[I486S]
暗号プロトコル理論

藤崎 英一郎

北陸先端科学技術大学院大学

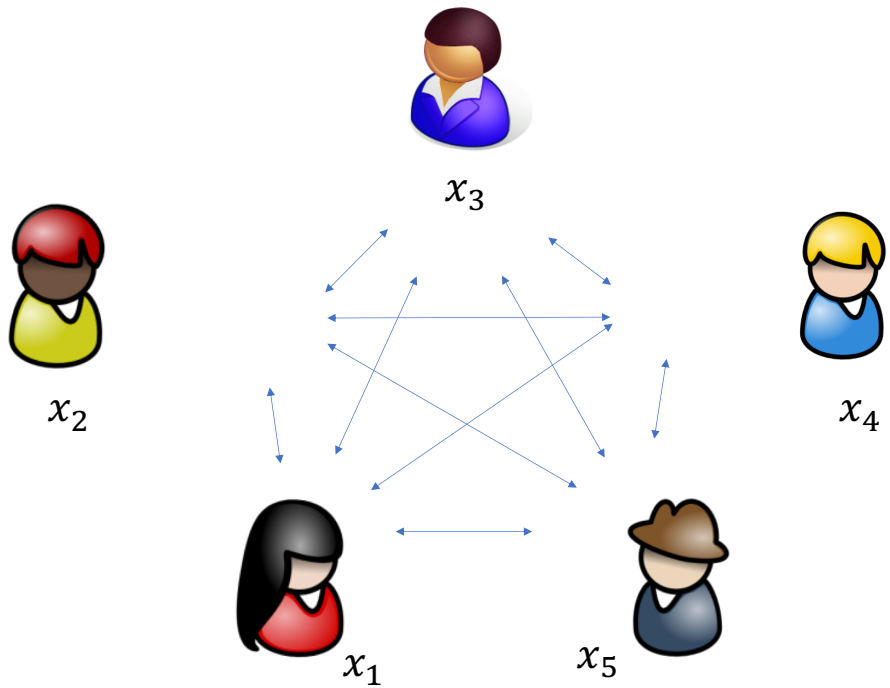
2020 年 7 月 7 日

本日の講義の内容

1. マルチパーティ計算
2. 安全性モデル

マルチパーティ計算

- 参加者: $P_1, \ldots, P_n$.
- 各 P_i への秘密の入力: $x_i \in \{0,1\}^\lambda$
- 全参加者への入力（公開情報）: 関数 $F : \{0,1\}^{n\lambda} \to \{0,1\}^*$.
- 各 P_i への出力：$F(x_1, \ldots, x_n)$. より一般的には、参加者ごとに違う出力をすることも許す.
- ネットワーク: Pair-wise private & synchronized.



MPC の構成（準備）

準備

- $F : \{0,1\}^{n\lambda} \to \{0,1\}^*$: 多項式時間計算可能関数
- C_F: F を計算する多項式サイズの AND, OR, NOT で構成された論理回路
- K: $n < \#K$ な有限体
- C_F の論理演算子への入力 $b \in \{0,1\}$ を $b \in \{0,1\} \subset K$ と K の元とみなす。
- AND, OR, NOT の論理ゲートを K 上の演算に置き換える。
  - $b \wedge b' \iff b \cdot b' \in K$.
  - $b \vee b' \iff b + b - b \cdot b' \in K$.
  - $\neg b \iff 1 - b \in K$.

能動的攻撃者存在の元での Shamir 秘密分散

- 参加者数は n、攻撃者数は t で、$n > 3t$.
- $a \to [a]$: a の秘密分散（コミット）。[BGW88] を使うことで、プロトコルが受理された時は正しく $[a]$ が行われたことが保証される。
- $[a] \Rightarrow a$: $a = (f(\alpha_1), \ldots, f(\alpha_n))$ を復元するとき、攻撃者が自分のシェアを正しく提示しなくても Reed-Solomon 符号の (Welch/Berlekamp) 復号によりもとの多項式 $f(X)$ が復元できる。

$$\hat{f}(\alpha_i) = f(\alpha_i) + e_i \text{ s.t. } W_H(\boldsymbol{e}) \leq t \quad \Longrightarrow \quad \text{Reconstruct } f(X)$$

用語等の定義

- $[a]_i$:　秘密 a が、参加者間で正しく線型秘密分散されており、a を P_i が保持している状態。
- $[a]_\emptyset$: 誰も a を 1 人で保持していない状態
- $[a]$:　誰が a を 1 人で保持しているか興味ないので明記していない場合
- $P_i : a \to [a]_i$:　P_i が、秘密 a を正しく秘密分散する行為（コミットメント）。
- $P_i : [a]_i \Rightarrow a$:　P_i が、秘密 a を各参加者に配ったシェアを証拠として broadcast して開示する行為。
- $[a]_i \Rightarrow a$:　全参加者が a のシェアを broadcast して、全員が a を認識する行為。
- $P_i : a \leftarrow [a]_j$:　参加者が各自の $[a]_j$ のシェアを P_i に秘密通信で送り、P_i に a を教える行為。この行為により、$[a]_i$ という状態にもなることに注意。

