IIW 15: New Authentication Method for Mobile Devices

Francisco Corella (fcorella@pomcor.com)
Karen Lewison (kplewison@pomcor.com)
Pomcor (http://pomcor.com/)

User Authentication Challenges in Mobile Devices

- Ordinary passwords:
  - It is difficult enter high-entropy passwords
    - Difficult to type on small touchscreen keyboard
    - Entering different types of characters requires switching keyboards
  - Password characters are echoed by the keyboard itself, defeating the echo-suppression feature of the password box
- One-time passwords (OTP)
  - Cumbersome
  - Limited security
    - OTP can be intercepted or observed
    - OTP remains valid for several minutes

Highlights of the New Method

- No passwords (neither ordinary passwords nor one-time passwords)
- Public key cryptography without certificates
- Optional biometric authentication, without storing a biometric template
- Optional use of a trusted 3rd party
- App developers insulated from cryptographic and biometric complexities
- No browser modifications needed on mobile devices
- Can be adapted for desktop/laptop use via browser plug-ins

Use Cases

- No-user-input (1-factor) web login
- High security (2- or 3-factor) web login
- Enterprise login
- Use of 3rd party personal data store
- Social login without a password
- *Mobile data protection*

Ingredients

- Main ingredients:

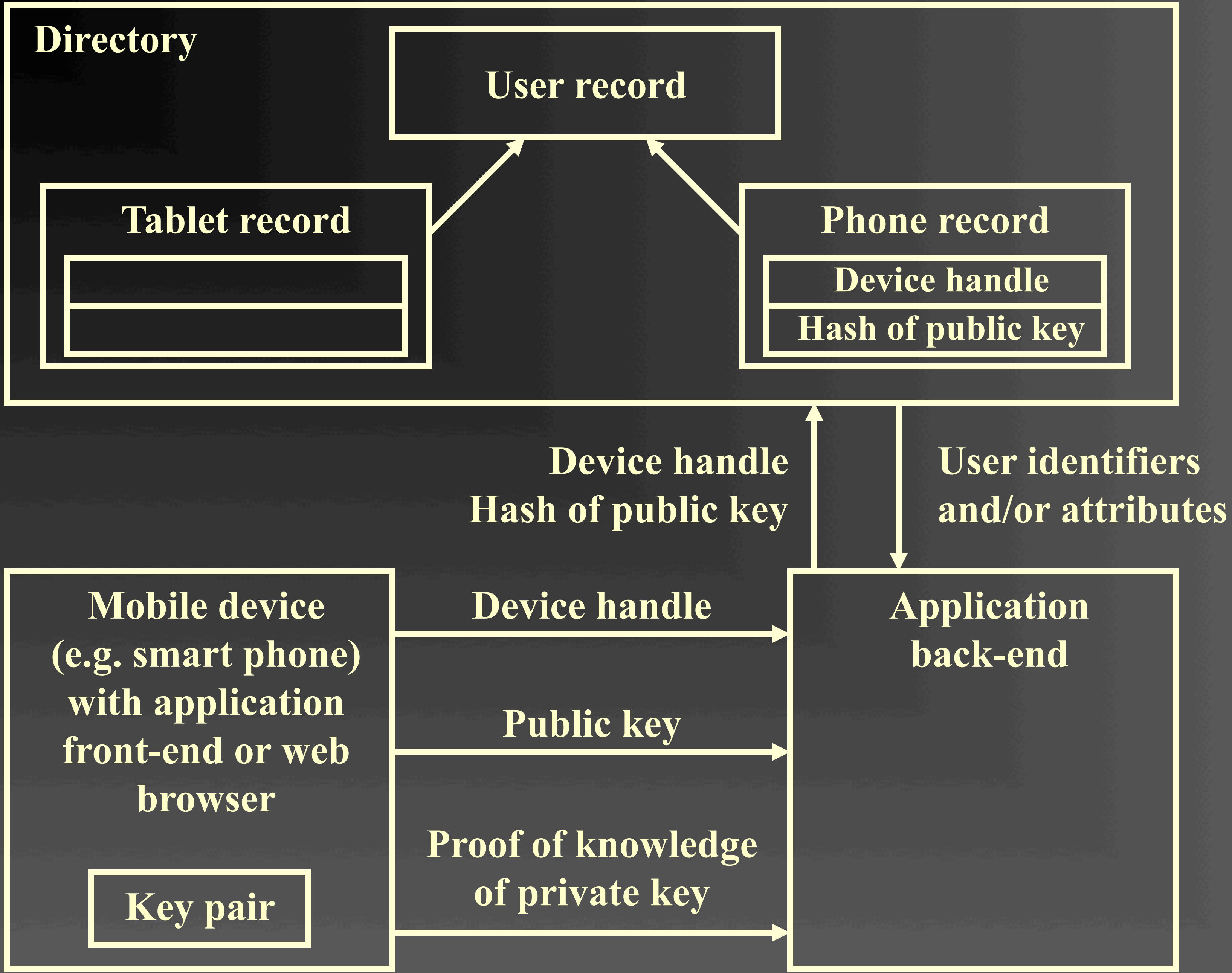
  1. Authentication with a raw key pair
  2. RSA key pair regeneration
  3. Derivation of biometric key from iris image
  4. Encapsulation of cryptographic and biometric processing

- Optional ingredients:

  5. Use of 3rd personal data repository (optional)
  6. Delegated authorization and social login (optional)

1. Authentication with a Raw Key Pair

- Mobile device → application (back-end):
  - Database handle that refers to a device record that contains the hash of public key and refers to user record (“device handle’)
  - Public key
  - Proof of knowledge of private key
- Application → directory / user database
  - Database handle of device record
  - Hash of public key
- Directory / user database → application
  - User identifier(s) and/or attribute(s)



Directory

User record

Tablet record

Phone record

Device handle

Hash of public key

Device handle
Hash of public key

User identifiers
and/or attributes

Mobile device
(e.g. smart phone)
with application
front-end or web
browser

Key pair

Device handle

Public key

Proof of knowledge
of private key

Application
back-end

2. Key Pair Regeneration as an Alternative to Tamper Resistance

- A private key stored in a mobile device must be protected if the device is lost or stolen, but today's phones and tablets lack tamper-resistant storage
- The private key could be encrypted under a key-encryption key derived from user input such as a PIN, but that would make the PIN vulnerable to an offline brute-force guessing attack
- Instead we propose to regenerate the key pair from the PIN
- All PINs produce well-formed key pairs, so PINs cannot be tested and an offline attack is not possible

RSA Key Pair Regeneration from a PIN

(Notations as in Handbook of Applied Cryptography, §8.2)

- Retain the prime factors *p* and *q* of the modulus, but not the encryption and decryption exponents *e* and *d*
- Generate *d* as a randomized hash of the PIN with seed *s*, of same length as the modulus (e.g. using the PRF of TLS)
- Compute *e* such that $1 < e < \varphi$ and $ed \equiv 1 \pmod{\varphi}$
- Only p, q and s are stored in the device

RSA Key Pair Regeneration from a PIN (Continued)

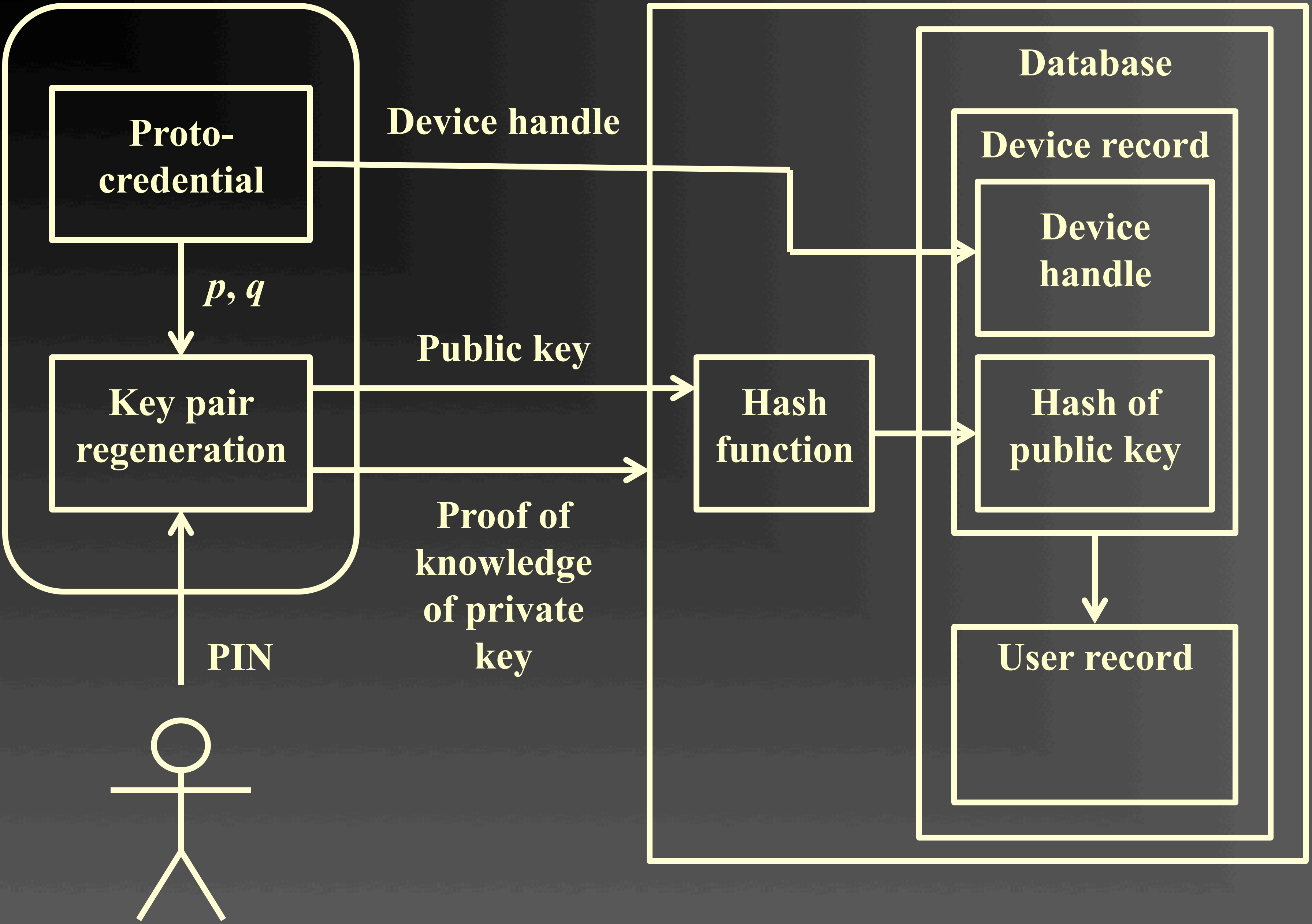
- Problem: what if gcd(d,φ) ≠ 1?
- Solution:
  - Remove from d all prime factors $r < 100$ shared with φ.
  - During initial key generation, if d has prime factors $r' > 100$ shared with φ, we start over with different p and q
  - The probability of having to start over is only 0.2%

