Gamifying supply chain security

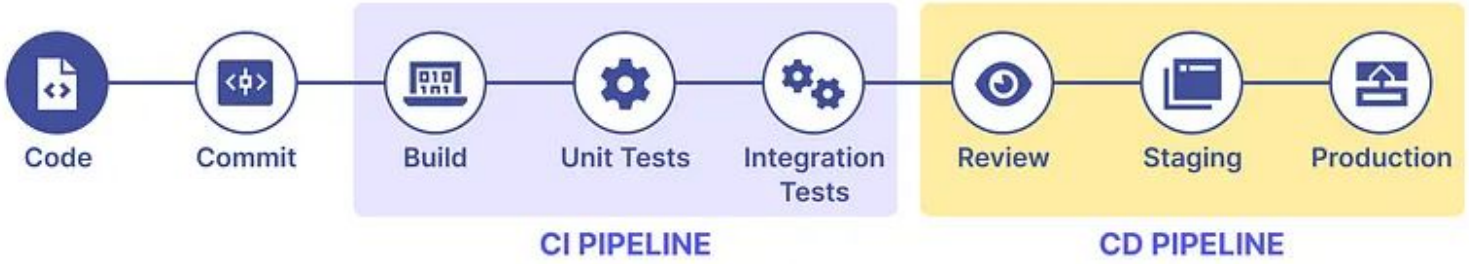
Jacopo Mauro
University of Southern Denmark

Personal background

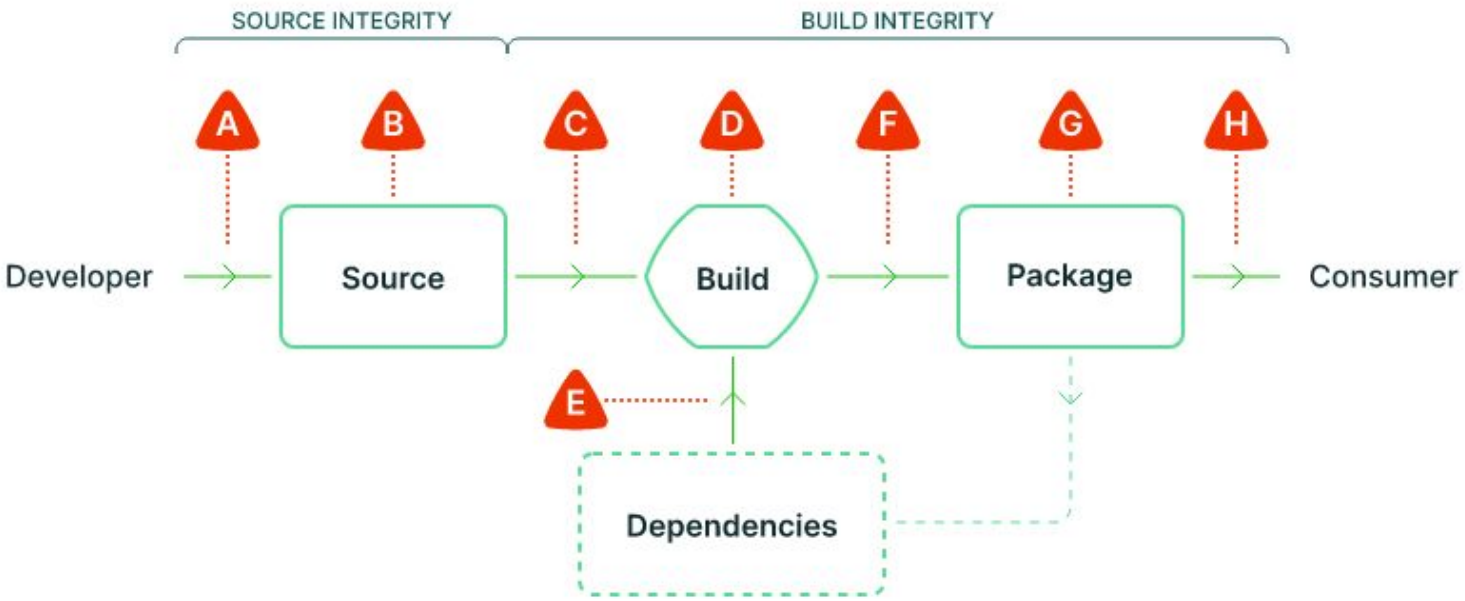
- Professor at SDU
  - Course on Microservice and DevSecOps
  - Thesis in cybersecurity
- Co-organizers of the Danish cybersecurity challenge (De Danske Cybermesterskaber)
  - Locals, regionals, nationals
  - In nationals ~100 people (15 to 25) to play Jeopardy style CTF
  - Interested in try to define new variety of CTFs (e.g., DevOps, FM, Phishing)

