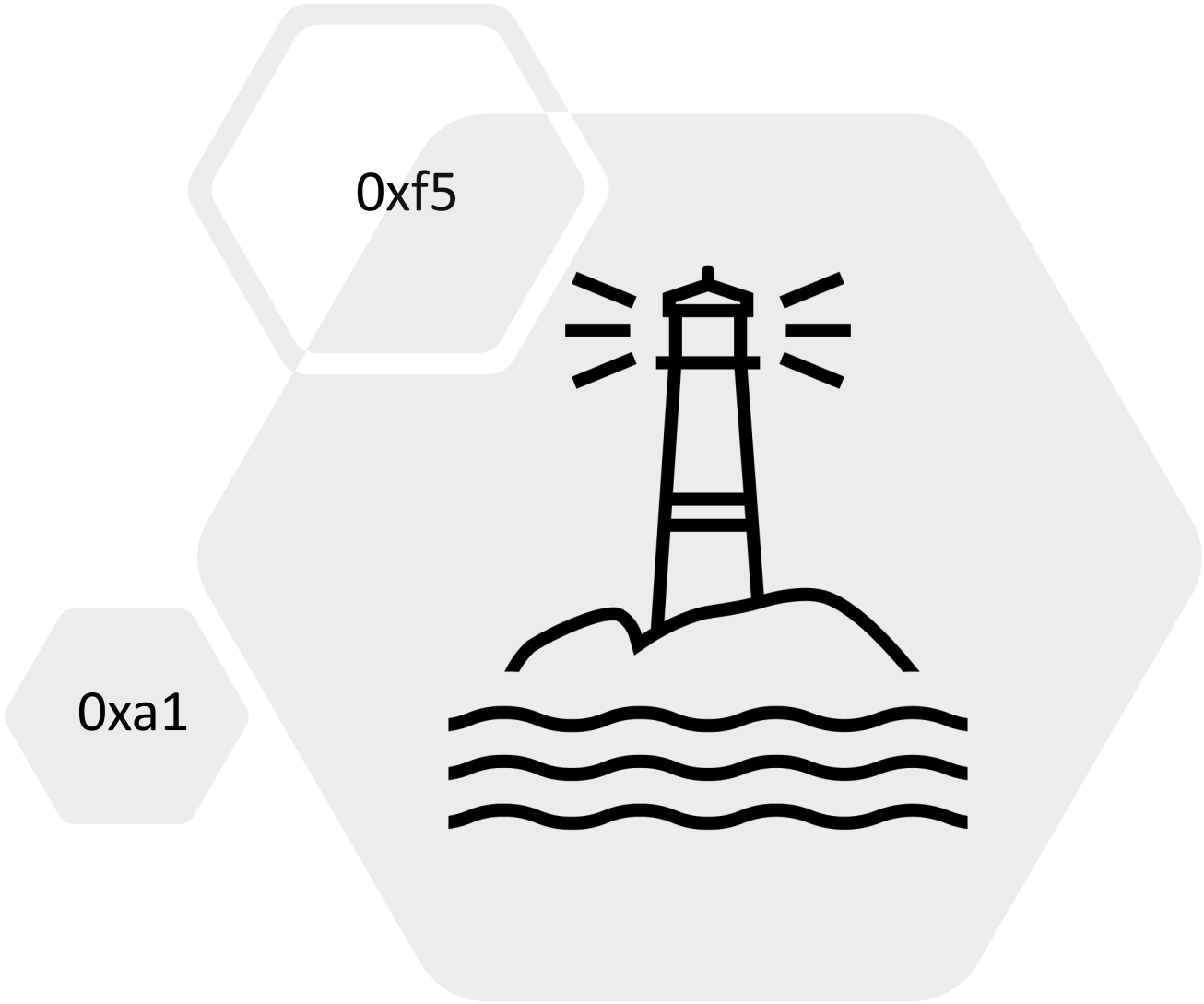
Random Beacon Protocols



Adithya Bhat
PURDUE UNIVERSITY

Tuesday, November 16, 2021

Protocols for Randomness

- A coin-toss protocol is a protocol between nodes to generate a random number (usually a single bit)
- A random beacon protocol is a protocol between nodes to generate a sequence of random numbers
- Both protocols have a requirement that the generated outputs are random, i.e., given a uniformly random string and the outputs, no polynomial time algorithm can distinguish the outputs

