SANS

Analyst Program

Survey

SANS 2022 DevSecOps Survey: Creating a Culture to Significantly Improve Your Organization's Security Posture

Written by **Chris Edmundson** and **Kenneth G. Hartman**

September 2022

SYNOPSYS®

Executive Summary

The SANS 2022 DevSecOps survey examines the progress made over the past year toward improving organizations' security posture and operational effectiveness by aligning the development, security, and operations teams around secure DevOps cultural ideals, practices, and tools. Respondents representing a broad range of industries, job roles, and organization sizes participated.

The survey results indicate that, more than ever, applications are being hosted in multicloud, hybrid environments using virtual machines (VMs), containers, and serverless functions. Such environments present security challenges because of the inherent differences among the various cloud service providers and the very different demands of on-premises hosting.

The survey questions investigate topics such as the DevSecOps landscape, application hosting in the cloud, methods of securing multiple cloud environments at scale, container security, and to automation of compliance functions. We also look at DevSecOps practices and tools, along with challenges and success factors.

The final section, "Moving Forward," summarizes the key takeaways of each preceding section and advises organizations to continue to promote DevSecOps practices (such as conducting blameless retrospectives), to leverage technologies (such as Cloud Security Posture Management and Cloud Workload Protection Platforms) in order to cope with scale, and to monitor or experiment with new, trending technologies (such as artificial intelligence, data science, and GitOps) that show promise for improving DevSecOps.

Key Findings

- **Not all applications run in the cloud. This year, more companies have some applications on-premises, but overall, most companies have a smaller percentage of their total applications on-premises than last year.**
- **Cloud-hosted VMs are still favored over containers and serverless functions.**
- **There is a clear trend away from using a single cloud hosting provider to run most of an organization's workloads. The number of respondents that indicated their organization was using Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP) to run more than 75% of their application workloads fell noticeably, an average of 2.5 percentage points since 2021 (see Table 1).**
- **Although the majority of respondents use Cloud Security Posture Management (CSPM), only 17% use it in at least three-quarters or more of their AWS accounts, Azure subscriptions, or GCP projects.**
- **There was a 20-point year-over-year increase in the usage of Docker containers on-premises, but when containers are used in the cloud, the preference is for container services that use CSPM.**
- **Although the use of Continuous Integration/Continuous Deployment (CI/CD) tools that employ CSPM remained stable over the past two surveys, there was a significant increase in the usage of open source CI/CD tools at the expense of third-party, commercial CI/CD tools.**
- **Immutable infrastructure provisioning, blameless retrospectives, and chaos engineering remain the most underutilized practices within DevSecOps.**

A Snapshot of the Respondents

As seen in Figure 1, the 341 respondents were a geographically diverse group from organizations of all sizes, with a strong bias toward security roles.

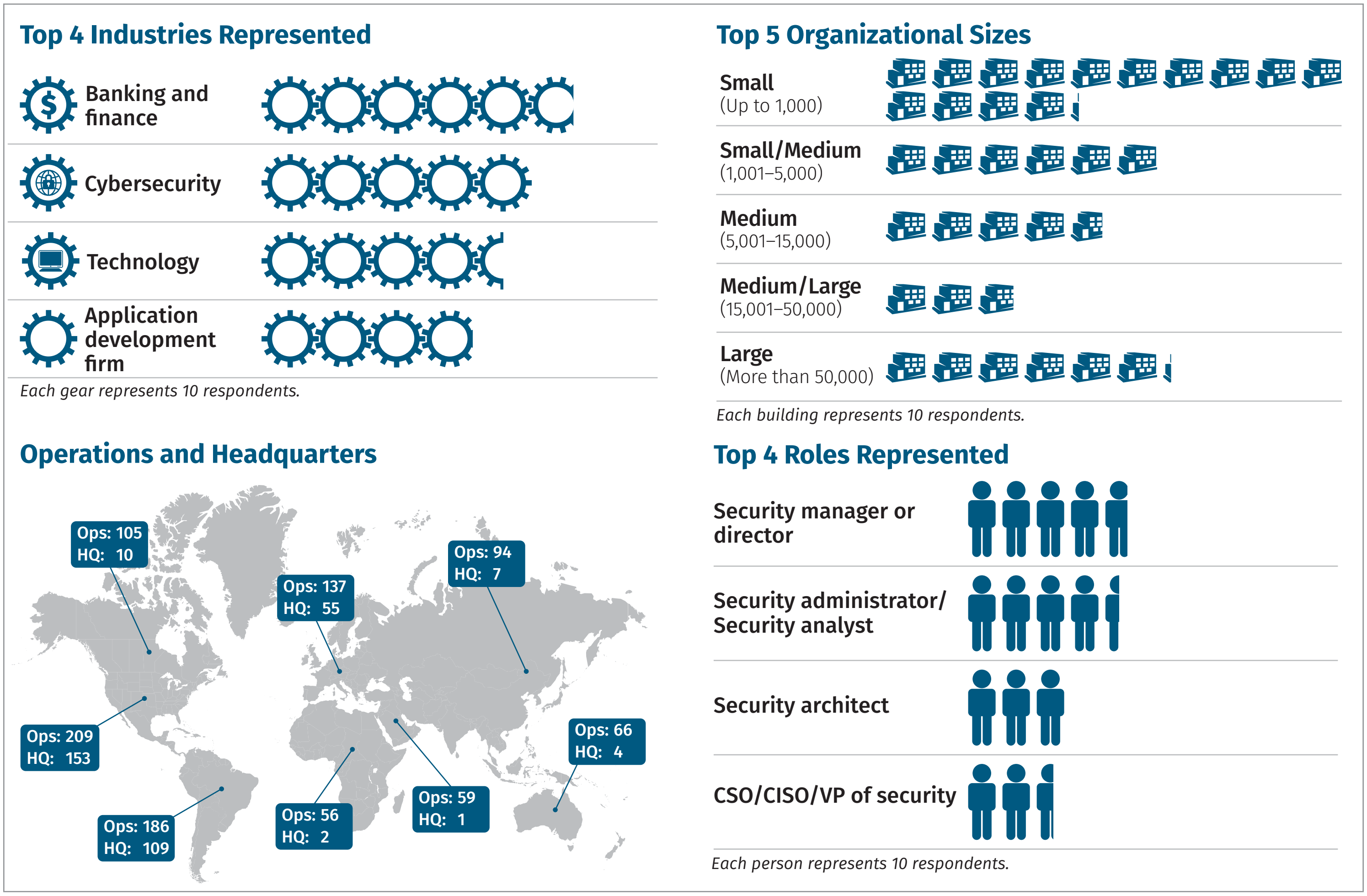

Figure 1. Demographics of Survey Respondents

More than half of the respondents were from the top four responding industries, with the next-largest sector being government, at 7.9%.

Small organizations represented more than 31% of all respondents, while the largest organizations took third place, and the other respondents were distributed relatively evenly in the other organization size categories.

At nearly 80%, the Western Hemisphere was disproportionately represented. The third-largest geographical area was Europe, at 16.1%. Nonetheless, respondents came from organizations with headquarters and operations worldwide.

Just over half of the respondents worked in roles directly related to security.

Understanding the DevSecOps Landscape

Although the focus on the cloud is rising, the survey results are a reminder that not all applications are based in the cloud. In fact, 8% of respondents said their organization runs 100% of their applications on-premises, whereas only 9% said their organization is not running any applications on the premises at all. See Figure 2.

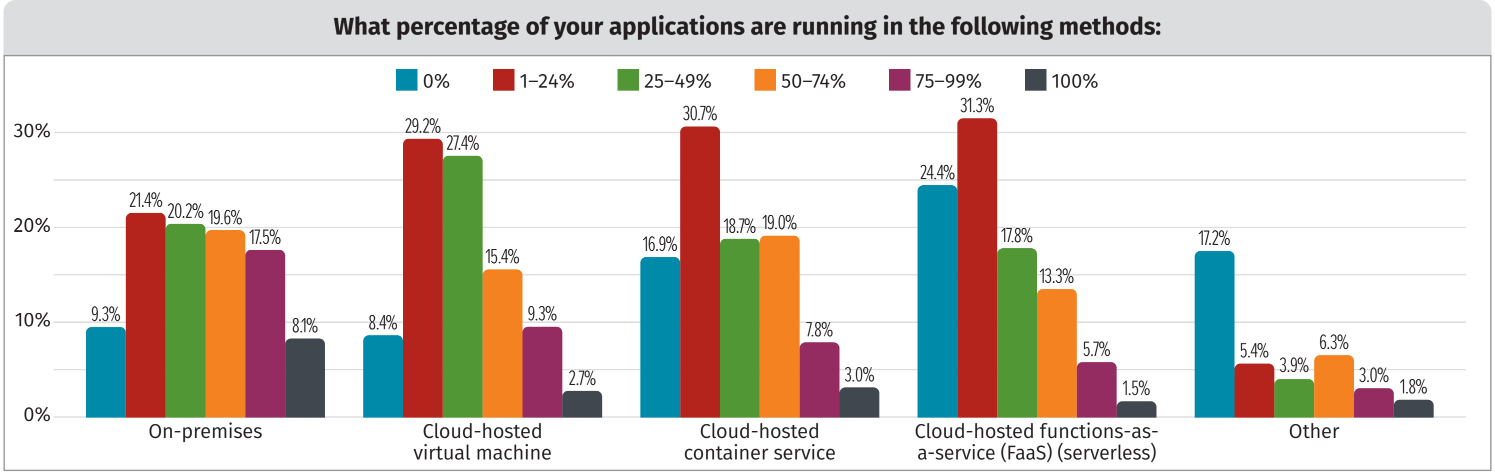

Figure 2. Most Commonly Used Platforms for Applications

This year, 65% of respondents said that their organization runs at least 25% of its applications on-premises. This is trending downward. In 2021, 83% said that at least 25% of their applications were running on-premises.

