Gearset SILVERLINE

Deciphering DevSecOps

for Salesforce Teams

Over the last few years, many Salesforce teams have matured their DevOps processes so they can collaborate more easily, manage multiple workstreams, and deliver projects quickly and incrementally. Ultimately, DevOps enables teams to get more value out of Salesforce for their businesses.

As DevOps best practices become more familiar to the Salesforce ecosystem, some teams are beginning to wonder about DevSecOps — a concept that has been trending in the wider world of software development.

What is DevOps?

To understand DevSecOps, we have to begin with DevOps itself.

DevOps is a set of processes and tools that teams use to streamline their development workflows. But the fundamental problem that DevOps aims to solve is related to people. Specifically, DevOps is designed to break down the silos in which developer and operation teams work. For example, in the world of Salesforce development, DevOps helps to bridge the divide between developers and admins. It brings teams together so everyone can play a part in the development and release process, end to end.

For some, DevOps is just another word for release management, but it really describes a more specific approach, characterized by close collaboration, agile development, and shorter release cycles. DevOps helps you:

- Get finished work to end users sooner
- Tighten the feedback loop and encourage user-driven development
- Debug releases more easily

High-performing Salesforce teams release faster to deliver more business value

Survey results show DevOps has a positive impact on businesses, both in terms of time savings and faster rates of delivery. Of those surveyed, 52% of respondents are now able to deploy in less than one hour. Elite and high-performing teams are forging ahead with high-frequency releases, short deployment times, and low change failure rates as a result of implementing DevOps processes. Teams who haven't made the switch to DevOps yet are at real risk of losing out on the clear financial benefits and increased business agility achieved by establishing a high-performing DevOps process.

What is DevSecOps?

If DevOps is about bringing together the people, processes, and tools for development and operations, DevSecOps emphasizes the need to make sure security isn't left out. It's important to note that DevOps best practices already enhance security dramatically, and there's a sense in which DevSecOps is really just DevOps done well. But it's also true that many businesses still have people, processes, and tools for security that are siloed away from DevOps. Breaking down those silos creates greater value and accelerates team pace.

In the context of Salesforce, DevSecOps really means two things:

1. Making sure all the people whose roles involve developing, maintaining, and protecting the business's Salesforce environments can work together seamlessly, enhancing security as a result.

2. Making sure admins, developers, and other leaders in InfoSec (Information Security) and Compliance are educated on, contribute to, and reinforce the security protocols of the organization through use of DevOps tools and processes.

By placing security at the heart of the DevOps practice, not only can organizations innovate faster and release more often, they can more mindfully and intentionally follow best practices for security and compliance.

69%
of teams working on Salesforce releases say they're responsible for backups

31%
say they're not

Source: State of Salesforce DevOps 2022 survey

Our DevSecOps Playbook

Although Salesforce offers world-class tools and processes to help organizations on their digital transformation journeys, great security and collaboration depends on people. In our experience, many teams are new to the ideas presented in this eBook, and that's OK! In fact, some of the best DevSecOps processes we've seen have started with assembling a great team of cross-functional roles — and learning about best practices together. But first: who should have a seat at the table?

Designing Your Center of Excellence

Certainly, Salesforce architects, admins, and developers are instrumental in starting your DevSecOps initiative off on the right foot. Ultimately, responsibility for the success of the Salesforce platform lies in the hands of this core team, but it doesn't stop there. Ideally, you will have a clear, well-documented process and expectations for how to test new features, how to ensure they align with security requirements, and how you will ultimately release to users and get them trained.

