USENIX Security 2024 – 08/16/2024

Privacy-Preserving Data Aggregation with Public Verifiability Against Internal Adversaries

Marco Palazzo, Florine W. Dekker, Alessandro Brighente, Mauro Conti, Zeki Erkin

Cyber Security Group, Delft University of Technology

SPRITZ Security and Privacy Research Group, Università di Padova

Data aggregation
Examples

- Process and summarize data to extract insights from it
- Examples
    - Censuses
    - COVID-19
    - Smart grids
    - Medical data

https://www.economist.com/books-and-arts/2020/04/16/a-lively-and-enlightening-history-of-the-census

Privacy concerns

Medical data

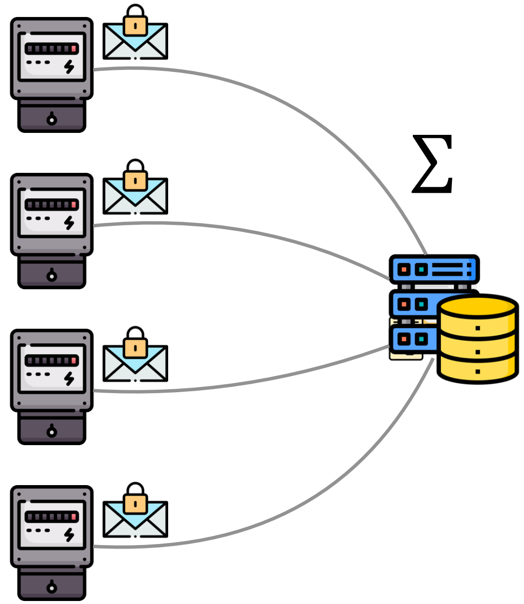
- Fines
- Lack of trust leads to harmful behaviours
    - Not disclosing “embarrassing” conditions
    - Self-treating

Why we need verifiability

- Intermediate stations in smart grids may be hacked
- Reporters are not trusted
- Incorrect medical data may lead to wrong diagnoses

Verifiable privacy-preserving data aggregation

- Compute a statistic from a set of private inputs
- No unauthorized party learns the individual inputs
- Only the final result is revealed
- The correctness of the result can be verified



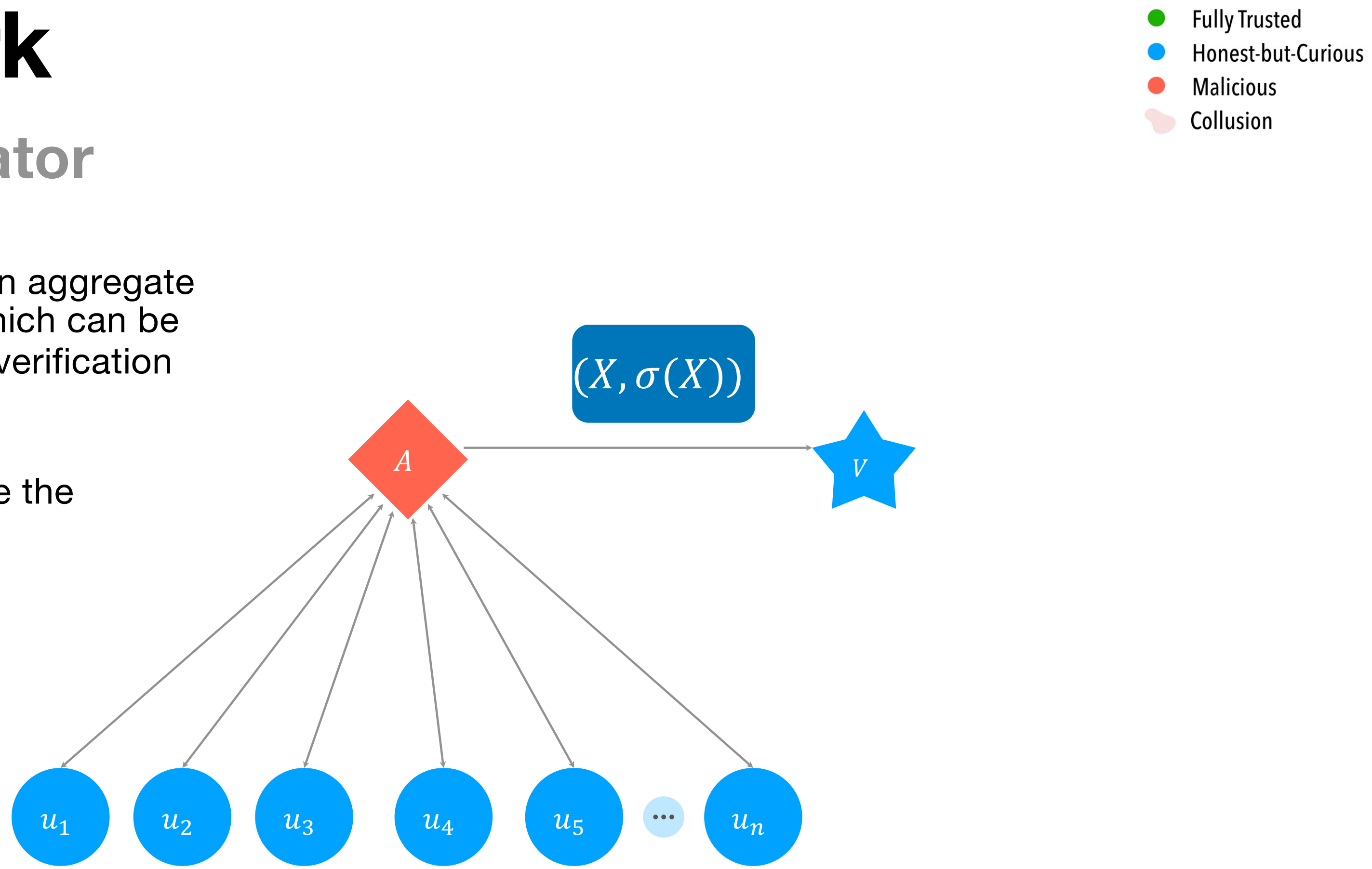
TU Delft

Related work

Malicious aggregator

The aggregator must provide an aggregate signature of the summation, which can be verified by anyone holding the verification key.

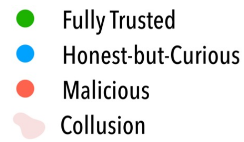
The aggregator cannot produce the signature by itself.



