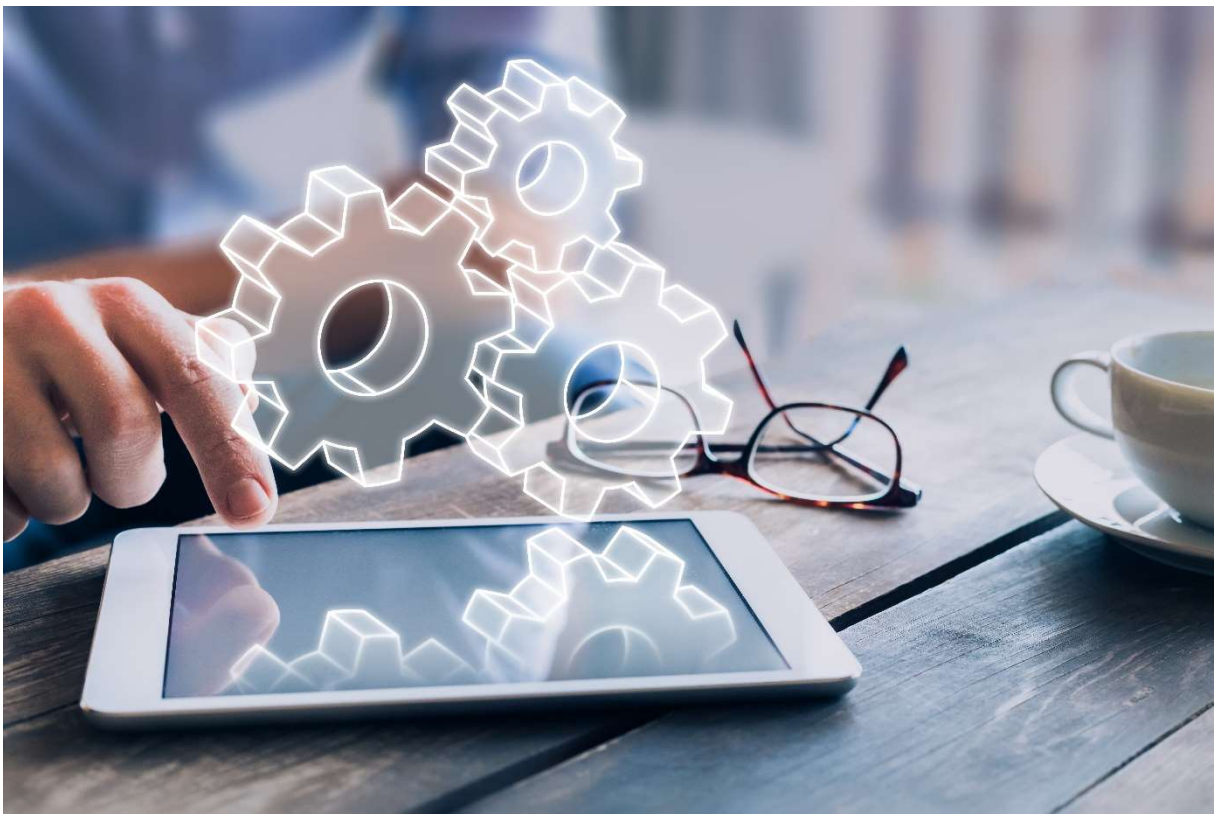
5 NARZĘDZI NETDEVOPS

DZIĘKI KTÓRYM ŁATWO WKROCZYSZ W ŚWIAT AUTOMATYZACJI SIECI



Nowoczesny Sieciowiec

Cześć,

Jest mi niezmiernie miło, że zdecydowałeś się ściągnąć niniejszy dokument i poświęcić chwilę na jego przeczytanie. Oznacza to, że myślisz poważnie o staniu się nowoczesnym sieciowcem :-) Zadbałem o to, aby dokument ten był krótki, żeby nie tracić Twojego cennego czasu, ale też jednocześnie, żeby był dla Ciebie jak najbardziej wartościowy.

