2019

State of DevOps Report

Presented by

puppet

circleci

splunk>

Sponsored by

ANITIAN

f5

servicenow

Contents

1

Executive summary

We're on our eighth year of publishing the State of DevOps Report, the longest-standing and most widely referenced DevOps research on the planet. Last year's survey results revealed an evolutionary model of DevOps that helps us better understand how organizations start with DevOps and scale successfully. Our objective was to provide pragmatic, prescriptive guidance for organizations struggling to achieve success with DevOps.

As we shared our results with organizations around the world, people kept bringing up one of our key findings: that the most evolved organizations were able to integrate security into the software development cycle, with excellent results. These firms had achieved not only better integration of development, operations and security, but also automation of security measures.

For most companies we've spent time with though, integrating security into the software delivery lifecycle is an unrealized ideal, and an obstacle to furthering their DevOps evolution. Why is that the case? When security practices are so well understood, why is it so hard to integrate security into DevOps?

Call us cynical, but we believe it's because good security practices don't pay the bills. Good security is not a competitive differentiator. Getting new features out faster, on the other hand, gives you the clear competitive advantage of being early to market. So feature delivery naturally becomes the top priority.

To change this dynamic, organizations need to prioritize security from the top and incentivize all teams to share responsibility for it — not just designated security specialists. Security teams need to be good partners to the rest of the business, enabling other teams to establish sound practices.

Unfortunately, good intentions alone don't change habits. Firms that are undergoing DevOps transformations want and need guidance on how to integrate security. So just as we set out last year to discover whether there are patterns and practices that lead to successful DevOps transformation, we decided this year to explore any possible patterns and practices that help organizations integrate security into the software development lifecycle. Just as important, we wanted to discover whether security integration leads to better business outcomes.

Key findings

1. Doing DevOps well enables you to do security well.

Firms that have security integrated throughout the software delivery lifecycle are much more likely to be using DevOps practices across the enterprise. For these firms, DevOps has moved beyond early local optimizations such as dev teams adopting version control and continuous integration, and infrastructure teams testing infrastructure alongside application code. DevOps practices now affect the business itself.

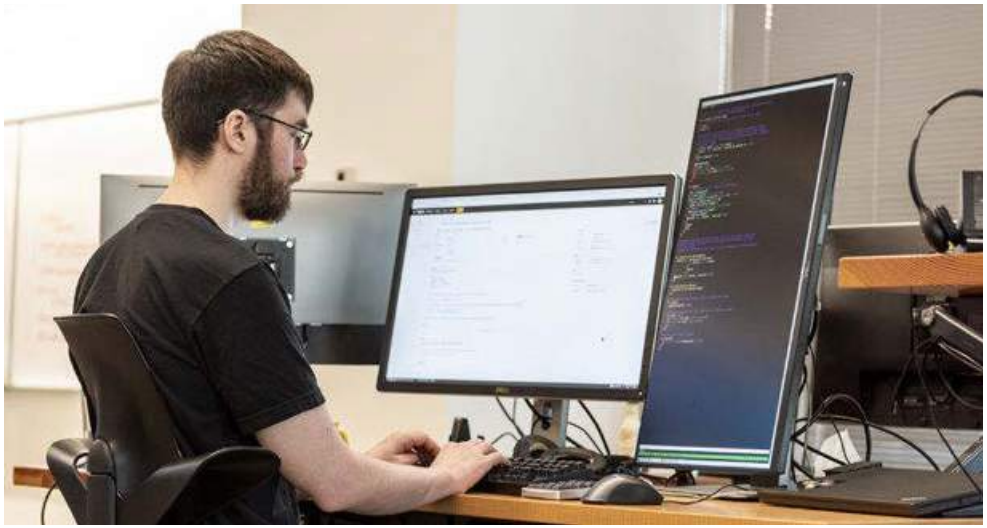
We found that 22 percent of the firms at the highest level of security integration have reached an advanced stage of DevOps evolution. The DevOps principles that drive good outcomes for software development — culture, automation, measurement and sharing — are the same principles that drive good security outcomes. Reliability, predictability, measurability and observability in your deployments create not just intrinsically more secure environments, but also, when combined with a strong automation practice, enable speed of response to security issues as they arise.

A strong DevOps culture also supports stronger security. A culture of sharing, where teams collaborate using common tools and work towards common goals; where delivery teams have strong autonomy, yet it's relatively easy to cross organizational boundaries to get work done — this is a culture where security can be truly a shared responsibility, where issues can be identified early and resolved in the best possible way.

2. Integrating security deeply into the software delivery lifecycle makes teams more than twice as confident of their security posture.

Eighty-two percent of survey respondents at firms with the highest level of security integration said their security policies and practices significantly improve their firm's security posture. Compare this with respondents at firms with no security integration — just 38 percent had that level of confidence.

Integrating security at every stage of the software delivery lifecycle is more than just shifting security checks to the left. Security integration requires a completely different approach, one that emphasizes cross-team collaboration and empowers delivery teams to autonomously prevent, discover and remediate security issues. Breaking down knowledge silos between teams, and collaborating to improve security both raise overall awareness of security concerns making it more likely that everyone — even those outside the security team — will adopt known patterns for security protection.



Firms that have integrated security at all stages of delivery collaborate early, often and most importantly, deeply.

The top five practices that improve security posture are:

1. Security and development teams collaborate on threat models.
2. Security tools are integrated in the development integration pipeline so engineers can be confident they're not inadvertently introducing known security problems into their codebases.
3. Security requirements — both functional and non-functional are prioritized as part of the product backlog.
4. Security experts evaluate automated tests, and are called upon to review changes in high-risk areas of the code (such as authentication systems, cryptography, etc.).
5. Infrastructure-related security policies are reviewed before deployment.

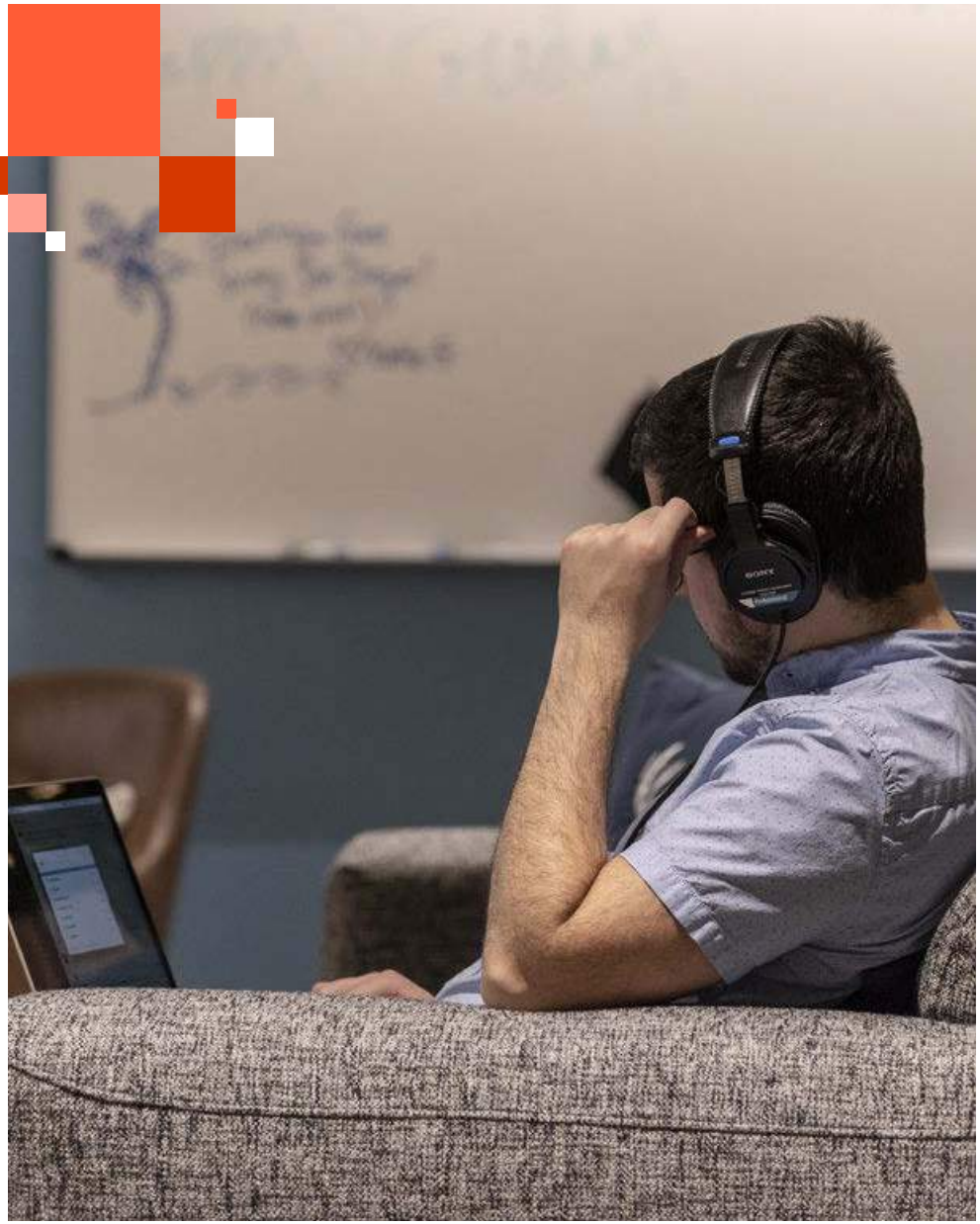
All these practices are things that most firms today know they should do, but aren't doing because it's hard. The reward for doing them, though, is greater confidence in your security posture. While there will always be threats you didn't anticipate, you'll know you can recognize and react to those threats effectively.

3. Integrating security throughout the software delivery lifecycle leads to positive outcomes.

We hypothesized that firms at the highest level of security integration are able to deploy more frequently, remediate vulnerabilities faster, and have a more positive attitude towards security and audits. We found:

- Firms at the highest level of security integration are able to deploy to production on demand at a significantly higher rate than firms at all other levels of integration — 61 percent are able to do so. Compare this with organizations that have not integrated security at all: Fewer than half (49 percent) can deploy on demand.
- To our surprise, time to remediate vulnerabilities did not dramatically decrease at higher levels of security integration, though the difference between the highest level of integration and the lower levels is significant.
- Firms with deeper security integration were able to more effectively prioritize security improvements over feature delivery, and also were better able to halt a push to production in order to address a security issue.

The more security is integrated into the software delivery lifecycle, the more delivery teams see security as a shared responsibility. And as integration increases, so does the perception that audits minimize risk to the business. In organizations with a high level of security integration, identified security issues are prioritized by the business.



4. Security integration is messy, especially in the middle stages of evolution.

The patterns we see as organizations adopt DevOps practices and scale DevOps success are the same patterns we see as they integrate security into software delivery. One common element: The middle stages are difficult and sometimes frustrating.

As with any new project, when you're first starting out, you don't know what you don't know. You get started, and soon you're feeling good about picking off the low-hanging fruit and getting some small wins on the board.

Then you hit your first big roadblock. Now you can see the underlying complexity that's been masked over by years of duct tape and glue. You tackle the roadblock, but as you resolve it, new obstacles appear. You resolve one roadblock after another, and it gets frustrating, but after a while, you see that your team can overcome issues as they arise. Finally, after what feels like a long time (but usually shorter than you think), you can point to measurable success.

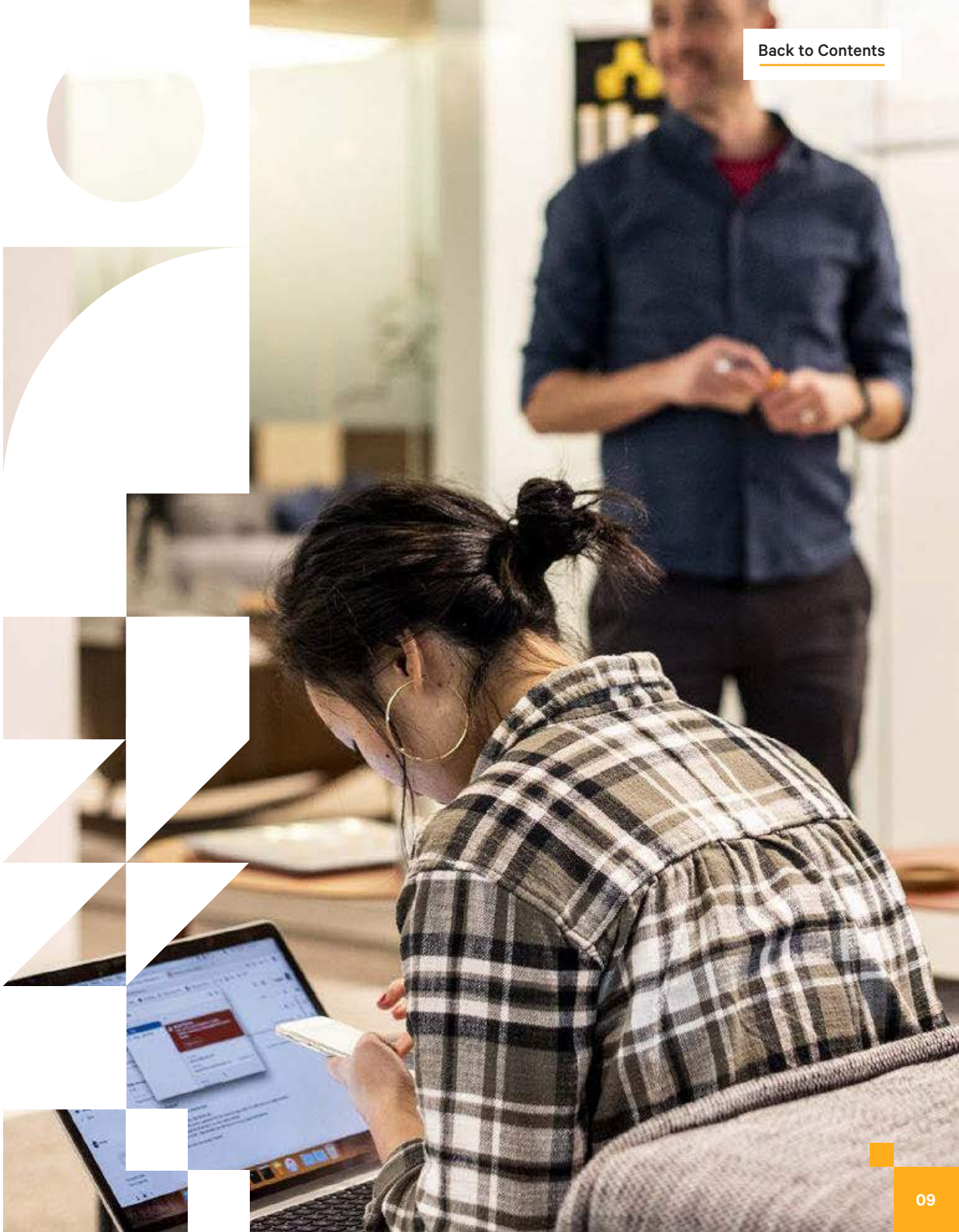
In these slog-through-it middle stages, security and delivery teams experience higher friction while collaborating, software delivery slows down, and audit issues both increase and require immediate attention. This is the messy reality of organizational change, and it's a natural part of evolution. Stick with it, focus on how much you've reduced manual toil, cultivate camaraderie with your colleagues, and things will improve. You and your team will reap the rewards of all that hard work as you move on to more advanced practices, and you'll start seeing quicker results.

2

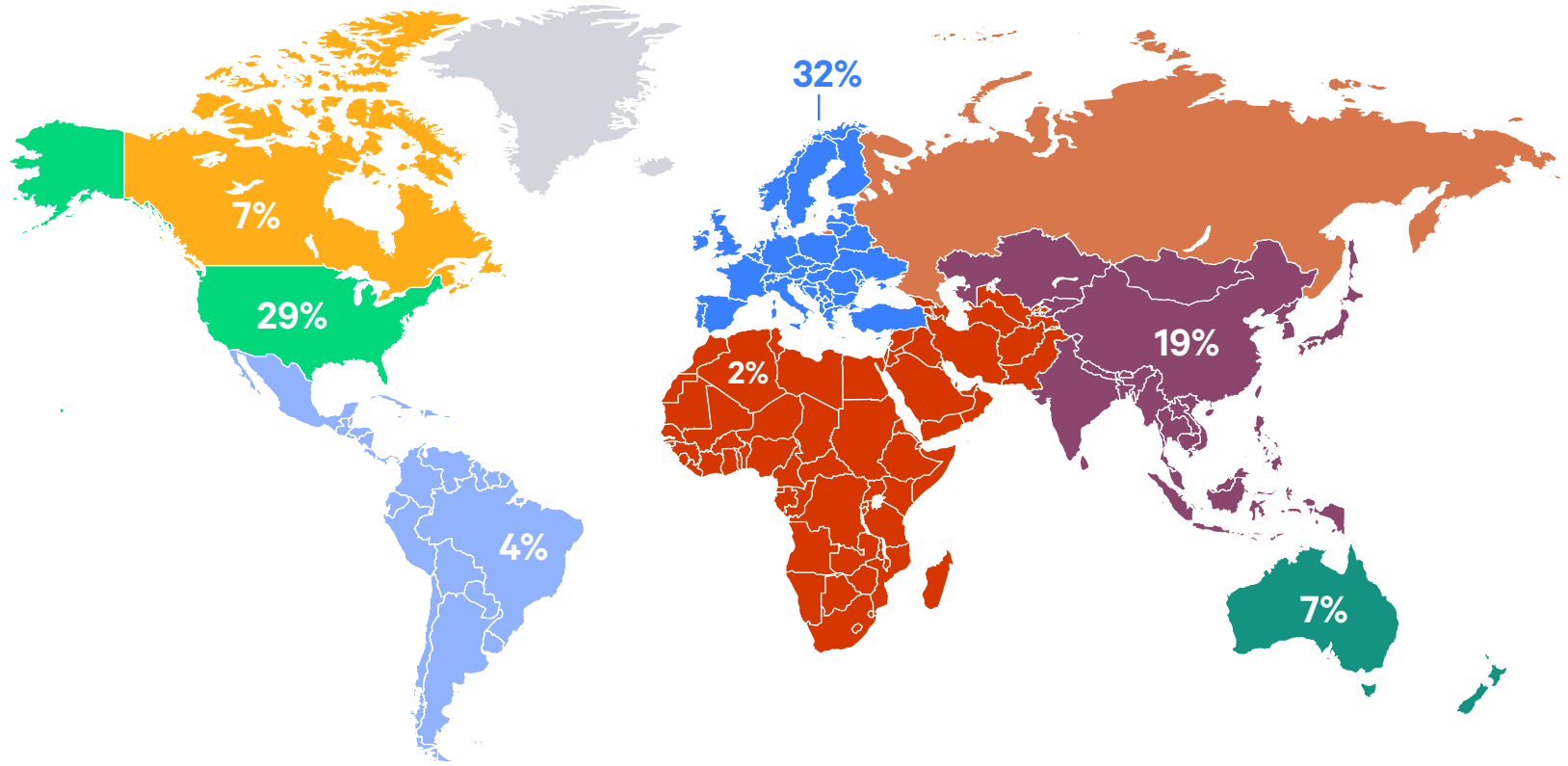
Who took the survey

For the 2019 State of DevOps Report we continued our focus on getting wider global representation. This year we had the highest percentage of survey responses from Europe, Asia, Australia and New Zealand that we've ever had since the first State of DevOps survey in 2012.

Thanks to everyone — all 2,949 of you — who took the survey this year.



