Security in CI/CD pipelines with F5

Shain Singh, Principal Security Architect @ F5, ss@f5.com

Leon Seng, Modern Apps Practice Lead @ F5, leon@f5.com

Our Speakers

Shain Singh

Principal Security Architect

https://github.com/shsingh

https://www.linkedin.com/in/shsingh/

Leon Seng

ANZ Modern Apps Lead

https://github.com/leonseng

https://www.linkedin.com/in/leonseng/

DevSecOps has not been successful in most enterprise organisations

This sentiment is seen throughout industry

https://sched.co/1BSRd

70% Of developers admit to skipping security due to delivery timeframes

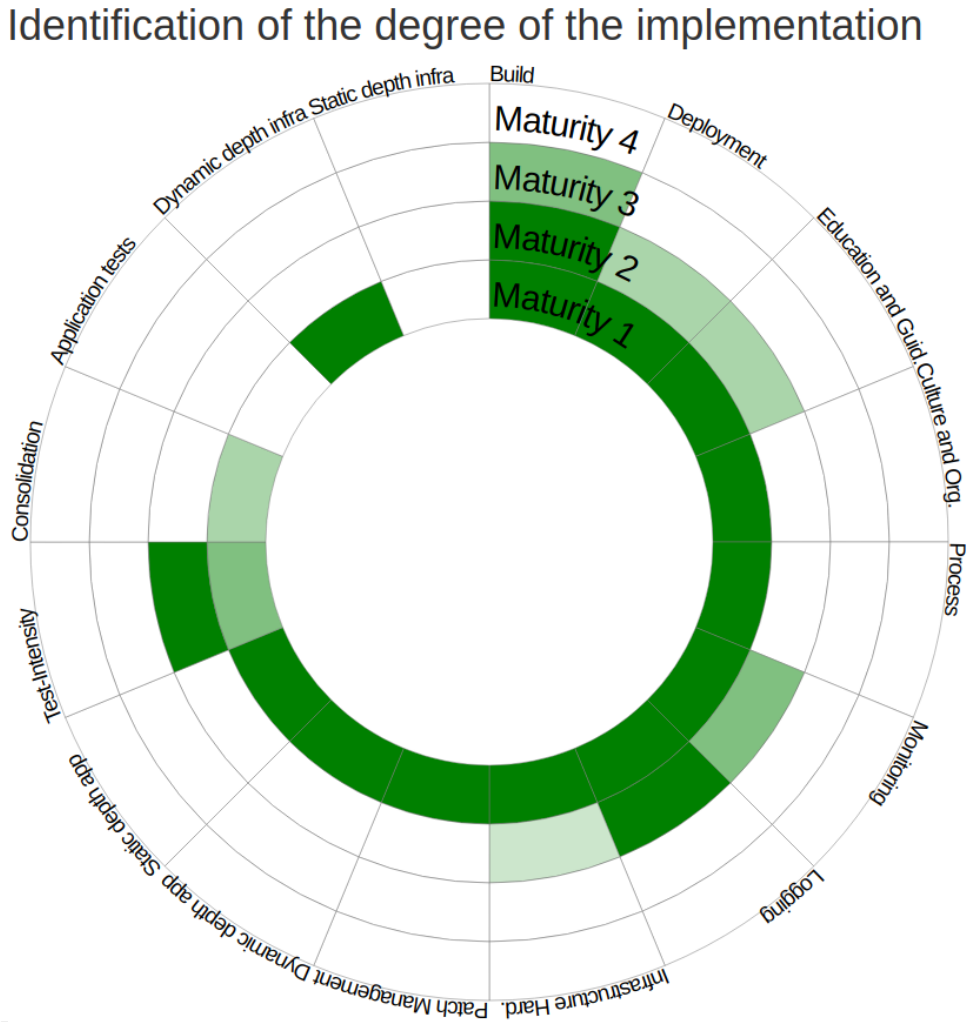
81% Of devs admit to pushing code with known vulnerabilities

96% Of cloud breaches are self-inflicted

Source: Osterman Research

Why have we not made much progress?

