11th International Workshop on Science Gateways (IWSG 2019), 12-14 June 2019

FuncX: High Performance Function-as-a-Service for Science

Kyle Chard, Ryan Chard, Tyler Skluzacek, Yadu Babuji, Zhuozhao Li, Ian Foster
University of Chicago and Argonne National Laboratory
Chicago, Illinois 60605
Email: chard@uchicago.edu

***Abstract*—Increases in data volume, variety, and velocity, combined with widespread adoption of machine learning, interactive computing, and real-time computing, are transforming scientific workloads from long-running batch jobs to short-duration function-based workloads. Simultaneously, the availability of high speed networks, universal trust fabrics, and containerization, alongside the adoption of high level scripting languages means that computation can occur wherever it makes the most sense: for example where data or specialized software or hardware are available, or where computing is fast, plentiful, or cheap. In industry, similar trends have motivated the adoption of Function-as-a-Service (FaaS) models in which programming functions are executed on an elastic pool of computing resources, and where users need not concern themselves with the location in which these functions are executed or the burden of managing compute resources or virtual machines.**

In response to the need to support function-based workloads, for example in workflows and science gateways, we have developed a flexible, scalable, and high performance function execution ecosystem called funcX. funcX builds upon components of the Parsl parallel scripting library to enable efficient and reliable execution of Python functions on a variety of computing resources including clouds, clusters, and supercomputers. Users first register a function with funcX, defining the function body (in Python), input/output signature, and Python and system dependencies. funcX dynamically creates a Singularity container from these dependencies and deploys specialized function execution management code to the container. The function may then be invoked by passing a set of input parameters or files to the funcX executor. The funcX executor, deployed on a compute resource, elastically manages a pool of containers by submitting batch jobs to a scheduler or using cloud APIs to provision nodes. It uses a high performance pilot job model to deploy appropriate containers and manage the execution of tasks. We have scaled this model to more than 8,000 concurrent workers and millions of function executions using the Theta supercomputer. In this talk we will introduce funcX, outline its architecture, describe how it can be used in science gateways, and highlight a range of scientific scenarios in which it is being used.

***Keywords*—Function-as-a-Service, Python, high performance computing**