UNBOUND

A Full CryptoCurrency Custody Solution Based on MPC and Threshold ECDSA

Yehuda Lindell, Bar-Ilan University and Unbound Tech

Ariel Nof, Bar-Ilan University

Samuel Ranellucci, Unbound Tech

Motivation

BITCOIN CRIME SEPTEMBER 29, 2018 23:50 CET

Oklahoma Duo Arrested for $14 Million Cryptocurrency Theft

BITCOIN

$1.1 billion in cryptocurrency has been stolen this year, and it was apparently easy to do

By BLOOMBERG September 20, 2018

Hackers stole $60 million of digital coins from a Japanese exchange, the latest in a string of thefts that have kept many institutional investors wary of putting their money in cryptocurrencies.

- **June, 2011** – 25,000 BTC ($775k by a user known as "ALLINVAIN")
- **March, 2012** – 46,703 BTC ($6M on Bitcoinica)
- **September, 2012** – 24,000 BTC ($250k on Bitfloor)
- **November ,2012** – 263,024 BTC ($3.42M on Bitcoin Saving & Trust)
- **November, 2013** – 1,296 BTC ($1.46M on BIPS); 4,100 BTC ($5.6M on Inputs); $6,000 BTC ($6.7M on PicoStocks)
- **February, 2014** – 650,000 BTC ($368M on MT.GOX)
- **March, 2014** – 150 BTC ($101k on bitCoin); 896 BTC ($572k on Flexcoin)
- **July, 2014** – 3,700 BTC ($2M on Mintpal $2m); 5000 BTC ($1.8M on Bitpay)
- **January, 2015** – 7,170 BTC ($1.82M on BTer.com); 3,000 BTC ($777k on Kipcoin); 1,000 BTC ($230k on 796 Exchange); 18,866 BTC ($4.3M on BITSTAMP)
- **March, 2015** – 150 BTC ($3.2k on COINAPULT)
- **May, 2015** – 1,500 BTC ($350k on BITFINEX)
- **October, 2015** – 10 BTC ($3.2k on Purse)
- **January, 2016** – 13,000 BTC; 3,000,000 Litecoin ($5.8M on Cryptsy)
- **March, 2016** – 81 BTC ($33k on COINTRADER); 469 BTC, 5,800 ETH 1,900 Litecoins ($230k on ShapeShift)
- **May, 2016** – 250 BTC, 185,000 ETH, 1,900 Litecoin ($2.14M on gatecoin)
- **June, 2016** – 3,500,000 ETH ($53M on DAO)
- **July, 2016** – $85k on Steemit
- **August, 2016** – 119,756 BTC ($65M on Bitfinex)
- **October, 2016** – 2,300 BTC ($2.6M on Bitcurex)
- **February, 2017** – $444,000 on erocoin
- **July, 2017** – 153,037 ETH (Parity); 37,000 ETH ($7M on COINDASH); $1M on Bithumb; $8.5M on Veritaseum
- **August, 2017** – 1,500 BTC ($500k on enigma)

Custody and Protection

- **Cryptocurrency protection needs**
  - **Exchanges**
    - High turnover – need to speed up transactions
      - Can take days to weeks today on exchanges
    - Separation of vaults (large, medium, small)
      - Higher protection on larger vaults

  - **Custody solutions** (for banks/financial institutions)
    - Small turnover – complex transactions acceptable (and desired)
    - Very large amount of funds
    - Offered to high-end customers

