

Apache Cassandra 2.0

Przewodnik instalacji i konfiguracji w systemie Debian Wheezy

Jan Baranowski, Michał Kaik
Politechnika Poznańska

17 czerwca 2014

1 Wstęp

Niniejszy dokument ma za zadanie przedstawić proces instalacji i konfiguracji serwera baz danych NoSQL *Apache Cassandra* w środowisku rozproszonym. Na potrzeby demonstracji zakłada się że środowisko to będzie składać się z kilku węzłów połączonych siecią lokalną.

Apache Cassandra jest serwerem baz danych NoSQL, początkowo rozwijanym przez *Facebooka* na potrzeby umożliwienia efektywnego wyszukiwania wiadomości w skrzynce odbiorczej. Obecnie (od marca 2009 roku) *Cassandra* jest rozwijana przez *Apache Foundation* (jako jeden z projektów top-level) i stanowi podstawę dla zestawu narzędzi *DataStax*.

Cassandra powstała jako narzędzie mające w założeniu cechować się:

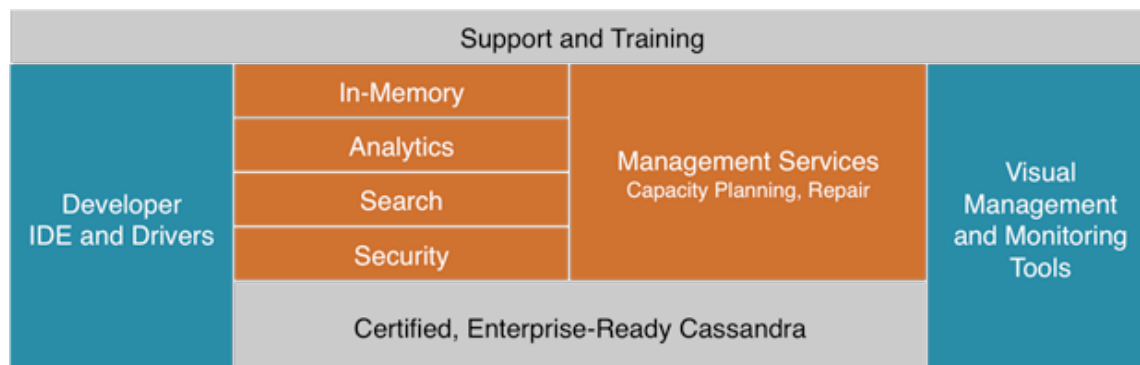
- wysoką dostępnością (*Cassandra* jest określana jako zawsze zapisywalna baza danych)
- niskim opóźnieniem wykonywanych operacji (*ang.* latency)
- odpornością na awarie (możliwością replikacji danych, brakiem komponentów, których awaria może zdestabilizować system (*ang.* single points of failure))
- możliwością regulacji kompromisu pomiędzy szybkością działania a odpornością na awarie i spójnością replik
- relatywnie prostym modelem danych

Zarówno opisanie kolumnowego modelu danych wykorzystywanego w *Cassandrze*, jak i mechanizmów, dzięki którym spełnia ona założenia projektowe nie jest celem tego dokumentu. Autorzy mogą jedynie podać propozycje publikacji, które opisują wspomniane zagadnienia. Zarówno dla modelu danych, jak i budowy wewnętrznej *Cassandry* będzie to przede wszystkim dokumentacja udostępniana przez firmę *DataStax* [3]. Kolumnowy model danych, przedstawiony z perspektywy osób pracujących z bazami relacyjnymi został doskonale (choć może zbyt obszernie) opisany w *Cassandra — The Definitive Guide* [2]. Z kolei architektura *Cassandry* w dużym skrócie (choć zapewne w sposób wystarczający by służyć jako wstęp do tego dokumentu) została przedstawiona w prezentacji [1].

Procedurę przygotowania klastra przedstawiono na przykładzie systemu operacyjnego Debian GNU/Linux 6, ponieważ uchodzi on za jedno z lepszych rozwiązań dla serwerów.

