INFORMATION SYSTEMS EDUCATION JOURNAL

Volume 16, No. 6
December 2018
ISSN: 1545-679X

In this issue:

iscap
Information Systems & Computing Academic Professionals

The **Information Systems Education Journal** (ISEDJ) is a double-blind peer-reviewed academic journal published by **ISCAP** (Information Systems and Computing Academic Professionals). Publishing frequency is six times per year. The first year of publication was 2003.

ISEDJ is published online (http://isedj.org). Our sister publication, the Proceedings of EDSIGCON (http://www.edsigcon.org) features all papers, panels, workshops, and presentations from the conference.

The journal acceptance review process involves a minimum of three double-blind peer reviews, where both the reviewer is not aware of the identities of the authors and the authors are not aware of the identities of the reviewers. The initial reviews happen before the EDSIGCON conference. At that point papers are divided into award papers (top 15%), other journal papers (top 30%), unsettled papers, and non-journal papers. The unsettled papers are subjected to a second round of blind peer review to establish whether they will be accepted to the journal or not. Those papers that are deemed of sufficient quality are accepted for publication in the ISEDJ journal. Currently the target acceptance rate for the journal is under 40%.

Information Systems Education Journal is pleased to be listed in the Cabell's Directory of Publishing Opportunities in Educational Technology and Library Science, in both the electronic and printed editions. Questions should be addressed to the editor at editor@isedj.org or the publisher at publisher@isedj.org. Special thanks to members of AITP-EDSIG who perform the editorial and review processes for ISEDJ.

Information Systems Education Journal

The Enhanced Virtual Laboratory: Extending Cyber Security Awareness through a Web-based Laboratory

Michael Black
mblack@southalabama.edu

Debra Chapman
dchapman@ southalabama.edu

Angela Clark
amclark@ southalabama.edu

School of Computing, Information Technology
University of South Alabama
Mobile, AL 33509, USA

Abstract

The Enhanced Virtual Laboratory (EVL) is the product of a Department of Defense grant to enhance outreach, research, and education for cyber security. Through web-based laboratories, EVL allows users to remotely experience interactive content with virtual machines inside a modern web browser providing constructivism-based, student-centered, active-learning instructional activities. Additionally, EVL provides a framework for content centralization that allows designated educators to share their knowledge and techniques of cyber security. By providing users an add-on free environment requiring no additional software to be installed, EVL provides a new and improved method of remotely delivering cyber security content to different users with variable depths of security knowledge.

Keywords: virtual lab, teaching lab, labs, web based

1. INTRODUCTION

It is well known that the United States is experiencing a critical shortage of qualified Information Assurance (IA) professionals. President Barack Obama (2009) stated that the "cyber threat is one of the most serious economic and national security challenges we face as a nation." In February 2014, the Obama Administration announced the launch of the Cybersecurity Framework, through Executive Order 13636, which directed NIST to develop a framework to serve as a guide for organizations hosting critical infrastructure to enhance their cybersecurity (Whitehouse, 2013; Obama 2013). Demand for trained cyber security professionals has outpaced other IT jobs by a significant margin. As noted in the US Department of Labor Occupational Outlook Handbook, the job outlook for 2012-22 for information security analysts is expected to grow about 37%, much faster than average. (Bureau of Labor Statistics, 2014). Formal cybersecurity education programs have been established as part of the National Initiative for Cybersecurity Education (NICE) as well to support increases in STEM literacy and build skills and competencies for the National workforce. (NICE, 2010) One method to attract well-qualified students to the field of IA is through experiential learning opportunities. (Maxim and Elenbogen, 2009; Verma, 2011; National Research Council, 2011).

The Enhanced Virtual Laboratory (EVL) is a content-driven web-based environment that delivers centralized cyber security content alongside live virtual machines for an interactive and seamless educational experience. EVL was developed as part of a Department of Defense capacity building grant - awarded to extend the institution's cybersecurity outreach, research, and education efforts. Through this system, educators can centralize their knowledge and techniques of cyber security. Users can obtain this shared content through a web interface to experience an environment that provides guided tasks alongside live, isolated virtual machines. This experience allows a user to actively learn, in a constructivist-based format, while using the actual environment instead of pre-defined emulators. Improved learning takes place through the use of authentic *real-world* relevant tasks and problems that require students to expand their knowledge through active problem solving (Ertmer & Newby, 2013). The integration of the technology with a structured learning activities provides for technology-based, student-centered educational opportunities that put students in charge of their own learning (Gayton & Slate, 2002-2003). Addtionally, EVL is fully functional inside any modern web browser without requiring extra add-ons or client-side software. By using this virtual laboratory environment, resources are not limited to a single use and can be accessed remotely, thereby extending the institution's Information Assurance outreach capabilities. The virtual lab can at any time be re-tasked to support the desired target audience in a timely manner. EVL provides a new and improved method of educating users about cyber security and computing related subjects.

2. BACKGROUND

Creating environments that facilitate active learning has become a priority for educational institutions, government agencies, and private corporations. Not only does active learning help maintain interest, especially with younger learners, but it also enables reinforcement of the material that is presented and helps provide a meaningful context, facilitating better knowledge retention (Perez-Sabater, et al. 2011; Korwin and Jones, 1990). Specifically, technology-based learning environments have been shown to create student-centered educational environments that provide students the opportunity to develop a sense of control and individual responsibility for their own individual learning (Tsai, 2012). Just as STEM programs in the form of field trips or summer camps provide active learning experiences to expose students to areas of interest such as biology and engineering, laboratory experiences in computing can expose students to the areas of cyber security, promoting inquiry-based learning, allowing exploration of these areas in greater depth. Also, laboratory modules can be used to spread general cyber security awareness, helping to increase knowledge regarding safe computing practices to the public at large.

Constructivist learning theory contends that students are responsible for creating their own knowledge and learning based on their individual experiences. Each student creates their own meaning through individual interpretation of the instructional materials (Ertmer & Newby, 1993). Meaning and knowledge is situationally determined as it is created by each learner as new information is integrated into their existing knowledge base. The student determines what they learn based on the information received from the outside world (Koohand, Riley, & Smith, 2009). Effective learning takes place when students apply existing knowledge to new or different situations. This requires learning tasks to be considered relevant and realistic to the student (Ertmer & Newby, 1993). Learning is an adaptive process that is based on prior learning, new experiences, and social interactions. The primary focus of constructivism is problem solving and higher order processing, not simply data representation (Koohang et al., 2009).

Hands-on learning in a lab environment provides constructivist-based activities that allows learners to experiment with different relevant *real-world* scenarios and to test configurations without the detrimental consequences of practicing security techniques in a live domain. Realistic experimentation of security concepts can be costly regarding hardware and software. While there are a myriad of third-party training solutions that are available, many of these products are not customizable for institutions to tailor as they wish. Also, many of these products require the user to install additional software to their machine and may be costly to obtain. Current products are built either to guide users through a scripted scenario or to simulate security techniques, thus limiting the ability to explore and experiment. These solutions are not practical when being used to provide an introduction to cyber security due to their heavy initial investment.

Physical laboratories require substantial resources regarding the financial investment in hardware and software as well as setup and configuration. Once a physical lab has been configured, it is usually dedicated to a small group of individuals and a specific learning objective. This does not allow for repurposing a lab promptly for other learning opportunities. Physical space and energy requirements paired with computing equipment cost makes it difficult for many institutions to justify the capital expenditure, especially when considering the respectively small number of students which would receive benefit from the physical computer lab. Also, individuals must be present to use the resources in the physical lab.

Another important caveat in teaching security-related concepts is the need for isolation. Isolation is extremely important to prevent any malicious use of the machines to the public Internet. Also, misconfiguration of network services by users learning security techniques also poses vulnerabilities. However, physical isolation means that internal machines must be accessed onsite, which creates a geographical limitation in delivering training and education.

Related work

The use of virtual laboratories is not a new concept (Hay, Dodge, & Nance, 2008). Organizations have leveraged the use of virtual laboratories for quite some time, employing a wide variety of approaches. As educational institutions with cyber security programs have also had to address increased growth in enrollment and constrained budgets, virtualization has enabled many to extend their learning experiences to support a larger number of individuals. These solutions not only allow these institutions to extend their resources to serve more students, but they also provide a method of extending the classroom and physical laboratory curriculum with a distance learning approach.

Training well-qualified cyber security professionals requires significant, meaningful hands-on active-learning laboratory experiences to build cyber skills to defend networks and to protect the nation's infrastructure. As noted by Zlateva et al., (2008) "…in few fields is the contrast between theory and practice as stark and as important to reconcile as it is in information security: cryptographic algorithms draw on the most abstract branches of mathematics while their correct (or incorrect) application decides vital problems ranging from the confidentiality of the nation's critical infrastructure to the privacy of personal information."

Padman et al. (2002) stated that "One of the main impediments to establishing an IA program is the requirement of a laboratory facility that will reinforce concepts taught in class with hands-on experiences." One early implementation illustrating the need to virtualize the cyber security curriculum started in 2001 by the United States Military Academy at West Point (USMA). The Information Warfare Analysis and Research Lab (IWAR) allowed students to practice theoretical topics on virtual machines inside a private network. However, IWAR was only accessible through local workstations inside a closed environment.

Other implementations, such as the Rochester Institute of Technology's solution, have made use of remote desktop protocol (RDP). This solution required the use of Microsoft Terminal Services and Remote Assistance to be employed for students to connect to running virtual machines. This configuration necessitated extra client-side software and for users to know the IP address of a specific virtual machine to connect to the system (Border, 2007).

The Advanced System Security Education Research and Training (ASSERT) lab created by the University of Alaska at Fairbanks provided cyber security education through distance learning by accessing virtual machines using a web-based portal (Nance, et al. 2009). ASSERT gave users the ability to practice cyber security techniques but did not provide detailed instructions alongside the virtual machine to assist users. A system that could provide users with detailed lesson plans would allow that user to further their understanding of topics, which are being practiced inside the virtual machine. The Open University of Catalonia employed a system to teach networking to remote students through the development of the Virtual Networking Laboratory (VNLab). This system employs a combination of several different systems to provide a user with a laboratory experience, which includes Cisco NETLAB+ (Prieto et al., 2008). Although many individual systems can be combined to create an e-learning system, a remote laboratory can be built to deliver all the necessary functionality without depending on other systems and vendors.

3. OBJECTIVES

A fundamental goal of the EVL project was to provide the institution with a means to support active-based learning activities as outreach to prospective and current students, as well as the DoD and other federal, state, and local agencies through our cyber security center. Once fully completed, training modules will be available for DoD training centers, government agencies, and other CAE institutions that do not have their own virtualization infrastructure. By using a virtual environment, the lab is not limited to a single use and also allows the institution to avoid many of the hardware and geographic limitations of physical computer lab environments.

To accomplish these goals, the EVL project had three main objectives: 1) provide a fully web-based interface, 2) deliver authentic and isolated virtual machine environments, and 3) centralize and deliver content inside an environment with minimal complexity. Each of these is discussed further in the following subsections.

Web-Based Interface

EVL's web-based interface was designed to be accessible by any user, provides a control mechanism for virtual machines, and delivers a centralized content driven environment. EVL can operate in all modern web browsers that support HTML5 elements. The web-based interface allows all users to access the interface without needing specialized software or hardware remotely. Through the web interface, EVL provides server-side controls and client-side initiation for the virtual machine sessions by utilizing a customized engine, web-sockets proxy, and the noVNC project. Through this implementation, users experience a seamless, platform-independent session using isolated virtual machines without needing extra client-side software or add-ons. This environment prevents EVL from needing client-side administrative permission to operate. Lastly, the EVL interface allows educators to contribute and collaborate on content that can be shared to authenticated users. Additionally, these contributors are provided a specialized interface for customizing virtual machines to be connected with their content. The EVL interface is a fully featured product that allows users to receive dynamically rendered content that connects with live isolated virtual machines. Figure 1 (appendix) provides a graphical depiction of a content example and Figure 2 (appendix) illustrates the content creation interface. As shown in Figure 1 (appendix), the user is provided with detailed lesson plans explaining the topics which are being practiced inside the virtual machine. This allows the user to have the machine interface and the instructions side-by-side, eliminating the need for separate online or physical documentation.

True Virtual Environment

EVL uses a web-based interface to deliver embedded, isolated virtual machines to any authenticated user. Emulation can be used imitate aspects of a computer operating system to support specific learning objectives providing relevancy. Emulation can be a functionally acceptable technique but carries many far-reaching consequences. The administrative process associated with creating lab exercises when using emulation mandates that each exercise be manually developed to fulfill the goals of the exercise. Each lab exercise would be limited to a finite set of predetermined commands and responses. Virtualized operating systems give the user a *real-world* experience as they respond exactly as if the operating system was running on dedicated, physical hardware resources. Also, the user is presented with the same errors and responses, just as they would by using a physical machine. The web interface achieves this feature without needing additional client-side software or add-ons due to its customized engine. The customized engine masks the complexities of virtualization from the client-side to allow the user to focus on the learning objective. The engine is responsible for automating tasks associated with the virtual infrastructure. Traditionally, the overhead associated with the configuration of the educational environment has been a burden on faculty and students alike. In standalone virtual environments, educators usually provision virtual machines on a per student basis. The engine simplifies this process by dynamically provisioning virtual machines based on learning objectives rather than a per student basis.

Minimal Complexity

A top priority of the Enhanced Virtual Laboratory engine is removing the complexities found when interacting with virtual infrastructure. The ability to remotely access a standalone virtual environment can sometimes require the use of virtual private networks or utilize public IP addresses for connectivity. The engine simplifies this issue by aggregating connectivity to virtual machines through a dedicated proxy. The proxy manages the connections between the client and virtual machine without the overhead of specialized networking or use of proprietary applications distributed with the virtualization platform. EVL abstracts the complexities of delivering virtualization providing a seamless experience for any depth of user knowledge and centralizes contributed content from educators into one location. EVL's web interface uses its customized engine to auto generate and remove virtual machines as a lesson is started and completed. This abstraction prevents the user from needing to have any pre-existing knowledge of virtualization. Instead, EVL handles all functionality on the server side to give the user a straightforward experience. In standalone virtual environments, educators must provision virtual machines for each user. The engine simplifies this process by dynamically provisioning virtual machines based on a selected learning object rather than a per student basis. An educator is only required to associate a virtual machine with a learning exercise at the time the exercise is created. EVL is content driven by educators who collaborate and create learning modules in one centralized location. This allows educators to create the content once and deliver to many students without being constrained to certain types of client operating systems or configurations.

4. SOLUTION

EVL uses the latest web technologies to deliver a fully featured web interface and content management system (CMS). These technologies include the latest web programming languages including HTML5, CSS3, JavaScript, and JQuery. Through these updated web languages, EVL can deliver a responsive learning system.

Additionally, HTML5 utilizes new elements that are essential to EVL. Canvasses and web-sockets are the keys to the open source project noVNC by Kanaka. The EVL system makes use of another open source project as its CMS framework, Django. This Python-based framework allows EVL to render client-side HTML5 web pages through server-side Python functions dynamically. Django also provides database abstraction and standardization for EVL. Through Django, EVL users cannot directly query the database for security purposes, and EVL can continue to be updated without needing content developers to understand proprietary code.

Using Python as its server-side scripting language, EVL natively imports its engine’s python-based classes. Through the engine’s python classes, EVL generates, edits, and removes virtual machines on a XenServer hypervisor. Without needing any client-side action, the engine allows EVL to deliver any virtual machine to the user through a web sockets proxy. The proxy accepts and directs requests based on a unique identifier that is generated by the engine.

The entire process from request to virtual machine connection is completed in four steps. First, an authenticated user request specific content from EVL. Then EVL activates its engine by requesting it to generate virtual machines that have been assigned to the requested content. Third, the engine responds with the IP address, port number, and unique token identifier for the requested virtual machines. Then EVL responds to the client with the retrieved information in the form of an HTML5 web page. Lastly, the noVNC JavaScript code is executed in the background on the client side to start all VNC sessions into the virtual machines. When all the steps are completed, the client side web page has both the content and virtual machines embedded alongside each other.

As figure 3 demonstrates, a user activates the EVL CMS by navigating to the URL of the web page. Following the user properly authenticating with the system, data is retrieved for the user to make a task selection. This selection determines what specific data is retrieved and which virtual machines the EVL CMS will ask the EVL Engine to create. Once all of the data is retrieved a page is rendered and allows the user to interactively use the content alongside the virtual machines until that user is ready to start over. Together the CMS and engine work in unison to deliver a fully featured virtualization experience without being dependent on any extra client-side software or add-ons.

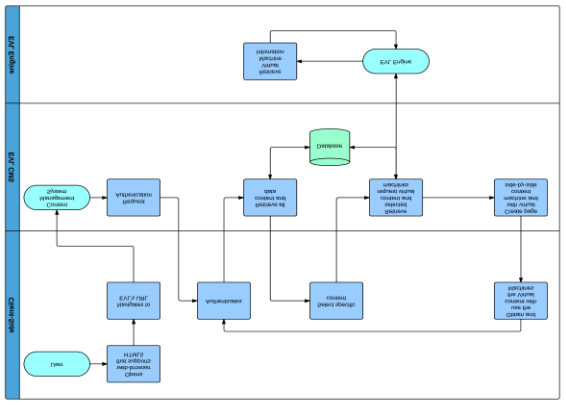


Figure 3: Conceptual Model

5. DISCUSSION

The EVL development group utilized a team-based approached that divided the project into two equal parts. One team designed the interfaces or the content management system (CMS), and the other team designed the engine. Working in unison, the teams would build features that were tested against the other team’s components until no errors or design flaws were found. This partitioned but collective - design process allowed each team to focus on separate objectives but still gather feedback and requests from each other. As shown in Figure 4, the content management system and the engine are two separate project components that combine to form the Enhanced Virtual Laboratory.

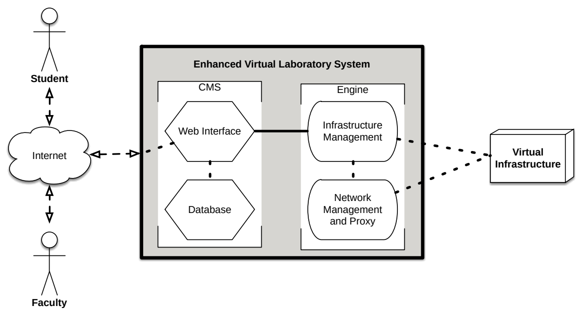

Student
Internet
Faculty
Enhanced Virtual Laboratory System
CMS
Web Interface
Database
Engine
Infrastructure Management
Network Management and Proxy
Virtual Infrastructure

Figure 4: Project Architecture

The teams have created a product that is usable from any location, system, or browser. However, EVL still has a few requirements for operation. First, the user must have an Internet connection. The system is fully web-based, meaning that to access the system a connection must be established via the Internet. Second, the user must be using a recent web browser. Updated browsers utilize HTML5 which is essential to delivering the embedded virtual machines. Third, EVL is content driven and must have educators adding content for learners to use. Fourth, the client-side firewall must allow requests and responses on port 8080. As all requests to the virtual machine VNC connections are routed through a proxy, a port other than the default web traffic port is used to make those connections.

6. USABILITY AND PERFORMANCE TESTING AND EVALUATION

EVL was pilot tested in a classroom of twenty students. During this test session, the students were instructed to complete a series of *real-world* relevant exercises that would guide them on how to set up a network with different operating systems. Each constructivist-based lesson progressively built on the previous one and increased the virtual machine count by one. Therefore, on the first lesson, the students were tasked with configuring a Linux network card using one virtual machine. Next, they configured Windows network cards using two virtual machines, a Windows 7 and Windows XP. Lastly, the students used a lesson that instructed them to configure both Linux and Windows network cards using three virtual machines: Ubuntu Linux, Windows 7, and Windows 8. Almost all students were able to complete the first and second lessons successfully. As students began to move into the third and final lesson, a bottleneck occurred where the engine did not connect the users and virtual machines. After reviewing the data collected during testing, within 31 minutes, 42 virtual machines were created and successfully connected to by students. However, it was found that 122 virtual machines were created that were never successfully connected to by students. The testing session revealed an issue in the engine's proxy service that has now been resolved. Future testing has been scheduled to confirm that the problem is resolved and that no other performance issues exist with the current configuration.

7. LESSONS LEARNED

EVL went through many design changes throughout its creation; however, the main goal to extend cybersecurity awareness stayed consistent. Each feature of EVL has been made to enhance accessibility, improve reliability, and simplify future contributions. For example, the EVL original design was completely propriety. This design provided all the needed features but made contributing to the system a complex and time cumbersome task. Therefore, the EVL scope switched to using an open source framework that would simplify the process of allowing multiple parties to add to the system in the future.

The EVL design still has certain limitations. With a finite amount of hardware resources, EVL can only deliver a certain number of virtual machines before exhausting the available resources. Also since the system is primarily built for outreach, there are no existing mechanics for grading or assuring that a user completed the content. Additionally, EVL does not allow virtual machines to persist among user sessions. This means when the user switches between content the virtual machines are refreshed. Although these limitations are inside the current version, with EVL's modular design updates can be implemented to lengthen the scope of the project.

8. CONCLUSIONS AND FUTURE WORK

EVL is a new and improved method of delivering specialized content alongside virtual machines for effective education, research, and outreach. EVL provides a fully OS independent and modern web-browser independent interface that educators can use to contribute or collaborate on content to build student-centered, active-learning, constructivist-based educational activities. EVL is a system that delivers a seamless experience since it uses the latest web technologies and server-side scripting to require very minimal resources from the audience's machine. The active-learner, constructivist-based, relevant activities provide effective hands-on learning. Additionally, EVL has a standardized framework which allows updates and feature additions to be contributed. These contributions can come in different forms. With more hardware resources, EVL can deliver more virtual machines to larger audiences. Since the EVL system works as a tool for centralization, contributors could add resources to the environment. Then they could use the system for distributing to their audiences without needing individual infrastructures. With more content contributors adding to EVL, the system can provide different types of education to a wider range of audiences. The EVL system was built for cybersecurity awareness outreach, but due to its nature of using live virtual machines, many different disciplines could be taught within the system, such as secure software engineering or applications development. EVL can support a community of educators from different disciplines working together to expand outreach, research, and education.

An additional area of interest for future study could be to examine how students feel use of this type of system impacts their learning and helps maintain or further their interests in the study of cyber security. Once EVL is expanded with additional training modules, this is the goal of the research team in understanding its effectiveness and impact on student learning outcomes.

9. REFERENCES

Border, C. (2007). The development and deployment of a multi-user, remote access virtualization system for networking, security, and system administration classes. SIGCSE Bull., 39(1), 576–580. doi:10.1145/1227504.1227501

Bruce R. Maxim and Bruce S. Elenbogen. (2009). Attracting K-12 students to study computing. In *Proceedings of the 39th IEEE international conference on Frontiers in education conference* (FIE'09). IEEE Press, Piscataway, NJ, USA, 119-123.

