From Midshipmen to Cyber Pros: Training Minority Naval Reserve Officer Training Corp Students for Cybersecurity

Naja A. Mack
University of Florida
Gainesville, FL
najamac@ufl.edu

Kevin Womack
Morehouse College
Atlanta, GA
kevin.womack@morehouse.edu

Earl W. Huff, Jr.
Clemson University
Clemson, SC
earlh@clemson.edu

Robert Cummings
Morehouse College
Atlanta, GA
robert.cummings@morehouse.edu

Negus Dowling
Morehouse College
Atlanta, GA
negus.dowling@morehouse.edu

Kinnis Gosha
Morehouse College
Atlanta, GA
kinnis.gosha@morehouse.edu

ABSTRACT

Computing is not only a fundamental part of our daily lives — our national security depends on it. It is essential that members of the Reserve Officer Training Corps (ROTC) be well-versed in computing techniques designed to combat cyber-attacks that continually improve in sophistication and frequency levels. Through computer science education, prospective military leaders are being taught the critical and analytical thinking skills necessary to address these threats. This paper outlines and discusses how underrepresented minority Naval Reserve Officer Training Corps (NROTC) midshipmen, with the majority having little-to-no programming skills or background, were taught the foundational aspects of computer science and exposed to specific cybersecurity concepts in a 15-week program. The program produced many insightful outcomes, one being the development of a tool that collected and analyzed publicly available social media data through the application of data mining algorithms and techniques. Additionally, interviews, which were conducted upon the completion of the program, documented notable findings on the program's effectiveness and on key concepts for teaching computer science to minority beginners.

CCS CONCEPTS

• **Social and professional topics → Computer science education**; • **Applied computing → Military**;

KEYWORDS

cybersecurity, programming, data mining, military, rotc

ACM Reference Format:
Naja A. Mack, Kevin Womack, Earl W. Huff, Jr., Robert Cummings, Negus Dowling, and Kinnis Gosha. 2019. From Midshipmen to Cyber Pros: Training Minority Naval Reserve Officer Training Corp Students for Cybersecurity. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19), February 27-March 2, 2019, Minneapolis, MN, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3287324.3287500

SIGCSE '19, February 27-March 2, 2019, Minneapolis, MN, USA

ACM ISBN 978-1-4503-5890-3/19/02…$15.00
https://doi.org/10.1145/3287324.3287500

1 INTRODUCTION

As people's lives become increasingly immersed in the digital realm, it is necessary for developers and consumers to acquire knowledge in security to protect against intrusion and exploitation [2]. A recent report on cyber-related threats [3] showed an increase in remote entry attempts and dangerous user behavior, thus creating opportunities for attackers to exploit vulnerable systems. Primarily, the government, financial, professional and information sectors receive the highest volume and variety of malicious activity [3]. Cybersecurity represents the body of technologies, processes, and practices used to protect networks, devices, elections, programs, and data from a variety of attacks and unauthorized access [9]. With technology becoming a more significant part of our lives, cybersecurity will continue to be an integral aspect of personal and professional careers. More important than ever before, the basic foundations of cybersecurity should be instilled in job trainings, educational programs, and even in academic majors.

This paper presents findings from a 15-week program in which members of the Naval Reserve Officer Training Corps (NROTC) from three Historically Black Colleges and Universities (HBCUs), Morehouse College, Spelman College, and Clark Atlanta University, were taught core concepts in computer science and cybersecurity and tasked to develop a tool for threat and vulnerability analysis. Results from post-program interviews helped highlight program efficacy and offered suggestions

2 LITERATURE REVIEW

2.1 Computer Science Education for Non-Majors

The retention, and ultimate success, of students across the nation, has been an ongoing issue within the computer science community. The Theory of Constructivism, while seemingly encompassing most students, explains fallacies and learning deficits [1, 11]. It posits that students do not receive information as it is taught; instead, they fabricate their version of the information provided, based on prior knowledge or experience, an attempt to conceptualize processes and manifest them in the form of something familiar [1, 6]. Because computer science can be articulated in an unfamiliar manner, many students misunderstand the teachings and therefore struggle in, even basic introductory courses, which are typical requirements for 4-year degrees in post-secondary education [7].