Ansible, Git, REST, CICD, Jenkins, zapewne słyszałeś niejednokrotnie przynajmniej część z tych pojęć. Świat IT wkroczył bardzo mocno w automatyzację i programowanie wszystkiego. Proces ten zatacza coraz większe kręgi i rozwija się coraz dynamiczniej. Nam, sieciowcom, wciąż jest bardzo ciężko się w tym trendzie odnaleźć. O jaki trend chodzi? Oczywiście o DevOps. Nie uciekniemy przed nim, ale możemy mu stawić czoła i nie jest to wcale trudne.

DevOps nie jest pojęciem nowym. W wielu organizacjach jest ono bardzo dobrze znane, a metodologia i narzędzia z nim związane wykorzystywane są na porządku dziennym. Jest to trend, który ma zbliżyć do siebie świat programistów i świat infrastruktury poprzez automatyzację i uwspólnianie całego procesu tworzenia, modyfikacji i utrzymania usług, a w szczególności aplikacji.

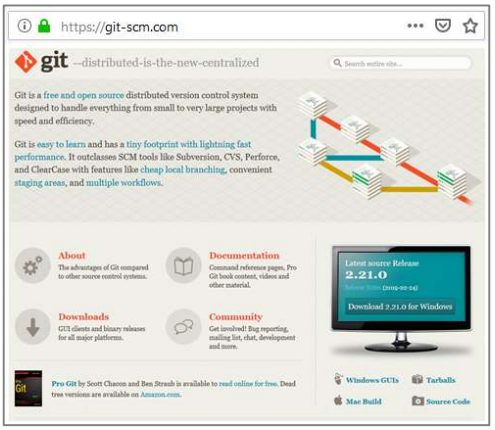
Jakiś czas temu pojawiło się też pojęcie NetDevOps. Czym ono się wyróżnia? Otóż, do tej pory dział utrzymania rozumiany był jako zespół zarządzający infrastrukturą systemowo-serwerową. Ale, infrastruktura to nie tylko serwery. Jest to też sieć, bez której żadna aplikacja w dzisiejszych czasach nie ma racji bytu. Wkomponowanie w procesy DevOps elementów sieciowych nie jest trywialne. Podobnie jak w procesie DevOps, kluczową rolę odgrywają narzędzia. To dzięki nim możemy zacząć pracować nad wspólnym modelem usług, zaczynając od infrastruktury, przez system operacyjny, na aplikacji skończywszy. A najłatwiej jest zacząć od narzędzi, które są na pograniczu automatyzacji i programowania. Bo czym innym jest konfiguracja urządzeń sieciowych, jak nie ich programowaniem.

Pozwól, że przedstawię Ci poniżej 5 podstawowych narzędzi, z którymi warto się zapoznać, aby móc łatwo i skutecznie wkroczyć w świat automatyzacji sieci. Serdecznie zapraszam do lektury.

Krzysztof Załęski

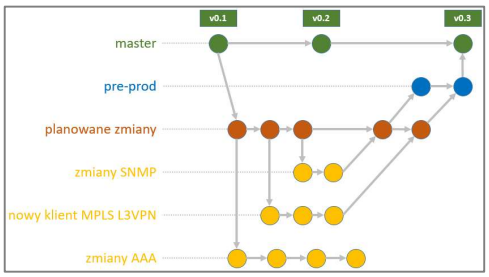
nowoczesnysieciowiec.pl

1. GIT (https://git-scm.com)



Git jest jednym z najbardziej popularnych, darmowych systemów wersjonowania. Służy on do śledzenia zmian w plikach tekstowych. Używany jest głównie przez programistów do zarządzania kodem źródłowym, ale ze względu na swoją elastyczność zdobywa coraz większe uznanie wśród sieciowców. Możemy go wykorzystywać przy modyfikacji zarówno plików tekstowych z konfiguracją urządzeń, ale też przy tworzeniu zaawansowanych mechanizmów automatyzacji z wykorzystaniem Ansible czy Jinja2 (o nich dalej). Git pozwala nam na łatwe zarządzanie zmianami, zachowując ich spójność oraz historię.

