



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

Hi! Nice to meet you



Shain Singh

Principal Security Architect
ss@f5.com

 <https://github.com/shsingh>
 <https://www.linkedin.com/in/shsingh/>



Leon Seng

ANZ Modern Apps Lead
leon@f5.com

 <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

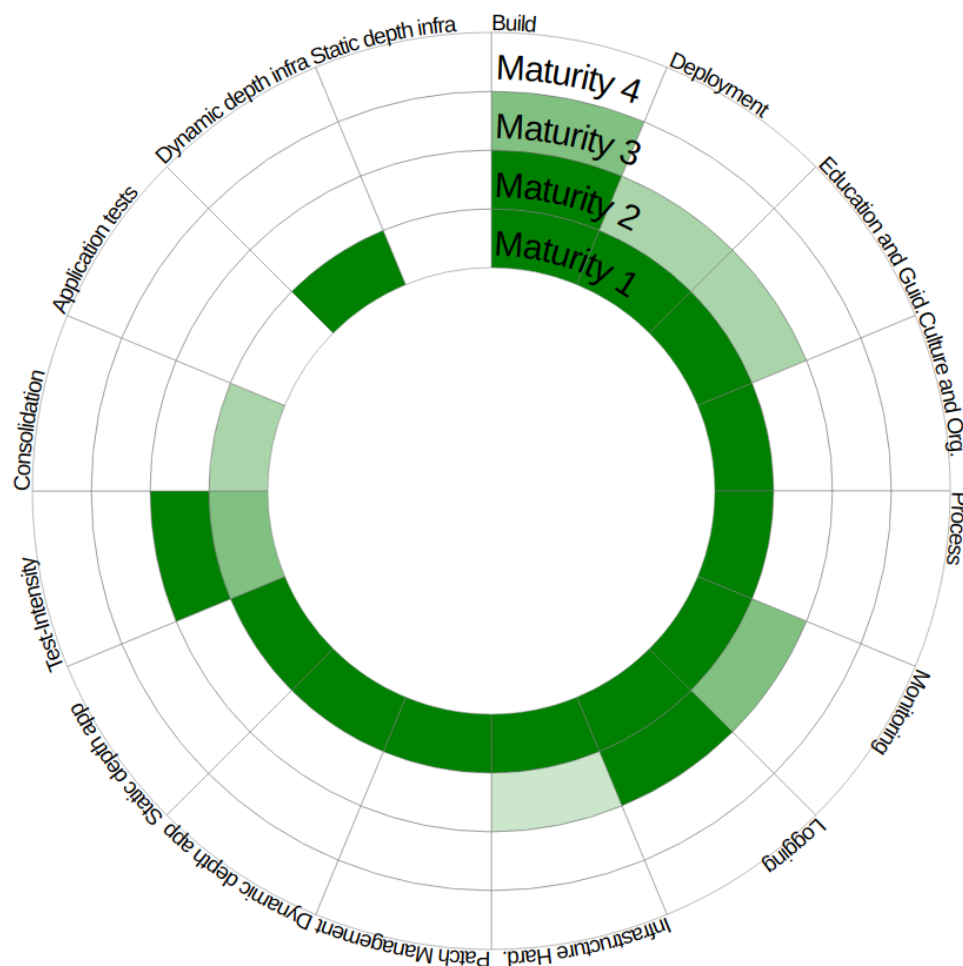
96% Of cloud breaches are self-inflicted

Source: Osterman Research

Why have we not made much progress?

