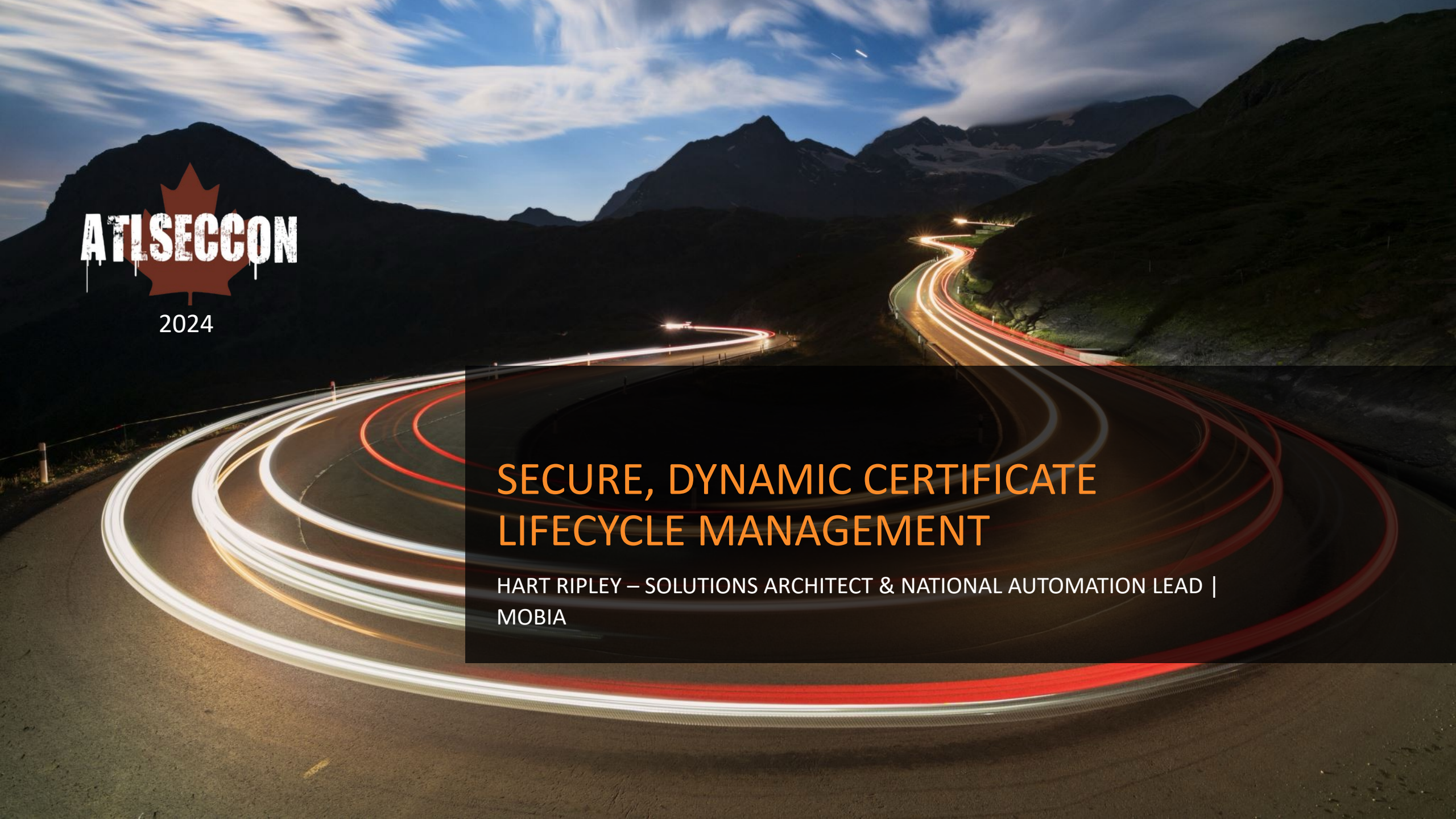


SECURE, DYNAMIC CERTIFICATE LIFECYCLE MANAGEMENT

HART RIPLEY – SOLUTIONS ARCHITECT & NATIONAL AUTOMATION LEAD | MOBIA

AGENDA

- Why should we automate (and forget) certificate management
- Developers and certificates on-demand/self-serve
- Auto rotation of certificates
- Kubernetes-native workflows & certificate requirements
- Custom domain certificates
- Kubernetes platform certificates
- Multiple certificate providers for workload types
- Dev certificates vs. Production
- Certificates shared across Kubernetes namespaces

