introducing

ANSIBLE

sales
Haid-und-Neu-Str. 18, 76131 Karlsruhe
Germany

about me

yes, I caught this myself

David Heidt

DevOps Engineer @msales

lots of aws, lots of ansible

I go fishing

I have two children (less time to go fishing, but still fun)

I play The Legend of Zelda

I'm not a layout professional

david.heidt@msales.com

Twitter: @witsches, @msalestech

but why?

David uses ansible. It's super effective!

- agentless
- ssh + python is basic
- straight forward
- start from scratch
- even with existing servers
- incredibly good documentation
- human-readable yml

playbooks

you know what this does whithout knowing ansible. It is OK to feel good now

```
- hosts: redis_server

  vars:
    redis_password: "foobar"

  tasks:

    - apt: update_cache=yes upgrade=dist

    - apt: name=redis-server state=installed

    - lineinfile: dest=/etc/redis/redis.conf line="requirepass {{ redis_password }}"
      notify:
        - restart redis

  handlers:
    - name: restart redis
      service: name=redis-server state=restarted enabled=yes
```

inventory

INI-like format. same-same but different

```
[webworker]
web0.example.com
web1.example.com

[loadbalancer]
lb.example.com

[redis_server]
10.0.0.99
```

more inventory

with hostvars, host groups and unicorns

```
[webworker]
web0.example.com
web1.example.com
web2.example.com unicorns=3

[loadbalancer]
lb.example.com wizardry=True

[webstack:children]
loadbalancer
webworker
```

modules

how to conquer the world

- control the system (filesystems, lvm, service-control, user accounts...)
- work with packaging (apt, yum, brew, but also bundler, composer, pip, bower, ...)
- working with files (Permissions, copy, templating, regex, patch)
- manage databases
- configure monitoring
- control the cloud

modules

how to conquer the world the way you like it

```
- name: restart nginx
  service: name=nginx state=restarted

- name: restart nginx
  service:
    name: nginx
    state: restarted
```

```
ansible all -m service -a "name=nginx state=restarted"
```

write playbooks

make it do what you want it to do

- nested yaml dictionaries
- control structures, loops
- jinja2 template engine
- registering module output as runtime dict
- limits, tags

vaults

better not share credentials while conquering the world

- can encrypt any data structure yml used by ansible
- not only variables, but also tasks or handlers
- 'ansible-vault' ships as executable with ansible
- easy migration: encrypt existing files
- vaults in public, passwords in trusted places
- vaults are AES (shared-secret) encrypted

new in ansible 2: blocks

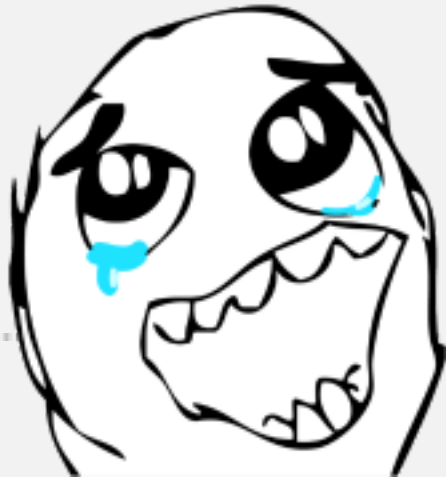
conquering the world in small units

- group similar tasks
- with error handling and rescue
- use conditionals and tags only once
- blocks can be nested!
- block variables are existent in the block only
- any_errors_fatal triggers rescue for all hosts

more new stuff in ansible 2

... for con... OK, I stop it

- execution strategy plugins
- many, many new modules
- improved error messages
- playbook execution engine rewrite
- includes can use dynamic elements



see it in action? live demo.

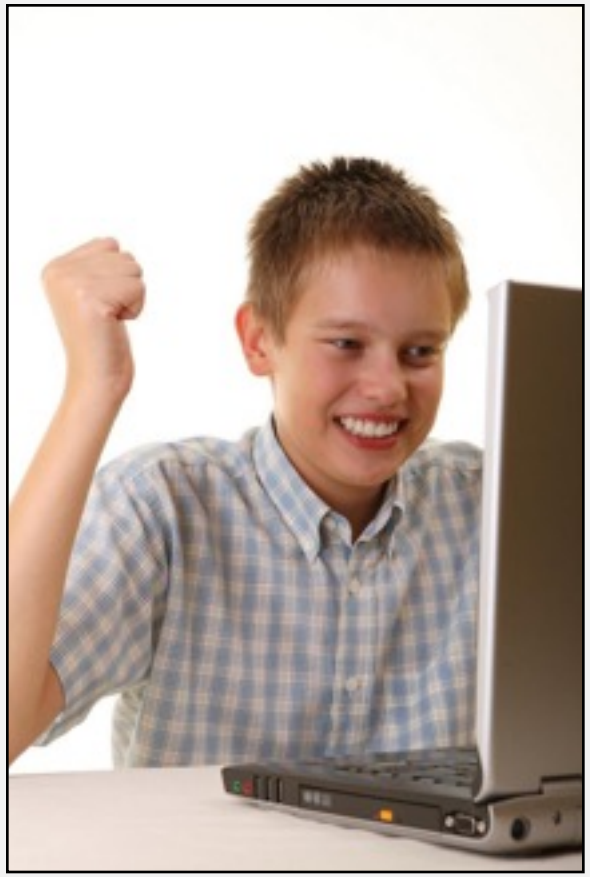
best practices

yeah, this is also in the docs. I know.

- use a provisioning server and agent-forwarding when working in a team
- store your playbook in a repository
- When you can do something simply, do something simply.
- use production and staging
- use ansible galaxy (re-use or just for inspiration)
- browse the documentation now and then

h4cks

best practices we find pretty useful @msales



production inventory

```
[all:vars]
environment=production
aws_access_key=AKIAIOSFO7PRODEXAMPLE
```

staging inventory

```
[all:vars]
environment=staging
aws_access_key=AKIAIOSFOSTAGEEXAMPLE
```

h4cks

best practices we find pretty useful @msales

use (special) tags in playbooks and tasks

```
[…]
  roles:
    - { role: facts, tags: [ 'always','facts' ] }
    - { role: commonsetup, tags: [ 'commonsetup' ] }
    - { role: php5, tags: [ 'php5','webstack'] }
    - { role: nginx, tags: [ 'nginx','webstack']}
    - { role: monitoring, tags: [ 'monitoring'] }
    - { role: telegraf, tags: [ 'monitoring'] }
[…]
```

h4cks

best practices we find pretty useful @msales

use an (almost) empty shared role for frequently used vars

```
me@server:~# tree shared_roles/aws_dict/
shared_roles/aws_dict/
└── vars
    └── main.yml
```

ansible is not for servers only

wait, you were working with servers?

- deployment
- cloud
- monitoring
- docker
- vagrant
- what do you use ansible for? help me, audience!

ansible only for geeks like us?

```
- name: '"let's try tower"
  uri: url=https://www.ansible.com/tower
  when: geek is not defined
```

-> www.ansible.com/tower

actually, there are some pretty cool features for geeks, too.

Thanks.

repository url: https://github.com/msales/ansible-talk

msales GmbH

Haid-und-Neu-Str. 18, 76131 Karlsruhe

Germany

follow us @msalestech

info@msales.com www.msales.com +49 721 91138 0