# ZKFlow: ZKP on Corda

Alexey Koren

ZKProof5, November 16, Tel Aviv

ING

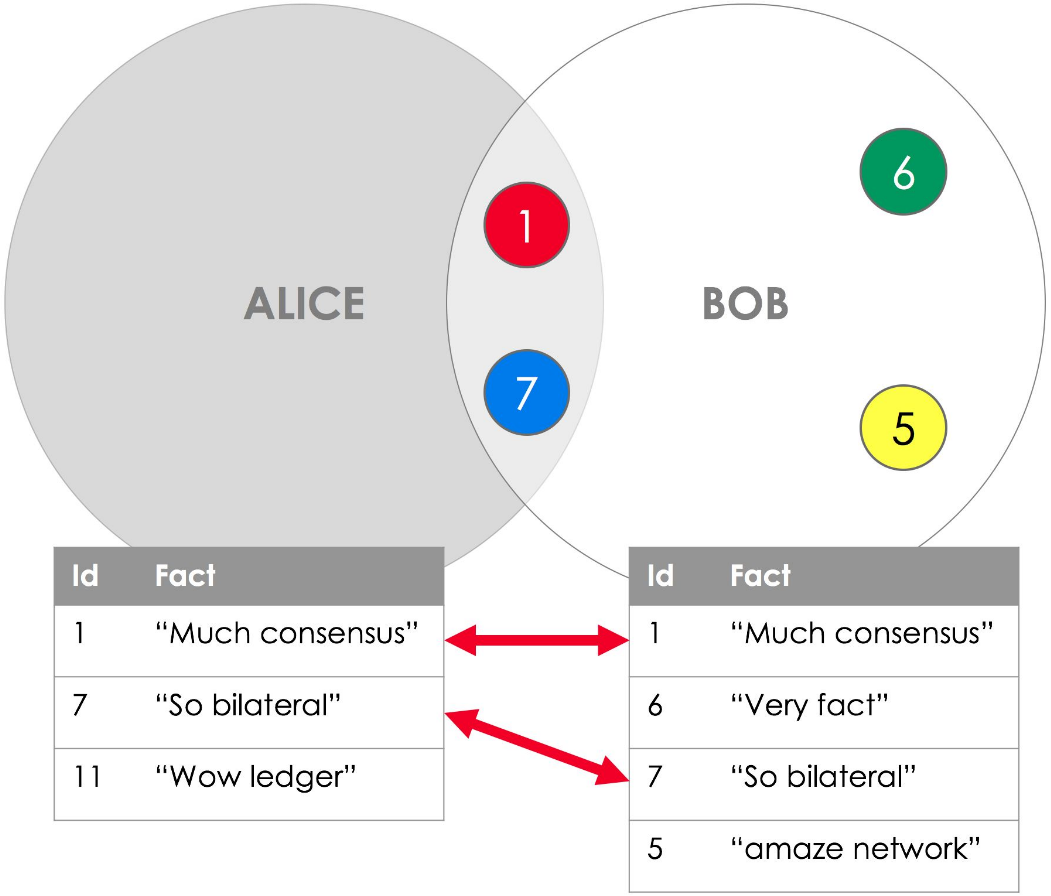
# Corda privacy model

DAG UTXO

Transaction data (and chain history) is visible only to participants

Notary service provides protection from double-spending

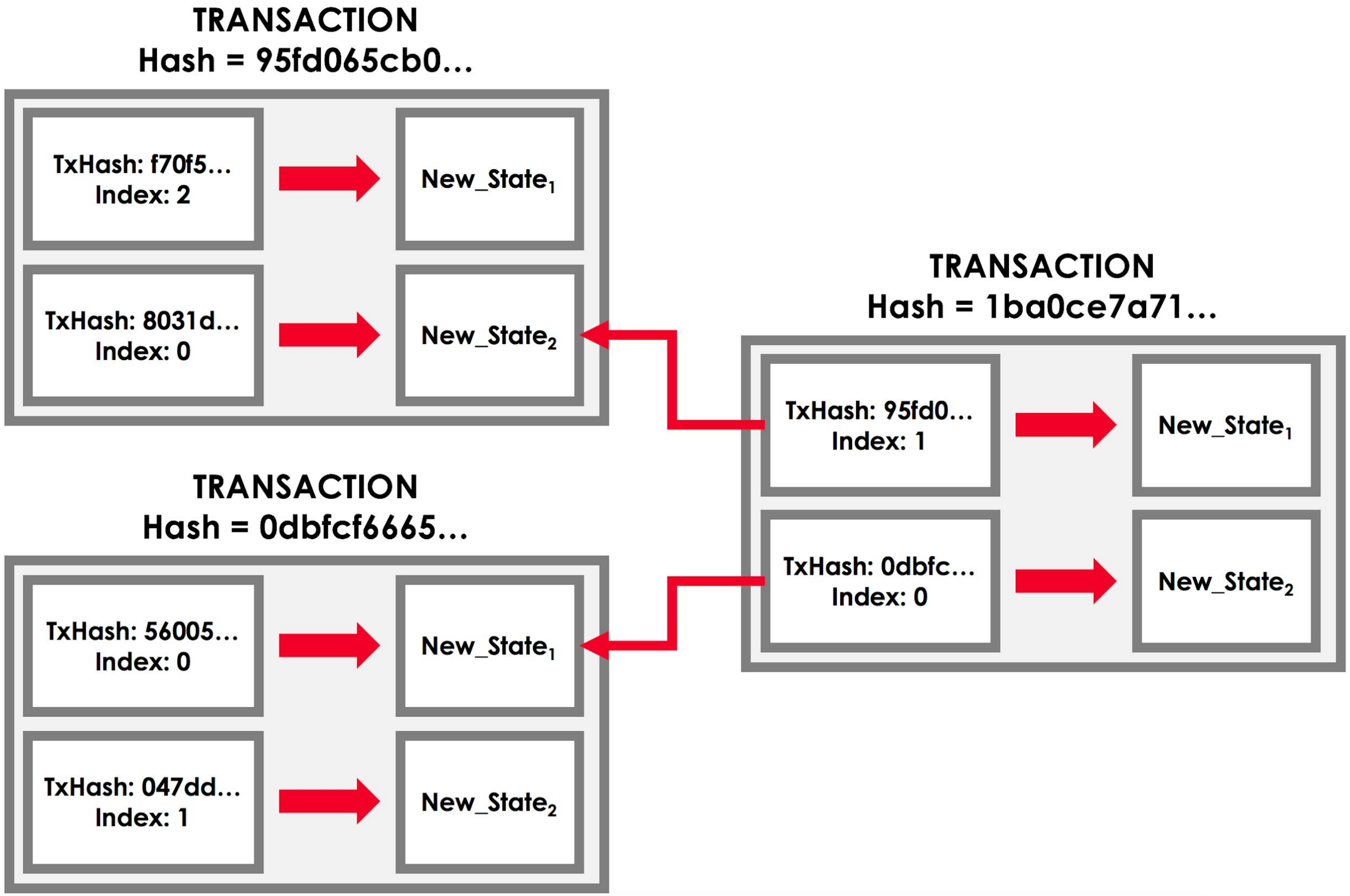