不正者が全体の 1/3 未満であれば、BGW VSS の Distribution は、$[a]$ を正しくコミットし、Reconstruction は、開示を正しく行うこと常にできるので、上記の具体的な実現法の一つである。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。
- $[a], [b] \to [a + b]$: 線形性から問題なし。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。
- $[a], [b] \to [a + b]$: 線形性から問題なし。
- $\lambda, [a] \to [\lambda a]$: 線形性から問題なし。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。
- $[a], [b] \to [a + b]$: 線形性から問題なし。
- $\lambda, [a] \to [\lambda a]$: 線形性から問題なし。
- $\lambda, [a] \to [a + \lambda]$: Shamir SS なら簡単。$[a + \lambda] = [a] + \lambda := (a_1 + \lambda, \ldots, a_n + \lambda)$.

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。
- $[a], [b] \to [a + b]$: 線形性から問題なし。
- $\lambda, [a] \to [\lambda a]$: 線形性から問題なし。
- $\lambda, [a] \to [a + \lambda]$: Shamir SS なら簡単。$[a + \lambda] = [a] + \lambda := (a_1 + \lambda, \ldots, a_n + \lambda)$.
- $[a], [b] \to [ab]$ をどうするか。$ab = \sum \lambda_{i,n} a_i b_i$ なので、各 P_i に a_i, b_i から自主的に $a_i b_i$ を作ってもらわなければいけない。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。
- $[a], [b] \to [a + b]$: 線形性から問題なし。
- $\lambda, [a] \to [\lambda a]$: 線形性から問題なし。
- $\lambda, [a] \to [a + \lambda]$: Shamir SS なら簡単。$[a + \lambda] = [a] + \lambda := (a_1 + \lambda, \ldots, a_n + \lambda)$.
- $[a], [b] \to [ab]$ をどうするか。$ab = \sum \lambda_{i,n} a_i b_i$ なので、各 P_i に a_i, b_i から自主的に $a_i b_i$ を作ってもらわなければいけない。
- 回路への入力をビット $a \in \{0, 1\}$ にしなければいけない。

能動的攻撃者 (active adversary) 対策方針

（大方針）能動的攻撃者を受動的攻撃者と同じことしかできないよう強制する。それでも従わない場合は、プロトコルから排除され、攻撃者の秘密は開示されプロトコルが続行される。

- $P_i : a \to [a]$: (BGW) VSS に変更して、不正な P_i でも正しく秘密分散せざるを得ないようにする。
- $[a] \Rightarrow a$: 全参加者が a のシェアを broadcast して、a を復元するとき、能動的攻撃者が正しくないシェアを broadcast しても、誤り訂正符号の技術を使って、正しく a を復元する。
- $[a], [b] \to [a + b]$: 線形性から問題なし。
- $\lambda, [a] \to [\lambda a]$: 線形性から問題なし。
- $\lambda, [a] \to [a + \lambda]$: Shamir SS なら簡単。$[a + \lambda] = [a] + \lambda := (a_1 + \lambda, \ldots, a_n + \lambda)$.
- $[a], [b] \to [ab]$ をどうするか。$ab = \sum \lambda_{i,n} a_i b_i$ なので、各 P_i に a_i, b_i から自主的に $a_i b_i$ を作ってもらわなければいけない。
- 回路への入力をビット $a \in \{0, 1\}$ にしなければいけない。
  $\Rightarrow [a(a-1)] = [0]$ を証明する！

