Security Risks in Zero Knowledge Proof Cryptocurrencies

Zhiniang Peng of 360 Core Security

360 INTERNET SECURITY CENTER

@edwardzpeng

Who am I

Zhiniang Peng

Ph.D. in cryptography

Security researcher @Qihoo 360

Twitter: @edwardzpeng

Research areas:

- Software security
- Applied cryptography
- Threat hunting

Outlines

Introduction

Security Risks in ZKP Cryptocurrencies

- Implementation vulnerability
- Trust risk
- Info leak in Tx
- Crypto fail
- Others

ZKP for hackers

- Key theft
- Selling hacked database
- Selling 0day

Conclusion

The Privacy of Bitcoin

Bitcoin: decentralized digital currency

Public verifiable, No anonymity

Privacy issues:

- Personal cash flow
- Account balance

Money becomes unequal：

- Black money
- Souvenir Coin

Privacy VS Public Verifiability

From	Alice
To	Bob
Amount	1

From	Scrooge
To	Donald
Amount	2

…	…
…	…
…	…

From	Bob
To	Eve
Amount	1

Privacy → Encryption?

From	Enc(A)
To	Enc(B)
Amount	Enc(1)

From	Enc(S)
To	Enc(D)
Amount	Enc(2)

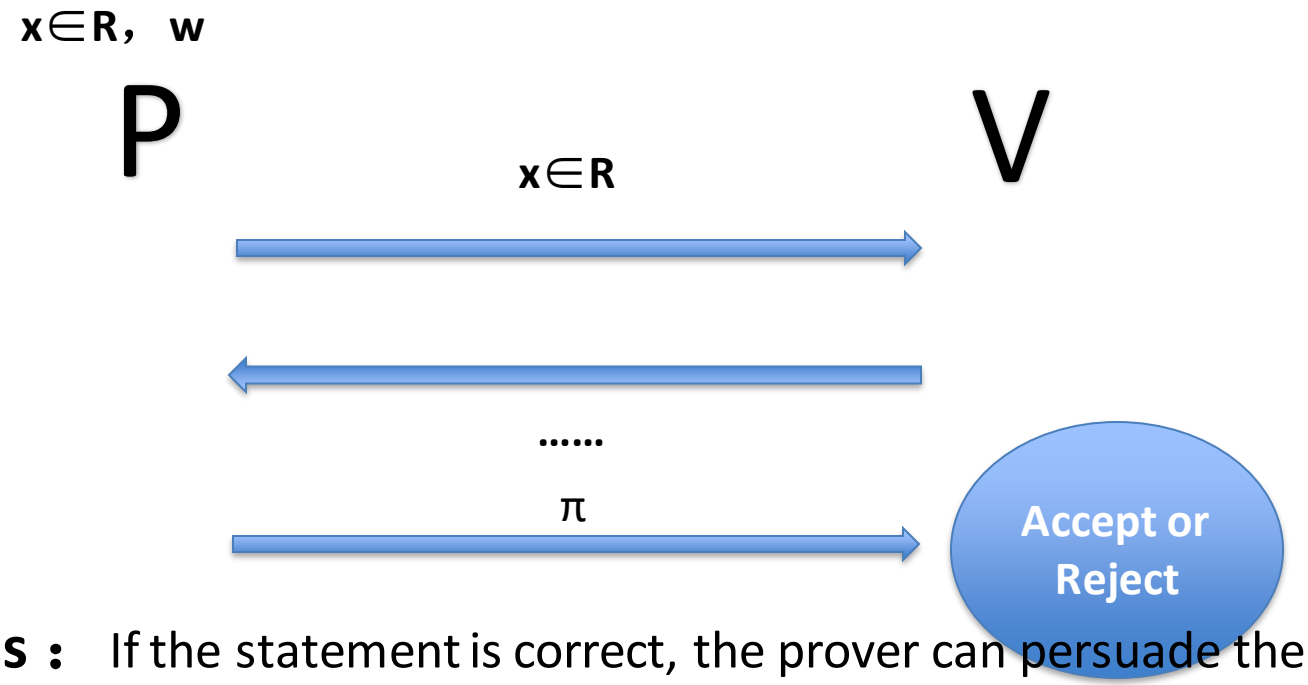
…	…
…	…
…	…

From	Enc(B)
To	Enc(E)
Amount	Enc(1)

Encryption Conflict with Public verifiability

@edwardzpeng

ZKP



Completeness： If the statement is correct, the prover can persuade the verifier

Soundness： If the statement is wrong, the prover cannot persuade the verifier

Zero knowledge： The verifier is unable to obtain any other information except that the statement is correct

Non interactive zero knowledge proof (NIZK)

If factorization is difficult, for any NP language there is a NIZK

ZKP with Bitcoin

From	Enc(A)
To	Enc(B)
Amount	Enc(1)
Proof	π

From	Enc(S)
To	Enc(D)
Amount	Enc(2)
Proof	π'

From	Enc(B)
To	Enc(E)
Amount	Enc(1)
Proof	π''

From	c_1
To	c_2
Amount	c_3
Proof	π'''

Encryption all the transactions

Proof π:

Non interactive zero knowledge proof

Prove that encrypted transactions are legal

Encryption Conflict with Public verifiability (Solved!)

