## Inleiding Cassandra

Apache Cassandra is een opensource, NoSql (Not only Sql) database die de dag van vandaag door vele internationale bedrijven in gebruik is genomen. Bedrijven zoals Netflix, Instagram en het CERN hebben in Cassandra een performante, stabiele databank gevonden, die de beschikbaarheid van hun data garandeert.

Deze garantie wordt geboden door de structuur waarmee Cassandra data beheert. Een databank bestaat uit een collectie van noden die een cluster vormen. Het beheer gebeurt decentraal, elke node heeft dezelfde rol, dit maakt dat wanneer een node offline is, de andere ongehinderd blijven werken. Om het volle potentieel van Cassandra te benutten, is het aangeraden om de noden niet enkel over verschillende servers maar ook over verschillende datacenters te verspreiden. Bij deze setup mag er een datacenter offline gaan, de gebruikers zullen hier geen weet van hebben. En wanneer het datacenter terug online is, zal de data gedeeld worden met de herstelde noden. Dit is ook positief voor de schaalbaarheid. Wanneer een node wordt toegevoegd, haalt deze zijn configuratie op bij een bestaande node en sluit zich naadloos aan in de cluster.

Bij het aanmaken van een cluster kan men een replicatie factor meegeven. Deze geeft aan op hoeveel verschillende noden een record moet bewaard worden. Bij een factor 3 zal elk databaserij op 3 noden beschikbaar zijn. Des te groter te factor, des te groter de performantie en stabiliteit maar ook de duplicatie aan data vergroot mee.

Eén van de grootste verschillen met een standaard relationele databank is dat het linken van verscheidene tabellen niet mogelijk is. In plaats daarvan werkt Cassandra met collecties die rechtstreeks in een kolom worden opgeslagen. Dit wordt mogelijk door het aanmaken van UDTs (User Defined Types). Wanneer een lijst van UDTs (of primitieve types) wordt opgeslagen, wordt deze omgevormd naar een json-achtige structuur. Doordat alle data in één rij zitten, hoeven er geen joins gedaan te worden over meerdere tabellen wat de performantie aanzienlijk verhoogt.

Er zijn ook enkele nadelen verbonden aan het kiezen voor Cassandra als database, één van deze nadelen is dat het, out-of-the-box, onmogelijk is om tabellen te querieën zoals je een SQL zou doen. Dit komt omdat Cassandra werkt met key-value paren om de data op te slagen en dus enkel aan de hand van een key (of deel van een key) naar een record gezocht kan worden. Dit heeft ook als gevolg dat aggregatie functies enkel op partitie niveau kunnen worden uitgevoerd.

Aangezien Cassandra een non-relationele databank is, is een ander nadeel de afwezigheid van transacties op databank niveau. Enkel op rijniveau kan men gebruik maken van lightweight transacties door een conditie toe te voegen voor een schrijf operatie. Hiervoor zou men bv. een version id kunnen toevoegen aan een tabel zodat wanneer men een schrijf operatie wil doen, er gekeken kan worden of de versies nog matchen. Indien dit niet het geval is wordt de operatie niet uitgevoerd en de gebruiker hiervan op de hoogte gebracht.

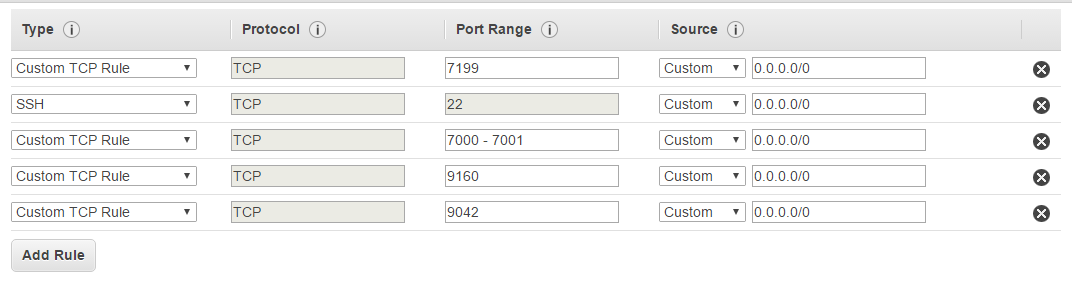
## Een Cassandra cluster opzetten

Nu we onze databank gekozen hebben, is het tijd dat we deze opzetten. De stappen die nodig zijn, beginnend van niets, om een cluster op te zetten worden in dit segment besproken. We zullen eerst de machines aanmaken, vervolgens Cassandra installeren en als laatste stap onze nodes met elkaar in contact brengen. Voor dit project heb ik gekozen om met 3 nodes te werken, dit om toch enige performantie te hebben zonder het prijzig te maken.

**De machines**

Als eerste maken we de 3 machines aan waarop we Cassandra zullen draaien. Via de AWS-console navigeren we naar EC2 -> Instances waar we de ‘Launch instance’ kiezen. Het eerste scherm laat ons kiezen welk besturingssysteem we willen gebruiken, hier kiezen we Amazon Linux AMI. In stap 2 bepalen we hoe krachtig onze machines zullen zijn. Het is perfect mogelijk om met de t2.micro (deze is gratis te gebruiken bij een ‘Free Tier’) maar met slecht 1 CPU en 1GB RAM heeft deze niet zo veel te bieden. Hier heb ik voor de t2.large geopteerd, deze zal beter aansluiten bij onze behoeften. Stap 3 hoeven we enkel het aantal instances dat we willen aanmaken wijzigen naar 3, de rest laten we ongewijzigd. Stap 4 en 5 slagen we over, deze zijn niet van toepassing. In Stap 6 maken we een nieuwe security group aan die de poorten nodig voor het gebruik van onze databank openzet. Voor onderstaande poorten maken we een ‘Custom TCP Rule’ en maken we ze compleet publiek (in productie omgevingen is dit ten strengste af te raden.) door de toegestane IP-adressen op ‘0.0.0.0/0’ te zetten.

