Homomorphic Proxy Re-Authenticators

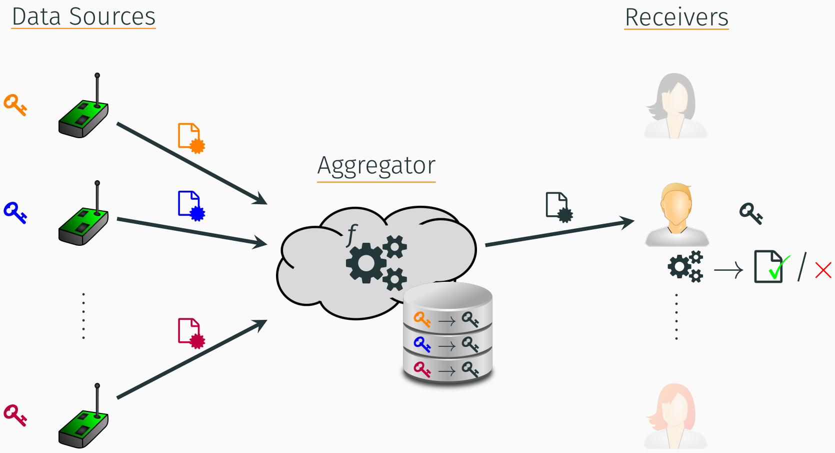
and Applications to Verifiable Multi-User Data Aggregation

David Derler, Sebastian Ramacher, and Daniel Slamanig

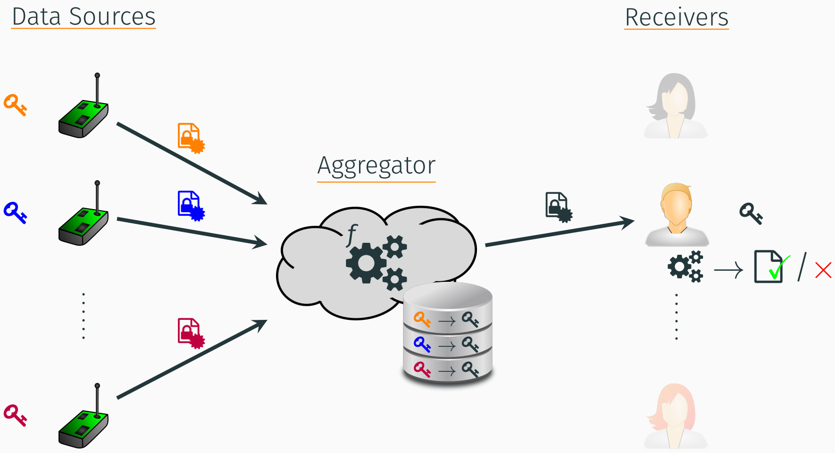
April 2017—FC 2017, Sliema, Malta

Graz University of Technology

Overview



Overview



Data Sources

Aggregator

Receivers

Goals

End-to-end authenticity

- Protect data from unauthorized manipulation
- Preserve source authenticity

Conceal original data

- Receiver only learns result of computation and f

Conceal computation result

- Aggregator does neither learn inputs nor result

Setting

- Independent keys for all parties
- Non-interactive re-key generation

→ No centralized setup!

Related Work

Proxy re-cryptography (semi-trusted proxy)

- Re-encryption: 🔐 → 🔐 using 🔑 → 🔑 [BBS98, ID03, AFGH06]

→ Pailler encryption with split key [ARHR13]

→ Fully homomorphic encryption based [MLO16]

- Re-signing: 📄 → 📄 using 🔑 → 🔑 [BBS98, ID03, AH05, LV08]

Homomorphic authenticators

- 📄 ← f(📄, 📄,… , 📄) under single key Overview in [Cat14]
- Multi-key homomorphic authenticators [FMNP16,DS16,LTWC16]

Related Work contd.