EMPHASIS HAS BEEN ON EFFICACY FOR INDIVIDUAL APPLICATIONS OVER FULL COVERAGE OF ALL DEPLOYMENTS



https://dsomm.owasp.org/

DevSecOps Maturity Model (DSOMM) Level 1
Basic understanding of security practices

Recommendations:
- Never fail a build pipeline – security scans will have false positives
- Investigate static and dynamic tools for the DevOps pipeline
- Build expertise with tools and analyse results
- Collaborate with development teams to resolve issues

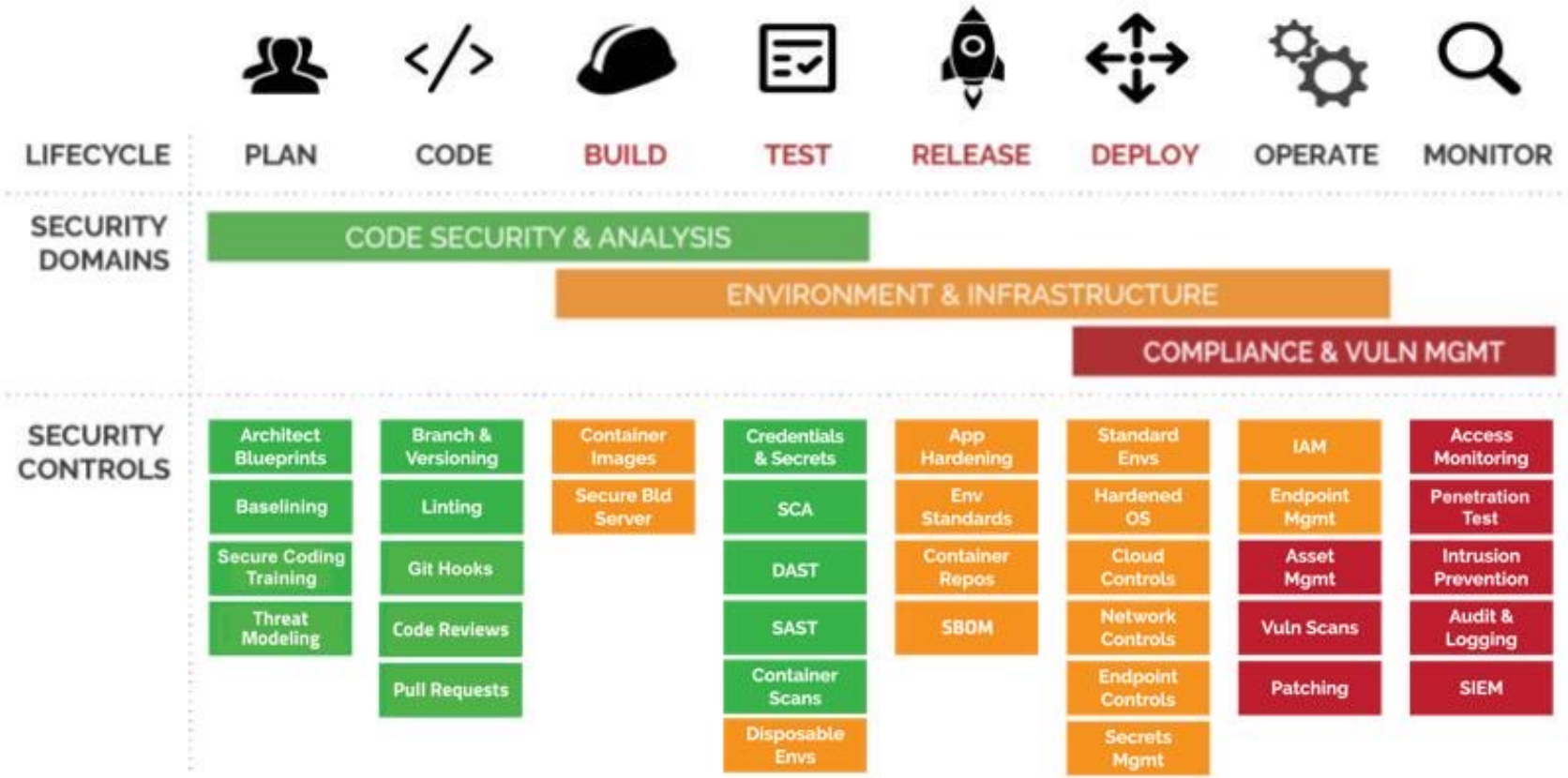
DevSecOps Maturity Model (DSOMM) Level 2
Adoption of basic security practices

Recommendations:
- Investigate tweaking tools from their default settings for tuning
- Storing results from tools in a consolidated environment
- Starting a security champion program

DevSecOps Maturity Model (DSOMM) Level 3
High adoption of security practices

DevSecOps Maturity Model (DSOMM) Level 4
Advanced deployment of security practices at scale

