Zabbix Agent 2

ZABBIX 2020 Conference BENELUX

Viktors Tjarve
ZABBIX C Developer


ZABBIX? ZABBIX? ZABBIX?
ZABBIX?
ZABBIX? ZABBIX? ZABBIX?
ZABBIX?

Functionality of Zabbix Agent

- Daemon
- Collects data ⟶ Zabbix Server
- Stabile
- Efficient
- Written in C
- Flexible
  - UserParameter
  - system.run[]
  - Modules

UserParameter

ZABBIX

```
UserParameter=systemd.units, systemctl list-unit-files
```

- Defined in configuration file
- Simple to create
- For specific not pre-defined agent checks
- Run any command on the level of agent privileges
- Can be forked

system.run[*]

ZABBIX

- Regular item key
- Simple to create
- Run any command on the level of agent privileges
- No need to restart agent to make changes

system.run[*]

ZABBIX

Scary stuff

```
zabbix_agentd.conf:
Server=0.0.0.0/0
EnableRemoteCommands=1
LogRemoteCommands=0
User=root
AllowRoot=1

Operating system:
shell> iptables -F
shell> systemctl stop firewalls
shell> setenforce 0
```

Modules

- As fast as Zabbix native agent metrics
- Have to be written in C

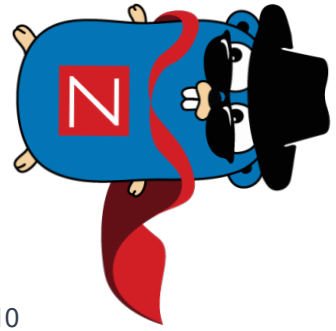
```
void main() {
    printf(“Hello Utrecht!\n")
}
```

Zabbix agents

Zabbix Agent 2

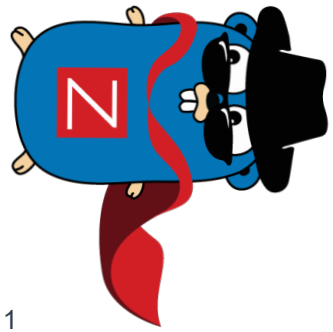
Who is this younger brother?

- Written in Go
- Backwards compatibility
  - Protocols
  - Configuration
  - Metrics
- Multiple parallel task execution for active checks
- Out-of-the-box *systemd* monitoring
- Custom plugin support on Windows
- Ease of developing custom plugins



Who is this younger brother?

- State preservation
- Continuous connection
- Custom plugin configuration
- Third-party traps
- Logging
- Timeouts implemented on the plugin level



What's the goal?

ZABBIX

Create the perfect monitoring tool

Always there is space for improvement

- Cover popular services
- Reduce need for:
  - expanding Zabbix functionality for complicated use cases
  - searching custom solutions on-line
  - adapting solutions found by others

Zabbix Agent - PostgreSQL

✔ Zabbix Agent

github.com/sergiotocalini/zapgix

share.zabbix.com/zapgix

Module for monitoring PostgreSQL

ibzbxpgsql (Lib-Zabbix-PostgreSQL) provides detailed and granular monitoring of PostgreSQL servers using a native Zabbix agent module

✔ Database Monitor

cavaliercoder.com/libzbxpgsql/

share.zabbix.com/postgresql-monitoring-for-zabbix

Monitoring agent for PostgreSQL

Tool for monitoring PostgreSQL with wide range of features

✔ Zabbix Agent Active

github.com/postgrespro/mamonsu

share.zabbix.com/mamonsu-monitoring-agent-for-postgresql

PostreSQL monitoring

Supports PostgreSQL 9.6+

✔ Low Level Discovery (LLD) ✔ Zabbix Agent Active

www.linuxelite.com.br/monitoramento/monitorando-postgresql-9-6-no-zabbix/

share.zabbix.com/postgresql-9-9

PosgreSQL partitioning

Scripts for partitioning postgresql

✔ DB Patch

github.com/cavaliercoder/zabbix-pgsql-partitioning

share.zabbix.com/postgresql-partitioning-scripts

PostgreSQL monitoring template for Zabbix

pg_monz enables various types of monitoring of PostgreSQL such as alive, resource, performance, etc. It supports some constitution patterns which includes single PostgreSQL pattern, HA pattern with Streaming Replication and load balancing pattern with pgpool-II

