# The Rise of Cryptographic Authentication

**Presentation at SJSU, April 5, 2018**

**Updated April 9**

Francisco Corella
fcorella@pomcor.com
Karen Lewison
kplewison@pomcor.com

# Cryptographic authentication

- Authentication by demonstrating possession/knowledge of a cryptographic key
- Very successful for server authentication
  - X.509 TLS certificates
- Limited use on the client
  - TLS client certificates
  - PIV Authentication certificates in PIV or CAC cards
  - Anonymous credentials

# Passwords are still the dominant client authentication technology

- *Two-party* authentication
  - E.g. user authentication to a web app
  - User registers password, submits password at authentication time to be recognized as repeat visitor
- *Three-party* authentication via federated login, e.g. login with Facebook
  - Relying party (RP) redirects browser to identity provider (IdP)
  - IdP authenticates user as repeat visitor with password
  - IdP redirects browser back to RP, passing user info

# But problems with passwords have reached a critical mass

- Phishing
- Proliferation of passwords to be remembered by user
- Password reuse leading to capture on malicious sites
- Frequent breaches of large password databases
- Difficulty of entering high-entropy passwords on smart phone

# Alternatives have emerged but have their own issues

- Two-factor authentication with one-time password (OTP) sent in message or produced by hardware or software generator
  - Carries cost and friction
- Explosion of biometric authentication modalities and architectures
  - Local biometric verification to unlock device
  - Remote biometric verification by voice or face recognition to log in to a bank

# Hard problems in biometric authentication

- Presentation attack detection (spoofing detection)
  - Voice morphing
  - Fake finger created from photograph
    - MSU research program on fake fingers
  - IARPA research program
- Adversarial perturbation attacks against face recognition
  - Glass frames
  - ICLR 2018

But cryptographic authentication, alone or in combination with other authentication methods, is rising as an effective solution for client authentication

# This is made possible by recent technology developments

- JavaScript morphing into a powerful and fast programming language, usable on client, server (under Node.js) and native apps (within React)
- Emergence of many web APIs
  - Web Storage API
    - Provides "HTML5 *localStorage*"
  - IndexedDB API
  - Web Cryptography API
  - Service Worker API
  - Web Authentication API

# Cryptographic authentication supports many different architectures, which can be classified along two facets

- Parties involved
  - Two-party (2P), repeat-visitor authentication
  - Authentication by trusted third party (3P)
    - Involved or not involved at transaction time
- Factors
  - One factor (1F)
  - Multiple factors (2F, 3F)
    - All of them verified remotely
    - Some of them verified locally to unlock the cryptographic credential

# A sample of cryptographic authentication architectures for web applications

A. Key pair stored in browser (2P, 1F)
B. 2P-1F with credential in secure storage
C. 2P-2F with biometric unlocking of credential
D. Like (A) + password, with *joint hash* of password and public key (2P, 2F) stored in database
E. Third-party cryptographic credential, presented by a "service worker", with issuer not involved in transaction (3P, 1F)
F. "Rich credential" with credential issuer not involved in transaction, using a "service worker" (3P, 3F)

# A. Key pair stored in browser

- Credential is username plus DSA, ECDSA or RSA key pair, which replaces password
  - Stored in browser, protected by *same origin policy*
  - Private key may be made unextractable even by same origin
- *Registration*: JavaScript (JS) front-end (FE) of web application generates key pair, registers pubic key with back-end (BE) (and proves possession of private key to back-end)
- *Login*: JS FE proves possession of private key
- Proving possession of private key means signing a challenge from the BE
  - JS redirection + HTTP redirection

# B and C: FIDO specifications

- Key pair is kept in an *authenticator*, which may provide secure storage (protected against malware and possibly against physical tampering), such as:
  - USB dongle
  - TPM
  - TEE
- Use of the key pair may be unlocked by fingerprint or face recognition, provided by the platform (phone or laptop)
- Optional authenticator attestation
- The W3C Web Authentication API specifies how the JS FE accesses the authenticator

# D. Joint hash of password and public key*

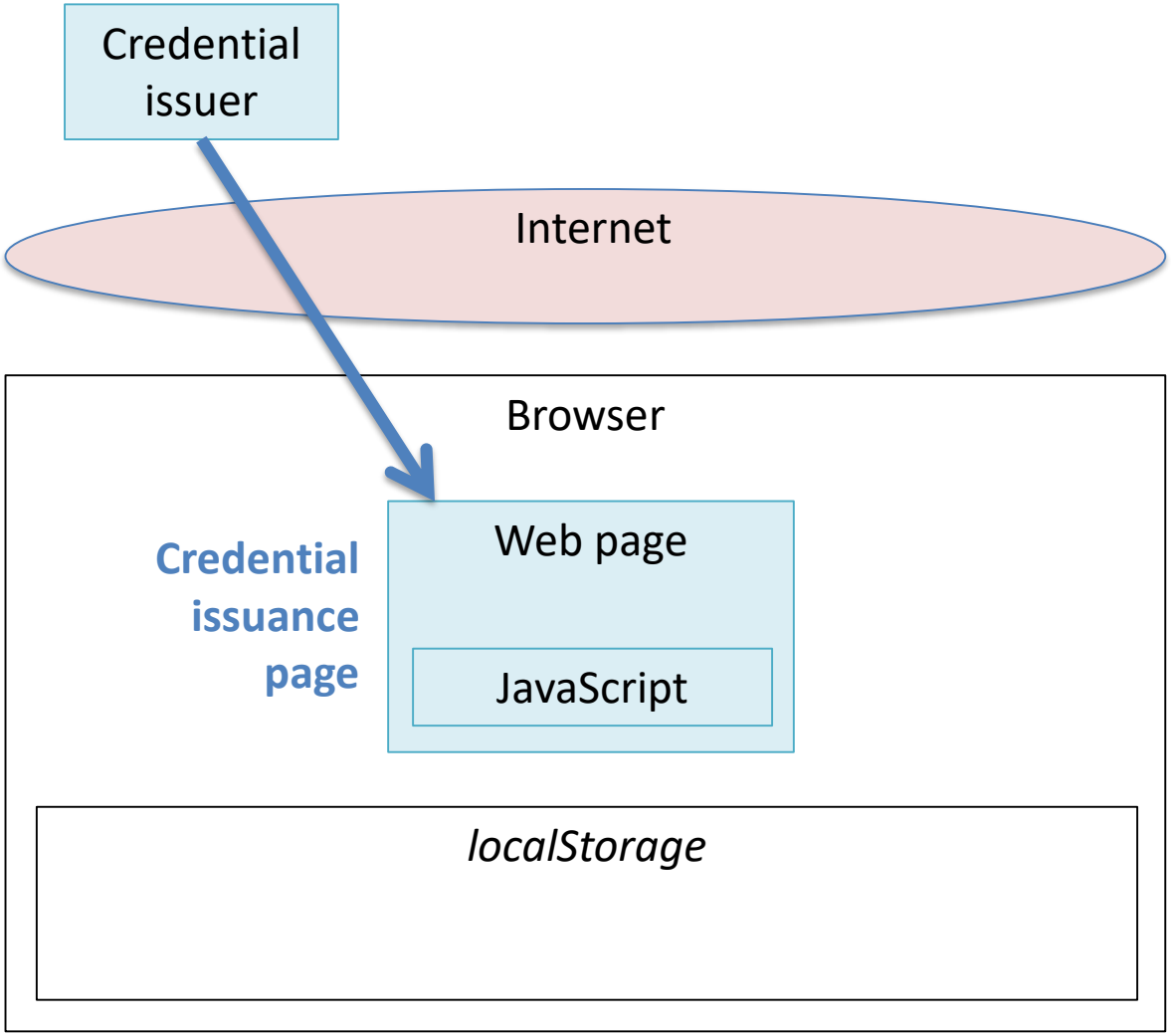
- Browser registers public key and password
- BE stores a joint hash of public key and password (instead of a salted hash of the password) in the user database
- Password is secure against a breach of the database

* As mentioned during the talk, Pomcor has been granted a patent related to this architecture (US Patent 9887989).

