Hortonworks

Hadoop YARN Services

Xuan Gong
xgong@hortonworks.com

Steve Loughran
stevel@hortonworks.com

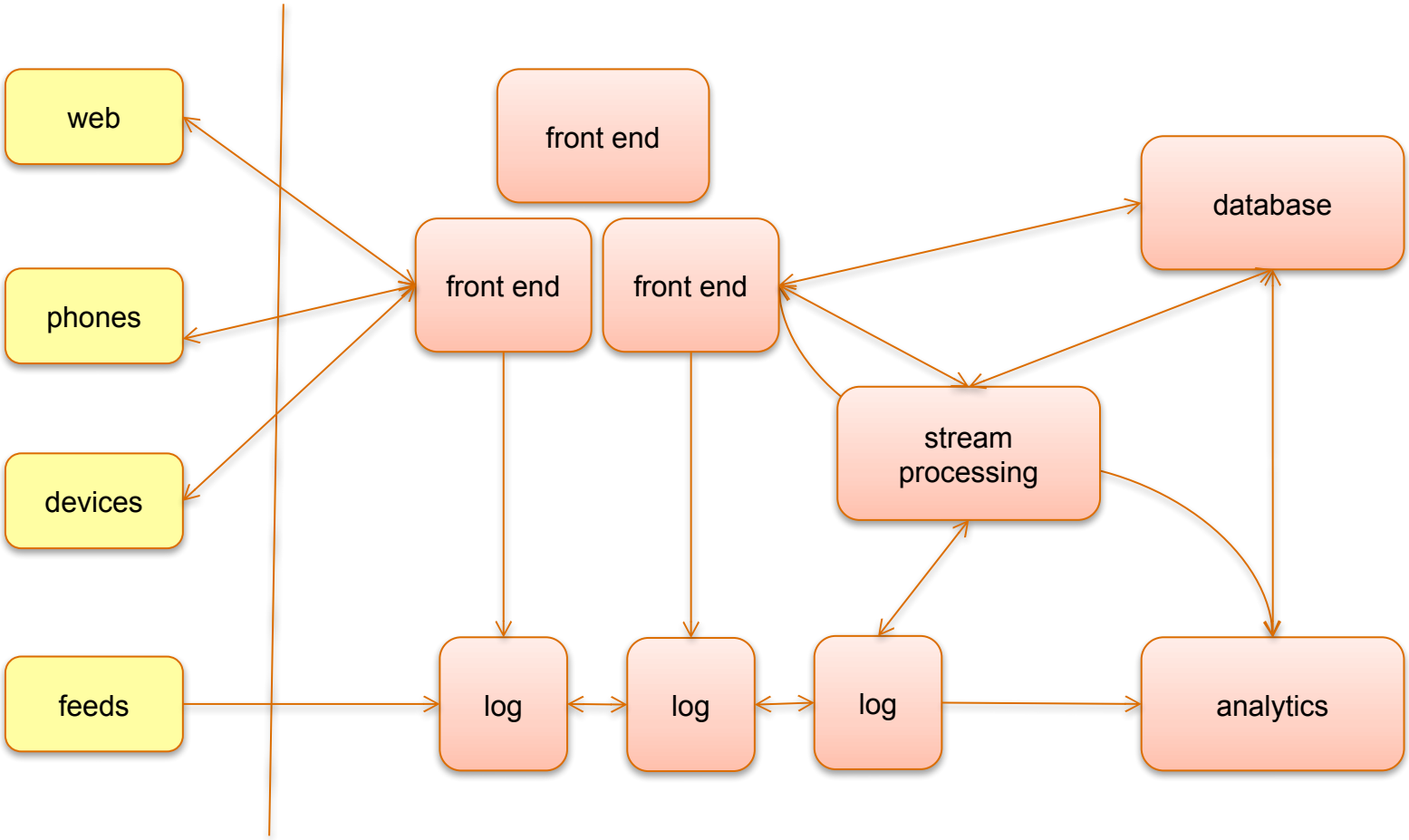
Apache Hadoop + YARN:
An OS for data

An OS can do more than SQL statements

An OS can do more than run admin-installed apps

An OS lets you
run whatever you want!

