OPEN SOURCE AUTOMATION DAY

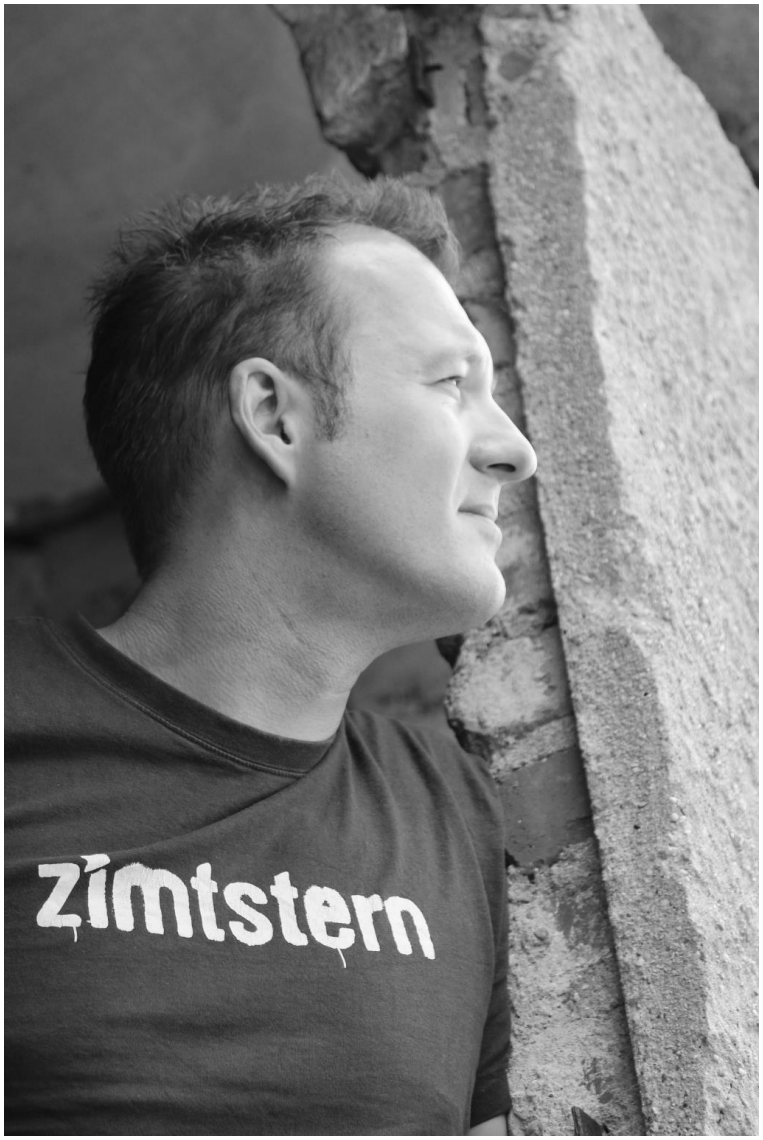
Müller

„Infrastructure as Code“ oder Warum OPS Automatisierung benötigen

22.10.2018

Jens Schanz

Über mich



Jens Schanz

- Beruflich
- Teamleiter 2nd Level Support „Filialsysteme“
- Linux- / Unix-Admin seit 1999
- Infrastruktur-Architekt
- Fauler Sack (Monitoring- und Automatisierungs

- Privat
- Verheiratet, 2 Kinder, keine Zeit
- Smarthome-Builder

Müller in Zahlen

Firmenname: Müller Holding Ltd. & Co. KG

Firmensitz (Verwaltung): 89081 Ulm-Jungingen,

Albstraße 92

Geschäftsführer: Erwin Müller

Zahl der Mitarbeiter: rund 35.000 (überwiegend Fachkräfte)

Zahl der Auszubildenden: rund 950

Gesamtzahl Filialen: derzeit 828, davon 540 in Deutschland, 55 in der Schweiz, 81 in Österreich, 12 in Spanien, 18 in Slowenien, 36 in Ungarn, 86 in Kroatien

Filialen mit Naturshop: derzeit 210 davon 132 in Deutschland, 8 in der Schweiz, 37 in Österreich, 8 in Slowenien 13 in Ungarn und 12 in Kroatien

Filialgröße: 400 bis über 4.500 m² Verkaufsfläche

Gesamtlagerfläche: 246.416 m², davon 17.998 m² Lager Ungarn, 9.251 m² Lager Schweiz, 1.080 m² Lager Spanien

