RSA VDFs

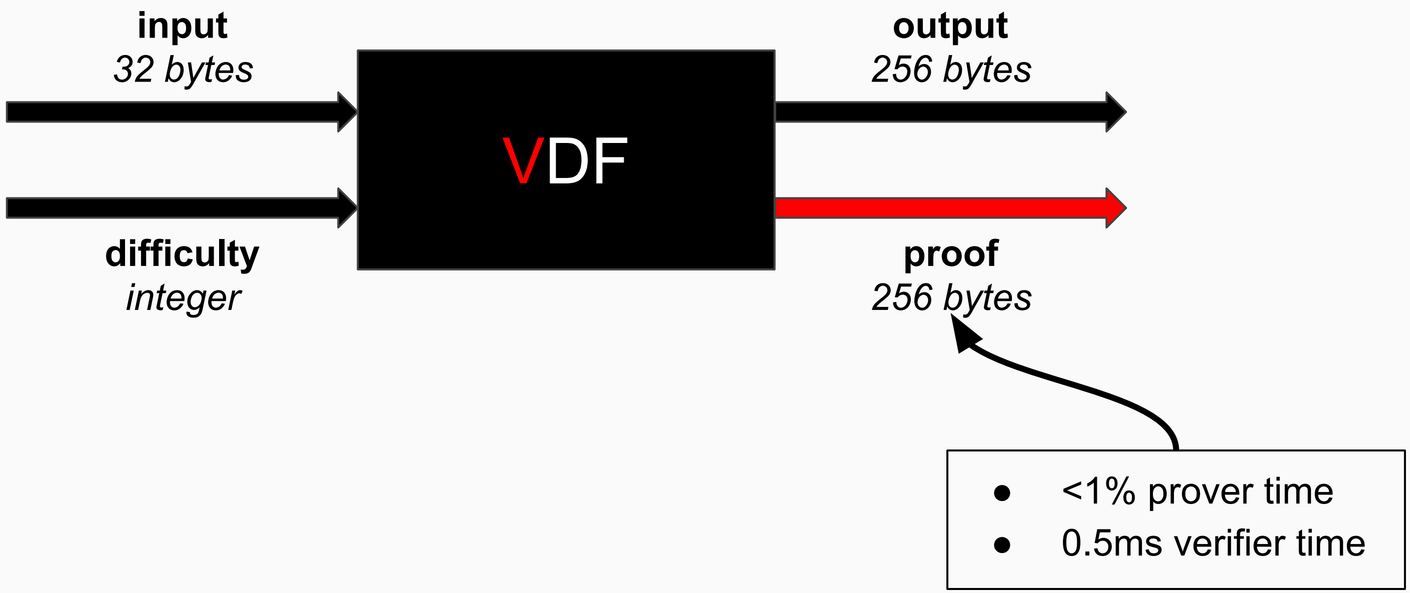
crash course

Verifiable Delay Function (VDF)

Verifiable Delay Function (VDF)

Verifiable Delay Function (VDF)

Verifiable Delay Function (VDF)



Modular exponentiation

SQUARING

```
x -> x^2 % N
```

```
N # unknown factorisation 2048-bit RSA modulus
```

