Bringing Hadoop to Fedora

Putting the elephant in the room

Presented by
Pete MacKinnon & Rob Rati
Big Data/MRG
Red Hat Inc.

Today's Topics

1. Hadoop background
2. Fedora Java packaging
3. Hadoop packaging
4. Future
5. Questions

Hadoop Background

Big Data

- Data storage and analytics at web scale
  - Web page indexing
  - Social media content
  - Consumer data
- Exabytes of data
  - 1 exabyte == 1000 petabytes == 10^{17} kb
  - IDC estimates ~40-50 exabytes of digital universe content by 2020
- Traditional RDBMS infrastructure can't cut it
- New technology and techniques required...

The Premise

- Flip conventional data computing
- OLD
  - Centralized RDBMS servers stuffed with as much RAM and disk as they can handle
- NEW
  - De-centralized and distributed
  - Data storage is replicated in blocks across commodity hardware
  - The algorithmic tasks (i.e., MapReduce) are sent out to the data *where it lives*
  - Tasks are tracked and results collected and distilled

Brief history

- 2003
  - Google research papers describing distributed file system and map/reduce algorithm
- 2004
  - Doug Cutting starts developing Nutch in Java
- 2006
  - Cutting joins Yahoo! where research cluster is formed
  - 1.9 terabyte sort on 188 nodes in 47 hours
- 2008
  - Hadoop becomes top-level Apache project

