XP 2019 Panel: Security and Privacy

Dennis Mancl[1(✉)] and Steven D. Fraser[2]

[1] MSWX Software Experts, Bridgewater, NJ 08807, USA
dmancl@acm.org
[2] Innoxec, Santa Clara, CA, USA
sdfraser@acm.org

Abstract. In reaction to reports of recent high-profile software security and privacy failures in our always-on agile world, users and regulators are demanding that companies deliver more trustworthy and resilient systems. This panel discussed some of the strategies and best practices for "building-in security" to our products and systems in contrast to "bolting-on security" – and how threats should be assessed and mitigated to avoid the unintended consequences of flawed design decisions.

Keywords: Security · Privacy · Design

1 Security and Privacy at XP 2019

Security and privacy issues for today's computer systems and applications are in the headlines. Every day there are business stories about companies who lose customer data to hackers, stories about system outages caused by hackers, and stories about shadowy networks of criminals who take over computers and blackmail helpless companies, governments, and ordinary users. The panel discussion on security and privacy issues was motivated by the special challenges to building software systems in an agile way: even in a lightweight development process, all developers still need to pay attention to software quality, security from hackers, and protection of the private data of users.

The panel represented a cross section of industry experience in software applications. Landon Noll is a computer security expert, astronomer, and frequent traveler to the South Pole. He currently works for Cisco in California. Kelsey van Haaster is a software engineering expert at ThoughtWorks Australia who works on her company's internal security practices: passwords, security hygiene, and identity-related issues. Scott Ambler is a consultant, coach, and author of books on object-oriented design and Agile development methods, and he is currently Senior Consulting Partner at Scott Ambler + Associates based in Canada. Dennis Mancl worked as an internal software engineering and software quality expert at AT&T and Lucent in New Jersey. Robert Crawhall is a Principal Consultant at Innoxec in Ottawa, Ontario, where he developed technologies for government and industry, including nuclear power and telecom systems. Steven Fraser, the panel impresario, is based in California where he advises on tech transfer and open innovation strategies for Innoxec. Previously he was the Director of the Cisco Research Center and the Lead for HP's Global University Programs – and he has organized and delivered over 75 software engineering conferences, panels, and workshops.

R. Hoda (Ed.): XP 2019 Workshops, LNBIP 364, pp. 137–142, 2019.
https://doi.org/10.1007/978-3-030-30126-2_16

2 Security and Privacy Panel Discussion

Agile development creates special challenges in developing software systems that appropriately address security and privacy.

In this panel, the following issues were raised by the panelists: Security design should cover prevention, mitigation, and detection of security issues. Good "security hygiene" is an important practice for all software designers and developers. Many of our devices today are inherently insecure, and we need to be aware of the risks as we design our systems. In any development effort, especially in a fast-paced Agile development cycle, it is important to identify the security requirements up front and always include these requirements in the test planning and execution. Today's developers need to address more government regulations on security and privacy. In an Agile environment, customers need to be made aware of security tradeoffs. In addition, there is a significant "training gap" in security and privacy training.

Landon Noll started the discussion by asserting that we can never be perfect at preventing security and privacy issues in our software. We should try to prevent security problems when we can, but whatever we can't prevent we should try to mitigate, and whatever we can't mitigate we should at least try to detect.

Landon explained that testing, logging, and monitoring are the most important security-related practices and tools for development teams. Development teams need to think about security tests up front, and the teams need to have a process to continuously improve tests as the team learns more. An Agile development approach is valuable because it supports continuous improvement. Landon explained the value of Agile's focus on improvement, "I think Agile process gives you a really good way to be in this continuous process of improving your testing, testing your improvements, measuring your improvements, measuring the logging, and logging the measurement."

But there is no silver bullet for security and privacy. Landon warned about some of the risks of our current security technology. Many of our security technologies are inadequate. Landon pointed out, for example, "We demand too much from hash functions – more than they can deliver." He complained that many of our security standards say "use this cryptographic hash function," but the best hash functions "are being attacked faster than Moore's Law." Cryptographic hash functions are at the heart of authentication and communications security systems, including digital signatures and public key cryptography [1]. Landon concluded that designers of most secure systems don't pay enough attention to important activities in using encryption tools, such as key generation and key management.

Kelsey van Haaster emphasized the importance of good security hygiene. In her company, security is everyone's concern. "At ThoughtWorks, we ensure that every consultant (whether they are a software developer or a business analyst or a project manager) goes through our App Sec 101 training. When we are working with clients and building products, both for clients and internally, we take the approach that the security stories need to be there from iteration zero. We build that into part of our process right from the start."