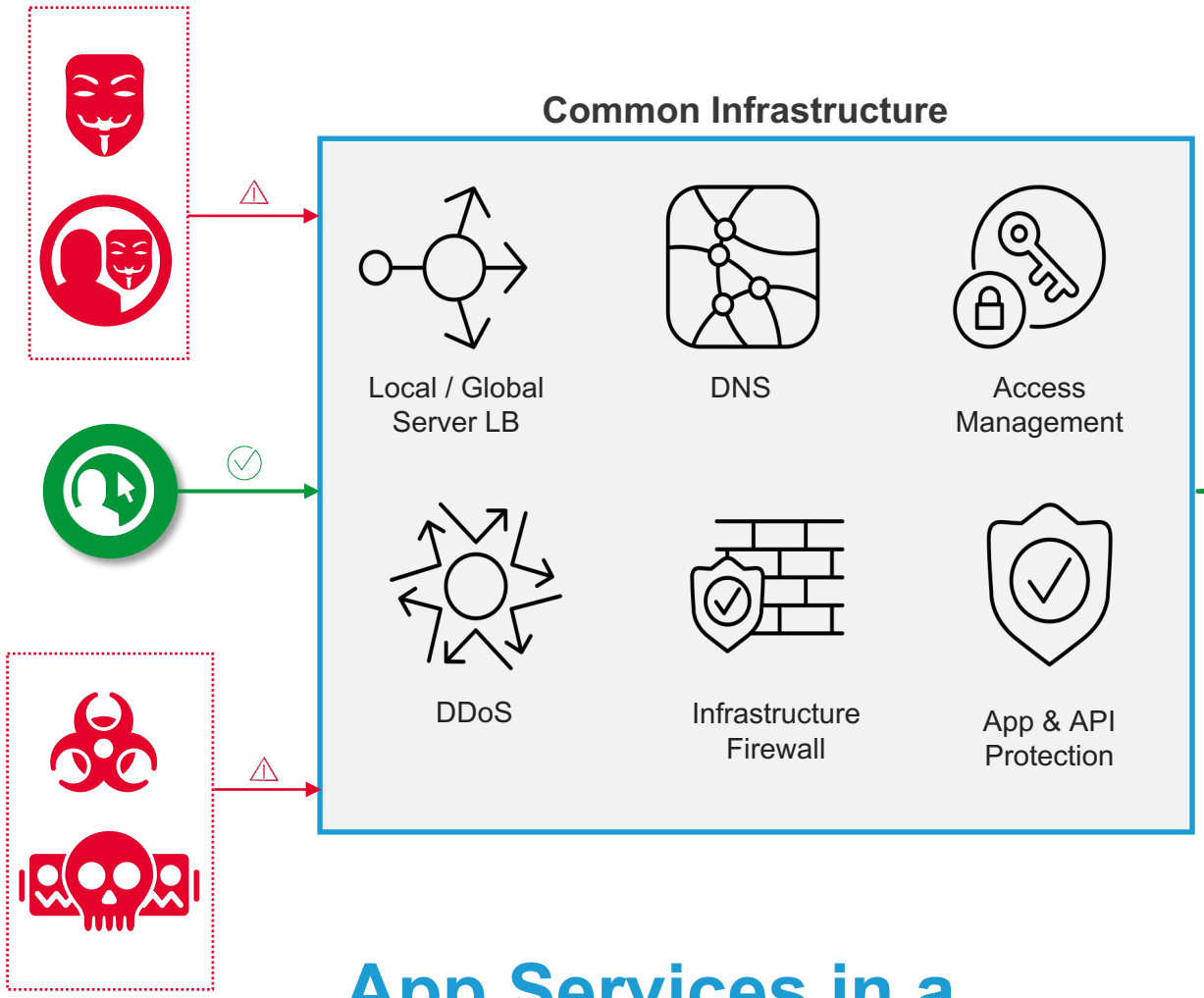
Separating tasks into domains of ownership



https://static.sched.com/hosted_files/owasp2022globalappsecsf/11/DevSecOps Playbook- A step-by-step guide to implementing a DevSecOps Program - Paul McCarty - Friday.pdf

