General Adversary Structures in BA and MPC with Active and Omission Corruption

Konstantinos Brazitikos and Vassilis Zikas

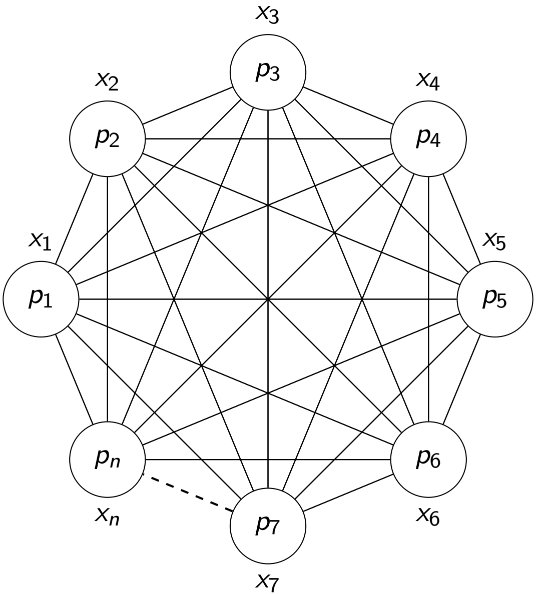
University of Edinburgh Georgia Tech

December 6

TCC 2024

Secure Multi-Party Computation

- Players $p_1, \ldots, p_n$
- Inputs $x_1, \ldots, x_n$
- Want to compute $f(x_1, \ldots, x_n)$



Setting the landscape

Perfect Security

- Information theoretic security, no setup, with zero error probability

Setting the landscape

Perfect Security

- Information theoretic security, no setup, with zero error probability

Adversary Characterisation

- Unbounded
- Static
- Rushing

Types of Corruption

Types of Corruption

Mixed Adversary

Types of Corruption

Mixed Adversary

- Active Corruption (Malicious)
  - Full access and control
  - Can deviate arbitrarily

Types of Corruption

Mixed Adversary

- Active Corruption (Malicious)
  - Full access and control
  - Can deviate arbitrarily
- Omission Corruption
  - No information leaks
  - Can obliviously block/erase any message

Motivation for Omissions

Motivation for Omissions

Omission corruption - A realistic type of failure

Motivation for Omissions

Omission corruption - A realistic type of failure

- Model real-life scenarios
  - Temporary connectivity issues (DoS, faulty connection, network outages, offline users)
  - If user can't follow the protocol entirely (going offline) but is still benign (unreliable but not malicious)

Motivation for Omissions

Omission corruption - A realistic type of failure

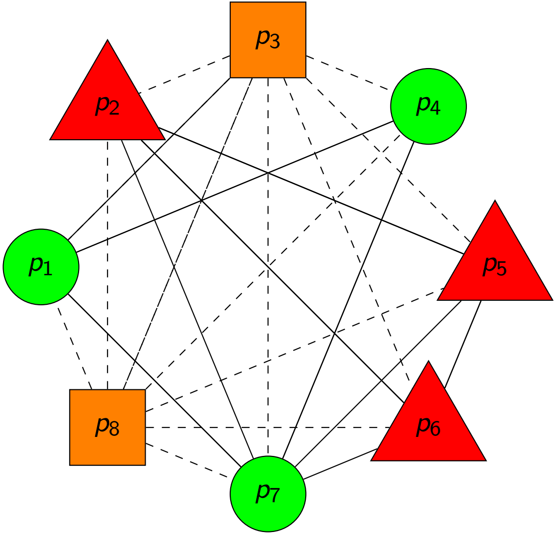
- Model real-life scenarios
  - Temporary connectivity issues (DoS, faulty connection, network outages, offline users)
  - If user can't follow the protocol entirely (going offline) but is still benign (unreliable but not malicious)
- Lies between active corruption and crash failures, more benign than the former, less benign than the latter

