
snyk
SNYK RESEARCH REPORT
Infrastructure as Code Security Insights
January 2021

Introduction

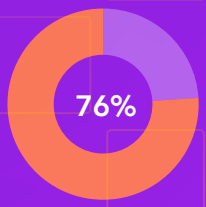
Cloud native applications are more than just the code developers create - today's applications include infrastructure as code (IaC) that dictate how the applications are setup on cloud infrastructure and how containerized applications will run on Kubernetes. The use of IaC allows for faster, repeatable deployments, but its usage also increases the burden on developers to secure not only their code, but also the infrastructure configuration, in addition to code dependencies and containers.

In this survey, Snyk sought to take stock of how IaC is being deployed by companies both large and small. Feedback from a wide range of roles in these companies went into our outlook on the state of IaC, highlighting the value of IaC, but also roadblocks to its widespread use and what we can do to overcome them. While our survey shows that many organizations have not coalesced on one "right" way to use IaC or who should be responsible for writing and maintaining it, we did find that respondents who are taking advantage of automated security testing for their IaC definitions are finding and fixing misconfigurations faster than their peers. The high performers in our survey are finding and fixing issues in their IaC definitions within a single day; whereas the lower performers take more than a week to realize there is a security issue and then then up to 2 more days to fix it.

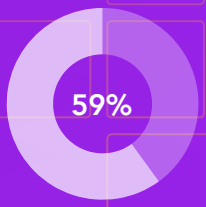
KEY TAKE AWYAY

Respondents performing automated security testing as part of their release pipelines where faster to find and fix vulnerabilities

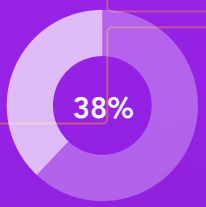
Can you fix an issue in under 1 day?



Teams with fully automated security checks



Teams with no or only partially automated security checks



Teams that only check security after deployment

Current IaC Practices

The benefits to speed and reliability when everything is in code and automated can be immense. But the benefits do come with a cost, namely an increasing burden on developers to secure not just their own code but it's dependencies, containers, and now, the infrastructure configuration. To start our research, we first explored what companies were taking on the challenge of implementing IaC and which tools they're currently using as they develop best practices.

We found that many companies are only starting out on their IaC journey, with 63% just beginning to explore the technology and only 7% stating they've implemented IaC to the best of current industry capabilities. While there are many tools either in use or being considered, 71% would prefer to standardize on a common toolset / workflow across all IaC configuration types and formats.

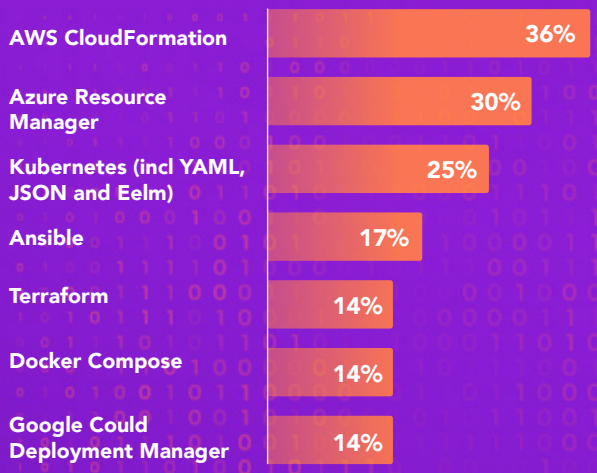
63%

of companies are just starting out

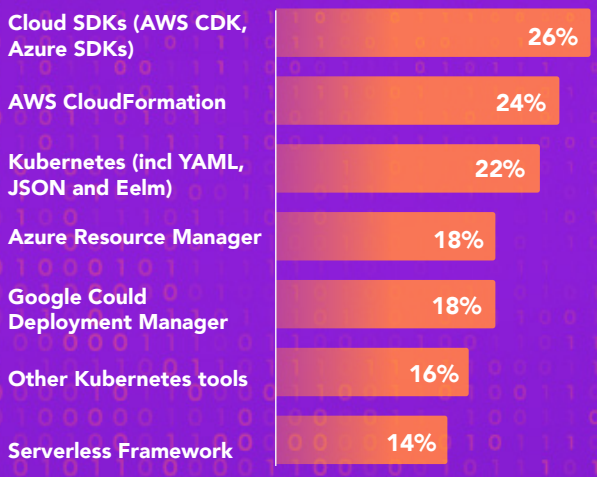
7%

of companies are implementing the best of current industry capabilities

In use today



Considering for the future



The opportunity is still wide open for most organizations to lay a firm foundation and implement the right tools and practices before widely adopting IaC.

