02_Rollen

March 13, 2018

1 Modular mit Ansible Rollen

Würde man seine Infrastruktur mit *Webservern*, *Datenbanken*, *Loadbalancer*, *Queues* usw. in **einem** Playbook verwalten, würde dies zu einer riesigen Abfolge von Aktionen (Spaghetticode) in **einem** Skript führen.

Darüber hinaus kann man eine Teil z.B. Datenbanken nicht in anderen Projekten weiter verwenden.

Hier kommen Rollen ins Spiel. Zum Beispiel für:

- nginx
- mysql
- mongodb
- tomcat
- ...

Neben Rollen gibt es noch `include` von Playbooks. Rollen sind aber weit aus mächtiger.

```
In [1]: cat ssh-add-passphrase.sh
```

```
#!/usr/bin/expect -f
spawn ssh-add /home/vagrant/.ssh/id_rsa
expect "Enter passphrase for /home/vagrant/.ssh/id_rsa:"
send "geheim\n";
interact
```

```
In [4]: chmod u+x ./ssh-add-passphrase.sh
```

```
In [5]: eval `ssh-agent -s` > /dev/null
        ./ssh-add-passphrase.sh
```

```
Enter passphrase for /home/vagrant/.ssh/id_rsa:
Identity added: /home/vagrant/.ssh/id_rsa (/home/vagrant/.ssh/id_rsa)
```

```
In [6]: ansible -m ping all
```

```
127.0.0.1 | SUCCESS => {    "changed": false,    "ping": "pong"}
192.168.60.11 | SUCCESS => {    "changed": false,    "ping": "pong"}
192.168.60.13 | SUCCESS => {    "changed": false,    "ping": "pong"}
192.168.60.12 | SUCCESS => {    "changed": false,    "ping": "pong"}
192.168.60.2 | SUCCESS => {    "changed": false,    "ping": "pong"}
192.168.60.21 | SUCCESS => {    "changed": false,    "ping": "pong"}
192.168.60.22 | SUCCESS => {    "changed": false,    "ping": "pong"}
```

```
In [7]: cd Kap2_Rollen
```

```
In [8]: pwd
```

```
/home/vagrant/projects/Kap2_Rollen
```

```
In [9]: tree --charset=ascii
```

```
.
|-- customhosts
|-- roles
|   |-- base
|   |   `-- tasks
|   |       `-- main.yml
|   `-- nginx
|       |-- files
|       |   |-- default.conf
|       |   `-- index.html
|       |-- handlers
|       |   `-- main.yml
|       |-- meta
|       |   `-- main.yml
|       `-- tasks
|           |-- configure.yml
|           |-- install.yml
|           |-- main.yml
|           `-- service.yml
|-- site.yml
`-- www.yml

8 directories, 12 files
```

Rollen werden u.a. unter dem Ordner `roles/` abgelegt. Können aber auch über mehrere Ordner verteilt werden, wie z.B. `/deploy/ansible/roles` und `/deploy/ansible/community/roles`. Dies kann über eine Datei **ansible.cfg** mit dem Eintrag

```
roles_path = /deploy/ansible/roles:/deploy/ansible/community/roles
```

geschehen.

Mehr über Rollen hier: http://docs.ansible.com/ansible/playbooks_roles.html

und über ansible.cfg: http://docs.ansible.com/ansible/intro_configuration.html

Jede Rolle bekommt einen Ordner mit ihrem Rolen-Namen. Diese Ordner können weitere Unterordner haben. Der wichtigste Unterordner ist `tasks/`. Darüber hinaus gibt aber noch weitere Ordner wie `handlers/`, `templates/`, `files/`, `meta/`, `vars/` uvw. Normalerweise beinhalten alle diese Ordner eine Datei mit dem Namen `main.yml`.

Wie können Rollen aufgerufen werden? Schauen wir uns das Playbook `site.yml` an:

```
In [6]: cat site.yml
```

```
---
# This is a sitewide playbook
- include: www.yml
```

```
In [10]: ansible-playbook --syntax-check site.yml
```

```
playbook: site.yml
```

In Playbooks können Teile mit include eingebunden werde. Diese werden im aktuellen Verzeichnis gesucht, wenn kein absoluter/relativer Pfad angegeben wurde.