✔ LLD ✔ Zabbix Agent

pg-monz.github.io/pg_monz/index-en.html

share.zabbix.com/postgresql

PostgreSQL monitoring with Zabbix

The Template DB PostgreSQL was formerly known as pgCayenne. It is a set of UserParameter for PostgreSQL monitoring, which consists of Zabbix agent configuration and XML Template for web monitoring. It is using built-in PostgreSQL system views and functions. The agent requires the standard psql client ...

github.com/lesovsky/zabbix-extensions/tree/master/files/postgresql

Zabbix plugin to monitor RDBMS

ZabbixDBA is fast, flexible, and continuously developing plugin to monitor your RDBMS. ZabbixDBA uses threading of DBI connections which is good for monitoring of multiple database instances simultaneously. Just configure it, run daemon and it will do all the job

github.com/netrusov/ZabbixDBA

Zabbix PostgreSQL scripts

Scripts for partitioning the Zabbix database on PostgreSQL.

github.com/cavaliercoder/zabbix-pgsql-partitioning

PostgreSQL partitioning

Scripts to install, manage, and remove PostgreSQL partitioning for Zabbix DB

github.com/robbrucks/zabbix-postgresql-auto-partitioning

Docker for postgreSQL

Docker container for Zabbix DB on postrges, automatic partitioning.

github.com/blackcobra1973/zabbix-db-postgresql

Zabbix PostgreSQL template

Agenless remote postgresql monitoring using zabbix

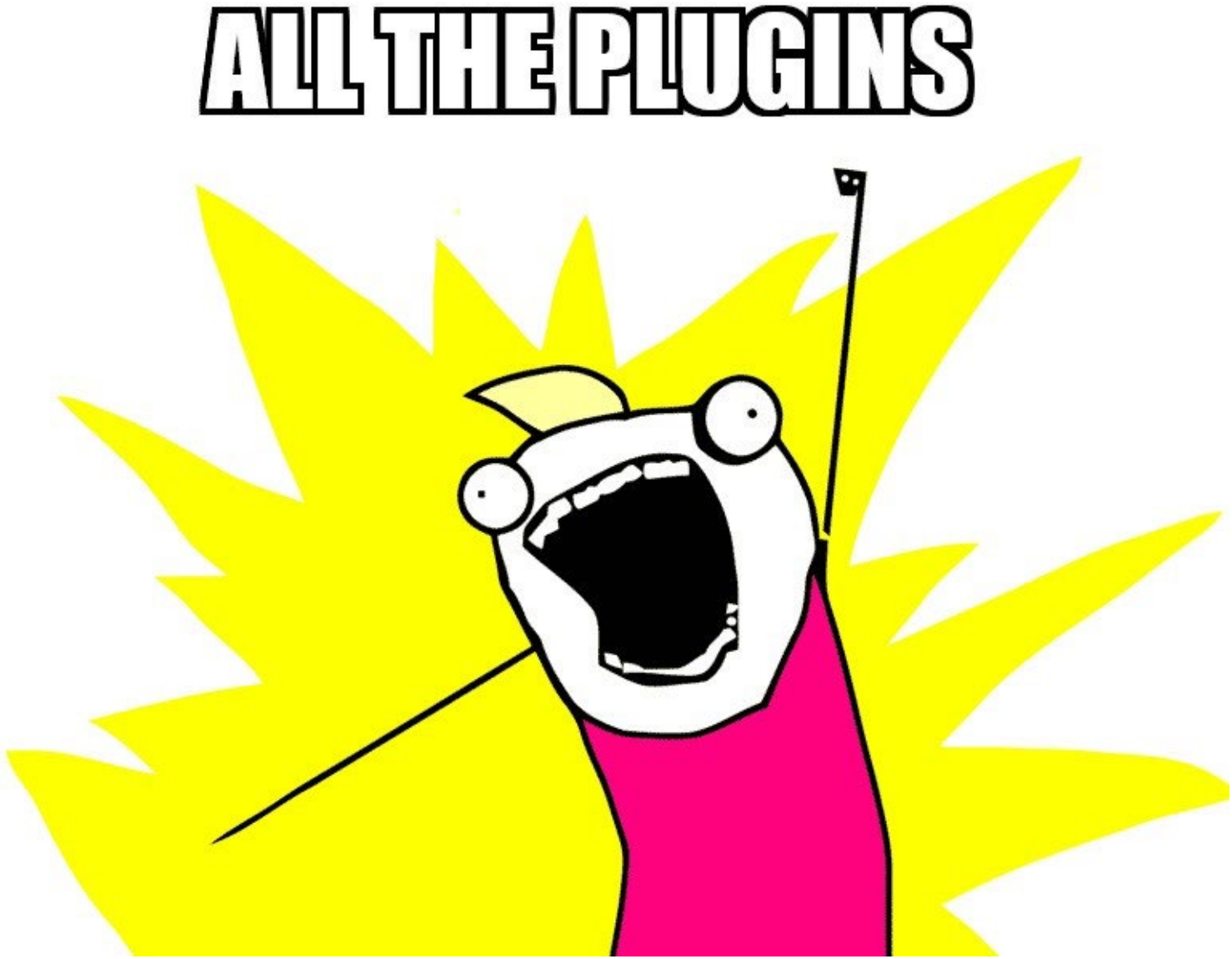
github.com/datorama/zabbix_postgresql_template

