Distributed Systems
TP n°4 - Grid deployment

Hadrien Croubois
hadrien.croubois@ens-lyon.fr

3/3/2016

All documents are available on my website: `http://hadriencroubois.com/#Teaching`

Disclaimer

Today's session is only to be done by those who have completed the previous one. If you haven't finished implementing the sliding window algorithm, please do that first. Make sure implemented the failure simulation part as well.

Topic

In this session, we will discuss about the deployment of Erlang programs. So far, you saw how to spawn processes in an Erlang context. You also saw how to have those processes communicate one with another. However, building complex distributed systems requires much more.

Today, you will have several increasingly difficult goals. The objective here is not only to implement simple features but also to think about the technical elements that have to be implemented to achieve the more difficult goals.

Question 1

a) Calling `erlang:node/0` gives me `nonode@nohost`. What does that mean ? Why is node naming important ? How to solve that ?

b) Try running two Erlang context on your machine. Use `net_adm:ping/1` to check if the connection can be established. You should get a `pong`, if you get a `pang` then something is not working properly.

c) Using SSH, try running two Erlang context on two different machines. Can you establish a connection ? If not try using a full name to solve the DNS issues

d) Have a look at the functions `erlang:register/2`, `erlang:registred/0`, `erlang:whereis/1` and `erlang:unregister/1`. What do you think they are used for ?

e) **[hard]** Have a look at `erlang:spawn/4` and experiment with it. What do you thin of this behaviour.

f) **[hard]** How does `erlang:spawn/4` handles custom module. Discuss about a bootstrapping method to send code to remote servers before spawning stuff their.

g) **[harder]** Implement everything needed to compile code, launch and register processes remotely. You should make all cases are handled (unregister à name, trying to register an already used name ...). You should start by defining the communication interface.

h) **[hardest]** Deploy your code and show how it can be used. Impress me !