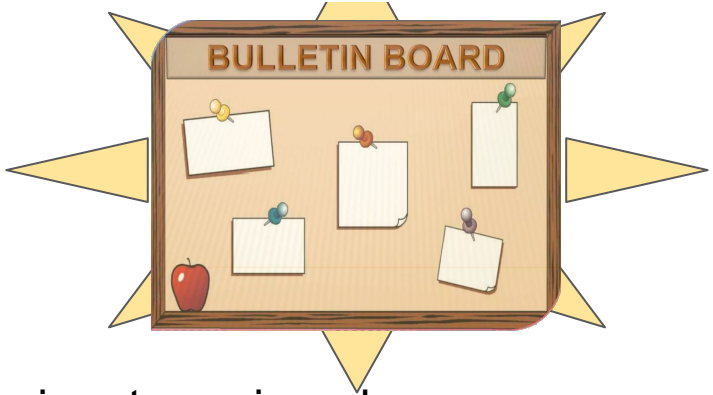
Bootstrapping Message-Linear-Constant-Round Consensus from a Bare PKI Setup, and Separation Bounds from the Idealized Message-Authentication Model

Matthieu Rambaud 2023-11-3 UMD crypto reading group

Goal: separation between the bare PKI setup..



t=0 : inputs assigned

.. and the union of all its mainstream implications:

authentication of the issuer of a message

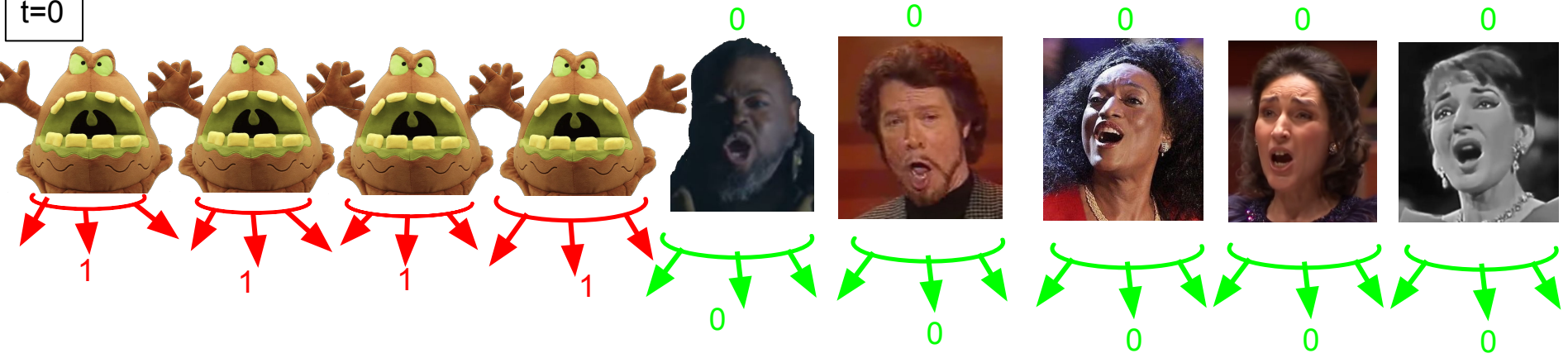
public random string

NIZK [Groth-Ostrovsky]

Toy model: corruptions = semi-honest or initially crashed.

Toy BA: multicast one's input bit..

t=0



t=1

..output the majority bit received

Linear communication with "conditional multicast" (a.k.a. self-sortition)

bulletin board, 𝄞 = public seed

t=0

t=1

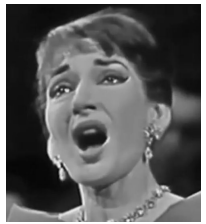
publishes

EvalProve(♪, 𝄞)

publicly available Verifiable Random Function (VRF oracle)

Verify((♪, 𝄞), 👍 , 📄)

"Yes"



(👍 , 📄) , with ℙ=λ/n
(else ∅)

0, (👍 , 📄)

accept 0

⇒ λ players in expectation are allowed to multicast

Problem if 𝄞 known <u>before</u> publication of keys: adversarial key-by-key picking

t=0

publishes ⇐

EvalProve(♪,𝄞)

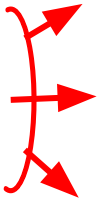
EvalProve(♩,𝄞)

EvalProve(♪,𝄞)

⋮

EvalProve(♪,𝄞)

(👍 , 📄)