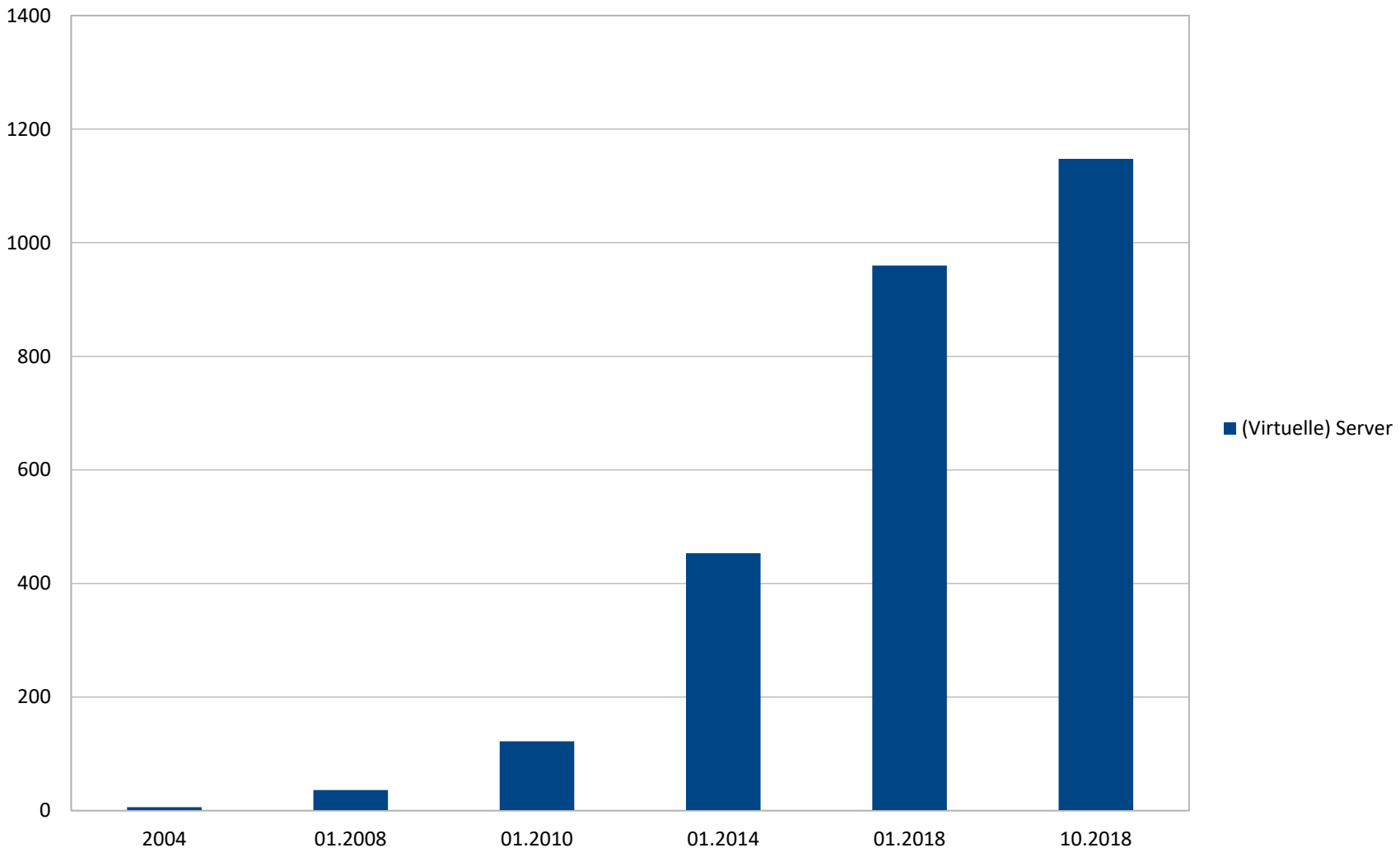
Abteilungen / Fachmärkte

- Drogerie (ca. 50.000 Artikel)
- Multi-Media (ca. 42.000 Artikel)
- Parfümerie (ca. 28.000 Artikel)
- Spielwaren (ca. 20.000 Artikel)
- Schreibwaren (ca. 19.000 Artikel)
- Haushalt und Ambiente (ca. 11.000 Artikel)
- Strümpfe (ca. 7.500 Artikel)
- Naturkosmetik (ca. 4.000 Artikel)
- Handarbeit (ca. 2.400 Artikel)
- OTC (ca. 1.500 Artikel)
- Bio Nahrung (ca. 3.000 Artikel)

Sortimentsvielfalt:

ca. 188.000 Artikel

Müller-Filial-Welt in Zahlen – Server



Realität

Ursachen

Ursachen

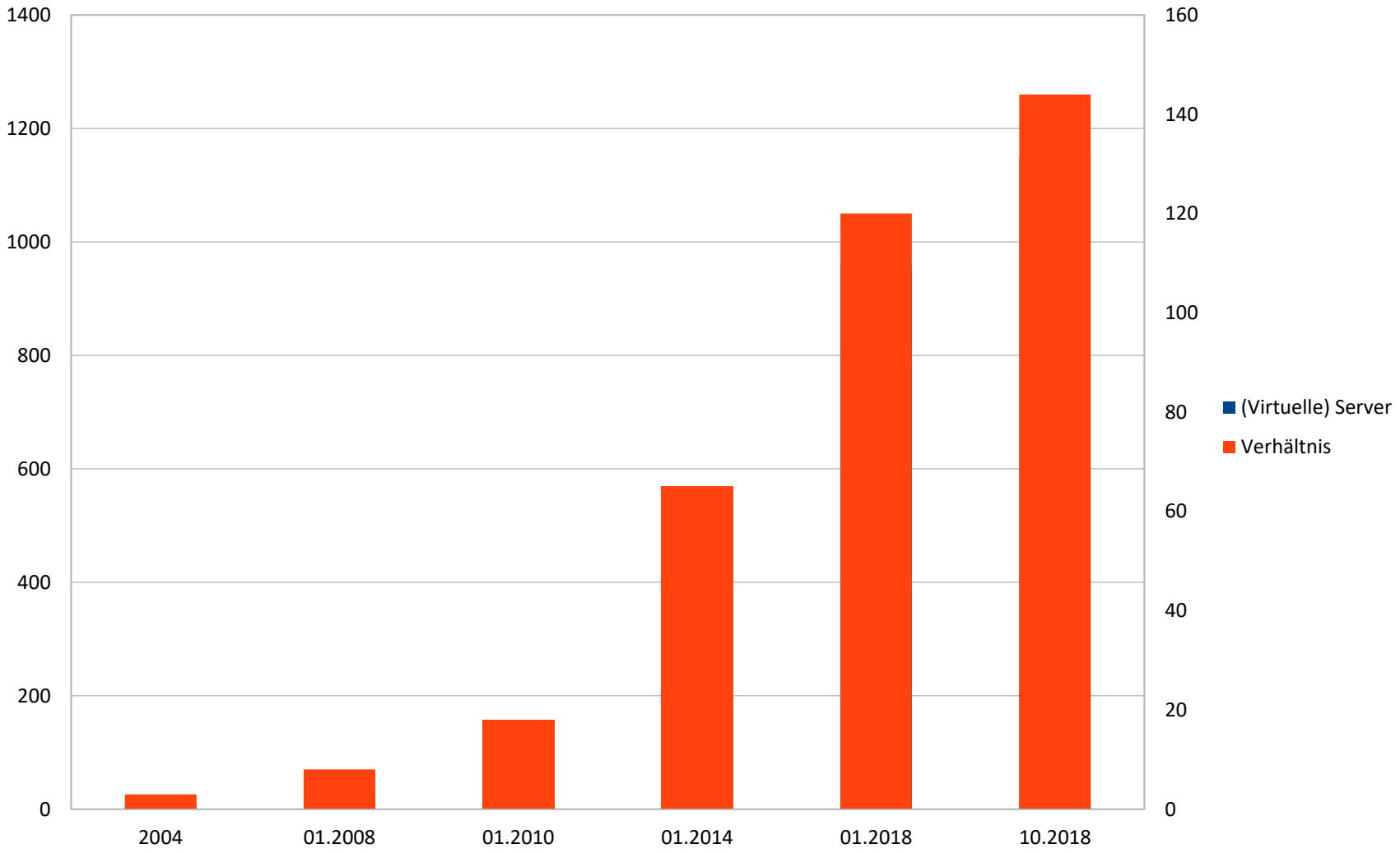
- **Ursachen**
  - Wandel der IT durch
  - „Scale-Up vs. Scale-Out“
  - Cloud-Technologien
  - Virtualisierungsfarmen
  - „Snowflake“-Server
  - Erosion der Infrastruktur
  - Konfigurationsprobleme

