BTI 4201: Secret sharing, symmetric key management

Christian Grothoff

BFH

9.6.2023

Learning Objectives

Key management

Shamir Secret Sharing

Key escrow and recovery: From Shamir to Anastasis

Threshold Signatures

Part I: Key management

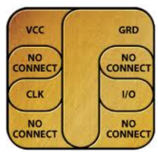
Software based Personal Security Environments (PSE): PKCS#12

PKCS#12 is the most common format for software PSEs:

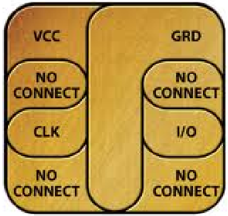
- PKCS#12 is a file container format used for storage and transport of private keys (and possibly certificates).
- Information is protected with a password-based symmetric key (e.g. a password).
- The security of a software PKCS#12 is based on the strength of the password protecting it.

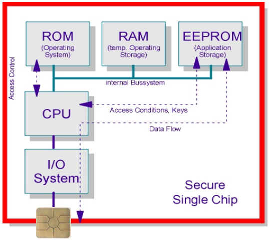
Problem: A PKCS#12 soft-token may be copied unnoticed.

Smartcards and Cryptotokens



Properties of Crypto-tokens/cards





- Crypto-cards have the ability of a secure container for secret data and have an executive platform for cryptographic algorithms.
- A Crypto-card looks like a "Black Box" from the outside, where some operations can only be used over a very restrictive hard- and software interface which is able to enforce specific security policies.
- Access to sensitive data areas (i.e. private keys) is physically "impossible" from the outside.

Example: Yubikey and Personal Identity Verification (PIV)

- Yubikey provides Smart Card functionality based on the Personal Identity Verification (PIV) interface specified in NIST SP 800-73.
- Yubikeys perform RSA or ECC sign/decrypt operations using a private key stored on the token, through common interfaces such as PKCS#11.
- Supported key sizes: RSA 2048 or ECC 256/384.
- The "universal smartcard minidriver" provides "standard smart" functionality and additional certificate and PIN management features.
- Special Yubikeys obtained FIPS 140-2 security level certification.

Hardware Security Modules (HSM)

Common functionality:

- Secure storing and use of keys
- Random number generator
- Key pair generation
- Digital signing
- Key archiving
- Acceleration for crypto schemes

Should protect keys against:

- Mechanical attacks
- Temperature attacks
- Manipulation of voltage
- Chemical attacks

Part II: Shamir Secret Sharing

Problem: Availability (1/3)

If you give one person a secret, it may get lost.

Problem: Availability (1/3)

If you give one person a secret, it may get lost.

$\Rightarrow$ So give it to more than one person!

Problem: Confidentiality (2/3)

If you give many entities a secret, it may get disclosed.

Problem: Confidentiality (2/3)

If you give many entities a secret, it may get disclosed.

$\Rightarrow$ So give them only a key share!

Problem: Scalability (3/3)

If you want k out of n entities to coordinate to recover a secret, there are

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \tag{1}$$

combinations to consider.

Problem: Scalability (3/3)

If you want k out of n entities to coordinate to recover a secret, there are

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \tag{1}$$

combinations to consider.

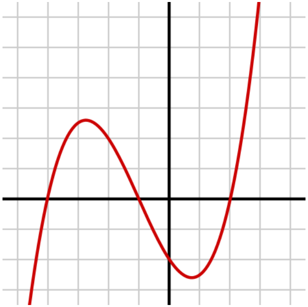
$\Rightarrow$ Use polynominals!

Polynominals

A polynominal of degree k is fully determined by $k+1$ data points

$$(x_0, y_0), \ldots, (x_j, y_j), \ldots, (x_k, y_k),$$

where no two x_j may be identical.



Lagrange Interpolation

The interpolation polynominal in the Lagrange form is:

$$L(x) := \sum_{j=0}^{k} y_j \ell_j(x)$$

where

$$\ell_j(x) := \prod_{\substack{0 \le m \le k \\ m \ne j}} \frac{x - x_m}{x_j - x_m} = \frac{(x - x_0)}{(x_j - x_0)} \cdots \frac{(x - x_{j-1})}{(x_j - x_{j-1})} \frac{(x - x_{j+1})}{(x_j - x_{j+1})} \cdots \frac{(x - x_k)}{(x_j - x_k)} \qquad (2)$$

for $0 \le j \le k$.