…which is important



Hortonworks

YARN Services:

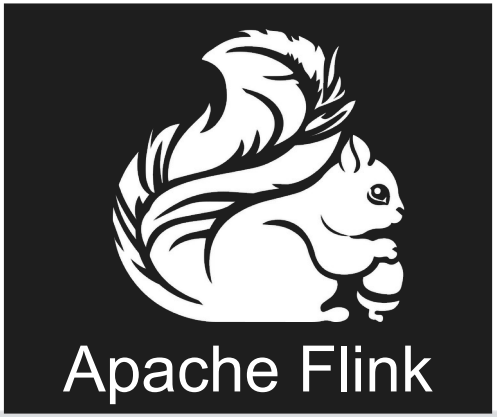
Long lived applications *within a Hadoop cluster*

Samza

Apache Slider (incubating)

(hosting: HBase, Accumulo, Storm…)

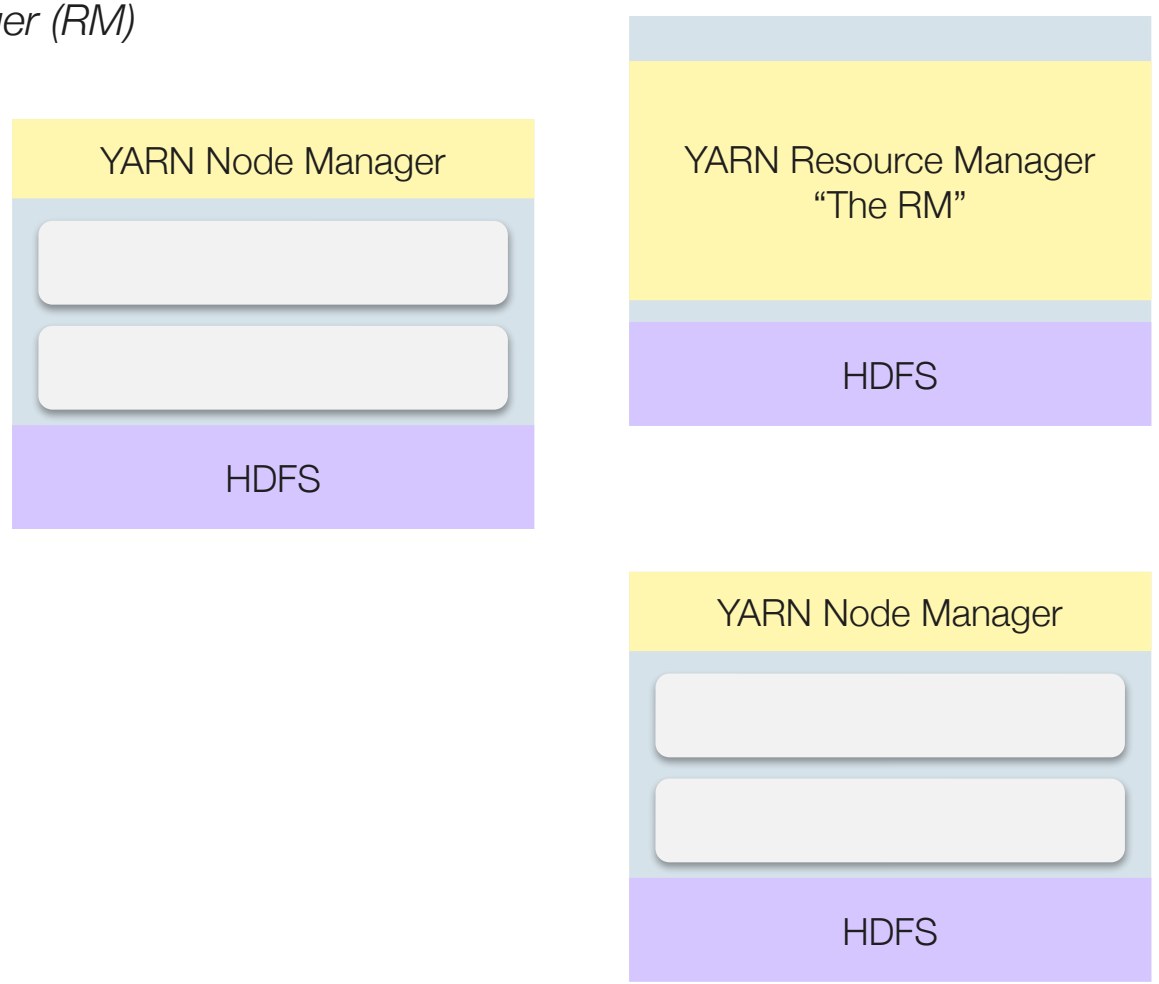
Kafka on YARN



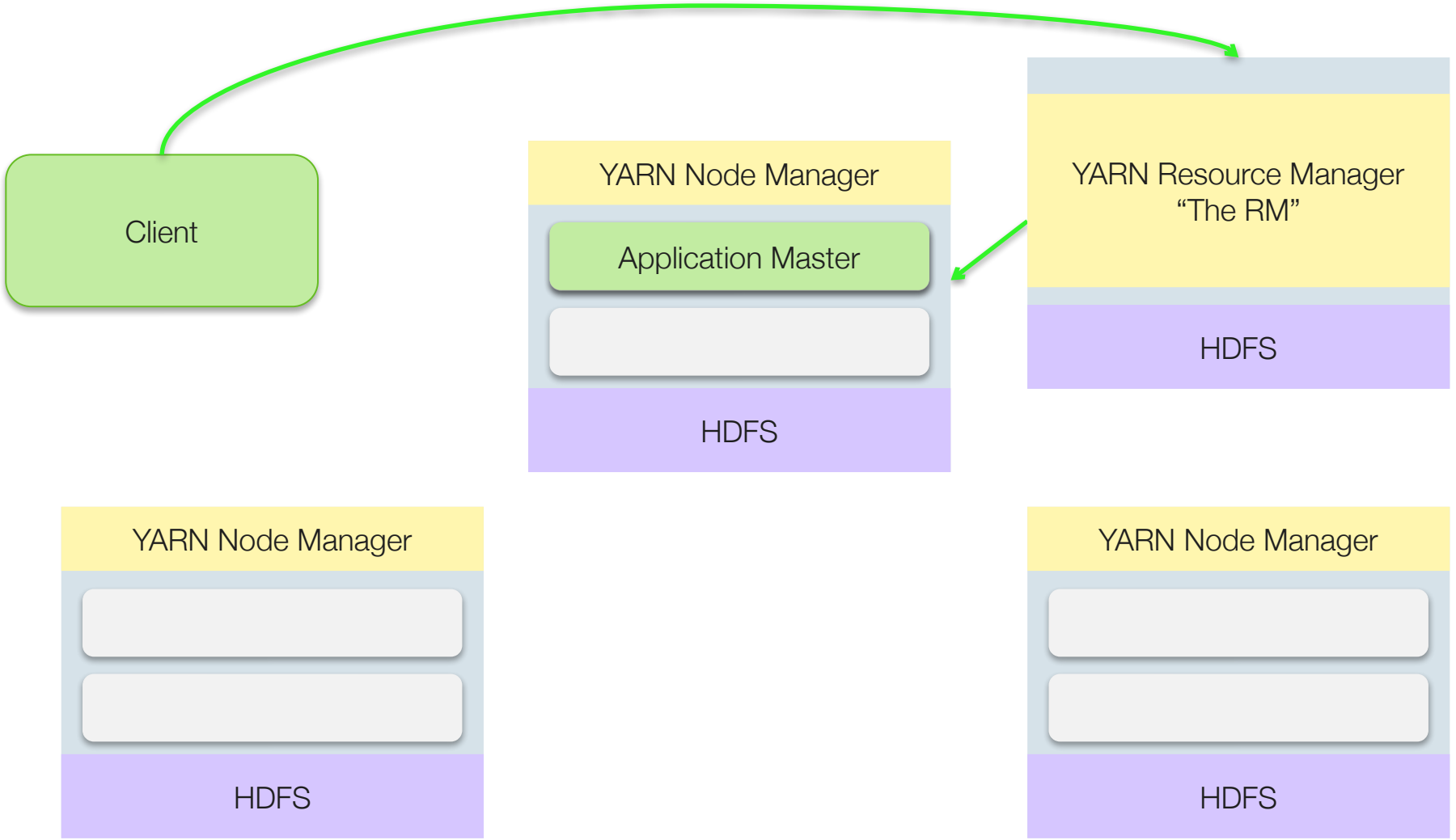
Hive LLAP Daemons

Background: YARN

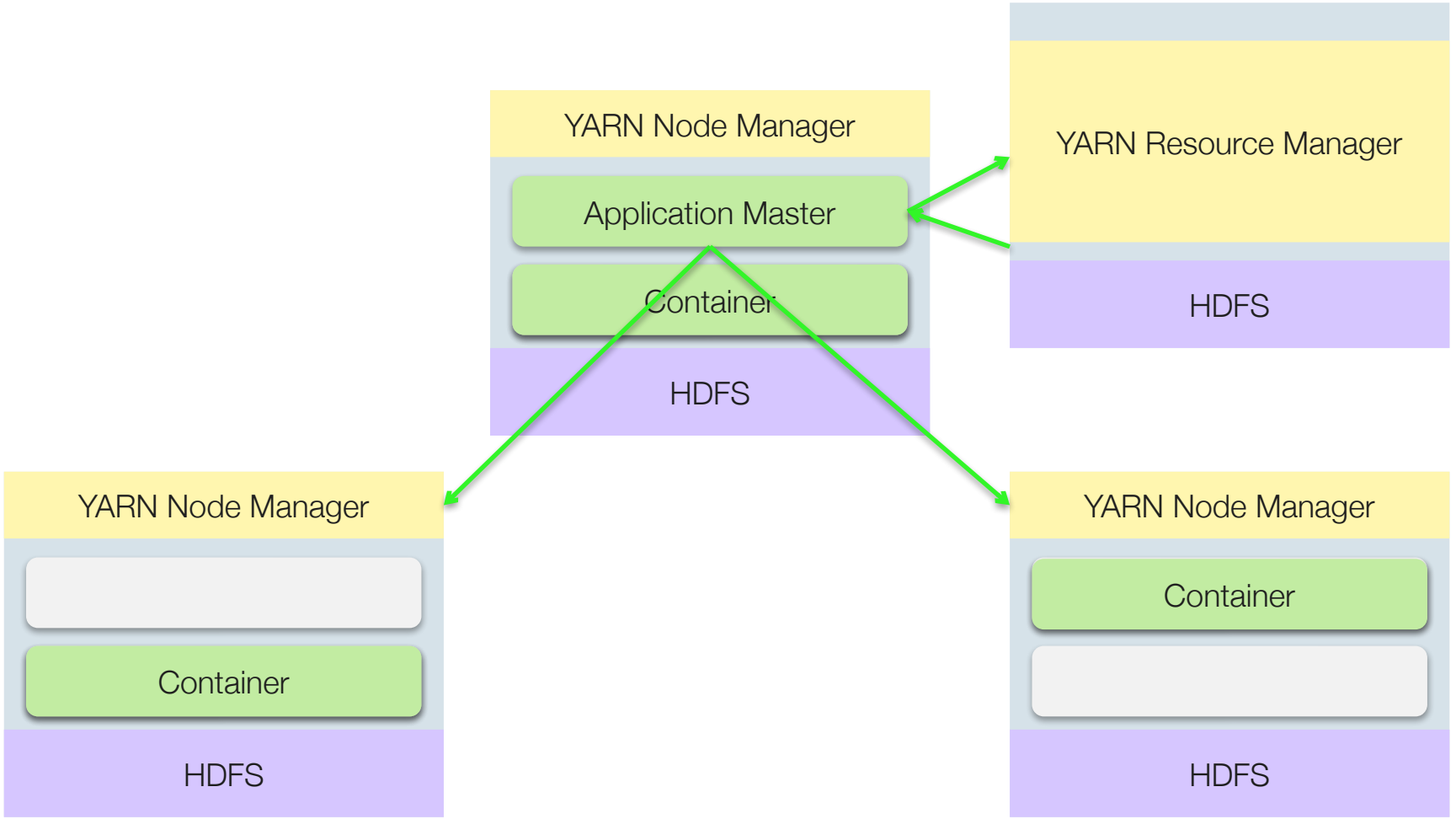
- Servers run YARN *Node Managers (NM)*
- NM's heartbeat to *Resource Manager (RM)*
- RM schedules work over cluster
- RM allocates containers to apps
- NMs start containers
- NMs report container health



