Aula 00 (Prof. Felipe Mathias e Raphael Lacerda)

BDMG (Analista de Desenvolvimento - Ênfase 2 - Infraestrutura e Segurança Cibernética) Desenvolvimento de software - 2024 (Pós-Edital)

Autor:
Felipe Mathias, Paolla Ramos

17 de Outubro de 2024

Índice

APRESENTAÇÃO DA AULA

Olá, alunos! Bem-vindos a mais uma aula do curso de Tecnologia de Informação para concursos públicos, no Estratégia Concursos.

Me chamo Felipe Mathias e serei seu professor na aula de hoje. Sou um catarinense de 30 anos, programador *front end* (ex-programador, se preferirem haha) e atuo como professor de cursos de Tecnologia da Informação voltados a concursos há mais de um ano. Assim como você, também vivo a vida de concurseiro, aguardando minha nomeação como Auditor Fiscal da Secretaria de Fazenda de Minas Gerais (SEF-MG), onde figuro no cadastro de reserva. Atualmente, continuo, em paralelo, estudando para concursos aguardando o meu grande sonho – o cargo de Auditor Fiscal da SEF-SC, com especialidade em TI.

Minha aventura no mundo do ensino surgiu de uma vontade interna de atuar como professor – sempre amei explicar as coisas, além de ter certa facilidade em expressar conceitos mais complexos para pessoas que talvez não tenham tanta experiência na área.

Meu objetivo aqui é digerir assuntos, desde os mais simples aos mais complexos, para que qualquer aluno consiga os entender, seja um programador, operador de infraestrutura, ou simplesmente um leigo que resolveu adentrar no mundo dos concursos e se deparou com TI no seu edital.

Gostaria de pedir que **sempre** vejam as questões comentadas durante a aula. Elas trazem conteúdo essencial para o aprendizado, muitas vezes abordando alguns pontos que não foram abordados no conteúdo e são essenciais para a resolução de questões.

Caso tenha alguma dúvida, não tenha receio de entrar em contato comigo nas minhas redes sociais (especialmente no meu Instagram, que deixarei abaixo), ou no fórum de dúvidas que os responderei assim que possível.

Ah, posto bastante coisa interessante de TI direcionada para concursos lá, dá uma olhadinha que algumas coisas podem te interessar. Volta e meio acerto alguma questão de prova por lá ;)

@fe.fiscal

t.me/fefiscal

ELASTICSEARCH

Conceitos Gerais

O ElasticSearch é uma ferramenta de pesquisa distribuída e um motor de análises. Ele funciona no "coração" da pilha ELK, ou ELK stack - um grupo de aplicações que trabalham em conjunto para fornecer um contexto de monitoramento e *logging*, composto pelo Elasticsearch, pelo Logstash e pelo Kibana.

O trabalho principal do Elasticsearch na pilha de aplicações é fazer a indexação dos dados, pesquisas e análises. Ele provê uma base de análise para todos os tipos de dados, estruturados, numéricos, não estruturados, geoespaciais, não importa - o Elasticsearch suporta. Até rimou 😂. Tudo isso graças à sua abordagem de indexação.

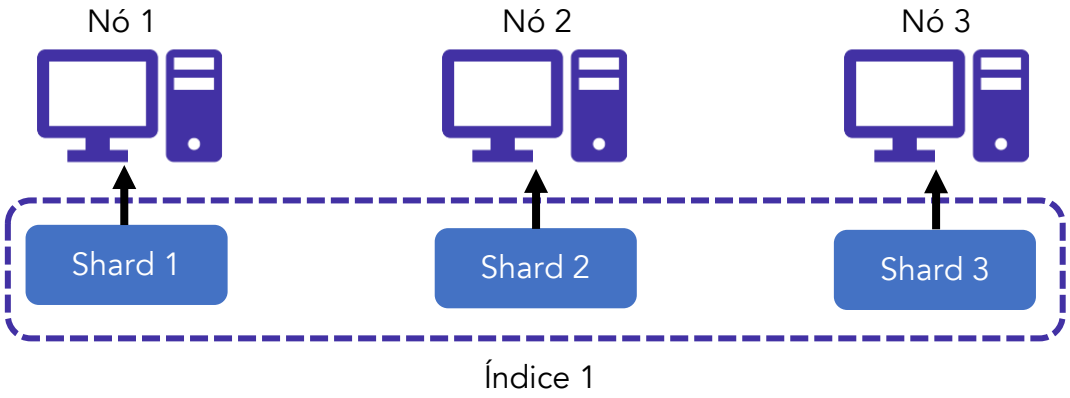
O Elasticsearch faz um armazenamento distribuído de documentos. Por esse motivo, muitas vezes ele é considerado um banco de dados NoSQL orientado a documentos, apesar de não ser um banco de dados propriamente dito. Para sua operacionalização, o Elasticsearch faz uso de arquivos no formato JSON.

Ao chegar no Elasticsearch, os dados são armazenados em documentos e alocados a um índice. Um índice é um agrupamento lógico de dados - pense nele como um conjunto de documentos. Seu objetivo é organizar e gerenciar os documentos dentro do ecossistema, organizando-os em pares de chave-valor. As chaves corresponderão aos campos do documento, enquanto os valores armazenam os dados relativo a esse campo.

ÍNDICES → AGRUPAMENTOS LÓGICOS DE DADOS (DOCUMENTOS)

Como esses índices usualmente agrupam numerosas quantidades de documentos, o Elasticsearch oferece uma forma de subdividir os índices. Esse componente é chamado de shard. Então, cada índice será composto por um ou mais *shards* - componentes esses que poderão ser alocados em diferentes nós do *cluster*, garantindo uma escalabilidade horizontal ao sistema.

A escalabilidade pode ser alcançada de duas formas - horizontal ou vertical. Na escalabilidade vertical, aumentamos o poder computacional de uma única unidade. Na escalabilidade horizontal, adicionamos mais nós ao conjunto de nós.



Para sua estruturação, o Elasticsearch usa uma estrutura de dados chamada de lista de índices invertidos. Esse índice invertido lista cada palavra única que aparece em um documento, e identifica todos os documentos em que essa palavra parece. A estrutura funciona mais ou menos da seguinte forma:

- Quando um documento é indexado, o Elasticsearch extrai toas as palavras únicas ou termos do documento
- Os termos são armazenados no índice invertido
- Para cada termo, o índice invertido mantém uma lista com todos os documentos que contêm esse termo
- Ao fazer uma pesquisa, a palavra é "procurada" no índice e, ao ser encontrada, os documentos nas quais ela aparece são retornados.

Essa indexação ocorre a nível de campo - ou seja, cada campo do arquivo JSON terá uma indexação. Esse índice comporta um par de chave-valor que aponta para determinado documento, ou campo de um documento. Para cada tipo de dado, um tipo diferente de índice e armazenamento do valor é usado, buscando trazer otimização para o programa. Por exemplo, campos de texto são armazenados em índices invertidos, arquivos numéricos são armazenados em árvores BKD, entre outros.

O Elasticsearch também pode funcionar sem esquema (*schema-less*), o que quer dizer que os dados podem ser indexados sem uma definição prévia de que abordagem ter, com base no tipo de dado analisado. A associação do tipo de dado com o tipo de índice é feita através de uma ferramenta chamada de mapeamento dinâmico. É possível desabilitar essa ferramenta e fazer uma associação direta, além de fazer indexações múltiplas a um mesmo valor.

(FUNDATEC/PGE RS/2021) Elasticsearch é um mecanismo distribuído de busca e análise de dados muito utilizado atualmente. Ele usa um índice invertido, que é projetado para permitir buscas textuais muito rápidas. Quanto ao seu modelo de dados, o Elasticsearch pode ser classificado como:

a) Orientado a colunas.
b) Orientado a documentos.
c) Orientado a grafos.
d) Banco de dados objeto-relacional.
e) Banco de dados XML.

Comentários:

Como vimos, o Elasticsearch usa documentos para armazenar os dados, no formato JSON - podendo ser considerado, portanto, orientado a documentos. (Gabarito: Letra B)

Apesar de ser possível o uso do Elasticsearch apenas como um repositório de documentos, isso seria jogar seu potencial fora - isso pois a aplicação que dá destaque para o programa são as suas pesquisas. Pesquisas do Elasticsearch são formas de recuperar os dados, quase como um *select*, no contexto do SQL.

Essas consultas podem ser feitas em diversas linguagens: Java, JavaScript, Go, .NET, PHP, Perl, Python e Ruby. Além disso, a API de pesquisa do Elasticsearch suporta consultas estruturadas, similar às consultas feitas no SQL, não estruturadas, do tipo full-text, e consultas complexas, que combinam duas formas diferentes de consultas. Todas providas através do Query DSL, linguagem usada para consultas compreensivas em arquivos JSON.

Além disso, é possível usar o Elastic como um motor de pesquisas similar ao que você acessa pelo Google, através de frases, palavras e prefixos - esses últimos com um recurso de *autocomplete*. Essa pesquisa personalizada e direcionada é garantida justamente pelas diferentes indexações dos dados

QUESTÃO DE PROVA

(CEBRASPE/TST/2024) ElasticSearch

a) é uma linguagem de programação.
b) suporta pesquisas de texto completo (full-text search).
c) é usado no armazenamento de banco de dados.
d) é um banco de dados relacional.
e) não suporta buscas em tempo real.

Comentários:

Vamos analisar as alternativas.

a) Errado. O Elasticsearch é uma ferramenta de "dupla funcionalidade". Ela armazena os dados em uma orientação a documentos, e também fornece um motor de buscas.
b) Certo. As pesquisas full-text são uma das abordagens permitidas pelo Elasticsearch.
c) Errado. Ele usado para armazenar dados, não banco de dados.
d) Errado. Se usado como banco de dados, ele se encaixa mais como um banco de dados orientado a documentos do que um banco de dados relacional.
e) Errado. A indexação do Elasticsearch demora menos de 1s para ocorrer - o que permite uma busca em tempo real.

Portanto, a alternativa correta é a letra B. (Gabarito: Letra B)

Nodes

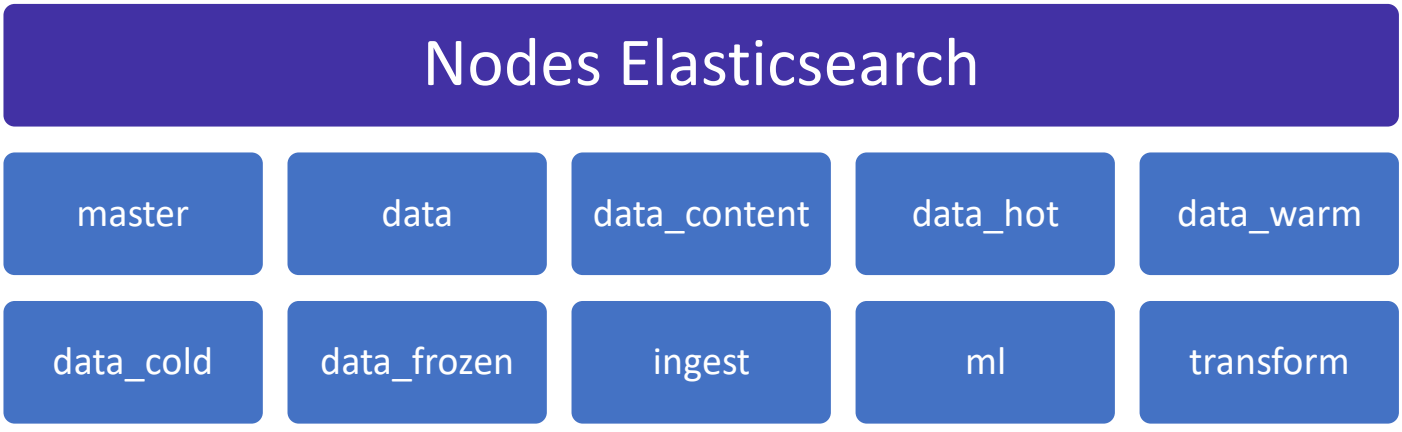
Um dos pontos centrais da estruturação do Elasticsearch é a alta disponibilidade dos dados. Isso é implementado graças à abordagem distribuída por natureza da ferramenta. Servidores (*nodes*) podem ser adicionados ao *cluster* para aumentar a escalabilidade horizontal do sistema, e o Elasticsearch automaticamente irá fazer uma distribuição dos arquivos ao longo dos nós disponíveis.

Isso é feito através da forma que a ferramenta implementa sua indexação. Por trás das cortinas, um índice nada mais é que uma forma de *sharding*, de fragmentação horizontal dos dados, onde cada *shard* é um índice independente. De forma geral, para o processamento, o Elasticsearch tem a seguinte ordem:

CONVERSÃO DOS DADOS EM DOCUMENTOS ⟹ ALOCAÇÃO DOS DOCUMENTOS EM DIVERSOS SHARDS ⟹ DISTRIBUIÇÃO DOS *SHARDS* AO LONGO DOS NÓS DO *CLUSTER*

Dentro do Elasticsearch, existem diferentes tipos de *nodes*. Cada tipo de *node* é responsável por realizar uma tarefa específica dentro do contexto da aplicação. Temos 11 tipos de node possíveis:

- master
- data
- data_content
- data_hot
- data_warm
- data_cold
- data_frozen
- ingest
- ml
- remote_cluster_client
- transform



Master node

O **`master node`** é responsável por ações leves ao longo do cluster, como criar ou deletar um índice, definir quais nós fazem parte do *cluster*, e decidir quais *shards* alocar a quais nós. Podemos ter uma pluralidade de nós elegíveis a mestre, e, em dado momento, qualquer um deles pode ser eleito como mestre - exceto os nós responsáveis apenas por votações.

Um ponto central para você entender os nós mestres é que não existe um nó desenvolvido exclusivamente como nó mestre - o apontamento do nó mestre é feito por um algoritmo de consenso, através de uma votação. Qualquer nó que participe da votação para eleição é considerado um nó elegível a mestre, mesmo que atue apenas votando, sem a capacidade de se tornar, de fato, um nó mestre.

Quando configuramos um nó, o arquivo de configuração (em YAML) do Elasticsearch recebe um parâmetro chamado de **`node.roles`**, que atribui um ou mais papéis a um nó. Apenas nós com o papel `master` apontado podem ser marcados como `voting_only` (apenas para votações). Esses nós participarão da votação, sem poder receber votos.

YAML
```yaml
node.roles: [ master, voting_only ]
```

Como o nó mestre é essencial na coordenação das atividades, uma boa prática recomendada é que ele não seja, em nenhum momento, sobrecarregado com tarefas. Para isso, os nós elegíveis a mestre devem ser configurados para assumirem essa posição de forma dedicada, dessa forma atuarão como uma espécie de nós coordenadores.

Um nó é considerado coordenador quando ele não armazena dados, nem participa no gerenciamento do *cluster* como um nó mestre. Ele age, basicamente, como um balanceador de

cargas inteligentes, roteando pedidos de clientes para os nós apropriados, atuando tanto na parte de armazenamento, quanto na parte de pesquisas.
ATENÇÃO! Para um *cluster* ser considerado de alta disponibilidade (HA - *High Availability*), ele deve conter pelo menos 3 nós elegíveis a mestre, sendo pelo menos dois desses sem o tipo *voting-only*. Dessa forma, se um dos nós tiver problema, ao menos uma outra opção para nó mestre estará disponível.

HIGH AVAILABILITY → PELO MENOS 3 NÓS ELEGÍVEIS, SENDO AO MENOS 2 NÃO EXCLUSIVOS PARA VOTAÇÃO

(Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Elasticsearch, julgue o item que segue.

Determinado cluster do Elasticsearch só pode ser considerado de alta disponibilidade se tiver ao menos 2 opções de nó mestre, para garantir que pelo menos um esteja operando em qualquer determinado momento.

Comentários:

Perfeito! Para ser considerado como de alta disponibilidade, um cluster deve ter ao menos 3 nós com o papel *master*, sendo que ao menos 2 não devem ser do tipo *voting-only*. (Gabarito: Correto)

Data node

O data node, ou nó de dados, são os nós que contém os documentos indexados. Os nós de dados fazem operações relacionados aos dados, como CRUD (Create, Read, Update e Delete), pesquisa e agregações. Essas operações exigem muito da memória, portanto é imperioso que haja um monitoramento frequente dos recursos desse nó.

A definição do Data node é feita da seguinte forma:

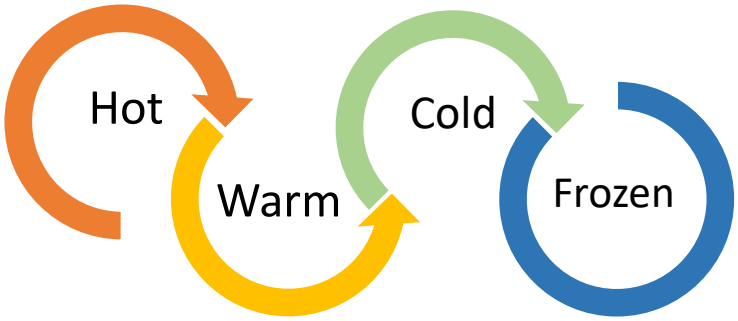
YAML
```yaml
node.roles: [ data ]
```

É possível adicionar temperaturas aos nós. Cada temperatura equivale a um *tier* do dado, baseado no seu tempo de vida, uso e importância do dado. Essas temperaturas ajudam a definir o ciclo de vida do dado de forma eficiente, otimizando a performance e o custo da infraestrutura. São 4 temperaturas diferentes: hot, warm, cold e frozen. Cada *tier* pode ser associado a um ou mais nós diferentes.

Vamos ver as definições:

- Hot data: o *tier* de dados "quentes" armazena dados acessados recentemente e/ou de forma frequente. Ele é otimizado para leitura rápida e operações de escrita, portanto deve ser provida de uma estrutura de hardware de alta performance. Exemplo: logs de métricas recentes.
- Warm data: o *tier* dos dados "mornos" lida com dados que ainda são necessários, ou seja, ainda são consultados, mas que não possuem uma frequência de interação tão elevada. Ele deve fazer um balanceamento entre performance e custo. Exemplo: logs de métricas mais antigas.
- Cold data: o *tier* dos dados "gelados" engloba os dados que são muito pouco acessados, mas que ainda necessitam ser armazenados para serem pesquisáveis. O foco passa a ser na redução do custo, apenas para manter uma capacidade de consulta ao dado. Exemplo: dados históricos, armazenados para *compliance*.
- Frozen data: o *tier* dos dados "congelados" armazena dados que quase nunca são acessados, mas que precisam ser armazenados para uma retenção de longo prazo. A prioridade é voltada 100% para o armazenamento de custo eficiente, já que os dados passam a ser imutáveis e sofrerão pouca ou nenhuma consulta. Exemplo: dados requeridos para fins legais ou regulatórios.

Com isso, temos a criação do "ciclo de vida" dos dados dentro do Elasticsearch.



Como cada *tier* exige uma estrutura diferente, é comum que diferentes nós lidem com diferentes categorias de dado. Para atribuir uma role, utilizamos o arquivo YAML:

```yaml
node.roles: [ data_hot ]
node.roles: [ data_warm ]
node.roles: [ data_cold ]
node.roles: [ data_frozen ]
```

Além disso, a definição do ciclo de vida de um dado é importante. Características que podem ser usadas para migrar o dado de um *tier* para o outro, automaticamente, envolvem, por exemplo, o

tempo de vida do dado e o tamanho, ou até mesmo a implementação de *scripts* para calcularem a média de leituras, mantendo o dado no *tier* adequado.
Especificamente para os dados no ***tier hot***, é uma prática comum definir uma ação de **`rollover`**. O *rollover* é uma ferramenta utilizada para gerenciar o crescimento de um índice ao longo do tempo. Ele permite que um índice transicione para um novo índice, quando determinado parâmetro for atingido - usualmente o tamanho do índice. Esse novo índice continuará a receber documentos, enquanto o antigo passa a assumir um estado apenas de leitura.

Um exemplo do código para definição do *rollover*, na fase *hot*, segue abaixo.

YAML

```
policy:
  phases:
    hot:
      min_age: 0ms
      actions:
        rollover:
          max_size: 50GB
          max_age: 7d
```

Veja que temos uma definição dupla - o *rollover* acontecerá em dois casos, quando o índice chegar a 50GB, ou quando seu tempo de vida chegar a 7 dias. O que acontecer primeiro irá disparar o *rollover*.

QUESTÃO DE PROVA

(FGV/TJDFT/2022) O analista de sistemas Lucas definiu uma nova política para o ciclo de vida de índices, denominada EspacoPolicy, no Elasticsearch. A EspacoPolicy, quando aplicada a um índice B, deve disparar rollover automático de B para um novo índice quando B atingir determinado nível de ocupação de espaço em disco.

Essa condição para o rollover de um índice baseado no nível de espaço em disco ocupado foi definida em EspacoPolicy, por Lucas, para a fase do ciclo de vida de índices:

a) hot;

b) warm;
c) cold;
d) frozen;
e) delete.

Comentários:

O *rollover* é uma técnica para "subir" um índice sempre que determinado limite for alcançado, usualmente um espaço em disco. Essa técnica somente pode ser utilizada na fase *hot* dos dados, pois é onde os dados são injetados e criados índices novos - nas outra fases não há mais criação de índices, apenas armazenamento de dados. (Gabarito: Letra A)

Content data node

Os **`content data nodes`** são parte do *tier* de conteúdo (*content*) - esse é um *tier* autônomo: enquanto os *tier* hot, warm, cold e frozen são dedicados aos fluxos de dados, normalmente *logs* do sistema, o tier *content* é único, e armazena um tipo especial de dado.

Os dados armazenados aqui usualmente compreendem uma coleção de itens, como catálogo de produtos ou arquivos de artigos. Enquanto os dados de *logs* constituem dados orientados a séries temporais, aqui nos nós de conteúdo temos dados mais estáticos, constantes ao longo do tempo.

Aqui as fases do ciclo de vida não se aplicam da mesma forma que nos *data nodes*, já que o objetivo é realmente manter o dado armazenado por muito tempo, com uma alta retenção, otimizando os nós que suportam esses dados para consulta.

O *tier* de conteúdo é obrigatório em um cluster Elasticsearch - isso pois os índices que não fazem parte da *stream* de dados (nós data, data_hot etc.) são automaticamente alocados nos nós **`content data`**.

Para atribuirmos a *role* de `data content`, usamos a seguinte sintaxe:

```yaml
node.roles: [ data_content ]
```

Ingest node

Os **`ingest node`**, ou nós de ingestão, são nós que executam *pipelines* de pré-processamento, compostas por um ou mais processadores de ingestão. De forma geral, não há unidades computacionais dedicadas para trabalharem como nós de ingestão, mas pode ser interessante,

dependendo do tipo e da quantidade de dados processados, ter um ponto dedicado a essa atividade.

Para criarmos um *role* de ingestão, usamos a seguinte diretiva:

```yaml
node.roles: [ ingest ]
```

(FGV/TJ RN/2023) A analista de suporte Ana gerencia o ELCluster na PGM de Niterói. O ELCluster agrupa nós do servidor de busca e análise de dados Elasticsearch. O Elasticsearch do ELCluster foi configurado para realizar uma série de transformações nos dados de entrada, a fim de extrair conteúdos específicos possivelmente presentes nos dados. Essas transformações são executadas em um pipeline antes da indexação dos dados. Ana constatou um desempenho abaixo do esperado nas transformações dos dados de entrada do ELCluster e concluiu serem necessários mais nós Elasticsearch dedicados à execução de pipelines de pré-processamento de dados.

Logo, Ana deve adicionar ao ELCluster mais nós Elasticsearch do tipo:

a) data;
b) ingest;
c) transform;
d) master-eligible;
e) machine learning.

Comentários:

Questão tranquila - Ana precisa de mais nós para pré-processamento de dados. Portanto, os nós que devem ser adicionados são os nós de ingestão, ou os *ingest node*. (Gabarito: Letra B)

Coordinating nodes

Como vimos anteriormente, os **`coordinating nodes`**, ou nós coordenadores, são nós de processamento, gerenciando e distribuindo tarefas ao longo do *cluster*. Seu objetivo aqui é trabalhar como um ponto intermediário, levando as tarefas para os nós necessários, coordenando a atividade.

Usualmente, nós atuam como coordenadores paralelamente à sua atividade principal. Porém, é possível criar nós exclusivos para a coordenação, e isso é feito não atribuindo nenhuma *role* a ele - por isso ele não aparece nas *roles* possíveis aos nós. O nó coordenador não é uma atribuição de role, e sim a falta de atribuição de role.

Veja como é feita a criação:

```yaml
node.roles: [ ]
```

Remote-eligeble nodes

Os nós **`remote-eligble`**, ou nós remotos elegíveis, são nós que atuam entre *clusters* diferentes, agindo como uma conexão entre diferentes *clusters*. Pode ser necessário propagar um pedido de tarefa ou até mesmo um dado de um *cluster* a outro, e os nós elegíveis como remotos atuam justamente fazendo essa ponte.

Com a conexão remota, também é possível sincronizar os dados entre os *cluster*, através de uma ferramenta de replicação *cross-cluster*, e fazer pesquisas através dos clusters, criando uma verdadeira rede de nós.

Um nó eleito como remoto recebe a *role* de **`remote_cluste_client`**. Veja:

```yaml
node.roles: [ remote_cluster_client ]
```

Machine Learning node

O nó de machine learning são nós destinados a rodar tarefas e gerenciar pedidos de API para ferramentas de machine learning. Para ser implementado, um nó de machine learning deve ser atribuído também como nó de cliente remoto, já que ele irá lidar com pedidos de API, lidando com interações externas ao *cluster*. Para criar um nó dedicado à tarefa de Machine Learning, temos a seguinte sintaxe:

```yaml
node.roles: [ ml, remote_cluster_client ]
```

Transform node

O transform node, ou nós de transformação, são direcionados a executar e receber pedidos de transformação através de uma API - e, por esse motivo, assim como os nós de machine learning, ele deve ser implementado como nó de cliente remoto.

Atividades de transformação, no contexto do Elasticsearch, são formas de rotacionar e sumarizar os dados, criando novos índices baseados em transformações dos dados originais. Essas

operações são úteis para criar agregações, sumários e outras dados derivados que auxiliem no entendimento dos dados e em análises mais complexas dos *logs*. Para criar um nó de transformação, usa-se a seguinte sintaxe:

YAML

```yaml
node.roles: [ transform, remote_cluster_client ]
```

Configurando o Cluster

O cluster do Elasticsearch representa o conjunto de nós (nodes), cada qual com um ou mais papéis específicos. A configuração do *cluster* se dá *nó a nó*, isso é, cada nó necessita de uma configuração específica, com determinados parâmetros a serem definidos. Essa configuração é feita num arquivo YAML, chamado de **`elasticsearch.yml`**. Veja um exemplo, para a configuração de um nó elegível a mestre:

YAML

```yaml
cluster.name: my-cluster
node.name: master-node-1
node.roles: [ master ]
network.host: 192.168.1.1
http.port: 9200
discovery.seed_hosts: ["192.168.1.2", "192.168.1.3"]
cluster.initial_master_nodes: ["master-node-1", "master-node-2"]
```

Nesse arquivo acima:

- cluster.name = refere-se ao cluster em que o nó está inserido.
- node.name = é o nome do nó sendo configurado. Cada nó deve ter um nome único.
- node.roles = especifica os papéis do nó.
- network.host = define o endereço lógico que o nó usará para se comunicar com outros nós.
- http.port = define a porta que serviço do nó será exposto.
- discovery.seed_hosts = define a lista inicial de nós agregados dentro do mesmo cluster, usado para descobrir os nós "vizinhos".
- cluster.initial_master_nodes = especifica os nós *master-eligible* iniciais na formação do cluster

Um dos parâmetros mais importantes talvez seja o **`seed_hosts`**, que faz o que é chamado de *bootstrap do cluster*. Isso é, ele irá verificar todos os outros nós do sistema, e montar o *cluster* Elasticsearch.

Feita a configuração, cada *node* deve ser iniciado manualmente, através do comando Bash **`./bin/elasticseach`**. Também é útil verificar a saúde do *cluster* através do uso do *endpoint* específico - o _cluster/health. A comunicação entre nós do Elasticsearch é feita através de métodos do protocolo HTTP, como GET, PUT e outros. Veja o comando:

BASH

```bash
curl -X GET "192.168.1.1:9200/_cluster/health?pretty"
```

Pesquisas

A pesquisa é o que diferencia o Elasticsearch de outras ferramentas de armazenamento distribuído de dados. Armazenar muitos fazem, mas a capacidade de pesquisar e retornar dados do Elastic é incomparável. Para operacionalizar as pesquisas, o Elasticsearch utiliza APIs RESTful como um motor de pesquisas, desenvolvido em cima do Apache Lucene - uma biblioteca de motores de busca, desenvolvida em Java.

O "ciclo" da pesquisa pode ser dividido em 4 etapas. A primeira é a requisição. Como estamos trabalhando com APIs REST, esse pedido é feito através do método HTTP GET. Veja um exemplo de requisição:

BASH

```
curl -X GET "http://localhost:9200/index_name/_search" -H 'Content-Type:
application/json' -d'
{
Texto da query
}
```

Na segunda etapa, temos o roteamento. Os nós coordenadores lidam com os pedidos e "quebram" a requisição em múltiplas operações de pesquisa, cada uma com um *shard* relevante como alvo. Após o encaminhamento aos *shards*, entra a fase de consulta (*query*). Cada *shard* executa a procura localmente e retorna um conjunto de resultados ao nó coordenador.

Essa consulta seleciona os IDs dos documentos e faz um *score* com base na relevância da consulta. Para calcular a relevância, o Elasticsearch utiliza um algoritmo de pontuação chamado de BM25 (Best Matching 25) como padrão. Ele é um algoritmo baseado na abordagem TF-IDF, calculando a relevância com base na frequência do termo e na frequência inversa do documento, entre outros fatores.

ALGORITMO DE PONTUAÇÃO → BM25

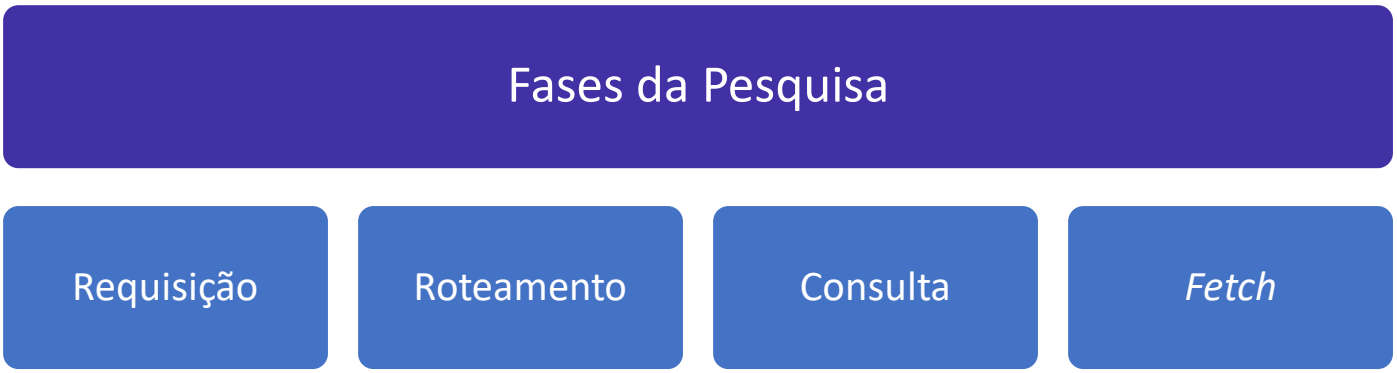
(CEBRASPE/TC DF/2023) Julgue o item a seguir, a respeito de elasticsearch e grafos.

O elasticsearch utiliza, por padrão, o algoritmo de pontuação BM25.

Comentários:

Exatamente! O padrão de algoritmo de pontuação, para classificação de retornos da pesquisa, é o BM25. (Gabarito: Certo)

Por fim, a consulta termina com a fase de ***fetch***, onde os resultados são coletados pelo nó coordenador, que compila o resultado e o retorna para o nó que fez o pedido. Com isso temos o ciclo completo da pesquisa.



Diferentes tipos de pesquisa podem ser realizados dentro do contexto do Elasticsearch. A saber:

- Match: procuram equivalência entre blocos de textos (*strings*). Pode ser direcionada a termos únicos, frases ou textos completos.
- Range: retorna documentos dentro de um determinado intervalo.
- Exists: retorna documentos que contenham determinado valor de campo.
- Prefix: encontra documentos que comecem com determinado prefixo.
- Bool: um tipo de *query* composta, que combina múltiplas consultas através de lógica booleana
- Constant Score: outra *query* composta, que "empacota" cada consulta e atribui uma pontuação constante a ela
- Join: são consultas destinadas a habilitar relações entre documentos. Dentre as consultas de *join*, temos a Nested, a Has_Parent, entre outros
- Geo: usada para pesquisas relacionadas a espaços geográficos. É possível delimitar a distância máxima de um dado para determinado par de coordenadas, por exemplo.
- Script: permite a execução de um script customizado para determinar se o documento é compatível ou não.

Entre diversos outros tipos utilizados. Veja que a gama de pesquisas e personalizações é enorme. Abaixo deixo um exemplo de um texto de uma pesquisa MATCH, escrito em JSON.

BASH

```
{
  "query": {
    "match": {
      "nome do campo": "termo de pesquisa"
    }
  }
}
```

Comandos CLI

Antes de adentrarmos os comandos de CLI (*Command Line Interface*), como o Elasticsearch trabalha com uma API RESTful, preciso recapitular com você os métodos HTTP - em especial 4 deles: GET, PUT, POST e DELETE. Esses métodos são formas como os clientes podem solicitar operação de servidores.

Cada método tem um intuito diferente:

- GET: Recupera dados de um servidor
- POST: envia dados para o servidor, para criar um novo recurso
- PUT: atualiza ou cria um recurso especificado no servidor
- DELETE: remove um recurso especificado no servidor

O Elasticsearch usará esses métodos para diferentes ações. Para acessar a API RESTful, utiliza-se comandos `curl`, seguido da diretiva `-x`, para definir que utilizaremos métodos HTTP. A sintaxe geral seguirá a seguinte estruturação:

```
curl -x MÉTODO <opções>
```

Dentre as opções, existem duas que se repetem frequentemente, portanto é importante que você saiba o que elas significam.

- -H = é o cabeçalho, que define o tipo do conteúdo da requisição para JSON
- -d = provê os dados para o *payload* do método HTTP

Além disso, podemos ter uma representação de um par de símbolos dentro das *strings* o `<>`. Ele indica campos variáveis. Por exemplo <Endereço> deverá ser substituído com o endereço do nó ou cluster em análise.

Um ponto de detalhe é a escolha do método de uso - principalmente na escolha entre PUT e POST, quando estamos indexando um novo documento. Basicamente, analisamos o seguinte ponto:

- PUT: usado queremos modificar um recurso ou criar um novo recurso com um ID específico. O endereço de criação especificará o ID, ficando, por exemplo, `/_doc/1`, onde o 1 corresponde ao ID.
- POST: usado apenas para criar recursos, sem se importar com o ID atribuído a ele. Nesse caso, o Elasticsearch atribuirá um ID automaticamente.

Vou passar alguns exemplos e irei explicar o que é a sintaxe de cada um.

1) Verificar o estado de saúde do cluster

```bash
curl -u elastic <Endereço>/_cat/health
```

Aqui, -u é uma abreviação para *username*, onde preenchemos com usuário:senha.

2) Listar todos os nós

```bash
curl -u elastic <Endereço>/_cat/nodes
```

3) Visualizar todos os índices

```bash
curl -u elastic <Endereço>/_cat/indices
```

4) Criar um índice

```bash
curl -X PUT "<Endereço>" -H 'Content-Type: application/json' -d'
{
  "settings" : {
      "index" : {
          "number_of_shards" : 3,
          "number_of_replicas" : 2
      }
  }
}'
```

A criação de um índice será sempre feita pelo método PUT. Além disso, acima temos um exemplo de um bloco de código relativa a um índice, em JSON, onde definimos os *shards* e a quantidade de replicas que esse *shard* terá.

5) Alocar um documento em um índice

Podemos utilizar os dois métodos, o PUT quando tivermos um ID definido do índice que quisermos alocar o documento, e o POST, quando não definimos o ID.

```bash
curl -X PUT "<Endereço>/_doc/ID" -H 'Content-Type: application/json'
-d'
{
  "field1" : "value1",
  "field2" : "value2"
}'
```

```bash
curl -X POST "<Endereço>/_doc" -H 'Content-Type: application/json'
-d'
{
  "field1" : "value1",
  "field2" : "value2"
}'
```

Veja que a diferença é pequena - somente mudamos o método e a especificação do ID (ou a ausência de especificação). Um outro ponto de detalhe é o *endpoint* da API utilizado. O POST sempre será usado com o _doc, mas o PUT pode ser usado em conjunto com outra *endpoint*, o _create. A diferença é simples:

- PUT e _doc = sempre irá criar um documento. Se outro já existir com o mesmo ID, será substituiído.
- PUT e _create = só irá criar um documento se o ID não existir previamente.

> Endpoint é um endereço específico de uma API (um URL ou URI) onde a aplicação pode ser acessada. É basicamente uma rota específica para acessar determinadas funcionalidades da aplicação. No Elasticsearch, o *endpoint* é especificado no endereço de acesso, logo após o método HTTP - por exemplo, `_doc` e `_update`.