Practical Considerations

- Our secrets will typically be integers. Calculations with floating points are *messy*.

$\Rightarrow$ Use finite field arithmetic, not $\mathbb{R}$.

Real world scalability

n / k	1	2	3	4	5	6
1	1	2	3	4	5	6
2		1	3	6	10	15
3			1	4	10	20
4				1	5	15
5					1	6
6						1

Other values:

- $\binom{10}{5} = 252$
- $\binom{20}{10} = 184756$
- $\binom{30}{15} = 155117520$
- $\binom{100}{50} \approx 10^{29}$

Scalability Problem?

How many people do you have to share your secrets with?

How many people realistically participate in recovery?

Part III: Anastasis[1]

[1] `https://anastasis.lu/`, based on a BFH Bachelor's thesis by D. Neufeld and D. Meister (2020)

THE PROBLEM

Confidentiality requires only consumer is in control of key material

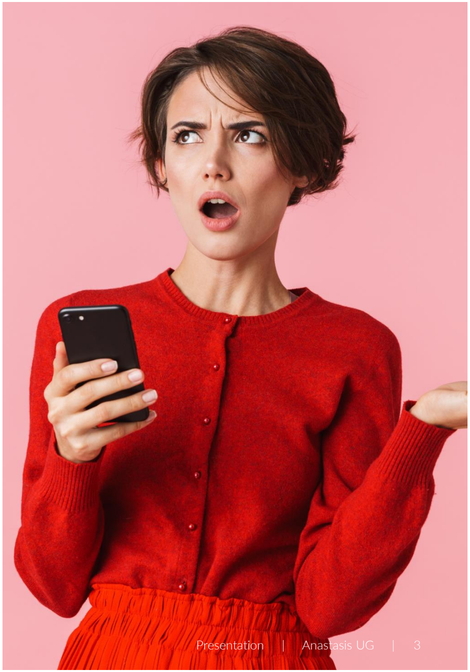
Consumers are unable to simultaneously ensure confidentiality and availability of keys

Cryptographic key-splitting solutions so far are not usable

European e-money issuers using electronic wallets must:[1]

- Enable consumers to always recover their electronic funds (i.e. if devices are lost)
- Not assume consumers are able to remember or securely preserve key material

[1] According to communication from ECB to Taler Systems SA.



THE PROBLEM

Confidentiality requires only consumer is in control of key material

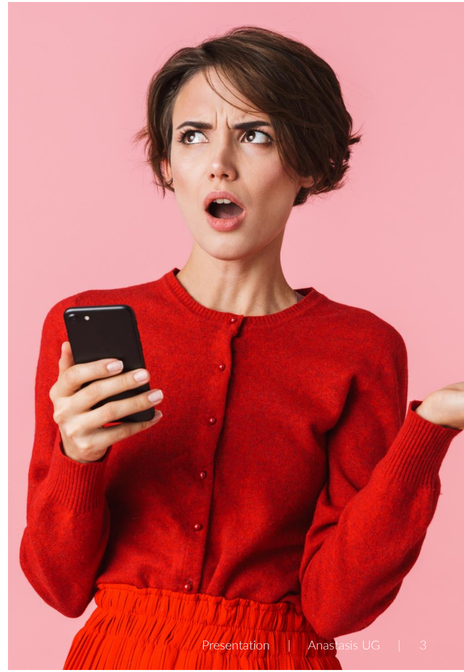
Consumers are unable to simultaneously ensure confidentiality and availability of keys

Cryptographic key-splitting solutions so far are not usable

European e-money issuers using electronic wallets must:[1]

