Jit

Report

# Survey: What Developers Think About Security in 2025 – and Why it Matters

Dive into the survey data to better understand the challenges and opportunities for engaging developers in your AppSec program and how to enable them to write secure code, faster.

The critical role that developers play in securing applications is widely recognized by many application security teams – who are often outnumbered by developers 50:1 or even 200:1. According to the “shift left” approach, empowering developers to write more secure code is the most scalable path to minimizing product security risk.

For this reason, there has been considerable investment in driving developer buy-in and engagement when designing application security programs. This raises the question: what do developers need to deliver more secure code?

Jit launched a survey of 150 developers across industries and company sizes to begin answering this question.

## Why did we run this survey?

This survey analysis explores the challenges developers face when trying to write secure code, how they perceive their role in securing applications, and the support they need from their organizations to succeed. By asking developers directly about their experiences, we aimed to shed light on areas that can be easily overlooked when designing application security programs, such as:

- How developers balance security with competing priorities like feature delivery.
- The tools and practices they find most effective in securing code.
- The organizational dynamics that influence their ability to engage with security tasks.

> With a deeper understanding of the developer’s perspective on efficiently mitigating security risk, application security teams can make informed decisions when designing a program to enable developers to deliver more secure code.

While the challenges and needs of security professionals are well-researched, this report aims to build a clearer understanding of how developers experience application security and to foster better alignment between their needs and organizational security goals. By focusing on this often-overlooked perspective, we hope to contribute to a broader conversation about how to empower developers as key stakeholders in securing applications.

# Key takeaways

## Security is not a widespread cultural priority among development teams

61% of participants indicated that security is not a priority or “somewhat important” to their development culture. This is consistent with the finding that a "Lack of organizational priority" was ranking as one of the top challenges to application security. That said, developers who cited strong collaboration with security teams were more likely to indicate a stronger security culture among their teams and higher confidence in securing their code.

## Application complexity & organizational obstacles rank as top challenges to code security

The term “complexity” was perceived in different ways, ranging from the variety of technologies in modern environments to large dependency trees. The top organizational obstacles included:

1. Lack of knowledge, training, and guidelines
2. Lack of organizational priority
3. Lack of time

*Note: these organizational challenges were all nearly the top challenge to application security, with just .11 average ranking separating them from the top challenge.

## Automated security testing is the preferred approach to code security, but poor integrations and noisy results inhibit progress.

Developers believe that implementing automated security testing (via tools like SAST, SCA, and secrets detection) is the most effective strategy to secure applications – but their integration into development workflows and noisy results were often cited as core challenges to code security. 53% of participants indicated that they had access to automated security tools, making it the most common strategy to improve code security.

## Developers are not confident in AI to help them secure their code compared to other resources.

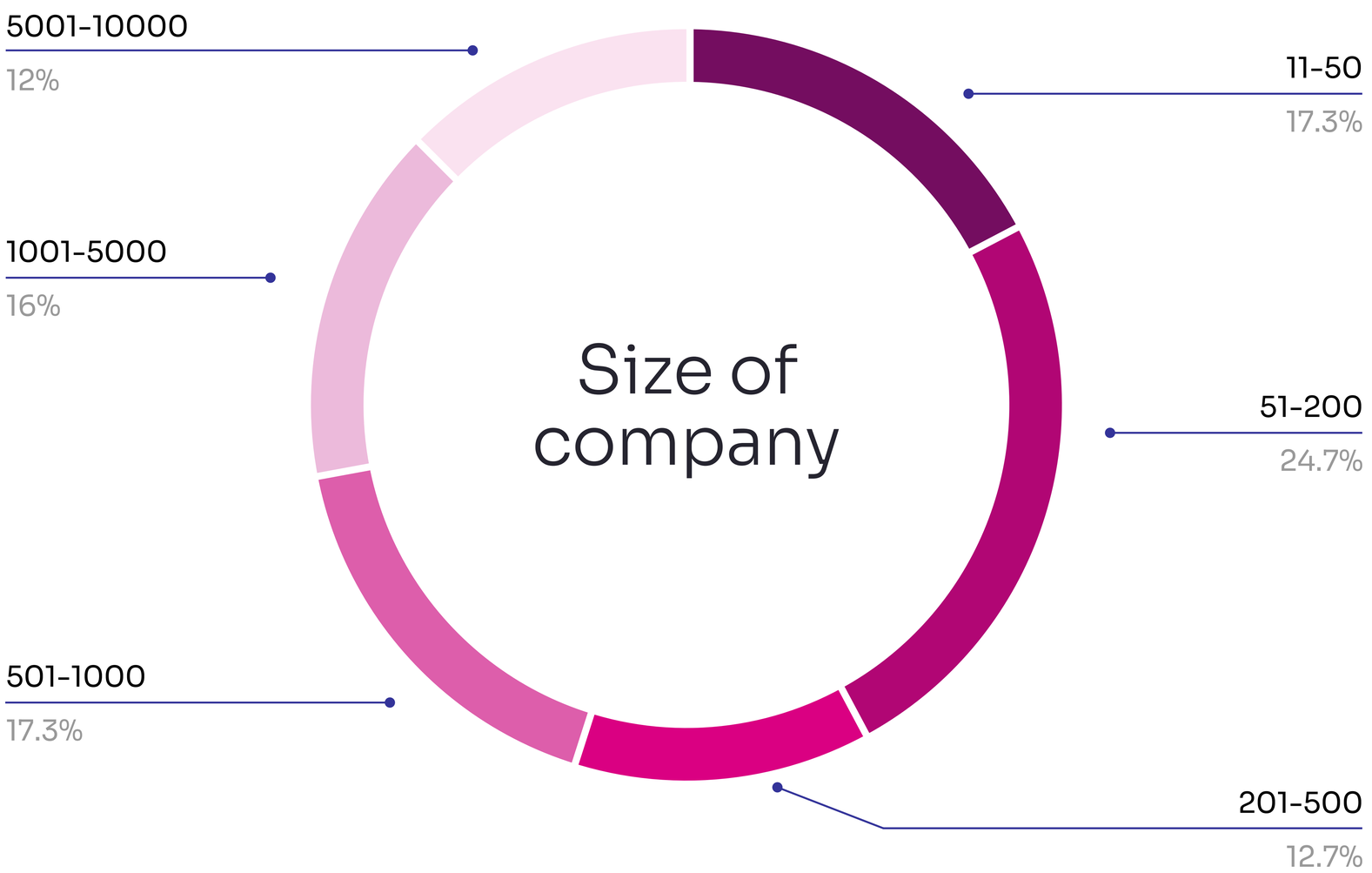
Coding assistants like Copilot were ranked as the least effective strategy for securing code, and AI was cited as the second least common source to answer questions related to code security.

