Efficient Rational Secret Sharing in Standard Communication Networks

Georg Fuchsbauer Jonathan Katz David Naccache

École Normale Supérieure

University of Maryland

TCC 2010

Our Contributions

New protocols for rational secret sharing

- Improved *efficiency* (no generic MPC; better parameters than prior work) and *optimal resilience*
- Work in *standard networks* (no simultaneous channels; no broadcast; can even handle asynchronous networks)
- Satisfy *strong solution concepts*

Our Contributions

New protocols for rational secret sharing

- Improved *efficiency* (no generic MPC; better parameters than prior work) and *optimal resilience*
- Work in *standard networks* (no simultaneous channels; no broadcast; can even handle asynchronous networks)
- Satisfy *strong solution concepts*

New solution concepts for rational cryptography

- (Computational) *strict* Nash; resistance to trembles
- Removing covert channels without physical assumptions

1. Rational Secret Sharing – An Overview

2. Prior Work

3. Our Protocols

(Classical) t-out-of-n Secret Sharing

Dealer *shares* a secret s among parties $P_1, \ldots, P_n$

Dealer D holding s computes shares s_i; gives s_i to P_i s.t.

- any group of size $\geq t$ can reconstruct s
- any group of size $< t$ has no information about s

Shamir's Secret Sharing

- D chooses random polynomial f of degree $t-1$ with $f(0) = s$
- Gives (signed copy of) $s_i = f(i)$ to each party P_i
- To reconstruct, all parties simultaneously broadcast their shares

(Classical) t-out-of-n Secret Sharing

Dealer *shares* a secret s among parties $P_1, \ldots, P_n$

Dealer D holding s computes shares s_i; gives s_i to P_i s.t.

- any group of size $\geq t$ can reconstruct s
- any group of size $< t$ has no information about s

Shamir's Secret Sharing

- D chooses random polynomial f of degree $t - 1$ with $f(0) = s$
- Gives (signed copy of) $s_i = f(i)$ to each party P_i
- To reconstruct, all parties simultaneously broadcast their shares

(Classical) t-out-of-n Secret Sharing

Dealer *shares* a secret s among parties $P_1, \ldots, P_n$

Dealer D holding s computes shares s_i; gives s_i to P_i s.t.

- any group of size $\geq t$ can reconstruct s
- any group of size $< t$ has no information about s

Implicit assumption

Each party either *honest* or *corrupt*; honest parties will cooperate during the reconstruction phase

Rational Secret Sharing

All players are rational and want to maximize their utility

Classical schemes fail in the rational setting: e.g., in Shamir's scheme broadcasting your share is *not* rational

Rational Secret Sharing

All players are rational and want to maximize their utility

Classical schemes fail in the rational setting: e.g., in Shamir's scheme broadcasting your share is *not* rational

This motivates the problem of *rational* secret sharing [HT04, GK06, LT06, ADGH06, KN08a, KN08b, OPRV09, MS09, AL09]:

- Set of n computationally bounded parties $P_1, \ldots, P_n$
- Sharing phase: D holds random s; gives share s_i to P_i
- Reconstruction phase: Players run protocol Π to reconstruct the secret

Utilities

- We say that P_i learns the secret iff it outputs s
  - Takes into account the fact that the $\{P_i\}$ are computationally bounded
  - Models learning partial information about the secret
- Outcome: $(o_1, \ldots, o_n) \in \{0,1\}^n$, with $o_i = 1$ iff P_i learned secret

Utilities

- We say that P_i learns the secret iff it outputs s
  - Takes into account the fact that the $\{P_i\}$ are computationally bounded
  - Models learning partial information about the secret
- Outcome: $(o_1, \ldots, o_n) \in \{0,1\}^n$, with $o_i = 1$ iff P_i learned secret
- P_i's utility: $\mu_i : \{0,1\}^n \to \mathbb{R}$
- Assumptions regarding players' utilities:
  1. Above all, players want to learn the secret
  2. Second, they prefer as few other players as possible learn it

Equilibrium Notions

A strategy σ_i is a prob. poly-time interactive Turing machine
Given strategies $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_n)$, we let $u_i(\boldsymbol{\sigma})$ denote the expected utility of P_i if each player P_j runs σ_j

(Computational) Nash equilibrium: no efficient deviation from the protocol increases expected utility (more than a negligible amount)

Equilibrium Notions

A strategy σ_i is a prob. poly-time interactive Turing machine
Given strategies $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_n)$, we let $u_i(\boldsymbol{\sigma})$ denote the expected utility of P_i if each player P_j runs σ_j

(Computational) Nash equilibrium: no efficient deviation from the protocol increases expected utility (more than a negligible amount)

Computational Nash (2-player case)