Motivation for Omissions

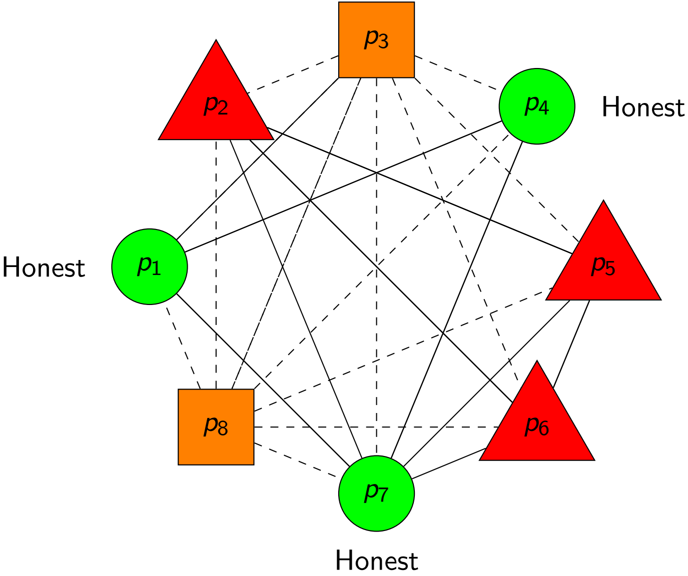
Omission corruption - A realistic type of failure

- Model real-life scenarios
  - Temporary connectivity issues (DoS, faulty connection, network outages, offline users)
  - If user can't follow the protocol entirely (going offline) but is still benign (unreliable but not malicious)
- Lies between active corruption and crash failures, more benign than the former, less benign than the latter
- A lot of recent work on omissions

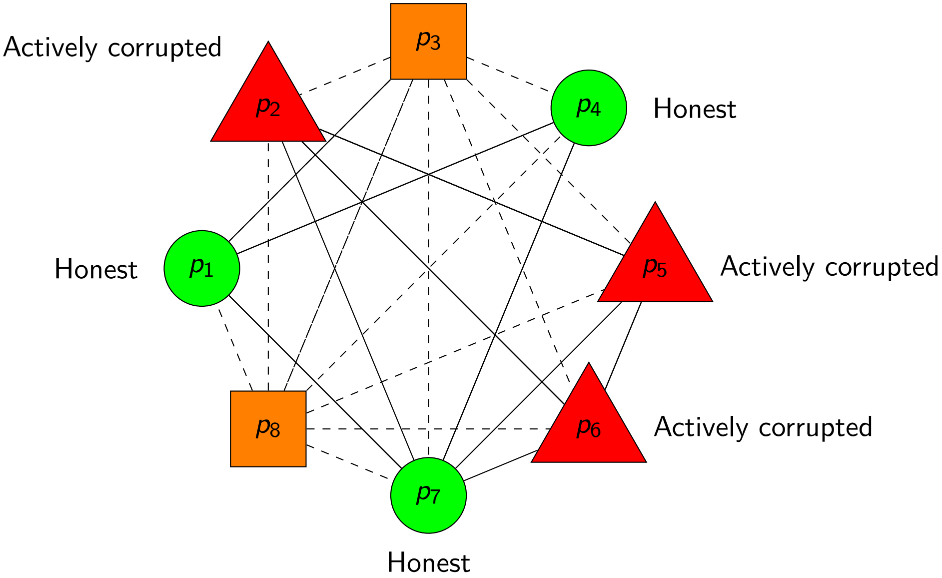
MPC with active and omission corruption



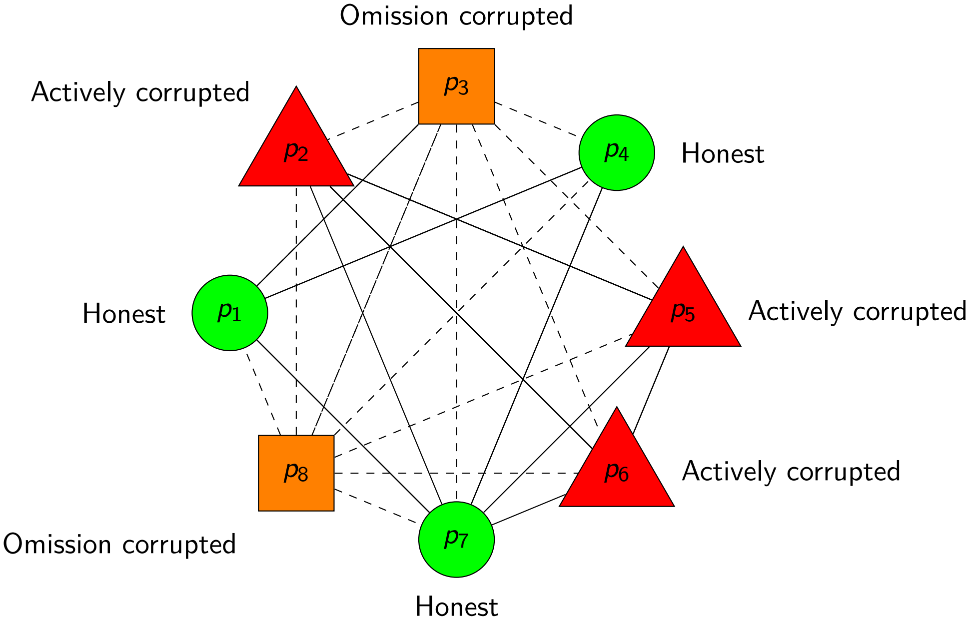
MPC with active and omission corruption



MPC with active and omission corruption



MPC with active and omission corruption



General Adversary Model [HM97]

