credativ

ICINGA2 TUTORIUM

FRÜHJAHRSFACHGESPRÄCH DER GUUG 2017

formorer / icinga2-tutorium

credativ

BEVOR WIR BEGINNEN

```
git clone https://github.com/formorer/icinga2-tutorium.git
cd icinga2-tutorium/vagrant/icinga2_icinga2web_grafana
vagrant up
```

formorer / icinga2-tutorium

credativ

VORSTELLUNGSRUNDE

credativ

WER BIN ICH?

- Senior Consultant credativ GmbH
- Debian/GRML Open Source Entwickler
- über 15 Jahre Erfahrung mit Monitoring
- Scrum Master

credativ

WER SIND SIE / WER SEID IHR?

- Du / Sie?
- Tätigkeitsfeld
- Erfahrungen mit Monitoring
- Erfahrungen mit Icinga2
- Erwartungen an das Tutorium


credativ


icinga

credativ

WAS IST ICINGA2

Icinga2 ist ein OpenSource Monitoring System das einfache und komplexe Monitoringszenarien abbilden kann.

Icinga2 ist kein Nagios Fork

FEATURES I

- Clusterfähig
- komplexe Konfigurationssprache
- Remote Agent
- Erweiterbar
- kompatibel zu den Monitoring Plugins
- Trending via
  - Graphite
  - InfluxDB
  - PNP4Nagios
  - OpenTSDB

credativ

FEATURES II

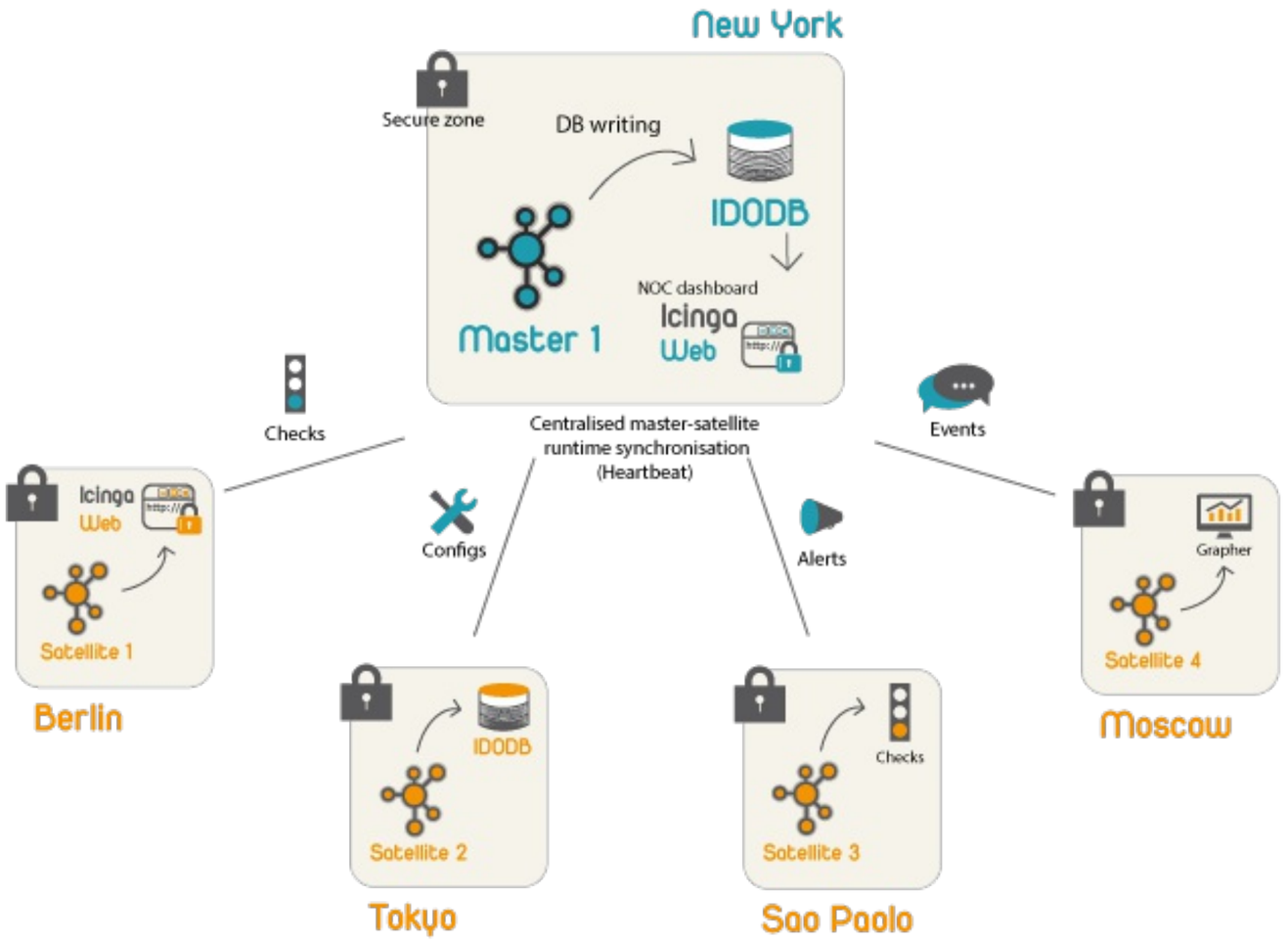
- Nativer Windows Support (Agent)
- semantisches Logging via GELF
- Darstellung von Businessprozessen
- ITL

