Punk Security

An introduction to DevSecOps

This is a high level introduction to the world of DevSecOps, covering the basic concepts of DevOps, DevSecOps and SecDevOps. We will explore the three concepts, then dive into some of the common approaches, and where to use them in the DevOps lifecycle.

Development teams are more productive when security and testing tools integrate into their workflow, identifying mistakes as soon as they are made.

Security teams need to ensure that a common approach is taken to protect the business and that there is proper auditing and safeguards in place to prevent an attack.

DevOps will help your organisation deliver new applications faster, cheaper, and in a more reliable way. Adding DevSecOps will provide that addition of security testing, governance, and compliance.

Punk Security are DevSecOps specialists and have worked with the industry leading vendors. We enable secure digital transformation journeys by finding the right balance between agility and security.

What is DevOps

DevOps is the union of traditional Development and Operations teams, using people, process and technology to rapidly and reliably deploy Business applications.

DevOps requires a shift from traditional isolated technology specific teams to create a more integrated union of developers and engineers.

Basically, DevOps is more than just the adoption of new technology and approaches. It requires process and culture changes too, from engineering all the way to board level.

DevOps brings several key benefits:

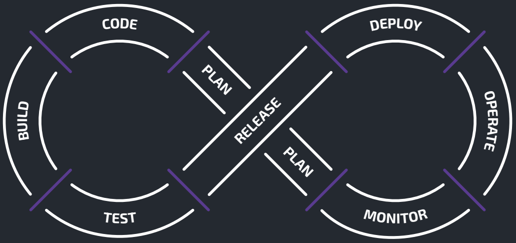
- Speed; quicker development cycles
- Rapid Delivery; automated deployments
- Reliability; enhanced logging and monitoring
- Scale; ability to flex solutions to meet customer requirements
- Improved Collaboration; Operation and development working closer
- Security; DevOps retains control and preserves compliance.
- Reduced costs; quicker development cycles and enhanced security, reduces time to market and wasted project effort.

DevSecOps vs SecDevOps

The main difference between DevSecOps and SecDevOps is that DevSecOps integrates security during the software development life cycle (SDLC) and SecDevOps looks at the security around SDLC.

DevSecOps

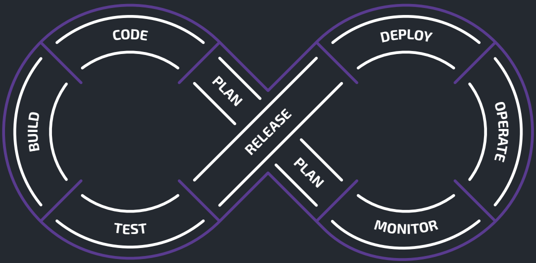
DevSecOps is generally referred to as adding in to the traditional DevOps stages.



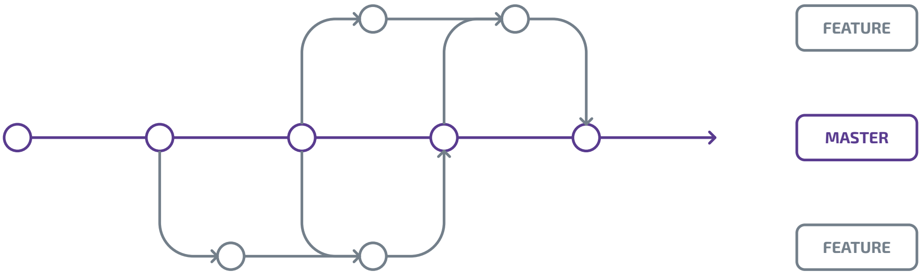
SecDevOps

SecDevOps is creating a strong framework for solutions and system to be built on. We focus on building the culture, promoting secure coding frameworks, implementing governance and policies, training developers and designing secure architecture.

Key:
Security ―



Source Control



Developers can all work together effectively using a Source Code Management (SCM) platform such as GitHub, Bitbucket or GitLab.