Description through adversary structure $\mathcal{Z}$

- More expressive than threshold model, can describe situations that threshold cannot
- Contains classes $Z_1, Z_2, \ldots$ that the adversary can select from

General Adversary Model [HM97]

Description through adversary structure $\mathcal{Z}$

- More expressive than threshold model, can describe situations that threshold cannot
- Contains classes $Z_1, Z_2, \ldots$ that the adversary can select from

Adversary class Z_i of structure $\mathcal{Z}$

Contains a pair (A_i, Ω_i) of corrupted parties

- Set of actively corrupted parties A_i
- Set of omission-corrupted parties Ω_i

General Adversary Model: An Example

$Z_1 = (\{p_1\}, \emptyset)$, $Z_2 = (\{p_2\}, \emptyset)$, $Z_3 = (\{p_3\}, \emptyset)$, $Z_4 = (\{p_4\}, \emptyset)$

	p_1	p_2	p_3	p_4
Z_1	α			
Z_2		α		
Z_3			α	
Z_4				α

4 player secure MPC with only one player corrupted

General Adversary Model: An Impossibility Example

$Z_1 = (\{p_1\}, \{p_3\})$, $Z_2 = (\{p_2\}, \{p_3\})$, $Z_3 = (\emptyset, \{p_4\})$

	p_1	p_2	p_3	p_4
Z_1	α		ω	
Z_2		α	ω	
Z_3				ω

p_3 or p_4 always corrupted, p_3 cannot send message to p_4

Previous work

	Gen. Adv.	Active	Omis.	Perf. Sec.	Comp. Sec.	
$t < n/3$		✓		✓		[PSL80]
$t < n/3$		✓		✓		[LPS80]
$A_1 \cup A_2 \cup A_3 \neq \mathcal{P}$	✓	✓		✓		[HM97]
$A_1 \cup A_2 \cup A_3 \cup (F_1 \cap F_2 \cap F_3) \neq \mathcal{P}$	✓	✓		✓		[BFH+]
$t < n/2$			✓	✓		[PR03]
$3t_a + 2t_\omega < n$		✓	✓	✓		[ZHM09]
$2t_a + t_r + t_s < n$		✓	✓		✓	[ELT22]
$2t_a + t_r + t_s < n$		✓	✓		✓	[LS23]
$2t_a + t_r + t_s < n$		✓	✓		✓	[LSS24]
This work	✓	✓	✓	✓		[BZ24]

Our contributions

Our contributions

General Adversary for Active and Omission corruption

- Sufficient and necessary security condition for Byzantine Agreement (BA)
- Sufficient and necessary security condition for MPC
- Both results are optimal and cannot be improved

Our contributions

General Adversary for Active and Omission corruption

- Sufficient and necessary security condition for Byzantine Agreement (BA)
- Sufficient and necessary security condition for MPC
- Both results are optimal and cannot be improved

Simulation based definitions and proofs

- First ever UC treatment of the problem
- All existing Gen. Adv. protocols use composition but no composable treatment

Our results: **Tight characterization for perfectly secure MPC**

Necessary and sufficient condition for MPC $C_{MPC}^{(A,\Omega)}(\mathcal{P}, \mathcal{Z})$

For an adversary structure $\mathcal{Z}$ and a player set $\mathcal{P}$ we can get secure MPC if and only if we have

- condition for BA
- condition for SMT for every pair of players

Our results: **Tight characterization for perfectly secure MPC**

Necessary and sufficient condition for MPC $C_{MPC}^{(A,\Omega)}(\mathcal{P}, \mathcal{Z})$

For an adversary structure $\mathcal{Z}$ and a player set $\mathcal{P}$ we can get secure MPC if and only if we have

- condition for BA
- condition for SMT for every pair of players

$$C_{MPC}^{(A,\Omega)}(\mathcal{P}, \mathcal{Z}) \iff C_{BA}^{(A,\Omega)}(\mathcal{P}, \mathcal{Z}) \wedge \forall p_s, p_r \in \mathcal{P} : \; C_{SMT}^{(A,\Omega)}(\mathcal{P}, \mathcal{Z}, p_s, p_r)$$

Our results: Security Condition for BA

Sufficient and Necessary Condition $C_{BA}^{(A,\Omega)}(\mathcal{P}, \mathcal{Z})$

For an adversary structure $\mathcal{Z}$ and a player set $\mathcal{P}$ we get secure BA if and only if the following holds.

