem

6.1 Pré-requisitos

* Efetuar login através do Azure CLI (az login)
* Utilizar o ansible com módulo azure (Em caso do CentOs, utilize o comando pip3)
  + Comandos : **pip install setuptools**
  + Comandos: **pip install ansible[azure**].
* Validar a subscrição do ambiente (az account set -s xxxx)
* Download do repositório scripts: <https://dev.azure.com/brk-vsts/Sistemas%20Comerciais%20-%20Infraestrutura%20TI/_git/Infraestrutura-TI?version=GBmaster>

6.2 Variáveis do Ambiente

Abaixo segue as informações sobre o arquivo **terraform.tfvars**. que deve ser editado antes de sua execução:

|  |  |
| --- | --- |
| **Variavel** | **Valor** |
| add\_space | Address Space que será usado na Azure  Padrão: 10.x.y.0/23 (obter com o time de rede) |
| aksname | Nome do cluster AKS  Padrão: aks-san5-<dev/qa/prdn>  n corresponde ao número do ambiente de produção, a partir de 1 |
| cr\_name | Nome do Container Registry  Padrão: crsan5<dev-qa-prdn>  n corresponde ao número do ambiente de produção, a partir de 1 |
| dnsprefix | Prefixo do DNS  Padrão:aks-san-<dev-qa-prdX>-dns  X corresponde ao número do ambiente de produção, a partir de 1 |
| dnsp | DNS Primário  Padrão: 10.157.16.4  Caso a vnet esteja isolada, não tiver acesso ao DNS acima, utilizar o DNS do Google 8.8.8.8, caso contrário haverá problema no provisionamento do AKS |
| dnss | DNS Secundário  Padrão: 10.157.16.5  Caso a vnet esteja isolada, não tiver acesso ao DNS acima, utilizar o DNS do Google 8.8.4.4, caso contrário haverá problema no provisionamento do AKS |
| env\_proj | Qual o ambiente do projeto  Padrão: DEV/QA/PRDn |
| loadbalance | SKU Load Balance  Padrão: Standard |
| location | Location para a criação no Azure  Padrão: eastus2 |
| nodecount | Total de node para o AKS  Padrão: 3 |
| nodesize | Size para o nodes AKS  Padrão:  Para dev e qa: Standard\_B8ms  Para produção: Standard\_D8as\_v4 |
| nome\_vnet | Nome da VNET  Padrão: AZR-VNET<nnn>-DC  Verificar número sequencial da Vnet (nnn) |
| nsg\_name | NSG para a VNET  Padrão: nsg-misan5<dev/qa/prodn> |
| project | Nome do projeto (Sera usado para definir nomes do RG):  Padrão: SAN5 |
| rg\_main | Nome do Resource Group do AKS (Formado de rg\_main+project+env\_proj,exemplo RG-SAN5-QA)  Padrão: RG |
| rg\_vnet | Nome do Resource Group da Azure (Formado de rg\_vnet+project+env\_proj,exemplo RG-REDE-SAN5-QA)  Padrão: RG-REDE |
| rt\_mgm | Nome do Router Table para subnet MGM  Padrão: rt-misan5<dev/qa/prodn> |
| subnet\_aks\_name | Nome da Subnet AKS  Padrão: subnet-aks-san5 |
| subnet\_aks | Range da Subnet AKS  Padrão: 10.x.y.0/24 (primeiros 256 endereços do Address Space). Vide item 4 |
| subnet\_mgm\_name | Nome da Subnet Mgm  subnet-mgm-san5 |
| subnet\_mgm | Range da Subnet Mgm  Padrão: 10.x.y.0/26 (próximos 64 endereços do Address Space). Vide item 4 |
| tag\_ambiente | Valor para TAG do Ambiente:  Padrão: <DEV/QA/PRODn> onde n corresponde ao número da instância de produção do SAN5 |
| tag\_app | Valor para TAG “Aplicacao”  Padrão: SAN5 |
| tag\_arquiteto | Valor para TAG “Arquiteto Responsável”  Padrão: Rafael Faita |
| tag\_fornecedor | Valor para TAG “Fornecedor”  Padrão: Accenture |
| tag\_respo | Valor para TAG “Responsavel”  Padrão: Marcelo Castanho |
| workspace\_name | Nome do Woskpace (Formado de workspace\_name+project+env\_proj,exemplo Workspace-Padrão: Workspace |

6.3 Estrutura Azure

Tempo aproximado de execução: 2 minutos

Execute os passos abaixo para implementar a estrutura principal:

1. Edite as variáveis em var\_terraform/terraform.tfvars de acordo com os valores em VARIAVEIS DO AMBIENTE.
2. Acesse a pasta 01\_Vnet da estrutura.
3. Execute o comando *terraform init -var-file=../var\_terraform/terraform.tfvars*.   
     