Introduction

- A coin-toss is ***like*** Byzantine Agreement, a single shot protocol
- A random beacon protocol is ***like*** SMR, iterative and can be pipelined

Motivation – Coin Toss

- A coin-toss protocol can be used to agree on something
- Examples:
  - who does the dishes in a house
  - who goes first in a contest
  - in sports
  - to break ties

Motivation – Random Beacon Protocols

- A random beacon protocol is used when continuous service is required
- Examples:
  - Proof-of-Stake systems
  - Cryptographic and SMR protocols
  - Lotteries and Casinos

Motivation

Coin Toss or Random Beacon?

Beacon Protocol as a Coin-Toss

Satoshi Grace Period

Many security conferences, including this one in the past, have claimed a firm deadline only to extend it by several days as the deadline approached. Keeping with the tradition started in FC19, we will implement a randomized deadline in a verifiable way.

All papers must be registered by Tue Sep 2, 2021. This means the titles, authors, abstracts, topics, submission options, conflicts, etc. (everything except the final PDF of the paper) must be entered into the submission system by this date. **This date is firm and will not be extended.**

On September 3, 2021, we will announce (in this space) a block height on the Bitcoin blockchain that we expect to be found the following day.

The selected block height is **698980**.

Once the block of that height is found and confirmed, let the *last hex digit* of the hash of that block be *L*. Then the FC22 paper submission deadline will be September (9 + ceil(sqrt(*L*))), 2021. In table form:

L	Paper submission deadline
0	Sep 9, 2021
1	Sep 10, 2021
2, 3, 4	Sep 11, 2021
5, 6, 7, 8, 9	Sep 12, 2021
A, B, C, D, E, F	Sep 13, 2021

When the paper submission deadline has been determined in this way, this page will be updated, and that deadline will be **firm**. The program chairs' interpretation of the above algorithm is final.

The FC22 **firm** submission deadline is **Mon Sep 13, 2021**.