We looked at how three different clusters of respondents to our survey fared when it comes to finding and fixing configuration issues that arise from using infrastructure as code:

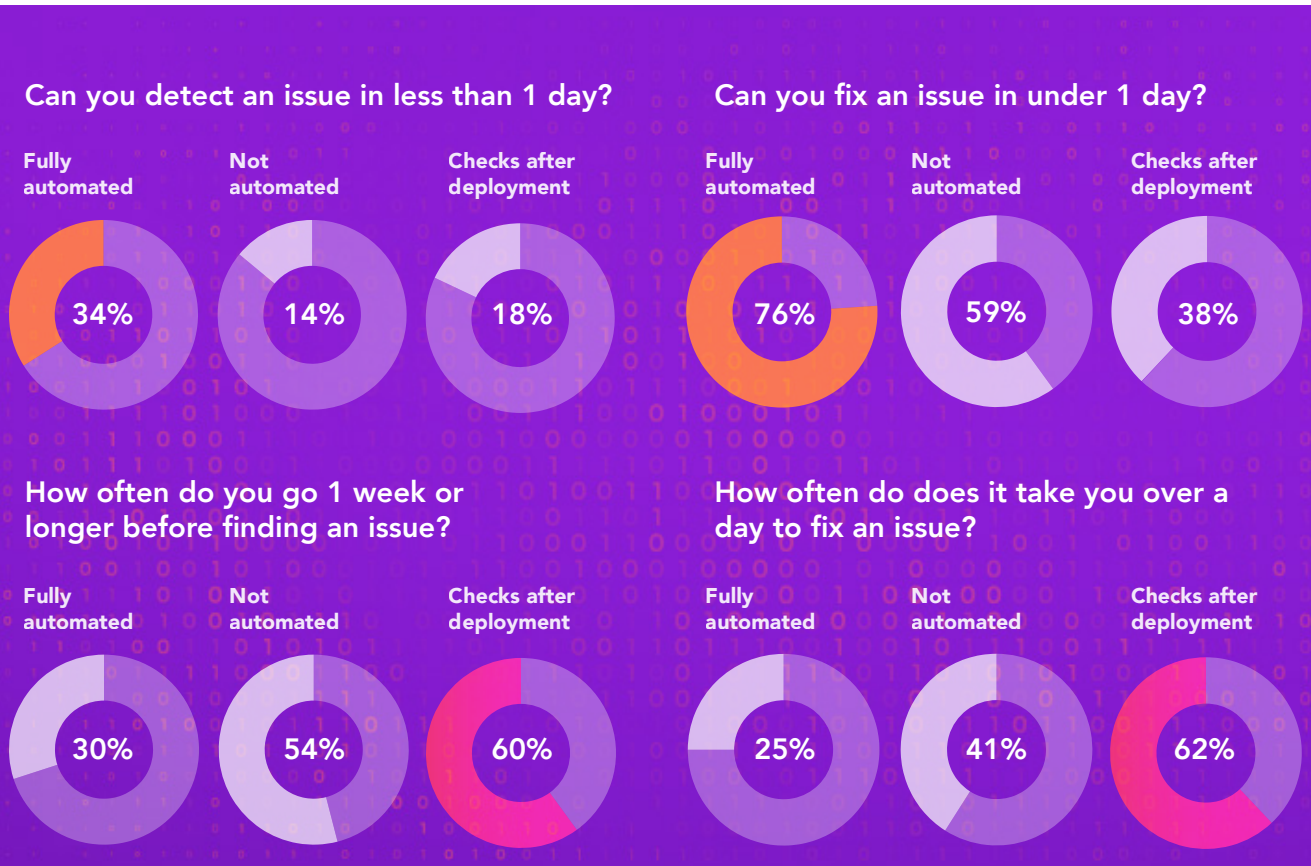
Full automation:
These respondents said they always performed automated security testing as part of their release pipelines.

Less than full automation:
This cluster includes respondents who have no automation up to those who have partial automation of security checks.