この方式はオリジナルの論文 [BGW88] の方式を整理し修正した [CDN15] に従っている。

本日の講義の内容

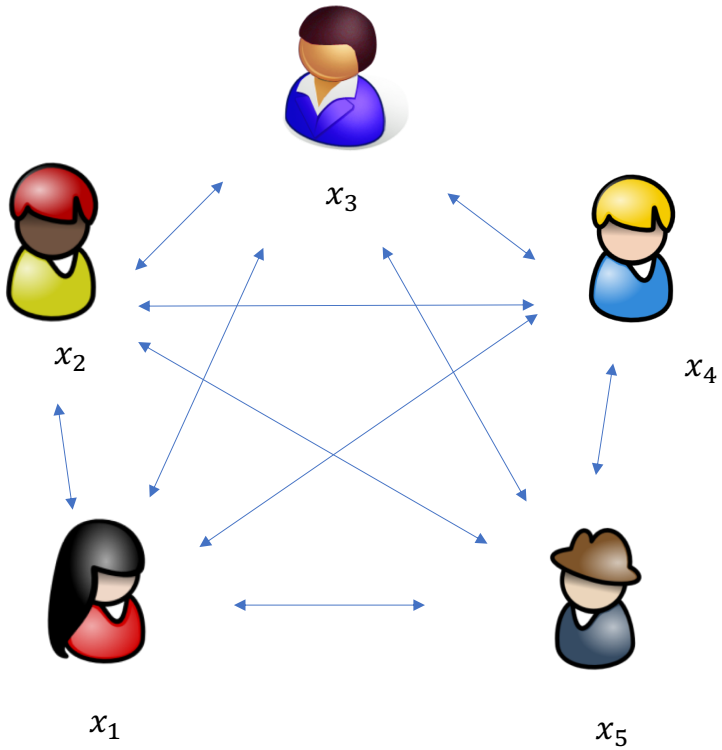
1. マルチパーティ計算
2. 安全性モデル

安全性モデル

大雑把に言うと、現実が、理想と同じ、または十分近くなれば安全という。

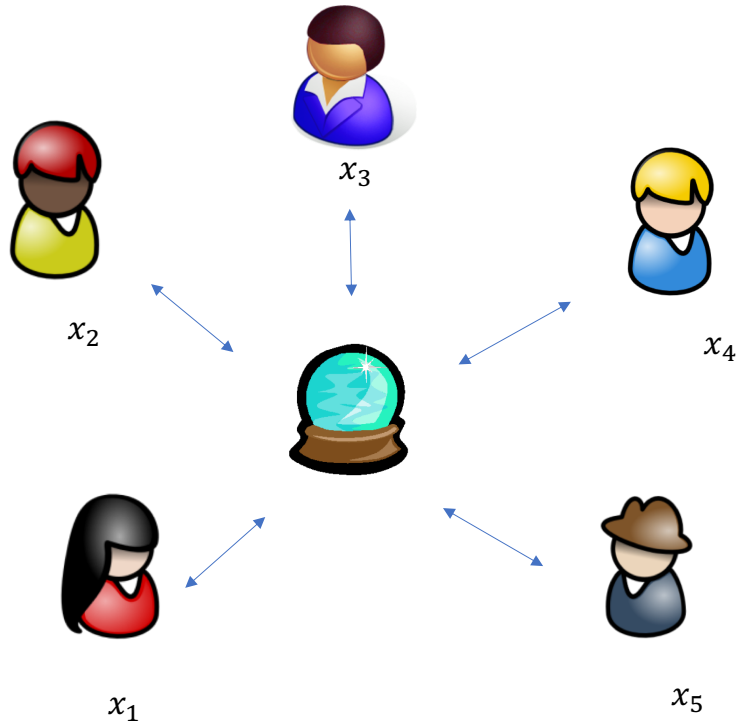
現実のプロトコル

互いに通信し計算結果を得る

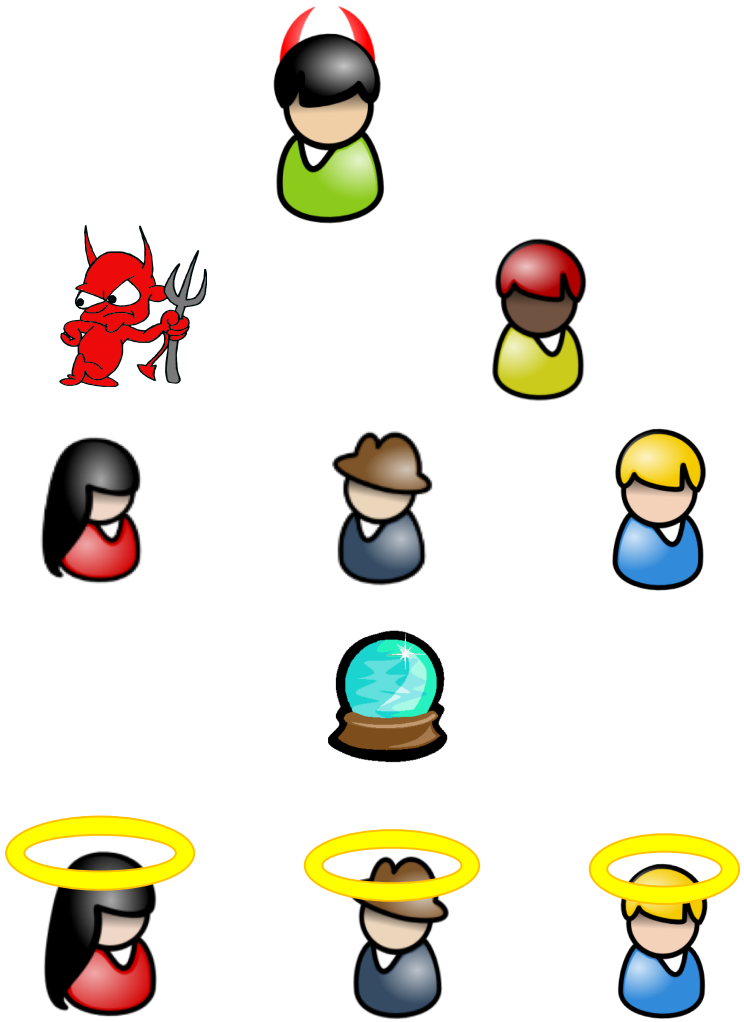


理想状態