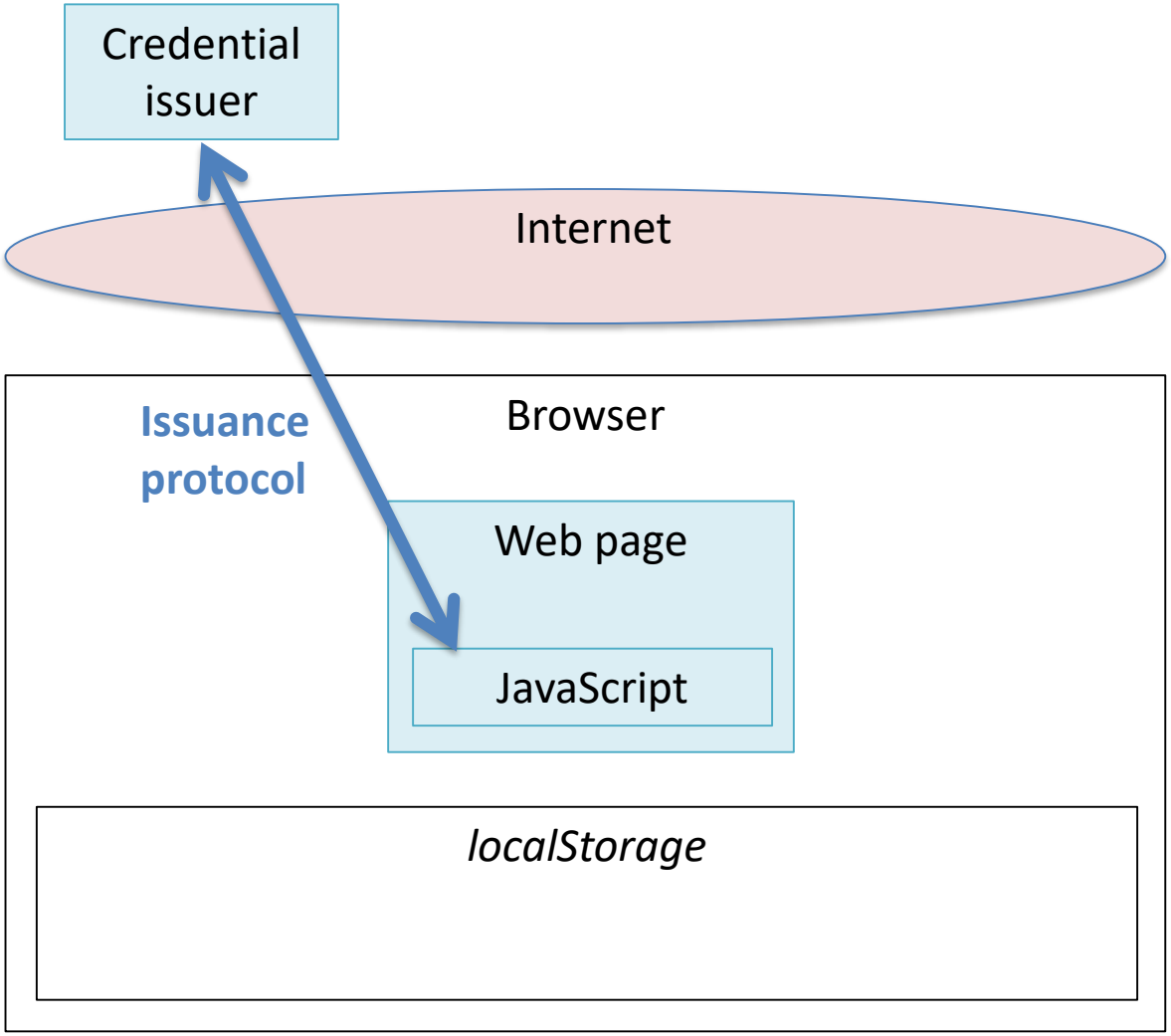
# E. Third-party cryptographic credential presented by a “service worker”

- A *service worker* is a *web worker* that can intercept HTTP requests to its back-end and respond locally
- At registration time, the credential issuer (the trusted third party) provisions the credential to browser and registers service worker with the browser
- At authentication time, relying party redirects to issuer, but redirected request is intercepted by service worker, which responds locally without involvement of the issuer back-end

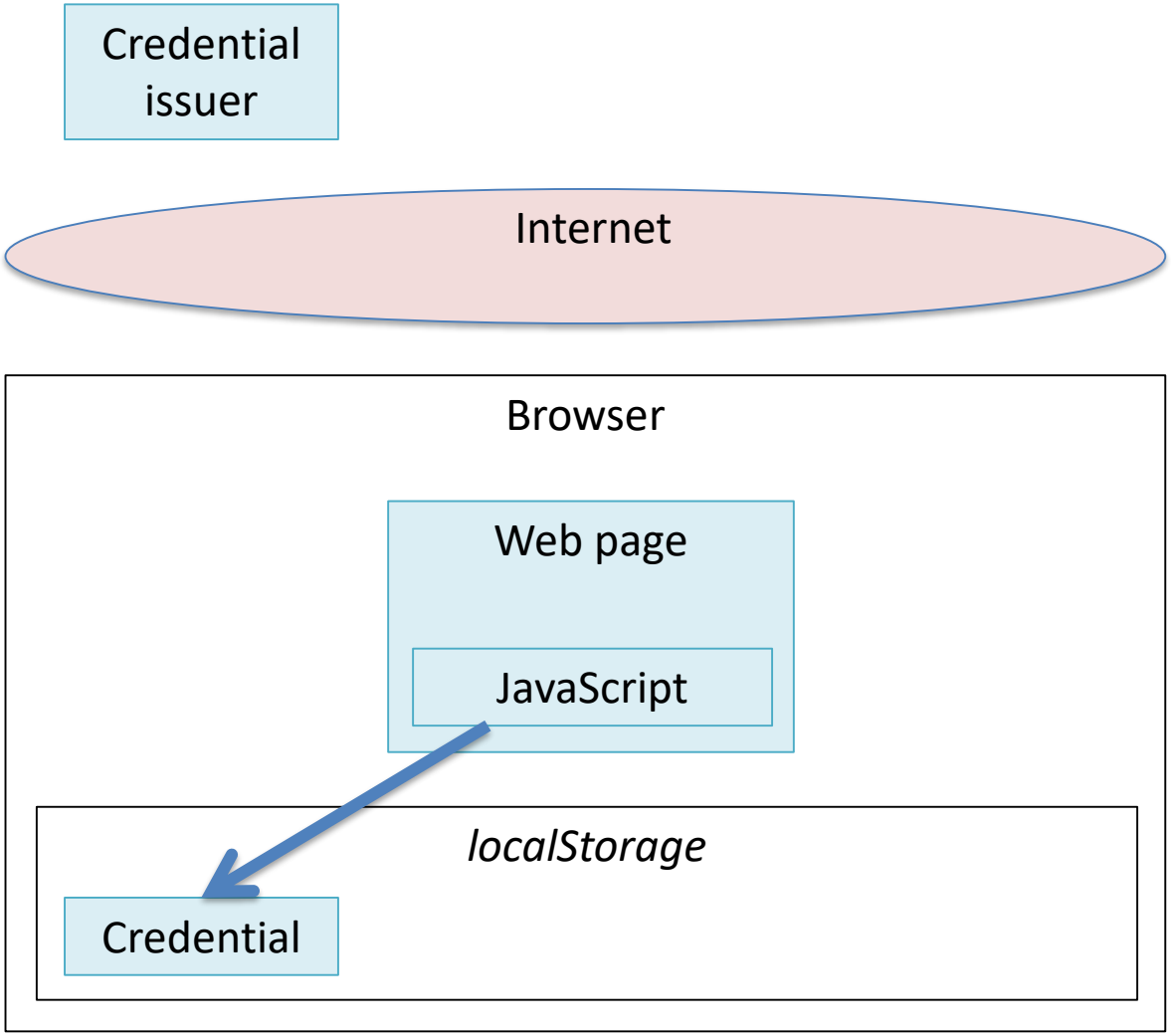
# Issuance


Credential issuer
Internet
Browser
Credential issuance page
Web page
JavaScript
localStorage