@edwardzpeng

zkSNARKs

High performance requirements for blockchain

Performance of Universal ZKP ☹

zkSNARKs

Succinct

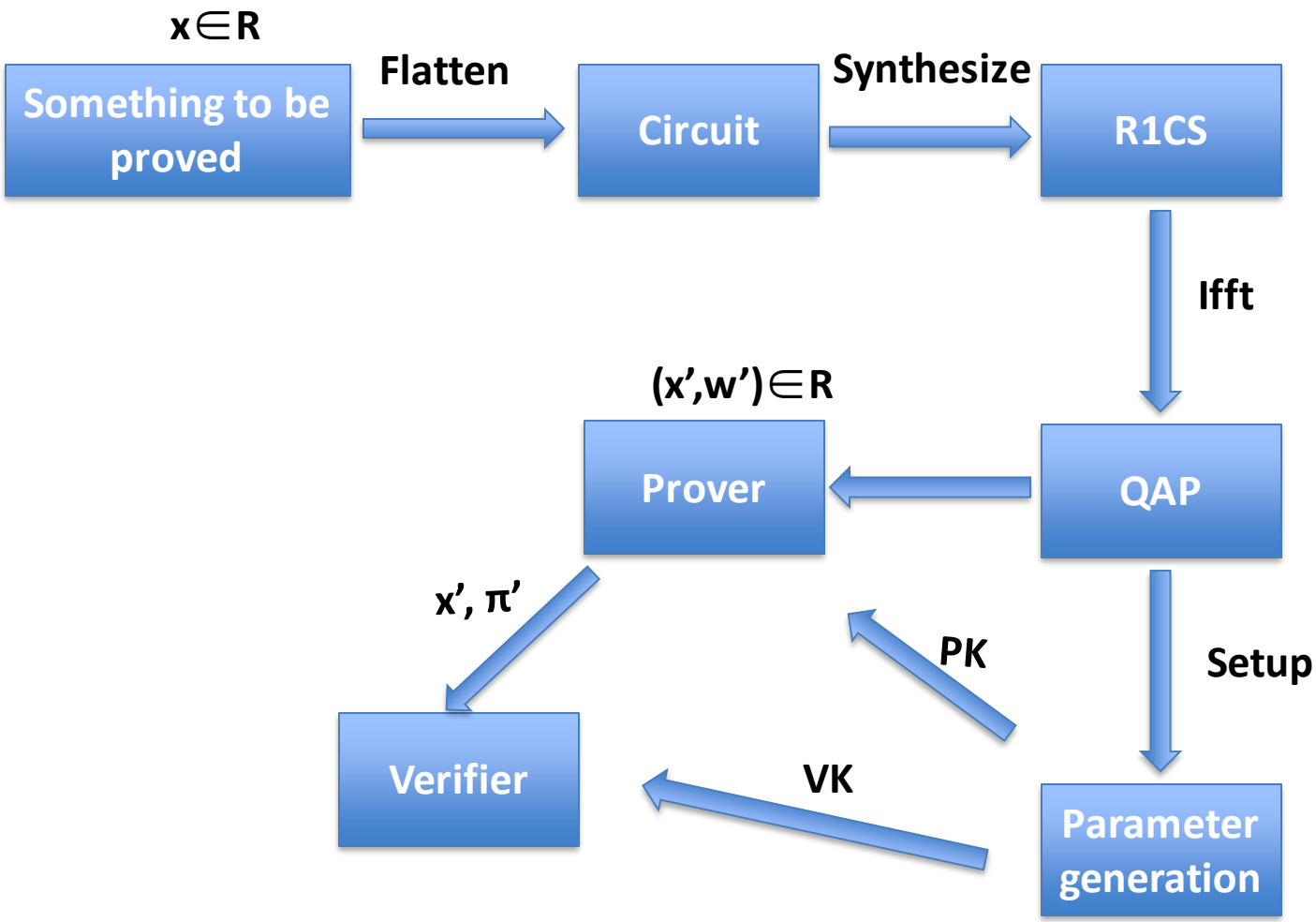
Non-interactive

Argument

of **K**nowledge

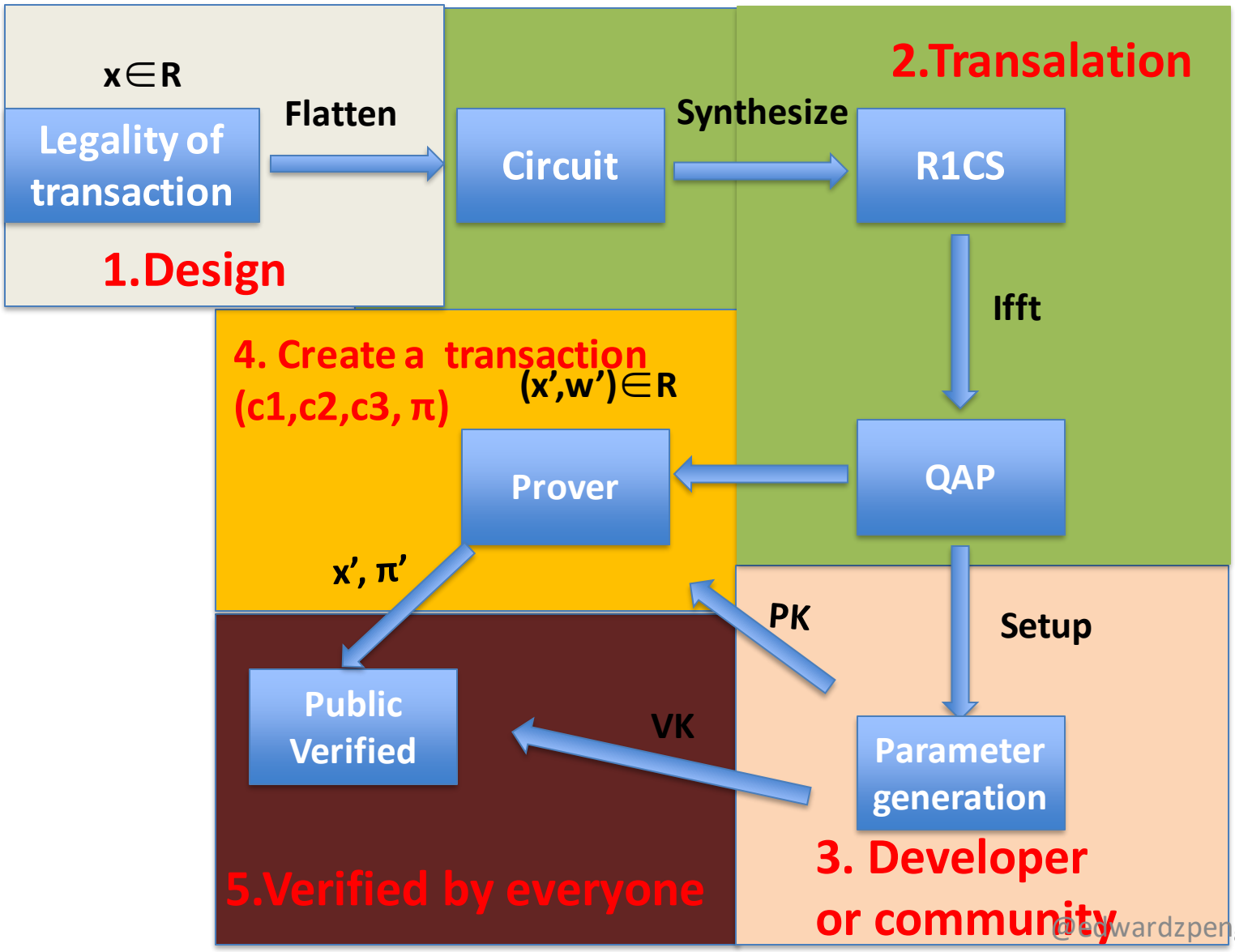
Zero-**k**nowledge

How zkSNARKs works

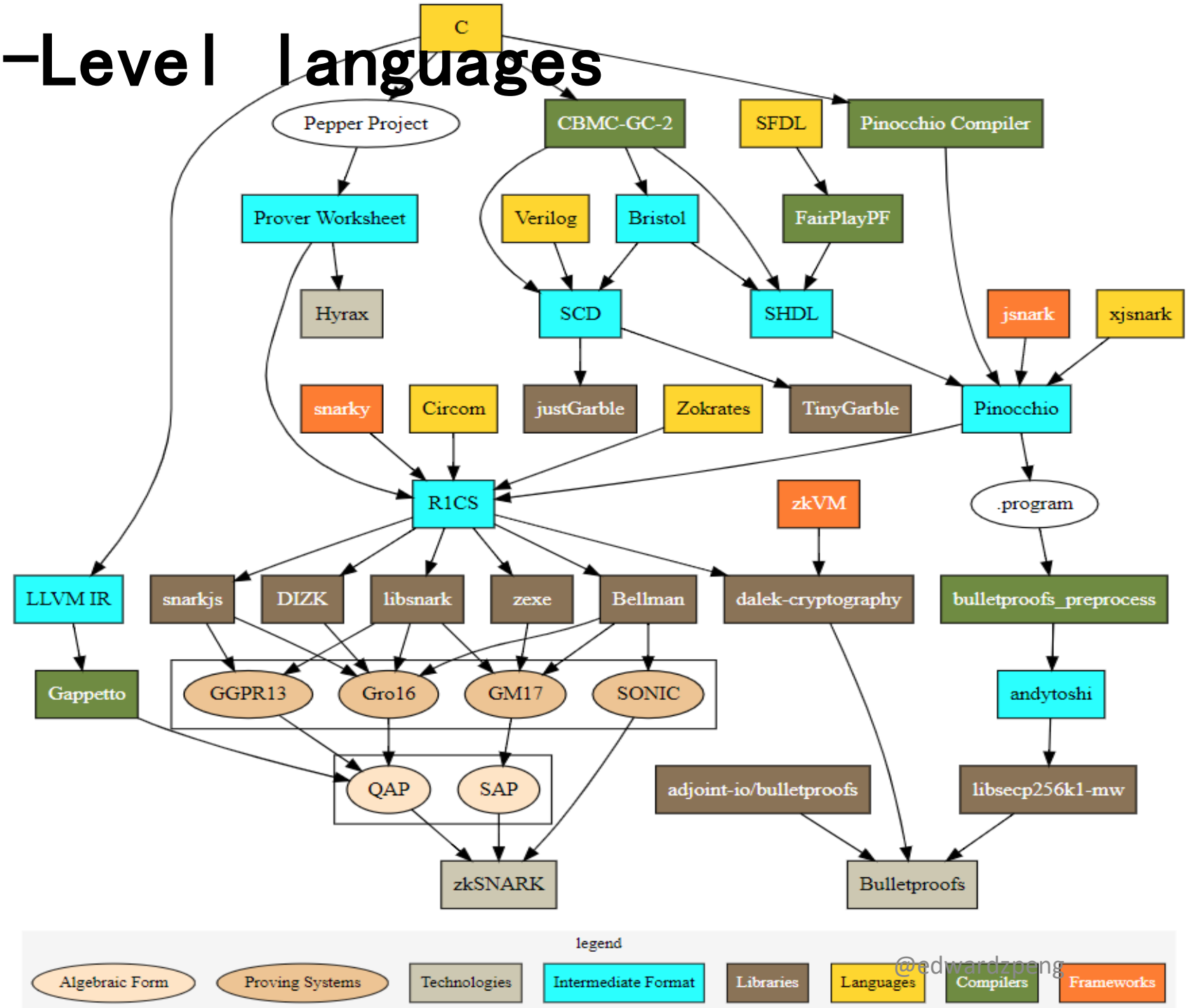


@edwardzpeng

zkSNARKs in Cryptocurrency



High-Level languages



Security Risks in ZKP Cryptocurrencies

Implementation vulnerability

Trust risk

Info leak in Tx

Crypto fail

Others

Implementation vulnerability

Category

Memory corruption ☹

- Most Implemented in : rust、 java、 go (memory safety)
- libSNARKs(C++): hard to exploit

Logic bug ☺

- Protocol Design
- Circuit Implementation
- Application logic

Crypto Implementation ☺

- New crypto -> new bugs!

Protocol Design

Complex

Privacy, performance, ease of use -> Complex

Lots of crypto

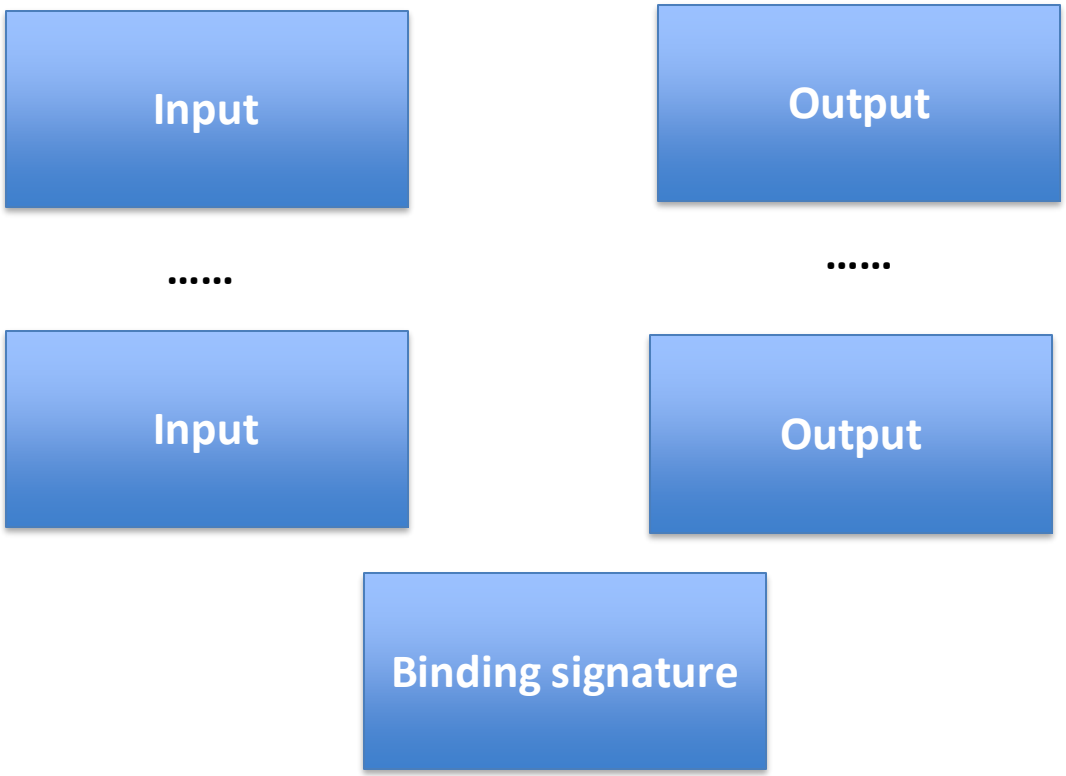
Only expert can review

Zcash for example

Zcash shield TX

Input
anchor
nullifier
rk
cv
Proof
spendAuthSig

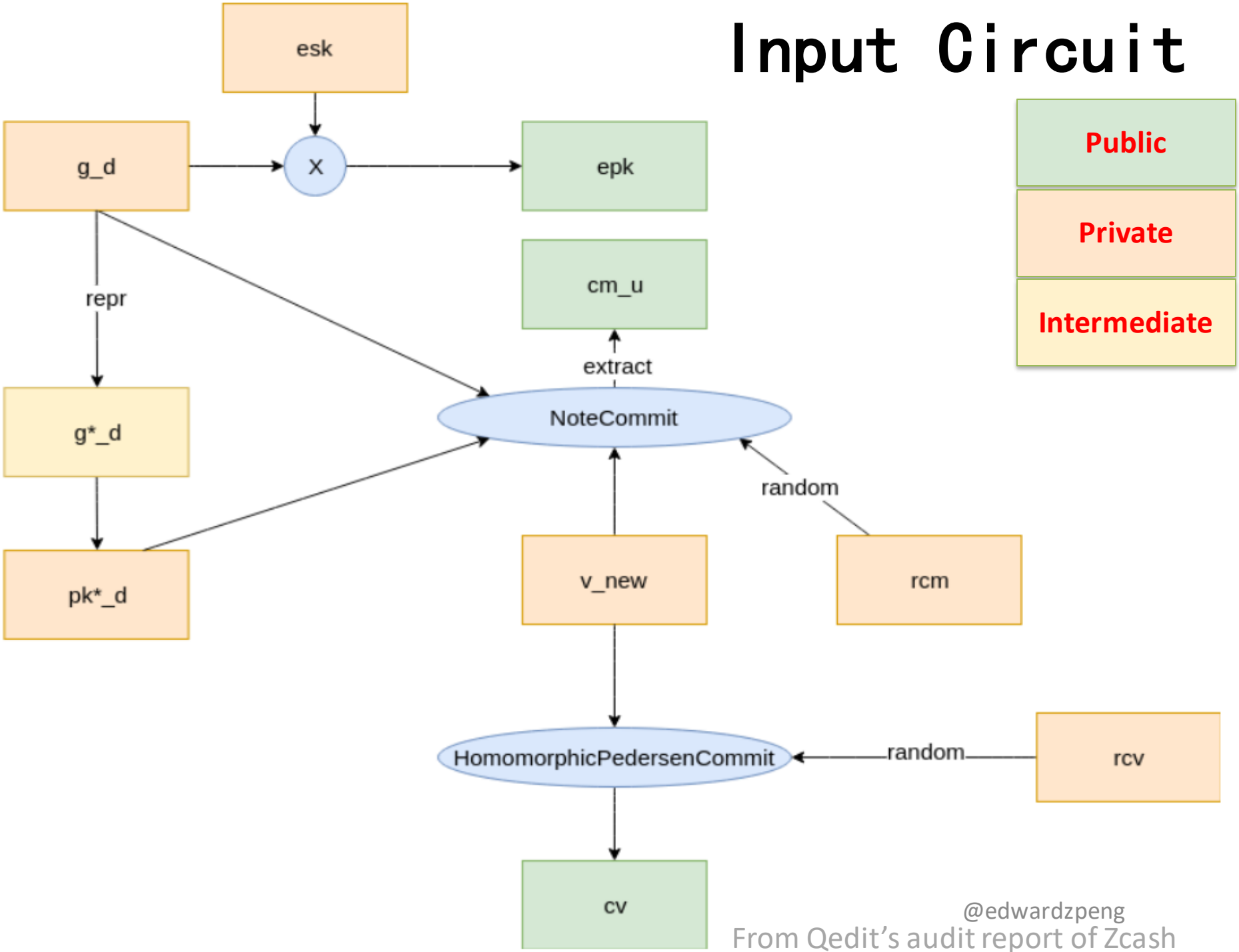
Output
cmu
epk
ciphertext1
ciphertext2
cv
Proof



