#ANSIBLEAUTOMATES

Automatyzacja Ansiblem - poza zarządzaniem konfiguracją systemów. Opowieści z pola bitwy.

Dariusz Puchalak
IT Consultant and Trainer
OSEC

redhat | ANSIBLE

Dariusz Puchalak

- 20+ lat Linux/Unix Sysadmin
- 10+ lat trener
- 4+ lat w OSEC

http://www.OSEC.pl

- Od 2009 na rynku
- doświadczona kadra (ACNI, RHCA)
- specjalizacja open-source
- subskrypcje, szkolenia, konsultacje

Czy automatyzacja to „tylko” zarządzanie konfiguracją?

Czy automatyzacja to „tylko” zarządzanie konfiguracją?

- Czy można zlecić wymianę routera prawie dowolnej osobie?
  - Upgrade firmwareu?
  - Wgranie właściwej konfiguracji?
  - Dodanie „niewspieranych” elementów konfiguracji?
  - Fizyczną wymianę sprzętu?

ubiquity-edgemax-initialize

DEMO

Ansible pod kątem Disaster Recovery

- ???

DR z Ansible

DEMO:

```
ansible-inventory -i hosts-demo2 –graph
ansible-inventory -i hosts-demo2 –host=kapral.fw.corp
ansible-inventory -i hosts-demo2 –list
ansible-inventory -i hosts-demo2 --graph --vars
```

Dokumentacja do DR z Ansible?

DEMO:

ansible-playbook production.yml -i hosts-demo2 –list-tags

ansible-playbook production.yml -i hosts-demo2 --list-tasks --tags shorewall

DR z Ansible jako narzędzie do wdrażania nowych systemów.

DEMO:

ansible-playbook production.yml -i hosts-demo2 –skip-tags check_mk,bareos

"Houston, we've had a problem here."

```
#Check if we can install 32bit steam client and install it.
- block:

  - name: Check for i386 architecture
    command: dpkg --print-foreign-architectures
    register: dpkg_architecture
    changed_when: False
    check_mode: no

  - name: Display dpkg architecture (dpkg --print-foreign-architectures)
    debug:
      var: dpkg_architecture
      verbosity: 1

  - block:
    - name: Add i386 architecture to dpkg
      command: dpkg --add-architecture i386

    - name: Update apt-get database
      apt:
        update_cache: yes

    when: '"i386" not in dpkg_architecture.stdout'
```

"Houston, we've had a problem here."

DEMO

```
puchalakd@neptune:~/Ansible-demo$ ansible-playbook playbook-debugger.yml

PLAY [localhost] ***************************************************************

TASK [wrong variable] **********************************************************
fatal: [localhost]: FAILED! => {"msg": "The task includes an option with an undefined variable
. The error was: 'wrong_var' is undefined\n\nThe error appears to have been in '/home/puchalak
d/Ansible-demo/playbook-debugger.yml': line 7, column 7, but may\nbe elsewhere in the file dep
ending on the exact syntax problem.\n\nThe offending line appears to be:\n\n  tasks:\n    - na
me: wrong variable\n      ^ here\n"}
[localhost] TASK: wrong variable (debug)>
```

Czy nam się systemy rozjechały?

time ANSIBLE_STDOUT_CALLBACK=actionable ansible-playbook production.yml --user rootdp --skip-tags bareos --limit some-linux -CD

Przykładowy wynik działania podczas pokazu u jednego z klientów to: 980 linii kontra 5563 :)

```
---
- hosts: windows
  gather_facts: yes

  tasks:
  - debug:
      msg: "Reboot following {{ inventory_hostname }} Windows systems."
    when: ansible_reboot_pending

- hosts: windows
  gather_facts: yes
  serial:
    - 1
    - 3
    - 5%
  max_fail_percentage: 10

  vars_prompt:
  - name: temp
    prompt: Press Enter to reboot selected Windows systems!!!
    default: 'Enter'
    private: no

  tasks:
```

```
  - name: Only when reboot needed
    block:
    - name: Schedule downtime in nagios
      nagios:
        action: downtime
        minutes: 10
        service: host
        host: '{{ inventory_hostname }}'
        comment: "Reboot scheduled"
      delegate_to: "{{ check_mk_server }}"

    - win_reboot:

    - name: wait for host/hosts to start up
      wait_for_connection:
        timeout: 600

    - name: start Windows time service
      win_service:
        name: W32Time
        start_mode: auto
        state: started

    - name: Delete downtime from nagios
      nagios:
        action: delete_downtime
        service: host
        host: '{{ inventory_hostname }}'
  ## comment is essential
        comment: "Reboot scheduled"
      delegate_to: "{{ check_mk_server }}"
      tags: test

    when: ansible_reboot_pending
```

```
"Ansible-demo/Windows_reboot.yml" 58L, 3471C written                    31,52         Top
```

