Editors: Jelena Mirkovic, mirkovic@isi.edu | Bill Newhouse, newhouse@nist.gov

Project-Based Learning Inspires Cybersecurity Students: A Scholarship-for-Service Research Study

Alan T. Sherman | University of Maryland, Baltimore County
Peter A.H. Peterson | University of Minnesota Duluth
Enis Golaszewski, Edward LaFemina, and Ethan Goldschen | University of Maryland, Baltimore County
Mohammed Khan | Prince George's Community College
Lauren Mundy | Montgomery College
Mykah Rather | Prince George's Community College
Bryan Solis | Montgomery College
Wubnyonga Tete | Prince George's Community College
Edwin Valdez | Montgomery College
Brian Weber and Damian Doyle | University of Maryland, Baltimore County
Casey O'Brien | Prince George's Community College
Linda Oliva | University of Maryland, Baltimore County
Joseph Roundy | Montgomery College
Jack Suess | University of Maryland, Baltimore County

Over four summer days in 2017, cybersecurity students at the University of Maryland, Baltimore County (UMBC) analyzed the security of a targeted portion of the UMBC campus network, discovering numerous flaws, creating proof-of-concept exploits, and providing practical recommendations for mitigation. We report on this novel summer research study; its technical findings; and takeaways for students, educators, and Information Technology Departments.

UMBC, a National Center of Academic Excellence in Cyberdefense Education and Research, is a midsize public university offering undergraduate and graduate tracks in cybersecurity leading to B.S., M.S., and Ph.D. degrees in computer science, computer engineering, and information systems and the master of professional studies degree in cybersecurity. UMBC is also a Cybercorps: Scholarship for Service (SFS) school, where students are supported for up to three years on the condition that, after graduation, they will work for federal, state, local, or tribal governments one year for each year of support.

In the fall of 2016, with support from the National Science Foundation, UMBC was one of 10 schools that pioneered a new strategy for recruiting talented cybersecurity professionals for government service: the university extended SFS scholarships to nearby partnering community colleges (CCs). To integrate the new CC students into the existing SFS cohort through a collaborative activity, Alan T. Sherman, UMBC professor and director of UMBC's Center for Information Security and Assurance (and one of the authors of this article), organized a four-day SFS summer research study at UMBC in the summer of 2017. Prof. Sherman also invited professors, researchers, UMBC graduate students, and National Security Agency (NSA) personnel to interact with the students as technical experts.

Everyone worked as a team on the same challenge: to analyze the network administration system's (NetAdmin's) web front end enabling modifications to the UMBC campus firewall. In support of the project, UMBC's Division of IT (DoIT) provided participants with all relevant source code and a functional copy of the environment for testing. At the end of each day, DoIT staff, including the primary NetAdmin author, met with the students. At the conclusion of the project, the student team identified several critical

Digital Object Identifier 10.1109/MSEC.2019.2900595
Date of publication: 14 May 2019

 Copublished by the IEEE Computer and Reliability Societies

vulnerabilities, devised exploits, and presented their findings and recommendations to DoIT.

This type of activity should be beneficial for any group of students. Our hope is that educators, IT Departments, and students at any institution may learn from our shared experiences in collaborative and real-world project-based learning (PBL) (see "Project-Based Learning"). Partnering with a real IT Department has many benefits: the study inspired students and enhanced students' skills, students and educators appreciated the authentic case study, DoIT received free security consulting, and the UMBC community gained improved security.[5]

The SFS Summer Study at UMBC

A hands-on study was appealing because it enabled collaboration, problem solving, and independent thinking in addressing an important, practical, rich, and challenging problem. We sought a problem that was complex but tractable. We also sought a project that, if successful, would benefit the UMBC community. Focusing on UMBC's home-grown NetAdmin had many attractive properties: NetAdmin's source code was available; DoIT could answer questions and provide information; and, since NetAdmin had never undergone a security evaluation, it seemed likely to have vulnerabilities.

Our task was to analyze the security of NetAdmin and the network architecture and to make recommendations to DoIT.

The in-person participants comprised six CC transfer students, three UMBC undergraduates, and one Ph.D. student. All students had at least a basic grounding in cybersecurity. Some students had much more expertise. Each participant signed a nondisclosure agreement (NDA) with DoIT.