(FGV/TJ TO/2022) No âmbito do ElasticSearch 8.x, a requisição que permite inserir um documento no índice (index) indice-01 com um _id igual a 1 é:

a) INSERT indice-01/1
b) NEW _doc/indice-01/1
c) POST indice-01/_create/1
d) PUT indice-01/_doc/1
e) WRITE _doc/indice-01/1

Comentários:

Como a questão quer inserir um documento num índice específico, precisamos usar o método PUT, direcionado ao *endpoint* _doc ou ao *endpoint* _create, que só criará o índice se não existir outro ID igual previamente. Das alternativas, a única que se encaixa com o requisitado é a letra D. Na letra C, temos um erro no uso de POST com a *endpoint* _create, além da especificação do ID. (Gabarito: Letra D)

6) Buscar por um documento específico

```bash
curl -X GET "<Endereço>/<Nome do índice>/_doc/<ID do documento>"
```

7) Deletar um documento específico

```bash
curl -X DELETE "<Endereço>/<Nome do índice>/_doc/<ID do documento>"
```

8) Atualizar um documento

A atualização também tem algumas nuances específicas. A começar pelo *endpoint* da API utilizado para esse fim - existem dois casos distintos:

- _doc → um *endpoint* usado para interagir diretamente com os documentos
- _update → outro *endpoint*, seu propósito específico é atualizar documentos existentes

Graças a essa interação, temos três interações possíveis com a atualização de documentos:

- PUT e _doc → caso o ID especificado já exista, irá substituir o arquivo na sua integridade
- POST e _doc → o Elasticsearch gerará um ID automaticamente, criando um novo arquivo
- POST e _update → permite a atualização parcial de documentos, especificando o campo o e valor a ser atualizado

Então, realisticamente, usamos duas combinações no dia a dia - PUT e _doc caso queiramos substituir integralmente um arquivo, ou POST e _update para substituir determinado campo. Veja um exemplo da aplicação com o uso do POST:

BASH

```
curl -X POST "<Endereço>/<Nome do índice>/_update/<doc_id>" -H 'Content-
Type: application/json' -d'
{
  "doc" : {
    "Campo a atualizar" : "Novo valor"
  }
}'
```

(Inédita/Prof. Felipe Mathias) Determinada entidade utiliza o Elasticsearch como forma de armazenamento e pesquisa dos *logs*. Determinado *log* foi armazenado com erro, e precisa ter parte de seu corpo atualizado. Assinale a alternativa que traz o método e o *endpoint* correto a serem determinados no comando:

a) GET e _update
b) PUT e _update
c) PUT e _doc
d) POST e _update
e) POST e _doc

Comentários:

Como a atualização é parcial, isso é, só precisamos modificar parte do documento já indexado, o método correto é o POST, aliado ao *endpoint* `_update`. Portanto, a alternativa correta é a letra D. (Gabarito: Letra D)

Análise de Texto

A análise de texto é o processo de converter textos não estruturados, como corpos de e-mail, mensagens e outros, para formatos estruturados, otimizando as pesquisas. A análise de texto é feita em dois momentos:

- Quando indexando os documentos.
- Ao realizar pesquisas em campos do tipo `text` (texto).

Com a análise de texto, o Elasticsearch faz uma espécie de Processamento de Linguagem Natural (PLN) e, para tanto, necessita de certas "adaptações" do texto para performar as consultas *full-text* de uma forma correta. A primeira delas é a tokenização, que quebra os textos em partes menores, usualmente em palavras individuais - essas partes são chamadas de *tokens*.

A tokenização permite a pesquisa por termos individuais, mas ainda temos um "problema" - a pesquisa ainda é feita de forma literal. A palavra "Estratégia" só irá ter como retorno outras palavras "Estratégia" idênticas, não retornando palavras com diferentes acentuações, diferentes padrões de maiúsculas e minúsculas, nada.

Para resolver esse problema, existe um processo denominado normalização. A normalização envolve trazer os *tokens* para um formato padrão, permitindo trazer operações de igualdade entre tokens que não são exatamente iguais, mas similares o suficiente para serem relevantes.

Feitas as configurações iniciais, entra o "motor" da análise de texto: o analyzer (analisador). O Elasticsearch vem com um analisador padrão, que serve para muitos casos - mas pode ser necessário modificá-lo e padronizá-lo, para atender padrões locais. Esses padrões podem atender padrões tanto de linguagem (português pt-br), quanto de estruturação de texto, com características particulares. Um analisador é composto por três "partes":

- Filtros de caracteres: os filtros de caracteres recebem o fluxo de caracteres originais e podem os transformar, adicionado, removendo ou modificando caracteres.
- Tokenizador: é o responsável por quebrar o texto em partes menores, denominadas *tokens*.
- Filtros de token: recebe o token e adiciona, remove ou os modifica com base em determinados filtros.

(FGV/TRT 16/2022) O ElasticSearch é o motor de buscas e análises por trás da Elastic Stack. Ele é capaz de proceder a análises de textos em dois momentos distintos (tempo de indexação e tempo de busca).

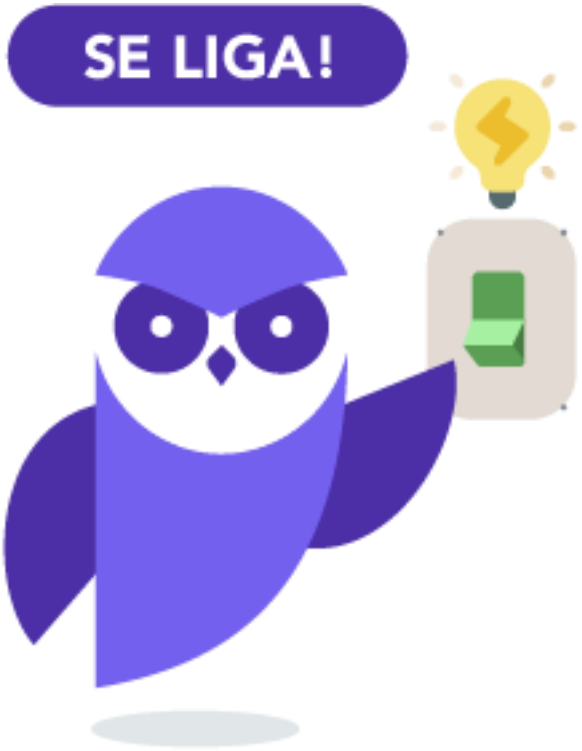
Os componentes analisadores de texto são denominados

a) Filtros Padrão, Filtros de Características e Filtros Personalizados.
b) Tokenizadores, Filtros de Log e Filtros Personalizados.
c) Filtros de Caracteres, Tokenizadores e Filtros de Tokens.
d) Filtros de Log, Filtros de Caracteres e Filtros de Tokens.
e) Filtros Personalizados, Filtros de Características e Filtros de Log.

Comentários:

Questão tranquila para quem acompanha o melhor material. Como acabamos de ver, os três componentes do analisador são os filtros de caracteres, o tokenizador e os filtros de tokens. Portanto, correta a letra C. (Gabarito: Letra C)

Como o Elasticsearch é multi-línguas, isso é, é utilizado em várias línguas (humanas) diferentes, é possível fazer uma adaptação local com os analisadores de linguagens. Essa configuração é feita através de um arquivo JSON, onde definimos alguns parâmetros. Veja o comando abaixo, e preste bastante atenção nos campos, já que sua literalidade já foi cobrada em provas:



Curl

```
PUT /brazilian_example
{
  "settings": {
    "analysis": {
      "filter": {
        "brazilian_stop": {
          "type":       "stop",
          "stopwords":  "_brazilian_"
        },
        "brazilian_keywords": {
          "type":       "keyword_marker",
          "keywords":   ["exemplo"]
        },
        "brazilian_stemmer": {
          "type":       "stemmer",
          "language":   "brazilian"
        }
      },
      "analyzer": {
        "rebuilt_brazilian": {
          "tokenizer":  "standard",
          "filter": [
            "lowercase",
            "brazilian_stop",
            "brazilian_keywords",
            "brazilian_stemmer"
          ]
        }
      }
    }
  }
}
```

(FCC/TRT 4/2022) No Elasticsearch, um conjunto de analisadores visam analisar textos de idiomas específicos. O analisador brasileiro pode ser reimplementado como um analisador personalizado, conforme o exemplo seguinte:

PUT /brazilian_example
{
"settings": {
"analysis": {
"filter": {
"brazilian_stop": {
"type": "stop",

```
      "stopwords": "_brazilian_"
    },
    "___I___": {
      "type":      "keyword_marker",
      "keywords":   ["exemplo"]
    },
    "brazilian_stemmer": {
      "type":      "stemmer",
      "language":   "brazilian"
    }
  },
  "analyzer": {
    "___II___": {
      "tokenizer": "standard",
      "___III___": [
        "lowercase",
        "brazilian_stop",
        "brazilian_keywords",
        "brazilian_stemmer"
      ]
    }}}}}
```

Preenchem, correta e respectivamente, as lacunas I, II e III:

a) brazilian_keywords – rebuilt_brazilian – filter
b) keywords_brazilian – brazilian_rebuilt – break
c) brazilian_words – brazilian_rebuilt – end_filtering
d) brazilian_marks – rebuilt_brazilian – filtering
e) words_brazilian – remake_brazilian – filter

Comentários:

Primeiramente, precisamos entender que informação está sendo inserida em cada campo. Em I, estamos definindo qual *keywords* estamos utilizando, isso passará ao Analyzer a linguagem de trabalho para encontrar as palavras chave. Como os outros parâmetros indicam que estamos trabalhando com o Brazilian Analyzer, o valor para o campo deve ser `brazilian_keywords`.

Em II definimos o *rebuilt*, que é uma personalização feita ao Analyzer, para atender a padrões locais. Como estamos "reconstruindo" o analisador brasileiro, o comando é rebuilt_brazilian. Por fim, em III, estamos definindo um tipo de filtro a ser aplicado na análise - sendo assim, o valor do campo é filter.

Ficamos com brazilian_keywords, rebuilt_analyzer e filter. (Gabarito: Letra A)

QUESTÕES COMENTADAS

ElasticSearch

01. (CEBRASPE/TST/2024) ElasticSearch

a) é uma linguagem de programação.
b) suporta pesquisas de texto completo (full-text search).
c) é usado no armazenamento de banco de dados.
d) é um banco de dados relacional.
e) não suporta buscas em tempo real.

02. (FGV/TJ RN/2023) A analista de suporte Ana gerencia o ELCluster na PGM de Niterói. O ELCluster agrupa nós do servidor de busca e análise de dados Elasticsearch. O Elasticsearch do ELCluster foi configurado para realizar uma série de transformações nos dados de entrada, a fim de extrair conteúdos específicos possivelmente presentes nos dados. Essas transformações são executadas em um pipeline antes da indexação dos dados. Ana constatou um desempenho abaixo do esperado nas transformações dos dados de entrada do ELCluster e concluiu serem necessários mais nós Elasticsearch dedicados à execução de pipelines de pré-processamento de dados.

Logo, Ana deve adicionar ao ELCluster mais nós Elasticsearch do tipo:

a) data;
b) ingest;
c) transform;
d) master-eligible;
e) machine learning.

03. (FCC/TRT 15/2023) Um Técnico de um Tribunal Regional do Trabalho está trabalhando com a pilha ELK (Elasticsearch, Logstash e Kibana) para uso em análise de dados. Pesquisando sobre a aplicação dessa pilha a fim de poder utilizá-la de modo otimizado, ele verificou que shard é

a) uma ferramenta de visualização e gerenciamento de dados usado para o Logstash.
b) um código usado para agregar e processar dados e enviá-los ao Elasticsearch.
c) uma capacidade fornecida pelo Elasticsearch para subdividir o índice em várias partes.
d) usado para agregar e processar dados e enviá-los ao Kibana. É um pipeline de processamento de dados do lado client que ingere dados de uma infinidade de fontes simultaneamente, os transforma e os envia para coleta.
e) um grupo (cluster) Elasticsearch de uma ou mais instâncias de nó que ficam no front end para enviar os resultados de uma pesquisa para o Kibana.

04. (CONSULPLAN/MPE MG/2023) ElasticSearch, Kibana e Logstash são softwares Open Source que compõem a pilha ELK. Com esse conjunto de ferramentas, é possível centralizar e armazenar bilhões de registros; efetuar buscas instantâneas; importar dados de diferentes formatos; coletar métricas de desempenho; gerar relatórios combinando vários filtros; e, criar dashboards dinâmicos, que transformam dados em informação com apenas alguns cliques. Em relação aos softwares Elasticsearch e Kibana, assinale a afirmativa correta.

a) ElasticSearch: permite junção entre diferentes índices.
b) Kibana: é uma interface web para analisar dados mantidos pelo ElasticSearch.
c) Kibana: é internacionalizável, ou seja, os filtros aplicados estão sempre visíveis e as operações de interjeição são aplicadas a todos os tipos de visualizações.
d) ElasticSearch: é um servidor de pesquisa que armazena os dados em forma de documentos e os disponibiliza no formato PDF. Uma de suas desvantagens é que ele não permite ainda disponibilizar tais documentos no formato JSON.

05. (CEBRASPE/SEPLAN RR/2023) A respeito de banco de dados, julgue o próximo item.

Elasticsearch é uma ferramenta baseada em SQL que, ao ser instalada no servidor de uma organização, recupera informações de diversos tipos de bancos de dados.

06. (CEBRASPE/TC DF/2023) Julgue o item a seguir, a respeito de elasticsearch e grafos.

O Elasticsearch utiliza, por padrão, o algoritmo de pontuação BM25.

07. (FGV/TRT 16/2022) O ElasticSearch é o motor de buscas e análises por trás da Elastic Stack. Ele é capaz de proceder a análises de textos em dois momentos distintos (tempo de indexação e tempo de busca).

Os componentes analisadores de texto são denominados

a) Filtros Padrão, Filtros de Características e Filtros Personalizados.
b) Tokenizadores, Filtros de Log e Filtros Personalizados.
c) Filtros de Caracteres, Tokenizadores e Filtros de Tokens.
d) Filtros de Log, Filtros de Caracteres e Filtros de Tokens.
e) Filtros Personalizados, Filtros de Características e Filtros de Log.

08. (FGV/TRT 13/2022) A plataforma Elasticsearch integra diversos produtos compondo a Elastic Stack, cada um desses produtos possui funcionalidades bem definidas.

O produto nativo da Stack que permite a visualização de dados é

a) dashviewer.

b) inforgram.
c) tableau.
d) kibana.
e) x-pack.

09. (CEBRASPE/BNB/2022) A respeito de banco de dados, julgue o item seguinte.

Elasticsearch é um processo de pesquisa que trabalha com grandes volumes de dados, processando requisições JSON bem como devolvendo dados JSON.

10. (CEBRASPE/TRT 8/2022) Mecanismos de pesquisa e de visualização de dados de aplicações web são fornecidos, respectivamente, pelas ferramentas

a) Elasticsearch e Zabbix.
b) Elasticsearch e Prometheus.
c) Kibana e Prometheus.
d) Prometheus e Zabbix.
e) Elasticsearch e Kibana.

11. (FCC/TRT 17/2022) O analista em tecnologia da informação avalia a implantação de ferramentas de busca e análise de dados distribuídos no TRT. Para implantar ferramenta ELK (Elasticsearch, Logstash e Kibana) o analista deve saber que o Elasticsearch

a) é a ferramenta utilizada para coletar, modificar e enviar dados para o Logstash.
b) é a ferramenta utilizada para visualizar os documentos gerados pelo Kibana.
c) é o elemento central da pilha ELK que armazena os seus dados centralmente para proporcionar busca rápida.
d) utiliza o mecanismo de busca direta por meio do id de cada evento gerado.
e) requer a porta UDP 161 para o agente e a 162 para o cliente abertas para receber dados do Logstash.

12. (FUMARC/TRT 3/2022) Analise as afirmativas a seguir sobre tecnologias utilizadas para monitoramento de ambientes tecnológicos.

I – Prometheus é um banco de dados de série temporal dimensional que extrai métricas de instâncias de aplicativos periodicamente com base na descoberta de serviço.
II – Kibana é uma ferramenta de visualização e exploração de dados usada para log e análise de séries temporais, monitoramento de aplicações e casos de uso de inteligência operacional.
III – Elasticsearch é um coletor de dados de código aberto que unifica coleções de dados para uma melhor compreensão dos dados e foi escrito na linguagem de programação Ruby.

Está CORRETO o que se afirma em:

a) I, II e III.
b) I, apenas.
c) I e II, apenas.
d) I e III, apenas.
e) II e III, apenas.

13. (FGV/TJ DFT/2022) O analista de sistemas Lucas definiu uma nova política para o ciclo de vida de índices, denominada EspacoPolicy, no Elasticsearch. A EspacoPolicy, quando aplicada a um índice B, deve disparar rollover automático de B para um novo índice quando B atingir determinado nível de ocupação de espaço em disco.

Essa condição para o rollover de um índice baseado no nível de espaço em disco ocupado foi definida em EspacoPolicy, por Lucas, para a fase do ciclo de vida de índices:

a) hot;
b) warm;
c) cold;
d) frozen;
e) delete.

14. (FGV/TJ TO/2022) No âmbito do ElasticSearch 8.x, a requisição que permite inserir um documento no índice (index) indice-01 com um _id igual a 1 é:

a) INSERT indice-01/1
b) NEW _doc/indice-01/1
c) POST indice-01/_create/1
d) PUT indice-01/_doc/1
e) WRITE _doc/indice-01/1

15. (FCC/TRT 4/2022) No Elasticsearch, um conjunto de analisadores visam analisar textos de idiomas específicos. O analisador brasileiro pode ser reimplementado como um analisador personalizado, conforme o exemplo seguinte:

```
PUT /brazilian_example
{
  "settings": {
    "analysis": {
      "filter": {
        "brazilian_stop": {
          "type":       "stop",
          "stopwords":  "_brazilian_"
```

```
      },
      "___I___":{
        "type":      "keyword_marker",
        "keywords":  ["exemplo"]
      },
      "brazilian_stemmer":{
        "type":      "stemmer",
        "language":  "brazilian"
      }
    },
    "analyzer":{
      "___II___":{
        "tokenizer":  "standard",
        "___III___":[
          "lowercase",
          "brazilian_stop",
          "brazilian_keywords",
          "brazilian_stemmer"
        ]
      }
    }
  }
}
```

Preenchem, correta e respectivamente, as lacunas I, II e III:

a) brazilian_keywords – rebuilt_brazilian – filter
b) keywords_brazilian – brazilian_rebuilt – break
c) brazilian_words – brazilian_rebuilt – end_filtering
d) brazilian_marks – rebuilt_brazilian – filtering
e) words_brazilian – remake_brazilian – filter

16. (FCC/TRT 4/2022) Para utilizar mecanismos de busca e análise de dados, um Técnico necessitou responder as seguintes questões:

I. Como é chamado o processo pelo qual os dados brutos são analisados, normalizados e enriquecidos antes de serem indexados no Elasticsearch?
II. O que o Kibana pode gerenciar?
III. O Elasticsearch armazena dados como documentos de que tipo?

Os itens I, II e III são respondidos, correta e respectivamente, com

a) ingestão de dados – Elastic Stack – JSON
b) Bingestão de dados – JSON – HTTP
c) Swagger – dados não formatados – HTML
d) indexação paralela – formatos não SQL – Elastic Stack
e) Swap-mode – Elastic Stack – formato não SQL

17. (FCC/TRT 4/2022) Sobre ferramentas de monitoramento, considere as seguintes características:

I. Permite monitoramento agentless (sem agentes) para diversos protocolos e conta com funções de auto-discovery (descoberta automática de itens) e low level discovery (descoberta de métricas em itens monitorados). Seus principais módulos do sistema de monitoramento são: server, proxy e agent.

II. Solução de código aberto para executar análises de dados, obtendo métricas que dão sentido à enorme quantidade de dados e para monitorar aplicativos com a ajuda de painéis personalizáveis. Se conecta a diversas fontes de dados, comumente chamadas de bancos de dados como Graphite, Prometheus, Influx DB, ElasticSearch, MySQL, PostgreSQL etc., que são algumas das muitas fontes de dados que suporta por padrão. Permite escrever plug-ins do zero para integração com várias fontes de dados diferentes.

III. Solução de busca e visualização de dados indexados no Elasticsearch e análise dos dados por meio da visualização em gráficos padrão ou apps integrados como o Lens, o Canvas e o Maps. Possibilita a análise visual de dados de um índice do Elasticsearch ou de vários índices.

As características I, II e III são, correta e respectivamente, correspondentes a

a) Grafana – Kibana – Zabbix.
b) Zabbix – Grafana – Kibana.
c) Kibana – Zabbix – Grafana.
d) Grafana – Zabbix – Kibana.
e) Zabbix – Kibana – Grafana.

18. (FCC/TRT 22/2022) Um Técnico, em uma máquina Linux com autorização e configurações ideais, fez a seguinte alteração no arquivo:

```
# ---------------- Network ----------------------------
#Set the bind address to a specific IP (IPv4 or IPv6):
#
network.host: localhost
```

Usando esse arquivo adequado, o Técnico especificou o localhost para que o Elasticsearch "escute" em todas as interfaces e IPs ligados. Para que esse ajuste seja validado, o Técnico deve salvar e fechar o arquivo

a) kibanaadmin.yml
b) htpasswd.users
c) elasticsearch.yml
d) elasticsearchadmin.yml
e) elasticsearch.sh

19. (CEBRASPE/SERPRO/2021) Com relação a conceitos de sistemas de arquivos distribuídos e sistemas de indexação, julgue o item a seguir.

Cada instância de elasticsearch é um nó; uma coleção de vários nós que podem trabalhar em harmonia formam um cluster elasticsearch.

20. (FUNDATEC/PGE RS/2021) Elasticsearch é um mecanismo distribuído de busca e análise de dados muito utilizado atualmente. Ele usa um índice invertido, que é projetado para permitir buscas textuais muito rápidas. Quanto ao seu modelo de dados, o Elasticsearch pode ser classificado como:

a) Orientado a colunas.
b) Orientado a documentos.
c) Orientado a grafos.
d) Banco de dados objeto-relacional.
e) Banco de dados XML.

21. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Elasticsearch, julgue o item que segue.

Determinado cluster do Elasticsearch só pode ser considerado de alta disponibilidade se tiver ao menos 2 opções de nó mestre, para garantir que pelo menos um esteja operando em qualquer determinado momento.

22. (Inédita/Prof. Felipe Mathias) Determinada entidade utiliza o Elasticsearch como forma de armazenamento e pesquisa dos *logs*. Determinado *log* foi armazenado com erro, e precisa ter parte de seu corpo atualizado. Assinale a alternativa que traz o método e o *endpoint* correto a serem determinados no comando:

a) GET e _update
b) PUT e _update

c) PUT e _doc
d) POST e _update
e) POST e _doc

GABARITO

GABARITO

1. Letra B
2. Letra B
3. Letra C
4. Letra B
5. Errado
6. Errado
7. Letra C
8. Letra D
9. Correto
10. Letra E
11. Letra C
12. Letra C
13. Letra A
14. Letra D
15. Letra A
16. Letra A
17. Letra B
18. Letra C
19. Correto
20. Letra B
21. Correto
22. Letra D

QUESTÕES COMENTADAS

ElasticSearch

01. (CEBRASPE/TST/2024) ElasticSearch

a) é uma linguagem de programação.
b) suporta pesquisas de texto completo (full-text search).
c) é usado no armazenamento de banco de dados.
d) é um banco de dados relacional.
e) não suporta buscas em tempo real.

Comentários:

O Elasticsearch é uma ferramenta com "dupla utilidade". Ele serve como um banco de dados orientado a documentos, e como uma ferramenta de pesquisa, através da indexação. Dada essa breve introdução, vamos analisar as alternativas.

a) Errado. O Elasticsearch não é uma linguagem de programação.
b) Certo. De fato, dentre seus métodos de pesquisa, estão as pesquisas em full-text.
c) Errado. Ele é usado no armazenamento de dados, não de bancos de dados.
d) Errado. O Elasticsearch usa dados semi-estruturados, não podendo ser classificado como relacional. Ele é um banco de dados orientado a documentos.
e) Errado. O Elasticsearch fornece acesso a pesquisas em tempo real, tendo em vista que seu processo de indexação de arquivos dura menos de 1s.

Portanto, a alternativa correta é a letra B.

Gabarito: Letra B

---

02. (FGV/TJ RN/2023) A analista de suporte Ana gerencia o ELCluster na PGM de Niterói. O ELCluster agrupa nós do servidor de busca e análise de dados Elasticsearch. O Elasticsearch do ELCluster foi configurado para realizar uma série de transformações nos dados de entrada, a fim de extrair conteúdos específicos possivelmente presentes nos dados. Essas transformações são executadas em um pipeline antes da indexação dos dados. Ana constatou um desempenho abaixo do esperado nas transformações dos dados de entrada do ELCluster e concluiu serem necessários mais nós Elasticsearch dedicados à execução de pipelines de pré-processamento de dados.

Logo, Ana deve adicionar ao ELCluster mais nós Elasticsearch do tipo:

a) data;

b) ingest;
c) transform;
d) master-eligible;
e) machine learning.

Comentários:

Dentro do Elasticsearch, existem diferentes tipos de *nodes*. Cada tipo de *node* é responsável por realizar uma tarefa específica dentro do contexto da aplicação. Temos 11 tipos de node possíveis:

- master
- data
- data_content
- data_hot
- data_warm
- data_cold
- data_frozen
- ingest
- ml
- remote_cluster_client
- transform

Para solucionar o problema da questão, Ana vai inserir "*mais nós Elasticsearch dedicados à execução de pipelines de pré-processamento de dados*". Os nós responsáveis por essas atividades, que recebem os dados e fazem transformações necessárias para que os dados atendam ao padrão requisitado, é o ingest node.

Gabarito: Letra B

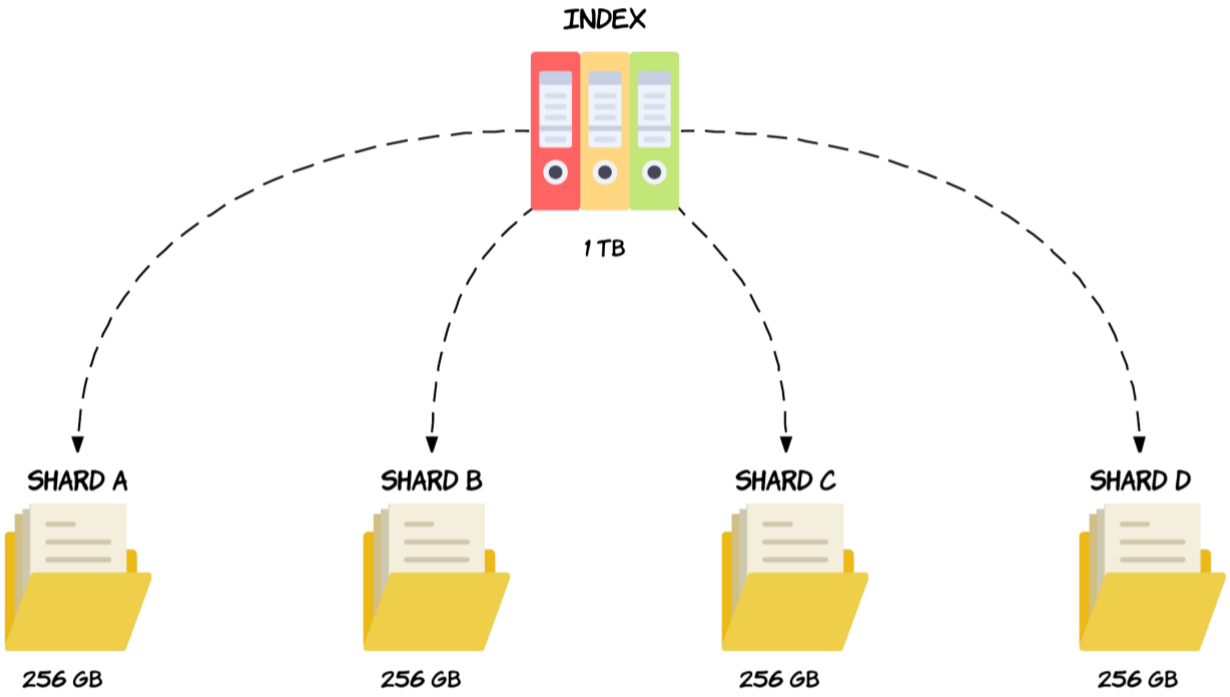
---

03. (FCC/TRT 15/2023) Um Técnico de um Tribunal Regional do Trabalho está trabalhando com a pilha ELK (Elasticsearch, Logstash e Kibana) para uso em análise de dados. Pesquisando sobre a aplicação dessa pilha a fim de poder utilizá-la de modo otimizado, ele verificou que shard é

a) uma ferramenta de visualização e gerenciamento de dados usado para o Logstash.
b) um código usado para agregar e processar dados e enviá-los ao Elasticsearch.
c) uma capacidade fornecida pelo Elasticsearch para subdividir o índice em várias partes.
d) usado para agregar e processar dados e enviá-los ao Kibana. É um pipeline de processamento de dados do lado client que ingere dados de uma infinidade de fontes simultaneamente, os transforma e os envia para coleta.
e) um grupo (cluster) Elasticsearch de uma ou mais instâncias de nó que ficam no front end para enviar os resultados de uma pesquisa para o Kibana.

Comentários:

O Elasticsearch realiza um processo de indexação de documentos, que representa um agrupamento lógico de dados, funcionando como uma coleção de documentos. Essa coleção pode ter subgrupos, que são chamados de shards. Esse shards são distribuídos entre os nós, com o objetivo de permitir uma escalabilidade horizontal. A alternativa que corretamente traz uma definição dos shards, embora simples, é a letra C.



Gabarito: Letra C

---

04. (CONSULPLAN/MPE MG/2023) ElasticSearch, Kibana e Logstash são softwares Open Source que compõem a pilha ELK. Com esse conjunto de ferramentas, é possível centralizar e armazenar bilhões de registros; efetuar buscas instantâneas; importar dados de diferentes formatos; coletar métricas de desempenho; gerar relatórios combinando vários filtros; e, criar dashboards dinâmicos, que transformam dados em informação com apenas alguns cliques. Em relação aos softwares Elasticsearch e Kibana, assinale a afirmativa correta.

a) ElasticSearch: permite junção entre diferentes índices.
b) Kibana: é uma interface web para analisar dados mantidos pelo ElasticSearch.
c) Kibana: é internacionalizável, ou seja, os filtros aplicados estão sempre visíveis e as operações de interjeição são aplicadas a todos os tipos de visualizações.
d) ElasticSearch: é um servidor de pesquisa que armazena os dados em forma de documentos e os disponibiliza no formato PDF. Uma de suas desvantagens é que ele não permite ainda disponibilizar tais documentos no formato JSON.

Comentários:

A questão aborda conhecimentos de dois dos componentes da pilha ELK - o Elasticsearch e o Kibana. Vamos analisar as alternativas.

a) Errado. Cada índice do Elasticsearch existe isoladamente, sem possibilidades de, nativamente, fazer junções.
b) Certo. O Kibana fornece uma interface de usuário na web para que usuários possam criar visualizações e analisar os dados mantidos no Elasticsearch.
c) Errado. O Kibana não é internacionalizável.
d) Errado. O Elasticsearch armazena os arquivos em JSON, não em PDF.

Portanto, a alternativa correta é a letra B.

Gabarito: Letra B

---

05. (CEBRASPE/SEPLAN RR/2023) A respeito de banco de dados, julgue o próximo item.

Elasticsearch é uma ferramenta baseada em SQL que, ao ser instalada no servidor de uma organização, recupera informações de diversos tipos de bancos de dados.

Comentários:

O Elasticsearch não é baseado em SQL. Suas consultas, ou pesquisas, são feitas através de requisições HTTP, utilizando uma API RESTful.

Gabarito: Errado

---

06. (CEBRASPE/TC DF/2023) Julgue o item a seguir, a respeito de elasticsearch e grafos.

O Elasticsearch utiliza, por padrão, o algoritmo de pontuação BM25.

Comentários:

Questão extremamente específica e difícil - mas que está correta. Ao retornar os documentos em uma pesquisa, o Elasticsearch elenca eles através de um *score*, uma pontuação. Para atribuir essa pontuação, é utilizado, por padrão, o BM25, um algoritmo baseado no conceito de TF-IDF, que usa a frequência do termo (tópico), baseado na frequência inversa dos documentos.

Gabarito: Errado

---

07. (FGV/TRT 16/2022) O ElasticSearch é o motor de buscas e análises por trás da Elastic Stack. Ele é capaz de proceder a análises de textos em dois momentos distintos (tempo de indexação e tempo de busca).

Os componentes analisadores de texto são denominados

a) Filtros Padrão, Filtros de Características e Filtros Personalizados.
b) Tokenizadores, Filtros de Log e Filtros Personalizados.
c) Filtros de Caracteres, Tokenizadores e Filtros de Tokens.
d) Filtros de Log, Filtros de Caracteres e Filtros de Tokens.
e) Filtros Personalizados, Filtros de Características e Filtros de Log.

Comentários:

Questão de nível bem avançado, que mistura um pouco de Elasticsearch com PLN. O Elasticsearch traz três componentes analisadores de textos diferentes:

- Filtros de caracteres: removem ou convertem caracteres especificais, como *stop words*, pontuações, espaços em branco, entre outros.
- Tokenizadores: responsáveis por dividir o texto em diferentes *tokens*, tornando as palavras em unidades capazes de serem analisadas pelo computador
- Filtros de Tokens: eles aplicam diferentes transformações nos tokens, fazendo, por exemplo, *stemming*, *lemming*, entre outros.

A alternativa que traz os três componentes é a letra C.

Gabarito: Letra C

---

08. (FGV/TRT 13/2022) A plataforma Elasticsearch integra diversos produtos compondo a Elastic Stack, cada um desses produtos possui funcionalidades bem definidas.

O produto nativo da Stack que permite a visualização de dados é

a) dashviewer.
b) inforgram.
c) tableau.
d) kibana.
e) x-pack.

Comentários:

A Elastic Stack é uma extensão da ELK Stack - onde temos Elasticsearch, Logstash, Kibana e Beats, um agente de dados. O programa responsável por fazer as visualizações de dados é o Kibana.

Gabarito: Letra D

---

09. (CEBRASPE/BNB/2022) A respeito de banco de dados, julgue o item seguinte.

Elasticsearch é um processo de pesquisa que trabalha com grandes volumes de dados, processando requisições JSON bem como devolvendo dados JSON.

Comentários:

Perfeito! O Elasticsearch faz pesquisas em seus documentos indexados, que são armazenados em JSON, através de requisições HTTP direcionadas a uma API RESTful. Essas requisições são feitas com corpo de dados em JSON, justamente para integrar com os documentos armazenados.

Gabarito: Correto

---

10. (CEBRASPE/TRT 8/2022) Mecanismos de pesquisa e de visualização de dados de aplicações web são fornecidos, respectivamente, pelas ferramentas

a) Elasticsearch e Zabbix.
b) Elasticsearch e Prometheus.
c) Kibana e Prometheus.
d) Prometheus e Zabbix.
e) Elasticsearch e Kibana.

Comentários:

A questão traz os seguintes programas:

- Elasticsearch: ferramenta de armazenamento de dados em forma de documentos, e de pesquisa (consulta) a esses dados
- Zabbix: ferramenta de monitoramento de rede
- Kibana: ferramenta para criar visualizações de dados
- Prometheus: banco de dados de séries temporais

Com isso em mente, vamos analisar as alternativas, procurando a que traz um mecanismo de pesquisa, e um de visualização de dados, respectivamente.

a) Elasticsearch e ~~Zabbix.~~

b) Elasticsearch e ~~Prometheus~~.
c) ~~Kibana~~ e ~~Prometheus~~.
d) ~~Prometheus~~ e ~~Zabbix~~.
e) Elasticsearch e Kibana.

Portanto, a alternativa correta é a letra E.

Gabarito: Letra E

---

11. (FCC/TRT 17/2022) O analista em tecnologia da informação avalia a implantação de ferramentas de busca e análise de dados distribuídos no TRT. Para implantar ferramenta ELK (Elasticsearch, Logstash e Kibana) o analista deve saber que o Elasticsearch

a) é a ferramenta utilizada para coletar, modificar e enviar dados para o Logstash.
b) é a ferramenta utilizada para visualizar os documentos gerados pelo Kibana.
c) é o elemento central da pilha ELK que armazena os seus dados centralmente para proporcionar busca rápida.
d) utiliza o mecanismo de busca direta por meio do id de cada evento gerado.
e) requer a porta UDP 161 para o agente e a 162 para o cliente abertas para receber dados do Logstash.

Comentários:

Vamos analisar cada alternativa, acerca dos conceitos básicos do Elasticsearch.

a) Errado. O Logstash que coleta e transforma dados para enviá-los ao Elasticsearch.
b) Errado. O Kibana que utiliza o Elasticseach como fonte para suas visualizações
c) Certo. O Elasticsearch é a peça central da pilha, armazenando os dados centralmente - aqui o centralmente se refere a uma única aplicação, não a um armazenamento centralizado, já que o Elasticsearch permite trabalharmos com computação distribuída.
d) Errado. A busca é feita pelos índices, não pelo ID.
e) Errado. O Elasticsearch é *agentless*. Além disso, utiliza a porta 9200 TCP para chamadas HTTP, e 9300 TCP para comunicação entre nós.

Sendo assim, correta a letra C.

Gabarito: Letra C

---

12. (FUMARC/TRT 3/2022) Analise as afirmativas a seguir sobre tecnologias utilizadas para monitoramento de ambientes tecnológicos.

I – Prometheus é um banco de dados de série temporal dimensional que extrai métricas de instâncias de aplicativos periodicamente com base na descoberta de serviço.

II – Kibana é uma ferramenta de visualização e exploração de dados usada para log e análise de séries temporais, monitoramento de aplicações e casos de uso de inteligência operacional.
III – Elasticsearch é um coletor de dados de código aberto que unifica coleções de dados para uma melhor compreensão dos dados e foi escrito na linguagem de programação Ruby.

Está CORRETO o que se afirma em:

a) I, II e III.
b) I, apenas.
c) I e II, apenas.
d) I e III, apenas.
e) II e III, apenas.

Comentários:

Vamos analisar cada item.

I. Correto. O Prometheus é um TSDB (Time Series DataBase), ou seja, ele armazena dados em séries temporais para extrair as métricas dos dados.
II. Correto. O Kibana é uma ferramenta que analisa dados e cria visualizações a partir desses dados, facilitando o entendimento e a percepção de ocorrências nos logs.
III. Errado. O Elasticsearch é uma ferramenta de armazenamento e pesquisa de documentos (dados), que foi escrita em Java, e não em Ruby.

Ficamos com I e II corretos - correta a letra C.

Gabarito: Letra C

---

13. (FGV/TJ DFT/2022) O analista de sistemas Lucas definiu uma nova política para o ciclo de vida de índices, denominada EspacoPolicy, no Elasticsearch. A EspacoPolicy, quando aplicada a um índice B, deve disparar rollover automático de B para um novo índice quando B atingir determinado nível de ocupação de espaço em disco.

Essa condição para o rollover de um índice baseado no nível de espaço em disco ocupado foi definida em EspacoPolicy, por Lucas, para a fase do ciclo de vida de índices:

a) hot;
b) warm;
c) cold;
d) frozen;
e) delete.

Comentários:

Um índice, dentro do Elasticsearch, pode passar por 4 fases ao longo do seu "ciclo de vida":

- Hot: o índice é ativo (recebe dados), e recebe pesquisas frequentemente.
- Warm: o índice não é mais ativo, mas ainda recebe pesquisas com boa frequência.
- Cold: o índice não é mais ativo, e recebe poucas consultas.
- Frozen: o índice não é mais ativo, e as consultas são quase inexistentes.

