Aditya Hurry

The Security Risks of Student Led Development Teams

by Aditya Hurry

Abstract

Students of computer science around the country are always looking for an outlet through which to utilize the skills and technologies they learn during college. Students at Tufts University, for example, through clubs like JumboCode, are given numerous opportunities to contribute to real world projects that fulfill real needs faced by companies. However, for the companies in question, the potential risk posed to them due to the information that they provide to their student-development teams may not outweigh the benefits. Students are just that: students; while they may be eager to learn, the software they build will not be the most secure. For example, giving students the opportunity to build a database management system for a non-profit that gives aid to differently abled people may be a good thing in principle; however, the data inside that system would, necessarily, be confidential and thus the security of that system is of paramount importance. Student teams are unfit for building secure applications, primarily because of the role that security plays in their education; that is to say, close to none. The technologies that students are often exposed to in academic settings, and thus are more likely to use in pseudo-professional ones, are not ideal. In some cases, even if the technology itself is suitable, students often haven't been taught how to use them with security in mind. This paper will argue that student-led development teams are a bad idea for any company dealing with even vaguely sensitive information.

Chapter One: Introduction

Companies around the country, and indeed around the world, are starved of people with computer science skills. The idea of the 'tech-industry' is a bit of a misnomer as well: every single industry in the world today has technology at its heart, doing the hard calculations required and keeping track of the terabytes of data generated on . As would make sense, computer scientists skills are in high demand – and US universities are ramping up their computer science departments to meet that demand. Students, however, are not usually content to wait until they receive their diplomas to get their hands dirty: many programs offer students the chance to put their budding skills to use, such as the California Institute of Technology's "Hack Society" class. The gaps in real-world opportunities offered by curricula are often filled by on-campus organizations – Tufts University's JumboCode is a prime example. These organizations' and classes' manifestos are certainly noble: to provide technological assistance to companies that would not otherwise be able to afford it, while allowing their students to flex their proverbial coding muscles. At first glance, it certainly seems like a win-win situation.

If only it were that easy.

The truth of the matter is, to use economic jargon, that the demand for technological chops is being met with the supply of an inferior good. That these students' skills are inferior goods is not entirely a fault of theirs, but rather of the universities they attend – colleges do not train their students in the fundamentals of computer security thoroughly enough for them to be set loose, as it were, into the professional world without some guidance.

Happily ensconced in their CS departments' programs and infrastructure, students rarely, if ever, experience security issues in college; the result when coding for real-world applications is to forget that those issues do exist in the wild.

Companies that utilize these students talents (or lack thereof) are also part of the problem: often lacking any technological expertise themselves, they are woefully ill-equipped to evaluate whether the app that a group of students built for them will compromise the integrity of their business.

This paper will build upon this argument – first by demonstrating the lack of education about computer security in the United States' top Computer Science departments; then by providing some exemplars of the kinds of companies using these students' services. Finally, this paper will provide some action items to bridge the gap between supply and demand in as safe a manner as possible.

To The Community

The genesis of this paper is personal experience: having been a project lead for JumboCode at Tufts University for two separate projects, I have had the unique privilege of building apps for real world applications – first for the wing of the US National Park Service that oversees the Boston Harbor Islands, then for Tufts University's own Admissions department. With this privilege came responsibility: making sure that the app we built would work was the first priority; it was not until I took a class on computer security that I realized the second priority – not compromising the integrity of the organization's business or the privacy of the users in question. The problem with the second of these priorities was not knowing where to

start. Upon a little reflection as to why this was so, the answer came quickly: I had never been taught where.

The concerns outlined in this paper are important precisely because they are live: every day, students contribute to building websites for clients without knowing what Cross-Site Scripting is, how a SQL injection happens, or how to protect against either. The companies in question must be made aware of the risks inherent in the use of students' (often) free labor. It is from this need that this paper was born, and it is this lack of information that this paper seeks to combat.

Chapter Two: Computer Science Curricula

In this section, we will look at the curricula of four of the top five computer science departments in the country, focussing specifically on the importance they place on security. We will focus specifically on four-year undergraduate programs. The top five programs in the country are, according College Choice rankings: The University of California, Berkeley; Massachusetts Institute of Technology (MIT); California Institute of Technology (CalTech); Georgia University of Technology (GeorgiaTech); and Carnegie Mellon University (CMU). Of these, we will look at all but GeorgiaTech's curriculum – the reason being that they do not follow a conventional credit system of requirement fulfillment, but rather operate in themed "threads", the requirements of which are not publicly available on the web.

University Of California, Berkeley

The University of California, Berkeley tops the list of College Choice's best undergraduate computer science departments. Alas, even at the best of institutions, apparently, security classes are not required.

One of only two schools in the top five to do so, UCBerkeley does offer a class on security specifically – covering a wide gamut of domains, "including the web, networking, operating systems, and cryptography[, ...] It goes over security techniques, programming involved to repel attacks, encryption, computer safety, [and] networking basics" among other things. This is certainly positive – the fact that it is not a requirement to graduate is not, however.

