
Signing Eiffel events

Magnus Bäck, Axis Communications

Agenda

Quick Eiffel event signing primer

Problem statement

Possible solutions

Why should we sign Eiffel events?

Structure of a signed event

```
{
  "meta": {
    "security": {
      "authorIdentity": "CN=joe,DC=example,DC=com",
      "integrityProtection": {
        "alg": "ES256",
        "signature": "b25zIDI0IGFwciAyMDI0IDIxOjQ0OjM0IENFU1..."
      }
    }
  },
  "data": { ... },
  "links": [ ... ]
}
```

Canonical JSON

Reliable signing and verification requires that everyone agrees on how data structures should be serialized into JSON. This is defined in RFC 8785, *JSON Canonicalization Scheme* (JCS).

```
{  
  "foo": "bar\u0000A"  
}
```



```
{"foo": "bar\n"}
```

Signing how-to

```
event.meta.security.authorIdentity = "CN=joe,DC=example,DC=com"
event.meta.security.integrityProtection.alg = "ES256"
event.meta.security.integrityProtection.signature = ""
event.meta.security.integrityProtection.signature =
    sign(private_key, sha256(canonical_json(event)))

sig = event.meta.security.integrityProtection.signature
event.meta.security.integrityProtection.signature = ""
verify(public_key, sig, sha256(canonical_json(event)))
```

Ecosystem support

.NET SDK (sort of)

Go SDK

eiffel-broadcaster Jenkins plugin

**But which public key should we use
to verify an event's signature?**

Use existing `meta.security.integrityProtection.publicKey`

Bundling the public key in the event itself protects against corruption, but doesn't authenticate the sender.

If new publishers could send a “bootstrap” event with the key they intend to use this could be picked up and saved for later.

Introduce meta.security field for certificate

If the event includes a certificate signed by a CA accepted by the consumer, this will be enough to authenticate the event.

The certificate's subject would be required to match meta.security.authorIdentity.

Configuration in each consumer

Each consumer would be configured with a list of (event sender DN, public key) mappings.

Shared configuration

There's a common configuration source where all consumers can look up event sender DNs and get public keys.

- One or more files in a git repository.
- A web service or similar API.

If possible, use a certificate here to allow consumers to verify that the public key comes from a certificate issued to the expected subject.

Questions?

Use the Sigstore toolbox