Bureau of Labor Statistics, U.S. Department of Labor, *Occupational Outlook Handbook, 2014-15 Edition*, Information Security Analysts, on the Internet at http://www.bls.gov/ooh/computer-and-information-technology/information-security-analysts.htm (last visited *March 03, 2014*).

C Pérez-Sabater, B Montero-Fleta, M Pérez-Sabater, B Rising (2011). "Active learning to improve long-term knowledge retention" Actas del XII Simposio Internacional de Comunicación Social, 75-79.

Ertmer, P., & Newby, T. (1993). Behaviorism, cognitivism, constructivism: comparing critical features from an instructional design perspective . *Performance Improvement Quarterly*, 50-72.

Gaytan, J. A., & Slate, J. R. (2002-2003, Winter). Multimedia and the college of business: A literature review. Journal of Research on Technology in Education, 35(2), 186-205.

Hay, B., Dodge, R., & Nance, K. (2008). Using Virtualization to Create and Deploy Computer Security Lab Exercises. In S. Jajodia, P. Samarati, & S. Cimato (Eds.), Proceedings of The Ifip Tc 11 23rd International Information Security Conference (Vol. 278, pp. 621–635). Springer US. Retrieved from http://dx.doi.org/10.1007/978-0-387-09699-5_40

Koohang, A., Riley, L., & Smith, T. (2009). E-learning and constructivism: From theory to application. *Interdisciplinary Journal of E-Learning and Learning Objects*, 91-109.

Korwin, Anthony R.; Jones, Ronald E. (1990). "Do Hands-On, Technology-Based Activities Enhance Learning by Reinforcing Cognitive Knowledge and Retention?" *Journal of Technology Education*, v1 n2 p26-33 Spr 1990.

Nance, K., Hay, B., Dodge, R., Seazzu, A., & Burd, S. (2009). Virtual laboratory environments: methodologies for educating cybersecurity researchers. Methodological Innovations Online, 4(3), 3–14.

National Initiative for Cybersecurity Education (NICE) (2010\) Relationship to President's Education Agenda, 19 April 2010, (last visited *March 18, 2014*) http://www.whitehouse.gov/sites/default/files/rss_viewer/cybersecurity_niceeducation.pdf

National Research Council. (2011) *Learning Science Through Computer Games and*

Simulations. Washington, DC: The National Academies Press, 2011.

Obama, Barack (2013). "Executive Order – Improving Critical Infrastructure Cybersecurity" Retrieved from http://www.whitehouse.gov/the-press-office/2013/02/12/executive-order-improving-critical-infrastructure-cybersecurity, (last visited *March 18, 2014*).

Office of the Press Secretary (2014). "Launch of the Cybersecurity Framework" Retrieved from http://www.whitehouse.gov/the-press-office/2014/02/12/launch-cybersecurity-framework, (last visited *March 18, 2014*).

Padman, Vikram, and Nasir Memon (2002). "Design of a Virtual Laboratory for Information Assurance Education and Research." Workshop on Information Assurance and Security. Vol. 1. 2002. 1555.

Prieto-Blázquez, J., Arnedo-Moreno, J., & Herrera-Joancomartí, J. (2008). An Integrated Structure for a Virtual Networking Laboratory. IEEE Transactions on Industrial Electronics, 55(6), 2334–2342. doi:10.1109/TIE.2008.921231

Schafer, J., Ragsdale, D. J., Surdu, J. R., & Carver, C. A. (2001). The IWAR range: a laboratory for undergraduate information assurance education. Journal of Computing Sciences in Colleges, 16(4), 223–232.

The White House. President Barack Obama (2009). Remarks by the President on Securing Our Nation's Cyber Infrastructure." Retrieved from http://www.whitehouse.gov/issues/foreign-policy/cybersecurity (last visited 03 Mar. 2014).

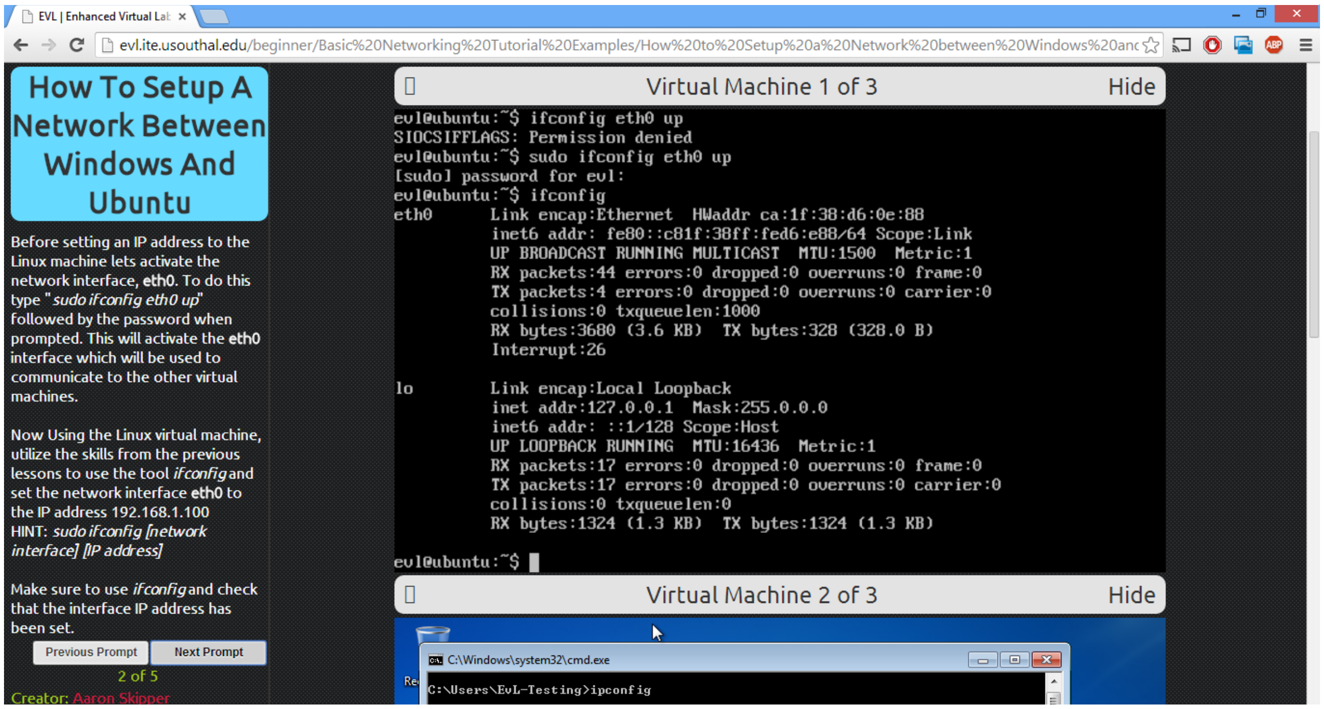
Tsai, S. C. (2012). Integration of multimeida courseware into ESP instruction for technological purposes in higher-education technology. *Educational Technology and Society, 15*(2), 50-61.

Verma, A; M. Talaiver; S. McKinney; D. Dickerson; S. Dwivedi; D. Chen (2011) "Attracting K-12 Students towards Engineering Disciplines with Project Based Learning Modules." *Proceedings of the ASEE Conference*, Vancouver, Canada, June 26-29, 2011.

Zlateva, T, L. Burstein, A. Temkin, A. MacNeil, and L. Chitkushev (2008). Virtual Laboratories for Learning Real World Security, Proceedings of the Colloquium for Information Systems Security Education, University of Texas, Dallas, TX, June 2008.

Appendices and Annexures

Figure 1: Content Example


EVL | Enhanced Virtual Lab
evl.ite.usouthal.edu/beginner/Basic%20Networking%20Tutorial%20Examples/How%20to%20Setup%20a%20Network%20between%20Windows%20anc
Virtual Machine 1 of 3
Hide
Virtual Machine 2 of 3
Hide
Previous Prompt
Next Prompt
2 of 5
Creator: Aaron Skipper

How To Setup A Network Between Windows And Ubuntu

Before setting an IP address to the Linux machine lets activate the network interface, **eth0**. To do this type "*sudo ifconfig eth0 up*" followed by the password when prompted. This will activate the **eth0** interface which will be used to communicate to the other virtual machines.

Now Using the Linux virtual machine, utilize the skills from the previous lessons to use the tool *ifconfig* and set the network interface **eth0** to the IP address 192.168.1.100
HINT: *sudo ifconfig [network interface] [IP address]*

Make sure to use *ifconfig* and check that the interface IP address has been set.

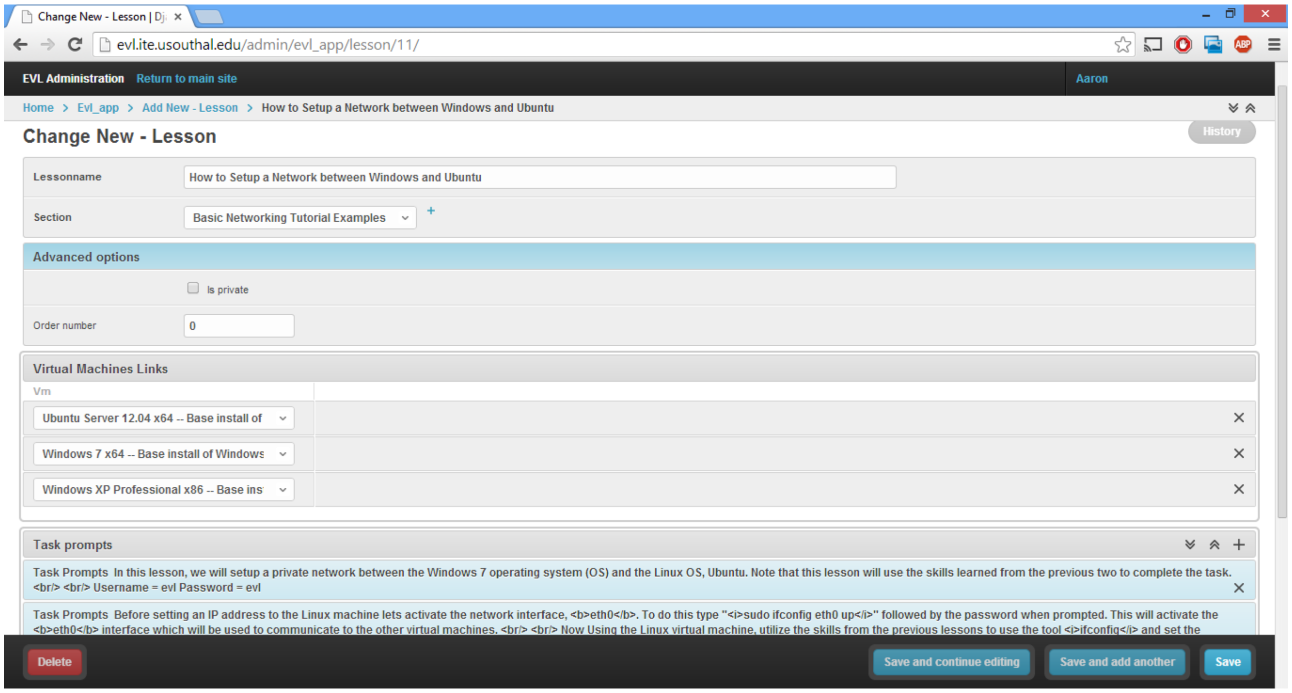
```
evl@ubuntu:~$ ifconfig eth0 up
SIOCSIFFLAGS: Permission denied
evl@ubuntu:~$ sudo ifconfig eth0 up
[sudo] password for evl:
evl@ubuntu:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr ca:1f:38:d6:0e:88
          inet6 addr: fe80::c81f:38ff:fed6:e88/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:44 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3680 (3.6 KB)  TX bytes:328 (328.0 B)
          Interrupt:26

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:17 errors:0 dropped:0 overruns:0 frame:0
          TX packets:17 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1324 (1.3 KB)  TX bytes:1324 (1.3 KB)

evl@ubuntu:~$
```

```
C:\Windows\system32\cmd.exe
C:\Users\EvL-Testing>ipconfig
```

Figure 2: Content Creation Example


Change New - Lesson | Dj
evl.ite.usouthal.edu/admin/evl_app/lesson/11/
EVL Administration
Return to main site
Aaron
Home > Evl_app > Add New - Lesson > How to Setup a Network between Windows and Ubuntu
History
Delete
Save and continue editing
Save and add another
Save

Change New - Lesson

Lessonname	How to Setup a Network between Windows and Ubuntu
Section	Basic Networking Tutorial Examples

Advanced options

☐ Is private

Order number: 0

Virtual Machines Links

Vm
- Ubuntu Server 12.04 x64 -- Base install of
- Windows 7 x64 -- Base install of Windows
- Windows XP Professional x86 -- Base ins

Task prompts

Task Prompts In this lesson, we will setup a private network between the Windows 7 operating system (OS) and the Linux OS, Ubuntu. Note that this lesson will use the skills learned from the previous two to complete the task. <br/> <br/> Username = evl Password = evl

Task Prompts Before setting an IP address to the Linux machine lets activate the network interface, <b>eth0</b>. To do this type "<i>sudo ifconfig eth0 up</i>" followed by the password when prompted. This will activate the <b>eth0</b> interface which will be used to communicate to the other virtual machines. <br/> <br/> Now Using the Linux virtual machine, utilize the skills from the previous lessons to use the tool <i>ifconfig</i> and set the

Low-cost Cluster Computing Using Raspberry Pi with Mathematica

Blake Jacobus
bjacobus@millikin.edu

RJ Podeschi
rpodeschi@millikin.edu

Tabor School of Business
Millikin University
Decatur, IL 62522

Abstract

The costs of higher education continue to rise as budgetary and financial pressures strain universities and its students. In particular, students in the sciences rely on computational software like Wolfram Research's Mathematica in their research and studies. The software and its associated syntax are highly useful tools and are necessary skills in the areas of engineering, mathematics, physics, and data science. However, the software is costly and is difficult for colleges and students to afford. With the advent of inexpensive credit card-sized computing devices like the Raspberry Pi and its partnership with Mathematica, the software can now be used at no cost. However, processing and speed are limited and performance is affected on a Raspberry Pi. Through the use of cluster computing, execution times of algorithms using Mathematica can be decreased while maintaining a lower cost than Mathematica's traditional licensing model. This research reports the design and configuration of a Raspberry Pi cluster for use with Mathematica in addition to the results of performance benchmark tests between algorithms executed on one node and four nodes. This work makes an important contribution to both information systems and science disciplines to decrease software licensing costs without sacrificing performance. Conveniently, this research project provided an opportunity for an undergraduate information systems major to learn and understand cluster computing in an experiential learning independent study project.

Keywords: Raspberry Pi, Mathematica, cluster computing, high performance computing, remote kernel processing

1. INTRODUCTION

Small private universities appeal to students across the world for various reasons. Smaller classes, closer relationships, and condensed campuses are among the most common benefits. However, a bantam budget is not likely to make the list of appealing traits for any school. Unfortunately, many institutions are forced to make sacrifices when it comes to funding, and studies centered around the management of information systems and technology suffer greatly from a lack of physical assets. Likewise, the cost of higher education continues to rise as students carry higher debt load upon graduation. While the average annual cost of a four year university in 1985 was $18,910, students are paying at least $37,990 as of 2015, an increase that has trended upwards every single year for the past three decades (U.S. Department of Education, 2016). Fortunately, the advent of the Raspberry Pi computing device has allowed many students across the world to hold a multi-core system in the palm of their hand for far less than the price of a textbook. The Raspberry Pi is affordable and appealing to learners and enthusiasts alike, but their primary drawback is the inability to process complex computations with any speed, allowing for faster execution times and opportunity for more iterative development.

Among those who depend on powerful processing units are mathematicians, engineers, and various scientists that utilize Mathematica, a "symbolic mathematical computation program" with the ability to process visual and aural data, generate 3D models, and a plethora of other functions used across a variety of fields (Trustees of the California State University, 2017). In the past, Mathematica has been a tool used only by very well-funded entities. The licensing for Mathematica, not unlike the equipment required to leverage its functionality, has a premium price tag. Many institutions with full funding for hardware may still find themselves restricted to licensing only a handful of workstations.

Raspberry Pi brought multicore computing to the eager hands of many financially challenged institutions and the implementation of a free Mathematica license for the Raspbian operating system allowed for the proliferation of computing research on a smaller budget. The Raspberry Pi was, however, unable to accommodate the resource-intensive features of Mathematica, requiring users to search for ways to optimize their small Raspberry Pi for more complex computations. This led to the utilization of Mathematica's remote kernel function, which allows Raspberry Pi devices to pool resources from other Raspberry Pi devices on the same network with a compatible Mathematica license to operate as a cluster. A cluster is defined as a group of similar or identical computer, connected by a computer network that pool resources to provide services or run applications (Burd, 2016). Configuring computers into a cluster can exponentially increase the processing speed. This clustering technique made it possible for students and professionals to distribute workloads across machines to solve complex equations and further expand the applications of the Raspberry Pi computer.

This research project was primarily conducted as an independent research project by a senior information systems major. This paper represents the outcome of a hands-on opportunity to better understand cluster computing and its potential benefits to higher education. This work presents the groundwork to build lab exercises to teach cluster computing in courses related to computer architecture, networking, and infrastructure.

This paper first reviews the Raspberry Pi, its uses, along with background information related to Mathematica. A detailed account of the process of constructing a multi-node Raspberry Pi cluster in conjunction with Mathematica's remote kernel function are provided. Results are shared to demonstrate the performance benefits achieved with Mathematica in a multi-node cluster environment. This work is an important contribution to the field for students and/or education institutions in search of a lower-cost, higher-performance environment for utilizing Mathematica and its computational features.

2. REVIEW OF LITERATURE

Raspberry Pi

In 2009, the Raspberry Pi Foundation was established at Cambridge University by six scientists including its leader, Eben Upton, to develop an ultra-low-cost computing (ULCC) platform to address the lack of interest in programming from grade-school age students (Andrews, 2013; Heeks & Robinson, 2013). The goal for designers was to keep the price of the credit-card sized motherboard to $35 to make it affordable to get in the hands of children in British schools (Harris, 2015). The most recent version of the Raspberry Pi released in 2016, version 3 model B, consists of the following specifications as seen in Table 1 (Raspberry Pi Foundation, n.d.).

Raspberry Pi 3 Model B Specifications
• Quad Core 1.2GHz Broadcom BCM2837 64bit CPU • 1GB RAM • BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board • 40-pin extended GPIO • 4 USB 2 ports • 4 Pole stereo output and composite video port • Full size HDMI • CSI camera port for connecting a Raspberry Pi camera • DSI display port for connecting a Raspberry Pi touchscreen display • Micro SD port for loading your operating system and storing data • Upgraded switched Micro USB power source up to 2.5A

Table 1.

Using system on chip (SOC) technology, advances in integration have effectively enabled manufacturers the ability to shrink an entire desktop onto this handheld motherboard. The motherboard, with the exception of the Broadcom graphics processor, is entirely open source and meant to run a slimmed down distributions of Debian or Arch-based Linux (Heeks, 2013). While requiring no fans or moving parts, the Raspberry Pi relies on a microSD card for its storage, and can connect to peripherals using USB or HDMI for graphics (Shuurman, 2015).

The Raspberry Pi, due to its low cost, has been successful in reaching underdeveloped countries by providing access to general computing and teaching programming skills to youth (Schuurman, 2015). British school systems now require that all primary school students learn key ideas of computer science and understand computational thinking to accommodate the increased demand for employees with technical acumen (Naughton, 2012). With millions of units purchased for the purpose of democratizing programing and computer science education, the Raspberry Pi's easy-of-use and small footprint has led to the unintended development of several initiatives. Home automation devices for security, digital signage, and a multitude of Internet of Things (IoT) are being tested and incubated through this low-cost platform (Edwards, 2013). As a result, the Raspberry Pi has become a lost-cost computer for teaching, research, and innovation.

Cluster Computing
A modern approach to systems architecture involves distributing computing resources (e.g. CPU and memory) across a set of nodes with identical or similar hardware, and communicate over a specialized network. Known as cluster computing, this methodology replaces previous-generation mainframe and high-cost supercomputers such as MPP (Massively Parallel Processors). Cluster systems are typically configured on commodity hardware, thereby creating cost efficiencies (Gropp, Lusk, & Sterling, 2003). Cluster systems have the advantage of being able to scale up as additional resources are needed as additional nodes can be added to the cluster. Cluster systems generally include six hierarchical layers which include: internetworking, computation, operating systems, compilers, distributing programming models, middleware, and applications (Fung, Li, & Myers, 2005).

Massive multi-processor servers require a large up-front investment while cluster configurations rely on off-the-shelf (or commodity) hardware and can be implemented for a fraction of the cost. Likewise, cluster systems scale well and can grow as resource needs (CPU and memory) grow. As such, the Raspberry Pi is an ideal candidate for architecting a cluster system for this particular project.

Mathematica
Wolfram Research, founded by Stephen Wolfram, developed and released Mathematica in 1988 to solve computational problems through the use of a graphical user interface (GUI) (Wolfram Research, 2017b). Wolfram's flagship product was developed so that the scientific, engineering, mathematical, and computing fields had a software platform to perform a vast array of computations, complete with its own syntax language. The software engine is capable of creating visualizations, performing data analysis, geometric computation or machine learning. Mathematica, as of July 2017, is available through a cloud subscription or a standalone desktop license and are priced at $575 per year or $1,150 per installed machine, which includes upgrades and support (Wolfram Research, 2017d).

The use of Mathematica is prevalent in higher education and can be found in a range of subjects including, but not limited to: computer science, engineering, mathematics, and physics. Previous research has cited its effectiveness in providing interactive simulations in chemical engineering courses (Falconer & Nicodemus, 2014), increased student outcomes in linear algebra (Rahmawati, et. al., 2017), and studying projectile motion in physics (Hutem & Kerdmee, 2013).

Raspberry Pi Meets Mathematica
In 2013, Wolfram Research released their Mathematica software as a free license when installed on a Raspberry Pi running an approved Linux distribution (Wolfram). This allowed anyone with the credit-card sized device access to the same computational platform as researchers in an effort to build the knowledge base in the educational systems. While Mathematica on a Raspberry Pi is functional, it has its limitations due to lower CPU power and available RAM than a typical desktop. To be able to perform computations with better response times, multiple Raspberry Pis are needed through Mathematica's clustering feature. Through the combination of Mathematica's feature-set and Raspberry Pis open and low-cost platform, the multi-note clusters can be built to increase processing and response times.

3. METHODOLOGY

Hardware Configuration

The Raspberry Pi cluster consists of four Raspberry Pi Model B devices configured on a vertical enclosure. The enclosure allows for proper cooling, secure transportation, and proper cable management (see Figure 1). The total cost of the four-note cluster procured through Amazon.com, including peripherals, cables, and display, was less than $500. The parts used in the four-node cluster are outlined in Table 2.

