YOUR 2024 GUIDE TO A

Smarter DevOps

nvisia
connect. build. enable.

Discover more resources
at nvisia.com.

Contents

3 **Speed, Stability, & a Culture of Innovation: The Power of DevOps**

It's Common Sense for Anyone in Software

5 **The DevOps Way**

The Three Tenants of DevOps

5 **So, What Does the Best DevOps in Action Look Like?**

The Top-Performer Cloud Native Toolkit

7 **Making DevOps Work for You – Getting Started**

First things first: trust, communication, and an improvement mindset.

OK, but now what?

- **8** . . . . 1. Begin with a DevOps Backlog
- **8** . . . . 2. Build your DevOps Capability Matrix
- **9** . . . . 3. Start small & grow with the Crawl-Walk-Run Model
- **10** . . . 4. Remember, One Size does NOT Fit All

11 **Measuring Success**

Quantitative Key Metrics You Should Track

There's More to DevOps than Plain Numbers

16 **Still Not Convinced?**

18 **Your Essential Action List**

19 **Have Questions, or Want to Learn More?**

20 **Sources & Further Reading**

Speed, Stability, and a Culture of Innovation

The Power of DevOps

DevOps is a powerful tool that promises speed, stability, and a drive for continuous improvement and innovation. It's workflow, technology, and culture wrapped into one.

Just how fast? Try **106% faster**, from lead time to deploy, with more reliable results, too (*The State of DevOps Report, 2019*).

You've likely heard of DevOps before and may already be implementing it within your organization. Whether you're already familiar with it or just dipping your toe in the water, this guide is here to help walk you through a broader scope, how to make the most of DevOps, and why it's important to reach your full DevOps potential.

After a high-level overview that delves into all DevOps can do when approached through a 360 degree lens, this guide walks you through actionable ways that you can improve DevOps practices within your organization—culturally, and by employing a backlog, a capability matrix, the Craw-Walk-Run Model, and gauging key metrics. Finally, you'll come to the Action List that outlines your next steps towards a smarter DevOps.

It's common sense for anyone in software.

Today, every company is a software company. In the digital era, delivering effective, up-to-date products to customers is critical for businesses. From web clients for consumers to applications that enable company employees to do their jobs efficiently, software products drive your company's success.

In a world that evolves at breakneck speeds, you cannot afford to wait for long software release cycles.

Getting new, relevant features in front of users quickly is key in gaining an edge over competitors. DevOps is all about doing exactly that: getting software products to market quickly. Not only does a more effective practice bring speed to your software releases, but it contributes to stability and confidence in what is released, as well.

At its best, DevOps encompasses the whole of your IT department. It relies on efficient product management to discover software product features, and teams that get things in front of your users as quickly as possible. Time is spent innovating, experimenting, testing, and problem-solving in a continuous flow. Not a moment goes to waste.

Consider COVID-19

The business landscape changed almost overnight. Educational institutions and offices alike were forced to migrate to an entirely digital environment. In acknowledgment of this paradigm shift, products like ZOOM and Microsoft Teams excelled, adding new features to support a remote workforce, online education, & other pandemic-related communication.

However, many institutions that struggled to adapt to the rapid change, such as schools that suffered record-high absence rates while attempting distance learning, found themselves unable to thrive in lieu of the crisis *(Virus Forced Schools Online, but Many Students Didn't Follow).* DevOps enables quick action that can make or break an organization in such a fast-paced world.

The DevOps Way

DevOps isn't just a workflow; it's a mindset.

DevOps is not just about upgrading to better tools or a streamlined workflow; this is about a culture and attitude that spans all areas of software creation and implementation within your organization.

While it can help with specific products a team is working on, DevOps also has the power to streamline your entire IT department. By automating repetitive tasks, like tickets and changes, developers are able to spend their time on what they're passionate about: writing great code for challenging projects—ones that contribute to business value. The results are resilient, manageable, and observable, creating high-impact changes and frequent releases.

So, What Does the Best of DevOps in Action Look Like?

Per the 2022 State of DevOps Report, "The use of cloud computing has a positive impact on overall organizational performance. Respondents that used cloud were 14% more likely to exceed in organizational performance goals than their non-cloud using peers." There is a strong correlation between high performance organizations and the use of cloud native technology. The best tech of today and the standard of tomorrow lies in the cloud. To make the most of DevOps and compete with top industry performers, it's a foundational necessity.

