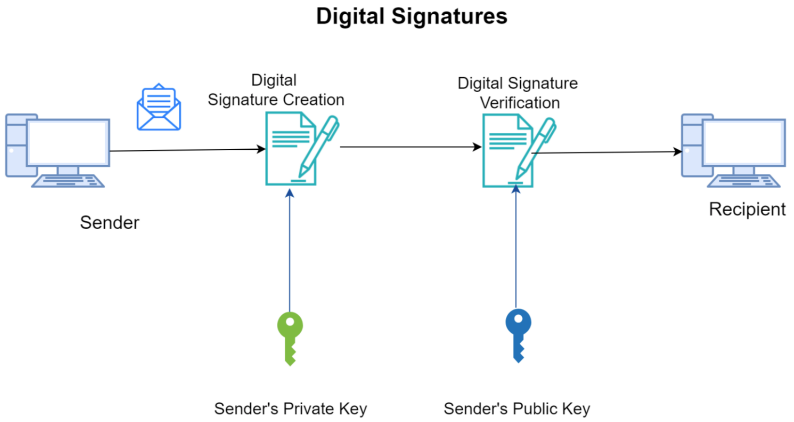
A New Hash-Based Enhanced Privacy ID Signature Scheme

Liqun Chen[1], Changyu Dong[2], Nada El Kassem[1],
Chris Newton[1], Yalan Wang[1]