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

Identification of the degree of the implementation



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

Compliance can be continuous and automated



Cloud Controls Matrix
Security Guidance For Critical Areas of Focus in Cloud Computing



Benefits, Risks and Recommendations For Information Security



Cybersecurity Framework

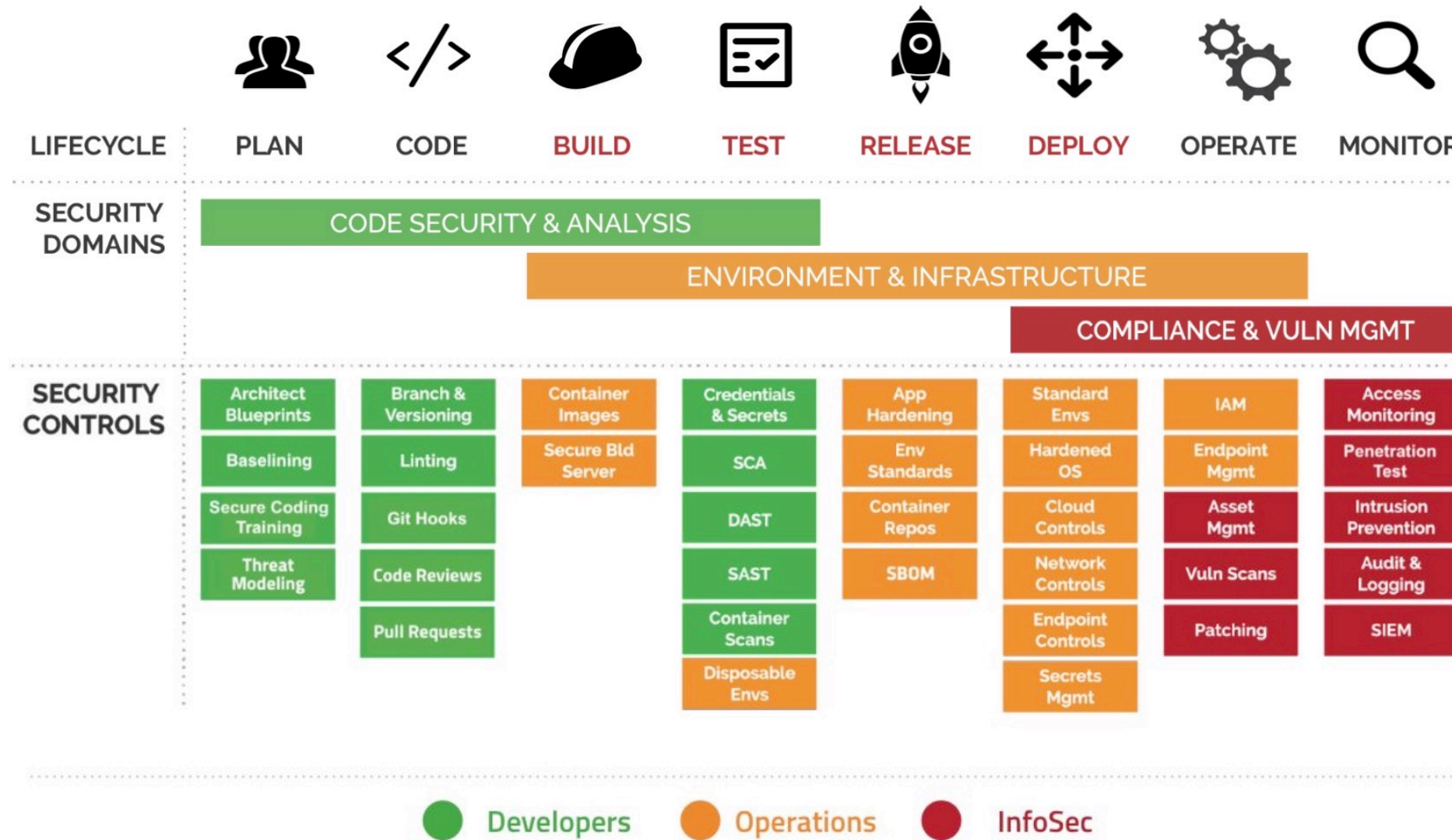


Secure Cloud Computing Architecture

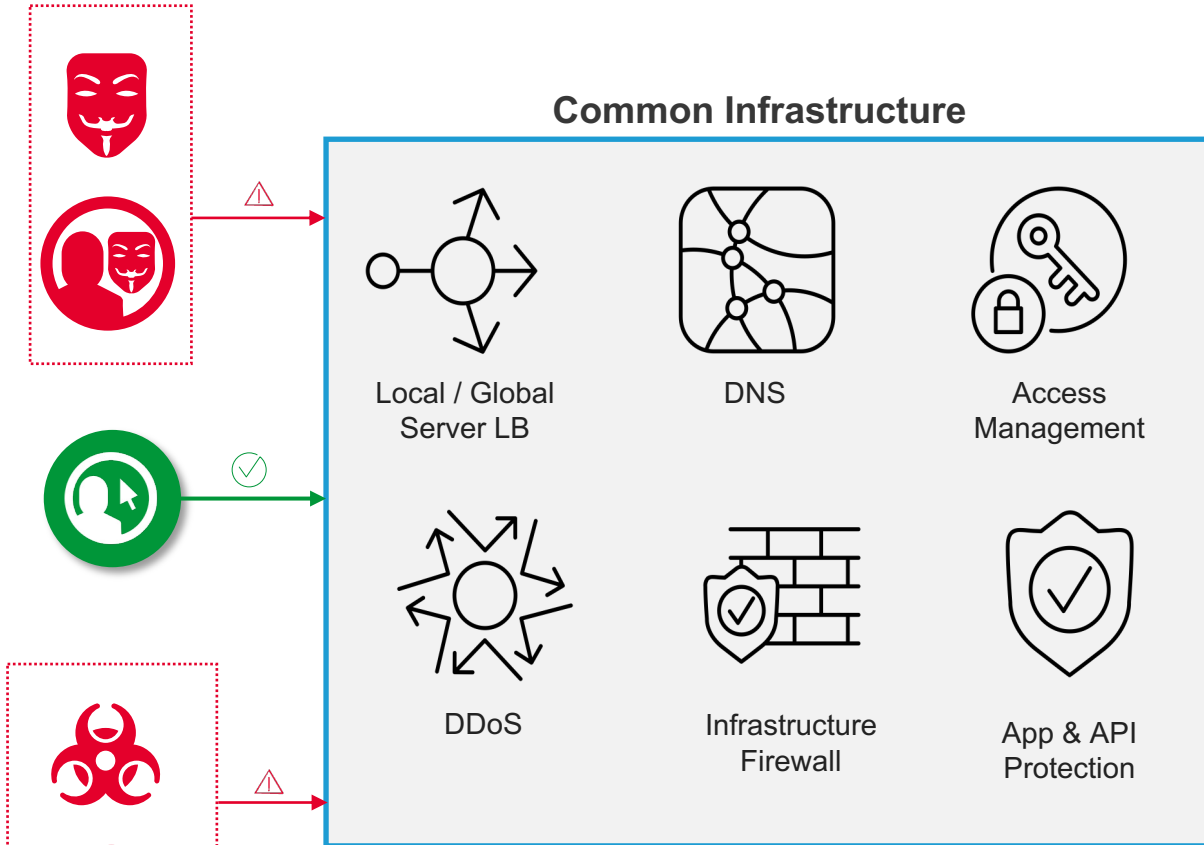


CIS Benchmarks

Separating tasks into domains of ownership



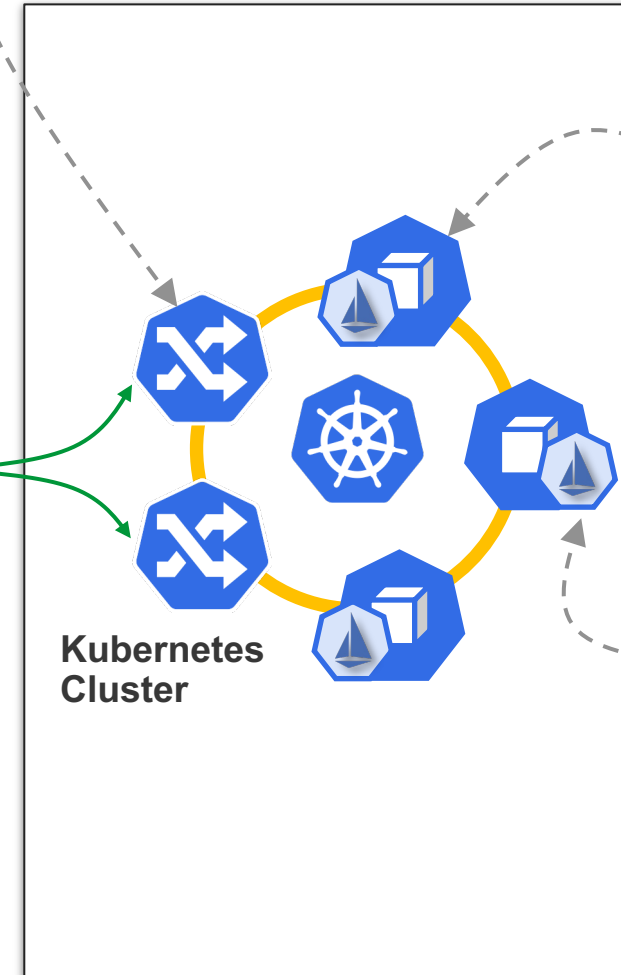
I thought F5 made load balancers – what do they have to do with DevSecOps?



