Detailed Role Assignments

- Very Small (Up to 10 workers, No High Availability)
  - 1 Master Node
    - NameNode, YARN Resource Manager, Job History Server, ZooKeeper, Impala StateStore
  - 1 Utility/Edge Node
    - Secondary NameNode, Cloudera Manager, Hive Metastore, HiveServer2, Impala Catalog, Hue, Oozie, Flume, Relational Database, Gateway configurations
  - 3-10 Worker Nodes
    - DataNode, NodeManager, Impalad, Llama

Detailed Role Assignments

- Small (Up to 20 workers, High Availability)
  - 2 Master Nodes
    - NameNode (with JournalNode and FailoverController), YARN Resource Manager, ZooKeeper
    - (1 Node each) Job History Server, Impala StateStore
  - 1 Utility/Edge Node
    - Cloudera Manager, Hive Metastore, HiveServer2, Impala Catalog, Hue, Oozie, Flume, Relational Database, Gateway configurations
    - (requires dedicated spindle) Zookeeper, JournalNode
  - 3-20 Worker Nodes
    - DataNode, NodeManager, Impalad, Llama

Detailed Role Assignments

- Medium (Up to 200 workers, High Availability)
  - 3 Master Nodes
    - (3 Nodes) Zookeeper, JournalNode
    - (2 Nodes each) NameNode (with FailoverController), YARN Resource Manager
    - (1 Node each) Job History Server, Impala StateStore
  - 2 Utility Nodes
    - Node 1: Cloudera Manager, Relational Database
    - Node 2: CM Management Service, Hive Metastore, Catalog Server, Oozie
  - 1+Edge Noded
    - Hue, HiveServer2, Flume, Gateway configuration
  - 50-200 Worker Nodes
    - DataNode, NodeManager, Impalad, Llama

Detailed Role Assignments

- Large (Up to 500 workers, High Availability)
  - 5 Master Nodes
    - (5 Nodes) Zookeeper, JournalNode
    - (2 Nodes) NameNode (with FailoverController)
    - (2 different Nodes) YARN Resource Manager
    - (1 Node each) Job History Server, Impala StateStore
  - 2 Utility Nodes
    - Node 1: Cloudera Manager, Relational Database
    - Node 2: CM Management Service, Hive Metastore, Catalog Server, Oozie
  - 1+Edge Noded
    - Hue, HiveServer2, Flume, Gateway configuration
  - 200-500 Worker Nodes
    - DataNode, NodeManager, Impalad, Llama

Memory Allocation

Item	RAM Allocated
Operating System Overhead	2 GB (minimum)
DataNode	1-4 GB
YARN NodeManager	1 GB
YARN ApplicationManager	1 GB
YARN Map/Reduce Containers	1-2GB/Container
HBase RegionServer	4-12 GB
Impala	128 GB (can be reduced with spill-to-disk)

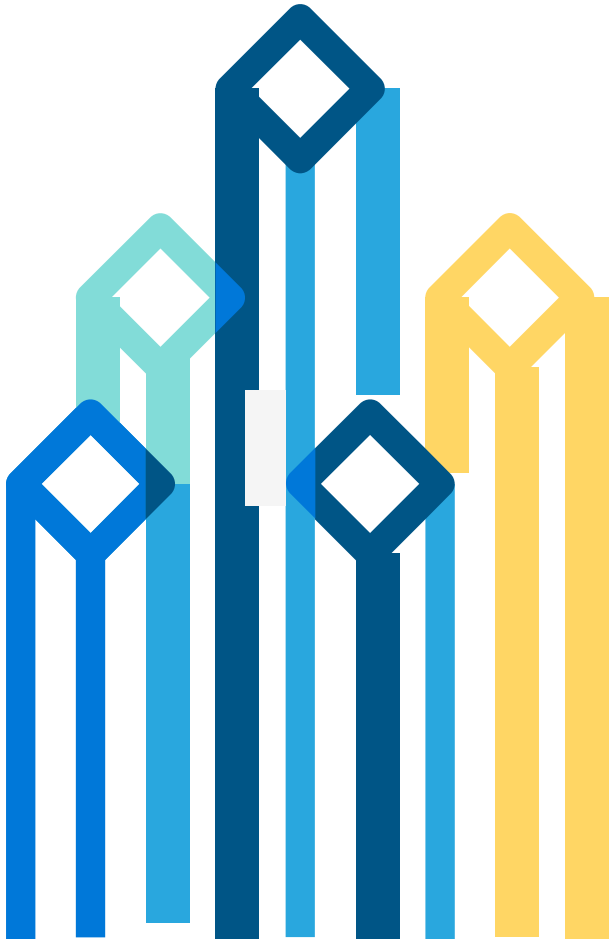
cloudera

Questions?

cloudera

Apache Hadoop Operations for Production Systems: Configuration

Philip Zeyliger



Agenda

- Mechanics
- Key configurations
- Resource Management

cloudera

What's there to configure, anyway?

- On a 100-node cluster, there are likely 400+ (HDFS datanode, Yarn nodemanager, HBase RegionServer, Impalad) processes running. Each has environment variables, config files, and command line options!
- Where your daemons are running are configuration too, but often implicit. Moving from machine A to machine B is a configuration change!
- Most (but not all) settings require a restart to take effect.
- A management tool will help you with "scoping." Some configurations must be the same globally (e.g., kerberos), some make sense within a service (HDFS Trash), some per-daemon

$ vi /etc/hadoop/conf/hdfs-site.xml

- Configs are key-value pairs, in a straightforward if verbose XML format
- When in doubt, place configuration everywhere, since you might not know whether the client reads it or which daemons read it.
- Dear golly, use configuration management.

Editing a configuration

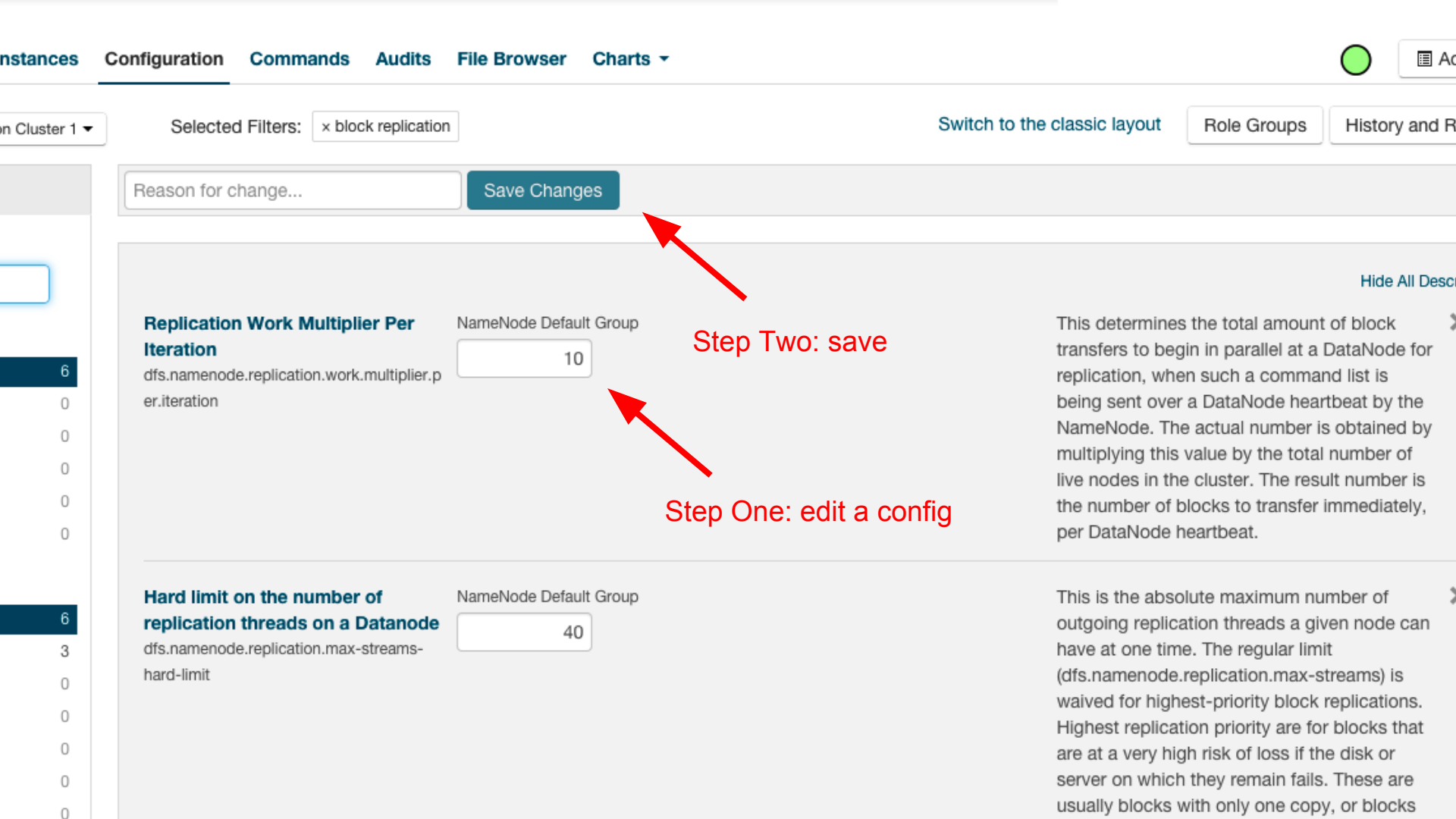
- Quick demo!