Motivation: DevOps

CI/CD Pipeline

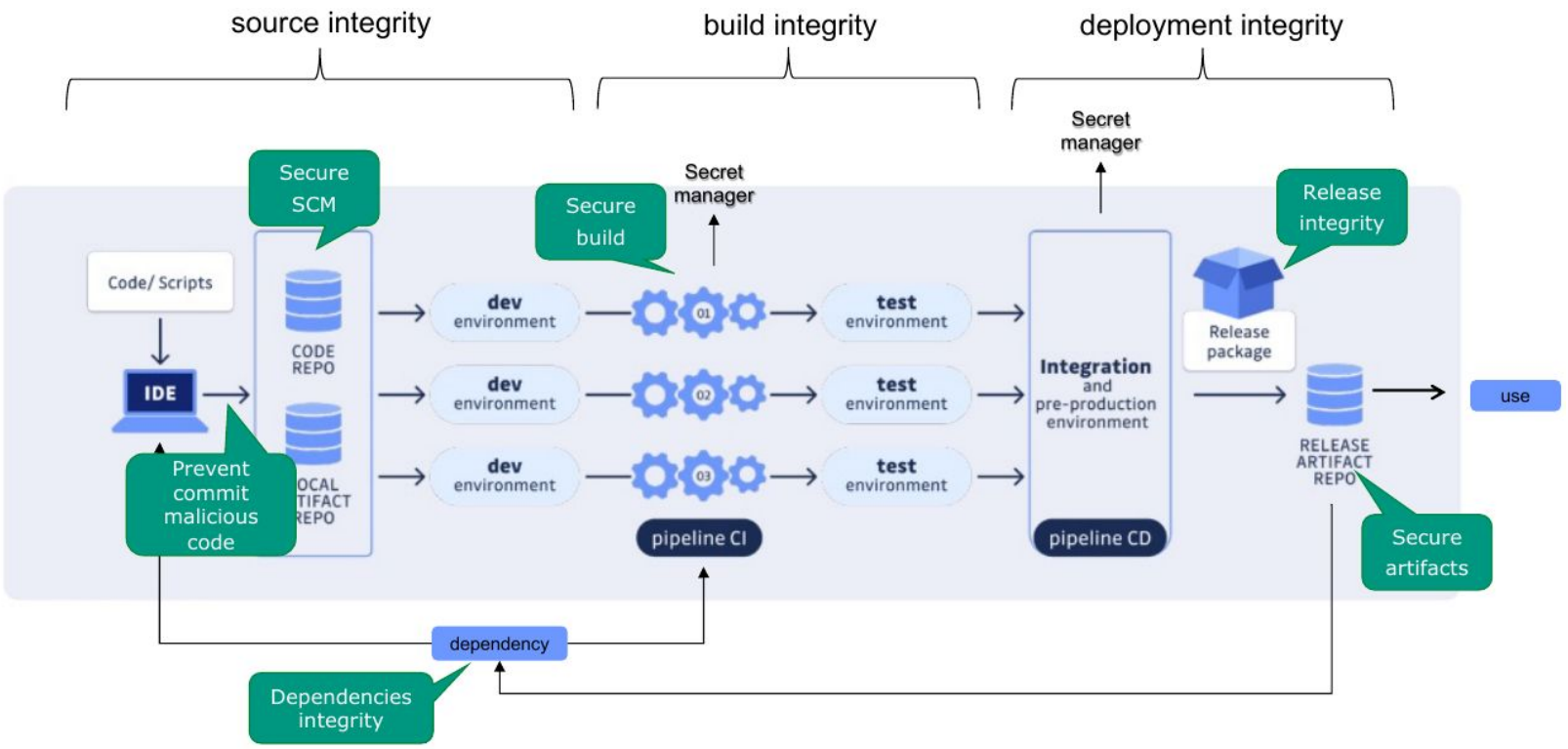


Threats



A Submit unauthorized change
B Compromise source repo
C Build from modified source
D Compromise build process
E Use compromised dependency
F Upload modified package
G Compromise package repo
H Use compromised package