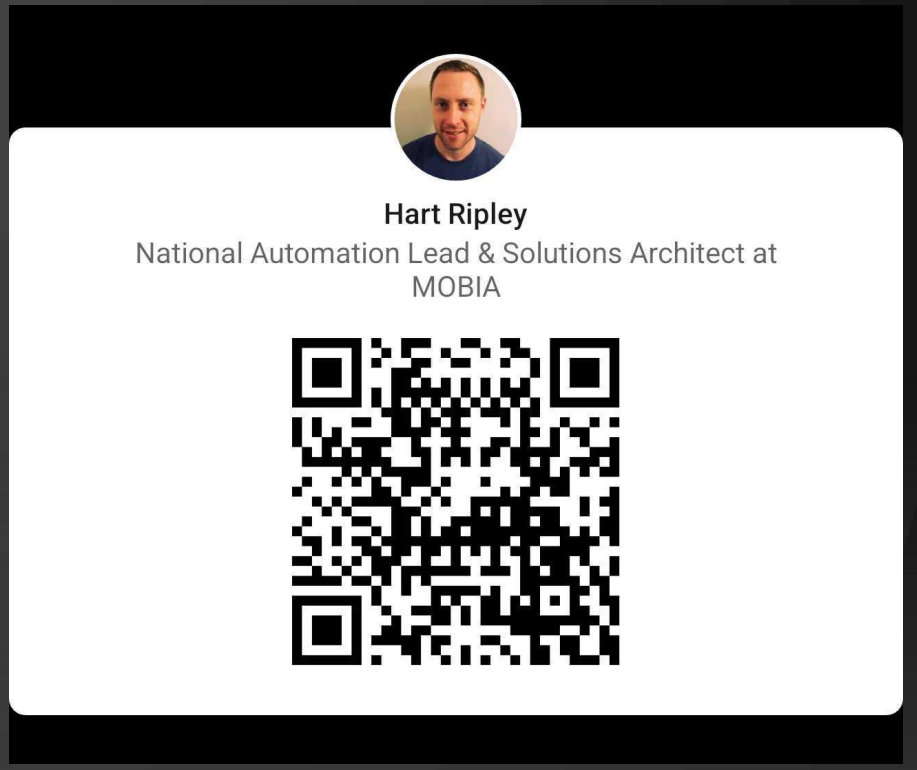
WHOAMI

Networking administration and operations background

Hands-on technical experience

DevOps and automation meets networking and infrastructure

3 Years at MOBIA as Solutions Architect



WHY CERTIFICATE AUTOMATION FOR KUBERNETES?

- You run applications and care about security
- Operators/administrators prefer to sleep instead of rotate certificates
- You like to standardize on process and tooling
- DevOps and automation are part of your culture
- You want to empower platform consumers, AppDev teams to operate in a self-service and secure manner (real DevOps)

COMMON CERTIFICATE OPERATIONAL HEADACHES

- Short-lived validity (when manually managing lifecycle) – no auto rotation
- SAN / DNS Alt Name adds/changes
- Multiple certificate authorities (CA's) or certificate providers to manage
- Sandboxing/dev/lab environments
- Custom domain certificates
- Manual effort and administration for operations and AppDev teams
- Repetitive task: good use-case for automation
- Cleanup unused/expired certificates/CRL's (usually not done)

WHY ARE CERTIFICATES IMPORTANT | COMMON USE-CASES

Secure transmission (TLS)

Trust and authenticity

Testing websites, front-ends, Load Balancers, Ingress, Gateways

Using certificates is a formality, lifecycle management is an afterthought

Ephemeral workloads need to be secure

DR/failover certificate issuance/management/revocation

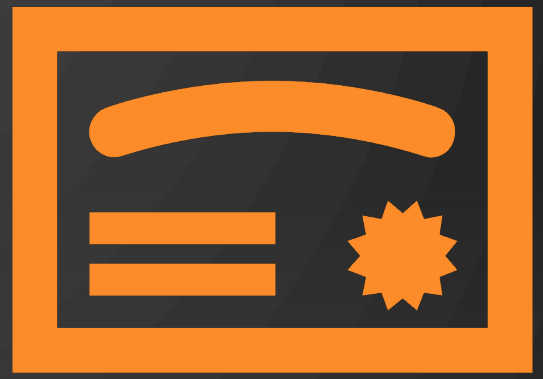
Platform certificates and lifecycle management

WHY AUTOMATE

- Web portal for self-service certificate requests
- Tools like Ansible orchestrate certificate deployment, testing, rotation across entire platform landscape
- Tenants and platform consumers manage their own certificate lifecycle
- Cert-manager, GitOps as standardized, multi-platform/hybrid tools for certificate lifecycle management

TRUE PAAS

- Certificate Management as a Service
  - Tenant and users of the platform get least privileged ability to provision and leverage certificates within their namespace(s)
  - Platform operators manage cross-namespace or shared certificates
  - Wildcard certificates used for all environment-specific suffixed workloads/applications running on the platform – annotations & labels used by application teams to leverage these certificates
  - Monitoring, metrics, and alerting