* 9042: wordt gebruikt om met een client een verbinding aan te gaan.
* 7000: wordt gebruikt voor communicatie tussen de nodes
* 7160: wordt gebruikt door Thrift clients
* 7199: JMX, monitoring van de node
* 22: SSH port



Indien deze stap afgehandeld is, krijgen we een overzicht te zien. Als alles naar wens is klikken we op ‘Launch’ waardoor er een prompt opengaat. Vanuit deze prompt kunnen we een pem-file aanmaken en downloaden, het is zeer belangrijk om deze op een veilige plaats op te slaan, deze nodig is om via ssh een connectie naar onze machines te maken.

**Cassandra als service**

Nu we onze machines hebben is het tijd om hiervan Cassandra nodes te maken. Via het ‘Instance’ scherm kunnen we achterhalen wat de publieke IP-adressen zijn van deze machines (laten we ervan uitgaan dat deze 1.0.0.1,1.0.0.2 en 1.0.0.3 zijn.). Volgende stap moet op elke machine identiek herhaalt worden buiten de nodige aanpassingen aan de cassandra.yaml. Laten we eerst op onze machines inloggen via het commando:

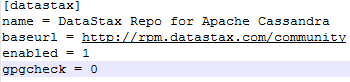
ssh – i ‘verkregen pem-file’ [ec2-user@1.0.0.1](mailto:ec2-user@1.0.0.1)

Nu zijn we ingelogd als ec2-user op onze machine. Om Cassandra 3.x te kunnen draaien hebben we Java 1.8 nodig, via het commando ‘java -version’ kunnen we dit controleren. Indien de versie lager ligt, zijn we verplicht om Java 1.8 te installeren. Een mogelijke manier om dit te doen is als volgt:

1. ga naar het path van de huidige jvm
   1. cd /usr/lib/jvm
2. download de java 1.8 tar
   1. sudo wget --no-cookies --no-check-certificate --header "Cookie: gpw\_e24=http%3A%2F%2Fwww.oracle.com%2F; oraclelicense=accept-securebackup-cookie" <http://download.oracle.com/otn-pub/java/jdk/8u121-b13/e9e7ea248e2c4826b92b3f075a80e441/jdk-8u121-linux-x64.tar.gz>
3. pak deze uit
   1. sudo tar xzf jdk-8u121-linux-x64.tar.gz
4. voeg de java 1.8 jar toe als java alternatief
   1. sudo alternatives --install /usr/bin/java java /usr/lib/jvm/jdk1.8.0\_121/bin/java 2
5. zorg dat deze nu als default jvm gebruikt word
   1. sudo alternatives --config java
   2. kies 2 en bevestig

Als alles correct verlopen is zien we nu 1.8 staan wanneer we opnieuw het commando ‘java -version’ uitvoeren.

Nu volgt de installatie van Cassandra zelf, hierbij is de eerste stap het toevoegen van de datastax.repo zodat we via het yum commando Cassandra kunnen installeren. Als dit gebeurt is kunnen we via yum zowel cassandra als nodetool installeren.

1. Maak het bestand datastax.repo aan
   1. sudo touch /etc/yum.repos.d/datastax.repo
2. bewerk deze met de nodige data
   1. sudo nano /etc/yum.repos.d/datastax.repo
   2. 
   3. Opslaan en nano afsluiten
3. Installeer Cassandra
   1. sudo yum install dsc30
4. Installeer Nodetool
   1. sudo yum install cassandra30-tools

Als we nu Cassandra zouden opstarten op de 3 machines, hebben we 3 clusters met elks 1 node gemaakt. Vanzelfsprekend is dit niet het gezochte resultaat en willen we 1 cluster met 3 nodes, om dit mogelijk te maken moeten we volgende aanpassingen doen aan de cassandra.yaml (locatie: /etc/cassandra/conf/cassandra.yaml):

1. cluster een zelfgekozen naam geven
   1. cluster\_name: 'Aem-cluster'
2. seeds aanpassen
   1. seeds: "1.0.0.1,1.0.0.2"
3. listen\_address leegmaken zodat deze gelijk is aan het broadcast adres
   1. listen\_address:
4. publieke IP-adres van de machine toekenen aan het broadcast\_address
   1. bv voor instance 1.0.0.1 -> broadcast\_address: 1.0.0.1
5. rpc\_address publiek zetten zodat client met onze cluster kunnen connecteren
   1. rpc\_address: 0.0.0.0
6. broadcast\_rpc\_address wijzigen naar publieke IP-adres
   1. bv voor instance 1.0.0.1 -> broadcast\_rpc\_address: 1.0.0.1

Nu dat onze yamls correct staan kunnen we op onze machines cassandra starten met het commando ‘sudo service cassandra start’. Na enkele minuten zien we dan dat de 3 nodes elkaar gevonden hebben via het commando ‘nodetool status’.

Nu we onze databank hebben kunnen we onze nodige modellen toevoegen, het gebruikte script kan je terugvinden in de bijlagen. Eenmaal onze databank operationeel is kunnen we beginnen aan de service die deze zal aanspreken.