JESA

The USENIX Journal of Education in System Administration

Volume 2, Number 1 • November 2016

usenix
THE ADVANCED
COMPUTING SYSTEMS
ASSOCIATION

JESA

The USENIX Journal of Education in System Administration

Volume 2, Number 1 • November 2016

In This Issue

JESA Editorial Board

Chief Editors

Kyrre Begnum, *Oslo and Akershus University College of Applied Sciences*

Charles Border, *Rochester Institute of Technology*

Editorial Board

Melissa Danforth, *California State University, Bakersfield*

Æleen Frisch, *Exponential Consulting*

Guy Hembroff, *Michigan Technological University*

Erik Hjelmås, *University College of Gjøvik*

Scott Orr, *Indiana University—Purdue University Indianapolis*

Andrew Seely, *University of Maryland University College (UMUC)*

Niels Sijm, *University of Amsterdam*

Stephen Smoogen, *RedHat*

Steve VanDevender, *University of Oregon*

Charles Wiseman, *Wentworth Institute of Technology*

Erez Zadok, *Stony Brook University*

©2016 by The USENIX Association

All Rights Reserved

This volume is published as a collective work. Rights to individual papers remain with the author or the author's employer. Permission is granted for the noncommercial reproduction of the complete work for educational or research purposes. USENIX acknowledges all trademarks herein.

ISBN 978-1-931971-34-8 ISSN 2379-4798

usenix
THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

The USENIX Journal of Education in System Administration
Volume 2, Number 1 • November 2016
www.usenix.org/jesa/0201

INDUSTRY'S VOICE

Can We Make Progress in (System Administration) Education?

By Mark Burgess

A few years ago, I struck up a conversation with a woman at a cafe in San Jose, Silicon Valley, and she asked me what I did. Having been a physicist, then a computer scientist, and leaving a hard-won university professorship in computer science to start a company engaging with diverse areas of the business world, writing a popular science book, some music, and finally leaving company to write a series of abstruse theoretical papers about the functional semantics of space and time, I really had no idea how to even begin to answer her question. I wasn't even sure which country I was in. I gave her my card, and she emailed me some days later. "You're a teacher," she said. I was not expecting that, but I decided that it was a good answer!

My Teaching Transformation

Looking back over the years, I've spent a lot of my life in teaching, both in industry and academia. It began at university. I taught undergraduate tutorials as a graduate student, then took a summer job at an English secondary school, teaching teenagers science. After writing an early book on C programming, I taught C programming to Acorn computers (who later became ARM) and its clients in the late 80s, at the dawn of the 16-bit microcomputer era. After postdoc'ing, I taught physics, maths, and then computing to undergraduates at University College in Oslo. Finally, after 10 years as a lecturer, I started postdoctoral programmes in Network and System Administration, and had a go with graduate students, too. I left the university, looking to escape from its trials and limitations, and began to educate industry in the possibilities of applying the fruits of what I'd learned about IT infrastructure. The vehicle this time was what industry charmingly calls "productized innovation." So, however I see myself in the mirror, the title of teacher seems to be a fair cop, and I can even claim above-average experience.

Today, I have become a different kind of teacher: a relatively public figure, writer, and advisor. I advise the smallest and the largest IT companies on the planet about technology and self-reliant infrastructure, including the use of techniques like "AI," now very much in vogue. These companies span from the far East to the far West, where cultures differ enormously. When I began, I imagined I might be helping to push the boundaries of knowledge, developing new ideas, but actually the work has always been mainly about explaining the most basic notions, or points of departure. The first lesson of teaching is: everything is obvious once you know how, and nothing is obvious until you do.

So, like Joni Mitchell, I've looked at life (and clouds) from both sides now. Business and academia are two very different worlds, but each are facing the same kinds of problems. In IT, industry is winning.

I am a kind of consultant, but I am not a hired gun. In all of these endeavours, it has been important to maintain what one might call a code of ethics, or set of principles so as not to force-feed answers which I claim to know or which I think they want to hear, but rather try to help the people in these organizations to prosper by showing them what the actual questions may be, so that they might find the answers for themselves. That is inevitably shaped and coloured by my own thinking, but if I do the job well, that is only the disposable delivery mechanism for valuable content.

There are aspects of academia that are just like business, thanks to management culture. As an academic, one is encouraged to feed students simple commodity answers by government, as if there were a standard curriculum for life. This is a cheap approach to delivering the service of education. However, one-way delivery does not support continuous improvement, but the product may have bugs and defective behaviour. The short term costs are low, but the Return on Investment (ROI) may also be low. It is the application of commodity economics. Premium service options might include personal tutorials, individual attention, feedback, and the development of individually tailored

outcomes. Obviously the premium product is harder to mass produce, so it will cost more. But is the customer willing to pay? It's a race to the bottom.

This is all cheesy economic language used by business, but I would put it differently. I like to repeat a slogan that I wholeheartedly believe in: good answers are usually short-lived and contextual, but good questions last a lifetime. Questions stimulate thought, and they are always an opportunity to think anew, bringing new insight at best, and training the thinking muscles at worst. The need for deep thinking has never been more apparent than in system administration, where "how to" recipes and "best practices" dominate over general analysis. For years I have tried to understand why this is.

There Is Nothing More Theoretical Than Best Practice

In 1998, I was asked by an editor at John Wiley & Sons if I would write a textbook about system administration, based on a new course I was developing for the college in Oslo. At the time, Evi Nemeth's *UNIX System Administration Handbook*, and Æeleen Frisch's *Essential System Administration* were the state of the art practitioner guides for UNIX. Their focus was how to get things done on the diverse menagerie of Unices of the day. I wanted to do something complementary: focus on unifying and long-lasting principles rather than document the rapidly changing diversity. Moreover, I wanted to unify Networking with Systems (what today we would call "server" admin), which had radically different approaches to management, so naturally my book was called *Principles of Network and System Administration*. The book has been (by academic standards) only moderately successful in terms of sales, but fell far short of the popularity of the practitioner and experiential guides. It received excellent reviews and vehemently hostile reviews, polarizing readers like much of the other work I have done. I wrote a follow-up book for a graduate course called *Analytical Network and Systems Administration*, which people told me was "too hard—you should just give us the answer." I have been told "dumb it down" more times than I care to count by colleagues on both sides of the academic divide, but never by students, who are happy to be lifted up. This is a paradox. Students are hungry for knowledge, but are being sold cheap commodities by teachers?

Over time, those two books on system administration have made only a tiny impact on the world. They are not considered to be the "cool books" in the field, and even educators have largely rejected them in favour of books written by practitioners for practitioners. Teachers of system administration appear to want to pass on an experience: the same experience they had of being a system administrator, without questioning if there might be something better. System administration talks extensively about "best practices," though practitioners will argue fiercely about which practices are indeed the best ones. This is a game of conquest (oneupmanship) rather than of discovery. The unspoken rule could be that system administrators have to figure out knowledge amongst themselves, within the culture of tribal practice. Outsiders are not really welcome. When I was practicing system administration directly, I felt this too. More recently, I have often felt like an outsider, swimming against this stream.

Modern Tribes Versus Education

Is IT conservative in the face of rapid evolution? Around 2004, I wrote a very different kind of book called *Slogans*, in which I predicted that the ease of direct mobile communication by electronic devices would lead to a rise of tribal behaviours and fragmentation of societal values as people began to bypass their surroundings and speak only with the smaller circles on their speed-dials. Without belabouring the point, I believe we see this happening all around us, from the creeping politics of nationalism to our smartphone and social media contacts. System administrators were amongst the first to have access to these highly directed channels of communications, and it could be that this helped in the tribalization of their culture. It is no surprise that a group of former underdogs would resent outsiders, especially if it tried to peddle an imposed form of education. As one of the architects of an attempted certification programme around the turn of the millennium, I can attest that even insiders who tried to formalize training were met with scepticism.

Over the years, I have received both great enthusiasm and pretty ugly hate-mail from industry practitioners for pursuing my ideas. In his excellent book *Guns, Germs, and Steel*, Jared Diamond explains how human progress spread throughout the world, and why some countries or regions greatly outpacing others. World reaching empires

grew along side isolated tribes who never rose above primitive settlements. Jared points out that one can never really know whether tools, ideas, and innovation will spread (pay attention sales managers); some cultures are simply resistant to change, for cultural, ideological, or religious reasons. Some cultures live in abject poverty rather than accept modernization. After 25 years of interacting with the world of IT, I conclude that change has not been widely forthcoming because of both poor communication skills and cultural inertia.

In a tribal world, people look a leader for guidance. Today's role models for IT are the technology startups who pour millions of invested dollars into marketing campaigns to seduce IT practitioners. Open source commercialization has become like a major league of fan organizations masquerading as technology companies, each with their own private conference and merchandising. Business has hacked the IT community brilliantly. What could universities possibly do to compete with this big money? Universities have, in a sense, been hacked too, as they accept the popular tools and in the interests of "industry relevance." Materialism has bought tribal allegiance. Only an even more powerful allegiance could overpower this tradition within closed empires, and bring about changes like the cloud services: the rise of Google, Amazon, and the Internet's cool empires dazzle with their full-spectrum technology development and social impact.

Ivory Wombs

The point of academia is to distance itself from the day to day struggle in order to see the world through a wider lens, and then pass back what it has learnt. But what if no one is listening? Should academia simply be one of the crowd? If deep thought is valuable, why, then, is a principled i.e. "academic" approach to system administration consistently dismissed, in favour of a populist practical approach, even by teachers and academics? Do we have nothing to learn by stepping back? I think there are two plausible answers here.

The first comes from the deeply tribal, gamified culture of those drawn to IT. IT tribes identify with their work, their tools, and they like to ascend through the levels of the game by personal merit. In the 1990s, System Administration was considered an underdog in IT. Its special skills were not recognized at all by academia, they had to be won by hard experience. This fuelled a hero culture, with special interest groups and conferences deepening the inward looking. Having been the underdog, allegiances are naturally pointed inwards. Today's open source projects have continued this, tribalizing users into fiercely loyal followings. Which team do you support? Linux or FreeBSD? OpenStack or CloudStack? Puppet or Chef? The idea that universities might steal legitimacy, with an impartial (perhaps imperial) standardized curriculum, attacked this culture. A book written without attaching to an accepted tribal brand was not real knowledge from the horses mouth. As business became the primary employer of IT workers, this conflict between self-centric IT culture and the outward-looking needs of employers became an very public issue, called DevOps. It has tried to rein in moody individuals to become the service-minded collaborators that the world of commerce needs. To reach this, practitioners need to feel empathy for their users, like teachers for students. Not just pushing facts into a repository, but building a trusted relationship.

The second reason for failing to make an impact was related to a dilution of expertise accompanying the Internet's Big Bang. With the new millennium, e-commerce exploded onto the world with the World Wide Web, leading to a thousand-fold growth in the IT infrastructure over a decade. The older grey-beard generation of experts was diluted by the youthful if inexperienced enthusiasm of a new generation, learning by the seat of their pants. Many university students abandoned studies altogether for the offer of highly paid jobs based on only a minimum of skill. On-the-job training replaced education, for a while, and the industry never quite recovered from this skilled labour shortage. Many job tasks have been automated away, many systems completely reinvented without even knowing what came before, all in order to scale the knowledge deficit. Much of the knowledge of a generation was obscured and had to be rediscovered. When the tooling changes faster than you can publish a book, what is the point of trying to formalize? Google and Amazon (with more foresight than universities) took the unique approach of vacuuming up all the PhDs, but they bottled all the research activities in IT infrastructure behind closed doors, publishing and sharing their results only five years or more after filing for patents. This concentration of expertise resulted in the shift to "cloud" or infrastructure sold as a service. The problem of the tribal system administrator was bypassed, but the technical difficulties were just repackaged and pushed back onto users through APIs.

A Role for Universities? (Long-Term Thinking)

Universities are partly to blame for what I would call a serious lack of preparedness for the challenges of industry. As the custodians of society's knowledge, they ought to project a wider and deeper view. I am not talking about technology skills. Political pressures force educational institutions to teach what it thinks industry needs. So they look to the past, rather than leading the way or even looking ahead where industry is looking. Political influences have tried to apply market economics to education, making them conservative. Studies last year probably showed that 8 out of 10 businesses used Oracle databases; but is that the future? Statistics like these distort the view of our industry by focusing on the long tail of legacy rather than on the ripened seeds of change. Legacy technology changes much too slowly for the rapid evolution of IT. University curricula take years to adapt and approve, and teachers are already worked to the bone, leaving best practice courses hopelessly outdated unless they can seek a new mandate.

Where are the curricula about obvious long-lasting principles like system resilience, and scalable architectures taught? How about business alignment? Software developers today come to work for Internet giants believing that they can design software on a PC or laptop and a magic load balancer will elastically scale their application to a global market. When a system fails to behave well, our industry has no better answer than to call it a "bug" and pass the buck for someone to figure out the answer by random testing.

Across the industry, I see engineers following patterns designed for a different era, captured and bottled in training and certification programmes, and taught beyond their usefulness. Instead of asking hard questions, I see university programmes copying certification procedures with a certain righteousness, because who are they to abandon tribal ways? Surely industry knows best. Many of these patterns were long since abandoned by the likes of Google and Amazon, but are still very much the state of the art for industry practitioners.

What Industry Wants Is Not Life-Long Learning (Short-Term Thinking)

The knowledge gap in IT has allowed a few well-known Internet giants to go far beyond the realm of coursework provided by university educations. This has given them cult status. They even make universities (apart from Stanford and Berkeley) appear slightly foolish. The large data centres have become like the Large Hadron Colliders of IT—privileged environments that probe the very edge of a high-energy universe. Bachelor degrees lie far behind what began by necessity and became proprietary knowledge in Silicon Valley and Seattle. This trend weakens academia's credibility, and it can do little to compete.

What seems ironic is that many of the lessons relearned by datacentre engineers were, in fact, predictable by the kind of theory I tried to introduce into the field, and perhaps this was how problems were solved somewhere deep inside the belly of Google; but, the majority of industry and academia is failing in its responsibility to look in depth or ahead. We are used to waiting for a faster CPU to fix the bottlenecks and paper over the design issues by brute force. There is nothing harder to change than culture.

So what does industry want from students? In the language of business, business wants "people who can deliver." The language sounds as cheesy to academics as academic language sounds trite to business. The divide in communication and culture is as large as ever, even as topics like continuous delivery (actually the work of two science graduates), are directly analogous to the scientific method. Employers are looking for maturity and responsibility, enthusiasm, and an ability to work towards a goal, rather than cool kids with "mad skills." Business is mainly about execution of a fixed and conservative plan. They are happy to pay to make problems go away because it is a matter of survival. In IT culture, on the other hand, tribal warriors will happily work a hundred hours, unpaid, to avoid paying a penny to business, because of a distorted sense of gaming morality. (Richard Stallman was wrong. It is not about free speech anymore; it is about free beer.) The zeitgeist of DevOps rose to remind IT tribes that their self-inflicted tribalism can only be supported if someone makes the money to pay them in the first place. A respect for employers and employment could be taught. Employees need to cooperate, not only game their environments for their own amusement (even arguing it to be in the best interests of others). University does not seem to explain these economic realities of business to students.

The idea of life-long learning came out of schools and universities as a social concern. It is not a part of business ethics in a majority of firms. Business always waits until it is too late to innovate. Business thinks in short-term returns, not in long-term social prosperity. If you want to actually innovate, go and start a new company and hire new people, or buy someone else's company! What business wants is a strong work ethic and the ability to apply canned knowledge quickly and methodically to new sales opportunities. It also wants people who can explain what they have done in Powerpoint bullets of one syllable, so that they can ask for money from investors who are not experts. Universities are naive about these needs, and about teaching students relevant skills. Business does not care so much for the systematic, it prefers quick fixes (often to its detriment). It wants quality as long as this does not interfere with "time to market." Initiatives like continuous delivery, DevOps, and agile also take issue with these habits, but it has taken years of evangelism by a few prophets to make a small dent on industry practices.

Rebuilding Trust

What academics apparently ignore about business is that business works almost entirely by forming trusted relationships—skills are merely a means to an end. Skills are the icing on a pretty ordinary cake of human interaction. It took me a few years to see through MBA preaching about matrix organizations and customer satisfaction to what was really going on in business. Ultimately business is about trust. And both business and academia feel that the other has let it down.

Can something be done? Can trust be restored? The alumni of the elite universities, especially in Silicon Valley, are also tribal and naturally favour their own institutions and build relationships. If other educational providers are going to hack IT culture to educate and win trust (rather than merely train in obvious traditions), they need to recognize IT behaviours and adapt to its tribal ways too, forging their own strong identity. Teachers could show strength by bridging the ever widening gap between questions and answers, analysis and process. Business needs employees willing to see more than technology: employees who are willing to and who want to help others, not only win points in their own meritocracy. But they also need to prepare students for the end of this game, and the beginning of the next one.

At the time of this writing, there is a major discussion brewing about the future of work, the "next economy" and the coming of automation (the robots). All hype aside, a major shift is indeed coming that will shake the very foundations of our industrial society, changing once again the meaning of our lives. We will no longer be the machine of production, defined by our work. Commoditized dinners are replacing gourmet cuisine in technology as well as in fast food. This is the first step to a greater reliance on automation. We sacrifice uniqueness of systems for scaling to mass production. This too will change again in the future, as we enter the more automated era, and are able to focus on special needs once again. We will look back with amusement at the things that humans did manually, and how we suffered through the one-size-fits-all era of early cloud technologies. There will be a new market in "artisan system design" for smaller scale enterprises (think macciato, cortado, and frappuchino systems). New skills will emerge, but they will be more service oriented.

The reality is that the basic skills of a society do not change as much as appearances would suggest, but attitudes do. But we are limited and so new offerings tend to be like old offerings, only slightly modified. It may take many generations to get from hunting and gathering to hydroponic farming, or from plugging motherboards to gleaming cloud datacentres, from caterpillar to butterfly. The core may be the same, but the face of it is quite different and the effect on jobs can be large.

Universities could help to teach students about these challenges. What happens today is that academics still set aside principles of educating, and fall back to "skill training," imitating the industry practice of "ask no questions, learn the skill" for the present but not the future. Many employers look for people who studied humanities and learned IT by themselves, for their wider appreciation of human values. No one can capture the latest and still master the breadth, so narrowly trained graduates end up learning out-of-date tools, and end up reinventing new ones instead of building on past progress. The problem with training for an approach based on human skill is that the future does not hold much hope for this kind of work. Automation and "AI" will erode the need for it over the next few decades. The butterfly will hatch.