CERTIFICATE LIFECYCLE PERSONAS

- Developers
- Operators/Administrators
- Platform(s)
- Services
- Automation tools
- Readers
- Endpoints
- Bots

MEET OUR CERTIFICATE CONSUMERS

DEV

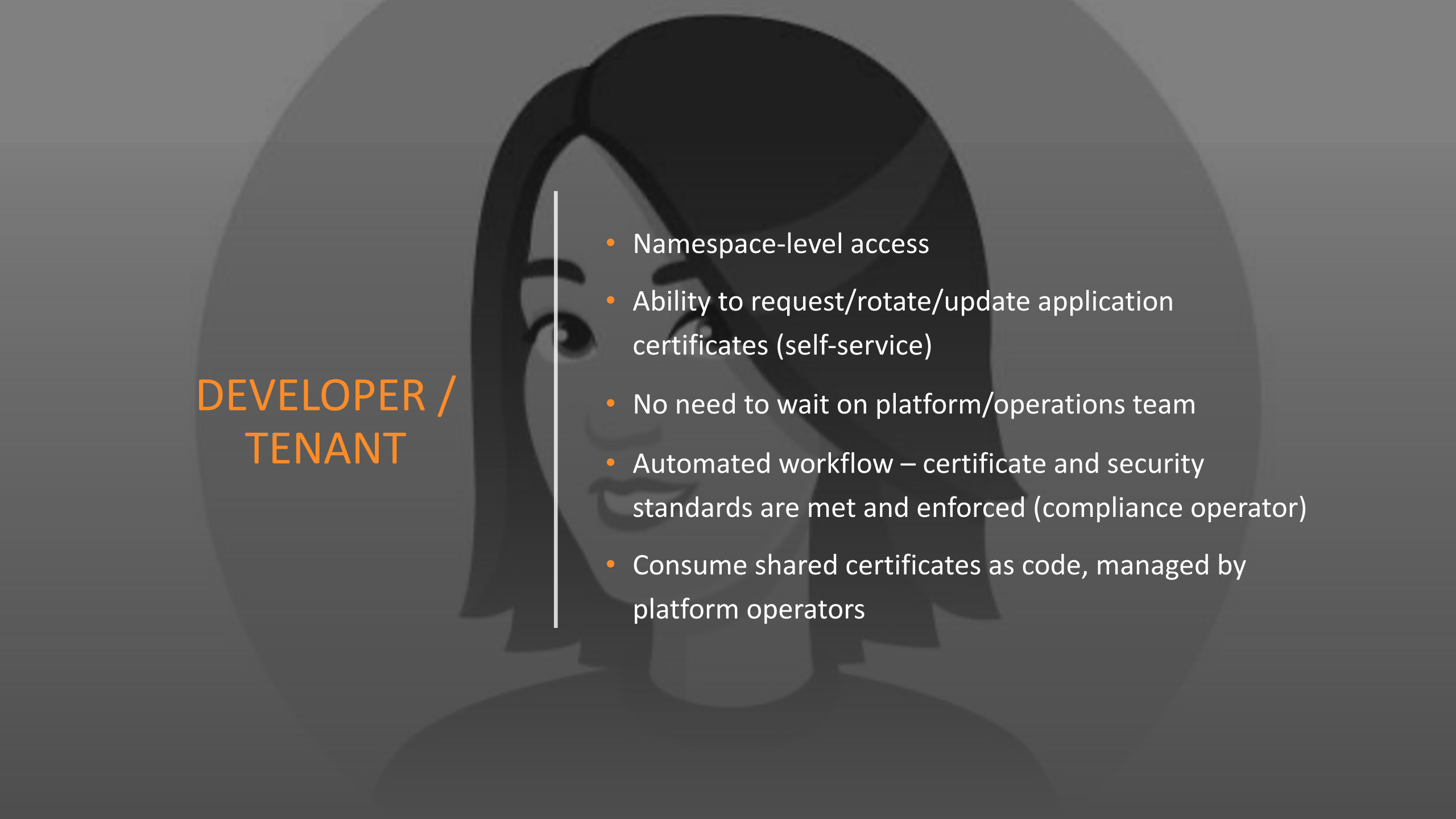
- Certificates as Code
- Website/front-ends
- Dev/Test
- Self-service

OPERATOR

- Enterprise CA chain of trust
- Automation
- Multiple use-case management
- Compliancy & Governance