  - **Wallets**
    - For end users, small amounts of funds

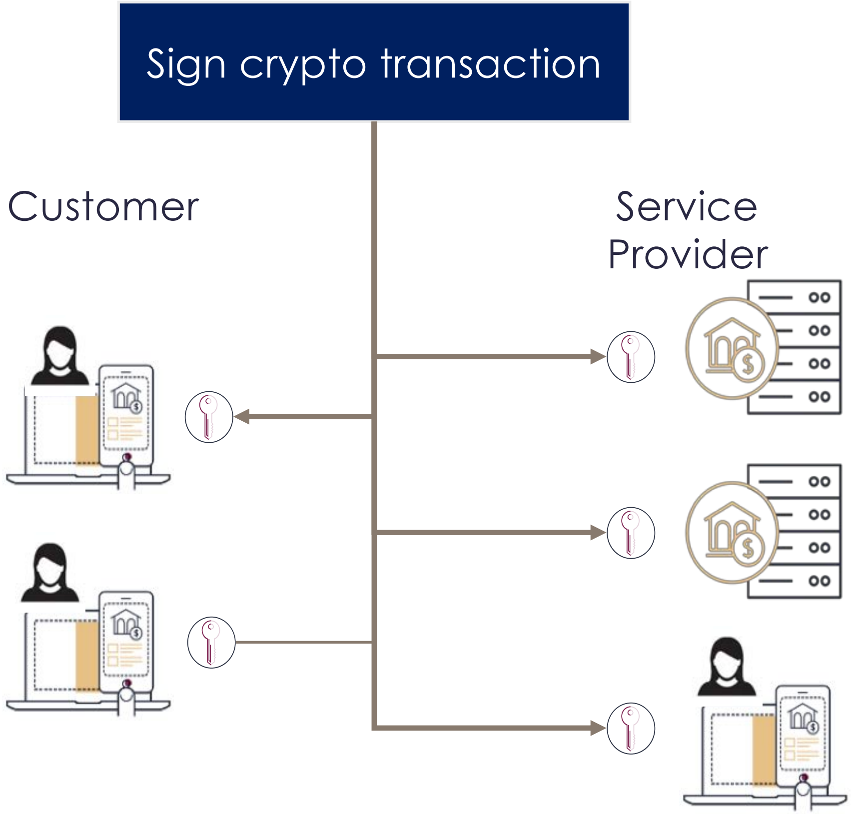
Solution Platform Requirements

- **High security**
  - Protection against key theft and *fraudulent key usage*
- **Backup and disaster recovery**
- **Flexibility**
  - Fine tune security vs usability (ease of transfer)
  - Broad support
    - Different coins/systems
    - Different signing algorithms
    - Standards (e.g., BIP032/BIP044)

Cryptographic Core – Threshold Signing

- **Multiparty protocols with full threshold security for malicious adversaries**
  - Support ECDSA, EdDSA, Schnorr
  - Supports distributed key generation
  - Achieves proactive security (post-compromise security)
- **Rich access structures supported for all**
  - Support AND/OR of sets of parties
  - Different structures for different levels of sensitivity/security
- **Two types of parties:**
  - Online signing parties: run actual MPC protocol (hold subset of shares)
  - Offline authorization parties: approve operation and provide their shares to online parties to carry out protocol

Custody Setting



A Protocol vs a Platform

- **Threshold cryptographic core is central, but not enough**
- **Many other elements needed, and influence cryptographic core**
  - Secure backup and disaster recovery
  - Support standard key derivation
  - Proactive security (post-compromise security)
  - Party administration (add/remove parties)
- **The above all needs to work with the core threshold signing protocol**

Our Solution – Additional Components

- **Publicly-verifiable backup with ZK proofs**
  - RSA (or any) public key provided (don't need additively homomorphic)
    - Each party encrypts its share of the key with RSA
    - Each party proves in ZK that the encryption is correct
      - For share x_i all parties know $Q_i = x_i \cdot G$, and so statement is well defined
- **ZK proof idea**
  - Encrypt r_i and $r_i - x_i$, and publish $R_i = r_i \cdot G$
  - Upon challenge to open one, let t_i be decrypted value
    - If open r_i, verify that $R_i = t_i \cdot G$
    - If open $r_i - x_i$, verify that $R_i - Q_i = t_i \cdot G$
  - Use Fiat-Shamir for public verifiability