Client creates App Master



“AM” requests containers



Short lived apps have it easy

- **failure**: clean restart
- **logs**: collect at end
- **placement**: by data
- **security**: Kerberos delegation tokens
- **discovery**: launcher app can track

Long-lived services don't

- **failure**: stay up
- **logs**: ongoing collection
- **placement**: availability, performance
- **security**: stay secure over time
- **discovery**: locatable by any client

YARN-896

Support for YARN services

YARN-896

Log aggregation

Kerberos token renewal

Gang scheduling

Service registration & discovery

Net & Disk resources

Windowed failure tracking

REST

Container reuse

Container resource flexing

Anti-affinity placement

Labelled nodes & queues

Container signalling

Applications to continue over AM restart

Hadoop 2.6

Log aggregation

Kerberos token renewal

(Docker)

Gang scheduling

Service registration & discovery

Net & Disk resources

Windowed failure tracking

REST

Container reuse

Container resource flexing

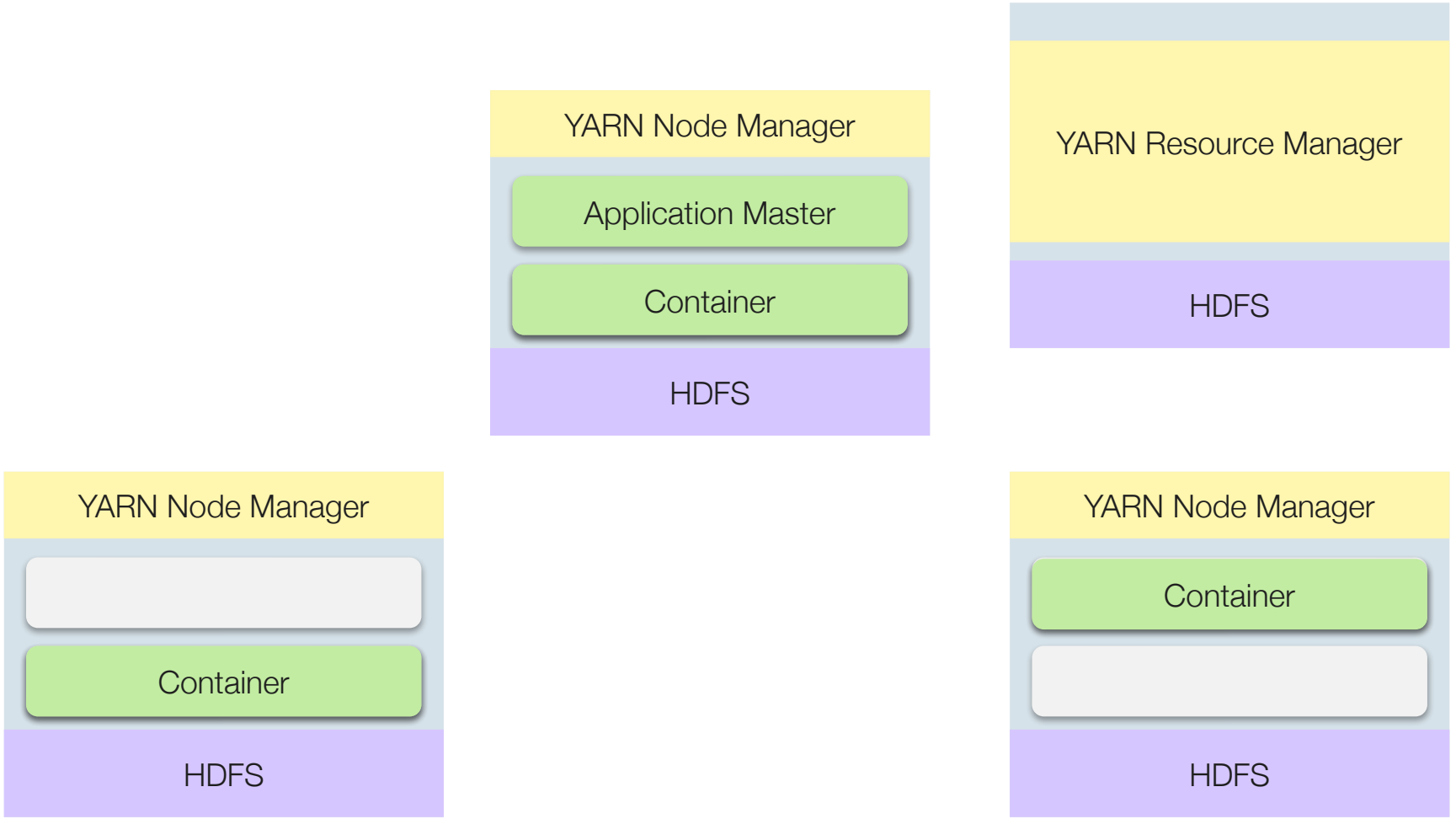
Anti-affinity placement

Labelled nodes & queues

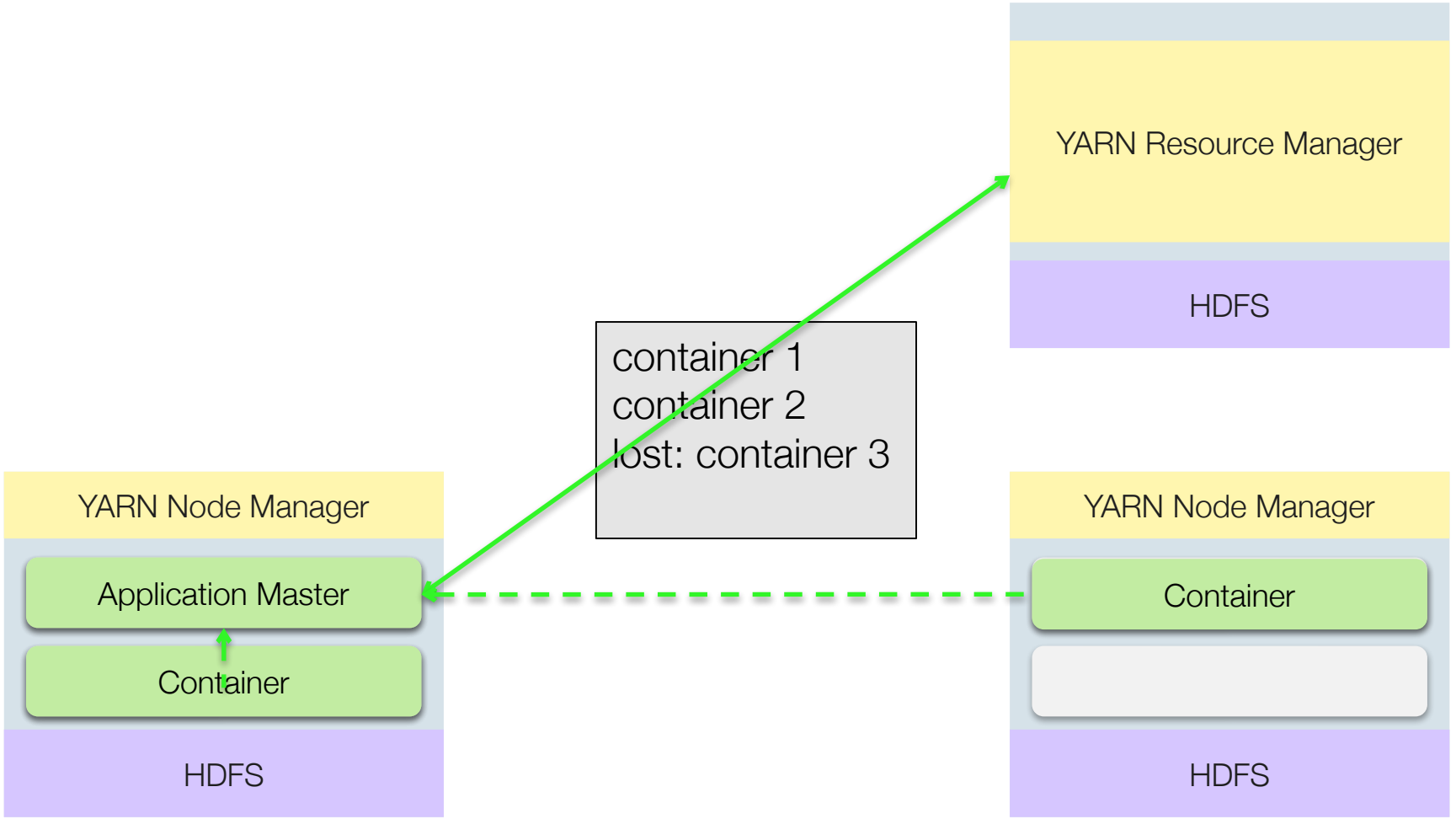
Container signalling

Applications to continue over AM restart

Failures



Failures



Easy: enabling

```
// Client
 amLauncher.setKeepContainersOverRestarts(true);
 amLauncher.setMaxAppAttempts(8);


// Server
List<Container> liveContainers =
  amRegistrationData.getContainersFromPreviousAttempts();
```