[1] University of Surrey
[2] Guangzhou University

PQCrypto, June 2024

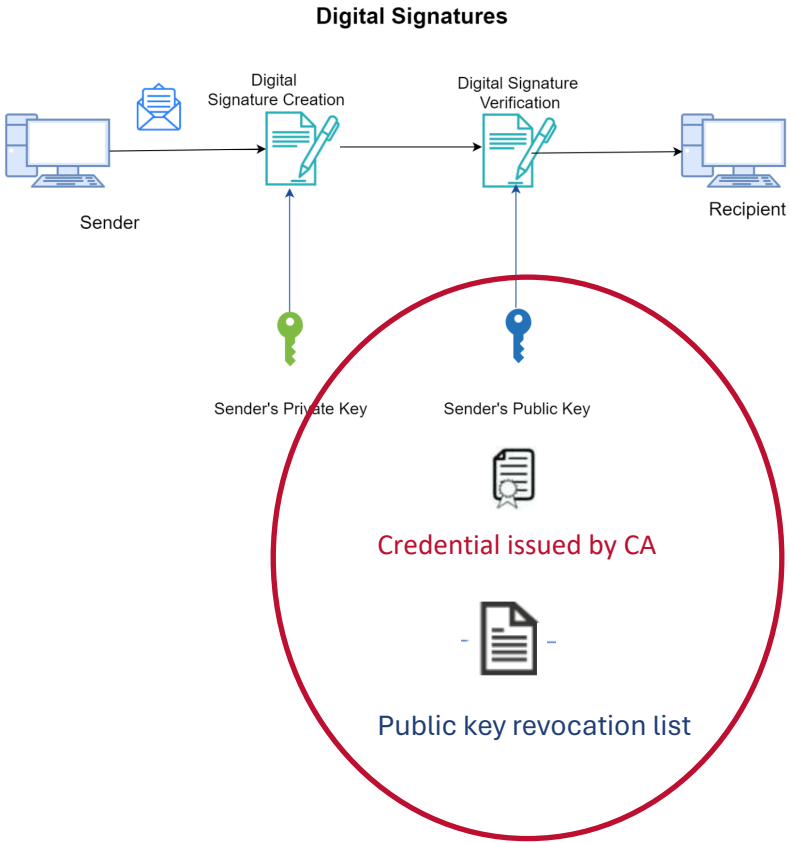
EPID vs traditional signatures



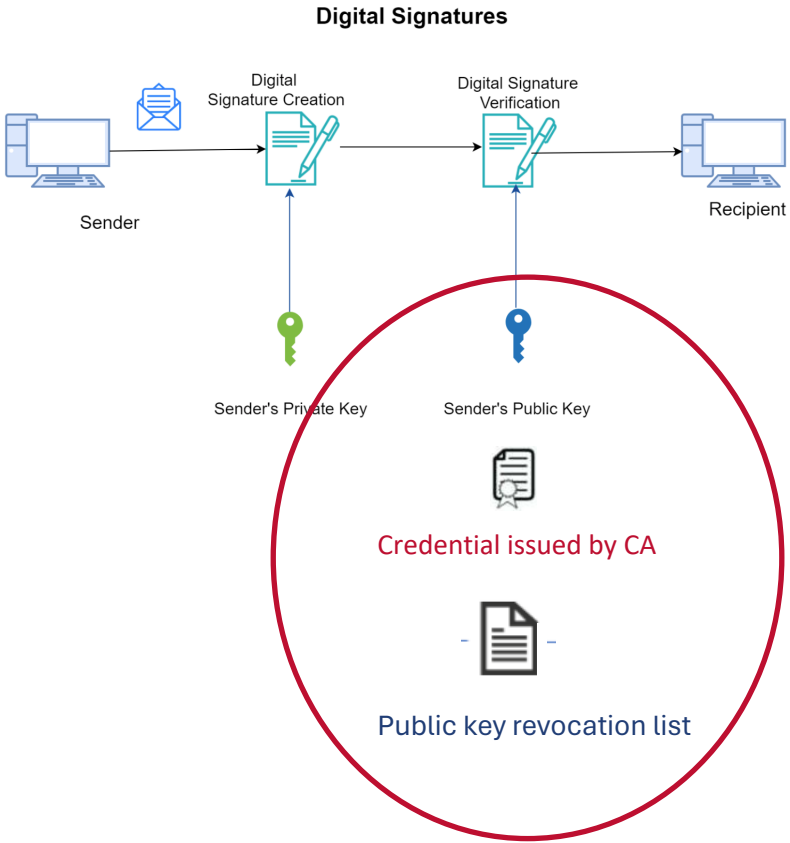
Credential issued by CA

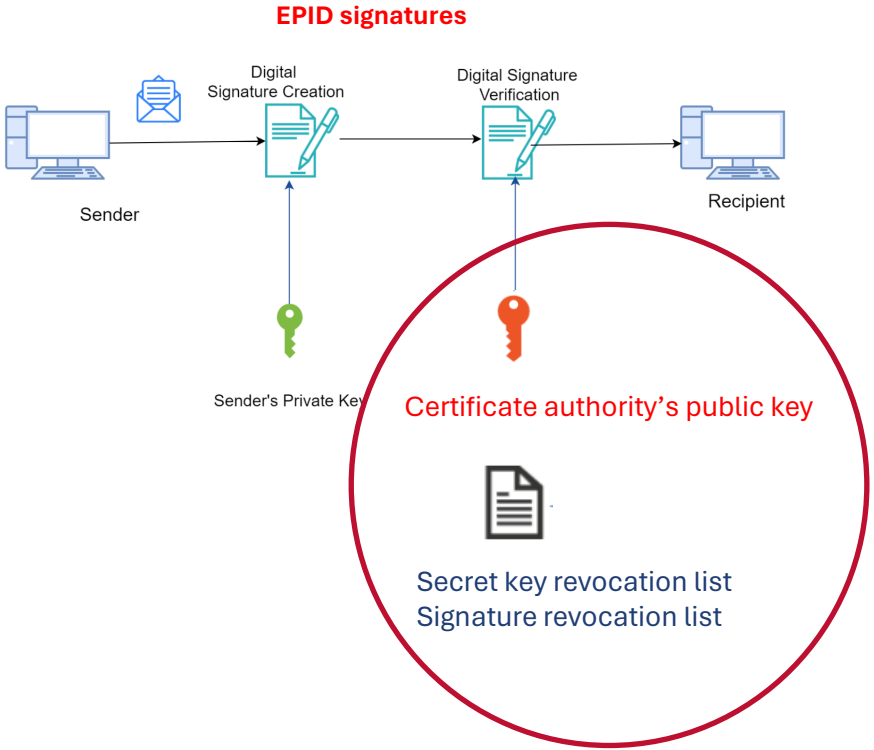
Public key revocation list

EPID vs traditional signatures



EPID vs traditional signatures





EPID properties

- Unforgeability
  - Only authorised users can sign
  - A legitimate signer must be authenticated by a certificate authority
- Anonymity
  - Given a signature, the signer's public key and certificate are not revealed to the verifier
- Secret key revocation
  - Revealed secret keys are in a revocation list
  - If a signer's secret key is on the list, any signatures created using this key are rejected
- Signature-based revocation
  - Revoked signatures are on a revocation list
  - A signer is asked to prove their key was not used to sign any signatures in the revocation list

EPID – quotes from Intel

- A perfect example usage of Intel® EPID is to prove that a hardware device is genuine.
- Intel is providing the Intel® EPID SDK open source and encouraging device manufacturers to adopt it as an industry standard for device ID in IoT.
- Billions of existing devices, including most Intel® platforms manufactured since 2008, create signatures that need Intel® EPID verification.
- In 2016, Intel as a certified EPID Key Generation Facility, announced that it has distributed over 4.5 billion EPID keys since 2008.

