

**ROTEIRO PARA MONITORAMENTO**

**Nomes:**

Davi Abade

Gustavo Lima

João Pedro

Maxuell Chaves

Nicoly Zillig Dias

São Paulo

2022

**SUMÁRIO**

[1 Introdução 3](#_Toc103937654)

[2 Requisitos de ambiente 4](#_Toc103937655)

[3 Configurações 4](#_Toc103937656)

[3.1 Configurações dos scripts 7](#_Toc103937657)

[4 Scripts Checklist 8](#_Toc103937658)

[4.1 VERIFICAÇÕES GENÉRICAS NA CONSOLE AWS 8](#_Toc103937659)

[4.2 IDENTIFICAÇÃO DO SERVIDOR 8](#_Toc103937660)

[4.3 VERIFICAÇÕES SSH 8](#_Toc103937661)

[5 Userparameter 9](#_Toc103937662)

[5.1 Arquivo Userparameter do Checklist 10](#_Toc103937663)

[6 Zabbix Trapper 11](#_Toc103937664)

[7 Criação de items 13](#_Toc103937665)

[8 Criação de trigger 15](#_Toc103937666)

[8.1 Regular expressions com triggers 15](#_Toc103937667)

[9 Conclusão 19](#_Toc103937668)

# Introdução

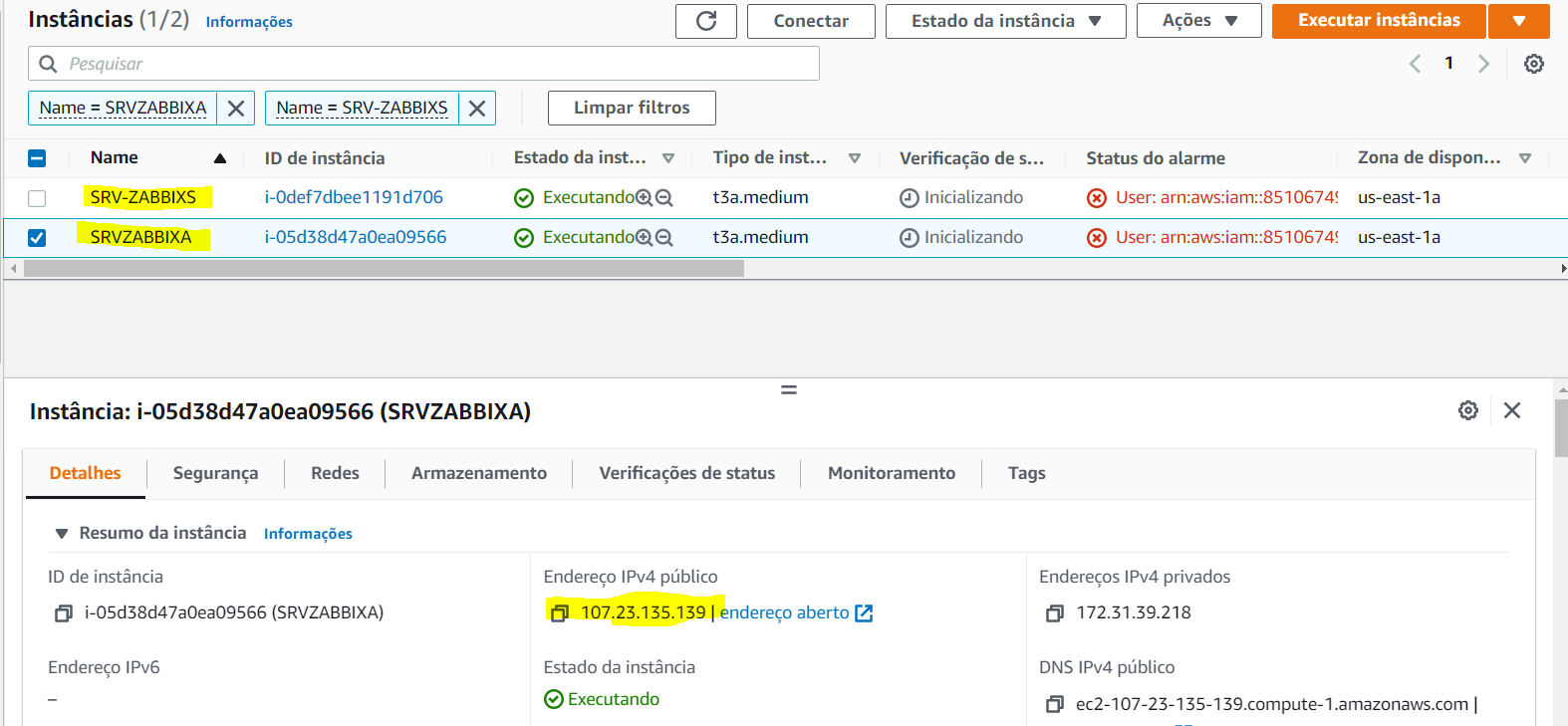
Este documento tem como objetivo explicar como é feito o monitoramento da AWS via Zabbix, demonstrando passo a passo as configurações necessárias e ferramentas que são utilizadas.

# Requisitos de ambiente

* Em primeiro lugar, é necessário instalar o [AWS CLI](https://awscli.amazonaws.com/AWSCLIV2.msi).
* Possuir acesso a permissões do [IAM](https://aws.amazon.com/pt/iam/features/manage-permissions/) e [AWS Auto Scaling](https://aws.amazon.com/pt/autoscaling/).
* Um servidor e agente [ZABBIX](https://www.zabbix.com/" \t "_blank).

# Configurações

* Rodar instância na AWS

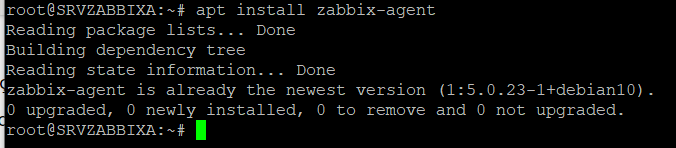


* Instalar os pacotes de serviço do Zabbix via CLI.