2 Wybór dystrybucji

Na samym początku nowi użytkownicy *Cassandra* stają przed wyborem dystrybucji tego oprogramowania. Możliwości są dwie: instalacja standardowej wersji dostarczonej przez *Apache Foundation* oraz instalacja platformy *DataStax* — zestawu narzędzi obudowujących *Cassandra* i dostarczających m.in. funkcje pozwalające na uproszczenie zarządzania klastrem, wizualne monitorowanie, analizę obciążeń węzłów, itp. (por. rysunek 1).



Rysunek 1: architektura *DataStax* w odniesieniu do *Cassandra*, źródło: [8]

Wybór jest ważny o tyle, że pomimo wspólnego fundamentu, jakim jest *Cassandra*, procesy instalacji obu dystrybucji nie mają ze sobą nic wspólnego, tj. (według wiedzy autorów) nie da się doinstalować do dystrybucji *Apache* platformy *DataStax*.

W tym miejscu należy wspomnieć o twórcach platformy — firmie o zaskakującej nazwie *DataStax*, zajmującej się dostosowaniem *Cassandra* do potrzeb przedsiębiorstw poprzez m.in. rozwój i testowanie bazy danych, dostarczanie narzędzi ułatwiających administrację systemem, organizację szkoleń, certyfikację personelu technicznego, itd. Bodaj najbardziej znaczącym wkładem *DataStax* w rozwój projektu jest opracowanie szeregu driverów/konektorów dla różnych języków programowania, dzięki którym programiści mogą korzystać z *Cassandra* w sposób analogiczny do rozwiązań relacyjnych (np. tak jak w przypadku *MySQL Connectors*, por. [6]) i rozbudowa dokumentacji technicznej platformy, włączając w to dokumentację plików konfiguracyjnych, architektury *Cassandra* i języka *CQL* (odpowiednika *SQL* dla kolumnowych baz danych).

Na potrzeby niniejszego dokumentu założono, że właściwym wyborem będzie dystrybucja *Apache Cassandra*, ze względu na to, że dokument ma pełnić rolę wprowadzenia do technologii, nie zaś pozwalać na błyskawiczne wdrożenie systemu (*ang.* rapid deployment). Poza tym, przewodniki instalacji i konfiguracji *DataStaxa* dostępne są np. na stronie [7].

3 Instalacja

Cassandra zostanie zainstalowana przy użyciu narzędzia *APT* z repozytorium *Apache Software Foundation*. Wymagane będzie dodanie odpowiedniego repozytorium do listy źródeł *APT*a. Dodatkowo ze względu na to, że *Cassandra* napisana jest w Javie, pokazany zostanie proces instalacji *Oracle JRE* w systemie *Debian* (ta maszyna wirtualna jest zalecana przez twórców *Cassandra*).

3.1 Wybór maszyny wirtualnej Java

Apache Software Foundation dostarcza pakiety zawierające *Cassandra* w formacie `*.deb`, a także repozytorium dla *APT*a. Jedną z zależności pakietu `cassandra` jest pakiet `openjdk-7-jre`. Jest to w pewnym sensie sprzeczne z zaleceniami twórców bazy danych, ponieważ ci rekomendują maszynę wirtualną *Oracle* jako środowisko uruchomieniowe (nawet w testowanej wersji bazy 2.0 odpowiedni

komunikat wyświetlany jest w logu). Nie zmienia to faktu, że testy (funkcjonalne, lecz nie wydajnościowe) przeprowadzone na potrzeby tego artykułu udowodniły, że *Cassandra* działa bez zarzutu także w środowisku *OpenJDK*.

Należy zatem wybrać pomiędzy stosowaniem się do zaleceń a prostotą instalacji.

