Instream SREcon17

Harsh Sharma

SRE @ Linkedin
(Platform & Horizontal)

Agenda

- Requirements
- Possible Solutions
- Instream Design and Architecture
- Implementation with sample code
- Challenges and Learning
- Future Goals

Requirements

Scalable

Scalable Global distribution of products

Flexible

Flexibility of Canary Support

Speed

Quicker version rollbacks and rollouts

Tracking

Real Time Tracking

Possible Solutions and their drawback ?

Generic Deployment

General Deployment model

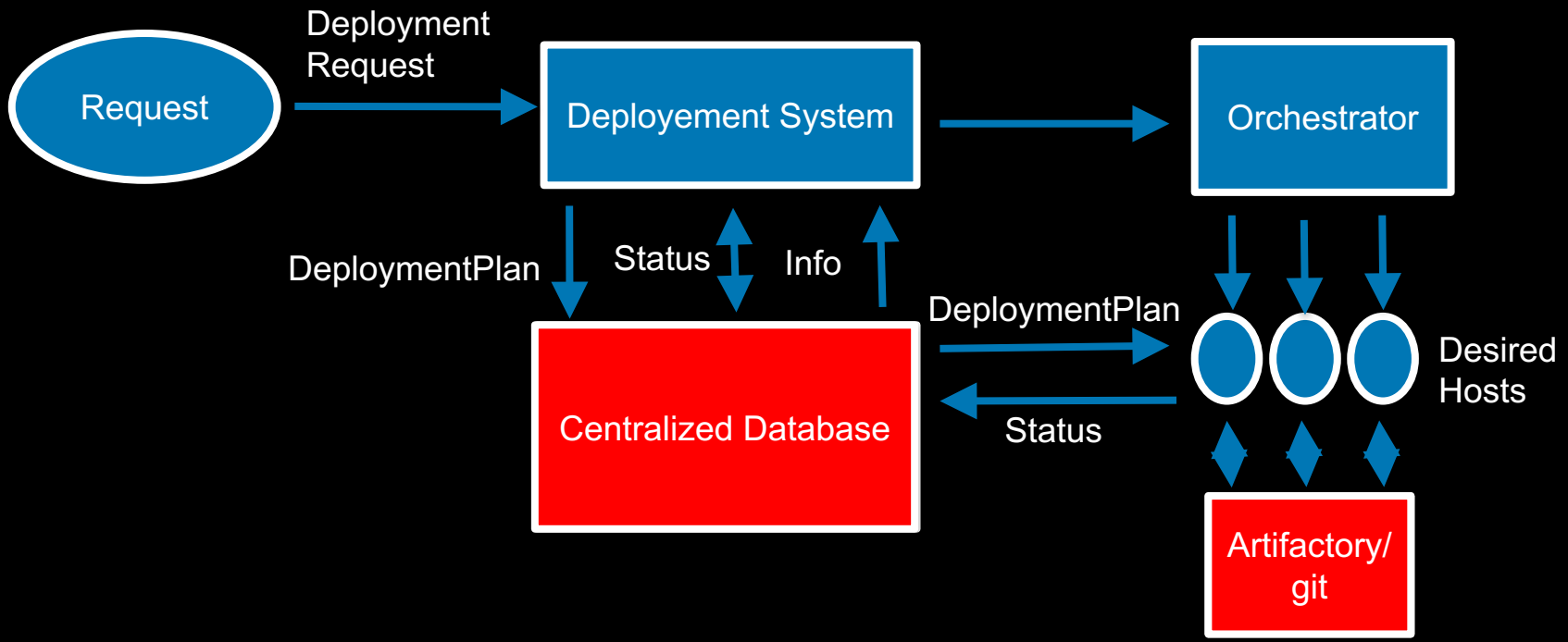
ECL

A mechanism to distribute CLI in Linkedin

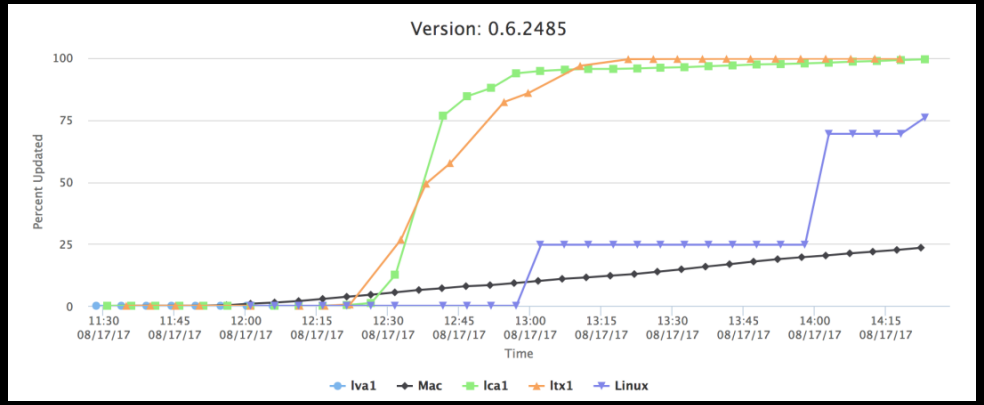
Dist Tree

A tree based distribution model

Generic Deployment System



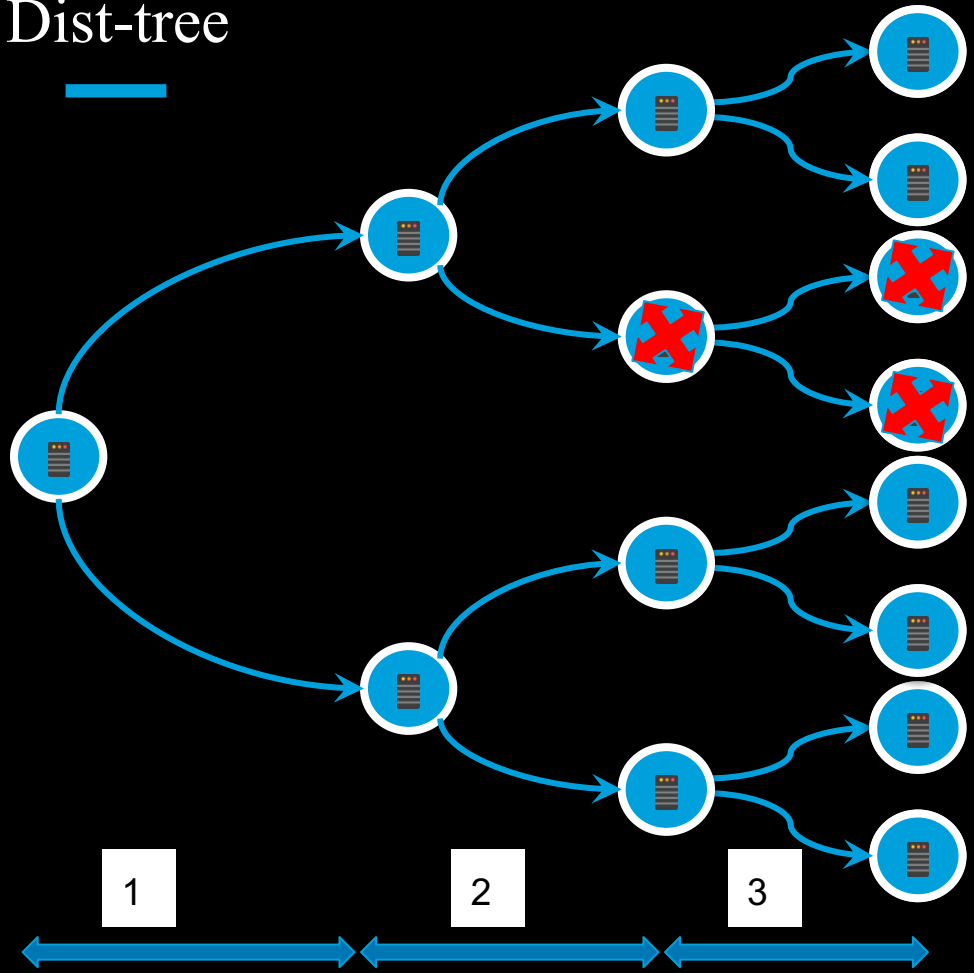
ECL



- No canary
- Global distribution in all fabrics
- Difficult and slow rollback
- No real time updates

