JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 22 (2022) 13–17
Received: 7 October 2021
Accepted:21 October 2021

Comparison of selected configuration management systems in Linux

Porównanie wybranych systemów zarządzania konfiguracją w systemie Linux

Szymon Dudziak*, Natalia Barbara Jakubczak, Maciej Pańczyk

Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract

The purpose of this article is to compare selected configuration management systems on Linux. These systems had to meet the condition of integration with Linux. The two of the most popular systems were chosen which are Puppet and Ansible. The comparison was based on a simple system configuration using these two systems in several aspects: installation, file and folder management, package management, user management, configuration of the Apache server and Firewall. Through this comparison, it was possible to determine which system is more suitable for a novice Linux system administrator.

Keywords: Ansible; Puppet; comparison; configuration management systems; Linux

Streszczenie

Celem artykułu jest przeprowadzenie porównania wybranych systemów zarządzania konfiguracją w systemie Linux. Systemy te musiały spełnić warunek integracji z systemem Linux. Zostały wybrane dwa z pośród najpopularniejszych systemów jakimi są Puppet oraz Ansible. Porównanie zostało oparte na prostej konfiguracji systemu z wykorzystaniem tych dwóch systemów w kilku aspektach: instalacji, zarządzania plikami i folderami, zarządzania pakietami, zarządzania użytkownikami, konfiguracji serwera Apache oraz zapory sieciowej. Dzięki takiemu porównaniu można było stwierdzić który system bardziej nadaje się dla początkującego administratora systemów Linux.

Słowa kluczowe: Ansible; Puppet; porównanie; zarządzanie konfiguracją; Linux

*Corresponding author

Email address: szymon.dudziak@pollub.edu.pl (S. Dudziak)

©Published under Creative Common License (CC BY-SA v4.0)

1. Wstęp

Wraz ze wzrostem ilości maszyn zarządzanych przez administratorów była potrzeba utworzenia wydajnych systemów wspomagających ich pracę. Takie oprogramowanie miało w znacznym stopniu skrócić i uprościć pracę administratorów.

Już na samym początku administracji systemami administratorzy próbowali uprościć swoją pracę przez tworzenie skryptów. W tym celu wykorzystywano znane im języki programowania. Przykładem takiego języka jest Python, który do teraz jest jednym z najbardziej popularnych języków programowania.

Zarządzanie konfiguracją jest bardzo rozbudowanym zagadnieniem, na które składa się wiele czynników. W trakcie tego procesu administrator ma za zadanie nadzorować zmiany wprowadzane do systemu, raportować jego stan jak i weryfikować jego kompletność oraz spójność. Ze względu na ciągły rozwój obszaru IT wymagane było utworzenie wydajnych systemów, które wspomagały administratora w codziennej pracy podczas konfiguracji nowych i utrzymywania starych jednostek fizycznych na tym samym poziomie technologicznym oprogramowania.

W tym momencie rynek oferuje wiele rozwiązań wspomagających zarządzanie infrastrukturą IT opartych zarówno na systemie Windows jak i różnych dystrybucjach Linux-a. Systemy te są w dalszym ciągu rozwijane i ulepszane a co za tym idzie oferują coraz to nowsze funkcje. Należałoby sprawdzić, czy systemy te wraz ze wzrostem funkcjonalności są w dalszym ciągu przyjazne dla początkującego użytkownika.

2. Cel i zakres badań

Celem badań zawartych w niniejszym artykule było sprawdzenie łatwości przeprowadzenia konfiguracji systemu Linux z wykorzystaniem Ansible oraz Puppet. Zakres badań obejmował instalację oprogramowania, zarządzania plikami i folderami, zarządzania pakietami na podstawie instalacji pakietu z interfejsem graficznym Xfce oraz zarządzania użytkownikami. Dla każdego z wymienionych oprogramowań utworzono niezbędne pliki konfiguracyjne. Na podstawie odczuć autorów stwierdzono, które oprogramowanie jest lepsze w danym aspekcie.

3. Przegląd literatury

Zapotrzebowanie na zaawansowane systemy zarządzania konfiguracją stale rośnie. Aby zapobiec dublowaniu badań przeprowadzono przegląd literatury.

