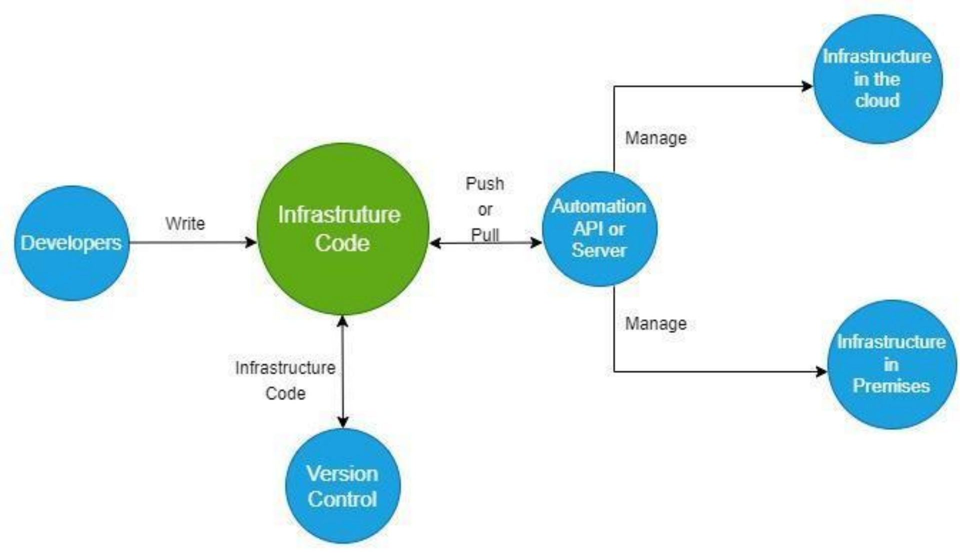
Infrastructure as Code

What is Infrastructure as code

The process of managing and provisioning computer data center resources through machine-readable definition files



Benefits

- Reduce human error in configuration
- Accountability
- Speed
- Scalability
- Reproducibility
- Modularity

Approaches to IAC code

- Declarative
  Define state and let the tool decide how best to install and configure the system to match it
- Imperative
  Specify the process to get to the desired state yourself

Security Issues

- Hard Coding Secrets

  can be mitigated with secrets management tools like vaults

- Misconfiguration
  Static analysis tools like checkov and terrascan can scan for common issues

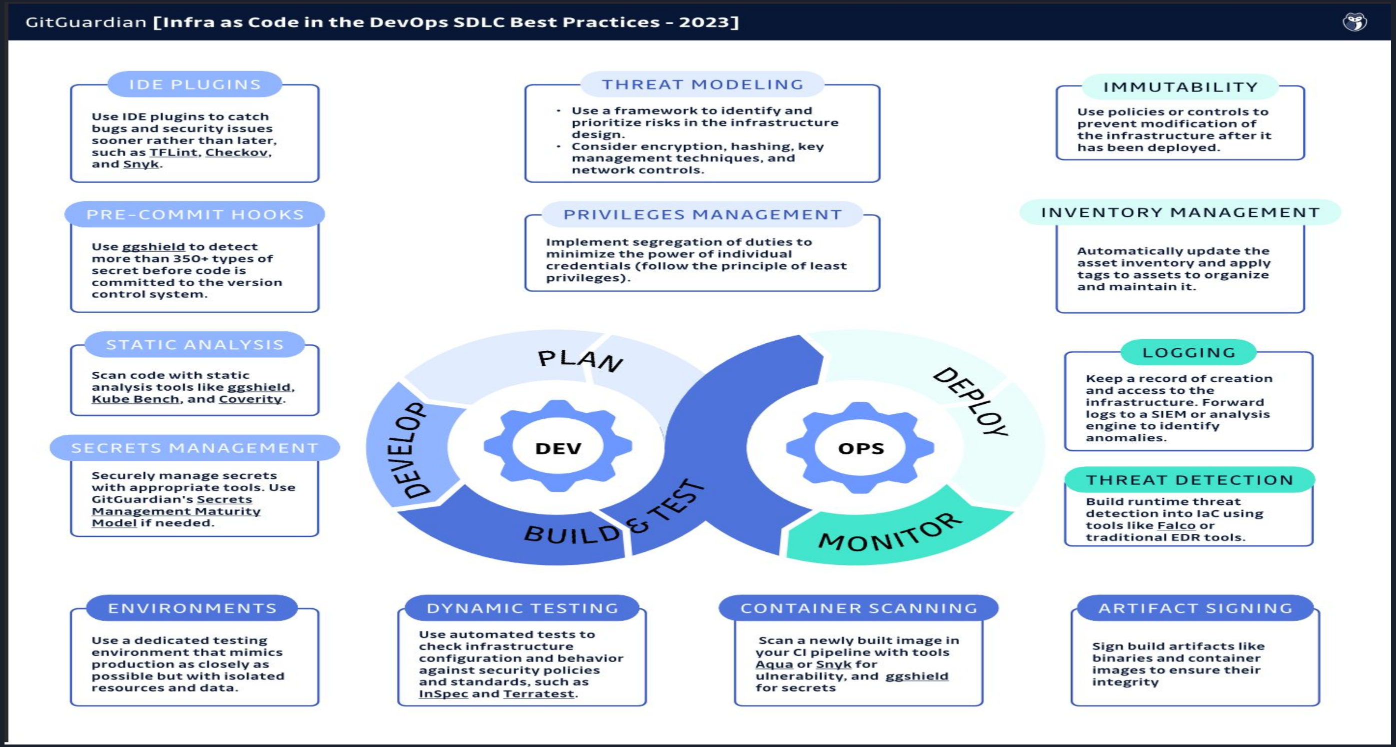
- Privilege issues
  ensure users and automated processes only have the minimally required access rights