*Wget https://repo.zabbix.com/zabbix/5.0/debian/pool/main/z/zabbix-release/zabbix-release\_5.0-1+buster\_all.deb*

* Instalar Zabbix Agent

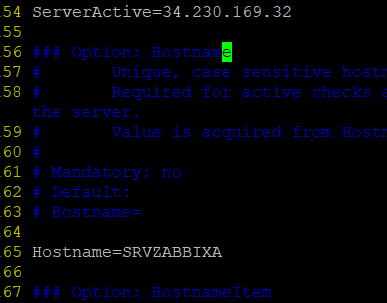
*apt install zabbix-agent*



* Configurar Hostname, Ip do Server e Server ativo do Zabbix no arquivo zabbix\_agentd.conf

*vim /etc/zabbix/zabbix\_agentd.conf*

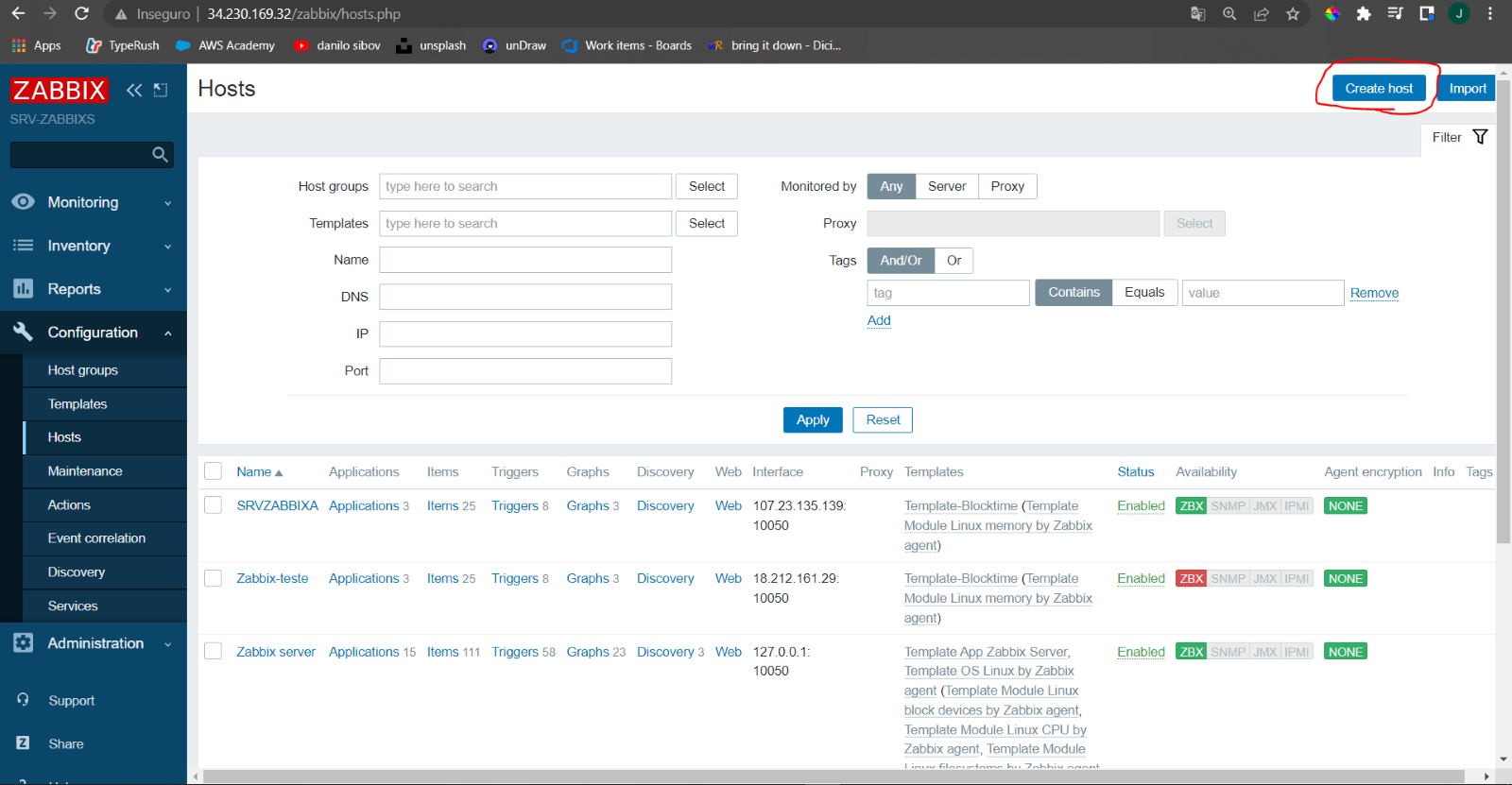
C:\Users\43017102807\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\2CF8D390.tmp

