KubeCon | CloudNativeCon

North America 2023

KubeCon | CloudNativeCon
North America 2023

cert-manager in 5 Levels of Difficulty

Tim Ramlot
Maintainer
Venafi

Ashley Davis
Maintainer
Venafi

Maël Valais
Maintainer
Venafi

Intro / cert-manager

X.509 certificate management for Kubernetes and OpenShift

11K+
GitHub Stars

380+
Contributors

1 million+
daily downloads

CNCF Incubating Project

Intro / 5 Levels of Difficulty

Level 1: Ingress and Gateway Annotations

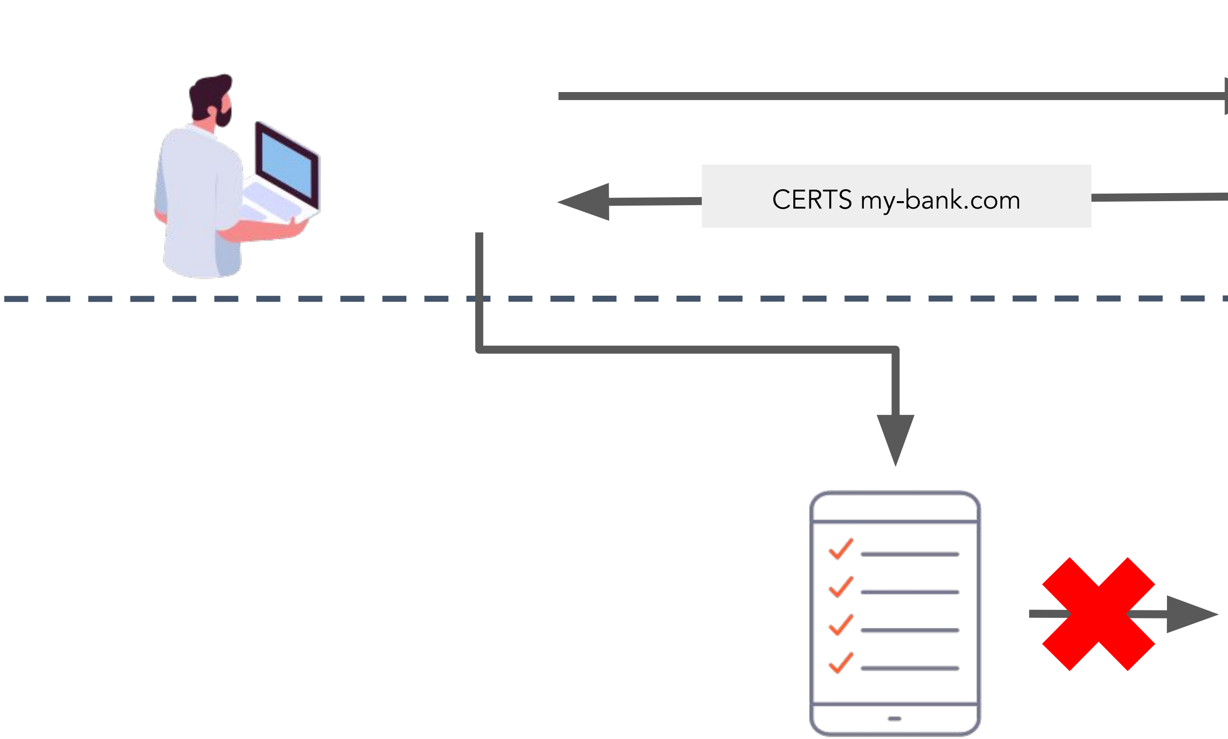
Level 2: Using the Certificate Resource

Level 3: Private PKI & trust-manager

Level 4: CSI-drivers and approver-policy

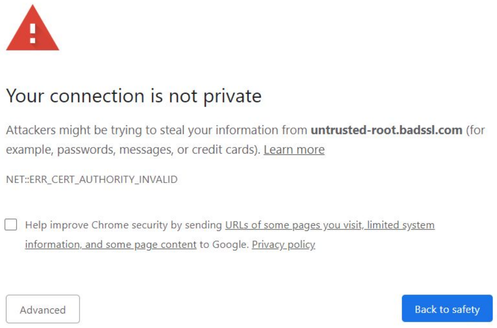
Level 5: Develop Custom Issuers and Plugins

Intro / Your Connection is Not Private



my-bank.com:443

CERTS my-bank.com



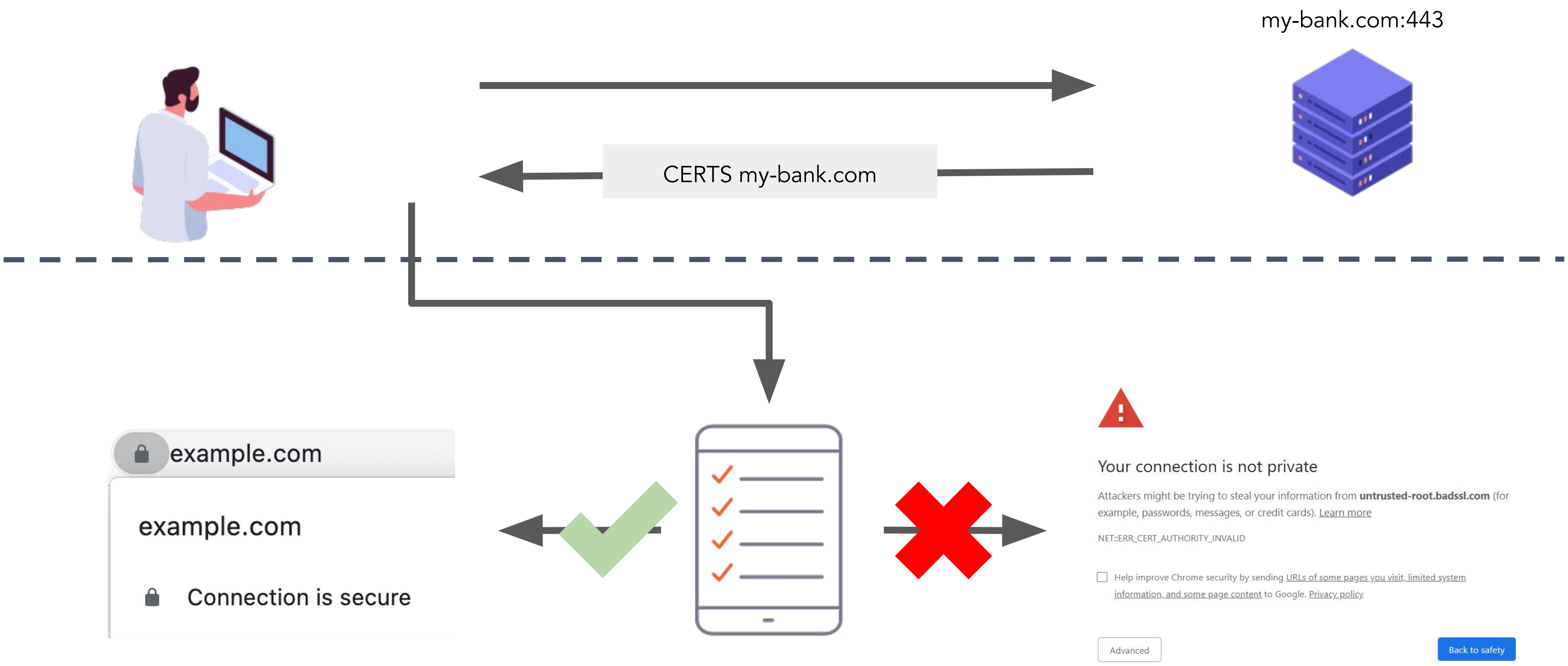
Your connection is not private

Attackers might be trying to steal your information from **untrusted-root.badssl.com** (for example, passwords, messages, or credit cards). Learn more

NET::ERR_CERT_AUTHORITY_INVALID

☐ Help improve Chrome security by sending URLs of some pages you visit, limited system information, and some page content to Google. Privacy policy

Intro / Your Connection is Private



example.com

example.com

Connection is secure

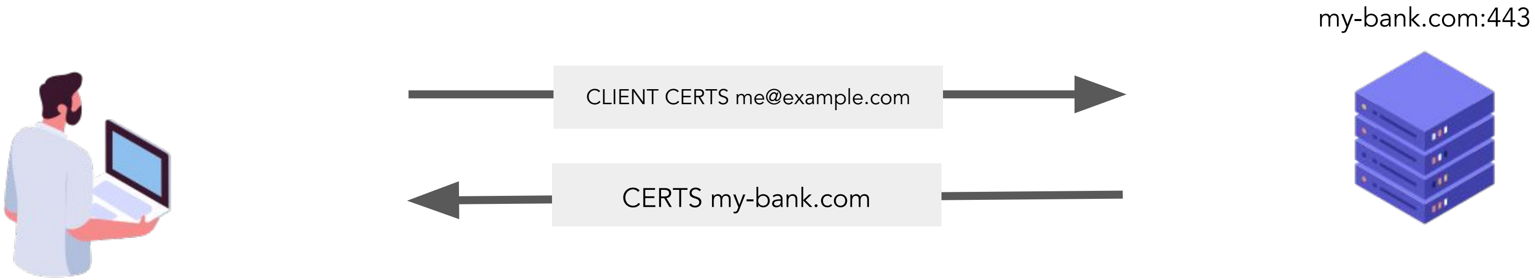
Your connection is not private

Attackers might be trying to steal your information from **untrusted-root.badssl.com** (for example, passwords, messages, or credit cards). Learn more

NET::ERR_CERT_AUTHORITY_INVALID

□ Help improve Chrome security by sending URLs of some pages you visit, limited system information, and some page content to Google. Privacy policy