Dist-tree

Typical Dist-tree

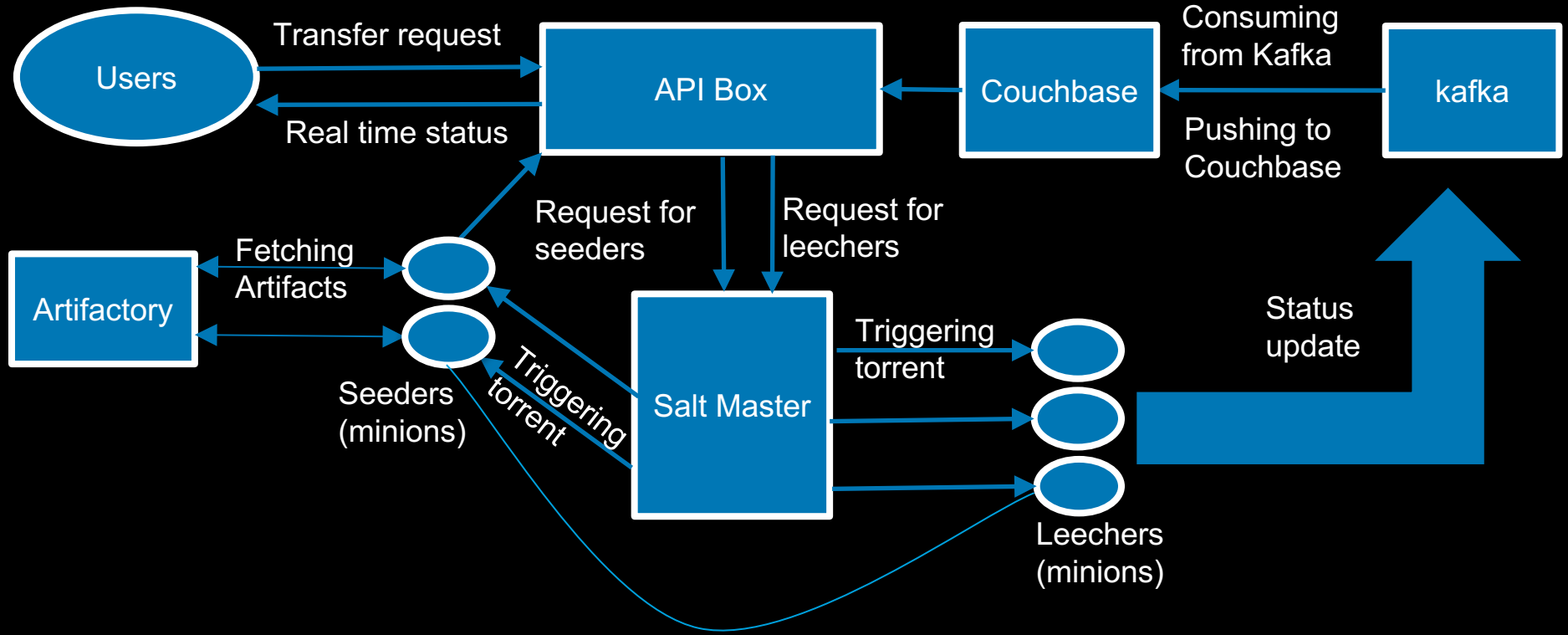


Instream

Bittorent protocol

- Torrent file
- Leechers
- Seeder
- Tracker

Instream Architecture.