The study took place from 9 a.m. to 5 p.m., Tuesday through Friday, in a large room with tables, a whiteboard, and a projector. Using a PBL approach,[2] we presented the challenge and challenge-related goals to the students and instructed them to formulate a strategy that would achieve the project's goals, while supporting sustained inquiry and reflection. Students organized themselves into teams, with each team exploring some aspect of the problem. For example, teams explored the network topology, the software environment, architectural issues, source code, and known software vulnerabilities. More experienced students emerged as leaders.

Project-Based Learning

Project-based learning (PBL) is an instructional approach in which small groups of students engage in authentic tasks and learn as they attempt to solve relevant problems. Students ask and revise questions, debate ideas, generate predictions, experiment, collect data, draw conclusions, communicate ideas and findings, refine approaches, and create products.[2]

PBL holds great promise in cybersecurity because there is a proliferation of complex challenges to engage students, sustain their interest, and direct their learning as they develop diverse approaches to solving real-world problems. In PBL, students are focused on tasks; they can try out a variety of solutions and receive timely feedback on their approaches. They engage in collaboration and reflection that deepens their learning and enhances the transferability of skills.

There are many examples of PBL in cybersecurity (e.g., the New Jersey Institute of Technology's Cyber-Real World Connections Summer Camp[S1] and Conklin and White's graduate course,[S2] which includes some elements similar to our study). We encourage the creation of more scholarly articles on this subject. We are strong believers in the value of PBL, as evidenced by our participation in the INSuRE Project.[S3]

References

S1. New Jersey Institute of Technology, "Cybersecurity Real World Connections Summer Boot Camp at NJIT," 2016. [Online]. Available: https://sci.njit.edu/gencyber/RWCCybersecurityCamp Brochure-Summer2016.pdf

S2. A. Conklin and G. White. "A graduate level assessment course: A model for safe vulnerability assessment," in *Proc. 9th Colloquium for Information Systems Security Education (CISSE)*, June 2005, pp. 109–114.

S3. A. T. Sherman et al., "The INSuRE Project: CAE-Rs collaborate to engage students in cybersecurity research," *IEEE Security Privacy*, vol. 15, no. 4, pp. 72–78, July–Aug. 2017.

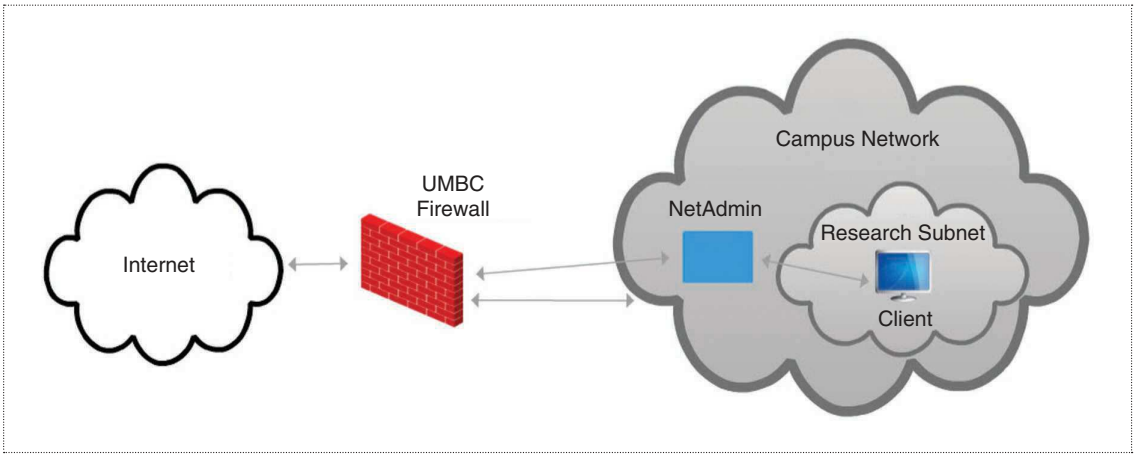

Figure 1. An illustration showing the architecture of the UMBC network, including the NetAdmin tool, which is accessible to machines on the research subnet.

