Begin by reading the resources in this module’s Resources section. Then **reflect on the questions below and on the understanding and importance of software security and a developer’s role in it.**

### What is your role in solving security concerns as a developer? What might solving security concerns as a developer involve?

Security concerns are of mounting importance for developers. Entire energy grids and healthcare systems are being hacked and held at ransom causing people to die (Maclean-Bristol, 2024). People also died during the incident with the security company CrowdStrike where their update that was functioning for 4 months caused a boot loop because of a Microsoft rapid response update to the part of the kernel their Falcon software interacts with (CrowdStrike, 2024). Software bugs in the radiotherapy equipment named Therac-25 killed people from integer overflow on bug counts to a race condition from moving the mouse too fast (Holzman, 2023), and this was a device that blessedly existed before the internet. Who knows what incredible harm it could have caused if it was attacked. Finally, a lack of security in the web overall has allowed misinformation and disinformation to spread at a rate that defies measurement, alienating families from each other, separating friends, and causing acts of terror which have led to multiple deaths (Suber, 2021; Jensen & Kane, 2021). Similar misinformation issues allowed SARS-CoV-2 to wreak havoc throughout the United States, causing more than 1.46 million excess deaths (Mathieu et al., 2024), a scale so beyond the pale that if lined up end-to-end, you could bury each of these Americans along the roadside from the border to Mexico at Loredo, Texas to the border to Canada at Detroit, Michigan.

All those preceding examples are to show that security and code quality are far more serious than they appear, as it is not just the profits that scale with this field, but the mistakes, too. As developers, we sit at an intersection between all disciplines and all walks of life. Our biases, when left unchecked, will be spread globally, and if those biases introduce security concerns, we become globally vulnerable. Assumptions about how software will function, who will use this software, and what accessibility concerns should be taken into consideration are all questions that can create security concerns that, when exploited, may lead to death. Assembling teams from diverse backgrounds whose thought processes are also similarly diverse is paramount. The idea of moving fast and breaking things belies an assumption that the things being broken are not just replaceable, but unlimited in number. Neither are true, and we pay a blood-tax for it.

These facts reinforce the simple fact that security is a part of software design that should be considered from inception. It should be a point of shame that our industry has to be coerced and forced to adopt bare minimum standards for introducing basic security measures in systems that determine the lives and futures of their users. Integrating basic security measures in software, such as having a basic access control system form a prebuilt library that only needs some variables filled out, again should feel rather shameful that it is so frequently not done.

### Where does security fall within the software stack and development life cycle?

The Software Development Lifecycle, or as Manico and Detlefsen correctly put it in *Iron-Clad Java* in chapter 10, the Secure Software Development Lifecycle, it squarely fits within the development cycle as-is. Rather, these parts of the team are often understaffed or overlooked, such as Security and Software Architects being stated as having “easy” 60-hour work weeks. Most of the easy offsets for these unsustainable hours often involves investing in existing staff knowing basic security patterns and knowing to avoid certain anti-patterns. Businesses are already interested in risk management and are often quite easily sold on the idea, so it is not a stretch of logic to include security from the outset by performing threat modeling in the requirements gathering phase as it is a genuine requirement, then in the design phase incorporating counters to that threat model inside the design of the code, and including security testing in the integration tests that we already write. The entire tech stack here easily coexists with the existing tech stack used globally.

### How might you add security measures to transform a DevOps pipeline into a DevSecOps pipeline?

As Jeganathan (2019) notes, each part of transforming into DevSecOps should be done slowly and with the intent to remain flexible. Allowing teams to no longer be siloed is the first, largest step, as they need to be reshaped into a larger collective team, a proper Blue Team. The next would be to slowly and iteratively introduce security literature and culture, adopting a different stance on software development. Starting with small projects where the consequences are more negligible for good practice runs, integrate singular tools to the development environment, slowly building up to larger projects. All the while, management should be doing their utmost to show how these measures will both increase future revenue and decrease future costs so that the development of these measures is not slowed or halted unnecessarily. Simply put, we are building a Ship of Theseus, only by the end of the exchange of parts, its hull should be iron-clad, its sail from heavy canvas, and its crew performing regular inspections of their vessel.

### The article suggests creating and following a plan to secure the entire DevOps life cycle. What is included in the suggested plan? Would you recommend following the plan?

As previously mentioned, the 2019 article by Jeganathan largely adopts the lessons learned from Psychology, slowly rolling out each measure to change DevOps to DevSecOps. The concept here is that each part of DevSecOps becomes normalized and enforced, and is made into institutional policy, one part at a time. A rapid shift in culture, even when it is both warranted and wanted, still causes massive upset as humans are wont to change but are not so inclined to adopt to massive change. Culture often shifts slowly, and forced cultural changes must similarly creep forward. From personal experience and from the world of professional psychology, building habits takes a great deal of time, and culture is merely a collective collection of habits. I agree with Jeganathan, we must slowly build these habits, slowly find ourselves shifting culture and having it be regular and routine.

Not parameterizing a SQL query should be met with a puzzled look, not implementing access control should find quizzical gazes, and resisting the idea of making a more secure system of software and secure internet should be considered quixotic and strange. These reactions will take time, certainly, but we already have a plan for how to make it a reality within software development.

References

CrowdStrike. (2024, August 6). Executive Summary: Root Cause Analysis — Channel File 29. <https://www.crowdstrike.com/wp-content/uploads/2024/08/Executive-Summary_Root-Cause-Analysis_Channel-File-291.pdf>

Holzman, E. (2023, June 16). *The Worst Computer Bugs in History: Race conditions in Therac-25*. BugSnag. <https://www.bugsnag.com/blog/bug-day-race-condition-therac-25/>

Jensen, M. A., & Kane, S. (2021). *QAnon-inspired violence in the United States: an empirical assessment of a misunderstood threat*. Behavioral Sciences of Terrorism and Political Aggression, 16(1), 65–83. <https://doi.org/10.1080/19434472.2021.2013292>

Manico, J., Detlefsen, A., & Kenan, K. (2014). Iron-Clad Java: Building Secure Web Applications (M. Smith, Ed.). McGraw-Hill Education.

Mathieu, E., Ritchie, H., Rodés-Guirao, L., Appel, C., Giattino, C., Hasell, J., Macdonald, B., Dattani, S., Beltekian, D., Ortiz-Ospina, E., & Roser, M. (2024, August). Excess mortality during the coronavirus pandemic (COVID-19). Our World in Data. <https://ourworldindata.org/excess-mortality-covid>

Suber, J. (2021, June 10). *Examining extremism: QAnon: Examining extremism*. Center for Strategic & International Studies. <https://www.csis.org/blogs/examining-extremism/examining-extremism-qanon>

Maclean-Bristol, C. (2024, June 21). *Have cyber attacks killed people?*. BC Training. <https://www.b-c-training.com/bulletin/have-cyber-attacks-killed-people-updated-june-2024/>