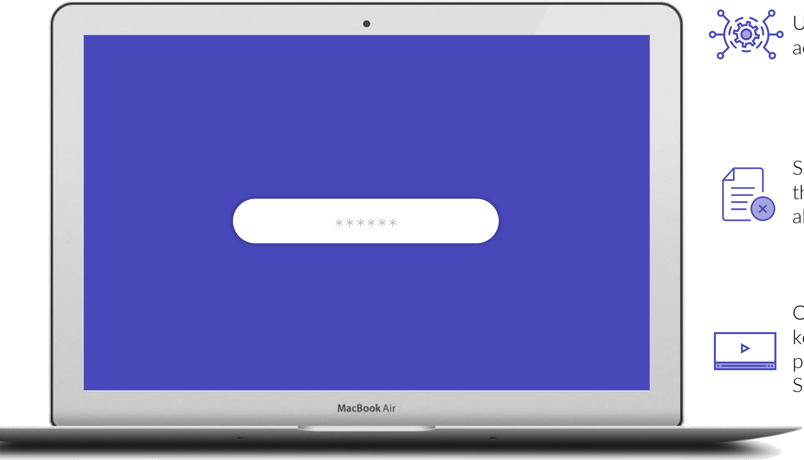
- Enable consumers to always recover their electronic funds (i.e. if devices are lost)
- Not assume consumers are able to remember or securely preserve key material

[1] According to communication from ECB to Taler Systems SA.



WHAT IS ANASTASIS?

ANASTASIS IS A KEY RECOVERY SERVICE.

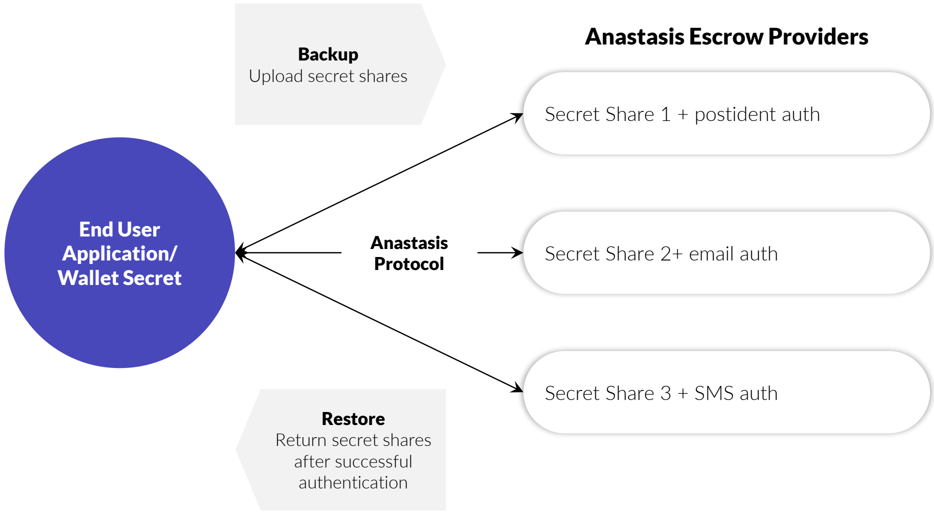


Users split their secret keys across multiple service providers

Service providers learn nothing about the user, except possibly some details about how to authenticate the user

Only the authorized user can recover the key by following standard authentication procedures (SMS TAN, Video-Ident, Security Question, eMail, etc.)

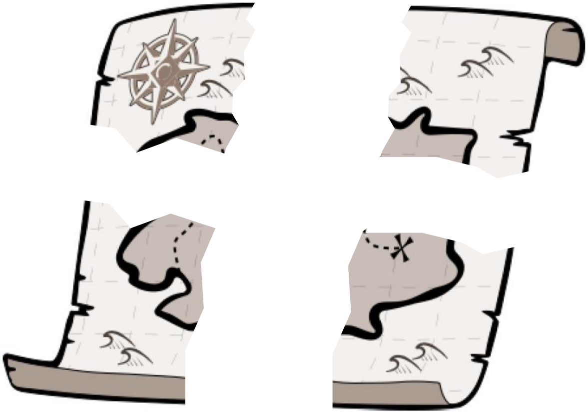
OVERVIEW


Backup
Upload secret shares
Anastasis Escrow Providers
Secret Share 1 + postident auth
End User
Application/
Wallet Secret
Anastasis
Protocol
Secret Share 2+ email auth
Secret Share 3 + SMS auth
Restore
Return secret shares
after successful
authentication

