Appendix: Neo4J Setup on AWS

E-63 Big Data Analytics
Harvard University, Autumn 2017

Tim Hagmann

September 12, 2017

Contents

Neo4J Setup

The following tutorial is concerned with the installation of Neo4J, a big data capable graph database. A graph database is a database that uses graph structures for semantic queries with nodes, edges and properties to represent and store data. A key concept of the system is the graph (or edge or relationship), which directly relates data items in the store. The relationships allow data in the store to be linked together directly, and in many cases retrieved with one operation. This contrasts with relational databases that permit managing the data in its natural structure (without imposing implementation aspects like physical record chains).

Neo4j is available in a GPL3-licensed open-source "community edition", with online backup and high availability extensions licensed under the terms of the Affero General Public License. Neo4j is implemented in Java and accessible from software written in other languages such as Python or R.

Because different OS enviroments can lead to different results, a cloud solution with AWS might be advisable. The following instructions will focus on a single, well-defined goal: setting up a Neo4J database in the cloud. The only prerequisite is an AWS account.

Deploy EC2

Log in to the AWS console (https://aws.amazon.com) and click on the EC2 icon under compute.

Create instance

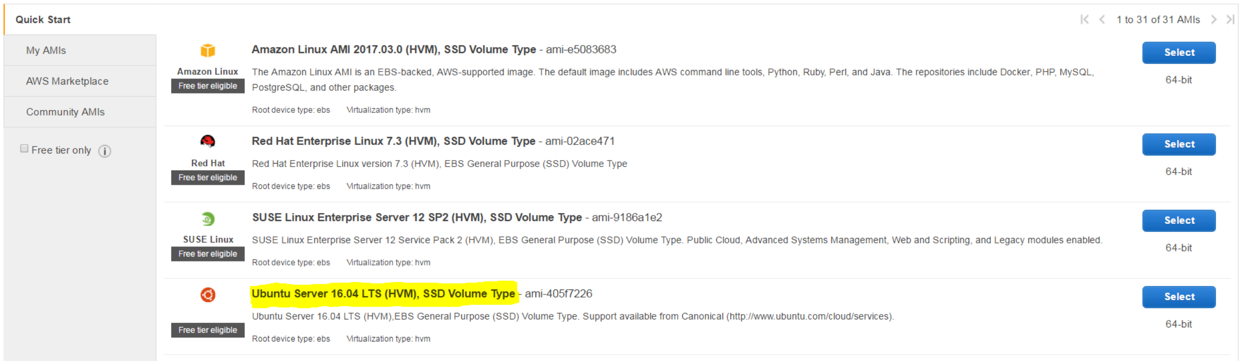
Search for "EC2" in the search bar. Click on "EC2" to start the EC2 wizard.

Step 1: Choose an Amazon Machine Image (AMI)

- Select the Ubuntu Server 16.04 image

Step 2: Choose an Instance Type

- Select the free tier t2.micro instance. This can also be changed later on when more capacity is needed.



Amazon Linux AMI 2017.03.0 (HVM), SSD Volume Type - ami-e5083683

Amazon Linux · Free tier eligible

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.

Root device type: ebs Virtualization type: hvm

Select · 64-bit

Red Hat Enterprise Linux 7.3 (HVM), SSD Volume Type - ami-02ace471

Red Hat · Free tier eligible

Red Hat Enterprise Linux version 7.3 (HVM), EBS General Purpose (SSD) Volume Type

Root device type: ebs Virtualization type: hvm

Select · 64-bit

SUSE Linux Enterprise Server 12 SP2 (HVM), SSD Volume Type - ami-9186a1e2

SUSE Linux · Free tier eligible

SUSE Linux Enterprise Server 12 Service Pack 2 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.

