ZABBIX '24 CONFERENCE

LATIN AMERICA

JUNE, 07-08, 2024. SÃO PAULO, BRAZIL

ZABBIX '24
CONFERENCE
LATIN AMERICA

Construindo plugins do Zabbix Agent 2 usando Go

kubectl apply about.yaml

Robert Silva

- DevOps Engineer na F1rst Tecnologia, uma empresa do Grupo Santander;
- Certificado ZCS, ZCP e ZCE (Versões 4.0, 5.0 e 6.0);
- Certificado Kubernetes CKA, CKAD;
- Atuando a +10 anos como consultor e arquiteto de soluções;
- Apaixonado por automação, focado em compartilhar conhecimento e experiências práticas do dia a dia;
- Organizador do DevOpsDays São Paulo 2024;
- Youtuber nas horas vagas.

ZABBIX '24
CONFERENCE
LATIN AMERICA

Esta palestra não possui nenhuma relação com o Banco Santander

O material apresentado a seguir não tem relação com o ambiente tecnológico do Santander.

Objetivo

Demonstrar o processo de criação de um novo plugin de monitoramento usando Go para estender cada vez mais as possibilidades de monitoramento e atender demandas cada vez mais complexas.

Motivação

A motivação inicial foi resolver um problema conhecido relacionado ao monitoramento de logs.

O problema

Para contextualizar vou usar como exemplo duas chaves de itens:

log[file,<regexp>,<encoding>,<maxlines>,<mode>,<output>,<maxdelay>,<options>,<persistent dir>]

logrt[**file regexp**,<regexp>,<encoding>,<maxlines>,<mode>,<output>,<maxdelay>,<options>,<persistent dir>]

“file regexp - the absolute path to file and the file name described by a regular expression. Note that only the file name is a regular expression.”

Referência: https://www.zabbix.com/documentation/current/en/manual/config/items/itemtypes/zabbix_agent#log

O problema

E se temos logs em vários caminhos em um servidor?

Como resolver este problema?

Algumas possibilidades:

- Listar todos os caminhos possíveis e criar os itens de forma fixa;
- Usar UserParameter para receber uma expressão regular:
  - Shellscript - Somente Linux;
  - Powershell - Somente Windows;
  - Python – Não é nativo.
- Plugin do Zabbix Agent 2.

Como desenvolver um plugin?

- Conhecimento em Go;
- Entender o funcionamento do SDK da Zabbix;
- Desenvolver o Código;
- Associar seu plugin ao Zabbix Agent 2 rodando nos servidores.

Desafios

- Encontrar um material de referência;
- Documentação desatualizada ou com informações soltas.

Minhas pesquisas

Developing plugins for Zabbix Agent 2

By Vadim Ipatov — March 9, 2020

Principais pontos:

- Fornece uma boa base, porém desatualizado;
- Ter que compilar o Zabbix Agent 2 para usar um plugin desenvolvido;
- Roadmap:
  - Loadable plugins, permitir que use novos plugins sem compilar o Zabbix Agent 2 novamente.

Referência: https://blog.zabbix.com/developing-plugins-for-zabbix-agent-2/9682/

Minhas pesquisas

This is a step-by-step tutorial to create a simple loadable plugin for Zabbix agent 2.

What you'll create

During this tutorial, you will add a new loadable plugin **MyIP**. The plugin will implement 1 metric called **myip**, which returns the external IP address of the host where Zabbix agent 2 is running.

Part 1: Writing the Go code

In this section you will learn how to write the plugin that adds a new metric to Zabbix agent 2.

1. Create a new directory *myip* in */usr/local/zabbix/go/plugins/*.

2. Create the file *main.go* inside *myip* directory and define the name of your Go package.

Principais pontos:

- Ter um diretório especifico para desenvolver o plugin:
  - Torna o desenvolvimento complexo;
  - Ter que desenvolver em um servidor ou máquina com Zabbix Agent 2 instalado.
- Temos diferentes estações de trabalhos: Windows, Linux, Mac, como lidar com isso?
- Neste link já fala sobre o **loadable plugin.**