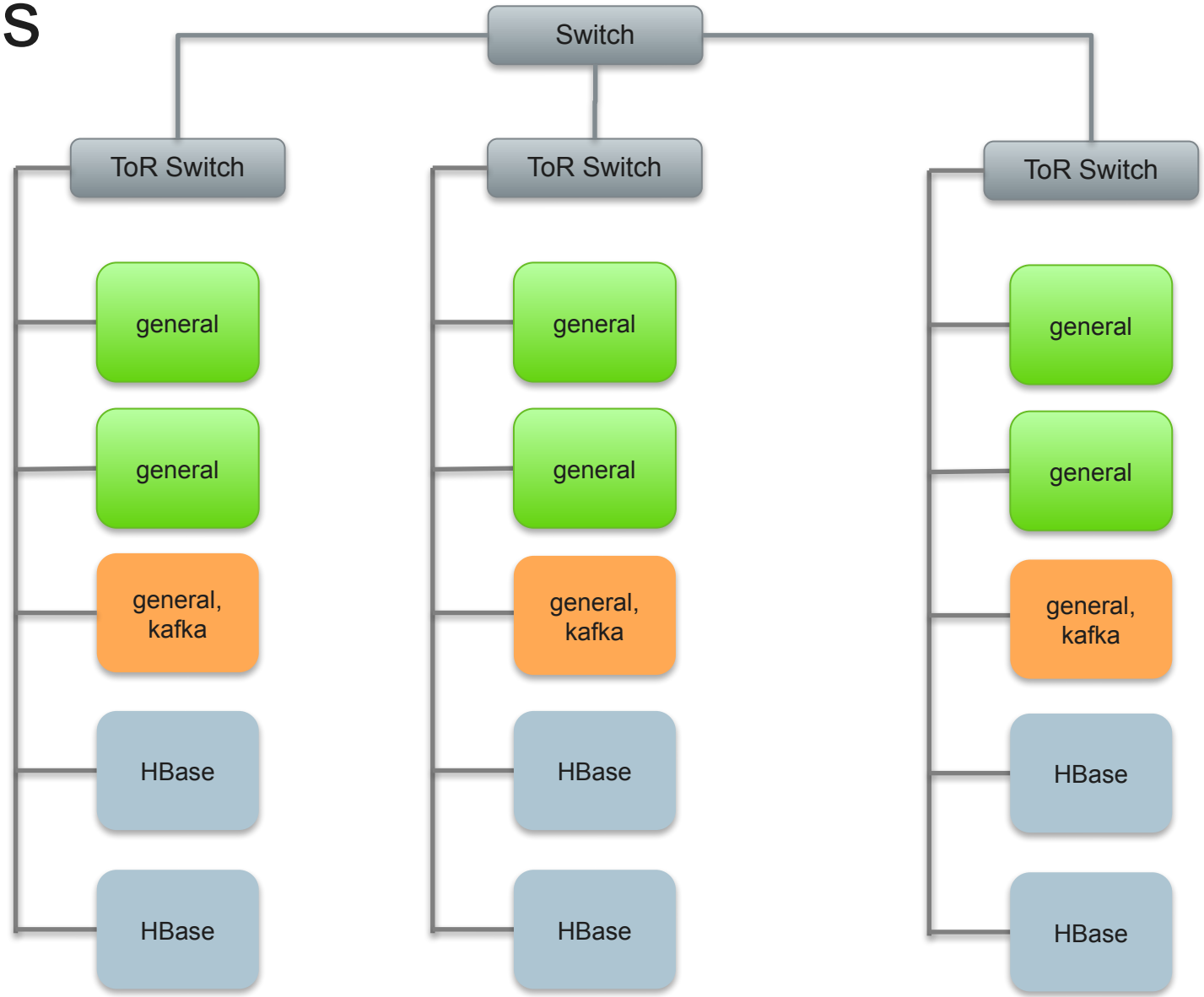
Harder: rebuilding state

Persisted	Rebuilt	Transient
Specification	Node Map	Event History
Placement History	Component Map	Container Queues

Log Aggregation

```
<property>
  <name>yarn.log-aggregation-enable</name>
  <value>true</value>
</property>
```

Labels


Switch
ToR Switch
ToR Switch
ToR Switch
general
general
general, kafka
HBase
HBase
general
general
general, kafka
HBase
HBase
general
general
general, kafka
HBase
HBase

Labels

```
$ yarn rmadmin
...
  -addToClusterNodeLabels [label1,label2,label3]
-removeFromClusterNodeLabels [label1,label2,label3]
  -replaceLabelsOnNode [node1:port,label1,label2]
          -directlyAccessNodeLabelStore
```

YARN-913: Service Registry

```
$ slider resolve –path ~/services/org-apache-slider/storm1

{ "type" : "JSONServiceRecord",
  "external" : [ {
    "api" : "http://",
    "addressType" : "uri",
    "protocolType" : "webui",
    "addresses" : [ {
      "uri" : "http://nn.ex.net:4813"
    } ]
  }, {
    "api" : "classpath:org.apache.slider.publisher.configurations",
    "addressType" : "uri",
    "protocolType" : "REST",
    "addresses" : [ {
      "uri" : "http://nn.ex.net:4813/ws/v1/slider/publisher/slider"
    }]
  } } ] }
```

Internal and external endpoints

```
"internal" : [ {
