Practical TLS Advice
for 2021-2030


Ted Hahn, TCB Technologies, Inc.


Mark Hahn, Ciber Global

https://gitlab.com/markphahn/practical-tls-advice

Motivation:

You want to run a secure application.

What does that mean?

- **Encryption-in-transit**
- End-to-end

https://gitlab.com/markphahn/practical-tls-advice

Part 1: Basics

https://gitlab.com/markphahn/practical-tls-advice

The Public Trust domain

Public CA Lists : CAs, Intermediates, and Certificates

- Google
  - Chrome, Android, "Distroless" docker images
- Mozilla
  - Ubuntu, FreeBSD
- Apple
  - iOS, MacOS
- Microsoft
  - Windows

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            04:69:93:b5:5d:d4:9e:82:4d:99:ae:15:a1:4e:30:cc:9e:a9
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=US, O=Let's Encrypt, CN=R3
        Validity
            Not Before: Sep 16 02:58:20 2021 GMT
            Not After : Dec 15 02:58:19 2021 GMT
        Subject: CN=demo1.do.tcbtech-corp.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:ac:de:6e:66:de:29:b1:e4:23:de:f7:52:34:b7:
```

https://gitlab.com/markphahn/practical-tls-advice

X.509 version 3 structure

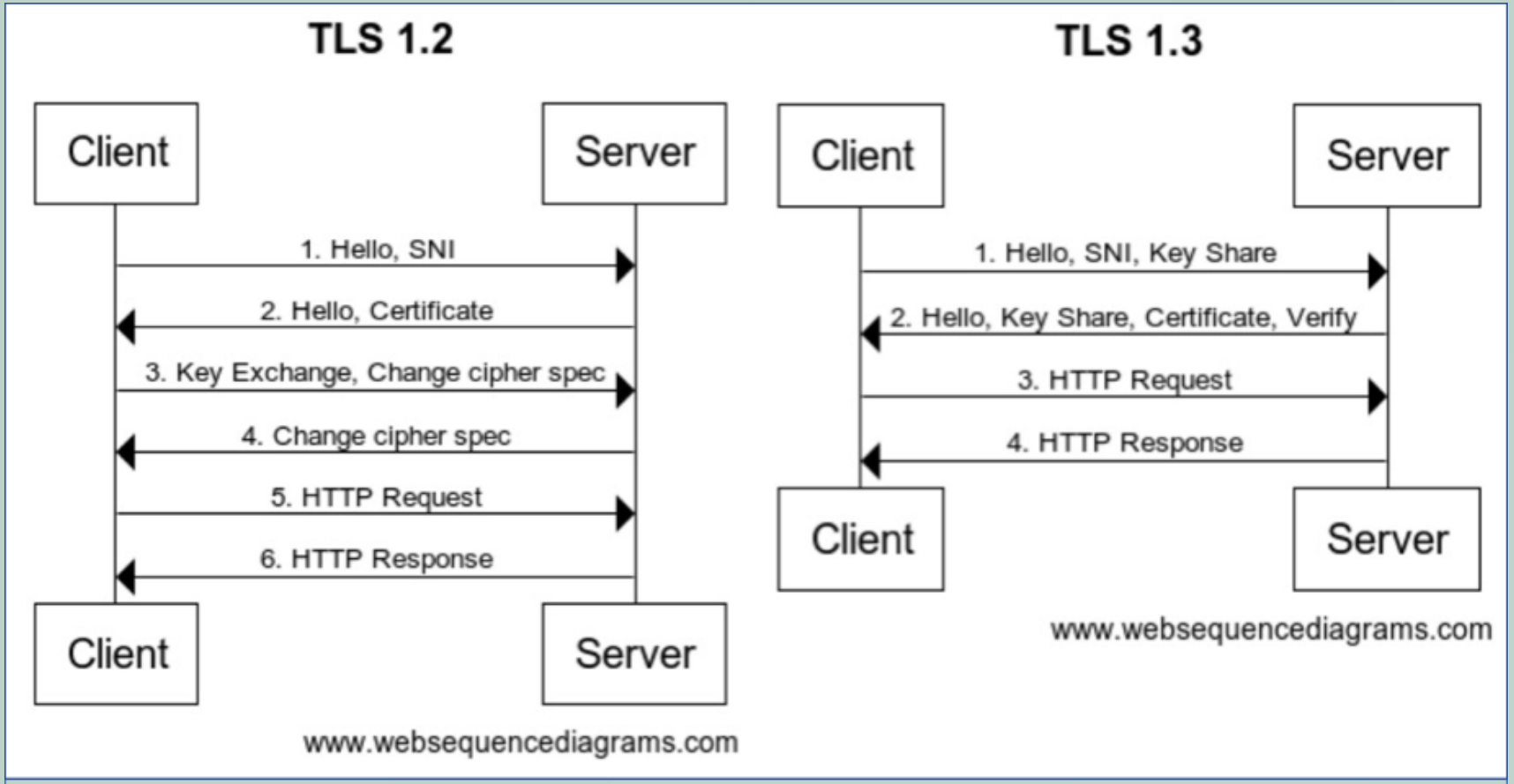
```
Certificate
    Version Number
    Serial Number
    Signature Algorithm ID
    Issuer Name
    Validity period (Not Before, Not After)
    Subject name
    Subject Public Key Info (Public Key Algorithm, Subject Public Key)
    Issuer Unique Identifier (optional)
    Subject Unique Identifier (optional)
    Extensions (optional)
        ...
Certificate Signature Algorithm
Certificate Signature
```

X.509 version 3 extensions