At the heart of the problem is students' lack of interest in the subject itself, which is attributed to factors such as voluntary disengagement of the student, redundant communicative media programs, and perceived lack of relevant examples of computing in everyday life [8, 11]. A disengaged student will often revert to memorization, rather than investing the time and energy in understanding the material. As a result, critical thinking skills are not fostered, potentially leading to failure in these introductory courses. Secondly, the ability to relate what is learned to something familiar is crucial to retention. One study found that students learn better if they can relate the material at hand to something they enjoy or like; however, this can be difficult to do when learning computer science [8]. The question then becomes: How do we bridge the gap between programming and relevant real-world applications, when most introductory courses exclude examples or projects that illustrate how computer science can be applied to solve issues in today's society?

2.2 Computer Science, Cybersecurity and ROTC

The foundation of an understanding of cybersecurity is developed during the learning of fundamental programming concepts, like iteration and data structures. Many institutions of higher education have implemented labs or programming competitions to pique the interest of students in hopes to garner the interest of sharp, young, and impressionable minds [4]. Schools across the nation are using Computer Science to assess how vulnerable people are being hacked on email and social media platforms [5].

Colleges, universities, and service academies possess the tools necessary to produce cyber-literate leaders. Moreover, while there is a high demand for these leaders, few institutions provide the framework that properly combines computer science with other departments and programs to demonstrate how the field can be applied. Computer Science should be integrated into all programs in order to produce the most cyber-competent leaders. Because of the vital role that they play in the creation of future military leaders, the U.S. service academies have already begun to spearhead computer science synthesized programs. The Military Academy CYBER Education Working Group was established in 2012 with the goal of augmenting undergraduate cyber education at the military academies. The Working Group believes that cyber education should include knowledge that all officers, cyber leaders and all highly technical officers should know [10].

3 PROGRAM OVERVIEW

The program enabled NROTC midshipmen from three Historically Black Colleges and Universities (HBCUs) in Atlanta, GA, Morehouse College, Spelman College, and Clark Atlanta University, to contribute technically to cyber, and electronic warfare (EW) technology domains through a rigorous, product-oriented research-and-development (R&D) program. The ultimate deliverable of the program with the Navy ROTC midshipmen was to produce a functional beta version of a data mining tool that would stream tweets from a user account and perform informative analysis on the retrieved tweets. The cohort sessions were scheduled to take place over 15 weeks, with the first three weeks at the close of the Fall semester and 12 working weeks in the following Spring semester. Two undergraduate research assistants, both African-American males and a postdoctoral researcher, an African-American female, were appointed to the project to assist in supporting the midshipmen's learning. They were also responsible for maintaining the gradual progression of the software engineering behind the prototype. In order to facilitate learning while also aiming to complete the team project by the end of the program period, one-hour class sessions were held at the beginning of every week. Sessions were, often led by the undergraduate assistants and overseen by the postdoctoral researcher, with supplemental office hours available by appointment later in the week. With the limited experience of the program participants, classes covered Python basics such as:

(1) Basic Syntax - Students learned fundamental commands that store variables, print messages, and perform mathematical operations
(2) Loops - Students learned the structure of various loops and wrote their own to solve their own problems.
(3) Functions - Students learned how to create functions to reduce repeated code and increase usability.
(4) Lists - Students learned how to create and iterate through lists.

After learning python basics, the midshipmen learned how to carefully integrate Tweepy, an importable application programming interface (API) library that would be used for collecting and analyzing publicly available user information and tweets.