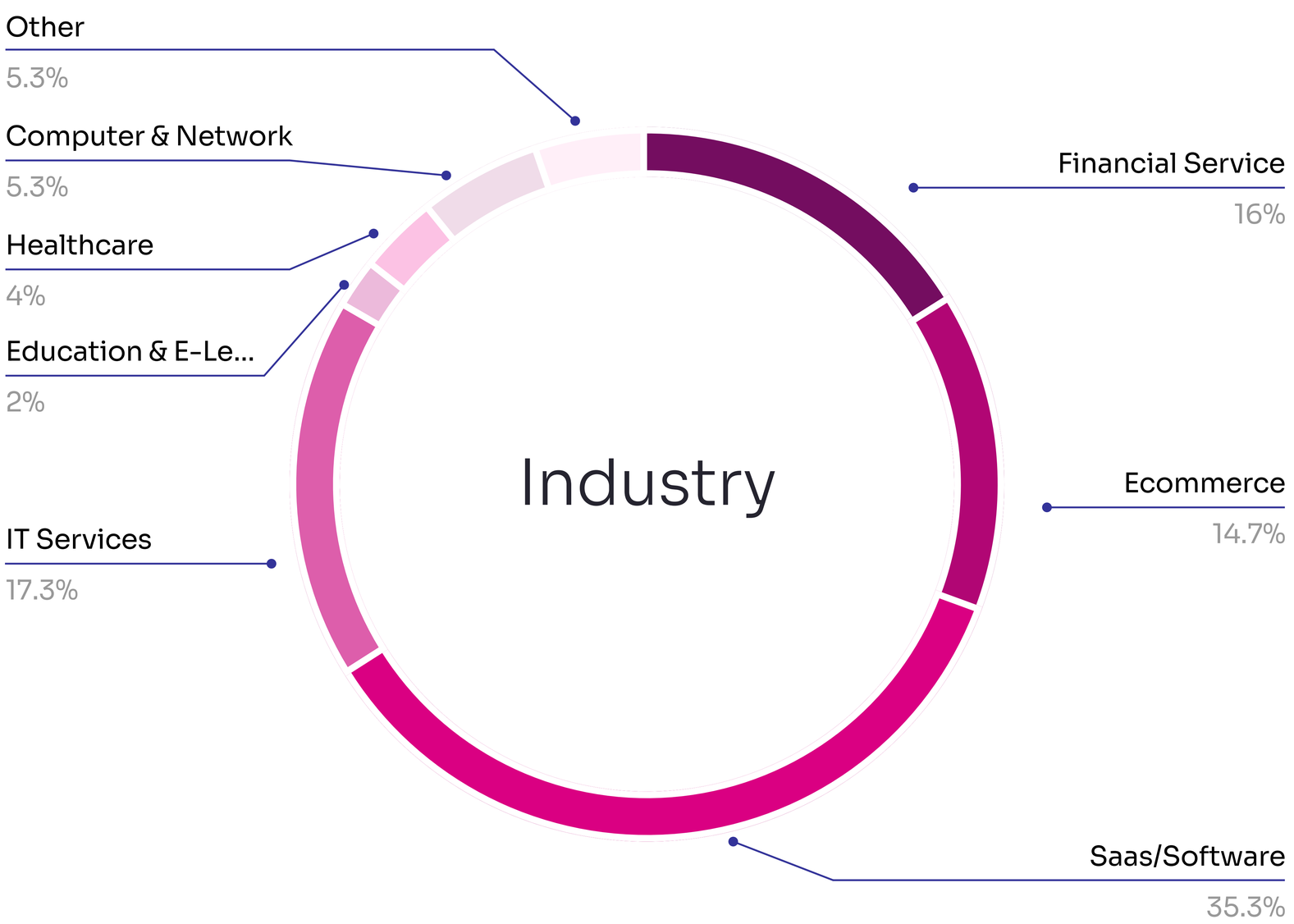
## Online documentation (e.g., OWASP, vendor documentation) is the most common place to answer questions related to code security, but it may not be sufficient.

Of those who selected “Lack of knowledge, training, and guidelines” as their top challenge to application security, “Online documentation” was the most common resource to answer questions related to code security.

# Who did we survey?

Jit surveyed 150 developers from an array of industries and company sizes.



# Survey results and analysis

## 1 What are the biggest challenges you face when it comes to securing your code? Please rank from greatest to least impact.

| Rank | Capabilities | Average rank score (Higher # = more challenging) |
|---|---|---|
| 1 | Complexity of modern app architectures | 4.79 |
| 2 | Lack of knowledge, training, and guidelines | 4.74 |
| 3 | Lack of organizational priority | 4.71 |
| 4 | Lack of time | 4.68 |
| 5 | Lack of automated security tooling | 4.06 |
| 6 | Noisy alerts and high false positive rates from scanners | 3.69 |
| 7 | Other (please specify in the next question) | 1.33 |

The “Complexity of modern app architectures” was cited as the top challenge by developers when addressing code security, but not by much. The next three top ranked challenges are all related to lacking organizational resources and priority, including: “Lack of knowledge, training, guidelines”, “Lack of organizational priority”, and “Lack of time” were nearly just as common challenges to code security.

Interestingly, there were varying interpretations of “app architecture complexity” when describing the challenges of code security. In question two, we asked participants to elaborate on their question to answer one, which described the security challenges of app complexity in different ways:

### Understanding security nuances for a wide array of technologies

“Our current architecture consists of many microservices in the cloud, integrated with many other third-party external components, with different configurations of internal kubernetes cluster access policies and external access policies. All these are prone to mistakes, making the security challenge harder.”

### Understanding the managing the security of interactions across services

“With the use of modern software and services relying on each other to validate users and permissions, any assumptions made by callee services that are not already fulfilled by caller services may result in unauthorized access to sensitive data.”

### Managing known vulnerabilities within dependency chains

“[We use] Node.js projects where we rely on the npm ecosystem. The complex, interconnected dependency chain creates a very large threat surface and it's almost impossible to meaningfully vet all of the transitive dependencies.”

To summarize the findings, developers believe application security is becoming harder as architectures grow more complex, and the limited resources and time constraints in most companies prevent developer enablement to overcome that complexity. This is summed up nicely by one participant:

> With a deeper understanding of the developer’s perspective on efficiently mitigating security risk, application security teams can make informed decisions when designing a program to enable developers to deliver more secure code.

## 2 Please elaborate on the answer you provided in the previous question. Why does the top challenge make code security harder?

This is an open ended question where developers could describe the top challenges they faced when securing their code. Here are the most common themes:

### Organizational and Resource Prioritization

Developers consistently cite a lack of organizational prioritization for security, with tight deadlines and a focus on feature delivery leaving little room for proactive security efforts. Limited time, budget, and leadership support compound these challenges, making it difficult to allocate resources toward security improvements. Here are some direct quotes from participants on this topic:

> “The biggest issue was that [the product team] is always trying to get features out so we don't really focus on security unless we are under the gun to be compliant.”