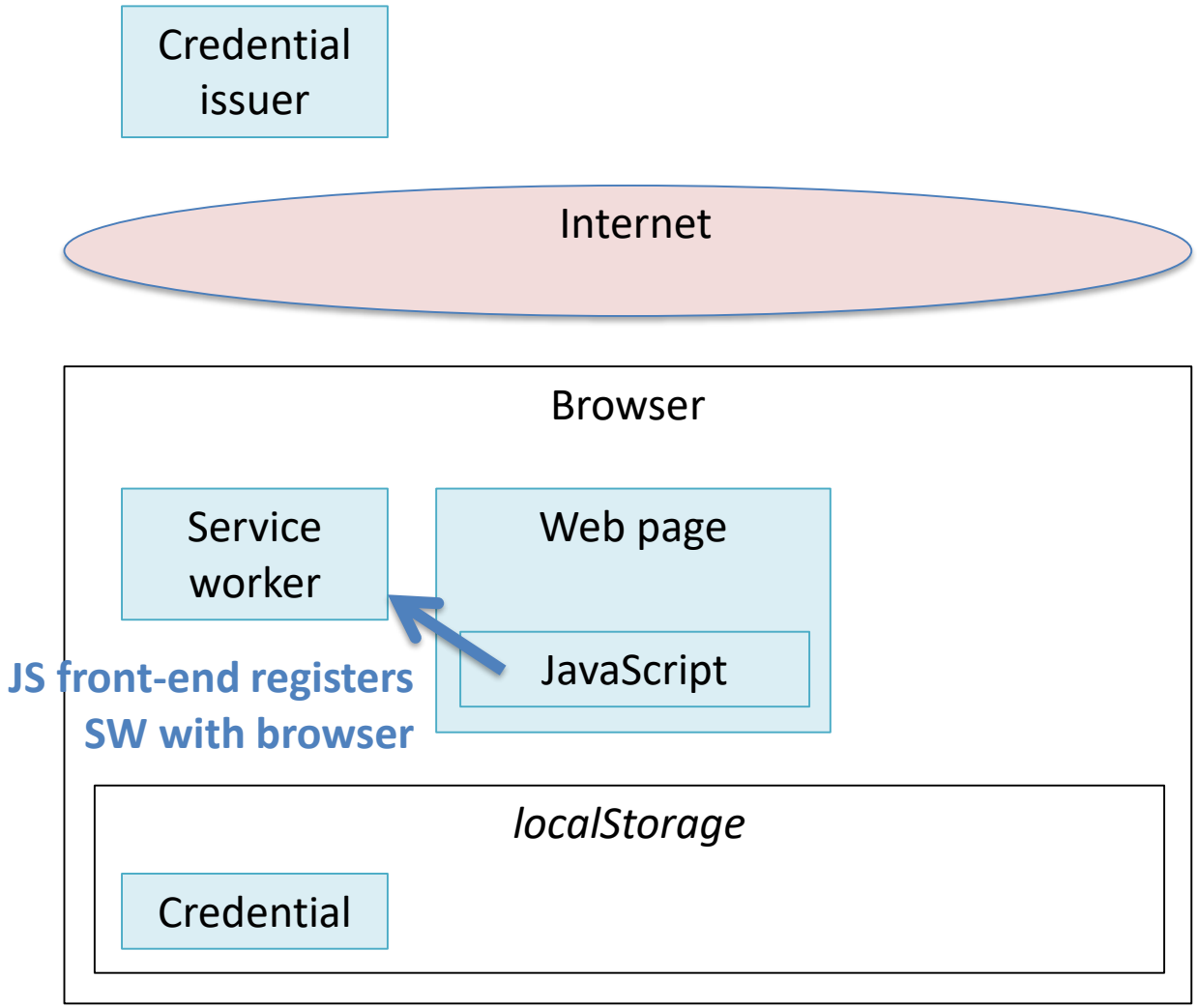
# Issuance


Credential issuer
Internet
Issuance protocol
Browser
Web page
JavaScript
localStorage