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

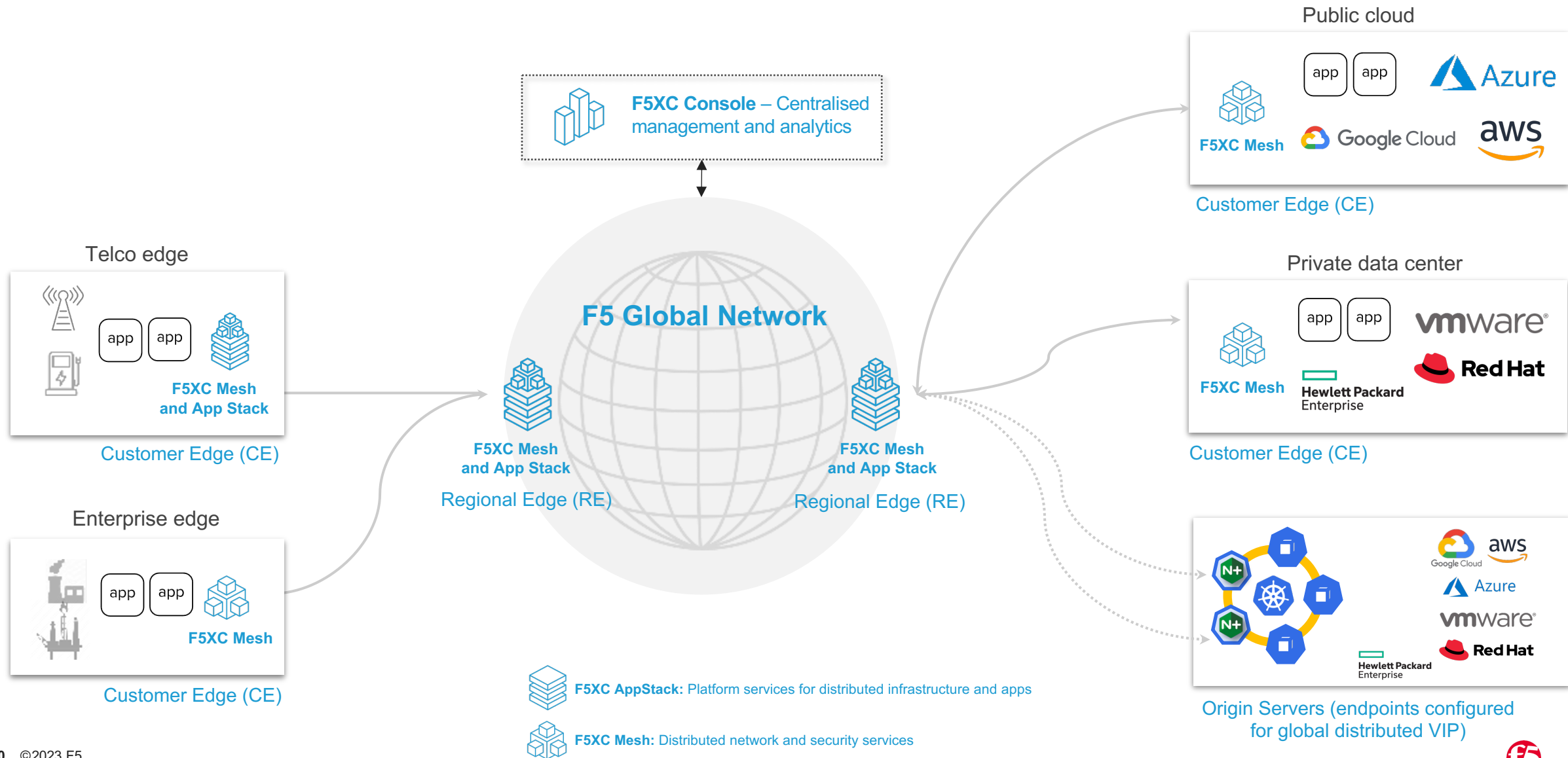
Service Mesh

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

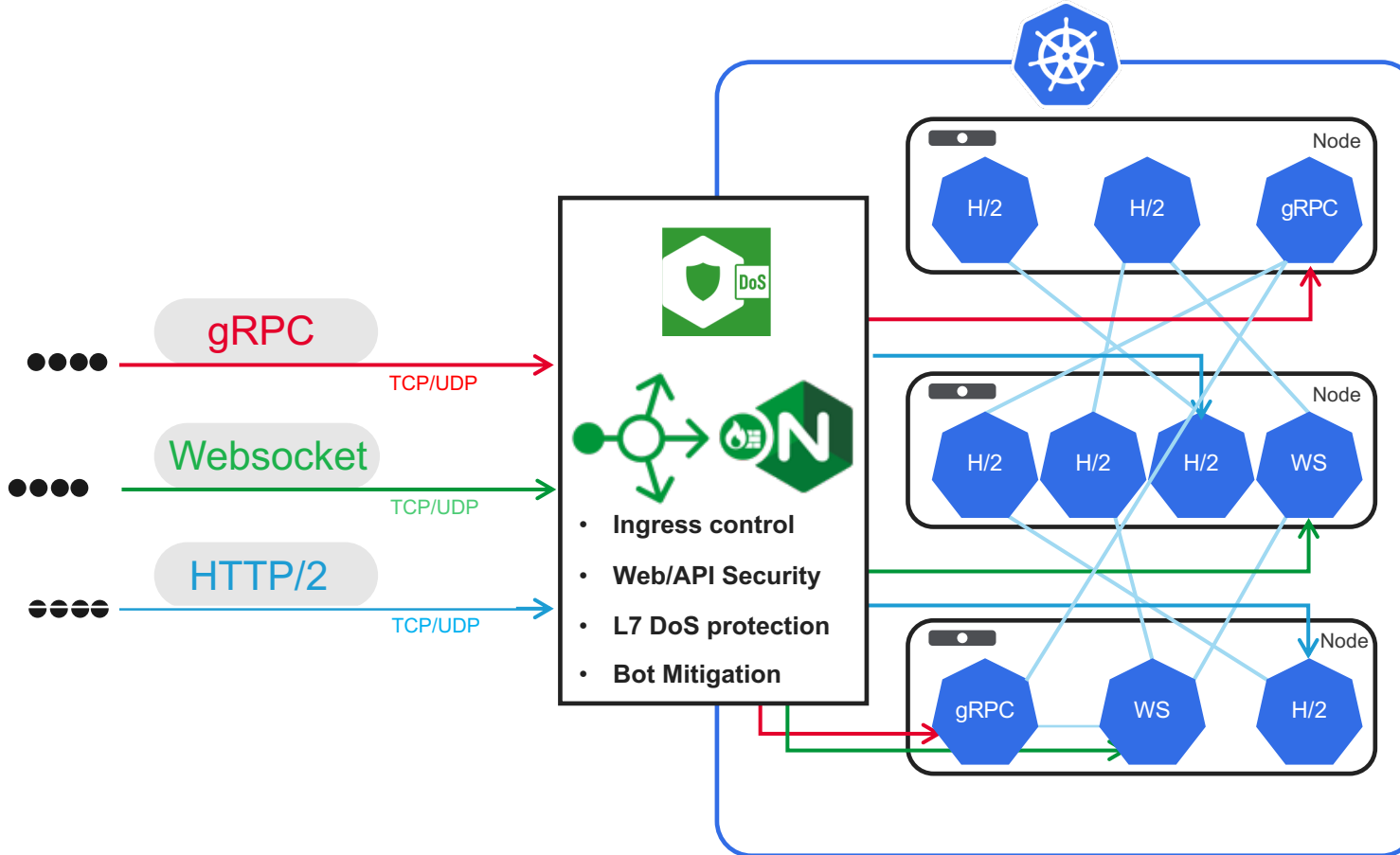
Cloud Microservices PaaS

- 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
- Bot Mitigation
- 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
 - (Kubernetes native manifest)

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)

“Do I need encrypted data at rest and in transit for internal apps”

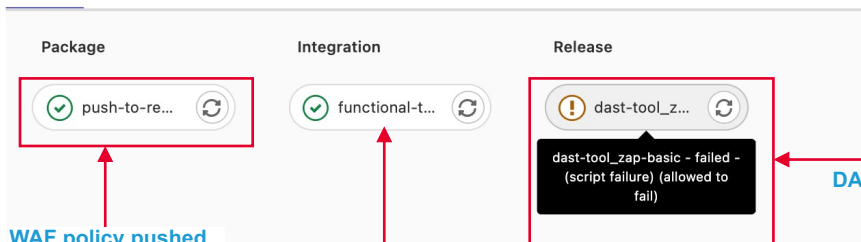
- Secrets management (e.g Hashicorp Vault)
- mTLS via service mesh
- LetsEncrypt Certbot for TLS certificates

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

Pipeline Needs Jobs 3 Failed Jobs 1 Tests 0



WAF policy pushed

Functional application testing

DAST post-release

```
awaf_policy-hapi-fhir.json 766 Bytes
1 {
2   "policy": {