pgmon_2ndq is a set of postgresql monitoring functions implemented in the server.

The functions are installed in their own database and connect back to all databases in the server to get database specific infoA cronjob gets the monitoring info from these functions in a single database call and saves it to a file

github.com/postsql/pgmon_zabbix

Coding time



Coding time

ZABBIX

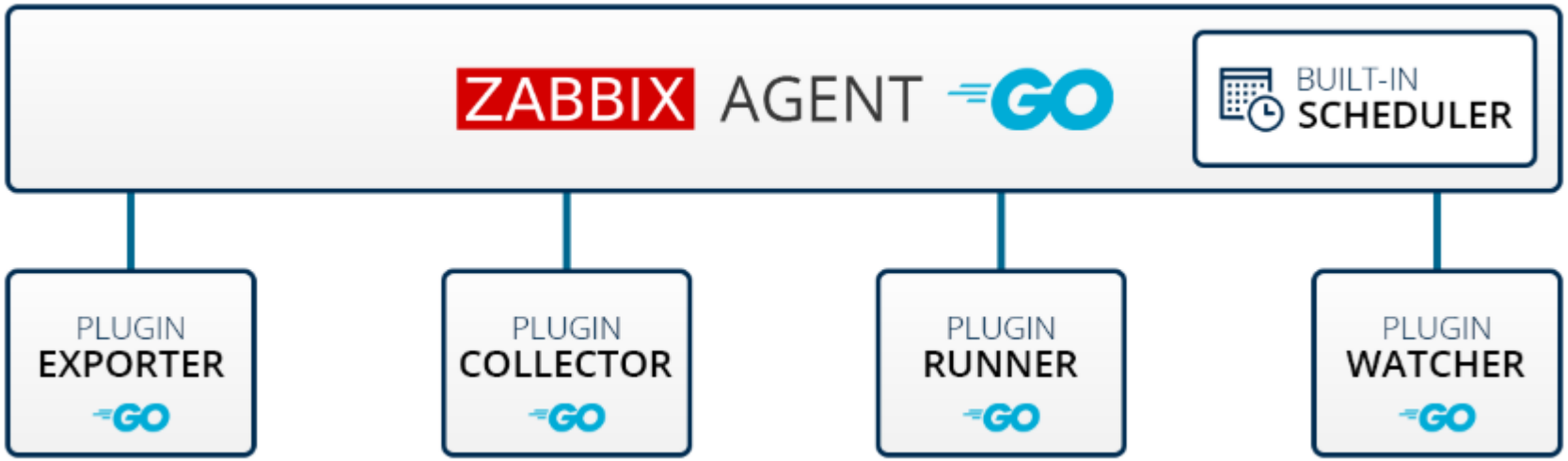
```
package packageName

import "zabbix.com/pkg/plugin"

type Plugin struct {
    plugin.Base
}

var impl Plugin

func (p *Plugin) Export(key string, params []string, ctx plugin.ContextProvider)
        (res interface{}, err error) {
    // Write your code here
    return res, err
}

func init() {
    plugin.RegisterMetrics(&impl, “PluginName", "key", "Description.")
}
```

The backbone of Agent2

ZABBIX

- Connector
- Listener
- Scheduler

Interfaces



- Exporter → pull
- Watcher → push
- Collector

plugin.Exporter

ZABBIX

```
// Exporter - interface for exporting collected metrics
type Exporter interface {
  // Export method exports data based on the key 'key' and
  // its parameters 'params'.
  Export(key string, params []string,
      context ContextProvider) (interface{}, error)
}
```

plugin.Exporter

ZABBIX

- MaxCapacity = 100
- SetCapacity(capacity int)
- Plugins.<PluginName>.Capacity = 1

plugin.Collector

ZABBIX

```
// Collector - interface for periodical metric collection
type Collector interface {
  Collect() error
  Period() int
}
```

plugin.Watcher

ZABBIX

```
// Watcher - interface for fully custom monitoring
type Watcher interface {
  // Watch method instructs plugin to watch for events based on
  // item configuration in 'requests'.
  Watch(requests []*Request, context ContextProvider)
}
```