3.2 Instalacja i konfiguracja Oracle Java

Pakiet z *OpenJDK* jest instalowany jako pakiet zależny podczas instalacji *Cassandry*. Jeżeli jednak administrator zdecyduje się użyć *Oracle JVM*, poniżej przedstawiona jest skrótowa procedura instalacji tego oprogramowania w systemie *Debian*. Stanowi ona kompilację najprostszych rozwiązań i tym różni się od większości przewodników dostępnych w internecie, że poza konsolą systemu nie wymaga żadnych dodatkowych narzędzi (np. mechanizmu transferu plików z maszyny-terminala do maszyny-serwera).

Pierwszym krokiem jest pobranie pakietu oprogramowania (JRE, nie JDK) ze strony *Oracle*. Standardowo by pobrać plik należy zaakceptować umowę licencyjną, jednak przy odpowiedniej konfiguracji możliwe jest ominięcie tego kroku [4]:

Listing 1: pobieranie *Oracle JRE*

```
$ wget --no-cookies \  
> --no-check-certificate \  
> --header "Cookie: oraclelicense=accept-securebackup-cookie" \  
> "http://download.oracle.com/otn-pub/java/jdk/7u60-b19/jre-7u60-linux-i586.tar.gz" \  
> -O /tmp/jre-7u60-linux-i586.tar.gz
```

Popularnym sposobem instalacji pobranego oprogramowania jest wypakowanie archiwum (zazwyczaj do katalogu */opt*) i ręczna konfiguracja ścieżki systemowej (por. [4]). *Debian* dostarcza jednak narzędzie pozwalające przekonwertować archiwum do pakietu *DEB* [5].

Należy je zainstalować, a potem użyć:

Listing 2: budowa pakietu *DEB* z *Oracle JRE*

```
$ apt-get install java-package  
  
$ fakeroot make-jpkg /tmp/jre-7u60-linux-i586.tar.gz
```

Powstały w ten sposób pakiet *DEB* należy zainstalować. Po instalacji należy ustawić *Oracle JVM* jako domyślną maszynę wirtualną w systemie.

Listing 3: instalacja i konfiguracja *Oracle JRE*

```
$ dpkg -i /tmp/oracle-j2re1.7_1.7.0+update60_i386.deb  
  
$ update-alternatives --config java  
There are 2 choices for the alternative java (providing /usr/bin/java).  
  
  Selection Path Priority Status  
-----  
  0 /usr/lib/jvm/java-7-openjdk-i386/jre/bin/java 1051 auto mode  
* 1 /usr/lib/jvm/j2re1.7-oracle/bin/java 316 manual mode  
  2 /usr/lib/jvm/java-7-openjdk-i386/jre/bin/java 1051 manual mode  
  
Press enter to keep the current choice[*], or type selection number:  
  
$ java -version  
java version "1.7.0_60"  
Java(TM) SE Runtime Environment (build 1.7.0_60-b19)  
Java HotSpot(TM) Client VM (build 24.60-b09, mixed mode)
```

Według wiedzy autorów Cassandra nie wymaga ustawiania zmiennej `JAVA_HOME` dla żadnego z użytkowników (ani roota, ani użytkownika *cassandra*, właściciela demona). Jeżeli jednak zajdzie taka potrzeba, zainstalowane maszyny wirtualne można znaleźć w katalogu `/usr/lib/jvm`.

Listing 4: ustawianie `JAVA_HOME`

```
$ echo "export JAVA_HOME=/usr/lib/jvm/j2re1.7-oracle/" >> /home/<user>/.bashrc
```

3.3 Konfiguracja repozytorium

By móc zainstalować *Cassandrę*, należy dodać odpowiednie repozytorium *Apache Software Foundation* do źródeł programu *APT*. Zgodnie z konwencją każda większa wersja *Cassandry* znajduje się w osobnym repozytorium. Na potrzeby tego dokumentu w systemie testowym zostanie zainstalowana *Cassandra 2.0.8*.

