GitLab

A maturing DevSecOps landscape

2021 Global Survey results

- 4300 respondents
- Dramatic advances in release/deployment frequencies, automation, security
- Developers, operations and security pros in their own words

Table of contents

Introduction

For the fourth year in a row, we asked DevOps teams to tell the truth about their practices and processes, their challenges and their careers. With a global pandemic swirling, we were surprised when nearly 4,300 people took time to do just that this past February.

We were even more surprised by the results. This year, for the first time ever, DevOps became serious. It's somber. It's grown up. *It's happening*. We didn't ask a single question about Covid, but the answers seem shaped by that lingering experience. It's as if, in the face of calamity, teams everywhere decided to focus on what mattered most, whether that was automation, or testing, or embracing cutting-edge technologies.

In 2021, teams are poised to step out of the DevOps "culture" battle and into the real work of technology implementation and (surprisingly) upbeat results.

60% of developers are releasing code 2x faster than before, thanks to DevOps – up 25% from (pre-pandemic) 2021.

72% of security pros rated their organizations' security efforts as "good" or "strong" – up 13% over 2021.

56% of ops teams members said they are "fully" or mostly automated – up 10% from 2021.

Almost 25% of respondents claimed to have full test automation – up 13% from 2021.

75% of teams are either using AI/ML or bots for test/code review, or they're planning to – up 41% from 2021.

Last year dev, sec, and ops said they needed better communication and collaboration skills for their future careers. This year, after an intense period of enforced soft skills, their priorities have shifted dramatically to AI/ML (devs), subject matter expertise (sec),and advanced programming (ops).

Anecdotally, we heard about the hard work that went into these results: mindset shifts, tough discussions, and detailed analysis. DevOps isn't easy (even for us) but it seems a focus on outcomes using real data can help teams of all sizes in all countries move forward.

As always, we'll remind you this is our survey, so don't be surprised if respondents mention us or use our products (roughly 50% of survey takers are GitLab customers). Also, 43% of our survey takers have been "doing" DevOps for between three years and five years or more, so they're seasoned practitioners with what are often aspirational results. Your results may be different, and that's ok.

Let's get started.

Overview

2021 DevSecOps Survey top findings

DevOps = broader technology base

CI/CD still matters, but DevOps platforms are on the rise as is AI/ML.

The trouble with testing

For the third year in a row, respondents pointed to testing as the primary reason for release delays.

Want better code quality?

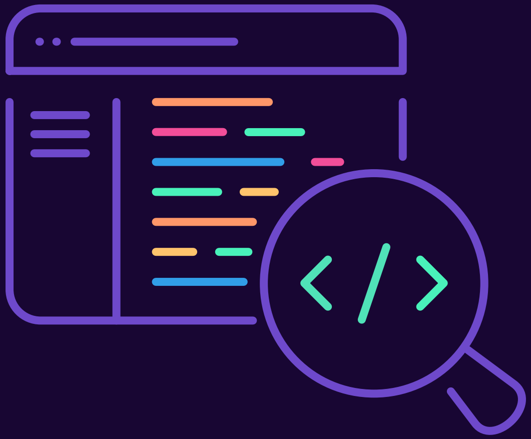
Then it’s DevOps hands down. Other benefits include improved time to market and better planning.

Getting serious

Last year, teams talked about Kubernetes and microservices; this year, they’re using them, or planning to soon.

Not just dev and ops

A DevOps platform offers something for everyone – literally. A full 23% said everyone in their company uses the DevOps platform.



The starting point

Here's a closer look at the nearly 4,300 (4294) people who completed the survey between February and early March 2021.

