Forward-Secure Proxy Re-Encryption and Relations to HIBE and Puncturable Encryption

Daniel Slamanig, Graz University of Technology

Joint work with: David Derler · Stephan Krenn · Thomas Lorünser · Sebastian Ramacher · Christoph Striecks

March 22, 2017; Paderborn University

Graz University of Technology

Talk Outline

Forward Security

Proxy Re-Encryption

Forward-Secure Proxy Re-Encryption (fs-PRE)

- From Binary Tree Encryption

Fully Puncturable Encryption (FPuE)

- From Hierachical Identity-Based Encryption (HIBE)

fs-PRE from FPuE

Motivation: Forward Security

Conventional Setting

- Cryptographic keys often in use for a long time
- Key compromise at some point affects all past key uses

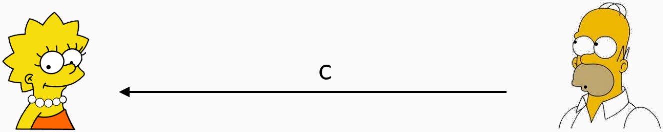
Forward Security

- Evolve secret key continuously (erase old key)
- But keep public-key constant
- Key compromise no longer affects previous key uses

Forward-Secure Public-Key Encryption

PKE, but evolve secret keys and encrypt with respect to epoch

Alice: $(\mathsf{sk}_A^{(0)}, \mathsf{pk}_A)$



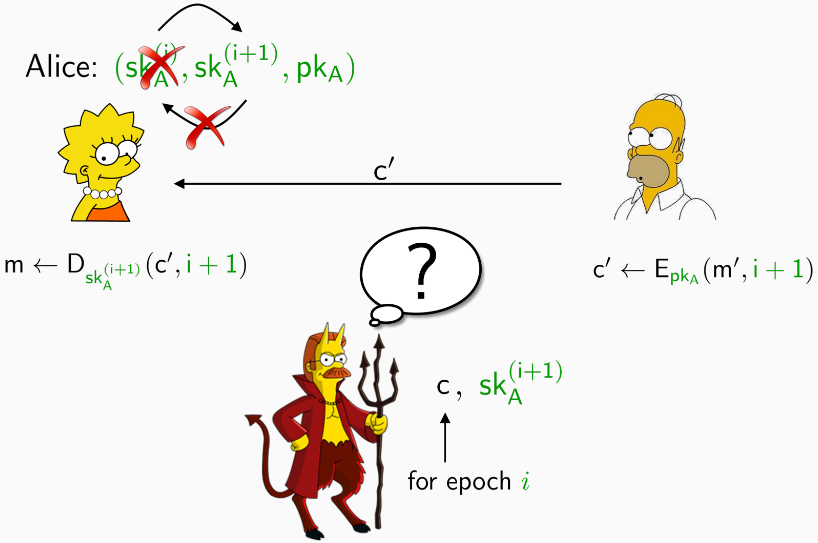
$m \leftarrow D_{\mathsf{sk}_A^{(0)}}(c, 0)$

$c \leftarrow E_{\mathsf{pk}_A}(m, 0)$

$$\begin{tikzcd}
\text{Epoch: } 0 : \mathsf{sk}_A^{(0)}, \arrow[r, bend left] & 1 : \mathsf{sk}_A^{(1)}, \arrow[r, bend left] & \ldots, \arrow[r, bend left] & i : \mathsf{sk}_A^{(i)}, \arrow[r, bend left] & \ldots
\end{tikzcd}$$

sk and pk size sublinear ; trivial for key size $O(\#\text{epochs})$

Forward-Secure Public-Key Encryption



Alice: $(\cancel{\mathsf{sk}_A^{(i)}}, \mathsf{sk}_A^{(i+1)}, \mathsf{pk}_A)$

c'

$\mathsf{m} \leftarrow \mathsf{D}_{\mathsf{sk}_A^{(i+1)}}(\mathsf{c}', \mathsf{i}+1)$

$\mathsf{c}' \leftarrow \mathsf{E}_{\mathsf{pk}_A}(\mathsf{m}', \mathsf{i}+1)$

$\mathsf{c}, \ \mathsf{sk}_A^{(i+1)}$

for epoch i

IND-CPA/CCA security defined in the obvious way

Forward Security: Related Work

Key-Exchange

- First work on FS [C. Günther, EuroCrypt'89]
- 0-RTT [F. Günther et al., EuroCrypt'17] (related techniques)

Encryption

- Public-key encryption [Canetti et al., EuroCrypt'03]
- Private-key encryption [Bellare & Yee, CT-RSA'03]

Signatures/Identification Schemes

- [Bellare & Miner, Crypto'99]
- [Abdalla et al., EuroCrypt'02]

Our Work

- FS for Proxy re-encryption [Blaze et al., EuroCrypt'98; Ateniese et al., NDSS'05]

Proxy Re-Encryption

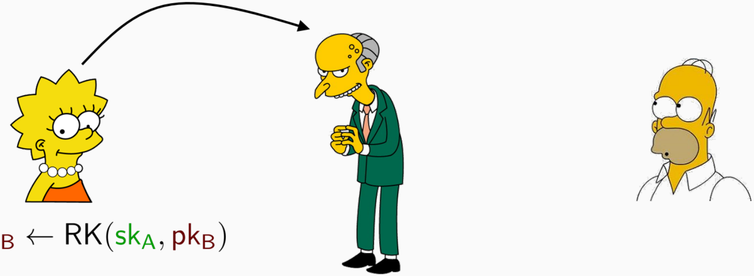
Transform ciphertext under one public key into one under another public key