Cloud-hosted VMs are still preferred over cloud-hosted container services or cloud-hosted functions-as-a-service (FaaS, also called “serverless”). In this year’s survey, 55% of the respondents said that at least 25% of their applications run on VMs, compared with 49% of the respondents using cloud-hosted container services for at least 25% of their applications and 38% using FaaS.

The security implication of using a mix of VMs, containers, and serverless is that each of these technologies must be properly secured. For that to happen, DevSecOps teams must have the skills and tooling to secure all three methods. Organizations that use the cloud often can relegate some of the mundane security tasks to their cloud service providers, but this is not the case with applications hosted on-premises.

The survey responses suggest that organizations should consider hosting solutions that shift more security management to the cloud service provider. For example, 24% of respondents said they are not using FaaS, and 17% said they are not using cloud-hosted container services.

TAKEAWAY

DevSecOps teams may be underutilizing containers and serverless functions. Both serverless functions and containers lend themselves to CI/CD deployment and can be used to create immutable, performant, and secure applications that are cost-effective compared with VMs.

Application Hosting in the Cloud

It is apparent that companies are using multiple cloud services and that the distribution of applications running on each of the big three cloud service providers is starting to even out. Of the nearly 92% of respondents that said they are using the cloud:

- 77% have applications running on AWS
- 72% have applications running on Azure
- 56% have applications running on Google Cloud

On top of that, 25% said they are using other cloud hosting providers, including Alibaba Cloud, Oracle Cloud, IBM Cloud, and SAP-HANA Enterprise Cloud.

Another important finding is that there is a clear trend away from using a single cloud hosting provider to run the majority of an organization's workloads. Table 1 shows the percentage of respondents that use AWS, Azure, and GCP to run 75% or more of their applications as collected in the 2021 and 2022 annual surveys.

Table 1: Respondents Concentrating 75% or More of Workloads on a Single CSP, 2021 and 2022.

	AWS	Azure	GCP
2022	21.6%	14.8%	5.9%
2021	27.1%	18.4%	7.2%

An organization may choose to distribute its workloads across multiple cloud service providers for a variety of reasons, including business continuity planning and negotiating positions. The obvious security implication of using multiple cloud service providers is that each provider's environment must be properly secured, but they all work differently. The work multiplies with each additional provider used.

One way that leading DevSecOps teams are coping with the multicloud challenge is to create platform-agnostic applications, typically using containerization, so that the application can run in any cloud service provider's container service or even on-premises with the necessary infrastructure in place.

Securing Multiple Cloud Environments at Scale

As companies leverage multiple cloud service providers, with a mix of VMs, containers, and serverless, the challenge of ensuring that all those cloud resources are properly secured increases. To evaluate this challenge, the 2022 DevSecOps survey asked two questions:

- To what extent has your organization adopted Cloud Security Posture Management (CSPM) software, either commercial or open source?
- To what extent has your organization adopted Cloud Workload Protection Platform (CWPP) software? (See Figures 3 and 4.)

CSPM is underutilized. Although most respondents said they are using either a commercial or an open source CSPM tool, fewer than 20% of respondents said they are using the solution for 75% or more of their AWS accounts, Azure subscriptions, or GCP projects.

CWPP products are also underutilized. Although more than half of the respondents said their organization uses CWPP, a much smaller percentage (less than 17%) uses it in at least 75% or more of their AWS accounts, Azure subscriptions, or GCP projects.

CSPM software can help DevSecOps teams ensure that the cloud environments that host their applications are properly configured and secured using industry best practices, but only if this software is used consistently across all cloud accounts.

Similarly, CWPPs provide various security services for workloads regardless of whether the work is performed by VMs, containers, or serverless functions. In the past, this would typically require multiple agents to be installed, causing a drain on VM resources. CWPP is an essential technology for ensuring that the systems hosting one's applications are secure.

TAKEAWAY

As organizations continue to move away from using a single cloud hosting provider, the work of securing each cloud environment increases exponentially. Organizations should consider using or increasing their adoption of commercial or open-source CSPM software to ensure that each cloud infrastructure is secure. Similarly, CWPPs can be used to protect their compute resources.

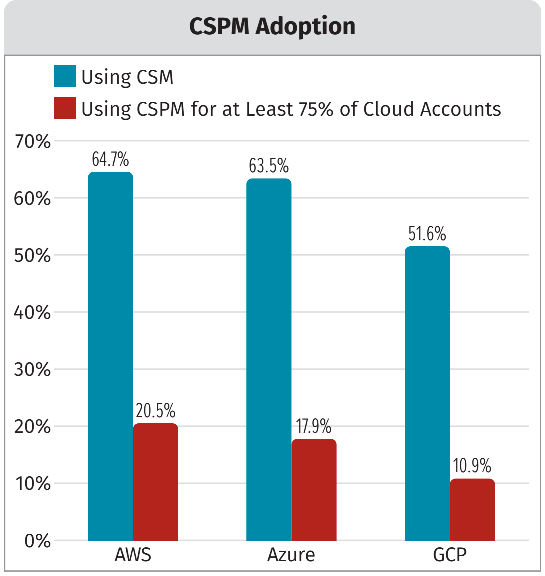

Figure 3. Extent of CSPM Adoption

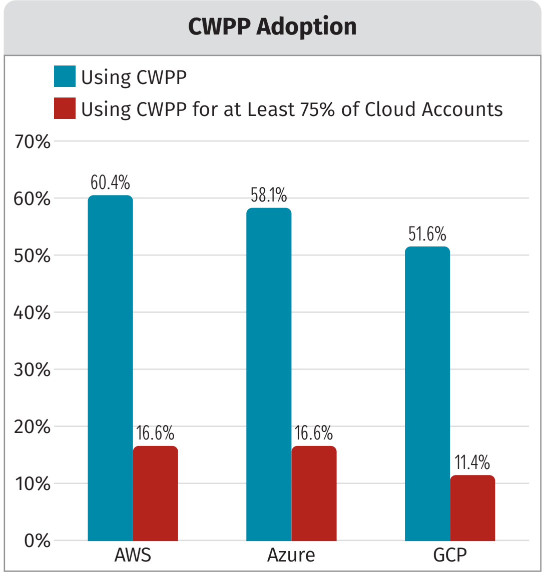

Figure 4. Extent of CWPP Adoption

Security at Velocity

Asked how often their organization delivers system changes to production, 61% of respondents said weekly or more frequently, and about 32% said changes were delivered at least once per day or on a continuous basis. Greater frequency has been the trend for the past six years. See Figure 5.

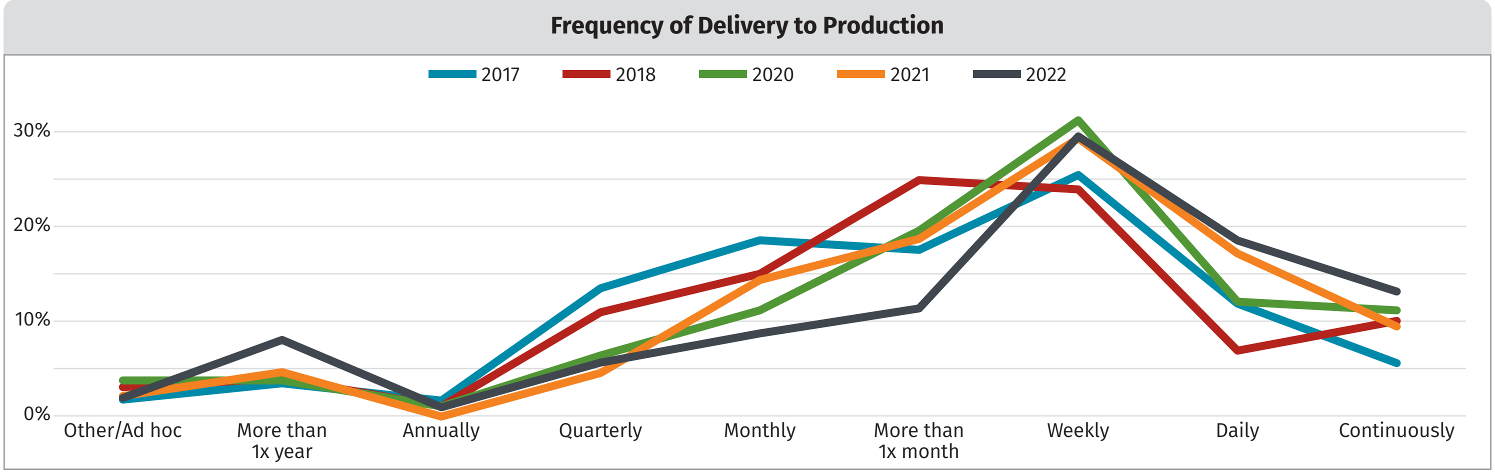

Figure 5. Frequency of Delivery to Production, 2017–2022

Investments in CI/CD tooling allow organizations to make small changes to their production codebase faster, with many teams able to deliver a constant stream of changes pushed to production. With the ratio of developers to security engineers increasing, it is clear that the only way to keep pace is to automate security testing in the CI/CD pipeline so that every code push is evaluated for security flaws.