Modular exponentiation

SQUARING

$$x \to x^2 \ \% \ N$$

N # unknown factorisation 2048-bit RSA modulus

T SQUARINGS

$$x \to x^2 \to x^4 \to \ldots \to x^{2^{**}T}$$

x # VDF input

T # time parameter

Modular exponentiation

SQUARING

$$x \rightarrow x^2 \% N$$

`N # unknown factorisation 2048-bit RSA modulus`

T SQUARINGS

$$x \rightarrow x^2 \rightarrow x^4 \rightarrow \dots \rightarrow x^{2**T}$$

`x # VDF input`

`T # time parameter`

OUTPUT

$$y = x^{2**T} \% N$$

`y # VDF output`

Sequentiality assumption

Time-lock puzzles and timed-release Crypto

Ronald L. Rivest*, Adi Shamir**, and David A. Wagner***

Revised March 10, 1996

OUTPUT

$$y = x^{2**T} \% N$$

```
y # VDF output
```

Timelock puzzle

- set by Ron Rivest in 1999
- designed to take 35 years

Timelock puzzle

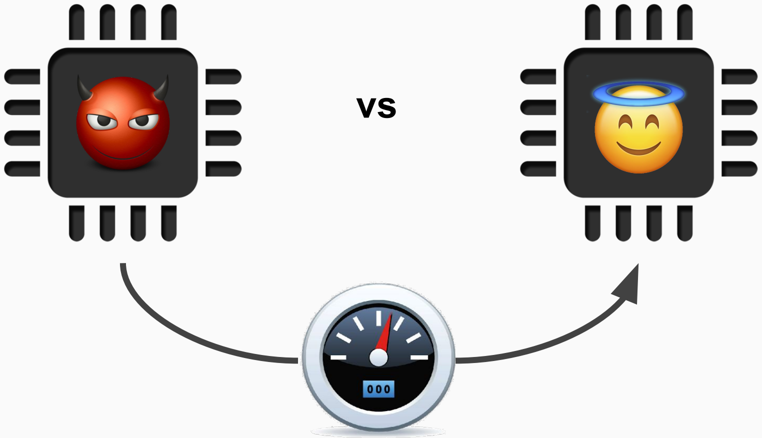
- set by Ron Rivest in 1999
- designed to take 35 years
- solved in 3 months with FPGA
- new low-depth algorithm

Timelock puzzle

- set by Ron Rivest in 1999
- designed to take 35 years
- solved in 3 months with FPGA
- new low-depth algorithm

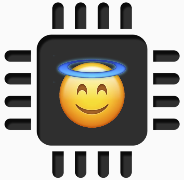
Safety assumption

attacker hardware **commodity hardware**



$$\boxed{\text{speed advantage} \leq A_{max}}$$

Safety assumption

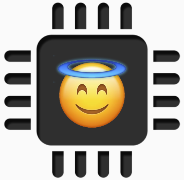
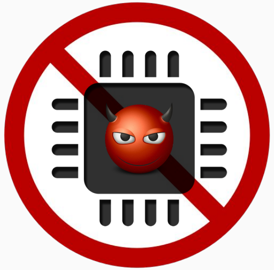
CPU	FPGA	
1 us	**30 ns**	**1 ns**

CØSMOS

ethereum

Filecoin

Safety assumption

CPU	FPGA		
1 us	**30 ns**	**1 ns**	**16 ps**

COSMOS

ethereum

Filecoin

Lower bounds

16 ps

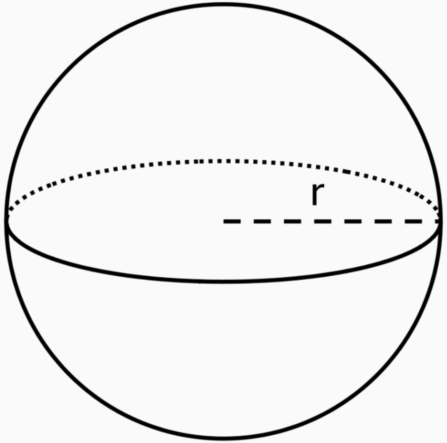
physics

complexity theory

Lower bounds

16 ps

physics



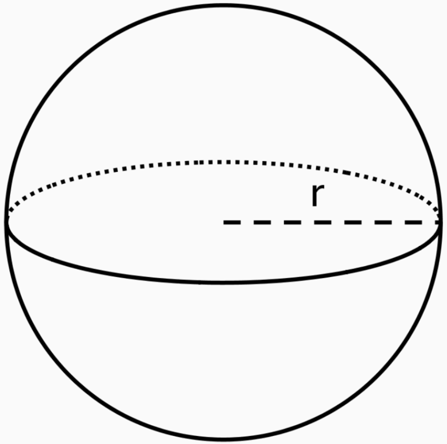
< **4.8mm radius**
per squaring

complexity theory

Lower bounds

16 ps

<u>physics</u>



< **4.8mm radius**
per squaring

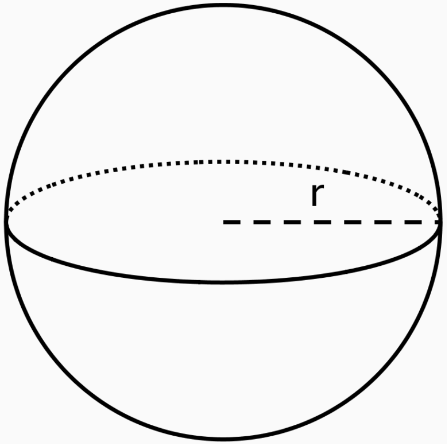
<u>complexity theory</u>

Result: n-bit modular squaring requires log(n) depth in the average case.