```
X509v3 Subject Alternative Name:
    DNS:demo1.do.tcbtech-corp.com, DNS:demo2.do.tcbtech-corp.com
X509v3 Key Usage: critical
    Digital Signature, Key Encipherment
X509v3 Extended Key Usage:
    TLS Web Server Authentication, TLS Web Client Authentication
X509v3 Basic Constraints: critical
    CA:FALSE
X509v3 Subject Key Identifier:
    25:DD:9F:3F:0D:C4:65:EA:5F:FF:5D:69:E7:F6:75:03:46:B3:C2:7F
X509v3 Authority Key Identifier:
    keyid:14:2E:B3:17:B7:58:56:CB:AE:50:09:40:E6:1F:AF:9D:8B:14:C2:C6
Authority Information Access:
    OCSP - URI:http://r3.o.lencr.org
    CA Issuers - URI:http://r3.i.lencr.org/
X509v3 Certificate Policies:
    Policy: 2.23.140.1.2.1
    Policy: 1.3.6.1.4.1.44947.1.1.1
      CPS: http://cps.letsencrypt.org
```

https://gitlab.com/markphahn/practical-tls-advice

Technical Details about TLS



A10 Networks Blog, Babur Khan, August 3, 2020
https://www.a10networks.com/blog/key-differences-between-tls-1-2-and-tls-1-3/

https://gitlab.com/markphahn/practical-tls-advice

Cipher suites in TLS 1.3.

- TLS_AES_128_GCM_SHA256
- TLS_AES_256_GCM_SHA384
- TLS_CHACHA20_POLY1305_SHA256
- TLS_AES_128_CCM_SHA256
- TLS_AES_128_CCM_8_SHA256

versus 36 choices in TLS 1.2

- https://go.dev/blog/tls-cipher-suites

Part 2: Trust on the Web

https://gitlab.com/markphahn/practical-tls-advice

Trust on the web

- Certificate Transparency
- Use short lived certificates (90 days)
- Use HSTS
- Use cookie management
- Use separate domains

https://gitlab.com/markphahn/practical-tls-advice

How to get certificates into applications

- Idea 1: Load Balancer in front. Rely on cloud provider
- Idea 2: ACME Certificate

https://gitlab.com/markphahn/practical-tls-advice