ROLE

41.67% Software Developer / Software Engineer

8.59% DevOps Engineer

6.87% Development/Engineering Leadership

7.88% Software Architect

4.56% Other

2.52% DevOps Leadership

2.81% Technology Executive - CIO / CTO/VP

4.11% Project Manager

1.66% Site Reliability Engineer

2.57% Systems Administrator

1.91% Systems Engineer / Network Engineer

2.23% Product Manager

1.56% Quality Assurance

1.14% Operations Leadership

GENDER

14.95% Female

81.86% Male

0.93% Non-binary/third gender

0.51% Prefer to self describe

1.75% Prefer not to say

INDUSTRY

40.04% Computer Hardware / Services / Software / SaaS

6.3% Banking / Financial Services

4.78% Other

18.05% Education

5.29% Business Services / Consulting

3.4% Telecommunications

2.19% Media & Entertainment

1.91% Healthcare

2.24% Government

REGION

20.28% Europe and Russia

10.78% North America

50% Asia

2.28% South America

0.77% Australia and New Zealand

1.82% Africa

0.98% Middle East

NUMBER OF EMPLOYEES

25.37% 1 - 10 people

21.34% 11 - 100 people

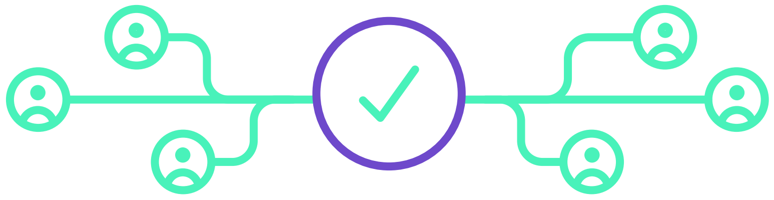
5.11% 101 - 500 people

8.14% 501 - 1,000 people

12.31% 1,001 - 10,000 people

11.08% 10,000+ people

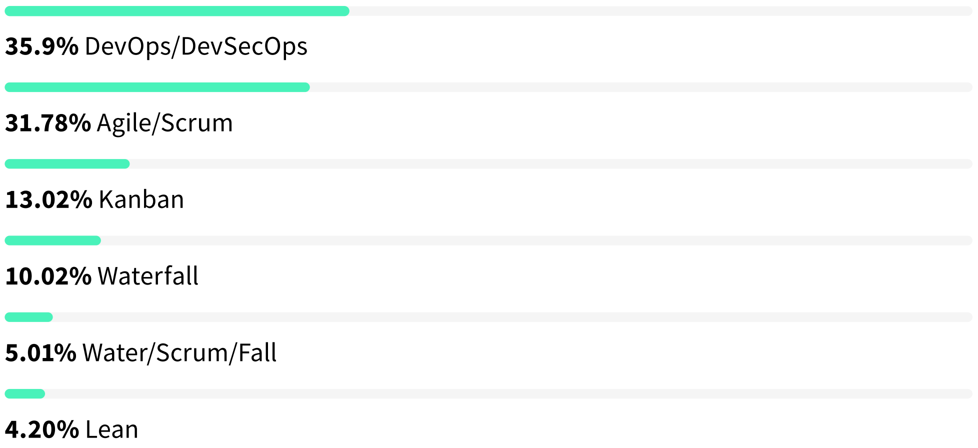
5.92% Don’t know



Software development today

In 2021, a majority of survey respondents (35.9%) told us their teams develop software using DevOps or DevSecOps, followed closely by Agile/Scrum at 31.78%. That's a big jump for "self-identified" DevOps usage in a year: in 2020 only 27% of teams described their process as DevOps or DevSecOps. Just over 10% said their teams use Waterfall (up from under 8% in 2020) and, identical to last year, 5% of teams are "creative" and describe their process as Water/Scrum/Fall.

MOST PRACTICED DEVELOPMENT METHODOLOGIES:



35.9% DevOps/DevSecOps

31.78% Agile/Scrum

13.02% Kanban

10.02% Waterfall

5.01% Water/Scrum/Fall

4.20% Lean

Just over 30% of respondents said their DevOps practices are between one and three years old. Almost 27% have had DevOps in place for a year or less, while nearly 23% have been doing it for five or more years. About 20% are in the DevOps "sweet spot" of between three and five years, meaning they've known success and are comfortable with the processes and routines.

What do today's DevOps implementations look like? CI/CD was the most likely to be part of the process, followed by DevSecOps, test automation, and a DevOps platform. In 2020, just 4% of respondents used AI/ML in DevOps; this year, 11.5% reported they do.

Other technologies mentioned include:

- **Infrastructure as code (Iac)**
- **GitOps**
- **Kubernetes**
- **NetDevOps (networking at scale)**
- **Platform "bootstrapping" using CI/CD**
- **Extensive use of SREs**
- **Unix**

For the second year in a row, respondents said devs are the most likely to benefit from a DevOps practice (36%), followed by ops (22%), security (16%), with QA and the business side coming in at 13%.

The top three reasons to choose DevOps? Code quality, faster time to market, and security. Other clear benefits from a DevOps practice: improved communication/collaboration and happier developers, both of which rated much higher in 2021 than in 2020.

Almost 59% of survey respondents said their teams deploy multiple times a day, once a day, or once every few days, a percentage nearly identical to last year's and one that likely tracks with the over 61% of survey takers who work at companies with 500 employees or less. All told, 28% deploy continuously (multiple times a day), while 15% deploy once a week, 10% once a month, and under 7% once every few months.

Not surprisingly, the vast majority of survey respondents participate in open source projects – over 69% this year up from 63% last year. More than 29% said they contributed to Gitlab, while 14% are involved with Kubernetes, and 13% with VS Code. Nearly 19%, though, said they are involved in "other" projects, many of them smaller and lesser-known (a trend we saw last year as well).

"Testing delays everything."

"Many people find it a chore to review code."

"Operation failures due to rushed or ignored planning."

Testing remains tough

For the third year in a row, a majority of survey takers resoundingly pointed to testing as the area most likely to cause delays. The other bottlenecks include planning, code development, and code review, again reflecting what we've seen in our 2019 and 2020 surveys.

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

"Covid** (mentioned a number of times)."

"Testing can be both slow in writing and running."

"Communication between business and development is hard."

"Security is not integrated at all into the development process :(."

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

"Finding someone for code review can be hard (1 day average). After that business tests take time to be complete (2-4 days on average)."

"Testing is not yet fully automated in the deployment cycle; hoping to improve that with our move from BitBucket + Jenkins/drone to GitLab."

"A majority of our delays are created by multiple teams being involved, under different management umbrellas, without one cohesive vision at the top layer."

"We are currently using the same technology stack that was chosen five years ago, so it is not up to date with modern practices yet like TDD, version control, and hexagonal architecture."

"Developers are sometimes unaware they have to do code reviews. They aren't sure how to perform them and if they are effective. Sometimes they are skipped so the process can go through."

Threading the testing needle is a challenge, but there are some small signs of forward momentum. Almost 25% of teams report full test automation (more than double what was reported last year) and 28% of respondents say they're at least half-way there. Roughly 34% of survey takers said developers test some of their own code (up from 31% last year) and 32% said automated testing happens as code is written, a big jump from 25% in 2020.

But 25% of teams either have no test automation or are just starting to think about it, and 9% admit their teams haven't shifted testing far enough left.