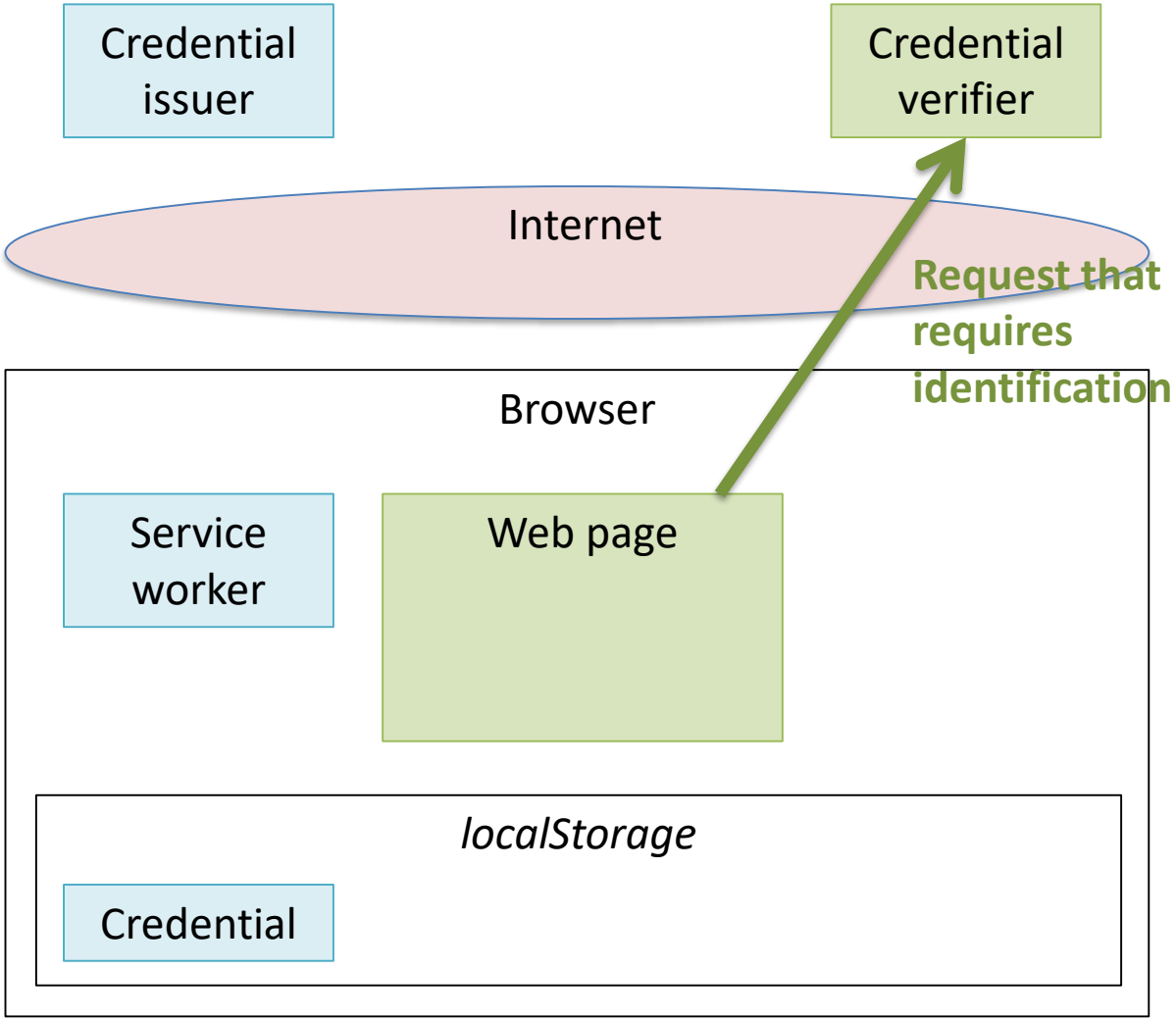
# Issuance


Credential issuer
Internet
Browser
Web page
JavaScript
localStorage
Credential

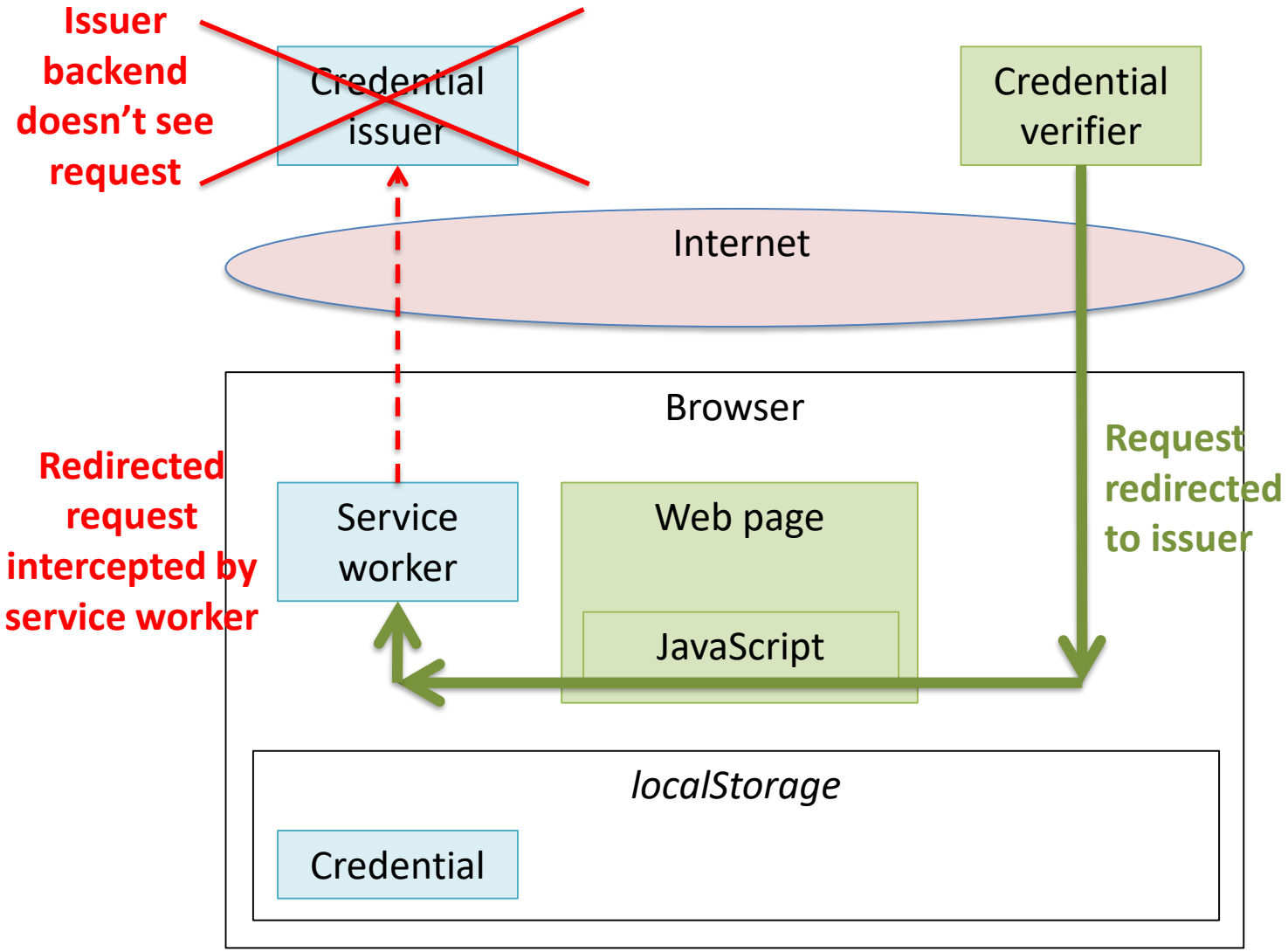
# Issuance


Credential issuer
Internet
Browser
Service worker
Web page
JavaScript
JS front-end registers SW with browser
localStorage
Credential