EPID History & State-of-the-Art

- EPID was originally proposed by Brickell and Li in 2007
  - As a new DAA (direct anonymous attestation) scheme with enhanced privacy ID
  - The first EPID scheme was based on RSA
- ECC-based EPID
  - Included in TPM (Trusted Platform Module) version 2.0 specifications
  - Specified in ISO/IEC 20008 in 2013
- Lattice-based EPID
  - A small number of schemes
- EPID from symmetric primitives
  - By Boneh, Eskandarian and Fisch in 2019
  - In this work, we aimed to design a more efficient EPID scheme from symmetric primitives

Various Signatures from Symmetric Primitives

- Traditional signatures from symmetric primitives
  - Hash-based signatures
    - One-time signatures
    - Few-time signatures
    - Stateful signatures
    - Stateless signatures
  - Picnic-style signatures
    - Using a Non-Interactive Zero-Knowledge Proof (NIZKP) to prove a one-way function

- Anonymous signatures
  - Ring signatures
  - Group signatures
  - Direct anonymous attestation (DAA)
  - Enhanced Privacy ID (EPID)

Challenges to EPID from Symmetric Primitives

- Signature-based revocation
  - This is the performance bottleneck
- Group size – the level of anonymity
  - Many existing hash-based anonymous signatures use a Merkle tree to arrange group membership credentials, so the group size is small
  - We aim to have a big group size, up to 2^{60}
  - This is another challenge for performance
- Our responses
  - Separate the implementation of revocation from group membership proof to minimise their impact on each other

Our EPID scheme (I)

EPID signing

Sender's private key

Signature revocation list (SRL)

CA's public key

Credential issued by CA

Zero-knowledge proof of credential

Zero-knowledge proof of not being revoked

F – a keyed pseudorandom function
sk_u – signer's private key
sid – signature ID
$sst = F(sk_u, sid)$ – signature signing token
$\mathsf{SRL} = \{..., (sid_j, sst_j), ...\}$ – signature revocation list
Pick a nonce r, compute a NIZKP

$$\pi_R : \mathcal{P}\{(sid, sst, r, \forall_j\ (sid_j, sst_j) \in \mathsf{SRL}, A_j); (sk_u)|$$