Two UMBC professors and two NSA experts visited each day to answer technical questions. Late each afternoon, representatives from DoIT, including the primary NetAdmin script author, joined the group for a discussion. Students unable to attend in person joined a student-led one-hour evening chat session via Google Hangouts.

The Problem

The UMBC network has 10,000 users; more than 15,000 devices connect to the network daily. That makes defending the UMBC network a daunting challenge. One part of the defense is a firewall between the Internet and the UMBC network. All campus traffic must pass through this firewall. One of UMBC's internal subnets is for computers used in research projects. Users on these computers often need to connect to and from the Internet on various ports. This requires permission to enable data to pass through the firewall. DoIT originally processed firewall exceptions manually, which was time-consuming and error prone. NetAdmin, launched in 2006, facilitates exceptions to UMBC's default-deny firewall policy. Our task was to analyze the security of NetAdmin and the network architecture and to make recommendations to DoIT.

NetAdmin allows faculty and staff who are authenticated through the myUMBC single sign-on (SSO) system to create firewall exceptions for their machines on the research subnet. As shown in Figure 1, NetAdmin sits behind the UMBC firewall, so it can be accessed only from the campus network or by virtual private network (VPN) users.

The adversary's main goal was to make unauthorized changes to the UMBC firewall without detection.

User groups, including faculty, staff, and superusers, are defined in a file in NetAdmin's application directory. Superusers may view, modify, or create any rule for any Internet Protocol address on the UMBC network (not only on the research subnet). Faculty and staff may create, modify, or delete rules for certain common ports [e.g., Secure Shell (22), HTTP (80)] associated with research subnet addresses they "own." Rules violating these restrictions must be submitted out of band to DoIT for special consideration. Since machine owners could modify only rules affecting their own machines, DoIT reasoned that NetAdmin introduced little risk.

Written in PHP 5.1.6 and residing on a dedicated Linux server running Apache 2.2.3, NetAdmin receives firewall rules from client browsers and applies those rules to UMBC's firewall through application programming interface (API) calls. To authenticate the rules to the firewall, NetAdmin includes a 360-bit symmetric API key file stored in the application directory of the NetAdmin server. This file is neither digitally signed nor integrity protected. In case of failures and restarts, NetAdmin stores rules and logs in local unstructured files. Each rule is described by one record, which is delimited by a newline. Pipe characters delimit fields.

For more than a decade, NetAdmin ran untouched and worked well, with no detected compromises. No one, however, had ever subjected NetAdmin to a thorough security evaluation. In planning discussions, DoIT suggested analyzing

NetAdmin in the same way that a penetration testing team might. Students were encouraged to follow whatever approach they thought best and were given access to DoIT staff, who provided appropriate information as requested.

Our adversarial model was an outsider with compromised faculty or staff credentials or a malicious faculty or staff insider on the research subnet with the knowledge, skills, and resources of an excellent computer science graduate student. The adversary's main goal was to make unauthorized changes to the UMBC firewall without detection. The group analyzed NetAdmin in its operational context, including whether cryptography was being properly used, but did not consider attacks on the cryptography itself, the servers' physical security, social engineering of DoIT staff, or recovery after disaster or compromise.

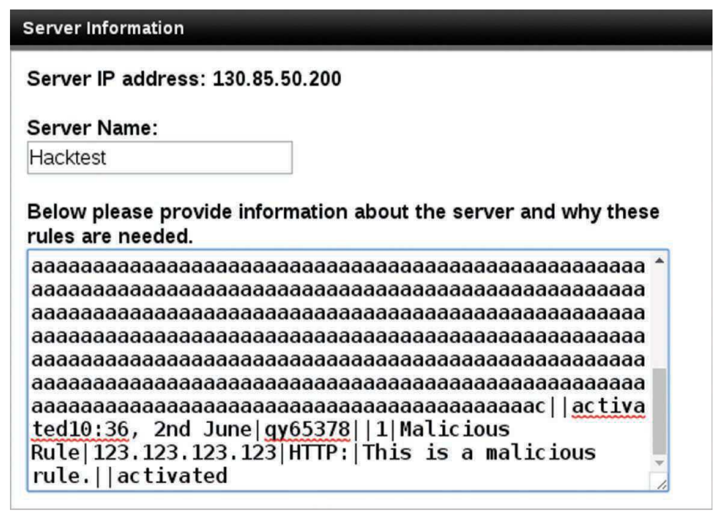

Figure 2. A screenshot of the NetAdmin web interface with record overflow.

