Understanding Developer Security Archetypes

Ita Ryan[a,b], Utz Roedig[a,b,c], and Klaas-Jan Stol[a,b,d]
[a]*School of Computer Science and Information Technology, University College Cork*
[b]*Science Foundation Ireland Centre for Research Training in Advanced Networks for Sustainable Societies*
[c]*CONNECT—the Science Foundation Ireland Research Centre for Future Networks and Communications*
[d]*Lero—the Science Foundation Ireland Software Research Centre*
Cork, Ireland
ita.ryan@cs.ucc.ie, u.roedig@cs.ucc.ie, klaas-jan.stol@lero.ie

***Abstract*—As software systems penetrate our everyday lives, security has risen to be a key concern. Despite decades of research leading to new tools and practices for writing secure code, achieving security as a key attribute remains highly challenging. We observe that much of the literature considers developers to be homogeneous and interchangeable. The differing circumstances of developers that might play a role in the writing of secure code have not been clearly defined. In this position paper we introduce the concept of developer security archetypes. Specifically, we suggest two key factors: developers' personal interest in security, and the support that developers receive from their environment. Together, these two dimensions define four archetypes which can be uniquely characterized. By distinguishing developer archetypes, we seek to better understand how developers perceive security-related issues in systems development, as well as how to better support them.**

***Index Terms*—developer centred security, archetype, developer security, software security, developer.**

I. Introduction

In recent years there has been a focus on how the developer can be centred in discussions of software security. Acar et al. [1] argued that developers cannot be expected to be experts in secure coding. They suggested applying the lessons learned from twenty years of research on usable security to the context in which developers write code. Developer Centred Security (DCS) has emerged as a term for the study of how developers can be encouraged and facilitated to write secure code [2]. For the purposes of this paper we use the classic definition of "software security" from McGraw: "the idea of engineering software so that it continues to function correctly under malicious attack" [3]. We define a developer as a writer of software code for use by others.

We argue that developers are not homogenous and interchangeable, and operate at different levels of enthusiasm and expertise when it comes to security. Moreover, they operate in multiple different environments, from security-aware organisations and open source teams to very small single-developer enterprises. We posit that developers can thus be thought of as existing along a spectrum in two dimensions: their personal interest in security and secure coding, and the support that is available for secure coding in their environment. To truly put developers at centre-stage, we must examine the role of these two factors on their security stance. We maintain that this interplay between developers' interest in security and developers' environment affects what interventions are of use to individual developers.

In this position paper, we introduce a two-dimensional typology that defines four developer security archetypes. By defining these archetypes, we seek to extend our understanding of developer centred security, the interventions that are available to developers to write secure code, and which activities might support developers in this task. When considering such support activities, we can reflect on which developer archetype they primarily assist, and whether we can broaden or tweak them to assist others. We argue that the archetype most in need of support, which we call Optimists, should be specifically considered when new interventions are devised. We maintain that centring the Optimists will enhance security for all developers.

This paper proceeds as follows. In the next section we introduce the two factors which we argue play a key role in how developers approach security and secure coding. These form the two dimensions, which define the four developer security archetypes, which we discuss in Sec. III. We discuss the implications of these archetypes in Sec. IV.

II. The Role of Environment and Personal Motivation

There is a considerable body of research on developers and software security that discusses the role that the developer's environment plays in their secure coding practices. Haney et al. [4] conducted an interview study with individuals representing companies who use cryptography in their products. They found that these security-mature organisations had a security mindset, with a strong security culture and deep security expertise. Assal and Chiasson [5] conducted a survey study focusing on the human factors of software security. Using factor analysis, they found that "company-wide engagement" and "personal strategies" were the two main factors influencing participants' software security strategies. Pieczul et al. [6] discussed the developer demographics that should be considered when designing developer studies. They included "developer skill and experience" and "team, structure, culture and policies" as key demographic factors. Danilova et al. [7] considered how a company's emphasis on security influences or dictates its developers' security activities. Morales et al. [8] looked at how the cohesiveness of the development team is vital for security.