$$sst = F(sk_u, sid) \wedge \forall_j\ A_j = F(F(sk_u, sid_j), r)\}$$

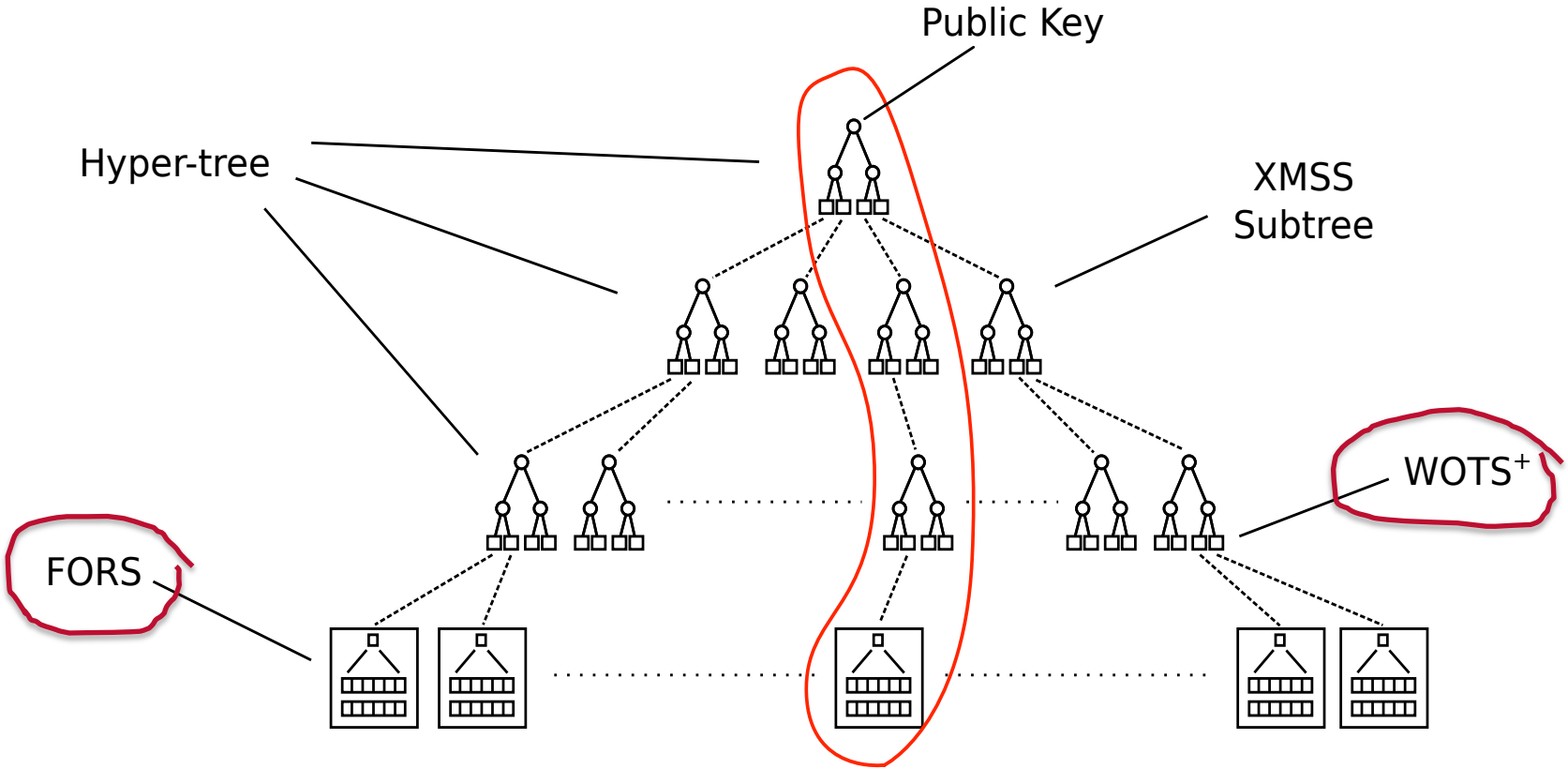
A verifier can compute $B_j = F(sst_j, r)$.
If $A_j \neq B_j$, $sst_j \in \mathsf{SRL}$ was not signed under sk_u.

To prove sk_u is certified and is used in $sst = F(sk_u, sid)$.

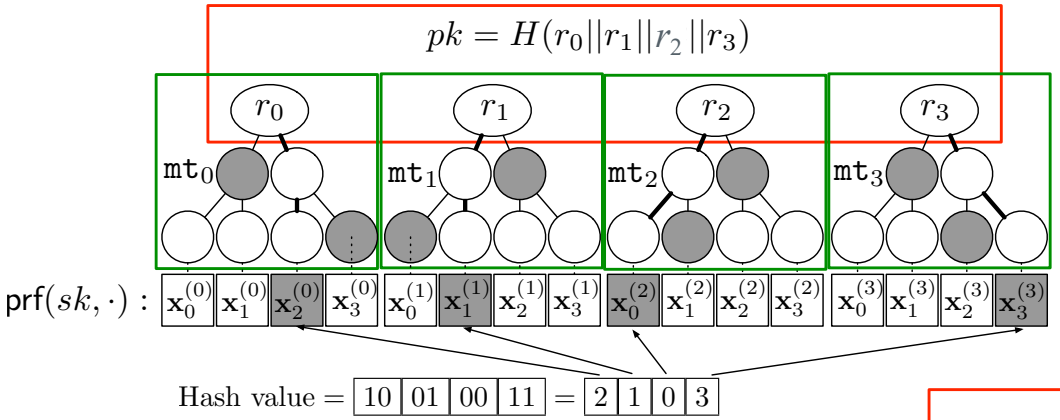
Our EPID scheme (II)