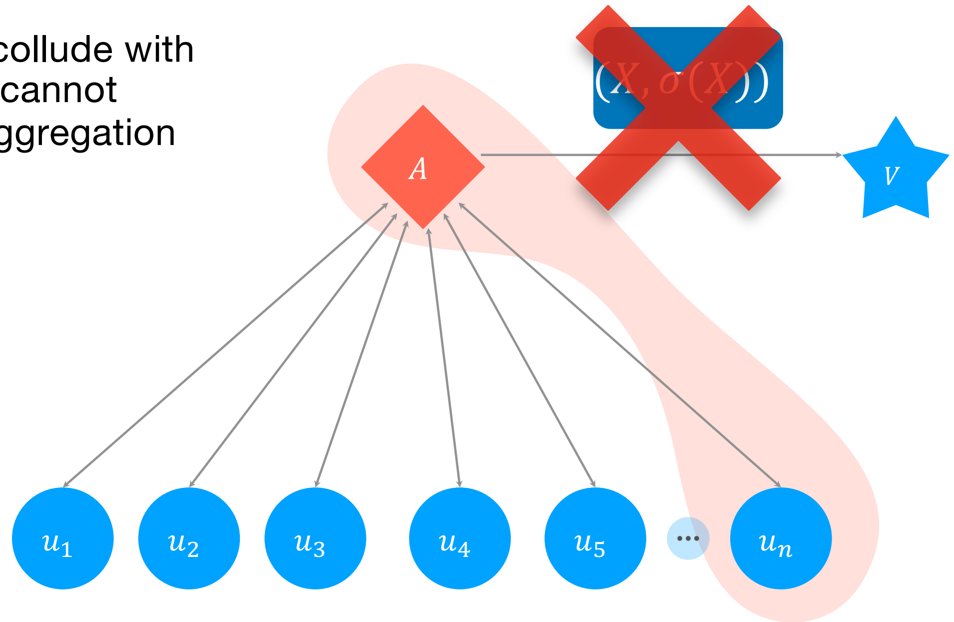
- Iraklis Leontiadis, Kaoutar Elkhiyaoui, Melek Önen, and Refik Molva. PUDA - privacy and unforgeability for data aggregation. In Michael K. Reiter and David Naccache, editors, Cryptology and Network Security 14th International Conference, CANS 2015, Marrakesh, Morocco, December 10-12, 2015, Proceedings, volume 9476 of Lecture Notes in Computer Science, pages 3–18. Springer, 2015. doi:10.1007/978-3-319-26823-1_1.
- Bence Gabor Bakondi, Andreas Peter, Maarten H. Everts, Pieter H. Hartel, and Willem Jonker. Publicly verifiable private aggregation of time-series data. In 10th International Conference on Availability, Reliability and Security, ARES 2015, Toulouse, France, August 24-27, 2015, pages 50–59. IEEE Computer Society, 2015. doi:10.1109/ARES.2015.82.

Related work

Malicious aggregator and users



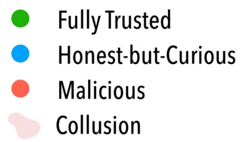
If the aggregator is allowed to collude with at least 1 user, these schemes cannot guarantee the integrity of the aggregation anymore

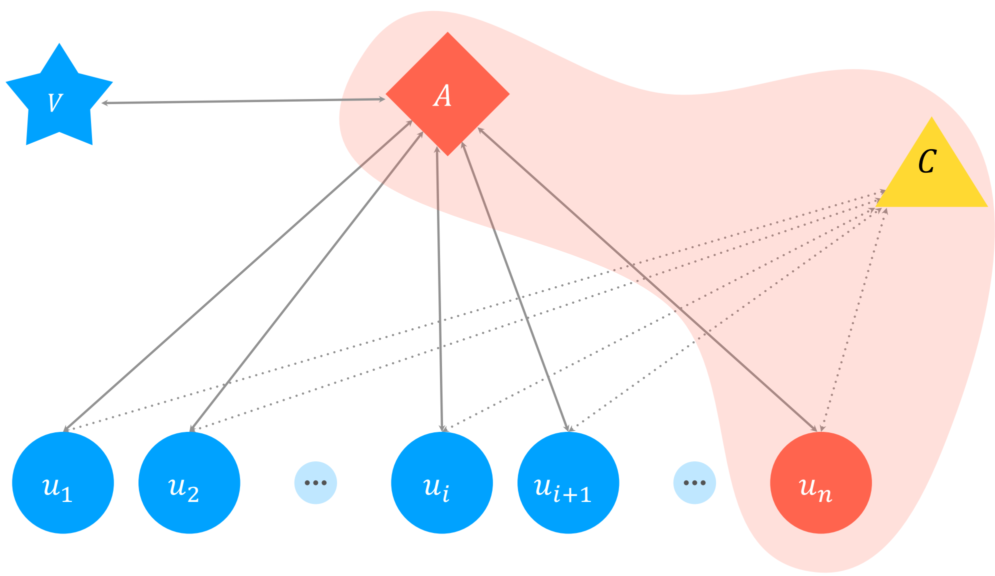


- Iraklis Leontiadis, Kaoutar Elkhiyaoui, Melek Önen, and Refik Molva. PUDA - privacy and unforgeability for data aggregation. In Michael K. Reiter and David Naccache, editors, Cryptology and Network Security 14th International Conference, CANS 2015, Marrakesh, Morocco, December 10-12, 2015, Proceedings, volume 9476 of Lecture Notes in Computer Science, pages 3–18. Springer, 2015. doi:10.1007/978-3-319-26823-1_1.
- Bence Gabor Bakondi, Andreas Peter, Maarten H. Everts, Pieter H. Hartel, and Willem Jonker. Publicly verifiable private aggregation of time-series data. In 10th International Conference on Availability, Reliability and Security, ARES 2015, Toulouse, France, August 24-27, 2015, pages 50–59. IEEE Computer Society, 2015. doi:10.1109/ARES.2015.82.

Related work

Malicious aggregator and users





- LL21 introduces an additional honest-but-curious party called the Converter to help with the construction of the signatures.
- The verifier must be a fully-trusted external party.
- Only pairwise collusions between each party are permitted. However, a flaw in the protocol may allow an aggregator that colludes with 1 user to forge arbitrary signatures.