QTY	Item
4	Raspberry Pi 2 Model B V1.1 devices
1	GeauxRobot Raspberry Pi 4-layer Dog Bone Stack Enclosure
4	16GB microSD cards (FAT format)
1	microSD/SD Adapter
4	RPi AC/microUSB power adapters
1	6-outlet electrical strip
1	Encore 8 Port NWay 10/100 switch
5	CAT5e network cables
1	HDMI Cable
1	Monitor or television with HDMI port
1	Computer mouse
1	Computer keyboard

Table 2. Cluster Parts List.

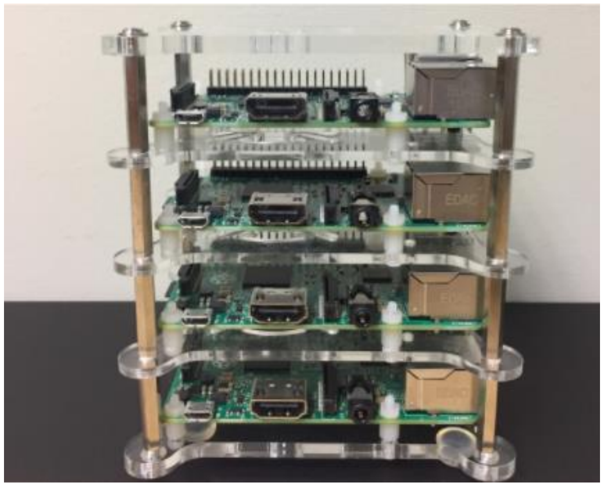


Figure 1. Raspberry Pi Enclosure

A Raspberry Pi device stores secondary data on one microSD card. The Raspberry Pi Foundation recommends using an 8GB SD card, but 16GB cards were used for this cluster to ensure ample storage space (Raspberry Pi Foundation, n.d.d). These cards must be formatted using FAT file system and an image of the latest Raspbian image must be installed on the SD cards. A table containing names, functions, and web sources for all software tools used to create this Raspberry Pi cluster is located in Table 7, found in Appendix A. The process for formatting and mounting the cards is as follows:

1. Format all four microSD cards using SDFormatter application
2. Download Raspbian image from Raspberry Pi Foundation
3. Write image to a single microSD card (known hereafter as RPi01) using Win32 Disk Imager
4. Configure Raspbian OS settings using `raspi-config` command (detailed in Software Configuration section below)
5. Read image from RPi01 using Win32 Disk Imager and write to Windows machine
6. Write image RPi01 to the three other microSD cards, known hereafter as RPi02, RPi03, and RPi04
7. Login to RPi02, RPi03, and RPi04 using PuTTY interface from remote workstation and use `raspi-config` command to change the machine names and passwords according to Table 3. All other configuration settings remain the same.

Software Configuration

8 Update	Wait for updates to finish
1 Change User Password	RPi01*, RPi02*, RPi03*, and RPi04*
2 Hostname	RPi01-04
3 Boot Options	B1 Desktop / CLI B4 Desktop Autologin Desktop GUI
5 Interfacing Options	Enable P2 SSH
6 Overclock	High*
A1 Expand Filesystem	
A3 Memory Split	16**
* Overclocking the CPU yielded minimal increases in processing speed. ** Memory Split allocates more or less RAM to GUI processing, allowing for more performance when processing equations and less graphical overhead.	

Table 3. Raspi-Config Settings.

Raspbian's operating system settings are configured using the raspi-config interface, which can be accessed by entering the `raspi-config` command into the Raspbian terminal. The following settings in Table 3 must be adjusted to ensure that the individual devices are optimized for clustering:

Network Configuration
After securing each Raspberry Pi device to the enclosure and developing a system for powering each node using the USB cables, the Raspberry Pi devices must be connected to a central switch for remote management and remote kernel communication. Each of the four devices pictured in Figure 1 have an Ethernet port. The cluster network is constructed by attaching a CAT5e network cable to each network port and attaching the other end of each cable to ports 1-4 of the network switch. A fifth CAT5e cable is inserted into port 5 of the network switch, and the opposite end of that cable is inserted into a network router that facilitates the communication between all connected devices and the Internet.

Passwordless SSH Configuration
In order for Mathematica to issue commands across all four Raspberry Pi devices, SSH must be utilized. This requires the sharing of public SSH keys between all devices (Raspberry Pi Foundation, n.d.a). Public key sharing is accomplished by following these steps:

1. Create .ssh directory on each Raspberry Pi device
2. Generate SSH keys on each Raspberry Pi device
3. Copy the public key from each device to every other device's .ssh directory

All devices will be able to login to each other using SSH without entering a password due to the public key sharing. Commands from the Raspbian terminal and the Wolfram terminal (or Mathematica GUI) can be submitted without entering any passwords.

Mathematica Remote Kernel Configuration
Mathematica processes evaluations using a local kernel, a notebook, and an evaluation (Wolfram Research, 2017c). The local kernel represents the processor(s) present on the local machine. A notebook is used as a value processing interface for Mathematica evaluations, which are strings processed by the kernel. The purpose of building the cluster was to simulate a supercomputer, which relies on the computation of variables in tandem for increased efficiency through load balancing, otherwise known as parallel processing. Mathematica allows for the configuration of remote kernels, which can be used to perform parallel evaluations. Remote kernels must be able to communicate within a common network. Parallel evaluations take a single evaluation and delegate the process of evaluating the string to all processing units (or remote kernels) indicated in the evaluation. Splitting the processing task among multiple processors, or kernels, increases the efficiency and speed of the evaluation. The process for configuring remote kernels in Raspbian distribution of Mathematica is as follows:

Step	Command/Process
1	From a Mathematica notebook: `FrontEndTokenExecute ["PreferencesDialog"]`
2	Select 'Parallel' tab
3	Select 'Remote Kernels' under 'Parallel Kernel Configuration'
4	Select 'Add Host'
5	`Hostname`: <ip address of RPi>
6	`LaunchRemote`: default commands
7	Change 'Kernels' value to `4`
8	Select 'Enable'

Table 4. Remote Kernel Configuration Process

After completing these steps, Mathematica can then be directed to start the remote kernels and issue evaluation tasks using specific Parallel commands.

Mathematica Parallel Computing
Once the remote kernels are configured, evaluating commands in parallel are relatively simple through the use of the `ParallelCombine` command in the Wolfram Language (Wolfram Research, 2017e). This command evaluates a normal expression by distributing parts of the computation to every available remote kernel. The kernels process the computation in tandem and return the results as a single solution. The `ParallelCombine` command is added after the `AbsoluteTiming` command. The following functions were evaluated on the Local Kernel to establish a control by which the parallel evaluations could be compared:

Extract prime numbers from data set of 1 to 1000 (Local Prime)
```
AbsoluteTiming[Prime[Range[1000]]]
```

Extract prime numbers from data set of 1 to 1000 (Parallel Prime)
```
AbsoluteTiming[ParallelCombine[Prime[Rang
e[1000]]]]
```

4. RESULTS

Each evaluation listed above was performed 10 times. Every time an evaluation was performed, the Raspberry Pi kernels were completely reset by closing the Mathematica application and restarting it. This was done to clear the internal system caches of stored results. Mathematica contains a feature that indexes evaluations temporarily so that repeated evaluations can be resolved faster (Wolfram Research, 2017e). However, this feature skewed the data originally collected for the Raspberry Pi cluster. While the first evaluation was a true test of processing speed, the subsequent evaluations all yielded equally fast resolution times, which were faster than the original evaluation. One proposed solution was to use the `ClearSystemCache[]` command to wipe the evaluations indexed in Mathematica, but this did not appear to serve its purpose (Wolfram Research, 2017a). Therefore the most efficient, though undoubtedly rudimentary, technique was to close Mathematica and restart it, forcing all kernels to close their connections and restart along with the graphical interface. The data collected while using this technique are located in Table 5.

Local Prime	Parallel Prime
29.48	8.85
29.31	8.83
29.7	8.79
29.65	8.79
28.98	8.79
29.28	8.87
29.17	9.05
29.51	8.94
29.2	9.02
29.09	8.86

Table 5. Local and Parallel Evaluations

From these data sets, it can be concluded that parallel kernel evaluations are, on average, completed 70.42% faster than local kernel evaluations according to the `AbsoluteTiming` function of Mathematica.

In addition, ten parallel evaluations were performed on the Raspberry Pi cluster when the processors were overclocked (OC) and when the Memory Split function (MS) was adjusted to allocate more memory to graphical processing. The average evaluations are located in Table 6.

From these data sets, it was discovered that overclocking does increase processing speed, but only by 12.53%. Adjusting the Memory Split setting from a 16 MB allocation to 256 MB yielded a decrease in processing speed of 3.53%. However, it's interesting to note that Memory Split did not affect processing speed dramatically. Allocating additional memory to the GPU via the Memory Split settings will decrease the graphical processing overhead generated by Mathematica's interface. This may lead to a more optimized system if the CPU efficiency continues to be affected by a factor of less than 5% (Raspberry Pi Foundation, n.d.c).

Evaluation	Memory Split/ Overclock Setting	Time
Local	16/None	29.69
Parallel	16/None	8.78
Parallel w/OC	16/High	7.68
Parallel w/MS	256/None	9.09

Table 6.

5. DISCUSSION AND CONCLUSIONS

A key limitation to the development of the Raspberry Pi cluster was the gap in documentation between desktop versions of Mathematica and the Raspberry Pi distribution. The Mathematica user interface allowed for all of the function creation and evaluation features required to test the cluster, but both formal and informal sources (e.g. forums) lacked a standard direction for configuring remote kernels on Raspberry Pi devices. The Preferences Dialogue Menu, pictured in Figure 2 in Appendix A, is not accessible unless the user evaluates the following command in Mathematica: `FrontEndTokenExecute["PreferencesDialog"]`

Prior to discovering this command, the process of establishing remote kernel connections in Mathematica consisted of canvassing outdated Mathematica and Raspberry Pi forums for lengthy functions that attempted to activate remote kernels using commands in both the Wolfram CLI and Mathematica. One example of a remote kernel activation function that managed to generate an error message, an optimistic sign at the time, can be referenced in Appendix A, Figure 3 (Quantum, 2016).

A perpetual error message appeared following every attempt to evaluate the remote kernels afterwards:

```
"The kernel failed to connect to the front end. (Error = MLECONNECT). You should try running the kernel connection outside front end."
```

It was only by piecing together advice from various forums related to the Desktop version of Mathematica that the Preferences dialog menu became accessible.

Ultimately, this proof-of-concept was effective in building a working Raspberry Pi cluster to demonstrate the differences in processing times between a single node and multiple nodes. Effectively, the cluster could save an individual over $1,100 in software licensing costs.

There is a trait valued universally across various industries: the ability to do more with less. At the very least, this project demonstrates to the user that parallel processing can be used to leverage subpar resources to create something more valuable. This idea, and the proliferation of its practice, is vastly beneficial to the modern student. In a world where throwing something away is more commonplace than working to fix or improve it, the act of coupling processors together is a prime example of resourcefulness and ingenuity. Outside of the technical enhancements students may reap from this model of Mathematica processing, students may also find themselves extending these practices to their studies and eventual jobs. It’s not difficult to write a business case for why spending should be allocated to a new resource, but it is uniquely valuable to have an employee that understands how to leverage depreciated resources to match current demand. That is one of the primary values behind this Raspberry Pi model.

An unintended consequence of this research project is the value this cluster has in I.S. education to teach and demonstrate how cluster computing can be applied. Previous work by Doucet and Zhang outline the benefits of learning cluster computing through the use of Raspberry Pis. Learning outcomes include a hands-on experiential learning opportunity along with learning a better understanding of how cluster computing could be used to solve computational problems (2017). Their research, along with this work, could be used to build a series of labs for student learning in a networking or I.T. infrastructure course to further connect theory to practice.

It is conceivable that the configuration of a Raspberry Pi/Mathematica cluster could be more automated to lower the overhead for institutions and students alike. Alternatively, clusters could be managed by the institution or IS/IT students for remote access by users. The technology allows for the use of Mathematica at a lower cost than purchasing a standalone license, which can ease the financial burden from departments or students. More research is needed to test other features of Mathematica including advanced algorithms, visualizations, and data analysis. In addition, benchmark tests need ran comparing processing times between the cluster and various models of typical workstations. Ideally, the next study would place the Raspberry Pi cluster in the hands of instructors and students to mimic applicable workloads.

6. REFERENCES

Andrews, C. (2013). Easy as Pi. *Engineering and Technology*, 8(3), 34-37. doi:10.1049/et.2013.0302

Burd, S. (2015). Systems Architecture 7e. Boston, MA: Cengage Learning

Doucet, K., Zhang, J. (2017). Learning Cluster Computing by Creating a Raspberry Pi Cluster. *Proceedings of the SouthEast Conference on - ACM SE 17,* 191-194. doi:10.1145/3077286.3077324

Edwards, C. (2013). Not-so-humble Raspberry Pi Gets Big Ideas. *Engineering and Technology*, 8(3), 30-33. doi:10.1049/et.2013.0301

Falconer, J., Nicodemus, G. (2014). Interactive Mathematica Simulations. *Chemical Engineering Education*, 48(3). Pp. 165-174

Fung, C., Li, J., & Myers, D. (2005). Evaluation of a Small Scale Cluster Computing System for Parallel Intelligent Technique Applications. *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics*, 388-393.

Gropp, W., Lusk, E., & Sterling, L. (2003). *Beowulf cluster computing with Linux.* Cambridge, MA: MIT Press.

Harris, S. (2015). Faster Raspberry Pi brings low-price computing power to education. *Engineer*. (Online Edition), 1.

Heeks, R., Robinson, A. (2013). Emerging Markets Ultra-Low-Cost Computing and Developing Countries. *Communications of the ACM.* 56(8), pp. 22-24.

Hutem, A., Kerdmee, S. (2013). Physics Learning Achievement Study: Projectile, using Mathematica program of Faculty of Science and Technology Phetchabun Rajabhat

University Students. *European Journal of Physics Education*, 4(3), pp. 22-33.

Naughton, J. (2012, March 31). Why all our kids should be taught how to code. *The Guardian*.

Quantum, The Physicist. *Remote Kernel through SSH*. (2016, February 3). Retrieved from https://mathematica.stackexchange.com/questions/65953/remote-kernel-through-ssh

Rahmawati, N.D., Nugroho, A.A., & Harun, L. (2017). *Proceedings of the International Conference on Mathematics: Education, Theory, and Application*. 1(1), pp. 157-164.

Raspberry Pi Foundation (n.d.a). *Passwordless SSH Access.* Retrieved from https://www.raspberrypi.org/documentation/remote-access/ssh/passwordless.md

Raspberry Pi Foundation (n.d.b). *Raspberry Pi 3 Model B*. Retrieved from https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

Raspberry Pi Foundation (n.d.c). *Raspi-Config*. Retrieved from https://www.raspberrypi.org/documentation/configuration/raspi-config.md

Raspberry Pi Foundation (n.d.d). *SD Card*. Retrieved from https://www.raspberrypi.org/learning/hardware-guide/components/noobs-card/

Schuurman, D. (2015). Introducing Open Source and the Raspberry Pi to Schools in Developing Nations. *Perspectives on Science and Christian Faith*. 6(1). pp. 50-53.

Trustees of the California State University (2017). *What is Mathematica?* Retrieved from http://www.calstatela.edu/its/services/software/mathematica.php

Wolfram Research (2017a). *ClearSystemCache*. Retrieved from http://reference.wolfram.com/language/ref/ClearSystemCache.html.en

Wolfram Research (2017b). *Company Information.* Retrieved from https://www.wolfram.com/company/

Wolfram Research (2017c). *Documentation Center*. Retrieved from http://reference.wolfram.com/language/

Wolfram Research (2017d). *Mathematica Pricing*. Retrieved from https://www.wolfram.com/mathematica/pricing/colleges-universities-individuals.php

Wolfram Research (2017e). *ParallelCombine*. Retrieved from http://reference.wolfram.com/language/ref/ParallelCombine.html

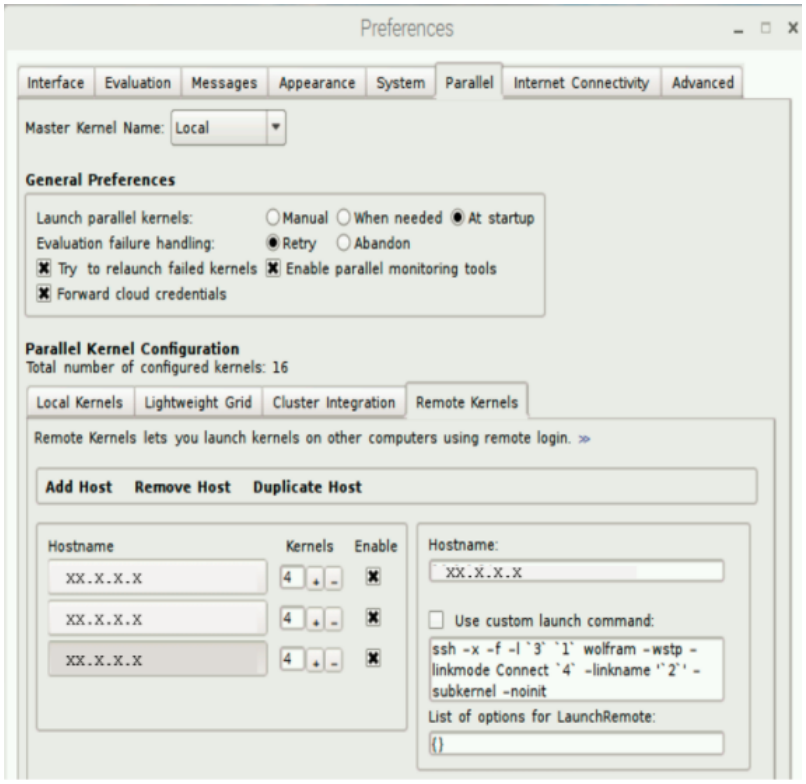
Wolfram, Stephen. (2013). *Putting the Wolfram Language (and Mathematica) on Every Raspberry Pi*. Retrieved from http://blog.stephenwolfram.com/2013/11/putting-the-wolfram-language-and-mathematica-on-every-raspberry-pi/

U.S. Department of Education, National Center for Education Statistics. (2016). Digest of Education Statistics, 2015 (NCES 2016-014),Chapter 3. https://nces.ed.gov/fastfacts/display.asp?id=76

Appendix A

Resource/Tool		**Function**
7-Zip	http://www.7-zip.org/download.html	A file archiver used to unzip Raspbian image file for distribution to microSD cards.
PuTTY	https://www.putty.org	A SSH and Telnet client that allows the user to issue commands simultaneously to Raspberry Pi devices from a Windows workstation.
Raspbian image	https://www.raspberrypi.org/ downloads/raspbian/	Operating system for Raspberry Pi
SDFormatter	https://www.sdcard.org/downloads/ formatter_4/eula_windows/index.html	Tool used to format the microSD card used by the Raspberry Pi for secondary storage of Raspbian operating system and all other system files
Win32 Disk Imager	https://sourceforge.net/projects/ win32diskimager/	Allows the user to read and write Raspbian images to and from microSD cards for uniform image provisioning throughout cluster

Table 7. Raspberry Pi Cluster Software Tools



Master Kernel Name: Local

General Preferences

Launch parallel kernels: Manual When needed [✓ At startup]
Evaluation failure handling: [✓ Retry] Abandon
[✓ Try to relaunch failed kernels] [✓ Enable parallel monitoring tools]
[✓ Forward cloud credentials]

Parallel Kernel Configuration
Total number of configured kernels: 16

Remote Kernels lets you launch kernels on other computers using remote login. »

Hostname	Kernels	Enable
XX.X.X.X	4	✓
XX.X.X.X	4	✓
XX.X.X.X	4	✓

Hostname: XX.X.X.X

Use custom launch command:

```
ssh -x -f -l `3` `1` wolfram -wstp -
linkmode Connect `4` -linkname '`2`' -
subkernel -noinit
```

List of options for LaunchRemote: {}

Figure 2. Mathematica Preferences Dialog Menu

```
Needs["SubKernels`RemoteKernels`"]
Parallel`Settings`$MathLinkTimeout = 100;
       user = "pi";
password = "RPi02*";
ssh = "export LD_LIBRARY_PATH=;ssh";
math = "MathKernel" <> " -wstp -linkmode Connect `4` linkname `2` -subkernel -noinit
>& \ /dev/null &";
number = 4;
machine = "XX.X.X.X";
remote = SubKernels`RemoteKernels`RemoteMachine[machine, ssh <> " " <> user <> "@" <>
machine <> " \"" <> math <> "\"", number]

Print[remote // InputForm]
kerns = LaunchKernels[remote]

ParallelEvaluate[$MachineName]
(*CloseKernels[]*)
```

Figure 3. Sample Function to Activate Remote Kernels

Infrastructure Tools for Efficient Cybersecurity Exercises

Jim Marquardson
jimarqua@nmu.edu
College of Business
Northern Michigan University
Marquette, MI 49855, USA

Abstract

Academics are responding to the call from industry for graduates armed with cybersecurity skills. A common challenge that educators face is creating effective cybersecurity curriculum that prepares students with practical skills upon graduation. While hands-on exercises are a powerful method for teaching and assessing cybersecurity skills, these exercises can be difficult to create, require large infrastructure investment, or waste valuable learning time simply configuring the learning environment. In recent years, industry has demanded increased infrastructure automation. These tools have matured and help make provisioning and configuring hardware and software easier. Infrastructure automation tools can help cybersecurity educators create exercises that can scale without adding substantial burden on the educators.

Keywords: Cybersecurity, Infrastructure automation, Curriculum design and development, Computer Security

1. INTRODUCTION

Malicious actors know how to find and exploit weaknesses in systems. They not only know the definitions of key cybersecurity concepts, but they know how to operationalize those concepts. To effectively defend against malicious actors, it is critical that cybersecurity students can apply the skills they learn in the classroom. The National Security Agency / Department of Homeland Security Centers of Academic Excellence in Cyber Defense program defines knowledge units with skills that cybersecurity students should obtain in their degree programs (NSA / DHS, 2013). It is important to note that the knowledge units often list outcomes where students should be able to use, apply, operate, configure, install, and analyze key cybersecurity technologies. These active verbs reinforce the need for cybersecurity students to have practical, hands-on skills.

There are many challenges when creating technical exercises to teach and assess cybersecurity skills. Clearly, some infrastructure is required. Students must have computers with administrative privileges so they can install software and make configuration changes. Often, dedicated cybersecurity labs with networking equipment and servers are needed. Lab networks may need to be segmented from the general campus network. Capital and operational costs musts be considered when designing the infrastructure to support a cybersecurity program. While private labs provide an excellent space for conducting cybersecurity exercises, maintaining infrastructure and configuring systems can be a major burden. For example, giving students an exercise where they configure firewalls on a server might require one new server be created for each student in the class. If done manually, setting up a single lab exercise can be extremely taxing on educators.

Information technology has long aimed to increase operational efficiency in organizations. The increasing adoption of cloud computing has spurred developments in infrastructure automation. Cloud computing is sold with the promise of quick and easy provisioning and deprovisioning of computing resources to scale with demand. To reduce the cost of provisioning resources, cloud providers have leveraged existing tools and built tools to automate infrastructure. Educational institutions can leverage these same infrastructure automation tools within their private infrastructures.