Unsurprisingly, frustration with the lack of automated testing is clear:

> **"Automated testing is ignored 'due to time constraints.'"**
>
> **"Testing? That's an interesting idea."**
>
> **"We intended to do TDD but it usually ends up being after the fact."**
>
> **"I try to write my code with TDD when it's possible; it's complicated when writing React components, or when changing a function that is not tested with many side effects and many inputs and the tech lead forbids (me) to refactor it at the moment .... ='(."**

The strongest light at the end of the testing tunnel may be found in the use of artificial intelligence/machine learning. In 2020, just 16% of survey respondents said they had "bots" testing their code or an AI/ML tool in place for test; this year the percentage was just over 41%. All told, 25% of respondents use bots to test their code, 16% use AI/ML to review code before a human sees it, and 34% are exploring the idea of AI/Ml but haven't done anything about it yet. Exactly one-quarter of respondents aren't using AI/ML in test.

Where the tools rank

Almost 85% of survey takers use Git for source control (down from 92% last year), while almost 4% use Team Foundation Server, and 2% use CVS. Just 5% of respondents said they don't use any source control.

GitLab is the tool of choice for CI/builds (34%), followed by Jenkins (21%), GitHub Actions (14%), and BitBucket (8%).

Just over 37% of survey takers said they "partially" use microservices, while 34% fully use them (up from 26% last year), and 28% don't use them at all. Some respondents said they were planning to or are investigating microservices, while one said, "We're planning to move to them in the next year or two."

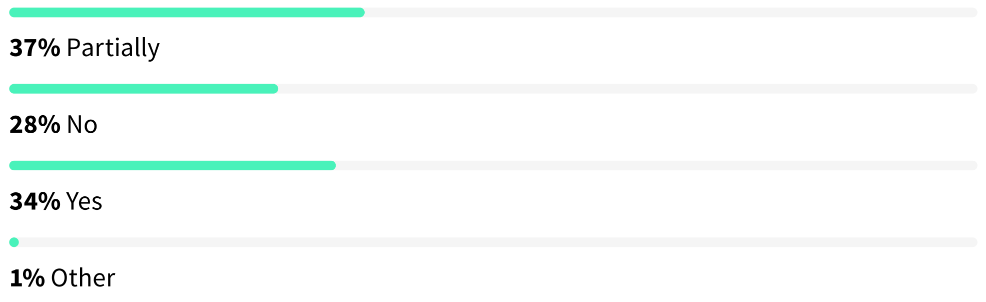
But when it comes to Kubernetes, it's definitely a "What a difference a year makes" situation: In 2020, only 38% of our survey takers used K8s, as it's known. This year, 46% use Kubernetes, while 37% do not (down from 50% last year).

DevOps teams not yet on board are much closer to actually implementing Kubernetes this year than last year:

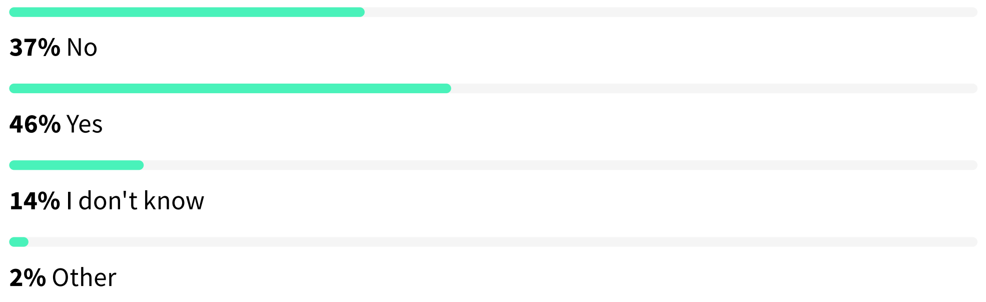
> **"Not yet, but this is very much in-plan for our desired approach."**
>
> **"Planned for 2021."**
>
> **"Yes, but not for every workload. ECS, serverless are also used."**
>
> **"We've tried but now it's just not needed for small production environments."**
>
> **"We are moving to K8s this year. Woot."**

Low code/no code development tools are also being taken more seriously this year. Last year 75% of respondents told us they don’t use them; this year, 41% use low code/no tools, while 59% do not.

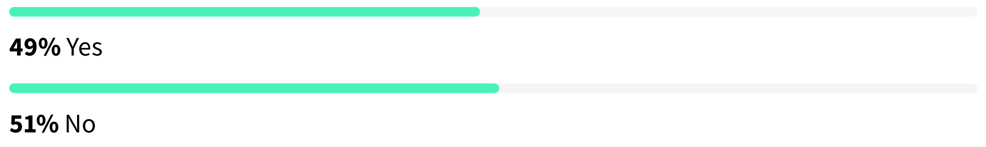
HAS YOUR ORGANIZATION ADOPTED MICROSERVICES?



DOES YOUR ORGANIZATION USE KUBERNETES?



DOES YOUR ORGANIZATION USE A LOW CODE OR NO CODE TOOL?



The role of the DevOps platform

This year we asked survey respondents for the first time about DevOps platform usage. Slightly over 70% said their teams use a DevOps platform (but we left it up to survey takers to actually define what a DevOps platform means to them). The top benefits of a DevOps platform? Better DevOps, improved collaboration, easier automation, and visibility/traceability were named as the biggest benefits. Respondents also offered other ways a DevOps platform was helping their teams:

> **“More ownership of everything to do with the product.”**

> **“Reduced mean time to recovery (MTTR), quicker time to market, reduced lead time for fixes, and fewer change failures.”**

> **“Reliability, repeatability, consistency, productivity.”**

Not surprisingly, the group most likely to use a DevOps platform is the DevOps team (43%), but 23% said “everyone” in their company uses the platform.

Developers

Development top findings

DevOps = faster releases

Almost 60% of devs are releasing code 2x faster, thanks to DevOps.