RSA Key Pair Regeneration from a PIN (Continued)

- Note: retaining *p* and *q* does not reduce security
  - They could be computed from the key pair
  - They are often retained to take advantage of the Chinese Remainder Theorem
- Note: *d* not vulnerable to small-decryption-exponent attacks because it is only a few bits shorter than the modulus

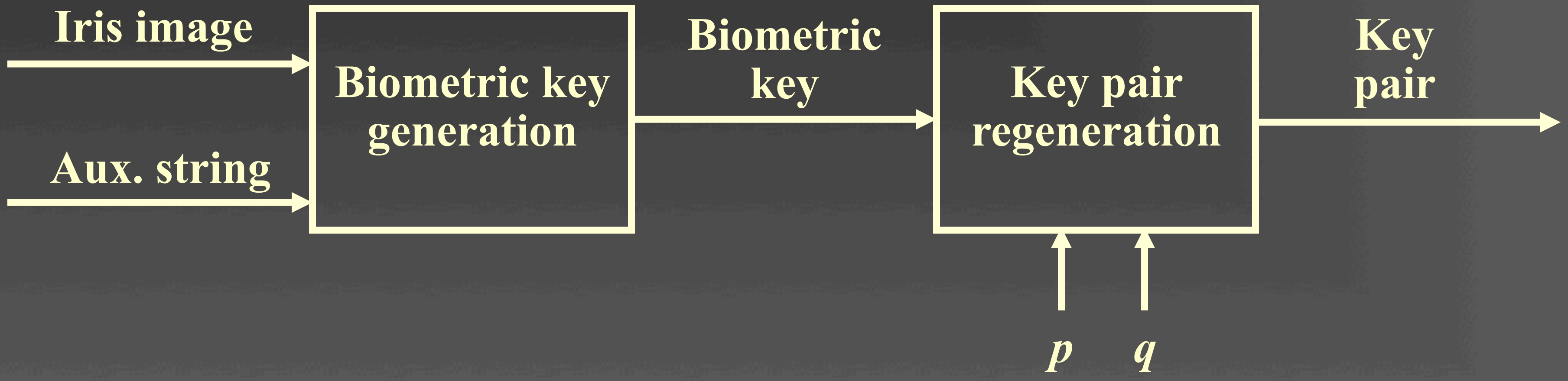
Regeneration from PIN + Authentication

- Device contains *protocredential* (*h*, *p*, *q*, *s*) (where *h* is the device handle)
- User enters PIN
- Device regenerates key pair
- Device sends device handle and public key to app back-end, and demonstrates knowledge of private key
- App back-end hashes public key, locates devices record and verifies it contains hash of public key, then locates user record


Database
Proto-credential
Device handle
Device record
Device handle
p, q
Public key
Key pair regeneration
Hash function
Hash of public key
Proof of knowledge of private key
PIN
User record

3. Regeneration from Biometric Key

- Biometric key generated from an iris image (to be taken by device camera) and an auxiliary string
  - *F. Hao, R. Anderson, and J. Daugman. Combining Cryptography with Biometric Effectively. IEEE Trans. Comput., 55(9):1081-1088, 2006.*
  - Biometric template not at risk because not used

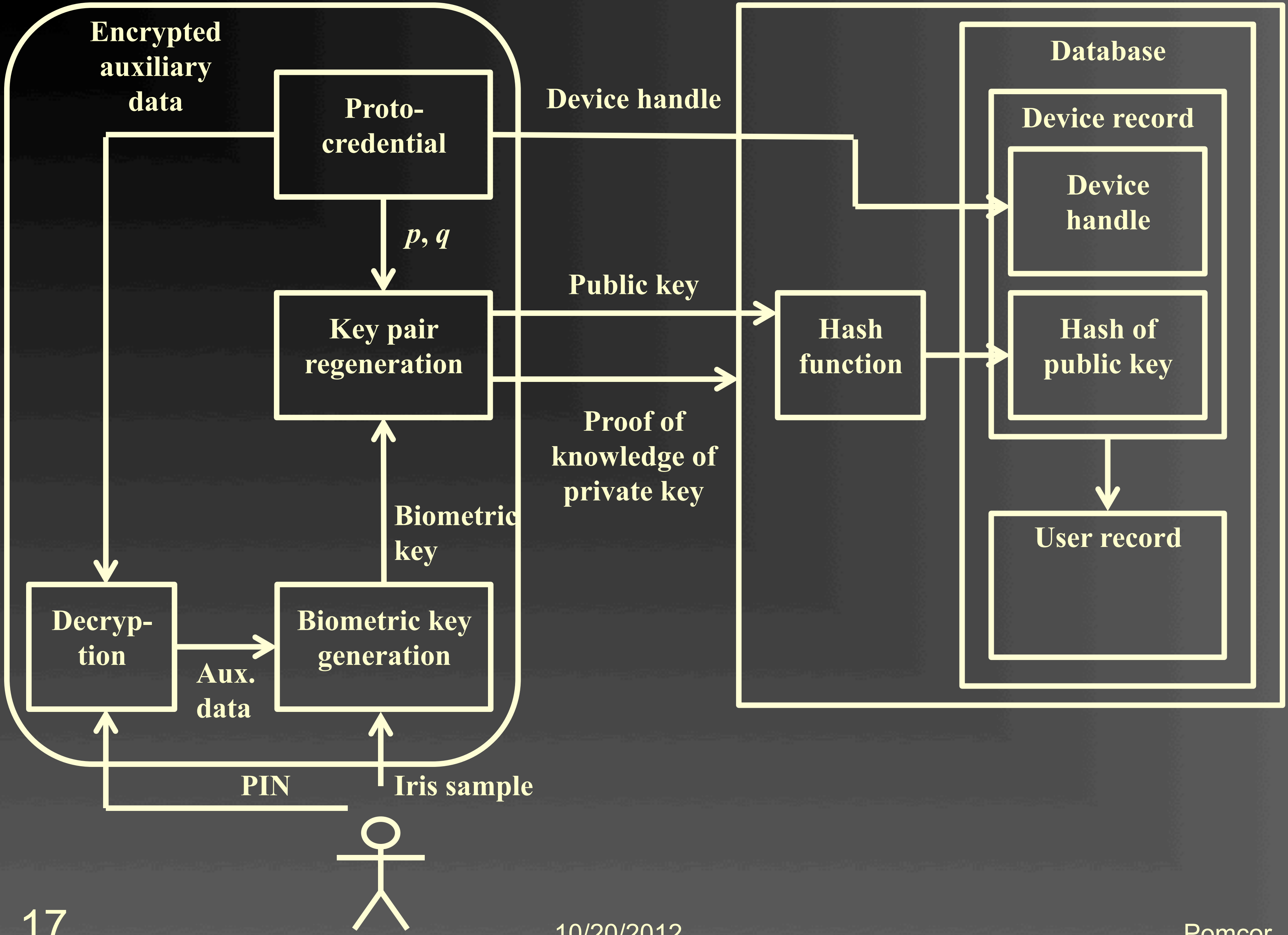


Biometric Key Generation

- Error correction scheme is used to correct small deviations from a *codeword*
- Enrollment:
  - Generate random codeword *C*
  - Obtain iris reference sample *R*
  - → Auxiliary string *A* = *C* xor *R*
- Biometric key generation
  - Use auxiliary string *A*
  - Obtain iris sample *S*
  - Compute *A* xor *S* = (*C* xor *R*) xor *S* = *C* xor (*R* xor *S*)
  - Error correction: *C* xor (*R* xor *S*) → *C*
  - *C* used as the biometric key, tolerates small variations in *S*

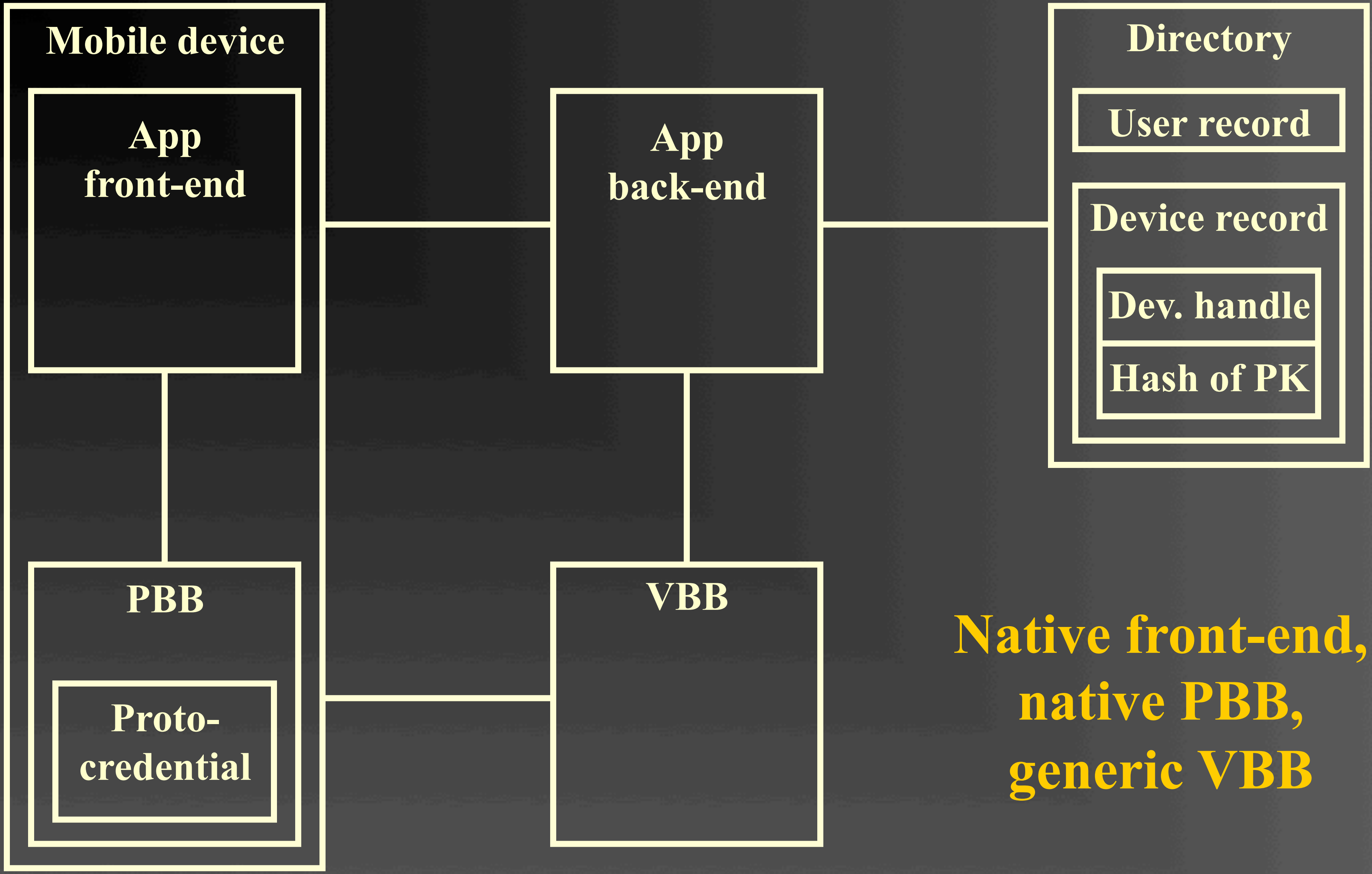
Three-Factor Authentication

- Factors:
  - PIN
  - Iris sample
  - Protocredential stored in mobile devicd
- Protocredential:
  - Device handle
  - Auxiliary data (*C* xor *R*) encrypted by PIN
  - RSA prime factors *p*, *q*

