WebAuthn Network Transport

Increasing WebAuthn adoption, one `transport` at a time.

Duo Security is now part of Cisco.

The WebAuthn spec currently supports four transports

```
[usb, nfc, ble, internal,
lightning, cable, (https?)]
```

The “dream scenario”

- Nobody thinks too hard about picking a specific authenticator
- People use whatever hardware is available to them in a secure way

Google’s approach: CaBLE

- CaBLE was submitted to both the WebAuthn and CTAP2 specs
  - V1 required RP participation, V2 is purely between client-platform and authenticator
- In the V2 CaBLE PR (to CTAP2), a cloud service can facilitate binding by pre-sharing a key between the client-platform and authenticator
  - Otherwise, an out-of-band method (QR code) is used to share a secret before binding
- Once a binding is established, a BLE channel is used to communicate with the authenticator

We believe the CaBLE transport **can be extended** beyond BLE.

Why is a network transport desirable?

- Low common denominator for compatibility
- Allows for novel solutions without being prescriptive
- Makes the ecosystem **safer**

```
Object.assign(navigator.credentials, hybridCredentials);
```

Why is a network transport desirable?

- Low common denominator for compatibility
- Allows for novel solutions without being prescriptive
- Makes the ecosystem **safer**

```
Object.assign(navigator.credentials, hybridCredentials);
```

Real problem,
unsafe solution.

nick: I am underwhelmed with some of the FIDO account recovery stuff. having some guidelines in the spec would be great.

... would like to have a better story around roaming mobile authenticators.

... want to consider multiple transports.

nick: we are thinking about a network based transport

agl: we are thinking about this, but not at liberty to say how we view it.

nick: what is main concern.

agl: it is the guarantee the authenticator near machine, but if we run that over network that is a different sort of thing.

nick: I don't think bluetooth proximity comes with unphishability. If there is way we can maintain unphisable properties over transport, we will explore that.

... I think people will opt for usability.

agl: if the exosystem degrades in that way, we have to ponder on that.

... but over network it is not webauthn at that point,

nick: i agree

... but want to look at main authentication being delegated.

“If the ecosystem degrades in that way, we have to ponder on that.”

- **@agl**

“Let’s degrade the ecosystem.”

- **Duo Labs**, probably

We should get out in front of this now, before the ecosystem **degrades** any further.

BLE

- Requires spatial proximity between the authenticator and client-platform
  - A useful property, but not what provides phishing resistance

HTTPS

- Would work in cases where BLE connectivity is not possible
- Loses spatial proximity properties, but can maintain phishing resistance

Phishing Resistance in WebAuthn

Phishing resistance is provided by:

- Lack of shared secrets
  - Credentials are scoped to the Relying Party
  - Ceremonies are not replayable like static passwords
- Origin scoping
  - Client-platform is responsible for preventing homograph-style attacks
- Binding between client-platform and authenticator
  - Currently provided as a consequence of the physically connecting devices or using a platform authenticator
  - Provided in the CaBLE V2 PR by establishing a cryptographic binding between the devices
    - This is also what we've done when prototyping this solution, albeit somewhat differently
    - We should just use the approach introduced by CaBLE, but not limit it to the BLE transport

