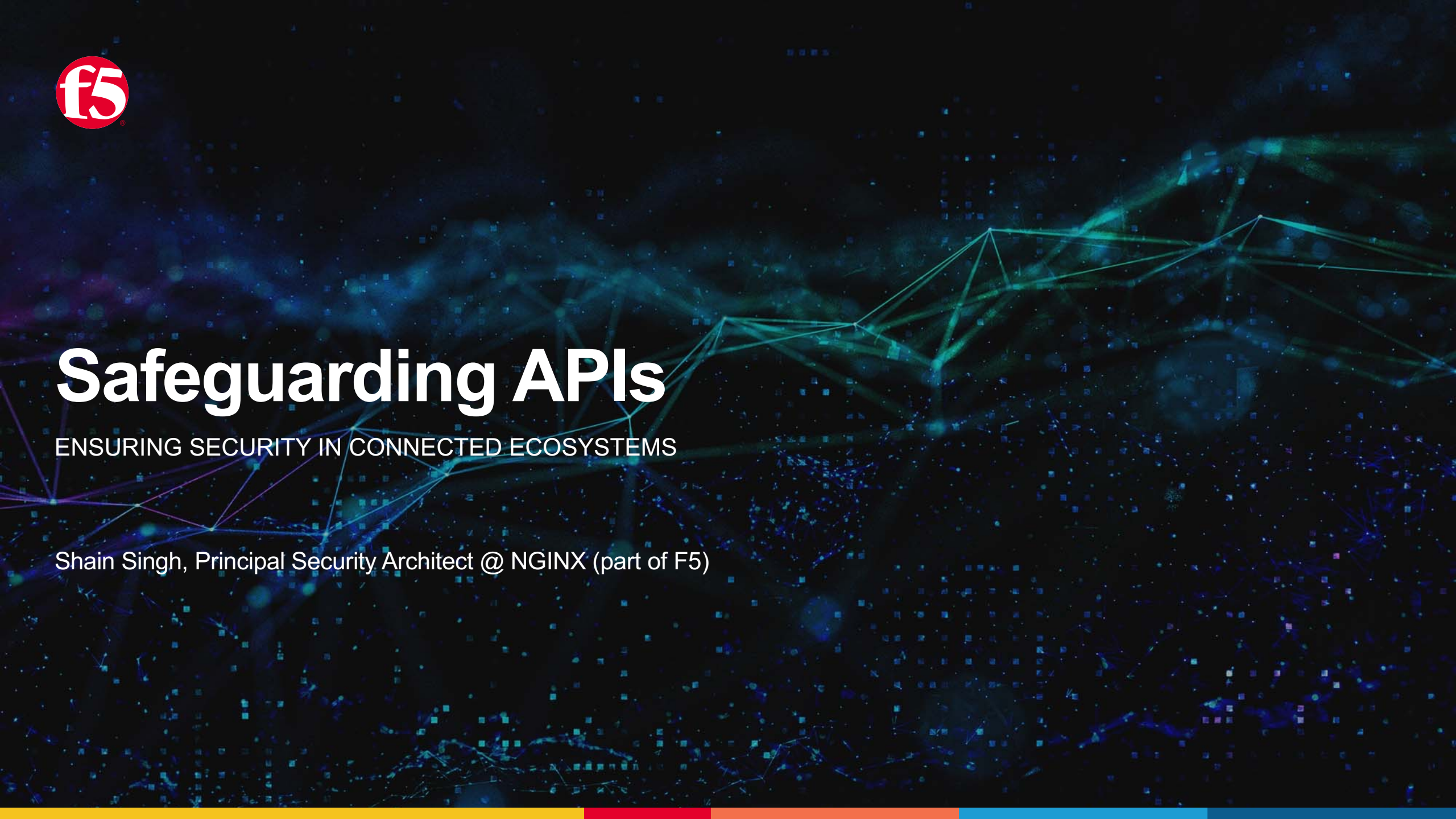
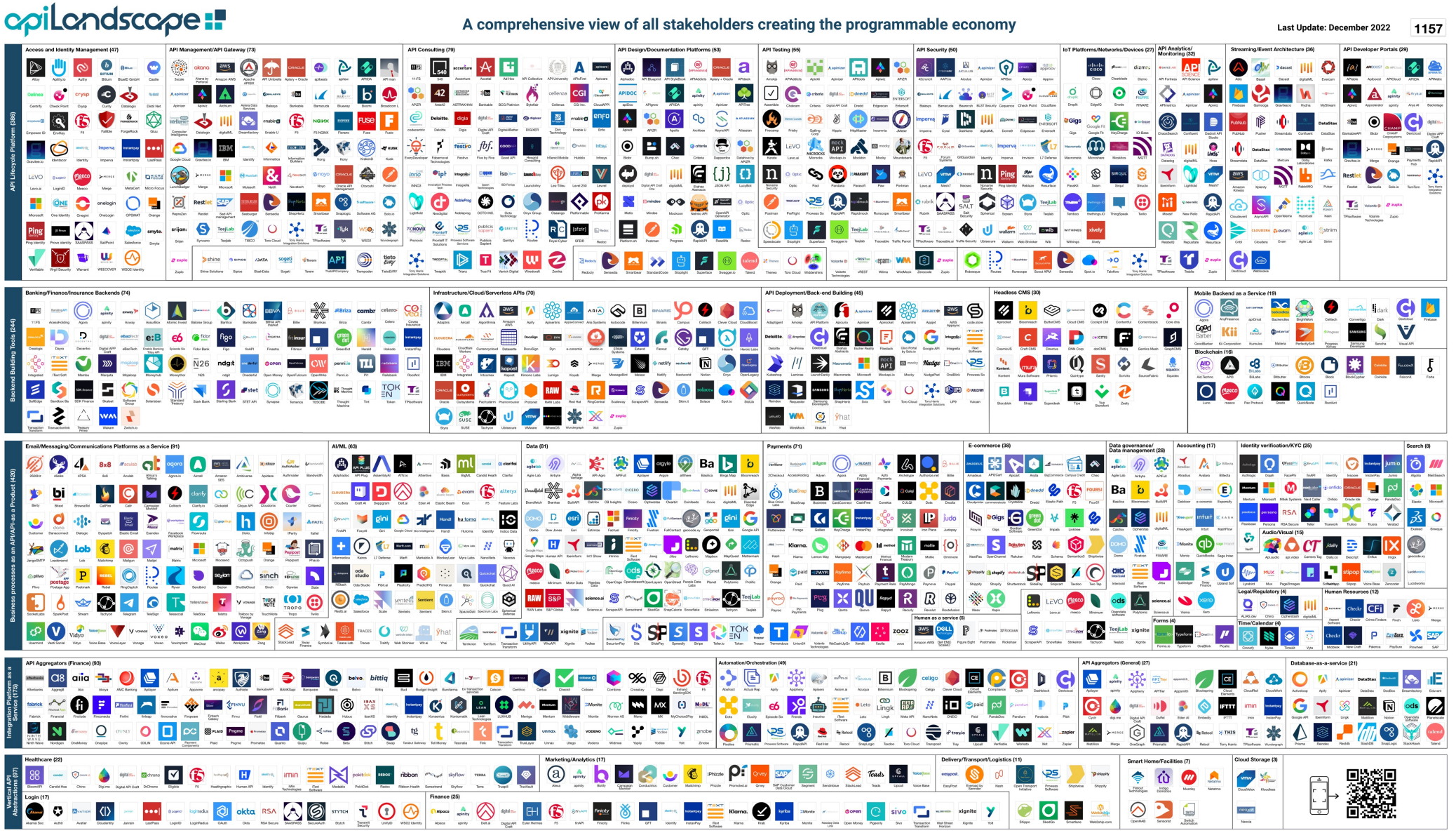