Vulnerabilities, Attacks, and Risks

At the start of our four-day study, the student-led team of 10 individuals focused on identifying risks, potential vulnerabilities, and related attacks, many of which were extremely serious. NetAdmin ran on an unpatched, out-of-date, and unsupported operating system (OS), Linux 2.6.18, which has at least 463 vulnerabilities (https://www.cvedetails.com). Violating the principle of least privilege,[1] the firewall API key used by NetAdmin permitted arbitrary changes to the campus firewall (not just to the research subnet). Compromise of the NetAdmin server would therefore be very severe. An attacker could issue arbitrary firewall rules affecting the entire campus; modify log files, rules, and user groups; and exfiltrate the firewall API key, all of which are stored as unencrypted text without integrity protection.

While DoIT was not aware of any attack involving NetAdmin, the potential attacks listed were feasible and could be executed by skilled students.

Students found some of the most common software security errors.[3] NetAdmin did not adequately validate or sanitize inputs. For example, NetAdmin permitted firewall rules to include text descriptions but did not strip HTML or JavaScript. This made it possible for someone to conduct code injection attacks,[4] which could victimize users and administrators through their browsers. JavaScript payloads could submit rules to NetAdmin in the background. The malicious code could execute arbitrary commands on the NetAdmin server. The malicious code could, for example, initiate commands to exfiltrate the firewall API key.

Similarly, NetAdmin did not validate the length of rule descriptions, which meant that the system could be vulnerable to possible record-overflow attacks and/or denial-of-service attacks. In particular, NetAdmin's use of the PHP command **fgetcsv**() assumed (without verifying) that each record was at most 999 bytes. As shown in Figure 2, if a user (or adversary) entered a rule longer than 999 bytes, the additional bytes would be accepted as a new and valid record.

Communication between users and NetAdmin was unencrypted HTTP without integrity protection, allowing an adversary to read and modify all traffic. By modifying data sent to NetAdmin, an adversary could set firewall rules enabling unauthorized access to the user's machines or launch an injection or record-overflow attack. Also, while NetAdmin authenticated the firewall using a self-signed certificate, the firewall did not authenticate NetAdmin; it required only that requests contain the API key. Additionally, since the firewall's key was self-signed, compromise of

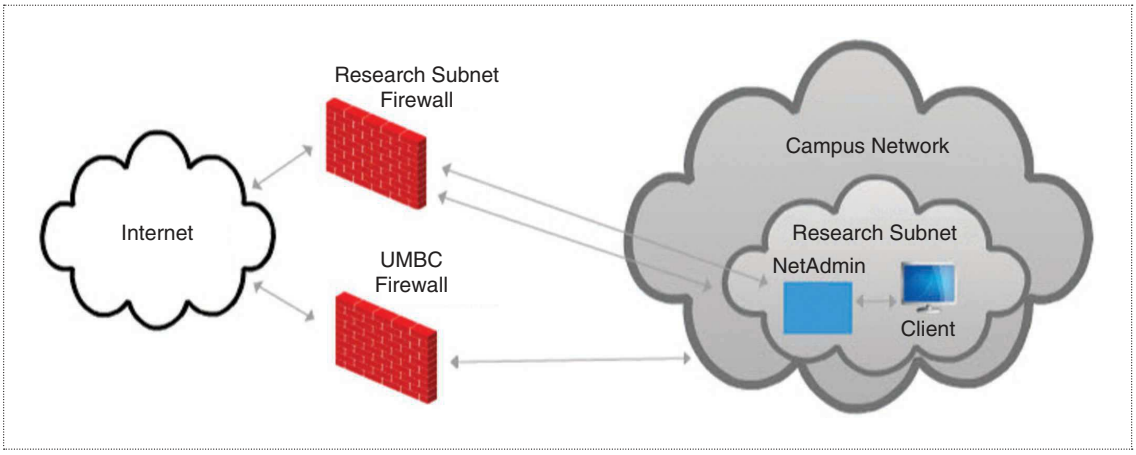

Figure 3. An illustration showing the recommended architecture to provide compartmentalized defense. This design restricts failure of the research subnet firewall to the research subnet.