Responses by global region



USA: 29%

Canada: 7%

Asia: 19%

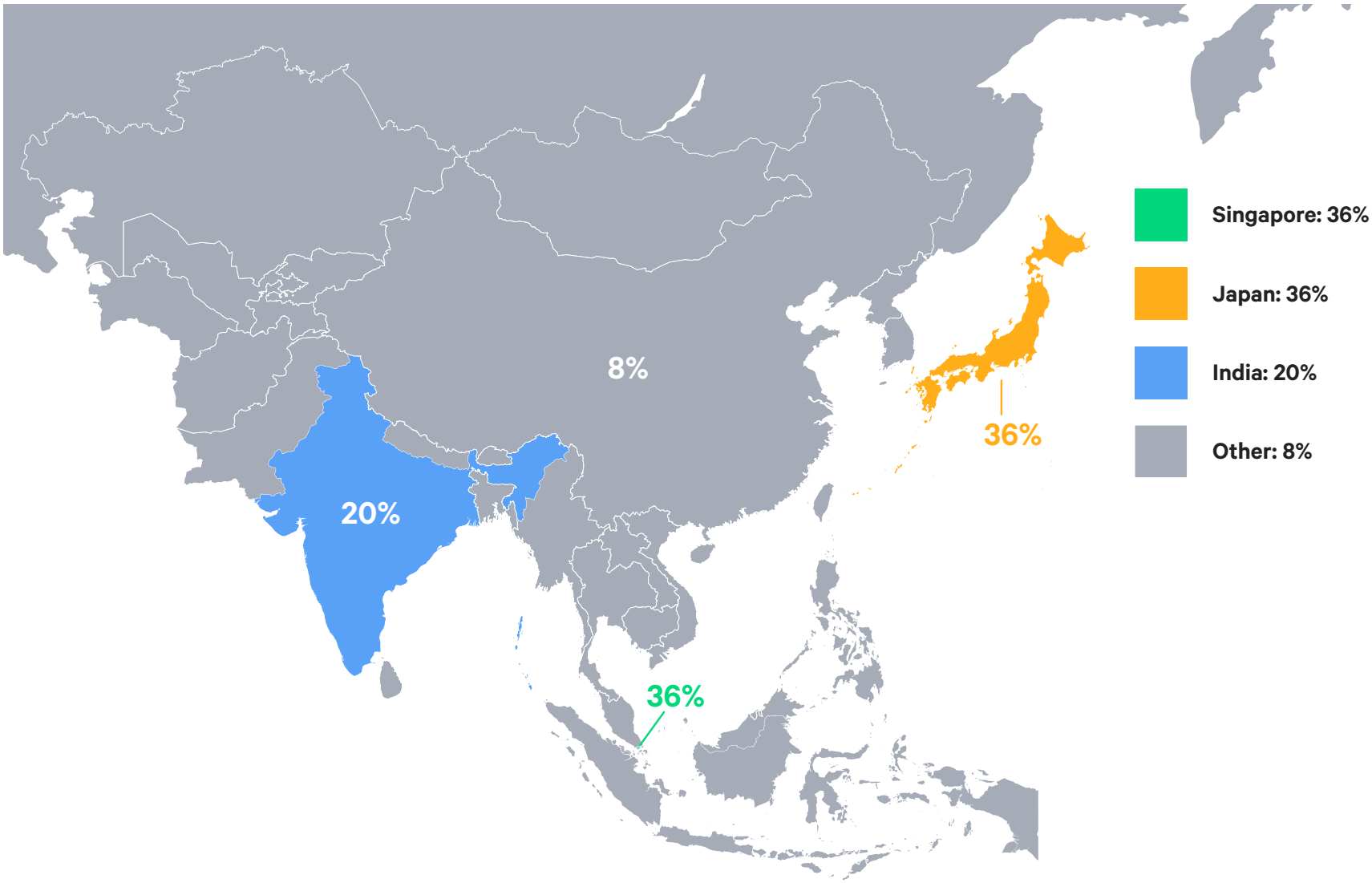
Europe: 32%

Australia & New Zealand: 7%

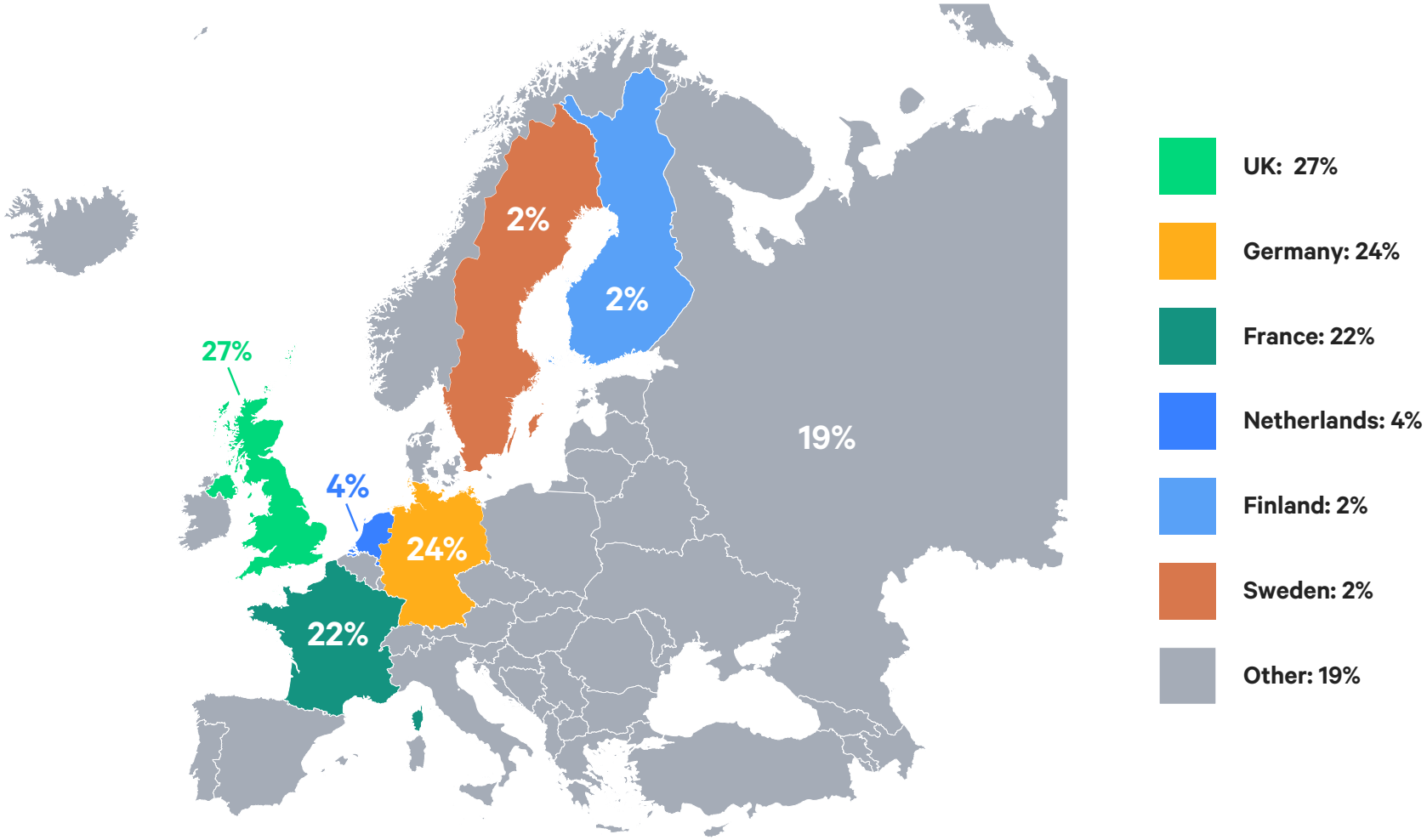
Mexico, Central & South America: 4%

Africa & Middle East: 2%

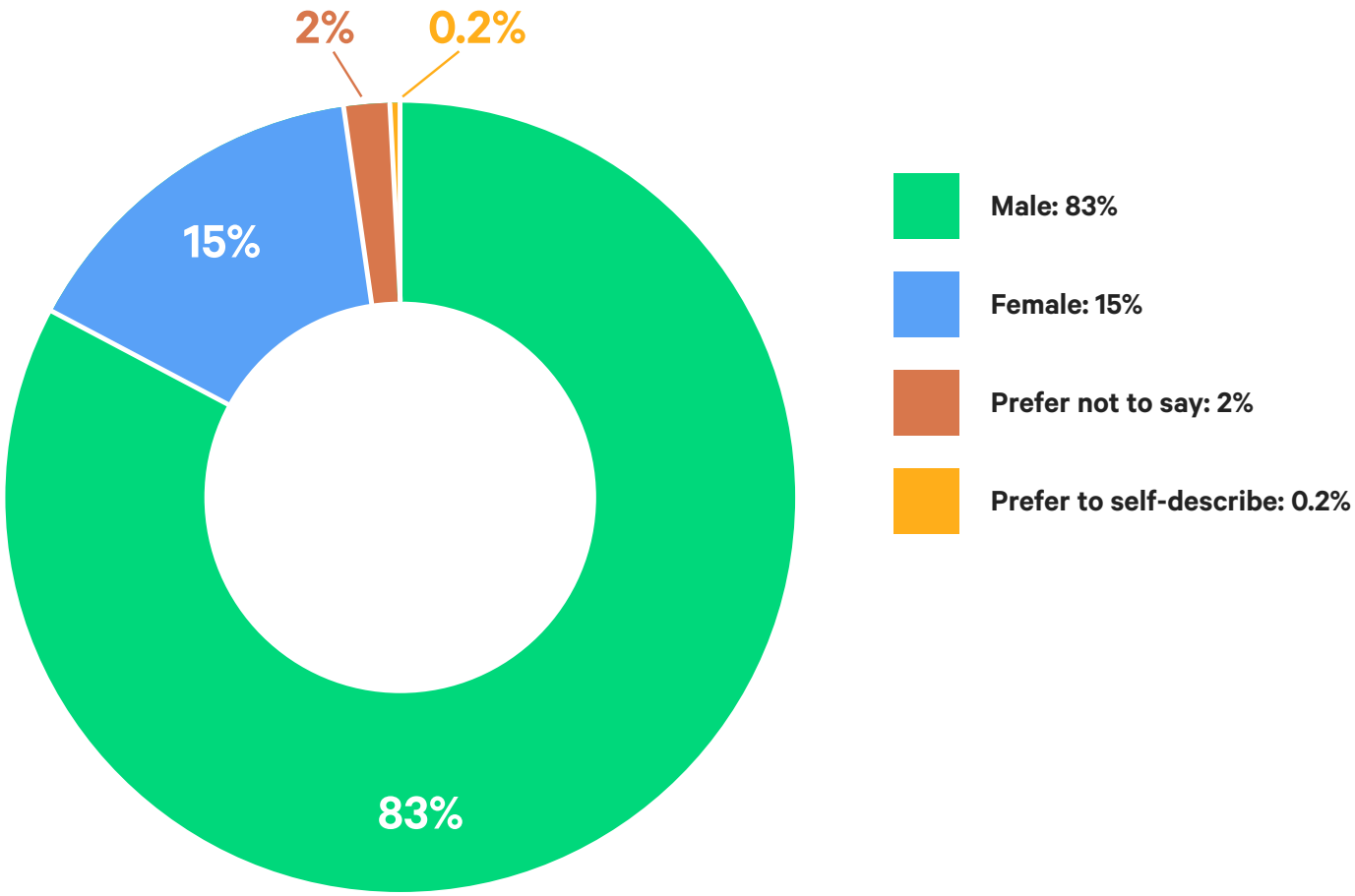
Responses in Asia by country



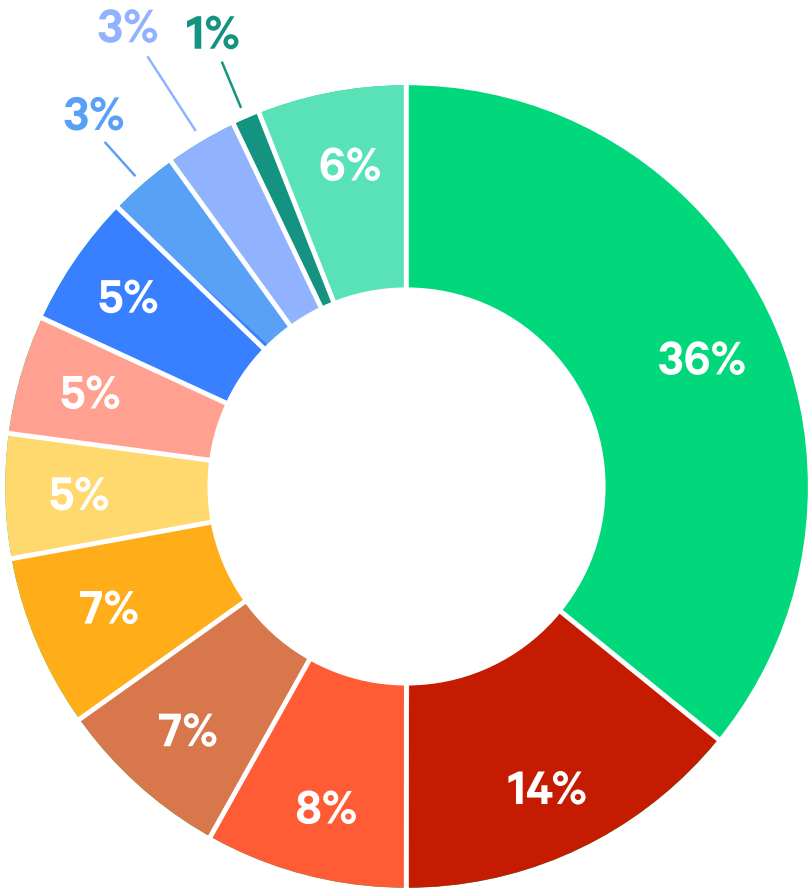
Responses in Europe by country



Gender identity

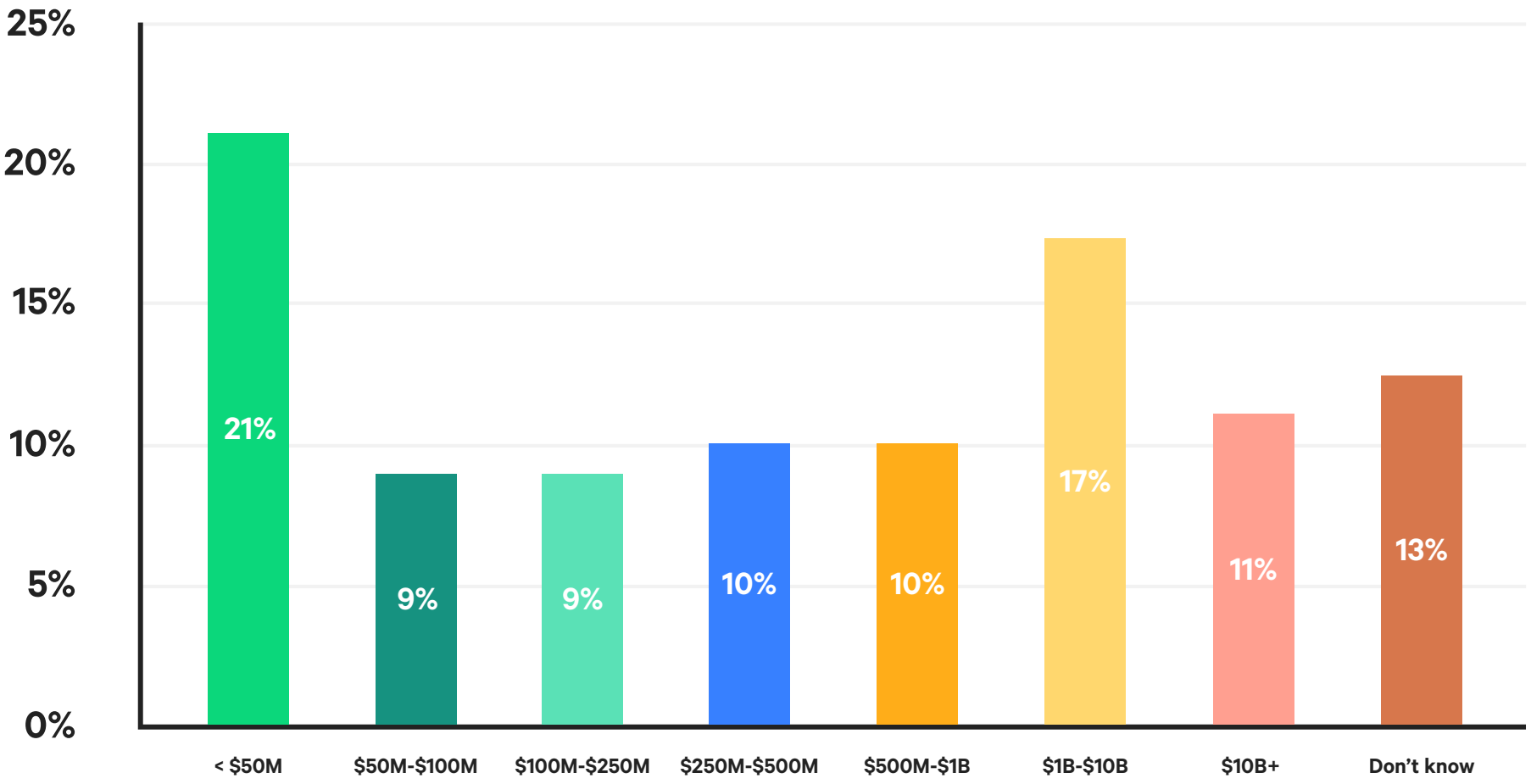


Principal industry

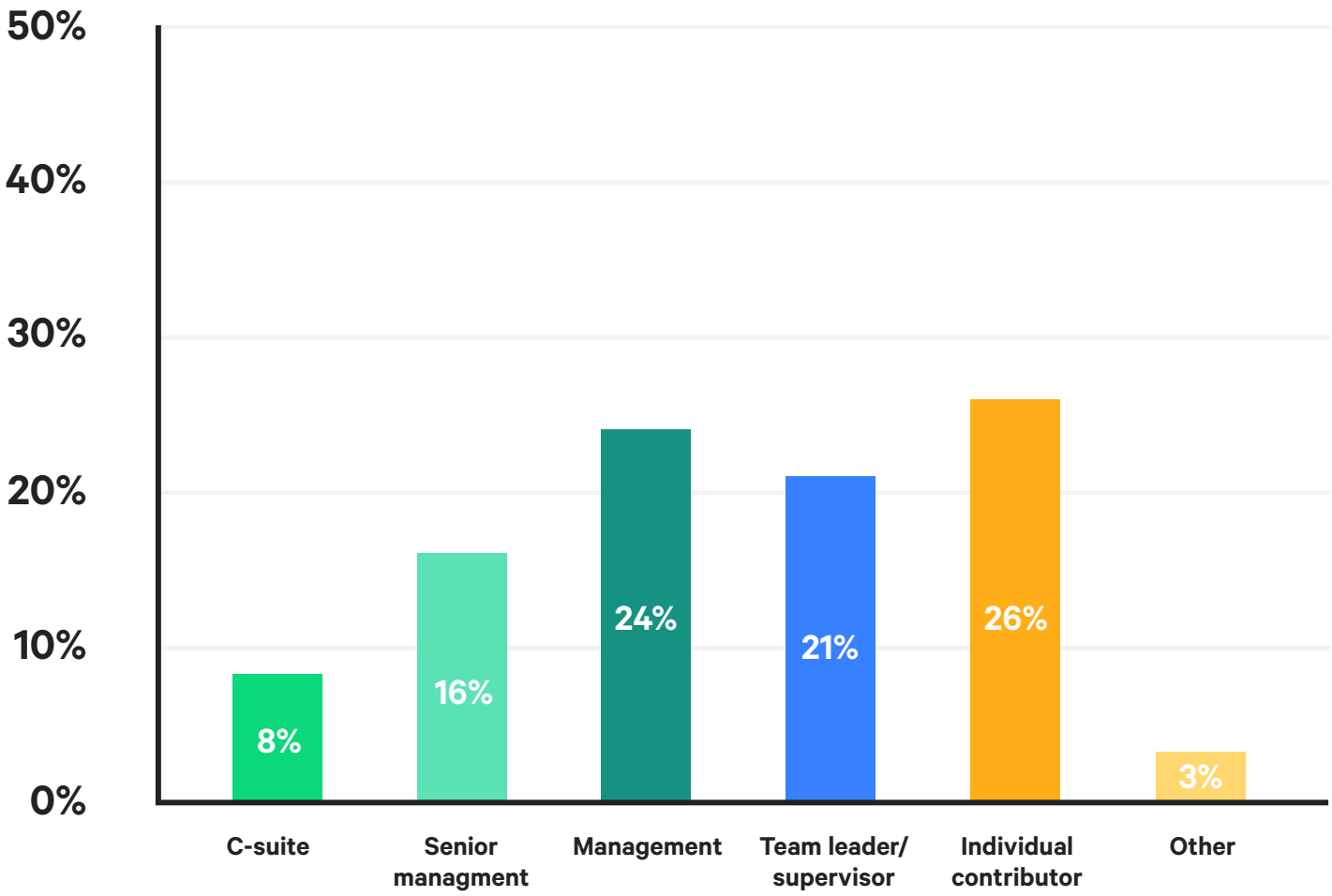


- Technology: 36%
- Financial services & insurance: 14%
- Industrials & manufacturing: 8%
- Retail, consumer, & eCommerce: 7%
- Media & telecom: 7%
- Government: 5%
- Education: 5%
- Life sciences, healthcare & pharmaceuticals: 5%
- Media / entertainment: 3%
- Energy & resources: 3%
- Non-profit: 1%
- Other: 6%

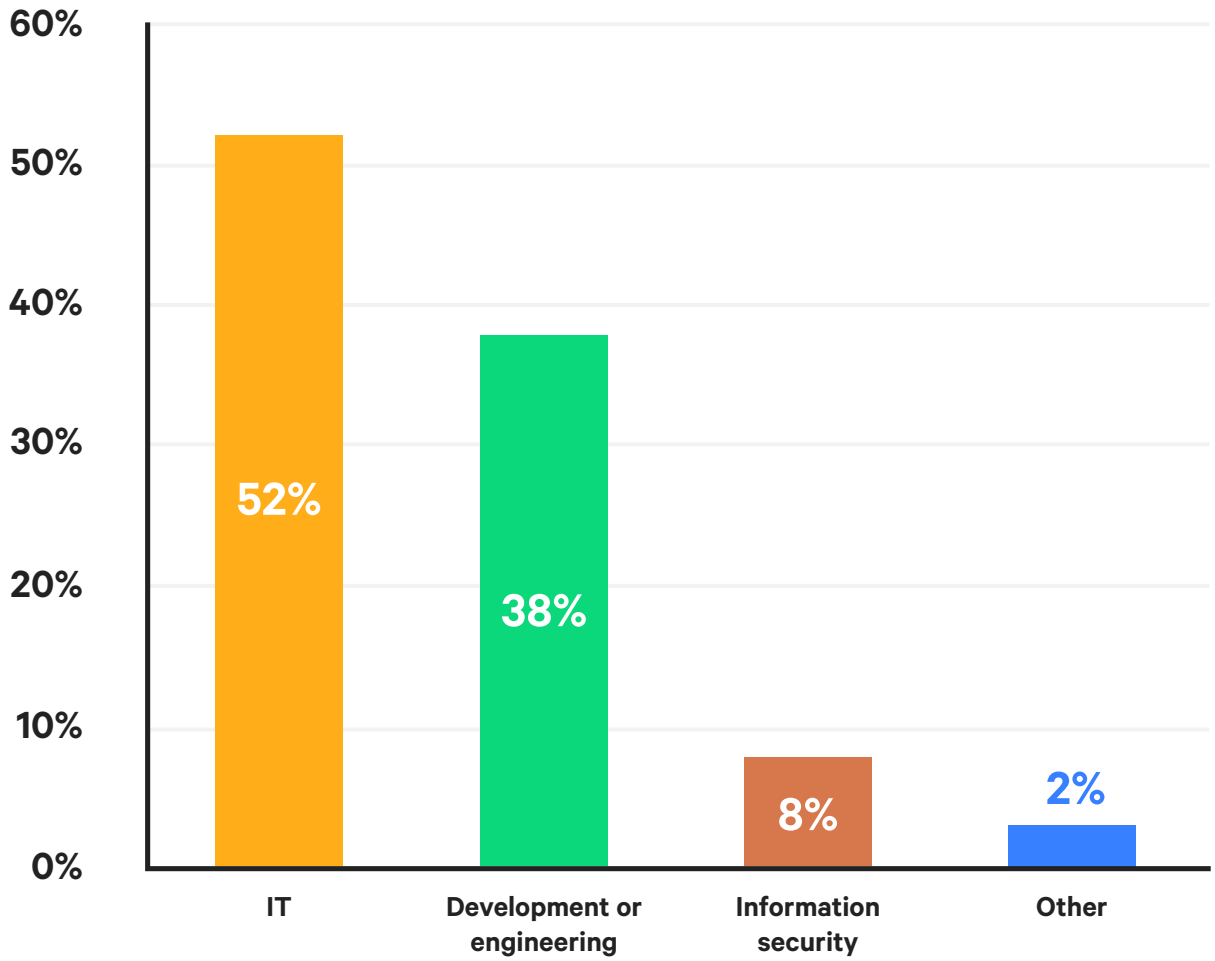
Organization annual revenue



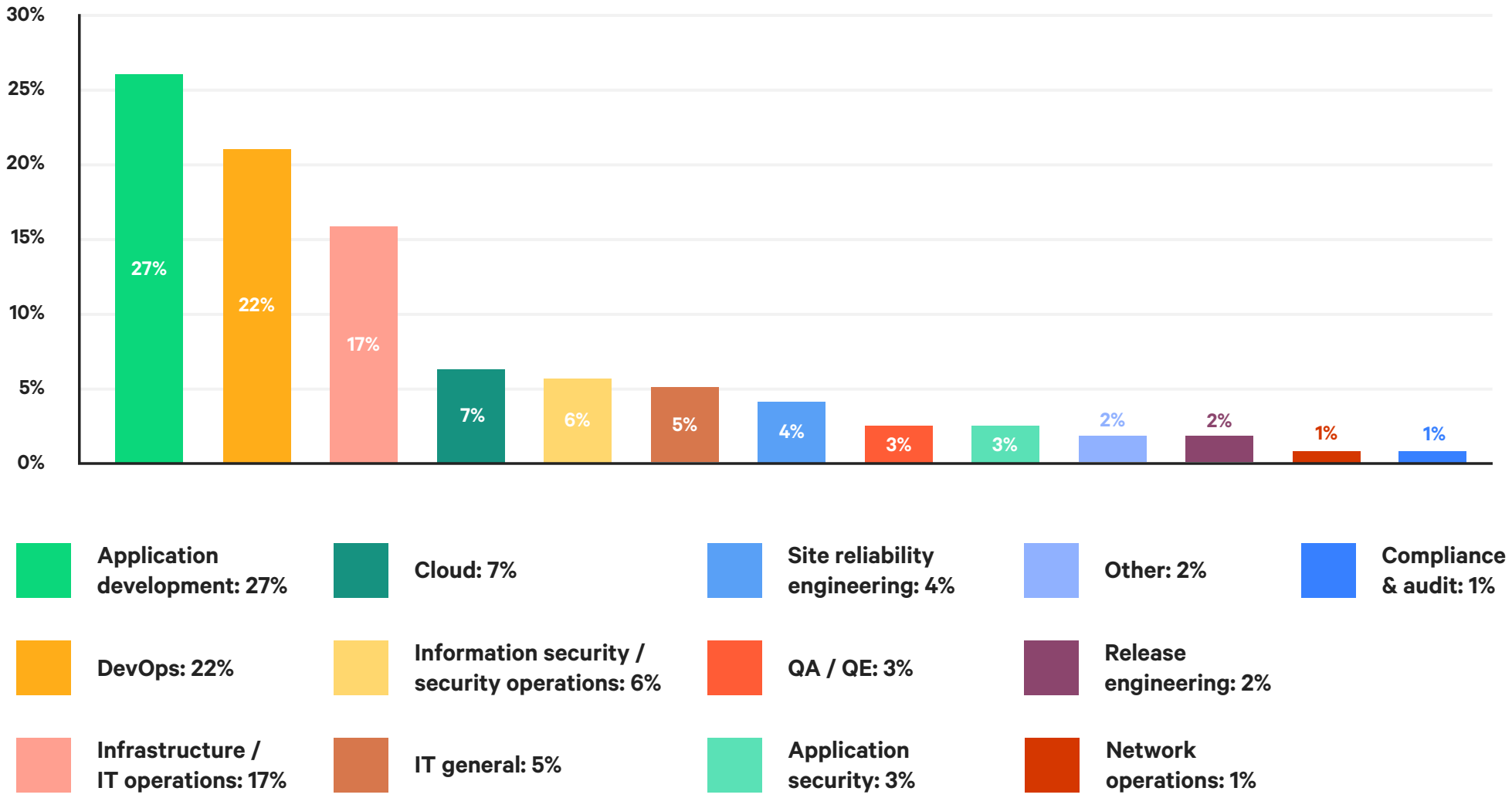
Role within organization



Department


60%
50%
40%
30%
20%
10%
0%
52%
38%
8%
2%
IT
Development or engineering
Information security
Other

Team

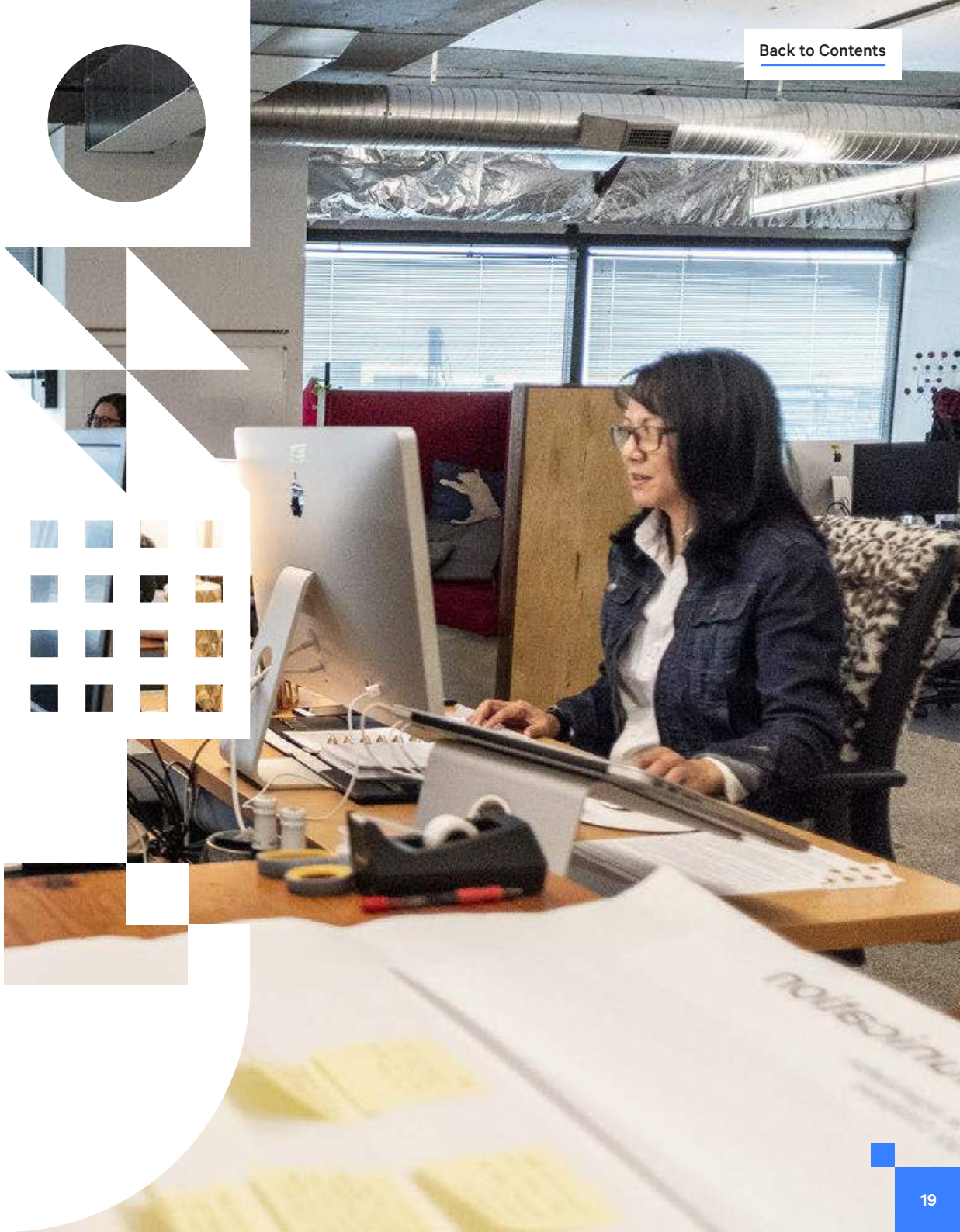

30%
25%
20%
15%
10%
5%
0%
27%
22%
17%
7%
6%
5%
4%
3%
3%
2%
2%
1%
1%
Application development: 27%
Cloud: 7%
Site reliability engineering: 4%
Other: 2%
Compliance & audit: 1%
DevOps: 22%
Information security / security operations: 6%
QA / QE: 3%
Release engineering: 2%
Infrastructure / IT operations: 17%
IT general: 5%
Application security: 3%
Network operations: 1%