Coin Tossing

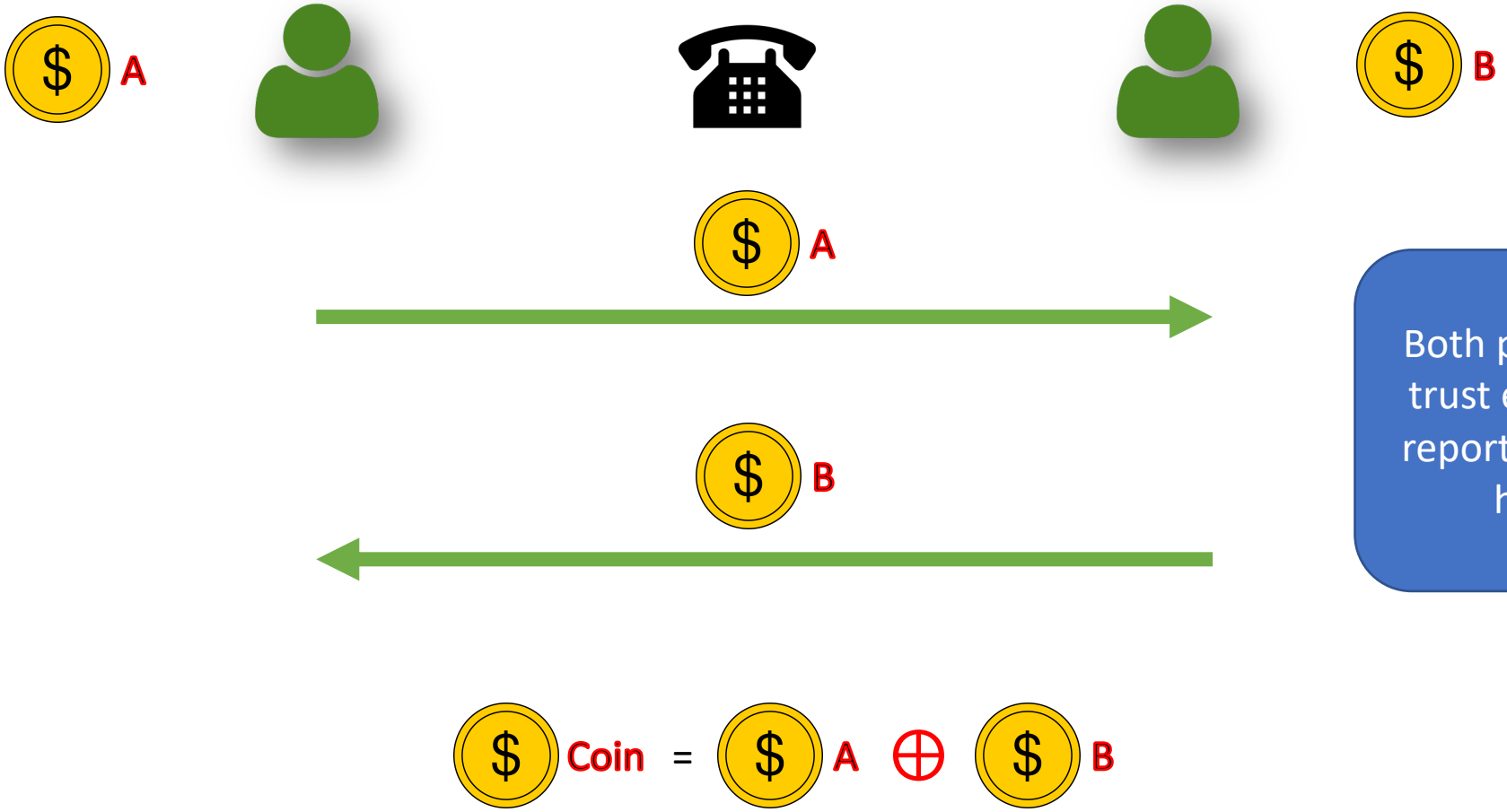
Is coin-tossing a meaningful problem when only two-parties are involved?