More interfaces

ZABBIX

- Runner
- Configurator

plugin.Runner

ZABBIX

```
// Runner - interface for managing background processes
type Runner interface {
  // Start method activates plugin.
  Start()
  // Stop method deactivates plugin.
  Stop()
}
```

plugin.Configurator

ZABBIX

```
// Configurator - interface for plugin configuration
// in agent conf files
type Configurator interface {
  // Configure method passes global and private plugin
  // configuration after it has been activated.
  Configure(globalOptions *GlobalOptions, privateOptions interface{})
  // Validate method validates private plugin
  // configuration during agent startup.
  Validate(privateOptions interface{}) error
}
```

plugin.Configurator

```
KeepAlive int `conf:"optional,range=60:900,default=300"`
```

Combine interfaces

ZABBIX

system/cpucollector:

Collector: collects and aggregates data

Exporter: returns data on request

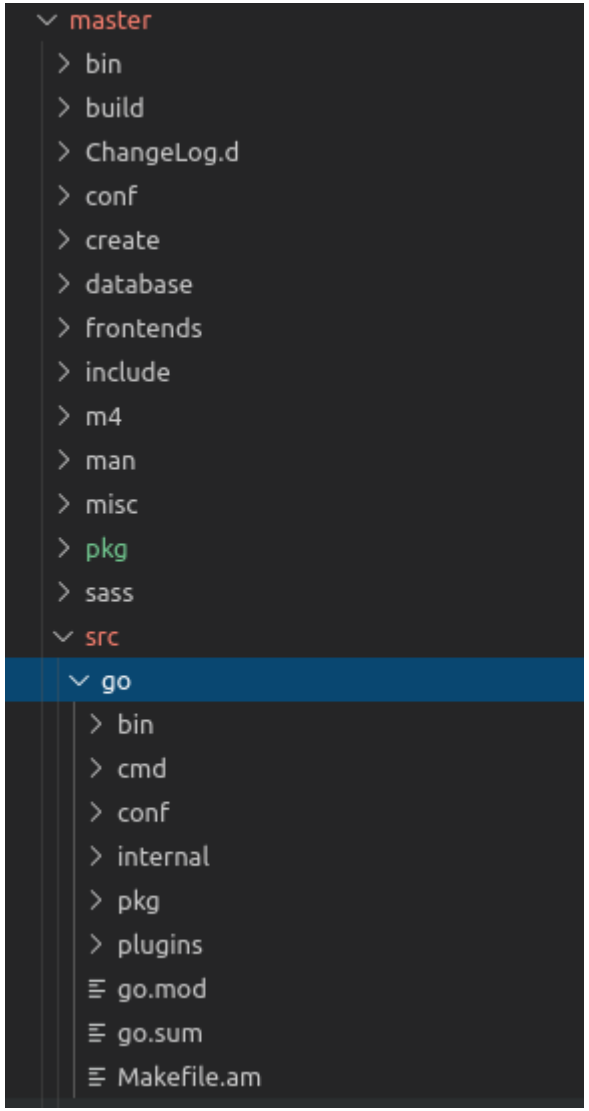
Runner: [de]initializes structures

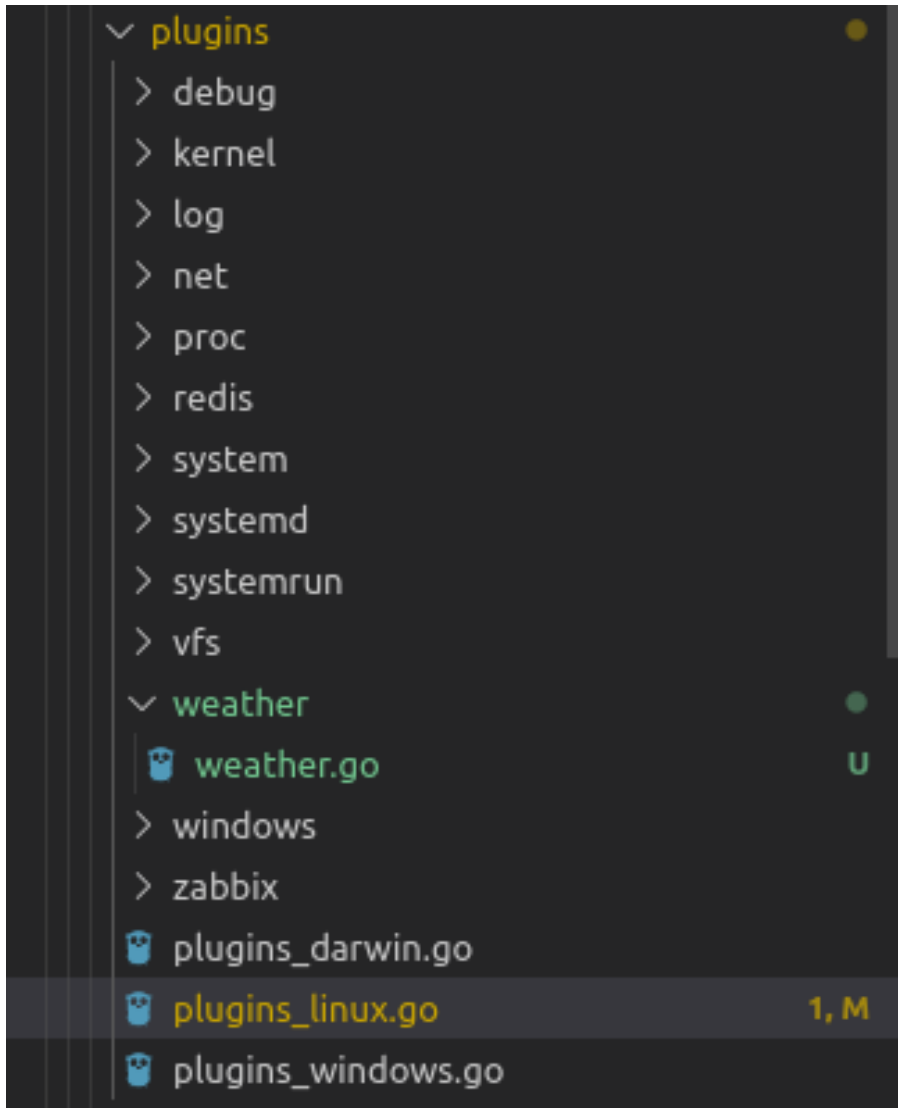
Plugin types

ZABBIX

- **Internal**: go/internal/agent/plugin_*.go
- **External**: go/plugins

Source code file tree





Hello world!

ZABBIX

1. Import plugin

```
package weather
import  "zabbix.com/pkg/plugin"
```

2. Declare our structure using Base

```
type Plugin struct {
  plugin.Base
}
var impl Plugin
```

Hello world!

ZABBIX

3. Using the interface

```
func (p *Plugin) Export(key string, params []string,
        ctx plugin.ContextProvider) (result interface{}, err error) {