3

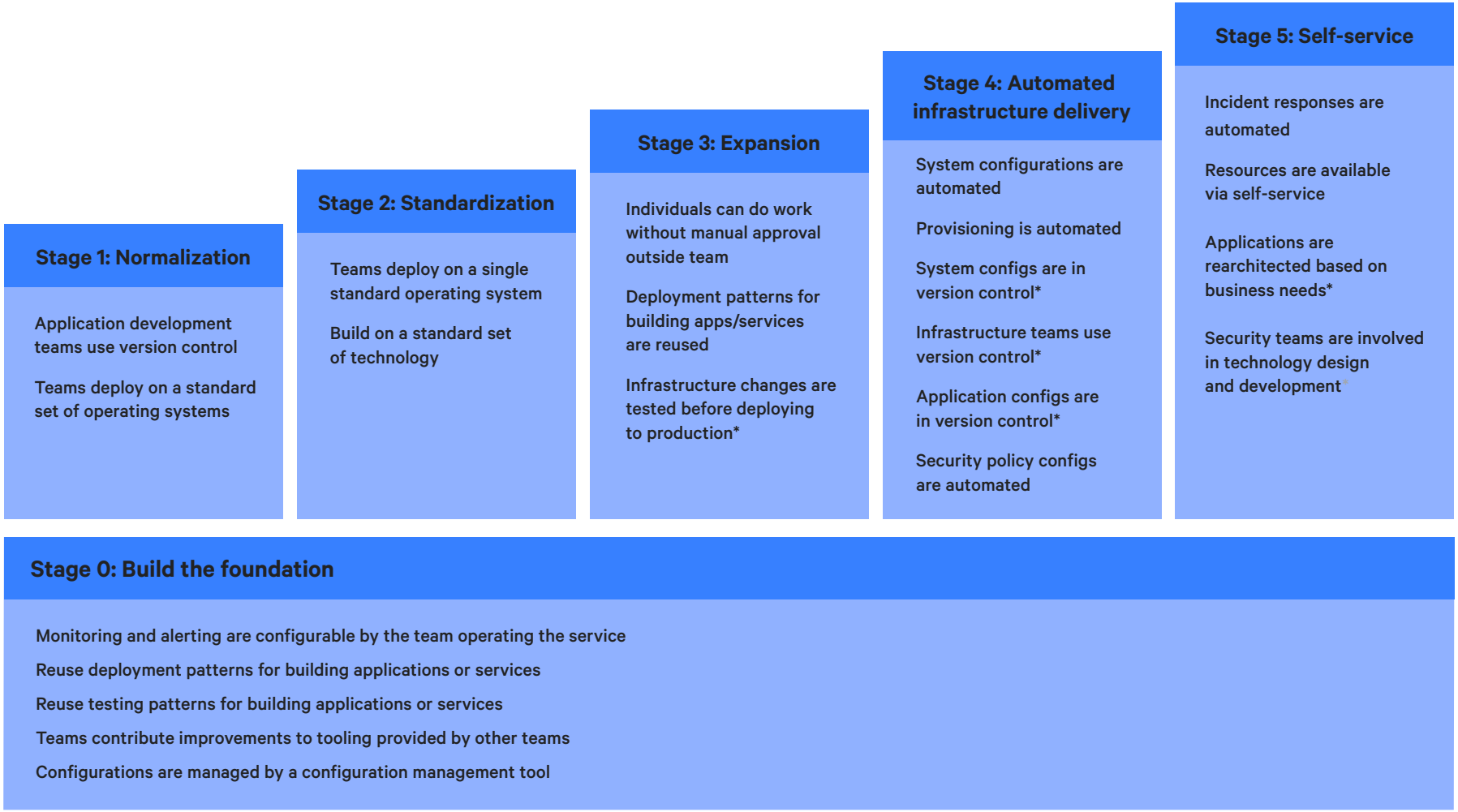
DevOps evolution and the importance of security integration

In 2018 we sought to provide a pragmatic and prescriptive guide to getting started with DevOps and succeeding at scale. While we firmly believe there's no one-size-fits-all when it comes to DevOps, there are clearly established principles, patterns and practices you can apply to achieve success faster.

In the 2018 report, we identified five distinct stages of DevOps evolution, each characterized by critical practices that help you achieve success and progress to the next phase of your DevOps journey. We also identified five foundational practices that organizations must adopt to be successful in their DevOps evolution.



5 Stages of DevOps evolution



* These practices are highly correlated with the stage

We found the organizations that had evolved the furthest on their DevOps journeys were successfully applying automation to security considerations. At Stage 4 of DevOps evolution, for example, teams were automating security policy configurations. This helped teams progress to Stage 5, where we found the key practice of automated incident response. We also learned that organizations at Stage 5 involved their security teams in technology design and deployment.

The highly evolved teams we encountered in last year's report were not simply shifting security left. They had cultivated a powerful blend of high-trust environments, autonomous teams, and a high degree of automation and cross-functional collaboration between application teams, operations and security teams. The result? Security becomes a shared responsibility across delivery teams that are empowered to make security improvements. Security teams are able to act in an advisory role, leading to time-saving and security-enhancing capabilities such as automated incident response. Further, these teams were able to implement transparent security policies as code.

DevOps and security integration: discovering patterns and practices

As we shared last year's findings with organizations undergoing DevOps transformation, people kept asking us, "But how do we integrate security into the software delivery lifecycle?" So we designed this year's survey to uncover whether specific patterns and practices enable tighter security integration.

Our eight years of DevOps research have shown that when operational requirements are built into the software delivery lifecycle, organizations realize better outcomes — faster and more frequent deployments, fewer errors, faster time to remediation and less manual work. We wanted to know if integrating security into the software cycle has a similar impact on outcomes, particularly:

- Key software delivery performance metrics
- Responsiveness to security issues
- How security is perceived throughout the organization
- Sentiments around audits

Spoiler alert: The answer is yes. Just as the software delivery process becomes smoother, more efficient and produces fewer defects at the higher levels of DevOps evolution, so an organization's security posture becomes stronger when security is completely integrated into the software development cycle.

Successful DevOps practices help the different teams involved with software delivery feel more like colleagues and less like adversaries. It's the same with integrating security: It fosters a feeling of cooperation between the security, dev and ops teams. People perceive that security is more valuable to the business where there's more integration, and they're also more likely to perceive security as a responsibility that's shared within their organization. Of course, when people hold these beliefs, it's even more likely that security will be treated as a priority across the technical groups.

Why is it so hard to integrate security?

A cynical, but not uncommon view is that security practices are little more than security theater: designed to avert blame rather than measurably improve security posture. Security is also frequently viewed as a bottleneck to deployment because security checks mostly happen at the end of the delivery lifecycle. These checks are often manual and cause delay; if issues are found, there's unplanned work for dev, test and ops teams, causing further delays and frustration.

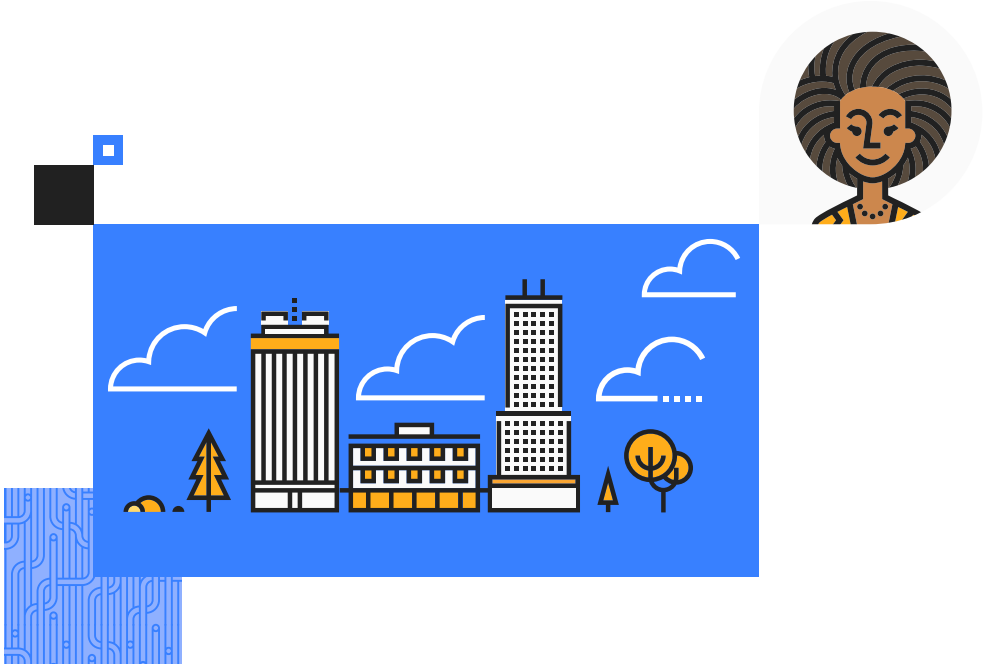
Often there's pressure to deploy a feature, which leads to compromises that create risk for the business. People concerned about deadlines or hot competition may decide to release a product with the security issue unresolved, intending to fix the issue in a subsequent release. At best, this creates a delay period when the code may be quite vulnerable. It's also not unusual for recognized security issues to slip people's attention entirely once the product is out the door.

Adopting DevOps practices opens the door to integrating security into the entire software delivery lifecycle — in fact, we'd argue security integration should be a natural part of any DevOps effort. But many companies are getting stuck in the middle stages of DevOps evolution, making it difficult to begin security integration.

Why do so many organizations evolve to a certain point on their DevOps journey, but no further? Nic Whittaker, head of platform engineering and DevOps at Virgin Atlantic, is responsible for overseeing the transformation towards increased agility across technology delivery and operations in his company. As he put it, "Getting through the middle is hard. No one ever talks about how hard it is."

It doesn't help that, even as you're doing the hard work, the goal posts keep shifting. Technology and practices that were state-of-the-art eight years ago, when we first published the State of DevOps Report, no longer give any company a competitive edge.

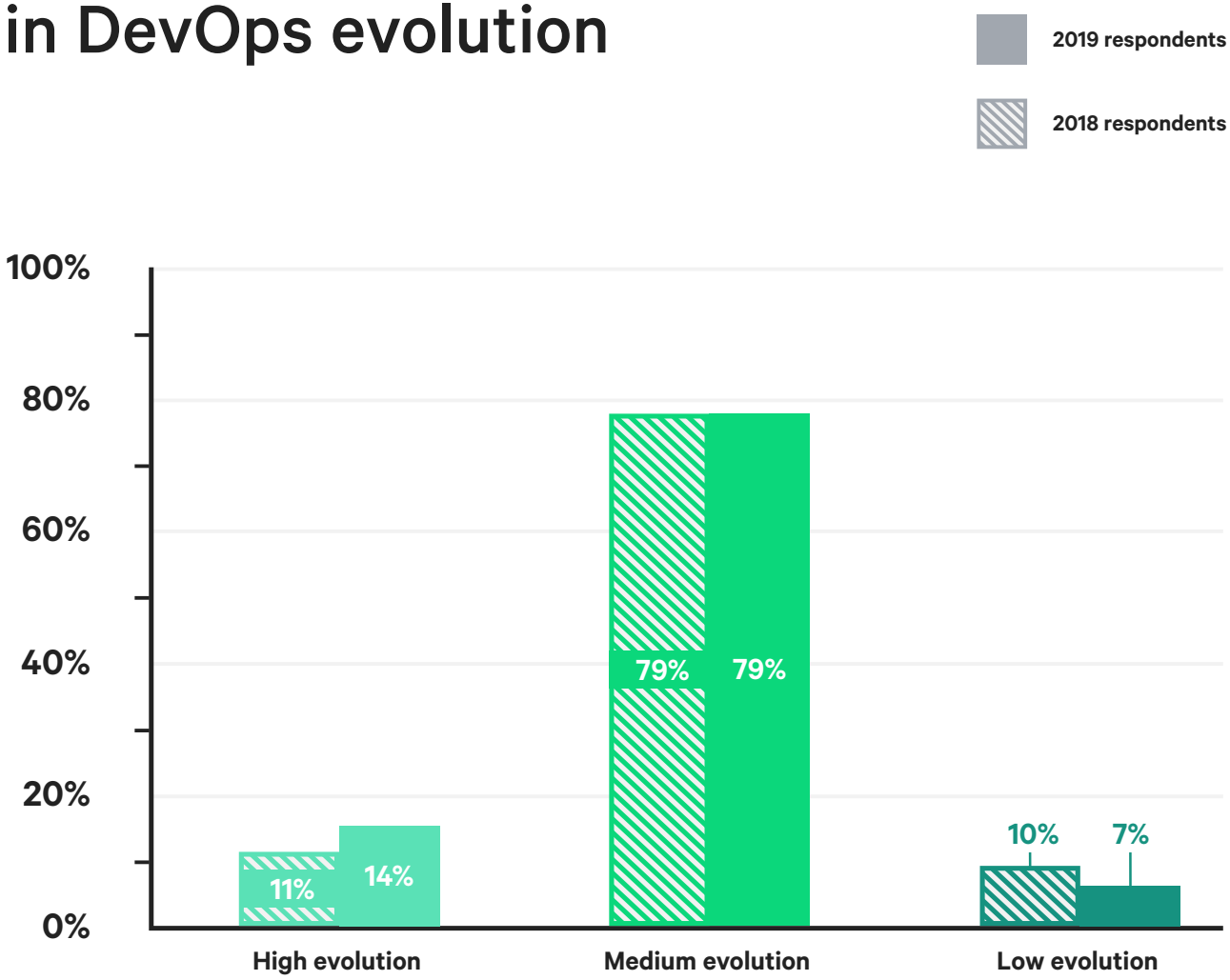
To help people think of security integration as a normal part of the DevOps journey, we wanted to understand how the DevOps evolutionary model maps to levels of security integration. We included questions this year that would allow us to validate that the DevOps evolutionary model still holds, and it does. We also designed questions that would tell us how deeply security is integrated into an organization's normal software delivery process. The answers allowed us to define levels of integration in a meaningful way that we could then consider alongside stages of DevOps evolution.



We found parallels in our analysis of both the DevOps evolutionary model itself and the degree of security integration that takes place as DevOps evolves:

- Efforts will stall in the middle as things get more complicated. This is also where the business needs to be flexible. It must change its processes to allow teams to collaborate with speed by removing unnecessary manual approval steps, and look for ways to implement the required controls and traceability within an automated system.
- Sharing and collaboration are the most impactful things you can do to improve your DevOps initiative, and that's just as true for the security aspects of your initiative.
- As an organization evolves, teams go from adopting automation to address local pain points — for example, automating the deployment of a single service or application — to automating business processes that touch many teams. Automated incident response and self-service capabilities are examples of the higher-level automation that we see in the later stages of DevOps evolution. As security practices become more integrated within the business, they also span more teams.

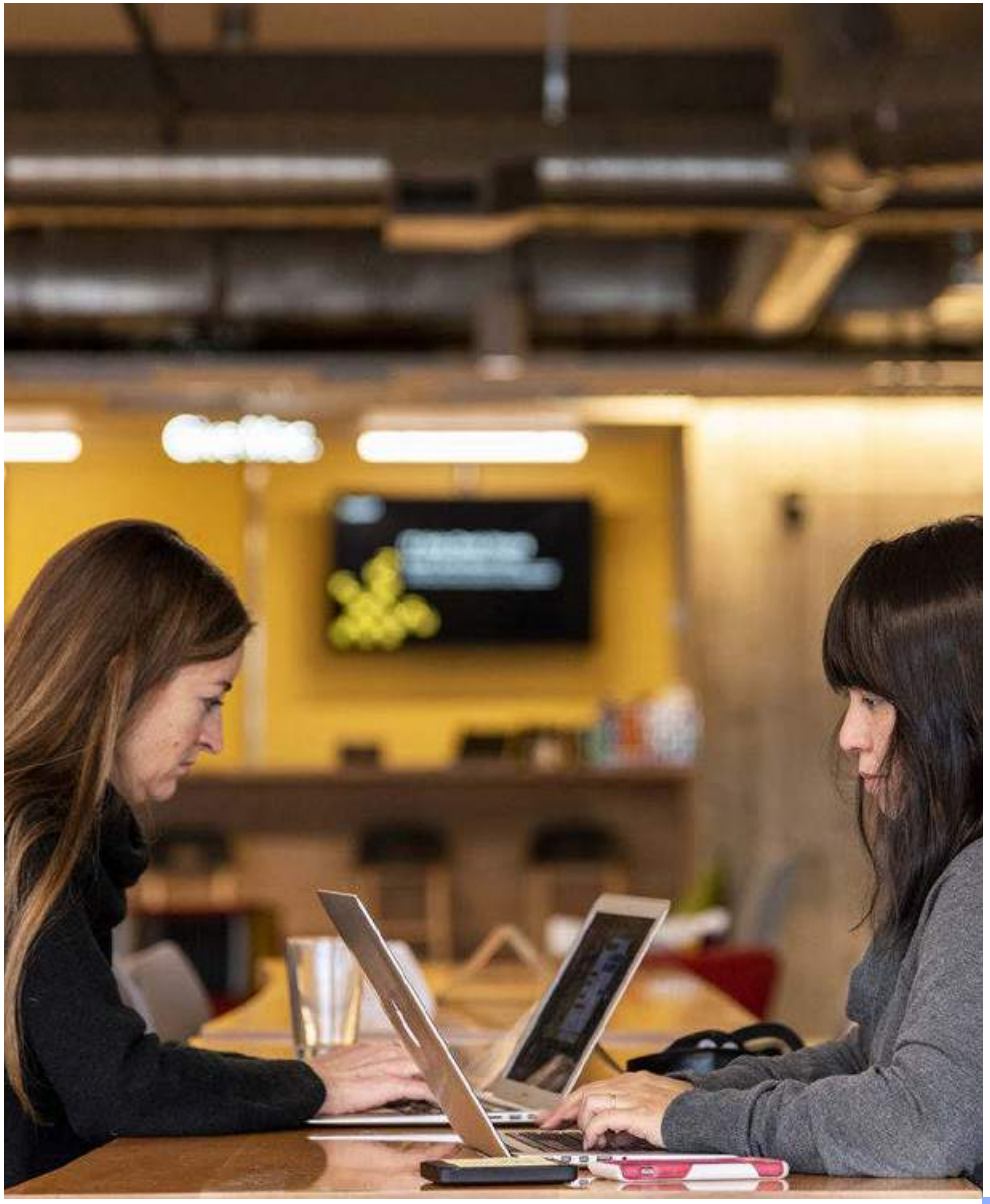
2018 vs. 2019 respondents in DevOps evolution



Stuck in the middle?

We found that the vast majority of firms surveyed this year (79 percent) are in the group we characterize as Medium on the DevOps evolutionary scale — the same as last year. Fourteen percent of the overall sample were in the High group, an increase of three percentage points over last year.

This shows that getting to the middle is relatively straightforward, but advancing beyond the middle is still challenging. To progress out of the middle stages, you should focus on measuring both business outcomes and metrics that show how day-to-day toil is being reduced and alleviated (planned vs. unplanned work, deployment pain, Severity 1 incidents, etc.). Being able to visualize your progress when things still seem hard can be a powerful motivator, and just as important, can make it much easier to see what should come next, thus leading you forward. See the 2018 State of DevOps Report for pragmatic and prescriptive guidance to help your company make progress on your DevOps journey.



DevSecOps or Simply DevOps?

While we appreciate the emphasis on Security that the term DevSecOps brings, we're sticking to just DevOps. We believe security is an integral part of both the Dev and Ops domains. But if using the term DevSecOps helps drive heightened awareness for the importance of building security into all aspects of software delivery, we're all for it.

Plus, if we keep putting every responsibility people should do in the name, we'll run out of room for the hashtag.

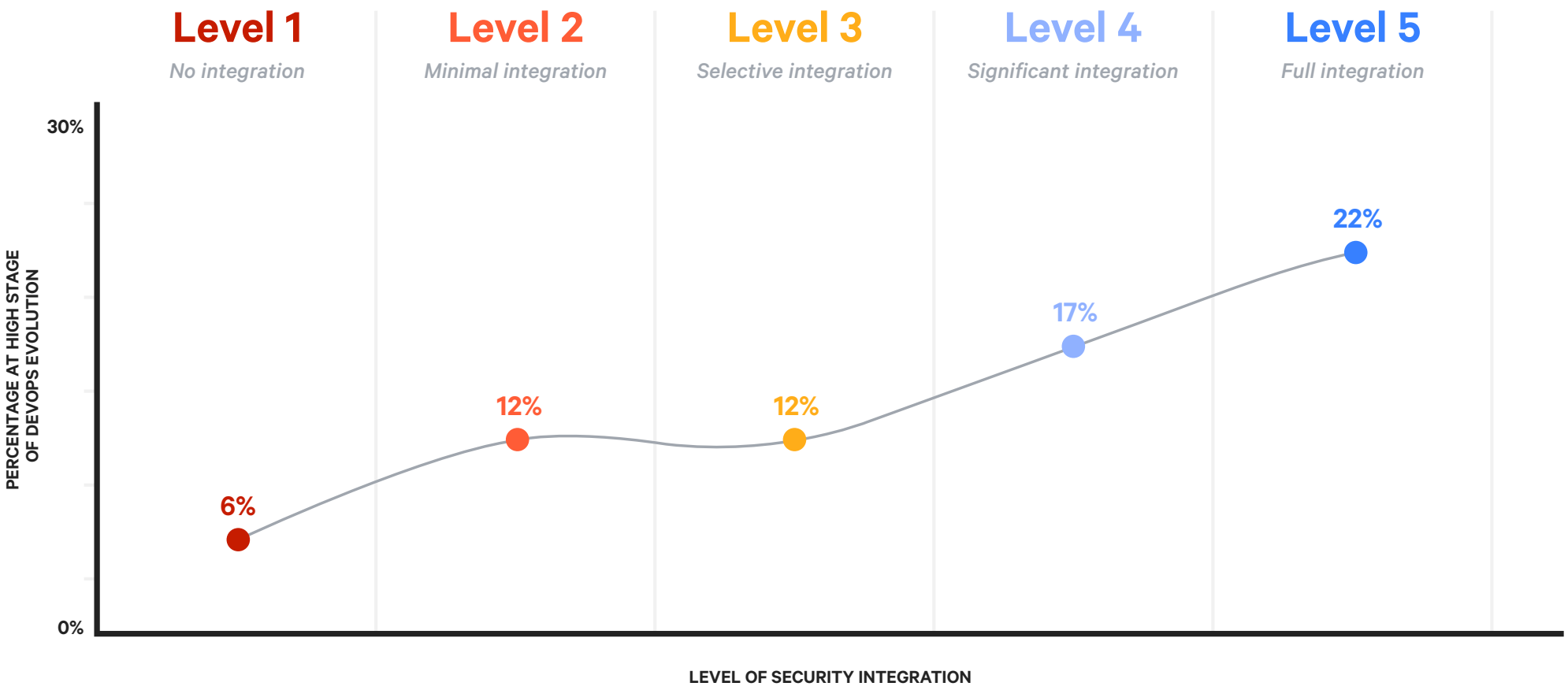
#DevSecITSMTestAutomationMonitoringObservabilityPeopleFinanceMarketingQAOps

In the 2018 State of DevOps Report, we saw that security practices become significant in the higher stages of DevOps evolution. The same correlation appears this year, but the other way around: Firms that have achieved higher levels of security integration are much more likely to be at a high stage of DevOps evolution.

Fourteen percent of this year's survey group is highly evolved. As you can see in the following chart depicting levels of security integration, only 6 percent of firms at the lowest level of security integration are at an advanced stage of DevOps evolution. But at Level 5 (the highest level of security integration), 22 percent of firms are at an advanced stage of DevOps evolution. In fact, as firms achieve higher stages of DevOps evolution, the degree of security integration becomes not only greater, but also more significant. (For statistical details around this finding, please see the Methodology section.)

This year's findings are clear: Good security practices and better security outcomes are enabled by DevOps practices. So if you want to improve your security posture, start implementing DevOps practices.

