Developers Need Support, Too: A Survey of Security Advice for Software Developers

Yasemin Acar, Christian Stransky,* Dominik Wermke, Charles Weir,† Michelle L. Mazurek,‡ and Sascha Fahl
*Leibniz University Hannover, *CISPA, Saarland University, †Security Lancaster, ‡University of Maryland*
{acar,wermke,fahl}@sec.uni-hannover.de; stransky@cs.uni-saarland.de; c.weir1@lancaster.ac.uk; mmazurek@umd.edu

***Abstract*—Increasingly developers are becoming aware of the importance of software security, as frequent high-profile security incidents emphasize the need for secure code. Faced with this new problem, most developers will use their normal approach: web search. But are the resulting web resources useful and effective at promoting security in practice? Recent research has identified security problems arising from Q&A resources that help with specific secure-programming problems, but the web also contains many general resources that discuss security and secure programming more broadly, and to our knowledge few if any of these have been empirically evaluated. The continuing prevalence of security bugs suggests that this guidance ecosystem is not currently working well enough: either effective guidance is not available, or it is not reaching the developers who need it. This paper takes a first step toward understanding and improving this guidance ecosystem by identifying and analyzing 19 general advice resources. The results identify important gaps in the current ecosystem and provide a basis for future work evaluating existing resources and developing new ones to fill these gaps.**

1. Introduction

The last two decades have seen explosive growth in the creation and usage of software, as we now use computers and digital devices to manage almost every aspect of our lives: to communicate, to plan, to manage our finances, to do our shopping, and to remember all our security information. Software holds sensitive information about us, controls our financial transactions, enables our personal communication and social networking, and holds the intimate details of our lives. This growth has led to a commensurate rise in the number of people working as software developers. In 1997, the Bureau of Labor Statistics estimated just over 500,000 computer programmers in the United States; by 2016, this category had been revised into four sub-categories and more than tripled, to 1.6 million employees [1].

The increasing ubiquity of software has also led to the increasing ubiquity of security bugs and associated attacks [2]. An important question, therefore, is how to help the increasing population of developers adopt effective security practices and write more secure code.[1]

Balebako et al. surveyed and interviewed more than 200 app developers and concluded most approached security issues using web search, or by consulting peers [3]. A survey by many of the current authors concluded the same, and also determined experimentally the surprising result that programmers using digital books achieved better security than those using web search [4]. Nadi et al. found that Java developers perceived cryptography APIs as too low-level and preferred more task-based solutions in documentations [5]. Further work from several of the current authors concluded that documentation is critical to security outcomes when developers use unfamiliar cryptography APIs [6].

When developers search the internet for guidance, some of the most popular results will often point to Stack Overflow.[2] Oftentimes, the developer's search will lead to a code snippet on Stack Overflow, and the developer can be tempted to copy and paste the snippet into their own code. This behaviour has been shown to often lead to operational but insecure code, widespread across hundreds of thousands of apps [4], [6], [7], [8], [9], [10]. As Stack Overflow's effect on code security has been investigated in depth, we focus our analysis elsewhere. Similarly, code completion systems found in IDEs often are not evaluated for the evolving context found in real-world situations [11].

Beyond crowdsourcing application-specific problems and documenting particular APIs, the web also contains many general resources about security and secure programming. While we would hope that these resources are helpful, to our knowledge few if any have been empirically tested for effectiveness. Moreover, the continuing prevalence of security bugs suggests that this guidance ecosystem is fundamentally broken: either effective guidance is not available (if it is even possible), or it is not reaching the developers who need it. We speculate that this web-based guidance is particularly important for developers working outside of large mainstream corporations, who do not have access to professional security teams or well-developed toolchains and frameworks supporting secure programming (e.g., Google's Tricorder [12]).

