Javascript Crypto &
The Case Against Crypto Reductionism

Ben Adida
Mozilla

Workshop on Real-World Crypto
January 2013

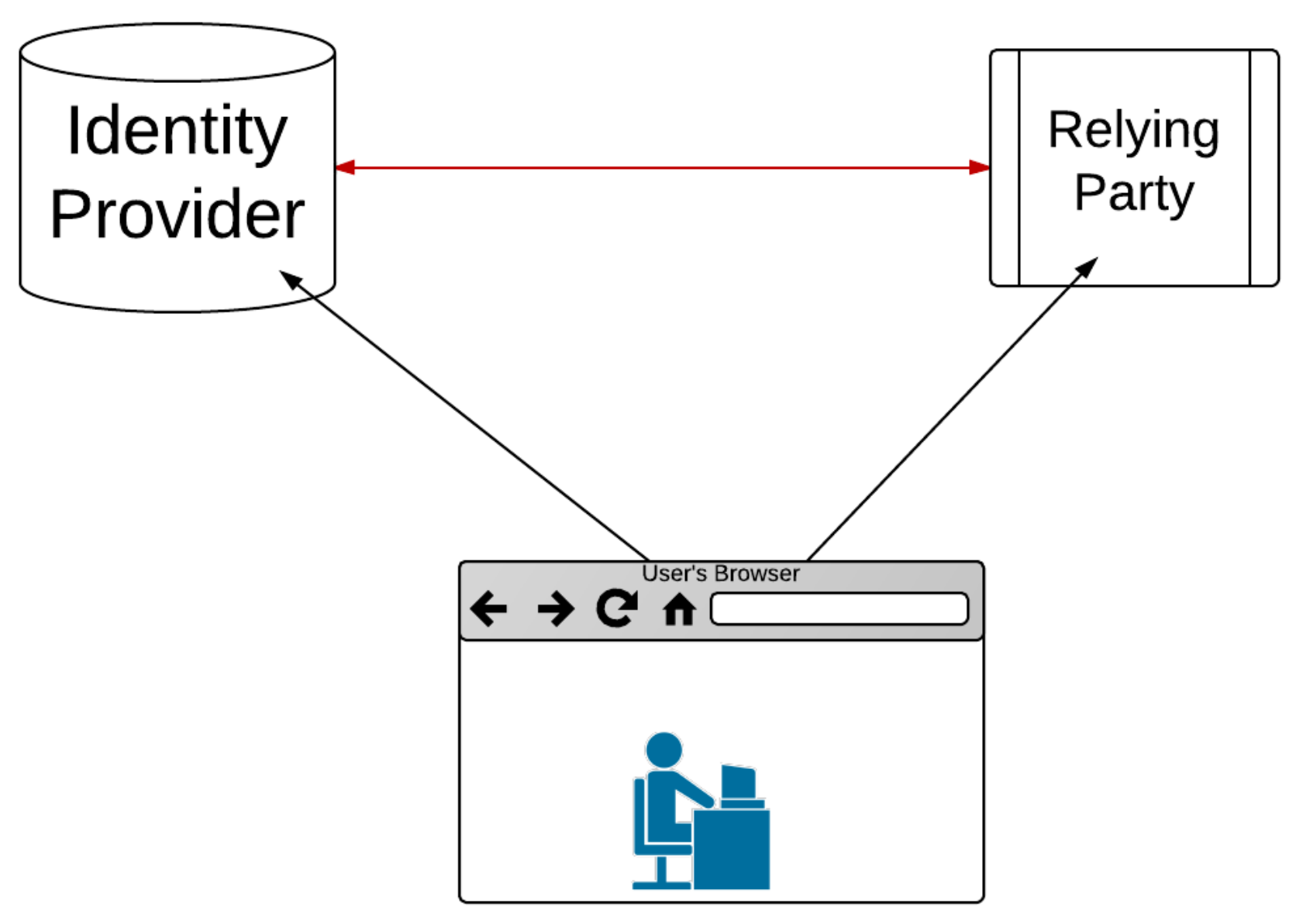
promote openness, innovation & opportunity on the Web.

