GENI Mini Project

You are allowed to work in teams of 4 (or less). You are asked to investigate more deeply a certain aspect of networking experimentally, over the GENI testbed.[1] The topic is up to you, though we list a few suggestions below. The final report should be published on GitHub and describe your problem, objectives, experimental methodology, results and analysis, along with a complete set of instructions that allows for the reproducibility of your setup and results. **The final report is due on Gradescope on Thursday, December 8, 2022 (no late submissions will be allowed).**

Basic Skills: *Students need to have basic knowledge of GitHub and should be familiar with command-line usage for all the projects.* **All final projects should be available on GitHub with the code and readme with clear deployment and usage instructions.**

Project Ideas:

Note: Given your project proposal would involve an application or protocol, the purpose of using GENI is to have control over the network, i.e., you can measure network throughput, delay, run an emulator on a link to set some losses or a certain rate, place VMs on different GENI racks/sites, define the topology/connectivity, use different underlying (e.g., transport) protocols, etc. and study the impact of the network on the application or protocol.

Password Cracker

Design a distributed system (like Hadoop), where a user submits the md5 hash of a 5-character password (a-z, A-Z) to the system using a web interface. The web interface with the help of worker nodes cracks the password by a brute force approach. Students need to implement the web-interface as well as the management service that will dedicate jobs to worker nodes. The web-interface and the management service can be on the same machine, but worker nodes need to be different machines. The systems should be scalable, i.e., you can add/remove workers on the fly. The management service should use the REST API or socket programming to communicate with the worker nodes.
Skills: CGI-bin, socket programming.

Image Recognition Application

Implement an image-recognition service which has a web interface, where a user should be able to submit an image query and the service should use any of image recognition techniques, e.g., Squeeze Net, Google Net or any deep learning neural network which classifies the image and returns the answer to the user. You don't need to worry about training the neural network and can use pre-trained weights. The web interface and the recognition systems should be on separate nodes, and you should connect them using socket programming or rest API.
Skills: CGI-bin, socket programming, usage of libraries/packages/models like SqueezeNet, etc.

Performance Comparison of various TCP versions

You would be required to study different aspects of TCP flavors (e.g., Reno, Cubic, Vegas, BBR) in terms of delays, throughput, fairness, etc. You can also read up on Explicit Congestion Notification (ECN) and compare the performance of TCP with ECN to without ECN. This TCP Congestion Control blog should be helpful to get you started, but you are expected to go beyond the results shown in that blog and rigorously analyze and compare the different TCP versions under different network conditions.
Skills: command-line usage, experiment design, TCP flavors, deploying open-source code.

Analyzing Transmission Policies for Adaptive Video

[1] Please use instaGENI racks as instaGENI racks have been recently upgraded and have more resources. You can check the status of GENI racks (sites) here: https://flsmonitor.fed4fire.eu/genitests. Try using any of the instaGENI racks with the most free VMs.

You would be required to study the tradeoff between different metrics of video quality (average rate, interruptions, and variability of rate) in an adaptive video delivery system, e.g., DASH (Dynamic Adaptive Streaming over HTTP). You would investigate different decision policies for adaptively selecting the video rate based on current network data rate. This Adaptive Video blog should be helpful to get you started, but you are expected to go beyond the results shown in that blog and carefully investigate the video quality metrics under different network conditions and for different rate adaptation algorithms.

Other Ideas

There are many GENI tutorials available, some of which we did. You can propose to extend any of them to study certain aspects in more detail. For example, this tutorial studies OSPF and you may want to study its scalability or compare it to some other routing protocol. Another tutorial studies firewalling and you may want to examine issues related to intrusion detection and prevention.