

# Plugging the Holes: Security and Compatibility

Owen O'Malley  
Yahoo! Hadoop Team <sup>TM</sup>  
owen@yahoo-inc.com

YAHOO!



## Who Am I?

---

- ! Software Architect working on Hadoop since Jan 2006
  - ! Before Hadoop worked on Yahoo Search's WebMap
  - ! My first patch on Hadoop was Nutch-197
  - ! First Yahoo Hadoop committer
  - ! Most prolific contributor to Hadoop (by patch count)
  - ! Won the 2008 1TB and 2009 Minute and 100TB Sort Benchmarks
- ! Apache VP of Hadoop
  - ! Chair of the Hadoop Project Management Committee
  - ! Quarterly reports on the state of Hadoop for Apache Board



# What are the Problems?

---

- ! Our shared clusters increase:
  - ! Developer and operations productivity
  - ! Hardware utilization
  - ! Access to data
- ! Yahoo! wants to put customer and financial data on our Hadoop clusters.
  - ! Great for providing access to all of the parts of Yahoo!
  - ! Need to make sure that only the authorized people have access.
- ! Rolling out new versions of Hadoop is painful
  - ! Clients need to change and recompile their code



# Hadoop Security

---

- ! Currently, the Hadoop servers trust the users to declare who they are.
  - ! It is very easy to spoof, especially with open source.
  - ! For private clusters, we will leave non-security as option
- ! We need to ensure that users are who they claim to be.
- ! All access to HDFS (and therefore MapReduce) must be authenticated.
- ! The standard distributed authentication service is Kerberos (including ActiveDirectory).
- ! User code isn't affected, since the security happens in the RPC layer.



# HDFS Security

---

- ! Hadoop security is grounded in HDFS security.
  - ! Other services such as MapReduce store their state in HDFS.
- ! Use of Kerberos allows a single sign on where the Hadoop commands pick up and use the user's tickets.
- ! The framework authenticates the user to the Name Node using Kerberos before any operations.
- ! The Name Node is also authenticated to the user.
- ! Client can request an HDFS Access Token to get access later without going through Kerberos again.
  - ! Prevents authorization storms as MapReduce jobs launch!



# Accessing a File

---

- ! User uses Kerberos (or a HDFS Access Token) to authenticate to the Name Node.
- ! They request to open a file **X**.
- ! If they have permission to file **X**, the Name Node returns a token for reading the blocks of **X**.
- ! The user uses these tokens when communicating with the Data Nodes to show they have access.
- ! There are also tokens for writing blocks when the file is being created.



# MapReduce Security

---

- ! Framework authenticates user to Job Tracker before they can submit, modify, or kill jobs.
- ! The Job Tracker authenticates itself to the user.
- ! Job's logs (including stdout) are only visible to the user.
- ! Map and Reduce tasks actually run as the user.
- ! Tasks' working directories are protected from others.
- ! The Job Tracker's system directory is no longer readable and writable by everyone.
- ! Only the reduce tasks can get the map outputs.



# Interactions with HDFS

---

- ! MapReduce jobs need to read and write HDFS files as the user.
- ! Currently, we store the user name in the job.
- ! With security enabled, we will store HDFS Access Tokens in the job.
- ! The job needs a token for each HDFS cluster.
- ! The tokens will be renewed by the Job Tracker so they don't expire for long running jobs.
- ! When the job completes, the tokens will be cancelled.



# Interactions with Higher Layers

---

- ! Yahoo uses a workflow manager named Oozie to submits MapReduce jobs on behalf of the user.
- ! We could store the user's credentials with a modifier (oom/oozie) in Oozie to access Hadoop as the user.
- ! Or we could create Token granting Tokens for HDFS and MapReduce and store those in Oozie.
- ! In either case, such proxies are a potential source of security problems, since they are storing large number of user's access credentials.



# Web UIs

---

- ! Hadoop and especially MapReduce make heavy use of the Web UIs.
- ! These need to be authenticated also...
- ! Fortunately, there is a standard solution for Kerberos and HTTP, named SPNEGO.
- ! SPNEGO is supported by all of the major browsers.
- ! All of the servlets will use SPNEGO to authenticate the user and enforce permissions appropriately.



# Remaining Security Issues

---

- ! We are not encrypting on the wire.
  - ! It will be possible within the framework, but not in 0.22.
- ! We are not encrypting on disk.
  - ! For either HDFS or MapReduce.
- ! Encryption is expensive in terms of CPU and IO speed.
- ! Our current threat model is that the attacker has access to a user account, but not root.
  - ! They can't sniff the packets on the network.



# Backwards Compatibility

---

- ! API
- ! Protocols
- ! File Formats
- ! Configuration



# API Compatibility

---

- ! Need to mark APIs with
  - ! Audience: Public, Limited Private, Private
  - ! Stability: Stable, Evolving, Unstable
  - @InterfaceAudience.Public*
  - @InterfaceStability.Stable*
  - public class Xxx {...}*
  - ! Developers need to ensure that 0.22 is backwards compatible with 0.21
- ! Defined new APIs designed to be future-proof:
  - ! MapReduce – Context objects in `org.apache.hadoop.mapreduce`
  - ! HDFS – FileContext in `org.apache.hadoop.fs`



# Protocol Compatibility

---

- ! Currently all clients of a server must be the same version (0.18, 0.19, 0.20, 0.21).
- ! Want to enable forward and backward compatibility
- ! Started work on Avro
  - ! Includes the schema of the information as well as the data
  - ! Can support different schemas on the client and server
  - ! Still need to make the code tolerant of version differences
  - ! Avro provides the mechanisms
- ! Avro will be used for file version compatibility too



# Configuration

---

- ! Configuration in Hadoop is a string to string map.
- ! Maintaining backwards compatibility of configuration knobs was done case by case.
- ! Now we have standard infrastructure for declaring old knobs deprecated.
- ! Also have cleaned up a lot of the names in 0.21.



# Questions?

---

- ! Thanks for coming!
- ! Mailing lists:
  - ! [common-user@hadoop.apache.org](mailto:common-user@hadoop.apache.org)
  - ! [hdfs-user@hadoop.apache.org](mailto:hdfs-user@hadoop.apache.org)
  - ! [mapreduce-user@hadoop.apache.org](mailto:mapreduce-user@hadoop.apache.org)
- ! Slides posted on the Hadoop wiki page
  - ! <http://wiki.apache.org/hadoop/HadoopPresentations>