How to Setup a Node.js Server on AWS and Run a Universal React/Redux App.

1 Create a VM

- **Create a new AWS NodeJS instance via Bitnami: http://aws.bitnami.com**
  - you can use any other cloud clatform, but AWS works fine.
  - create a bitnami account
  - you will also need to create an AWS account to link to the Bitnami account.

---

2 Download React Boilerplate & Install Dependencies

- **Download the private key(s) off Bitnami dash and store them somewhere safe.**
  - when you visit that Bitnami dash for the first time, it may ask you to create a vault password, which **CAN NOT** be retrieved

- **Login via SSH:** `ssh -i [your-private-key].pem bitnami@99.172.34.32`
  - Change directory to `/home/bitnami/` and `mkdir projects` and `cd projects`
  - you can create a custom Node app in this folder and the following instructions will be essentially the same, but let's use a React/Redux Boilerplate which sets up a LOT of the otherwise tedious work and is configured with SEO in mind:

```
1. git clone https://github.com/ccurtin/universal-react-redux-boilerplat
   e-plus.git
2. cd universal-react-redux-boilerplate-plus
3. npm install
```

- **Start DEV server w/** `npm run dev`
- **Create PRODUCTION BUILD w/** `npm run build && npm start`
- The site isn't accessible just yet! We need to open port `3000` on AWS.

---

3 Open and Forward HTTP ports

- Open your AWS Console Panel (accessible via Bitnami Dashboard)
- Click **Security Groups** under **NETWORK & SECURITY**
- find the correct Group Name/Instance and click on it.
- in the Tabs below, click on **Inbound** and **Edit** to add a new rule.
- *Type* should be Custom TCP Rule, *port*: 3000, *Source*: Anywhere.

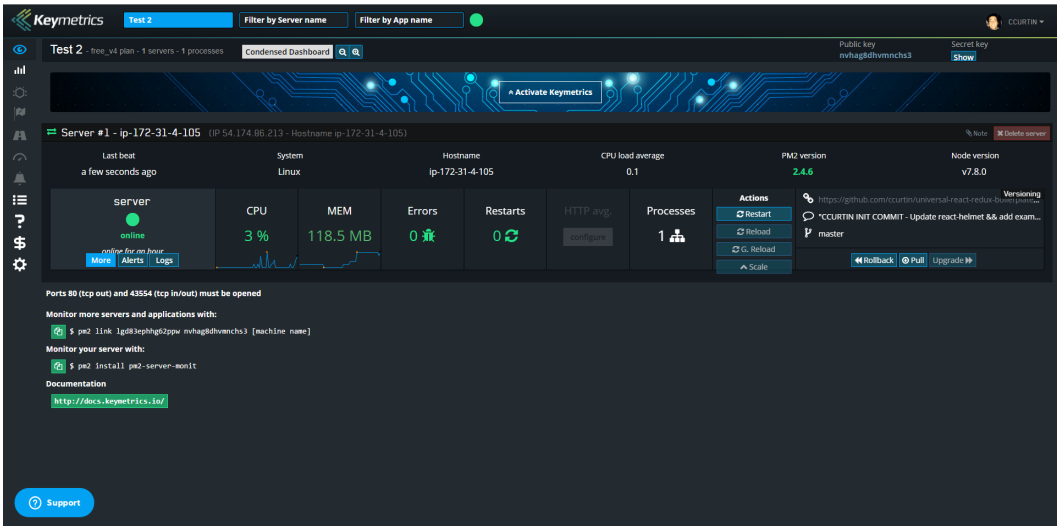
- Make sure to click Save.
- Now your app should now be accessible via your IP/URL with port 3000 (*you can find your IP/URL on Bitnami Dashboard*)

- But we want the site to be accessible via port 80 instead. Setting your app to port 80 directly is looked down upon so we will forward any traffic from port 80 to port 3000.
- **Forward all traffic from port `80` to port 3000`**
- `sudo iptables -I INPUT 1 -p tcp --dport 80 -j ACCEPT`
- `sudo iptables -A PREROUTING -t nat -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 3000`

---

4 Continuously Run Project on Server

- Next, we need to make sure the app continuously runs, we will use the node module **PM2** to do that.
- install PM2 (most likely will need to run as sudo(or `sudo su -` for root as sudo)): `sudo npm install pm2 -g`
- Now within your project folder, start the application: `pm2 start ./bin/server` OR for simple apps, similarly: `pm2 start app.js`
  Metrics:**
  - sign up at https://keymetrics.io/
  - create a new bucket
  - copy the command into your SSH console and DONE!
  - can monitor a process via SSH with: 'pm2 monit'
  - it's that easy! (they only offer a demo/very light weight dashboard. This is also a paid service)
  - **Keymetrics Example:**


Keymetrics
Test 2
Filter by Server name
Filter by App name
CCURTIN
Test 2 - free_v4 plan - 1 servers - 1 processes
Condensed Dashboard
Public key
nvhag8dhvmnchs3
Secret key
Show
Activate Keymetrics
Server #1 - ip-172-31-4-105 (IP 54.174.86.213 - Hostname ip-172-31-4-105)
Note
Delete server
Last beat
a few seconds ago
System
Linux
Hostname
ip-172-31-4-105
CPU load average
0.1
PM2 version
2.4.6
Node version
v7.8.0
server
online
online for an hour
More
Alerts
Logs
CPU
3 %
MEM
118.5 MB
Errors
0
Restarts
0
HTTP avg.
configure
Processes
1
Actions
Restart
Reload
G. Reload
Scale
Versioning
https://github.com/ccurtin/universal-react-redux-boilerplate...
"CCURTIN INIT COMMIT - Update react-helmet && add exam...
master
Rollback
Pull
Upgrade
Ports 80 (tcp out) and 43554 (tcp in/out) must be opened
Monitor more servers and applications with:
$ pm2 link lgd83ephhg62ppw nvhag8dhvmnchs3 [machine name]
Monitor your server with:
$ pm2 install pm2-server-monit
Documentation
http://docs.keymetrics.io/
Support