> “My organization prioritizes shipping as fast as possible over security. PM/EM is constantly pushing for things to get shipped faster, and our performance reviews are heavily weighted towards shipping something shiny (and not making sure it's secure).”

> “Security isn't an issue because it doesn't make money, development of products, features, etc instead take priority. Security only because an issue if it somewhere affects a sale, e.g. client requirement, government regulation, etc.”

### Complexity and Knowledge Gaps

Automated security tools are frequently criticized for generating excessive false positives, being difficult to integrate into workflows, and lacking the contextual understanding needed to address nuanced vulnerabilities. Developers need tools that are actionable, easy to use, and seamlessly embedded in their existing workflows, as well as visibility into their application's overall security posture.

Check out the quotes listed in question one to see the different ways participants interpreted “The complexity of modern app architectures”.

### Tooling and Process Challenges

The complexity of modern software architectures—distributed systems, microservices, and dependency chains—creates a vast and difficult-to-manage attack surface. Developers often lack the specialized training needed to identify and address security vulnerabilities effectively, especially in secure architecture and advanced threat mitigation.

Here are some direct quotes from participants on this topic:

> “We spend far too much time and resources digging into false positives rather than fixing actual security concerns.  Other than this being organizational waste, it erodes confidence in our scanning tools so that actual concerns aren't dealt with in a timely manner.”

> “It's quite hard to have good and in-depth automation tools. Many of them produce a lot of false positives and alerts that the security team has to sift through. Also, they might miss more subtle, context-specific issues that a human could catch.”

## 3 What do you believe are the most impactful strategies to secure your code? Please rank from most to least important.

| Rank | Capabilities | Average rank score (Higher # = more challenging) |
|---|---|---|
| 1 | Automated testing (SAST, SCA, Secrets detection) in the CI/CD pipeline or IDE | 5.21 |
| 2 | Utilizing security frameworks and libraries | 4.88 |
| 3 | Security training and awareness programs | 4.51 |
| 4 | Manual code reviews focused on security | 4.09 |
| 5 | Building internal developer champions for code security | 3.70 |
| 6 | Maintaining an internal wiki on code security best practices | 3.25 |
| 7 | Using a code assistant like Copilot | 2.35 |

The top ranked strategies for code security – automated testing and security frameworks/libraries – suggest that developers would prefer to secure their code as they go, rather than building security skills and knowledge through educational resources or documentation.

### Automation testing tools are the preferred solutions

The developer participants indicated automated testing tools (SAST, SCA, secrets detection) as the clear favorite strategy for securing code. This is likely related to the fact that 52% of surveyed developers already have automating security tooling in place (indicated in Question 4) – making it the most popular implemented solution for code security.

One participant indicated time-savings as the reason he preferred the technologies, which is consistent with other survey findings that suggest limited time:

> “Automation is key for implementing code security—developers often don’t have the time or resources to prioritize code security.”

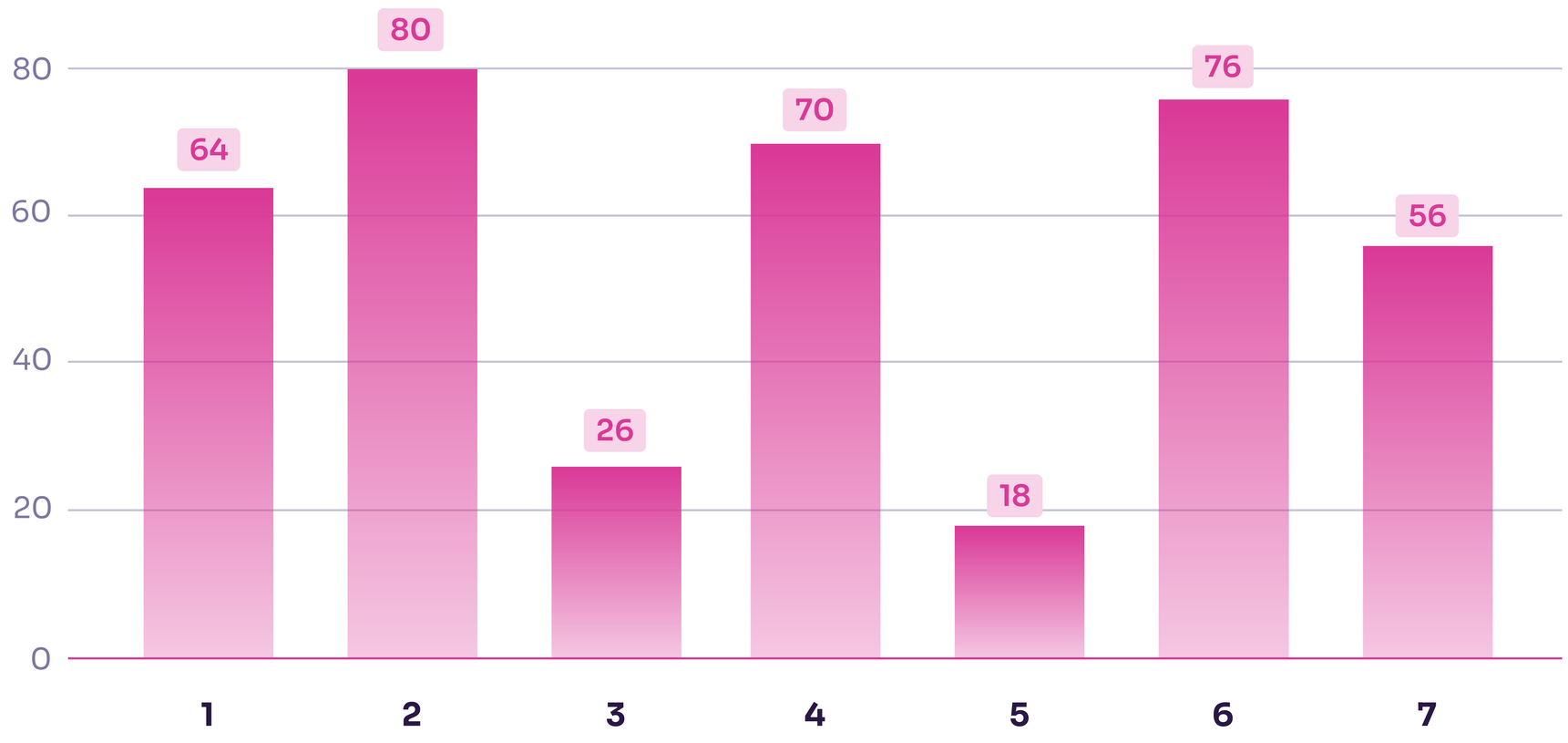
Another participant highlighted the depth of scanning analysis as an advantage:

> “Devs find ways to make short cuts that can lead to code that can be easily compromised. Automated tooling can detect these short cuts or mistakes a lot better than a human looking over a PR can.”

