# Remote Side-Channel Attacks on Anonymous Transactions

## In Zcash & Monero

**Florian Tramèr** and Dan Boneh and Kenny Paterson

USENIX Security Symposium

Stanford University

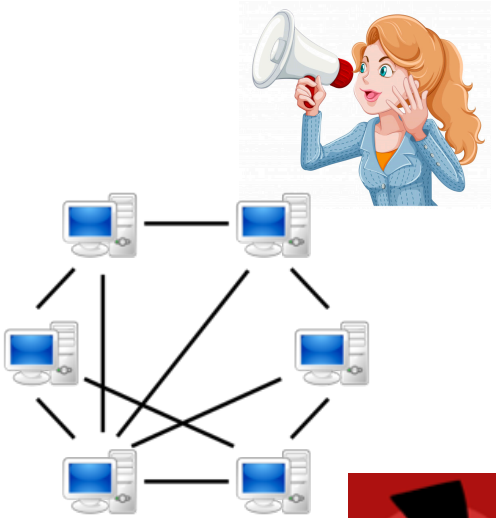
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

# Meet Alice the Anonymous Activist Blogger

# Alice's Lack of Privacy

Send $5 to $PK_A$

Signed by $SK_{Bob}$



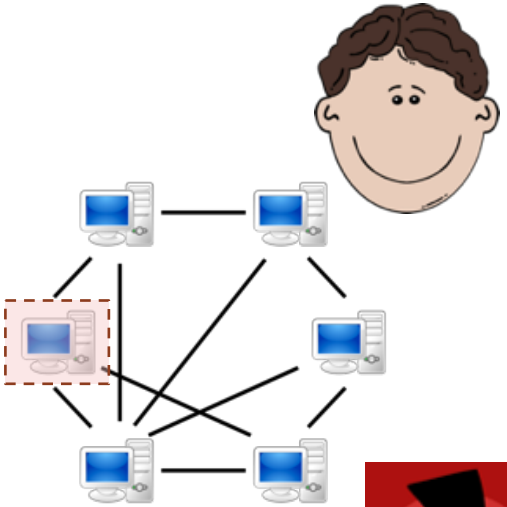


The activist just received $5 from Bob

# Alice's Lack of Privacy

Send $5 to $PK_{Bob}$

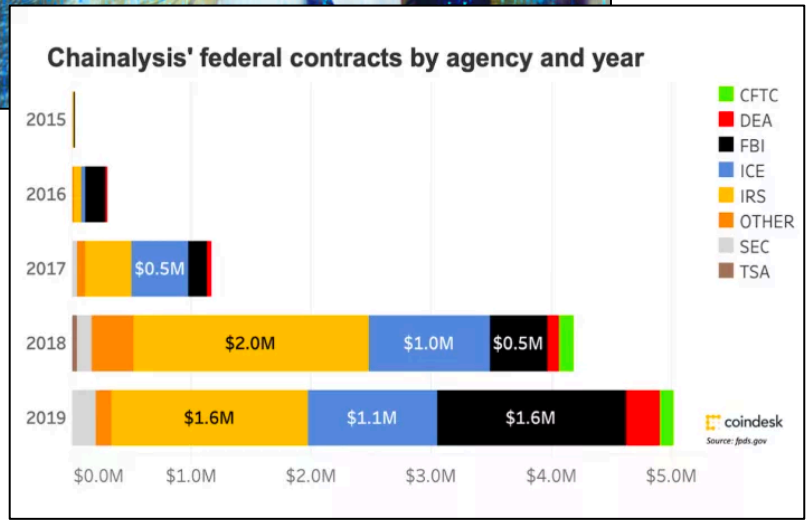
Signed by $SK_A$



This P2P node belongs to the activist!