Dynamicznie dodajmy hosta.

```
- name: playbook
  gather_facts: no
  hosts: localhost
#  become: true

  vars_prompt:
  - name: target
    prompt: "Put servername"
    private: no

  tasks:
  - add_host:
      hostname: "{{ target }}"
      group: test
      ansible_host: "{{ target }}"
      ansible_port: 22

- name: playbook
  gather_facts: yes
  hosts: test

  tasks:
  - debug:
      msg: "Hostname: {{ ansible_hostname }}"
~
~
~
                                                1,1           All
```

Gdy sieć nie pozwala – OpenSSH na ratunek.

.ssh/config:

```
...
Host hostB
        ProxyCommand ssh hostA nc %h %p
        HostName 10.1.8.31
Host hostA
        HostName 172.16.48.10
...
```

```
bash$ ssh hostB
```

Gdy sieć nie pozwala – OpenSSH na ratunek.

.ssh/config:

```
Include config-*
```

.ssh/config-important.corp:

```
Host *.important.corp
  ProxyCommand ssh jumphost.corp nc %h %p
  IdentitiesOnly yes
  IdentityFile ~/.ssh/important.corp/id_ecdsa
  ControlMaster auto
  ControlPersist 30m
  ControlPath ~/.ssh/ControlPath/%C
  UserKnownHostsFile ~/.ssh/known_hosts ~/.ssh/known_hosts_important.corp
```

Gdy sieć nie pozwala – OpenSSH na ratunek.

.ssh/config-dom:

```
Host unison
  User puchalakd

Match Host unison !exec "host -t A unison.net.puchalak"
  HostName 192.168.1.125
  ProxyCommand ssh pluton-remote nc %h %p

Host unison
 HostName 192.168.1.125
```

Gdy sieć nie pozwala – OpenSSH na ratunek.

Crontab systemu gdzieś na obrzerzach internetu bez dostępu po VPNie za NATem.

```
* * * * * flock -w 0 -n /root/.call.back ssh -i /root/.call.back -o BatchMode=yes -o ExitOnForwardFailure=yes -R 12345:localhost:22 puchalakd@somewhere.internet || exit 0
```

.ssh/config:

```
Host private.behind.nat
  HostName 127.0.0.1
  Port 12345
  ProxyCommand ssh pluton-remote nc %h %p
```

Gdy sieć nie pozwala – OpenSSH na ratunek.

man ssh_config

DynamicForward 1080

Specifies that a TCP port on the local machine be forwarded over the secure channel, and the application protocol is then used to determine where to connect to from the remote machine.

Gdy sieć nie pozwala – OpenSSH na ratunek.

http://docs.ansible.com/playbooks_environment.html

It is quite possible that you may need to get package updates through a proxy, or even get some package updates through a proxy and access other packages not through a proxy.

Gdy sieć nie pozwala – OpenSSH na ratunek.

http://docs.ansible.com/playbooks_environment.html

```
- hosts: all
  remote_user: root

  tasks:
    - apt: name=cobbler state=installed
      environment:
        http_proxy: http://proxy.example.com:8080
```

Gdy sieć nie pozwala – OpenSSH na ratunek.

https://docs.ansible.com/ansible/latest/user_guide/playbooks_async.html

To avoid blocking or timeout issues, you can use asynchronous mode to run all of your tasks at once and then poll until they are done.

Pomysł: Asynchroniczne taski z ssh

DynamicForward do systemu gdzie nie ma blokady internetu

RemoteForward z portem na którym zrobiliśmy DynamicForward

environment: localhost:RemoteForwardPort

Wynik: nielimitowany internet dla systemu w dowolnym miejscu.

Najlepsze praktyki.

- Role
  - Budujcie role
  - Role powinny robić tylko jedną rzecz
  - Uwaga na konflikt nazw zmiennych
  - Podwójna uwaga na handlery
  - Rola może i powinna być uniwersalna (ale nie zawsze wiele dystrybucji, Linux, Windows, urządzenia sieciowe ogarnięte w jeden roli mają sens. Lepiej użyć zależności lub include_role/import_role).

Najlepsze praktyki.