Closing the Mines

Some believe that software engineering will be the only skill required in the future. We should simply make everyone computer literate to stay employed. Should we be teaching five year olds programming? We've seen many efforts to expand the boundaries of IT tribes through "learn coding" initiatives. But programming alone is just another basic skill, a language for self-expression, like learning Spanish. How shall we teach new generations what to say? What are the challenges for a future society? If programmers and system administrators are trapped in a world no larger than the size of a screen, then computer programmers are little more than modern coal miners. Their world is narrow and replaceable. They may become tattoo artists, copying imagery imagined only by others and transferring it to a new medium. This too is replaceable by automation. Without the skill to think and question, society will be trapped in a rut, while a consumer growth economy steams on by. Caterpillars may emerge as butterflies, only to find that the world is now underwater.

Education (as opposed to training) is an uncomfortable anomaly in the industrial world. It does not fit the "mold and replicate" model well, so we struggle to scale it. It is about building a relationship to a speciality or a field, and relationships are expensive. Equally, we have not yet learned to fully live along side the surplus of information available to us that could lighten the load. We live in a vending machine age, getting anything we want at the push of a button. This is a world where pattern-repetitive labour stands to be replaced by automation. The ease with which we can access answers makes it difficult for students to appreciate the value of new questions, and knowledge as a relationship (not just a one-night stand). Could a university education still help them?

The business-university gap is not necessarily a bad thing, as long as both sides continue to stare at each other across the wall. Isolation from information is needed for deep thought. Invention, innovation in industry terminology, is a process with two parts: there is the mixing of genes (ideas) and then the a protracted isolated to gestate the germs of idea into a testable prototype. In the USA, academics move freely back and forth between industry and academia. Isolation can be a force for good. In the past, knowledge was isolated into ivory towers in order to give experts time to think and develop. Their separation from the day to day was an advantage. But it was more common for them to be connected to the world beyond out of hours. Today, the culture of business and universities has grown so far apart that their goals are not even aligned. Academia's incentives, twisted by governmental politics, have fuelled competitive hostility for grants between rival researchers, and more of a search for reputation than understanding. Universities look mainly inward, with a different system of values.

Looking Ahead, Choosing a Direction

Teaching (preparing a new generation for a productive and dignified life) is, remarkably, one of the least incentivized pursuits in our societies. We are geared to maintaining a status quo, even as technology punctures the equilibrium. It is itself treated as the kind of commoditized welfare work. That is regrettable, but it is also a natural progression, given the changes in social structure across the developed world. Since the industrial revolution, we have identified our lives (our sense of dignity and purpose) by our work (by our tribes). And our work has been largely repetitive productivity, fuelling a macroeconomics, whose success is measured by growth: "do it fast and do it more" seems to have become the depth of our aspiration.

I have made something like a career out of thinking ahead and playing agent provocateur. One could say that I am still a rogue hunter-gatherer who could never live with a tribe. I have swum against the stream, been tolerated, vilified, and eventually won a kind of endangered species license to be different. The reality is that people hire me, hoping for a quick answer, to bypass their own need to learn and think for themselves. I still refuse to do that, and luckily I have not been fired yet. I am not the only one. This tells me that there is hope. Change happens through persistence and socialization. But teaching, not just disseminating, is important. The relationship is crux of human society.

Our concept of knowledge has changed in the information age, but ultimately knowledge is a relationship, a repeated visitation of something or someone so that we learn to know that something as a friend. We increasingly trust others' experiences and recommendations because there are only so many relationships we can afford. New forms of social knowledge-passing have entered the field. Blogging has become a very important source of learning online. Blogs may

introduce ideas to a popular audience as well as demonstrate specific procedures and practices. They are a supplement to Wikipedia. They bridge a gap between isolated expertise and thought provoking discourse. I can envisage a day when national service draft papers call in retirees to teach younger generations of their experiences. What will be important in the future is how we socialize knowledge, in the new landscape of information and communication.

This sounds nice, but reality intrudes. What I see, everywhere I go, is that lofty ambitions get dialled back to make life easier for students and teachers under pressure. To be a great teacher, one needs to be passionate and singular of purpose. It costs a huge outpouring of emotional energy, which not everyone is willing to give. Goals become eroded as commoditization sets in. Scale that! We get microwave dinners in place of bespoke cuisine. This single idea is the generic challenge for humanity's future economy: to scale service without loss of individuality. After that, it's robots all the way down.

We Are All Industry, and Always Learning

The future of IT and education lies as much in the boardrooms of business as it does in academia. Business and university do not belong together; they serve different roles, but they can work along side. Universities should not expect donations, and business should not expect free labour. Both are usually struggling to make ends meet. Unless industry and academia can both think ahead, in a more societally responsible way, our economy is vulnerable to collapse, as outpaced humans are traded for canned automation, and redundancy brings unemployment, alienating an entire generation who have no plan for their lives.

The cycle of life needs both the quick genetic mixing of ideas and the slower gestation of the prototype, learning by selection on each cycle. We need to make time for education. It doesn't have to be given all up front. It could be embedded in every week of our lives. In IT, industry seems to have taken over many of the roles, leaving learning institutions out in the cold. Schools and universities are now more for the maturation of life skills than for learning and reasoning. But every cyclic process also needs to know history. The memory of what came before (and why) is the only way to become consistently better. History sets a standard.

Universities may well have to adapt to deliver more about the history and philosophy of their field than on the details of tooling, to become the GPS of learning rather than the big data fileserver, because one day all the answers could be old and withered, and only questions will remain.

The USENIX Journal of Education in System Administration
Volume 2, Number 1 • November 2016
www.usenix.org/jesa/0201

The Uptime challenge: A learning environment for value-driven operations through gamification

KYRRE BEGNUM, Oslo University College of Applied Sciences
STIAN STRØM ANDERSSEN, University of Oslo

This project aims to incorporate a specific perspective from the profession of system administration into a course taught at BSc level. The perspective is that of value-driven operation. Students are given access to a voluntary game which resembles a realistic scenario where they will manage a website given to them in a real cloud environment. The students earn points for the amount of time the website is kept up despite several automatic obstacles introduced by the game system. The goal is to let students reflect about the strategies that they use in order to optimize site resilience and how that leads to increased value generation.

1. INTRODUCTION

System administration is a profession that is still in its infancy world-wide. With society's dependance on IT systems, the role of the system administrator changes from "IT guy" to "guardian of vital civic functions". Becoming a system administrator does not require formal training or certification, yet it can be argued that in some cases their role is as important for society as nurses, teachers and law enforcement, all of which are professions with certification demands and corresponding education programs.

There are only a few undergraduate programs that focus on system administration. They are commonly formalized as computer science programs with topics belonging to the technical side of system administration, such as: operating systems, security, network protocols, scripting and configuration management. These courses fit naturally within the computer science umbrella, but given that system administration as a profession has a heterogenous knowledge-base, there are topics that are missing. Examples of such topics are: Organizational structures and leadership, communication skills, technical writing, economics, law and ethics. System administrators apply general computer science on a specific case which is the combination of IT systems and the organization depending on them. An education towards that profession should therefore encompass skills and knowledge that will make them successful in that setting.

Established professions, such as nursing and teaching, are characterized by a value system and identity that is adopted and internalized by the candidate as part of their education. It can be argued whether, at the time of writing, this identity is in place or even properly defined for the profession of system administration. Evidence can be found of activities that aim to identify this identity, but as a whole this process will require more time. One example was a workshop named "State of the profession" which was organized in conjunction with the USENIX Large Installation System Administration conference in 2012[USENIX 2012]. Here, the topic of lack of identity was lamented by the community, especially as it hurts cooperation with other professions and users in general, such as economists, developers and management. A lack of identity makes it hard to manage the expectations from the rest of an organization. In organizations such as LOPSA (League of professional system administrators) the topic on *what it means* to be a system administrator is still actively discussed in member forums. LOPSA also released a code-of-ethics that is a good example of an effort to represent the values that are idealized by professionals[LOPSA 2003].

In Limoncelli's *The practice of network and system administration* some chapters are devoted to what it means to be a professional system administrator, such as *Perception and visibility* and *Being happy* [Limoncelli et al. 2007]. In 2014, Kim et. al published the novel *The phoenix project: A novel about IT, DevOps, and helping your business win* used a narrative style to show a software project gone awry from the perspective of a system administrator thereby providing the reader with a unique insight into his value system and professional codex. Given the novels popularity and reception, there is reason to believe that it captures some of the spirit of the profession[Kim et al. 2014].

From an educators perspective, understanding the profession one is representing is difficult when the profession itself is so new and not well defined. However, through the process of defining learning outcomes, other educators have created models that make it easier to define and map the professional beyond their basic skills. One such inspiration can be found in Hardens work on characterizing learning outcomes for medical education. Harden creates a model of circles where the innermost circle describes *doing the right thing* followed by an outer circle of *doing the thing right* which speaks to the quality and standards of the application of the correct method. Finally, the outer circle is named *the right person doing it* which addresses the standards and value system of the individual [Harden 1999b; 1999a]. The same model can well describe the profession of system administration from a profession perspective. However, when we look at current education programs, only the innermost circle is the focal point.

One reason for the lack of holistic focus may come from the fact that there is little literature yet on what represents the outer circles of Hardens model that are written for the system administration professional. There are a few examples of chapters in books that deal with some of these issues, like mentioned above, but this is insufficient as a base for a course in any of these topics. From a teachers perspective, this is a challenge. We understand that there are certain unspoken truths in the profession that are not part of the student literature. This value-system, that is transferred in an unstructured way, either through co-ops or when the candidate starts working somewhere, is what educators want to teach in the same way the technical issues are taught. Harden's model is intuitive and simple enough to start developing descriptions of skills, behavior and attitudes that make up the profession so that values and their focus can be made more clear in the development of learning outcomes in system administration programs.

In this project, we focus on one such professional perspective from the profession that is rather new and usually overlooked. We will attempt to create a teaching environment for it so that students may learn it as part of their education. The concept of interest is that of *Value-driven operations* which is our own term for the concept. The idea is that the performance of an operations team is seen as a result of the amount of value it helps the organization to produce. One might be quick to assume that money is the obvious value, and this is certainly a strong driver for many organizations, but there are other examples. Consider volunteers working for a non-profit organization that has just had a major breakthrough in media due to a natural disaster or humanitarian effort. The operations team will increase the organizations value, if they manage to keep the web-based donation system running throughout the surge of donations that come in. Systems that deal with public outreach may have a higher value if they manage to remain accessible in times of crisis. An operations team also provides value, if it enables the developers to produce and put into production software features to their systems that either increase profit or drives competitiveness. In essence, any time an organization sets forth a mission, IT must be able to relate what they do and why to how it helps the organization achieve it.

Keeping a value-driven-operations mindset means the following:

— Understanding their ”value” in what the organization strives to achieve
— Identifying the processes in the organization where IT is involved and their impact on value
— Enabling continuous improvement to the processes so that value can be achieved more often, effectively or predictably

Knowledge about value-driven operations provides an alternative perspective to how IT works in an organization to the more traditional views of striving for security or achieving the highest possible performance. There is no ”best” in these perspectives. In our view, *the right person doing it*, to use Harden's phrase, is one who can balance all of the perspectives in the light of a particular context. Security and performance are topics often discussed while value-driven operations is virtually absent.

2. ORGANIZATION OF EDUCATION

We wanted to create a teaching environment that had the following attributes:

— Increases student motivation
— Enables self-assessment and self-regulation
— Creates engagement and foster discussions in class
— Provides a direct feedback loop between the students actions and value creation
— Has industry relevance

In order to address the first item, we want to use the concept of gamification. Gamification means incorporating learning activities through games and can increase motivation by providing more fun and engaging ways to work with the subject.

The game in this case will be in the form of a simulation, were the students are divided into groups of two. Each group represents their own company that is in charge of their own web-page. The company also makes money as long as the web-site is up and can be used. In other words: from an operations perspective, the value of the organization is tightly coupled with the uptime and resilience of the web-site.

The groups will get the web-site as downloadable software and have to make the site run in their environment. They are not allowed to change the code of the site itself, a typical scenario for the business world. When they go ”live” with their site, traffic is sent to the site creating a sense that something is happening and is the beginning of their simulation.

However, things happen to their site that may push the site into a faulty mode, at which point the company loses money. Bugs in the code, misconfiguration and outright automated sabotage will challenge the students to come up with architecture designs and develop their own tools that counteract these. What they spend their time on improving has to be constantly balanced based on how much more money they will make with the improvement and what they should do first.

Throughout the simulation, the amount of money earned is visible to all students at all time. This means that while they work, they can see the effect of their (and others) efforts. For example: a group decides to have multiple webservers in order to increase the redundancy of their infrastructure. After the task is finished, they suddenly realize that one of the two webservers has been shut down by the automated sabotage system. However, they note that they didn't take any damage financially as the other webserver kept serving webpages throughout. They discuss what happened and decide to also create a tool that can automatically starts servers that are shut down so that the site can grow back to it's intended state without human interaction.

The simulation is close to real life as development and operation are traditionally two separate groups with distinct goals. Designing a scaffolding of robustness around a shaky software solution is unfortunately a common situation and one that candidates should be prepared to tackle. Furthermore, the advent of cloud platforms such as Amazon AWS, has led to the deployment of massive services such as Dropbox and Netflix that run on clouds that do not guarantee for the safety of their virtual machines. This increases the need to create an infrastructure that can generate value through resilience.

3. THEORETICAL BACKGROUND

For profession-based educations, the balance between theoretical subjects and practice is an important and reoccurring subject. Programs curricula are often torn between vocational skills and abstract theories. Good arguments exist for either perspective, but the challenge is the continuous balancing of the two in light of changes in the field. Decreasing the gap between theory and practice for students is known as creating *coherence* in the program[Hatlevik and Smeby 2015].

A high level of coherence means the students experience a strong connection between theory and practice. For system administration education, this balance is often found in the combination of theoretical lectures, for example networking protocols and operating system concepts, and lab exercises such configuring a network and installing operating systems respectively. For the case described in this paper, we aim to create a similar experience. Discussing value-driven operations in class is highly theoretical and unless there would be an environment for experiencing it first-hand, the students might have a low coherence as the topic is not joined by a practical application.

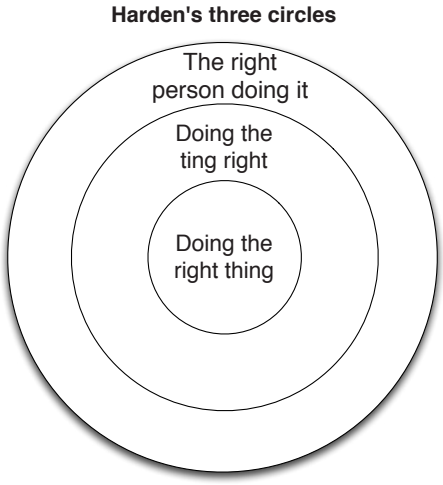

Fig. 1. Harden's three circles. The innermost circle defines what skills we would expect of a professional. The second circle speaks to the ability to apply the skills in the right manner given a particular context. The third circle represents the values and practices that drive the decisions and actions of the professional.

Hattie et al. argues for the use of feedback mechanisms as a method to increase the output of a teaching methodology[Hattie and Timperley 2007]. In their model, they argue that feedback can be given on four levels: Task level, Process level, Self-regulation level and, finally, Self level. The effectiveness of the feedback is relative to the level it is given on. For example, feedback on the self level, like "you are a good person" does not increase performance as much as feedback on the lower levels. In the framework we propose, students will get immediate feedback on their tasks through the automated scoring system. If they do a good job implementing a technical task, the score will automatically continue to increase while the opposite will happen if they make a mistake. According to Hattie, this is a common used strategy, however in our case it is automated so that it does not depend on a teacher or assistant to provide the feedback. Feedback of this kind is the most effective when it is coupled with information. When lack of information is the cause for a mistake, simply pointing out the mistake will not help the student unless the information is also made available. That aspect is not entirely covered in our solution. Students will see some additional technical performance metrics that can help them interpret their own results, but they need to have sufficient knowledge to use them as feedback.

Process level feedback enables the students to understand how they handled the process of conducting the task. How did they troubleshoot? How did they plan what to do? How did they test that their solution was working? This type of feedback is quite important as it focuses not only if the task was successful, but wether the approach was good or not. This resembles the two inner-most levels of Harden's model. "Doing the right thing" is related to successfully completing the task. "Doing the thing right" means completing the task in an effective and stable manner. For example: One typical failure will be that a random server will be shut down by the sabotage system. Successfully completing that issue is to simply boot the server up again. That will be the *right thing* to do. However, as this happens at a regular interval the students will not have solved that problem in the best possible way. They will have to spend the same amount of time repeating that step every time a server is taken down. It is a short fix, but over time it drains time. Dealing with that issue *the right way*, is to invest time in writing code that automatically boots servers that are shut down, thereby eliminating the need to handle such issues in the future. This type of feedback is present in our solution, but it is not obvious. This type of feedback is only available over time, when the student experience that dealing with reoccurring issues with automation and redundancy will result in less work in the long run.

Using games as a means to motivate students and increase learning has become a popular tool in educations through the increase of more digitized learning environments. One example is the work by Hembroff et. al, where they create a learning environment based on a game in a security class [Hembroff et al. 2015]. The difference in their approach is that the game contains a story and levels and will work on its own. In our case, the game takes place as their environment is *exposed* to the typical climate of a unstable solution running in a unstable environment. Furthermore, it requires them to do the normal assignments in the class in order to have an environment to work with. So we essentially play with their lab environment.

4. UPTIME CHALLENGE SPECIFICS

The site the students were to manage was a social network type solution written in PHP with MySQL as the database backend. Each student group is a company that will make money based on the uptime and availability of their social network. If the site should be down, the company will suffer financially and loose money. A growing balance is therefore a literal indicator that more value is generated as an effect of their work as sysadmins. Achieving their company's mission means to maximize reliability and uptime despite a growing user-based and other obstacles. At numerous occasions, they will face problems that challenge them to either do the right thing or the thing right, as Harden would put it.

The code for the site was made available through Github. The site was relatively simple, with users, posts and comments. The site was not meant for actual human interaction, but through scripts that were executed from a simulation engine which will be described in further detail later. The site existed in several versions, each with particular bugs and characteristics. For example, version 1 resulted in several PHP error messages in log files at every page visit. This would lead to a full disk and slow performance once the site grew larger. Another version had no cap on the amount users listed on the frontpage. As a result, as the user-base grew, the landing page got increasingly larger.

The students were instructed to use a particular version well knowing that it had some challenges. At later stages, we simulated upgrades in that students were allowed to use the next version available. The storyline was used that the software was developed by consultants so they had no option to go and modify the code themselves, they had to wait for the consultants to develop a new version first.