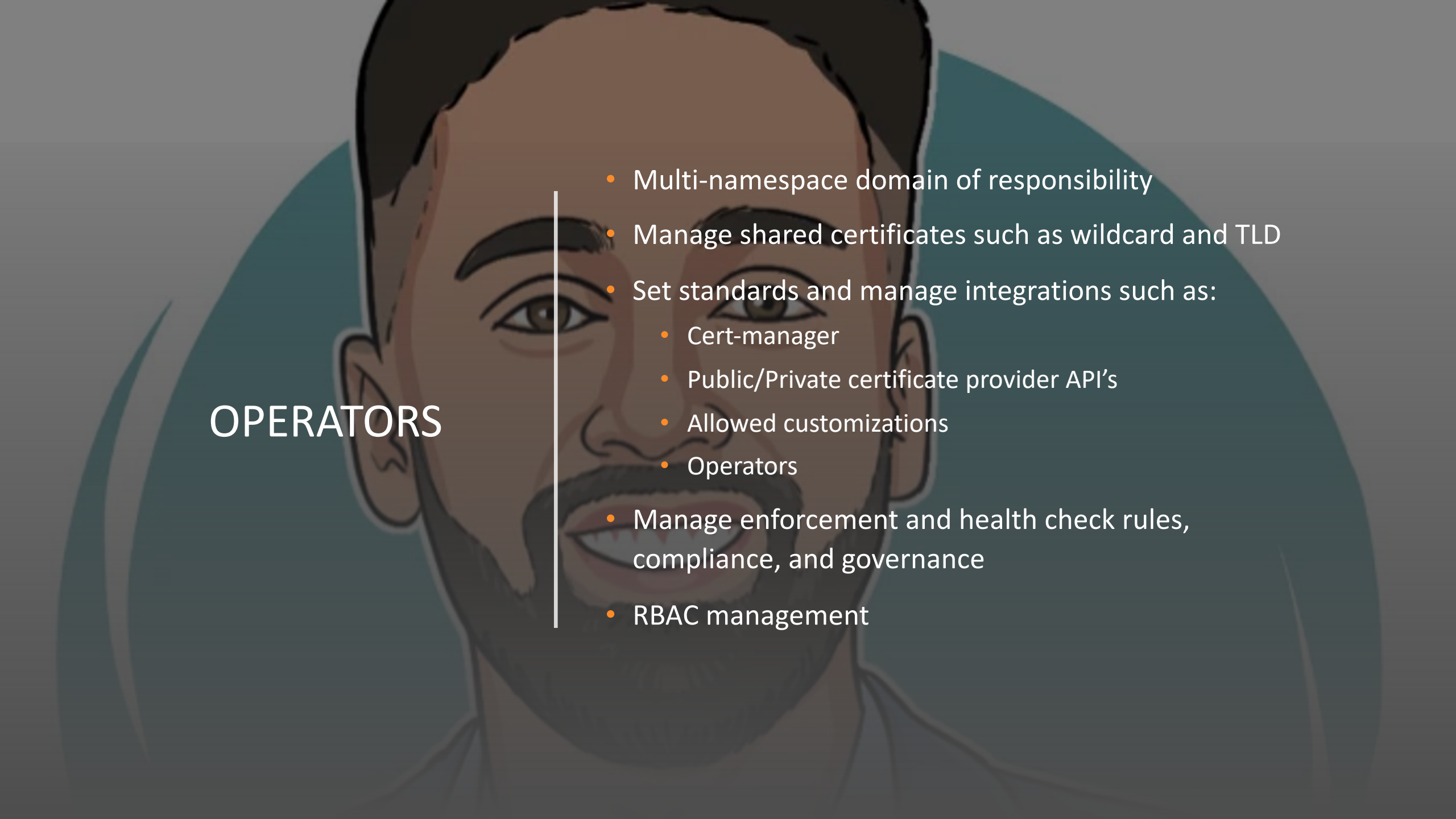
PLATFORM

- Requestor
- API/Integration
- Request/renew certificates
- RBAC



DEVELOPER / TENANT

- Namespace-level access
- Ability to request/rotate/update application certificates (self-service)
- No need to wait on platform/operations team
- Automated workflow – certificate and security standards are met and enforced (compliance operator)
- Consume shared certificates as code, managed by platform operators



OPERATORS

- Multi-namespace domain of responsibility
- Manage shared certificates such as wildcard and TLD
- Set standards and manage integrations such as:
  - Cert-manager
  - Public/Private certificate provider API’s
  - Allowed customizations
  - Operators
- Manage enforcement and health check rules, compliance, and governance
- RBAC management

PLATFORM

- API certificate management
- Ingress/Route certificate management
- Custom domain certificates
- Service Mesh Ingress/Gateway certificates, MTLS certificate rotation
- True PaaS services for consumers

THE MOVE TO DYNAMIC CERTIFICATE MANAGEMENT - HOW

- Ansible / Event-Driven automation runbooks
- HashiCorp Vault/Venafi/Let's Encrypt as CA's/Issuers
- Cert-manager
- Helm for operator lifecycle management
- Certbot
- OpenSSL – validity checking
- GitOps tooling (ArgoCD, Flux, GitHub Actions, GitLab Pipelines)
- Webhooks
- Pipelines
- Scripts
- API
- AI
- Etc, etc.