✓ **Yes**

✓ **Does a 2P-coin toss protocol exist?**

✓ **Yes**

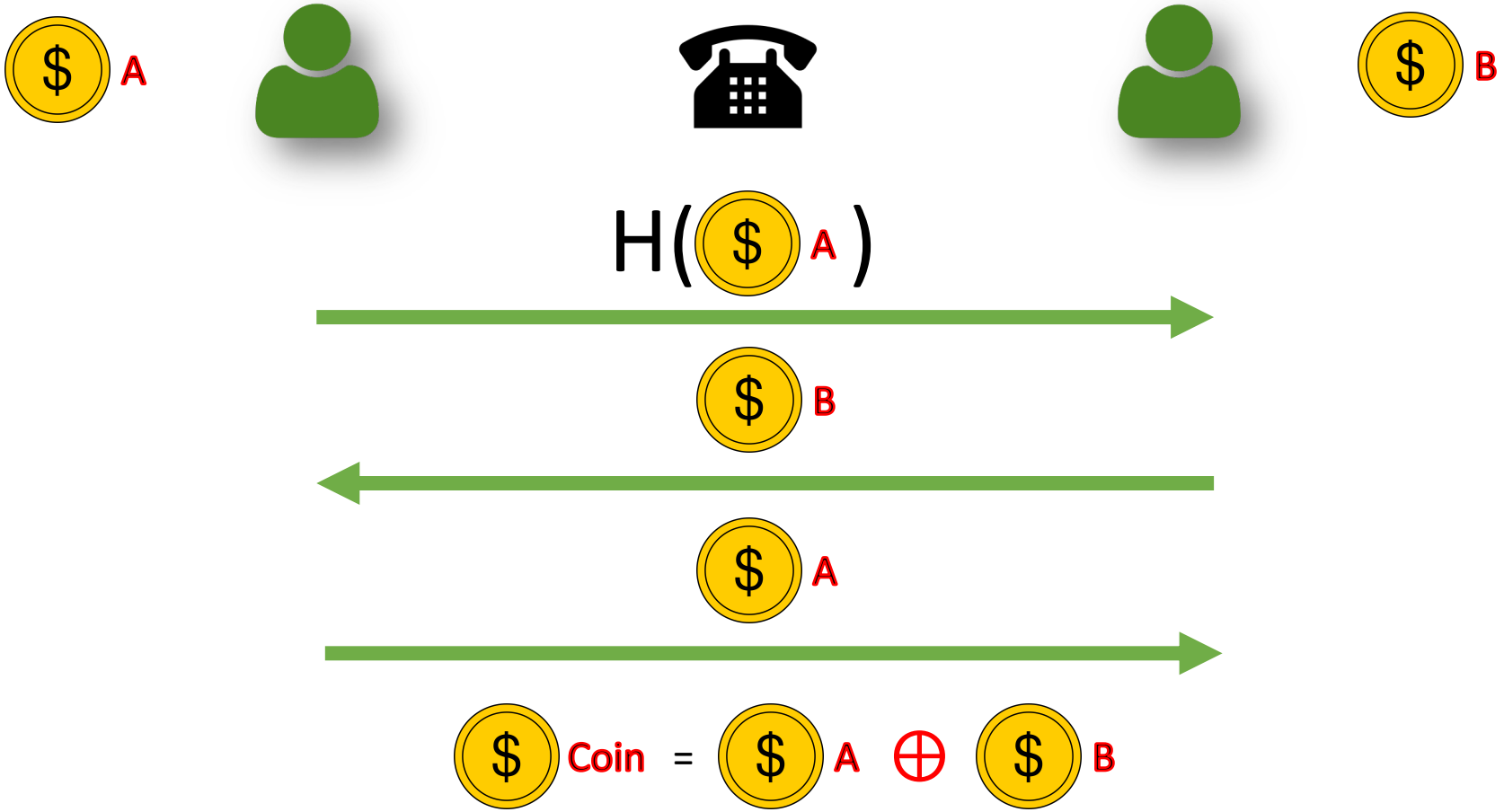
Two Party Coin Tossing[1]



Both parties do not trust each other to report the coin toss honestly.

[1]Blum, Manuel. "Coin flipping by telephone a protocol for solving impossible problems." *ACM SIGACT News* 15.1 (1983): 23-27.

Two Party Coin Tossing[1]



[1]Blum, Manuel. "Coin flipping by telephone a protocol for solving impossible problems." *ACM SIGACT News* 15.1 (1983): 23-27.

Two party Coin Tossing Protocol

- Two party protocols are in general of significant interest in cryptography
- 2P protocols (like Diffie-Hellman Key Exchange)
  - require minimum assumptions (no need of any synchrony or trust assumptions)
  - One of the parties can always be substituted with an SMR which is equivalent* to an honest party
  - Simpler to analyze

Random Beacon Protocols

- Unlike a single shot coin-tossing protocol, random beacon protocols can re-use previous instances
- Two key properties:
  - Unpredictability: Before a round, an adversary cannot know the beacon value
  - Bias-resistance: An adversary must not be able to influence the beacon values, i.e., the set of beacon values must be indistinguishable from a uniform distribution