# Alice's Lack of Privacy





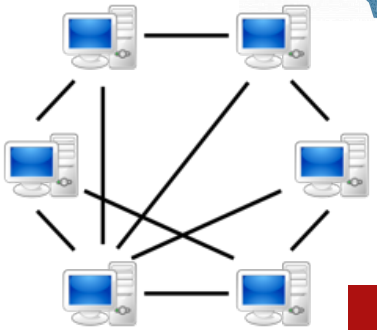
# The Solution: Anonymous Transactions

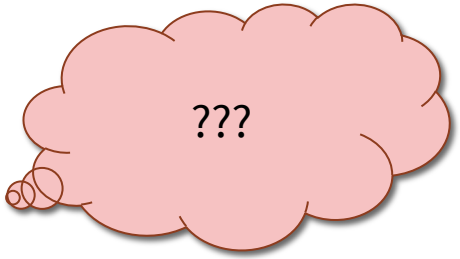
Zcash, Monero and others

Send **Enc**($5) to **Enc**($PK_A$)

Signed by **Enc**($SK_{Bob}$)

+ zk-proof π

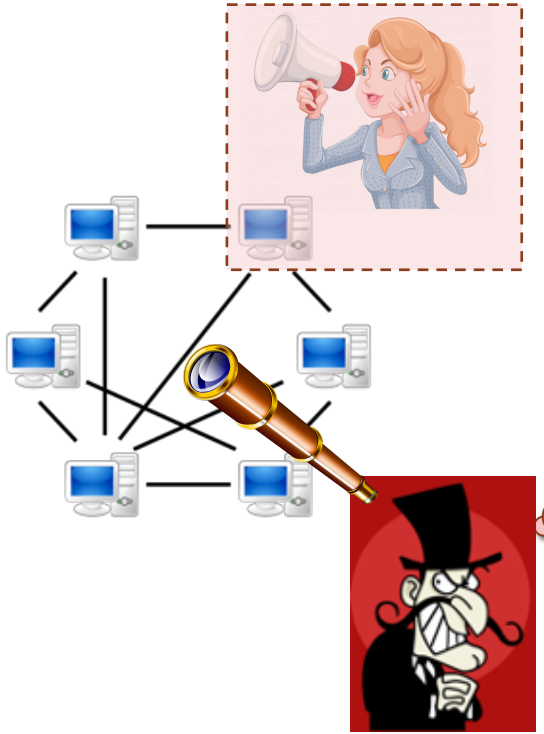





- Bob received $5 from previous txs
- These funds haven't been spent yet
- Bob knows $SK_{bob}$

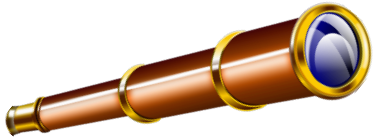
# Our Attacks: Identifying Transaction Recipients

Send Tx to Enc($PK_A$)

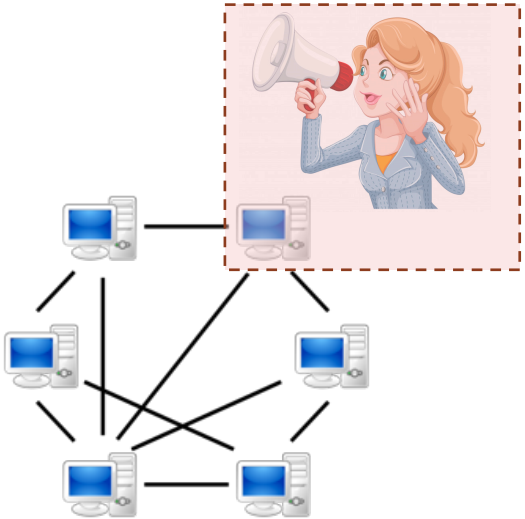


I know which node belongs to the transaction recipient

# Our Attacks: Linking an Address to a Node



Send Tx to Enc($PK_A$)



I know which P2P node belongs to the activist

# Summary of Results

**Remote side-channel attacks on various system components of anonymous transactions**

1. A general attack framework for any anonymous transaction system
2. Specific attack instantiations for Zcash and Monero
   - Determine the P2P node of *any* transaction recipient
   - Link a (diversified) public key to an IP address
3. Attacks beyond de-anonymization (for Zcash):
   - Remotely crash user nodes
   - ~ Remotely extract a user's secret viewing key
   - ~ Learn transaction amounts by timing a zk-proof generation

# Summary of Results

**Remote side-channel attacks on various system components of anonymous transactions**

**We have disclosed these vulnerabilities to Zcash and Monero and they have all been fixed!**

The general issues we found, and the lessons we learned, extend to other anonymous payment systems