Proxy Re-Encryption

Alice: $(\mathsf{sk}_A, \mathsf{pk}_A)$

Proxy

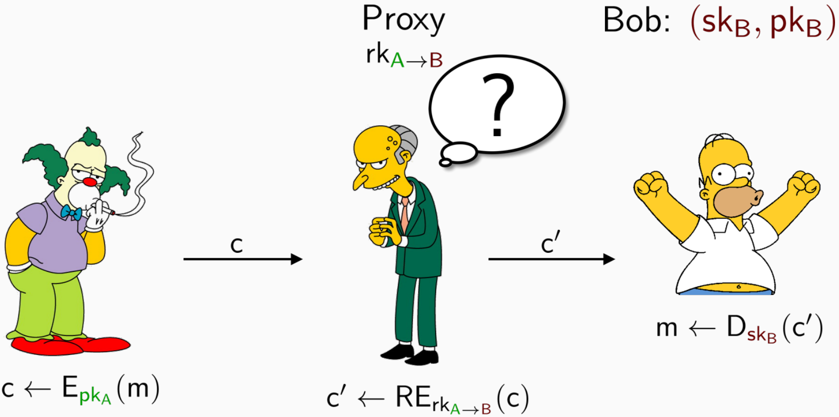
Bob: $(\mathsf{sk}_B, \mathsf{pk}_B)$



$\mathsf{rk}_{A \to B}$

$\mathsf{rk}_{A \to B} \leftarrow \mathsf{RK}(\mathsf{sk}_A, \mathsf{pk}_B)$

Proxy Re-Encryption



Proxy

$rk_{A \to B}$

Bob: (sk_B, pk_B)

c

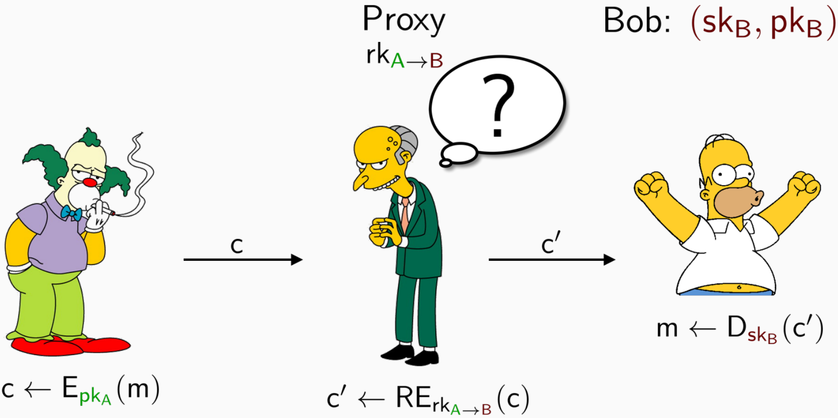
c'

$c \leftarrow E_{pk_A}(m)$

$c' \leftarrow RE_{rk_{A \to B}}(c)$

$m \leftarrow D_{sk_B}(c')$

Proxy Re-Encryption



Proxy

$\mathsf{rk}_{A\to B}$

Bob: $(\mathsf{sk}_B, \mathsf{pk}_B)$

$\xrightarrow{c}$

$\xrightarrow{c'}$

$c \leftarrow E_{\mathsf{pk}_A}(m)$

$c' \leftarrow RE_{\mathsf{rk}_{A\to B}}(c)$

$m \leftarrow D_{\mathsf{sk}_B}(c')$

Two types of ciphertexts: Re-encryptable or not (omitted)

Proxy Re-Encryption: Applications

Increasingly popular primitive (large scale EU projects, CFRG)

Store and forward

- E-Mail forwarding: delegate access to other parties
- E-Mail SPAM filtering

Outsourced storage

- Store data encrypted on untrusted servers, e.g., the cloud
- Central access control server re-encrypts content (or content encryption keys)

Forward-Secure Proxy Re-Encryption

Evolving Keys

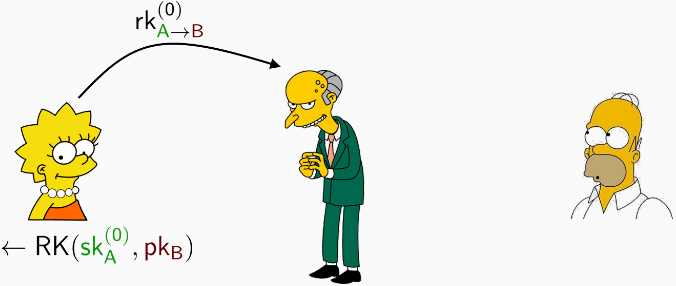
- Users evolve their private keys
- Proxy evolves re-encryption keys

Forward Security

- Secret key from $i+1$ does not allow to decrypt previous ciphertexts
- Re-encryption key from $i+1$ does not allow to re-encrypt previous ciphertexts

Forward-Secure Proxy Re-Encryption

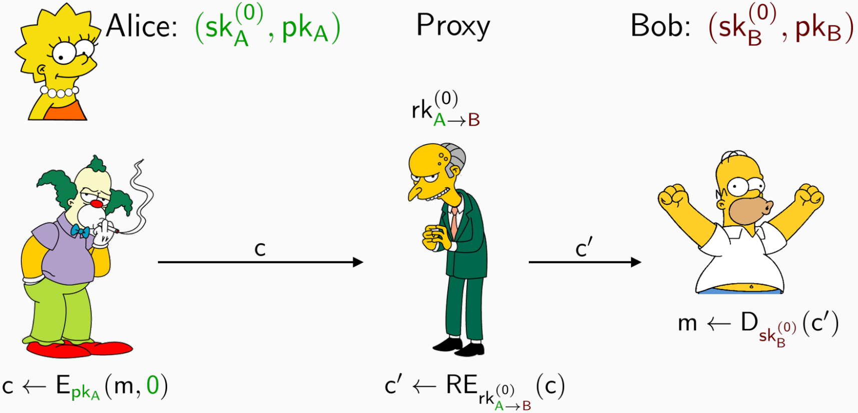
Alice: $(\mathsf{sk}_A^{(0)}, \mathsf{pk}_A)$ Proxy Bob: $(\mathsf{sk}_B^{(0)}, \mathsf{pk}_B)$



