Konfiguráció menedzsment ansible-vel

Kadlecsik József
MTA Wigner FK
<kadlecsik.jozsef@wigner.mta.hu>

Tartalom

- Konfiguráció menedzsment általában
- Verziókezelés
- Ansible telepítés
- Host inventory
- Playbook alapok
- Role és include
- Változók
- Templating
- Feltételek
- Hibakezelés
- Ansible vault
- Tagek
- Best practices

Konfiguráció menedzsment I.

- OS konfiguráció (fájl) menedzsment
- Infrastructure as code (IaC)
  - Bare-metal, virtualizált, konténerek, microservices
- Előnyök
  - (Humán) erőforrásigény csökkentés
  - Sebesség
  - Kockázatcsökkentés

Konfiguráció menedzsment II.

- Megközelítési módszerek: „what, how, why”
  - Deklaratív (funkcionális): mi a célzott konfiguráció
    - A kívánt állapot definiálása
  - Imperatív (procedurális): hogyan kell megváltoztatni a rendszert, hogy azt elérje
    - Milyen parancsokat milyen sorrendben kell végrehajtani
- Módszerek:
  - Push management → target
  - Pull management ← target

Konfiguráció menedzsment III.

Eszköz	Módszer	Megközelítés	Nyelv
Ansible	push	deklaratív és imperatív	python
cfengine	pull	deklaratív	C
Chef	pull	imperatív	ruby
Puppet	pull	deklaratív	ruby
SaltStack	push és pull	deklaratív és imperatív	python

Miért ansible?

- Nem kell hozzá agent, csak ssh!
- Öndokumentált a YAML szintaxisú konfigurációs fájljain keresztül
- Python
- Mindenre van modul, vagy könnyedén megírható
- Unix way!

Verziókezelés

- Git :-)
  - init; add/rm/mv; commit; push, pull

Ansible telepítés

- Kontroll gépen
  - Python (2.6, 2.7; van 3.5 support)
    - python-jinja2
  - Csomagból, vagy inkább:
    - `git clone git://github.com/ansible/ansible.git`
- Menedzselt gépeken:
  - ssh access
    - Root vagy sudo root
  - Python
    - python-simplejson

Host inventory I.

- „hosts” fájl
  - a menedzselt hostok és csoportjainak listája INI formátumban (YAML is támogatott)
- Default csoportok
  - all
  - ungrouped
- Inventory változó könyvtárak:
  - host_vars/
  - group_vars/

Host inventory II.

```
[web]
ansible-2 ansible_host=193.224.22.179 ansible_port=2222
ansible-4 httpd=lighttpd

[mail]
ansible-[15:18]

[cluster:children]
web
mail
```

Host inventory III.

- Az inventory dinamikusan generálható
  - Tetszőleges forrásból (pl. adatbázisból, DNS-ből, stb.)
  - Tetszőleges nyelven írt programmal
  - YAML szintaxis

Playbook

- Ansible konfigurációs nyelve YAML szintaxissal, Jinja2 tempate támogatással: site.yml

```
---
- hosts: web
  become: true            # remote_user: root
  vars:
     web_package: "{{ httpd|default('apache2') }}"
     web_service: "{{ httpd|default('apache2') }}"
  tasks:
    - name: Ensure webserver is at the latest version
      apt: name="{{ web_package }}" state=latest
    - name: Ensure webserver is running
      service: name="{{ web_service }}" enabled=yes state=started
```

YAML alternatív szintaxisok

- Lista

```
mylist: [ 1, 2, 3, 4 ]
```

- Dictionary

```
mydict: {"a": 1, "b": 2, "c": 3, "d": 4 }
```

Playbook: hosts

- Hostok megadása:
  - `hosts: host0[:host1...]`
  - `hosts: group0[:group1…]  # OR`
  - `hosts: group0:!group1  # exclude`
  - `hosts: group0:&group1  # AND`
  - `hosts: 192.168.1.*`
  - `hosts: web*.exaple.com`

