Mapping the DevSecOps Landscape

2020 Survey Results

- Over 3,650 respondents from 21 countires
- DevOps successess, challenges, and struggles
- Developers, operations, security, and testers

Table of contents

Introduction

We kicked off our 2020 Global DevSecOps Survey with one thought in mind:

> **Is the role of the developer changing?**

If every company is a software company, it's a critical question to explore.

Nearly **3,700** people shared their DevOps experience with us, many in great detail. We heard about faster software releases, sweeping shifts left and continuous deployments that were, literally, continuous. Ops pros explained how they now spend more time in the cloud, and security teams complained devs don't take sufficient responsibility for bugs.

And yes, the developer's role is changing. But operations, security, and test are changing too, every bit as dramatically.

No matter where you are on your journey — brand new to DevOps or with five years under your belt — and no matter the size of your company, it's vital to understand and define the way these roles are evolving and determine the best direction for your team.

This report offers an inside look at how DevOps practitioners around the world tackle the same challenges you face. With a mix of hard data and real stories, we hope you'll find wisdom and encouragement. No one ever said DevOps was easy — not even GitLab. But it is clear that a mix of the right tools, culture, collaboration, and a willingness to try and fail will move software development teams forward.

That said, keep the following in mind: About half of our respondents came from companies with 1,000 or fewer employees, meaning they likely aren't working with many (or any) legacy systems. About **35%** have been doing DevOps for between one and three years, and about **60%** of them are our customers. Smaller companies starting to get their DevOps rhythms are likely to report aspirational results that aren't reflective of a majority of organizations. Learn from them, but also remember their reality may not be your own, and that's ok. Also, with a majority of survey takers as customers, don't be surprised that they like or use our product.

Overview

2020 DevSecOps Survey top findings

CD is real

Nearly **60%** deploy multiple times a day, once a day, or once every few days. That's up from **45%** last year.

Coming of age

Over **25%** of companies are in the DevOps "sweet spot" of three to five years of practice. And another **37%** are well on their way, with between one and three years under their belts.

Want better code quality?

Want better code quality? It's DevOps FTW. The majority of respondents said code quality was the biggest benefit of choosing DevOps.

Still works in progress

The majority of respondents aren't using K8s and microservices yet, but they are investigating them.

Test is still hard

Today **47%** of companies say testing is the number one reason for delays. That's down slightly from last year's survey but still frustrating to hear.

Job satisfaction

Job satisfaction: **66%** or more of respondents say their organization's processes and tools allow them to succeed and innovate.

The starting point

Here's a closer look at the nearly **3,700** people who completed the survey between mid-January and the end of February 2020.

Role

41.21% *Software Developer / Software Engineer*

10.08% *DevOps Engineer*

9.27% *Development/Engineering Leadership*

7.56% *Software Architect*

5.09% *Other*

4.93% *DevOps Leadership*

3.89% *Technology Executive - CIO / CTO/VP*

2.89% *Project Manager*

2.33% *Site Reliability Engineer*

2.04% *Systems Administrator*

2.01% *Systems Engineer / Network Engineer*

1.65% *Product Manager*

1.56% *Quality Assurance*

1.04% *Operations Leadership*

Gender

88.26% *Male*

7.51% *Female*

3.02% *Prefer not to say*

0.63% *Non-binary/third gender*

0.58% *Prefer to self describe*

Industry

39.98% *Computer Hardware / Services / Software / SaaS*

8.5% *Banking / Financial Services*

7.84% *Other*

7.32% *Education*

7.16% *Business Services / Consulting*

4.17% *Telecommunications*

4.09% *Media & Entertainment*

3.73% *Healthcare*

3.4% *Government*

Region

19.31% *U.S.A.*

10.74% *India*

9.99% *Germany*

4.5% *UK*

5.09% *Other*

2.75% *Canada*

2.75% *Netherlands*

2.69% *Brazil*

2.25% *Spain*

2.08% *Italy*

1.89% *Australia*

1.83% *Mexico*

1.53% *Indonesia*

1.17% *Poland*

1.17% *Malaysia*

1.11% *Hungary*

1.11% *Sweden*

1.08% *Switzerland*

1.03% *Russia*

1% *Austria*

1% *Turkey*

Number of employees

19.66% *1 - 10*

28.43% *11 - 100*

16.26% *101 - 500*

7.29% *501 - 1,000*

14.78% *1,001 - 10,000*

11.74% *10,000+*

1.84% *Don’t know*

Software development today

A majority of survey respondents (**36%**) said their teams use Agile/Scrum, while **27%** said DevOps or DevSecOps. It's important to put those results in context, however: Survey takers could choose all that apply and, in theory, any combination of Agile, Kanban, Lean, or Scrum could be part of a DevSecOps practice, including Water/Scrum/Fall, which **6%** of respondents said was in use.

> **Most practiced development methodologies:**
>
> **36.37%** *Agile/Scrum*
>
> **27.13%** *DevOps/DevSecOps*
>
> **18.95%** *Kanban*
>
> **7.76%** *Waterfall*
>
> **5.52%** *Water/Scrum/Fall*
>
> **4.27%** *Lean*

A little more than **35%** of companies have had DevOps in place for between one and three years, making them relatively new practitioners and perhaps in environments where other methodologies are in use as well. Interestingly, nearly as many survey takers (**18%**) said they've been doing DevOps for a year or less as those who've been practicing it for over five years (**20%**). About **27%** are in what might be called a DevOps "sweet spot" of three to five years, i.e., they're comfortable and successful with the practice.

The word is out about the importance of CI/CD to DevOps, at least based on our survey. Nearly **38%** said their DevOps implementations include CI/CD, while **29%** said test automation, **16%** said DevSecOps, and nearly **9%** use multicloud.

Survey takers offered a detailed look at some of their other DevOps practices:

> **DataOps and monitoring**
>
> **Tight personal Dev and Ops coupling**
>
> **GitOps**
>
> **Continuous collaboration with maintainers and even stakeholders available for clarifications at all stages of the pipeline.**
>
> **Not much DevOps; it's more of an aspiration.**
>
> **IaC, TDD, user validation of features before and after deployment, deployment before release, dark releases, and a pike of other stuff.**
>
> **Centralized logs, metrics, and alerts. Cost optimization.**

Who benefits the most from a DevOps practice? Hands down it's developers, according to **34%** of our survey takers, followed by Ops at **26%**, QA (**17%**), security (**13%**), and the business side at **9%**.