$\mathsf{rk}_{A \to B}^{(0)}$

$\mathsf{rk}_{A \to B}^{(0)} \leftarrow \mathsf{RK}(\mathsf{sk}_A^{(0)}, \mathsf{pk}_B)$

Forward-Secure Proxy Re-Encryption



Alice: $(sk_A^{(0)}, pk_A)$

Proxy

Bob: $(sk_B^{(0)}, pk_B)$

$rk_{A \to B}^{(0)}$

c

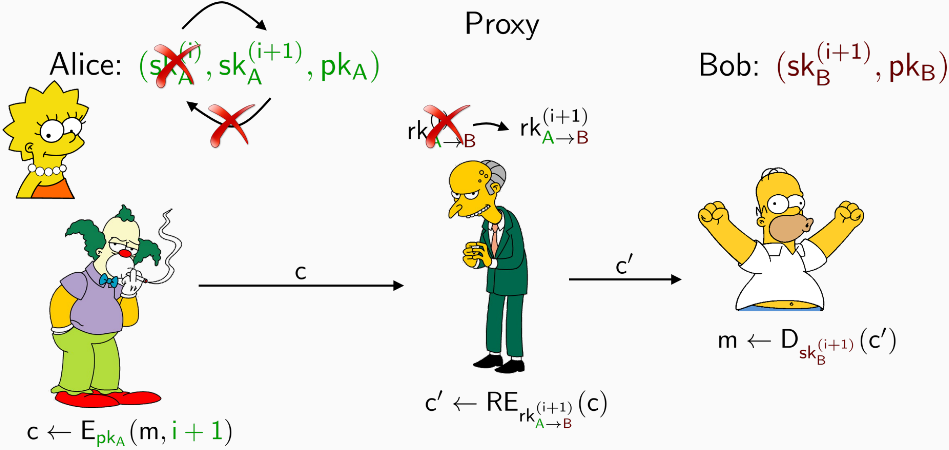
c'

$c \leftarrow E_{pk_A}(m, 0)$

$c' \leftarrow RE_{rk_{A \to B}^{(0)}}(c)$

$m \leftarrow D_{sk_B^{(0)}}(c')$

Forward-Secure Proxy Re-Encryption



Alice: $(\cancel{sk_A^{(i)}}, sk_A^{(i+1)}, pk_A)$

Proxy

Bob: $(sk_B^{(i+1)}, pk_B)$

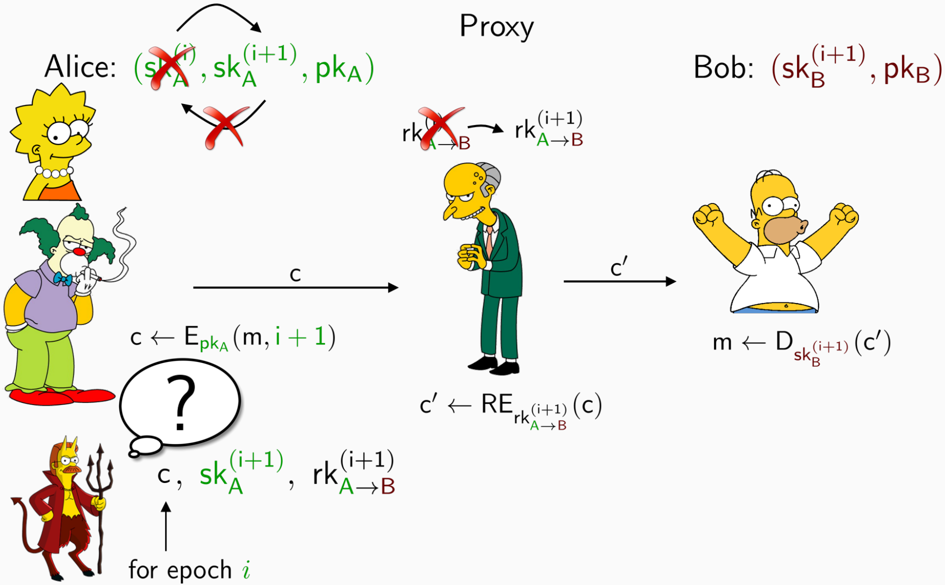
$\cancel{rk_{A\to B}^{(i)}} \rightarrow rk_{A\to B}^{(i+1)}$

$c \leftarrow E_{pk_A}(m, i+1)$

$c' \leftarrow RE_{rk_{A\to B}^{(i+1)}}(c)$

$m \leftarrow D_{sk_B^{(i+1)}}(c')$

Forward-Secure Proxy Re-Encryption



Alice: $(\cancel{sk_A^{(i)}}, sk_A^{(i+1)}, pk_A)$

Proxy

Bob: $(sk_B^{(i+1)}, pk_B)$

$\cancel{rk_{A\to B}^{(i)}} \rightarrow rk_{A\to B}^{(i+1)}$

c

c'