Grupa badaczy jakimi są Sanjeev Thakur, Chand Gupta, Nishant Singh oraz Soloman Geddam w pracy pod tytułem „Mitigating and Patching System Vulnerabilities Using Ansible: A Comparative Study of Various Configuration Management Tools for IAAS Cloud” [1] podjęli się porównania następujących narzędzi jakimi są Ansible, Chef, Puppet oraz Salt. Wynikiem ich badań było stwierdzenie, że najlepiej rozwijającym się narzę-

dziem do zarządzania konfiguracją jest Ansible. Według autorów wynika to z faktu, że Ansible wspiera oraz oferuje szeroką gamę szybkich wdrożeń oraz napraw.

W pracy pod tytułem „ Software Configuration Management: A comparison of Chef, CFEngine and Puppet" [2] Fredrik Önnberg porównał narzędzia Chef, CFEngine oraz Puppet. Problemem jakim się zajął było sprawdzenie, czy wymienione w pracy oprogramowania wspierają wskazane w pracy systemy operacyjne. W pracy podano ile pakietów jest instalowanych podczas instalacji poszczególnych narzędzi.

„Learning Ansible 2 – Second Edition" [3] autorstwa Fabia Alessandra Locati'ego pozwala na poznanie procesu automatyzacji konfiguracji na podstawie najnowszej (dostępnej w chwili wydania) wersji narzędzia AnsibleGet. W oparciu o wcześniej wymienione oprogramowanie twórca zaznajamia czytelnika z podstawowymi elementami Ansible w skład których wchodzą np. moduły, czy podręczniki.

W trzecim wydaniu książki autorstwa Johna Arundel'a „Puppet 5 Begginer's Guide" [4] zawarto wszystkie niezbędne informacje odnoszące się do oprogramowania jakim jest Puppet. Książka ta zawiera praktyczne przykłady wykorzystania tego narzędzia, które wspomagają użytkownika w procesie konfiguracji oraz instalacji serwera. Dodatkowo dzięki pokazaniu kolejnych kroków postępowania przedstawiane są kluczowe elementy wykorzystywane w pracy administratora systemów. W książce znajdują się także informacje odnośnie instalacji pakietów, konfiguracji plików, tworzenia użytkowników i innych kluczowych elementów pozwalających na bezproblemową pracę podczas konfiguracji systemu.

W artykule pod tytułem „Ansible vs Puppet" [5] Jarek Grzabel przedstawia różnice pomiędzy Ansible a Puppet. Według autora Ansible jest bardziej podatny na przypadkowe pomyłki od Puppet. W Puppet jest ustawiony interwał czasowy, w którym można jeszcze wprowadzić zmiany w kodzie, w Ansible wszystkie zmiany są wykonywane natychmiastowo.

Aayushi Johari w swoim artykule „Chef vs Puppet vs Ansible vs Saltstack: Which Works Best For You?" [6] porównuje podane w tytule systemy pod kątem dostępności, łatwości instalacji, zarządzania konfiguracją czy kosztem utrzymania. Ansible na tle pozostałych produktów odznacza się łatwością instalacji oraz niską ceną. Zarówno wykorzystując SaltStack jak i Ansible w łatwy sposób można przeprowadzić konfigurację systemu. W artykule zaznaczono również popularność poszczególnych systemów. Najbardziej zyskującym popularność produktem jest Ansible.

Jak do tej pory żaden badacz nie odniósł się do przeanalizowania wyboru z punktu widzenia początkujących użytkowników Ansible lub Puppet jako oprogramowania do zarządzania konfiguracją.

4. Obiekty badań

W pracy użyto dwóch systemów jakimi są Ansible, który został utworzony w 2012roku przez Michal'a DeHaan'a [7], oraz Puppet, utworzony w 2005 roku przez Luke Kanies'a [8], natomiast pierwszą wersję wydano w 2011 roku.

W tabeli 1 zawiera podstawowe informacje o wybranych systemach [9].