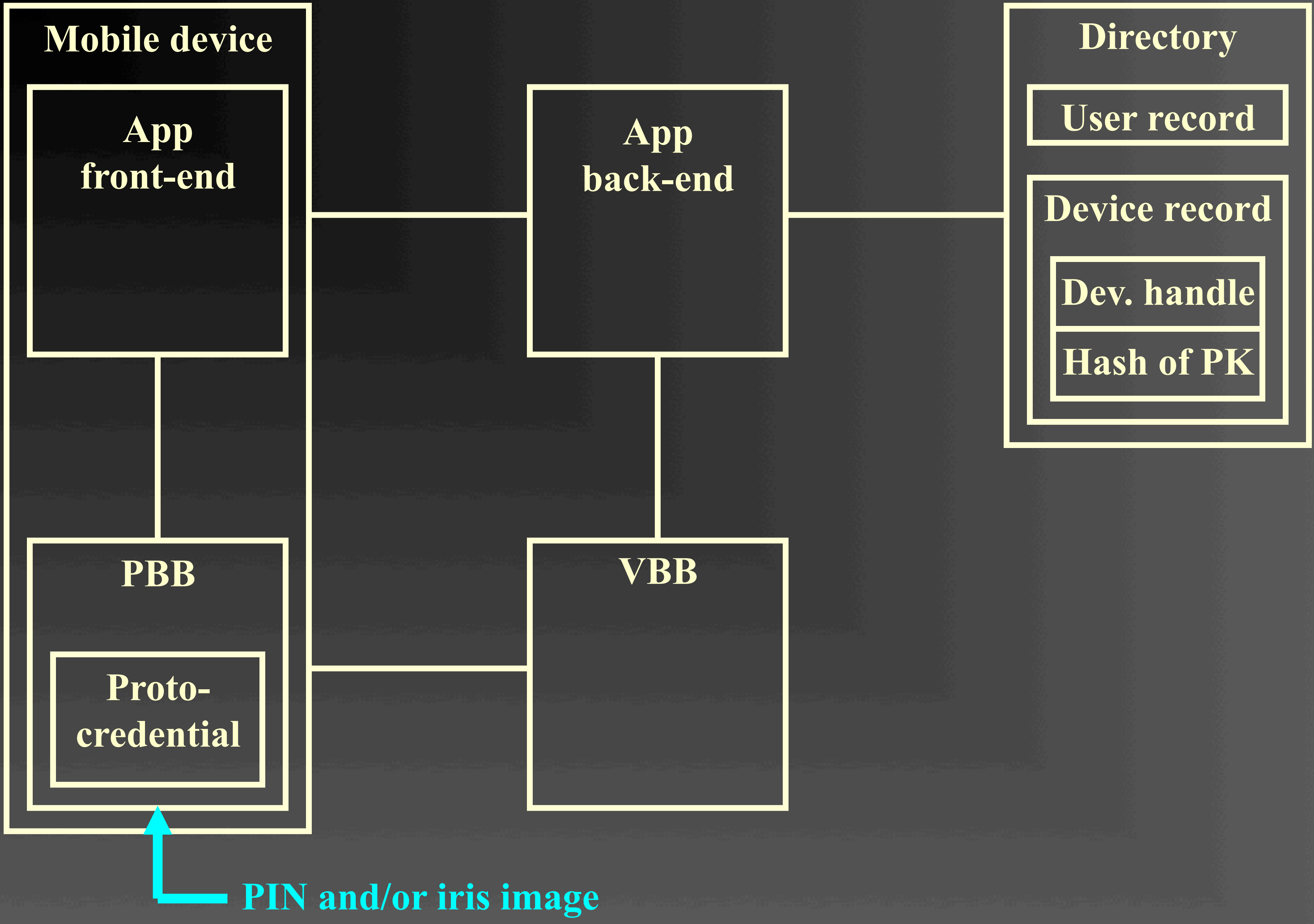

Encrypted auxiliary data
Proto-credential
Device handle
Database
Device record
Device handle
p, q
Public key
Key pair regeneration
Hash function
Hash of public key
Proof of knowledge of private key
Biometric key
User record
Decryp-tion
Biometric key generation
Aux. data
PIN
Iris sample

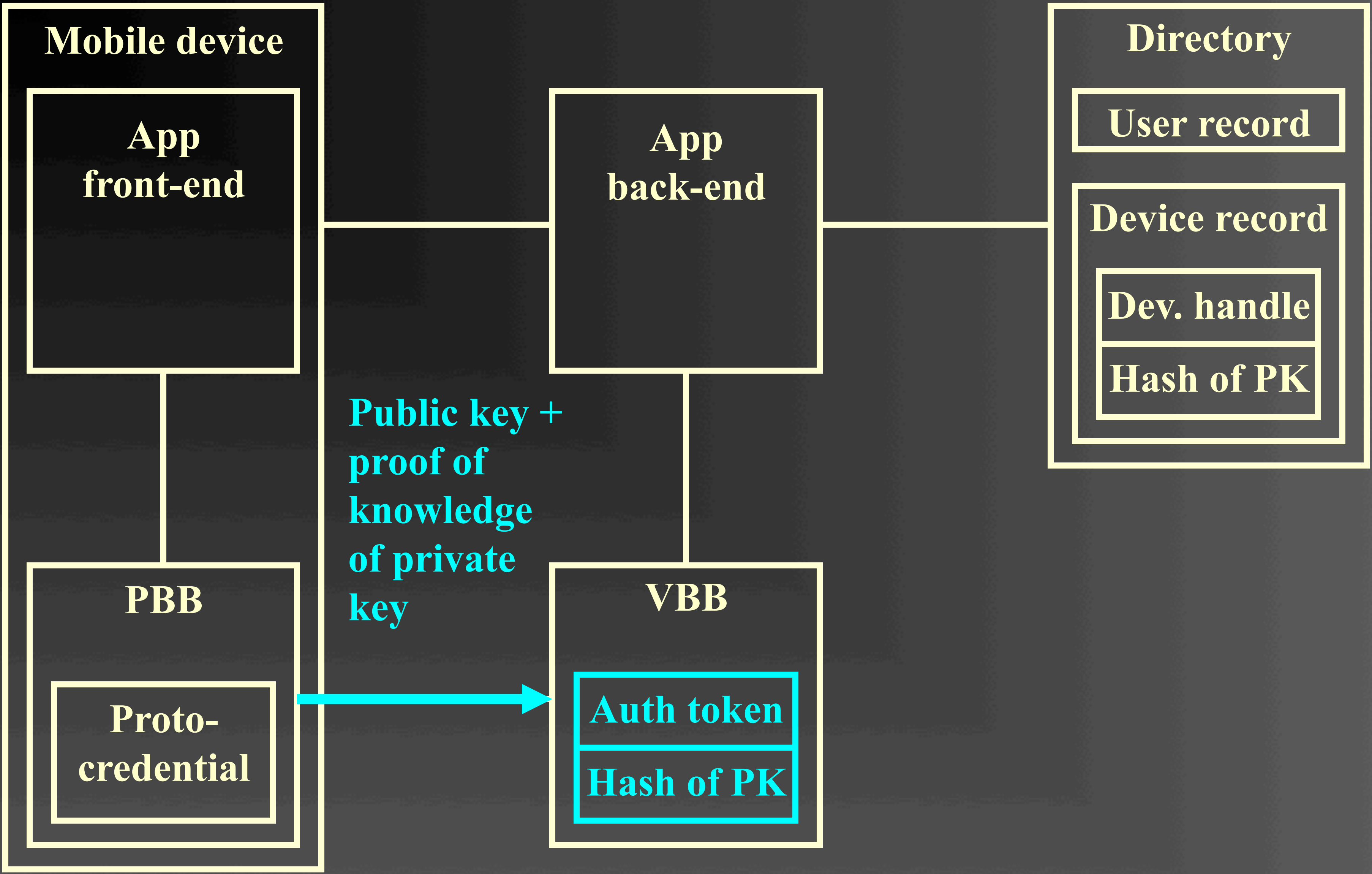
4. Encapsulation of Cryptographic and Biometric Software

- Prover Black Box (PBB) in mobile device
  - Obtains PIN and optional iris image, regenerates key pair
- Verifier Black Box (VBB) online
  - Verifies proof of knowledge of private key
- App developer does not have to know cryptography or biometrics
- Many configurations options
  - PBB: in OS / in app / separate app / browser plug-in
  - VBB: in app back-end / server appliance / trusted 3rd party



Native front-end,
native PBB,
generic VBB


Mobile device
App
front-end
PBB
Proto-
credential
App
back-end
VBB
Directory
User record
Device record
Dev. handle
Hash of PK
PIN and/or iris image