In this paper, we take a first step toward understanding and improving this guidance ecosystem. We have identified and analyzed 19 guidance websites, to understand what currently available guidance says; what it does and does not cover; and how it is structured. We found that, overall, this general-purpose guidance does not often provide concrete examples, tutorials, or exercises to help developers practice the concepts being described. We also found that while some

1. For simplicity, throughout this paper we refer to *security and privacy* as *security*.

2. https://stackoverflow.com/

978-1-5386-3467-7/17 $31.00 © 2017 IEEE
DOI 10.1109/SecDev.2017.17

critical topics like secure networking, user input validation, and software testing are well represented, other important concepts like program analysis tools, data minimization, and social engineering are rarely mentioned. These results provide a foundation for future research to fill gaps as well as to empirically evaluate existing guidance.

2. Selecting Online Resources

We collected online developer resources relevant to security by searching for variations of "developer," "security" and "guide" on various online search engines. This gives us a collection of links to online resources, some of them collections of other online resources, which we exclude from our analysis in favour of directly investigating the linked resources themselves.

We found 19 sites designed to help developers with security related questions. We excluded forum posts and Q&A platforms such as Stack Overflow, and included guides such as blog posts from authoritative sources and official guidelines from software providers and non-profit organizations such as OWASP.[3]

We focused on those types of guidelines since they carry authority for developers who feel the need to look up security-related questions and concepts. The guidelines we found covered mobile application security (Android, iOS and Windows Mobile), web security, and general secure programming. We focused on these three areas since they cover the vast majority of today's software development; additionally, mobile and web security issues have received a lot of attention from the security and privacy research community in the past.

Table 1 shows the resources we analyzed. We use the term 'handbook' to describe structured websites containing relatively large amounts of information; 'article' to describe smaller sites, mainly arranged linearly.

3. Evaluating Resources

We evaluated the 19 guides using a content-analysis-based manual coding approach [13]. We first defined a code book that listed desirable features that might be present in a secure-development guide. These features are listed as the heading to Table 4. We identified several main categories:

3.1. Source

The source category describes the author of a guide: we distinguish between official guidelines written by a framework/platform company, e.g., Google for Android or Apple for iOS, and a third-party organization (for-profit or non-profit, e.g., OWASP) providing a guide as a community service.

3. cf. https://www.owasp.org

3.2. Content Organization

This category distinguishes different features that describe the content organization of a guide. We distinguish two different types of guides: brief, single-section articles and more detailed, multi-section handbooks. We noted whether a guide contained a tutorial, defined as a step-by-step walkthrough of a specific real-world example, or any exercises to help developers become familiar with a certain security mechanism in a risk-free way. Whenever we found code examples, we checked if they were ready to use to solve certain small tasks, e.g., encrypting a file or establishing a secure HTTPS connection. We thought this to be important as ready-to-use code snippets can be copied directly into the programmer's source code without major changes. We distinguish these ready-to-use snippets from examples that present or describe specific API calls one at a time rather than in usage context. In addition, we checked whether a guide contained references to code repositories such as GitHub or BitBucket. We noted when a guide referenced external articles, blog or Wikipedia posts, or other external information sources. We noted the last time that the guide was updated as a measure of whether it was outdated or maintained. Finally, we analyzed how easy it was to find specific information in a guide: this included checking whether a guide had multiple hierarchical sections to avoid extensive scrolling, whether multiple levels of information granularity were provided (e.g., for novices, experienced programmers, and experts) and whether the guide was easily searchable.

3.3. Covered Topics

We next analyzed the different topics that were covered in the guides we examined. We started from an initial list of topics we thought were important, and added others as we encountered them in different guides. The final list of topics included:

- **Obfuscation**: Motivation for code obfuscation obfuscation techniques and tools.
- **Cryptography**: Encrypted data storage, secure key generation, etc.
- **Secure networking**: TLS/SSL, HTTPS, etc.
- **Storage management**: Backups, secure deletion, and proper management of shared storage.
- **Privileges**: Responsible use of privileges, principle of least privilege.
- **User input**: Proper validation of user input to avoid, e.g., SQL injection, cross-site scripting, and memory errors.
- **Use of tools**: Using of automated security-test tools including linters and other program-analysis techniques.
- **Mobile security**: Mobile-specific topics including privacy implications of mobile device tracking and geolocation.
- **Library use**: Using trusted libraries to avoid reinventing the security wheel, validating that selected libraries do not introduce malicious behavior.