Ursachen

- **Scale-Up vs. Scale-Out**
  - Scale-Up funktioniert nur begrenzt
  - Scale-Out funktioniert hingegen nahezu unendlich

- **Scale-Out bedeutet aber auch**
  - Mehr Systeme
  - Mehr Aufwand
  - Mehr Fehler
  - Mehr Probleme
  - … und vor allem: **„Weniger Zeit!“**

Ursachen – Wachstum und Zeit



Ursachen

- **„Snowflake“-Server**
  - *A snowflake server is different from any other server on your network. It‘s special in ways that can‘t be replicated.*

  Dont‘t touch that server. Don‘t point at it. Don‘t even look at it.

  - Kief Morris -

- **Summe der „Snowflake“-Server**
  - Fragile Infrastruktur

Ursachen

- **Konfigurationsprobleme**
  - Manuelle Änderungen an Konfigurationsdateien
  - Konfigurationsfehler (Schreibfehler, Syntaxfehler, …)
  - Konfiguration ist nicht persistent

Ursachen

- **Erosion der Infrastruktur**
  - Betriebssystem-Upgrades, Kernel-Patches
  - Software-Upgrades (OpenSSL, SSH, MariaDB, …)
  - Logfiles
  - Cron-Jobs
  - Manuelle Eingriffe aller Art

Ursachen

Lösungsansätze

Lösungsansätze

- **Automatisierte Systemadministration**
  - Reproduzierbar
  - Parallelisierbar
  - Orchestrierbar

Lösungsansätze

- **Automatisierung**
  - „Script all the things“-Strategie
  - Baseline über Templates
  - Automatische Provisionierung neuer Systeme auf Knopfdruck (Golden Image, debootstrap, Autoyast, ...)
  - Automatische Konfiguration ALLER Systeme (z.B. Puppet, Chef, …)
  - Einheitliche Build- und Deployment-Prozesse

Lösungsansätze

- **Automatisierung**
  - Systeme werden und sind dadurch einfach reproduzierbar
  - Systeme werden und sind dadurch „einfach“ ersetzbar

„Wenn du SSH benutzt um ein System manuell zu konfigurieren, kannst du auch gleich ein Puppet-Modul oder einen Ansible-Task dafür schreiben!“

Lösungsansätze

- **Transparenz**
  - Konfiguration der Infrastruktur ist sichtbar und kann nachvollzogen werden → Messbar!
  - Jeder kann Änderungen an der Infrastruktur vornehmen, sofern er autorisiert ist.
- **Kultur und Blickwinkel verändern**
  - „Cattle And Cows“ anstelle „Pets and cats“
  - „Alle reden von Backup, keiner von Restore“
  - Trennung Applikation und Nutzdaten

Lösungsansätze

- **Regeln**

  - **#1**
  - **„If you break it, will you notice it“**

  - **#2**
  - **„If you break it, can you fix it“**

Lösungsansätze

- **Keep-It-Simple**
  - Möglichst wenig Abstraktionsschichten
- Kein anderer Bereich stapelt mehr Abstraktionsebenen über- und nebeneinander als die IT
  - In nicht linear skalierenden Systemen
  - Mit kaskadierenden Abhängigkeiten (horizontal und vertikal)
- Mit jedem Abstraktionslayer wird das ganze noch undurchschaubarer
- Irgendjemand muss das im Fehlerfall debuggen (können)

Lösungsansätze

- **Keep-It-Simple**
  - Möglichst wenig unterschiedliche Technologien
  - Keine Software mit Hipster-Bingo-Abhängigkeiten aus unbekannten Quellen nutzen

  - *„Schreibe Computerprogramme so, dass sie nur eine Aufgabe erledigen und diese gut machen.“*

  – Douglas McIlroy -

Lösungsansätze

- **Keep-It-Simple**
  - Möglichst bewährte und robuste Technologien nutzen
  - Implementierung und Administration „By Force“
    - „Immer Stecker ziehen ...“
    - „qm stop“ vs. „qm halt“
    - „halt -f“ vs. „shutdown“
    - „kill -9 <pid>“ vs. „kill <pid>“
  - Systeme per Skript zufällig ausschalten (Chaos-Monkey-Prinzip)
- **Warum?**
  - Wenn die Systeme das nicht überleben, hat man im Ernstfall ein richtiges Problem.

Lösungsansätze

- **Keep-It-Simple**
  - Viel Feenstaub unterwegs mit halblebiger Technik

  - *„Die typische Mittelstands-IT ist aber nicht Google oder Facebook mit spezialisierten Infrastruktur- oder DevOps-Teams die jedes Buzz-Word implementieren und betreiben können.“*

Blast from the past

Infrastructure as Code - 2004

3. Konfiguration mit YAST Control Center

→ Security and Users → Local Security → [x] Network Server → Details

- [x] Checking new passwords
- Password length — Minimum: 6
- Password encr. Method — → MD5
- Days of PW change warning — Minimum: 1 — Maximum: 99999
- Days befor PW expires warning — 14
- How to interpret CTRL+ALT+ENTF — Ignore
- Shutdown behaviour of KDM — only root
- Login
  - Seconds wait after incorrect login — 3
  - [x] Record failed login attempts
  - [x] Record successful login attempts
  - [ ] Allow remote graphical login

Die anderen Einstellungen auf Standard belassen

Installationspfad ändern:

(entsprechend der Version 32- oder 64-Bit)

→ Yast -> Software → Installation Source

Add → NFS →

server: server.localdomain

directory: /install/SLES10/SLES10_32bit/CD1

→ OK

Den Zugriff auf die Installationsquelle von CD deaktivieren.