The Tenants of DevOps

Flow

From beginning to end, flow is increased by removing constraints and reducing WIP.

Feedback

Create tight feedback loops with automation and testing to catch errors quickly and continuously correct them.

Continuous Improvement

Adopt an attitude of experimentation and practice where it's encouraged to take risks, learn from mistakes, and hone skills for mastery.

What are the benefits of cloud native technology?

- Cloud native technology empowers organizations to build and run scalable applications in modern, dynamic environments. It employs containers, service meshes, microservices, immutable infrastructure, and declarative APIs.
- It enables loosely coupled systems that are resilient, manageable, and observable, allowing high-impact changes frequently
- It fosters and sustains an ecosystem of open source, vendor-neutral projects

(CNCF via Github)

The Top-Performer Cloud Native Toolkit

Cloud native is critical because it allows you to leverage a variety of tools that take your organization's technology to the cutting edge. There is a vast landscape of programs associated with every phase of cloud-enabled DevOps. While the tools used to perform operations using a DevOps model are vital to execution, they are only a piece of the puzzle. However, it's worth noting that the technologies you'll need will fulfill a set of roles:

Microservice Architectures break down apps into bite-size pieces so that multiple team members can work on different pieces of a product at the same time.

Containerizing Traditional Apps means packaging up legacy programs so that they can function in modern environments and play-nice with other apps.

Containers & Kubernetes allow loosely coupled systems to work together seamlessly.

Continuous Integration & Deployment (CI/CD) bring testing and automation into your deployment processes so that you're never left waiting on manual, error-prone results.

Infrastructure-as-Code (IaC) Platforms ensure your systems are readily portable and adaptable by using declarative scripting.

The Goals & Gotchas of DevOps Initiatives

By now, most organizations have dipped their toe in the DevOps waters. In most cases the results come up short of their expectations, not to mention the cost of tools, staff and turnover eating up much of the ROI. While many struggle to get started, some organizations are emerging and seeing results.

Over the last 5 years the nvisia DevOps team, in partnership with enterprises large and small, has had a front row seat to many organizational journeys. We're sharing our lessons learned in this blog series, but first, it's important to understand the business value DevOps can bring.

Reliability is Key

According to the Accelerate State of DevOps 2022 *"use of hybrid and multi-cloud (and private) seems to have a negative impact on software delivery performance indicators – MTTR, lead-time, and deployment frequency – unless respondents had high levels of reliability."*

Reliability is critical to the success of your cloud endeavors.

The "why" behind DevOps initiatives

It is hard to deny the results from high performing DevOps organizations. The Accelerate State of DevOps Reports provided by DORA (acquired by Google in 2018), provides an excellent summary of the difference between low and elite performers. From the Google DevOps site:

Speed of deployments

*The best teams deploy **973x more frequently** and have lead times **6750x faster** when compared to low performers.*

Stability of your software

*High performers don't trade off speed and stability. The best teams recover from incidents 6570x faster and have **change fail rates 3x lower.***

Security in from the start

*High performers spend **50% less time fixing security issues** compared to low performers.*

What does this really mean? At the end of the day, Elite DevOps performers are able to:

- Commit code to source code control and deploy to production in less than 1 hour

(The rest of the list can be found on the next page!)

- Perform 2 production deployments in a single day
- Recover (MTTR – mean time to recover) from deployment problems in less than 1 hour
- Deploy at will with 0-15% Failure Rate

Each one of these items saves valuable time of highly paid professionals, reduces costly errors and increases the stability and quality of your production systems.

To be successful in a DevOps initiative, you should measure these sorts of business metrics, understand where your IT teams are experiencing problems, and make efforts to improve the numbers. Installing tools does not magically make you a master of DevOps. Like any well-conceived initiative, a DevOps initiative should target business problems and success must be tied to some measurable outcomes.

Common DevOps Anti-Patterns

As we've engaged with struggling organizations, we've noticed some patterns, or really anti-patterns, that led to their issues. For brevity's sake, here are the most common along with some simple recommendations.

Centralized Operations Team

(Rebrand Jenkins Operations team as DevOps)

Many operations teams own central tools like databases and old-school Jenkins pipelines. These teams are understandably looking for relevance in the new distributed, cloud native world. In the new world functional standards and specification for tooling may still lie in the traditional central group, but the running the tools are now distributed within the application team's landing zones.