URLs and passwords:

http://tiny.cloudera.com/strata1 (user1/strata2015) (Kerberos)
admin@54.153.123.57 pass: admin321

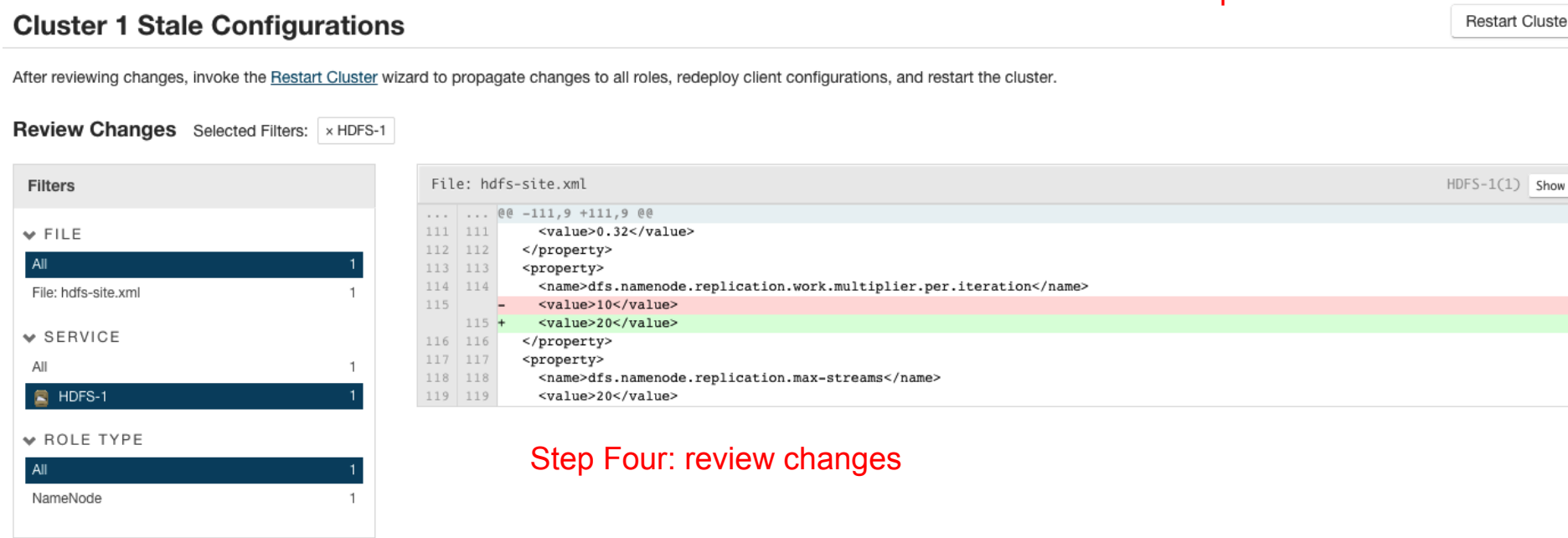
http://tiny.cloudera.com/strata2 (user2/strata2015)
admin@54.67.90.80 pass: admin321

http://tiny.cloudera.com/strata3 (user3/strata2015)
admin@54.153.83.11 pass: admin321


nstances
Configuration
Commands
Audits
File Browser
Charts
n Cluster 1
Selected Filters:
× block replication
Switch to the classic layout
Role Groups
History and R
Reason for change...
Save Changes
Hide All Desc
Replication Work Multiplier Per Iteration
dfs.namenode.replication.work.multiplier.per.iteration
NameNode Default Group
10
Step Two: save
Step One: edit a config
This determines the total amount of block transfers to begin in parallel at a DataNode for replication, when such a command list is being sent over a DataNode heartbeat by the NameNode. The actual number is obtained by multiplying this value by the total number of live nodes in the cluster. The result number is the number of blocks to transfer immediately, per DataNode heartbeat.
Hard limit on the number of replication threads on a Datanode
dfs.namenode.replication.max-streams-hard-limit
NameNode Default Group
40
This is the absolute maximum number of outgoing replication threads a given node can have at one time. The regular limit (dfs.namenode.replication.max-streams) is waived for highest-priority block replications. Highest replication priority are for blocks that are at a very high risk of loss if the disk or server on which they remain fails. These are usually blocks with only one copy, or blocks

Step Three: at top-level, note that restart is needed

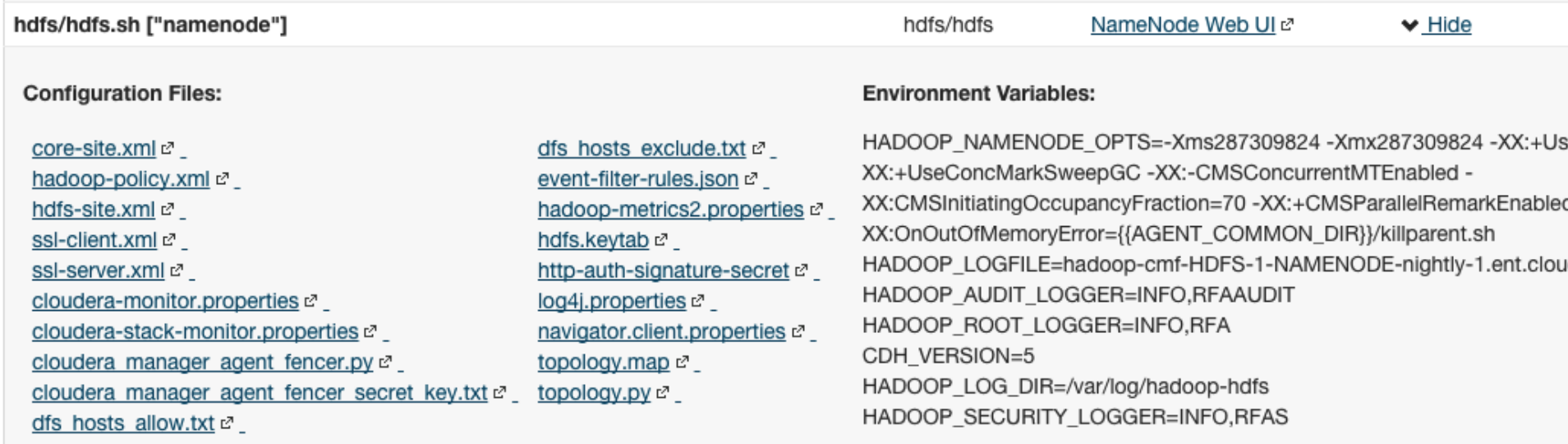
Step five: restart



Step Four: review changes

Show me the files!