$\Pi = (\sigma_1, \sigma_2)$ induces a computational Nash eq. iff for all efficient σ'_1

$$u_1(\sigma'_1, \sigma_2) \leq u_1(\sigma_1, \sigma_2) + \mathrm{negl}(k)$$

(and similarly for P_2)

Insufficiently strong to rule out some naturally "bad" protocols

Here: Stronger Equilibrium Notions

Several suggestions in prior work for strengthening Nash solution concept; these have problems of their own

Here, we introduce two new notions (based on suggestions in [Katz08])

- Computational strict Nash: detectable deviations decrease utility
  - Implies that there is a *unique* legal message at each point in the protocol — no covert channels! (An explicit goal in other work.)
- Stability w.r.t. trembles: best to follow protocol even if other parties may deviate (arbitrarily) with small probability

See the paper for formalizations

Prior Work

Common Approach [HT04,GK06,. . .]

- Idea: Proceed in iterations; punish players for incorrect behavior
- In each iteration, dealer distributes shares of either
  - the real secret with some probability β
  - a fake secret otherwise
- Players broadcast their shares simultaneously
  - If a player deviates, all others stop protocol
  - If fake secret reconstructed $\Rightarrow$ go to next iteration
- To cheat, a party has to guess the *real* iteration; thus if β is small enough it is rational to follow the protocol
- Online dealer can be simulated using secure MPC

Prior Work

Common Approach [HT04,GK06,...]

- Idea: Proceed in iterations; punish players for incorrect behavior
- In each iteration, dealer distributes shares of either
  - the real secret with some probability β
  - a fake secret otherwise
- Players broadcast their shares simultaneously
  - If a player deviates, all others stop protocol
  - If fake secret reconstructed $\Rightarrow$ go to next iteration
- To cheat, a party has to guess the *real* iteration; thus if β is small enough it is rational to follow the protocol
- Online dealer can be simulated using secure MPC

Prior Work

Common Approach [HT04,GK06,...]

- Idea: Proceed in iterations; punish players for incorrect behavior
- In each iteration, dealer distributes shares of either
  - the real secret with some probability β
  - a fake secret otherwise
- Players broadcast their shares simultaneously
  - If a player deviates, all others stop protocol
  - If fake secret reconstructed $\Rightarrow$ go to next iteration
- To cheat, a party has to guess the *real* iteration; thus if β is small enough it is rational to follow the protocol
- Online dealer can be simulated using secure MPC

Prior Work

Common Approach [HT04,GK06,. . .]

- Idea: Proceed in iterations; punish players for incorrect behavior
- In each iteration, dealer distributes shares of either
  - the real secret with some probability β
  - a fake secret otherwise
- Players broadcast their shares simultaneously
  - If a player deviates, all others stop protocol
  - If fake secret reconstructed $\Rightarrow$ go to next iteration
- To cheat, a party has to guess the *real* iteration; thus if β is small enough it is rational to follow the protocol
- Online dealer can be simulated using secure MPC

Prior Work

Common Approach [HT04,GK06,. . .]

- Idea: Proceed in iterations; punish players for incorrect behavior
- In each iteration, dealer distributes shares of either
  - the real secret with some probability β
  - a fake secret otherwise
- Players broadcast their shares simultaneously
  - If a player deviates, all others stop protocol
  - If fake secret reconstructed $\Rightarrow$ go to next iteration
- To cheat, a party has to guess the *real* iteration; thus if β is small enough it is rational to follow the protocol
- Online dealer can be simulated using secure MPC

Drawbacks of Previous Work

Using generic secure MPC is inefficient

Communication networks:

- All prior work seems to require broadcast
- Most prior work needs simultaneous broadcast
- Other work relies on physical assumptions

Kol and Naor give a protocol that does not use generic secure MPC, and does not assume simultaneous channels

Drawbacks of Previous Work

Using generic secure MPC is inefficient

Communication networks:

- All prior work seems to require broadcast
- Most prior work needs simultaneous broadcast
- Other work relies on physical assumptions

Kol and Naor give a protocol that does not use generic secure MPC, and does not assume simultaneous channels

Advantages of our protocols

- Shares of bounded length; better round complexity
- Resistance to coalitions
- No broadcast channel needed; even asynchronous networks ok
- Different solution concepts

Main Ideas

Main Idea I

- Rely on same high-level structure (using real/fake iterations) as in previous work
- Previous work allows parties to recognize the real iteration *as soon as it occurs*
  - Inherently requires simultaneous channels
- Here, the good iteration is not identified until the *following* round

Main Ideas

Main Idea I

- Rely on same high-level structure (using real/fake iterations) as in previous work
- Previous work allows parties to recognize the real iteration *as soon as it occurs*
  - Inherently requires simultaneous channels
- Here, the good iteration is not identified until the *following* round

Main Idea II

- Real iteration identified using *verifiable random functions* (VRFs)
  - VRFs can be replaced by trapdoor permutations
- Unique proofs ensure a unique legal message in each round