Only post-deployment checks:
This cluster may use some automation, but they only perform checks after infrastructure is deployed, either via audit tools, pen testing, or investigating security incidents

It may come as no surprise that the fully automated group outperformed both of the other groups at both discovering and fixing issues. When it comes to finding issues, the high performers were able to discover issues in less than a day roughly twice as often as respondents in the other two groups. And the fully automated cluster was able to fix issues quickly, in less than half a day, over 60% more often than either of the other clusters.

There were differences at the other end of the responses, too. The two lower performing clusters took 1 week or more to discover IaC issues in over half their cases, where the fully automated cluster only took that long 30% of the time. Fixing the issues is where the cluster that only runs post-deployment checks really suffered. They were only able to fix these IaC issues in less than a day half as often as the fully automated respondents, and in 62% of the cases it took longer than a day to implement the fix.



Does Speed Equate to Safety?

Currently, modern applications deploy automatically on infrastructure created and configured via code. As a result, security so often takes a back seat to a speedy deployment, meaning configuration issues are not uncovered until after these applications have been deployed. Even Gartner states, "By 2025, 70% of attacks against containers will be from known vulnerabilities and misconfigurations that could have been remediated."

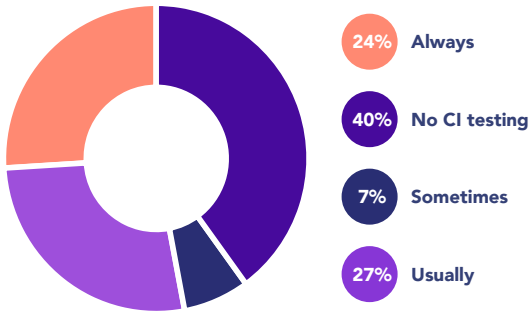
Yet, all this does not necessarily mean speed is inherently risky when it comes to IaC. In fact, the automated testing and release gates that are in place for other forms of code can be used with IaC and help make security best practices part of the development and release process. The highest performers in this survey - those who are both finding and fixing configuration issues fastest - are already doing exactly that.

By 2025

70%

of attacks against containers will be from known vulnerabilities and misconfigurations that could have been remediated.

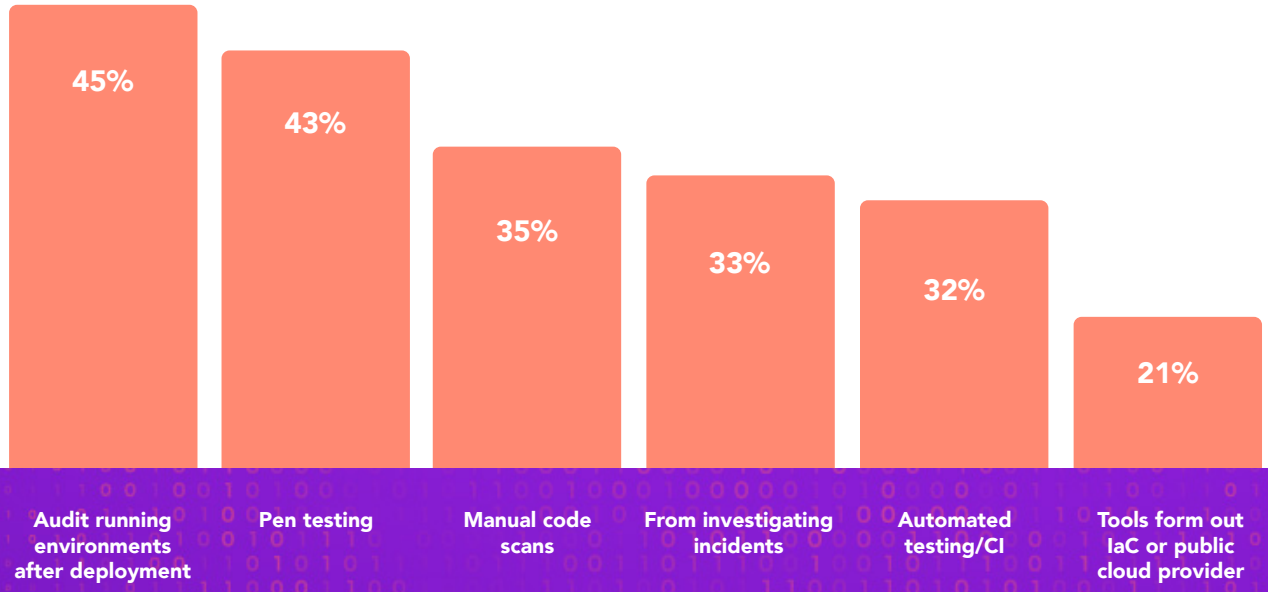
Do you include IaC security tests in your IC pipeline