Root device type: ebs Virtualization type: hvm

Select · 64-bit

Ubuntu Server 16.04 LTS (HVM), SSD Volume Type - ami-405f7226

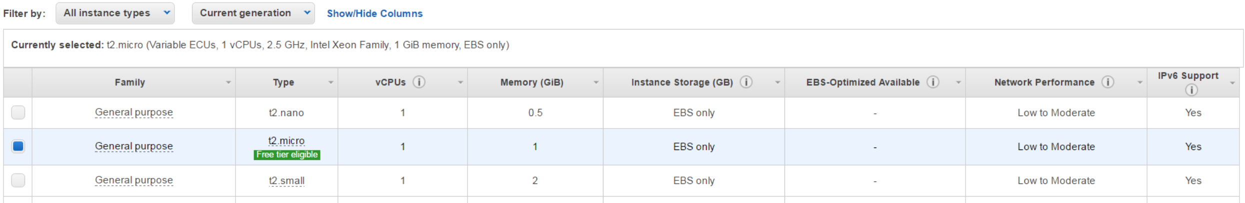
Free tier eligible

Ubuntu Server 16.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Root device type: ebs Virtualization type: hvm

Select · 64-bit

Figure 1: "Choose an Amazon Machine Image (AMI)"

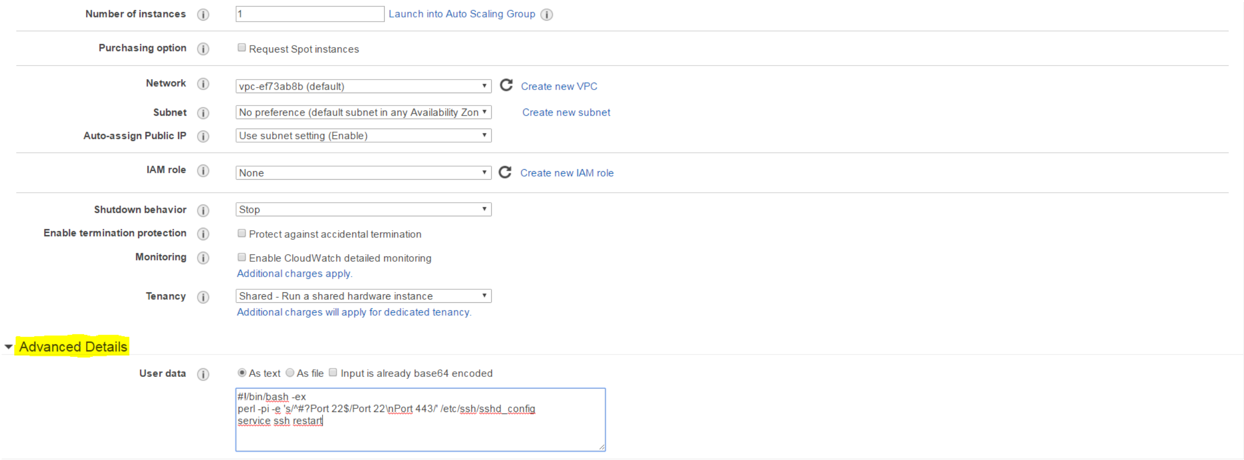


Filter by: All instance types Current generation Show/Hide Columns

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
□	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
■	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
□	General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes

Figure 2: "Choose an Instance Type"



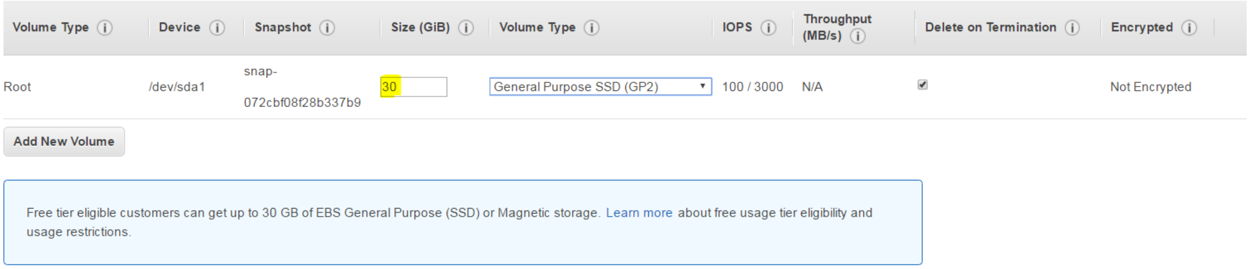
Number of instances	1 Launch into Auto Scaling Group
Purchasing option	□ Request Spot instances
Network	vpc-ef73ab8b (default) Create new VPC
Subnet	No preference (default subnet in any Availability Zon Create new subnet
Auto-assign Public IP	Use subnet setting (Enable)
IAM role	None Create new IAM role
Shutdown behavior	Stop
Enable termination protection	□ Protect against accidental termination
Monitoring	□ Enable CloudWatch detailed monitoring Additional charges apply.
Tenancy	Shared - Run a shared hardware instance Additional charges will apply for dedicated tenancy.