The course covers topics such as caching, loadbalancing, database replication, scaling and backup. At every topic, a new component is added to the infrastructure so that what started as a single webserver with a database server ends up as a site with several webservers, redundant load-balancers, replicated and redundant databases, caching servers and backup servers. Throughout this weekly evolution, the simulation engine kept using the site non-stop and increase its users, posts and comments. As any start-up, they had to grow while the site was active. Every site started with a baseline of 50 users, but most groups ended up with more than 30.000 users and over a 100.000 posts.

The technical infrastructure used was an OpenStack cloud. One goal was to be as realistic as possible with the technical setup. OpenStack and Ubuntu virtual servers were a technology base that had a high transferral value to the real world. Each student group had it's own tenant in the cloud with resource quotas associated with it. Using this platform, we could build our own simulation framework into the same cloud and we also gained access to their virtual servers. The cloud allowed us to track the resource usage of each tenant and this was also included into the simulation. Each company had to ”pay” for how many resources they used, so the most payoff was gained from the leanest yet scalable solution, closely mimicking the real world.

4.1. Simulation engine

The simulation engine was responsible for several actions:

— Constant usage of the students sites (landing-page visits, new users, new posts and comments)
— Measuring site availability and response time
— Award/deduct money to each company based on availability and response time

— Sabotage the companies sites by randomly shutting down students servers
— Deduct money based on the cost of a company's infrastructure

The students sites usage is based on 24-hour profiles with traffic variation both based on time of day and with an additional noise factor to allow for small spikes. The amount of usage is calculated by the simulation engine for 5-minute intervals and sent as a job to a group of workers. An example of one such job would be that Company X got a sustained rate of 7 landing page visits per second for the next five minutes. Traffic would peak around work-hours and be low otherwise. In order to reduce the pressure of the job workers and cloud overall, each company existed in a different time zone so they would peak at different times during the day.

Each company's site was checked for availability every five minutes. If the site loaded successfully and a particular text was found on the page (a text that would only be there if the site functioned properly i.e. the database etc. worked.), the company was awarded a base amount of money. An additional bonus was awarded in case the site loaded in less than 0.5 seconds. The bonus declined rapidly in case of a slower loading time. If the site was available, but did not work probably (typically the text was not found because the database was down, or the students had redirected traffic to an emergency page), then a small fraction of the base amount was awarded. If the site would not answer at all, then the base amount would be deducted from their balance instead.

Inspired by Netflix's Chaos Monkey[Tseitlin 2013], we also introduced sabotage to simulate a unpredictable infrastructure. The sabotage was relatively benign in that it did not destroy anything the students had done, but merely shut down virtual machines. Every two hours one random server belonging to each company was shut down. The effect could be dramatic if the company lost their single webserver in the middle of the night. Doing the right thing, would be to start the server again in the morning with the losses that would entail, but a group that had either invested in writing a small script that checked for downed servers would maybe only get a very small penalty. If they had redundant webservers, they may not get a penalty at all. These are examples of putting Harden's three circles into practice. The two latter strategies would mean doing the thing right and with a small investment of time save time later and increase value. For convenience, the sabotage was disabled during lab hours to not interfere too much while students were working actively on the course. On some occasions, the sabotage was more manual. For example, at one point we logged into all loadbalancers, shut down the loadbalancing service and removed the configuration file so the service would not start again directly. These types of occurrences were intended to facilitate troubleshooting.

Money deduction based on infrastructure size was based on the same principles as Amazon AWS: you pay for uptime factored by the size of the virtual server. The main difference was that the cost was calculated every 5 minutes and only for each server that was in fact up in that period as opposed to an hourly charge regardless of system uptime in AWS. This was to encourage scaling and a more dynamic system.

4.2. Simulation infrastructure

Development of the simulation architecture was completed as part of a MSc project. Our goal was to create an extensible framework were several features could be added at a later stage and could potentially serve other use cases in different courses at the same time. For example, we envisioned that in the course there would be periods of increased traffic, due either to a trending topic or campaign. During these periods, the traffic would be scaled up and the potential risks and rewards likewise. So we wanted to create something where these types of additional components could be added at a later stage.

The architecture was based on the idea of individual service managers for each sub-component utilizing a common framework of database access and job queues. The framework itself had an API available to all sub-service managers in order to interact with the data layer. A manager would be in charge of one set of actions, for example, deciding the amount of traffic for all sites at any point in time. The manager would create jobs that are sent out on a message queue where corresponding

workers waited for jobs. The managers would also interact with the database layer. In some cases for information retrieval, like fetching the corresponding IP address for a companies site, or information changes, like awarding more money. Each individual worker job ended up with a report that was stored in the database. This enabled students to look at the history of actions as well as us to find evidence of student activity.

During the run of the course, with 15 student groups / companies, we used 15 virtual servers hosting job workers and three other servers with database, API and managers respectively. Using message queues, the architecture could scale based on necessity and our desired traffic workload.

Students could get feedback on their progress through plots and score listings that were made available to them and were continuously updated. During the course, some students wrote alternative dashboards that had better visualizations and graphics. One student even wrote a mobile app to track the state of the site and its current company balance based on our data. Students also set up their own monitoring and in one case there was a friendly dispute where our system had flagged a site as down while their data said it was up. All of these cases were strongly welcomed by the staff and introduced not only interesting problems to discuss in class but also was interpreted as signs of student engagement.

5. CASE ASSESSMENT

Due to the nature of the course, there is no option to conduct a test / control study in order to establish the success of using this uptime challenge. Even though conducting a comparative experiment can be valuable in a statistical setting, its practice in an educational setting has been criticized by Freeman et. al [Freeman et al. 2014] in cases where there is strong reason to assume that it will be of benefit. They point out that if there is a underlying understanding that one teaching method is more advantageous to the student, offering it only to half of them in a test / control setting is unethical towards the students in the test group, where such methods are not used. Furthermore, as the game will go on for most of the duration of the course, it would mean that in a comparative study, half the class would witness as the other half would be having special discussions and participate in a game that is meant to be fun. Thus, the decision was made to let all students partake and rather measure learning on all students throughout the course.

The data collection was done in the form of questionaries conducted in the beginning and end of the semester as well as reviewing work they hand in during the course. The questions were originally asked in Norwegian, but have been translated to english here for the sake of this text.

Throughout the course, the data traces from the management system itself will be available for analysis. This enables us to track students engagement by observing the degree of points scored throughout the semester.

6. RESULTS

The uptime challenge was utilized in the course ”Database and application management” at the Norwegian Technical University during the spring semester of 2016. The course is a mandatory course for the BSc in Network and System administration as well as an elective course for other computer science students.

As the class started, the initial survey was answered by 25 students of which nine of the students had some prior industry experience. 8 of the students had this course as a mandatory course, the rest took it as an elective. 19 of the students (76%) agreed or somewhat agreed to the statement ”I am motivated by working on course activities outside the mandatory tasks.” When asked about what activities they believed were most central to their own learning, being at lectures, trying out things for themselves and working together with others were the strongest ones reported.

The survey also asked the students to explain how they understand the term ”System administrator”. Almost all students wrote a definition that relates either to being ”the person responsible for all the systems” or listed specific tasks, like ”In charge of securing the infrastructure” or ”Monitors that everything is up”. None of the answers were connected to the contribution that the role has to the a companies mission. However, when asked about what additional properties besides being

”secure” would make a solution be considered as ”good”, the students focused more on the softwares role, like ”employees can work more effectively” or ”User friendliness”. Furthermore, when asked in what ways they believe an IT solution can add value to an organization, the vast majority of the answers centered around the increased effectiveness of the users and the resulting potential for money savings.

Overall, the data suggested that most students had a technical and low-level attitude towards IT systems and being a system administrator. They did not see a system administrator as someone directly connected to a company's mission, but rather someone who keeps the lights on. This was expected as this is a prevailing attitude towards the profession.

After the course, we conducted a survey where we got 23 replies. We asked the students to what degree they agreed to the following statement: *The uptime challenge motivated me to spend more time on the course.*[1] The alternatives were on the following scale:

— *Do not agree*
— *Somewhat disagree*
— *Neither agree nor disagree*
— *Somewhat agree*
— *Agree*

All but one student either agreed or somewhat agreed to the statement (11 on each). Furthermore, the students were also asked wether they agreed to the statement: *The uptime challenge game motivated me to use more time on activities related to the course that were not directly to the uptime challenge itself.* Here, 16 students either agreed or somewhat agreed, while five neither agreed nor disagreed and two somewhat disagreed.

The students were also asked the same questions from the beginning of the course, as to how they understood the term ”System administrator”. There was no particular change from the beginning of the course, but further inspection showed an increased perception of what the role meant, rather than what the role does. Of the 23 respondents, 14 used the term ”responsible” or ”responsibility” when describing a system administrator. Only seven students used that term in the first survey. There were less examples of tasks in the end-survey than in the first and more on achieving service availability and ensuring effectiveness of workers. This may indicate that the opinion of the system administrator changed towards the effect a good system administrator achieves in the organization rather than the particular tasks they do.

6.1. Tracking individual groups performance

Overall, the data traces from the worker reports showed that the majority of the groups earned money most of the time. In order for this to happen they have had to implement automated mechanisms to recover from the sabotage, which were not mandatory tasks given to them. In addition, they had to compensate for the growing size of the site by adding more capacity to their infrastructure. However, we found examples of groups that did not attempt to optimize value. They would normally boot up all the downed servers in the morning of the lab-day and then start loosing them again the day after when the sabotage started again. However, no student voiced direct negativity towards participating in the challenge. Also, it was not investigated if students wanted to participate but did not have the time to do so. We also do not know if they attempted to improve the robustness but were simply unsuccessful in doing so.

As an example, we present the distribution of two groups that had different performance profiles. For simplicity, we call them group A and B. Group A did not implement any automation and as a result ended up with low rewards and often deductions. From the distribution plot we can see how the distribution is almost uniform. The reason the rewards are often higher is due to the added bonus for response time. Group B took great interest in the challenge and often experimented with

[1] The questions in all surveys were in Norwegian, but have been translated to fit the language of this paper.

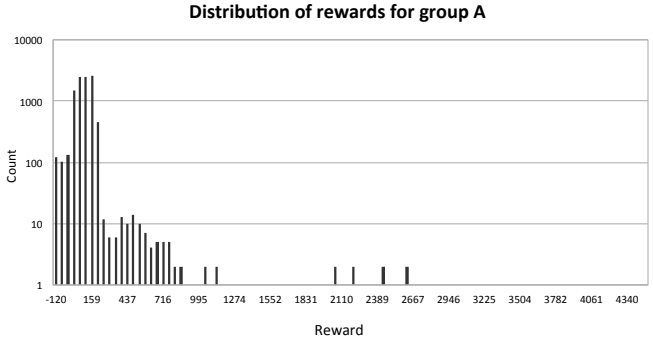

Fig. 2. The distribution of rewards for Group A. Without the right mechanisms in place to ensure robustness, deductions became common.

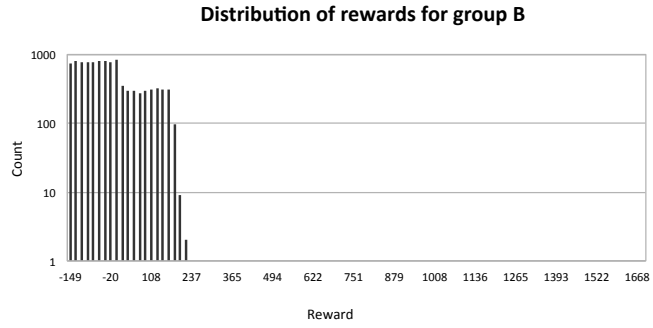

Fig. 3. The distribution of rewards for Group B. With time spent on counteracting sabotage the right way and a focus on optimizing performance, the rewards were plenty.

alternative technologies to squeeze the most performance out of their site. As a result, one can see from the distribution that they achieved almost entirely high rewards.

7. DISCUSSION

From the results of the survey we can argue that the uptime challenge was something that motivated students to show effort in our course. This aligns with our impression from teaching also. The fact that several students developed their own tools and apps to better get a notion of the current financial ranking and status of their company, tells us that the challenge was engaging. The uptime challenge created a storyline and scaffolding on which most topics in the course could be pinned.

7.1. Achieving a higher-level view of a system administrator

Our initial main goal was to connect some more abstract ideas about value creation into a framework that let the students experience how it relates to actual technical tasks and strategies. Throughout the period, and based on the predominance of the students low-level view of what the term system administration we started to ask ourselves if our uptime challenge will also influence their perception towards a more high-level view, like that the system administrator is essential in making sure the IT infrastructure helps achieve the organizations mission. Given the increase in the word ”responsibilty” one might be hopeful of that, but it was not explicitly researched.

We see, however, how we could tailor the challenge to include more organizational thinking and interaction. For example, the students could have opportunities to argue for increased budgeting and resources. We could also have specific goals that had to be achieved at a specific point in time, like a new release or campaign by their company, where uptime would be of particular important to the companies mission.

7.2. Availability of software

All the software written that is part of the uptime challenge is available as open source. This includes the web-site used by the students and the simulation infrastructure. The tools are continuously improved to help teachers easily manage student groups in the system.

However, there is certainly a cost associated with the uptime challenge. As will most lab environments, maintaining the lab often times falls on the teacher and a lab engineer. Our experience was that setting the system up in the beginning of the class was the most time consuming period. Once the system was up and groups were added to it, it was mostly a matter of scaling for optimized utilization. Through the use of Docker containers and the puppet configuration language, scaling was not a time consuming task.

Being dependent on a cloud or other virtual infrastructure is a potential limiting factor. We have not investigated how one could split the solution so students could run the challenge on their own computers. We definitely see the potential value for this in terms of self-learning and this could be further investigated.

8. CONCLUSION AND FUTURE WORK

This project wanted to create a lab environment and simulation that enables students to reflect over the relationship between their operations work and the value that is generated by the managed solution as a result of it. Our implemented lab environment is a adaptable solution that can work for several different types of internet-based sites as cases and has a high degree of customization. The environment was tested in a course and surveys at the beginning and end of the course were used to track students satisfaction and attitudes. Our results show that students considered the environment to be engaging and motivating. Most students took advantage of the opportunity despite it being an optional activity in the course. Future work will focus on making the solution adaptable and usable by other institutions and investigate if a self-study solution can be developed from it.

REFERENCES

Scott Freeman, Sarah L Eddy, Miles McDonough, Michelle K Smith, Nnadozie Okoroafor, Hannah Jordt, and Mary Pat Wenderoth. 2014. Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the National Academy of Sciences* 111, 23 (2014), 8410–8415.

MH Davis M. Friedman RM Harden, JR Crosby. 1999a. AMEE Guide No. 14: Outcome-based education: Part 5-From competency to meta-competency: a model for the specification of learning outcomes. *Medical teacher* 21, 6 (1999), 546–552.

Ronald M Harden. 1999b. AMEE Guide No. 14: Outcome-based education: Part 1-An introduction to outcome-based education. *Medical teacher* 21, 1 (1999), 7–14.

Ida Hatlevik and Jens-Christian Smeby. 2015. Programme coherence and epistemological beliefs. *Nordic Psychology* ahead-of-print (2015), 1–18.

John Hattie and Helen Timperley. 2007. The power of feedback. *Review of educational research* 77, 1 (2007), 81–112.

Guy Hembroff, Lucas Hanson, Tim Vanwagner, Scott Wambold, and Xinli Wang. 2015. The Development of a Computer & Network Security Education Interactive Gaming Architecture for High School Age Students. *The USENIX Journal of Education in System Administration* (2015), 25.

Gene Kim, Kevin Behr, and George Spafford. 2014. *The phoenix project: A novel about IT, DevOps, and helping your business win*. IT Revolution.

Tom Limoncelli, Christina J Hogan, and Strata R Chalup. 2007. *The practice of system and network administration*. Pearson Education.

LOPSA. 2003. System Administrators code of ethics. https://www.usenix.org/sites/default/files/code_of_ethics_poster_english.pdf. (January 2003). [Online; accessed January 2016].

Ariel Tseitlin. 2013. The antifragile organization. *Commun. ACM* 56, 8 (2013), 40–44.

USENIX. 2012. Workshop review: State of the profession. https://www.usenix.org/blog/workshop-review-state-profession. (December 2012). [Online; accessed January 2016].

The USENIX Journal of Education in System Administration
Volume 2, Number 1 • November 2016
www.usenix.org/jesa/0201

Software Defined Networking for Systems and Network Administration Programs

Bruce Hartpence, Rochester Institute of Technology
Rossi Rosario, Rochester Institute of Technology

Academic programs can be very successful when they include industry best practices, innovations and techniques in addition to theory and background. This approach has historically been a tenet of the networking and systems administration program at the Rochester Institute of Technology. Software-defined networking is an excellent example of a technology which combines theory and emerging practice. Software Defined Networking or SDN includes components that stretch across networking and systems administration curricula including servers or controllers, virtualization, OpenFlow enabled network elements, communication pathways, opportunities for automation, telemetry from the network, dynamic response to system demand and more.

These characteristics, and because SDN experiments and courses can be implemented in either virtual or non-virtual facilities, make SDN an outstanding platform for teaching the principles of network and systems administration. Graduate students can also take advantage of the environment encompassed by SDN topologies to further their understanding of systems design, management, testing and communication protocols. This paper will describe some of the SDN projects run at the Rochester Institute of Technology (RIT), the impact on curriculum and some of the environments used. The challenges associated with running the projects and courses within a lab environment will also be illustrated. How and why many of the ideas and new industrial developments were integrated into the classroom will be central to the ideas presented.

1. INTRODUCTION

Understanding what is meant by the term Software Defined Networks (SDN) is not always a simple task as the term has been applied to so many different ideas. For the purposes of this paper, Software Defined Networking will be used to describe an innovative approach to network construction, configuration and monitoring. While it is relatively new (most of the ground work was completed in Stanford circa 2008) it is transforming our thinking regarding communication architectures [McKeown et al. 2008]. SDN is a novel network architecture in that it provides a segmented design where the controlling entities of the network (control plane) are detached from the forwarding functionality (data plane), allowing for efficient and granular control over the data flow. As result, a logical centralization of the network is possible. This centralization is in the form of a controller that communicates with network elements which can simplify network management and improve visibility into the network. One improvement is that SDN provides a much more dynamic and informed view into the network via flow tables. SDN also provides a mechanism through which networks may be shared among different tenants, each having their own control over network switches and routers [Sherwood et al. 2010]. An important point is that the controller may be able to dynamically make changes to network elements based on feedback or code without human intervention.

These improvements are difficult to achieve in traditional networks. Traditional networking and systems administration refers to the static and laborious techniques for designing, managing and deploying network elements, servers and services. Examples include the need to interface with each element manually, attempting to determine traffic patterns via telemetry from servers or protocols, deploying virtual machines and applying security mechanisms.