$c \leftarrow E_{pk_A}(m, i+1)$

$m \leftarrow D_{sk_B^{(i+1)}}(c')$

$c' \leftarrow RE_{rk_{A\to B}^{(i+1)}}(c)$

$c,\ sk_A^{(i+1)},\ rk_{A\to B}^{(i+1)}$

for epoch i

Forward-Secure Proxy Re-Encryption

Stronger Guarantees

- Even if receivers’ key leaks old ciphertexts are not in danger

Forward-Secure Proxy Re-Encryption

Stronger Guarantees

- Even if receivers' key leaks old ciphertexts are not in danger

Standard Requirements (fs-PRE^{-} notion)

- Porting standard notions to forward-security setting
- Secret key from interval j does not leak that of $j - 1$
- Ciphertexts of interval j indistinguishable for both levels even when seeing re-encryption keys for $j - 1$

Forward-Secure Proxy Re-Encryption

Stronger Guarantees

- Even if receivers' key leaks old ciphertexts are not in danger

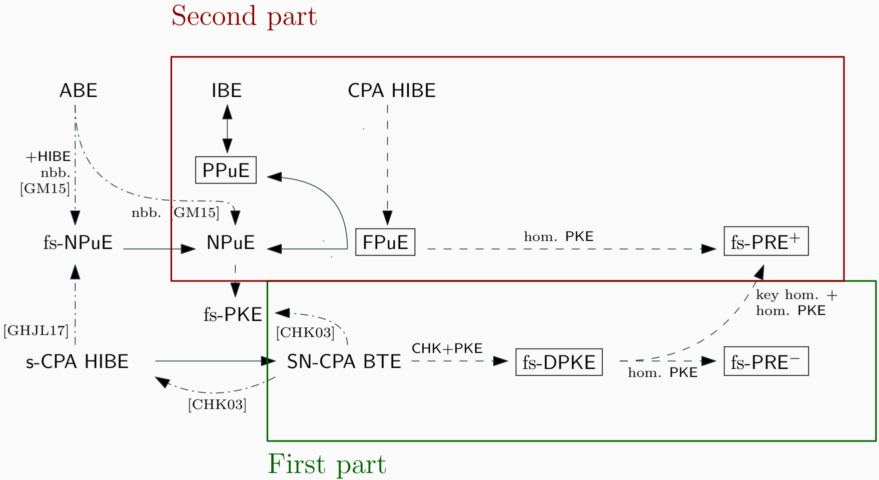
Standard Requirements (fs-PRE^- notion)

- Porting standard notions to forward-security setting
- Secret key from interval j does not leak that of $j-1$
- Ciphertexts of interval j indistinguishable for both levels even when seeing re-encryption keys for $j-1$

Strengthened Security (fs-PRE^+ notion)

- Proxy needs to be involved so that receiver can decrypt
- Missing in all proxy re-encryption models so far

Overview


Second part
ABE
IBE
CPA HIBE
PPuE
+HIBE
nbb.
[GM15]
nbb. [GM15]
fs-NPuE
NPuE
FPuE
hom. PKE
fs-PRE+
fs-PKE
[GHJL17]
[CHK03]
key hom. +
hom. PKE
s-CPA HIBE
SN-CPA BTE
CHK+PKE
fs-DPKE
hom. PKE
fs-PRE−
[CHK03]
First part

First Part: fs-PRE from Binary Tree Encryption

Start from Binary Tree Encryption

- Relaxed version of selectively secure HIBE

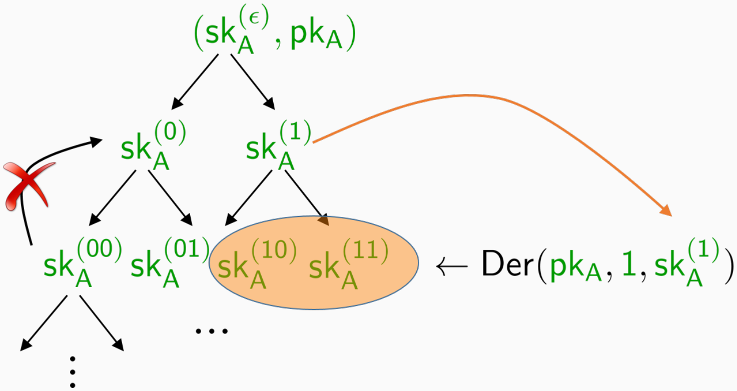
Forward-Secure Delegatable Public Key Encryption (fs-DPKE)

- Apply CHK [Canetti et al., EuroCrypt'03] compiler to BTE
- Combine with Public Key Encryption (PKE)

Forward-Secure Proxy Re-Encryption

- Implied from fs-DPKE using homomorphic PKE (fs-PRE^{-})
- Require key-homomorphism of fs-DPKE (fs-PRE^{+})

Binary Tree Encryption



$(\mathsf{sk}_A^{(\epsilon)}, \mathsf{pk}_A)$

$\mathsf{sk}_A^{(0)}$ $\mathsf{sk}_A^{(1)}$

$\mathsf{sk}_A^{(00)}$ $\mathsf{sk}_A^{(01)}$ $\mathsf{sk}_A^{(10)}$ $\mathsf{sk}_A^{(11)}$ $\leftarrow \mathsf{Der}(\mathsf{pk}_A, 1, \mathsf{sk}_A^{(1)})$

