Puppet

Wstęp

Puppet jest jednym z popularnych narzędzi typu Configuration Management Tool. Swoją popularność zdobył dzięki temu, że działa na wielu platformach systemowych (Linux, Windows, BSD, OS X, Solaris i inne), posiada modułową budowę, ma szeroką funkcjonalność, ciągle poszerzaną. Choć narzędzia do zarządzania i konfiguracją wielu serwerów wydają się nie być proste w implementacji i zarządzaniu - trzeba nauczyć się specyficznego języka konfiguracji - to inwestycja w to narzędzie bardzo szybko się zwraca. Managerowie chętnie wdrażają te narzędzia gdyż pozwalają na oszczędności w zasobach ludzkich. Od teraz do zarządzania więcej niż setką serwerów wystarczy jeden administrator. Narzędzia te szczególnie sprawdzają się w środowiskach powtarzalnych. Praktycznie wszystkie zadania administracyjne można przekuć w blok konfiguracyjny a potem zaimplementować na setkach serwerów. Od teraz zmiana głównego serwera NTP na 100 maszynach może zająć 5 minut a nie 5 dni. Od teraz można przygotować konfigurację / wdrożenie aplikacji na środowisku przedprodukcyjnym, przetestować a następnie w kilka minut zaaplikować na produkcji. Dziś większość firm IT z więcej niż 10 serwerami do zarządzania używa narzędzi CMT gdyż jest to narzędzie optymalizujące pracę i wprowadzające standardy na wielu maszynach na raz.

Puppet jest obecnie (04 Feb 2017), obok Chef jednym z najbardziej popularnych narzędzi. Korzystają z niego takie firmy i organizacje jak AT&T, Nasa, Atlassian, CiscoWebEx, EMC, GitHub, Intel, McAfee, Nestle, RedHat, Sony, Uber, Spotify, Verizon, WMWare i wiele innych (w sumie ponad 30 tysięcy aktywnych instalacji). PuppetLabs zatrudnia ponad 500 pracowników, zainwestował w swoje oprogramowanie pond 86 milionów dolarów.

Puppet jest rozwijany w dwóch głównych gałęziach:

1. darmowej open source - Puppet;
2. płatnej, komercyjnej ze wsparciem - Puppet Enterprise.

Komponenty i instalacja

Puppet jest narzędziem typu server-agent. Aby więc zarządzań jakimkolwiek systemem operacyjnym należy zainstalowań na nim agenta:

Instalacja klienta - puppet agent

```
apt install puppet
```

który będzie łączył się co 30 minut do tzw. puppetmastera:

Instalacja serwera - puppetmaster

```
apt install puppetmaster
```

Agent pobiera co 30 minut konfigurację z puppetmastera, weryfikuje, czy stan sytemu jest zgodny z pobraną konfiguracją a jeśli nie wprowadza zmiany w życie.

Konfiguracja podstawowa

Zainstalowane oprogramowanie puppet na serwerze i agentach należy odpowiednio skonfigurować.

Konfiguracja puppetmastera

Domyślna konfiguracja strony serwerowej w Debian jest właściwie gotowa do użycia po instalacji z paczki. Można upewnić się czy serwer działa i skonfigurować go tak, aby uruchamiał się przy starcie:

```
root@master:/etc/puppet# systemctl enable puppetmaster
Synchronizing state for puppetmaster.service with sysvinit using update-rc.d...
Executing /usr/sbin/update-rc.d puppetmaster defaults
Executing /usr/sbin/update-rc.d puppetmaster enable
root@master:/etc/puppet# systemctl start puppetmaster
root@master:/etc/puppet# netstat -ltnp | grep 8140
tcp        0      0 0.0.0.0:8140            0.0.0.0:*               LISTEN      1976/ruby
```

Jeśli po drodze między agentami a serwerem są firewalle należy otworzyć ruch od agentów do serwera na port 8140 TCP.