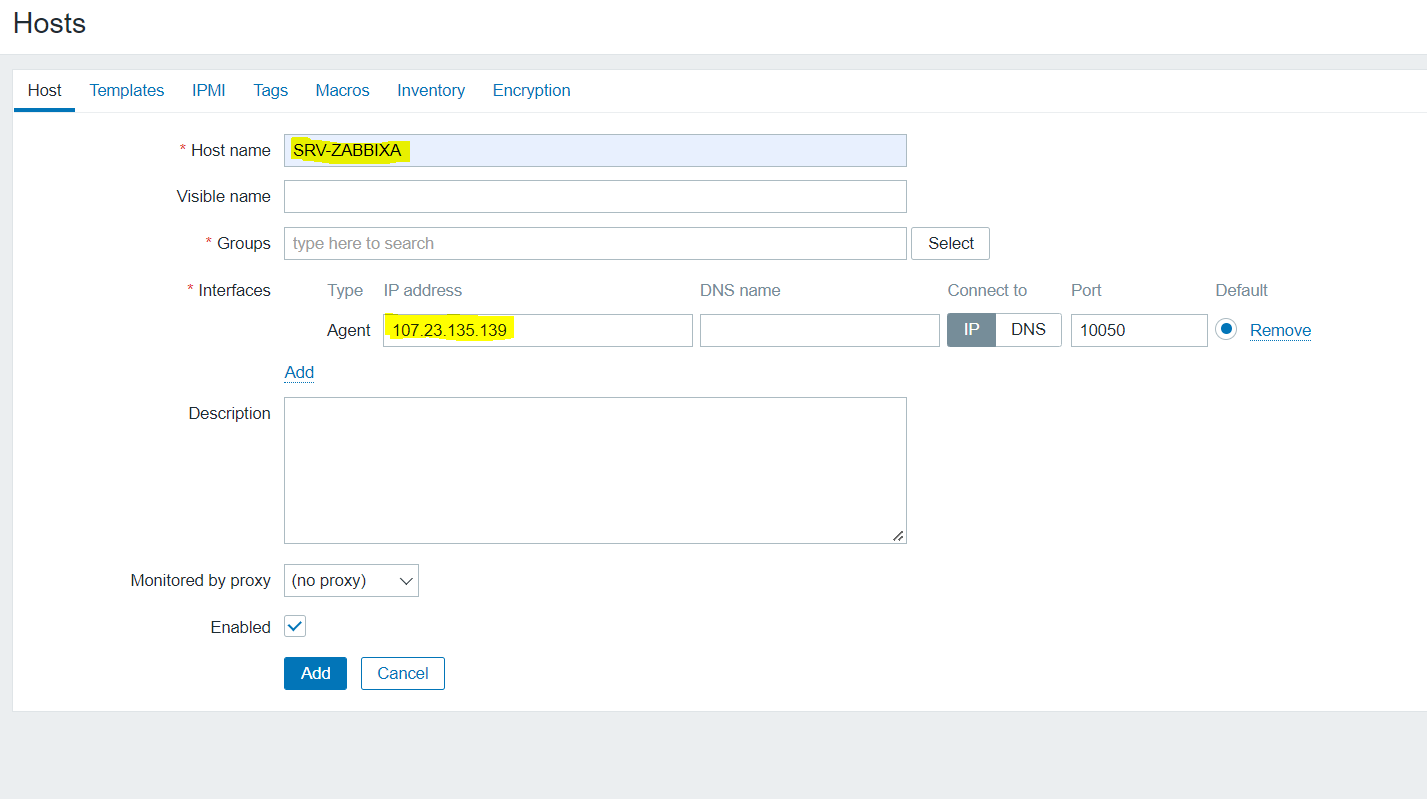


Obs: Após configuração do arquivo de configuração do zabbix-agent reinicie o serviço.

*Systemctl restart zabbix-agent*

* Criar uma nova Host no painel do Zabbix passando o Ip publico da AWS





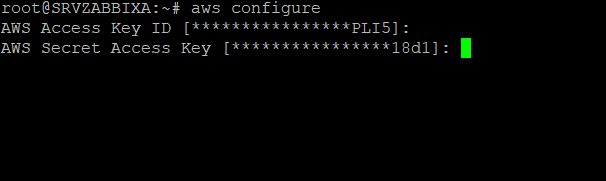
* Reiniciar o serviço

*service zabbix-agent restart*

* Instalar os pacotes do [AWS CLI](https://docs.aws.amazon.com/pt_br/cli/latest/userguide/getting-started-install.html)

Informar aws\_access\_key\_id e aws\_secret\_access\_key do IAM no AWS CLI

*aws configure*



* Liberar os scripts remotos no arquivo zabbix\_agentd.conf

*vim /etc/zabbix/zabbix\_agentd.conf*

EnableRemoteCommands=1

LogRemoteCommands=1

DenyKey=system.run[]

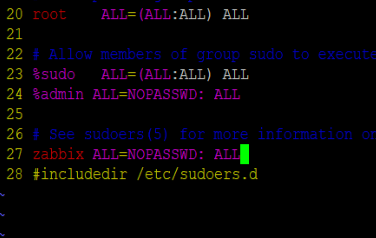
AllowKey=system.run[ \*]

* Liberar o sudo no Zabbix

*vim /etc/sudoers*

Escreva *“zabbix ALL=NOPASSWD: ALL”* na linha 27 do repositório

## Configurações dos scripts



1-Criar uma pasta na qual serão armazenados os scripts

*mkdir /etc/zabbix/scripts*

C:\Users\43017102807\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\54E476B4.tmp

2- Criar Scripts que serão atrelados aos itens do Checklist

Exemplo:

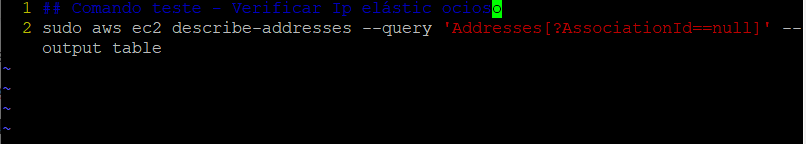
-Criar um arquivo do script dentro da pasta ‘Scripts’

*vim /etc/zabbix/scripts/exemploscript*

C:\Users\43017102807\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\48F472E2.tmp

-Colocar o comando dentro do arquivo

**Obs: Os comandos da AWS precisam ter o “sudo” antes.**



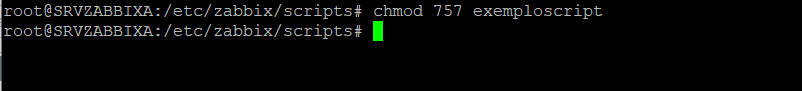
sudo aws ec2 describe-addresses --query 'Addresses[?AssociationId==null]' --output table

-Sair e salvar

*:Wq*

**Obs: a pasta na qual está localizada os scripts devem ter permissão 757 ou 777**

chmod 757 exemploscript



-Repita isso com todos os scripts

# Scripts Checklist

## VERIFICAÇÕES GENÉRICAS NA CONSOLE AWS

1.1 - Verificar se possui S3. ✅   
*sudo aws s3 ls | wc -l*

1.2 - Verificar se possui versionamento no S3.   
(manual)

1.3 - Verificar destinatários SNS   
\*   
1.4 - Verificar usuários e políticas IAM ✅    
*aws iam list-users*

1.5 - Verificar valores das faturas (Comparar últimas)   
(manual)   
1.6 - Possui faturas pendentes?   
(manual)

### IDENTIFICAÇÃO DO SERVIDOR

2.1 - Verificar a execução dos snapshots diários. ✅   
*data=(date +'%Y-%m-%d')*

*aws ec2 describe-snapshots --query "Snapshots[?(StartTime>='{data}')].{Time:StartTime}" | grep -v "[" | grep -v "]" | grep -v "{" | grep -v "}," | grep -v "}" | wc -l*