各自の秘密を  に預けると計算結果を返してくれる



Environment Security



Environment Z

Adversary A / Simulator S

Parties (Alice, Bob, Charlie), who execute Protocol π

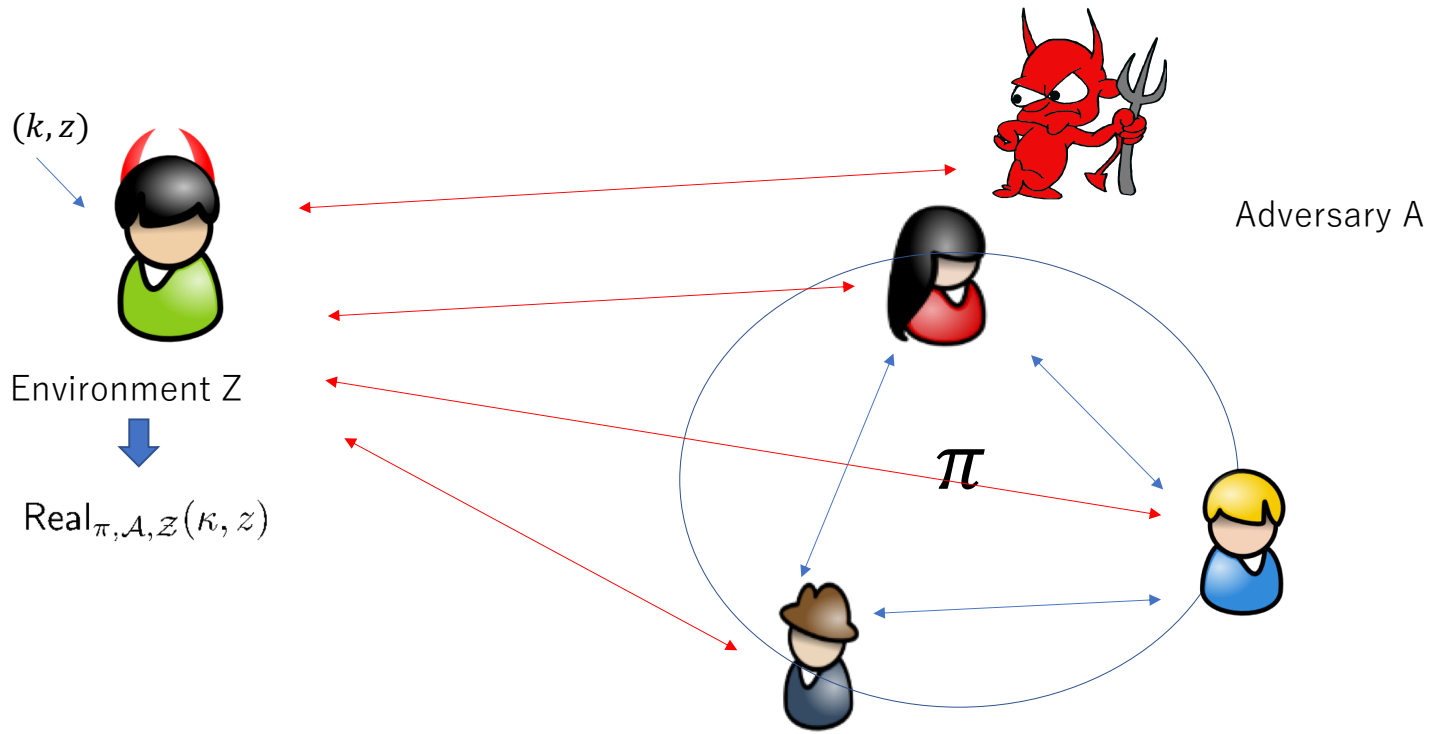
Ideal Functionality F

``functionality'' of F = ``functionality'' of π

Dummy Parties (Alice, Bob, Carlie)