3     "name": "policy-hapi-fhir",
4     "description": "HAPI Policy for FHIR",
5     "template": {
6       "name": "POLICY_TEMPLATE_API_SECURITY"
7     },
8     "enforcementMode": "blocking",
9     "server-technologies": [
10      {
11        "serverTechnologyName": "MySQL"
12      },
13      {
14        "serverTechnologyName": "Unix/Linux"
15      }
16    ],
17    "signature-settings": {
18      "signatureStaging": false
19    },
20    "policy-builder": {
21      "learnOnlyFromNonBotTraffic": false
22    },
23    "open-api-files": [
24      {
25        "link": "https://gitlab.com/shainsingh/hapi-fhir/-/raw/master/hapi-fhir_swagger.json"
26      }
27    ]
28  }
29 }
```

```
sec-release.gitlab-ci.yml 1 KB
1 services:
2   - docker:dind
3
4 sec-dast_baseline:
5   stage: sec-release
6   image:
7     name: owasp/zap2docker-weekly
8   before_script:
9     - mkdir /zap/wrk
10  script:
11    - zap-baseline.py -t https://hapi.f5labs.dev/fhir -I -J dast_baseline_scan-results.json
12  after_script:
13    - mv /zap/wrk/dast_baseline_scan-results.json /builds/shainsingh/hapi-fhir/
14  rules:
15    - if: $scan_periodic != "nightly"
16      when: always
17  artifacts:
18    paths: [dast_baseline_scan-results.json]
19    when: always
20    expire_in: one week
21    allow_failure: true
22
23 sec-dast_full:
24   stage: sec-release
25   image:
26     name: owasp/zap2docker-weekly
27   before_script:
28     - mkdir /zap/wrk
29  script:
30    - zap-full-scan.py -t https://hapi.f5labs.dev/fhir -I -J dast_full_scan-results.json
31  after_script:
32    - mv /zap/wrk/dast_full_scan-results.json /builds/shainsingh/hapi-fhir/
33  rules:
34    - if: $scan_periodic == "nightly"
35      when: always
36  artifacts:
37    paths: [dast_full_scan-results.json]
38    when: always
39    expire_in: one week
40    allow_failure: true
```