Recommendation

Instead of paving the cow path, establish a (cloud) platform engineering capability, responsible for ensuring the consistent deployment of the centralized teams standards and specifications, using infrastructure as code (IaC) and automated pipelines.

Science projects in production

(The rockstar dev teams that turned GhostOps)

Many early DevOps successes in larger organizations were born from high profile application development teams staffed by highly-skilled, passionate technologists with incredible grit. Sounds great, but since very little thought was given to day-2 operations, it becomes unsustainable. Subsequently, team members either burn out or bounce to the next cool team, leaving the application owners in a bad place.

Recommendation

Infrastructure as code and automation pipelines to declaratively manage your infrastructure are good insurance policies to prevent GhostOps.

Lead with tools

(There's no problem that a tool vendor can't fix)

While DevOps related tools are perfectly necessary for things like automation, deployments, security and observability, you need collaborate with actual application teams through an evolutionary process to solicit their input and establish the context of tool usage.

Recommendation

Remember that "DevOps is an organizational and cultural movement that aims to increase software delivery velocity, improve service reliability, and build shared ownership among software stakeholders" *(Google Cloud DevOps)*.

Many DevOps problems can be avoided altogether with agile implementation and design, informed by the DevOps way(s).

The "DevOps Team"

(New function = new team)

While it seem reasonable to create a new DevOps team to support DevOps capability, this notion is a tool oriented approach to find a place for the Pipeline tools to live. DevOps is way more than Jenkins, Gitlab, Github or Azure DevOps.

Recommendation

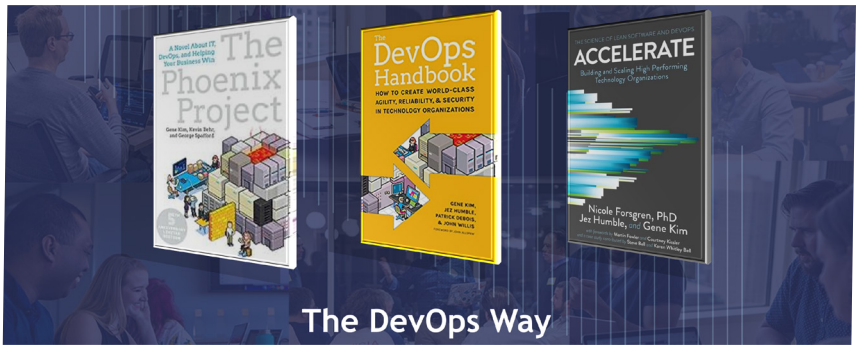
Again, **DevOps is an organization and cultural movement that spans all of your technology organization**. It can't be neatly tucked into a single organizational silo. Explore a matrixed platform engineering capability instead.

DevOps is way more than tools

While many attempts have been made to articulate this point more politely, rarely has it been stated less ambiguously than this quote:

> *"Stop talking about the blinky shit! ...it's about the people and the culture"*
>
> **– Chris Roberts, DevSecOps Expert Panelist**

DevOps culture, with the core concepts built upon lean manufacturing (aka lean software) concepts, is beautifully portrayed in this legendary series of MUST, MUST READ books. *The Phoenix Project* is a fictional account of DevOps in a gripping story masterfully designed to get you thinking about DevOps concepts. If you have been in IT for any length of time, you will immediately relate to most of the characters in this gripping tale. Next is the *DevOps Handbook*, the non-fictional guide to getting started with DevOps. Finally, we have *Accelerate*, research associated with organizations who adopt DevOps practices and culture.



DevOps journeys are fraught with challenges, but deliver real business value to the organizations that take a smart approach to them.

Rockin' Cloud Native DevOps

Blending DevOps culture with Cloud Native Technology and Platform Engineering is a great way to boost your DevOps initiative to new heights! The nvisia DevOps team, in partnership with enterprises large and small, has had a front row seat on many DevOps adoption journeys. Now we're revealing our "secret sauce" – tips, tricks and hacks to help those who are striving to become elite performers.