DYNAMIC CERTIFICATE WORKFLOW

DEV / PROD CERTIFICATE CONSIDERATIONS

- Development Certificate Requirements
  - Short-lived
  - More control
  - Staging environment testing
  - Free

- Production Certificate Requirements
  - Longer-lived
  - Cross-site
  - Health/validity checks/service uptime
  - Auto-rotation
  - Self-serve
  - SaaS/Paid certificate provider
  - Error control
  - Fully automated lifecycle

RBAC & NAMESPACE-SCOPED CONTROLS

namespace-secret-mount-only-role

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: app_namespace
  name: secret-mount-only-role
rules:
- apiGroups: [""]
  resources: ["secrets"]
  verbs: ["list", "watch"]
```

the-issuer.yaml

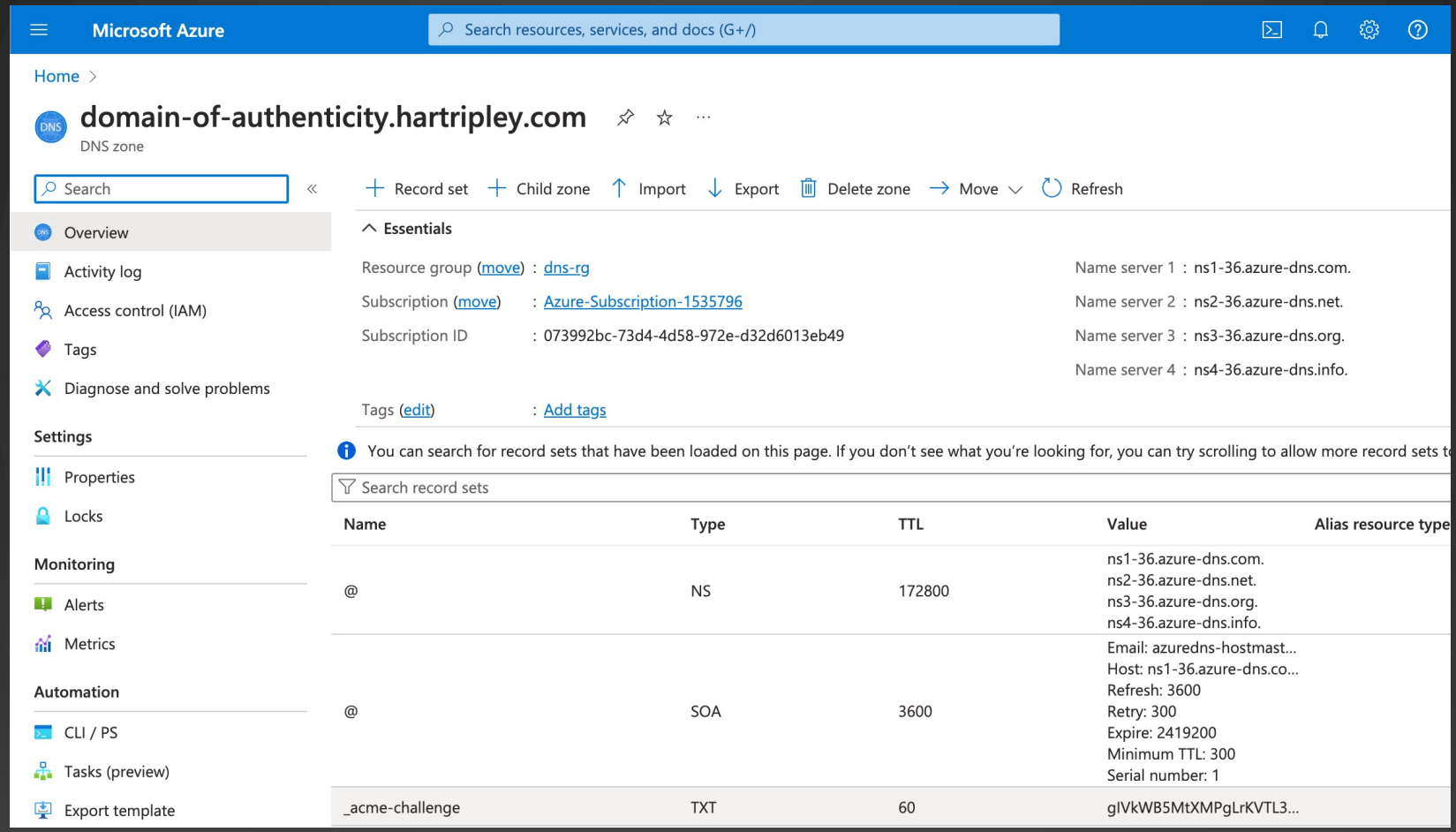
```yaml
apiVersion: cert-manager.io/v1
kind: Issuer
metadata:
  name: the-issuer
  namespace: cert-manager
spec:
  acme:
    server: https://acme-staging-v02.api.letsencrypt.org/directory
    privateKeySecretRef:
      name: issuer-letsencrypt-staging
    solvers:
    - dns01:
        azureDNS:
          clientID: $AZURE_CERT_MANAGER_SP_APP_ID
          clientSecretSecretRef:
            name: azuredns-config
            key: azuredns-sp-secret
          subscriptionID: $AZURE_SUBSCRIPTION_ID
          tenantID: $AZURE_TENANT_ID
          resourceGroupName: $AZURE_DNS_ZONE_RESOURCE_GROUP
          hostedZoneName: $AZURE_DNS_ZONE
          environment: AzurePublicCloud
```

CERTIFICATE REQUEST AS CODE

certificate-as-code.yaml