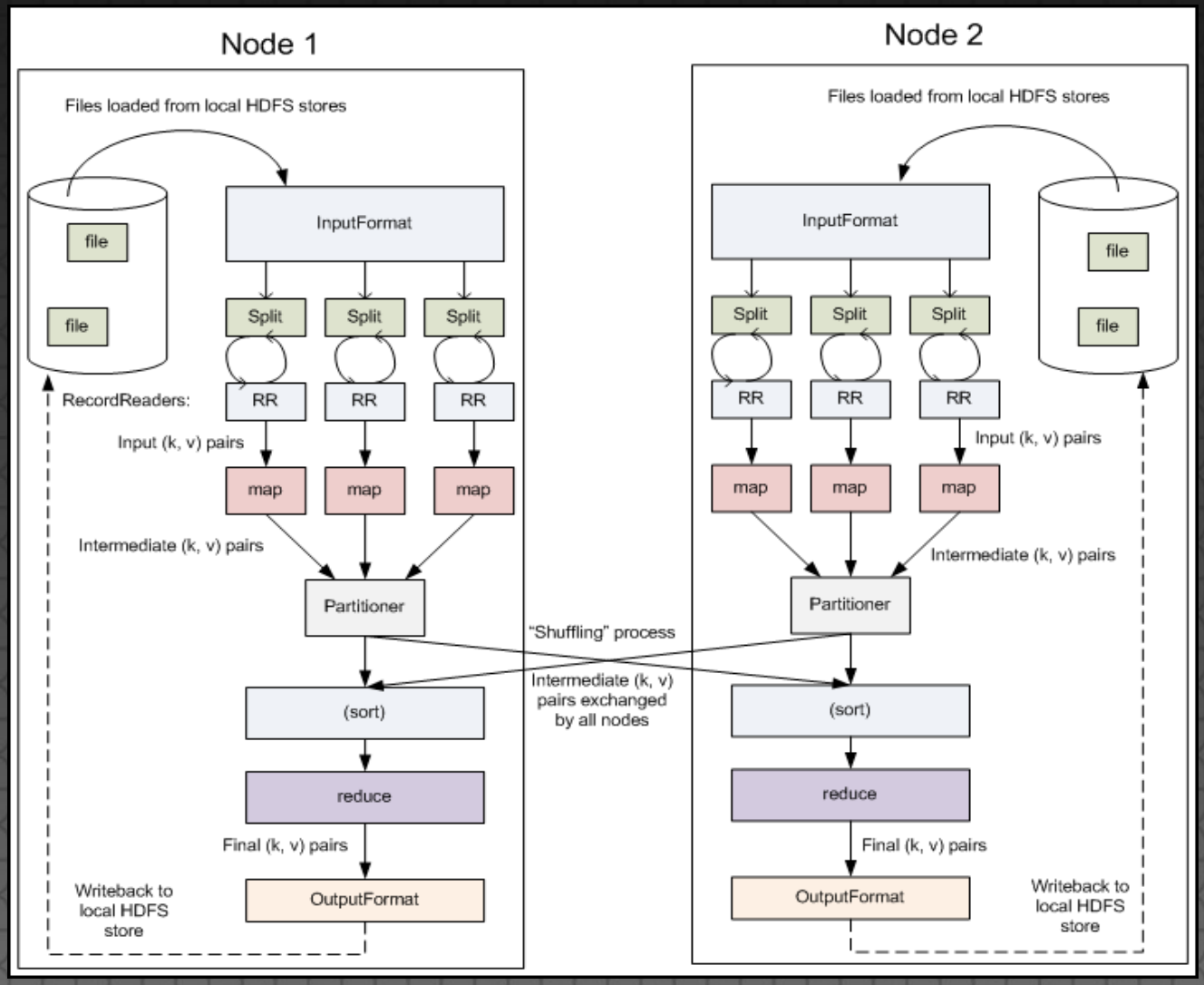
Hadoop HDFS basics

- Single NameNode
- Multiple DataNodes attached to a NameNode
- Data is stored in blocks (typically 128Mb) that are replicated across DataNodes (servers, racks)
- NameNode stores file metadata for clients
- DataNode stores a checksum for every 512 bytes
- If checksum validation fails then framework moves onto other replicas

Hadoop MR basics

- Essentially a query application of the Hadoop framework
  - A client of HDFS
- Parallelization, fault tolerance, data distribution, load balancing
  - all responsibilities of Hadoop framework
  - programmer provides the map algorithm and the reduce algorithm
- Streaming and pipelining of data
  - MR implementation can be C++ also
- Counters, sorts, joins, etc.

MR algorithm



fedora

Hadoop map

```
public static class Map
    extends Mapper<LongWritable, Text, Text, IntWritable> {
     private final static IntWritable one = new IntWritable(1);
     private Text word = new Text();

     public void map(LongWritable key, Text value, Context context)
       throws IOException, InterruptedException {
         String line = value.toString();
         StringTokenizer tokenizer = new StringTokenizer(line);
         while (tokenizer.hasMoreTokens()) {
             word.set(tokenizer.nextToken());
             context.write(word, one);
         }
     }
```