Neste contexto, o Elasticsearch fornece uma ferramenta chamada de rollup. Ela serve para que um novo índice possa ser criado, quando o índice em "alimentação" atinge seu limite superior, usualmente um tamanho específico ou um tempo. Assim, esse índice passa para o próximo *tier* (fase) e um novo índice é criado, agregando os dados.

Como o *rollup* é uma ferramenta que só pode trabalhar com um índice que está recebendo documentos, ela só pode ser aplicada no *tier hot*. Que nos leva à resposta da questão - a letra A.

Gabarito: Letra A

---

14. (FGV/TJ TO/2022) No âmbito do ElasticSearch 8.x, a requisição que permite inserir um documento no índice (index) indice-01 com um _id igual a 1 é:

a) INSERT indice-01/1
b) NEW _doc/indice-01/1
c) POST indice-01/_create/1
d) PUT indice-01/_doc/1
e) WRITE _doc/indice-01/1

Comentários:

Quando queremos alocar um documento em um índice, podemos fazê-lo de 3 formas distintas:

- Usando PUT para o *endpoint* _doc, com o ID sendo obrigatório para a definição do comando. Aqui sempre criaremos um documento e, caso exista um documento com o mesmo ID, ele será substituído.
- Usando PUT com o *endpoint* _create. Aqui também precisamos especificar o ID, mas o documento só será criado se não houver outro ID igual no diretório.
- Usando POST com o *endpoint* _doc, sem a necessidade de especificar o ID. O Elasticsearch automaticamente atribuirá um ID novo a esse documento.

Como queremos usar o id (1), precisamos usar o PUT. A alternativa correta nos traz o uso do PUT associado ao *endpoitn* _doc - ficaremos com o comando PUT indice-01/_doc/1 portanto.

Gabarito: Letra D

15. (FCC/TRT 4/2022) No Elasticsearch, um conjunto de analisadores visam analisar textos de idiomas específicos. O analisador brasileiro pode ser reimplementado como um analisador personalizado, conforme o exemplo seguinte:

```
PUT /brazilian_example
{
  "settings": {
    "analysis": {
      "filter": {
        "brazilian_stop": {
          "type":       "stop",
          "stopwords":  "_brazilian_"
        },
        "___I___": {
          "type":       "keyword_marker",
          "keywords":   ["exemplo"]
        },
        "brazilian_stemmer": {
          "type":       "stemmer",
          "language":   "brazilian"
        }
      },
      "analyzer": {
        "___II___": {
          "tokenizer":  "standard",
          "___III___": [
            "lowercase",
            "brazilian_stop",
            "brazilian_keywords",
            "brazilian_stemmer"
          ]
        }
      }
    }
  }
}
```

Preenchem, correta e respectivamente, as lacunas I, II e III:

a) brazilian_keywords – rebuilt_brazilian – filter
b) keywords_brazilian – brazilian_rebuilt – break
c) brazilian_words – brazilian_rebuilt – end_filtering
d) brazilian_marks – rebuilt_brazilian – filtering
e) words_brazilian – remake_brazilian – filter

Comentários:

Primeiramente, precisamos entender que informação está sendo inserida em cada campo - nessa questão específica, de altíssimo nível, a banca explora um recurso chamado de Analisador (especificamente, o Brazilian Analyzer).

Em I, estamos definindo qual *keywords* estamos utilizando, isso passará ao Analyzer a linguagem de trabalho para encontrar as palavras chave. Como os outros parâmetros indicam que estamos trabalhando com o Brazilian Analyzer, o valor para o campo deve ser `brazilian_keywords`.

Em II definimos o *rebuilt*, que é uma personalização feita ao Analyzer, para atender a padrões locais. Como estamos "reconstruindo" o analisador brasileiro, o comando é rebuilt_brazilian. Por fim, em III, estamos definindo um tipo de filtro a ser aplicado na análise - sendo assim, o valor do campo é filter.

Ficamos com brazilian_keywords, rebuilt_analyzer e filter - gabarito letra A.

Gabarito: Letra A

---

16. (FCC/TRT 4/2022) Para utilizar mecanismos de busca e análise de dados, um Técnico necessitou responder as seguintes questões:

I. Como é chamado o processo pelo qual os dados brutos são analisados, normalizados e enriquecidos antes de serem indexados no Elasticsearch?
II. O que o Kibana pode gerenciar?
III. O Elasticsearch armazena dados como documentos de que tipo?

Os itens I, II e III são respondidos, correta e respectivamente, com

a) ingestão de dados – Elastic Stack – JSON
b) Bingestão de dados – JSON – HTTP
c) Swagger – dados não formatados – HTML
d) indexação paralela – formatos não SQL – Elastic Stack
e) Swap-mode – Elastic Stack – formato não SQL

Comentários:

Vamos analisar cada item, e entender que conceito ele aborda.

I. Esse processo é denominado ingestão de dados.
II. O Kibana gerencia o Elasticsearch - mais precisamente, os dados nele armazenados.
III. Os dados do Elasticsearch são armazenados em JSON.

Ficamos com *ingestão de dados - Elasticearch - JSON*, portanto, gabarito letra A.

Gabarito: Letra A

---

17. (FCC/TRT 4/2022) Sobre ferramentas de monitoramento, considere as seguintes características:

> I. Permite monitoramento agentless (sem agentes) para diversos protocolos e conta com funções de auto-discovery (descoberta automática de itens) e low level discovery (descoberta de métricas em itens monitorados). Seus principais módulos do sistema de monitoramento são: server, proxy e agent.
>
> II. Solução de código aberto para executar análises de dados, obtendo métricas que dão sentido à enorme quantidade de dados e para monitorar aplicativos com a ajuda de painéis personalizáveis. Se conecta a diversas fontes de dados, comumente chamadas de bancos de dados como Graphite, Prometheus, Influx DB, ElasticSearch, MySQL, PostgreSQL etc., que são algumas das muitas fontes de dados que suporta por padrão. Permite escrever plug-ins do zero para integração com várias fontes de dados diferentes.
>
> III. Solução de busca e visualização de dados indexados no Elasticsearch e análise dos dados por meio da visualização em gráficos padrão ou apps integrados como o Lens, o Canvas e o Maps. Possibilita a análise visual de dados de um índice do Elasticsearch ou de vários índices.

As características I, II e III são, correta e respectivamente, correspondentes a

a) Grafana – Kibana – Zabbix.
b) Zabbix – Grafana – Kibana.
c) Kibana – Zabbix – Grafana.
d) Grafana – Zabbix – Kibana.
e) Zabbix – Kibana – Grafana.

Comentários:

Vamos analisar cada item.

I. O item explora o conceito da ferramenta de monitoramento Zabbix.
II. A solução que faz análises, obtendo métricas e criando visualizações é o Grafana. É possível identificar a ferramenta e diferenciá-la do Kibana devido à sua multiplicidade de integrações.
III. Aqui temos a definição do Kibana, ferramenta de criação de visualização no contexto da *stack* Elastic.

Portanto, nossa resposta é Zabbix - Grafana - Kibana.

Gabarito: Letra B

---

18. (FCC/TRT 22/2022) Um Técnico, em uma máquina Linux com autorização e configurações ideais, fez a seguinte alteração no arquivo:

---------- Network ----------
#Set the bind address to a specific IP (IPv4 or IPv6):
#
network.host: localhost

Usando esse arquivo adequado, o Técnico especificou o localhost para que o Elasticsearch "escute" em todas as interfaces e IPs ligados. Para que esse ajuste seja validado, o Técnico deve salvar e fechar o arquivo

a) kibanaadmin.yml
b) htpasswd.users
c) elasticsearch.yml
d) elasticsearchadmin.yml
e) elasticsearch.sh

Comentários:

Questão "tranquila" e direita. Essas configurações de *localhost*, portas e IPs são feitas no arquivo de configuração do Elasticsearch - um arquivo YAML que se chama `elasticsearch.yml`. Correta, portanto, a letra C.

Gabarito: Letra C

---

19. (CEBRASPE/SERPRO/2021) Com relação a conceitos de sistemas de arquivos distribuídos e sistemas de indexação, julgue o item a seguir.

Cada instância de elasticsearch é um nó; uma coleção de vários nós que podem trabalhar em harmonia formam um cluster elasticsearch.

Comentários:

O Elasticsearch funciona numa abordagem de computação distribuída. Nesse sentido, a questão está correta - cada instância do Elasticsearch representa um nó dentro e, ao agregarmos vários nós, criamos o que é chamado de *clsuter* Elasticsearch.

Gabarito: Correto

---

20. (FUNDATEC/PGE RS/2021) Elasticsearch é um mecanismo distribuído de busca e análise de dados muito utilizado atualmente. Ele usa um índice invertido, que é projetado para permitir buscas textuais muito rápidas. Quanto ao seu modelo de dados, o Elasticsearch pode ser classificado como:

a) Orientado a colunas.
b) Orientado a documentos.
c) Orientado a grafos.
d) Banco de dados objeto-relacional.
e) Banco de dados XML.

Comentários:

O Elasticsearch funciona duplamente - quando atua como um banco de dados, ele armazena os dados dentro de índices, no formato de documentos JSON. Portanto, o Elasticsearch pode ser considerado um banco de dados orientado a documentos.

Gabarito: Letra B

---

21. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Elasticsearch, julgue o item que segue.

Determinado cluster do Elasticsearch só pode ser considerado de alta disponibilidade se tiver ao menos 2 opções de nó mestre, para garantir que pelo menos um esteja operando em qualquer determinado momento.

Comentários:

Perfeito! Para ser considerado como de alta disponibilidade, um cluster deve ter ao menos 3 nós com o papel master, sendo que ao menos 2 não devem ser do tipo voting-only.

Gabarito: Correto

---

22. (Inédita/Prof. Felipe Mathias) Determinada entidade utiliza o Elasticsearch como forma de armazenamento e pesquisa dos *logs*. Determinado *log* foi armazenado com erro, e precisa ter parte de seu corpo atualizado. Assinale a alternativa que traz o método e o *endpoint* correto a serem determinados no comando:

a) GET e _update
b) PUT e _update
c) PUT e _doc
d) POST e _update
e) POST e _doc

Comentários:

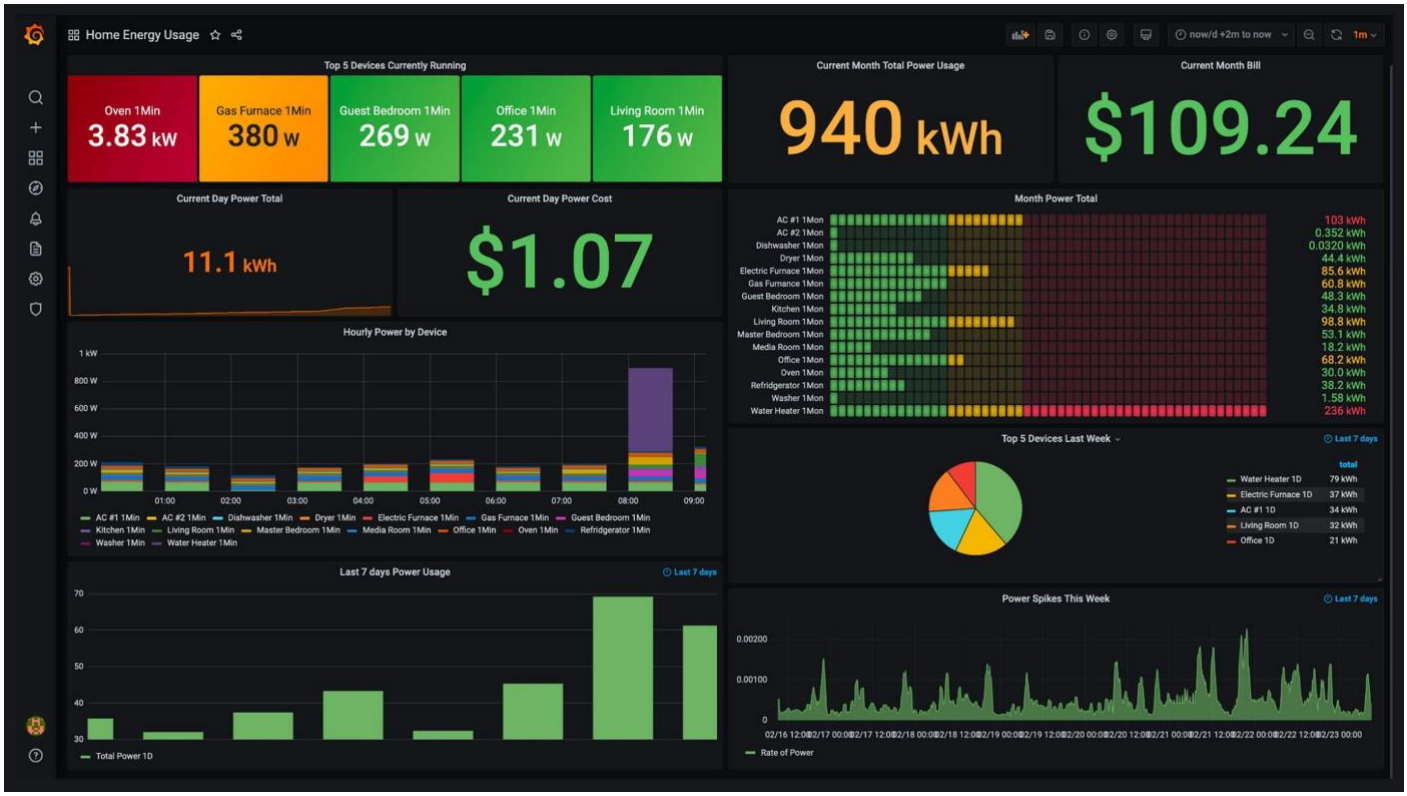
Como a atualização é parcial, isso é, só precisamos modificar parte do documento já indexado, o método correto é o POST, aliado ao *endpoint* `_update`. Portanto, a alternativa correta é a letra D.

Gabarito: Letra D

GRAFANA

Conceitos Gerais

Grafana é um software open source, que permite a consulta, visualização, criação de alertas e a análise exploratória de métricas e *logs*, independentemente do local de armazenamento. Seu funcionamento é direcionado para os TSDB (*time series database*), porém possui ferramentas de *plugins* que permitem integração com bancos de dado relacionais, NoSQL, ferramentas de *ticket* e *pipelines* CI/CID, como o GitLab. Com isso, ele consegue abranger uma grande gama de fontes de dados.



Acima, um dashboard no Grafana

O Grafana oferece um conjunto de aplicações integradas com diferentes objetivos, de forma a criar um ecossistema mais completo. As aplicações que você deve conhecer são:

- Grafana Open Source: é a aplicação principal, destinada à criação de visualizações e suporte a *analytics*. Ela permite a criação de tabelas, gráficos, *dashboards*, entre outros.
- Grafana Loki: é um conjunto de componentes que podem ser agregados em uma pilha completa de *logs*.
- Grafana Tempo: é uma ferramenta de back-end destinada ao rastreamento de dados de alto volume.

- Grafana Mimir: ferramenta de armazenamento pra longo períodos, integrada ao Prometheus.
- Grafana OnCall: aplicação de gerenciamento de respostas a incidentes.
- Grafana Cloud: plataforma de *logging* e métricas operando num sistema de nuvem, através de aplicações SaaS
- Grafana Enterprise: a versão de distribuição comercial do Grafana, que inclui aplicações extras, em relação à versão open source.

Para essa aula, o foco do nosso estudo girará em torno do Grafana Open Source, com menção às outras aplicações quando necessário.

QUESTÃO DE PROVA

(VUNESP/CIJUN/2023) No contexto da ferramenta Grafana, o Grafana Loki é

a) um navegador especializado em visualização de dashboards.
b) um sistema de agregação de logs.
c) uma ferramenta de gerenciamento de respostas a incidentes.
d) uma aplicação para realizar testes de desempenho em nuvem.
e) uma ferramenta de integração do Grafana com o Microsoft Active Directory.

Comentários:

O Grafana Loki é um conjunto de componentes que permite a agregação de *logs*, formando uma pilha completa de informações - portanto, a alternativa correta é a letra B. Quanto às demais alternativas:

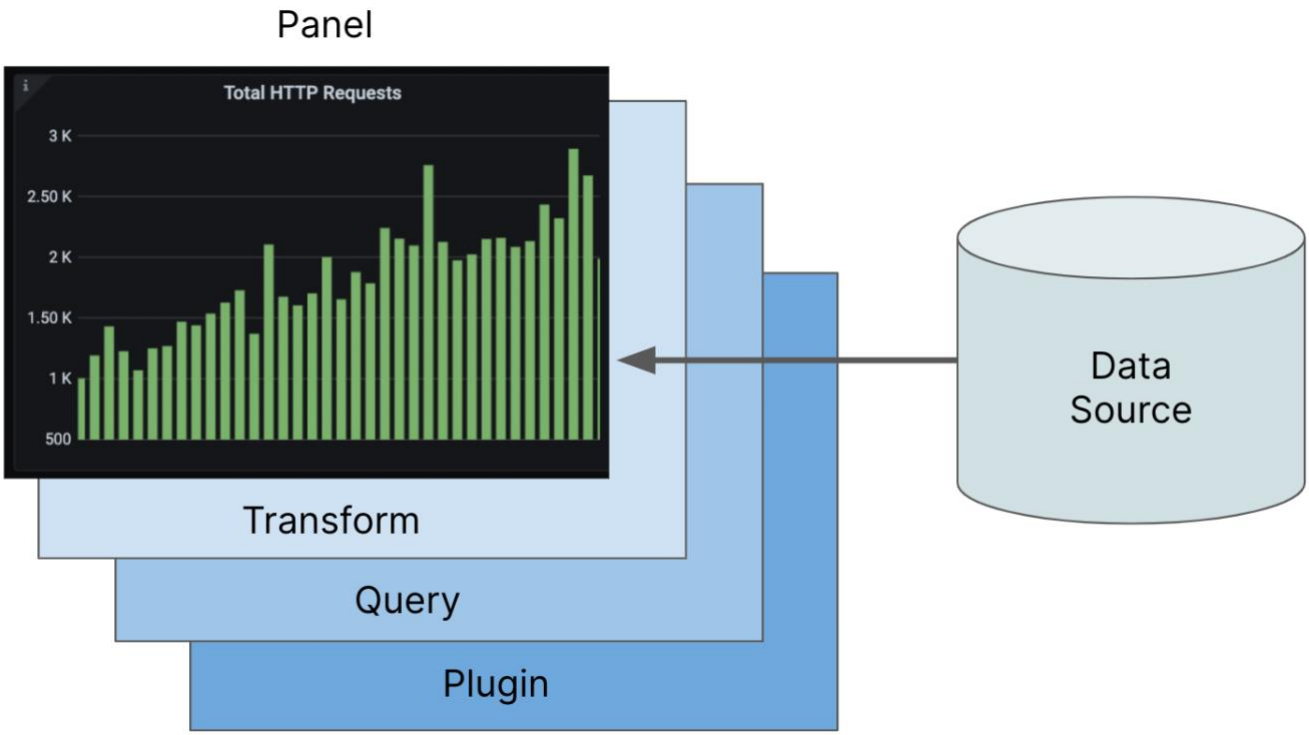
a) Refere-se ao Grafana Cloud.
b) É o nosso gabarito.
c) Refere-se ao Grafana OnCall.
d) Refere-se ao k6, uma ferramenta que foi adquirida Grafana Labs.
e) É uma das aplicações fornecidas pelo Grafana Enterprise.

Ficamos com a letra B como gabarito. (Gabarito: Letra B)

Dashboard

Um dashboard no Grafana consiste em painéis de visualizações de dados, com gráficos, mapas de calor, dados tabulares, *cards*, entre outros. Esses painéis são criados a partir da transformação de dados brutos, das fontes de dados, em visualizações, fornecendo uma abordagem diferente às informações.

O processo para a criação de uma visualização envolve a passagem dos dados por três "portões": um plugin, uma consulta (*query*) e uma transformação, que é opcional.



As origens de dados (*data sources*) são as fontes de onde o Grafana irá ler os dados. O Grafana oferece alta integração, então essas fontes podem ser bancos de dado SQL e NoSQL, o Grafana Loki ou Mimir, Prometheus, APIs baseadas em JSON, ou simples arquivos CSV. Inclusive, as diferentes fontes de dados podem formar uma visualização no *dashboard*.

Os Plugins são ferramentas que trazem novas capacidades ao Grafana. Existem diversos tipos de *plugins*, mas nessa *pipeline* de extração de dados da origem, destacam-se os *plugins* direcionados às origens de dados. Seu trabalho é pegar uma consulta (*query*) realizada e recuperar os dados da origem, de acordo com os parâmetros passados.

As consultas (*queries*), por sua vez, são formas de filtrarmos a quantidade de dados retornados. Muitas vezes queremos métricas ou *logs* específicos, e não é interessante retornar toda a base de

dados da origem. Para isso, usamos as consultas, que definem os parâmetros do que será retornado.
Como temos várias origens, os formatos dos dados consultados podem variar - atualmente, o Grafana suporta cerca de 155 tipos de dados diferentes, como CSV, JSON e XML. Para não termos "problemas" na hora da criação das visualizações, devido à compatibilidade de formatos, os dados são unificados em uma estrutura chamada de *data frame*.

Feita as consultas e com os dados "em mãos", podemos, opcionalmente, fazer as transformações. Elas são aplicadas quando o formato dos dados não atendem aos requisitos definidos, onde manipulamos os dados. Podemos fazer concatenações, por exemplo, juntando campos de nome e sobrenome em um campo de 'nome completo'.

Com tudo pronto, começamos a criar os painéis, que são os elementos que compõem os *dashboards*. Cada painel fornecerá um tipo de visualização diferente. Os painéis funcionam como contêineres, expondo as visualizações e fornecendo controles de manipulação dinâmicos. Dessa forma, podemos ter gráficos mais customizáveis.

QUESTÕES COMENTADAS

Grafana

01. (CPCON UEPB/UEPB/2023) O Grafana é uma poderosa ferramenta de visualização de dados que permite criar painéis interativos e personalizados. Com suporte a várias fontes de dados, o Grafana oferece recursos avançados para monitorar e analisar métricas, facilitando a compreensão e tomada de decisões baseadas em dados. Considerando o uso do Grafana, avalie as proposições a seguir:

I- O Grafana é uma ferramenta open-source.
II- O Grafana é uma ferramenta de visualização de dados que suporta a integração com várias fontes de dados.
III- O Grafana possui suporte nativo para a criação de alertas em tempo real.

A partir do uso do Grafana, é CORRETO o que se afirma em:

a) I e III apenas.
b) I e II apenas.
c) II apenas.
d) I, II e III.
e) II e III apenas.

Comentários:

Vamos analisar cada item.

I. Certo. O Grafana é open source.
II. Certo. O Grafana se integra com várias fontes - Prometheus, bancos de dados relacionais, MongoDB, entre outros.
III. Errado. Não há suporte nativo para a criação de alertas no Grafana. Para criarmos alertas, é necessário usar o Grafana OnCall.

Portanto, corretos os itens I e II.

Gabarito: Letra B

---

02. (VUNESP/CIJUN/2023) No contexto da ferramenta Grafana, o Grafana Loki é

a) um navegador especializado em visualização de dashboards.
b) um sistema de agregação de logs.

c) uma ferramenta de gerenciamento de respostas a incidentes.
d) uma aplicação para realizar testes de desempenho em nuvem.
e) uma ferramenta de integração do Grafana com o Microsoft Active Directory.

Comentários:

O Grafana Loki é um conjunto de componentes que permite a agregação de logs, formando uma pilha completa de informações - portanto, a alternativa correta é a letra B. Quanto às demais alternativas:

a) Refere-se ao Grafana Cloud.
b) É o nosso gabarito.
c) Refere-se ao Grafana OnCall.
d) Refere-se ao k6, uma ferramenta que foi adquirida Grafana Labs.
e) É uma das aplicações fornecidas pelo Grafana Enterprise.

Portanto, correta a letra B.

Gabarito: Letra B

---

03. (FCC/ISS MANAUS/2019) O Grafana é uma plataforma para a criação de dashboards de monitoração de sistemas computacionais. Dentre os vários tipos de dashboards disponibilizados no site oficial do Grafana https://grafana.com/dashboards, existe o

a) Trackmap.
b) 3D Box.
c) 2D Gauge.
d) Windmap.
e) Status Light.

Comentários:

Questão um pouco absurda da FCC - esse conhecimento é extremamente específico e avançado e, a meu ver, desnecessário. Temos 153 páginas de *dashboards*, cada página com aproximadamente 40 opções diferentes. A resposta correta para a questão é Trackmap - mas essa ferramenta é, hoje em dia, deprecada e não é mais utilizada.

Gabarito da banca: Letra A
Gabarito atual: Anulada

QUESTÕES COMENTADAS

Grafana

01. (CPCON UEPB/UEPB/2023) O Grafana é uma poderosa ferramenta de visualização de dados que permite criar painéis interativos e personalizados. Com suporte a várias fontes de dados, o Grafana oferece recursos avançados para monitorar e analisar métricas, facilitando a compreensão e tomada de decisões baseadas em dados. Considerando o uso do Grafana, avalie as proposições a seguir:

I- O Grafana é uma ferramenta open-source.
II- O Grafana é uma ferramenta de visualização de dados que suporta a integração com várias fontes de dados.
III- O Grafana possui suporte nativo para a criação de alertas em tempo real.

A partir do uso do Grafana, é CORRETO o que se afirma em:

a) I e III apenas.
b) I e II apenas.
c) II apenas.
d) I, II e III.
e) II e III apenas.

Comentários:

Vamos analisar cada item.

I. Certo. O Grafana é open source.
II. Certo. O Grafana se integra com várias fontes - Prometheus, bancos de dados relacionais, MongoDB, entre outros.
III. Errado. Não há suporte nativo para a criação de alertas no Grafana. Para criarmos alertas, é necessário usar o Grafana OnCall.

Portanto, corretos os itens I e II.

Gabarito: Letra B

---

02. (VUNESP/CIJUN/2023) No contexto da ferramenta Grafana, o Grafana Loki é

a) um navegador especializado em visualização de dashboards.
b) um sistema de agregação de logs.

c) uma ferramenta de gerenciamento de respostas a incidentes.
d) uma aplicação para realizar testes de desempenho em nuvem.
e) uma ferramenta de integração do Grafana com o Microsoft Active Directory.

Comentários:

O Grafana Loki é um conjunto de componentes que permite a agregação de logs, formando uma pilha completa de informações - portanto, a alternativa correta é a letra B. Quanto às demais alternativas:

a) Refere-se ao Grafana Cloud.
b) É o nosso gabarito.
c) Refere-se ao Grafana OnCall.
d) Refere-se ao k6, uma ferramenta que foi adquirida Grafana Labs.
e) É uma das aplicações fornecidas pelo Grafana Enterprise.

Portanto, correta a letra B.

Gabarito: Letra B

---

03. (FCC/ISS MANAUS/2019) O Grafana é uma plataforma para a criação de dashboards de monitoração de sistemas computacionais. Dentre os vários tipos de dashboards disponibilizados no site oficial do Grafana https://grafana.com/dashboards, existe o

a) Trackmap.
b) 3D Box.
c) 2D Gauge.
d) Windmap.
e) Status Light.

Comentários:

Questão um pouco absurda da FCC - esse conhecimento é extremamente específico e avançado e, a meu ver, desnecessário. Temos 153 páginas de *dashboards*, cada página com aproximadamente 40 opções diferentes. A resposta correta para a questão é Trackmap - mas essa ferramenta é, hoje em dia, deprecada e não é mais utilizada.

Gabarito da banca: Letra A
Gabarito atual: Anulada

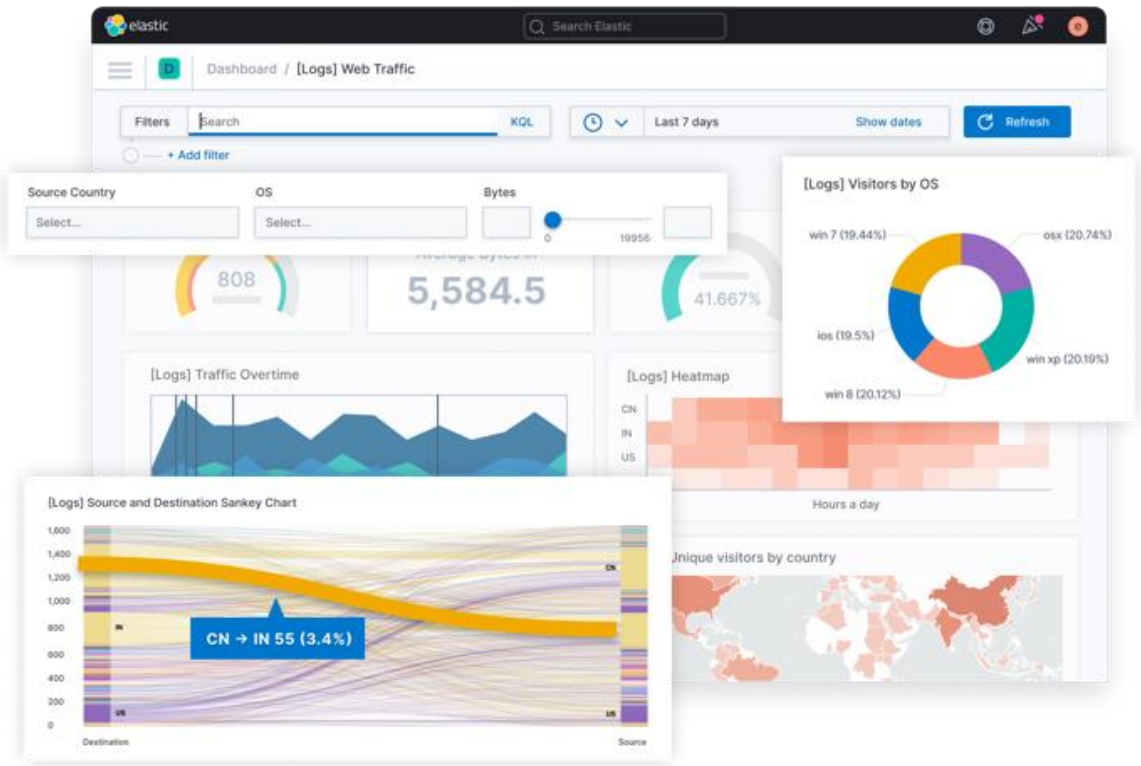
GABARITO

1. Letra B
2. Letra B
3. Letra A

KIBANA

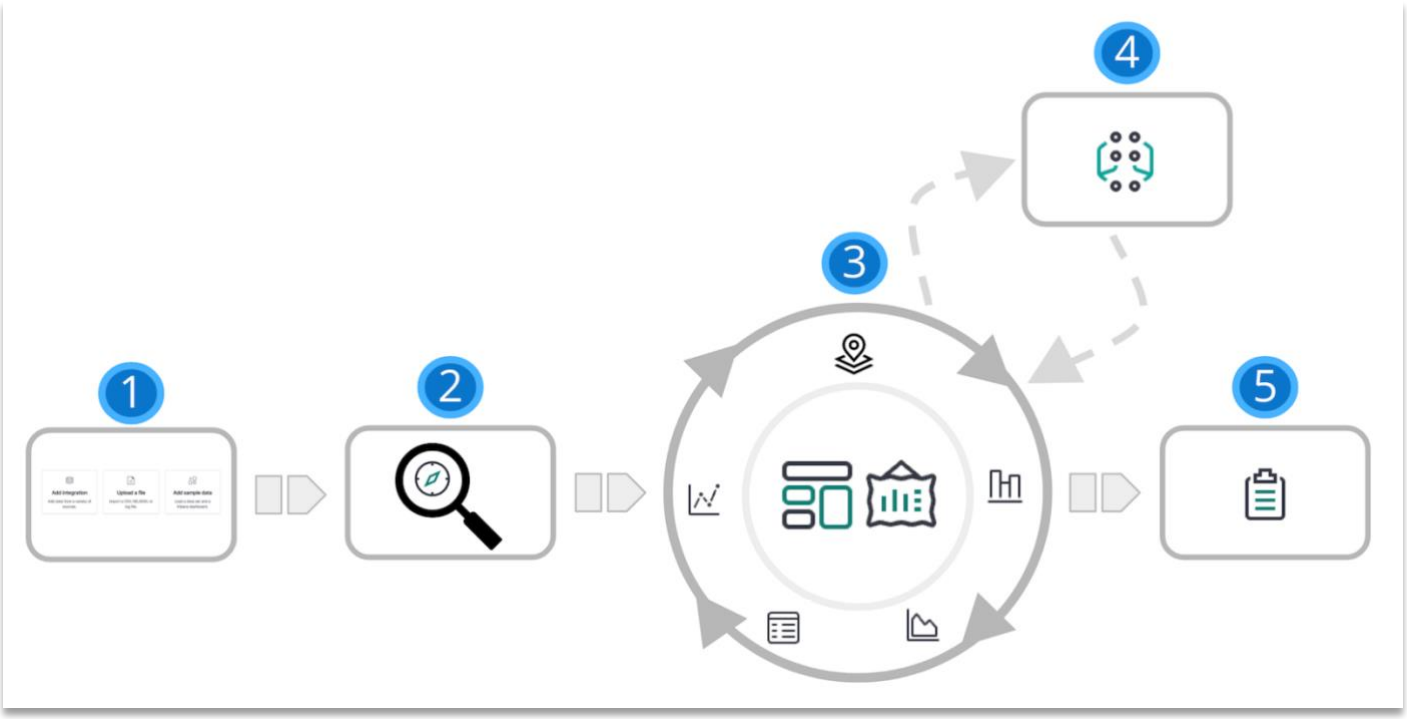
Conceitos Gerais

O Kibana é uma ferramenta de interface de usuário (User Interface - UI), que permite a criação de visualizações e navegação dos dados gerados e gerenciados em toda a *stack* ELK. Ele atua no topo, a "cereja no topo do bolo", que facilita toda a parte de análises com gráficos e visualizações dos mais variados tipos.



Um dos pontos fortes do Kibana é o suporte a análises, com o Kibana Analytics. Ele cria um fluxo para a criação de visualizações e análises a partir de 5 etapas:

1) Adicionar os dados: Kibana aceita diversas integrações de entrada, além da integração padrão com a *stack* Elastic.
2) Explorar: com a ferramenta `Discover`, é possível pesquisar por *insights* ocultos, relacionamentos e outras informações nos dados
3) Visualizar: o Kibana fornece muitas formas de criar visualizações para os dados, partindo dos Dashboards.
4) Modelar o comportamento dos dados (opcional): é possível usar Machine Learning para modelar o comportamento dos dados, fazendo previsões e análises.
5) Compartilhar: com o modelo de *analytics* completo, é possível compartilhá-lo com outros usuários e analistas.



QUESTÃO DE PROVA

(FGV/TRT 13/2022) A plataforma Elasticsearch integra diversos produtos compondo a Elastic Stack, cada um desses produtos possui funcionalidades bem definidas.

O produto nativo da Stack que permite a visualização de dados é

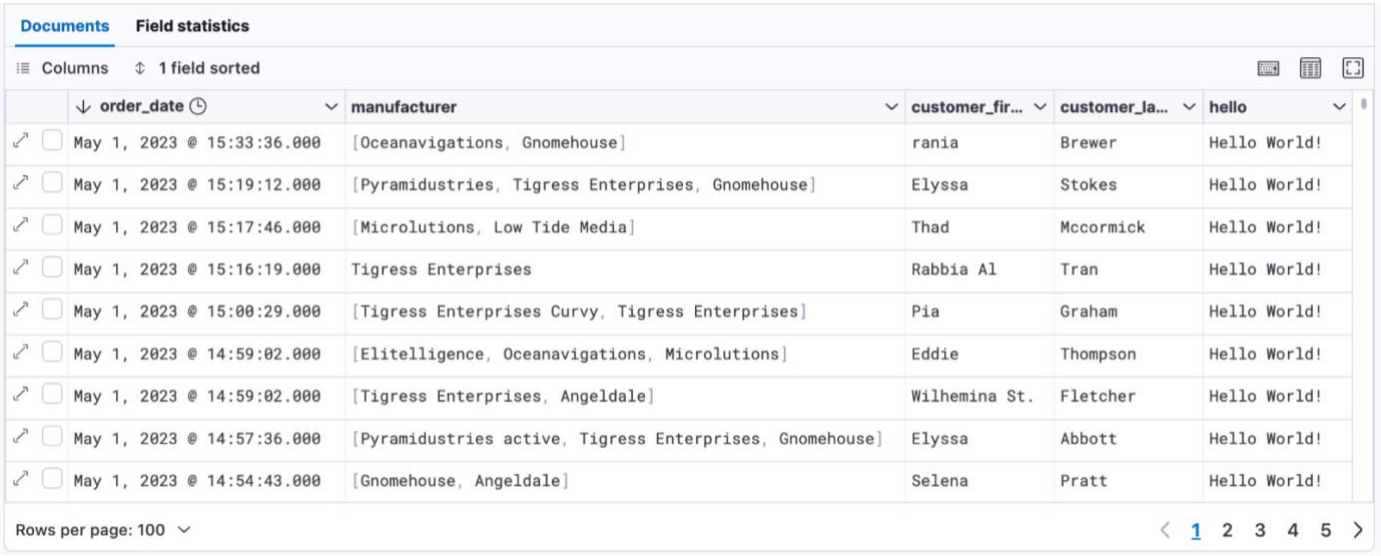
a) dashviewer.
b) inforgram.
c) tableau.
d) kibana.
e) x-pack.

Comentários:

Na Elastic Stack, o produto destinado à visualização de dados é o Kibana. (Gabarito: Letra D)

Kibana Discover

O Kibana Discover é o componente da arquitetura da aplicação responsável por fazer uma análise exploratória dos dados. Ele fornece uma Interface de Usuário que permite a pesquisa e filtragem de dados, retornando informações em uma visualização tabular. Também é possível salvar cada uma dessas pesquisas para apresentá-las posteriormente, como visualizações em *dashboards*.



Para selecionarmos um conjunto de dados para a aplicação do Discover, é necessário o uso de Data Views. Uma Data View é uma forma do Kibana acessar os dados armazenados no Elasticsearch. Ela pode apontar para um ou mais índices, fluxos de dados ou apelidos (*alias*) de índices. Para poder criar uma Data View, é necessário que o usuário tenha privilégios `Data View Management` no Kibana, e `view_index_metadata` no Elasticsearch.

Também é possível trabalhar com dados adicionados ou combinações de dados, implementandos diferentes scritps, através dos campos de tempo de execução, ou *runtime fields*. Ele é um campo que é calculado no momento da consulta, em vez de ser armazenado no Elasticsearch. Isso permite que os usuários criem novos campos dinamicamente sem modificar os dados subjacentes ou os mapeamentos de índice.

(Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre o programa Kibana, julgue o item.

Os campos de dados chamados de *runtime fields* são campos de tempo de execução, que permitem interações dinâmicas e a inserção de scripts no contexto do Kibana Discover.

Comentários:

Perfeito! Os *runtime fields* são campos cujos valores são calculados "na hora", permitindo a inserção de *scripts* que manipulem dados. (Gabarito: Correto)

Linguagens de Consulta

Dentro do Kibana, podemos usar três linguagens de consulta diferentes. Vamos vê-las.

EL|QL

A primeira das linguagens é o EL|QL, ou ElasticSearch Query Language. Ela é uma linguagem de consultas criada para trabalhar com o Elasticsearch, oferecendo capacidades para retornar dados estruturados e consultas analíticas. Sua sintaxe é muito similar ao SQL, veja:

```bash
SELECT campo1, campo2 FROM nome_índice
```

A EL|QL é uma linguagem que permite filtragem dos dados, agregações, operações complexas, como junções e uniões, muito similar ao SQL. A sua aplicação principal se dá no contexto da descoberta dos dados, junto do Kibana Discover.

Kibana Query Language

A segunda linguagem, e mais relevante para o nosso estudo, é uma linguagem específica do Kibana - a Kibana Query Language (KQL). Ela é uma linguagem de consulta simples, baseada em texto, tendo como único objetivo a filtragem de dados. Por esse motivo, é impossível fazer funções complexas na KQL - ou seja, agregações, junções e outros.

O KQL é usado para filtrar os dados em três diferentes abordagens:

- Campos que existam nos dados
- Campos que sejam compatíveis com determinado valor
- Campos que estejam dentro de determinado intervalo

Vamos ver as três abordagens.

Filtros de existência

Aqui nosso objetivo é verificar se determinado campo ou valor existe nos dados. Para filtrar se determinado valor indexado existe, usa-se o operador asterisco `*`. Vou lhe trazer os exemplos da documentação oficial do Kibana, já que as bancas tendem a usar justamente esses exemplos para criar as questões.

Nesse exemplo abaixo, estamos buscando documentos onde o valor `http.request.method` exista e seja indexado.

```bash
http.request.method: *
```

Filtros por valor

Os filtros de compatibilidade buscam por uma correspondência entre o valor que queremos encontrar, e valores contidos nos documentos (dados). Pense nos documentos como dados em pares de campo e valor. Na filtragem anterior, de existência, tivemos uma orientação aos campos - estamos verificando a existência dos campos. Aqui a análise é direcionada para o valor.

Por exemplo, se quisermos encontrar todos os documentos onde o campo `http.request.method` tem o valor `GET` associado, usamos a seguinte sintaxe:

```bash
http.request.method: GET
```

A especificação de um campo é opcional. Por exemplo, se quisermos retornar todos os documentos que tenham o valor 'Estratégia', independentemente do campo que está associado, usamos a seguinte sintaxe:

```bash
Estratégia
```

Também é possível negar uma consulta, começando a consulta com o comando **`NOT`**. Nesse caso, nosso objetivo é retornar todos os documentos que não atendam a determinada especificação. Pegando o mesmo exemplo que fizemos no começo dessa seção, se quisermos retornar todos os campos `http.request.method` que não possuem o valor `GET`, usaremos:

```bash
NOT http.request.method: GET
```

QUESTÃO DE PROVA

(FGV/Câmara dos Deputados/2023) A Kibana Query Language (KQL) é uma linguagem de consulta simples baseada em texto para filtrar dados.

De acordo com essa linguagem, para encontrar os documentos que possuem um campo http.request.method com valor diferente de GET, deve ser utilizada a seguinte sintaxe:

a) NOT http.request.method: GET
b) http.request.method <> 'GET'
c) http.request.method != GET
d) ! http.request.method: GET
e) http.request.method NOT "GET"

Comentários:

Veja que essa questão trouxe a literalidade da documentação, copiando o exemplo. Para encontrarmos valores diferentes de GET precisamos fazer uma negação da sintaxe - portanto, o comando será NOT http.request.method: GET. (Gabarito: Letra A)

Filtro por intervalos

O filtro de intervalos procura buscar valores dentro de determinado intervalo de valores. Aqui direcionamos a pesquisa para os campos, mas o que será analisado serão seus respectivos valores. Por exemplo, podemos pesquisar por valores para o campo http.response.bytes maiores que 1000 e menores ou igual a 2000, na seguinte sintaxe:

BASH

```
http.response.bytes > 10000 and http.response.bytes <= 20000
```

É possível pesquisar intervalos para valores em *strings*, não apenas numéricos. Por exemplo, podemos pesquisar todos os documentos produzidos durante determinado tempo, com base no momento da pesquisa. Para isso usamos a diretiva `@timestamp`, seguido do intervalo de tempo que queremos retornar.

```bash
@timestamp < now-24h #retorna documentos criados nas últimas 24h
@timestamp < now-2d #retorna documentos criados nos últimos 2 dias
@timestamp < now-2w #retorna documentos criados nas últimas 2 semanas
```

Pesquisas com elementos coringa

As pesquisas do KQL são feitas de forma literal. Isso é, se colocarmos "Estratégia", só retornaremos como resultado os campos ou valores que contenham "Estratégia". Pode ser necessário pesquisar por padrões - por exemplo, todos os *status* de erro numa resposta a um método HTTP que comecem com 4 (como erro 404, 405, entre outros). Para isso, usamos o operador `*`, que corresponderá a 0 ou múltiplos caracteres quaisquer.

Pegando nosso exemplo, precisaríamos pesquisar um campo http.response.status_code, para retornarmos as respostas de requisições, cujo valor seja igual a 4*. Ficará da seguinte forma:

```bash
http.request.status_code: 4*
```

Ou, alternativamente, poderíamos pesquisar erros em requisições que não sejam erros de cliente (ou seja, 4xx). Basta negarmos a sintaxe acima:

```bash
NOT http.request.status_code: 4*
```

Lucene Query Syntax

A linguagem de consulta do Apache Lucene, que é um motor de pesquisas sobre o qual o Elasticsearch é desenvolvido, é uma alternativa ao KQL. Ele fornece a possibilidade de consultas mais complexa e avançadas, como pesquisas por expressões regulares (*regex*) e compatibilidade difusa de termos.

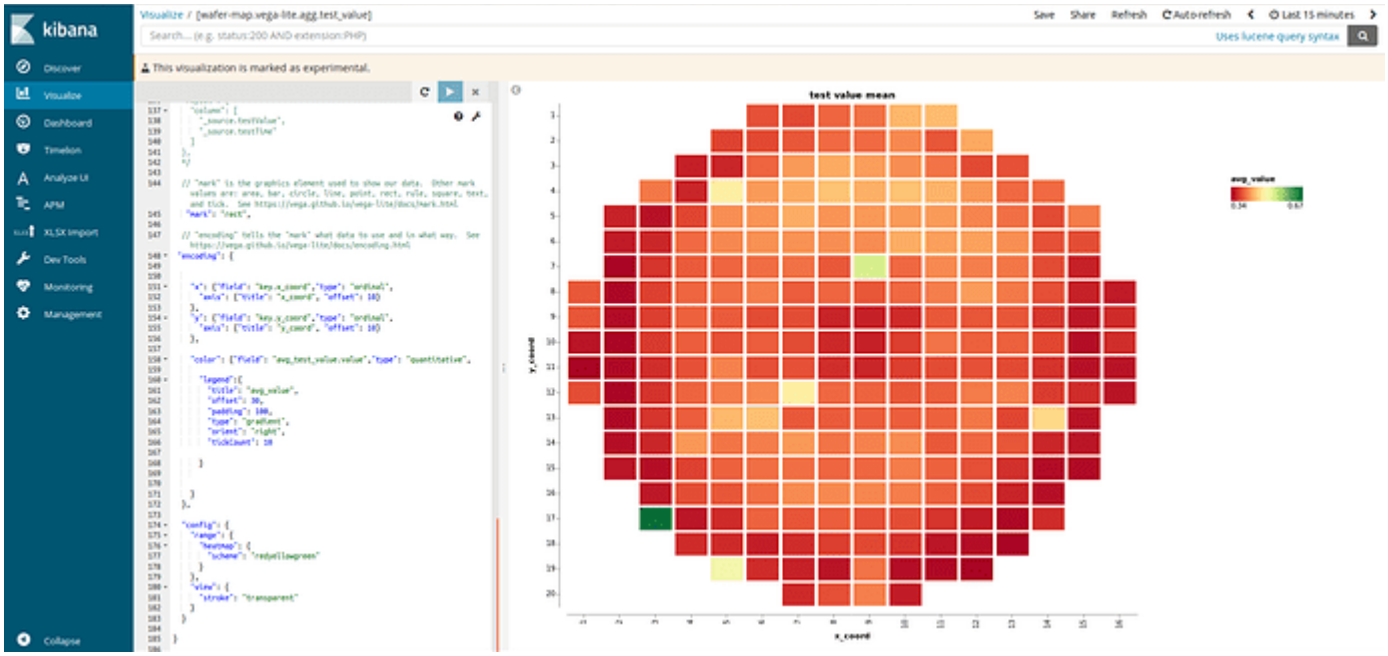
No exemplo abaixo, consultamos todos os campos que contenham 'status' e cujo valor associado varie entre 400 e 499.

BASH

```
status:[400 to 499]
```

Visualizações

As visualizações são a "cereja no bolo" do Kibana, já que elas que criam formas de visualizarmos a informação, fazendo comparações, grafismos, e, de modo geral, facilitando as diferentes abordagens de *analytics*. Outras ferramentas capazes de criar visualizações que costumam cair em provas, como o Power BI, tem suas formas de visualização exploradas - então é exatamente isso que veremos nessa seção: os diferentes gráficos criados pelo Kibana.



Visualização	Descrição
Gráfico de Área	Exibe dados quantitativos com a área entre o eixo e a linha preenchida com cor, mostrando tendências ao longo do tempo.
Gráfico de Linha	Traça pontos de dados ao longo de uma linha, ideal para mostrar tendências ao longo do tempo.
Gráfico de Barras	Compara diferentes grupos ou rastreia mudanças ao longo do tempo, disponível nos formatos horizontal ou vertical.
Gráfico de Pizza	Mostra proporções e porcentagens entre um conjunto de categorias, útil para ilustrar partes de um todo.
Tabela de Dados	Exibe dados brutos em um formato tabular para análise detalhada.
Métrica	Exibe um único número representando uma métrica de resumo, como contagem ou média.
Medidor (Gauge)	Mostra progresso ou um valor único dentro de um intervalo, tipicamente usado para acompanhamento de KPIs.
Markdown	Permite adicionar texto descritivo e informações usando sintaxe Markdown.

Nuvem de Tags	Visualiza dados de texto em formato de nuvem, onde o tamanho de cada palavra indica sua frequência.
Mapa de Calor	Exibe a densidade e distribuição dos dados em um espaço bidimensional.
Vega	Permite a criação de visualizações personalizadas usando a gramática Vega e Vega-Lite para usuários avançados.

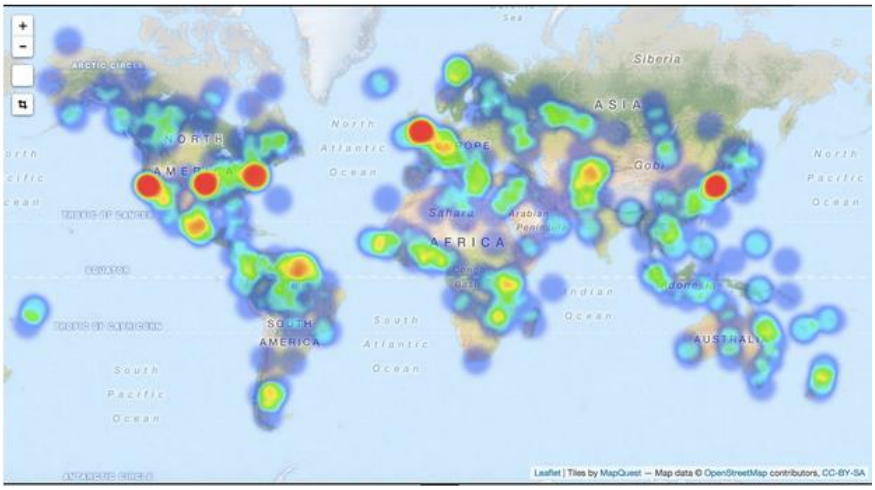
QUESTÃO DE PROVA

(Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Kibana, julgue o item abaixo.

As visualizações no Kibana são uma parte essencial na sua aplicação. Quando se deseja visualizar informações por posicionamento geográfico, uma das abordagens a ser utilizada pode ser o uso de mapas de calor aplicados a mapas geográficos.

Comentários:

Perfeito! O Kibana permite a combinação de diferentes mapas. O mapa de calor com distribuição geográfica pode ser interessante em análises de rede de alta abrangência, *logs* de sites, entre outros. Veja uma visualização desse gênero:



O gabarito é correto, portanto. (Gabarito: Correto)

QUESTÕES COMENTADAS

Kibana

01. (FGV/Câmara dos Deputados/2023) A Kibana Query Language (KQL) é uma linguagem de consulta simples baseada em texto para filtrar dados.

De acordo com essa linguagem, para encontrar os documentos que possuem um campo http.request.method com valor diferente de GET, deve ser utilizada a seguinte sintaxe:

a) NOT http.request.method: GET
b) http.request.method <> 'GET'
c) http.request.method != GET
d) ! http.request.method: GET
e) http.request.method NOT "GET"

Comentários:

Como queremos valores diferentes de GET, devemos pesquisar pelo método e fazer uma negação da pesquisa. Essa negação é feita pelo uso de NOT na frente da sintaxe - ficando, portanto `NOT http.request.method: GET`.

Gabarito: Letra A

---

02. (FGV/Câmara dos Deputados/2023) No contexto das ferramentas e conceitos relacionados à análise de logs e ao monitoramento de desempenho, assinale a afirmativa incorreta.

a) Logstash é uma ferramenta de processamento de logs que pode coletar, transformar e encaminhar dados para vários destinos.
b) Kibana é uma plataforma de visualização de dados que permite aos usuários criar dashboards personalizados para visualizar dados de logs e métricas.
c) Logstash suporta uma variedade de plugins de entrada, filtros e saídas, permitindo a personalização do processamento de logs.
d) Kibana permite a realização de pesquisas avançadas, análise de dados e visualização de dados em tempo real.
e) Logstash e Kibana são utilizados exclusivamente para análise de logs e não podem ser integrados com outras ferramentas de monitoramento e observabilidade.

Comentários:

Vamos analisar cada uma dessas alternativas - procurando a afirmativa incorreta.

a) Certo. O Logstash é uma ferramenta que faz a coleta, transformação e encaminhamento dos dados, principalmente na ELK stack.
b) Certo.
c) Certo
d) Certo.
e) Errado.

Portanto, a alternativa incorreta é a letra E.

Gabarito: Letra E

---

03. (FGV/TRT 16/2022) A ferramenta de desenvolvimento e depuração, utilizada no Kibana 7.0, que permite que um analista faça parser de dados não estruturados presentes em diferentes tipos de arquivos de logs de servidores web ou de banco de dados se denomina

a) logprofiler.
b) logstach.
c) timelion.
d) beats.
e) grok.

Comentários:

A ferramenta que faz um *parser* a partir de expressões regulares (*regex*), permitindo transformar dados não estruturados em estruturados para uma análise mais precisa - que inclusive se repete em outros componentes da *stack* - é o grok.

Gabarito: Letra E

---

04. (FGV/TRT 13/2022) A plataforma Elasticsearch integra diversos produtos compondo a Elastic Stack, cada um desses produtos possui funcionalidades bem definidas.

O produto nativo da Stack que permite a visualização de dados é

a) dashviewer.
b) inforgram.
c) tableau.
d) kibana.
e) x-pack.

Comentários:

A ferramenta da *stack* que é responsável pelas visualizações é o Kibana. Quanto aos demais programas apresentados:

- Dashviewer: ferramenta de código fechado, destinado à análise de dados.
- Infogram: ferramenta online para criação de infográficos.
- Tableau: plataforma de análise visual, que permite análise exploratória dos dados
- X-pack: extensão do Elasticsearch, adicionando recursos de segurança.

Portanto, correta a letra D.

Gabarito: Letra E

---

05. (COMPERVE/UFRN/2020) O ELK Stack diz respeito a um conjunto de projetos relacionados ao processamento, visualização e busca de informações, utilizados em diversos contextos nas aplicações de hoje em dia, em especial aquelas destinadas a lidar com grandes volumes de dados. Nesse contexto, os projetos Kibana e Beats implementam mecanismos de

a) Visualização e Agente para servir dados, respectivamente.
b) Busca e Visualização, respectivamente.
c) Pipeline de ingestão e Agente para servir dados, respectivamente.
d) Busca e Pipeline de ingestão, respectivamente.

Comentários:

O Kibana é um componente destinado a criar visualizações, com base em dados armazenados no Elasticsearch. Já o Elastic Beats é uma plataforma para agentes de dados, que visam fazer entregas de dados de programas e aplicações de origem para o Logstash ou Elasticsearch. A alternativa que aponta corretamente a descrição é a letra A.

Gabarito: Letra A

---

06. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre o programa Kibana, julgue o item.

Os campos de dados chamados de runtime fields são campos de tempo de execução, que permitem interações dinâmicas e a inserção de scripts no contexto do Kibana Discover.

Comentários:

Perfeito! Os *runtime fields* são campos cujos valores são calculados "na hora", permitindo a inserção de scripts que manipulem dados.

Gabarito: Correto

LISTA DE QUESTÕES

Kibana

01. (FGV/Câmara dos Deputados/2023) A Kibana Query Language (KQL) é uma linguagem de consulta simples baseada em texto para filtrar dados.

De acordo com essa linguagem, para encontrar os documentos que possuem um campo http.request.method com valor diferente de GET, deve ser utilizada a seguinte sintaxe:

a) NOT http.request.method: GET
b) http.request.method <> 'GET'
c) http.request.method != GET
d) ! http.request.method: GET
e) http.request.method NOT "GET"

02. (FGV/Câmara dos Deputados/2023) No contexto das ferramentas e conceitos relacionados à análise de logs e ao monitoramento de desempenho, assinale a afirmativa incorreta.

a) Logstash é uma ferramenta de processamento de logs que pode coletar, transformar e encaminhar dados para vários destinos.
b) Kibana é uma plataforma de visualização de dados que permite aos usuários criar dashboards personalizados para visualizar dados de logs e métricas.
c) Logstash suporta uma variedade de plugins de entrada, filtros e saídas, permitindo a personalização do processamento de logs.
d) Kibana permite a realização de pesquisas avançadas, análise de dados e visualização de dados em tempo real.
e) Logstash e Kibana são utilizados exclusivamente para análise de logs e não podem ser integrados com outras ferramentas de monitoramento e observabilidade.

03. (FGV/TRT 16/2022) A ferramenta de desenvolvimento e depuração, utilizada no Kibana 7.0, que permite que um analista faça parser de dados não estruturados presentes em diferentes tipos de arquivos de logs de servidores web ou de banco de dados se denomina

a) logprofiler.
b) logstach.
c) timelion.
d) beats.
e) grok.

04. (FGV/TRT 13/2022) A plataforma Elasticsearch integra diversos produtos compondo a Elastic Stack, cada um desses produtos possui funcionalidades bem definidas.

O produto nativo da Stack que permite a visualização de dados é

a) dashviewer.
b) inforgram.
c) tableau.
d) kibana.
e) x-pack.

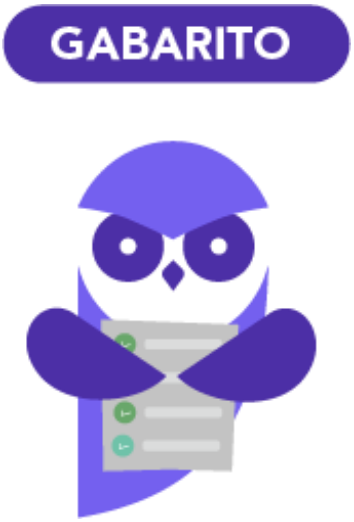
05. (COMPERVE/UFRN/2020) O ELK Stack diz respeito a um conjunto de projetos relacionados ao processamento, visualização e busca de informações, utilizados em diversos contextos nas aplicações de hoje em dia, em especial aquelas destinadas a lidar com grandes volumes de dados. Nesse contexto, os projetos Kibana e Beats implementam mecanismos de

a) Visualização e Agente para servir dados, respectivamente.
b) Busca e Visualização, respectivamente.
c) Pipeline de ingestão e Agente para servir dados, respectivamente.
d) Busca e Pipeline de ingestão, respectivamente.

06. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre o programa Kibana, julgue o item.

Os campos de dados chamados de runtime fields são campos de tempo de execução, que permitem interações dinâmicas e a inserção de scripts no contexto do Kibana Discover.

GABARITO



1. Letra A
2. Letra E
3. Letra E
4. Letra D
5. Letra A
6. Correto

LOGSTASH

Conceitos Gerais

O Logstash é um mecanismo de coleta de dados, de código aberto, com capacidades de operacionalizar *pipelines* de dados em tempo real (*stream pipelines*). O Logstash pode unificar dinamicamente dados de diferentes fontes, e normalizá-lo em várias destinações, fazendo etapas de processamento e tratamento nos dados, permitindo análises avançadas e visualizações.

(FGV/Câmara dos Deputados/2023 - Adaptada) No contexto das ferramentas e conceitos relacionados à análise de logs e ao monitoramento de desempenho, julgue o item abaixo.

Logstash é uma ferramenta de processamento de logs que pode coletar, transformar e encaminhar dados para vários destinos.

Comentários:

Perfeito! O Logstash coleta dados de diversas fontes, os transforma e os encaminha para diversos tipos de destinos diferentes. (Gabarito: Certo)

O Logstash é um dos componentes da pilha de programas destinados à análise de *log* e monitoramento de sistemas chamada de ELK stack - em conjunto com o Elasticsearch e o Kibana. O Logstash atua como um elemento inicial, com foco em ingestão e transformação dos dados. Ele é o primeiro contato dos dados e *logs* com a pilha.

De forma geral, a pilha funciona da seguinte forma:

LOGSTASH ⇒ ELASTICSEARCH ⇒ KIBANA

INDO MAIS FUNDO!

Funcionamento da ELK Stack:

O Logstash fará o processo de transformação, enriquecimento de dados, pré-processamento e ingestão geral dos dados. O Elasticsearch pegará esses dados e fará uma indexação e armazenamento, em formato de documentos, além de prover formas de pesquisa e análise dos dados. Por fim, o Kibana irá consultar os dados para criar visualizações e gráficos, facilitando a parte de *analytics* dos logs, criando uma forma completa de análise.

Só o entendimento desse "fluxograma" da pilha de programas já lhe permite responder questões. Veja uma questão, em que é possível encontrar a resposta somente com o conhecimento da posição do Logstash na cadeia.

(FCC/TRT 22/2022) Considere o código abaixo presente em um arquivo de configuração criado por um analista, que especifica qual plug-in será utilizado em determinado contexto e as configurações deste plug-in.

```
input { stdin { } }
output {
     elasticsearch { hosts => ["localhost:9200"] }
}
```

As configurações mostram que, quando esse arquivo for lido, será gerada uma saída para o Elasticsearch. Trata-se de um arquivo de configuração do

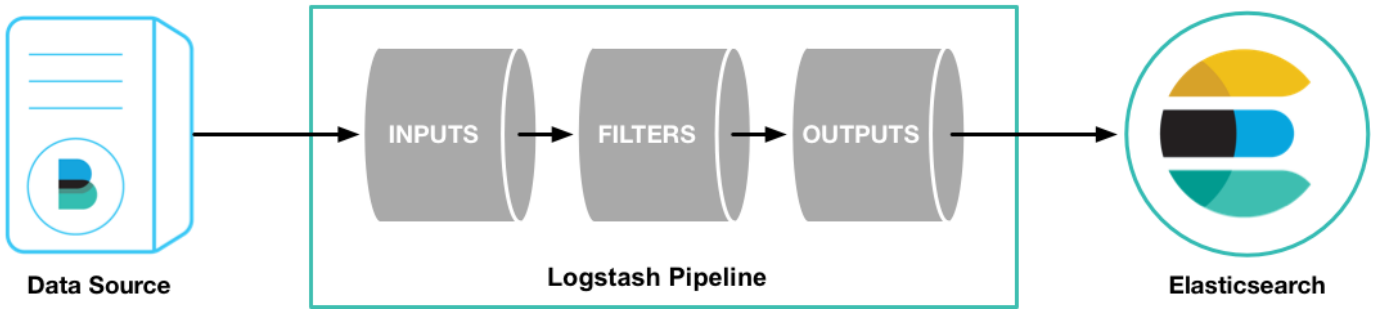
a) Kibana.
b) Rancher.
c) Jenkins.
d) Logstash.
e) ElasticData.

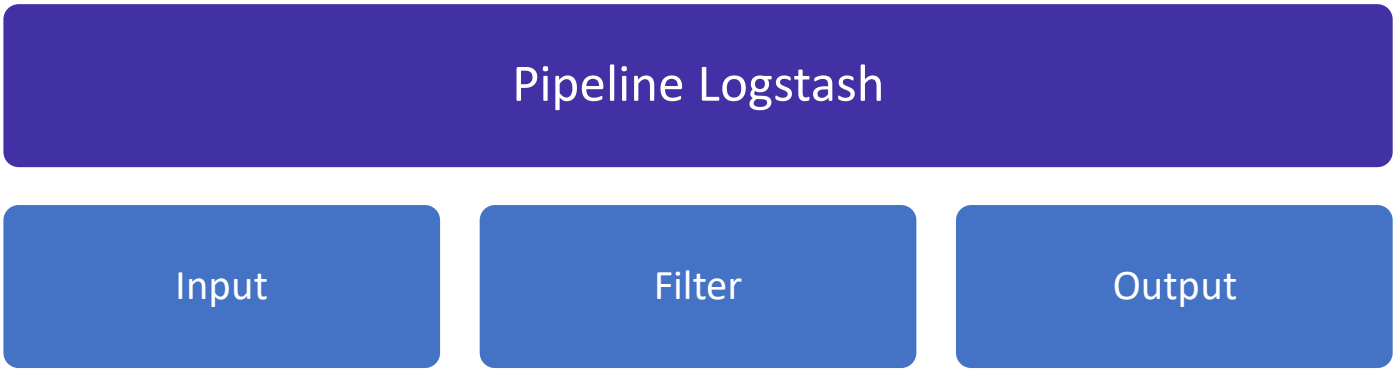
Comentários:

Mesmo sem saber muito sem o programa, sabemos uma coisa: seu *output*, ou seja, sua saída é ligada ao Elasticsearch. O programa, dos listados nas alternativas, que faz uma saída para ele é justamente o Logstash. (Gabarito: Letra D)

Pipeline Logstash

A *pipeline* de dados é a ferramenta responsável por levar os dados dos pontos de origens aos locais de destino - usualmente ao Elasticsearch, mas não limitado a ele. Numa *pipeline*, existem dois elementos obrigatórios: os **`inputs`** e os **`outputs`**. Ou seja, obrigatoriamente teremos dados de entrada e dados de saída. Alternativamente, é possível aplicar filtros **`filter`** no meio do caminho, para uma transformação dos dados.





Os **`inputs`** são componentes que definem as fontes de onde o Logstash irá ler os dados. Eles são responsáveis por coletar dado de uma diversidade de fontes, como *logs*, filas de mensageria, bancos de dados e outros, e passá-los para a *pipeline* do Logstash. Existem algumas formas de `input`, variando conforme o tipo de fonte e dado utilizado.

O tipo de *input* é definido a partir dos ***plugins***. Os principais são:

- File (arquivo) ⇒ lê arquivos de *log* e outros arquivos de texto. Usualmente associado à coleta de *logs*.
- Stdin ⇒ lê eventos de *inputs* padrão.
- HTTP ⇒ recebe dados através de requisições HTTP ou HTTPS.
- Syslog ⇒ recebe dados na porta 514 para *logs* do sistema, de acordo com o RFC3164.
- Redis ⇒ lê os dados do Redis, um SGBD de dados chave-valor.

Dentre vários outros. Você pode conferir a lista completa de *plugins* clicando aqui.

(Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre as ferramentas de *logging*, julgue o item abaixo.

Para a integração com diferentes fontes de dado no Logstash, usa-se os *plugins* de input. Um exemplo de *plugin* é o Syslog, que recebe *logs* de sistema na porta 443.

Comentários:

Muito cuidado! A questão está quase 100% correta, a não ser por um erro crucial - o Syslog ouve na porta 514, que é a porta do protocolo Syslog, não na 443, porta do HTTPS. (Gabarito: Errado)

Já os **`outputs`** são a fase final da *pipeline*. Após passar pelos filtros (que veremos em seguida, em uma seção específica, devido à sua importância), os dados podem ser encaminhados para diferentes repositórios. No Logstash, os dados são tratados como unidades discretas, chamadas de eventos. É uma questão terminológica, já que o Logstash lida com uma variedade de tipos de dados, alinhando o programa às arquiteturas baseadas em evento.

Assim como nos `inputs`, aqui são os plugins que definem a saída e fazem essa "integração". Veja os principais:

- elasticsearch: direciona a saída de dados para um *cluster* Elasticsearch.
- file: escreve eventos para arquivos no disco.
- http: envia eventos para
- tcp ou udp: envia os eventos para sockets TCP ou UDP
- csv: exporta os eventos como arquivos separados por vírgula
- email: envia um e-mail para um endereço específico, quando o *output* é recebido
- pipe: encaminha os eventos como *input* de outro programa
- syslog: exporta os eventos como logs de sistema, no protocolo syslog

Você também pode conferir a lista completa de *plugins* de `output` clicando aqui.

(Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre as ferramentas de *logging*, julgue o item abaixo.

Deseja-se encadear a saída de uma *pipeline* de dados no Logstash com o *input* stdin (standard) de outra *pipeline* Logstash. Nesse caso, deve-se usar o *plugin pipe*.

Comentários:

Perfeito! Caso ao *output* precise ser ligado a entradas padrões de outros programas, como a *stdin* do próprio Logstash, a saída a ser escolhida deve ser a *pipe*. (Gabarito: Correto)

Filtros

Os filtros fazem parte da *pipeline* do Logstash, mas, diferentemente dos `inputs` e `outputs`, eles são elementos opcionais. Eles são responsáveis por processamentos intermediários nos eventos do Logstash, usualmente aplicados de forma condicional, com base em determinadas características do evento.

Aqui também trabalhamos com *plugins*, que fazem diferentes trabalhos. Esse talvez seja o tópico mais importante sobre filtros, já que cada *plugin* atua independentemente e com certa relevância no processamento, podendo ser aplicada uma multiplicidade de filtros. Os principais filtros do Logstash são:

Filtro	Descrição
aggregate	Permite agrupar eventos logicamente relacionados e aplicar operações de agregação sobre eles. Útil para cálculos de média, soma, etc., em registros relacionados.
cidr	Verifica se um endereço IP está dentro de uma ou mais faixas de CIDR (Classless Inter-Domain Routing).
cipher	Criptografa ou descriptografa campos específicos usando algoritmos de cifra.
clone	Duplica eventos, permitindo modificações subsequentes em um clone enquanto o original permanece inalterado.
csv	Analisa linhas de texto formatadas em CSV, transformando-as em eventos e mapeando colunas para campos.
date	Analisa e converte cadeias de caracteres de data em objetos de data do Elasticsearch, ajustando para diferentes formatos e fusos horários.
drop	Remove eventos do pipeline de processamento. Útil para descartar dados irrelevantes ou indesejados.
dissect	Quebra eventos não estruturados em campos de *string* com base em delimitadores.
elasticsearch	Copia campos de eventos de *logs* anteriores no Elasticsearch para eventos correntes.
elastic_integration	Provê processamento adicional ao Logstash em dados provenientes de integrações com o Elasticsearch.
environment	Adiciona variáveis de ambiente aos eventos. Útil para incorporar informações do ambiente do sistema ao processamento.
fingerprint	Gera um identificador único para cada evento com base em um ou mais campos. Pode usar algoritmos de hash como MD5, SHA-1, etc.

geoip	Anexa informações geográficas (latitude, longitude, cidade, país, etc.) com base em endereços IP. Utiliza bancos de dados geográficos.
grok	Analisa textos não estruturados, aplicando expressões regulares para extrair campos específicos. Muito usado para logs.
metricize	Converte eventos em métricas, útil para transformar dados de log em séries temporais para monitoramento.
mutate	Realiza modificações gerais em campos de eventos, como renomear, remover, substituir, transformar em maiúsculas ou minúsculas, etc.
prune	Remove campos de eventos com base em critérios configuráveis, como tamanho do campo ou padrões de nome.
range	Permite a filtragem de eventos com base em intervalos de valores. Útil para selecionar eventos que se enquadram dentro de um determinado intervalo numérico.

(FGV/TJDFT/2022) O analista Pedro definiu no Logstash do TJDFT um novo pipeline de processamento de dados de nome SPipeline. A saída definida em SPipeline exige que os dados sejam estruturados. No entanto, a entrada definida em SPipeline consiste em um arquivo de texto arbitrário e não estruturado.

A fim de estruturar a entrada do SPipeline com o uso de expressões regulares, Pedro deve adicionar ao SPipeline o filtro do Logstash:

a) drop;
b) mutate;
c) grok;
d) clone;
e) aggregate.

Comentários:

Expressões regulares, ou regex (*regular expressions*), são padrões utilizados para identificar, buscar e manipular cadeias de texto. No Logstash, para adicionar essas expressões na fase de filtros, deve-se usar o filtro grok. Quanto aos demais filtros:

- drop = remove eventos da pipeline
- mutate = usado para modificar campos específicos
- clone = cria uma cópia dos dados originais
- aggregate = faz uma agregação dos dados

Portanto, a alternativa correta é a letra C. (Gabarito: Letra C)

QUESTÕES COMENTADAS

Prometheus

01. (FGV/TJDFT/2022) O analista Pedro definiu no Logstash do TJDFT um novo pipeline de processamento de dados de nome SPipeline. A saída definida em SPipeline exige que os dados sejam estruturados. No entanto, a entrada definida em SPipeline consiste em um arquivo de texto arbitrário e não estruturado.

A fim de estruturar a entrada do SPipeline com o uso de expressões regulares, Pedro deve adicionar ao SPipeline o filtro do Logstash:

a) drop;
b) mutate;
c) grok;
d) clone;
e) aggregate.

Comentários:

Os filtros são ferramentas adicionadas à pipeline do Logstash para remover ou transformar informações. No caso da questão, o filtro destinado a estruturar dados de entrada com o uso de expressões regulares (*regex*) é o grok - aliás, esse é um componente que se repete em vários programas, quando se necessita trabalhar com expressões regulares e estruturação.

Gabarito: Letra C

---

02. (FCC/TRT 22/2022) Considere o código abaixo presente em um arquivo de configuração criado por um analista, que especifica qual plug-in será utilizado em determinado contexto e as configurações deste plug-in.

```
input { stdin { } }
output {
    elasticsearch { hosts => ["localhost:9200"] }
}
```

As configurações mostram que, quando esse arquivo for lido, será gerada uma saída para o Elasticsearch. Trata-se de um arquivo de configuração do

a) Kibana.
b) Rancher.

c) Jenkins.
d) Logstash.
e) ElasticData.

Comentários:

O código especificado especifica o *input* (standard/padrão) e um *output*, para o Elasticsearch no *localhost:9200*. A ferramenta que tem um *output* para o Elasticsearch, e conta com esse arquivo de configuração, é o Logstash.

Gabarito: Letra C

---

03. (FGV/Câmara dos Deputados/2023 - Adaptada) No contexto das ferramentas e conceitos relacionados à análise de logs e ao monitoramento de desempenho, julgue o item abaixo.

Logstash é uma ferramenta de processamento de logs que pode coletar, transformar e encaminhar dados para vários destinos.

Comentários:

Perfeito! O Logstash coleta dados de diversas fontes, os transforma e os encaminha para diversos tipos de destinos diferentes.

Gabarito: Correto

---

04. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre as ferramentas de logging, julgue o item abaixo.

Para a integração com diferentes fontes de dado no Logstash, usa-se os plugins de input. Um exemplo de plugin é o Syslog, que recebe logs de sistema na porta 443.

Comentários:

Muito cuidado! A questão está quase 100% correta, a não ser por um erro crucial - o Syslog ouve na porta 514, que é a porta do protocolo Syslog, não na 443, porta do HTTPS.

Gabarito: Errado

---

05. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre as ferramentas de logging, julgue o item abaixo.

Deseja-se encadear a saída de uma pipeline de dados no Logstash com o input stdin (standard) de outra pipeline Logstash. Nesse caso, deve-se usar o plugin pipe.

Comentários:

A afirmativa está certa! Caso ao output precise ser ligado a entradas padrões de outros programas, como a stdin do próprio Logstash, a saída a ser escolhida deve ser a pipe.

Gabarito: Letra C

QUESTÕES COMENTADAS

Prometheus

01. (FGV/TJDFT/2022) O analista Pedro definiu no Logstash do TJDFT um novo pipeline de processamento de dados de nome SPipeline. A saída definida em SPipeline exige que os dados sejam estruturados. No entanto, a entrada definida em SPipeline consiste em um arquivo de texto arbitrário e não estruturado.

A fim de estruturar a entrada do SPipeline com o uso de expressões regulares, Pedro deve adicionar ao SPipeline o filtro do Logstash:

a) drop;
b) mutate;
c) grok;
d) clone;
e) aggregate.

02. (FCC/TRT 22/2022) Considere o código abaixo presente em um arquivo de configuração criado por um analista, que especifica qual plug-in será utilizado em determinado contexto e as configurações deste plug-in.

```
input { stdin { } }
output {
    elasticsearch { hosts => ["localhost:9200"] }
}
```

As configurações mostram que, quando esse arquivo for lido, será gerada uma saída para o Elasticsearch. Trata-se de um arquivo de configuração do

a) Kibana.
b) Rancher.
c) Jenkins.
d) Logstash.
e) ElasticData.

03. (FGV/Câmara dos Deputados/2023 - Adaptada) No contexto das ferramentas e conceitos relacionados à análise de logs e ao monitoramento de desempenho, julgue o item abaixo.

Logstash é uma ferramenta de processamento de logs que pode coletar, transformar e encaminhar dados para vários destinos.