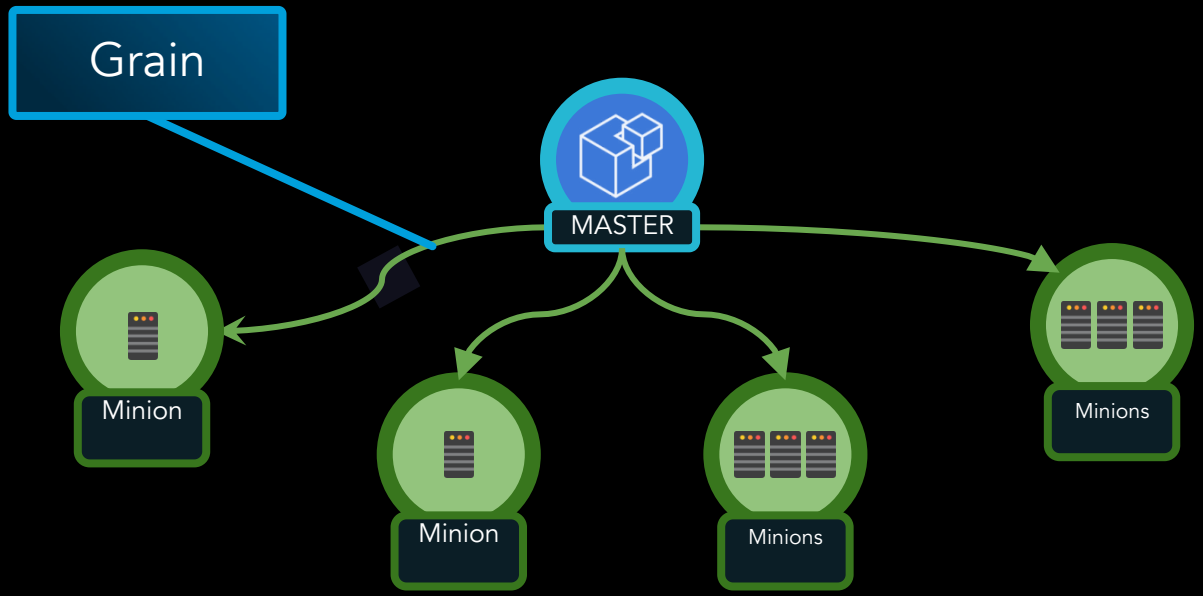
Libtorrent as torrent client

- C++ Library
- Easy to use python bindings

```
import libtorrent as lt
import time

ses = lt.session()
ses.listen_on(6881, 6891)

e = lt.bdecode(open("test.torrent", 'rb').read())
info = lt.torrent_info(e)

params = { 'save_path': '.', \
        'storage_mode': lt.storage_mode_t.storage_mode_sparse, \
        'ti': info }
h = ses.add_torrent(params)

s = h.status()
while (not s.is_seeding):
        s = h.status()

        state_str = ['queued', 'checking', 'downloading metadata', \
                'downloading', 'finished', 'seeding', 'allocating']
        print '%.2f%% complete (down: %.1f kb/s up: %.1f kB/s peers: %d) %s' % \
                (s.progress * 100, s.download_rate / 1000, s.upload_rate / 1000, \
                s.num_peers, state_str[s.state])

        time.sleep(1)
```

Saltstack as remote execution engine

- Triggering salt module from salt master remotely.



Saltstack as remote execution engine

- Rest API to contact salt master
- Targeting through grain

```
salt_payload = {'expr_form': 'compound', 'client': 'local_async', 'fun': 'splay.splay', 'arg': [splay, 'gd.startdropship']}
salt_payload['kwarg'] = {}
salt_payload['tgt'] = 'L@'+','.join(hosts)+' and G@osmajorrelease:6'
saltapi_url = 'https://{0}:{1}'.format(salt_master, salt_master_port)
salt_request = requests.post(saltapi_url, data=json.dumps(salt_payload),headers=SALTAPI_HEADERS, verify=False)
```

Real time status using Kafka and Couchbase

- Real time status reflection through cli.
- Minions send updates to kafka using kafka rest API’s
- Kafka Consumer consume from kafka and push to couchbase.
- Couchbase view for indexing and querying of data.

API and CLI

- CLI example.

```
~ instream -p obhc-agent -v 0.1.34 -f fabric_name -ho '*' -m torrent --variant rhel6
{"status":"submitted","torrentid":"04de2a2a32a49f417f00"}
```

```
~ instream status -t 04de2a2a32a49f417f00 -f fabric_name -m progress
Success Percentage: 22.8093947606
Failure Percentage: 0.0
Pending Percentage: 77.1906052394
```

```
~ instream status -t 04de2a2a32a49f417f00 -f fabric_name -m success
[u'success', u'2017-07-11 21:36:52.364934', u'host-1']
[u'success', u'2017-07-11 21:35:41.707639', u'host-2']
[u'success', u'2017-07-11 21:44:38.317073', u'host-3']
[u'success', u'2017-07-11 21:44:38.317073', u'host-4']
[u'success', u'2017-07-11 21:44:38.317073', u'host-5']
[u'success', u'2017-07-11 21:44:38.317073', u'host-6']
```

Results

- With ECL to distribute some package roughly it takes 3-4 hours
- With this model, per fabric we are distributing in 10 mins

Requirements	Status
Scalable Global distribution of products	Done
Flexibility of Canary Support	Done
Quicker version rollbacks and rollouts	Done
Tracking	Done

Challenges and Learnings

Bandwidth

Bandwidth control in production cluster

Magnet Link

Transferring torrent file to all the minions: Magnet Link

DHT

To use it or not ?

DMZ Boxes

Direct Download Method

Swarm Health

When to stop the torrent ?

Future Goals

Orchestration

- Pluggable with other configuration management tools
- Puppet/CFengine/Ansible

Data Source

- Make data source pluggable.
- Git/Hadoop/File sharing

Deployment

- Make generic global deployment model
- start/stop/restart/status/service check

Thank
you