Advanced Details

User data ● As text ○ As file □ Input is already base64 encoded

```
#!/bin/bash -ex
perl -pi -e 's/^#?Port 22$/Port 22\nPort 443/' /etc/ssh/sshd_config
service ssh restart
```

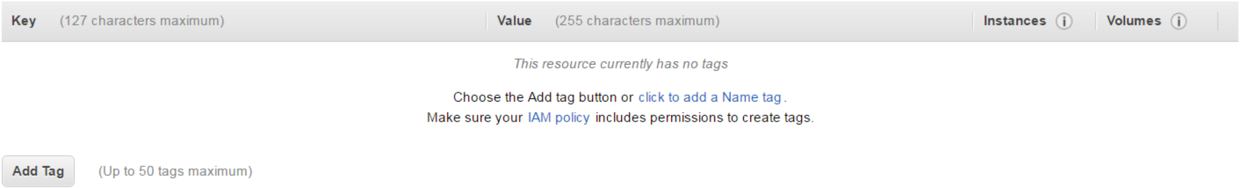
Figure 3: "Configure Instance Details"



Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encrypted
Root	/dev/sda1	snap-072cbf08f28b337b9	30	General Purpose SSD (GP2)	100 / 3000	N/A	☑	Not Encrypted

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. Learn more about free usage tier eligibility and usage restrictions.

Figure 4: “Add Storage”



Key (127 characters maximum)	Value (255 characters maximum)	Instances	Volumes

This resource currently has no tags

Choose the Add tag button or click to add a Name tag.
Make sure your IAM policy includes permissions to create tags.

(Up to 50 tags maximum)

Figure 5: “Add Tags”

Step 3: Configure Instance Details

- When using other instances such as p2.xlarge, click on “Request Spot Instances”. Spot instances are significantly cheaper than normal instances. It is a way for Amazon to sell excess capacity at reduced prices.
- (Optional) Click on “Enable CloudWatch Detailed Monitoring”. This will enable additional services like automatically shutting down an idle instance. You will be warned that additional charges may be incurred, which will go against your allotment. Consider it like buying insurance.
- (Optional) Under Advanced Details a custom startup script can be run. This can be useful, when you’re behind a company firewall and the ssh port (22) is blocked. In order to circumvent this, the following bash/perl script can be run. It runs the ssh on port 443:

```
#!/bin/bash -ex
perl -pi -e 's/^#?Port 22$/Port 22\nPort 443/' /etc/ssh/sshd_config
  service ssh restart
```

Step 4: Add Storage

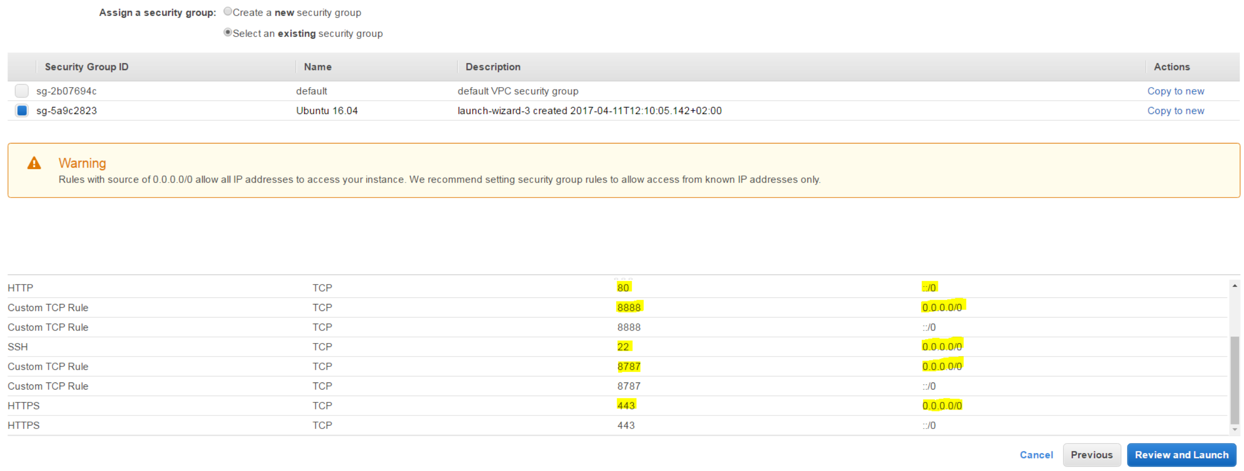
- The default SSD storage of 8 GB is sometimes not enough when installing all the necessary software. That is why 30GB is chosen.

