DevFest Vienna 2015

21.11.2015

Ansible - Simple IT Automation

Gerhard Hipfinger
openForce

Who am I

- Gerhard Hipfinger
- Founder of openForce in 2002
- Java Developer
- Scala Developer
- System and Software Architecture addicted
- Linux/Mac guy
- Entrepreneur

How did we start?

Everything was hand crafted.

Documentation was had no priority for us.

After the first system crash we've learned by the hard way to improve.

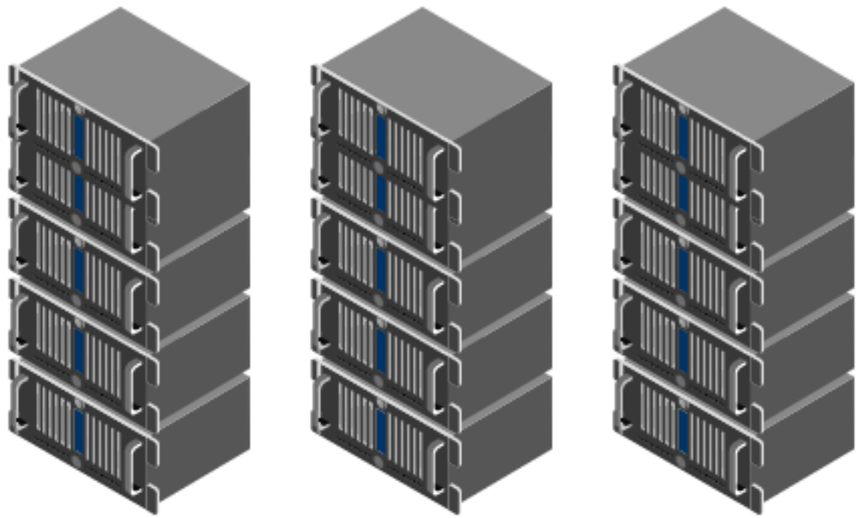
Later in 2008

> About 10 physical servers with 30 VM's.

> We started to automize. First shell scripts then Puppet.

> A system crash was not a disaster anymore but still cumbersome.

And now?



No more hardware! Complete outsourced IaaS.

About 50 VM's and counting...

We want to develop great software. So we need a great server environment too!

How to handle that?

> Handcrafted Servers

- hard to maintain
- time/cost intensive
- repeatable task and error prone
- leads to bad quality
- hard and therefore - never - completely documented