Percentage of firms at high stage in DevOps evolution

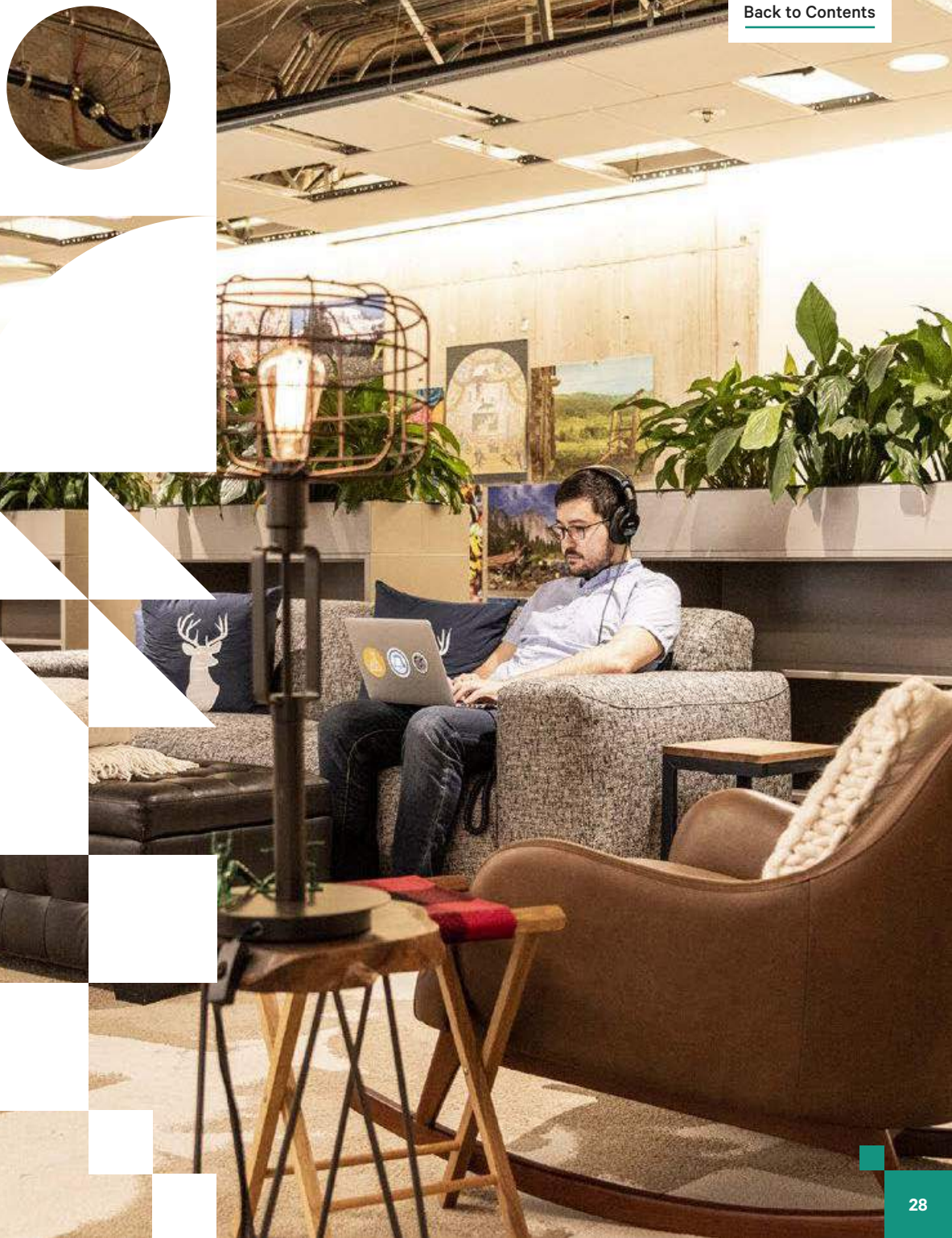


Levels of security integration

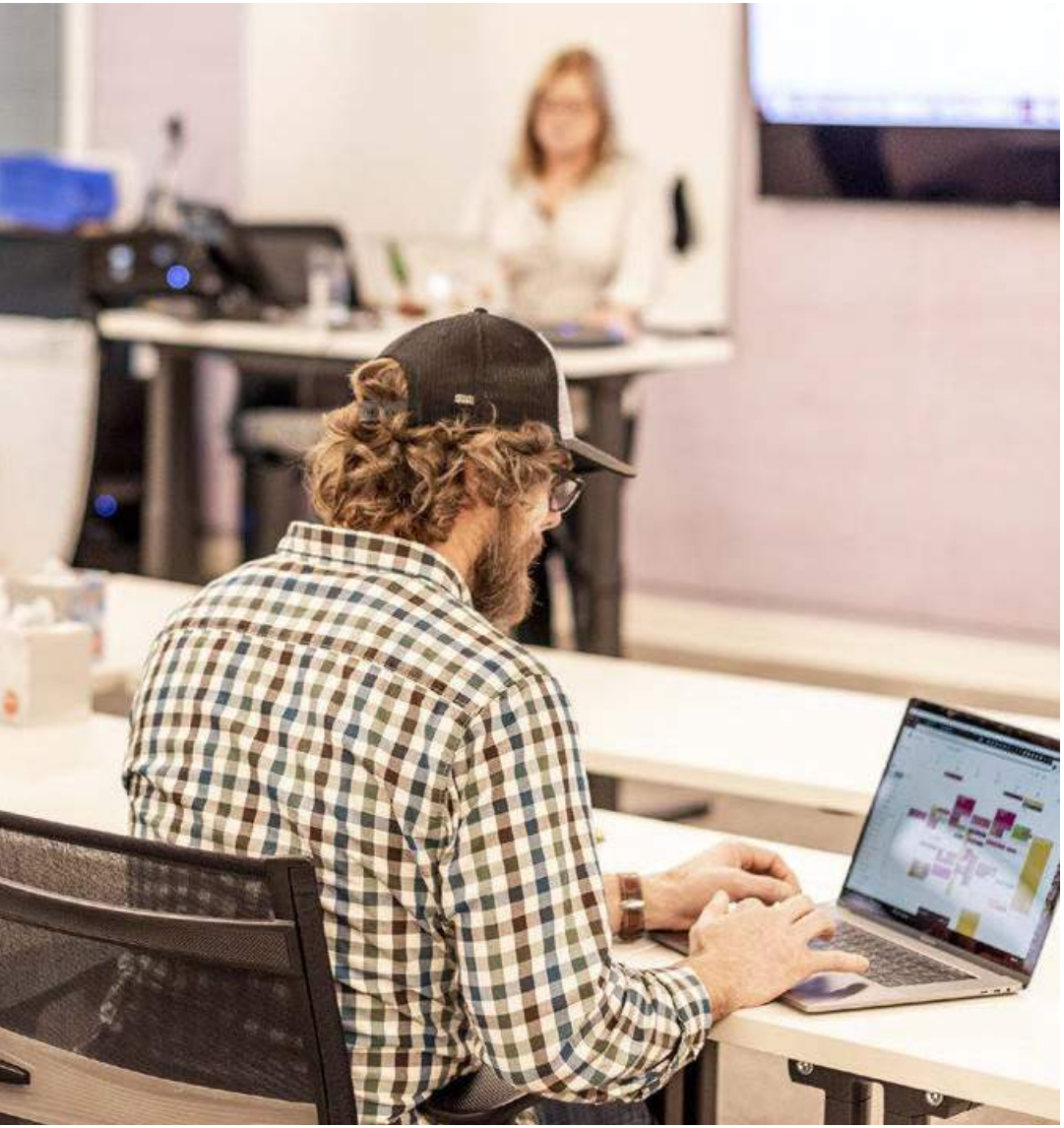
Throughout this report, you see references to the levels of security integration — Levels 1 through 5 — to categorize how much an organization's security practices have been integrated into the software delivery lifecycle.

In this year's survey we asked, "During which of the following phases of your software delivery cycle is security involved?" We defined the phases of software delivery as:

- Requirements
- Design
- Building
- Testing
- Deployment



Back to Contents



Though much of the research was focused on software delivery, we also asked about the operational aspects of security integration in production, with questions focused on:

- Vulnerability management and remediation
- Security policy automation
- Audit findings and responses

We broke respondents' answers out into five levels describing how deeply security is integrated into the software delivery cycle:

- Level 1 - No integration of security in any of the phases
- Level 2 - Minimal integration (one of five phases)
- Level 3 - Selective integration (two of five phases)
- Level 4 - Significant integration (three or four of five phases)
- Level 5 - Full integration (all phases)

It's important to note that our findings don't point to a single path for integrating security into the software delivery lifecycle. At Level 3, for example, survey respondents reported nine different combinations of practices (one such pattern: security is integrated only during the requirements phase and the testing phase). Companies are integrating security in many different ways, and their success depends on which outcomes they prioritize, as well as what is actually possible in a given situation.

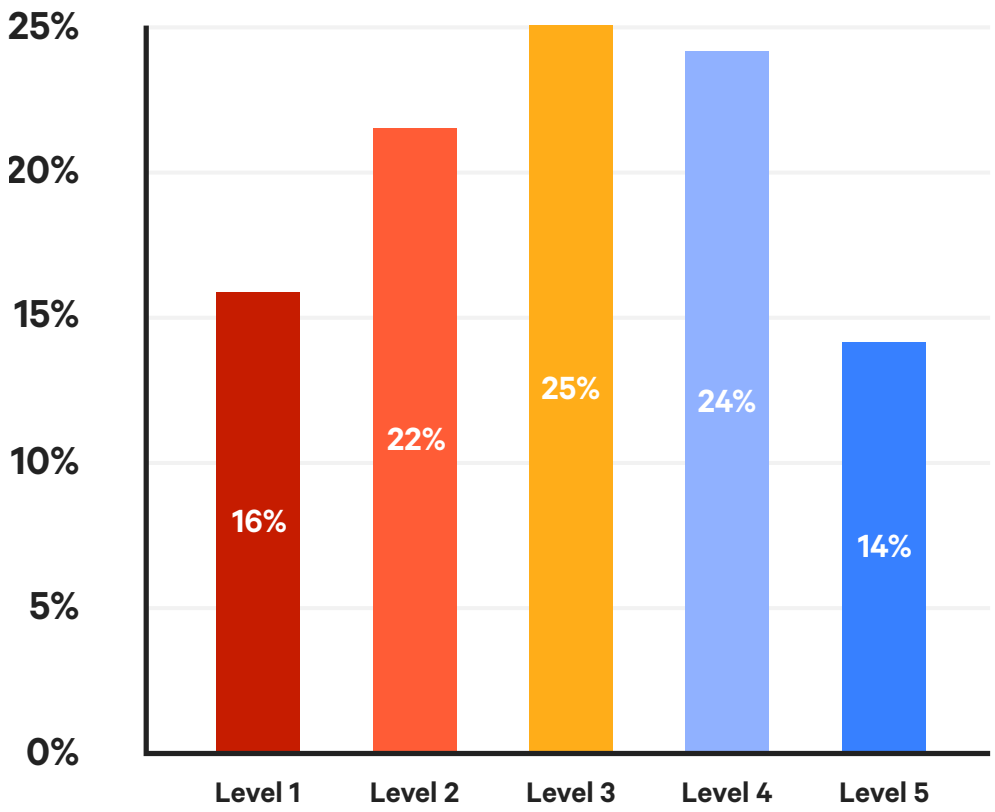
Sixteen percent of the firms where survey respondents work were at Level 1, or no integration. The majority of firms at Level 1 (75 percent) get involved in the software delivery lifecycle on an ad-hoc basis — that is, when issues are reported in production or an audit is scheduled.

Sixty percent of firms include security in two or fewer phases of their software delivery cycle. Most firms start integrating security at the testing phase. Once that's established, they progress to integrating security at the deployment phase; at this point, we say they're at Level 2, with security integrated at two phases of software development. At Level 3, most firms have incorporated security at the building phase.

As we've said above, there are a number of different ways that firms go about integrating security at each phase, but these are the most common patterns we found:

- **At Level 1, integration is ad hoc and only when issues are reported in production or an audit is scheduled.**
- **At Level 2, integration is in the testing phase.**
- **At Level 3, integration is in the testing and deployment phases.**
- **At Level 4, integration is in the testing, deployment, and building phases.**
- **At Level 5, firms are integrated at all five phases.**

Percentage of respondents at each level of security integration



5

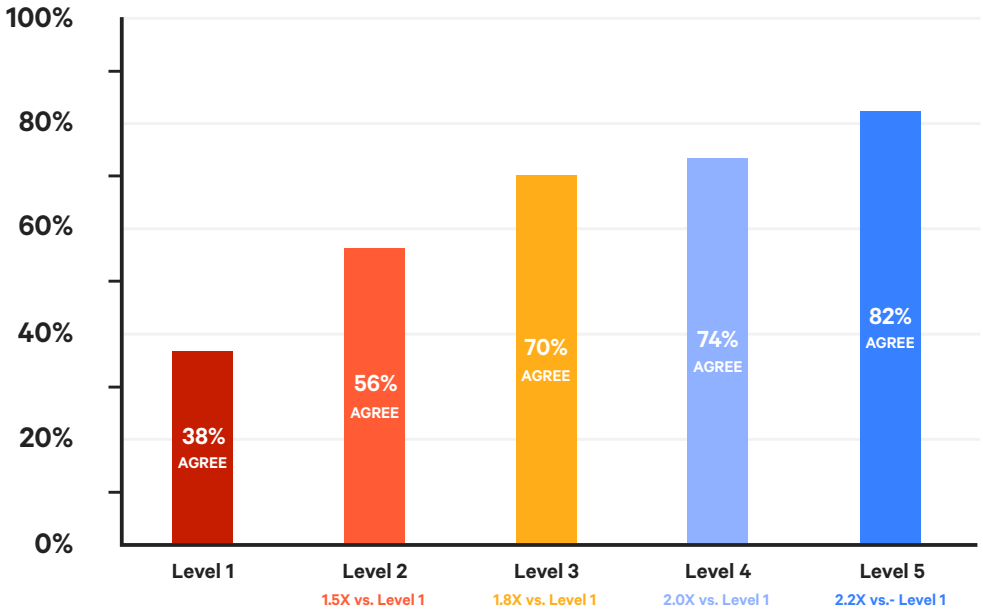
Improving security posture

In working with many organizations over the years, we've encountered a common perception that security processes and procedures are mostly security theater, with very little impact on overall security posture. This mindset doesn't help already strained relationships between the security team and other delivery teams.

Most of the existing guidance on improving security posture is tactical advice — things like implementing identity and access management, maintaining your asset inventory, controlling administrative privileges, etc. These are well understood practices, but organizations often don't implement them fully because it's hard and expensive. Or if they do, they apply rigid change control measures that slow everything down.

Neither of these approaches addresses security effectively. They come out of the mindset that security considerations are separate from software design and creation. Our survey results tell us what's really needed, and effective, is the opposite: deep collaboration between teams to include security considerations from the very first stages of software planning and design, right through every subsequent stage: building, testing, deployment, and maintenance of code in production.

As organizations go about integrating security, people need to ask: How do we make security easier to implement, as well as more agile and iterative? The answers — as well as the questioning itself — will be key to building a security-minded culture, one where delivery teams have both the knowledge and autonomy to identify, prioritize and fix security issues, while holding everybody accountable for security outcomes.



Security confidence and integration

Lacking a way to actually test just how secure any given organization in the survey really is, we decided to ask our respondents how confident they felt about security in their firm. Did they feel that their company's security policies and processes improved its security posture?

Eighty-two percent of respondents at companies in the highest level of security integration felt that their security policies and practices significantly improve their security posture. Compare this with the respondents from firms with no security integration: Just 38 percent feel the same. The biggest jump in confidence occurs when security goes from being not integrated at all to being minimally integrated (Level 1 to Level 2); we see an improvement of 18 percentage points. Full integration (Level 5) delivers an improvement of 44 percentage points over Level 1 — more than doubling confidence in an organization's security posture. Even with minimal integration between security and delivery teams, the benefits are felt across teams.

We wanted to find out which practices actually drive improvements in security posture. Not surprisingly, practices that are specific to a single team, such as penetration testing and static code analysis, do not have as much impact on improving security posture as the practices that require cross-team collaboration.

This is parallel to the DevOps evolution model, where we found that cross-team sharing is key to scaling DevOps success. The foundational DevOps practices — the ones with the most significant impact across the entire DevOps evolutionary journey — all depend heavily on the principle of sharing.

The top five practices with the greatest impact on security posture share some other attributes too: They make security easier to implement, more agile and more iterative. To take a couple of examples: threat modeling is a creative exercise that helps teams gain awareness of threats and get on the same page with testing and remediation plans. Automated security tests make the team more agile by removing the gating process of manual security checks before deploying to production. You don't have to do anything special to plug them into your continuous delivery pipeline, given the mechanism is already there — they can be treated like any other test.

The biggest takeaway is that improving your security posture isn't just about moving some security practices to an earlier phase of the software lifecycle. It's about adopting a different way of working, one that emphasizes cross-team collaboration and shared empathy. DevOps, in fact.

Practices that affect security posture in rank order

Below we've listed all of the practices we tested in order of their impact on security posture. While all of the practices are an important part of a strong security program, the top five practices highlight just how critical cross-team collaboration is for improving security posture.

1. Security and development teams collaborate on threat models.
2. Security tools are integrated in the development integration pipeline so engineers can be confident they're not inadvertently introducing known security problems into their codebases.
3. Security requirements — both functional and non-functional — are prioritized as part of the product backlog.
4. Security experts evaluate automated tests, and are called upon to review changes in high-risk areas of the code (such as authentication systems, cryptography, etc.).
5. Infrastructure-related security policies are reviewed before deployment.
6. Security personnel review and approve major code changes before deployment
7. Domain-specific tests (e.g., tests that assess application with security-aware context, such as the way your application does authentication or has access to data)
8. Developers can provision a security-hardened infrastructure stack on demand.
9. Security requirements are treated as design constraints.
10. Security personnel review and approve minor code changes before deployment.
11. Security review occurs after new application code is released to production.
12. Penetration testing (e.g., vulnerability trigger or hacker tool testing)
13. Infrastructure is provisioned and configured automatically using security-approved procedures.
14. Dependency checkers (e.g., tools that check for the latest version of npm packages, RubyGems, etc)
15. Static code analysis
16. Security requirements are treated as a design constraint.

Security and development teams collaborate on threat models.

Threat models are a collaborative exercise where representatives from delivery teams, security, business stakeholders and possibly other parts of the business get together to talk about what assets an attacker would be after, and how that adversary might craft exploits to gain access to them. Then the team catalogs those potential threats to both the application and supporting infrastructure, and brainstorms countermeasures to mitigate them. When security and delivery teams collaborate on threat models, they are able to build security into the design of the application, discuss security impacts and trade-offs (such as cost, timeline and scope). Threat modeling helps to identify potential issues at the earliest stage of planning and design, inform the security testing plan, and most importantly, the collaboration and discussion build empathy and trust between developers, security, and business teams.

Security tools are integrated in the development integration pipeline so engineers can be confident they're not inadvertently introducing known security problems into their codebases.

Tools are not going to solve all of your security woes, but they will automate routine tasks and make life easier for security teams, allowing them to exercise creative thinking — as an attacker would — to find security holes.

There is no one tool that will capture all your security issues, so a comprehensive security program should include testing at all stages of the software delivery lifecycle. Start with developers unit testing their code in compliance with secure coding standards and verifying that no vulnerabilities exist before integrating changes into the build, while testing engineers look for application-level vulnerabilities. Application security testing (AST) tools (static application security testing, interactive application security testing and dynamic application security testing tools), software composition analysis (SCA) tools, and practices such as threat modeling, unit testing and pen testing help security and development teams catch security risks and vulnerabilities before production, when they are cheaper to fix.

Security requirements, both functional and non-functional, are prioritized as part of the product backlog.

Many software development teams today work according to Agile principles, with scrum teams releasing code in short, frequent, iterative cycles.

This is great for learning and continuous improvement, as well as faster product delivery. But once you're working the Agile way, it's hard to implement security requirements using traditional methods — the paper-based policies, manual security checks at the end of the development cycle, and so on.

Embedding a security expert in the scrum team can resolve this disconnect by enabling greater knowledge sharing and expanding everyone's awareness of security concerns. A security expert can come from the security team, but doesn't have to.

Security-minded scrum team members provide guidance on implementing security controls, including providing the definition of done and user story requirements for product backlog items. This is a big cultural shift for organizations, and necessary, creating the collaboration and feedback loops that allow teams to build safer and more resilient software.

Scaling this approach, however, can be difficult for large enterprises that have one centralized security team supporting hundreds of application development teams. You could have dedicated application security engineers embedded in the app dev teams, and they should need to routinely interact with the security team. This approach, however, is also difficult to scale, given the shortage of application security engineers.

The most scalable option for many organizations is incentivizing developers to be responsible for secure development. Additionally, ops would be responsible for secure deployment and infrastructure. And then you make sure both the dev and ops teams are responsible for mitigation.

Infrastructure-related security policies are reviewed before deployment.

It's no surprise that one of the top CIS controls is about hardening configurations for applications and operating systems ***(see CIS Control 5)***. System hardening is an ongoing process, not a one-and-done task. While benchmarks, like the CIS Benchmark, are very useful for providing prescriptive guidance, they should be seen as a starting point. Every organization's applications and environments are unique, and hardening measures need to be balanced with functionality and the ability to deliver the application or service. Security teams may not be aware of the operational concerns associated with specific security settings, and operations teams may not be aware of security holes caused by not enforcing specific settings. By collaborating before deployment, security and operations teams can share knowledge and make informed trade-offs.

Security training for software engineers

Want to know how to do security training right? Read this blog post by Tad Whitaker, "Why we hired two DefCon hackers to teach our team to think like deviants."

Find it here:

https://circleci.com/blog/why-we-hired-two-defcon-hackers-to-teach-our-team-to-think-like-deviants/

(Tad is a CircleCI security engineer, the founder of Day of Shecurity and the organizer of the Bay Area OWASP Meetup.)

Security experts evaluate automated tests, and are called upon to review changes in high-risk areas of the code (such as authentication systems, cryptography, etc.).

In the traditional approach to ensuring application security, code is pushed to a user acceptance testing (UAT) or staging environment (or sometimes production) where it is then manually inspected by the security team. Inevitably, an issue is discovered that requires a fix that kicks off the whole cycle again. Moving testing to earlier in the software delivery lifecycle means that bugs get caught earlier when they're much cheaper to fix. Just as you would automate routine quality tests, so can you automate routine security checks. For example, you can automate tests for known weaknesses and misconfigurations, like those identified in the ***OWASP Top 10.***

Vulnerabilities can live anywhere in the stack: in the application code, third party components, APIs, servers, databases, network devices, firewalls, cloud storage, etc. The potential attack surface is vast, making automated testing for known weaknesses your best defense against attacks.

There will always be new threats, edge cases and changes to the stack that require you to reevaluate your tests. Security experts should continually collaborate with other teams — development, operations, network, storage, middleware, cloud, etc. — to share knowledge and ensure these tests are up to date.

Automation of routine security checks gives time back to security teams to do things like:

- Collaborate with other delivery teams to secure and test applications during development and at deployment.
- Create feedback loops to ensure that misconfigurations discovered during testing are properly patched, and procedures are revised to avoid the same misconfiguration in the future.
- Evaluate tests when the application or environment changes and look for edge cases.
- Review changes in high-risk areas of the code (such as authentication systems, cryptography, etc.).

Testing in production

A few years ago, testing in production was a meme. It's quickly becoming a way of life. In SaaS environments and environments of significant complexity, the ability to replicate all services, data, scale, traffic and other variables is somewhere between too expensive and impossible.

Companies are switching to testing in production using deployment safety techniques such as canary deployments, blue/green deployments, or other processes to deploy changes and validate them before pushing them out to the entire environment.

This reduces deployment risk, and is enabled by designing for resilience to allow continuous testing.

In the security space, you may hire a team to run penetration testing on your production application. This won't replace security tests in continuous integration, but will augment them.

Practices and their impact on security posture

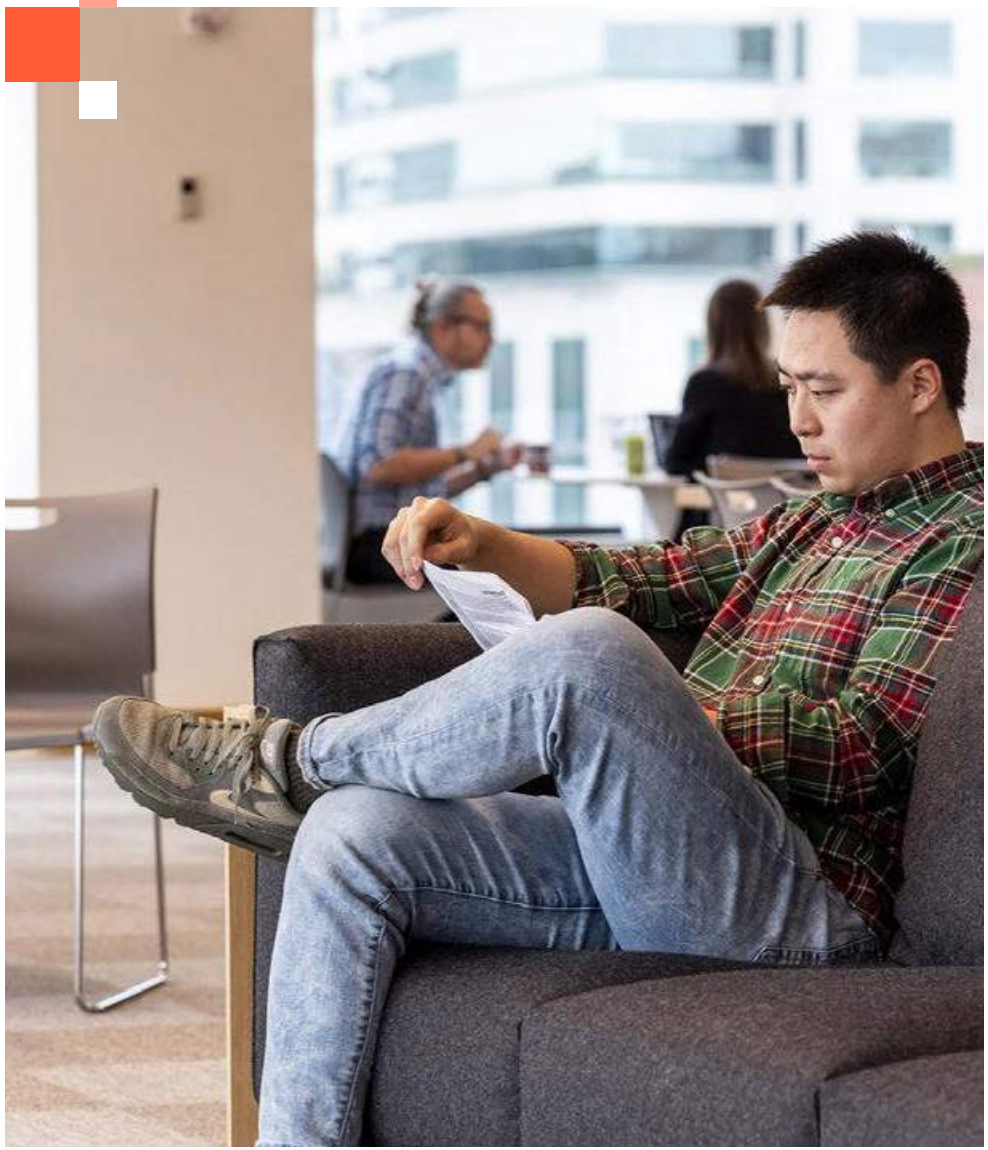
In the following quadrant, we show practices according to how important they are for your security posture and how frequently they're used. You've probably seen quadrants like this one before; in this one, the further right a practice is, the bigger the impact it has on security posture improvement. The higher up the practice is, the more frequently it's used. So in the top right quadrant you'll find practices that are frequently used and also highly impactful to security posture, and in the bottom right quadrant, practices that are highly impactful but less frequently used.

The practices that are both higher in use and have a higher impact on security posture are:

- Infrastructure-related security policies are tested and reviewed before deployment.
- Security requirements are prioritized as part of the product backlog.

