Platform as a Service

by William Gustafsson & Niklas Frisk

Setup

Run: “nodejs PaaS.js” on the server to launch.
Requirements:

- Ruby [3]
  - Fog [4]
- Nodejs [5] (“npm install XXX” for libraries)
  - express [6]
  - node-control [1]
  - child_process [7]
  - fs [8]

Usage

Using your webbrowser, you can access the server with 130.240.233.77:8080.
Choose your nodejs application to upload and what port it needs to run.
Press upload, then wait for a response.
Use the path in the response to find your uploaded application.

How we made it

Client side is fully web based, for REST communication with our server.
Server side we use nodejs[5] for everything except communication with openstack, where we use ruby and fog.
Why we use ruby is because to the api we tried for nodejs to openstack communication lacked features we wanted, this api was called pkgcloud[9].

Discussion

We found the lab entertaining and very educating. In hindsight we probably used too little of external finished products and did a lot of it ourselves.

Design notes/Discussion

- Currently the application does not handle existing virtual machines on the openstack server nor despawning of the virtual machines the application has spawned. We have chosen this approach due to development time constraints.
- The port is added to the subdomain name to distinguish between different sessions

uploading files with the same name.

- The load balancing and dns parts of the application is designed for only 2 ip's (and the server ip) available on the openstack server. These parts would need some redesigning to work with n ip's.
- Since the answer to the http post is delayed until all the computations are done, machines spawned and is ready to be used, the answer can take a couple of seconds to be received.
- During execution a log is created by node-control [1], this log contains the ssh commands done.
- dossh.js uses the node-control libraries[1], these however does not support dynamic ip addresses and needs to be rewritten to support the ip-ranges chosen if the ip addresses of the slaves should change.
- Since we do not change or check anything written in the nodejs files uploaded we decided that the user must provide the server with the port the application is using, this will be mapped as the internal port of the docker container.
- The DNS uses xip.io[10] for finding the subdomain. The subdomains are used to redirect the user to the uploaded programs.
- Since we create new machines and reuses ip's we got some problems with ssh security, this was solved by disabling key checking of the subnet [2].

Thanks

Johan Kristansson for help with the ruby script used to spawn virtual machines.

References

[1] https://github.com/tsmith/node-control
[2] http://linuxcommando.blogspot.se/2008/10/how-to-disable-ssh-host-key-checking.html
[3] https://www.ruby-lang.org/en/
[4] http://fog.io/
[5] http://nodejs.org/
[6] http://expressjs.com/
[7]http://nodejs.org/api/child_process.html
[8]http://nodejs.org/api/fs.html
[9]https://github.com/nodejitsu/pkgcloud
[10]http://xip.io/

Learning nodejs:

https://leanpub.com/nodebeginner