Build Your own Distributed Framework on Mesos using Clojure

Presented By:

Edward Paget
Prasanna Gautam

Sept 24,2015

Steps

1. Start Virtualbox VMs - Master and 2 Slaves
2. Generate a mesos-framework
3. Store State using Zookeeper
4. Recover from Failure
5. Leader Election
6. Web UI
7. Testing the Scheduler

What is Mesos

- Mesos is a cluster manager
  - Handles scaling up resources
  - Allows Dynamic Allocation
  - Resource Isolation
  - Works using resourceOffers to Frameworks

Mesos Essential Vocabulary

Term	Definition
Mesos Master	Coordinator, Offers Resources to Frameworks
Mesos Slaves	Where your tasks are run
Framework	Your Application on Mesos
Task	What Actually Runs
Scheduler	Handles Offers from mesos
Executor	Handles the launching of tasks
Zookeeper	Mesos uses for Coordination

Setup

Getting your Development Environment Ready

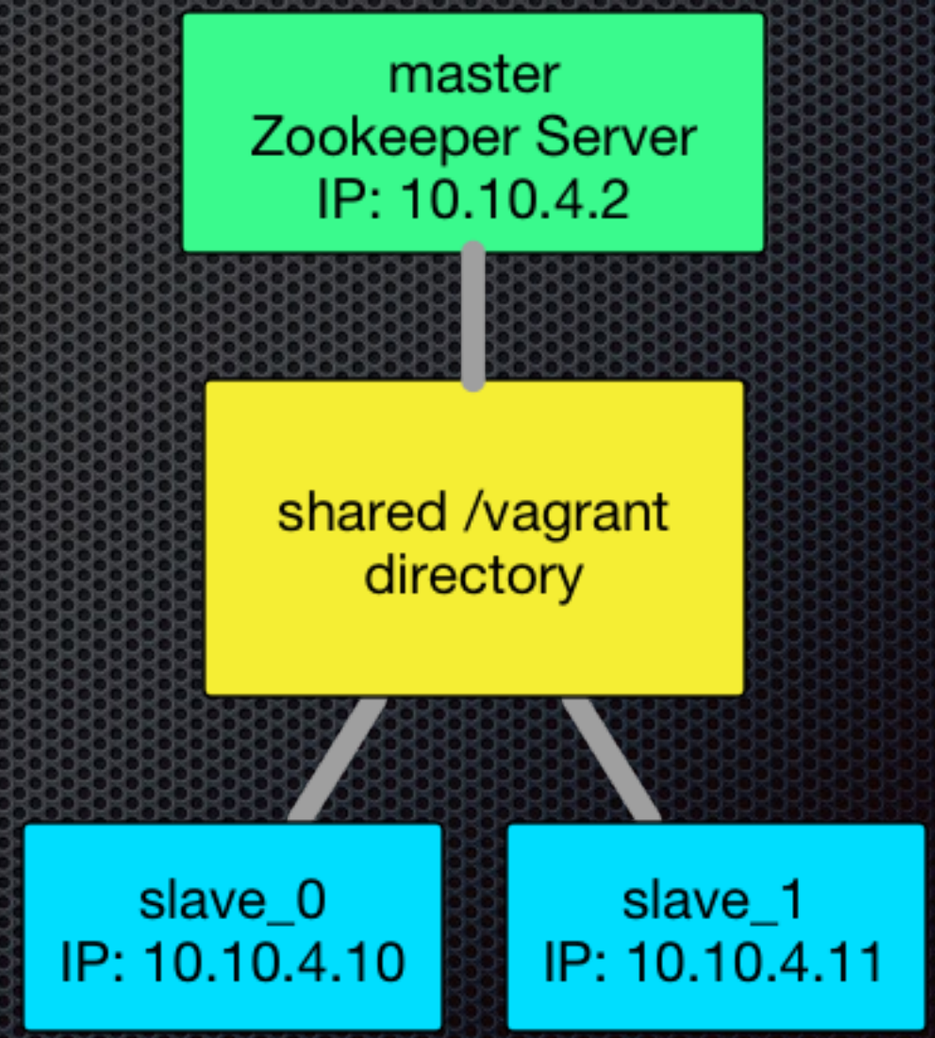
- Please insert the provided USB drives in your computers
- Requirements:
  - OS: Linux, OSX (Sorry we didn’t test on Windows)
  - Latest Vagrant and Virtualbox
    - They’re provided in the USB disk too