Another interesting, and worrying, aspect of UCBerkeley's computer science curriculum is that its class on the architecture of the internet, which covers the "concepts and fundamental design principles that have contributed to the Internet's scalability and robustness", does not explicitly address the network security vulnerabilities that are part-and-parcel of operating on the internet.

The only other class wherein security is explicitly mentioned as part of the syllabus is UCBerkeley's Operating System and System Programming class. While this class is also not required, it is highly recommended for students to take it.

Massachusetts Institute of Technology

The Massachusetts Institute of Technology, coming in at second in College Choice's rankings, has a woeful lack of computer science education as part of its curriculum. A simple control-F on its computer science department's course listing page reveals only one mention of "security" at all: as part of its Computer System Engineering class.

Their Elements of Software Construction makes no mention of the importance of including security from the get-go of the design process; the lack of a focus on security in this class is especially troubling given the preponderance of malware on the internet today.

California Institute of Technology

The California Institute of Technology comes in third on College Choice's listing of best undergraduate CS programs in the country. Little wonder then that, given the top two's lack of security education, CalTech follows suit.

The only mention of security on CalTech's course listing page is for its Quantum Cryptography class. To add insult to injury, this class is only open to PhD students and undergraduates in Quantum and Molecular Computing sequence on offer.

Similar searches for "safety" and "privacy" yield similarly disheartening results – the former resulting in none, the latter in three – all for Introduction to Data Privacy. However, even this Data Privacy does not approach privacy from within a CS paradigm; rather it focusses on "the tradeoffs between useful computation on large datasets and the privacy of those from whom the data is derived".

CalTech is unique among its peers in the top five in one way, however – their Hack Society class. Built specifically to address the "large gap between the public and private sectors' effective use of technology," Hack Society's presence in CalTech's course listing is troublesome given what is not present: any indication that students building "innovative solutions to problems faced by society" are sure that they are not adding to those same problems by creating exploitable software. The fact that this course advertises its connection with industry – "corporate, nonprofit, and government partners" – highlights the fact most in industry do not even realize that this is a problem.

Carnegie Mellon University

Rounding out the pack, Carnegie Mellon Computer Science curriculum is a breath of fresh air in an otherwise squalid scene.With courses in subject ranging from the specifics of security as it relates to programming in C and C++, to Information Security, and Computer and

Network Security, CMU certainly offers its students ample opportunities to learn about security as it relates to their chosen field.

CMU also, however, stops short of making a knowledge of security a requirement to graduate. Instead, within one of the required elective categories available, security is addressed in two of the five classes consistently available. This is not ideal, of course, but is certainly more encouraging than the requirements of CMU's peers.

Tufts University

Tufts University's computer science curriculum fares slightly better than some of those in the top-five, but not appreciably so.

While a class in security is available, it is not a requirement to graduate: it is just one of many upper level electives available to CS majors. Their web development class does, however, also cover some aspects of security on the web.

Tufts University is also home to the inspiration for this paper, the student group named JumboCode, which offers its members' talents to non-profits in the Greater-Boston area in need of software work.

Chapter Three: The Companies Concerned

In this section, we will look at some of the people in the market for the work students produce – whether that be students themselves, non-profits, government agencies, or corporate partners. What all these groups have in common, most of the time, is a singular lack of knowledge about the dangers that these technology poses for them. While that lack of knowledge may be slightly more excusable for private citizens downloading a random app off the Google Play store, it is not excusable for the companies who commission these apps.

We will be looking specifically at two of JumboCode's clients – one past and one present: the Boston Harbor Islands, the Boston equivalent of the National Park Service; and the Tufts Admissions department.

Boston Harbor Islands

In December of 2015, the Boston Harbor Islands management responded to an advertisement placed by JumboCode. They had a problem – the ferry schedules that they advertised on their website were too clunky and hard to navigate for their older clientele. What they wanted JumboCode to do was to build a better version of it – a ferry scheduling app that would allow users to plan a day's itinerary, or just figure out when they should leave to make the ferry on time.

The reason they came to JumboCode was twofold – firstly because their office had a sum total of one employee dedicated to maintaining the entire technological aspect of their operation and keeping the website up-to-date and functioning. They simply did not have the bandwidth

within the organization to spare someone to work on an app for them. The second reason was funding – they had no funds to hire an outside contractor to build this app for them. This is where JumboCode came into the picture: the ability to get some much needed work done for free was too good an opportunity to miss.

While the app in question did not have any significant security vulnerabilities, seeing as it did not deal with sensitive data or have the need for user authentication, it was symptomatic of a larger problem: that the client would not have had any clue if it did have security vulnerabilities. Before the commencement of any work on the project, there was absolutely no disclosure of the skill level of the members of the team, some of whom were first-years taking their first computer science class that semester; neither was there any oversight during the development process, during which members of the team were given full access to the website in question.