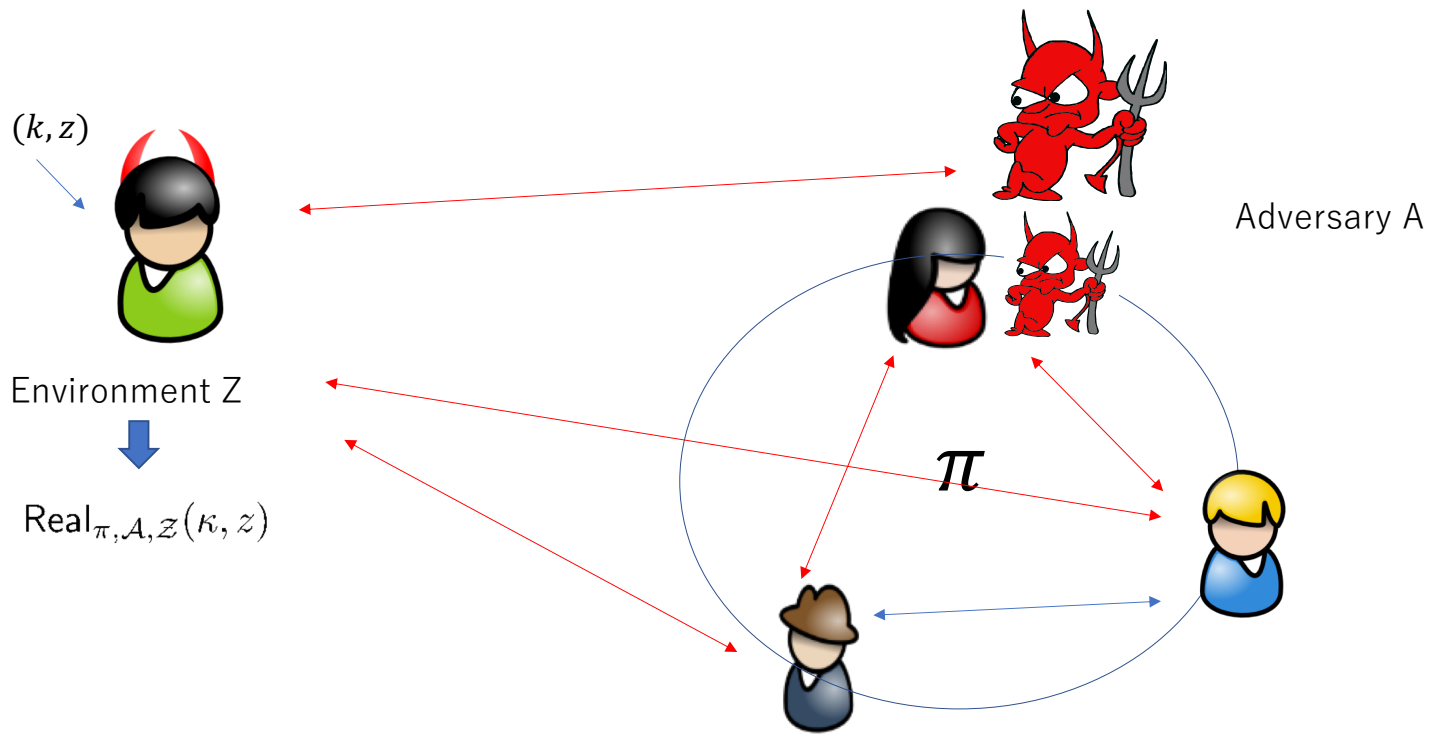
Real World

Environment $\mathcal{Z}$ は、 赤のチャネルから情報を得る。Adversary $\mathcal{A}$ は、$\mathcal{Z}$ の意のままに行動し、情報を $\mathcal{Z}$ に伝える。