Investing in the process

DevOps teams didn't see small incremental tech changes in 2021 - they added the big guns: SCM, CI/CD, a DevOps platform, and automated testing.

What's missing?

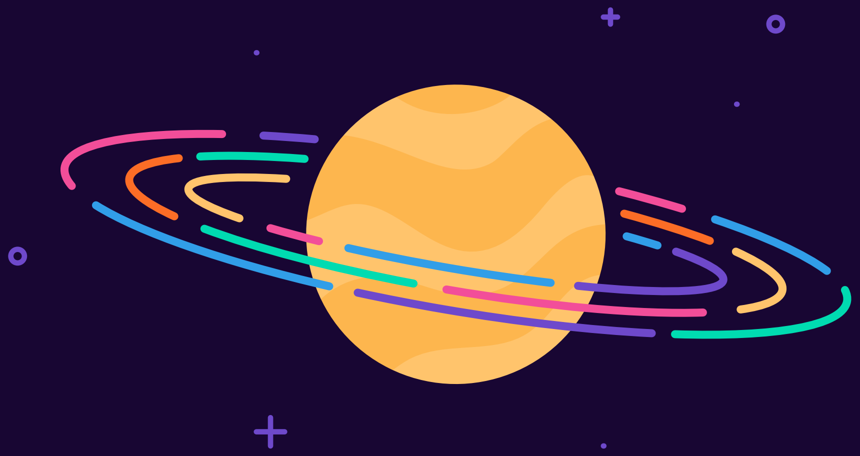
More testing of all types and more (and different) code reviews.

Role changing continues

Devs continue to take on jobs that used to be the sole responsibility of ops.

Future facing

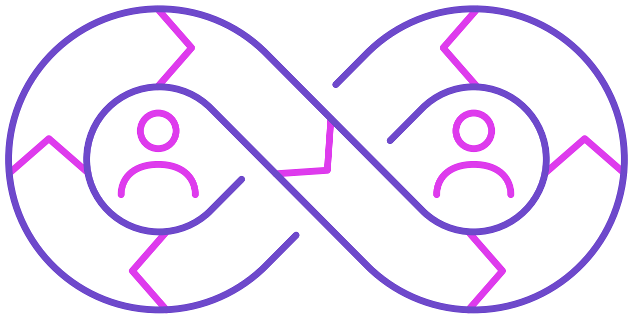
A full 30% of devs think an understanding of AI/ML will be critical to the next step in their career.



Devs and DevOps

If you feel the need for speed (when it comes to code release), DevOps continues to be the right choice. Just over 84% of devs told us they're releasing code faster than before. About 57% said code is being released twice as fast (a big jump from last year's 35%), and 19% said code goes out the door 10x faster.

Why is code being released more quickly? We asked developers what's changed in their process. Just over 21% of survey respondents said they've added source code management to their DevOps practice (up from 15% last year), while almost 18% added CI and 13% added CD. Nearly 12% said adding a DevOps platform has sped up the process, while just over 10% have added automated testing.



WHAT CHANGES HAVE YOU MADE TO YOUR SOFTWARE DEVELOPMENT PROCESS?

21.02%	Source code management
17.74%	Continuous integration
13.59%	Continuous delivery
11.65%	DevOps platform
10.38%	Automated testing
5.29%	Toolchain integration
4.12%	Planning tools/methodology
3.95%	Cloud Native
3.52%	Severless
2.89%	Requirements Management
2.7%	Automated security testing/shift left security
2.38%	Release Orchestration
0.6%	Other

DevOps teams are adding new processes, but there are also some mindset shifts going on. We asked developers to take a deep dive into what's really made it possible for them to release code faster.

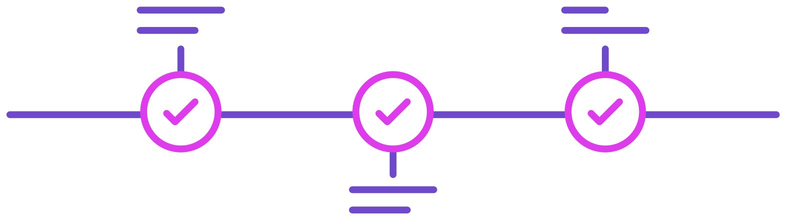
Many said the process was a holistic one:

> "By capturing more errors and bad practices in the development phase we are able to deploy more often and securely to production."
>
> "Our team adopted microservices on a new project and then fully embraced continuous delivery. To get to continuous delivery, we need to assure quality, so we have automated tests built-in. Investing in these areas allowed our team to deploy 2000 times to production over a year, where in the past we would deploy maybe 6 times."

"We are releasing code globally instead of into specific locations with automated deployments. Principally, cutting commit-to-live time (by removing batching) encouraged smaller changes incurring less overhead (due to removing a coping strategy of increasing scope)."

"We divide and conquer: Splitting the code into more modules has helped decrease debug time, improved stability, and allow a mix and match approach."

> "We changed our release plans so versions developed concurrently depend less on each other."
>
> "We evaluated the team and did value stream mapping and finalized the desired state. In most of the cases we found the team needs an automated pipeline for faster delivery and immediate feedback so that they can act fast rather than later. We also moved security left so that developers can fix security issues fast. We also made sure developers are doing code review in a collaborative way though pull requests."



But automation was also critical:

> “We have full automation from dev to production.”
>
> “I use automatic semantic versioning to speed up releases.”
>
> “We automate everything possible, to be able to test our product ‘like in real life’ without any downside. This increases confidence and simplifies tests for everything.”
>
> “We got rid of manual deployments.”
>
> “Massively expanding our CI capabilities has decreased deploy time significantly.”
>
> “Integration testing has been a big plus in how confident we are to release automatically and deliver a version. We are now able to deliver any day.”
>
> “It helps that devs don't need to keep track of test running; they just need to push and pipeline will check there code before merge to master.”