   Execute o comando *terraform plan -var-file=../var\_terraform/terraform.tfvars*.
4. Execute o comando: *terraform apply -var-file=../var\_terraform/terraform.tfvars*. Após executar este comando, irá exibir a mensagem abaixo:
5. Digite yes e tecle <ENTER>. Após finalizar, irá exibir a imagem abaixo:

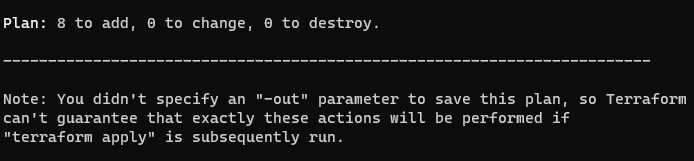


6.4 Estrutura Kubernetes

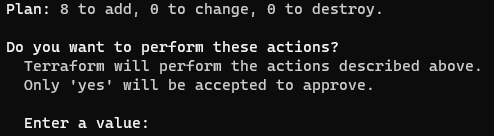
Tempo aproximado de execução: 7 minutos

Execute os passos abaixo para implementar a estrutura principal:

1. Edite as variáveis em /var\_terraform/terraform.tfvars de acordo com os valores em VARIAVEIS DO AMBIENTE.
2. Acesse a pasta 02\_Kubernetes da estrutura.
3. Execute o comando *terraform init -var-file=../var\_terraform/terraform.tfvars*.
4. Execute o comando *terraform plan -var-file=../var\_terraform/terraform.tfvars*. Será exibida a mensagem abaixo:



1. Execute o comando : terraform apply -var-file=../var\_terraform/terraform.tfvars. Após executar este comando, irá exibir a mensagem abaixo:



1. Digite yes e tecle <ENTER>. Após finalizar, irá exibir a imagem abaixo:



Observação: é importante guardar o arquivo de estado após a execução dos scripts de terraform. O arquivo de estado é um tipo de inventário e tem que identico ao conteudo da Cloud. Caso ocorra discrepância entre o arquivo de estado e os recursos provisionados poderá ocorrer alteração e/ou exclusão de componentes de forma indevida

6.5 Provisionamento de Componentes Kubernetes

Variáveis do Ambiente ANSIBLE

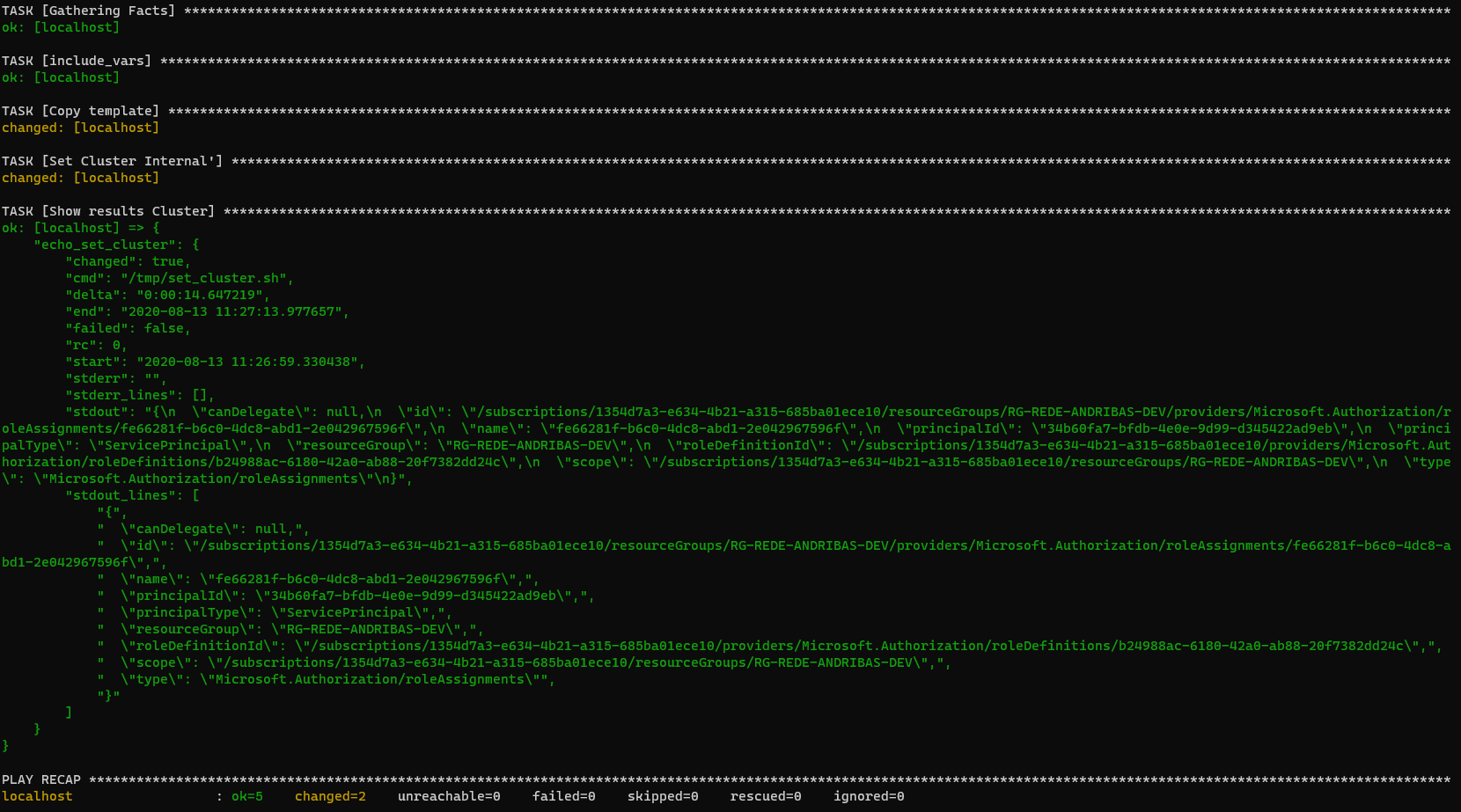
Abaixo segue as informações sobre o arquivo **var\_ansible/vars.yml**. que deve ser editado antes de sua execução:

|  |  |
| --- | --- |
| **Variavel** | **Valor** |
| rg\_main | Resource Group do AKS  Como criado pelo terraform no item 6.2 Variáveis do Ambiente  Padrão: RG-SAN5-<DEV/QA/PRODn> |
| rg\_vnet | Resource Group da Rede  Como criado pelo terraform no item 6.2 Variáveis do Ambiente  Padrão: RG-REDE-SAN5-<DEV/QA/PRODn> |
| cluster\_name | Nome do Cluster AKS  Padrão: aks-san5-<dev/qa/prodn> |
| replicas | Qtd replicas do AKS  Padrão: 3 |
| path\_scripts | Diretorio de download scripts  Padrão: /tmp |
| version\_istio | Versao do Istio  Padrão: 1.6.5  Versão utilizada durante o setup do ambiente. |
| user\_kiali | usuario kiali  Padrão: kiali |
| pass\_kiali | senha user kiali  Padrão: kialiSan5  Alterar manualmente após a instalação |
| namespace\_kiali | Namespace para o Kiali  Padrão: istio-system |
| install\_kiali | Instalar Kiali (false para nao)  Padrão: true |
| user\_grafana | usuario grafana  Padrão: grafana |
| pass\_grafana | senha user grafana  Padrão: grafanaSan5  Alterar manualmente após a instalação |
| namespace\_grafana | Namespace para o Grafana  Padrão: istio-system |
| install\_grafana | Instalar Grafana (false para nao)  Padrão: true |
| namespace\_elastic | Namespace para o ELASTIC  Padrão: elastic  Deverá ser desinstalado após a aquisição do ELK em modo SaaS |
| namespace\_fluentd | Namespace para o Fluentd  Padrão: fluentd  Deverá ser reconfigurado após a aquisição do ELK em modo SaaS |
| namespace\_kafka | Namespace para o Kafka  Padrão: Kafka  Deverá ser desinstalado após a aquisição do Confluent Cloud em modo SaaS |
| path\_tmp | Diretorio download yml  Padrão: /tmp |
| size\_kafka | Tamanho para o node Kafka (em GB)  Padrão: 50GB |
| location | Location do Azure  Padrão: eastus2 |
| managed\_instance | Nome para Manage Instance  Padrão: misan5<dev/qa/prodn> |
| tier\_value | Valor do Tier  Padrão General Purpose para dev e qa Business Critical para produção |
| provider | Banco Provider  Padrão: sql |
| name\_Tier | Nome do Tier  Padrão: GP\_Gen5 para dev e QA e BC\_Gen5 para produção |
| vCores | valor do vCore  Padrão: 4 |
| storageSizeInGB | Size do Storage SQL (GB)  Padrão: 32 |
| useradmin | user admin do banco  Padrão: misan5adm |
| passadmin | passwd admin do banco  Padrão: Pa$$w0rd12345678  Alterar manualmente após a instalação |
| vnet | Nome da VNET  Padrão: AZR-VNET<nnn>-DC  Como criado pelo terraform no item 6.2 Variáveis do Ambiente |
| subnet | Nome da Subnet associada ao banco  Padrão: subnet-mgm-san5 |
| nsg\_net | Nome do NSG associado ao AKS  Padrão: aksnsg |
| router\_tb | Nome do router\_table  Padrão: rt-misan5<dev/qa/prodn> |