Random Beacon Protocols

- Two simple protocols:
  - Random Oracles in Constantinople
  - Drand (a variant of random oracles in Constantinople)

Random Oracles in Constantinople[2] / Drand

Public Parameters: $g \in \mathbb{G}, H: \{0,1\}^* \to \mathbb{G}$

$$p(x) = c_0 + c_1 x + \ \ldots + c_t x^t$$

$$sk_i \ := p(i), pk_i \ := g^{sk_i}, sk = c_0 = p(0)$$

- Coin Toss for round i
  - Send $H(i)^{sk_i}$ to all the nodes
  - Reconstruct $H(i)^{sk}$ using $t+1$ $H(i)^{sk_i}$ values
  - The beacon for round i is $H(i)^{sk}$

How to verify if the received $H(i)^{sk_i}$ is valid?

[2]Cachin, Christian, Klaus Kursawe, and Victor Shoup. "Random oracles in Constantinople: Practical asynchronous Byzantine agreement using cryptography." *Journal of Cryptology* 18, no. 3 (2005): 219-246.

Random Oracles in Constantinople[2] / Drand

Public Parameters: $g \in \mathbb{G}, H: \{0,1\}^* \rightarrow \mathbb{G}$

$$p(x) = c_0 + c_1 x + \ \ldots + c_t x^t$$

$$sk_i \ := p(i), pk_i \ := g^{sk_i}, sk = c_0 = p(0)$$

Prove that $\boldsymbol{H(i)^{sk_i}}$ and $\boldsymbol{pk_i = \ g^{sk_i}}$ have the same exponent

- Two techniques:
  - Pairings
  - Zero Knowledge proofs (in particular, proofs of discrete log equality)

[2]Cachin, Christian, Klaus Kursawe, and Victor Shoup. "Random oracles in Constantinople: Practical asynchronous Byzantine agreement using cryptography." *Journal of Cryptology* 18, no. 3 (2005): 219-246.

Random Oracles in Constantinople[2] / Drand

Public Parameters: $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2, H: \{0,1\}^* \to \mathbb{G}_1$

$$p(x) = c_0 + c_1 x + \ \ldots + c_t x^t$$

$$sk_i \ := p(i), pk_i \ := g_2^{sk_i}, sk = c_0 = p(0)$$

Prove that $\boldsymbol{H(i)^{sk_i}}$ and $\boldsymbol{pk_i = \ g_2^{sk_i}}$ have the same exponent

- Check $e'\left(H(i)^{sk_i}, g_2\right) = e'(H(i), pk_i) = e(H(i), g_2)^{sk_i}$

[2]Cachin, Christian, Klaus Kursawe, and Victor Shoup. "Random oracles in Constantinople: Practical asynchronous Byzantine agreement using cryptography." *Journal of Cryptology* 18, no. 3 (2005): 219-246.