Load Balancer profiles to use

- Use the Mozilla Configurator
- AWS: ELBSecurityPolicy-FS-1-2-Res-2019-08
- GCP: MODERN profile.
- Azure: AppGwSslPolicy20170401S

Use separate domains: URL Organization

- External
  - `www.example.com/`
  - `www.example.com/api1`
  - `www.example.com/api2`
- Corporate
  - `example-corp.com/app1`
  - `example-corp.com/app1/api1`
  - `example-corp.com/app2`
  - `example-corp.com/app2/api1`

Use separate domains:
More mature URL Organization

- Marketing
  - `www.example.com/`
- External
  - `app.example.com/`
  - `login.app.example.com/`
  - `api.app.example.com/api1`
  - `api2.app.example.com/`
- Corporate
  - `sso.example-corp.com`
  - `app1.example-corp.com`
  - `app2.example-corp.com/app2`
  - `example-corp.com/app2/api1`

https://gitlab.com/markphahn/practical-tls-advice

Revocation of trust

- Certificate Transparency
- CRLs - Certificate Revocation Lists
- OCSP - Online Certificate Status Protocol

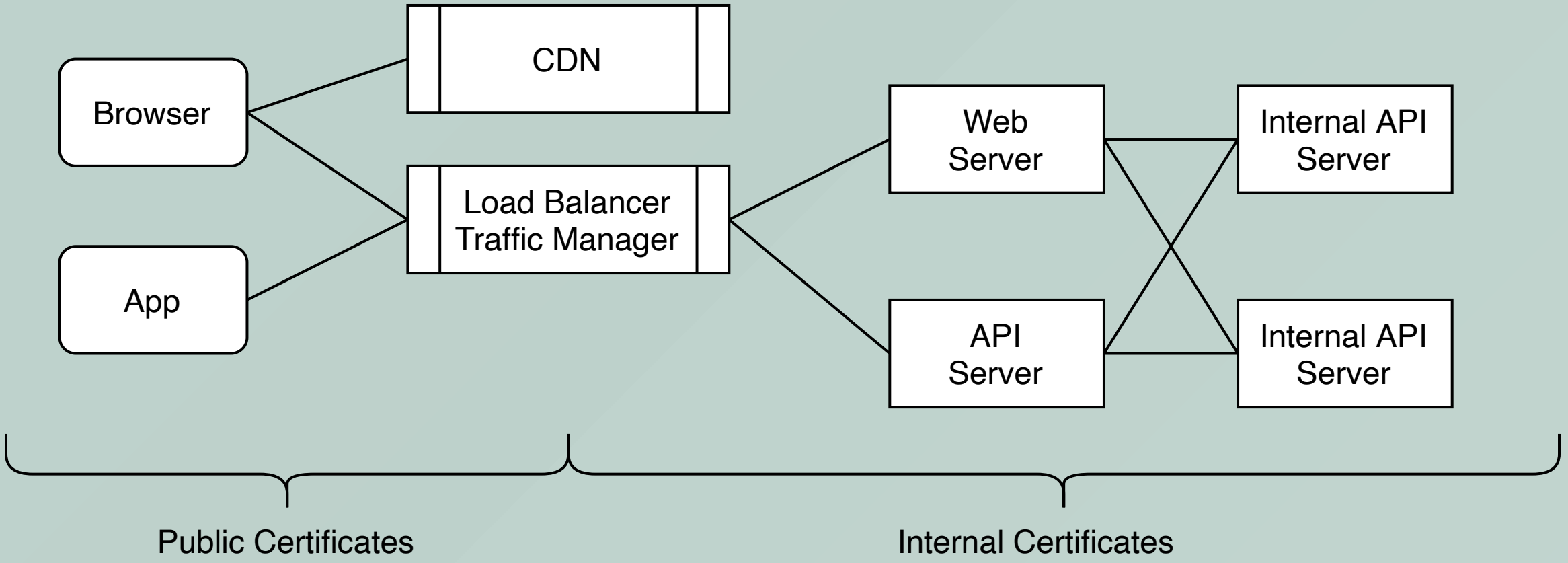
Part 2: in summary

- What is a **Trust Domain**
  - Certificates, TLS, OCSP, et. al. are the building blocks of trust domains
  - A CA
  - Intermediates
    - https://letsencrypt.org/certificates/
  - Claims

Part 3: Local Trust Domains

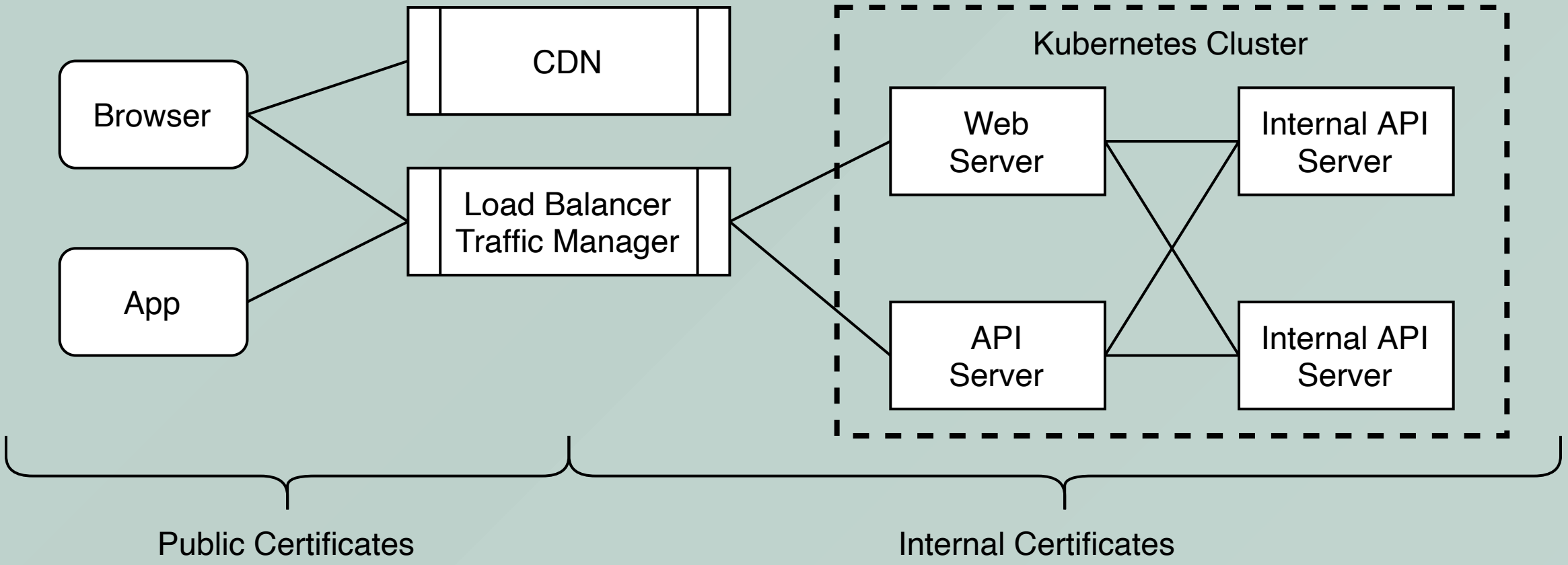
https://gitlab.com/markphahn/practical-tls-advice

Zones of Trust



https://gitlab.com/markphahn/practical-tls-advice

Zones of Trust



https://gitlab.com/markphahn/practical-tls-advice

How to run a local CA

- Cert Manager
- Let's Encrypt Boulder
- Cloudflare CFSSL
- Istio
- Vault
- Cloud HSM tools

https://gitlab.com/markphahn/practical-tls-advice

Trust Model for your Trust Domain

Model : SNI and trust your Private CA cert