SDN networks are logically segmented on three general regions: Application layer the management plane where the network programming portion takes place, Control Layer in which the controllers or network brains reside, and finally the forwarding devices or Openvswitches (OVS) - the nodes that execute the forwarding decisions according to the flows rules dictated by the controllers [Kreutz et al. 2015]. This is ilustrated in Figure 1. This concept breaks with traditional paradigms of network design and operation, opening up a whole new path for research exploration.

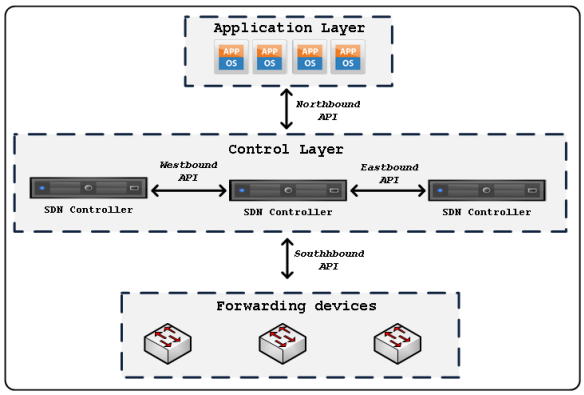

Fig. 1: SDN segmentation

The communication channel between the controller and the network elements (southbound interface) is made possible via OpenFlow protocol [Specification 2011]. OpenFlow is the first standardized communication interface for programmable networks. It was envisioned to provide an open and programmable platform in which flow table entries could be inserted and modified easily, without worrying about vendor specific configurations [McKeown et al. 2008]. Although it started just as the research vehicle for programmable networks, nowadays, most of the controllers support this communication protocol. Figure 2 depicts the OpenFlow channel between the controller and network element. SDN provides an excellent opportunity for educators to

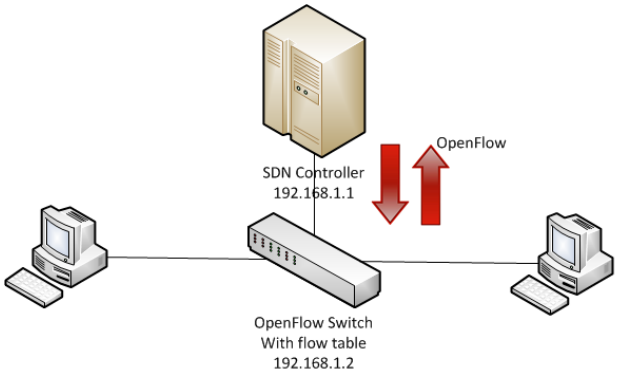

Fig. 2: OpenFlow Channel

bridge theoretical concepts and in lab experiences. This is not only because of the nature of SDN architectures but also because SDN (or the idea of being software defined with a controller) has expanded into areas such as storage (SDS), wide area networking (SD-WAN) and data centers (SDDC). SDN also leverages several other technologies such as virtualization thus integrating virtual and traditional networks. The separation of control and data planes are well documented in the literature as are the problems associated with traditional network design and operation. These problems extend into the administrator space because historically servers and the network have been unable to respond dynamically to changes or provide visibility into processes. With the creation of open source controllers, such as Pox, and the the ability to virtualize elements of the topology, these problems and their potential solutions can be deployed and observed in the lab.

While controller software such as NOX or POX [Gude et al. 2008] is open source and stable, making it relatively simple to work with, it does not represent the current state of SDN research or industry. Fortunately, the SDN structure is modular which allows the educator to easily shift from one to another. This is demonstrated in one of the projects described herein. Thus it is possible to integrate the latest changes from production work. In addition, central components such as Open vSwitch [Pfaff et al. 2015] can be held at a particular patch level or embrace the latest developer modification. The projects described in this paper will not only discuss their construction and results but some of the challenges associated with the management of resources. These challenges vary when the architecture is constructed on-site as opposed to in a remote facility such as the Global Environment for Network Innovation or GENI [Elliott 2008].

One can make the case that the software-defined approach can be applied to network and systems administration programs by simply looking at the components and skills necessary to deploy the architecture. However, innovative and practical use cases can be found in industry as well. Facebook is an organization that is willing to share some of its project work. The Facebook network, database and infrastructure teams were faced with not only complex problems associated with performance and network traffic but massive log files reporting hardware and software failures. To address these challenges, Facebook engineering created the Facebook Auto Remediation service or FBAR [Hoose 2011]. FBAR has the ability to automatically run code in response to these failures. The result is that the system adopts a software defined automation approach that saves on man hours, time and money. Examples like this show the interrelated nature of problems and how the software defined approach can used by multidisciplinary teams.

The rest of this paper is organized as follows; section 2 provides some background on standard network/sys admin coursework and how SDN integrates into these areas. Section 3 discusses SDN components and the lab environment used to support courses and labs deploying these topologies. Section 4 is an overview of two major projects run at RIT, their results and challenges. Section 5 will be a discussion of our experiences and the potential for related courses and labs. Section 6 concludes.

2. SYSTEMS/NETWORK ADMINISTRATION CURRICULUM TOPICS

As SDN is a relatively new topic, it is fair to question where it would fit and why it should be adopted. Information Technology (IT) and related programs commonly contain courses covering key ideas and best practices for the industry into which graduates will soon enter. The goals of this coursework include preparing students for their careers and instilling the ability to be lifetime learners and solve problems. That is, to be able to grab new technologies or ideas and incorporate them into current and new generation architectures.

RIT is no different. Having a core of networking and systems administration courses, we have also added courses that embrace trends in industry and capture important methodologies. SDN provides an excellent opportunity to further these ideals, especially when combined with virtualization. A brief description of our foundation courses follows. While these may read like a section of the course catalog, the shortened descriptions are included here to indicate what we consider to be core tenets of the curriculum and the additions necessary to keep current and provide a vibrant learning environment.

— Task Automation Using Interpretive Languages - An introduction to the Unix operating system and scripting in the Perl and Unix shell languages. Includes basic user-level commands, control and data structures in Perl.

— Systems Administration I - This course is designed to give students an understanding of the role of the system administrator in large organizations. The technologies discussed in this course include: operating systems, system security, and service deployment strategies.
— Systems Administration II - This course goal is to help students understand the roles and responsibilities of system administrators in modern enterprise organizations. The responsibility of the System Administrator is to maintain, manage, secure and monitor the network and the services it provides.

These first three comprise the core of the systems administration courses with the next group describing the network based coursework.

— Intro to Routing and Switching - This is an introduction to wired network infrastructures, topologies, technologies and protocols required for effective end-to-end communication. Basic security concepts are also introduced at the local area network communication level.
— Wireless Networking - This course is designed to provide the student with an understanding of the protocols, principles and concepts of radio and optical communication as they apply to wireless data networking for local area networks and peripherals.
— Network Services - An investigation of the tasks of selecting, configuring and administering services in an internetworking environment. Topics include the TCP/IP protocol suite, service administration including DHCP, DNS, SSH, Kerberos, and an introduction to directory services.

These brief descriptions mirror what is seen in many programs and the courses have been extended by a collection of courses designed to incorporate critical areas.

These newer courses also addressed a problem encountered by some of our incoming graduate and undergraduate students; some technologies have become such an integral part of our programs that students may not have had an adequate background before being expected to hit the ground running in a lab environment. The best example of this is virtualization. It is difficult to separate virtualization from networking or systems administration in most modern communication architectures. Similarly, we had integrated virtualization into many of our courses, assuming that students would understand what they were to do and how to use the tools. We discovered that this was not always the case. To address this challenge we created a virtualization course and adopted a full stack approach.

— Virtualization - This course will take the students through the evolution of virtualization. The course begins with virtual network topologies such as VLANs, trunks and virtual routing and forwarding. From there we examine the various host-based virtualization platforms, bare metal hypervisors, server virtualization, storage and cloud computing.
— Configuration Management - This course teaches students advanced techniques in the Perl language. Techniques include the use and construction of object oriented scripts, user administration and monitoring, file system walking and checking, and computer and network security issues.
— Storage Architectures - This course provides students with a theoretical as well as hands-on exposure to enterprise scale storage technologies such as storage area networks and network attached storage. Students will study SCSI, Fibre Channel, IP Storage, Infiniband, and Fibre Channel over Ethernet.
— Seminar in Software Defined Networking - Students will take traditional network architectures and, utilizing a combination of virtualized platforms, logical network topologies and emerging standards, will complete construction of a network based on controllers, OpenFlow and Open Virtual Switch (OVS).

Additions and modifications to the courses are made based on input from industrial advisory boards, careers fair participants, industry best practices and research projects. Software defined networking is a topic that regularly comes up as it embodies next generation structures along with so many of the needs we see today. These include an increased emphasis on tool creation,

orchestration, visibility and integration or collaboration between elements.

This paper is not about the success or lessons learned from a particular course; these listings indicate our desire to incorporate aspects gleaned from research projects and interaction with industry partners. SDN serves as an excellent example of how this comprehensive architecture can provide students with not only excellent learning opportunities, but a forward looking research agenda. As controller and Openflow based SDN is an emerging technology, there are ample research topics and targets for graduate capstone projects.

2.1. The Sys and Net Admin relationship to SDN

Having discussed the content of the program, we might now investigate the relationship between SDN architectures and systems/network administration coursework. Why might it be a good fit? Many of our systems and network administration topics can be thought of as applied, meaning that they represent very real, job oriented tasks. But included in program goals is the need to provide the background, reasons and higher thinking associated with the practice. Similarly, a complete understanding of SDN requires that the researcher or student examine the traditional methods used in network operation and the problems that are created. Moving to the new paradigm means that we will be using new skills and thinking about the forwarding process in a different way. Several of the courses in this list were either created or updated as a direct result of our experiences with SDN topologies and an over-reliance on assumed skills.

Like so many areas within the industry, there is increase in the amount of scripting and device administration for normal operation. Examples include the call for greater automation and orchestration platforms. Networking tasks are also becoming more systems or server administration like. In addition, the separation of the various planes and the virtual nature of elements requires that we understand more of the reasons as to why something is done and the under-the-hood operation. SDN is a multifaceted, comprehensive architecture that can be multipurpose and complex. Operating an SDN architecture requires that nearly the entire collection of systems administration skills be brought to bear.

As discussed previously, SDN architectures include both hardware and software components. While vendors continue to claim that SDN can still be managed using mechanisms that are similar to the console and command line interfaces so common to routers and switches, it is hard to avoid the realization that working with virtualized elements, many of which are open source, is very like the tasks we see in systems administration. Management of the controller and the virtual environment are very similar to running any other server. In fact, many network devices behave and operate like hypervisors. As will be discussed later in this paper, running an SDN research or test project is very much like a development environment in that the connections and nodes are remote and virtual. In addition, the resources supporting the SDN build are a practical implementation of our systems administration topics. Several examples can be found in configurations and revision history, the use of git repositories, interaction with the open source community, management of the hypervisor software and chassis, patch or updates to the systems, software installs to the virtual machines and desktops, automation or scripting of tasks, processing of data, managing SSH connections and keys, performance testing, user accounts and access.

If we use the examples from Facebook, is an integration between teams and ideas. While we might not choose to convert a routing course to controller based SDN, we might choose to adopt management techniques and dynamic configuration as a part of the class. In systems administration we see tasks such as log file growth and monitoring of servers or intrusion detection systems taking up a greater percentage of our time. So, we choose to adopt automation to address a subset of the task to free up valuable human resources. The FBAR system is full of ideas overlapping other areas that we have also chosen to adopt. These include Devops and orchestration tools such as

Kupernetes [Bernstein 2014] and Chef.

SDN offers several benefits over the traditional network architecture but it does comes with some added complexity and the potential to demand a greater level of programming and management. Someone, local or remote, must build, provide and support SDN projects. Add to this that the elements are running in virtual machines and we see an increased set of administrative tasks. From this point of view, SDN architectures and peripheral components dovetail very nicely with standard systems and network administration curricula because the students and faculty can be directly engaged with an architecture requiring all of their skills and knowledge.

3. IMPLEMENTING SDN PROJECT AND COURSE WORK

In order to experiment with SDN architectures, a project might start with a couple of very basic topologies. Perhaps the easiest would be using Mininet which is a self-contained SDN environment that runs in a virtual machine [Lantz et al. 2010]. Mininet enables the construction of a wide variety of SDN topologies from the command line interface. Mininet can also be scripted facilitating expansive testing and experimentation. However, this self-contained structure limits student and faculty exposure to actual operating network elements, even though Mininet constructs can be accessed via the command line. Thus, all links and traffic are simulated. For our purposes, the decision was made to use actual physical and virtual SDN deployments in projects and courses. The first project described in section 4 was built by a graduate student for her capstone. The second was built by faculty as a test case for coursework and research projects. As a result of these successes, further research was promoted and several components were assigned to students for course laboratory builds.

3.1. Physical and Hybrid Topologies

As one might imagine, the construction of a physical topology requires a greater commitment of resources. In the base topology, this would require four physical boxes; two for network nodes, one for the controller and one for the network switch. These computers need not be powerful and in fact, our initial topologies were built with cast off equipment from the department. We started with physical hardware only and allocated four machines per SDN topology. Other than the need for machines, the central node required 3-4 network interface cards, depending on the management interface configuration. For example, the computer running openvswitch (OVS) was connected to all three of the other nodes but eventually a fourth management connection was added. The management connections allowed remote configuration and patch updates. From an educational point of view, building a physical topology is rewarding because the student or research must touch and configure everything. Problems, configuration or otherwise, must be solved by the project members. In addition, results, network traffic, server needs and the devices themselves were immediately accessible. From a project management perspective, space becomes a real concern.

As with any physical deployment, heat and ventilation can also be a concern. Over the course of this work, there were times when the entire experiment had to be moved to an office space that under normal conditions has sufficient environmental controls. However, the equipment shown in Figures 4 and 7 can give off a surprising amount of heat and noise. Temperature in the space went above eighty-five degrees and the noise made working in the space untenable.

As we will see later in this paper, some of these can be virtualized resulting in a hybrid topology. Even with the virtual infrastructure, the hypervisor chassis, any external nodes and the cabling are all tangible items that must be allocated. In this second variation, only a single computer has to be allocated, alleviating many of the space and resource concerns. However, this box should have a minimum of 8GB of RAM and a hard drive sufficient for stored install media and virtual machines. In this configuration the network nodes, controller and OVS are running in virtual machines. The computer will be the hypervisor chassis which adds an additional systems administration task. Another externally connected management node is also required although this

can be the researcher or student laptop. In our build we used VMWare ESXi 5.5 and vSphere for the hypervisor and management components.

3.2. Virtual Topology

It is possible to run the entire project at a remote site which removes the need for physical equipment. The only local requirement would be an access or management node. GENI is an engineering and educational environment that allows students to create any number of virtual machines and their associated connections. Each virtual machine has an external management connection. GENI is a fully scriptable environment that also allows the installation of additional software. Those familiar with Planetlab, Emulab or Amazon Web Services based virtual machines will have a good mental image of the system. We have used GENI virtual machines and connections in a wide variety of activities including capstones, small experiments and even running entire courses. An example of the GENI interface used by researchers and students can be seen in Figure 3.

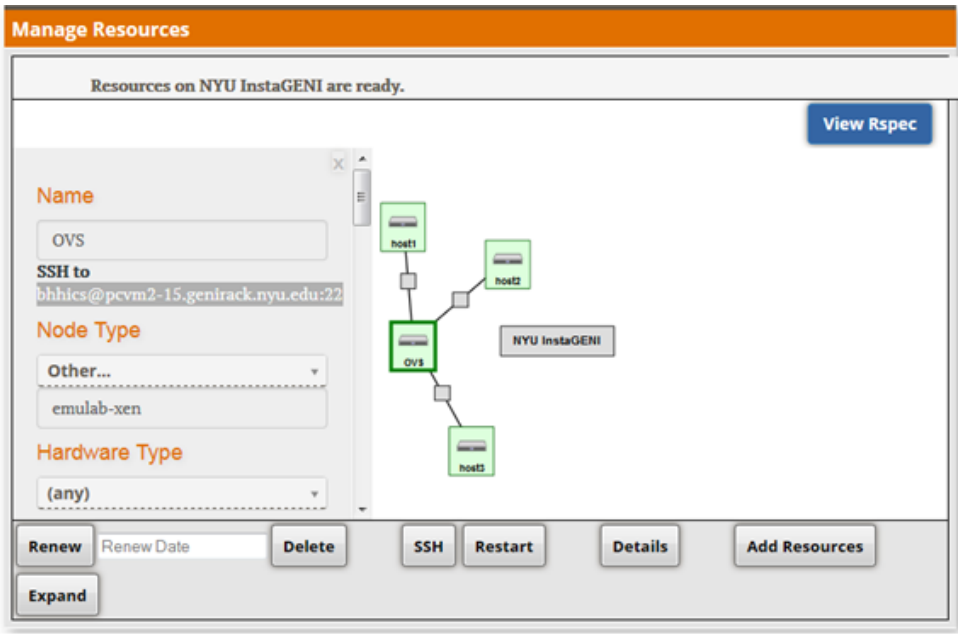

Fig. 3: GENI jacks interface

The advantage of using GENI was that experiments were not limited by the number of VMs or their connections. In addition, there was little to manage locally. On occasion, the system does experience slower performance but on the whole our experiences with the system were very good. There is a certain intangible satisfaction when the researcher is able to walk into the lab and directly interface with the testbed but this may be outweighed by the benefits.

Undergraduate students that are already familiar with our lab infrastructure have been given the SDN physical topology to build as part of the seminar. About half of the class reached all of the topology goals but these results are inconclusive because of the variation in student background. What was clear from the seminar was that the skills required came from both systems administration and networking. Graduate students are assigned the virtual build on GENI. This gives them the opportunity to work from several locations (many of our students are distance) and they do not need to maintain a lab presence for the entire semester. This is a regular part of our Enterprise

Networking graduate course. Each of the projects described in the next section will provide details on the equipment requirements and the challenges encountered. In the second project, a comparison between experiments run on GENI and the physical environments is also included.

4. COMPLETED RESEARCH PROJECTS

The following projects were run on campus during the last two years. They form the basis of the work going forward and the modules integrated into the systems and network administration coursework.

4.1. Traffic flow comparison between SDN controllers: Pox, Floodlight and OpenDaylight

A vast variety of SDN controllers have been deployed to date, mainly distinguished by the programming language employed, the protocols used to interact with the forwarding layer and the support of distributed cluster designs. This particular graduate capstone project examines the traffic behavior and network performance of multiple SDN controllers, using specific network setups while distinctive simulated TCP and UDP data streams traverse the network. This assessment intended to provide some insights about the operational features of different SDN controllers, but mainly a comparison analysis to distinguish traffic flow patterns using similar network environments among the controllers. Similarly to previous research, this project included the following:

— Deployment of the virtualized infrastructure, using an in house bare metal server and GENI
— SDN deployment: Controllers, virtual switches and end devices
— Definition of traffic generation mechanisms and tools for data gathering
— Definition of benchmarks for analysis

In overall, six independent experiments were performed to evaluate the performance statistics and traffic flow behavior between the controllers. These experiments were based, for the most part, on throughput (Mbps), delay (ms) and jitter (ms) measurements, taking into consideration the infrastructure employed.