$\vdots$ $\ldots$

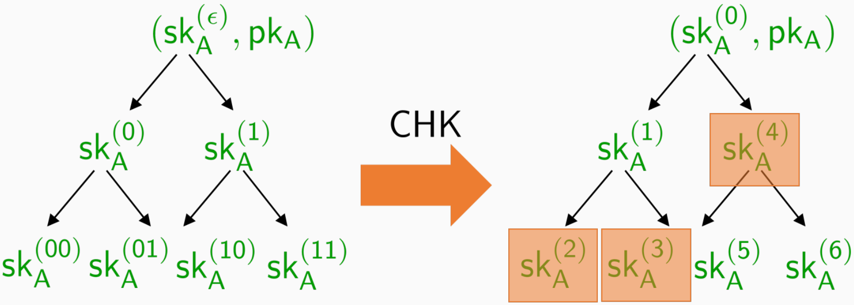
Encrypt using pk_A and node id; Decrypt with secret key for node (or for a prefix)

Construction: [Canetti et al., EuroCrypt'03] from bilinear DDH

From BTE to fs-PKE

Apply CHK compiler to BTE to obtain fs-PKE

- For N intervals use BTE of depth ℓ with $N = 2^{\ell+1} - 1$
- Intervals are node labels from pre-order traversal of tree
- Secret key for interval i: key for node i and all right siblings on path to root

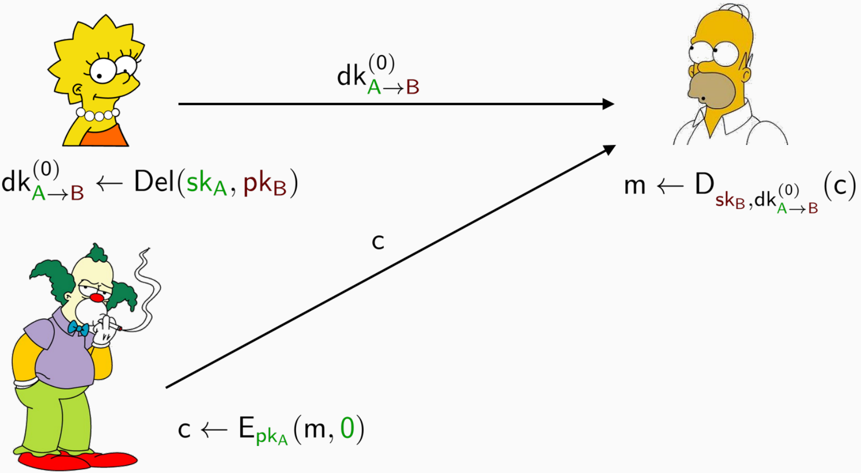


for epoch 2

Forward-Secure DPKE

Alice: $(sk_A^{(0)}, pk_A)$

Bob: (sk_B, pk_B)



$dk^{(0)}_{A \to B}$

$dk^{(0)}_{A \to B} \leftarrow \mathsf{Del}(sk_A, pk_B)$

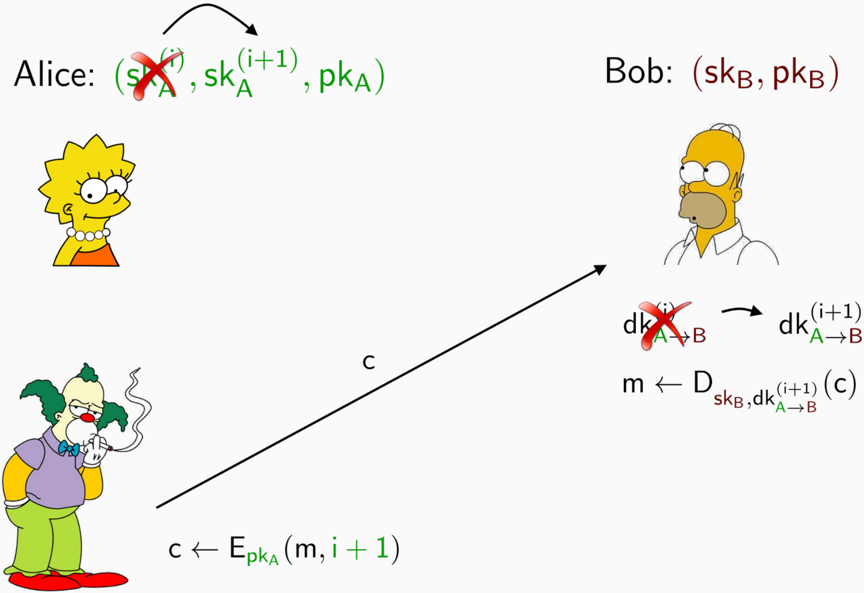
$m \leftarrow D_{sk_B, dk^{(0)}_{A \to B}}(c)$

c

$c \leftarrow E_{pk_A}(m, 0)$

Omitting Bob’s forward-security for simplicity

Forward-Secure DPKE



Alice: $(\cancel{sk_A^{(i)}}, sk_A^{(i+1)}, pk_A)$

Bob: (sk_B, pk_B)

$\cancel{dk_{A\to B}^{(i)}} \rightarrow dk_{A\to B}^{(i+1)}$

$m \leftarrow D_{sk_B, dk_{A\to B}^{(i+1)}}(c)$

$c \leftarrow E_{pk_A}(m, i+1)$

Constructing fs-DPKE

Requirements

- Bob needs secret sk_{B} and delegation $\mathsf{dk}^{(0)}_{\mathsf{A}\to\mathsf{B}}$ to decrypt
- Delegation $\mathsf{dk}^{(0)}_{\mathsf{A}\to\mathsf{B}}$ is not a secret
- Secret sk_{A} and $\mathsf{dk}^{(0)}_{\mathsf{A}\to\mathsf{B}}$ need to be evolved