Step 5: Add Tags

- Nothing is changed/done here.

Step 6: Configure Security Group

- This is an important step. If the necessary ports are not opened, it isn’t possible to connect to neo4j Server.
- The following ports are opened: 80 (HTTP), 22 (SSH), HTTPS (443), RStudio Port (8787), Jupyter Port (8888), Neo4J http (7474), Neo4J https (7473), Neo4J Bolt (7687).
- It is allowed to connect to them from any IP (0.0.0.0). It would also be possible to restrict the IP Range.



Assign a security group: ○ Create a **new** security group
● Select an **existing** security group

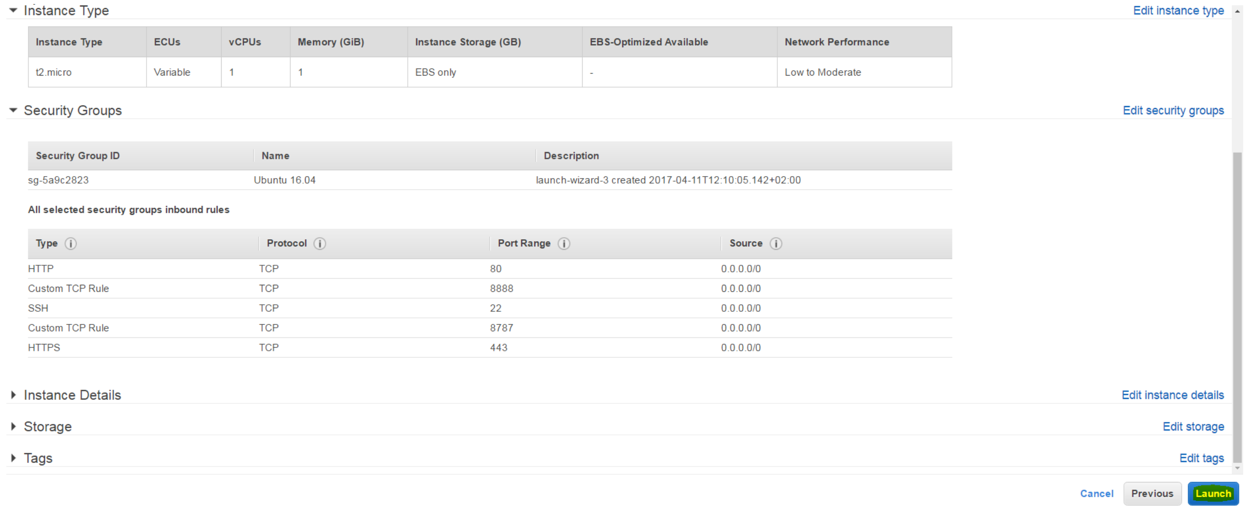
	Security Group ID	Name	Description	Actions
□	sg-2b07694c	default	default VPC security group	Copy to new
■	sg-5a9c2823	Ubuntu 16.04	launch-wizard-3 created 2017-04-11T12:10:05.142+02:00	Copy to new

⚠ Warning
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

HTTP	TCP	80	::/0
Custom TCP Rule	TCP	8888	0.0.0.0/0
Custom TCP Rule	TCP	8888	::/0
SSH	TCP	22	0.0.0.0/0
Custom TCP Rule	TCP	8787	0.0.0.0/0
Custom TCP Rule	TCP	8787	::/0
HTTPS	TCP	443	0.0.0.0/0
HTTPS	TCP	443	::/0