These experiments were mainly instantiated in a hybrid infrastructure comprised by a physical box running VMware 5.5. As shown in Table I below, the system was not robust, but provided a stable and reliable environment while the research was underway.

Table I: SDN Virtualization Equipment

Description	Value
Operating System	VMware ESXi 5.5.0 build-2068190
System Model	Power Edge R520
Installed memory	16GB
Disk	Raid10 - 4 x 931GB = 1.81TB
Processor	4 x Intel(R) Xeon (R) CPU E5-2403 0 @ 1.80GHz

The network architecture consisted, mostly, in six virtual machines from where we used three as controllers, two as clients and one as openvswitch. Other nodes were used for routing and testing purposes, as described in the Figure 4.

The secondary system, in which we also performed several tests, was built in GENI. This topology mirrored the design elaborated in the lab, except for the routing and management devices. The virtual nodes were instantiated in multiple independent slices, which are isolated network instances in GENI cloud, as observed in Figure 5

Both the local and remote environments used a base installation of Ubuntu LTS 14.X, a Linux flavor with well known repositories and stable responses for OVS, SDN controllers, and OpenFlow in general. A clean installation of the virtual machine, prior SDN configuration,

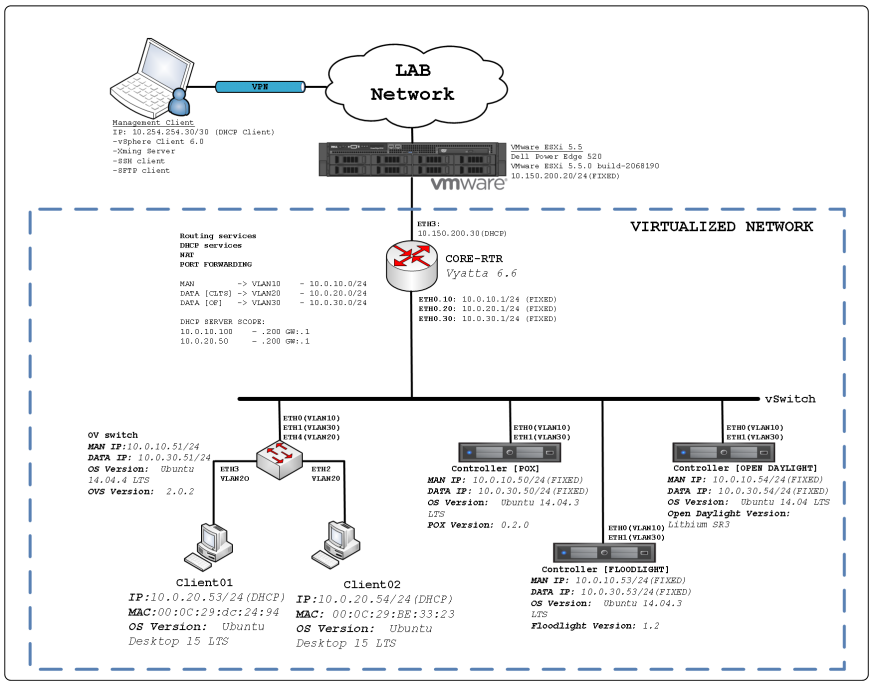

Fig. 4: Local network architecture

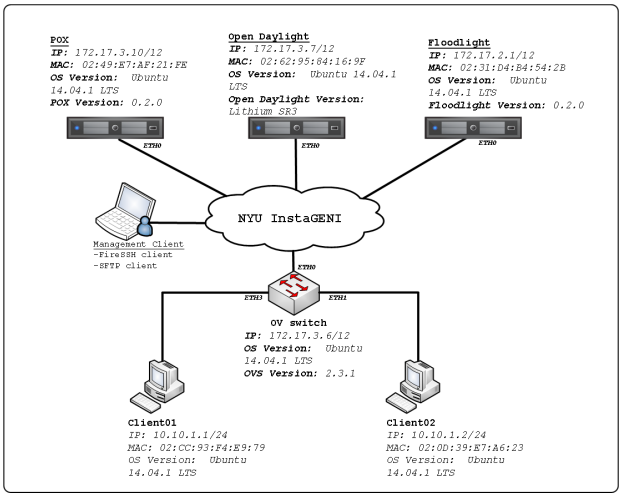

Fig. 5: Remote network architecture

contained the VWware tools, OpenSSH and updated repositories.

In summary, more than 300,000 samples were gathered among all tests to be able to provide an accurate comparison analysis between the controllers. Some traffic generation tools that made this research possible were IPERF and Ping, for TCP/UDP flows and ICMP packets.

After observing and analyzing the results, we were able to identify some interesting patterns in all controllers when dealing with queuing and processing flow times. We concluded that the injected delays by controllers are not relevant once the flow entries are installed in the forwarding device. Even though the flow construction set up times varies drastically from controller to controller, as well as their expiry times, the performance of the network measured in throughput and latency with respect to the clients, is consistently similar among all controllers.

Building this type of project allow students understand the concepts of network virtualization and SDN, as they are not given with a pre-fixed deployment, such as Mininet, as all the components are built from scratch. In our particular case, we ran into few difficulties even before dealing with our intended scope (SDN) -network isolation via virtual switches (in VMware hypervisor) for ensuring communication between the nodes and openvswitch was not possible in the first attempt, network segmentation for management traffic was not part of the initial design and providing a clean and functional installation of nodes and controllers was not an easy task either. Using a virtual environment resulted favorable, as the work could be backed up before initiating a new phase.

4.2. SDN testbed construction

The testbed described here was build to investigate the efficacy of both physical and virtual architectures for use in other projects and student work. Performance metrics were also examined. In order to experiment with SDN or SDN enabled technologies and ideas, the researcher must have either SDN enabled devices (such as those manufactured by Brocade, HP, Nicera or BigSwitch) or have access to SDN virtual machines or VMs. For example we might use Linux VMs and install controller/switch packages such as POX and OVS. When working with VMs, compute resources can be obtained from either a local or cloud facility. Cloud resources include Amazon Web Services or even private clouds. Locally we might use clusters or build a smaller setting such as one based on VMWare ESX.

The problem with general compute resources is that they may not be appropriate for the task at hand. For example, clusters may be well suited for providing significant processing capability but they are not very efficient when the need is for a large number of VMs. GENI (Global Environment for Network Innovation) is an architecture built for the purpose of experimentation with SDN type topologies. However, the resources are time restricted and storage is extremely limited. This project sought to explore the issues associated with construction of an SDN infrastructure. An examination of remote facilities will also be completed and the two approaches will be compared. The original project goals included:

— Develop a list of necessary hardware and software components.
— Deploy both ESX and KVM hypervisors and interconnect them with the wired network.
— Deploy SDN controllers and the associated openvswitch.
— Deploy network hosts
— Provide an explanation of system operation.
— Develop a testing methodology and collect a suite of tools for researchers and educators.

Time permitting, one of the supplemental tasks will be to emulate the Internet of Things or IoT. The IoT can have several meanings including machine to machine communication, very often with a collection of sensors. This emulation will be attempted using Mininet. Mininet is a virtualized network environment that can be extended to a physical network.

4.2.1. Testbed Construction and Testing - Phase 1. Since we did not possess switches or routers capable of communicating via OpenFlow, the devices had to be virtualized or installed as a virtual switch. Thus phase 1 was the construction of the virtualized infrastructure. The amounted to installation of VMWare ESX, the base virtual machines, software and some portion of the physical

network in order to allow management nodes to communicate with the virtual machines. An early goal was to build a similar box based on Kernel Virtual Machine (KVM). However, there was not enough hardware and so this task was dropped.

ESX is a bare metal hypervisor that is managed remotely via VMWare vSphere. Ubuntu Linux was chosen as the base distribution for ease of use and the apt package manager. Four virtual machines were created; POX controller, OVS (Open vSwitch) and two network nodes. POX and OVS are packages that had to be installed and configured on top of Ubuntu. The virtual machines had to be connected from the virtual switch inside of ESX to the hardware switch of the testbed. The internal virtualized topology can be seen in Figure 6. This image shows not only the integration with virtualization but that network elements themselves are virtualized. Thus, there is the need to build VMs, configure them for operation and integrate them with a physical infrastructure.

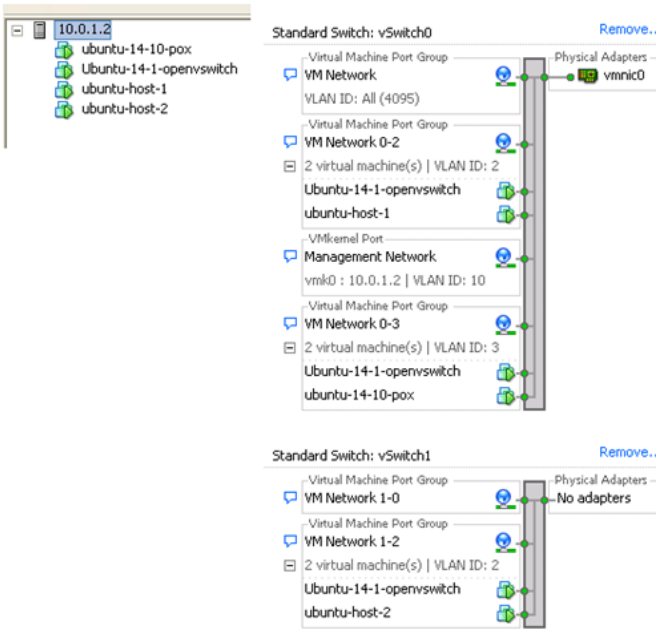

Fig. 6: ESXi Topology

Lastly, the VMs must be allowed to connect through the SDN topology within ESX to the outside world. This network configuration is non-trivial as it requires understanding of several advanced networking topics including VLANs, trunks, port forwarding and network address translation. At the conclusion of this phase, the first series of tasks for the project were completed. Once the topology was constructed, a collection of SDN experiments was selected for comparison against running them on GENI. These include layer 2 and layer 3 connectivity experiments.

4.2.2. Testbed Construction and Testing - Phase 2. The physical topology that was deployed for the project is shown in Figure 7. While the topology and tests were completed, there were still a couple of secondary activities that would increase the value of the project. One such task was to develop a methodology that might make the overall architecture less onerous to work with and therefore more practical for experimentation.

To this end, the router and switch configurations were modified for external access and the security of the system was increased. Security improvements include installation and configuration of

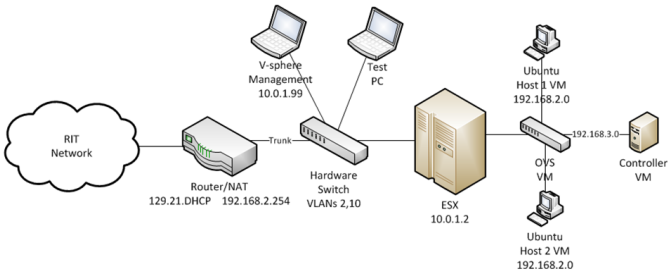

Fig. 7: Additional Physical Topology

Secure Shell servers (sshd) on virtual machines and hardening of the router.
Another modification was to the flow table tools. Flow tables contain all of the information necessary for the processing of packets but they can be difficult to read and interpret. Accessing them is straight-forward (ovs-ofctl dump-flows br0) but the information is not tailored for what the user requires. To alleviate some of this a small script was written that serves as a front end for the switch. The idea is that the user can either obtain the entire table or query the switch for specific information.

4.2.3. Testbed Construction and Testing - Phase 3. The last component of the project was to examine the testing methodologies that might be used on an SDN topology and add an aspect that might emulate the IoT. As mentioned previously, Mininet would be deployed for this purpose. Mininet is running in a separate VM with a switch and series of hosts. The challenge is in integrating Mininet into the topology because two switches will seek to connect to the same controller. Mininet commands can be run interactively but it is more efficient to use the Python API and so a script was written to handle the tasks of the Mininet topology. The entire integrated topology is shown in Figure 8.

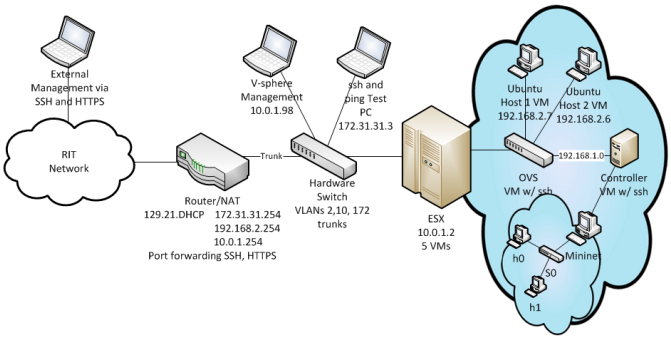

Fig. 8: Complete Topology

In testing and acquiring data for this project, there are a couple of aspects of the architecture to be examined, most notably SDN on GENI vs. SDN on ESX and SDN vs. traditional networking. In the first case the results were inconclusive but this is simply because the architectures are so similar. Even though one is local and the other remote, the systems are still running in virtual machines and the virtual machines are resident on the same hardware.

However, there can be minor performance differences because of the resources allocated for each VM and the hardware configuration of the hypervisor chassis. The term minor is used because

there are only 5 VMs running on a chassis with 8GB of RAM. While this does have an impact, a larger number of VMs would have been more telling. Attempting to manage and maintain remote resources can be a pain and there are resource limitations, but in this case no clear winner emerged.

When compared to traditional networking, SDN has the potential for greater management traffic because of the controller and the need for OpenFlow messages to be passed between network elements. Unless statically configured (which defeats the purpose), OpenFlow must be involved in every instantiation or modification of flows. This adds overhead, albeit only on the commencement of the flow. The impact becomes less as the flow life or packet train grows. One of the tools used to test the topology was iperf [**?**] which can be configured to send TCP or UDP packet flows. The average results from a base 30 second test are shown in the following tables.

The first set of values shown in Table II is from the virtualized SDN topology.

Table II: iperf SDN test Result Example

Window Size	Transfer	Bandwidth	Window Size	Transfer	Bandwidth
2.19KB	101.67MB	28.4Mbps	2.14KB	79.83MB	22.67Mbps
7.81KB	462.5MB	129Mbps	7.63KB	239.67MB	66.93Mbps
15.6KB	771.67MB	215.67Mbps	15.2KB	313MB	87.4Mbps

This second set of values shown in Figure III is from a comparable physical network.

Table III: Baseline Transfer Values (MB) vs. Windows Size

Window	Cisco	Juniper	D Link
4K	772.6 (16.85)	788.5 (3.41)	814.9 (33.64)
8K	1798 (6)	1820 (30.66)	1824 (23.75)
16K	1835(5)	1820 (10)	1831 (3)
64K	2793(9)	2725 (33.54)	2800 (16.73)

Unfortunately these results only lead to more questions. Due to limited resources, tests had to be run on available hardware and software. These results have far more variables than desired as they include Windows and Linux operating systems. The same is true of the iperf client software. Scaling the window sizes (a problem with different clients) does solve the apparent disparity in output. However, this may serve to underscore the changing impact of OpenFlow over flows of longer duration.

5. DISCUSSION

As systems and network administration programs seeking to provide the understanding and tools for success in either industry or graduate programs, there is the need to match suitable lab experiences with program goals. Early in the education process, these experiences can be found in software installation and management of small systems. As the student grows or a research agenda is established, finding a suitable match can be more challenging. Integrating the latest industry trends, keeping current and developing a comprehensive environment within which to work can add further challenges. In our own program we have wrestled with these issues and have discovered that the combination of software defined networks and the related trends in industry have provided a wide variety of opportunities for students and faculty. Graduate students have also discovered a new and rich vein of capstone topics to mine as SDN has grown to include areas such as SD-WAN, SD-storage and SD-data centers. We believe that having access to an SDN architecture for testing and research is critical for those working in the communication and networking fields. We have argued that SDN is an architecture that, along with related ideas of orchestration, dynamic programming

and automation can add to any program. The creation of courses like Configuration Management and the adoption of tools such as Puppet, Chef and Kupernetes support this point of view. The virtualization course adopts the ”full-stack” approach which requires a comprehensive understanding of the systems rather than just the hypervisor and thus takes a more ”Devops-ish” direction.

But obtaining access to resources is another question. For many, it is not easy to obtain the physical resources required. While this might mean actual computers, it can also mean space or administrative personnel. Specifically, the construction of even a small SDN topology requires four nodes. While the controller and the network element (OVS) can reside in the same box, they still require resources. As this base topology, seen in Figure 2, is the one we used with our undergraduate students. Virtualizing helps because given enough memory, two teams might share the same chassis by running VMs and not interfere with each other. This topology also requires the students to perform a myriad of sys admin activities including accounts, software control, installation and authentication. But these projects have also shown that a local testbed requires significant resources and some level of expertise to configure and manage. So, one of the things that makes SDN and the related activities so attractive as a basis for many systems and network administration students may be lacking in either the students or the faculty member. But, if th steep learning curve is embraced with a local testbed, the students and faculty members gain considerable knowledge and experience as they are exposed to a broader domain of technologies and design requirements. As a starting point, GENI might be considered and the GENI conferences offer travel grants in order to bring educators and researchers up to speed. Again, the base topology is emulated on GENI in Figure 3.

As mentioned previously, many of the experiments resulted in either additional capstone projects for students or more in-depth research projects. The fist project outlined in section 4 is an excellent example of a successful graduate capstone. For example, the creation of the testbed led directly to intensive testing and a comparison between metrics on virtual and physical topologies [Hartpence and Kwasinski 2015]. Some of these tests indicate that for many experiments, a cloud environment like GENI is an effective and appropriate solution. For this reason, and because graduate students are less familiar with our lab infrastructure, we adopted on line or remote resources for graduate coursework. But in cases where increased storage or local control over resources are desired, a local chassis and testbed is still a superior solution.

6. CONCLUSION

Emerging technologies, such as SDN and network virtualization, are gaining increased attention in the industrial sector. The same is true for strategies such as orchestration and automation. Academic programs should be aligned to cope up with these new developing areas, to ensure the student body is well prepared for future endeavors. In this paper we have discussed what might be called traditional network and systems administration curricula and how these courses can be furthered or augmented by SDN topics and experiences. SDN provides an integrated environment which required many if not all of the background and skills obtained in a systems or network administration program, particularly if the goal is to work with a comprehensive SDN infrastructure. Our projects, the successful modification to coursework and industry examples such as the Facebook Auto Remediation System all lead us to believe that SDN and the related ideas a an important part of networking and systems administration programs.