Current IaC Security Practices

While the highest performers are finding and fixing security issues as part of their release pipelines, this type of automated testing is still nascent when it comes to security testing. Of those surveyed, 60% said their current workflow for IaC and configuration code does go through continuous integration (CI) testing, but security checks are not always part of those tests. Only 32% of respondents include security checks in their pipelines. In fact, most security issues are still being discovered after deployment, through pen testing, audits, and investigating security incidents. For those who are *only* using these post-deployment checks, it takes a week or more to discover a security issue in half the cases and over a day to fix those issues in nearly 2/3 of the cases. All in all, that's potentially 9 days of running with a security vulnerability versus less than one day for the highest performers.

How do you find out about security issues in your configurations and IaC?



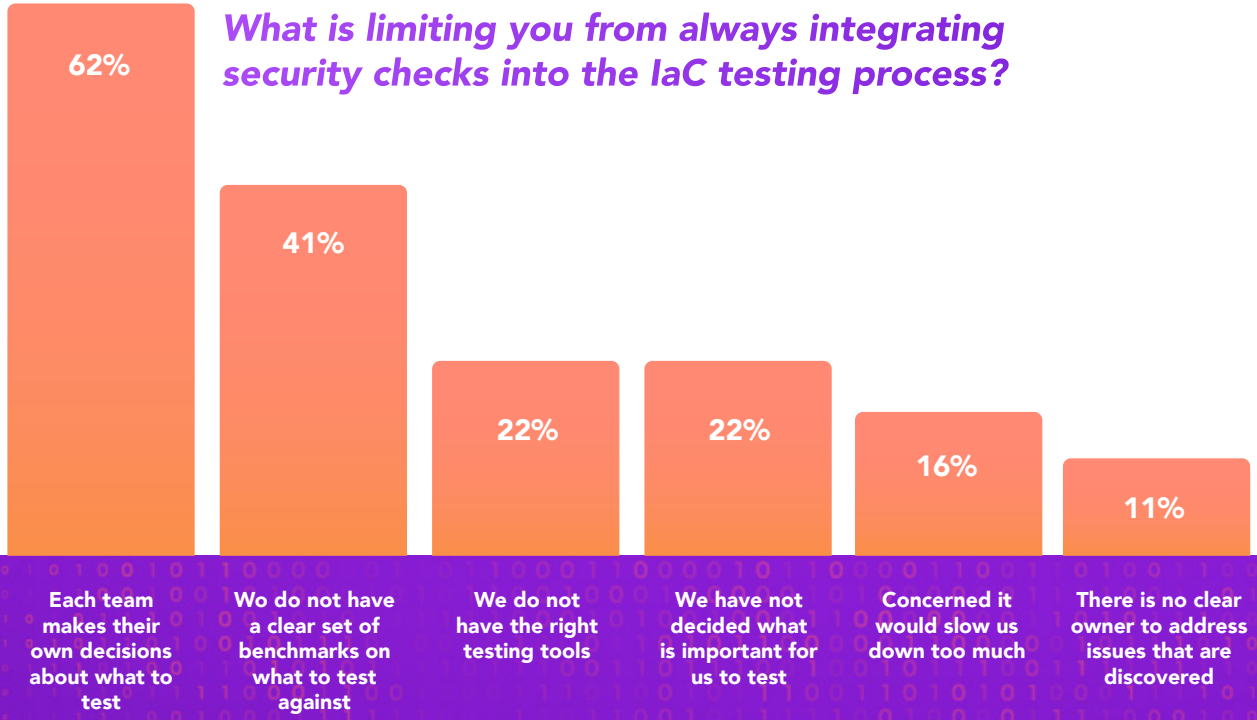
So what is standing in the way of making a change?

For those who said their IaC and configuration code goes through CI testing, the biggest barrier to integrating security checks is a lack of standardized best practices on what to check, with each of their separate teams making their own decision about what to test. When you couple that with the 41% who said their barrier was unclear benchmarks for security, the shortest path to improved IaC security can be paved with better tools that offer clearer guidance, while still providing teams with the freedom to determine what's most important for their needs.

