The Cost of Fluidity: Communication Complexity Trade-offs in Fluid MPC

Shancheng Zhang[1], Zongyang Zhang[1(✉)], and Bernardo Magri[2]

[1] Beihang University, Beijing, CN
{zscbuaa,zongyangzhang}@buaa.edu.cn
[2] University of Manchester, Manchester, UK
bernardo.magri@manchester.ac.uk

Abstract. Secure multi-party computation (MPC) enables mutually distrustful parties to jointly evaluate a function on their private inputs. Classic MPC protocols, however, assume a *static* set of participants in which every party must remain online throughout the entire computation. Recent advances have introduced MPC models with *dynamic participation*, such as Fluid MPC, in which computation steps are delegated to a sequence of committees that change across epochs. This approach improves robustness, enabling parties to go offline once their roles are complete. Yet, this flexibility comes at a cost: the most efficient dynamic-participation MPC protocols still incur communication overheads exceeding traditional MPC by more than an order of magnitude.
In this work, we formalize the communication complexity of (d, n)-threshold secret-sharing-based Fluid MPC. We prove a tight trade-off between communication cost and the adversary's corruption threshold, showing that linear communication complexity $O(n)$ is impossible when the corruption threshold t exceeds a proportion of d. Matching this bound, we construct a protocol with a communication cost of $9.3n$ elements per multiplication gate against a semi-honest adversary and $37.3n$ elements against a malicious adversary. A `C++` implementation confirms that our approach brings the cost of fluidity within practical limits.

1 Introduction

Secure multiparty computation [27] enables a set of mutually distrusting parties to compute a function on their private inputs while revealing nothing beyond the prescribed output. Over the past decade, the community has dramatically improved MPC's efficiency in both honest and dishonest-majority settings [8, 9, 22]. Yet all traditional protocols assume a fixed group of parties that remain online throughout the computation. This requirement is brittle for long-running tasks, such as training deep-learning models: a single network hiccup can stall the entire computation.

Recently, MPC models with *dynamic participation* have been introduced, in which the computation is carried out by committees that may change during execution. Notable examples include the YOSO model [19] and the Fluid MPC

model [7], where the existing Fluid MPC protocols [1, 2, 7, 24] are relatively efficient in practice. In the Fluid MPC model, the computation is divided into *epochs*, each executed by its own committee, with only a brief handoff phase between consecutive ones. This design allows parties to join or leave between epochs. When each epoch is restricted to a single communication round (that is, a committee may send messages only once to its successor), the protocol achieves what the authors call *maximal fluidity*.

The first Fluid MPC protocol incurred $O(n^2)$ field-element communication per gate in the honest majority setting, where n is the committee size [7]. Rachuri et al. [24] and Bienstock et al. [1] progressively extended this paradigm to the dishonest majority setting; notably, [1] achieves linear communication in the honest majority setting via a P_{king} mechanism. Subsequent works, such as [2,15,28], have reduced overhead further by adopting relaxed security models, including fluid participation or reduced corruption thresholds. However, while [1] remains the state-of-the-art regarding communication efficiency for the standard setting, its costs remain significantly higher than those of traditional MPC protocols like [8,20]. This gap currently limits the practical adoption of Fluid MPC. These observations prompt two natural questions:

(1) What is the inherent trade-off between communication cost and security settings for MPC with maximal fluidity?

(2) Can we design maximally Fluid MPC protocols that meet this bound while approaching the efficiency of the best traditional MPC schemes?

Addressing either question is challenging: no systematic study of Fluid MPC's communication complexity yet exists. In traditional MPC, the efficiency of deterministic protocols (without amortization) is fundamentally constrained by the Dolev-Reischuk bound [14], which establishes the impossibility of achieving Byzantine Agreement with a communication complexity of $o(n^2)$. However, this lower bound is specific to malicious adversaries within a static committee. It does not extend to the Fluid MPC setting, where committees are dynamic, nor does it apply when the adversary is restricted to semi-honest behavior. Bienstock *et al.* [1] observation, that transferring shares between committees under dishonest majority costs $O(n^2)$ communication in the preprocessing phase, hints that super-linear overhead may be unavoidable, but no formal lower bound for other settings is known, such as honest majority. Closing this gap by establishing theoretical limits and developing nearly optimal protocols would directly benefit large-scale privacy-preserving applications.

1.1 Our Contributions

We advance towards answering these questions by investigating the communication restriction of *secret-sharing-based* Fluid MPC in the *maximal-fluidity* setting.

In particular, we establish an identical communication *cost reduction* from the multiplication gate of Fluid MPC protocols to a secret-sharing scheme that

transmits shares from one committee to the future, which is defined as $\mathcal{F}_{\text{reshare}}$ in [11]. We then show that achieving $\mathcal{F}_{\text{reshare}}$ with $O(n)$ communication complexity requires the corruption threshold t to be strictly limited relative to the security threshold d of the underlying secret-sharing (where d is one less than the reconstruction threshold). In particular, $t < d/2$ is necessary for optimal communication efficiency. This bound implies that sublinear communication is unattainable for any information-theoretically secure Fluid MPC protocol utilizing secret sharing for inter-committee transfers. Finally, we design a maximally fluid protocol for $t < n/4$ that achieves a concrete communication cost comparable to the most efficient traditional MPC protocols.

In more details, our contributions are:

1. *Reduction between Reshare and Fluid MPC.* We formalize the transfer gate intuition from [10], proving that $\mathcal{F}_{\text{reshare}}$ serves as a proxy for the communication bounds of any Fluid MPC protocol based on secret sharing. This result remains valid even for protocols that lack an explicit resharing phase [10,12], providing a generalized framework for evaluating inter-committee communication costs.
2. *Communication Framework & Complexity Trade-off.* Using $\mathcal{F}_{\text{reshare}}$ as a tool, we distinguish between the corruption threshold t and the security threshold d, introducing parameters to model inter-party communication. We show that any $O(n)$-communication protocol must exhibit a specific communication pattern with a cost of $\lambda\gamma(n-\mu)+T(\mu,n)$ field elements[3]. This allows us to formalize the trade-off between communication overhead and the corruption threshold (e.g., $t < (1-1/(\lambda+1))(d+2)$), proving that communication complexity is fundamentally restricted even under semi-honest security.
3. *Optimal protocol.* Under the bound, we propose an *optimal* reshare scheme expanded from [1] and based on Shamir sharing that applies when $t \leq d/2$ and $d < n/2$. Building on this scheme, we construct a maximally fluid MPC protocol that is maliciously secure with abort. The protocol communicates on average $9.3n$ field elements per multiplication in the semi-honest case and $37.3n$ in the malicious case, roughly a six-fold improvement over the current best result [1].
4. *Implementation and evaluation.* Our prototype, written in `C++` and simulated in OMNeT++, confirms our results and shows only moderate overhead compared with traditional MPC protocols, such as ATLAS [20], underscoring the practicality of our approach.

1.2 Related Work

Our paper extends the recent line of Fluid MPC protocols [1,2,7,11,24], which aim to keep communication low even when committees change over time. A broader body of work addresses multiparty computation with dynamic or one-shot participants (which called Maximal fluidity). Furthermore, some works try to achieve high efficiency by relaxing the fluidity [28].

[3] $\lambda, \gamma > 0$ are constants, $\mu = o(n)$, and $T(\mu, n) = O(n)$.

One-round communication. The YOSO (You Only Speak Once) series of protocols, initiated by Gentry *et al.* [19] and expanded in follow-up papers [4, 5, 13], restricts every party to a single outgoing message. Although the adversary model differs from that of Fluid MPC, both frameworks share the goal of scaling to large, unstable networks. David *et al.* [10] bridge the two settings and propose a dynamic-participant MPC protocol that guarantees output delivery. Escudero et al. [15] recently proposed a computational YOSO MPC protocol using packed secret sharing that achieves sub-linear communication for large n. Our results show that while the $O(n)$ lower bound is strict in the information-theoretic setting, it only applies to computational protocols when t is large relative to d and n is below a certain threshold. Thus, the results in [15] circumvents these boundaries, providing an intuitive confirmation of the limits established in this work.

Application-driven variants. Fluid-style techniques have inspired several specialised protocols. Gama *et al.* [17] handle "Delayed Parties" in star-shaped networks, achieving robustness against nodes that come online late. Escudero *et al.* [16] develop locally repairable MPC, which shares the Fluid objective of maintaining progress despite participant churn.

Together, these results highlight the growing interest in MPC schemes that tolerate large, dynamic populations, a direction our work continues by clarifying the fundamental communication limits of Fluid MPC.

2 Technical Overview

This section sketches the core ideas behind our results. We begin by showing how the reshare functionality in [11] captures the communication bound of Fluid MPC. We then state a simplified communication cost bound theorem, illustrated through a concrete $n = 7$ example, highlighting the communication bottleneck. Finally, we outline our matching, efficient Fluid MPC construction.

Throughout this work, computations are performed over a finite field $\mathbb{F}$ of prime order p, and we denote the target circuit as C. The dynamic parties are partitioned into a sequence of committees $\mathcal{C}^1, \mathcal{C}^2, \ldots$ of sizes $n_1, n_2, \ldots$, respectively. Each committee $\mathcal{C}^j$ utilizes a (d_j, n_j) threshold secret-sharing scheme and is assumed to withstand up to t_j corruptions.[4] For ease of exposition, this section assumes a uniform setting where $n_j = n \geq 3$, $d_j = d$, and $t_j = t$ for all j.