Management should employ metrics to ensure that 100% of the codebase is deployed using CI/CD pipelines complete with security tests. Once all applications are subjected to security testing with each pass through the CI/CD pipeline, new security tests can be introduced to raise the bar until all security requirements are achieved. It is important to remember that security tests can only be designed to test for known issues. Therefore, penetration tests and bug bounties still have an essential role in a comprehensive application security program—to find unknown security issues. Cloud-Native Application Protection Platforms are being used to characterize normal application behavior and enforce zero-trust principles as an additional countermeasure against exploited security flaws. See the “DevSecOps Tools and Practices: What Works?” section.

Automated Compliance

Policy-as-code and CSPM are different techniques to enforce compliance policies automatically. In each of the past two SANS DevSecOps surveys, more than 60% of respondents (61.8% in 2021 and 60.3% in 2022) said that at least 50% of their organization's compliance policies are enforced automatically. Still, the number of respondents who said that 100% of their policies are enforced automatically increased in 2022 from 2021 (18.4% versus 5.1%). See Figure 6.

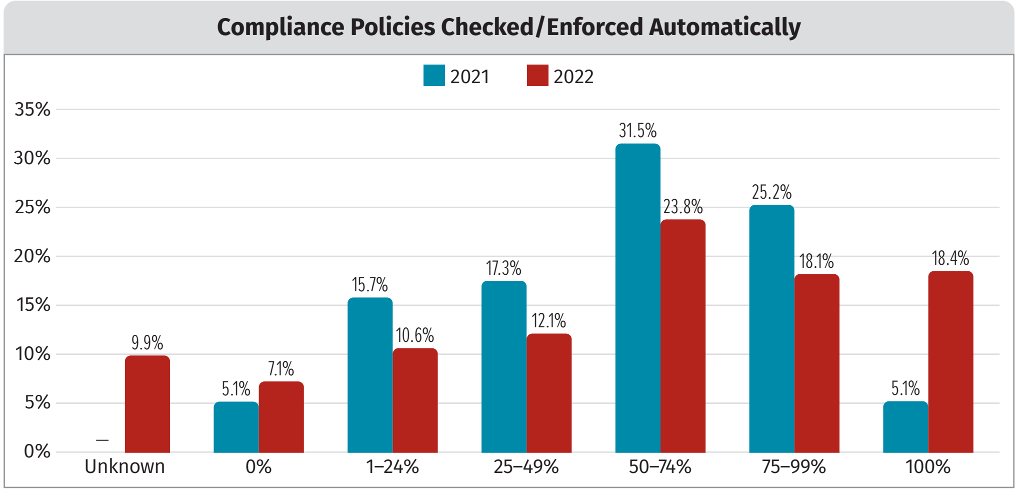

Figure 6. Percentage of Compliance Policies Checked or Enforced Automatically

The increased use of automated checking or enforcement of compliance policies shows that DevSecOps principles are starting to have a more significant impact. Security teams have realized that to cope with enterprise scale and development agility, they must apply DevOps principles to their own practices. Meanwhile, DevOps teams are integrating policy-as-code tests into their CI/CD pipelines to validate security policy compliance. These tests are cost-effective, too; writing a security test has a one-time cost that quickly approaches zero per test when that test is performed frequently. Both practices are helping organizations meet the goal of scalable continuous compliance.

Securing Container Services

When it comes to container orchestration tools, one question for cloud customers is whether to use Kubernetes, Docker Swarm, or some other option, and another is whether to use the tool as a managed service or manage it themselves on either cloud-hosted or on-premises VMs. Figure 7 shows the choices of container orchestration tools that respondents' organizations have made.

The survey responses this year show a noteworthy uptick in the use of on-premises container management over 2021:

- 51% on-premises Docker Engine versus 31% in 2021
- 38% on-premises Kubernetes versus 31% in 2021

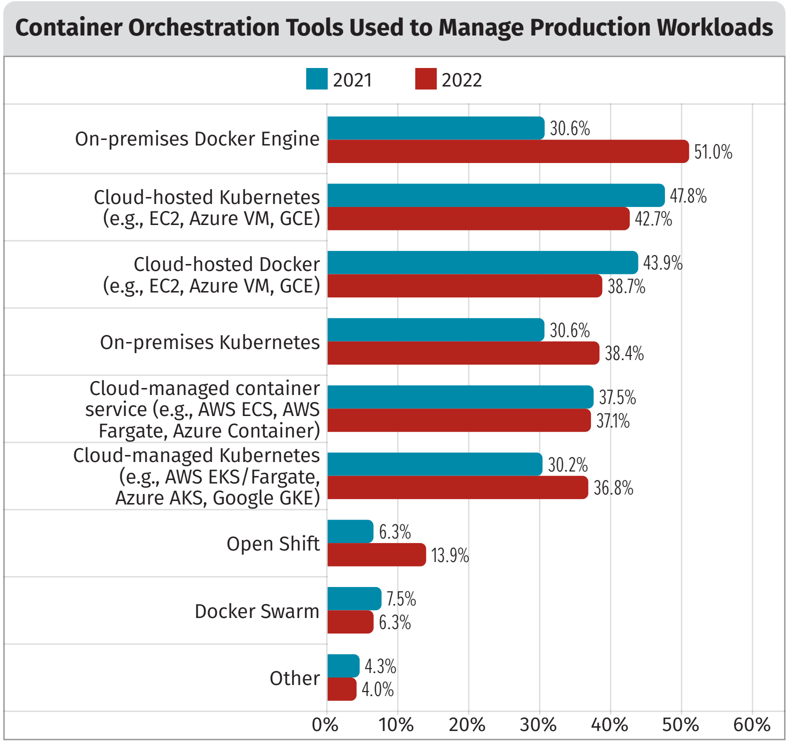

Figure 7. Container Orchestration Tools Used to Manage Production Workloads

The self-management of containers running in the cloud dropped by approximately five percentage points for both Docker Engine and Kubernetes:

- 39% cloud-hosted Docker Engine versus 44% in 2021
- 43% cloud-hosted Kubernetes versus 48% in 2021

The use of cloud-managed container services is essentially unchanged from last year, but there was an overall 7% increase in the use of cloud-managed Kubernetes services.

The most likely explanation for the shift to cloud-managed Kubernetes services is that organizations will typically use a managed service offering from a cloud service provider versus managing the service themselves if the managed service is deemed mature enough and offers the flexibility that the customer needs.

The increase in on-premises container management reflects a trend toward cloud-agnostic applications and the increased use of DevOps tools, techniques, and practices across the board—even for data that cannot be hosted in the public cloud based on its data classification.

TAKEAWAY

DevSecOps teams in 2022 are using on-premises container orchestration more than in 2021, but when they run containers in the cloud, they tend to use managed services rather than manage the container orchestration software themselves. Cloud-managed services generally provide improved security and financial benefits that DevSecOps teams should explore.

Programming Environments and Risks

Asked which languages and platforms in their application portfolio have been the greatest source of risk or exposure to their organization, respondents most often cited the languages used for web application development (e.g., Java, JavaScript, PHP, and HTML), along with Android. See Figure 8.

Because of these concerns, teams with mature secure DevOps practices have integrated static application security testing (SAST) and dynamic application security testing (DAST) into their CI/CD pipelines. Integrating security testing tools into the CI/CD tool chain can be effective, but only if the programming languages being used are supported by the tools. Application security reviewers will need to use only manual processes, such as code reviews, to find vulnerabilities in the code that cannot be scanned by security testing tools.