Inside this platform, developers can have multiple versions of their application code and get huge benefits like peer approval, version controlling, full audit logging and collaboration. Each version of their code is called a branch and these branches should eventually merge back together.

It is important to agree a branching strategy.

Some teams have one main branch and developers create temporary branches for small changes. These branches then merge back to the main branch when those changes are complete.

Other teams have multiple main branches, with more complicated strategies for merges.

We can also use SCM for our infrastructure definitions, documentation and policies.

What are Pipelines

A pipeline is a way to define build, test, and deploy practices that the development (Dev) and operations (Ops) teams use. One of the primary purposes of a pipeline is to keep the software development process fast, easy and safe.

An easier way to think about DevOps pipelines is to relate it to a manufacturing plant. Before the manufacturer releases its product to the marketplace, it must pass through numerous assembly stages, tests, and quality checks.

For example, a car manufacturing plant will build the chassis, add the motor, wheels, doors, electronics, and a finishing paint job to make it appealing to customers.

Each stage will have to pass certain criteria checks before the car can continue. The same is true with the DevOps pipelines, except we integrate additional software components and run tests to validate quality. Slow, manual test routines reduce the speed of development or are skipped all together.

There are three types of pipelines

01. **Continuous Integration (CI)**

02. **Continuous Delivery (CD)**

03. **Continuous Deployment (CD)**

What is CI

A Developer usually work on a small part of a large application to fix an issue or add new features. This small change could break the wider application and it is very difficult to predict this.

Continuous integration (CI) is a method of automatically testing code to quickly identify errors within a shared code repository.

This enables development teams to quickly write independent code and validate that it doesn't impact on other parts of the application.

A good branching and testing strategy are essential before the benefits of CI can be realised. This will then allow relevant CI test to be run at the correct time, quickening the developer feedback cycle.

CI enables the following:

- Confidence to make bigger changes
- Quicker release cycles, as testing is integrated
- Safer releases, as the application has proven quality
- Reduces cost by finding issues early

There is no point in running every test on all branches. The key is to balance the requirement for quality assurance whilst reducing development bottlenecks and repetitive tasks.

What is CD

How do we deploy the new features and bugfixes into environments quickly, reliably and in a repeatable way?

There are two ways:

- Continuous Delivery (click a button)
- Continuous Deployment (fully automated)

Continuous Delivery enables the rapid deployment of code into production-like or production systems, based upon the results from the CI quality gates.

Typically, this will allow development teams to push their changes into production-like systems for further business acceptance testing and validation.

After the validation is completed, development and/or release teams, can then push to production.

Continuous Deployment uses the same processes but automatically pushes new releases through various environments and then into production.

Continuous Deployment enables rapid development, with some businesses claiming tens of releases per day, but at a greater risk to production stability.

Start with Continuous Delivery, keeping a “human in the loop”, and implement Continuous Deployment when ready.

Security Tooling

There is an abundance of Security tools, both opensource and commercial. Finding the right tool for your company and DevOps teams will always be difficult, but there are nine areas you should focus on:

01. **Secret scanning**

02. **Dependency checker**

03. **Static Application Security Testing (SAST)**

04. **Integrated Application Security Testing (IAST)**

05. **Dynamic Application Security Testing (DAST)**

06. **Infrastructure As Code (IAC) scanning**

07. **Container Scanning**

08. **Web Application Protection (WAF)**

09. **Runtime Application Security Protection (RASP)**

01
Secret Scanning

Developers can all work together effectively using a Source Code Management (SCM) platform such as GitHub, Bitbucket or GitLab.

Developers need passwords and keys to test their code and sometimes these secrets can find their way into the SCM platform alongside the application code.

Once inside the SCM tool, these secrets are visible to other developers and could end up recorded in other systems and log files. Uncontrolled secrets are dangerous.

Secret detection tools immediately spot these secrets, notifying the right people to get the secrets changed and updated in all the required places. We call this secret rotation and some secrets are easier to rotate than others.

Secrets pushed to the central server are difficult to remove and should always be rotated. Secret detection should alert the security team, not just the developers.