Binding all the input and output

@edwardzpeng

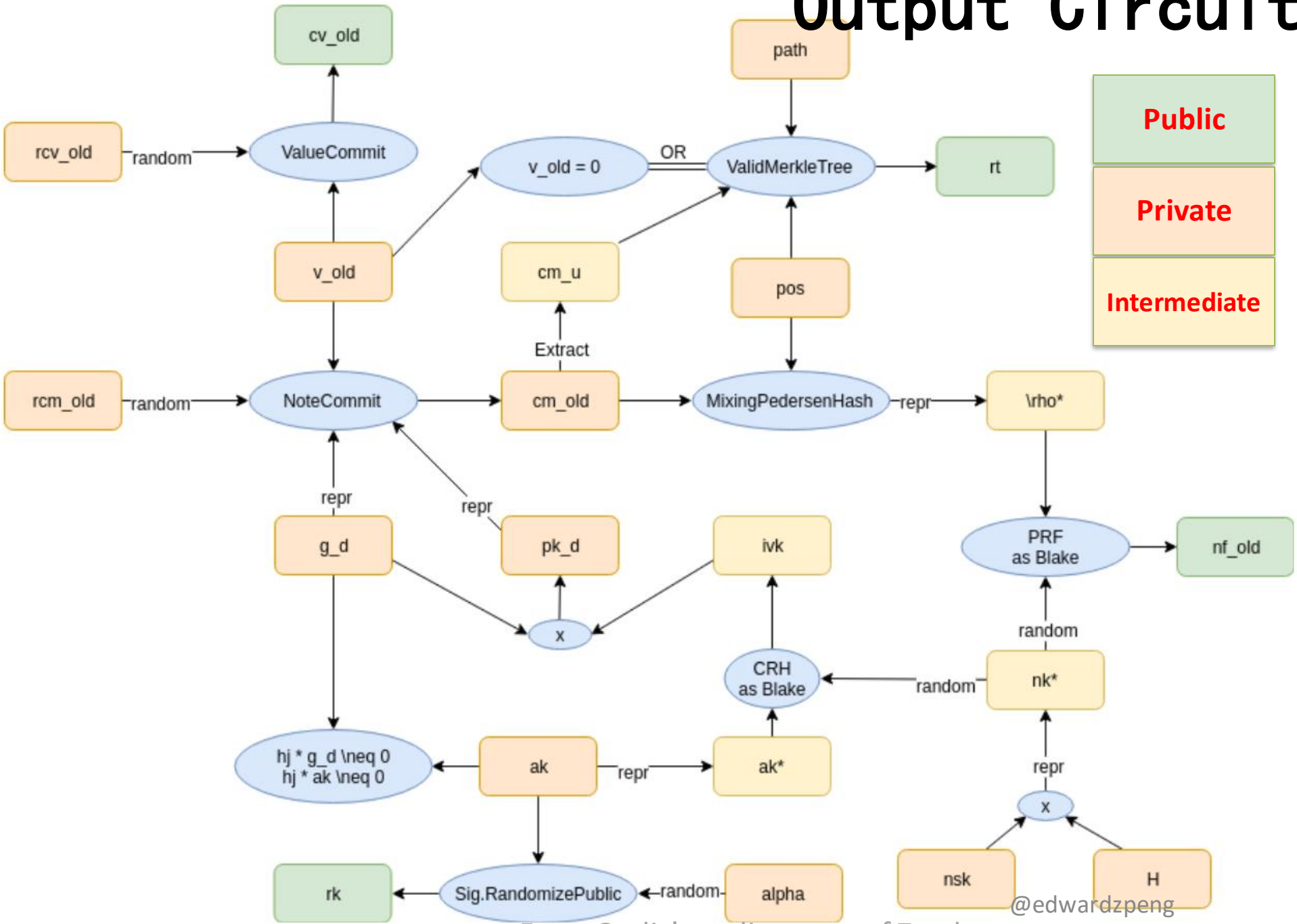
Input Circuit



@edwardzpeng

From Qedit's audit report of Zcash

Output Circuit



@edwardzpeng

From Qedit's audit report of Zcash

Protocol Design bug

Zcash Faerie Gold attack:

Attacker send two note with the same rho

Only one can be used

Fake money

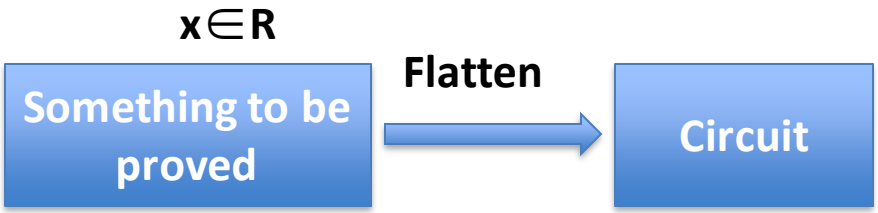
Fix: Force to be different by crypto

URL： https://github.com/zcash/zcash/issues/98

Similar vulnerability also occur in Monero

Circuit Implementation

What is Circuit?



The same logic implemented twice:

In typical programing languages and in Circuit

Inconsistence between two implementation

Result in critical vulnerability

Heavily check in all the popular projects

Bugs in Application

Application developer calls ZKP lib to build application

Due to the lack of sufficient understanding of the ZKP, their code is easy to be vulnerable.

Semaphore double spend

Vulnerability allowing double spend #16

Closed poma opened this issue on 26 Jul · 2 comments

poma commented on 26 Jul • edited

Looks like in Semaphore.sol#L83 we don't check that nullifier length is less than field modulus. So `nullifier_hash + 21888242871839275222246405745257275088548364400416034343698204186575808495617` will also pass snark proof verification if it fits into uint256, allowing double spend.

Example of 2 transactions:

https://kovan.etherscan.io/tx/0x5e8bf35ad76a086b98698f9d20bd7b6397ccc90aa6f85c1c5debc0262be5458a
https://kovan.etherscan.io/tx/0x9a47cc8daec9d0a5e9a860ada77730190124f9864a5917dcb8f41773d94cfc1a

21

To prevent double spend : a unique Nullifier for each Tx

Semaphore: If Nullifier n works, Nullifier n+p also works

Double spend if you don't check the length

@edwardzpeng

Tron double spend

Tron use Librustzcash to build their privacy solution

Directly use functions in the Lib without constrains on variables

itHub, Inc. [US] | github.com/tronprotocol/java-tron/commit/c4729f1e7131e81786f95f7d3694ff6bed87749e?diff=unified

Add constraints on some variables.

develop (#2352)

skipjack8 committed on 4 Jul — 1 parent 2653e1d — commit c4729f1e7131e81

Showing 2 changed files with 31 additions and 7 deletions.

35 src/main/java/org/tron/common/zksnark/LibrustzcashParam.java

```
@@ -49,6 +49,18 @@ public static void validVoucherPath(byte[] voucherPath) throws ZksnarkException
49  49        }
50  50      }
51  51
    52  +   public static void validValueParams(long value) throws ZksnarkException {
    53  +     if(value < 0){
    54  +       throw new ZksnarkException("Value shoud be non-negative.");
    55  +     }
    56  +   }
    57  +
    58  +   public static void validPositionParams(long value) throws ZksnarkException {
    59  +     if(value < 0){
    60  +       throw new ZksnarkException("Position shoud be non-negative.");
    61  +     }
    62  +   }
    63  +
52  64      interface ValidParam {
53  65
```

@edwardzpeng

Tron nullifier

One transaction has multiple input

Does not check the duplicate

src/main/java/org/tron/core/actuator/ShieldedTransferActuator.java

```
@@ -103,9 +103,14 @@ private void executeTransparentTo(byte[] toAddress, long amount) throws Contract
     }

     //record shielded transaction data.
-    private void executeShielded(List<SpendDescription> spends, List<ReceiveDescription> receives) {
+    private void executeShielded(List<SpendDescription> spends, List<ReceiveDescription> receives)
+        throws ContractExeException {
       //handle spends
       for (SpendDescription spend : spends) {
+        if (dbManager.getNullfierStore().has(
+            new BytesCapsule(spend.getNullifier().toByteArray()).getData())) {
+          throw new ContractExeException("double spend");
+        }
         dbManager.getNullfierStore().put(new BytesCapsule(spend.getNullifier().toByteArray()));
       }
```

@edwardzpeng

Crypto Implementation

New crypto -> new bugs!

LibSNARKs:

- R1CS-to-QAP reduction bug
  - Linear dependent -> soundness error
  - Fix: Increase redundancy
- https://eprint.iacr.org/2015/437.pdf

Zcash side channel attack

- Reject attack, Ping attack
- Node processing a transaction related with himself:
  - Side channel in decryption time -> Info leak
- Break the anonymity
- https://crypto.stanford.edu/timings/pingreject.pdf

Trust Risk

Trust Risk

Basic ideas in zkSNARKs：

Generate the challenge (x) ahead of time

keep encrypted (x), discard plaintext (x)

Verifying the proof (A,B,C):

$$A \cdot B = \alpha \cdot \beta + \frac{\sum_{i=0}^{\ell} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x))}{\gamma} \cdot \gamma + C \cdot \delta$$

Undetectable backdoor (x)

You can create proof for any statement if you know x

Create fake proof -> create fake money ☺

Zero knowledge proof -> undetectable!

To solve Trust risk：

Generate encrypted (x) by secure multiparty computation (MPC) ----> nobody know the plaintext of (x)

@edwardzpeng

Zcash MPC phase 1

ZcashFoundation / powersoftau-attestations

Watch 7 | Star 16 | Fork 17

Code | Issues 0 | Pull requests 0 | Actions | Projects 0 | Wiki | Security | Insights

Branch: master | powersoftau-attestations / 0001 / | Create new file | Upload files | Find file | History

pera Updating mailing list URLs — Latest commit 79ee8b1 on 3 Jan 2018

..		
README.md	Updating mailing list URLs	2 years ago
report.asc	Andrew Miller's attestation.	2 years ago

README.md

Andrew Miller

- Assistant Professor at the University of Illinois, Urbana-Champaign
- Chair of the Zcash Foundation Board
- https://soc1024.com/
- Mailing list post: https://lists.z.cash.foundation/pipermail/zapps-wg/2017/000007.html
- See `./report.asc` for the signed attestation.