To date, few educators have published clear, actionable strategies for creating cybersecurity exercises that meet student needs without excessive burden on the educator to maintain infrastructure. The following sections in this paper describe some of the most popular tools for virtualization and infrastructure automation that educators can use to develop cybersecurity exercises. Practical recommendations are given to help educators known when adoption of the tool can be beneficial. The purpose of these overviews is to introduce the tool, the benefits the tools provide, and a high-level overview of key concepts.

2. VIRTUALIZATION

Virtualization technologies allow one or more guest virtual machines to run on a physical host. Virtualization can be broken down into two main categories: client-side virtualization and server-side virtualization. The advantages and potential drawbacks of adopting each solution in an educational environment will be discussed in the following sections.

Client-side Virtualization

With client-side virtualization, students run one or more guest operating systems on their computer in a similar way to how they run applications. Changes made inside the virtual machine do not affect the host operating system. A student running Microsoft Windows as her primary desktop operating system could run a Windows Server virtual machine and a Linux virtual machine to observe the interaction of those two systems. The top virtualization platforms for desktop systems are VMWare Workstation Player (*VMWare Workstation Player*, 2017) and Oracle's VirtualBox (*VirtualBox*, 2017).

VMWare Workstation Player is proprietary software. It should be noted that VMWare has changed its licensing model and product offering several times over the past several years. Currently, VMWare Workstation must be licensed for use in the classroom. It is impossible to know if VMWare will continue to support the product or change its licensing structure. VMWare Player has proven to work well for desktop virtualization, but building on top of a closed platform can be problematic.

VirtualBox is an open source desktop virtualization platform maintained by Oracle. VirtualBox is free and supports a wide variety of guest operating systems such as Windows desktop editions, Windows server editions, Linux, BSD, and Solaris. Because VirtualBox is open source, the community could feasibly support the product if Oracle stopped updating the platform. Because of these reasons, VirtualBox became the platform of choice when developing our cybersecurity curriculum.

There are several key advantages of using client-side virtualization for cybersecurity exercises. First, students can run cybersecurity exercises completely on their personal computers. Students would not need access to a dedicated campus computer lab, thereby reducing the infrastructure investment required and increasing accessibility. Another benefit is the ability to segment cybersecurity traffic from the rest of the network. It is possible to run client-side virtualization without any network connectivity, making it a good choice for students with network connectivity challenges.

Perhaps one of the key benefits of running cybersecurity exercises through virtual machines rather than using students' host operating systems is the ability to segment network traffic. A virtual machine can be configured so that network traffic never leaves the client machine. This segmentation prevents students from accidentally performing malicious actions on the network. Two key networking modes available in VirtualBox will be explained. With network address translation mode enabled, a virtual machine can connect to the internet, but the virtual machine cannot interact with other virtual machines running on the same guest. With internal networking, virtual machines on the same guest can communicate with each other but cannot connect to the internet. Students may need to switch network modes during exercises. For example, a student might use the network address translation mode to connect to the internet and download software packages, then switch to the internal networking mode to communicate with other virtual machines and prevent traffic from reaching other networks accidentally.

Some drawbacks exist in client-side virtualization that prevent it from being the definitive solution in cybersecurity exercise development. First, student computers may be limited by hardware capabilities. A single cybersecurity exercise might require several virtual servers to be running simultaneously. The exercises that can be conducted may be constrained by RAM, CPU, or hard disk. Modern Windows Server operating systems require at least 2GB RAM to run reasonably well, so running three virtual servers may be infeasible on older hardware. Each virtual server can take between 1-10GB on average. Computers with smaller solid-state disks typically perform well but often sacrifice capacity. Some hardware may simply not have enough capacity to install multiple operating systems. Second, supporting a wide number of devices can be challenging. Though modern virtualization software can be installed on Windows, MacOS, and Linux, differences between client configurations can lead to time spent troubleshooting virtualization software. For example, some laptops have virtualization features disabled in the BIOS by default which can be corrected, but can cause confusion. Also, anti-virus has interfered with the successful installation of virtualization software leading to problems completing exercises.

Server-side Virtualization
Microsoft's Hyper-V (*Hyper-V*, 2017) and VMWare vSphere (*vSphere*, 2017) are two popular server virtualization platforms used in the data center. At a high level, both platforms run virtual servers on top of physical hardware. Unlike client-side virtualization platforms that run virtual machines on top of a complete operating system layer, modern server-side platforms run virtual machines on a thin hypervisor layer which gives virtual machines more direct access to hardware resulting in better performance.

The VMWare vSphere Hypervisor is a free product with limited functionality that is installed directly on server hardware. Administrators install the vSphere Client on their computers and connect to the server to manage virtual machines. While vSphere Hypervisor is a low-cost option, key productivity features are missing (such as the ability to clone an existing server). Like VMWare vSphere Hypervisor, Microsoft Hyper-V Server 2016 is a free, thin virtualization layer that sits on top of the physical hardware. Hyper-V can also be enabled on a modern Windows Server by enabling the Hyper-V role.

Both vSphere and Hyper-V allow administrators to over-commit resources to accommodate more virtual hardware. For example, a physical server might have 32GB RAM. On that physical server, 32 virtual servers can be created and assigned 2GB RAM each. Because servers rarely use all available RAM, each virtual server should run fine despite the overallocation. Using either platform, licenses for Windows Server guest operating systems must be obtained as usual.

The VMWare and Microsoft platforms are both mature and good candidates for deploying virtual servers in a private cybersecurity lab. Choosing one platform over the other will likely be driven by vendor preference, licensing costs, or compliance with established information technology standards.

Major benefits of server-side virtualization include centralized control, scalability, and policy enforcement. Because administrators have complete control over the infrastructure, exercises can be designed and tested in a stable environment. Administrators can control all aspects of the environment from operating system versions, firewalls rules, and software installed. Students are less likely to have to spend time troubleshooting extraneous issues in a tightly controlled environment. Next, leveraging the same tools as large cloud providers, infrastructure can be built to scale computing capacity to meet student needs so that issues of students' computers lacking sufficient resources are negated. Lastly, because the infrastructure is centrally managed, policy regarding network traffic and acceptable use can be carefully monitored.

A clear drawback of server-side virtualization is initial investment. Hardware must be purchased, configured, and actively managed throughout its lifecycle. Dedicated lab administrators may need to be hired to manage the computing environment. Lastly, increased control comes at the cost of increased administrative burden. With client-side virtualization, students typically manage their own infrastructure. With server-side virtualization, the educator is responsible. Server-side virtualization can mean more time creating and configuring the infrastructure.

3. INFRASTRUCTURE AUTOMATION

Virtual machines frequently need specific software and configurations. Manual installation and configuration can be time consuming and error prone. Fortunately, infrastructure automation tools exist to streamline the process of creating virtual machines, installing software, and making configuration changes. The following sections describes some of the top tools that have emerged that can help educators be more effective.

Vagrant
Vagrant is a tool that makes provisioning and deprovisioning virtual machines efficient (*Vagrant*, 2017). Though Vagrant can be used to provision virtual machines on several virtualization platforms, emphasis will be given here on its integration with VirtualBox. A core concept in vagrant is a Vagrant box—a partially configured virtual machine used in lieu of an operating system DVD image for the creation of a virtual machine. The typical workflow for creating a virtual machine using VirtualBox without Vagrant involves downloading an ISO, creating a blank virtual machine, attaching the ISO, botting the virtual machine, following the installation prompt, and waiting for the operating system to be installed. The entire process can take dozens of clicks and 30 minutes of waiting.

Using Vagrant, a virtual machine can be created with just two commands: vagrant init [version]; vagrant up. Table 1 shows the commands needed to create a virtual machine in VirtualBox using Vagrant and connecting through SSH. The entire process takes approximately 4 minutes to complete.

```
C:\temp> vagrant init ubuntu/xenial64
C:\temp> vagrant up
C:\temp> vagrant ssh
```

Table 1: Creating an Ubuntu Server Virtual Machine with Vagrant

Creating virtual machines with Vagrant requires fewer steps, fewer decisions, and completes in less time. With Vagrant, the cost of breaking a virtual machine accidentally or intentionally is low because they can be recreated easily.

In addition to creating generic servers, instructors can create Vagrant configuration files that carry out post-installation configurations automatically. A Vagrantfile is a Ruby file that tells Vagrant which virtual machines to create along with basic configuration settings. A single cybersecurity exercise, such as address resolution protocol spoofing, might require three virtual machines. Table 2 shows a sample Vagrantfile that defines three virtual machines and assigns them private IP addresses. The Vagrantfile could be distributed to students via a learning management system. Then, students would copy the file to their hard drive, navigate to the folder in the command prompt, then run the command "vagrant up." The three virtual machines would then begin the boot process without any additional input required by the students.

A Vagrantfile that installs the Moodle learning management system on a single virtual server as part of the provisioning process is shown in Appendix A. Vagrant can leverage configuration management tools such as Ansible and Chef (described in the subsequent sections), but much can be done using shell scripting. The advantage of shell scripting is that special configuration management software is unnecessary. The downside to shell scripting is the potential time requirement needed to write and troubleshoot custom scripts.

The main benefit of Vagrant is the ease with which virtual machines can be created. The drawbacks include the learning curve to create Vagrantfiles. Vagrant also introduces another tool that must be updated. Both the Vagrant software and the boxes must periodically be updated to fix bugs and obtain the latest patches from operating system vendors. The learning curve for Vagrant is low for basic usage. Advanced Vagrant features can be introduced over time.

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure(2) do |config|

config.vm.provision "shell", inline: "echo
Starting victim, middle, and server"

config.vm.define "victim" do |victim|
  victim.vm.box = "ubuntu/trusty64"
  victim.vm.host_name = "victim"
  victim.vm.network "private_network",
     ip: "192.168.10.10"
end

config.vm.define "middle" do |middle|
  middle.vm.box = "ubuntu/trusty64"
  middle.vm.host_name = "middle"
  middle.vm.network "private_network",
    ip: "192.168.10.50"
end

config.vm.define "server" do |server|
  server.vm.box = "ubuntu/trusty64"
  server.vm.host_name = "server"
  server.vm.network "private_network",
     ip: "192.168.10.100"
end

end
```

Table 2: Vagrantfile Defining Three Servers

Ansible
Ansible is RedHat's infrastructure automation tool that aims to automate simple and complex infrastructures (*Ansible*, 2017). Administrators use the Ansible language to define playbooks. Playbooks describe system configurations that should be applied to target systems by the Ansible automation engine. A paid add-on, the Ansible Tower can monitor and apply playbooks on a large infrastructure. Ansible does not require custom agents to run on the server; instead it applies all changes over SSH.

The playbook in Table 3 demonstrates the human readable format of the Ansible language. An administrator with some Linux experience can understand that the playbook ensures that the Apache web server and PostgreSQL database servers are running. The example comes from the Ansible online documentation ("Intro to Playbooks," 2017).

```
---
- hosts: webservers
  remote_user: root

  tasks:
  - name: ensure apache is at the
latest version
    yum: name=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2
dest=/etc/httpd.conf

- hosts: databases
  remote_user: root

  tasks:
  - name: ensure postgresql is at the
latest version
    yum: name=postgresql state=latest
  - name: ensure that postgresql is
started
    service: name=postgresql
state=started
```

Table 3: Sample Ansible Playbook

Ansible must be installed on a Linux control node—this is the only machine that needs Ansible installed. Unfortunately, the Linux requirement means that students running Windows or MacOS operating systems cannot use Ansible to create virtual machines on their own computers. For this reason, Ansible is most applicable for private labs where the instructor wants full control over the infrastructure.

An example will help illustrate the value of Ansible. Suppose an instructor wants to configure 50 virtual servers to ensure that they have nmap installed for a port scanning exercise. First, the instructor must create the Ansible playbook. Next, the instructor would run the following command to apply that playbook to all hosts defined in the playbook: "ansible-playbook cyber-exercise-1.yml." Ansible would login to each virtual server and install nmap if needed.

Key benefits of Ansible include a low learning curve compared to other infrastructure management tools, lack of an agent required on the remote servers, and scalability. However, the introduction of any configuration management tool requires that the administrator be trained in its use. And as mentioned previously, the Ansible controller must run Linux, though the controller does not have to be a dedicated server.

Chef
Chef is one of the first widely used infrastructure automation platforms (*Chef*, 2017). Chef is open source software created by Chef Software, Inc. Like Ansible, Chef requires a master controller to communicate with nodes. One difference is that Chef's master must be a dedicated server, whereas the Ansible client can be run from any Linux computer. Also, whereas Ansible performs configuration changes on nodes using SSH, Chef communicates to custom Chef agents on the nodes.

Chef's documentation and tutorials are extensive. However, the learning curve for using Chef is steeper than Ansible. For example, the introduction tutorial for infrastructure automation is estimated to take eight hours. The new Chef user can quickly become overwhelmed with cooking related tools—recipes, knives, supermarkets, kitchens, etc. But the maturity of the platform and extensive features make it a robust solution for complex infrastructure needs.

Chef starts with a cookbook. Cookbooks are written in the Ruby programming language—a dynamic language with similar readability to Python. A cookbook contains one or more recipes, files, libraries, attributes, and additional environment information. Recipes instruct chef how to configure the system. Cookbooks are registered on the chef server and pushed out to chef nodes.

To create a cybersecurity exercise using Chef, an instructor would first need to create a cookbook. The Chef Development Kit comes with command line tools to create a skeleton cookbook. Recipes must be added to the cookbook. Table 4 shows a sample recipe to install the Apache web server.

This recipe would be saved in a Ruby .rb file within the cookbook folder.

```
package "apache2" do
  action :install
end
```

Table 4: Sample Chef Recipe

When the recipe is complete, the knife tool is used to upload the cookbook to the chef server. Then, the knife command can be used to run the recipe on nodes.

Because Chef an Ansible are similar tools, their benefits can be directly compared. Chef’s advantages include extensive documentation, and active community, and a mature, tested solution. Compared to Ansible, Chef has a steeper learning curve, is more complex, and uses more computing resources. Chef is more likely than Ansible to force the educator to spend time managing the configuration management.

4. CONCLUSIONS

Advances in virtualization and infrastructure automation tools make it easier than ever to develop and deploy cybersecurity exercises. No single tool can solve all infrastructure and configuration management challenges. For the organizations that lack dedicated lab environments, the Vagrant and VirtualBox combination is a mature, flexible solution for creating virtual environments on the fly quickly. For organizations with dedicated lab environments, Ansible, Chef, and server-side virtualization platforms are solutions that should be explored. Ansible is a lighter weight solution for organizations that want to take their first steps using configuration management. Chef would be an appropriate choice for more complex configuration scenarios, though the steeper learning curve than Ansible should be taken under consideration.

The technologies shared in this paper have been evaluated for fit in a cybersecurity program. It is hoped that educators continue to share cybersecurity exercise best practices so that the discipline can move forward quickly to meet the increasing need for qualified professionals.

5. REFERENCES

Ansible. (2017). Retrieved from https://www.ansible.com/

Chef. (2017). Retrieved from https://www.chef.io

Hyper-V. (2017). Retrieved from https://www.microsoft.com/en-us/cloud-platform/server-virtualization

Intro to Playbooks. (2017). Retrieved June 14, 2017, from http://docs.ansible.com/ansible/playbooks_intro.html#playbook-language-example

NSA / DHS. (2013). *NSA / DHS National Centers of Academic Excellence in Cyber Defense (CD) Knowledge Units* (pp. 1–73). Retrieved from https://www.iad.gov/NIETP/documents/Requirements/CAE-CD_Knowledge_Units.pdf

Vagrant. (2017). Retrieved from https://www.vagrantup.com/

VirtualBox. (2017). Retrieved from https://www.virtualbox.org/

VMWare Workstation Player. (2017). Retrieved from http://www.vmware.com/products/player/playerpro-evaluation.html

vSphere. (2017). Retrieved from https://www.vmware.com/products/vsphere.html

Appendices and Annexures

Appendix A – Additional Tables and Figures

Vagrantfile that Installs Moodle
The following code can be used to install the Moodle learning management system as part of the virtual machine provisioning process. The Vagrantfile could be distributed to students via a course website. Student would download the Vagrantfile to their computers, open a command prompt, navigate to the folder with the Vagrant file and run "vagrant up." Vagrant would then create a new virtual machine, download required packages, and complete the Moodle installation in the background. When the process completes, students could open a web browser and view the Moodle installation at http://localhost:8888.

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/xenial64"
  config.vm.network "public_network", ip: "192.168.1.99"
  config.vm.synced_folder "./", "/vagrant_share"

  #Provisioning instructions leveraged from:
  # https://docs.moodle.org/31/en/Step-by-step_Installation_Guide_for_Ubuntu
config.vm.provision "shell", inline: <<-SHELL
sudo su
echo "nameserver 8.8.8.8" > /etc/resolv.conf #Fix DNS resolving "bug" in Xenail
apt-get update
debconf-set-selections <<< 'mysql-server mysql-server/root_password password mysqladmin'
debconf-set-selections <<< 'mysql-server mysql-server/root_password_again password mysqladmin'
apt-get -y install mysql-server
apt-get -y install apache2 mysql-client php7.0 libapache2-mod-php7.0
apt-get -y install graphviz aspell php7.0-pspell php7.0-curl php7.0-gd php7.0-intl php7.0-mysql
php7.0-xml php7.0-xmlrpc php7.0-ldap php7.0-zip
echo "Installing php-mbstring and php-soap (optional Moodle components)"
apt-get -y install php-mbstring php-soap
service apache2 restart
apt-get -y install git-core
cd /opt
git clone git://git.moodle.org/moodle.git
cd /opt/moodle
git branch --track MOODLE_32_STABLE origin/MOODLE_32_STABLE
git checkout MOODLE_32_STABLE
cp -R /opt/moodle /var/www/html/
mkdir /var/moodledata
chown -R www-data /var/moodledata
chmod -R 777 /var/moodledata
chmod -R 0755 /var/www/html/moodle

echo "default_storage_engine = innodb" >> /etc/mysql/mysql.conf.d/mysqld.cnf
echo "innodb_file_per_table = 1" >> /etc/mysql/mysql.conf.d/mysqld.cnf
echo "innodb_file_format = Barracuda" >> /etc/mysql/mysql.conf.d/mysqld.cnf
service mysql restart
mysql -u root --password=mysqladmin -e "CREATE DATABASE moodle DEFAULT CHARACTER SET utf8
COLLATE utf8_unicode_ci;"
mysql -u root --password=mysqladmin -e "create user 'moodleadmin'@'localhost' IDENTIFIED BY
'moodleadmin';"
```

```
mysql -u root --password=mysqladmin -e "GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,CREATE
TEMPORARY TABLES,DROP,INDEX,ALTER ON moodle.* TO moodleadmin@localhost IDENTIFIED BY
'moodleadmin';"

echo "<?php  // Moodle configuration file

unset(\\$CFG);
global \\$CFG;
\\$CFG = new stdClass();

\\$CFG->dbtype    = 'mysqli';
\\$CFG->dblibrary = 'native';
\\$CFG->dbhost    = 'localhost';
\\$CFG->dbname    = 'moodle';
\\$CFG->dbuser    = 'moodleadmin';
\\$CFG->dbpass    = 'moodleadmin';
\\$CFG->prefix    = 'mdl_';
\\$CFG->dboptions = array (
  'dbpersist' => 0,
  'dbport' => '',
  'dbsocket' => '',
);

\\$CFG->wwwroot   = 'http://127.0.0.1:8888';
\\$CFG->dataroot  = '/var/moodledata';
\\$CFG->admin     = 'admin';

\\$CFG->directorypermissions = 0777;

require_once(__DIR__ . '/lib/setup.php');
" >> /var/www/html/moodle/config.php
echo "Changing the Document root"
sed -i "s/DocumentRoot \\\/var\\\/www\\\/html/DocumentRoot \\\/var\\\/www\\\/html\\\/moodle/"
/etc/apache2/sites-available/000-default.conf
service apache2 restart
echo "The installation has completed."
echo "Open a browser on your host and go to http://127.0.0.1:8888 to complete configurations."
SHELL

end
```

Server on a USB Port: A custom environment for teaching systems administration using the Raspberry Pi Zero

Michael Black
mblack@southalabama.edu

Ricky Green
reg1521@jagmail.southalabama.edu

School of Computer and Information Sciences
University of South Alabama
Mobile, AL 36688 USA

Abstract

For students seeking careers not related to the computer sciences, the task of installation, configuration and maintenance of software on business class systems may be an essential skill. Dedicated computers are essential to these kinds of courses, but the constraints of university computer labs prohibit their use. Other solutions exist, but require more time to configure and require more computer skills than are typically found in students seeking these careers. The Raspberry Pi computer, introduced in 2012, has been used with some success for this purpose. However, its portability is impaired by the need for external peripherals to make it work, such as a keyboard, mouse, and display. The Raspberry Pi can utilize a laptop for these peripherals, but this requires some additional effort. The recently released Raspberry Pi Zero eliminates much of that effort.

Keywords: teaching material, Linux, Raspberry Pi, systems administration, server applications

1. INTRODUCTION

Designing computer laboratories for Computer Science related courses brings many challenges. The challenges of designing and implementing computer laboratories are even greater for courses that are not related to Computer Science, yet still require computers configured specifically for the course. For students seeking these careers, the task of installation, configuration and maintenance of software on business class systems may be an essential skill. The amount of computer knowledge required for this type of systems administration is not as extensive as it is for courses in the computer sciences, but it is more extensive than a basic course in general computing.

A specific requirement for these types of courses is the student must possess an account with elevated permissions to the operating system installed on the computer for performing administrative tasks, such as installing and maintaining software packages. Access control at this level is almost impossible due to the risk management practices at the local level (Davison, 2015). Also, computer labs that are designed for teaching advanced computing courses are typically isolated from the campus network and do not allow full Internet access (Conlon & Mullins, 2011). These constraints can only be overcome by using something other than the traditional computer lab.

In 2012, the University of St. Andrews recognized the constraints of the typical university computer laboratory when designing an Open Access Bioinformatics research course for biology students. This research course was designed to prepare biology students with basic systems administration skills and the confidence to discover software solutions and implement them as required on a Linux based server. This discovery and implementation process required a certain amount of system administration training suitable for undergraduate bioinformatics students without the technical skills that are typically introduced to computer science students (Barker et al., 2013).

A solution to the problem must allow elevated permissions to the student, which would necessarily allow the student to render the individual computer completely unusable. These mistakes can be costly and time-consuming to correct in a typical university computer lab environment. It may not be necessary for these students to learn how to repair the damage created by these mistakes, but the recovery time from them can be considerable. The solution must provide students with adequate amounts of time to experiment with the system and complete their assignments (Owen & Black, 2003). Therefore, the ideal solution should not attempt to prevent these events from happening but must implement a mechanism for quickly recovering from these mistakes while requiring little if any additional skills beyond what students would normally experience in a traditional computer lab.