Intro / Your Authentication is Mutual

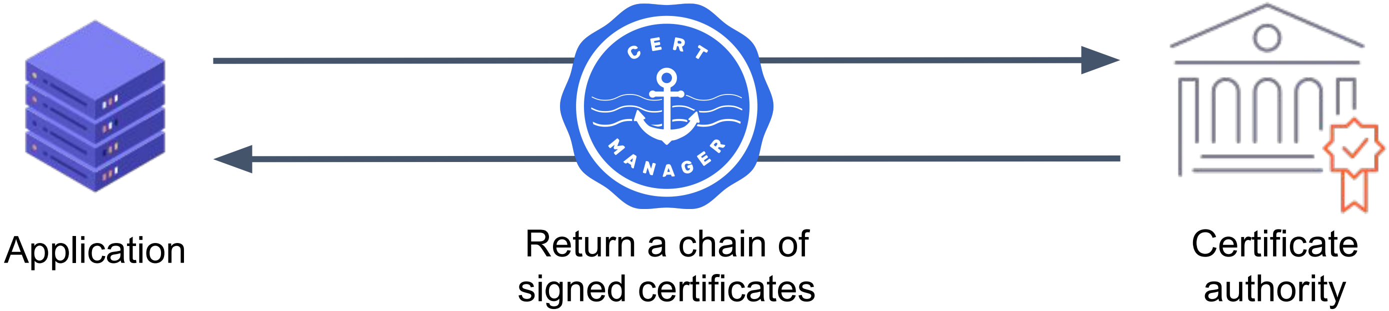


my-bank.com:443

CLIENT CERTS me@example.com

CERTS my-bank.com

Level 1 / Issuing a Certificate using cert-manager

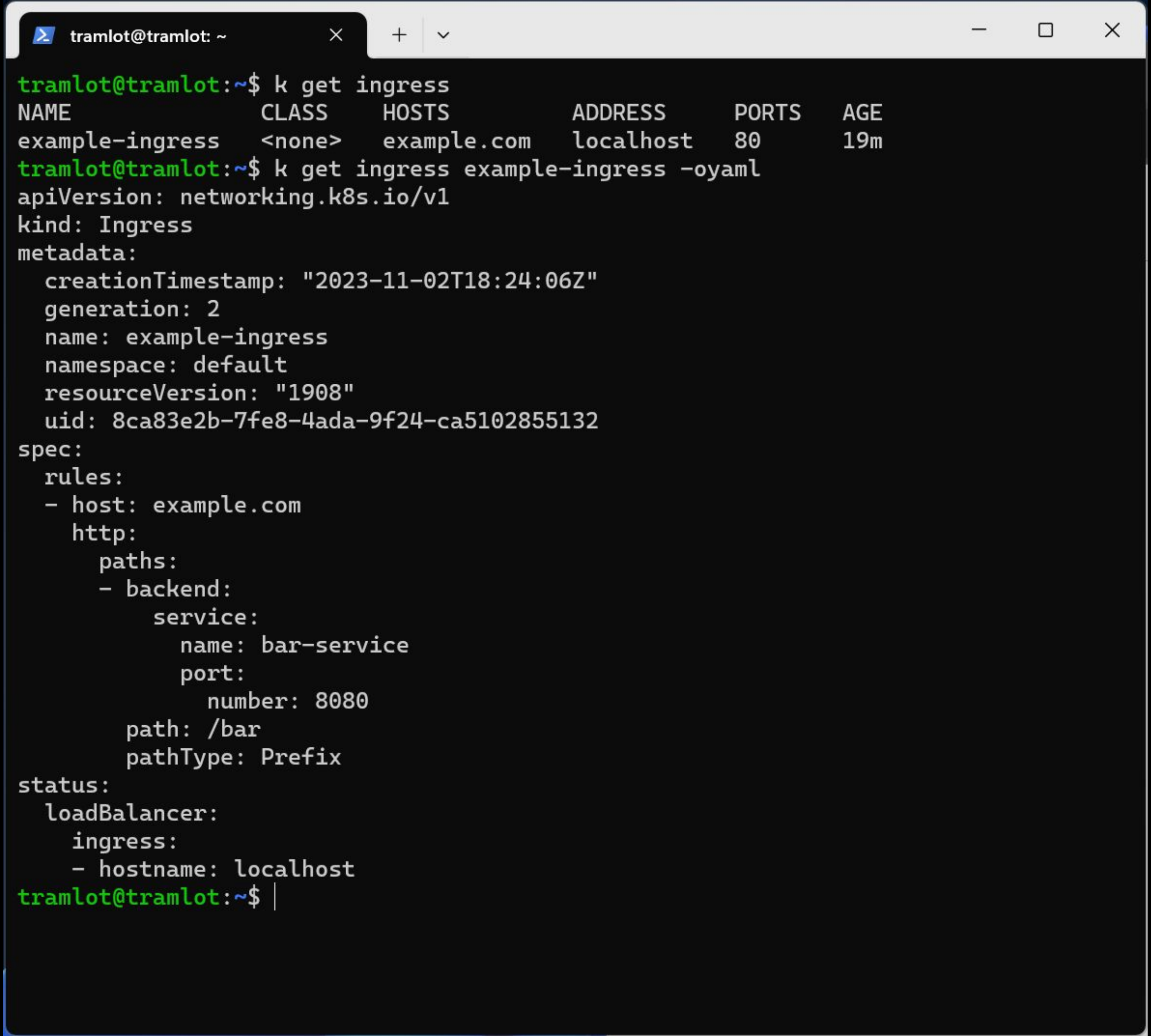


Request a Certificate

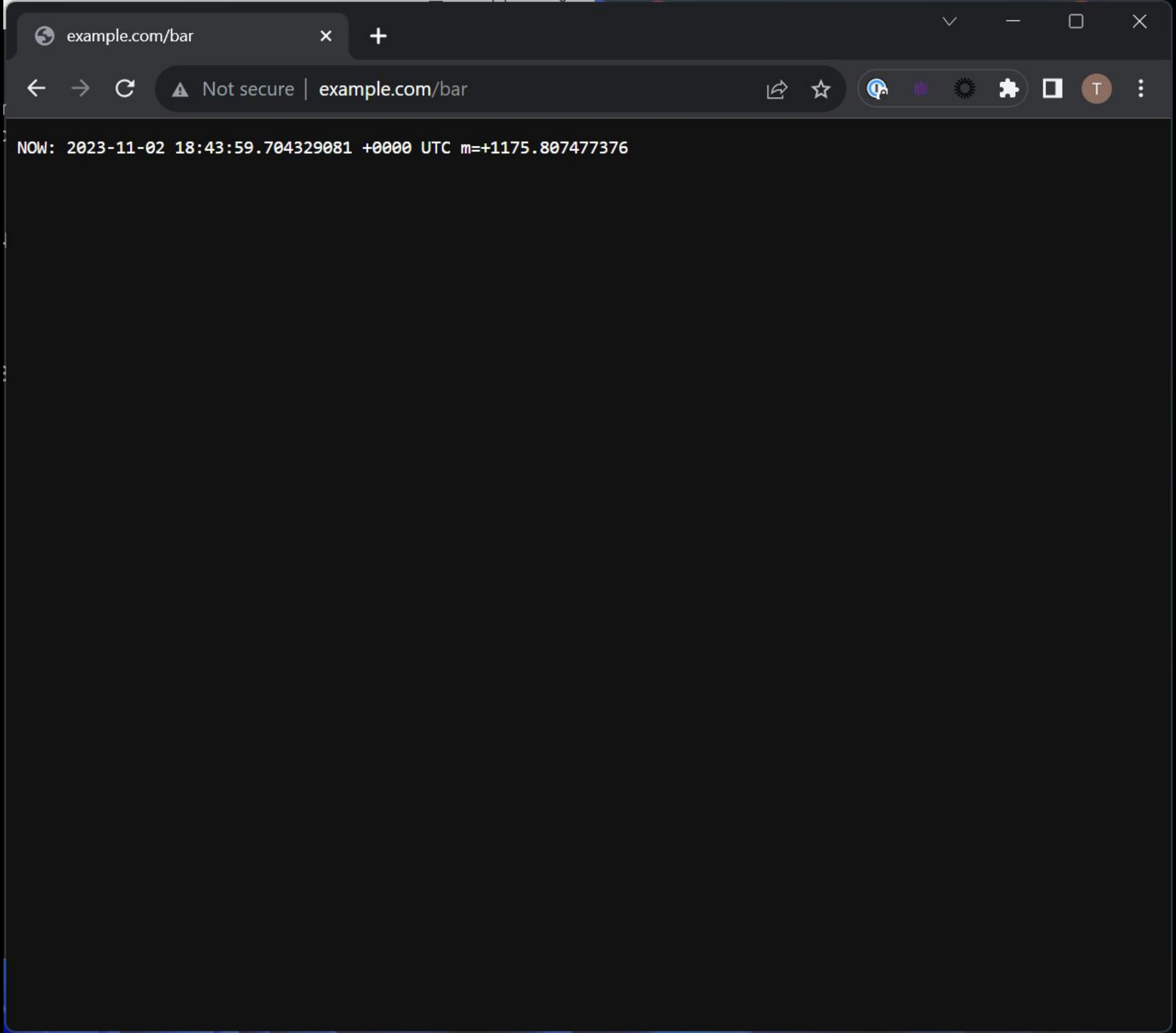
Application

Return a chain of signed certificates

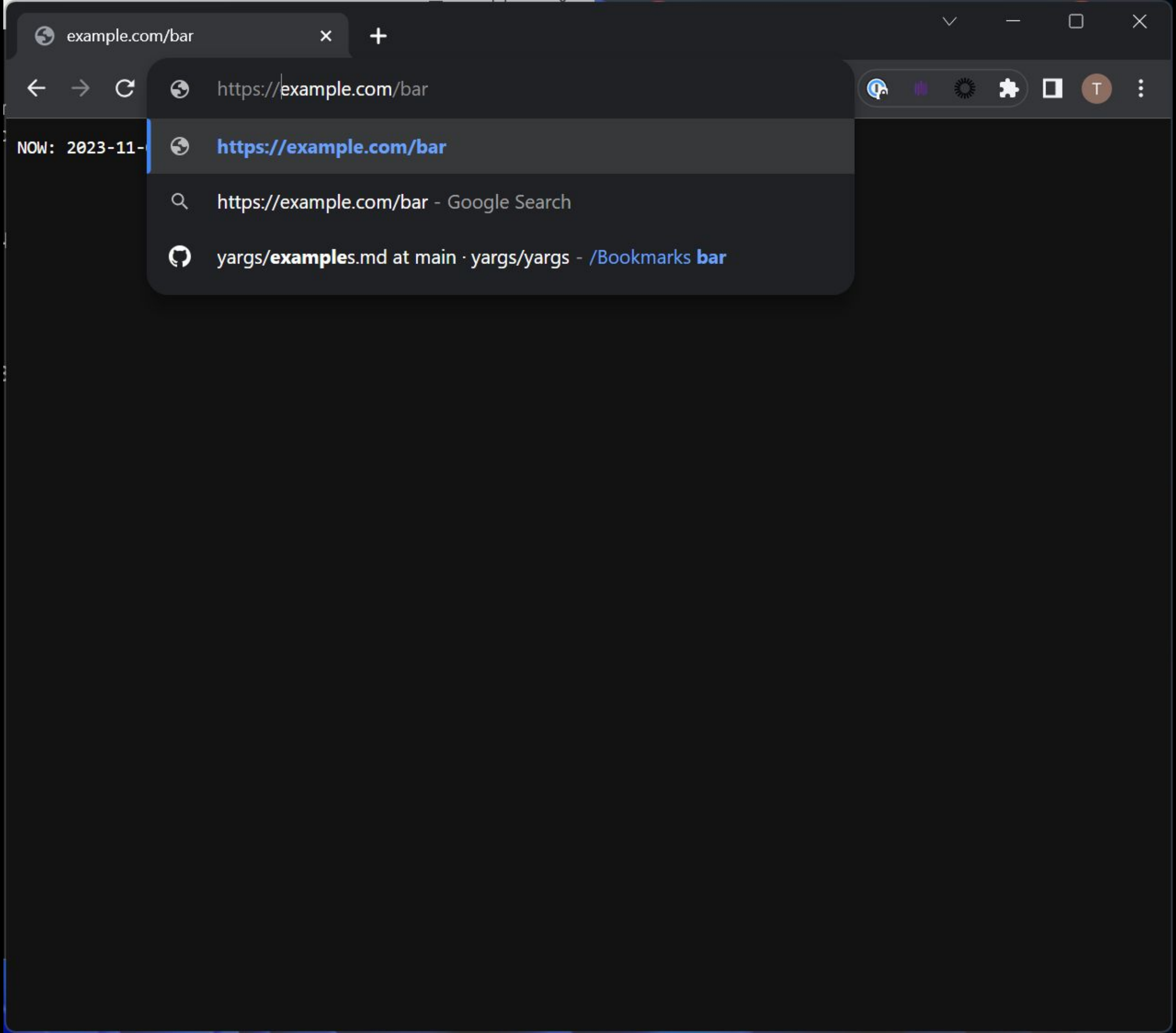
Certificate authority



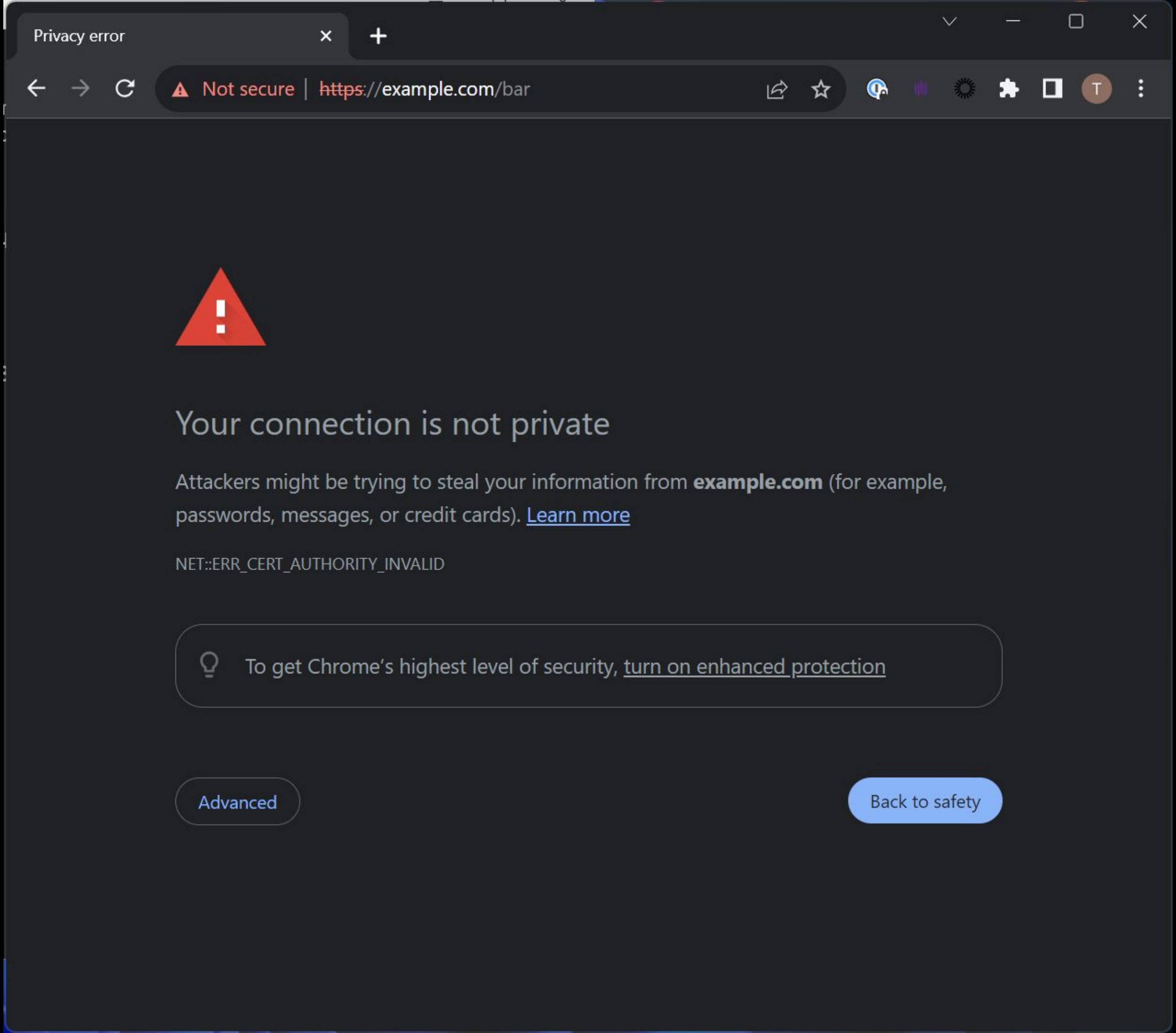
```
tramlot@tramlot:~$ k get ingress
NAME              CLASS    HOSTS         ADDRESS     PORTS   AGE
example-ingress   <none>   example.com   localhost   80      19m
tramlot@tramlot:~$ k get ingress example-ingress -oyaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  creationTimestamp: "2023-11-02T18:24:06Z"
  generation: 2
  name: example-ingress
  namespace: default
  resourceVersion: "1908"
  uid: 8ca83e2b-7fe8-4ada-9f24-ca5102855132
spec:
  rules:
  - host: example.com
    http:
      paths:
      - backend:
          service:
            name: bar-service
            port:
              number: 8080
        path: /bar
        pathType: Prefix
status:
  loadBalancer:
    ingress:
    - hostname: localhost
tramlot@tramlot:~$
```



```
NOW: 2023-11-02 18:43:59.704329081 +0000 UTC m=+1175.807477376
```


example.com/bar
https://example.com/bar
https://example.com/bar
https://example.com/bar - Google Search
yargs/examples.md at main · yargs/yargs - /Bookmarks bar
T

```
NOW: 2023-11-
```