4. Verschiedene Anpassungen

- Datei **/etc/ntp.conf** anpassen

```
##
## Outside source of synchronized time
##
server ntp.localdomain        # IP address or hostname of server
```

Dienst installieren, damit xntpd automatisch in Runlevel 3 und 5 gestartet wird

```
insserv ntp
rcntp start
```

Try to get initial date and time via NTP from ntp.localdomain done
Starting network time protocol daemon (NTPD)

- Hardware – Clock setzen

```
hwclock --systohc
```

SSH Protokoll anpassen

```
vi /etc/ssh/sshd_config

#Port 22
Protocol 2                                  # anpassen
#ListenAddress 0.0.0.0
#ListenAddress ::

rcsshd restart
```

Dienst sysstat automatisch starten

```
chkconfig sysstat on
```

Datei /boot/grub/menu.lst anpassen

```
title SUSE Linux Enterprise Server 10
   root (hd0,4)
   kernel /vmlinuz root=/dev/cciss/c0d0p7 vga=0x317   resume=/dev/cciss/c0d0p6
splash=verbose showopts
   initrd /initrd
```

Datei /etc/sysconfig/boot anpassen

```
vi /etc/sysconfig/boot

## Type:      yesno
## Default:   yes
#
# Run all scripts or rather start/stop all services
# which are independent from each other in parallel.
#
RUN_PARALLEL="no"                                   # in no ändern
```

Infrastructure as Code – 2006

•server.localdomain:/data/vserver_images/install auf Zielserver kopieren

•linux-2.6.17.13.tar.gz in Verzeichnis /usr/src entpacken

•in das Verzeichnis /usr/src/linux-2.6.17.13 wechseln

make -j8 modules_install

make install

mkinitrd -k vmlinuz-2.6.17.13-vs2.0.2.1-smp -i initrd-2.6.17.13-vs2.0.2.1-smp -M /boot/System.map-2.6.17.13-vs2.0.2.1-smp

•evtl. neuen VServer Eintrag in /boot/grub/menu.lst anpassen

title SUSE Linux Enterprise Server 10 Vserver

kernel (hd0,4)/vmlinuz-2.6.17.13-vs2.0.2.1-smp root=/dev/cciss/c0d0p7 vga=0x317 resume=/dev/cciss/c0d0p6 splash=silent showopts

•neuen Eintrag über YAST als Default-Bootoption markieren

•dietlibc_x86_64.tar.gz entpacken

•dietlibs installieren -> cd dietlibc -> install bin-x86_64/diet /usr/local/bin/

•Über YAST folgende Pakete nachinstallieren

Python-devel

Gcc-c++

E2fsprogs-devel

Pkgconfig

•util-vserver rpms installieren -> rpm -Uvh --nodeps *.rpm

•Init-Skripte deaktivieren

chkconfig -d v_gated

chkconfig -d v_httpd

chkconfig -d v_named

chkconfig -d v_portmap

chkconfig -d v_sendmail

chkconfig -d v_smb

chkconfig -d v_xinetd

•Init-Skripte löschen

rm v_gated v_httpd v_named v_portmap v_sendmail v_smb v_xinetd

•SSH-Config anpassen

vi /etc/ssh/sshd_config

Parameter ListenAddress 0.0.0.0 anpassen auf IP-Adresse

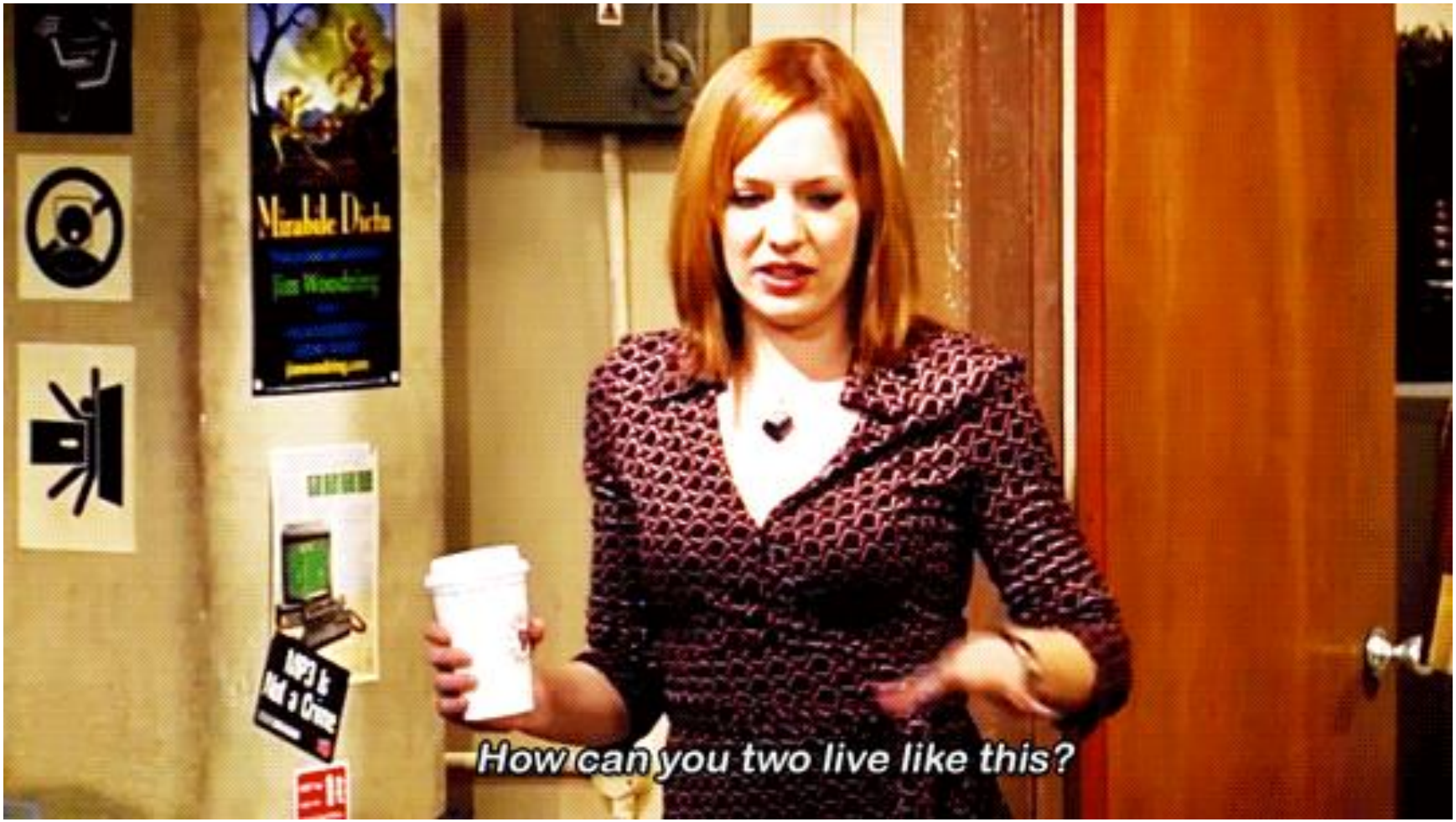
•Portmap-Config anpassen

vi /etc/init.d/portmap

Zeile "startproc $PORTMAP_BIN" in "chbind --ip <eigeneIp> startproc $PORTMAP_BIN" ändern