Tabela 1: Porównanie systemu Ansible oraz Puppet

	System	Ansible	Puppet
	Strona WWW	ansible.com	puppet.com
Języki i formaty	**Implementacja**	Python	Ruby
	Konfiguracja	YAML	Własny
Demony	**Szablon**	Jinja	Embedded Ruby
	Serwer	Nie	Opcjonalnie
	Klient	Nie	Opcjonalnie

4.1. Wstępne porównanie systemów Ansible i Puppet

Ansible jest otwartoźródłowym oprogramowaniem służącym do zarządzania konfiguracją. Można go wykorzystać do konfiguracji systemów Windows, Linux, MacOS i innych systemów UNIXopodobnych. Napisany został w oparciu o język Python.

Do komunikacji z drugim systemem wykorzystuje protokół SSH, PowerShell lub inne API. Nie wymaga instalacji agenta, co powoduje wzrost bezpieczeństwa.

Konfiguracja w Ansible wykorzystuje uprzednio przygotowane pliki oparte o język YAML lub poprzez wprowadzanie poleceń AdHoc. Ansible wykorzystuje szablony Jinja.

Podstawowym elementem konfiguracji użytym w Ansible jest moduł. Istnieje możliwość napisania modułu w oparciu o język skryptowy taki jak Python, czy Bash. Dzięki wykorzystaniu modułu wynik końcowy zawsze zostanie zrównany do jednakowego stanu. Działanie takie powstaje na przykład po przywróceniu działania systemu po awarii. Po wykonaniu modułu wszelkie elementy z nim związane są czyszczone.

Puppet jest otwartoźródłowym oprogramowaniem przeznaczonym do zarządzania konfiguracją. Można go wykorzystać do konfiguracji systemów operacyjnych Windows, Linux, czy MacOS oraz innych. Oparty on został o język Ruby, który wykorzystywany jest przy pisaniu manifestów.

Komunikacja między maszynami zachodzi poprzez agenta. Z tego powodu potrzebna jest głębsza konfiguracja tego oprogramowania do poprawnego działania.

W celu konfiguracji Puppet wykorzystuje własny format plików konfiguracyjnych. Istnieje możliwość wykorzystania wcześniej przygotowanych szablonów napisanych w języku Ruby. Wraz z rozwojem Puppeta zaistniała możliwość konfiguracji poprzez wprowadzanie poleceń AdHoc.

Język programowania Puppet opiera się na paradygmacie deklaratywnym. Oznacza to, że opisuje stan końcowy konfigurowanej maszyny po wprowadzeniu

zmian. Język ten opisuje stan systemu komputerowego w kategoriach zasobów. Są one reprezentowane przez podstawowe konstrukcje sieci oraz systemu operacyjnego. Administrator łączy zasoby w manifesty, które zawierają finalny stan systemu. Pliki manifestów przechowywane są na maszynie serwerowej. Instrukcje konfiguracyjne dla agentów są wynikiem kompilacji plików manifestowych.

5. Metoda badań

Badanie łatwości wykonywania konfiguracji przeprowadzono na maszynach opartych na tym samym systemie operacyjnym. Końcowym efektem było zbudowanie poprawnie działającego środowiska operacyjnego zawierającego niezbędne podstawowe oprogramowanie.

Porównywana była łatwość konfiguracji z wykorzystaniem plików konfiguracyjnych, proces ich pisania oraz dostęp do dokumentacji. Wynikiem badań była subiektywna ocena tworzona z punktu widzenia niedoświadczonych administratorów. W celu zrealizowania badań nie wykorzystano zewnętrznej grupy badawczej.

6. Platforma testowa

Badania były przeprowadzone z wykorzystaniem maszyn wirtualnych. Poniżej zaprezentowano jednostkę fizyczną na której zainstalowano poszczególne maszyny.

- procesor: AMD Ryzen 5 3600X 3.8GHz, 6 rdzeni, 12 wątków,
- pamięć RAM: 16GB DDR4 3200MHz,
- dysk: Patriot 1TB M.2 PCIe NVMe,
- system operacyjny: Windows 10 Education,
- karta graficzna: NVIDIA GeForce GTX 1660 Ti.

Na każdej z maszyn zainstalowano tę samą wersję systemu operacyjnego CentOS 8 Stream.

7. Obszary testowe

Dla każdego z systemów wyodrębniono wspólne obszary testowe:

- Instalacja oprogramowania,
- Zarządzanie plikami i folderami,
- Zarządzania pakietami,
- Zarządzanie użytkownikami,
- Konfigurowanie serwisów.