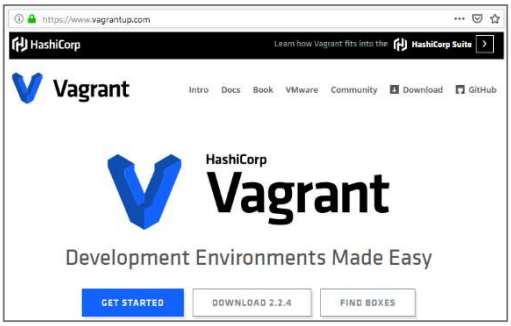
Git jest systemem rozproszonym. Oznacza to, że każdy administrator pracuje na własnej, lokalnej kopii plików, dokonując w nich zmiany, które później wysyłane są na serwer. W trybie takim Git daje największe możliwości, głównie w zakresie współpracy z innymi administratorami i systemami do automatyzacji. Może być też używany lokalnie na stacji roboczej bez konieczności instalowania serwera. Wszelkie zmiany w plikach bazują na tak zwanych migawkach (ang.: snapshot), a i ich historia przechowywana jest w postaci trzewa katalogów, dzięki czemu możemy w dowolnym momencie wrócić do poprzednich wersji.



Jedną z najczęściej wykorzystywanych funkcji Gita jest tworzenie gałęzi kodu. Idea działania tego mechanizmu polega w skrócie na tym, że nie pracujemy na gałęzi głównej (master), tylko tworzymy kopię konfiguracji (branch) pod inną nazwą i w niej dokonujemy zmiany (np. dodajemy nową usługę). Gałąź główna w centralnym repozytorium powinna zawierać tylko zatwierdzone i w pełni działające konfiguracje urządzeń sieciowych. Modyfikacje w nowej gałęzi mogą być niezależnie testowane i weryfikowane, a po ich zatwierdzeniu przez architekta, mogą być połączone z gałęzią główną (ang.: merge). W przypadku wystąpienia problemów z nową konfiguracją mamy możliwość odrzucenia, pozostawiając poprzednią wersję.

Git współpracuje z różnymi serwerami, na których przechowywane są centralnie repozytoria kodu. Może to być Git Server, GitLab czy GitHub.

2. Vagrant (https://www.vagrantup.com)



Vagrant to system, wspomagający proces uruchamiania i testowania konfiguracji systemów operacyjnych, aplikacji, a nawet niektórych urządzeń sieciowych, zanim zostaną one wprowadzone produkcyjnie. Sam z siebie, nie jest niezależnym narzędziem, a raczej nakładką na popularne systemy wirtualizacji, takie jak Oracle VirtualBox, VMware Fusion czy nawet Docker.

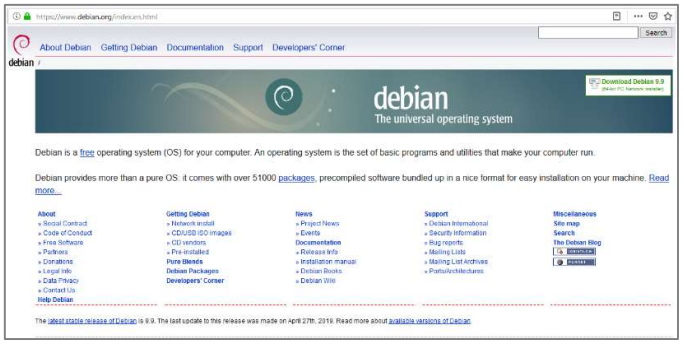
Odzwierciedlenie fizycznego środowiska produkcyjnego, na etapie testowania kodu zawsze było nie lada wyzwaniem. Bardzo trudno jest zapewnić infrastrukturę identyczną jak ta w produkcji, głównie ze względów finansowych. Na szczęście, większość środowisk może być zwirtualizowana. Dotyczy to systemów operacyjnych, baz danych, aplikacji, ale też i urządzeń sieciowych. Coraz więcej producentów udostępnia wirtualne obrazy, dając tym samym administratorom sieci możliwość wkomponowania się w proces DevOps'owy.

```
# -*- mode: ruby -*-

Vagrant.configure("2") do |config|
  config.vm.define "nxos1" do |nxos|
    nxos.vm.box = "nxos/7.0.3.I7.3"
    nxos.vm.post_up_message = "Nexus 9000v-1 is UP"

    nxos.vm.provider "virtualbox" do |vm|
      vm.memory = "4096"
      vm.gui = true
      vm.name = "nxos-1"
      vm.customize ['modifyvm', :id, '--uart1', '0x3F8', 4, '--uartmode1', 'disconnected']
      vm.customize ['modifyvm', :id, '--uart2', '0x2F8', 3, '--uartmode2', 'disconnected']
    end
  end
end
```

