Confidential and verifiable telemetry using Zero-Knowledge Proofs

Janwillem Swalens, Paarijaat Aditya,
Lode Hoste, Emad Heydari Beni,
Majid Salehi

7 May 2026 – zkSummit

Use cases of ZKP

Payments & Blockchain

e.g. Zcash: shielded transactions,
zk-Rollup

Identity & Credentials

e.g. Verifiable Credentials,
Polygon ID: selective disclosure

Voting & Governance

e.g. ZKVote,
DAO governance

Privacy-preserving analytics

e.g. zkTLS,
Private set intersection

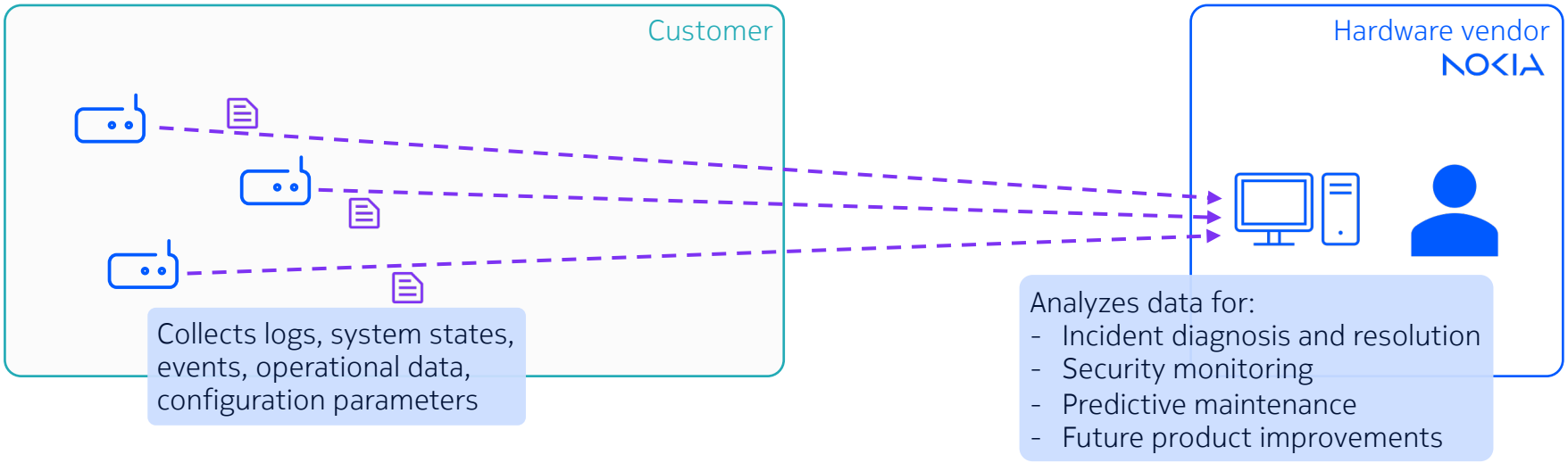
zkML & Verifiable Compute

e.g. zkML, zkCNN: verifiable ML
zkVMs: general-purpose computation

Telemetry

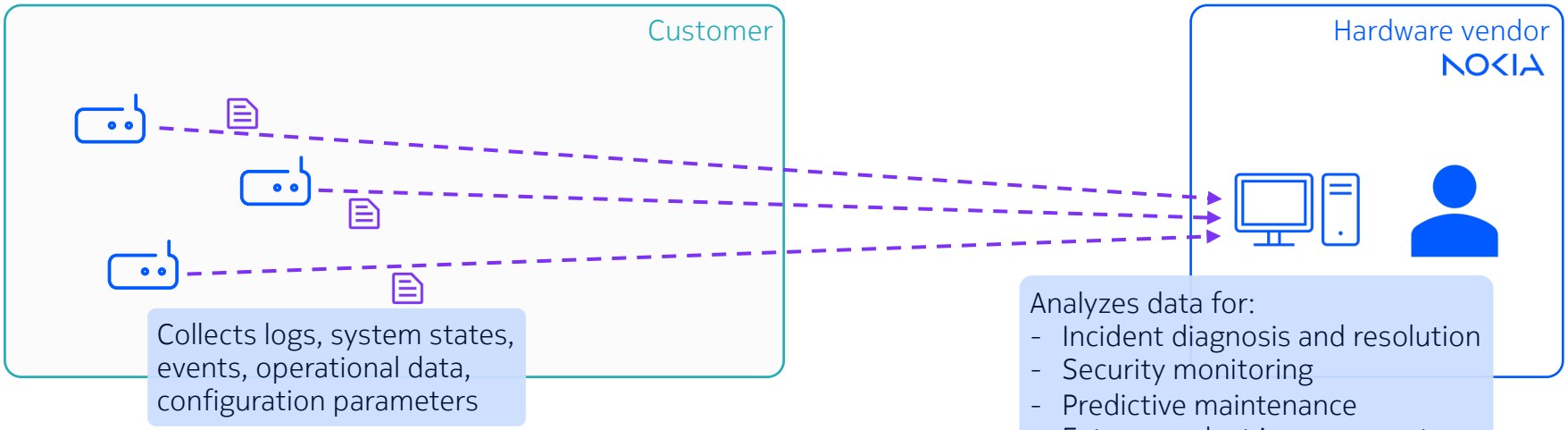
Our work