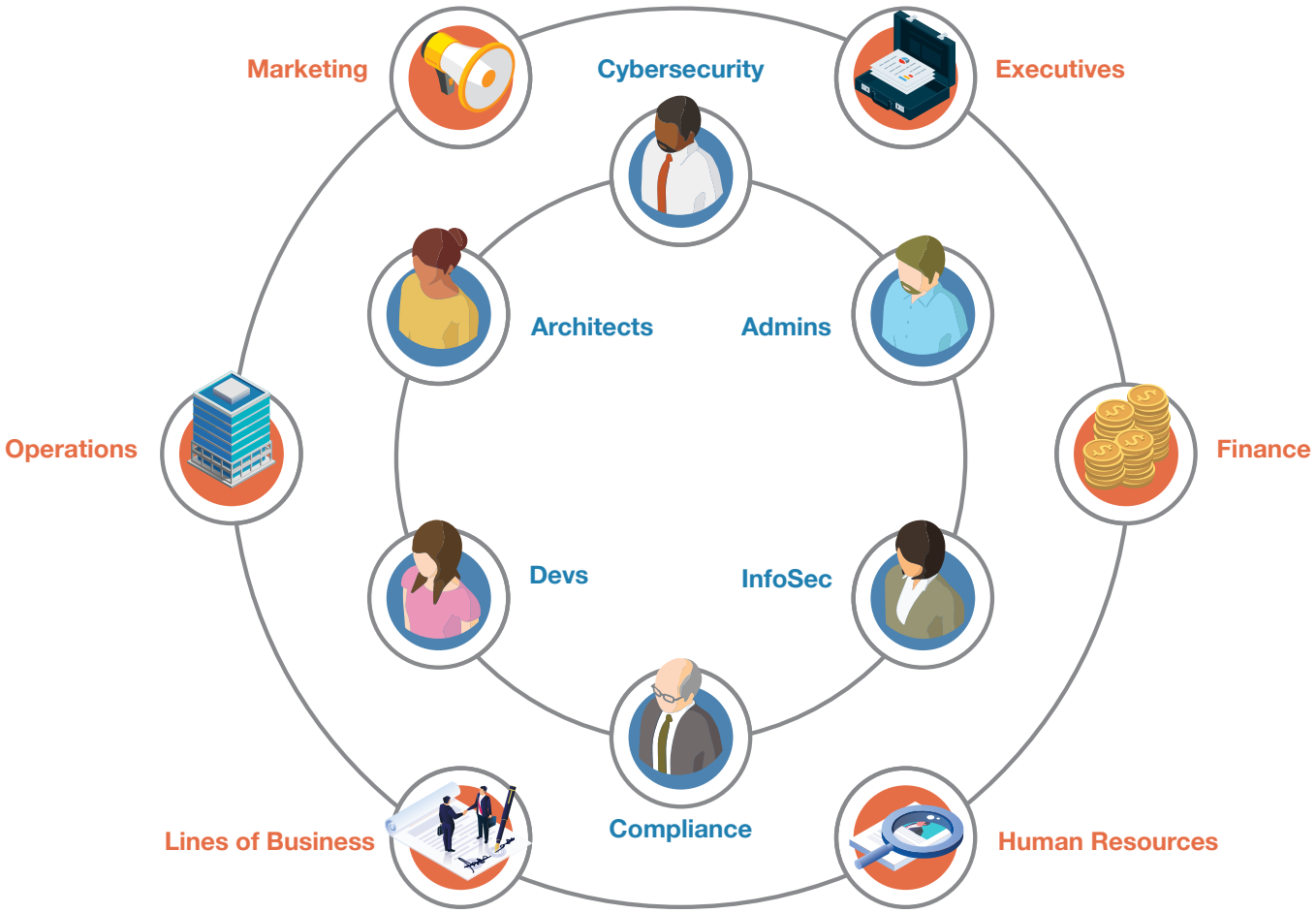
In addition to this core team, there are several other advisory roles that are important to include in a solid DevSecOps practice: InfoSec, Cybersecurity, and Compliance. Each of these roles offers different lenses into the threats your organization might face and how your systems might be vulnerable to potential breaches or data exfiltration by employees. According to the Identity Theft Research Center, there have been more than 12,000 data breaches since 2005!

In many companies, InfoSec and Compliance are handled by different teams, and sometimes there's a dedicated Security or CyberSecurity team that monitors and protects all their systems. Some smaller companies may blend these concepts together in IT.

Whatever your company's size, making an organizational commitment to security requires that all of these roles are informed and involved when your team is architecting and designing solutions. It's vital that InfoSec and CyberSecurity advise your Salesforce administrators and developers who aren't security experts, and Compliance should also be involved to ensure that all regulations — global and local — are taken into consideration early on.