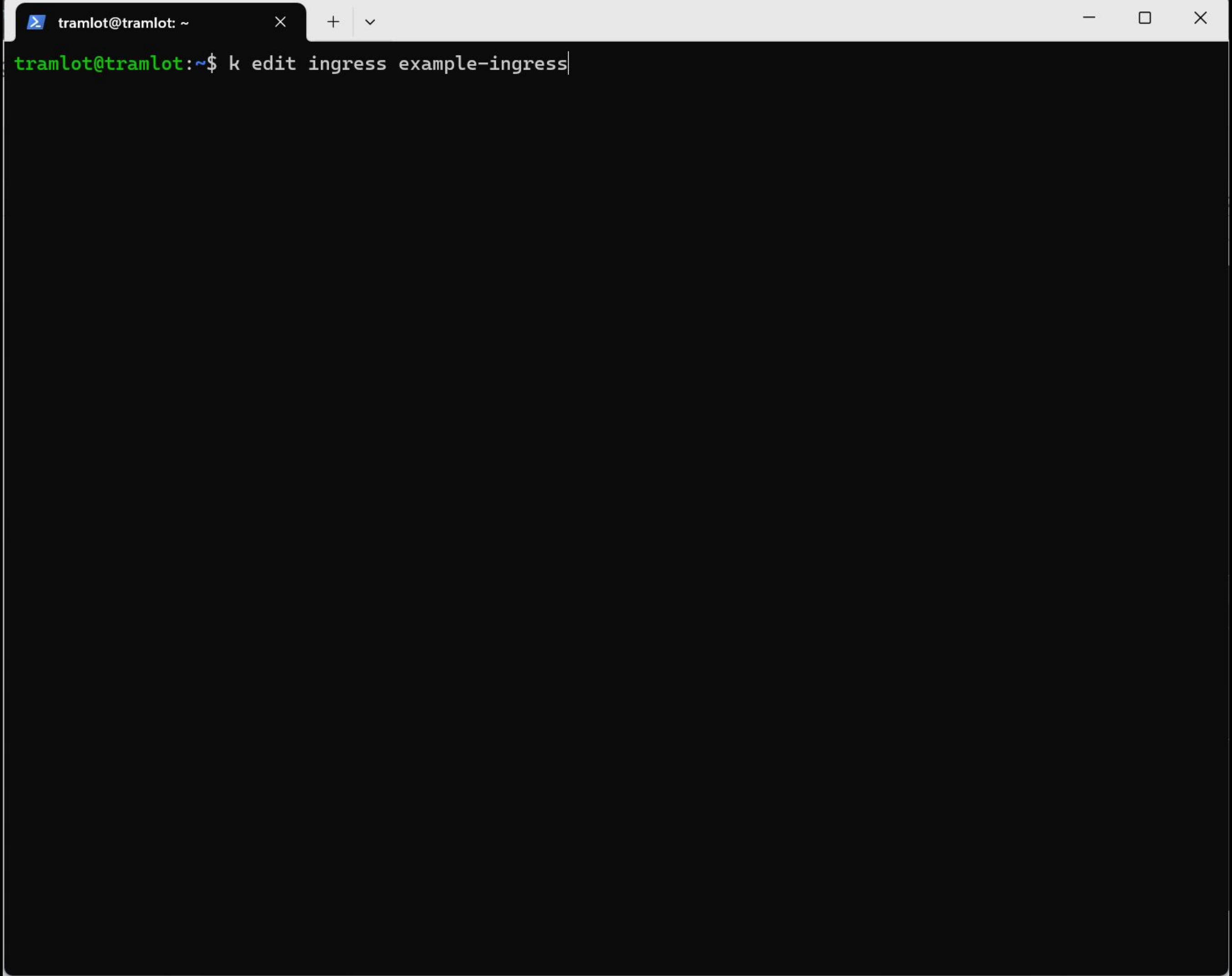


Your connection is not private

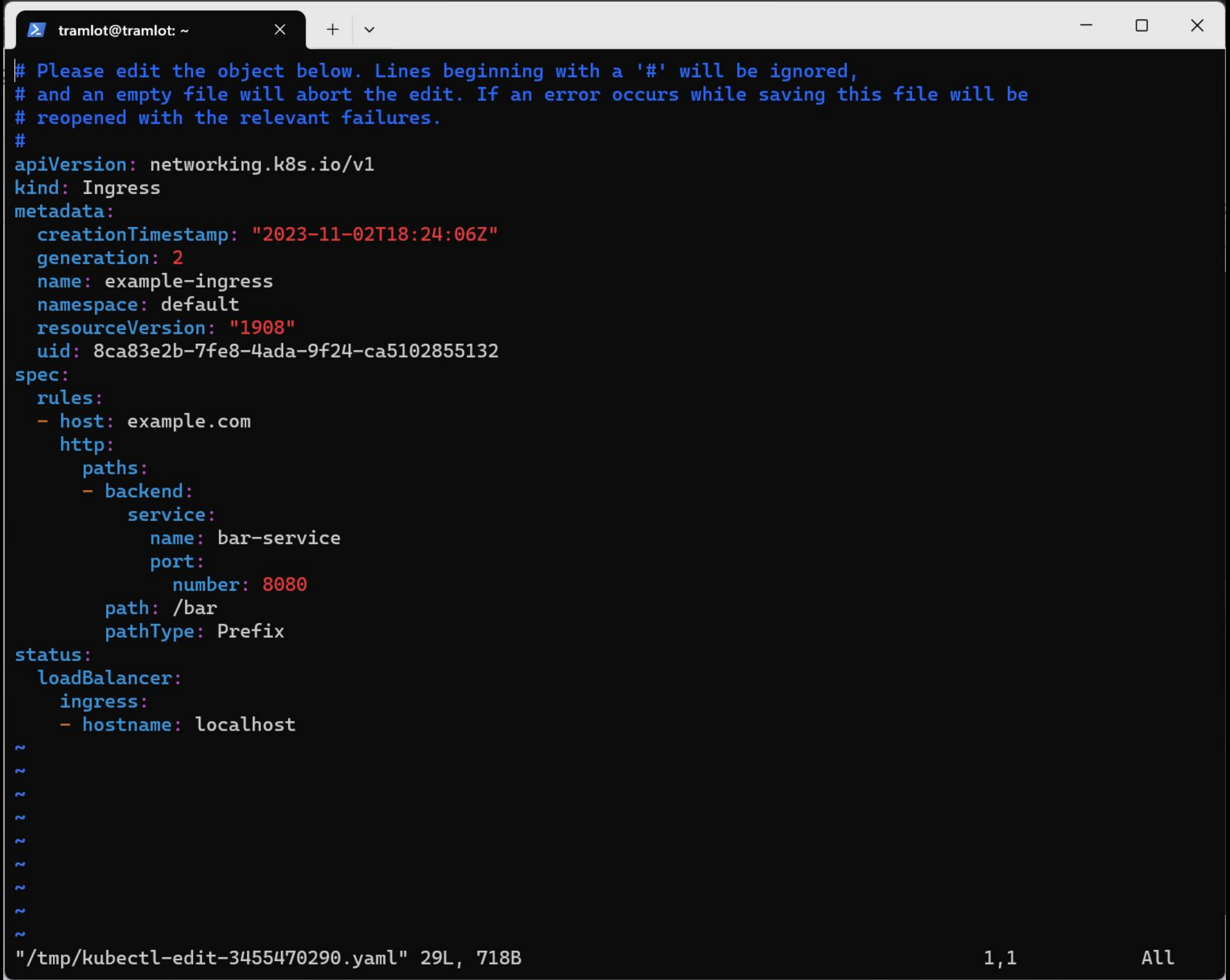
Attackers might be trying to steal your information from **example.com** (for example, passwords, messages, or credit cards). Learn more

NET::ERR_CERT_AUTHORITY_INVALID

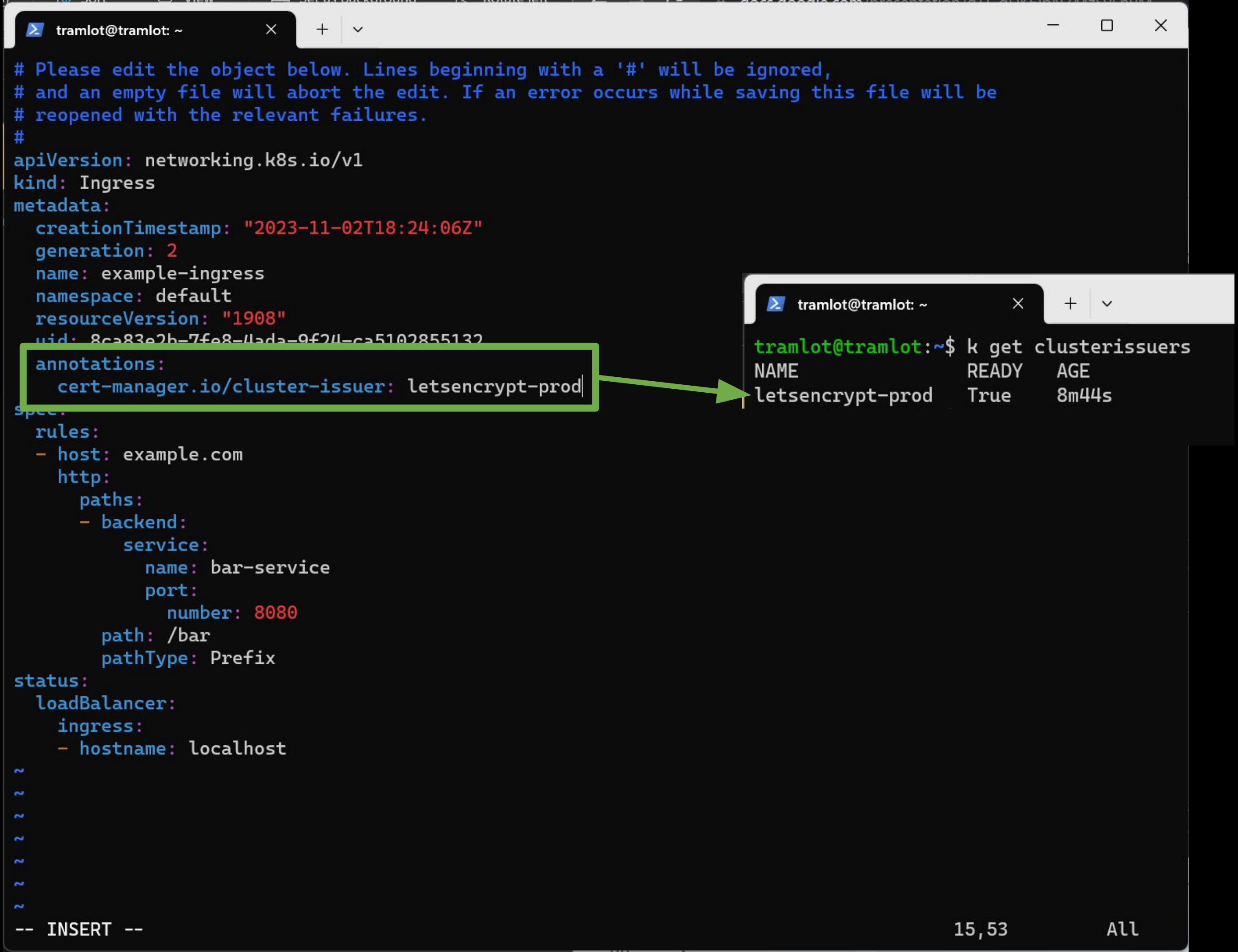
To get Chrome’s highest level of security, turn on enhanced protection



```
tramlot@tramlot:~$ k edit ingress example-ingress
```