Telemetry set-up



NOKIA BELL LABS

The customer cares about confidentiality

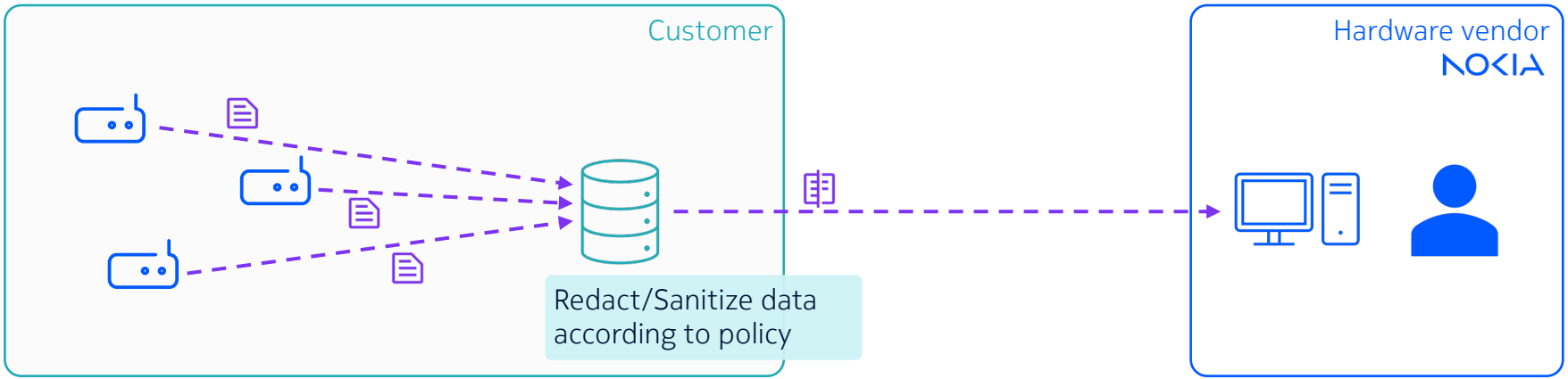


Customer does not want to share **commercially-sensitive** or **privacy-sensitive** data.
E.g. IP/MAC addresses, names of systems, information about competitors' devices

Customer may define a **policy** of acceptable data use and transformations.
E.g. redact IP addresses, only track critical errors

NOKIA BELL LABS

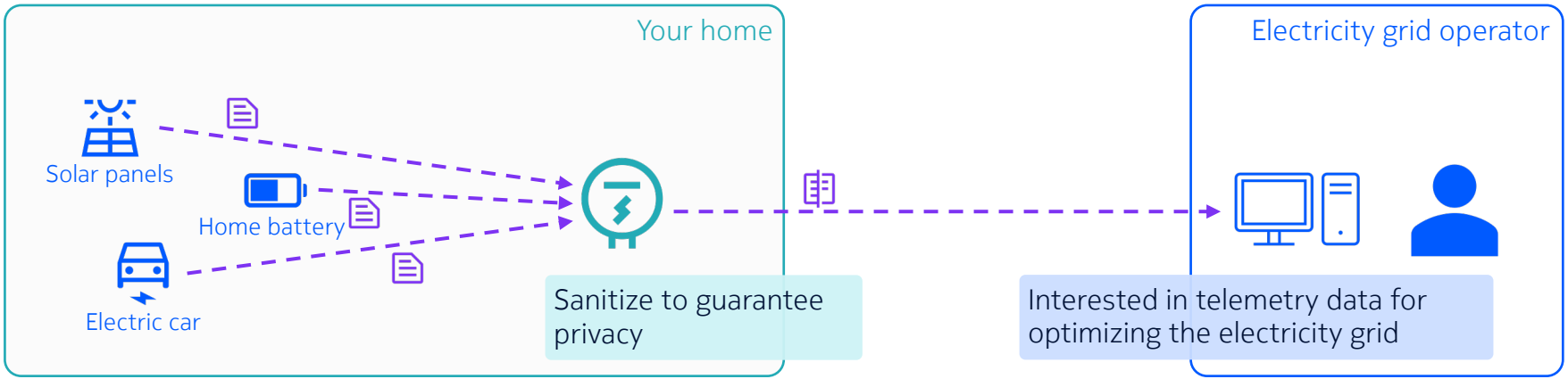
The vendor cares about authenticity and provenance



How does the vendor know this data was processed correctly?
Not doing so may be in the customer's interest!
The data should be **authentic**.