Lower bounds

16 ps

physics



< **4.8mm radius**
per squaring

complexity theory

Result: n-bit modular squaring requires log(n) depth in the average case.

Caveats:

- single squaring
- binary representation
- fan-in 2 gates

Applications

Use cases

randomness

ethereum

COSMOS

Tezos

Harmony

Use cases

randomness

ethereum COSMOS Tezos

Harmony

- 100 people, one by one, enter a dark room to reroll a set of dice.

Use cases

randomness

ethereum

COSMOS

Tezos

Harmony

- 100 people, one by one, enter a dark room to reroll a set of dice.
- Lights turn on after the last person, revealing a fair random number.

Use cases

randomness

ethereum

COSMOS

Tezos

Harmony

- 100 people, one by one, enter a dark room to reroll a set of dice.
- Lights turn on after the last person, revealing a fair random number.
- The VDF ensures lights are not turned on early.

Use cases

randomness

proof of space

proof of replication

Use cases

randomness

proof of space

proof of replication

proof of history

anti-frontrunning

Use cases

randomness

ethereum

COSMOS

Tezos

Harmony

proof of space

chia

proof of replication

Filecoin

proof of history

solana

anti-frontrunning

INJECTIVE PROTOCOL

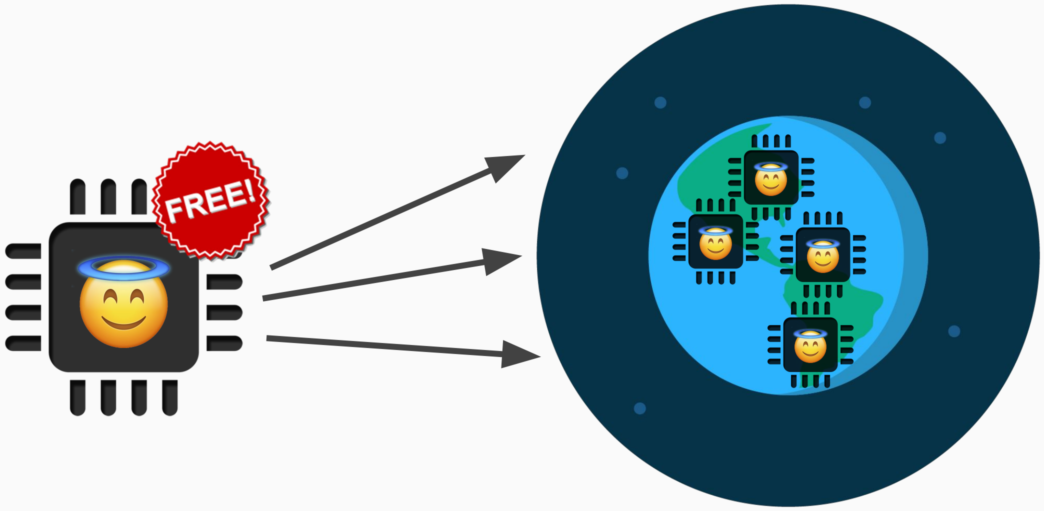
long tail

- objective fork choice
- expiring zk-proofs
- guaranteed output delivery
- timelocks