Playbook: users

- User, akinek a nevében belépünk: remote_user
- User, akinek a nevében végre akarunk hajtani egy utasítást: become_user, become, become_method

```
- name: Remove pgsql test user from test db
  postgresql_user:
    db: test
    name: test
    priv: ALL
    state: absent
  become_user: postgres
  become: true
```

Playbook: taskok

- A feladatok listája, amelyeket a megadott hostokon a megadott sorrendben, egymás után végrehajtunk
- A feladatoknak **idempotent**nek kell lenni!
  - Többször végrehajtva ugyan az a hatása, mint egyszer
- Minden task kezdődjék egy „name:” opcióval:
  - A tasklist így öndokumentált
- A key=value argumentumok átírhatók YAML szintaxisra (key: value sub-elemek)

Különleges task-ok

```
- hosts: all
  environment:
    - SHELL_EXTRA_ENV: ”foo”
  pre_tasks:
    - name: Update apt cache
      apt: update_cache=yes cache_valid_time=3600
  tasks:
  post_tasks:
   - debug:
       msg: ”Nice playbook!”
```

Playbook: command, shell

- Használjunk modult, amikor csak lehet és kerüljük a command, shell modulok használatát.
- Visszatérési értéket figyelembe veszi az ansible

```
- name: Enable SELinux
  command: /sbin/setenforce 1
  ignore_errors: true
```

Playbook: handlers

- Változások kezelésére „notify” triggeren keresztül:
  - Minden handler egyszer fut le (ha volt változás) a tasks lista után.
  - A végrehajtáskor a handlerek definíciós sorrendje számít.
  - Ha szükséges, az adott lépésig triggerelt handlerek azonnali végrehajtása kierőszakolható.

Playbook: handlers

```
handlers:
  - name: Restart memcached
    service: name=memcached state=restarted
  - name: Reload apache
    service: name=apache2 state=reloaded
tasks:
  - name: Generate config file from template
    template: src=template.j2 dest=/etc/...
    notify:
      - Restart memcached
      - Reload apache
```

Playbook: listen handlers

```
handlers:
  - name: Restart memcached
    service: name=memcached state=restarted
    listen: Restart webservices
  - name: Reload apache
    service: name=apache2 state=reloaded
    listen: Restart webservices
tasks:
  - name: Generate config file from template
    template: src=template.j2 dest=/etc/config
    notify:
      - Restart webservices
```

Playbook: copy

- Az egyik leggyakrabban használt task modul

```
- name: Copy apache2 config file
  copy:
    src: apache2.conf
    dest: /etc/apache2/apache2.conf
  notify:
    - Reload apache
```

Ansible config és shell változók

- ansible.cfg

```
[defaults]
remote_user = debian
become_method = sudo
ask_pass = false
private_key_file = ~/.ssh/ansible.key


[ssh_connection]
# sudoers: disable 'requiretty' in /etc/sudoers
pipelining = true
```

- Shell paraméterek

```
PYTHONPATH=/usr/local/ansible/lib
PATH=$PATH:/usr/local/ansible/bin
ANSIBLE_CONFIG=$HOME/playbook/ansible.cfg
export PYTHONPATH ANSIBLE_CONFIG
```

Ad-hoc parancsok

```
ansible -i hosts mail -a "/usr/bin/reboot"
ansible -i hosts mail -m copy \
      "src=/file/path dest=/target/path"
ansible -i hosts mail -m apt \
      "name=package state=present"
```

Playbook futtatás :-)

```
ansible-playbook -i hosts --syntax-check site.yml
ansible-playbook -i hosts --list-tasks site.yml
ansible-playbook -i hosts --check site.yml
ansible-playbook -i hosts --check --diff site.yml
ansible-playbook -i hosts -h host|group site.yml
```

Roles I.

- Taskok, handlerek, változók automatikus betöltése egy rögzített könyvtár-struktúrán keresztül:

```
site.yml
roles/apache2/tasks/ , handlers/
              files/ , templates/
              vars/ , defaults/
              meta/
```

Roles II.