2.1 Reshare: A Tool for Fluid MPC Analysis

In [1] it is shown that the only step absent between traditional MPC and Fluid MPC is transferring shares (generated Beaver triple) from $\mathcal{C}^1$ to $\mathcal{C}^3$. Following [1, 11], we call this transfer "reshare". In this work we use the functionality $\mathcal{F}_{\mathsf{reshare}}$ in [11], but with difference for permitting any distance between source and target committees; the secret can be sent by committee $\mathcal{C}^j$ to committee $\mathcal{C}^{j'}$ for $j < j'$.

[4] Specifically, any subset of at least $d_j + 1$ parties can reconstruct the shared secret.

A bit more formally, let $[v]^{j}_{d_j}$ denote the (d_j, n_j) threshold shares of v held by $\mathcal{C}^j$. The functionality moves these shares through the intervening committees so that $\mathcal{C}^{j'}$ ends with an independently randomised $(d_{j'}, n_{j'})$ share $[v]^{j'}_{d_{j'}}$; every party $P_i^{j'} \in \mathcal{C}^{j'}$ receives a fresh share $v_i^{j'}$, and security holds against a malicious adversary with abort.

Reshare as the core of Secret Share-based Fluid MPC. In this work, we demonstrate that the reshare functionality is more than a convenience; it is actually the *communication bottleneck* that separates Fluid MPC from classical MPC protocols. [10] notices that if a secret share-based Fluid MPC protocol is asked to evaluate one multiplication, the resulting transcript *is* a reshare protocol.[5] Inspired by it, we formally prove that any complexity restriction for $\mathcal{F}_{\mathsf{reshare}}$ therefore *strictly* transfers to Fluid MPC protocols of that kind, and any efficient reshare construction immediately yields an equally efficient Fluid protocol. Consequently, if a given communication bound and security setting makes reshare impossible, then any Fluid MPC protocol under the same constraints is also ruled out.

2.2 Treating t and d Separately

Recall that a (d, n) threshold secret sharing scheme tolerates any coalition of up to d colluding parties. Most Shamir-based MPC protocols simply set the corruption threshold t equal to d ($t = d$); additive schemes in a dishonest-majority setting push this to the extreme with $t = d = n - 1$. To analyse communication limits of Fluid MPC, we *decouple* these two parameters. We note that although our framework permits $d \geq t$ (i.e. the reconstruction threshold may exceed the privacy threshold), it *does not* rely on ramp secret-sharing [3]: we assume the adversary learns at most t shares in one parties' committee, so perfect privacy holds for t shares, and no information leaks in the region between t and d shares.

Next, we state a streamlined version of our reshare communication cost restriction under the simplifying assumption that every committee uses identical parameters (n, d, t). Informally, the result shows that "cheap" reshare (and hence a cheap Fluid MPC) forces a strict cap on t.

Simplified theorem. Assume a reshare implementation is allowed only $O(n)$ field elements of communication. Label the final recipient committee $\mathcal{C}^{j'}$. Then, partition its parties into (1) at most $\mu = o(n)$ "heavy" parties that together receive no more than $T(\mu, n) = O(n)$ elements ($T(0, n) = 0$), and (2) $n - \mu$ "light" parties, each of which can receive a single message from each of up to λ parties

[5] More specifically, a secret share based Fluid MPC protocol evaluating a circuit that contains a single multiplication gate that multiplies any shares of value $[x]_d$ by $[1]_d$ is equivalent to reshare the value $[x]_d$. Furthermore, since most existing dynamic-participant MPC protocols are based on secret sharing, we implicitly refer to secret-sharing-based Fluid MPC protocols whenever discussing Fluid MPC.

in $\mathcal{C}^{j'-1}$, with each message containing at most γ field elements, for $\mu \geq 0$ and fixed constants $\lambda, \gamma > 0$. Thus the entire hop from $\mathcal{C}^{j'-1}$ to $\mathcal{C}^{j'}$ costs at most $\lambda\gamma\,(n-\mu) + T(\mu, n)$ field elements. If, in addition,

$$t \;\geq\; \left(1 - \tfrac{1}{\lambda+1}\right)(d+2),$$

then, even against a *semi-honest* adversary,

- no information-theoretically secure reshare exists, and
- no PPT or statistically secure reshare exists whenever
  - $t \geq d$, or
  - $t < d$ and n is below a moderate threshold.[6]

We make a few remarks about the previous theorem statement. First, this theorem actually propose a trade-off between the communication and the adversary corruption threshold t. Intuitively speaking, if we consider schemes that can tolerate a larger proportion of adversary corruptions, the communication structure must be more complex, leading to increased communication overhead. Conversely, to achieve a more efficient communication scheme, the adversary's corruption threshold must not exceed a specific proportion, that is $t \;\geq\; \left(1 - \tfrac{1}{\lambda+1}\right)(d+2)$.

Second, traditionally, the security of MPC protocols with computational security is mainly measured based on the size of the finite field or ring used, and the number of parties n is not included in the security parameter. However, we observe that under certain conditions (especially for PPT or statistical adversaries), the security parameter κ must include information about the size of n. Furthermore, although this informal theorem restricts the number of parties in all committees to be the same, similar conclusions can still be drawn for settings where the number of parties is unrestricted, although the restrictions on inter-committee communication will be more complex. Finally, the proof relies solely on the abstraction $\mathcal{F}_{\mathsf{reshare}}$ and does not assume any particular sharing technique. Treating additive sharing as an $(n-1, n)$ share scheme therefore places the impossibility result in [1] as a special case of our theorem.

This theorem can be directly transferred to secret share based Fluid MPC according to the relationship between it and reshare. Next, we briefly illustrate the core of our proof technique for the special and weakened case where $t = d$ and reshare occurs between two contiguous committees, whereas our main restriction applies to $t < d$ and committees with *any distance*.

Informal and weakened lemma. Consider two disjoint committees $\mathcal{C}^1$ and $\mathcal{C}^2$ with the same parameter settings, no information-theoretically secure protocol can implement $\mathcal{F}_{\mathsf{reshare}}$ between them using only $O(n)$ field-element communication when $t = d$.

[6] As we will discuss later, at least $n \geq 72$ is required for 2^{64} field security.

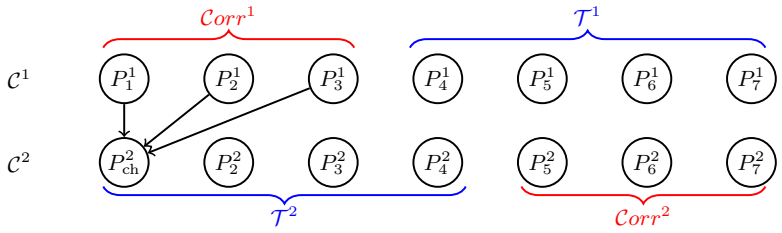

Fig. 1. Message transmission of resharing between two 7-party committees. Three shaded nodes in $\mathcal{C}^1$ send messages to a single light-blue receiver P_{ch} in $\mathcal{C}^2$. The adversary (red) first corrupts those three senders, then corrupts three additional parties in $\mathcal{C}^2$. With the three shares it now holds plus the computable share of P_{ch}, the adversary recovers v.

Proof intuition. A reshare protocol that costs $O(n)$ field elements has room for only linear pass of messages from $\mathcal{C}^1$ to $\mathcal{C}^2$. Consequently, at least one party in the target committee, call it the chosen party P_{ch}, must receive inputs from *no more than* t senders in $\mathcal{C}^1$; otherwise the outgoing traffic from $\mathcal{C}^1$ alone already scales as $\Omega(n(t+1)) = \Omega(n^2)$.

The adversary proceeds in two phases: (1) It waits until the t senders $\mathcal{S}_{\text{ch}}^1 \subseteq \mathcal{C}^1$ have delivered their messages to P_{ch} and then checks if it just corrupted exactly those parties. This choice succeeds with probability $\varepsilon_t = 1/\binom{n}{t} > 0$ when corruption is random. (2) Then the adversary corrupts any t parties in $\mathcal{C}^2 \setminus \{P_{\text{ch}}\}$ with a probability of 1 because the adversary is adaptive. Privacy breaks because P_{ch} is honest, its output share v_{ch}^2 is a deterministic function f of the t incoming messages $\{\boldsymbol{m}_i\}_{P_i^1 \in \mathcal{S}_{\text{ch}}^1}$ and the local randomness $\boldsymbol{r}$ it generates. Notice that v_{ch}^2 is uniquely determined by the other $n-1$ shares, the randomness does not affect the result of f. The adversary, who now holds all the messages $\{\boldsymbol{m}_i\}$, can *compute* $v_{\text{ch}}^2 = f(\{\boldsymbol{m}_i\}, 0^*)$ without corrupting P_{ch}. Adding the t shares obtained in the second round of corruptions gives the adversary $t+1 = d+1$ shares in total, exactly the number needed to reconstruct the secret v. This violates perfect privacy and yields the desired contradiction. Fig. 1 depicts the flow for a concrete example with $n = 7$ and $t = d = 3$.