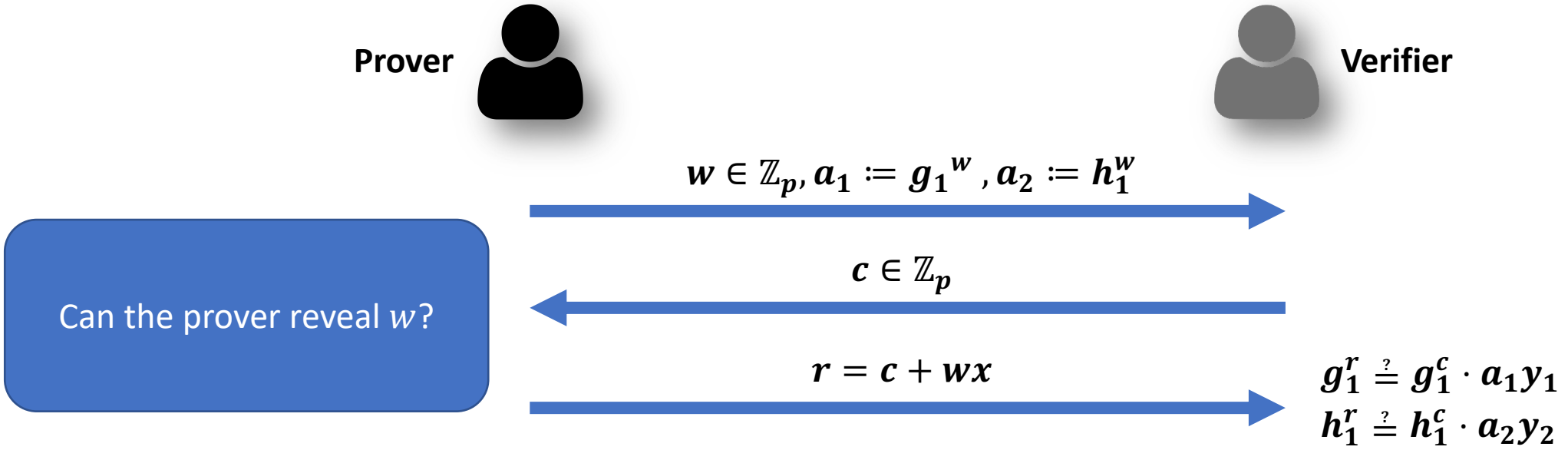
Discrete Log Proof of Equality[3]

Prove that $g_1^{x_1}, y_2 = h_1^{x_2}$ have the same exponent, i.e., $x_1 = x_2$

Public Parameters: $g_1, h_1 \in \mathbb{G}$

Verifier Information: $y_1, y_2 \in \mathbb{G}$

Prover Information (Witness): $x \in \mathbb{Z}_p$



[3]Chaum, David, and Torben Pryds Pedersen. "Wallet databases with observers." In *Annual international cryptology conference*, pp. 89-105. Springer, Berlin, Heidelberg, 1992.

Discrete Log Proof of Equality[3]

What happens if the prover picks c?

Public Parameters: $g_1, h_1 \in \mathbb{G}$

Verifier Information: $y_1, y_2 \in \mathbb{G}$

Prover Information (Witness): $x \in \mathbb{Z}_p$

Prover

Verifier

Prover can pick random r, c and set $a_1 = \frac{g_1^r}{g_1^c \cdot y_1}$, $a_2 = \frac{h_1^r}{h_1^c \cdot y_2}$

Now, , a_1 and a_2 need not have the same exponent, and thus y_1 and y_2 need not have the same exponent

$\boldsymbol{w \in \mathbb{Z}_p, a_1 := g_1^w, a_2 := h_1^w}$

$\boldsymbol{c \in \mathbb{Z}_p}$

$\boldsymbol{r = c + wx}$

$\boldsymbol{g_1^r \stackrel{?}{=} g_1^c \cdot a_1 y_1}$

$\boldsymbol{h_1^r \stackrel{?}{=} h_1^c \cdot a_2 y_2}$

[3]Chaum, David, and Torben Pryds Pedersen. "Wallet databases with observers." In *Annual international cryptology conference*, pp. 89-105. Springer, Berlin, Heidelberg, 1992.

Fiat Shamir Heuristic[4]

- The problem in the previous interactive version of the protocol is that c must be chosen ***after*** seeing a_1, a_2 by the verifier
- FS Heuristic: Set $c = H(a_1, a_2)$
- Now the adversary cannot arbitrarily choose a_1, a_2

[4]Fiat, Amos, and Adi Shamir. "How to prove yourself: Practical solutions to identification and signature problems." In *Conference on the theory and application of cryptographic techniques*, pp. 186-194. Springer, Berlin, Heidelberg, 1986.