An additional requirement of the solution must include a limit on the amount of class time it takes to prepare the solution for use by the students. The ideal solution would be a platform that is preconfigured and ready to use by all students on the first day of class. In summary, the requirements must allow elevated permissions to the student, provide a quick and easy recovery mechanism, be ready to use on day one, and be easy enough to use for students that are not seeking a degree in computer science.

Approaches to Alternatives
The baseline approach is a computer lab specifically configured for the course. A dedicated computer laboratory may be considered the ideal solution for students and instructors in these types of courses because the hardware and software can be easily isolated for security purposes and can be standardized to eliminate compatibility issues. Also, it is possible to plan a quick recovery mechanism when the hardware and software are standardized in the computer lab. However, each student would be required to use the exact machine for each assignment as they build their skills. Each student would need the ability to continue where they left off, so the state of each machine needs to remain intact throughout the course. These limitations make the dedicated lab solution very inflexible since the computers would not be available for any other purpose. This inflexibility makes it a costly solution to implement since each computer in the laboratory is essentially dedicated to a single student for the duration of the course.

A similar approach is to utilize a laptop for each student. Unlike the dedicated computer lab, this approach is specifically designed to provide a dedicated machine per student throughout the course. This solution is flexible if the laptop is compatible with the requirements of the course and remains serviceable. These laptops can be issued by the University, which may help with consistency in hardware configuration and support, but may be considered redundant when most students already possess their laptop and carry it with them to classes (Kay & Lauricella, 2014). Student-owned laptops are not always standardized, which presents potential compatibility issues with the software to be used in the course. Students will be tasked with installing and configuring the software environment on their laptops for the course, which requires additional time and computer skills that the student may not possess. Also, there is no quick recovery mechanism in the event of a mistake when using student laptops. Utilizing university owned laptops could eliminate many of these constraints, but this approach is at least as expensive to implement and maintain as a traditional computer lab while maintaining the same inflexibility.

An iterative approach is to provide students with standard virtual machine images using virtualization software such as VirtualBox or VMWare Workstation. These virtual machine images can be run on any computer, from lab computers to student laptops. Since the hardware is identical on each virtual machine, and the virtualization software is available for all major operating systems, this approach can help eliminate the inconsistent configurations and compatibility issues including the inconsistent hardware among the student laptops. Many of these virtual machine images can reside on the same hardware and booted on demand, reducing the inflexibility of the dedicated computer lab approach (Murphy & McClelland, 2009). This approach takes additional time and skill to download and import the virtual machine images, especially when using student laptops. Recovery from most mistakes can be quick but requires a few more skills than using a real computer with a locally installed operating system. The virtual machine solution eliminates many problems but still fails in the requirement of being ready to use on the first day and requires a few extra skills to use effectively.

Another approach is to utilize Live CD's or bootable USB sticks that are configured to boot an entire operating system that is independent of the operating system installed on the computer itself. This approach is portable, inexpensive and flexible, but may require configuration changes to the computer to enable it to work. Newer computers and laptops have replaced the standard BIOS with UEFI (Unified Extensible Firmware Interface) and Secure Boot, which was designed to prevent the operating system from being compromised via rootkits (Wilkins & Richardson, 2013). As a result, these computers will not allow a USB stick to boot without making changes to the system itself. These changes require additional skill and time to setup and may render the installed operating system inoperative until the changes are reversed.

A central server approach with a separate login for each user is a flexible approach for most computer laboratories. This central server approach requires a small amount of initial setup for the student to install the client software on their laptop for remote control and may allow them to access the server from home. This approach incurs a high upfront cost to implement the hardware and infrastructure required to support the laboratory environment. It is also unwise to grant elevated permissions to multiple students on a central server, as one mistake by a student can take down the entire host. Because administrator access cannot be allowed, the shared server approach cannot be used for the key understanding of software installation and configuration (Barker et al., 2013). Due to this limitation, the central server approach is not suitable for this type of course.

A hybrid approach consisting of individual virtual machines on a central server could be a flexible solution. Each virtual machine could be configured for each student with the appropriate permissions, yet isolated from the host to eliminate the security risk of multiple users holding elevated permissions. This Virtual Computer Lab solution maintains the high upfront costs associated with the central server approach and may incur additional costs due to the higher hardware demands for executing multiple virtual machines simultaneously (Murphy & McClelland, 2009). This solution also requires additional administration for the host operating system and the supporting infrastructure. However, it does not require student time for setup, specific skills to use, and has a very quick recovery mechanism if the host administrator is available.

An alternative approach that is gaining popularity is the use of an inexpensive computer called the Raspberry Pi, which is the approach that was eventually used by the University of St. Andrews for the Bioinformatics course. In their approach, the University of St. Andrews loaned a Raspberry Pi computer and associated peripherals to students for the duration of the course (Barker et al., 2013). The Raspberry Pi is a very small, inexpensive single board computer that runs the Linux operating system. The peripherals consisted of a keyboard, mouse, power adapter, a preconfigured SD card with the operating system and associated applications, and a USB stick for backups. An LCD could not be issued as a peripheral, but the students could use an LCD from the computer lab during class. A lab kit such as this allows students to take the lab equipment home to complete their homework and experiments on their own time (Reck & Sreenivas, 2016). Even with the device limitations and the costs of the extra peripherals, the Raspberry Pi was considered an effective and low-cost approach (Barker et al., 2013). The Raspberry Pi solution allowed students to have full access to a suitable operating system on dedicated, standardized hardware without the limitations of the traditional approaches.

Reasoning for the Raspberry Pi Approach

The ideal solution is a dedicated computer for each student, with full administrative access so students can build their systems management skills throughout the course. The constraints of traditional and advanced computer labs essentially prevent their use in these non-computer science related courses (Barker et al., 2013). The Raspberry Pi device lowers the entry cost of a dedicated lab to the point where flexibility and cost are no longer a major concern. The Raspberry Pi approach presents a standard platform that meets the requirements for many system administration courses.

Problems with the Raspberry Pi Approach

Although the Raspberry Pi is a model for power efficiency and low cost, it's processing power and memory is limited. At the time the Bioinformatics course at the University of St. Andrews was initiated, the only model of the Raspberry Pi available was the $35 Model B, and the stripped down $25 Model A that was designed for low power embedded applications. These tiny computer boards were designed for power efficiency and low cost at the expense of processing and memory limitations. These early models utilized a single core 700Mhz 32bit ARMv6 processor with 512MB of RAM. The performance of this small computer is far slower than modern desktop and laptop computers, but the processing power was deemed more than adequate for the task of the bioinformatics administration course (Barker et al., 2013). Current models of the Raspberry Pi computer have greatly increased processing capability and memory up to 1GB for the same $35, but are still far behind the performance of standard computers and laptops. For other courses that require more processing power or memory, the Raspberry Pi may not be an adequate solution.

For courses where the processing power of the Raspberry Pi is adequate, the most cumbersome issue with the Raspberry Pi approach is the requirement of additional peripherals to make use of them. These peripherals also increase the both the cost of the solution and the amount of equipment the students must carry with them if they are to use them outside of the classroom.

Overcoming Problems with the Raspberry Pi Approach
External peripherals do not need to be large and cumbersome. Portable external accessories are available that can lighten the load of the student. Portable wireless keyboards with built-in touch pad pointing device, such as the $40 Logitech K400, are much easier to stow in a backpack than typical keyboards and reduces two peripherals to one. For the display, the $70 official Raspberry Pi 7” Touchscreen accessory utilizes the dedicated display connector on the Raspberry PI, providing 800x480 display resolution. When mounted in an appropriate case ($20-$30), the Raspberry Pi will essentially become a portable all-in-one touch enabled computer for under $200. However, at this price point, the cost of the system is in the territory of an inexpensive laptop that can provide much better performance without requiring extra time and skill to assemble all the pieces.

A potential solution to the problem with external peripherals is to use the Raspberry Pi without them. The Raspberry Pi can be configured to connect to a laptop using an Ethernet cable, allowing the laptop to create a remote session to the Raspberry Pi over this network connection. The Model B versions of the Raspberry Pi include built-in Ethernet ports that support 10/100Mbit Ethernet. The Ethernet ports on the Raspberry Pi feature auto crossover detection, so that they can be used with standard Ethernet patch cables. A short network patch cable between the Ethernet port of the Raspberry Pi and the Ethernet port on the laptop provides the necessary physical network connection between them. The Raspberry Pi board can be powered directly from a USB port on the laptop, so a short USB to MicroUSB cable can handle this function. A small amount of configuration on both the Raspberry Pi computer and the laptop is required to establish the network connection. Much of this configuration can be scripted or preprogrammed on the SD card image on the Raspberry Pi before the course starts.

Illustration 1: Raspberry Pi 3 configured as a portable computer with a 7-inch touch screen and portable wireless keyboard.

In late 2015, the Raspberry Foundation released a new model of Raspberry Pi called the Zero. This board is less than half the size of the traditional Raspberry Pi boards, yet retains full hardware compatibility with them. The reduction in board size also reduced the components on the board as well, creating a simplified design and lowering the cost to $5 per board. This board is deceptively limited, with only a single usable microUSB port for communication, a separate microUSB that is only used to supply power, and a mini HDMI port available for the display. The Raspberry Pi Zero can be used in a traditional mode, but will require an adapter for the mini HDMI port and a special USB OTG hub to break out the single microUSB port into several full-sized USB ports. The latest version of the Raspberry Pi Zero board now includes built-in Bluetooth and Wi-Fi capability, further reducing the number of peripherals required.

The hardware specifications for the Raspberry Pi Zero are nearly identical to the specifications of the original Raspberry Pi. The processors are the same model, but the Raspberry Pi Zero’s single core processor is clocked at 1Ghz to provide a noticeable speed boost over the older board. The memory remains the same as the older board at 512MB. What differentiates the Raspberry Pi Zero from all other models is that the microUSB communication port is configured as a USB OTG port, allowing it to act as either a USB host or client device. This new feature presents another approach that has the potential to reduce many of the problems of the traditional Raspberry Pi based solution.

With the USB OTG port and the Linux-based operating system that runs on the Raspberry Pi Zero, the device can present itself to a host computer as a USB peripheral that is defined by the software running on it. These devices include a keyboard, mouse, Ethernet adapter, Serial adapter, and USB Memory stick to name a few. The host laptop identifies these virtual USB devices and automatically loads the appropriate drivers for them when it is connected. When properly configured, the Raspberry Pi Zero can also present itself to the host computer as any combination of these devices at the same time. The only hardware required is the Raspberry Pi Zero board, a preconfigured micro SD card, the host laptop provided by the student, and a single USB cable that supplies both power and the communication connection to the Raspberry Pi Zero. The USB OTG port on the Raspberry Pi Zero makes the “Server on a USB port” solution possible.

2. HOW WE TESTED

Setup Environment

The Raspberry Pi Zero was configured to present itself to the host laptop as a combination of an Ethernet adapter, Serial Adapter, and USB Memory stick. By presenting itself as a USB Ethernet adapter for the laptop, the Raspberry Pi Zero can establish a TCP/IP based network between itself and the laptop just as a standard Raspberry Pi would when using an Ethernet patch cable. By presenting itself as a USB Serial port, the Raspberry Pi Zero can communicate with the host using a standard terminal program for text-only mode computing. By presenting itself as a USB Memory stick to the laptop, the Raspberry Pi Zero can store preconfigured programs and documentation that are usable on the host system to connect to the Raspberry Pi Zero itself. As a result, this approach presents all the benefits of the traditional Raspberry Pi approach using the laptop as a host, while reducing the time, skills, and extra equipment required for configuring the host laptop to connect to the Raspberry Pi.

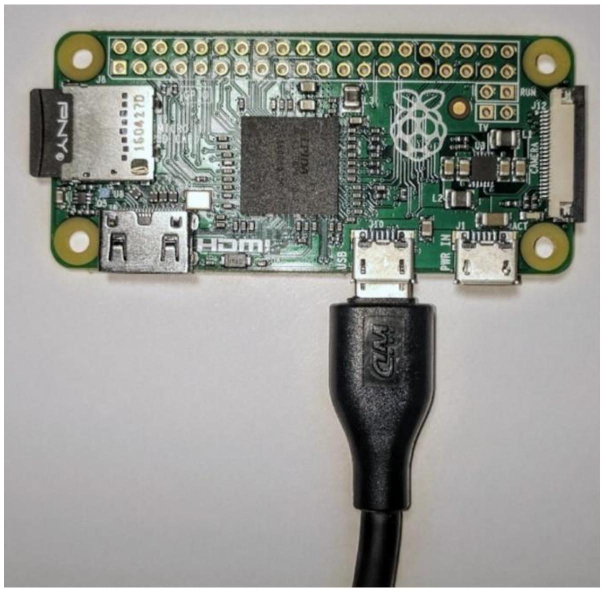


Illustration 2: Proper connection of microUSB cable when connecting to a laptop.

Student Environment

Once the Raspberry Pi Zero is configured, the only requirements for using it are the Raspberry Pi device itself, the microSD card, a laptop computer, and a USB cable (Type A to microUSB). To use the device, connect the microUSB side of the USB cable to the Raspberry Pi Zero port labeled USB (Illustration 2) then connect the other end of the USB cable to the USB port of the laptop. The Raspberry Pi Zero LED will illuminate, and the boot sequence should be complete within 90 seconds. Once booted, the operating system installed on the laptop will detect the new USB hardware and install the drivers. One of these drivers will be for the mass storage device, which should be mountable to the host operating system and function just like a USB thumb drive with full permissions. Once mounted, this drive is configured to contain user documentation and applications suitable for the host Operating System installed on the laptop. The supported host Operating Systems include Windows, Apple OSX, and Linux. The applications consist of portable versions of PuTTY and VNC programs, each preconfigured to connect to the Raspberry Pi Zero itself over the USB network adapter driver. Each set of programs are organized in separate folders labeled for each of the compatible host Operating Systems. The student selects the folder appropriate for their laptop Operating System and executes the appropriate application. The VNC application presents the student with the graphical remote desktop interface of the Raspberry Pi Zero, and the PuTTY application presents the student with a remote terminal session to the Raspberry Pi Zero.

Illustration 3: Raspberry Pi Zero connected to a Windows 7 laptop with a single USB cable for power and communication. Shown with PuTTY and VNC sessions open.

3. RESULTS AND DISCUSSION

As of late 2016, 11 million Raspberry Pi computers were sold throughout the world since its release in 2012 (Thomas, 2016). Originally designed to promote computer science education in developing countries, the Raspberry Pi became much more than its developers anticipated. This computer is now in the hands of millions of people all over the world who are contributing software, hardware accessories, documentation, and training materials. All Raspberry Pi models are compatible with each other, including the new Raspberry Pi Zero that is the main topic of this paper.

The Raspberry Pi as a platform provides a general-purpose computing environment suitable for most courses designed to train students in administrative functions in a Linux based environment. It has the potential to meet all four of the requirements listed in the ideal solution when compared to the dedicated computer laboratory. It is less expensive than dedicated computer laboratories or server based solutions, and it is easier and quicker to implement and start to use than virtualized servers. Once the configuration is complete, the Operating System image can be replicated to multiple SD cards and distributed to the students with the Raspberry Pi board and cable. By configuring the SD card images ahead of time, the Raspberry Pi devices can be ready to use by the students on the first day of class. Backups of user data can be performed using built-in Linux utilities, and complete recovery from a mistake is as simple as swapping out an SD card for another pre-configured card.

4. CONCLUSION

The Raspberry Pi computing platform has already made history in teaching computing topics in all levels of education, from elementary schools to University campuses. It is recognized for its ease of use, extreme low cost, and the support it is receiving by educators, students, and enthusiasts throughout the world. The non-profit Raspberry Pi Foundation that manages the development of the platform has ensured that future versions of the platform are released slowly and remain compatible to ensure long term viability. The entire platform is based on Free and Open Source concepts, including most of the hardware and its GNU/Linux based operating system.

This paper began as a study towards making the Raspberry Pi less cumbersome for students by eliminating the need for using external peripherals by utilizing the student's own laptop computers. An exhaustive search through many peer-reviewed articles produced several instances of Raspberry Pi use in academia, but none of them discussed using the platform in the classroom with this configuration. One paper discussed the problem with external peripherals as a significant drawback when using the Raspberry Pi in the classroom (Barker et al., 2013).

Using two separate computers in a direct-attached networked configuration is not new or revolutionary. It is quite common for Raspberry Pi developers and enthusiasts to connect the device to their laptops with an Ethernet cable for communications and a USB cable for power. This is done to enhance portability and make use of advanced tools installed on the laptop to aid in development. The Raspbian Operating System that runs on the Raspberry Pi already has everything it needs for remote connection from a laptop, including SSH and VNC server services. Client software for the laptop, such as PuTTY and VNC client, is freely available online for Windows, Mac OSX and Linux operating systems. The most difficult part of this configuration is downloading, installing, and configuring the software and network settings on the host computer.

The Raspberry Pi Zero was introduced to the market as this study was being conducted, and presented a potentially simplified method of connecting the Raspberry Pi to a laptop. The USB2GO interface coupled with the GadgetFS file system available in the Raspbian operating system eliminates the need for the Ethernet patch cable and allows the device to automatically configure the network settings for itself and the host. The emulated mass storage device can contain files for the host, such as preconfigured portable versions of the afore-mentioned PuTTY and VNC client. Adding these programs to the device itself removes the need to download, install and configure local versions of these programs on the host computer. From the perspective of the student, the setup and configuration process is not just simplified, it is eliminated.

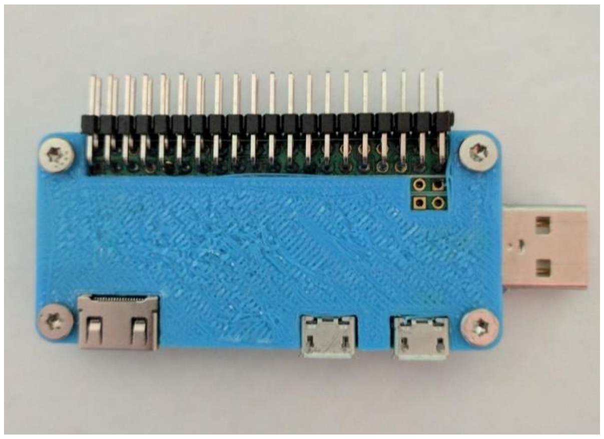


Illustration 4: Raspberry Pi Zero modified with built-in USB connector, GPIO Header, and 3D printed case.

5. REFERENCES

Barker, D., Ferrier, D. E., Holland, P. W., Mitchell, J. B., Plaisier, H., Ritchie, M. G., & Smart, S. D. (2013). 4273π: Bioinformatics education on low cost ARM hardware. *BMC Bioinformatics*, *14*, 243. https://doi.org/10.1186/1471-2105-14-243

Conlon, M. P., & Mullins, P. (2011). ISEDJ Creating and Using a Computer Networking and Systems Administration Laboratory Built Under Relaxed Financial Constraints. *Information Systems Education Journal*, *9*(4), 4.

Davison, C. B. (2015). Assessing It Student Performance Using Virtual Machines. *Tech Directions*, *74*(7), 23–25.

Kay, R. H., & Lauricella, S. (2014). Investigating the Benefits and Challenges of Using Laptop Computers in Higher Education Classrooms. *Canadian Journal of Learning and Technology*, *40*(2). Retrieved from https://libproxy.usouthal.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=eric&AN=EJ1030425&site=eds-live

Murphy, M., & McClelland, M. (2009). My Personal Computer Lab: Operating in the "Cloud." *Information Systems Education Journal*, *7*(93). Retrieved from http://isedj.org/7/93/index.html

Owen, W., & Black, M. (2003). Designing Labs for a Sequence of Network Courses. *Information Systems Education Journal*, *1*(31). Retrieved from http://isedj.org/1/31/index.html

Reck, R. M., & Sreenivas, R. S. (2016). Developing an Affordable and Portable Control Systems Laboratory Kit with a Raspberry Pi. *Electronics*, *5*(3), 36. https://doi.org/10.3390/electronics5030036

Thomas, J. (2016, November 27). Raspberry Pi computer sales hit 11 million. Retrieved April 1, 2017, from http://www.cambridge-news.co.uk/news/cambridge-news/raspberry-pi-computer-sales-hit-12237410

Wilkins, R., & Richardson, B. (2013, September). UEFI Secure Boot in Modern Computer Security Solutions 2013. Retrieved from http://www.uefi.org/sites/default/files/resources/UEFI_Secure_Boot_in_Modern_Computer_Security_Solutions_2013.pdf

Annexures

Platform/approach	Cost to implement/maintain per student	Allows elevated permissions	Setup time	Recovery time	Skill level to implement/use
Typical computer lab	$$$$	-	-	-	-
Dedicated computer lab	$$$$	X	High	High	Low
Dedicated laptop – university issued	$$$$	X	High	Medium	Low
Dedicated laptop – student owned	-	X	High	High	High
Virtual machine on student laptop	-	X	High	Medium	High
Virtual machine on shared server	$$$$	X	Medium	Medium	Medium
Live CD/USB	$$	X	Low	Low	High
Central server with remote access	$$$$	-	-	-	-
Traditional Raspberry Pi with peripherals	$$$	X	Low	Low	Low
Raspberry Pi Zero with student laptop	$$	X	Low	Low	Low

Using Learning Journals to Increase Metacognition, Motivation, and Learning in Computer Information Systems Education

Guido Lang
guido.lang@quinnipiac.edu
Quinnipiac University
Hamden, CT 06518

Abstract

While regular reflection has been found to be a key practice in agile software development, the use of learning journals in computer information systems (CIS) education has yet to be studied systematically. Learning journals are writing-to-learn interventions that use writing as a medium to facilitate metacognition. A randomized controlled trial investigating the effects of learning journals on metacognition, motivation, and learning was implemented in an undergraduate computer information systems course on web design. Students were randomly assigned to respond to five metacognitive writing prompts (learning journal condition) or five non-metacognitive writing prompts (control condition) over the course of ten weeks. Results suggest that while learning journals increase metacognitive awareness and intrinsic motivation, they do not affect learning directly. A post-hoc quantitative content analysis of the learning journals found that certain linguistic dimensions are associated with higher metacognition, motivation, and learning. While students' use of assent and informal words in learning journals is positively correlated with metacognitive awareness and intrinsic motivation, their use of differ words is negatively correlated with metacognitive awareness, intrinsic motivation, and final grades. Hence, instructors should implement learning journals and consider targeted coaching to help students achieve greater metacognition, motivation, and learning.

Keywords: learning journals, metacognition, motivation, learning

1. INTRODUCTION

Agile software development emphasizes regular reflection in order to enable continuous learning (Nerur & Balijepally, 2007). Reflective practice helps developers determine if and to what extent processes should be expanded, adapted, altered, or abandoned (Hoda, Babb, Nørbjerg, 2013). In fact, the Reflective Agile Learning Model provides specific guidelines for embedding self-reflection into an agile software development cycle through reflection-in-action and reflection-on-action (Babb, Nørbjerg, Hoda, 2014). While reflection-in-action emphasizes reflecting on an incident while it occurs, reflection-on-action emphasizes reflecting on an incident after it occurred (Schön, 1984). Reflection-on-action may involve reflective writing in the form of journal entries.