TU Delft

Leontiadis, Iraklis, and Ming Li. "Secure and Collusion-Resistant Data Aggregation from Convertible Tags." International Journal of Information Security 20, no. 1 (February 2021): 1–20. https://doi.org/10.1007/s10207-019-00485-4.

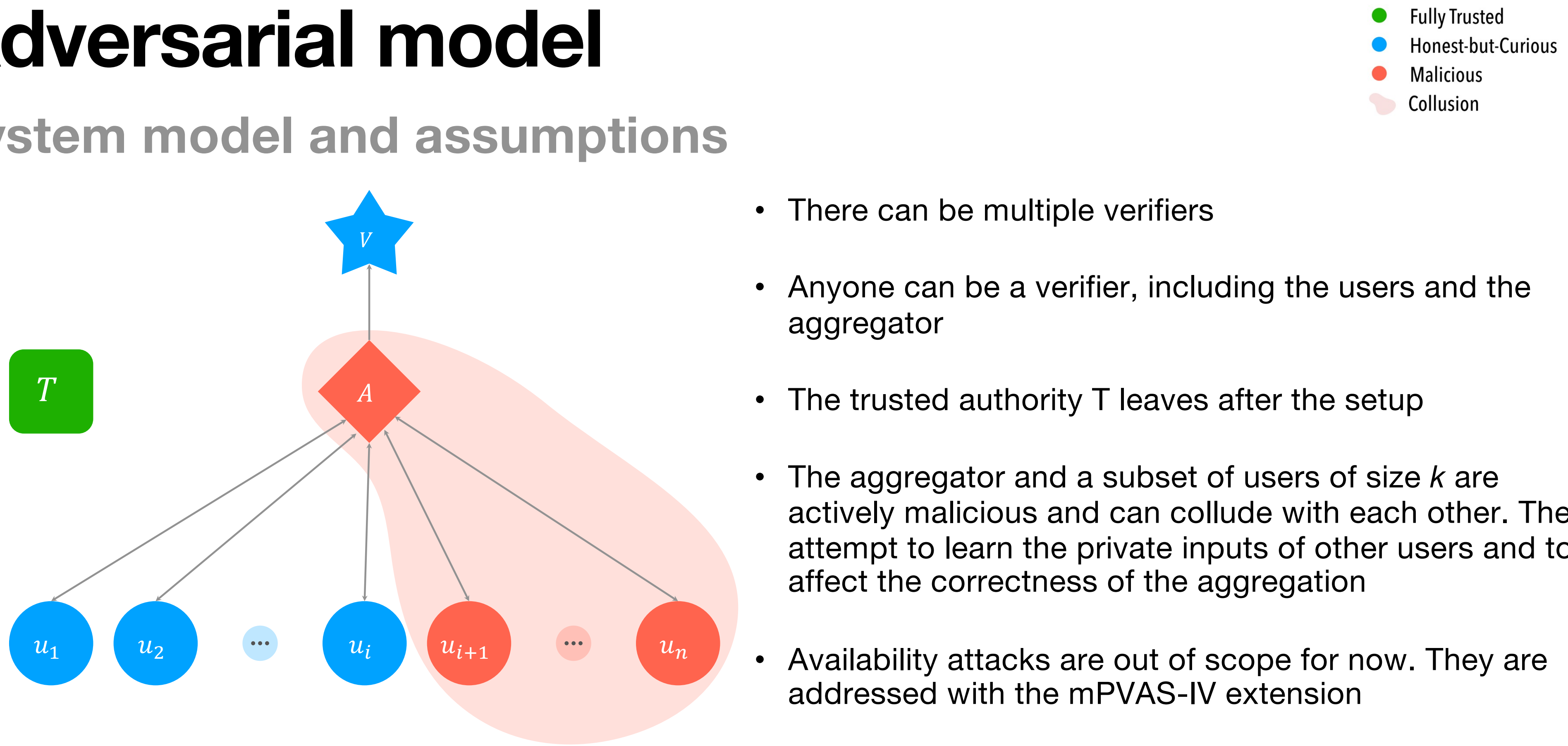
Our goal

- A privacy-preserving data aggregation scheme with **public verifiability** achieve
  - **Confidentiality** of the private inputs
  - **Integrity** and **Authenticity** of the aggregate statistic (the sum)
- with
  - a **malicious aggregator**
  - **multiple malicious users**
- without relying on additional semi-trusted parties during execution.

TU Delft

Adversarial model

System model and assumptions



- There can be multiple verifiers
- Anyone can be a verifier, including the users and the aggregator
- The trusted authority T leaves after the setup
- The aggregator and a subset of users of size *k* are actively malicious and can collude with each other. They attempt to learn the private inputs of other users and to affect the correctness of the aggregation
- Availability attacks are out of scope for now. They are addressed with the mPVAS-IV extension

TU Delft

Our contribution

- **mPAS: Publicly Verifiable Aggregate Signatures with Malicious Participants**
- mPAS+: Reduced communication cost by grouping users.
- mPAS-IV: Detection and removal of malicious users.
- mPAS-UD: Exit strategy without restarting the protocol.

TU Delft

Publicly Verifiable Aggregate Signatures with Malicious Participants (mPVAS)

mPVAS

Goal

- Each user starts from a commitment of this form (initial signature)

Authenticates each user

$$(H(t))^{sk_i} \cdot (g_1)^{x_{i,t}}$$

Identifies the aggregation round

The private input value

TU Delft

mPVAS

Goal

- The goal is to aggregate all submitted signatures

$$(H(t))^{\sum sk_i} \cdot (g_1)^{\sum x_{i,t}}$$

TU Delft

mPVAS

Goal

- Since the generators are public, the input value can easily be modified by multiplying the signature by $g_1^{x'}$
- To prevent this, we can wrap the signature under an additional exponent *s* that must not be disclosed to the aggregator

$$(H(t)^s)^{\sum sk_i} \cdot (g_1^s)^{\sum x_{i,t}}$$

TU Delft

mPVAS

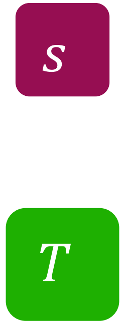
Goal

- mPVAS can be run in parallel to another privacy-preserving data summation scheme
  - mPVAS computes the aggregate signature, the data summation protocol computes the sum of the inputs
- The sum can also be extracted from the signature if the input space is small enough