“To be honest, continuous delivery helped us the most to speed up our delivery with our clients.”

“We automated the building process, code linting, and testing of key parts of our project. We put all of these into

“We are not relying on developers to have remembered to create and run tests for their code before deploying.”

“Automation in every possible way.”

All that automation has translated into a huge list of things devs told us they no longer have to do, including:

"Having to manually create backups and do manual testing and deployment."

"I don't have to keep written cheat sheets."

"Testing of back-end code by hand."

"Security and dependency management."

"Raw coding."

"Run lint. (It's auto run when using git commit.)"

"Building containers."

"Relying on code review to have caught all the test scenarios. We now use a coverage scanning tool to tell us if we've got it all."

"Waiting."

"Extended periods of indecision."

"Tests on local environments are no longer required."

"Tedious security red tape reviews, weeks setting up a server, delaying for change management windows."

"The dev teams no longer perform production deployments."

"No more blocking on every dev push."

"Communicate on individual changes and instead use GitLab as a platform to collaborate and communicate more effectively."

What aren't devs working on that they would like to be? It's a long list, starting with way more testing and code review:

Shift left security

Dynamic testing, e.g., RAM, memory leaks, CPU

More code review (x100)

More testing, more automation on commits

Microservices

More open source

Performance optimizations and accessibility

More static analysis earlier to shorten the feedback loop ahead of pull requests

More automated tests. Code coverage analysis. Intelligent test subset selection, so branches fail faster

Reusing code/logic already deployed for new projects

Switch to a GitOps operation model

Better planning, writing more detailed requirements, involving stakeholders more

Have improved approval processes with stakeholders

AI testing

DevSecOps platform

Integrate AI/ML for writing code

TDD, BDD, testing against mocks, CI/CD

Each time a bug is fixed we should add a regression test systematically

More machine learning and pair programming

Tackle very old legacy code

"Think twice, code once"

Developer daily life

In a trend that we saw starting in 2020, developer roles continue to shift, taking on more responsibility for what were traditionally ops roles. Nearly 26% said they instrument the code they've written for production monitoring (up from just 18% last year), while 38% define and/or create the infrastructure their app runs on. About 13% monitor and respond to that infrastructure.

Nearly 45% of survey respondents said they review code weekly, and 22% do it bi-weekly (up from 14% last year). But anecdotally, some developers tell a different tale about code review on their teams, ranging from not doing it at all, to conducting code reviews on every single merge request/ticket/pull. Many told us they review code daily, or even multiple times a day. Not surprisingly, nearly 60% of developers said code reviews were "very valuable" when it comes to security and code quality. Code reviews are most likely to be done via an online "chat" service and devs said they far preferred to review code in an IDE vs. a browser.

Devs are spending time on code review, but a majority of them aren't spending much time on toolchain integration and maintenance – 41% said they spend less than 10% of their time doing so monthly, while 20% said they spend between 11% and 20% of their time on those tasks.

Who sets dev's priorities? This year 43% of devs said they set their own priorities (a big change from 2020 when only 24% said this), while 38% said product managers, and 19% said the business side. When prioritizing work and features, cost of development is the most important priority to developers (43%), followed by developer workload (34%), and product roadmap (31%).

Security

A full 39% of developers feel fully responsible for security in their organizations (up from 28% last year), while 32% said they shared the burden with other teams. And the optimism about security improving is reflected by developers too: 75% said they feel their organizations make it possible for them to avoid breaches.

How do developers describe their teams efforts to keep things secure?

> **“AI”**
>
> **“Security red and blue teams auditing running applications, using quality tools and checking dependencies.”**
>
> **“Through code reviews and automated testing.”**
>
> **“Risk analysis in the design phase and (you can) avoid the problems with security issues.”**
>
> **“Mostly through vulnerability scanning of libraries, container images and hosts.”**

“It’s all up to the developer!”

“Code review; monitoring and implementing best practices for security; frequent backups; strong passwords.”

Looking to the future

In a striking change from 2020, 30% of developers told us an understanding of AI/ML is the most important skill for their future careers; last year it was 22% and was second to soft skills. Soft skills, like communication and collaboration, are still important and they, along with cutting edge programming languages, were both cited by 18% of respondents, followed by GitOps at 14%, and IoT/blockchain at 11%.

Anecdotally, survey takers also said they wanted to know more about cloud/cloud native, cross-platform development, low code, data science, Python, and cryptography. But this quote sums it up well:

> **“I see substantial growth in developers as responsible for their platforms. This is the DevOps notion writ large and fast - there is substantial work to string the world of microservices together, for instance.”**

“We run static code analysis in our CI/CD pipeline. We should do more than that.”

“Some expensive tools are all we need.”

Security

Security top findings

DevSecOps is real

A full 72% of security pros rated their organizations’ security efforts as either “strong” or “good.” Wishful thinking it might be, but it’s a significant increase in optimism from a group not necessarily known to be upbeat.

The shift left continues

DevSecOps teams are running more DAST, SAST, container, and dependency scans than ever before.

Sec and dev are friendlier

But there is still confusion over who “owns” security and the finger-pointing game is strong.