credativ

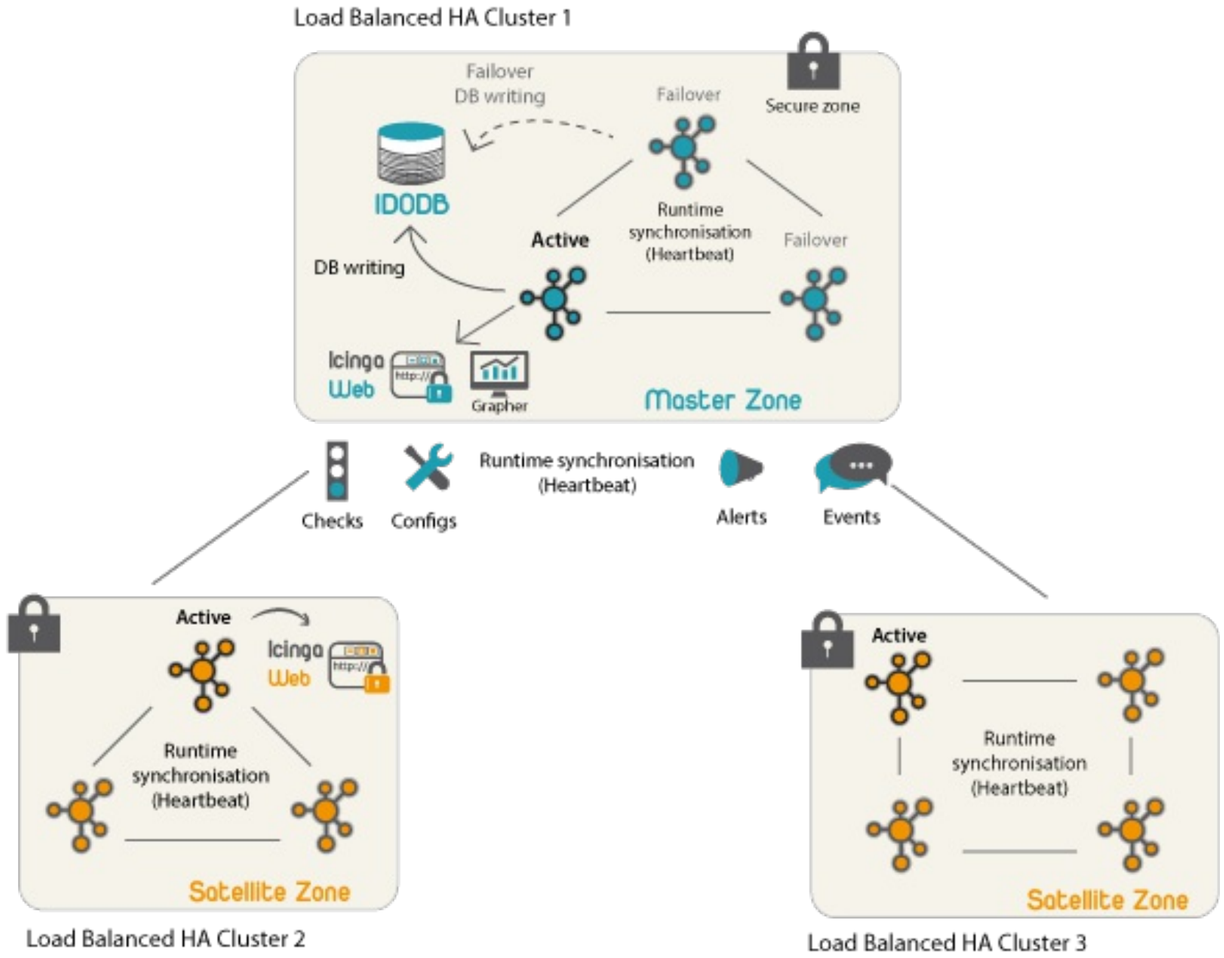
CLUSTERING

- Master/Satellite
- Master/Master (HA)
- Automatische Lastverteilung (Worker)
- Mehrstufige Setups
- Konfigurationssynchronisation
- X509