Hadoop reduce

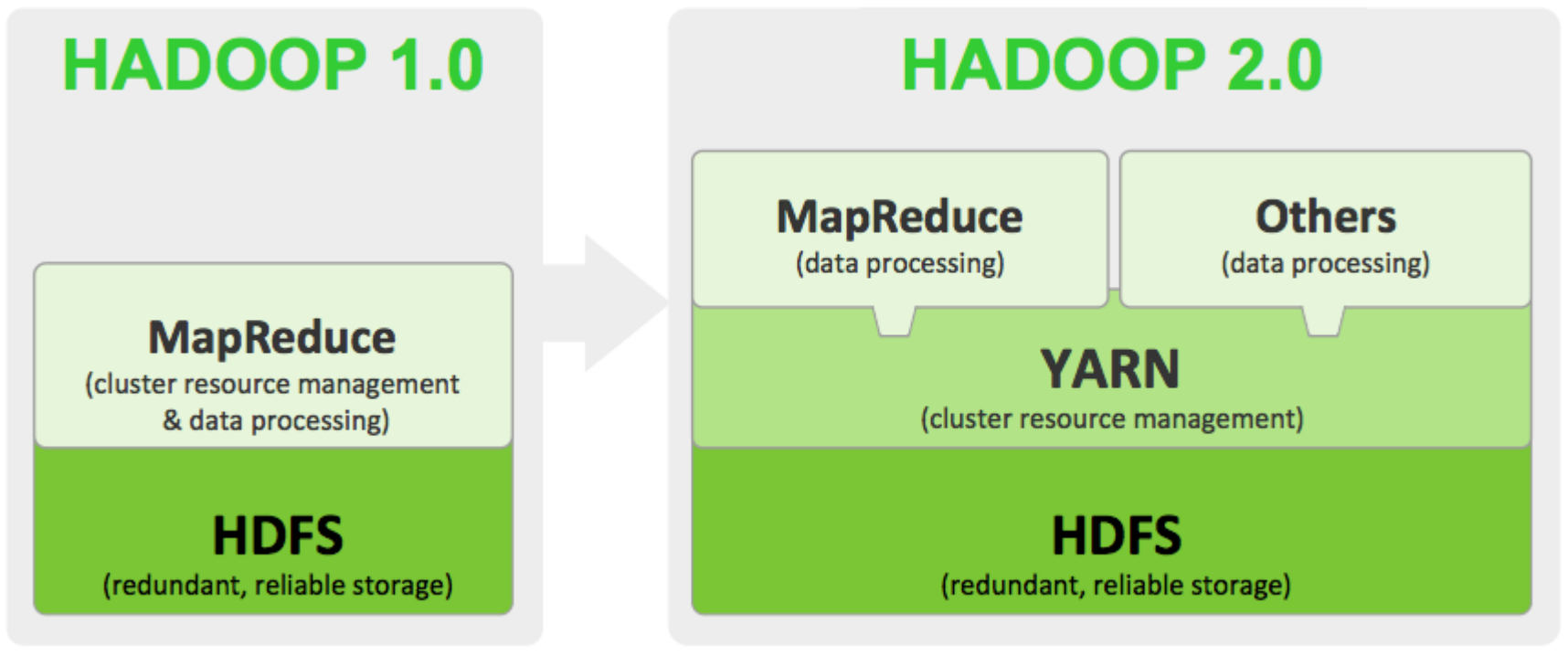
```
 public static class Reduce
    extends Reducer<Text, IntWritable, Text, IntWritable> {


    public void reduce(Text key, Iterable<IntWritable> values, Context
context)
      throws IOException, InterruptedException {
        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }
        context.write(key, new IntWritable(sum));
    }
 }
```