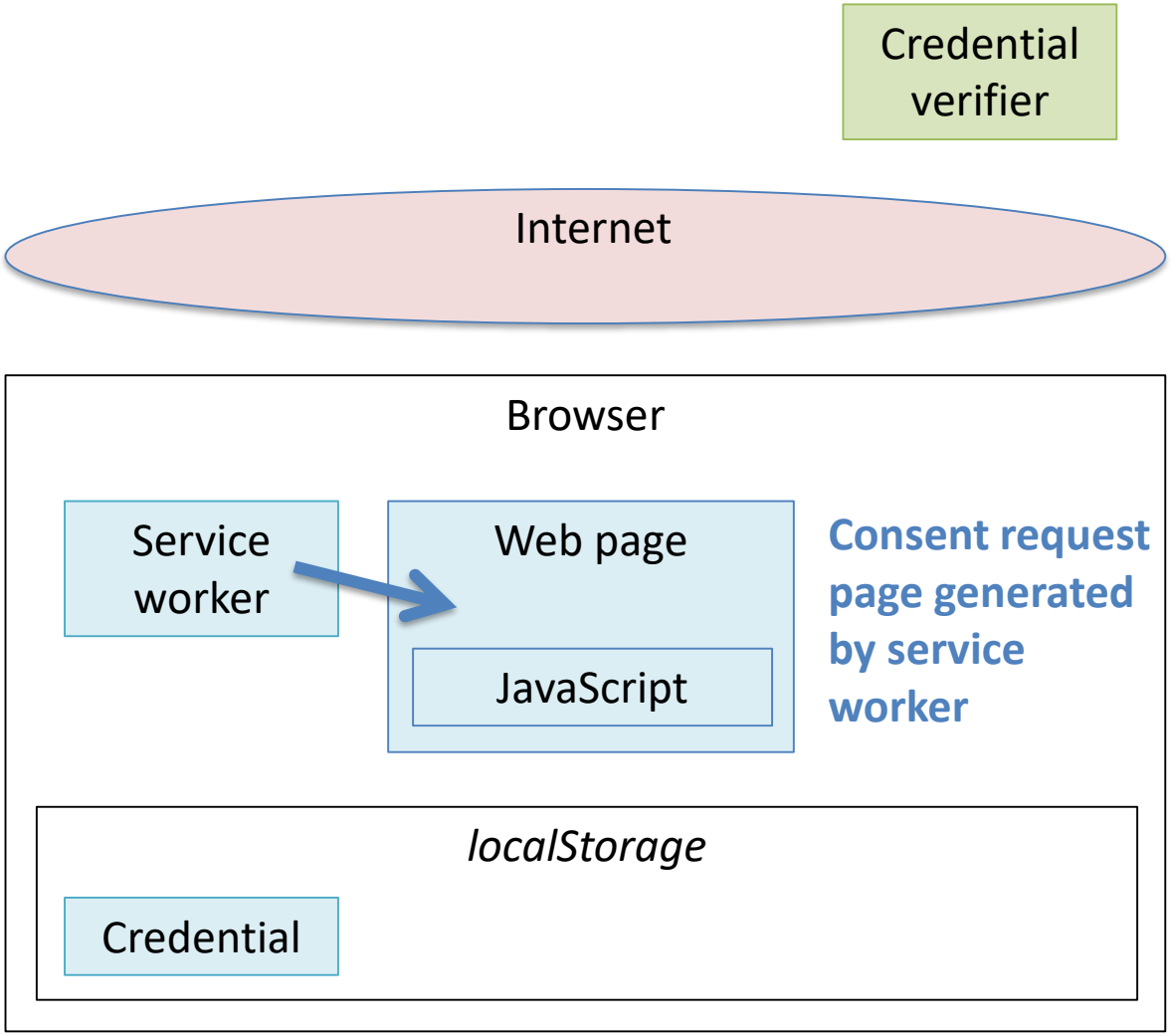
# Presentation



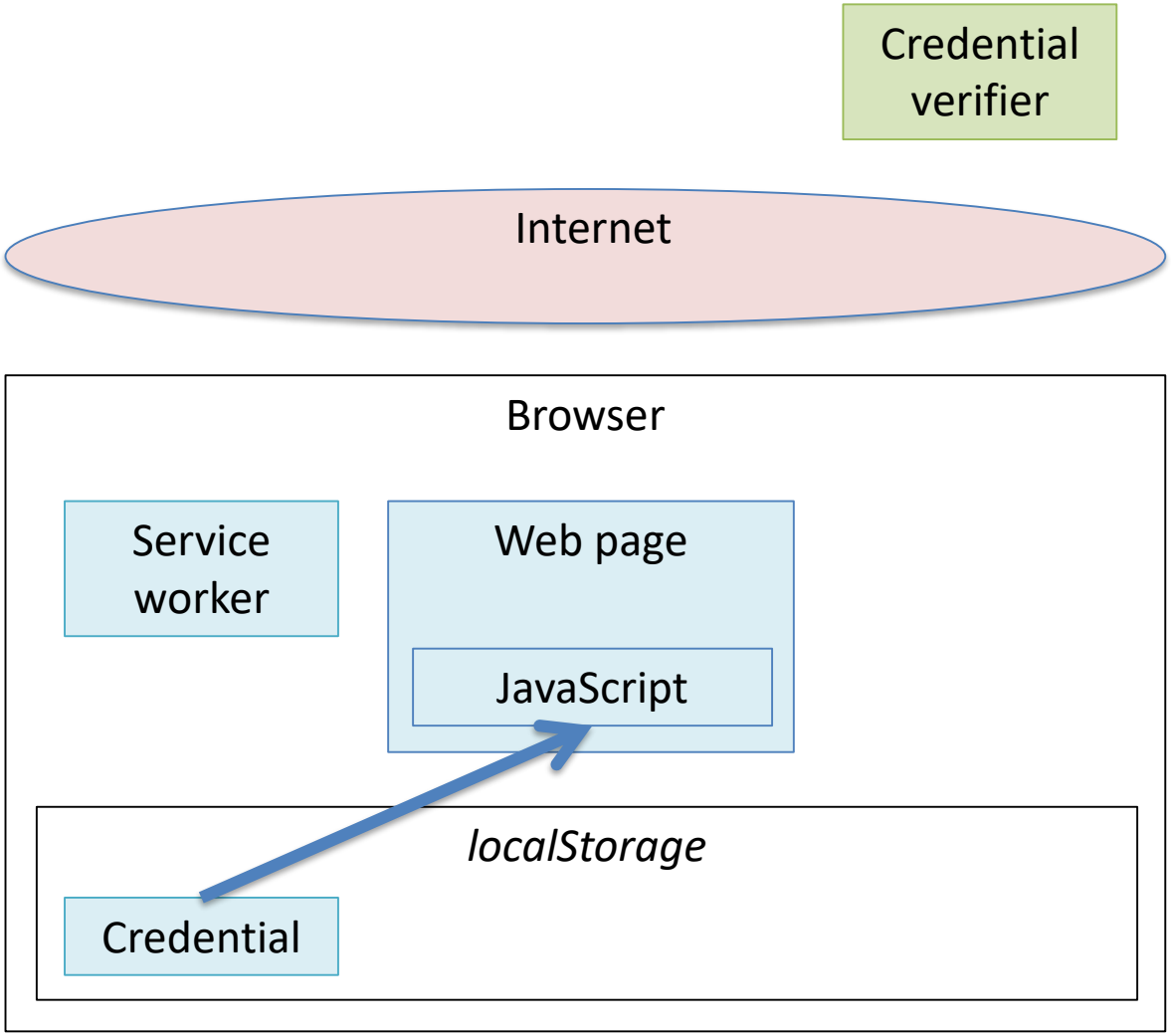
# Presentation



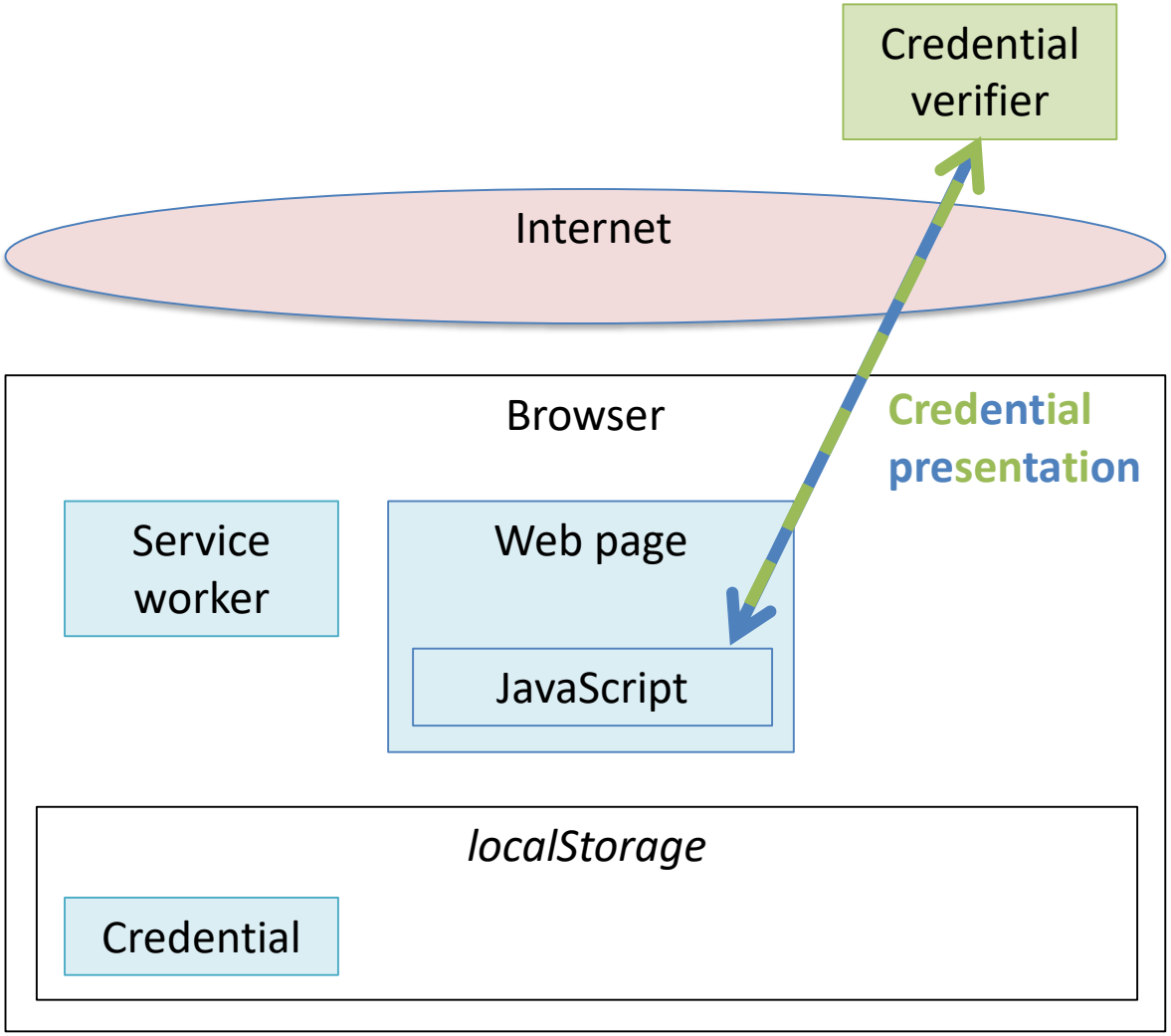
# Presentation



# Presentation



# Presentation



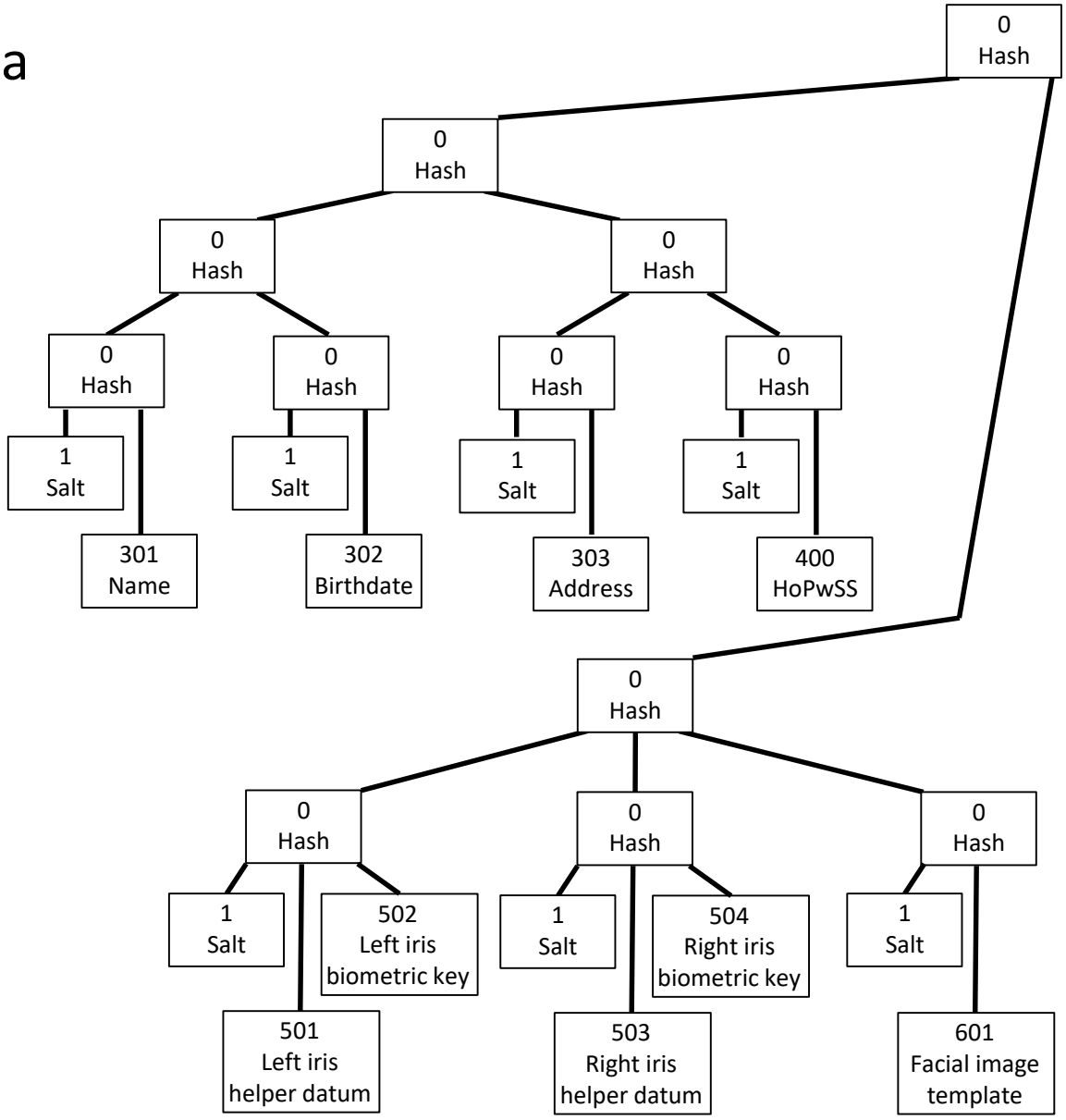
# F. “Rich credential”

- Cryptographic credential extended to support 3F authentication
- Includes biometric and password verification data signed by the issuer
- Allows biometric and password factors to be verified by relying party without prior relationship of the user with the relying party
- Uses a *typed hash tree* to provide selective disclosure of attributes and selective presentation of verification factors