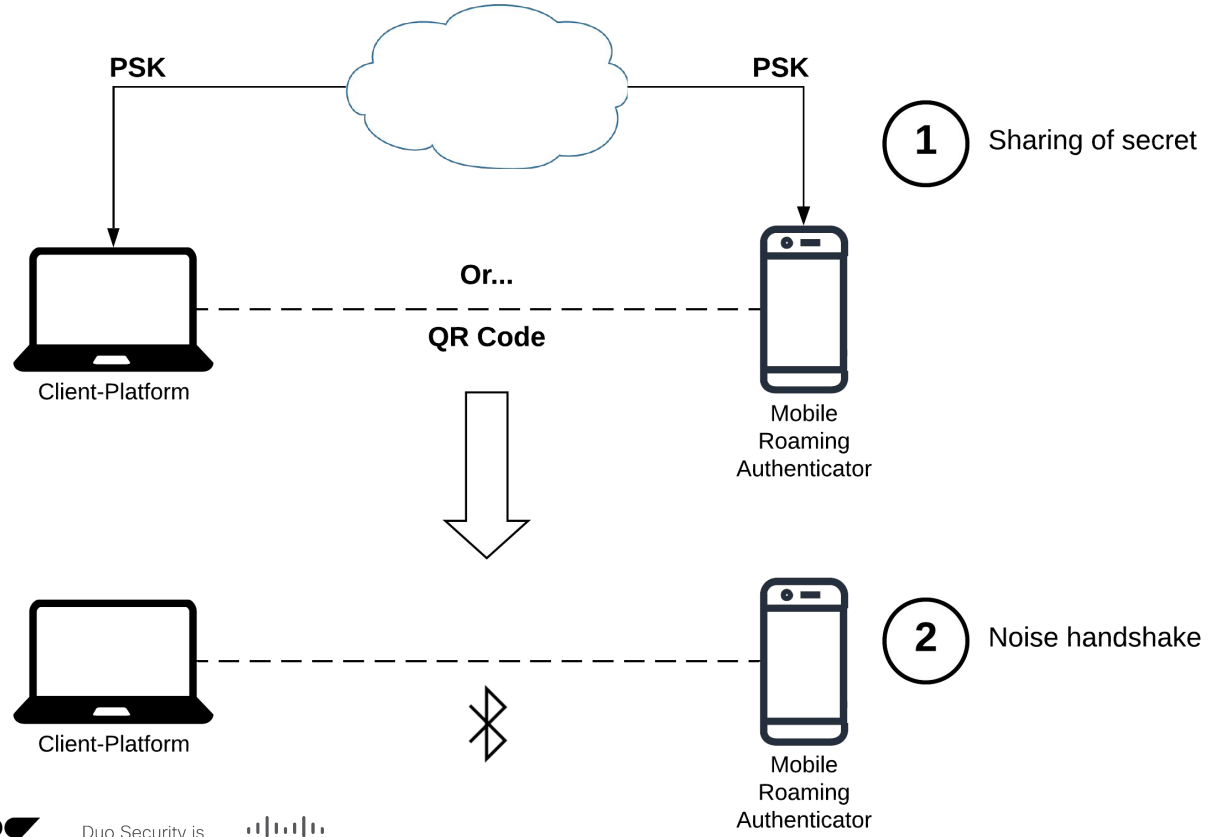
We can preserve **phishing resistance** with a network transport.

Authenticator Proximity != Phishing Resistance

- CaBLE approach
  - Pairing generally serves two main purposes:
    - (1) Establishing encryption credentials for secure discovery and communication, and
    - (2) ensuring that the two devices communicating are the two the user intends
- Can we maintain these properties without spatial proximity?

Yes,  (we think).

