Identity, Security, etc. API Issues

Eric Rescorla

`ekr@rtfm.com`

Overview of Topics

- DTLS
  - Controlling my own DTLS key
  - Examining remote DTLS parameters
- Identity
  - Examining my own identity

DTLS Key Control Requirements

- Keys are scoped to origin
- Be able to use the same key repeatedly
  - Avoid repeatedly generating keys
  - Enable key continuity/auditing
- Be able to use multiple distinct keys
- Be able to generate a temporary key

- Application needs to be able to control this

DtlsIdentity Constraint

```
{
  mandatory : [
   {
     DtlsIdentity : "ekr@example.com"
   }
  ]
}
```

- DTLS Keys are stored under `DtlsIdentity` value D
- If no key exists with name D it is made and stored
- If key exists with name D that key is reused
- "falsy" (`false`, `null`, ...) `DtlsIdentity` values never match anything
  - ... this means make a fresh key pair for this call

Alternative Design: use WebCrypto

- JS creates a WebCrypto key
  - `pc.setDtlsKey()` API call to impose the key
  - JS is responsible for figuring out what keys to use
- Problem: private key needs to be unavailable to JS
  - Otherwise Identity isn't secure
- WebCrypto keys can be marked unexportable
  - But this doesn't mean an unexportable key was never known
- This is going to need a bunch of WebCrypto bookkeeping that doesn't exist yet
  - Has this private key *ever* been available to the JS

What about the other side's public key

- Would be nice to know the other side's public key
  - For key continuity
- We Justin, Martin, EKR went back and forth on this
  - And decided that less is more
- Proposal: a binary version of the other side's keys

New API

- `pc.remoteCertificates` contains a list of other side's certificates
  - As base64-encoded (?) blobs
- The raw certificate can just be used as a lookup key
  - ... or parsed with WebCrypto
- No claims about the browser's opinion of the certificates

Recap: remote identity

- Remote identity is directly observable

```
dictionary RTCIdentityAssertion {
    DOMString idp;
    DOMString name;
};
```

- Stored as `pc.peerIdentity`

What about my own identity?

- Would be nice to be able to observe this
- We have `pc.onidentityresult` to notify when assertion obtained
  - It doesn't have a defined argument ("TODO")

Proposal

- `onidentityresult` takes a `RTCIdentity` argument corresponding to the obtained identity
- Rename `peerIdentity` to `remoteIdentity` to match `remoteDescription`
- `localIdentity` contains my own identity (can be `null`)