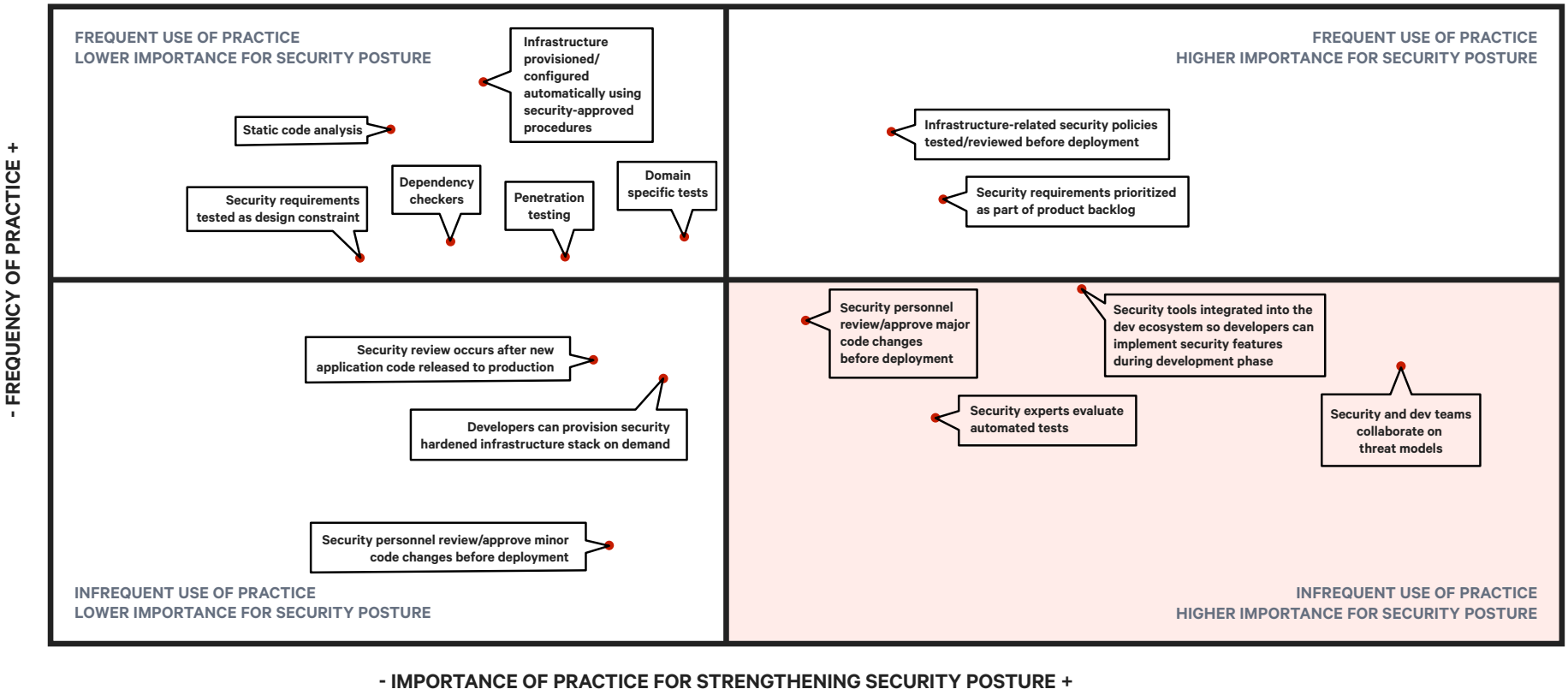
The practices that are used less frequently and have a high impact on security posture are:

- Security personnel review / approve major code changes before deployment.
- Security experts evaluate automated tests.
- Security tools integrate into the development ecosystem so developers can implement security features during the development phase.
- Security and dev teams collaborate on threat models.



Security practices and their effects on security posture

Check the lower right quadrant to see if you recognize any practices here as things you aren't doing yet. These are the practices you should focus on to have the greatest impact on your company's security posture.



6

Integrating security into the software delivery lifecycle leads to positive outcomes

We started with the core hypothesis that by shifting security left — building security into the software delivery lifecycle from the beginning — organizations would see positive outcomes. Good security practices require good collaboration and feedback loops, and just like operational considerations, should be taken into account from the beginning of the software delivery lifecycle right through every subsequent phase.

Performance outcomes

Deployment frequency

This year we asked two questions around deployment frequency: “How often are you capable of deploying to production?” and “How often do you deploy to production?” As a follow-up, we asked if deployment frequency was limited by business needs or by technology and process.

It is encouraging to see that even those organizations where security isn’t integrated are able to deploy on demand, or at least twice per day. When we first started the State of DevOps survey eight years ago, improving deployment frequency was a big concern for nearly everyone, and for most organizations, deploying even once a month was a stretch.

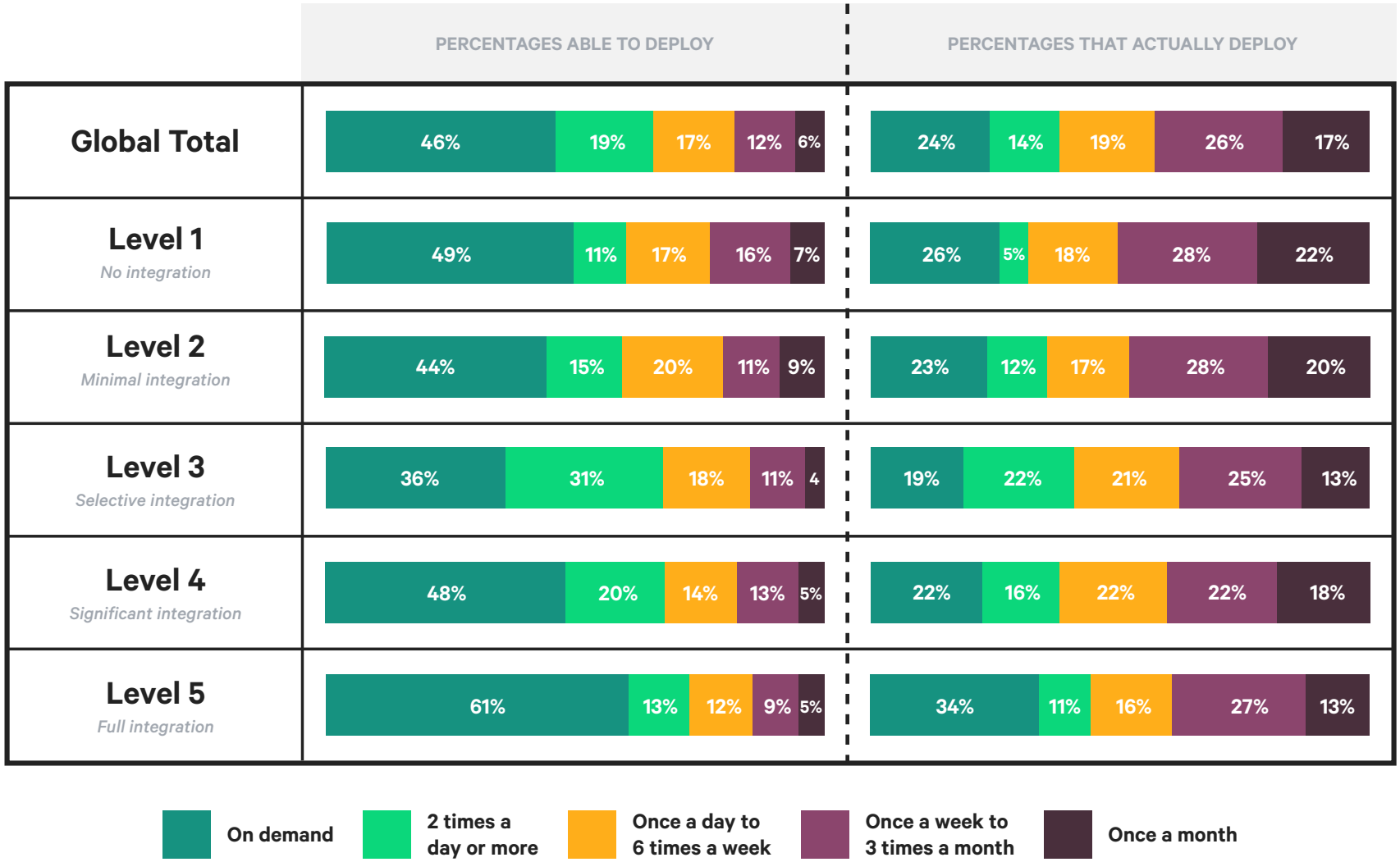
Over the years, we’ve seen deployment frequency steadily improve, with high performers deploying multiple times per day (2017 State of DevOps Report) and medium performers deploying multiple times per week. Nowadays, it’s not uncommon to hear of teams that can actually deploy more frequently but are constrained by the business (for example, waiting for the results of a user engagement experiment, marketing launch requirements, etc.).

At Level 3, on-demand deployment capability declines significantly before it begins improving again in Levels 4 and 5. We believe this is due to more interaction with security, as manual approvals that were previously done post-deployment have been shifted to before deployment. Going through the pain of this middle stage is a natural, normal part of shifting left. It’s also necessary for building trust and continuing to satisfy compliance in advance of automating these same auditing and approval processes. Just know that things do get better: **Sixty-one percent of organizations at Level 5 (those with complete security integration) can deploy to production on demand.**

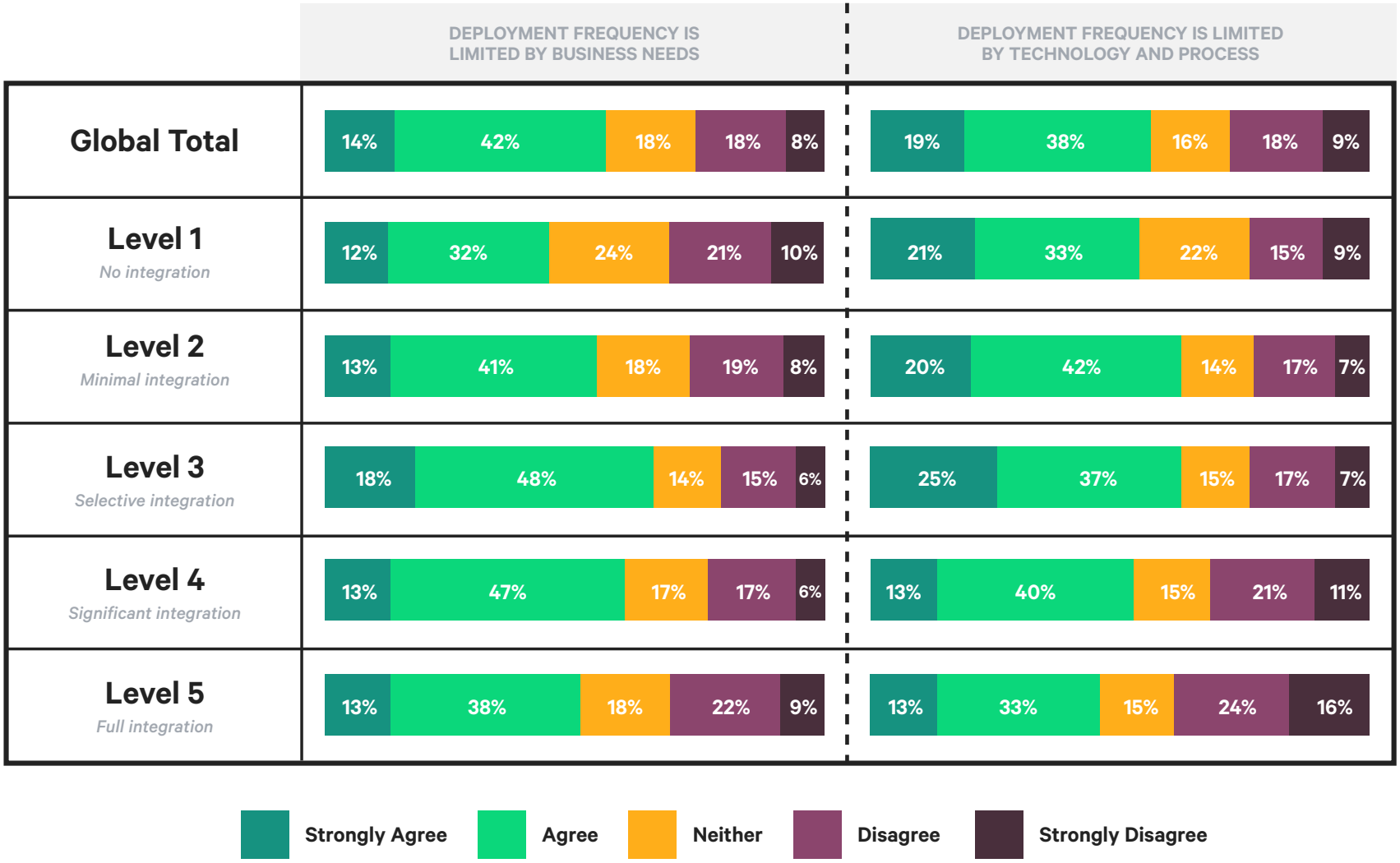
This is significantly different at a statistical level from Levels 1 through 4.

We found it interesting to notice that for organizations at any level of security integration, only about half of those that can deploy to production on demand actually do deploy on demand. Thirty-four percent of Level 5 organizations actually deploy on demand, compared to the 61 percent that are capable of doing so.

Ability to deploy vs. actually deploying



Limitations on deployment frequency



Time to remediate critical vulnerabilities

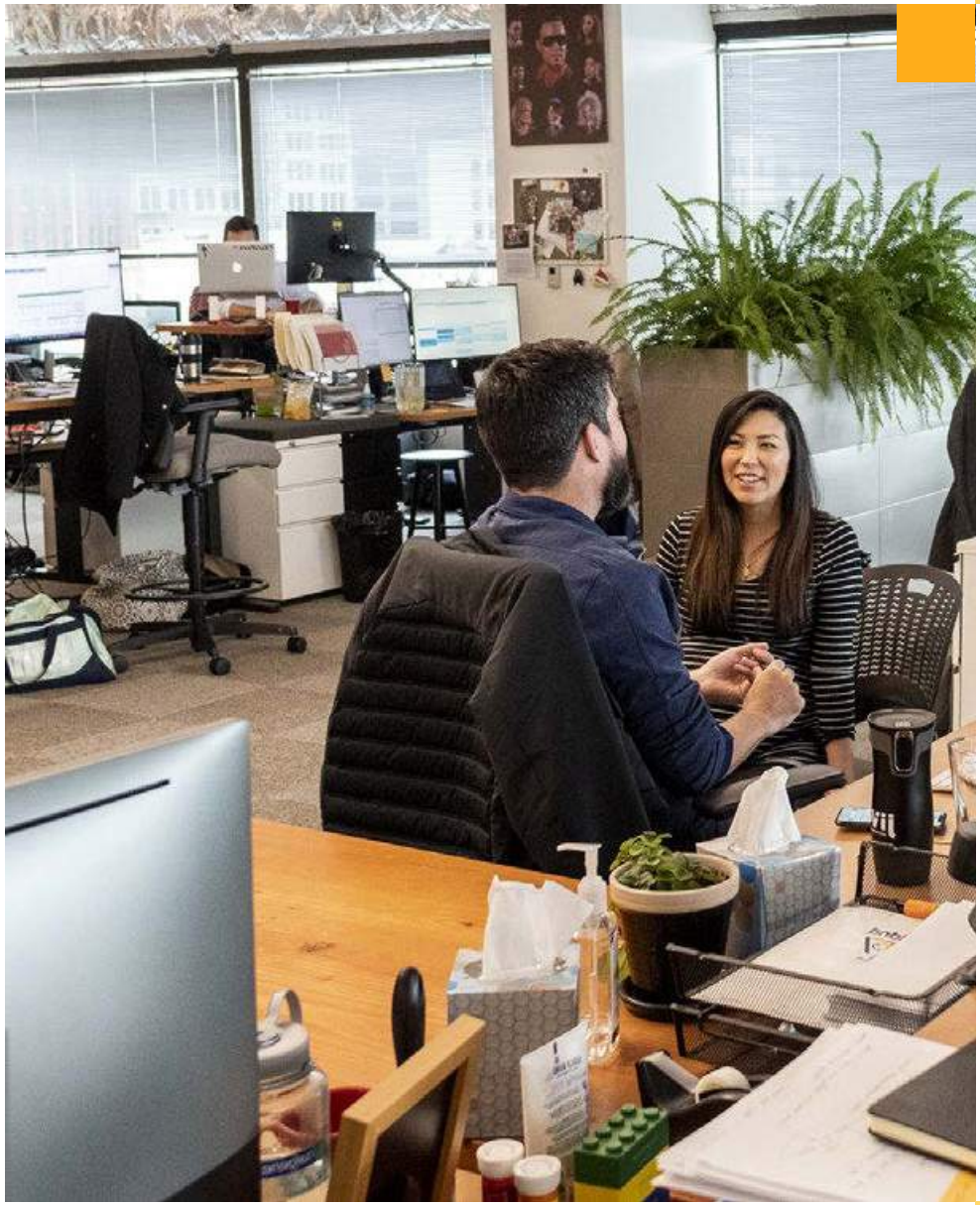
One of our hypotheses was that at higher levels of integration, time to remediate critical vulnerabilities would dramatically decrease. We were surprised to see that was not the case. Very few respondents are able to remediate in less than one hour. Most are able to remediate in less than one week.

- **Only 7 percent of total respondents are able to remediate a critical vulnerability in less than one hour.**
- **Thirty-two percent of respondents are able to remediate in one hour to less than one day.**
- **Thirty-three percent of respondents are able to remediate in one day to less than one week.**

The differences in remediation times between low and high levels of security integration are both important and statistically significant:

- **Eleven percent of respondents at Level 5 are able to remediate in less than one hour compared to 6 percent of respondents at Levels 2, 3 and 4. Though this difference is just 5 percentage points, it is statistically significant.**
- **Thirty-one percent of respondents at Level 1 are able to remediate in one hour to less than one day, compared to 38 percent of Level 4 respondents and 39 percent of Level 5 respondents. This improvement of 7 to 8 percentage points may not seem huge, but it is statistically significant.**

The J curve we see for other performance outcomes also describes this ability to remediate critical security vulnerabilities. The cost of integrating security practices is felt most in the middle stages of security integration, and the payoff happens in the later stages.



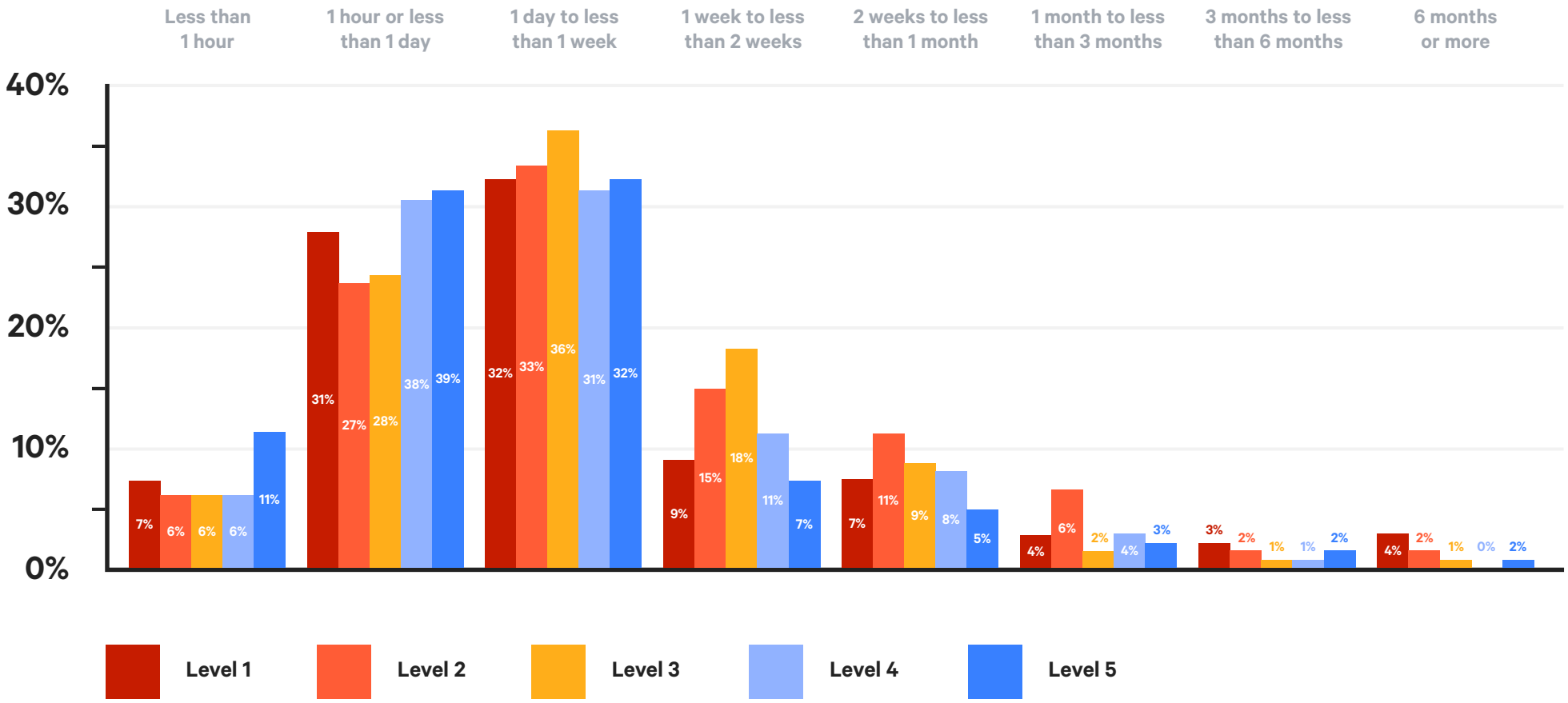
We suspect this is because firms at Level 1 have fewer handoffs and stakeholders to work with, and for those at Level 5, thorough and complex business processes are properly modelled and accounted for.

The improvement in time to remediation is important beyond statistical significance. Uncorrected vulnerabilities leave the business open to attack. Having a defined vulnerability management process and the ability to quickly deploy a change reduces risk and makes the process less disruptive for delivery teams, allowing more improvements to be made in other areas.

We suspect the reason time to remediation isn't quicker is the many steps and handoffs in the vulnerability management and remediation workflow. First, the security team typically will use a scanning tool to obtain vulnerability information. However, they're often referring to stale asset inventory data, usually in a spreadsheet with outdated host and application information. Then there's the back-and-forth between ops and security admins to gather the current inventory (which may be a non-trivial event when you're dealing with a large number of assets). Then the security team reruns the scans. Once the scans are complete, they hand off a report of affected items to another team, which then has to prioritize which assets they will fix, locate the assets, test and deploy patches. Typically, this entire process is manual or only semi-automated.

Automation for security remediation and evidence capture is something we see when firms are further along their security integration journey. The lessons and patterns uncovered during deployment automation can certainly be applied here, as pushing out security updates often rides on top of an automated deployment mechanism.

Security integration and time to remediate critical security vulnerabilities



Process improvement outcomes

Prioritizing security improvements over feature delivery

A common security trade-off is feature delivery being prioritized over security requirements. In order to hit deadlines, organizations may be forced to choose between fixing a security issue and pushing out a feature they promised to a customer.

We found that firms with deeper security integration were able to prioritize fixing critical, high and medium-level security issues over feature delivery more effectively than their peers. We believe that's because the awareness of security and its importance has spread in these organizations far beyond just the security and operations personnel, and that safeguarding code and data has become a cultural value.

This makes sense when you consider that organizations with deeper security integration are also the firms that have reached higher levels of DevOps evolution. They've already been through the hard work of expanding DevOps principles and practices across multiple teams, and security concerns are part of that picture.

In addition to prioritizing security issues, companies that have integrated security deeply into the software cycle are also more effective at prioritizing automation of security controls. Automation of security controls may be included in non-functional requirements for feature delivery, even though the controls may be put in place by teams supporting feature delivery, rather than the team developing the feature. This means that as part of specifying a feature, the security aspects, threat models, compliance plans, and patching lifecycle are defined along with the feature's functional requirements, and they ship alongside the feature — not as afterthoughts after the feature is in production and being managed by an ops team.