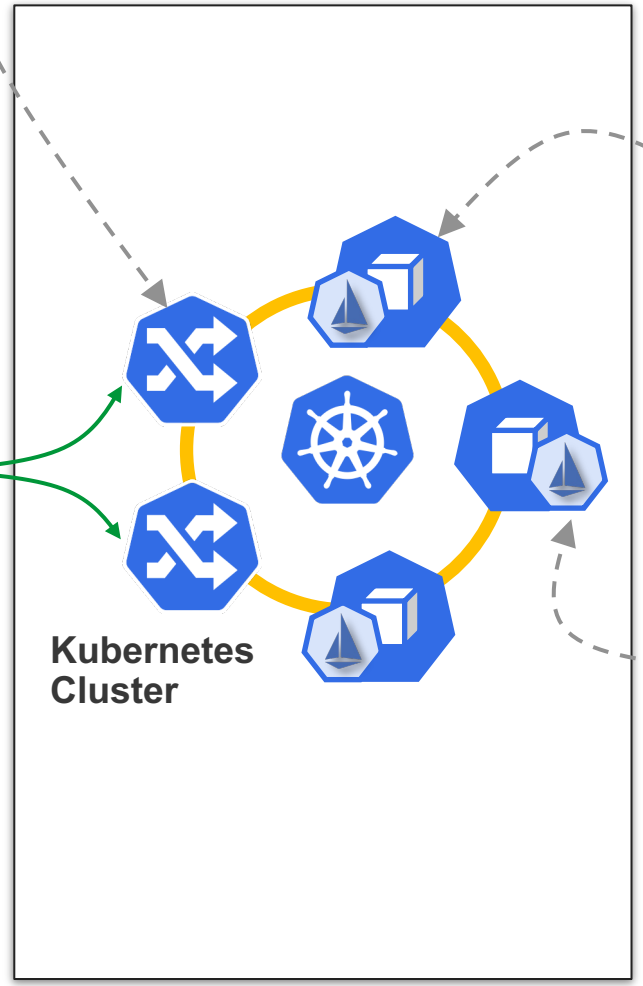
I thought F5 made load balancers – what do they have to do with DevSecOps?

App Services in a Modern Architecture



Ingress (with API Gateway)

- Layer 7 routing for traffic entry point coming into Kubernetes
- Built for HTTP traffic. TCP/UDP for non-HTTP traffic
- May include API Gateway implementation