MASTER MIT SATELLITEN



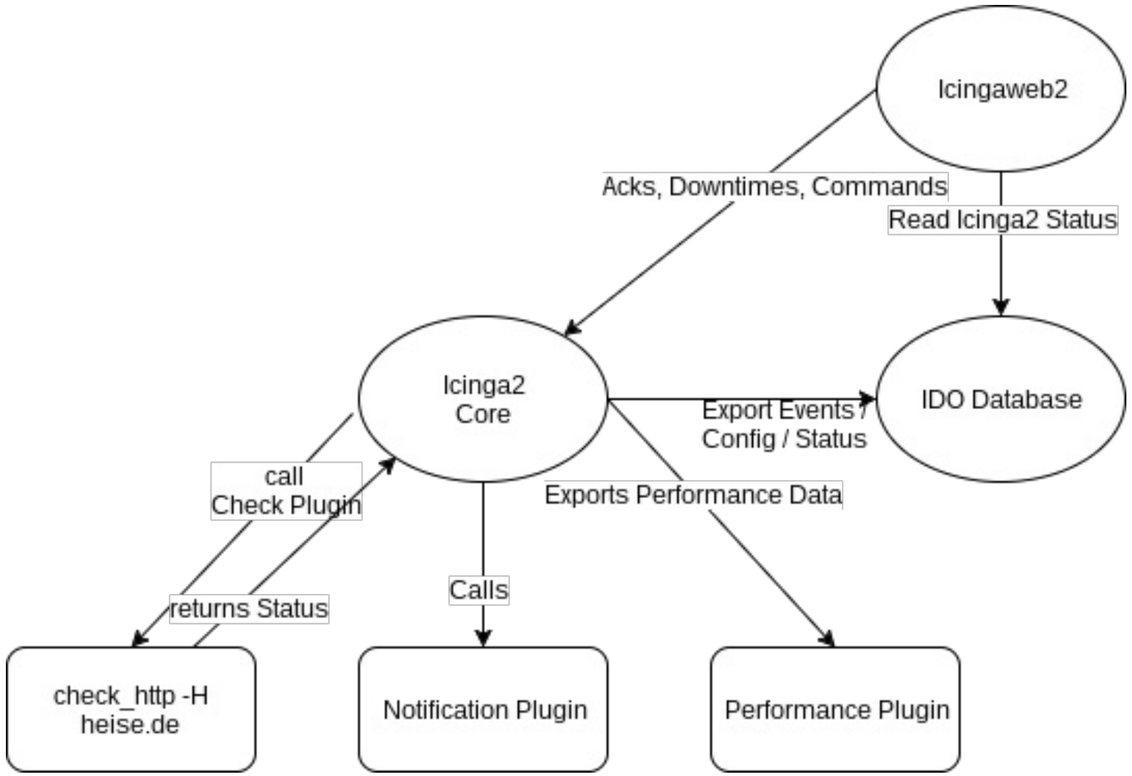
HA MASTER


Load Balanced HA Cluster 1
Failover
DB writing
Failover
Secure zone
IDODB
Runtime
synchronisation
(Heartbeat)
Active
Failover
DB writing
Icinga
Web
Grapher
Master Zone
Checks
Configs
Runtime synchronisation
(Heartbeat)
Alerts
Events
Active
Icinga
Web
Runtime
synchronisation
(Heartbeat)
Satellite Zone
Load Balanced HA Cluster 2
Active
Runtime
synchronisation
(Heartbeat)
Satellite Zone
Load Balanced HA Cluster 3

credativ

GRUNDLAGEN

GRUNDPRINZIP



credativ

HOSTS

- Zu jedem gemonitorten Objekt gehört ein Hostobjekt
- Die Verfügbarkeit des Hosts ergibt sich durch den Status des hostalive checks
- Hosts können den Status **UP**, **DOWN** und **UNREACHABLE** haben

credativ

SERVICES

- Services werden immer einem Host zugeordnet
- Ihr Status ergibt sich aus dem Ergebnis ihres check_commands
- Hosts können den Status **OK**, **WARNING**, **CRITICAL** und **UNKNOWN** haben.

credativ

HARD UND SOFTSTATES

Ein Service Check muss eine bestimmte Anzahl (max_check_attempts) von Ergebnissen durchlaufen, bevor Notifizierungen ausgelöst werden und der **HARD** Status erreicht wird.

credativ

KONFIGURATIONSDATEIEN

credativ

VERZEICHNISSTRUKTUR

```
icinga2
├── conf.d
│   ├── app.conf
│   ├── apt.conf
│   ├── commands.conf
│   ├── downtimes.conf
│   ├── groups.conf
│   ├── hosts.conf
│   ├── notifications.conf
│   ├── satellite.conf
│   ├── services.conf
│   ├── templates.conf
│   ├── timeperiods.conf
│   └── users.conf
├── constants.conf
├── features-available
│   ├── api.conf
│   ├── checker.conf
│   ├── command.conf
│   ├── compatlog.conf
│   ├── debuglog.conf
│   ├── gelf.conf
│   ├── graphite.conf
│   ├── icingastatus.conf
│   ├── ido-mysql.conf
│   ├── influxdb.conf
│   ├── livestatus.conf
│   ├── mainlog.conf
│   ├── notification.conf
│   ├── opentsdb.conf
│   ├── perfdata.conf
```

credativ

HOST OBJEKT

```
object Host "myhostname" {
  /* Import the default host template defined in `templates.conf`. */
  import "generic-host"

  /* Specify the address attributes for checks e.g. `ssh` or `http`. */
  address = "127.0.0.1"
  address6 = "::1"

  /* Set custom attribute `os` for hostgroup assignment in `groups.conf`. */
  vars.os = "Linux"

  /* Define http vhost attributes for service apply rules in `services.conf`. */
  vars.http_vhosts["http"] = {
    http_uri = "/"
  }

  /* Define disks and attributes for service apply rules in `services.conf`. */
  vars.disks["disk"] = {
    /* No parameters. */
  }
  vars.disks["disk /"] = {
    disk_partitions = "/"
  }

  /* Define notification mail attributes for notification apply rules in `notifications.conf`. */
  vars.notification["mail"] = {
    /* The UserGroup `icingaadmins` is defined in `users.conf`. */
    groups = [ "icingaadmins" ]
  }
}
```

credativ

SIMPLE SERVICE

```
apply Service "procs" {
  import "generic-service"

  check_command = "procs"

  assign where host.name == "myhostname"
}
```

CHECK_PROCS

credativ

OVERRIDE VARIABLES IN SERVICE

```
apply Service "procs" {
  import "generic-service"

  check_command = "procs"

  vars.procs_argument = "some argument"
  assign where host.name == "myhostname"
}
```