Response file:

- https://powersoftau-transcript.s3-us-west-2.amazonaws.com/15729e0edc4201dc5ee6241437d926f614cb4214ff1b9c6fbd73daf401639f7a4238cf04bc94edac9f2ad037003daab9a4408ba7c62a4413dc2a0ddd683bd719
- https://s3.amazonaws.com/socrates1024_a/response-2017-11-10-amiller

Generating encrypted (x)

Public verifiable

More than 100 participants

Secure as long as at least one person is honest

@edwardzpeng

Zcash MPC phase 2

github.com/zcash-hackworks/sapling-mpc/wiki

Participants:

- Sean Bowe
  - 4caed5dfc4fd6959b5181c4adc31fe013c58b438c360d30af5a40fe02cc19a9ad56344679524e07fbad6166cc44ee35e1f6e70e5267b46ce93d059cad3ab7ed5
- Ian Munoz
  - dd8ca189564354d5c9405b16b32f8dd0ed4644e4184e9fbc9feaccebd732ad8276033318b0b70fa5b1f59f49ce3ed42f3f575b9e6219c11d3e28a82ed24ca990
- Jason Davies
  - 851b81478d1ce92c1f7116c8f355c999447e36a151b97f36ff8d5cbf9b98a5b00431b2a087da2038d201669dd3a8eba8bbd2d92ea8e348b92e1b51b09fe5bb5f
- Larry Ruane
  - 82641343d6840bcb218a7cef8f46d6f6678fff3692e8ec6b9bfc44245737d0c047dd441fd3608ea1463302a8c9521dcd716df3b114741bcdf4da5add68bf35d3
- Gabor Losonci
  - be9fe35787982dda14b81a796dbab4a5ecfe1bf2d56c6b0186e4552b5a942d61a5378c4c903aa3087226b7bdf0fcf7808eaa6a2b1a1ae0d42793bb9836718a23
- Michael Perklin
  - ec782c5b9b33d98750012e3d1d62f34b490acd2d392a30629e98596198143e414536f30bd9414ee01dc26fc7d132ca245e400c738073a68cad9e439a5acb83a6
- Alexey Ermishkin
  - ce07ea6e537f106c0b9b70a5aff9dcc8105dfc1f5f69e32d272f7749e2b41ba728e3ef860d57ab933bdfb458c675bd5cd46c08429a1229dc8b21dc8e95f5815b

Generate parameter for specific circuit

Some project lack this phase:

-> backdoor again

@edwardzpeng

Personal experience in MPC

Lots of project are doing MPC

Ethereum, Tron

Some personal experience

Zcash: no reply ☹

Tron: Waiting for 3 month now ☹

Ethereum: https://github.com/weijiekoh/perpetualpowersoftau/blob/master/0011_zhiniang_response/zhiniang_attestation.txt

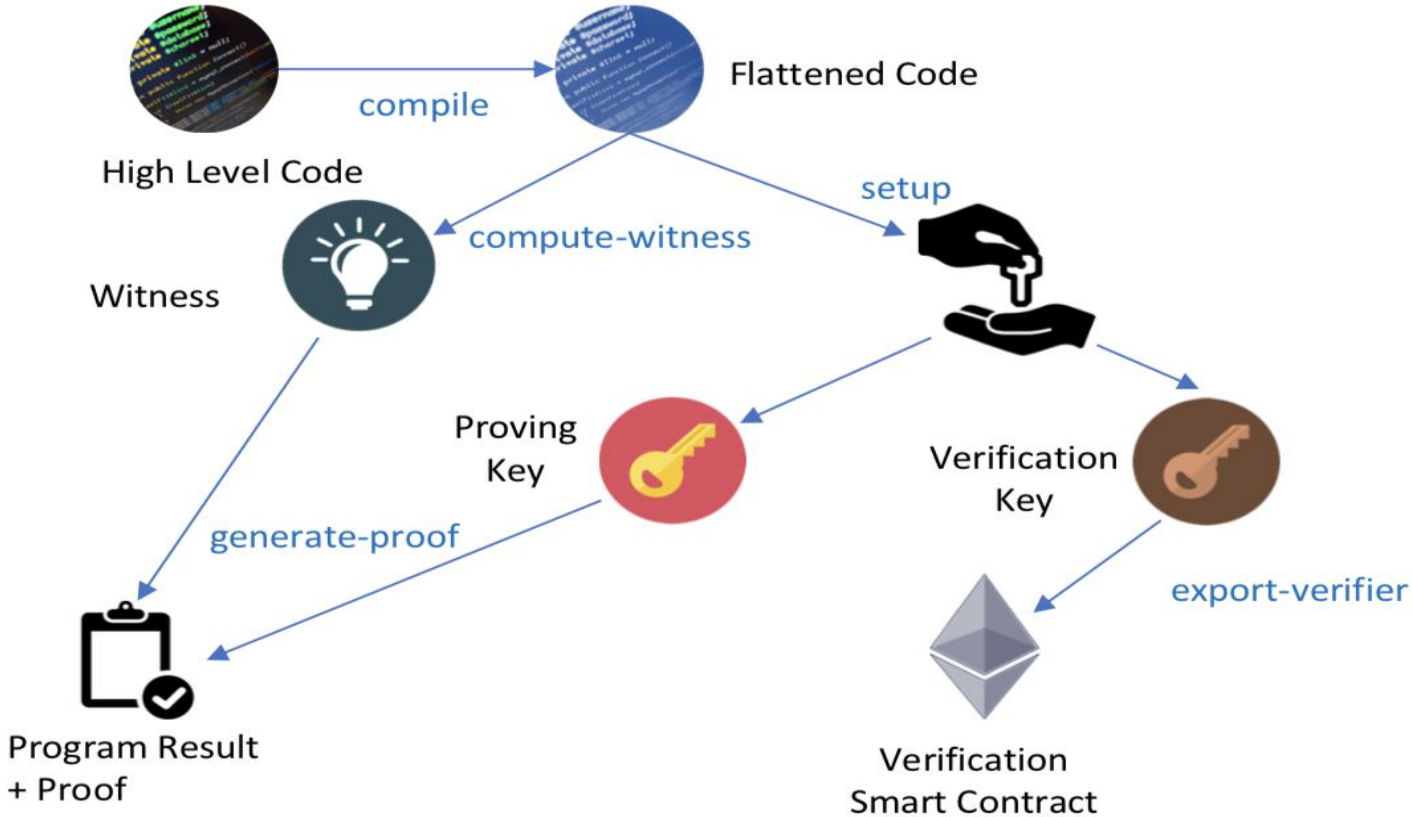
MPC process is completely controlled by organizer

Black Box style ☹

If you're not involved, you can't be 100% sure

@edwardzpeng

demo: ZoKrates



ZoKrates: a toolbox for zkSNARKs on Ethereum

Compile your high level code into verification smart contract

Black box setup: No MPC

Backdoor ☹

@edwardzpeng

Other ZKP systems

	libSNARK [14]	Ligero [6]	Bulletproofs [17]	Hyrax [50]	libSTARK [9]	Aurora [12]	Libra
$\mathcal{G}$	$O(C)$ per-statement trusted setup	no trusted setup					$O(n)$ one-time trusted setup
$\mathcal{P}$	$O(C \log C)$	$O(C \log C)$	$O(C)$	$O(C \log C)$	$O(C \log^2 C)$	$O(C \log C)$	$O(C)$
$\mathcal{V}$	$O(1)$	$O(C)$	$O(C)$	$O(\sqrt{n} + d \log C)$	$O(\log^2 C)$	$O(C)$	$O(d \log C)$
$\lvert\pi\rvert$	$O(1)$	$O(\sqrt{C})$	$O(\log C)$	$O(\sqrt{n} + d \log C)$	$O(\log^2 C)$	$O(\log^2 C)$	$O(d \log C)$
$\mathcal{G}$	1027s	NA					210s
$\mathcal{P}$	360s	400s	13,000s	1,041s	2,022s	3199s	201s
$\mathcal{V}$	0.002s	4s	900s	9.9s	0.044s	15.2s	0.71s
$\lvert\pi\rvert$	0.13KB	1,500KB	5.5KB	185KB	395KB	174.3KB	51KB

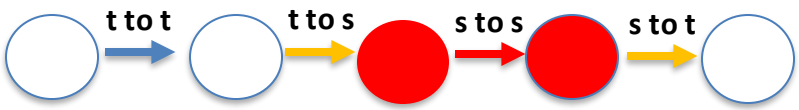
Lots of new schemes have emerged recently,

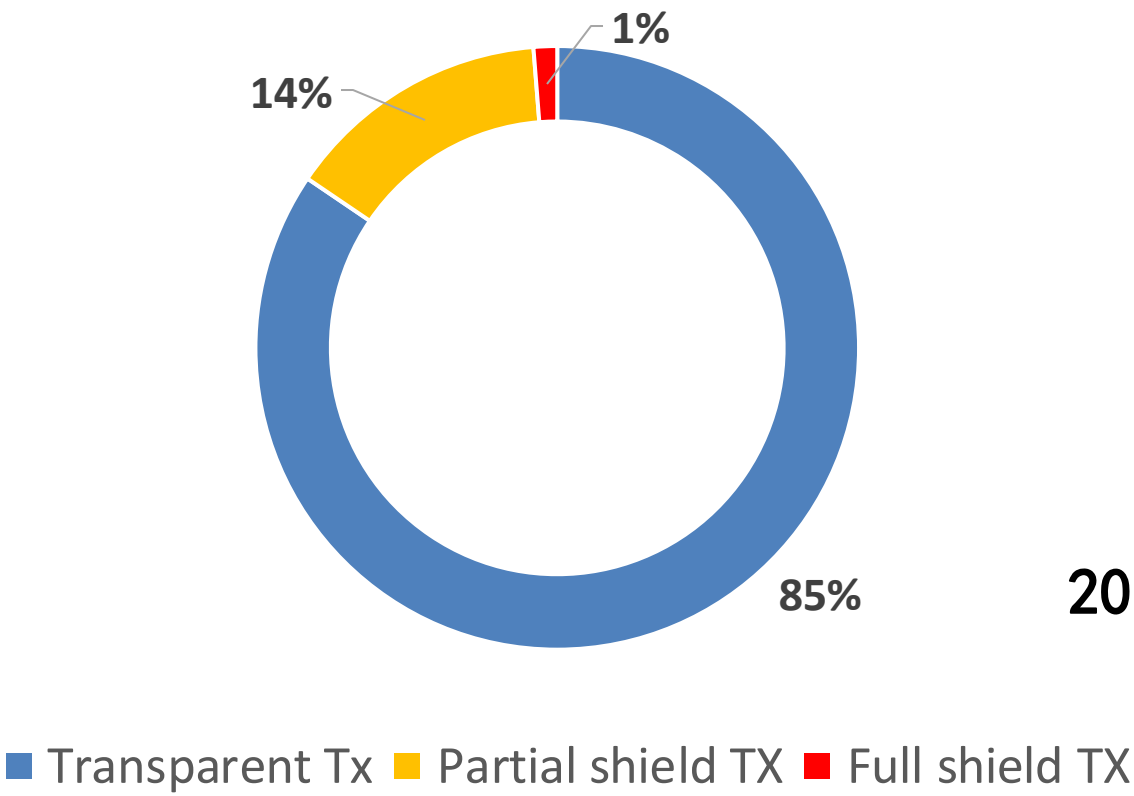
Trade-off between security, performance, trust model.