Pods

- Runs app in a container / CNF

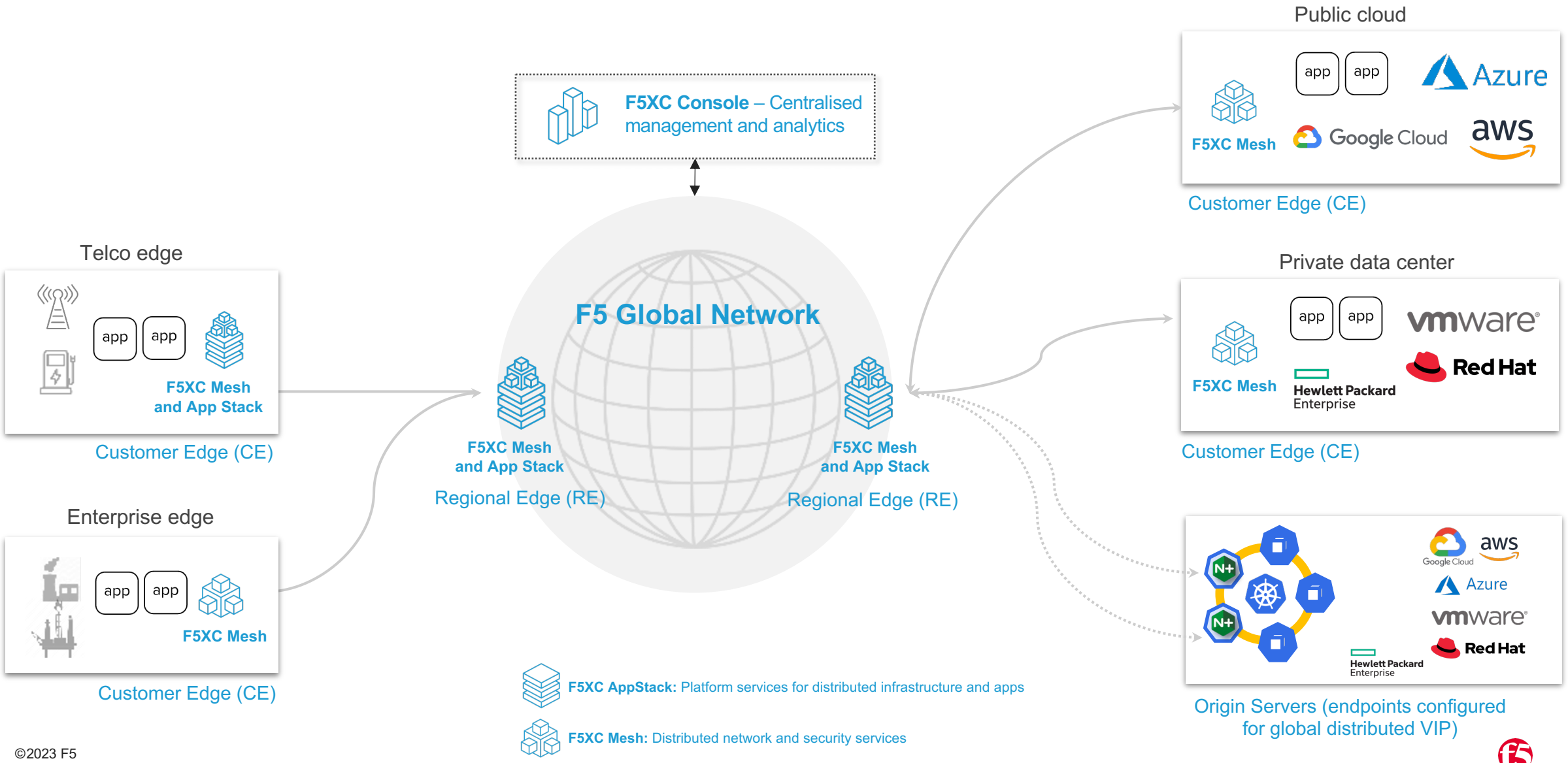
Service Mesh

- Open Source Service Mesh implementation (Istio)
- Injects Sidecar to every pod
- Enforces routing, security with mTLS, etc.
- Provides traceability of pod communication

