6.824 2020 Lecture 20: Blockstack

why are we looking at Blockstack?
  it touches on three questions that interest me:
    how to build a naming system -- a PKI -- a critical missing piece
    a non-crypto-currency use of a blockchain
    a very different architecture for web sites, might someday be better
  Blockstack is a real system, developed by a company
  it does have some users and some apps written for it
  but view it as an exploration of how things could be different and better
    not as "this is definitely how things should be"

what's a decentralized app?
  apps built in a way that moves ownership of data into users's hands
    and out of centrally-controlled web sites
  there are many recent (and older) explorations of this general vision.
  the success (and properties) of Bitcoin has prompted a lot of recent activity

old: a typical (centralized) web site
  [user browsers, net, site's web servers w/ app code, site's DB]
  users' data hidden behind proprietary app code
  e.g. blog posts, gmail, piazza, reddit comments, photo sharing,
    calendar, medical records, &c
  this arrangement has been very successful
    it's easy to program
  why is this not ideal?
    users have to use this web site's UI if they want to see their data
    web site sets (and changes!) the rules for who gets access
    web site may snoop, sell information to advertisers
    web site's employees may snoop for personal reasons
    disappointing since it's often the user's own data!
  a design view of the problem:
    the big interface division is between users and app+data
    app+data integration is convenient for web site owner
    but HTML as an interface is UI-oriented.
      and is usually not good about giving users control and access to data

new: decentralized apps
  [user apps, general-purpose cloud storage service, naming/PKI]
  this architecture separates app code from user data
    the big interface division is between user+app and data
    so there's a clearer notion of a user's data, owned/controlled by user
    much as you own the data on your laptop, or in your Athena account
  requirements for the storage system
    in the cloud, so can be accessed from any device
    general-purpose, like a file system
    paid for and controlled by user who owns the data
    sharing between users, modulo permissions, for multi-user apps
    sharing between a user's apps, modulo permissions
    similar to existing services like Amazon S3

what's the point?
  easier for users to switch apps, since data not tied to apps (or web sites)
  easier to have apps that look at multiple kinds of data
    calendar/email, or backup, or file browser
  privacy vs snooping (assuming end-to-end encryption)

how might decentralized applications work?
  here's one simple possibility.
  app: a to-do list shared by two users
    [UI x2, check-box list, "add" button]
  both contribute items to be done
  both can mark an item as finished
  a public storage system,
    key/value data owned by each of U1 and U2
  users U1 and U2 run apps on their computers
    maybe as JavaScript in browsers
    the apps read other user's public data, write own user's data
  the app doesn't have any associated server, it just uses the storage system
  each user creates a file with to-do items
    and a file with "done" marks
  each user's UI code periodically scans the other user's to-do files
  the point:
    the service is storage, independent of any application.
    so users can switch apps, write their own, add encryption to
    prevent snooping, delete their to-do lists, back them up,
    integrate with e-mail app, &c

what could go wrong?
  decentralization is painful:
    per-user FS-like storage much less flexible than dedicated SQL DB
    no trusted server to e.g. look at auction bids w/o revealing
    cryptographic privacy/authentication makes everything else harder
    awkward for users as well as programmers
  current web site architecture works very well
    easy to program
    central control over software+data makes changes (and debugging) easy
    good solutions for performance, reliability
    easy to impose application-specific security
    successful revenue model (ads)

now for Blockstack

why does Blockstack focus on naming?
  names correspond to human users, e.g. "robertmorris"
  name -> location (in Gaia) of user's data, so multiple users can interact
  name -> public key, for end-to-end data security
    so I can check I've really retrieved your authentic data
    so I can encrypt my data so only you can decrypt it
    since storage system is not trusted
  lack of a good global PKI has been damaging to many otherwise good security ideas
    so Blockstack started with names

Blockstack claims naming is hard, summarized by "Zooko's triangle":
  1. unique (global) i.e. each name has the same meaning to everyone
  2. human-readable
  3. decentralized
  claim: all three would be valuable (debatable...)
  claim: any two is easy; all three is hard

example for each pair of properties?
  unique + human-readable : e-mail addresses
  unique + decentralized : randomly chosen public keys
  human-readable + decentralized : my contact list

```
why is all three hard?
  can we add the missing property to any of our three schemes?
  no, all seem to be immediate dead ends

summary of how Blockstack gets all three?
  Bitcoin produces an ordered chain of blocks
  Blockstack embeds name-claiming records in Bitcoin blocks
  if my record claiming "rtm" is first in Bitcoin chain, I own it
  unique (== globally the same)?
  human-readable?
  decentralized?

is this kind of name space good for decentralized apps?
  is unique (== global) valuable?
    yes: I may be able to remember names I already know.
    yes: I can give you a name, and you can use it.
    yes: I can look at an ACL and guess what it means.
    no: human-readable names aren't likely to be very meaningful if chosen from global pool
        e.g. robert_morris_1779 -- is that me? or someone else?
        how about "rtm@mit.edu"?
    no: how can I find your Blockname name?
        how can I verify that a Blockstack name is really you?
  other (possibly bad) ideas:
    only public keys, don't bother with human-readable names
      each person keeps separate "contact list" with names they understand
      naturally decentralized
      not "unique" thus no need for Bitcoin
    central entity that reliably verifies human identity

what are all the pieces in Blockstack?
  client, browser, application, blockstack.js
  Blockstack Browser (meant to run on client machine)
  Bitcoin's block-chain
  Blockstack servers
    read Bitcoin chain
    interpret Blockstack naming records to update DB
    serve naming RPCs from clients
    name -> pub key + zone hash
  Atlas servers -- store "zone records"
    a name record in bitcoin maps to a zone record in Atlas
    zone record indicates where my Gaia data is stored
    keyed by content-hash, so items are immutable
    you can view Atlas as just reducing the size of Blockstack's Bitcoin transactions
    Atlas keeps the full DB in every server
  Gaia servers
    separate storage area for each user (i.e. end-users)
    key -> value
    backed by Amazon S3, Dropbox, &c
      Gaia makes them all look the same
    most users use Gaia storage provided by Blockstack
    user's profile contains user's public key, per-app public keys
    user can have lots of other files, containing app data
    apps can sign and/or encrypt data in Gaia
  S3, Dropbox, &c
    back-ends for Gaia

NAME CREATION
```