ID	Title	Organization	Description
1.	Safeguard your code: 17 security tips for developers	InfoWorld, an online magazine for IT and business professionals	Article dated 2013
2.	Best Practices for Security & Privacy	Google	Online Android training materials from Google.
3.	Secure Coding Practice Guidelines	UC Berkeley	Guidance on ways to satisfy application software security requirements, mainly in the form of links to other resources.
4.	Start with Security: A Guide for Business	US Government Federal Trade Commission	Paper with guidelines on corporate IS security
5.	Android Secure Coding Standard	Software Engineering Institute, CMU	Wiki with extensive guidelines.
6.	Mobile Application Security: 15 Best Practices for App Developers	Checkmarx, a development tool vendor	Short blog article
7.	Top 10 Secure Coding Practices	Software Engineering Institute, CMU	Short summary of 10 general secure coding principles
8.	Secure Coding Guide	Apple	Extensive online handbook covering technical aspects of iOS security
9.	Secure Mobile Development Guide	NowSecure, company specializing in app security testing and support	Online handbook covering many aspects of mobile app security.
10.	Secure Coding Practices Quick Reference Guide Project	OWASP, a not-for-profit dedicated to application security	Checklist of around 100 short bullet-point entries around general coding
11.	Secure Coding Cheat Sheet	OWASP	More detailed handbook describing principles of general coding.
12.	Security Guidance for Applications	Microsoft	Handbook with security guidance for web applications (outdated)
13.	Security Checklist for Software Developers	CERN, a research organization	Site with general guidelines and tips for specific languages.
14.	Website security	Mozilla	Extensive training article on web application security.
15.	Web Fundamentals: Security and Identity	Google	Several tips on web app security, with emphasis on Google tools.
16.	Developer How To Guide	SANS, a not-for-profit specializing in software security training	Article on how to avoid three common web security vulnerabilities.
17.	8 Tips for Better Mobile Application Security	UpWork, a developer recruitment site	Blog article
18.	TOP 25 Most Dangerous Software Errors	SANS with MITRE, a non-profit research company	Handbook, exploring types of security errors
19.	Intro to secure Windows app development	Microsoft	Extensive handbook to secure programming in the MS Windows environment.

TABLE 1. THE 19 SECURITY GUIDES THAT WE IDENTIFIED AND ANALYZED.

- **Testing**: Measures for examining finished or deployed systems, including code review, fuzzing and penetration testing.
- **Data minimization**: Limiting the collection, storage, and transmission of personal information to protect users' privacy.
- **Regulations**: Security and privacy laws and regulations in various jurisdictions.
- **Threat modeling**: Design-level analysis of security requirements.
- **Logging**: Keeping records to enable post-hoc auditing.
- **Password advice**: Guidelines for the secure creation and storage of passwords (e.g., salting and hashing stored passwords).
- **Social engineering**: Tricking people into making security errors, such as giving away secret data improperly.

4. Results and Discussion

Table 4 provides an overview of our results.

Sources and Organization. We found that the majority (>55%) of the resources we analyzed were written by companies. All of these companies are in some way involved in secure software development or benefit from it. In 16% of cases, the guides were published as part of the official developer documentation written by the vendor of a development framework such as Apple for iOS, Google for Android, and Microsoft for Windows Mobile. Interestingly, about a third of all guides were written by non-profit organizations like OWASP.

Most of the guides for which we could identify update times (10 of 15) were last updated within the last two years; one (Microsoft web-application guidance, ID 12) could be considered entirely obsolete, as it dated from 2003. These findings suggest that when developers search for secure-programming resources, they will frequently but not always encounter up-to-date ones.