- core-site.xml, hdfs-site.xml, dfs_hosts_allow, dfs_hosts_exclude, hbase-site.xml, hive-env.sh, hive-site.xml, hue.ini, mapred-site.xml, oozie-site.xml, yarn-site.xml, zoo.cfg (and so on)
- e.g., /var/run/cloudera-scm-agent/process/*-NAMENODE



Let’s take a look!

- Demo/Activity: find the files via UI

URLs and passwords:

http://tiny.cloudera.com/strata1 (user1/strata2015) (Kerberos)

admin@54.153.123.57 pass: admin321

http://tiny.cloudera.com/strata2 (user2/strata2015)

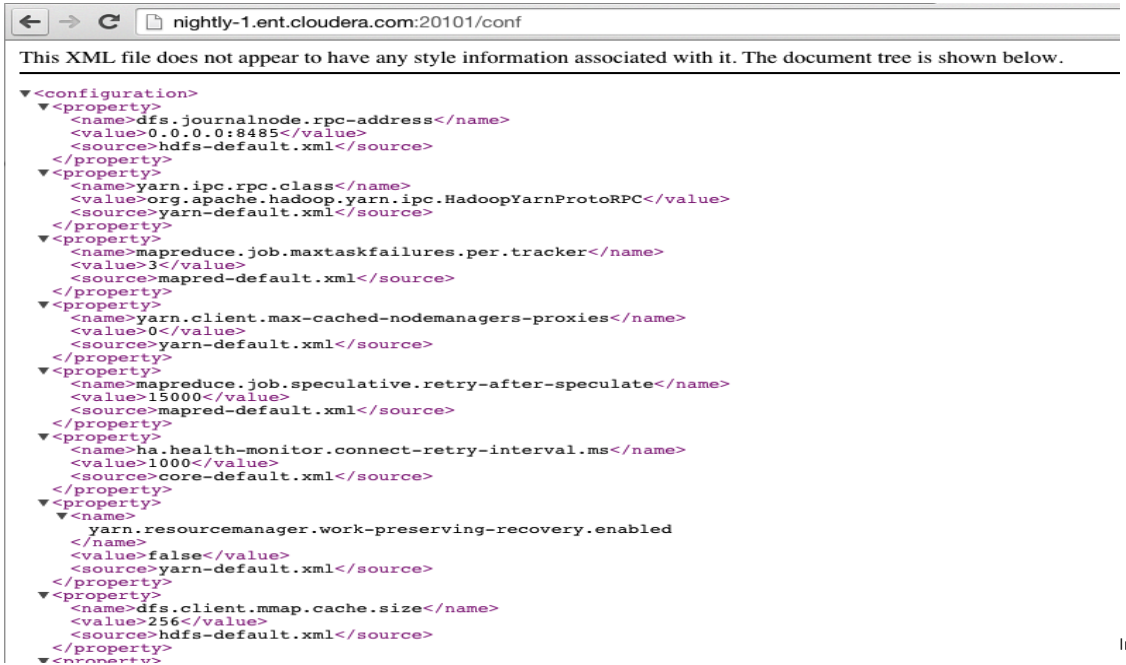
admin@54.67.90.80 pass: admin321

http://tiny.cloudera.com/strata3 (user3/strata2015)

admin@54.153.83.11 pass: admin321

How to double-check?

- http://tiny.cloudera.com/strata-nn-conf
  http://ec2-54-153-123-62.us-west-1.compute.amazonaws.com:50070/conf



Key Configuration Themes (everyone)

- Security
- Ports
- Heap Sizes
- Local Storage
- JVM (use Oracle JDK 1.7)
- Databases (back them up)

Rack Topology

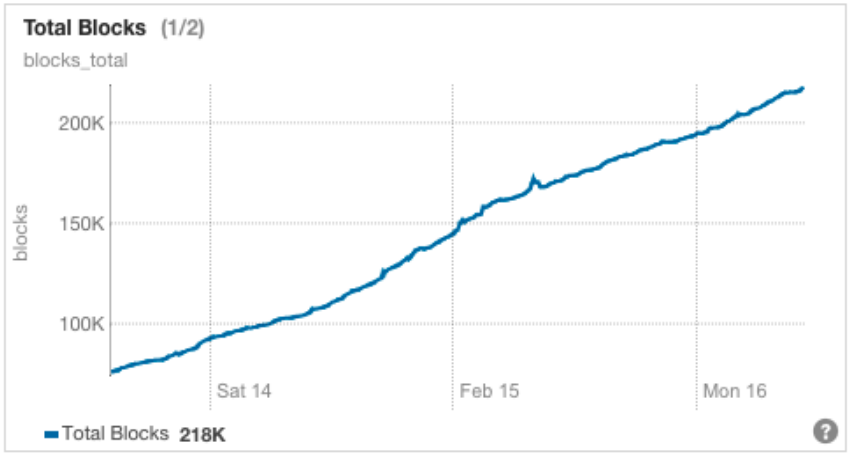
- Hadoop cares about racks because:
  - Shared failure zone
  - Network bandwidth
- When operating a large cluster, tell Hadoop (by use of a rack locality script) what machines are in which rack.

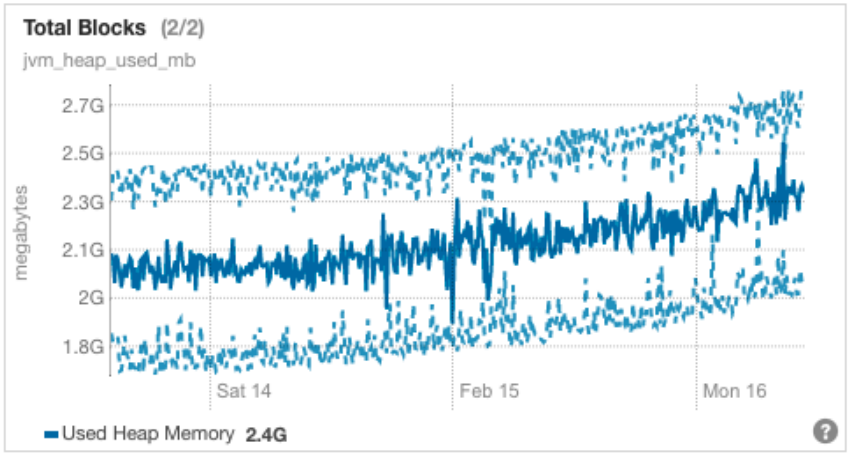
Networking

- Does your DNS work?
- Like, forwards and backwards?
- For sure?
- Have you really checked?

HDFS Configurations of Note

- Heap sizes:
  - Datanodes: linear in # of blocks
  - Namenode: linear in # of blocks and # of files






```
select blocks_total, jvm_heap_used_mb
where roletype=DATANODE and hostname RLIKE "hodor-016.*"
```

cloudera

HDFS Configurations of Note

- Local Data Directories
  - dfs.datanode.data.dir: one per spindle; avoid RAID
  - dfs.namenode.name.dir: two copies of your metadata better than one
- High Availability
  - Requires more daemons
    - Two failover controllers co-located with namenodes
    - Three journal nodes

Yarn Configurations of Note

- Yarn doles out resources to applications across two axes: memory and CPU
- Define per-NodeManager resources
  - yarn.nodemanager.resource.cpu-vcores
  - yarn.nodemanager.resource.memory-mb

Resource management in YARN

- Fannie and Freddie go in on a large cluster together. Fannie ponies up 75% of the budget; Freddie ponies up 25%.
- When cluster is idle, let Freddie use 100%
- When Fannie has a lot of work, Freddie can only use 25%.
- The “Fair Scheduler” implements “DRF” to share the cluster fairly across CPU and memory resources.
- Configured by an “allocations.xml” file.