Scott Ambler and Landon Noll told some cautionary tales about the theft of information from our computers and devices.

Scott had security and privacy anecdotes from his days working for IBM where some global customers of IBM would routinely try to steal data from visiting outsiders. His colleagues in the IBM sales organization knew which companies were "bad actors," and they would warn him, "Don't bring your phone and laptop to this customer site." They suspected that even an encrypted hard drive could be vulnerable.

Landon reported about the deficiencies of today's hardware: "The overall state of computer hardware is somewhere between appallingly poor and unacceptably poor." Every computer is vulnerable if a hacker finds a way to load and execute small selected chunks of code. According to Landon, "All commercially available microprocessors today are subject to having their integrity compromised by a sequence of non-privileged instructions." This is bad news for digital security for applications such as containers, virtual machines, IoT, and cloud computing.

Dennis Mancl advocated for the importance of security requirements – which need to be part of the requirements documentation even in Agile development projects. "Security requirements are really important, because if you are going to test a system, what are you going to test it against? If there are no security requirements, they don't care about it."

The ITU-T X.805 security architecture recommendations include a set of eight security dimensions, and every system needs to have some security requirements in each area [2]. The security dimensions are linked to potential security risks, threats, and vulnerabilities. Access control and authentication requirements specify the rules for physical and network access to a system and the rules for application and network passwords. Non-repudiation requirements explain the transaction records and logs that must be maintained to allow stakeholders to prove that an event took place. Data confidentiality and communications security requirements ensure that data is kept private. Data integrity requirements describe the internal system processes to prevent data from unauthorized duplication or modification. Availability requirements define the responses to system overload and to denial of service attacks. Privacy requirements prevent disclosure of information about users' network communications activities, such as the IP address of a user or websites the user has visited.

Security requirements are not easy to write. When writing security requirements, it helps to consult with experienced testers. Testers can give constructive suggestions for key security requirements to include in requirements documents or user stories, based on the testers' experiences with security testing for similar systems.

Security analysis often needs to focus on "negative" behavior. Many security requirements are "misuse cases" – key scenarios that explain what things need to be prevented or mitigated. In Agile development, too many of the user stories written by developers and users are unrealistically positive: most of them are "the system acts nice" stories. It is a good idea to get together with friends who are testers when doing user story brainstorming, because testers are better trained to think about negative cases.

There are many new security and privacy regulations in place, such as the European Union's General Data Protection Regulation (GDPR), which was put into effect in 2018. Robert Crawhall explained the pressures on governments to create new security regulations.

Robert agreed that governments really want commercial computer systems to be more secure and private, but "government has a very limited toolbox." Their standards

are intended to improve focus on certain issues by companies. “When the government gets to control the supply chain, they start implementing standards to try and cover their butts.” This explains complex standards like the GDPR privacy rules. It is impossible for governments to catch every violation of the new security or privacy regulations, but they can make an example of a few big companies who don’t follow the rules. “If you’re the European Union, you bring in GPDR for the same reason – and you hope that handing out a few $5 billion fines will cause industry to sit up and take notice.”

Some of the audience members added their own observations. “Today, I got an email request from where I work – telling me to log in using my credentials to do security training on how not to click bait in email.” Landon was interested: “Did you do the training?” “No, of course not.”

The same audience member had another complaint about the effect of GDPR on application interfaces. Users are asked to agree to allow access in the middle of an application, and this access permission process is extremely superficial. “I would suggest that GDPR makes us significantly less secure and private than we have ever been. Because to access anything, no matter what it is, you have to say ‘Yes, I agree’. So all of a sudden, we are giving permission on purpose to access data, and we are not reading all of the levels of permission we are giving.”

One question from the audience explained the problems of getting customers to be interested in security. “Often your customer – when you do work for a large company and they have a Product Owner for an Agile project – they want their features. So they find it hard to prioritize security.” Even if the customer understands the need for some good security hygiene and some up-front analysis of security issues, it can be easy for them to put a lower priority on adding new security functionality to face new threats. Some customers are just hard to convince that they should invest in security prevention and attack analysis.

Kelsey explained how at ThoughtWorks, the teams actively include businesspeople in the project inception process – the early discussions where security scenarios and other iteration zero tasks are planned. She explained, “They don’t need to get involved in the technical details, but there is always a financial or a business or a reputational risk that those folks care about.”

Robert commented that a company needs strong accountability and transparency in its decision-making process. Early in his career, he worked on the design of nuclear power plants, and the “signoff” process ensured that engineers took quality seriously. “In a 40-year engineering career, the one day I still remember is when I put my stamp on the design of a nuclear pressure vessel, with implications for northern Ontario if I made a mistake on the calculations.”