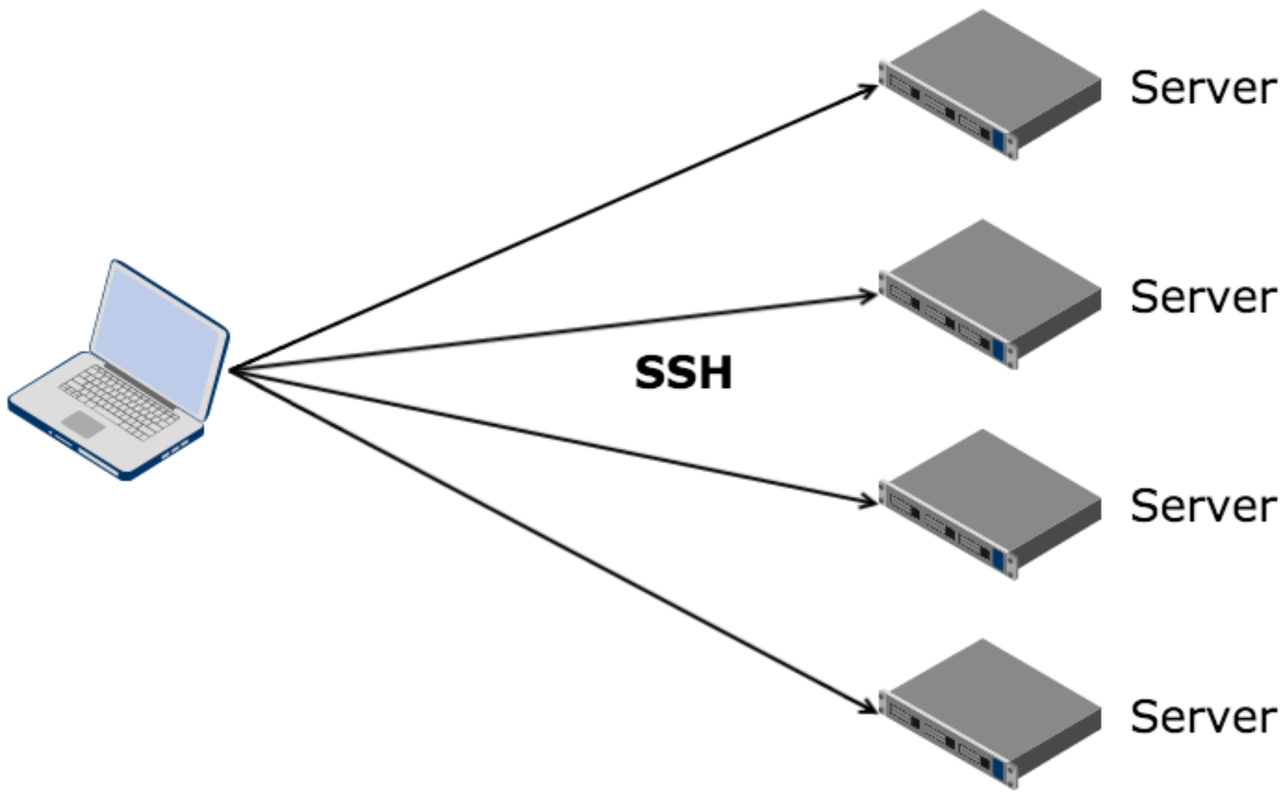
Ansible to the rescue!

Why Ansible is the right tool for us - and maybe you

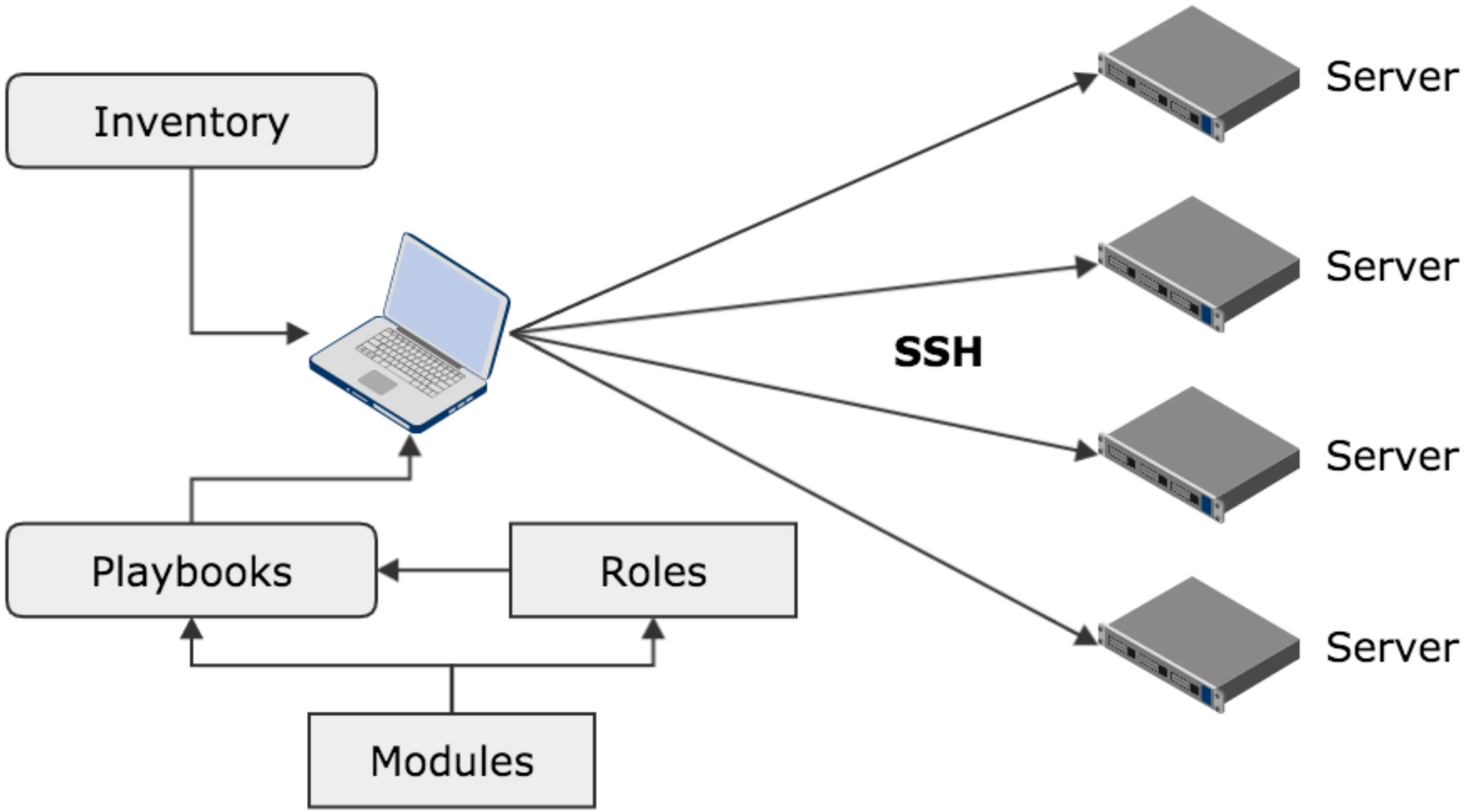
- no master server
- no client software
- simple but powerful configuration
- flat learning curve
- we can „code“ our environment

