Developing a Big Data Chapel
Chris Taylor, Department of Defense

This talk will cover a year of development efforts tuning aspects of Chapel for use as a data science platform in a production cloud environment. The production cloud in question uses HDFS for storage and features job orchestration using the Mesos cloud operating system. One effort developed software tools to specifically run Chapel jobs on a Mesos cloud. Another effort required tuning the Chapel HDFS library and the Chapel runtime to remove external dependencies on the Java Virtual Machine (JVM) and the Hadoop Java archives. A last effort implemented three machine learning algorithms using Chapel's parallel computing capabilities. The machine learning effort will highlight an expanded effort to develop a Chapel module similar to Python's Theano library.

Mesos implements multi-tenancy resource and task assignment using cgroup containers. Tools used to run programs on a Mesos orchestrated cloud are referred to as Mesos Frameworks. Typically a Mesos Framework consists of two programs, a Scheduler and an Executor. The talk will highlight some of the initial challenges running Chapel on a Mesos cloud, why a new Chapel-Mesos Framework was required, how the current Chapel-Mesos Framework operates, and some future work to improve the "Chapel-Mesos experience".

Compute clouds using HDFS storage pose a small deployment challenge for Chapel programs. Chapel's HDFS support uses the default C-bindings offered by Hadoop. Under the hood, Hadoop's C-bindings use the JVM to load Hadoop Java archives in order to interact with HDFS. Reliance on the JVM poses a systems administrative hurdle; each compute node or Docker container executing a Chapel program requires an up-to-date deployment of the JVM and the Hadoop Java archives. Some of the more popular GNU/Linux distributions do not provide statically linked versions of the JVM and ensuring each compute node or Docker container has an up-to-date Hadoop Java archives can complicate Mesos orchestration. The talk will detail work that removed the JVM and the Java archive dependencies by integrating PivotalHD's libhdfs3, a pure C++ client library for interacting with HDFS, into the Chapel runtime's I/O system.

A data science platform needs machine learning packages that can be used or extended for analytic development and deployment. This past year, three machine learning algorithms were implemented in Chapel. Each algorithm's performance characteristics along with how the algorithm uses Chapel's parallel feature set will be presented. The three algorithms implemented for this talk are Latent Dirichlet Allocation using Collapsed Gibbs Sampling for topic modeling, Logistic Regression with Mini-batch Stochastic Gradient Descent for classification, and Random Forests for classification.

The talk will finish with a highlight of some current work developing a Chapel module similar in form to Python's Theano library. Development of a Theano-styled Chapel module could provide a solution for one of the resiliency issues highlighted in last year's CHIUW technical talk "Parallac: Using Chapel with ARM Clusters".