Organizations should develop processes and standards for selecting languages, relying on factors such as whether the company's CI/CD scanning tools support the language, the extent of developer experience with the language, and the risks inherent in using a particular language. Of particular concern are compiled, memory-unsafe languages such as C, C++, and Assembly. We recommend that organizations migrate to memory-safe languages for new projects and address weaknesses in older code whenever components need to be rewritten for other purposes.[1]

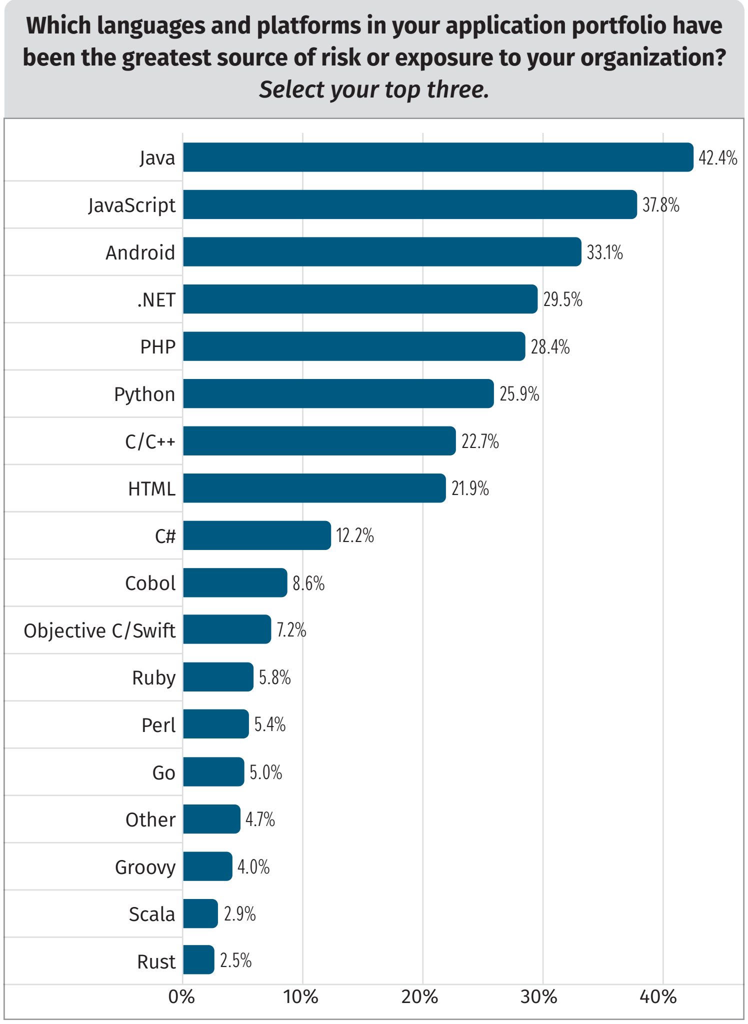
Which languages and platforms in your application portfolio have been the greatest source of risk or exposure to your organization?
Select your top three.

Figure 8. Languages and Platforms Posing the Greatest Risk or Exposure

[1] Internet Security Research Group, "What Is Memory Safety and Why Does It Matter?" www.memorysafety.org/docs/memory-safety

Languages commonly used for scripting, such as Python and Perl, and even Bash scripts, may introduce risks that the organization has not fully assessed. Although scripts that automate various processes are an important company asset, the life cycle of the script is often not managed the same way that application code is managed. Scripts may not undergo a security testing regimen or peer review and may not always be committed to a source code control system, such as a Git repository.

TAKEAWAY

DevSecOps teams should limit the programming languages approved for new development projects based on security risks and support of security testing tools, among other factors, and they should refactor older code written with memory-unsafe languages as opportunities arise.

DevOps Foundational Practices

We were curious about cloud security architects' role in DevSecOps improvements. The survey told us:

- 58% of the respondents said that their company has personnel focused on cloud security architecture and that this team includes DevSecOps process improvement as part of its focus. We clarified that the architects, as we defined them, did not have additional development, security operations, or production operation duties.
- 21% said that their cloud security architects are not focused on DevSecOps process improvement, indicating that only teams with development, security operations, or production operation duties have DevSecOps process improvement as a focus.
- 18% indicated that their DevSecOps process improvement efforts are "spurious and ad hoc."

As with any other best practice, DevSecOps process improvement can only happen with focused intention. The development, security operations, and production operation teams have the biggest stake in DevSecOps process improvement, but insight can come from personnel not involved in the daily grind.

As shown in Figure 9, the use of on-premises open source tools such as Jenkins has overtaken the use of cloud-hosted CI/CD tools. This interesting result might be related to cost or could reflect the earlier observation that more of the respondents in the 2022 survey are running their applications on-premises than those responding in 2021.

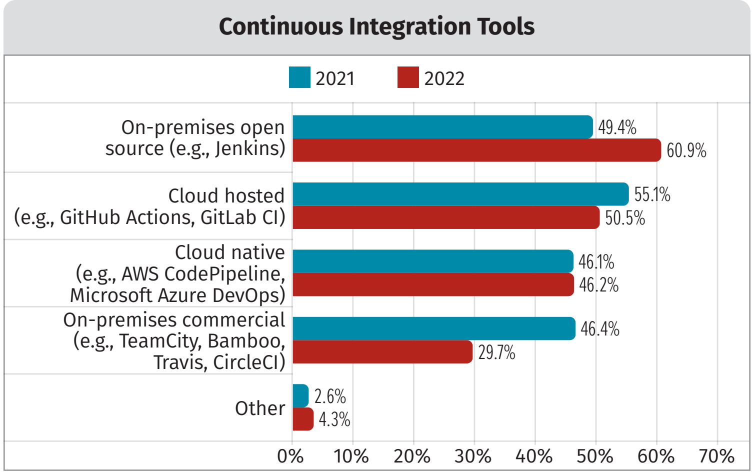

Figure 9. Continuous Integration Tools Usage, 2021 and 2022

Another anomaly is that the use of on-premises commercial tools such as TeamCity and Bamboo has dramatically decreased, from 46% of respondents last year to 30% in 2022. This might be attributed to shrinking budgets that make less-expensive, open-source tools more attractive. As for the use of cloud-native tools from the various cloud providers, the survey results this year were much the same as in 2021, with around 46% of respondents saying their organization uses them.

Companies must ensure that their CI/CD tools are implemented securely, especially when deployed on-premises or self-managed in the cloud. The supply chain attack on SolarWinds Orion® software[2] has taught malicious actors that compromising the tools that build and deploy software can be an effective tactic.

Figure 10 shows an across-the-board increase in security testing at each phase of the build and release workflow except for the use of integrated development environment (IDE) plug-ins for security testing.

Architecture/design still tops this list, suggesting widespread agreement that security testing should be addressed early in the build and release workflow.

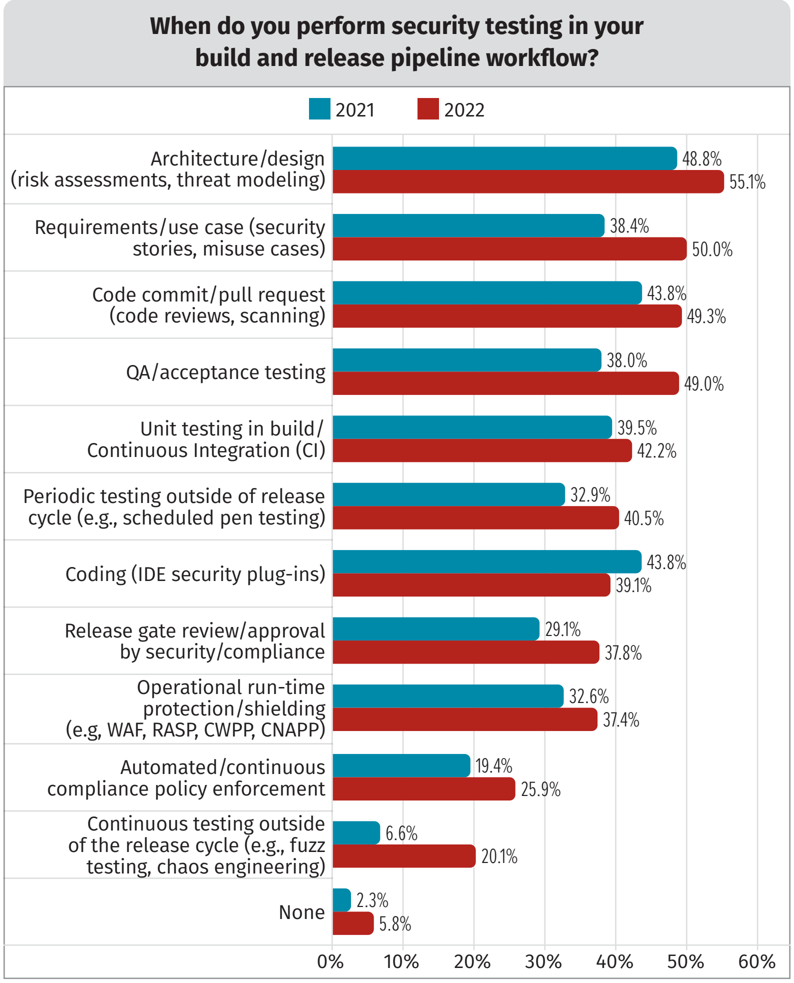

Figure 10. The Timing of Security Testing in Build and Release Pipeline Workflows, 2021 and 2022

Code commit/pull request is still considered an important phase for security testing, and this year's survey shows an appropriate jump in testing at both the requirements/use case phase and the QA/acceptance phase, because the two go hand in hand.