$\Rightarrow$ **Getting the cryptography right is not enough!**

# Summary of Results

**Remote side-channel attacks on various system components of anonymous transactions**

1. A general attack framework for any anonymous transaction system
2. Specific attack instantiations for Zcash and Monero
   - **Determine the P2P node of *any* transaction recipient**
   - Link a (diversified) public key to an IP address
3. Attacks beyond de-anonymization (for Zcash):
   - Remotely crash user nodes
   - ~ Remotely extract a user's secret viewing key
   - ~ **Learn transaction amounts by timing a zk-proof generation**

# De-anonymizing Zcash Transactions

# Receiving Transactions in Zcash

Commitment to a "coin"

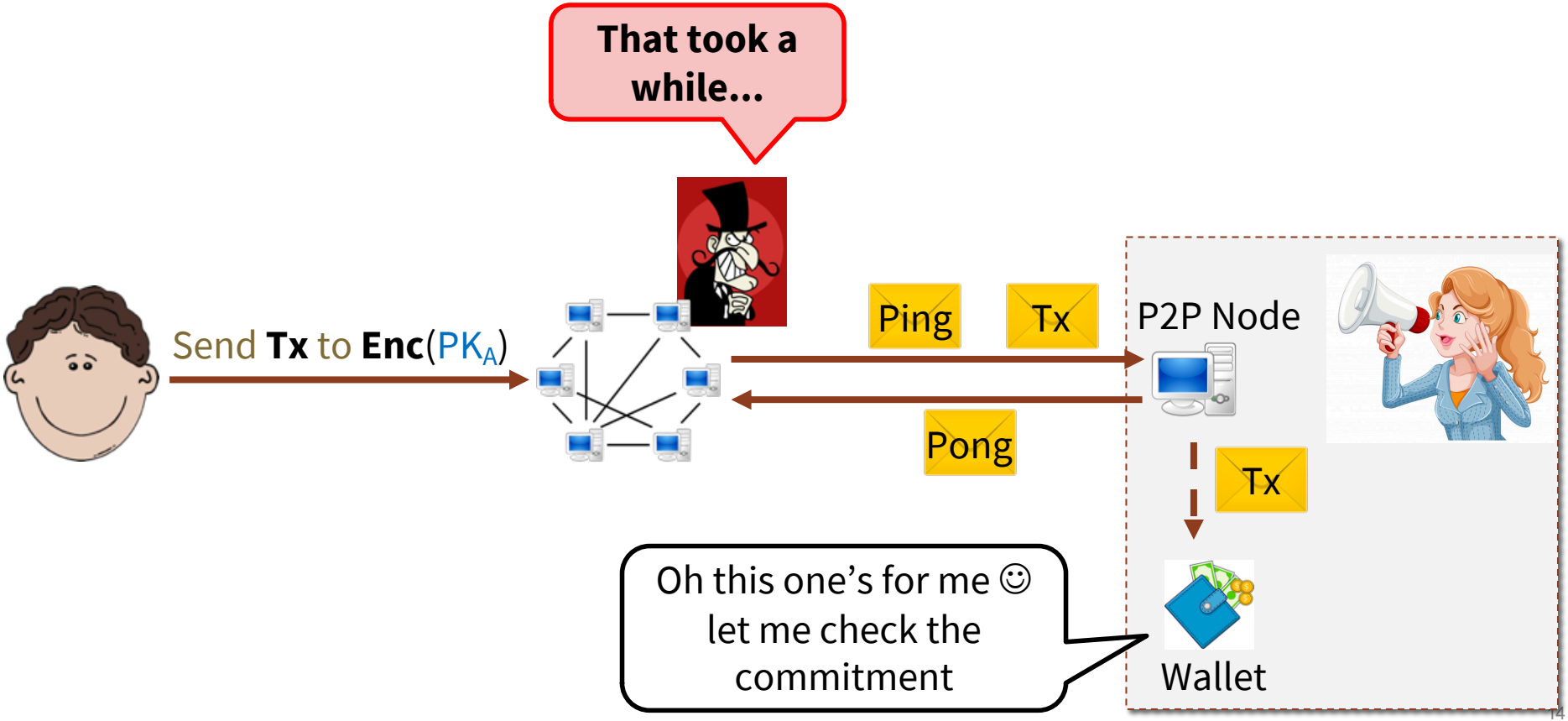
Commitment opening encrypted under the recipient's public key

```
OnReceive(Tx={Comm,C,...}):
    1) Note = Decrypt(SK_A, C)
    2) if Note = ⊥, return
    3) ($v, r) = Note
    4) Check that Comm = Commit(PK_A, $v; r)
```