mozilla Persona

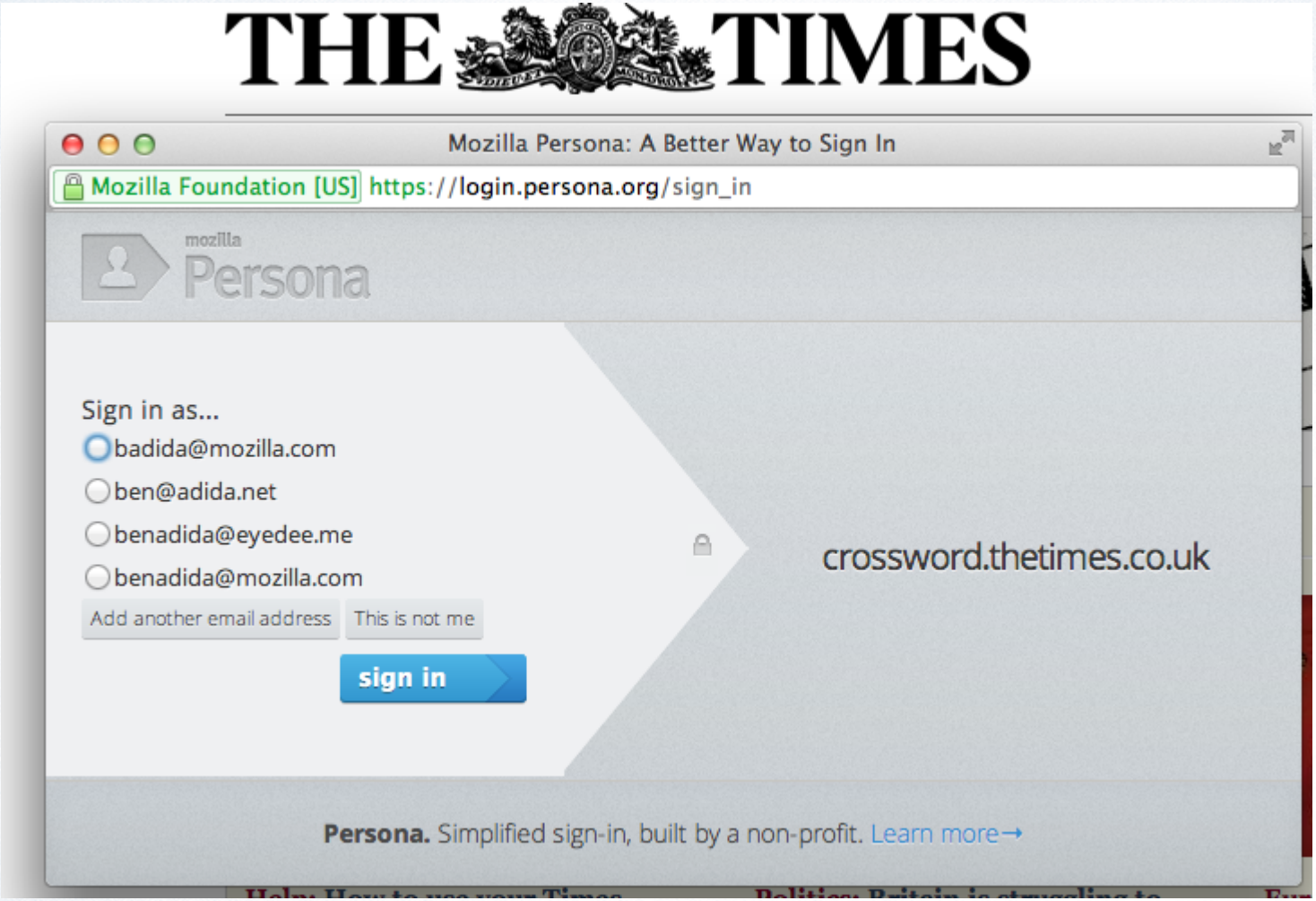
previously

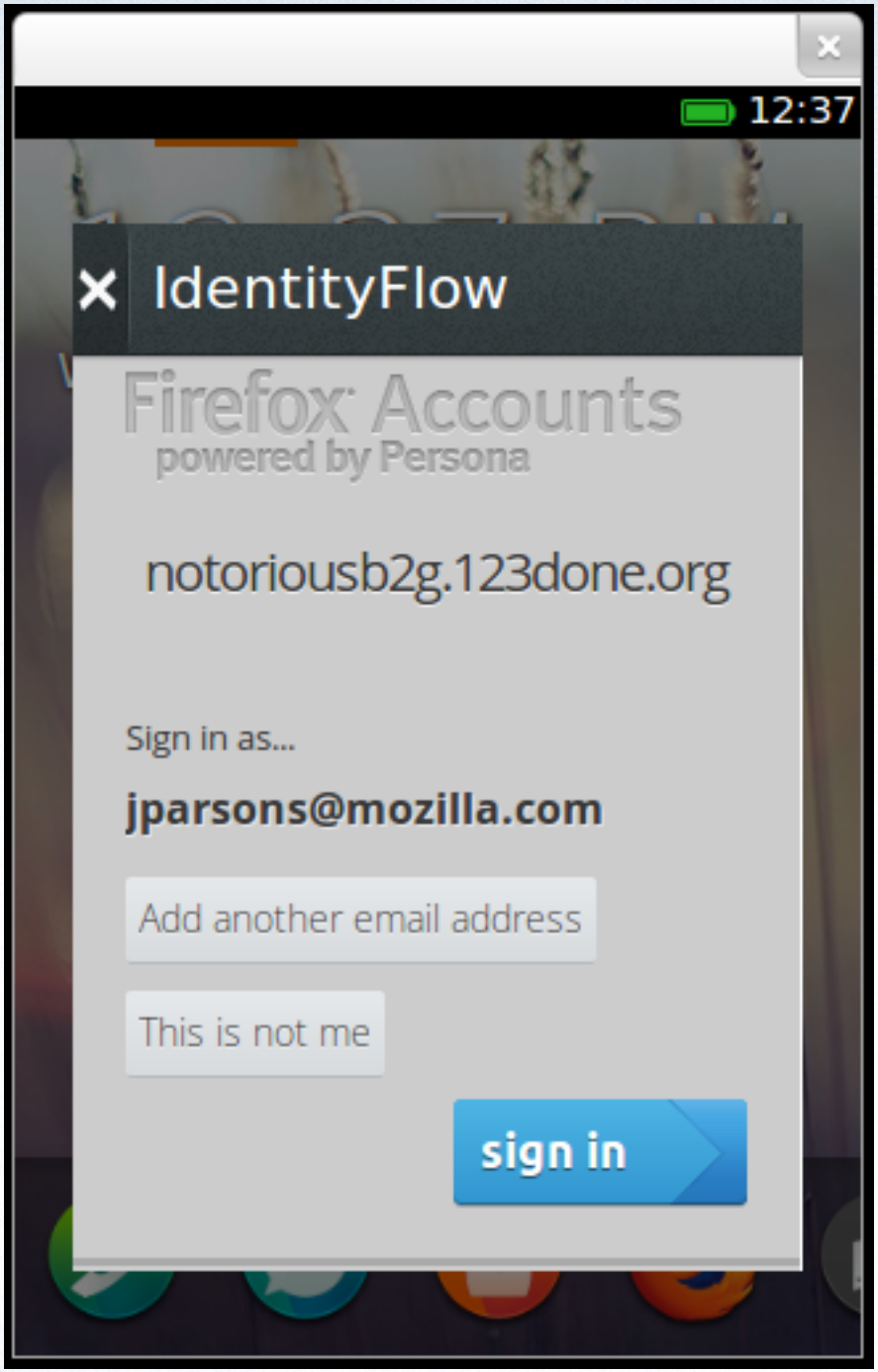
Browser ID

- easy to deploy
- easy to use
- privacy protecting


Identity Provider
Relying Party
User's Browser


Sign in with your Email


THE TIMES
Mozilla Persona: A Better Way to Sign In
Mozilla Foundation [US]
https://login.persona.org/sign_in
mozilla
Persona
Sign in as...
badida@mozilla.com
ben@adida.net
benadida@eyedee.me
benadida@mozilla.com
Add another email address
This is not me
sign in
crossword.thetimes.co.uk
Persona. Simplified sign-in, built by a non-profit. Learn more→