Telemetry on end-user devices

E.g. energy management in your home



You do not want to share **private** data.
E.g. when you are home, which devices you are using

How does the operator know this data was not manipulated?
This may be in the customer's interest!
E.g. shift consumption to off-peak hours
The data should be **authentic**.

Swalens, Hoste, Heydari Beni, and Trappeniers (2024). "zkStream: a Framework for Trustworthy Stream Processing." In *Middleware '24*. https://doi.org/10.1145/3652892.3700763
→ Open sourced at https://github.com/SSR-Bell/zkstream

Requirements

for trustworthy telemetry systems

Vendor manufactures hardware or software that is installed at the customer's premise.

The customer desires:

Confidentiality: sensitive data must be redacted or sanitized

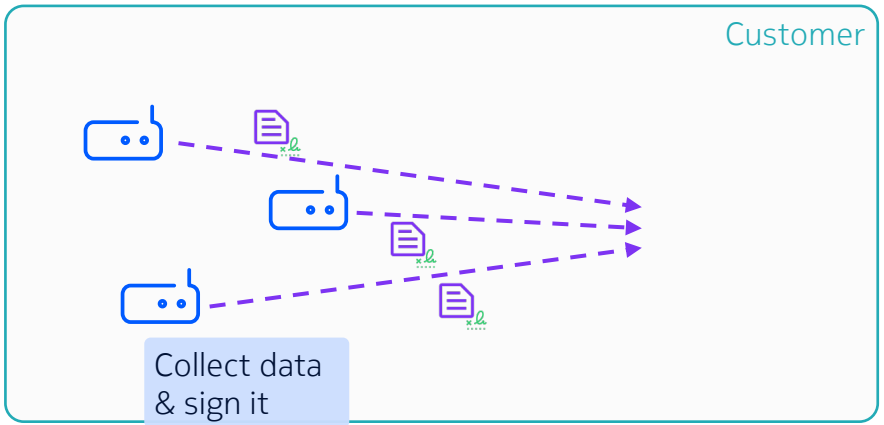
Computational integrity: sanitization must happen according to predefined algorithm (encoding the policy)

The vendor desires:

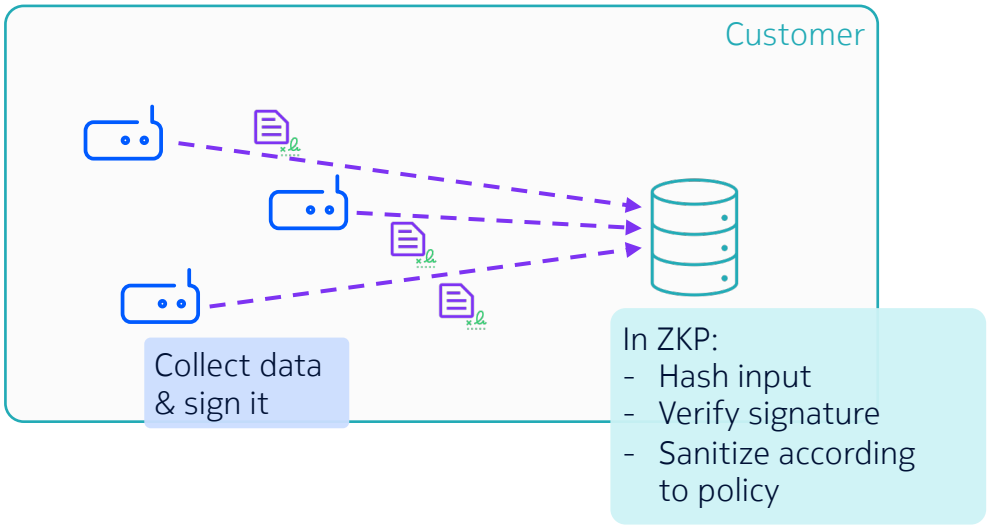
Authenticity: original device data was used, no tampering with input

Computational integrity: sanitization happened according to agreed-upon algorithm, no tampering with output