With these key contributors to DevSecOps identified, the next step is to reinforce this commitment to security by establishing a DevOps Center of Excellence (COE), a clearly defined group of people who have accountability, responsibility, and governance around security needs. The core team should include the key roles above, and the COE should advise other system stakeholders and executives about upcoming changes to help reinforce policies across business lines. These stakeholders may include representation from shared services like marketing and operations, individual lines of business, and executives that need to be in the loop about the roadmap for security.

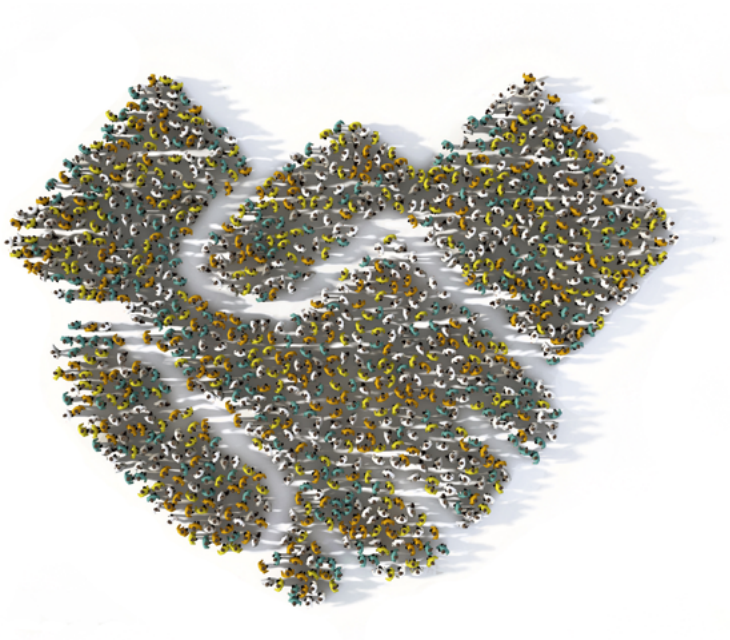
Centers of Excellence: The Stakeholders



Building Trust through Learning and Collaboration

Once you've identified who will manage, advise, and oversee your DevSecOps COE, it is important that they learn about and understand the breadth and depth of security features available on the Salesforce platform — as well as what the formal security policies of the organization require. If you're kicking off a newly minted team, this is a great opportunity to host a DevSecOps workshop.

The DevSecOps team should host regular forums for the COE to discuss the cross-functional nature of security and its impact, as well as develop a future-looking roadmap of items that need oversight and planning. This group can be a great forum for creating feedback loops about what's working, what's not, and what's coming next. Through a solid cadence of communication and trust-building, the DevSecOps COE will become instrumental in demonstrating the good-faith effort to follow security practices, and creating an organizational mindset that encourages everyone to take individual responsibility.



Establishing Your DevSecOps Charter

Once your COE team is established and everyone is educated on the security options and requirements of the organization, it's important to start with a clear charter for the group so that everyone is clear on what DevSecOps represents, and what the intent of your work together is. Creating a security-conscious culture requires vigilance on the part of all employees, as well as strong leadership, careful planning, and open and transparent communication across the organization.

Your company likely requires all employees to take regular security awareness training, so they understand how to handle extra sensitive PII (personally identifiable information) and PHI (protected health information). DevSecOps adds an additional operational dimension and a number of questions your COE should be prepared to answer and audit regularly:

- What are the security requirements we must follow?
- Where are the most risks in our development process today?
- What are we doing to address these risks?
- What's on our roadmap that might create new or different risks?
- How are we proactively planning for and monitoring these risks?
- What threat models do we face?