    "api" : "classpath:org.apache.slider.agents.secure",
    "addressType" : "uri",
    "protocolType" : "REST",
    "addresses" : [ {
      "uri" : "https://nn.ex.net:4813/ws/v1/slider/agents"
    } ]
  } ]
```

Internal: for an application's own use.
External: for clients, Web UIs and other apps

Security

- Token expiry a core Kerberos feature
- Token expiry inimical to service longevity
- Specifically: token delegation
- After 72h (default)

YARN updates the RM/AM tokens but not HDFS, ZK, ….

How do apps cope?

Do nothing → apps can run up to 72h
–*All*

Keytabs → apps can run forever; keytabs need to be managed (securely)
–*Slider*

Client push → running/scheduled client updates AM; AM forwards to containers
–*Twill*

AM keytab → containers ask for new tokens
–*Spark via SPARK-5342*

…so you can now:

write long lived apps

…with failure resilience

…and centralised log viewing

…and labelled/isolated placement

…in secure clusters

TODO

Log aggregation

Kerberos token renewal

Gang scheduling

Service registration & discovery

Net & Disk resources

Windowed failure tracking

REST

Container reuse

Container resource flexing

Anti-affinity placement

Labelled nodes & queues

Container signalling

Applications to continue over AM restart

Questions?

For some code, see
http://slider.incubator.apache.org/