Safeguarding APIs

ENSURING SECURITY IN CONNECTED ECOSYSTEMS

Shain Singh, Principal Security Architect @ NGINX (part of F5)

How we tend to think of the API ecosystem



SecOps need to improve for enabling speed

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

81% Of developers admit to pushing code with known vulnerabilities

96% Of cloud breaches are self-inflicted

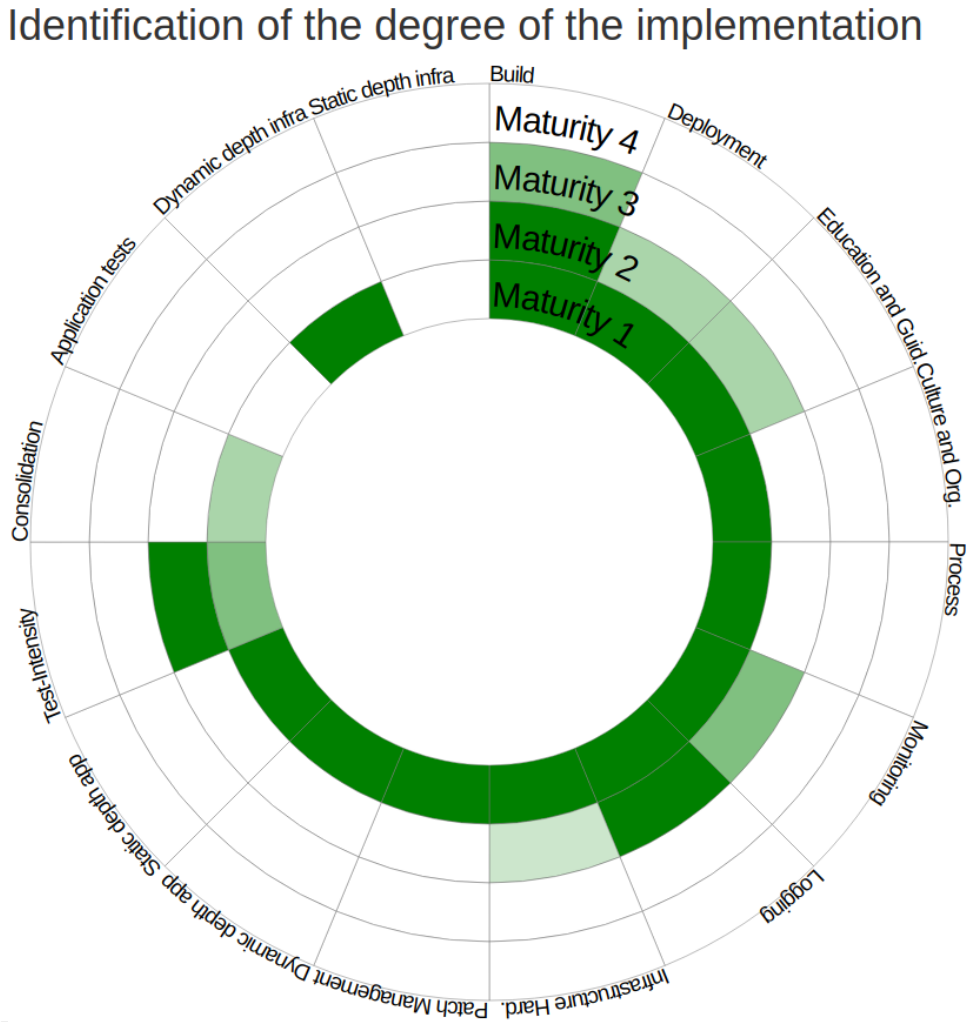
Source: Immersive Labs

curl –L whois.shain.io


```
{
  "$schema": "https://raw.githubusercontent.com/jsonresume/resume-
schema/v1.0.0/schema.json",
  "basics": {
    "name": "Shain Singh",
    "label": "Principal Security Architect, OCTO, OSPO @ F5 | Project Co-Lead @
OWASP"
  },
  "profiles": [
    {
      "network": "LinkedIn",
      "url": "https://www.linkedin.com/in/shsingh/"
    },
    {
      "network": "Twitter",
      "url": "https://twitter.com/shainsingh"
    },
    {
      "network": "Github",
      "url": "https://github.com/shsingh"
    }
  ],
  "work": [
    {
      "name": "OWASP® Foundation",
      "position": [
        "Project Co-Lead - Machine Learning Security Top 10",
        "Project Co-Lead - Machine Learning Security Verification Standard"
      ]
    },
    {
      "name": "Cloud Security Alliance",
      "position": [
        "DevSecOps Working Group",
        "Zero Trust Working Group"
      ]
    },
    {
      "name": "F5",
      "position": [
        "Open Source Program Office (OSPO) - Open Source Advocate",
        "Office of CTO (OCTO) - Innovation Ambassador",
        "Principal Security Architect [APCJ]"
      ]
    }
  ]
}
```