02
Dependency Checker

No one builds a complete application without shared code. All modern applications make use of hundreds of third-party or opensource components, which are out of the control of our development teams.

To prevent a modern application from breaking, a given version of these components are used in the application so that they do not suddenly change and break.

When a security issue is found and fixed in one of these dependencies, your application needs to make use of the fixed version and not carry on using the vulnerable component.

To do this, we scan components for reported issues and then automatically test that the fixed versions don't introduce new issues.

03

SAST

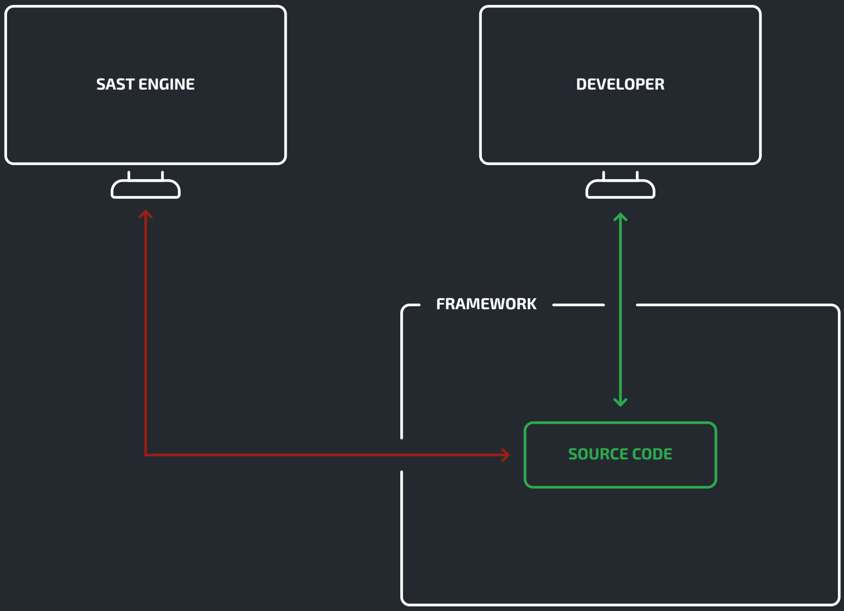
Applications are written in human-readable text and follow very rigid guidelines. This structure allows computers to understand and run the code but it also means security tools can understand the code and identify vulnerabilities.

SAST tools perform complex analysis on the code to identify security bugs without ever running it.

SAST tools are very effective at identifying obvious vulnerabilities and run rapidly. This speed allows developers to fix issues and produce a new code version very quickly.

Some SAST tools even run on the developers computer so they get recommendations as they write the code.

Unfortunately, SAST tools miss the more complex vulnerabilities.



04

IAST

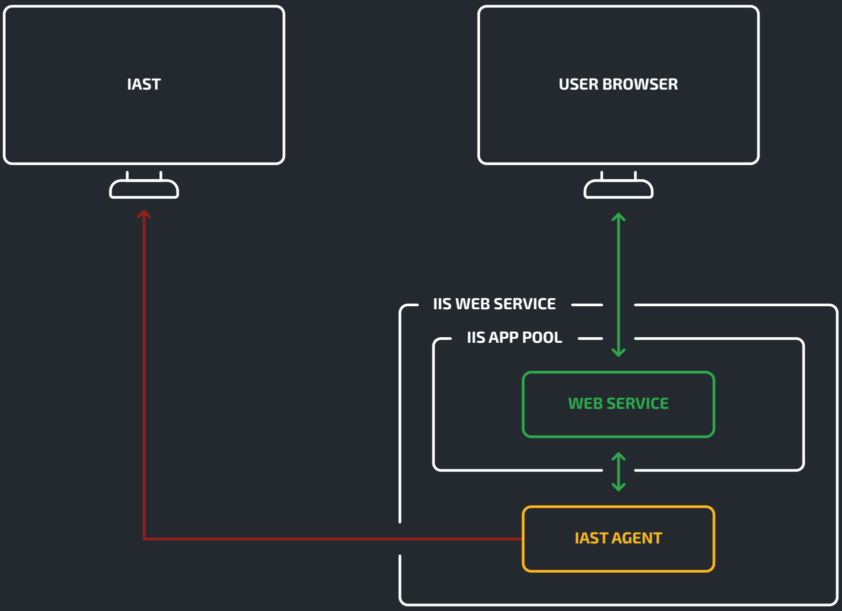
IAST tools run within the application, gaining visibility inside the memory space of the application to check all the components for security bugs. From here they can trace values as they transit through the application until a security issue occurs.