Tools

- Ansible
- Terraform
- Chef
- Puppet
- Salt

Code Example

Best Practices



GitGuardian **[Infra as Code in the DevOps SDLC Best Practices - 2023]**

IDE PLUGINS

Use IDE plugins to catch bugs and security issues sooner rather than later, such as TFLint, Checkov, and Snyk.

PRE-COMMIT HOOKS

Use ggshield to detect more than 350+ types of secret before code is committed to the version control system.

STATIC ANALYSIS

Scan code with static analysis tools like ggshield, Kube Bench, and Coverity.

SECRETS MANAGEMENT

Securely manage secrets with appropriate tools. Use GitGuardian's Secrets Management Maturity Model if needed.

THREAT MODELING

- Use a framework to identify and prioritize risks in the infrastructure design.
- Consider encryption, hashing, key management techniques, and network controls.

PRIVILEGES MANAGEMENT

Implement segregation of duties to minimize the power of individual credentials (follow the principle of least privileges).

IMMUTABILITY

Use policies or controls to prevent modification of the infrastructure after it has been deployed.

INVENTORY MANAGEMENT

Automatically update the asset inventory and apply tags to assets to organize and maintain it.

LOGGING

Keep a record of creation and access to the infrastructure. Forward logs to a SIEM or analysis engine to identify anomalies.

THREAT DETECTION

Build runtime threat detection into IaC using tools like Falco or traditional EDR tools.

ENVIRONMENTS

Use a dedicated testing environment that mimics production as closely as possible but with isolated resources and data.

DYNAMIC TESTING

Use automated tests to check infrastructure configuration and behavior against security policies and standards, such as InSpec and Terratest.

CONTAINER SCANNING

Scan a newly built image in your CI pipeline with tools Aqua or Snyk for ulnerability, and ggshield for secrets

ARTIFACT SIGNING

Sign build artifacts like binaries and container images to ensure their integrity

Reference links

https://en.wikipedia.org/wiki/Infrastructure_as_code

https://www.crowdstrike.com/cybersecurity-101/infrastructure-as-code-iac/

https://www.ansible.com

https://www.terraform.io/

https://blog.gitguardian.com/infrastructure-as-code-security-best-practices-cheat-sheet-included/

https://cybersecuritynews.com/infrastructure-as-code-security/

https://www.digitalocean.com/community/tutorials/how-to-use-ansible-to-automate-initial-server-setup-on-ubuntu-20-04