Fine tuning versus Comprehensive Coverage

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

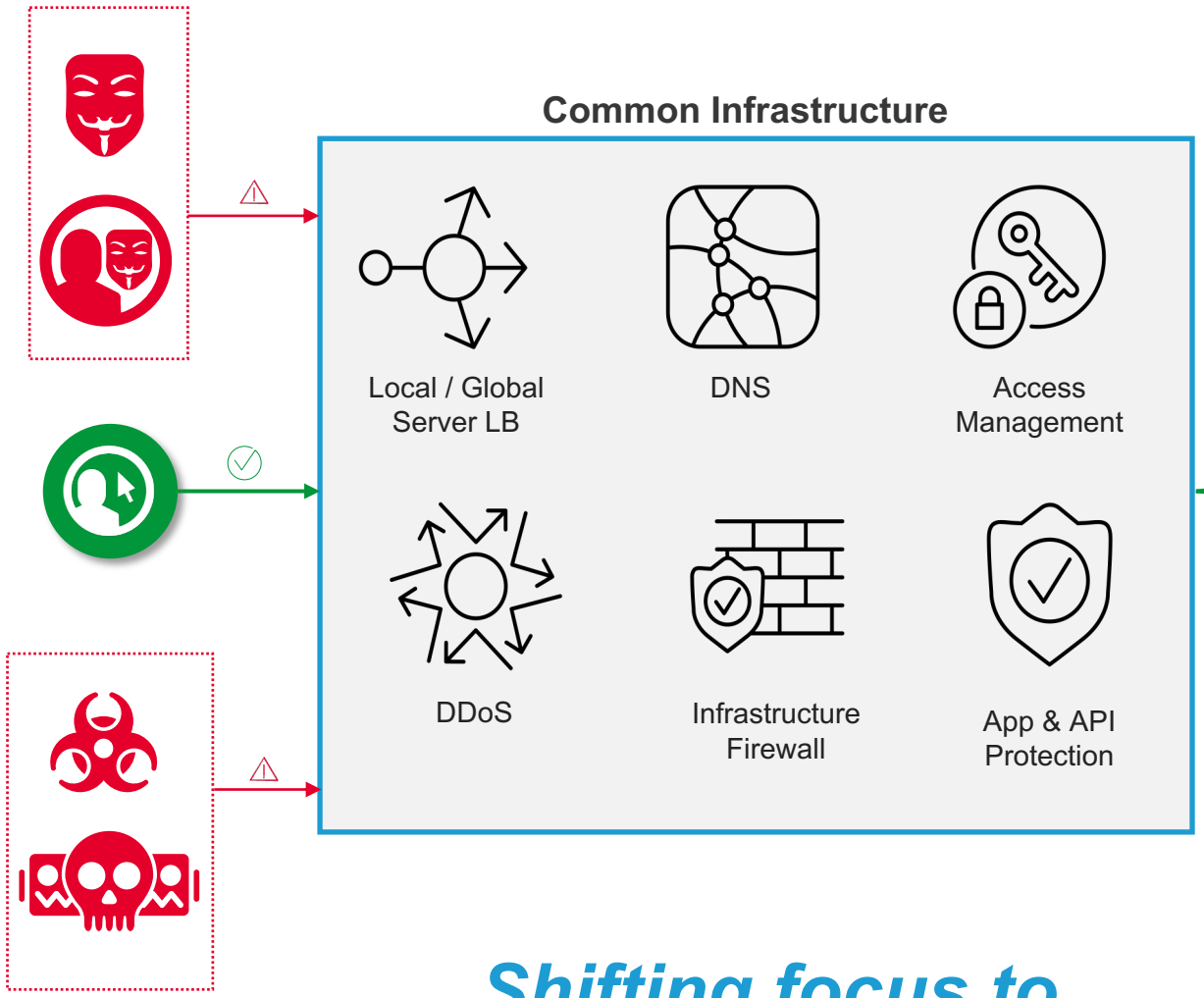
DevSecOps Maturity Model (DSOMM) Level 2
Adoption of basic security practices