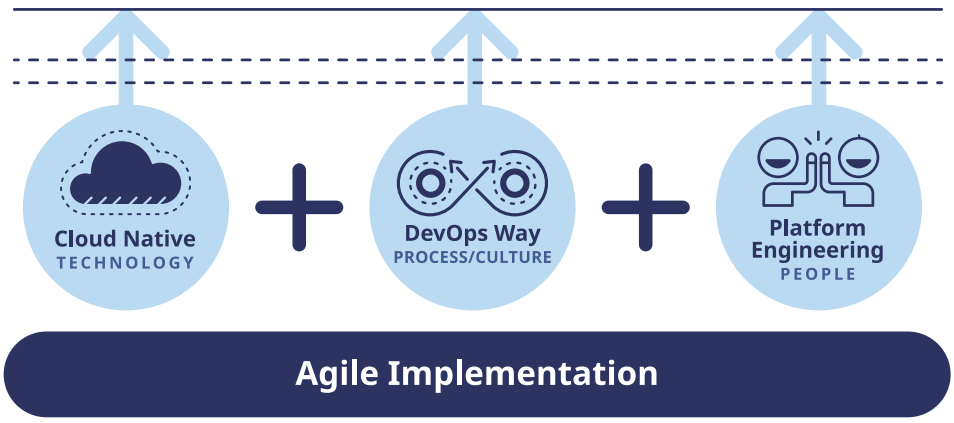
#1 Adopting the 3+1 View of Cloud Native DevOps

Many organizations make the mistake of thinking DevOps is done after they establish a few CI/CD pipelines. Stopping there leaves a lot on the table. In fact, we dare to say those organizations missed the point of DevOps.

While DevOps culture and practices are a critical component of succeeding in a cloud native DevOps world, they are not the only thing. To get "elite performer" results, DevOps needs to be blended with other enabling aspects into a broader initiative that, like any great enterprise evolution, can be presented in terms of people, process and technology.

So for our first tip, the nvisia DevOps Team presents the 3+1 View of Cloud Native DevOps as shown below. We use this approach to re-frame our client conversations regarding DevOps adoption, assessments and roadmaps. Here we blend aspects of Cloud Native (Technology) with DevOps culture (Process) and Platform Engineering capabilities (People) into an Agile implementation to quickly establish flow to deliver value in each sprint.

3+1 View of Cloud Native DevOps



Each of the pillars need to be raised together to achieve meaningful capability improvement within an enterprise and move the needle on specific business goals. Please note the +1 Agile Implementation at the base of the diagram. Agile adoption is the lynchpin for an organization moving from knowing to fully understanding Cloud Native DevOps. You can't fake it forever – every sprint you dig deeper and understand more.

#2 Agile Adoption of Cloud Native DevOps

This means organizing work across enterprise departments as epics and features, then grooming a backlog of prioritized stories/tasks for each sprint. This can be really hard. Keep in mind that each sprint needs to measurably move you toward achieving a new or improved capability that brings business value. For example, a business goal may dictate that experiments need to be run every week to support product management. That means that you need to be able to release at least once per week without impacting the end user's experience. So, what are you going to do in each sprint to get the organization closer that goal?

Depending on your Agile process skills and maturity, making this work can be more or less challenging. As a bit of inspiration, we provided a partial sample of a Capability/Goal matrix below. For each of the Capabilities shown, we list some key business goals and a definition of done. Another really important consideration is the cross functional participation required to be successful. *For more on that, please see the next tip.*

#3 Make sure all the seats are set at the Cloud Native DevOps table

From the beginning, identify all of the seats at the table (RACI style) + Tools adoption and training related to each area. Make sure these areas are represented during each sprint so the organization can share the journey with a common purpose. Otherwise, most folks will try to do too much at once and step on each others toes. The main purpose is to create a cross functional team that will start to punch holes in the silos and eventually knock them down. Below is a partial view of a capability planning tool we use at nvisia to align goals and roles across the enterprise. It can be helpful to identify epics and plan a series of sprints.

