Safeguard Istio* Service Mesh via Confidential Computing

Iris Ding/irisdingbj/ cloud software engineer
Addepalli, Srinivasa R/saddepalli/ Senior Principal Engineer

#IstioCon

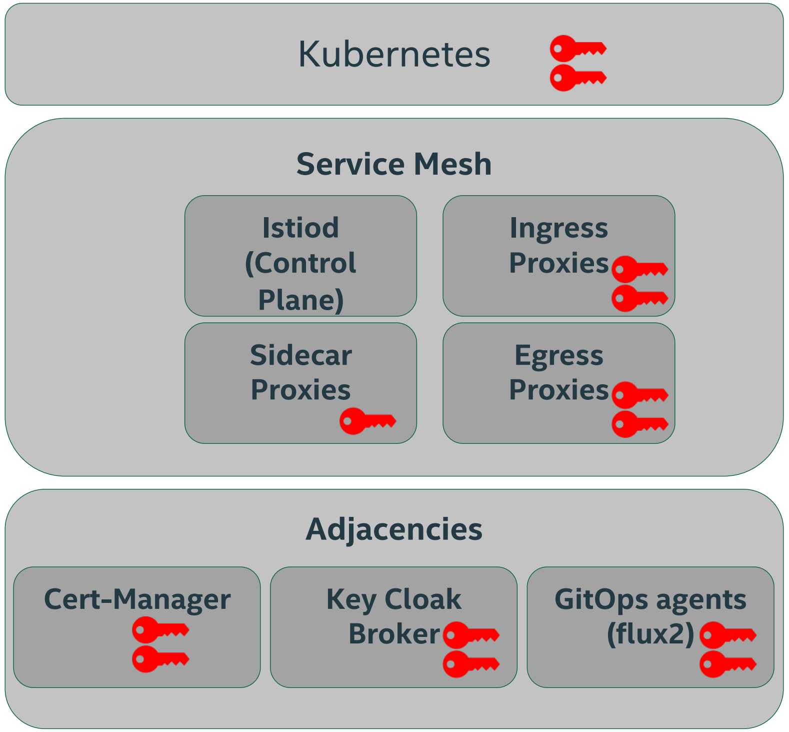
Agenda

- Background
  - Private keys in service mesh and its adjacencies
  - Possible attack surfaces
  - Current private key security methods and need for distributed HSM
- Deep dive
  - Confidential computing – support from processors
  - Solutions
    - Distributed HSM for CA Keys
    - Distributed HSM for mTLS keys
    - Distributed HSM for application keys

Service Mesh – Intel initiatives themes

- Simplify
  - Comprehensive, ease of deployment in multi-provider, multi-location, and multi-tenant environments
- Accelerate
  - Reduce resource usage and reduce tail latency with Intel® architecture accelerators
- Secure
  - Private key secure signing and confidential computing of critical components
- Optimize
  - Software optimizations to reduce resource usage
- Multi-tenant ready
  - Performance and security isolation among tenant workloads
- Extend
  - Address new applications and network services

Service Mesh (ISTIO*/Envoy*) and private keys



Many private keys in Kubernetes* with Service Mesh and its adjacencies (Used for authentication, certificate enrollments & signing)

- Service accounts keys
- Server applications' keys
- Client applications' keys
- Service-to-Service mTLS keys
- Certificate authority keys
- JWT signing keys
- Git token protection keys

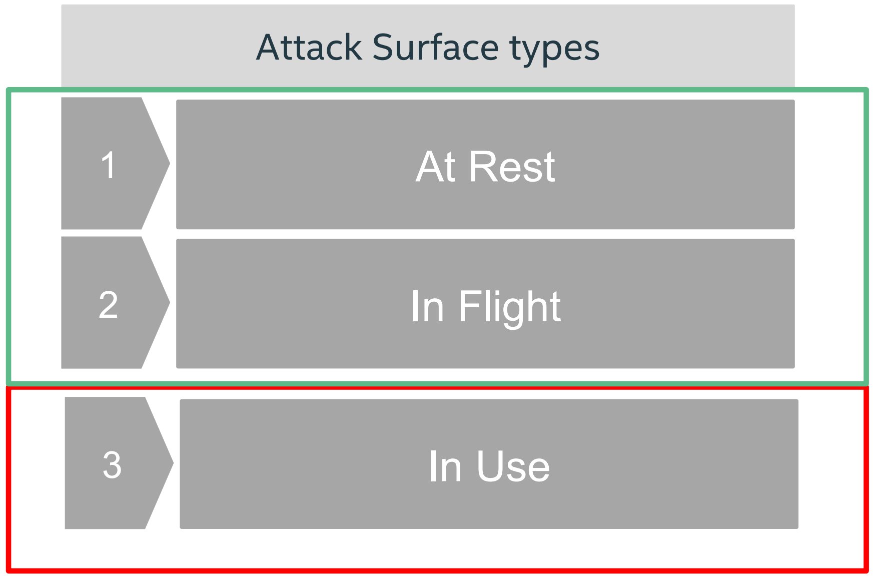
Many of the keys are not ephemeral

Any compromise of the keys can result in

- Impersonation
- Certificate issuance to bad actors

#IstioCon

Private key stealing – attack surface types



Decent protection today
- Via K8s secrets
- K8 secrets encrypted (using Vault) or Selective encryption of keys using Mozilla sops

This attack surface is not yet plugged comprehensively
Private keys are in clear in memory