In the context of computer information systems (CIS) education, students can be encouraged to engage in reflection-on-action through learning journals. Learning journals are writing-to-learn interventions that use writing as a medium to facilitate metacognition (Cooper, 2006). Metacognition is the ability to understand and control one's own learning processes (Schraw & Dennison, 1994). It has been shown to be an important predictor of academic success (Pintrich, 2002) that can be learned and further developed (White & Frederiksen, 1998). A significant amount of research on learning journals has produced mixed findings, suggesting that their effectiveness is highly context-dependent (Bangert-Drowns, Hurley, & Wilkinson, 2004). Only few studies have evaluated learning journals in business disciplines (e.g. Cathro, O'Kane, & Gilbertson, 2017) and their effectiveness has yet to be empirically validated in the context of CIS education.

To help address this gap, the present research evaluates the effectiveness of learning journals in increasing metacognition, motivation, and learning through a randomized controlled trial in an undergraduate CIS course on web design.

2. BACKGROUND

The importance of writing for learning has been explored at least since the early 1970s (Emig, 1977). Early work focused on proposing general arguments without explicating and testing the mechanisms by which learning might be enhanced through writing-to-learn interventions (Ackerman, 1993). Subsequent research in the 1980s began to define and disentangle the effects of various contextual factors, such as the specific nature of the writing prompts (Durst & Newell, 1989). Since then, a large number of studies have focused on the conditions under which writing appears to facilitate learning. In particular learning journals, which are writing tasks that foster beneficial metacognitive learning strategies, have been widely studied in the context of higher education (Langer, 2002).

However, a comprehensive meta-analysis of research on learning journals found considerable variation in their effect on metacognition, motivation, and learning (Bangert-Drowns, Hurley, & Wilkinson, 2004). Moderators that were identified to potentially influence effectiveness include the overall treatment length, amount of time spent writing, and use of metacognitive reflection prompts. Surprisingly, longer writing assignments were found to be counterproductive in classroom contexts. Taken together, these findings suggest that effective learning journals tend to be semester-long assignments using metacognitive reflection prompts that can be completed in less than 10 minutes.

The present work empirically validates these recommendations through a randomized controlled trial in an undergraduate CIS course. It was hypothesized that students who maintain a learning journal over the course of the semester will subsequently exhibit greater metacognition (H1), motivation (H2), and learning (H3), than students who do not maintain a learning journal. Figure 1 depicts the research model.

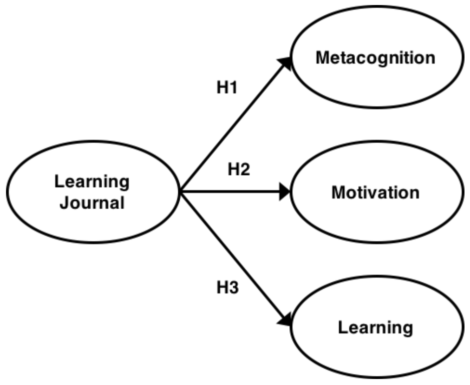


Figure 1. Research Model

3. METHODOLOGY

A randomized controlled trial was implemented in three sections of CIS 267 (HTML & CSS), which was taught at Quinnipiac University in Fall 2016 (N = 98). CIS 267 is an undergraduate elective CIS course that places a heavy emphasis on experiential, hands-on learning through weekly coding projects.

At the beginning of the semester, all students completed a pre-test survey measuring demographic factors, previous knowledge ("What is your knowledge of HTML and CSS?" anchored at 1: None at all and 5: A great deal), and learning style (LSI; Kolb & Kolb, 2005). Students were asked to complete a journal writing assignment every two weeks for ten weeks of the semester, totaling five journal entries. Students were randomly assigned to one of two treatment conditions, which determined the content of their journal writing assignment.

Table 1. Writing Prompt Used in Learning Journal Condition

Please create a post about the coding projects you completed in this course so far. Specifically, please write one short paragraph for each of the following questions: 1. *What are the similarities/differences between the coding projects?* 2. *What was the ideal strategy for completing the coding projects?* 3. *What will you do differently when working on coding projects in the future?*

In the learning journal condition, the journal writing assignment consisted of a prompt that was meant to facilitate metacognition. The prompt was designed following an established metacognitive questioning strategy (Mevarech & Kramarski, 1997). Table 1 shows the metacognitive writing prompt that was used in the learning journal condition.

In the control condition, the journal writing assignment consisted of a prompt that was unrelated to metacognition. The specific prompt used in the control condition can be found in Table 2 below.

Table 2. Writing Prompt Used in Control Condition

Please create a post about a website that has won the "Site of the Day" award from AWWWARDS. Choose one website from which you would possibly like to incorporate one or more design elements into your final project. Please write one short paragraph for each of the following questions:

1. *Which website did you choose and why?*
2. *What design element(s) would you possibly like to incorporate into your final project and why?*
3. *Judging by the websites you saw while browsing the awards, how common are the design element(s) that you chose?*

Over the course of ten weeks, each student answered the same prompt a total of five times, i.e. students in the learning journal condition answered the writing prompt that was meant to facilitate metacognition every two weeks and students in the control condition answered the writing prompt that was unrelated to metacognition every two weeks. At the end of the semester, all students completed a post-test survey measuring metacognitive awareness (Schraw & Dennison, 1994) and intrinsic motivation (McAuley, Duncan, & Tammen, 1987). Students' final grades were used as a measure of learning. Final grades were calculated based on students' performance in weekly coding projects (weighted 70%), a final project (weighted 15%), a final paper, class participation, and the journal assignments (each weighted 5%). Thus, the research employs a single factor (learning journal vs. control) between subjects experimental design.

4. RESULTS

Demographics and Randomization

A total of N = 98 students participated in the study. Detailed demographics of the sample are presented in Table 3.

Table 3. Sample Demographics

Gender	
Male	70 (71%)
Female	28 (29%)
Class Level	
Freshman	0 (0%)
Sophomore	12 (12%)
Junior	34 (35%)
Senior	52 (53%)

Forty-nine (50%) students were assigned to each treatment condition. Multiple independent-samples t-tests were conducted to evaluate if the assignment of students to conditions was random with regards to gender, class level, previous knowledge, and learning style. Neither gender nor class level was different between treatment conditions (t < 1.22, p > .1). Previous knowledge was relatively low (M = 2.00, SD = .76) and also not different between conditions (t = -1.38, p > .1). Learning style was measured using summative dimension values of the LSI (Kolb & Kolb, 2005). Students exhibited a diverging learning style, which is characterized by an emphasis of Concrete Experience (M = 36.11, SD = 4.91) over Abstract Conceptualization (M = 29.47, SD = 5.55) and Reflective Observation (M = 30.27, SD = 5.79) over Active Experimentation (M = 24.15, SD = 5.26). No significant differences of LSI values between conditions were observed (t < 1.69, p > .1). This learning style profile appears to be common among undergraduate CIS students at Quinnipiac University, as it mirrors the results obtained in a previous, unrelated study (Lang, 2017). The students' aggregate learning style profile is shown in Figure 2.

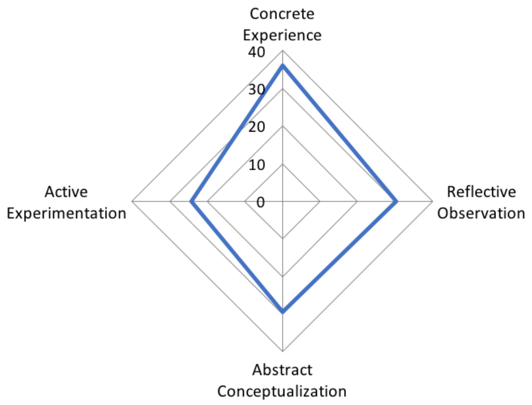


Figure 2: Learning Style Profile

These findings suggest that the assignment of students to conditions was indeed random with regards to gender, class level, previous knowledge, and learning style.

Moreover, multiple independent-samples t-tests were conducted to evaluate the randomness of missing values in the pre- and post-test surveys as well as the journal writing assignments. A total of 18 students had at least one missing value in the pre-test survey, post-test survey, or journal writing assignments. No significant differences emerged (all *t*s < .65, *p*s > .52), suggesting that missing values were indeed randomly occurring.

The writing prompts for both treatment conditions were designed to elicit the same amount of writing. Overall, students in both treatment conditions wrote journal entries of comparable length, as measured by word count (see Table 4). Likewise, journal entries in both treatment conditions exhibited a declining trend in terms of word count over time. Thus, any observed differences in the dependent variables cannot be attributed to differences in the amount of writing between treatment conditions.

Taken together, these results indicate that the random assignment of students to treatment conditions was successful and that students' behavior across treatment conditions was comparable with regards to missing responses and the length of journal entries.

Table 4. Word Count of Journal Entries

	Learning Journal Condition	*Control Condition*	*Difference*
Entry 1	209.32 (78.68)	229.23 (116.86)	19.91^{ns}
Entry 2	169.69 (64.68)	183.28 (104.68)	13.59^{ns}
Entry 3	150.37 (48.79)	175.15 (113.78)	24.77^{ns}
Entry 4	156.01 (54.49)	171.97 (85.87)	15.96^{ns}
Entry 5	147.95 (54.37)	170.85 (88.27)	22.90^{ns}
Total	833.34 (301.01)	930.48 (509.47)	97.14^{ns}

^{ns} *p* > .1

Dependent Variables

The data were analyzed using partial least squares path modeling in R (plspm package version 0.4.9). A two-step approach based on the recommendations by Henseler, Hubona, and Ray (2016) was used: First, the reliability and validity of the measurement model was established. Based on previous research (Teo & Lee, 2012), metacognitive awareness was modeled using two factors: knowledge about cognition (MA-K) and regulation of cognition (MA-R). Likewise, intrinsic motivation (IM) was modeled using a single factor (Monteiro, Mata, and Peixoto, 2015). Final grade was modeled as a single-item factor. Likewise, learning journal was modeled as a single-factor using a dummy variable (*0: Control condition, 1: Learning journal condition*). Items with factor loadings of .40 or less and items with higher cross-loadings on other factors were removed from further analysis. As a result, the final measurement model exhibits adequate reliability and validity (all Dillon-Goldstein's *ρ*s > .84, Cronbach's *α*s > .78). Detailed results supporting the reliability and validity of the measurement model can be found in Appendix A.

Next, the path coefficients of the model were evaluated using a bootstrapping method with 100 samples. The results support most of the hypothesized effects: At the end of the semester, students in the treatment condition exhibited greater knowledge about cognition than students in the control condition (*β* = 0.25, *p* < .05). Moreover, students in the treatment condition showed greater regulation of cognition than students in the control condition (*β* = 0.25, *p* < .05). Likewise, students in the treatment condition had higher intrinsic motivation than students in the control condition (*β* = 0.22, *p* < .05). However, no difference in final grades between students in the treatment and control conditions was observed (*β* = 0.01, *p* > .1). Thus, H1 and H2 are supported, while H3 is not supported. Figure 3 shows the results of the path analysis.

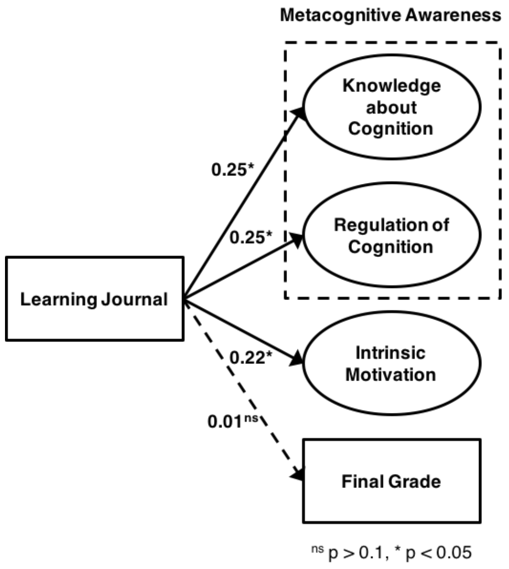


Figure 3. Results

Taken together, these findings provide evidence that learning journals increase metacognition and motivation.

Post-Hoc Quantitative Content Analysis
A post-hoc quantitative content analysis of the learning journals was performed in order to shed light on the potential mechanisms underlying the observed effects. The Linguistic Inquiry and Word Count (LIWC) software (Pennebaker et al., 2015) was used to quantitatively analyze the learning journal content. The LIWC software compares words against a comprehensive dictionary and counts the percentage of words that reflect different emotions, thinking styles, social concerns, and other psychologically-relevant categories (Tausczik & Pennebaker, 2010).

The LIWC software generated data for each journal entry across 93 linguistic dimensions. The linguistic dimensions were subsequently entered into separate correlation analyses with the scaled factor scores for each of the dependent variables: both subscales of metacognitive awareness, i.e. knowledge about cognition (MA-K) and regulation of cognition (MA-R), intrinsic motivation (IM), and final grade (Grade). Three linguistic dimensions had several significant correlations with the dependent variables, as shown in Table 5.

Table 5. Correlations between Linguistic Dimensions and Dependent Variables

	MA-K	*MA-R*	*IM*	*Grade*
Assent	.34*	.27†	.27†	.06^{ns}
Differ	-.20^{ns}	-.29*	-.41**	-.32*
Informal	.19^{ns}	.32*	.27†	-.04^{ns}

^{ns} $p > .1$, †p < .1, * $p < .05$, ** $p < .01$

The linguistic dimension of "assent," which includes words such as "absolutely," "agree," and "alright," was positively correlated with knowledge about cognition (r = .34, p < .05), regulation of cognition (r = .27, p < .1), and intrinsic motivation (r = .27, p < .1). This suggests that students who wrote learning journals using more assent words exhibited greater knowledge about cognition, regulation of cognition, and intrinsic motivation than students who wrote learning journals using less assent words.

The linguistic dimension of "differ," which includes words such as "actually," "although," and "despite," was negatively correlated with regulation of cognition (r = -.29, p < .05), intrinsic motivation (r = -.41, p < .01), and final grade (r = -.32, p < .05). This result indicates that students who wrote learning journals using more differ words had lower regulation of cognition, intrinsic motivation, and final grades than students who wrote learning journals using fewer differ words.

Lastly, the linguistic dimension of "informal," which includes words such as "badass," "cool," and "geeky," was positively correlated with metacognitive awareness (again, although correlations with both subscales were positive, only the correlation with regulation of cognition (r = .32, p < .05) and intrinsic motivation (r = .27, p < .05). This finding suggests that students who wrote learning journals using more informal words subsequently showed higher regulation of cognition and intrinsic motivation than students who used less informal words in their learning journals.

Table 6 provides additional examples for each of the three linguistic dimensions.

Table 6. Examples for Linguistic Dimensions

Dimension	*Examples*
Assent	absolutely, agree, alright, indeed, yes
Differ	actually, although, despite, however, otherwise
Informal	badass, cool, geeky, kinda, sucks

Taken together, the results of the post-hoc quantitative content analysis suggest that linguistic dimensions in learning journals may affect metacognition, motivation, and learning. In particular, the use of assent and informal words may increase metacognitive awareness and intrinsic motivation, while the use of differ words may decrease metacognitive awareness, intrinsic motivation, and final grades.

5. DISCUSSION

The results of the randomized controlled trial lend support to the hypotheses that learning journals increase metacognition (H1) and motivation (H2). However, the relationship between learning journals and learning (H3) is not as straight forward, as no direct effect has been observed. Although relatively small in size, these effects were found after random assignment of students to treatment conditions, which suggests that they hold across different genders, class levels, levels of previous knowledge, and learning styles. Since students were asked to reflect upon their learning every two weeks for a total of ten weeks, it is likely that the regular practice of metacognition combined with the focus on self-help and continuous improvement, ultimately helped students increase their metacognition and motivation.

The post-hoc analysis focusing on linguistic dimensions of the learning journals indicates that assent and informal words may have the potential to magnify these effects, while differ words may play an attenuating role. Stated differently, on the one hand, learning journals that focused on positive insights with which students agreed and that used informal language were associated with higher levels of metacognitive awareness and intrinsic motivation. On the other hand, learning journals that focused on contrasting insights with disagreement were associated with lower levels of metacognitive awareness, intrinsic motivation, and final grades.

The implications of these findings for CIS instructors are two-fold: First, instructors are well-advised to incorporate learning journals into their classes. Although learning journals are not a silver bullet to increase learning, they increase metacognition and motivation. Given the numerous benefits of increasing metacognition and motivation for students inside and outside the classroom, the additional work required in administering and grading these assignments appears to be worth the effort. Second, instructors should consider guiding students in their learning journal writing to focus on positive insights with which they agree, while encouraging the use of informal language. This could be accomplished through targeted coaching and feedback for students.

However, the results of this study, along with its implications, must be viewed in light of the limitations of this study. First, the relatively small sample (*N* = 98) may have been composed of students that were predisposed to react favorably to a learning journal assignment. Second, the fact that the experiment relied solely on CIS students in a web design class may have accidentally enhanced the effectiveness of the learning journal treatment. Third, the specific writing prompts used in the treatment conditions were unique to the subject matter and may not be easily transferable to other CIS courses. Fourth, alternative measures of the dependent variables are available, which may alter the reported effects. Finally, the post-hoc analysis was correlational in nature, which implies that the use of linguistic dimensions may also be the outcome – and not the cause – of higher metacognition and motivation.

Given the shortcomings of this study, additional research is needed to further support its implications. In particular, future research should consider investigating the effects of different metacognitive learning journal prompts, as well as the effects of learning journal coaching strategies in CIS classes.

6. CONCLUSION

Regular reflection is a key component of agile software development. However, learning journals, which are writing-to-learn interventions aimed at increasing metacognition, have hitherto not been systematically investigated in the context of CIS education. Previous research on the effectiveness of learning journals in other disciplines has found mixed results. Moreover, little attention has been given to the effectiveness of learning journals in business and engineering disciplines in general, and CIS in particular. To fill this gap, a randomized controlled trial investigating the impact of learning journals on metacognition, motivation, and learning was conducted in three sections of an undergraduate CIS elective course on web design (*N* = 98).

The findings suggest that while learning journals increase metacognitive awareness and intrinsic motivation, they do not directly affect final grades. A post-hoc quantitative content analysis further suggests that certain linguistic dimensions of the learning journals may differentially affect these dependent variables. In particular, the use of assent and informal words may increase metacognitive awareness and intrinsic motivation, while the use of differ words may decrease metacognitive awareness, intrinsic motivation, and final grades. Given the benefits of increased metacognition and motivation for students, CIS instructors should integrate learning journals into their classes. However, future research should investigate the effects of different metacognitive prompts and coaching when implementing learning journals in CIS courses.

7. REFERENCES

Ackerman, J. M. (1993). The Promise of Writing to Learn. *Written Communication*, *10*(3), 334-370.

Babb, J., Nørbjerg, J., & Hoda, R. (2014). Embedding Reflection and Learning into Agile Software Development. *IEEE Software, 31*(4), 51-57.

Bangert-Drowns, R. L., Hurley, M. M., & Wilkinson, B. (2004). The Effects of School-

Based Writing-to-Learn Interventions on Academic Achievement: A Meta-Analysis. *Review of Educational Research, 74*(1), 29-58.

Cathro, V., O'Kane, P., & Gilbertson, D. (2017). Assessing Reflection: Understanding Skill Development Through Reflective Learning Journals. *Education + Training, 59*(4), 427-442.

Cooper, J. W. (2006). At Issue: Journal Writing in Career and Technical Education: A Tool to Promote Critical Thinking Skills. *Journal of STEM Teacher Education, 43*(2), Article 7.

Durst, R. K., & Newell, G. E. (1989). The Uses of Function: James Britton's Category System and Research on Writing. *Review of Educational Research, 59*(4), 375-394.

Emig, J. (1977). Writing as a Mode of Learning. College *Composition and Communication*, 28, 122-12.

Henseler, J., Hubona, G., & Ray, P. A. (2016). Using PLS Path Modeling in New Technology Research: Updated Guidelines. *Industrial Management & Data Systems, 116*(1), 2-20.

Hoda, R., Babb, J., & Nørbjerg, J. (2013). Toward Learning Teams. *IEEE Software, 30*(4), 95–98.

Kolb, A. Y., & Kolb, D. A. (2005). The Kolb Learning Style Inventory—Version 3.1 2005 Technical Specifications. Department of Organizational Behavior, Weatherhead School of Management, Case Western Reserve University, Cleveland, OH.

Lang, G. (2017). Agile Learning: Sprinting Through the Semester. *Information Systems Education Journal, 15*(3), 14-21.

Langer, A. M. (2002). Reflecting on Practice: Using Learning Journals in Higher and Continuing Education. *Teaching in Higher Education, 7*(3), 337-351.

McAuley, E., Duncan, T., & Tammen, V. V. (1987). Psychometric Properties of the Intrinsic Motivation Inventory in a Competitive Sport Setting: A Confirmatory Factor Analysis. *Research Quarterly for Exercise and Sport*, 60, 48-58.

Mevarech, Z. R., & Kramarski, B. (1997). IMPROVE: A Multidimensional Method for Teaching Mathematics in Heterogeneous Classrooms. *American Educational Research Journal*, 34, 365-395.

Monteiro, V., Mata, L., & Peixoto, F. (2015). Intrinsic Motivation Inventory: Psychometric Properties in the Context of First Language and Mathematics Learning. *Psychology/ Psicologia Reflexao e Critica, 28*(3), 434-443.

Nerur, S., & Balijepally, V. (2007). Theoretical Reflections on Agile Development Methodologies: The Traditional Goal of Optimization and Control is Making Way for Learning and Innovation. *Communications of the ACM, 50*(3), 79–83.

Pennebaker, J. W., Booth, R. J., Boyd, R. L., & Francis, M. E. (2015). Linguistic Inquiry and Word Count: LIWC2015. Austin, TX: Pennebaker Conglomerates.

Schön, D. (1984). *The Reflective Practitioner: How Professionals Think In Action*. Basic Books: New York.

Schraw, G., & Dennison, R. S. (1994). Assessing Metacognitive Awareness. *Contemporary Educational Psychology*, 19, 460-475.

Tausczik, Y. R., & Pennebaker, J. W. (2010). The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods. *Journal of Language and Social Psychology, 2*(9), 24-54.

Teo, T., & Lee, C. (2012). Assessing the Factorial Validity of the Metacognitive Awareness Inventory (MAI) in an Asian Country: A Confirmatory Factor Analysis. *International Journal of Educational and Psychological Assessment, 10*(2), 92-103.

White, B. Y., & Frederiksen, J. R. (1998). Inquiry, Modeling, and Metacognition: Making Science Accessible to All Students. *Cognition and Instruction*, 16, 3–118.

Editor's Note:

This paper was selected for inclusion in the journal as an EDSIGCON 2017 Meritorious Paper. The acceptance rate is typically 15% for this category of paper based on blind reviews from six or more peers including three or more former best papers authors who did not submit a paper in 2017.

.