how does one register a Blockstack name?
  (https://docs.blockstack.org/core/wire-format.html)
  the user does it (by running Blockstack software)
  user must own some bitcoin
  two bitcoin transactions: preorder, registration
  preorder transaction
    registration fee to "burn" address
    hash(name)
  registration transaction
    name (not hashed)
    owner public key
    hash(zonefile)
  Blockstack info hidden inside the transactions, Bitcoin doesn't look at it
    but Bitcoin signatures/hashes cover this Blockstack info

why *two* transactions?
  front-running

why the registration fee? after all there's no real cost.

what if a client tries to register a name that's already taken?

what if two clients try to register same name at same time?

is it possible for an attacker to change a name->key binding?
  after all, anyone can submit any bitcoin transaction they like

is it possible for Blockstack to change a name->key binding?

STORAGE

how does the client know where to fetch data from?
  starting with owning user's name, and a key
  apps probably use well known keys, e.g. "profile" or "todo-list"
  bitcoin/blockstack, hash(zone), gaia address

how does the client check that it got the right data back from Gaia?

how does the client know data from Gaia is fresh (the latest version)?
  owner signed the data when writing
  where can others get the owner's public key, to check signature?

how does Gaia know whether to let a client write/change/delete?

what about encryption for privacy?
  if only the owner should see the data?
  if one other user should see the data, in addition to the owner?
  if just 6.824 students should see the data?

PRIVATE KEYS

never leaves user's device(s)
  so you don't have to trust anything other than your device and Blockstack's software
  each of your devices has a copy of your master private key

"master" private key only seen by Blockstack Browser
  too sensitive to let apps see or use it

  protected by pass-phrase, then in clear while user is active

Blockstack Browser hands out per-app private keys
  so each app has more or less separate encrypted storage
  makes it hard for one user's different apps to cooperate
    sometimes that's what you want
    sometimes you do want sharing among your own apps

DISCUSSION

here are some questions to chew on.
  about naming
  about decentralized applications
you can view them as criticism.
or as areas for further development.

Q: could blockstack be used as a PKI for e-mail, to map rtm@mit.edu to my public key?
   blockstack names vs e-mail addresses?
   what does a blockstack name mean?

Q: why is PKI hard in general?
   lost pass-phrases and keys
     recovery (mother's maiden name? SMS? e-mail?)
   what does a name mean? connection to "real" identity?
   how to go from intuitive notion of who I want to talk to, to name?
   some progress, e.g. Keybase

Q: for naming and PKI, is there strong value in decentralization?
   can we have a centralized but secure naming system?
     who can we all trust for a global-scale system?
   indeed what value can a central authority realistically deliver?
   would adoption be easier with decentralization?

Q: could blockstack use a scheme like Certificate Transparency instead of Bitcoin?
   CT can't resolve conflicts, only reveal them.
     different CT logs may have different order
       so CT can't say which came first
     it's Bitcoin mining that resolves forks and forces agreement
   the fee aspect of Blockstack seems critical vs spam &c, relies on cryptocurrency
   in general, open block-chains only seem to make sense w/ cryptocurrency

Q: is Blockstack convenient for programmers?
   all code in client, no special servers
     hard to have data that's specific to the app, vs each user
     indices, vote counts, front-page rankings for Reddit or Hacker News
   SQL queries
   cryptographic access control, groups, revocation, &c
   hard to both look at other users' secrets, and keep the secrets
     e.g. for eBay
   maybe only worthwhile if users are enthusiastic...

Q: is decentralized user-owned storage good for user privacy?
   is it better than trusting Facebook/Google/&c web sites to keep data private?
     vs other users, hackers, their own employees?
   can Blockstack storage providers watch what you access?
   what if app, on your computer, snoops on you?
     after all, it's presumably still Facebook or whoever writing the app.

```
  is cryptographic access control really feasible?
  you still have to trust the provider to preserve your data
    and to serve up the most recent version
    if you trust them that much, why not trust them to keep it secret too?

Q: is decentralized user-owned storage good for user control?
  do users want to switch applications a lot for the same data?
  do users want to use same data in multiple applications?
  does either even work in general, given different app formats?

Q: will users be willing to pay for their own Gaia storage?

CONCLUSION

what do I take away from Blockstack?
  I find the overall decentralization vision attractive.
  the whole thing rests on a PKI -- any progress here would be great
    a general-purpose mapping from all users to their public keys would be very useful
  surprising that we can have decentralized human-readable name allocation
    but unclear whether decentralized human-readable names are a good idea
  separating cloud data from applications sounds like a good idea
    but developers will hate it (e.g. no SQL).
    not clear users will know or care.
    not clear whether users will want to pay for storage.
  end-to-end encryption for privacy sound like a good idea
    private key management is a pain, and fragile
    encryption makes sharing and access control very awkward
  you still have to trust vendor software; not clear it's a
    huge win that it's running on your laptop rather than
    vendor's server.

all that said, it would be fantastic if Blockstack or something
  like it were to be successful.
```