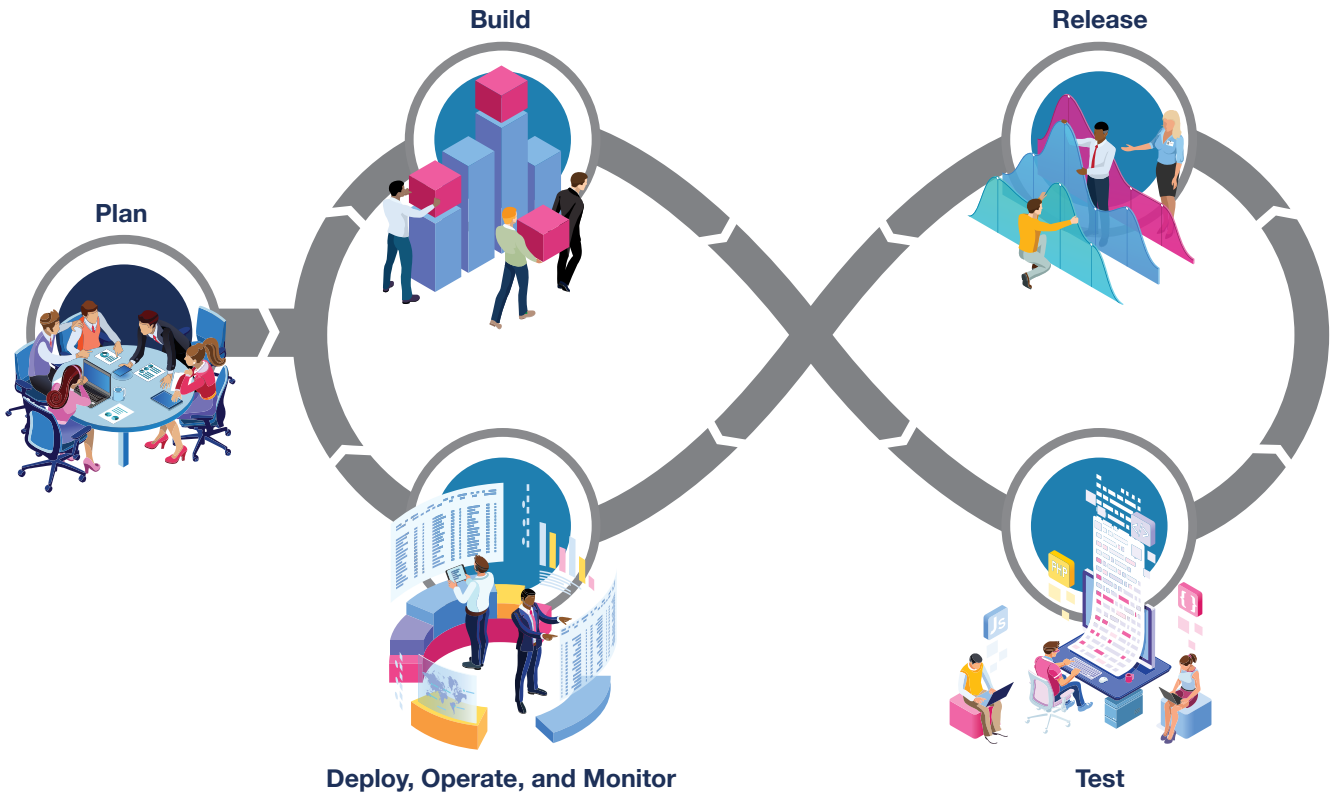
DevSecOps is all about embedding security into the development process from the beginning and engaging the team throughout every step of the cycle. In the past, we have seen some business stakeholders deprioritize security or treat it as an afterthought, given the choice between using budget on new features or addressing risk. But it's important that your COE understands the goal is to shift from a reactive security posture to a proactive one that is constantly evolving alongside your business. Security should feel like a partner to the business instead of an obstacle, and collaborate throughout the development process to ensure usability isn't unduly impacted by security requirements.

Defining How Your DevSecOps Process will Run

In addition to answering the questions above, your DevSecOps charter should define in detail how you expect your COE team to engage security throughout the development lifecycle. Putting these rules of engagement down on paper is a great way to get the cross-functional communication started — from requirements gathering and design, to development and testing, and all the way to training your users on how to take full advantage of new system features.

You may find it useful to organize your rules of engagement into the distinct phases below, with which many DevOps pros will no doubt be familiar. As you put your plan to paper, where should you start? On the following pages, we've outlined some great questions for your team to ask about each stage of your DevSecOps process.

DevSecOps: The Process



1. Plan

As you approach new roadmap initiatives, individual projects, or feature enhancements, how will you define and design for the threat models presented by the project? Is there a risk for data breach or data leakage, and how will you proactively prevent that from happening? What national or local security policies do you need to plan for as you develop your requirements? In our experience, these threat models may look similar for companies within an industry like Financial Services or Healthcare, so do your research and plan for these questions together!