- Use a modified SPHINCS+ as an EPID credential
  - Modifying WOTS^{+}
  - Modifying FORS
- Use a Picnic-style signature to provide NIZKP
  - Masking all sensitive inputs and outputs
  - Using a partial proof for a better performance
- Chain two separate NIZKPs
  - Connecting the non-revoking proof with the credential proof
- The security of our scheme was proved under the Universal Composability (UC) model

SPHINCS+

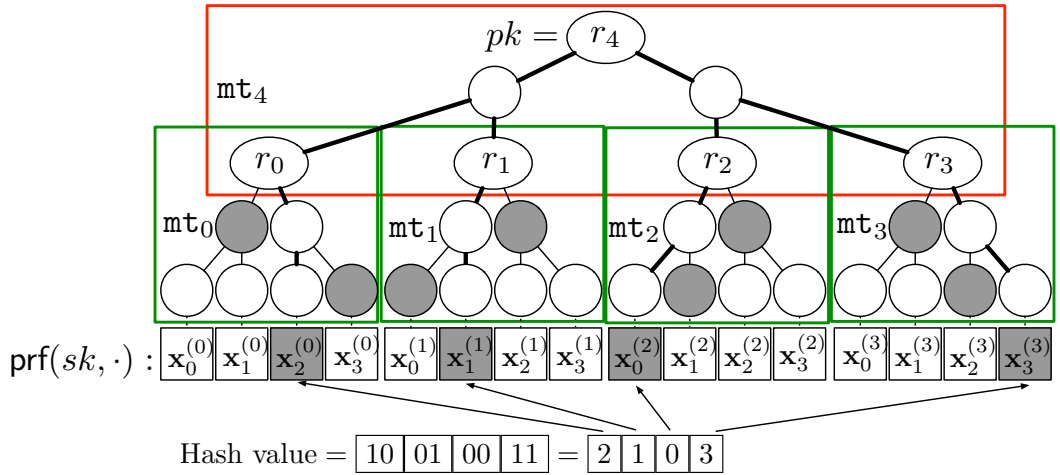


M-FORS (Modified FORS)

