The Redesign and Reimplementation of the Stevens Institute of Technology Campus Network

Group 12
March 17, 2003

Faculty Advisor
Prof. Sumit Ghosh

Group Members
Grace Shabo
Nicholas Evans
Philip Tan
Jay Grossman
Hans Kim

We pledge our honor that we have abided by the Stevens Honor System.

I. Abstract

We present a simulation approach for the previously designed next generation campus network for the Stevens Institute of Technology. This simulation will allow us to verify our conclusions at the end of the design process about what would be an adequate infrastructure to support the current and future needs of the Stevens campus network users. Here we describe our results, to date, in the construction of the simulation network as well as the simulation philosophy we will follow. The goal is to simulate our design, create a performance index to allow comparison of server hardware between our simulation network and the production network, and extrapolate our results so we can provide specifications for the finalized production ready network.

II. Project Progress

The design approach of the prototype will be the implementation of two key areas in the overall redesign of the campus network. Currently the team is in the process of implementing a mockup of Attila using multiple computers set up to do specific tasks.

Redesign of Attila

As described in last semester's final report, Attila will be implemented as a set of servers, each handling a specific service that Attila currently provides to the Stevens community. In order to draw any reasonable conclusions on the feasibility in terms of performance of our design we need to simulate its capacity. We will accomplish this by using identically configured systems behind a load balancing server to model what an actual implementation of our design would be. By scaling down the number of “users” to some multiple of what the current campus network actually sees, we can scale any results we find to determine what changes must be made to our original design. Since we cannot afford the equipment to test the core network infrastructure (i.e. Cisco 6509 switches, 7507 routers) we will focus on the service infrastructure which we can model using our existing hardware. In order to obtain fairly consistent, scalable results computers that are very similar in configuration will be used. To aid any conclusions we draw as to what hardware is required for a final version of the network, a performance index will be generated which can be used to compare our test systems to more advanced hardware.

Performance Index

Our performance index will consist of various benchmark programs run on our mockup systems as well as industry standard server configurations we have access to. The performance index will consist of file system benchmarks using the “Bonnie” open source utility (http://www.textuality.com/bonnie/), the Nbench Linux/Unix port of Byte Magazine’s Bytebench utility (http://www.tux.org/~mayer/linux/bmark.html), and network benchmarks on the server’s loopback address using the Netperf network test suite (http://www.netperf.org/). We will provide a direct comparison of these numbers to those of inexpensive high performance rack mountable servers in use on the streaming network of Veracast Communications (http://www.veracast.com/). If possible we will run the suite on other server systems to get additional metrics to determine what hardware we would need to support a given number of users on the final production network specification.

Testing Configuration

The servers will all be connected to a low-end Cisco switch, which is in the process of being acquired by a group member. Using this switch will allow us some flexibility in configuration of VLAN's and will ensure that the interconnection of the servers in the cluster will not be a limiting factor. Using this low end Cisco switch will give us a looking glass to compare our captured traffic results to a per-interface statistic provided by the switch operating system. Our model servers are pieced together from spare parts, but our actual cluster server components are identical hardware. They are Intel Pentium II 350 Mhz systems with identical SCSI disk subsystems, memory capacities, and operating systems. Other components have been purchased to complete any additional systems needed for application of "user" load, monitoring, and results gathering. We recently completed the construction of our two service providing cluster servers and 5 other supporting systems. All systems will be FreeBSD 4.7-RELEASE based as this is the operating system we would recommend for the final production network. FreeBSD provides a stable, high performance, free, and easily maintainable environment, but the production network could easily be a Linux based network if the IT department felt more comfortable working with that operating system. All services are standard open source programs and will run on either base operating system.

By using our two identical systems as the service providers we can validate our load balancing configuration and draw direct conclusions as to the number of final servers, at a given performance, we will need to support the current and future load on the Stevens campus network. Using dissimilar systems would force us to make assumptions as to the relative scale between systems in the cluster and the loads each system sees during the simulation. We plan to simulate 5-10% of the current campus network users as well as the maximum number of users these two identical servers can endure. By simulating an easy multiple of the current network users we can simplify the conclusion and extrapolation process once we complete our tests. The software we will be using includes the following:

IPFilter – http://www.ipfilter.org/

Thttpd – http://www.acme.com/software/thttpd/

Postfix – http://www.postfix.org/

Sendmail – http://www.sendmail.org/

Bind – http://www.isc.org/products/BIND/

Lukem-ftpd – ftp://ftp.netbsd.org/pub/NetBSD/misc/lukemftp/

Openssh – http://www.openssh.org/

Apache – http://httpd.apache.org/

We have and will continue to write contributing scripts to make system building simpler or to facilitate simulation. As we'll note later we will need to script our background traffic operations. This will involve a relatively simple shell script to start various instances of Netperf. Our current script is used to quickly get a server system on the network and install any required software packages. This will allow us to quickly rebuild our servers when we begin to simulate a new service. We will attach any supplemental scripts in our final report.

<u>Testing Philosophy</u>

The testing and simulation procedure must be carefully designed to provide a legitimate sample of the campus community. Due to the wide variety of network usages of different students and faculty, it is fairly difficult to obtain an accurate sample. Since we do not have the hardware needed to run every service in the Attila cluster at the same time we will test each service individually. While this will be indicative of the individual loads on each service cluster it will not represent the overall traffic passing through the load balancer and switching environment. In order to mimic those other types of traffic, whether background broadcast traffic or other service traffic that we are not currently testing, we will simulate each service separately and monitor the traffic usage of each. This will require a two pass testing approach, the first to gather the throughput numbers of each service, and the second to test those individual services again with the "background traffic" of the other services we are not testing at the moment. Background traffic will be generated by the "Netperf" application, and will provide us with the ability to monitor any additional latency introduced by other services. We will script several instances of "Netperf" to apply the basic patterns of background traffic seen from other services. Our goal is not to just supply a continuous stream of traffic, as this will not accurately simulate users performing specific operations on the network. Adding traffic in approximately the same mix as our baseline tests of each service will allow us to determine the effects of additional user interaction with the network on which ever service we are testing in a current simulation.

Potential Caveats

Given that the test utilities we are using are all standard open source programs we do not anticipate any major problems during the simulation. As of now the biggest hurdle we may run into is the entry of a large amount of users to simulate. Depending on how time consuming the process of entering fake user data is, we may scale our simulation down to a smaller percentage representation of the campus user base. While this may make our results less granular and our extrapolation susceptible to a greater degree of variance, it may be required to size down the simulated users if our first choice of users is too unwieldy. Another option may be to look into programs that generate user data quickly, or allow us to enter new users into the system quickly, and is something we are investigating.

Costs

Since most of our hardware is either our own or borrowed, our costs for the project are low. We will be utilizing the $250 allowed expense money to buy additional hardware for building client systems.

Conclusion

Our current timetable gives us approximately 1 month to complete construction of any additional systems we require for the simulation, create a user-base, run the baseline simulation, write and configure any background traffic scripts we need based on the results of the baseline run, run the final simulation, and finalize our results. The attached Gantt chart shows the work remaining for the rest of the semester. We feel we have more than enough time to complete all the tasks before us.