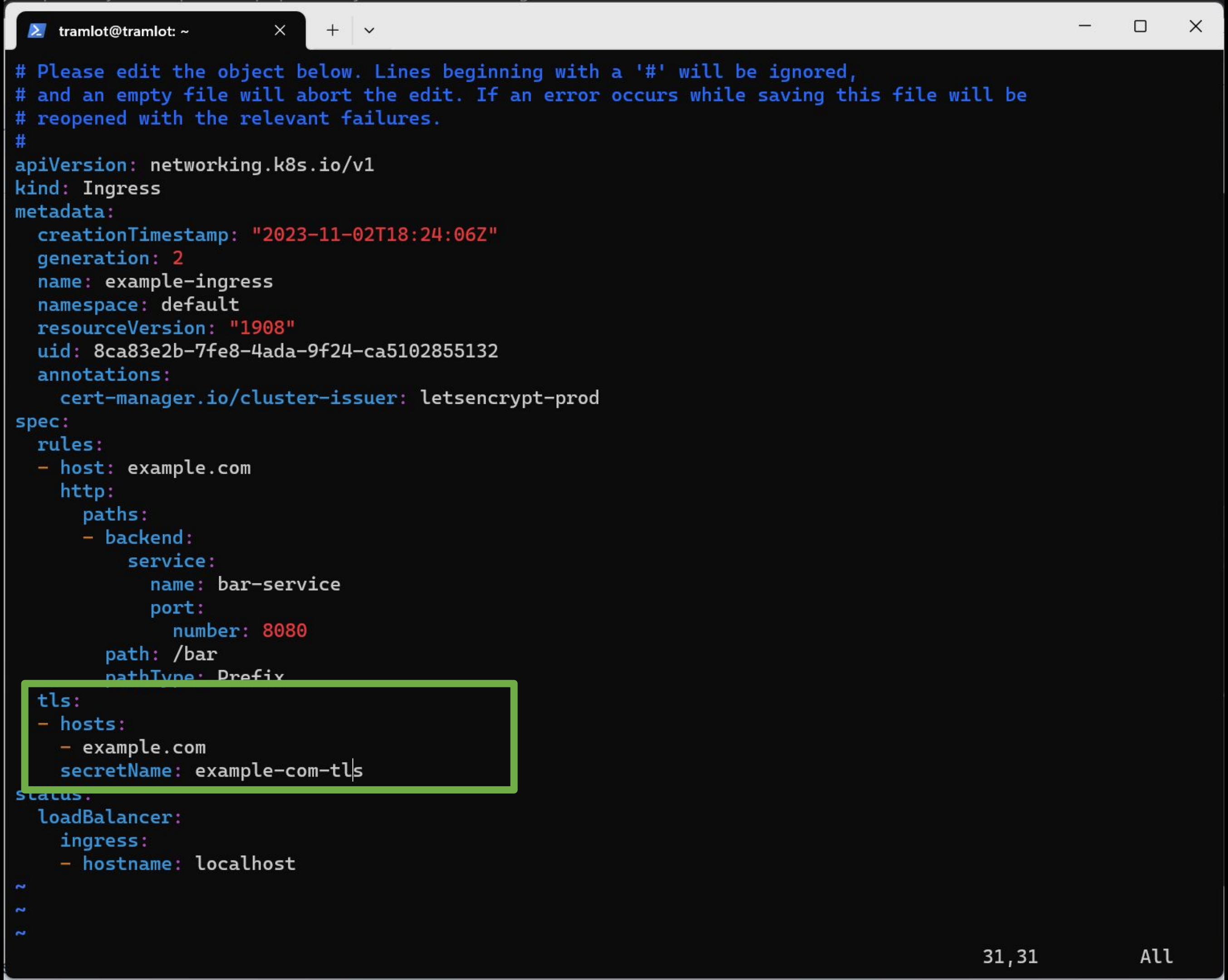


```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  creationTimestamp: "2023-11-02T18:24:06Z"
  generation: 2
  name: example-ingress
  namespace: default
  resourceVersion: "1908"
  uid: 8ca83e2b-7fe8-4ada-9f24-ca5102855132
spec:
  rules:
  - host: example.com
    http:
      paths:
      - backend:
          service:
            name: bar-service
            port:
              number: 8080
        path: /bar
        pathType: Prefix
status:
  loadBalancer:
    ingress:
    - hostname: localhost
~
~
~
~
~
~
~
~
~
"/tmp/kubectl-edit-3455470290.yaml" 29L, 718B                                  1,1           All
```

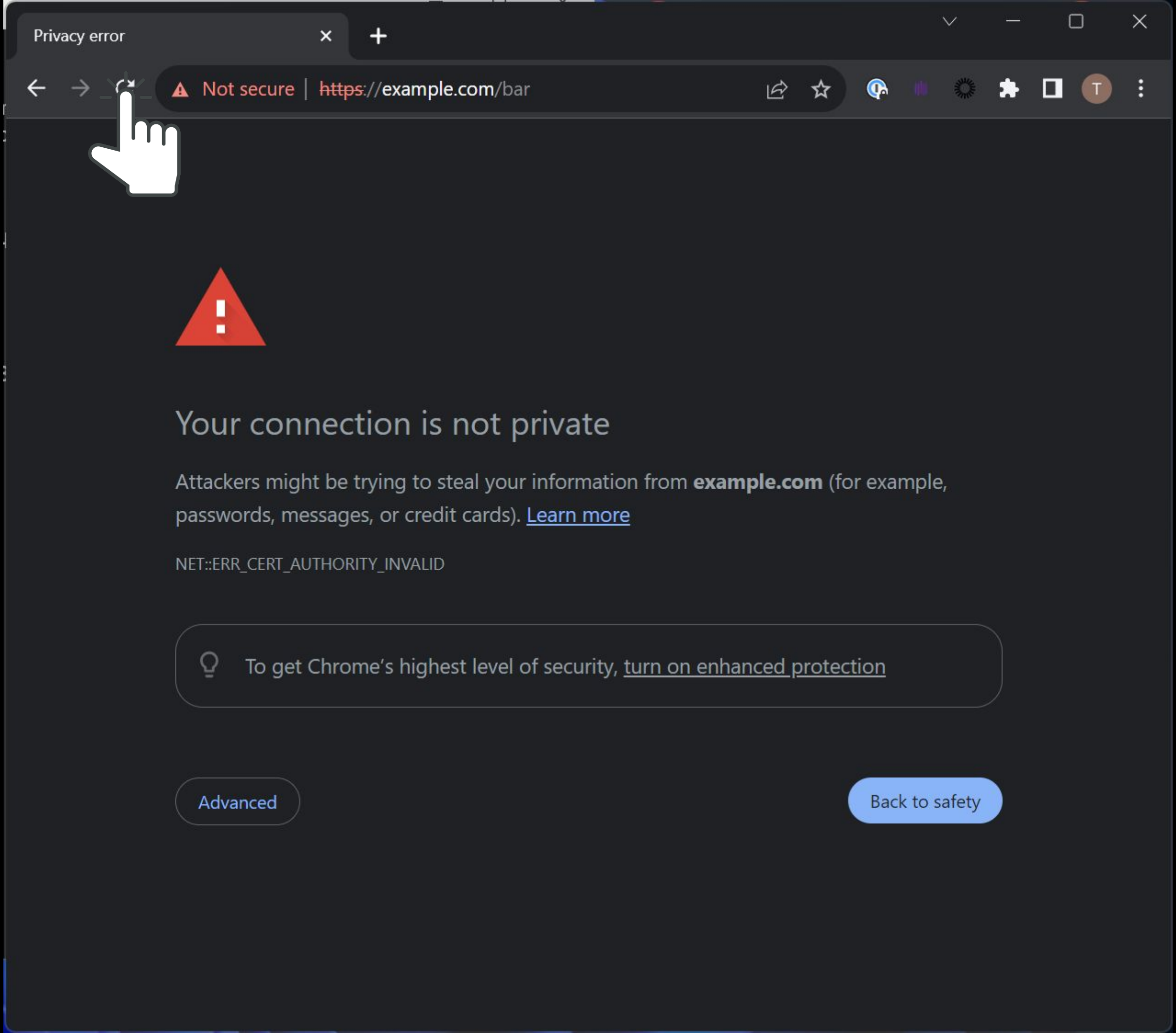


```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  creationTimestamp: "2023-11-02T18:24:06Z"
  generation: 2
  name: example-ingress
  namespace: default
  resourceVersion: "1908"
  uid: 8ca83e2b-7fe8-4ada-9f24-ca5102855132
  annotations:
    cert-manager.io/cluster-issuer: letsencrypt-prod
spec:
  rules:
  - host: example.com
    http:
      paths:
      - backend:
          service:
            name: bar-service
            port:
              number: 8080
        path: /bar
        pathType: Prefix
status:
  loadBalancer:
    ingress:
    - hostname: localhost
~
~
~
~
~
~
~
-- INSERT --                                                    15,53         All
```

```
tramlot@tramlot:~$ k get clusterissuers
NAME               READY   AGE
letsencrypt-prod   True    8m44s
```



```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  creationTimestamp: "2023-11-02T18:24:06Z"
  generation: 2
  name: example-ingress
  namespace: default
  resourceVersion: "1908"
  uid: 8ca83e2b-7fe8-4ada-9f24-ca5102855132
  annotations:
    cert-manager.io/cluster-issuer: letsencrypt-prod
spec:
  rules:
  - host: example.com
    http:
      paths:
      - backend:
          service:
            name: bar-service
            port:
              number: 8080
        path: /bar
        pathType: Prefix
  tls:
  - hosts:
    - example.com
    secretName: example-com-tls
status:
  loadBalancer:
    ingress:
    - hostname: localhost
~
~
~
                                                                  31,31         All
```

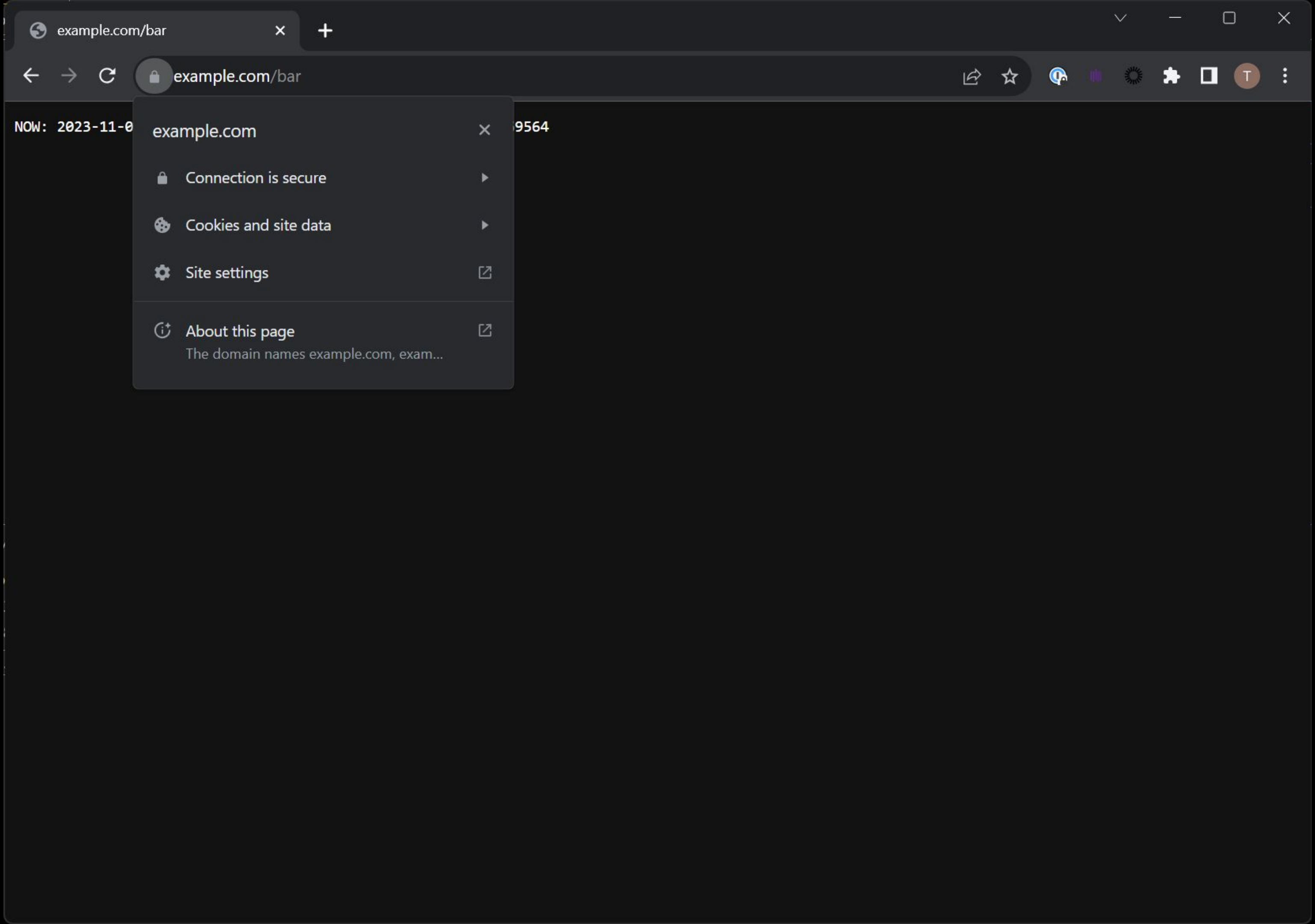


Your connection is not private

Attackers might be trying to steal your information from **example.com** (for example, passwords, messages, or credit cards). Learn more