- This only applies to infrastructure applications
- Business users use public certificates on the `corp` domain
- Create a CA bundle with your private CA and inject it into your infrastructure applications
- Use short certificate lifetimes
- Don't add your private CA Cert to your public roots of trust
- Add Revocation and Transparency later

https://gitlab.com/markphahn/practical-tls-advice

How to run a local CA

- Cert Manager
  - Run as an internal CA
  - Run as front end to ACME, e.g. Let’s Encrypt
- Let's Encrypt Boulder
  - Boulder is the software that runs Let's Encrypt.
  - Supports everything Let's Encrypt does

How to run a local CA

- Vault
  - Vault is a popular choice from Hashicorp
  - Vault can be used as a backend to Certificate Manager
- Cloudflare CFSSL
  - CFSSL implements a signing server, allowing you to build your own CA
  - CFSSL it maintains a certificate transparency log

How to run a local CA

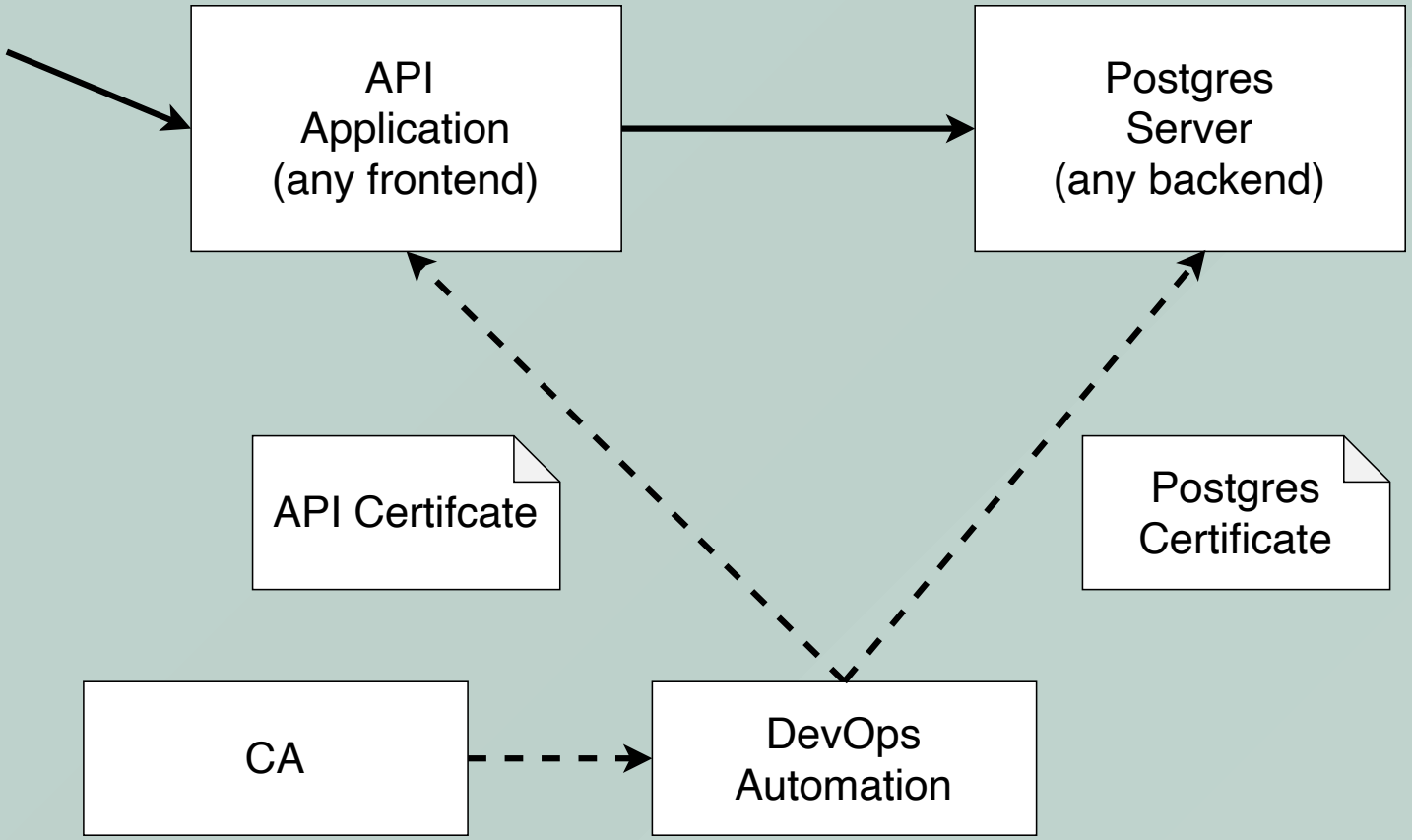
- Istio
  - Istio runs a CA Issuer, simplifying deployment
  - It also provides proxies for your traffic
- Cloud HSM tools
  - Run on dedicated hardware

How to get certificates into applications, *reprise*

- Put your Private Key, Certificate, and CA Bundle in a single clear folder
- Generate certificates that include both server and client authentication
- Use Kubernetes Secrets to provide certificates and and private keys to applications
- Use automation to push certificates to secure locations on your legacy style infrastructure

See `examples` folder in https://gitlab.com/gauntletwizard_net/kubetls

Example



https://gitlab.com/markphahn/practical-tls-advice

Monitor your certificates.