Instalacja opierała się na przygotowaniu środowiska do dalszej pracy. W niej skonfigurowano odpowiednio każdy systemów.

Zarządzanie plikami oraz folderami opierało się o wykonanie odpowiednio przygotowanych skryptów które automatycznie tworzyły, edytowały lub usuwały zadany element.

Zarządzanie pakietami wymagało przygotowanie skryptu, który zainstaluje dany pakiet w odpowiedniej wersji. Na potrzeby badań wykorzystano najnowsze wersje wybranych pakietów.

Zarządzanie użytkownikami pozwoliło na tworzenie, edycję oraz usuwanie użytkowników. Pozwalało na wybranie ścieżki do katalogu domowego, przypisaniu użytkownikowi dowolnego hasła oraz wybranych uprawnień.

Konfigurowanie serwisów w głównej mierze opierało się na restartowaniu i uruchamianiu usług. Dodatkowo wykonano konfigurację zapory ogniowej dokonując w niej wpis umożliwiający bezproblemową pracę serwera Apache.

Listing 1: Przykładowy skrypt konfigurujący dla oprogramowania Ansible

```
[root@ansible ~]# cat user.yml
---

- hosts: all
  gather_facts: yes
  vars_files:
          - haslo.yml

  tasks:
          - name: Tworzenie nowej grupy
            group:
                    name: "test-group"
                    gid: 3500
                    state: present

          - name: Tworzenie uzytkownikow
            user:
                    name: "{{ item }}"
                    password: "{{ user_password
                        | password_hash('sha512') }}"
                    update_password: on_create
                    groups: "test-group"
                    shell: /bin/bash
            loop:
                  - test
                  - teststudent
[root@ansible ~]# cat haslo.yml
user_password: test
```

Na Listingu 1 zaprezentowano skrypt napisany dla oprogramowania Ansible. Przedstawia on konfigurację użytkowników oraz nowej grupy. W tym celu wykorzystano mechanizm pętli.

Listing 2: Przykładowy skrypt konfigurujący dla oprogramowania Puppet

```
group { 'test':
 ensure => present,
 gid => 3500,
}

file { '/home/test-user':
 ensure => directory,
 mode => '0750',
 owner => 'test-user',
}

file { '/home/student':
 ensure => directory,
 mode => '0750',
 owner => 'student',
}

user { 'test-user':
 ensure => present,
 uid => 3501,
 home => '/home/test-user',
 shell => '/bin/bash',
 groups => ['test'],
 password => pw_hash('test', 'SHA-512', 'password'),
}

user { 'student':
 ensure => present,
 uid => 3502,
 home => '/home/student',
 shell => '/bin/bash',
 groups => ['test', 'root'],
 password => pw_hash('teststudent', 'SHA-512', 'password'),
}
```

Listing 2 zawiera analogiczną konfigurację dla oprogramowania Puppet. Widoczna jest większa złożo-

ność tego skryptu względem skryptu napisanego dla Ansible.

Całość badań oparto na uprzednio przygotowanych skryptach, dzięki którym w łatwy sposób można było zarządzać konfiguracją systemu.

8. Wyniki

W tabeli 2 zawarto badane w wybranych systemach zarządzania konfiguracją systemu Linux aspekty, które oceniono w skali od 1 (trudne) do 5 (łatwe).

Tabela 2: Przedstawienie wyników łatwości konfiguracji z wykorzystaniem danego oprogramowania

	Ansible	**Puppet**
Instalacja oprogramowania	5	2
Zarządzanie plikami i folderami	4	4
Zarządzanie pakietami	4	4
Zarządzanie użytkownikami	5	3
Konfigurowanie serwisów	4	4

Wyniki te oparto o badania systemów, na których wykonano podobne czynności. Tylko wtedy można było w rzetelny sposób ocenić poszczególne oprogramowania. Mimo różnic można było poddać ocenie te elementy, które wykonano na obu systemach.