- Végrehajtáskor:
  - Minden pre_tasks végrehajtódik
  - Minden triggerelt handler
  - Minden függőségi role
  - Minden task
  - Minden triggerelt handler
  - Minden post_tasks
  - Minden triggerelt handler

Roles III.

- Függőségek

`roles/php/meta/main.yml:`

```
---
dependencies:
  - { role: apache2 }
  - { role: memcached }
```

Include/Import I.

- Tartalom újra felhasználása
  - Statikus: import_plays, import_role, import_tasks
  - Dinamikus: include_role, include_tasks, include_vars

Include/Import II.

- Statikus (import_*):
  - Ansible parsing alatt feldolgozza
  - Parent task opciók elérhetők a child taskokban
  - Nem használható loop-ban
- Dinamikus (include_*):
  - Ansible runtime-kor dolgozza fel
  - Loop-ban használható
  - Tagek (--list-tags), taskok (--list-tasks) nem listázhatók ki ansbile által

Roles IV.

- Szerepekre változókkal lehet hivatkozni

```
---
hosts: web
  tasks:
    - import_role:
        name: apache2
      vars:
        https: true
```

Roles V.

- Feltételes betöltés

```
# roles/apache2/tasks/main.yml (debian.yml,redhat.yml)
---
- name: Handle Debian
  import_tasks: debian.yml
  when: ansible_os_platform|lower == 'debian'
- name: Handle RedHat
  import_tasks: redhat.yml
  when: ansible_os_platform|lower == 'redhat'
```

Roles VI.

- Feltételes betöltés

```
# roles/apache2/tasks/main.yml
---
- name: Set variable for distribution
  set_fact: osname="{{ ansible_os_platform|lower }}"
- name: Load distribution specific role file
  import_tasks: "{{ osname }}.yml"
```

Változók

- YAML változók: skalár, list, dictionary

```
users:
  - name: kissp
    shell: bash
  - name: nagyp
    shell: zsh
```

- Jinja2 hivatkozás

```
users[0]['name']
users[0].name
```

Változó megadása

- „Bárhol”:
  - cmdline
  - Inventory
  - Playbook
  - Role
- Precedencia jól definiált és intuitív

Facts

- Adott hostról ansible által begyűjtött adatok
  - Hostname, IP címek
  - OS, distribúció neve, verziója
  - Virtualizáció és típusa
  - ...
- Az adatok másik host számára elérhetők:

```
{{ hostvars['hostname']['ansible_default_ipv4']['address'] }}
```

Local facts

- Hostról begyűjtött adatok tetszőlegesen bővíthetők

```
# /etc/ansible/facts.d/drbd.fact

#!/usr/bin/perl

use JSON;

my $primary = `/usr/sbin/drbd-overview`;

my $data = { primary => $primary =~ /Connected Primary/ ? 1 : 0 };

my $JSON->new;

print $json->pretty->encode($data), "\n";
```

- Elérhető ansible_local ág alatt:

```
{{ hostvars['hostname'].ansible_local.drbd.primary }}
```

Local facts kezelése ansible-vel :-)

```
handlers:
- name: Call local facts
  setup:
tasks:
  - name: Copy local facts
    copy: src={{ item }} dest=/{{ item }}
    with_items:
      - etc/ansible/facts.d/drbd.fact
    notify:
      - Call local facts
  - name: Run local facts if changed
    meta: flush_handlers
```

Regisztrált változók I.

