Algorand As "*The*" Blockchain Protocol

Byzantium Agreement

- Consensus in a decentralized network environment
  Potentially faulty setting, uncontrolled – no central authority/intermediary units, open to powerful adversary, time-constraint
- Honesty of 2/3rd+ of the votes for 1/2+ accuracy of the result
  Theoretical bound of Byzantium Agreement
- Fast – much faster than traditional BA systems
- Binary consensus
  - A True|False outcome
    "*Yes(/No) this block is(/not) valid*"
  - *Which block*
    - The one proposed by round-leader
    - Consistency – what to deliberate on
  - Reached on proposed block **whenever** 2/3+ of votes are in honest hands

Rounds

Round – Logical unit of process
Algorand – Sequence of rounds

- Block approval process
  - i) Propose a block
  - ii) Deliberate on it
  - iii) Reach consensus when/if 2/3+ of verification committee agree
- Byzantium agreement – 4-9 steps
- Result:
  - Either a certified block of approved transactions or an empty block
  - Empty block = consensus can't be reached
  - A third possibility – approval of a fake block?
    - i.e. forks/double spending
    - Negligible
- Algorand keeps executing rounds one after the other
  - Each with round# incremented
    round-0 (the genesis round), *round*-1, *round*-2, ...
  - 1 round – 1 block
    Into the immutable ledger and in that order

Role Players

Selected by cryptographic sortition

- Leader – **propose a block (i)**
  - One leader for each round – the first step
  - Sets "what to deliberate on" for "this" round
    - The block – set of transactions to be approved
    - Consistency for binary consensus
    - Dishonest leader sending out multiple blocks to propose
      - No sufficient majority for YES|NO
        No consensus till timeout
      - Worse than one with invalid content
        Gets disapproved right off
- Verifiers – **deliberation and consensus (ii) & (iii)**
  - Vote to dis/approve the block proposed by leader
  - Second step and after
  - One *Set of Verifiers* (SV) for each step
    - 1500 verifiers/step in one tested deployment [CM17]

How Long Does It Take To Approve A Block?

i.e., ***How long does a round take***?

- 4-9 steps [M17]:
  - 1st step: block proposal by round-leader
  - Latter steps: (dis/)approval of the proposed block by SVs
- Worst case – 9 steps

  dishonest round-leader
  - Propose several blocks
  - SVs can't see a consistent block to agree on
  - "Exit" without an agreement – empty-block
- Likeliest case – 4 steps
  - A single block proposed
  - SV agrees right away
- Bound by network propagation delay

  No other time-boundary – process time negligible

How Long To Approve A Block – Time Parameters

2 parameters:

Λ – round setup

λ – step duration

- Upper limits – large enough to cover propagation delay
- Synchronize steps – decentralized environment
- Start timing for round *r*

  .. as soon as you see the block produced by round *r*-1
- $\Lambda \sim$ *Candidate block formed and circulated in the network*
  - Full block content – maximal payset

    large messages travel slow
- $\lambda \sim$ *Messages circulated between steps*
  - Wait for $\Lambda + (2i\text{-}1)\lambda$ to execute step *i*
  - Less than Λ – smaller messages during consensus

So – How Long!?

Round duration = $\Lambda + (2k\text{-}1)\lambda + 2\lambda = \Lambda + (2k+1)\lambda$

k: #steps

$+2\lambda$ for round result be seen

- Suggested values [CM17]

  Λ= 1 minute

  λ = 10 seconds

- Worst case – 9 steps

  4 minutes 10 secs

- Expected case – 4 steps

  2.5 minutes

  "*Will see your xn confirmed within 2.5 minutes*"

  Recall: no forks!

An Underlying Security Fact – Time

- Role players are targets of adversary from the time they are known in the network
  - Corrupt the leader – sub-maximal/illegitimate payset, multiple blocks, manipulate the seed, ...
  - Corrupt the SV members – the judges!
  - ..or, just DoS – make them disfunctional
- Remedy: diminish the time-slice role-players are known in the network
  - Cryptographic sortition – know you're a role player in secrecy
  - Cryptographic sortition – don't speak till necessary
  - Player replaceability – speak only once

Cryptographic Sortition

A role assignment lottery

Raison d'être

2-fold

1) Unbiased assignments
   - Random – nobody is more equal
   - Proportional to stakes – design requirement

2) Secrecy of role players
   - Not unnecessarily exposed

     Out of sight of the adversary
   - Hiding something?
     - Nope. Info no use to anyone till it's relevant

       Except to the adversary!
     - Relevance: "who's certifying it"

       Right on the message!

The lottery

- Winner gets a role in the step
- Ticket#: randomly assembled from your userID, round#, step#
- Result known only to ticket-holder until he comes out

Cryptographic Sortition – The Process

ticket# = **.H(*sign*(*r*,*s*,Q_r))**

where

r: round#

s: step#

Q_r: random seed // known from the block of previous round

Compose the ticket#:

- Compose message to be signed: $m = (r,s,Q_r)$
- Sign *m* using your static key: $sign(r,s,Q_r)$
- Hash it: $H(sign(r,s,Q_r))$
- Map to a fractional value *f* in [0,1]: $f = .H(sign(r,s,Q_r))$
  .. so that $f<p$ with probability *p*, i.e. no bias!

See whether you won:

- Compare with *p*

You are a role player in round/step (*r*,*s*) iff **.H(*sign*(*r*,*s*,Q_r)) < *p***

Cryptographic Sortition – The Stakes

Modify parameters

A suggested implementation*:

- Compute $K' = \frac{n * a_{r-k,i}}{A_{r-k}}$, $K = \lfloor K' \rfloor$

  where

  A_{r-k} : total coins in the system k rounds before

  $a_{r-k,i}$: user i's coins in the system k rounds before

  $n = |SV|$: targeted number of verifiers

- Add K into ticket#: .H($sign((i, K+1), r, s, Q_r)$)

User i has K votes in (r,s)

+ 1 vote if .H($sign((i, K+1), r, s, Q_r)$) < K'-K

Average in target domain: $A_{AVG} = \frac{A_{r-k}}{n}$

One vote for each A_{AVG} + the partial chances

*"*A More Complex Implementation*", Section 8, [CM17]

Cryptographic Sortition – Properties

Specific to (*round, step, user*)

- Can't be cheated – role player
  - One ticket per player
  - Random ticket#
- Can't be manipulated – the adversary
  - Can't influence "which ticket"
  - Q_r – not known till previous round is concluded
- Private
  - No reliance to an external process
  - No one knows until the user tells
  - Too late by the time the user tells
- Easily verifiable – role credentials
- Negligible overhead – computational and message

Player Replaceability

Speak only once

- Send out single message
  - Cast your vote
  - Role credentials
  - No other step duty
- Another set of verifiers the next step

  Cryptographic sortition tells "who"
- Role players do not retain any state info necessary/useful for the steps after

  They know what all else know
  - Replaceability
  - Never a useful target to adversary
- Additional overhead?
  - SV members know themselves already – no time-delay
  - Round-leader selected as candidate leader messages propagate
  - Negligible process overhead