Name	YARN					
	Weight	%	Virtual Cores Min / Max	Memory Min / Max	Max Running Apps	Scheduling Policy
root	1	100.0%	- / -	- / -	-	DRF
fannie	3	75.0%	- / -	- / -	-	DRF
freddie	1	25.0%	- / -	- / -	-	DRF

Quick Demo/Activity

- Look at heap sizes

http://tiny.cloudera.com/strata-heap

URLs and passwords:

http://tiny.cloudera.com/strata1 (user1/strata2015)

admin@54.153.123.57 pass: admin321

http://tiny.cloudera.com/strata2 (user2/strata2015)

admin@54.67.90.80 pass: admin321

http://tiny.cloudera.com/strata3 (user3/strata2015)

admin@54.153.83.11 pass: admin321

Activity

What is the current HDFS block size?

URLs and passwords:

http://tiny.cloudera.com/strata1 (user1/strata2015) (K

admin@54.153.123.57 pass: admin321

http://tiny.cloudera.com/strata2 (user2/strata2015)

admin@54.67.90.80 pass: admin321

http://tiny.cloudera.com/strata3 (user3/strata2015)

admin@54.153.83.11 pass: admin321

cloudera

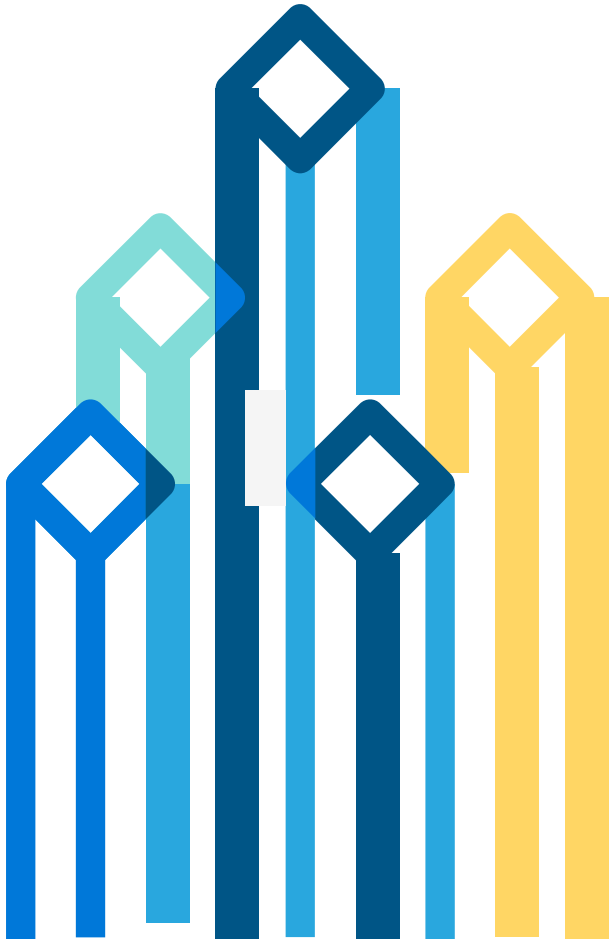
cloudera

Hokey-Pokey Break (You put your right CPU in…)

cloudera

Apache Hadoop Operations for Production Systems: Troubleshooting

Kathleen Ting



Troubleshooting

Managing Hadoop Clusters

Troubleshooting Hadoop Systems

Debugging Hadoop Applications

Troubleshooting

Managing Hadoop Clusters

Troubleshooting Hadoop Systems

Debugging Hadoop Applications

cloudera

Understanding Normal

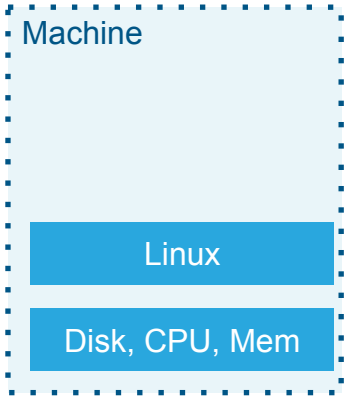
- Establish normal
  - Boring logs are good
- So that you can detect abnormal
  - Find anomalies and outliers
  - Compare performance
- And then isolate root causes
  - Who are the suspects
  - Pull the thread by interrogating
  - Correlate across different subsystems

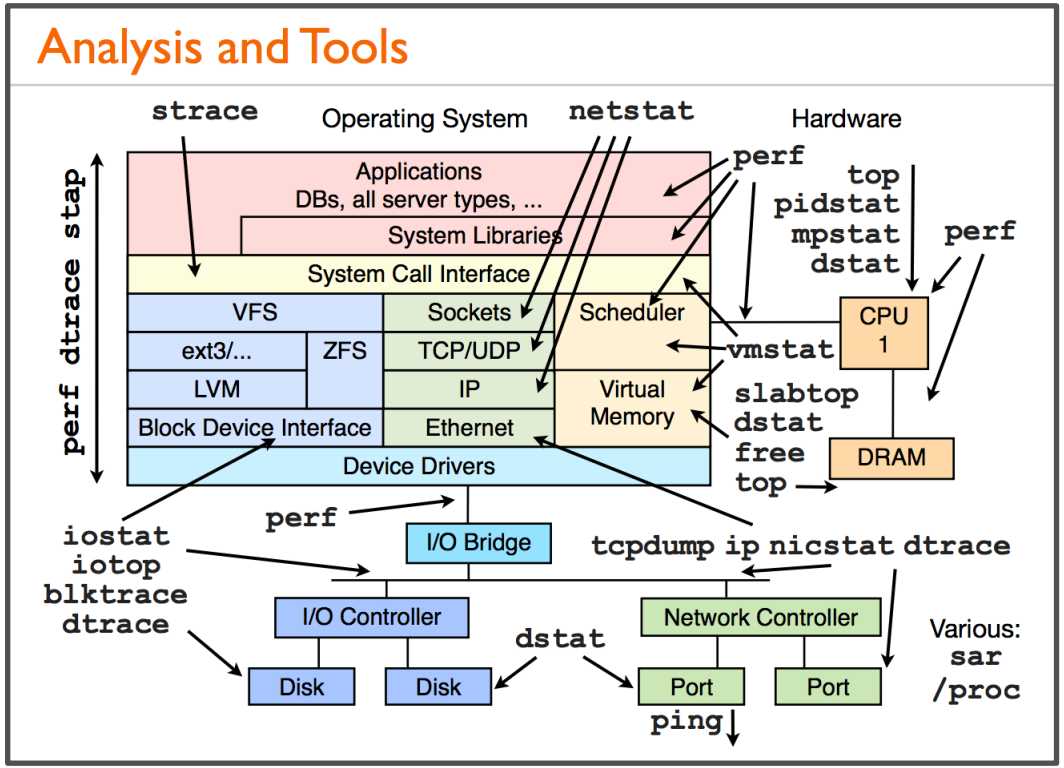
Basic tools

- What happened?
  - Logs
- What state are we in now?
  - Metrics
  - Thread stacks
- What happens when I do this?
  - Tracing
  - Dumps
- Is it alive?
  - listings
  - Canaries
- Is it OK?
  - fsck / hbck

cloudera

Diagnostics for single machines





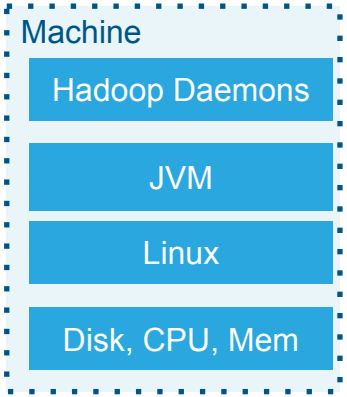
Analysis and Tools via @brendangregg

cloudera

Diagnosis Tools

	HW/Kernel
Logs	/log, /var/log dmesg
Metrics	/proc, top, iotop, sar, vmstat, netstat
Tracing	Strace, tcpdump
Dumps	Core dumps
Liveness	ping
Corrupt	fsck