Figure 6: "Configure Security Group"

Step 7: Review Instance Launch



▾ Instance Type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

▾ Security Groups

Security Group ID	Name	Description
sg-5a9c2823	Ubuntu 16.04	launch-wizard-3 created 2017-04-11T12:10:05.142+02:00

All selected security groups inbound rules

Type	Protocol	Port Range	Source
HTTP	TCP	80	0.0.0.0/0
Custom TCP Rule	TCP	8888	0.0.0.0/0
SSH	TCP	22	0.0.0.0/0
Custom TCP Rule	TCP	8787	0.0.0.0/0
HTTPS	TCP	443	0.0.0.0/0

▸ Instance Details

▸ Storage

▸ Tags

- When you attempt to launch the instance it will ask if you have a keypair. - If you have not used SSH before and do not have a keypair, select "Create a new keypair" - Download the .pem file and store it in a safe place - You will need to import the .pem file to connect your instance over ssh. See below.

Login EC2

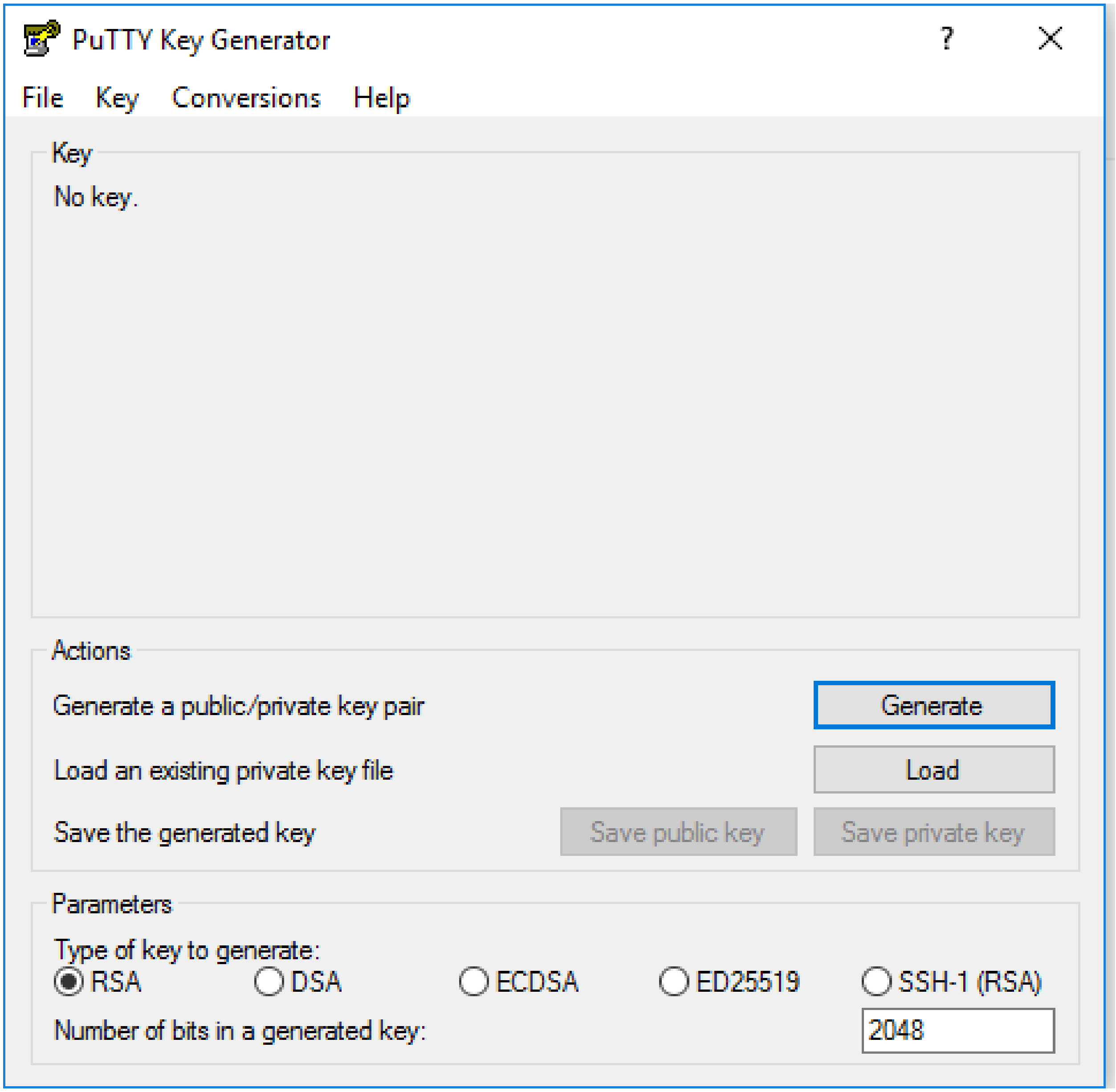
Connect to the instance over ssh which establishes a terminal session to your newly created instance.