The number one reason to do DevOps? The majority said code quality, followed by improved time to market, less manual testing, happier developers (!!), and increased communication/collaboration.

Nearly **60%** of survey respondents said their organizations deploy multiple times a day, once a day, or once every few days, which perhaps reflects the large number of responses from smaller (and thus presumably more nimble) companies. Just **11%** said they deploy once a month and only **8%** said every few months.

For the second year in a row, survey takers resoundingly pointed to test (**47%**) as the area most likely to cause delays. (To be fair to QA departments everywhere, though, last year the percentage was **49%** so there has been some improvement.) Other areas that slow the process down include planning (**39%**), code development (**30%**), and code review (**28%**).

Anecdotally, our respondents had a lot to say about these areas:

“Not enough tests (or none) and then the code doesn’t work in production. Some collaborators have poor IT skills.”

“Planning is somewhat heavyweight and a little less than agile.”

“Many people find it a chore to review code.”

“Too little testing done too late.”

“We are slow and do not test very well. We do Big Bang deployments.”

“No designated tester so the developer or whoever gets time deals with it.”

“We do a lot of testing. So we deploy several variations of the same code in different branches multiple times a day. This often gets confusing.”

“I am working without much planning.”

“Planning in the form of some teams doing waterfall and others doing ‘wagile.’”

“Poor planning leads to a lot of doubling back.”

“Code reviews can take a long time due to the lack of reviewers.”

“We have a strict code review process and it often takes several days for the reviewer to respond to requests for review.”

“Code review takes time and every developer has to explain how he achieved what he did.”

Open source continues to be an important piece of software development, but it's also clear that smaller meaningful projects remain attractive to developers. Just over **63%** of those surveyed said they participate in open source projects, but their efforts are spread broadly across both well-known and lesser-known projects. Exactly half of survey takers chose "other" and indicated their open source efforts were aimed at small personal projects. Slightly more than **17%** said they contributed to GitLab; other commonly mentioned projects were Typo3, WordPress, smaller projects for GitHub, and Django.

Where the tools rank

Git is the choice for source control for **92%** of the survey takers. Just **2%** use no source control and even smaller percentages mentioned Team Foundation Server and Subversion.

Almost **59%** of those surveyed use GitLab, while **23%** use GitHub, and **11%** use BitBucket. For builds, **60%** use GitLab and **38%** use Jenkins. Almost **11%** use GitHub Actions while almost **9%** don't use anything.

Almost **40%** of survey takers said they "partially" use microservices, while **26%** fully use them and **31%** don't use them at all. Some respondents said they were planning to or are investigating microservices, while one said, "We tried it and didn't like it." The picture is not that different when it comes to Kubernetes: **38%** use it while **50%** do not.

Anecdotally, it seems many organizations are investigating Kubernetes, however:

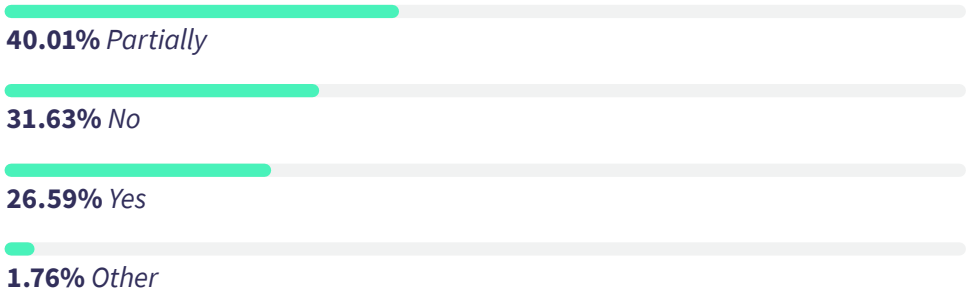
> **"Starting soon."**
>
> **"We're trying to get there."**
>
> **"It's a priority for our platform team."**

"We're currently using swarm orchestration in Docker Enterprise but plan on moving to Kubernetes later this year."

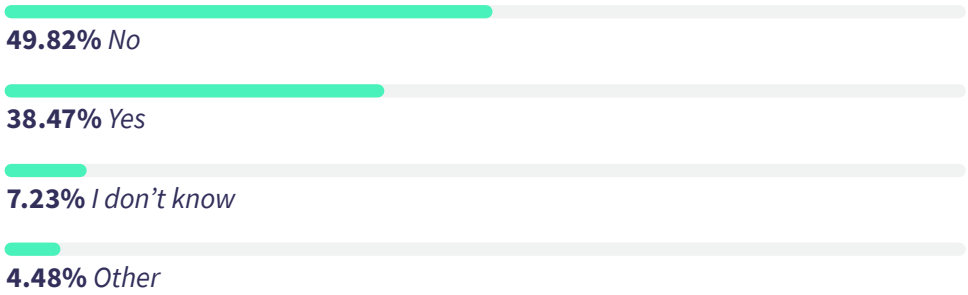
"We are on the path to get our monolithic server into a sert of microservices and the goal is to use Kubernetes to help on this side."

And despite the buzz around low code/no code tools, more than **75%** of respondents don't currently use one.

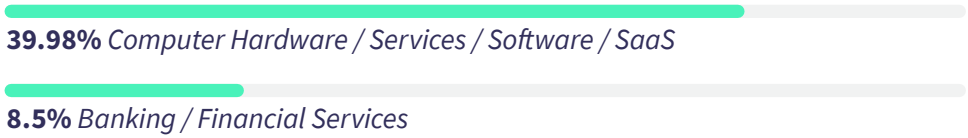
Has your organization adopted microservices?



Does your organization use Kubernetes?



Does your organization use Kubernetes?



A look at the status quo

On the whole, software development teams appear reasonably happy with their organization's processes, tools, and ability to innovate. Almost **69%** agreed at some level that the development processes used in their organizations are the best for the job, and an even higher percentage (**76%**) think those processes help them innovate. Almost **83%** think the tools on board are designed to help them succeed, while **73%** think the tools are the best available. And finally more than three-quarters (**77%**) think the tools on board help them innovate.

Developers

Development top findings

DevOps = faster releases

If you're a developer, DevSecOps just works. Nearly 83% of them report they're releasing code more quickly.

Investing in the future

The top three areas for investment this year: CI, SCM, and test automation.