•Überflüssige Dienste deaktivieren#

Infrastructure as Code – Blast from the past

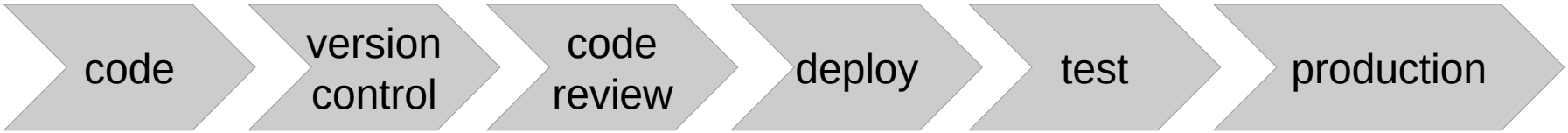


„Infrastructure as Code“

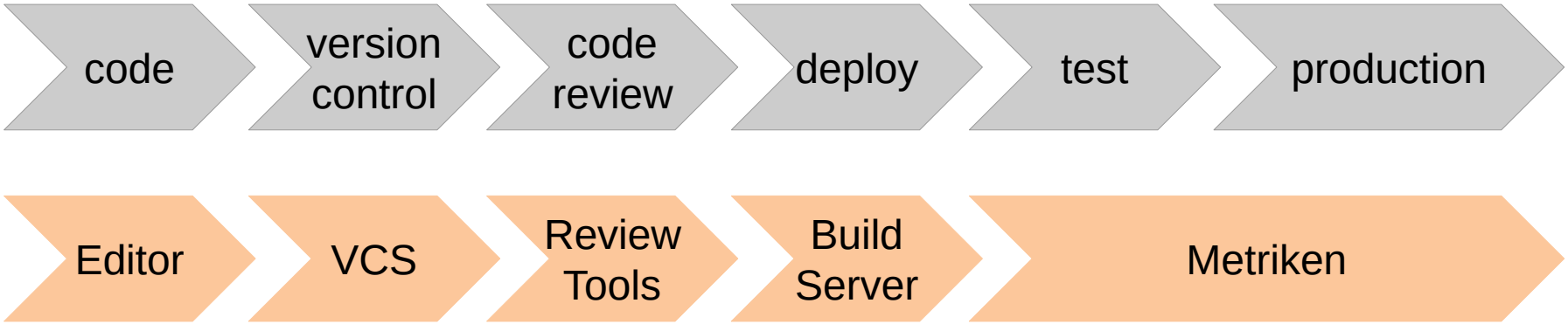
- **Was ist Infrastructure as Code?**
  - „Infrastructure as Code“ ist eine Methode um IT-Prozesse zu automatisieren. Sie wird hauptsächlich dazu verwendet, um eine Reihe statischer Schritte mehrmals auf verschiedenen Servern, mit demselben Ergebnis, zu wiederholen.
  - „Infrastructure as Code“ verwendet meist High-Level- oder deskriptive Sprachen um die Bereitstellung auf den unterschiedlichen Betriebssystemen zu ermöglichen und zu vereinfachen.

- **Wie funktioniert „Infrastructure as Code“**
  - „Infrastructure as Code“ übernimmt etablierte Software-Entwicklungs-Mechanismen und überträgt diese in die Welt der „Systemadministration“.
  - „Infrastructure as Code“ kann mit dem entsprechenden Tool z.B. einen MariaDB-Server vollständig automatisch installieren, prüfen ob der Prozess läuft, entsprechende Benutzer und Datenbanken anlegen – alles per Code. <u>Wiederholbar und beliebig oft und zu jeder Uhrzeit.</u>