Diagnostics for the JVM



- Most Hadoop services run in the Java VM, intros new java subsystems
  - Just in time compiler
  - Threads
  - Garbage collection
- Dumping Current threads
  - jstacks (any threads stuck or blocked?)
- JVM Settings:
  - Enabling GC Logs: -XX:+PrintGCDateStamps -XX:+PrintGCDetails
  - Enabling OOME Heap dumps: -XX:+HeapDumpOnOutOfMemoryError
- Metrics on GC
  - Get gc counts and info with jstat
- Memory Usage dumps
  - Create heap dump: jmap –dump:live,format=b,file=heap.bin <*pid*>
  - View heap dump from crash or jmap with jhat

cloudera

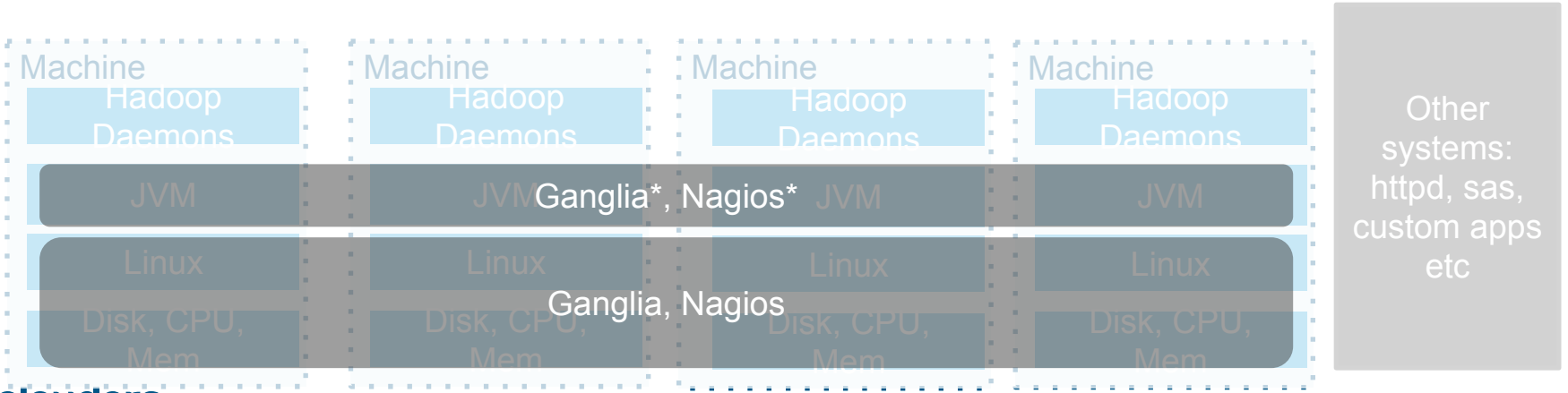
Diagnosis Tools

	HW/Kernel	JVM
Logs	/log, /var/log dmesg	Enable gc logging
Metrics	/proc, top, iotop, sar, vmstat, netstat	jstat
Tracing	Strace, tcpdump	Debugger, jdb, jstack
Dumps	Core dumps	jmap Enable OOME dump
Liveness	ping	Jps
Corrupt	fsck	

cloudera

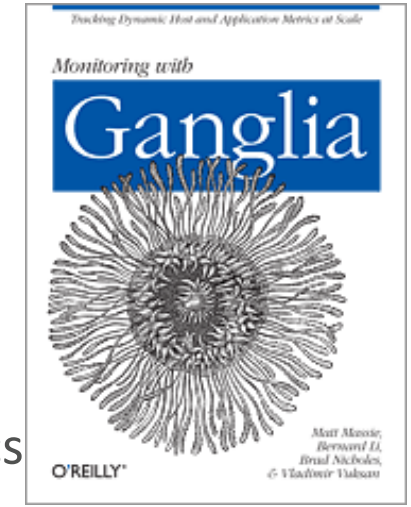
Tools for lots of machines

Tools for lots of machines

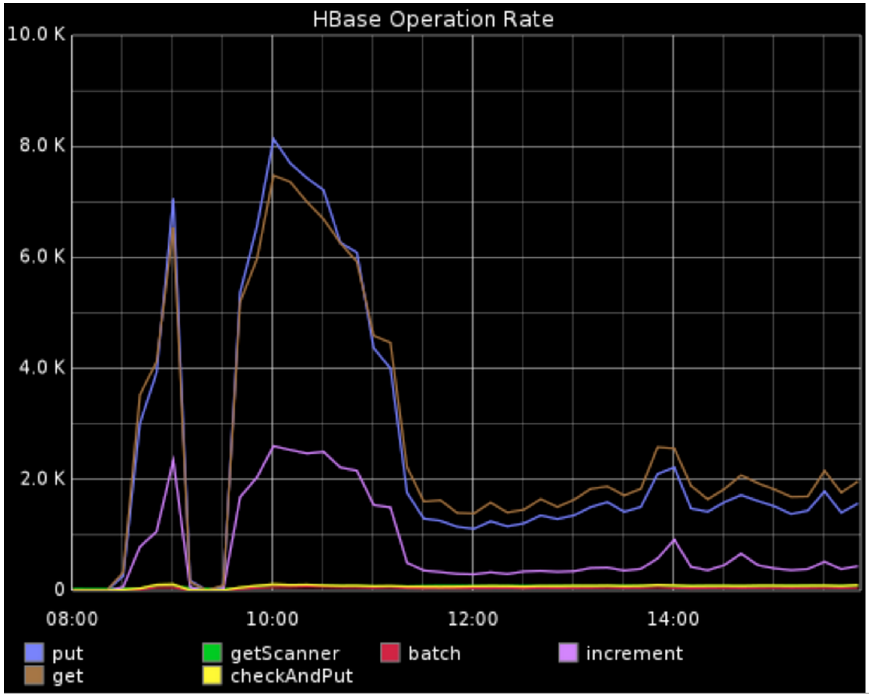


Ganglia

- Collects and aggregates metrics from machines
- Good for what's going on right now and describing normal perf
- Organized by physical resource (CPU, Mem, Disk)
  - Good defaults
  - Good for pin pointing machines
  - Good for seeing overall utilization
  - Uses RRDTool under the covers
- Some data scalability limitations, lossy over time.
- Dynamic for new machines, requires config for new metrics



Graphite



- Popular alternative to ganglia
- Can handle the scale of metrics coming in
- Similar to ganglia, but uses its own RRD database.
- More aimed at dynamic metrics (as opposed to statically defined metrics)