Mobile device

App front-end

PBB

Proto-credential

App back-end

VBB

Auth token

Hash of PK

Public key + proof of knowledge of private key

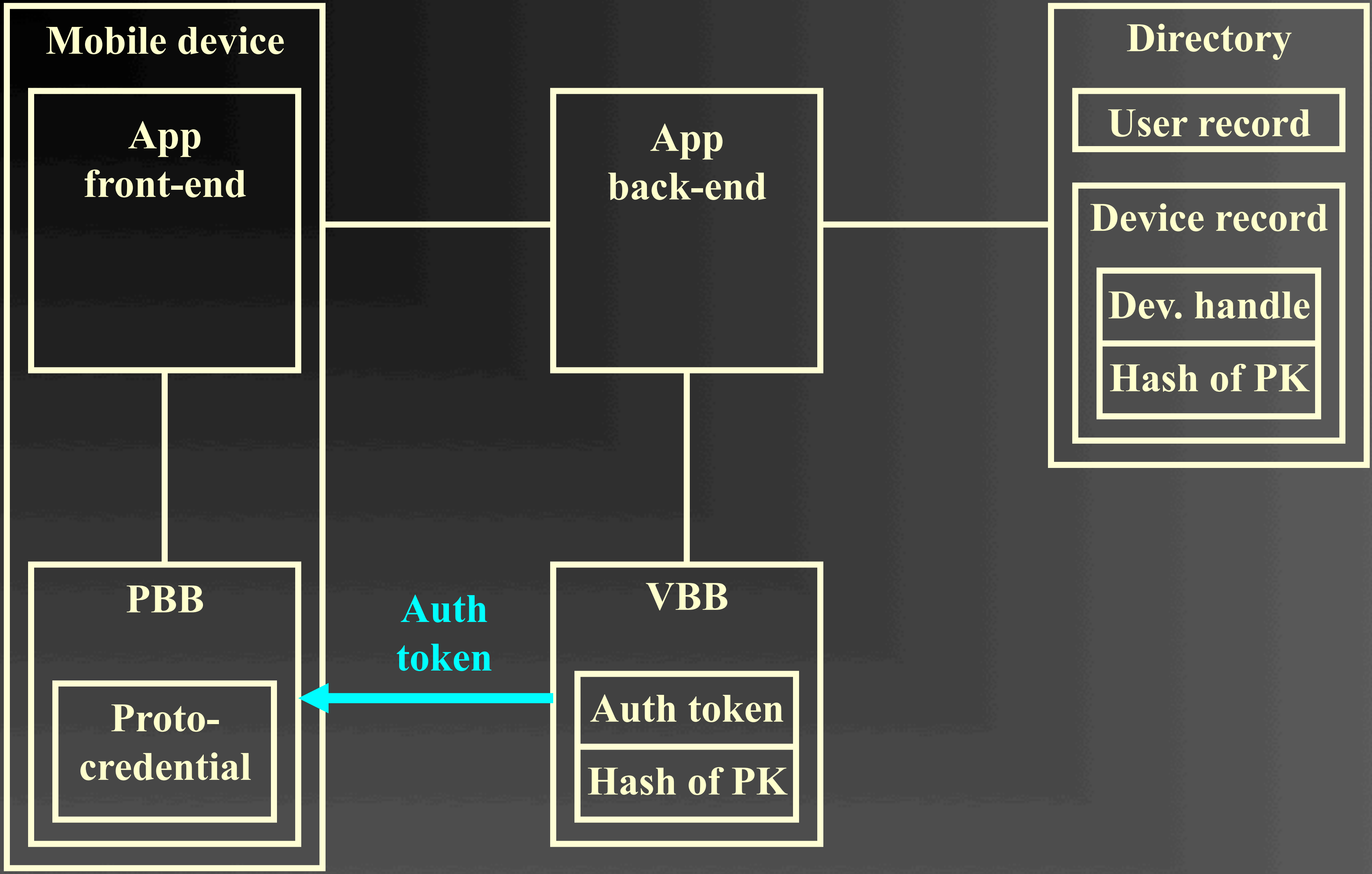
Directory

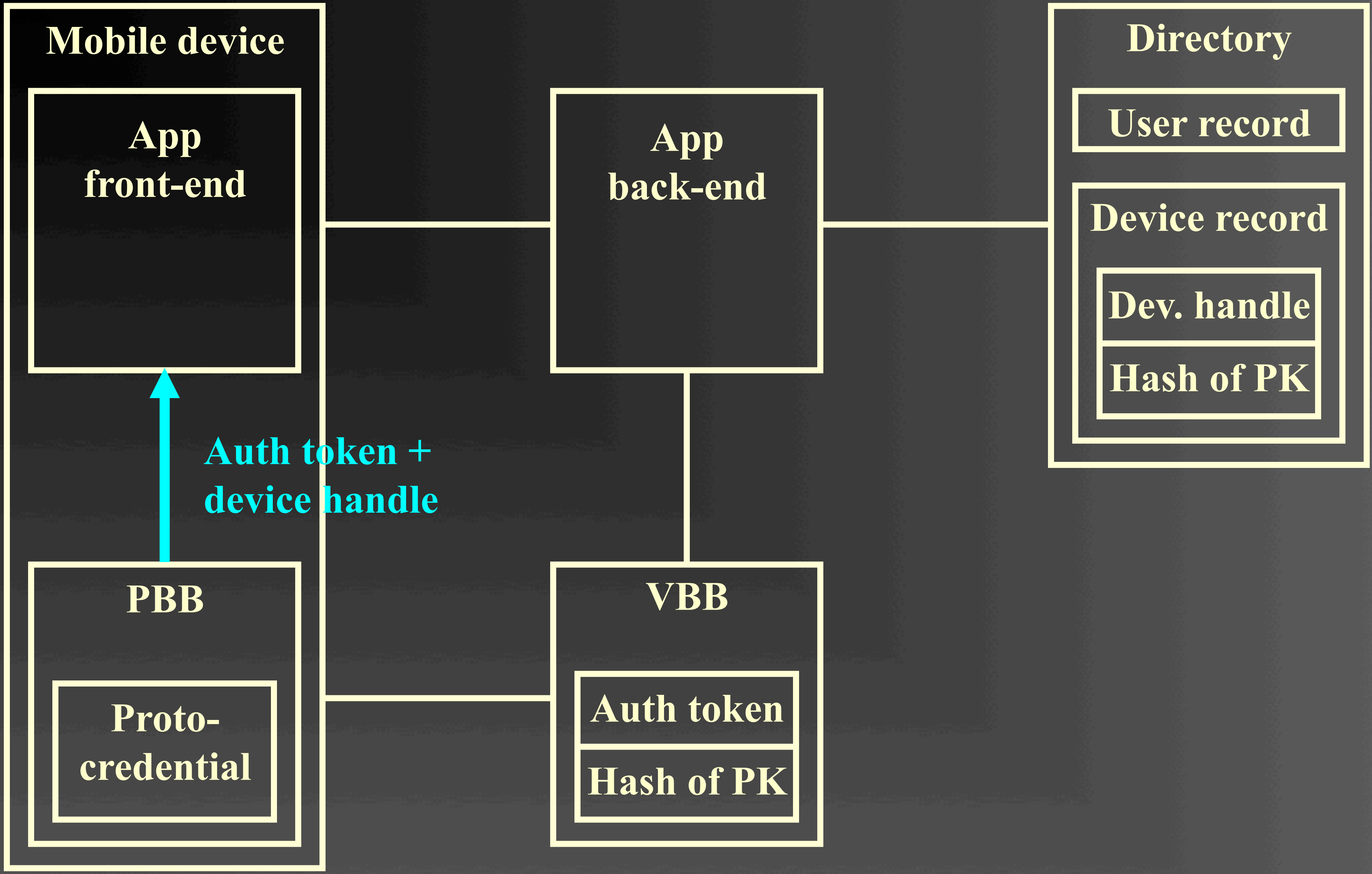
User record

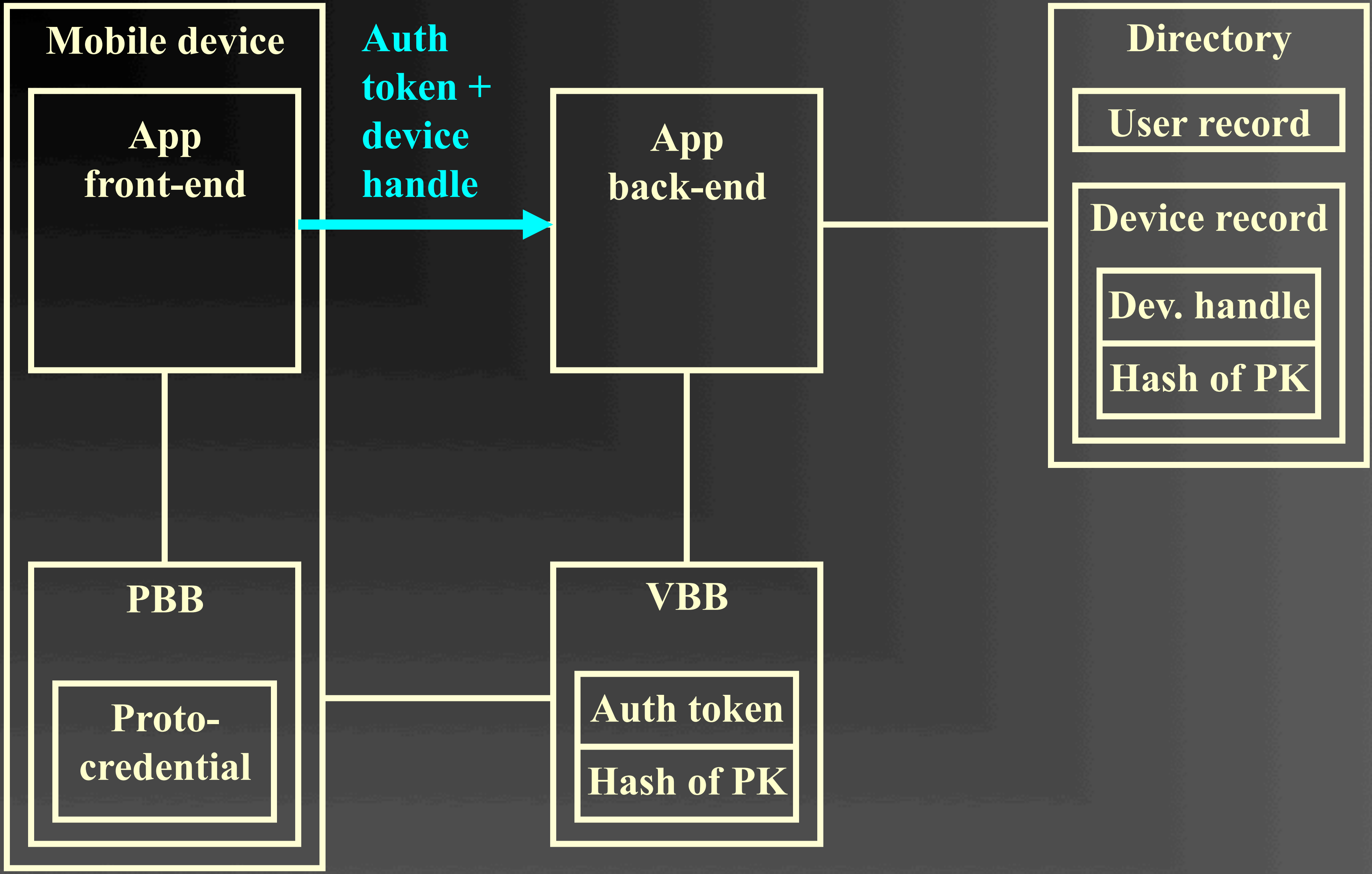
Device record

Dev. handle

Hash of PK


Mobile device
App
front-end
PBB
Proto-
credential
Auth
token
App
back-end
VBB
Auth token
Hash of PK
Directory
User record
Device record
Dev. handle
Hash of PK