More options for the future project ☺ @edwardzpeng

Info leak in Tx

Zcash Transaction Statistics



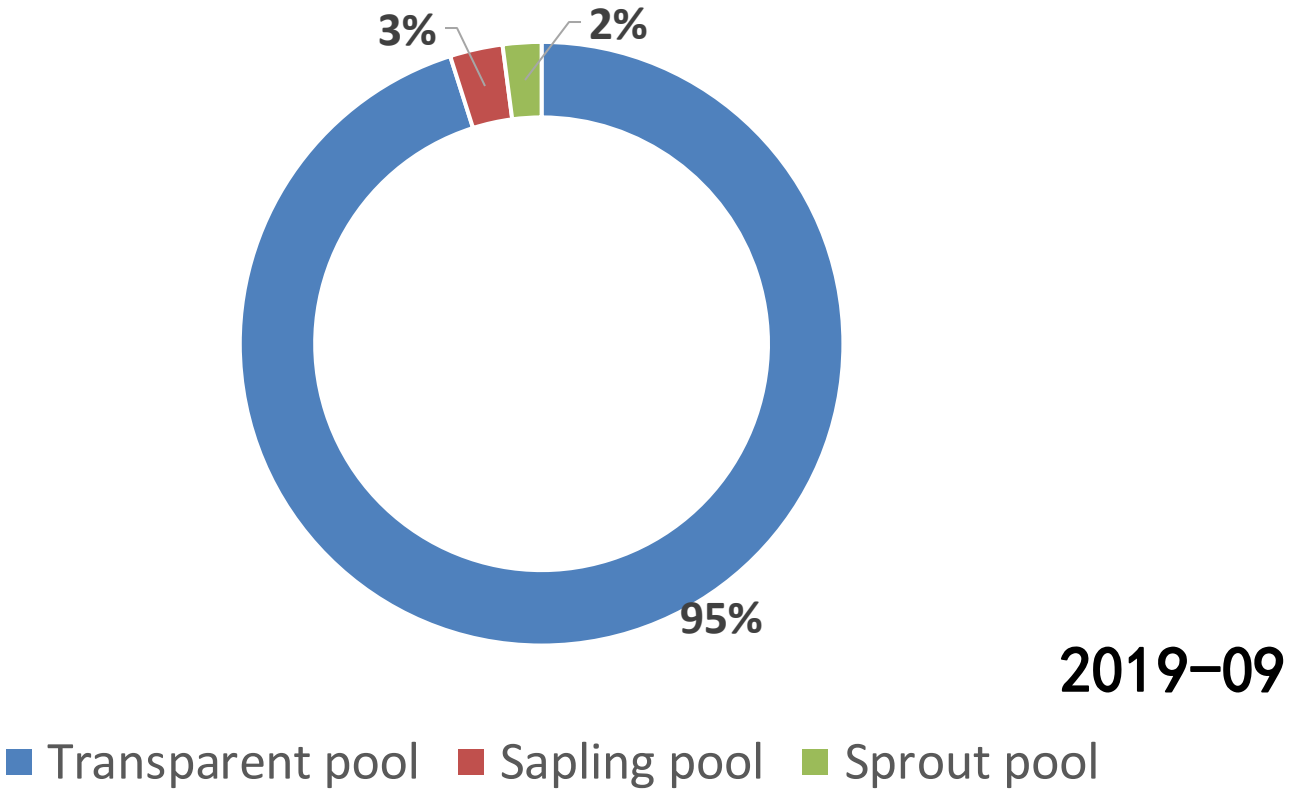


2019-09

Most transactions are traceable

@edwardzpeng

Zcash Coin Statistics



@edwardzpeng

An Empirical Analysis of Anonymity in Zcash

George Kappos, Haaroon Yousaf, Mary Maller, and Sarah Meiklejohn
University College London
{georgios.kappos.16,h.yousaf,mary.maller.15,s.meiklejohn}@ucl.ac.uk

On the linkability of Zcash transactions

Jeffrey Quesnelle
University of Michigan-Dearborn

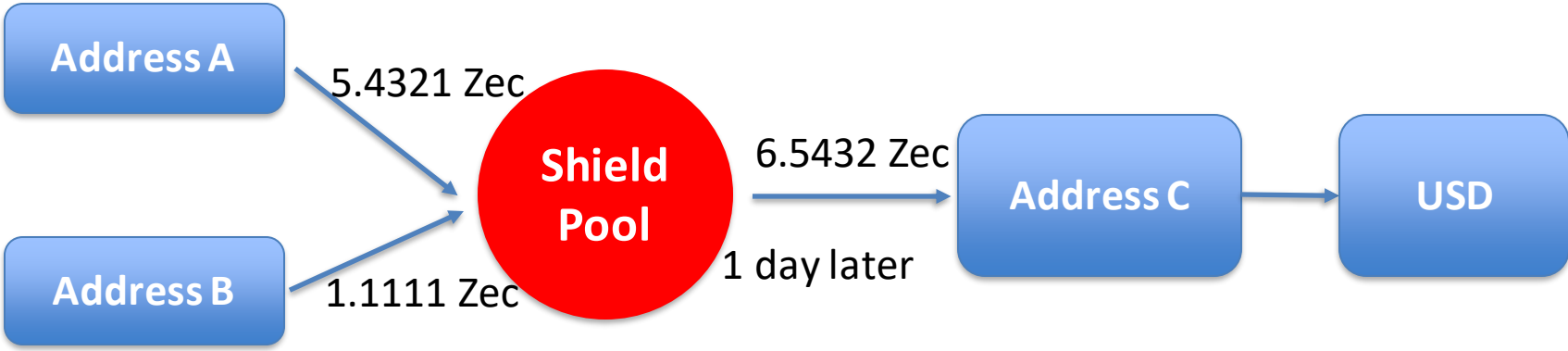
An Empirical Analysis of Traceability in the Monero Blockchain

Malte Möser, Kyle Soska, Ethan Heilman, Kevin Lee, Henry Heffan, Shashvat Srivastava
Kyle Hogan, Jason Hennessey, Andrew Miller, Arvind Narayanan, Nicolas Christin

PETS 2018: The 18th Privacy Enhancing Technologies Symposium

@edwardzpeng

Empirical Analysis



Info leak in Tx

- Similar user habits
- Relevant address
- Equal amount
- Similar time

Most of the shield transactions are traceable!

@edwardzpeng

Privacy of Lightweight Node

Most individual are using lightweight node

- Mobile phone, Embedded Wallet
- Do not have all block data

ZKP computation cost heavily

- Not so friendly to lightweight node
- Improving ☺

Needs to hand over the decryption key to a full node

- ZKP transaction need to be decrypted by full node
- Then relay to the lightweight node
- Almost no privacy for lightweight node ☹

"Privacy" versus "Ease of use"

There is no good solution in the market

Tron scanNote RPC

```
@Override
public void scanNoteByIvk(GrpcAPI.IvkDecryptParameters request,
    StreamObserver<GrpcAPI.DecryptNotes> responseObserver) {
  long startNum = request.getStartBlockIndex();
  long endNum = request.getEndBlockIndex();
  try {
    DecryptNotes decryptNotes = wallet
        .scanNoteByIvk(startNum, endNum, request.getIvk().toByteArray());
    responseObserver.onNext(decryptNotes);
  } catch (BadItemException | ZksnarkException e) {
    responseObserver.onError(getRunTimeException(e));
  }
  responseObserver.onCompleted();
}
```

Lightweight node Needs to hand over the decryption key to a full node

@edwardzpeng

Info leak in Tx

Reason:

Almost no privacy for lightweight node

Most Exchange only supports transparent transaction

Countermeasures:

Use shield transaction

Use new address each time

Split the amount

Wait for enough time

Best practice:

Only use shield transaction

t

Crypto Fail

Risk in Crypto

ZKP technology is relatively new

- Paper publish in 2016, large scale use in 2017
- Some kind of radical

Parameter selection and optimization are also radical

- Time will tell the truth

Provable security

- Rely on too many hard problems
- Some problems are not standard
- The proof itself need more auditing

Zcash Counterfeiting Vulnerability

CVE-2019-7167

- Discover in 2018/03 (Ariel Gabizon), publish in 2019
- Vulnerability in ZKP system [BCTV14]
- Anyone can create fake proof -> create fake Zcash
- Affecting multiple projects

It take 8 month to fix

- Change the whole proof system and upgrade the whole network
- It’s Zero knowledge -> No one knows whether it has been exploited

Zcash official “believe” that it has not been exploited：

- Few people have the skill to find this vulnerability
- Total amount of Zcash seem remain unchanged

Zcash Counterfeiting Vulnerability

CVE-2019-7167

- [BCTV14] doesn't have provable security
- [BCTV14] redundant elements in parameter result in fake proof
- The basic idea of this attack is quite simple

[BCTV14] upgrade to [Groth16]

- Provable security

Another crypto bug found by Bryan Parno before

- Happens to be unexploitable

Will it be the last time?

Zcash Hash Collusion Attack

Zerocash hash collusion attack

- Commitment: $COMM_{rcm}$ and $COMM_s$
  Need to be computationally bind
- Truncated to 128bits, result in 64bits security
- Result in double spend
- Potentially creating arbitrary amount of currency

Other issues

Perfect Zero Knowledge?

Perfect zero knowledge scheme in Mathematics

[Groth16]

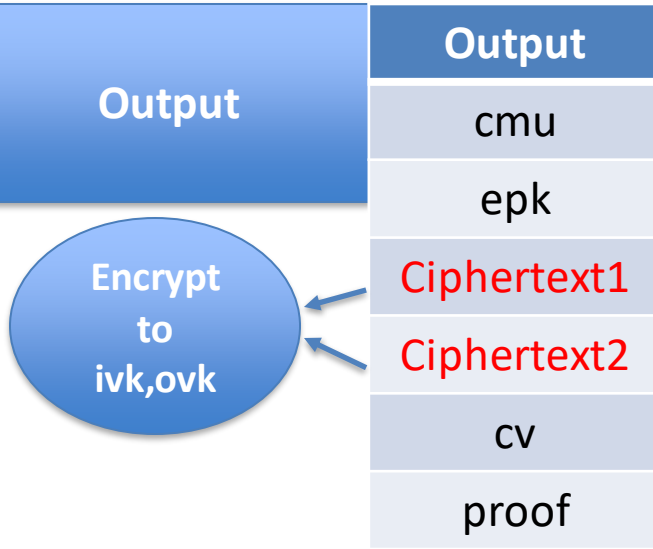
Info leak in real usage:

Zcash：

Proof is perfect zero knowledge

Ciphertext security rely on hard problem

Transaction is not zero knowledge



Perfect Zero Knowledge ≠ Perfect Privacy

Maybe 20 or 30 years latter, the encryption is broken

Your privacy will gone, remember every thing is on the chain☹

@edwardzpeng

Unlinkability of diversified addresses

Rely on the DDH problem in JubJub curve

Not a standard ECC curve, optimize for ZKP performance

Maybe broken someday in the future

All the transaction is on the chain,

Unlinkability will not remain forever.

Side Channel Attack

Always ignored in traditional software security

- hard to exploit
- Not directly result in compromise

Very important in a privacy system

- Privacy is directly compromised

Groth16 side channel attack

Timing attack:

To compute a proof, you need to calculate (A,B,C)

$$A = \alpha + \sum_{i=0}^{m} a_i u_i(x) + r\delta$$

$$B = \beta + \sum_{i=0}^{m} a_i v_i(x) + s\delta$$

$$C = \frac{\sum_{i=\ell+1}^{m} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x)) + h(x)t(x)}{\delta} + As + rB - rs\delta$$

Computing time of A,B,C strongly related with private input a_i

Other side channel: Cache

@edwardzpeng

ZKP for Hackers

@edwardzpeng

Key Theft

Bob need a signing key of some authority,

He found Alice on the Dark net.