The lemma above already rules out $O(n)$ communication when $t = d$, even against a semi-honest adversary. It therefore carries over *a fortiori* to the malicious R-adaptive setting used elsewhere in the paper.

2.3 Efficient Fluid MPC Based on Optimal Reshare

Once an $O(n)$ reshare is available (Section 2.1), a fully Maximally-Fluid MPC over Shamir sharing follows almost verbatim: each multiplication consists of one reshare plus $O(n)$ communication [1]. The informal bound, however, shows that such efficiency forces a tight relation between the privacy and reconstruction thresholds: with inter-committee bandwidth capped at $\lambda\gamma\,(n-\mu)+T(\mu,n)$, PPT security for all committee size becomes impossible whenever $t \geq \left(1-\frac{1}{\lambda+1}\right)(d+2)$.

Choosing optimal parameters. For optimal efficiency, we set $\lambda = \gamma = 1,\ \mu = 0$, so the hop from one committee to the next costs *at most* n field elements. Under the bound above, this forces $t \leq d/2$, which we adopt from here on. Our reshare scheme expends the work of [1] and supports every linear (d, n) scheme with $t \leq d/2$ and therefore applies far beyond Shamir secret sharing.

Building the full protocol. With the optimal reshare in place, we construct a Maximally-Fluid MPC that uses Shamir sharing together with the double-random-sharing technique, avoiding the complicated generation of Beaver triple. We re-engineer the random-share generators $\mathcal{F}_{\mathsf{double\text{-}rand}}$ and $\mathcal{F}_{\mathsf{zero}}$ from traditional MPC so that each is produced in batches by linearly combining one random share from every party and is suitable for Fluid setting.

Performance. Against a semi-honest adversary, one multiplication gate completes in three rounds, compared with seven in [1]. The average bandwidth per gate drops to $9.3n$ field elements (semi-honest) and $37.3n$ field elements (malicious). These figures approach the best traditional MPC protocols. A prototype written in `C++` and simulated in OMNeT++ [26] confirms that the overhead remains modest when compared with ATLAS [20].

3 Security Model and Preliminaries

In this section, we introduce some preliminaries needed for our work. We will first present the definition of Fluid MPC, as well as some definitions that we need. Finally, we provide some cryptographic concepts used in this paper.

3.1 Fluid MPC and Security Model

Next, we give a brief introduction on the Fluid MPC concepts from [7]. We discuss it in more details in Appendix A.

We consider a client-server model within a universe $\mathcal{U}$ of all clients and servers, formalizing the concept of dynamic parties. Roughly speaking, clients remain unchanged, while servers are allowed to change continuously during the computation. The entire computation process can be divided into the input stage, execution stage, and output stage. The input stage and output stage are similar to static party MPC, where fixed clients provide inputs during the input stage and receive computation results during the output stage. The execution stage in Fluid MPC is divided into epochs, each executed by a static set of parties, and further divided into the computation phase and hand-off phase. According to the epoch number, we refer to the set of parties executing $epoch_j$ as $\mathcal{C}^j$ and the set of clients as $\mathcal{C}^{\text{clnt}}$. The size for committee $\mathcal{C}^j$ is denoted as n_j while the corruption threshold is denoted as t_j ($t_j = \Theta(n_j)$).

The computing idea mainly follows the 'layered circuit' similar to [7], which constructs the whole arithmetic circuit C by layers. Each layer consists of multiple arithmetic gates. The maximum number of arithmetic gates contained in each layer is called the width of the circuit. In this paper, we set the width w to be at least $\Omega(n)$.

3.2 Preliminaries

Linear (d, n)-threshold secret sharing scheme [25]. The linear (d, n)-threshold secret sharing scheme supports linear combination calculation. Similar with setting of t_j, we set (d_j, n_j)-threshold secret sharing scheme is used in committee $\mathcal{C}^j$, $d_j = \Theta(n_j)$. The algorithms are listed below:

- $\Pi_{\text{LSS}}.\text{Share}(x) \rightarrow [x]_d$: A dealer shares the secret value $x \in \mathbb{F}$ and, to be concise, generates n secret shares $\{x^1, \ldots, x^n\}$, denoted as $[x]_d$.
- $\Pi_{\text{LSS}}.\text{Rec}([x]_d, i) \rightarrow x$: One party P_i is allowed to obtain the original secret x after receiving at least $d + 1$ shares.
- $\Pi_{\text{LSS}}.\text{Open}([x]_d) \rightarrow x$: This procedure allows every party to obtain the original secret x with at least $d + 1$ shares as input.

Moreover, we denote shares of x held by parties in set S as $[x]_S$, and denote shares of x held by committee $\mathcal{C}^j$ as $[x]^j$. Set of trusted honest parties in committee $\mathcal{C}^j$ is denoted as $\mathcal{T}^j$, and the set of corrupted parties is denoted as $\mathcal{C}orr^j$. Furthermore, we denote $\bigcup S$ as the union set of all elements in the set S.

4 Complexity Limitation of the Fluid MPC Based on Threshold Secret Share

First, we recall the definition of the reshare functionality $\mathcal{F}_{\mathsf{reshare}}$ in [11]. The relationship between it and the secret share based Fluid MPC protocol will then be illustrated. Furthermore, with the tool $\mathcal{F}_{\mathsf{reshare}}$, the restriction for Fluid MPC based on threshold secret sharing will be proved under different settings.

4.1 Existential Relationship Between Reshare and Fluid MPC

Firstly, we recall the functionality $\mathcal{F}_{\mathsf{reshare}}$ here, which slightly extends the definition in [11]. In $\mathcal{F}_{\mathsf{reshare}}$, we consider reshare from committee $\mathcal{C}^j$ to $\mathcal{C}^{j'}$ for any $j < j'$. This extension is necessary for restriction theorems about the complexity of Fluid MPC (like the Lemma 1).

Furthermore, considering reshare from one committee to the next defined in [11] is useful for the construction of Fluid MPC protocol, we denotes $\mathcal{F}_{\mathsf{reshare}}$ of two-layers as $\mathcal{F}_{\mathsf{rs-two}}$. Here we only present the functionality of $\mathcal{F}_{\mathsf{rs-two}}$.

Inspired from the transfer gate idea in [10], we present Lemma 1 here to demonstrate the relationship between $\mathcal{F}_{\mathsf{reshare}}$ and Fluid MPC, which illustrates why we use $\mathcal{F}_{\mathsf{reshare}}$ as a tool to analyze Fluid MPC.

Lemma 1 *Let Π_{MPC} be a secret share-based Fluid MPC protocol with maximum-fluidity that computes a multiplication gate in $j' - j + 1$ rounds and has the total communication cost of $F(n)$ field elements, i.e., $\mathcal{C}^j$ has shares of inputs and each party in $\mathcal{C}^{j'}$ receives a share of the computation result. Then, under the same committee setting and adversary capabilities, there exists a protocol π_{reshare} that implements $\mathcal{F}_{\mathsf{reshare}}$ among $\mathcal{C}^j$ to $\mathcal{C}^{j'}$ and has $F(n)$ field elements communication cost.*

Functionality 1: $\mathcal{F}_{\mathsf{reshare}}$

Functionality: $\mathcal{C}^j$ reshares a re-randomised (d_j, n_j)-threshold sharing $[v]^j_{d_j}$ to committee $\mathcal{C}^{j'}, j' > j$.

1. In epoch j, $\mathcal{F}_{\mathsf{reshare}}$ receives $v_i^j, P_i^j \in \mathcal{T}^j$ and reconstructs v. $\mathcal{F}_{\mathsf{reshare}}$ further computes the shares of $[v]^j_{d_j}$ held by corrupted parties in $\mathcal{C}^j$, and sends these shares to the adversary.
2. In epoch j', $\mathcal{F}_{\mathsf{reshare}}$ randomly generates shares of $[v]^{j'}_{d_{j'}}$. $\mathcal{F}_{\mathsf{reshare}}$ firstly sends shares of corrupted parties in $\mathcal{C}^{j'}$ to the adversary and waits for its response.
   - If the adversary replies continue, $\mathcal{F}_{\mathsf{reshare}}$ sends other shares to honest parties in $\mathcal{C}^{j'}$.
   - If the adversary replies abort, $\mathcal{F}_{\mathsf{reshare}}$ sends abort to honest parties.

Functionality 2: $\mathcal{F}_{\mathsf{rs-two}}$ [11]

Functionality: $\mathcal{C}^j$ reshares a re-randomised (d_j, n_j)-threshold sharing $[v]^j_{d_j}$ to committee $\mathcal{C}^{j+1}$.

Proof. To show this, we can construct a protocol π_{reshare} that implements $\mathcal{F}_{\mathsf{reshare}}$ using the existing Π_{MPC} protocol as described next.

1. $\mathcal{C}^j$ invokes Π_{MPC} to compute one multiplication gate Multiply$([v]^j_{d_j}, [1]^j_{d_j})$.
2. $\mathcal{C}^{j'}$ receives computing result $[v]^{j'}_{d_{j'}}$ as the reshare result.