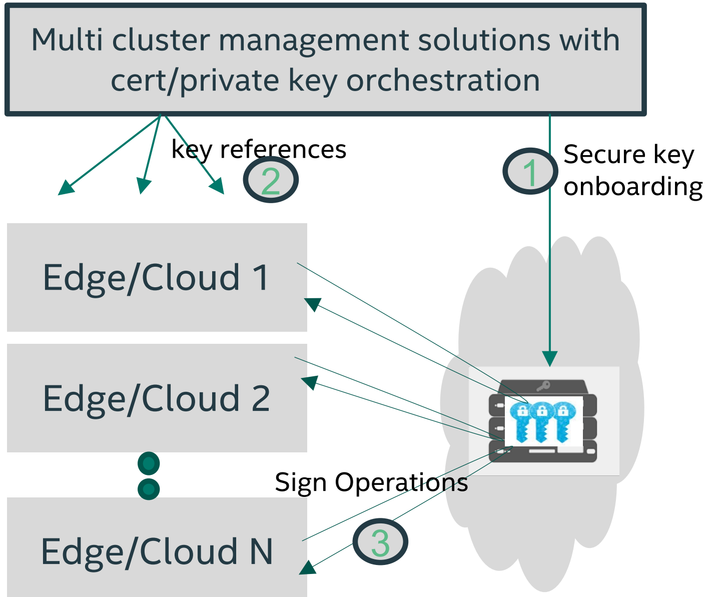
Private key stealing – distributed attack surfaces

Multi cloud and edge computing

1. Multiple infrastructure administrators – higher chance of compromise (malicious admins)
2. Edge locations are not physically secure - higher chance of HW based memory scraping
3. Multiple instances of application instances with varied threat protection – higher chance of successful exploits

HSMs play an important role to address these challenges – HSMs that never expose private keys out of its boundary

Private key security – network-based HSM

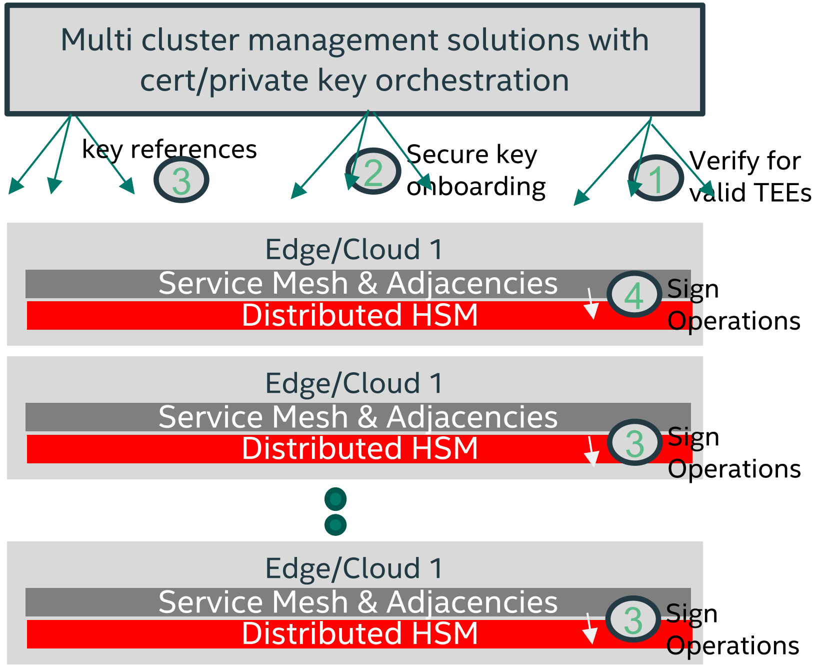


Highly secure
- Private keys are never exposed in clear in Edge/Cloud locations

Challenges:
- High latency of sign operations -> negates the drivers for edge computing
- Lower transaction rate (lower TPS)
- Higher cost if multiple HSMs are used

Need for distributed HSMs to address security needs of distributed computing

Private key security – with distributed hsm

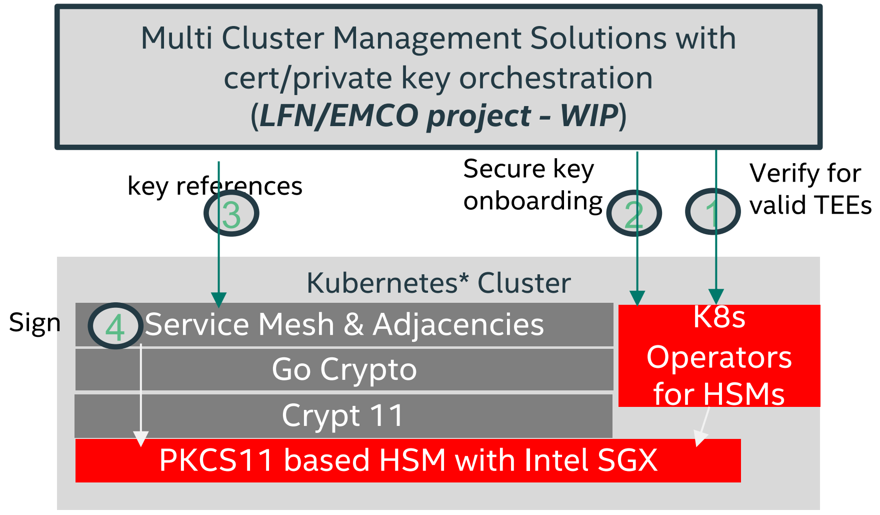


Highly secure
- Private keys are never exposed in clear in Edge/Cloud locations

No more challenges of
- High latency
- Lower transaction rate (Lower TPS)
- Higher cost

How?
- Leverage new processor TEE technologies
- Sign operations are just function calls to trusted enclaves

Cloud native distributed HSM – technologies


Multi Cluster Management Solutions with cert/private key orchestration (*LFN/EMCO project - WIP*)
key references
3
Secure key onboarding
2
1
Verify for valid TEEs
Kubernetes* Cluster
Sign
4
Service Mesh & Adjacencies
Go Crypto
Crypt 11
K8s Operators for HSMs
PKCS11 based HSM with Intel SGX

Centralized key management (LFN/EMCO)
- For large number of K8s clusters
- For large number of applications

To verify for genuine enclave and PKCS11 SW

To onboard keys
- Tenant specific CA Cert/Private-key
- Ingress/Egress application keys

