## CSci 5271
## Introduction to Computer Security
## Day 24: Electronic cash and Bitcoin

Stephen McCamant
University of Minnesota, Computer Science & Engineering

## Outline

Previous e-cash and techniques

Bitcoin design

Announcements, Ex. 3/4 debrief

Bitcoin experience

Bonus: anonymity overlays

## Kinds of Internet payments

- Credit/debit cards: most popular
  - Wide adoption among consumers, little consumer fraud liability
  - Restrictive merchant procedures
- PayPal
  - Easier to accept payments
  - Centrally managed to deal with fraud

## Ideal: electronic cash

- Direct transactions without third party
- No transaction fees
- Potentially anonymous
- Non-revocable: buyer bears fraud risk

## Micropayments

- Claim: what the web needs is small payments to support content
  - Too small for existing mechanisms
- One idea (Peppercoin): simulate small payment with small probability of larger payment
- Actual market for micropayments has been small
  - Most buyers and sellers prefer free + other revenue

## Blinded signatures

- Sign something without knowing its value
  - Often used together with randomized auditing
  - For RSA, multiple message by $r^e$, $r$ random
- Allows a bank to "mint" coins that can still be anonymous

## Challenge: double spending

- Any purely electronic data can be duplicated, including electronic money
- Can't allow two copies to both be spent
- Shows ideal no-third-party e-cash can't be possible

## Puzzles / proof-of-work

- Computational problem you solve to show you spent some effort
- Common: choose $s$ so that $h(m \parallel s)$ starts with many 0 bits
- For instance, required solved puzzles can be a countermeasure against DoS

## Hashcash and spam

- Idea: use proof of work to solve email spam problem
- Puzzle based on date and recipient
- Legitimate users send only a few messages
  - Problem 1: mailing lists
  - Problem 2: spam botnets
- Never caught on

## Hash trees and timestamp services

- Merkle tree: parent node includes hash of children
- Good hash function $\rightarrow$ root determines whole tree
- Can prove value of leaf with log-sized evidence
- Application: document timestamping (commitment) service

## Outline

Previous e-cash and techniques

Bitcoin design

Announcements, Ex. 3/4 debrief

Bitcoin experience

Bonus: anonymity overlays

## Bitcoin addresses

- Address is basically a public/private signing key pair
  - Randomized naming, collision unlikely
- At any moment, balance is a perhaps fractional number of bitcoins (BTC)
- Anyone one can send to an address, private key needed to spend

## Global transaction log

- Basic transaction: Take $x_1$ from $a_1$, $x_2$ from $a_2$, …, put $y_1$ in $a'_1$, $y_2$ in $a'_2$, …
  - Of course require $\sum_i x_i = \sum_j y_j$
- Keep one big list of all transactions ever
- Check all balances in addresses taken from are sufficient

## Bitcoin network

- Use peer-to-peer network to distribute transaction log
- Roughly similar to BitTorrent, etc. for old data
- Once a client is in sync, only updates need to be sent
- New transactions sent broadcast

## Consistency and double-spending

- If all clients always saw the same log, double-spending would be impossible
- But how to ensure consistency, if multiple clients update at once?
- Symmetric situation: me and "me" in Australia both try to spend the same $100 at the same time

## Bitcoin blocks

- Group ~10 minutes of latest transactions into one "block"
- Use a proof of work so creating a block is very hard
- All clients race, winning block propagates

## Bitcoin blockchains

- Each block contains a pointer to the previous one
- Clients prefer the longest chain they know
- E.g., inconsistency usually resolved by next block

## Regulating difficulty

- Difficulty of the proof-of-work is adjusted to target the 10 minute block frequency
- Recomputed over two-week (2016 block) average
- Network adjusts to amount of computing power available

## Bitcoin mining

- Where do bitcoins come from originally?
- Fixed number created per block, assigned by the client that made it
- Incentive to compete in the block generation race
- Called *mining* by analogy with gold

## Outline

Previous e-cash and techniques

Bitcoin design

Announcements, Ex. 3/4 debrief

Bitcoin experience

Bonus: anonymity overlays

## Group project presentations

- Start next week, run three lectures
- Plan 10 minute presentation plus 2 minutes Q&A
- One student per group presents
- Slides, BYO laptop recommended

## December deadlines

- Final project progress reports: Monday 12/1
- Exercise set 5: Thursday 12/4
- Project final reports: Wednesday 12/10

## TCP congestion control

- Congestion control is a voluntary mechanism
- Forge reset packets to misbehaving hosts?
  - Used in reality for other sorts of misbehavior
- Blacklist misbehaving addresses
  - Can be misused by a dishonest adversary
  - Note: MAC spoofing is local-net only

## Bad MACs

- Pre-authenticate by sending MAC of zeros
  - Related to problem of CBC-MAC on varying lengths
- CTR-Encrypt hash appended to the end
  - Encryption doesn't protect integrity
  - Especially stream-cipher style modes

## Protocol droids

- $A \to C$: $N_A$, $\text{MAC}_K(N_A)$
- $C \to A$, $\text{MAC}_K(\text{MAC}_K(N_A))$
- Problem 1: freshness
- Problem 2: oracle perspective

## Hashing and signing

- Problems with letting yourself do random things
  - General policy on security definitions
  - Problems in particular applications
- Effort to find a good/bad collision?
  - Generally-applicable extension of birthday attack

## Seeding a PRNG

- Entropy required for unpredictability
- Black-box attacks easy, reverse engineering also possible
- Bad ideas:
  - `time()`
  - Process ID
  - Time XOR PID
- How to do better?