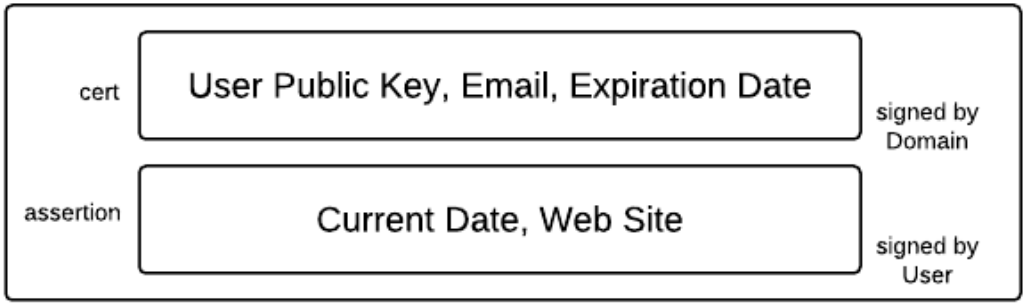

12:37
IdentityFlow
Firefox Accounts
powered by Persona
notoriousb2g.123done.org
Sign in as...
jparsons@mozilla.com
Add another email address
This is not me
sign in

How to implement it

- Ask for authentication

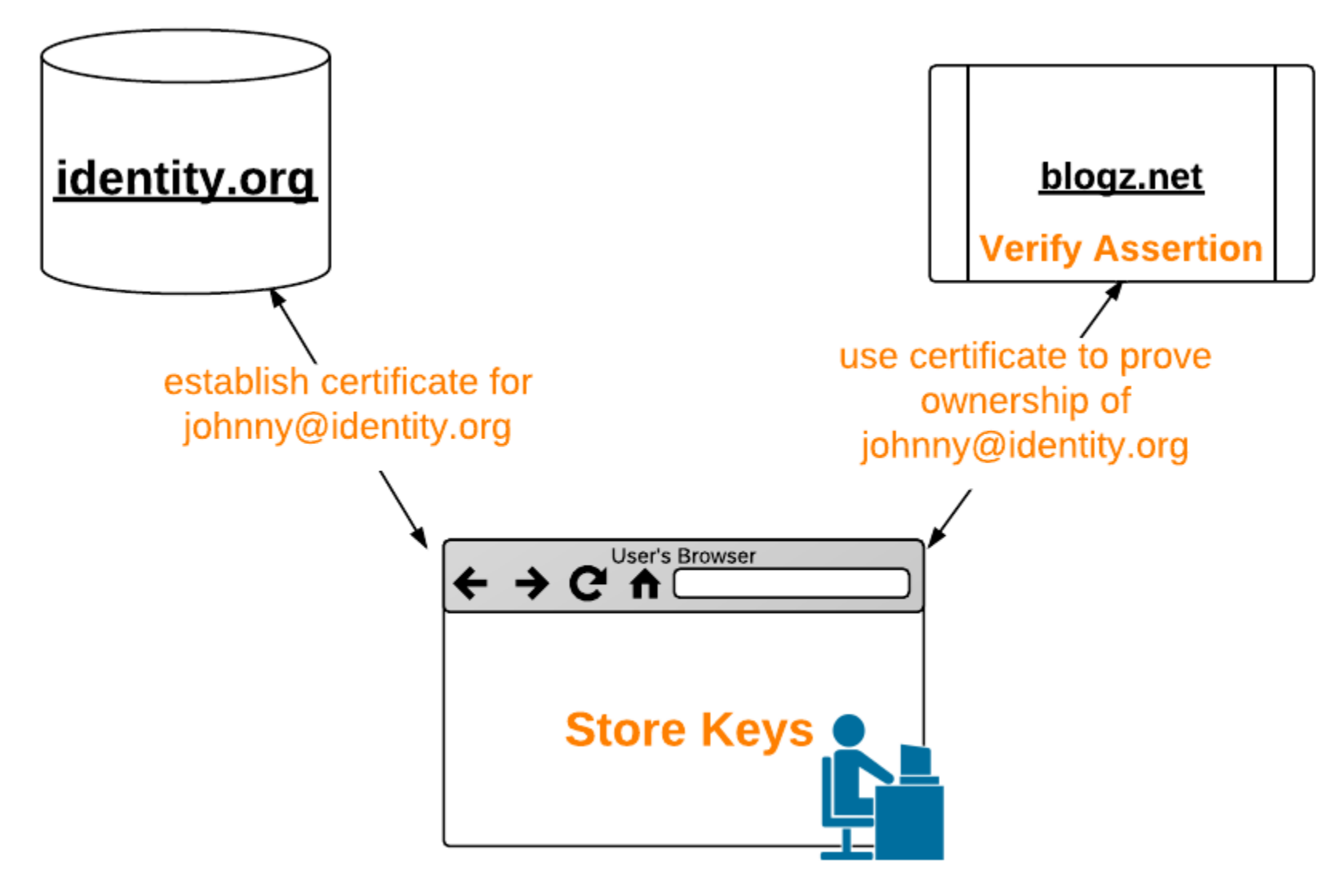
```
navigator.id.request();
```

- Obtain proof of email address ownership:

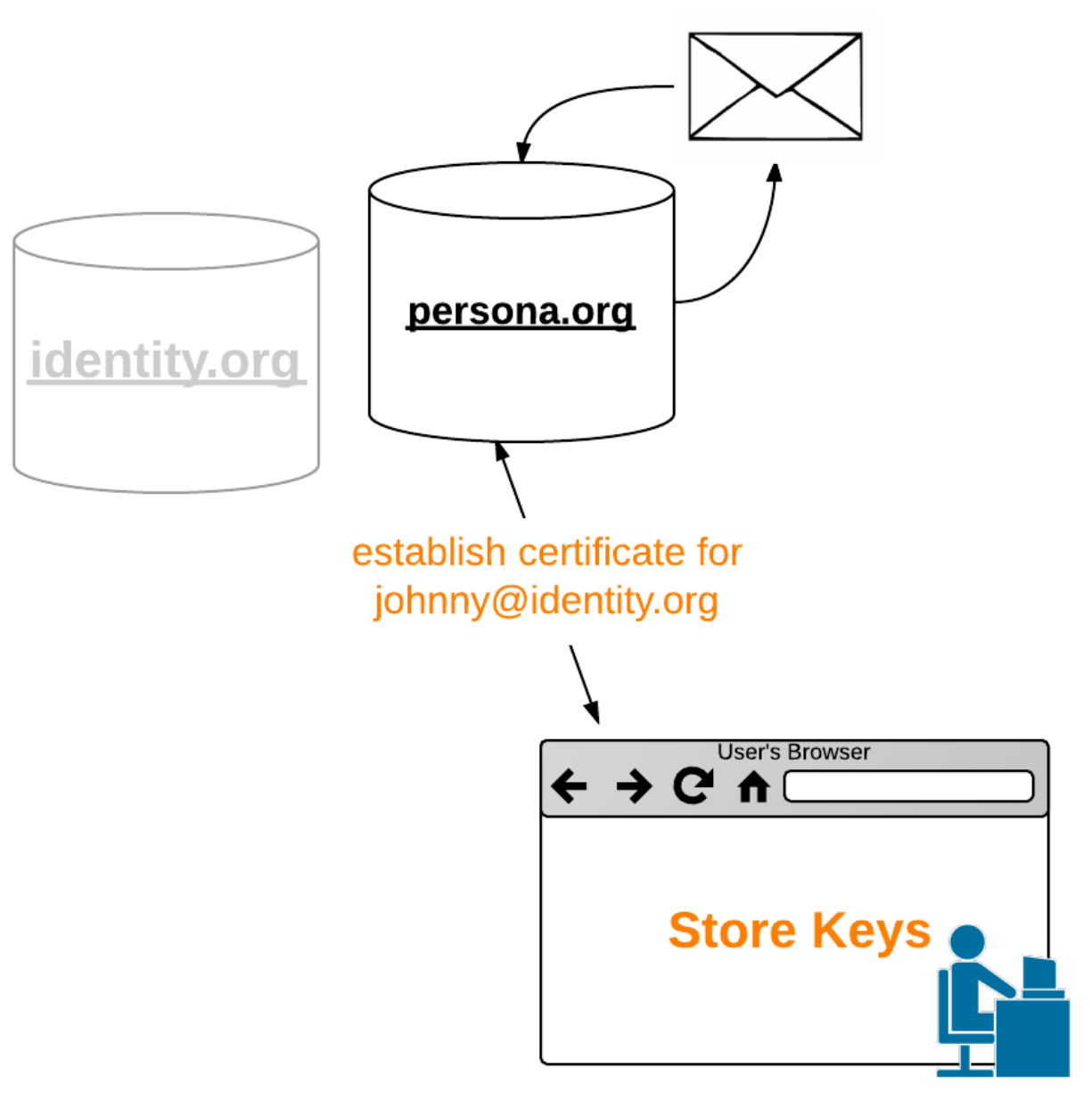


- Verify it and go.