More testing STAT

Devs are nearly unanimous in the need for their organizations to do more testing. Apparently, you can't have too much testing.

A new job description

Devs report responsibilities (creating/monitoring/maintaining infrastructures) that look a lot like what Ops used to do.

Who owns security?

It depends on who you ask. But, more than a quarter of devs feel **solely responsible** for security, indicating that shifting security left has begun in earnest.

Devs and DevOps

How can you release code faster? The answer is DevOps. Almost **83%** of developers say they're releasing code faster and more often today than ever before thanks to DevOps. About **35%** report code is released twice as rapidly, while nearly **25%** said it's now being released 10 times faster. That said, **29%** actually don't know how much faster code releases are (but are clearly confident it's faster than it used to be). One of the major issues reported by many survey takers is that their organizations simply don't have any way to accurately measure release speed.

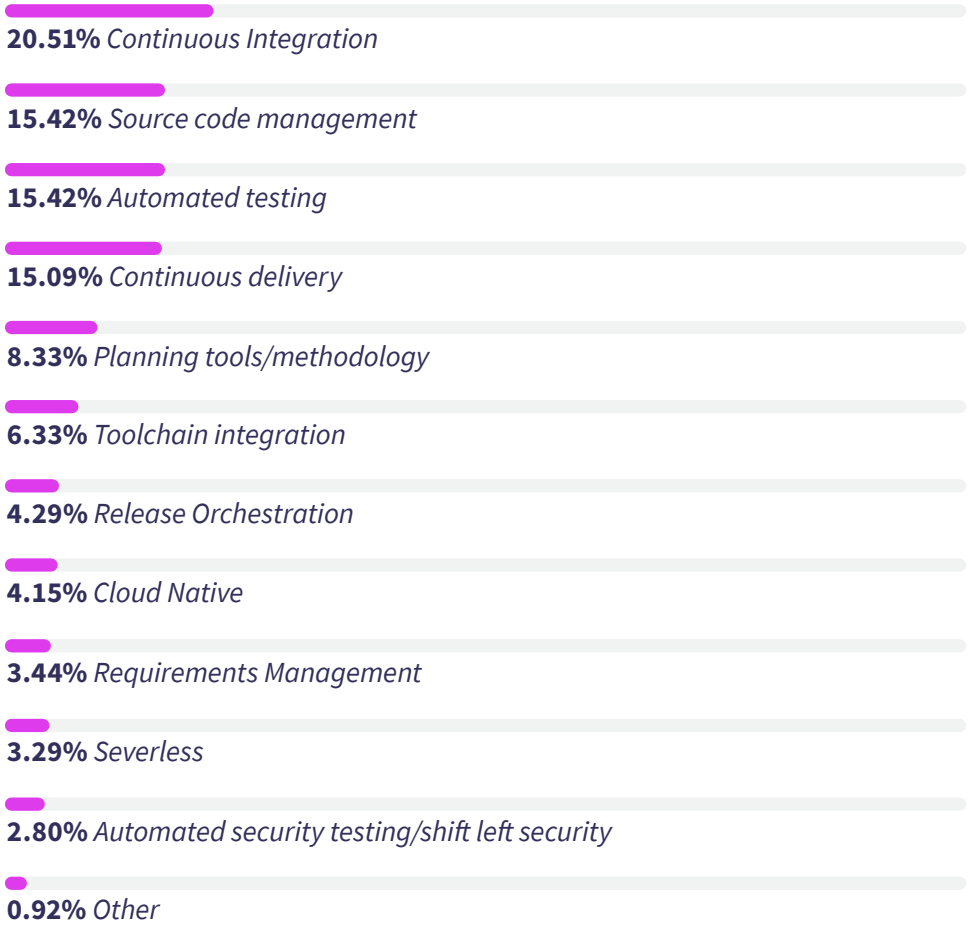
> **"We don't measure yet."**
>
> **"Unattainable ROG. LIGHTYEARS AHEAD."**
>
> **"Somewhere between 2x and 10x."**
>
> **"Up from 2 days to hours."**

"Developers are now deploying at will whereas earlier deployments had to be planned and scheduled outside business hours."

"Went from once per week (plus emergency releases) to 3x per week (plus emergency releases)."

Releases are coming more quickly, but why? We asked developers what's changed in their process. Almost **21%** said they've added CI to their development cycle, while just over **15%** brought in source code management, automated testing and CD. About **8%** have upped their game when it comes to planning tools and methodologies. Anecdotally, developers also said they're trying Kubernetes, integrations with low code tools, and more communication.

What changes have you made to your software development process?



Developers shared some detailed insights into how these changes have helped them speed up the delivery of code:

“A templatized CI/CD process has significantly sped up build and deploy times to multiple environments in multiple clouds.”

“Automated testing and continuous integration have made our deployments safer and more optimized. Now everyone in the team has the permission to deploy the code.”

“Deployment has become a non-task. Bootstrapping new projects is 10x faster because of the reusable infrastructure.”

“Automation within the CI/CD pipeline (including test automation and the actual CD automation part) has significantly increased the delivery speed of our team.”

“Pre-deployment tests have provided more confidence that the product is ready to be released, also delivery frequency has increased.”

“We've set up automated processes to build, test, and deploy code using a mixture of our own tools and open source tools.”

“Automated tests, automated deployment on code review approval.”

“Automated testing using GitLab CI has meant less overhead when reviewing code and quicker and safer deploys.”

“Ticket with 7 departments to “button press” went from 6 weeks to 2 hours.”

“We're now avoiding duplicates because everything is automated. There is no need to run test commands as it's performed automatically.”

“CI and CD tremendously reduced time for build and deploy applications and eliminated problems with the build environment.”

“Because of Test Driven Development (TDD) and CI we build and deploy rapidly with confidence.”

“Automation has made one-click testing and deployment possible for us.”

“With GitLab, everything is on one platform, allowing us to fly through the development process and collaborate more.”

> **“We reduced our CI build queue time by 75%, which allowed developers to have test results faster and allows QA to have build artifacts to test faster.”**
>
> **“Splitting up code changes into smaller, more manageable pieces for the code review. Started using TDD.”**

That’s what they’re doing, and how they’re doing it. But what don’t they do any more?

“No need to manually merge my code and push to staging and then production.”

“We still do the same things, just much faster and much simpler”

“We don't have to code our product to work with different platforms. We can just code our product and integrate it with a tool to work with different platforms.”