Utworzenie środowiska testowego może być jednak procesem dość długotrwałym i skomplikowanym. Wiąże się on z przygotowaniem konfiguracji w kilku różnych narzędziach (za pomocą interfejsu graficznego), zainstalowanie odpowiedniego systemu i wgraniu testowanej konfiguracji. Dzięki Vagrant'owi proces ten możemy skrócić i przede wszystkim oprogramować. Wygląd naszego środowiska opisujemy w pliku tekstowym zwanym Vagrantfile. Zawiera on informację o tym, jakie obrazy wirtualne będą uruchamiane, jakie będą ich parametry, a także w jaki sposób będą połączone ze sobą sieciowo. Pełne obrazy (tzw. box) systemów operacyjnych dostępne są publicznie. Obrazy urządzeń sieciowych można pobrać ze strony producentów. Całe środowisko jesteśmy w stanie uruchomić jednym poleceniem *vagrant up*.

Pliki tekstowe Vagrant'a łatwo jest wersjonować za pomocą Gita, a sama konfiguracja może być współdzielona między administratorami, co pozwala na utrzymanie jednolitego środowiska testowego. Jedną z zalet Vagranta jest możliwość automatycznego wgrania konfiguracji do uruchomionych urządzeń wirtualnych np. za pomocą Ansible. Dodatkowo, po zakończeniu pracy możemy skasować całe środowisko poleceniem *vagrant destroy*, pozostawiając czystą, niezaśmieconą stację roboczą.

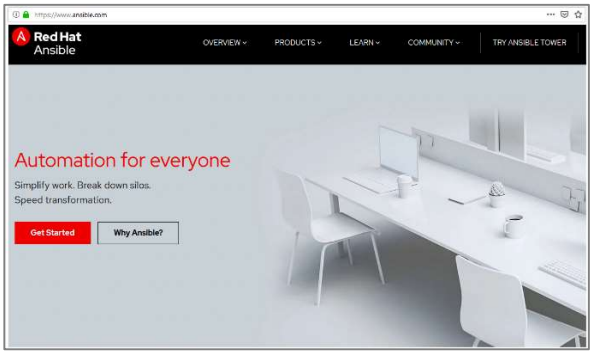
3. Linux



Tak, tak, Linux. Choć jest to systemem operacyjny, to w przypadku automatyzacji, programowania i całego procesu NetDevOps staje się narzędziem wręcz niezbędnym. Nie chodzi tu zmianę swojego systemu na stacji roboczej, a raczej zapoznanie się z nim w stopniu przynajmniej podstawowym. Można w tym celu wykorzystać takie narzędzie jak Vagrant, powołując lokalnie maszynę wirtualną z testowym Linux'em. Na publicznym repozytorium Vagrant'a dostępnych jest wiele obrazów dla różnych dystrybucji Linuxa, na których możemy się uczyć nie tylko samego systemu, ale również testować inne narzędzia NetDevOps'owe.

Dlaczego Linux? Część narzędzi, takich jak Git czy Vagrant dostępnych jest na większość popularnych systemów operacyjnych. Niestety, zdecydowana większość pozostałych narzędzi dostępna jest głównie dla systemów Linux'owych. Jednym z nich jest np. Ansible, bez którego trudno sobie wyobrazić automatyzację sieci. Dodatkowo, bardzo dużo usług serwerowych i aplikacyjnych uruchamianych jest pod Linux'em. Jako administratorzy sieciowi będziemy niejednokrotnie współpracowali z działami programistów czy systemowców, tworząc wspólne konfiguracje, więc wiedza na temat powłoki (ang.: shell), struktury katalogów, plików konfiguracyjnych i podstawowych narzędzi (np. vim) staje się niezbędna (a na pewno przydatna).