TU Delft

mPVAS

1. Setup phase



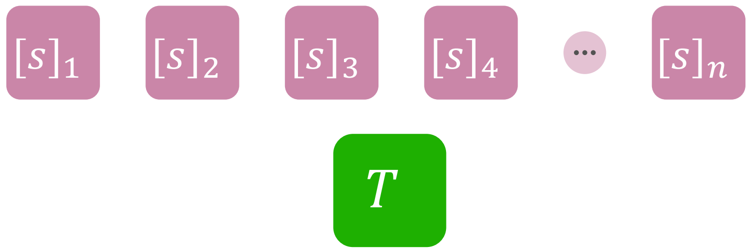
The trusted dealer chooses a random secret $s \in \mathbb{Z}_p$

TU Delft

mPVAS

1. Setup phase

- Users can collude with the aggregator, so we must also protect *s* from them
- Assume at most *k* malicious users, then we can split the secret into *k+1* shares



$(k+1, n)$ - Shamir Secret Sharing

TU Delft

mPVAS

1. Setup phase

mPVAS

1. Setup phase

mPVAS

1. Setup phase

Verification Key

$(g_2^s)^{\sum sk_i}$

g_2^s

T

TU Delft

mPVAS

1. Setup phase

mPVAS

1. Setup phase

mPVAS

1. Setup phase

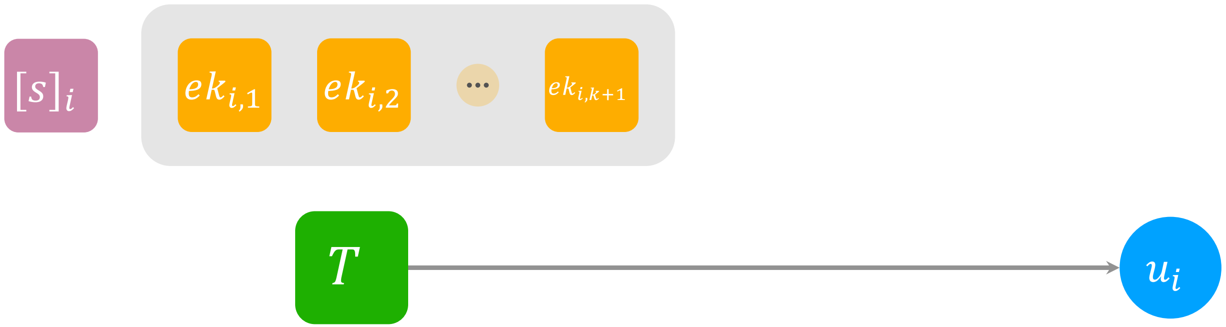
Dealer generates $k + 1$ random keys $ek_{j,i} \in \mathbb{Z}_p$ for each user such that