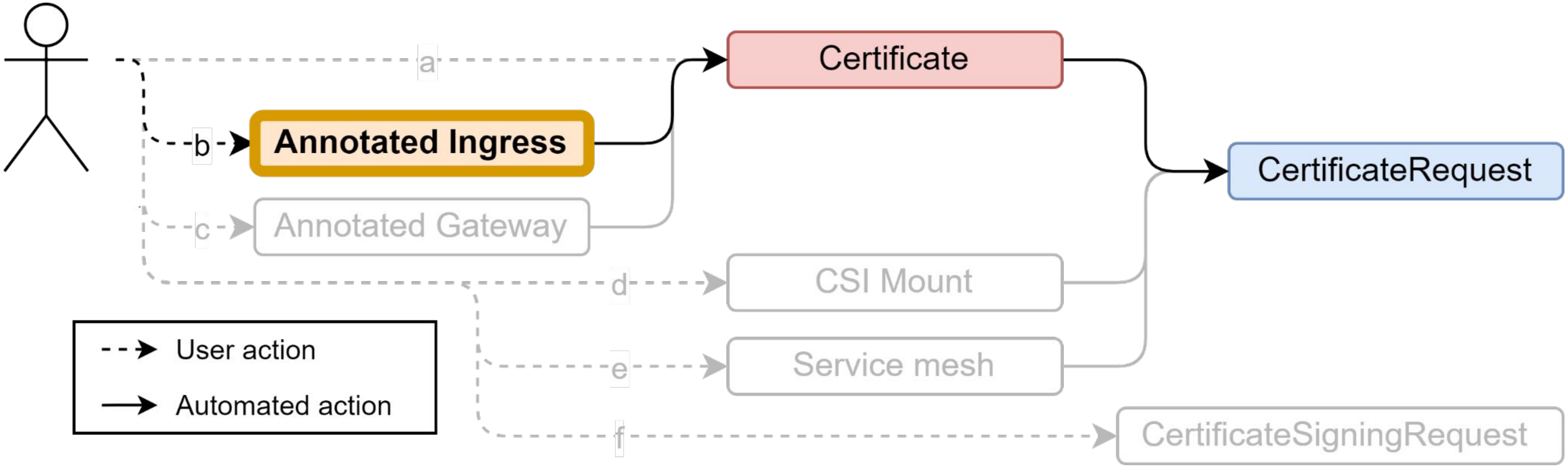
NET::ERR_CERT_AUTHORITY_INVALID

To get Chrome’s highest level of security, turn on enhanced protection

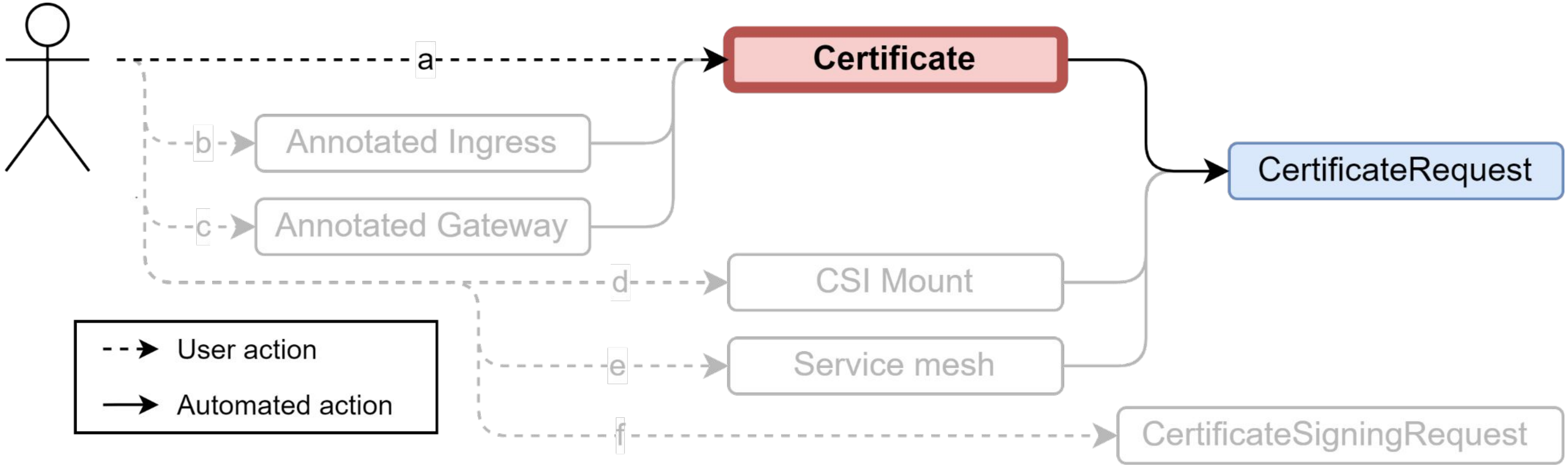


```
NOW: 2023-11-0                         9564
```

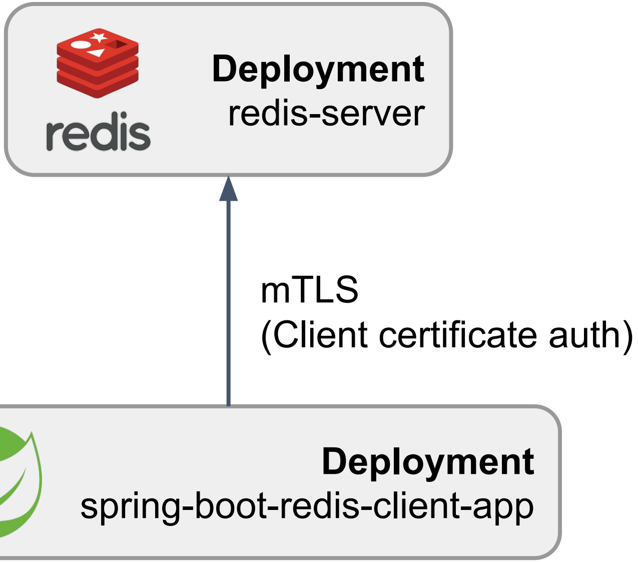
Level 1 / Provision Certs using Ingress Annotations



Level 2 / Using the Certificate Resource

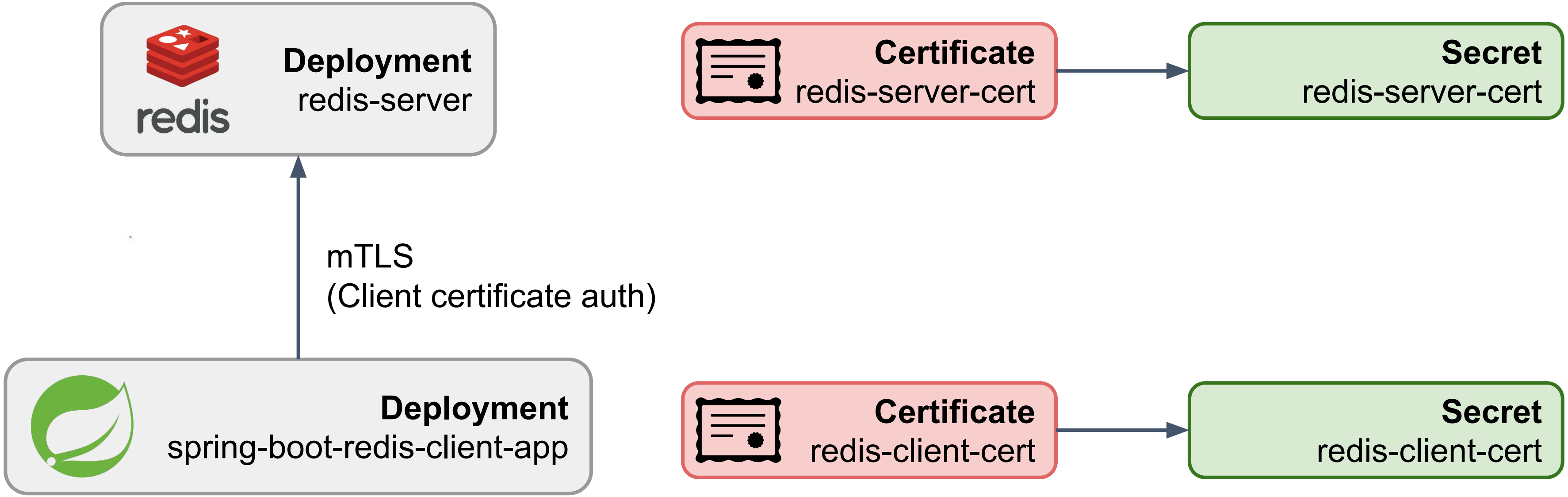


Level 2 / Using the Certificate Resource



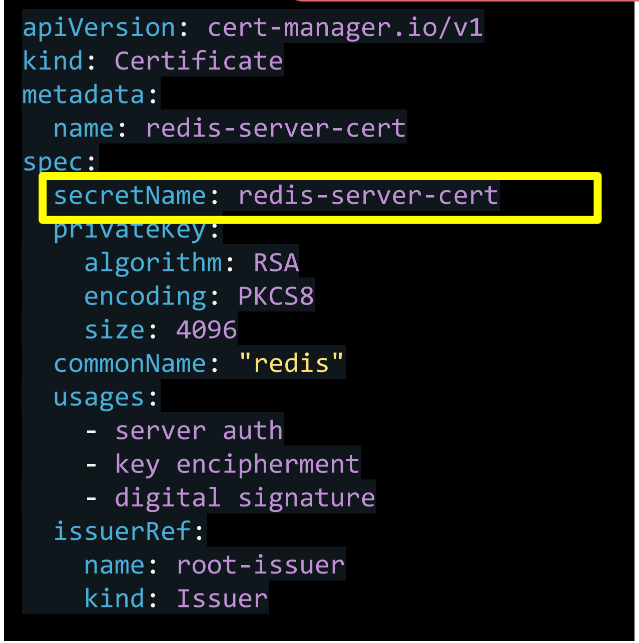
Learn more:
https://tanzu.vmware.com/developer/guides/kubernetes-mtls/

Level 2 / Using the Certificate Resource



Level 2 / Using the Certificate Resource

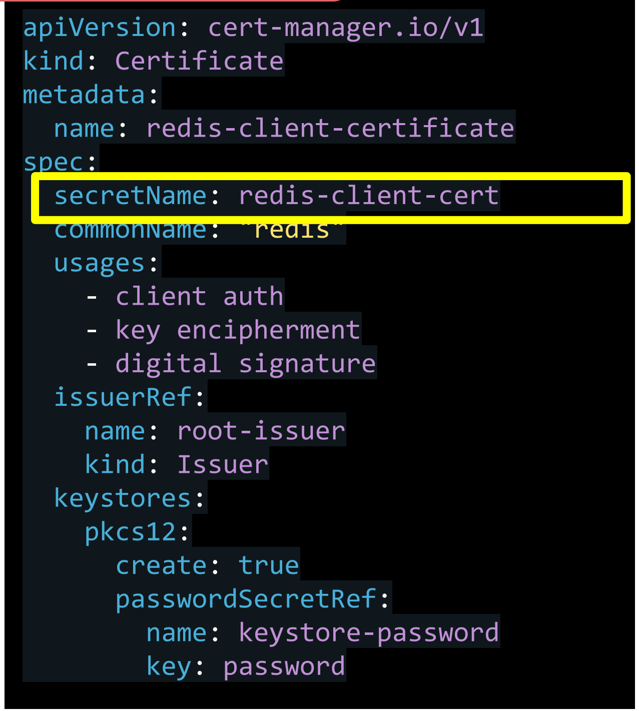
Certificate
redis-server-cert



```yaml
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: redis-server-cert
spec:
  secretName: redis-server-cert
  privateKey:
    algorithm: RSA
    encoding: PKCS8
    size: 4096
  commonName: "redis"
  usages:
    - server auth
    - key encipherment
    - digital signature
  issuerRef:
    name: root-issuer
    kind: Issuer
```

Secrets mounted to the Pods

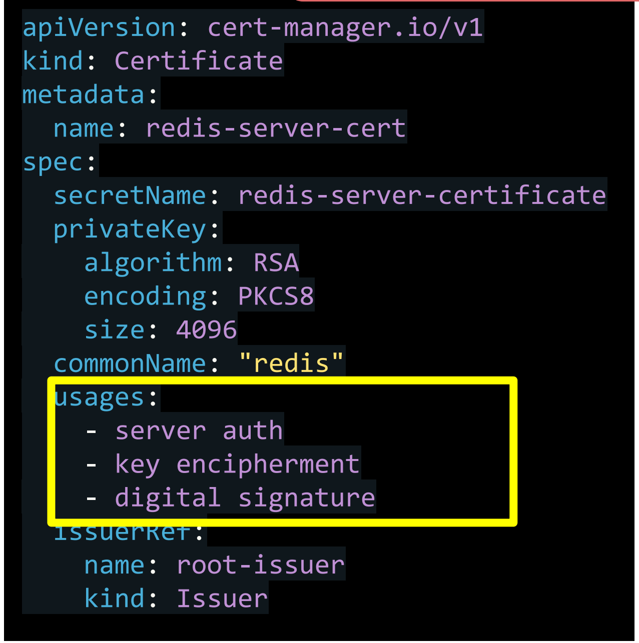
Certificate
java-spring-cert



```yaml
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: redis-client-certificate
spec:
  secretName: redis-client-cert
  commonName: "redis"
  usages:
    - client auth
    - key encipherment
    - digital signature
  issuerRef:
    name: root-issuer
    kind: Issuer
  keystores:
    pkcs12:
      create: true
      passwordSecretRef:
        name: keystore-password
        key: password
```

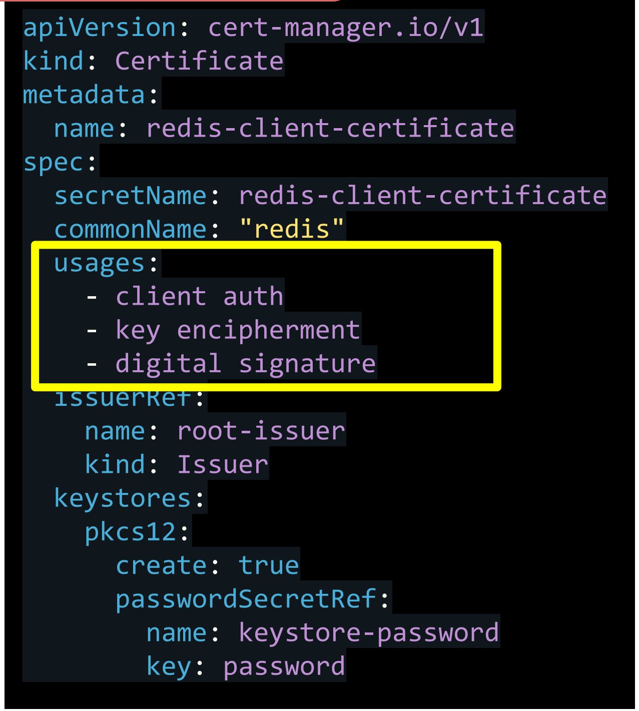
Level 2 / Using the Certificate Resource

Certificate
redis-server-cert



```yaml
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: redis-server-cert
spec:
  secretName: redis-server-certificate
  privateKey:
    algorithm: RSA
    encoding: PKCS8
    size: 4096
  commonName: "redis"
  usages:
    - server auth
    - key encipherment
    - digital signature
  issuerRef:
    name: root-issuer
    kind: Issuer
```