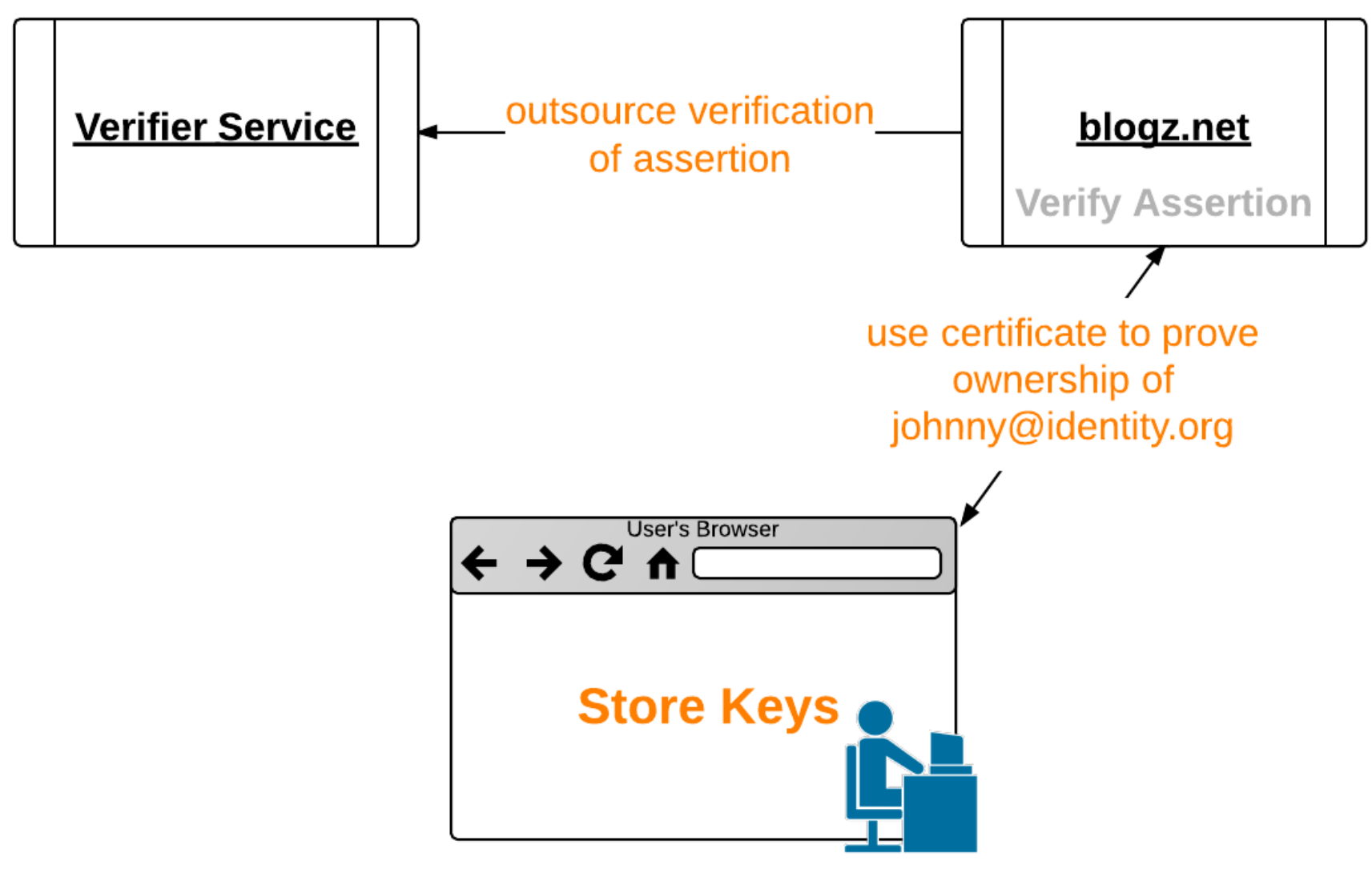
How it works


identity.org
blogz.net
Verify Assertion
establish certificate for
johnny@identity.org
use certificate to prove
ownership of
johnny@identity.org
User's Browser
Store Keys