SIMPLIFIED PROCESS FLOW
STEP 1: RECOVERY INFORMATION

SIMPLIFIED PROCESS FLOW

STEP 2: SPLIT RECOVERY INFORMATION

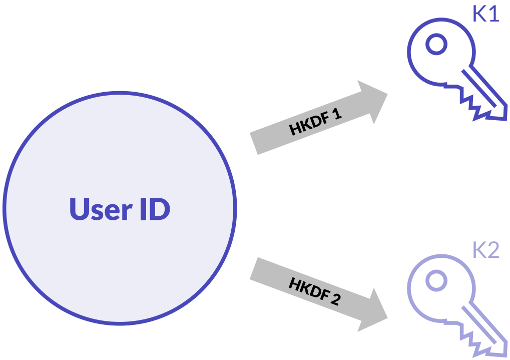


Presentation | Anastasis UG | 7

SIMPLIFIED PROCESS FLOW
STEP 3: USER IDENTIFICATION

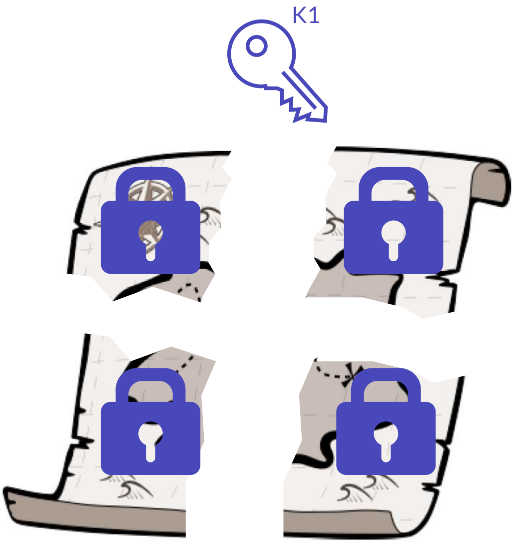
SIMPLIFIED PROCESS FLOW

STEP 4: KEY DERIVATION



SIMPLIFIED PROCESS FLOW

STEP 5: ENCRYPT PARTS



Presentation | Anastasis UG | 10

SIMPLIFIED PROCESS FLOW

STEP 6: ADD TRUTH

+ H (answer to security question)