Our results: Security Condition for BA

Sufficient and Necessary Condition $C_{BA}^{(A,\Omega)}(\mathcal{P}, \mathcal{Z})$

For an adversary structure $\mathcal{Z}$ and a player set $\mathcal{P}$ we get secure BA if and only if the following holds.

$C_{BA}^{(A,\Omega)}(\mathcal{P}, \mathcal{Z}) \iff$ For any three classes with indices i, j, k :

$$A_i \cup A_j \cup A_k \cup (\Omega_i \cap \Omega_j) \neq \mathcal{P}$$

In contrast with the condition for active/fail: $A_i \cup A_j \cup A_k \cup (F_i \cap F_j \cap F_k) \neq \mathcal{P}$ [AFM99]

Condition for (detectable) Secure Message Transmission (SMT)

Necessary and sufficient condition $C_{SMT}^{(A,\Omega)}(\mathcal{P}, \mathcal{Z})$

Condition for (detectable) Secure Message Transmission (SMT)

Necessary and sufficient condition $C_{SMT}^{(A,\Omega)}(\mathcal{P}, \mathcal{Z})$

We have detSMT between a pair of parties p_s, p_r if and only if the following holds:

Condition for (detectable) Secure Message Transmission (SMT)

Necessary and sufficient condition $C_{SMT}^{(A,\Omega)}(\mathcal{P}, \mathcal{Z})$

We have detSMT between a pair of parties p_s, p_r if and only if the following holds:
$C_{SMT}^{(A,\Omega)}(\mathcal{P}, \mathcal{Z}) \iff$ For any three indices i, j, k :

$$p_s \in (\Omega_i \cap \Omega_j) \wedge p_r \in \Omega_k \implies$$
$$A_i \cup A_j \cup \boldsymbol{\Omega_k} \cup (\Omega_i \cap \Omega_j) \neq \mathcal{P}$$

Condition for (detectable) Secure Message Transmission (SMT)

Necessary and sufficient condition $C_{SMT}^{(A,\Omega)}(\mathcal{P},\mathcal{Z})$

We have detSMT between a pair of parties p_s, p_r if and only if the following holds:
$C_{SMT}^{(A,\Omega)}(\mathcal{P},\mathcal{Z}) \iff$ For any three indices i, j, k :

$$p_s \in (\Omega_i \cap \Omega_j) \wedge p_r \in \Omega_k \implies$$
$$A_i \cup A_j \cup \boldsymbol{\Omega_k} \cup (\Omega_i \cap \Omega_j) \neq \mathcal{P}$$

– and respectively for $(p_r \in \Omega_i \cap \Omega_j \wedge p_s \in \Omega_k)$

Our approach for Omissions

Our approach for Omissions

Difficulty dealing with them

Cannot tell whose fault it is when a message is dropped

Our approach for Omissions

Difficulty dealing with them

Cannot tell whose fault it is when a message is dropped

Our strategy

- Make protocols identifiable to detect omission-corrupted players
- Parties are either publicly identified or self-identified (we call them zombies) and step down from participating

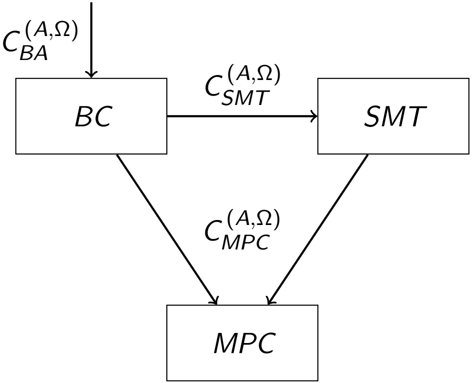
Pathway of our solution

Pathway of our solution

Our structure

- Consensus/Broadcast primitive
- Detectable SMT primitive
- Detectable MPC
- Robust MPC

Overview



Questions?