



# **The Art of Monitoring**

TüBiX 2017 - Stephan Tesch

# Über mich

- ▶ Stephan Tesch
- ▶
- ▶ Lebt und arbeitet in Tübingen
- ▶ Seit 20+ Jahren im \*nix Umfeld zuhause
- ▶ Seit ca. 14 Jahren mit Monitoring Lösungen im Netzwerk- / Security Umfeld am werkeln
- ▶ Seit 7 Jahren Anhänger von Icinga
- ▶ Seit 2 Jahren Fan von Icinga 2

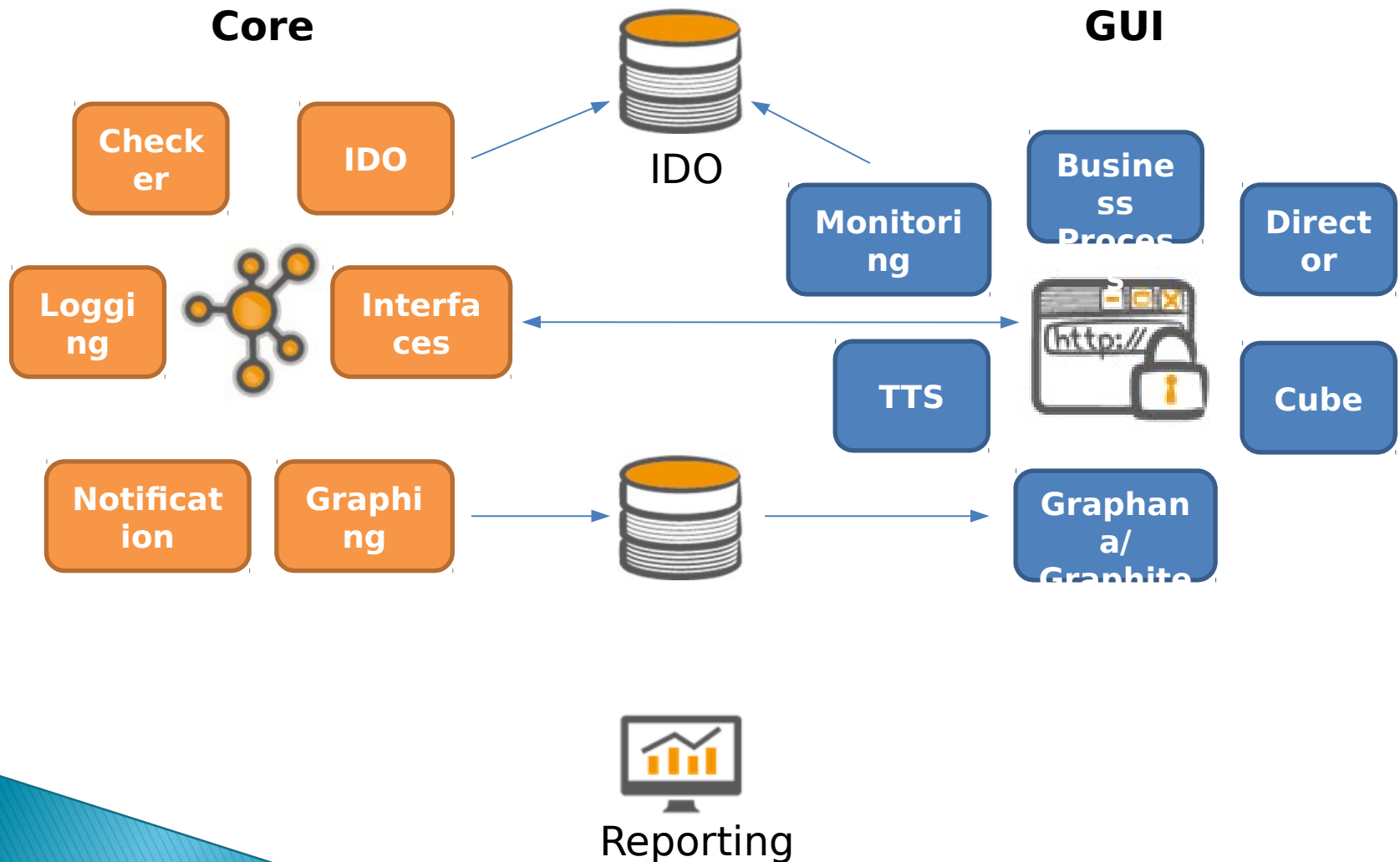
# Icinga - Was macht das?