(public key crypto)

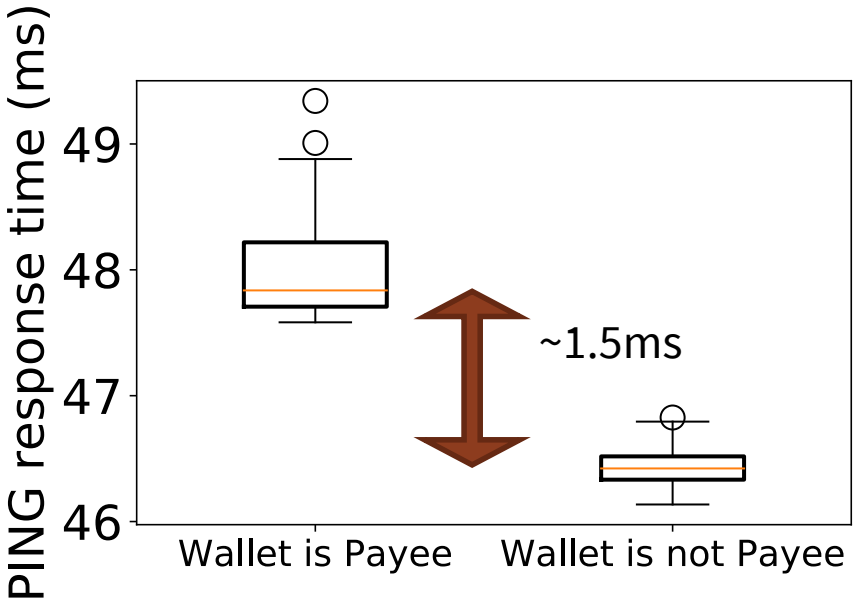
This check ensures that the coin is spendable

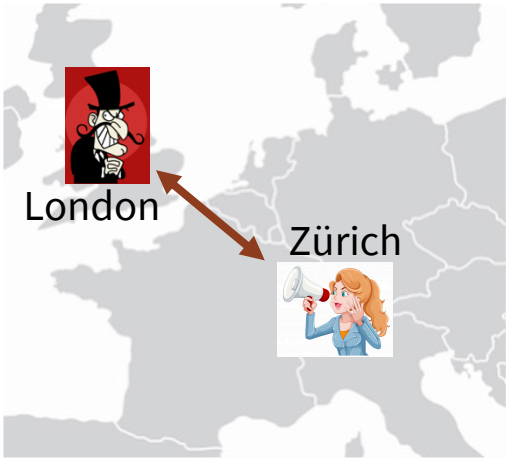
# The PING Attack



# The PING Attack

Adversary can use timing side-channel to infer receiver of **any** Tx



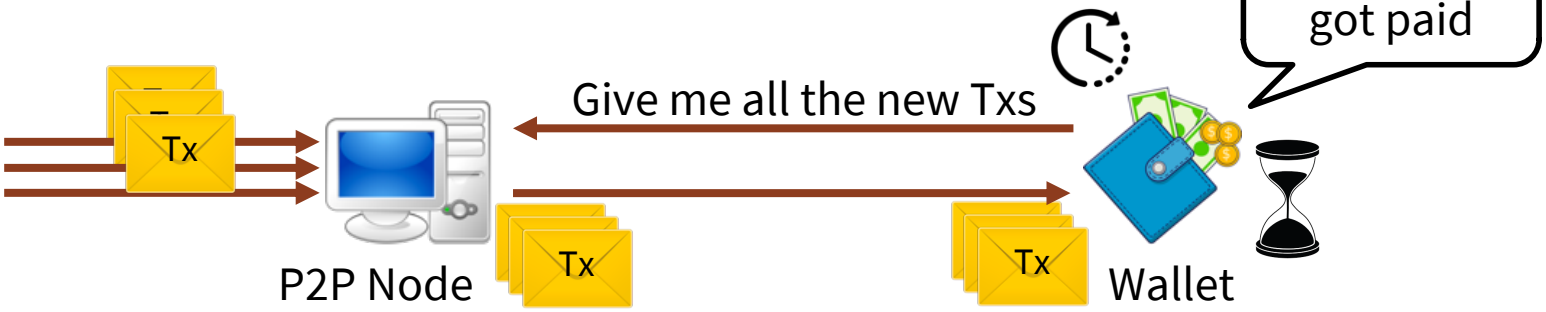
# What Went Wrong?