Facing the future

Security pros feel they need subject matter expertise to excel in their future careers, but nearly the same percentage said soft skills would be most important. Also of interest: advanced programming and AI/ML

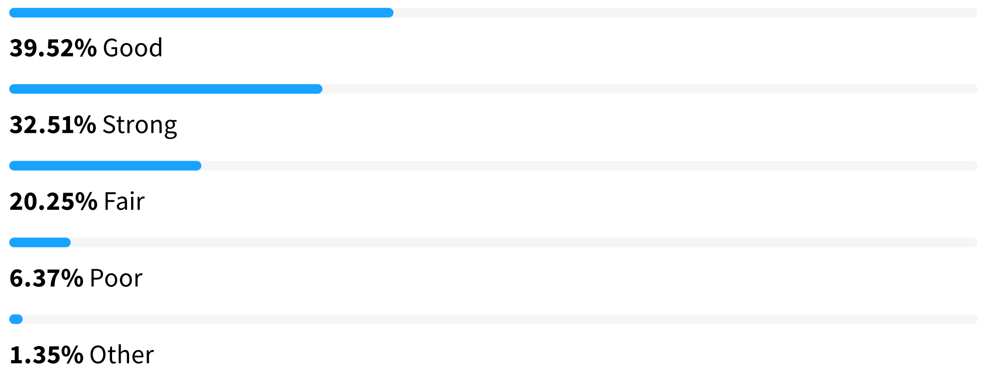
Security and DevSecOps

Could 2021 be the year that DevSecOps becomes a reality? Perhaps. An unprecedented 72% of security pros reported their organizations' security efforts were either "good" or "strong." That's a significant change from last year when only 59% said the same thing. The largest year over year increase was in the "strong" category – last year only 19.95% of respondents considered their security posture in that light compared to nearly 33% in 2021.

Roles are changing

Perhaps one reason security pros are so bullish about their teams' efforts is the fact that security roles continue to evolve. Nearly 28% said they are now part of a cross-functional team (identical to last year's results), while 26% are now more focused on compliance (up nearly 4% from 2020) and 24% are more involved in daily tasks/more hands on (down slightly from last year). About 20% said they didn't see any changes in their roles, a percentage nearly identical to last year's survey.

HOW WOULD YOU RATE YOUR ORGANIZATION'S SECURITY EFFORTS?



39.52% Good

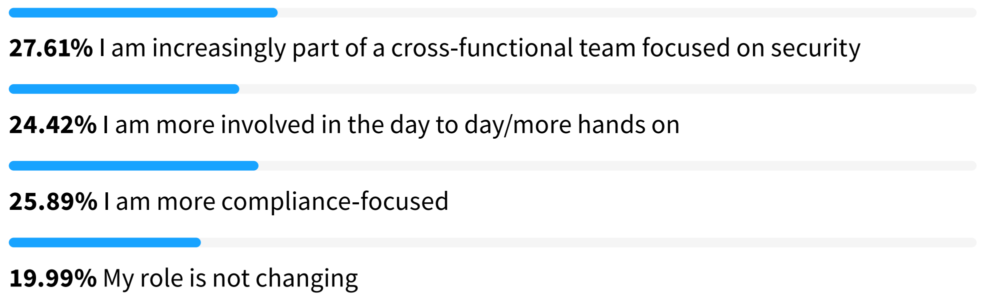
32.51% Strong

20.25% Fair

6.37% Poor

1.35% Other

IN YOUR EXPERIENCE HOW IS THE SECURITY ROLE CHANGING?



27.61% I am increasingly part of a cross-functional team focused on security

24.42% I am more involved in the day to day/more hands on

25.89% I am more compliance-focused

19.99% My role is not changing

"I think security, in most cases, is not a single person's specialization. Security must be a practice of every member of the team from the frontend developer to the system administrator (also non tech roles)."

Shifting left

Another positive sign: Security is also continuing to shift left and at a faster pace than we've seen before. Over 70% of security pros report their teams have shifted left (i.e., moved security earlier in the development process), up from 65% last year.

Dig in deeper though, and some curious dichotomies surface. Scanning has certainly increased: Today 53% of developers run SAST scans (a dramatic jump from last year's less than 40%) and 44% run DAST scans (up a lot from last year's 27%). And well over 50% of security pros report their devs scan containers, run dependency scans, and do license compliance checks.

But while there are more scans run, most results aren't easily available to developers. In fact, just 23% of teams put SAST lite scanners in a web IDE, and only 20% pull scan results into a web pipeline report for devs. DAST, dependency, and container scans fare worse: Only 16% make DAST and dependency scans easily available and 14% do the same for container scans. These results show the barest improvement over 2020; last year fewer than 19% of companies put SAST results in a report for devs and less than 14% did so for DAST.

For a security shift left to work, devs have to be able to get access to results while in their IDEs, so this remains a work-in-progress area for sure.

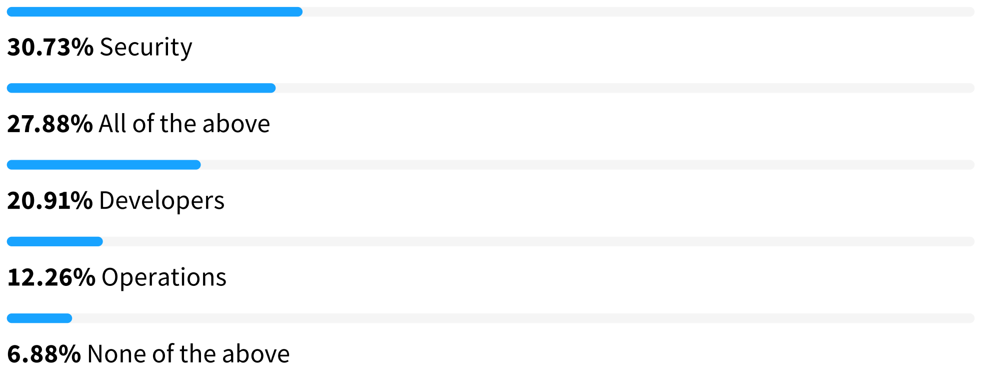
Another work-in-progress is the sometimes contentious relationship between security teams and developers. This year's survey showed the finger pointing remains in full force, but, surprisingly, at much lower percentages than we've seen in the past. Last year a whopping 93% of security pros said developers caught 25% or less of the available bugs to be found in existing code (meaning three-quarters of the bugs were left for sec to find later). This year, just 45% of security team members said the same thing and a previously unheard of 37% said devs actually find between one-quarter and one-half of all bugs.