- Task modulok változóban elmentett eredménye

```
- name: Check enabled extra apache2 modules
  stat: path=/etc/apache2/mods-enabled/{{ item }}.load
  register: mod_enabled
  with_items:
    - “{{ extra_modules }}”
- name: Print registered variable
  debug: var=mod_enabled
```

Regisztrált változók II.


```
{"failed": false,
 "mod_enabled": {
    "results": [
       {
         "item": "ssl",
         "stat": {
            "exists": false,
         },
       },
```

Regisztrált változók III.

```
- name: Check enabled extra apache2 modules
  stat: path=/etc/apache2/mods-enabled/{{ item }}.load
  register: mod_enabled
  with_items:
    - “{{ extra_modules }}”
- name: Enable extra apache2 modules
  command: /usr/sbin/a2enmod {{ item.item }}
  when: item.stat.exists == false
  with_items: “{{ mod_enabled.results }}”
```

Jinja2 I.

- Változók felhasználása ansible playbook-ban, template fájlokban
  - Tesztek
    - Jinja2 builtin
      - defined, undefined, …
    - match, search
    - is_file, is_dir, exists, …

```
- do something
  when: myvar|defined and myvar|match("foo*")
```

Jinja2 II.

- Filterek
  - Default értékek: `{{ myvar|default("foo"|[]|{}) }}`
  - Halmaz műveletek: unique, union, intersect, ..
  - sort(reverse,case, attr); dictsort(case, key|value)
  - JSON query
  - ipaddr filter
  - `{{ (myvar == true) | ternary("a","b") }}`
  - basename, dirname
  - regexp

Jinja2 III.

- Paraméter kihagyása

```
- name: Copy files and set access rights
  copy:
    src: "{{ item.name }}"
    dest: .ssh/"{{ item.name }}"
    mode: "{{ item.mode|default(omit) }}"
  with_items:
    - name: ssh_key.pub
    - name: ssh_key
      mode: 600
```

Templating

- Statikus fájl helyett dinamikusan generált fájl
  - copy, „files” könyvtár
  - template, „templates” könyvtár

```
# /etc/postfix/main.cf.j2
…
myhostname = {{ ansible_fqdn }}
relayhost = {{ mail_gateway }}
…
```

Feltételek I.

- „when”
  - „and”, „or”, zárójelezés, „and” megadható listaként

```
when:
  - ansible_distribution == "Debian"
  - ansible_virtualization_role == "guest"
```

  - Filterek természetesen használhatók

```
when:
  - ansible_distribution|lower == "debian"
  - ansible_virtualization_role == "guest"
```

Feltételek és import_tasks, roles

- Az ansible a feltételt az importált task, role minden task elemére alkalmazza!

```
tasks:
  - import_tasks: tasks/debian.yml
    when: ansible_os_family == 'Debian'


tasks:
  - import_role: name=debian
    when: ansible_os_family == 'Debian'
```

Feltételek és include_tasks

- Az ansible a feltételt csak az include_tasks, include_role műveletre alkalmazza:

```
tasks:
  - include_tasks: tasks/debian.yml
    when: ansible_os_family == 'Debian'
```

Feltételes var fájl include

- „vars_files” lista esetén az első talált fájlt használja

```
- hosts: all
  vars_files:
    - common.yml
    - [ "{{ ansible_os_family|lower }}.yml",
        "os_default.yml" ]
```

Első találat copy, template esetén

- vars_files-hoz hasonló

```
- name: Copy sshd_config
  copy: src={{ item }} dest=/etc/ssh/sshd_config
  with_first_found:
    - files:
      - "sshd_config.{{ ansible_hostname }}"
      - sshd_config.default
```

Loops

- „when” minden iterációnál kiértékelődik
- Standard loop: „with_items”:
  - Listaelem bármi lehet
- with_nested
- with_dict
- with_first_found
- with_fileglob
  - „etc/foo/conf.d/*”
- with_lines
  - „cat /etc/aliases”

Hibakezelés I.

- Parancsvégrehajtás hibájának figyelmen kívül hagyása:
  - ignore_errors: yes
- Mit tekintsünk hibának:
  - failed_when: result.rc == 0 or result.rc >= 2
- Mit tekintsünk változásnak:
  - changed_when: false

Hibakezelés II.