Aggregator oblivious encryption (AOE) [RN10, SCR+11]

- Aggregation of data from multiple sources
- Semi-trusted aggregator only learns final result
- AOE with homomorphic tags → verifiability [LEÖM15]
- Not possible to hide outputs from aggregator
- Trusted distribution of keys

… also other lines of work on data aggregation

Bottom line

- Nothing covers all our requirements

Contribution

Homomorphic Proxy Re-Authenticators (HPRA)

- Multi-user data aggregation
- Under independent keys for sources
- Verifiability of evaluations of general functions
- Privacy w.r.t. the aggregator

Homomorphic Proxy Re-Encryption (HPRE)

- Formal definitions
- Construction for linear functions

Construction of HPRA

- For the class of linear functions
- Suitable linearly homomorphic MAC
- Privacy via HPRE for linear functions

Homomorphic Proxy Re-Authenticators

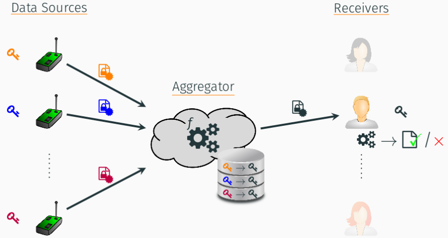
Algorithms

- Parameter/key generation: **Gen**, **SGen**, **VGen**
- Signature generation/verification: **Sign**, **Verify**
- Re-key generation: **SRGen**, **VRGen**
- Aggregation/verification algorithms: **Agg**, **AVerify**

Remarks

→ **Verify** is optional

→ Re-key generation non-interactive



Unforgeability

Non-collusion assumption

- Of sources and aggregator
- Impossible to circumvent

→ Colluding parties could authenticate everything

Signer unforgeability

- Intractable to produce forgery
- For coalition of dishonest sources
- As long as aggregator remains honest

Aggregator unforgeability

- Natural counterpart of signer unforgeability
- Dishonest aggregator, honest signers

Privacy

Input privacy

- Evaluation of f on authenticated vectors hides inputs

→ Same information as when only seeing f and y

Output privacy

- Aggregator neither learns inputs
- Nor result of evaluation of f on inputs

HPRA for Linear Functions - Achieving Input Privacy

Basic idea

- Combine linearly homomorphic signature scheme
- With compatible linearly homomorphic MAC

+ Mechanism to “switch” keys

Building blocks

- Adaption of network coding signatures (tag based) [BFKW09]
- Convert [BFKW09] to MAC

+ Prove MAC unforgeable under adversarially chosen tags

+ Prove security of overall construction

Construction Sketch

Setup

- Bilinear group setting $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, prime order p
- Public parameters: $(g_i)_{i \in [\ell]} \in \mathbb{G}^\ell$
- Source: $\mathsf{sk} \leftarrow \beta \in \mathbb{Z}_p$, $\mathsf{pk} \leftarrow (g^\beta, g^{1/\beta})$
- Receiver: $\mathsf{sk} \leftarrow \alpha \in \mathbb{Z}_p$
- Re-signing key: $g^{\alpha/\beta}$

Signature under source key (lives in $\mathbb{G}$)

$$\sigma \leftarrow \left(H(\tau || g^\beta) \cdot \prod_{i \in [\ell]} g_i^{m_i}\right)^\beta$$

Convert to MAC under receiver's key (lives in $\mathbb{G}_T$)

$$\mu \leftarrow e(\sigma, g^{\alpha/\beta}) = e\left(\left(H(\tau || g^\beta) \cdot \prod_{i \in [\ell]} g_i^{m_i}\right), g\right)^\alpha$$

Construction Sketch - Security

Unforgeability (ROM)

- Signer unforgeability: UF of MAC (bilinear DDH)
- Aggregator unforgeability: bilinear CDH variant

Input privacy

- For all $\vec{m}_1$, $\vec{m}_2$ with $f(\vec{m}_1) = f(\vec{m}_2)$
- Signatures/MACs identically distributed