3.1 Participants

There were 18 total Navy ROTC midshipmen program participants (7 female and 11 male), 3 of whom were Computer Science majors (2 female and 1 male). About 40% of the total cohort were STEM majors at the time of enrollment in the program. In comparison to other cohorts within the same grant, the backgrounds and majors of this particular group of midshipmen were rather unconventional. At other institutions, cohorts were chosen to be purely STEM majors, or even solely computer science majors. The purpose in opting for diversity was the hope that students of different areas of study could bring varying viewpoints to the team project, which would ultimately make for a better, more user-friendly tool. Furthermore, selecting students of various majors would give project leaders the chance to introduce students of different majors to programming and computer science.

3.2 In-Class Activities

With tight deadlines on the team project, making sure that no midshipman was left behind in the content of the lectures was a top priority. To address this, the undergraduate assistants and their post-doctoral advisor developed strategic assignments that were to be completed during class meetings to check the understanding and comprehension of the midshipmen. The assignments ranged in difficulty from practicing simple arithmetic and assignment logic in the Python IDLE shell to complex function definitions and implementations, looping over complex data structures, and capturing user input to create simple applications to be run in the terminal.

At the initial cohort meetings, lessons were taught in a lecture style with simple coding exercises that would be done as a class, typically at the end of the slide deck of that day's session. For example, with the Python IDLE shell projected on the screen, one of the undergraduate assistants would ask the class what one should type to perform some assignment and arithmetic logic; together the class would tell the assistant what to type at the prompt. Later in the program, in-class assignments/activities and completeness checks were introduced, while pair-programming was phased out. Program participants were given assignments that were to be developed individually and submitted as a Python file. These Python files required the midshipmen to create more complex programs that included function definition, looping, and basic data structures, specifically lists and dictionaries.

3.3 Team Project

The team project given to the midshipmen was to create a data mining tweet analysis tool that would hypothetically provide a military team with enough relevant information to identify a user that could pose a potential threat to national security. The software application would be built entirely in Python, utilizing the Tweepy API library for extracting tweets and user data from Twitter and Tkinter to design an acceptable user interface. The software tool was collectively designed to perform four main functions:

(1) Create a basic user profile when given a user's Twitter handle;
(2) Create a customized/filtered user profile given a user's Twitter handle;
(3) Perform overall text analysis that would provide a word-cloud of the most frequently used words in the tweets of a particular profile given a user's Twitter handle and identify the five most frequently used words; and
(4) Perform overall sentiment analysis that would provide the percentage of positive, negative, and neutral tweets for a profile given a user's Twitter handle.

The midshipmen utilized the knowledge of Python basics and data structures from in-class activities and assignments in four distinct teams. Each team worked to deliver one of the four aforementioned basic requirements of the software to the best of their ability. The undergraduate assistants and the post-doctoral researcher were available for support for any issues that would arise; but ultimately, the midshipmen were actual owners of the project in its totality. Weekly sprints were utilized with team syncs at the beginning of each week to allow the program leaders to check in on the progress of the students in their respective tasks. Upon its completion in a month's time, the software tool was presented to program directors and leaders to secure further funding for continued learning and research.

3.4 Program Evaluation

Instructors were interviewed on the following areas to offer insight on the effectiveness of the program: the midshipmen's performance during the development of the tool, the tool's effectiveness to produce intended results, the tool's usability, and the tool's prospective user experience. Instructors, as well as 14 of the 18 midshipmen, were also interviewed with oepen and close ended questions on the learning experience, the program's strengths and weaknesses, and individual learning and teaching preferences. Results were analyzed through inductive qualitative content analysis.

4 RESULTS

4.1 Team Project

The instructors of the program were interviewed to assess the outcome of the developed tool. While developing the tool, the instructors agreed that the biggest challenge was getting the midshipmen to retain the Python programming concepts. Additionally, teamwork difficulties, including poor communication, lack of responsibility and accountability for individual tasks, and poor collaboration skills, interfered with the development of the tool. Nevertheless, the tool was completed to specification, and its usability was assessed. One instructor praised the success of the tool's demo. Another instructor believed the tool was usable at the minimum requirements and could use further aesthetic and GUI improvements and usability testing. The instructors agreed that the effectiveness of the tool matched its intention. However, there was still room for fine-tuning if the tool were to be regularly used. According to the instructors, potential users in cybersecurity would have high value in using the tool that the midshipmen built.