CUSTOM ATTRIBUTES AND MACROS

credativ

KONFIGURATIONSSPRACHE

credativ

KOMMENTARE

```
// Kommentar
/*
 * Auch ein Kommentar
*/
# sogar ich bin ein Kommentar
```

EINFACHE DATENTYPEN

```
27.3 # floating point number
```

```
2.5m # duration (ms (milliseconds),
       s (seconds),
       m (minutes),
       h (hours) and
       d (days))
```

```
"Hello World" # string
{{{
  Multi
  Line
  String
}}}
```

```
true / false # boolean
```

```
null
```

credativ

ARRAYS

```
[ "hello", 42 ]
```

credativ

DICTIONARIES

```
{
  address = "192.168.0.1"
  port = 443
  ssl = true
}
```

credativ

APPLY DIREKTIVE

- Die Apply Direktive wendet Objekte auf andere Objekte an
- Services auf Hosts
- Notifications auf Services/Hosts
- Abhängigkeiten
- Scope wird durch die assign und ignore Regeln bestimmt

```
object HostGroup "postgresql-server" {
  display_name = "PostgreSQL Server"

  assign where match("*psql*", host.name)
  ignore where match("*test", host.name)
}
```

credativ

```
apply Notification "notify-complex-customer" to Service {
  import "compex-customer-notification"

  assign where match("*has gold support 24x7*", service.notes) \\
    && (host.vars.customer == "customer-xy" || host.vars.always_notify == true)
  ignore where match("*internal", host.name) || \\
    (service.vars.priority < 2 && host.vars.is_clustered == true)
}
```

credativ

APPLY FOR DIREKTIVE

- Anwenden von Objekten auf Basis von Listen or Dictionaries
- Ermöglicht komplexe Objekterstellung

credativ

MIT LISTEN

```
object Host "hostname" {
  ...
  vars.partitions = [ '/', '/boot' ]
}

apply Service "partition " for (partition in host.vars.partitions) {
        import "generic-service"
        check_command = "disk"
        vars.disk_partition = partition
        display_name = "Partition " + partition
        assign where host.vars.partitions
}
```

credativ

MIT DICTIONARIES

```
object Host "hostname" {
  ...
  vars.http_vhosts["http"] = {
    http_uri = "/"
    http_ssl = true
  }
}

apply Service "vhost_" for (vhost => config in host.vars.http_vhosts) {
        import "generic-service"
        check_command = "http"
        vars += config
        display_name = "Virtual Host " + vhost
}
```

credativ

CHECKS

credativ

BASICS

- Icinga2 selbst liefert keine Checks mit
- Alle Checks die mit Nagios kompatibel sind werden auch mit Icinga2 funktionieren
- Viele Checks sind in der ITL vordefiniert

credativ

MONITORING PLUGINS

- **Die** Standard Checks für Nagios/Icinga/Icinga2
- monitoring-plugins.org

credativ

ICINGA EXCHANGE

- Portal für Checks

credativ

FUNKTIONSWEISE VON PLUGINS

- Icinga interpretiert den Returnstatus eines Plugins

RC	Result
0	OK / UP
1	WARN / UP
2	CRITICAL / DOWN
3	UNKNOWN / DOWN

- Ausgabe von Plugins wird ignoriert
- Sollte den Monitoring Plugin Development Guidelines folgen
- Plugins werden grundsätzlich ohne Shell aufgerufen

CHECK COMMAND

```
object CheckCommand "check_http" {
  command = [ PluginDir + "/check_http" ]

  arguments = {
    "-H" = "$http_vhost$"
    "-I" = "$http_address$"
    "-u" = "$http_uri$"
    "-p" = "$http_port$"
    "-S" = {
      set_if = "$http_ssl$"
    }
    "--sni" = {
      set_if = "$http_sni$"
    }
    "-a" = {
```

credativ

PASSIVE CHECKS

credativ

- Passive Checks führen selbst keine aktiven Tests aus
- Sie erwarten Ergebnisse von außen (API/Commandsocket)
- Über freshness Checks kann beim ausbleiben von Ergebnissen ein Alarm ausgelöst werden (last check + check interval)

credativ

```
object Service "Passive" {
  import "generic-service-custom"

  host_name             = "host.domain.com"
  check_command         = "passive"

  enable_notifications  = 1
  enable_active_checks  = 1
  enable_passive_checks = 1
  max_check_attempts    = 1
  check_interval        = 87000
}
```

credativ

NOTIFICATIONS

- Notifications sind Scripte, die bei bestimmten Events aufgerufen werden
- Notifications können an einen User und/oder eine Gruppe gebunden werden.
- Jede Notification kann einen Filter haben, der bestimmt zu welchen Zeiten sie ausgelöst wird
- Jeder User kann einen Filter haben, der bestimmt wann er notifiziert werden will.
- Über types und states Filter kann man einstellen bei welchen Events die Notification auslöst.