```yaml
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: le-wc-staging
  namespace: cert-manager
spec:
  secretName: le-wc-staging
  issuerRef:
    name: the-cluster-issuer
    kind: ClusterIssuer
  commonName: '*.domain-of-authenticity.hartripley.com'
  dnsNames:
  - domain-of-authenticity.hartripley.com
  - '*.domain-of-authenticity.hartripley.com'
  acme:
    config:
    - dns01:
        provider: azuredns
      domains:
      - domain-of-authenticity.hartripley.com
      - '*.domain-of-authenticity.hartripley.com'
```

DNS ZONE TXT


Microsoft Azure
Search resources, services, and docs (G+/)
Home
domain-of-authenticity.hartripley.com
DNS zone
Search
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Settings
Properties
Locks
Monitoring
Alerts
Metrics
Automation
CLI / PS
Tasks (preview)
Export template
Record set
Child zone
Import
Export
Delete zone
Move
Refresh
Essentials
Resource group (move) : dns-rg
Subscription (move) : Azure-Subscription-1535796
Subscription ID : 073992bc-73d4-4d58-972e-d32d6013eb49
Tags (edit) : Add tags
Name server 1 : ns1-36.azure-dns.com.
Name server 2 : ns2-36.azure-dns.net.
Name server 3 : ns3-36.azure-dns.org.
Name server 4 : ns4-36.azure-dns.info.
You can search for record sets that have been loaded on this page. If you don't see what you're looking for, you can try scrolling to allow more record sets t
Search record sets
Name
Type
TTL
Value
Alias resource type
@
NS
172800
ns1-36.azure-dns.com.
ns2-36.azure-dns.net.
ns3-36.azure-dns.org.
ns4-36.azure-dns.info.
@
SOA
3600
Email: azuredns-hostmast...
Host: ns1-36.azure-dns.co...
Refresh: 3600
Retry: 300
Expire: 2419200
Minimum TTL: 300
Serial number: 1
_acme-challenge
TXT
60
gIVkWB5MtXMPgLrKVTL3...

SECRET MIRRORING ACROSS NAMESPACES

wildcard-certificate-secret-template.yaml

```yaml
apiVersion: v1
data:
  ca.crt: ''
  tls.crt: ''
  tls.key: ''
kind: Secret
metadata:
  name: le-wc-staging
  namespace: cert-manager
  annotations:
    kubed.appscode.com/sync: "cert=lets-encrypt-wc" # Label value to sync
certificates to labelled namespaces/projects
type: kubernetes.io/tls
```

runwhen-local-ns.yaml

```yaml
apiVersion: v1
kind: Namespace
metadata:
  labels:
    cert: lets-encrypt-wc
  name: runwhen-local
```