Appendix A: Survey Instruments

Table 7. Construct Descriptive and Reliability Measures

	Learning Journal	*MA-K*	*MA-R*	*IM*	*Final Grade*
Mean	0.49	5.83	5.25	6.07	94.64
SD	0.50	0.60	0.87	0.59	8.51
AVE	1.00	0.30	0.42	0.43	1.00
ρ	1.00	0.84	0.96	0.91	1.00
α	1.00	0.78	0.95	0.88	1.00

Table 8. Inter-Construct Correlations

	Learning Journal	*MA-K*	*MA-R*	*IM*	*Final Grade*
Learning Journal	1.00				
MA-K	0.25	1.00			
MA-R	0.25	0.64	1.00		
IM	0.22	0.45	0.29	1.00	
Final Grade	0.01	0.18	0.27	0.32	1.00

Table 9. Item Loadings for Regulation of Cognition (MA-R) Scale

Item	*Loading*
I pace myself while learning in order to have enough time.	0.66
I think about what I really need to learn before I begin a task.	0.63
I set specific goals before I begin a task.	0.61
I ask myself questions about the material before I begin.	0.74
I think of several ways to solve a problem and choose the best one.	0.74
I read instructions carefully before I begin a task.	0.49
I organize my time to best accomplish my goals.	0.51
I slow down when I encounter important information.	0.59
I consciously focus my attention on important information.	0.71
I focus on the meaning and significance of new information.	0.62
I create my own examples to make information more meaningful.	0.54
I try to translate new information into my own words.	0.58
I use the organizational structure of the text to help me learn.	0.57
I ask myself if what I'm reading is related to what I already know.	0.62
I ask myself periodically if I am meeting my goals.	0.62
I consider several alternatives to a problem before I answer.	0.63
I ask myself if I have considered all options when solving a problem.	0.82
I periodically review to help me understand important relationships.	0.71
I find myself analyzing the usefulness of strategies while I study.	0.77
I find myself pausing regularly to check my comprehension.	0.69
I ask myself questions about how well I am doing while learning something new.	0.74
I change strategies when I fail to understand.	0.63
I re-evaluate my assumptions when I get confused.	0.69
I stop and go back over new information that is not clear.	0.66
I know how well I did once I finish a test.	0.59
I ask myself if there was an easier way to do things after I finish a task.	0.66
I summarize what I've learned after I finish.	0.51
I ask myself how well I accomplish my goals once I'm finished.	0.65
I ask myself if I have considered all options after I solve a problem.	0.71
I ask myself if I learned as much as I could have once I finish a task.	0.56

Table 10. Item Loadings for Knowledge about Cognition (MA-K) Scale

Item	*Loading*
I understand my intellectual strengths and weaknesses.	0.47
I know what the teacher expects me to learn.	0.43
I am good at remembering information.	0.62
I have control over how well I learn.	0.47
I am a good judge of how well I understand something.	0.59
I learn more when I am interested in the topic.	0.44
I try to use strategies that have worked in the past.	0.53
I find myself using helpful learning strategies automatically.	0.67
I learn best when I know something about the topic.	0.51
I know when each strategy I use will be most effective.	0.64

Table 11. Item Loadings for Intrinsic Motivation (IM) Scale

Item	*Loading*
I enjoyed the projects in this course very much.	0.76
The projects in this course were fun to do.	0.72
I thought the projects in this course were boring.	-0.48
The projects in this course did not hold my attention at all.	-0.47
I would describe the projects in this course as very interesting.	0.66
I thought the projects in this course were quite enjoyable.	0.69
I think I am pretty good at the projects in this course.	0.71
After working at the projects in this course for awhile, I felt pretty competent.	0.73
I am satisfied with my performance at the projects in this course.	0.65
I was pretty skilled at the projects in this course.	0.62
I couldn't do the projects in this course very well.	-0.52
I believe the projects in this course could be of some value to me.	0.69
I think that doing the projects in this course is useful for me.	0.68
I think that doing the projects in this course is useful for my career.	0.62
I think the activities in this course are important to do because they can help me in my career.	0.65
I would be willing to do the projects in this course again because they have some value to me.	0.69
I think doing the projects in this course could help me to get a job/internship.	0.74
I believe doing the projects in this course could be beneficial to me.	0.68
I think the projects in this course are important.	0.59

Triangulating Coding Bootcamps in IS Education: Bootleg Education or Disruptive Innovation?

Leslie J Waguespack
lwaguespack@bentley.edu
Computer Information Systems
Bentley University
Waltham, Massachusetts 02452, USA

Jeffry S. Babb
jbabb@wtamu.edu
Computer Information and Decision Management
West Texas A&M University
Canyon, Texas 79016, USA

David J. Yates
dyates@bentley.edu
Computer Information Systems
Bentley University
Waltham, Massachusetts 02452, USA

Abstract

Coding bootcamps number in the hundreds world-wide despite repeated predictions of their demise over the past few years. Fueled by a resurgent economy and a persistent shortage of app developers and computer systems engineers, bootcamps tout a fast-track to a six-figure salary for as little as one-eighth the tuition dollars or time investment of a nominal four-year information systems baccalaureate degree. Bootcamps represent an enticing opportunity for: a) high school graduates unconvinced of the return on the time and money investment in a liberal arts education, b) college graduates who find their career potential limited by their baccalaureate major, or c) experienced workers seeking a change of profession. Although potentially disruptive, and generally neither accredited nor affiliated academically, bootcamps introduce opportunities for innovation in terms of structure, organization, curriculum, and pedagogy for traditional computing education in higher education, which we explore in this paper.

Keywords: IS education, Coding bootcamps, IS curriculum, IS workforce preparation

1. INTRODUCTION

The emergence of coding bootcamps is due in part to the shortage of computing professionals graduating from universities and the broad demand for individuals with hands-on software skills (Geron, 2013). According to Wikipedia, these bootcamps "provide a vocational training for free or a fraction of the cost of a college degree and are a part of the 'Edtech Disruption of Higher Education'" (Wikipedia, 2017b, p. 2). In addition to being less expensive than a college degree, coding bootcamps take less time by delivering an immersive learning experience, often in 8 to 12 weeks, after which students have learned how to code in a specific domain, e.g., web or mobile software development. Some programs even go into more depth within a domain, e.g. front-end development, iOS, Android or cloud-native development, and some offer a portfolio of such programs. Since most students prefer to learn as part of a community, especially during an immersive (and intense) experience, there are many more classroom-based than on-line bootcamps. Employing in-person cohorts like their military namesake, they offer emotional and psychological support that engenders a sense of confidence and professionalism (Barnett, Basom, Yerkes, & Norris, 2000). And, presumably for job placement reasons, these programs tend to cluster in population centers with a significant presence of technology companies. In the United States, for example, many of the well-known bootcamps have classrooms in San Francisco and New York. Recent diversification away from "just coding" bootcamps has given rise to camps focused on applications, e.g. data analytics, and infrastructure, e.g. Internet of Things.

So, since their inception in 2012, how are these alternative education programs doing? A survey of most U.S. graduates conducted by SwitchUp.Org (2017a), draws the following conclusions based on data gathered from 2014 to 2016:

- 63% of code bootcamp graduates reported increase in salary (in 2016 the average annual salary increase six months after graduation was more than $22,000);
- 80%+ of graduates were satisfied with their bootcamp education (just under 15% were dissatisfied);
- Average class size is 30 with a 1-to-3.8 student instructor ratio;
- Coding bootcamps are a far cheaper and accelerated option than learning to code at a university (the average bootcamp took 10.8 weeks in 2016 and cost $12,800);
- Women learning how to code represent 43% of the bootcamp alumni; and
- The bootcamp market is growing rapidly, projected to double from 2016 to 2017.

This report goes on:

> "There is no doubt that 21st century technology education is trending towards transparent, outcome-driven metrics. … However, key questions remain: Can the type of salary increase seen from the data be sustained in the long-term? As the supply of developers increases to match the demand, will the job market get tighter, or will the creation of tech jobs continue to outmatch the supply of developers over the next few years?" (SwitchUp.Org, 2017a, p. 7)

2. WHAT IS A BOOTCAMP?

Coding bootcamps offer technology-focused training programs that teach programming, frameworks, systems and tools which are in demand in many entry-level software developer positions. Most of these programs teach people with little or no technical coding background how to code, build and deploy applications.

Most information systems and computer science students spend four years to complete their degree. Code bootcamps are designed to distill skills from a four-year degree that are in the greatest market demand and infuse them with relevant methodologies and practices to bridge the perceived gap between contemporary academia and the real world of professional coding (Janicki, Cummings, & Kline, 2014; Yourdon, 2002). With an average program duration of less than 11 weeks, this requires a combination of a singular focus on high demand skills and technologies and high-impact learning with no frills.

As for colleges and universities, there are differences in how different coding bootcamps teach and prepare their students to enter the technology workforce. Because bootcamps lack oversight by federal and state governments or by accrediting bodies, any assessments or judgements about their quality are largely anecdotal. Many differentiating themes that emerge in both favorable and unfavorable anecdotes, however, are familiar to the EDSIG membership and EDSIGCON audience:

- Quality and focus of the curriculum;
- Technical training and know-how of instructors;
- Number of full-time vs. part-time instructors;
- Quality of instruction;
- Emphasis on group projects that simulate real-world development; and
- Availability of mentorship and tutoring for students.

3. BOOTCAMPS AS COMPUTING PROGRAMS

If nothing else, coding bootcamps represent a distinct departure from the prevalent models of career preparation followed by tradition institutions of higher education. A technology focus is obvious in a 2017 ranking of coding bootcamps by an industry monitoring website, SwitchUp.org (2017b), that identifies "The Best of 2017." Table 1 lists their ranking of 31 coding bootcamps and the "catalog" of technology training advertised by each.

		Technology Training Most Frequently Offered																
Ranking	SWITCHUP™ 31 Top Rated Boot Camps	JavaScript	HTML	CSS	Ruby / Rails	ExpressJS/Node	AngularJS	Front-End Web	Full-Stack	UX Design	JQuery	Mobile App	SQL	Git	Swift	iOS	MongoDB	React.js
1	App Academy	■	■	■	■								■					■
2	Le Wagon	■	■	■	■													
3	Bloc	■	■	■	■		■			■					■	■		
4	Ironhack	■	■	■	■	■	■										■	
5	Startup Institute																	
6	Flatiron School	■	■	■					■						■	■		■
7	Epicodus								■	■		■						
8	Actualize	■	■	■	■								■	■				
9	Founders and Coders	■				■												■
10	Dev League	■	■	■		■	■				■							
11	Designation									■								
12	Fullstack Academy	■	■	■		■					■							■
13	V School							■	■	■		■						
14	The Software Guild								■			■						
15	Codeup								■									
16	Grand Circus							■	■			■						
17	Starter League	■								■								
18	Tech Talent South	■	■	■	■													
19	Makers Academy													■				
20	Code Fellows	■						■	■			■				■		
21	Code Foundry	■	■	■			■					■						
22	The Iron Yard	■	■	■		■		■			■		■	■				
23	Dev Mountain	■				■	■			■	■				■	■	■	
24	Thinkful	■	■	■		■	■			■					■			
25	Lighthouse Labs	■	■	■	■	■	■								■			
26	Launch Academy	■	■	■	■			■					■	■				
27	Hack Reactor	■	■			■					■		■	■			■	
28	Coding Dojo	■	■	■	■	■	■	■			■						■	
29	Galvanize	■	■	■				■										
30	Dev Bootcamp	■	■	■	■								■					
31	General Assembly	■	■	■	■			■	■	■								
	# of These Camps Offering	23	19	18	11	10	8	8	8	8	6	6	6	5	5	4	4	4

Table 1. Code Bootcamp Technology Courses

In contrast, even the most technically oriented academic programs in colleges and universities require a significant investment of time and effort to develop a "liberally educated" citizenry. The Association of American Colleges and Universities expresses this model of liberal education thusly:

> "An approach to college learning that empowers individuals and prepares them to deal with complexity, diversity, and change. This approach emphasizes broad knowledge of the wider world (e.g., science, culture, and society) as well as in-depth achievement in a specific field of interest. It helps students develop a sense of social responsibility; strong intellectual and practical skills that span all major fields of study, such as communication, analytical, and problem-solving skills; and the demonstrated ability to apply knowledge and skills in real-world settings." (AACU, 2014, p. 1)

The AACU reported that 74% of surveyed employers in 2013 recommended this model of liberal education for college-bound students. Achieving the breadth of study ascribed to a liberal education involves on the order of 120 to 140 academic credit hours. Each credit hour unit translates into 15 hours of class time and 30 hours of student preparation, according to the U.S. Department of Education, International Affairs Office (USDoE, 2008).

In Table 2 that follows, the contrast between curriculum models, a typical twelve-week coding bootcamp versus the IS degree programs targeting two-year associate and four-year baccalaureate degrees, is dramatic (NCES, 2017).

Also, the difference in the student's overall program cost is significant, 1/4th to 1/8th the cost of an associate program or 1/5th to 1/10th the cost of a baccalaureate program. But, the most compelling differences are entailed by the singular focus of bootcamps on software development, programming. By eschewing the breadth aspect of the *liberal education*, the bootcamps typically require no study of the humanities, sciences, or post-secondary mathematics. In contrast to college programs in IS, by concentrating exclusively on code development and technical IT skills, the bootcamp applies virtually all the contact hours of instruction to coding related topics: more than twice the contact hours typically devoted to coding in the associate degree and nearly 80% more than in the baccalaureate degree. Furthermore, the bootcamp requires only about three calendar months rather than four or eight twelve-week semesters spanning two to four years for the associate or baccalaureate programs, respectively.

(Comparative Dimension)		Coding Bootcamp		Associate IS Program		Baccalaureate IS Program
Program Duration (typical)		3 month		2 year		4 year
Terms per program		1		4		8
Courses per term		6		5		5
Courses per programs		6		20		40
Credit hours per course		3		3		3
Credit hours per Program		18		60		120
Contact Hours per Credit		80		45		45
Curricular Coverage (typical)	# credits		# credits		# credits	
Coding Instruction	17	1360	9	405	12	540
Data Management	0.5	40	6	270	6	270
Development Theory	0.25	20	1	45	6	270
Computing Infrastructure	0	0	3	135	6	270
Application Domain Instruction	0.25	20	12	540	30	1350
Team Theory	0	0	2	90	9	405
Quantitative Methods	0	0	9	405	12	540
Humanities	0	0	9	405	24	1080
Sciences	0	0	9	405	15	675
Total Contact Hours of Instruction		1440		2700		5400
Low End Program Cost		$10,000		$40,000		$80,000
Program Cost per Hour		$7		$15		$15
Tuition		$7,000		$16,000		$32,000
Estimated Room & Board, Etc.		$3,000		$24,000		$48,000
High End Program Cost		$25,000		$120,000		$240,000
Program Cost per Hour		$17		$44		$44
Tuition		$17,000		$90,000		$180,000
Estimated Room & Board, Etc.		$8,000		$30,000		$60,000

Table 2. Bootcamp vs. IS Program Comparison

Triangulating Coding Bootcamps in the Curricular Geography of CC2005

The singularity of focus that bootcamps exhibit is further demonstrated in the curricular focus within the domain of computing education. The Association for Computing Machinery, ACM, and the Institute for Electrical and Electronics

Engineering, IEEE, have consistently worked to normalize the structure and evolution of computing education through a series of published curricular guidelines for particular computing disciplines (CS, CE, IT, IS, and SE), as well as mapping the overall landscape as it did with Computing Curriculum 2005, CC2005 (Shackelford, McGettrick, Sloan, Topi, Davies, Kamali, Cross, Impagliazzo, LeBlanc, & Lunt, 2006, pp. 6-21). In that CC2005 report (the most recent and comprehensive cross-discipline analysis), the task force created graphic characterizations of "what students in each of the disciplines typically do after graduation." Each discipline is portrayed on a field of competency as a "footprint" of proficiency gained by completing the respective academic program. (See Figure 1.)

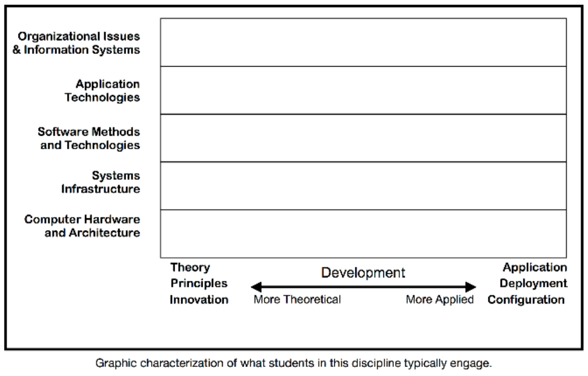
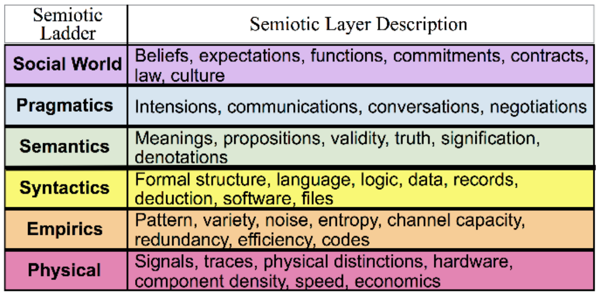

Organizational Issues & Information Systems
Application Technologies
Software Methods and Technologies
Systems Infrastructure
Computer Hardware and Architecture
Theory Principles Innovation
Development
More Theoretical
More Applied
Application Deployment Configuration
Graphic characterization of what students in this discipline typically engage.

Figure 1 - CC2005 Field of Computing Competency

The field of competency delineates computing activities ranging on the Y-axis from hardware issues on the bottom to organizational policy and information management at the top. The X-axis depicts purely *applied* involvement in computing activities to the far right to purely *theoretical* engagement of computing topics to the left.

To emphasize the degree of abstract conceptualization required to bridge between the physical and social world of computing as depicted along the vertical dimension, we superimpose the semiotic ladder as exposed by the footprint of the respective CC2005 discipline. A semiotic framework explicates the expression and transmission of ideas, knowledge, and meaning through human communications (Liu, 2000; Stamper, 1973, 1988). (See Figure 2.) Ascent along the Y-axis of the field of competency (Figure 1) entails a progressive amplification of domain modeling skills and contextualized interpretation requiring a commensurate proficiency in dealing with the complexity of the social context.



Semiotic Ladder	Semiotic Layer Description
Social World	Beliefs, expectations, functions, commitments, contracts, law, culture
Pragmatics	Intensions, communications, conversations, negotiations
Semantics	Meanings, propositions, validity, truth, signification, denotations
Syntactics	Formal structure, language, logic, data, records, deduction, software, files
Empirics	Pattern, variety, noise, entropy, channel capacity, redundancy, efficiency, codes
Physical	Signals, traces, physical distinctions, hardware, component density, speed, economics

Figure 2 - Semiotic Continuum of Constructs

The framework (aka semiotic ladder) depicted in Figure 2 orients and categorizes contextual concerns spanning the sociological and technological landscape that information systems design practice must navigate. The "ladder" represents layered abstractions progressing continuously from bottom to top, anticipating components both material and conceptual arranged as layers of scaffolding one atop the other. Each layer anticipates building blocks in a gradient of abstraction, a vocabulary of metaphorical constructs. Each layer is reminiscent of a virtual machine encapsulating the details of the supporting layers to present a homologous array of structural and behavioral resources upon which to examine the dialog between IS developers and IS consumers.

Although our discussion focuses on the jusxtaposition of coding bootcamps and IS education, we include the footprint of computer science in Figure 3 as an orienting reference point. CS graduates may be engaged in purely theoretical work ranging from efficient utilization of hardware components to systems management supported by machine learning. CS graduates are not generally engaged in off-the-shelf systems deployment or configuration. They are seldom responsible for organizational policy or design of low-level hardware for information infrastructure.

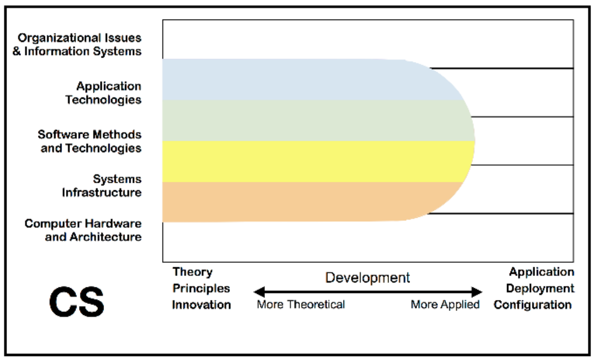

Organizational Issues & Information Systems
Application Technologies
Software Methods and Technologies
Systems Infrastructure
Computer Hardware and Architecture
CS
Theory Principles Innovation
Development
More Theoretical
More Applied
Application Deployment Configuration

Figure 3 - Competency Target of CS (2005)

In CC2005's characterization of the activity of IS graduates, the full breadth of organizational information management policy and operational systems management appears without a significant involvement in hardware or software development theory and practice. (See Figure 4.) Software development is confined to applications and the configuration and deployment of off-the-shelf computing resources focussing largely on supporting business policies and functions.

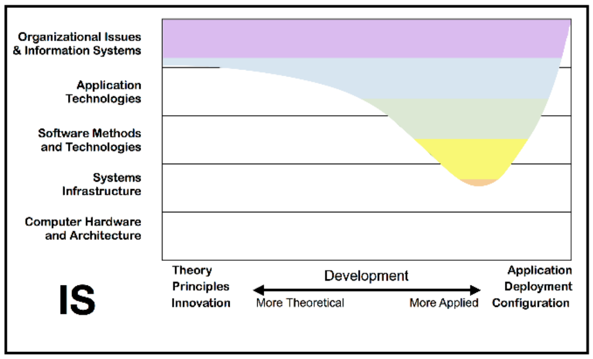

Organizational Issues & Information Systems
Application Technologies
Software Methods and Technologies
Systems Infrastructure
Computer Hardware and Architecture
IS
Theory Principles Innovation
Development
More Theoretical
More Applied
Application Deployment Configuration

Figure 4 - Competency Target of IS (2005)

In our interpretation of the most recent IS curriculum guideline, IS2010, that task force appears to have interpreted the engagement of IS graduates as receding from direct engagement in software development by assuming a more consumer relationship with software systems (Topi, Valacich, Wright, Kaiser, Nunamaker, Sipior, & Vreede, 2010). (See Figure 5.) The task force appears to have envisioned IS graduates more focused on business systems as operational support by adopting a greater dependence upon third-party or out-sourced systems development rather than as builders themselves of strategic artefacts of business.