4. Ansible (https://www.ansible.com)



Ansible, to jeden z najbardziej rozpowszechnionych, darmowych systemów do automatyzacji konfiguracji. Jego popularność wynika głównie z tego, że nie wymaga on żadnego agenta zainstalowanego na docelowym urządzeniu. Konfiguracja odbywa się za pomocą SSH. Dzięki temu jest on bardzo uniwersalny i wpasowuje się idealnie w środowisko sieciowe. Na chwilę obecną jest on dostępny tylko na systemy Linuxowe.

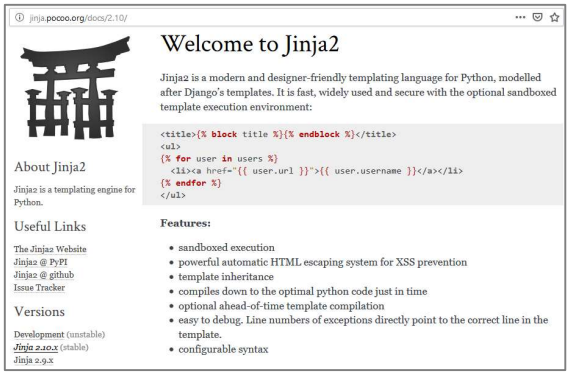
Ansible pracuje na plikach tekstowych (łatwe wersjonowanie za pomocą Gita), w których opisany jest cały proces (tzw. playbook), nie tylko konfiguracji, ale też sposobu jej testowania. Wykorzystuje on format zapisu danych w postaci YAML. Konfiguracja może być podzielona na funkcjonalne składowe, co czyni Ansible systemem modularnym i bardzo elastycznym. Ma on wbudowaną bardzo dużą ilość dodatków do obsługi systemów operacyjnych, usług serwerowych, aplikacji, a także urządzeń sieciowych i dostawców chmury publicznej.

```
---
- name: Configure NXOS
  hosts: nxos
  connection: local
  gather_facts: no
  tasks:

  - name: Enable features
    nxos_feature:
      feature: "{{ item.feature }}"
      provider: "{{ nxapi }}"
    with_items: "{{ features }}"

  - name: L3 interfaces
    nxos_interface:
      interface: "{{ item.interface }}"
      description: "{{ item.desc }}"
      admin_state: up
      mode: layer3
      provider: "{{ nxapi }}"
    with_items:
      - "{{ l3_if }}"
```

Największą jednak zaletą Ansible jest idempotentność. Oznacza to, że proces konfiguracyjny uruchomiony wiele razy daje nam zawsze ten sam wynik. W przypadku różnego rodzaju skryptów, a nawet zwykłego wklejania konfiguracji, nałożenie się lub dodanie niektórych elementów może spowodować nieoczekiwany skutek. Ansible, weryfikuje każdą zmianę i jeżeli dany wpis już istnieje to go ignoruje. Możemy w ten sposób testować wielokrotnie nasz playbook, wprowadzając tylko nowe zmiany.

Ansible jest bardzo intensywnie wykorzystywany przez administratorów systemów operacyjnych i aplikacji. Mając wspólne narzędzie możemy stworzyć spójną konfigurację usługi, uwzględniając wszystkie jej elementy, łącznie z infrastrukturą. Dzięki jednolitemu opisowi możliwe jest też skasowanie wszystkich elementów konfiguracji powiązanych z daną usługą, bez obawy, że w infrastrukturze pozostaną nieużywane wpisy.