Do pliku `/etc/apt/sources.list` należy dopisać:

Listing 5: nowe źródła pakietów dla *APT*a

```
deb http://www.apache.org/dist/cassandra/debian 20x main
deb-src http://www.apache.org/dist/cassandra/debian 20x main
```

Próba pobrania spisu zawartości repozytorium zakończy się niepowodzeniem, ponieważ *APT* nie zna kluczy publicznych dla repozytorium *Apache Software Foundation*. Te należy dodać w następujący sposób:

Listing 6: pobieranie kluczy publicznych repozytorium *ASF*

```
$ gpg --keyserver pgp.mit.edu --recv-keys F758CE318D77295D
$ gpg --export --armor F758CE318D77295D | sudo apt-key add -

$ gpg --keyserver pgp.mit.edu --recv-keys 2B5C1B00
$ gpg --export --armor 2B5C1B00 | sudo apt-key add -
```

Po zakończeniu wszystkich operacji należy pobrać spis zawartości repozytoriów:

Listing 7: odświeżanie list pakietów

```
$ apt-get update
```

3.4 Instalacja *Cassandry*

Ostatnim krokiem procesu instalacji jest zainstalowanie *Cassandry* z użyciem *APT*a:

Listing 8: instalacja *Cassandry*

```
$ apt-get install cassandra
```

W efekcie *Cassandra* powinna zostać zainstalowana i uruchomiona.

4 Konfiguracja węzła

Ta część dokumentu opisuje czynności związane z konfiguracją węzła *Cassandry* tak, by był on zdolny dołączyć do klastra. *Cassandra* jest systemem P2P, więc o utworzeniu klastra węzły decydują wspólnie w oparciu o taką samą nazwę klastra zdefiniowaną w pliku konfiguracyjnym i włączony kanał komunikacji poprzez plotkowanie. Kanał ten jest domyślnie wyłączony, zatem każdy węzeł *Cassandry* w sieci będzie z początku tworzył osobny klaster.

4.1 Stan systemu po instalacji

Zaraz po instalacji w systemie pojawia się nowa usługa: **cassandra**. Jest ona domyślnie uruchomiona. Jej konfiguracja zapobiega możliwości dołączenia węzła do jakiegokolwiek klastra ze względu na zablokowanie mechanizmu plotkowania (dokładnie: agent implementujący algorytm plotkujący nasłuchuje wyłącznie na adresie IP 127.0.0.1).

Pliki konfiguracyjne można znaleźć w katalogu `/etc/cassandra`. Są wśród nich:

- `/etc/cassandra/cassandra-env.sh` — skrypt definiujący zmienne środowiskowe w formie argumentów dla maszyny wirtualnej Java (np. wielkość stosu, ale także m.in. włączenie/wyłączenie mechanizmu *JMX*).
- `/etc/cassandra/cassandra-rackdc.properties` — informacja o tym, w którym fizycznie racku i centrum danych znajduje się obecny węzeł.
- `/etc/cassandra/cassandra-topology.properties` — przybliżone informacje o tym, w których rackach i centrach danych znajdują się inne węzły klastra. Plik ten jest wykorzystywany przez jedną z kilku wersji algorytmu rozmieszczania replik (*snitcha*).
- `/etc/cassandra/cassandra-topology.yaml` — plik analogiczny do poprzedniego, wykorzystywany jednak przez inną wersję algorytmu.
- `/etc/cassandra/cassandra.yaml` — główny plik konfiguracyjny *Cassandra*.

Pliki danych (w tym: commit logi) znaleźć można w katalogu `/var/lib/cassandra`. Czyszcząc zawartość trzech znajdujących się w nim podkatalogów można zresetować stan klastra. W celu wykonania tej czynności autorzy nie zalecają jednak ich całkowitego usuwania (podczas testowania mogą one zostać utworzone na nowo z prawem zapisu tylko dla roota, co uniemożliwi normalny start usługi).

Plik logu znaleźć można w `/var/log/cassandra/system.log`.