Building, maintaining and operating an SDN environment for testing can be a considerable undertaking depending on the scope of the projects, the desired experimentation and the number of users. Some of the challenges associated with bringing an SDN architecture into fruition were outlined here including space, physical machines and heat. Since these challenges are not insignificant, this paper also describes the facilities used to mitigate the biggest issues. GENI is an on-line, well-supported infrastructure used primarily for the support of SDN projects and research.

Courses, advanced projects or simple tests can all make use of virtual machines and connections built within GENI.

Software Defined Networking provides an ample domain for research and investigation, as it is still a fairly new and diversified technology. Projects as the ones discussed in this paper, showcase various ways universities can introduce these new concepts to the scholars. By transitioning from the theory to practice, students gain important knowledge which could be then translated to improved ideas and novel designs for existing technologies.

REFERENCES

David Bernstein. 2014. Containers and cloud: From lxc to docker to kubernetes. *IEEE Cloud Computing* 1, 3 (2014), 81–84.

Chip Elliott. 2008. GENI-global environment for network innovations.. In *LCN*. 8.

Natasha Gude, Teemu Koponen, Justin Pettit, Ben Pfaff, Martín Casado, Nick McKeown, and Scott Shenker. 2008. NOX: towards an operating system for networks. *ACM SIGCOMM Computer Communication Review* 38, 3 (2008), 105–110.

Bruce Hartpence and Andres Kwasinski. 2015. Performance evaluation of networks with physical and virtual links. In *Global Information Infrastructure and Networking Symposium GIIS, 2015*. IEEE, 1–6.

Peter Hoose. 2011. Facebook Auto Remediation. (2011).

Diego Kreutz, Fernando MV Ramos, Paulo Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig. 2015. Software-defined networking: A comprehensive survey. *Proc. IEEE* 103, 1 (2015), 14–76.

Bob Lantz, Brandon Heller, and Nick McKeown. 2010. A network in a laptop: rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*. ACM, 19.

Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. 2008. OpenFlow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review* 38, 2 (2008), 69–74.

Ben Pfaff, Justin Pettit, Teemu Koponen, Ethan Jackson, Andy Zhou, Jarno Rajahalme, Jesse Gross, Alex Wang, Joe Stringer, Pravin Shelar, and others. 2015. The design and implementation of open vswitch. In *12th USENIX symposium on networked systems design and implementation (NSDI 15)*. 117–130.

Rob Sherwood, Michael Chan, Adam Covington, Glen Gibb, Mario Flajslik, Nikhil Handigol, Te-Yuan Huang, Peyman Kazemian, Masayoshi Kobayashi, Jad Naous, and others. 2010. Carving research slices out of your production networks with OpenFlow. *ACM SIGCOMM Computer Communication Review* 40, 1 (2010), 129–130.

OpenFlow Switch Specification. 2011. Version 1.2 Wire Protocol 0x03. *Open Network Foundation* (2011).

The USENIX Journal of Education in System Administration
Volume 2, Number 1 • November 2016
www.usenix.org/jesa/0201

An Innovative Approach to Bridge a Skill Gap and Grow a Workforce Pipeline: The Computer System, Cluster, and Networking Summer Institute

CAROLYN CONNOR, Los Alamos National Laboratory
ANDREE JACOBSON, New Mexico Consortium
AMANDA BONNIE, Los Alamos National Laboratory
GARY GRIDER, Los Alamos National Laboratory

Sustainable and effective computing infrastructure depends critically on the skills and expertise of domain scientists and of committed and well-trained advanced computing professionals. However, in its ongoing High Performance Computing (HPC) work, Los Alamos National Laboratory noted a persistent shortage of well-prepared applicants, particularly for entry-level cluster administration, file systems administration, and high speed networking positions. Further, based upon recruiting efforts and interactions with universities graduating students in related majors of interest (e.g., computer science (CS)), there has been a long standing skillset gap, as focused training in HPC topics is typically lacking or absent in undergraduate and in even many graduate programs. Given that the effective operation and use of HPC systems requires specialized and often advanced training, that there is a recognized HPC skillset gap, and that there is intense global competition for computing and computational science talent, there is a long-standing and critical need for innovative approaches to help bridge the gap and create a well-prepared, next generation HPC workforce. This paper places this need in the context of the HPC work and workforce requirements at Los Alamos National Laboratory (LANL) and presents one such innovative program conceived to address the need, bridge the gap, and grow an HPC workforce pipeline at LANL. The Computer System, Cluster, and Networking Summer Institute (CSCNSI) completed its 10th year in 2016. Herein is the story of the CSCNSI, its evolution, a detailed description of the design of its Boot Camp, and a summary of its success and some key factors that have enabled that success.

1. INTRODUCTION

The High Performance Computing Division at Los Alamos National Laboratory (LANL) manages world-class supercomputing centers in support of the Laboratory's national security science mission. This includes specifying, operating, and assisting in the use of open and secure high performance computing, storage, and emerging data-intensive information science production systems for multiple programs. More broadly, LANL, and national laboratories in general, have a strong need for scientists and staff who understand and have the ability to operate, use, and help advance high performance computing systems and technology.

As an example, the Trinity supercomputer, the most recent supercomputer to arrive at LANL, was designed to exceed computational performance of 40 Petaflops/s and is a next step toward the goal of achieving exascale computing (a million, trillion operations per second) [NAS Report 2014; ASCAC Report 2014]. Trinity is the first of the National Nuclear Security Administration's (NNSA's) Advanced Simulation and Computing (ASC) Program's advanced technology systems. Once fully installed, Trinity will be the first platform large and fast enough to begin to accommodate finely resolved 3D calculations for full-scale, end-to-end weapons calculations. The Trinity system will reside in LANL's Strategic Computing Center in the Nicholas C. Metropolis Center for Modeling and Simulation. The Strategic Computing Center is a 300,000-square-foot building. The vast floor of the supercomputing room is 43,500 square feet, almost an acre in size [National Security Science 2013].

Trinity will enable at least eight times greater application performance than Cielo, the current NNSA supercomputer sited at LANL. Trinity will introduce the "Burst Buffer" (a storage layer between the compute and storage systems to relieve the bandwidth ingest burden on the parallel

This work was performed at Los Alamos National Laboratory, supported by the U.S. Department of Energy contract DEAC52-06NA25396, and at the New Mexico Consortium, with additional support from the National Science Foundation under awards CNS-1042537 and CNS-1042543 (PRObE). http://www.nmc-probe.org/. This release is under LA-UR-16-26535

file system) [Liu et al. 2012] and "Advanced Power Management" (adding power adjustment capability to the BIOS) as part of the platform [Laros et al. 2016]. These technologies will be provided as part of a fully integrated system consisting of compute nodes (>19,000), memory (>2PB), high speed interconnect, and parallel file system (>80 PB of usable capacity) as part of the planning for exascale systems [Trinity Press Release 2015].

In order to prepare for, operate, and fully utilize Trinity and its successors, the national laboratories require a workforce with the skills to

- Manage and operate a highly complex computing infrastructure that includes miles of cable, hundreds of thousands of connections, and tens to hundreds of thousands of storage devices;
- Expertly translate computing industry trends into a robust high performance computing environment;
- Design and operate the physical infrastructure supporting LANL's computing needs;
- Operate and enable effective use of the full high performance computing environment;
- Leverage high performance computing systems, processes, and expertise, beyond traditional HPC, to advance innovative LANL scientific efforts;
- Engage in research, development, and state-of-the-art software engineering, supporting development, design, and effective use of increasingly complex large-scale data and computational environments.

While HPC has become a long-standing and essential tool to advance scientific discovery and achieve national goals [Graham et al. 2005; Congressional Hearing 2013; NAS Report 2016] and is itself an area of active and advancing research [The Next Platform 2016; ASCAC Report 2014], there exists a gap between the technical skillsets needed, particularly at the entry level, and the exposure and skillsets provided in current undergraduate (and even graduate level) academic programs. Recognizing this gap and the resulting unrelenting organizational strain due to an inadequately prepared entry-level pipeline, LANL set out just over a decade ago to actively engage to bridge this gap for a specific pipeline need, establishing the Computer System, Cluster, and Networking Summer Institute (CSCNSI) in 2007.

With nearly five highly successful years under its belt and a proven program, LANL collaborated with the newly established New Mexico Consortium [NMC 2016] on a joint proposal to the National Science Foundation (NSF) with the result that the CSCNSI received five years of partial NSF support (2011-2015) as an educational component of PRObE, the Parallel Reconfigurable Observational Environment. [Gibson et al. 2013] PRObE is an NSF-funded low-level, large-scale computer systems research facility located in Los Alamos, NM and housed by the New Mexico Consortium (NMC) at the Los Alamos Research Park. The primary mission of PRObE is to provide the national systems research community resources in the form of several large-scale testbeds (of 1000+ computing nodes) built from repurposed supercomputers from LANL and dedicated to computer systems research. As such, PRObE is the only facility of its kind in the world.

This paper describes the CSCNSI, a unique cluster computing technical enrichment program. In Section 2, we describe CSCNSI program details, including motivation, program overview, goals, cluster Boot Camp execution and extension, HPC Seminar Series, team research project concept, and the student research Mini-Showcase. Section 3 discusses program results and impact. Section 4 offers a discussion of notable keys to success. Section 5 discusses reproducibility experiments and potential. Finally, Section 6 provides concluding remarks.

2. COMPUTER SYSTEM, CLUSTER, AND NETWORKING SUMMER INSTITUTE

2.1 Motivation

The growing importance and pervasiveness of high performance computing for solving increasingly complex problems coupled with the fast-paced evolution of the HPC environment and the technical complexity of the challenges in the on-going march to the next generation of supercomputers has widened the entry-level skill gap for HPC organizations. LANL found itself largely unable to find student interns and early-career applicants with sufficient background and

practical skills to fill its workforce needs within its HPC environment [Connor et. al. 2016]. In response to this gap, LANL created the CSCNSI to seed a strong entry-level applicant pool from which it would be able to staff and then further develop its HPC workforce as those staff progress through their individual career paths.

2.2 Institute Goals

To accomplish the overarching pipeline and workforce development objectives, three developmental focus areas were identified and a methodological approach developed in which students work in small project teams to execute real-world projects on computer clusters that they assemble and configure to achieve the goal triad listed below.

— Practical skill development. The CSCNSI is designed to provide students with hands-on experience in setting up, configuring, administering, testing, monitoring, and scheduling of computer systems, supercomputer clusters, and computer networks.

— Technical broadening. The CSCNSI exposes students to a variety of key HPC-related topics, technologies, and problems of interest through execution of a technical research project, as well as participation in tutorials, seminars, and facility tours, including a full facilities and machine room tour of the Metropolis Supercomputing Center at LANL.

— Professional development. The CSCNSI provides students with a targeted set of professional development opportunities, including teaming and written and oral communication skill development through a series of workshops and training sessions and assignments, including resume writing, technical project execution, technical poster development, and technical presentation development and coaching.

A second set of workforce development goals is achieved in the form of staff revitalization through mentoring, lecturing, and facilitating research collaboration opportunities.

2.3 CSCNSI Overview

The primary objective of the CSCNSI is to provide a thorough introduction to the techniques and practices of cluster computing with a specific focus on system administration for large computing clusters. This objective is accomplished by leveraging and building upon the background and curriculum currently provided in a typical undergraduate computer science or computer engineering program, which has historically been more theoretically than practically based. Application to the CSCNSI is a competitive process in which student applications are solicited nationally. Although there is no hard restriction, the program targets students at a typical Junior-level and above to provide a pool of applicants with sufficient technical background to maximize their ability to absorb the material and grow from the experience. Between nine and twelve students are selected each year. Students typically have had no cluster computing exposure prior to beginning the program. Many express that HPC education is either not commonplace or not available at their university. The vast majority has never seen, much less been on the floor of a computer machine room.

At the end of every summer, staff updates the CSCNSI trifold brochure for the next session. The brochure describes the program, lists the application requirements, and gives application instructions. It is posted on LANL and NMC websites and is included in the informational materials that are sent out to Fall on-campus university career fairs, as well as career tables at various Fall technical conferences (e.g., the Tapia Diversity in Computing Conference (held in September), the Grace Hopper Celebration of Women in Computing (held in October), and the Supercomputing Conference (held in November). In addition, HPC staff share the brochure with their professional networks. Over the years, the program has benefitted from graduates recommending and publicizing the program to other students when they return to their campuses and through their social networks. In a departure from previous years, the application deadline for CSCNSI 2017 has been moved up to December 1, 2016. Previously, the deadline had been in mid-February with notification in early March; however, it has become evident that many students

are receiving and accepting internship offers by the end of the Fall semester. This change brings the CSCNSI in line with that trend.

The application package consists of resume, transcripts (initially unofficial, but updated to official upon selection), one letter of recommendation from a faculty member, and a letter of intent confirming U.S. Citizenship and categorizing the applicant's familiarity with command-line Linux (e.g., no Linux, novice, intermediate, advanced). Further, the letter of intent should discuss the applicant's technical and personal strengths, goals, and interests and relate them to why the student believes he or she should be selected, as well as to how a CSCNSI appointment would help the student achieve said goals. Prior experience with and working knowledge of command-line Linux is a program pre-requisite. Eligibility includes meeting minimum grade point average requirements of 3.0 for undergraduate students and 3.2 for graduate students.

The program includes lecture, laboratory, and professional development components. Students explore current challenges in high performance computing (HPC) through the inclusion of an extensive seminar series consisting of seminars given by practitioners and researchers actively engaged in HPC-related topical areas. At its core, the CSCNSI is an intensive, project-driven, technical summer program during which each small student team actively builds a computer cluster and then carries out an HPC-related research project under the advisement of an assigned mentor or mentor team. The CSCNSI is executed as a paid 10-week undergraduate research internship. The NSF award allows the CSCNSI to be co-hosted by LANL and NMC, with program costs, including funding of student internships, split appropriately between the two institutions and funding sources to support the various aspects of the program. Student salaries are set in accordance with LANL's standard student salary tables, which are based upon where a student is in their educational program (i.e., number of credit hours earned to date). A small amount of support for travel expenses is included.

2.4 Execution

The CSCNSI is structured as a set of complementary components that together realize the goal triad described in Section 2.2. After on-boarding and orientation activities, students are assigned to small teams that spend approximately two focused weeks building a small, but fully functional supercomputer (one cluster per team), as described in Section 2.5. The second phase of the program consists of attending a set of HPC Seminars (described briefly in Section 2.6), executing research projects on the clusters that the students have built (Section 2.7), and documenting and reporting research project results in the form of a technical poster and team presentation (Section 2.8).

2.5 Cluster Boot Camp

The first phase of the program is CSCNSI Cluster Boot Camp (a.k.a. Boot Camp) in which students work in teams of three under the tutelage of an instructor and teaching assistant (both typically CSCNSI graduates) to build small supercomputers (one cluster per team), including everything from installing the hardware into the computer rack and labeling and connecting the cables and computer nodes to installing and running an operating system and software stack similar to what national supercomputing sites use to operate their very large computing clusters. Each team is provided ten to twelve computer nodes, a computer rack, and all required cabling, equipment, and machine room space to build a fully operational computer cluster over approximately a two-week period. Boot Camp lecture/laboratory topics include the following.

- Setting up and wiring a cluster,
- Linux review/Operating system configuration,
- Linux services/ Head node configuration,
- Network booting,
- General networking,
- Warewulf Cluster Management/ Diskless booting,
- Scripting,
- Monitoring and benchmarking/ Ganglia setup,

- Parallel programming/MPI setup,
- Lightweight Directory Access Protocol (LDAP), Kerberos, and
- Configuration management (e.g., Puppet)

However, it is important to note that Boot Camp as described above and currently implemented was not the original design, perhaps because it is not an immediately obvious starting point. The process that brought us to the boot camp model is described in the next section.

2.5.1 From Tutorial to Boot Camp – the Evolution of an Educational Program

Everything has a beginning. When the original 9-week CSCNSI was launched (summer of 2007), the brand new, four-week long systems administration curriculum with attached project was an organized mass of extracts from various talks on subjects in the field of systems administration and systems software. As with any other program, it had to start somewhere. The first group of students consisted of nine mostly locally acquired recruits who dove into a grand experiment in which no one (including organizers and instructors) fully knew what to expect. A carefully crafted schedule of 90-minute lecture on a selected topic followed by a 2.5hr laboratory (i.e., practical implementation session for the three teams of three students each) with essentially step-by-step instructions on how to perform the specific task on the experimental equipment was imposed on students with very little background in either Linux nor systems administration at scale. The cluster hardware – at the time – was dual socket, dual core, computer servers with 8 GB of memory, and a high-speed interconnect – parts of a decommissioned LANL computer cluster. Each team received 10 servers and detailed instructions on how to wire them together after a head node had been installed. The cluster was built, nodes PXE booted (Preboot Execution Environment, i.e., booting computers via a network), and statefully installed on a local hard drive on each node, using Anaconda on CentOS 5.

Boot Camp in its current form retains the topically advancing lecture/laboratory format toward the goal of building a fully functional and stable computer cluster; however, within a few years, Boot Camp emerged as a distinct and highly focused phase, condensed to the first two weeks. Boot Camp has evolved from program core to an essential rite of passage that launches the students into the project phase, which in turn, has become the new program core.

The larger goal remains - to grow in each student a system administration skillset suited to an HPC environment supporting computing at scale, such as would be found at a national laboratory generally, and LANL itself more specifically. While it might seem like the obvious choice to mimic the software and hardware configurations in LANL's production computing environment, there were compelling reasons that led us to stray from this path. First, it was recognized early on that there is a set of foundational concepts that are central to building a computer cluster (e.g., the concepts of operating system (OS), provisioning, and configuration) that is independent of software or tool choice, much as there is a set of foundational programming concepts that transcend choice of programming language. Further, there were practical constraints to be considered which led to some practical implementation choices.

2.5.2 The Choice to Rely on Open Source Software

LANL has a long history of building large computer clusters, and increasingly (like most other supercomputing facilities) has moved from large custom-built compute systems to more general purpose, off the shelf Linux-based clusters because of the clear benefits of using common rather than specialized components both for availability and economic reasons. While there are commercially supported Linux distributions, it was a purposeful decision to use open source software for the CSCNSI. One overriding reason is a funding level that requires minimization of the purchase of often-expensive licenses for systems that are only used for teaching purposes and for only a few weeks each year.