**Code assistants were ranked as the least impactful code security strategy**

While tools like these can enhance developer productivity, their role in code security appears to be limited. Developers may not trust AI for security tasks.

## 4 How does your company support you in building secure applications (select all that apply)?




1. Regular code security training
2. Implemented security scanners (e.g. SAST, SCA, secrets detection)
3. Building security champions within development teams
4. Dedicated security team detects vulnerabilities and triages to developers
5. No Support
6. Code reviews that focus (at least partially) on security
7. By maintaining internal docs on code security best practices

For this question, participants were directed to select all methodologies their company implements to support them in building secure applications.

### Continuous code security analysis is preferred over domain knowledge

According to these results, companies prefer continuous code analysis methodologies rather than approaches focused on expanding developers' knowledge or skills. This is consistent with developer preferences indicated in the previous question. The most common forms of support include:

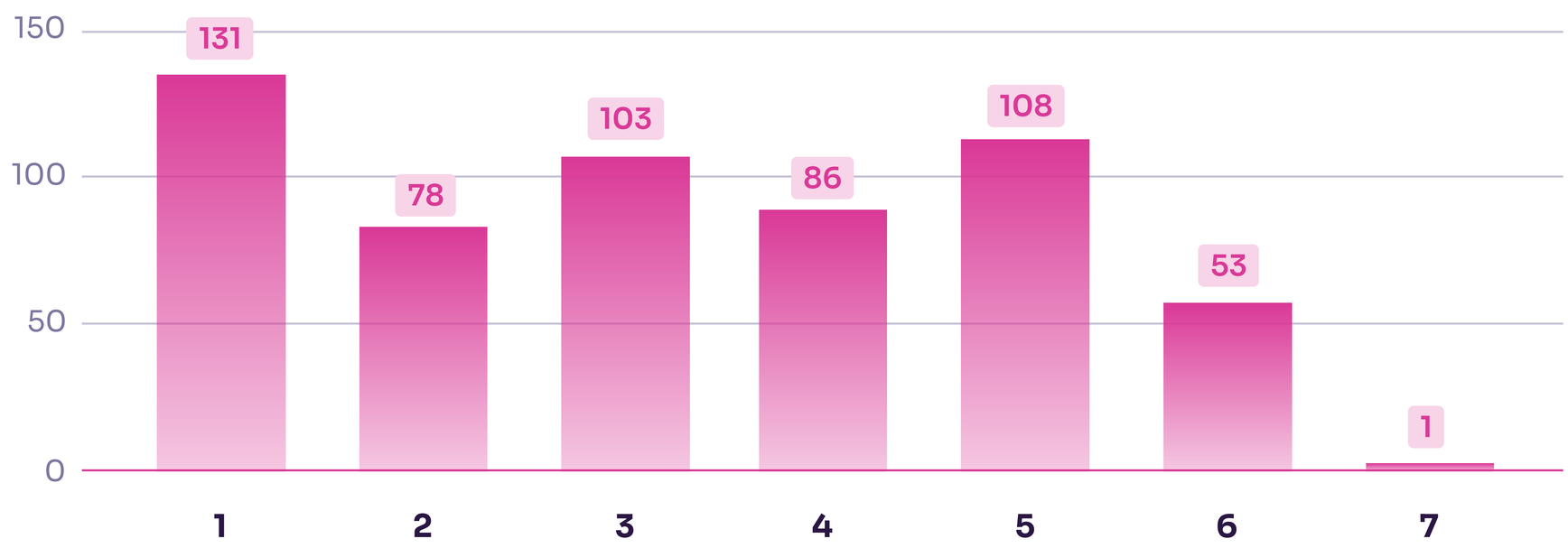
- Implementing security scanners like SAST and SCA (80 participants)
- Conducting code reviews with a security focus (76 participants)
- Involving dedicated security teams to detect vulnerabilities and triage issues (70 participants)

While these strategies are effective for directly addressing vulnerabilities in the code, they aren't necessarily designed to equip developers with the deeper knowledge or training needed to independently secure their applications. This is reflected in the lower numbers for knowledge-driven initiatives, such as regular training (64 participants), internal documentation (56 participants), and building security champions (26 participants).

This divide aligns with findings elsewhere in the survey. Developers ranked lack of knowledge, training, and guidelines as the second biggest challenge to code security (Question 1). One respondent noted:

> "Lack of knowledge and training is the top challenge because only security engineers really know what to look for and what to do. As a software engineer, I don't have expertise to be able to enforce and ensure code security."

### 5 Where do you go to answer code security questions (select all that apply)?




1. Online documentation (e.g., OWASP, vendor documentation)
2. ChatGPT / AI
3. Forums, blogs, and communities (e.g., Stack Overflow, Reddit)
4. Internal resources (e.g., company wiki, internal docs)
5. Colleagues or peers (internal security champions or security teams)
6. Training courses or workshops
7. No resources to answer code security questions

The survey shows that developers rely heavily on external resources for code security guidance. The most frequently used resource is “Online documentation” like OWASP and vendor documentation (131 participants), and the third most frequent being “Forums, blogs, and communities” like StackOverflow and Reddit (103 participants).

Colleagues or peers (108 participants) was the most frequently cited resource for internal resources, while wikis and internal docs (86 participants) and training courses (53 participants) are less commonly used.

Notably, ChatGPT/AI tools was the third least used resource for code security among participants. This is consistent with findings from Question 3, which cited coding assistants like Copilot as the least effective strategy for code security.