4.2 Jak debugować konfigurację?

Przed rozpoczęciem wprowadzania zmian do plików konfiguracyjnych zaleca się wyłączenie automatycznego uruchamiania usługi. Błędnie skonfigurowany węzeł może dołączyć do nieodpowiedniego klastra, a jego usunięcie w takim przypadku nie jest zadaniem trywialnym. Efekt można osiągnąć wykonując polecenie:

Listing 9: wyłączenie uruchamiania *Cassandra* na jej domyślnych runlevelach

```
$ insserv -r cassandra
```

Najprostszym sposobem sprawdzenia poprawności konfiguracji jest uruchomienie węzła w trybie jawnym (w przeciwieństwie do demona). Służy do tego podane poniżej polecenie, które wypisze log działania węzła na standardowe wyjście.

Listing 10: testowe uruchamianie *Cassandra*

```
$ cassandra -f # -f for foreground
```

Dodatkowo można zarchiwizować taki log poleceniem `tee plik.log`, które skopiuje standardowe wejście na standardowe wyjście i do podanego pliku:

Listing 11: testowe uruchamianie *Cassandra* z archiwizacją logu

```
$ cassandra -f | tee /tmp/cassandra_testrun.log
```

UWAGA! Jeżeli *Cassandra* zostanie uruchomiona w trybie jawnym podczas gdy równolegle usługa działa w tle, zamiast informacji o błędzie w logu tej pierwszej pojawi się informacja o wyjątku `NullPointerException`.

UWAGA! Jeżeli pierwsze uruchomienie *Cassandry* odbędzie się w trybie jawnym z poziomu użytkownika innego niż `cassandra`, usługa nie będzie miała prawa zapisu do nowo utworzonych katalogów danych, nie będzie więc w stanie się uruchomić.

UWAGA! Jeżeli zmienna `JAVA_HOME` dla użytkownika uruchamiającego polecenie `cassandra -f` jest ustawiona, ale prowadzi do niepoprawnie zainstalowanej maszyny wirtualnej, *Cassandra* nie uruchomi się, ale ani skrypt startowy ani log nie poinformują o błędzie.

4.3 Pliki konfiguracyjne

Role podstawowych plików konfiguracyjnych zostały przedstawione w punkcie 4.1. Tutaj (w tabeli 1) zostaną opisane parametry konfiguracyjne (z `/etc/cassandra/cassandra.yaml`), których wartości powinny zostać świadomie dobrane przed uruchomieniem pojedynczego węzła.

Konfigurację węzła ułatwia sam plik konfiguracyjny *Cassandry*, zawierający obszernie komentarze.

Tablica 1: podstawowe parametry konfiguracyjne *Cassandry* (w `/etc/cassandra/cassandra.yaml`).

Parametr	Linia	Komentarz
<code>cluster_name</code>	10	nazwa klastra, którego ten węzeł jest członkiem
<code>seed_provider/parameters/seeds</code>	192	Węzeł <i>Cassandry</i> nie odkrywa innych węzłów automatycznie, stąd dla algorytmu plotkowania wymagana jest lista początkowych "punktów kontaktowych". Oczywiście punkty powinny być tak dobrane by nie dopuścić do partycjonowania klastra.
<code>listen_address</code>	297	Adres IP kanału komunikacyjnego pomiędzy węzłami. Agent algorytmu plotkującego będzie nasłuchiwał na tym adresie. Jeżeli wartość nie zostanie tutaj podana, <i>Cassandra</i> wybierze adres IP localhosta.
<code>start_native_transport</code>	310	Czy uruchomić binarny protokół transportowy wykorzystywany przez drivery DataStax.
<code>native_transport_port</code>	312	Port dla binarnego protokołu transportowego. Jest on podawany w kodzie klienta gdy ten łączy się z określonym węzłem klastra.
<code>start_rpc</code>	324	Czy uruchomić serwer <i>Thrift RPC</i> . Wyłączenie serwera RPC uniemożliwi dostęp do węzła przez np. <code>cqlsh</code> .
<code>rpc_address</code>	335	Adres IP na którym ma nasłuchiwać serwer <i>Thrift</i> .

