Project 1706 netKarma Experiences Report

Sep 30, 2010

This report summarizes our experiences in provenance capture in GENI. In year 1 of the project, we focused efforts on capturing provenance through the GENI User Shell tool, GUSH. The goal was to capture as much information about the experiment as possible by capturing what GUSH knew about hosts, processes, data objects, etc. GUSH is an extensible execution management system for GENI. It is being used to deploy, execute and monitor applications running on the GENI experimental networks.

In our experimental studies, we utilized PlanetLab, where we run GUSH to deploy and execute applications. GUSH requires that users describe their experiments or computation either through an XML document or on the command line. It uses this information to locate and access the remote resources in PlanetLab. In its execution flow, GUSH contacts a host to deploy the application under consideration. Then, it runs the application; captures all the standard output produced by the application during its execution; and dumps it into a log file. Thus, as result of each experiment run, Gush produces an experiment log file.

Provenance capture was given effect by developing a tool called an Adaptor that examines GUSH log files. The Adaptor uses the GUSH experiment log files and a set of rules to derive provenance information and maps them into the data model of the Karma repository. The adaptor is simply a generic log processing unit for log files, which comprise of two sub-units: Log Parser, Notification Generator. The Log Parser module is used to process log files to extract provenance information, while the Notification Generator is used for generating and sending provenance notifications to Karma repository.

We found that the GUSH log file differs between experiments that are specified on the GUSH command line versus experiments that are specified through an XML document that is submitted as input to GUSH. *This was unexpected, and uniformity of log files could be a suggested improvement by the GUSH team.* The test experiments that come with GUSH are simple in that the output goes through stdout. We have been working with a richer application, namely a MapReduce application, but MapReduce does its own node scheduling, which hides some of the deployment information from netKarma.

We found that the Adaptor needed additional rules to infer connections between processes and the outputs (or data products created in provenance terms) as their ordering in the log file is nondeterministic.

We are currently harvesting information about hosts and circuits from the GMOC database to augment the host descriptions we have in netKarma that are associated with experimental runs. As of this report, we have no experience based suggestions to offer the GMOC team as things are proceeding smoothly.