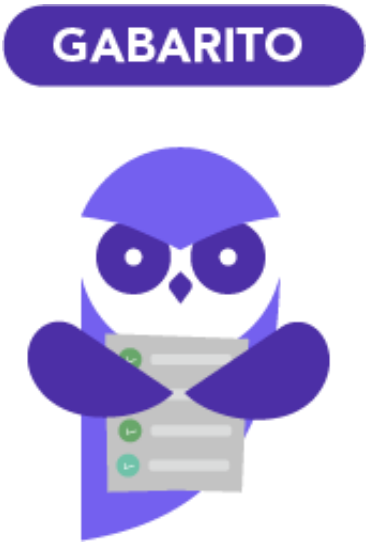
04. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre as ferramentas de logging, julgue o item abaixo.

Para a integração com diferentes fontes de dado no Logstash, usa-se os plugins de input. Um exemplo de plugin é o Syslog, que recebe logs de sistema na porta 443.

05. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre as ferramentas de logging, julgue o item abaixo.

Deseja-se encadear a saída de uma pipeline de dados no Logstash com o input stdin (standard) de outra pipeline Logstash. Nesse caso, deve-se usar o plugin pipe.

GABARITO



1. Letra C
2. Letra D
3. Correto
4. Errado
5. Correto

PROMETHEUS

Conceitos Gerais

O Prometheus é um sistema de monitoramento e alerta, open-source, originalmente construído para o SoundCloud, uma plataforma web de músicas. Ele age como uma espécie de banco de dados, coletando e armazenando métricas como dados de séries temporais. Um dado de série temporal é um dado "carimbado" com o tempo (*timestamp*), relativo ao momento em que foi gravado (com o carimbo).

Os dados no Prometheus são multidimensionais, armazenados em texto puro, no formato de pares de chave-valor, chamados de labels (etiquetas). Para determinada métrica, teremos essas *labels*, organizadas em chave-valor, seguidos de um *timestamp* baseado no padrão Unix, ou seja, a contagem de tempo é feita a partir de 1 de janeiro, 00:00:00, de 1970. Além disso, um valor relativo à métrica é adicionado antes do *timestamp*, destinado a apontar a quantidade de ocorrências da métrica armazenada. Veja um exemplo de dados de requisições HTTP:

```
http_requests_total{
    method="GET",
    handler="/home",
    status="200"} 15432  1685205600000
```

```
http_requests_total{
    method="GET",
    handler="/home",
    status="404"} 56  1685205720000
```

```
http_requests_total{
    method="DELETE",
    handler="/api/v1/items",
    status="200"} 18  1685205840000
```

Acima, tempos os seguintes valores:

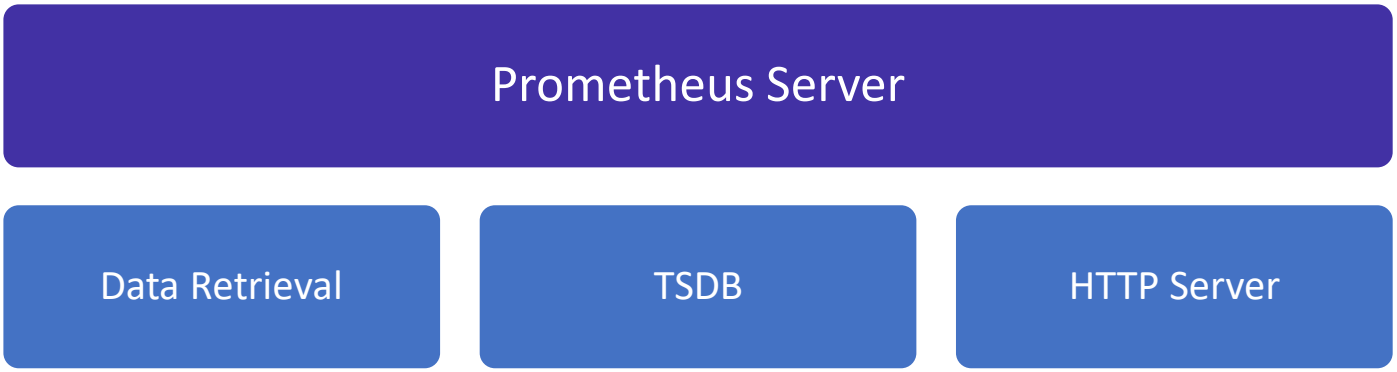
- Métrica em análise: é o elemento que encadeia e define os valores. No exemplo, o `http_requests_total`.
- Labels: são os pares de chave e valor, como `method="GET"`
- Valor: mede a quantidade de ocorrências da métrica. É o primeiro valor que aparece após as chaves.

- Timestamp: mede o momento de carimbo da métrica.

Arquitetura

Apesar de ter lhe exposto o Prometheus como uma espécie de banco de dados de séries temporais, ele é muito mais que isso. Uma pluralidade de componentes formam o Prometheus, coordenados por um componente central - o Prometheus Server. Esse servidor é dividido em três partes distintas:

- Recuperador de Dados/Data Retrieval Worker: também chamado de *Data Scraping*, é o responsável por coletar as métricas dos alvos. Para isso, utiliza-se de requisições HTTP para alvos específicos e coleta as métricas. Quando coletadas, elas são armazenadas como dados de séries temporais.
- Banco de Dados de Séries Temporais/Time Series DataBase (TSDB): o servidor responsável por armazenar os dados em séries temporais.
- Servidor HTTP: existe para permitir a recuperação dos dados armazenados dentro do banco de dados. São feitas consultas a partir da linguagem nativa do Prometheus, o PromQL, para o servidor HTTP, que, por sua vez, consulta o TSDB e recupera os dados.



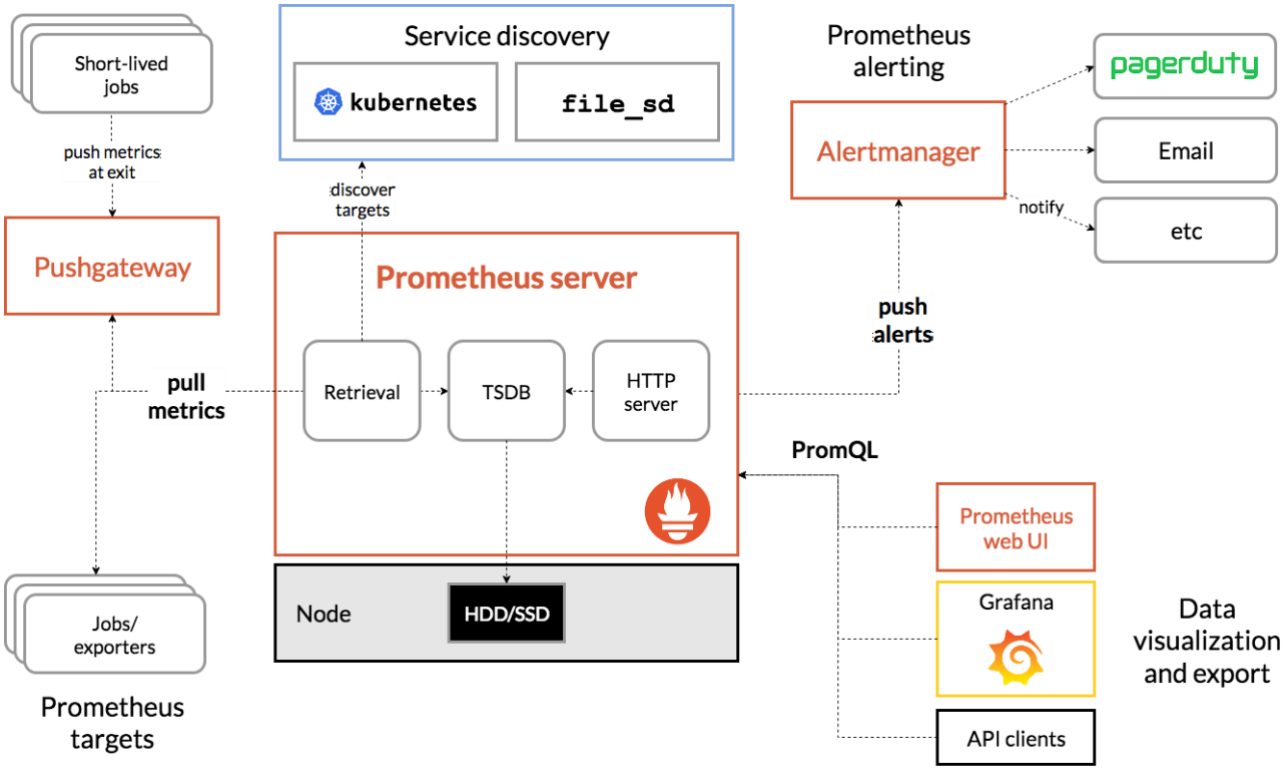
Além disso, diversos outros componentes opcionais compreendem a solução completa do Prometheus. Os principais, que você deve saber, são:

- Exportadores: são componentes intermediários, entre os sistemas alvo, para coletas de dados, e o Prometheus. Eles extraem as métricas, as transformam em formatos de dados suportados pelo Prometheus, e expõem um *endpoint* HTTP para a recuperação dos dados. Existem alguns exportadores nativos:
    - Node Exporters (Linux)
    - Windows
    - MySQL
    - Apacha
    - HAProxy
- Pushgateway: o Prometheus é um modelo baseado em *pull*, isso é, ele ativamente recupera os dados de outros sistemas, por iniciativa própria. É possível que as aplicações monitoradas

enviem as informações diretamente ao Prometheus, usando um modelo baseado em *push*, a partir do Pushgateway.

- AlertManager: é um componente responsável por gerenciar os alertas criados pelo Prometheus. Ele permite o roteamento, agrupamento e processamento dos alertas, permitindo o envio para endereços de e-mail, plataformas de *chat*, entre outros. O Prometheus, por si só, apenas gera os alertas - ele não os envia. O AlertManager preenche essa segunda função.
- ServiceDiscovery: é um componente essencial, que permite ao Prometheus descobrir automaticamente os alvos de monitoramento, sem necessitar de qualquer configuração manual.
- Ferramentas visuais: São ferramentas que utilizam o PromQL para acessar os dados e criar visualizações. A principal ferramenta com esse fim, no contexto do Prometheus, é o Grafana. Além disso, nativamente o Prometheus fornece o Prometheus Web UI também com esse fim.

Veja um *overview* dos componentes:



(Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Prometheus, julgue o item abaixo.

O Prometheus atua somente num modelo *pull-based*, isso é, os dados só chegam a ele a partir de suas próprias requisições, sem a possibilidade de um encaminhamento ativo de métricas.

Comentários:

Nativamente sim, o Prometheus funciona num modelo *pull-based*. Porém, com o uso do Pushgateway, é possível adotar uma abordagem *push-based*, recebendo ativamente os dados de métricas dos alvos de análise. Portanto, incorreta a questão. (Gabarito: Errado)

Tarefas e Instâncias

Para coordenar a coleta de dados, ou *scrape*, o Prometheus utiliza-se de dois conceitos: jobs, ou tarefas, e instances, ou instâncias. São eles que coordenam todo o trabalho de recuperar dados e métricas dos sistemas alvo para serem armazenados no TSDB.

Uma instância representa um único *endpoint* que o Prometheus passa coletando os dados. O termo *scrape*, em inglês, significar "raspar", e representa muito bem o processo pelo qual o Prometheus passa. Os dados vão se acumulando em um *endpoint* e em um intervalo definido de tempo, o Prometheus faz um *pull request*, uma requisição para recuperar esses dados e armazenar no seu TSDB.

Já uma tarefa, ou um job, representa um agrupamento lógico de instâncias, usualmente relacionado a um mesmo tipo de métrica e atividade. Tipicamente, uma tarefa corresponde a aplicações ou serviços específicos que se deseja monitorar. Por exemplo, podemos analisar os *endpoints* de uma aplicação em conjunto através de uma tarefa.

A configuração de tarefas e instâncias é feita num arquivo YAML denominado **`prometheus.yml`**. A configuração fica numa seção chamada de **`scrape_configs`**. Veja um exemplo:

YAML

```yaml
scrape_configs:
  - job_name: 'web_server'
    static_configs:
      - targets: ['localhost:8000', 'localhost:8001']

  - job_name: 'database'
    static_configs:
      - targets: ['localhost:5432']
```

Acima temos a criação de dois *jobs*. O primeiro, de nome 'web server', atua ouvindo nas instâncias localhost:8000 e localhost:8001, fazendo a coleta nesses *endpoints*. Já o segundo, chamado de 'database', atua na instância presente na porta localhost:5432.

O Prometheus adiciona automaticamente duas etiquetas (*labels*) para cada métrica coletada, para identificar a origem. São justamente as identificações do trabalho e da instância específica que está sendo utilizada. Veja:

```
http_requests_total{
    job="web_server",
    instance="localhost:8000"} 1234
```

Métricas

No contexto do Prometheus, métricas são medidas quantitativas que proveem informações sobre o estado, comportamento e performance de um sistema. Essas medidas cobrem um grande variedade de dados, incluindo informações de utilização de recursos, erros, requisições HTTP, latência, entre outros. O Prometheus oferece suporte para 4 tipos de métricas diferentes:

- Counter (contagem)
- Gauge (medição)
- Histogram (histograma)
- Summary (sumário)

(Inédita/Prof. Felipe Mathias) Assinale a alternativa abaixo que não corresponde a uma das métricas disponíveis no programa Prometheus.

a) Counter
b) Histogram
c) Gauge
d) Average
e) Summary

Comentários:

O Prometheus conta com 4 métricas: Cunter, Gauge, Histogram e Summary. A única que não está nessa lista é a apontada na letra D - Average. (Gabarito: Letra D)

Vamos ver cada métrica!

Counter

Um contador, ou ***counter***, é umas métrica cumulativa que representa um único contador monótono crescente, cujo valor só pode aumentar ou ser reiniciado para zero em determinado

reinício. É uma ótima métrica para representar números de solicitações atendidas, de tarefas, de erros, ou outros elementos que exijam uma contagem.

Gauge

O *gauge* é uma métrica que representa um valor numérico único, que pode, arbitrariamente, aumentar e diminuir. Eles são usados tipicamente para valores de medidas, como temperatura ou uso corrente de memória, ou para contagens que necessitam de uma abordagem que permitam o aumento ou a diminuição de valores, como números de requisições concorrentes.

Histogram

Um histograma é um tipo de métrica de amostragem, destinada a observar e registrar a distribuição de eventos em um conjunto de *buckets*, uma forma de agrupamento desses eventos. Eles são úteis para medições de frequência e magnitude de determinados eventos, como durações de requisições, tamanho de respostas e outros dados numéricos.

Aqui cada *bucket* corresponderá a um agrupamento de eventos, podendo ser direcionado a tipo de eventos, aplicações, *endpoints*, entre outros. Dessa forma, são criadas amostras que permitem a comparação entre valores, além de formas de agregação, como a soma.

Summary

Os *summaries*, ou sumários, também são métricas de amostragem, destinados a observar e registrar a distribuição de eventos, principalmente no que tange a sua duração ou tamanho, de forma muito similar aos histogramas. Porém, os sumários fornecem outras funcionalidades nas agregações, particularmente em volta da precisão das estimativas - o que reflete em cálculos, como os cálculos de quantis de distribuição.

(ProPGe UFSCAR/UFSCAR/2019) As bibliotecas cliente do Prometheus oferecem diversos tipos de métrica. Segundo as recomendações do Prometheus, qual o tipo mais adequado de métrica para medir a quantidade de exceções que ocorrem em determinado trecho de um código instrumentado?

a) Info.
b) Counter.
c) Gauge.
d) Summary.
e) Histogram.

Comentários:

A medida que estamos analisando é a quantidade de exceções que ocorrem em um código implementado. Veja que é uma métrica numérica, e que somente irá aumentar - já que não temos como diminuir a quantidade de erros ocorridos em uma execução do código. Portanto, a métrica ideal nesse caso é a *counter*. (Gabarito: Letra B)

PromQL

A PromQL, ou Prometheus Query Language, é uma linguagem funcional de consulta que permite o usuário selecionar e agregar dados de séries temporais em tempo real. O resultado das expressões podem ser demonstrados em gráficos, vistos como dados tabulares ou consumidos por sistemas externos a partir de APIs HTTP.

No PromQL, podemos lidar com 4 tipos de dados diferentes:

- Vetor de Instante (Instant Vector): um conjunto de séries temporais contendo um único exemplo para cada série, todos compartilhando um único *timestamp*.
- Vetor de Intervalo (Range Vector): um conjunto de séries temporais contendo uma variedade de pontos de dados ao longo do tempo, para cada série - ou seja, com múltiplos *timestamps*.
- Escalar (Scalar): um número simples, de ponto flutuante.
- String: uma string de caracteres quaisquer.

Ao elaborar as consultas, é possível o uso de modificadores. Os modificadores são como as cláusulas do SQL, servindo para filtrar e manipular os dados retornados pelas consultas. Os principais modificadores são:

- Comparador de etiquetas (Label Matcher): filtra os dados com base no valor do *label*.
- Offset: cria lapsos temporais ou de espaço nos dados, como retornar dados 5 minutos no passado.
- Operador de Agregações (Aggregation Operators): faz agregações das séries temporais.

(ProGPe UFSCAR/UFSCAR/2019) O monitoramento de aplicações e servidores é uma das tarefas mais relevantes do cotidiano do desenvolvedor de software. O Prometheus é um sistema de monitoramento para serviços e aplicações. Ele trabalha coletando as métricas de seus alvos em determinados intervalos, avaliando expressões de regras, exibindo os resultados e também podendo gerar alertas, caso alguma condição for observada como verdadeira. Assinale a alternativa que mostra qual é a linguagem utilizada para consultas pelo Prometheus:

a) MySQL.
b) SQL Server.
c) PromQL.
d) MAQL.
e) PL/SQL.

Comentários:

Questão tranquila - a linguagem de consultas no Prometheus é o PromQL. (Gabarito: Letra C)

QUESTÕES COMENTADAS

Prometheus

01. (FGV/TRT 13/2022) Sobre a ferramenta de monitoramento Prometheus, analise as afirmativas a seguir.

I. Armazena fundamentalmente todos os dados como séries temporais.
II. Possui uma linguagem para consulta das métricas armazenadas denominada PromQL.
III. Coleta métricas através do protocolo HTTP.

Está correto o que se afirma em

a) I, apenas.
b) II, apenas.
c) III, apenas.
d) I e II, apenas.
e) I, II e III.

Comentários:

Vamos analisar cada item.

I. Certo. Os dados do Prometheus são marcados com um carimbo de tempo, e armazenados em pares de chave-valor orientados a esse valor de tempo, o que torna o Prometheus um banco de dados de séries temporais.
II. Certo. Prometheus Query Language, ou PromQL, é a linguagem nativa para consultas no Prometheus.
III. Certo. O Prometheus usa uma API RESTful para suas coletas - que é manuseada a partir de requisições HTTP.

Portanto, todos os itens estão corretos.

Gabarito: Letra E

---

02. (ProGPe UFSCAR/UFSCAR/2019) O monitoramento de aplicações e servidores é uma das tarefas mais relevantes do cotidiano do desenvolvedor de software. O Prometheus é um sistema de monitoramento para serviços e aplicações. Ele trabalha coletando as métricas de seus alvos em determinados intervalos, avaliando expressões de regras, exibindo os resultados e também podendo gerar alertas, caso alguma condição for observada como verdadeira. Assinale a alternativa que mostra qual é a linguagem utilizada para consultas pelo Prometheus:

a) MySQL.
b) SQL Server.
c) PromQL.
d) MAQL.
e) PL/SQL.

Comentários:

Questão bem tranquila - a linguagem nativa para consultas no Prometheus é o Prometheus Query Language, ou PromQL.

Gabarito: Letra C

---

03. (ProGPe UFSCAR/UFSCAR/2019) Em relação ao Prometheus, analise as seguintes afirmações como verdadeira (V) ou falsa (F) e assinale a alternativa correta:

I. O Prometheus tem múltiplos modos de visualização de dados, incluindo uma integração com o Grafana.
II. O Prometheus não armazena os dados como séries temporais.
III. O Prometheus provê uma linguagem de consulta funcional chamada PromQL (Prometheus Query Language).

a) I(V), II(V), III(F).
b) I(V), II(V), III(V).
c) I(V), II(F), III(F).
d) I(V), II(F), III(V).
e) I(F), II(F), III(F).

Comentários:

Vamos analisar cada item.

I. Verdadeiro. O Prometheus é conhecido por uma ampla integração com diferentes repositórios de dados - incluindo o Grafana.
II. Falso. O Prometheus armazena justamente os dados em séries temporais.
III. Verdadeiro. De fato, a PromQL é a linguagem de consulta nativa do programa.

Portanto, temos V-F-V.

Gabarito: Letra D

---

04. (ProGPe UFSCAR/UFSCAR/2019) O Prometheus é um conjunto de ferramentas de código aberto para monitoramento de aplicações e alerta de sistemas. Tendo em vista este contexto, analise as afirmativas I, II e III e assinale a alternativa correta.

I. O Prometheus trabalha com um modelo de dados multidimensional com dados de série temporal.
II. O Prometheus é adequado para gravar série temporal puramente numérica. Ele se ajusta bem ao monitoramento centrado na máquina, porém não é adequado para o monitoramento de arquiteturas dinâmicas orientadas a serviços.
III. Cada servidor Prometheus é independente, não dependendo do armazenamento de rede ou de outros serviços remotos.

a) I - verdadeira, II - falsa, III - verdadeira.
b) I - falsa, II - verdadeira, III - verdadeira.
c) I - verdadeira, II - verdadeira, III - verdadeira.
d) I - verdadeira, II - falsa, III - falsa.
e) I - falsa, II - falsa, III - falsa.

Comentários:

Vamos analisar cada item.

I. Verdadeiro. O modelo de dados do Prometheus é multidimensional, e eles são armazenados como séries temporais.
II. Falso. O Prometheus lida somente com dados numéricos, realmente. Porém, ele é adequado sim para arquiteturas orientadas a serviço, principalmente arquiteturas de microsserviços.
III. Verdadeiro. Os servidores do Prometheus são, de fato, independentes.

Portanto, ficamos com V-F-V.

Gabarito: Letra A

---

05. (ProGPe UFSCAR/UFSCAR/2019) As bibliotecas cliente do Prometheus oferecem diversos tipos de métrica. Segundo as recomendações do Prometheus, qual o tipo mais adequado de métrica para medir a quantidade de exceções que ocorrem em determinado trecho de um código instrumentado?

a) Info.
b) Counter.
c) Gauge.
d) Summary.
e) Histogram.

Comentários:

Estamos analisando uma métrica que somente pode crescer - afinal, após ocorrida determinada exceção, esse número não irá mais diminuir. Nesse sentido, a métrica de análise ideal é a counter.

Gabarito: Letra B

---

06. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Prometheus, julgue o item abaixo.

O Prometheus atua somente num modelo *pull-based*, isso é, os dados só chegam a ele a partir de suas próprias requisições, sem a possibilidade de um encaminhamento ativo de métricas.

Comentários:

Nativamente sim, o Prometheus funciona num modelo *pull-based*. Porém, com o uso do Pushgateway, é possível adotar uma abordagem *push-based*, recebendo ativamente os dados de métricas dos alvos de análise. Portanto, incorreta a questão.

Gabarito: Errado

---

07. (Inédita/Prof. Felipe Mathias) Assinale a alternativa abaixo que não corresponde a uma das métricas disponíveis no programa Prometheus.

a) Counter
b) Histogram
c) Gauge
d) Average
e) Summary

Comentários:

O Prometheus conta com 4 métricas: Cunter, Gauge, Histogram e Summary. A única que não está nessa lista é a apontada na letra D - Average.

Gabarito: Letra D

LISTA DE QUESTÕES

Prometheus

01. (FGV/TRT 13/2022) Sobre a ferramenta de monitoramento Prometheus, analise as afirmativas a seguir.

I. Armazena fundamentalmente todos os dados como séries temporais.
II. Possui uma linguagem para consulta das métricas armazenadas denominada PromQL.
III. Coleta métricas através do protocolo HTTP.

Está correto o que se afirma em

a) I, apenas.
b) II, apenas.
c) III, apenas.
d) I e II, apenas.
e) I, II e III.

02. (ProGPe UFSCAR/UFSCAR/2019) O monitoramento de aplicações e servidores é uma das tarefas mais relevantes do cotidiano do desenvolvedor de software. O Prometheus é um sistema de monitoramento para serviços e aplicações. Ele trabalha coletando as métricas de seus alvos em determinados intervalos, avaliando expressões de regras, exibindo os resultados e também podendo gerar alertas, caso alguma condição for observada como verdadeira. Assinale a alternativa que mostra qual é a linguagem utilizada para consultas pelo Prometheus:

a) MySQL.
b) SQL Server.
c) PromQL.
d) MAQL.
e) PL/SQL.

03. (ProGPe UFSCAR/UFSCAR/2019) Em relação ao Prometheus, analise as seguintes afirmações como verdadeira (V) ou falsa (F) e assinale a alternativa correta:

I. O Prometheus tem múltiplos modos de visualização de dados, incluindo uma integração com o Grafana.
II. O Prometheus não armazena os dados como séries temporais.
III. O Prometheus provê uma linguagem de consulta funcional chamada PromQL (Prometheus Query Language).

a) I(V), II(V), III(F).
b) I(V), II(V), III(V).
c) I(V), II(F), III(F).
d) I(V), II(F), III(V).
e) I(F), II(F), III(F).

04. (ProGPe UFSCAR/UFSCAR/2019) O Prometheus é um conjunto de ferramentas de código aberto para monitoramento de aplicações e alerta de sistemas. Tendo em vista este contexto, analise as afirmativas I, II e III e assinale a alternativa correta.

I. O Prometheus trabalha com um modelo de dados multidimensional com dados de série temporal.
II. O Prometheus é adequado para gravar série temporal puramente numérica. Ele se ajusta bem ao monitoramento centrado na máquina, porém não é adequado para o monitoramento de arquiteturas dinâmicas orientadas a serviços.
III. Cada servidor Prometheus é independente, não dependendo do armazenamento de rede ou de outros serviços remotos.

a) I - verdadeira, II - falsa, III - verdadeira.
b) I - falsa, II - verdadeira, III - verdadeira.
c) I - verdadeira, II - verdadeira, III - verdadeira.
d) I - verdadeira, II - falsa, III - falsa.
e) I - falsa, II - falsa, III - falsa.

05. (ProGPe UFSCAR/UFSCAR/2019) As bibliotecas cliente do Prometheus oferecem diversos tipos de métrica. Segundo as recomendações do Prometheus, qual o tipo mais adequado de métrica para medir a quantidade de exceções que ocorrem em determinado trecho de um código instrumentado?

a) Info.
b) Counter.
c) Gauge.
d) Summary.
e) Histogram.

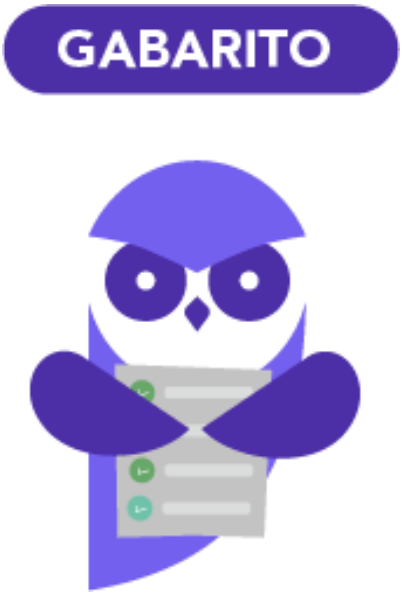
06. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Prometheus, julgue o item abaixo.

O Prometheus atua somente num modelo *pull-based*, isso é, os dados só chegam a ele a partir de suas próprias requisições, sem a possibilidade de um encaminhamento ativo de métricas.

07. (Inédita/Prof. Felipe Mathias) Assinale a alternativa abaixo que não corresponde a uma das métricas disponíveis no programa Prometheus.

a) Counter
b) Histogram
c) Gauge
d) Average
e) Summary

GABARITO



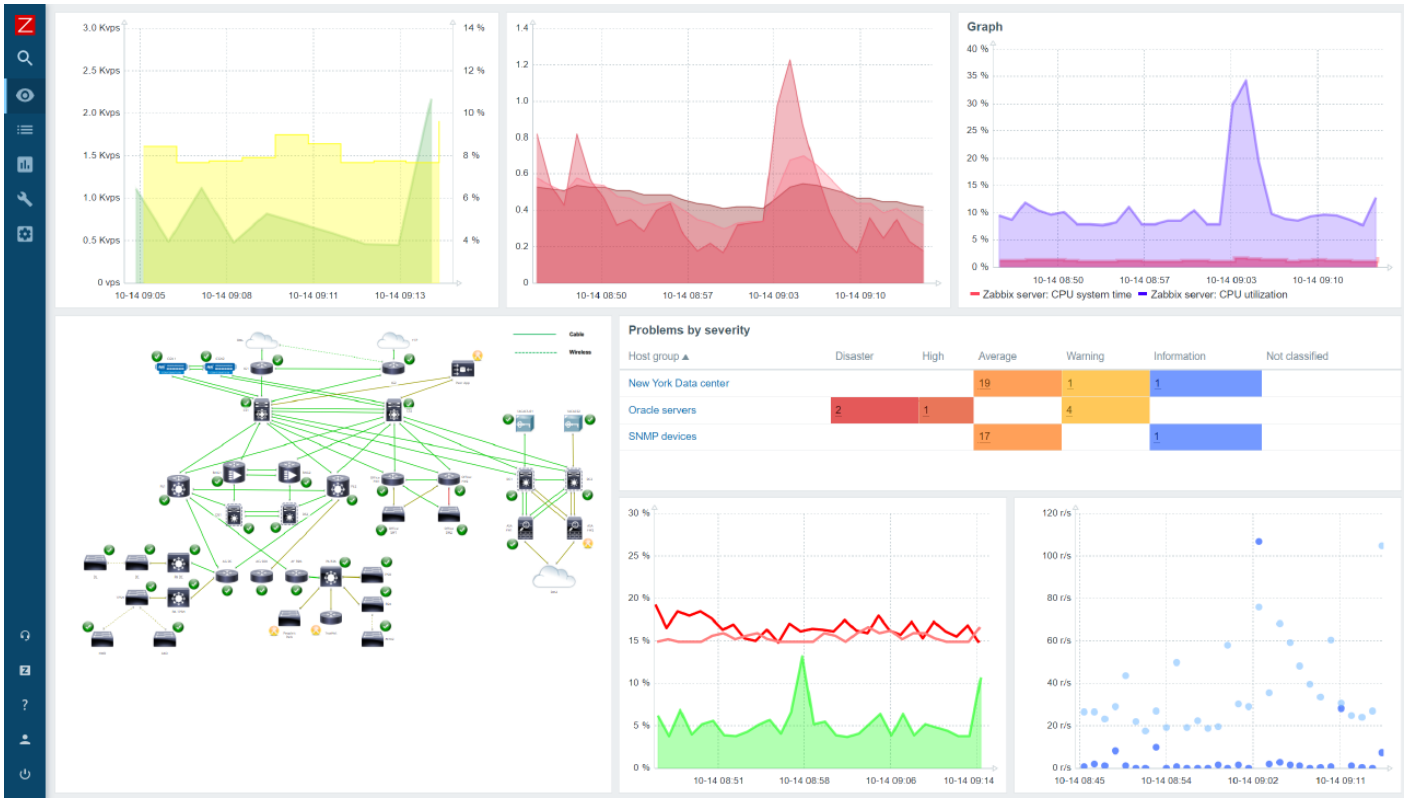
1. Letra E
2. Letra C
3. Letra D
4. Letra A
5. Letra B
6. Errado
7. Letra D

ZABBIX

Conceitos Gerais

O Zabbix é uma ferramenta de código aberto, destinada a monitoração distribuída. Seu objetivo é monitorar parâmetros de redes, servidores, máquinas virtuais, bancos de dados, aplicativos *web*, entre outros. Ele implementa mecanismos de acompanhamento e notificação, permitindo configuração nativa de alertas baseados em e-mail para praticamente qualquer evento, garantindo uma resposta rápida a diversos incidentes.

O Zabbix é uma ferramenta "completa". Outras aplicações de monitoramento, muitas vezes, necessitam de programas externos ou uma pilha de programas para fornecer todas as funcionalidades exigidas, como extração de *logs*, armazenamento e criação de visualizações. O Zabbix traz uma abordagem completa, desde a captura das informações necessárias até a geração completa de visualizações para acompanhamento.



A coleta de informações do Zabbix se dá de duas diferentes formas: *pooling* e *trapping*. No *pooling*, o Zabbix consulta ativamente os nós da rede e serviços, em intervalos especificados, coletando as informações - ou seja, a iniciativa de coleta das informações parte do Zabbix em direção aos demais componentes analisados.

Essa coleta é feita através de requisições feitas pelo servidor Zabbix ou por um *proxy*, que encaminha essa requisição ao nó de onde deseja-se obter informações. Essa comunicação usa uma diversidade de protocolos, como SNMP, HTTP, TCP e ICMP.

Já o *trapping* é um método onde os sistemas monitorados enviam dados e alertas (chamados de *trap message*) para o servidor Zabbix sempre que um evento ocorre. Ou seja, a iniciativa de envio da informação aqui parte dos nós do sistema, para o Zabbix. Aqui usa-se protocolos mais orientados a eventos, como as *traps* do SNMP, do próprio Zabbix, e outros. Dessa forma conseguimos um monitoramento quase em tempo real.

QUESTÃO DE PROVA

(CEBRASPE/TRT 8/2022) Assinale a opção que apresenta ferramenta utilizada para monitorar a infraestrutura de TI, como redes, servidores, máquinas virtuais e serviços em nuvem.

a) Zabbix
b) Kibana
c) Grafana
d) Elasticsearch
e) Prometheus

Comentários:

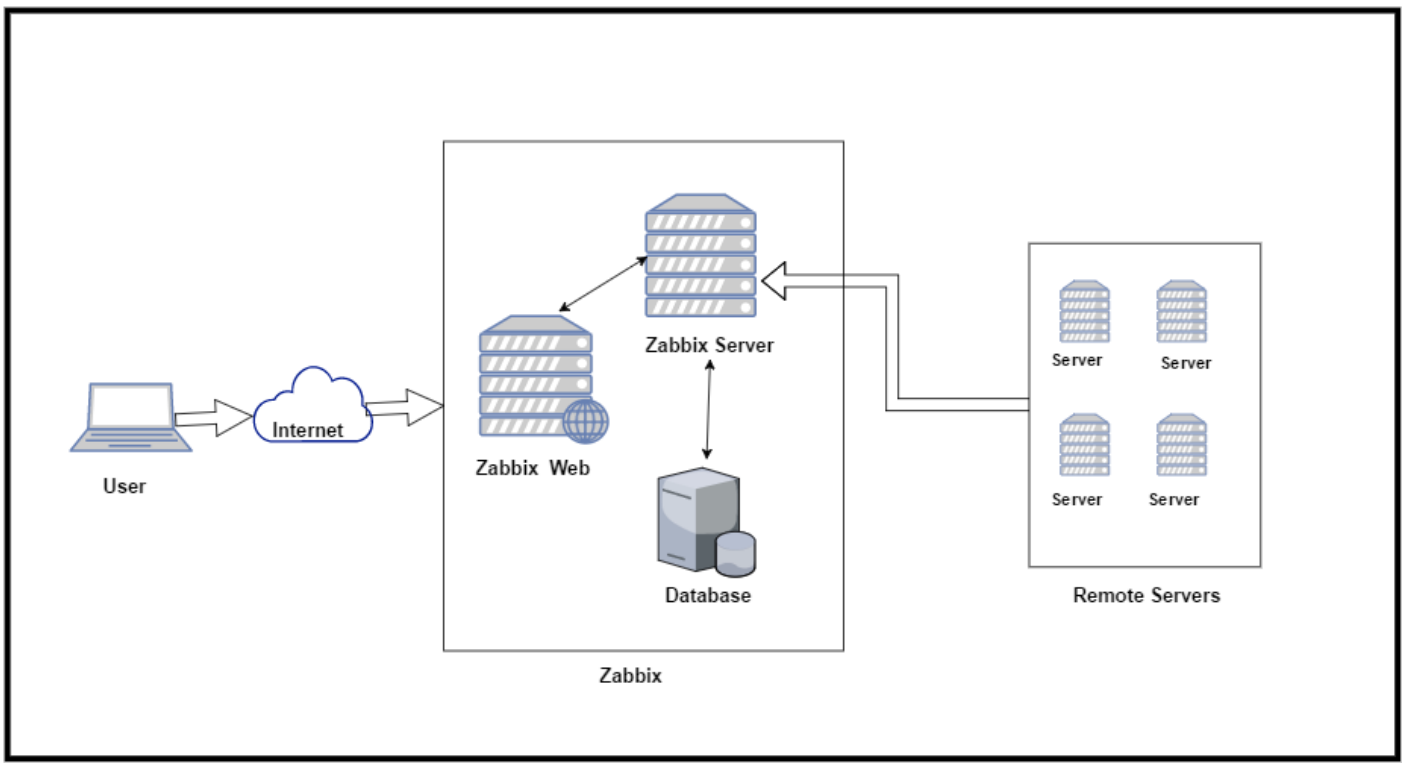
Vamos analisar cada alternativa.

a) Certo. Zabbix é uma ferramenta de monitoramento de infraestrutura, de acordo com o requisitado pelo enunciado.
b) Errado. Kibana é uma ferramenta de criação de visualizações.
c) Errado. Assim como o Kibana, o Grafana também é responsável por criar visualizações.
d) Errado. Elasticsearch é um sistema de armazenamento, indexação e pesquisa de dados.
e) Errado. Prometheus é um sistema de armazenamento e processamento de dados de séries temporais.

A alternativa correta é, portanto, a letra A. (Gabarito: Letra A)

Arquitetura

Antes de explicarmos a estrutura do Zabbix, que é onde muitas bancas baseiam suas questões, veja um esquema que nos traz a visão geral do programa.



OBS: Para esta aula, estamos usando a versão 6 do Zabbix.

Uma solução básica de monitoramento no Zabbix conta com três componentes centrais: o Zabbix Server, a Interface Web e o Banco de Dados (SGDB). Além disso, como mecanismos de coleta de informações, temos os agentes e os proxies. Vamos ver todos esses componentes agora.

Componentes Centrais

No Zabbix, temos 3 componentes centrais que atuam em conjunto: o Zabbix Server, o Zabbix Web e o banco de dados.

Zabbix Server

O Zabbix Server é o ponto central, o principal componente de toda a arquitetura do sistema de monitoramento. Ele é responsável por gerenciar a coleta e recebimento dos dados, calcular os estados para disparar gatilhos e enviar notificações aos usuários. Ele recebe todos os dados enviados pelos outros componentes - como os *agentes* e os *proxies*.