These results show that the "shift left" principle, which holds that security is best addressed early in the development life cycle, is being followed by DevSecOps practitioners. This is a positive development.

[2] Palo Alto Networks Unit 42, "SolarStorm Supply Chain Attack Timeline," December 23, 2020, https://unit42.paloaltonetworks.com/solarstorm-supply-chain-attack-timeline/

DevSecOps Tools and Practices: What Works?

For managing risk throughout the software development life cycle (SDLC), organizations can take advantage of several security tools and practices. Table 2 shows the results from the survey when respondents were asked which tools and techniques they found most useful.

Table 2. 2022 Survey Results on Usefulness of Various Security Testing Practices and Tools

Practice/Technique/Tool	Very Useful	Useful	Not Useful	Total Useful
Web application firewalls: A plug-in or appliance that filters out specific signatures in HTTP traffic. The major cloud platforms offer native WAF services that provide basic protection against common HTTP and DoS attacks. Several container security vendors and open source products provide container WAF features for detecting attacks within a container network.	45.2%	38.8%	4.1%	84.0%
Periodic vulnerability scanning: A fundamental part of any vulnerability management program, regular, periodic (e.g., monthly) automated vulnerability scans leave a window of exposure open in rapidly changing environments.	38.4%	44.9%	4.8%	83.3%
Secure coding training for developers and engineers: Training developers and engineers in defensive coding and secure programming concepts, risks, and techniques is key to shifting responsibilities for security to a stage early in the design and coding life cycle. Developers also need security training to be effective participants in threat modeling, perform code reviews, understand and accept SAST tools, and so on.	42.5%	39.8%	4.1%	82.3%
Automated static analysis: Organizations can perform automated code scanning such as SAST at multiple points in engineering workflows to catch common coding mistakes and vulnerabilities: • High-fidelity, immediate checks in IDE using integrated plug-ins • Custom rules in pre-commit hooks • High-fidelity, fast, incremental checks in CI (time-bound) • Checks for embedded secrets in code and repositories, later in build (e.g., git-secrets, OWASP Sedated, truffleHog) • Deeper scans of the entire code repo on a regular frequency (e.g., nightly), outside of the build pipeline Different engines catch different problems, so you need multiple scanners to get high levels of confidence. SAST should prove an easy sell to DevOps teams because it works at a level that they care about and understand. SAST also has a training effect. Over time, developers will change their coding practices in response to tool findings. But to be accepted, the tools must be fast, accurate (focused on problems that developers and engineers will fix), and integrated early into coding workflows.[3]	41.2%	40.8%	4.8%	82.0%
Continuous vulnerability scanning: Continuous scanning for changing threats and vulnerabilities allows organizations to rapidly identify new risks and priorities and to track the status and success of patching programs. It involves frequent scanning and incremental analysis of results to surface changes in security posture and risks.	42.9%	36.1%	4.8%	79.0%
Internal penetration testing: Internal testers are generally more familiar with the domain and environment, but most organizations lack the technical expertise required for effective penetration testing.	39.1%	38.8%	5.8%	77.9%
Open source/third-party dependency analysis: Software dependency, or software composition analysis (SCA), tools automatically identify licensing problems and known vulnerabilities in code dependencies (libraries and frameworks). Organizations can perform this scanning as a gating process on pull requests to identify added dependencies, and later during the build process to check for newly reported vulnerabilities.	30.6%	46.3%	5.8%	76.9%
Third-party penetration testing: The effectiveness of periodic manual testing by outside experts depends heavily on testers' expertise, their familiarity with the domain and environment, and the time they have available to conduct the tests. Testing is generally done to meet compliance requirements. Point-in-time assessments such as this offer limited value in continuously changing, Agile environments, but they are one of the most effective ways to find real security vulnerabilities. Good pen testers find problems that scanners and other tests cannot. Although organizations cannot use penetration tests to verify that a system is secure, they can and should use them as a health check on the state of the secure SDLC. Teams should swarm on the results, conducting a post-mortem review to understand what they missed and improve their training and testing.	33.0%	43.9%	7.8%	76.9%

[3] "How Google and Facebook Do Code Analysis," https://techbeacon.com/devops/how-google-facebook-do-code-analysis

Table 2. 2022 Survey Results on Usefulness of Various Security Testing Practices and Tools (Continued)

Practice/Technique/Tool	Very Useful	Useful	Not Useful	Total Useful
Upfront risk assessments before development starts: Lightweight risk assessments identify applications and services that have compliance, security, or operational risks early in the design and concept stages. Development teams work with security engineers to identify sensitive information, map threat scenarios, and identify data and services that need to be protected. See Mozilla's Rapid Risk Assessment as an example.[4]	34.7%	41.5%	3.4%	76.2%
Network detection and response (NDR)/network traffic analysis (NTA): NDR and NTA capabilities are important for understanding, managing, and securing east-west network communications in hybrid cloud architectures and container-based microservices. They provide deep visibility into network traffic and use machine learning to identify real-time threats and attacks. Public cloud providers' traffic-mirroring services have made it easy to deploy these technologies, as part of a "shift right" approach to operational security. Commercial and open source products enable deep packet inspection for east-west container traffic to detect and prevent network-based attacks.	30.6%	44.9%	5.4%	75.5%
Threat modeling, attack surface analysis, or architecture/design reviews: Threat modeling or design reviews focused on identifying security threats and risks are difficult to implement in iterative, incremental development, where the design is under continuous refinement.	34.0%	40.8%	6.1%	74.8%
Container/image security scanning: Organizations can and should do static scanning of containers at different points for different threats: • Scanning container images or code templates for common configuration mistakes and checking against best practices for setup and use of containers • Looking inside container image layers to identify dependencies and known vulnerabilities in the software • Scanning containers in registries for vulnerable images (now provided by most registry services)	32.0%	40.8%	6.1%	72.8%
Next-generation web application firewalls: NGWAFs provide higher-fidelity, cloud-based runtime protection for web applications/APIs, which consolidates threat information with application context. The boundary between NGWAF and RASP can be blurry.	35.0%	37.1%	5.1%	72.1%
Manual code review: Lightweight peer code reviews, done through pull requests, have become a common practice in modern software development, but their effectiveness in finding vulnerabilities depends on the reviewers' skills and knowledge, the time they have available, and their priorities. Research at organizations including Microsoft and Google shows that reviewers spend more time on readability and maintainability problems than looking for real defects. Although readability is important (difficult-to-read code likely contains more defects and proves riskier to change), reviewers need to be trained to look for more fundamental problems—including training in secure coding and defensive coding. Manual code reviews are more effective when combined with SAST because automated scans can find subtle problems that reviewers may not be aware of.	22.1%	48.6%	16.7%	70.7%
Dynamic application security testing (DAST): Organizations can perform dynamic scanning of web applications/APIs late in the CI/CD pipeline, after the code is built and the system is provisioned for functional testing, integration testing, and acceptance testing. They can proxy these tests through a scanner to automatically execute passive and active vulnerability scans.	31.6%	38.8%	4.4%	70.4%
Security stories, abuser stories, or evil-user stories to inject security into requirements backlog: Developers can be asked to identify personas for "bad users" (e.g., fraudsters, hackers, insider threats) and write requirements (user stories) from a negative perspective, to identify threats to the system or service. "As a hacker, I want to…". A variation of standard Agile practices, security stories encourage development teams to think outside of functional user stories and to address security requirements and risks in design and development.	25.5%	44.6%	6.5%	70.1%
Compliance reviews or audits by a third party: Audits and assessments of control programs such as SOC 2, ISO 27001, and PCI are required for regulated industries. In DevOps environments, these assessments present challenges around continuous incremental change, CD, and "you build it, you run it," which can violate requirements for separation of responsibilities.	27.2%	42.9%	10.9%	70.1%
File integrity monitoring/host-based intrusion detection system (HIDS): File integrity monitoring and HIDS such as Tripwire and OSSEC track changes to files and configuration data. In rapidly changing Agile environments, this results in lots of noise, making it difficult to separate authorized changes from suspicious activity.	27.2%	42.5%	8.8%	69.7%
Virtual patching: This refers to applying protection in WAFs, RASP, and CWPP at runtime to block known attack signatures as a rapid response to newly discovered vulnerabilities. Organizations can use virtual patching as a temporary stopgap until they can apply and roll out a patch to the software.	29.9%	38.8%	7.8%	68.7%
Cloud Security Posture Management: CSPM tools and services automatically scan cloud instances to detect and catch common configuration mistakes and known threats and to enforce security and compliance policies. Any cloud environment has too many configuration options and too many compliance and security policies to review manually.	28.6%	35.0%	5.4%	63.6%
Interactive Application Security Testing: IAST instruments the application runtime and detects application vulnerabilities as a system or service runs. This passive testing technique depends on other tests (e.g., functional tests) to exercise the code.	23.5%	39.5%	6.5%	63.0%

[4] https://infosec.mozilla.org/guidelines/risk/rapid_risk_assessment.html

Table 2. 2022 Survey Results on Usefulness of Various Security Testing Practices and Tools (Continued)