Rauf et al. [9] discussed developers as social beings and the resulting impact on their security choices. How they relate to other developers and to their context are seen as crucial influences on their security behaviour. The authors advocated further study of how context influences the security posture of solo developers such as app developers.

Developer motivation to code securely is also widely studied. In a qualitative usability study, Naiakshina et al. [10] primed ten of the 20 participating students to code securely. They found that the ten participants who were not primed made no attempt to write secure code. Primed participants did better, especially when mandated to use an API with security helper classes. In a follow-up study using freelance developers, Naiakshina et al. [11] again found that specifying security as a requirement was the main influence on the security of the resulting code. Assal and Chiasson [5] found that identifying with security importance is the most motivating factor for software security, followed by workplace environment and perceived negative consequences. Xiao et al. [12] looked at why security tools spread. They found that the largest influence is co-worker recommendations. Although they did not focus on developer motivation, they did find that an attack on a developer's software can raise the developer's security awareness; this factor is also mentioned by Assal and Chiasson [13]. Haney et al. [4] found that maturity and experience are common characteristics of developers in companies with a strong security culture.

III. The Developer Security Archetypes

Developer interest in and environmental support for secure coding, when placed on abscissa and ordinate axes, can be used to create a two-dimensional typology that defines four developer archetypes (see Fig. 1). The archetypes bring different developer circumstances into sharp focus. It is easier to assess the potential role of a proposed technique or tool when the target cohort or cohorts can be clearly identified and described.

We now position and describe the four developer archetypes. We emphasise that these are archetypes, and in reality the extent to which a developer is positioned along the two dimensions will vary. The purpose of these archetypes is not to perfectly characterize each developer, but rather to extend our understanding of different types of developers in relation to attitudes to security and secure coding.

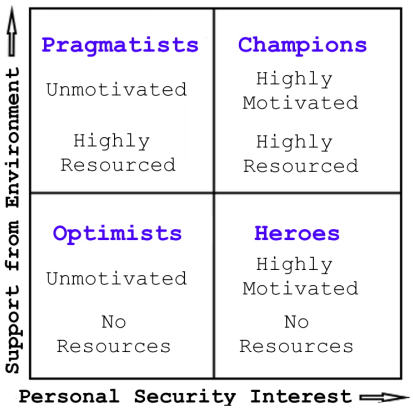

Fig. 1. The four developer security archetypes. The archetypes bring different developer influences with regard to secure coding into focus. The developer's tendency to use secure coding practices increases as we travel upwards and right along the axes.

A. Pragmatists

We adopt the term Pragmatists to describe developers who have no personal interest in security, but are provided with security-related training and tools in a collaborative environment. Although their focus is on other aspects of the work, they will follow secure coding processes if these are mandated and not too inconvenient. Xiao et al. [12], in a study of secure development tool adoption, noted that all developers in their study who were mandated to use security tools did use them. However, since Pragmatists have no personal interest in security and secure coding, they may attempt to bypass inconvenient secure coding guidelines. For example, Ashenden et al. [14] described a developer telling their company's security team that they *"are like a rock in a stream, we just flow around you."* Tomas et al. [15] also found occasional conflict, with one developer laughing about security culture "propaganda".

B. Champions

Champions are developers with an interest in security who are working in environments in which they are supported and resourced. The term "Champions" is widely used both in industry [16] and in academia [2] to describe developers with a special interest in security in the corporate environment. Security teams, which are notoriously short-staffed, often liaise with Champions and leverage their security expertise [17]. Haney et al. [4], in an interview study of experienced developers working in organisations that develop cryptographic products, encountered a population of developers many of whom were Champions. The researchers described individuals who were highly committed to security and communicated strong security values to the rest of the organisation.