Liveness assumption

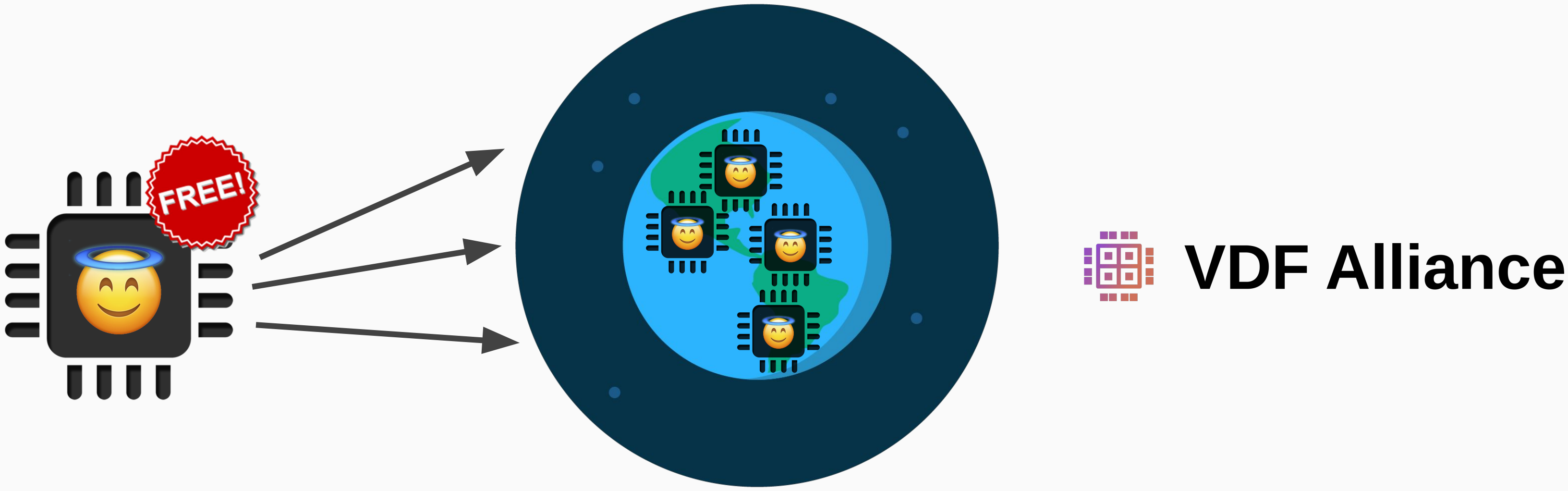
≥1 online VDF rig

Liveness assumption



≥1 online VDF rig

Liveness assumption



≥1 online VDF rig

Provers

Original papers

- “Verifiable Delay Functions”—**Boneh, Bonneau, Bünz, Fisch**
- “Efficient Verifiable Delay Functions”—**Wesolowski**
- “Simple Verifiable Delay Functions”—**Pietrzak**

Original papers

- “Verifiable Delay Functions”—**Boneh, Bonneau, Bünz, Fisch**
- “Efficient Verifiable Delay Functions”—**Wesolowski**
- “Simple Verifiable Delay Functions”—**Pietrzak**

published
June 2018

- “A Survey of Two Verifiable Delay Functions”—**Boneh, Bünz, Fisch**

Wesolowski prover

OUTPUT

$$y = x^{2**T} \% N$$

Wesolowski prover

OUTPUT

```
y = x^(2**T) % N
```

PROOF

```
p = x^(2**T//r) % N
```

```
r # random 128-bit prime (Fiat-Shamir)
```

Wesolowski prover

OUTPUT

$$y = x^{2**T} \% N$$

PROOF

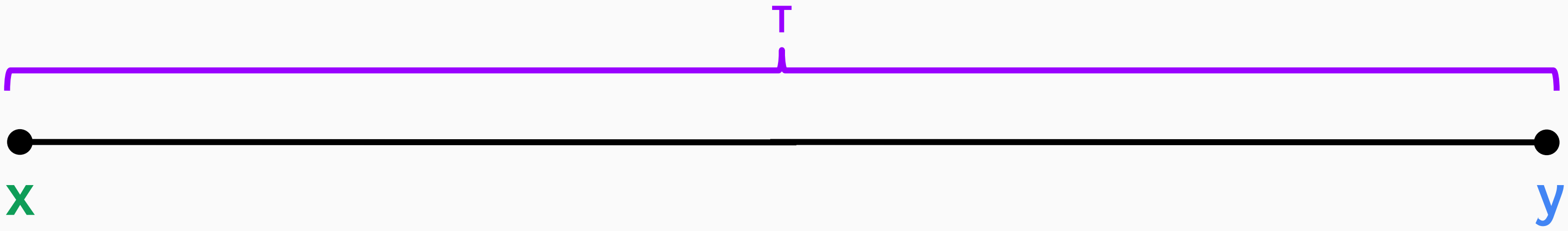
$$p = x^{2**T//r} \% N$$

```
r # random 128-bit prime (Fiat-Shamir)
```