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

Runtime environment for applications



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

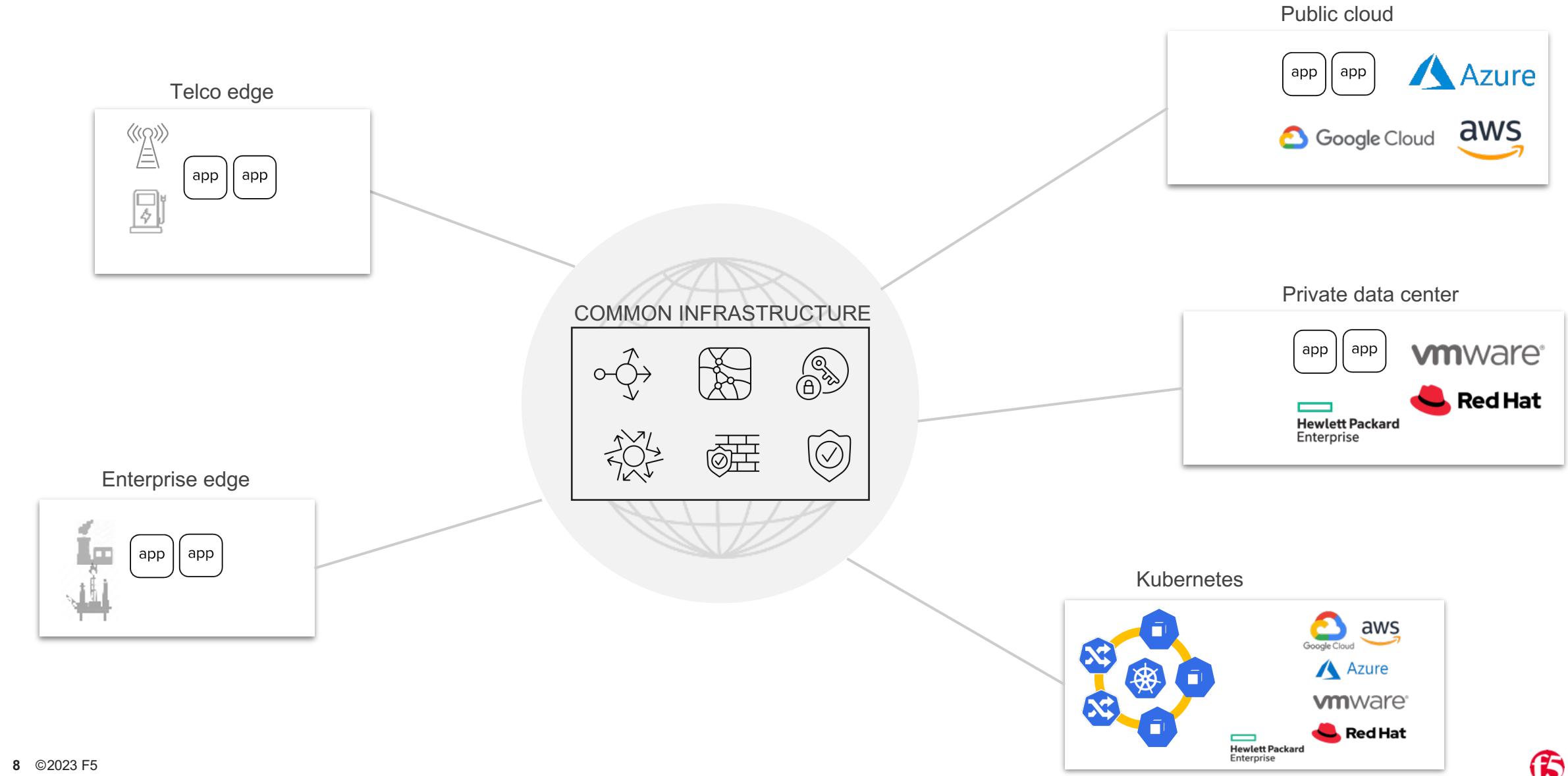