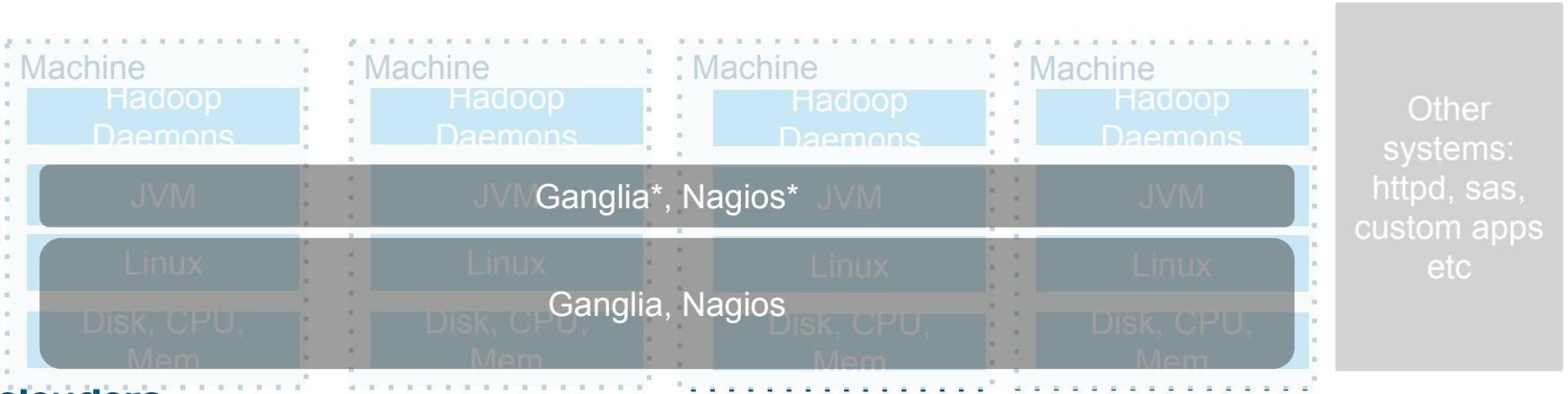
cloudera

Nagios

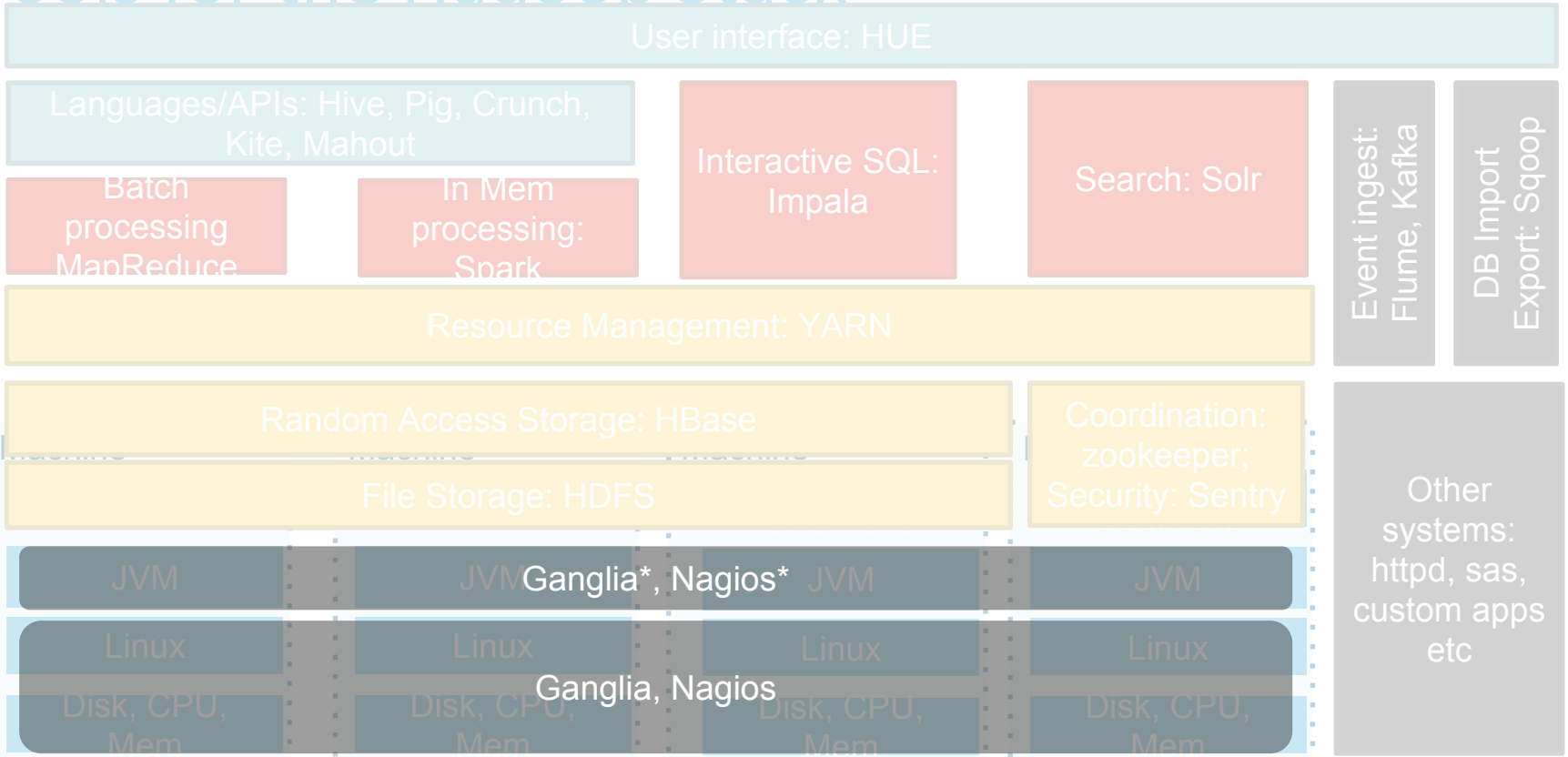
- Provides alerts from canaries and basic health checks for services on machines
- Organized by Service (httpd, dns, etc)
- Defacto standard for service monitoring

- Lacks distributed system know how
  - Requires bespoke setup for service slaves and masters
  - Lacks details with multi-tenant services or short-lived jobs

