

Debian i Ansible, czyli jak automatyzować stare RHELE/OELE.

Dariusz Puchalak

10 października 2023

- ▶ 25+ lat Linux/Unix Sysadmin
- ▶ 15+ lat trener
- ▶ 9+ lat w OSEC
- ▶ 9+ lat z Ansible

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

Ścieżka edukacyjna Ansible.

- ▶ Lista aktualnych kursów, egzaminów:
<https://osec.pl/red-hat-ansible-automation/>
- ▶ RH294 - RHCE - Podstawowy kurs Ansible
- ▶ DO417 - Ansible dla inżynierów Windows (alternatywa dla RH294)
- ▶ DO457 - Ansible dla inżynierów sieciowych (alternatywa dla RH294)
- ▶ RH358 - „Dawne RH300 @RHEL7”, czyli usługi Linuksowe ręcznie i Ansiblem
- ▶ DO374 - „Dawne Advanced Ansible”, czyli zaawansowane elementy Ansible (kontynuacja RH294)
- ▶ DO467 - „Dawny Tower”, czyli jak używać Ansible Automation Platform

Spis treści.

Kontener z Ansible.

Własny kontener z Ansible na Debianie.

Pytania?

Ansible, Python a dystrybucje Linuksa.

- ▶ Ansible 2.12 - koniec wsparcia dla Pythona 2 na kontrolerze
- ▶ Ansible 2.13 - koniec wsparcia dla Pythona 2.6 na zarządzanych systemach (RHEL 6 i klony)
- ▶ Ansible 2.14 - koniec wsparcia dla Pythona 3.8 na kontrolerze
- ▶ Ansible 2.16 - koniec wsparcia dla Pythona 3.9 na kontrolerze i Pythona 3.5 na zarządzanych systemach
- ▶ Ansible 2.17 - koniec wsparcia dla Pythona 2 i 3.6 na zarządzanych systemach (RHEL 7 i klony, Debian 8)

Po co?

- ▶ U mnie działa. 8]
- ▶ Nie mam starych systemów które chcę automatyzować.
- ▶ Nie chcę korzystać z nowszych wersji Ansible.
- ▶ Nie korzystam z Ansible.

Use cases.

- ▶ Kapsuła czasowa na konkretną wersję Ansible, Pythona, bibliotek Pythonowych, innego oprogramowania - do utrzymania ciągłości automatyzacji systemów które „wypadną ze wsparcia” nowszych wersji
- ▶ Jak u mnie działa, to musi i u Ciebie 8] - czyli zapakowanie wszystkich zależności (biblioteki, programy, kolekcje) w kontener.
- ▶ Łatwe użycie różnych wersji Ansible/Kolekcji.
- ▶ Łatwość z jaką można „strzelać” do zarządzanych systemów Ansiblem lokalnie, zamiast tylko z centralnej lokalizacji.
- ▶ Bezpieczeństwo - lepsza izolacja uruchamianych zadań.

EE - Execution Environments

- ▶ Ansible w kontenerze(ach).

Przydatne narzędzia do Ansible EE.

- ▶ ansible-navigator
- ▶ ansible-builder
- ▶ podman/docker
- ▶ ansible-runner

- ▶ Część RH AAP
- ▶ można używać standalone (bez RH AAP)
- ▶ daje TUI do pracy z Ansiblem (uruchamianie)
- ▶ daje nam dodatkową „interaktywność” w przeglądaniu co wyników działania playbooka
- ▶ pozwala łatwo skorzystać z Ansible Execution Environment (ale tego nie wymusza)

▶ DEMO

```
---
ansible-navigator:
#   enable-prompts: False
#
  execution-environment:
    container-engine: podman
    enabled: True
    image: localhost/ee-debian12-full:2.15-20230922
#   pull:
#     arguments:
#       - "--tls-verify=false"
#     policy: never
```

```
volume-mounts:
  - src: "/mnt/dokumentacja"
    dest: "/mnt/dokumentacja"
    options: "Z"
logging:
  level: warning
#   append: False
  file: logs/ansible-navigator.log
#   mode: stdout
playbook-artifact:
  enable: True
  save-as: logs/{playbook_name}-artifact-{time_stamp}.json
```

- ▶ Rozwiązanie do modyfikacji i tworzenia własnych kontenerów gł. dla ansible-navigatora/ansible-runnera.