Intel® Software Guard Extensions (Intel® SGX):
- For secure enclaves

PKCS11 software for Intel SGX enclaves

Integration with GoCrypto via Crypto11

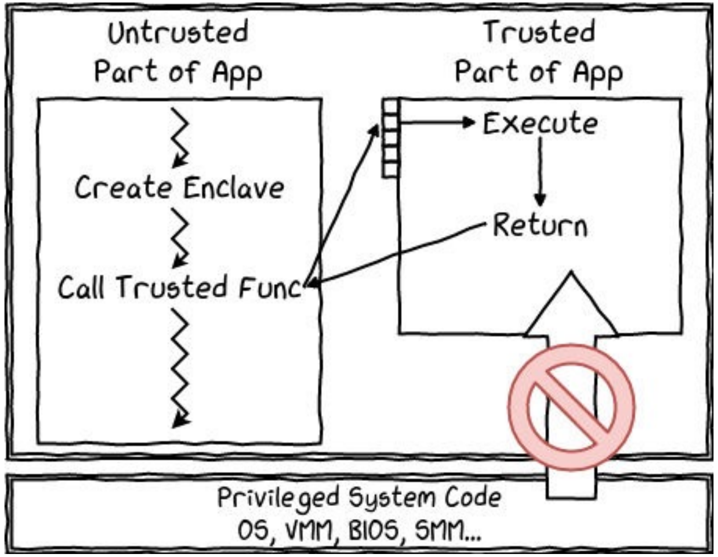
Operator to get Intel SGX enclaves quotes and onboard keys

Deep dive : Cert-Manager CA key security, mTLS private key security, applications' private key security

#IstioCon

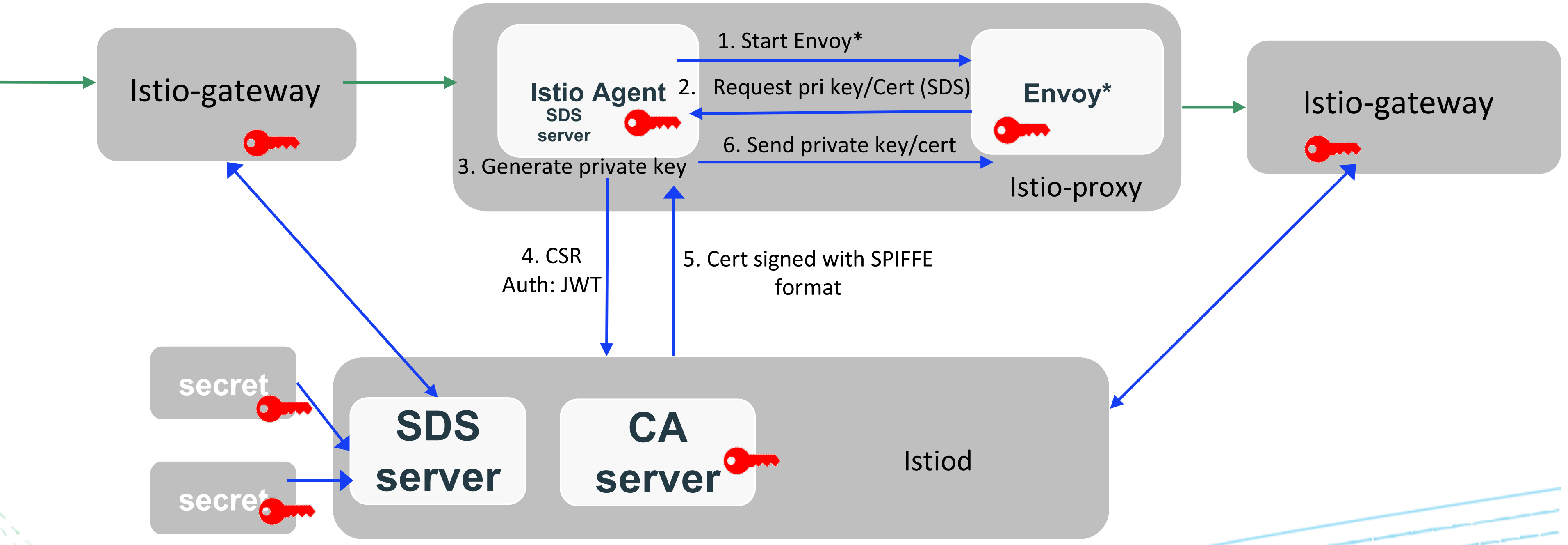
*Other names and brands may be claimed as the property of others.

Enclaves Based on Intel® Software Guard Extensions



- ❑ **Memory Encryption**
- ❑ **Access Control**
- ❑ **Remote Attestation**
- ❑ **Sealing**