- Saját init script kezelése :-)

```
- name: Check that init script already enabled
  shell: /bin/ls /etc/rc?.d/*myservice
  register: ls_rc_myservice
  ignore_errors: true
  changed_when: false
- name: Enable init script
  command: /usr/sbin/update-rc.d myservice defaults
  when: ls_rc_myservice|failed
```

Ansible vault I.

- Jelszavak biztonságos/titkosított tárolása
  - .gitignore és git-ben nem tárolt var fájl(ok)
  - ansible vault
- Vault: bármi titkosítható
  - String
  - Strukturált adat
  - Tetszőleges fájl (copy, template)
  - YAML fájlban tárolható titkosított érték
- Szükséges: „pip install -U cryptography”

Ansible vault, fájl műveletek

- Create/encrypt/decrypt/view/edit

```
$ cat vars.yml
---
var1: foo
var2: bar
$ ansible-vault --vault-id secretfile encrypt vars.yml
$ cat vars.yml
$ANSIBLE_VAULT;1.1;AES256
64383434303531326432666665643862386439313632383239333539373438623230323638313234
66666366656530616134663438623165643137326430313330a6565653838356331386133326264363
23226439353631313737303931353930613265343466343034373332646663343763636437623562
6263346163646662370a393238373136636539326134333866396464663766343335383861656637
37326430613133616239633131303962653330383438313636323766393738323262
```

Ansible vault, string műveletek

- encrypt_string

```
$ ansible-vault encrypt_string --vault-id vault_id --name
'somevar' 'value'

somevar: !vault |

          $ANSIBLE_VAULT;1.1;AES256
346332346230643933383537366239366565393366393061386535373863646
5343437636530656 2
653431353636373461326565373436663766313165666532 0a6463343538623
3343365393561646 2
373232613831373661616634353333316636636532623165393634356338396
1636330313736616 2
386638323361303038 0a396536353861346139643734306338396462336332 6
362613234373539 34
```

Vault ajánlott használata

- Pl. group_vars/groupname fájl helyett:
  - group_vars/groupname/vars

```
---
var1: foo
passwd1: "{{ vault_passwd1 }}"
```

  - group_vars/groupname/vault

```
---
vault_passwd1: !vault | ...
```

Tagek

- A taskok, role-ok tagelhetők
  - Tetszőleges számú tag lehet
  - A handlereknek a megfelelő taggel rendelkezni kell
  - A tag segítségével a playbook lejátszása hoston belül leszűkíthető!

```
tasks:
  - import_role:
      name: apache2
    tags:
      - apache2
      - httpd
```

Ansible Galaxy

- Egymással megosztható role-ok gyűjteménye

```
$ ansible-galaxy search slapd

Found 50 roles matching your search:

 Name                      Description
 ----                      -----------
 debops.slapd              Manage slapd - OpenLDAP server
 …
```

Komplex rendszerek kezelhetők

- Dinamikus inventory

```
# etc/postfix/relay_domains.j2
{% for t in hostvars.gw.tenant._data %}
{% if hostvars.gw.tenant.data[t].state == 'present' %}
{% set relay = hostvars.gw.tenant.data[t].ansible_ssh_host %}
{% for d in hostvars.gw.tenant.data[t].domains %}
{{ d }} relay:[{{ relay }}]
{% endfor %}
{% endif %}
{% endfor %}
```

Nem elég? :-)

- Bővíthető
  - `{{ (var['foo'] is defined)|ternary(var['foo'], bar) }}`
  - `{{ var | dict_exists(foo, bar) }}`

```
# filter_plugins/myfilter.py

…

def dict_exists(d, e, defval):
  if e in d:
    return d[e]
  else:
    return defval

...
```

Best practices

- Kövessük az ajánlott könyvtár struktúrák valamelyikét
- Használjunk csoport, host változókat
- Használjunk role-okat
- Használjunk tag-eket
- Mindig nevezzük meg a task-okat
- Ne bonyolítsuk túl
- Használjunk verziókezelő rendszert