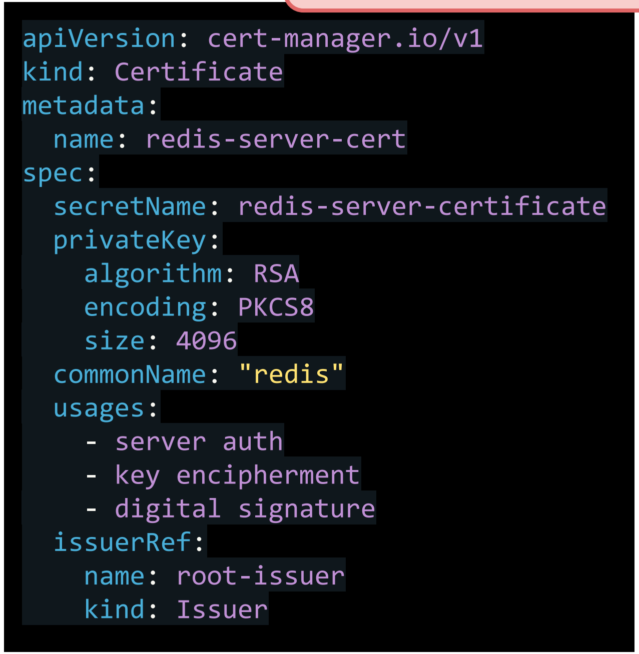
Certificate
java-spring-cert



```yaml
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: redis-client-certificate
spec:
  secretName: redis-client-certificate
  commonName: "redis"
  usages:
    - client auth
    - key encipherment
    - digital signature
  issuerRef:
    name: root-issuer
    kind: Issuer
  keystores:
    pkcs12:
      create: true
      passwordSecretRef:
        name: keystore-password
        key: password
```

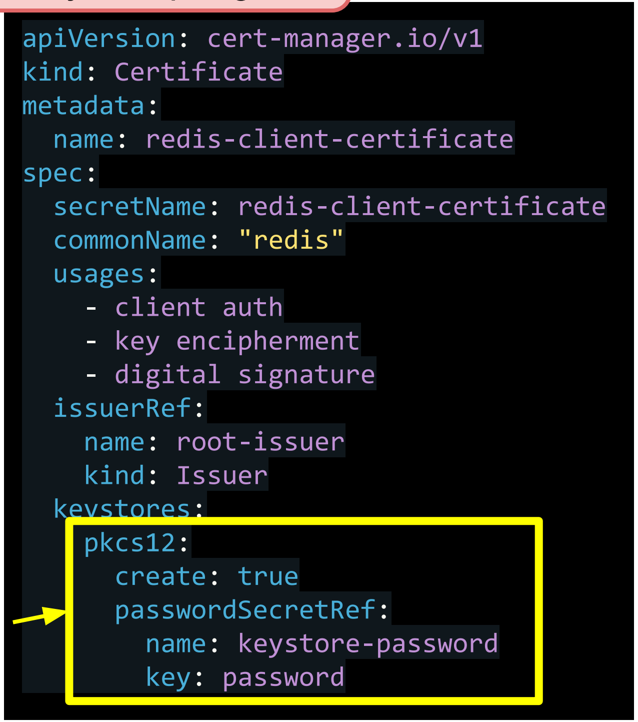
Level 2 / Using the Certificate Resource

Certificate
redis-server-cert



```yaml
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: redis-server-cert
spec:
  secretName: redis-server-certificate
  privateKey:
    algorithm: RSA
    encoding: PKCS8
    size: 4096
  commonName: "redis"
  usages:
    - server auth
    - key encipherment
    - digital signature
  issuerRef:
    name: root-issuer
    kind: Issuer
```

Certificate
java-spring-cert

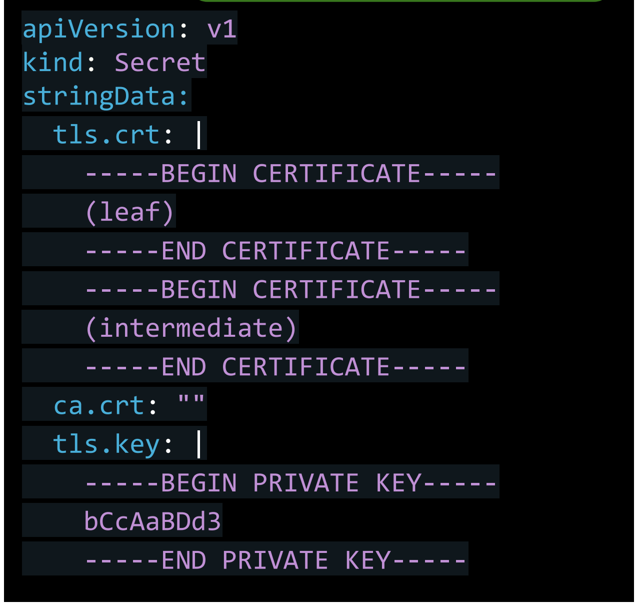


```yaml
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: redis-client-certificate
spec:
  secretName: redis-client-certificate
  commonName: "redis"
  usages:
    - client auth
    - key encipherment
    - digital signature
  issuerRef:
    name: root-issuer
    kind: Issuer
  keystores:
    pkcs12:
      create: true
      passwordSecretRef:
        name: keystore-password
        key: password
```

Java

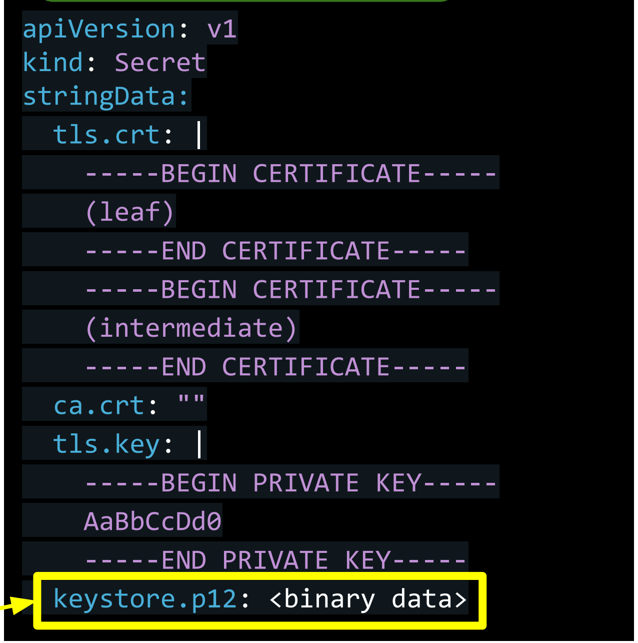
Level 2 / Using the Certificate Resource

Secret
redis-server-cert



```
apiVersion: v1
kind: Secret
stringData:
  tls.crt: |
    -----BEGIN CERTIFICATE-----
    (leaf)
    -----END CERTIFICATE-----
    -----BEGIN CERTIFICATE-----
    (intermediate)
    -----END CERTIFICATE-----
  ca.crt: ""
  tls.key: |
    -----BEGIN PRIVATE KEY-----
    bCcAaBDd3
    -----END PRIVATE KEY-----
```

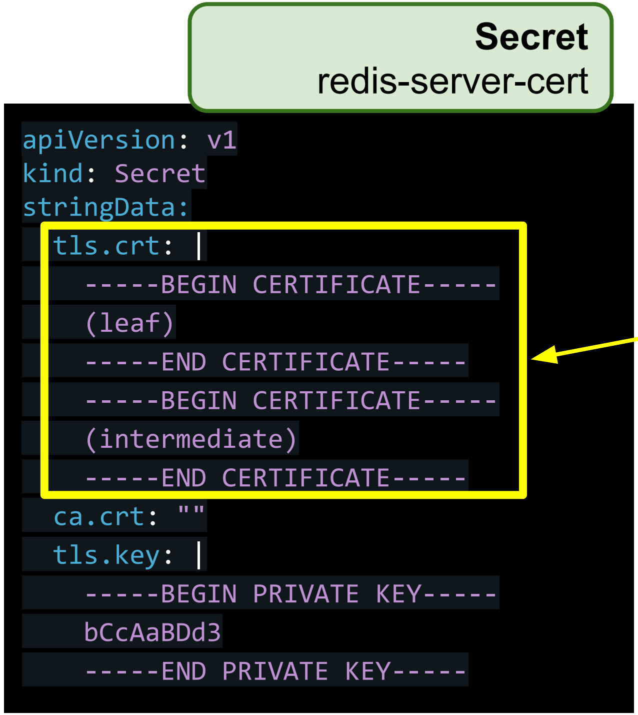
Secret
redis-client-cert



```
apiVersion: v1
kind: Secret
stringData:
  tls.crt: |
    -----BEGIN CERTIFICATE-----
    (leaf)
    -----END CERTIFICATE-----
    -----BEGIN CERTIFICATE-----
    (intermediate)
    -----END CERTIFICATE-----
  ca.crt: ""
  tls.key: |
    -----BEGIN PRIVATE KEY-----
    AaBbCcDd0
    -----END PRIVATE KEY-----
  keystore.p12: <binary data>
```

Java

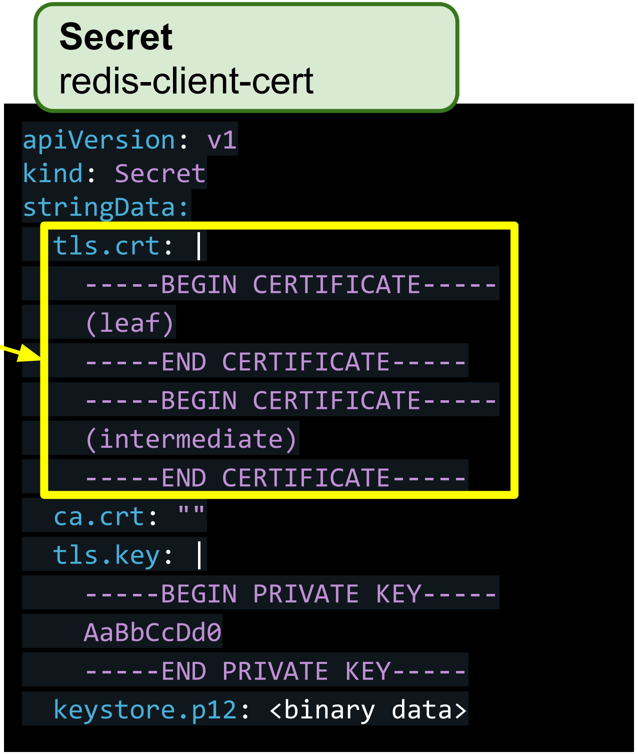
Level 2 / Using the Certificate Resource



Secret
redis-server-cert

```yaml
apiVersion: v1
kind: Secret
stringData:
  tls.crt: |
    -----BEGIN CERTIFICATE-----
    (leaf)
    -----END CERTIFICATE-----
    -----BEGIN CERTIFICATE-----
    (intermediate)
    -----END CERTIFICATE-----
  ca.crt: ""
  tls.key: |
    -----BEGIN PRIVATE KEY-----
    bCcAaBDd3
    -----END PRIVATE KEY-----
```

tls.crt =
Certificate
chain



Secret
redis-client-cert

```yaml
apiVersion: v1
kind: Secret
stringData:
  tls.crt: |
    -----BEGIN CERTIFICATE-----
    (leaf)
    -----END CERTIFICATE-----
    -----BEGIN CERTIFICATE-----
    (intermediate)
    -----END CERTIFICATE-----
  ca.crt: ""
  tls.key: |
    -----BEGIN PRIVATE KEY-----
    AaBbCcDd0
    -----END PRIVATE KEY-----
  keystore.p12: <binary data>
```

Level 2 / In Summary

✅ Certificate instead of Ingress = more control (example: PKCS12 for Java)

Level 3 / Private PKI and Trust

What if we need more control over issuance?