Infrastructure as Code - Workflow

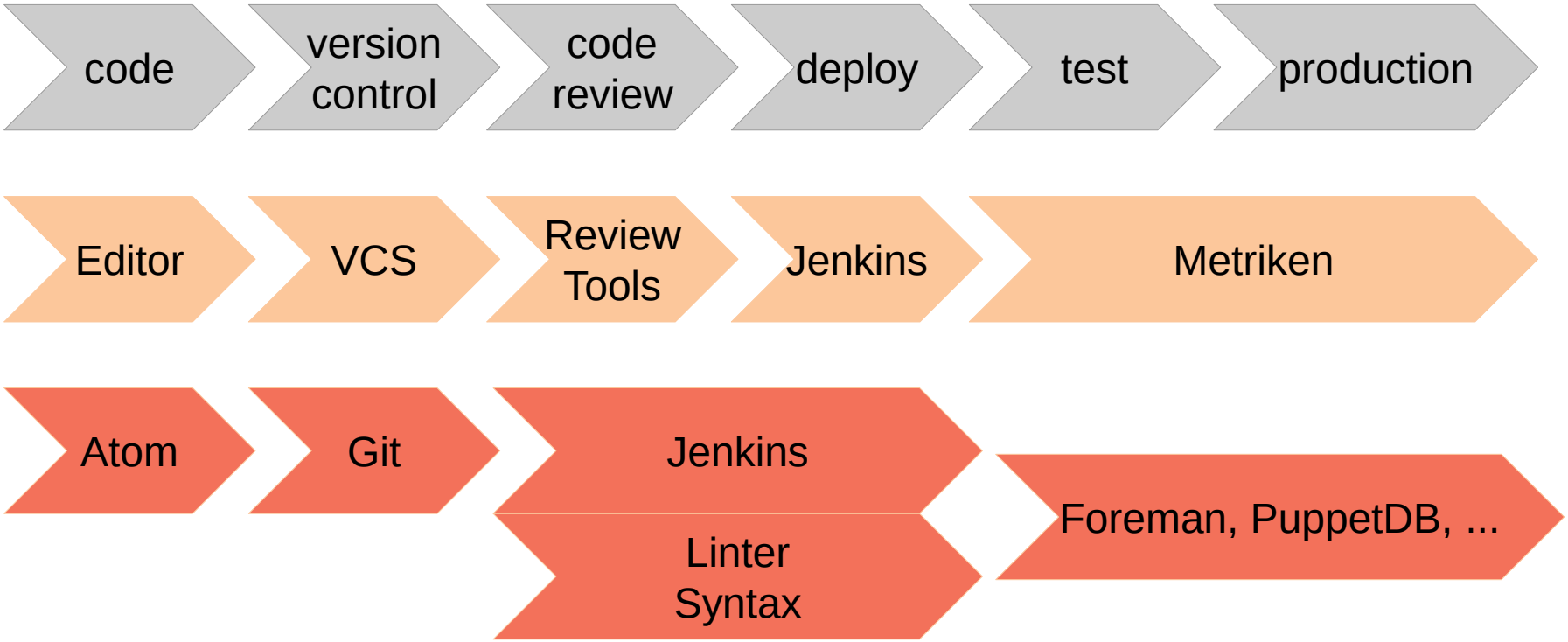


Infrastructure as Code - Workflow



Beispiel

Infrastructure as Code - Workflow



Infrastructure as Code - Code

Datei Bearbeiten Auswahl Anzeigen Los Debuggen Terminal Hilfe

EXPLORER

GEÖFFNETE EDITOREN

customize.sh jenkins

UNBENANNT (ARBEITSBEREI...

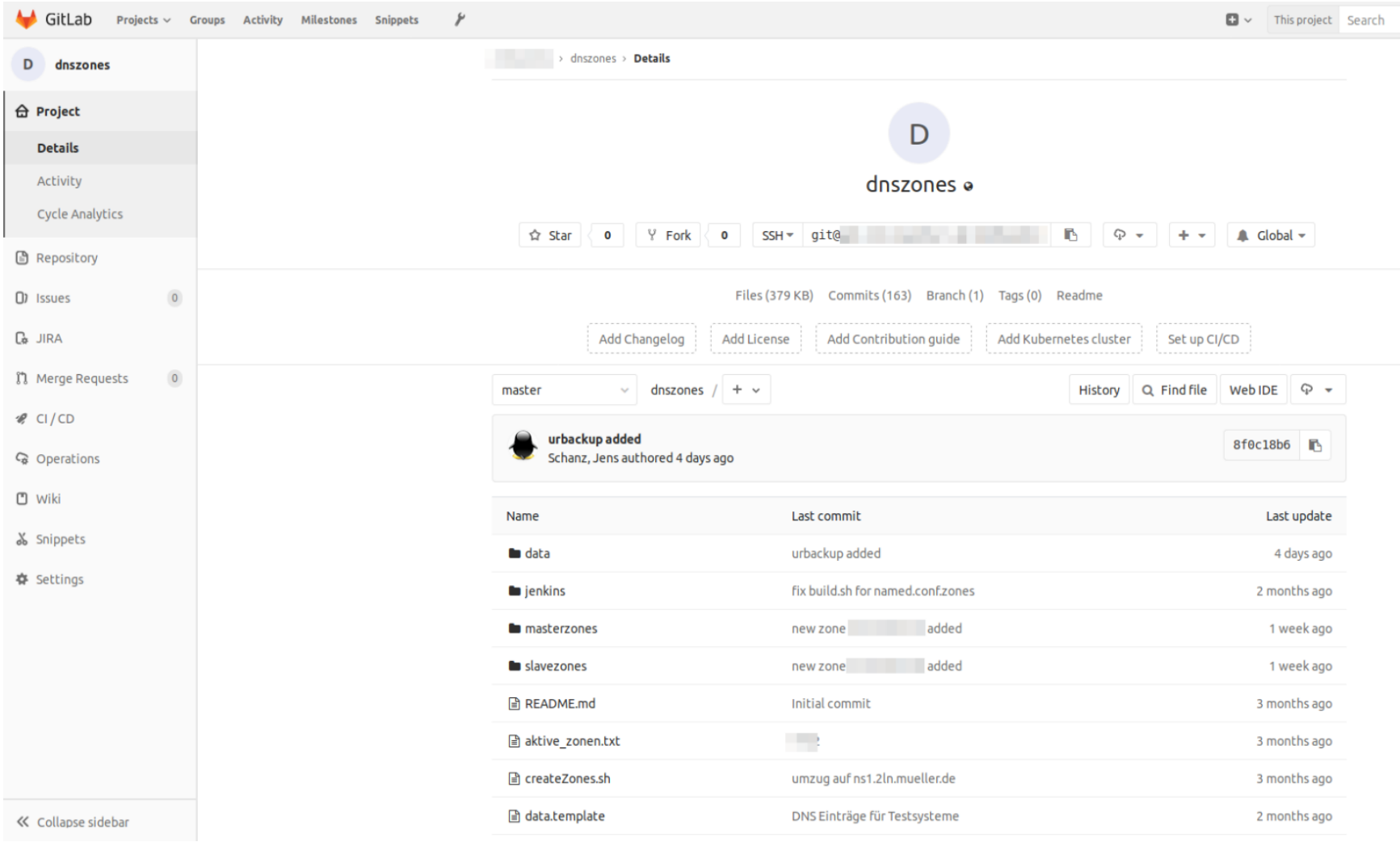
- dnszones
  - data
  - jenkins
    - build.sh
    - check.sh
    - customize.sh
  - masterzones
  - slavezones
  - aktive_zonen.txt
  - createZones.sh
  - data.template
  - named.conf.local
  - named.conf.logging
  - named.conf.statistics
  - named.conf.zones
  - named.logrotate
  - README.md
  - zones.txt