Note that in addition to invoking Π_{MPC}, it is necessary to generate random shares of 1, which can be introduced via input gate or collaboratively produced by all parties, which will introduce n field elements communication cost. However, we argue that the generated shares of 1 can be reused. So if we consider at least $\Omega(n)$ reshare operations, then the evenly distributed consumption for generating shares of 1 will be negligible. Thus, the built π_{reshare} has the same communication cost of $F(n)$ field elements. ■

Therefore, directly from Lemma 1, we obtain the following Corollary 1 that conditions the existence of Fluid MPC on the existence of reshare.

Corollary 1 *Under any fixed execution model, if there exists no protocol π_{reshare} that securely realizes $\mathcal{F}_{\mathsf{reshare}}$ with communication cost at most $F(n)$ (field elements), then no secret-sharing-based Fluid MPC protocol Π_{MPC} can securely evaluate a multiplication gate with communication cost at most $F(n)$.*

4.2 Linear Protocol for $d = t$ is Impossible

Given Corollary 1, the existence of linear protocols can be discussed. We begin with the setting that $d = t$, which is widely used in traditional MPC based on

Shamir secret share [20] or repeated secret share [22]. Firstly, we present Lemma 2 to discuss the situation where every committee has the same settings. The proof mainly follows the idea demonstrated in Section 2.2, which focuses on the last layer of communication between $\mathcal{C}^{j'}$ and $\mathcal{C}^{j'-1}$. According to the information controlled by the adversary, the contradiction will be constructed.

Without loss of generality, we simplify the notation by analyzing an instance of $\mathcal{F}_{\mathsf{reshare}}$ that concludes at committee $\mathcal{C}^2$. This transformation re-indexes the final two committees, replacing $\mathcal{C}^{j'-1}$ and $\mathcal{C}^{j'}$ with $\mathcal{C}^1$ and $\mathcal{C}^2$, respectively.

Lemma 2 *Assume a uniform setting where every committee $\mathcal{C}^j$ with size n, corruption threshold t, and (d, n)-threshold secret sharing of $d = \Theta(n)$. If $t \geq d$, then there exists no protocol π_{reshare} that securely realizes $\mathcal{F}_{\mathsf{reshare}}$ with $O(n)$ communication complexity against a PPT adversary.*

Due to space constraints, the proof of Lemma 2 is deferred to Appendix B.4.

Consequently, we have established that realizing $\mathcal{F}_{\mathsf{reshare}}$ with $O(n)$ communication is impossible when $d = t$ for fixed-size committees. Since the constant-size setting is a special case of the variable-size setting, this impossibility result extends immediately to the general case. We formalize this general impossibility in Corollary 2. Furthermore, building on Claim 2 in the proof of Lemma 2, Corollary 3 characterizes the required communication structure for any valid $O(n)$ reshare protocol.

Corollary 2 *Let each $\mathcal{C}^j$ be a committee using (d_j, n_j)-threshold secret sharing scheme and $d_j = \Theta(n)$. If the corruption threshold satisfies $t_j \geq d_j$, then there exists no protocol π_{reshare} that securely realizes $\mathcal{F}_{\mathsf{reshare}}$ with $O(n)$ communication complexity against a semi-honest PPT adversary.*

Proof. Assuming that there exists a π_{reshare} realizing $\mathcal{F}_{\mathsf{reshare}}$ from $\mathcal{C}^{2-(j'-j)}$ to $\mathcal{C}^2$ with a communication complexity of $O(n)$ field elements. The setting of $t_j > d_j$ is obviously impossible. As for the $t_j = d_j$ situation, similar to the Part 3 in the proof for Lemma 2, we denote the event that $\mathcal{S}_i^1$ is a subset of $\mathcal{C}orr^1$ as A_i. The probability can be computed as $p_i = \Pr[A_i] = \binom{n_1-\lambda}{t_1-\lambda} / \binom{n_1}{t_1}$. This means that the *adaptive* adversary can compromise the security of π_{reshare} with at least $\epsilon_{\mathrm{ad}} \geq p_i$ probability. Moreover, the probability that the *static* adversary compromises the security of π_{reshare} is at least

$$\epsilon_{\mathrm{st}} \geq p_i \cdot p_{\mathrm{st}} = p_i \cdot \frac{\binom{n_1}{t_2}}{\binom{n_2}{t_2}} = \frac{\binom{n_1-\lambda}{t_1-\lambda}\binom{n_1}{t_2}}{\binom{n_1}{t_1}\binom{n_2}{t_2}}.$$

Noting that p_i and ϵ_{st} are both polynomial functions of n_1 and n_2, the attack strategy is suitable for PPT adversaries. ■

Corollary 3 *Assume a $\mathcal{F}_{\mathsf{reshare}}$ implementation is allowed only $O(n)$ field elements of communication. Then, parties in $\mathcal{C}^{j'}$ can be divided into (1) at most $\mu = o(n)$ parties that together receive no more than $T(\mu, n) = O(n)$ elements*

($T(0,n)=0$), and (2) $n-\mu$ parties, each of which can receive a single message from each of up to λ parties in $\mathcal{C}^{j'-1}$, with each message containing at most γ field elements, for $\mu=o(n)\geq 0$ and fixed constants $\lambda,\gamma>0$. Thus the costs of the last communication round (from $\mathcal{C}^{j'-1}$ to $\mathcal{C}^{j'}$) are at most $\lambda\gamma\,(n-\mu)+T(\mu,n)$ field elements.

4.3 $t < d$ is Not Enough

As demonstrated in Corollary 1, Lemma 2 and Corollary 2, the setting of traditional MPC protocols [6, 18, 20], which have $t=d$, is impossible for $O(n)$ Fluid MPC. Moreover, we will illustrate in this section that there are limitations even when $t<d$. We will still begin with the setting where every committee has the same size, n, and show the conclusion for the variable situation later. For the sake of notational simplicity, in this section, we still consider the $\mathcal{F}_{\mathsf{reshare}}$ from the committee $\mathcal{C}^{2-j'+j}$ to $\mathcal{C}^2$

Information-theoretic adversary. Recalling previous proof for Lemma 2, when we consider $d-t>0$, the adversary must corrupts $\mathcal{C}^1$ in an appropriate manner, ensuring that $\mathcal{C}orr^1$ encompasses at least $d-t+1$ of the $\mathcal{S}_i^1\in\mathcal{S}^1$, where $\mathcal{S}_i^1$ means the set of parties in $\mathcal{C}^1$ who send message to P_i^2. This allows the adversary to acquire shares held by $d-t+1$ parties in $\mathcal{C}^2$ without corruption. Subsequently, by corrupting other t honest parties in $\mathcal{C}^2$, the adversary can obtain $d+1=d-t+1+t$ shares, thus compromising the privacy of the protocol. Given that λ represents the number of parties included in $\mathcal{S}_i^1$, it is important to consider the possibility of an adversary successfully controlling them. Consequently, we will discuss different value ranges for t and λ separately:

-*When* $\lfloor\frac{t}{\lambda}\rfloor\leq n-\mu$, we assert that for any constant $\lambda<t$ (it always holds considering $\lambda=O(1)$ while $t=\Theta(n)$), regardless of the communication arrangement generated by π_{reshare} which realizes $\mathcal{F}_{\mathsf{reshare}}$, there exists at least one strategy for the adversary to have non-zero probability to corrupt $\mathcal{C}^1$ such that $\mathcal{C}orr^1$ encompasses at least $\lfloor\frac{t}{\lambda}\rfloor$ of the $\mathcal{S}_i^1\in\mathcal{S}^1$. To achieve this, considering that $\forall i,|\mathcal{S}_i^1|\leq\lambda$, it suffices to arbitrarily select $\lfloor\frac{t}{\lambda}\rfloor$ $\mathcal{S}_i^1$ from $\mathcal{S}^1$, and ensure that $\mathcal{C}orr^1$ contains the union of these selected $\{\mathcal{S}_i^1\}$, which totally has $\lfloor\frac{t}{\lambda}\rfloor\cdot\lambda\leq t$ parties need to be corrupted.

Based on the previous discussion, when $\lfloor\frac{t}{\lambda}\rfloor\geq d-t+1$, the adversary will have the capability to compromise the protocol's privacy. Since $\lfloor\frac{t}{\lambda}\rfloor>\frac{t}{\lambda}-1$, it suffices to ensure that $\frac{t}{\lambda}-1\geq d-t+1$, which is equivalent to:

$$t\geq\frac{d+2}{1+\frac{1}{\lambda}}=\big(1-\tfrac{1}{\lambda+1}\big)(d+2)=\big(1-\tfrac{1}{\lambda+1}\big)d+2\big(1-\tfrac{1}{\lambda+1}\big)\tag{1}$$

It is worth noting that this section requires $t<d$, so we need to check whether the above condition can be satisfied. We point out that the condition becomes unsatisfiable, leading to $d\leq\big(1-\frac{1}{\lambda+1}\big)(d+2)$, if and only if $2\lambda\geq d$, which is clearly impossible. Thus, in this situation (including the setting about the value of λ and μ), an information-theoretical security protocol π_{reshare} to realize

Table 1. Minimal n to realize $O(n)$ costs $\mathcal{F}_{\mathsf{reshare}}$ against PPT R-*static* adversary

Adversary's knowledge of realising $\mathcal{F}_{\mathsf{reshare}}$	Security levels		
	64-bit	128-bit	256-bit
No information	72	142	280
Fully deterministic	No security		
Completely random	> 1000	-	-

Note: This table shows the minimal committee size to realize $O(n)$ costs $\mathcal{F}_{\mathsf{reshare}}$ against PPT R-*static* adversary with different communication information known by adversary for the last round when $t \geq (1-\frac{1}{\lambda+1})(d+2)$, specially for $\lambda = 1$, $\mu = 0$ and $d = n/2$. 'No information' refers to the restriction regardless of the strategy used. And adversary has no information of that. 'Fully deterministic' means that the communication strategy between two committees is completely determined and public. Note that this is the strategy employed by all existing Fluid MPC schemes, and under this strategy, no level of security can be achieved. 'Completely random' means that each party selects the recipient of their message randomly. Adversary can know this random strategy but has no idea of the exact transmit ways. In this case, until $n = 1000$, the adversary's success rate remains above 0.4, making it extremely difficult to achieve even the most basic level of security.