Practice/Technique/Tool	Very Useful	Useful	Not Useful	Total Useful
Cloud Workload Protection Platforms: CWPPs provide runtime protection for containers and cloud instances. This broad technology category covers services that offer different levels and types of runtime shielding, including network segmentation, system integrity protection, application control/whitelisting, behavioral monitoring, host-based intrusion prevention, and optional anti-malware protection.	22.8%	37.1%	4.8%	59.9%
Cloud-Native Application Protection Platforms: CNAPPs are an integrated set of security and compliance capabilities designed to help secure and protect cloud-native applications across development and production.[5]	21.8%	37.8%	3.7%	59.6%
Fuzz testing: Fuzzing refers to automated negative testing for APIs, file-based applications, and protocols. Security researchers use fuzzing because it can find real vulnerabilities with few false positives. It proves especially useful and important for applications written in memory-unsafe languages such as C and C++ (although fuzzing is also available for languages such as Go, Rust, Java, and Python). Fuzzing can be difficult to add into CI/CD, because good fuzzing takes time to set up and run, and it takes more time to interpret the test results. However, modern cloud-based fuzzing services make this easier, using limited test cases in the build pipeline (for example, GitLab now offers fuzzing options) or continuously outside of CI/CD.	22.8%	35.4%	9.5%	58.2%
Runtime Application Self-Protection: RASP instruments the application runtime (e.g., JVM or .NET CLR) to provide operational visibility into attacks and to defend against common application security attacks and language- or framework-specific vulnerabilities. RASP adds some runtime overhead and operational complexity in return for runtime protection.	21.4%	36.1%	7.1%	57.5%
Bug bounties: In these programs, organizations invite outside white-hat testers (such as security researchers or ethical hackers) to continuously test systems and report vulnerabilities, on a reward basis. Bug bounty programs at organizations such as Google, Apple, Microsoft, and Facebook attract a lot of attention, but the programs require a lot of work to set up and manage.	19.0%	36.7%	9.5%	55.7%
Other	7.8%	9.9%	1.4%	17.7%

The 2022 survey results compare almost identically with those from 2021 in organizations that leverage **build automation, automated testing**, and **CI** (see Figure 11), and it seems important to call them out as foundational DevSecOps practices.

The use of **continuous monitoring and measurement** is notably more widespread in 2022 than in previous years, suggesting that it is contributing to the success of DevSecOps teams. Increases in **automated testing** being performed indicate that secure coding practices are increasingly influential in DevSecOps workflow processes.

On the other hand, **immutable infrastructure provisioning, blameless retrospectives**, and **chaos engineering** remain underutilized within DevSecOps.

Which of the following practices does your organization follow?
Select all that apply.

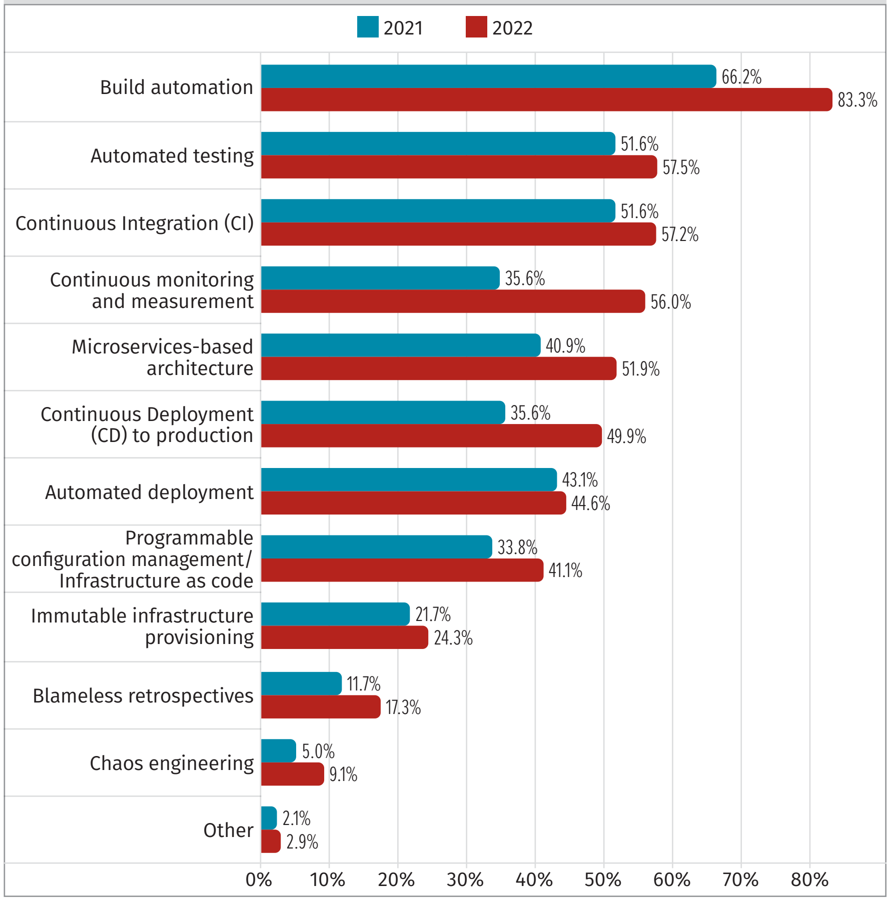

Figure 11. Respondents' Adoption Rates of Various DevSecOps Practices, 2021 and 2022

[5] https://blog.aquasec.com/what-is-cnapp

Organizations that want to make DevSecOps advancements can benefit by contemplating how these techniques can serve them. With 50% of respondents saying they are using CD, the next step is to ensure that all changes to systems are made via the CI/CD pipeline and to treat any attempt to circumvent this change management process (regardless of intention) as a security incident—to "electrify the fence."

When there are incidents, whether they are service level incidents or security incidents, a blameless retrospective can be used to ensure that the organization learns how to avoid failing the same way twice. DevSecOps teams that have effective blameless retrospectives improve the fastest, because nonfatal mistakes are the fastest way to learn, when properly examined.

In fact, advanced organizations accelerate this learning process by injecting controlled failure into their systems using chaos engineering techniques. As Netflix, a pioneer in using chaos engineering, put it, "If we aren't constantly testing our ability to succeed despite failure, then it isn't likely to work when it matters most—in the event of an unexpected outage."[6]

It is important to stress that a "crawl, walk, run" approach is best when starting out with chaos engineering. Many practitioners recommend starting with a tabletop exercise.

TAKEAWAY

Organizations that want to make rapid DevSecOps advancements should consider implementing immutable infrastructure provisioning, blameless retrospectives, and chaos engineering to accelerate learning while improving confidentiality, integrity, and availability.

KPIs and Metrics

KPIs are a limited set of measurements taken over time that are selected to help management measure performance and take corrective action. KPIs can be leveraged to better understand trends from a historical perspective and forecast trends that may occur in the future. They can assist in ascertaining DevSecOps process stability and capability while also providing insight into alternative processes and tools and into how to manage change in an organization.

Figure 12 shows that the **number of open security vulnerabilities** continues to be the top KPI and that **time-to-fix security vulnerabilities** remains the number two KPI for measuring the success of DevSecOps activities.

Interestingly, the use of these KPIs appears to correspond to the 54% of respondents stating that their organization resolves critical security issues within a week or less. The value of these metrics is conveyed by the axiom "What is measured is controlled" and its corollary, "What is not measured is not controlled." Management must have the appropriate visibility to focus organizational resources on the underperforming metrics.

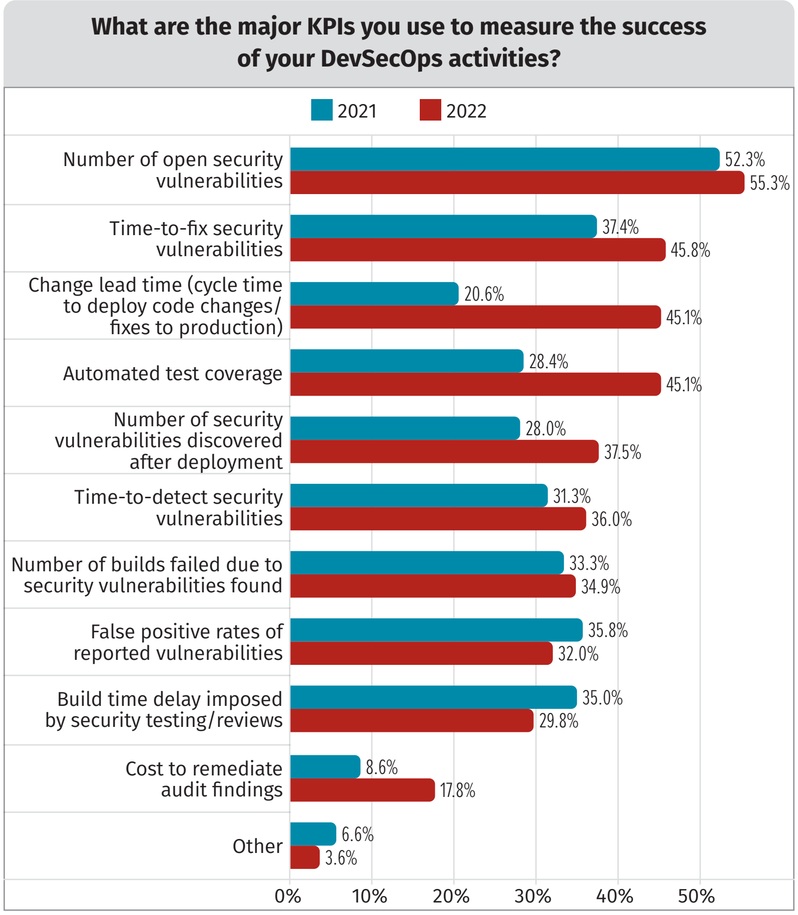

Figure 12. Top KPIs Used in Respondents' Organizations, 2021 and 2022

[6] Netflix Technology Blog, "5 Lessons We've Learned Using AWS," December 16, 2010, https://netflixtechblog.com/5-lessons-weve-learned-using-aws-1f2a28588e4c