4.4 Weryfikacja stanu węzła

Szeroko pojęty stan węzła po konfiguracji można sprawdzić na kilka sposobów. Przede wszystkim po uruchomieniu usługi należy sprawdzić czy ta faktycznie działa i gdy tak nie jest, przejrzeć log.

W następnej kolejności można sprawdzić czy *Cassandra* nasłuchuje na portach zdefiniowanych w pliku konfiguracyjnym:

Listing 12: sprawdzanie na których portach nasłuchuje *Cassandra*

```
$ netstat -ln4p #listening, numeric, IPv4 only, with owner process information
```

Kolejnym sposobem jest próba połączenia się z węzłem poprzez Thrift API.

Listing 13: dostęp do *Cassandra* przez *Thrift API*.

```
$ cqlsh localhost 9160 #use "quit;" to exit CQL shell
```

Wreszcie można wyświetlić stan klastra, do którego należy węzeł:

Listing 14: sprawdzanie stanu klastra

```
$ nodetool --host localhost -p 7199 status # 7199 is the JMX port
Datacenter: datacenter1
=====
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
-- Address Load Tokens Owns (effective) Host ID Rack
UN 127.0.0.1 98.76 KB 256 100.0% d85efd66-29fb-46d7-b824-a4cc3fc4e75b rack1
```

5 Konfiguracja klastra

5.1 Łączenie węzłów w klaster

Zakładając, że odpowiednie węzły spełniają następujące warunki:

- nazwa klastra w plikach konfiguracyjnych wszystkich węzłów jest taka sama
- agent algorytmu plotkującego nasłuchuje na adresie IP interfejsu osiągalnego z sieci lokalnej (parametr `listen_address`)
- początkowe punkty kontaktowe są zdefiniowane tak by każdy węzeł mógł ostatecznie dowiedzieć się o wszystkich pozostałych

by połączyć je w klaster wystarczy jedynie uruchomić na nich usługę `cassandra`.

5.2 Weryfikacja konfiguracji klastra

Połączenia między węzłami klastra mają charakter dwukierunkowy. Oznacza to, że jeżeli węzeł nie jest widoczny w sieci, to (o ile działa) sam także nie widzi jej pozostałej części.

Stąd najprostszym sposobem na sprawdzenie czy wszystkie węzły zostały odpowiednio połączone jest sprawdzenie jaki stan klastra widzi dowolny z jego węzłów. Informację tę może podać narzędzie `nodetool`.

Listing 15: sprawdzanie stanu klastra po dołączeniu węzłów

```
$ nodetool -host localhost -p 7199 status
Note: Ownership information does not include topology; for complete information, specify a keyspace
Datacenter: DC1
=====
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
-- Address Load Tokens Owns Host ID Rack
UN 10.0.0.101 63.38 KB 256 35.3% c6b6661a-c501-45ad-acc1-3e17fea6d241 NODE1AND2
UN 10.0.0.100 63.48 KB 256 31.0% c45900c6-6866-4b5a-b32b-cc0158bca524 NODE0
UN 10.0.0.102 63.5 KB 256 33.7% 4d598617-7f3c-480c-9310-f0eae90ef96b NODE1AND2
```

Jeżeli jeden z węzłów nie dołączył do klastra lub znajduje się w stanie DOWN, należy sprawdzić czy uruchomił się prawidłowo, czy log nie zawiera informacji o błędzie sieci lub agenta algorytmu plotkującego (w pewnych sytuacjach taki błąd nie powstrzymuje usługi przed uruchomieniem), wreszcie czy adres IP dla agenta i początkowe punkty kontaktowe zostały poprawnie zdefiniowane w pliku konfiguracyjnym.