“Manual testing”

“Manual deployments”

“No longer need to write extensive deploy mechanisms.”

“Strict TDD.”

“Sync the code between multiple devs; Git does it well.”

“Manually test, argue about code style, update dependencies.”

“No longer using fat slow IDEs :D”

“Argue a lot about requirements; I've decided to leave that to more business-y people.”

“Create a ticket to ask Ops to deploy.”

That being said, developers aren't complacent. In fact, the list of things they feel they **should** be doing now is longer than the list of things they're no longer doing.

Writing more tests

Testing

Automating tests

Unit tests

Functional tests

Security tests

A/B testing

End-to-end tests

More testing

More test cases

Shift left on testing

More test cases to cover 100% of everything

Automating everything

CI/CD x3

Code reviews

Faster code reviews

Better code reviews

More code reviews

Containers

"Everything"

Developer daily life

It's a changing world for developers and that continues to be true when it comes to their responsibilities. Dev has bled into operations, clearly: **35%** of developers say they define and/or create the infrastructure their app runs on and **14%** actually monitor and respond to that infrastructure - a role traditionally held by Ops. Over **18%** instrument their code for production monitoring, while **12%** serve as an escalation point when there are incidents.

When it comes to priority setting, it's product managers FTW – **46%** of respondents said PMs set the course, while **30%** said it was the business side and just **24%** cited developers. And when it comes to prioritizing work and features, **33%** said the product roadmap timeline is the most important, followed by the estimated cost of development, and the ROI (both approximately **25%**).

Nearly half of all developers, **49%**, conduct code reviews weekly and only **14%** do so biweekly. But nearly **20%** offered alternate views (and it's important to remember that code reviews are one of the top four steps that survey takers said delay releases.) Although some developers said they did no code reviews, a large number said code reviews are a daily occurence.

"Each merge request is code reviewed by a peer; no team code review."

"Constantly – every merge request triggers and requires review before acceptance."

"Code reviews for new merge requests every day."

Not surprisingly then, **95%** of respondents said code reviews are either very or moderately valuable for improving security and code quality.

They're busy doing code reviews, but most developers are apparently not spending time maintaining and integrating their tool chains. Almost **60%** said they spend either no time or less than **10%** of their time on toolchain integrations or maintenance, and about **22%** spend just **11%** to **20%** of their time on that task.

Speaking of tools, nearly **87%** of developers said API coverage and support is either very important, important, or somewhat important when it comes to choosing a development tool.

Security

The subject of security is a tricky one no matter where you sit in a development organization. Ideally, security is shifted left, moving it much closer to those creating the code, and **28%** of our developer respondents said that they are completely responsible for security in their organizations. But **41%** said they were "responsible, but part of a bigger team," while **21%** said they do their part but "someone else" actually owns the security responsibility.

"I am the only one who actually cares about security in my org."

"I regularly put security suggestions in the 'box of suggestions to be ignored.'"

> **"I actively advocate for security best practices but it falls on deaf ears."**
>
> **"App security is not really a focus at our company."**

All that said, a majority of respondents, **66%**, said their organization does make it possible for them to avoid hacks.

> **"The way dev teams handle security varies wildly."**
>
> **"Communication between the team, code reviews, tracking commits, reviewing the work, computer and processes to maintain privacy."**
>
> **"Code review and continuous reevaluation of projects and we strictly remove unused code."**
>
> **"There's a security team, but it doesn't involve face to face with us, the dev team. So we just run the dev process without counting on them."**
>
> **"We rely on static code analysers to keep track of bad practices and bugs that could result in vulnerabilities."**

“It varies project to project. DevOps is usually tasked with 'protecting' our environments. We devs try to follow industry standards code-wise.”

“We have a daily security audit on our entire stack including containers. Also one of our maintainers is a highly security-conscious developer, who points all of us in the right direction when we stray.”

> **“A lot of security concerns are handled by a dedicated team, sitting in the frontline of all site traffic. Monitoring, identifying and responding always involves them.”**
>
> **“Company neither enables nor hinders security, it's currently up to the engineers.”**
>
> **“Throttling, blacklisting IPs, disabling user accounts automatically.”**
>
> **“You write it, you try to break it (from the outside).”**

Looking to the future

Like their counterparts in Ops, security, and test, a majority of developers (**29%**) think “soft skills,” including communication and collaboration are going to be most important for their future career development. At the same time, **22%** said AI/ML (a much higher percentage than reported from any other group), and **17%** said GitOps. Nearly **15%** said they would benefit from knowledge of advanced programming languages.

Anecdotally, a large number of survey takers said they’d benefit most from a better understanding of DevOps overall, something that tracks with the large numbers of practitioners who feel they are still in the early stages of the transition. Others said they want to know more about low code/no code, containers, serverless, and infrastructure as a code. But this quote sums it up well:

> **“SecOps, GitSecDevOps, PartyOps, AllTheAboveOps”**

About three-quarters of developers (**71%**) feel prepared or well prepared for the future, but **24%** said they’re “not very prepared,” and roughly **3%** said they’re overwhelmed.

Security

Security top findings

DevSecOps = changing roles

Security can be found on cross-functional teams and working closely in collaboration with developers, both of which represent significant change from the past.

You call that a shift left?

Sec pros report their orgs are shifting security left but they really aren't doing the scans to support that claim.

Why can't we be friends?

Just like in last year's survey, security pros think devs don't find enough of the bugs at the earliest stages and are slow to prioritize fixing them.

Cutting edge is left out

Most sec teams don't have security processes in place for microservices/containers/APIs/cloud native or serverless.

Who owns it?

Almost **33%** of security respondents said they were responsible for security. But nearly as many, **29%**, said **everyone** was responsible for security. Clarity is needed.