Jeśli zamierzamy publikować za pomocą puppeta wygodnie pliki (nie zastanawiaj się, na pewno będziesz to robić 😉) warto zweryfikować sekcję `[files]` pliku `/etc/puppet/fileserver.conf`:

/etc/puppet/fileserver.conf - przykładowa konfiguracja, zezwalami na dostęp tylko hostom z domeny lab.itmz.pl

```
# Define a section 'files'
# Adapt the allow/deny settings to your needs. Order
# for allow/deny does not matter, allow always takes precedence
# over deny
[files]
  path /etc/puppet/files
  allow *.lab.itmz.pl
#  allow *.example.com
#  deny *.evil.example.com
#  allow 192.168.0.0/24
```

Konfiguracja agenta

Wskazanie serwera

Po instalacji agenta z paczki w systemie Debian należy wskazać w pliku konfiguracyjnym `/etc/puppet.conf` nazwę serwera puppetmaster:

/etc/puppet.conf

```
[agent]
server=master.lab.itmz.pl
```

Pierwsze połączenie z puppetmaster

Podstawową komendą za pomocą możemy ręcznie wywołać na agencie puppet (kliencie) pobranie konfiguracji z puppet-master (serwerze puppet) jest:

```
puppet agent --no-daemonize --onetime --verbose
```

Ponieważ jest to długa komenda często przzogotowuje się alias bądź skrypt który wywołuję tą komendę z parametrmi, np.:

```
root@agent01:~# cat /usr/local/sbin/puppet.run.once.sh
#!/bin/sh
#puppetfile
puppet agent --no-daemonize --onetime --verbose


root@agent01:~# ls -l /usr/local/sbin/puppet.run.once.sh
-rwxr-x--- 1 root root 71 Feb 22 17:36 /usr/local/sbin/puppet.run.once.sh
```

Pierwszym krokiem jest próba pobrania konfiguracji z puppet-master co spowoduje wygenerowanie certyfikatu i podłączenie się puppetmastera:

```
oot@agent01:~# puppet agent --no-daemonize --onetime --verbose
Info: Caching certificate for ca
Info: csr_attributes file loading from /etc/puppet/csr_attributes.yaml
Info: Creating a new SSL certificate request for agent01.lab.itmz.pl
Info: Certificate Request fingerprint (SHA256): 7D:8B:BF:CF:CA:C8:12:A4:00:03:AE:AA:11:EB:37:BE:0B:7B:54:43:1D:
61:E4:D5:C8:6B:D1:FC:DA:22:8C:6B
Info: Caching certificate for ca
Exiting; no certificate found and waitforcert is disabled
```

Puppet master ma już informację o agencie, natomiast po jego stronie należy autoryzować agenta aby mógł on dalej pobierać konfigurację.

Autoryzowanie agenta po stronie puppetmastera

```
root@master:~# puppet cert list #lista agentów oczekujcych na podpis certyfikatu
  "agent01.lab.itmz.pl" (SHA256) 7D:8B:BF:CF:CA:C8:12:A4:00:03:AE:AA:11:EB:37:BE:0B:7B:54:43:1D:61:E4:D5:C8:6B:
D1:FC:DA:22:8C:6B


root@master:~# puppet cert sign "agent01.lab.itmz.pl" #uwierzytelnienie agenta01
Notice: Signed certificate request for agent01.lab.itmz.pl
Notice: Removing file Puppet::SSL::CertificateRequest agent01.lab.itmz.pl at '/var/lib/puppet/ssl/ca/requests
/agent01.lab.itmz.pl.pem'
```

Autoryzowanie agenta po stronie puppetmastera może być realizowane automatyczne dla wybranych hostów z domeny. Listę hostów, które są autoryzowane automatycznie można podać w pliku `/etc/puppet/autosign.conf` Można w nim również używać symboli wieloznacznych:

```
root@master:/etc/puppet# cat autosign.conf
*.lab.itmz.pl
```

Powtórne połączenie z puppetmaster

```
root@agent01:~# puppet agent --no-daemonize --onetime --verbose
Info: Caching certificate for agent01.lab.itmz.pl
Info: Caching certificate_revocation_list for ca
Info: Caching certificate for agent01.lab.itmz.pl
Notice: Skipping run of Puppet configuration client; administratively disabled (Reason: 'Disabled by default on
new or unconfigured old installations');
Use 'puppet agent --enable' to re-enable.

root@agent01:~# puppet agent --enable
root@agent01:~# puppet agent --no-daemonize --onetime --verbose
Info: Retrieving pluginfacts
Info: Retrieving plugin
Info: Caching catalog for agent01.lab.itmz.pl
Info: Applying configuration version '1486166362'
Notice: /Stage[main]/Main/Node[agent01]/Package[htop]/ensure: ensure changed '1.0.3-1' to 'purged'
Notice: /Stage[main]/Puppet_run_once/File[/usr/local/sbin/puppet.run.noop.sh]/ensure: defined content as '{md5}
bba40d22209bea7f3362429a080260fe'
Notice: Finished catalog run in 1.87 seconds
```