UWAGA! Z zawartości pliku konfiguracyjnego `/etc/cassandra/cassandra.yaml` można wywnioskować, że zawsze jednym z punktów kontaktowych powinien być localhost. Jako adres IP punktu nie powinien być jednak podawany adres 127.0.0.1, lecz adres interfejsu na którym nasłuchuje agent (parametr `listen_address`).

5.3 Działania dodatkowe

W obecnym stanie klastr powinien już działać prawidłowo. Każdy węzeł powinien przyjmować od użytkowników zapytania i odpowiadać na nie w sposób adekwatny do stanu klastra.

Jednak by zapewnić stabilne działanie klastra przez dłuższy okres wymagane są dodatkowe czynności administracyjne.

5.3.1 Synchronizacja zegarów

Dokumentacja Cassandra udostępniona przez Apache Software Foundation wspomina o tym, że do oceny czy propagowane do węzłów zmiany schematu (np. dodanie nowej rodziny kolumn) są przedawnione (zatem powinny być zignorowane) wykorzystywany jest lokalny zegar czasu rzeczywistego (RTC).

By zapewnić poprawność propagacji takich zmian należy synchronizować zegary węzłów np. poprzez protokół NTP.

W systemie Debian funkcję synchronizacji czasu systemowego z serwerem NTP dostarcza pakiet `ntp`. Pakiet ten jest rekomendowany do instalacji przez pakiet `cassandra`. Po instalacji automatycznie zostaje uruchomiona usługa `ntp` bazująca na puli serwerów dostarczonych razem z pakietem. Należy sprawdzić czy usługa ta jest zainstalowana, uruchomiona i czy serwery NTP są dostępne:

Listing 16: weryfikacja poprawności działania klienta NTP

```
$ dpkg -l | grep ntp
ii ntp 1:4.2.6.p5+dfsg-2 i386 Network Time Protocol daemon and utility programs
$
$ service ntp status
[ ok ] NTP server is running.
$
$ ntpq -p # lists the NTP servers being queried
      remote refid st t when poll reach delay offset jitter
=====
*ntp.tktelekom.p 194.29.130.252 2 u 146 256 377 7.188 9.360 0.889
+afrodyta.comple 194.29.130.252 2 u 89 256 377 11.952 10.228 1.769
xpcolka.of.pl 80.50.231.226 2 u 35 256 377 13.027 190.872 26.614
+96-7.cpe.smnt.p 213.199.225.30 3 u 55 256 377 7.046 9.511 1.096
$
```


5.3.2 Czyszczenie

5.3.3 Konfiguracja uwierzytelniania

6 Zarządzanie klastrem

6.1 Monitoring węzłów i klastra

6.2 Przyłączanie i odłączanie węzłów

6.3 Backup danych

Literatura

- [1] Jan Baranowski, Michał Kaik, Maciej Urbański,
Cassandra, Highly available, scalable storage system,
Politechnika Poznańska 2014
- [2] E. Hewitt,
Cassandra, The Definitive Guide,
O'Reilly 2010
- [3] *DataStax*,
<http://www.datastax.com/>,
dostęp 17 czerwca 2014 r.
- [4] Daniel Stavrovski,
Installing Oracle JAVA 7 on Debian Wheezy,
<http://d.stavrovski.net/blog/post/installing-oracle-java-7-on-debian-wheezy>,
dostęp 16 czerwca 2014 r.
- [5] *Java/Sun, Debian Wiki*,
<https://wiki.debian.org/Java/Sun>,
dostęp 16 czerwca 2014 r.
- [6] *MySQL Connectors*,
<http://www.mysql.com/products/connector/>,
dostęp 16 czerwca 2014 r.
- [7] *NoSQL Apache Cassandra Documentation*,
<http://planetcassandra.org/documentation/>,
dostęp 16 czerwca 2014 r.
- [8] *Why DataStax?*,
<http://www.datastax.com/why-datastax>
dostęp 16 czerwca 2014 r.