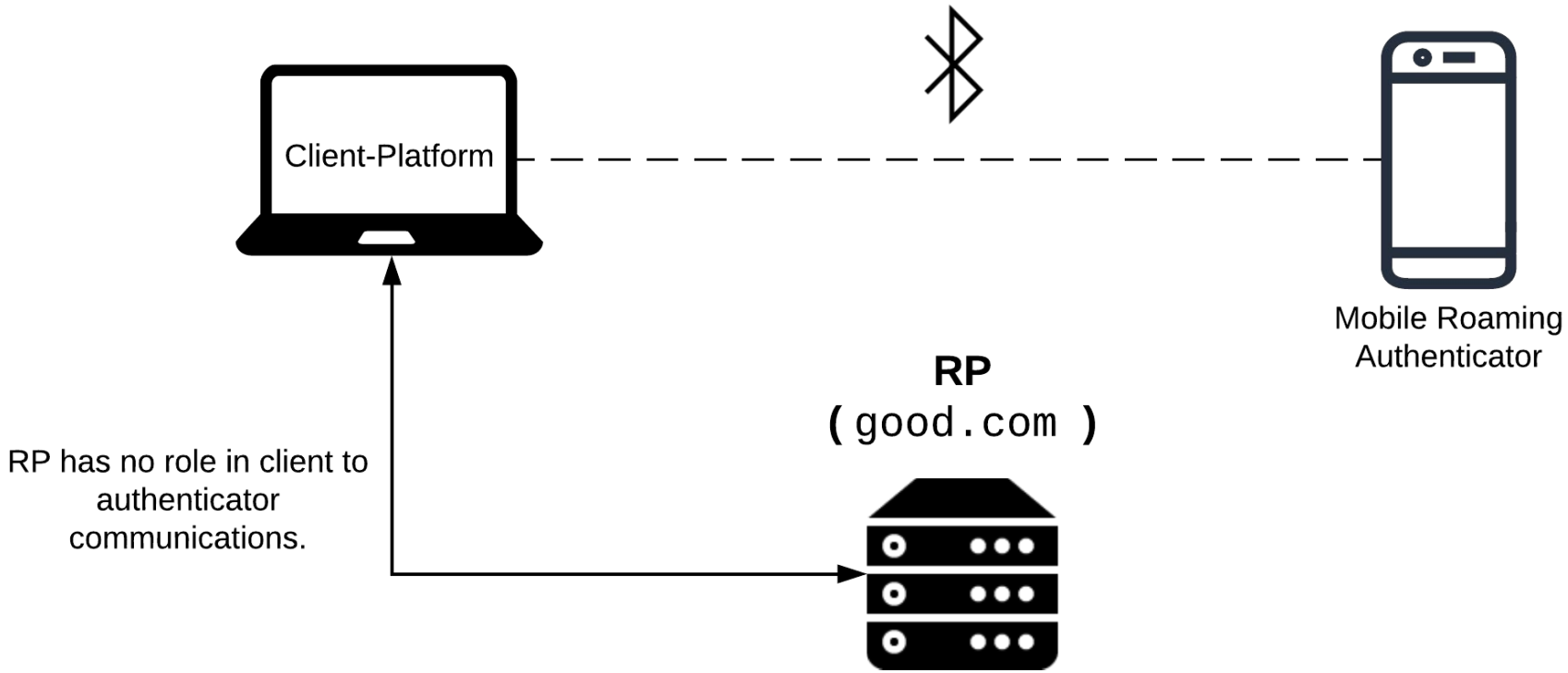
Binding Establishment



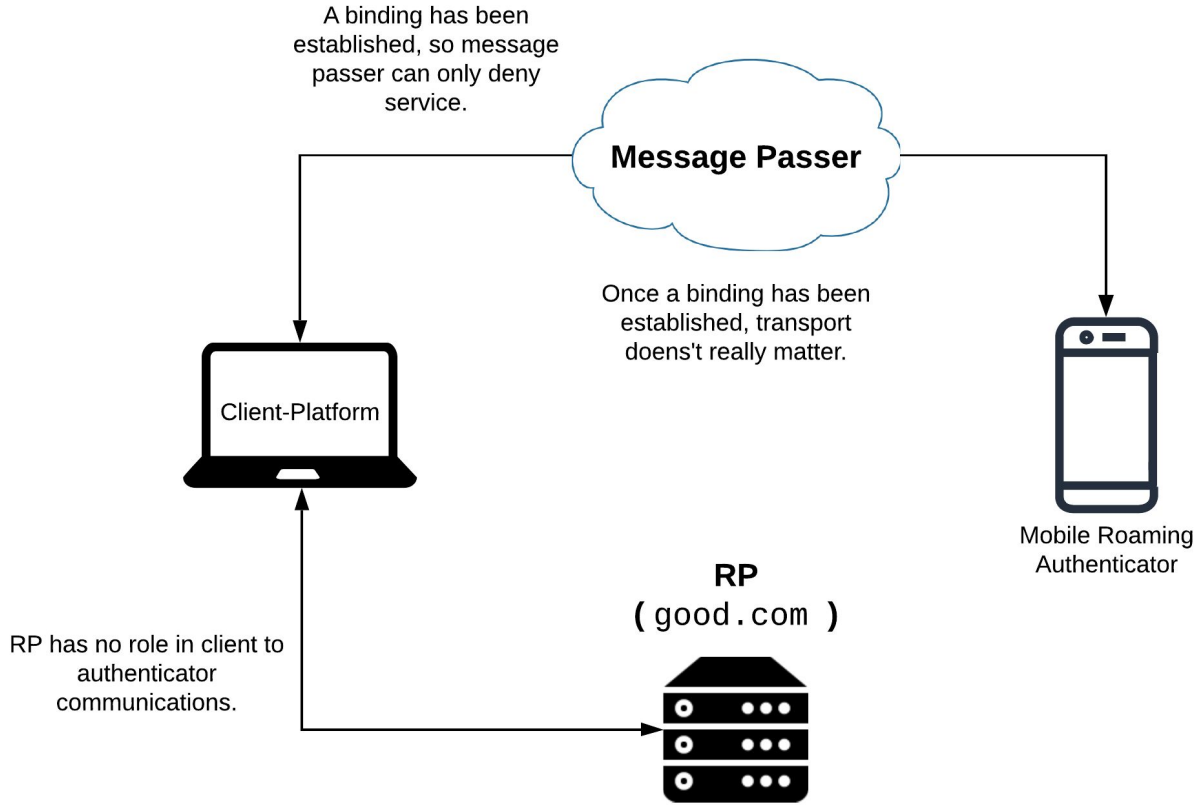
Our Approach to Pairing

- QR Code Contents
  - Address of message relay
  - Public-key of client-platform
  - Other metadata

CTAP2 Interactions (BLE)



CTAP2 Interactions (Network Transport)



DUO LA

Components of a Network Transport

DUO LABS

Duo Security is now part of Cisco.