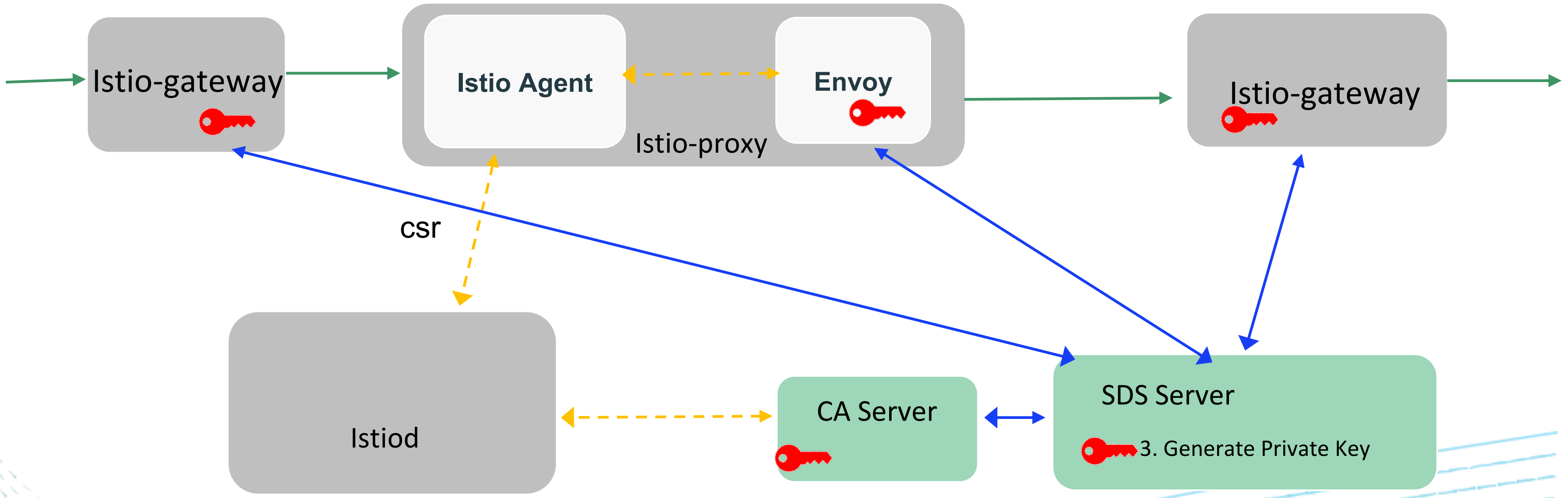
Security flow



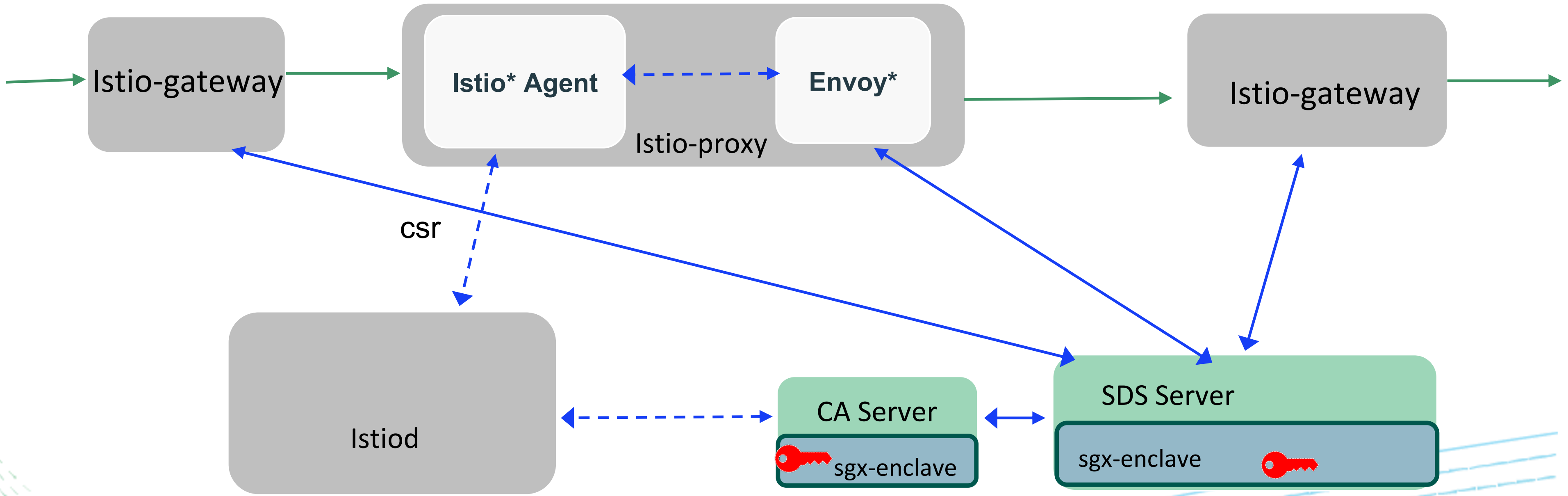
#IstioCon

*Other names and brands may be claimed as the property of others.