$\mathcal{F}_{\mathsf{reshare}}$ is impossible. In particular, when $\lambda = 1$, meaning that parties in $\mathcal{L}$ receive message inputs from at most one party in $\mathcal{C}^1$, we have $t \geq \frac{d}{2} + 1$. Considering t is an integer, the constraint condition will be $t > \frac{d}{2}$.

-When $n - \mu < \lfloor \frac{t}{\lambda} \rfloor$, this situation can only arise when $\lambda = 1$ and $n - \mu < t$. Specifically, there will be $n-\mu$ parties in $\mathcal{C}^2$ can only receive single message from single party in $\mathcal{C}^1$. The adversary can just randomly corrupt t parties in each committee. Then it has $\binom{n-\mu}{t-\mu}/\binom{n}{t}$ probability to get all shares hold by parties in $\mathcal{C}^2 \backslash \mathcal{L}$ and $t - \mu$ more shares, only has no idea of $n - t < \mu$ parties' shares. Notice each of these $n - t < \mu$ parties can only receive single message from single party in $\mathcal{C}^1$. However, adversary also do random corruption in $\mathcal{C}^1$, it has $\binom{n-(n-t)}{t-(n-t)}/\binom{n}{t} > \binom{n-\mu}{t-\mu}/\binom{n}{t}$ probability to just control all inputs of these $n - t$ parties and gets there shares. So, overall, even static PPT adversary can get all shares hold by $\mathcal{C}^2$ with at least $\left(\binom{n-\mu}{t-\mu}/\binom{n}{t}\right)^2$ probability, which is obviously unsafe. For the sake of simplicity, the following discussions in this paper will *no longer consider* the analysis for the case when $n - \mu < \lfloor \frac{t}{\lambda} \rfloor$.

PPT adversary. Moreover, if we take the PPT adversary into consideration, we need to further determine the probability of the adversary successfully executing the attack when $t \geq (1 - \frac{1}{\lambda+1})(d+2)$. As mentioned in footnote 11, the message transfer strategy of π_{reshare} to realize $\mathcal{F}_{\mathsf{reshare}}$ will influence the best corruption strategy of the adversary. Thus, we have to construct the proof by adversary's knowledge of π_{reshare}'s strategy.

Specifically, we consider three types of adversary's knowledge for 'no information', 'fully deterministic' and 'completely random'. For brevity, illustrations of the adversary successfully attacking with information of these strategies are provided in Appendix B.1.

Finally, based on the given attack success probabilities for PPT adversaries under different settings, we can list the minimum number of parties n required to achieve different security levels. These are shown in Table 1. In which we

consider settings of $\lambda = 1$, $\mu = 0$ and $d = n/2$ (or $d = (n-1)/2$ with odd n) for $t \geq \left(1 - \frac{1}{\lambda+1}\right)(d+2)$, which implies $t = d/2$ under *static* adversary. Considering that as the adversary's capability increases with the proportion of t to n increases, the scheme will become increasingly difficult to resist the adversary's attacks, requiring a larger n to achieve the same security level. Therefore, the values of n in Table 1 are representative. It is worth noting that the 'fully deterministic' strategy used by existing schemes [1, 2, 7, 24], where the adversary knows all parties' transmission targets during the hand-off phase, *cannot* resist PPT adversaries for arbitrary values of n.

4.4 Restriction for $O(n)$ Complexity $\mathcal{F}_{\mathsf{reshare}}$ and Fluid MPC

We have fully demonstrated the limitations for designing $\mathcal{F}_{\mathsf{reshare}}$. In this section, we will conclude the discussion shown before and further illustrate the situation of Fluid MPC protocol.

Restriction for $O(n)$ complexity $\mathcal{F}_{\mathsf{reshare}}$. Assuming that π_{reshare} is a procedure to realize $\mathcal{F}_{\mathsf{reshare}}$, we give the Theorem 1 of limitations for designing π_{reshare}. We assume the number of parties to a linear arrange of a constant number n, i.e., $\forall j \in [D], an \leq n_j \leq bn$, where a and b are two constant numbers. The proof is shown in Appendix B.2.

Theorem 1 *If the communication cost of realizing $\mathcal{F}_{\mathsf{reshare}}$ is required to be $O(n)$. Then, the parties in the last transmitted committee $\mathcal{C}^{j'}$ can be divided into two categories. One category consists of $\mu = o(n) \geq 0$ parties. These parties can receive, without restriction, a total of no more than $T(\mu, n_{j'-1}, n_{j'}) = O(n)$ field elements from any parties in the $\mathcal{C}^{j'-1}$, and $T(0, n_{j'-1}, n_{j'}) = 0$. The other category consists of the remaining $n_{j'} - \mu$ parties, and each party can receive at most γ field elements from each of up to λ parties in the $\mathcal{C}^{j'-1}$, for fixed constants $\lambda, \gamma > 0$.*

Thus the entire communication from $\mathcal{C}^{j'-1}$ to $\mathcal{C}^{j'}$ costs at most $\lambda\gamma(bn - \mu) + T(\mu, n_{j'-1}, n_{j'})$ field elements. If, in addition, $t_{j'} + \frac{t_{j'-1}}{\lambda} \geq d_{j'} + 2$ (equivalent to $t \geq (1 - \frac{1}{\lambda+1})(d_{j'} + 2)$ when $t_{j'} = t_{j'-1} = t$), then, even against a static semi-honest adversary,

- *no information-theoretically secure $\mathcal{F}_{\mathsf{reshare}}$ implementation exists, and*
- *no PPT secure $\mathcal{F}_{\mathsf{reshare}}$ implementation exists whenever 1.$t_{j'} \geq d_{j'}$, or 2.$t_{j'} < d_{j'}$ and n is below a moderate threshold.*

Restriction for $O(n)$ Fluid MPC. Now, using Corollary 1 along with Theorem 1, we can derive Theorem 2 that provides conditional constraints for designing Fluid MPC with $O(n)$ complexity.

Theorem 2 *If the communication cost of secret share based protocol Π_{MPC} to realize multiplication gate of $\mathcal{F}_{\mathsf{DABB}}$ is required to be $O(n)$. Without loss of generality, we consider that committee $\mathcal{C}^j$ has the input of the multiplication gate*

and executes the protocol to let $\mathcal{C}^{j'}$ gain the result shares. Then, the parties in $\mathcal{C}^{j'}$ can be divided into two categories. One category consists of $\mu = o(n) \geq 0$ parties. These parties can receive, without restriction, a total of no more than $T(\mu, n_{j'-1}, n_{j'}) = O(n)$ field elements from any parties in the $\mathcal{C}^{j'-1}$, and $T(0, n_{j'-1}, n_{j'}) = 0$. The other category consists of the remaining $n_{j'} - \mu$ parties, and each party can receive at most γ field elements from each of up to λ parties in the $\mathcal{C}^{j'-1}$, for fixed constants $\lambda, \gamma > 0$.

Thus the entire communication from $\mathcal{C}^{j'-1}$ to $\mathcal{C}^{j'}$ costs at most $\lambda\gamma(bn - \mu) + T(\mu, n_{j'-1}, n_{j'})$ field elements. If, in addition, $t_{j'} + \frac{t_{j'-1}}{\lambda} \geq d_{j'} + 2$ (equivalent to $t \geq (1 - \frac{1}{\lambda+1})(d_{j'} + 2)$ when $t_{j'} = t_{j'-1} = t$), then, even against a static semi-honest adversary,

- *no information-theoretically secure Π_{MPC} exists, and*
- *no PPT secure Π_{MPC} exists whenever 1. $t_{j'} \geq d_{j'}$, or 2. $t_{j'} < d_{j'}$ and n is below a moderate threshold.*

Proof. As illustrated in Corollary 1, if we consider the multiply computation of Π_{MPC}, then it can be constructed to a procedure π_{reshare} that can realize $\mathcal{F}_{\mathsf{reshare}}$ with same communication cost between each layer from committee $\mathcal{C}^j$ to $\mathcal{C}^{j'}$. According to the restrictions in Theorem 1 for π_{reshare}, these conditions are also applicable to Π_{MPC}. ■

5 Efficient Fluid MPC Protocol Based on Random Double Sharings

Recall the idea presented in the introduction: we want to design a Fluid MPC protocol meets the bound while approaching the efficiency of the best traditional MPC schemes. We design an efficient Fluid MPC based on Shamir secret share and random double sharing. It can be seen that the protocol is optimal when $\lambda = 1, \mu = 0$ and $\gamma = 1$, considering that $\lambda \geq 1$ and $\mu \geq 0$. In these conditions, $t < d/2 + 1$ must be meet to avoid the bound for realizing $\mathcal{F}_{\mathsf{DABB}}$. Therefore, to realize the efficiency, we propose a Fluid MPC protocol for $\lambda = 1, \mu = 0, \gamma = 1$, and $t \leq d/2$. This protocol achieves maximal Fluidity with the constant number of parties in each committee. We assume that $\forall j, |\mathcal{C}^j| = n$ and use (d, n)-Shamir secret sharing, where $d < n/2$. We will begin with basic constructions and then demonstrate the final protocol later.