credativ

NOTIFICATION BEISPIEL

```
object Notification "localhost-ping-notification" {
  host_name = "localhost"
  service_name = "ping4"

  command = "mail-notification"

  users = [ "user1", "user2" ]

  types = [ Problem, Recovery ]
}
```

credativ

NOTIFICATION COMMAND

```
object NotificationCommand "mail-host-notification" {
  import "plugin-notification-command"

  command = [ SysconfDir + "/icinga2/scripts/mail-host-notification.sh" ]

  env = {
    NOTIFICATIONTYPE = "$notification.type$"
    HOSTALIAS = "$host.display_name$"
    HOSTADDRESS = "$address$"
    HOSTSTATE = "$host.state$"
    LONGDATETIME = "$icinga.long_date_time$"
    HOSTOUTPUT = "$host.output$"
    NOTIFICATIONAUTHORNAME = "$notification.author$"
    NOTIFICATIONCOMMENT = "$notification.comment$"
    HOSTDISPLAYNAME = "$host.display_name$"
```

NOTIFICATION SCRIPT

```
#!/bin/sh
template=`cat <<TEMPLATE
***** Icinga  *****

Notification Type: $NOTIFICATIONTYPE

Service: $SERVICEDESC
Host: $HOSTALIAS
Address: $HOSTADDRESS
State: $SERVICESTATE

Date/Time: $LONGDATETIME

Additional Info: $SERVICEOUTPUT
```

credativ

ESKALATIONEN

- Eskalationen ermöglichen es Notifications nach einer bestimmten Zeit zu eskalieren
- Mehrere Eskalationen können gleichzeitig und überlappend aktiv sein

credativ

EIN BEISPIEL

Wenn nach 10 Minuten keiner auf den Alarm per E-Mail reagiert hat, wird eine SMS an das Team herausgeschickt. Sollte nach weiteren 20 Minuten auch hier keiner reagieren erzeugt Asterisk mit Sprachsynthese einen bösen Anruf an den Vorgesetzen.

credativ

KONFIGURATION

```
object User "first_alert" {
  display_name = "First Alert by Mail"
  vars.email = "it@localhost"
}
object User "second_alert" {
  display_name = "Second Alert by SMS"

  vars.mobile = "+49 1723 1234567"
}
apply Notification "first_alert" to Service {
  import "generic-notification"
  command = "mail-notification"
  users = [ "first_alert" ]
  times = {
    begin = 0m
```

credativ

DOWNTIMES

- Downtimes ermöglichen es Ausnahmezeiten festzulegen in denen keine Notifizierungen erfolgen sollen
- z.B.: (un)geplante Wartungen, Zeiten mit hoher Last

credativ

FIXED DOWNTIMES

- Beginnen und enden zu einem festgelegten Zeitpunkt

FLEXIBLE DOWNTIMES

- Flexibles Fenster
- Beginnt erst ab eintreten eines Events während der Downtime Periode
- Endet nach eintreten des Events und der definierten Downtime Dauer

```
apply ScheduledDowntime "backup-downtime" to Service {
  author = "icingaadmin"
  comment = "Scheduled downtime for backup"

  ranges = {
    monday = "02:00-03:00"
    tuesday = "02:00-03:00"
    wednesday = "02:00-03:00"
    thursday = "02:00-03:00"
    friday = "02:00-03:00"
    saturday = "02:00-03:00"
    sunday = "02:00-03:00"
  }

  assign where "backup" in service.groups
```

credativ

DEPENDENCIES

- Abhängigkeiten werden benutzt, um Beziehungen zwischen Hosts und/oder Services auszudrücken
- Abhängigkeiten werden zur Erreichbarkeitsberechnung benutzt
- Abhängige Services / Hosts erzeugen keine Notifications, wenn ihr Parent nicht erreichbar ist
- zwischen einem Host und seinen Services besteht eine implizite Abhängigkeit
- Abhängigkeiten reagieren nur auf Hard States (ausser ignore_soft_states ist false)

credativ

```
apply Dependency "internet" to Host {
  parent_host_name = "dsl-router"
  disable_checks = true
  disable_notifications = true

  assign where host.name != "router"
}

apply Dependency "disable-host-service-checks" to Service {
  disable_checks = true
  assign where true
}
```

credativ

ACKNOWLEDGMENTS

- Ein Acknowledgment zeigt das man einen Alarm zur Kenntniss genommen hat
- Nach einem Acknowledge erfolgen keine weiteren Alarme
- Das Webfrontend sortiert die bestätigten Alarme hinter die unbestätigten Alarme
- **Sticky** Acknowledgement: das Acknowledgment verschwindet erst bei einem **OK** Status
- **Expiring** Ack: Wenn sich um das Problem nicht nach einer bestimmten Zeit gekümmert wurde, wird die Bestätigung entfernt

credativ

AGENTEN

credativ

ICINGA2 AS SATELLITE

- Icinga2 im Clustermode wird als Satellit betrieben
- Konfigurationsdateien werden über Zonensynchronisation verteilt
- Checks werden vom **Satelliten** gescheduled und ausgeführt

