Preventing catastrophic cryptocurrency attacks

Neha Narula
MIT Digital Currency Initiative
Financial Cryptography 2019
MIT Digital Currency Initiative

**Educators**
Build industry capacity by teaching courses and advising students

**Conveners**
MIT has a history of standards setting, and providing a common platform
We’re neutral—no ICOs, most don’t hold material amounts of cryptocurrency

**Researchers**
Contribute research and core open-source development addressing scalability, privacy, and security
Cryptocurrency is not ready for billions of users

- Many challenges remain in scalability, interoperability, usability, and privacy
- There is increasing security risk with new, unproven protocols and latent implementation bugs
Current state of cryptocurrency security

- Thousands of cryptocurrencies and codebases
- Varied levels of security experience
- Attackers can easily and anonymously exploit vulnerabilities for financial gain
This talk

• Experience with a disclosure
• Lessons learned
• Open questions
Three vulnerabilities

- **IOTA**
  - Signature forgeries
  - Steal money
  - $1.2B

- **Bitcoin Cash**
  - Chain split
  - Double spend
  - $24B

- **Bitcoin**
  - DoS
  - Inflation
  - Halt network
  - Create new money
  - $116B
Important note

- All of these bugs were **disclosed** to developers
- As far as we know they were **not exploited**
- The developers all **deployed mitigations** for them
- These vulnerabilities **no longer impact** the security of any of the cryptocurrencies mentioned here
This talk

• Experience with a disclosure
  – A signature forgery attack on IOTA’s multisig
  – Breaking the Curl-P-27 hash function
  – Disclosure

• Lessons learned

• Open questions
800M dollar marketcap

Custom hash function called Curl
## IOTA Background: Terminology

<table>
<thead>
<tr>
<th></th>
<th><strong>Bitcoin</strong></th>
<th><strong>IOTA</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Payment</td>
<td>Transaction</td>
<td>Bundle</td>
</tr>
<tr>
<td>Currency 🍀</td>
<td>1 Bitcoin ~ $3.9K</td>
<td>1M IOTA ~ $0.30</td>
</tr>
</tbody>
</table>
# IOTA Background: Terminology

<table>
<thead>
<tr>
<th></th>
<th><strong>Bitcoins</strong></th>
<th><strong>IOTA</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Payment</td>
<td>Transaction</td>
<td>Bundle</td>
</tr>
<tr>
<td>Currency</td>
<td>1 Bitcoin ~ $3.9K</td>
<td>1M IOTA ~ $0.30</td>
</tr>
<tr>
<td>Representation</td>
<td>Bits (0, 1) bytes (8 bits)</td>
<td>Trits (-1, 0, 1) trytes (3 trits)</td>
</tr>
</tbody>
</table>
Why did we look at IOTA?

New cryptocurrency that solves all the problems! Scalable! No fees! Decentralized!

Tadge, you have to stop saying everything sucks. Prove it.

No.

Hey Ethan, take a look at this hash function…

Fine.

There goes my weekend!
What is our attack?

- Bob and Eve have funds under joint control and wish to spend them.
- Bob signs a payment where he gets $2M and Eve gets almost nothing.
- Eve forges Bob’s signature and instead sends a payment where she gets $2M and Bob gets almost nothing.
- Chosen message setting: Eve gets to create the payment Bob signs.
This talk

• Experience with a disclosure
  – A signature forgery attack on IOTA’s multisig
  – Breaking the Curl-P-27 hash function
  – Disclosure

• Lessons learned

• Open questions
What is multisig?

“Two-person” rule for nuclear launch
Using 2-of-2 multisig for payments

A valid payment requires $k$-of-$n$ signatures. Example 2-of-2:

Why multisig? Added security.

- Attacker has to compromise both keys
- Can store keys in isolated locations (cold storage)
- Used by many exchanges

Spending from a multisig address

$\text{sig}_{\text{Alice}} \quad \text{sig}_{\text{Bob}}$
IOTA Background: Signatures

IOTA's signature scheme:

- IOTA builds on Winternitz One-Time Signatures (WOTS)
- IOTA modifies WOTS
  ...to hash messages with Curl-P-27 prior to WOTS

IOTA_Sign(sk, m):
  \[ h_m = \text{Curl-P-27}(m) \]
  \[ \text{sig} = \text{WOTS_Sign}(sk, h_m) \]
  return sig
IOTA Background: Signatures

IOTA's signature scheme:
- IOTA builds on Winternitz One-Time Signatures (WOTS)
- IOTA modifies WOTS...to hash messages with Curl-P-27 prior to WOTS

The signature scheme details don’t matter because in IOTA, payments are hashed before they are signed.