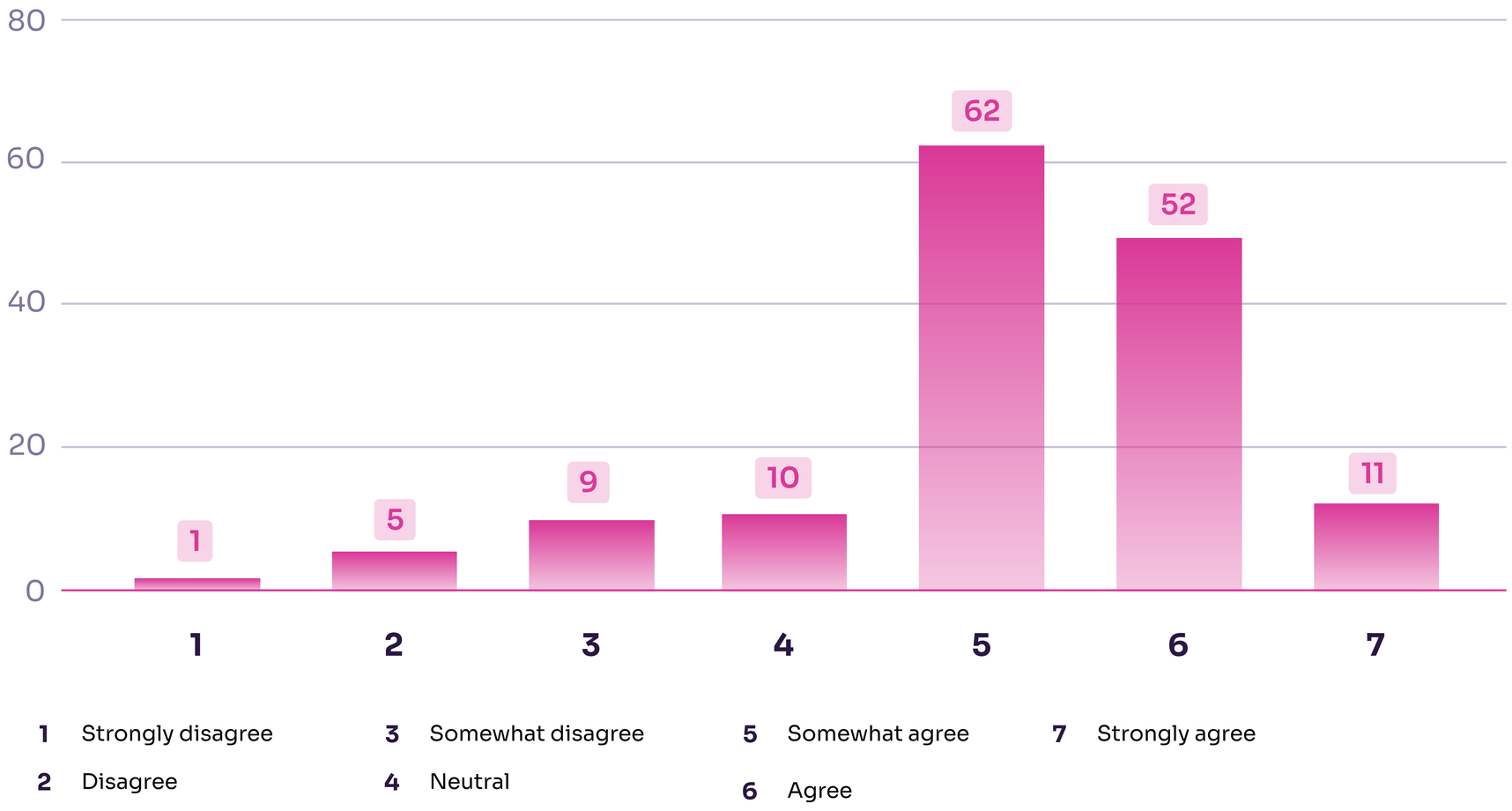
### Online resources are likely not enough to overcome knowledge gaps

Of those who selected “Lack of knowledge, training, and guidelines” as their top challenge to application security, the most common source to answer code security-related questions were “Online documentation”, and the least common was “Training courses or workshops” (aside from those who listed “No resources”)

| Where do you go to answer questions about code security? | Selected “Lack of knowledge, training & guidelines” as top AppSec challenge |
|---|---|
| Online documentation (e.g., OWASP, vendor documentation) | 22 |
| ChatGPT / AI | 13 |
| Forums, blogs, and communities (e.g., Stack Overflow, Reddit) | 18 |
| Internal resources (e.g., company wiki, internal docs) | 16 |
| Colleagues or peers (internal security champions or security teams) | 19 |
| Training courses or workshops | 6 |
| No resources to answer code security questions | 0 |

This could suggest that online resources alone are not enough for developers to overcome knowledge gaps in application security, and that training courses and workshops are a more effective strategy to boost domain knowledge.

## 6 On a scale of 1-7, how strongly do you agree or disagree with the following statement? "I can consistently and independently deliver secure code."




76% of developers express moderate confidence in their ability to consistently and independently deliver secure code, with 62 participants selecting "somewhat agree" and 52 selecting "agree." Only 7% of participants strongly agree, while only 10% participants (combining scores 1–3) explicitly disagree to varying degrees.

The average rating was 5.18 among all participants.

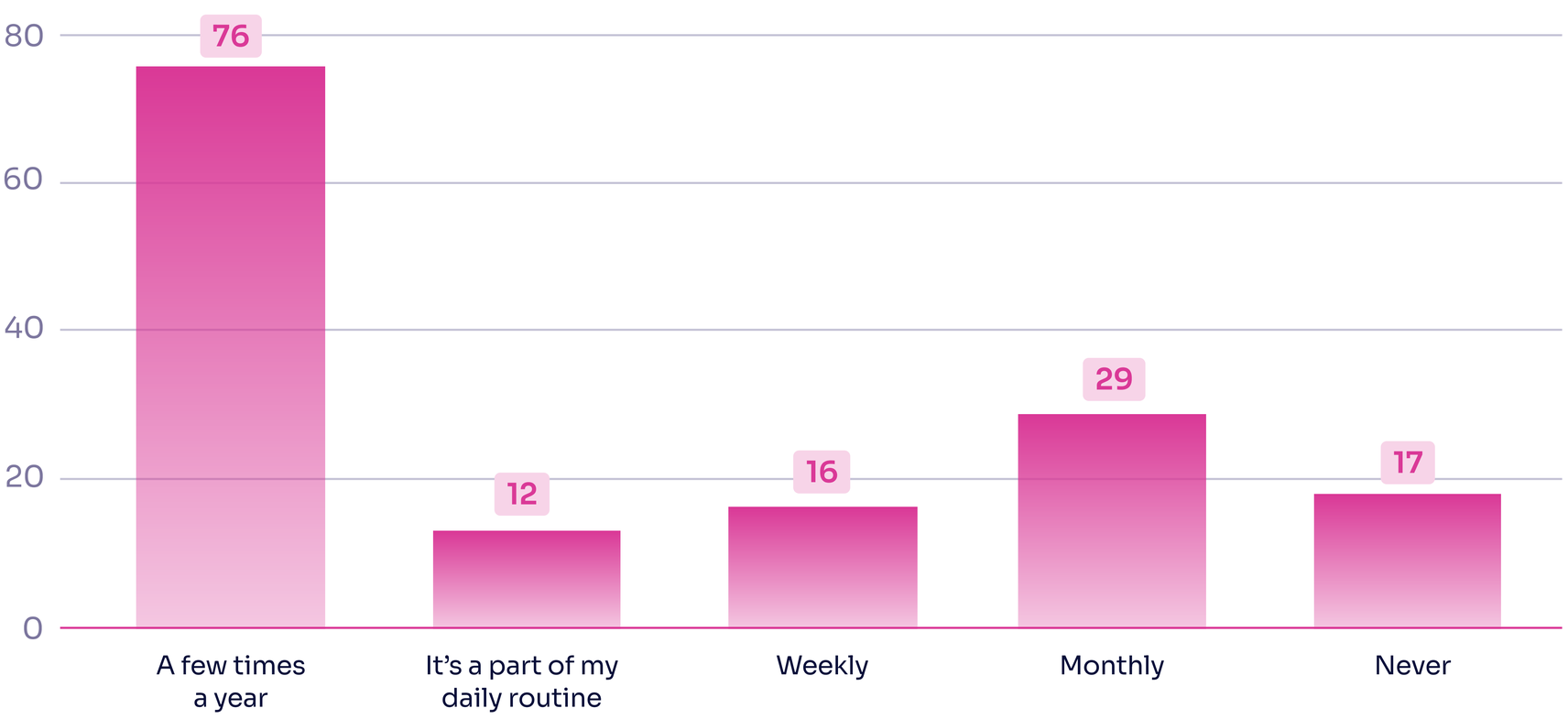
### The relationship between developer confidence and ability