5.1 Random Sharings Generation

At first, in the Fluid MPC protocol and reshare scheme proposed in this paper, random double sharings $[r]_d, [r]_{2d}$ are needed for the multiplication scheme. At the same time, the random zero sharings $[0]_d$ are also needed for the construction of the $\mathcal{F}_{\mathsf{reshare}}$. Therefore, to be rigorous, we first present the implementations of two functionalities $\mathcal{F}_{\mathsf{double-rand}}$ and $\mathcal{F}_{\mathsf{zero}}$. These concepts draw parallels with those presented in [1]. Thus, we only briefly demonstrate usages without detailed definitions, and we refer readers to their work for formal definitions. Procedures that implement these two functionalities are shown in Appendix B.3.

Functionality 3: $\mathcal{F}_{\mathsf{double-rand}}$ [1]

Functionality: Distribute degree-d and degree-$2d$ random shares of the same random value r to $\mathcal{C}^j$.

Functionality 4: $\mathcal{F}_{\mathsf{zero}}$ [1]

Functionality: Distribute degree $2d$ random shares of $o = 0$ to $\mathcal{C}^j$.

5.2 Efficient Reshare for Shamir Secret Sharing

Given limitations in Section 4 and Theorem 1, we propose a procedure π_{reshare} for resharing $[r]_d$ using a (d, n)-Shamir threshold secret sharing, under the optimal conditions $t \leq \frac{d}{2}$ and $\lambda = 1, \gamma = 1, \mu = 0$. According to the Corollary 3 and Theorem 1, these settings contribute to the theoretically optimal communication cost for the implementations of $\mathcal{F}_{\mathsf{reshare}}$. In brief, to realize reshare, we add a random sharing of zero $[0]_d$ to $[r]_d$ and let the parties transfer their shares one-to-one. Since $t \leq d/2$, the adversary can get at most d shares, which ensures the security of reshare. We first propose the $\pi_{\mathsf{rs-two}}$ for the two-layers situation [7]. Then, π_{reshare} is built by repeating execute $\pi_{\mathsf{rs-two}}$. We give the Lemma 3 to demonstrate the proposed procedure can securely compute functionality $\mathcal{F}_{\mathsf{reshare}}$ without checking the correct of output, which actually achieves *weak privacy*.

Lemma 3 *The procedure π_{reshare} can securely compute the functionality $\mathcal{F}_{\mathsf{reshare}}$ with Weak Privacy in the $\mathcal{F}_{\mathsf{zero}}$-hybrid model in the presence of a malicious R-adaptive adversary and committees have same settings of t, d, n and $t \leq d/2$.*

The proof of Lemma 3 is shown in Appendix B.5. Furthermore, a correctness verification stage must be considered to achieve the *secure with abort*. We use the robust circuit idea designed in [7], which is informally described below.

1. All parties in $\mathcal{C}^j$ have random share $[r]_d^j$ and inputs $\{[v]_d^j, [rv]_d^j\}$.
2. All parties in $\mathcal{C}^j$ use π_{reshare} to reshare $[v]_d^j, [rv]_d^j$ and $[r]_d$ separately.
3. Open the random number r, then all parties open share $[dis]_d = [rv]_d^{j'} - r \cdot [v]_d^{j'}$ and check if $dis = 0$.

Consider that the r is not known to the adversary before it is opened, the probability of the adversary cheating successfully is statistically negligible. Moreover, although the stages described before can defeat a malicious adversary, we do not need to do the checking for every reshare implementation separately. The check for all gates and reshares can be done together with the robust circuit idea before the output stage of the Fluid MPC protocol.

[7] It is worth noting that $\pi_{\mathsf{rs-two}}$ bears some similarities to the work in [1]. However, the scheme proposed in this paper applies to all linear secret sharing schemes and is proven to be theoretically optimal.

Procedure 1: $\pi_{\mathsf{rs-two}}$

Usage: $\mathcal{C}^j$ reshare a re-randomized (d, n)-threshold share $[v]_d^j$ to committee $\mathcal{C}^{j+1}$.

1. $\mathcal{C}^j$ invoke $\mathcal{F}_{\mathsf{zero}}$ to get a random zero sharing $[0]_d^j$.
2. The parties of $\mathcal{C}^j$ locally compute $[v']_d^j = [v]_d^j + [0]_d^j$.
3. $\forall P_i^{j-1} \in \mathcal{C}^{j-1}$, each party sends v'^j_i to $P_i^{j+1} \in \mathcal{C}^{j+1}$.
4. $\mathcal{C}^{j+1}$ output $[v']_d^{j+1}$.

Procedure 2: π_{reshare}

Usage: $\mathcal{C}^j$ reshare a re-randomized (d, n)-threshold share $[v]_d^j$ to committee $\mathcal{C}^{j'}$.

1. For k in $[j, j'-1]$, $\mathcal{C}^k$ invokes $\pi_{\mathsf{rs-two}}$ to reshare the $[v]_d^k$ to $[v]_d^{k+1}$
2. $\mathcal{C}^{j'}$ outputs $[v]_d^{j'}$.

5.3 Secure Multiplication Based on Shamir Secret Sharing

Using the procedures we presented before, in this section, we propose an efficient multiplication procedure π_{mult} for Fluid MPC. Overall, this procedure originates from a series of Shamir MPC multiplication protocols [8, 20, 21] based on P_{king} and random double sharings. However, in the Fluid MPC settings, random double sharings $[r]_d^j, [r]_{2d}^j$ are generated in real time and used to mask multiplication results in $\mathcal{C}^j$. After masking, degree-d sharings $[r]_d^j$ used for unmasking need to be reshared to $\mathcal{C}^{j+2}$.

Procedure 3: π_{mult}

Usage: $\mathcal{C}^j$ hold $[x]_d^j, [y]_d^j$, and $\mathcal{C}^{j+2}$ output $[x \cdot y]_d^{j+2}$.

1. $\mathcal{C}^j$ invokes $\pi_{\mathsf{double-rand}}$ to obtain $[r]_d^j, [r]_{2d}^j$.
2. The parties of $\mathcal{C}^j$ locally compute
$$[e]_{2d}^j = [x]_d^j \cdot [y]_d^j + [r]_{2d}^j.$$
Then, all parties in $\mathcal{C}^j$ jointly select a special party P_{king} in $\mathcal{C}^{j+1}$. Parties send $[e]_{2d}^j$ to P_{king} and invoke π_{reshare} so that $\mathcal{C}^{j+2}$ get $[r]_d^{j+2}$.
3. $P_{king} \in \mathcal{C}^{j+1}$ locally runs $\Pi_{\mathrm{LSS}}.\mathrm{Rec}([e]_{2d}^j)$ to obtain $e = x \cdot y + r$ and broadcasts e to $\mathcal{C}^{j+2}$.
4. All parties in $\mathcal{C}^{j+2}$ locally compute $[x \cdot y]_d^{j+2} = e - [r]_d^{j+2}$.

It is worth noting that π_{mult} operates with maximal Fluidity, and all communications can be completed in a one-way manner. The computations of the committees are independent of each other and can be performed simultaneously.

5.4 Efficient Fluid MPC Protocol

With all the preparation, we present our efficient Fluid MPC protocol in this section, which achieves maximal fluidity and is secure against the *malicious R-adaptive* adversary. The proposed protocol assumes that all committees have the same number of parties and use the same (d, n)-threshold secret sharing scheme. Similar to the idea of [7], we will first present the *weak privacy* protocol and then modify it to the malicious security. The *weak privacy* conception for Fluid MPC was defined in [7], which protects privacy in a malicious R-adaptive adversary environment without guaranteeing the correctness of the output.

Weak Privacy Scheme The Protocol Π_{wpmain} demonstrated here achieves *Weak Privacy* under *malicious R-adaptive adversary.*

Protocol 4: Π_{wpmain}

Input: $\forall P_i \in \mathcal{C}^{\text{clnt}}$, if P_i wants to generate a Shamir sharing of input x_i, then it executes $\Pi_{\text{LSS}}.\text{Share}(x_i)$ to get $[x_i]_d^{\text{clnt}}$.

Computing: Each committee will sequentially compute the arithmetic gates contained in the corresponding layer of circuit C. Each layer of the circuit will involve three committees, executing the following for different arithmetic gates:

Linear combination: To compute linear combination of $([x_1]_d^j, \cdots, [x_\ell]_d^j)$ and $\boldsymbol{a}^T$, all parties in committee $\mathcal{C}^j$ locally obtain $[z]_d^j = [a_1 x_1 + \cdots + a_\ell x_\ell]_d^j = ([x_1]_d^j, \cdots, [x_\ell]_d^j) \cdot \boldsymbol{a}^T$. Then, all parties in committee $\mathcal{C}^j$ invoke π_{reshare} to let committee $\mathcal{C}^{j+2}$ get $[z]_d^{j+2}$.