Extensible architecture for Istio*

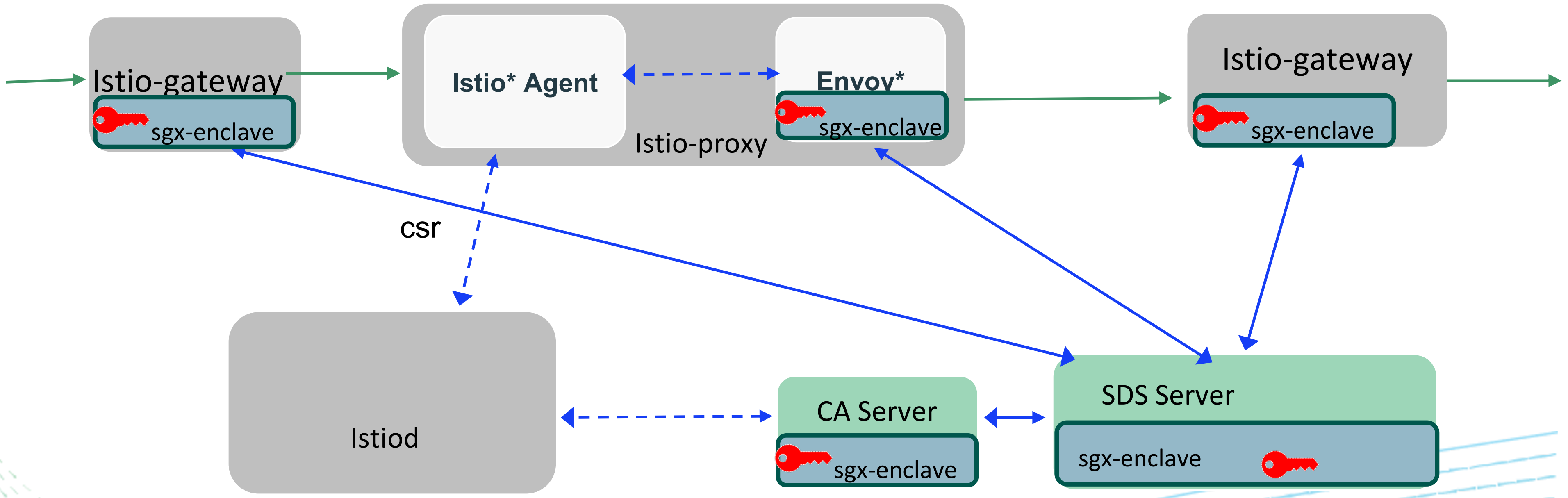


Leverage HW TEE as HSM

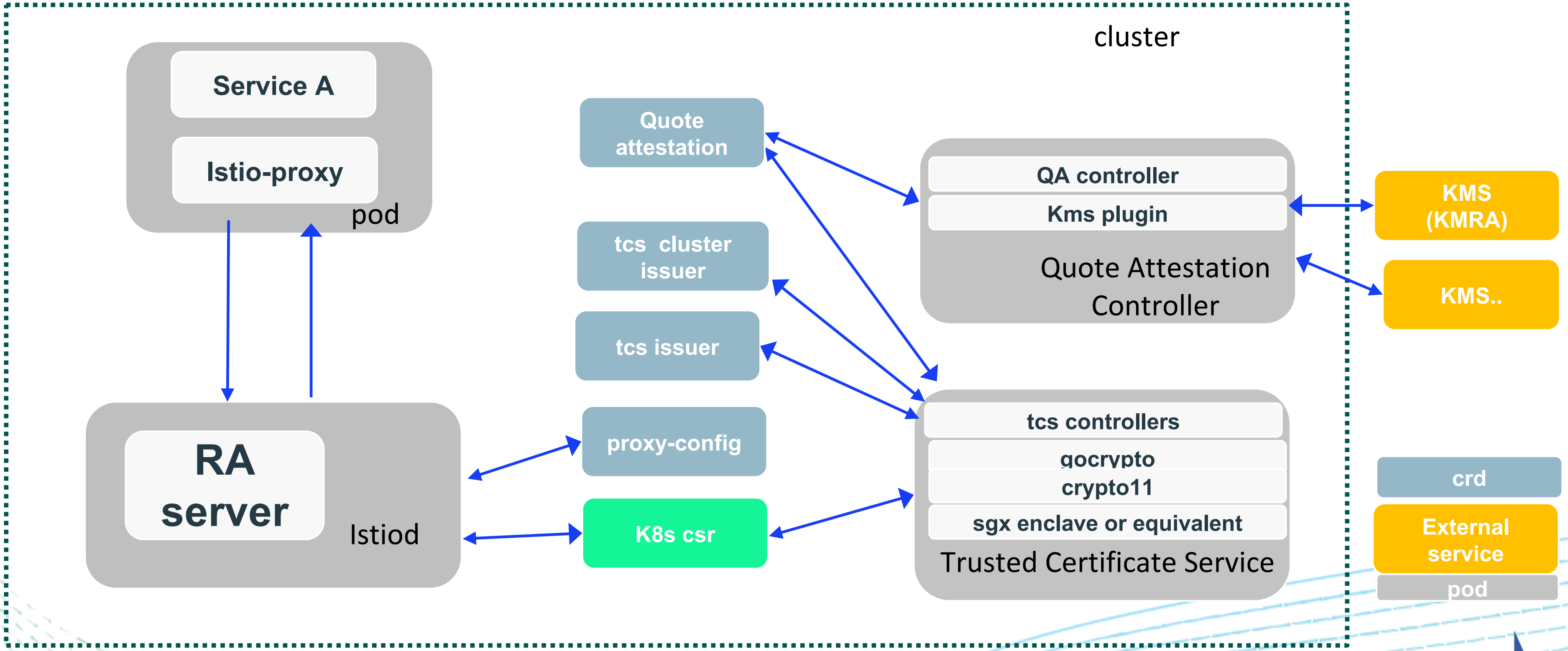


*Other names and brands may be claimed as the property of others.