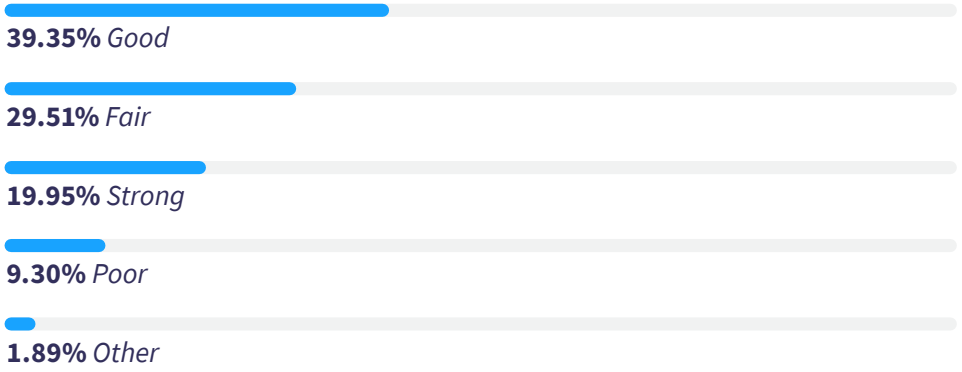
Security and DevSecOps

It's not a shock to hear that application security remains a work in process at many companies. While **39%** of respondents rated their organization's security efforts as "good," nearly **30%** said they were only "fair," and just **20%** called them "strong."

"We are paranoically secure."

> **"I would say good, but it can always be improved. If anyone says 'strong,' they are lying."**
>
> **"The team is trusted to do its own security research and implementation. We don't know how good or bad we are."**

How would you rate your organization's security efforts?

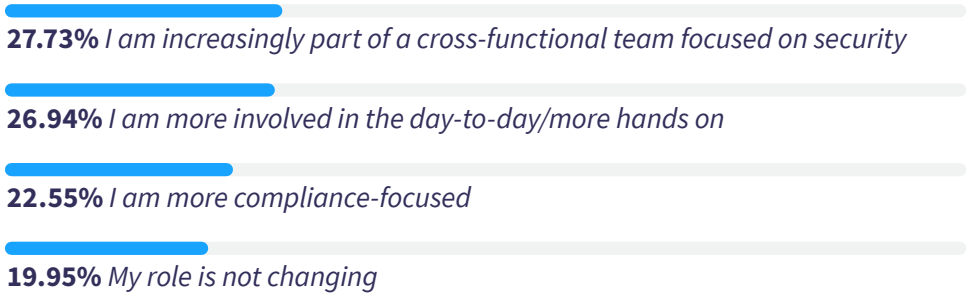


Roles are changing

After what seemed like an eternity of being outsiders looking in to software development, security pros now report their roles are beginning to change. Nearly **28%** reported being part of a cross-functional team focused on security (perhaps really putting the "sec" in DevSecOps). And almost the same number, **27%**, said they were more hands on and involved in the day to day development activities. About **20%** said they didn't see any changes in their roles.

"(Security) is becoming less focused into silo positions and more of a Jack of all trades role."

In your experience how is the security role changing?



Shifting left

A full **65%** of security pros reported their organizations are, in fact, shifting security left, meaning that security is brought into the development process earlier.

But it's a fair question to ask how early is early? Only **24%** said their companies have static application security testing (SAST) scanners in a web IDE. More concerningly, though, less than **19%** put SAST scan results into a pipeline report a developer can access. Dynamic application security testing (DAST) fares even worse — less than **14%** of companies give developers access to those reports. The bottom line: If you want to enable developers to find and fix vulnerabilities, you have to give them the scan results in their pipelines or native workflows.

Over **60%** of developers don't actually run SAST scans, and **73%** don't conduct DAST scans. A majority of devs, **56%**, don't run container scans, and only roughly half run compliance scans. One bright note: about **57%** do conduct dependency scans.

As we saw in last year's survey, security pros and developers often remain at odds about priorities. When asked how developers do finding bugs versus security teams, **93%** gave developers credit for discovering only **25%** or less of the bugs to be found in existing code, leaving three-quarters of the bugs for security to find at a later stage in the process.

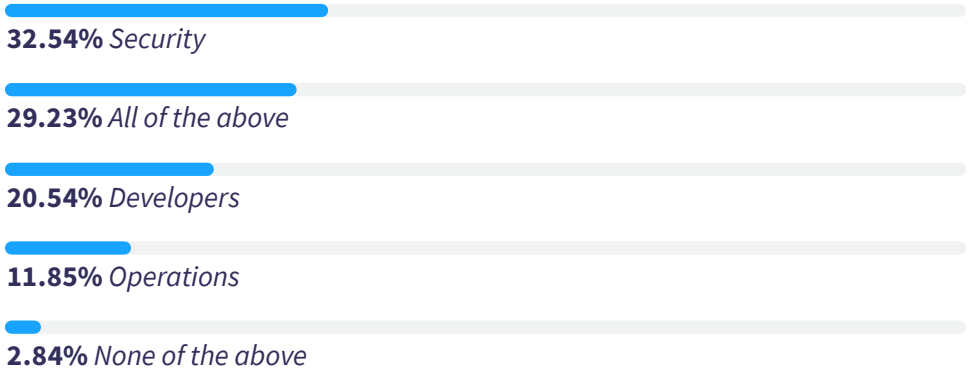
And the clashing perspectives don't stop there: While **65%** of security pros agreed at some level that it was a developer performance metric to find vulnerabilities, an even higher percentage (**69%**) said it was difficult to get the developers to actually prioritize fixing the bugs. All told, **61%** of security respondents agreed at some level that vulnerabilities were mostly found by security pros (not developers) after code is merged in a test environment (which is relatively late in the process). Clearly this situation remains far from ideal.

Who's in charge?

As we saw in last year's survey, part of the challenge with security is the question of ownership. Most respondents, roughly **33%**, said the security team was responsible for security but almost as many people, **29%**, said **everyone** was responsible.

> **"The DevOps team is in charge."**
>
> **"We engage external experts and also utilize in-house expertise."**
>
> **"We don't have separate security, developers and operations; we are DevSecOps (and more)."**

In your organization, which group is primarily responsible for security?



The nitty gritty

Security testing remains a serious source of frustration: Over **42%** said it happens too late in the lifecycle, **36%** reported it was hard to understand, process, and fix any discovered vulnerabilities, and **31%** found prioritizing vulnerability remediation an uphill battle. And nearly **30%** complained it was hard to find someone to actually fix the bugs.

If vulnerabilities are found, **64%** said the severity level is what matters the most, followed by the category of vulnerability (**47%**), the number of vulnerabilities solved (**42%**), and the number of vulnerabilities spotted (**36%**).

When it comes to more modern application technologies - microservices, APIs, containers, and cloud native/serverless — security pros were most likely to say they did not have processes in place to monitor and protect them. Those that did relied heavily on tools including Prometheus, Grafana, and AWS Cloudwatch.