Setup

- Go to your USB Directory (/usb)
- * /Volumes/STL15MESOS on OSX
- * /media/STL15MESOS (most likely) on Linux
- Install Virtualbox from /usb/virtualbox
- Install Vagrant from /usb/vagrant/

Setup:Hello-Mesos

- cd /usb/code/hello-mesos
- vagrant up



Setup:Hello-Mesos

- run ./bin/diagnose-vagrant-setup.sh

```
➜ hello-mesos git:(master) ✗ ./diagnose-vagrant-setup.sh
10.10.4.2 master
Success:  Valid Master IP 10.10.4.2
imok
Success:  Zookeeper is running on Master
2
Success:  Valid number of slaves 2
10.10.4.2
Success:  Master IP in the machine matches with 10.10.4.2
mesos-master start/running, process 6046
Success:  mesos-master service is running
HTTP/1.1 200 OK
Success:  mesos-master is listening on http://10.10.4.2:5050
Checking 10.10.4.10
HTTP/1.1 200 OK
Success:  mesos-slave on 10.10.4.10 is reachable from Master
Checking 10.10.4.11
HTTP/1.1 200 OK
Success:  mesos-slave on 10.10.4.11 is reachable from Master
Checking from slave_0
HTTP/1.1 200 OK
Success:  mesos-master on http://10.10.4.2:5050 is rechable from slave_0
Checking from slave_1
HTTP/1.1 200 OK
Success:  mesos-master on http://10.10.4.2:5050 is rechable from slave_1
```

Demo: (go)

- vagrant ssh master
- cd /vagrant
- lein repl
- (go)

```
user=> (go)
I0923 19:14:33.185351  2559 sched.cpp:157] Version: 0
2015-09-23 19:14:33,187:2534(0x7f117b676700):ZOO_INFO
2015-09-23 19:14:33,188:2534(0x7f117b676700):ZOO_INFO
2015-09-23 19:14:33,188:2534(0x7f117b676700):ZOO_INFO
2015-09-23 19:14:33,188:2534(0x7f117b676700):ZOO_INFO
2015-09-23 19:14:33,189:2534(0x7f117b676700):ZOO_INFO
2015-09-23 19:14:33,189:2534(0x7f117b676700):ZOO_INFO
2015-09-23 19:14:33,190:2534(0x7f117b676700):ZOO_INFO
2015-09-23 19:14:33,190:2534(0x7f117b676700):ZOO_INFO
2015-09-23 19:14:33,191:2534(0x7f117b676700):ZOO_INFO
2015-09-23 19:14:33,193:2534(0x7f117a674700):ZOO_INFO
2015-09-23 19:14:33,194:2534(0x7f117a674700):ZOO_INFO
I0923 19:14:33.195088  2590 group.cpp:313] Group proc
I0923 19:14:33.195560  2590 group.cpp:787] Syncing gr
I0923 19:14:33.195868  2590 group.cpp:385] Trying to
I0923 19:14:33.197350  2590 detector.cpp:138] Detecte
I0923 19:14:33.198221  2590 group.cpp:656] Trying to
W0923 19:14:33.199096  2590 detector.cpp:444] Leading
I0923 19:14:33.199683  2590 detector.cpp:481] A new l
I0923 19:14:33.201305  2591 sched.cpp:254] New master
I0923 19:14:33.202042  2591 sched.cpp:264] No credent
I0923 19:14:33.204704  2591 sched.cpp:448] Framework
:ready
```

Demo: (Web Interface)

- http://10.10.4.2:5050/#/

Active Tasks

Find...