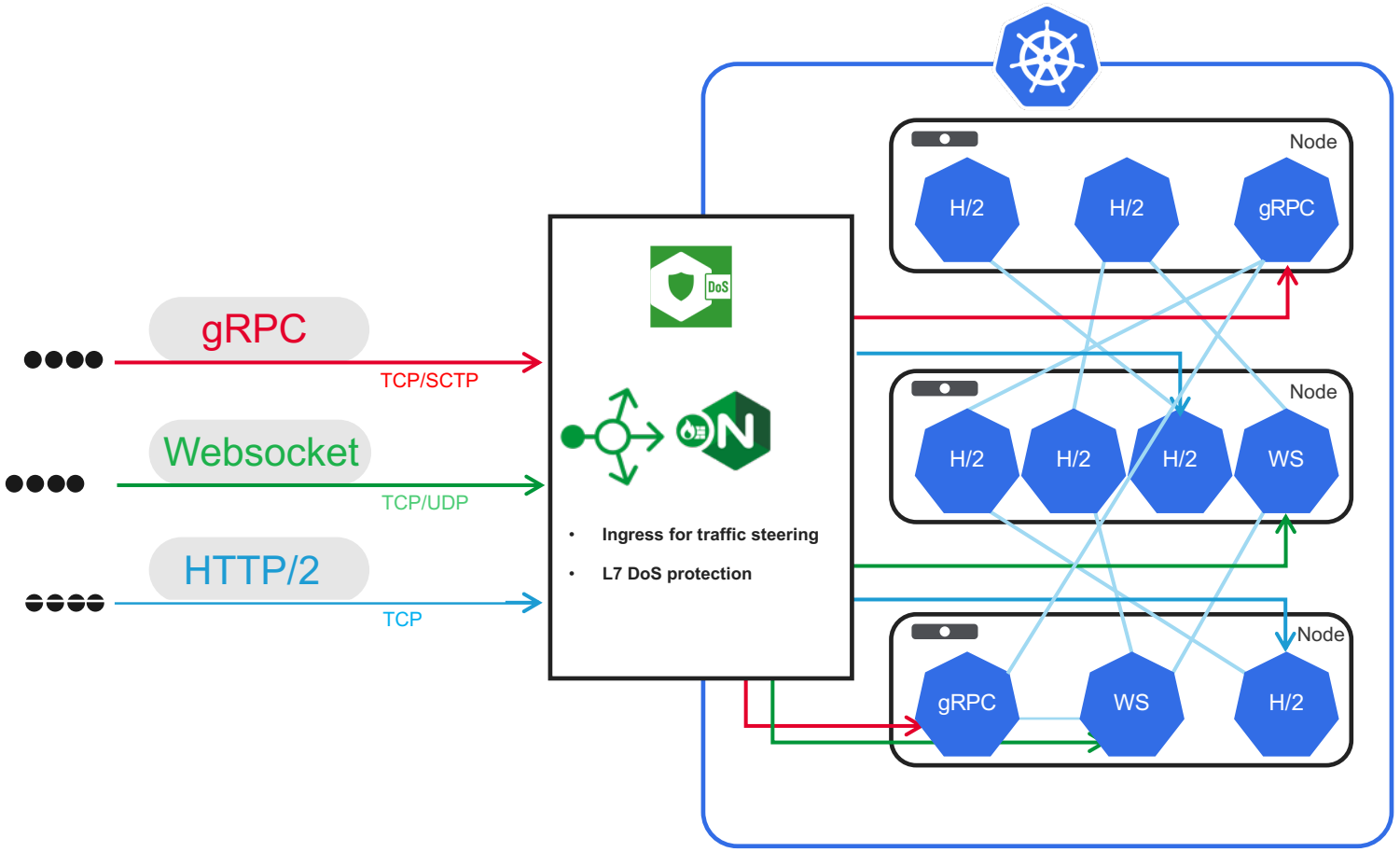
Cloud Microservices PaaS

- On-prem private cloud (e.g: VMware)
- Public cloud (e.g: AWS, Azure, GCP)

App Services in macro-services (global) environment



App Services in micro-service environment



Capabilities:

- DoS protection for:
  - HTTP
  - gRPC
  - Websocket
- Web application and API security
- OpenAPI Spec (Swagger) enforcement
- Attack Signature/Schema Validation inside:
  - HTTP
  - XML
  - JSON
  - gRPC
  - Websockets
  - GraphQL
- TCP SYN flood protection via eBPF
- Comprehensive Ingress and AuthN/AuthZ capabilities

Deployment:

- NGINX Ingress Controller

Should we put a WAF in front of all applications?

Security controls that are now mainstream

“Should I create ACLs for non-internet facing apps”

- Docker/Kubernetes service definitions
- Public cloud network ACLs (e.g. AWS security groups)

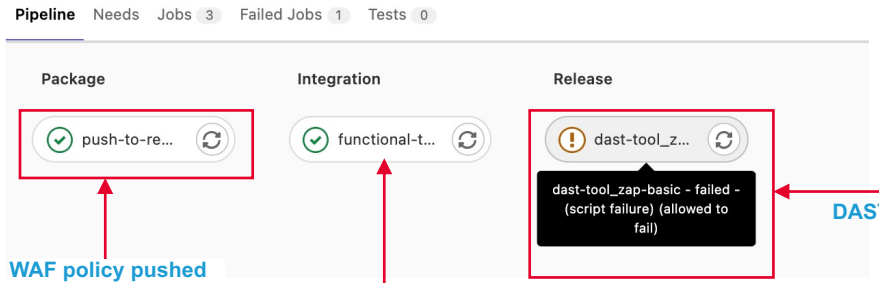
“Should I remove cleartext data at rest and in transit for internal apps”

- Secrets management (e.g Hashicorp Vault)
- mTLS via service mesh
- LetsEncrypt Certbot for TLS certificates

Can we not implement web application protection if we make deployment simple?

- NGINX App Protect WAF and DoS configuration via Kubernetes manifests, deployed via Continuous Delivery tooling
- Functional testing of application post WAF deployment removes potential for false positives

Example – Post deployment WAF effectiveness



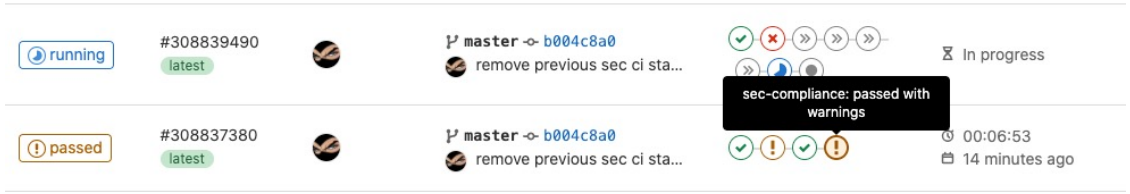
awaf_policy-hapi-fhir.json 766 Bytes — Edit | Web IDE | Lock | Replace | Delete