UMBC's signing key could enable an adversary to forge certificates and impersonate the firewall.

Other risks were exposed. For example, UMBC's one-firewall design provided no architectural protection. NetAdmin was accessible via the campus VPN, facilitating remote attacks. If an adversary could hijack a user's SSO session, that adversary could masquerade as that user to NetAdmin.

While DoIT was not aware of any attack involving NetAdmin, the potential attacks listed were feasible and could be executed by skilled students. As proof of concept, students implemented record-overflow and injection attacks.[5]

Recommendations

After identifying attacks, the students recommended a number of mitigations: the NetAdmin software, including the OS and all supporting software, should be kept current with security patches to mitigate off-the-shelf exploits; all input should be sanitized and validated on the server side; HTML, Javascript, and special characters (e.g., pipe) should be prohibited in rules; and size limits should be enforced to stop overflow attacks.

Also, NetAdmin should use different API keys for superusers and faculty, with the latter affecting only the research subnet. API key establishment and storage might be improved by encrypting the API keys and keeping digests for integrity checking. The digests could be kept offline for periodic manual integrity checks, but the plaintext API keys are actively needed by the server during operation; keeping the encrypted API keys and digests locally would have limited value given that there is no secure place on the NetAdmin server to store them. As mentioned, compromise of the NetAdmin server would be catastrophic; in this case, the keys would be revealed. There is no perfect solution for the key-storage issue.

Figure 3 shows a two-firewall approach with better segmentation, where the research subnet firewall and the main campus firewall use separate keys. Regardless, communications between the NetAdmin server and users should use end-to-end encryption with authentication and integrity protection, and the firewall and NetAdmin should authenticate each other using certificates signed by a certificate authority.

Using a direct, physical connection between NetAdmin and the proposed research subnet firewall would improve physical security. Segmenting NetAdmin into a web front end, for validating and sanitizing input, and a back end, for performing additional validation and for communicating with the firewall, would add defense in depth. These services should run under separate accounts and be restricted in other ways (e.g., no unnecessary software or communication with unnecessary hosts). Disallowing connections from the campus VPN would reduce the potential for remote attacks, though it would be difficult to prevent an adversary from logging into NetAdmin after establishing a VPN connection to another campus machine. Performing periodic internal and external audits of NetAdmin's software and firewall rules would help sustain security.

Takeaways

We hope that educators, IT departments, and cybersecurity program managers can benefit from our experience.

Educators and Study Organizers

Overall, the study went very smoothly, and PBL sustained inquiry and critical thinking. Most students quickly became absorbed in

the project and were productive, although some students could have benefited from some prior preparation. Engagement level varied, but everyone made contributions. A few students were somewhat uncomfortable with the undirected and open-ended model. However, in a follow-up survey, 100% of participants reported that the project increased their cybersecurity knowledge and skills (86% strongly agreed and 14% agreed). Participants identified the following elements as valuable: teamwork, hands-on nature of the task, real-world challenge, critical thinking, and problem solving. All participants reported that they would recommend the summer study project to other cybersecurity students.

Having a virtual copy of the production system for experimentation was extremely valuable as was having access to the original developer. Posing questions to DoIT in a Google Doc and receiving answers throughout the day was effective and helpful as was having local security experts available for consultation. In-person discussions were facilitated by a video projector, whiteboard, and students' personal devices. We recommend having numerous power strips available. Evening chat sessions allowed remote students to participate. Chat worked better than video because it provided a written record and facilitated asynchronous use. Summer internships can create scheduling conflicts; we now hold the study during the January intersession.

IT Departments

IT departments often run obsolete and unpatched systems because they know that updates will take valuable staff time and might break the system, requiring even more staff time to fix. Our study, however, demonstrates that keeping software systems up to date is not optional. We also exposed and exploited numerous common vulnerabilities and suggested improvements. IT departments elsewhere could benefit from similar analysis.

We were fortunate to enjoy remarkably strong support and cooperation from DoIT, and we commend members of the department for their constructive attitude. Teams at other schools, however, might face a defensive administration that fears embarrassment or is unwilling to trust students. We believe that careful selection of participants and the use of NDAs should reassure administrators that students in the project can be trusted. Our hope is that, by welcoming and encouraging analysis of their systems, other IT departments and student teams can learn while enhancing the security of their communities.