ID	Name	State	Started ▼	Host	
f7bfe0fe-11a5-4670-a40f-858b2684d734	hello-mesos	RUNNING	2 minutes ago	10.10.4.10	Sandbox
748688f9-fcb0-4a8c-9e42-07eef2668278	hello-mesos	RUNNING	2 minutes ago	10.10.4.11	Sandbox

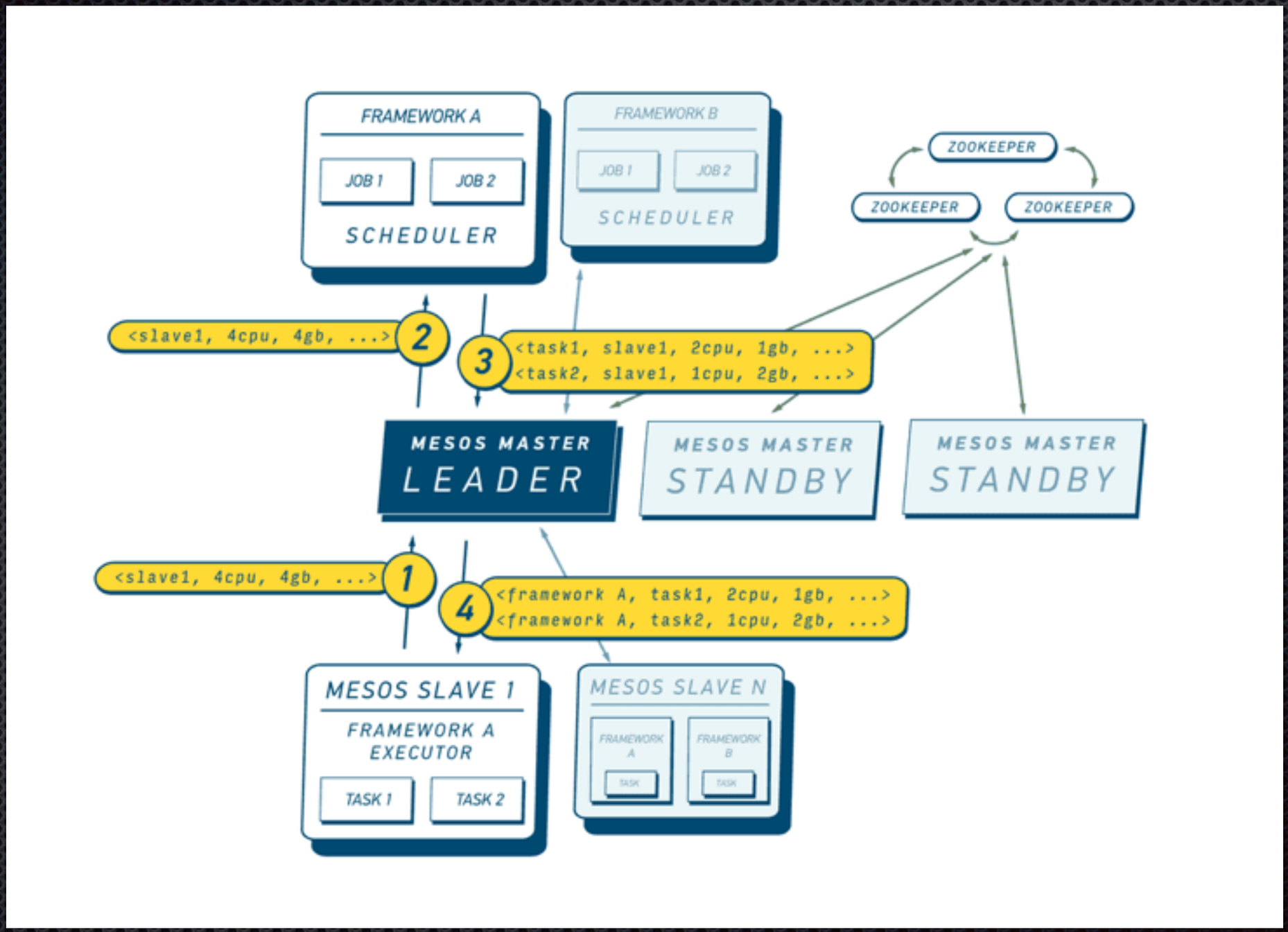
```
Registered executor on 10.10.4.10
Starting task f7bfe0fe-11a5-4670-a40f-858b2684d734
sh -c 'while true; do echo "Hey Mesos"; sleep 5; done'
Hey Mesos
Forked command at 2255
Hey Mesos
Hey Mesos
```

Active Frameworks

Find...