2.2 - Verificar se possui volumes sem uso. ✅   
*aws ec2 describe-volumes --filters Name=status,Values=available*

2.3 - Verificar instâncias reservadas. ✅   
*aws ec2 describe-reserved-instances*

2.4 - Verificar Elastic IP ocioso. ✅   
*aws ec2 describe-addresses --query 'Addresses[?AssociationId==null]' --output table*

2.5 - Verificar Security Group (EC2 + RDS ) ✅   
*aws ec2 describe-security-groups --output table*   
*aws rds describe-db-security-groups --output table*

2.6 - Verificar opção Autoscalling (Habilitada ou não) ✅   
*aws autoscaling describe-auto-scaling-instances*

2.7 - CloudWatch, alertas configurados, se está “OK”?   
(manual)   
2.8 - CloudWatch, alertas configurados, se está “Insufficient”? Corrigir.   
(manual)   
2.9 - CloudWatch, alertas configurados, se está Erro? Corrigir.   
(manual)

## VERIFICAÇÕES SSH

3.1 - Verificar se possui upgrades disponíveis (apt list --upgradable) - Corrigir com script ✅   
*apt list --upgradable*

3.2 - Verificar reinicialização pendente - Corrigir com script ✅   
*if ps -C shutdown > /dev/null; then*   
*echo "Shutdown is pending"*   
*else*   
*echo "Shutdown is not scheduled"*   
*fi*

3.3 - Verificar espaço em disco (df -h). ✅   
*df -h*

3.4 - Verificar uso de memória ram/swap (free -h) ✅   
*free -h*

3.5 - Validar antivírus   
(manual)   
3.6 - Validar fuso horário (date) ✅   
date

3.7 - Verificar Crons ✅   
*cat /etc/crontab*

3.8 - Verificar backups dos dias da semana ✅   
*cat /etc/crontab*

3.9 - Analisar logs de acesso Apache (buscar URL/HTTP/Code) ✅   
*tail -f /var/log/apache2/access.log*

4.0 - Analisar logs de erro Apache, sinalizar e enviar ao cliente ✅   
*tail -f /var/log/apache2/error.log*

# Userparameter

**UserParameter** é uma forma de estender as funcionalidades do Zabbix Agent. Com esta funcionalidade podemos criar novas chaves de itens de acordo com as nossas necessidades.

- Criar o arquivo **userparameter**

*touch /etc/zabbix/zabbix\_agentd.d/userparemeter\_aws.conf*

Após a criação dos scripts e da pasta ‘userparameter’

Entre no arquivo **‘userparameter’** e configure de acordo com o exemplo abaixo

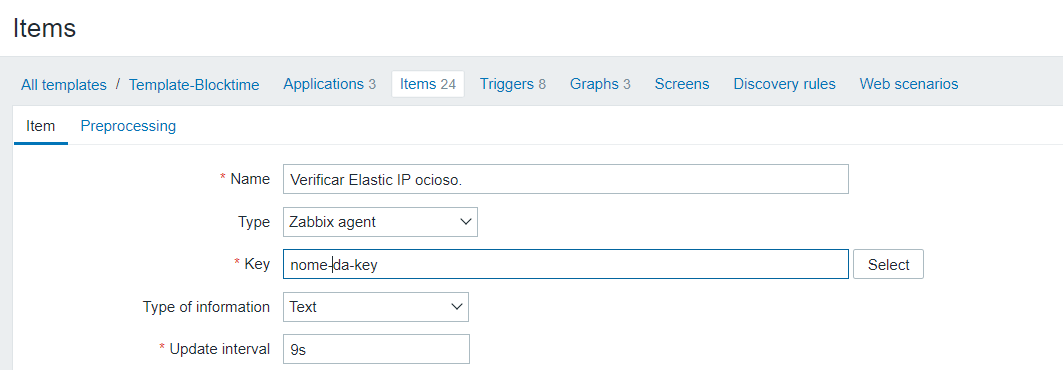
*vim /etc/zabbix/zabbix\_agentd.d/userparemeter\_aws.conf*

Exemplo:

* *Userparameter={nome-da-key},{/Caminho/exemploscript*

“**nome-da-key**” - Nome da key que você quer que seja configurada no zabbix como item personalizado.

C:\Users\43017102807\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\A83BF68C.tmp



**“Caminho” -** Caminho na qual você configurou o script

*\*/etc/zabbix/scritps/exemploscript****\****

C:\Users\43017102807\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\A035C698.tmp

Exemplo na prática:

* *Userparameter=exemploscript,/etc/zabbix/scripts/exemploscript*

-Repita isso com todos os itens, se for o caso

## Arquivo Userparameter do Checklist

**Em Linha:**

UserParameter=Verificarsnap,/etc/zabbix/scripts/teste.sh

##      QUANTIDADE DE S3(BUCKETS)

UserParameter=Verificar,/etc/zabbix/scripts/Verificar

##      VERIFICAR USUARIOS DO IAM

UserParameter=Verificar4,/etc/zabbix/scripts/Verificar4

##      VERIFICAR SE POSSUI VOLUMES SEM USO

UserParameter=Verificar22,/etc/zabbix/scripts/Verificar22

##      VERIFICAR INSTÂNCIAS RESERVADAS

UserParameter=Verificar23,/etc/zabbix/scripts/Verificar23

##      VERIFICAR ELASTIC IP OCIOSO

UserParameter=Verificar24,/etc/zabbix/scripts/Verificar24

##      VERIFICAR SECURITY GROUP EC2 + RDS

UserParameter=Verificar25,/etc/zabbix/scripts/Verificar25

##      VERIFICAR OPÇÃO DE AUTOSCALLING

UserParameter=Verificar26,/etc/zabbix/scripts/Verificar26

##      VERIFICAR SE POSSUI UPGRADES DISPONIVEIS

UserParameter=Verificar31,/etc/zabbix/scripts/Verificar31

##      VERIFICAR REINICIALIZAÇÃO PENDENTE

UserParameter=Verificar32,/etc/zabbix/scripts/Verificar32

##      VERIFICAR ESPAÇO EM DISCO

UserParameter=Verificar33,/etc/zabbix/scripts/Verificar33

##      VERIFICAR USO DE MEMORIA RAM/SWAP

UserParameter=Verificar34,/etc/zabbix/scripts/Verificar34

##      VALIDAR FUSO HORÁRIO

UserParameter=Verificar36,/etc/zabbix/scripts/Verificar36

##      VERIFICAR CRONS

UserParameter=Verificar37,/etc/zabbix/scripts/Verificar37

##      VERIFICAR BACKUPS DOS DIAS DA SEMANA

UserParameter=Verificar38,/etc/zabbix/scripts/Verificar38

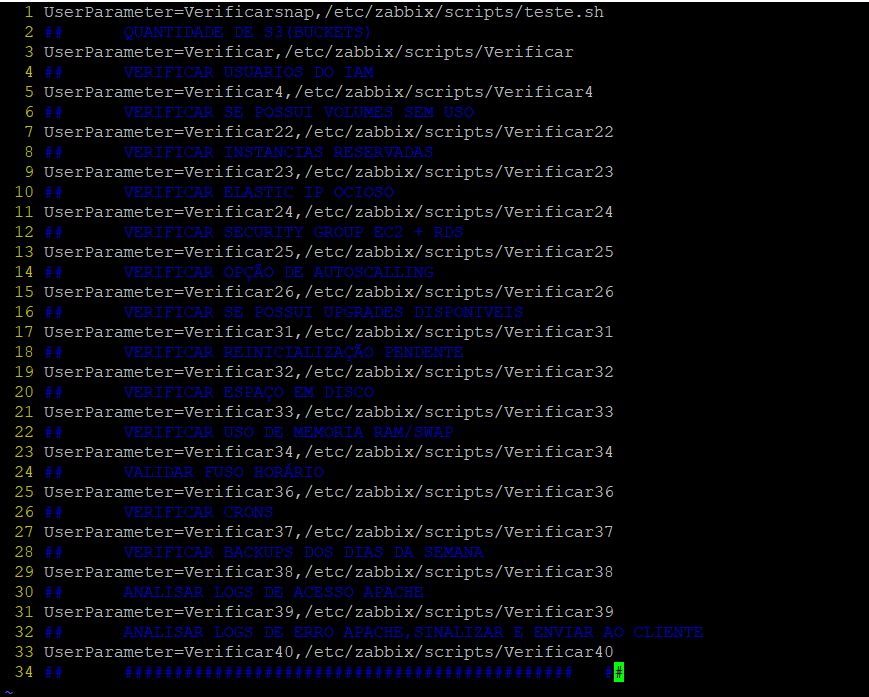
##      ANALISAR LOGS DE ACESSO APACHE

UserParameter=Verificar39,/etc/zabbix/scripts/Verificar39

##      ANALISAR LOGS DE ERRO APACHE, SINALIZAR E ENVIAR AO CLIENTE

UserParameter=Verificar40,/etc/zabbix/scripts/Verificar40

**No Repositório ‘’Vim’’:**



* **ATENÇÃO:**Salve e saia do arquivo (wq) e de o comando:

*systemctl restart zabbix-agent.service*

Para reiniciar o serviço zabbix-agent

# Zabbix Trapper

Alguns scripts ultrapassam o limite máximo de tempo de resposta e com isso geram um timeout. Então, para resolver este problema precisamos utilizar o **Zabbix Trapper.**

Como pré-resquisito temos que instalar o **Zabbix Sender** no servidor, e para isso utilizamos o comando:

*apt install zabbix-sender*

C:\Users\43017102807\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\411B7164.tmp

Para configurar o **Zabbix Trapper** temos que utilizar o comando:

*crontab –e*

Para definirmos o tempo que um comando será executado:

C:\Users\43017102807\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\A1D73112.tmp

Dentro do crontab temos que digitar o seguinte comando no final do arquivo:

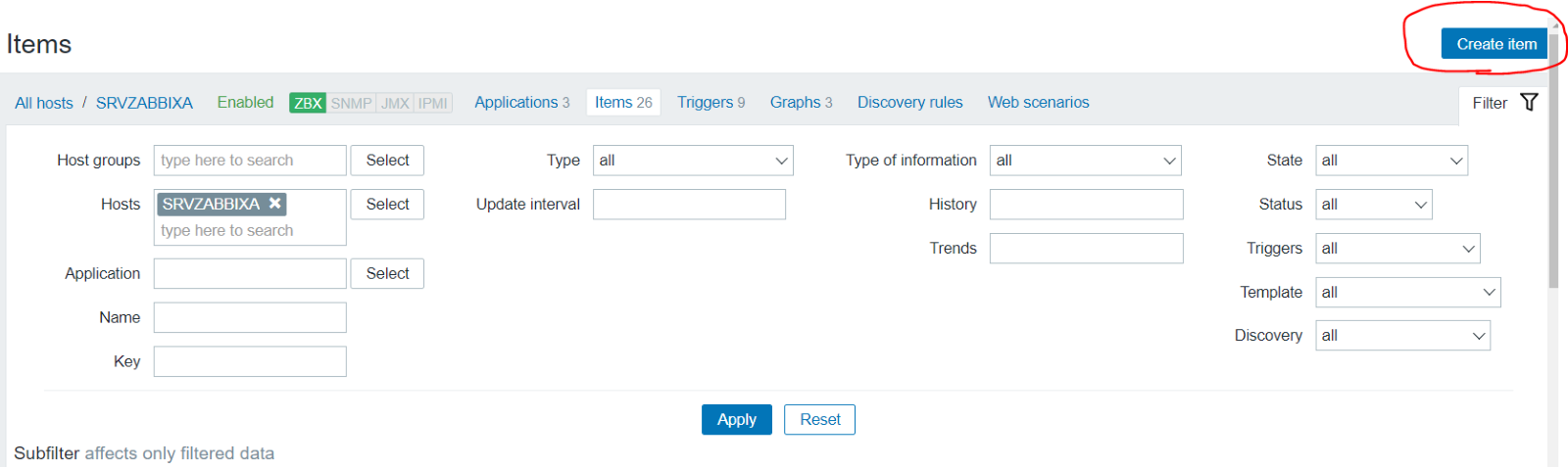
*\*/tempo(s) \* \* \* \* zabbix\_sender –z ip-do-zabbix-server -s “Nome-do-host” -k nome-da-chave  –o $(caminho-do-arquivo)*