Example – Post deployment compliance

running

#308839490

latest

master -> b004c8a0

remove previous sec ci sta...

✓

✗

»

»

»

sec-compliance: passed with warnings

In progress

passed

#308837380

latest

master -> b004c8a0

remove previous sec ci sta...

✓

!

✓

!

00:06:53

14 minutes ago

Pipeline Needs Jobs 4 Failed Jobs 2 Tests 0

Sec-pre_build

Sec-package

Sec-release

Sec-compliance

✓

sec-source_...

↺

!

sec-os_hard...

↺

✓

sec-dast_ba...

↺

!

sec-complia...

↺

sec-package.gitlab-ci.yml 760 Bytes

Edit Web IDE

```
1 services:
2   - docker:dind
3
4 sec-os_hardening:
5   stage: sec-package
6   image: ansible/galaxy
7   before_script:
8     - mkdir -p ~/.ssh
9     - echo "$DEPLOYMENT_SERVER_SSH_PRIVKEY" | tr -d '\r' > ~/.ssh/id_rsa
10    - chmod 600 ~/.ssh/id_rsa
11    - eval "$(ssh-agent -s)"
12    - ssh-add ~/.ssh/id_rsa
13    - echo -e "Host *\n\tStrictHostKeyChecking no\n\n" > ~/.ssh/config
14  script:
15    - echo "[prod]" >> inventory.ini
16    - echo "$DEPLOYMENT_SERVER" >> inventory.ini
17    - export ANSIBLE_STDOUT_CALLBACK=json
18    - ansible-galaxy install dev-sec.os-hardening
19    - ansible-playbook -i inventory.ini ansible-hardening.yml > sec-os_hardening-results.json
20  artifacts:
21    paths: [sec-os_hardening-results.json]
22    when: always
23    expire_in: one week
24    allow_failure: true
```

