Proposed College Curriculum Changes for Producing Secure Developers

Christine Fossaceca
MIT Lincoln Laboratory
christine.fossaceca@ll.mit.edu

The need for more robust software is evident from the increasing number of cyberattacks occurring daily. [1] However, the fear of sophisticated nation-state actors and zero-day vulnerabilities is partially misplaced. Although these are formidable enemies, companies and governments should be more concerned about a major threat from the inside: poorly constructed code. A search of the 2017 CVE database shows that there are still new buffer overflow vulnerabilities being found [7], despite those being among the most basic type of exploits. This leads to the question: Why are developers still implementing programs with simple vulnerabilities?

The first place to look may be the educational background of software developers. One major problem is that students who want to become software engineers see cybersecurity related courses and think, "That doesn't apply to me". Then those students become developers, leaving security concepts to be implemented by a "security team". Security researcher Sarah Zatko gave a presentation [5] at the Hackers of Planet Earth (HOPE) Conference in 2014 diagnosing this systemic issue as "security afterthought syndrome", and lamented that cybersecurity isn't prioritized by many professors or taught by universities. Two years later, Professor Ming Chow of Tufts University and his colleague, Professor Roy Wattanasin of Brandeis University, replied to Zatko at HOPE 2016 [3], where they discussed being inspired by her presentation and made changes on their own campuses to address cybersecurity in computer science education.

In order to determine if other colleges and universities were following the urgings of experts in the security community by making curriculum changes, I recently conducted a survey of over 100 colleges and universities in the United States and presented the results at the IEEE Secure Development (SecDev) Conference. I worked with two of my interns at MIT Lincoln

Laboratory, and we reviewed the Computer Science curriculums of select schools, which were chosen based on their US News and World Report Rankings [6]. The schools were in the 2017 listings for "Top 50 Nationally Ranked", "Top 50 Regionally Ranked", and "Top 50 Computer Science Programs".

In the first part of the research, we looked at every curriculum and course description, searching to see if any required courses had the word "security" in the description. We found that 97 percent of computer science programs had at least one course that mentioned the word security in the description, however, only 31% of schools actually required one of those courses in their curriculum. Furthermore, it was determined that the word "security" is too ambiguous to rely on as a metric, as word "security" meant cryptography, network protocol security, privacy, forensics, or cyber policy, just to name a few categories discovered in the survey.

In the second part of the survey, we looked at the accreditations of the schools, and noted that the majority of top tier schools were ABET accredited (50% of Regionally Ranked schools, 92% of Nationally Ranked schools, and 94% of the Top Computer Science schools). This suggests that the ABET committee drives the curriculum requirements for these schools. A search of the ABET computer science curriculum turns up a requirement for computer science programs, "To have an understanding of professional, ethical, legal, security, and social issues and responsibilities." [4] Although some schools didn't have ABET accreditation, they usually had another accreditation listed on their website, and their curricula were quite similar to those of the ABET schools.

We are producing more software than ever before, in a landscape where there are also more malicious actors, so most software developers unknowingly have a target on their backs. We have to start preparing college students to enter the increasingly adversarial environment of the Internet by building security concepts into computer science and engineering education. Although there will always be new kinds of cyberattacks, computer science students should be well-informed about old attacks. As an example, students who are learning C programming should not be taught to use strcpy() without learning what a buffer overflow is. This issue was addressed in 2010 by three Carnegie Mellon professors who were planning to implement changes in the Computer Science curriculum to increase "our emphasis on the need to make software systems highly reliable." [2] Today, freshmen at Carnegie Mellon do, indeed, learn buffer overflow vulnerabilities in the required course 15-222 Principles of Imperative Computation, where students focus on the "correctness of programs", not "security".

I assert that graduating computer science students who go on to become software developers without learning secure coding practices ahead of time are left to learn on the job, and when a more experienced developer isn't auditing their work, another simple bug is implemented in production code, waiting to be discovered by the adversary. It is proposed that more schools follow the model of Carnegie Mellon in teaching secure programming techniques. To do this, reaching out to accreditation establishments and advocating for changes in curriculum requirements is necessary, as well as promoting the use of phrases such as "correctness of code" and "expected execution" rather than the vague word "security". This will in turn produce graduates who will be less likely to write programs with commonly known vulnerabilities.

References

[1] 2017 Internet Security Threat Report. Symantec Corporation, Apr. 2017, www.symantec.com/security-center/threat-report.

[2] Bryant, Randall E., Sutner, Klaus and Stehlik, Mark J. "Introductory Computer Science Education at CarnegieMellon University: A Deans' Perspective." Aug. 2010, www.cs.cmu.edu/~bryant/pubdir/cmu-cs-10-140.pdf.

[3] "Computer Science's Curricula Failure-What do we do now?" Chow, Ming and Wattanasin, Roy. HOPE 2016

[4] "Criteria for Accrediting Computing Programs, 2017-2018." *ABET*, Accreditation Board for Engineering and Technology, 2017, www.abet.org/accreditation/accreditation-criteria/criteria-for-accrediting-computing-programs-2017-2018/.

[5] “How to Prevent Security Afterthought Syndrome”. Zatko, Sarah. HOPE 2014

[6] *Rankings and Advice*. U.S. News & World Report, 2017, www.usnews.com/rankings.

[7] Security Vulnerabilities Published In 2017.” CVE Details Search, MITRE Corporation, 2017, www.cvedetails.com/vulnerability-list.php?vendor_id=0&product_id=0&version_id=0&page=1&hasexp=0&opdos=0&opec=0&opov=0&opcsrf=0&opgpriv=0&opsqli=0&opxss=0&opdirt=0&opmemc=0&ophttprs=0&opbyp=0&opfileinc=0&opginf=0&cvssscoremin=0&cvssscoremax=0&year=2017&month=0&cweid=0&order=3&trc=9201&sha=815cc1a3d2c4b72bf23b8a2fa85939f5ab0041c0

Bio

Christine Fossaceca is a cybersecurity researcher at the MIT Lincoln Laboratory, focusing on tool creation, exploit development, vulnerability research, and reverse engineering. She first became interested in cybersecurity education when she entered the workforce as a recent graduate and started feeling overwhelmingly unprepared to write “unhackable” code. In speaking with other recent graduate friends, she noticed a trend among software developers to rely heavily on “security teams” to pentest their code for them in the deployment process, rather than the developers themselves following any particular set of secure coding practices. In her discussions, the nervous laughter of her colleagues usually covered up a real fear of causing a major security breach because the review team didn’t patch something. She started to question, “Why did my professors even teach me strcpy()? Why didn’t the databases course include a section where we tried to perform SQL injections on our classmates? Why didn’t any of my classes encourage me to use a debugger like gdb?” After becoming interested in the topic, she became involved with the IEEE Secure Development conference (IEEE SecDev) and formed a group on improving security education with collaborators from Google and Tufts University.