Mobile device
App
front-end
Auth token +
device handle
PBB
Proto-
credential
App
back-end
VBB
Auth token
Hash of PK
Directory
User record
Device record
Dev. handle
Hash of PK



Mobile device

App front-end

Auth token + device handle

App back-end

Directory

User record

Device record

Dev. handle

Hash of PK

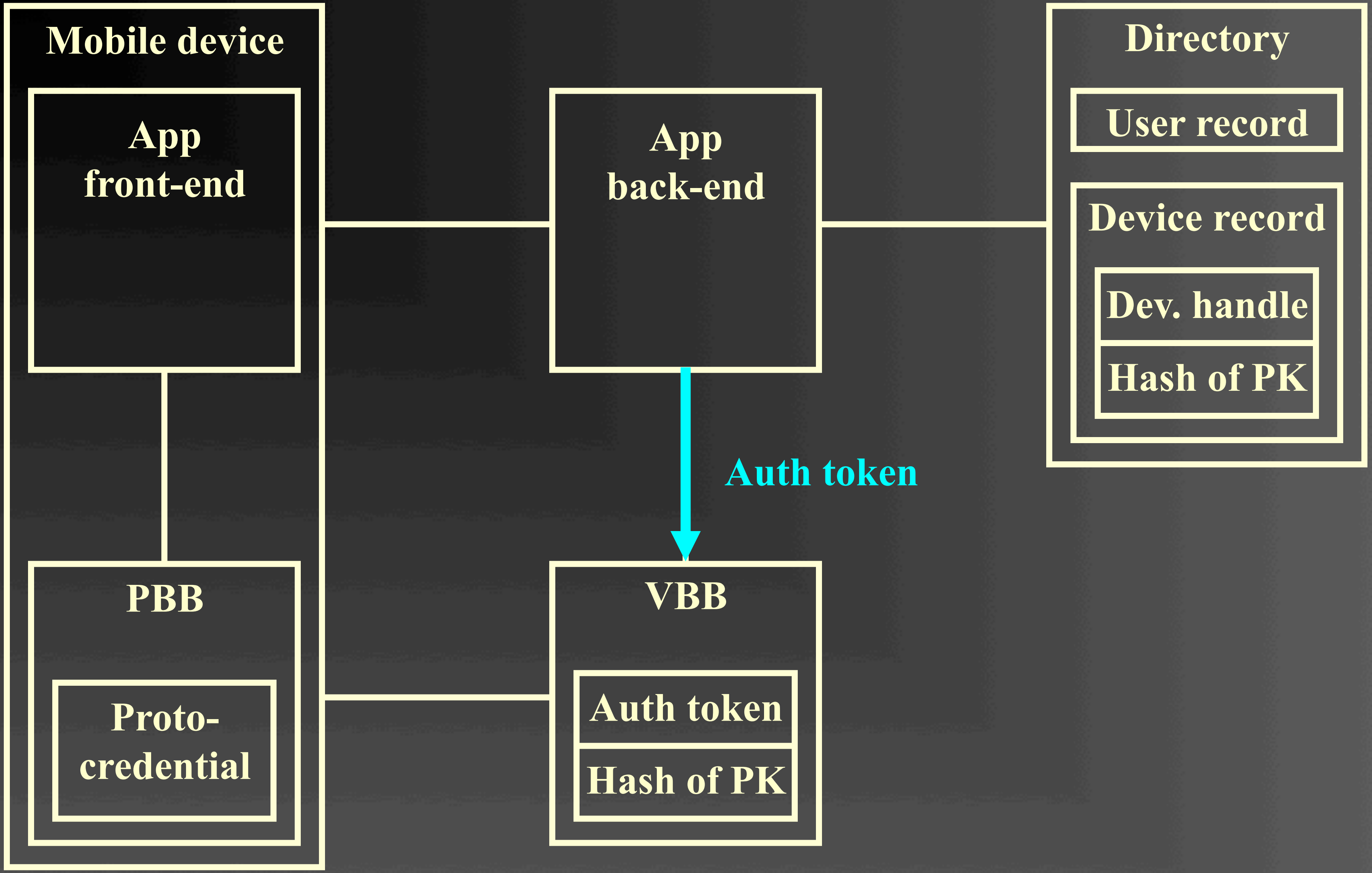
PBB

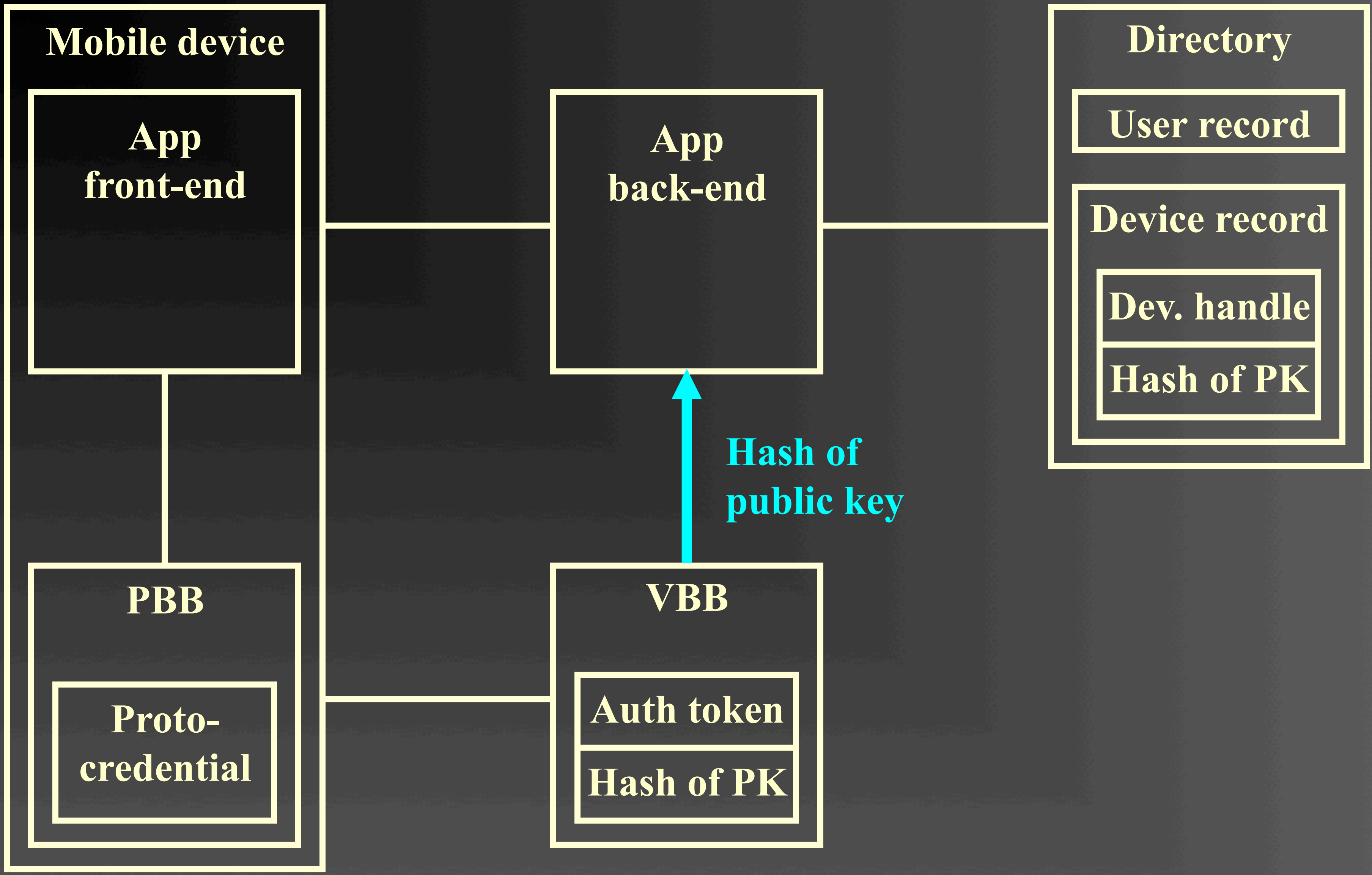
Proto-credential

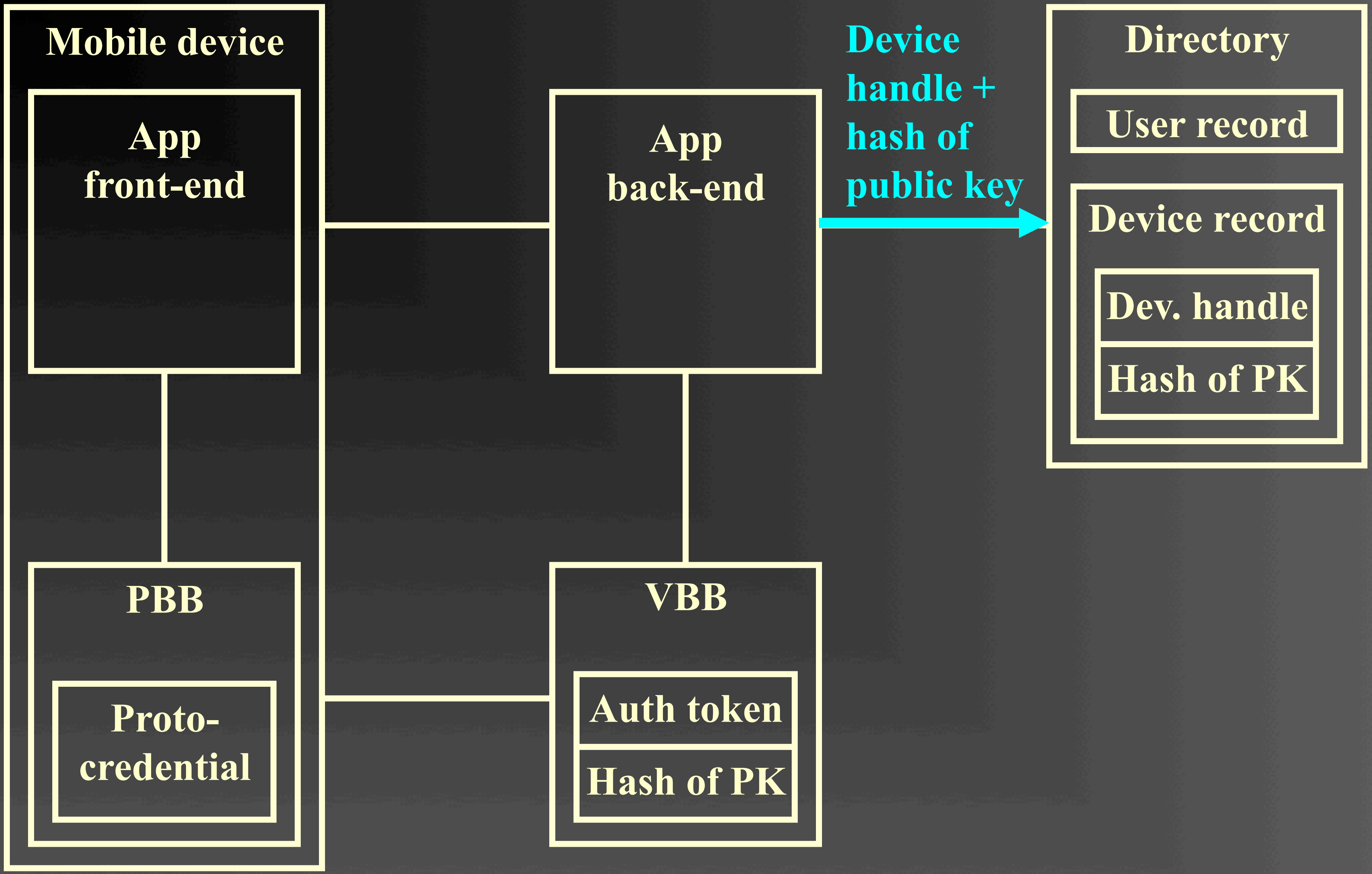
VBB

Auth token

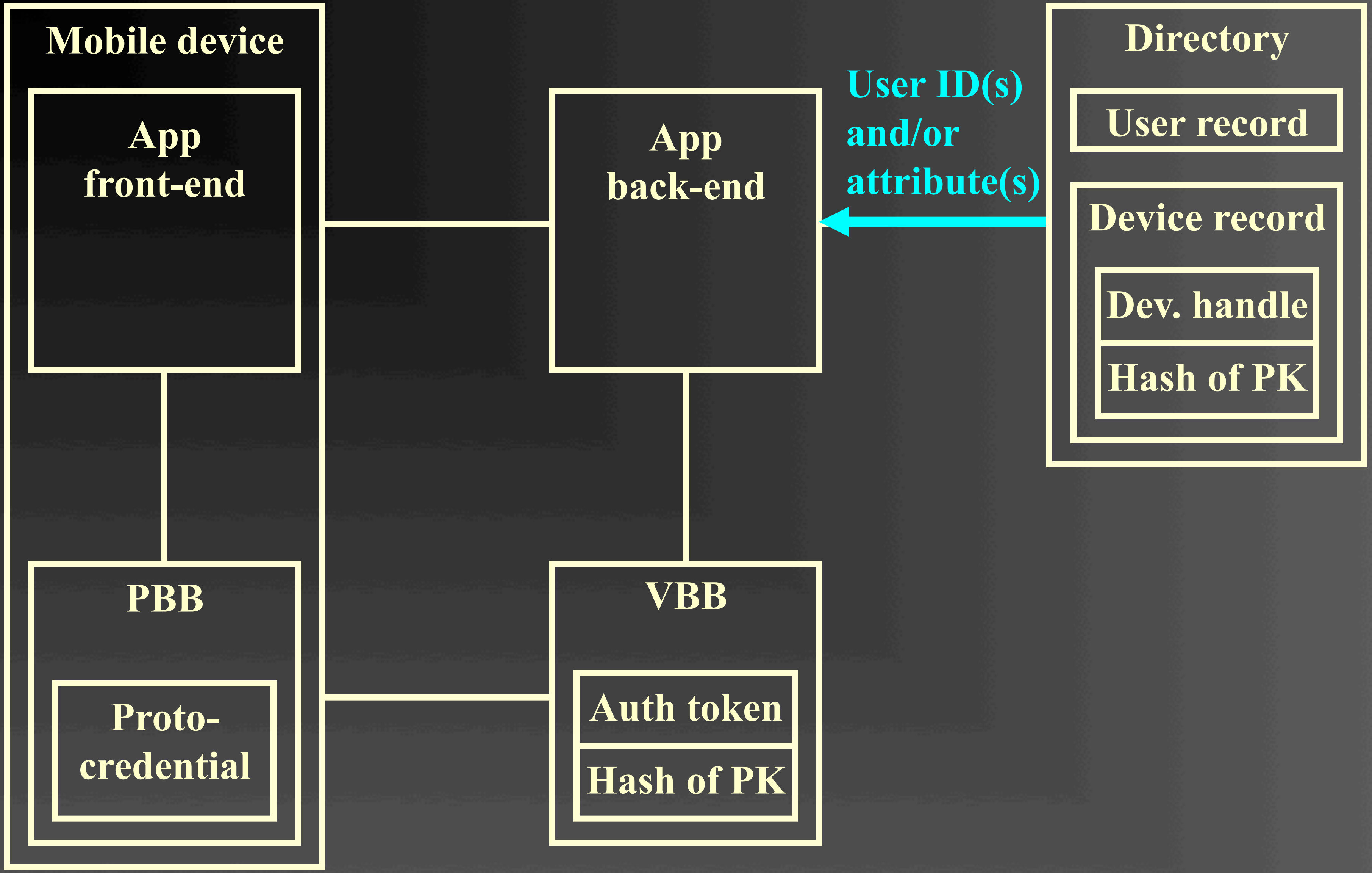
Hash of PK


Mobile device
App
front-end
PBB
Proto-
credential
App
back-end
Auth token
VBB
Auth token
Hash of PK
Directory
User record
Device record
Dev. handle
Hash of PK


Mobile device
App
front-end
PBB
Proto-
credential
App
back-end
Hash of
public key
VBB
Auth token
Hash of PK
Directory
User record
Device record
Dev. handle
Hash of PK



Mobile device
App front-end
PBB
Proto-credential
App back-end
VBB
Auth token
Hash of PK
Device handle + hash of public key
Directory
User record
Device record
Dev. handle
Hash of PK



Mobile device

App front-end

PBB

Proto-credential

App back-end

VBB

Auth token

Hash of PK

User ID(s) and/or attribute(s)