When asked specifically about cloud native and serverless, nearly **64%** said they have added no security capabilities at all.

Most security pros report a mixed bag of how their organization actually operates. Over **62%** say security leadership in their companies can get an accurate view of their team's performance, but **53%** agree at some level that their efforts to fix bugs are slowed by red tape.

> **"It's just a lot of manpower."**

> **"We monitor the system and collect metrics to ensure applications running on containers are performing properly. Metrics are tracked and analyzed in real time to determine if applications are meeting expected goals."**

Looking to the future

When asked what skills will be important for the future, nearly **28%** said soft skills like communication and collaboration (an answer which tracks with their developer, operations, and test counterparts, by the way). But **25%** of security pros also thought subject matter expertise would be important and **22%** said advanced programming skills. About **15%** said it would be important to have a better understanding of artificial intelligence and machine learning.

"Infrastructure knowledge. Your code can be clean all day, but if someone opens up the wrong firewall port or if there is a logic flaw in the application, lack of code level vulnerabilities doesn't matter."

> **"The more the better. Old-school hacking mindset and technical excellence is a must though."**

> **"Advanced knowledge of Kubernetes and containers."**

> **"White hat skills."**

Operations

Operations top findings

DevSecOps = changing roles

Over **60%** report new and different responsibilities because of DevOps.

In the cloud

Over half of Ops pros say their primary focus today is managing the organization's cloud services.

Automation is on the move

Almost **40%** of operations team members said their development lifecycle is "mostly" automated.

Blending in with dev

Almost **70%** of ops pros report that devs can provision their own environments, a sure sign of shifting responsibilities brought on by new processes and changing technologies.

Staying secure

Over 21% of Ops pros say they feel solely responsible for security in their organizations.

Operations and DevSecOps

While it's safe to say DevSecOps has brought change to every software development role, perhaps nowhere is that more true than in operations where process changes, tech changes, and cultural changes all seem to collide.

We asked Ops pros to briefly outline their daily responsibilities, and their responses reflected this changing mix of skills and responsibilities:

"60% new project work and 40% operations/fire-fighting/developer support."

"Anything between dev and ops. From planning to deployment but not monitoring and maintaining apps in production."

"Ensure reliability and availability, improve developer efficiency, automation, tools, and observability."

"Build out and improve CI/CD platform."

"Jack of all trades."

"Keep the lights on."

And those changes are coming directly from a switch to DevSecOps. A majority of Ops team members (**60%**) report that DevOps has changed those responsibilities over time. Today, **42%** see their role as primarily managing hardware and infrastructure, while **52%** say their first priority is managing cloud services. And they're not bogged down with tracking compliance issues; over **55%** said they spend **10%** or less of their time dealing with audit and compliance issues.

So many tools

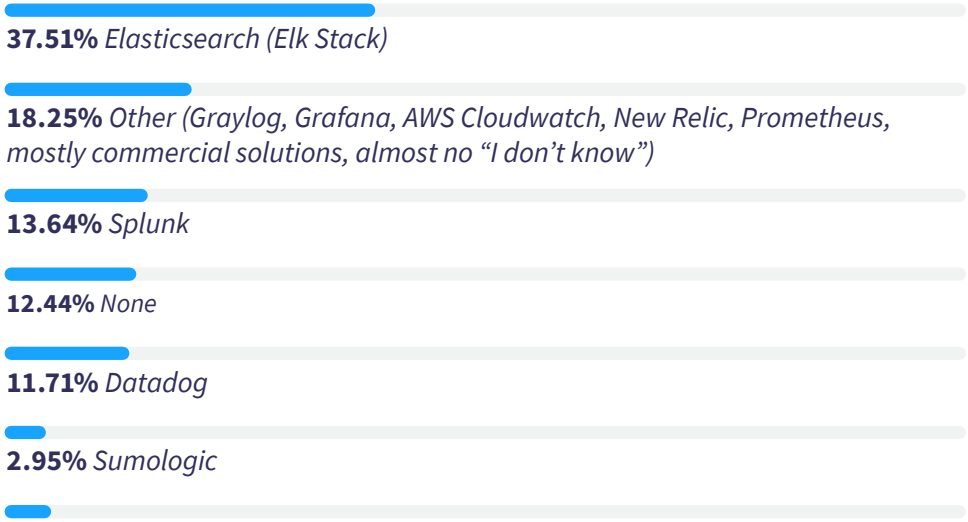
The Ops team has changing responsibilities but still a lot of tools to manage. Luckily, operations team members have a lot of street cred when it comes to tools and processes: **75%** agree at some level that their recommendations about tools and practices are likely to be followed by their employers.

A majority of ops teams **(65%)** use between two and five monitoring tools, while **18%** use one, and about **12%** don't use any at all. A majority of those monitoring (**64%**) use a tool that feeds real-time data to developers. Logging, where companies capture and view application logs, and metrics monitoring are the most important categories to monitor, according to a majority of Ops pros, while ping and synthetic monitoring are least important.

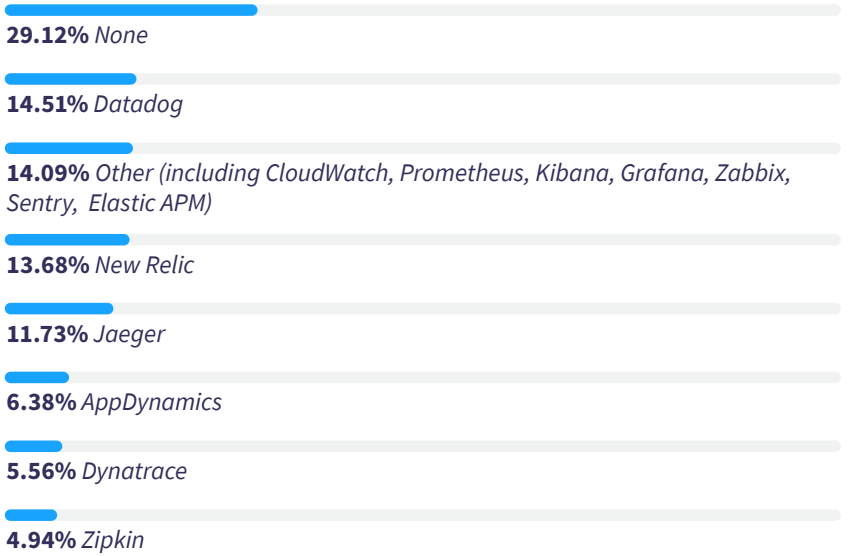
Most operations teams choose Elasticsearch to view/capture logs (**38%**), followed by Splunk (**14%**), and Datadog (**12%**). Nearly **20%** chose "other," however, and mentioned choices ranging from Graylog to Grafana, CloudWatch, Prometheus, and New Relic. Nearly **30%** of companies say they don't use any tools to capture app metrics (traces), but of those which do Datadog was the most popular at **15%**. Nearly **36%** of companies use Prometheus to capture time series metrics.