ICINGA2 AS REMOTE COMMAND EXCEUTION BRIDGE

- Icinga2 wird im Cluster mode als Command Endpoint betrieben
- Check Commands werden über Zonensynchronisation verteilt
- Checks werden vom **Master** gescheduled aber vom **Agent** ausgeführt
- Remote Agents werden über den command_endpoint angesprochen

credativ

KONFIGURATIONSBEISPIEL

```
object Host "myhost" {
  ...
  vars.remote_client = "myhost"
}

apply Service "apt" {
  import "generic-service"
  check_command = "apt"
  if (host.vars.remote_client) {
    command_endpoint = host.vars.remote_client
  }
  assign where host.vars.distribution == "Debian"
}
```

credativ

PERFORMANCE DATEN

- Performance Daten können benutzt werden damit **externe** Systeme diese auswerten können
- Performance Daten sind maschinenlesbare Ausgaben der Checkergebnisse
- Sie werden insbesondere für Trending benutzt
- Die Daten werden **nicht** von Icinga für Monitoring benutzt
- Definition

```
 /usr/lib/nagios/plugins/check_disk -u GB -w 10% -c 5% /
DISK OK - free space: / 43 GB (18% inode=86%);| /=192GB;212;224;0;236
```

credativ

POPULÄRE SYSTEME

- InfluxDB + Grafana
- OpenTSDB + Grafana
- Graphite + Grafana
- pnp4nagios

credativ

DISTRIBUTED MONITORING

credativ

ZONEN

- Zonen ermöglichen es mehr als einen Endpunkt für einen bestimmten Bereich zu haben
- Mehr als einen Endpunkt in einer Parent Zone -> Hochverfügbarkeit
- Mehr als einen Endpunkt in einer Child Zone -> Lastverteilung
- HA Features ermöglichen es Features wie IDO oder Notifications hochverfügbar zu betreiben

credativ

SCHNITTSTELLEN

credativ

ICINGA2 API

- JSON/REST basierte API
- Objekte können:
  - ausgelesen
  - modifiziert und
  - erzeugt werden
- Authentifikation via Username/Passwort oder X509 Clientcert
- Permissionsystem

BEISPIELE

DELETE A SERVICE

```
curl -u $ICINGA2_API_USER:$ICINGA2_API_PASSWORD  \
    -H 'Accept: application/json' -H 'X-HTTP-Method-Override: DELETE' -X POST \
    -k "https://$ICINGA2_HOST:$ICINGA2_API_PORT/v1/objects/services/api_dumm
```

KASKADIERTES LÖSCHEN EINES HOSTS

```
curl -u $ICINGA2_API_USER:$ICINGA2_API_PASSWORD \
     -H 'Accept: application/json' -H 'X-HTTP-Method-Override: DELETE' -X POST \
     -k "https://$ICINGA2_HOST:$ICINGA2_API_PORT/v1/objects/hosts/api_dummy_
```

COMMAND SOCKET

- Kommunikation mit Icinga2 über einen Unix Socket
- von Nagios *geerbt*
- kann benutzt werden um bestimmte Aktionen durchzuführen
- kann nur Acks und Downtimes anlegen

START SERVICE CHECKS

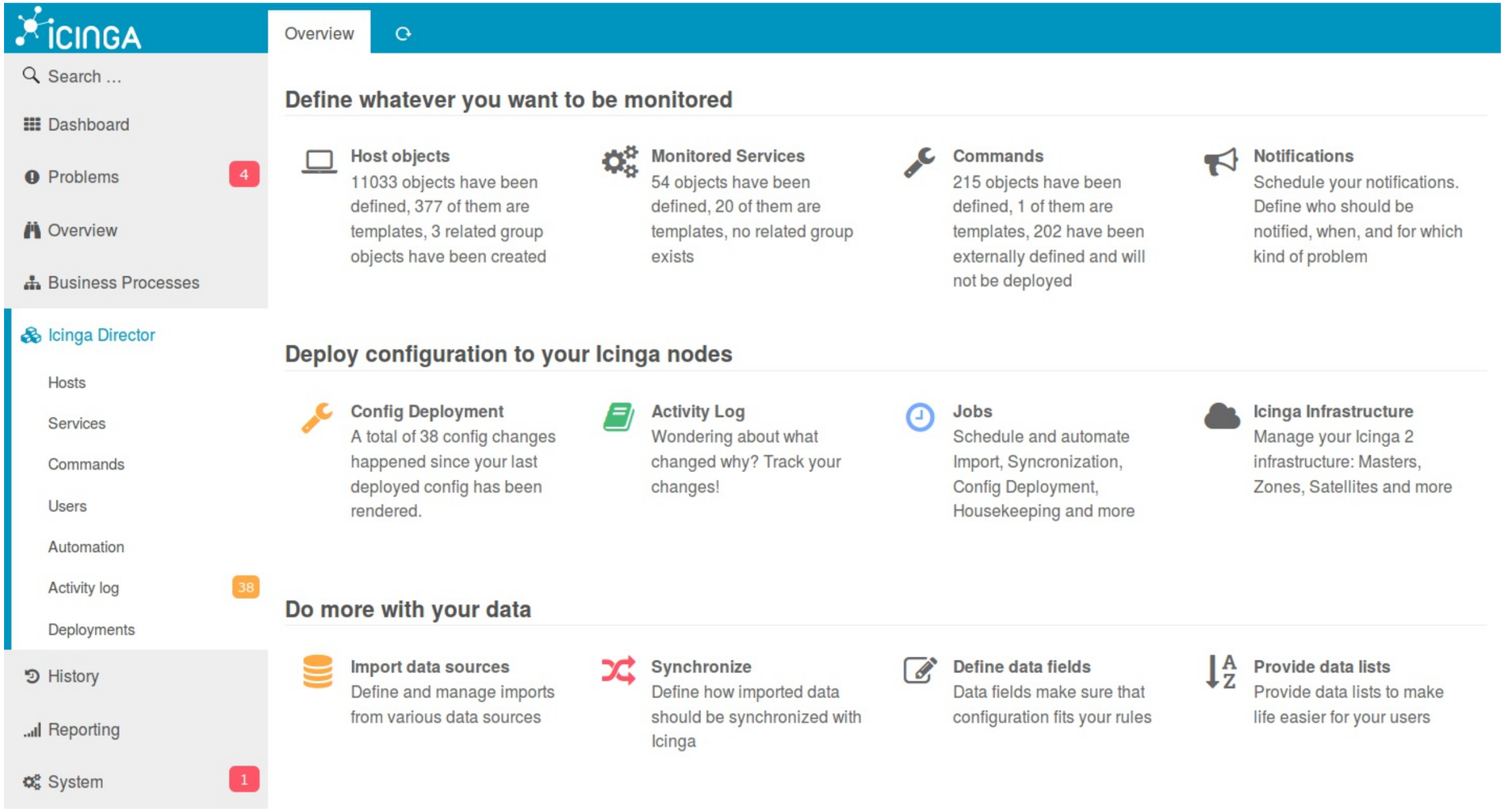
```
now=`date +%s`
/bin/printf "[%lu] START_EXECUTING_SVC_CHECKS\n" $now > $commandfile
```