9. Analiza odczuć badaczy

Porównując oba systemy stwierdzono, że instalacja Ansible jest dużo łatwiejsza w stosunku do instalacji Puppet. Do prawidłowego działania Ansible wymagana jest instalacja samego pakietu, dopisania grup w pliku konfiguracyjnym oraz wymiany kluczy SSH. Puppet wymaga instalacji agenta, przez co czas samej instalacji jest znacznie wydłużony. Ponadto ze względu na mnogość elementów, które należy przygotować do poprawnego działania w przypadku Puppet istnieje możliwość uszkodzenia instalacji. Z tego powodu łatwość instalacji określono jako umiarkowanie trudną.

Zarządzanie plikami i folderami nie stanowiło problemu zarówno w przypadku wykorzystania Ansible jak i Puppet. Oba z tych systemów posiadają rozwiązania umożliwiające w prosty sposób obsługę plików i folderów. Dużym plusem dla Ansible była mnogość preinstalowanych modułów. W Puppet, aby wykonać bardziej zaawansowane rzeczy należy ręcznie zainstalować poszczególne rozszerzenia.

Przechodząc do zarządzania pakietami uznano, że zarówno w Ansible jak i Puppet ta funkcjonalność jest łatwa to przyswojenia. Ansible posiada pętlę, która wykona dane zadanie n-krotnie, zależnie od ilości podanych argumentów. W Puppet można utworzyć zmienną, w której przechowana będzie tablica z zawartością. Następnie wykonując dane zadanie opierające je na danej zmiennej wykona ona to zadanie n-krotnie.

Uznano, że tworzenie, edycja jak i usuwanie użytkowników było nieco łatwiejsze w Puppet niż w Ansible. Moduł związany z użytkownikami według autorów lepiej sprawdzał się w tworzeniu użytkowników właśnie w Ansible. Tworząc nowego użytkownika nie jest potrzebne uprzednie tworzenie katalogu domowego, czy ustawianie mu uprawnień. W Ansible wszystko dzieje się automatycznie, co w porównaniu do Puppet jest dużym plusem. W Puppet wszystkie powyższe kroki należy wykonać, aby można było utworzyć użytkownika.

Odnosząc się do zarządzania serwisami stwierdzono, że na bardzo podstawowym poziomie dosyć prosto można obsłużyć wszystkie serwisy. W obu przypadkach istnieją odpowiednie moduły, które wspomagają użytkownika w procesie zarządzania. Podczas badań podjęto się prostej konfiguracji zapory ogniowej w celu poprawnego działania serwera Apache. W obu przypadkach istnieje możliwość pracy z zaporą sieciową. W Ansible istnieje preinstalowany moduł, a w Puppet można taki zainstalować lub wykorzystać moduł *Exec*, który umożliwia wykonanie wprowadzonego do niego polecenia.

10. Wnioski

Oba z porównywanych narzędzi do zarządzania konfiguracją systemów Linux posiadają wiele funkcji wspomagających pracę administratora.

Zdecydowanie łatwiejszym oprogramowaniem do instalacji jest Ansible. Ansible w przeciwieństwie do Puppet-a nie wymaga agenta, a sam proces jego konfiguracji jest prostszy i klarowniejszy dla początkującego użytkownika. W Puppet istnieje wiele zawiłości, które mogą zniechęcić potencjalnego użytkownika.

Zarządzanie plikami i folderami nie stanowiło problemu zarówno przy używaniu Ansible jak i Puppet. Obydwa systemy były w tym aspekcie przyjazne użytkownikowi.

Zastosowanie klasowości znanej z języków programowania w Puppet i pętle zaimplementowane w oprogramowaniu Ansible pozwoliło na bezproblemową pracę z pakietami. Dzięki nim w łatwy sposób można instalować, aktualizować lub usuwać wiele pakietów. Dodatkowo taki zabieg wpływa na optymalizację skryptu, co oznacza, że wystarczy napisać zadanie instalacji jednokrotnie, a dane oprogramowanie powieli jego reprezentację w zależności od ilości elementów w pętli. Uznano, że w jednym i drugim systemie nie ma większych problemów z przeprowadzeniem instalacji jednego lub więcej pakietu.