Overall, these resources did not tend to contain concrete, low-level guidance. We found only five guides (IDs 5, 12, 15, 16, 19) that contained ready-to-use code snippets; most

			Content Organization										Covered Topics																
Resource ID	Source	Last Update[1]	Type[2]	Ready to use code	API examples	Exercise	Tutorials	Repository	Citations	Layered Information	Searchable	Sectioned	Obfuscation	Cryptography	Secure Networking	Storage Management	Privileges	User Input	Use of Tools	Mobile Security	Library Use	Testing	Data Minimization	Regulations	Threat Modelling	Logging	Password Advice	Social Engineering
ID 1 (InfoWorld)	third-party	2013-02-04	a	○	○	○	○	○	○	○	●	○	○	●	○	○	○	●	○	○	○	○	●	○	●	○	●	○
ID 2 (Google Android)	vendor	?	h	○	●	○	○	○	○	●	●	○	○	●	●	●	●	●	○	●	○	○	○	○	●	○	●	○
ID 3 (UC Berkeley)	organization	?	a	○	○	○	○	○	●	●	●	●	○	●	●	●	●	●	●	○	●	●	○	○	●	○	●	○
ID 4 (US FTC)	organization	2015-06	a	○	○	○	○	○	○	●	●	●	○	○	●	●	●	○	○	●	○	○	●	○	●	●	●	○
ID 5 (SEI, CMU)	organization	2016-07-11	h	●	○	○	○	○	○	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○	●	●	○	○
ID 6 (Checkmarx)	third-party	2015-08-19	a	○	○	○	○	○	○	●	●	●	○	●	●	○	●	○	●	●	○	●	●	●	○	○	○	●
ID 7 (SEI, CMU, Top 10)	organization	2011-03-01	a	○	○	○	○	○	○	●	●	●	○	○	○	○	●	●	●	○	○	●	○	○	●	○	○	○
ID 8 (Apple)	vendor	2016-09-13	h	○	●	○	○	○	○	●	●	○	○	●	●	●	●	●	●	●	●	●	○	○	○	○	○	●
ID 9 (NowSecure)	third-party	2017-03-05	h	○	●	○	○	○	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	●	○	●
ID 10 (OWASP, quick reference)	non-profit	2010-08-11	h	○	○	○	○	○	○	○	●	○	○	●	●	●	●	●	○	○	●	●	●	○	●	●	●	○
ID 11 (OWASP, cheat sheet)	non-profit	2017-04-18	h	○	○	○	○	○	●	●	●	○	○	●	●	○	●	●	○	○	●	●	●	○	○	●	●	○
ID 12 (Microsoft)	vendor	2003-07-01	h	●	○	○	○	○	●	●	●	○	○	●	●	●	●	●	○	○	○	●	●	○	●	●	○	○
ID 13 (CERN)	organization	?	a	○	○	○	○	○	●	●	●	○	○	●	○	○	●	●	●	○	●	●	○	○	●	●	○	○
ID 14 (Mozilla)	non-profit	2017-05-24	h	○	○	○	○	○	○	●	●	○	○	○	●	○	○	●	○	○	○	○	●	○	○	○	●	○
ID 15 (Google)	vendor	2017-05-22	h	●	○	○	○	○	●	●	●	○	○	○	●	○	○	○	○	○	○	○	○	○	○	○	○	●
ID 16 (SANS)	non-profit	?	h	●	○	○	○	○	○	●	●	○	○	○	●	○	○	●	○	○	○	○	○	○	○	○	○	○
ID 17 (UpWork)	third-party	2017-01-15	a	○	○	○	○	○	○	●	○	○	●	●	●	○	○	○	○	○	○	●	○	○	○	○	○	○
ID 18 (SANS, MITRE)	non-profit	2011-07-27	h	○	○	○	○	○	○	●	●	●	○	●	●	○	●	●	○	○	○	○	○	○	○	○	○	○
ID 19 (Microsoft)	vendor	2017-02-08	h	●	○	○	●	●	●	●	●	●	○	●	●	●	●	○	○	●	○	○	○	●	○	○	○	○