2. Build

Once you define your requirements and have a build plan in place, it's important to look at your backlog and ensure you have made accommodations for all your security requirements. Do your backlog stories have acceptance criteria for the security needs presented by each feature? Have you embedded stories for permissions and system settings that might be impacted? In our experience, there are also different considerations for declarative features versus ones you might custom-build:

Custom Development

For custom-built features that might require classes, code, triggers, custom metadata, and more, how will you be reviewing that code for quality? Be sure to embed a manual code review in your development processes to make sure your developers are writing code that meets the security requirements you've outlined. Don't forget to inspect any open-source libraries you might be using, and any other code assets impacted by your custom development work to ensure they all meet muster. Finally, how might automated code scanning tools help your development process? In our experience, these types of tools can really help in your spot-checking and quality control efforts!

Declarative Development

Since so much can be accomplished with clicks and not code on the Salesforce platform, it's imperative that your team defines standards and best practices you expect in declarative features. As your Salesforce environment grows, it's important to include clear descriptions on fields and objects for how they're used. Be diligent about labeling your process builders and flows, and applying appropriate security settings to each and every new declarative feature you build. In addition to this manual effort, there are some great automated tools that can help you make sure the declarative configuration in your org is following security best practices like data classification on every field, profile and permission sets auditing, and data visibility controls that help surface the changing shape of data and configure smart alerts to warn of unusual changes.

Regardless of your custom or declarative development efforts, how will you plan to track the user behavior of your new app, release, or feature? In our experience with heavily regulated Financial Services and Healthcare companies, it's important to include items in your build backlog that help with employee behavior tracking and alerts on suspicious activity, especially if it might contain large data extracts, personally identifiable information, or other regulated activities like chat or support channels.

3. Test

Most DevOps teams are comfortable with embedding security reviews inside their existing QA (quality assurance) and integration testing processes already, but how can you take it to the next level? To evolve your DevSecOps posture as you approach testing for your new features and releases, consider these important emerging requirements:

Penetration Testing

This will become more prevalent as Salesforce apps become more client- and partner-facing. What will be your penetration testing plan as you expose apps to others outside of your collective VPNs and firewalls via communities and Experience Cloud, Lightning Web Components, custom mobile apps, chatbots, and more?

Static Application Security Testing

SAST is becoming more critical as Salesforce code bases grow in size and complexity and development teams rely more on external open-source libraries. Have you incorporated a static code analysis tool as part of your development process? Are you auditing your Apex, Visualforce, Lightning Components, and Javascript for the OWASP top 10, as well as Salesforce security best practices? We're big fans of Clayton and PMD, but there are many tools out there that can support this process.

Interactive Application Security Testing

There are increasingly advanced tools that can run in your browser to identify security issues within your Salesforce application at runtime. These will capture runtime vulnerabilities and analyze 3rd-party modules, such as managed package components that SAST does not include. Do you have a plan to incorporate them into your functional testing process and remediate the issues identified?

4. Release

As your teams move past the development and testing phases into your release plan, security should have a seat at the table! Auditing is a great topic to consider: How will you audit the trail of changes? How will you track who asked for the change and why the change was made? What does the change connect to and when was it done? The Salesforce audit trail has limited information about auditing and isn't alway easily reportable, so it's important you track changes carefully. Your audit and release log is a key component of your DevSecOps workflow, and tools like Confluence, Jira, and Git repositories can be instrumental to your release management operations.

5. Deploy, Operate, and Monitor

Finally, as your team releases features and functionality to your end users, there are many DevSecOps considerations to think about:

Training and Change Management

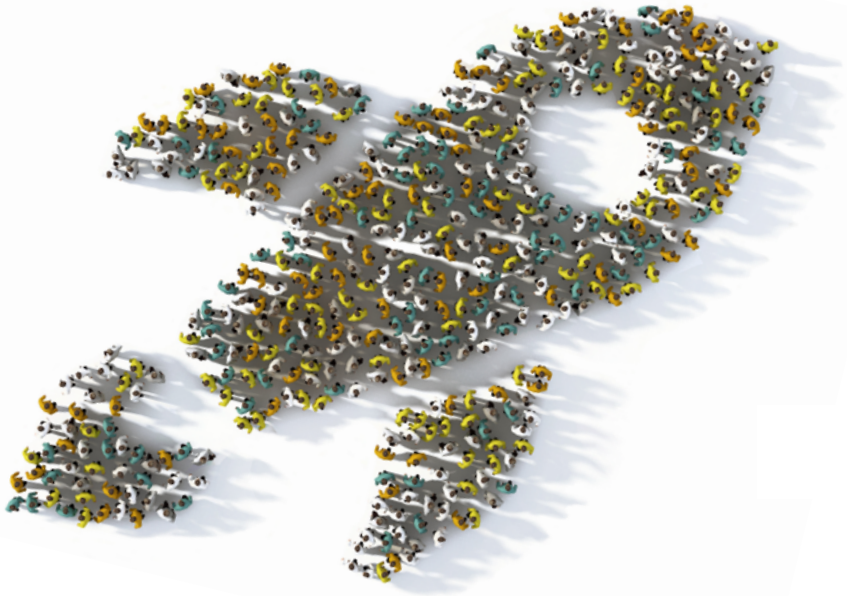
How will you educate and train your users about the security requirements and features in your new release? How will you monitor their usage and the uptake? What's in it for them and how will complying with these security requirements make their lives easier or better? It's important to communicate these benefits and expectations in your training efforts and release notes.

Backup and Restore Capabilities

In our experience, it's critical that you have your data and metadata backed up in an easily restorable way as you plan for your deployments … just in case. What are your plans to make sure these backups get created as part of your deployment? What tools will you use? Please note that some environments may require specialized tools — we can certainly help advise on what might be the best option for you!

System and User Monitoring

As you prepare for deployment and monitor the uptake of your new features and releases, event monitoring can be a very useful way to receive alerts and keep tabs on the situation. Salesforce's suite of Shield tools provides some nice dashboards, and many integrate to tools like FairWarning, Splunk, or NewRelic for application development monitoring and security monitoring needs, so that they can keep track of user behavior across systems.



Tools and Processes for DevSecOps

When deciding what tools and processes to adopt, you need to keep DevSecOps front of mind. If you evaluate the security risks in your existing or planned development and release process, you can make sure the right tools and processes are in place to mitigate these risks.

First, evaluate your process end to end. Different teams will identify different risks, and the likelihood or severity of those risks will also vary. But a typical development and release process will have vulnerabilities such as:

- Unauthorized metadata changes made in production
- Bugs and errors released to production
- Code failing silently in production due to breaking changes
- Sensitive data deployed to testing environments during sandbox seeding
- Data getting lost or corrupted in production due to user error, malicious actions, or faulty integrations
- Sensitive records exposed because permissions metadata gets corrupted
- Metadata being lost or corrupted due to a risky release

Next, consider what your process will look like as it evolves. Will new risks be introduced? Will the likelihood and severity of risk be increased? You need tools and processes that can scale with you to ensure continued and enhanced security and efficiency.

Having identified the risks, it's easier to establish what security measures you need and what DevSecOps will look like for your team. In general, you need tools and processes for the following:

Visibility

You need to see when anything is going wrong, so you can respond quickly and have confidence that things are secure the rest of the time.

Mitigation

The severity of most risks can be reduced. Some risks can be eliminated altogether.

Restoration

When a fault or incident occurs, you need tools and processes that help you recover quickly and reliably.

More specifically, here are seven key DevSecOps tools and processes that your team should consider.

Backup and restore

Backups are fundamental for security. But it's not obvious to every business that they need to back up their Salesforce orgs. Gearset's *State of Salesforce DevOps 2022* report shows that 27% of teams don't regularly back up their orgs, and 41% haven't implemented a backup solution for Salesforce data.

Putting in place a tried and tested backup and restoration solution for both Salesforce data and metadata should be a top priority for any team thinking about DevSecOps. Some teams manually back up data for peace of mind, but would struggle to restore that data in the event of severe data loss because they don't have backup metadata to restore the shape of the org first. Other teams have a mature DevOps process that effectively safeguards their metadata, but data backups are overlooked. Either way, these teams aren't prepared for disaster recovery.

Backups work best when they're woven into a broader DevSecOps process. The team responsible for managing releases should be able to back up production on demand before a risky release, without switching between tools. And when a team comes to restore lost data or corrupted metadata, using familiar deployment workflows to restore boosts confidence and accelerates time to recovery.