41%

said their barrier was unclear benchmarks for security

What is limiting you from always integrating security checks into the IaC testing process?



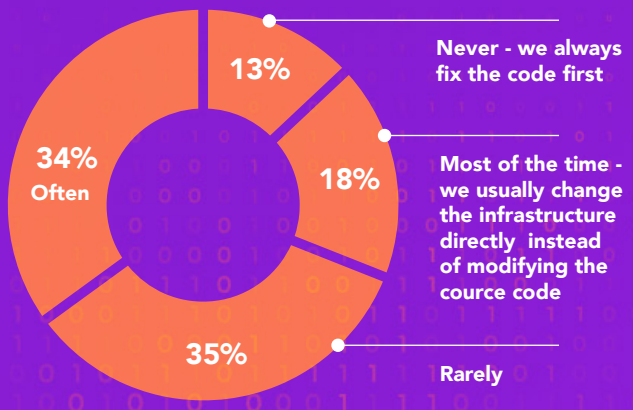
Infrastructure Remediation

Finding the issue is just one part of the issue - once an issue is discovered somebody has to fix it. When faced with a choice, 52% of respondents claim they usually remediate a security issue by directly tweaking the infrastructure instead of addressing it by modifying the IaC source code. This opens up the possibility for a number of issues in the long-term because the infrastructure and the codified definitions used to create it will start to drift; either that or the modified infrastructure will be reset to its misconfigured state on the next deployment. For those that choose this manual remediation path, their reasoning is split between a lack of standardization, knowledge, and communication, along with a desire to speed up the fixes as much as possible.

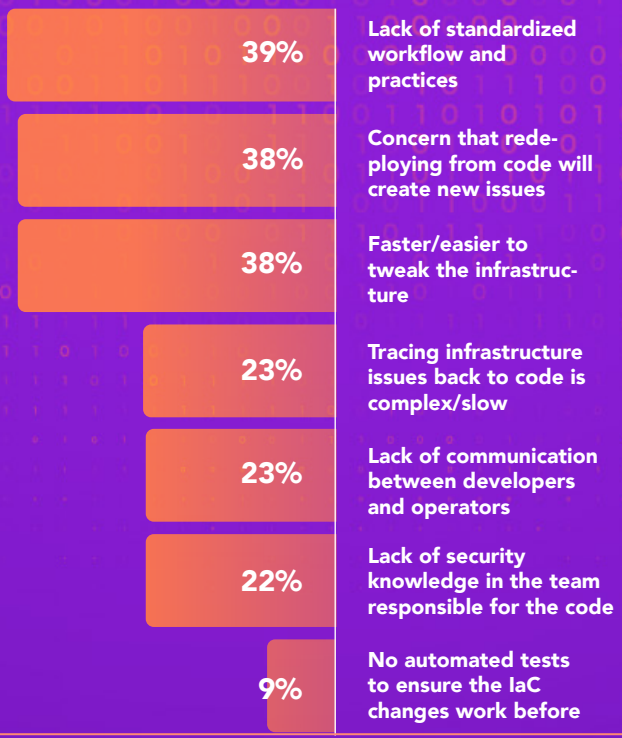
52%

remediate a security issue by directly tweaking the infrastructure instead of addressing it by modifying the IaC source code

How often do you directly modify infrastructure, rather than fixing the configuration in your IaC code?



Why do you directly modify the infrastructure instead of fixing the code?



While a lack of standardized workflow and practices was the leading reason respondents chose to remediate a security issue manually, a total of 61% of respondents also pointed to speed-related issues. Namely that it's faster and/or easier to tweak the infrastructure because tracking issues back to the IaC definitions is too complex and/or slow. Again, this points to two underlying issues:

First, when security checks are only performed **after** infrastructure has been deployed, it's too late in the process. It separates the security checks from the code and it's also likely that pen test reports and audit tools don't provide data that's directly actionable by the owners of the IaC code.

Second, for teams that are stretched thin, a lack of bandwidth could lead to the decision (consciously or not) to overlook some security in favor of speed. For those teams that are at their limit, the right tools can make a world of difference to equip developers with what they need to prioritize security. But before a tool can be decided on, teams must first determine who holds final responsibility for IaC.