6.6 Ativar Cluster AKS

Tempo aproximado de execução: 2 minutos

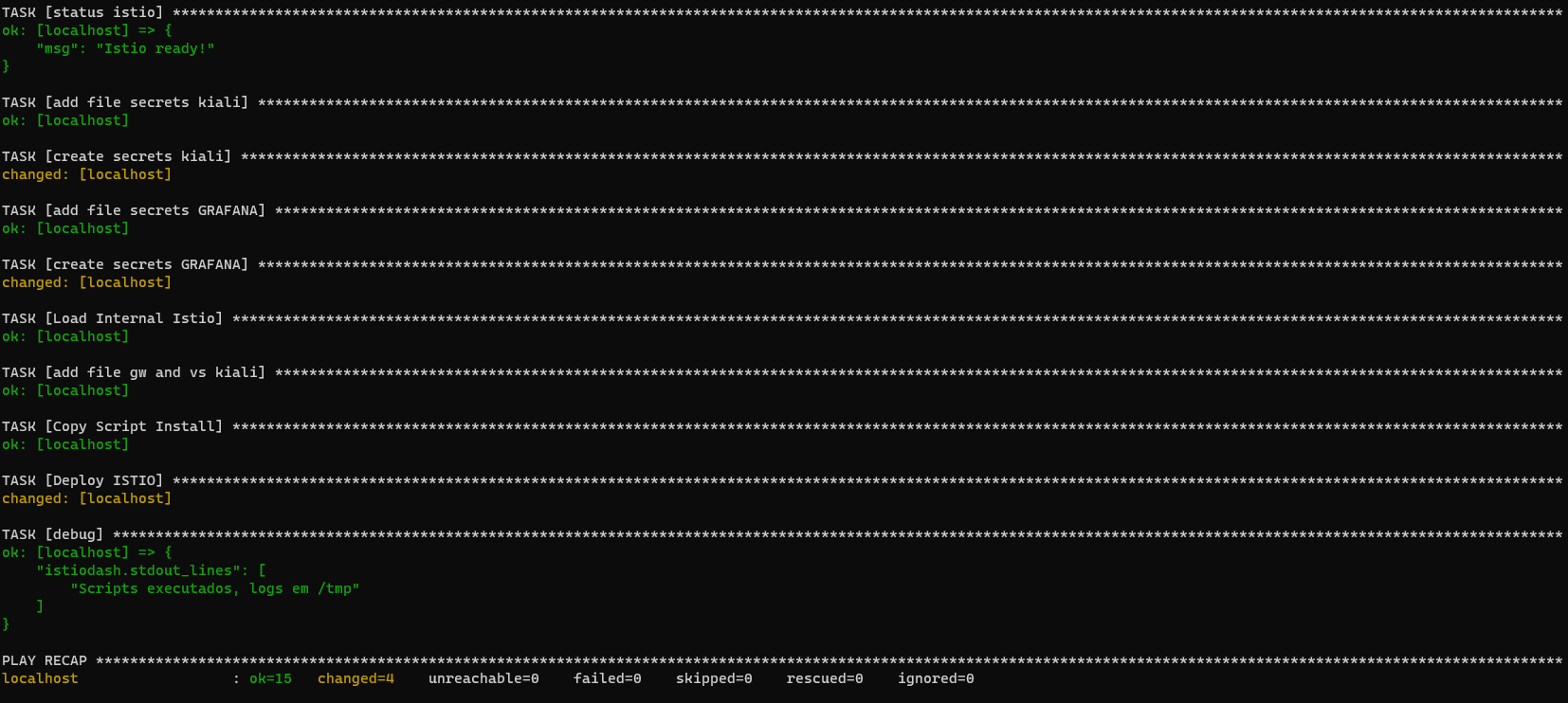
1. Efetue a conexão com o AKS, através do comando ***az aks get-credentials --resource-group xxx --name yyy***, onde
   1. Xxx – Nome do Resource Group do AKS
   2. Yyy – Nome do Cluster AKS
2. Edite as variáveis em **var\_ansible/vars.yml** de acordo com os valores em VARIAVEIS DO AMBIENTE ANSIBLE
3. Acesse a pasta 03-ansible-set-cluster execute o comando **ansible-playbook main.yml**.
4. Após a execução, será exibida a mensagem abaixo:



6.7 Instalação Istio

Tempo aproximado de execução: 12 minutos

1. Efetue a conexão com o AKS, através do comando ***az aks get-credentials --resource-group xxx --name yyy***, onde
   1. Xxx – Nome do Resource Group do AKS
   2. Yyy – Nome do Cluster AKS
2. Edite as variáveis em **var\_ansible/vars.yml** de acordo com os valores em VARIAVEIS DO AMBIENTE ANSIBLE
3. Acesse a pasta **04-ansible-istio** e execute o comando ***ansible-playbook main.yml***.
4. Após a execução, será exibida a mensagem abaixo:



6.8 Instalação Elastic

Tempo aproximado de execução: 1 minuto

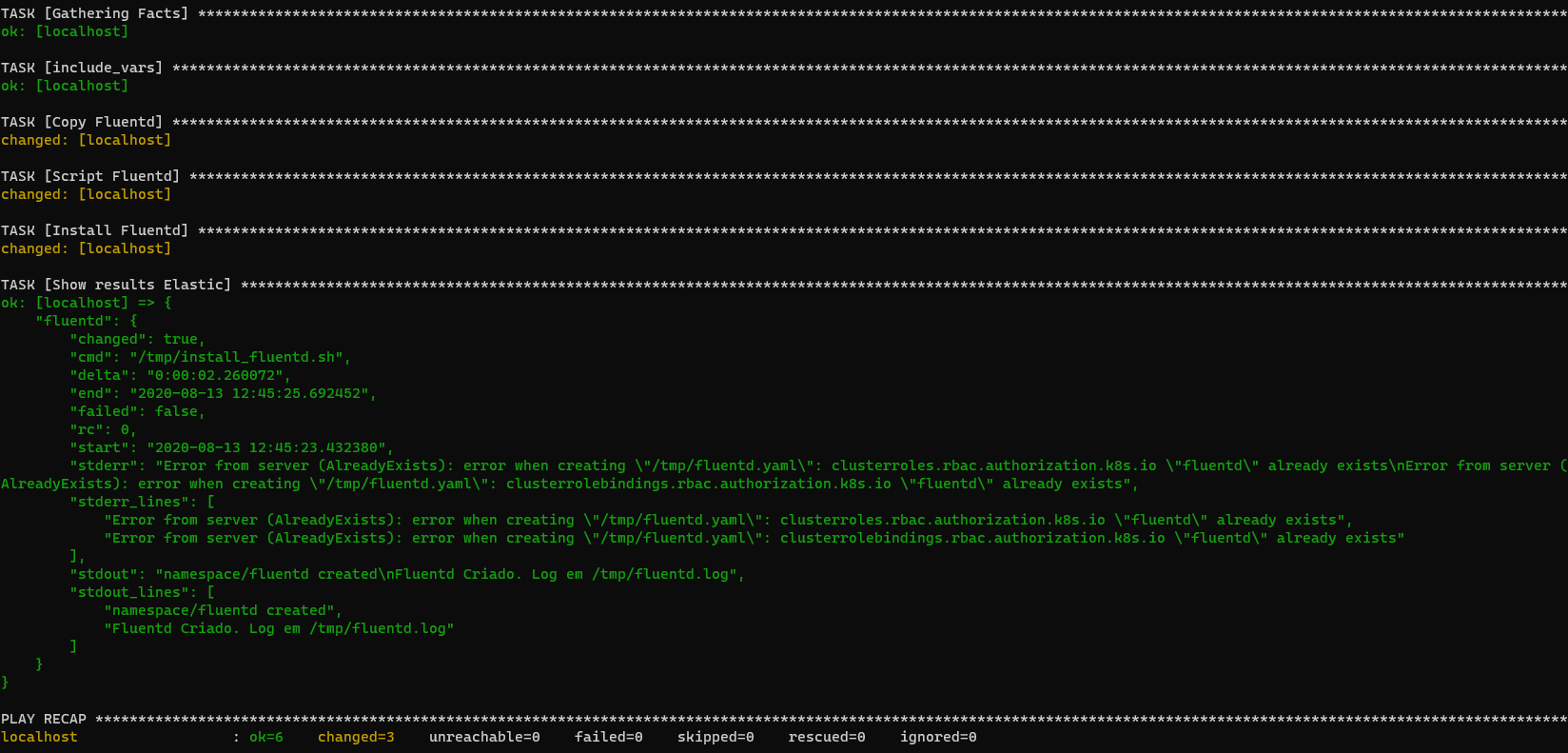
1. Efetue a conexão com o AKS, através do comando ***az aks get-credentials --resource-group xxx --name yyy***, onde
   1. Xxx – Nome do Resource Group do AKS
   2. Yyy – Nome do Cluster AKS
2. Edite as variáveis em **var\_ansible/vars.yml** de acordo com os valores em VARIAVEIS DO AMBIENTE ANSIBLE
3. Acesse a pasta **05-ansible-elastic** e execute o comando ***ansible-playbook main.yml***.
4. Após a execução, será exibida a mensagem abaixo:



6.9 Instalação Fluentd

Tempo aproximado de execução: 2 minutos

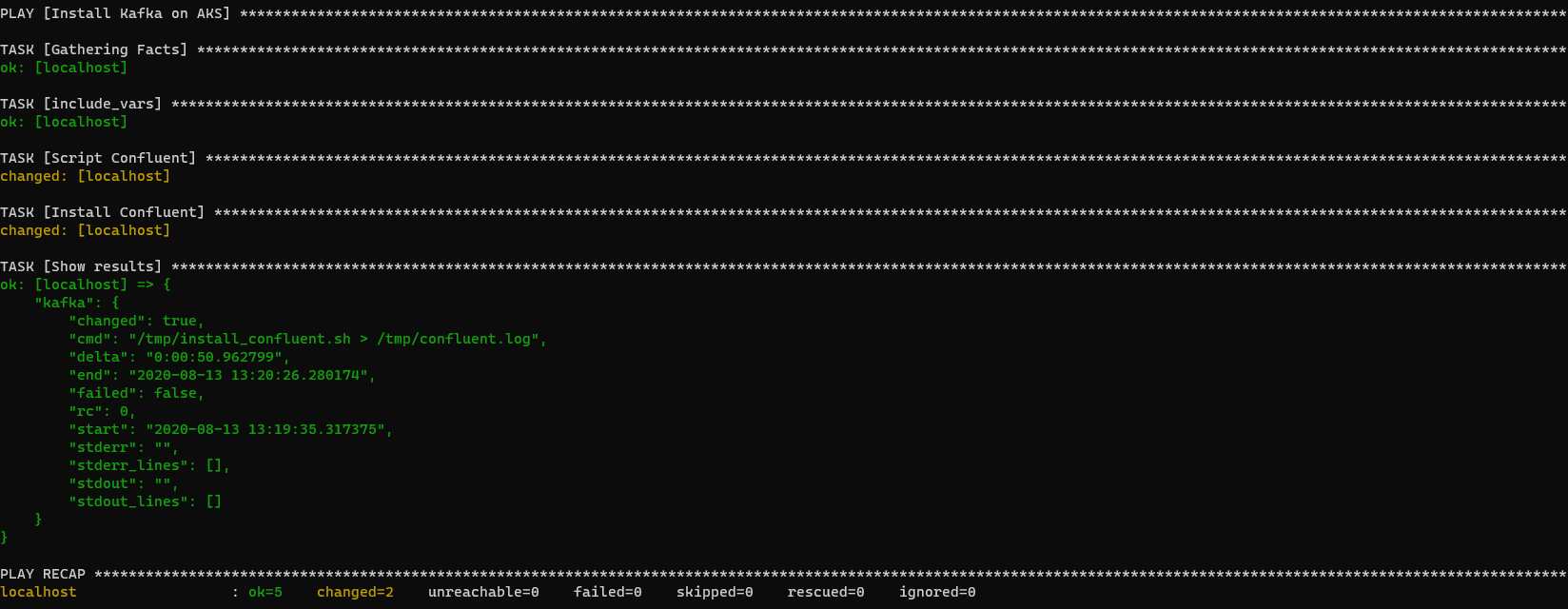
1. Efetue a conexão com o AKS, através do comando ***az aks get-credentials --resource-group xxx --name yyy***, onde
   1. Xxx – Nome do Resource Group do AKS
   2. Yyy – Nome do Cluster AKS
2. Edite as variáveis em **var\_ansible/vars.yml** de acordo com os valores em VARIAVEIS DO AMBIENTE ANSIBLE
3. Acesse a pasta **06-ansible-fluentd** e execute o comando ***ansible-playbook main.yml***.
4. Após a execução, será exibida a mensagem abaixo:



6.10 Instalação Kafka

Tempo aproximado de execução: 1 minuto

1. Efetue a conexão com o AKS, através do comando ***az aks get-credentials --resource-group xxx --name yyy***, onde
   1. Xxx – Nome do Resource Group do AKS
   2. Yyy – Nome do Cluster AKS
2. Edite as variáveis em **var\_ansible/vars.yml** de acordo com os valores em VARIAVEIS DO AMBIENTE ANSIBLE
3. Acesse a pasta **07-ansible-kafka** e execute o comando ***ansible-playbook main.yml***.
4. Após a execução, será exibida a mensagem abaixo:



6.11 Instalação SQL Manage Instance

Tempo aproximado de execução: 3 horas

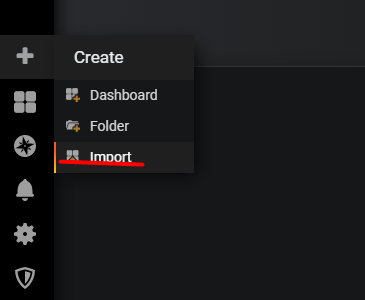
1. Efetue a conexão com o AKS, através do comando ***az aks get-credentials --resource-group xxx --name yyy***, onde
   1. Xxx – Nome do Resource Group do AKS
   2. Yyy – Nome do Cluster AKS
2. Edite as variáveis em **var\_ansible/vars.yml** de acordo com os valores em VARIAVEIS DO AMBIENTE ANSIBLE
3. Acesse a pasta **08-ansible-sqlmi** e execute o comando ***ansible-playbook main.yml***.
4. Após a execução, será exibida a mensagem abaixo:

6.12 Instalação Dashboard Confluentd no Grafana

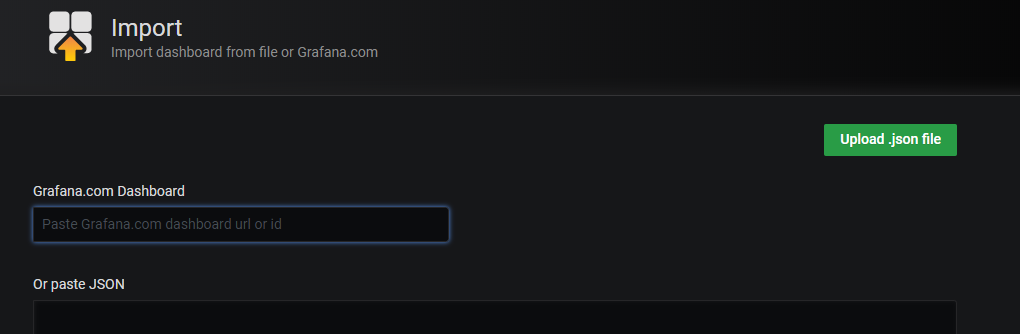
1. Digite o comando abaixo para abrir o dashboard do grafana: **istioctl dashboard grafana**. Será informado o link de acesso , conforme abaixo:



1. Acesse o endereço informado e digite o usuário e senha de acesso do Grafana
2. Clique em **+** e selecione a opção **Import**

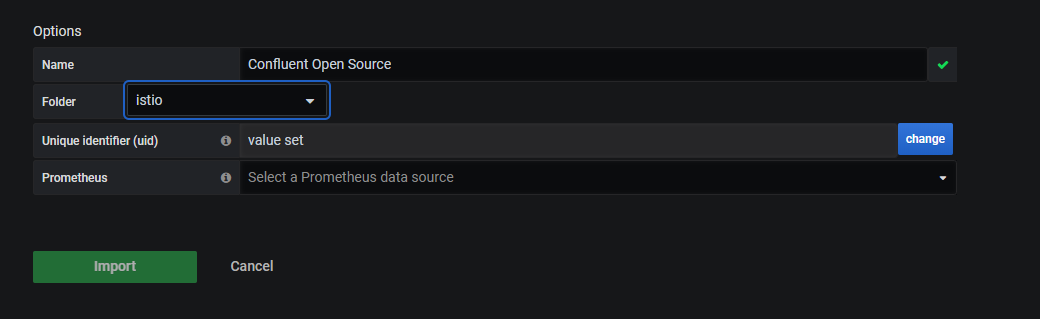


1. Em import, clique em ***Upload .json file*** e e selecione o arquivo em 07-ansible-kafka\templates\ confluent-open-source-grafana-dashboard.json



1. Clique em OK. Em seguida selecione as configurações abaixo:

* Em **Folder**, selecione **Istio**;
* Em **Uid**, clique em **Change.**
* Em **Prometheus**, selecione a opção **Prometheus**