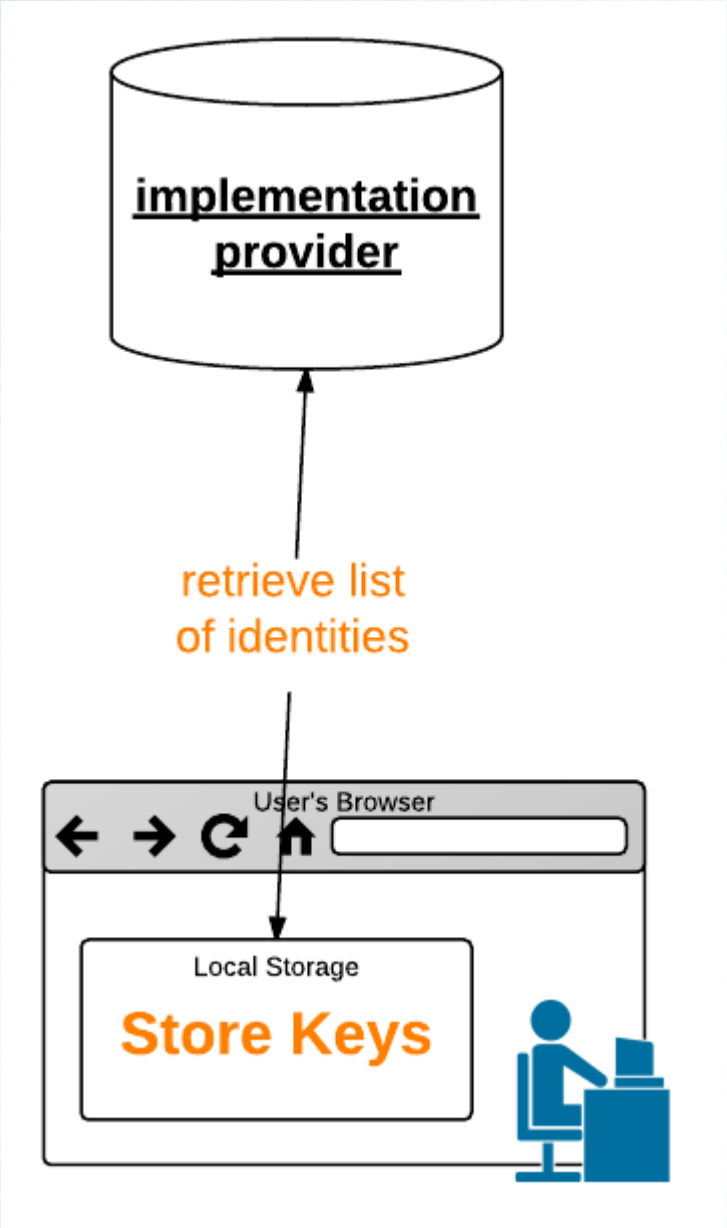
We Make it Work Today



We Make it Work Today



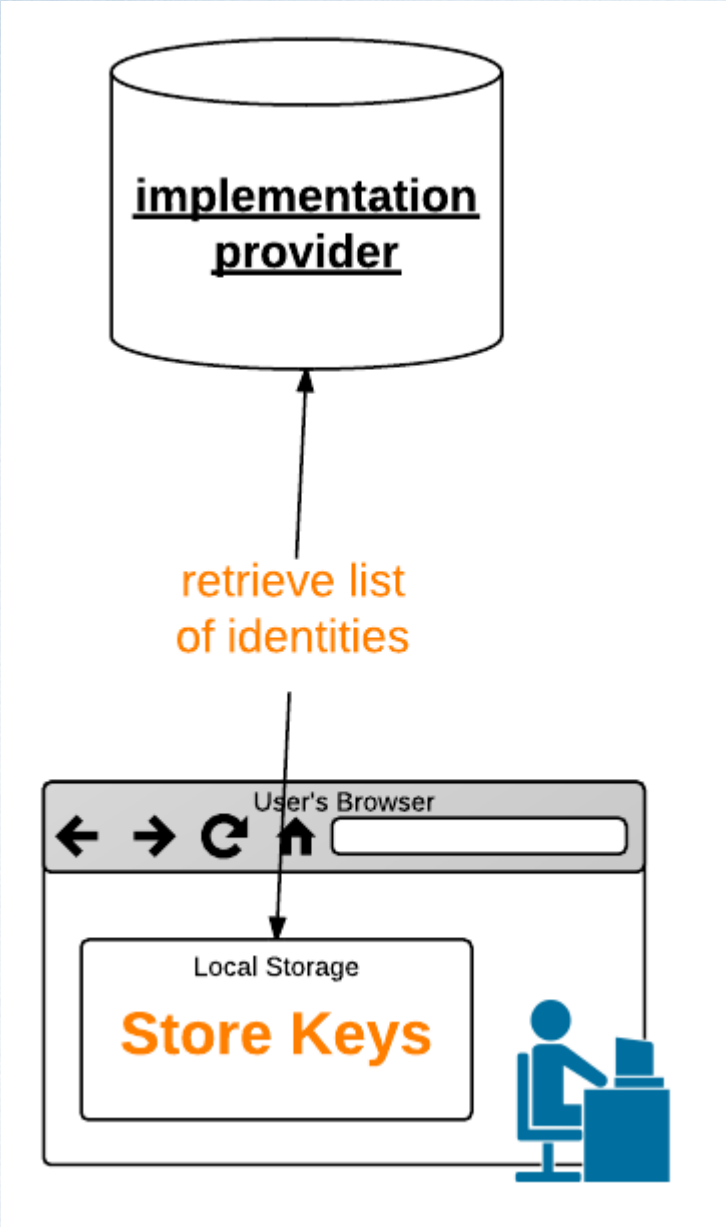
We Make it Work Today



- include a JS library
- that implements navigator.id.*
- backend server & postMessage communication to shim new browser functionality.

Why this approach?

- works today, in 20 LoC, on all browsers with a shim Javascript library
- as browsers implement native support, the user experience improves dramatically while web sites don't need to change a thing
- deploying new distributed technology is hard scaffold + strategy for removing scaffolding



- in-browser crypto for e2e encryption. Clipperz, SJCL, Mozilla, Helios, ...
- "web sites that do crypto in browser Javascript are doomed."
- user has no idea **what crypto** is running, or even **if crypto** is running.
- if attacker breaks into server, can change client crypto.
- so what's the point? Do it on the server.

Highly Pragmatic Reasons

- progressive enhancement to web
- packaged web apps are coming
- increasingly stronger guarantees of code integrity (CSP, HSTS, ...)

But even without that...

Mistake #1

Trust is all-or-nothing

if I trust server to deliver crypto code,
I might as well trust it with my data.

Mistake #2

Compromises are all-or-nothing

if attacker can extract DB data
can also modify served JS code.

Mistake #3

All targets are equally attractive

a server with millions of user accounts
vs. a single user's computer

Mistake #4

Implementations are perfect

e.g. discarding information is the same as never receiving it in the first place.

security in practice
is about defense in depth

Crypto in browser is
one darn good defense

threat models are not all-or-nothing