Toda informação de configuração da monitoração é armazenada no Banco de Dados. O Zabbix Server, em intervalos frequentes, usualmente a cada minuto, irá buscar no banco de dados as listas de itens que devem ser monitorados, e inclui-los nas rotinas.

O Server roda em segundo plano, como um *daemon*. Abaixo, deixo uma forma de iniciá-lo utilizando uma termina de CLI.

```Shell
cd sbin
./zabbix_server <parâmetros>
```

Existem diversos parâmetros que podem ser passados juntos com a inicialização do servidor. Dois deles você precisa saber:

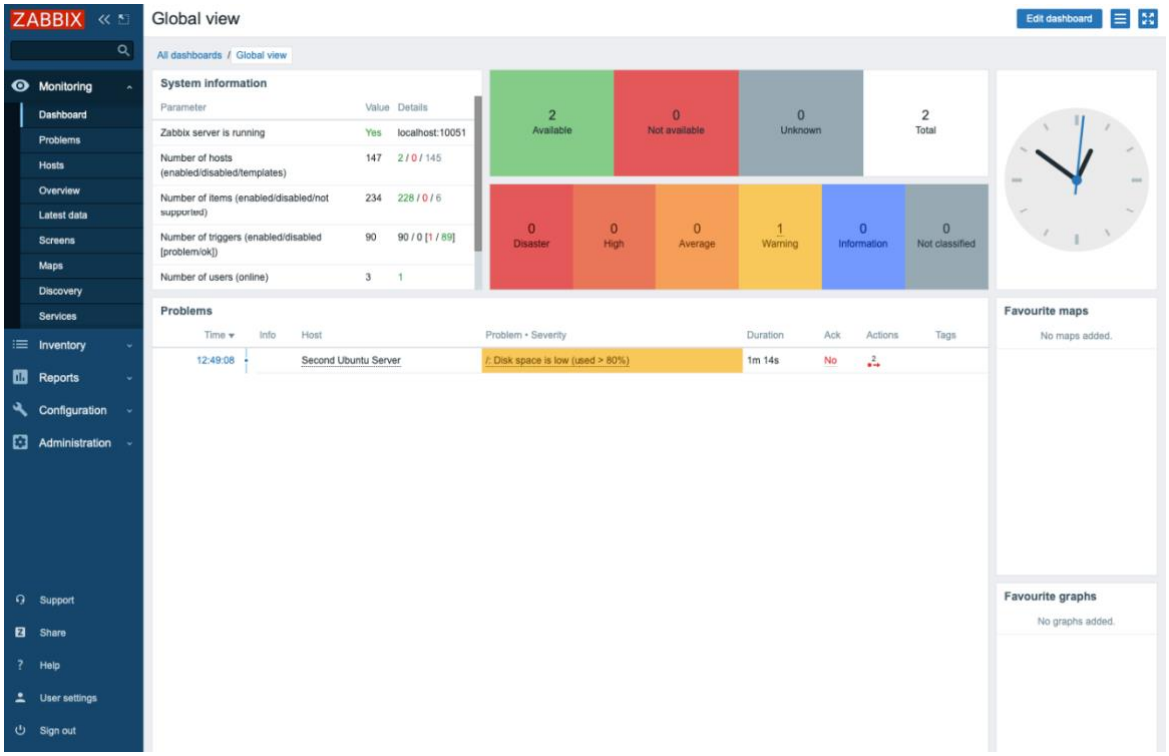
- `-c` ou `--config <arquivo>` ⇒ passa o caminho absoluto para o arquivo de configuração. Por padrão, ele fica em /etc/zabbix/zabbix_server.conf
- `-R` -ou `-runtime-control <opção>` ⇒ executa funções administrativas

Alternativamente e de forma opcional, pode-se adotar a abordagem de alta disponibilidade (*High Availability - HA*). Quando adotada, teremos múltiplas instâncias, ou nós do Zabbix Server. Cada nó deve ter uma configuração separada, com arquivos de configuração, script, controle de acesso, criptografia e outros. Apesar da configuração individual, todos os nós devem acessar um mesmo banco de dados, para garantir uma maior integridade e leituras mais consistentes.

Zabbix Web

O Zabbix Web é a interface de usuário, o *front-end* do Zabbix, baseado em aplicações web, ou seja, acessado através de um navegador de internet. Ele cria um portal, com interfaces amigáveis aos usuários, para configuração e gerenciamento dos diversos componentes do ecossistema do Zabbix, além de fornecer uma plataforma para monitoramento dos dados.

A interface web parece-se com isso:



O *frontend* do Zabbix é desenvolvido em PHP. Por esse motivo, servidores web para hospedá-lo precisam suportar essa linguagem, o que nos leva a dois servidores principais para integração com o Zabbix - o Nginx e o Apache. Para acessar a interface, após instalada, usamos:

- Para o Apache: http://<ip ou nome do server>/zabbix
- Para o Nginx: http://<ip ou nome do server>

Note que, nessa abordagem, as comunicações são feitas a partir de requisições HTTP. Também é possível implementar o protocolo seguro SSL/TSL para essas comunicações, adicionando uma camada de segurança.

Ao configurar a interface web, deve-se conectar ao mesmo banco de dados utilizado pelo Zabbix Server, onde estão todas as configurações e endereços IP necessários para o correto funcionamento da interface.

Banco de Dados

O banco de dados do Zabbix tem dois propósitos distintos - armazenar os dados de métricas e *logs*, e armazenar dados de configuração. Como o Zabbix funciona num esquema de computação distribuída, é o banco de dados que forma um elo que une todos os componentes.

A implementação do banco de dados é feita na hora da instalação do Zabbix Server ou do Zabbix Proxy. Para sua implementação, utiliza-se um SGBD externo para manusear o banco de dados. O

Zabbix oferece suporte a alguns SGBDs diferentes: MySQL, MariaDB, PostgreSQL, TimescaleDB, Oracle e SQLite (este suportado apenas no Zabbix Proxy).

(FGV/TRT 16/2022) A ferramenta Zabbix 6 é amplamente usada para monitoramento de redes; sua instalação e configuração requer pacotes de terceiros.

Com relação aos seus requisitos de instalação, analise os itens a seguir.

I. O Zabbix é construído em torno de servidores web, servidores de banco de dados e da linguagem de script Python.
II. Os servidores de bancos de dados suportados, por padrão, são MySQL, PostgreSQL ou MS SQL Server.
III. O frontend do Zabbix é suportado, por padrão, por servidores Web Apache.

Está correto o que se afirma em

a) I, II e III.
b) I apenas.
c) II apenas.
d) III apenas.
e) II e III, apenas.

Comentários:

Questão muito interessante, sobre os três componentes *core* do Zabbix. Vamos analisar cada item.

I. Errado. De fato, temos servidores web, tanto para o Zabbix Server quanto parao Zabbix Web e servidores para banco de dados. Entretanto, o Zabbix utiliza a linguagem PHP para sua estruturação, não o Python.
II. Errado. Dentre os SGBD apresentados, não há suporte, por parte do Zabbix, ao MS SQL Server. Somente ao MySQL e ao PostgreSQL.
III. Certo. O Zabbix funciona com dois servidores que suportam PHP - Nginx e Apache.

Portanto, correto somente o item III. (Gabarito: Letra D)

Outro ponto que pode aparecer na prova são as portas utilizadas. Existem duas portas principais - 10050 para o Zabbix Server e checagens passivas, e 10051 para os agentes ativos Zabbix.

Agentes

Os Agentes do Zabbix são implementados nos alvos de monitoramento, para monitorar recursos e aplicações locais, como uso de memória, estatísticas de processador, entre outros. Com isso, podemos afirmar que o Zabbix usa uma arquitetura gerente-agente, onde o gerente reside no Zabbix Server, e os agentes são enviados para os componentes que serão monitorados.

O trabalho do agente é simplesmente coletar as informações locais e enviar para o servidor, que fará seu processamento. Como essa comunicação é frequente, eventuais falhas de disco ou memória no componente em que o agente reside indica uma falha no sistema, que pode gerar o disparo de um alerta pelo Zabbix.

(VUNESP/UNICAMP/2022) O componente da solução de monitoramento Zabbix que permite a coleta de métricas específicas do Sistema Operacional da máquina monitorada, como CPU e memória, é o:

a) Zabbix Server
b) Zabbix Agent
c) Zabbix Proxy
d) Zabbix Metric
e) Zabbix Client

Comentários:

A questão descreve o funcionamento do Zabbix Agent, que fica alocado nas máquinas monitorados, coletando métricas locais. (Gabarito: Letra B)

Os agentes podem fazer checagens dos sistemas de duas formas: ativamente ou passivamente. No *check* passivo, o agente responde a uma requisição de dados, advinda do Zabbix Server ou de algum Proxy. Já no *check* ativo, os agentes consultam a lista de itens para monitoramento no Zabbix Server e faz um processamento independente, enviando ativamente novos valores, em intervalos definidos.

Em sistemas operacionais derivados do Unix, os agentes rodam como *daemon*, em segundo plano. Já no Windows, eles rodam como um serviço do Windows. Além disso, é possível iniciar múltiplos agentes em um mesmo dispositivo.

(VUNESP/Pref. F.co Morato/2022) No contexto do software de monitoramento Zabbix, o conceito de checagem passiva é caracterizado pelo seguinte processo:

a) O sistema operacional do dispositivo no qual o agente Zabbix está instalado envia os dados monitorados diretamente para o servidor Zabbix, sem passar pelo agente.
b) O usuário do dispositivo no qual o agente Zabbix está instalado precisa confirmar, por meio de uma interface de usuário, o envio de cada dado monitorado ao servidor Zabbix.
c) Primeiro, o agente Zabbix obtém uma lista de itens para monitoramento do servidor Zabbix. Periodicamente, os dados monitorados são enviados do agente ao servidor sem que este requisite.
d) O agente Zabbix, em execução no dispositivo monitorado, responde a requisições de dados do servidor Zabbix.
e) O agente Zabbix salva localmente um histórico de dados monitorados, sem enviá-los ao servidor Zabbix.

Comentários:

A checagem passiva é caracterizada pela concentração do polo de início de requisições de informações dentro do Zabbis Server. O Agente só irá agir após requisitado pelo servidor, fazendo o monitoramento e enviando os dados para o componente central. A alternativa que melhor descreve esse processo é a letra D.

Interessante também ressaltar a alternativa C, que traz a abordagem de monitoramento ativo. Mas, como a questão cobra o conhecimento sobre a checagem passiva, a alternativa correta é a letra D. (Gabarito: Letra D)

Bom, tudo que vimos até agora refere-se aos agentes padrões do Zabbix. Porém, existe uma nova geração de agentes, chamadas de Agentes 2. Ele foi desenvolvido para:

- Reduzir a quantidade de conexões TCP
- Prover uma abordagem de checagem concorrente melhorada
- Ser extensível com plugins
- Substituir os agentes tradicionais

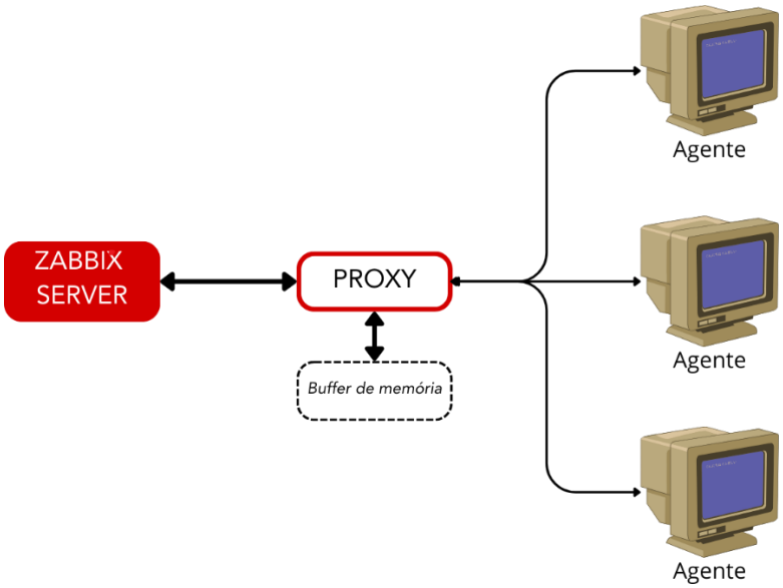
Esse novo agente é escrito na linguagem Go, e fornece o mesmo suporte para checagem ativas e passivas que o agente tradicional. Por padrão, sempre que reiniciado, um agente será configurado para checagens ativas. Além disso, é possível implementar uma checagem concorrente. Assim, um agente pode fazer checagem múltiplas e simultâneas.

Proxy

Um Zabbix Proxy é um processo que coleta dados de monitoramento de um ou mais equipamentos monitorados e os envia para o Zabbix Servir, trabalhando autonomamente em prol

do servidor. Todos os dados coletados são armazenados localmente, em *buffers*, e então transferidos ao servidor ao qual o Proxy pertence.

Eles irão atuar como um intermediário, entre os agentes e o servidor, veja:



O uso de *proxies* é opcional, mas pode ser muito beneficial para distribuir a carga de trabalho do servidor Zabbix, sem sobrecarregá-lo.

▪

(VUNESP/UNICAMP/2022) Em uma rede monitorada pela Solução de Monitoramento de Redes Zabbix, o Zabbix Proxy permite que

a) sejam replicados os dados do Zabbix Server para outro Zabbix Server.
b) a interface web de configuração do Zabbix Server possa ser acessada por meio de um HTTP Proxy.
c) o banco de dados do Zabbix Server possa ser compartilhado com outras instâncias do Zabbix Server.
d) sejam coletados dados de hosts localizados em redes distintas por meio de um único ponto de acesso.
e) sejam enviados dados para serviços externos, como os de gestão de incidentes para produção de alertas.

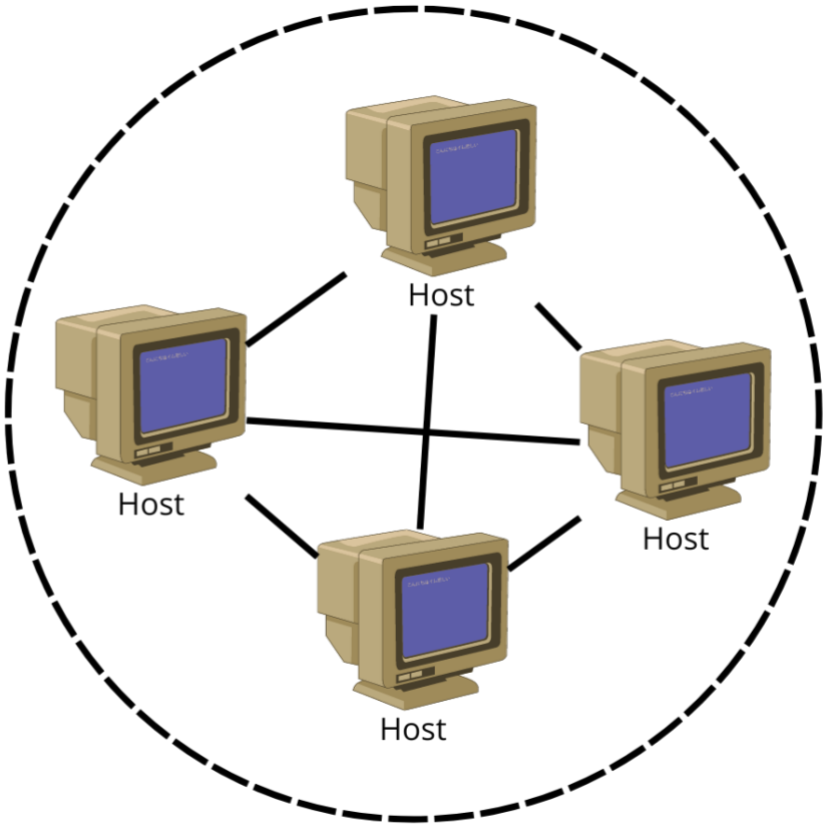
Comentários:

O Proxy atua como um coletor, que recebe e/ou coleta as informações advindas dos agentes, facilitando a carga de trabalho do servidor. Ele armazena as mensagens num *buffer* local e periodicamente as envia para o servidor, atuando como um ponto intermediário no fluxo de coleta das métricas. A alternativa que melhor descreve esse funcionamento é a letra D. (Gabarito: Letra D)

Hosts

No contexto do Zabbix, um *host* é qualquer dispositivo físico ou virtual, aplicação, serviço ou qualquer outra forma lógica que possa ser monitorada e ter suas métricas e parâmetros coletados. A criação e a configuração de *hosts* é um dos processos mais essenciais no Zabbix, já que eles definem quem deverá ser monitorado.

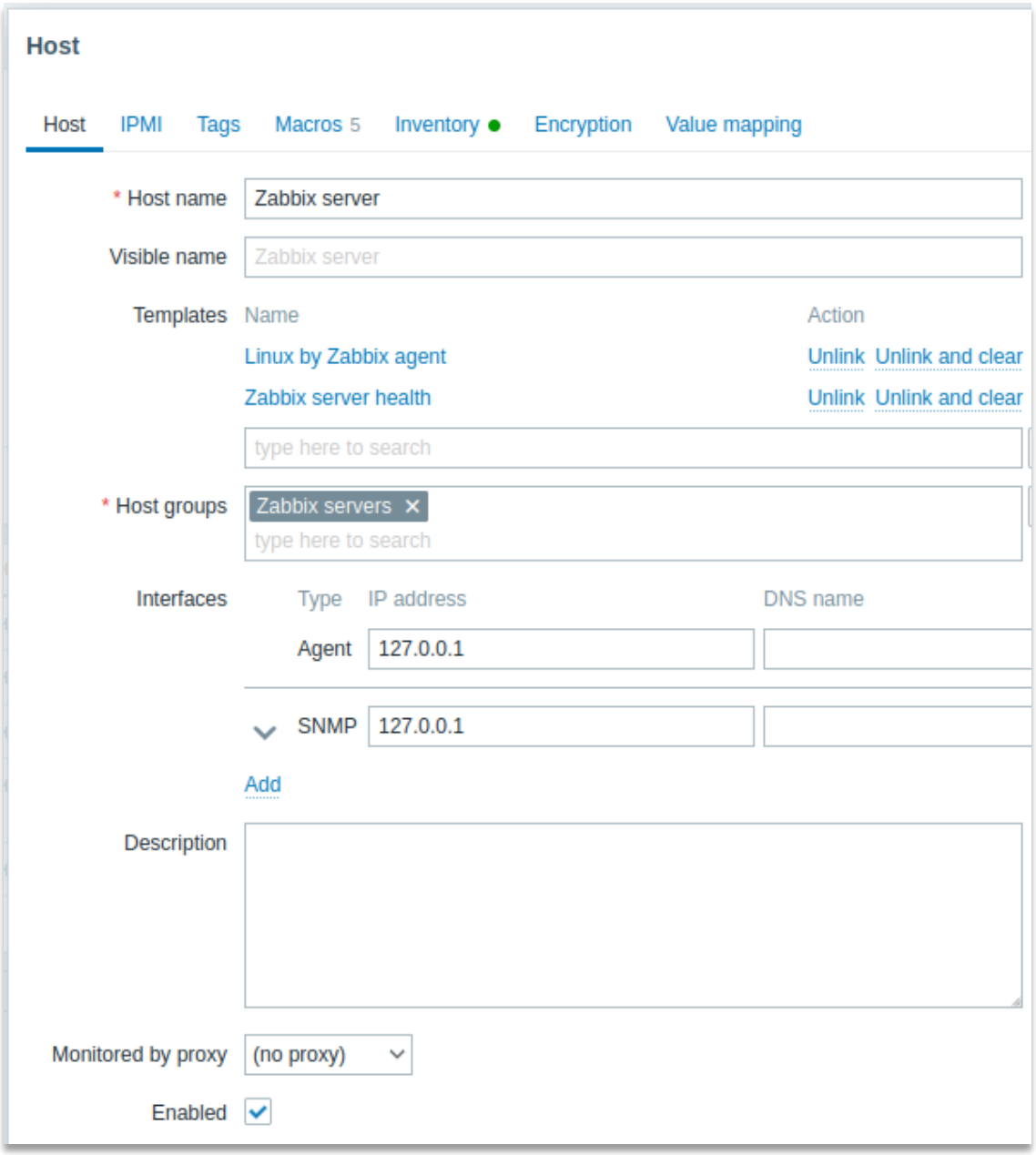
Quando agrupados, os *hosts* representam os *hosts groups*. Esses grupos podem ser pontos geográficos, unidades com mesmo fim lógico, ou qualquer outro parâmetro que ajude a organizar os processos. Além disso, um mesmo *host* pode pertencer a múltiplos grupos diferentes.



Cada *host* contará com uma lista de componentes:

- Um nome único, que servirá como forma de referência para comandos e outros
- Uma interface, que expõe os *endpoints* de coleta de informações
- Itens, que definem as métricas que serão coletadas
- Gatilhos, que trazem automação para o monitoramento
- Gráficos associados às suas métricas, para melhor visualização
- Entre outros

Veja a interface de configuração de um *host*:



Perceba os parâmetros que podemos passar para a configuração - nome, *templates* (que são formas pré-prontas de configuração), interfaces, grupos e até mesmo a atribuição de *proxies*.

Após a configuração, todos os detalhes dos *hosts* ficam armazenados num elemento chamado de inventário (*inventory*). Essa ferramenta centraliza as informações e facilita o gerenciamento de detalhes, como hardware, software, e outros atributos associados a cada um dos *hosts*. Dessa forma temos um repositório compreensivo de cada *host*, facilitando o uso da ferramenta.

Utilidades de Linha de Comando

O Zabbix oferece três funcionalidades, ou utilidades, utilizadas na linha de comando (CLI - *Command Line Interface*) para aprimorar o sistema: Sender, Get e JS. Vamos vê-las.

OBS: Não vi ser cobrado ainda, mas nunca duvido das bancas. Todas essas utilidades podem ser implementadas também em Python, através da biblioteca `zabbix_utils`.

Sender

O Sender é uma utilidade com o objetivo de enviar dados de performance para o Zabbix, para que possam ser processados. Essa utilidade é usada em *scripts* de longa duração, para envio periódico de disponibilidade e de dados de performance. Para enviar esses dados diretamente ao Server, é necessário um item do tipo *trap*, assunto que veremos logo em seguida.

Essa ferramenta é chamada de utilidade de CLI pois ela funciona a partir das interfaces de linha de comando, como o Bash ou o Shell. No caso do Sender, a sintaxe para envio das informações será escrita no nó *host*, isso é, o nó que deseja enviar informações. A sintaxe genérica tem o seguinte formato:

```
Shell
zabbix_sender -z <endereço do servidor> -p <porta> -s <nome do host> -k
<chave do item> -o <valor>
```

Onde:

- -z ⇒ especifica o endereço do servidor Zabbix ou do proxy.
- -p ⇒ especifica a porta do servidor. Por padrão, usa-se a porta 10051/TCP.
- -s ⇒ especifica o nome do host, definido na configuração do Zabbix.
- -k ⇒ especifica a chave do item para o dado que está sendo enviado.
- -o ⇒ especifica o valor enviado ao servidor.

Podemos ter então, por exemplo, a seguinte sintaxe:

```
Shell
zabbix_sender -z 192.168.1.100 -s MyHost -k custom.item -o 42
```

Acima, o valor 42 está sendo enviado do *host* chamado de MyHost para o item custom.item. Esse item está armazenado num servidor Zabbix locado no endereço 192.168.1.100. Veja que não foi definida a porta para a conexão, portanto estamos usando a porta padrão do Zabbix, 10051.

Get

O **`Get`** é, basicamente, o oposto do `Sender`. Enquanto o `Sender` atua no *host* enviando o dado para o servidor, o **`Get`** é realizado dentro do contexto do servidor, para recuperar as informações dos diferentes *hosts*. Essa ferramenta é utilizada para o *troubleshooting* (solução de problemas) dos agentes, quando eventuais falhas acontecem.

A sintaxe do Get é muito similar ao Send, inclusive nos parâmetros. Veja:

```
Shell
zabbix_get -z <endereço do servidor> -p <porta> -s <nome do host> -k
<chave do item> -t <tempo de timeout>
```

A diferença é que não temos mais o parâmetro de valor `-o`, já que queremos recuperar justamente esse número. Além disso, temos um parâmetro de *timeout*, o `-t`, que define um tempo para que a requisição seja descartada. Esse valor pode variar entre 1 e 30 segundos, sendo, por padrão, de 30 segundos.

JS

O JS é uma utilidade utilizada para incorporar testes de *script*. Com essa utilidade, podemos incorporar *scripts* no Zabbix e os executar, passando parâmetros de *string* e printando o resultado. Os *scripts* são executados a partir do *engine* de *script* incorporado do Zabbix. O JS do nome é derivado do JavaScript, formato de arquivo utilizado para armazenar o *script*.

O comando para rodarmos o JS pode ser passado com um arquivo de input -i, ou com um parâmetro -p.

```
Shell
zabbix_js -s <arquivo de script> -i <arquivo de input>
zabbix_js -s <arquivo de script> -p <arquivo de parâmetro>
```

Os parâmetros apresentados, além dos outros possíveis, são:

- -s ⇒ especifica o arquivo do *script*, no formato .js.
- -i ⇒ especifica o arquivo de *input*, que servirá como uma espécie de parâmetro.

- -p ⇒ especifica o parâmetro, diretamente no comando.
- -t ⇒ especifica um tempo de *timeout*.
- -l ⇒ define um nível de *log* para o script.

QUESTÃO DE PROVA

(FEPESE/CIASC/2017 - Adaptada) Analise as afirmativas abaixo com relação ao software Zabbix.

1. O Zabbix Get é um utilitário de linha de comando que pode ser utilizado para enviar dados para o Zabbix Server.
2. O Zabbix suporta tanto "pooling" quanto "trapping".
3. O Zabbix Get é um utilitário de linha de comando que pode ser utilizado para se comunicar com o agente de monitoração do Zabbix e requisitar um dado do agente.

Assinale a alternativa que indica todas as afirmativas corretas.

a) São corretas apenas as afirmativas 1 e 2.
b) São corretas apenas as afirmativas 1 e 3.
c) São corretas apenas as afirmativas 1, 2 e 3.
d) São corretas apenas as afirmativas 2 e 3.
e) Nenhuma afirmativa é correta

Comentários:

Vamos analisar cada item.

1. Errado. O utilitário que é utilizado para enviar dados de um *host* para o Zabbix Server é o Sender, não o Get.
2. Certo. O Zabbix atua em ambas as abordagens.
3. Certo. O Get é o comando executado pelo Server para recuperar dados dos diferentes *hosts*, usualmente quando há problemas nas conexões. Ele faz uma requisição de determinada informação que deve ser respondida pelo host.

Os itens corretos são, portanto, 2 e 3. (Gabarito: Letra D)

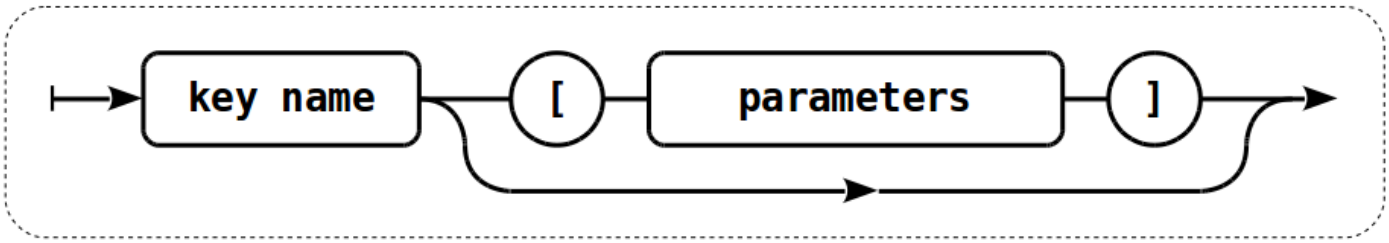
Itens

Os itens são blocos fundamentais no Zabbix, utilizados para monitorar vários aspectos dos *hosts*, como servidores, dispositivos, aplicações e outros. Um item é considerado uma métrica individual, e define o tipo de dado que deve ser coletado nas entidades monitoradas. Após coletado, o dado pode ser usado para alertas, análises e outros.

Para que possa ser especificado o tipo de dado que irá ser coletado do *host*, usa-se um conceito chamado de **`chave do item`**. Essa chave, ou *key*, é um identificador único que especifica exatamente a métrica ou o dado que um item deve coletar. Com isso, percebemos a abordagem chave-valor (*key-value*) empregada no contexto do Zabbix.

Uma chave de item é definida por dois campos:

- Nome da chave (*key name*): a chave do item, que define qual a métrica deve ser coletada e identifica unicamente o item.
- Parâmetros (*parameters*): são valores atribuídos à chave, que trazem detalhes adicionais ou contexto às métricas coletadas



Atrelado a cada chave, teremos os valores, que são as medidas ou métricas coletadas nos *hosts*. Antes de enviar esse dado para o banco de dados, é possível o pré-processamento desses dados, procurando deixá-lo em formatos mais adequados para o banco de dados. Pode-se aplicar processamento de *scripts*, conversão de formatos (JSON, CSV etc.), entre outros.

O Zabbix conta com 18 tipos de itens diferentes. Cada item possui chaves de itens suportadas pré-definidas. Caso queira verificar na íntegra, como leitura extra, já que as bancas tendem a não cobrar esse conteúdo, você pode clicar aqui. Vou montar uma tabela com os diferentes itens, para você ter acesso mais rápido a esse conteúdo.

Item	Descrição
Agente Zabbix	Coleta dados do host onde o agente Zabbix está instalado. Essa coleta pode ser feita de forma passiva, por padrão, ou ativa, declarada expressamente na criação do item.
Agente SNMP	Coleta dados de dispositivos de rede usando o Protocolo Simples de Gerenciamento de Rede (SNMP). Pode obter informações de dispositivos como roteadores, switches e impressoras.
Traps SNMP	Recebe e processa traps SNMP, que são notificações assíncronas enviadas por dispositivos habilitados para SNMP quando certos eventos ocorrem.
Verificações IPMI	Coleta dados de dispositivos usando a Interface Inteligente de Gerenciamento de Plataforma (IPMI), frequentemente usada para monitorar a saúde do hardware e o gerenciamento de servidores.
Verificações Simples	Realiza verificações básicas de rede, como ping, para determinar a disponibilidade e a capacidade de resposta de serviços e dispositivos de rede.
Monitoramento de Log	Monitora arquivos de log no host, capturando e analisando entradas de log com base em padrões especificados.
Itens Calculados	Usa dados de outros itens para realizar cálculos e transformações, permitindo análises e agregações de dados complexas.
Verificação Interna	Monitora métricas internas do servidor Zabbix, como o número de valores processados por segundo ou o número de triggers em estado de problema.
Verificação SSH	Executa comandos em hosts remotos via SSH e coleta a saída. Útil para monitorar sistemas onde o agente Zabbix não pode ser instalado.
Verificação Telnet	Executa comandos em hosts remotos via Telnet e coleta a saída. Semelhante às verificações SSH, mas usa o protocolo Telnet.
Verificação Externa	Executa scripts ou programas externos para coletar dados. Esses scripts são rodados como scripts de Shell ou binários, e executados pelo servidor ou proxy Zabbix.
Itens Trapper	Aceita dados enviados ao servidor ou proxy Zabbix por aplicativos ou scripts externos, proporcionando uma maneira flexível de integrar com outros sistemas.

Monitoramento JMX	Coleta dados de aplicativos Java usando Java Management Extensions (JMX), permitindo monitorar detalhadamente o desempenho e as métricas de aplicativos Java.
Monitoramento ODBC	Coleta dados de bancos de dados usando Open Database Connectivity (ODBC), permitindo monitorar o desempenho do banco de dados e executar consultas SQL personalizadas.
Itens Dependentes	Usa dados de um item mestre para realizar processamento ou transformação adicional, reduzindo a carga nos hosts monitorados ao reutilizar dados existentes.
Agente HTTP	Coleta dados de serviços web realizando solicitações HTTP, útil para monitorar a disponibilidade e o desempenho de aplicações web e APIs.
Verificações Prometheus	Coleta dados de endpoints Prometheus, permitindo a integração com ambientes monitorados pelo Prometheus e aproveitando métricas do Prometheus no Zabbix.
Itens de Script	Executa scripts personalizados no servidor ou proxy Zabbix para coletar dados, em JavaScript ou Python, proporcionando flexibilidade para monitorar uma ampla gama de métricas personalizadas e realizar tarefas complexas de coleta de dados.

QUESTÃO DE PROVA

(Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta de monitoramento Zabbix, julgue o item abaixo.

Os itens no Zabbix são componentes que definem ações e tipos de dados a serem coletados. Dentre os itens, destaca-se as *traps* SNMP, que fazem uma comunicação ativa e assíncrona com o servidor Zabbix através do protocolo SNMP.

Comentários:

Perfeito! Uma *trap* SNMP é uma mensagem que não segue o fluxo regular de mensagens, pois ela parte de uma iniciativa dos *hosts* - caracterizando uma comunicação ativa. Além disso, essas *traps* ficam armazenadas no MIB, um repositório local, permitindo uma comunicação assíncrona com o servidor. (Gabarito: Certo)

Gatilhos (*trigger*)

Os gatilhos, ou ***triggers***, são componentes fundamentais no monitoramento do Zabbix, sendo composto por expressões que avaliam os dados coletados pelos itens, e representam seu estado corrente. Enquanto itens são usados para coletar dados, é impraticável seguir todos esses dados o tempo todo, devido ao seu volume. Os ***triggers*** implementam expressões que permitem definir um limite para o qual um dado é considerado aceitável.

Se determinado dado atingiu alguma métrica definida como limite, seja tamanho, quantidade, tipo de informação, um *trigger* é disparado, criando uma mudança de estado. Dentro dos gatilhos, podemos ter três estados (*status*) diferentes:

Status	Descrição
OK	Um estado normal de *trigger*.
Problem	Algum problema ocorreu, como, por exemplo, carga de processamento muito elevada
Unknown	O valor do ***trigger*** não pôde ser calculado

O *trigger* funciona basicamente como um limitador. Valores abaixo do limite são ignorados; valores acima do limite disparam o gatilho mudando o valor de `OK` para `Problem` (ou `unknown`, se não for calculável).

Vamos para um exemplo: imagine que estamos querendo verificar o processamento da carga da CPU. No nosso exemplo queremos achar uma média da carga durante 5 minutos. A sintaxe para implementarmos esse *trigger* seria a seguinte:

```
Shell
avg(/<nome do host>/<chave do item>, 5m) > 2
```

Acima, especificamos o endereço do host e a chave do item que queremos, que irá especificar a carga da CPU. Essa chave usualmente é a system.cpu.load. Esse valor é passado como parâmetro para a função `avg` (*average* - média), que calculará a média a cada 5 minutos. Usualmente os valores chegam de 1 em 1 minuto, então 5 valores serão agregados para o cálculo.

Caso a média encontrada seja superior a 2, o gatilho será disparado e haverá uma alteração de *status* na métrica para `Problem`. Esse disparo pode resultar em determinada ação, que é uma resposta a esse problema, que pode consistir na execução de um *script*, voltado para a resolução automática do problema, ou um alerta por e-mail, SMS ou outros, para uma resolução manual.

QUESTÃO DE PROVA

(SELECON/CM São Gonçalo/2022) Na atividade de monitoramento de infraestrutura, o Zabbix traz uma série de definições que são necessárias para a sua operação. Dentre elas, quatro são fundamentais:

I. É uma entidade lógica que agrupa itens de interesse para o monitoramento. Exemplos são um servidor, um roteador ou um switch.
II. É um indicador que será monitorado, como a taxa de uso de CPU, o espaço em disco ou o tempo de resposta de um serviço.
III. E um recurso a partir do qual alguma notificação será gerada automaticamente. E criado no caso da necessidade de notificações sobre a ocorrência de eventos específicos, tais como espaço em disco de um servidor ou consumo de banda de um link.
IV. É uma atividade que será executada quando o recurso em III for acionado. São exemplos o envio de e-mail, SMS ou a execução de scripts.

Os termos definidos em I, II, III e IV são, respectivamente:

a) item, trigger, ação e hosts
b) trigger, ação, hosts e item
c) ação, hosts, item e trigger
d) hosts, item, trigger e ação

Comentários:

Mesmo sendo de uma banca menor, essa é uma excelente questão. Vamos associar cada item aos componentes do Zabbix.

I. A entidade lógica que agrupa os itens de monitoramento é o próprio dispositivo sendo avaliado - denominado de *host*.
II. O item se refere ao item, que especifica um indicador que será monitorado pelo Zabbix.
III. Essa é a definição de um *trigger*.
IV. A atividade executada após o *trigger* é uma ação.

Portanto, ficamos com host - item - trigger e ação. A alternativa correta é a letra D. (Gabarito: Letra D)

Eventos

Eventos são acontecimentos diversos no contexto do Zabbix. Por exemplo, a mudança de estado causada pelo ativamento de um *trigger* ocasiona um evento. Dentro do Zabbix, temos diversos eventos que podem ser detectados e capturados. Cada evento recebe um carimbo de tempo (*timestamp*), e pode servir como base para uma determinada ação.

São diversos eventos no contexto do Zabbix. Os principais são:

- Eventos de *trigger* ⇒ ocorre quando um *trigger* altera seu estado (OK → PROBLEM → OK).
- Eventos de serviço ⇒ ocorre quando um serviço altera o seu próprio estado.
- Eventos de descoberta ⇒ ocorre quando um novo *host* ou serviço é descoberto.
- Eventos de auto registro ⇒ ocorre quando agentes ativos são registrados pelo servidor.
- Eventos internos ⇒ ocorre quando itens ou regras de descoberta deixam de ser suportados, ou quando um *trigger* vai para o estado `unkown`

(Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Zabbix, julgue o item abaixo.

Ao ser disparado e modificar o estado de um item para *unkown*, o *trigger* gera um evento de *trigger*.

Comentários:

Muito cuidado! Somente a mudança OK → PROBLEM e PROBLEM → OK disparam eventos de *trigger*. Quando a mudança é para `unknown`, temos um evento interno. (Gabarito: Errado)

Descoberta

O Zabbix oferece um serviço automático de descoberta, destinado a encontrar redes, componentes, terminais, servidores ou outros elementos na rede e que possuam potencial para serem monitorados. Muitas vezes a descoberta manual é trabalhosa, exige configurações intensivas, e o serviço de descoberta traz uma solução para esse gargalo.