Securing Pipeline



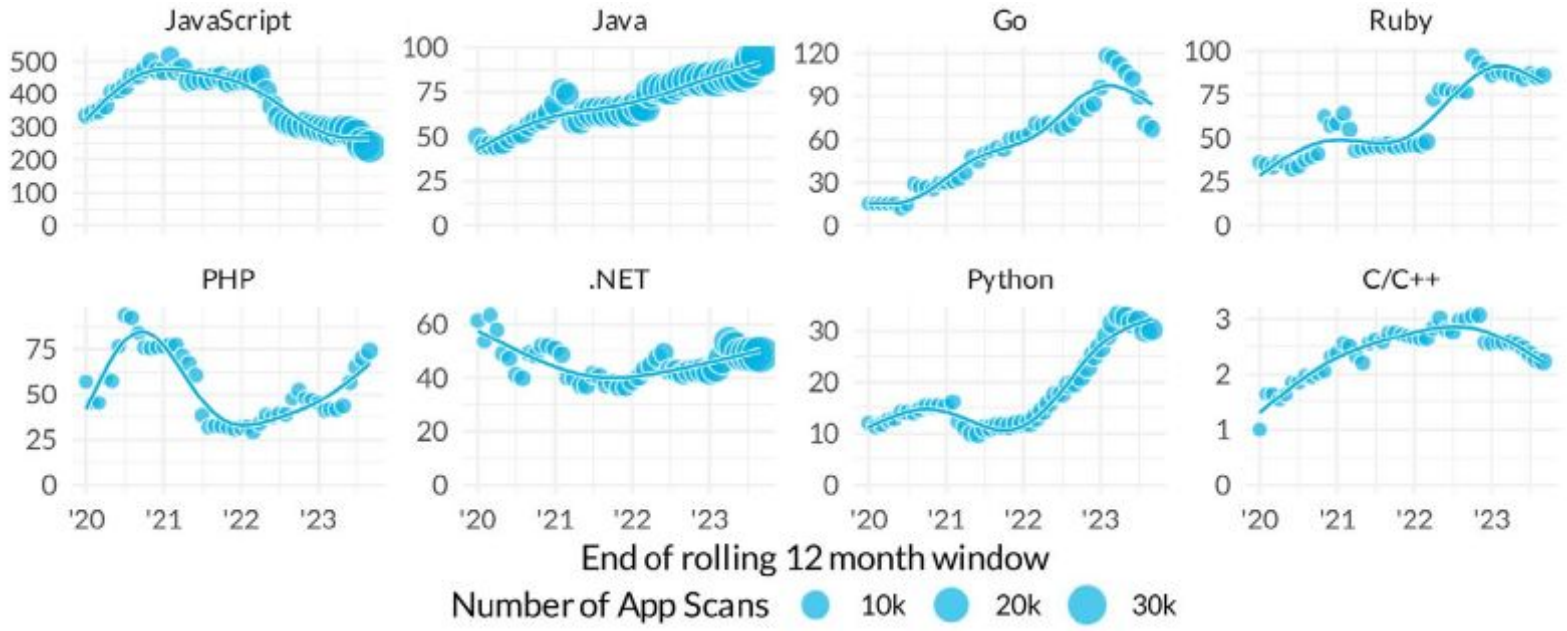
Examples of Attacks

- IDE - Octopus Scanner Malware (2020)
  - Attacked NetBeans IDE
- Dependencies - Log4Shell attack (2021)
  - Vulnerability in Apache Log4j logging framework
- Repos - Canonical Github account breach (2019)
  - Github account compromised
- Pipeline CI - Codecov (2022)
  - Script to share env variables from the CI of Codecov customers
- Release Package - SolarWinds (2020)