The problem here was trust: too much of it. A common method of ensuring security in computer science is the principle of least privilege – giving only as much access is absolutely necessary. This principle was certainly not put in practice in this instance when it should have been.

Tufts Admissions

The Tufts Admissions Department reads over 20,000 applications every year in order to build each incoming freshman class. An integral part of the admissions process is the events that the department puts on through the year for prospective students – from smaller, department-

based open houses in the fall, to events like Jumbo Days in the spring, which draws over a thousand admitted students to campus every year.

In the spring of 2016, JumboCode was commissioned to build an app for the Admissions department that is intended to serve as a prospective students one-stop-guide for the admissions events on campus on a particular day. The app has two components – the front-end app to display all the information required, and a back-office application to allow for data entry to update the app.

The admissions department sidestepped the trust issue by assigning an internal representative to oversee the development of the app. However, some of the problems remained – with students of varying experience making up the team, the possibility of faulty software is certainly present. For example, a vulnerability in the back-office application could very easily propagate to the front-end, which could result in the defacement of the user-facing portion of the app: at best, it would be embarrassing; at worst it could render the app unusable.

Chapter Four: Action Steps

For Universities

The lack of cybersecurity education is not a new one. It is, however, still a largely ignored one. The only way to ensure that students of computer science have a working knowledge of the security vulnerabilities inherent in their chosen field is to include it as a requirement to graduate. The United States government must move to treating computer science in the same way that they treat engineering – by requiring programs to be accredited.

Part of this accreditation should be a compulsory security class, as well as one on the ethics of computing. Anything less would not do: leaving it up to students has proven ineffective, partly because of the attraction of more "in" sub-fields within computer science. Given the choice between an artificial intelligence course and a computer security course, and time enough to only take one, too many students would choose the former. By framing it as a choice, it is placed at the same level of importance as other sub-disciplines; this is a false choice – whatever the discipline, an eye to security must be incorporated from the get-go.

On the other side of this equation is the people facilitating student-client partnerships.

Clubs like JumboCode and classes like CalTech's Hack Society should have their own internal quality management systems, from clear coding standards to pre-deployment minimum-testing requirements. Furthermore, they should be wary of accepting projects that have significant scope for vulnerabilities – the first step to solving a problem is accepting that there is one.

For Companies

The primary responsibility of any business is to ensure the integrity of its operations. Putting blind trust in any party is unwise – indeed, this principle forms the backbone of all contract and business law. That this principle is not exercised when it comes to the technology that clients of a business will interact with is bewildering.

For clients of student-led development teams, this paper has one piece of advice: not to take things on faith. When contracting with student-led teams, clients have a responsibility to know who is working for them; they also have a responsibility to check the software that is built for them independently. This should entail running static scans of said technology, as well as insisting on thorough testing of all code before it is deployed.

Chapter Five: Conclusion

There are certainly problems with the current state of student-led real-world development. Getting rid of organizations like JumboCode is not the answer – real world projects do provide invaluable experience for any student of computer science; they also have the potential to provide significant value-additions for their clients, if executed well. What is needed is an acknowledgment of the limitations of students by both parties – the students themselves and their clients.

Absent any significant legislation around requirements for computer science undergraduates, the lack of security education is unlikely to go away. In an ideal world, individual universities would take up the mantle of ensuring that their students be equipped to deal with the problems they may face "in the wild," as it were. Until that becomes a reality, a recalibration of expectations on the part of clients is in order – if they would not ask a business-school undergraduate to implement an investment strategy, they should not ask a computer science one to implement their financial database. To do so would be akin to playing with fire.

Works Cited

Bertram, Vince M. "The Shocking Data Security Gap in Computer Science Education." THE Journal. Public Sector Media Group, 5 Dec. 2016. Web. 14 Dec. 2016.

“Carnegie Mellon Course Listing”. (http://www.csd.cs.cmu.edu/content/bachelors-curriculum-admitted-2014)

“Best Computer Science Programs.” Best Computer Science Programs | Top Computer Science Schools | US News Best Graduate Schools, US News And World Report, 2014, grad-schools.usnews.rankingsandreviews.com/best-graduate-schools/top-science-schools/computer-science-rankings.

MaryAlice, Bitts-Jackson. “There’s an App for That.” Dickinson College, Dickinson College, 30 Mar. 2015, www.dickinson.edu/news/article/1504/there_s_an_app_for_that.

“Philanthropy, Meet the Future.” JumboCode, www.jumbocode.org/.

“University of California, Berkeley Course Listing” (https://eecs.berkeley.edu/resources/undergrads/cs)

“Massachusetts Institute of Technology Course Listing” (http://student.mit.edu/catalog/m6a.html)

"California Institute of Technology Course Listing" (http://www.cms.caltech.edu/academics/ugrad_cs)

"Carnegie Mellon University Course Listing" (https://www.csd.cs.cmu.edu/academics/undergraduate/overview#bscsexplore)