$$\sum_{j=1}^{n} \sum_{i=1}^{k+1} ek_{j,i} = 0$$

TU Delft

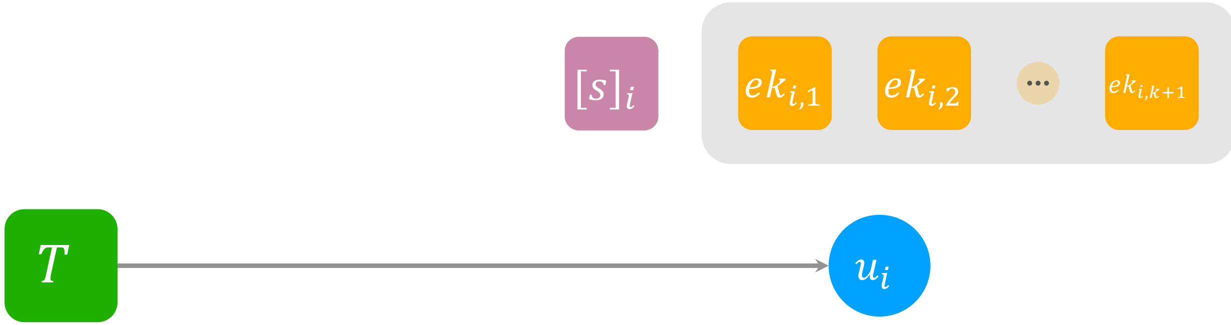
mPVAS

1. Setup phase



mPVAS

1. Setup phase

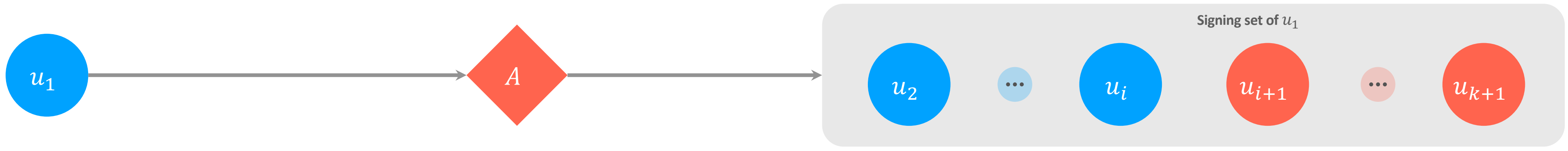


TU Delft

mPVAS

2. Signing phase

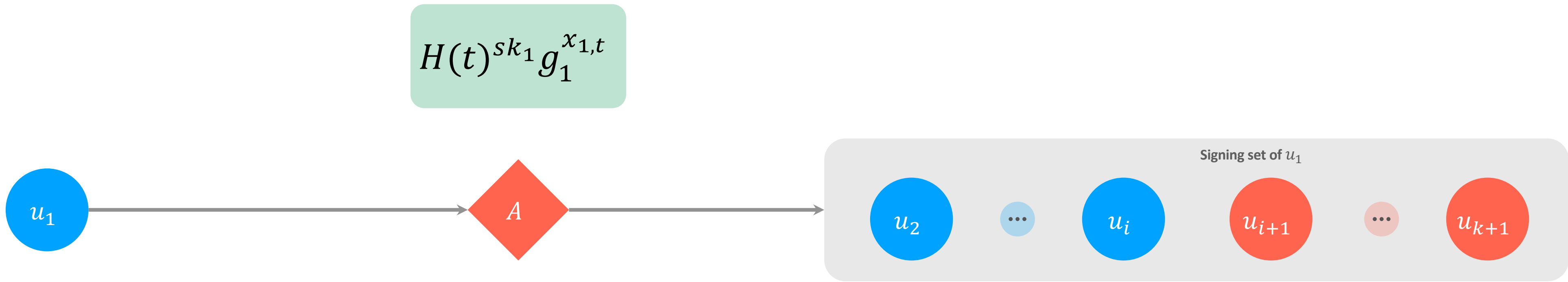
$$H(t)^{sk_1} g_1^{x_{1,t}}$$



mPVAS

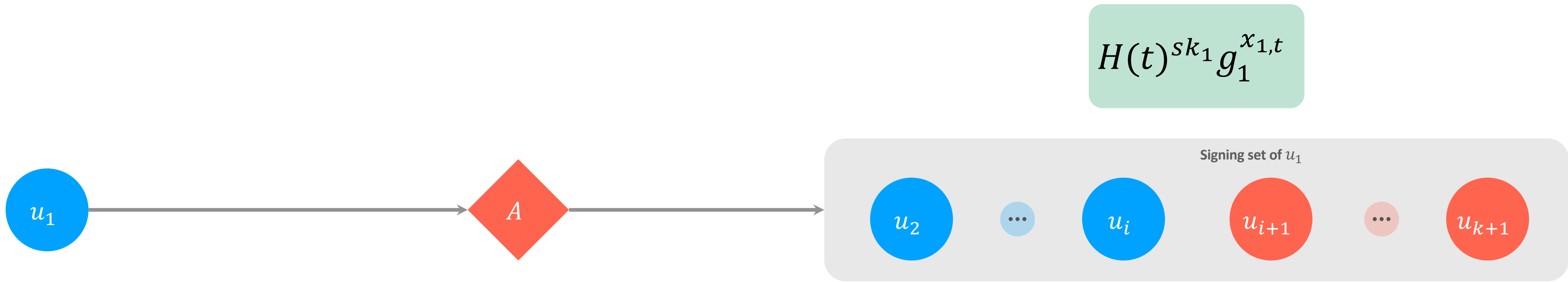
2. Signing phase

Tampering with $g_1^{x_{1,t}}$ here leads to a malformed final signature



mPVAS

2. Signing phase

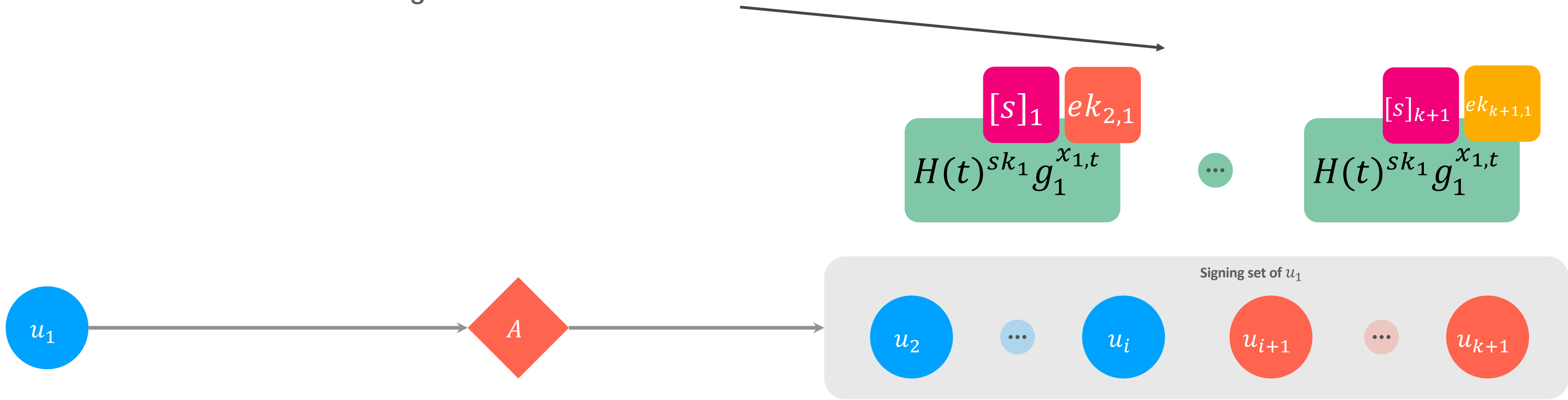


TU Delft

mPVAS

2. Signing phase

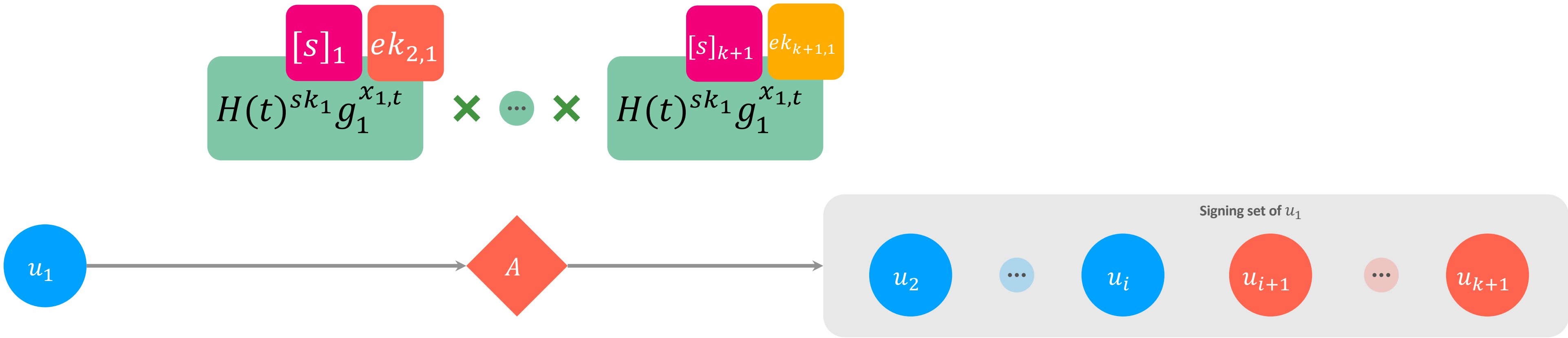
Each user in the signing set adds its share $[s]_j$ of s in the exponent and adds one masking factor $H_1(t)^{ek_{j,1}}$ to the signature