Referência: https://www.zabbix.com/documentation/current/en/devel/plugins/how_to

Qual exemplo seguir?

```
package main

import (
    "fmt"
    "io/ioutil"
    "net/http"
    "git.zabbix.com/ap/plugin-support/plugin/container"
    "git.zabbix.com/ap/plugin-support/plugin"
)
```

Referência: https://www.zabbix.com/documentation/current/en/devel/plugins/how_to

```
22
23              "golang.zabbix.com/agent2/internal/agent"
24              "golang.zabbix.com/agent2/pkg/glexpr"
25              "golang.zabbix.com/agent2/pkg/itemutil"
26              "golang.zabbix.com/agent2/pkg/zbxlib"
27              "golang.zabbix.com/sdk/conf"
28              "golang.zabbix.com/sdk/errs"
29              "golang.zabbix.com/sdk/plugin"
30      )
```

Referência: https://github.com/zabbix/zabbix/blob/release/6.4/src/go/plugins/log/log.go

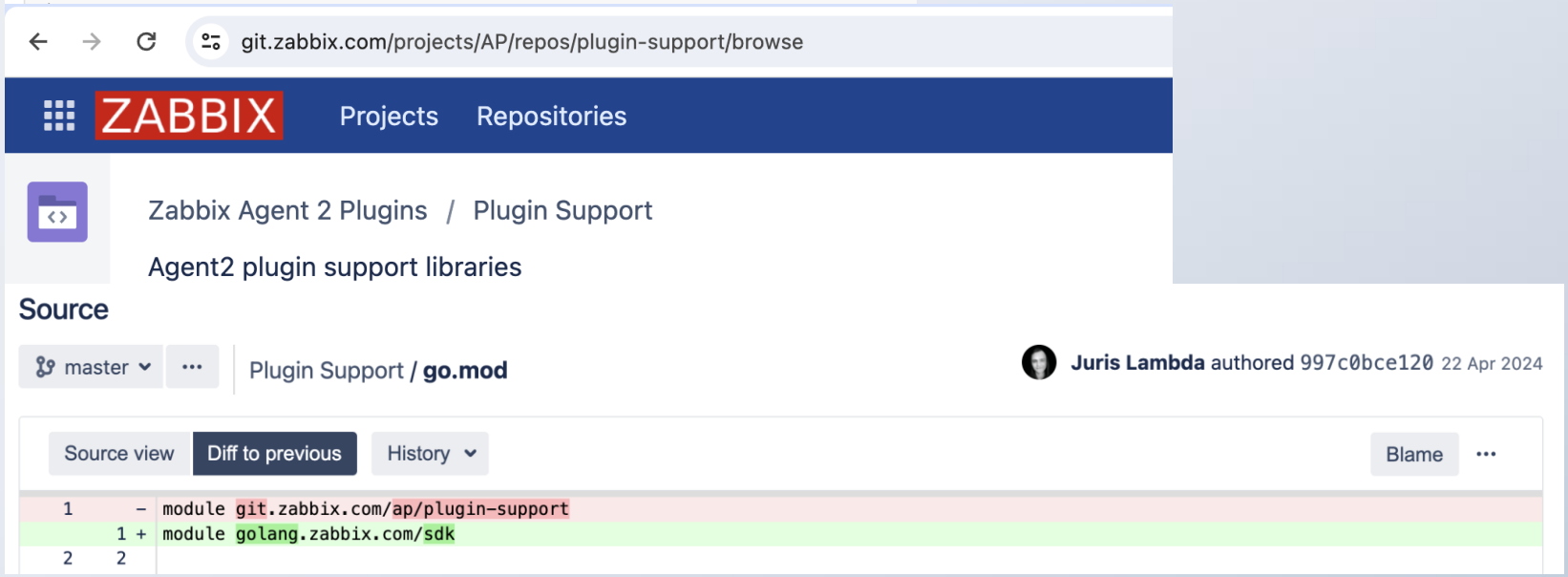
Entendendo o SDK

Since Zabbix 6.0, all the built-in plugin configuration parameters are located in Zabbix agent 2 source code. All the configuration parameters for loadable plugins are located in a separate repository.