And while 83% of security pros agreed at some level that finding bugs is a developer performance metric, nearly the same percentage (81%) complained it was difficult to get devs to make bug fixes a priority. In the end, 77% of security pros agreed at some level that bugs are mostly found by them (and not devs) after code is merged in a test environment.

Who's in charge?

The question of security "ownership" remains a tricky one in nearly every organization, and that's particularly true when it comes to the security team. Almost 31% told us they (security) were fully responsible for it, but almost 28% said everyone was responsible. That response was eerily similar to last year's, and underscores the need for clarity on this subject.

IN YOUR ORGANIZATION, WHICH GROUP IS PRIMARILY RESPONSIBLE FOR SECURITY?



About the bugs

Security testing remains a sticking point for team members. It's happening too late in the process (over 42%), and nearly the same percentage said it was a struggle to unpack, process, and fix vulnerabilities. Almost 37% said it was tough to track the status of the bug fixes, and 33% said it was hard to prioritize the remediations. Finally, 32% said it was difficult to just find someone to fix the problems.

When bugs are found, severity level is most important (59%), followed by category (51%), number of vulnerabilities solved (41%), and time elapsed since found (37%).

Modern application development strategies -- including microservices and containers -- are increasingly popular, but only roughly half of security pros reported having processes in place to monitor and protect them. Teams using monitoring tools most often mentioned Prometheus, Grafana, and AWS Watch.

> **"Developers have their own observability stack."**
>
> **"There really isn't [a process] in place."**
>
> **"Our flagship acts like a closed system that accepts very little user input and because it runs on top of App Engine the containers are practically self-securing."**

The security outlook is a bit brighter when it comes to cloud native and serverless, however. Last year 64% of respondents said their organizations had nothing in place to secure cloud native and serverless, but this year 53% of teams have built it in.

Looking to the future

When it comes to what will help them most in their future careers, security pros were nearly evenly split: 22.95% said subject matter expertise while 22.91% said soft skills like communication and collaboration. (In 2020, soft skills were the hands down winner; this year's change may be a reflection of pandemic-enforced collaboration improvements, leaving room for something else to focus on.) Almost 21% said advanced programming, while nearly the same percentage said AI/ML.

"Security knowledge, pen testing, bug bounties, a good knowledge of Linux, the technologies used and the runtime (containers, servers, etc.)."

> **"Cloud and serverless architecture."**
>
> **"Cloud skills."**
>
> **"Malware analysis and threat intelligence, penetration testing."**
>
> **"Automation and DevOps"**

Operations

Operations top findings

DevOps = change

Over 62% report new and different responsibilities because of DevOps.

In the cloud, and not

More than half of ops pros primarily manage cloud services, but almost the same percentage are firmly planted on the ground and focused on hardware and infrastructure.

Automation gets real

Almost 19% of ops teams report full automation, while 37% are “mostly” automated.

In the future...

Ops pros think programming will be the most important skill they can have, a big change from 2020’s focus on soft skills.

Operations

The struggle can be real in operations, placed as it is as gatekeeper to past, present, and future technologies and methodologies. Not surprisingly, roles continue to evolve rapidly in operations, and ops pros told us DevOps is the reason.

What do those changes look like?

"Everything from provisioning servers to managing people. Most of the stuff in between is building automation platforms to do the day-to-day work."

"I plan the company roadmap for software development, manage the entire developer team, and come up with R&D efforts."

"Maintain the tools of the DevOps toolchain in operational condition and continue to improve the platform and practices."

"I'm a DevOps coach."

"I'm a platform engineer."

"I'm a Jack of all trades...a lil bit of everything I can get myself into."

"DevOps, SRE monitor and make sure the platform works."

Today, 49% of ops pros see their role as primarily managing hardware and infrastructure (a big jump from last year's 42%), while 56% say their first priority is managing cloud services (a four point increase from last year).

They're also spending more time on compliance than they were in 2020 – last year over 55% said they spent very little time (10% or less) dealing with audit and compliance issues, but this year just 36% reported spending so little time. In fact, 29% now say they spend about one-quarter of their time dealing with audits and compliance issues.

Still so many tools

And it can't be operations without a lot of tools. Nearly 50% of ops pros said their teams use between two and five monitoring tools (down sharply from 65% last year), while 28% don't use any monitoring tools at all. All told, 72% of ops teams use tools that make it easy to feed developers real time data (and just over 40% of those said it was because their organization used a DevOps platform).

Logging (capturing and viewing application logs) is the most important monitoring category, followed by metrics.

The top choice for logging is Elasticsearch (28%) followed by Splunk (21%) and Datadog (19%). (Last year, Elasticsearch was used by 38% of ops teams, while 14% used Splunk and 12% chose Datadog.) When it comes to tracking application metrics, 21% use Datadog and nearly 14% use New Relic. Nearly 20%, however, don't use any tool to keep track of metrics. Prometheus is the tool of choice for capturing time-series metrics for the second year in a row: 30% of ops teams use it, followed by Datadog at nearly 23%.