Kubernetes Cluster

Cloud Microservices PaaS

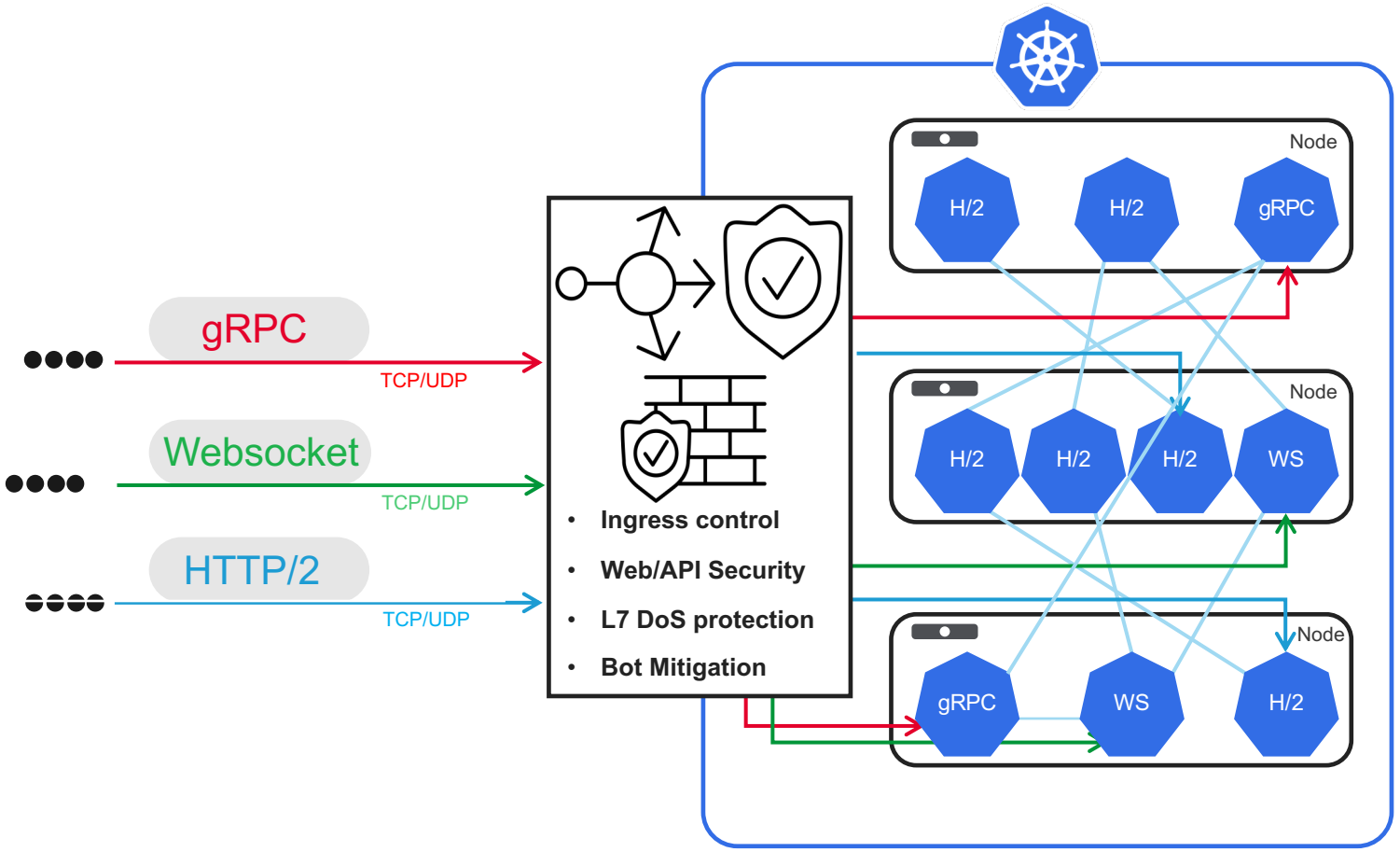
- On-prem private cloud (e.g: VMware)
- Public cloud (e.g: AWS, Azure, GCP)