sec-compliance.gitlab-ci.yml 694 Bytes

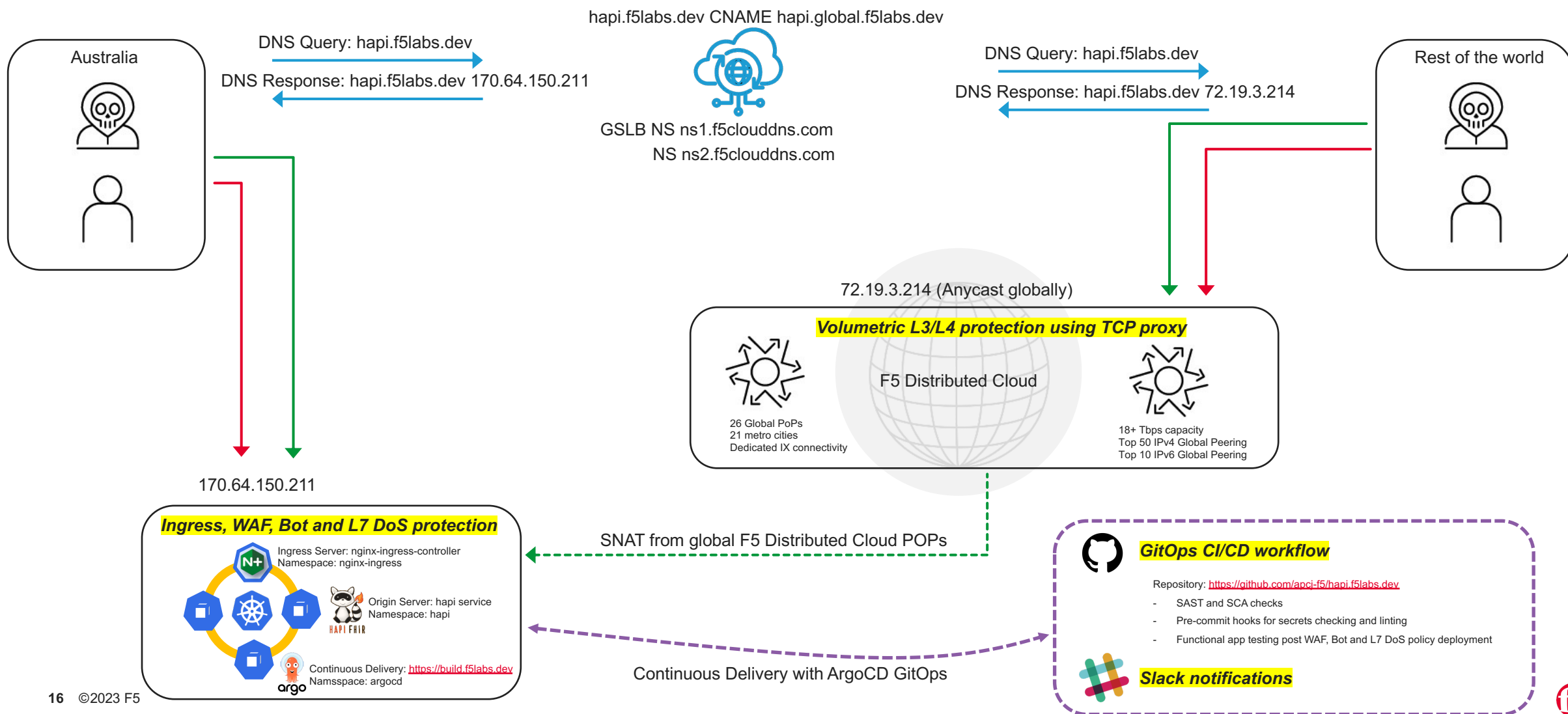
Edit Web IDE Lock Replace Delete



```
1 services:
2   - docker:dind
3
4 sec-compliance:
5   stage: sec-compliance
6   image:
7     name: chef/inspec
8  only:
9    - "master"
10  environment: production
11  before_script:
12    - mkdir -p ~/.ssh
13    - echo "$DEPLOYMENT_SERVER_SSH_PRIVKEY" | tr -d '\r' > ~/.ssh/id_rsa
14    - chmod 600 ~/.ssh/id_rsa
15    - eval "$(ssh-agent -s)"
16    - ssh-add ~/.ssh/id_rsa
17    - echo -e "Host *\n\tStrictHostKeyChecking no\n\n" > ~/.ssh/config
18  script:
19    - inspec exec https://github.com/dev-sec/linux-baseline -t ssh://root@$DEPLOYMENT_SERVER -i /id_rsa --chef-license accept --reporter json:/opt/sec-
20  artifacts:
21    paths: [sec-compliance-results.json]
22    when: always
23    allow_failure: true
```



Reference Implementation – hapi.f5labs.dev



Key Takeaways

- Remember the “People” and “Process” portions of DevOps
- Start small, then increment - DSOMM Level 1
- Apply declarative WAF policies for use in pipelines
- Intent is to have security across all apps, everywhere
- A manual WAF policy in transparent mode may be less effective than an automated declarative policy in blocking mode*



A force for a better digital world