27%
of teams don't regularly back up their orgs

41%
say they haven't implemented a backup solution for Salesforce data

Source: Gearset's *State of Salesforce DevOps 2022 Report*

Monitoring and alerts

Monitoring tools give you full insight into the state of your org. Specifically, you need to keep track of the changes to data and metadata in production.

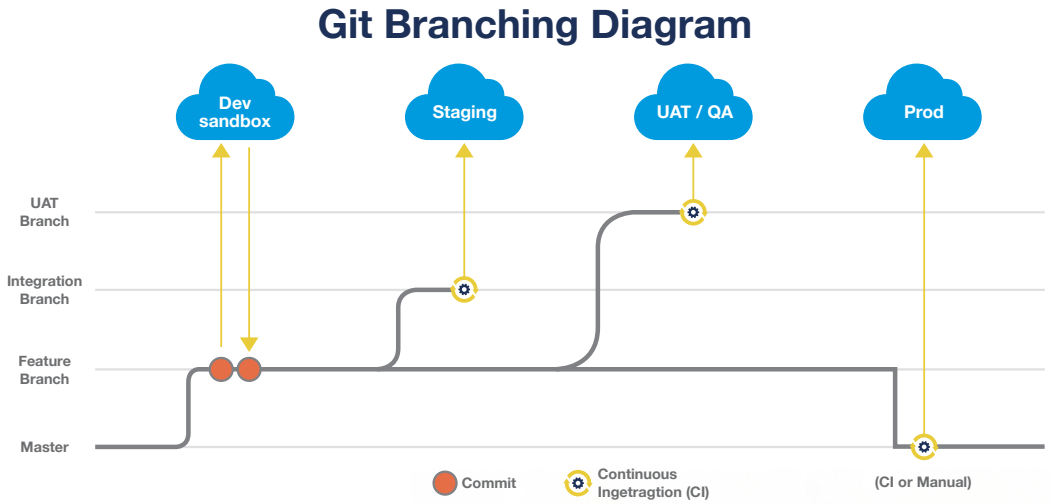
The ability to make declarative changes within Salesforce is one of its key strengths. But it's also a potential weakness if people continue making changes directly in production when their team has put in place a proper release management process. Metadata monitoring allows you to identify any such changes in production, making sure that everything is accounted for in your audit trail — even changes made outside the usual process. Problematic changes can be identified and removed, while hotfixes can be deployed back and synced with other environments, such as your Git repository if you're using version control.

Keeping an eye on the data in your org poses a different problem. Data changes all the time, so it's no good trying to keep track of every record that's added, changed, or deleted. But it is important to know when something unusual has happened, such as hundreds of contacts being deleted suddenly. A configurable system such as the smart alerts tool in Gearset's backup solution lets you specify what levels of data churn might indicate a security incident for your business.

Version control

Version control is fundamental to DevOps, and also brings significant security benefits. Git enables teams to collaborate effectively and enforces workflows that make the release process more robust. Instead of production being the single source of truth for your team, and so a single point of failure, the metadata in production is backed by a new source of truth: the main branch. The metadata in main should always be deployable so that your team can release the latest version of your metadata on demand.

A source-driven workflow should involve teammates reviewing each other's pull requests to merge new work into the main branch. This improves the quality of work and builds a culture of collaboration, helping teams to catch bugs and errors long before they're released to production. Git also automatically generates an audit trail of all pull requests and commits, which makes it easier to debug releases. Teams can see who made what change when and why. And, when the need arises, Git supports reverting merges and rollbacks to earlier states.



CI/CD

Once teams have adopted version control successfully and built a release pipeline around their chosen Git branching strategy, they're ready to begin automating parts of that pipeline. The aim is to automate repeatable steps for promoting work along the pipeline: the continuous integration (CI) of each developer's work into one environment, and the continuous deployment (CD) of all this work to environments such as QA or UAT (user acceptance testing).

Automation encourages short iterative processes and is therefore an essential part of DevSecOps, which is all about working in an agile way to accelerate your release cadence. Smaller, more frequent releases don't just get value into end users' hands sooner and shorten feedback loops. They also make it easier to recover from any issues because it's easier to debug a small release — there's less code and configuration to look at!