The responses in this year's survey show that **change lead time, automated test coverage**, and **number of security vulnerabilities discovered after deployment** all were higher than in the 2021 survey. This suggests that organizations recognize the importance of having additional coverage for the automated testing of security best practices that are in line with release management efforts, with the objective of pushing code into production faster. The response rate for the **cost to remediate audit findings** KPI is lower than expected, but this may be related to the difficulty in calculating those costs.

It's worth noting that although security requirements traditionally have been considered a bottleneck for production deployments, the right metrics focus the DevSecOps team on the proper priorities, including the security of the system.

On the other hand, metrics that appear to be the most prevalent of the surveyed KPIs—such as cycle time to deploy code fixes to production, automated test coverage, and number of security vulnerabilities discovered after deployment—offer opportunities for companies to benchmark against peer organizations.

TAKEAWAY

Organizations should leverage KPIs to focus the organization on the next most important areas for improvement. Benchmarking metrics with peer organizations can be used to garner management support and helps demonstrate due care.

Top DevSecOps Challenges and Success Factors

A successful DevSecOps approach requires continuous collaboration between the development, security, and operations teams. As the DevSecOps team comes to agreement on processes, tools, and techniques, its decisions get codified in the enhancements to the CI/CD pipeline, resulting in a gradual increase in the overall maturity of an organization's SDLC. Figure 13 shows respondents' opinions in both 2021 and 2022 about what has contributed most to their security program's success.

Securing management buy-in has supplanted **improving communication across dev, ops, and security** as the No. 1 factor of success. Unsurprisingly, **automating build/test/deploy/provisioning workflow** and **integrating automated security testing into developer/engineering tool chains and build/deploy workflows** continue to be considered of high importance to the success of DevSecOps programs. Overall, this represents a focus on progressing the dramatic push within organizations to develop a deep-rooted culture where the ownership of security is shared among the various teams.

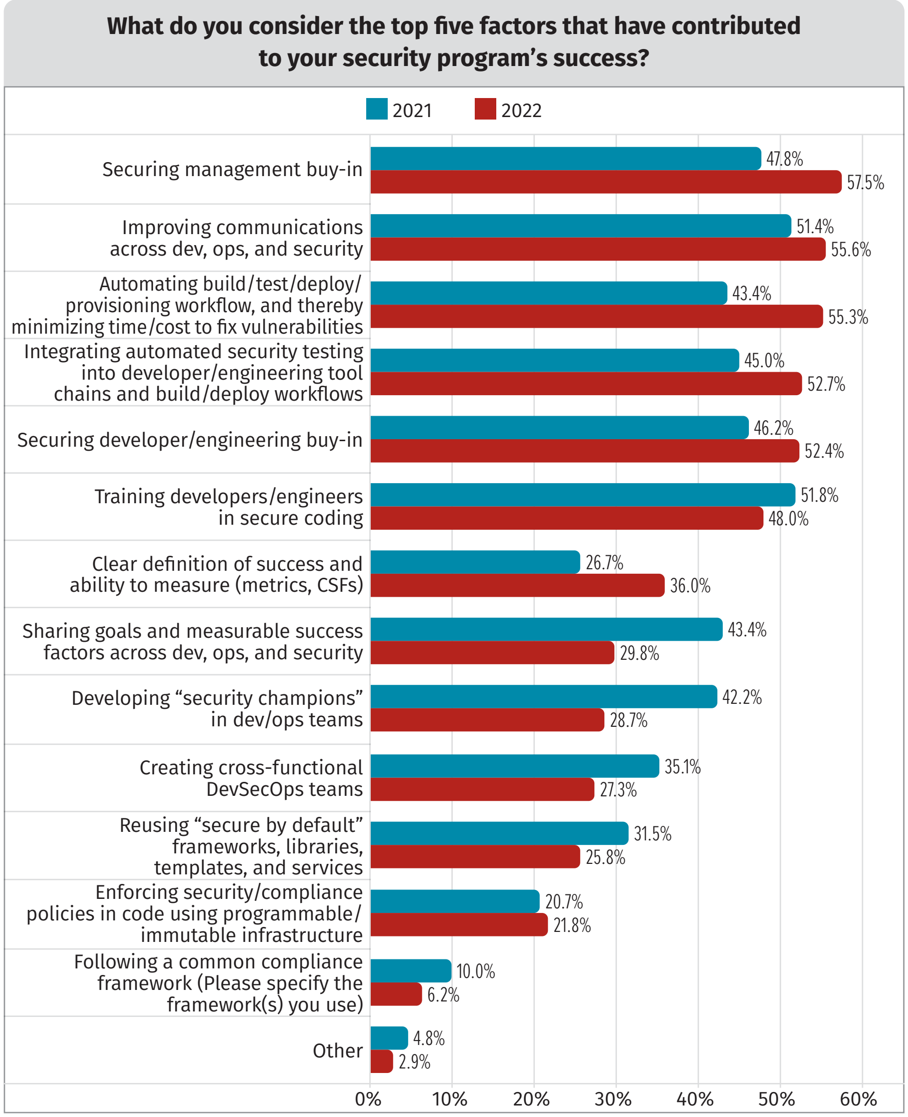

Figure 13. Respondents' Ranking of DevSecOps Success Factors, 2021 and 2022

As for challenges to success, it is also no surprise to see **shortage of cloud security personnel/skills** at No. 1 (tied with **lack of developer/engineer buy-in**; see Figure 14).

Somewhat at odds with each other are the changes in responses about the challenges of **insufficient budget/funding for security programs and tools**, which dropped by a bit more than 10 percentage points, and the closely related **lack of management buy-in**, which rose by nearly the same amount. It's also worth noting that although **improving communication across dev, ops, and security** was ranked as the No. 2 success factor, **organizational silos between development, operations, and security teams** is still a big challenge, along with the associated **lack of transparency into development/operations work**.

With leadership support and encouragement, it is evident the solution to these barriers is to foster better workplace communication among the development, security, and operations teams. Additionally, the results point to the need to attract persons who embrace DevSecOps practices by continuously reinforcing those who thrive by solving problems and enjoy being innovative.

TAKEAWAY

Organizations should continue to foster a culture of a shared responsibility model for security throughout teams, processes, and projects by leveraging documentation, training, and the socialization of DevSecOps best practices.