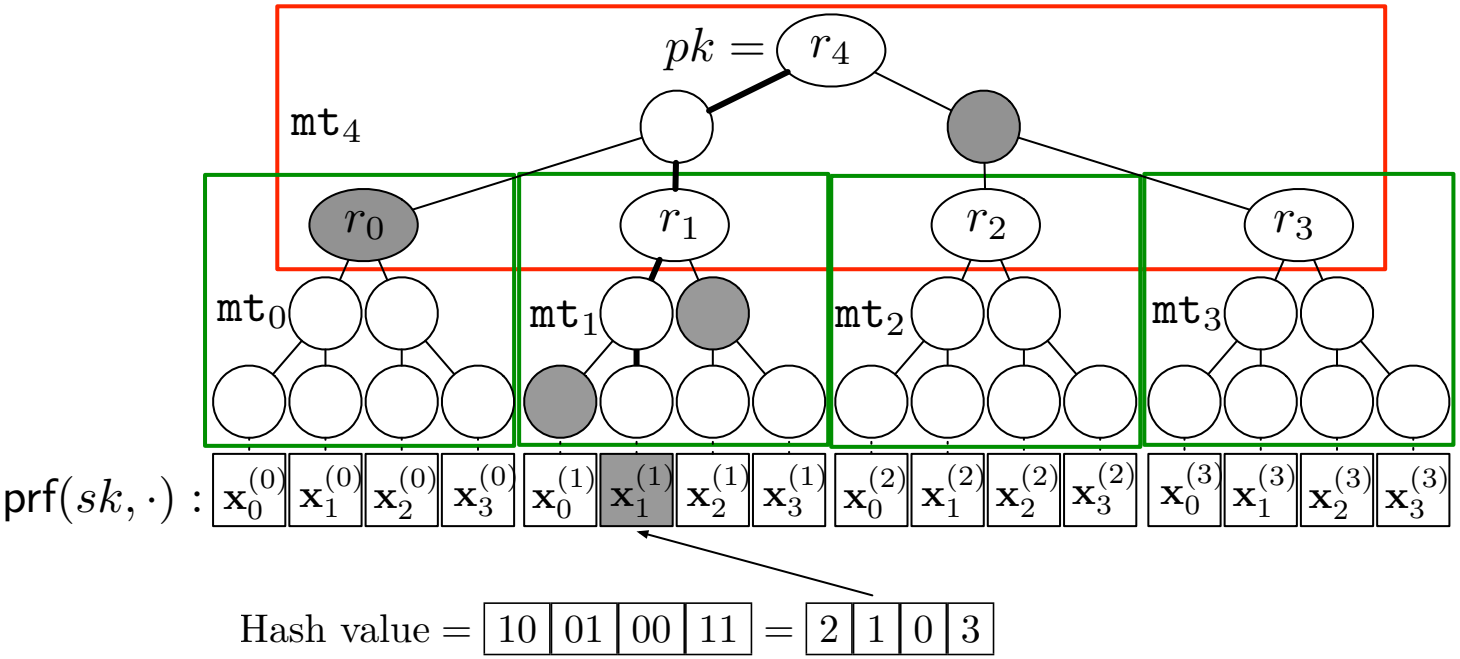


FORS

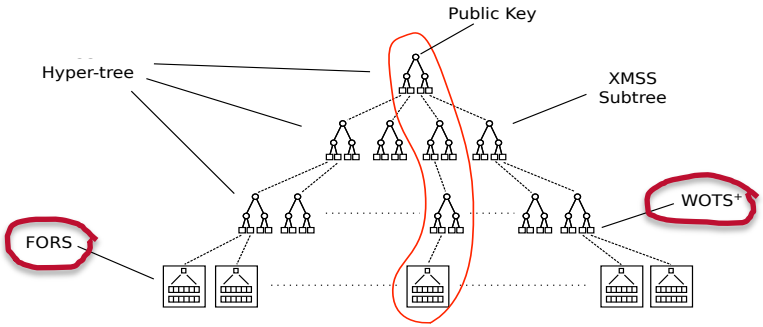
M-FORS



M-FORS Partial Proof

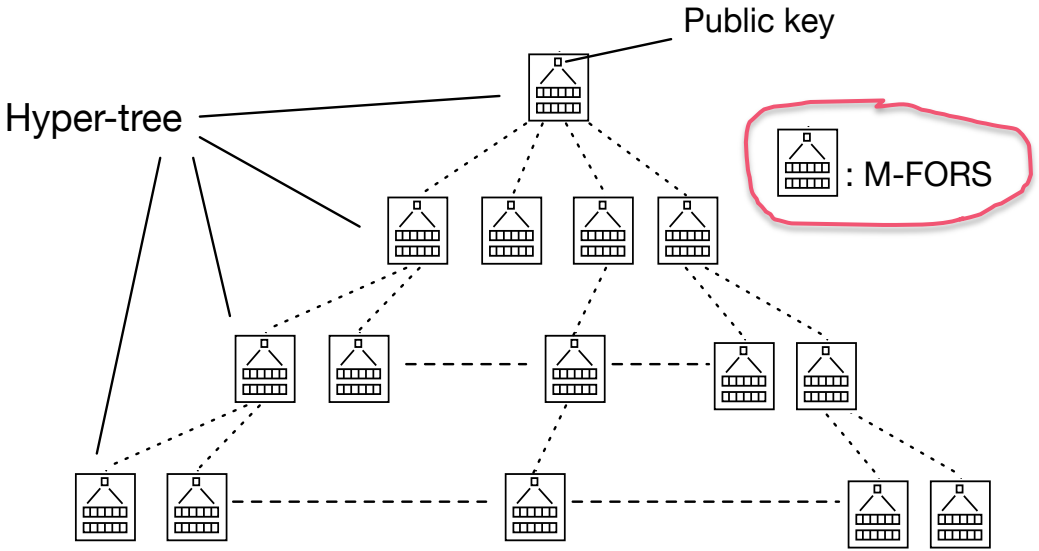


F-SPHINCS+ (Modified SPHINCS+)



SPHINCS+

F-SPHINCS+



Conclusions

- We propose a new EPID scheme from symmetric primitives
  - It can support a large group size of up to 2^{60}
  - It holds the EPID security properties under the UC model
- It makes use of three building blocks:
  - A hash-based signature as an EPID credential
  - A Picnic-style signature to prove the possession of that credential in a NIZK manner
  - An efficient NIZKP of not being revoked
- We have implemented our EPID scheme
  - Improving the performance will be possible if either a more efficient stateless hash-based signature scheme than F-SPHINCS+ or an efficient Picnic-style signature scheme is developed
- This work is still in its early stages

Thank you!
Questions?

liqun.chen@surrey.ac.uk