Tools for the Hadoop Stack



Tools for the Hadoop Stack



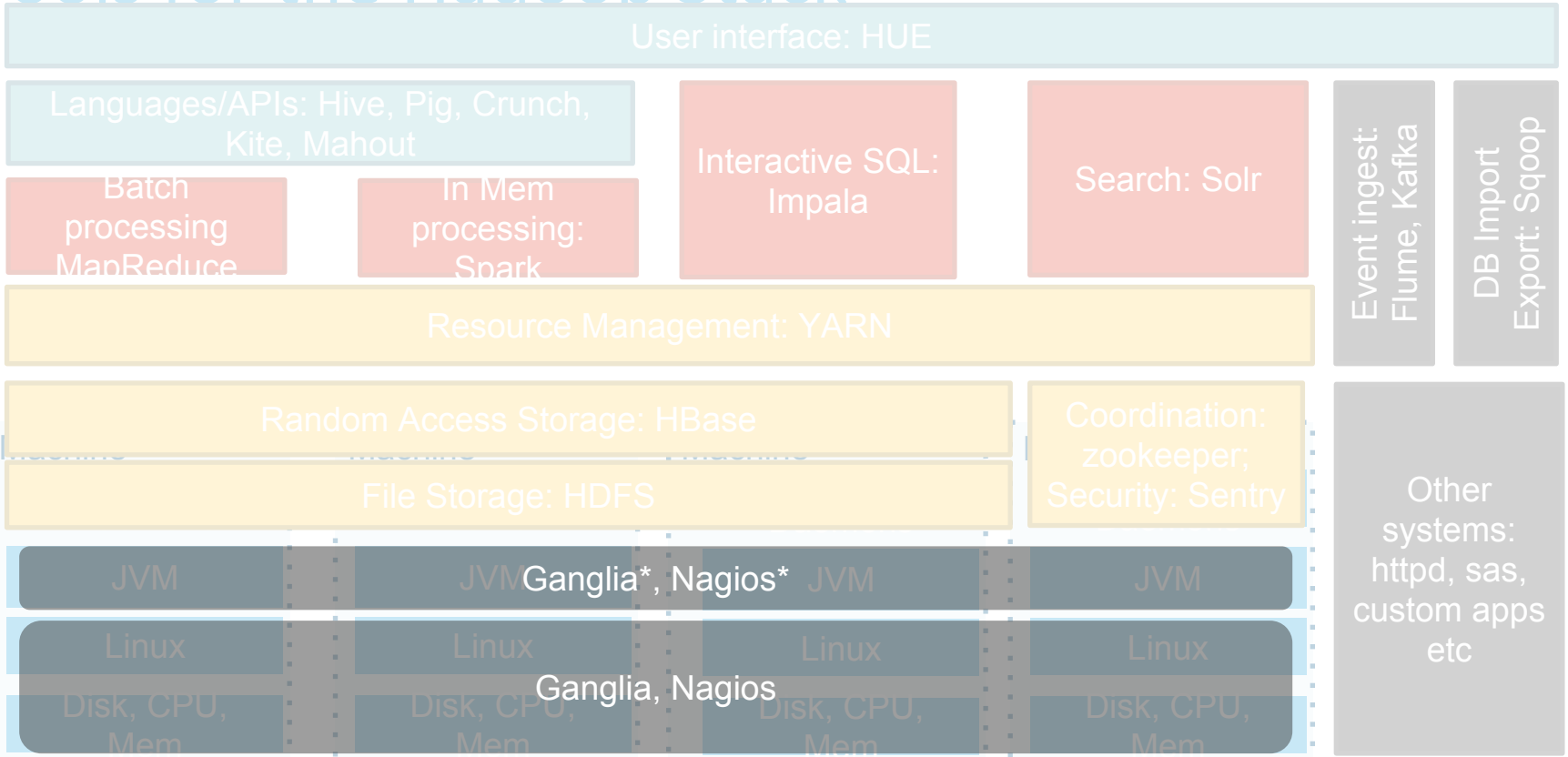
Diagnostics for the Hadoop Stack

- Single client call can trigger many RPCs spanning many machines
- Systems are evolving quickly
- A failure on one daemon, by design, does not cause failure of the entire service
- Logs:
  - Each service's master and slaves have their own logs: /var/log/
  - There are lot of logs and they change frequently
- Metrics:
  - Each daemon offers metrics, often aggregated at masters
- Tracing:
  - Htrace (integrated into Hadoop, HBase, recently in Apache Incubator)
- Liveness:
  - Canaries, service/daemon web uis

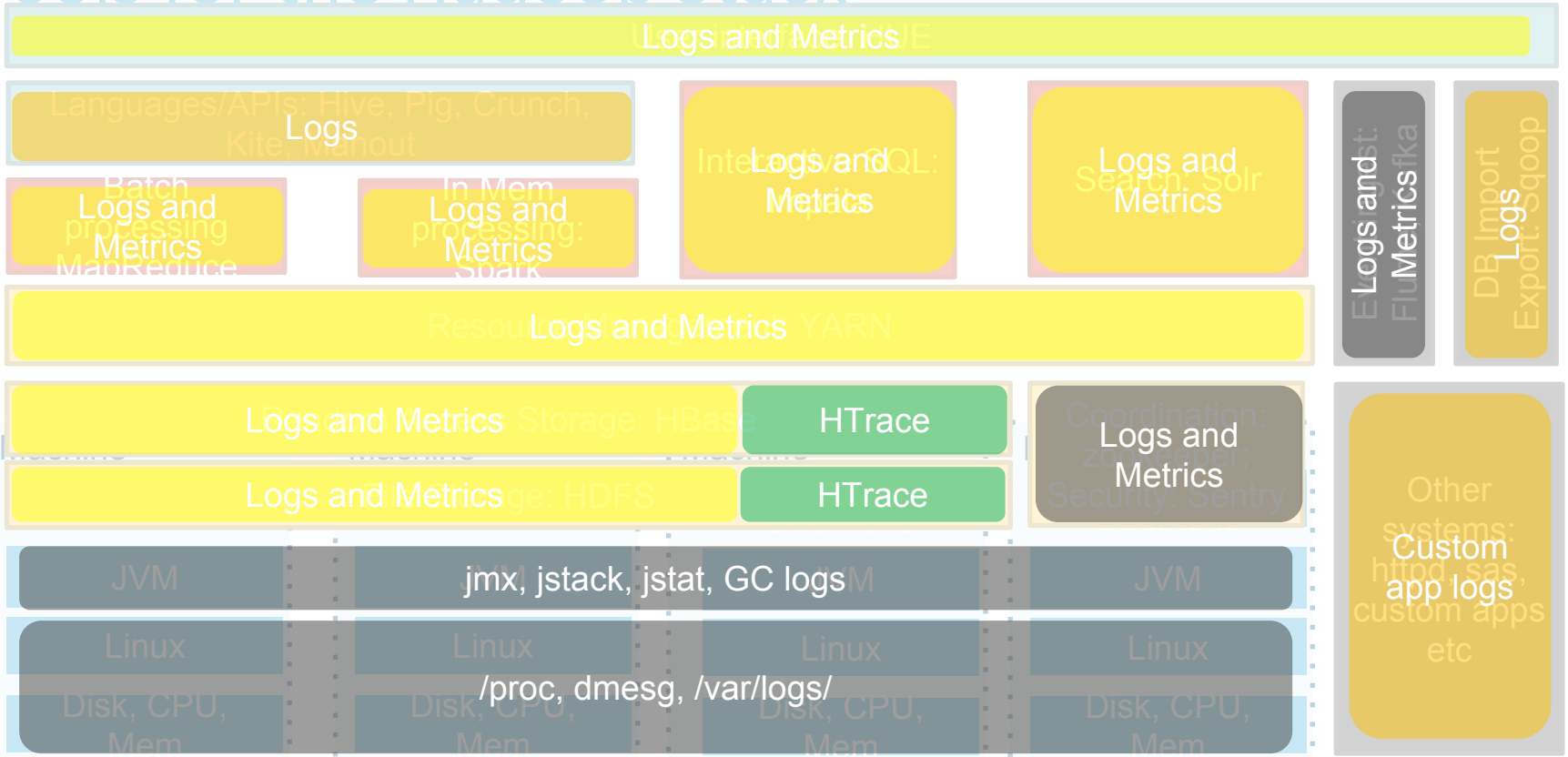
Diagnosis Tools

	HW/Kernel	JVM	Hadoop Stack
Logs	/log, /var/log dmesg	Enable gc logging	*:/var/log/hadoo|hbase|…
Metrics	/proc, top, iotop, sar, vmstat, netstat	jstat	*:/stacks, *:/jmx,
Tracing	Strace, tcpdump	Debugger, jdb, jstack	htrace
Dumps	Core dumps	jmap Enable OOME dump	*:/dump
Liveness	ping	Jps	Web UI
Corrupt	fsck		HDFS fsck, HBase hbck