Our Solution – Additional Components

- **Support BIP032/044 key derivation in MPC**
  - Derive all keys from master key – enables backup only of master key

- **BIP derivation in MPC**
  - Naïve: use garbled-circuit based MPC for SHA256/SHA512 derivation
    - Cheating party can input different key share and render backup useless
  - Verified:
    - We define MPC protocol that verifies that correct shares are input (utilizing public key)
    - Uses cut-and-choose like method inside circuit itself

Our Solution – Additional Components

- **Proactive (post compromise) security**
  - Refresh shares held by parties – breach of any subset of an authorized quorum in a period reveals nothing
  - Achieved by jointly generating and sharing a ***random polynomial*** with zero constant-term, and adding to shares
- **Party administration**
  - Re-share in order to replace parties
  - Necessary for offline parties, as expected to be for employees

Threshold ECDSA

- **Long-standing open problem to simultaneously achieve**
  - Full threshold (any number of corrupted parties), and
  - Efficient key generation
- **Two party:**
  - Based on Paillier additively-homomorphic encryption [MR04], [GGN16], [L17]
  - Based on OT [DKLS18] (higher bandwidth, faster time)
- **Multiparty:**
  - Honest majority [GJKR96]
  - Any number of corrupted [GGN16]
    - Key generation requires multiparty Paillier generation – impractical

New Threshold ECDSA Protocol

- **We present a new protocol (CCS 2018)**
  - Relies on hardness of Paillier and DDH
- **In parallel to this work [GG18] (also at CCS 2018)**
  - Similar performance (based on theoretical analysis)

ECDSA Signatures

- Let G be a generator of an EC group of order q
- Let m be the message to be signed

$R = (r_x, r_y) = k \cdot G$

$$s = k^{-1} \cdot (H(m) + r \cdot x) \bmod q$$

k is a random value

x is the secret key (shared among the parties in threshold case)

- The signature is (r, s)

Threshold ECDSA

- The main challenge: Compute k^{-1} and $R = k \cdot G$ for a random shared k in a secure distributed manner
  - This is non-linear and not "MPC friendly"

$R = (r_x, r_y) = k \cdot G$

$$s = k^{-1} \cdot (H(m) + r \cdot x) \bmod q$$

Solution of [GGN16]

- Each party chooses two random shares: k_i, ρ_i
  - Use additive sharing: $k = k_1 + \cdots + k_n$ and $\rho = \rho_1 + \cdots + \rho_n$
- Each party sends $k_i \cdot G$ and $Enc_{pk}(\rho_i)$ to all the other parties (+ ZK)
  - $R = k \cdot G = k_1 \cdot G + \cdots + k_n \cdot G$
  - Use additive homomorphism to get $Enc_{pk}(\rho) = \sum_{i=1}^{n} Enc_{pk}(\rho_i)$
- Each party sends $Enc_{pk}(k_i \cdot \rho) = k_i \cdot Enc_{pk}(\rho)$ to others (+ZK)
  - Use additive homomorphism to get $Enc_{pk}(k \cdot \rho) = \sum_{i=1}^{n} Enc_{pk}(k_i \cdot \rho)$
  - Run secure decryption to obtain $k \cdot \rho$ and locally compute $\rho^{-1} \cdot k^{-1}$
- This is enough to generate an encrypted signature

Instantiating the Additively Homomorphic Encryption

- **In [GGN16], used Paillier**
  - Distributed key generation for more than 2 parties is impractical
  - Complicated ZK proofs - working over 2 groups with different sizes…