```
kubectl get secrets -n runwhen-local

NAME                                  TYPE                                  DATA   AGE
le-wc-staging                         kubernetes.io/tls                     2      3h47m
```

INGRESS CERTIFICATE LIFECYCLE MANAGEMENT

ingress-annotation.yaml

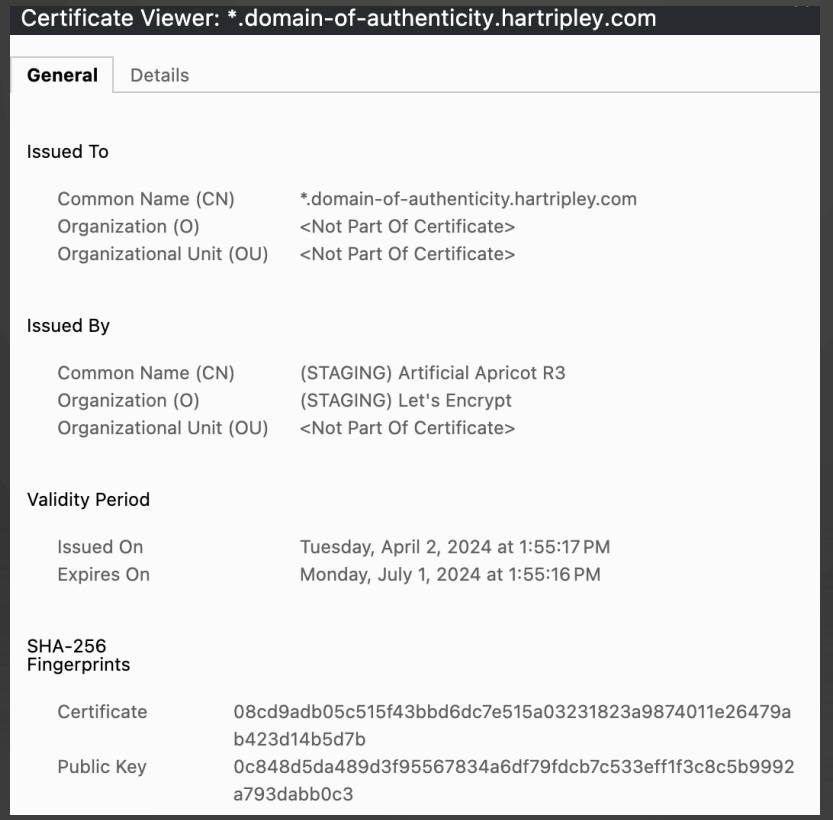
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: nginx
  annotations:
    kubernetes.io/ingress.class: nginx
    cert-manager.io/cluster-issuer: the-cluster-issuer
spec:
  rules:
  - host: app.domain-of-authenticity.hartripley.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: app-service
            port:
              number: 80
  tls:
  - hosts:
    - app.domain-of-authenticity.hartripley.com
```

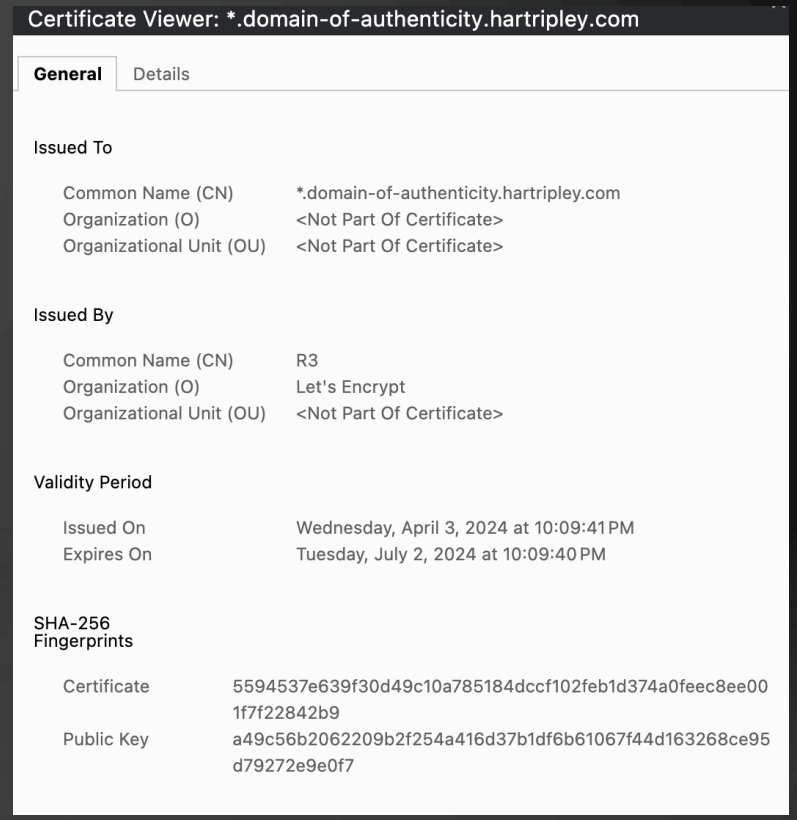
AUTOMATED ROUTE CERTIFICATE INJECTION (OPENSHIFT)