A lightning-fast release process also means teams can respond quickly when a bug is identified. Often the response will be to roll forward, getting the fix out in the next release and making that release as soon as possible. If that's not possible, teams need to roll back the release that broke things — something that's more straightforward with smaller releases.

Test automation

The purpose of testing is, of course, to improve work and reduce the likelihood that anything substandard or even dangerous is released. Folding test automation into your DevSecOps process usually means having tests run at several stages of the release cycle, which dramatically improves security.

Unit testing makes sure your Apex classes execute as intended. Testing code before it's released reduces the risk that faulty changes make it as far as production. But it's equally as important that you continue testing the code already in production. If code begins to fail silently, you'll only notice once it begins to cause damage. Automated unit testing of all the Apex classes in your org allows you to monitor these tests and rectify code after a breaking change. It also tracks your code coverage, so you never fall below the 75% threshold at which Salesforce will no longer allow you to deploy. Remember, it's a key principle of DevSecOps that you should be able to release on demand.

Static code analysis, another form of testing that can be automated, improves code quality. It doesn't execute your Apex classes to observe the output, but searches your code for anything that your team has agreed to avoid. Rulesets for code quality include rules ensuring the security of code. For instance, PMD includes a rule that checks redirects in an Apex class are not made to user-controlled locations, preventing attackers from redirecting users to phishing sites.

Data masking

In addition to their automated tests, most teams manually test that their changes behave as intended by deploying them to a QA, UAT, or staging environment and verifying their behavior. Usually this environment is a full or partial sandbox — a replica of production in terms of metadata and perhaps also data. Data is vital for the most robust testing. The quirks found in real-life datasets provide the edge cases that might cause a new feature to fail, which is why teams often want production data in their testing environment.

The security implications of this are clear: copying sensitive data, perhaps including personally identifiable information, to another org doubles the risk of unauthorized exposure. Data masking solves this dilemma by allowing you to obscure the production data you deploy to testing environments, so you can carry out robust testing procedures without compromising on security or breaching compliance requirements.

Everything depends on deployment success

For Salesforce, specifically, none of the above can be achieved until deployments are running quickly and reliably. Successful deployments are the foundation on which DevSecOps stands. Since the platform-native tooling for deployments has a success rate of approximately 50% and can take teams upwards of 10 hours for each deployment, teams practicing DevSecOps must use other deployment tools.

Solving the problem of slow and error-prone deployments unlocks the door to DevSecOps, and third-party solutions such as Gearset are being used by thousands of teams to build modern workflows that incorporate version control, automated testing, CI/CD, and more.

Since the platform-native tooling for deployments has a success rate of approximately

50%

and can take teams upwards of 10 hours for each deployment, teams practicing DevSecOps must use other deployment tools.

Secure your org with DevSecOps

This ebook was written in a collaboration between Gearset and Silverline. Gearset's DevOps solution includes all of the tools explained in this ebook, and we're on hand to advise you.

Gearset

Gearset is the complete Salesforce DevOps solution, with powerful tools for unparalleled deployment success, continuous delivery, automated testing and backups. Thousands of Salesforce professionals have already used Gearset's cloud-based app to run millions of deployments, back up billions of records, and save billions of dollars through productivity improvements.

Founded in 2015 by DevOps experts, Gearset is designed to help every Salesforce team apply DevOps best practices to their development and release process, so they can rapidly and securely deliver higher-quality projects. With inbuilt intelligence that solves the fundamental challenges of Salesforce DevOps, Gearset is a uniquely reliable solution trusted by more than 1000 companies, including McKesson, Accenture and IBM.

SILVERLINE

Silverline creates rewarding experiences for our team, our clients, and the world we live in. We tailor digital transformation solutions to meet your specific needs by leveraging insights acquired through 10+ years in the business and thousands of engagements along with real-world expertise gained across Media and Entertainment, Financial Services and Healthcare industries. From strategic planning and implementation to managed services, we guide clients through every phase of their journey, enabling continuous value with the Salesforce platform. We also offer CalendarAnything, a popular scheduling application on the AppExchange, as well as industry-proven accelerators. For more information visit: **https://silverlinecrm.com/**