Non-interactive Discrete Log Proof of Equality

Prove that $g_1^{x_1}, y_2 = h_1^{x_2}$ have the same exponent, i.e., $x_1 = x_2$

Public Parameters: $g_1, h_1 \in \mathbb{G}$

Verifier Information: $y_1, y_2 \in \mathbb{G}$

Prover Information (Witness): $x \in \mathbb{Z}_p$

Prover

$$w \in \mathbb{Z}_p, a_1 := {g_1}^w, a_2 := h_1^w$$

$$c \leftarrow H(a_1, a_2), r = c + wx$$

$$\pi = (r, a_1, a_2)$$

Verifier (World)

$$c \stackrel{?}{=} H(a_1, a_2)$$

$$g_1^r \stackrel{?}{=} g_1^c \cdot a_1 y_1$$

$$h_1^r \stackrel{?}{=} h_1^c \cdot a_2 y_2$$

Drand[5]

- $H(i)^{sk}$ is also a BLS signature on the message i
- Drand uses technique 1 (pairing)
- Adds an assumption that the signature is unique and unpredictable
  - If it was predictable the signature scheme wouldn't be secure
- Drand assumes unique signatures for BLS, random oracles, and more
- Random Oracles in Constantinople uses technique 2 (NIZK) and only requires the discrete log and random oracle assumption

[5]https://drand.love/

Protocol Analysis

- The communication complexity of both protocols is $O(n^2)$
- The protocols output beacons that are
  - Unpredictable: Any set of t nodes cannot predict the beacon values
  - Bias-resistant: Any set of t nodes cannot change/influence the beacon value

Protocol Analysis

- The setup requires the public keys to be parts of a degree-t polynomial
- Therefore, to change or add a node, the public keys need to be re-generated
- This is known as a DKG protocol which requires $O(n^3)$ communication complexity

State of the art Random Beacon Protocols - Synchronous

Table 1: Comparison of related works on Random Beacon protocols in standard synchrony

Protocol		Res.(t)	Unpred.	Comm. Compl		Adp. Adv.	Re-usable Setup	No DKG?	Assumption
				Best	Worst				
Cachin et al./Drand	[17, 25]	49%	1	$O(\kappa n^2)$	$O(\kappa n^2)$	✗	✗	✗	Threshold Secret/BLS
Dfinity	[4, 30]	49%	$O(\kappa)$	$O(\kappa n^2)$	$O(\kappa n^3)^*$	✗	✗	✗	Threshold BLS
HERB	[20]	33%	1	$O(\kappa n^3)$	$O(\kappa n^3)$	✗	✗	✗	Threshold ElGamal
HydRand	[43]	33%	$O(\min(\kappa, t))^\dagger$	$O(\kappa n^2)$	$O(\kappa n^3)$	✗	✓	✓	PVSS
HydRand (Worst)	[43]	33%	$t+1$	$O(\kappa n^2)$	$O(\kappa n^3)$	✗	✓	✓	PVSS
RandChain	[29]	33%	$O(\kappa)$	$O(\kappa n^2)$	$O(\kappa n^3)$	✓	✗	✗	PoW
RandHerd	[47]	33%	$O(\kappa)$	$O(\kappa c \log n)^\P$	$O(\kappa n^4)$	✗	✗	✗	Threshold Schnorr
RandHound	[47]	33%	1	$O(\kappa c^2 n)^\P$	$O(\kappa c^2 n^2)^\P$	✗	✓	✓	Client based, PVSS
RandRunner	[42]	49%	$t+1$	$O(\kappa n^2)$	$O(\kappa n^2)$	✓	✓	✓	VDF
RandShare	[47]	33%	1	$O(\kappa n^3)$	$O(\kappa n^4)$	✓	✓	✓	VSS
GRandPiper		49%	$O(\min(\kappa, t))^\dagger$	$O(\kappa n^2)$	$O(\kappa n^2)$	✗	✓	✓	PVSS, q-SDH
GRandPiper (Worst)		49%	$t+1$	$O(\kappa n^2)$	$O(\kappa n^2)$	✗	✓	✓	PVSS, q-SDH
BRandPiper		49%	1	$O(\kappa n^2)^\S$	$O(\kappa n^3)$	✓	✓	✓	VSS, q-SDH

Bhat, Adithya, Nibesh Shrestha, Aniket Kate, and Kartik Nayak. "RandPiper-Reconfiguration-Friendly Random Beacons with Quadratic Communication." *IACR Cryptol. ePrint Arch.* 2020 (2020): 1590.