C. Optimists

The "optimistic bias," which causes people to believe that misfortune will not happen to them, was used by Assal and Chiasson [13] as a possible explanation for cavalier developer attitudes to code security. In this spirit, we adopt the term Optimists to describe developers who have little or no interest in coding securely and are not supported or encouraged to do so within their environment. They are optimists because they believe that their code is probably fine, or someone else is taking care of security [18], or their software is too insignificant to be attacked, or there is nothing worth stealing in the system, or they will be gone by the time there is a breach [2], [13], [19], [20]. Optimists abound in secure development literature. While investigating the propagation of insecure code snippets from Stack Overflow, Bai et al. [21] interviewed 15 software developers about insecurities they had found in the developers' GitHub projects. Post-interview, only two of the developers fixed their projects. Of the remaining 13, one deleted their

project, one deleted their GitHub account, and the remaining developers ignored the vulnerabilities. Palombo et al. [20] were surprised by developers' reaction to a serious security flaw that potentially enabled remote code execution. Developers downplayed the issue, deciding not to fix it to avoid causing problems elsewhere in the software. Their organisation did not appear to assign any time or resources to security, or to prioritise security in any practical way.

D. Heroes

Assal and Chiasson [13] identified developer "heroes" who personally prioritise security even though they are working in environments where security is not prioritised. If working in a collaborative environment, they may provide the only security checks within a company or team. Our Hero archetype also encompasses solo developers such as app developers who take the time to familiarise themselves with the security requirements for the environments they are coding in. A developer who is highly motivated to code securely is more likely to research, discover, and implement security techniques and tools, even when not supported to do so in their environment [12]. Heroes may find themselves almost in conflict with their employers when trying to improve the security of a code base. Security researchers Palombo et al. [20] experienced this dynamic when embedded with a development team that received no security support. Their suggested security fixes were rejected until they carefully calibrated them to the code base to ensure minimal impact on the time and attention of the development team.

IV. Discussion

We now discuss implications of our typology, which helps to differentiate among different types of developers in relation to security-related attitudes.

Champions are widely regarded in industry as essential to the smooth working of the Software Security Groups that oversee secure coding in large organisations. These groups liaise with Champions on development teams to promote secure coding techniques within the teams [2] [22]. An initial analysis of the four archetypes might suggest that the solution to the problem of insecure code is to turn all developers into Champions. To achieve this, it would be necessary not only to educate all developers in secure coding techniques, but to engage their interest so that they use the techniques. Furthermore, all organisations and open source teams would have to embrace and prioritise the need for secure software development. Otherwise, Optimists would merely be converted to Heroes, who still lack much of the support needed to code securely.

Achieving these changes is a worthy objective, but until it is realised we must accept that all four developer archetypes exist and they need different types of support to help them to code securely. The existence of the four archetypes leads us to the following research questions.

A. Developer Motivation

What causes developers to move from left to right, indicating an increase in security motivation? This area comprises developer motivators such as training and experience. Witschey et al. [23] have found that information from peers can have an impact. Assal and Chiasson [5] found that the top six motivations towards software security are self-driven motivations such as caring about the user, with financial rewards considered least motivating. While drive (arguably) cannot be taught, understanding the implications of poor security scored highly. This suggests that activities such as Weir et al.'s security interventions [24] should have a motivating effect. Weir et al. devised a lightweight series of interventions to motivate teams to develop securely. Activities include incentivisation meetings with teams to introduce core security practices. Other security practices are introduced on an ad-hoc basis if appropriate to the discussion. Follow-up workshops help to copper-fasten the ideas discussed within the teams, assisting with motivation and keeping security firmly in sight.

The "Motivating Jenny" (www.motivatingjenny.org) project explores the reasons why developers do not consistently and routinely use security methods and tools. The project seeks to find ways to engage and motivate developers to code securely. It uses several different media [25] and ethnographic studies [26] to examine how developers discuss and engage with security. "Motivating Jenny" researchers have devised a workshop format to help developers to grapple with security topics [27]. Their work focuses on leveraging developers' values and professional pride to attract them to secure coding. The "Motivating Jenny" project, Weir's work and similar projects aim to interest and educate developers in security, moving them from the left to the right of our typology.

A complementary question is whether there are circumstances that cause developers to regress, moving from right to left? If so, how can these be avoided?