  - Access to SolarWinds development infrastructures and malware injection

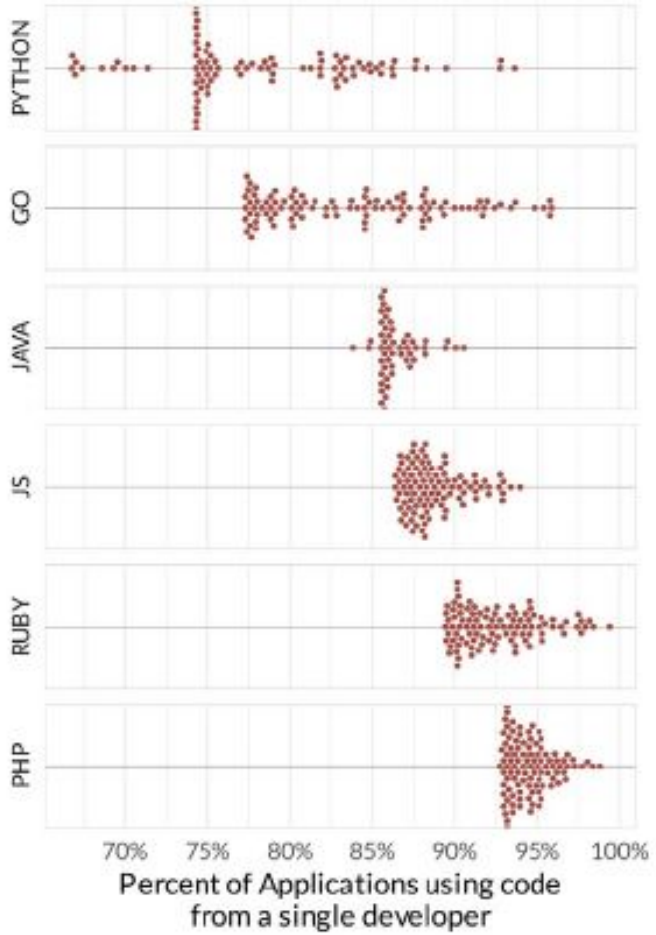
More available at https://www.sonatype.com/resources/vulnerability-timeline

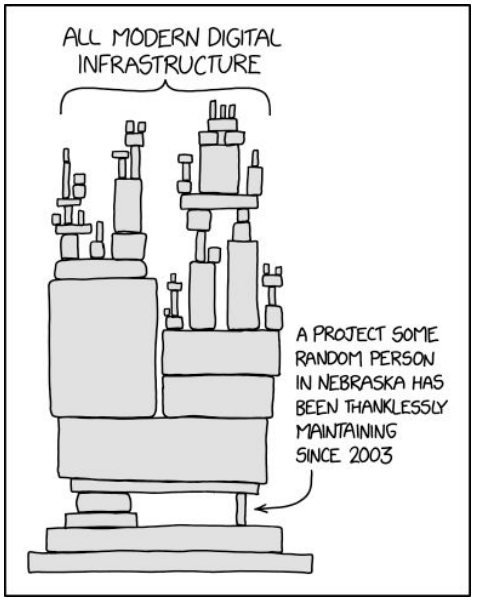
Libraries per application

“In many respects, development teams have shifted from writing software to assembling software.” Chris Wysopal, CTO Veracode



One Developer Code is everywhere





CTF

How do we gamify Supply Chain Attacks?