Additionally, there is a built-in expectation and desire for the students to be challenged, and sometimes in unexpected ways, whether it be with incompatible kernel and driver versions, undiscovered bugs, or undocumented features. A primary program goal is to grow practical problem solving and troubleshooting skills; therefore, part of the evolved teaching methodology is

a communicated expectation that students will encounter roadblocks and will experience micro failures because, as experienced systems administrators have learned, there are things that one does not know or cannot figure out the first time around. It is our expectation that students will run into these types of realistic problems and setbacks and will struggle, but ultimately find solutions.

Open source software often creates opportunity in this regard, as it may create situations where you are at the mercy of forces outside of your control. An example is the delayed release of Community Enterprise Operating System version 6 (CentOS 6) as a direct result of the lead of the development team going incognito for several months [CentOS 2016] [Redhat 2016] [Slashdot 2009]. In the case of the CSCNSI, the result was use of CentOS 5 two years past the release of Red Hat Enterprise Linux version 6 (RHEL 6). Another example of forces beyond our control relates to the CSCNSI's chosen systems management suite, Warewulf (from Berkeley National Laboratory (LBNL)) [Warewulf 2016], which as of this writing has still not yet announced an official release for CentOS 7, the operating system to which we hope to migrate the CSCNSI soon, as discussed below. The most recent update was 2014.05.28 [Warewulf 2016]. As a result, instead of using a currently available beta version for CentOS 7, the CSCNSI 2016 students remained with CentOS 6 to minimize unforeseen complications beyond the scope of the institute to solve. As a point of instruction, of course, the reasoning behind the choice was communicated to the students to help them appreciate the challenges.

2.5.3 The Choice of Operating System

CentOS is based on RHEL; RHEL is the basis for the Tri-lab Operating Systems Stack (TOSS), which is utilized at LANL. Since version 3, TOSS has moved to CentOS 7 for its base. RHEL/CentOS is also the base testing distribution for many hardware vendors like Mellanox who typically release drivers for the RHEL stack early. RHEL also has fewer kernel updates (at least traditionally) than many other distributions, and by default it has a longer maintenance window. RHEL is primarily used on several of LANL's non-supercomputer systems. The CSCNSI was designed as a skillset pipeline, so there are clear advantages in helping students become familiar with CentOS. Additionally, there are clear advantages to using an OS that is familiar to the project mentors; therefore, CentOS has continued to be the OS of choice.

2.5.4 Some Specific Application Examples

The Warewulf cluster provisioning system. Part of the curriculum, which was added after the 2nd year, was to execute a diskless boot of the compute nodes (after they had been statefully installed). There are many cluster managers out there now, but at the time, the primary players on the open source market were ROCKS [ROCKS 2016] and Perceus [Vuong 2012] (now Warewulf [Warewulf 2016]). The instructor team tested both, and Perceus was chosen for a few reasons: one – it is explicitly written for the purpose of booting clusters with diskless compute nodes (which was becoming commonplace); two - it was already seeing some utilization at LANL, so some of the mentors were familiar with it; and finally - ROCKS essentially uses RHEL's Anaconda [Fedora Project 2016] tool, which was already part of the stateful install that students perform as part of the curriculum before provisioning the computer cluster disklessly.

The Lightweight Directory Access Protocol (LDAP). To LDAP or not to LDAP, that is the question. LDAP is an excellent tool for user and group management. However, as several years' cohorts experienced, LDAP is anything but lightweight, particularly with respect to configuration and ease of use. What was expected to be an afternoon of enjoyment turned into a half week or more of agony for many teams, just in trying to get LDAP to function as intended. Typically, the struggle was not because the students were doing something majorly wrong, the devil was simply in the details. Due to student frustration and the fact that LDAP is not natively used in Warewulf, combined with the even more complicated configuration scheme used in OpenLDAP version 2.4 (with the lack of configuration files), LDAP was set aside and left as an extra exercise for student teams that progress faster than others. LDAP was still covered as a lecture topic, as it's an important tool in a system administrator's arsenal; however, with limited time allowed for each Boot Camp topic, the implementation of LDAP was reserved for the more adventurous teams. CSCNSI 2015 revived LDAP as a core topic. While it is still one of the more difficult topics of the Boot Camp, it also remains highly valuable for students as they transition into the workplace.

The Puppet configuration management tool. Puppet [Puppet 2016] is also an interesting open source choice. LANL – at the time, and still today – is a heavy user and supporter of cfengine [cfengine 2016]. One year, a CSCNSI student team was assigned as its research project to evaluate Puppet vs. cfengine for cluster configuration and maintenance. At the end of the summer, the result was clear. Puppet had a larger user community, was open source with a large developer base, and was easier to configure and understand. This led to the use of Puppet in the CSCNSI, as well as to an increased use of Puppet at LANL. [1]

2.5.5 Program Evolution: The Addition of Cluster Mini-Boot Camp

In 2015, an addition to the core CSCNSI program was proposed. A cluster Mini-Boot Camp was planned and executed as a separate skill-broadening intensive for new or early career HPC staff without or with limited previous computer cluster experience or background. The training format is a condensed, six-day version of the regular CSCNSI Boot Camp in which participants work under the guidance of an instructor through a series of lecture, demonstration, and hands-on implementation cycles to build and configure a working computer cluster. The 2015 pilot was well executed and well received, and two additional Mini-Boot Camps were executed in 2016, one immediately preceding and one immediately following the regular 10-week CSCNSI. The second included LANL Staff, other non-CSCNSI LANL students, as well as a Ph.D. graduate student team from New Mexico State University, demonstrating the broad applicability of the program.

2.5.6 The Only Thing Constant is Change: Considerations for CSCNSI 2017

The CSCNSI was conceived to be a continuously improving, living program. Technology is ever changing. Every year the CSCNSI evolves to stay on the cutting edge, and 2017 will be no exception. Approximately every four to five years, the CSCNSI affects a complete hardware refresh, which will happen in 2017 when the hardware purchased in 2013 will be replaced.

While the overall topics covered in the fast-paced Boot Camp and guest lectures are essentially the same now as they were when the CSCNSI program first started in 2007, it's important to note that the content is in no way, shape, or form static. As systems software (e.g., OSs) evolve, so must the accompanying course material. Most of the projects that students execute, deal with advanced topics at the bleeding edge of computation, and it is rarely an option to use older software. Therefore, before each summer, the lecture slides are updated to reflect the latest software revisions available in our standard OS distribution, as well as updated communications technology (for example InfiniBand (a very high throughput, very low latency computer networking communications standard) at various signal rates, from 10GBit/s to 108 GBit/s (Commonly referred to as SDR, DDR, QDR, FDR, and EDR), have all been covered as each higher speed technology has been released). It is one of the major strengths of the program. The CSCNSI does not have outdated textbooks, but a living, hands-on curriculum.

For example, RHEL 7 is being implemented as both management boxes and in TOSS, version 3, in clusters at LANL. RHEL 7 includes some major changes to the OS and requires understanding of both Security-Enhanced Linux (SELinux) and systemd -- two topics not currently in the CSCNSI course material. It is expected that CSCNSI 2017 will utilize CentOS 7 for the OS with added course material specific to the changes between CentOS 6 and 7, including but not limited to, SELinux, systemd, kernel changes, default filesystems (xfs replacing ext4), and changes in run levels.

As stated before, Warewulf only has a beta form for CentOS 7; however, as Warewulf is under continuous development at a sister national laboratory, LBNL, there is opportunity to reach

[1] Unfortunately, the practical use of a configuration management tool has gone missing from the course materials in recent years; the compressed schedule only allows a certain number of subjects, and bringing LDAP back into the main curriculum was probably the cause of practical configuration management dropping out. Both mentors and students are interested in including configuration management in the core curriculum. Whether it remains Puppet or something else is chosen (e.g., Chef) [chef 2016], is yet to be decided.

out for guidance and support. In addition, further evaluation to identify a best fit provisioning tool is underway. With this in mind, Puppet and other configuration management tools are also under evaluation for implementation into an ever-changing curriculum.

Other topics still currently under review include a change from DHCP/DNS (Dynamic Host Configuration Protocol (a protocol for automating the configuration of computers that use TCP/IP)/Domain Name System (a system used on the Internet to map the easily remembered names of host computers (domain names) to their respective Internet Protocol (IP numbers)) to Kea (a new open source DHCPv4/DHCPv6 server being developed by the Internet Systems Consortium to provide a very high-performance; extensible DHCP server engine for use by enterprises and service providers); monitoring tools besides Ganglia (note that LANL makes heavy use of Zenoss and Splunk - but both are relatively heavy, i.e., time consuming, topics for a 10-week program); and further review of other tool and system versions.

2.6 HPC Seminar Series

The technical broadening component is interwoven throughout the program in the form of a range of activities including facility tours and an extensive seminar series designed to provide the students with a broad exposure to topics across the spectrum of HPC. Practitioners actively engaged in building, supporting, designing, and using various aspects of HPC systems and resources at Los Alamos provide the HPC seminars. The seminar set has some variation from year to year to capture timely topics in addition to a core set of foundational topics. As an example, the HPC Seminar Series topics were as follows for the 2016 session.

- HPC Technology Overview
- The What and Why of HPC Division
- Mission Science at Los Alamos National Laboratory
- LANL Cluster Overview
- Resiliency and Reliability
- HPC Facilities Overview
- Cluster Configuration Management
- Glance and SLURM: User-Defined Image Management on HPC Clusters
- HPC Monitoring
- HPC File Systems
- HPC User Support and Consulting
- Design and Implementation of a Scalable Monitoring System for Trinity
- HPC Networking
- Neural Networks for Cross-Compiler Malware Detection
- Containers, WC, and Charliecloud
- UNIX Host Security
- Unclassified Weapons Program Overview
- Relational Statistical Study of Social Networks
- Storage (Campaign + HPSS)
- Design and implementation of a Scalable HPC Monitoring System
- DRM (Job Scheduling and Resource Management)
- Programming and Runtime Environment
- HPC Storage Systems: Serving Data to the Lunatic Fringe
- HPC Security
- LANL HPC History Project
- MarFS: A scalable POSIX File System/Data Lake on Cloud Objects
- HPC Student Research Mini-showcase

2.7 Team Research Projects

As previously noted, the team research projects are a core component of the CSCNSI. With Boot Camp complete, each team works with an assigned technical mentor or mentor team consisting of HPC subject matter experts to execute a real-world research project on their newly assembled and configured clusters.

Prior to the arrival of the students, the mentor teams have designed and proposed research projects. Ideally project proposals are solicited from mentors several months before student selection begins. Sometimes, project ideas for the following year spring immediately from results obtained in the current year. Early discussion of projects among mentors allows time to further vet project ideas that have a strong foundation but lack implementation detail.. Each project proposal requires a minimum of two mentors, a detailed summary of the main goals, and any specialized hardware beyond what is typically provided (e.g., additional switches or interface cards for an advanced networking project or a large disk unit for a storage project). The proposal should also outline baseline project goals, as well as some "stretch goals" in the event that the team completes the baseline goals ahead of schedule. This helps maintain project structure from start to finish, and allows teams to make independent progress even with diverse skillsets and drive.

The team projects are based upon highly relevant research questions with direct applicability in an HPC production environment. Projects are often very timely and a result of efforts needed by LANL divisions that may have been put on a backburner due to more urgent production issues or lack of staff time to complete. As such, projects are rarely, if ever, repeated. Most are true research projects by their nature, and this is clearly communicated to the students as they start each summer. Some students choose to continue work on their projects once they return to their school, or as an internship working with their mentors to explore the subject at a deeper level.

It is not uncommon for CSCNSI projects to have immediate and/or broader impact, as will be noted in Section 3 below. Example research projects over the past five years have included the following. A complete list of projects with links to project posters is available [CSCNSI Project Page 2016].

- InfiniBand Performance Characterization through Machine Learning (2016),
- Comparison of High Performance Network Options: EDR InfiniBand versus 100Gb RDMA Capable Ethernet (2016),
- Developing an HPC Monitoring Pipeline with Rabbit MQ (2016),
- Performance Studies of Parallel Erasure Coding on Clustered Micro Storage Servers (2016),
- Improving and Tracing Lustre Metadata (2015),
- Ceph: An Open Source Object Store (2015),
- Docker File System Isolation (2015),
- Docker: Testing the Waters (2015),
- Scalability of InfiniBand-Connected LNET Routers (2014),
- The Effects of SSD Caching on the IO Performance of Unified Storage Systems (2014),
- Backups Using Storage Clusters (2014),
- Monitoring I/O on Data-Intensive Clusters (2014),
- Cloud Management with Open Stack (2013),
- A comparison of Library Tracking Methods in High Performance Computing (2013),
- Functional Assessment of Erasure Coded Storage Archive (2013),
- Network Service Security Through Software Defined Networking (2013),
- Building a Parallel Cloud Storage System Using OpenStack's Swift Object Store (2012),
- Scalable Node Monitoring (2012),
- iSSH vs. Auditd: Intrusion Detection in High Performance Computing (2012),
- Gluster FS: One Storage Server to Rule Them All (2012).

2.7.1 Example Project Abstract (2016): InfiniBand Performance Characterization Through Machine Learning

The demand for user-defined software stacks (UDSS) within supercomputing centers in increasing. The demand is motivated by a variety of user needs which include software dependencies, portability of environments, build-time requirements, consistent environments, and usability. Linux container and Virtual Machines are among the popular modern options that support UDSS implementation. At Los Alamos National Laboratory, the High Performance Computing Division has developed a container implementation known as Charliecloud. In the individual desktop environment, Charliecloud's network performance was comparable to bare metal; however, Charliecloud and Virtual Machines were untested across a clustered environment. In this project, Team Cyan executed select benchmarks and evaluated the resulting data for Charliecloud and Virtual Machines across a 9-node cluster in order to contrast their network performances against bare metal at scale. Our results show a negligible difference in network performance between Charliecloud and bare metal. QEMU/KVM showed slightly worse network performance compared to bare metal in all benchmarks except latency [Easterday, et. al., 2016].

2.7.2 Example Project Abstract (2016): Comparison of High Performance Network Options - EDR InfiniBand versus 100Gb RDMA Capable Ethernet

InfiniBand (IB) has long been the network of choice for high performance computing (HPC). However, advancements in Ethernet and IB technology, as well as other high-performance networks, have made it necessary to analyze the performance of these network options in detail – specifically, we look at 100Gb Ethernet and IB. Advancements in Ethernet include upwards of 100Gb data rates and standardization of RDMA-over-Converged-Ethernet (Routable RoCE). Similarly, IB has introduced Enhanced Data Rate (EDR) hardware, which nearly doubles previous bandwidth, increasing it to 100Gb. The goal of this study is to compare and contrast these two options by looking at their respective bandwidth and latency performance, as well as message injection rates and deployment effort. This research will allow a clear definition of the solution space for the challenges and problems being faced by networks with HPC workloads [Erickson et. al., 2016].

2.8 Student Research Mini-showcase

The CSCNSI culminates in the presentation of team research results at a student research mini-showcase. The Mini-showcase is a technical symposium and poster session for LANL student interns at all levels (e.g., undergraduate, post baccalaureate, graduate, post-doctoral) engaged in computing work and research at LANL. The Mini-showcase provides a unique opportunity for the students to present their research and results, engage in professional networking with other students and staff, broaden their technical exposure and expertise, and celebrate technical achievement.

Specifically, the CSCNSI student deliverables are a technical poster for the poster session and a 20-minute technical presentation on their research for students and staff. Many students at the undergraduate level have not yet engaged in a research project, prepared a technical poster, or prepared a technical talk. In preparation for this event, and as part of CSCNSI's professional development goal, the students receive training sessions and coaching on how to prepare a poster, how to put together technical presentation material, as well as coaching in presentation delivery.

3. CSCNSI IMPACT

2016 marked the completion of the 10th Annual CSCNSI. Over the past decade, the core program has hosted 118 individual students participants from 37 distinct universities nationwide, including 5 New Mexico universities. Program graduates leave with a unique set of skills that are highly sought after at LANL and other locations that operate high performance computing facilities and/or engage in a wide variety of computing activities. We can point to multiple indicators of success in achieving the stated goals and objectives.

- Strong student intern pipeline: Over the past decade, fifty to sixty percent (50%-60%) of CSCNSI participants have returned to LANL for follow-on internships and post-baccalaureate appointments. Other graduates have returned to intern at NMC, work for other government

agencies (e.g., Sandia National Lab (SNL), Lawrence Livermore National Lab (LLNL), NASA Langley, US Army Research, Development and Engineering Command (RDECOM), and NCSA), have entered graduate school, or have moved on to computing positions in private industry.

- Strong early-career pipeline: Approximately forty percent (40%) of those students who have returned to LANL for follow-on internships have been converted to permanent staff positions (i.e., approximately 20% of total program participants have been converted to permanent staff).
- Broad institutional impact: Over the years, CSCNSI graduates have gone on to contribute at LANL in at least 25 different groups spanning 15 unique technical divisions.
- Broader impact: In addition to the Student Research Mini-showcase, student teams are encouraged to submit their research posters to external technical venues. Over the years, CSCNSI teams have had their posters accepted at the Student Poster Session at annual computing conferences [Weins et al. 2014; Stitt et al. 2015; Erickson et al. 2016; Easterday et al. 2016]. A CSCNSI 2011 student team submitted their results to the Graph 500 Committee at the 2011 Supercomputing Conference (SC11), which officially certified the team's test results, placing them at #27 and #33 on the 2011 Graph 500 list [Graph 500 List, 2011]. In addition, team research projects have attracted industry interest (e.g., the work of a 2010 CSCNSI team motivated a Mellanox white paper [Shainer 2011] and the work of a 2016 team has catalyzed substantive technical compatibility follow-on discussions between two networking switch vendors).
- Staff engagement (project mentors): Mentors find the student engagement and technical exchange to be revitalizing and beneficial. Over the years, dozens of LANL staff members have engaged as CSCNSI project mentors, with more than 20 returning to mentor multiple years. Staff are motivated to participate because they have the opportunity to influence, design, and mentor timely research projects that are directly relevant to their professional focus and tasking, often choosing questions that they want answers to, but have not had time to dig into themselves.
- Staff engagement (guest lecturers): In addition, each year the CSCNSI hosts on the order of 25 predominantly LANL staff guest lecturers to execute the HPC Seminar Series and associated lectures who share their expertise on a variety of computing, science, and professional development topics. The intent of the HPC Seminar Series is to provide the students with a broad exposure to topics across the spectrum of HPC. The presenters are practitioners actively engaged in building, supporting, designing, and using various aspects of HPC systems and resources at LANL. The seminar set has some variation from year to year to capture timely topics in addition to including a core set of foundational topics.
- Positive Exit Survey Results: Exit survey results consistently indicate a positive and highly productive participant experience. Graduates self-report a significant increase in breadth, depth, and technical skill base.
- CSCNSI graduates return: Over the years, a significant number (10-15%) of CSCNSI graduates have returned in subsequent years to serve in a variety of roles, including CSCNSI instructor, teaching assistant, guest lecturer and/or technical project mentor. We interpret this as an indicator of a healthy, relevant, and engaging program.
- Program duplication: In the summer of 2016, Lawrence Livermore National Laboratory piloted a program closely patterned on the CSCNSI at its site. Argonne National Laboratory (ANL) has also expressed interest in beginning a similar program. Both national laboratories have expressed that they are experiencing similar and significant HPC pipeline challenges and are looking for methods to address them [Meetings and conversations 2015, 2016].