Sample High Level Responsibility Chart			
Capability/ Goal	**Local Containerized Development** • Faster Dev Onboarding • Better Dev Innovation – Happier developers • Incremental Upgrades • Immutable Containers (exactly like Prod)	**Containerized Application Architecture Pattern For App Stack (Java, Node, .Net Core, etc)** • Faster time to features • Strong Security posture	**Containerized Platforms for Production workloads** • Fast, stable anytime code deployments • Secure, observable workloads • Complaint, auditable deployments with full traceability
Done When...	*Developers can clone, update, locally build, test and push code that deploys to Dev Cluster.*	*Developers pull starter applications that use approved base images, all images are scanned before pushing.*	*Releases are easy, secure, verifiable, and don't (usually) require service outages*
Dev Team	**Training/Use:** • Docker (for Devs) • Local development cycle and tools • Unit Test run /published • GitFlow(ish) Branch and Merge **Accountable:** • Feedback to Ops and Platform team • Following process • Improving dev tools	**Trained/Use:** • Approved Docker Files • Containerized application standards • Sample App Content • Code quality scanner • Vulnerability remediation **Accountable:** • Zero critical vulnerabilities • Improving app code quality (Team and Sample Apps)	**Trained/Use:** • Application health and performance instrumentation • APM tools to evaluate deployments • SRE function to focus on user experience • Data Classification **Accountable:** • Production Application monitoring and troubleshooting • Data protection and audits

Platform Engineering Team	**Trained/Use:**
	• Intro to Azure DevOps
	  • Repos
	  • Pipelines
	• Intro to Terraform
	• Docker (P.E. focused)
	• Image lifecycles and tools
	• Intro to Kubernetes for Admins
	• Introduction to Istio
	Accountable:
	• Deploying Dev Infrastructure as Code
	• Push triggers
	• Merge policies
	• Pipeline permissions
	• Pipelines for build, push and release into develop namespace)
		Trained/Use:
		• Scanning tools
		• Image lifecycle policies
		  • Smoke tests
		  • Vulnerability Thresholds
		  • Code Quality thresholds
		Accountable:
		• Deploying and testing sample applications
		• Pipelines to support build and release of Golden images
		• Add vulnerability scanning to
			Trained/Use:
			• Centralized logging, monitoring and alerting tools
			• Dynamic security scanning tools
			• Compliance controls and auditing
			• Policy management for IaC and K8s
			Accountable:
			• Deployment Staging and Prod env as Code
			• Implementing baseline security spec
			• Deploying agents for Security/Cost/ Compliance/Logging Tools as Code

Security	**Trained/Use:**
	• Azure DevOps
	  • Permission groups/AD integration
	  • Repo Permission and Policy management
	  • Cross Repo permissions
	  • Pipeline permissions and service accounts
	  • Azure integration
	  • Private network access
	  • Private build agent
	  • Service connections
	Responsible:
	• Pipeline controls
	  • SCC policy enforcement
	  • Service connections access control for Dev Resources
	• Handing of local dev Secrets
		Trained/Use:
		• Docker Fundamentals (Build/ Buildkit)
		• Docker: Security for Images/ Repos
		• SBOMs
		• Docker Image Scans and Remediation
		• Docker base image pipelines
		Accountable:
		• Golden Base Images
		• Container Scanning
		• Static code scanning
		• Lint Rules
			Trained/Use:
			• Cloud Security (i.e., Wiz.io)
			• NIST Container/Host specs
			• Scanning tool policy management
			• Policy management tools
			• Kubernetes Fundamentals
			  • Certficate management
			  • Secret Management
			  • MTLS w/in cluster
			Responsible:
			• Platform hardening standards (NIST Special Publication 800-190)
			• Dynamic code scanning policies (Fail/Allow)
			• Endpoint scanning (i.e., Crowdstrike)
			• Policy definition for K8s (IPA/ Gatekeeper)
			• Policy for App LZ (i.e, Azure Policy)
			• Secret management/lifecycle (CSI integration)
			• Choose and procure Cloud Security Tool (like Wiz.io)

Sample High Level Responsibility Chart			
Capability/ Goal	**Local Containerized Development** • Faster Dev Onboarding • Better Dev Innovation – Happier developers • Incremental Upgrades • Immutable Containers (exactly like Prod)	**Containerized Application Architecture Pattern For App Stack (Java, Node, .Net Core, etc)** • Faster time to features • Strong Security posture	**Containerized Platforms for Production workloads** • Fast, stable anytime code deployments • Secure, observable workloads • Complaint, auditable deployments with full traceability
Done When...	*Developers can clone, update, locally build, test and push code that deploys to Dev Cluster.*	*Developers pull starter applications that use approved base images, all images are scanned before pushing.*	*Releases are easy, secure, verifiable, and don't (usually) require service outages*
Operations	**Responsible:** • Dev Cluster resources • Networking Resources • IAM managed RBAC/ABAC to shared dev resources **Accountable:** • Cloud accounts/ subscription • Service Principals or Account Roles	**Accountable:** • Container Repository w/ vulnerability and License scanning • Local Desktops that Container dev ready	**Responsible:** • Cost monitoring • Resource consumption • Network design and controls • DNS, Certificates and Secrets