Włączenie na stałe usługi puppet

Dla pewności uruchamiamy usługę puppet oraz włączamy, aby uruchamiana była na starcie:

```
root@agent01:~# systemctl enable puppet
Synchronizing state for puppet.service with sysvinit using update-rc.d...

root@agent01:~# systemctl restart puppet

root@agent01:~# systemctl status puppet
 puppet.service - Puppet agent
   Loaded: loaded (/lib/systemd/system/puppet.service; enabled)
   Active: active (running) since Sat 2017-02-04 00:12:32 GMT; 3s ago
  Process: 10118 ExecStart=/usr/bin/puppet agent $DAEMON_OPTS (code=exited, status=0/SUCCESS)
 Main PID: 10127 (puppet)
   CGroup: /system.slice/puppet.service
           10127 /usr/bin/ruby /usr/bin/puppet agent

Feb 04 00:12:32 agent01 puppet-agent[10127]: Reopening log files
Feb 04 00:12:32 agent01 puppet-agent[10127]: Starting Puppet client version 3.7.2
Feb 04 00:12:32 agent01 systemd[1]: Started Puppet agent.
Feb 04 00:12:33 agent01 puppet-agent[10131]: Finished catalog run in 0.09 seconds
```

Bez powyższego agent nie będzie łączył się co 30 minut do puppetmastera aby pobrać aktualną konfigurację.

Konfiguracja

Jest wiele poradników, które wprowadzają w konfigurację puppeta krok po kroku. Te, które można polecić na początek to:

- Puppet Cookbook - wiele przykładów użycia
- Puppet Learning VM - do pobrania z oficjalnej strony puppet

Pierwszy wpis konfiguracyjny

W plikach manifestów (rozszerzenie `.pp`) zamieszczane są zapisy konfiguracyjne dla agentów podłączonych do węzłów. Główny plik manifestu to `/etc/puppet/manifests/site.pp` Od tego pliku puppet zaczyna czytać konfigurację.

/etc/puppet/manifests/site.pp

```
node 'agent01' {
  file { '/tmp/puppetized-dir':
    ensure => 'directory',
    owner  => 'root',
    group  => 'puppet',
    mode   => '0755',
  }
}
```

Dzięki powyższemu wpisowy w pliku `site.pp`, `agent01` po pobraniu konfiguracji z puppetmastera doprowadzi do takiego stanu w systemie, że w katalogu /tmp/ znajdował się będzie katalog `puppetized-dir`, właścicielem będzie root, grupą puppet a uprawnienia tego pliku będą ustawione na 755. Dla noda 'agent01' możemy zdefiniować więcej zasobów i ich atrybutów. W tym samym pliku - `site.pp` można zdefiniować więcej nodów a każdy z nich może mieć zdefiniowanych wiele zasobów.

Zasoby - resources

Konfiguracja hosta z użyciem puppeta jest oparta o definiowanie zasobów jakie maja znaleźć się na hoście wraz z ich atrybutami (cechami). Przykładowo, najpopularniejszy zasób typu plik, może być zdefiniowany następująco:

```
file { '/etc/shadow':
  ensure  => 'present', #definiujemy, e plik ma istnie (zmie na 'absent', jeli ma by usunity)
  owner   => 'root',         #waciciel
  group   => 'shadow',        #grupa waciciel
  mode    => '640',               #uprawnienia
  type    => 'file',        #typ (plik czy katalog)
}
```

Są inne zasoby w Puppet, nie tylko pliki i katalogi, choć to one głównie budują system. Zasobem są także punkt montowania, konto użytkownika, pakiet do zainstalowania, klucz ssh, wpis w cron, uruchomienie skryptu i wiele innych. Listę wszystkich wbudowanych zasobów oraz ich atrybuty tut aj (dla Puppet v.3.7).