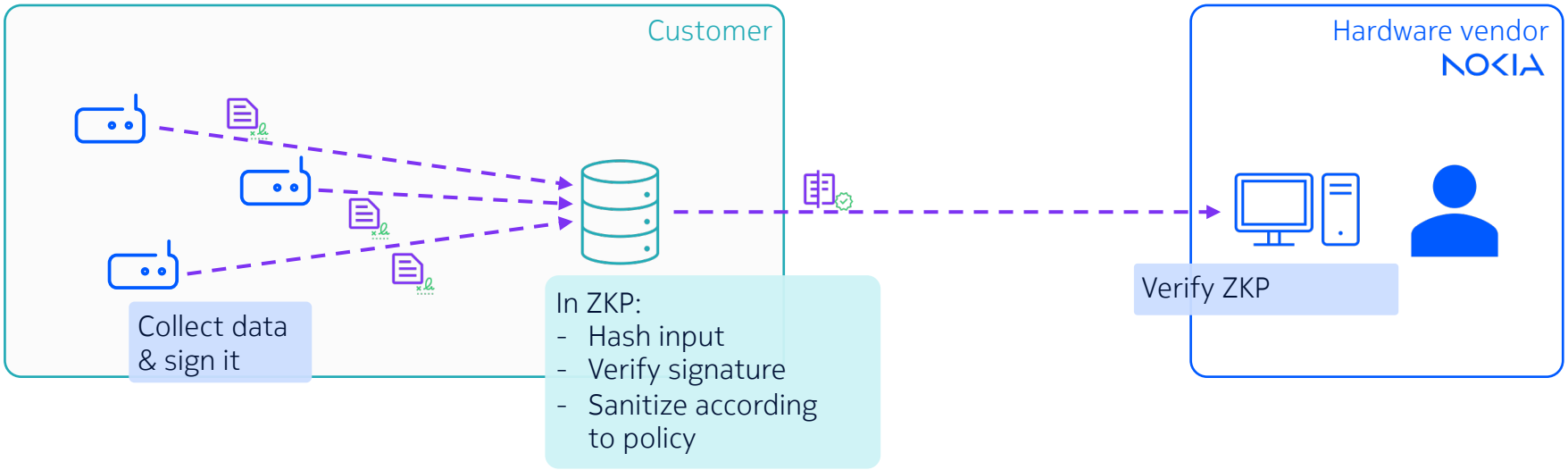
Protocol: 1. Device signs raw data



Protocol: 2. Prover redacts data and generates ZKP



Protocol: 3. Verifier verifies



This protocol guarantees

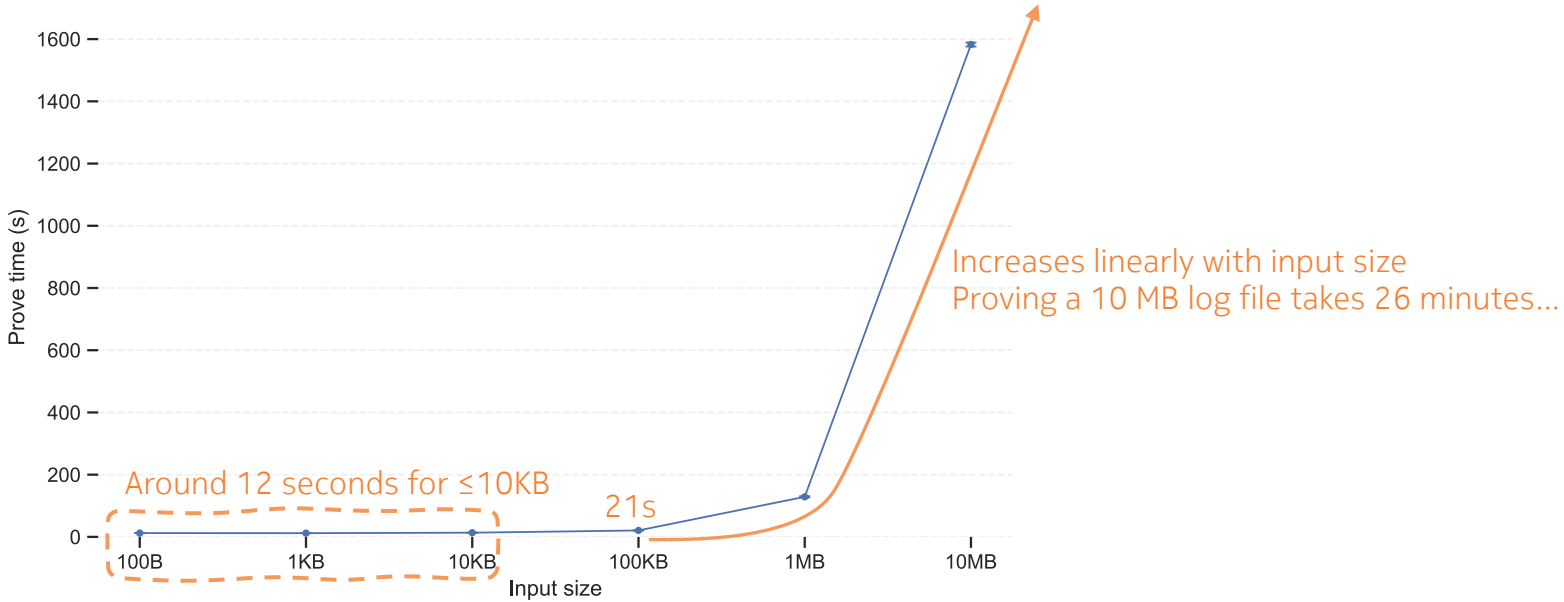
Confidentiality
sensitive data is sanitized by customer