credativ

ERWEITERUNGEN / INTEGRATION

ICINGA-WEB2-DIRECTOR

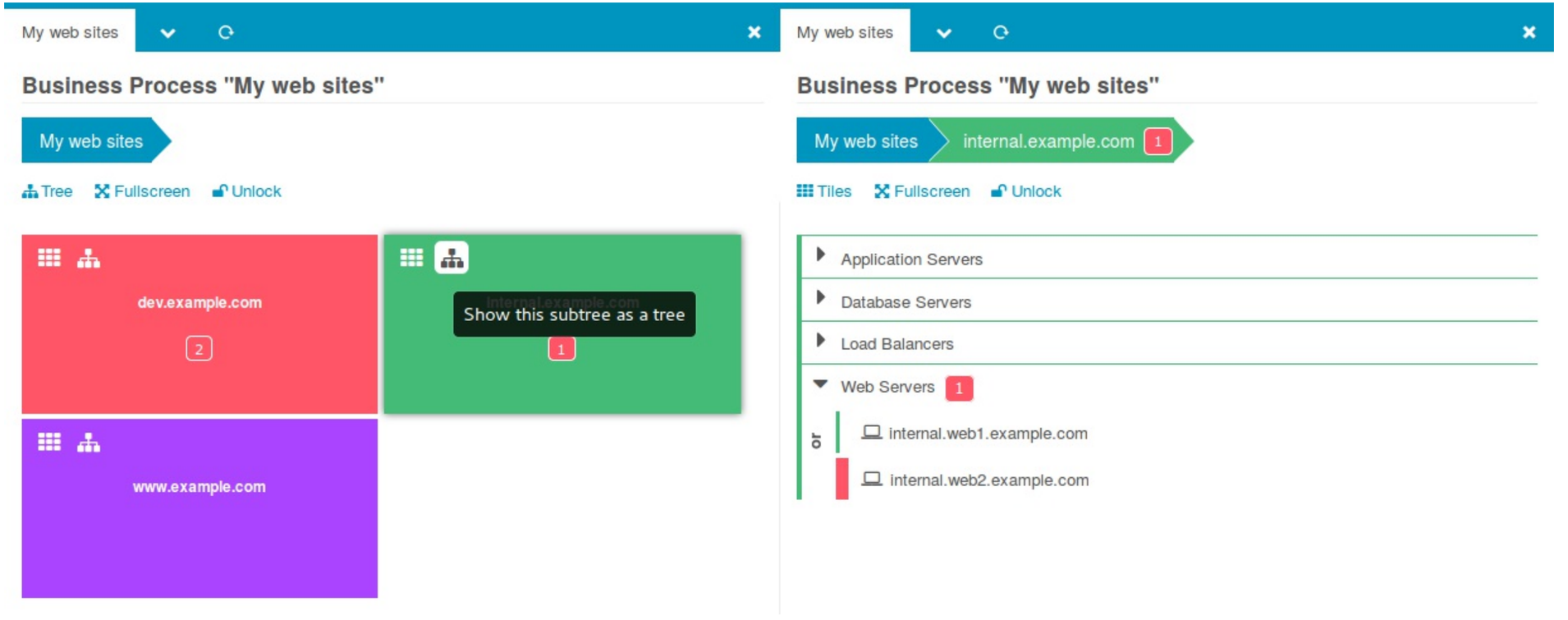
- Icingaweb2 Modul
- *Konfigurationswebfrontend* für Icinga2
- Module für den Import von verschiedenen Quellen:
  - CSV
  - Puppetdb
  - LDAP
  - Dateien
  - AWS
- REST API