How Ansible works

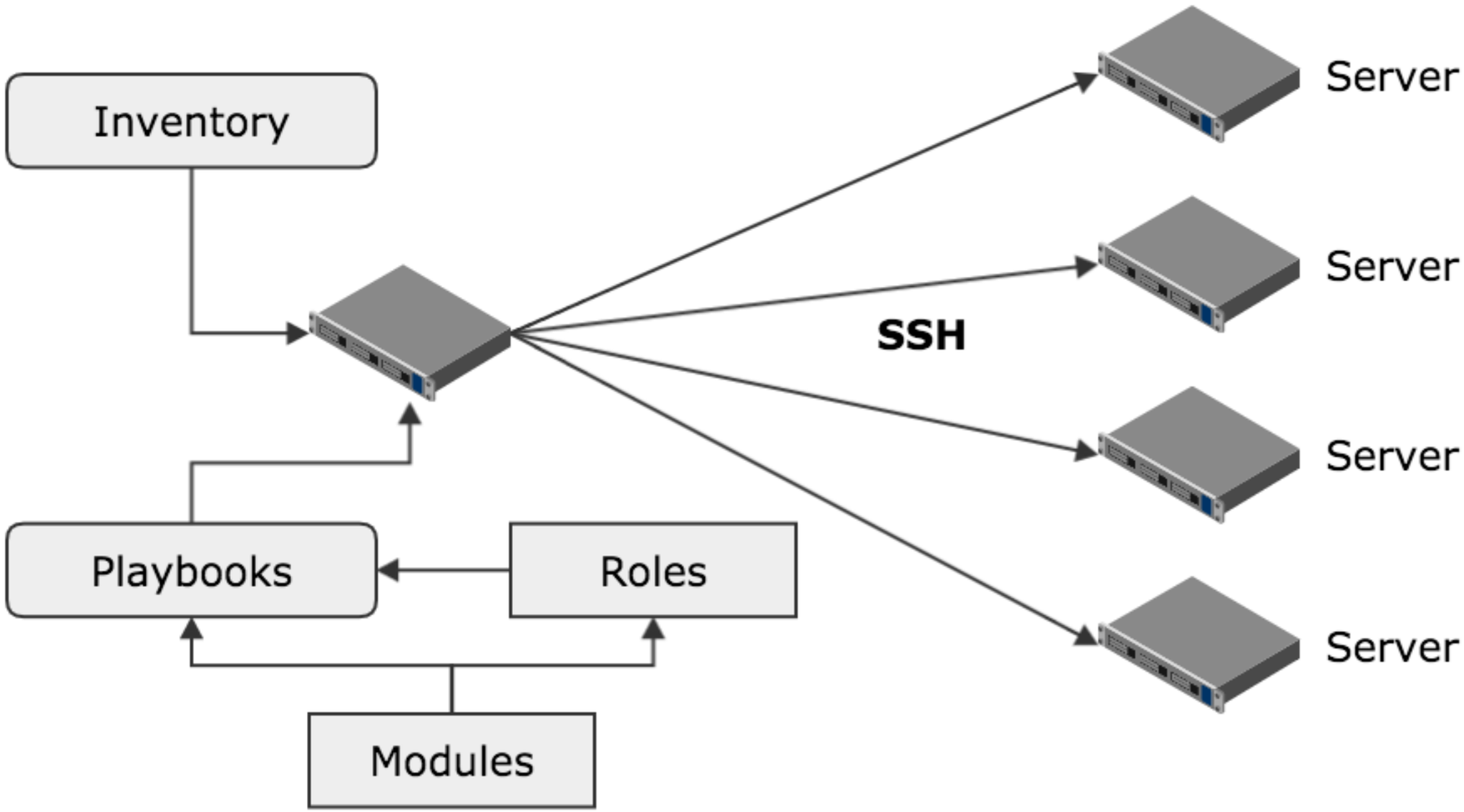
> It works like a human. Login via ssh and do the work.



