Continous Integration & Continous Deployment -
For the new nameserver infrastructures of DENIC eG

15/11/15 – Christian Petrasch
petrasch@denic.de
Agenda

1. Motivation
   • Short overview about the old structures
   • Problems with the old deployment technics
   • CI/CD

2. How to reach the goal – Development steps to CI/CD
   • Requirements
   • Developing concepts and the processes

3. Implementation in the new infrastructure and the final processes
   • The new infrastructure
   • The final process
   • Deployment strategy - check

4. Automated tests

5. Further benefits / Future plans

6. Questions
Motivation

Short overview about the old structures

- DE, enum and several customer TLDs
- 18 locations
- Nearly 300 servers for DNS

- Configuration via CFengine2
- Nameserver software rollout via CFengine2
- Virtualization via images and XEN (not XenServer)
- Monitoring via Nagios
Motivation

Problems with the old deployment technics

- Server images are hard to maintain and keep up to date
- CFengine2 is oversized / Migration from CF2 to CF3 complicated
- A lot of manual, decentral configuration (storage, monitoring, server rollout)
- No centralized administration
- Routing control (slow and unreliable)
Motivation

Consolidated....

• Every change takes a lot of time - For a new customer we needed several weeks – AAAAAAAAAARGGGGGHHHH!!!!!!

• Old processes with this amount of servers not very handy –

• Configuration failures were nearly unavoidable

• Images prevent nearly all kind of server updates because of too much downtimes

And we change our hardware.....WE NEED STH. NEW !!!!!
Motivation

What to do?

Maybe Continuous integration!!

Colleagues from other team already do this ...!
Motivation

- Software defined datacenter ?
- Virtualization ?
- Immutable servers ?
- Hmmm ?

But in reality for DENIC...

- Infrastructure as Code
- Automated tests
- Possibility to change sth. without service outages
- Staging infrastructure
- Possibility to integrate updates and changes continuously
Motivation

At 2012 we started with CI/CD with our registry.
Motivation?

After we got a lot of experiences with the registry datacenter CI/CD infrastructure we knew, that CI/CD technics give us a lot advantages...

- Less outages
- Increased agility with changes and updates
- Improved scaling of hardware resources
- Do better and automated testing

Good … so do this for DNS infrastructure and solve the old problems and obstacles !!
Motivation

Technics have great benefits for DNS infrastructure:

- Reinstallation of a single location fully automated
- Big security increase because of immutable servers, after rebuild possible malware etc. removed
- Automated deployment and testing, parallel installations of all locations should be possible
- Centralized administration platform
- Implement new server only in database and installation fully automated
- Improved control of nameservers and routing via GUI
- Improved testing
- Improved automation at monitoring

For our customers this would be also a great benefit!!
We could react very much faster and implement updates and customers on time!!
How to reach the goal – Development Steps to CI/CD for DNS

Developing concepts

Questions

• Same tools as registry platform?
• Which virtualization?
• How to do configuration management?
• Which software to control everything?
• What is needed to connect everything?

What to do

• Consolidate requirements
• Evaluate and test several hypervisors
• Decide how to want to do configuration management
• Evaluate and test orchestration solutions
• Develop a toolstack
How to reach the goal – Development Steps to CI/CD for DNS

Requirements (the most important)

• Highly reliable and stable
• High cost efficiency
• Central administration
• Easy to use, update, test and automate
• Highly automated
• minimized unavoidable downtimes and guarantee of no service outage during reinstallation
The hypervisor decision

- Tested VMware, XENserver, KVM
- Tested for Performance, Cost efficiency, Security

Reasons:

- Performance good enough
- Security better than KVM
- Free, VMWare too expensive

DECISION:

XENserver (free license)
Configuration management

Reasons:

- Has a good Python API
- We need cobbler as installserver, why not as configuration tool
- Cobbler can do configuration templating during installation
- No need for Cfengine etc.