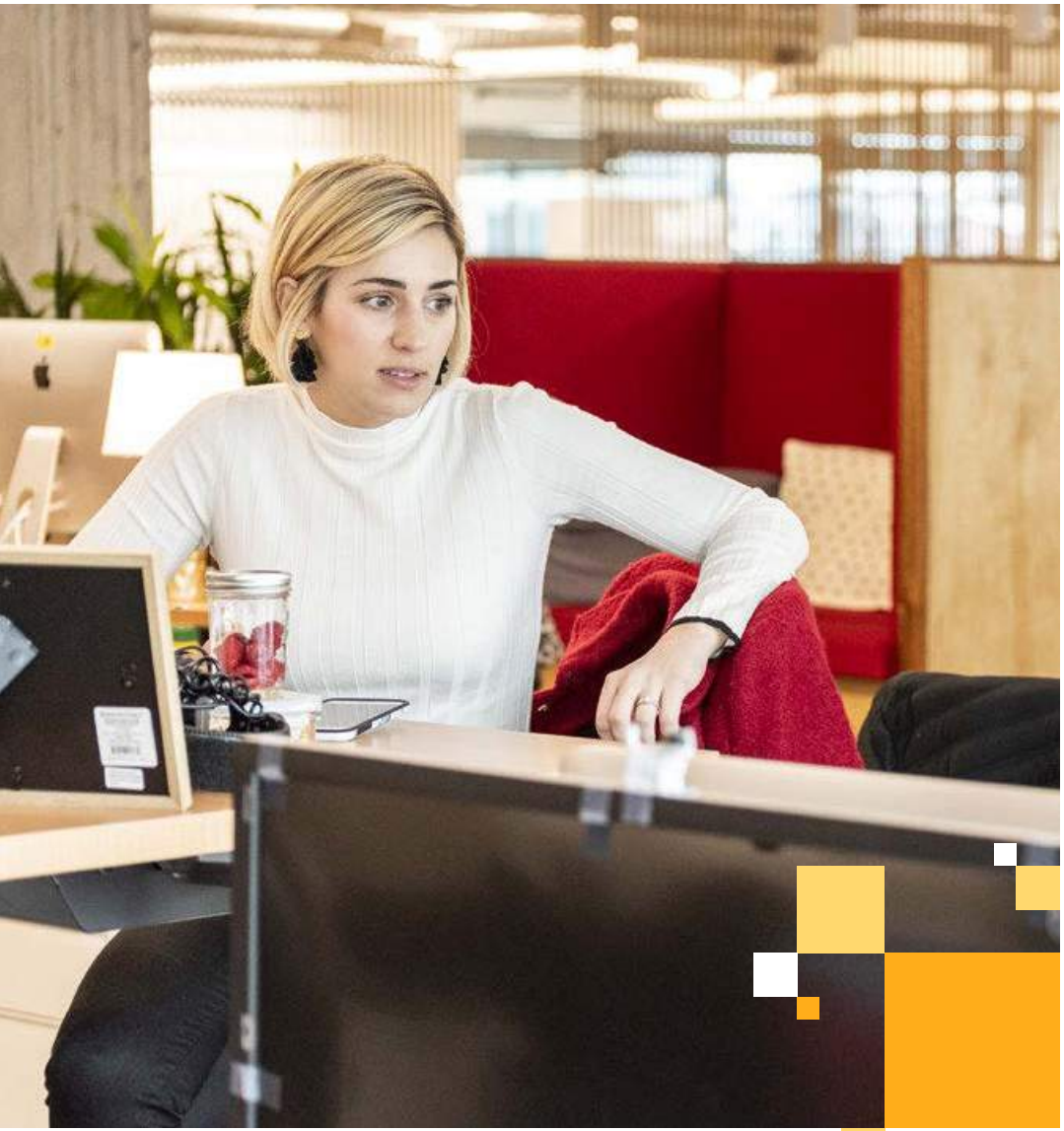
Branded vulnerabilities

Since Heartbleed, it's become a common practice to brand vulnerabilities with a cute name, a logo, a website and possibly a t-shirt. While it's easy to poke fun at this phenomenon, it does appear to have heightened awareness and driven demand for better security. A wider range of news organizations move quickly to report on a vulnerability when it has an interesting name, and so awareness spreads far beyond the usual confines of InfoSec circles.

This raised awareness isn't a bad thing. When vulnerabilities are named and branded, organizations are asked about their risk and patching of these vulnerabilities much earlier than when a vulnerability isn't branded. Thus many organizations likely address the vulnerability more quickly, making their systems, data and their customers' data safe earlier than they would otherwise.

In the case of at least one company we know, it normally contacted the support department of one of its major vendors monthly to get the latest list of security patches. When a branded vulnerability received publicity, however, the company would call support immediately to ask for a plan for mitigation of the branded vulnerability. The urgent request was often spurred by someone higher up than normal hearing about the vulnerability and demanding an action plan for dealing with it.

This isn't to say all vulnerabilities should be branded, of course. But branding vulnerabilities certainly does seem to drive awareness and action, and that's not a bad thing.

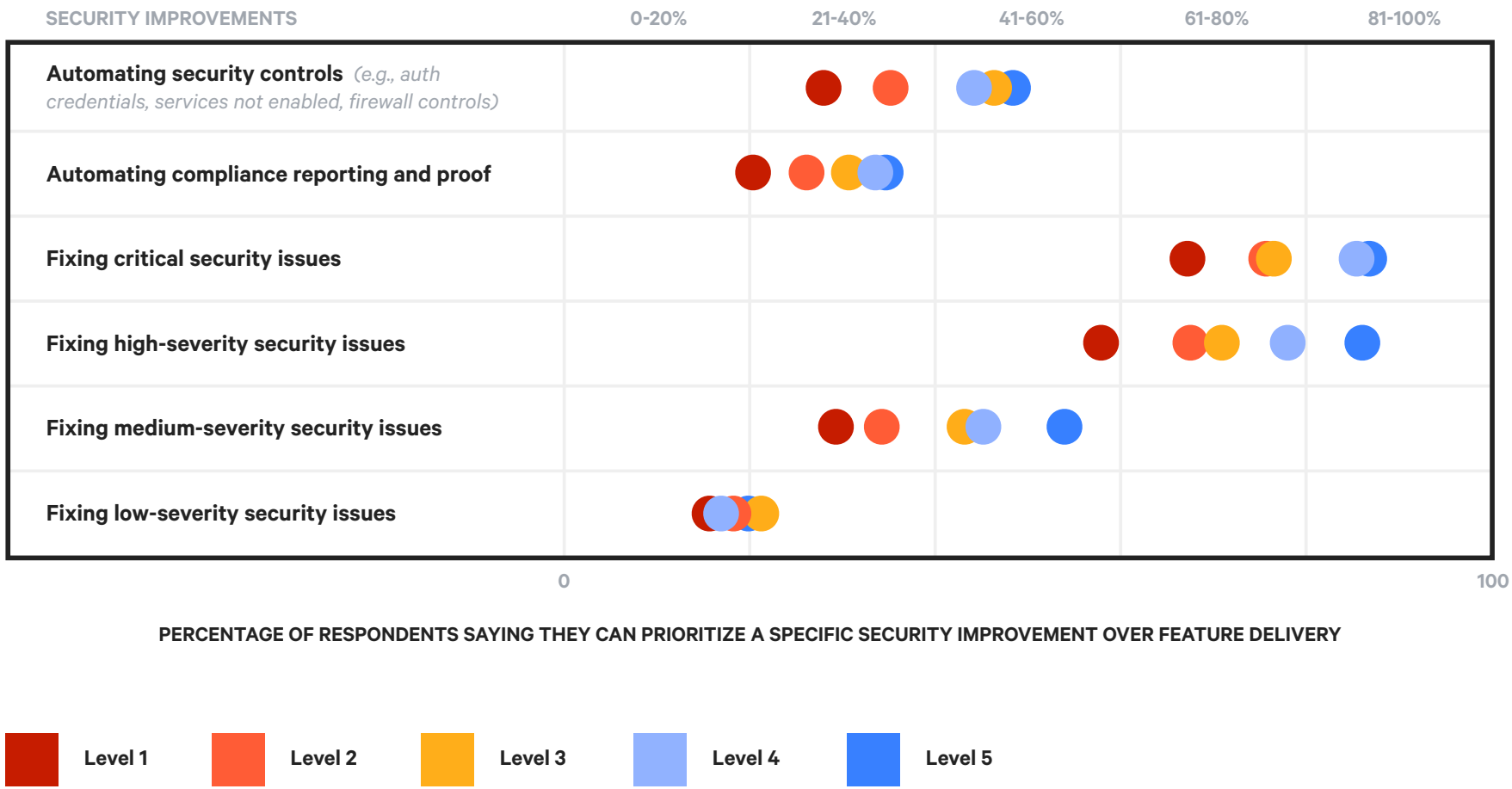


According to the CIS Top 20 Critical Controls, automating security configurations for hardware and software on mobile devices, laptops, workstations and servers is one of the most effective and foundational practices to prevent attacks. Given the strong recommendation by a recognized community of security experts, one would expect this practice to be a top priority for most companies. However, even the organizations with full integration are split between prioritizing automation of security controls (49 percent) and feature delivery (51 percent).

Firms that are serious about improving their security posture should be investing more in automation. Given the current state of our tool ecosystem, that's easier said than done, so software vendors should be working at making security automation easier.

Like remediating low-severity issues, automating compliance reporting and proof is a low priority compared to feature delivery for organizations at all levels of security integration. We aren't sure exactly why, but it could be because some group apart from the delivery team — a compliance team, for example — is responsible for compliance reporting. Another reason could be that automated systems make compliance reporting easier, anyway. So the need for an automated compliance-reporting system isn't nearly as urgent, once other systems have been automated.

Security integration and ability to prioritize security vs. feature delivery



Delivering features vs. safeguarding customer data

Overall, it's clear that feature delivery still competes head to head with security work on a daily basis, yet in many surveys beyond our own research, C-level executives say they lose sleep over their information security. This may be true, but looking at what actually happens when there's a big data breach, you can see why some business leaders do prioritize feature delivery over security. For example, in the largest publicized data breach in history, Equifax failed to protect the Social Security numbers and other private information of 147 million people. The headline figure for the settlement — up to $700 million in compensation for those affected — sounds big, but amounts to less than $5 for each person whose data was stolen. Two successive Equifax executives did have to testify to Congress, but one retired with a $90 million severance package, and the current CEO made about $20 million last year. With outcomes like this, you can see why security might take a back seat.

Even in smaller, lower-profile companies that may feel they can't afford to lose customers' confidence, the trade-offs may fall in favor of feature delivery. Sure, a security breach can result in a person losing their job, but the company still gets to deliver a desirable feature set to market more quickly than if it waited until everything was secure. The incentives for the business may not always align with security concerns, particularly if security isn't a differentiator for that company's business or industry.

Scoring a vulnerability or issue

Throughout this report, you see use of the terms critical, high, medium and low to denote how priority is assigned to vulnerabilities or other security issues. When a supplier or other third party alerts you to a vulnerability, they're likely to have done the scoring and assessment of the vulnerability for you. Many security researchers and software suppliers, including open source contributors, rely on a system for scoring vulnerabilities called the Common Vulnerability Scoring System (CVSS), This system weighs the risks and problems of a vulnerability by examining how easily an attack can be performed and what type of attack can be directly performed.

There are cases where you might want to reassess a vulnerability because of the way your firm relies upon the technology or because it's a first-party problem, but generally a CVSS score gives you a good initial indicator of the risk.

Stopping the train

We found that companies with fully integrated security practices are better able to stop a push to production for all types of security issues. As an example, our data show firms integrating security throughout the lifecycle are more than twice as likely to be able to stop a push to production for a medium security vulnerability.

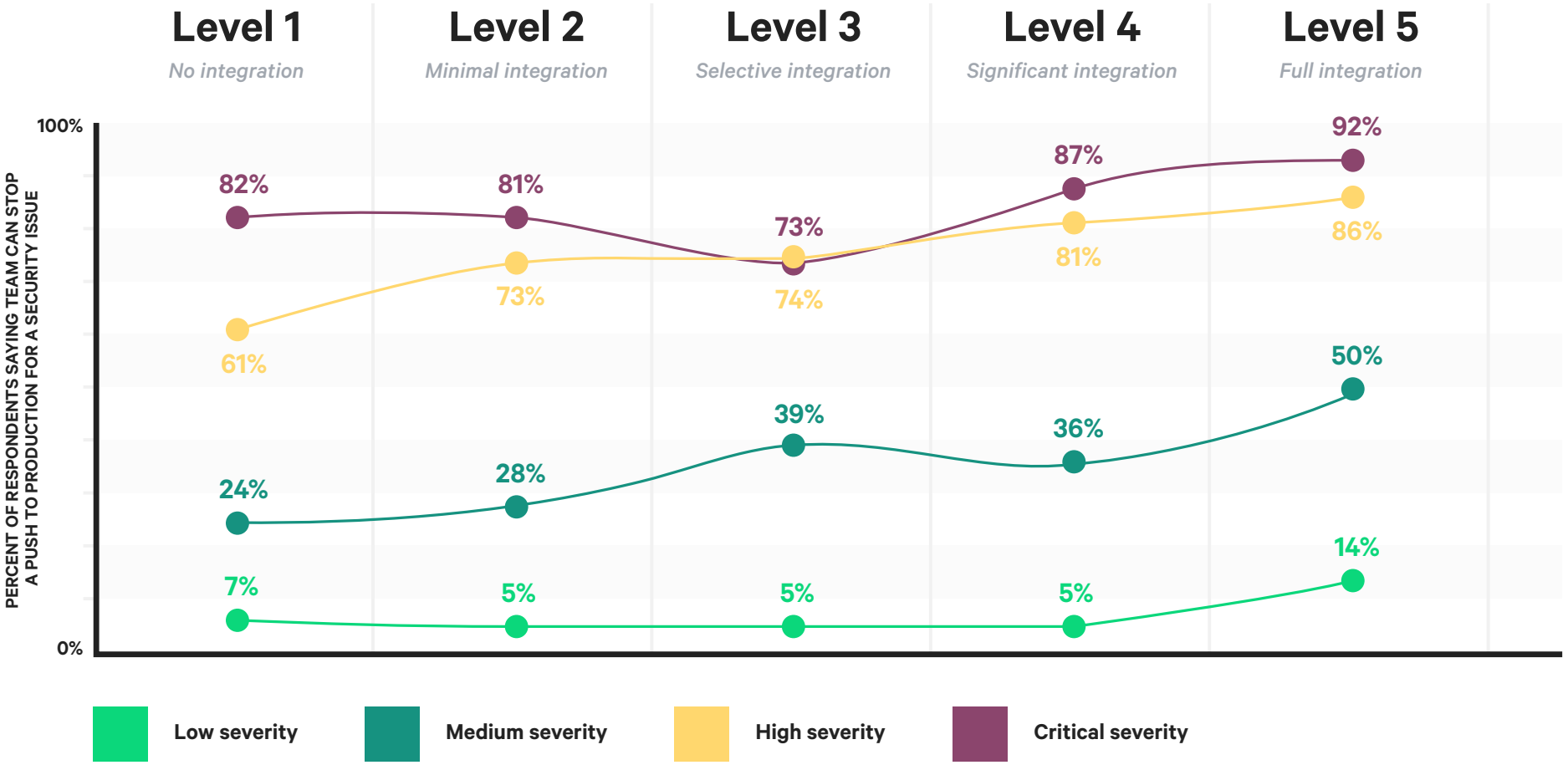
These more advanced organizations have a lot of automation in place, and having an automated process to prevent pushing known-vulnerable code allows for easier risk mitigation. If you do have to stop the train, you already have a good process for restarting it and getting it back on track.

When you can easily stop a push to production, your company is protected from the risks of releasing insecure code — risks like breaching customer privacy, making your own information and secrets vulnerable, and opening your systems to theft. This ability can also make auditors a lot more confident in your systems, speeding the audit process.

Allowing the delivery team to make the call on stopping a push to production is empowering for the team. It shows they're trusted to use their knowledge of both the technology and the business to do what is best for the customer and the company. By moving this decision making away from a centralized security team (or other security gatekeeper) and giving it to the delivery team, you're also making a statement about security being a shared responsibility. All of these benefits drive towards stronger DevOps practices. Last but not least, giving the decision-making power to the delivery team is better for performance. It makes the whole process faster, removing the bureaucratic back-and-forth that slows things down so much.

Security integreation and ability to stop the train

Ability to stop a push to production because of a security issue, by severity level.



Perception & sentiment outcomes

Security as a shared responsibility

In large, siloed organizations, assuring security is often seen as the sole responsibility of the security team. This is because:

- **Security is a highly specialized field where exact protection and defense implementations vary greatly due to runtime environments, developer languages and tools, attack surface area, assets in use, and the user base.**
- **Security requires a different way of thinking from programming. It's often about looking for ways to make code fail, rather than looking for ways to make code work.**
- **Security is a moving target, with attackers gaining new capabilities daily.**
- **Developers are often incentivized to build and release new features fast, and security may not be top of mind for them.**

A more holistic approach to software security is needed, one that integrates security measures throughout the software delivery lifecycle:

- **Identifying security requirements during the requirements phase**
- **Following secure coding standards**
- **Continuously testing for vulnerabilities**
- **Comprehensive logging and monitoring in production**

When responsibility for security is shared across delivery teams, rather than siloed within one team, security issues are caught earlier — there are more eyes looking for potential security problems. And it's much less expensive to remediate vulnerabilities earlier in the cycle than to remediate in production.

According to a study by IBM System Science Institute[1], the cost of fixing defects increases exponentially later in the software development life cycle. It costs 6 times more to fix a bug found during implementation than to fix one identified during design; 15 times more if it's identified in testing; and 100 times more during regular maintenance once the code is in production.

For security bugs, the costs can soar even higher. A security flaw in production can cost a company actual money if attackers can get at cash accounts. If trade secrets or proprietary data are stolen, a company may face heavy losses as they're sold to competitors. If customer data is lost, the company may face lawsuits, and customers who've lost confidence in the company may take their business elsewhere.

---

1 *https://www.researchgate.net/publication/255965523_Integrating_Software_Assurance_into_the_Software_Development_Life_Cycle_SDLC*

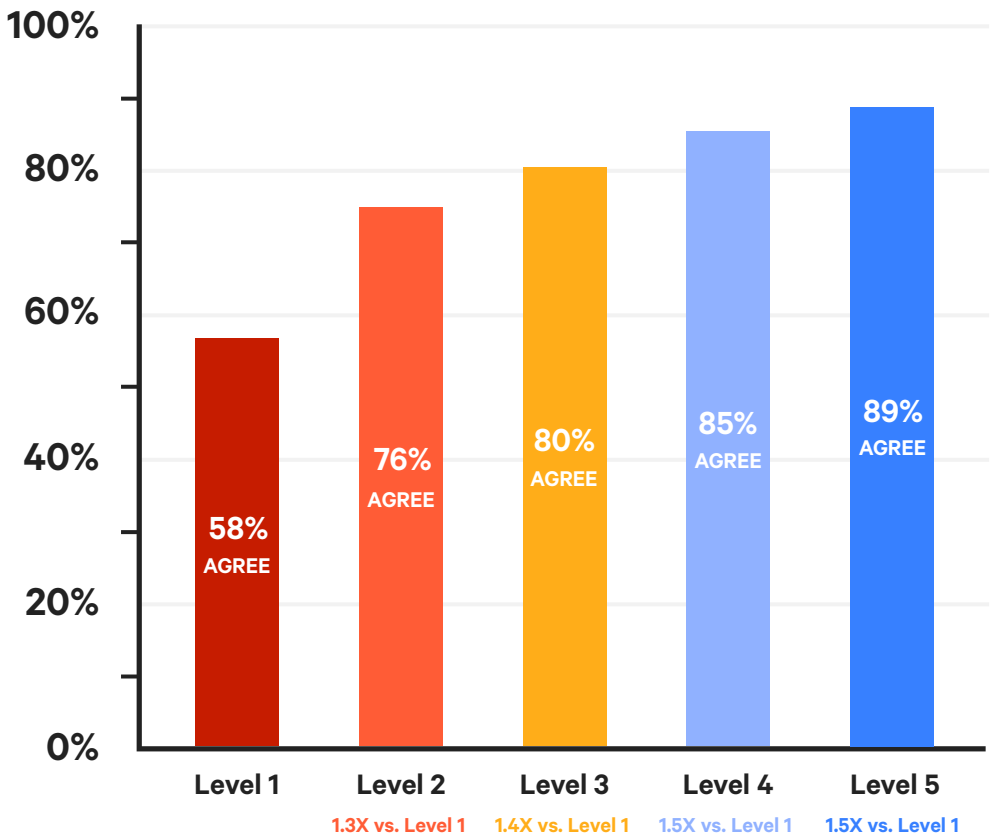
Security and externally developed code

Note that when we're talking about feature development and security issues, it's easy to fall into the mental trap of thinking about code as being developed entirely in-house, and any security issues resulting solely from that code. The reality is that modern software development relies heavily on external components and libraries, many of them open source, with their own release cycles and updates.

According to the Snyk State of Open Source Security 2019 Report, vulnerabilities in indirect dependencies account for 78 percent of overall vulnerabilities in today's software. Improving your security posture involves not just writing secure code, but instituting the processes and practices that keep you aware of vulnerabilities via dependencies, and then remediating these as they arise. The workflows required for identifying, assessing and quickly remediating vulnerabilities found in dependencies require the same technical and cultural foundations that DevOps practices are built on — automation, autonomy, measurement and collaboration.

Security as a shared responsibility

Respondents agree security is a shared responsibility across delivery and security teams.



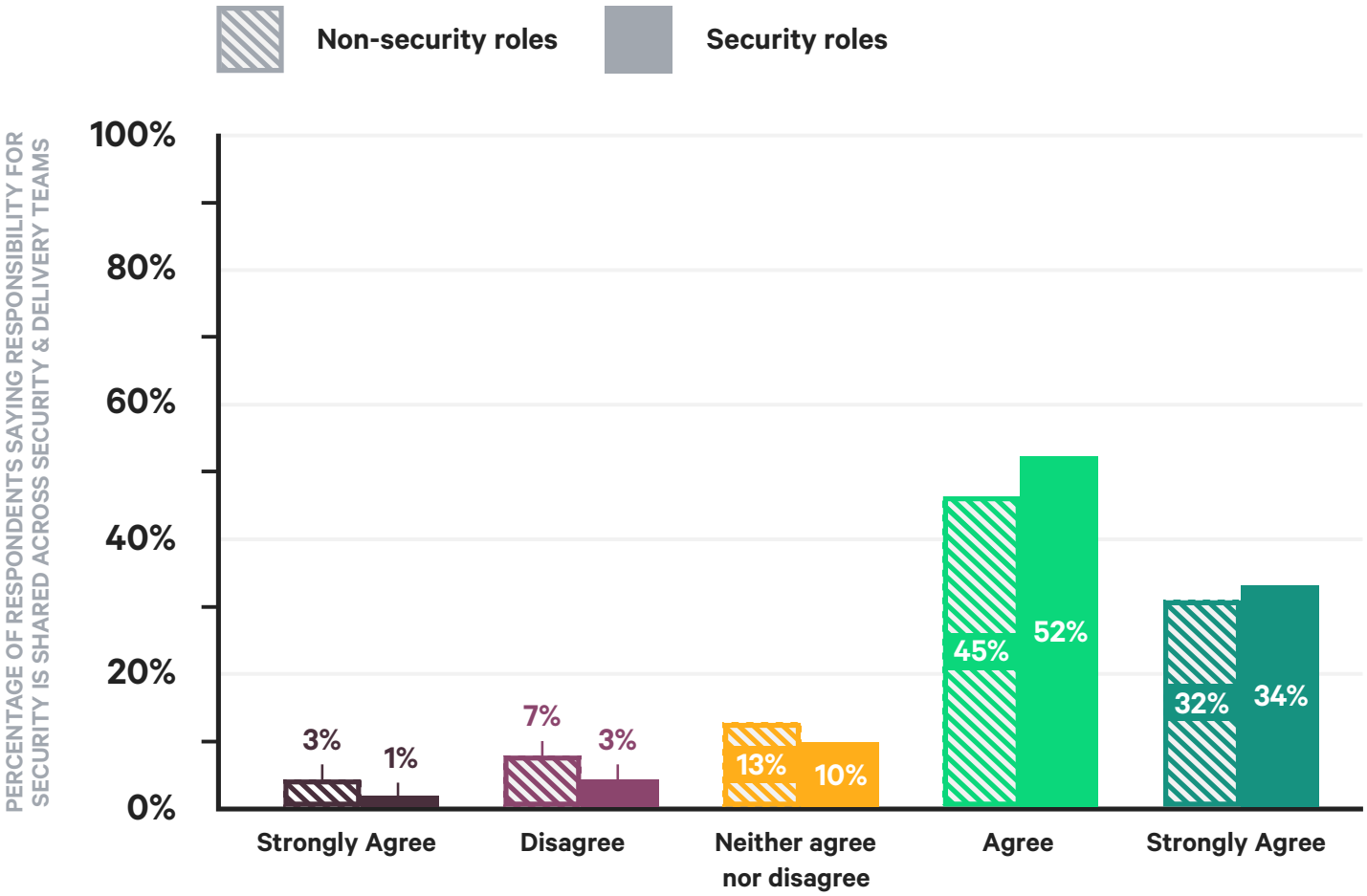
Security as a shared responsibility

We found that the more security is integrated into the software development lifecycle, the more delivery teams see security as a shared responsibility. In fact, seeing security as a shared responsibility improved by 31 percentage points between Level 1 and Level 5.

Seeing security as a shared responsibility makes it far more likely that everyone involved with software delivery will pay close attention to potential security issues, be more diligent about following security policy and processes for each stage of the software cycle, and be more willing to halt deployment when a security issue warrants it. Friction between teams over security lessens, too.

Security as a shared responsibility

Responsibility for security is shared across security and delivery teams.



Security as a shared responsibility: different perspectives

We asked if responsibility for security is shared across security and delivery teams. We were curious to see whether security professionals would give a different answer from respondents who aren't security specialists.

We were surprised by what we found. Fifty-two percent of security professionals agreed that security is a shared responsibility at their firm, compared to 45 percent of those who aren't in security roles.

We're not sure whether this means security professionals take a more generous view of their colleagues, or whether security professionals are more likely to work at companies that are further along with security integration, so people really do share responsibility more. That said, it could simply mean that security professionals interpreted the question to mean security should be a shared responsibility.

Security integration and audit outcomes

We wanted to know if better security integration results in better audit outcomes. We asked how often audits result in any of these issues:

- Issues that require immediate correction.
- Issues that require correction as a high priority.
- Low-priority issues that can be prioritized and scheduled later.

Forty-seven percent of firms at Level 3 always or often find issues that demand immediate correction, compared to just 27 percent of Level 1 firms and 29 percent of Level 5 firms. We observe this same pattern for both high- and low-priority issues: Firms in the middle ranges of security integration report finding issues more often, while Level 1 firms and Level 5 firms report about the same frequency of security issues of all priority levels.

This similarity between the most evolved and least evolved firms is perhaps ironic, but it makes sense when you consider the details. It's likely that Level 1 firms aren't looking all that deeply at their security, and are finding fewer, but more widespread foundational issues such as weak root passwords, shared admin accounts, hardcoded credentials in source code, etc. It's also likely that Level 5 firms have already worked through and resolved fundamental, widespread security issues, so are now able to delve deeper and look more closely at process and practices, finding issues related to specific services (for example, the payroll service is accessible from the contractor network whenever a batch processing job runs).

Companies at the middle levels — those just starting to integrate security — are likely to have an abundance of security issues they're finding and working on, both broader issues and more service-specific ones. They're trying to get their hands around a lot more as they include security considerations in more stages of their software development process.

Audit frequency also contributes to these differences. Organizations with little to no security integration are audited less frequently, and the expectations and requirements of those audits are lower. In Level 5 firms, the staff expects audits, prepares for them, and generally can deal with a deep inspection, which results in a relatively painless set of findings.

In the middle levels of security integration, the number of yearly audits goes up. Departments and teams are far enough along with security integration to identify gaps, but haven't reached a level of maturity to handle the findings with ease — thus teams rush to create new process in response to findings, likely needing to rework previous processes that may not have been intentionally designed.