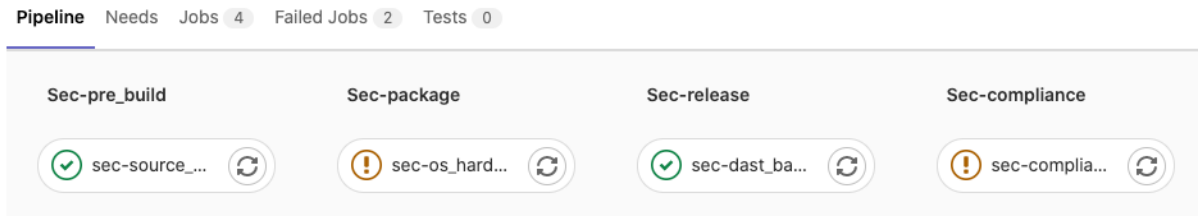
```
{
    "policy": {
        "name": "policy-hapi-fhir",
        "description": "HAPI Policy for FHIR",
        "template": {
            "name": "POLICY_TEMPLATE_API_SECURITY"
        },
        "enforcementMode": "blocking",
        "server-technologies": [
            {
                "serverTechnologyName": "MySQL"
            },
            {
                "serverTechnologyName": "Unix/Linux"
            }
        ],
        "signature-settings": {
            "signatureStaging": false
        },
        "policy-builder": {
            "learnOnlyFromNonBotTraffic": false
        },
        "open-api-files": [
            {
            "link": "https://gitlab.com/shainsingh/hapi-fhir/-/raw/master/hapi-fhir_swagger.json"
            }
        ]
    }
}
```

sec-release.gitlab-ci.yml 1 KB — Edit | Web IDE | Lock | Replace | Delete

```
services:
  - docker:dind

sec-dast_baseline:
  stage: sec-release
  image:
    name: owasp/zap2docker-weekly
  before_script:
    - mkdir /zap/wrk
  script:
    - zap-baseline.py -t https://hapi.f5labs.dev/fhir -I -J dast_baseline_scan-results.json
  after_script:
    - mv /zap/wrk/dast_baseline_scan-results.json /builds/shainsingh/hapi-fhir/
  rules:
    - if: $scan_periodic != "nightly"
      when: always
  artifacts:
    paths: [dast_baseline_scan-results.json]
    when: always
    expire_in: one week
  allow_failure: true

sec-dast_full:
  stage: sec-release
  image:
    name: owasp/zap2docker-weekly
  before_script:
    - mkdir /zap/wrk
  script:
    - zap-full-scan.py -t https://hapi.f5labs.dev/fhir -I -J dast_full_scan-results.json
  after_script:
    - mv /zap/wrk/dast_full_scan-results.json /builds/shainsingh/hapi-fhir/
  rules:
    - if: $scan_periodic == "nightly"
      when: always
  artifacts:
    paths: [dast_full_scan-results.json]
    when: always
    expire_in: one week
  allow_failure: true
```