4.2 Midshipmen Interviews

The midshipmen were asked a series of yes/no questions. Three (22%) of the midshipmen experienced coding in a course prior to the program. Twelve (86%) midshipmen agreed to being more interested in computer science than in the beginning of the program, while 2 (14%) of them disagreed. Only one (7%) of the midshipmen believed it was not necessary to take a computer science or Python course prior to the program. All of midshipmen surveyed believed actual semester courses, apart from lab sessions, would improve their training. The majority of the midshipmen (86%) agreed that the lab was a good learning environment and that having a different environment to learn in was effective. Furthermore, the majority (86%) also preferred to figure out the homework on their own; further elaboration on this topic varied in rationale, including reasonings like being comfortable with problem-solving independently and time-consumption of walking through homework versus figuring it out independently. On the subject of out-of-class assignments, 92% of the midshipmen surveyed believed that the homework assignments were relevant, as their skills practiced through homework were applicable in later use in developing the data mining tool.

Midshipmen were also asked a series of additional open-ended questions on the topic of the varying instruction styles exercised in the program. The students ultimately preferred the teaching techniques of Powerpoints, learning supplemented with examples, working hands-on with assignments, and learning concepts prior to practicing them (as opposed to an examples-first approach). Although suggestions to improve teaching methods varied, many agreed with increasing the number of hands-on assignments. Furthermore, the results suggest that the midshipmen strongly liked computer science after completing the program and found it very necessary for the future. The midshipmen believed participation in the program enhanced their resumes and skillsets, aligning them more closely to serve in a cybersecurity field post-graduation. They generally had no problems with the time allotted to execute activities or assignments and their challenges in the program varied person-to-person; namely completing the homework, participating successfully in group assignments, and independent pacing proved to be some of the pain points, to name a few. Group learning preference varied as there was no significant skew in the survey results toward working alone or in a group. It is to be noted that the midshipmen would have liked to see the assignments be more concept-specific, an increased amount of homework for more opportunities for practice, and a better distribution of skill amongst group members in the team assignment.

4.3 Instructor Interviews

The instructors were asked a series of open-ended questions. The first questions were on teaching methods. Instructors varied between preferring instruction in the format of Powerpoints and interactive exercises. The instructors agreed instruction could have been improved by paying better attention to detail and by making the increase in difficulty gradual. Instructors also agreed the time spent in sessions were productive. Having two undergraduate assistant instructors was favored, particularly with the ability to be hands-on with the midshipmen. Having a lead instructor of the program was also favored to keep order and organization. Having consistent source material helped with lesson-planning as well as being equipped with the necessary tools; although a dedicated computer lab would've helped.

Instructors were also questioned on factors that differentiated and affected students directly. Having a course dedicated for such a program as an elective was determined to be beneficial. Prerequisite coding experience requirements were recommended depending on the task and rigor that the tool needs to be. Challenges occurred with teaching the midshipmen jargon at the same time as coding instruction, as well as ensuring the tool in the team project was working. Instructors believed the midshipmen had the most challenge with retaining the concepts and, more so, being able to explain them. There was no significant variance in coding performance between STEM and non-STEM midshipmen students, though students did vary in personality and persistence. One factor of persistence that was noticed was a midshipmen's visitation with the instructors during the instructor's office hours.

5 DISCUSSION

5.1 Team Project

The challenges the midshipmen faced in developing the tool more than likely originated from a lack of understanding that ultimately affected how individuals on a team communicated with each other and took ownership of individual tasks. This notion is supported by other accounts of teamwork improving when the tool development progressed, as students had better retained basic computer programming concepts. While the software application built by novice students is in beta version, the midshipmen's work and success has been measured with a focus on base-level usability and effectiveness. This will allow prospective users to see value in the product and support its continued development as the midshipmen continue to learn about computer science and enhance their skill.