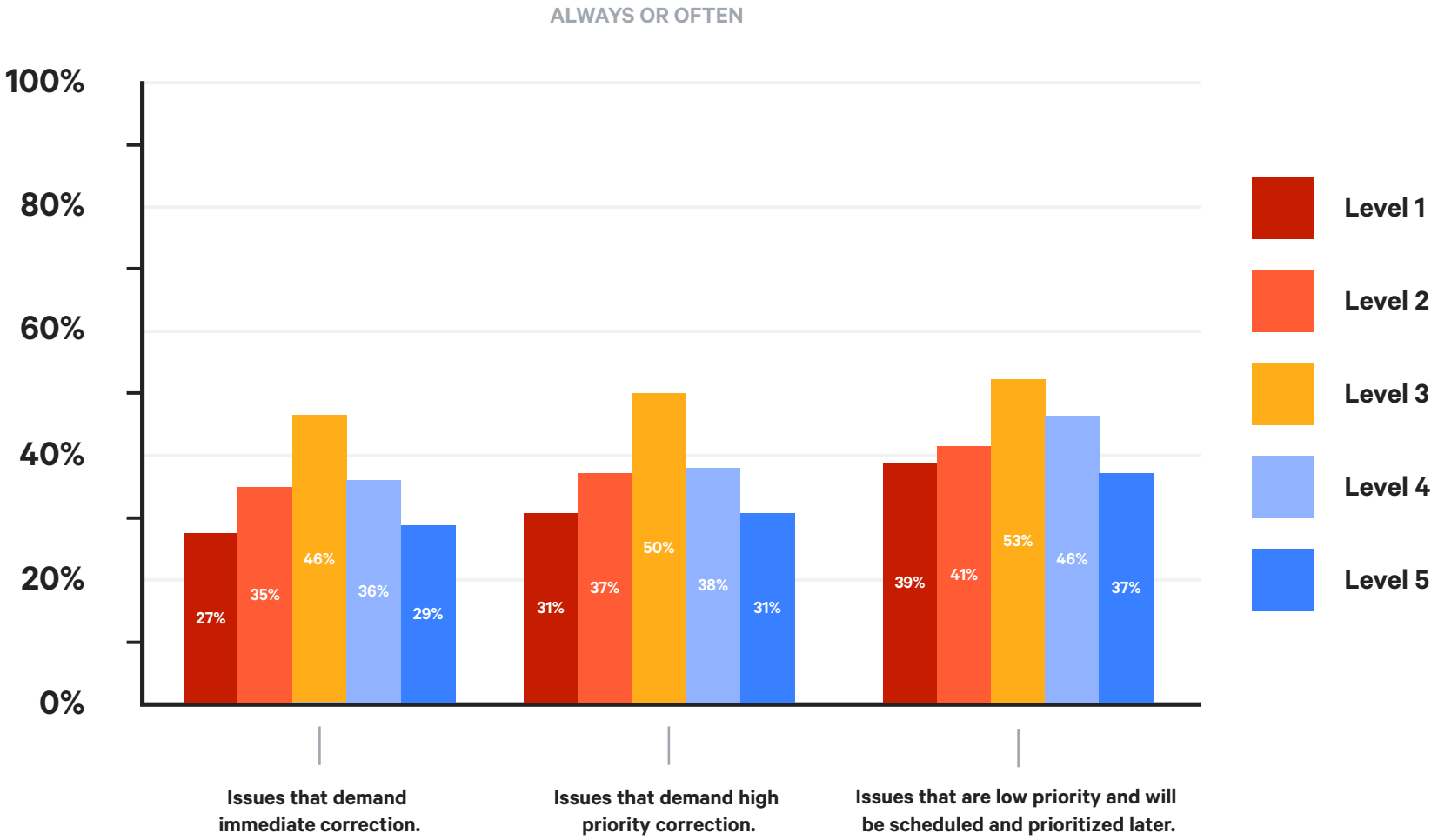
Regardless of the level of security integration, audits are still disruptive to the business. Some work has to be stopped to get an audit done. Just as with processes that mature and improve, though — for example, deployments — audits get easier the more often they're done, and the more they're automated. For example, doing a SOC 2 Type 2 report can shorten audits because several other auditing standards can use it as a baseline.

Audits: a definition for this report

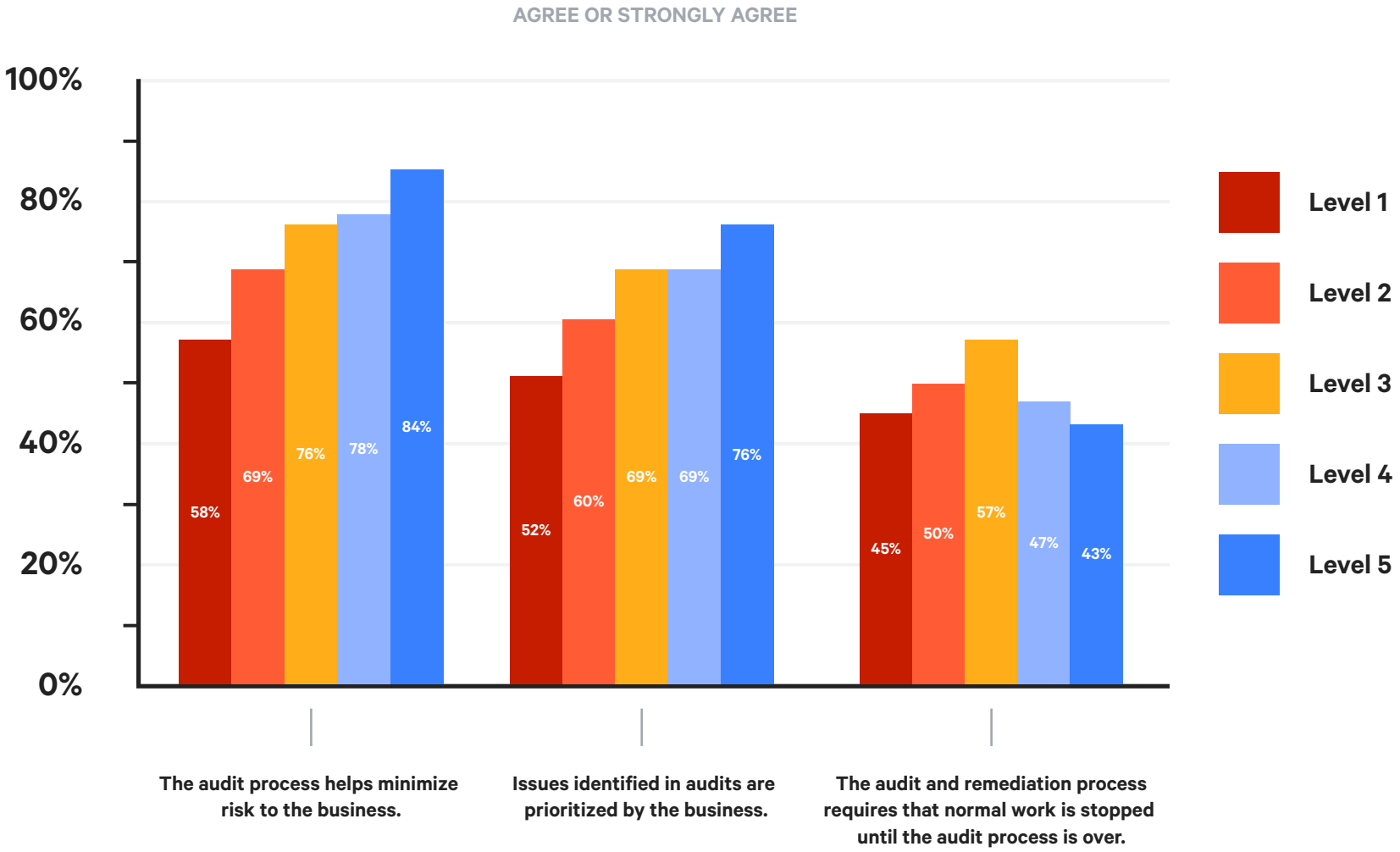
Organizations go through all kinds of audits. For the purposes of this research, we understand "audit" to mean a formal audit that's performed by in-house auditors, consultants, customers, or an audit firm.

Audits are often performed to test and verify compliance with auditing requirements set by regulatory and certification bodies. A few common examples: FedRAMP, Sarbanes-Oxley, PCI, SOC2, ISO9001, GDPR, HIPAA and other standards in healthcare, insurance and finance.

Respondents saying audits uncover issues always or often



Sentiment around audits by level of security integration

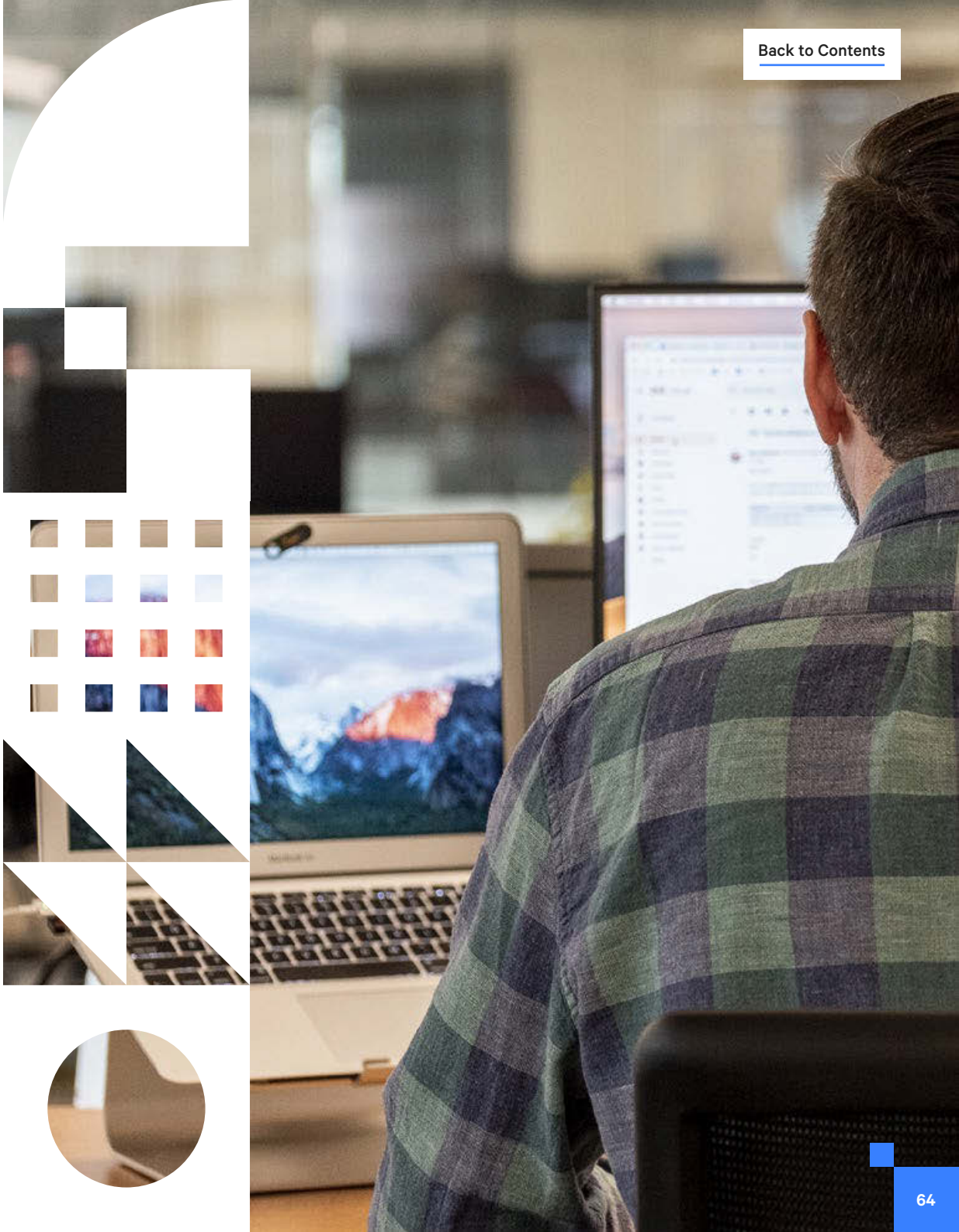


7

Security integration is messy

While security integration does lead to great outcomes, the path to them isn't necessarily easy. We noticed a pattern in Levels 2 and 3 where things get worse before they get better. After a promising start, hopes for DevOps transformation are buoyed. But instead of seeing incremental improvement as you become more integrated, things start to get messy. But as we saw from the outcomes in the previous section, if you stick through it, there is a light at the end of the tunnel.

This is known as the J curve, a pattern we've observed in previous reports. In the 2016 State of DevOp Report, we saw that medium performers spent more time on rework than low performers. Again, in the 2017 report, we saw that medium performers were doing more manual work than low performers when it comes to deployment and change approval processes. That's because medium performers are starting to uncover all the buried complexity that led to glue and duct tape, technical debt and manual controls in the first place. It's important to keep in mind that this is a natural and temporary transition phase during the evolutionary process.



Security integration does not eliminate friction

We wondered if friction between security and delivery teams decreases as they become more integrated. We were surprised to find that significant friction exists regardless of the level a firm has reached. We thought it would be better as a firm integrated security more deeply, but instead, we see the characteristic "messy in the middle" pattern that we mention throughout this report.

At Level 1, friction is somewhat lower than at higher levels of security integration, because security and delivery teams rarely interact. Firms at Levels 2 and 3 experience more friction because they're starting to work more closely together to build security into the delivery pipeline. This is new territory, and people have to change their perspectives and expectations, which is hard. Friction starts to normalize again at Levels 4 and 5, as the teams get more comfortable working together and have processes and patterns they can rely on.

There's a significant difference in feelings about friction between respondents who work in security jobs and those who work in non-security jobs. For people in security jobs, there is a massive jump in respondents experiencing friction from Level 1, at 38 percent, to Level 2, at 65 percent. At Level 3, a full 70 percent of security pros — 32 percentage points more than at Level 1 — are feeling "a lot of friction" (the highest level possible). Level 4, at 51 percent, is not as bad. But overall, the friction security pros feel is much worse at all levels of integration above Level 1 until they get all the way to Level 5, where integration is complete. Just 25 percent report a lot of friction at this point, finally dipping below the Level 1 respondents.

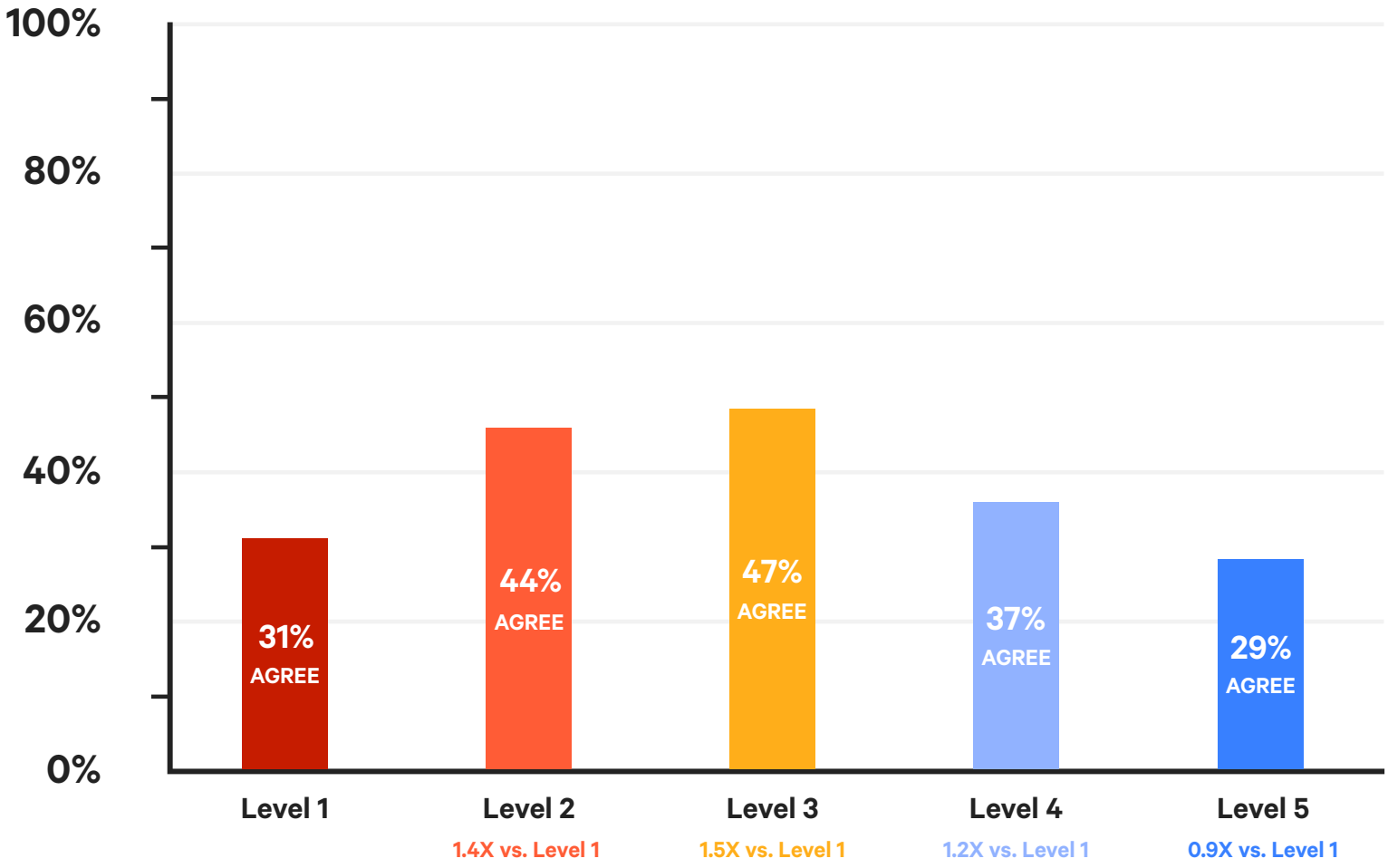
To compare, the greatest number of non-security respondents reporting they feel a lot of friction (the highest level possible) is at Levels 2 and 3, where 43 percent feel this way. That's 12 percentage points greater than the number at Level 1, where there's no integration at all. Respondents at Level 4 are more or less back to normal — just 4 percent more respondents report feeling a lot of friction than at Level 1.

DevOps is about cultural change, and it's critical to the success of DevOps to recognize how difficult cultural change is for everyone involved. So it's important to note, as you go about trying to integrate security into the software cycle, that your security team will perceive this process change as introducing a lot more friction to their work lives. They'll feel this increased friction sooner than their colleagues in other roles, and they'll feel the impact for longer.

There's an offset for our security friends, though: Over the long run, security integration really does alleviate friction for a significant percentage of security practitioners. The percentage of security respondents dealing with a lot of friction drops from Level 1 to Level 5 by 14 percentage points. By contrast, the number of non-security respondents reporting a lot of friction drops by just 2 points. So while security pros will find integrating with software delivery harder than their colleagues do, they will ultimately reap some of the greatest benefits of this cultural change.

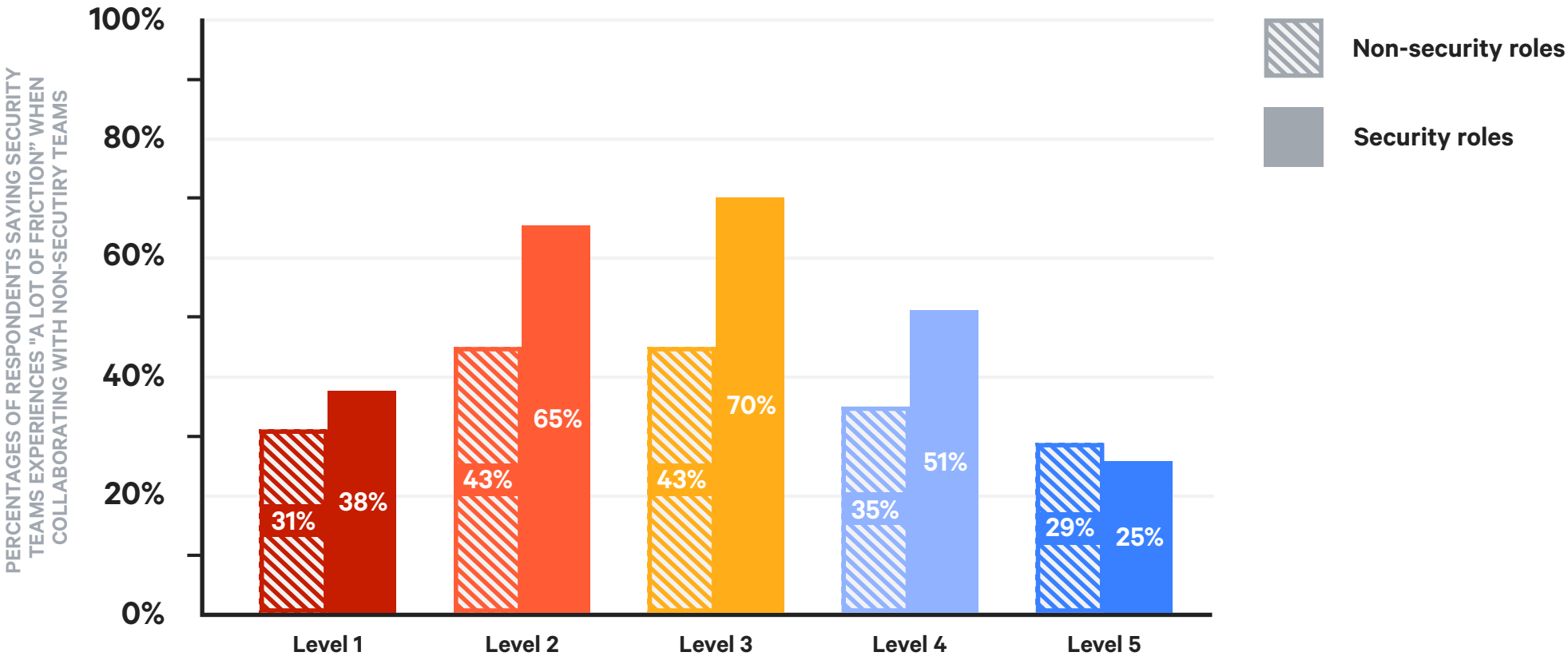
Security integration and friction between teams

Respondents feel security team encounters a lot of friction when collaborating with delivery teams.



Friction between teams, security vs. non-security roles

Security professionals experience more friction than those in non-security roles.



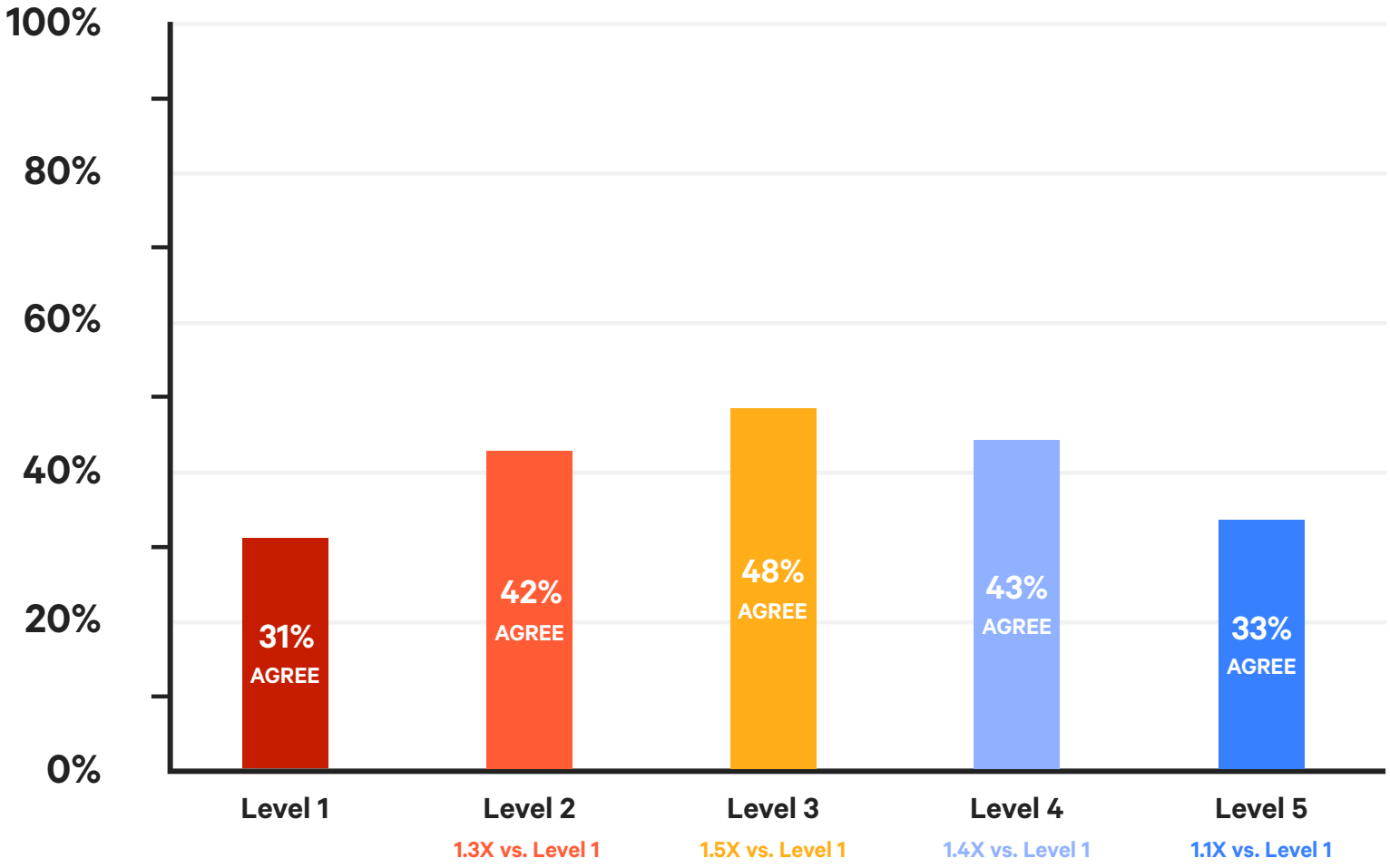
Security integration slows things down for a while

We wanted to know if being at a more advanced level of security integration changes the perception that security is a major constraint to delivering software quickly. Again, at Level 3, agreement is the highest at 48 percent, but by Level 5, it goes back down. Friction at Level 5 is a bit higher than at Level 1, but the difference is not significant. This tells us that full integration does not actually cause more friction than no integration. Full integration does not necessarily make delivery quicker, but it doesn't slow it down significantly, either.