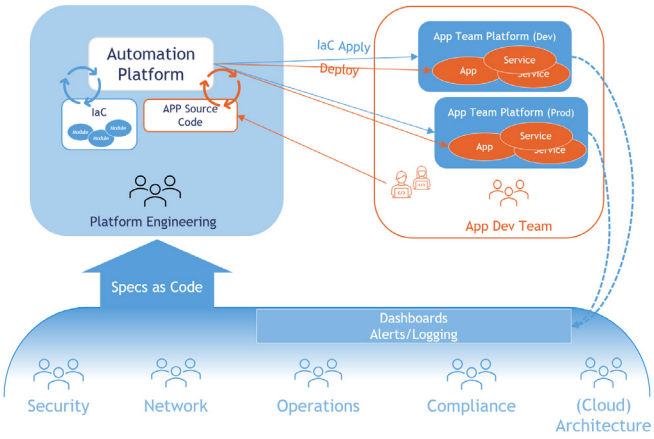
Compliance and Risk	**Responsible:**
	• SOD of application promotion process
	• SCC policies
	• Pipeline controls
	  • Reviews
	  • Approvals
	  • Policy Checks
		Responsible:
		• Local dev certificate management
		• Local DNS
		• Container promotion separation of duties
			Responsible:
			• SOD in Production Clusters
			• Policy/control audits monitoring

Enterprise/ Cloud Architecture	**Responsible:** • Select/recommend architecture patterns • Supporting tools/configurations • Process for application delivery pipeline for Dev	**Responsible:** • Defining, assembling architectural for containerized services by stacks • Service discovery • TLS termination • Externalizing env variables, and managing secrets	**Responsible:** • Overall cloud Hub and Spoke architecture • Application Landing Zone Architype
IAM	**Accountable:** • Enabling Service Connections to Dev Spoke resources by Assigning Service Principal Permissions or Account Roles Policies • Assigning RBAC for Dev Spoke AD groups (Admin and Dev)	**Accountable:** • Role mapping for SSO tools • Assigning RBAC to SSO tools	**Accountable:** • Enabling Service Connections to Staging and Prod Spoke resources by assigning Service Principal Permissions or Account Roles Policies • Assigning RBAC for Staging and Prod Spoke AD groups (Admin and Dev)

#4 Establish (cloud) platform engineering capabilities

Establish platform engineering capabilities as the hub between all enterprise functions. Generally, the platform engineering capability exists to implement reusable tools and self-service capabilities with automated infrastructure operations (IaC and Pipelines) to improve the developer experience and productivity. The enterprise benefit is fast deployment of reference architectures using standardized tools, components and automated processes.

In this model, specifications are realized as code and consistently applied to all application team landing zones. Hence, this group is responsible for collaborating with Security, Networking, Operations, Compliance/Risk, and Enterprise (cloud) architecture to evolve the enterprise posture of applications and also to make compliance/performance of such specifications observable to the same constituents. Application teams can shift left by using the approved resources from the platform engineering team.



#5 Look for accelerators

Tap into experienced resources who have been involved in a journey to cloud native DevOps. These could be pockets of resources within your existing organization (maybe from acquisitions) or external experts like nvisia. Additionally, look for accelerators (detailed working samples that often include IaC) to lift cloud architecture and cloud native technology. While accelerators should be sought to inform your perspective and educate your teams, be very careful about going all in on any particular one. Many are academic or may lock you into a vendor solution.

At nvisia, we have developed a practical proven variation of the Cloud Adoption Framework (CAF) accelerators for both Azure and AWS. We call it Digital Foundations and use it to get our clients up and running faster with less rework. Bringing a prebuilt library of proven components (IaC code and pipelines) to realize architectural best practices can reduce timelines from years to months!

Conclusion

Cloud adoption journeys are filled with promise and challenge. Look to build a strong Platform Engineering team to own the effort, rely on infrastructure as code, and move to a hub-and-spoke model to enable teams, without constraining them.

Still Not Convinced?

A quick look at what the data has to say.

"Delivering software quickly, reliably, and safely is at the heart of technology transformation & organizational performance... including profitability, productivity, & customer satisfaction."