1. Após selecionar todas as opções, clique em **Import.**

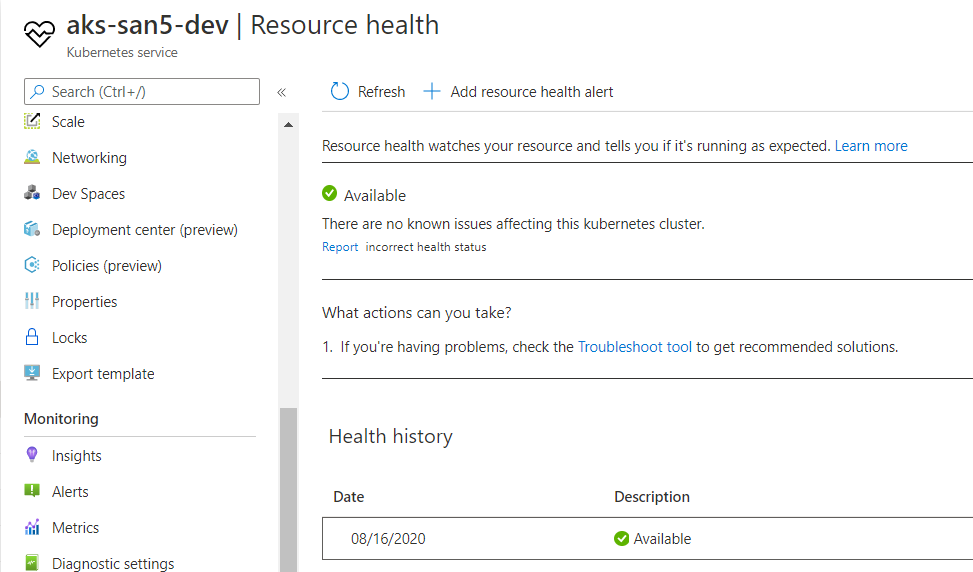
6.12 Instalação do api gateway

# Throubleshooting

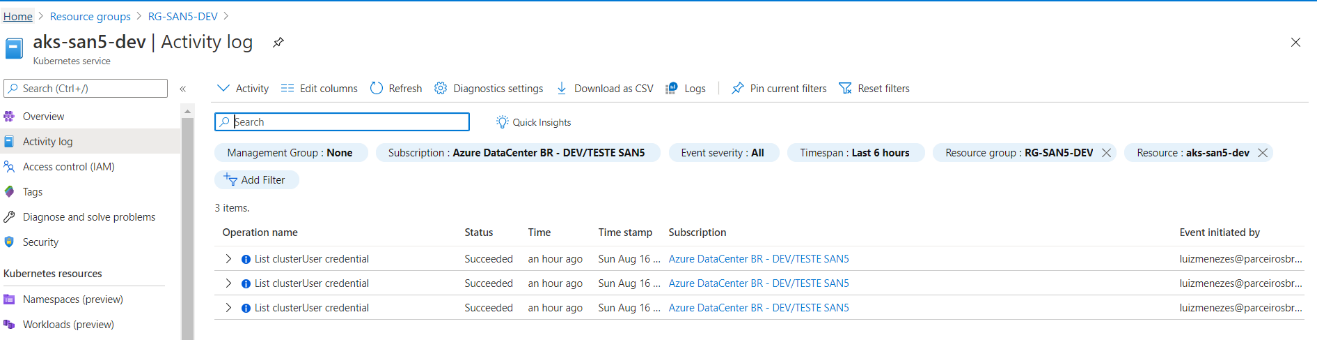
7.1 Azure

Para um troubleshoot inicial do Azure, siga os seguintes passos:

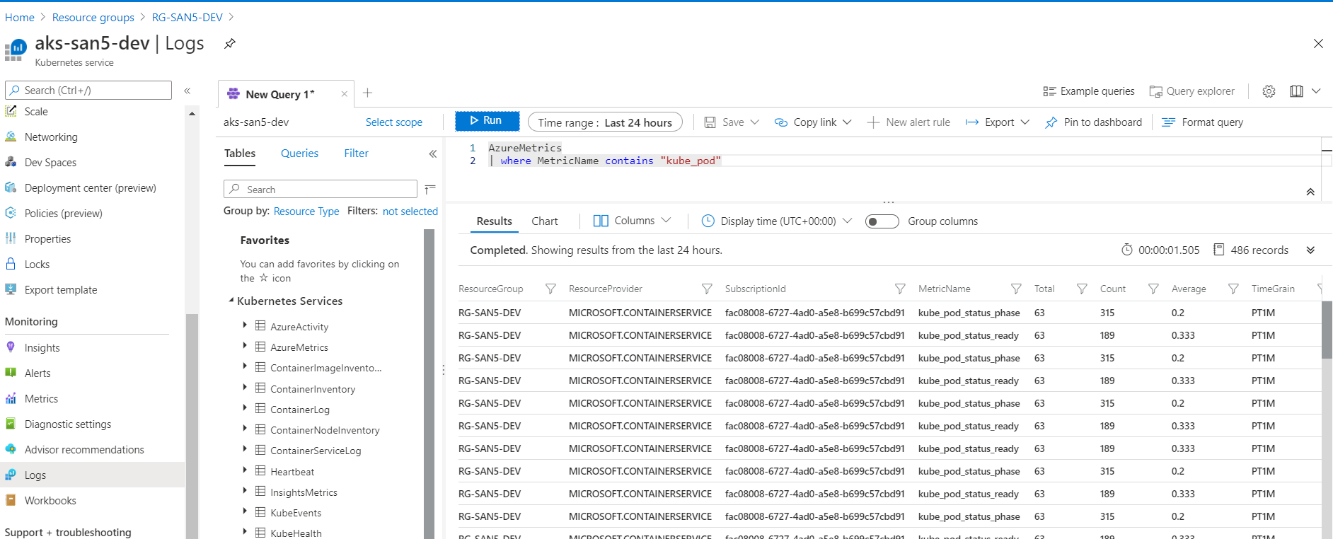
* Verifique o Resource Health do Componente:



* Verifique o Activity Log para validar as últimas alterações do componente em questão:



* Na Blade “Monitoring” existem várias informações importantes para troubleshooting;
* Utilize a opção “Log” para realizar consultas mais elaboradas e ter resultados personalizados:



7.2 Kubernetes

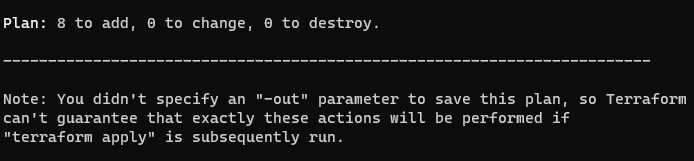
Comandos básicos para troubleshoot do Kubernetes

* Verifiquei a saúde dos nodes: ***kubectl get nodes***
* Verifique a saúde dos pods: ***kubectl get pods --all-namespaces***
* Verifique a saúde dos services: ***kubectl get svc --all-namespaces***
* Verifique os logs de um pod específico: ***kubectl logs –f <nome do pod> -n <Nome do Namespace>***

# Melhores Práticas do Terraform

Ao utilizar o Terraform, procure adotar as seguintes práticas no seu dia-a-dia para ter um código mais limpo e bem elaborado:

* Não utilize o “.” em nomes das variavéis;
* Sempre armazene a última versão do arquivo de estado “.tfstate” no repositório correto do devops para evitar descompaso entre ele a cloud;
* Utilize sempre o comando “terraform plan” para validar as alterações que você quer fazer. Após a saida do comando verificar quantos itens serão adicionados, modificados ou destruidos para saber se realmente o resultado será o desejado:



* Conferir minuciosamente a saida do comando acima para garantir que não haverá nenhuma alteração indevida;
* Procure identar o código de maneira a tornar menos complexa a leitura do arquivo;
* Uma vez implantado o script Terraform, **não faça alterações através do portal a fim de evitar descompasso entre o arquivo de estado e a nuvem**;
* Utilização do VSCODE + Git Desktop irá facilitar muito o desenvolvimento dos códigos e atualização dos mesmos no repositório;
* Todos os códigos foram adicionados a um repositório no Azure Devops. Utilize sempre este repositório atualizando os arquivos que estão lá. Link: <https://dev.azure.com/brk-vsts/Sistemas%20Comerciais%20-%20Infraestrutura%20TI/_git/Infraestrutura-TI>

# 

# Perguntas frequentes (FAQ)

1. **Como acessar o dashboard do Kubernetes no AKS?**

Kubernetes inclui um painel da web que pode ser usado para operações básicas de gerenciamento. Este painel permite exibir o status de integridade básico e as métricas dos seus aplicativos, criar e implantar serviços e editar aplicativos existentes. A descrição de com acessar este dashboard se encontra na documentação do Azure, neste [link](https://docs.microsoft.com/pt-br/azure/aks/kubernetes-dashboard).

1. **Como fazer para adicionar um nó no AKS no ambiente SAN5?**

Deve ser feito via **Terraform**, de forma a manter a consistência da configuração do ambiente, evitando o deprovisionamento de algum componente do Azure indevidamente.

Vide melhores práticas do uso do Terraform no item 8.

1. **Como fazer para alterar o tamanho dos nós do AKS do SAN5?**

Deve ser feito via **Terraform**, de forma a manter a consistência da configuração do ambiente, evitando o deprovisionamento de algum componente do Azure indevidamente.

Vide melhores práticas do uso do Terraform no item 8.

1. **Como fazer para aumentar o tamanho do banco de dados SQL no SAN5?**

Deve ser feito via **portal**, através da opção **Compute + Storage** em SQL Managed Instance.

1. **Como fazer para reinstalar / fazer upgrade do fluentd no SAN5?**

Deve ser feito via Ansible, conforme o passo 6.9 do documento. Lembrando que sempre deve editar as variáveis de acordo com os valores desejados.

1. **Como fazer para reinstalar / fazer upgrade do Istio no SAN5?**

Deve ser feito via Ansible, conforme o passo 6.7 do documento. Lembrando que sempre deve editar as variáveis de acordo com os valores desejados.

1. **Como fazer para reinstalar / fazer upgrade do Elastic no SAN5?**

Deve ser feito via Ansible, conforme o passo 6.8 do documento. Lembrando que sempre deve editar as variáveis de acordo com os valores desejados.

1. **Como fazer para reinstalar / fazer upgrade do Kafka no SAN5?**

Deve ser feito via Ansible, conforme o passo 6.10 do documento. Lembrando que sempre deve editar as variáveis de acordo com os valores desejados.

1. **Como fazer upgrade do Kubernetes? Posso fazer pelo portal?**

Deve ser feito via **Terraform**, conforme o passo 6.4 do documento. Lembrando que sempre deve editar as variáveis de acordo com os valores desejados.

# Anexos

Score para decisão de arquitetura



Configuração de referência da vnet de trânsito



Decisões de Arquitetura