- Tags
  - Używać zawsze i wszędzie :)
  - Testować w trybie dry-run
  - Testować poprzez uruchomienie tylko z danym tagiem (–tags testowany)
  - Pamiętać o dobrym otagowaniu zadań z modułami non-idempotent (np.. shell, raw, command). Oraz jeśli nic nie zmieniają to dodaniu check_mode: no .
  - Min 2-3 tagi:
    - Ogólny do zadania
    - Szczegółowy dot. danego zadania

Łączyc kod dla systemów Linuksowych.

```
# Load a variable file based on the OS type, or a default if not found.
- include_vars: "{{ item }}"
  with_first_found:
    - "../vars/{{ ansible_distribution }}-{{ ansible_distribution_major_version | int}}.yml"
    - "../vars/{{ ansible_distribution }}.yml"
    - "../vars/{{ ansible_os_family }}.yml"
    - "../vars/package_default.yml"
  when: package_prerequisites is not defined
  tags: ['ubertooth' ]

- name: Install ubertooth prerequisites
  package:
    name: "{{ item }}"
    state: latest
  with_items: "{{ package_prerequisites }}"
  tags: ['ubertooth' ]
```

Oddzielnie traktować Linuksy i Windowsy.

```
---

# Include variables
- name: gather os specific variables
  include_vars: "{{ item }}"
  with_first_found:
    - files:
        - "{{ ansible_distribution }}-{{ ansible_distribution_major_version}}.yml"
        - "{{ ansible_distribution }}.yml"
        - "{{ ansible_os_family }}.yml"
      skip: true # Skip if no var files found
  tags: [ vars, 'check_mk-agent', 'check_mk-server' ]

# Check-mk-agent
- { include: check-mk-agent-linux.yml, when: ansible_system is defined and ansible_system == "Linux" }
- { include: check-mk-agent-windows.yml, when: ansible_system is defined and ansible_system == "Win32NT" }
- { include: check-mk-agent-generate-config.yml , when: ansible_system is defined and ansible_system == "Linux" or ansible_system == "Win32NT" }
```

Oddzielnie traktować Linuksy i Windowsy.

```
puchalakd@neptune:~/Ansible-demo$ ansible -m setup w2k12 | grep reboot
        "ansible_reboot_pending": false,
puchalakd@neptune:~/Ansible-demo$ ansible -m setup localhost | grep reboot
puchalakd@neptune:~/Ansible-demo$

root@pluton:~# needrestart -rl -pk ; echo $?
CRIT - Kernel: 4.9.0-7-amd64!=4.9.0-8-amd64 (!)|Kernel=2;0;;0;2
2
```

Urządzenia sieciowe

- Traktować każdą rodzinę jak osobną bajkę.
- Używać Ansible 2.5+

Pluginy

- Przeglądać dokumentacje pluginów. Można znaleźć coś ciekawego np:

https://docs.ansible.com/ansible/2.5/plugins/callback/selective.html

This callback only prints tasks that have been tagged with print_action or that have failed. This allows operators to focus on the tasks that provide value only.

Jinja filters.

```
# SSH access for admins
SSH/ACCEPT:$LOG         any:{{ admin_ips | join(",") }}   fw
# SSH access for monitoring
SSH/ACCEPT:$LOG         any:{{ check_mk_servers | join(",") }}    fw
# Proxy for apt-get/yum to access repositories
ACCEPT:debug    fw      any:{{ proxy_repo | urlsplit('hostname') }}       tcp     {{ proxy_repo | urlsplit('port') }}
# Syslog remote
ACCEPT:debug    fw      any:{{ rsyslog_server }}        tcp     514
# system mails
ACCEPT:debug    fw      any:{{ mail_relay }}        tcp     25
# NTP
ACCEPT:debug    fw      any:{{ ntp_servers | join(",") }} udp ntp

                                                                                1,1           All
```

Jinja filters.

```
- name: Create remote users for share Umowy
  win_user:
    account_disabled: no
    account_locked: no
    name: "{{item.name}}"
    fullname: "{{item.fullname if item.fullname is defined else ''}}"
    description: "{{item.description if item.description is defined else ''}}"
    groups: "{{item.groups | join(',')}}"
    password: "{{item.password if item.password is defined else default_password}}"
    groups_action: add
    password_expired: no # Don't require user to change password on first login
    password_never_expires: yes
    state: present
    update_password: on_create # If it exists do not update password
    user_cannot_change_password: yes
  with_items: "{{ users }}"
  tags: [users, config, windows]
-- INSERT --                                                    1,44          Top
```

Jinja filters.

```
##### data from host_vars/system.dns.name/local_users.yaml
default_password: Pa$$w0rd

- name: Delete old users
  win_user:
    name: "{{item.name}}"
    state: absent
  with_items: "{{ users_deleted }}"
  tags: [users, config, windows]
```

```
users:
  - fullname: Joe Test
    name: test1
    groups: test1, test2
    description: Test user 1
  - fullname: Joe Doe
    name: test2
    groups: test2, admins
    description: Test user 2

users_deleted:
  - name: Jan Testowy
```

```
-- INSERT --                                          34,1          65%
```

Pytania?

Dariusz.Puchalak@osec.pl