Example – Post deployment compliance





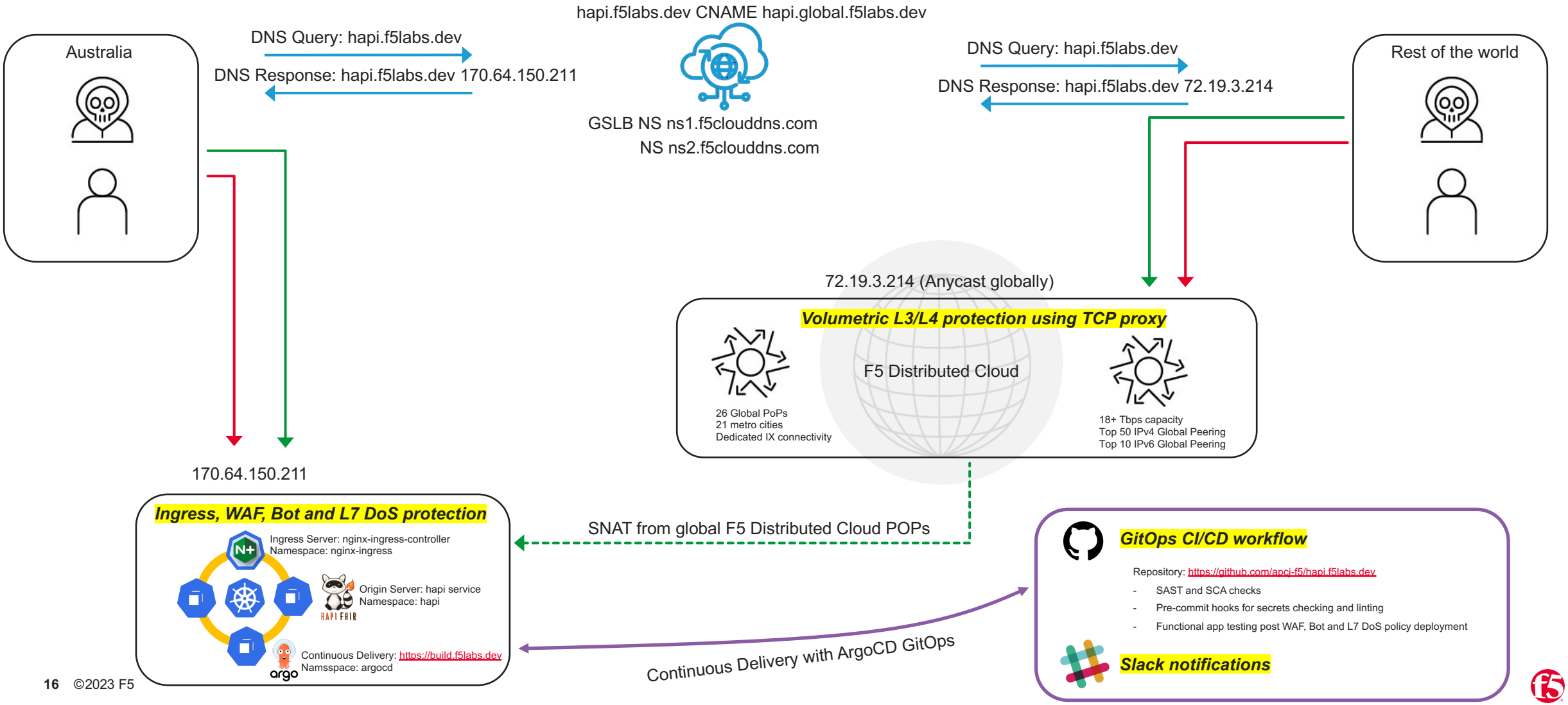
sec-package.gitlab-ci.yml 760 Bytes — Edit | Web IDE

```yaml
services:
  - docker:dind

sec-os_hardening:
  stage: sec-package
  image: ansible/galaxy
  before_script:
    - mkdir -p ~/.ssh
    - echo "$DEPLOYMENT_SERVER_SSH_PRIVKEY" | tr -d '\r' > ~/.ssh/id_rsa
    - chmod 600 ~/.ssh/id_rsa
    - eval "$(ssh-agent -s)"
    - ssh-add ~/.ssh/id_rsa
    - echo -e "Host *\n\tStrictHostKeyChecking no\n\n" > ~/.ssh/config
  script:
    - echo "[prod]" >> inventory.ini
    - echo "$DEPLOYMENT_SERVER" >> inventory.ini
    - export ANSIBLE_STDOUT_CALLBACK=json
    - ansible-galaxy install dev-sec.os-hardening
    - ansible-playbook -i inventory.ini ansible-hardening.yml > sec-os_hardening-results.json
  artifacts:
    paths: [sec-os_hardening-results.json]
    when: always
    expire_in: one week
  allow_failure: true
```

sec-compliance.gitlab-ci.yml 694 Bytes — Edit | Web IDE | Lock | Replace | Delete

```yaml
services:
  - docker:dind

sec-compliance:
  stage: sec-compliance
  image:
    name: chef/inspec
  only:
    - "master"
  environment: production
  before_script:
    - mkdir -p ~/.ssh
    - echo "$DEPLOYMENT_SERVER_SSH_PRIVKEY" | tr -d '\r' > ~/.ssh/id_rsa
    - chmod 600 ~/.ssh/id_rsa
    - eval "$(ssh-agent -s)"
    - ssh-add ~/.ssh/id_rsa
    - echo -e "Host *\n\tStrictHostKeyChecking no\n\n" > ~/.ssh/config
  script:
    - inspec exec https://github.com/dev-sec/linux-baseline -t ssh://root@$DEPLOYMENT_SERVER -i /id_rsa --chef-license accept --reporter json:/opt/sec-
  artifacts:
    paths: [sec-compliance-results.json]
    when: always
  allow_failure: true
```


Reference Implementation – hapi.f5labs.dev



A force for a better digital world