Achieving Output Privacy

Basic idea

- Use input private scheme

+ Encrypt vectors with HPRE

→ Evaluate function on signatures and ciphertexts

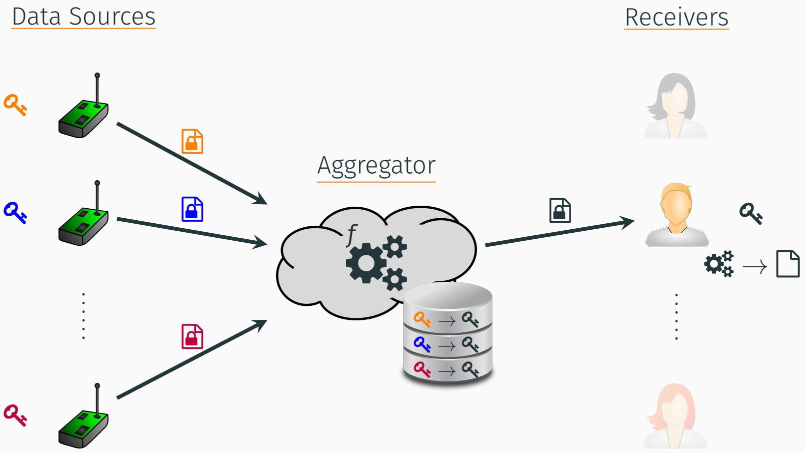
Additional Obstacles

- Signatures still publicly verifiable!
- MAC for sources no option (interactive key generation)

→ Blind signature with blinding value g^r

+ Use HPRE to encrypt blinding value

Homomorphic Proxy Re-Encryption (HPRE)



Homomorphic Proxy Re-Encryption (HPRE)

Conventional PRE scheme

+ Additional algorithm Eval
- Evaluate functions f on ciphertexts
- Decryption yields evaluation of f on the plaintexts

Nice feature

- Collect data from multiple sources
- Re-encrypt to receiver
- Evaluate function on re-encrypted ciphertexts

Extensions of security model

- Eval is public $\rightarrow$ no changes up to correctness extension
+ New multi-target IND-CPA $\rightarrow$ tailored to our HPRE usage

HPRE - Instantiation

Observation

- Many PRE schemes ElGamal based
- Exponential ElGamal is linearly homomorphic

$$(g^{r_1}, g^{m_1} g^{x r_1}) \cdot (g^{r_2}, g^{m_2} g^{x r_2}) = (g^{r_1 + r_2}, g^{m_1 + m_2} g^{x(r_1 + r_2)})$$

→ Apply this to [AFGH06] PRE scheme

Extend to vectors

- Straight forward extension

+ Reduce ciphertext size via randomness reuse [BBKS07]

HPRE - Instantiation contd.

Decryption

- Yields $m' = g^m$, need to compute $m = \log_g m'$
- Numerical values in order of millions to billions

✓ Entirely practical

Putting the Pieces Together - Output Privacy

Signatures still publicly verifiable

- Possible to verify guesses

→ Blind signature with g^r

- r uniformly random in $\mathbb{Z}_p$
- Obtaining r not efficiently possible

✓ However, obtaining g^r (resp. $e(g^r, g)$) sufficient

Conclusions

New notion of HPRA

- ✓ Multi-source data aggregation under independent keys
- ✓ End-to-end authenticity and verifiability of computations
- ✓ Support for general functions

Two modular HPRA construction

- ✓ Construction for linear functions
- ✓ Novel linearly homomorphic MAC
- ✓ Strong privacy via the new notion of HPRE

Open Questions

- Instantiation for function beyond linear ones
- Signature instead of MAC for receivers
- Construction in standard model

Thank you.

Full version available as IACR ePrint Archive Report 2017/086

✉ david.derler@iaik.tugraz.at 🐦 @dderler

Supported by EU H2020 prisma cloud and CREDENTIAL Secure Cloud Identity Wallet