  // https://github.com/chubin/wttr.in
  response, err := http.Get(fmt.Sprintf("https://wttr.in/~%s?format=%%t", params[0]))
  if err != nil {
    return nil, err
  }
  defer response.Body.Close()

  temp, _ := ioutil.ReadAll(response.Body)

  return string(temp)[0:len(temp)-4], nil
}
```

Hello world!

ZABBIX

4. Registration of metrics

```
func init() {
  plugin.RegisterMetrics(&impl, "Weather", "weather.temp",
        "Returns Celsius temperature.")
}
```

```
// impl – pointer to the implementation of the plugin
// name – plugin name
// params – list of metrics and their descriptions (key1, descr1, key2,
descr2, keyN, descrN...)
func RegisterMetrics(impl Accessor, name string, params ...string)
```

Hello world!

5. Adding the plugin to the list of imports in file *plugins_<platform>.go*

```
package plugins

import (
  _ "zabbix.com/plugins/kernel"
  _ "zabbix.com/plugins/log"
// ...
  _ "zabbix.com/plugins/weather"
)
```

Compilation

```
$ cd <zabbix-source>
$ ./bootstrap.sh; ./configure --enable-agent2; make install
```

That's it!

```
$ zabbix_agent2 -t weather.temp[utrecht]
weather.temp[utrecht]                   [s|+6]
```

Later on more simple way of compilation can be used:

```
$ go run <zabbix-source>/src/go/cmd/zabbix_agent2/zabbix_agent2.go
```

Monitoring of monitoring

ZABBIX

```
$ zabbix_agent2 -R metrics
...
[Weather]
active: true
capacity: 0/100
tasks: 0
weather.temp: Returns Celsius temperature.
...
```

QUESTIONS?
THANK YOU!

Viktors Tjarve
ZABBIX C Developer

ZABBIX 2020 Conference BENELUX

zabbix