TU Delft

mPVAS

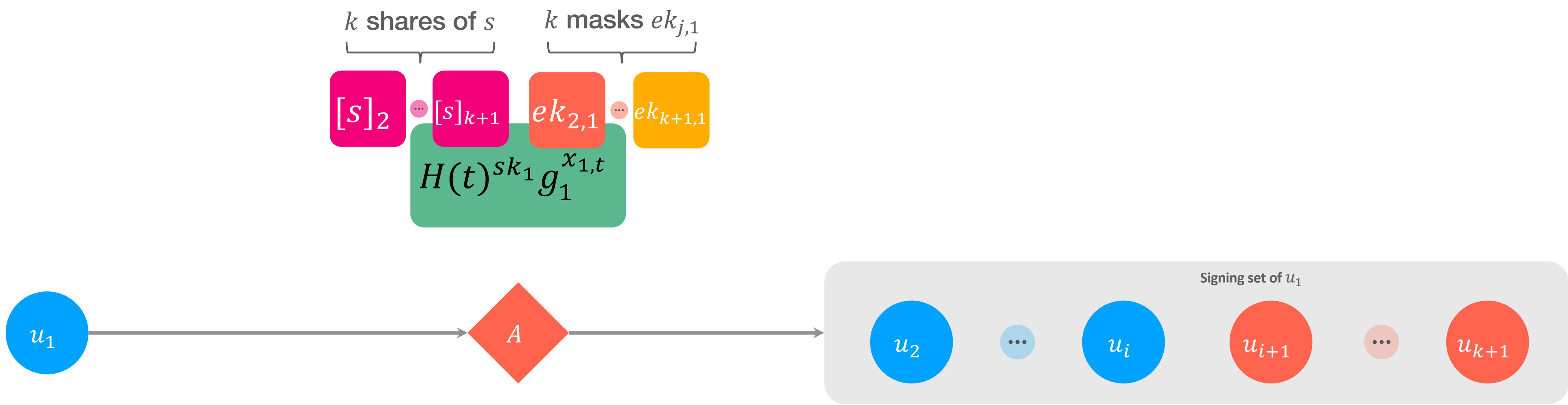
2. Signing phase



TU Delft

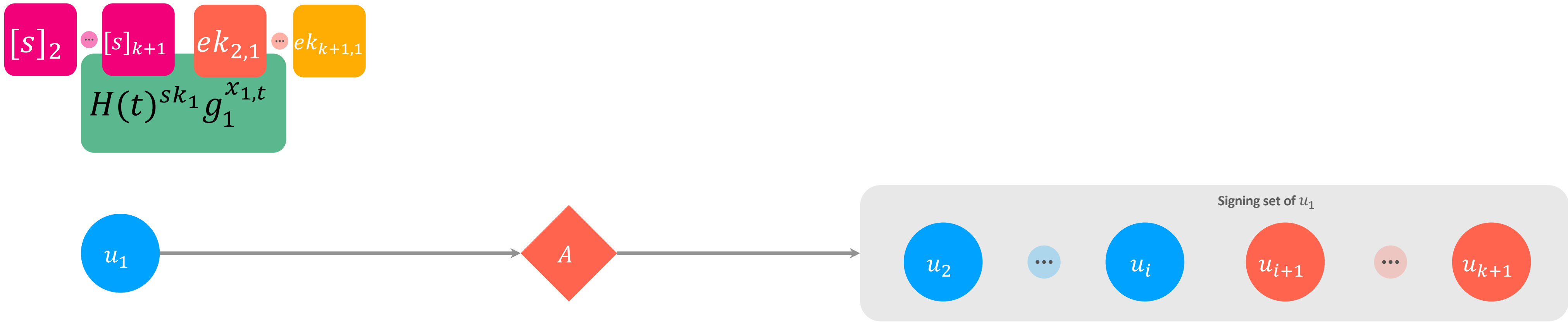
mPVAS

2. Signing phase



mPVAS

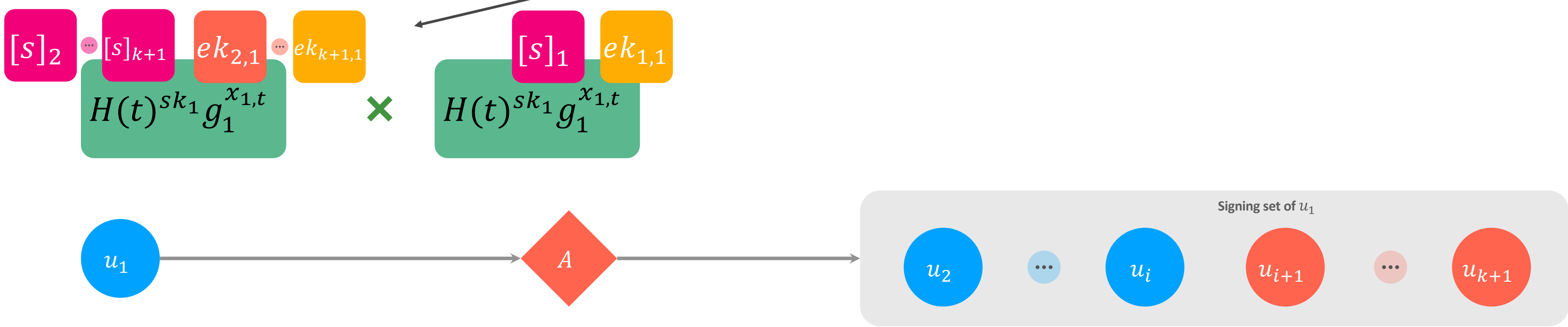
2. Signing phase



mPVAS

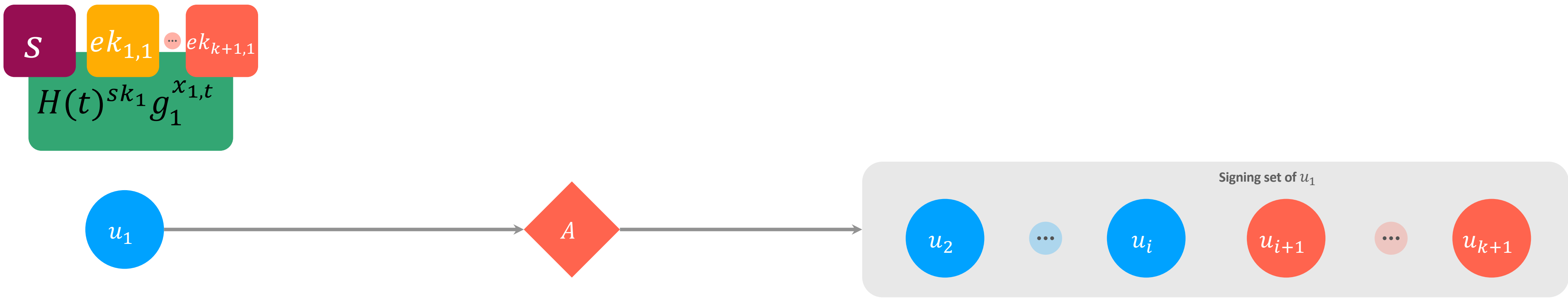
2. Signing phase

If $g_1^{x_{1,t}}$ had been previously tampered with, this multiplication would lead to an invalid final user signature