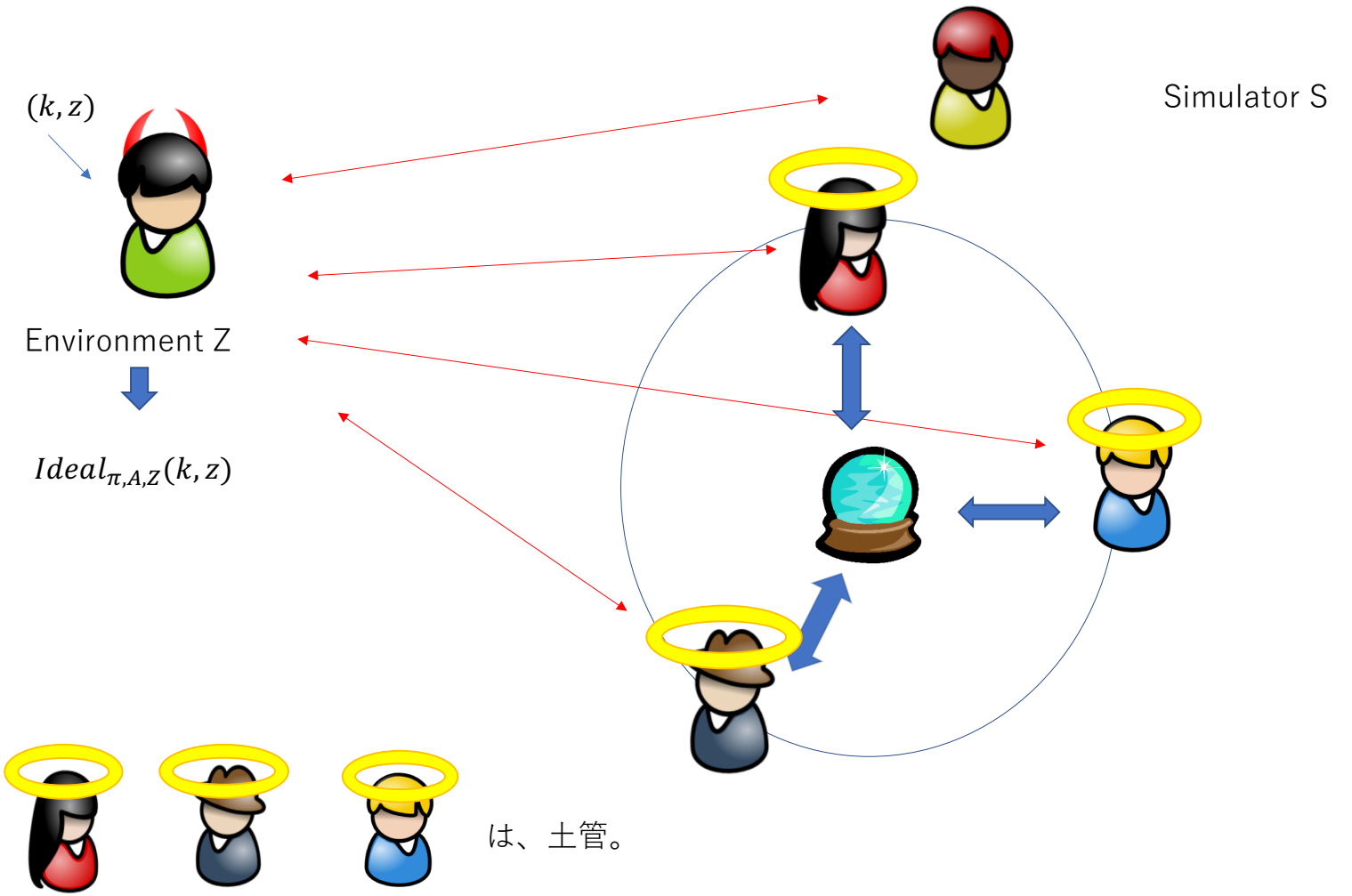
Real World (Alice Corrupted)

Environment $\mathcal{Z}$ は、 赤のチャネルから情報を得る。Adversary $\mathcal{A}$ は、$\mathcal{Z}$ の意のままに行動し、情報を $\mathcal{Z}$ に伝える。$\mathcal{A}$ は、Alice の（今までの内部情報を全て得て）代わりにプロトコルに参加する。