State of the art Random Beacon Protocols – Partially Synchronous

Table I: Comparison of existing randomness beacon protocol.

	Network model	Fault Tolerance	Adaptive Adversary	Liveness / Availability	Unpredictability	Bias-resistance	Communication Cost (total)	Computation Complexity	Public Verification Complexity	Cryptographic Primitives	Setup Assumption
Cachin et al. [60]	async.	1/3	✗	✓	✓	✓	$O(\lambda n^2)$	$O(n)$	$O(1)$	Uniq. th-sig.	DKG
RandHerd [72]*	async.	1/3♣	✗	✓	✓	✓	$O(\lambda c^2 \log n)$♣	$O(c^2 \log n)$	$O(1)$	PVSS+CoSi	DKG
Dfinity [23]	partial sync.	1/3	✗	✓	✓	✓	$O(\lambda n^2)$	$O(n)$	$O(1)$	Uniq. th-sig.	DKG
Drand [2]	sync.	1/2	✗	✓	✓	✓	$O(\lambda n^2)$	$O(n)$	$O(1)$	Uniq. th-sig.	DKG
HERB [29]	sync.	1/3	✗	✓	✓	✓	$O(\lambda n^4)^{\ddagger}$	$O(n)$	$O(n)$	Partial HE	DKG
Algorand [41]	partial sync.	1/3♣	✗	✓	$\Omega(t)$	✗	$O(\lambda cn)$♣	$O(c)$	$O(1)$	VRF	CRS
Proof-of-Work [60]	sync.	1/2	✗	✓	$\Omega(t)$	✗	$O(\lambda n)$	very high	$O(1)$	Hash func.	CRS
Ouroboros [53]	sync.	1/2	✗	✓	✓	✓	$O(\lambda n^4)^{\ddagger}$	$O(n^3)$	$O(n^3)$	PVSS	CRS
Scrape [25]	sync.	1/2	✗	✓	✓	✓	$O(\lambda n^4)^{\ddagger}$	$O(n^2)$	$O(n^2)$	PVSS+Broadcast	CRS
Hydrand [68]	sync.	1/3	✗	✓	$t+1$	✓	$O(\lambda n^2 \log n)$	$O(n)$	$O(n)$	PVSS	CRS
RandRunner [67]	sync.	1/2	✓	✓	$t+1$	✓	$O(\lambda n^2)$	VDF	$O(1)$	VDF	CRS
GRandPiper [13]	sync.	1/2	✗	✓	$t+1$	✓	$O(\lambda n^2)$	$O(n^2)$	$O(n^2)$	PVSS	$q-$SDH
BRandPiper [13]	sync.	1/2	✓	✓	✓	✓	$O(\lambda n^3)$	$O(n^2)$	$O(n^2)$	VSS	$q-$SDH
SPURT	partial sync.	1/3	✗	✓	✓	✓	$O(\lambda n^2)$	$O(n)$	$O(n)$	PVSS+Pairing	CRS

Das, Sourav, Vinith Krishnan, Irene Miriam Isaac, and Ling Ren. "SPURT: Scalable Distributed Randomness Beacon with Transparent Setup." *IACR Cryptol. ePrint Arch.* 2021 (2021): 100.

Conclusion and Open Questions

- Coin-tossing : BA :: Random Beacon : SMR (analogous)
- Can we achieve sub-quadratic random beacon protocols? (Sub-quadratic BA is possible assuming randomness)