If you can break the hash function, you can forge signatures!
Exploiting colliding bundles: Unauthorized payments

1. Eve creates **two** special bundles which have the **same** hash
2. Eve asks Bob to sign the bundle paying him
3. Eve **copies** Bob’s signature from the benign bundle to the evil bundle
4. Eve signs and broadcasts the evil bundle

-----

1) **Pays Eve**

2) **Pays Bob**

3) **Pays Eve**

4) Eve broadcasts this payment: **Pays Eve**

Bob never saw or authorized this payment!
Placing collisions to pay different amounts

<table>
<thead>
<tr>
<th>Payee</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice:</td>
<td>100</td>
</tr>
<tr>
<td>Eve:</td>
<td>1</td>
</tr>
<tr>
<td>Carol:</td>
<td>100</td>
</tr>
<tr>
<td>Bob:</td>
<td>2541865828330</td>
</tr>
</tbody>
</table>

- Target Value fields for differing trits
- Create two colliding bundles which differ in 26\textsuperscript{th} trit of two message blocks
Placing collisions to pay different amounts

<table>
<thead>
<tr>
<th>Payee</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice:</td>
<td>100</td>
</tr>
<tr>
<td>Eve:</td>
<td>1</td>
</tr>
<tr>
<td>Carol:</td>
<td>100</td>
</tr>
<tr>
<td>Bob:</td>
<td>2541865828330</td>
</tr>
</tbody>
</table>

- Target Value fields for differing trits
- Create two colliding bundles which differ in 26\textsuperscript{th} trit of two message blocks
- **Limitations:** Can only play this trick in specific places
This talk

• Experience with a disclosure
  – A signature forgery attack on IOTA’s multisig
  – **Breaking the Curl-P-27 hash function**
  – Disclosure process

• Lessons learned

• Open questions
To forge signatures we need to find colliding msgs for Curl-P-27:

Curl-P-27\((-1, 0, 1, 1, \ldots, -1) \equiv \text{Curl-P-27}(0, 1, 0, 0, 0, \ldots, 0)\)
Curl-P-27 uses a sponge-like construction

Security depends on the transform function $t$
The transformation function in Curl-P-27 is just the repeated application of a permutation + a simple S-Box

**AES S-Box**

<table>
<thead>
<tr>
<th>00</th>
<th>01</th>
<th>02</th>
<th>03</th>
<th>04</th>
<th>05</th>
<th>06</th>
<th>07</th>
<th>08</th>
<th>09</th>
<th>0a</th>
<th>0b</th>
<th>0c</th>
<th>0d</th>
<th>0e</th>
<th>0f</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>63</td>
<td>7c</td>
<td>77</td>
<td>7b</td>
<td>f2</td>
<td>6b</td>
<td>6f</td>
<td>c5</td>
<td>30</td>
<td>01</td>
<td>67</td>
<td>2b</td>
<td>fe</td>
<td>d7</td>
<td>ab</td>
</tr>
<tr>
<td>10</td>
<td>ca</td>
<td>82</td>
<td>c9</td>
<td>7d</td>
<td>fa</td>
<td>59</td>
<td>47</td>
<td>f0</td>
<td>ad</td>
<td>d4</td>
<td>a2</td>
<td>af</td>
<td>c9</td>
<td>a4</td>
<td>72</td>
</tr>
<tr>
<td>20</td>
<td>b7</td>
<td>fc</td>
<td>93</td>
<td>26</td>
<td>36</td>
<td>3f</td>
<td>f7</td>
<td>cc</td>
<td>34</td>
<td>a5</td>
<td>e5</td>
<td>f1</td>
<td>71</td>
<td>d8</td>
<td>31</td>
</tr>
<tr>
<td>30</td>
<td>04</td>
<td>c7</td>
<td>23</td>
<td>c3</td>
<td>18</td>
<td>96</td>
<td>05</td>
<td>9a</td>
<td>07</td>
<td>12</td>
<td>80</td>
<td>e2</td>
<td>eb</td>
<td>27</td>
<td>b2</td>
</tr>
<tr>
<td>40</td>
<td>89</td>
<td>b6</td>
<td>a7</td>
<td>7c</td>
<td>5e</td>
<td>3b</td>
<td>db</td>
<td>e3</td>
<td>0c</td>
<td>ed</td>
<td>70</td>
<td>3d</td>
<td>79</td>
<td>ad</td>
<td>c6</td>
</tr>
<tr>
<td>50</td>
<td>25</td>
<td>19</td>
<td>13</td>
<td>e5</td>
<td>0c</td>
<td>a5</td>
<td>af</td>
<td>be</td>
<td>c3</td>
<td>1d</td>
<td>e1</td>
<td>11</td>
<td>6b</td>
<td>ea</td>
<td>2d</td>
</tr>
<tr>
<td>60</td>
<td>04</td>
<td>5b</td>
<td>67</td>
<td>76</td>
<td>55</td>
<td>e2</td>
<td>65</td>
<td>c9</td>
<td>17</td>
<td>18</td>
<td>1e</td>
<td>1c</td>
<td>7c</td>
<td>69</td>
<td>1a</td>
</tr>
<tr>
<td>70</td>
<td>3b</td>
<td>1d</td>
<td>7a</td>
<td>c9</td>
<td>1b</td>
<td>c1</td>
<td>9b</td>
<td>5e</td>
<td>e6</td>
<td>7f</td>
<td>44</td>
<td>47</td>
<td>b2</td>
<td>ee</td>
<td>59</td>
</tr>
<tr>
<td>80</td>
<td>2a</td>
<td>0d</td>
<td>69</td>
<td>f5</td>
<td>16</td>
<td>d5</td>
<td>e8</td>
<td>a7</td>
<td>1d</td>
<td>6d</td>
<td>69</td>
<td>2b</td>
<td>dl</td>
<td>ea</td>
<td>6a</td>
</tr>
<tr>
<td>90</td>
<td>fe</td>
<td>46</td>
<td>96</td>
<td>62</td>
<td>8e</td>
<td>1b</td>
<td>5c</td>
<td>e8</td>
<td>17</td>
<td>26</td>
<td>4b</td>
<td>c7</td>
<td>d7</td>
<td>9d</td>
<td>c5</td>
</tr>
</tbody>
</table>