- University Collaboration and Outreach: In 2016, LANL and NMC jointly explored various modes of university outreach and collaboration. The CSCNSI hosted a graduate student team consisting of Computer Science (CS) and Physics Ph.D. students from New Mexico State University to engage in Mini-Boot Camp training. The outreach experiment was quite successful, the students reported that they learned a lot and their advisors intend to use their newly gained skills to help administer research clusters in the CS and Physics departments. LANL staff also worked with the New Mexico Institute of Technology to develop a mechanism for CSCNSI students to receive college credit for completion of the program. The successful course pilot was offered as a third-year, three credit hour CS course during the 2016 summer session. Half of the CSCNSI students enrolled, successfully completed the course, and receiving college credit for their work in the 2016 CSCNSI.

4. KEYS TO SUCCESS

The CSCNSI is a carefully designed program where every aspect and component has been created and honed to achieve the overall objectives of bridging a widely recognized and organizationally felt HPC skillset gap and of growing a well-prepared and excited HPC entry-level workforce. The institute's long-standing and consistent success is due to its adherence to a core set of fundamental philosophies. These include a commitment to continuous improvement and a commitment to an essential set of teaching philosophies and methodologies that result in targeted, hands-on practical skill development, technical broadening, and professional development, which in turn uniquely prepare students to enter the HPC workforce.

From application process through execution to evaluation, the program is carefully organized to maximally engage busy, high-performing HPC subject matter experts and to do so with minimal burden and with an eye to providing value in return for their mentorship.

- The program removes the administrative burden of mentoring students by design. The CSCNSI framework absorbs the administrative tasks typically associated with student mentoring, including on-boarding, office space, required training, paperwork, scheduling, lecture room reservation, machine room equipment, machine room maintenance, desktop computers, poster printing, etc., removing the burden from the staff and thus allowing them to fully direct their efforts toward technical mentoring of the research projects.
- The program provides a full time instructor who leads the base instruction and is always available to the students during project time to keep projects moving and on track.
- The CSCNSI works closely with mentor teams to help them design, plan, and execute highly relevant and timely research projects that are directly related to or an extension of their professional interests and duties, making mentoring a win/win proposition for the mentors and their home organizations. In turn, the students are interacting with a wide range of HPC technical leaders and actively engaged expert practitioners (through their projects and through their attendance and interaction at the HPC Seminars), working on cutting edge equipment and engaging in relevant and timely research questions.
- Staff receive an extended interview with students who are attaining the exact HPC skill base and technical exposure that they are looking for, meaning that their recruiting burden is lessened and organizations receive well-prepared follow-on interns (CSCNSI graduates) that are ready to hit the ground running.
- Mentor teams consist of two to three staff members, thus lessening the individual mentoring burden and time commitment and allowing the mentoring interactions to fit more easily into busy work schedules, while ensuring that mentors are highly available to students and teams remain highly successful.
- The Boot Camp is designed to be comprehensive and challenging while ensuring that all students have the resources and the support to be successful.

Another key to success is the CSCNSI's focus on teamwork, both in the mentor teams and as a student engagement and teaching methodology. One of the most important components of the CSCNSI, both as predicted, and as reported by the participating students in surveys during and

after program completion, is the opportunity to work in a team, as well as the close interaction of students across teams. The planning and placement of the teams start months before the program commences. The CSCNSI is advertised nationally through job fairs, forums, conferences, recruiting trips, and by, most appreciated, word of mouth (we have many applicants from colleges of previous program graduates). Applications are accepted during approximately a six-week time-window. A committee of technical staff from LANL and the NMC reviews the student resumes, and the top students are selected and invited to the CSCNSI. A typical summer invites nine to twelve students for attendance. Once offers are accepted, student teams are formed, considering both student skill sets, project needs, and typically an attempt to place unique schools (if possible) in each team. This increases the diversity of the teams and makes sure that teams are as well prepared as possible for the research project portion of the program. A good example for placement consideration is a project that requires heavy coding; students with a stronger background or preference for coding would likely be directed to that team.

The cluster building exercise, our signature Boot Camp, proves to be much more than just a crash course in Linux and systems administration. It is also an incredibly useful tool in team building, as each team is taken through a myriad of obstacles, and a very intense start of the summer. This helps to set the team up for productive and positive interactions during the research phase. In the early CSCNSI offerings, each team was introduced to their project and mentors before Boot Camp, but they were not allowed to do project work prior to the completion of Boot Camp. This allowed mentors to be introduced to students earlier, and they could also be involved in Boot Camp as extra support to the instructor team during exercises. Beginning with the 2015 and 2016 summers, those introductions have been held until after the successful completion of Boot Camp to allow students to focus on cluster building without distraction. In both scenarios the students have striven to build clusters correctly in order to successfully utilize them for their known/unknown project. They are motivated to work to build as stable and reliable cluster as possible to facilitate their project work and avoid lost project time due to cluster rebuilds and fixes. As Boot Camp concludes, none of the teams want to be the one still fixing their cluster, but rather, they want to be launching their research projects.

In the past few years, we have introduced non-technical, team building courses and training to further encourage team building and an understanding of different working styles. In particular the 2016 exercise evaluated each student's working style (through survey) and interactively set out to develop ways to interact, given the differences between team members, and to better understand and value and benefit from similarities and differences and varied problem solving approaches.

The CSCNSI instructor teaching philosophy is essentially "baptism by fire." There is a large amount of information to digest in a short amount of time, a need to divide and conquer, and a great need to share knowledge between team members. Questions are not answered until students have at least tried Google first. "Have you tried Googling that?" is a very common answer in the first week of Boot Camp. The idea is to give students hints at how to do things but not show them the solution. There are often many ways to get things working correctly. The students become familiar with a situation in which things do not immediately work. They experience both success and failure in a setting that is a microcosm of a real world production-computing environment. CSCNSI provides real, latest generation tools for students to work with (for instance, many have never heard of 100 Gbit/s network switch let alone InfiniBand interconnects). While common in a DoE national laboratory environment, they are almost unheard of in an academic setting, simply because brand new technology can be cost-prohibitive.

5. PROGRAM REPRODUCIBILITY

The CSCNSI had humble beginnings. It was born of a desire to solve a persistent problem in LANL's HPC workforce pipeline. Its goals were specific to filling a system administration focused, entry-level workforce gap that was deeply felt within computing organizations at Los Alamos National Laboratory. Thoughts of designing a program that could be reproduced beyond LANL to address a national gap more broadly were not a part of the early goal set or discussions. However, inquiries about the program have continued to grow, as has the realization that other

national laboratories and organizations are also struggling with very similar skillset gaps and pipeline challenges.

Informally, LANL has openly shared its experiences and lessons learned in creating and evolving the CSCNSI over time. In the summer of 2016, LLNL piloted a program closely patterned on the CSCNSI at its site, but modified to target students even earlier (some as early as high school level). We are looking forward to meeting with LLNL to discuss their experiences, successes, and challenges. ANL has also expressed interest in beginning a similar program. We intend to continue those conversations and reach out more broadly. Conversations with LLNL and ANL and others indicate that there is interest in replicating full summer programs of this type elsewhere because the target skills are broadly applicable, hard to find, and in high demand. Funding and facilities and equipment to support duplication are often central topics of discussion in those conversations.

The Mini Boot Camp adaptation and associated pilots has shown great potential in terms of reproducibility, applicability, and outreach, and we are continuing to engage in conversations with the NSF and others about these possibilities. For instance, NMC is actively pursuing a mechanism through PRObE to recycle decommissioned cluster equipment from LANL to create and install modest-sized clusters at, for instance, underserved universities throughout the nation. Also, based on a small survey conducted, many universities (not just the underserved) would gladly accept donations of gently used computer equipment in order to boost their on-site computing capabilities without significant investment. We believe that an on-site Mini Boot Camp type educational component to develop the in-house knowledge to maintain the clusters and sustain this deployed computing capability would be of great value.

Our experiment this past summer in which we hosted a multi-disciplinary team of graduate students to participate in a Mini Boot Camp is an indicator that there may also be significant outreach opportunity and cross-training potential in establishing repeatable Mini Boot Camps through NMC for a broad spectrum of external participants, including research teams seeking to install and maintain research specific clusters or computing faculty seeking to initiate a cluster administration curriculum at their community college or university. The Ph.D. physics student who participated this past summer specifically noted his advisor's motivation for sending him and his own motivation for attending were rooted in the realization that almost every field of modern physics requires numerical computations and simulations which demand large amounts of computing power to enable the parallelization of calculations. He noted that the availability of high-performance computing resources and small-scale clusters enable new approaches to challenging scientific questions and that his research group was seeking the knowledge to administer and maintain such clusters to further its research goals. We have strong indication of external interest in the CSCNSI program, and we will continue to activity engage in conversations and activities to share our experience and to address and narrow the gap.

6. CONCLUSIONS

The HPC skillsets required at national laboratories are many and must address multi-faceted needs, including but not limited to computer architecture, operating systems, algorithms, application development, data analytics, machine learning, computational modeling, system architecture, networking, storage, system monitoring, programming languages, workflow, system administration, facilities and operations. It is unlikely that any single program or initiative or university curriculum will ever be able to fill all of these needs in isolation or that students will gain these skills on their own. Hence the necessity for continued effort, collaboration, programs, and innovative initiatives continues to grow.

Our metrics indicate that the CSCNSI has been successful in terms of its objectives to create strong student interns and an early-career pipeline for LANL that provides both HPC-specific technical skills and professional skills well suited for the HPC environment. Both the organization and the student participants benefit from what is essentially an extended interview to examine opportunity and fit. Students are able to network to find opportunities of specific professional interest. Over the past decade, CSCNSI graduates have returned for follow-on internships at a

high rate, integrated well into their chosen opportunities, and have been converted to career staff at LANL at a high rate.

This paper described the motivation and genesis that led LANL to think creatively and to design something that existed nowhere else, remains unique, and is now being patterned. Underlying design philosophies and principles were presented and were shown to link directly back to the generation of desired pipeline objectives and results. A detailed discussion of program evolution grounded in an underlying philosophy of continuous improvement was provided with specific implementation examples. In the end, the CSCNSI has been successful far beyond expectations and continues to create opportunity and generate success for its students, its graduates, the staff mentors, and LANL's HPC Division. Students comment and say, "I've learned more in these nine weeks of summer than I did in my first two years of college." The CSCNSI Boot Camp provides both foundation and springboard on multiple levels. Through the course of history, CSCNSI has been — in terms of recruiting and retention — one of the most successful student programs at Los Alamos National Laboratory. LANL is an organization that prides itself on its student programs and boasts about the large fraction of current staff members who started as LANL students. CSCNSI is proud to be part of that legacy, and we see no signs of slowing as the need for qualified computer systems professionals will continue to increase as high end computing becomes more ubiquitous within both industry and government.

ACKNOWLEDGMENTS

The authors would like to thank all of the LANL staff mentors who are the engine behind the continued success of the CSCNSI, past instructors, including Andree Jacobson, Dane Gardner, Matthew Broomfield, and Jarrett Crews, the CSCNSI Teaching Assistants, as well as all of the students who have participated in the program and provided valuable feedback to help us evolve and improve the program. The co-authors are grateful to Gary Grider, an innovator to his core, who articulated the critical entry-level HPC pipeline need and then reached out to partner with other creative and adventurous souls to co-found the CSCNSI to address it.

REFERENCES

NAS Report. 2016. Future Directions for NSF Advanced Computing Infrastructure to Support U.S. Science and Engineering in 2017-2020, National Academy of Sciences Final Report, National Academies Press. Washington, D.C., 2016. Doi: 10.17226/21866. Downloaded PDF, http://www.nap.edu/catalog/21886/future-directions-for-nsf-advanced-computing-infrastructure-to-support-us-science-and-engineering-in-2017-2020

ASCAC Report. 2014. Top Ten Exascale Research Challenges, DOE ASCAC Subcommittee Report. February 10, 2014. Downloaded PDF, http://science.energy.gov/~/media/ascr/ascac/pdf/meetings/20140210/Top10reportFEB14.pdf

Trinity Press Release. 2015. National Nuclear Security Administration Ensures Safety and Security of U.S. Nuclear Stockpile with Cray XC40 "Trinity" Supercomputing Solution. Trinity Cray Press Release, 2015. http://science.energy.gov/~/media/ascr/ascac/pdf/meetings/20140210/Top10reportFEB14.pdf

Graham, S., Smir, M. and Patterson, C., 2005. Getting up to Speed: The Future of Supercomputing. Technical report, National Research Council of the National Academies.

Congressional Hearing. 2013. America's Next Generation Supercomputer: The Exascale Challenge: Hearing before the Subcommittee on Energy, Committee on Science, Space, and Technology," House of Representatives, One Hundred Thirteenth Congress, First Session, Wednesday, May 22, 2013. Online: Accessed August 22, 2016. Http://purl.fdlp.gov/GPO/gpo40360

The Next Platform. 2016. Exascale Timeline Pushed to 2023: What's missing in Super Computing? Web page editorial. The Next Platform, April 27, 2016. Accessed August 22, 2016. http://www.nextplatform.com/2016/04/27/exascale-timeline-pushed-2023-whats-missing-supercomputing/

National Security Science. 2013. "The Tip of the Iceberg: Say, what is "under the floor" of a supercomputer?" National Security Science, April 2013. Available online: http://www.lanl.gov/discover/publications/national-security-science/2013-april/_assets/docs/under-supercomputer.pdf

Liu, N., Cope, J., Carns, P., Carothers, C., Ross, R., Grider, G., Crume, A., and Maltzahn, C. 2012, On the Role of Burst Buffers in Leadership-Class Storage Systems. IEEE 28th Symposium on Mass Storage Systems and Technologies, April 2012, pp.1-11.

Laros, J.H. III, Pedretti, K., Grant, R.E., Olivier, S., Levenhagen, M., DeBonis, D., Pakin, S., Martin S., Kappel, M., Falde, P. 2016. ACES and Cray Collaborate on Advanced Power Management for Trinity, 2016 Proceedings of the Cray User Group, London, May 2016. https://cug.org/proceedings/cug2016_proceedings/includes/files/pap107.pdf

New Mexico Consortium. Online: Accessed August 25, 2016. http://newmexicoconsortium.org/

CHEF, Online: Accessed August 25, 2016. http://www.chef.io/

ROCKS Clusters. Online. http://www.rocksclusters.org/wordpress/

http://www.rocksclusters.org/roll-documentation/xen/5.4.3/using-virtual-clusters.html

Anthony Vuong <avuong1@uci.edu> & Harry Mangalam harry.mangalam@uci.edu, The Perceus Provisioning System, version 1.14, Jan 21, 2012. Online. http://moo.nac.uci.edu/~hjm/Perceus-Report.html

CentOS. Online: Accessed August 25, 2016. https://wiki.centos.org/About/Product

Redhat. Online: Accessed August 25, 2016. https://access.redhat.com/articles/3078

Slashdot. Online: https://linux.slashdot.org/story/09/07/30/130249/CentOS-Project-Administrator-Goes-AWOL?from=rss

Slashdot. Online: https://linux.slashdot.org/story/09/08/01/1443221/centos-administrator-reappears

Warewulf. Online. Accessed August 25, 2016. http://warewulf.lbl.gov/trac

Fedora Project. Online. https://fedoraproject.org/wiki/Anaconda

CF Engine. Online. https://cfengine.com/

Puppet. Online. https://puppet.com/

Garth Gibson, Gary Grider, Andree Jacobson, and Wyatt Lloyd. 2013. PRObE: A Thousand-Node Experimental Cluster for Computer Systems Research. USENIX ;login:, Vol 38, No 3, June 2013.https://www.usenix.org/publications/login/june-2013-volume-38-number-3/probe-thousand-node-experimental-cluster-computer

The Graph 500 List. Official 2001 Results. http://www.graph500.org/results_nov_2011

Shainer, G., 2011. Los Alamos National Labs and Student Team Validate Performance Benefits of RDMA over Converged Ethernet (RoCE). Feb 21, 2011, available online. http://www.techonline.com/electrical-engineers/education-training/tech-papers/4228835/los-alamos-national-labs-and-student-team-validate-performance-benefits-of-rdma

National Center for Women & Information Technology Fact Sheet. https://www.ncwit.org/ncwit-fact-sheet

Grace Hopper Celebration of Women in Computing. http://gracehopper.anitaborg.org

ACM Richard Tapia Celebration of Diversity in Computing. http://tapiaconference.org

CSCNSI Student Research Project webage. 2016. http://www.lanl.gov/projects/national-security-education-center/information-science-technology/summer-schools/cscnsi/student-projects.php

Thomas Stitt, Amanda Bonnie, Zach Fuerst. 2015. Virtualizing File Transfer Agents for Increased Throughput on Single Host; Poster; Supercomputing Conference, SC'15 http://sc15.supercomputing.org/sites/all/themes/SC15images/tech_poster/tech_poster_pages/post192.html

Carson L. Wiens, Joshua M. C. Long, Joel R. Ornstein. 2014. I/O Monitoring in a Hadoop Cluster; Poster; Supercomputing Conference, SC'14. http://sc14.supercomputing.org/sites/all/themes/sc14/files/archive/tech_poster/tech_poster_pages/post144.html

Kari Erickson, Luke Kachelmeier, Faith Van Wig. 2016. Comparison of High Performance Network Options: EDR InfiniBand versus 100Gb RDMA Capable Ethernet; Poster; accepted to the Supercomputing Conference, SC'16. (Technical project mentors: Susan Coulter and Howard Pritchard).

Hunter Easterday, Jordan Ogas, Adam Smith. 2016. InfiniBand Performance Characterization Through Machine Learning; Poster; accepted to the Supercomputing Conference, SC'16. (Technical project mentors: Timothy Randles and Reid Priedhorsky).

Carolyn Connor, Andree Jacobson, Amanda Bonnie, Gary Grider. 2016. Next Generation HPC Workforce Development: The Computer System, Cluster, and Networking Summer Institute. EDUHPC 2016 Workshop held in conjunction with Supercomputing Conference SC'16. Salt Lake City, UT. November 2016.

CSCNSI Project page. Online: Accessed October 6, 2016. http://www.lanl.gov/projects/national-security-education-center/information-science-technology/summer-schools/cscnsi/student-projects.php

Submitted August 2016; accepted September 2016; revised October 2016.