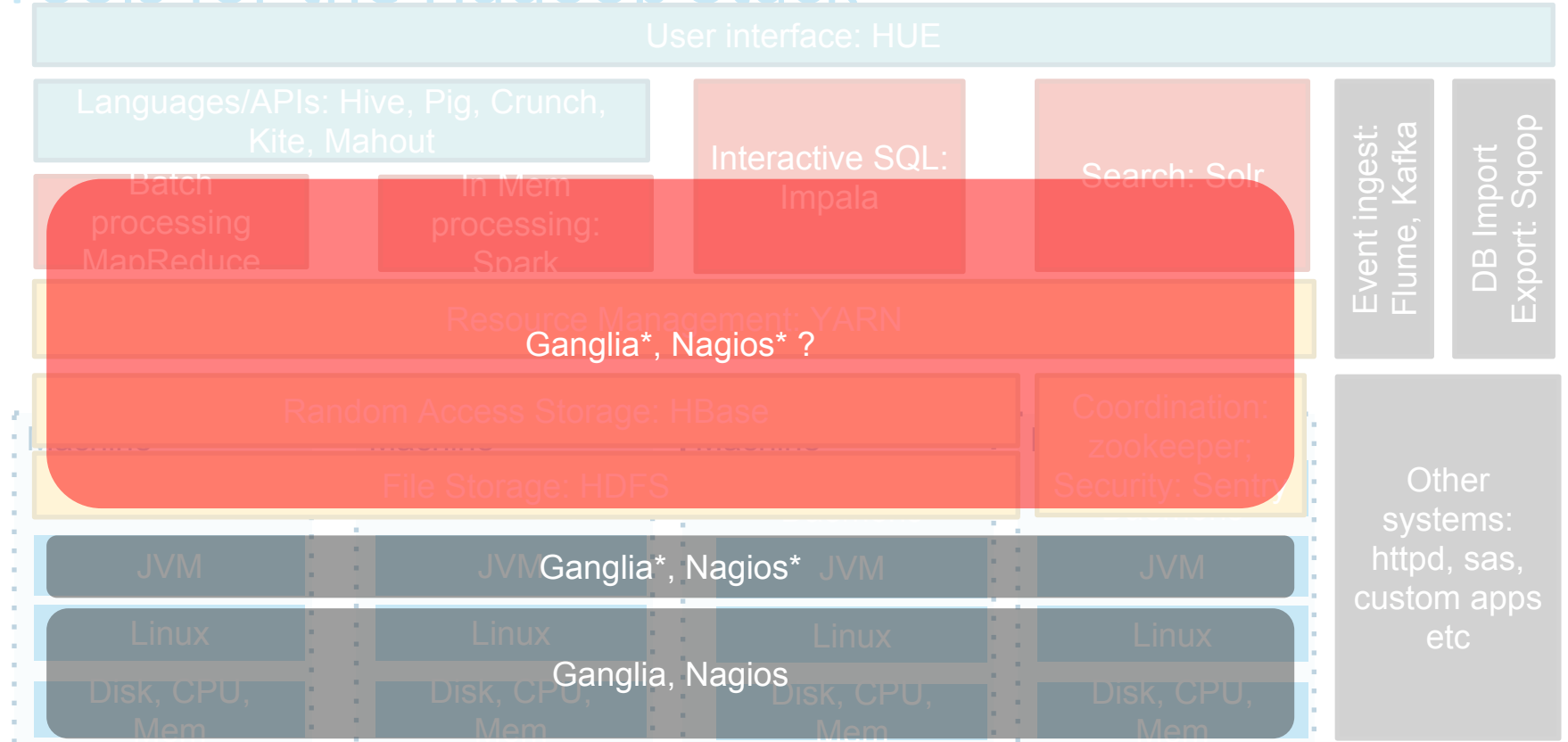
Tools for the Hadoop Stack



Tools for the Hadoop Stack



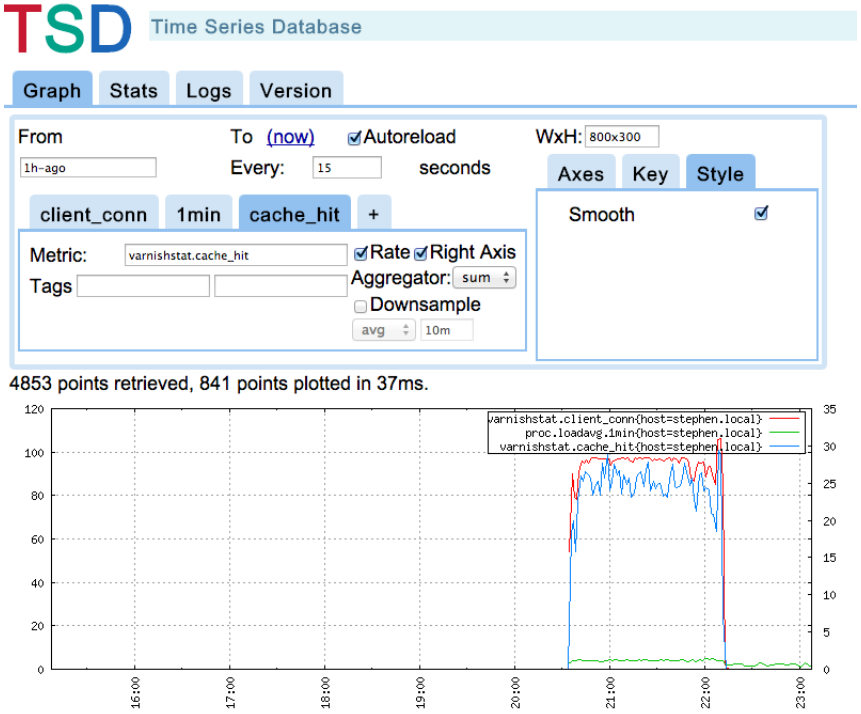
Tools for the Hadoop Stack



Ganglia and Nagios are not enough

- Fault tolerant distributed system masks many problems (by design!)
  - Some failures are not critical – failure condition more complicated
- Lacks distributed system know how
  - Requires bespoke setup for service slaves and masters
  - Lacks details with multitenant services or short-lived jobs
- Hadoop services are logically dependent on each other
  - Need to correlate metrics across different service and machines
  - Need to correlate logs from different services and machines.
  - Young systems where Logs are changing frequently
  - What about all the logs?
  - What of all these metrics do we really need?
- Some data scalability limitations, lossy over time
  - What about full fidelity?

OpenTSDB

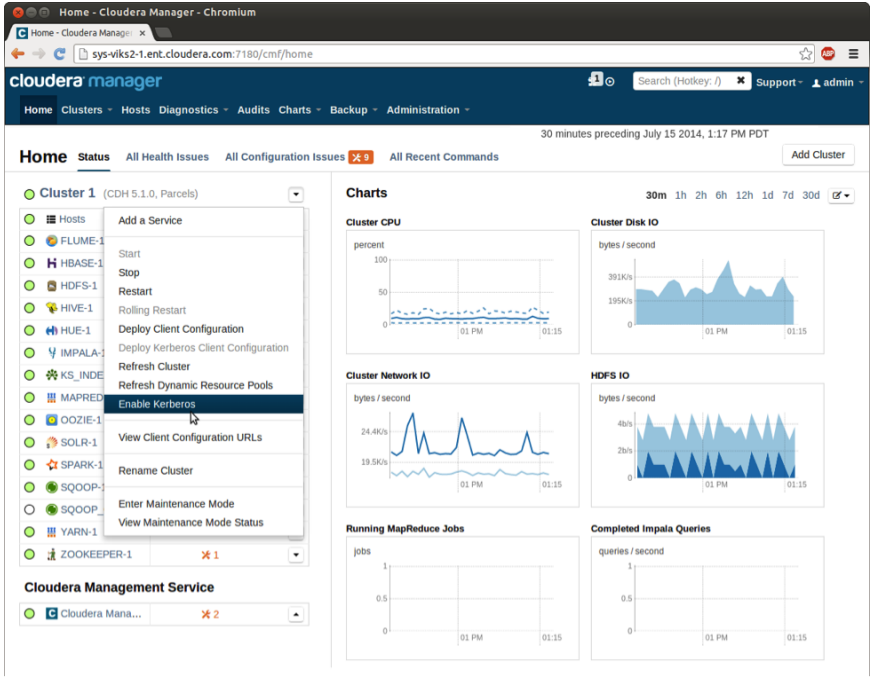


- OpenTSDB (Time Series Database)
  - Efficiently stores metric data into HBase
  - Keeps data at full fidelity
  - Keep as much data as your HBase instance can handle.

- Free and Open Source

cloudera

Cloudera Manager



- Extracts hardware, OS, and Hadoop service metrics specifically relevant to Hadoop and its related services.
  - Operation Latencies
  - # of disk seeks
  - HDFS data written
  - Java GC time
  - Network IO
- Provides
  - Cluster preflight checks
  - Basic host checks
  - Regular health checks
- Uses LevelDB for underlying metrics storage
- Provides distributed log search
- Monitors logs for known issues
- Point and click for useful utils (lsof, jstack, jmap)
- Free