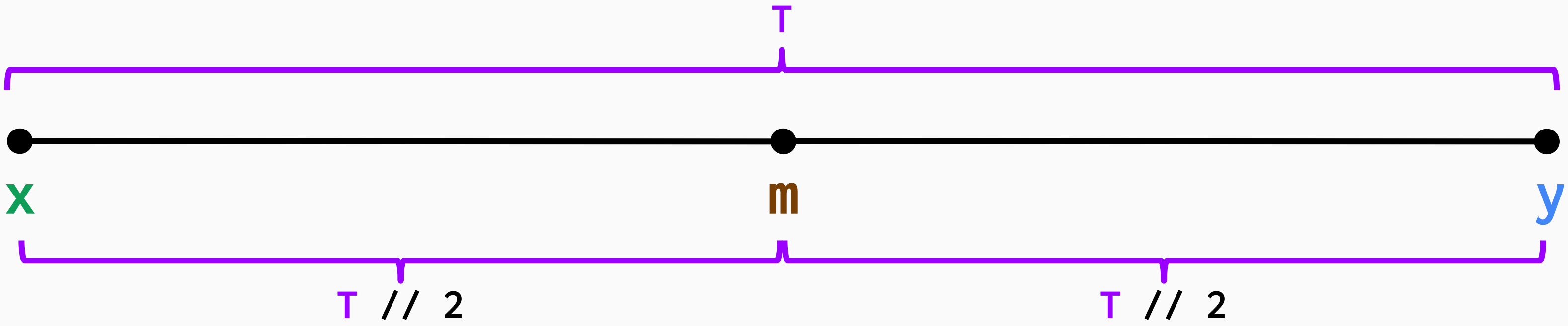
VERIFICATION

$$y == p^{r} * x^{2**T\%r} \% N$$

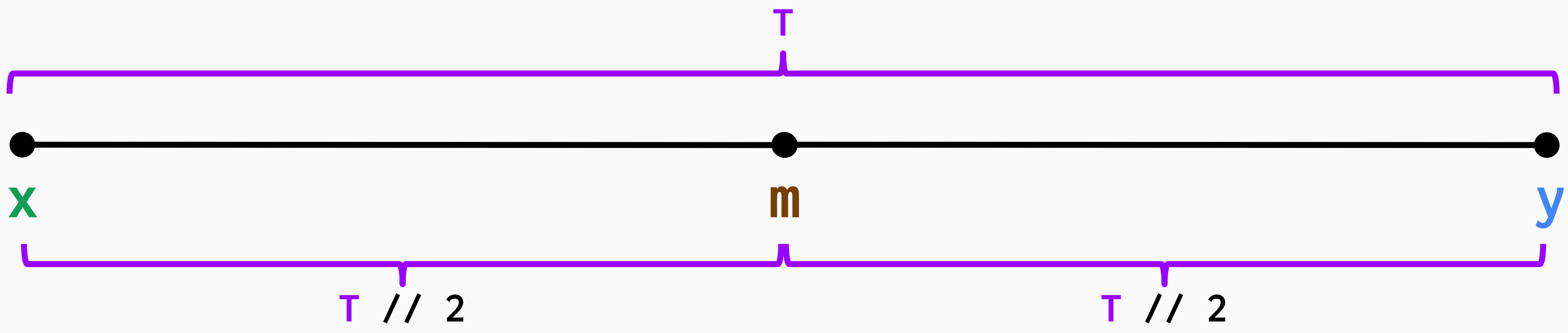
Pietrzak prover



Pietrzak prover

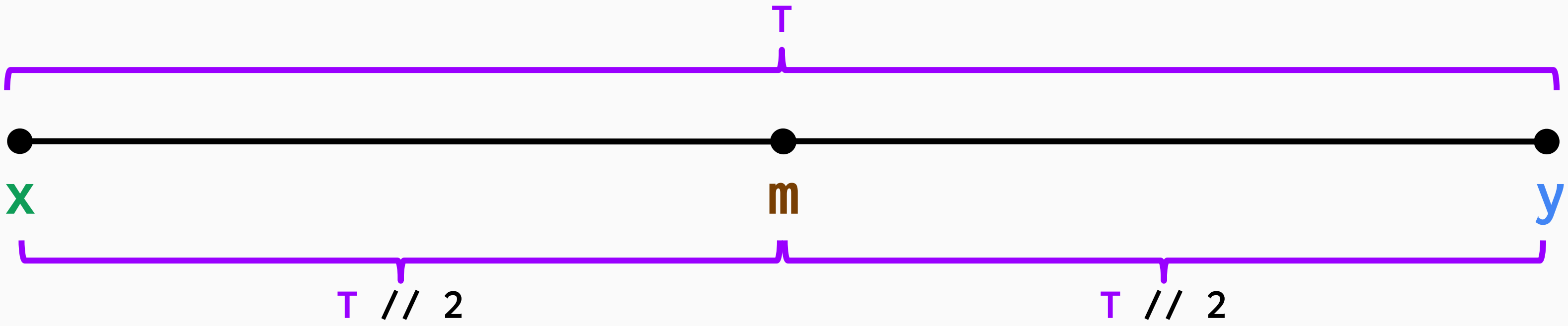


Pietrzak prover



$$\boxed{y == x^{2**T}} \quad \Leftrightarrow \quad \boxed{y == m^{2**(T\ //\ 2)} \text{ and } m == x^{2**(T\ //\ 2)}}$$

Pietrzak prover



$$\boxed{y == x^{2**T}} \quad \Leftrightarrow \quad \boxed{y == m^{2**(T\ //\ 2)} \text{ and } m == x^{2**(T\ //\ 2)}}$$

$$\Leftrightarrow \quad \boxed{r \text{ random and } ym^{r} == (mx^{r})^{2**(T\ //\ 2)}}$$