But they don’t trust each other,

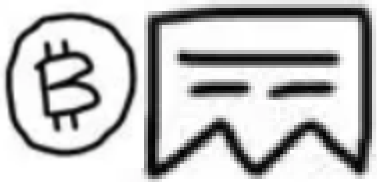
and there is no an trusted third party.

How can they fairly make a deal without trust?

Smart Contract: Key Theft

Alice

Bob

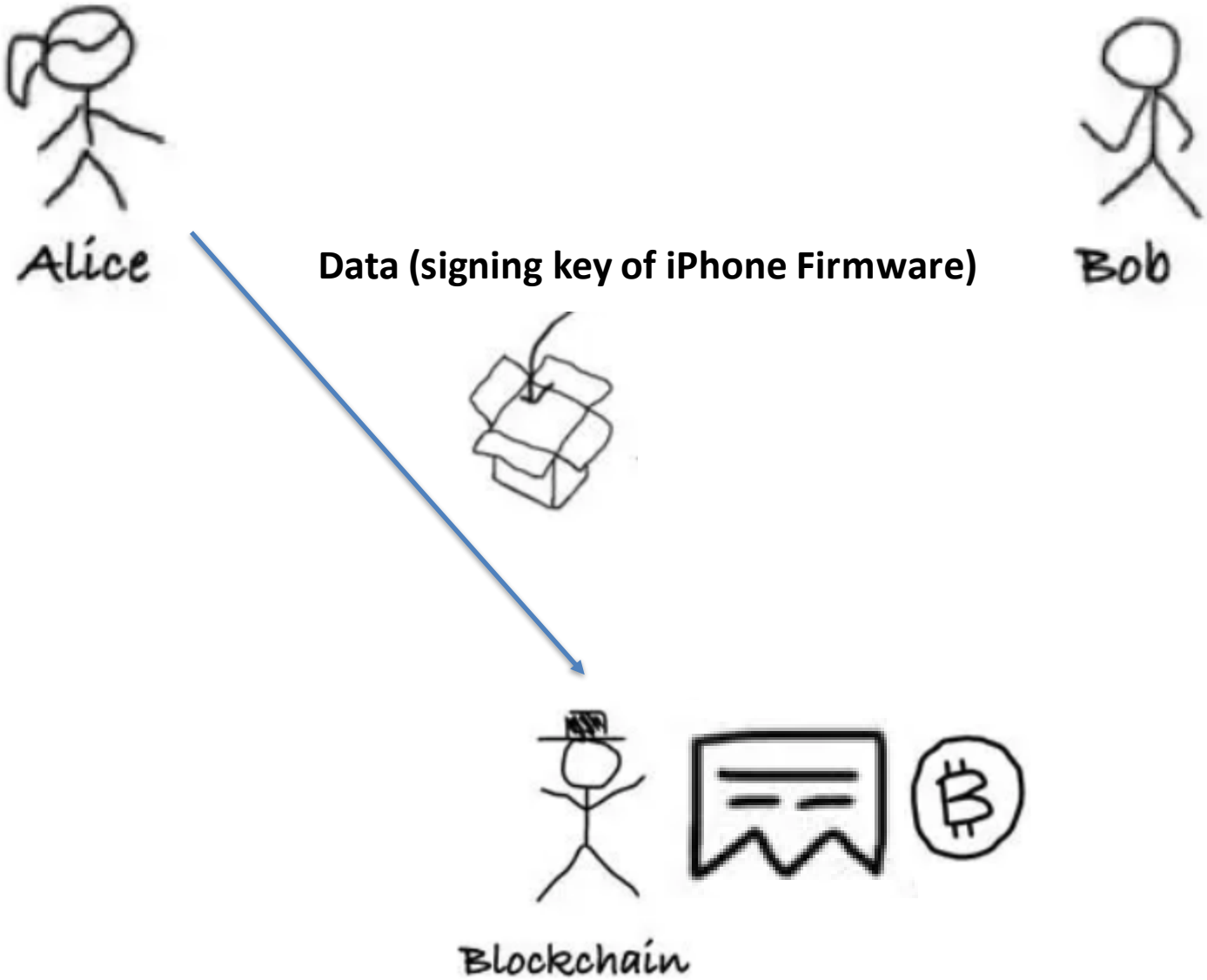


Setup an Contract
If data x satisfied f(x)=0,
then send money to Alice

Blockchain

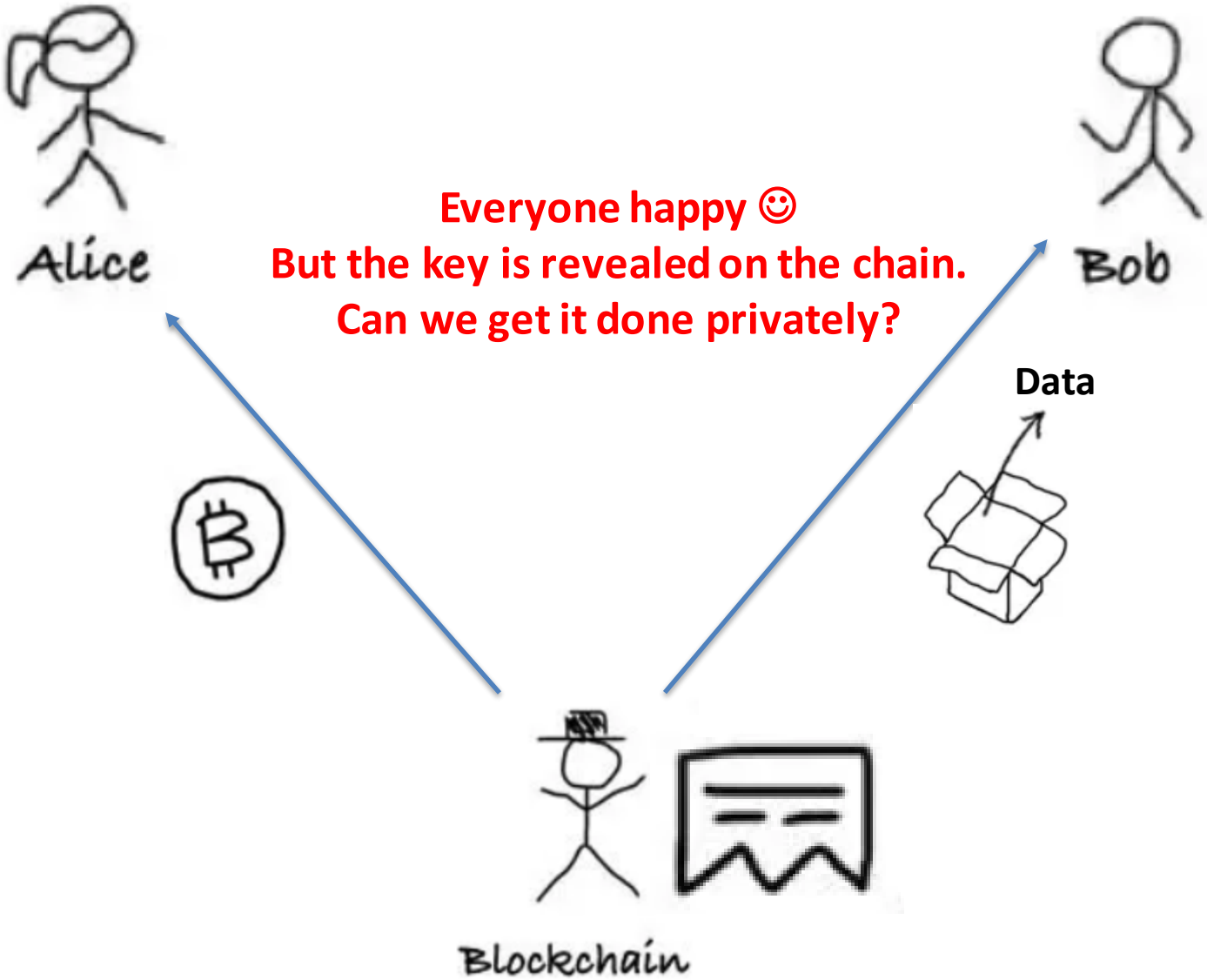
@edwardzpeng

Smart Contract: Key Theft



@edwardzpeng

Smart Contract: Key Theft



@edwardzpeng

Zero Knowledge contingent Payment (ZKCP)

Alice

Data (signing key of iPhone Firmware)

Encrypted data

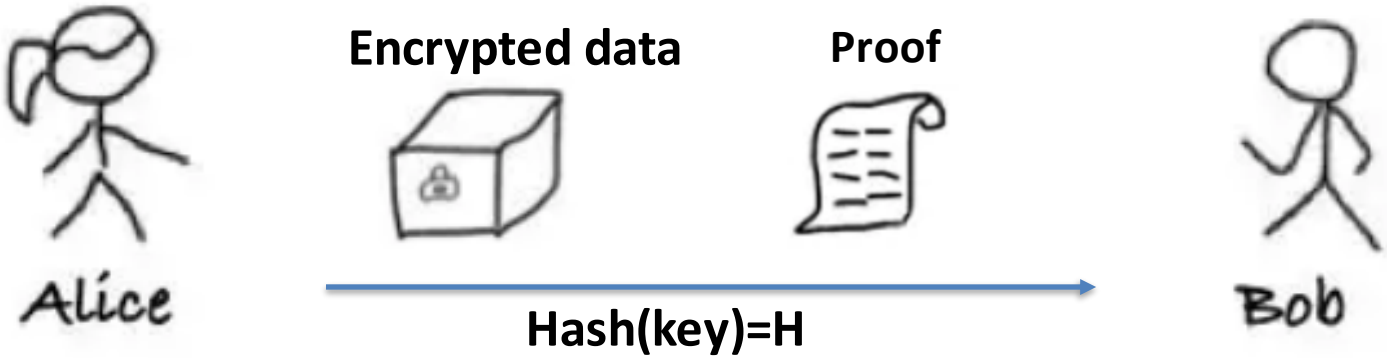
Proof:

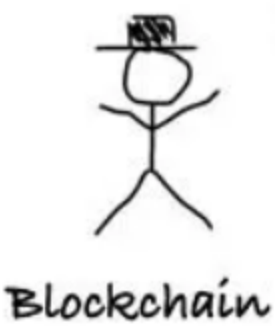
The data satisfied f(x)=0

Hash(Key)=H

@edwardzpeng

ZKCP



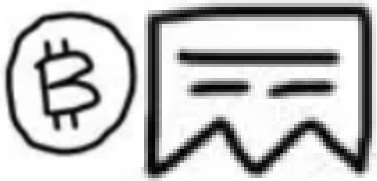


@edwardzpeng

ZKCP

Alice

Bob

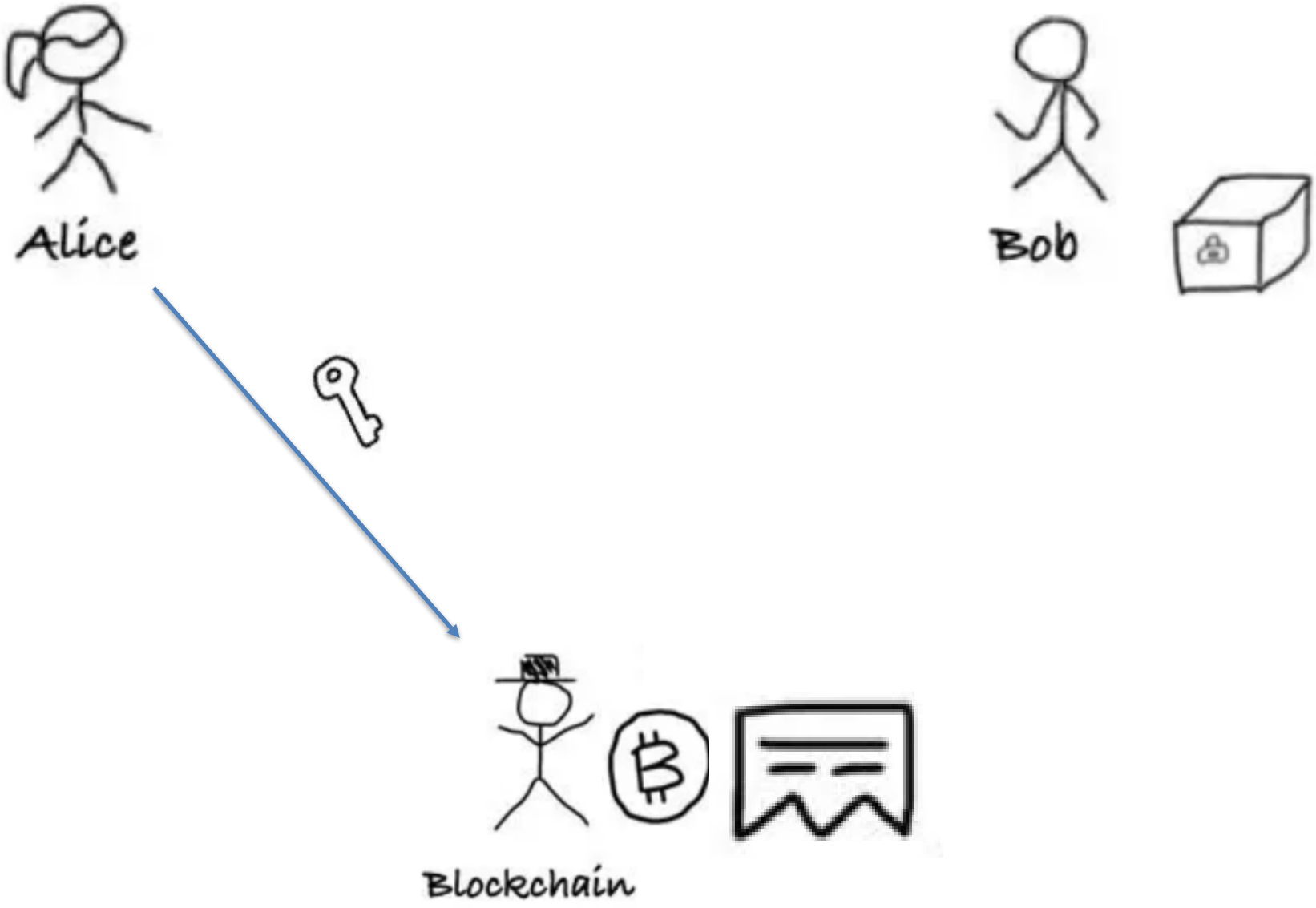


Setup an Contract
If K satisfied Hash(K)=h,
then send money to Alice

Blockchain

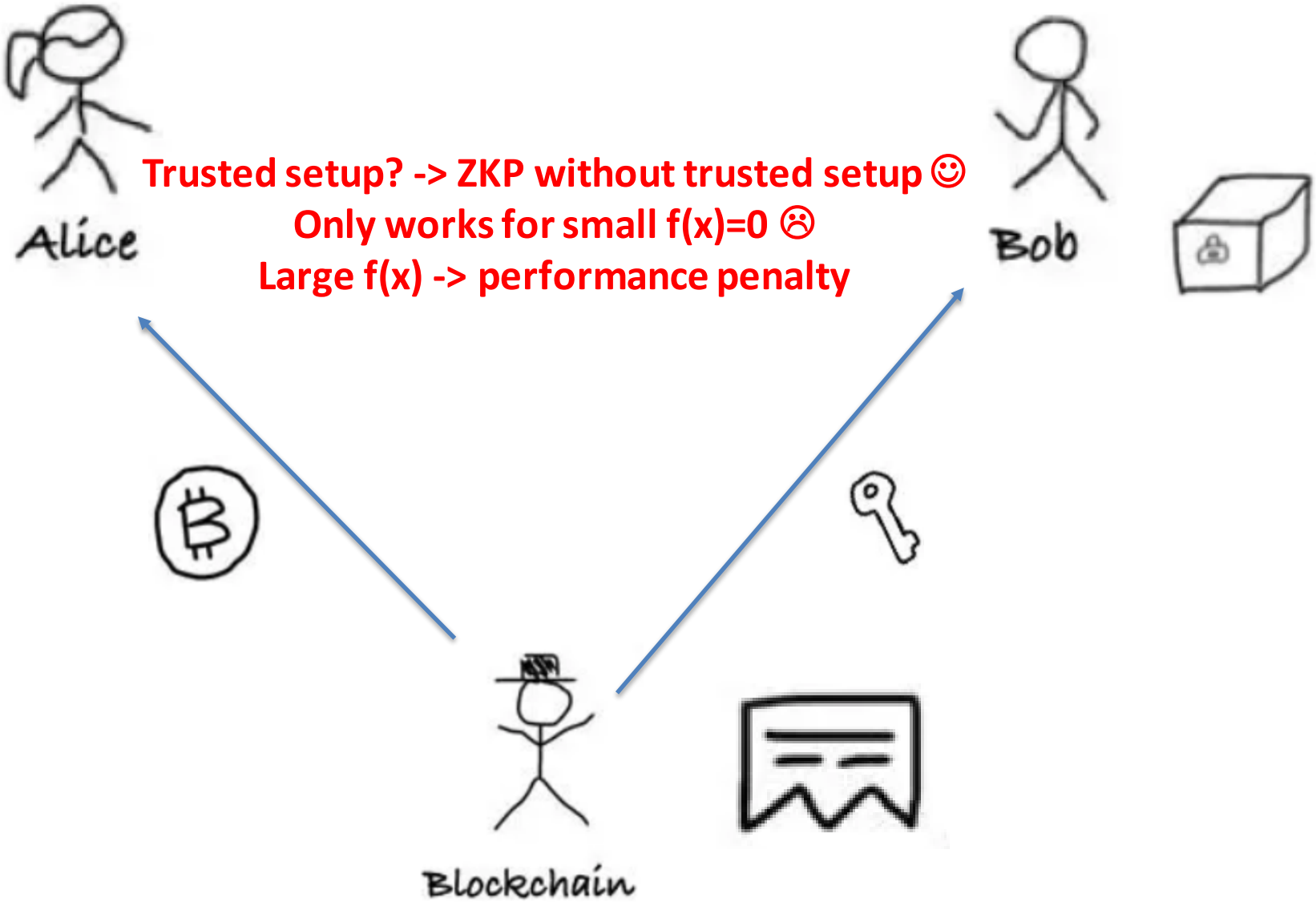
@edwardzpeng

ZKCP



@edwardzpeng

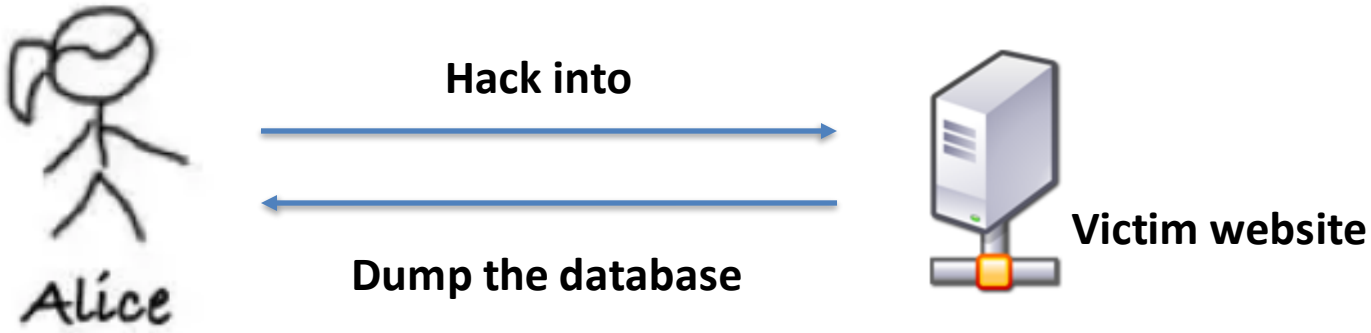
ZKCP



Trusted setup? -> ZKP without trusted setup ☺
Only works for small f(x)=0 ☹
Large f(x) -> performance penalty

@edwardzpeng

Selling Hacked Database



	Username	Telephone	Email	Password
X1	Edward	77777	e@360.cn	edward
X2	Zhiniang	88888	z@360.cn	123456
	……	……	……	……
Xn	Peng	99999	a@360.cn	201911

Fake database?

What if the database is a fake one?

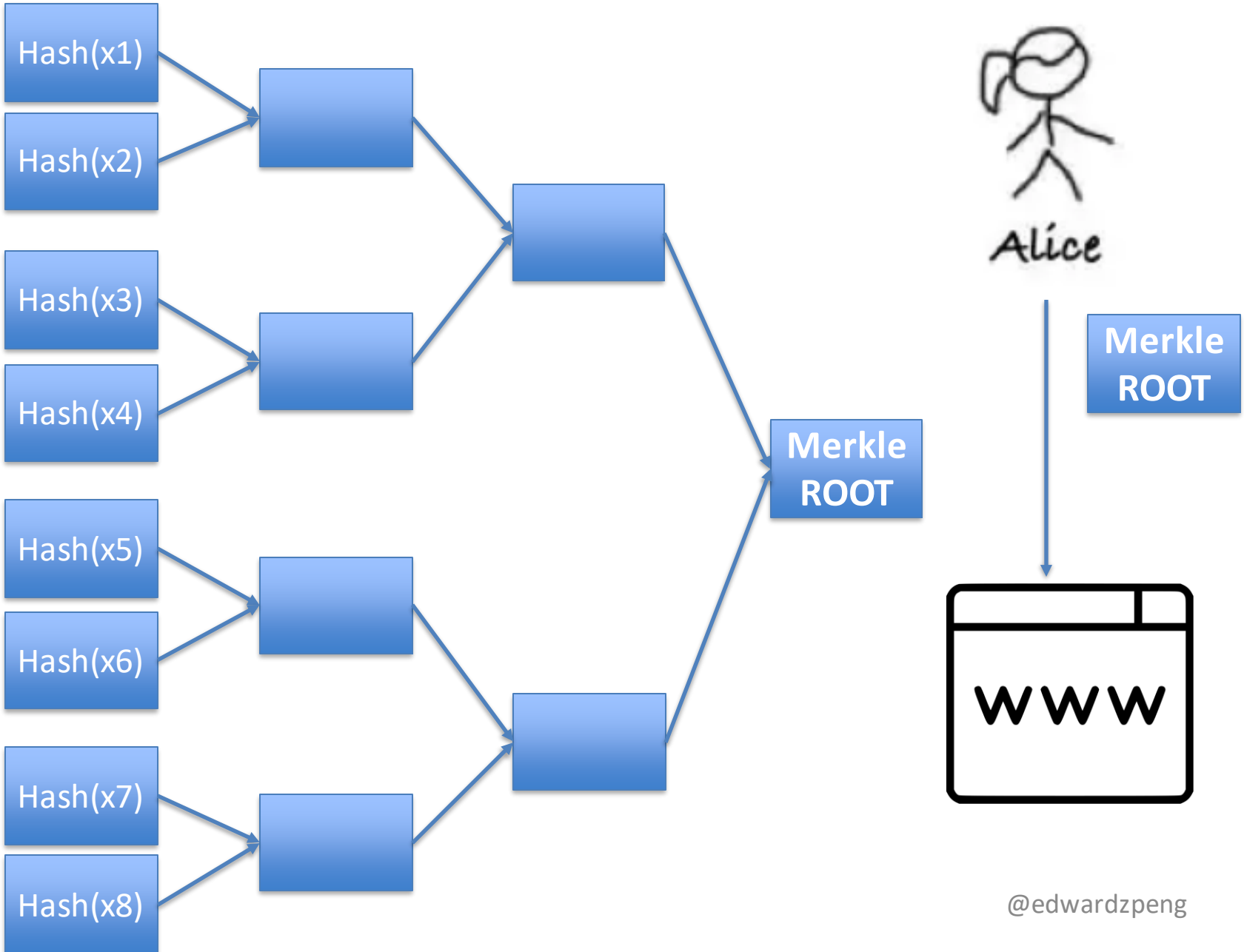
There is no function f(x) to measure the correctness