Computational integrity
ZKP proves that sanitization happened correctly

Authenticity
signature proves that original input data was used;
ZKP proves that output was not manipulated

Performance is a concern for large log files

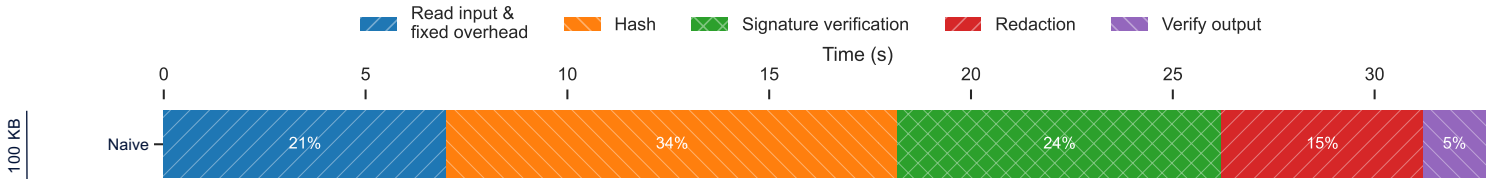
We deal with large log files!
Some files are over 60 MB: > 1 million lines of logs, command outputs, diagnostics, etc.
(For resolving incidents, more information is better.)



Results on Apple M4 Max, 64 GB RAM
Using OpenVM v1.5.0

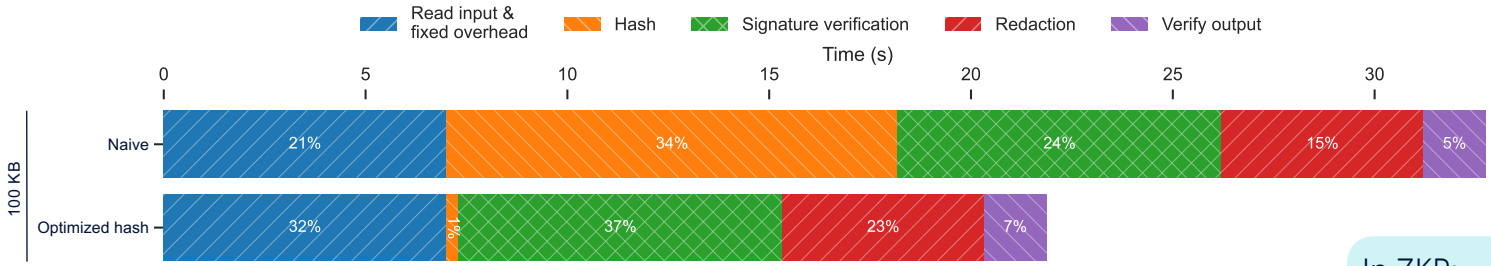
NOKIA BELL LABS

Performance breakdown



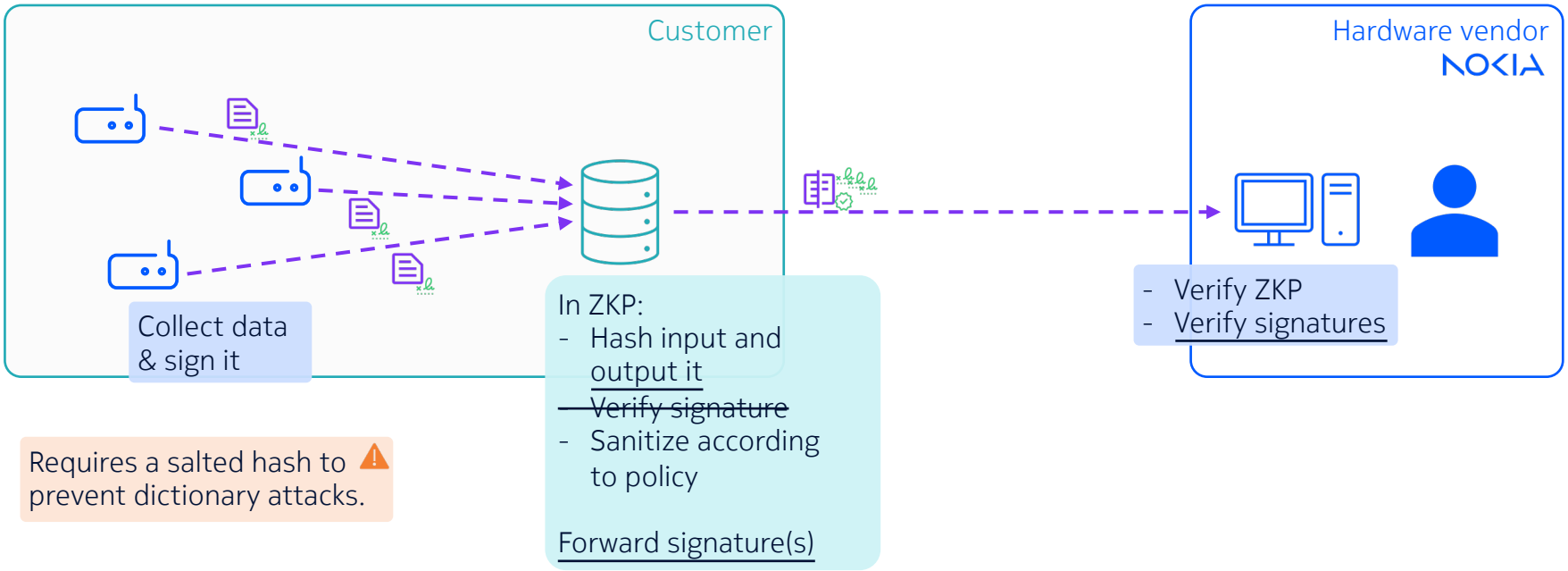
In ZKP:
- Read inputs
- Hash input
- Verify signature
- Sanitize according to policy
- Verify if actual output matches expected output