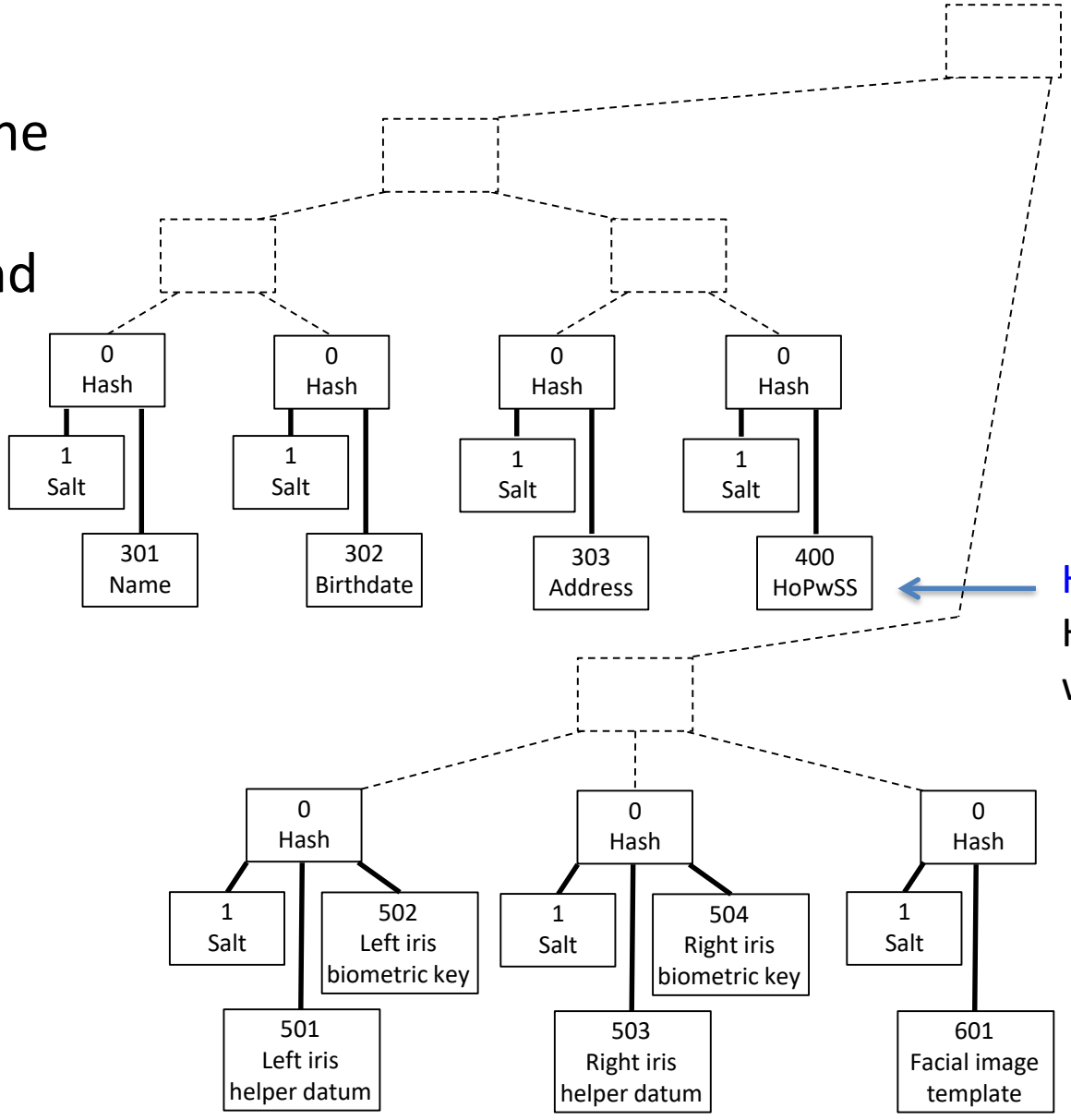
# Typed hash tree

- Each node has a *type* in addition to a label
- The label of a node is a cryptographic hash of the types and labels of its children
- Internal nodes have a *distinguished type*
- A subtree can be pruned, leaving a *dangling node* (a leaf node with the distinguished type)
- The root label serves as an *omission-tolerant cryptographic checksum* of the type-label pairs of the non-dangling leaf nodes
- The rich credential comprises a private key and a certificate that binds the public key to the root label