61%

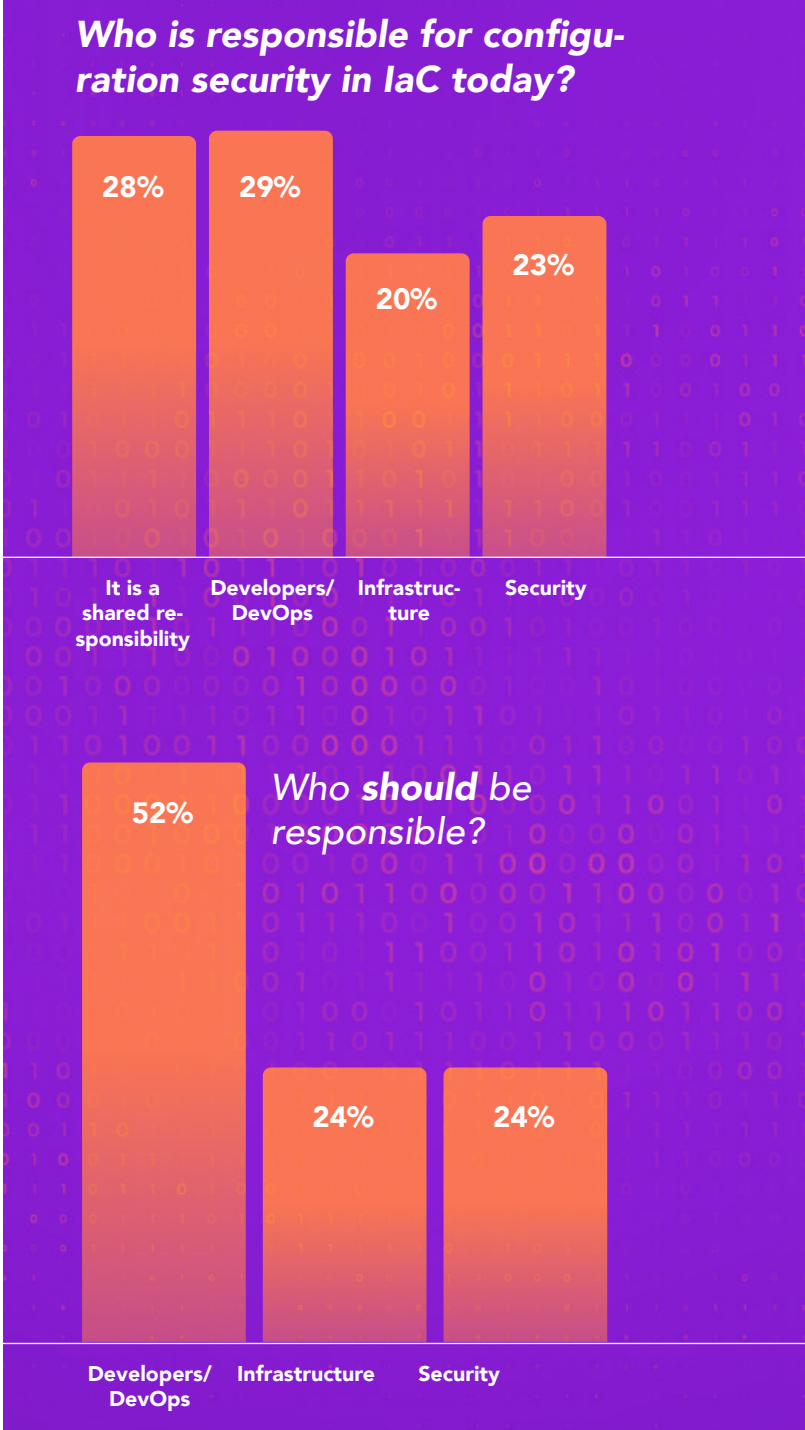
said speed-related issues were the reason they remediate a security issue manually

Who Holds Responsibility for IaC Security?

One of the barriers to shifting IaC security left was that teams struggled to standardize practices across their organization, leaving each team to audit IaC as they see fit. In addition to the obvious security issues this presents, it speaks to a larger disconnect on responsibility. A common theme of this survey is the difficulty to pin down security ownership when it comes to IaC - so where does the industry currently stand?

Today it seems there is no consensus on who is responsible for the security within IaC. Developers and DevOps roles have a slightly bigger role than other individual teams and a good number say it's a shared responsibility, potentially fitting in to the newer DevSecOps models. When asked which team **should** be responsible for IaC security, if it is not a shared responsibility, the answers shifted heavily to the developers / DevOps groups.