Exemplo:

*\*/120 \* \* \* \* zabbix\_sender –z 34.230.169.32 -s “SRVZABBIXA” -k verificarSnap –o $(/etc/zabbix/scripts/Verificar21)*

C:\Users\43017102807\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\86BB42F0.tmp

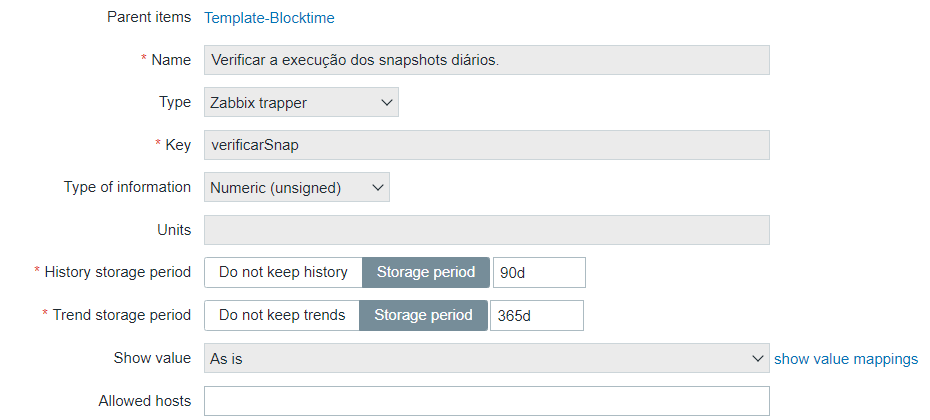
Em seguida, precisamos criar o **item** no Zabbix

**Clicamos** em **create** item

**Configurações do item:**

**1°**Colocamos o **Nome Do Item, o Type (Zabbix Trapper) e a Key.**

Obs: **nome-da-key** precisa estar configurada no Crontab



**2°** Verificar se os dados estão sendo exibido em L**atest Data** (que fica em monitoramento)

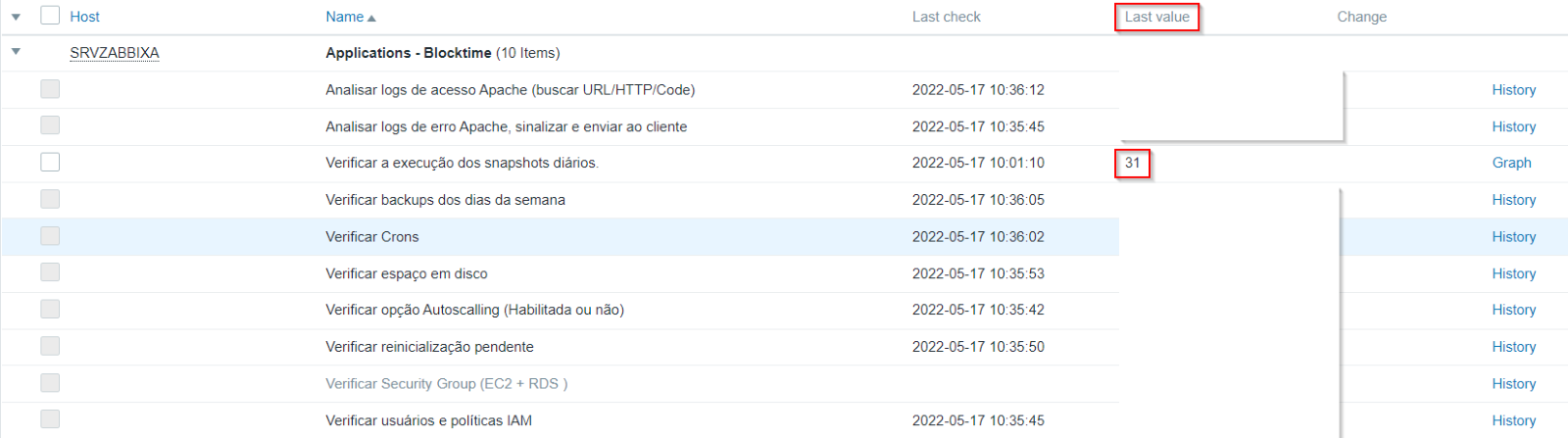
Um ponto importante é que os dados serão exibidos conforme o **tempo** que foi especificado no **Crontab,** caso vocês queiram acelerar esse tempo para que os dados sejam exibidos mais rápido temos que executar o comando no próprio terminal do servidor

Por exemplo:

*zabbix\_sender –z 34.230.169.32 -s “SRVZABBIXA” -k verificarSnap –o $(/etc/zabbix/scripts/Verificar21)*

C:\Users\43017102807\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\3BD3FCAA.tmp

E depois é só verificar em **Latest Data** com o nome do item que foi criado, e os dados serão exibidos “rapidamente”



# Criação de items

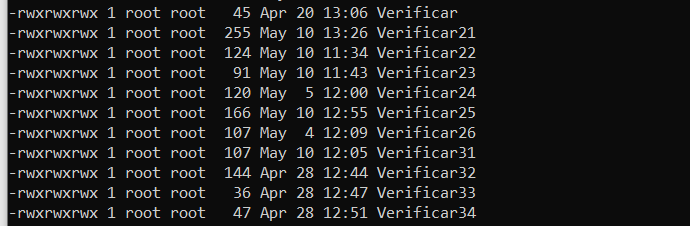
Para criarmos um item personalizado, precisamos saber a chave e o formado dos dados que será exibido pelo script, por exemplo:

Se vai ser em formato de **Number** ou **String**, no nosso caso utilizamos os dois formatos.

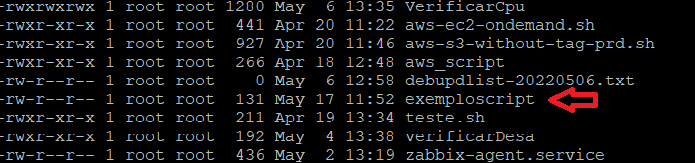
Então precisamos criar um item a partir do **Host,** onde foi configurado os scripts

 personalizados.