route-annotation-certs.yaml

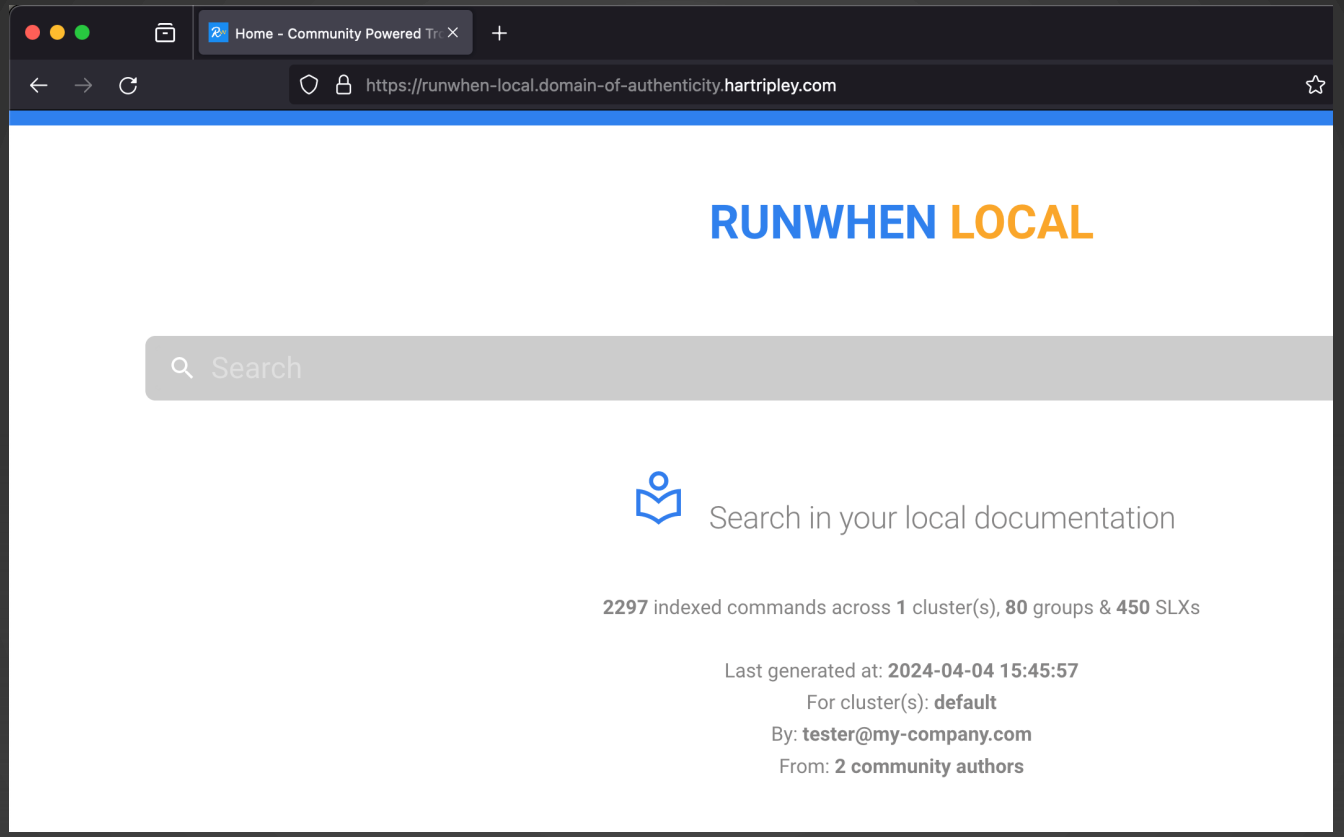
```yaml
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  labels:
    app.kubernetes.io/instance: runwhen-local
    app.kubernetes.io/managed-by: Helm
    app.kubernetes.io/name: runwhen-local
    app.kubernetes.io/version: 0.3.10
    helm.sh/chart: runwhen-local-0.0.24
    annotations:
      cert-utils-operator.redhat-cop.io/certs-from-secret: le-wc-prod
  name: runwhen-local-prod
  namespace: runwhen-local
spec:
  host: runwhen-local.domain-of-authenticity.hartripley.com
  port:
    targetPort: mkdocs
  tls:
    termination: edge
  to:
    kind: Service
    name: runwhen-local
    weight: 100
  wildcardPolicy: None
```

CUSTOM DOMAIN CERTIFICATES





APPLICATION WITH WILDCARD CERTIFICATE



https://runwhen-local.domain-of-authenticity.hartripley.com/

IN PRACTICE

- Kubernetes platform (xKS, OpenShift, etc.)
- Source Code Management Automation/orchestration (cert-manager/Ansible/GitOps)
- ClusterRole/Role/SA’s
- Ingress/Route’s use of certificates
- Working application
- Certificates across namespaces (*)
- Auto rotation
- Updating certificate requirements (as Code)

GETTING STARTED / NEXT STEPS

- Kubernetes Config Syncer
- cert-manager
- Cert Utils Operator
- Code Ready Containers (CRC)
  - Direct mirror
- Backstage / RH Dev Hub
- RunWhen

OTHER USE-CASES

- X509 checks
- Ansible to build certificate and validate chains
- Ansible to update root or intermediate/issuing certificates within chain
- Self-serve front-end
- Part of onboarding/namespace creation
- RBAC/custom certificate roles (least-privileged)
- Service Mesh/gateway certificates
- Pipeline jobs for certificate validation

LET’S CONNECT

https://github.com/ripleyhart