cisco

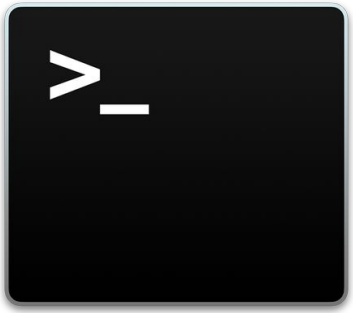
Serialization

- Serialization over-the-wire is one consideration for a network-based WebAuthn transport

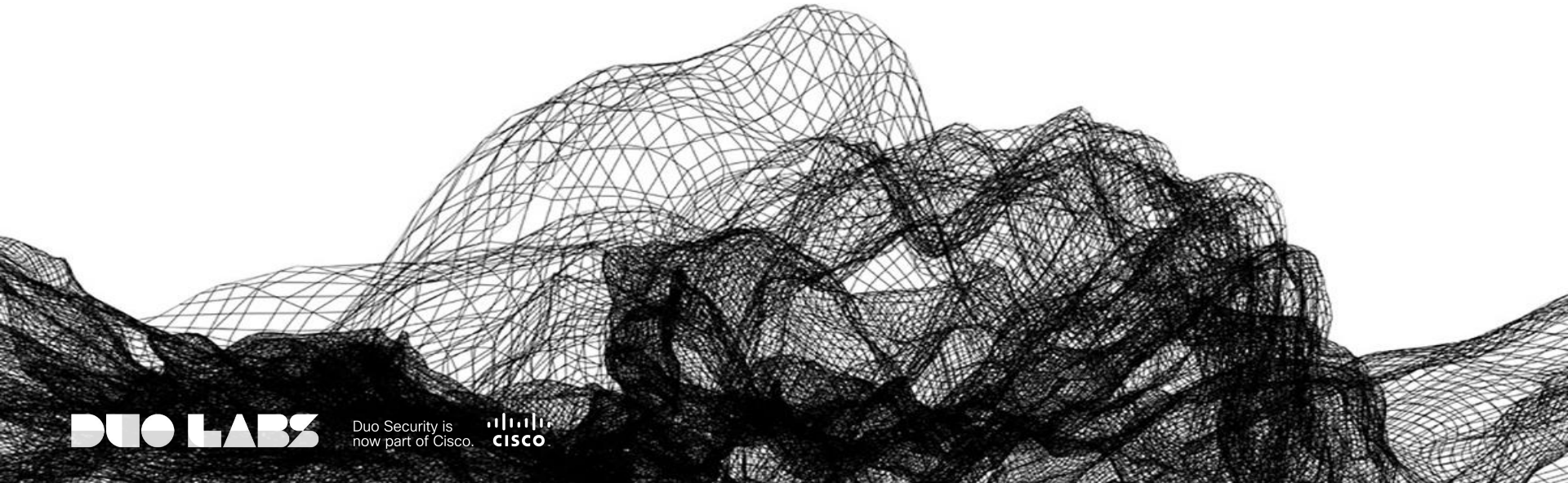
- One approach that we've implemented in our own prototypes, (and have seen others do the same), is
  - Replace all `ArrayBuffers` with URL-safe Base64-encoded strings
  - `JSON`ify the resulting object