State of DevOps Report, 2019

The highest performers are two times as likely to meet or surpass organizational performance goals – odds that are in your favor. DevOps isn't just a new way of doing things; it's a method that carries business value and allows your company to respond at lightning speed to the ever-changing tech landscape.

If numbers speak to you more than words,

top performance with DevOps brings some astonishingly positive metrics. Per T*he State of Devops Report, 2019,* when comparing elite performers with low performers, the elite group has:

208x

more frequent code deployments

106x

faster lead time from commit to deploy

2604x

faster time to recover from incidents

7x

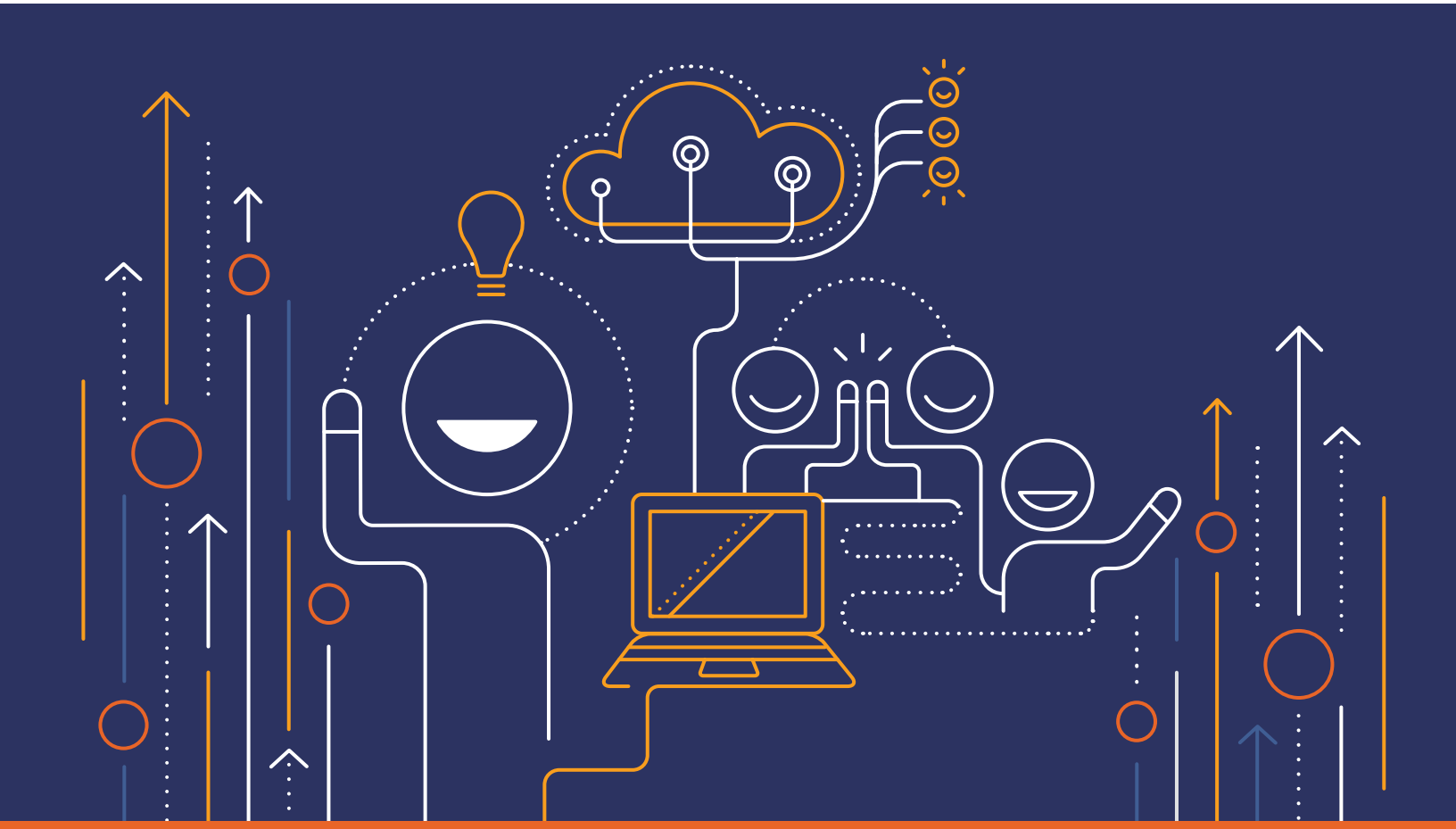
lower change failure rate (i.e 1/7 as likely to fail)