Require having the full CI/CD pipeline

- Many services
- Container images
- Many moving parts and tools

Problems

- Complexity
- Many resources needed (CPU, RAM, Disk)
- Security (i.e., how to allow docker in docker?)

1st Experiment

Platform for reproducible, isolated, and quickly deployable CTF environments

Goal

- Support for containers and VMs
- Strong isolation
- Facilitate the deployment and testing of challenges
  - Every user has provided unique URI to access a challenge
  - Possibility to access the challenge via HTTPS and SSH (on same port)

Tech Stack

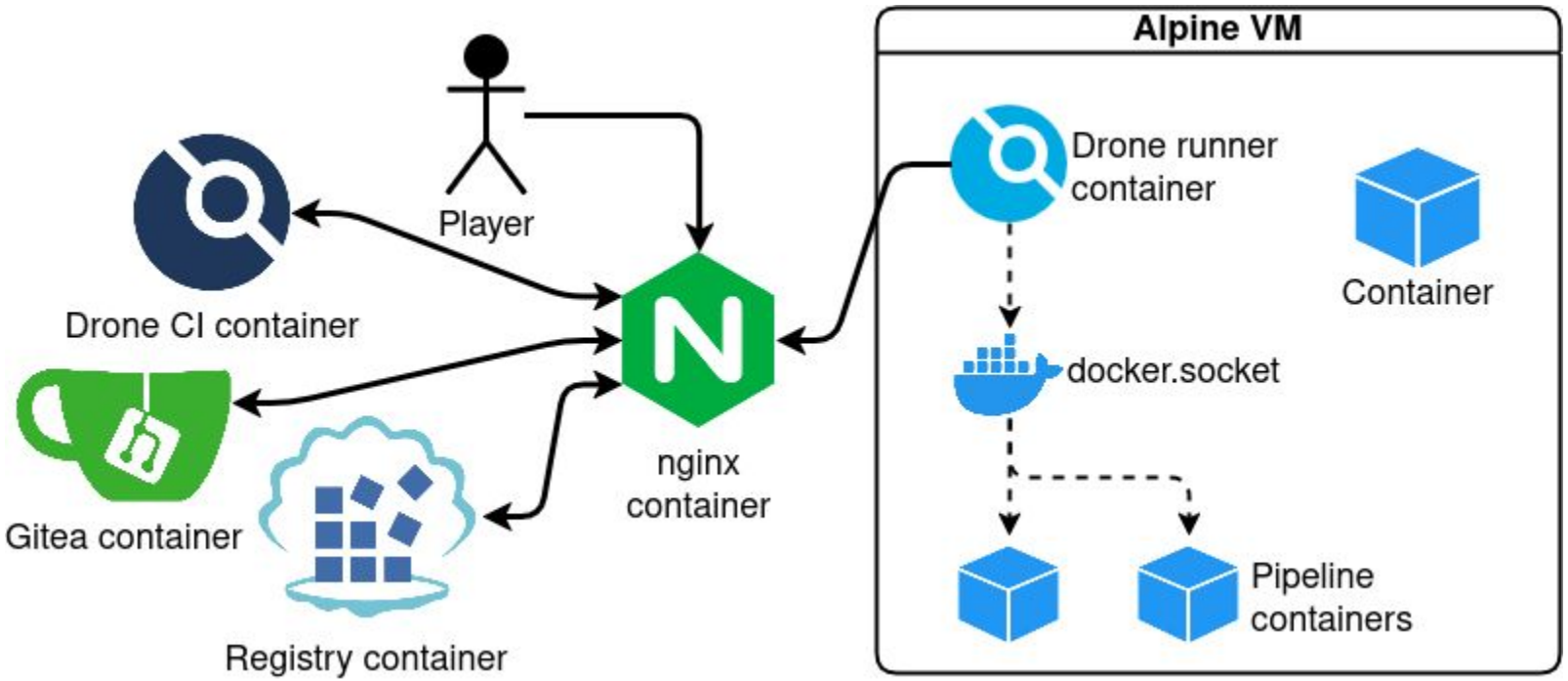
- Deployer Backend – Creates challenge environments (VMs via KubeVirt or containers + Pulumi)
- CTFd – Competition frontend integrating with backend via custom plugins
- Keycloak – Identity & Access Management (SSO, RBAC)
- Smallstep CA – Issues certificates for all internal services and SSH
- Monitoring Stack – Prometheus, Loki, Grafana
- Feature Flags – Unleash