Configuration

- Another essential component of a network transport is authenticator config
- Cryptographic binding between authenticator and client is a `MUST`
    - A URI scheme that allows a user to click a link provided by a cloud authenticator service and automatically use it for authentication would be great. Some things that would need to be configured may be:
        - API endpoints for making requests to the authenticator
        - Credentials for authenticating to the remote authenticator
        - A friendly name
        - Enterprise administration of a cloud authenticator



Protocol Options



Proposal 1: WebAuthn JSON via HTTP `POST`

- We've taken this approach in internal prototypes, and are aware of it being adopted at other organizations

- In our prototype, we encrypt the request body before sending, but a simpler version could look like the following

```
Host: auth.credentialprovider.tld
Content-Type: application/json
Authorization: Bearer sOmEsIgNaTuRe

POST /credential/create

{
  "authenticatorExtensions": "",
  "clientDataHash":
"1knXlawRhU7i3Al91QeNG6h+RfkCItKdHutQzUjxILQ=",
  "credTypesAndPubKeyAlgs": [
    [
      "public-key",
      "-7"
    ]
  ],
  "requireResidentKey": false,
  "requireUserPresence": true,
  "requireUserVerification": false,
  "rp": {
    "name": "Acme, Inc",
    "id": "webauthn.io"
  },
  "user": {
    "name": "moons@example.com",
    "displayName": "moons",
    "id": "XzKmn0tpskE8Kpi89plZ1A=="
  }
}
```

Proposal 2: CTAP2 (CBOR) via HTTP `POST`

- The previous approach requires the party on the other end to do some of the work that is typically the responsibility of the client

- It could be valuable to treat the transport as solely a transport and use it just to communicate binary data, *but...*
  - It's harder to debug because CBOR decoding is necessary
  - CBOR was chosen by CTAP to function in bandwidth-constrained environments
    - The overhead of a TLS connection and HTTP headers is enough to effectively negate the savings we would get by using CBOR
  - ...but, it might make sense to use CBOR/CTAP anyway since that's already been specified