Confidence does not necessarily imply expertise or ability in the domain of code security. In fact, a false sense of confidence among developers' ability to deliver secure code could have negative consequences for application security posture.

> At Jit, we recommend continuous security testing for every code change to highlight issues within the coding environment – providing objective feedback on code security.

This degree of confidence in these results is surprising considering the findings from the next question on how often participants were involved with application-security-related activities.

## 7 How frequently are you involved in application security-related activities (such as security reviews, issue resolution, threat modeling) during the development lifecycle?




62% of participants indicated that they are involved with application security-related activities a few times a year or not at all. Just 22% of participants indicated that they were involved with application security-related activities on a weekly or daily basis.

### Infrequent engagement reflects broader organizational challenges

The limited frequency of security involvement aligns with challenges like time constraints and lack of organizational prioritization (Question 1). Developers frequently reported that security is deprioritized in favor of feature delivery, leaving them with little time to actively engage in security processes – which is highlighted by one of the participants:

> "Leadership wants 'secure/bug-free code' automatically, without prioritizing it or including that time in estimations. New flashy features always come first."

These findings are also consistent with insights into the security culture of development teams (Question 10), with the majority of participants stating that "Security is not a priority and hardly discussed" and "Security is somewhat important and discussed occasionally".

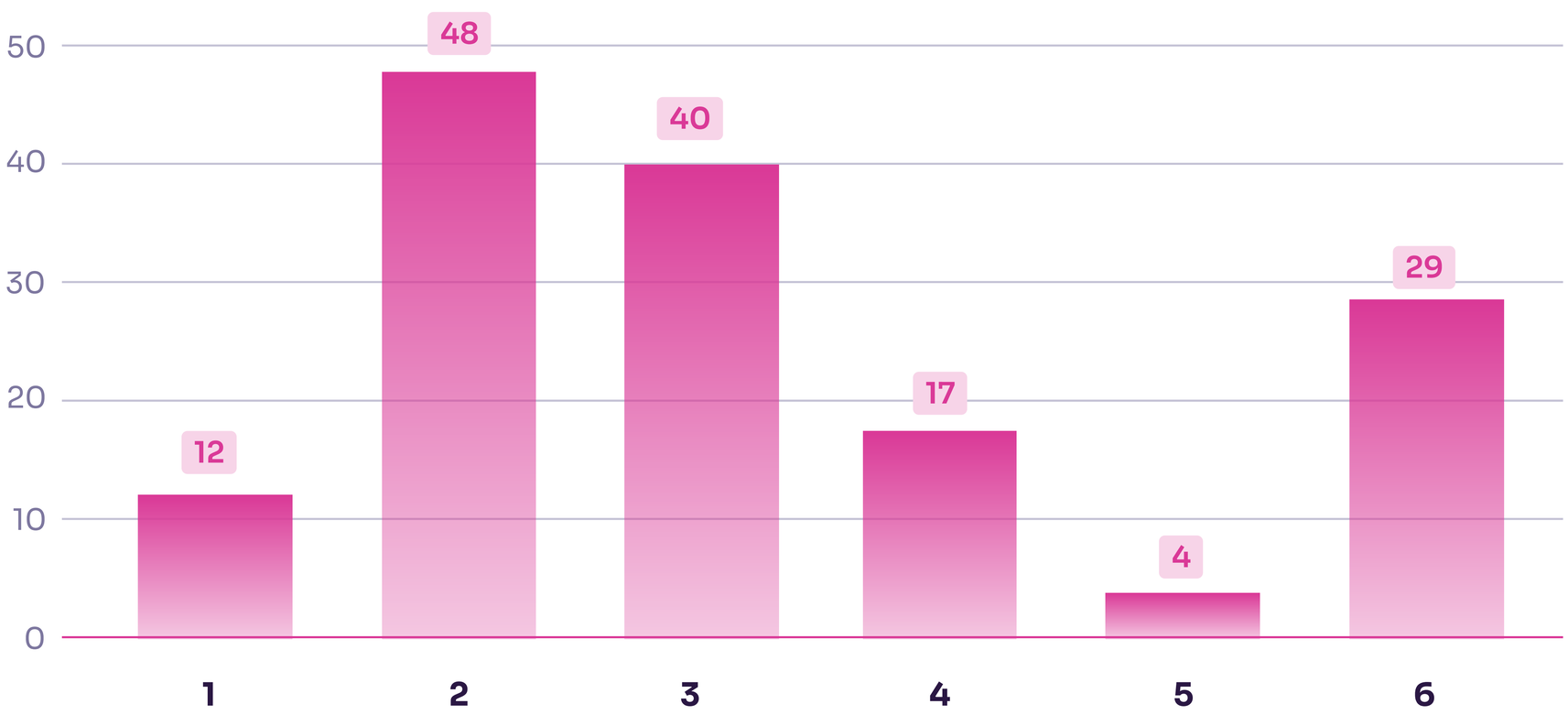
### Correlation: How does involvement with AppSec-related activities impact developer confidence in securing their code?

By analyzing participants' frequency of involvement in AppSec-related activities and calculating the average security confidence score for each group, we can observe how regular engagement influences developers' confidence in securing their code.

| How frequently are you involved in application security-related activities during the development lifecycle? | On a scale of 1-7, how strongly do you agree or disagree with the following statement? "I can consistently and independently deliver secure code." (1=strongly disagree, 7=strongly agree) |
|---|---|
| Never | 4.2 |
| A few times a year | 5.2 |
| Monthly | 5.3 |
| Weekly | 5.6 |
| It's part of my daily routine | 5.5 |

According to this data, we can see that participants with more frequent involvement in AppSec-related activities have a higher degree of confidence in their ability to secure their code.