B. Organisational Motivation

What causes organisations, and indeed open source teams, to move upwards, indicating an increase in security motivation? Customer engagement, industry standards and legislation are frequently cited [4] [8]. Are there other motivators? Conversely, are there circumstances which cause organisations or teams to move downwards, lessening their focus on security? If so, how can these be avoided? Organisations will rarely declare their lack of interest in security, making unmotivated organisations difficult to study.

C. Supporting the Optimists

DCS research often focuses on interventions and tools for developers who are highly resourced in a security-aware environment, highly motivated to code securely, or both. We argue that special thought must be given to how to support the Optimist, who has not (yet) developed an interest in security and does not benefit from a collaborative security environment. Although it is tempting to plan to train them all, the high numbers of new entrants to software development [28] and the low barriers to entry [29] make it a Sisyphean task. Furthermore, Optimists with an interest in security become Heroes, still under-resourced for secure coding. Given that there will always be Optimists, how can they best be supported?

V. Conclusion

In this position paper we introduced four developer security archetypes, organized along the two axes of personal interest in software security and environmental support for software security. Defining these four archetypes sheds light on a blind spot in the current literature on developer centred security, and can help broaden our understanding of issues surrounding software developers and security. Whereas prior literature has started to recognize human factors such as developer motivation, this paper contributes a systematic conceptualisation of developers in relation to secure software development. Summarising, the Pragmatist is uninterested in security, but will often use security tools provided in their environment. The Champion is a security enthusiast operating in an environment where security is emphasised. Heroes work in a low-security environment, but their own interest in and drive towards security should help them to secure their code. The Optimist is uninterested in security and codes in an environment where security is not considered. Optimists are entirely dependent on the security behaviour of the tools they use to make their code secure.

We argue that our typology helps to clarify issues around secure software development. Individual developers may move between these four archetypes, whether via training, other motivators or a change of role or employer. However, the four archetypes will continue to be populated for the foreseeable future. Interventions and tools aiming to enhance secure coding should be tailored to the appropriate archetypes. Solutions that treat developers as a homogeneous group are likely to lose the opportunity to address as many developers as possible.

Acknowledgment

This publication was financially supported by Science Foundation Ireland under Grant numbers 18/CRT/6222, 13/RC/2077_P2, 13/RC/2094_P2, and 15/SIRG/3293. For the purpose of Open Access, the authors have applied a CC-BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

References

[1] Y. Acar, S. Fahl, and M. L. Mazurek, "You are not your developer, either: A research agenda for usable security and privacy research beyond end users," in *IEEE Cybersecurity Development (SecDev)*, 2016, p. 3–8.

[2] M. Tahaei and K. Vaniea, "A survey on developer-centred security," in *IEEE European Symposium on Security and Privacy Workshops*, 2019.

[3] G. McGraw, "Software security," *IEEE Security and Privacy*, vol. 2, no. 5, p. 80–83, 2004.

[4] J. Haney, M. Theofanos, Y. Acar, and S. Spickard Prettyman, "'We make it a big deal in the company': Security mindsets in organizations that develop cryptographic products," in *Proceedings of the Fourteenth Symposium on Usable Privacy and Security*, 2018, p. 357–373.

[5] H. Assal and S. Chiasson, "'Think secure from the beginning': A survey with software developers," in *CHI Conference on Human Factors in Computing Systems*, 2019.

[6] O. Pieczul, S. Foley, and M. E. Zurko, "Developer-centered security and the symmetry of ignorance," in *Workshop on New Security Paradigms*, 2017, p. 46–56.

[7] A. Danilova, A. Naiakshina, and M. Smith, "One size does not fit all: a grounded theory and online survey study of developer preferences for security warning types," in *Proc. 42nd ICSE*, 2020.

[8] J. A. Morales, T. P. Scanlon, A. Volkmann, J. Yankel, and H. Yasar, "Security impacts of sub-optimal devsecops implementations in a highly regulated environment," in *15th International Conference on Availability, Reliability and Security*, ser. ARES '20, 2020.

[9] I. Rauf, D. van der Linden, M. Levine, J. Towse, B. Nuseibeh, and A. Rashid, "Security but not for security's sake: The impact of social considerations on app developers' choices," in *42nd International Conference on Software Engineering Workshops*, 2020.