Validating vs non-validating notary



| Id | Fact |
|---|---|
| 1 | "Much consensus" |
| 7 | "So bilateral" |
| 11 | "Wow ledger" |

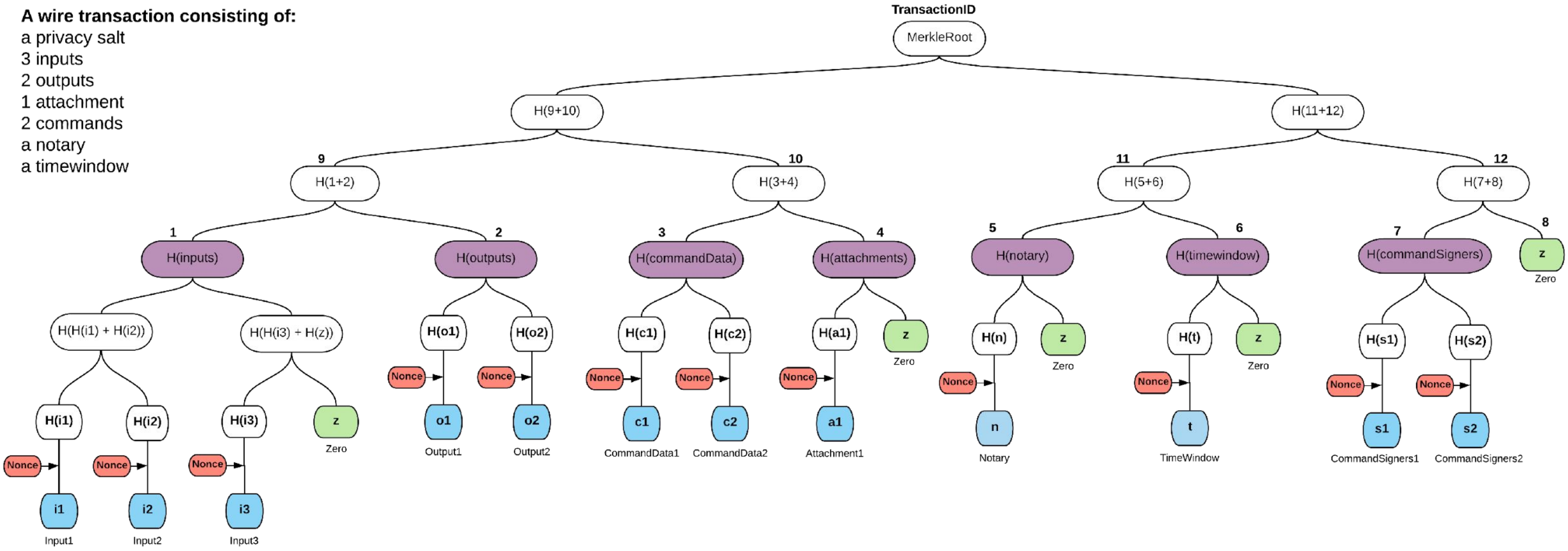
| Id | Fact |
|---|---|
| 1 | "Much consensus" |
| 6 | "Very fact" |
| 7 | "So bilateral" |
| 5 | "amaze network" |