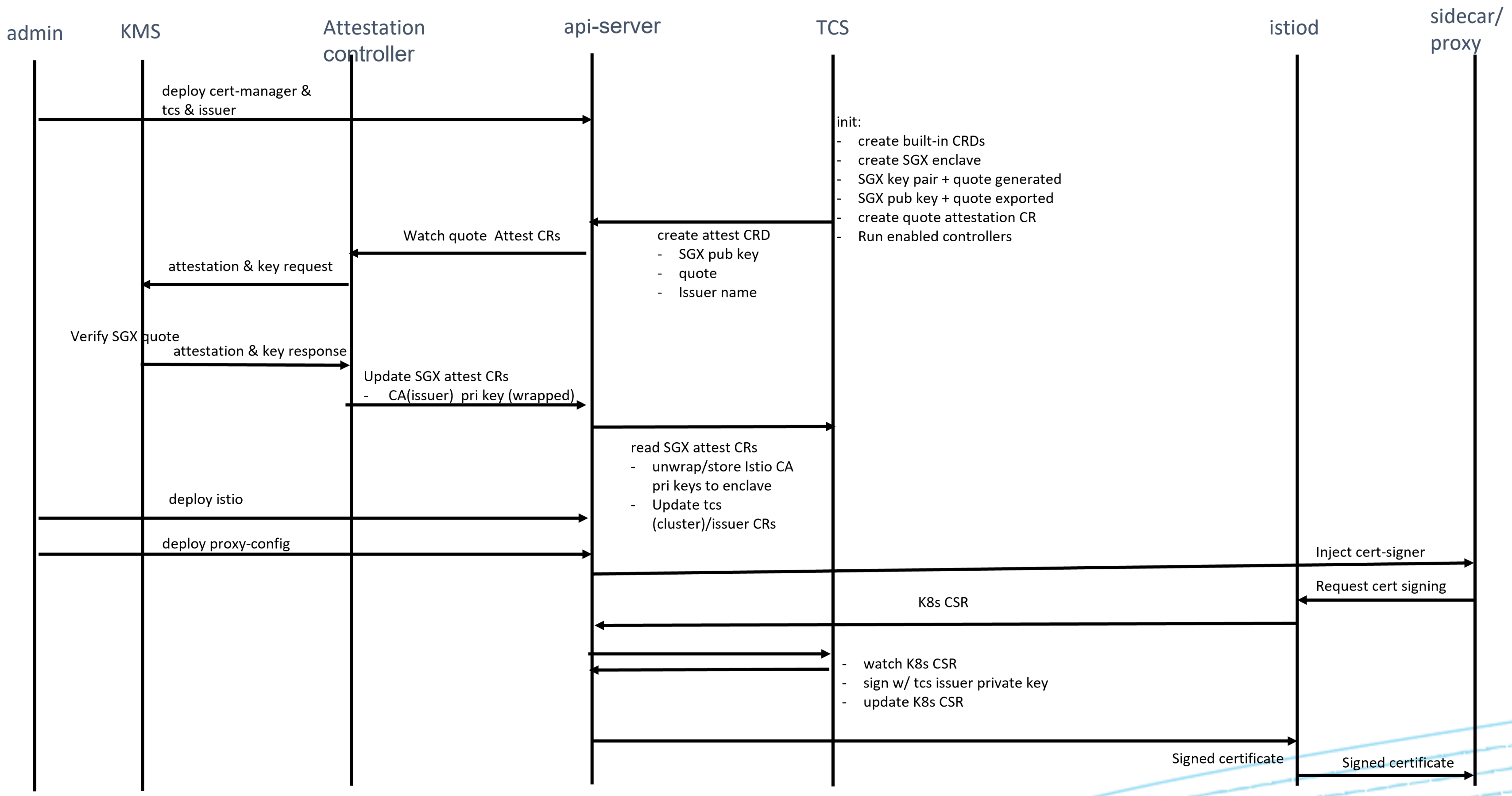
Leverage HW TEE as HSM



Distributed HSM for CA keys



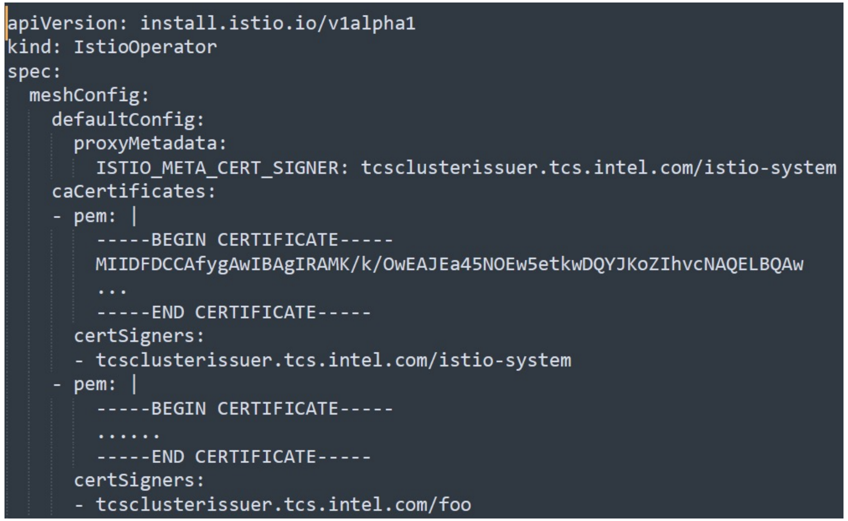
#IstioCon



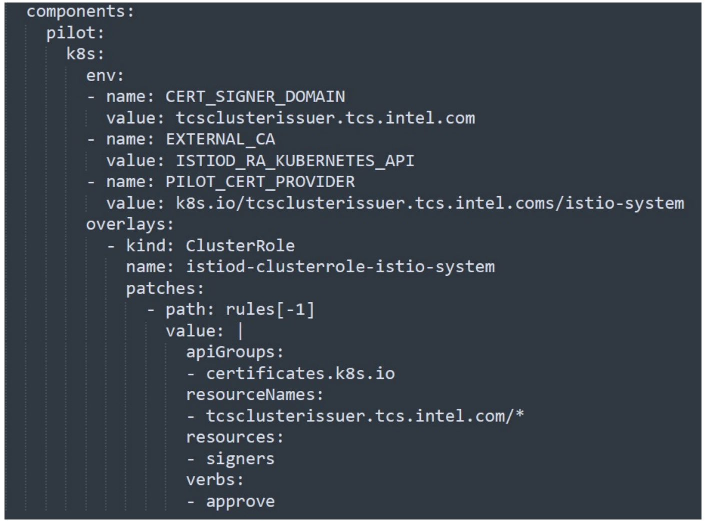
#IstioCon

SGX refers to Intel® Software Guard Extensions. *Other names and brands may be claimed as the property of others.

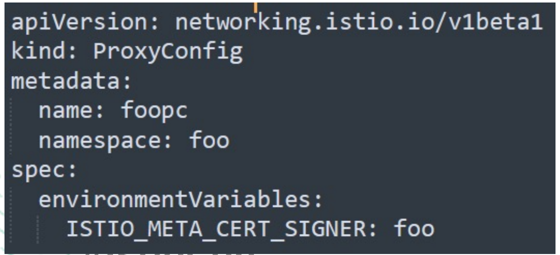
Distributed HSM for CA keys



```yaml
apiVersion: install.istio.io/v1alpha1
kind: IstioOperator
spec:
  meshConfig:
    defaultConfig:
      proxyMetadata:
        ISTIO_META_CERT_SIGNER: tcsclusterissuer.tcs.intel.com/istio-system
    caCertificates:
    - pem: |
        -----BEGIN CERTIFICATE-----
        MIIDFDCCAfygAwIBAgIRAMK/k/OwEAJEa45NOEw5etkwDQYJKoZIhvcNAQELBQAw
        ...
        -----END CERTIFICATE-----
      certSigners:
      - tcsclusterissuer.tcs.intel.com/istio-system
    - pem: |
        -----BEGIN CERTIFICATE-----
        ......
        -----END CERTIFICATE-----
      certSigners:
      - tcsclusterissuer.tcs.intel.com/foo
```