Strong oversight can also help. Robert explained that in his experience in the development of nuclear power plants, the ultimate yes/no decisions on all quality-related issues were signed off by the “head of quality” for the company, who reported directly to the CEO. If a decision went wrong, it was clear who was responsible.

Scott and Robert reported on aspects of customer apathy and ignorance. Scott pointed out the inevitability of customer apathy: “The problem is you’re fighting human nature. It’s human nature not to think about bad things.” He talked about how many people build houses in river floodplains or in coastal areas vulnerable to hurricanes. And Robert pointed out the differences between the layout of the original

Canadian telephone trunk network and the design of the internet exchange network a generation later. The main east-west trunks of the older telephone networks went 45 miles north of Toronto: "if they lost Toronto, they didn't want to lose the telephone connection between Montreal and Calgary." But in the 1990s, Bell Canada chose to install their main Toronto internet exchange in downtown Toronto in the One Front St. building, at street level, subject to flooding and perhaps vulnerable to a terrorist attack. Robert comment was "it's because it became deregulated."

One questioner lamented the lack of education in digital security issues and suggested that security training needs to start in elementary schools. "At what point should we teach software developers how code is compromised based on data, and proven practices to avoid those compromises?… in all of my time and all of my education since I started teaching at the undergraduate level in 1971, I have never seen a course teach this." If no one learns about digital security issues, developers will continue to build systems with weak security.

Robert and Landon suggested that teaching about digital security ought to be done when middle schoolers or elementary school students first learn about programming.

Robert explained that as part of the Cyber New Brunswick initiative, the schools in the Canadian province of New Brunswick now all include cyber security concepts in the middle school curriculum. Robert explains, "They need to catch kids before they make the STEM/not-STEM decision."

Landon explained that it is important for young people to learn to think about what can go wrong. He offered his experience in teaching security thinking at kids camps: "I get them to do something simple like print 'Hi mom.' Then I ask 'How can you mess it up?' One clever girl, she thought a little bit, then she went over and turned off the computer." Once you have figured out how to mess something up, "you need to think about how to fix that messing up," and then go back and forth with the students. "They learn that they maybe need to save their work." It starts them thinking about how to analyze problems.

Scott explained that the education and training gap about security is part of a bigger problem. "When are we going to start teaching people to code/program? Forget asking when we're going to teach programmers how to be secure, when are we going to ask our people to have adequate educations? I work in banks and insurance companies, and most of the developers in the IT space don't have a background in programming. They don't have the fundamental skills of their jobs."

Scott explained that we would be making the same comments in a panel on User Experience or Performance, because there are similar gaps in knowledge and experience across industry in those areas.

Kelsey pointed out that companies can play a role by putting computer security resources in the hands every employee and their families. "We provide every employee with a family subscription for a password manager, and we insist that they in fact use it." Kelsey points out that this can save money: "it is cheaper than something going horribly wrong." This is a policy that can have a positive long-term impact: "Kids of our employees are growing up understanding about their own personal security."

3 Summary

Security and privacy will continue to be important issues to address for Agile development. The panelists all warned about the gap between security threats and the current security technologies. They pointed out the deficiencies in software development training – not enough focus on what can go wrong. In a corporate environment, security is everyone's concern and all developers and users need to practice good security hygiene. It is essential to document the security requirements in the course of developing a system. Some security and privacy standards, such as the GDPR privacy rules, may actually create new security and privacy risks. There can be a disconnect between the development teams and the Product Owner about the relative value of work items that improve an application's security infrastructure. Finally, everyone needs some encouragement and training in security and privacy issues: something as simple as a company-provided family subscription for a password manager can improve the awareness of good security practices.

Landon summarized the main message about security and privacy for the audience: "Train early. Test often." He concluded with a message of Agile improvement: An Agile process supports "continuously improving your testing, testing your improvements, measuring your improvements, measuring the logging, logging the measurement." Small group of developers can "go off and try to see if they can break something, and figure out how to mitigate that." It's a process that requires new software to be tested early: "and continue testing from the foundations, as your code needs to work – and to work correctly – with the reliability, privacy, and integrity that you need."

References

1. Chi, L., Zhu, X.: Hashing techniques. ACM Comput. Surv. **50**(1), 1–36 (2017). https://doi.org/10.1145/3047307
2. International Telecommunications Union (2003): ITU-T X.805 Security architecture for systems providing end-to-end communications. https://www.itu.int/rec/T-REC-X.805-200310-I