DECISION:

Do it with Cobbler..
How to reach the goal – Development Steps to CI/CD for DNS

The orchestration

- Centralized
- User interface
- Easy to implement, maintain and expand

Reasons:

- Ansible is written in Python, many plugins
- Easy to learn
- Usable for nearly every hardware
- Jenkins is a good GUI with a lot of Plugins

DECISION:

Do it with Ansible and Jenkins
How to reach the goal – Development Steps to CI/CD for DNS

How to connect all tools to a complete toolstack

- Common API
- Maybe plugins, modules, etc. available

Reasons:

- Tools are all in Python (API) (Python)
- Inhouse experiences (PostgresSQL)
- Netconf: Easy to implement as python module in ansible

DECISION:

Do it with Postgres, Python, netconf
Deployment Strategy

- Two possibilities
  - Switch between BLUE and GREEN infrastructure (like registry)
    - Advantage: Building of servers not in environment with current production
  - Serial deployment server by server
    - Advantage: Service could be provided continuously without double the hardware costs.

DECISION:

Do it serial, supposing service could be provided with no outage with one nameserver during reinstall. Furthermore double number of hardware in location was not really an option
Implementation in the infrastructure and the final processes

The final processes

Processes

- Installation (and reconfiguration) of servers
- Implemented automated testing
- Administration of the deployment strategy
- Administration of the infrastructure via Jenkins and Ansible
- Routing administration of all anycast clouds
- Central monitoring and logging
Implementation in the infrastructure and the final processes

DENIC developed parts of the toolstack:

Nslconfig (Python)
- Edit and read configuration items in database:
  - profiles, locations, customers, systems etc.

Cobbler_control (Python)
- Communicates with cobbler api and do all jobs which are necessary

PostgreSQL
- Database for all information, which is necessary to build and configure servers.

Ansible
- Ansible is the logic for all purposes behind jenkins
Implementation in the infrastructure and the final processes

The new staging process

1. Development
   - NSL Server
   - cobbler install
   - Development
2. Integration/Testing
   - Data for testing
3. Test
4. Serial Rollout
   - Production
Implementation in the infrastructure and the final processes

The final process and the deployment strategy

- **Input**
- **Control**
- **Test**
- **Install**
- **Routing**
- **Maintenance**

Diagram:

- Input to DB via Tools
- Input via GUI
- Controlserver
- Monitoring
- ZABBIX
- Monitoring
- Output
- Automated Testing
- Reposync/Installation
- Installserver (Cobbler)
- Infrastructure Log/Monitoring
- Nameserver Maintenance
- Nameserver Location
- DNS Traffic
Implementation in the infrastructure and the final processes

**Deployment Strategy**

- Decision with serial deployment was a good decision
  (Rebuild NSL UK1 15/09/17 – 8:05 am – 8:49 am)
Automated tests

Integration stage rebuilds at least every night

- Integration tests
  - Functional tests to ensure nameserver could answer all record types we have in our zone
  - Functional tests to ensure nameservice answers during reinstallation
  - Nightly lightweight nameserver software performance tests
Automated tests

- Smoketests (performed by ansible)
  - All server
    - Installed and functional monitoring, ssh, ntp, passwords correct set, defined hardening standards are fulfilled
  - For each type of server in our NSL special tests
    - Check that all necessary processes are running and responding
    - Nameserver
      - Processes: Check if DSC is installed and running
      - Services:
        - Check if unallowed AXFR possible
        - Nameserver answers at every configured interface
Further Benefits

- Goals reached !!
  - Central Admin-GUI
  - Rebuild of a location in 45 min
  - Parallel installation of all locations possible because of serial deployment
  - Production-readyness is ensured through smoketests
  - Automated monitoring integration with templates
Future plans

- Automation of network (SDN) including automated testing
- Automation of hypervisor installations and updates
- Implement additional monitoring and measurements
- Rebuild location faster
- Increasing test coverage
Thank you!

Questions?

petrasch@denic.de