# Checking transaction history

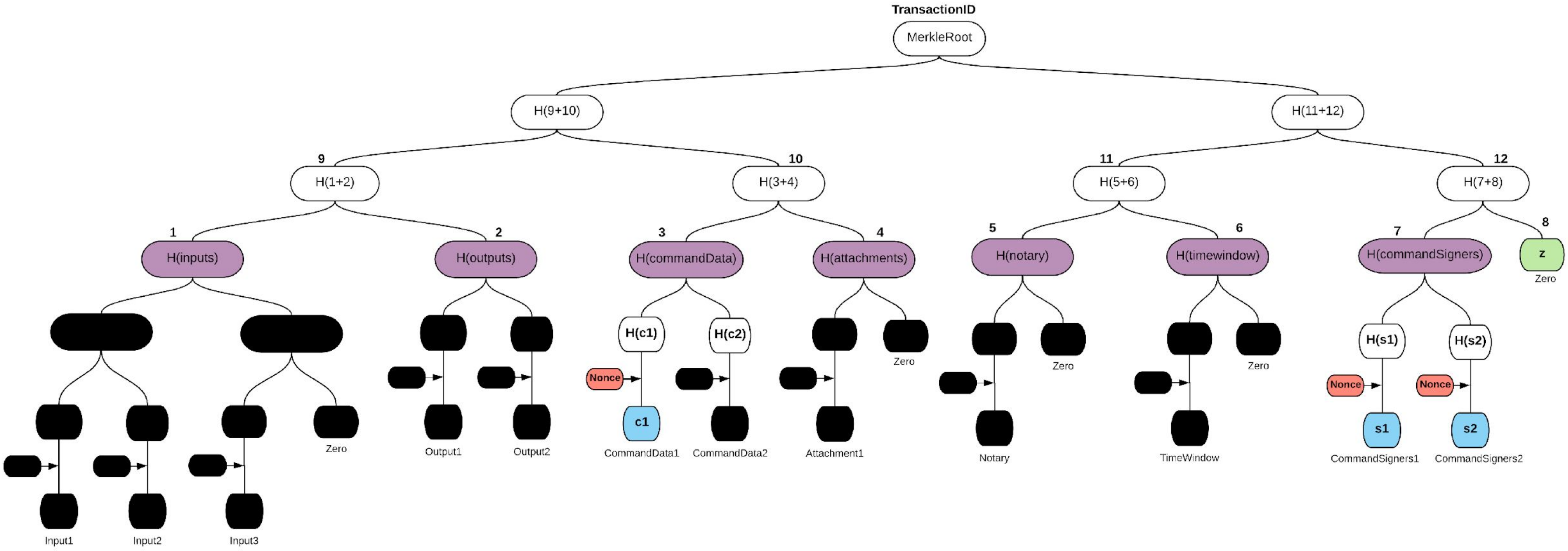


# Corda transaction

**A wire transaction consisting of:**
a privacy salt
3 inputs
2 outputs
1 attachment
2 commands
a notary
a timewindow



# Filtered Corda transaction



# Toolchain and reasoning

Corda is written in Kotlin (smart contracts as well)

Big goal is to make circuit generation as seamless as possible for smart contract developer

How bankers choose ZKP?

ZKP of choice - Zinc by Matter Labs
- smart contracts in Rust-like language
- compiled to simplistic assembly to be executed on ZincVM to generate R1CS

2 avenues for compilation:
- partial codegen from Kotlin  + Rust business logic
- full Kotlin compilation

# Results

A lot of fun

2 patents

Everything is open-sourced (in progress):
- ZKFlow (the solution itself)
- ZKKrypto (zk-friendly primitives library in Kotlin)
- Kotlin compiler

# Thank you!

ING