It's an AWS cloud world for the majority of operations teams (**35%**), but **21%** said they use Google Cloud Platform, and nearly **18%** provision Azure. Over **18%** either don't use a public cloud hosting service or don't know which one their organization uses.

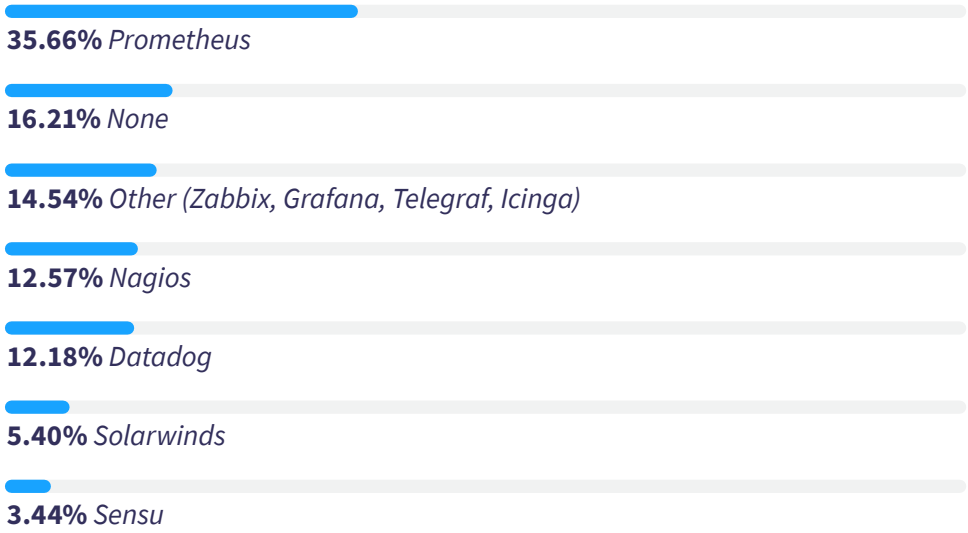
Which tools do you use to view/capture logs?



Which tools do you use for app metrics (traces)?



Which tools do you use to capture time-series metrics?



Working with development

A slight majority of Ops teams, **38%**, described the development lifecycle in their company as "mostly" automated, while **35%** said it was "partially," and nearly **16%** indicated they were "just beginning." Only **3%** said there was no automation at all, and just **8%** opted for "complete" automation.

But no matter the level of automation, over **80%** of Ops pros said it was "very or extremely important" to them to have visibility into the development side. Along with good visibility, an Ops team needs sufficient notice to support the Dev side, and **76%** agree at some level that they're getting the "heads up" they need to be efficient. In another sign of change on the Ops side, over **69%** agreed at some level that developers are able to provision their own environments. And nearly **66%** said Devs in their organizations receive and address feedback on security issues during the development process.

Speaking of security, most operations team members (**45%**) believe they're responsible for it, but as part of a bigger team. Just over **23%** say "they do their part but someone else owns it," while **21%** feel completely responsible.

Looking at the future

Like their colleagues in development, security, and testing, a slim majority of operations professionals (**31%**) believe soft skills like communication and collaboration will be most important for their future careers. Programming followed closely at nearly **29%**, followed by subject matter expertise (**23%**), and AI/ML (**12%**).

> **"Operating system and tools skills (i.e., Kubernetes, Docker) troubleshooting skills, SecOps, NoOps."**

"A more hacker-like skillset is required."

"Understanding the layers between where code runs and the end hardware it runs on. After all, 'serverless' isn't."

"To convince their employers that they should not be 'ops professionals' and their 'ops' skills should be part of the broader development team."

Testing

Testing top findings

DevSecOps = closer collabs

About **33%** of testers report closer (and presumably happier) collaboration with developers than ever before.

Automation is happening, slowly

Just **12%** claim to have full test automation, but almost **35%** of testers said their orgs are almost half way there.

Some shift left

Almost **75%** of testers say their orgs have shifted testing left (meaning closer to development).

QA departments aren't shrinking

Almost **60%** said their teams are the same size they were last year, despite advances in automation.

The bots have it

A small but intriguing percentage of test teams (**16%**) either use "bots" to review their code or have an AI/ML tool in place for testing.

Securing security

23% of testers think they're solely responsible for security in their organizations.

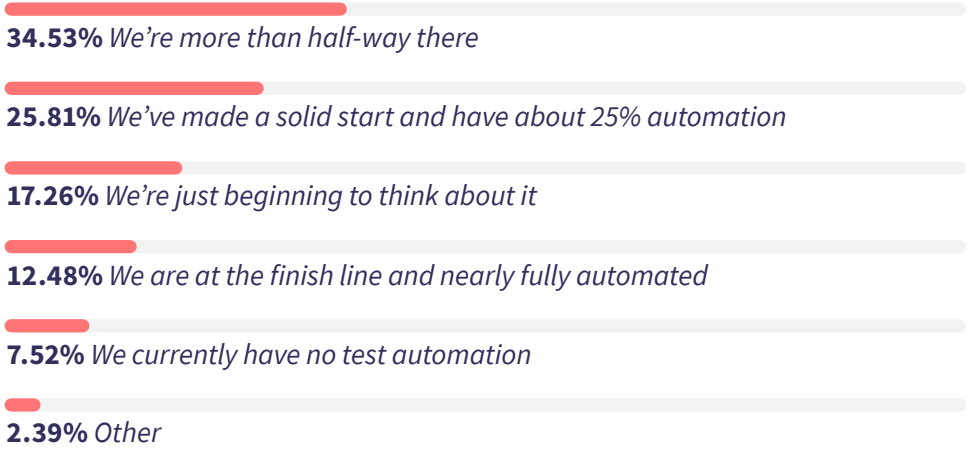
Testers and DevSecOps

Let's face it — software testing is hard. So hard, in fact, that for the second year in a row developers, security, and operations pros who took our survey were unanimous that testing remains the number one reason for delays.