1st Experiment - Outcome

- Tried as platform to submit CTF as bachelor projects
- Allowed the possibility to implement challenge and script to solve the challenge
  - Developers can self deploy and check challenge
  - Automatic checking of the challenge solution
    - If they are not too resource hungry
    - If verifier did not need a URL to be used by challenge
- VM are really heavy
  - 4GB for each of them is a lot
  - Slow startups (minutes)
- Caching for images across VM not trivial

2nd Attempt

Idea: deploy a CI/CD pipeline in a VM run inside a container



2nd Attempt

- Outer Container does not require privilege access
- Qemu allows to run VM and container escape challenges can be used
- Runners run in VM (they require docker in docker)
- Other things can also be deployed outside the VM for performance (if secure)
- Can run in systems in which connectivity is disabled after starting the challenge (e.g.., Haaukins)

Cons

- Still require a lot of resources
- Complex to have everything running together
  - E.g., bypass authentication of gittea

2nd Attempt - Results

First version used in 2024 edition of Danish National Competition

- Developed by Oliver Nordestgaard
- 1 challenge - 5 flags
  - Discover exposed Git repositories and extract leaked credentials and secrets hidden in commit history
  - Modify a CI pipeline to exfiltrate secrets and registry credentials that were assumed safely stored
  - Abuse leaked API credentials to escalate to admin and execute commands through the deployment pipeline to access the flag
  - Leverage admin control to enumerate organization-wide pipeline secrets in Drone and leak a user’s deleted-repo secret
  - Compromise the Git/Drone admin account to enable trusted pipelines and break out of Docker to retrieve the final server flag → docker escape
- Some got the first flags, none solve them all

2nd Attempt - Results

Refined version in 2025 edition of Danish National Competition

- Developed by William Bundgaard
- Added docker register
- Empowering DevOps Security - 1 flag
  - Focus on supply-chain tampering inside a CI/CD pipeline
  - Players manipulate a base image to exfiltrate secrets from a dependent pipeline
  - 7 players solved it
  - Too big for regionals, better for nationals (100 participants)
  - https://github.com/williamBundgd/Empowering_DevOps_Security_2
- Glass Ocean - 2 flags
  - Focus on Docker escape and misconfigured CI/CD runners
  - Players exploit host-mounted docker.sock to gain control over containers and pivot to hidden resources
  - 2 players solved it
  - Feedback: giving access to docker.sock is too much
  - https://github.com/williamBundgd/GlassOcean_CTF

Future Work

- Improve DevOps challenges
  - Other CI servers
  - Multi-stage supply chain flows (e.g., artifact signing, SBOM validation, provenance checks)
  - Other weak artifacts (logs, config leaks, pipeline dashboards)
  - Develop a scenario generator that produces template repositories, pipelines, secrets, and misconfigurations automatically
  - K8S in Docker (& GitOps) challenges?
- No idea how
  - Insecure IaC pipelines