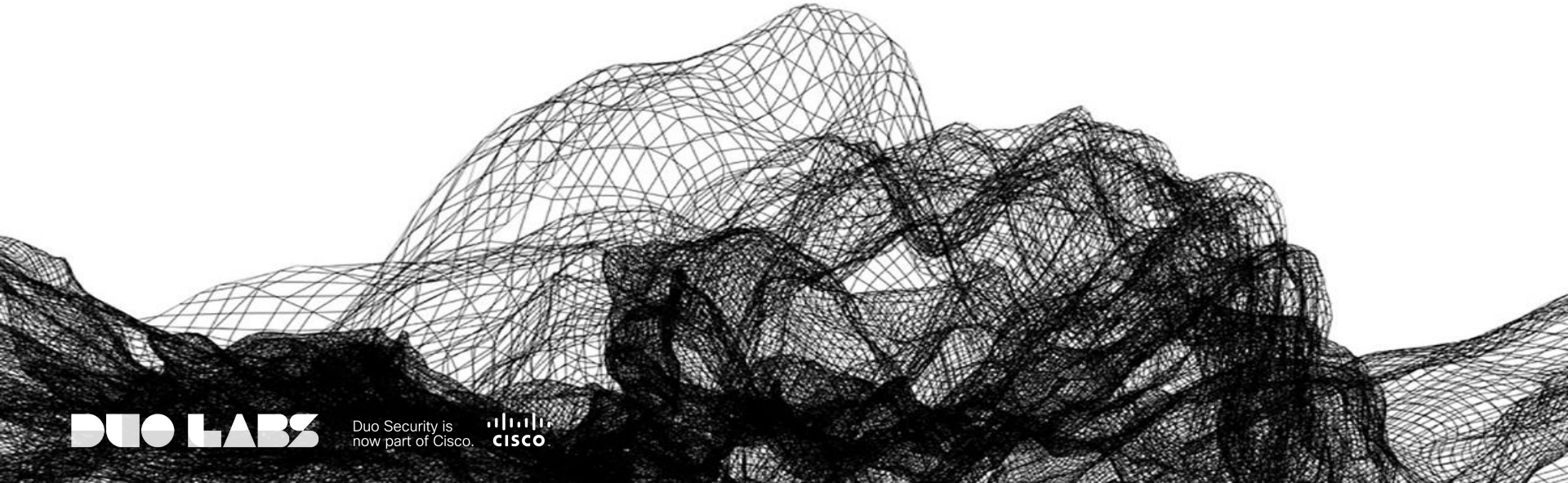
```
Host: auth.credentialprovider.tld
Content-Type: application/json
Authorization: Bearer sOmEsIgNaTuRe

POST /credential/create

WligyzaRcMkDrAVvfr0qeGzAqtOCh8DPpS2pYH6i6
GVQ9N8OK0X/t+XpMqD8FLH4V4DMZr6NZL2QlUVHJ5
YIhRyzZPzE4nzEFAm5b9tyooXm82SpdE/00GuqBkZ
8oWoAM5NzxRq2u200uhR5nRN2c5LjXHnhUtsD2VA7
BEEqWNG6JXilpOrc+UEBTHHaEeyzxIm/vQc5MNNED
BMg+ynFjQCc0Na1qh1xlb3ohU3VzTAnD5bmn1f+FA
8ao5bD63Kpsghr0eHbOovDRhXWAe2D1xP/988kZ7s
123lTN/4793HN50yyerXYFhpaPXap/Yt81tDvlbIL
HML/w3o9a31Od12mF3ssfkIEiMC6Rsn9pLPmBovKG
u1JSEcCpRVCNn7CCbSHYWDVu+TfMdjJKVaAVj0gA+
JrUmkl7WnGXV9S4t9aYbG6KfAbMElg9EfMFmaIw2I
sTqR5t0vJcmmwTz51GeMOLe5F5fwwgqA6BE3eCdkX
3X9d1HJI5TMMseKyThPTHsOuorLcwolBJ7Dj3HJoF
TFRw+x7VCksuS1I52M7R94zWKx0WdCnzeM1OnvqN0
ozAvmLIWxvkcN/+nsy3aBa9LSvADgCmCYEO/yUxH+
IlHUUjeMH2Dwz3ynk9C6g+V6rWt0tkD4w53vAB/mj
51eGf67wdLI6/3LRvBkD62CISIVJv5JitIx37S68W
```

Appendix A: Delegated Authenticator Extensions

- If a network transport is too prescriptive, we could allow browsers to expose APIs that extensions could register themselves with
- Similar to the existing proxy APIs in Chrome and Firefox
- The browser is still able to implement the security checks that are the responsibility of the client, but can delegate communication with the authenticator to another party

Demo

Questions?



DUO LABS

Duo Security is now part of Cisco.

cisco

Thank you!

Twitter: @**wellhydrated**, @**futureimperfect**

Email: nmooney@duo.com, jbarclay@duo.com

Duo Security is now part of Cisco.