Source Code: https://git.zabbix.com/projects/ZBX/repos/zabbix/browse/src/go/conf/zabbix_agent2.d/plugins.d
Loadable plugins: https://git.zabbix.com/projects/AP

```
package main

import (
    "fmt"
    "io/ioutil"
    "net/http"
    "git.zabbix.com/ap/plugin-support/plugin/container"
    "git.zabbix.com/ap/plugin-support/plugin"
```



ZABBIX '24 CONFERENCE LATIN AMERICA

Entendendo o SDK

Um plugin é um pacote Go que define a estrutura e implementa uma ou várias interfaces de plugin: Exporter, Collector, Configurator, Runner, Watcher.

Uma interface em Go é um tipo que define um método.

https://git.zabbix.com/projects/AP/repos/plugin-support/browse/plugin/types.go

Desde a versão 6.0, dois tipos de plugins são suportados:

- Built-in plugins (plugins integrados):
  - Necessário compilar o Zabbix Agent.
- Loadable plugins (plugins carregados):
  - Não precisa compilar o Zabbix Agent.

Referência: https://www.zabbix.com/documentation/guidelines/en/plugins

plugin.Exporter

- É a interface mais simples e é o suficiente para a maioria dos plugins;
- Pode consultar e retornar um valor, vários valores, um erro ou nada;
- Aceita chaves, parâmetros e contexto;
- Permite acesso simultâneo:
  - Limite de 100 chamadas simultâneas.

plugin.Collector

- É usado para plugins que precisam coletar dados regularmente;
- Não retorna dados, então precisa do Exporter para retornar os dados;
- Usado quando precisamos coletar dados com frequência e armazenar em cache até que o Zabbix Server ou Zabbix Proxy solicite esses dados.
- Possui duas funções:
  - Collect() = Possui a lógica da coleta de dados;
  - Period() = Possui a lógica do período de coleta de dados.

plugin.**Configurator**

- Utilizado para fornecer ao plugin parâmetros de configuração a partir dos arquivos de configurações do Zabbix Agent 2;
- Caso precise ler informações das configurações do Zabbix Agent 2 use esta interface.

plugin.**Runner**

- Permite realizar a inicialização quando o plugin é ativado;
- E parar quando o plugin é interrompido;
- Desta forma permite iniciar ou parar uma thread em segundo plano, liberar recursos não utilizados, encerrar conexões e etc.

plugin.**Watcher**

- Permite criar uma estrutura de coleta de dados sem utilizar o Scheduler do Agent;
- Como por exemplo itens de traps;
- O desenvolvedor pode criar sua lógica de programação para coletar os dados.

Limitações das interfaces

- Built-in plugins (plugins integrados):
  - Todas as interfaces.
- Loadable plugins (plugins carregados):
  - Exporter, Runner, Configurator.

ZABBIX '24
CONFERENCE
LATIN AMERICA

O que você precisa para começar?

Dicas para iniciar

Visual Studio Code Dev Containers

Go standards – Project Layout

zabbix-agent2-plugin-dirsearch
- README.md = Instruções de do repositório
- assets = Outros arquivos para acompanhar seu repositório (imagens, logotipos, essa apresentação, etc)
- build = Arquivos de build, Binários, Dockerfile, docker-compose.yaml, Manifestos kubernetes
- cmd = Tudo que vai ser usado APENAS pelo seu plugin
- docs = Documentacão do seu plugin
- examples = Exemplos gerais de uso
- go.mod = Arquivos do módulo
- go.sum = Arquivos do módulo

ZABBIX '24
CONFERENCE
LATIN AMERICA

O mínimo de um plugin

main.go

minimal-plugin > main.go

```go
package main
import (
    "fmt"
    "golang.zabbix.com/sdk/plugin/container"
    "golang.zabbix.com/sdk/plugin"
)
type Plugin struct {
    plugin.Base
}

var impl Plugin

func (p *Plugin) Export(key string, params []string, ctx plugin.ContextProvider) (result interface{}, err error){
    p.Infof("received request to handle %s key with %d parameters", key, len(params))
    var data string = "This is a minmal plugin"
    return data, nil
}

func init() {
    plugin.RegisterMetrics(&impl, "MinimalPlugin", "minimalplugin", "Return the external IP address of the host where agent is
    running.")
}
```

Pacotes externos

Estrutura do plugin

Exporter

Lógica do plugin

Registra métrica

Nome do Plugin

Chave do item

Descrição da chave

```go
func main() {
    h, err := container.NewHandler(impl.Name())
    if err != nil {
        panic(fmt.Sprintf("failed to create plugin handler %s", err.Error()))
    }
    impl.Logger = &h

    err = h.Execute()
    if err != nil {
        panic(fmt.Sprintf("failed to execute plugin handler %s", err.Error()))
    }
}

```

Cria a instancia do Plugin

ZABBIX '24
CONFERENCE
LATIN AMERICA

E um exemplo de uso real?