[1] ?: date not specified [2] a: short article, h: detailed handbook

TABLE 2. FEATURES OF THE GUIDES, AS DETERMINED BY CONTENT ANALYSIS. A SHADED CIRCLE (●) INDICATES A GUIDE EXHIBITS THIS FEATURE; AN EMPTY CIRCLE (○) INDICATES IT DOES NOT.

of these offered help on secure network connections using TLS. An additional three guides included API examples that do not rise to the level of snippets: Google's Android security and privacy guide (ID 2) exhibits the Android cryptography API, Apple's Secure Coding Guide (ID 8) gives examples for platform-specific security mechanisms, and NowSecure (ID 9) contains negative examples that demonstrate how not to use their SSL API. Interestingly, all these API examples come from mobile application guides.

Sadly, none of the guidelines we checked contained exercises that would have helped developers to learn security APIs or frameworks. Similarly, only the Microsoft Windows handbook (ID 19) contained a tutorial introducing developers to secure web programming techniques or referenced an external GitHub code repository with example code demonstrating specific security features. Only six guides referenced external information sources; a lack of such citations potentially undermines a reader's confidence in the guide's accuracy and inhibits further reading and learning. Overall, these results provide evidence of an important guidance gap noted in our prior work [4]: official documents and corporate guidelines do not provide the same level of detail and focus on utility as, for example, Q&A sites.

On the positive side, we were heartened to observe that most guides provide layers of advice that can target readers with different skill levels and are easily searchable.

Coverage of Topics. We found significant variation in coverage across topics. Some important topics are well covered: cryptography, secure networking, privilege management, and user input validation were all covered in at least 14 guides, and testing was covered in 10. On the other hand, there are several potentially critical topics with little to no coverage. Social engineering, which is a source of many security horror stories from phishing to recklessness with USB drives, is covered by only four guides. Logging, which is critical for being able to audit a system and understand any potential problems, is covered by only seven. Despite years of intensive research and commercial development creating and improving program-analysis tools, these tools are also only mentioned in seven guides. Another topic mentioned in only seven guides is data minimization, a critical modern concept in a world of exploding data mining. Some of the guides we analyzed may be too old to recognize the critical importance of data minimization; nonetheless it is concerning that developers searching for security help may not encounter it. Using and validating trusted libraries is an important and well-regarded practice, but it is mentioned by only six guides, perhaps because it is assumed to be implied. Finally, we note that laws and regulations are mentioned in only two guides. One might assume this is an issue for lawyers and executives rather than for software developers, but independent developers and small companies (the kind of developers most likely to be searching for guidance on the internet rather than talking to a company's dedicated security team) may not have separate compliance departments either. Further, developers who have some knowledge of legal requirements are less likely to make accidental errors that

violate regulations.

We speculate that these coverage limitations arise from the nature of the authors creating these guides. The coverage suggests a predominantly traditional approach to security, based around technical support for developers. Looking at the results, we may conclude that in future better results may come from having a cross discipline team create the documentation, with representation from test, support, product management, and legal experts.

5. Conclusion

Our brief survey of general security guidance available on the web provides some insight into what developers—especially those without formal security training and/or without corporate security support—may encounter when they search for information about how to write secure code. They will find accessible information, appropriately layered and searchable, with good coverage of cryptography, secure networking and the handling of user input and privileges. However there are significant areas of concern: some readily available advice is outdated; most of this general-purpose guidance does not provide concrete examples or exercises; and some critical topics like program analysis tools, logging/auditing, and data minimization are not well represented. To remedy this would require a rather different team of authors from traditional security writers: pedagogical experts to generate exercises and tutorials, and human-centred security experts and legal experts to deal with social engineering and regulations.

Our prior work found that "official" guidance (from Google and from books) could promote stronger security outcomes than community-based guidance from Stack Overflow [4]. The results of this survey, however, underscore another conclusion from that work: these "official" documents may not necessarily provide the content and format that developers want or need in practice.