For Windows 7 Users

Windows users may not have an ssh client installed. If you need ssh for Windows, download PuTTY (http://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html).

You will need to convert the .pem key from AWS into a .ppk key in PuTTY Key Generator - Select Import Key under Conversions - Save as Private Key with RSA selected

The below screenshot shows the PuTTy key generator.



No key.

Generate a public/private key pair

Load an existing private key file

Save the generated key

Type of key to generate:
● RSA ○ DSA ○ ECDSA ○ ED25519 ○ SSH-1 (RSA)

Number of bits in a generated key: 2048

Figure 7: “PuTTy key generator”

Launch PuTTY and you will see the below screen.

Navigate to SSH -> Auth, browse to add the .ppk file and Open.

Login as user ubuntu.

For MacOS and Linux or Windows 10 users

Open a terminal window. **Note:** On a Mac or on Linux you may need to change the permissions of your ssh keyfile if you get the following error when attempting to ssh to your instance:

Permission denied (publickey)

Change the permission of your ssh keyfile as follows if you get the above error message.

```
chmod 600 <ssh_key>
```

Use the public IP of the running instance and username "ubuntu". For example:

```
ssh -i <your_ssh_keyfile> ubuntu@<your_AWS_public_DNS_name_or_IP_address>
```

Add new user

```
sudo adduser <username>
sudo adduser <username> sudo
```

Copy ssh key

Copy the ssh key to the new user

```
sudo cp ~/.ssh/authorized_keys /home/<username>/.ssh/authorized_keys
su <username>
cd ~
chmod 700 .ssh
chmod 600 .ssh/authorized_keys
```

Enable Swapping

It might be necessary to enable swapping. This is especially the case with the smaller instances. You can make the size of 1024 also bigger.