Sketch of the Protocol for $n = 2$

Sharing of $s \in \{0,1\}^\ell$

- Choose real round $r^* \sim \text{GeomDist}(\beta)$
- Generate keys for VRF: $(pk_i, sk_i), (pk'_i, sk'_i)$ for $i \in \{1,2\}$
- Give to P_1 (analogously for P_2):

$$\left(sk_1,\ sk'_1,\ pk_2,\ pk'_2,\ \text{share}_1 := F_{sk_2}(r^*) \oplus s,\ \text{signal}_1 := F_{sk'_2}(r^* + 1)\right)$$

Sketch of the Protocol for $n = 2$

Sharing of $s \in \{0,1\}^\ell$

- Choose real round $r^* \sim \text{GeomDist}(\beta)$
- Generate keys for VRF: $(pk_i, sk_i), (pk'_i, sk'_i)$ for $i \in \{1,2\}$
- Give to P_1 (analogously for P_2):

$$\left(sk_1,\ sk'_1,\ pk_2,\ pk'_2,\ \text{share}_1 := F_{sk_2}(r^*) \oplus s,\ \text{signal}_1 := F_{sk'_2}(r^*+1)\right)$$

Reconstruction phase (P_1's view, iteration r)

- Send $F_{sk_1}(r), F_{sk'_1}(r)$ and proofs
- Receive $y^{(r)}, z^{(r)}$ and proofs. Then:
  - If $\text{signal}_1 = z^{(r)}$ then output $s^{(r-1)} := \text{share}_1 \oplus y_2^{(r-1)}$ and halt
  - If P_2 aborted or sent incorrect proofs, output $s^{(r-1)}$ and halt
  - Otherwise, go to next iteration

Sketch of the Protocol for $n = 2$

Sharing of $s \in \{0,1\}^\ell$

- Choose real round $r^* \sim \text{GeomDist}(\beta)$
- Generate keys for VRF: $(pk_i, sk_i), (pk'_i, sk'_i)$ for $i \in \{1,2\}$
- Give to P_1 (analogously for P_2):

$$\Big(sk_1,\ sk'_1,\ pk_2,\ pk'_2,\ \text{share}_1 := F_{sk_2}(r^*) \oplus s,\ \text{signal}_1 := F_{sk'_2}(r^* + 1)\Big)$$

Reconstruction phase (P_1's view, iteration r)

- Send $F_{sk_1}(r), F_{sk'_1}(r)$ and proofs
- Receive $y^{(r)}, z^{(r)}$ and proofs. Then:
  - If $\text{signal}_1 = z^{(r)}$ then output $s^{(r-1)} := \text{share}_1 \oplus y_2^{(r-1)}$ and halt
  - If P_2 aborted or sent incorrect proofs, output $s^{(r-1)}$ and halt
  - Otherwise, go to next iteration

Theorem

For appropriate choice of β, the above protocol induces a computational strict Nash equilibrium that is stable w.r.t. trembles.

Avoiding Verifiable Random Functions

Observation: in our protocol, VRFs are only evaluated in order

Idea

- Assume f trapdoor permutation with associated hardcore bit h, let y be random in $\mathrm{Dom}(f)$
- Define VRF(1) as $h(f^{-1}(y)), \ldots, h(f^{-\ell}(y))$

  Define VRF(2) as $h(f^{-\ell-1}(y)), \ldots, h(f^{-2\ell}(y))$

  $\vdots$
- Verifiable, since f efficiently computable

Extension to the "Exactly t-out-of-n" Case

- Dealer chooses r^*, assignes VRFs F_i, F'_i to P_i
- Makes t-out-of-n Shamir shares:
  - $s_1, \ldots, s_n$ of s
  - $z_1, \ldots, z_n$ of 0
- Each player P_i gets
  - s_j blinded by $F_j(r^*)$
  - z_j blinded by $F'_j(r^* + 1)$ for all j

Extension to the "Exactly t-out-of-n" Case

- Dealer chooses r^*, assignes VRFs F_i, F'_i to P_i
- Makes t-out-of-n Shamir shares:
  - $s_1, \ldots, s_n$ of s
  - $z_1, \ldots, z_n$ of 0
- Each player P_i gets
  - s_j blinded by $F_j(r^*)$
  - z_j blinded by $F'_j(r^* + 1)$ for all j
- Reconstruction
  - every player sends $F_i(r), F'_i(r)$
  - constructs polynomial to determine $r^* + 1$

Theorem

Theorem

Assume exactly t parties are active during the reconstruction phase. Then for appropriate choice of β, the above protocol induces $(t-1)$-resilient computational strict Nash equilibrium that is stable w.r.t. trembles.

See paper for:

- Extensions of the protocol for the case when $> t$ players may be active during reconstruction
- Definitions and a protocol for the case of asynchronous networks

Thank you! ☺