Shifting focus to post-deployment

App Services at a macro-environment level



App Services at a micro-environment level



Security Controls:

- 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
- AuthN/AuthZ

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, AWS KMS)
- mTLS via service mesh
- LetsEncrypt Certbot for TLS certificates

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

- WAF and L7 DoS configuration via Kubernetes manifests, deployed via Continuous Delivery tooling
- Functional testing of application post WAF deployment removes potential for false positives

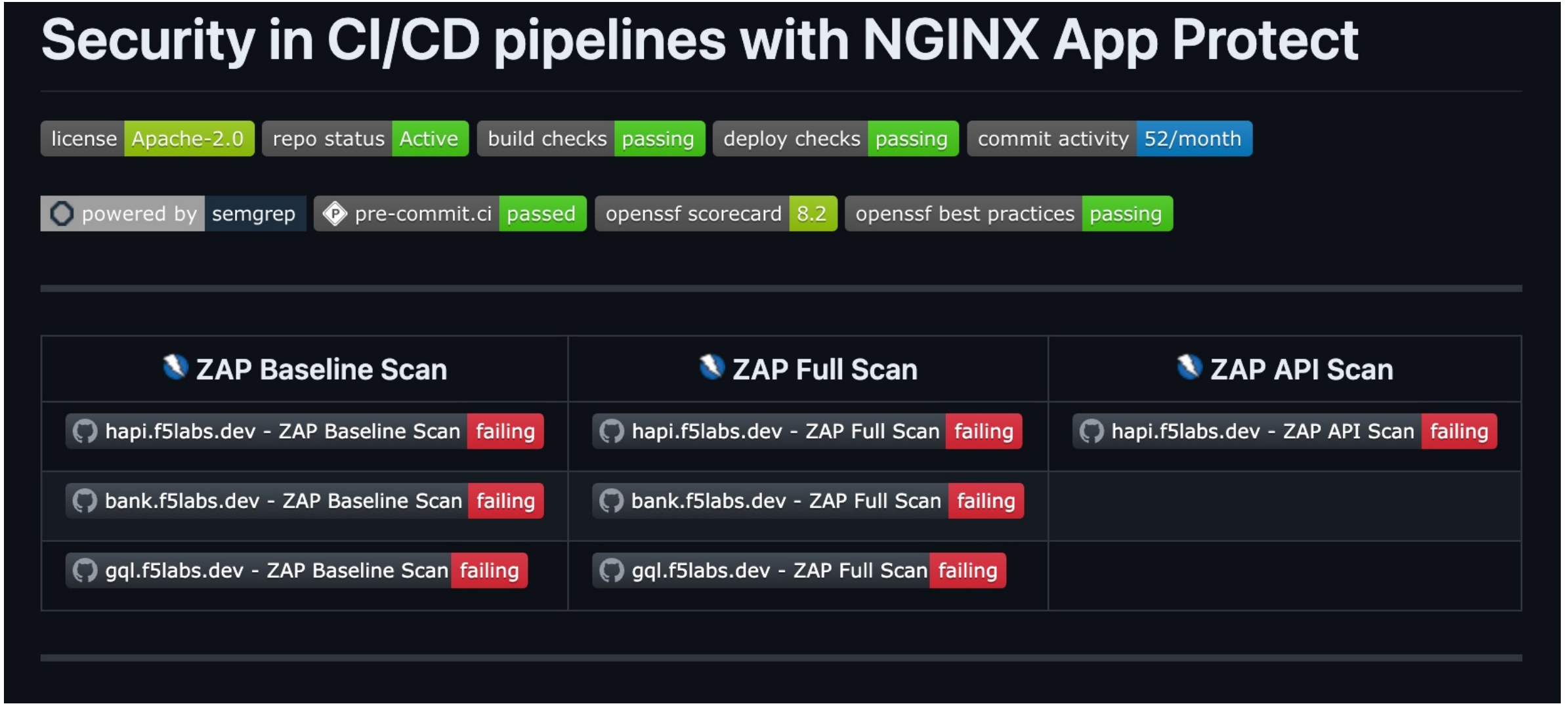
Demonstration

https://github.com/apcj-f5/nap-devsecops-demo

Links for Demonstration

- GitOps repository
  - https://github.com/apcj-f5/nap-devsecops-demo
- Continuous Delivery
  - https://build.f5labs.dev
- Monitoring
  - https://ops.f5labs.dev
- Apps
  - Health Records API https://hapi.f5labs.dev
  - Bank https://bank.f5labs.dev
  - GraphQL https://gql.f5labs.dev