W przypadku zarządzania użytkownikami według autorów to Ansible jest lepszym wyborem. W Ansible tworzenie użytkowników opierało się na jednym zadaniu. Moduł odpowiedzialny za zarządzanie użytkownikami w tym systemie jest na tyle rozbudowany, że tworzenie bądź usunięcie użytkownika nie sprawia większych trudności.

Ostatnim porównywanym aspektem było zarządzanie serwisami. Sam proces obsługi serwisu w obu systemach jest bardzo podobny. Zarówno w Puppet jak

i Ansible można uruchamiać, restartować, czy zatrzymywać serwisy. W obu przypadkach stwierdzono, że praca z demonami nie jest trudna, jednak wymaga dużego doświadczenia.

Wykorzystując poszczególne systemy można wyjść poza obręb podstawowych funkcji. Ustawienie zapory ogniowej jest jednym z bardziej zaawansowanych elementów i wymaga przemyślenia całego procesu. Oba systemy wspierały pracę z zaporą ogniową i umożliwiały jej konfigurację w dość przejrzysty sposób. W systemie Ansible jest od razu wbudowany moduł umożliwiający wykonywanie poleceń na zaporze. W Puppet istnieje opcjonalny moduł do pracy z Firewall-em.

Według badaczy zaletą Puppet jest możliwość wykonania danego zadania tylko w przypadku, gdy dany pakiet lub serwis istnieje. Oczywiście w Ansible też jest taka możliwość, jednak porównując te dwa systemy to w Puppet ta funkcjonalność jest łatwiejsza do opanowania przez niedoświadczonego użytkownika. W Ansible wymagane jest pisanie długich warunków, co nie jest w dłuższej perspektywie czasu wygodne.

Porównując strukturę pliku konfiguracyjnego Ansible oraz Puppet stwierdzono, że bardziej przyjazną formę mają pliku pisane w języku YAML. Według autorów są one bardziej przejrzyste oraz prostsze w pisaniu. W przypadku funkcjonalności lepsze okazały się pliki własne Puppet, ponieważ język ten został stworzony specjalnie dla tego narzędzia.

Podsumowując, zarówno jeden jak i drugi system w dużym stopniu wspomaga pracę administratora systemów. Dzięki zastosowanych w nich modułach upraszcza pracę i skraca czas potrzebny na wykonanie monotonnych czynności. Nie da się jednoznacznie określić, który z podanych systemów jest lepszy. Według badaczy składnia Ansible oraz proces instalacji są bardziej przyjazne użytkownikowi. Puppet bardziej nadaje się do pracy w dużych korporacjach niż w małych firmach. Oba systemy znajdą swoje grono odbiorców i wspomogą pracę wszystkich administratorów, którzy nie są zamknięci na technologiczne nowości.

Literatura

[1] S. Thakur, S. C. Gupta, N. Singh, S. Geddam, Mitigating and patching system vulnerabilities using ansible: A comparative study of various configuration management tools for iaas cloud. In Information Systems Design and Intelligent Applications, Springer New Delhi (2016) 21-29.

[2] F. Önnberg, Software Configuration Management: A comparison of Chef, CFEngine and Puppet, (2012) http://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-6040.

[3] F. Locati, Learning Ansible 2 – Second Edition, Packt Publishing Ltd., 2016.

[4] J. Arundel, Puppet 5 Beginner's Guide – Third Edition. Packt Publishing Ltd., 2017.

[5] J. Grzabel , Puppet Vs Ansible, https://www.devopsgroup.com/blog/puppet-vs-ansible/, [13.07.2021].

[6] A. Johari, Chef vs Puppet vs Ansible vs Saltstack: Which Works Best For You?, https://www.edureka.co/blog/chef-vs-puppet-vs-ansible-vs-saltstack/, [13.07.2021].

[7] Opis oraz pierwsze wydanie oprogramowania Ansible, https://en.wikipedia.org/wiki/Ansible_(software), [14.06.2021].

[8] Opis oraz pierwsze wydanie oprogramowania Puppet, https://en.wikipedia.org/wiki/Puppet_(software), [14.06.2021].

[9] E. Nemeth, G. Snyder, T. R. Hein, T. Adelstein, B. Lubanovic, T. Limoncelli, UNIX i Linux przewodnik administratora linux. Rozdział 23, Helion, 2018.