**Curl-P-27 S-Box**

![Curl-P-27 S-Box](image)
Curl-P-27: Reducing collision resistance

Choose a random message $-1011110101\ldots-1$

Flip a trit

$-1011010101\ldots-1$

If we are clever about choosing the message this increases to

$>1/2^{22.87} = 1\text{ out of 7.6 million}$

In cryptographic terms this is 23-bit collision resistance.
IOTA bundles: unconstrained tag field

As the likelihood of a collision is at least 1 out of 7.6 million we need to try many messages (bundles) before we are successful

<table>
<thead>
<tr>
<th>address</th>
<th>tag</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DKSDJFLS...R</td>
<td>99999...JKQGB</td>
<td>22000000...</td>
</tr>
<tr>
<td>QWEWEABZ...9</td>
<td>99999...LKQCB</td>
<td>00000010...</td>
</tr>
<tr>
<td>ABEPCMQQ...Z</td>
<td>99999...VBN99</td>
<td>00050000...</td>
</tr>
</tbody>
</table>

We can change the 81-trit tag field in IOTA bundles
Tags have no impact on transaction validity
Curl-P-27 modifies sponge to overwrite

Differences are erased as new message blocks overwrite the first third of the state
How do we create collisions in Curl-P-27?

Plan: ensure all the diffs are in first 3rd of the state
github.com/mit-dci/tangled-curl
This talk

• Experience with a disclosure
  – A signature forgery attack on IOTA’s multisig
  – Breaking the Curl-P-27 hash function
  – Disclosure

• Lessons learned

• Open questions
IOTA fixes our signature forgery vulnerability

In July 2017 we disclosed this to the IOTA devs...
...in response the IOTA devs replaced Curl-P-27 with Kerl

---

<table>
<thead>
<tr>
<th>Kerl is used in IOTA for the following tasks:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
</tr>
<tr>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>Address generation</td>
</tr>
<tr>
<td>Signature generation</td>
</tr>
<tr>
<td>Signature verification</td>
</tr>
<tr>
<td>Essence calculation (bundleHash)</td>
</tr>
<tr>
<td>Proof of Work</td>
</tr>
<tr>
<td>Transaction Hash</td>
</tr>
<tr>
<td>Milestone verification</td>
</tr>
</tbody>
</table>

Curl-P-N: $N$ number of rounds

https://github.com/iotaledger/kerl
IOTA claims this was a backdoor

“[..] Curl-P was indeed deployed in the open-source IOTA protocol code as a copy-protection mechanism to prevent bad actors cloning the protocol and using it for nefarious purposes. Once the practical collisions were uncovered, its purpose as a copy-protection mechanism was of course rendered obsolete.”