Performance breakdown



In ZKP:
- Read inputs
- Hash input
- Verify signature
- Sanitize according to policy
- Verify if actual output matches expected output

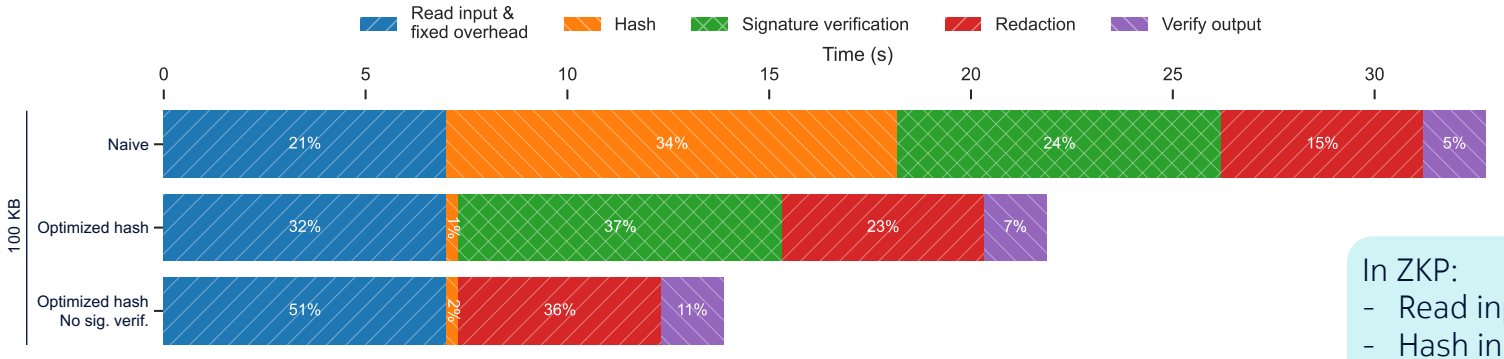
Outsource signature verification to final verifier



Swalens, Hoste, Heydari Beni, and Trappeniers (2024). "zkStream: a Framework for Trustworthy Stream Processing." In *Middleware '24*. https://doi.org/10.1145/3652892.3700763