- ▶ Überwachung und Alarmierung von Geräten und Diensten aller Art, die über ein Netzwerk erreichbar sind
- ▶
- ▶ Dazu laufen Check Skripte, die melden ob alles OK ist, eine Warnung ausgeben oder melden, dass ein kritischer Zustand erreicht wurde
- ▶
- ▶ Je nachdem kann eine Benachrichtigung erfolgen
- ▶

# Icinga - Origins

- ▶ Icinga 1.x startete 2009 als Fork von Nagios
- ▶ Fokus war die schnellere Entwicklung des Tools und mehr Beteiligung der Community
- ▶ Aufteilung in GUI (Classic UI)/ Web 2.0 GUI (Icinga Web) und Core
- ▶ Erstes 2.x Release (Technology Preview) in 2012
- ▶ Erstes stabiles Release in 2014
- ▶ Aktuelle Version -> 2.6.3
- ▶ Wikipedia: The name Icinga is a Zulu word meaning "it looks for", "it browses" or "it examines"<sup>[8]</sup> and is pronounced with a click consonant.<sup>[9]</sup>

# Icinga 2 - Komponenten



Grafik: adaptiert von <https://www.icinga.com>

# Icinga 2 - Komponenten

- ▶ Was gibt es sonst noch?
  - Icingabeat - Integration in ELK
  - Dashing - Reduzierung der Werte auf Dashlets
  - NagVis - diverse Visualisierungen
  - ???



# Icinga 2 - Checke



Check  
er

- ▶ \*\_**DIE**\_\* Core Komponente des Monitoring Systems
- ▶ Dynamischer Scheduler, der basierend auf der Config die Checks ausführt
- ▶ Kann über die Interfaces beeinflusst werden
- ▶ Je nach Zustand der überwachten Dienste gelten andere Intervalle für die Checks
- ▶ Checks liefern OK, WARNING, CRITICAL oder UNKNOWN zurück
- ▶ Ggf. zusätzlich Details, Performance Werte

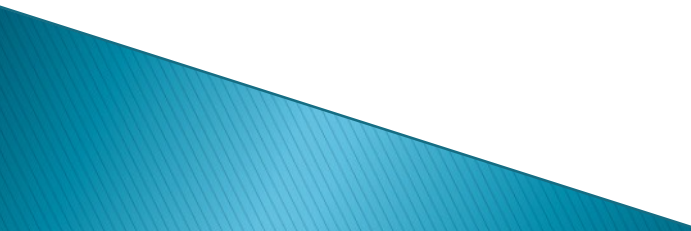
# Icinga 2 - Features

```
# icinga2 feature enable xyz
```

- ▶ Checker
- ▶ Interfaces
  - API | Command | Livestatus
- ▶ Graphing
  - Graphite | InfluxDB | opentsdb | perfdata
- ▶ IDO
  - mysql|pgsql
- ▶ Logging
  - Mainlog | compatlog | debuglog | gelf
- ▶ Notification



# Wie fange ich an

- ▶ Basis Installation (z.B. yum install icinga2)
  - ▶ CheckCommand Definition
  - ▶ Service Definition
  - ▶ Host Template Definition
  - ▶ Host Definition
- 
- A blue decorative triangle with a fine grid pattern is located in the bottom-left corner of the slide.

# Plugins

- ▶ Viele werden in der ITL schon definiert
  - Das reduziert den Konfigurationsaufwand deutlich
- ▶ Wenn doch ein spezieller Check benötigt wird

```
object CheckCommand "ive_cpu2" {  
  import "snmp"  
  vars.snmp_oid = ".1.3.6.1.4.1.12532.10.0"  
  vars.snmp_warn = host.vars.cpu_warn  
  vars.snmp_crit = host.vars.cpu_crit  
  vars.snmp_label = "CPU"  
  }  
  ◦  
  ◦
```

# Plugins

## ▶ Alternative für eigene Skripte

```
object CheckCommand "netscreen-nsrp2" {
  import "plugin-check-command"
  import "ipv4-or-ipv6"
  command = [ PluginContribDir + "/check_snmp_vrrp.pl" ]
  vars.nsrp_address = "$check_address$"
  arguments = {
    "-H" = vars.nsrp_address
    "-T" = "nsc"
    "-s" = host.vars.nsrp_state
    "-C" = host.vars.snmp_community
    "-2" = ""
  }
}
```

# Host Definition

- ▶ Host Definitionen sind einfach und übersichtlich

```
object Host "ive.tesch.cx" {  
  import "ive-template"  
  address = "192.168.10.147"  
  display_name = „My Juniper SA“  
  vars.sla = "3"  
  vars.parents = [ "moria.tesch.cx" ]  
}
```