```
# TYPE probe_ssl_earliest_cert_expiry gauge
probe_ssl_earliest_cert_expiry 1.637018287e+09
# HELP probe_ssl_last_chain_expiry_timestamp_seconds Returns last SSL chain expiry in timestamp seconds
# TYPE probe_ssl_last_chain_expiry_timestamp_seconds gauge
probe_ssl_last_chain_expiry_timestamp_seconds 1.637018287e+09
```

https://gitlab.com/markphahn/practical-tls-advice

Remote trust domains

- Communicate with Public Endpoints
- Create a Trust domain for your external connections

https://gitlab.com/markphahn/practical-tls-advice

Thank you

- Ted Hahn, TCB Technologies, Inc.
  - THahn@TCBTech.com
- Mark Hahn, Ciber Global
  - mhahn@ciber.com

- https://gitlab.com/markphahn/practical-tls-advice

Appendix: Relevant RFCs:

- TLS 1.3 - RFC 8446
- TLS 1.2 - RFC 5246
- x509v3 - RFC 5280
  - https://en.wikipedia.org/wiki/X.509
- OCSP - RFC 2560

Vendors also have their own sets of TLS advice

- Google
- Mozilla
- Apple
- Microsoft has many technical documents.

https://gitlab.com/markphahn/practical-tls-advice