Constructing fs-DPKE

Requirements

- Bob needs secret sk_B and delegation $\mathsf{dk}^{(0)}_{\mathsf{A}\to\mathsf{B}}$ to decrypt
- Delegation $\mathsf{dk}^{(0)}_{\mathsf{A}\to\mathsf{B}}$ is not a secret
- Secret sk_A and $\mathsf{dk}^{(0)}_{\mathsf{A}\to\mathsf{B}}$ need to be evolved

Concrete Construction

- Keys: BTE (fs-security compiler) and PKE key
- Delegation key: secret key of BTE encrypted under public key of receivers' PKE

Obtaining fs-PRE^{-}

Build upon fs-DPKE

- Give delegation key $\mathsf{dk}^{(0)}_{A \to B}$ as re-encryption key to proxy

Obtaining fs-PRE^{-}

Build upon fs-DPKE

- Give delegation key $\mathsf{dk}^{(0)}_{A \to B}$ as re-encryption key to proxy
- Re-encryption is just assembling ciphertext and delegation key

Obtaining fs-PRE^{-}

Build upon fs-DPKE

- Give delegation key $\mathsf{dk}^{(0)}_{A \to B}$ as re-encryption key to proxy
- Re-encryption is just assembling ciphertext and delegation key
- How can the proxy evolve the delegation key?
    - $\mathsf{dk}^{(0)}_{A \to B}$ is an encryption of a BTE secret key under receivers public key!

Obtaining fs-PRE^{-}

Build upon fs-DPKE

- Give delegation key $\mathsf{dk}^{(0)}_{A \to B}$ as re-encryption key to proxy
- Re-encryption is just assembling ciphertext and delegation key
- How can the proxy evolve the delegation key?
  - $\mathsf{dk}^{(0)}_{A \to B}$ is an encryption of a BTE secret key under receivers public key!

Solution

- Proxy homomorphically evaluates the evolution algorithm
  - Derivation algorithm of BTE
  - Can evolve re-encryption keys without learning them

Obtaining fs-PRE^{-}

Build upon fs-DPKE

- Give delegation key $\mathsf{dk}^{(0)}_{A \to B}$ as re-encryption key to proxy
- Re-encryption is just assembling ciphertext and delegation key
- How can the proxy evolve the delegation key?
    - $\mathsf{dk}^{(0)}_{A \to B}$ is an encryption of a BTE secret key under receivers public key!

Solution

- Proxy homomorphically evaluates the evolution algorithm
    - Derivation algorithm of BTE
    - Can evolve re-encryption keys without learning them
- Isn't that expensive? 😐

Obtaining fs-PRE^{-}

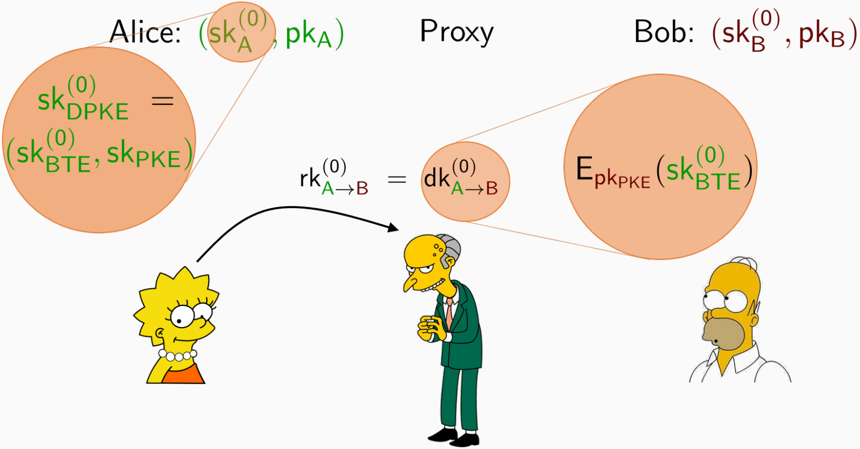
Build upon fs-DPKE

- Give delegation key $\mathsf{dk}^{(0)}_{A \to B}$ as re-encryption key to proxy
- Re-encryption is just assembling ciphertext and delegation key
- How can the proxy evolve the delegation key?
  - $\mathsf{dk}^{(0)}_{A \to B}$ is an encryption of a BTE secret key under receivers public key!

Solution

- Proxy homomorphically evaluates the evolution algorithm
  - Derivation algorithm of BTE
  - Can evolve re-encryption keys without learning them
- Isn’t that expensive? 😐
- Derivation algorithm of Canetti et al. BTE requires only linear operations! ☺

Obtaining fs-PRE$^-$ cont.



Alice: $(\mathsf{sk}_{\mathsf{A}}^{(0)}, \mathsf{pk}_{\mathsf{A}})$

Proxy

Bob: $(\mathsf{sk}_{\mathsf{B}}^{(0)}, \mathsf{pk}_{\mathsf{B}})$

$\mathsf{sk}_{\mathsf{DPKE}}^{(0)} = (\mathsf{sk}_{\mathsf{BTE}}^{(0)}, \mathsf{sk}_{\mathsf{PKE}})$

$\mathsf{rk}_{\mathsf{A}\to\mathsf{B}}^{(0)} = \mathsf{dk}_{\mathsf{A}\to\mathsf{B}}^{(0)}$

$\mathsf{E}_{\mathsf{pk}_{\mathsf{PKE}}}(\mathsf{sk}_{\mathsf{BTE}}^{(0)})$

Obtaining fs-PRE$^-$ cont.