Product Solutions Open Source Enterprise Pricing

Search or jump to... / Sign in Sign up

robertsilvatech / zabbix-agent2-plugin-dirsearch Public

Notifications Fork 2 Star 0

Code Issues Pull requests Actions Projects Security Insights

main 2 Branches 0 Tags Go to file Code

Robert Silva and **Robert Silva** Change clone for https 9e354a1 · 6 hours ago 11 Commits

build/bin	New binarys and fix description key	6 hours ago
cmd/dirsearch	New binarys and fix description key	6 hours ago
docs	Add steps using dev container	7 hours ago
.gitignore	Add gitignore	8 hours ago
README.md	Change clone for https	6 hours ago
go.mod	Add f1rst files	8 hours ago
go.sum	Add f1rst files	8 hours ago

About

Plugins developed for zabbix agent 2

Readme

Activity

0 stars

1 watching

2 forks

Report repository

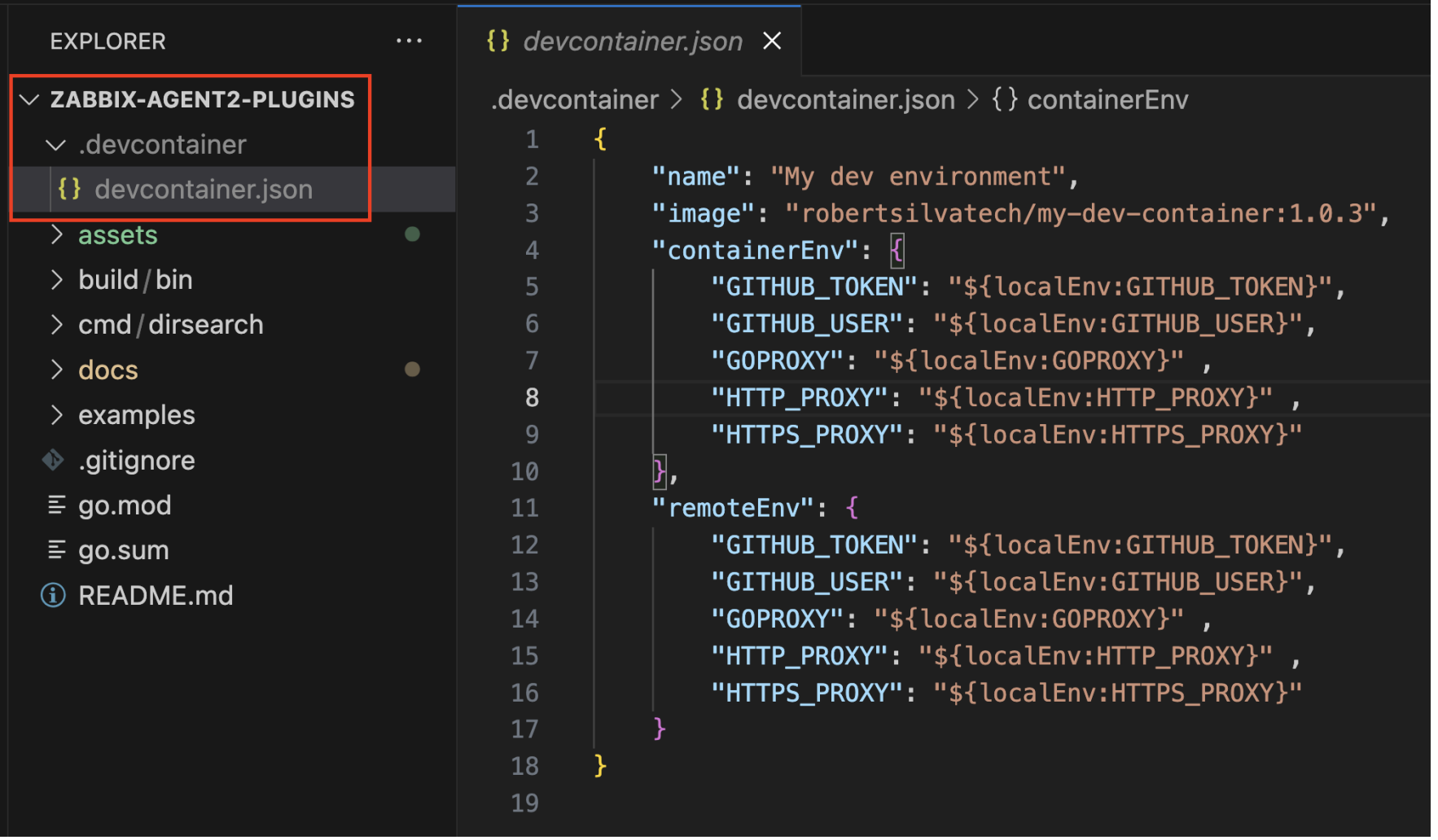
Releases

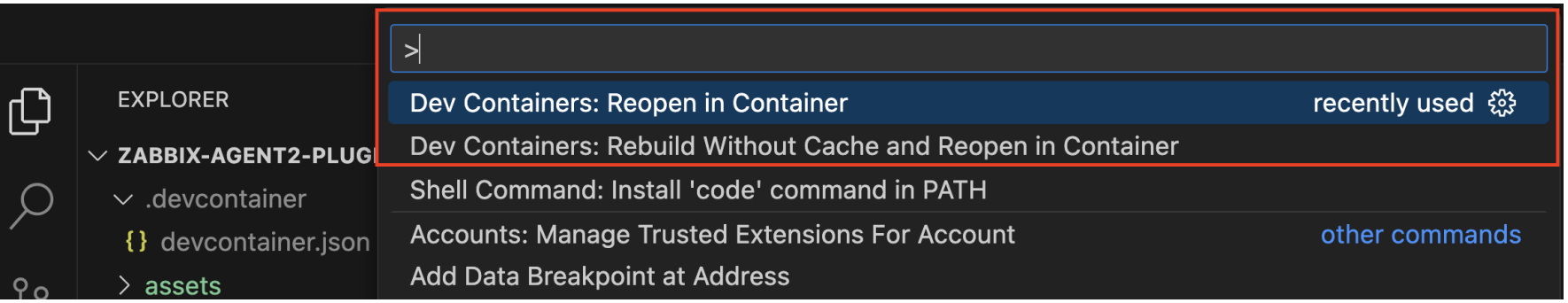
No releases published

Packages

ZABBIX '24
CONFERENCE
LATIN AMERICA

Ambiente de desenvolvimento





ZABBIX '24
CONFERENCE
LATIN AMERICA

my-dev-container / Dockerfile

robertsilvatech add kyverno 1997511 · 5 months ago History