5. Jinja2 (http://jinja.pocoo.org/docs/2.10/)



Jinja2 to system służący do definiowania wzorców konfiguracji. Powstał on pierwotnie na potrzeby języka Python, ale ponieważ Ansible napisany jest właśnie w tym języku, to Jinja2 znakomicie się z nim integruje. Wzorzec to plik tekstowy, zawierający elementy stałe oraz znaczniki definiowane jako zmienne, wypełniane automatycznie przez Ansible w trakcie uruchamiania playbooka. Zmienne mogą pochodzić z zewnętrznego źródła (centralna baza CMDB) lub z lokalnych plików konfiguracyjnych definiowanych za pomocą notacji YAML. Po przetworzeniu wzorca powstaje docelowy plik, który możemy skopiować do zdalnego urządzenia ręcznie lub za pomocą funkcji kopiowania, wbudowanej w Ansible.

```
hostname {{inventory_hostname}}

banner motd %
**** $(hostname) ****
%

no ip domain-lookup
ip domain-name {{system.domain.name}}
snmp-server contact {{snmp.contact}}
snmp-server location {{snmp-location}}
snmp-server host {{snmp.hosts}} traps version 2c
ntp authentication-key 0 md5 {{ntp.key}}
ntp master 8
logging server {{logging.host}} use-vrf {{logging.vrf}}
logging timestamp milliseconds
logging monitor 6
logging level local5 7
no logging console
{% if loopback.ip is defined %}
interface Loopback0
 ip address {{loopback.ip}} 255.255.255.255
{% endif %}

line console
  exec-timeout 10
line vty
  exec-timeout 15
```

Do czego możemy wykorzystać wzorce? W wielu organizacjach proces automatyzacji w ogóle nie istnieje, a zmiany w sieci odbywają się często na zasadzie wklejania z notatnika. Zdarza się, że zmiany te różnią się od siebie na każdym urządzeniu, co prowadzi do braku spójności, a w efekcie do bałaganu. Ma to negatywny wpływ na zarządzanie infrastrukturą. Przejście z klasycznych procesów na pełną automatyzację jest nie lada wyzwaniem. Sieć jest elementem krytycznym i nie zawsze możemy sobie pozwolić od razu na pełną automatyzację. Ansible jest potężnym i rozbudowanym narzędziem z dużą ilością modułów sieciowych, takich jak SNMP, AAA, logowanie, interfejsy, VLANy i wiele, wiele innych. Korzystanie z nich daje dużo zalet (np. idempotentność), ale zapoznanie się z nimi i całym procesem zmian jest czasochłonne. Dzięki Jinja2 możemy zacząć korzystać z modelu NetDevOps'owego i takich narzędzi jak Ansible, wprowadzając etap przejściowy. Istniejący proces zmian w sieci możemy usprawnić, ujednolicając konfigurację za pomocą wzorców. W dalszym ciągu możemy wprowadzać zmiany wklejając je z notatnika, ale mamy przynajmniej pewność, że zmiany te są spójne. Co więcej, po wygenerowaniu takiej konfiguracji możemy poddać ją procesowi testowania, wykorzystując Vagrnata, chociażby po to, żeby zweryfikować poprawność składni.

Podsumowanie

Świat IT zmienia się bardzo dynamicznie. Biznes wymusza na działach IT coraz szybsze wprowadzanie usług na rynek. Działy systemowców i programistów radzą sobie z tym znakomicie. Niestety, infrastruktura sieciowa pozostaje w tyle. Nie mamy innego wyjścia, musimy się do tego procesu dostosować, a najłatwiej to zrobić poprzez wspólne narzędzia. Narzędzi DevOpsowych jest niezliczona ilość i ciężko będzie poznać je wszystkie. Nam, inżynierom sieciowym, wystarczy naprawdę kilka. Wymienione przeze mnie powyżej stanowią elementarną bazę, od której powinniśmy zacząć.

Jeżeli spodobał Ci się ten dokument i chciałbyś podzielić się na jego temat opinią lub chciałbyś dowiedzieć się czegoś więcej na temat automatyzacji sieci, pisz śmiało na krzysztof@nowoczesnysieciowec.pl

Zapraszam też serdecznie do odwiedzania portalu *nowoczesnysieciowiec.pl*. Będą na nim publikowane wpisy, dzięki którym wejście w świat automatyzacji sieci stanie się jeszcze łatwiejsze.

Pozdrawiam serdecznie,

Krzysztof Załęski

nowoczesnysieciowiec.pl