Level 3 / Public PKI vs Private PKI

Public

⬆

- Easy set up
- Wide support
- No CA cert to handle

⬇

- Rate limits
- Complicated issuance
- Valuable targets

Private

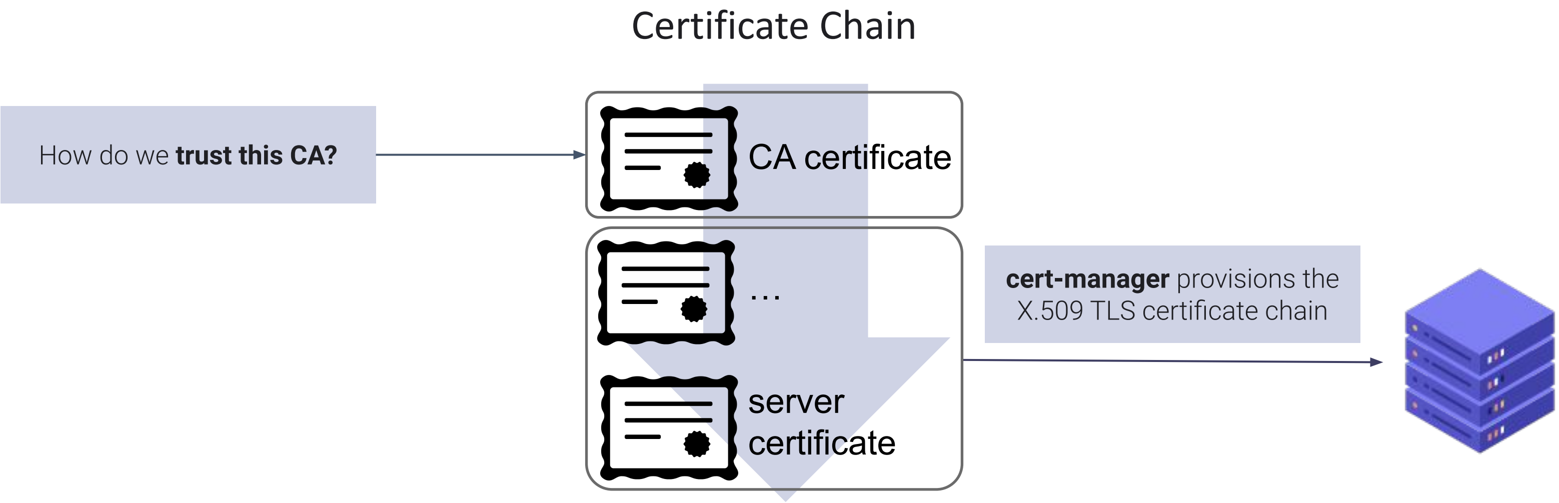
⬆

- Total control
- No rate limits
- Flexibility

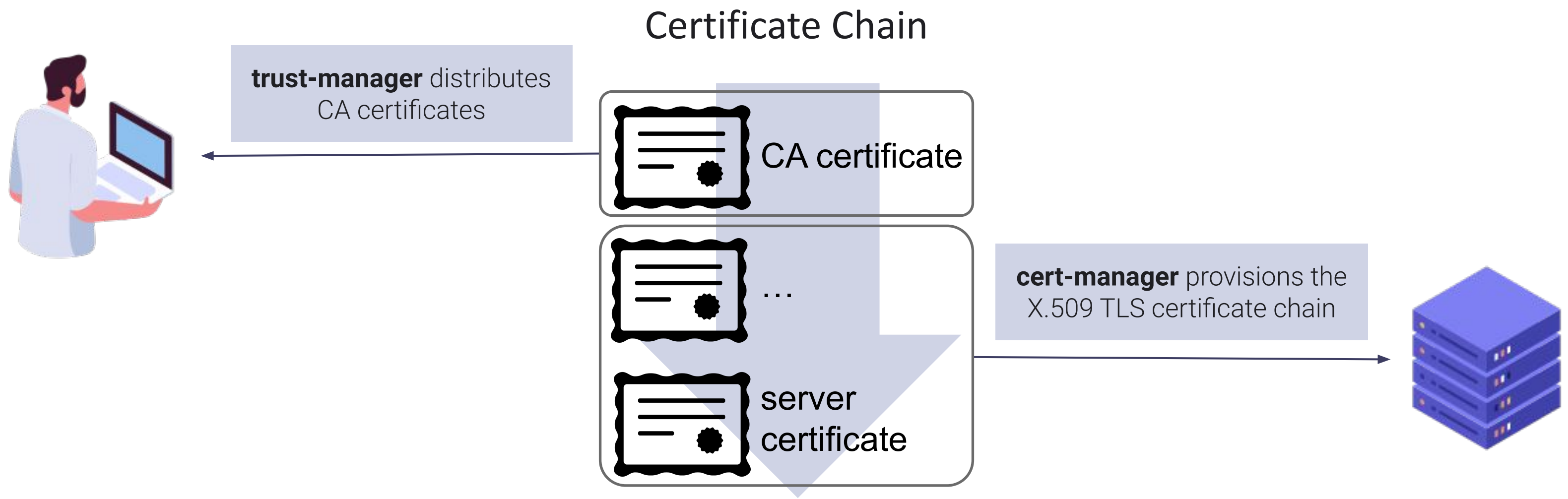
⬇

- More to manage
- Rotation is complicated
- Harder to understand

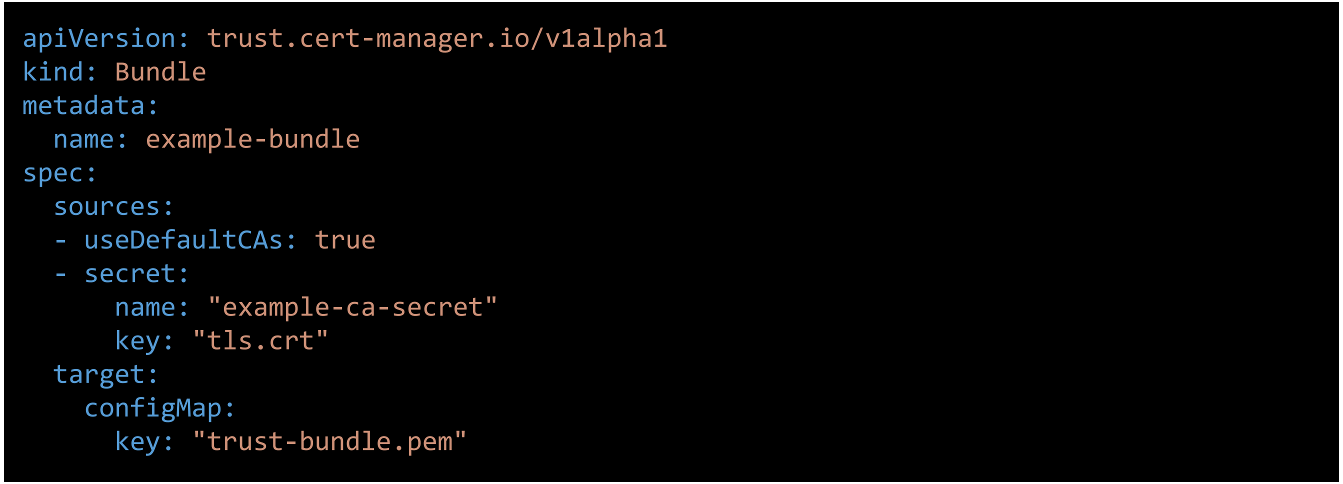
Level 3 / Private PKI: Trust Issues


Certificate Chain
How do we **trust this CA?**
CA certificate
...
server certificate
cert-manager provides the X.509 TLS certificate chain

Level 3 / Private PKI: trust-manager

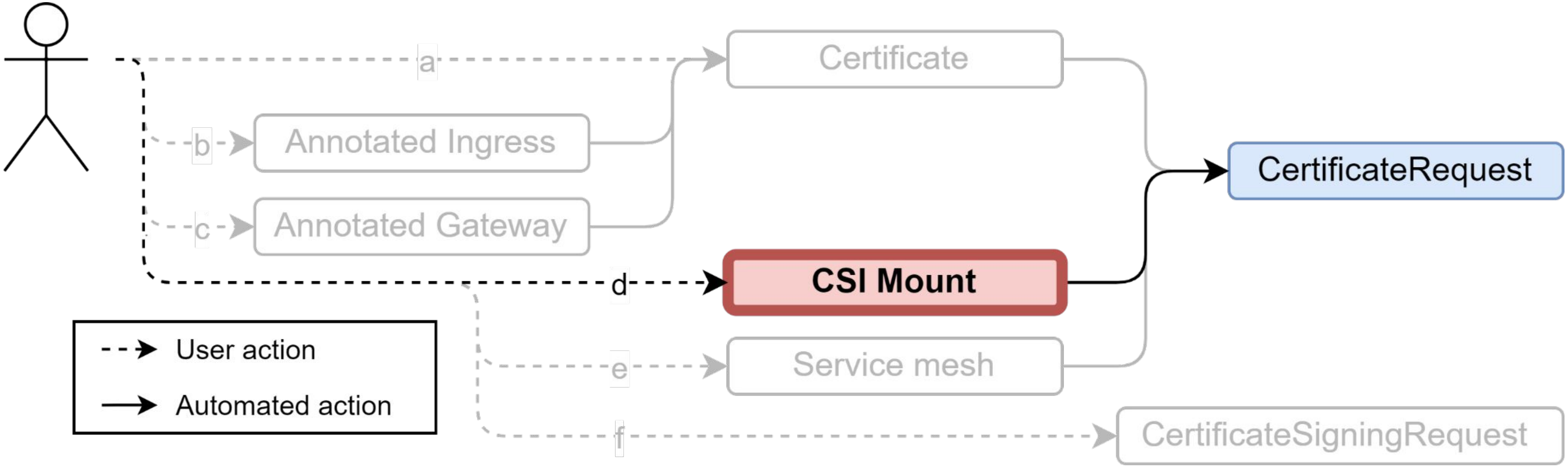

Certificate Chain
trust-manager distributes CA certificates
CA certificate
…
server certificate
cert-manager provisions the X.509 TLS certificate chain

Level 3 / trust-manager

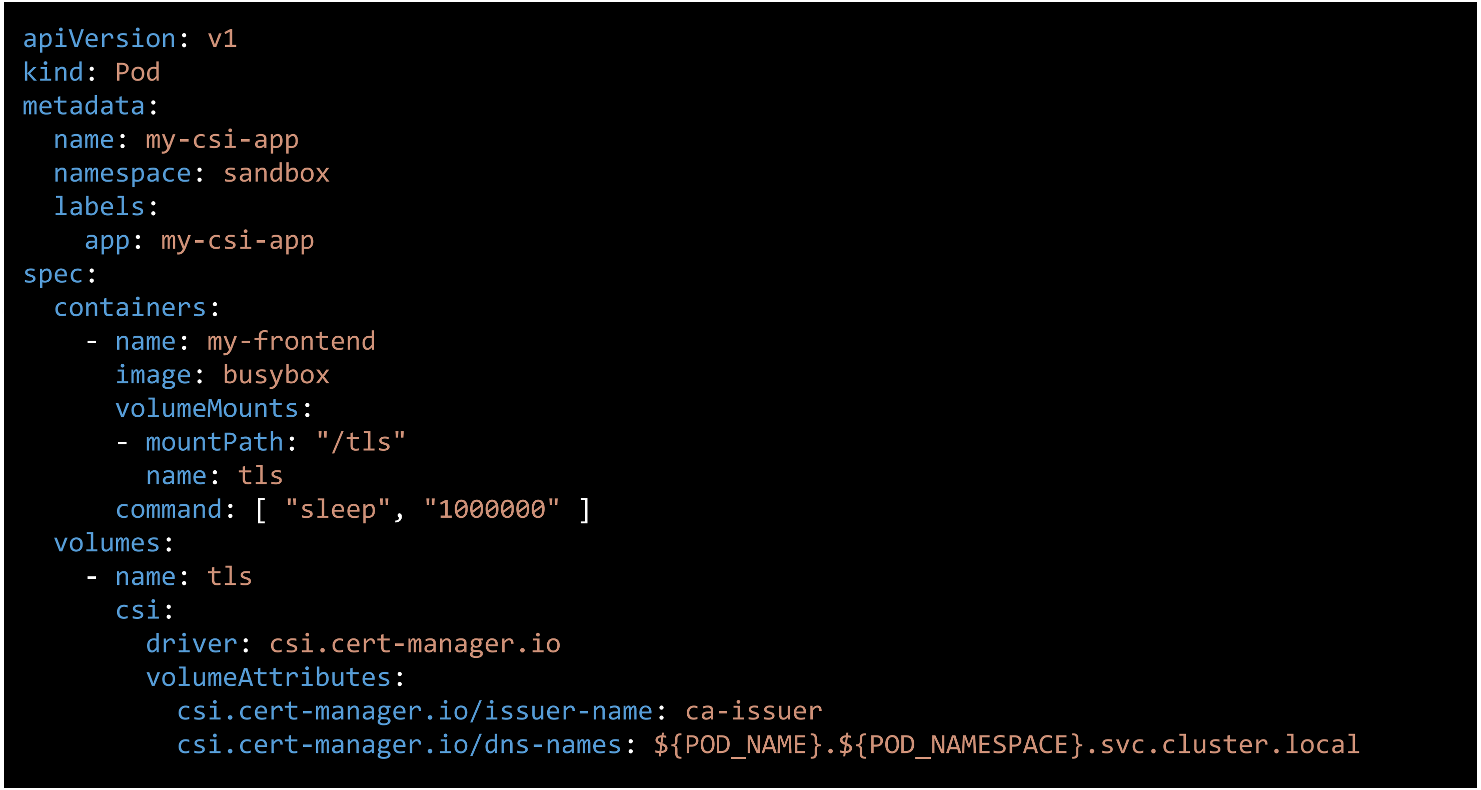


```yaml
apiVersion: trust.cert-manager.io/v1alpha1
kind: Bundle
metadata:
  name: example-bundle
spec:
  sources:
  - useDefaultCAs: true
  - secret:
      name: "example-ca-secret"
      key: "tls.crt"
  target:
    configMap:
      key: "trust-bundle.pem"
```