Alice: $(\mathsf{sk}_A^{(i)}, \mathsf{pk}_A)$

Proxy

Bob: $(\mathsf{sk}_B^{(0)}, \mathsf{pk}_B)$

$\nearrow \mathsf{dk}_{A\to B}^{(i+1)}$

$\nearrow E_{\mathsf{pk}_{\mathsf{PKE}}}(\mathsf{sk}_{\mathsf{BTE}}^{(i+1)})$

$$\mathsf{rk}_{A\to B}^{(i)} = \cancel{\mathsf{dk}_{A\to B}^{(i)}}$$

$$\cancel{E_{\mathsf{pk}_{\mathsf{PKE}}}(\mathsf{sk}_{\mathsf{BTE}}^{(i)})}$$

w/o decryption!

Key evolution (as in fs-*PRE*$^-$ construction); homomorphically

Obtaining fs-PRE$^+$

Stronger notion: re-encryption required prior to decryption

- Clearly not satisfied by previous construction!
- After one re-encryption receiver knows BTE delegation key
- Can decrypt every ciphertext without involvement of proxy

Obtaining fs-PRE$^+$

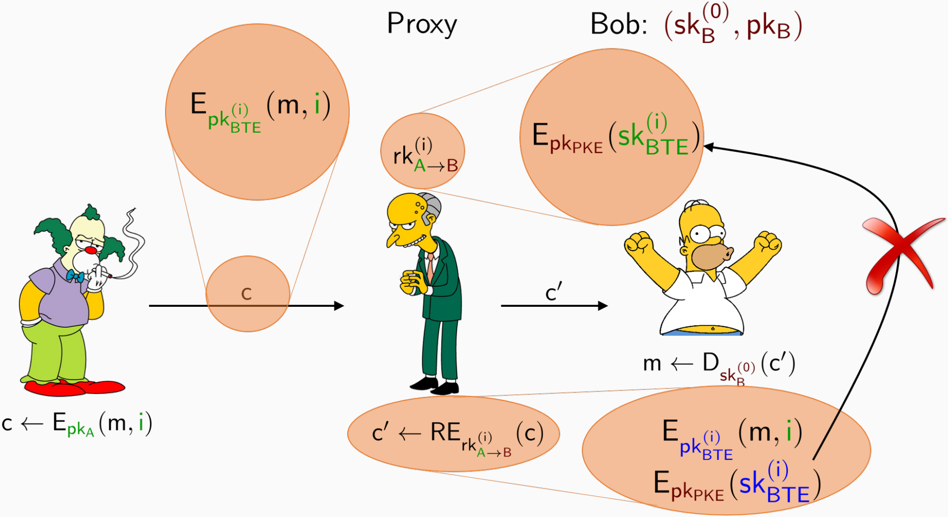
Stronger notion: re-encryption required prior to decryption

- Clearly not satisfied by previous construction!
- After one re-encryption receiver knows BTE delegation key
- Can decrypt every ciphertext without involvement of proxy

Tweaking previous construction

- Re-encryption key is encryption of BTE secret key
- Change BTE secret and public key in re-encryption key and ciphertexts homomorphically
  - Key-homomorphic BTE & hom. of PKE compatible

Obtaining fs-PRE$^+$



Proxy

Bob: $(sk_B^{(0)}, pk_B)$

$E_{pk_{BTE}^{(i)}}(m, i)$

c

$rk_{A\to B}^{(i)}$

$E_{pk_{PKE}}(sk_{BTE}^{(i)})$

c'

$m \leftarrow D_{sk_B^{(0)}}(c')$

$c \leftarrow E_{pk_A}(m, i)$

$c' \leftarrow RE_{rk_{A\to B}^{(i)}}(c)$

$E_{pk_{BTE}^{(i)}}(m, i)$

$E_{pk_{PKE}}(sk_{BTE}^{(i)})$

Switch keys $pk_{BTE}^{(i)}$ and $sk_{BTE}^{(i)}$ to fresh random keys $pk_{BTE}^{(i)}$ and $sk_{BTE}^{(i)}$ w/o decryption (key-homomorphism)

Second Part: fs-PRE from Fully Puncturable Encryption

Puncturable Encryption

- Update (puncture) a secret key such that it does no longer decrypt a certain ciphertext

Second Part: fs-PRE from Fully Puncturable Encryption

Puncturable Encryption

- Update (puncture) a secret key such that it does no longer decrypt a certain ciphertext

Constructions

- [Green & Miers, S&P'15]
  - Sel. secure HIBE/BTE + Attribute-Based Encryption (with specific malleability properties on keys)
- [F. Günther et al., EuroCrypt'17]
  - Adapt. secure HIBE

Second Part: fs-PRE from Fully Puncturable Encryption

Puncturable Encryption

- Update (puncture) a secret key such that it does no longer decrypt a certain ciphertext

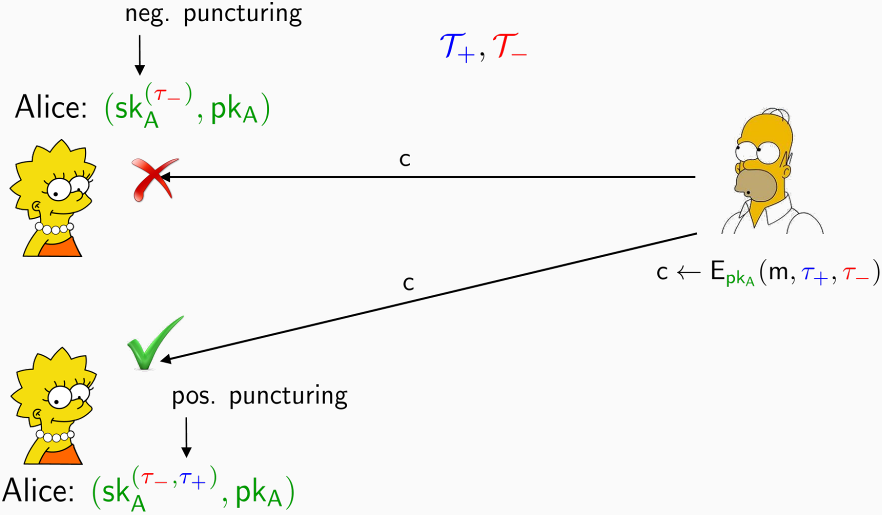
Constructions

- [Green & Miers, S&P'15]
  - Sel. secure HIBE/BTE + Attribute-Based Encryption (with specific malleability properties on keys)
- [F. Günther et al., EuroCrypt'17]
  - Adapt. secure HIBE

Generalizing this Primitive?