IAST tools allow you to analyse backend application such as API services for security vulnerabilities, whilst the service undergoes function and non-functional testing.

IAST tools do not require any complex security testing or security skills. IAST tools will analyse all application processes (inbound, outbound, internal) and has a very low false positive finding rate.

Some IAST tools can be run in the development IDE so they get instant feedback as they write code.

The limitation is that IAST does not work with databases and user interfaces (UI).



05

DAST

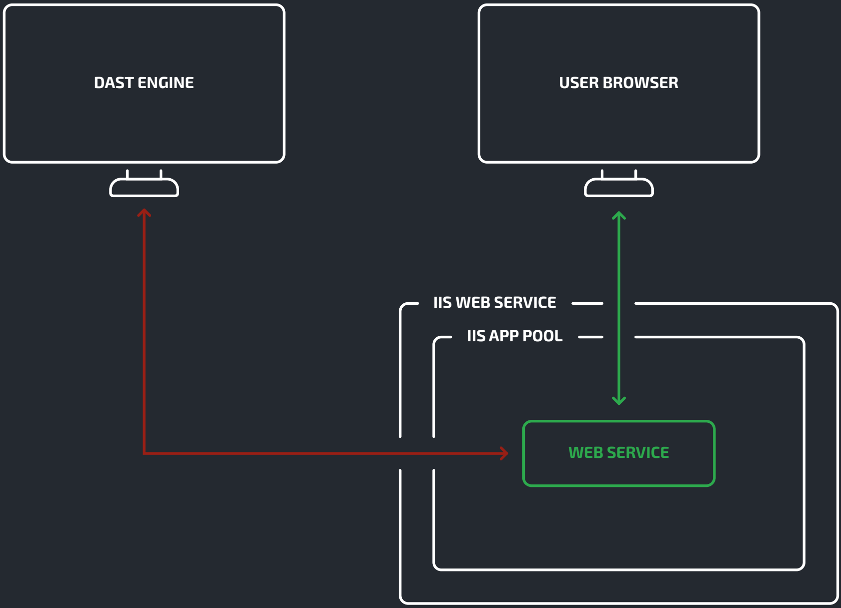
Some security vulnerabilities can be found by observing how the application responds to certain requests, and this tactic is used by security engineers and attackers alike.

Penetration testers use tooling to drive out these issues by sending millions of requests to the application and looking for unusual responses.

DAST tooling automates this process, running automated scans against the running application and monitoring for new issues.

DAST can find complex issues that SAST will not but is also prone to false positives.

Use multiple DAST engines and look out for scanners that target your application framework specifically.



06

IAC Scanning

Infrastructure-As-Code tools, such as Terraform and CloudFormation, give us the ability to rapidly and repeatably deploy infrastructure locally or in the cloud. We can quickly deploy networks, firewalls and servers as quickly as we deploy applications.

However, IaC configurations are prone to miss-configuration as developers might not fully understand the cloud platform or recommended best practices.

IAC scanning tools read the IaC configuration and report issues such as miss-configurations, security issues and vendor poor practice.

Using IAC scanners reduces the risk of cloud solution misconfigurations, and it can be easily integrated into a CI pipeline with huge benefits. In the event of a security issue, the environment will simply never be built.

07

Container Scanning

Applications used to run on production servers, whilst developers ran their own development servers and kept a production like environment for testing. Keeping these environments consistent was difficult and applications would behave differently in each one.