ICINGA
Search ...
Dashboard
Problems 4
Overview
Business Processes
Icinga Director
Hosts
Services
Commands
Users
Automation
Activity log 38
Deployments
History
Reporting
System 1
Overview
Define whatever you want to be monitored
Host objects
11033 objects have been defined, 377 of them are templates, 3 related group objects have been created
Monitored Services
54 objects have been defined, 20 of them are templates, no related group exists
Commands
215 objects have been defined, 1 of them are templates, 202 have been externally defined and will not be deployed
Notifications
Schedule your notifications. Define who should be notified, when, and for which kind of problem
Deploy configuration to your Icinga nodes
Config Deployment
A total of 38 config changes happened since your last deployed config has been rendered.
Activity Log
Wondering about what changed why? Track your changes!
Jobs
Schedule and automate Import, Syncronization, Config Deployment, Housekeeping and more
Icinga Infrastructure
Manage your Icinga 2 infrastructure: Masters, Zones, Satellites and more
Do more with your data
Import data sources
Define and manage imports from various data sources
Synchronize
Define how imported data should be synchronized with Icinga
Define data fields
Data fields make sure that configuration fits your rules
Provide data lists
Provide data lists to make life easier for your users

credativ

ICINGAWEB2-BUSINESS-PROCESS-VIEW

- Modellierung von Business Views
- *Manager* Kompatibilität
- Simulation von Incidents
- webbasierter Editor


My web sites
Business Process "My web sites"
My web sites
Tree
Fullscreen
Unlock
dev.example.com
2
Show this subtree as a tree
1
www.example.com
My web sites
Business Process "My web sites"
My web sites
internal.example.com
1
Tiles
Fullscreen
Unlock
Application Servers
Database Servers
Load Balancers
Web Servers
1
or
internal.web1.example.com
internal.web2.example.com

credativ

SLACK INTEGRATION

- formorer/icinga2-slack-notification
- spjmurray/slack-icinga2 (2way)
- richardhauswald/icinga2-slack-notifications

credativ

ICINGA2 DASHING

- Sinatra/Dashing basiertes Dashboard
- Benutzt die API


192.168.33.5:8005/icinga2
ICINGA
Host Problems
2
Total hosts: 5
Last updated at 11:10
Service Problems
11
Total services: 49
Last updated at 11:10
Checks (1 min)
Host (active) 2
Service (active) 126
Avg latency: 0.0s
Last updated at 11:10
Problems
c2-mysql-2 - ping4
c2-mysql-2 - ssh
c2-mysql-2 - bp-mysql-connection-time
c2-mysql-2 - bp-mysql-open-files
c2-web-1 - bp-http-response
Last updated at 11:10
Hosts Down
2
Last updated at 11:10
Services Critical
7
Last updated at 11:10
Services Warning
2
Last updated at 11:10
Services Unknown
2
Last updated at 11:10
Hosts Ack
0
Last updated at 11:10
Services Ack
0
Last updated at 11:10
Hosts Downtime
0
Last updated at 11:10
Services Downtime
0
Last updated at 11:10
DOWN Nov 27 c2-mysql-2 PING CRITICAL - Packet loss = 100%
DOWN Nov 27 c2-web-1 PING CRITICAL - Packet loss = 100%
CRITICAL 0m 9s Soft 1/3 icinga2: random-005 Hello from icinga2
CRITICAL 0m 19s icinga2: random-004 Hello from icinga2
UNKNOWN 0m 11s Soft 2/3 icinga2: random-001 Hello from icinga2
WARNING 0m 9s Soft 1/3 icinga2: random-003 Hello from icinga2

credativ

BONUS: ABGEFAHRENES ZEUG

credativ

CLUSTERCHECKS IN ICINGA2 DSL

```
object CheckCommand "clustercheck" {
  import "plugin-check-command"
  command = {{
    var cmd = [ PluginDir + "/check_dummy" ]
    var cluster_nodes = macro("$cluster_nodes$")
    var host_failed = [ ]
    for (cluster_node in cluster_nodes) {
      log("Check state of " + cluster_node)
      /* get host state from node */
      var host = get_host(cluster_node)
      /* wenn der Host kritisch ist zählen wir den Fehlerwert hoch
        und speichern den kritischen Host in einer Variable */
      if (host.state_raw != 0 && host.state_raw != 1) {
        log(cluster_node + " failed")
        host_failed += [ cluster_node ]
      }
    }
    if (len(host_failed) >= number(macro("$cluster_critical$"))) {
      cmd += [ "2", "Cluster ist in einem kritischen Zustand. Kritische Hosts: " + host_failed.join(",") ]
    } else {
      if (len(host_failed) >= 1) {
        cmd += [ "0", "Cluster hat Probleme. Kritische Hosts: " + host_failed.join(",") ]
      } else {
        cmd += [0, "Cluster ist OK"]
      }
    }
    return cmd
  }}
}
object Host "node1" {
```

credativ

DYNAMISCHE OBJEKTERSTELLUNG

```
/**
 * define customerenvs - projectname, customername, customerdomainpart
 */

vars.custenvs.swisslitest01 = { customername = "customer swissli", domainpart = "swissli-test" }
vars.custenvs.gipfelitest01 = { customername = "customer gipfeli", domainpart = "gipfli-test" }
vars.custenvs.schwingerlitest01 = { customername = "customer schwingerli", domainpart = "schwingerli-test" }

for (custenv_name => config in vars.custenvs) {
  object Service "shibboleth-http-external" use(custenv_name, config){
  vars += config
    import "remote-service"
    display_name = "Shibboleth HTTP service (external)"
    check_command = "http"
    vars.http_address = "idp." + vars.domainpart + ".external.ch"
    vars.http_vhost = "idp." + vars.domainpart + ".external.ch"
    vars.http_port = "443"
    vars.http_ssl = "true"
    vars.http_uri = "/idp/"
    vars.http_expect = "HTTP/1.1 200"
    host_name = custenv_name + "-app09-shibidp01.int.net"
    check_interval = 15m
    retry_interval = 5m
  }
}


```