TestingNov2009

Nov 2009 Testing Framework Conference Call

Some of the people on the Hadoop list are organising a quick conference call on the topic of testing, this wiki page is to go with it

JIRA Issues

- HADOOP-6332 Large-scale Automated Test Framework
- HADOOP-6248 Circus: Proposal and Preliminary Code for a Hadoop System Testing Framework
- HDFS-708 : A stress-test tool for HDFS.

Use Cases

Here are some of the use cases that come up when you consider testing Hadoop

Benchmarking

One use case that comes up is stress testing clusters; to see the cluster supports Hadoop "as well as it should", and trying to find out why it doesn't, if it is not adequate. What we have today is TeraSort, where you have to guess the approximate numbers then run the job. TeraSort creates its own test data, which is good, but it doesn't stress the CPUs as realistically as many workloads, and it generates lots of intermediate and final data; there is no reduction.

- Benchmarking slides

Basic Cluster Health Tests

There are currently no tests that work with Hadoop via the web pages, no job submission and monitoring. It is in fact possible to bring up a Hadoop cluster in which JSP doesn't work, but the basic tests all appear well -even including TeraSort, provided you use the low-level APIs.

Proposals:

- Create a set of JUnit/HtmlUnit tests that test the GUI; design these to run against any host. Either check out the source tree and run the against a remote cluster, or package the tests in a JAR and make this a project distributable.

- We may need separate test JARs for HDFS and mapreduce.

Testing underlying platforms

We need to test the underlying platforms, from the JVM and Linux distributions to any Infrastructure-on-Demand APIs that provide VMs on demand, machines which can run Hadoop.

JVM Testing

An IBM need; can also be used to qualify new Sun releases. Any JVM defect which stops Hadoop running at scale should be viewed as a blocking issue by all JVM suppliers.

- Need to be able to install latest JVM build, run the stress tests.

OS Testing

Test Hadoop working on the target OS. If Hadoop is packaged in an OS specific format (e.g. RPM), those installations need to be tested.

- Need to be able to create new machine images (PXE, kickstart, etc.), then push out Hadoop to the nodes and test the cluster.
- Cluster setup times can be significant if you have to reboot and re-image physical machines.

IaaS Testing

Hadoop can be used to stress test Infrastructure as a Service platforms, and is offered as a service by some companies (Cloudera, EC2).

Hadoop can be used on Eucalyptus installations using EC2 client libraries. This can show up problems with Eucalyptus (different fault messages compared EC2, time zone/clock differences.

Other infrastructures will have different APIs, with different features (private subnets, machine restart and persistence)

- Need to be able to work with different infrastructures and unstable APIs.
- Machine Allocation/release becomes a big delay on every test case that creates new machines

- Testing on EC2 runs up rapid bills if you create/destroy machines every junit test method, or even every test run. Best to create a small pool of machines at the start of the working day, release them in the evening. And to have build file targets to destroy all of a developer's machines -and to run it at night as part of the CI build.
- Troubleshooting on IaaS platforms can be interesting as the VMs get destroyed -the test runner needs to capture (relevant) local log data.
- SSH is the primary way to communicate with the (long-haul) cluster, even from a developer's local machine.
- Important not to embed private data -keys, logins, in build files, test reports or disk images
- Sometimes on EC2, and more often on smaller/unstable clusters, the allocated machines don't come up or "aren't right". You need to do early health checks on the machines, and if they are unwell, release and reallocate them.
- For testing local Hadoop builds on IaaS platforms, the build process needs to scp over and install the Hadoop binaries and the configuration files. This can be done by creating a new disk image that is then used to bootstrap every node, or you start with a base clean image and copy in Hadoop on demand. The latter is much more agile and cost effective during iterative development, but doesn't scale to very-large clusters (1000s of machines), unless you delegate the task of copy/install to the first few tens of allocated machines. For EC2, one tactic is to upload the binaries to S3, and have scripts on the nodes to copy down and install the files.

See: A Cloud Tools Manifesto

Qualifying Hadoop on different platforms

Currently Hadoop is only used at scale on RHEL + Sun JVM, because that is what Yahoo! run their clusters on, and nobody else is running different platforms in their production clusters -or if they are, they aren't discussing it in public.

- It would be interesting to start collecting experiences with running Hadoop on other platforms -different Unix flavours in particular, even if this is not a formal pre-release process.
- Windows and OS/X support Hadoop, reluctantly, with Windows being the most reluctant. Nobody admits to using Windows in production, and it may not get tested at any serious scale before a release is made.

What would it take to test Hadoop releases on different operating systems? We'd need clusters of real or virtual machines and then run any cluster qualification tests on them; publish the results. This would not be a performance game; throughput isn't important, it's more "does this work on a specific OS at 100+ machines"?

Exploring the Hadoop Configuration Space

There are a lot of Hadoop configuration options, even ignoring those of the underlying machines and network. For example, what impact does blocksize and replication factor have on your workload? What different network card configuration parameters give the best performance? Which combinations of options break things?

When combined with IaaS platforms, the configuration space gets even larger.

Manually exploring the configuration space takes too long; currently everyone tries to stick closed to the Yahoo! configurations which are believed to work -whenever someone strays off it, interesting things happen. For example, setting a replication factor of only 2 found a duplication bug; running Hadoop on a machine that isn't quite sure of its hostname shows up other assumptions as things you can not rely on.

- There is existing work on automated configuration testing, notably the work done by Adam Porter and colleagues on Distributed Continuous Quality Assurance
- (Steve says) in HP we've used a Pseudo-RNG to drive transforms to the infrastructure and deployed applications, this explores some of the space and is somewhat replicable.

Proposal: Make this a research topic, pull in the experts in testing, and give encouragement to work on this problem. Offering cluster time may help.

Testing applications that run on Hadoop

This was goal of Alex's Circus prototype: something to make it easier for you to be confident that your code will work.

Testing changes to Hadoop, fast

Hadoop unit/functional testing is slow with MiniMR/MiniDFS cluster setup and teardowns per test. This could be addressed by having more Mini cluster reuse, but it could be even faster if people could push out newly compiled JARs and test them at scale.

Testing Hadoop Distributions

This is a problem which Cloudera and others who distribute/internally package and deploy Hadoop have: you need to know that your RPMs or other redistributables work.

It's similar to the cluster acceptance test problem, except that you need to create the distribution packages and install them on the remote machines, then run the tests. The testing-over-IaaS platforms use cases are closer.

- Testing RPM upgrades from many past versions is tricky.

Simulating Cluster Failures

Cluster failure handling -especially the loss of large portions of a large datacenter, is something that is not currently formally tested. There are big fixes that go into Hadoop to test some of this, but loss of a quarter of the datanodes is a disaster that doesn't get tested at scale before a release is made.

- Network failures can be simulated on some IaaS platforms just by breaking a virtual link
- Forcibly killing processes is a more realistic approach which works on most platforms, though it is hard to choreograph

Proposal: Any Infrastructure API ought should offer the opportunity to simulate failures, either by turning off nodes without warning, or (better) breaking the network connections between live nodes.