So what's stopping these security responsibilities from shifting further left? Mostly, it's confidence in the broader organization's ability to readily spot and fix issues in the code.



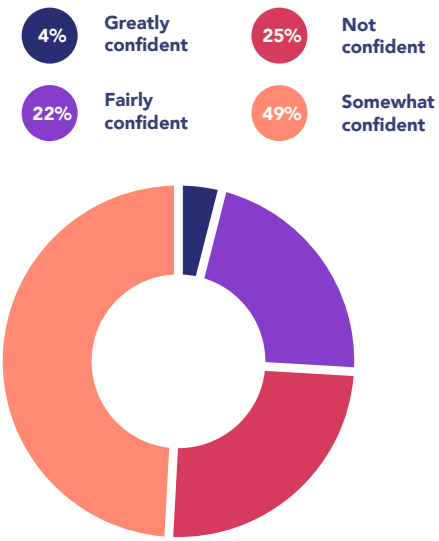
Snyk Infrastructure as Code: Find and fix configuration security issues the way cloud native experts do.

A clear and scalable solution is to invest in the tools and training needed to drive up confidence and help with bandwidth for these teams, allowing them to deploy code quickly and securely. In the same report cited above, Gartner also sees the potential for these automated tools and predicts that, by 2025, organizations will speed up their remediation of coding vulnerabilities by 30% with code suggestions applied from automated solutions, reducing time spent fixing bugs by 50%.

Snyk's long-standing developer-first approach led to the creation of Snyk Infrastructure as Code (Snyk IaC) to help solve these problems. This latest tool moves the security controls for infrastructure and configurations to the beginning of the development lifecycle, so developers can proactively determine whether their application and infrastructure specifications are safe. Designed to fit a developer's workflow, Snyk IaC helps pinpoint how to write secure Kubernetes and Terraform configurations, and even provides automated fixes as code in your choice of source code management systems. Together with Snyk Container and Snyk Open Source, you can finally embed your security expertise across your entire development organization.

To secure your organization and learn more about Synk IaC visit **snyk.io/product/infrastructure-as-code-security/.**

How confident are you in your ability to spot configuration issues in IaC?



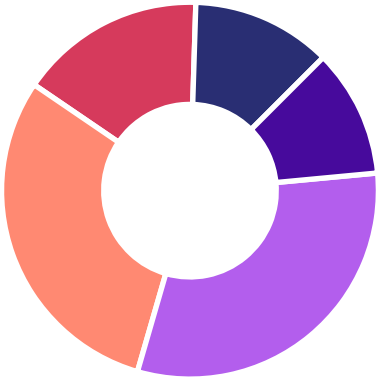
What would make you more confident in your organisation's ability to spot IaC misconfigurations?

Survey Methodology

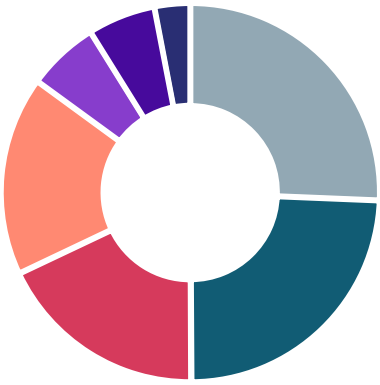
This vendor neutral research was independently conducted by Virtual Intelligence Briefing (ViB). ViB is an interactive on-line community focused on emerging through rapid growth stage technologies. ViB's community is comprised of more than 2.2M IT practitioners and decision makers who share their opinions by engaging in sophisticated surveys across multiple IT domains.

The survey methodology incorporated extensive quality control mechanisms at 3 levels: targeting, in-survey behavior, and post-survey analysis. The Calculated Margin of error at a 95% confidence level is 3.9%.

After receiving 543 responses from members of our opted-in 2M+ IT community, we screened out about 120 respondents who met the role, level and company size requirements, but who indicated they were not currently using, or considering using, the IaC / Configuration tools listed in the survey. This extensive process led to a survey pool of 481 qualified individuals in order to present the most accurate look at the current state of IaC.



- 11% Cloud & Platform
- 12% Architects
- 30% Developers and DecOps
- 16% Security & Compliance
- 31% Infrastructure



- 28% 1 - 500
- 23% 500 - 1000
- 14% 1000 - 2000
- 15% 2000 - 5000
- 8% 5000 - 10,000
- 8% 15,000+
- 4% 10,000 - 15,000


snyk
snyk.io