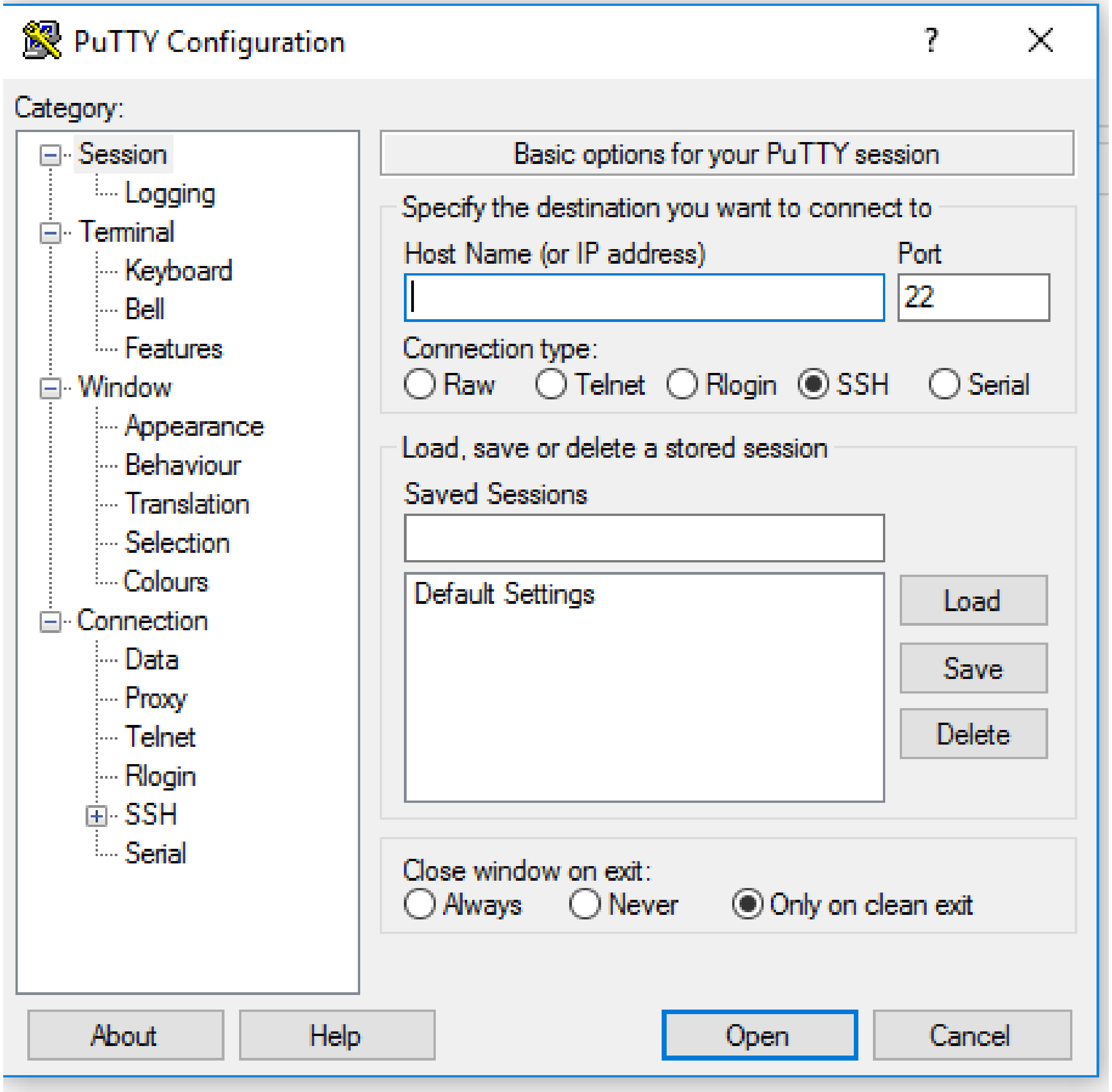
```
sudo /bin/dd if=/dev/zero of=/var/swap.1 bs=1M count=1024
sudo /sbin/mkswap /var/swap.1
sudo chmod 600 /var/swap.1
sudo /sbin/swapon /var/swap.1
```

In order to activate swapping at startup append the following line to the */etc/fsab* file.

```
sudo nano /etc/fstab
swap        /var/swap.1 swap    defaults        0   0
```

Install Neo4J

In order to facilitate the neo4j installation a small bash script can be downloaded and executed from my github account.



PuTTY Configuration

Category:

- Session
  - Logging
- Terminal
  - Keyboard
  - Bell
  - Features
- Window
  - Appearance
  - Behaviour
  - Translation
  - Selection
  - Colours
- Connection
  - Data
  - Proxy
  - Telnet
  - Rlogin
  - SSH
  - Serial

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address):