ID ▼	Host	User	Name	Active Tasks	CPUs	Mem	Max Share	Registered	Re-Registered
...5050-1817-0001	master	vagrant	hello-mesos	2	1	256 MB	50%	9 minutes ago	-
...5050-1817-0000	master	root	marathon	0	0	0 B	0%	12 minutes ago	-

How do Frameworks work?



https://www.antonlindstrom.com/2015/03/29/introduction-to-apache-mesos.html

Why Clojure?

- Clojure is a great example of “Making Easy Things Easy & Hard Things Possible”
- Distributed Systems are hard because of state management
  - You have to be very explicit
  - Clojure already forces you into this mindset
- Cleaner Code
- A REPL for your distributed system
- clj-mesos is really great!

Why Clojure? Java

```
Protos.TaskID taskId = Protos.TaskID.newBuilder()
    .setValue(Integer.toString(taskIDGenerator.incrementAndGet())).build();

Protos.ContainerInfo.DockerInfo.Builder dockerInfoBuilder = Protos.ContainerInfo.DockerInfo.newBuilder();
dockerInfoBuilder.setImage("busybox");

Protos.ContainerInfo.Builder containerInfoBuilder = Protos.ContainerInfo.newBuilder();
containerInfoBuilder.setType(Protos.ContainerInfo.Type.DOCKER);
containerInfoBuilder.setDocker(dockerInfoBuilder.build());

Protos.CommandInfo.Builder commandInfoBuilder = Protos.CommandInfo.newBuilder();
commandInfoBuilder.setShell(true);
commandInfoBuilder.setValue("while true; do echo \"Hey Mesos\"; sleep 5; done");

Protos.TaskInfo task = Protos.TaskInfo.newBuilder()
    .setName("task " + taskId.getValue())
    .setTaskId(taskId)
    .setSlaveId(offer.getSlaveId())
    .addResources(Protos.Resource.newBuilder()
                  .setName("cpus")
                  .setType(Protos.Value.Type.SCALAR)
                  .setScalar(Protos.Value.Scalar.newBuilder().setValue(1)))
    .addResources(Protos.Resource.newBuilder()
                  .setName("mem")
                  .setType(Protos.Value.Type.SCALAR)
                  .setScalar(Protos.Value.Scalar.newBuilder().setValue(128)))
    .setContainer(containerInfoBuilder)
    .setCommand(commandInfoBuilder)
    .build();
```

Why Clojure? clj-mesos

```
(defn docker-task-info
  [uuid {:keys [slave-id]}]
  [{:name "hello-mesos"
    :task-id uuid
    :slave-id slave-id
   :resources {:cpus min-cpu
               :mem min-mem}
   :command {:shell true
             :value "while true; do echo \"Hey Mesos\"; sleep 5; done"}
   :container {:type :docker
               :docker {:image "busybox"}}}])
```

Hello-World Repository

- git clone https://github.com/clj-mesos-workshop/hello-mesos.git

Step 1

Generating a Mesos-Framework

Step 1: Commands

- Try
  - lein new mesos-framework awesome-framework
- Complete version
  - git checkout step-one-generate-from-template