Developers can now use containers to package up an environment so it's consistent everywhere. A moment-in-time snapshot of a working environment that's been tried and tested.

We can scan these containers for security vulnerabilities without needing to scan the production servers and polluting logs. If we find an issue then we simply build, test and deploy a new container.

08

WAF

Web Application Firewalls (WAFs) are the first line of defence and protect web applications from attacks. They analyse all web traffic before it reaches the actual applications and prevents attacks by simply dropping any attackers requests.

To stop DDoS attacks from flooding the applications network, an internet based WAF should be used. The WAF will handle the DDoS traffic whilst forwarding on real customer traffic, preventing disruption, and hiding the real location of your application servers.

A WAF will prevent common attacks, but has additional benefits:

- Soak up DDoS attacks
- Produce a log of all web activity
- Block geographic regions
- Enforce captcha forms
- Deny access to parts of the application

09

RASP

As attackers become ever more sophisticated, so must our methods to detect security violations and protect applications.

Runtime Application Self-Protection (RASP) allows us to block malicious activity in the production application as it happens by running inside the application for maximum visibility.

RASP solutions can analyse the build-up of an attack and block known attacks such as SQL injection, XSS, library CVE exploits, account take overs and some zero day vulnerabilities.

The RASP detections can be used to trigger automated responses such as blocking the users access or triggering a security alert.

The use of RASP technologies in on-prem, cloud, containerised, or serverless workloads better enables companies to protect business applications.

Monitoring & Logging

Applications, servers, third-party services and cloud providers all produce logs that are critical to understanding application performance and finding security problems. It is common to find these logs being handled differently or simply discarded.

Businesses need to agree a standard approach to handling these different log sources and then enforce that approach by bringing these logs into a single platform. This ensures that developers can be responsible for the quality of logs but do not need to maintain a logging platform.

By bringing all logs into one place, security teams can apply a common set of alerts and respond to cyber incidents much faster.

The business needs to consider:

- What data should be redacted
- How long logs should be kept for
- Who needs access to these logs
- What notification channels are available

Secret Management

Applications make use of passwords and keys to connect to other services such as databases and third-party APIs. Collectively these are known as secrets and they need to be protected and easily changed.

Secrets need to be protected to ensure they cannot be stolen, accidentally leaked or changed. Developers and administrators will make use of secrets in their daily activities, so secrets should be changed often.

To manage secrets effectively you should ensure that:

- The secret is only available to those who need it
- All access to the secret is logged
- The secret can be changed easily
- The secret is encrypted when transmitted
- The secret is changed often
- The secret is encrypted when stored

Secrets management is not the time for creative thinking, stick to tried and tested approaches.

Finding Management

Implementing DevSecOps tooling allows developers to see security issues inside their applications that they never knew existed. This is hugely valuable but introduces a new flow of work and the initial backlog of work can be daunting.

Tools should be introduced in a phased approach to reduce the peak size of the backlog, allowing developers time to tune out false positives and triage findings.

Each tool will behave differently and developers should not be expected to interact with each one manually.

Critical issues should trigger alerts to the developer, typically by blocking their CI pipeline and preventing the release of work.

Detailed findings should be sent to their existing backlogs. Each tool will have its own ways of reporting issues but all can be pushed into the existing workflow management solution, such as Jira or Azure DevOps.

Allow developers to manage findings like every other feature and bugfix by pushing them into their existing backlogs.

Punk Security specialise in DevSecOps consulting and Cloud Security services

InfoSec

We provide traditional InfoSec services, like penetration testing of web applications and infrastructure.

Open source advocates

We contribute to existing opensource products and maintain two of our own. Check out pwnSpoof and SMBeagle

Any Cloud

We work with all major cloud platforms, allowing us to audit environments and build secure automation no matter which flavour you use.

Any Tool

We don't just resell one vendor, we use the right tools to suit each client and we always ensure we build the simplest and most robust configuration.

Reach out to us at:

Phone
+44 161 660 3545

Website
punksecurity.co.uk

Email
info@punksecurity.co.uk

LinkedIn
punk-security-limited