P2P node and wallet are tightly decoupled

⇒ Node & wallet are in completely different layers of the protocol stack

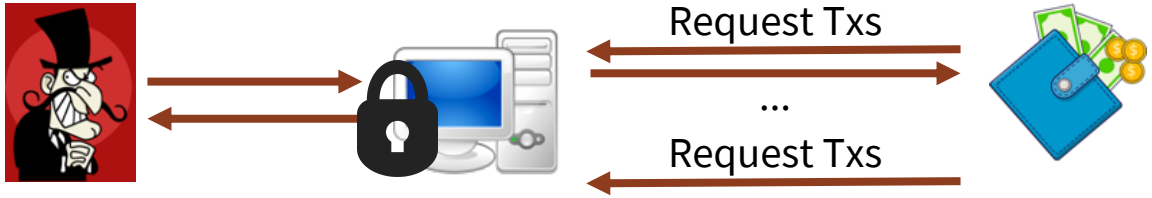
⇒ The P2P node should just act as a DB for the wallet

**Improved Design:** (and now also in !)



So why was Monero also vulnerable?

# Exploiting Leaks at Synchronization Points




✗ *Timing of wallet's requests leaks wallet's processing time*

Time between requests = 60s + time to process txs →

```
while True:
  txs = request_txs()
  process(txs)
  sleep(60)
```

✗ *Monero P2P node acquires **global mutex** to process a request*

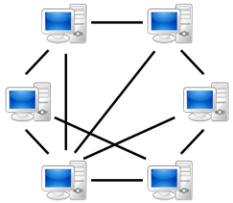
Fixed!

# Timing side channels in zkSNARK proof generation

Send Enc($5) to Enc($PK_{Bob}$)

Signed by Enc($SK_A$)

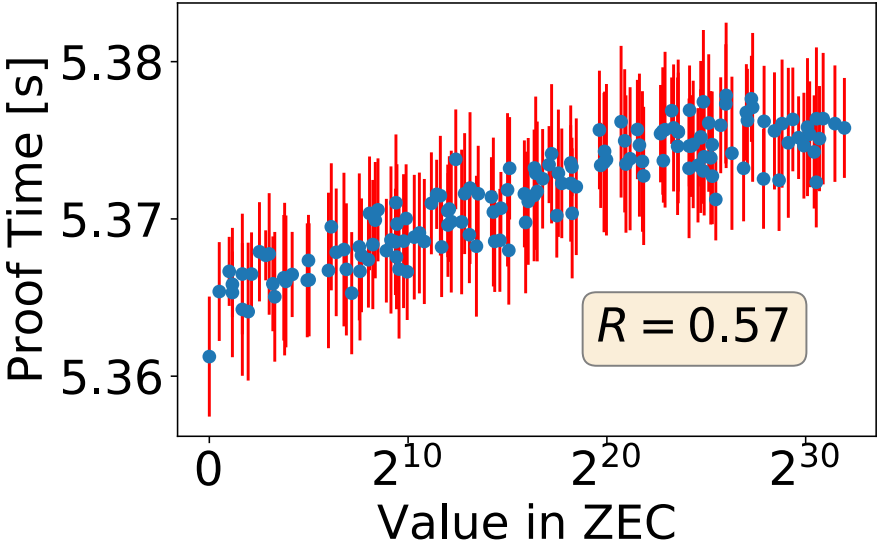
+ zk-proof π



Cryptographic proof that the transaction is valid

**Zero-knowledge**: proof leaks nothing about $PK_{Bob}$, $SK_A$, $5, ..., right?

# Timing side channels in zkSNARK proof generation




Transaction generation time leaks (some) information about value!

# Conclusions and Lessons Learned

**Anonymity is hard!**

- Flaws are not (only) in the complicated cryptography
- Be careful when inheriting designs from non-anonymous currencies (e.g., Bitcoin → Zcash)
- Develop constant-time crypto implementations

**Anonymity = good crypto + good systems design**

https://crypto.stanford.edu/timings

tramer@cs.stanford.edu