E a partir dai colocamos os scripts para monitorar a AWS.



Nesse caso, nós estamos utilizando como exemplo o **“exemploscript”**



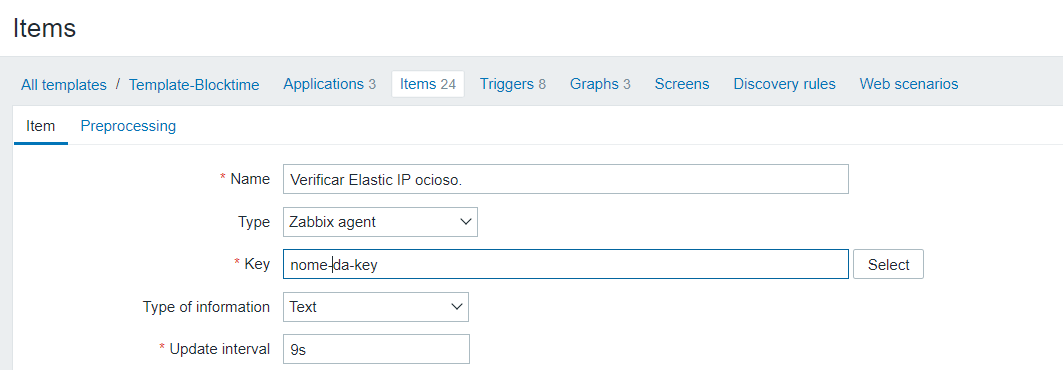
Dentro de cada arquivo colocamos os scripts específicos e chamamos esses scripts no UserParameter (como foi descrito em alguns exemplos acima).

E para criarmos esses items no zabbix server temos que criar um item a partir do host que está configurado o UserParameter.

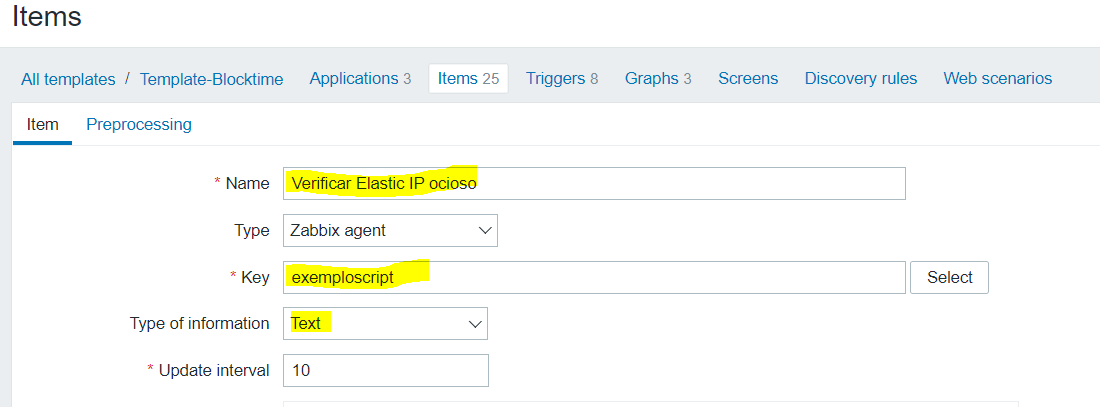
**Zabbix:**

Clicamos em create item e depois especificamos a key que foi criado no UserParameter o type of information (se será texto ou número), o type onde configuramos o userparameter, “nesse caso é o zabbix agent” , e também o nome do item.

C:\Users\43017102807\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\F9C23B42.tmp



**EX:**

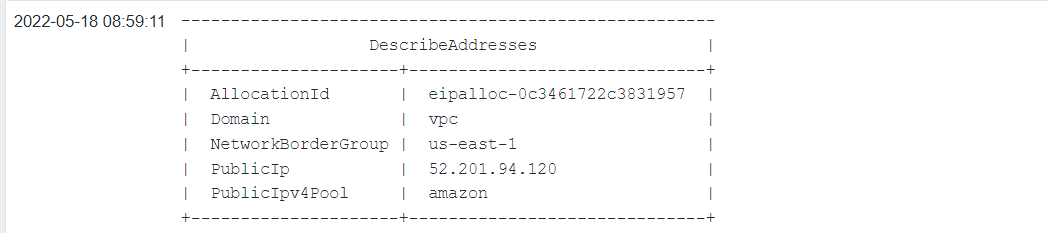


Logo mais abaixo vai ter uma opção de add

(e assim adicionamos o item) e verificamos depois se está gerando dados



Gerou dados, então a criação de item personalizados, deu certo!!!



# Criação de trigger

Gatilho ou trigger é um recurso de programação executado sempre que o evento associado ocorrer. Trigger é um tipo especial de procedimento armazenado, que é executado sempre que há uma tentativa de modificar os dados de uma tabela que é protegida por ele.

Pode se disser também que ela é um tipo de a

**Nenhuma entrada de sumário foi encontrada.**

lerta para se executar alguma ação.

Para construir uma trigger é necessário:

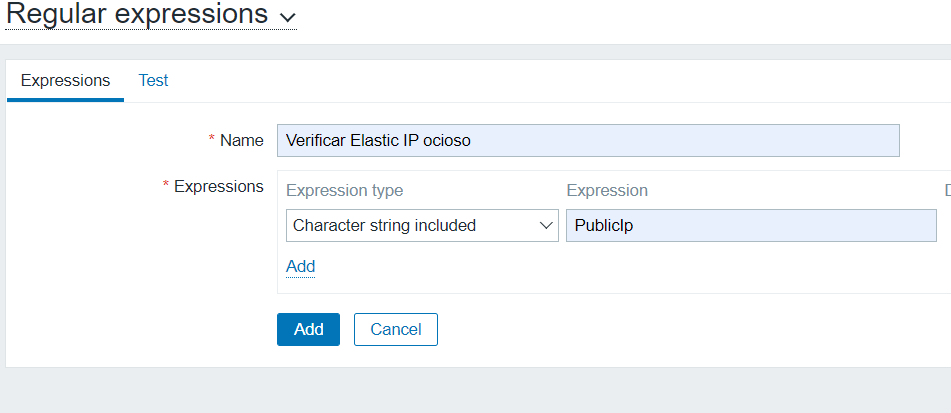
* Acessar: *Configuração* → *Hosts* ou *Configuração* → *Templates*
* Clicar no link *Triggers* na linha do host ou template
* Clicar no botão *Criar trigger* situado no canto superior da tela (ou no nome da trigger que se deseje atualizar a configuração)
* Informar os parâmetros da trigger no formulário

## Regular expressions com triggers

Para efetuar o que foi pedido pela empresa, nós iremos utilizar as expressões regulares, que através delas nós informamos para o Zabbix o que seria o certo e errado.