mPVAS

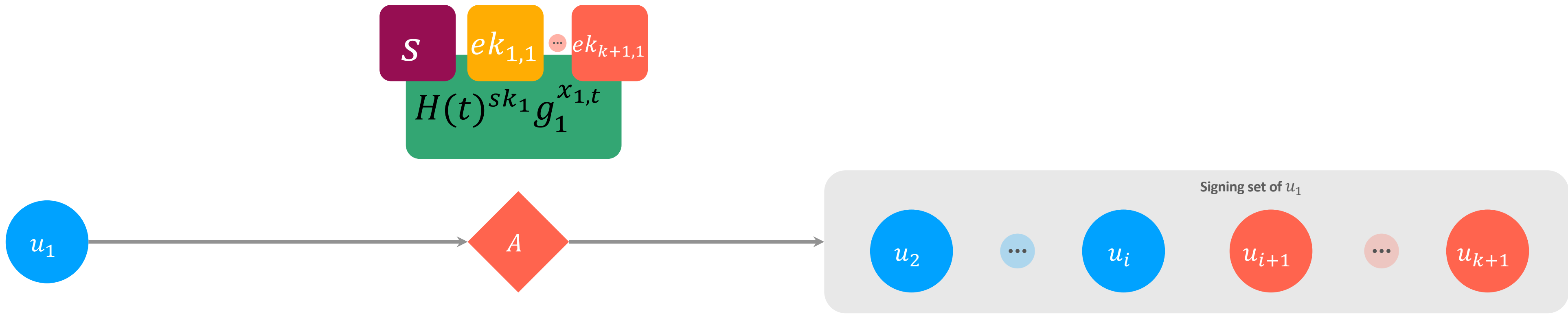
2. Signing phase



TU Delft

mPVAS

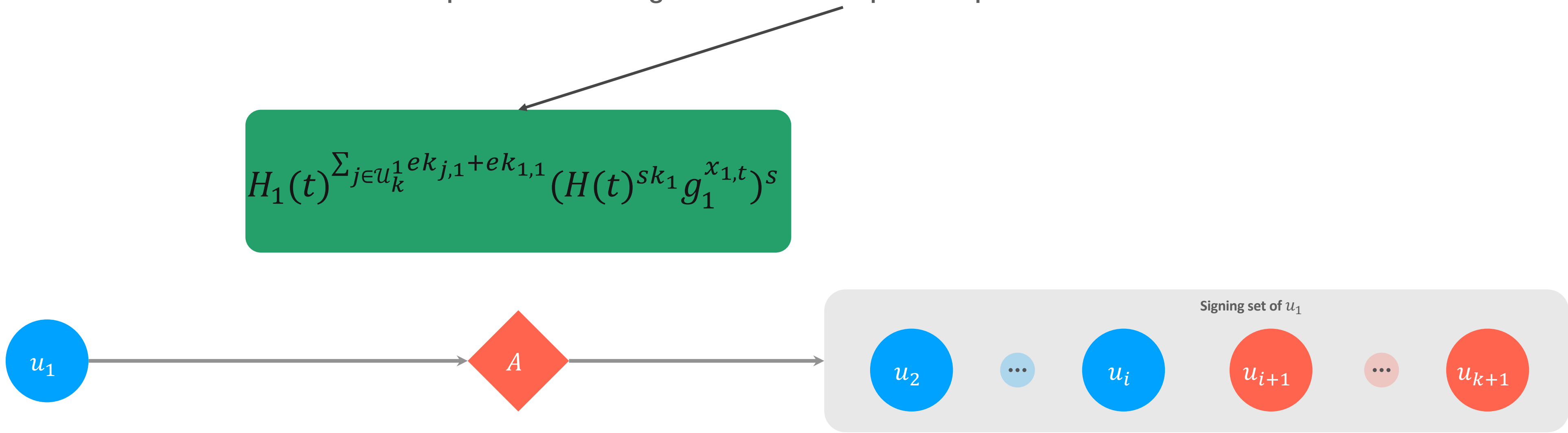
2. Signing phase



mPVAS

2. Signing phase

Complete final user signature. These steps are repeated for each user.



mPVAS

3. Signature aggregation phase

$$\prod_{i=1}^{n} H_1(t)^{\sum_{j\in\mathcal{U}_k^i} ek_{j,i} + ek_{i,i}} \left(H(t)^{sk_i} g_1^{x_{i,t}}\right)^s$$

A

mPVAS

3. Signature aggregation phase

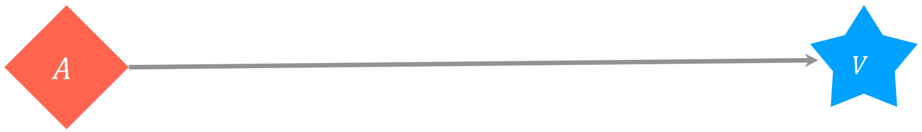
$$(H(t)^s)^{\sum sk_i} \cdot (g_1^s)^{\sum x_{i,t}}$$

A

mPVAS

3. Signature aggregation phase

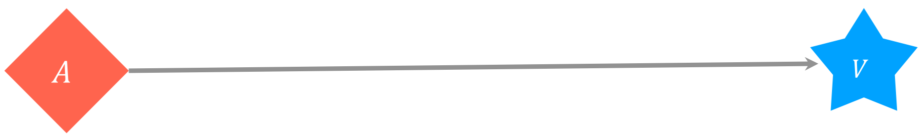
$$(H(t)^s)^{\sum sk_i} \cdot (g_1^s)^{\sum x_{i,t}}$$



mPVAS

3. Signature aggregation phase

$$(H(t)^s)^{\sum sk_i} \cdot (g_1^s)^{\sum x_{i,t}}$$



TU Delft

mPVAS

4. Verification phase

V

Verification Key

$(g_2^s)^{\Sigma sk_i}$ g_2^s

$$e(H(t), (g_2^s)^{\Sigma sk_i}) \cdot e\left(g_1^{\Sigma x_{i,t}}, g_2^s\right)$$

$$= e((H(t)^s)^{\Sigma sk_i}, g_2) \cdot e((g_1^s)^{\Sigma x_{i,t}}, g_2)$$

$$\stackrel{?}{=} e((H(t)^s)^{\Sigma sk_i} \cdot (g_1^s)^{\Sigma x_{i,t}}, g_2)$$