## Web server false alarms

- Attack is unlikely to appear in benign traffic
  - Illegal UTF-8 rep. of path traversal
- Best way to inject false positives?
  - IP spoofing not easy for TCP
- Takeaway: FP/FN rates depend on attacker

## Outline

Previous e-cash and techniques

Bitcoin design

Announcements, Ex. 3/4 debrief

**Bitcoin experience**

Bonus: anonymity overlays

## Where Bitcoin came from

- Paper and early implementation by Satoshi Nakamoto
  - Generally presumed to be a pseudonym
- "Genesis block" created January 2009
  - Containing headline from The Times (of London) about a bank bailout

## Current statistics

- Block chain 331,500 blocks, ~30GB
- 13.5M BTC minted (many presumed lost)
- Theoretical value at market exchange rate > $1 billion
- Millions of addresses, probably many fewer users
- Mining power: 300 petahash/sec

## What can you buy with Bitcoin?

- Random stuff from many small online retailers
- Novelty/trials of some in-person purchases
- Donations to like-minded non-profits
- Illegal drugs (Silk Road successors)
- Murder for hire: currently probably a fraud

## Bitcoin as a currency

- Can be exchanged for dollars, etc.
  - Currently pretty cumbersome
- In some ways more like gold than fiat currencies
  - No central authority
  - Price changes driven more by demand than supply
- Exchange rate trend: volatile but upward(?)

## Deflation and speculation

- Some people want bitcoins to spend on purchases
  - Demand based on "velocity"
  - Supply does not keep up with interest
  - So, value of 1 BTC has to go up
- Others want bitcoins because they think the price will go up in the future
  - Self-fulfilling prophecy
  - But vulnerable to steep drops if expectations change

## Bitcoin mining trends

- Exponentially increasing rates
- CPU → GPU → FPGA → ASIC
- Specialized hardware eclipsing general purpose
  - Including malware and botnets
- Recent price trends suggest continuing investment

## Enforcing consistency

- Structure of network very resistant to protocol change
  - Inertia of everybody else's code
- Changes unpopular among miners will not stick
- Minor crisis last March: details of database lock allocation cause half of network to reject large block

## Stealing bitcoins

- Bitcoins are a very tempting target for malware
  - Private keys stored directly on client machines
  - Theft is non-reversible
  - Much easier than PayPal or identity theft
- Standard recommendation is to keep keys mostly offline

## Bitcoin (non-)anonymity

- Bitcoin addresses are not directly tied to any other identity
- But the block chain is public, so there's lots of information
  - List of largest balances on Wikipedia
  - Academic research: today's second paper
- Real unlinkability is a research topic

## Outline

Previous e-cash and techniques

Bitcoin design

Announcements, Ex. 3/4 debrief

Bitcoin experience

Bonus: anonymity overlays

## Traffic analysis

- What can you learn from encrypted data? A lot
- Content size, timing
- Who's talking to who

  → countermeasure: anonymity

## Anonymous remailers

- Anonymizing intermediaries for email
  - First cuts had single points of failure
- Mix and forward messages after receiving a sufficiently-large batch
- Chain together mixes with multiple layers of encryption
- Fancy systems didn't get critical mass of users

## Tor: an overlay network

- Tor (originally from "the onion router")
  - `https://www.torproject.org/`
- An anonymous network built on top of the non-anonymous Internet
- Designed to support a wide variety of anonymity use cases

## Low-latency TCP applications

- Tor works by proxying TCP streams
  - (And DNS lookups)
- Focuses on achieving interactive latency
  - WWW, but potentially also chat, SSH, etc.
  - Anonymity tradeoffs compared to remailers

## Tor Onion routing

- Stream from sender to $D$ forwarded via $A$, $B$, and $C$
  - One Tor circuit made of four TCP hops
- Encrypt packets (512-byte "cells") as $E_A(B, E_B(C, E_C(D, P)))$
- TLS-like hybrid encryption with "telescoping" path setup

## Client perspective

- Install Tor client running in background
- Configure browser to use Tor as proxy
  - Or complete Tor+Proxy+Browser bundle
- Browse web as normal, but a lot slower
  - Also, sometimes `google.com` is in Swedish

## Anonymity loves company

- Diverse user pool needed for anonymity to be meaningful
  - Hypothetical Department of Defense Anonymity Network
- Tor aims to be helpful to a broad range of (sympathetic sounding) potential users

## Anti-censorship

- As a web proxy, Tor is useful for getting around blocking
- Unless Tor itself is blocked, as it often is
- *Bridges* are special less-public entry points
- Also, protocol obfuscation arms race (currently behind)

## Hidden services

- Tor can be used by servers as well as clients
- Identified by cryptographic key, use special rendezvous protocol
- Servers often present easier attack surface

## Intersection attacks

- Suppose you use Tor to update a pseudonymous blog, reveal you live in Minneapolis
- Comcast can tell who in the city was sending to Tor at the moment you post an entry
  - Anonymity set of 1000 $\rightarrow$ reasonable protection
- But if you keep posting, adversary can keep narrowing down the set

## Exit sniffing

- Easy mistake to make: log in to an HTTP web site over Tor
- A malicious exit node could now steal your password
- Another reason to always use HTTPS for logins

## Browser bundle JS attack

- Tor's Browser Bundle disables many features try to stop tracking
- But, JavaScript defaults to on
  - Usability for non-expert users
  - Fingerprinting via NoScript settings
- Was incompatible with Firefox auto-updating
- Many Tor users de-anonymized in August'13 by JS vulnerability patched in June'13

## Next time

- Group project presentations