RSA modulus

Unsatisfactory approaches

RSA challenge

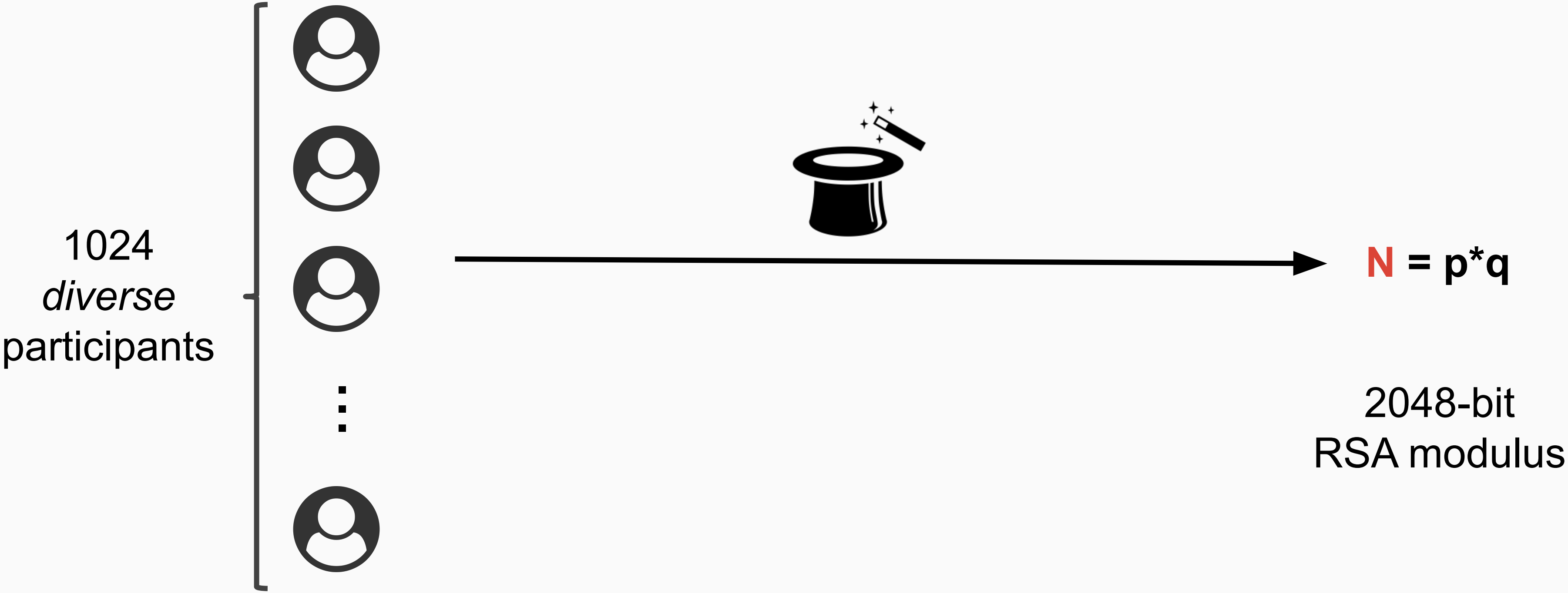
1991

Unsatisfactory approaches

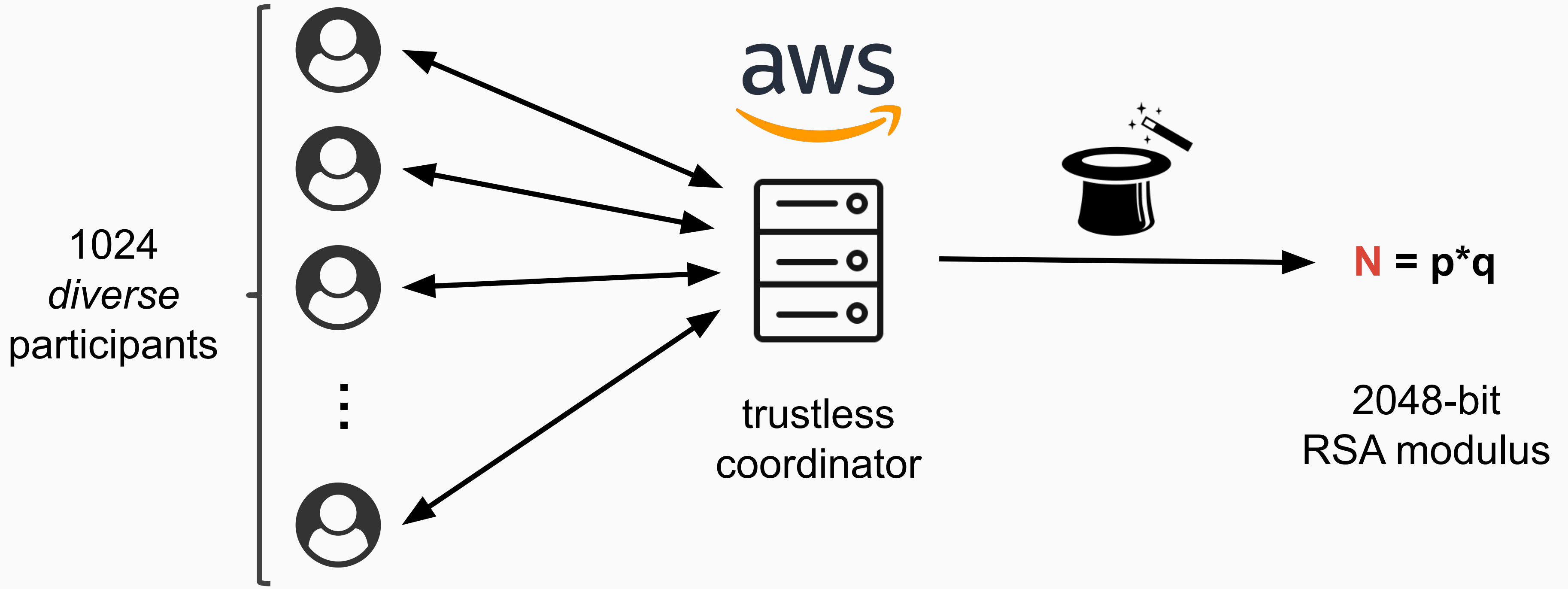
RSA challenge

RSA UFOs

RSA ceremony



RSA ceremony



RSA MPC

modulus size	2048 bits (two 1024-bit factors)
security	(n – 1)-maliciously secure
participants	1024

RSA MPC

modulus size	2048 bits (two 1024-bit factors)
security	(n – 1)-maliciously secure
participants	1024

LIGERO

UNIVERSITY of ROCHESTER

Northeastern

Bar-Ilan University

RSA MPC

modulus size	2048 bits (two 1024-bit factors)
security	(n – 1)-maliciously secure
participants	1024

synchronicity	synchronous
communication	<100 MB
duration	<10 minutes
rounds	<10 rounds

LIGERO

UNIVERSITY of ROCHESTER

Northeastern

Bar-Ilan University

RSA MPC key ideas

Passive adversary

- constructive sieving
- compute products (threshold AHE)
- Boneh-Franklin bi-primality test

RSA MPC key ideas

Passive adversary

- constructive sieving
- compute products (threshold AHE)
- Boneh-Franklin bi-primality test