Multiplication: To compute multiplication of $[x]_d^j$ and $[y]_d^j$, committee $\mathcal{C}^j$ invokes π_{mult} on them to let committee $\mathcal{C}^{j+2}$ get the result $[z]_d^{j+2} = [x \cdot y]_d^j$.

Output: Parties in the last committee $\mathcal{C}^{last}$ open the circuit result shares $[\boldsymbol{z}]_d^{last} = ([z_1]_d^{last}, \cdots)$ to the clients. Clients then reconstruct them and output $\boldsymbol{z}$.

Theorem 3 *Let $\mathcal{A}$ be a malicious R-adaptive adversary, protocol Π_{wpmain} can compute $\mathcal{F}_{\mathsf{DABB}}$ with Weak Privacy in the presence of $\mathcal{A}$ in the $\mathcal{F}_{\mathsf{commit}}$-hybrid model, controlling at most t servers in each epoch and d clients.*

For the sake of simplicity, the proof is not shown here, interested readers can refer to Appendix B.6.

Malicious Security With Abort. Finally, considering that Π_{wpmain} is a linear-based Fluid MPC protocol, according to the *Malicious Security Compiler* and Theorem 2 proposed in CGG$^+$21 [7], Π_{wpmain} can achieve **Malicious R-Adaptive Security With Abort** under the same fluidity conditions by computing a robust version circuit $\widetilde{C}$ of the layered arithmetic circuit C, in which $\widetilde{C}$ has one more layer than C and mostly with its width increasing to at most $4w + 4$ compared to the width w of C.

5.5 Efficiency Analysis and Implementation Comparison.

We primarily consider the communication complexity (measured in the number of field elements) required to compute a single multiplication gate under both semi-honest and malicious R-adaptive adversary settings for these three protocols. It is worth noting that BEP23 [1] does not provide specific implementations for functionalities such as $\mathcal{F}_{\mathsf{double-rand}}$ and $\mathcal{F}_{\mathsf{zero}}$. In the following comparison, we assume that their implementation is similar to what is proposed in this work.

Table 2. Average Communication Cost for Single Multiplication Gate Evaluation

Protocols	Types	Threshold	Semi-honest	Malicious
CGG$^+$21 [7]	Fluid	$t < n/2$	n^2	$4n^2$
BEP23 [1]		$t < n/2$	$60n$	$232n$
BEP24 [2]*		$t < n/3$	-	$O(n)$
Our Work		$t < n/4$	$9.3n(9\frac{1}{3}n)$	$37.3n(37\frac{1}{3}n)$
ATLAS [20]	Traditional	$t < n/2$	$4n$	$4n$

* Notice the work BEP24 [2] with $t < n/3$ threshold aims for high security (perfectly secure) and uses similar computation technic with BEP23 [1], we omit the specific communication cost.

As shown in Table 2, both our work and CGG$^+$21 [7] utilize robust arithmetic circuits to achieve malicious security with abort. The size of the robust circuit $\widetilde{C}$ scales proportionally to the original circuit C, depending on the specific structure of C. To compare the efficiency of our protocol, we consider the worst-case scenario, where a multiplication gate, when converted to a robust arithmetic circuit, results in a computational load equivalent to 4 multiplication gates. Even in this scenario, and ignoring constant-level communication costs (which are significantly higher in [1] than in our protocol), our protocol demonstrates a significant efficiency advantage. Compared to BEP23 [1], our protocol incurs only 1/6 of its communication overhead for both semi-honest and malicious security.

To emphasize the efficiency advantages of the protocol proposed in this paper, we further conduct a simulation comparison with ATLAS [20], the state-of-the-art traditional MPC scheme. Interested readers are referred to Appendix C for details.

6 Conclusion

In this work, we studied the communication complexity of Fluid MPC in the maximally fluid setting and identified resharing as the key abstraction governing its cost. By formalizing the reduction from Fluid MPC multiplication to reshare, we obtained a communication resilience trade-off with security settings. Motivated by this bound, we then present concrete Fluid MPC constructions with linear communication per multiplication gate, obtaining improvements over prior work in both the semi honest and malicious with abort settings.

Acknowledgments. This work was supported by the National Cryptologic Science Fund of China (2025NCSF02022), the National Natural Science Foundation of China (62372020).

References

1. Bienstock, A., Escudero, D., Polychroniadou, A.: On linear communication complexity for (maximally) fluid MPC. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology – CRYPTO 2023, Part I. Lecture Notes in Computer Science, vol. 14081, pp. 263–294. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 20–24, 2023). https://doi.org/10.1007/978-3-031-38557-5_9
2. Bienstock, A., Escudero, D., Polychroniadou, A.: Perfectly secure fluid MPC with abort and linear communication complexity. IACR Communications in Cryptology **1**(4) (2025). https://doi.org/10.62056/aesg89n4e
3. Blakley, G.R., Meadows, C.: Security of ramp schemes. In: Blakley, G.R., Chaum, D. (eds.) Advances in Cryptology – CRYPTO'84. Lecture Notes in Computer Science, vol. 196, pp. 242–268. Springer Berlin Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 1984). https://doi.org/10.1007/3-540-39568-7_20
4. Campanelli, M., David, B., Khoshakhlagh, H., Konring, A., Nielsen, J.B.: Encryption to the future - A paradigm for sending secret messages to future (anonymous) committees. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology – ASIACRYPT 2022, Part III. Lecture Notes in Computer Science, vol. 13793, pp. 151–180. Springer, Cham, Switzerland, Taipei, Taiwan (Dec 5–9, 2022). https://doi.org/10.1007/978-3-031-22969-5_6
5. Cascudo, I., David, B., Garms, L., Konring, A.: YOLO YOSO: Fast and simple encryption and secret sharing in the YOSO model. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology – ASIACRYPT 2022, Part I. Lecture Notes in Computer Science, vol. 13791, pp. 651–680. Springer, Cham, Switzerland, Taipei, Taiwan (Dec 5–9, 2022). https://doi.org/10.1007/978-3-031-22963-3_22
6. Chida, K., Genkin, D., Hamada, K., Ikarashi, D., Kikuchi, R., Lindell, Y., Nof, A.: Fast large-scale honest-majority MPC for malicious adversaries. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018, Part III. Lecture Notes in Computer Science, vol. 10993, pp. 34–64. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 19–23, 2018). https://doi.org/10.1007/978-3-319-96878-0_2
7. Choudhuri, A.R., Goel, A., Green, M., Jain, A., Kaptchuk, G.: Fluid MPC: Secure multiparty computation with dynamic participants. In: Malkin, T., Peikert, C. (eds.) Advances in Cryptology – CRYPTO 2021, Part II. Lecture Notes in Computer Science, vol. 12826, pp. 94–123. Springer, Cham, Switzerland, Virtual Event (Aug 16–20, 2021). https://doi.org/10.1007/978-3-030-84245-1_4

8. Damgård, I., Nielsen, J.B.: Scalable and unconditionally secure multiparty computation. In: Menezes, A. (ed.) Advances in Cryptology – CRYPTO 2007. Lecture Notes in Computer Science, vol. 4622, pp. 572–590. Springer Berlin Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2007). https://doi.org/10.1007/978-3-540-74143-5_32
9. Damgård, I., Pastro, V., Smart, N.P., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Safavi-Naini, R., Canetti, R. (eds.) Advances in Cryptology – CRYPTO 2012. Lecture Notes in Computer Science, vol. 7417, pp. 643–662. Springer Berlin Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2012). https://doi.org/10.1007/978-3-642-32009-5_38
10. David, B., Deligios, G., Goel, A., Ishai, Y., Konring, A., Kushilevitz, E., Liu-Zhang, C.D., Narayanan, V.: Perfect MPC over layered graphs. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology – CRYPTO 2023, Part I. Lecture Notes in Computer Science, vol. 14081, pp. 360–392. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 20–24, 2023). https://doi.org/10.1007/978-3-031-38557-5_12
11. David, B., Mondal, A., Satish, R.: Rumors MPC: GOD for dynamic committees, low communication via constant-round chat. In: Hanaoka, G., Yang, B. (eds.) Advances in Cryptology - ASIACRYPT 2025 - 31st International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, VIC, Australia, December 8-12, 2025, Proceedings, Part V. Lecture Notes in Computer Science, vol. 16249, pp. 102–132. Springer (2025). https://doi.org/10.1007/978-981-95-5116-3_4
12. Deligios, G., Konring, A., Liu-Zhang, C., Narayanan, V.: Statistical layered MPC. In: Boyle, E., Mahmoody, M. (eds.) Theory of Cryptography - 22nd International Conference, TCC 2024, Milan, Italy, December 2-6, 2024, Proceedings, Part IV. pp. 362–394. Lecture Notes in Computer Science, Springer (2024). https://doi.org/10.1007/978-3-031-78023-3_12
13. Dettling, G., Liu-Zhang, C., Masserova, E., Rambaud, M., Urban, A.: Broadcast for dynamic committees without trusted setup. IACR Cryptol. ePrint Arch. **2025**, 2078 (2025)
14. Dolev, D., Reischuk, R.: Bounds on information exchange for byzantine agreement. J. ACM **32**(1), 191–204 (1985). https://doi.org/10.1145/2455.214112
15. Escudero, D., Masserova, E., Polychroniadou, A.: Towards scalable YOSO MPC via packed secret-sharing. In: Hanaoka, G., Yang, B. (eds.) Advances in Cryptology - ASIACRYPT 2025 - 31st International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, VIC, Australia, December 8-12, 2025, Proceedings, Part V. pp. 68–101. Lecture Notes in Computer Science, Springer (2025). https://doi.org/10.1007/978-981-95-5116-3_3
16. Escudero, D., Tjuawinata, I., Xing, C.: On information-theoretic secure multiparty computation with local repairability. In: Tang, Q., Teague, V. (eds.) Public-Key Cryptography - PKC 2024 - 27th IACR International Conference on Practice and Theory of Public-Key Cryptography, Sydney, NSW, Australia, April 15-17, 2024, Proceedings, Part II. Lecture Notes in Computer Science, vol. 14602, pp. 205–239. Springer (2024). https://doi.org/10.1007/978-3-031-57722-2_7
17. Gama, M., Beni, E.H., Orsini, E., Smart, N.P., Zajonc, O.: MPC with delayed parties over star-like networks. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023, Part I. Lecture Notes in Computer Science, vol. 14438, pp. 172–203. Springer, Singapore, Singapore, Guangzhou, China (Dec 4–8, 2023). https://doi.org/10.1007/978-981-99-8721-4_6