customize.sh

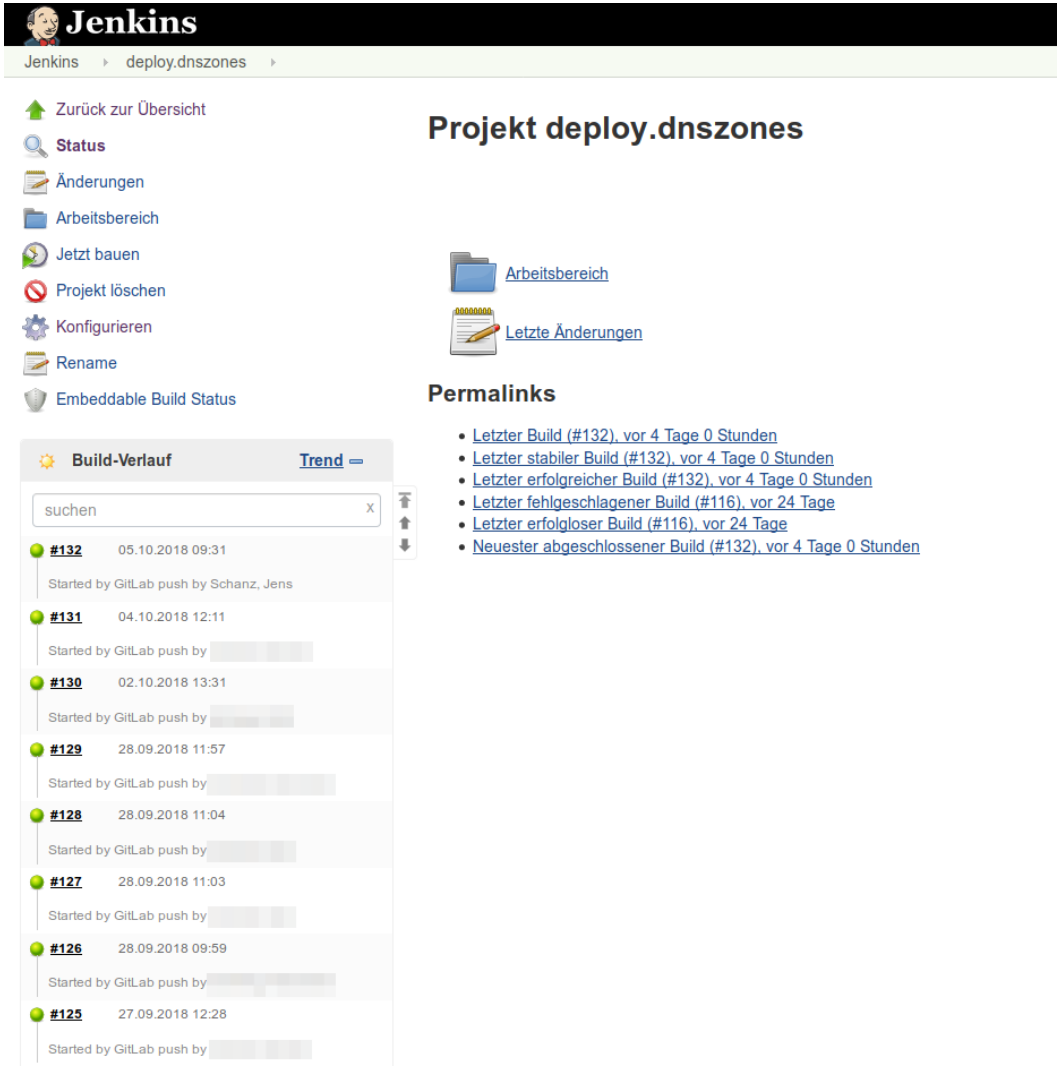
You, 3 months ago | 2 authors (You and others)

```
#!/usr/bin/env bash                                  , 3 months ago • deployment by jenkins

##########################################
# WARNING:
# Managed by jenkins - DO NOT execute locally. Let jenkins do the job
##########################################

# check if $BUILD_NUMBER is set
if [ -z "$BUILD_NUMBER" ]; then
    BUILD_NUMBER="0"
fi

ZONEFILE_LIST="data/*.db"

for ZONEFILE in $ZONEFILE_LIST
do
    #first: replace @@serial@@ by current timestamp
    sed -i "s/@@serial@@/$BUILD_NUMBER/g" "$ZONEFILE"

done
```

Infrastructure as Code – Version Control



Infrastructure as Code – Code Review



Infrastructure as Code – Code Review

GEÖFFNETE EDITOREN
check.sh jenkins
UNBENANNT (ARBEITSBEREI...
dnszones
data
jenkins
build.sh
check.sh
customize.sh
masterzones
slavezones
aktive_zonen.txt
createZones.sh
data.template
named.conf.local
named.conf.logging
named.conf.statistics
named.conf.zones
named.logrotate
README.md
zones.txt
GLIEDERUNG

```
55
56  global_ip_list=''
57
58  for zonefile in $zonefile_list
59  do
60      zone=$(echo "$zonefile" | sed -n "s|^.*/\([^/]*\)\.db|\1|p")
61      echo "... validating zonefile \"$zonefile\" for zone \"$zone\""
62
63      if [ -n "$do_syntax_check" ]; then
64          if sudo /usr/sbin/named-checkzone "$zone" "$zonefile"; then
65              echo "OK: $zonefile successfully verified"
66          else
67              echo "ERROR: could not verify $zonefile"
68              all_checks_passed= 0
69          fi
70      fi
71
72      zone_ip_list=$(sed -n 's|^[^ ;].*[^0-9]\([0-9]\{1,3\}\.[0-9]\{1,3\}\.[0-9]\{1,3\}\.[0-9]\{1,3\}\).*|\1|p' $zonefile)
73      global_ip_list=$(echo -e "$global_ip_list\n$zone_ip_list")
74
75      if [ -n "$do_ping_check" ]; then
76          #still alive? # exclude [redacted] zones
77          zonetest=$(! [[ $zone =~ ^([0-9]{4}|[redacted]) ]])
78          if [ "$?" -eq "1" ]; then
79              echo "  OK: zone $zone is excluded from pingtest"
80          else
81              if [ "$zone_ip_list" != '' ]; then
82                  while read -r ip; do
83
84                      full_line=$(grep "$ip" $zonefile)
85                      hostname=$(echo "$full_line" | sed -n 's|\([^ ]*\).*|\1|p')
86                      exclusiontest=$(! [[ $excludelist =~ (^|[[:space:]])$hostname($|[[:space:]]) ]])
87
88                      if [ "$?" -eq "1" ]; then
89                          echo "  OK: $hostname is excluded from pingtest"
90                      else
91                          response=$(ping -c 2 "$ip")
92                          responsecode=$?
93                          if [ "$responsecode" -ne "0" ]; then
94                              echo "  ERROR: ip not reachable: $full_line"
95                              all_checks_passed=0
96                          #  else
97                          #      echo "  OK: $ip reachable"
98                          fi
99                      fi
100
101                 done  < <( echo "$zone_ip_list" );
102             fi
103         fi
```

Infrastructure as Code – Code Review

Jenkins › deploy.dnszones ›

General | HTML5 Notification Configuration | Source-Code-Management | Build-Auslöser | Buildumgebung | **Buildverfahren** | Post-Build-Aktionen

Buildverfahren

Shell ausführen

Befehl

```
#!/usr/bin/env bash
sudo apt update
sudo apt install -y bind9utils
```

Liste der verfügbaren Umgebungsvariablen

Erweitert...