Code Blame 43 lines (33 loc) · 1.56 KB Code 55% faster with GitHub Copilot Raw

```
1  FROM python:3.10-slim
2
3  RUN apt update && apt install vim curl wget unzip telnet tcpdump ncat openssl gettext -y
4
5  # jq & yq
6  RUN apt install jq wget -y && \
7      wget https://github.com/mikefarah/yq/releases/latest/download/yq_linux_amd64 -O /usr/bin/yq &&\
8      chmod +x /usr/bin/yq
9
10 # helm
11 RUN curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 && \
12     chmod 700 get_helm.sh && \
13     ./get_helm.sh
14
15 # AWS CLI
16 #RUN curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip" && \
17 RUN curl "https://awscli.amazonaws.com/awscli-exe-linux-aarch64.zip" -o "awscliv2.zip" && \
18     unzip awscliv2.zip && \
19     ./aws/install
20
21 # Kubectl
22 RUN curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl" && \
23     chmod +x kubectl && mv kubectl /usr/local/bin
24
25 # ArgoCD CLI
26 RUN curl -sSL -o argocd-linux-amd64 https://github.com/argoproj/argo-cd/releases/latest/download/argocd-linux-amd64 && \
27     install -m 555 argocd-linux-amd64 /usr/local/bin/argocd && \
28     rm argocd-linux-amd64
```