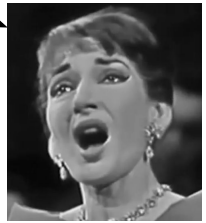
1, (👍 , 📄)

publicly available Verifiable Random Function (VRF oracle)

t=1

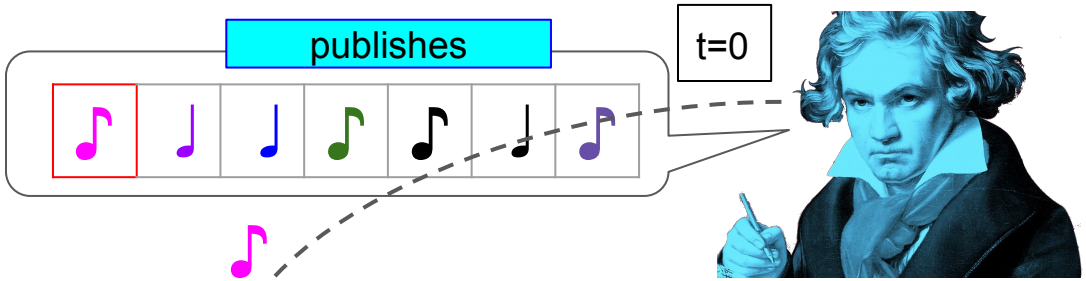
Verify((♪,𝄞),👍 , 📄)

“Yes”



accept 1

1st Way around: the imposed keys 🎵 model [Lindell et al,STOC'02] , [Chan-Pass-Shi EC'19, Podc'19, PKC'20] [Blum et al, TCC'20], [Blum et al, DISC'20]



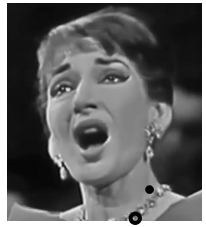
t=1

EvalProve(🎵,𝄞)
EvalProve(♩,𝄞)
EvalProve(🎵,𝄞)
⋮
EvalProve(🎵,𝄞)
(👍 , 📄)


VRF oracle

Verify((🎵,𝄞),👍 , 📄)

"No"

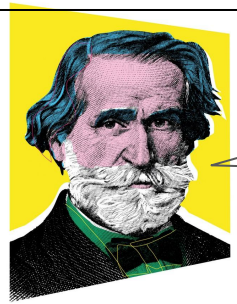


∅

1, (👍 , 📄)

2nd Way around: the unpredictable 𝄡 seed revealed <u>after</u> publication of keys model

Thunderella [Pass-Shi, EC'17], Algorand [Gilad et al, SOSP'17], Praos [David et al, EC'18], [Goyal et al, FC'21] and [Momose et al, CCS'22 and CCS'23]



t=0

t=1

Publishes

𝄡

EvalProve(♪,𝄞)

EvalProve(♩,𝄞)

EvalProve(♪,𝄞)

⋮

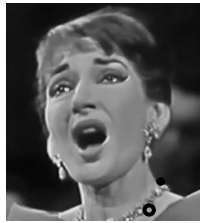
EvalProve(♪,𝄞)

(👍 , 📄)

VRF oracle

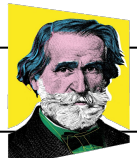
Verify((♪, 𝄡), 👍 , 📄)

"No"



∅

1, (👍 , 📄)

.. 😕 no existing implementation of  has linear complexity

Achieving linear complexity in the bare PKI model:

𝄡 = H(♪ ♩ ♪ ♪ ♪ ♩ ♪ ♫ ♪), H collision-free, e.g., identity

EvalsProofs(♪ ♩ ♪ ♪ , 𝄡)

ℙ(👍)=λ/n ℙ(👍)=λ/n ℙ(👍)=λ/n ℙ(👍)=λ/n

(1-ε)n/2 corruptions

EvalsProofs(♪ ♩ ♪ ♪ , 𝄢)

ℙ(👍)=λ/n ℙ(👍)=λ/n ℙ(👍)=λ/n ℙ(👍)=λ/n

EvalsProofs(♪ ♩ ♪ ♪ , 𝄞)

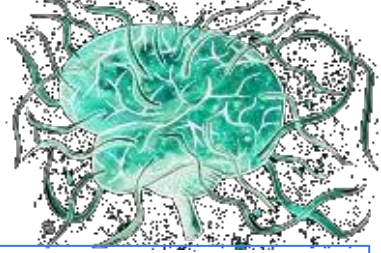
ℙ(👍)=λ/n ℙ(👍)=λ/n ℙ(👍)=λ/n ℙ(👍)=λ/n

VRF oracle

In each new vector output:
all "ℙ(👍)=λ/n" are independent
because 𝄞 is new
⇒ℙ(👍's ≥ λ/2 👍)=O(exp(-ε².λ))
⇒ over all q trials:
ℙ(👍's ≥ λ/2)=O(q.exp(-ε².λ))

Impossibility of authenticated consensus with subquadratic multicast complexity

corrupts on-the-fly players which multicast in the simulated execution, they send their simulated messages to Jim only



corrupts on-the-fly players which multicast in the simulated execution, they send their simulated messages to Jim only

World W_h,0: all players are initially honest, with input 0

Choose any Jim, leave it honest..

World W_h,1: all players are initially honest, with input 1