- Fully Puncturable Encryption (FPuE)
  - Negative and positive puncturing

Second Part: fs-PRE from Fully Puncturable Encryption cont.



neg. puncturing

$\mathcal{T}_+, \mathcal{T}_-$

Alice: $(\mathsf{sk}_A^{(\tau_-)}, \mathsf{pk}_A)$

c

c

$\mathsf{c} \leftarrow \mathsf{E}_{\mathsf{pk}_A}(\mathsf{m}, \tau_+, \tau_-)$

pos. puncturing

Alice: $(\mathsf{sk}_A^{(\tau_-, \tau_+)}, \mathsf{pk}_A)$

Second Part: fs-PRE from Fully Puncturable Encryption cont.

Fully Puncturable Encryption (FPuE)

- Generalization of puncturable encryption
- Black-box from adap. secure HIBE

Second Part: fs-PRE from Fully Puncturable Encryption cont.

Fully Puncturable Encryption (FPuE)

- Generalization of puncturable encryption
- Black-box from adap. secure HIBE

Forward-Secure Public Key Encryption

- Directly implied from FPuE (negative puncturing)

Second Part: fs-PRE from Fully Puncturable Encryption cont.

Fully Puncturable Encryption (FPuE)

- Generalization of puncturable encryption
- Black-box from adap. secure HIBE

Forward-Secure Public Key Encryption

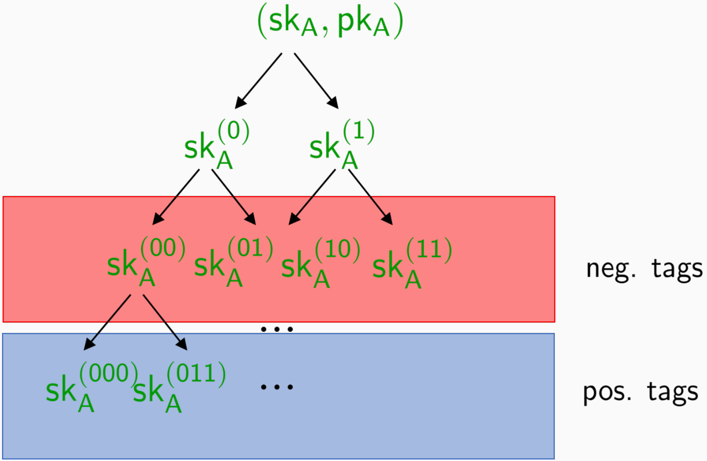
- Directly implied from FPuE (negative puncturing)

Forward-Secure Proxy-Re Encryption

- Forward-security from negative puncturing
- Single-use of re-encryption (fs-PRE^{+}) from positive puncturing

FPuE from HIBES

Consider HIBE for a complete binary tree



fs-PRE$^+$ from FPuE

Construction Sketch

- Re-encryption key: encryption of FuPE secret key

fs-PRE$^+$ from FPuE

Construction Sketch

- Re-encryption key: encryption of FuPE secret key
- Negative puncturing (τ_-)
    - Mapping negative tag space to epochs

fs-PRE$^+$ from FPuE

Construction Sketch

- Re-encryption key: encryption of FuPE secret key
- Negative puncturing (τ_-)
    - Mapping negative tag space to epochs
- Positive puncturing (τ_+)
    - Make FuPE secret key single-use (strong security notion)

fs-PRE$^+$ from FPuE

Construction Sketch

- Re-encryption key: encryption of FuPE secret key
- Negative puncturing (τ_-)
    - Mapping negative tag space to epochs
- Positive puncturing (τ_+)
    - Make FuPE secret key single-use (strong security notion)
- Puncturing on encrypted keys (re-encryption keys)
    - PKE of receiver needs to be homomorphic
    - Evaluate puncturing homomorphically (only linear operations in pairing based HIBEs)

Instantiations of fs-PRE

Building Block	$\lvert\mathsf{pk}\rvert$	$\lvert\mathsf{rk}^{(i)}\rvert$	$\lvert\mathsf{sk}^{(i)}\rvert$	$\lvert C\rvert$	Ass.
SN-CPA BTE	$O(\log N)$	$O((\log N)^2)$	$O((\log N)^2)$	$O(\log N)$	BDDH
FPuE via HIBE	$O(\log N)$	$O((\log N)^2)$	$O((\log N)^2)$	$O(1)$	DSG

DSG: Dual System Groups [Waters, Crypto'09]

Conclusions & Outlook

- Introduced forward-security for proxy re-encryption
- Strengthened security model (also for classical PRE)
- Two directions ; from BTE and FPuE
    - Instantiations in the standard model
- (Fully) Puncturable encryption is very interesting

Thank you.

Preprint available on request

✉ daniel.slamanig@iaik.tugraz.at @drl3c7er