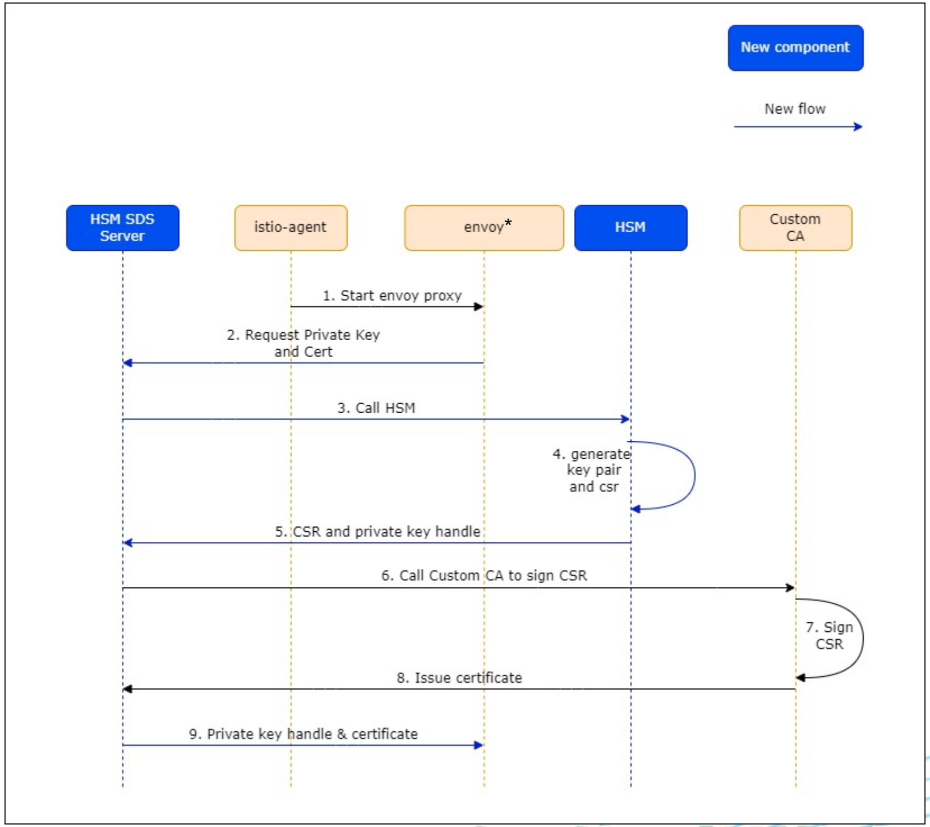
```yaml
components:
  pilot:
    k8s:
      env:
      - name: CERT_SIGNER_DOMAIN
        value: tcsclusterissuer.tcs.intel.com
      - name: EXTERNAL_CA
        value: ISTIOD_RA_KUBERNETES_API
      - name: PILOT_CERT_PROVIDER
        value: k8s.io/tcsclusterissuer.tcs.intel.coms/istio-system
      overlays:
        - kind: ClusterRole
          name: istiod-clusterrole-istio-system
          patches:
            - path: rules[-1]
              value: |
                apiGroups:
                - certificates.k8s.io
                resourceNames:
                - tcsclusterissuer.tcs.intel.com/*
                resources:
                - signers
                verbs:
                - approve
```



```yaml
apiVersion: networking.istio.io/v1beta1
kind: ProxyConfig
metadata:
  name: foopc
  namespace: foo
spec:
  environmentVariables:
    ISTIO_META_CERT_SIGNER: foo
```

https://github.com/intel/trusted-certificate-issuer

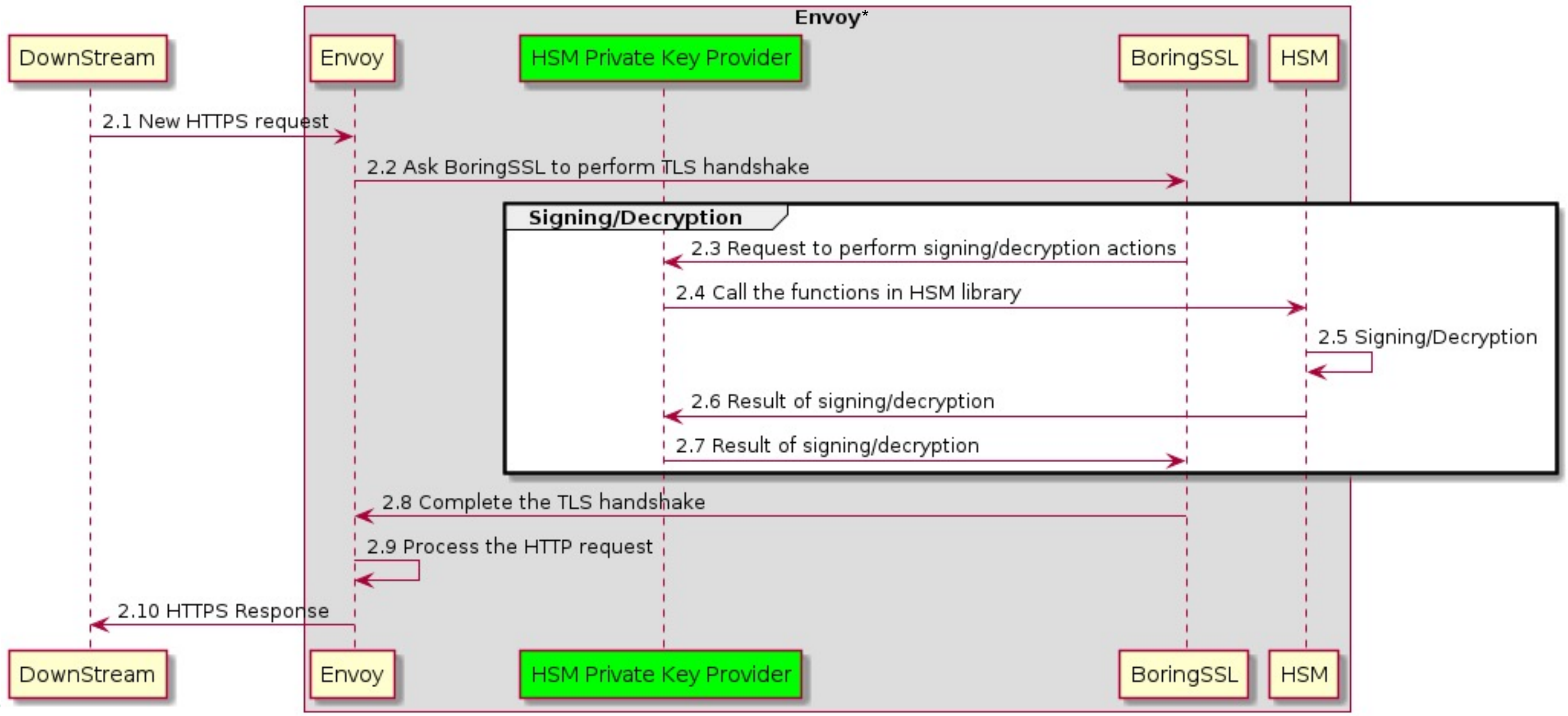
Distributed HSM for mTLS keys


New component
New flow
HSM SDS Server
istio-agent
envoy*
HSM
Custom CA
1. Start envoy proxy
2. Request Private Key and Cert
3. Call HSM
4. generate key pair and csr
5. CSR and private key handle
6. Call Custom CA to sign CSR
7. Sign CSR
8. Issue certificate
9. Private key handle & certificate

#IstioCon

*Other names and brands may be claimed as the property of others.