As usual when you're changing how you work, things get worse before they improve. Early stages of integration are troublesome as security practices are introduced into stages where they weren't before. Delivery speed takes a hit, too, and that's frustrating. These troubles do eventually dissipate as teams collaborate more smoothly to embed security in the delivery cycle, refine their processes and see the benefits of their work.

Security integration and speed of software delivery

Respondents feel security is a major constraint on ability to deliver software quickly.



When you're first trying to integrate security into your software delivery practices, your early wins are likely to be fast and simple, but the pace will soon slow. People have to learn what to do, do it all manually, learn some more, optimize, normalize, and ideally, ultimately, automate.

One common example is threat modelling. The first time a team does this, it feels strange. A facilitator asks you what assets a system or subsystem has, who would want them and why. You have to flip your thinking, taking the point of view of an adversary. Next you have to brainstorm, write down your findings, work them into design, code, and then review all the threat vectors identified during the modelling exercise. It's a time-consuming process. Once you get the idea, though, you can schedule these threat-modeling exercises at regular intervals, provide templates for people to work with ahead of time, look for common patterns when certain assets are desired, and build a body of knowledge to make the whole thing move faster. And these exercises will enable additional, higher-level security work.

This process of identification, creation, growth and refinement has to be repeated many times, and it involves myriad parties, depending on the nature of each process — change control boards, monitoring experts, policy departments, compliance reviews, auditors, user research, production control and more. Each process requires a feedback cycle, and there are several processes. All this shows why the middle stages of security integration can feel like a long uphill climb.

Security isn't compliance. Compliance isn't security.

It's always best to keep in mind that compliance with policy is not directly correlated with being secure. Policies can be outdated, address business requirements rather than actual threats, or simply be incomplete. Being compliant may still leave weak links in your IT estate. For example, if your policy says that a low-priority vulnerability must be mitigated or patched within 60 days, an attacker could use that vulnerability for gains any time during that 60-day window, even though you're fully complying with policy.

Companies undergoing audits are usually seeking to simply demonstrate compliance, and to provide proof of adherence to controls (whether internal or external). However, compliant status can be granted even if a company doesn't comply directly with the stated policy, but instead offers a "compensating control," or "exception." These alternatives may result in vulnerabilities.

"Compensating controls" generally mean the intent of the policy is met through secondary measures, and this can be demonstrated. Let's take the example of a separation-of-duties policy. A company may choose not to implement the policy for emergency fixes of production equipment, relying on a compensating control: All activity logs are sent to a system where the original actor doesn't have modification rights. While this procedure bypasses some change control processes in an emergency, it does provide evidence that it would be difficult for a bad actor to alter data and go undetected.

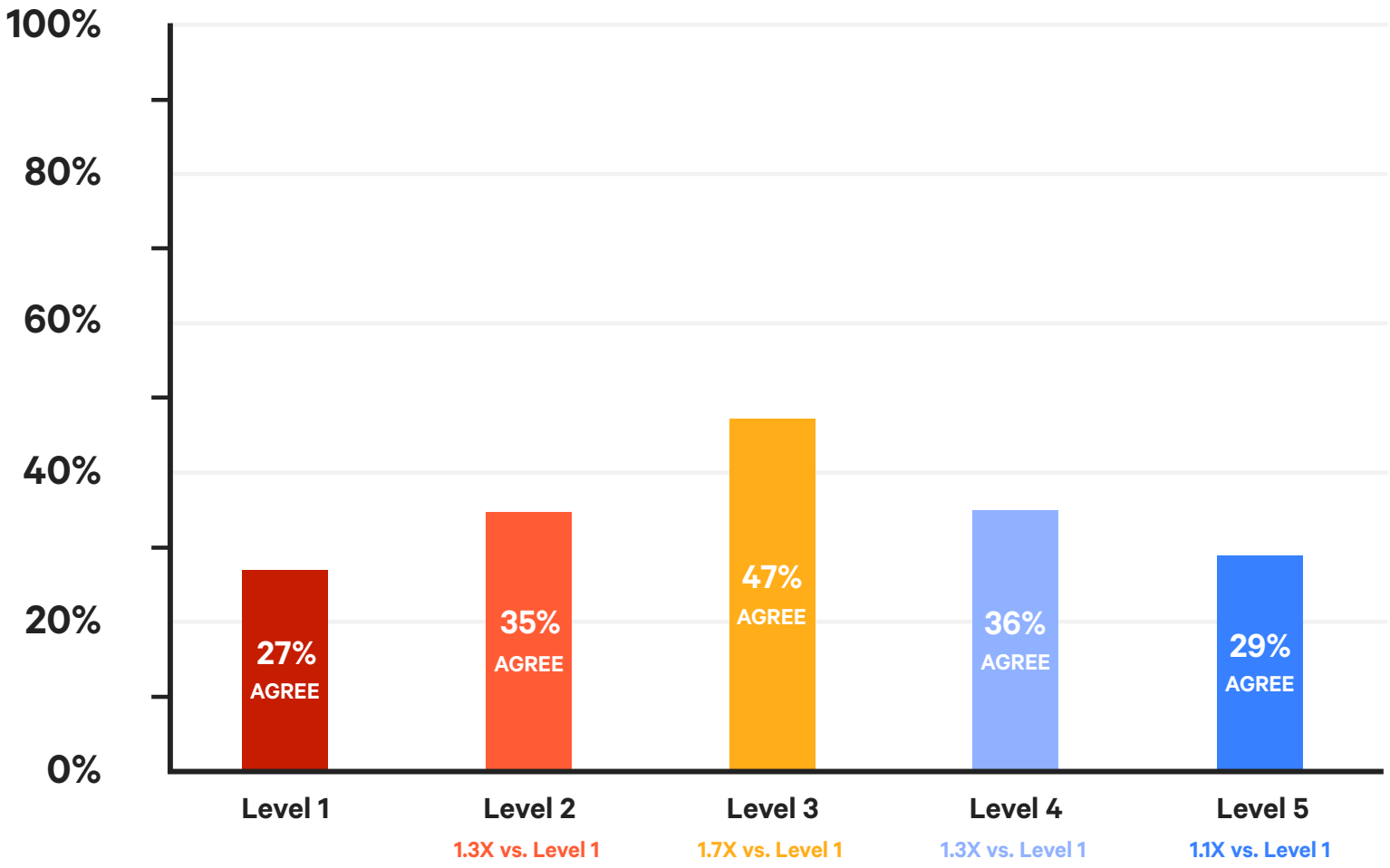
An "exception" is process and paperwork to address and document a known-deficient setup. These are often granted by a security or audit team because complying with the normal policy would cause lost business or unacceptable new costs. In some organizations, these are granted on a time-based pattern and must be reviewed and renewed periodically. Other organizations grant them once and rule an application or practice now "out of scope" for that policy. But attackers don't care about the signed-off email saying it was okay to relax security settings — they just look for a vulnerability to exploit.

Security integration doesn't mean you'll find fewer issues

We wanted to know whether deeper security integration would result in audits uncovering fewer issues that require immediate correction. Audits are typically disruptive events that compel teams to stop their normal work and address any serious issues found. With deeper integration, auditors are able to move beyond basic practices and into the core data protection practices and security processes beyond a basic break/fix mentality. This often results in more audit findings and lengthens the list of things to fix or improve. These findings, however, will likely be focused on a specific application or service, and so will probably seem more helpful, rather than frustratingly broad or general.

Security integration and audit issues

Security issues revealed by audits always or often require immediate correction.



8

Organizational structure

One of the most frequent questions we get is "What's the ideal organizational structure for DevOps success?" Our unsatisfying answer is "It depends." It depends on a lot of factors, including:

- The flexibility of your current org structure.
- The organizational culture.
- How siloed different functions are.
- The skillsets you currently have on your team.
- The relationship between teams and team leaders.

We wanted to find out how organizational structure affects levels of security integration. Here's what we discovered:

- Forty-eight percent of respondents had a central security team that supports delivery teams on demand. Fifty-seven percent of enterprises with revenue greater than $1 billion have a central security function supporting delivery teams on demand.
- Thirty-one percent have a centralized security function, and delivery teams also include a designated security expert. Forty-six percent of enterprises with revenue of more than $100 million and less than $1 billion have a centralized security function, and delivery teams that include a designated security expert.
- Only 14 percent of respondents had a decentralized security function where each delivery team has its own dedicated security expert. The decentralized structure is most common in smaller organizations with revenue less than $1 million.

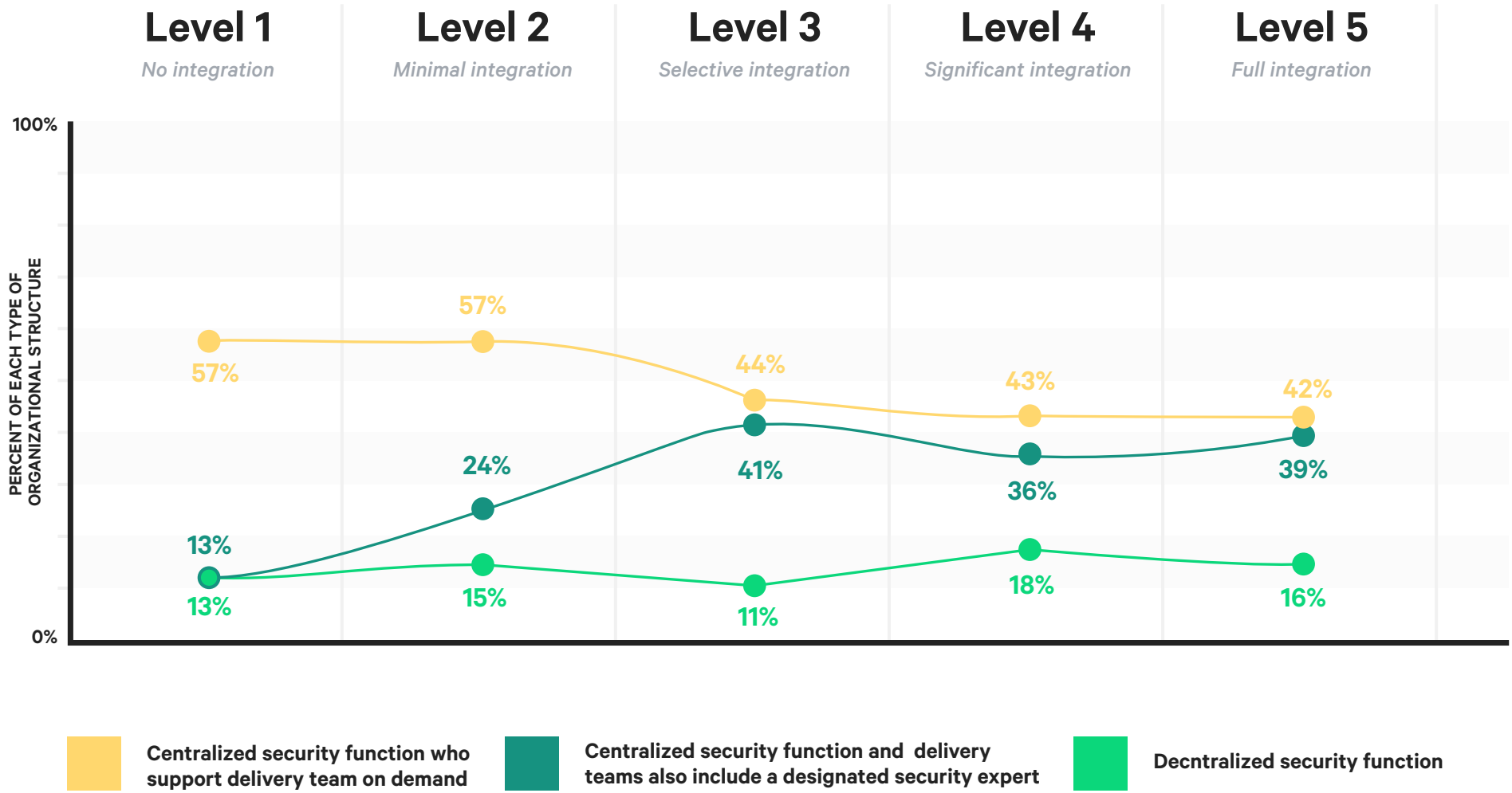
Organizational structure and the security function

	ALL RESPONDENTS	RESPONDENTS' TEAM SIZE			ANNUAL REVENUE OF RESPONDENT EMPLOYER		
		10 or less	11 to 30	31+	< $100M	< $100M to <$1B	$1B +
Centralized security function that supports delivery teams on demand	48%	51%	44%	49%	41%	42%	57%
Centralized security function and each delivery team also has a designated security expert	31%	20%	41%	40%	22%	46%	33%
Decentralized security function; each delivery team has a security expert	14%	19%	12%	9%	26%	8%	7%
Other org structure	6%	10%	3%	2%	10%	3%	3%

While a centralized security function that provides on-demand support is the most common organizational structure, you'll notice that at Level 3, it becomes less prevalent (13 percentage point decrease from Level 1 to Level 3). On the other hand, the percentage of firms with a centralized security function with designated security experts residing in delivery teams increases dramatically at Level 3 (28 percentage point increase from Level 1 to Level 3).

We suspect that in Levels 1 and 2, there is insufficient investment in security, so organizations rely on ad hoc support from the security team. But by the time an organization reaches Level 3, and security awareness and security culture become more prevalent, organizations are investing more in security. This makes sense. Having a security practitioner who's dedicated to application development, delivery, and operations, regardless of which team they are on (the DevOps team, or a central security team), can ensure security is treated as a design constraint.

Security integration and organizational structure



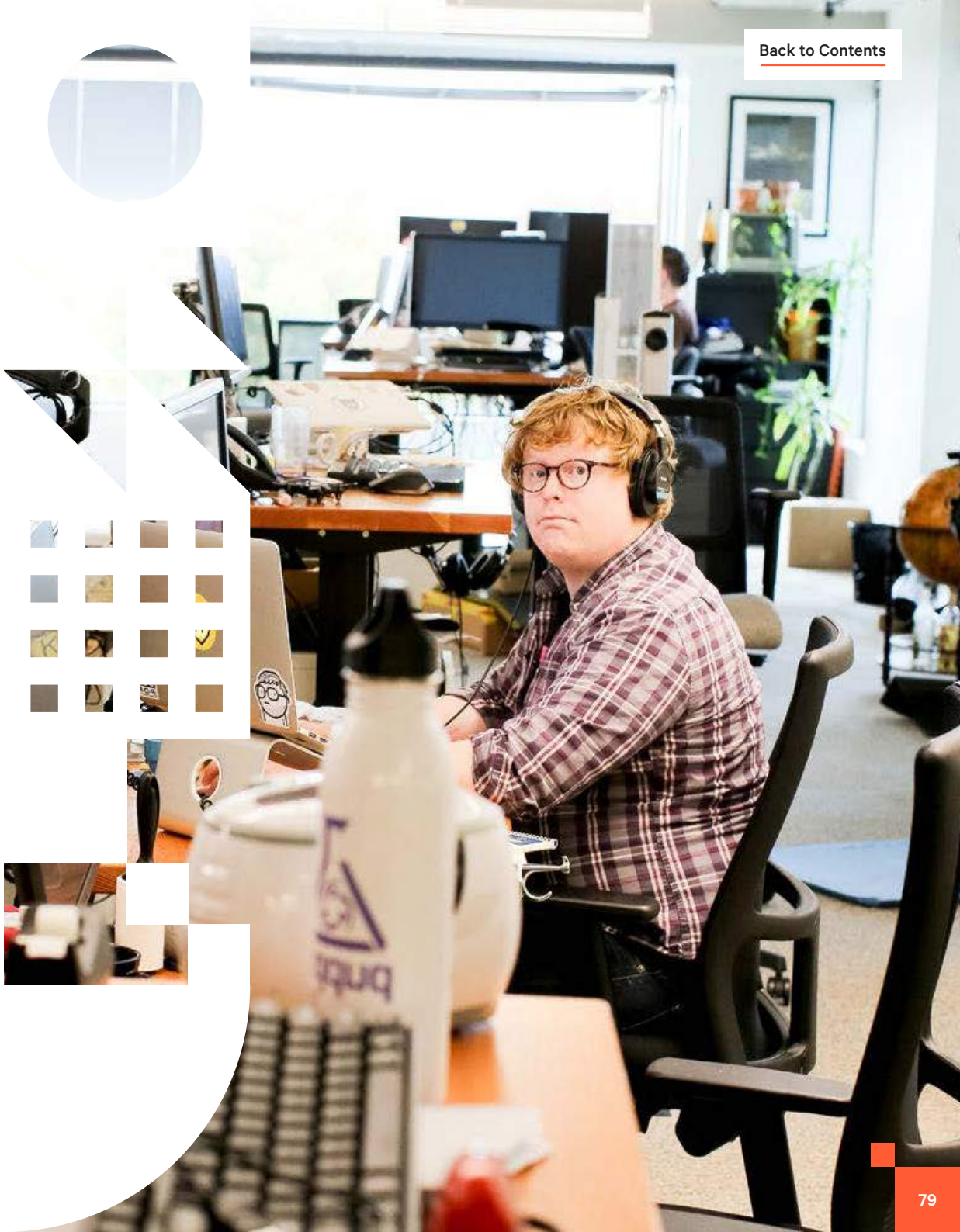
Back to Contents

Conclusion

Organizations that are serious about improving their security practices and posture should start by adopting DevOps practices. It's not coincidental that organizations at the highest levels of DevOps evolution also have fully integrated security practices. The sharing and cross-team collaboration that DevOps practices promote lay the groundwork for expanding those practices — and more importantly, the mindset — to more teams and functions across the enterprise. And that includes security.

It's true that everyone should care about the security of the application or service they're building, but people will continue prioritizing the work that's right in front of them unless they are incentivized to do things differently. That's why security needs to be prioritized from the top of the organization. That's also why it needs to be built into the entire software delivery lifecycle, so it's not regarded as an extra thing that needs to be done, but is as integral to software delivery as testing.

We hope this year's report helps you build the case for better security integration in your organization. We'd love to hear from you, and we're happy to answer any questions you may have. You can reach us at: ***devopssurvey@puppet.com.***



Methodology

The security model

The primary goal of this year's State of DevOps Report was to explore how firms integrate security into their DevOps processes, and further, how that integration affects security outcomes, software outcomes and the relationship between development, operations and security teams. Our hypothesis was that increased security integration would lead to better security outcomes while also expediting deployment rates and other general development goals. We also expected that certain organizational structures would be able to integrate security more successfully and with less friction

The model evaluates security integration through the number of touch-points security has in the delivery cycle. Respondents were asked if their security teams or security experts were typically involved in any phases of their delivery cycle, which we defined as **requirements, design, building, testing** and **deployment.** A score of "1" was given for each phase where security was integrated, and the values were then summed across the five phases to give an overall integration score. We found that where security was included in three phases or four phases of the software cycle, outcomes were similar enough to combine them into a single level. Thus we were left with a scale of 1 to 5 for level of security integration, where at Level 1, there's little to no integration, and at Level 5, security is fully integrated in all five phases of the software cycle.

Back to Contents

Drivers of security posture

First linear regression and next Shapley regression were used to evaluate the drivers of security posture. Multiple linear regression was used to assess the strength of the model, after which Shapley regression was used to understand the relative importance of the drivers. Drivers were reported out using Shapley regression values.

The DevOps model

Learnings from the 2018 State of DevOps Report were employed to recreate last year's DevOps model. The model encompasses five stages of DevOps evolution, which are used to categorize respondents into three evolution categories, Low, Medium and High. This year's recreation of the model matches the findings from last year, and provides considerable support for the validity and consistency of the model. The full methodology from last year's State of DevOps Report is available at *https://puppet.com/2018-state-of-devops-report-full-methodology.*

Target population and sampling method

This survey collected data from technical professionals with a working knowledge of their IT operations and software delivery process. The survey was conducted online from 21 May to 14 June 2019 and respondents were gathered through two avenues: a snowball sampling method and a professional panel.

Snowball sampling

Snowball sampling is a process where respondents are encouraged to share a survey with their networks, causing the sample to grow like a snowball rolling downhill. Promotion was done via email lists, social media and various partners, and the sample was collected globally, from Europe, the Middle East, Africa, the Asia-Pacific region and the Americas. Given the channels of promotion and the nature of snowball sampling, this portion of the sample is likely limited to firms and teams that were already familiar with DevOps, and as such, may be doing some of it.

Panel sample

The snowball sample was supplemented by a third-party panel made up of respondents acquired from third-party panel providers. Their presence reduces bias in the overall sample. Our third-party panel provider nurtures and maintains a quality, engaged membership panel built to support its market research clients and to benefit non-profit organizations. This panel provider's unique approach to recruiting yields a highly engaged group of people who are, in their role as panel members, dedicated to helping market research clients fulfill their information needs. The panel provider's unique non-profit recruitment method enables the firm to source C-suite executives, directors, and managers who have key decision-making authority. In addition to their non-profit relationships, this sample provider also utilizes trade association partners to help drive certain audiences into online surveys. This approach provides access to the appropriate sample for each survey. The advantages offered by this panel are core to our differentiation.

Author biographies

Andi Mann

Andi is chief technology advocate at Splunk and an accomplished digital business executive with extensive global expertise as a strategist, technologist, innovator, and communicator. For over 30 years across five continents, Andi has built success with Fortune 500 corporations, vendors, governments, and as a leading research analyst and consultant. Andi is also a sought-after commentator on business technology. He has been published in USA Today, The New York Times, Forbes, CIO, and The Wall Street Journal; presented at Gartner ITxpo, VMworld, CA World, Interop, Cloud Expo, and DevOps Summit; and participated and hosted interviews for radio, television, webcasts, podcasts, and live events.

Twitter: @andimann

Alanna Brown

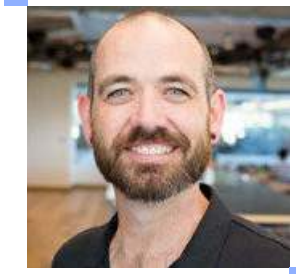
Alanna is senior director of community and developer relations at Puppet, where she's had the privilege of helping Puppet grow from a small startup to a global brand with thousands of customers around the world. She conceived and launched the first annual State of DevOps Survey in 2012, and has been responsible for the survey and report since then. In addition to heading up DevOps research, Alanna is also responsible for driving awareness, adoption and advocacy for Puppet's product portfolio.

Twitter: @alannapb

Michael Stahnke

Mike is vice president of platform engineering at CircleCI. Prior to this role, he was at Puppet, running engineering for Puppet Enterprise, open source Puppet, and SRE. Prior to Puppet he was an infrastructure architect, team lead and open source evangelist at Caterpillar Inc., where he spent inordinate amounts of time with auditors. He was also an author for the 2018 State of DevOps Report. Michael helped get the Extra Packages for Enterprise Linux (EPEL) repository off the ground in 2005, is the author of Pro OpenSSH (Apress, 2005), is an organizer of Devopsdays Madison, and rants continuously about technology, humans, and computers, while striving to learn more about them.

Twitter: @stahnma



Nigel Kersten

Nigel is field CTO at Puppet, responsible for bringing product knowledge and a senior technical operations perspective to Puppet field teams and customers, working on services strategy, and representing the customer in the product organization. He also works with many of Puppet's largest customers on the cultural and organizational changes necessary for large scale DevOps implementations. Nigel has served in a range of executive roles at Puppet across Product and Engineering over the last nine years, and came to Puppet from the Google SRE organization, where he was responsible for one of the largest Puppet deployments in the world.

Twitter: @nigelkersten

Report presented by

Puppet is driving the movement to a world of unconstrained software change. Its industry-standard platform automates the delivery and operation of the software that powers everything around us. More than 40,000 companies — including more than 75 percent of the Fortune 100 — use Puppet's open source and commercial solutions to gain situational awareness and drive software change with confidence. Based in Portland, Oregon, Puppet employs more than 500 people around the world.
Learn more at *puppet.com.*

CircleCI is the world's largest shared continuous integration and continuous delivery (CI/CD) platform, and the central hub where code moves from idea to delivery. As one of the most-used DevOps tools, CircleCI has unique access to data on how engineering teams work, and how their code runs. Companies like Samsung, Ford Motor Company, Spotify, Coinbase, PagerDuty, Stitch Fix, and BuzzFeed use CircleCI to improve engineering team productivity, release better products, and get to market faster.
For more information, visit *circleci.com.*

Splunk Inc. (NASDAQ: SPLK) turns data into business outcomes. Organizations use market-leading Splunk solutions to investigate, monitor, analyze and act on all forms of data — from the business, IT, security and the Internet of Things. Our powerful platform and unique approach to data have empowered companies to improve service levels, reduce operations costs, mitigate risk, enhance DevOps collaboration and create new product and service offerings. Join millions of users by trying Splunk software for free: *www.splunk.com/free-trials.*

Report sponsors

Anitian delivers security and compliance at ludicrous speed. Our Compliance Automation platform leverages the power and scale of the cloud, to automatically build, configure, and monitor autonomous cloud environments that accelerate compliance with frameworks such as PCI, FedRAMP, ISO27001, CJIS, and more. Anitian directly integrates with DevOps CICD pipelines to ensure continuous security and compliance alignment. Anitian's Compliance Automation platform is backed with 24/7 threat hunting, compliance guardrails, managed detection and response, and expert security services. Find out more at *www.anitian.com.*

F5 (NASDAQ: FFIV) gives the world's largest businesses, service providers, governments, and consumer brands the freedom to securely deliver every app, anywhere—with confidence. F5 delivers cloud and security application services that enable organizations to embrace the infrastructure they choose without sacrificing speed and control. For more information, go to *f5.com.*

ServiceNow (NYSE: NOW) is the fastest-growing enterprise cloud software company in the world above $1 billion. Founded in 2004 with the goal of making work easier for people, ServiceNow is making the world of work, work better for people. Our cloud-based platform and solutions deliver digital workflows that create great experiences and unlock productivity to almost 75% of the Fortune 500. Now we're making the world of DevOps work better. For more information, visit *www.servicenow.com/products/devops.html.*