## 8 How would you describe the collaboration between the development team and the security team at your company?




1 Excellent: We work closely together and have a strong partnership

2 Good: We collaborate effectively, but there is room for improvement

3 Fair: We have some collaboration and it is occasionally effective

4 Poor: Collaboration is minimal and ineffective:

5 Nonexistent: We don't have any direct collaboration with the security team

6 We don't have a security team

The majority of participants indicated moderately positive collaboration with their security team, with 58% of participants claiming "Good" or "Fair" collaboration with their security team. Just 14% of participants said their collaboration was "Poor" or "Nonexistent" and 8% of participants cited their collaboration as "Excellent".

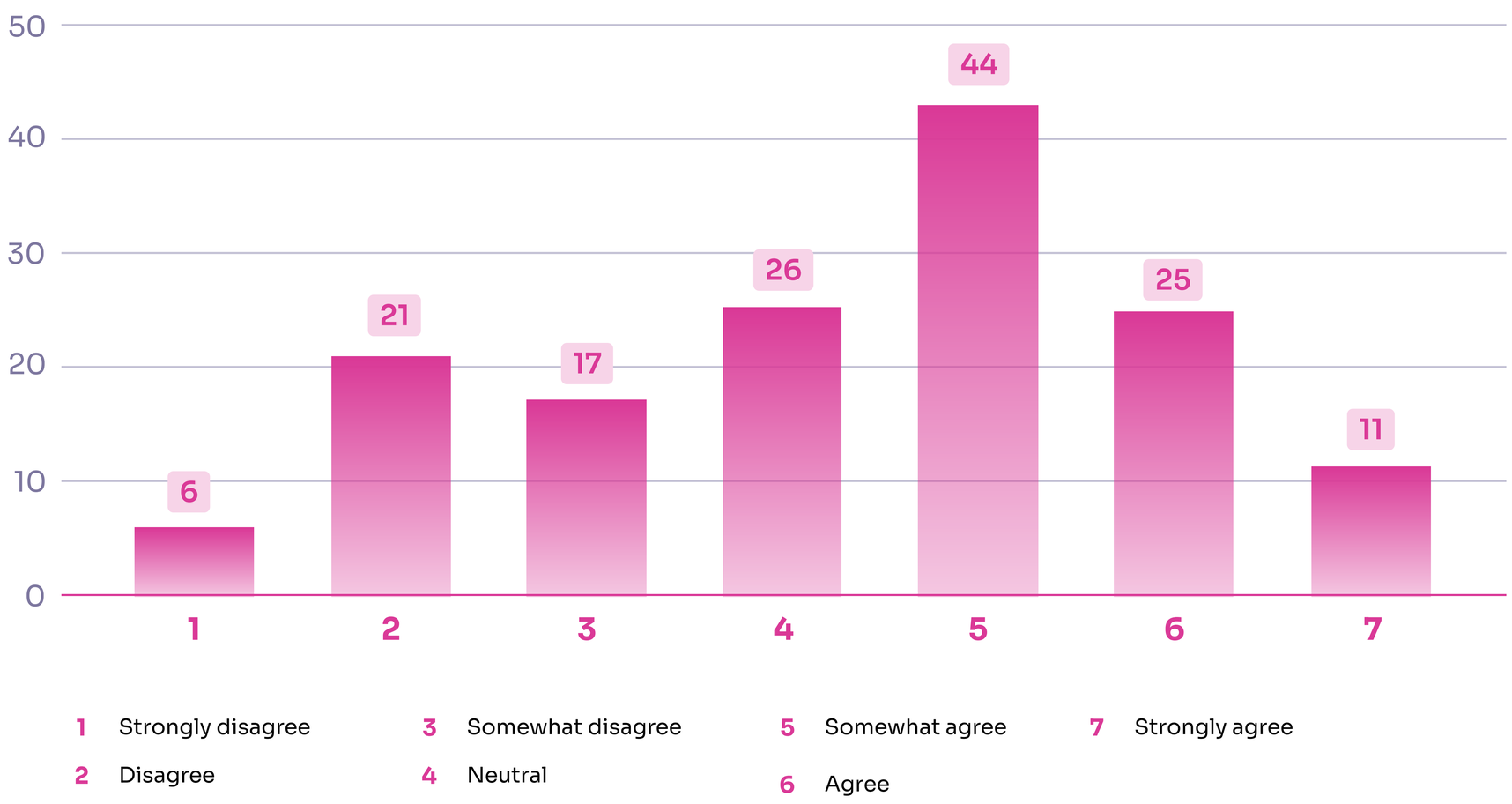
### Effective collaboration is correlated with strong security engineering cultures

Participants who indicated stronger collaboration with the security team were more likely to cite a stronger security culture within their development team (which was asked in Question 10). According to these results, security teams can impact the development cultures with close collaboration.

| How would you describe the collaboration between development team & security team | How would you describe the security culture within your development team? (Average) 1=not a priority, 2=somewhat a priority, 3=important priority, 4=top priority |
|---|---|
| Never | 1.9 |
| A few times a year | 2.2 |
| Monthly | 2.6 |
| Weekly | 3.1 |
| It's part of my daily routine | 2.2 |

While there is a persistent, if not cliché, theme in the security industry that development and security teams don't get along, this data suggests this view isn't widely held by developers.

**9** **How strongly do you agree or disagree with the following statement? "I have full visibility into the security of my services and the most critical security vulnerabilities that need to be resolved.**




While the most common response was that developers "somewhat agree" that they had full visibility into the security of their services, 47% of participants did not agree with that statement to varying extents.

The average of all responses was 4.33 – suggesting an overall moderate confidence in the visibility of service security. As suggested by one of the participants, those that heavily rely on third-party code may have difficulty understanding their security weaknesses:

> "When using third-party libraries, developers often have limited insight into their internal workings. This makes it harder to assess security risks or spot potential vulnerabilities. Without understanding the internal code, it's challenging to confidently ensure the security of the overall application."