- ▶ opisuje jak chcemy zbudować kontener:
 - ▶ co wybrać jako obraz źródłowy
 - ▶ wersja pythona
 - ▶ wersja ansible-core
 - ▶ wersja ansible-runnera
 - ▶ kolekcje Ansiblowne i ich wersje
 - ▶ pakiety z dystrybucji
 - ▶ pakiety Pythonowe (instalowane przez pip)
 - ▶ dodatkowe wymagania (przez hooks i additional_build_steps)

ansible-builder - execution_environment.yml

```
---
version: 3
images:
  base_image:
    name: quay.io/centos/centos:stream9
dependencies:
  python_interpreter:
    package_system: python39
    python_path: /usr/bin/python3.9
  galaxy:
    collections:
      - ansible.utils
additional_build_files:
  - src: files/rootCA.crt
    dest: configs
```

ansible-builder build

- ▶ na podstawie pliku `execution_environment.yml` tworzy środowisko do budowanie kontenera
- ▶ następnie buduje kontener
- ▶ wrzuca odpowiednią wersję Ansible, kolekcje, biblioteki, oprogramowanie zgodnie z definicjami w `execution_environment.yml`

ansible-builder create

- ▶ na podstawie pliku `execution_environment.yml` tworzy środowisko do budowanie kontenera (katalog context)
- ▶ teraz możemy sobie pooglądać jak to wygląda od środka

ansible-builder - fazy budowania (context/Containerfile)

- ▶ Base - pobrany obraz bazowy, ew. instalacja pythona, pip, ansible-runner, ansible-core/ansible
- ▶ Galaxy - instalacja kolekcji razem z zależnościami
 - ▶ context/_build/scripts/check_galaxy (weryfikacja instalacji ansible-galaxy)
- ▶ Builder - instalacja reszty (zdefiniowane przez nas zależności)
 - ▶ context/_build/scripts/introspect.py
 - ▶ context/_build/scripts/assemble (budowanie zależności i ich instalacja, sprzątnięcie)
- ▶ Final - zebranie pierwszych 3 faz razem i wygenerowanie
 - ▶ context/_build/scripts/check_ansible (weryfikacja instalacji ansible, ansible-runner)
 - ▶ context/_build/scripts/install-from-bindep

ansible-builder 3.0

- ▶ „Fix common issues of build context hacking [...] We want users to version the builder definition file instead of the Containerfile or the builder context.”
- ▶ „Single-file definition”
- ▶ „Consolidate ansible-builder components”
- ▶ „Ansible-builder version 3 allows you to specify any base image, any architecture, any Python version or any platform. „

Za <https://www.ansible.com/blog/unlocking-efficiency-harnessing-the-capabilities-of-ansible-builder-3.0>

- ▶ Faktycznie da się wszystko zmieścić w `execution-environment.yml` :)
- ▶ Ale nie do końca można użyć dowolnego kontenera jako bazy. :(
 - ▶ `context/Containerfile` - mocne założenia że będzie budować na „RHELish distro image”
 - ▶ Skrypty z `context/_build/scripts/` - mocne założenia że będzie działać na „RHELish distro image”
 - ▶ <https://github.com/ansible/ansible-builder/issues/553>
 - ▶ <https://github.com/ansible/ansible-builder/pull/543>

ansible-builder dla obrazu Debianowego

Jaki są problemy:

- ▶ Python - różnice w wersji Pythona, sposobach instalacji i bibliotek i programów w dystrybucji np.
 - ▶ brak pythona w obrazie [docker.io/library/debian:bookworm](https://hub.docker.io/library/debian:bookworm)
 - ▶ „ensurepip is disabled in Debian/Ubuntu for the system python.”
 - ▶ pip vs pipx
 - ▶ site-packages vs dist-packages
 - ▶ PEP 668 – {} Marking Python base environments as “externally managed”
- ▶ Należałości RHELowe.