How Ansible works



How Ansible works



The Inventory

The Inventory

```
[smtp]
192.168.100.10

[web]
192.168.100.20
192.168.100.21

[mongodb]
192.168.100.30
192.168.100.31

[elasticsearch]
192.168.100.40
192.168.100.41
192.168.100.42
```

> An inventory is a simple text file witch lists your servers optionally grouped by names.

The Inventory

```
[smtp]
smtp.openforce.com

[web]
web01.openforce.com
web02.openforce.com

[mongodb]
mongdb01.openforce.com
mongdb02.openforce.com

[elasticsearch]
esearch01.openforce.com
esearch02.openforce.com
esearch03.openforce.com
```

> You can use DNS names instead of IP addresses too

The Inventory

```
[smtp]
smtp.openforce.com

[web]
web[01:20].openforce.com
```

```
[mongodb]
mongdb[01:02].openforce.com


[elasticsearch]
esearch[01:03].openforce.com
```

> We can use enumerations to cleanly organize or inventory

The Inventory

```
[smtp]
smtp.openforce.com

[web]
web[01:20].openforce.com
```

```
[mongodb]
mongdb[01:02].openforce.com ntp=ntp1.pool.ntp.org

[mongodb:vars]
myvar=a_given_value
```

> And we can assign variables to hosts and host groups

The Inventory

```
[smtp]
smtp.openforce.com

[web]
web[01:20].openforce.com
```

> And we can build group of groups for even more structure

```
[mongodb]
mongdb[01:02].openforce.com ntp=ntp1.pool.ntp.org

[webanddb:children]
web
mongodb
```

The Playbook

The Playbook

```
---
- hosts: owncloud
  sudo: yes

  vars:
    dbname: owncloud
    dbuser: owncloud
    dbpassword: secret

  roles:
    - common
    - postgresql
    - nginx
    - owncloud
    - backupninja
    - remotebackupuser
```

A playbook is just a simple yaml file.

Perfectly human readable. Ansible guys did a great job in defining a intuitive DSL.

But you need practice to structure your playbooks and roles.