Distributed HSM for mTLS keys


Envoy*
DownStream
Envoy
HSM Private Key Provider
BoringSSL
HSM
2.1 New HTTPS request
2.2 Ask BoringSSL to perform TLS handshake
Signing/Decryption
2.3 Request to perform signing/decryption actions
2.4 Call the functions in HSM library
2.5 Signing/Decryption
2.6 Result of signing/decryption
2.7 Result of signing/decryption
2.8 Complete the TLS handshake
2.9 Process the HTTP request
2.10 HTTPS Response
DownStream
Envoy
HSM Private Key Provider
BoringSSL
HSM

#IstioCon

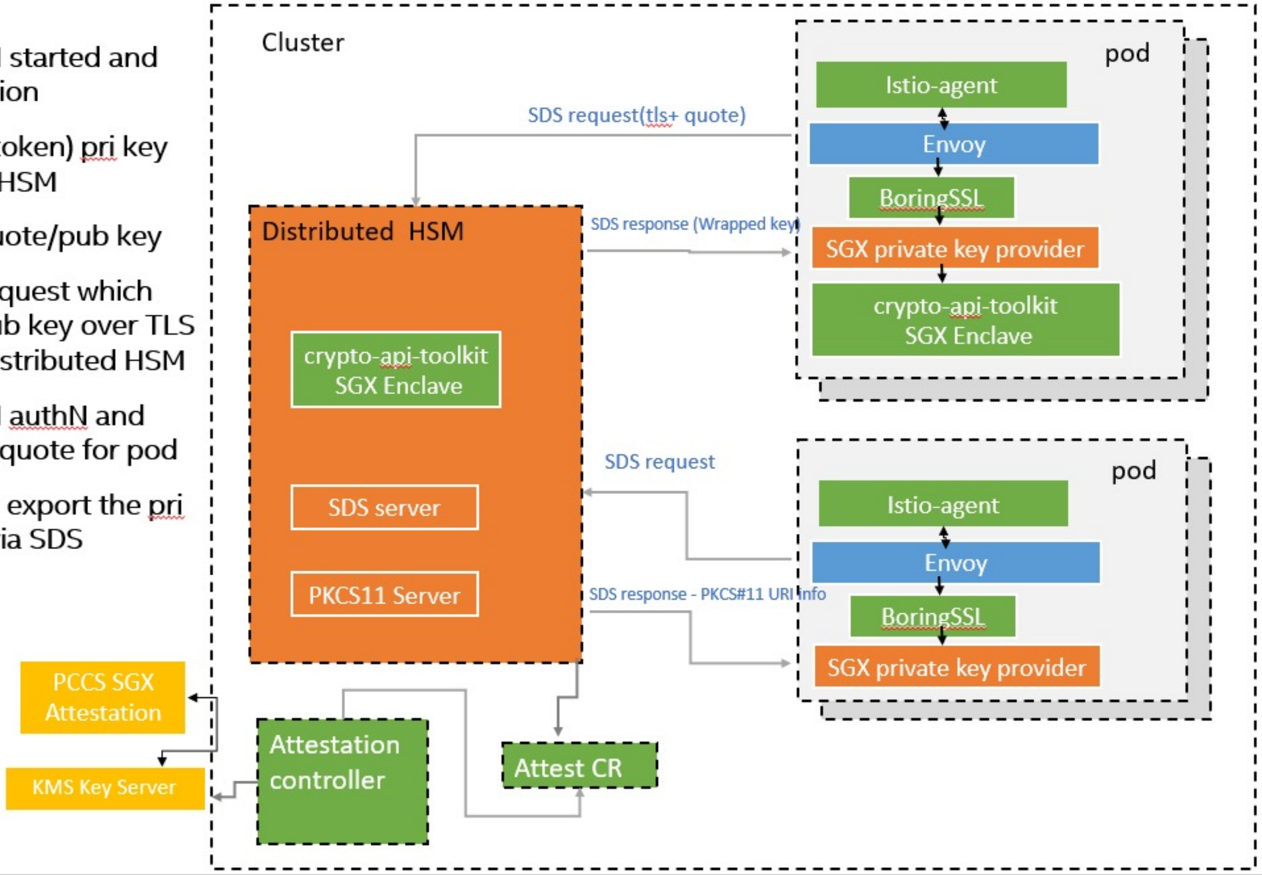
*Other names and brands may be claimed as the property of others.

Distributed HSM for application keys

- Private keys are in clear text in Kubernetes* secret
- Multi replica of gateway deployment which might be spanned across different nodes and auto-scaled
- Local HSM vs remote HSM

Distributed HSM for application keys

- Distributed HSM started and finished attestation
- Admin import (token) pri key into distributed HSM
- Pod generate quote/pub key
- Pod send sds request which carries quote/pub key over TLS connection to distributed HSM
- Distributed HSM authN and check incoming quote for pod
- distributed HSM export the pri key out to pod via SDS response



SGX refers to Intel® Software Guard Extensions. Other names and brands may be claimed as the property of others.

Current Status

- Distributed HSM for CA keys
  - Open Source project
  - Support multi-CA
  - External issuer for cert-manager eco-system
- Distributed HSM for mTLS keys
  - Design Proposal in Envoy*
- Distributed HSM for application keys (gateway)
  - Design Proposal in Istio*

*Other names and brands may be claimed as the property of others.

Thank you!

@handle
https://www.linkedin.com/in/srinivasa-addepalli-0589b51/
https://www.linkedin.com/in/iris-shao-jun-ding-7b87ba20/

Notices and Disclaimers