Directory

User record

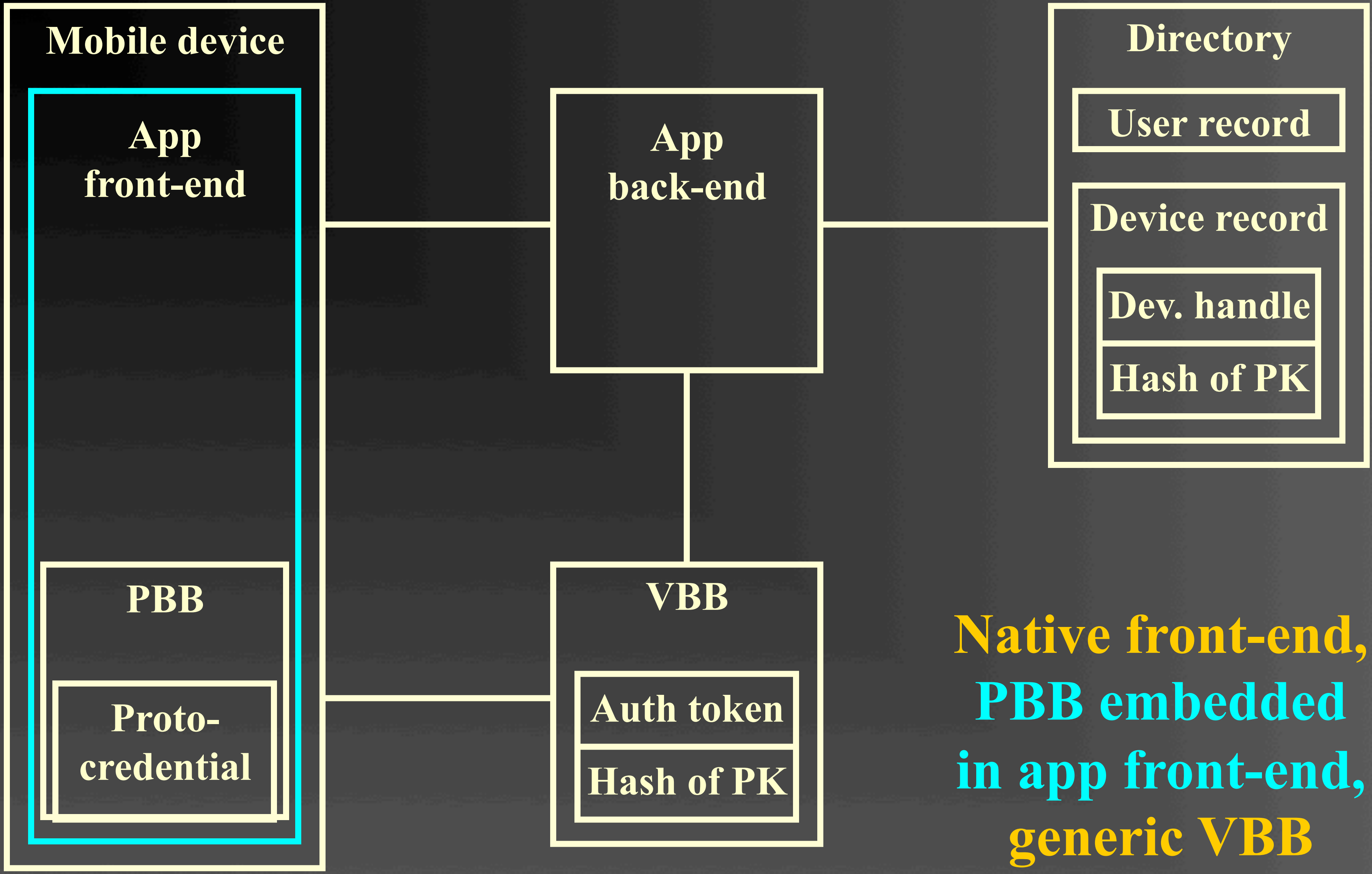
Device record

Dev. handle

Hash of PK

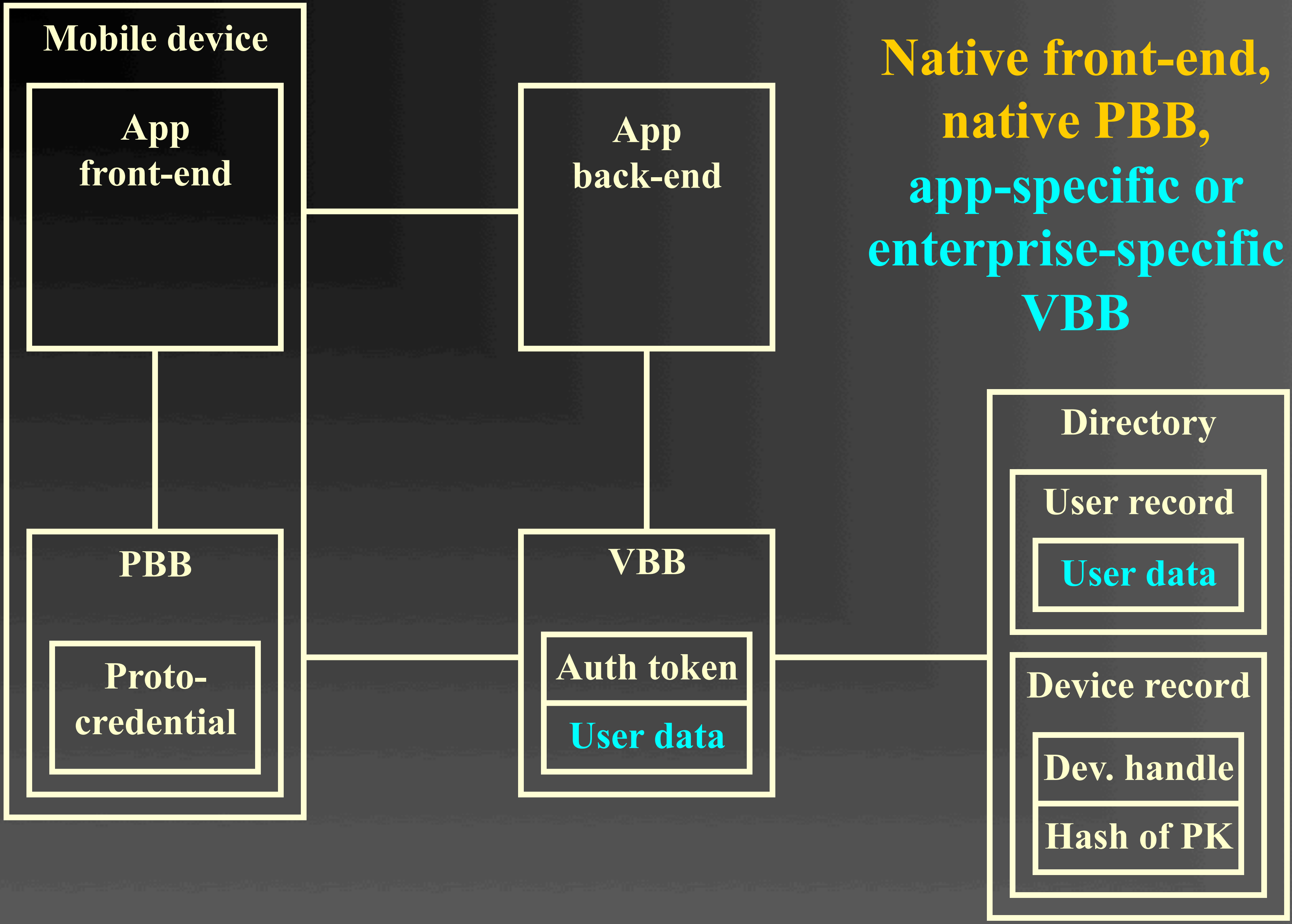
Many Possible Configurations

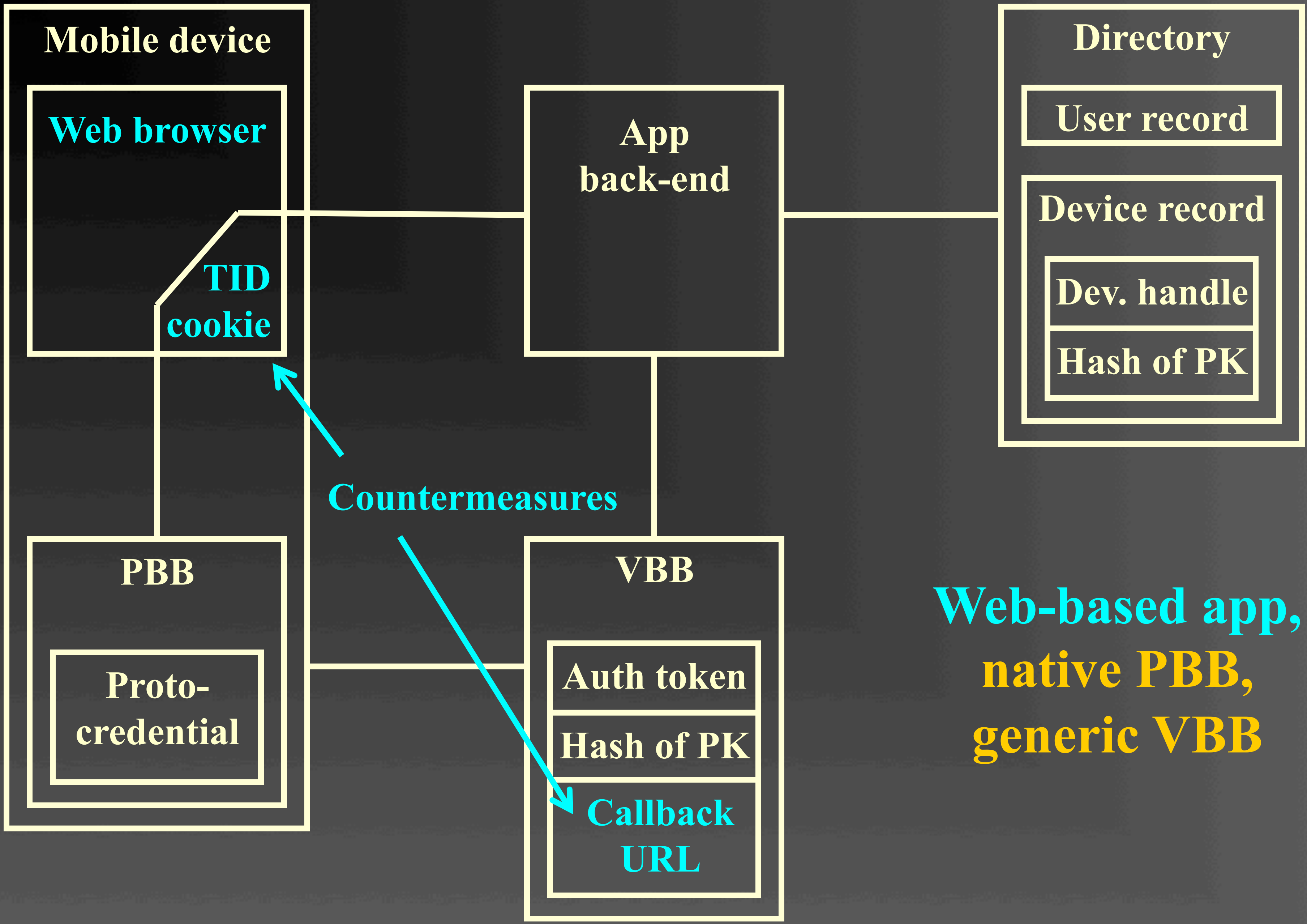
- App
  - May have native front-end (as shown), or
  - May be accessed through a web browser
- PBB
  - One credential for multiple apps
  - Different credentials for different apps
  - May be embedded in application front-end
  - Browser plug-in → works on desktops and laptops
- VBB
  - May be a generic server appliance
  - May be app- or enterprise-specific, and access the directory / database



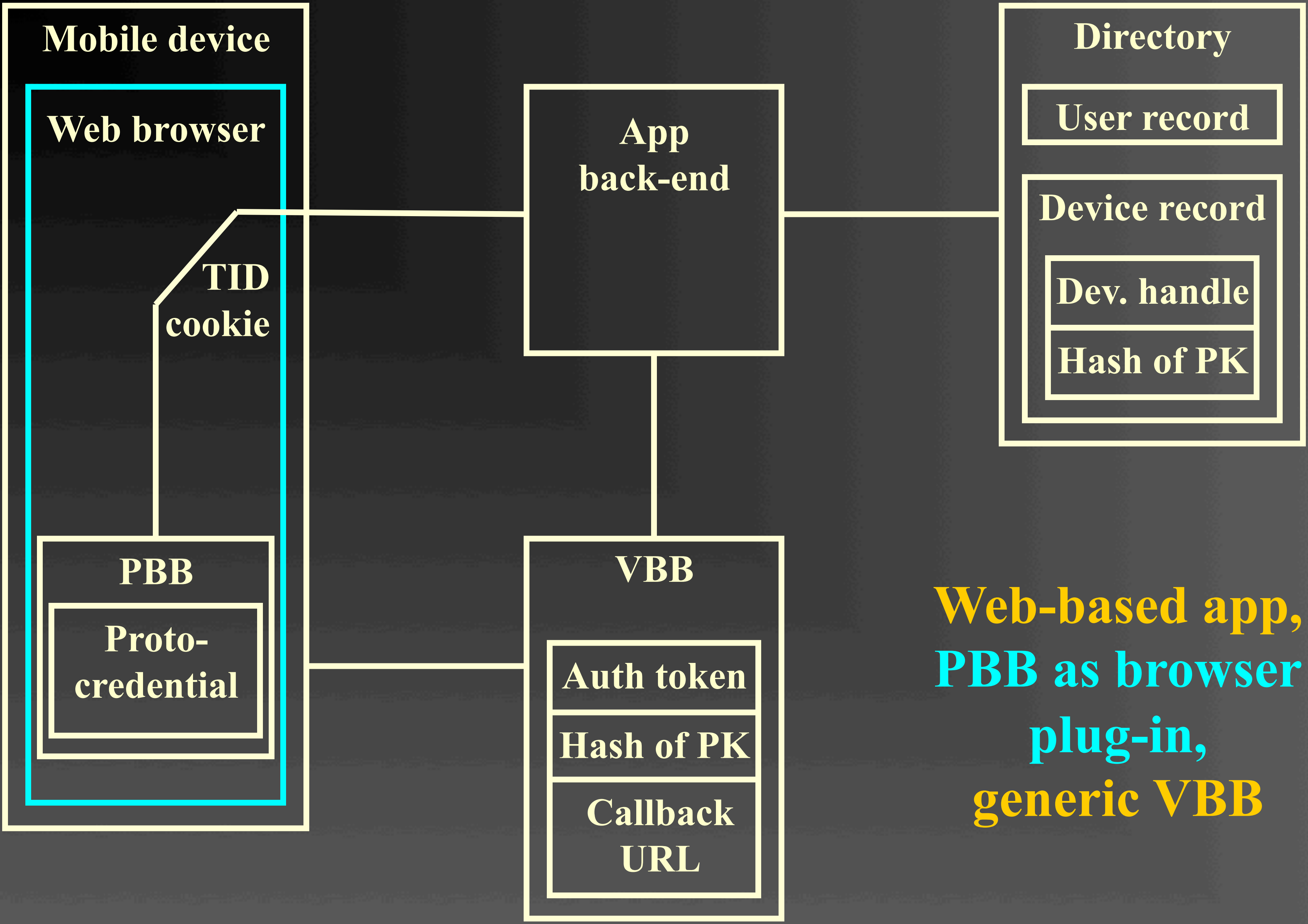
Native front-end, PBB embedded in app front-end, generic VBB