A descoberta de endereços do Zabbix é baseada nas seguintes informações:

- Intervalos de endereços IP
- Disponibilidade de serviços externos (FTP, SSH, WEB, POP3, IMAP, TCP etc.)
- Informações recebidas dos agentes Zabbix
- Informações recebidas dos agentes SNMP

O processo de descoberta de uma rede consiste em duas fases distintas: descoberta e ações.

Na descoberta, o Zabbix escaneia os intervalos de endereços IP periodicamente, baseado nas regras de descoberta armazenadas no banco de dados. Cada regra de descoberta possui um intervalo de verificação particular. Além disso, cada verificação de serviços e *hosts* dispara um evento de descoberta. Veja quais são os eventos possíveis:

Evento	Motivador do evento
Service Discovered	O serviço está *up*, após estar *down*, ou quando é descoberto pela primeira vez
Service Up	O serviço está *up*, após já estar *up*
Service Lost	O serviço está *down*, após estar *up*
Service Down	O serviço está *down*, após já estar *down*
Host Discovered	Pelo menos um serviço de um *host* está *up*, após todos os serviços estarem *down*, ou um serviço é descoberto sem ter seu *host* registrado
Host Up	Pelo menos um serviço está *up*, após pelo menos um serviço estar *up*
Host Lost	Todos os serviços estão *down*, após ao menos um estar *up*
Host Down	Todos os serviços estão *down*, após todos estarem *down*

Já as ações são resultados da descoberta. Elas envolvem ações, literalmente, por parte do Zabbix, de forma automática ou manual. Podemos ter envios de notificações, a adição ou remoção de um *host*, a adição ou remoção de um *host* de um grupo, entre outros. São respostas aos eventos, que podem ser configuradas para serem disparadas automaticamente ou como um alerta para instruções manuais.

Um ponto importante aqui são as regras de descoberta. Elas são instruções manuais, passadas diretamente na UI do Zabbix e armazenadas no seu banco de dados. Nele, definimos parâmetros como intervalo de endereços IP para varredura (por exemplo, 192.168.1.1-254), o intervalo de atualizações, o tipo de verificação de descoberta (com tipo de agente, porta, entre outros), entre outros critérios que ficarão armazenados no banco de dados e utilizados para a descoberta. É importante que para cada intervalo de endereços se tenha uma única regra.

Outra forma de registrar eventuais nós da rede que desejam ser monitorados é a partir do auto registro de agentes ativos. Como agentes ativos possuem a capacidade de enviar dados e informações diretamente para o servidor sem necessidade de uma requisição, eles usam esse mecanismo para "avisar" o servidor de sua existência, registrando-os em seu banco de dados.

Por fim, temos a descoberta de baixo nível, ou Low Level Discovery (LLD). Ela é uma forma de criar automaticamente itens, gatilhos e gráficos para diferentes entidades em um computador. Por exemplo, o Zabbix por automaticamente iniciar a monitorar interfaces do computador, sem a necessidade de criar itens para cada interface manualmente. Ele funciona, portanto, como uma forma de automação do processo de descoberta.

QUESTÃO DE PROVA

(Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Zabbix, julgue o item abaixo.

A Descoberta de Baixo Nível (LLD - Low Level Discovery) é uma ferramenta utilizada para implementar um processo de descoberta automático no ecossistema do Zabbix, encontrando serviços e *hosts* automaticamente, sem necessidade de configurações manuais.

Comentários:

Perfeito! O LLD introduz a automação no processo de descoberta, facilitando o monitoramento por parte do Zabbix, evitando erros de inserções manuais. (Gabarito: Certo)

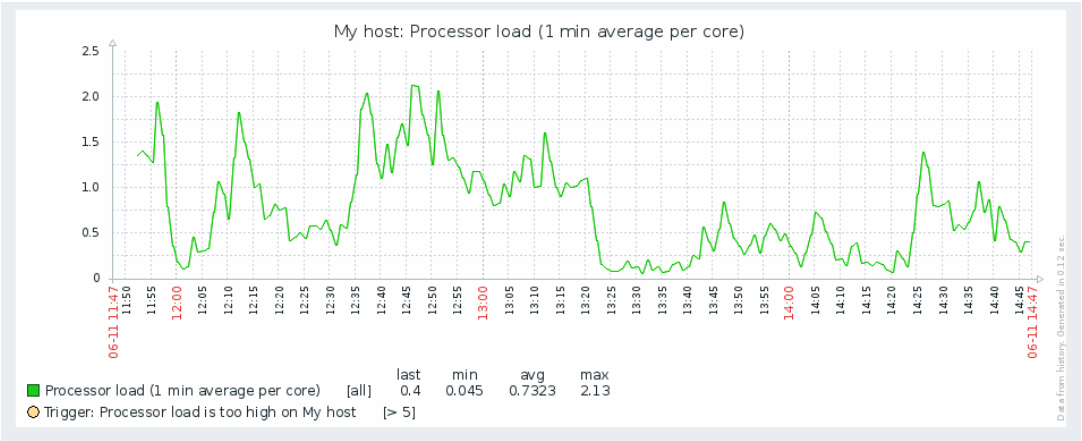
Visualização

Como comentei anteriormente com você, muitas ferramentas precisam de aplicações extras para criar as visualizações. No Zabbix não, toda a parte de visualização com gráficos, *dashboards* e outros, é nativa e integrada. Elas agem como um auxílio visual para o monitoramento dos dispositivos e serviços no Zabbix.

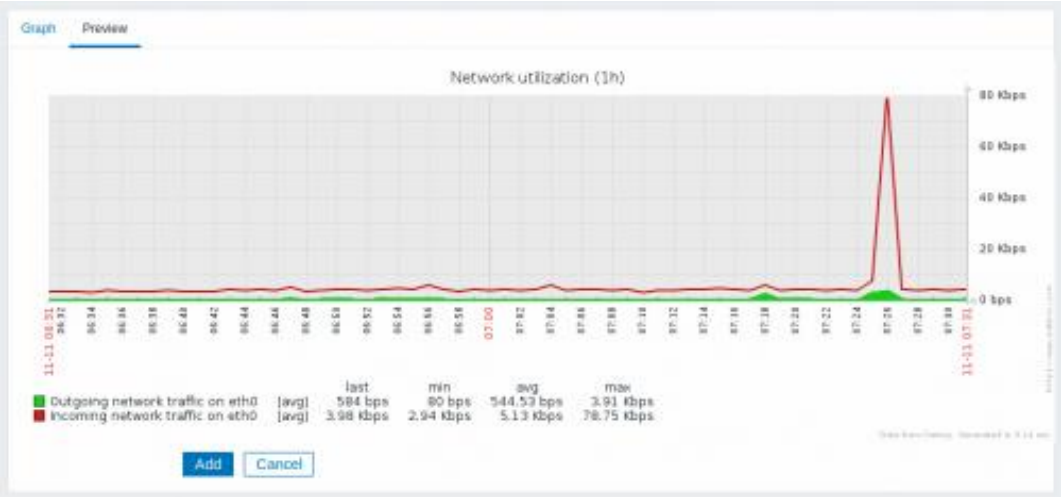
As visualizações são divididas em 3 grupos: gráficos, mapas de rede e dashboards.

Os gráficos são a principal forma de visualizar os dados que chegam no servidor. Como as quantidades de dados são muito altas, muitas vezes é difícil observar diretamente nos dados. Nesse contexto os gráficos ajudam e muito, já que fica mais fácil de analisar tendências, *outliers*, entre outros. O Zabix conta com 4 tipos de gráficos:

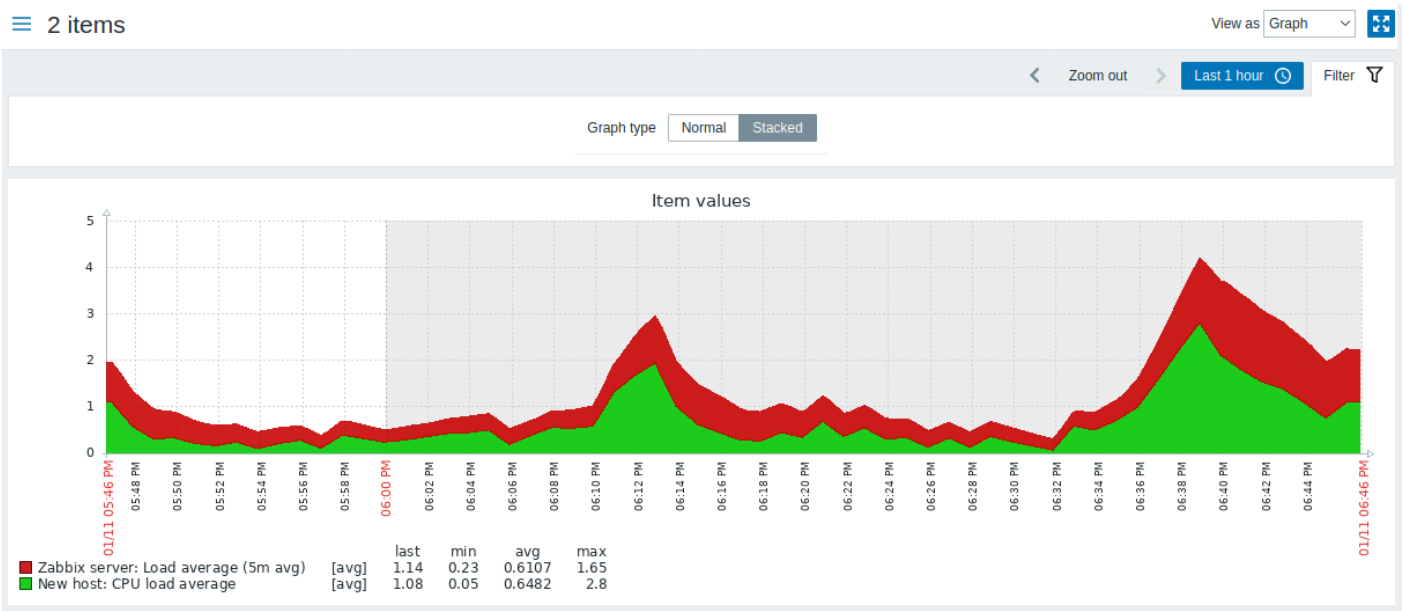
- Gráficos simples: são visualizações mais diretas, que envolvem apenas uma fonte de origem de dados.



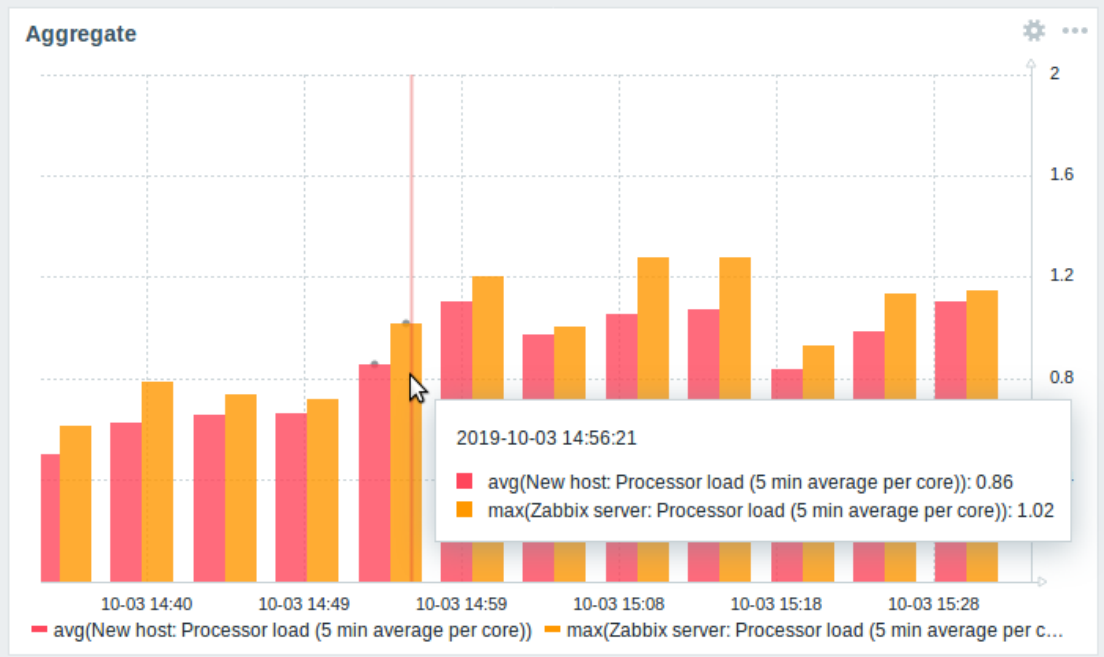
- Gráficos customizados: são gráficos derivados dos gráficos simples, ainda direcionados a só um dado, mas que permitem customizações de aparência.



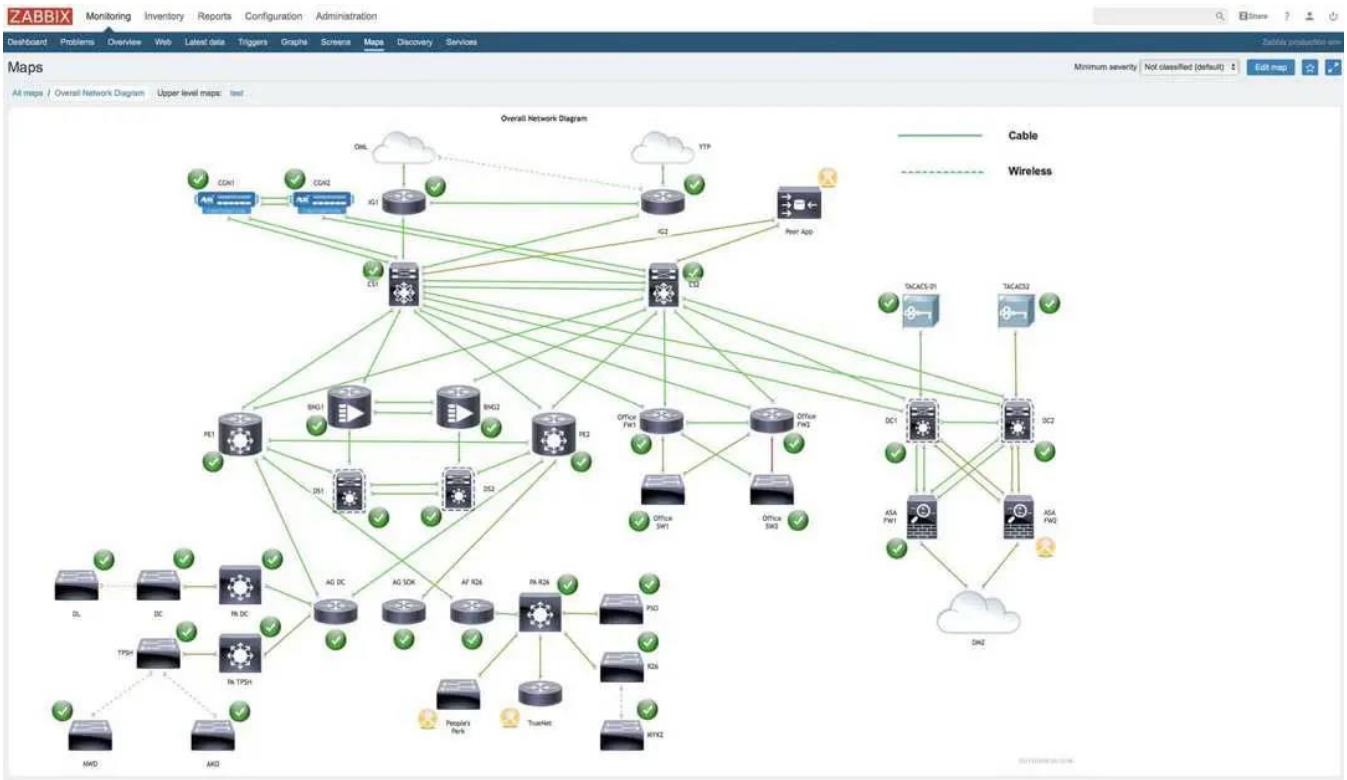
- Gráficos ad-hoc: permitem a análise de múltiplos dados simultâneos, permitindo a comparação entre diferentes métricas.



- Gráficos de agregação: aplicam funções de agregação sobre um ou mais dados. As funções de agregação disponíveis são: mínimo/máximo, média, quantidade, soma, primeiro/último valor.

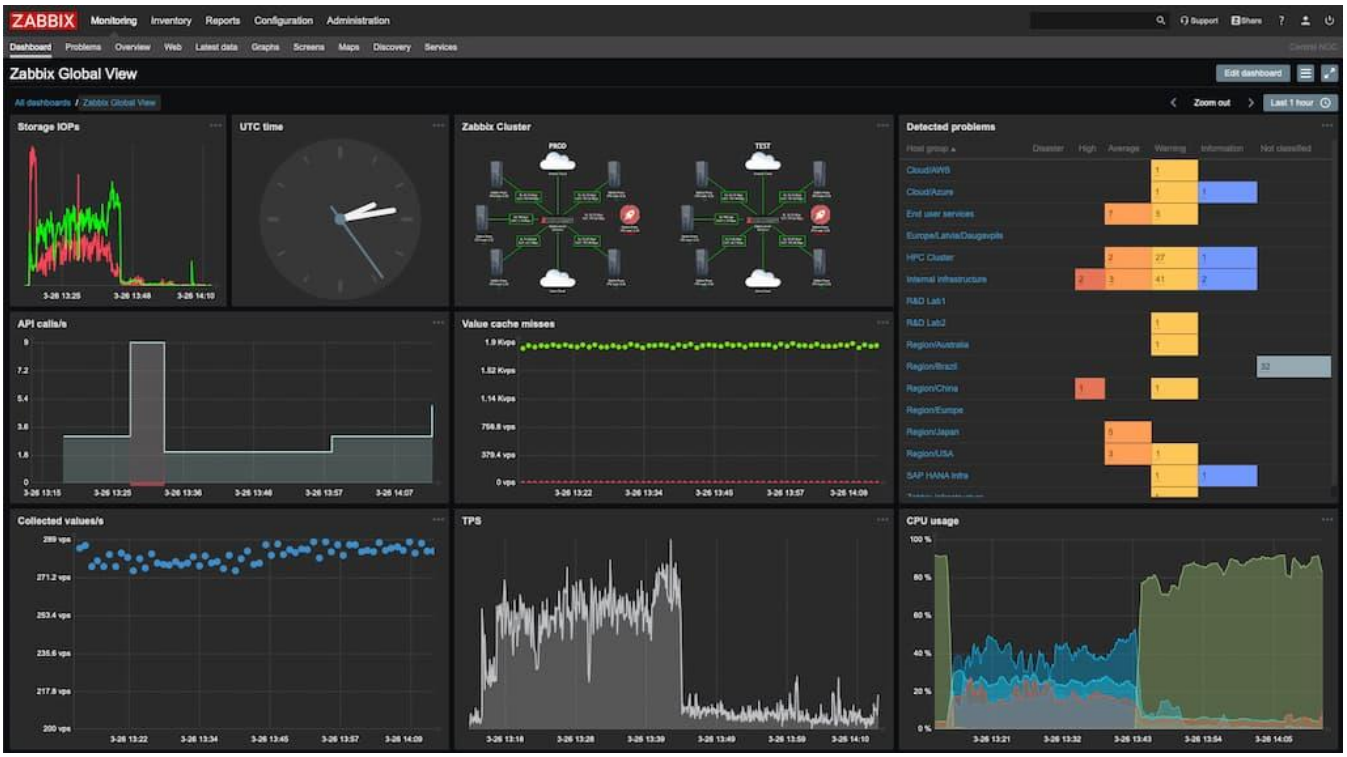


A segunda forma de visualização no Zabix são os mapas de redes. Ele é um mapa destinado a representar os *hosts* monitorados e seus relacionamentos, ligações entre eles, agrupamentos, de uma forma visual e fácil de identificar. Com ele fica mais fácil entender pontos de gargalo, eventuais problemas pontuais, ou outros problemas que podem aparecer na execução de redes.



Por fim, finalizamos as visualizações com os dashboards. Ele é uma ferramenta altamente configurável e customizável, que traz diversos elementos diferentes para visualização, baseado em

"widgets". Um *widget* é um contêiner que pode conter qualquer tipo de visualização gerada pelo Zabbix. Com o Dashboard temos uma visualização geral do sistema, agregando as principais visualizações e facilitando a manutenção de um sistema de rede.



API

Uma API é responsável por fazer comunicações entre clientes e servidores. No Zabbix, sua API permite a recuperação e modificação de configurações do programa, além de acessar dados históricos. Seus usos comuns são:

- Criar novas aplicações para trabalhar de forma integrada com o Zabbix.
- Integrar o Zabbix com aplicações de terceiros.
- Automatizar rotinas de tarefas.

A API do Zabbix é baseada em HTTP, e é integrada à parte de *frontend* da aplicação (Zabbix Web). Ela utiliza o protocolo JSON-RPC 2.0, o que implica numa organização realizada a partir de um conjunto de métodos, com abordagem request/response (requisição/resposta) feita usando arquivos JSON.

Aqui é importante destacarmos um ponto - embora muitas aplicações utilizem APIs (principalmente APIs RESTful) na sua própria estrutura, para fazer comunicações com seus componentes, esse não é o caso do Zabbix. Suas comunicações são feitas de forma direta, com protocolos nativos (como o agente Zabbix, que age como um protocolo), ou protocolos externos, como o SNMP e o ICMP.

QUESTÃO DE PROVA

(FGV/TRT 13/2022) Analise as afirmativas a seguir sobre a ferramenta de monitoramento Zabbix:

I. Descobre automaticamente dispositivos na rede por meio da varredura de um range de IP definidos.
II. Coleta dados com intervalos personalizados, inclusive com agendamento exato do momento da coleta.
III. Recebe por meio de API RESTful as traps SNMP dos dispositivos monitorados.

Está correto o que se afirma em

a) I, apenas.
b) II, apenas.
c) III, apenas.
d) I e II,apenas.
e) I, II e III.

Comentários:

Vamos analisar cada item.

I. Certo. Esse é o trabalho do serviço de descoberta.
II. Certo. O Zabbix permite definições personalizadas para a coleta, inclusive com agendamentos.
III. Errado. O Zabbix recebe diretamente as *traps* SNMP. As APIs utilizadas pelo Zabbix tem intuito de integração com elementos externos.

Portanto, estão corretos os itens I e II. (Gabarito: Letra D)

QUESTÕES COMENTADAS

Zabbix

01. (FGV/ALEP/2024) Um gerente de sistemas instalou o software de gerenciamento de rede Zabbix, em uma rede Linux. Após a instalação ele deseja saber se o serviço do Zabbix entrou em execução.

Para esse caso, ele deve utilizar o comando

a) ls –l | grep zabbix
b) find –iname zabbix
c) ps aux | grep zabbix
d) ls -ARC | grep zabbix
e) tasklist /PID | grep zabbix

Comentários:

Essa questão acaba cobrando mais conhecimentos de Bash que de Zabbix em si. O primeiro passo é encontrar os processos em execução no sistema o que é feito pelo comando *process status* `ps`. Só com isso já conseguiríamos responder à questão - mas vamos seguir. Acompanhado do ps, podemos utilizar `aux`, uma opção que define a formatação da saída. Usamos um pipe `|` para redirecionar a saída de um comando ao outro - e passamos o comando de filtragem, `grep`, com o valor zabbix, para retornar somente o processo do Zabbix.

Portanto, o comando completo fica `ps aux | grep zabbix`. Correta a letra C.

Gabarito: Letra C

---

02. (CONSULPLAN/CM BH/2024) Zabbix é uma plataforma de monitoramento de rede projetada para monitorar a disponibilidade e o desempenho de dispositivos de rede, servidores e recursos de TI. Sobre os componentes da arquitetura do Zabbix, assinale, a seguir, a alternativa que é responsável por coletar dados de monitoramento.

a) Zabbix Proxy.
b) Zabbix Agent.
c) Zabbix Server.
d) Zabbix Web Interface.

Comentários:

Apesar dessa questão ter sido anulada, é interessante a analisarmos e entendermos o motivo dessa anulação. Dentro do Zabbix, os dados podem ser coletados (nativamente) de duas formas: através dos agentes (Zabbix, SNMP), que atuam de forma ativa e passiva, ou através dos proxies. Então, por conter duas alternativas corretas, a banca, corretamente, anulou a questão.

Gabarito: Anulada

---

03. (CONSULPLAN/CM BH/2024) O Zabbix oferece suporte ao monitoramento de uma variedade de dispositivos e pode monitorar sistemas operacionais como Linux, Windows e MacOS. Considerando o funcionamento do Zabbix, como o Zabbix trata os eventos de disparo que ocorrem em um ambiente monitorado?

a) Aciona automaticamente a resolução do problema.
b) Gera um alerta e aguarda a intervenção do usuário.
c) Armazena os eventos em um banco de dados para análise posterior.
d) Ignora os eventos até que o administrador os verifique manualmente.

Comentários:

Após a ocorrência de um evento no Zabbix, por padrão, é emitido um alerta para o gerenciador dos dados, para que ele tome uma providência. Eventualmente, pode haver configurações da ferramenta visando a resolução automática dos problemas, mas esse não é o comportamento padrão - que envolve a identificação, a emissão de um alerta e o armazenamento dessa informação. Portanto, a alternativa correta é a letra A.

Gabarito: Letra A

---

04. (CONSULPLAN/CM BH/2024) A interface gráfica do Zabbix oferece uma maneira fácil de configurar hosts, itens, triggers, visualizar gráficos e relatórios, dentre outras funcionalidades. Assinale, a seguir, a interface gráfica usada para configurar e monitorar o Zabbix.

a) Zabbix Console.
b) Zabbix Dashboard.
c) Zabbix Control Center.
d) Zabbix Web Interface.

Comentários:

O Zabbix, em seu *core*, tem três componentes: Zabbix Web, Zabbix Server e o Banco de Dados. O componente que fornece uma interface gráfica de usuário (GUI) para o monitoramento dos dados é o Zabbix Web. Cuidado com o *dashboard*, ele é uma ferramenta visual dentro da interface web, mas não é a interface em si.

Gabarito: Letra D

05. (FCC/TRT 18/2023) Na arquitetura do Zabbix, o proxy Zabbix

a) realiza a autenticação dos usuários do Zabbix e da rede e dos seus componentes.
b) realiza o monitoramento dos recursos dos servidores e aplicações, enviando dados obtidos para o servidor.
c) é um componente opcional e pode proporcionar a redistribuição de carga de coletas.
d) permite a interação entre o servidor Zabbix e o administrador e, normalmente, é instalado na máquina do servidor.
e) gerencia o armazenamento de dados recebidos pelo Zabbix.

Comentários:

O Proxy Zabbix é um dos componentes opcionais de coleta do Zabbix, responsável por atuar como um intermediário entre o Zabbix Server e os nós do coleta. O Proxy coleta os dados e os armazena num *buffer* local, enviando-os em lote para o servidor, quando disponível. Seu funcionamento é essencial para distribuir as cargas de coleta do Server para diferentes polos (os próprios proxies). A alternativa que corretamente descreve seu funcionamento é a letra C.

Gabarito: Letra C

06. (VUNESP/UNICAMP/2023) Na arquitetura da ferramenta Zabbix, é papel do Zabbix proxy

a) estabelecer uma VPN entre o Zabbix server e cada dispositivo monitorado.
b) funcionar como servidor web para visualização remota de dados monitorados.
c) monitorar ativamente recursos locais do dispositivo onde está instalado e enviar dados ao Zabbix server.
d) receber dados de um ou mais dispositivos monitorados e enviar ao Zabbix server.
e) servir como banco de dados a ser utilizado pelo Zabbix server.

Comentários:

Outra questão cobrando conhecimentos sobre o Zabbix Proxy. Lembre-se, ele é uma ferramenta que atua entre o Server e os agentes, recebendo (passivamente) e coletando (ativamente) os dados dos dispositivos monitorados. Vamos analisar as alternativas, ver qual se coaduna a essa concepção.

a) Errado. O Proxy não estabelece uma VPN.
b) Errado. Essa alternativa descreve o Zabbix Web.
c) Errado. Essa alternativa descreve o Zabbix Agent.
d) Certo. Essa é a definição correta do Zabbix Proxy.
e) Errado. Essa alternativa descreve um banco de dados qualquer, utilizado pelo Server.

Gabarito: Letra D

07. (AOCP/IF MA/2023) No contexto do monitoramento de aplicações com Zabbix, qual é a característica que ajuda a coletar informações detalhadas sobre o desempenho e a saúde das aplicações?

a) Zabbix suporta apenas o monitoramento de aplicações desenvolvidas em linguagens específicas.
b) Zabbix utiliza exclusivamente notificações por e-mail para alertar sobre problemas de desempenho.
c) Zabbix requer que os desenvolvedores insiram manualmente cada métrica de desempenho no código-fonte.
d) Zabbix permite a criação de templates personalizados para facilitar a configuração de itens de monitoramento.
e) Zabbix envia relatórios de monitoramento apenas para desenvolvedores designados, ignorando a equipe de operações.

Comentários:

Vamos analisar cada alternativa.

a) Errado. O Zabbix é agnóstica a linguagens, independendo da linguagem utilizada no dispositivo.
b) Errado. Podemos ter diversas notificações, como por SMS.
c) Errado. O Zabbix oferece formas de automatização, como o LLD (Low Level Discovery).
d) Certo. Os templates servem para replicar instalações e configurações do Zabbix.
e) Errado. O envio é feita para desenvolvedores e para grupos também.

Portanto, correta a letra D.

Gabarito: Letra D

08. (FUNDATEC/PROCERGS/2023) Zabbix é uma ferramenta de monitoramento de rede de computadores capaz de monitorar uma infraestrutura completa. Possui licença GPL de uso gratuito e dispõe de uma grande variedade de opções. Em relação às principais características do Zabbix, analise as assertivas abaixo:

I. Tem suporte a uma grande variedade de sistemas operacionais como Linux, Windows, Solaris, OpenBSD, Mac OS X, entre outros.
II. Tem integração com banco de dados (MySQL, Oracle, PostgreSQL ou SQLite) e geração de gráficos em tempo real.

III. Realiza a monitoria de serviços simples, como HTTP, POP3, IMAP, SSH, entre outros e possui suporte nativo ao protocolo SNMP.

Quais estão corretas?

a) Apenas II.
b) Apenas III.
c) Apenas I e II.
d) Apenas II e III.
e) I, II e III.

Comentários:

Vamos analisar cada item.

I. Certo. O Zabbix é suportado por uma grande gama de sistemas operacionais, incluindo os apontados pela questão.
II. Certo. O Zabbix se integra com alguns bancos de dados - mais especificamente MySQL, MariaDB, Oracle, PostgreSQL, SQLite e TimescaleDB.
III. Certo. De fato, o Zabbix suporta esses protocolos e tem suporte nativo ao SNMP, como aos agentes SNMP.

Portanto, todos os itens estão corretos.

Gabarito: Letra E

---

09. (FURB/SAMAE JS/2023) Existe um protocolo de rede que pode ser utilizado para gerenciamento e monitoramento de redes ou dispositivos compatíveis conectados à rede. Por exemplo, você pode monitorar de forma remota uma bomba de uma estação de tratamento de água para coletar as informações de operação dela por meio de uma ferramenta específica, como o Zabbix. As informações podem ser, então, utilizadas para gerar relatórios de desempenho ou alertas de funcionamento. Assinale a alternativa correta que representa o protocolo de rede que permite tal função:

a) SNMP
b) NTP
c) IMAP
d) NNTP
e) SMTP

Comentários:

Questão tranquila. O protocolo, dos listados na questão, que permite o monitoramento remoto de sistemas é o SNMP - Simple Network Management Protocol.

Gabarito: Letra A

---

10. (FGV/CGU/2022) Jane integra a equipe de TI de seu órgão da Administração Pública Federal (APF), que está assinando documentos com seus clientes nos quais os níveis de serviços (Service Level Agreement – SLA) estão sendo definidos para cobrança do serviço prestado.

Para obter os dados para os relatórios de SLA, Jane deve implementar a coleta automática de dados de disponibilidade e geração de SLA no Zabbix, realizando as seguintes ações:

a) configuração da correlação de eventos no monitoramento de arquivos de registros (log) dos serviços para identificar as suas indisponibilidades;
b) instalação do protocolo SNMP nos servidores e configuração da comunidade SLA para permitir coletas múltiplas dos dados de limites definidos para os SLA;
c) configuração das funções preditivas para ativação de triggers que possam restabelecer a disponibilidade de um serviço em caso de incidente, garantindo o SLA acordado;
d) instalação do agente Zabbix nos servidores e configuração do monitoramento dos serviços ofertados e de um serviço de TI com esses servidores;
e) ativação da monitoração Web com cenários de testes automáticos e rotineiros para os serviços hospedados, para validação de sua disponibilidade.

Comentários:

Questão difícil. A coleta automática dentro do Zabbix é fornecida de algumas formas: com o uso de agentes Zabbix ativos, através de proxies ou através de mensagens *trap* do protocolo SNMP. Vamos ver qual alternativa, vendo qual aborda corretamente esse conceito.

a) Errado. Eventos são respostas do Zabbix a mudanças de estados ou acontecimentos no sistema, mas eles, por si, não realizam coletas de informações.
b) Errado. A instalação do protocolo não é suficiente para recuperar as informações, precisam ser operados eventuais agentes SNMP e *traps* para que essa comunicação seja realizada.
c) Errado. Os *triggers* usam informações que chegam para serem ativados, mas eles não são capazes de recuperar as informações.
d) Certo. É a instalação de agentes que fará essa comunicação ativa, fornecendo as informações necessárias de forma automática.
e) Errado. Apesar dos testes automáticas servirem para verificar o estado dos dispositivos, é necessária a informação que fornecerá subsídio aos testes - e isso é garantido pelos agentes.

Portanto, a alternativa correta é a letra D.

Gabarito: Letra D

---

11. (SELECON/CM SÃO GONÇALO/2022) Na atividade de monitoramento de infraestrutura, o Zabbix traz uma série de definições que são necessárias para a sua operação. Dentre elas, quatro são fundamentais:

> I. É uma entidade lógica que agrupa itens de interesse para o monitoramento. Exemplos são um servidor, um roteador ou um switch.
> II. É um indicador que será monitorado, como a taxa de uso de CPU, o espaço em disco ou o tempo de resposta de um serviço.
> III. E um recurso a partir do qual alguma notificação será gerada automaticamente. E criado no caso da necessidade de notificações sobre a ocorrência de eventos específicos, tais como espaço em disco de um servidor ou consumo de banda de um link.
> IV. É uma atividade que será executada quando o recurso em III for acionado. São exemplos o envio de e-mail, SMS ou a execução de scripts.

Os termos definidos em I, II, III e IV são, respectivamente:

a) item, trigger, ação e hosts
b) trigger, ação, hosts e item
c) ação, hosts, item e trigger
d) hosts, item, trigger e ação

Comentários:

Mesmo sendo de uma banca menor, essa é uma excelente questão. Vamos associar cada item aos componentes do Zabbix.

> I. A entidade lógica que agrupa os itens de monitoramento é o próprio dispositivo sendo avaliado - denominado de host.
> II. O item se refere ao item, que especifica um indicador que será monitorado pelo Zabbix.
> III. Essa é a definição de um trigger.
> IV. A atividade executada após o trigger é uma ação.

Portanto, ficamos com host - item - trigger e ação. A alternativa correta é a letra D.

Gabarito: Letra D

---

12. (FGV/TRT 16/2022) A ferramenta Zabbix 6 é amplamente usada para monitoramento de redes; sua instalação e configuração requer pacotes de terceiros.

Com relação aos seus requisitos de instalação, analise os itens a seguir.

I. O Zabbix é construído em torno de servidores web, servidores de banco de dados e da linguagem de script Python.
II. Os servidores de bancos de dados suportados, por padrão, são MySQL, PostreSQL ou MS SQL Server.
III. O frontend do Zabbix é suportado, por padrão, por servidores Web Apache.

Está correto o que se afirma em

a) I, II e III.
b) I apenas.
c) II apenas.
d) III apenas.
e) II e III, apenas.

Comentários:

Vamos analisar cada item.

I. Errado. De fato, temos os servidores web (tanto o Zabbix Server quanto o Zabbix Web), e um banco de dados. Mas a linguagem utilizada no Zabbix é o PHP para a interface web, e o JavaScript para execução de scripts.
II. Errado. Não há integração com o MS SQL Server.
III. Certo. Como o frontend é feito em PHP, ele necessita de um servidor que o suporte - que é o caso do Apache.

Portanto, apenas o item III é correto.

Gabarito: Letra D

---

13. (FGV/TRT 13/2022) Analise as afirmativas a seguir sobre a ferramenta de monitoramento Zabbix:

I. Descobre automaticamente dispositivos na rede por meio da varredura de um range de IP definidos.
II. Coleta dados com intervalos personalizados, inclusive com agendamento exato do momento da coleta.
III. Recebe por meio de API RESTful as traps SNMP dos dispositivos monitorados.

Está correto o que se afirma em

a) I, apenas.
b) II, apenas.
c) III, apenas.
d) I e II,apenas.
e) I, II e III.

Comentários:

Vamos analisar cada item.

I. Certo. Através do LLD (Low Level Discovery) é possível a descoberta automática de dispositivos na rede.
II. Certo. O Zabbix fornece a possibilidade de diversas abordagens na coleta de dados, inclusive com agendamento.
III. Errado. A comunicação das *traps* é feita de forma direta, do agente para o gerente, sem intermédio de uma API. No Zabbix, as comunicações agente-gerente não passam por APIs, são feitas de forma direta através dos protocolos.

Portanto, corretos os itens I e II.

Gabarito: Letra D

---

14. (CEBRASPE/TRT 8/2022) Assinale a opção que apresenta ferramenta utilizada para monitorar a infraestrutura de TI, como redes, servidores, máquinas virtuais e serviços em nuvem.

a) Zabbix
b) Kibana
c) Grafana
d) Elasticsearch
e) Prometheus

Comentários:

Vamos ver qual item traz uma ferramenta de monitoramento de infraestrutura de TI.

a) Certo. O Zabbix é uma ferramenta de monitoramento, que utiliza protocolos nativos e integração nativa com o SNMP.
b) Errado. Kibana é uma ferramenta de visualização usada na stack ELK.
c) Errado. Grafana também é uma ferramenta de visualizações, com múltiplas integrações.
d) Errado. Elasticsearch é uma ferramenta de armazenamento e pesquisa de dados.

e) Errado. Prometheus é um banco de dados de séries temporais.

Portanto, correta a letra A.