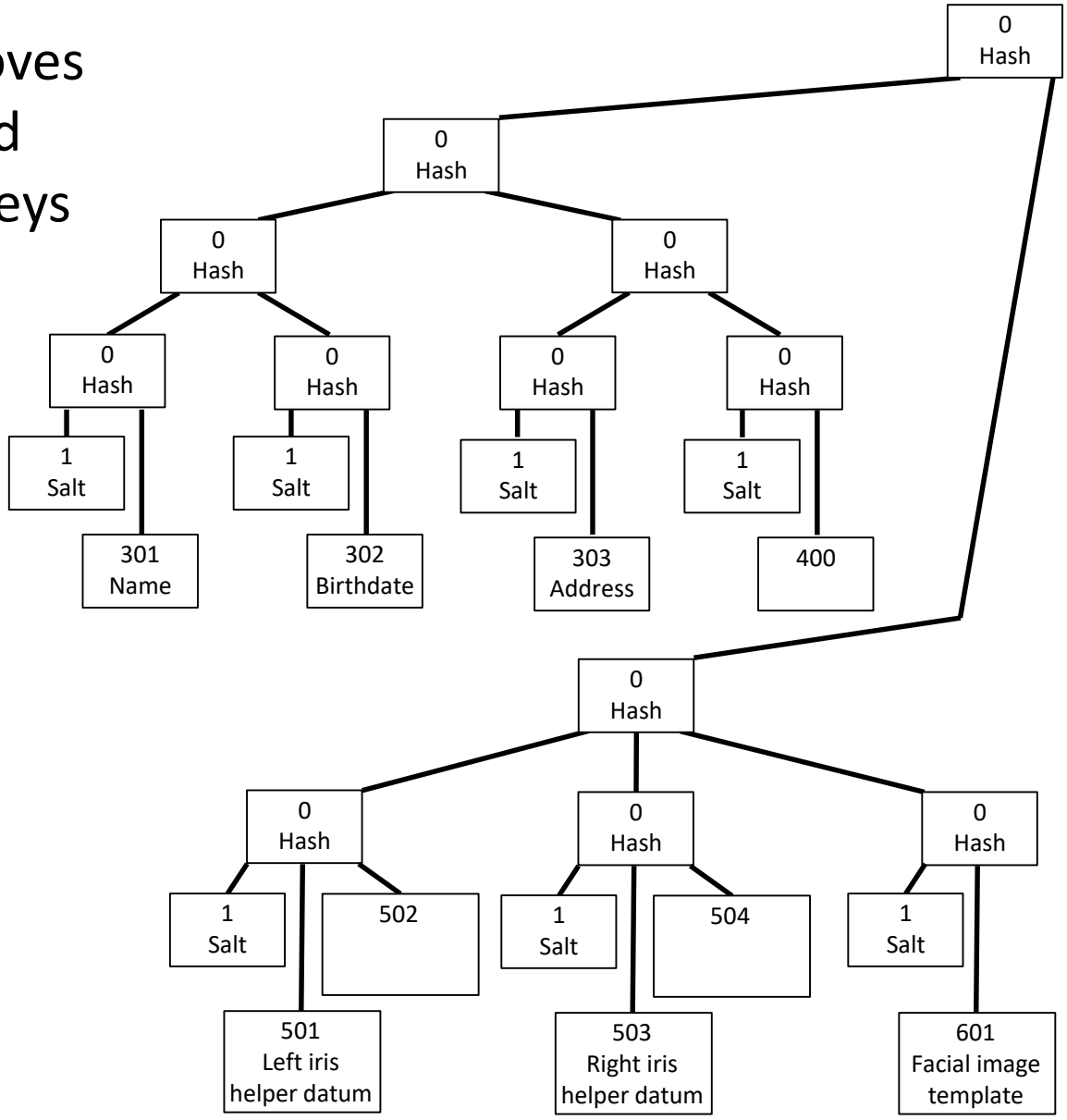
Example of a typed hash tree, in its issuance state



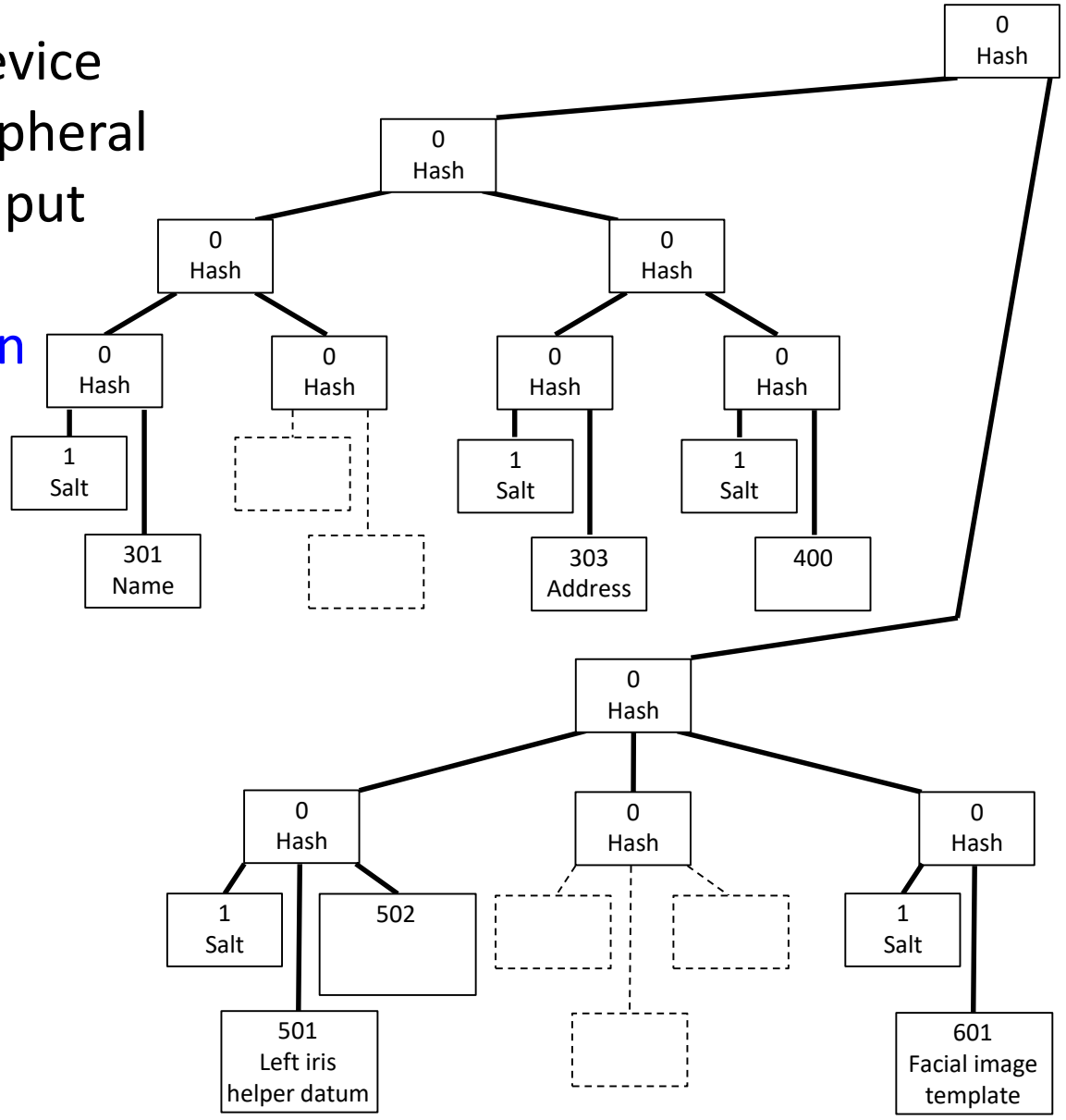
Peripheral subtrees, one for each attribute and each verification factor