Instantiating the Additively Homomorphic Encryption

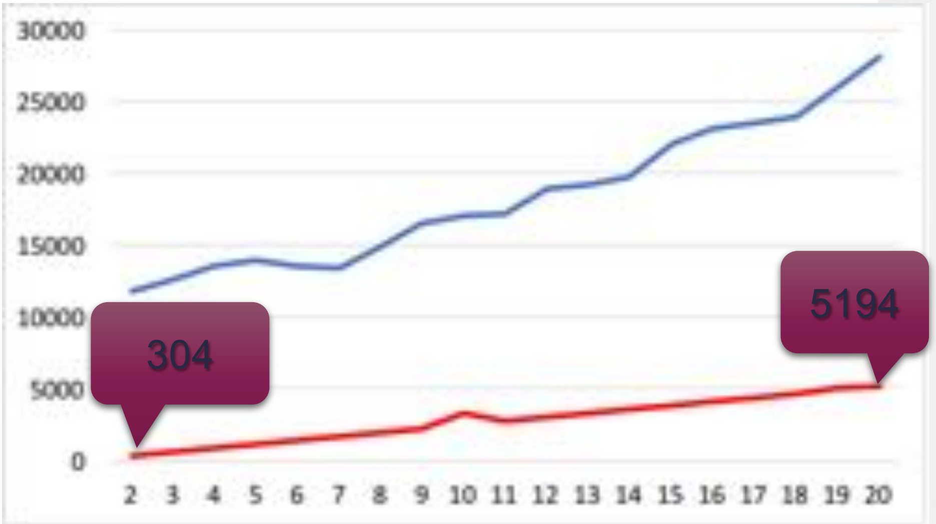
- **Our solution: use additively-homomorphic ElGamal-in-exponent**
  - $Enc_h(a) = (g^r, h^r \cdot g^a)$ (in EC notation, $Enc_h(a) = (r \cdot G, r \cdot h + a \cdot G)$)
  - Homomorphism:
    - Given $(u, v) = (g^r, h^r \cdot g^a)$ and $(x, y) = (g^s, h^s \cdot g^b)$ it holds that $(u \cdot x, v \cdot y) = \left(g^{r+s}, h^{r+s} \cdot g^{a+b}\right)$
    - Given $(u, v) = (g^r, h^r \cdot g^a)$ and c, have $(u^c, v^c) = (g^{r \cdot c}, h^{r \cdot c} \cdot g^{a \cdot c})$
  - Advantages
    - Encryption is in same group of the signature (no leakage)
    - Highly efficient ZK proofs
    - Highly efficient threshold key generation and decryption
  - Problem:
    - Cannot actually decrypt: "decryption" only reveals g^a (in EC: $a \cdot G$)

Decryption

- **In parallel to working in El Gamal**
  - Parties hold additive shares of values
  - *Addition*: locally add, and add El Gamal ciphertexts
  - *Multiplication by scalar*: run protocol to get additive shares of product, and multiply El Gamal in exponent
    - The multiplication protocol **only needs to be private**
- **Given shares of a and value $(g^r, h^r \cdot g^a)$, verify and reveal**
- **Private multiplication instantiations**
  - Based on oblivious transfer: very fast but very high bandwidth
  - Based on Paillier: low bandwidth, but more expensive
    - Paillier keys are local to each party (no distributed generation)

Experimental Results

- Experiments on AWS with 2.40GHz CPU, 1GB RAM, 1Gbps network
  - Single thread only
- Number of parties: 2 to 20 (all in the same region)
- Paillier-based private multiplication
  - ***OT much faster (order of magnitude)***
  - Open conjectures on Paillier



Summary

- **We present a new threshold ECDSA protocol**
  - Supports practical key generation, signing, and proactive security
  - Uses new paradigm for additively homomorphic encryption in MPC
- **Full platform with support for cryptocurrency protection**
  - Suitable for entire spectrum: wallet, exchange, custodian
    - Suitable also for other scenarios like CA signing
  - Includes verifiable backup, key derivation, online/offline parties
- **High security based on model of separation**

UNBOUND

Thank You

Two-party solution (for wallets) with ZK backup, verified BIP derivation, distributed key generation, refresh, and signing has been released as **open source**:

https://github.com/unbound-tech/blockchain-crypto-mpc