Repository with Security Controls



Application Security Policy applied at Ingress

nap-devsecops-demo / argocd / manifests / bank / **bank-ingress-policy.yaml**

shsingh shsingh/enhance waf for bank app (#229)

Code | Blame | 12 lines (12 loc) · 265 Bytes | Code 55% faster with GitHub Copilot

```yaml
apiVersion: k8s.nginx.org/v1
kind: Policy
metadata:
  name: "bank-ingress-policy"
spec:
  waf:
    enable: true
    apPolicy: "bank-waf-default-blocking-policy"
    securityLogs:
      - enable: true
        apLogConf: "bank-waf-log-policy"
        logDest: stderr
```

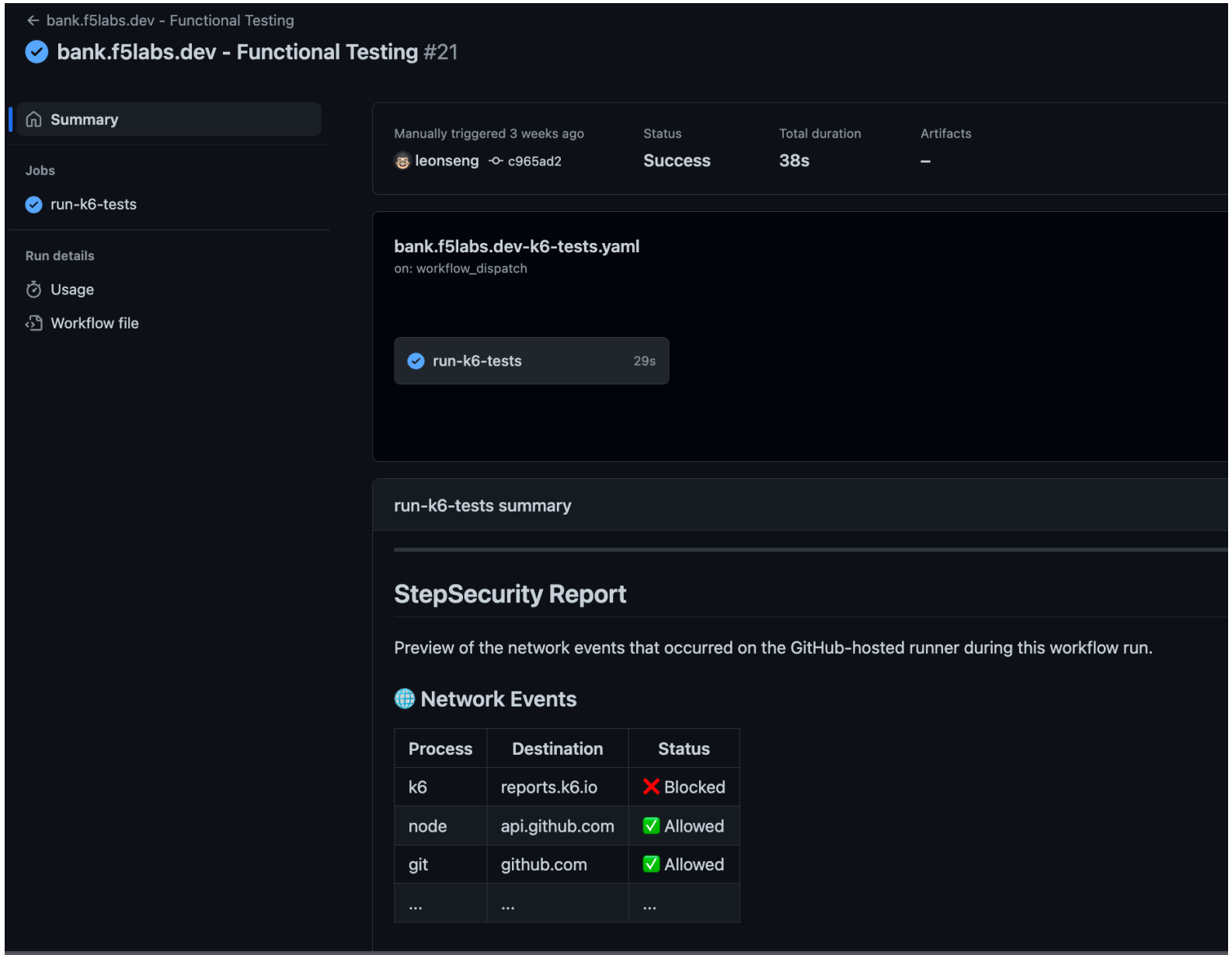
nap-devsecops-demo / argocd / manifests / bank / **bank-waf-default-blocking-policy.yaml**

shsingh custom responsepage works - change back to default | 7fd74cd · last month | History