Port: 22

Connection type:
○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

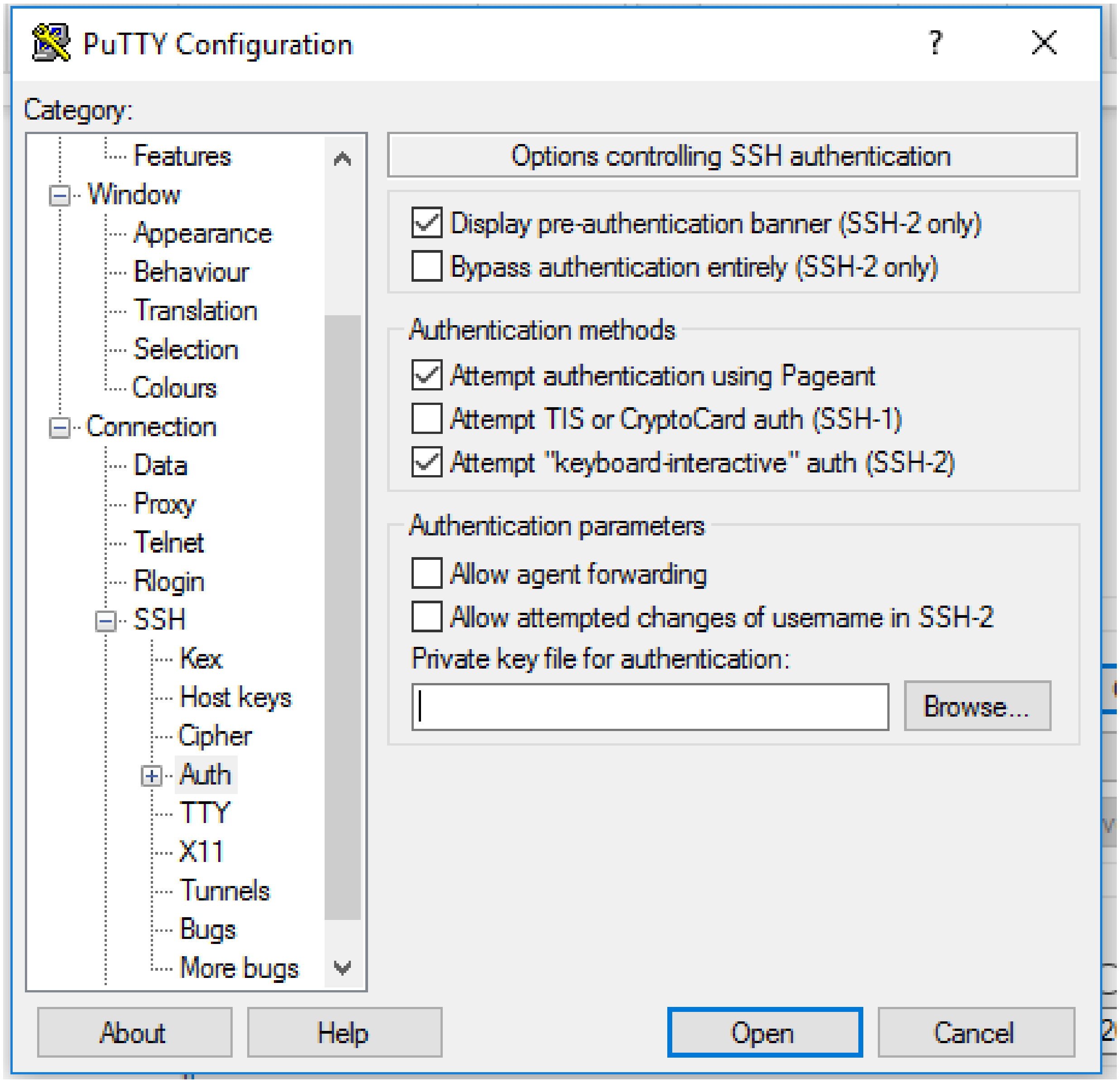
Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit:
○ Always ○ Never ● Only on clean exit

Figure 8: “Launch PuTTy”



Options controlling SSH authentication

☑ Display pre-authentication banner (SSH-2 only)

□ Bypass authentication entirely (SSH-2 only)

Authentication methods

☑ Attempt authentication using Pageant

□ Attempt TIS or CryptoCard auth (SSH-1)

☑ Attempt "keyboard-interactive" auth (SSH-2)

Authentication parameters

□ Allow agent forwarding

□ Allow attempted changes of username in SSH-2

Private key file for authentication:

Figure 9: “PuTTy Authentification”

```
wget https://raw.githubusercontent.com/greenore/linux-setup/master/setup_neo4j.sh
chmod +x setup_neo4j.sh
sudo ./setup_neo4j.sh
```

Service commands (optional)

The server should have started automatically and should also be restarted at boot. If necessary the server can be stopped with

```
service neo4j stop
```

and restarted with

```
service neo4j start
```

Edit server IP

```
sudo nano /etc/neo4j/neo4j.conf
```

Remove the # from the line below in the file

```
#org.neo4j.server.webserver.address=0.0.0.0
```

Restart the service

```
service neo4j restart
```

Accessing Neo4j

```
http://<your_AWS_public_DNS_name_or_IP_address>:7474/browser/
```