WHICH TOOLS DO YOU USE TO VIEW/CAPTURE LOGS?

Elasticsearch (Elk Stack)	**37.51%** in 2020	**28%** in 2021
Other	**7.52%** in 2020	**NA** in 2021
Splunk	**13.64%** in 2020	**20.58%** in 2021
Datadog	**11.71%** in 2020	**19.3%** in 2021
None	**12.44%** in 2020	**11.92%** in 2021
Sumologic	**2.95%** in 2020	**6.83%** in 2021
Logz.io	**3.5%** in 2020	**5.99%** in 2021

A majority of ops teams (roughly 34%) use the AWS cloud, while 24% use Microsoft Azure, and 23% use Google Cloud Platform. Just around 13% said their organization either doesn't use a public cloud or they don't know which public cloud is in use. Azure saw the biggest change from 2020: Last year not quite 18% provisioned the Microsoft cloud service.

WHICH TOOLS DO YOU USE FOR APP METRICS (TRACES)?

DataDog	**14.51%** in 2020	**21.02%** in 2021
None	**29.12%** in 2020	**19.46%** in 2021
New Relic	**13.68%** in 2020	**13.79%** in 2021
Jaeger	**11.73%** in 2020	**11.6%** in 2021
Zipkin	**4.94%** in 2020	**10.67%** in 2021
AppDynamics	**6.38%** in 2020	**10.15%** in 2021
Dynatrace	**5.56%** in 2020	**7.86%** in 2021
Other	**14.09%** in 2020	**5.46%** in 2021

WHICH TOOLS DO YOU USE TO CAPTURE TIME-SERIES METRICS?

Prometheus	**35.66%** in 2020	**30.06%** in 2021
Datadog	**12.18%** in 2020	**22.98%** in 2021
Nagios	**12.57%** in 2020	**12.38%** in 2021
None	**16.21%** in 2020	**12.17%** in 2021
Solarwinds	**5.4%** in 2020	**8.97%** in 2021
Sensu	**3.44%** in 2020	**8.76%** in 2021
Other	**14.54%** in 2020	**4.67%** in 2021

Working with development

Over 55% of ops teams told us their software development lifecycle was either completely or mostly automated. Just over 27% said it was partially automated and 11% said it was just beginning. Almost 6% said there was no automation at all. In 2020, just 8% of teams claimed full automation – nearly 19% said it this year.

Some things remain steady in the ops/dev relationship, including the fact that nearly 71% of ops pros said it was very or extremely important to them to have visibility into development, while 79% agreed at some level they get enough notice to support dev projects. But in a continuing sign of shifting roles, nearly 77% of ops pros said their devs are able to provision testing environments, which is an 8% increase from last year. And there is more actual DevSecOps happening: Just over 76% of ops teams agree at some level that devs are able to receive and address security issues during the development process (that's a 10% jump from last year).

Ops pros feel increasingly responsible for security in their organizations: 28% said they were solely responsible (up from 21% last year) but the majority (34%) believe they are responsible but as part of a bigger team.

"Integration of diverse platforms or tools"

"Scripting, basic hardware and networking, and cloud basics"

Looking to the future

The impact of a global pandemic and the challenges of remote work were clearly felt in every DevOps role, and ops is no exception. Last year, like their counterparts in dev, security, and test, a slim majority of ops pros (31%) said they believed soft skills like communication and collaboration would be most important to their future careers. This year, 25% said programming would be most important, followed by soft skills (24%), subject matter expertise (21%), and AI/ML (20%).

They had other thoughts about the future, including:

"System administration skills are still a big plus"

"Decentralization, ethical architecture and security"

"Can't say the world is really changing, all this cloud stuff is ok, but if you know the basics about computer networking, OSes - welcome on board! Hard skills are essential of course, but communication is a real problem sometimes."

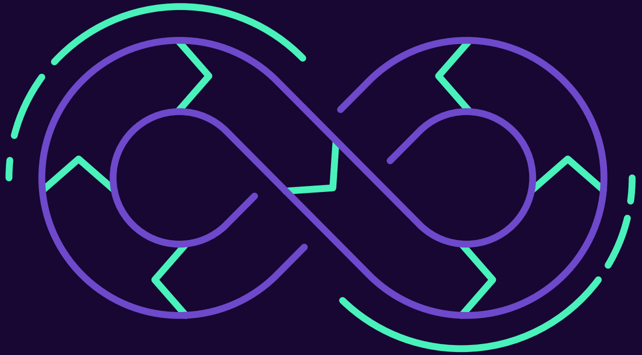
Keeping the DevOps momentum

The “gaining momentum” state of DevOps came through loud and clear from survey respondents when asked about their investment plans for 2021. Last year, DevOps teams were going to focus on the basics: automation, CI/CD and DevOps, and apparently it worked, because this year’s priorities show striking differences.

In 2021 the majority of survey takers will focus their investments on the cloud followed by AI. To put that in comparison, cloud was the 4th place pick last year and AI was in a distant 8th place. Automation and DevOps were the third and fourth place picks this year. Anecdotally, however, survey takers also expressed a lot of interest in machine learning and data science.

It’s clear DevOps teams are doing the real (and hard) work required to move forward. And to end on a good note, it seems the majority are ready for what’s ahead. Fully 48% of survey takers said they feel “somewhat prepared” for the future, while 27% said they were “well prepared.” Just 6% said they feel overwhelmed, a relatively small number considering the events of the last year.

So go forth and share these observations and results with your team, and see how your efforts compare. DevOps is very much a journey and not a destination; the trick is to keep the momentum going.



GitLab