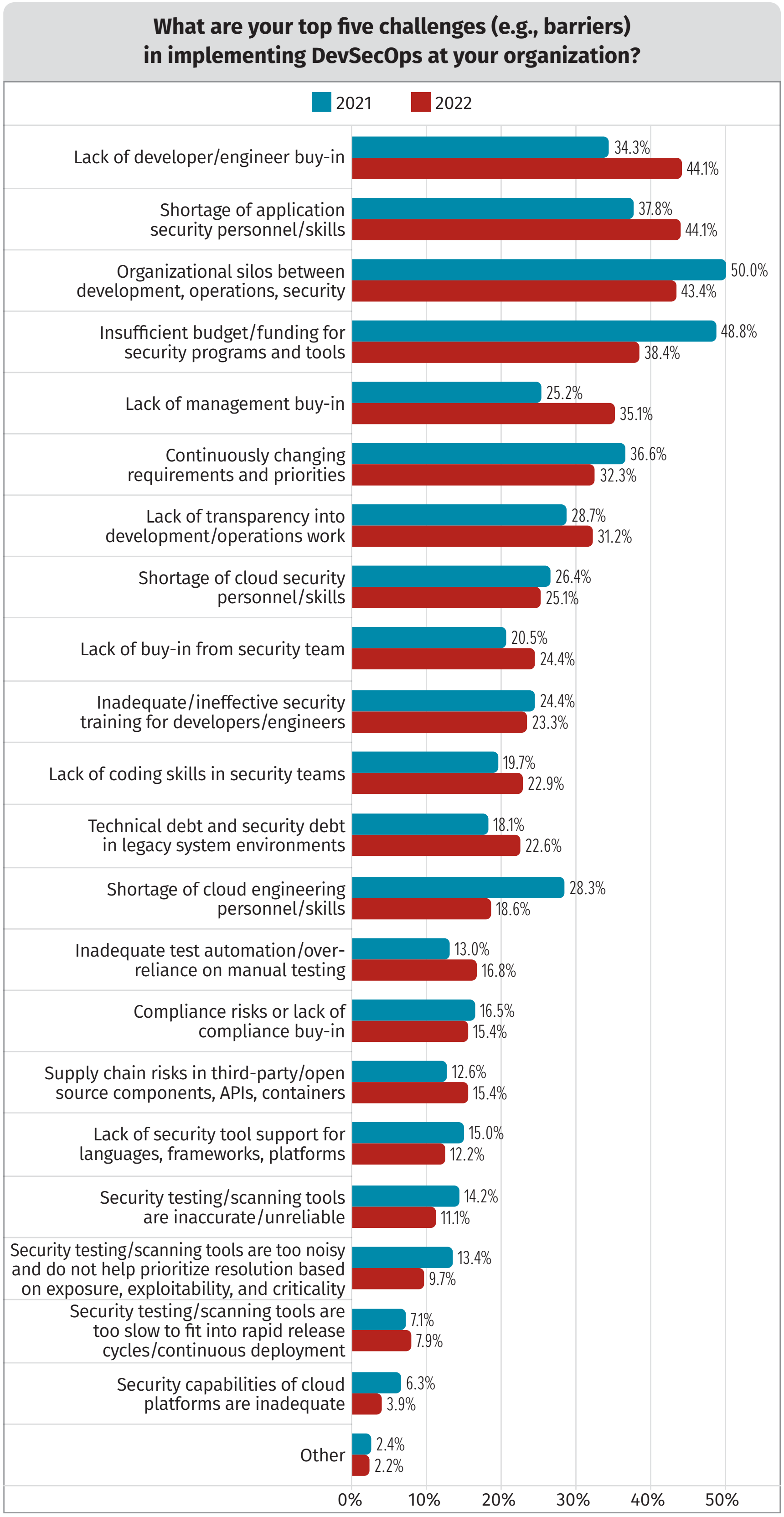

Figure 14. Respondents' Ranking of DevSecOps Challenges, 2021 and 2022

Trends to Watch

It will be interesting to see what lessons are learned by forward-thinking organizations that keep an eye on emerging trends, experiment with the adoption of several concurrent technologies, learn from the mistakes of others, and build a culture of blamelessness. Those that do will create opportunities for growth while fostering a sense of failing forward toward success. The extent to which respondents' organizations are exploring emerging technologies are listed in Table 3.

Table 3. Emerging Technology for DevSecOps

Extent your organization is exploring or integrating this future trend (% of respondents)	Unknown	Not at all	Conducting preliminary investigation	Experimenting or conducting pilot projects	Partially integrated	Fully integrated
Applying artificial intelligence or data science to improve DevSecOps	20.0%	32.5%	23.2%	10.0%	10.4%	2.9%
Integrating security operations into AI and MLOps	16.8%	26.4%	24.6%	15.4%	12.5%	3.2%
Integrating security operations into data science operations	17.5%	23.6%	18.9%	14.6%	17.9%	5.7%
Utilizing serverless computing to build, manage, and scale applications	16.8%	12.5%	20.4%	18.6%	20.4%	10.4%
Leveraging GitOps to test, verify, automate, deploy, and advance the principles of infrastructure as code	18.6%	13.6%	15.4%	21.8%	21.1%	8.9%
Developing software as microservices rather than as monolithic applications to improve the overall agility and flexibility for DevSecOps	14.6%	10.7%	15.4%	15.0%	29.6%	13.2%
Leveraging Application Security Orchestration and Correlation (ASOC) tools for DevSecOps	22.5%	16.8%	16.8%	13.9%	19.3%	9.6%

Innovative capabilities such as AI, GitOps frameworks, serverless, and microservices may help transform DevSecOps, with the possibility of further streamlining organizations' resource-constrained teams. Although it is still very early, it will be interesting to see what lessons the early adopters learn and how these lessons can transform DevSecOps practices.

The use of ASOC tools will likely increase in years to come, and the adoption of AI, machine learning, and other data science methodologies and tools will help to improve DevSecOps. Microservices offer DevSecOps teams the advantages of flexible, highly scalable, resilient, and easy-to-deploy code. Identity-based and network-based protections (such as microsegmentation) are being applied to enable organizations to achieve the widely sought zero-trust approach. Through the orchestration of microservices, containers, and serverless technology, DevSecOps has the potential to secure code more thoroughly than has ever been achieved before.

Leveraging data science and AI capabilities appears to be another great opportunity for DevSecOps to gain better visibility and improve both proactive and reactive capabilities. The survey results suggest that we should consider being open to more collaboration, so that more DevSecOps principles are incorporated in both data science and AI operations, as well as bringing those capabilities to DevSecOps. AI technologies may improve both productivity and accuracy for developers, and when developers are better able to hit their deadlines, more thorough security vulnerability checks will be possible, and they will more expediently find flaws in the code. AI software may also significantly reduce the risks associated with the SDLC and improve the rejection of false positives. Real-time security risk assessments will also allow applications to be produced at a faster rate. Overall, it seems there is a strong desire for cross-pollination between the DevSecOps and data science communities, heightened by the shortage of well-qualified personnel in both professions.

TAKEAWAY

Innovative and forward-thinking organizations that are willing to experiment with emerging technologies have a great opportunity to help build the future of DevSecOps; it will be exciting to see how these emerging technologies play out in the next generation of tools.

Moving Forward

DevSecOps seeks to unify the security team with the development and operations teams by increasing collaboration throughout the entire systems development life cycle. Although this requires a significant and ongoing investment by the organization, the benefits are well documented and include less security toil, reduced mean time to remediate security issues, and increased ownership of application security.[7]

Overall, the survey points toward the maturation of DevSecOps, and this is encouraging. The survey responses also illuminate opportunities to improve. Key takeaways include:

- DevSecOps teams may be underutilizing containers and serverless functions. Both serverless functions and containers lend themselves to CI/CD deployment and can be used to create immutable, performant, and secure applications that are cost-effective compared with VMs.
- As organizations continue to move away from using a single cloud hosting provider, the work of securing each cloud environment increases exponentially. Organizations should consider using or increasing their adoption of commercial or open-source CSPM software to ensure that each cloud infrastructure is secure. Similarly, CWPPs can be used to protect their compute resources.
- DevSecOps teams in 2022 are using on-premises container orchestration more than in 2021, but when they run containers in the cloud, they tend to use managed services rather than manage the container orchestration software themselves. Cloud-managed services generally provide improved security and financial benefits that DevSecOps teams should explore.

[7] DevOps Digest, “A Primer on Secure DevOps: Learn the Benefits of These 3 DevSecOps Use Cases,” July 18, 2022, www.devopsdigest.com/secure-devops-use-cases

- DevSecOps teams should limit the programming languages approved for new development projects based on security risks and support of security testing tools, among other factors, and they should refactor older code written with memory-unsafe languages as opportunities arise.
- Organizations that want to make rapid DevSecOps advancements should consider implementing immutable infrastructure provisioning, blameless retrospectives, and chaos engineering to accelerate learning while improving confidentiality, integrity, and availability.
- Organizations should leverage KPIs to focus the organization on the next most important areas for improvement. Benchmarking metrics with peer organizations can be used to garner management support and helps demonstrate due care.
- Organizations should continue to foster a culture of a shared responsibility model for security throughout teams, processes, and projects by leveraging documentation, training, and the socialization of DevSecOps best practices.
- Innovative and forward-thinking organizations that are willing to experiment with emerging technologies have a great opportunity to help build the future of DevSecOps; it will be exciting to see how these emerging technologies play out in the next generation of tools.

Organizations face mounting pressure to do more work with fewer resources—especially staff. DevSecOps is a solution to that pressure. The right KPIs will keep the team focused on the proper priorities, and investments in build and testing automation will increase agility, including around responding to security incidents.

The three critical security areas of focus for DevSecOps are:

- Automated test-driven security checks against defined standards
- Continuous monitoring and the leveraging of automatic remediation in order to react to attacks in a timelier fashion
- The ability to assess software early in the SDLC to minimize risks prior to code being deployed in production

Many organizations feel a desperate need for more highly qualified personnel in DevSecOps. Because demand continues to outweigh supply, there is a real need to spark more interest in this ever-changing field. To cope with the scarcity of talent amid competitive pressures, organizations should further leverage proven DevSecOps practices and explore emerging technological capabilities. This can mean harnessing some of the underutilized technology (e.g., CSPM, CWPP, AI, ML, ASOC) or applying new tools, technologies, and practices in pursuit of optimizing and streamlining DevSecOps (immutable infrastructure, zero-trust, benchmarking).

DevSecOps' ultimate objective is to significantly improve the organization's security posture and operational effectiveness by aligning the development, security, and operations teams. This survey showcases the progress made by the community, recognizes the challenges, and highlights areas for additional focus on the path to DevSecOps excellence.

Sponsor

SANS would like to thank this paper's sponsor:

SYNOPSYS®