In response to Ethan’s question “Did we discover a copy-protection backdoor in IOTA?” they write: “The answer to the first question is of course, yes, as we have explained above.”

Read IOTA’s full statement at blog.iota.org/11fdccc9eb6d
Troika: a ternary hash function

Reference document

Version 1.0.1
December 21, 2018
A new hash function appears

- In December 2018 IOTA announced the creation of a new ternary hash function Troika designed by Cybercrypt A/S
- €200,000 prize pool to break round-reduced variants

“Currently IOTA uses the relatively hardware intensive NIST standard SHA-3/Keccak for crucial operations for maximal security.”

“[...] we [...] started tackling the hardware side with new thinking in computational processing. A next generation of microprocessor architecture based on ternary logic for ultimate efficiency in IoT is the result. (A deep dive blog post on trinary’s superiority over binary will come soon).”

Read IOTA’s full statements at blog.iota.org/678e741315e8 and blog.iota.org/615d2df79001
This talk

• Experience with a disclosure
  – A signature forgery attack on IOTA’s multisig
  – Breaking the Curl-P-27 hash function
  – Disclosure process

• Lessons learned

• Open questions
Lessons learned
(for disclosers)

- Expect wildly different types of responses
- Be prepared to obtain legal representation
- Consider disclosing anonymously
Lessons learned
(for cryptocurrencies)

- Have a responsible disclosure policy
  - Contact address, GPG keys
- Support anonymous communication
Other reasons to disclose anonymously

- Potential to exploit vulnerability and make a lot of money
- Also potential to cause others to lose a lot of money
- If a vulnerability is exploited, you become a suspect and target

Cryptocurrencies should consider commensurate bounties!
There was no disclosure policy

It was hard to find contact information for developers

It was hard to contact them anonymously

It was hard to confirm receipt all since fixed!

medium.com/mit-media-lab-digital-currency-initiative/48a99b85aad4
Lessons learned (for cryptocurrencies)

- Have a responsible disclosure policy
  - Contact address, GPG keys
- Support anonymous communication
- Forge relationships with researchers and related implementations
Next vulnerability in bitcoin-core was disclosed by a Bitcoin Cash developer (u/awemany)

Roger Ver
Aug 9, 2018
Thank you for the responsible disclosure Cory. It is appreciated by myself and others in the community.

Paulo Falcao
Aug 9, 2018
It's people like you that makes me believe in crypto. Well done!

Alex Martell
Aug 9, 2018
Amazing story, amazing and laudable ethos. Every BCH holder owes you big time.
And probably every BTC holder too :)
Thank you
This talk

• Experience with a disclosure
  – An signature forgery attack on IOTA’s multisig
  – Breaking the Curl-P-27 hash function
  – Disclosure process

• Lessons learned

• Open questions
Open questions  
(for everyone)

- How do we coordinate disclosures across multiple cryptocurrencies?
- How should developers communicate the vulnerability and its mitigation across the cryptocurrency’s ecosystem?
1. Hide a fix for the inflation bug inside a fix for the DoS bug

2. Tell everyone about the DoS bug and fix to get them to upgrade as fast as possible

This effectively dropped a 0-day on many coins derived from bitcoin-core
Open questions (for everyone)

- How do we coordinate disclosures across multiple cryptocurrencies?
- How should developers communicate the vulnerability and its mitigation across the cryptocurrency’s ecosystem?
- Who should one even disclose to?
- Should the discloser or developers move vulnerable funds?
- How can we prevent vulnerabilities in the first place?
Maybe security doesn’t matter?

Price seems to be totally uncorrelated with vulnerabilities and attacks!

– Fixing exploits inspires confidence in developer teams
– The cryptocurrency market is currently small and irrational (it might not stay that way)
– Network attacks so far have been relatively small and those attacked are able to absorb the losses (it might not stay that way)
Cryptocurrency security is a public good

• A really bad attack could affect many coins and businesses

• Many bad attacks could reduce trust in cryptocurrencies and set us back years
Cryptocurrency security working group

1. Identify and circulate best practices
2. Write tests, run monitoring and security tools
3. Research to move to safer programming languages and on formal verification
Where to keep up with research?
Introducing

Cryptocurrency Research Review

discourse.mitcryptoresearch.org/
• Interdisciplinary (CS+economics+law)
• Experiment: speed, overlay, reviews, and submissions
• One place to look for high-quality, reviewed research
digital currency initiative
dci.mit.edu
@neha@mit.edu
digital currency initiative

@neha
dci.mit.edu
narula@mit.edu