Cybersecurity Program Managers

Extending scholarships to CC students has thus far has worked well. In recruiting CC students for our SFS program, we focus primarily on those pursuing associate degrees because they are more prepared to transfer to four-year schools, even though some associate of applied science programs include more cybersecurity coursework. While there is an opportunity cost in that a scholarship awarded to a CC student is not awarded to a student at UMBC, we are attracting highly qualified CC students, and the scholarship is a life-changing opportunity for some students, especially those from modest backgrounds. Our current approach is to support two CC graduates per year.

Our study engaged and motivated students, as evidenced by their findings and our survey results. We also demonstrated that there are highly capable students at CCs who can contribute to cybersecurity. While we integrated this study into the SFS program at UMBC, we feel this type of activity could be integrated into nearly any kind of cybersecurity program. Partnering qualified students with IT Departments can reap benefits for everyone: students gain exciting, concrete, hands-on collaborative experiences; educators are given rich and realistic case studies supporting project-based learning; and IT Departments receive free cybersecurity consultations. DoIT hired several of the participants to join its security team. We look forward to conducting similar studies each year and hope that other schools can also benefit from similar collaborations. ■

References

1. M. Bishop, *Computer Security: Art and Science*. Boston: Addison-Wesley, 2003.
2. P. C. Blumenfeld, E. Soloway, R. W. Marx, J. S. Krajcik, M. Guzdial, and A. Palincsar, "Motivating project-based learning: Sustaining the doing, supporting the learning," *Educ. Psychol.*, vol. 26, pp. 369–398, 1991.
3. S. Kaza, B. Taylor, and E. K. Hawthorne, "Introducing secure coding in CS0, CS1, and CS2: Conference workshop," *J. Computing Sci. Colleges*, vol. 3, pp. 11–12, June 2015.
4. Open Web Application Security Project, "The OWASP Foundation: The free and open software security community." [Online]. Accessed on: Sept. 16, 2018. Available: https://www.owasp.org/index.php/Main_Page.
5. A. T. Sherman et al., The SFS Summer Research Study at UMBC: Project-based learning inspires cybersecurity students, *Cryptologia*, to be published, Nov. 2018. [Online]. Available: arXiv:1811.04794.

Alan T. Sherman is with the University of Maryland, Baltimore County. Contact him at sherman@umbc.edu.

Peter A.H. Peterson is with the University of Minnesota Duluth. Contact him at pahp@d.umn.edu.

Enis Golaszewski is with the University of Maryland, Baltimore County. Contact him at golaszewski@umbc.edu.

Edward LaFemina is with the University of Maryland, Baltimore County. Contact him at edlafem1@umbc.edu.

Ethan Goldschen is with the University of Maryland, Baltimore County. Contact him at egold2@umbc.edu.

Mohammed Khan is with Prince George's Community College. Contact him at khanmoh1@umbc.edu.

Lauren Mundy is with Montgomery College. Contact her at lmundy1@umbc.edu.

Mykah Rather is with Prince George's Community College. Contact her at mrather1@umbc.edu.

Bryan Solis is with Montgomery College. Contact him at bsolis1@umbc.edu.

Wubnyonga Tete is with Prince George's Community College. Contact her at wtete1@umbc.edu.

Edwin Valdez is with Montgomery College. Contact him at evaldez2@umbc.edu.

Brian Weber is with the University of Maryland, Baltimore County. Contact him at brianw5@umbc.edu.

Damian Doyle is with the University of Maryland, Baltimore County. Contact him at damian@umbc.edu.

Casey O'Brien is with Prince George's Community College. Contact him at cobrien@nationalcyberwatch.org.

Linda Oliva is with the University of Maryland, Baltimore County. Contact her at oliva@umbc.edu.

Joseph Roundy is with Montgomery College. Contact him at Joseph.Roundy@montgomerycollege.edu.

Jack Suess is with the University of Maryland, Baltimore County. Contact him at jack@umbc.edu.

Digital Object Identifier 10.1109/MSEC.2019.2911826