In this work, we take the preliminary step of identifying and classifying a multitude of information sources. Further work is needed to assess their quality, and to understand how developers use these guides as well as how to increase their effectiveness. Therefore, the question of how best to organize online security help – guides, crowdsourcing sites, aggregators that combine various sources of security advice – as well as how to ensure that the quality of such advice remains high – remains open.

Thus, our results suggest two paths for understanding and improving the security-guidance ecosystem. First, we must examine whether and which of the gaps we have identified here—both in content and in format—represent serious omissions, and which are filled by other resources outside the scope of this paper. Second, we must continue to empirically evaluate the existing guidance, to understand which approaches do and don't prove to be effective and why. By understanding what current guidance is missing, where it succeeds, and where it fails, we can hope to provide a framework for developing better guidance, both now and as secure programming continues to evolve.

References

[1] Bureau of Labor Statistics, "Occupational Employment Statistics," 2016. [Online]. Available: https://www.bls.gov/oes/tables.htm

[2] S. Morgan, "Top 2016 Cybersecurity Reports," 2016. [Online]. Available: https://www.forbes.com/sites/stevemorgan/2016/05/09/top-2016-cybersecurity-reports-out-from-att-cisco-dell-google-ibm-mcafee-symantec-and-verizon/#2c8abdb51caf

[3] R. Balebako and L. Cranor, "Improving App Privacy: Nudging App Developers to Protect User Privacy," *IEEE Security & Privacy*, vol. 12, no. 4, pp. 55–58, 2014.

[4] Y. Acar, M. Backes, S. Fahl, D. Kim, M. L. Mazurek, and C. Stransky, "You Get Where You're Looking For: The Impact of Information Sources on Code Security," in *Proc. 37th IEEE Symposium on Security and Privacy (SP'16)*. IEEE, 2016.

[5] S. Nadi, S. Krüger, M. Mezini, and E. Bodden, ""Jumping Through Hoops": Why do Java Developers Struggle With Cryptography APIs?" in *Proc. 37th IEEE/ACM International Conference on Software Engineering (ICSE'15)*. IEEE, 2016.

[6] Y. Acar, M. Backes, S. Fahl, S. Garfinkel, D. Kim, M. L. Mazurek, and C. Stransky, "Comparing the Usability of Cryptographic APIs," in *Proc. 38th IEEE Symposium on Security and Privacy (SP'17)*. IEEE, 2017.

[7] J. Xie, H. R. Lipford, and B. Chu, "Why do programmers make security errors?" in *Proc. 2011 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'11)*. IEEE, 2011.

[8] S. Fahl, M. Harbach, H. Perl, M. Koetter, and M. Smith, "Rethinking SSL Development in an Appified World," in *Proc. 20th ACM Conference on Computer and Communication Security (CCS'13)*. ACM, 2013.

[9] S. Fahl, M. Harbach, Y. Acar, and M. Smith, "On The Ecological Validity of a Password Study," in *Proc. 9th Symposium on Usable Privacy and Security (SOUPS'13)*. USENIX Association, 2013.

[10] F. Fischer, K. Böttinger, H. Xiao, C. Stransky, Y. Acar, M. Backes, and S. Fahl, "Stack Overflow Considered Harmful? The Impact of Copy&Paste on Android Application Security," in *Proc. 38th IEEE Symposium on Security and Privacy (SP'17)*. IEEE, 2017.

[11] S. Proksch, S. Amann, S. Nadi, and M. Mezini, "Evaluating the Evaluations of Code Recommender Systems: A Reality Check," in *Proc. 31st IEEE/ACM International Conference on Automated Software Engineering (ASE'16)*. ACM, 2016.

[12] C. Sadowski, J. v. Gogh, C. Jaspan, E. Sderberg, and C. Winter, "Tricorder: Building a Program Analysis Ecosystem," in *Proc. 37th IEEE/ACM International Conference on Software Engineering (ICSE'15)*. IEEE, 2015.

[13] K. Krippendorff, *Content Analysis: An Introduction to Its Methodology (2nd ed.)*. SAGE Publications, 2004.