Gabarito: Letra A

---

15. (VUNESP/PREF. F.CO MORATO/2022) No contexto do software de monitoramento Zabbix, o conceito de checagem passiva é caracterizado pelo seguinte processo:

a) O sistema operacional do dispositivo no qual o agente Zabbix está instalado envia os dados monitorados diretamente para o servidor Zabbix, sem passar pelo agente.
b) O usuário do dispositivo no qual o agente Zabbix está instalado precisa confirmar, por meio de uma interface de usuário, o envio de cada dado monitorado ao servidor Zabbix.
c) Primeiro, o agente Zabbix obtém uma lista de itens para monitoramento do servidor Zabbix. Periodicamente, os dados monitorados são enviados do agente ao servidor sem que este requisite.
d) O agente Zabbix, em execução no dispositivo monitorado, responde a requisições de dados do servidor Zabbix.
- e) O agente Zabbix salva localmente um histórico de dados monitorados, sem enviá-los ao servidor Zabbix.

Comentários:

A checagem, no Zabbix, é o processo de trazer informações dos dispositivos monitorados para o servidor. Essa checagem pode ocorrer de forma ativa, quando os agentes enviam os dados para o servidor, ou passiva, quando o próprio servidor recupera os dados. Com isso em mente, a alternativa que corretamente descreve a checagem passiva é a letra D. Um ponto interessante, na alternativa C temos a explicação da abordagem ativo com o auto registro.

Gabarito: Letra D

---

16. (QUADRIX/CFO/2022) Com relação à rede virtual privada (VPN), às ferramentas de monitoração e gerenciamento de rede e à segurança de rede, julgue o item a seguir.

Dentro do contexto dos serviços de rede, a arquitetura do Zabbix organiza-se em um modelo denominado three-tier, o qual é composto pelas seguintes camadas: a aplicação; a rede em si; e a interface web.

Comentários:

O modelo *three-tier* está associado à disposição em camadas, o que não ocorre no Zabbix. O Zabbix é baseado em componentes autônomos e complementares - o Zabbix Server, o Zabbis Web Interface e um banco de dados. Portanto, incorreta a afirmativa.

Gabarito: Errado

---

17. (QUADRIX/CFO/2022) Com relação à rede virtual privada (VPN), às ferramentas de monitoração e gerenciamento de rede e à segurança de rede, julgue o item a seguir.

Por meio da plataforma Zabbix, é possível ter um servidor central de monitoramento e vários outros servidores subordinados a ele enviando as métricas para este servidor.

Comentários:

Perfeito! O Zabbix funciona no esquema de computação distribuída, onde podemos instalar o servidor em um nó central, que fará o monitoramento, e os diversos agentes nos outros dispositivos monitorados. Além disso, podemos acessar esse monitoramento de forma remota, a partir do Zabbix Web.

Gabarito: Correto

---

18. (QUADRIX/CRF PR/2022) Com relação à arquitetura do Zabbix, o elemento que é considerado opcional, uma vez que o Zabbix Server não depende dele para funcionar, e é um agregador de dados que faz a coleta dos clientes na rede remota em nome do Zabbix Server denomina-se

a) trigger.
b) evento.
c) template.
d) Zabbix Proxy.
e) Zabbix Agent.

Comentários:

O agregador de dados, que atua num ponto intermediário entre os dispositivos monitorados e o servidor, reduzindo sua carga de trabalho e criando uma forma distribuída de captura de eventos, é chamado de Zabbix Proxy.

Gabarito: Letra D

---

19. (VUNESP/UNICAMP/2022) Em uma rede monitorada pela Solução de Monitoramento de Redes Zabbix, o Zabbix Proxy permite que

a) sejam replicados os dados do Zabbix Server para outro Zabbix Server.
b) a interface web de configuração do Zabbix Server possa ser acessada por meio de um HTTP Proxy.
c) o banco de dados do Zabbix Server possa ser compartilhado com outras instâncias do Zabbix Server.
d) sejam coletados dados de hosts localizados em redes distintas por meio de um único ponto de acesso.
e) sejam enviados dados para serviços externos, como os de gestão de incidentes para produção de alertas.

Comentários:

Vamos analisar cada alternativa.

a) Errado. Esse é um processo de replicação, que não faz parte do proxy.
b) Errado. A alternativa descreve o Zabbix Web.
c) Errado. A alternativa descreve o banco de dados, utilizado pelo Zabbix.
d) Certo. O Proxy pode se conectar a múltiplos dispositivos, armazenando seus dados em um *buffer* local, até que sejam carregados no servidor.
e) Errado. A alternativa descreve o conceito de ações.

Portanto, correta a letra D.

Gabarito: Letra D

---

20. (VUNESP/UNICAMP/2022) O componente da solução de monitoramento Zabbix que permite a coleta de métricas específicas do Sistema Operacional da máquina monitorada, como CPU e memória, é o:

a) Zabbix Server
b) Zabbix Agent
c) Zabbix Proxy
d) Zabbix Metric
e) Zabbix Client

Comentários:

A ferramenta que faz coleta de métricas locais do dispositivo, como CPU, memória e outros, é o Zabbix Agent. Ele específica as métricas coletadas a partir dos itens, e as envia para o servidor, para futuras análises.

Gabarito: Letra B

---

21. (VUNESP/UNICAMP/2022) Na solução de monitoramento Zabbix, o nome da chave que deve ser consultada para auferir a disponibilidade do host é

a) agent.ping
b) agent.test
c) agent.check
d) agent.alive
e) agent.run

Comentários:

O Zabbix utiliza protocolos padrões, da pilha TCP/IP, para fazer verificações - então, são usados protocolos como o SNMP, TCP, ICMP, e outros. Para fazer a verificação de disponibilidade de um *host*, usa-se o comando **`ping`**, associado a um agente.

Gabarito: Letra A

---

22. (FCC/ISS MANAUS/2019) Após a instalação do sistema de monitoração Zabbix 3.0 para a autenticação no servidor, deve-se utilizar o usuário e senha padrão, respectivamente,

a) Root e root.
b) Admin e zabbix.
c) Superuser e admin.
d) Admin e admin.
e) Zabbix e zabbix.

Comentários:

A FCC tem uma tendência a cobrar informações extremamente específicas - e essa é uma das cobranças. Apesar da questão perguntar para o Zabbix 3, no Zabbix 6 ainda são os mesmos valores: Admin, para usuário, e zabbix, para senha. Esse é o conjunto de usuário e senha usado no primeiro acesso à interface web.

Gabarito: Letra B

---

23. (INSTITUTO VERBANA/UFG/2019) Os softwares de monitoramento e gerenciamento de redes e servidores são ferramentas importantes para o controle eficiente dos eventos que ocorrem no hardware da rede e também nas aplicações. O software de monitoramento de código aberto, com suporte a monitoração distribuída, configuração por meio de interface gráfica e que utiliza expressões lógicas denominadas triggers para verificar o estado de funcionamento de itens monitorados, é o

a) NAGIOS.
b) Zabbix.
c) NTOP.
d) SNMP.

Comentários:

Das alternativas apresentadas, a única que informa uma ferramenta de monitoramento de código aberto é a letra B - Zabbix. Quanto aos demais:

- NAGIOS: Também é uma ferramenta de monitoramento, mas não é open source.
- NTOP: Ferramenta de análise de tráfego de rede, que também não é open source.
- SNMP: Protocolo para gerenciamento de redes.

Gabarito: Letra B

---

24. (CONSULPLAN/CM BH/2018) O Zabbix é uma ferramenta utilizada para monitorar a disponibilidade e o desempenho de aplicações, ativos e serviços de rede. É moderna, Open Source, multiplataforma, sem custos de licenciamento, e sua licença é a GPLv2 (General Public License). Na comunicação entre o agente e o servidor Zabbix algumas portas podem ser utilizadas, sendo que são portas-padrão, mas que podem ser alteradas, caso seja necessário. Uma dessas portas é utilizada quando se está monitorando servidores de aplicações Java por meio do componente Java Gateway. Assinale a alternativa que apresenta tal porta com seu respectivo protocolo da camada de transporte.

a) Porta 10050 / Protocolo TCP.
b) Porta 10052 / Protocolo TCP.
c) Porta 10050 / Protocolo UDP.
d) Porta 10052 / Protocolo UDP.

Comentários:

Questão de altíssimo nível, dada a sua especificidade. Usualmente, o Zabbix opera nas portas 10050, para checagens e comunicações com o servidor, e 10051 para comunicações com os

agentes ativos. Porém, a questão quer uma ferramenta específica - o *gateway* de integração com ferramentas Java. Esse *gateway* opera numa porta específica - 10052/TCP.

Gabarito: Letra B

---

25. (COMPERVE/UFRN/2018) O Zabbix é um software que permite o monitoramento de vários parâmetros de uma rede de computadores. Ele possui um mecanismo flexível que permite que usuários configurem alertas baseados em e-mail para qualquer evento virtual possibilitando, assim, o acompanhamento da saúde e integridade de servidores. Uma das características do Zabbix é

a) o uso do protocolo de autorização OAuth para controle de acesso de sua API.
b) a criação automática de redes de computadores privadas.
c) o armazenamento de dados históricos em bancos de dados não relacionais.
d) a descoberta automática de dispositivos de rede de computadores.

Comentários:
Vamos analisar cada item.

a) Errado. O Zabbix usa um protocolo próprio de autenticação, baseado no SSO (Single Sign-On).
b) Errado. O Zabbix não cria redes.
c) Errado. Os bancos de dados em que os dados são armazenados são relacionais.
d) Certo. A descoberta automática é permitida no Zabbix, implementada principalmente através do Low Level Discovery (LLD).

Portanto, correta a letra D.

Gabarito: Letra D

---

26. (FCC/TRT/ 24/2017) Um Analista, ao consultar a documentação do Zabbix Appliance (3.0.0) para o sistema operacional Linux (Ubuntu 14.04.3), obteve as informações abaixo.

O Zabbix Appliance utiliza-se do IPTables com as seguintes regras configuradas:

– Portas abertas
– SSH (22 TCP)
– Zabbixagent (10050 TCP) e Zabbixtrapper (10051 TCP)
– HTTP (80 TCP) e HTTPS (443 TCP)
– SNMP trap (162 UDP)
– Consultas NTP liberadas (53 UDP)
– Pacotes ICMP limitados a 5 por segundo
– Qualquer situação diferente sendo bloqueada

Considerando os fundamentos de redes de computadores e as informações acima é correto afirmar:

a) O Zabbix é uma ferramenta de monitoramento de redes, servidores e serviços que não permite monitoramento agentless (sem agentes).
b) Como o servidor Zabbix é obrigatoriamente instalado em sistemas Unix ou Linux, não há agentes Zabbix disponíveis para ambientes Windows e OS.
c) O ICMP, assim como o TCP e o UDP, é um protocolo de controle também usado para a transmissão de dados, que desempenha diversas funções exclusivas para Linux como o ping, para verificar se uma determinada máquina está online.
d) No SNMP o item a ser monitorado ou gerenciado é um agente. Quem consulta (GET) ou solicita modificações (SET) é um gerente. O agente também tem a função de gerar alertas (TRAP).
e) Dentre as regras do IPTables para configurar o firewall pode-se utilizar o parâmetro "-s ALL", que se aplica simultaneamente aos três protocolos (SSH, HTTP e HTTPS), sem que seja necessário incluir uma regra separada para cada um.

Comentários:

Questão multidisciplinar, ótima. Vamos analisar cada item.

a) Errado. O Zabbix suporta uma abordagem chamada de *monitoramento simples*. Esse monitoramento funciona sem o uso de agentes.
b) Errado. Temos múltiplos sistemas suportados pelos agentes Zabbix.
c) Errado. O ICMP tem uma abordagem completamente diferente do TCP e UDP. Enquanto estes são protocolos para transporte, aquele é um protocolo destinado a relatórios de erros em redes.
d) Certo. Perfeita a descrição do SNMP. Detalhe para as *traps*, que são uma forma de comunicação assíncrona e ativa dos agentes SNMP.
e) Errado. No IPTables, precisamos incluir regras para cada um dos protocolos separadamente.

Sendo assim, ficamos com a letra D como nosso gabarito.

Gabarito: Letra D

---

27. (FEPESE/CIASC/2017) Analise as afirmativas abaixo com relação ao software Zabbix.

1. O Zabbix Get é um utilitário de linha de comando que pode ser utilizado para enviar dados para o Zabbix Server.
2. O Zabbix suporta tanto “pooling” quanto “trapping”.

3. A API Zabbix permite que você utilize o protocolo RPC Json para criar, atualizar e receber objetos Zabbix (como hosts, itens, gráficos, dentre outros) ou executar qualquer tarefa personalizada.
4. O Zabbix Get é um utilitário de linha de comando que pode ser utilizado para se comunicar com o agente de monitoração do Zabbix e requisitar um dado do agente.

Assinale a alternativa que indica todas as afirmativas corretas.

a) São corretas apenas as afirmativas 1 e 2.
b) São corretas apenas as afirmativas 3 e 4.
c) São corretas apenas as afirmativas 1, 2 e 3.
d) São corretas apenas as afirmativas 2, 3 e 4.
e) São corretas as afirmativas 1, 2, 3 e 4.

Comentários:

Vamos analisar cada um dos itens.

1. Errado. O utilitário que é utilizado para enviar dados de um host para o Zabbix Server é o Sender, não o Get.
2. Certo. O Zabbix atua em ambas as abordagens.
3. Certo. A API Zabbix é a ferramenta utilizada para integrar o programa com outras ferramentas de terceira parte - uma dessas ferramentas é a integração com chamadas remotas de procedimentos (RPC), permitindo que, remotamente, se criem objetos dentro do Zabbix.
4. Certo. O Get é o comando executado pelo Server para recuperar dados dos diferentes hosts, usualmente quando há problemas nas conexões. Ele faz uma requisição de determinada informação que deve ser respondida pelo host.

Os itens corretos são, portanto, 2 , 3 e 4.

Gabarito: Letra D

---

28. (AOCP/UFBA/2017) Considerando os conhecimentos sobre redes de computadores, julgue, como VERDADEIRO ou FALSO, o item a seguir.

Ferramentas de monitoração e gerenciamento de redes, como o Zabbix e o MRTG, fazem uso do protocolo SNMP para implementar suas funcionalidades.

Comentários:

Apesar de contarem com protocolos nativos, tanto o Zabbix quanto o MRTG dão suporte ao uso de SNMP como protocolo nativo do seu ecossistema.

Gabarito: Correto

---

29. (CEBRASPE/MEC/2015) Julgue o item seguinte, a respeito de sistemas de gerenciamento de rede, monitoramento e diagnóstico de ambientes computacionais.

Zabbix é uma solução integrada que provê diversos recursos de monitoração em um único pacote, por meio de verificações de disponibilidade e desempenho utilizando tanto trapping quanto polling do SNMP.

Comentários:

Perfeito. O Zabbix utiliza tanto o *trapping*, que acontece quando os dispositivos mandam os dados para o servidor, quanto o *pooling*, que é quando o servidor recupera os dados diretamente dos dispositivos.

Gabarito: Correto

---

30. (FUNCERN/IF RN/2013) nalise as proposições seguintes no que se refere às funcionalidades da ferramenta de monitoramento Zabbix versão 2.2.

I. Utiliza o protocolo SMTP para coletar dados das MIBs dos equipamentos.
II. Utiliza um agente próprio para coletar dados de servidores e serviços.
III. Pode automaticamente descobrir dispositivos de rede.
IV. Oferece uma ferramenta desktop para monitoramento, com geração de relatórios e gráficos.

Marque a opção em que estão enumeradas apenas proposições corretas.

a) I e IV.
b) II e IV.
c) II e III.
d) I e III.

Comentários:

Apesar de ser uma questão antiga, e direcionada ao Zabbix 2.2, todos os itens continuam atuais. Vamos analisá-los.

I. Errado. Cuidado! SMTP é um protocolo de correio eletrônico. O protocolo para coletar dado das MIBs é o SNMP - e de fato, ele é usado pelo Zabbix.

II. Certo. Há a possibilidade de usa de um agente próprio, denominado Agente Zabbix.
III. Certo. A descoberta automática é uma possibilidade no Zabbix, principalmente no LLD (Low Level Discovery).
IV. Errado. A ferramenta para monitoramento não é baseada no *desktop*. Ela é uma interface web.

Portanto, corretos os itens II e III.

Gabarito: Letra C

---

31. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta de monitoramento Zabbix, julgue o item abaixo.

Os itens no Zabbix são componentes que definem ações e tipos de dados a serem coletados. Dentre os itens, destaca-se as traps SNMP, que fazem uma comunicação ativa e assíncrona com o servidor Zabbix através do protocolo SNMP.

Comentários:

Perfeito! Uma trap SNMP é uma mensagem que não segue o fluxo regular de mensagens, pois ela parte de uma iniciativa dos hosts - caracterizando uma comunicação ativa. Além disso, essas traps ficam armazenadas no MIB, um repositório local, permitindo uma comunicação assíncrona com o servidor.

Gabarito: Correto

---

32. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Zabbix, julgue o item abaixo.

Ao ser disparado e modificar o estado de um item para unkown, o trigger gera um evento de trigger.

Comentários:

Muito cuidado! Somente a mudança OK → PROBLEM e PROBLEM → OK disparam eventos de trigger. Quando a mudança é para unknown, temos um evento interno.

Gabarito: Errado

---

33. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Zabbix, julgue o item abaixo.

A Descoberta de Baixo Nível (LLD - Low Level Discovery) é uma ferramenta utilizada para implementar um processo de descoberta automático no ecossistema do Zabbix, encontrando serviços e hosts automaticamente, sem necessidade de configurações manuais.

Comentários:

O item está correto. O LLD introduz a automação no processo de descoberta, facilitando o monitoramento por parte do Zabbix, evitando erros de inserções manuais.

Gabarito: Correto

LISTA DE QUESTÕES

Zabbix

01. (FGV/ALEP/2024) Um gerente de sistemas instalou o software de gerenciamento de rede Zabbix, em uma rede Linux. Após a instalação ele deseja saber se o serviço do Zabbix entrou em execução.

Para esse caso, ele deve utilizar o comando

a) ls –l | grep zabbix
b) find –iname zabbix
c) ps aux | grep zabbix
d) ls -ARC | grep zabbix
e) tasklist /PID | grep zabbix

02. (CONSULPLAN/CM BH/2024) Zabbix é uma plataforma de monitoramento de rede projetada para monitorar a disponibilidade e o desempenho de dispositivos de rede, servidores e recursos de TI. Sobre os componentes da arquitetura do Zabbix, assinale, a seguir, a alternativa que é responsável por coletar dados de monitoramento.

a) Zabbix Proxy.
b) Zabbix Agent.
c) Zabbix Server.
d) Zabbix Web Interface.

03. (CONSULPLAN/CM BH/2024) O Zabbix oferece suporte ao monitoramento de uma variedade de dispositivos e pode monitorar sistemas operacionais como Linux, Windows e MacOS. Considerando o funcionamento do Zabbix, como o Zabbix trata os eventos de disparo que ocorrem em um ambiente monitorado?

a) Aciona automaticamente a resolução do problema.
b) Gera um alerta e aguarda a intervenção do usuário.
c) Armazena os eventos em um banco de dados para análise posterior.
d) Ignora os eventos até que o administrador os verifique manualmente.

04. (CONSULPLAN/CM BH/2024) A interface gráfica do Zabbix oferece uma maneira fácil de configurar hosts, itens, triggers, visualizar gráficos e relatórios, dentre outras funcionalidades. Assinale, a seguir, a interface gráfica usada para configurar e monitorar o Zabbix.

a) Zabbix Console.

b) Zabbix Dashboard.
c) Zabbix Control Center.
d) Zabbix Web Interface.

05. (FCC/TRT 18/2023) Na arquitetura do Zabbix, o proxy Zabbix

a) realiza a autenticação dos usuários do Zabbix e da rede e dos seus componentes.
b) realiza o monitoramento dos recursos dos servidores e aplicações, enviando dados obtidos para o servidor.
c) é um componente opcional e pode proporcionar a redistribuição de carga de coletas.
d) permite a interação entre o servidor Zabbix e o administrador e, normalmente, é instalado na máquina do servidor.
e) gerencia o armazenamento de dados recebidos pelo Zabbix.

06. (VUNESP/UNICAMP/2023) Na arquitetura da ferramenta Zabbix, é papel do Zabbix proxy

a) estabelecer uma VPN entre o Zabbix server e cada dispositivo monitorado.
b) funcionar como servidor web para visualização remota de dados monitorados.
c) monitorar ativamente recursos locais do dispositivo onde está instalado e enviar dados ao Zabbix server.
d) receber dados de um ou mais dispositivos monitorados e enviar ao Zabbix server.
e) servir como banco de dados a ser utilizado pelo Zabbix server.

07. (AOCP/IF MA/2023) No contexto do monitoramento de aplicações com Zabbix, qual é a característica que ajuda a coletar informações detalhadas sobre o desempenho e a saúde das aplicações?

a) Zabbix suporta apenas o monitoramento de aplicações desenvolvidas em linguagens específicas.
b) Zabbix utiliza exclusivamente notificações por e-mail para alertar sobre problemas de desempenho.
c) Zabbix requer que os desenvolvedores insiram manualmente cada métrica de desempenho no código-fonte.
d) Zabbix permite a criação de templates personalizados para facilitar a configuração de itens de monitoramento.
e) Zabbix envia relatórios de monitoramento apenas para desenvolvedores designados, ignorando a equipe de operações.

08. (FUNDATEC/PROCERGS/2023) Zabbix é uma ferramenta de monitoramento de rede de computadores capaz de monitorar uma infraestrutura completa. Possui licença GPL de uso gratuito e dispõe de uma grande variedade de opções. Em relação às principais características do Zabbix, analise as assertivas abaixo:

I. Tem suporte a uma grande variedade de sistemas operacionais como Linux, Windows, Solaris, OpenBSD, Mac OS X, entre outros.
II. Tem integração com banco de dados (MySQL, Oracle, PostgreSQL ou SQLite) e geração de gráficos em tempo real.
III. Realiza a monitoria de serviços simples, como HTTP, POP3, IMAP, SSH, entre outros e possui suporte nativo ao protocolo SNMP.

Quais estão corretas?

a) Apenas II.
b) Apenas III.
c) Apenas I e II.
d) Apenas II e III.
e) I, II e III.

09. (FURB/SAMAE JS/2023) Existe um protocolo de rede que pode ser utilizado para gerenciamento e monitoramento de redes ou dispositivos compatíveis conectados à rede. Por exemplo, você pode monitorar de forma remota uma bomba de uma estação de tratamento de água para coletar as informações de operação dela por meio de uma ferramenta específica, como o Zabbix. As informações podem ser, então, utilizadas para gerar relatórios de desempenho ou alertas de funcionamento. Assinale a alternativa correta que representa o protocolo de rede que permite tal função:

a) SNMP
b) NTP
c) IMAP
d) NNTP
e) SMTP

10. (FGV/CGU/2022) Jane integra a equipe de TI de seu órgão da Administração Pública Federal (APF), que está assinando documentos com seus clientes nos quais os níveis de serviços (Service Level Agreement – SLA) estão sendo definidos para cobrança do serviço prestado.

Para obter os dados para os relatórios de SLA, Jane deve implementar a coleta automática de dados de disponibilidade e geração de SLA no Zabbix, realizando as seguintes ações:

a) configuração da correlação de eventos no monitoramento de arquivos de registros (log) dos serviços para identificar as suas indisponibilidades;
b) instalação do protocolo SNMP nos servidores e configuração da comunidade SLA para permitir coletas múltiplas dos dados de limites definidos para os SLA;

c) configuração das funções preditivas para ativação de triggers que possam restabelecer a disponibilidade de um serviço em caso de incidente, garantindo o SLA acordado;
d) instalação do agente Zabbix nos servidores e configuração do monitoramento dos serviços ofertados e de um serviço de TI com esses servidores;
e) ativação da monitoração Web com cenários de testes automáticos e rotineiros para os serviços hospedados, para validação de sua disponibilidade.

11. (SELECON/CM SÃO GONÇALO/2022) Na atividade de monitoramento de infraestrutura, o Zabbix traz uma série de definições que são necessárias para a sua operação. Dentre elas, quatro são fundamentais:

I. É uma entidade lógica que agrupa itens de interesse para o monitoramento. Exemplos são um servidor, um roteador ou um switch.
II. É um indicador que será monitorado, como a taxa de uso de CPU, o espaço em disco ou o tempo de resposta de um serviço.
III. E um recurso a partir do qual alguma notificação será gerada automaticamente. E criado no caso da necessidade de notificações sobre a ocorrência de eventos específicos, tais como espaço em disco de um servidor ou consumo de banda de um link.
IV. É uma atividade que será executada quando o recurso em III for acionado. São exemplos o envio de e-mail, SMS ou a execução de scripts.

Os termos definidos em I, II, III e IV são, respectivamente:

a) item, trigger, ação e hosts
b) trigger, ação, hosts e item
c) ação, hosts, item e trigger
d) hosts, item, trigger e ação

12. (FGV/TRT 16/2022) A ferramenta Zabbix 6 é amplamente usada para monitoramento de redes; sua instalação e configuração requer pacotes de terceiros.

Com relação aos seus requisitos de instalação, analise os itens a seguir.

I. O Zabbix é construído em torno de servidores web, servidores de banco de dados e da linguagem de script Python.
II. Os servidores de bancos de dados suportados, por padrão, são MySQL, PostreSQL ou MS SQL Server.
III. O frontend do Zabbix é suportado, por padrão, por servidores Web Apache.

Está correto o que se afirma em

a) I, II e III.

b) I apenas.
c) II apenas.
d) III apenas.
e) II e III, apenas.

13. (FGV/TRT 13/2022) Analise as afirmativas a seguir sobre a ferramenta de monitoramento Zabbix:

I. Descobre automaticamente dispositivos na rede por meio da varredura de um range de IP definidos.
II. Coleta dados com intervalos personalizados, inclusive com agendamento exato do momento da coleta.
III. Recebe por meio de API RESTful as traps SNMP dos dispositivos monitorados.

Está correto o que se afirma em

a) I, apenas.
b) II, apenas.
c) III, apenas.
d) I e II,apenas.
e) I, II e III.

14. (CEBRASPE/TRT 8/2022) Assinale a opção que apresenta ferramenta utilizada para monitorar a infraestrutura de TI, como redes, servidores, máquinas virtuais e serviços em nuvem.

a) Zabbix
b) Kibana
c) Grafana
d) Elasticsearch
e) Prometheus

15. (VUNESP/PREF. F.CO MORATO/2022) No contexto do software de monitoramento Zabbix, o conceito de checagem passiva é caracterizado pelo seguinte processo:

a) O sistema operacional do dispositivo no qual o agente Zabbix está instalado envia os dados monitorados diretamente para o servidor Zabbix, sem passar pelo agente.
b) O usuário do dispositivo no qual o agente Zabbix está instalado precisa confirmar, por meio de uma interface de usuário, o envio de cada dado monitorado ao servidor Zabbix.
c) Primeiro, o agente Zabbix obtém uma lista de itens para monitoramento do servidor Zabbix. Periodicamente, os dados monitorados são enviados do agente ao servidor sem que este requisite.

d) O agente Zabbix, em execução no dispositivo monitorado, responde a requisições de dados do servidor Zabbix.
e) O agente Zabbix salva localmente um histórico de dados monitorados, sem enviá-los ao servidor Zabbix.

16. (QUADRIX/CFO/2022) Com relação à rede virtual privada (VPN), às ferramentas de monitoração e gerenciamento de rede e à segurança de rede, julgue o item a seguir.

Dentro do contexto dos serviços de rede, a arquitetura do Zabbix organiza-se em um modelo denominado three-tier, o qual é composto pelas seguintes camadas: a aplicação; a rede em si; e a interface web.

17. (QUADRIX/CFO/2022) Com relação à rede virtual privada (VPN), às ferramentas de monitoração e gerenciamento de rede e à segurança de rede, julgue o item a seguir.

Por meio da plataforma Zabbix, é possível ter um servidor central de monitoramento e vários outros servidores subordinados a ele enviando as métricas para este servidor.

18. (QUADRIX/CRF PR/2022) Com relação à arquitetura do Zabbix, o elemento que é considerado opcional, uma vez que o Zabbix Server não depende dele para funcionar, e é um agregador de dados que faz a coleta dos clientes na rede remota em nome do Zabbix Server denomina-se

a) trigger.
b) evento.
c) template.
d) Zabbix Proxy.
e) Zabbix Agent.

19. (VUNESP/UNICAMP/2022) Em uma rede monitorada pela Solução de Monitoramento de Redes Zabbix, o Zabbix Proxy permite que

a) sejam replicados os dados do Zabbix Server para outro Zabbix Server.
b) a interface web de configuração do Zabbix Server possa ser acessada por meio de um HTTP Proxy.
c) o banco de dados do Zabbix Server possa ser compartilhado com outras instâncias do Zabbix Server.
d) sejam coletados dados de hosts localizados em redes distintas por meio de um único ponto de acesso.
e) sejam enviados dados para serviços externos, como os de gestão de incidentes para produção de alertas.

20. (VUNESP/UNICAMP/2022) O componente da solução de monitoramento Zabbix que permite a coleta de métricas específicas do Sistema Operacional da máquina monitorada, como CPU e memória, é o:

a) Zabbix Server
b) Zabbix Agent
c) Zabbix Proxy
d) Zabbix Metric
e) Zabbix Client

21. (VUNESP/UNICAMP/2022) Na solução de monitoramento Zabbix, o nome da chave que deve ser consultada para auferir a disponibilidade do host é

a) agent.ping
b) agent.test
c) agent.check
d) agent.alive
e) agent.run

22. (FCC/ISS MANAUS/2019) Após a instalação do sistema de monitoração Zabbix 3.0 para a autenticação no servidor, deve-se utilizar o usuário e senha padrão, respectivamente,

a) Root e root.
b) Admin e zabbix.
c) Superuser e admin.
d) Admin e admin.
e) Zabbix e zabbix.

23. (INSTITUTO VERBANA/UFG/2019) Os softwares de monitoramento e gerenciamento de redes e servidores são ferramentas importantes para o controle eficiente dos eventos que ocorrem no hardware da rede e também nas aplicações. O software de monitoramento de código aberto, com suporte a monitoração distribuída, configuração por meio de interface gráfica e que utiliza expressões lógicas denominadas triggers para verificar o estado de funcionamento de itens monitorados, é o

a) NAGIOS.
b) Zabbix.
c) NTOP.
d) SNMP.

24. (CONSULPLAN/CM BH/2018) O Zabbix é uma ferramenta utilizada para monitorar a disponibilidade e o desempenho de aplicações, ativos e serviços de rede. É moderna, Open

Source, multiplataforma, sem custos de licenciamento, e sua licença é a GPLv2 (General Public License). Na comunicação entre o agente e o servidor Zabbix algumas portas podem ser utilizadas, sendo que são portas-padrão, mas que podem ser alteradas, caso seja necessário. Uma dessas portas é utilizada quando se está monitorando servidores de aplicações Java por meio do componente Java Gateway. Assinale a alternativa que apresenta tal porta com seu respectivo protocolo da camada de transporte.

a) Porta 10050 / Protocolo TCP.
b) Porta 10052 / Protocolo TCP.
c) Porta 10050 / Protocolo UDP.
d) Porta 10052 / Protocolo UDP.

25. (COMPERVE/UFRN/2018) O Zabbix é um software que permite o monitoramento de vários parâmetros de uma rede de computadores. Ele possui um mecanismo flexível que permite que usuários configurem alertas baseados em e-mail para qualquer evento virtual possibilitando, assim, o acompanhamento da saúde e integridade de servidores. Uma das características do Zabbix é

a) o uso do protocolo de autorização OAuth para controle de acesso de sua API.
b) a criação automática de redes de computadores privadas.
c) o armazenamento de dados históricos em bancos de dados não relacionais.
d) a descoberta automática de dispositivos de rede de computadores.

26. (FCC/TRT/ 24/2017) Um Analista, ao consultar a documentação do Zabbix Appliance (3.0.0) para o sistema operacional Linux (Ubuntu 14.04.3), obteve as informações abaixo.

O Zabbix Appliance utiliza-se do IPTables com as seguintes regras configuradas:

– Portas abertas
– SSH (22 TCP)
– Zabbixagent (10050 TCP) e Zabbixtrapper (10051 TCP)
– HTTP (80 TCP) e HTTPS (443 TCP)
– SNMP trap (162 UDP)
– Consultas NTP liberadas (53 UDP)
– Pacotes ICMP limitados a 5 por segundo
– Qualquer situação diferente sendo bloqueada

Considerando os fundamentos de redes de computadores e as informações acima é correto afirmar:

a) O Zabbix é uma ferramenta de monitoramento de redes, servidores e serviços que não permite monitoramento agentless (sem agentes).

b) Como o servidor Zabbix é obrigatoriamente instalado em sistemas Unix ou Linux, não há agentes Zabbix disponíveis para ambientes Windows e OS.
c) O ICMP, assim como o TCP e o UDP, é um protocolo de controle também usado para a transmissão de dados, que desempenha diversas funções exclusivas para Linux como o ping, para verificar se uma determinada máquina está online.
d) No SNMP o item a ser monitorado ou gerenciado é um agente. Quem consulta (GET) ou solicita modificações (SET) é um gerente. O agente também tem a função de gerar alertas (TRAP).
e) Dentre as regras do IPTables para configurar o firewall pode-se utilizar o parâmetro "-s ALL", que se aplica simultaneamente aos três protocolos (SSH, HTTP e HTTPS), sem que seja necessário incluir uma regra separada para cada um.

27. (FEPESE/CIASC/2017) Analise as afirmativas abaixo com relação ao software Zabbix.

1. O Zabbix Get é um utilitário de linha de comando que pode ser utilizado para enviar dados para o Zabbix Server.
2. O Zabbix suporta tanto “pooling” quanto “trapping”.
3. A API Zabbix permite que você utilize o protocolo RPC Json para criar, atualizar e receber objetos Zabbix (como hosts, itens, gráficos, dentre outros) ou executar qualquer tarefa personalizada.
4. O Zabbix Get é um utilitário de linha de comando que pode ser utilizado para se comunicar com o agente de monitoração do Zabbix e requisitar um dado do agente.

Assinale a alternativa que indica todas as afirmativas corretas.

a) São corretas apenas as afirmativas 1 e 2.
b) São corretas apenas as afirmativas 3 e 4.
c) São corretas apenas as afirmativas 1, 2 e 3.
d) São corretas apenas as afirmativas 2, 3 e 4.
e) São corretas as afirmativas 1, 2, 3 e 4.

28. (AOCP/UFBA/2017) Considerando os conhecimentos sobre redes de computadores, julgue, como VERDADEIRO ou FALSO, o item a seguir.

Ferramentas de monitoração e gerenciamento de redes, como o Zabbix e o MRTG, fazem uso do protocolo SNMP para implementar suas funcionalidades.

29. (CEBRASPE/MEC/2015) Julgue o item seguinte, a respeito de sistemas de gerenciamento de rede, monitoramento e diagnóstico de ambientes computacionais.

Zabbix é uma solução integrada que provê diversos recursos de monitoração em um único pacote, por meio de verificações de disponibilidade e desempenho utilizando tanto trapping quanto polling do SNMP.

30. (FUNCERN/IF RN/2013) nalise as proposições seguintes no que se refere às funcionalidades da ferramenta de monitoramento Zabbix versão 2.2.

I. Utiliza o protocolo SMTP para coletar dados das MIBs dos equipamentos.
II. Utiliza um agente próprio para coletar dados de servidores e serviços.
III. Pode automaticamente descobrir dispositivos de rede.
IV. Oferece uma ferramenta desktop para monitoramento, com geração de relatórios e gráficos.

Marque a opção em que estão enumeradas apenas proposições corretas.

a) I e IV.
b) II e IV.
c) II e III.
d) I e III.

31. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta de monitoramento Zabbix, julgue o item abaixo.

Os itens no Zabbix são componentes que definem ações e tipos de dados a serem coletados. Dentre os itens, destaca-se as traps SNMP, que fazem uma comunicação ativa e assíncrona com o servidor Zabbix através do protocolo SNMP.

32. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Zabbix, julgue o item abaixo.

Ao ser disparado e modificar o estado de um item para unkown, o trigger gera um evento de trigger.

33. (Inédita/Prof. Felipe Mathias) Acerca dos conhecimentos sobre a ferramenta Zabbix, julgue o item abaixo.

A Descoberta de Baixo Nível (LLD - Low Level Discovery) é uma ferramenta utilizada para implementar um processo de descoberta automático no ecossistema do Zabbix, encontrando serviços e hosts automaticamente, sem necessidade de configurações manuais.

GABARITO

GABARITO

1. Letra C
2. Anulada
3. Letra A
4. Letra D
5. Letra C
6. Letra D
7. Letra D
8. Letra E
9. Letra A
10. Letra D
11. Letra D
12. Letra D
13. Letra D
14. Letra A
15. Letra D
16. Errado
17. Correto
18. Letra D
19. Letra D
20. Letra B
21. Letra A
22. Letra B
23. Letra B
24. Letra B
25. Letra D
26. Letra D
27. Letra D
28. Correto
29. Correto
30. Letra C
31. Correto
32. Errado
33. Correto

ESSA LEI TODO MUNDO CONHECE:
PIRATARIA É CRIME.

Mas é sempre bom revisar o porquê e como você pode ser prejudicado com essa prática.

1. 

Professor investe seu tempo para elaborar os cursos e o site os coloca à venda.

2. 

Pirata divulga ilicitamente (grupos de rateio), utilizando-se do anonimato, nomes falsos ou laranjas (geralmente o pirata se anuncia como formador de "grupos solidários" de rateio que não visam lucro).

3. 

Pirata cria alunos fake praticando falsidade ideológica, comprando cursos do site em nome de pessoas aleatórias (usando nome, CPF, endereço e telefone de terceiros sem autorização).

4. 

Pirata compra, muitas vezes, clonando cartões de crédito (por vezes o sistema anti-fraude não consegue identificar o golpe a tempo).

5. 

Pirata fere os Termos de Uso, adultera as aulas e retira a identificação dos arquivos PDF (justamente porque a atividade é ilegal e ele não quer que seus fakes sejam identificados).

6. 

Pirata revende as aulas protegidas por direitos autorais, praticando concorrência desleal e em flagrante desrespeito à Lei de Direitos Autorais (Lei 9.610/98).

7. 

Concurseiro(a) desinformado participa de rateio, achando que nada disso está acontecendo e esperando se tornar servidor público para exigir o cumprimento das leis.

8. 

O professor que elaborou o curso não ganha nada, o site não recebe nada, e a pessoa que praticou todos os ilícitos anteriores (pirata) fica com o lucro.

Deixando de lado esse mar de sujeira, aproveitamos para agradecer a todos que adquirem os cursos honestamente e permitem que o site continue existindo.