Issuer removes HoPwSS and biometric keys to put tree in its storage state

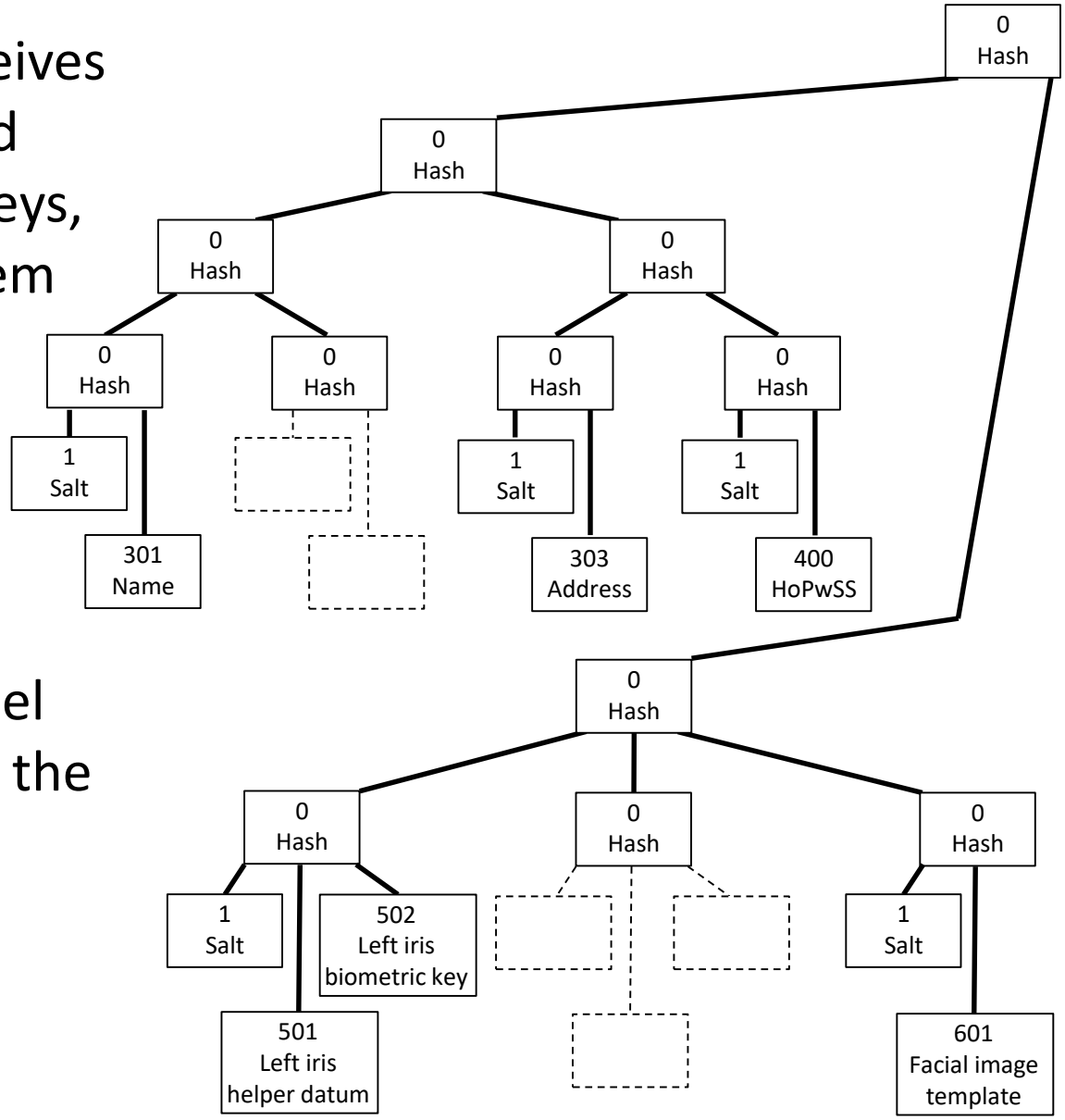


Subject's device prunes peripheral subtrees to put tree in its presentation state




Root label is omission-tolerant cryptogaphic checksum

Verifier receives HoPwSS and biometric keys, restores them to put tree in its verification state, computes the root label and verifies the certificate signature






pomcor
pomcor.com

# Thank you for your attention!

A demo of cryptographic authentication for a web application (architecture A) can be found at:

https://pomcor.com/pjcl/

For more information:

Francisco Corella

fcorella@pomcor.com

Karen Lewison

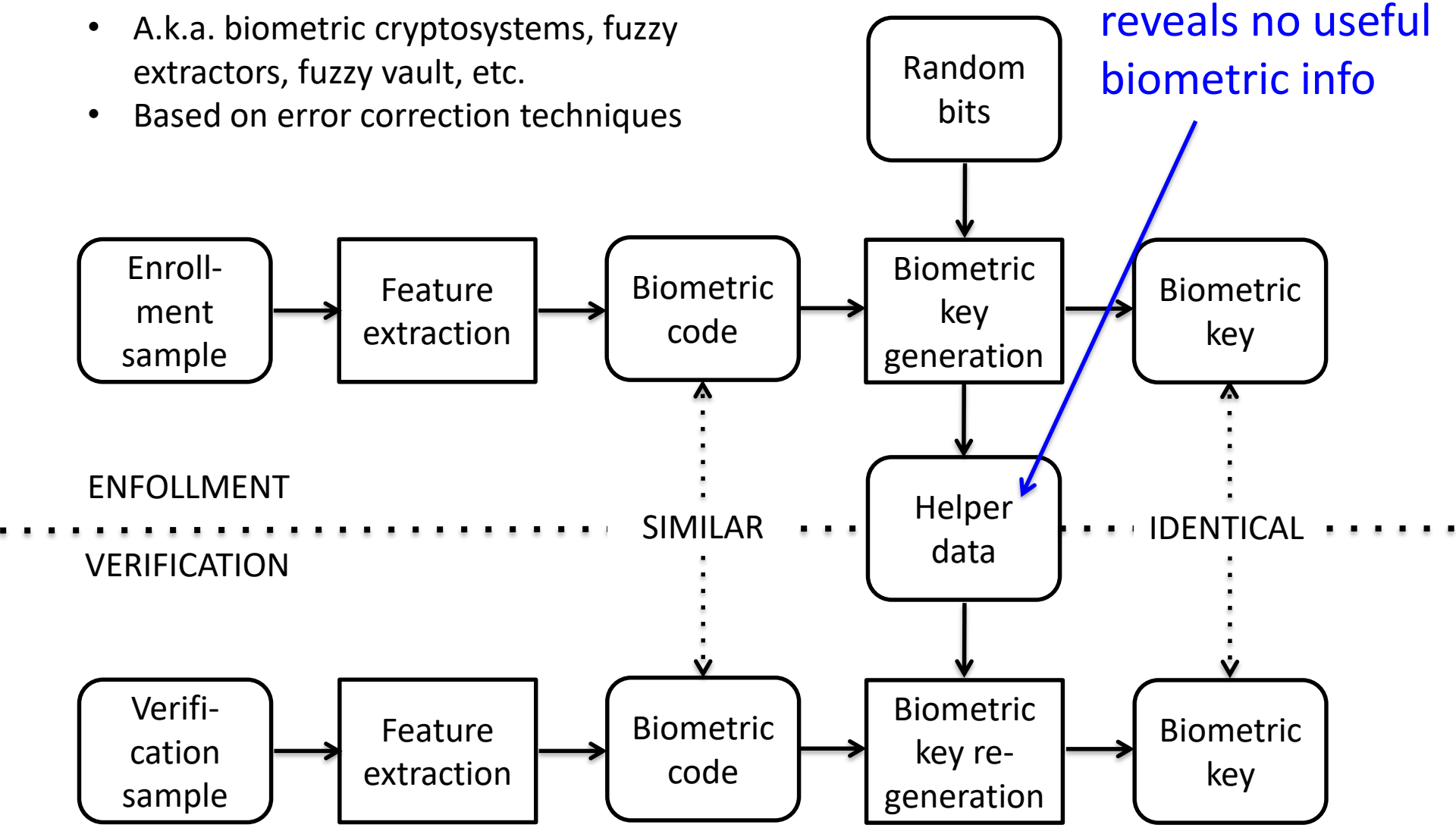
kplewison@pomcor.com

## Any questions?

# Appendix

# Revocable Biometrics

- A.k.a. biometric cryptosystems, fuzzy extractors, fuzzy vault, etc.
- Based on error correction techniques






pomcor
pomcor.com