ansible-builder dla obrazu Debianowego

Brak Pythona w minimalnym obrazie rozwiązujemy przez:

```
additional_build_steps:  
  prepend_base: |  
    RUN apt update ; apt install -y python3-pip python3-full
```

Domyślny python ustawiamy przez:

```
dependencies:  
  python_interpreter:  
    package_system: "python3"
```


ansible-builder dla obrazu Debianowego

Naleciałości RHELowe rozwiązujemy (dla Debiana 12) poprzez:

- ▶ nieużywanie ensurepip
- ▶ dodanie do pip install opcji `--break-system-packages`

Patrz skrypt: `ansible-builder_context_fix.sh`

ansible-builder dla obrazu Debianowego

```
ansible-builder_context_fix.sh
```

```
#!/bin/bash
```

```
sed --in-place -e '/$PYCMD -m ensurepip/d' \
```

```
-e 's/RUN $PYCMD -m pip install /RUN $PYCMD -m pip install
```

```
--break-system-packages /\
```

```
-e 's/$PIPCMD install $CONSTRAINTS/$PIPCMD install --break-system-packages
```

```
$CONSTRAINTS/\
```

```
context/Containerfile context/_build/scripts/assemble
```

```
context/_build/scripts/install-from-bindep
```

ansible-builder dla obrazu Debianowego

Workaround:

- ▶ 3 krokowe budowanie obrazu (zamiast 1 krokowego)

- ▶ Wygenerowanie środowiska do budowania konteneru

```
ansible-builder create --file ee-ansible_full_2.15.yml
```

- ▶ pozbycie się naleciałości RHELowych i zastąpienie ich Debianowymi

```
ansible-builder_context_fix.sh
```

- ▶ zbudowanie konteneru

```
cd context ; podman build -t ee-debian12-full:2.15-20230922 .
```

ansible-builder dla obrazu Debianowego

Co tracimy:

- ▶ Część informacji odnośnie obrazu w ansible-navigator images

Co zyskujemy:

- ▶ Znane nam środowisko Debianowe z Ansiblem, wszelkimi niezbędnymi nam narzędziami w kontenerze

execution-environment.yml - dla obrazu Debianowego

```
---
version: 3
dependencies:
  python:
    - pywinrm
  [...]
  system:
    - python3 [platform:dpkg]
  [...]
    - bind9-host [platform:dpkg]
    - csvkit [platform:dpkg]
  galaxy:
    collections:
      - debops.debops
  [...]
```

execution-environment.yml - dla obrazu Debianowego

```
ansible_core:
  package_pip: ansible
ansible_runner:
  package_pip: ansible-runner
python_interpreter:
  package_system: "python3"
images:
  base_image:
    name: docker.io/library/debian:bookworm
options:
  package_manager_path: /usr/bin/apt
additional_build_steps:
  prepend_base: |
    RUN apt update ; apt dist-upgrade -y ; apt install -y python3-pip
```

Kontener Debianowy z ansible-navigator

▶ DEMO

- ▶ To co używa Ansible Tower do uruchamiania ansible/ansible-playbook.
- ▶ Interfejs CLI i Pythonowy.
- ▶ Pozwala uruchamiać playbooki, role, moduły w AEE, jak i poza nim
- ▶ Pozwala wysyłać zdarzenia na zewnątrz
- ▶ Pozwala uruchamiać automatyzację Ansible na innych hostach.

Zdalne uruchamianie playbooków.

- ▶ `ansible-runner transmit . -p runner-demo.yml | pv | ssh pluton.sasiedzka.puchalak.net /home/puchalakd/.local/bin/ansible-runner worker --private-data-dir /tmp/Ansible/ | ansible-runner process .`

Przykłady

Przykłady: `https://dariusz.puchalak.net./Ansible/`

Pytania?

Dariusz.Puchalak@osec.pl

Dziękuję.