[10] A. Naiakshina, A. Danilova, C. Tiefenau, M. Herzog, S. Dechand, and M. Smith, "Why do developers get password storage wrong? a qualitative usability study," in *ACM SIGSAC Conference on Computer and Communications Security*, 2017.

[11] A. Naiakshina, A. Danilova, E. Gerlitz, E. von Zezschwitz, and M. Smith, "'If you want, I can store the encrypted password': A password-storage field study with freelance developers," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019.

[12] S. Xiao, J. Witschey, and E. Murphy-Hill, "Social influences on secure development tool adoption: Why security tools spread," in *Proc. 17th CSCW*, 2014.

[13] H. Assal and S. Chiasson, "Security in the software development lifecycle," in *14th USENIX Conference on Usable Privacy and Security*, 2018.

[14] D. Ashenden and D. Lawrence, "Security dialogues: Building better relationships between security and business," *IEEE Security Privacy*, vol. 14, no. 3, pp. 82–87, 2016.

[15] N. Tomas, J. Li, and H. Huang, "An empirical study on culture, automation, measurement, and sharing of devsecops," in *International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, 2019.

[16] M. W. Sammy Migues, John Steven, "BSIMM," https://www.bsimm.com/, 2020, [Online; accessed 17-December-2020].

[17] T. W. Thomas, M. Tabassum, B. Chu, and H. Lipford, "Security during application development: An application security expert perspective," in *CHI Conference on Human Factors in Computing Systems*, 2018.

[18] I. A. Tondel, M. G. Jaatun, and D. S. Cruzes, "IT security is from Mars, software security is from Venus," *IEEE Security Privacy*, vol. 18, no. 4, p. 48–54, Jul 2020.

[19] J. Xie, H. R. Lipford, and B. Chu, "Why do programmers make security errors?" in *2011 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2011, p. 161–164.

[20] H. Palombo, A. Z. Tabari, D. Lende, J. Ligatti, and X. Ou, "An ethnographic understanding of software (in)security and a co-creation model to improve secure software development," in *16thSymposium on Usable Privacy and Security*, 2020.

[21] W. Bai, O. Akgul, and M. L. Mazurek, "A qualitative investigation of insecure code propagation from online forums," in *2019 IEEE Cybersecurity Development (SecDev)*, 2019, p. 34–48.

[22] T. W. Thomas, M. Tabassum, B. Chu, and H. Lipford, "Security during application development: an application security expert perspective," in *CHI Conference on Human Factors in Computing Systems*, 2018.

[23] J. Witschey, S. Xiao, and E. Murphy-Hill, "Technical and personal factors influencing developers' adoption of security tools," in *ACM Workshop on Security Information Workers*. ACM Press, 2014, p. 23–26.

[24] C. Weir, I. Becker, J. Noble, L. Blair, A. Sasse, and A. Rashid, "Interventions for software security: creating a lightweight program of assurance techniques for developers," in *Proc. 41st ICSE (SEIP)*, 2019.

[25] T. Lopez, T. Tun, A. Bandara, M. Levine, B. Nuseibeh, and H. Sharp, "An anatomy of security conversations in stack overflow," in *41st ICSE (SEIS)*, 2019.

[26] T. Lopez, H. Sharp, T. Tun, A. K. Bandara, M. Levine, and B. Nuseibeh, "'Hopefully we are mostly secure': Views on secure code in professional practice," in *12th International Workshop on Cooperative and Human Aspects of Software Engineering*, 2019, p. 61–68.

[27] T. Lopez, H. Sharp, T. Tun, A. Bandara, M. Levine, and B. Nuseibeh, "Talking about security with professional developers," in *CESSER-IP*, 2019.

[28] DAXX, "How Many Software Developers Are in the US and the World?" https://www.daxx.com/blog/development-trends/number-software-developers-world, 2020, [Online; accessed 19-January-2021].

[29] "Stack Overflow Developer Survey - Education," https://insights.stackoverflow.com/survey/2020#education, 2020, [Online; accessed 19-January-2021].