Native front-end, native PBB, app-specific or enterprise-specific VBB



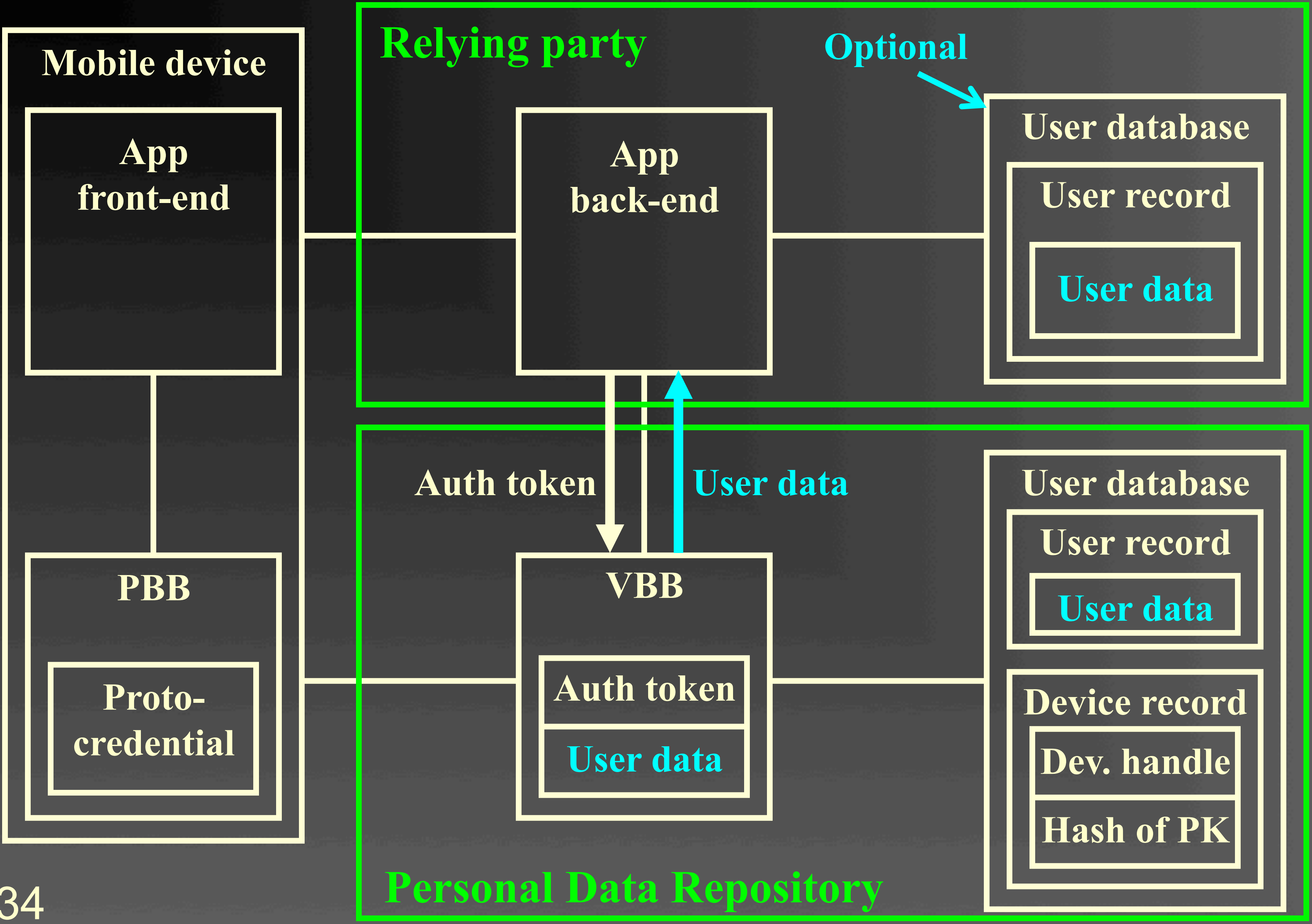


Web-based app, native PBB, generic VBB

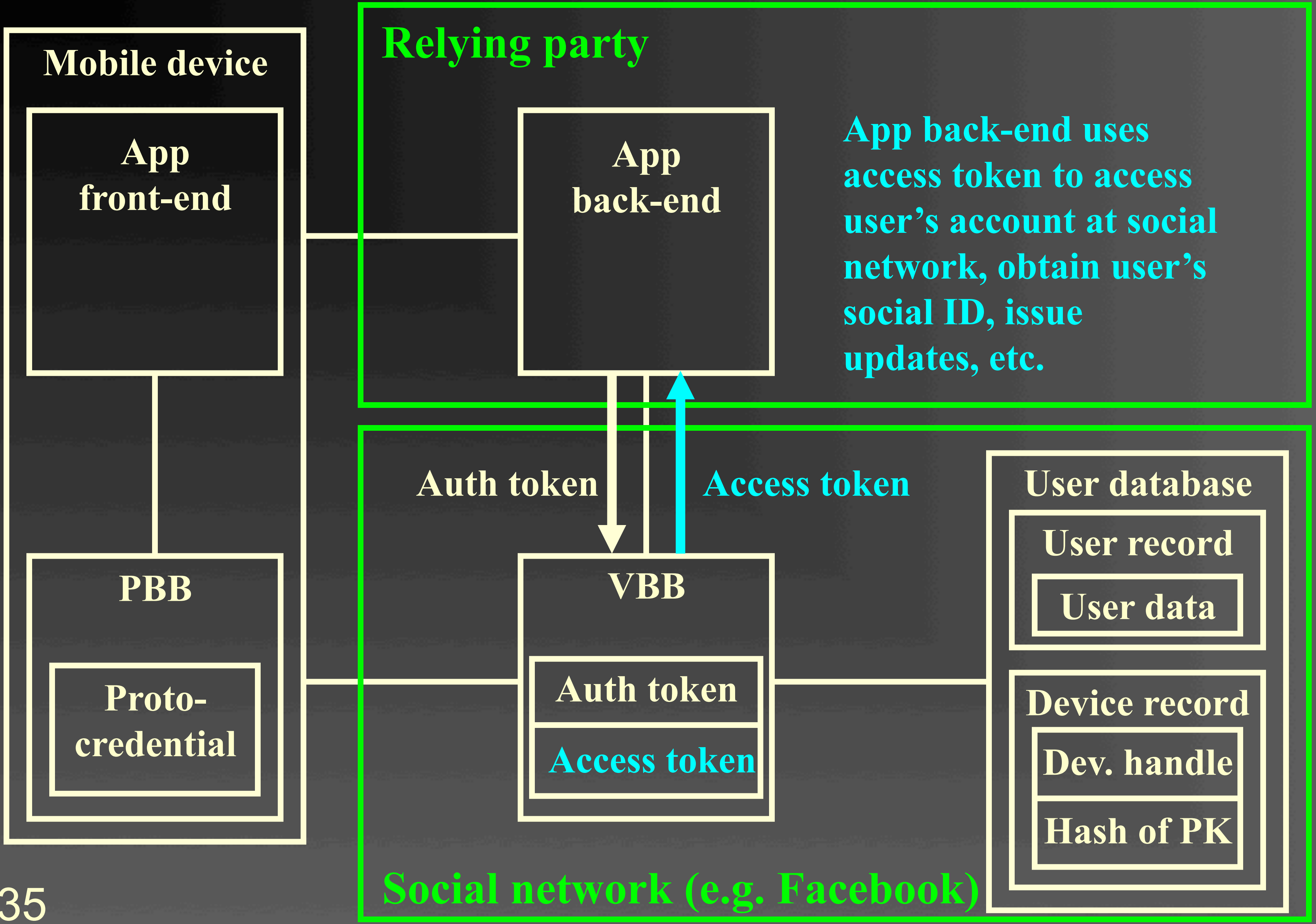


Web-based app,
PBB as browser
plug-in,
generic VBB

Third-Party Personal Data Repository



Social login without passwords



App back-end uses access token to access user’s account at social network, obtain user’s social ID, issue updates, etc.

Data Protection Challenge

- Problem: how to protect data stored in mobile device that is lost or stolen
  - Encrypt data?
    - Not secure if data encryption key is stored in device without tamper protection
  - Data encryption key derived from PIN?
    - Not secure because PIN is vulnerable to offline attack
  - Hardware key + PIN, as in iPhone?
    - Not secure because custom code can use the hardware key to crack the passcode
- Our authentication methods based on key-pair regeneration provide a solution

Solution

- Data encryption key stored in trusted server (or split over multiple servers with *k*-of-*n* Shamir secret sharing)
- To unlock phone and decrypt data, user authenticates to server(s) and obtains the data encryption key
- Trusted server(s) could be provided by
  - Mobile network operator, or
  - OS provider, or
  - Mobile device manufacturer, or
  - Mobile device manager, or
  - Ad-hoc data protection service trusted by user

For more information…

- Whitepapers
  - http://pomcor.com/whitepapers/MobileAuthentication.pdf
  - http://pomcor.com/whitepapers/DataProtection.pdf
- Recent blog posts at
  - http://pomcor.com/blog/
- Write to
  - fcorella@pomcor.com
  - kplewison@pomcor.com