# EC528 - Performance Analysis of Secure Multi-Party Computations in the Cloud

Hasnain Rehman (hasnain@bu.edu) Pierre-François Wolfe (pwolfe@bu.edu) Samyak Jain (samyakj@bu.edu) Suli Hu (sulihu@bu.edu) Yufeng Lin (yflin@bu.edu)

*Boston University, College of Engineering*

BOSTON UNIVERSITY

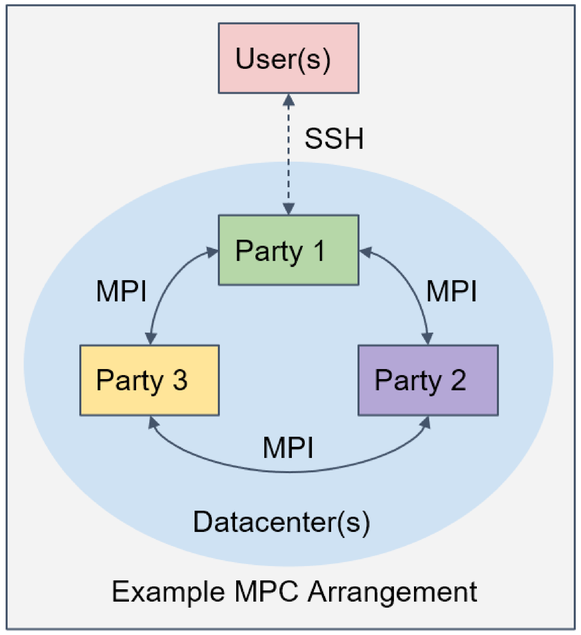
## Multi-Party Computation (MPC)

**MPC Benefits:**

- Enables mutually agreed computation using joint data
- Maintains privacy of data provided by each party
- No trust required in a single third-party for computation

**Some MPC Application Examples:**

- Marketplace with anonymous bidding
- Analyze medical data from multiple sources (HIPAA compliant)
- Salary trends for demographics from pool of companies




Example MPC Arrangement

## Project Goals

**Mentor Goals: (Already in progress at start of project)**

- Employ three-party Secret Sharing MPC
- Perform database queries across multiple private datasets
- Keep all operations secure vs. separating out insecure steps
- Implement clean MPC code with minimal dependencies

**Team Goals: (During Fall 2020 semester)**

- Deploy mentor MPC code on multiple platforms/configurations
- Improve benchmarking instrumentation for performance assessment
- Arrange a system of automation for easier ongoing testing
- Document and arrange information with an eye towards usability

## Deployments Explored

**Why these Deployments?**

- Local and remote development environments
- Clients/data may or may not be co-located
- Performance/deployment difficulty tradeoffs

**Deployments**

- Local (Bare-Metal or virtualized)
- Cloud-based Virtual Machines (MOC OpenStack)
- Cloud-based Containers (MOC OpenShift)
- Bare-Metal Clusters (CloudLab)

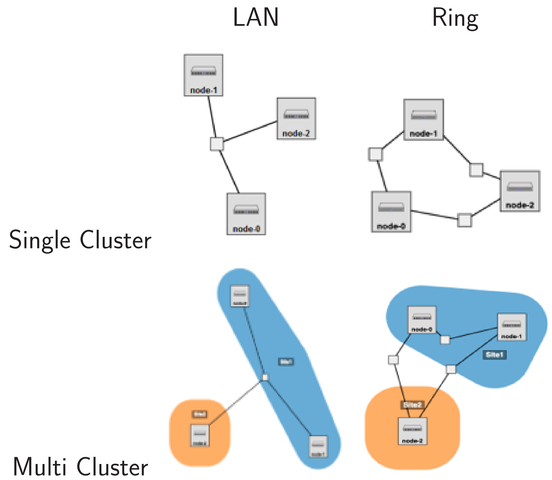
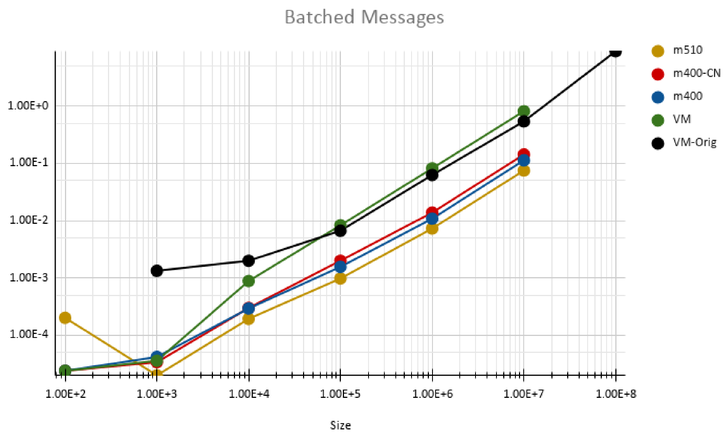



Table: CloudLab Topologies

## Preliminary Results




## Project Automation

**Features:**

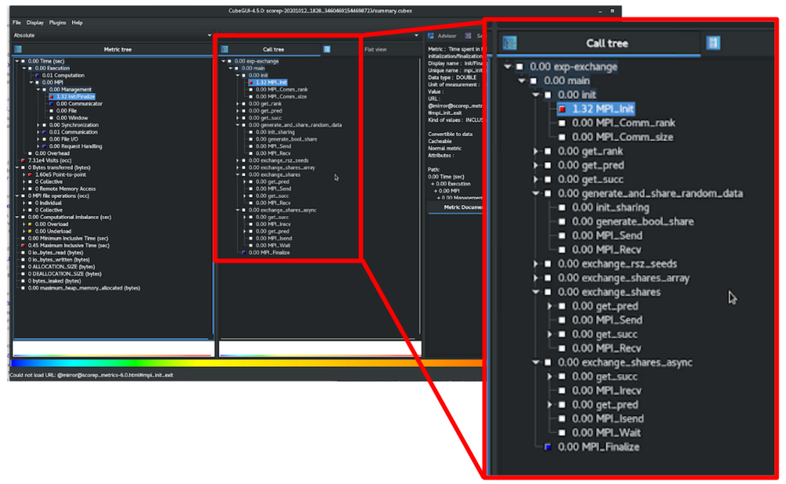
- Easy to use and extend
- Leads to a repeatable end configuration state
- Simplifies both setup and testing
- Can support most/all desired scenarios

**Evolution:**

1. Manual installation, configuration, and testing
2. Semi-automated: Shell scripts, geni-lib scripts, Dockerfiles, unpackable *.tar.gz,...
3. Automated Software: Ansible package installation, system configuration, software build, test execution, data retrieval

## Data Collection and Analysis

- C code instrumentation with clock_gettime
- Test input range and multiple samples output to *.csv
- Score-P wrapper for profiling MPI communication and more
- CUBE GUI for inspecting *.cubex and *.otf2 collections (below with execution tree expanded)




## Future Work

- Extend Ansible automation for additional OSs and environments
- Conduct additional MPI testing with tools developed
- Mentors wish to build a user frontend for deployment and testing