Step 2

Storing State using Zookeeper

Step 2: Commands

- git checkout step-two-add-zookeeper-for-state

Step 3

Recovering from Failure

Step 3: Commands

- Complete version
  - git checkout step-three-recover-from-failure

Step 4

Leader Election

Step 4: Commands

- git checkout step-four-leader-election

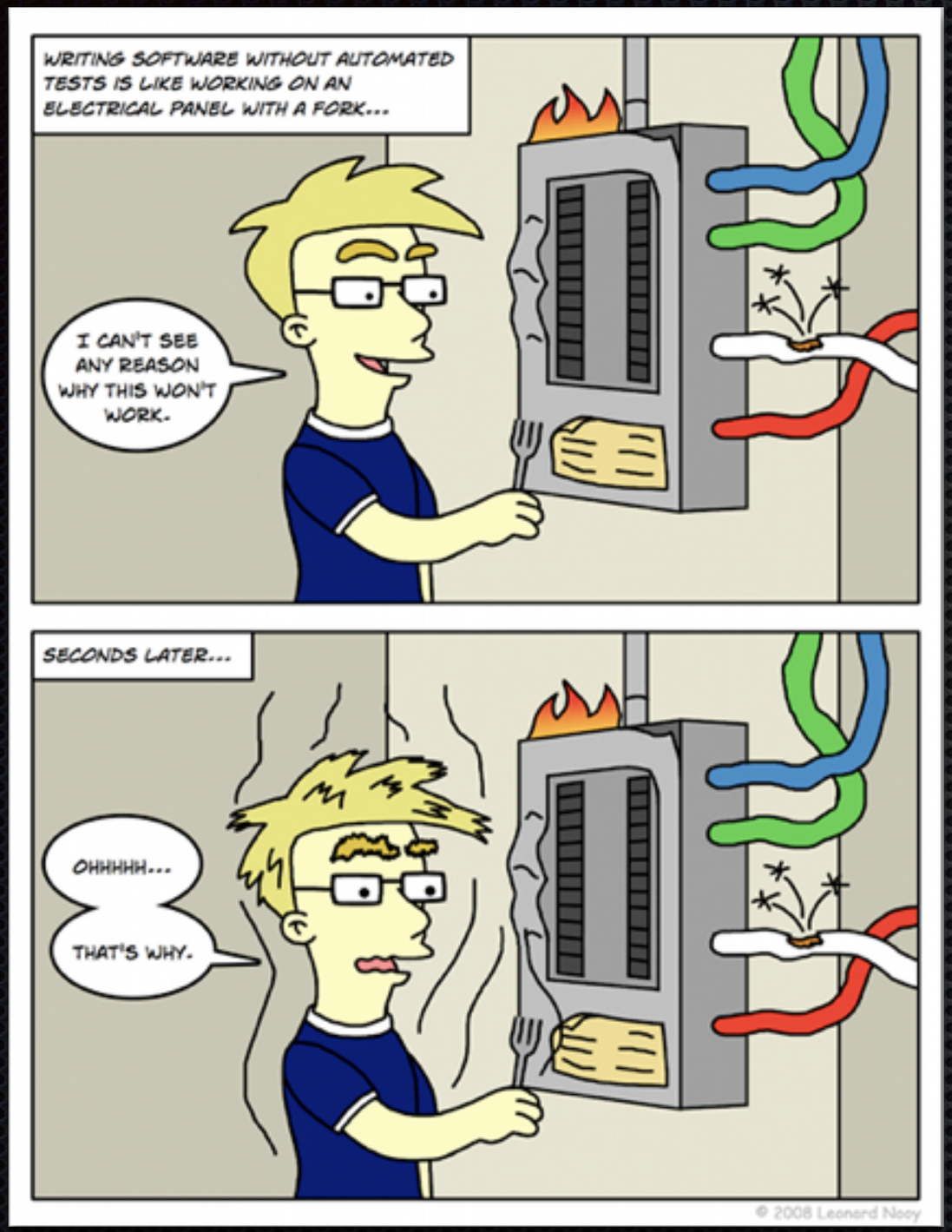
Step 5

Web UI


DOCTYPE HTML
<html>
<head>
<meta name
<meta nam
<meta n
<link
<scri
</hea
<bod

Step 5: Commands

- git checkout step-five-web-ui



Step 6

Testing

Step 6: Commands

- git checkout step-six-testing-the-scheduler

Welcome to StrangeLoop ‘15