Solution:

Alice can commit to the database

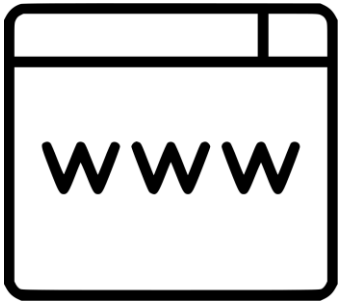
Bob buy some data to check

Publish a commitment



@edwardzpeng

Query for specific user



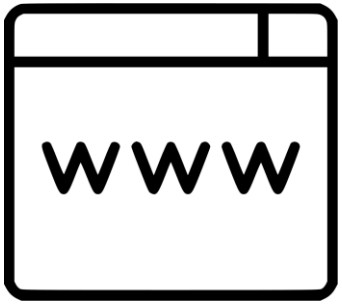
Query for “Zhiniang”

Search the database
X2 contain “Zhiniang”

X2 **Encrypted data**

@edwardzpeng

Reply



Query for “Zhiniang”

Encrypted X2

Proof:

X2 contain “Zhiniang”

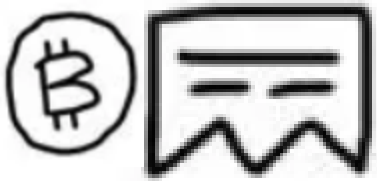
X2 is in the Merkle Tree

Hash(Key)=H

ZKCP

Alice

Bob

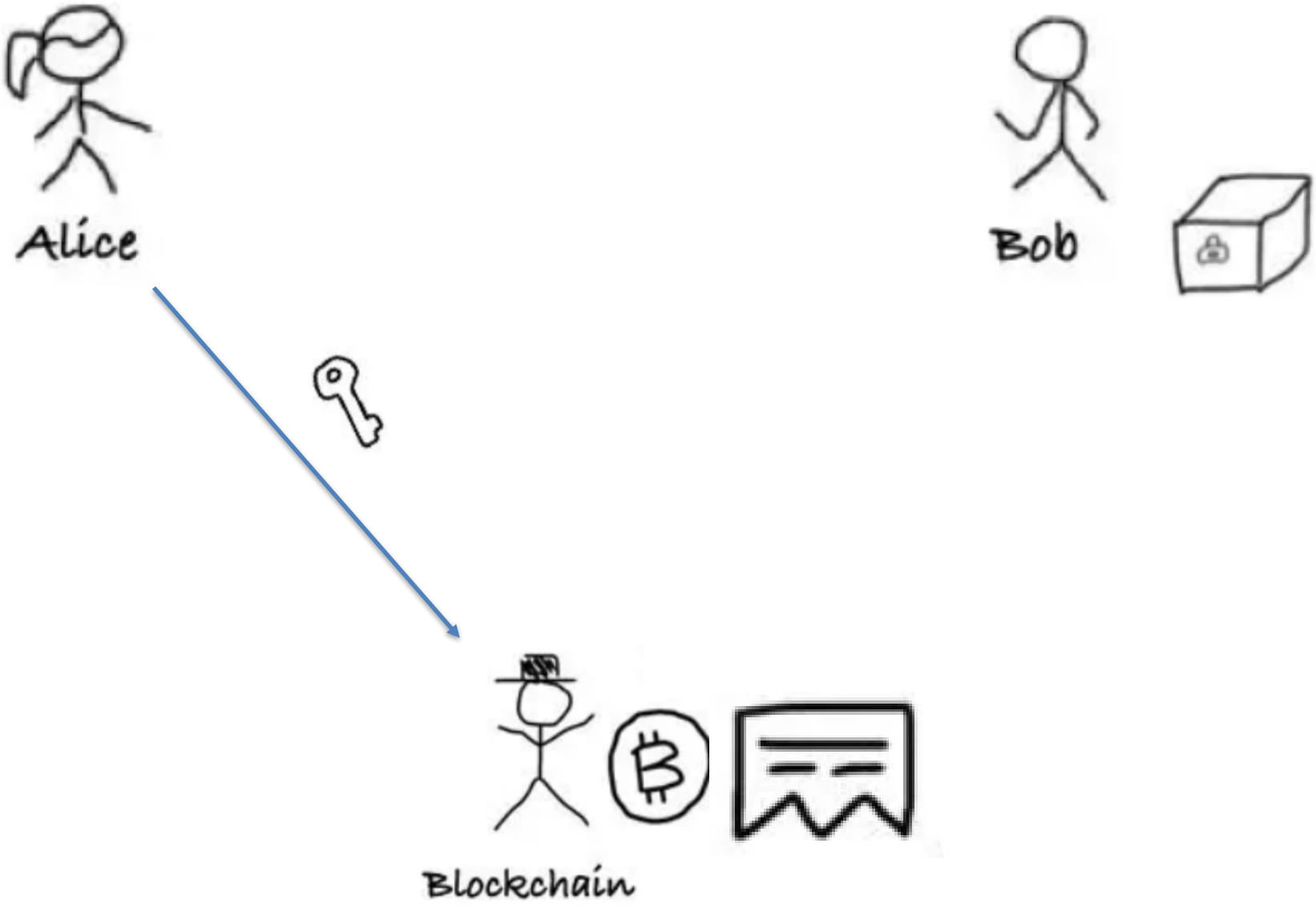


Setup an Contract
If K satisfied Hash(K)=h,
then send money to Alice

Blockchain

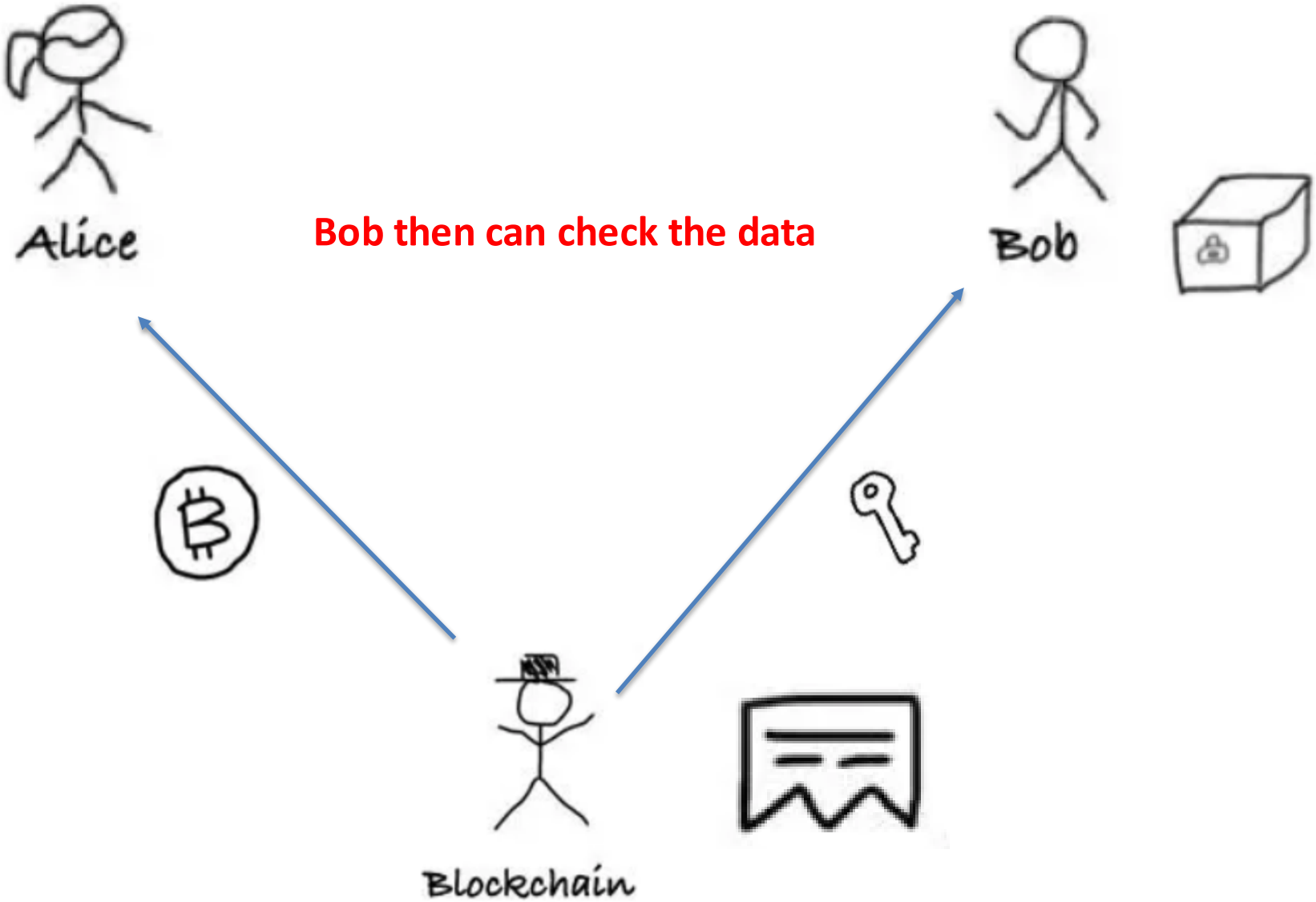
@edwardzpeng

ZKCP



@edwardzpeng

ZKCP



@edwardzpeng

Performance Problem

The performance?

The database may be more than 100G

Directly use ZKP on them will be a extremely slow

Efficient fair exchange protocol:

Fairswap: https://eprint.iacr.org/2018/740.pdf

zk-pod: https://github.com/sec-bit/zkPoD-node

Performance is reasonable

Can Alice fairly sell a 0day exploit?

F(X’)

Theoretically possible

Prove F(X’)=Run(“calc.exe”)

Then sell X’

Difficulty

Simulate F(X) correctly

Performance

zkSNARKs for Von Neumann Architecture

https://eprint.iacr.org/2013/879.pdf

@edwardzpeng

Conclusions

ZKP technology is relatively new and develops rapidly

There are risks, but it is improving

ZKP cryptocurrency can provide a strong anonymity

If you use it really carefully

ZKP application for hackers

Try to build your own protocol

Thanks

@edwardzpeng