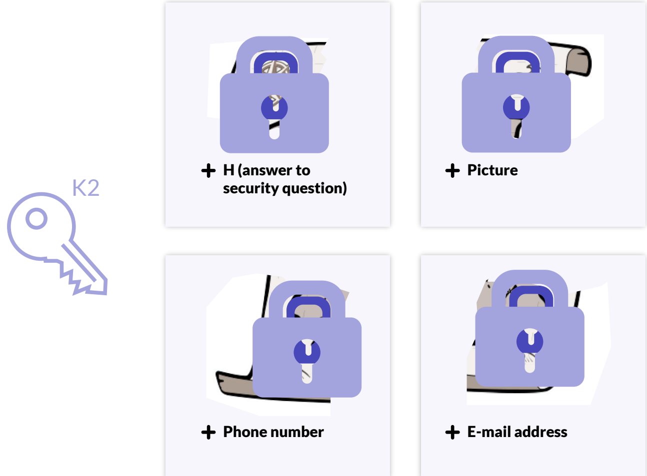
+ Picture

+ Phone number

+ E-mail address

SIMPLIFIED PROCESS FLOW

STEP 7: ENCRYPT TRUTH



SIMPLIFIED PROCESS FLOW

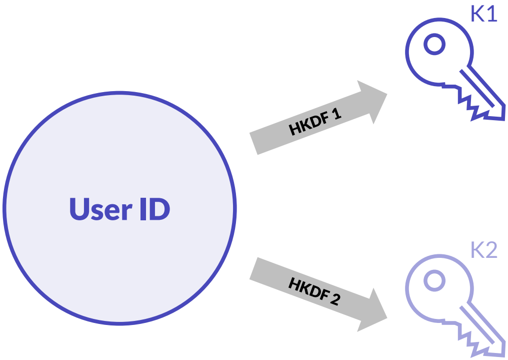
STEP 8: STORE DATA

Presentation | Anastasis UG | 13

SIMPLIFIED PROCESS FLOW

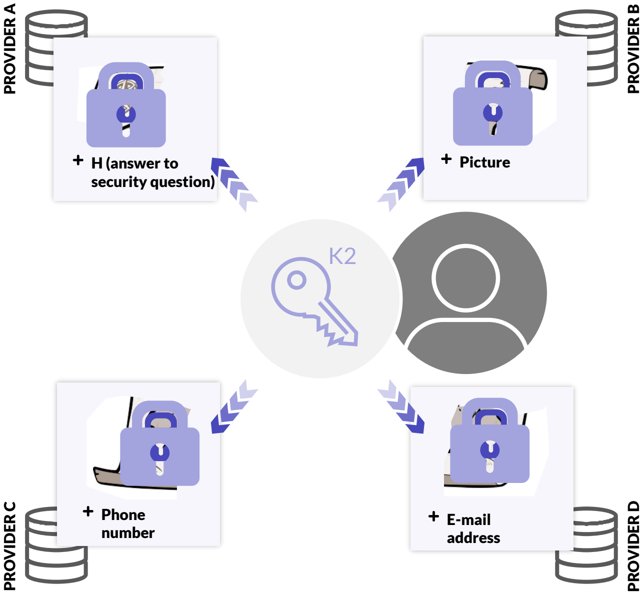
STEP 9: USER IDENTIFICATION

SIMPLIFIED PROCESS FLOW
STEP 10: KEY DERIVATION



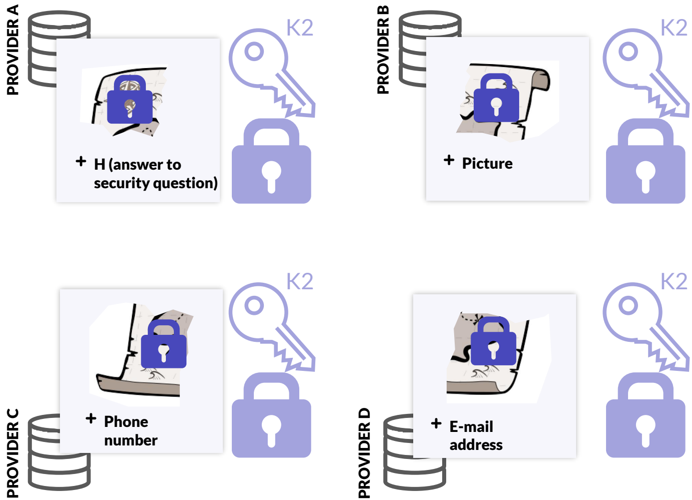
SIMPLIFIED PROCESS FLOW

STEP 11:
PROVIDE KEY



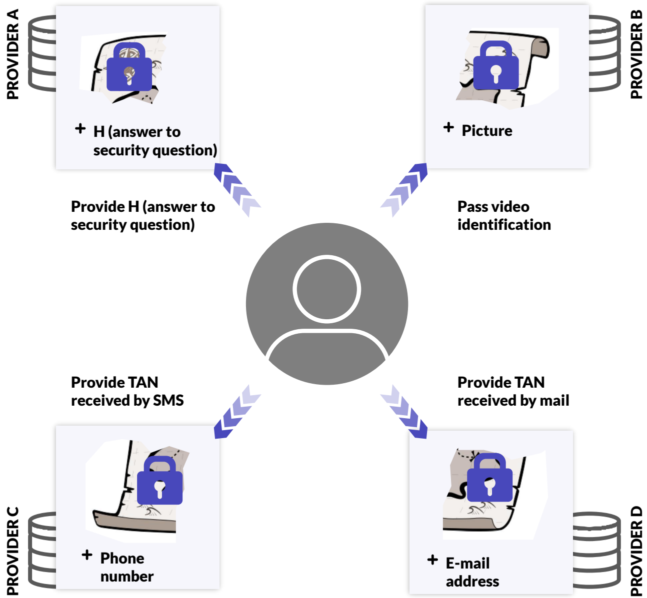
SIMPLIFIED PROCESS FLOW

STEP 12:
DECRYPT TRUTH



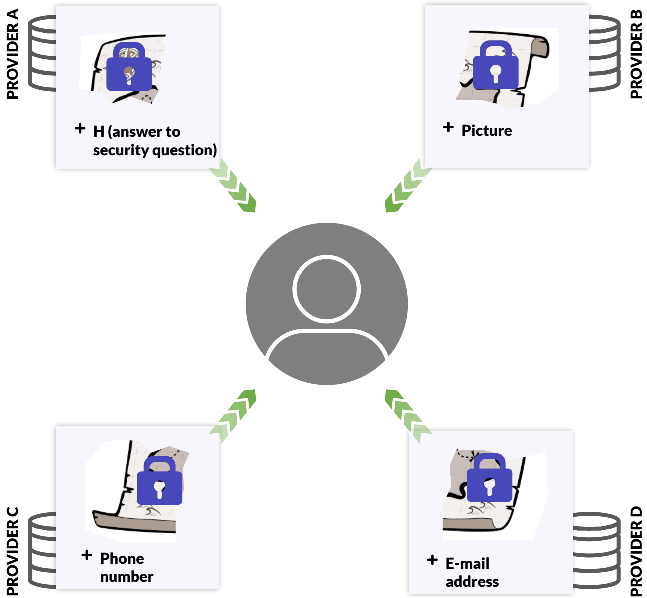
SIMPLIFIED PROCESS FLOW

STEP 13: AUTHENTICATION



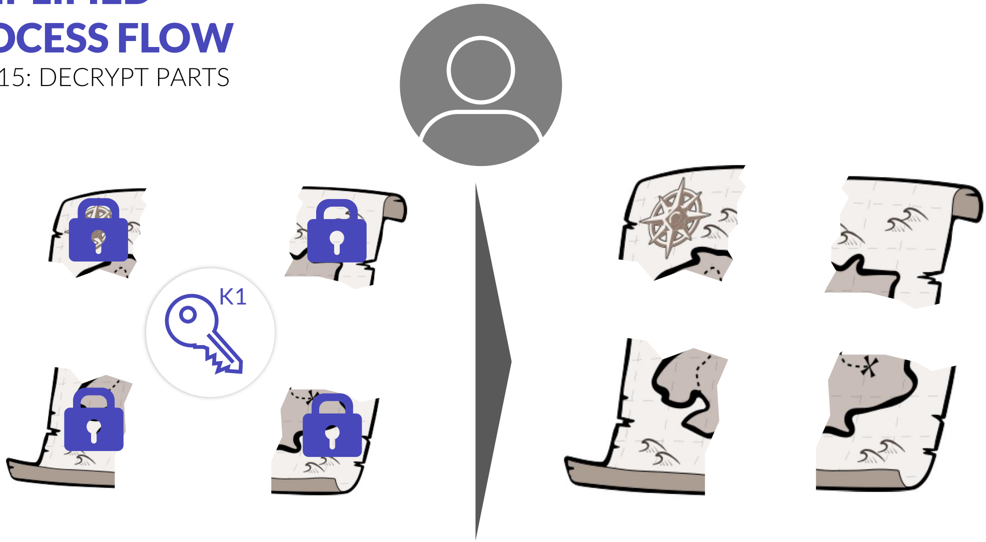
SIMPLIFIED PROCESS FLOW

STEP 14:
RECEIVE PARTS



SIMPLIFIED PROCESS FLOW

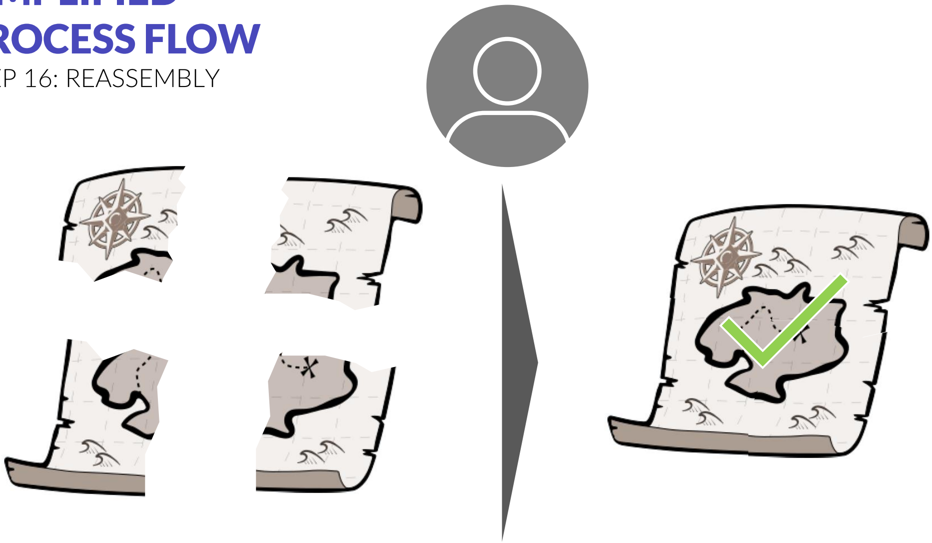
STEP 15: DECRYPT PARTS



Presentation | Anastasis UG | 20

SIMPLIFIED PROCESS FLOW

STEP 16: REASSEMBLY



Presentation | Anastasis UG | 21

SIMPLIFICATIONS