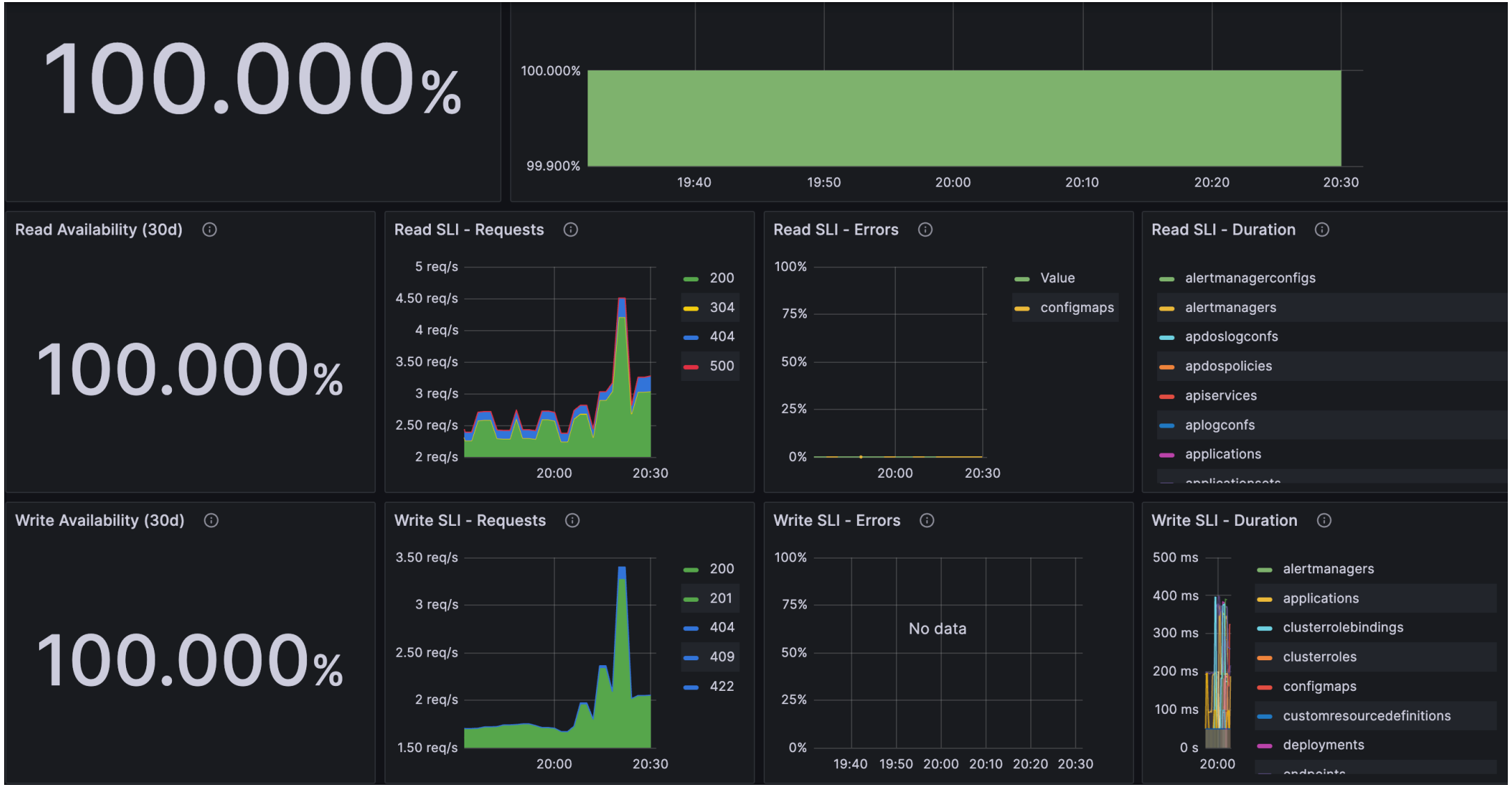
Code | Blame | 16 lines (16 loc) · 620 Bytes | Code 55% faster with GitHub Copilot | Raw

```yaml
apiVersion: appprotect.f5.com/v1beta1
kind: APPolicy
metadata:
  name: "bank-waf-default-blocking-policy"
spec:
  policy:
    name: "bank-waf-default-blocking-policy"
    template:
      name: POLICY_TEMPLATE_NGINX_BASE
    applicationLanguage: utf-8
    enforcementMode: blocking
    # response-pages:
    # - responsePageType: default
    # responseActionType: custom
    # responseContent: "The request URL was rejected. Please consult with your administrator. Your support ID is: <%TS.request.ID()%>"
    # responseHeader: "HTTP/1.1 403 Forbidden\r\nCache-Control: no-cache\r\nPragma: no-cache\r\nConnection: close"
```


Functional Tests with Security enforcement



Application Metrics Monitoring



Key Takeaways

- Start incorporating runtime environment controls into pipelines for feedback loops
- Start small, then increment – complete coverage over fine tuning
- Integrate into DevOps processes as opposed to just installing security tooling
- Goal is to have security across all apps, everywhere

A force for a better digital world