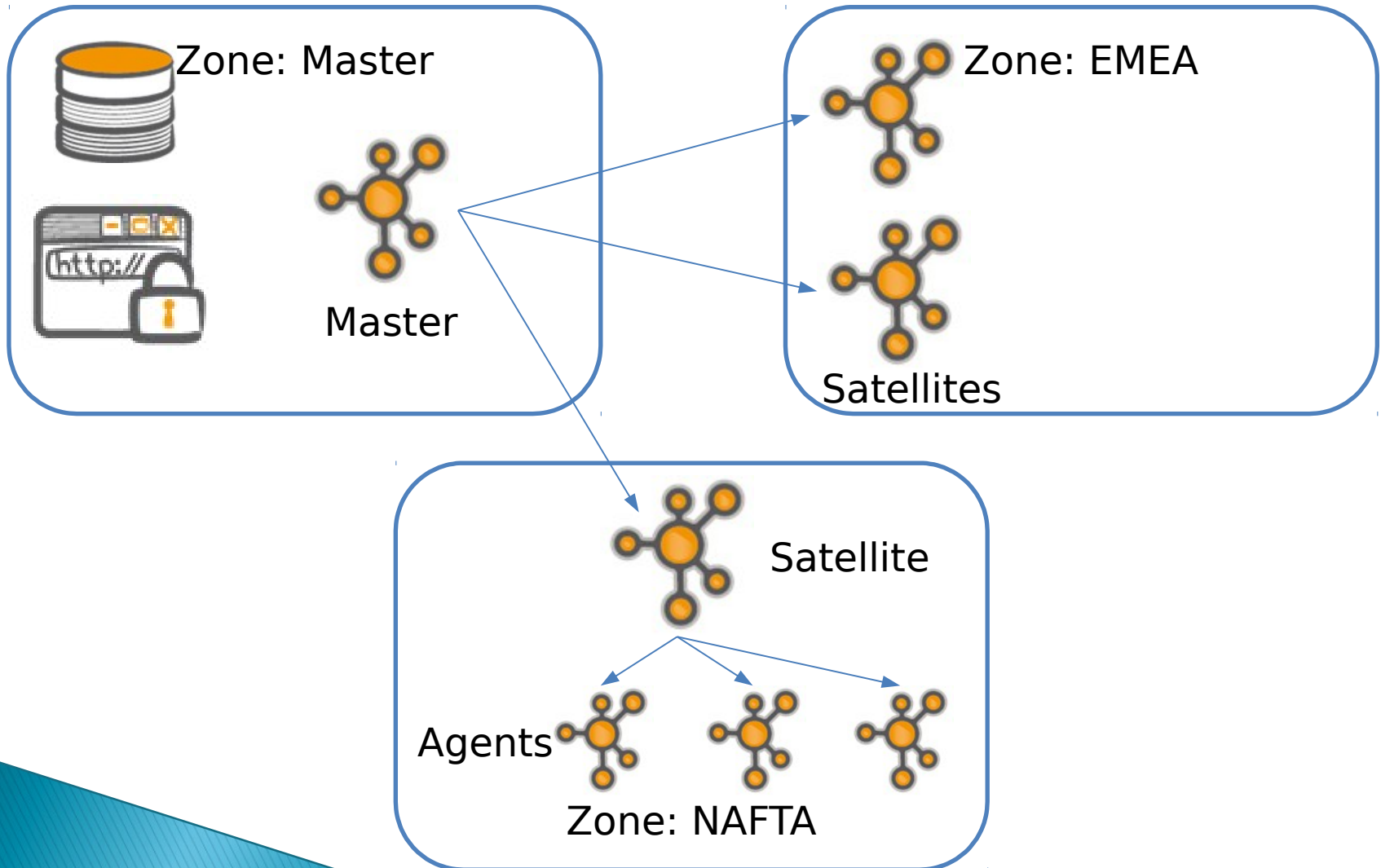
# Service Definitionen

- ▶ Sind super flexibel

```
object Service "uptime" {  
  host_name = "localhost"  
  check_command = "check_snmp"  
  vars.community = "public"  
  vars.oid = "DISMAN-EVENT-MIB::sysUpTimeInstance,"  
  groups = [ "all-services", "snmp" ]  
}
```

```
apply Service "ive_temp" {  
  import "generic-service"  
  display_name = "Temperature"  
  check_command = "ive_temp3"  
  assign where ( "sslvpn" in host.groups ) && \  
    ( host.vars.is_mac == true ) && \  
    host.vars.snmp_version != "2c"  
}
```

# Distributed Setup



# Advanced Foo

- ▶ Notification
  - Via Instant Messaging
    - ▢ Telegram
    - ▢
- ▶ Config Repo via SVN
  - Versionsverwaltung für Konfiguration (wenn nicht via Director verwaltet)
  - Konfigvalidierung via SVN hooks
- ▶

# Advanced Foo

- ▶ Verknüpfung von Checks verschiedener Zonen
  - Beispiel: Backup Checks
    - Backup Server in der Master Zone, Hosts sonstwo

```
object CheckCommand "backup" {
  import "plugin-check-command"
  command = [ PluginContribDir + "/checkbackup.sh" ]
  arguments = { "-H" = "$host_name$" }
}
apply Service "Backup Status" {
  import "generic-service"
  check_command = "backup"
  zone = "master"
  command_endpoint = BACKUPSERVER
  assign where host.vars.backup && ZoneName == "master"
  check_interval = 12h
}
```



# Nachteile

- ▶ Es gibt keine Parents mehr (im Vergleich zu Icinga 1.x / Nagios)
- ▶
- ▶ Eine automatisierte Migration der Nagios / Icinga 1.x Config gibt es nicht
- ▶
- ▶ SNMP Trap Integration ist furchtbar

# Demo

- ▶ Online Demo

# Fragen

- ▶ Entweder jetzt direkt, oder via  
stephan@tesch.cx