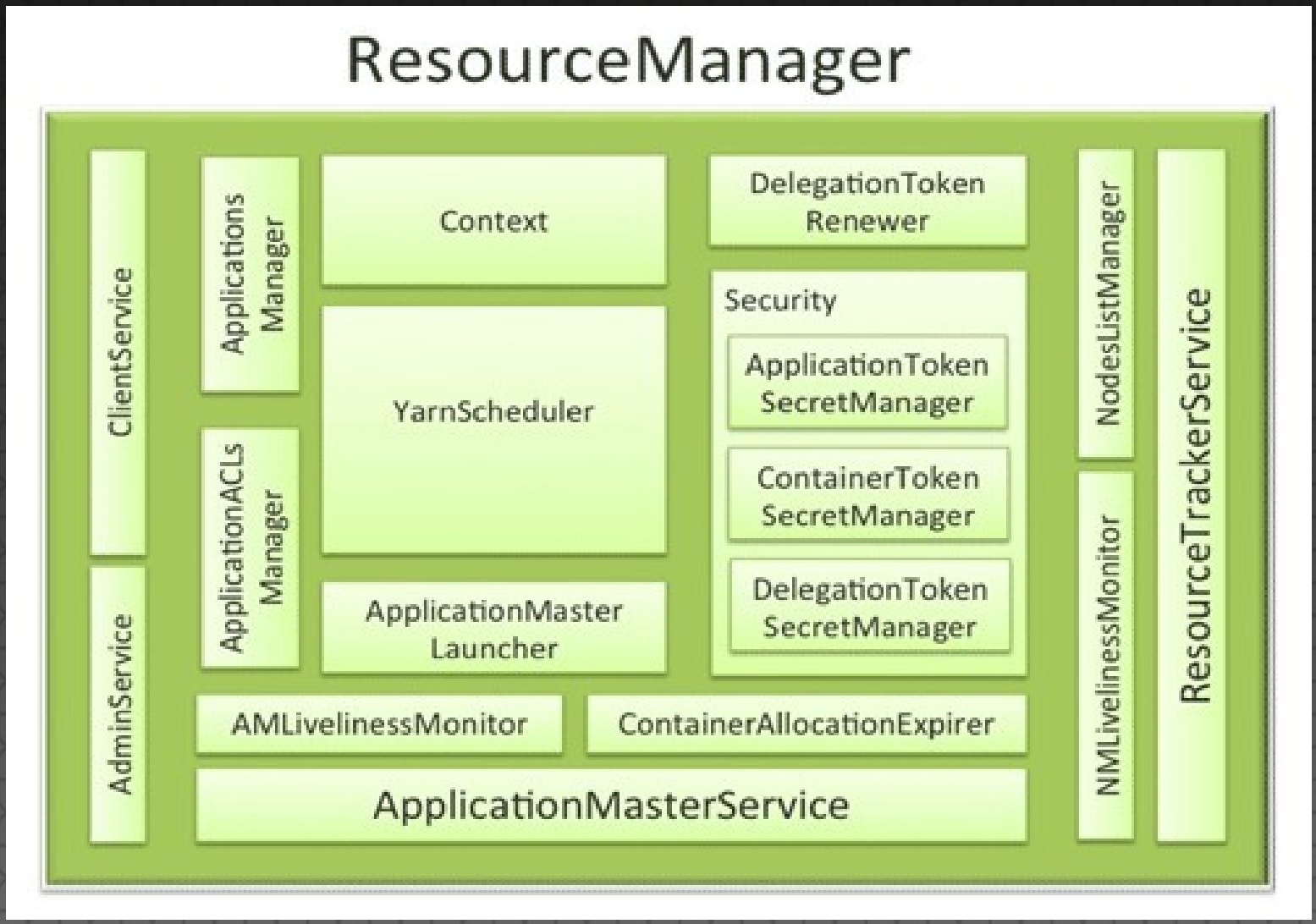
Hadoop MR driver

```
public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = new Job(conf, "wordcount");
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    job.setMapperClass(Map.class);
    job.setReducerClass(Reduce.class);
    job.setInputFormatClass(TextInputFormat.class);
    job.setOutputFormatClass(TextOutputFormat.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    job.waitForCompletion(true);
}
```

Hadoop evolution



YARN


ResourceManager
ClientService
AdminService
Applications Manager
ApplicationACLs Manager
Context
YarnScheduler
ApplicationMaster Launcher
DelegationToken Renewer
Security
ApplicationToken SecretManager
ContainerToken SecretManager
DelegationToken SecretManager
NodesListManager
NMLivelinessMonitor
ResourceTrackerService
AMLivelinessMonitor
ContainerAllocationExpirer
ApplicationMasterService

Ecosystem

- **Hadoop Common**: HDFS, MapReduce
- **Hive:** data warehouse system providing SQL-like language
- **HBase**: distributed, column-oriented store based on Google Bigtable
- **Mahout**: machine learning library built atop MR
- **Ambari**: provision, monitor, and manage Hadoop clusters
- And much, much more...

Fedora Java Packaging

Maven

- Hadoop is a Maven-based project
  - Build-time dependency management and compilation tool
  - “Makefiles” are written in XML; pseudo-hierarchical
  - Extensible for plugins
  - Pulls dependencies (typically Java jar files) from well-known centralized repositories to populate a local repo
  - Very popular in Apache and Java community at large

Packaging throwdown

- Fedora
  - *“Thou shalt have only one platform version, ideally the most recent”*
  - Software dependencies are *expected* to be compatible
- Maven
  - *“Use whatever version you want, wherever you want in your project”*
  - Jars live on in perpetuity in the Maven central repositories
  - Guides dependency compatibility but doesn't guarantee it

Tooling

- Java team has developed Maven bridge tools
- Pre-Fedora 19 (patched mvn)
  - mvn-rpmbuild: resolve jars for Fedora system repo, not Maven central repositories
  - mvn-local: try Fedora first, then fall back
- Fedora 19 and beyond (xmvn)
  - extensions with Fedora-aware spec macros
  - simplifies spec for Java Maven projects
  - mvn-build: syntactic sugar with commonly used “install” phase options
  - classpath building tools for symlinks, etc.