The Playbook

```
---
- hosts: owncloud
  sudo: yes

  vars:
    dbname: owncloud
    dbuser: owncloud
    dbpassword: secret

  roles:
    - common
    - postgresql
    - nginx
    - owncloud
    - backupninja
    - remotebackupuser
```

The host or group name from inventory file
How to authorize (we can define a username too)

Variables used in templates or subtasks

Roles that need to be applied for this host or group of hosts

The Playbook - Module/Tasks

```
- name: Install php-fpm packages
  action: apt name={{item}} state=present
  with_items:
    - php5-fpm
    - php5
    - php5-gd
    - php-xml-parser
    - php5-intl
    - php5-pgsql
    - php5-curl
    - php5-json
    - php5-ldap
```

Documentation is mandatory (!!!)

Module

Module options

A list type

The Playbook - Module/Tasks

```
- name: Install php-fpm packages
  action: apt name={{item}} state=present
  with_items:
    - php5-fpm
    - php5
    - php5-gd
    - php-xml-parser
    - php5-intl
    - php5-pgsql
    - php5-curl
    - php5-json
    - php5-ldap
```

Support for iteration and list types. But there is much more to discover!

1.000's of modules

conditionals, result processing, tagging, include files

The Role

The Role

```
/roles
  /backupninja
  /bootstrap
  /common
  /java
  /logstash
  /nginx
  /owncloud
  /postgresql
  /redis
  /remotebackupuser
```

A role is a well defined structure of reusable components in server provisioning/orchestration.

When you get the point with Ansible you mainly craft your own roles.

Each role is in its directory and has a well defined structure according to Ansible best practices.

The Role

```
/roles
  /common
    /defaults
    /files
    /handlers
    /tasks
      main.yaml
/templates
/vars
  main.yaml
```

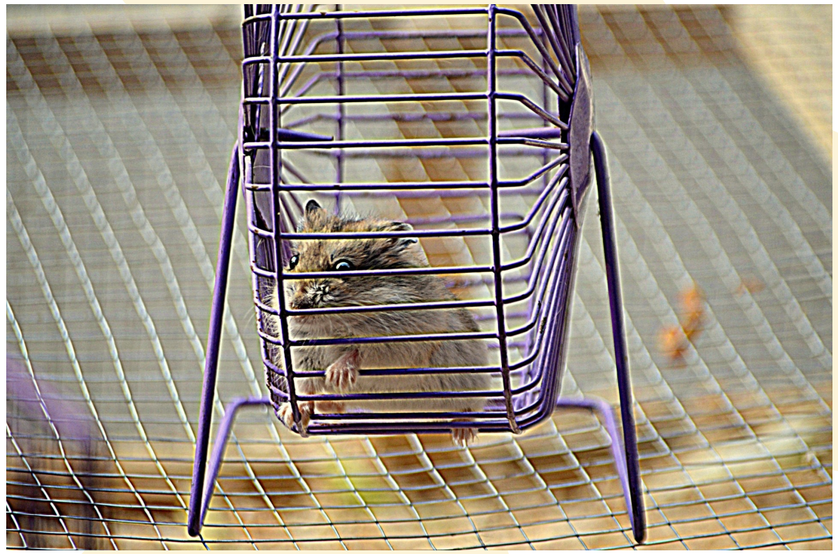
A role is a well defined structure of reusable components in server provisioning/orchestration.

When you get the point with Ansible you mainly craft your own roles.

Each role is in its directory and has a well defined structure according to Ansible best practices.

Once more - why does this work?

- Ansible gathers facts about the target host
- Facts are checkt against the task list
- Performs only tasks that would change facts
- Kinda „rsync“ for system configuration
- Requirement: All tasks need to be idempotent!



There is so much more...

- Dynamic Inventories
- Ad Hoc Commands
- Vagrant, AWS and other cloud services
- Ansible Galaxy
- But I hope I've made you curious...

Thank You

Gerhard Hipfinger (@nano4711)
http://hipfinger.net
http://openforce.com