Desenvolvendo o plugin

Criando diretório do plugin

```
mkdir -p cmd/dirsearch
mkdir -p build/bin
```

Criando o arquivo main.go

```
touch cmd/dirsearch/main.go
```

```go
func (p *Plugin) Export(key string, params []string, ctx plugin.ContextProvider) (result interface{}, err error) {
    p.Infof("received request to handle %s key with %d parameters", key, len(params))

    var lld []dirDiscovery

    dirSearch := strings.Split(params[0], "|")
    patterns := strings.Split(params[1], "|")
    patterns_array := strings.Join(patterns, "|")

    for _, dir := range dirSearch {
        fmt.Println(dir)
        filepath.WalkDir(dir, func(path string, d fs.DirEntry, err error) error {
            if err != nil {
                return err
            }
            if d.IsDir() {
                checkRegex := validadePath(patterns_array, path)
                if checkRegex {
                    lld = append(lld, dirDiscovery{Name: path})
                } else {
                    fmt.Println("Erro na regex", path)
                }
            }
            return nil
        })
    }
    b, err := json.Marshal(&lld)
    if err != nil {
        fmt.Println(err)
    }
    return string(b), nil
}
```

```go
func validadePath(pattern, path string) bool {
    patterns := strings.Split(pattern, "|")
    // fmt.Println(patterns)
    regexes := make([]*regexp.Regexp, 0)
    for _, pattern := range patterns {
        regexes = append(regexes, regexp.MustCompile(pattern))
    }
    for _, re := range regexes {
        validate := re.MatchString(path)
        if validate {
            return validate
        }
    }
    return false
}
```

Construindo o módulo do plugin

```
go mod init github.com/zabbix-agent2-plugin-dirsearch
GOPROXY=direct go get golang.zabbix.com/sdk/plugin@master
go mod tidy
go build -o build/bin/dirsearch cmd/dirsearch/main.go
GOOS=linux GOARCH=arm64 go build -o build/bin/dirsearch-linux-arm64 cmd/dirsearch/main.go
GOOS=linux GOARCH=amd64 go build -o build/bin/dirsearch-linux-amd64 cmd/dirsearch/main.go
```

Carregar o plugin para o Zabbix agent 2

```
echo 'Plugins.DirSearch.System.Path=/workspaces/zabbix-agent2-plugins/build/bin/dirsearch-linux-arm64' > /etc/zabbix/zabbix_agent2.d/plugins.d/dirsearch.conf
```

Testar a chave do plugin

```
zabbix_agent2 -t dir.search["/var","zabbix$"]
```

Output

```
dir.search[/var,zabbix$]                    [s|[{"name":"/var/log/zabbix"}]]
```

Supported keys

```
dir.search[dir_scan, regexp]
```

Parameters:

- dir_scan: The absolute path of the directory you want to search for subdirectories
- regexp: The regular expression that describes the required pattern

In this example we have the directory tree in /var and inside /var/log we have zabbix

```
├── local
├── lock -> ../run/lock
├── log
│   ├── anaconda
│   ├── audit
│   ├── btmp
│   ├── chrony
│   ├── cloud-init.log
│   ├── cloud-init-output.log
│   ├── cron
│   ├── dnf.librepo.log
│   ├── dnf.log
│   ├── dnf.rpm.log
│   ├── droplet-agent.update.log
│   ├── hawkey.log
│   ├── kdump.log
│   ├── lastlog
│   ├── messages
│   ├── private
│   ├── qemu-ga
│   ├── README -> ../../usr/share/doc/systemd/README.logs
│   ├── secure
│   ├── sssd
│   ├── tallylog
│   ├── wtmp
│   └── zabbix
```

The regex match will occur if it is a directory and if you include zabbix in the name, in this case **/var/log/zabbix**

ZABBIX '24
CONFERENCE
LATIN AMERICA

Obrigado!

ZABBIX