Beyond numbers, studies indicate that a combination of a responsive environment and perceived self-ability result in greater performance and self-confidence (*Druckman & Bjork*). By promoting a culture of flow, feedback, and continuous improvement that is reflected in technological processes, individuals feel like they are in an environment that supports them and values their success. Through learning, they gain valuable skills and improve their

performance, boosting confidence and willingness to innovate. This ultimately makes for happier employees and a healthier bottom line.

> ***"Industry velocity is increasing.*** *Many analysts are reporting the industry has 'crossed the chasm' with regards to DevOps and technology transformation, and our analysis this year confirms these observations. Industry velocity is increasing and speed and stability are both possible, with shifts to cloud technologies fueling this acceleration. This reaffirms the importance of technology that enables organization to deliver value to their stakeholders."*
>
> **State of DevOps Report, 2019**

In other words, **DevOps is critical to ensuring that your business can compete with and surpass others in your industry.** Falling behind now will cause concern in the short-term and stagnation in the long-term. By implementing smarter DevOps practices, your organization will see results reflective of those described above, prevent lethargy, and bring speed and optimized performance that provide a crucial competitive edge.



Your Smarter DevOps

Essential Action List

- **Assess where you're at** by examining key metrics. Pay attention to lead time, deployment frequency, change fails, and time to restore.
- **Create an agile backlog** of to-do and wish list items across your IT department. Look for opportunities for automation and increased communication.
- **Build your DevOps Capability Matrix** with user stories from the backlog.
- **Foster an environment** of communication and trust.
- **Define a TVP** for your first crawl sprint.
- **Complete a crawl sprint** that results in real code running in a cluster. Make it happen in less than two weeks.
- **Perform crawl and walk sprints** until you're ready to run.
- **Sprint, sprint, and sprint again,** pacing yourself while chipping away at the backlog.
- **Look back at your key metrics** to see how far you've come.

And with that, you've arrived at a smarter DevOps.

Have questions
or want to learn more?

The path to a smarter DevOps is attainable to all who set their mind to it, but made easier with experts who can guide you along the way. **nvisia** is proud to guide organizations, from small-scale to global enterprises, on that journey with tested knowledge and expertise.

Talk DevOps with nvisia.

Learn more at **nvisia.com.**

learn@nvisia.com

nvisia

Sources

nvisia Experts

Mark Panthofer
VP, DevOps & Cloud

Tim Liebl
Project Lead

Nick Schultz
Director

James Leiser
Sr. Technical Architect

Additional Sources

Deepti Suri

Director, Cloud Infrastructure DevOps, Automation, and DBA Services at Foot Locker

Cloud Native Definition v1.0

by CNCF, via GitHub

State of DevOps Report 2019

by DORA & Google Cloud

The Principles Underpinning DevOps

by Gene Kim

"Self-Confidence and Performance"

from Learning, Remembering, Believing: Enhancing Human Performance, pp. 170-175. Edited by Daniel Druckman and Robert A. Bjork, published by the National Research Council

Virus forced schools online, but many students didn't follow.

by Julie Watson and Carolyn Thompson, Associated Press for ABC News

Further Reading

The Phoenix Project

By Gene Kim, Kevin Behr, and George Spafford

A fiction novel that shows DevOps in action through stellar storytelling. It communicates the mindset and culture that surrounds DevOps.

The DevOps Handbook

By Gene Kim, Jez Humble, Patrick Debois, and John Willis

A non-fiction, dry equivalent of the Phoenix Project. It's a great toolkit but lacks a comprehensive framework to bring all the pieces together.

Measure What Matters

By John Doerr

This manual guides and informs how to bring DevOps into practice, helping set expectations, answer questions about what to do next, and how to measure results.

Team Topolgies: Organizing Business and Technology Teams for Fast Flow

By Matthew Skelton and Manuel Pais

IT consultants Skelton and Pais share an array of successful team patterns to help audiences choose the right DevOps topologies for their organization, keeping software healthy, teams happy, and business value growing as a result.

Thanks
for reading.

For more insights, visit nvisia.com/insights.

nvisia
connect. build. enable.