THE PREVIOUS ILLUSTRATION MAKES VARIOUS SIMPLIFICATIONS

- Policies to allow more flexible splitting than 4/4
- Recovery document to remember policies and providers
- Distinction between core secret and master secret
- Payment processing
- Provider salts
- Anti-DoS provisions in protocol / request limits
- Versioning
- Liability limitations

UNIQUE SALES PROPOSITIONS (USPS)

1 Distributed trust instead of single point of failure

2 Maximum privacy with respect to authentication data

3 Post-quantum security

4 Ease of use

5 Low cost, scalable cloud-based solution

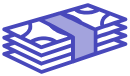
6 Transparent, Free Software solution

7 Generic API suitable for a range of applications

8 Customers can remain anonymous:
- Minimizes risk to Anastasis service provider in case database is exposed
- Makes it more difficult for attackers to fool authentication procedure

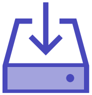
9 E-money issuer does not have to protect consumer data against its own staff and can respect consumer privacy

SOCIAL IMPACT OF ANASTASIS



Low-cost solution with minimal environmental impact

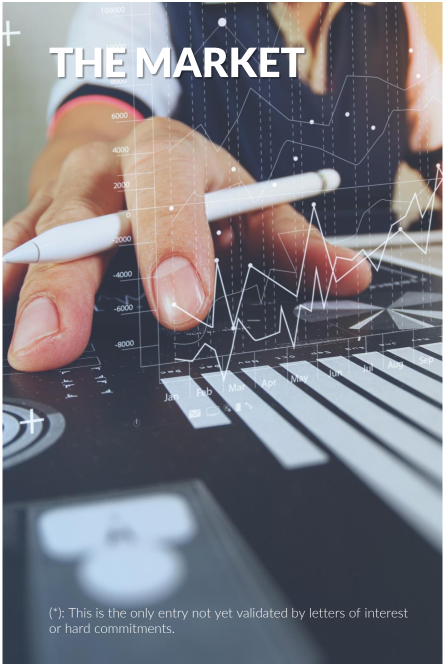
Increases informational self-determination by keeping consumers in control of their data



Free Software contributes to the global Commons


OPERATING MODEL
REVENUE
• E-money issuers pay Anastasis UG to offer service to consumers with wallets to satisfy their regulatory requirements (service must exist)
• Wallet operators pay Anastasis UG to assist with technical integration
• Consumers pay Anastasis UG for safekeeping and/or recovery (subscription)
EXPENSES
• Development and operations (staff costs)
• Server infrastructure
Presentation | Anastasis UG | 25

THE MARKET



- Electronic wallets for blockchain wallets and/or fiat currencies
- Key store for communication keys, such as OpenPGP or X.509
- Identity management solutions
- Password managers and disk encryption key material (*)

(*): This is the only entry not yet validated by letters of interest or hard commitments.