5.2 Midshipmen Interviews

After interviewing the midshipmen, a few things became rather clear about the program. Primarily, the work and material covered sparked an interest in the students, most of whom have never been exposed to computer programming. Additionally, the program succeeded in maintaining the interest of the more experienced students. Taking a Python class before coding a tool in Python would facilitate a Python-based coding program, but it is not necessary, so long as the program includes enough time to educate its participants. Providing a space exclusively for work was shown to have led to an increase in focus and productivity during these sessions.

Personality variances between students may also account for the willingness to work independently or with teams. On the one hand, figuring out homework on a student's own account forced them to review and develop their own understanding of the Powerpoint material presented in the sessions. Therefore, having a consistent course structure with homework for students to work independently may be integral to teaching minority military students computer programming. All but one midshipman believed homework helped build skills in which they would later apply to the data mining tool. The one midshipman who believed homework was not relevant to their success specified that the inactive periods between and difficulty of each assignment varied so significantly that they felt that they had a good grasp of the lab material one week and would then feel completely confused by the next.

Instructing minority students with little knowledge of computer science can be executed in a range of techniques. Namely, showing examples presented in a structured way and allowing students to perform hands-on activities. Examples and hands-on activities that are grounded in relatable and applicable topics garnered the interest of just about every student without fail, which would support the claim that computer science interest and enjoyment increases when students are aware of the myriad of possibilities for creativity and discovery within the field. Interestingly, as pacing and group work were revealed as challenges, students did not believe the time allotted to assignments were problematic. Teaching how to pace working on computer science assignments for both group assignments and independent work may be useful in the instruction of beginner minority students.

5.3 Instructor Interviews

From the viewpoint of the instructor, the interviews provided insight on teaching methods, necessary prerequisites for beginner programmers, and student factors. As teaching format preference varied between instructors, it is noted that instructors should respond to what attracts their students the best as well as how they are most comfortable instructing. Regardless, thorough instruction is necessary to train students successfully. As students who are new to programming are bound to find challenges, the difficulty of assignments may best be instructed by arranging topics from least difficult to most difficult; this suggestion may not be as feasible while developing a tool due to tool development ordinal constraints.

As one-hour sessions were productive in the instructor and student perspective, it seems to be other teaching strategies that will more significantly determine what time is needed. Scheduling course materials can factor in the number and effectiveness of instructor assistants who can better interact with students individually and minimize whole-class review time, as a lead instructor orchestrates the program. Having credible and consistent source material can be a way to maintain organization. Having a dedicated lab and/or elective course can also help maintain the organization of the course and reduce the workload and expectancies of the students.

Beginner students of any major do not need a coding prerequisite for enrolling a computer programming course with tool development unless the tool is required to be highly intricate; the more intricate, the more coding knowledge a student would need. There should be emphasis on educating students on jargon, as jargon between student and instructor was a recurrent theme in challenges for both instructors and students. In many cases, students could code but had troubles explaining what they were doing or asking for assistance, contributing to confusion and lack of productivity. Personality and persistence were also confounding variables in a students' performance interfering with a student's ability to work independently and with groups, their motivation to get the intended results even through challenging moments, and their willingness to ask for help and visit instructors during office hours. Personality and persistence must be taken into account for students as students will vary how they understand instruction and how they will go about completing tasks.

5.4 Other Program Insights

There were periods of transition, not only in instruction style, but also in the degree of productivity throughout the course. Initially, PowerPoint lectures in-class lecturing was the only methods for learning that were utilized and proved itself to be non-conducive for knowledge retention. Often by the next lecture, students would have a difficult time recalling what was reviewed in the previous week. When homework, examples, and hands-on activities were incorporated regularly, there was a noticeable improvement in knowledge retention and positive attitude amongst the vast majority of the students. Though pushback came from the program participants at first, it became clear after a short time that interactive assignments were the only way to honestly check for understanding and guarantee participant engagement in the program. Genuinely interested and curious midshipmen improved their coding abilities tremendously from the level at which they began.