Active adversary

- reveal failures
- zk-prove success

Generate candidates (additively homomorphic encryption)

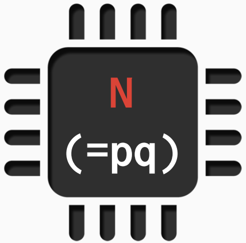
- **secret keys** sk_i
- **shared key** PK
- **encryption** $Enc_{PK}(m)$
- **decryption** $\Sigma\ Dec_{sk\_i}(c)$
- **shares** p_i, q_i

Generate candidates (additively homomorphic encryption)

- **secret keys** sk_i
- **shared key** PK
- **encryption** $Enc_{PK}(m)$
- **decryption** $\Sigma\ Dec_{sk\_i}(c)$
- **shares** p_i, q_i

encrypt	$Enc_{PK}(p_i)$
sum	$Enc_{PK}(p)$
encrypt	$Enc_{PK}(p*q_i)$
sum	$Enc_{PK}(p*q)$
decrypt	$Dec_{sk\_i}(Enc_{PK}(p*q))$
sum	$p * q$

Rebirth of RSA cryptography



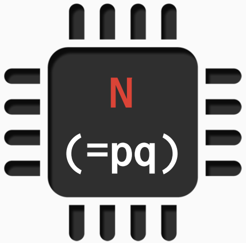
1ns per operation

RSA VDFs — 2018

RSA accumulators — 2002, 2007, 2018

RSA SNARKs — 2019

Rebirth of RSA cryptography



1ns per operation

RSA VDFs 2018

RSA accumulators 2002 2007 2018

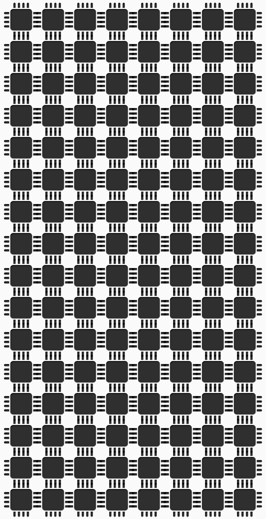
RSA SNARKs 2019

class groups (CPU)
10us per operation

chia

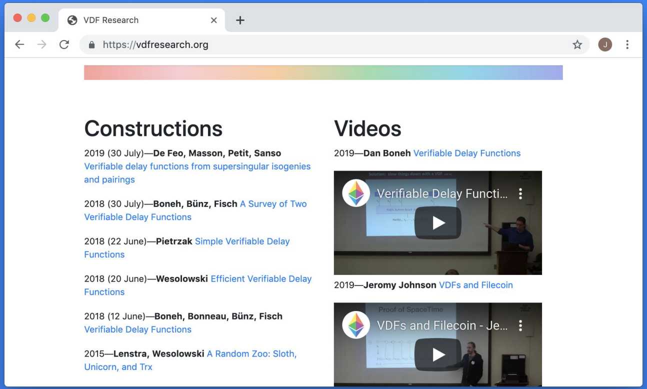
RSA SNARK "Supersonic" prover time

d*log(d) exponentiations ~ 128*d*log(d) multiplications

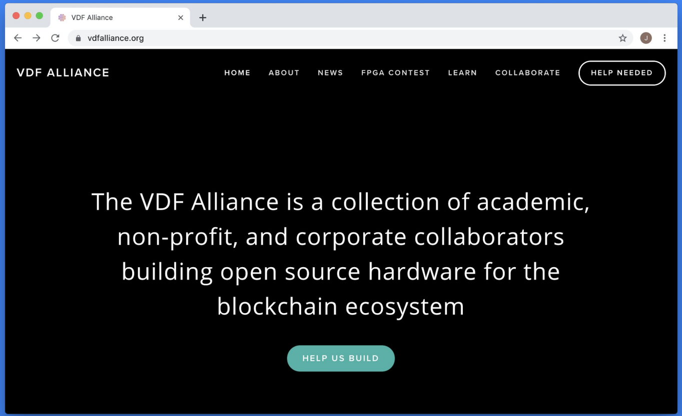


gates	prover time (128 cores)	proof size
2^10	10 us	10kB
2^20	20 ms	20kB
2^30	30 s	30kB

thank you :)



vdfresearch.org



vdfralliance.org