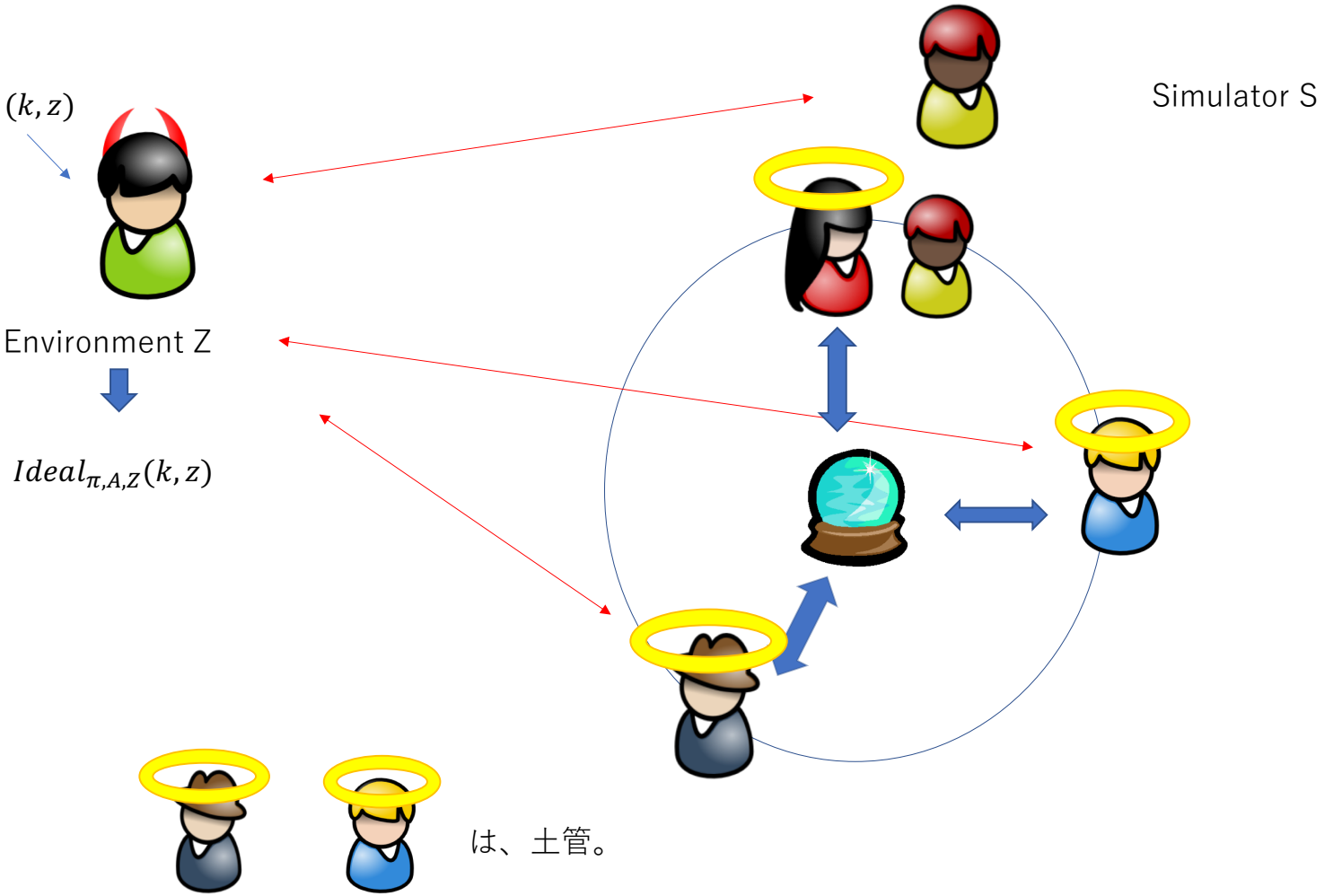
Ideal World

Environment $\mathcal{Z}$ は、 赤のチャネルから情報を得る。Simulator $\mathcal{S}$ は、$\mathcal{A}$ のふりをして、情報を $\mathcal{Z}$ に伝える。



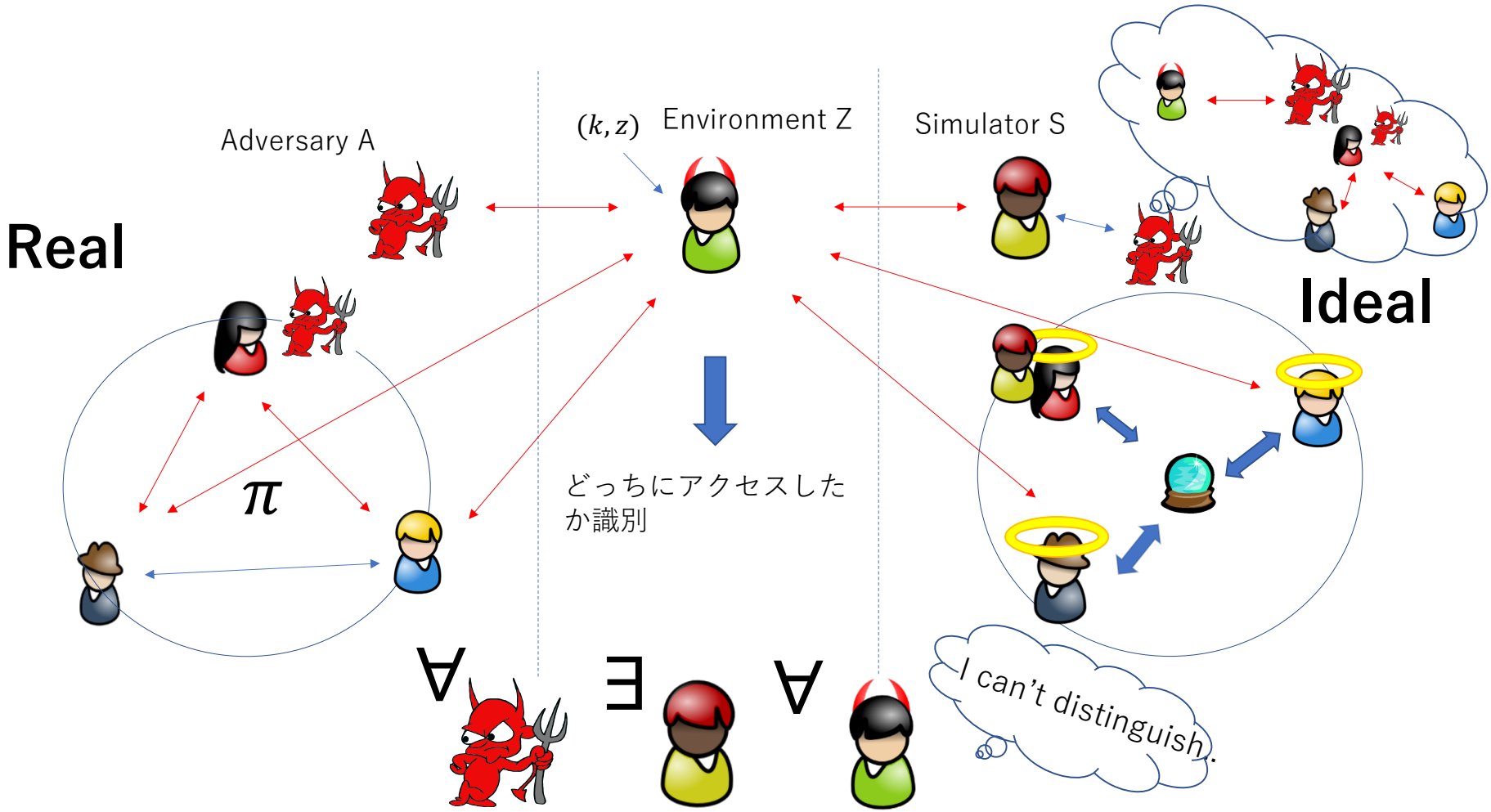
Ideal World (Alice Corrupted)