TU Delft

Extensions

- mPAS: Publicly Verifiable Aggregate Signatures with Malicious Participants
- **mPAS+: Reduced communication cost by grouping users.**
- **mPAS-IV: Detection and removal of malicious users.**
- **mPAS-UD: Exit strategy without restarting the protocol.**

Evaluation

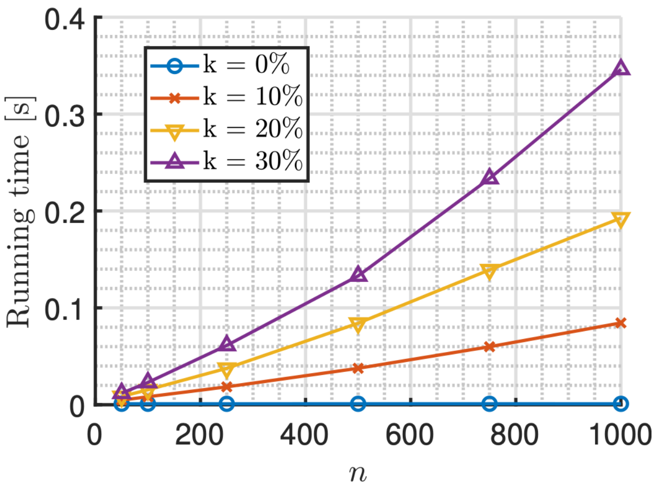
Setup

- Threadripper 7970X CPU... on a single core
- Python, with CHARM for pairing cryptography
- MNT224 as type-3 elliptic curve (112 bits of security)
- Basic implementation, no specific optimizations

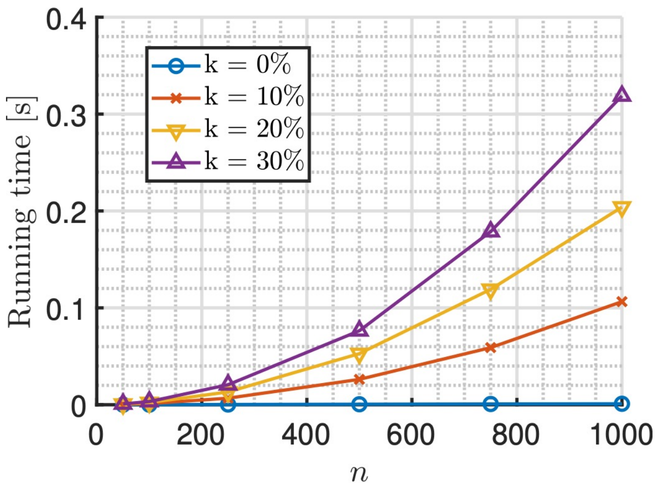
TU Delft

Evaluation

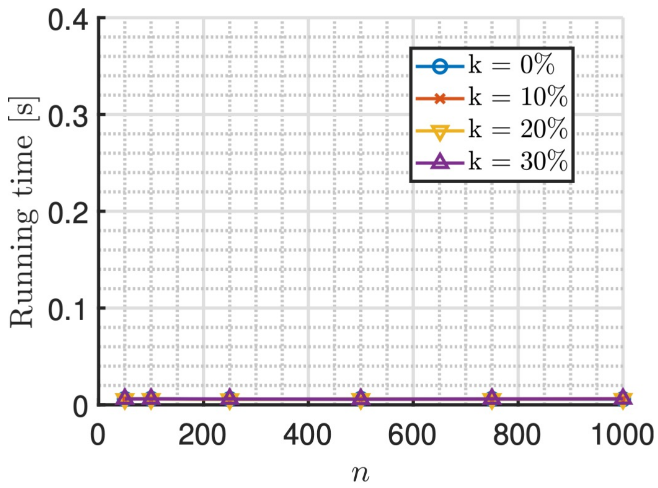
mPVAS – Empirical runtime



Single user



Aggregator



Verifier

TU Delft

Evaluation

Communication complexity

Protocol	Dealer	Agg.	User	Verifier	Ledger
[29]	$O(n)$	$O(1)$	$O(1)$	-	no
[34]	$O(1)$	$O(1)$	$O(1)$	-	$O(n)$
[37]	$O(n)$	$O(n^2)$	$O(n)$	-	no
mPVAS	$O(n)$	$O(kn)$	$O(k)$	-	no
mPVAS+	$O(n)$	$O(cn)$	$O(c)$	-	no
mPVAS-IV	$O(n)$	$O(kn)$	$O(k)$	-	no
mPVAS-UD	$O(n)$	$O(kn)$	$O(k)$	-	no

[29] Iraklis Leontiadis and Ming Li. Secure and collusion-resistant data aggregation from convertible tags. Int. J. Inf. Sec., 20(1):1–20, 2021. doi:10.1007/s10207-019-00485-4.
[34] Dimitris Mouris and Nektarios Georgios Tsoutsos. Masquerade: Verifiable multi-party aggregation with secure multiplicative commitments. IACR Cryptol. ePrintArch., page 1370, 2021. URL https://eprint.iacr.org/2021/1370
[37] Yanli Ren, Yerong Li, Guorui Feng, and Xinpeng Zhang. Privacy-enhanced and verification-traceable aggregation for federated learning. IEEE Internet Things J., 9(24):24933–24948, 2022. doi:10.1109/JIOT.2022.3194930.

TU Delft

Conclusion

Recap

- **Publicly verifiable** summation with input **confidentiality** and output **integrity**
- First scheme against **collusion** of aggregator and multiple **malicious** users
- Three extensions: improved communication, input validation, and availability
- Fast for practical applications (even without any optimisations)

TU Delft

Thank you very much for your time!

Special thanks to the anonymous reviewers.