


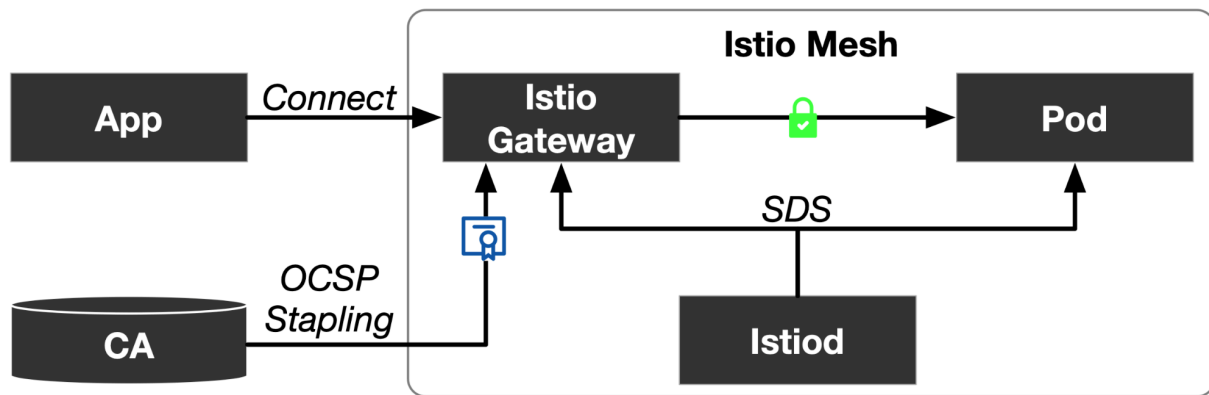
OCSP Stapling for Istio Gateways

Shared with Istio Community

	
Owner: Sebastien Michel (@sebltm) Łukasz Buczkowski (@lukbuc) Working Group:	Status: WIP In Review Approved Obsolete Created: 2023-01-16 Approvers: WG-lead-X [], WG-lead-Y [], ...

TL;DR

Istio currently reads external certificates (non-internal Istio certificates used for mTLS, but rather certificates used to secure external connections through Istio Ingress Gateways) but always relies on the client for validating the certificate's expiration status. This document proposes gathering OCSP staples from an external third-party provider and distributing them to Envoy using SDS.



Overview

Istio uses SDS to distribute both internal certificates (used for mTLS within the mesh) as well as external certificates (used to secure a downstream connection to an upstream, in-cluster service). This design document will only focus on the latter case.

Outside of a certificate's natural life (i.e the certificate's TTL), there are two main ways a Certificate Authority (CA) can take a non-expired certificate out of service and make it no longer valid: Certificate Revocation Lists (CRLs) and Online Certificate Status Protocol (OCSP):

Shared with Istio Community

- CRLs are lists of certificates maintained by the CAs, accessible through a CRL Distribution Point (CDP). The clients are meant to request this list from the CA during the TLS handshake and verify the absence of the certificate on the list. If the certificate's serial number appears on the list, it has been revoked and is deemed no longer valid. However in practice this adds another round trip to the handshake and isn't enabled by default in most browsers for example.
- OCSP is a protocol used to check the revocation status of individual certificates. The principle is similar to the above, but applied to a single certificate rather than a list: a client, upon receiving the certificate, communicates with the CA to check the revocation status. The response can be one of "good", "revoked" or "unknown". The challenge with OCSP is that it puts a lot of load on the CA as every client needs to get the certificate verified independently.

There is a solution to these problems in the form of OCSP Stapling: instead of each client having to request the revocation status of the certificate, the server (which owns the certificate and serves the content) requests the certificate revocation status of its own certificate in the form of a signed, time-stamped add-on. The server then presents both the certificate and this stapled-on response.

Envoy supports these OCSP Staples, and are configured in two places:

- The [DownstreamTlsContext](#) where the OCSP Staple policy is configured, one of either:
 - LENIENT_STAPLING: OCSP responses are optional
 - STRICT_STAPLING: OCSP responses are optional, but used if present and valid
 - MUST_STAPLE: OCSP responses are required
- The [TlsCertificate](#) where the OCSP Staple itself is configured either as inline bytes or as a file path

Istio does not yet support OCSP Stapling, I propose therefore using a third-party to fetch staples, and use Istio to distribute the certificate, key and OCSP staple to Envoy.

Use Guide

The user is expected to provide a pre-fetched OCSP staple in the same Secret as the TLS Certificate and Key.

```
kind: Secret
metadata:
  name: server-crt
data:
  tls.crt: RndvR1pYSXZZWGR6RUZvYURPSEt2QWt1c2wxSOMEBYTES
  tls.key: en1hd3F1VGJhd0xZ0THERBYTES
```

```
tls.ocsp: MjAyMy0wMi0wMSAxNjo1NOCSPBYTES
```

Additionally, to enforce an Envoy stapling policy by using an annotation:

```
kubernetes.io/ocsp-policy: < OPTIONAL | PREFERRED | MANDATORY >
```

Design

With the Secret as described, Istio will configure all the listeners that use the certificate with the matching OCSP policy. The most lenient is `LENIENT_STAPLING` which will use an OCSP Staple if present and valid, but won't warn or error if there is none. This is represented by the "OPTIONAL" mode in the annotation.

The next level of constraint is "PREFERRED" in which the TLS handshake will be completed without a OCSP Staple if there is none, but if one is present then it will be used and if invalid or expired an error will be raised and the Certificate will be rejected.

The final level of constraint is "MANDATORY" in which a valid OCSP Staple must be present or the Envoy configuration will fail.

Test Plan

- Pushing an OCSP Staple and the stapling policy to Envoy
- Pushing a certificate with the Mandatory policy to Envoy (ensure the configuration fails)
- Pushing a certificate with an invalid OCSP staple and the PREFERRED policy (ensure the configuration succeeds but the handshake fails)
- Pushing a certificate with no OCSP staple and the PREFERRED policy (ensure the configuration and handshake succeed)
- Pushing a certificate with no OCSP staple and the OPTIONAL policy (ensure the configuration and handshake succeed)
- Pushing a certificate with an invalid OCSP staple and the OPTIONAL policy (ensure the configuration and handshake succeed)
- Pushing a certificate with a valid OCSP staple and the OPTIONAL policy (ensure the configuration and handshake succeed and the staple is part of the handshake)