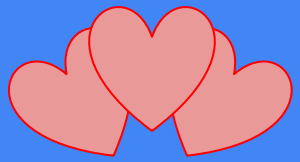
Credential Management API

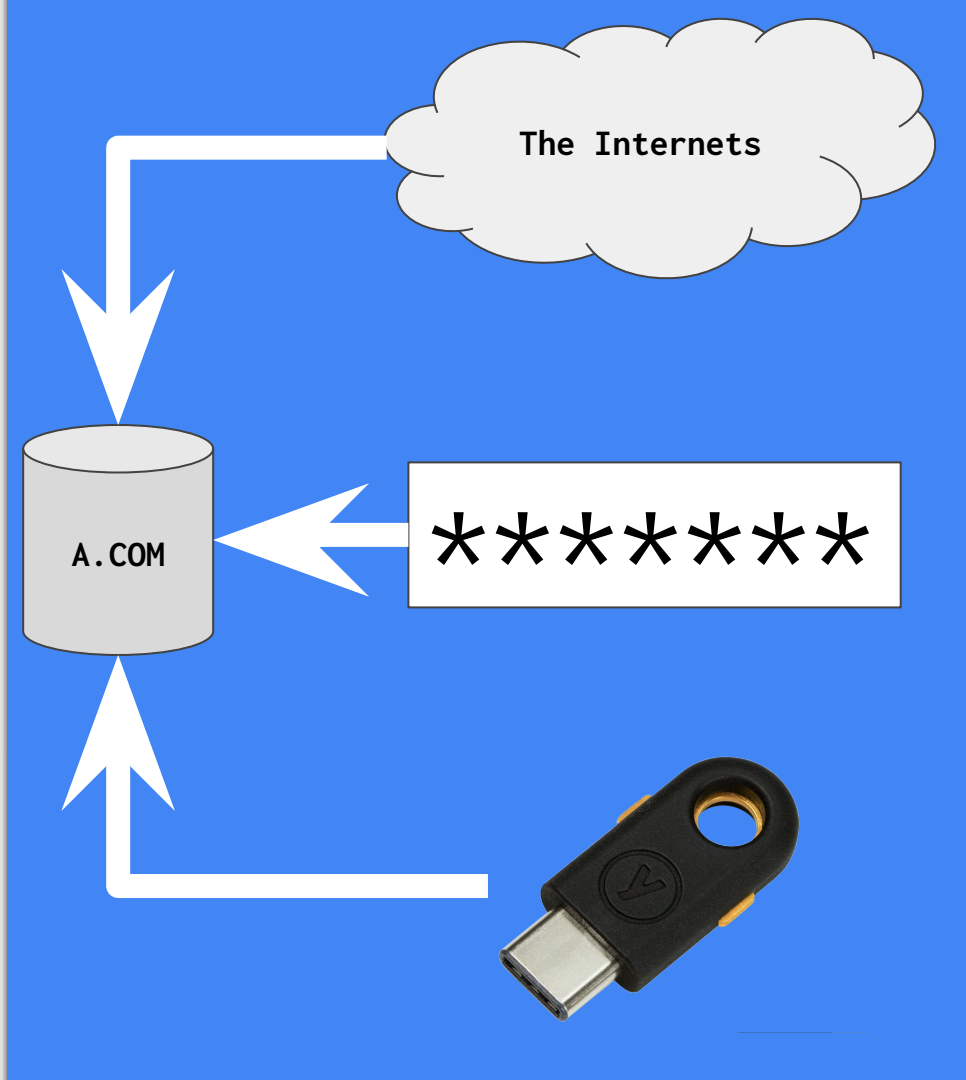


WebAuthn API

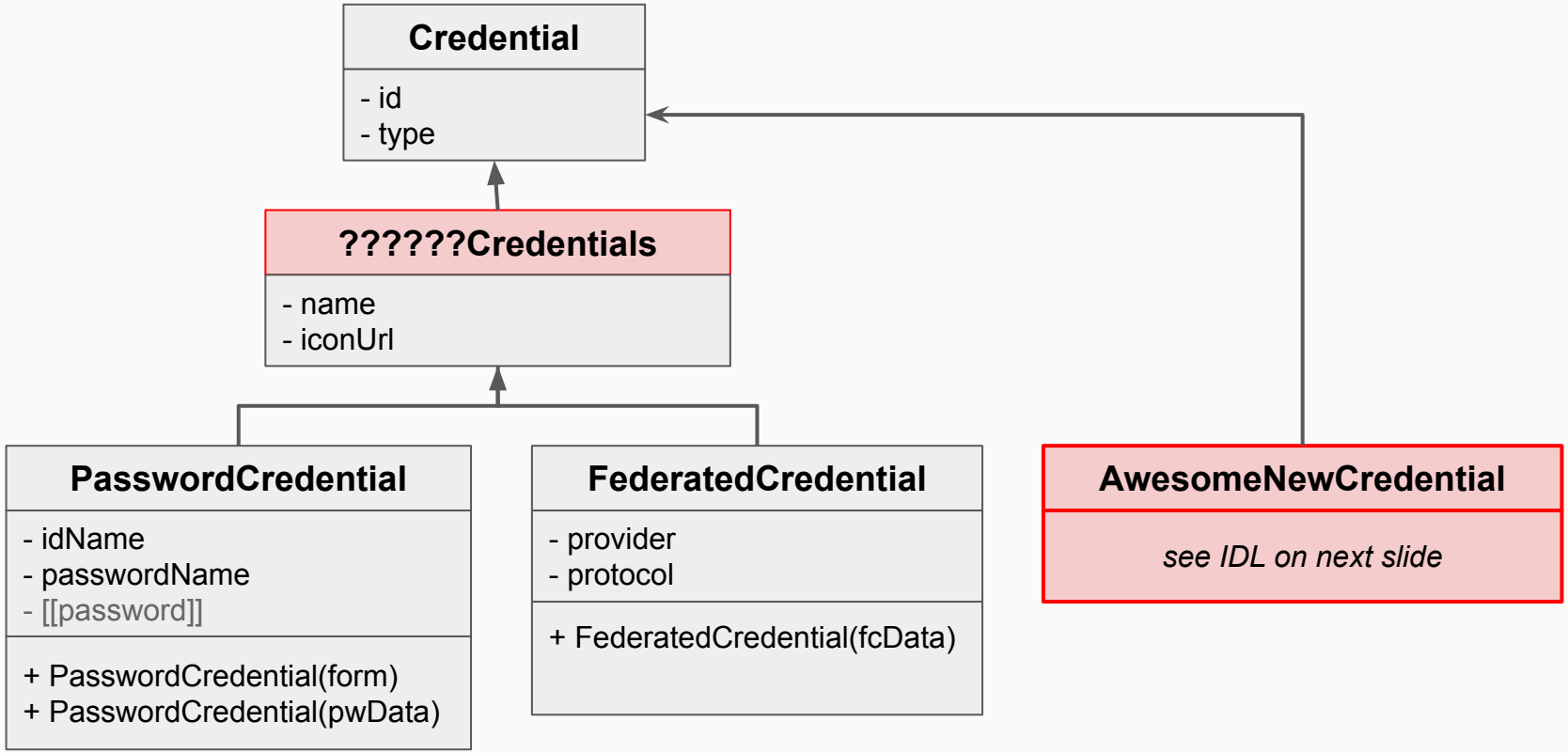
Slides at https://goo.gl/TszgD2

TL;DR:

Things are more similar than dissimilar, and it's worth our time to reflect that in the API.



Operation "Save Our Developers' Sanity": Option C`



Operation "Save Our Developers' Sanity": Option C`

```
interface AwesomeNewCredential : Credential {
  readonly attribute AwesomeNewAssertion assertion;
  static Promise<AwesomeNewCredential> register(AwesomeNewCredentialData data);
};

interface AwesomeNewCredentialAssertion {
 readonly attribute ArrayBuffer clientDataJSON;
 readonly attribute ArrayBuffer attestationObject;
 readonly attribute ArrayBuffer? signature;
};
```

(Spelled out in more detail at https://goo.gl/8Cmcrm)

Option C`: Registration

```
var c = await AwesomeNewCredential.register({
    rpDisplayName: "Acme",
    displayName: "John P. Smith",
    algorithms: [ "ES256", "RS256" ],
    challenge: "Y2xpbWIgYSBtb3VudGFpbg",
    options: {
      ...,
      extensions: { ... }
    }
});
// Send |c.assertion| to server to bind to an account.
```

Operation "Save Our Developers' Sanity": Option C`

```
dictionary AwesomeNewCredentialRequestOptions : CredentialRequestOptions {
  BufferSource challenge;
  unsigned long timeoutSeconds;
  USVString rpId;
  sequence <AwesomeNewCredentialDescriptor> allowList;
  WebAuthnExtensions extensions;
};
```

Option C`: Authentication

```
var c = await navigator.credentials.get({
    AwesomeNew: {
      challenge: "SGFuIFNvbG8gc2hvdCBmaXJzdC4",
      timeoutSeconds: 300,
      ...,
      extensions: {
       "webauthn.txauth.simple": "Wave your hands in the air like you just don’t care"
      }
    }
});
// Send |c.assertion| to server for verification.
```

Option C`: Whither `store()`?

```
var c = await AwesomeNewCredential.register({ ... });
// Send |c.assertion| to server to bind to an account. Then:

navigator.credentials.store(c);
// This could give the user agent the ability to choose the appropriate
// authenticator for a given origin, as it would understand the
// relationship between a given ID and a given origin.
//
// This would pave the way for a browser-driven multi-step `getAll()` in
// future iterations on the API. "Oh, this account on this origin uses this
// key. I'll prompt the user accordingly and return both a
// PasswordCredential and AwesomeNewCredential! Yay, me!"
```