Mapping

- Maven POM
- Fedora

<groupId> → /usr/share/java/*group*

<artifactId> → /usr/share/java/*group*/*artifact*

<version> → *ignored* (latest for that Fedora release)

fedora™

Hadoop Packaging

Strategy

- Started with 2.0.2-alpha, now 2.0.5-alpha
- Two objectives
  - Integrated patch set that could be applied from a spec
  - Series of defined patch groups that could be “atomically” offered up to upstream
- Tracking branches set up publicly at http://github.com/fedora-bigdata
- Maintain a topic branch from the integration for testing and parity with the upstream baseline
  - Some modifications are “non-upstreamable”

Adaptation

- 73 extrinsic dependencies in upstream Hadoop
- Not everything needs adjustment
  - Fedora Maven tooling resolves deps from local install for you, regardless of version
  - No API or serious breakage? You're in business!
- Tier 4: dep packages that are available and compatible
- Tier 3: explicit POM decl; groupId/artifactId adjustment
- Tier 2: source code changes for different API or behavior
- Tier 1: entire dependency is missing and needs packaging

Tiers

- Tier 4 examples
  - commons-io, commons logging, asm, etc.
- Tier 3 example
  - <groupId>tomcat</groupId> .....
  - <groupId>org.apache.tomcat</groupId>
- Tier 1
  - bookkeeper
  - zookeeper
  - jspc-maven-plugin
  - maven-native

Tier 2 in more detail

- Jetty 6.1.z to 9.0.z
  - Jetty 6 servlet engine could host Tomcat 5.5 Jasper (JSP) compilations
  - Not true for Tomcat 7 inside Jetty 9
    - *NoClassDefFoundError: org/apache/tomcat/InstanceManager*
    - Needed to swap out Jasper in favor of Glassfish JSP compiler implementation
  - SSL API changes for Jetty
- JUnit tests
  - some tests tied to API and error message content
  - assert logic overhaul

fedora™

Workflow

- Maven build is sequential
  - compile, install, test phases

fedora

Status

- Dependencies available in Fedora (missing since project initiation): 100%
- Adaptation of Hadoop 2.0.5a source via patches: 100%
- Hadoop spec completion: 100% (httpfs disabled)
- 2 of 10 offered patch sets accepted by Apache
- Will rebase to Hadoop 2.1 (2.2?) official release when available later this year
  - May stick with 2.0.5a due to F20 cut-off or just increment tarball
- Tests
  - 5545 pass, 7 fail, 16 errors, 27 skipped

Future

Software Collections

- Fedora Software Collections
  - Concurrently install multiple versions of the same RPM packages on your system
  - Allow you to build a conventional package and a Software Collection package from a single spec file
  - SC namespace ensures no collisions with incumbent/existing packages
  - However...
    - Java spec files would get very messy
    - *Unapproved* for official packages

fedora

Java Oversight

- Perhaps a Fedora Java dependency czar is the solution
- Someone who decides the appropriate API levels for a given release
- Identify and protect the “strategic” Java projects
- Abolish the package democracy in the Java space

fedora

Ring Theory

- Matthew Miller talk tomorrow
  - “*An Architecture for a More Agile Fedora*”
- Traditional Fedora space in a lower/inner ring
  - Kernel, Yum, RPM, Python 2, etc.
  - Canonical repository
- Higher rings
  - More freedom for SIGs to maneuver
  - Custom package systems? Maven?
  - SIG repositories
- How do the rings associate?
  - Is Maven fundamental to base or only SIG?

fedora™

Summary

- On track to have Hadoop 2 in Fedora 20
- The maintenance road ahead is a long one
  - also applies to ecosystem constituents as they come into the fold
  - hadoop-common gives a base
- Fedora needs to evolve to satisfy
  - admins (stability) AND
  - developers (agility)

Questions?

Contact:
bigdata@lists.fedoraproject.org