Atrybuty zasobu

Oprócz dokumentacji znajdującej się na oficjalnej tronie Puppet dostarcza narzędzie, dzięki któremu można podpatrzeć najpopularniejsze atrybuty zasobu istniejącego już w systemie:

```
root@master:/etc/puppet/lab-extra-files# puppet resource file /tmp
file { '/tmp':
  ensure => 'directory',
  ctime  => '2017-02-05 22:17:01 +0000',
  group  => '0',
  mode   => '1777',
  mtime  => '2017-02-05 22:17:01 +0000',
  owner  => '0',
  type   => 'directory',
}

root@master:/etc/puppet/lab-extra-files# puppet resource mount /
mount { '/':
  ensure  => 'mounted',
  device  => 'UUID=84bf4e09-fb8a-45b2-8158-c3eb6815fc3a',
  dump    => '0',
  fstype  => 'ext4',
  options => 'errors=remount-ro',
  pass    => '1',
  target  => '/etc/fstab',
}

root@master:/etc/puppet/lab-extra-files# puppet resource user vagrant
user { 'vagrant':
  ensure           => 'present',
  comment          => 'Vagrant Default User,,,',
  gid              => '1000',
  groups           => ['cdrom', 'floppy', 'sudo', 'audio', 'dip', 'video', 'plugdev', 'netdev', 'bluetooth'],
  home             => '/home/vagrant',
  password         => '$6$R339fuiL$yOKyh5dLq36WkZfkGLs2oW9LWTAmULszDkiI8lLMgTepU
/hmO5UKNIr9gRhf8fXVNXkHlBfmRoqvZ9ucE8WA./',
  password_max_age => '99999',
  password_min_age => '0',
  shell            => '/bin/bash',
  uid              => '1000',
}
```

Klasy

Tak jak funkcje czy procedury w językach programowania tak klasy w Pupecie pozwalają nam na rozbicie konfiguracji na mniejsze kawałki i wykorzystanie tych samych definicji zasobów w wielu miejscach. Zaaplikowanie tej samej konfiguracji do wielu lub setek serwerów - to codzienność w narzędziach CMT:

```
class net_tools {
  package { 'wget': ensure => 'installed' }
  package { 'nmap': ensure => 'installed' }
  package { 'dnsutils': ensure => 'installed' }
  package { 'tcpdump': ensure => 'installed' }
  package { 'telnetd': ensure => 'purged' } #pakiet powinien by odinstalowany!
}


node db03 {
  include net_tools
}


node frontsrv02 {
  include net_tools
}
```

Klasy mogą być również zawierać parametry deklaracji. Więcej informacji znajdziesz tutaj.

Choć nie spowoduje to zatrzymania puppetmaster dobrą praktyką jest utrzymywanie w porządku definicji klas i zamieszczać każdą z nich w osobnym pliku .pp o takiej samej nazwie jak nazwa klasy. Zbiór klas grupujemy natomiast w moduły.

Moduły

Zestaw klas budowanych wokół określonego tematu (np. klasy obsługujące konfigurację jednej usługi) możemy grupować w moduły. Możemy budować własne moduły - zamieszczamy je wtedy osobno w podkatalogach katalogu `/etc/puppet/modules/` - możemy też korzystać z tych napisanych przez innych i udostępnionych społeczności. Na przykład moduły dostarczające do wygodnej podmiany parametrów w plikach konfiguracyjnych to stdlib oraz inifile. W środowisku Puppeta wystarczy zainstalować moduł na puppetmaster, agenci sami pobiorą ten moduł do siebie.

instalacja dodatkowych modułów

```
root@master:~# puppet module install puppetlabs-stdlib
Notice: Preparing to install into /etc/puppet/modules ...
Notice: Downloading from https://forgeapi.puppetlabs.com ...
Notice: Installing -- do not interrupt ...
/etc/puppet/modules
 puppetlabs-stdlib (v4.15.0)


root@master:~# puppet module install puppetlabs-inifile
Notice: Preparing to install into /etc/puppet/modules ...
Notice: Downloading from https://forgeapi.puppetlabs.com ...
Notice: Installing -- do not interrupt ...
/etc/puppet/modules
 puppetlabs-inifile (v1.6.0)
```

Przykład drzewa plików własnego modułu:

```
root@master:/etc/puppet/modules# tree xm_puppetdemo/
xm_puppetdemo/
 files
    usr
        local
            bin
                report_date.sh
 manifests
    backup.tgz
    change_line_config_disable_root_sshd.pp
    cron_report_date.pp
    del_file.pp
    execute_script_once.pp
    init.pp
    inline_change_mysqld_config.pp
    local_hosts.pp
    local_users.pp
    restart_mysql_once.pp
    service_install_and_run_mysqld.pp
    service_ntpd.pp
    soft_dns_utils.pp
    soft_install_basic_tools.pp
    soft_netcat.pp
    soft_other.pp
 templates
     ntpd
         prod
            ntp.conf.erb
         test
             ntp.conf.erb
```