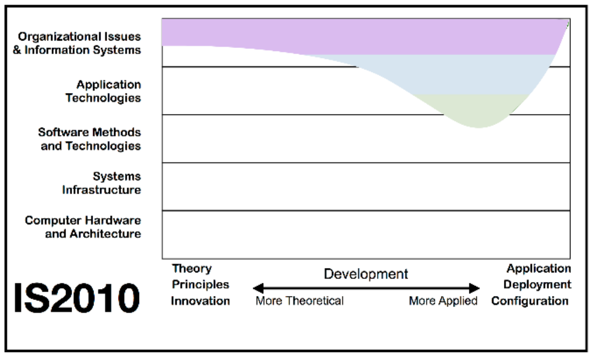

Organizational Issues & Information Systems
Application Technologies
Software Methods and Technologies
Systems Infrastructure
Computer Hardware and Architecture
IS2010
Theory Principles Innovation
Development
More Theoretical
More Applied
Application Deployment Configuration

Figure 5 - Competency Target of IS2010

Bootcamps have never been a curricular focus of the ACM or IEEE guidelines efforts in the mode of CC2005. However, we posit here a footprint depicting the competency target of coding bootcamps in Figure 6 as a means to visualize aspects of the relationship between bootcamps and IS curricula.

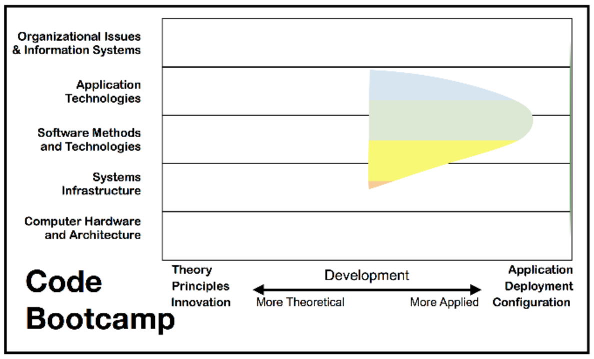

Organizational Issues & Information Systems
Application Technologies
Software Methods and Technologies
Systems Infrastructure
Computer Hardware and Architecture
Code Bootcamp
Theory Principles Innovation
Development
More Theoretical
More Applied
Application Deployment Configuration

Figure 6 - Competency Target of Bootcamps

Bootcamps are purposefully quite "single-minded." They focus on individual computing technologies confined primarily to software production. Their goal is skill-building, rather than problem shaping or theorizing. The problem environment is usually fixed in terms of technology, platform, and tool set. The bootcamp goal is to produce an efficient, reliable, "construction" worker. While there is likely significantly more room to extend professional skill in the realm of software construction, it is clear that industry needs people who can "hammer nails" and "saw wood" rather immediately, hence the bootcamp phenomenon.

Bootcamps vs Accredited Curricula and Programs

While the value of accreditation in higher education is the subject of disparate opinions, nonetheless institutions, schools, programs and curricula each can be (and are often) accredited and such accreditation becomes a mark of quality for various parties: governments, industry, consumers, and citizens (Eaton, 2000, 2012).

As this paper is targeted to faculty in information systems (IS) and computing disciplines, we specifically reflect upon the influence and impact of both AACSB and ABET accreditation. While it is not the case that all IS programs would be either housed in a college of business, nor would they necessarily be accredited by either AASCB (for business) or ABET (computing), these accreditation bodies serve as reasonable proxies by which we may understand the influence that these, and regional and national accreditations, have on curricula and the programs that deliver them.

In the case of AACSB, accredited schools are asked to articulate, measure, evaluate, and improve upon key learning goals for all of programs that fall under the auspices of accreditation such that specified standards are met and maintained. The standards, as is often the case also with regional accreditations, apply to the breadth of activities that extend beyond curriculum. However, the direct and indirect impacts of these standards on curriculum are certainly an intentional byproduct (Gray, Smart, & Bennett, 2017; Solomon, Scherer, Oliveti, Mochel, & Bryant, 2017). Nonetheless, the guidance for curricula, as a component of learning and teaching, are general and broad. Thus, AACSB will examine the processes that lead to a curriculum that focuses on relevant skills and knowledge expected of a particular degree program, any specifics are left to faculty execution of their processes. Thus, while the 2017 specification of AACSB Standard Eight requires an articulation of learning goals which are mapped into course content whereupon some assurances of learning are adhered in a process of curriculum management, these are processes without specificity of content. AACSB Standard Nine provides some expectation that content is consistent with what is normative to a degree program – citing a requisite to care for theories, ideas, concepts, skills, and knowledge, these are to be established in the college.

As many Information Systems curricula have some organizational component, the general business knowledge areas specified by AACSB in Standard Nine would naturally cover some portion of what can be articulated as an Information Systems curriculum. The ACM and AIS Curriculum Guidelines for Undergraduate Degrees in Information Systems (Topi et al., 2010) is the latest installment in a long line of IS curricula guidelines designed to fill in the gaps which, particularly those topical to computing, are not addressed by AACSB. Such model curricula espouse principles regarding what a "standard" IS curriculum might look like while also leaving space for local specializations and adaptations. While not without some controversy regarding the degree of specification of technology content (Longenecker, Feinstein, & Babb, 2013; Reynolds, Adams, Ferguson, & Leidig, 2017; Waguespack, 2011), IS2010 made some clear vital elements, such as data and information management, infrastructure, and Systems Analysis and Design, among others.

With respect to IS curricula, ABET also provides guidelines for programs seeking to acquire and maintain a program-level accreditation. The specifics of the ABET's Computing Accreditation Commission (CAC) extend beyond that provided in the IS2010 report, while perhaps providing less justification and philosophy behind the specifics. The 2017-2018 CAC criteria specify both content – one year or, typically, 10 courses that cover basic content such as: coverage of the fundamentals of application development, data management, networking and data communications, security of information systems, systems analysis and design and the role of information systems in organizations. Within that year's coverage is included advanced coursework to extend these fundamental topics, coverage of a professional environment in which information systems will be applied – often in business - and also quantitative methods and statistics.

The cross-verifying (and validating) and interleaving nature of these externally-validated accreditations on IS curricula are clear. What is less clear is the degree to which bootcamps are providing similar, if not better, grounding in the technical components of an IS curriculum. While the advantages of a college education, even in computing, are somewhat established in the marketplace (Carnevale, Cheah, & Strohl, 2013), it is reasonable to ask what advantages coding and technology bootcamps pose? This question is particularly poignant as there is growing evidence that the labor market may not continue to give preference to the fruits of "traditional" higher education over two-year degrees, diplomas, certificates, MOOCs, and now coding bootcamps (Jepsen, Troske, & Coomes, 2014).

Deciphering a Bootcamp Advantage

It is fair to say that bootcamps are dedicated to providing the maximum of "knowing how" with the minimum of "knowing that" with virtually no attention to "knowing why" (Claxton, 1997)! To achieve their teaching goal of "knowing how," bootcamps employ three tactics: a) topic isolation, b) cohort cohesion, and c) practice immersion.

Topic Isolation: Unlike college or university philosophies that blend a disciplinary focus into a context of liberal studies, bootcamps identify and isolate their curriculum and pedagogy concentrating on the tools and skill set of a niche software development task domain. Common domains are website development, client side or server side programming, mobile device apps, and platform-based application development environments (e.g., LAMP Stack, Ruby on Rails, JavaScript, Java, C#, HTML, CSS, ASP.NET, Python, Swift, iPhone, Android, etc.). (See Table 1.)

Cohort Cohesion: The bootcamp environment engages the group-learn ethic of its military name-sake. Working shoulder-to-shoulder with classmates who virtually all are aiming at the identical academic and tactical goal of IT employment, students gain comrades and competitors with whom and from whom to learn, and draw energy to hold fast to the intense and often grueling 40-hour-plus class weeks. The group familiarity gained in the early weeks of the bootcamp foreshadow the proximity that *industrial-strength* development experiences will engender. At the same time, cohorts offer opportunities to learn team communication and leadership lessons unscripted in the bootcamp curriculum.

Practice Immersion: The typical 40-hour class week provides the close-up demonstration of *introduction to explanation to demonstration to exercise to evaluation* in a cycle that within a cohort provides an academic variant of *close order drill*, "the memorizing of certain actions through repetition until the action is instinctive to the soldiers being drilled." (Wikipedia, 2017a) At the same time there is the opportunity in the presence of the instructor to immediately validate understanding of the introduction and explanation by seeing the technology demonstrated as implemented and then engage a development behavior to replicate the implementation. All this pedagogy proceeds while suppressing the disruptive intervention of days separating class sessions or attention distracted by the study of topics other than the technology subject at hand. These characteristics may accrue advantages that are worthy of further examination in our own community.

4. EXPLOITING BOOTCAMPS AS I.S. CURRICULAR RESOURCES

The natural reaction of college computing programs to coding bootcamps might be to "man the bulwarks" and mount maximum resistance to their rising popularity. Or, higher education might "write off" bootcamps as a philosophically inferior approach to education. But honestly, bootcamps pose a tempting alternative for career entry to the computing profession – not only to the student market, but also to a parental and legislative audience growing skeptical of the cost / benefit or return on investment of traditional higher education.

The fact is that for some time now, there persists a demonstrated shortfall of skilled software developers in the job market (Geron, 2013). Most academics would consider bootcamps a myopic choice, but, bootcamps can equip a committed high school graduate, disillusioned liberal arts degree holder, or a working professional tired of their current career the opportunity to enter the computing career field. But, is there an opportunity for academic programs, particularly IS, to take advantage of this emerging model of programming education?

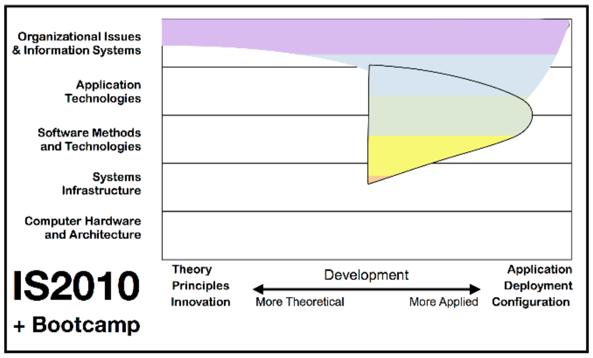

Organizational Issues & Information Systems
Application Technologies
Software Methods and Technologies
Systems Infrastructure
Computer Hardware and Architecture
IS2010
+ Bootcamp
Theory Principles Innovation
Development
More Theoretical
More Applied
Application Deployment Configuration

Figure 7 - Competency Target of IS (2010) + Code Bootcamp

A quick overlay of the posit we offer for the bootcamp footprint of competency onto that of IS2010 indicates that there is relatively little redundancy. (See Figure 7.) In fact, the combination is reminiscent of the IS footprint of CC2005, suggesting that perhaps some opportunities for curricula innovation present themselves.

IS Graduates Need Development Skills

Most IS programs envision their graduates' career entry into computing aligned with, if not embedded in, software systems development. To that end, even with the departure of software development requirements from IS with the IS2010 guidelines, most undergraduate IS programs today find it imperative to offer at least enough software development coursework to legitimize a place for that skill on their graduate's résumé. Relatively few IS graduates will place in positions that are primarily managerial or supervisory without some experience with programming responsibilities. Therefore, training for software development skills remains for the foreseeable future as requisite to career entry for IS graduates.

Teaching Coding Skills Costs IS Twice

Supporting software development coursework is doubly expensive for IS programs:

a) Consuming precious credit hours squeezed into business school programs dealing with the pressures for maintaining breadth in liberal arts within the strictures of business program accreditation; and,

b) the complexities of software development instruction that levies on faculty a burden of technical preparation and individualized student engagement that are not easily aligned with the models and areas of research promoted as flagship academic scholarship.

Search for a Win-Win Situation
Exploring ways to coopt bootcamps that teach coding skills may be mutually beneficial if they can: a) provide superior coding skill outcomes for students compared to the limited curricular resource for it in college and university programs, and b) lifting the training burden from IS faculty struggling to maintain a successful balance of teaching and research, both of which are grounded not in the computing but rather, the business disciplines where the primary standards of faculty evaluation reside. Some possible approaches are outsourcing software development skills training by accepting bootcamp completion for college credit as liberal arts coursework or as fulfilling some other distribution requirement, or insourcing the training as a summer intensive offering by the college. The latter might use underutilized housing and laboratory facilities and be staffed by a combination of practicing professionals, accomplished upper class students, and supervisory staff.

Teaming Up to Address the Skills Gap
Articulation agreements between bootcamps and IS programs can function as bilateral recruiting functions. Bootcamps can recommend IS programs for degree completion once they reach transition points in their development careers. And, colleges can recommend bootcamps as "test drives" for undecided students unsure of the two-year or four-year commitment to college. In either case, local businesses strapped by a shortage of programmers and app developers may want to explore internship, scholarship and mentorship arrangements to access the best and brightest prospects. These businesses may want to influence the bootcamp curricula regarding tools and skills appropriate for their information technology strategies, as well as, opportunities to upgrade or retrain the skill sets of their current employees.

5. DISCUSSION

Our exploration of coding bootcamps is not intended to malign or endorse the phenomenon, but rather to consider the challenges and opportunities. To summarize, we conclude with a simple SWOT analysis.

Strengths. We have elucidated the strengths of the code bootcamps as being very focused on specific technologies which are immediately valued and favored in the marketplace. Often located in population and technology hubs, the camp-to-employer food chain is compelling for the employer. These are fresh students who are ready to go with the timely skills required at an entry level. With career-switchers, employers get some of the polish and seasoning of work experience, which is generally favored in most industries evidenced in a lower unemployment rate for those with experience in almost any industry (Jepsen et al., 2014).

Weaknesses. Relative to the long-standing inertia of experience that traditional college-oriented programs and curricula in computing enjoy, there will likely be a wide range of providers and standards (or lack thereof) as the code bootcamp innovation diffuses and competition among providers increases. With no oversight, these bootcamps already deliver up mixed results with little recourse for students that feel short-changed. US Department of Education actions sanctioning ITT Technical University for fraudulent practices may be a cautionary tale here as we have witnessed some drawbacks in for-profit higher education (Morey, 2004).

Opportunities. As we have indicated earlier, two-year, four-year, and graduate institutions have the longstanding expertise in providing effective instructional environments. While many of these coding bootcamps are fitting in where they can, including dedicated commercial office spaces, institutions of higher education remain nexus points where a crossroads of research, instruction, technology, employers, and students can comingle. Rather than remaining averse to technology-wrought emerging models for instruction and learning (Hanna 1998; Hamilton 2016), institutions of higher education may do well to integrate this mode of delivery to realize its advantages and capitalize on the industry connections inherent in the code bootcamps. Often, higher education institutions are responsible for relationship building, an experientially-rich learning environment, and the maturation of students – particularly those of traditional college age. Regardless of what professional stage the code camper is, higher education can embrace elements and aspects of how programming, and other IT work, is increasingly seen as the "next blue-collar job" (Thompson, 2017). This is against a backdrop of futurist, near-desperate vision regarding a lack of employment opportunities in the face of automation, machine-learning, artificial-intelligence, and robotics (Clark, Graham, & Jones, 2017).

Threats. Perhaps of most interest to the IS academic would be appropriate questions about how/whether coding bootcamps will disrupt the market share that IS programs hold. In the "dot bomb" era, many sought out certifications and degrees from two-year and four-year institutions and any other means to get on the bandwagon of a super-heated bubble (Yourdon, 2002). Much has changed since that era. One change is the cocktail of outsourcing, offshoring, near-shoring, and on-shoring that pervades the labor market in software and systems development (Worley, 2012). Another is the advent of MOOCs and open education (Yuan & Powell, 2013). Further, the continued advances of service-dominant logics, Web 2.0, the Internet of Things, Social Media and Video Sharing create a new mix of information and learning vectors. Consider an event held in May of 2017 in Prague, Czech Republic, a country known for providing talent in business process and technology outsourcing, called Jobs Dev 2017 (Layman, Williams, Damian, & Bures, 2006; https://www.jobsdev.cz/). As an intersection to "facilitate developer-to-developer dialogue and offer a place where companies from a wide range of IT industries can meet with skilled programmers, freelancers, developers, and university graduates," this may represent an emerging trend where entry-level, mid-level, and senior-level talent can meet directly with employers. While any decent university job-fair would create the same facilitative environment, what if these meetings create a reality where the university is the unnecessary "middleman?" Increasingly, Codecademy, CodeHS, Coursera, Khan Academy, Lynda.com, and Udacity, among others, each can provide effective and focused instruction in the entry-level skills that get jobs, jobs that graduates of information systems programs are also vying for.

What if these new outlets will do a better job of teaching hands-on skills? How might we join, coopt or lead in this new environment? In fact, are we even now being left behind? Some information systems education researchers already seem to think so (Burns, Gao, Sherman, Vengerov, & Klein, 2014; Janicki et al. 2014). This paper invites continued inquiry and discussion regarding the coding bootcamp phenomenon.

6. REFERENCES

AACU. (2014). Association of American Colleges and Universities, retrieved July 7, 2017, www.aacu.org/leap/what-is-a-liberal-education#survey

Barnett, B. G., Basom, M. R., Yerkes, D. M., & Norris, C. J. (2000). Cohorts in educational leadership programs: Benefits, difficulties, and the potential for developing school leaders. *Educational Administration Quarterly*, 36(2), 255-282.

Burns, T. J., Gao, Y., Sherman, C., Vengerov, A., & Klein, S. (2014). Investigating a 21st century paradox: As the demand for technology jobs increases why are fewer students majoring in information systems? *Information Systems Education Journal*, 12(4), 4.

Carnevale, A. P., Cheah, B., & Strohl, J. (2013). Hard times: College majors, unemployment and earnings: Not all college degrees are created equal, retrieved July 7, 2017, cew.georgetown.edu/wp-content/uploads/HardTimes2015-Report.pdf

Clark, J. W., Graham, C. M., & Jones, N. (2017 August). Information systems and the problem of work: Protocol for a systematic review. Proceedings of Americas Conference on Information Systems (AMCIS 2017), Boston, MA.

Claxton, G. (1997). Knowing without knowing why, *The Psychologist*, May 1998, 217-220.

Eaton, J. S. (2000). Core Academic Values, Quality, and Regional Accreditation: The Challenge of Distance Learning. Council for Higher Education Accreditation, Washington, DC.

Eaton, J. S. (2012). The future of accreditation. *Planning for Higher Education*, 40(3), 8.

Geron, T., (2013, February 26) "Bill Gates, Mark Zuckerberg, Chris Bosh campaign for more programmers," Forbes, retrieved July 7, 2017, www.forbes.com/sites/tomiogeron/2013/02/26/bill-gates-celebrities-support-education-for-computer-programming/#5b21d60f7ff8

Gray, D. M., Smart, K. L., & Bennett, M. M. (2017). Examining espoused and enacted values in AACSB assurance of learning. *Journal of Education for Business*, 1-7.

Hamilton, E. (2016). Technology and the Politics of University Reform: The Social Shaping of Online Education. Palgrave Macmillan, New York, NY.

Hanna, D. E. (1998). Higher education in an era of digital competition: Emerging organizational models. *Journal of Asynchronous Learning Networks*, 2(1), 66-95.

Janicki, T., Cummings, J., & Kline, D. M. (2014). Information technology job skill needs and implications for information technology course content. *Information Systems Education Journal*, 12(6), 59.

Jepsen, C., Troske, K., & Coomes, P. (2014). The Labor-market returns to community college degrees, diplomas, and certificates. *Journal of Labor Economics*, 32(1), 95-121.

Layman, L., Williams, L., Damian, D., & Bures, H. (2006). Essential communication practices for Extreme Programming in a global software development team. *Information and Software Technology*, 48(9), 781-794.

Liu, K. (2000). *Semiotics in Information Systems Engineering*, Cambridge University Press, Cambridge, U.K.

Longenecker, H. E., Feinstein, D. L., & Babb, J. S. (2013). Is there a need for a Computer Information Systems model curriculum? In Proceedings of the Information Systems Educators Conference, ISSN: 2167, 1435-1447.

Morey, A. I. (2004). Globalization and the emergence of for-profit higher education. *Higher Education*, 48(1), 131-150.

NCES. (2017). National Center for Education Statistics, retrieved July 12, 2017, nces.ed.gov/globallocator

Reynolds, J., Adams, R., Ferguson, R., & Leidig, P. (2017). Programming in the IS Curriculum: Are requirements changing for the right reason? *Information Systems Education Journal*, 15(1), 80.

Shackelford, R., McGettrick, A., Sloan, R., Topi, H., Davies, G., Kamali, R., Cross, J., Impagliazzo, J., LeBlanc, R., & Lunt, B. (2006). Computing Curricula 2005: The overview report. *ACM SIGCSE Bulletin*, 38(1), 456-457.

Solomon, N. A., Scherer, R. F., Oliveti, J. J., Mochel, L., & Bryant, M. (2017). The perfect match: Factors that characterize the AACSB International initial accreditation host school and mentor relationship. *Journal of Education for Business*, 92(3), 114-120.

Stamper, R. K. (1973). *Information in Business and Administrative Systems*. John Wiley and Sons, New York, NY.

Stamper, R. K., Althous, K., & Backhouse, J. (1988). MEASUR, Method for Eliciting, Analyzing, and Specifying User requirements. In Olle, T. W., Verrijn-Stuart, A. A. & Bhabuts, L., (eds.), *Computerized Assistance During the Information Life Cycle.* Elsevier, Amsterdam, The Netherlands.

SwitchUp.org. (2017a), retrieved July 7, 2017, www.switchup.org/... /are-coding-bootcamps-worth-it-job-placement-market

SwitchUp.org. (2017b), retrieved July 8, 2017, www.switchup.org/research/best-coding-bootcamps

Thompson, C. (2017, February). The next big blue-collar job is coding. *Wired Magazine*, 24(12).

Topi, H., Valacich, J., Wright, R. T., Kaiser, K. M., Nunamaker, J. F., Sipior, J. C., & Vreede, G. J. (2010). IS 2010 curriculum guidelines for undergraduate degree programs in Information Systems, Association for Computing Machinery (ACM), Association for Information Systems (AIS), retrieved July 7, 2017, www.acm.org/education/curricula/IS%2020 10%20ACM%20final.pdf

USDoE. (2008). U.S. Department of Education, International Affairs Office, Structure of the U.S. education system: Credit systems, retrieved July 6, 2017, www2.ed.gov/about/offices/list/ous/internat ional/usnei/us/credits.doc

Waguespack, L. (2011). Design, the "straw" missing from the "bricks" of IS curricula. *Information Systems Education Journal*, 9(2), 101.

West, D., Rostal, P., & Gabriel, R. P. (2005 October), "Apprenticeship agility in academia." In Companion to 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications, 371-374.

Wikipedia. (2017a) “Close Order Drill”, retrieved 7/6/2017, en.wikipedia.org/wiki/Military_parade

Wikipedia. (2017b) Coding Bootcamps, retrieved 7/7/2017, en.wikipedia.org/wiki/Coding_bootcamp

Worley, L. (2012). Outsourcing, Offshoring, Nearshoring, Onshoring – What's Going On? *Legal Information Management*, 12(1), 9-11.

Yourdon, E. (2002, February). Preparing software engineers for the 'Real World'. Proceedings of 15th Conference on Software Engineering Education and Training, 2002, CSEE&T, p. 3.

Yuan, L. & Powell, S. (2013). *MOOCSs and Open Education: Implications for Higher Education*, Centre for education technology and interoperability standards, retrieved 7 July 2017, publications.cetis.org.uk/wp-content/uploads/2013/03/MOOCs-and-Open-Education.pdf