Então teremos que acessar:

* Administração → General → GUI → Regular Expressions
* Clicar em “Nova Expressão Regular”
* Informar os parâmetros da Expressão Regular
* Após isso, clique em add

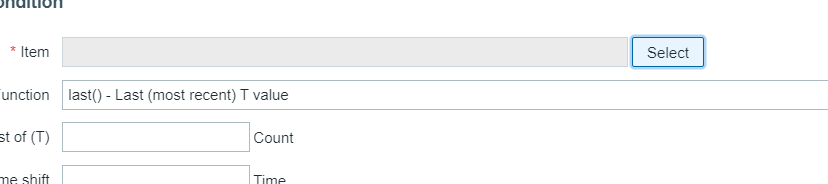


Neste exemplo, informei que se incluir o caractere ‘PublicIp’, o resultado será verdadeiro

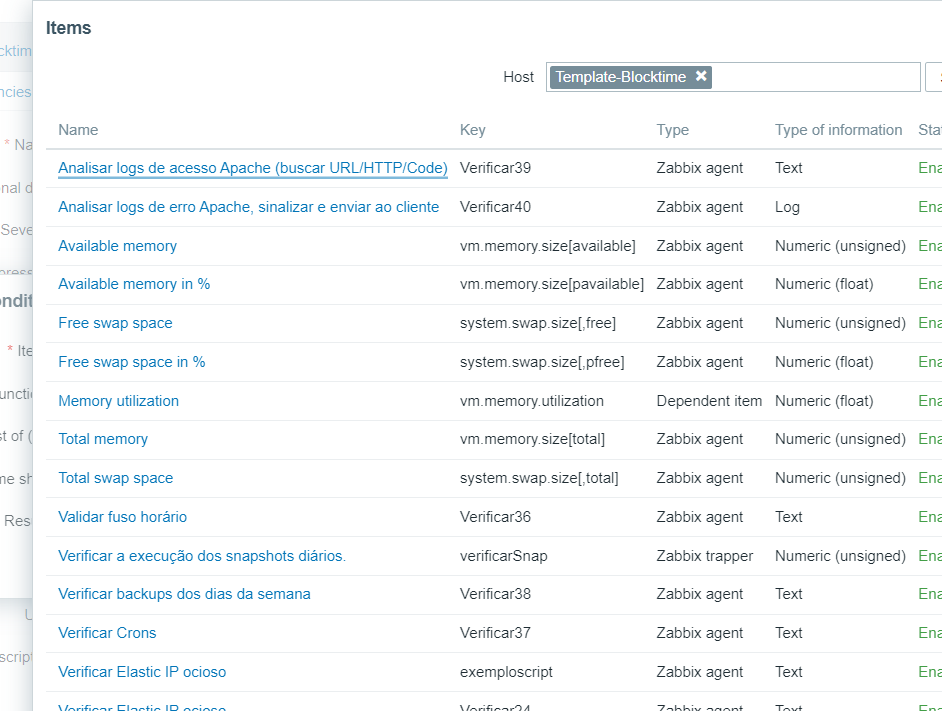
Para utilizarmos essa expressão regular em uma trigger, precisamos incluir ela a partir de um item de algum host que queremos alertar.

* Após definirmos a expressão regular ‘PublicIp’ como verdadeira, temos que criar uma condição na trigger.

Na configuração de uma trigger precisamos selecionar o item “botão select”



Escolhemos o item:



E depois, em “function” escolhemos “regexp()” - Verifica se o último (mas recente) valor confere com determinada expressão regular.

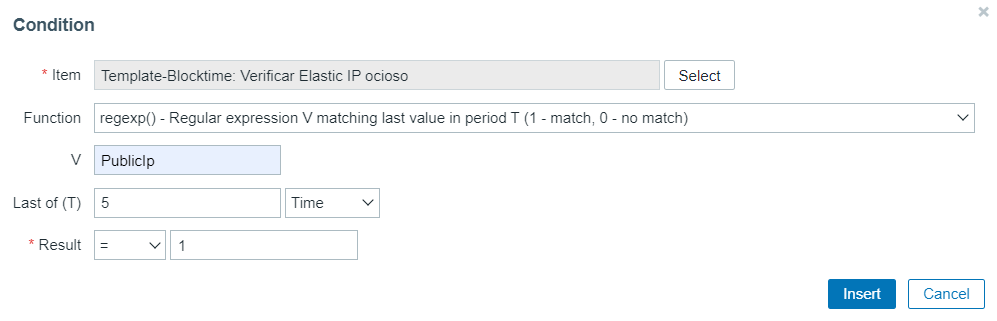
Em V: colocamos o valor que foi adicionado na Expressão regular “PublicIp”

Em last of(T): Tempo para fazer a consulta, nesse caso, utilizamos ‘5s’ (Escolha o tempo de acordo com a sua necessidade)

Result: pode ser encontrado (1) ou não encontrado (0)

**Lógica da trigger:**

* Se for encontrado (1), eu quero que ele me gere um alerta
* Nesse exemplo, se retornar PublicIp no comando significa que ele possui IP Elastic ocioso



* Logo após, é só inserir e verificar.

Retorno:



* Esse processo é utilizado aos demais scripts (Quais necessitam de trigger).

# Conclusão

O objetivo de monitorar a AWS via Zabbix foi alcançado porem, é necessário primeiro, executar o passo a passo de configurações. Também é recomendado atrelar os itens e triggers a um *Zabbix-Template* para a reutilização das configurações uma vez que feitas.