5.5 Limitations

As a specific program with only 18 midshipmen participants, some of the insight may not be completely generalizable. Qualitative analysis was conducted solely, complimenting the sample size and nature of the study. Therefore, the generalizability of findings may vary due to confounding variables that were either overlooked or not deemed necessary to cover through interviews. For example, the confounding variables of personality and self-efficacy, which were noticed during data analysis, were not directly assessed, yet were mentioned significantly. In addition, though there is no evidence to suggest, like any interview, there may have been instances of interviewee bias, leniency in rating the performance of the course, and accounts misrepresented through socially popular word choice. Findings support a contextualized understanding of a computer science course experience with novice computer programming students who were minority and military affiliated.

6 CONCLUSION

The course on cybersecurity fundamentals and the development of a data mining tool were successful, despite the lack of computer programming skills the majority of midshipmen had prior to the start of the program. The course experience evaluation, conducted by interviews, provided insight on teaching strategies and the most prominent challenges of new computer programmers. It is intended for this program to continue with plans to develop more in-depth tools. Future studies of the program will incorporate new confounding variables discovered through this study for more precise data analyses as well as incorporating growth measures and quantitative measures for increased generalizability and more rigorous insight.

ACKNOWLEDGMENTS

This material is also based in part upon work supported by the Defense Technical Information Center under Contract Number FA8075-14-D-0018-0026. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Technical Information Center.

This material is also based in part upon work supported by the National Science Foundation under Grant Number 1547793. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Mordechai Ben-Ari. 2001. Constructivism in computer science education. *Journal of Computers in Mathematics and Science Teaching* 20, 1 (2001), 45–73.
[2] Laurence Bradford. 2018. What You Need To Know About Cybersecurity In 2018. https://www.forbes.com/sites/laurencebradford/2018/03/30/why-people-should-learn-about-cybersecurity-in-2018/#5341dcba5d00
[3] Kwan Brown, Rebekah Lin and Bob Rudis. 2018. Quarterly Threat Report. https://www.rapid7.com/globalassets/_pdfs/research/rapid7-threat-report-2018-q2.pdf
[4] Art Conklin. 2005. The use of a collegiate cyber defense competition in information security education. In *Proceedings of the 2nd annual conference on Information security curriculum development*. ACM, 16–18.
[5] Arthur Conklin, Raymond E Cline, and Tiffany Roosa. 2014. Re-engineering cybersecurity education in the US: an analysis of the critical factors. In *System Sciences (HICSS), 2014 47th Hawaii International Conference on*. IEEE, 2006–2014.
[6] Steve Cooper and Steve Cunningham. 2010. Teaching computer science in context. *Acm Inroads* 1, 1 (2010), 5–8.
[7] Andrea Forte and Mark Guzdial. 2005. Motivation and nonmajors in computer science: identifying discrete audiences for introductory courses. *IEEE Transactions on Education* 48, 2 (2005), 248–253.
[8] Mark Guzdial and Elliot Soloway. 2002. Teaching the Nintendo generation to program. *Commun. ACM* 45, 4 (2002), 17–21.
[9] Nate Lord. 2018. What is Cyber Security? https://digitalguardian.com/blog/what-cyber-security
[10] Edward Sobiesk, Jean Blair, Gregory Conti, Michael Lanham, and Howard Taylor. 2015. Cyber education: a multi-level, multi-discipline approach. In *Proceedings of the 16th Annual Conference on Information Technology Education*. ACM, 43–47.
[11] Zoë J Wood, John Clements, Zachary Peterson, David Janzen, Hugh Smith, Michael Haungs, Julie Workman, John Bellardo, and Bruce DeBruhl. 2018. Mixed Approaches to CS0. (2018).