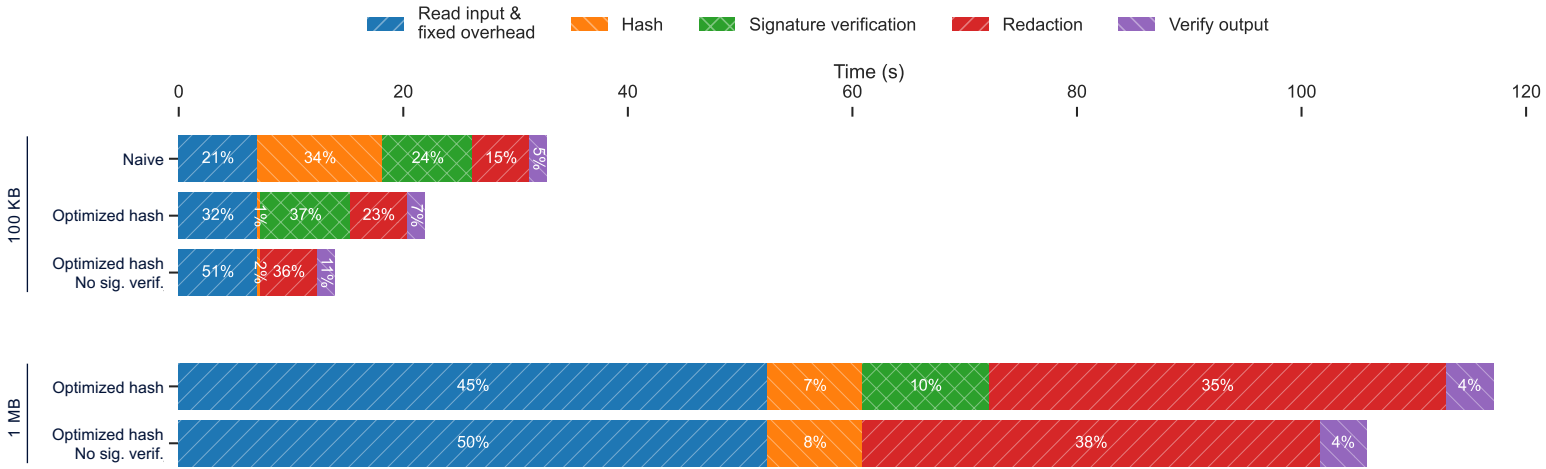
NOKIA BELL LABS

Performance breakdown

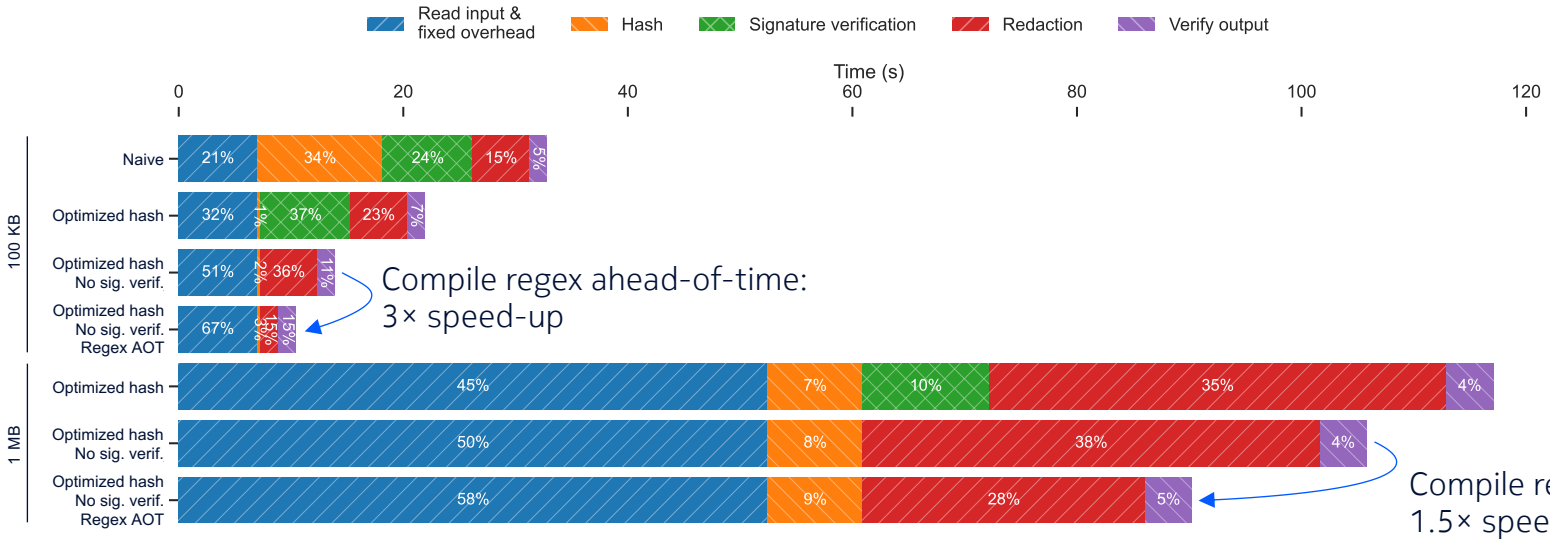


In ZKP:
- Read inputs
- Hash input
- ~~Verify signature~~
- Sanitize according to policy
- Verify if actual output matches expected output