MAIN RISKS AND MITIGATIONS

1 **IMPLEMENTATION RISK**
Straightforward design simplifies work

2 **INFORMATION SECURITY RISK**
Privacy-by-design minimizes loss

3 **DISTRIBUTION ON CUSTOMER SIDE**
Strong partners with implementation need

4 **CASH FLOW**
Cloud-based deployment with outsourcing of procedures that amortize only at scale

5 **USABILITY**
Will work with UX expert

Demo

Part IV: FROST & Frosix[2]

[2] Based on work by Joel Urech

Everything is Broken

Alice wants to create a cryptographic signature, but:

- No single piece of hardware is trusted
- No single service provider is trusted

But: Using t independent signature service providers might be ok!

If we need t providers, we probably should initially sign up with n providers so that we can still create signatures if only t/n are available...

FROST [1]

Flexible Round-Optimized Schnorr Threshold (FROST) is a t-out-of-n threshold signature scheme:

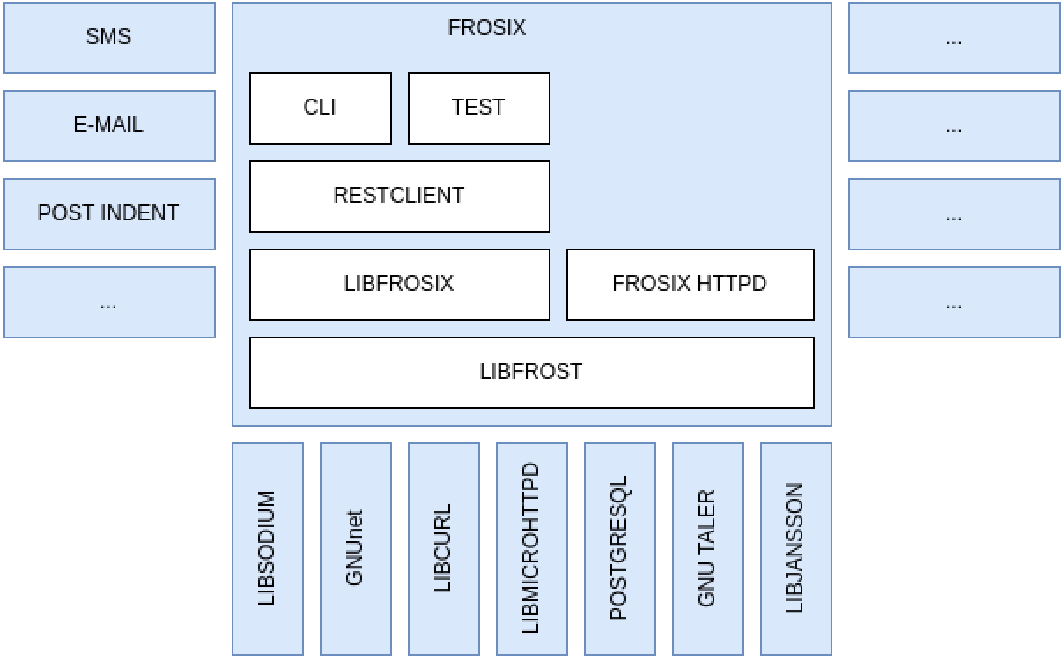
- Distributed key generation protocol can be used to ensure private key is never stored on a single device
- t providers required to collaborate to create digital signature

FROSIX

Free Software implementation for threshold signatures using FROST with:

- RESTful API to interact between signer and signing services
- Configurable authentication methods to authorize creation of signature
- Client should still use multiple devices (for authorization and to check distributed key generation) to avoid single point of failure
- Command-line tool to interact with FROSIX service providers

System components overview



FROSIX: Future Work

Open issues:

- Support additional signature schemes beyond EdDSA
- Pay signature service providers for their service
- Graphical user interfaces (Gtk+, WebUI, ...)

References I

Deirdre Connolly, Chelsea Komlo, Ian Goldberg, and Christopher A. Wood.
Two-round threshold schnorr signatures with frost.
Technical report, IRTF, 2023.
https://datatracker.ietf.org/doc/draft-irtf-cfrg-frost/.