Mehr über Include: http://docs.ansible.com/ansible/playbooks_roles.html#task-include-files-and-encouraging-reuse

Hier der Inhalt von www.yml

```
In [11]: cat www.yml
```

```
---
- hosts: www
  remote_user: vagrant
  become: yes
  pre_tasks:
  - debug:
      msg: 'I":" Beginning to configure web server..'

  roles:
     - nginx

  post_tasks:
  - debug:
      msg: 'I":" Done configuring nginx web server...'
```

Dieses Playbook wird nur für die Gruppe **www** ausgeführt.

Der Abschnitt **pre_tasks** bzw. **post_tasks** wird vor bzw. nach dem Task Abschnitt ausgeführt.

Unser Task Block wird hier durch eine Rolle **nginx** abgebildet.

Schauen wir uns zunächst den Meta (Beschreibung) zu dieser Rolle an.

```
In [14]: cat roles/nginx/meta/main.yml

---
dependencies:
  - { role: base }
```

Die Rolle `nginx` ist abhängig von der Rolle `base`. Daher hier die Rolle base.

```
In [9]: cat roles/base/tasks/main.yml

---
# essential tasks. should run on all nodes
 - name: creating devops group
   group: name=devops state=present
 - name: create devops user with admin previleges
   user: name=devops comment="Devops User" uid=2001 group=devops
 - name: install htop package
   action: apt name=htop state=present update_cache=yes
```

Hier werden unsere User, Gruppen und Grundlegende Pakete installiert.
Schauen wir uns die Rolle `nginx` im Ordner `tasks/` näher an.

```
In [10]: cat roles/nginx/tasks/main.yml \
             roles/nginx/tasks/install.yml \
             roles/nginx/tasks/configure.yml \
             roles/nginx/tasks/service.yml

---
# This is main tasks file for nginx role
 - include: install.yml
 - include: configure.yml
 - include: service.yml


---
 - name: add official nginx repository
   apt_repository: repo='deb http://nginx.org/packages/ubuntu/ lucid nginx'
 - name: install nginx web server and ensure its at the latest version
   apt: name=nginx state=latest force=yes
---
 - name: create default site configurations
   copy: src=default.conf dest=/etc/nginx/conf.d/default.conf mode=0644
   notify:
    - restart nginx service
 - name: create home page for default site
   copy: src=index.html dest=/usr/share/nginx/html/index.html
```

```
---
 - name: start nginx service
   service: name=nginx state=started
```

Die `install.yml` fügt das *ppa nginx repro* ein. Danach wird die neuste Version von *nginx* installiert. Auch wenn diese schon installiert wurde (force=yes).

In der `configure.yml` wird die Datei `default.conf` aus dem Ordner `files/` der Rolle nginx auf dem entsprechenden Zielpfad kopiert.

ACHTUNG: Das Modul `copy` schaut im Ordner `files/` nach ob die Datei `default.conf` existiert. Somit sind Rollen unabhängig von ihrer Installation.

```
In [11]: cat roles/nginx/files/default.conf

server {
    listen       80;
    server_name  localhost;

    location / {
        root   /usr/share/nginx/html;
        index  index.html;
    }
}
```

```
In [12]: cat roles/nginx/files/index.html

<html>
  <body>
    <h1>Ole Ole Ole </h1>
    <p> Welcome to FIFA World Cup News Portal</p>
   </body>
</html>
```

In der Datei `configure.yml` wird der Handler "restart nginx service" aufgerufen. Dieser wird über den Ordner `handlers/` in der `main.yml` gesucht und gefunden.

Mehr hierzu unter: http://docs.ansible.com/ansible/playbooks_intro.html#handlers-running-operations-on-change

```
In [13]: cat roles/nginx/handlers/main.yml

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

Eine lokales Inventory wird durch die Datei `customhosts` definiert.

```
In [14]: cat customhosts

[local]
localhost     ansible_connection=local

[www]
192.168.60.11 ansible_ssh_user=vagrant
192.168.60.12 ansible_ssh_user=vagrant
192.168.60.13 ansible_ssh_user=vagrant

[lb]
192.168.60.2 ansible_ssh_user=vagrant

[db]
192.168.60.21 ansible_ssh_user=vagrant
192.168.60.22 ansible_ssh_user=vagrant
```

Hier der Aufruf:

```
In [15]: ansible-playbook -i customhosts site.yml


PLAY [www] *********************************************************************

TASK [setup] *******************************************************************
ok: [192.168.60.13]
ok: [192.168.60.11]
ok: [192.168.60.12]

TASK [debug] *******************************************************************
ok: [192.168.60.11] => {    "msg": "I\":\" Beginning to configure web server.."}
ok: [192.168.60.12] => {    "msg": "I\":\" Beginning to configure web server.."}
ok: [192.168.60.13] => {    "msg": "I\":\" Beginning to configure web server.."}