Performance breakdown



Redaction using regular expressions

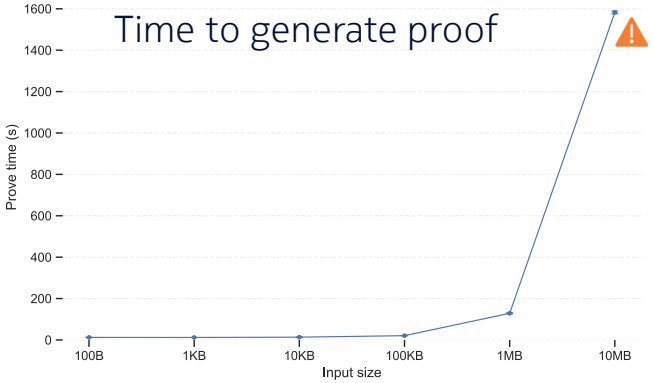


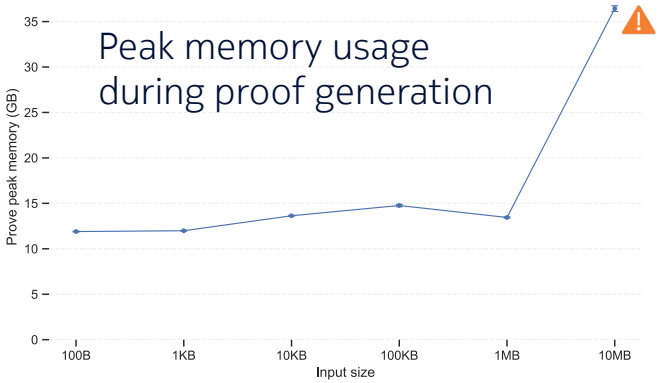
Work in progress: specialized instructions and optimized combination with commitment
⇒ ~15× speed-up compared to original

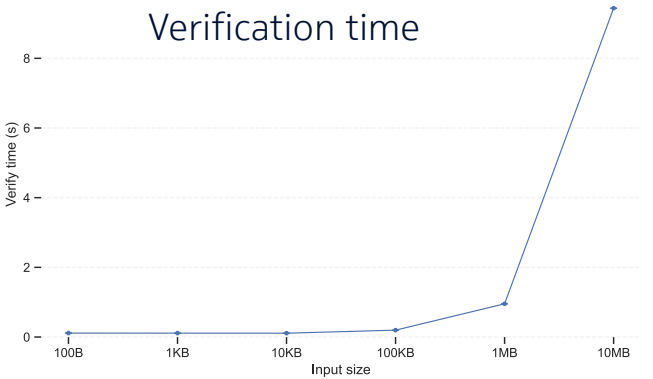
Related work:

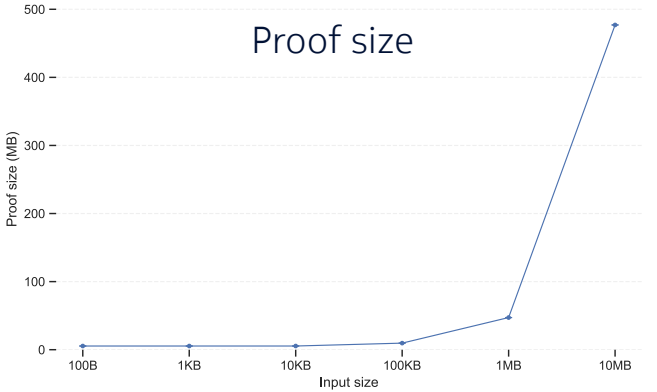
- Raymond et al. (2023). "Efficient zero knowledge for regular language." In *International Conference on Security and Privacy in Communication Systems*.
- Luo et al. (2023). "Privacy-preserving regular expression matching using nondeterministic finite automata." In *Cryptology ePrint Archive*.
- https://github.com/zkemail/zk-regex

What about: memory usage, verification time, proof size?





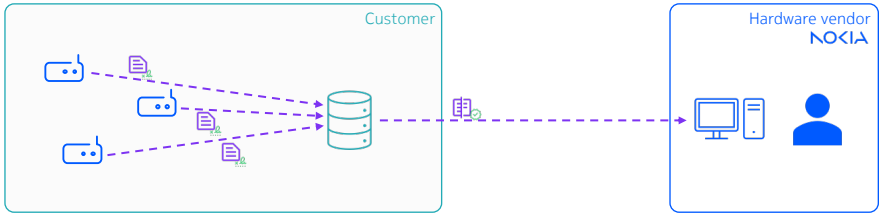




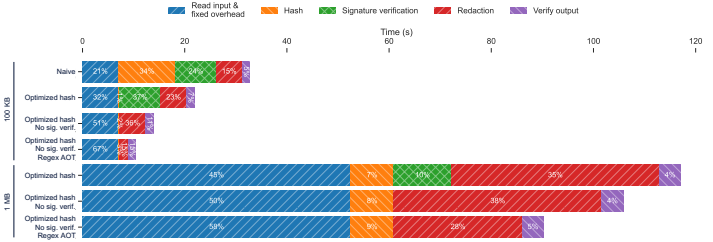
Results on Apple M4 Max, 64 GB RAM – real systems are more limited!
Using OpenVM v1.5.0

Summary & open challenges

We have been exploring a novel use of **ZKPs for verifiable and confidential telemetry**, guaranteeing **confidentiality** of sensitive data for the customer, **authenticity** of the data to the hardware vendor, and **computational integrity** of the predefined sanitization algorithm.



This has required many tricks and optimizations to **improve performance**, and will require even more **for large inputs**.



- Hashing, commitment to input
- Signature verification
- Parsing data
- Regular expressions
- Encryption/Decryption

Other challenges:

- Long-term support (decades!)
- Standardization?

NOKIA
BELL
LABS

Copyright and confidentiality

NOKIA BELL LABS