We know that cracking the testing code is the key to successful DevOps and it's clear that's still a struggle. But there are signs of forward momentum, at least when it comes to test automation. Almost **35%** of testers surveyed said their organizations were more than half-way to full test automation, and just over one-quarter are about **25%** automated. About **12%** say they're nearly fully automated while another **25%** are just starting on the journey or have no test automation whatsoever.

> **"We choose to have little functional test automation."**
>
> **"We made a start but it's not proving either effective or efficient as yet."**

Which tools do you use to capture time-series metrics?



And more good news: **74%** of organizations report they have shifted testing left, meaning they've moved it earlier into the development process. What does that actually mean? Approximately **31%** said developers test some of their code and **25%** said automated testing happens as code is being written. About **17%** said dev and test work as a team to test "as close to real time as possible," and about **9%** said they practice test-driven development. Just about **14%**, though, said they haven't shifted left very far and that testing is still not happening early enough.

"Practically no testing is happening."

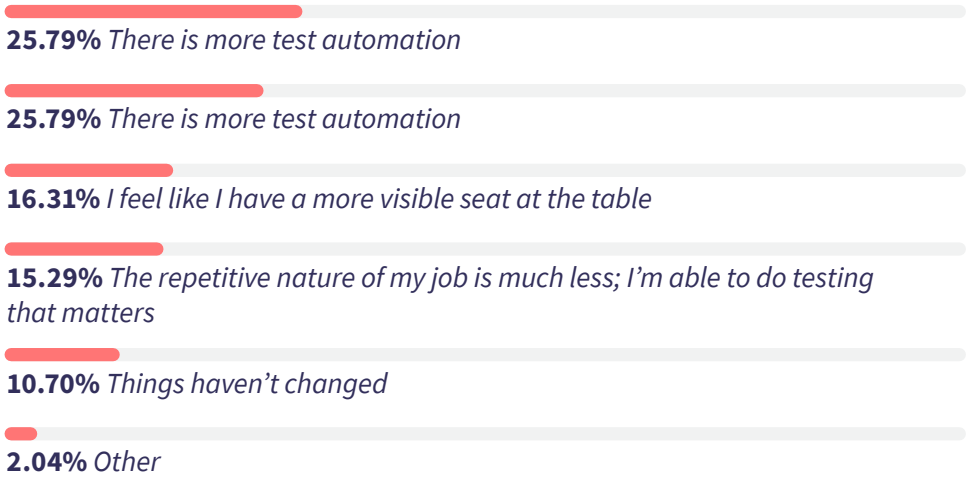
"We do TDD. QA and dev act as a team. We have automated tests running parallel with developing code."

"We have a separate QA section of our IT staff that focuses on testing and they collaborate with development to make fixes."

And DevOps has made a difference in QA's daily responsibilities. Nearly **30%** of testers say they're now working more closely with developers and **26%** report more test automation. Thanks to DevOps, **16%** of testers feel they have a more visible seat at the table and **15%** are now able to do more testing that matters rather than repetitive busy work.

> **"We have to write less paper and tickets and have faster reaction times."**
>
> **"We're all the same — dev team is the ops team."**
>
> **"We're starting to see light at the end of the tunnel."**

How have your daily responsibilities changed because of DevOps?



A new landscape

More modern testing methodologies are also reflected in where teams choose to place their QA departments. In over half of responding companies (**56%**), QA teams are embedded with developers. Only **22%** of testers said they were a separate organization and **15%** said they were part of operations.

"We don't have a QA/Test team; we practice TDD."

"We have none. Developers are responsible for all testing."

"We have teams where QA is embedded and a couple where it is a separate org."

These structural changes seem to be working. About **26%** of testers said they now have access to monitoring data, thanks to the way their DevOps team is structured. About **21%** have visibility across the organization (dev, sec, and ops), **18%** have "sufficient" automation, and **13%** have workflows that don't get in their way.

Most testers aren't losing their jobs either, despite widespread fears that automation might mean a decrease in headcount. Just shy of **58%** of respondents said their QA staffing has remained the same over the last year, while **25%** have seen an increase. Just **17%** of survey takers said the QA team has decreased in size.

The nitty gritty

Security is a complicated issue no matter where you sit in the organization, so it's not surprising that testers' views are split on the subject of responsibility for security testing. In fact, **23%** said they were completely responsible and **23%** said they were only partially responsible. Nearly **37%** said they share responsibility with security pros and developers and **16%** said they weren't responsible for security testing at all.

Testers were also divided on the role AI/ML is playing in their organizations. About **16%** either already use "bots" or have an AI/ML tool reviewing code before they see it, and another **18%** are exploring the idea. Just over **22%** said they've talked about it but nothing has come of it so far, and **44%** aren't considering AI/ML in testing at this time.

Looking to the future

Like their counterparts in dev, sec, and ops, a majority of testers (almost **31%**) said soft skills like communication and collaboration were the most important for their future. Coming in a close second was programming (**30%**), followed by subject matter expertise (**28%**).

> **"Usability and general knowledge of competitive solutions in the market"**
>
> **"Agile development, DevOps"**
>
> **"Grey brain cells and detective work"**

On the whole, **65%** of testers feel adequately prepared for a future in QA. But many reported a rapidly changing landscape that is potentially unsettling.

> **"We need more management."**
>
> **"I will never be prepared enough."**
>
> **"A lack of business experience leads to less efficiency in testing code."**
>
> **"Cultural change is the most difficult."**

Looking ahead

This survey was completed on February 29, 2020, and thus it cannot reflect the reality of the post-pandemic world.

Our survey takers had a long list of areas to focus on for the future. Automation, CI/CD and DevOps were the top picks but interest was expressed in AI, microservices and even augmented and/or virtual reality.

It's obvious any DevOps practice needs to have lifelong learning built into it. No one settles into DevOps — they simply try to keep pace.

And that's our best advice for you: Pay attention, be open to new things and think about how your team's roles and responsibilities are changing. How prepared is everyone for the future? What more should you be doing?

But let's end on a high note. We asked developers how prepared they are for the future: **71%** said prepared or very prepared, while less than **25%** said "not very prepared." But we like this comment left from one developer, who has the lifelong learning baked in:

> **I'm only prepared because I constantly keep tinkering on the side.**


GitLab