Głównym plikiem modułu jest `/etc/puppet/modules/xm_puppetdemo/manifests/init.pp` który zawiera definicję zasobów, które będą zaaplikowane, jeśli kasa zostanie wywołana poprzez:

```
include xm_puppetdemo
```

W każdym innym pliku manifestu można używać klas danego modułu poprzez wywołanie (przykład):

```
include xm_puppetdemo::cron_report_date
```

A jeśli klasy posidają argumenty (przykład):

```
class {'xm_puppetdemo::service_ntpd': environment => 'prod', ntpserver0 => '1.pl.pool.ntp.org' }
```

Powyżej klasa xm_puppetdemo::service_ntpd została zadeklarowana używając dwóch argumentów (environment i ntpserver0) z określonymi wartościami.

Inne

Zablokowanie dostępu agentowi do puppetmastera

Możliwe, że w infrastrukturze usunęliśmy jakąś maszynę bądź chcemy zablokować jej dostęp do puppetmastera:

```
#dla wywietlenia wszystkich agentów uwierzytlnionych na masterze:
root@master:/etc/puppet# puppet cert list --all
+ "agent01.lab.itmz.pl" (SHA256) C2:F9:6E:AF:E8:E8:9E:CD:22:6A:6A:6D:3F:43:0B:94:3D:49:60:8D:E7:73:FF:22:8C:0F:
6B:8C:F3:A4:2F:F3
+ "agent02.lab.itmz.pl" (SHA256) 2D:4F:66:AC:2B:2F:DA:32:EF:25:03:89:F3:EE:32:AD:5C:B4:A0:56:B1:74:84:A6:0C:66:
2F:D1:68:6D:7B:54
+ "agent03.lab.itmz.pl" (SHA256) BB:D6:7D:DC:B2:97:E9:35:8F:0B:8A:5A:2E:95:95:C0:D6:D3:E1:31:18:33:70:54:33:21:
39:E5:1D:9D:F7:06
+ "agent04.lab.itmz.pl" (SHA256) B5:CA:0E:DC:E8:AD:D1:44:BA:BB:25:BD:5E:E2:18:33:54:F3:24:69:42:A4:FB:FB:CB:68:
30:FF:1D:D4:74:F9
+ "master.lab.itmz.pl"  (SHA256) BB:3D:36:4D:83:1A:57:A2:7E:15:48:EA:BD:9A:E3:49:16:D3:F0:F6:98:1C:B9:E2:A7:55:
F6:57:05:EC:A6:0A (alt names: "DNS:master.lab.itmz.pl", "DNS:puppet", "DNS:puppet.lab.itmz.pl")


#wyczyszczenie informacji o agencie
root@master:/etc/puppet# puppet node clean agent01.lab.itmz.pl
Notice: Revoked certificate with serial 5
Notice: Removing file Puppet::SSL::Certificate agent01.lab.itmz.pl at '/var/lib/puppet/ssl/ca/signed/agent01.
lab.itmz.pl.pem'
agent01.lab.itmz.pl

#wycofanie certyfikatu:
root@master:/etc/puppet# puppet cert clean agent01.lab.itmz.pl
Notice: Revoked certificate with serial 5
```

Przywracanie agentowi dostępu do puppetmastera

Jeśli agent01 miałby być ponownie podłączony do puppetmastera należy usunąć na agencie katalog z certyfikatami:

```
root@agent01:~# rm -rf /var/lib/puppet/ssl/
```

i ponownie przejść przez proces konfiguracji agenta.

Zewnętrzne źródła informacji o Puppet

- Oficjalna dokumentacja dla wersji Puppet 3.7 - https://docs.puppet.com/puppet/3.7/index.html
- Wiele przykładów praktycznego użycia Puppeta - https://www.puppetcookbook.com/
- Wirtualna maszyna i podręcznik z podstaw puppeta - Puppet Learning VM (wymaga rejestracji na stronach Puppeta)
- Zalecenia odnośnie stylu języka manifestów - https://docs.puppet.com/puppet/latest/style_guide.html
- Wygodne narzędzie do poprawiania wizulanie składni pliku manifestów - http://puppet-lint.com/
- Porównanie Puppeta z innymi popularnymi narzędziami CMT Puppet vs Chef vs Ansible vs Salt