Shell ausführen

Befehl

```
#!/usr/bin/env bash
jenkins/customize.sh
```

Liste der verfügbaren Umgebungsvariablen

Erweitert...

Shell ausführen

Befehl

```
#!/usr/bin/env bash
jenkins/check.sh -s -d
```

Liste der verfügbaren Umgebungsvariablen

Infrastructure as Code – Code Review

```
Hit:7 http://debian.repository.[redacted]/icinga stretch InRelease
Hit:8 http://debian.repository.[redacted]/beats stretch InRelease
Hit:9 http://debian.repository.[redacted]/docker stretch InRelease
Hit:10 http://debian.repository.[redacted] jenkins InRelease
Get:11 http://debian.repository.[redacted]/stretch-java stretch/main amd64 Packages [1,946 B]
Fetched 6,250 B in 1s (5,759 B/s)
Reading package lists...
Building dependency tree...
Reading state information...
7 packages can be upgraded. Run 'apt list --upgradable' to see them.

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Reading package lists...
Building dependency tree...
Reading state information...
bind9utils is already the newest version (1:9.10.3.dfsg.P4-12.3+deb9u4).
0 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.
[deploy.dnszones] $ /bin/bash /tmp/jenkins5301889690471719171.sh
[deploy.dnszones] $ /bin/bash /tmp/jenkins3207246538016236052.sh
... validating zonefile "data/[redacted].in-addr.arpa.db" for zone "[redacted].in-addr.arpa"
zone [redacted].in-addr.arpa/IN: loaded serial 132
OK
OK: data/[redacted].in-addr.arpa.db successfully verified
-----------------------------------
... validating zonefile "data/[redacted].db" for zone "[redacted]"
zone [redacted]/IN: loaded serial 132
OK
OK: data/[redacted].db successfully verified
-----------------------------------
... validating zonefile "data/[redacted].in-addr.arpa.db" for zone "[redacted].in-addr.arpa"
zone [redacted].in-addr.arpa/IN: loaded serial 132
OK
OK: data/[redacted].in-addr.arpa.db successfully verified
-----------------------------------
... validating zonefile "data/[redacted].db" for zone "[redacted]"
zone [redacted]/IN: loaded serial 132
OK
OK: data/[redacted].db successfully verified
```

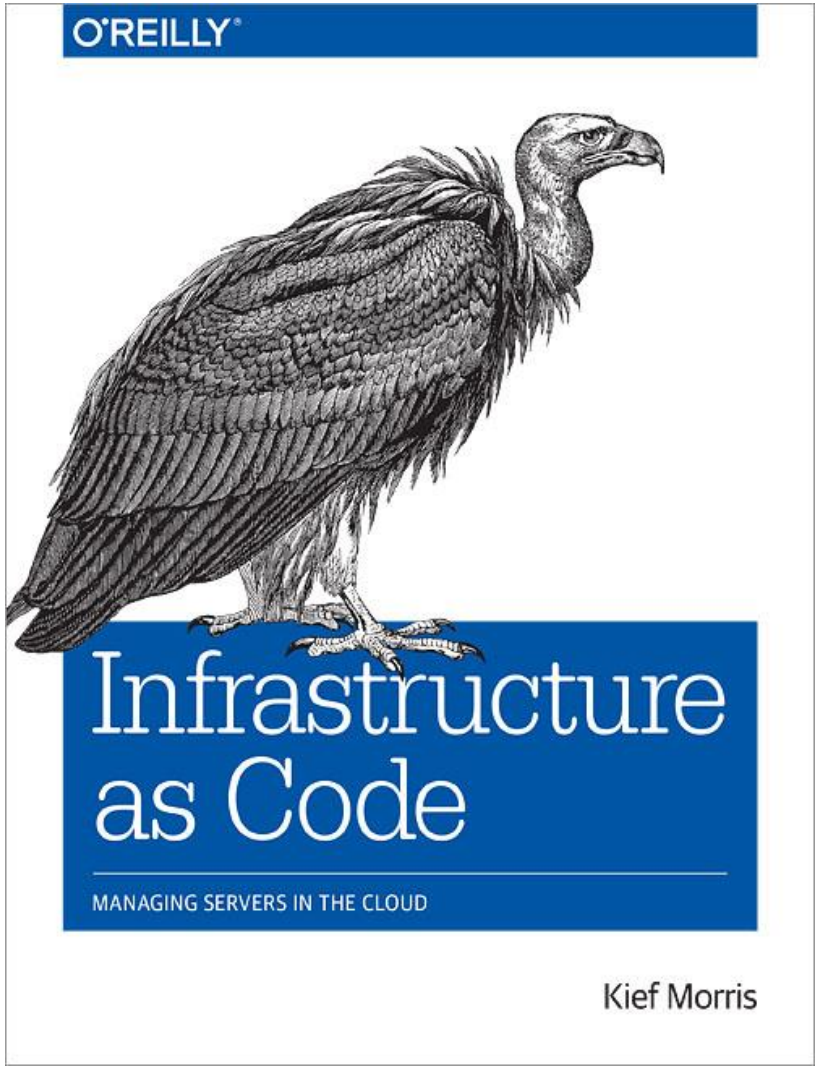
Infrastructure as Code – Code Review

```
-----------------------------------
... validating zonefile "data/[redacted].db" for zone "[redacted]"
zone [redacted]/IN: loaded serial 116
OK
OK: data/[redacted].db successfully verified
-----------------------------------
ERROR: duplicate ip addresses found:
data/[redacted]
data/[redacted]
Build step 'Execute shell' marked build as failure
Finished: FAILURE
```

Warum?

Deswegen ...

Literatur



- **Infrastructure as Code** Managing Servers in the Cloud
- Kief Morris
- ISBN-13: 978-1491924358