Environment $\mathcal{Z}$ は、 赤のチャネルから情報を得る。$\mathcal{S}$ は、Alice の（今までの内部情報を全て得て）代わりにプロトコルに参加する。



安全とは

Real と Ideal の世界を Environment $\mathcal{Z}$ が識別できなければ安全と定義



Adversary（攻撃者）

- Passive Adversary (aka semi-honest or honest-but-curious).
  - 受動的攻撃者
  - 正規のプロトコルからは逸脱しないが、プロトコルから得た情報から正直な参加者の秘密情報を得ようとする
- Active Adversary
  - 能動的攻撃者
  - プロトコルから逸脱しても良い。正直な参加者の秘密情報を得ようとすることと、プロトコルを失敗に終わらせようとする。
  - さらに、static case と adaptive case が存在する。
    - static adversary: プロトコルが始まる前に不正者が固定されている
    - adaptive adversary: プロトコル中に正直な参加者を corrupt して、今までの情報を得た上に不正者として操ることができる

知られている結果

不正者数（Adversary が corrupt できる参加者数）に応じた結果

	Passive	Active w/ broadcast	Active w/o broadcast
情報理論的安全	$< \frac{n}{2}$	$< \frac{n}{3}$	$< \frac{n}{3}$
統計的安全	$< \frac{n}{2}$	$< \frac{n}{2}$	$< \frac{n}{3}$
計算量的安全	n	$< \frac{n}{2}$	$< \frac{n}{2}$

ただし、$n > 2$.

- 情報理論的安全: Real と Ideal の分布が完全に同じ
- 統計的安全: Real と Ideal の分布の差がわずかな統計的差しかない。
- 計算量的安全: 多項式時間制限の Environment $\mathcal{Z}$ では、識別できないぐらいしか、Real と Ideal の分布に違いはない。

安全性

- すでに説明した BGW [BGW88] の受動的攻撃者に対する MPC は、参加者ごとの P2P の秘匿通信が仮定され、同期型のネットワークであれば、corrupt される参加者の数が半分未満であれば受動的攻撃者に対して情報理論的安全である。
- [CDN15] に記載される MPC（BGW のプロトコルの修正版）は、参加者ごとの P2P の秘匿通信が仮定され、同期型のネットワークであれば、corrupt される参加者の数が 1/3 未満であれば能動的攻撃者 (adavptively active?) に対して情報理論的安全である。

参考文献

[BGW88] M. Ben-Or, S. Goldwasser, and A. Wigderson.

Completeness theorems for non-cyptographic fault-tolerant distributed computation.

In Janos Simon, editor, STOC '88, pages 1–10. ACM, 1988.

[CDN15] Ronald Cramer, Ivan Damgård, and Jesper Nielsen.

Secure Multiparty Computation and Secret Sharing.

Cambridge University Press, 2015.