### Which implemented code security strategies yield improved visibility into security?

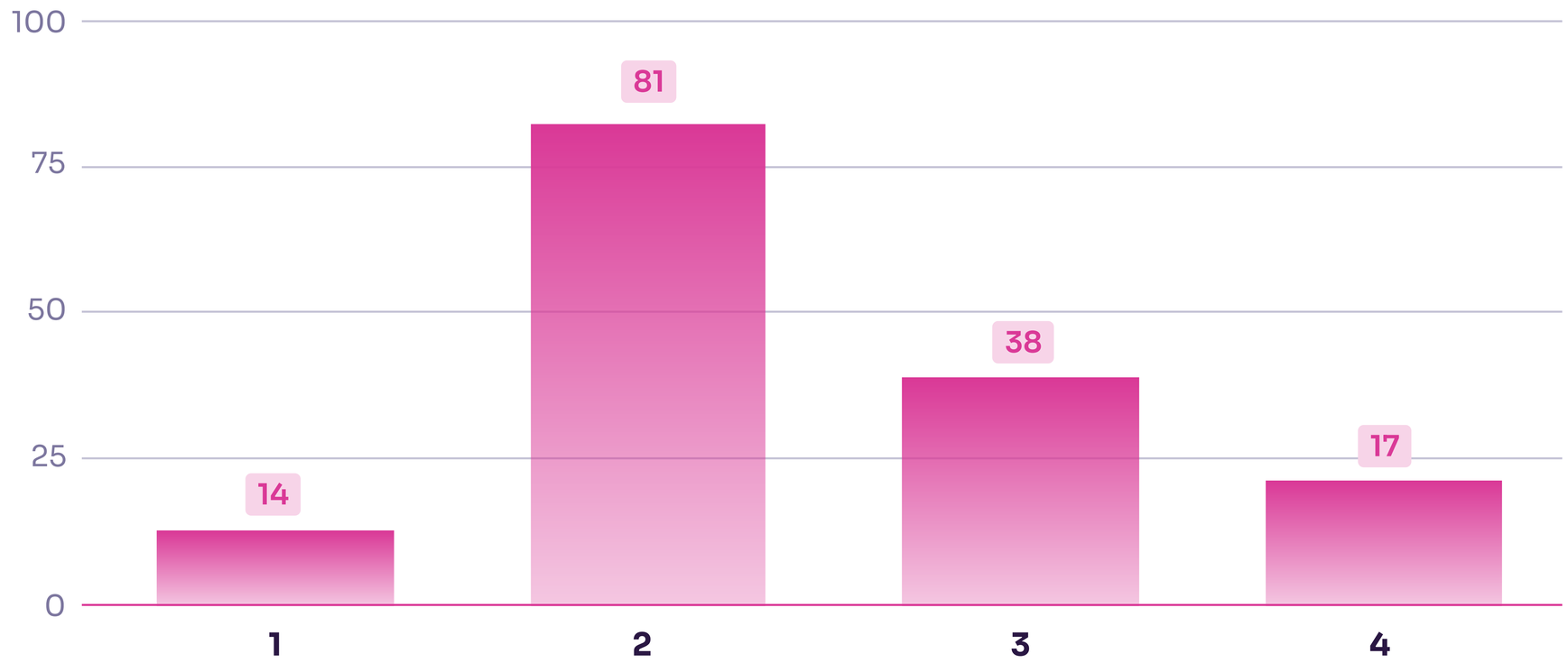
To assess how code security strategies influence developers' visibility into security, we can calculate the average visibility score for each group of participants who reported the implementation of specific strategies by their team.

| How does your company support you in building secure applications? | Average score: "I have full visibility into the security of my services and the most critical security vulnerabilities that need to be resolved." (1 = low visibility, 7 = high visibility) |
|---|---|
| Regular code security training | 4.5 |
| Implemented security scanners, like SAST, SCA, secrets detection, DAST, etc... | 4.7 |
| Building security champions within development teams | 5.1 |
| Dedicated security team detects vulnerabilities and triages to developers | 4.7 |
| No support | 3.3 |
| Code reviews that focus (at least partially) on security | 4.5 |
| By maintaining internal docs on code security best practices | 4.4 |

According to the data, AppSec teams can consider implementing the following code security strategies to improve developer visibility into security:

- Building security champions within development teams
- Implemented security scanners (e.g. SAST, SCA, secrets detection, etc...)
- Dedicated security team detects vulnerabilities and triages to developers

## 10 How would you describe the security culture within your development team?




1 Security is not a priority and hardly discussed

2 Security is somewhat important and discussed occasionally

3 Security is important, integrated into our routines, and discussed often

4 Security is a top priority, integrated and enforced throughout our routines, and discussed constantly

61% of participants indicated that security is “somewhat important” or not a priority at all in their development culture, and that it wasn’t integrated into their routines. This is consistent with answers from Question 1, which nearly ranked “organizational priority” as the top challenge to application security – as described by one of the participants:

> “Making security an organizational priority and creating a culture where you teach developers security are in my opinion the two most important things that would lead to better overall security.”

## Correlation between security culture and developer confidence in delivering secure code

Developers who indicated a stronger security culture within their team were more confident in their ability to deliver secure code

| Describe the security culture within your development team | On a scale of 1–7, how strongly do you agree or disagree with the following statement? "I can consistently and independently deliver secure code." (Average score)<br>1=strongly disagree, 7=strongly agree |
|---|---|
| Security is not a priority and hardly discussed | 4.5 |
| Security is somewhat important and discussed occasionally | 5.1 |
| Security is important, integrated into our routines, and discussed often | 5.4 |
| Security is a top priority, integrated and enforced throughout our routines, and discussed constantly | 5.7 |

As we saw in Question 8, there was also a correlation between strong collaboration with security teams and the security culture among development teams. Taken together with these findings, there seems to be a strong relationship between security culture, collaboration between development and security teams, and developers' confidence in their code security.

# Conclusion

This survey highlights the critical role of developers in securing applications and provides actionable insights for application security (AppSec) teams looking to design programs that better engage developers and improve security outcomes.

By exploring the developer perspective, the survey uncovers key challenges, preferred strategies, and the organizational dynamics that impact developers' ability to write secure code.

**A recurring theme throughout the findings is the tension between complexity, time constraints, and organizational support.**

Developers identified the complexity of modern application architectures, lack of knowledge and training, and organizational prioritization as their most significant challenges. These issues are compounded by inconsistent collaboration between security and development teams and limited opportunities for developers to engage with security tasks on a regular basis. Increasing collaboration with security teams was found to improve the security culture among development teams.

**Developers favor automated tools like SAST, SCA, and secrets detection as the most impactful strategies for securing code.**

**However, poor integrations with the developer environment and noisy results often limit their effectiveness.** At the same time, developers' reliance on external resources such as online documentation highlights gaps in internal training and support, particularly for those struggling with security knowledge.

**The survey also highlights the strong connection between security culture, collaboration with security teams, and developer confidence.**

Teams with better collaboration between development and security and a stronger security culture—where security is prioritized, integrated, and routinely discussed—report higher confidence in their ability to deliver secure code.

Ultimately, this survey serves as a guide for AppSec teams to better understand the developer experience and identify opportunities to improve alignment with security initiatives. By addressing the challenges developers face, fostering stronger collaboration, and equipping teams with effective tools and training, organizations can empower developers to play a proactive role in securing applications and reducing overall security risk.