TASK [base : creating devops group] ********************************************
changed: [192.168.60.12]
changed: [192.168.60.13]
changed: [192.168.60.11]

TASK [base : create devops user with admin previleges] *************************
changed: [192.168.60.11]
changed: [192.168.60.13]
changed: [192.168.60.12]

TASK [base : install htop package] *********************************************
changed: [192.168.60.11]
changed: [192.168.60.13]
changed: [192.168.60.12]
```

```
TASK [nginx : add official nginx repository] ***********************************
changed: [192.168.60.11]
changed: [192.168.60.12]
changed: [192.168.60.13]

TASK [nginx : install nginx web server and ensure its at the latest version] ***
changed: [192.168.60.12]
changed: [192.168.60.13]
changed: [192.168.60.11]

TASK [nginx : create default site configurations] ******************************
changed: [192.168.60.12]
changed: [192.168.60.11]
changed: [192.168.60.13]

TASK [nginx : create home page for default site] *******************************
changed: [192.168.60.13]
changed: [192.168.60.11]
changed: [192.168.60.12]

TASK [nginx : start nginx service] *********************************************
ok: [192.168.60.12]
ok: [192.168.60.11]
ok: [192.168.60.13]

RUNNING HANDLER [nginx : restart nginx service] ********************************
changed: [192.168.60.11]
changed: [192.168.60.13]
changed: [192.168.60.12]

TASK [debug] *******************************************************************
ok: [192.168.60.12] => {    "msg": "I\":\" Done configuring nginx web server..."}
ok: [192.168.60.11] => {    "msg": "I\":\" Done configuring nginx web server..."}
ok: [192.168.60.13] => {    "msg": "I\":\" Done configuring nginx web server..."}

PLAY RECAP *********************************************************************
192.168.60.11              : ok=12   changed=8    unreachable=0    failed=0
192.168.60.12              : ok=12   changed=8    unreachable=0    failed=0
192.168.60.13              : ok=12   changed=8    unreachable=0    failed=0
```

Beachten Sie dabei folgendes:

- Die pre_tasks und post_tasks Ausführung.
- Die Rolle base wird ausgeführt. Warum?

Der Funktionstest könnte so aussehen:

```
In [16]: curl 192.168.60.11

<!DOCTYPE HTML><html lang='en' dir='ltr' class='other other0'><head><meta charset="utf-8" /><met
PMA_commonParams.setAll({common_query:"?lang=en&collation_connection=utf8_unicode_ci&token=383f5
ConsoleEnterExecutes=false
AJAX.scriptHandler.add("jquery/jquery-2.1.4.min.js",0).add("whitelist.php?lang=en&amp;db=&amp;co
$(function() {AJAX.fireOnload("whitelist.php?lang=en&amp;db=&amp;collation_connection=utf8_unico
// ]]></script><noscript><style>html{display:block}</style></noscript></head><body id='loginform
    <div class="container">
    <a href="./url.php?url=https%3A%2F%2Fwww.phpmyadmin.net%2F" target="_blank" class="logo"><im
       <h1>Welcome to <bdo dir="ltr" lang="en">phpMyAdmin</bdo></h1><noscript>
<div class="error"><img src="themes/dot.gif" title="" alt="" class="icon ic_s_error" /> Javascri
<div class='hide js-show'><form method="get" action="index.php" class="disableAjax"><input type=
    <br />
    <!-- Login form -->
    <form method="post" action="index.php" name="login_form" class="disableAjax login hide js-sh
        <fieldset>
        <legend>Log in<a href="./doc/html/index.html" target="documentation"><img src="themes/do
                <label for="input_username">Username:</label>
                <input type="text" name="pma_username" id="input_username" value="" size="24" cl
            </div>
            <div class="item">
                <label for="input_password">Password:</label>
                <input type="password" name="pma_password" id="input_password" value="" size="24
            </div><div class="item">
                <label for="select_server">Server Choice:</label>
                <select name="server" id="select_server"><option value="1"  selected="selected">
<option value="2" >localhost  (root)</option>
</select></div></fieldset>
        <fieldset class="tblFooters">
            <input value="Go" type="submit" id="input_go" /><input type="hidden" name="target" v
    </form></div></div></body></html>
```

OOPS: Da leckt noch Öl raus, da muss man noch bei.

M.a.W: Es wird noch die pma geladen und nicht unsere Webseite. Unsere Rolle muss noch angepasste werden.