Level 4 / Provision Certificates with CSI Mounts

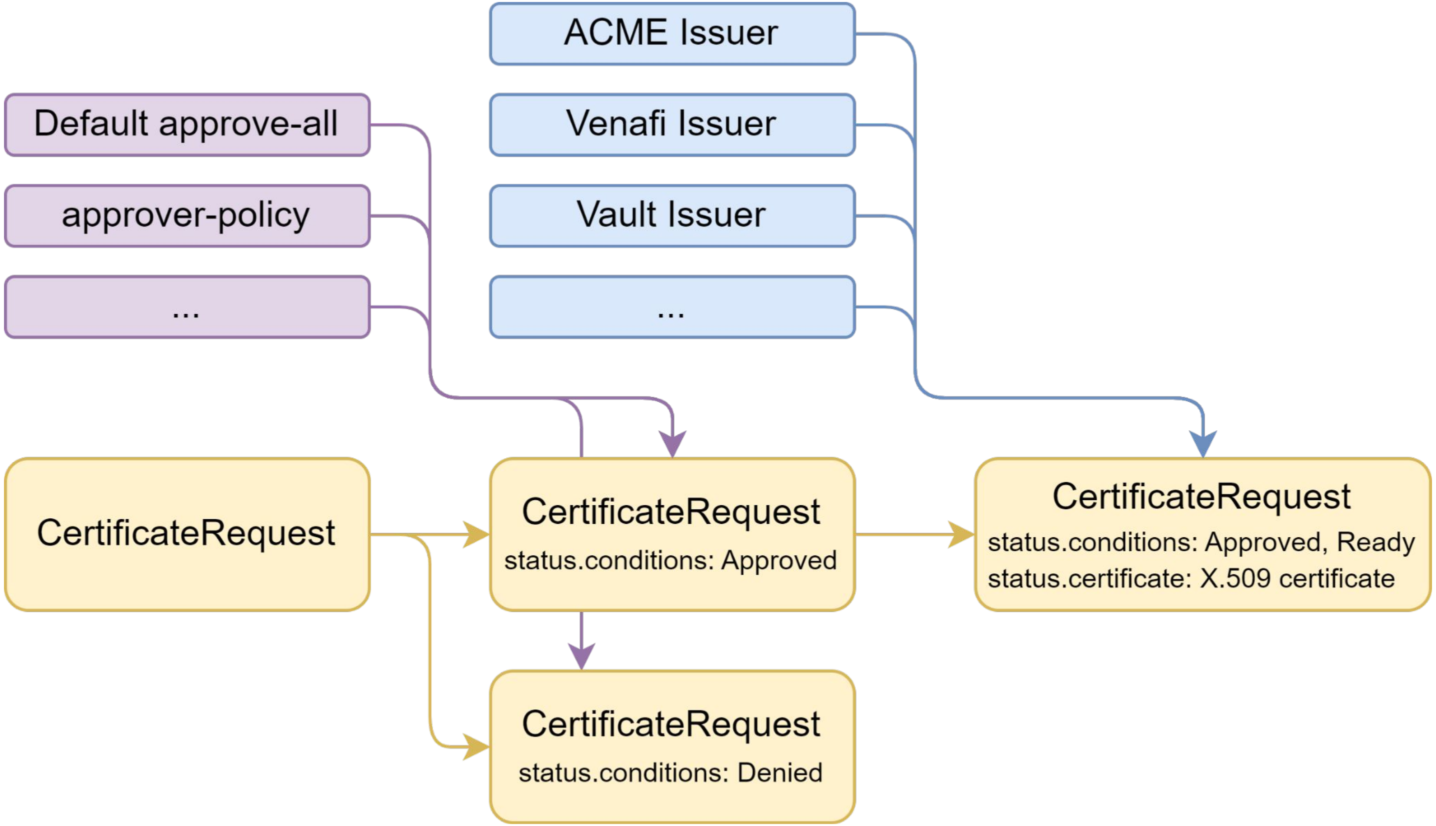


Level 4 / CSI Driver

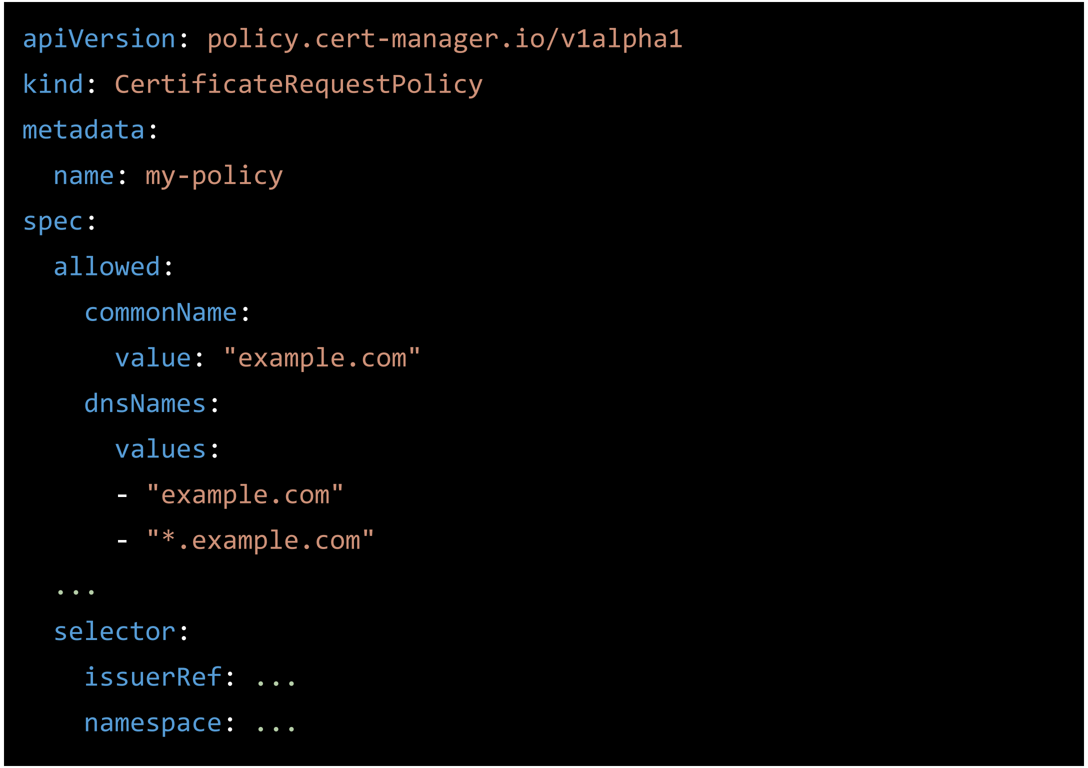


```yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-csi-app
  namespace: sandbox
  labels:
    app: my-csi-app
spec:
  containers:
    - name: my-frontend
      image: busybox
      volumeMounts:
      - mountPath: "/tls"
        name: tls
      command: [ "sleep", "1000000" ]
  volumes:
    - name: tls
      csi:
        driver: csi.cert-manager.io
        volumeAttributes:
          csi.cert-manager.io/issuer-name: ca-issuer
          csi.cert-manager.io/dns-names: ${POD_NAME}.${POD_NAMESPACE}.svc.cluster.local
```

Level 4 / Approver Policy



Level 4 / Approver Policy: CertificateRequestPolicy

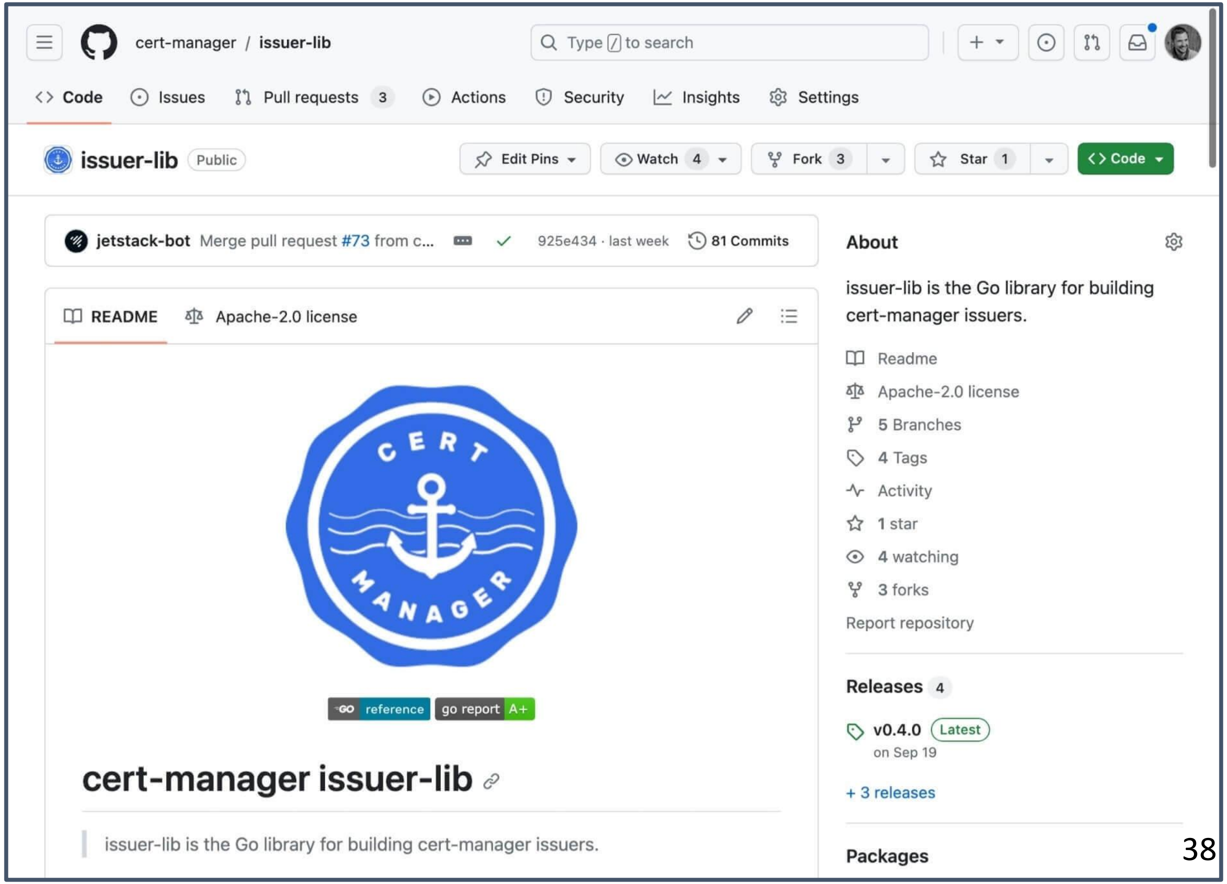


```yaml
apiVersion: policy.cert-manager.io/v1alpha1
kind: CertificateRequestPolicy
metadata:
  name: my-policy
spec:
  allowed:
    commonName:
      value: "example.com"
    dnsNames:
      values:
      - "example.com"
      - "*.example.com"
  ...
  selector:
    issuerRef: ...
    namespace: ...
```

Level 5 / Custom Issuers and Plugins

Integrating a New Certificate Issuer with issuer-lib

cert-manager/issuer-lib/



jetstack-bot Merge pull request #73 from c... 925e434 · last week 81 Commits

About

issuer-lib is the Go library for building cert-manager issuers.

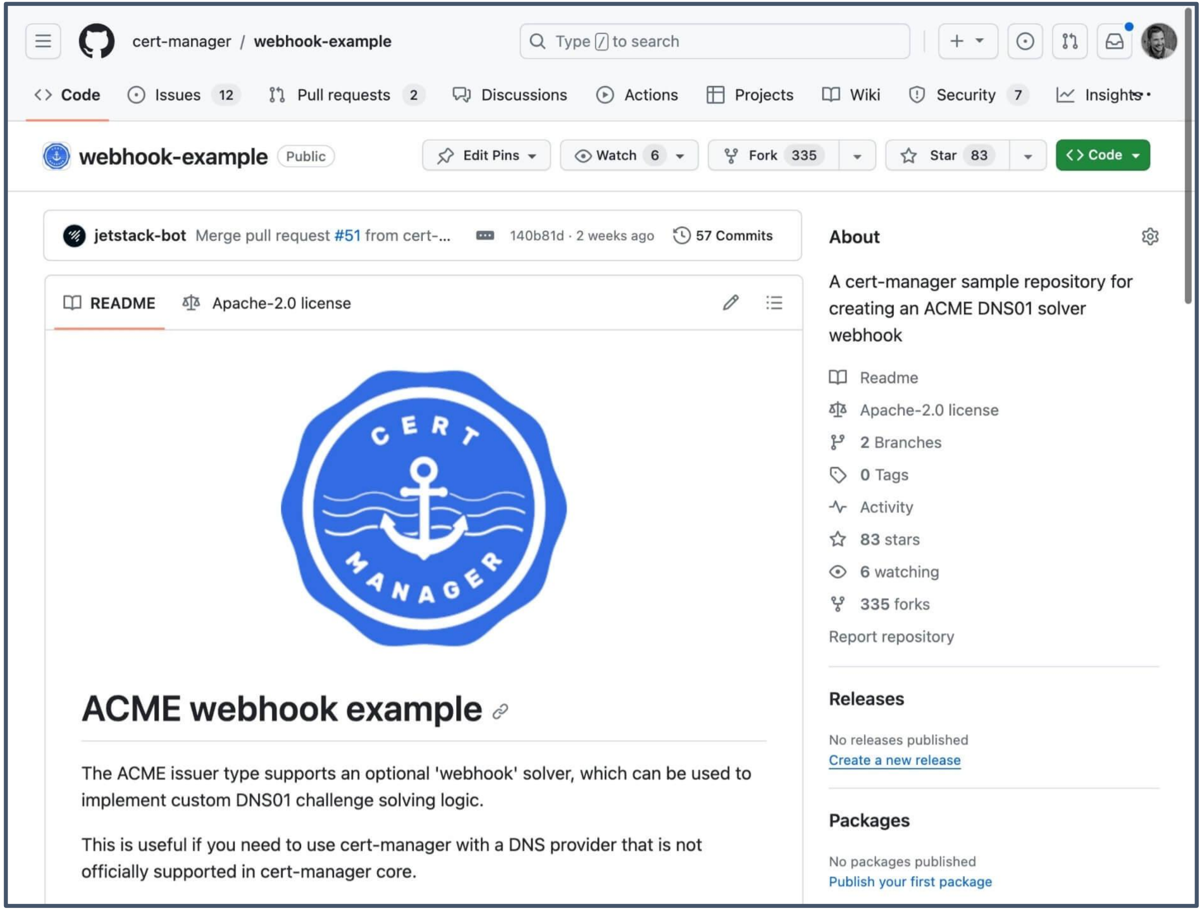
cert-manager issuer-lib

> issuer-lib is the Go library for building cert-manager issuers.

Level 5 / Custom Issuers and Plugins

Integrating With a New DNS Provider

cert-manager/webhook-example



A cert-manager sample repository for creating an ACME DNS01 solver webhook

ACME webhook example

The ACME issuer type supports an optional 'webhook' solver, which can be used to implement custom DNS01 challenge solving logic.

This is useful if you need to use cert-manager with a DNS provider that is not officially supported in cert-manager core.

Level 5 / Custom Issuers and Plugins

Integrating Your Own Approval Workflow

cert-manager/approver-policy

Building Your Own CSI Driver

cert-manager/csi-lib

Outro / Get in Touch: cert-manager Booth

Outro / Get in Touch: Meetings & Slack

- Daily “stand-ups” on European time
- North America friendly meeting every 2 weeks
- Slack is always available!

https://cert-manager.io/docs/contributing/

Outro / Get in Touch: Feedback

Leave feedback about the talk!

https://sched.co/1R2rN