18. Gennaro, R., Rabin, M.O., Rabin, T.: Simplified vss and fast-track multiparty computations with applications to threshold cryptography. In: Proceedings of the Seventeenth Annual ACM Symposium on Principles of Distributed Computing. p. 101–111. PODC '98, Association for Computing Machinery, New York, NY, USA (1998). https://doi.org/10.1145/277697.277716
19. Gentry, C., Halevi, S., Krawczyk, H., Magri, B., Nielsen, J.B., Rabin, T., Yakoubov, S.: YOSO: You only speak once - secure MPC with stateless ephemeral roles. In: Malkin, T., Peikert, C. (eds.) Advances in Cryptology – CRYPTO 2021, Part II. Lecture Notes in Computer Science, vol. 12826, pp. 64–93. Springer, Cham, Switzerland, Virtual Event (Aug 16–20, 2021). https://doi.org/10.1007/978-3-030-84245-1_3
20. Goyal, V., Li, H., Ostrovsky, R., Polychroniadou, A., Song, Y.: ATLAS: Efficient and scalable MPC in the honest majority setting. In: Malkin, T., Peikert, C. (eds.) Advances in Cryptology – CRYPTO 2021, Part II. Lecture Notes in Computer Science, vol. 12826, pp. 244–274. Springer, Cham, Switzerland, Virtual Event (Aug 16–20, 2021). https://doi.org/10.1007/978-3-030-84245-1_9
21. Goyal, V., Song, Y., Zhu, C.: Guaranteed output delivery comes free in honest majority MPC. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology – CRYPTO 2020, Part II. Lecture Notes in Computer Science, vol. 12171, pp. 618–646. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 17–21, 2020). https://doi.org/10.1007/978-3-030-56880-1_22
22. Mohassel, P., Rindal, P.: ABY^3: A mixed protocol framework for machine learning. In: Lie, D., Mannan, M., Backes, M., Wang, X. (eds.) ACM CCS 2018: 25th Conference on Computer and Communications Security. pp. 35–52. ACM Press, Toronto, ON, Canada (Oct 15–19, 2018). https://doi.org/10.1145/3243734.3243760
23. Nordholt, P.S., Veeningen, M.: Minimising communication in honest-majority MPC by batchwise multiplication verification. In: Preneel, B., Vercauteren, F. (eds.) ACNS 2018: 16th International Conference on Applied Cryptography and Network Security. Lecture Notes in Computer Science, vol. 10892, pp. 321–339. Springer, Cham, Switzerland, Leuven, Belgium (Jul 2–4, 2018). https://doi.org/10.1007/978-3-319-93387-0_17
24. Rachuri, R., Scholl, P.: Le mans: Dynamic and fluid MPC for dishonest majority. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology – CRYPTO 2022, Part I. Lecture Notes in Computer Science, vol. 13507, pp. 719–749. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 15–18, 2022). https://doi.org/10.1007/978-3-031-15802-5_25
25. Shamir, A.: How to share a secret. Commun. ACM **22**(11), 612–613 (1979). https://doi.org/10.1145/359168.359176
26. Varga, A.: OMNeT++, pp. 35–59. Springer Berlin Heidelberg, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12331-3_3
27. Yao, A.C.: How to generate and exchange secrets (extended abstract). In: 27th Annual Symposium on Foundations of Computer Science. pp. 162–167. IEEE Computer Society, Los Alamitos, CA, USA (1986). https://doi.org/10.1109/SFCS.1986.25
28. Zeng, Y., Yang, K., Feng, D., Zhang, M.: Ion: Concretely efficient submaximal-fluid MPC with linear communication. IACR Cryptol. ePrint Arch. **2025**, 1508 (2025), https://eprint.iacr.org/2025/1508

Appendix

A Fluid MPC and Security Model

Here, we separately introduce conceptions of fluid MPC and various dimensional definitions of the adversary $\mathcal{A}$ and corruption, referring readers to [1, 7, 24] for complete definitions.

A.1 Fluid MPC Model

Fluidity. This is defined as the minimum number of synchronous communication rounds in any epoch during the execution stage. We refer to a protocol as achieving maximal Fluidity if each epoch j lasts only one communication round, i.e., no interaction exists in the computation phase. In this paper, consistent with [1, 7, 24], we focus exclusively on maximal Fluidity.

Committee Formation. We consider that every committee is determined 'On-the-fly', which means that committee $\mathcal{C}^{j+1}$ is somehow determined and known to every party $P_i^j \in \mathcal{C}^j$ at the start of the hand-off phase of epoch j. Taking inspiration from [1], we assume that the environment $\mathcal{E}$ informs $\mathcal{C}^j$ about the committee $\mathcal{C}^{j+1}$ at the start of the hand-off phase of epoch j. We denote the number of parties in committee $\mathcal{C}^j$ as n_j. Finally, we denote the number of committees or epochs as D.

A.2 Security Model

Semi-honest or Malicious. Adversary $\mathcal{A}$ receives the corrupted parties' local state determined by environment $\mathcal{E}$. As in the traditional MPC setting, for a semi-honest adversary, $\mathcal{A}$ only knows the local state, while a malicious adversary $\mathcal{A}$ can take full control of the corrupted parties.

Corruption Threshold. Different from previous works [1, 7, 24], we consider a more refined corruption threshold. We denote t_j as the number of parties in committee $\mathcal{C}^j$ that can be adversary corrupt. We mainly consider the *retroactive effect* adversary, which means if a party is corrupted, then it will be counted to meet every corruption threshold of the committee that includes this party. To simplify, we can assume that every committee has different parties. Specifically, without loss of generality, we assume that for every committee $\mathcal{C}^j$, $t_j = \Theta(n_j)$ or, in other words, $n_j > t_j \geq \sigma n_j$, where σ is a constant positive number according to the security parameter.

Point of Corruption. As demonstrated before, the committee $\mathcal{C}^j$ is determined at the start of the hand-off phase of epoch $j-1$. Similar to the definition in [7], in a *static* manner, an adversary can only corrupt the parties when $\mathcal{C}^j$ is determined. An *adaptive* adversary, however, can corrupt parties at any time when $\mathcal{C}^j$ is alive.

Effect on Prior Epochs. Similar to [7], we consider a *retroactive effect* adversary. Consider that a party may be alive in multiple committees, *retroactive*

effect means when a party in $\mathcal{C}^j$ is corrupted, then committees before $\mathcal{C}^j$ who include this party have to count it to meet the corruption threshold. For simplicity, we can consider a party that only lives in one committee.

Dynamic Arithmetic Black Box. While we discuss the limitation of communication cost in the weakest setting, even with a *semi-honest R-static adversary*, the protocol we proposed can defend against a *malicious R-adaptive adversary* with abort. Security with abort is general for designing MPC protocols, ensuring that honest clients can all receive the output or all of the parties abort. That is the same as in previous works [1, 7, 24]. Moreover, we adapt the dynamic arithmetic black box (DABB) ideal functionality $\mathcal{F}_{\mathsf{DABB}}$ shown in [1, 24], which is based on the finite field $\mathbb{F}_p$ and includes 'Parameters, Init, Input, Next-Committee, Linear Combination, Multiply and Output'. Furthermore, to formally define communication and committee management, we use the functionality $\mathcal{F}_{\mathsf{commit}}$, which is also used in [1]. The formal definitions of $\mathcal{F}_{\mathsf{DABB}}$ and $\mathcal{F}_{\mathsf{commit}}$ are shown in Appendix A.3.

Weak Privacy. In our work, we adopt the idea of *Weak Privacy* and *Malicious Security Compiler* proposed in [7]. In brief, *Weak privacy* describes a Fluid MPC scheme that can protect privacy in a *malicious R-adaptive adversary* environment without guaranteeing the correctness of the output. The *Malicious Security Compiler*, on the other hand, provides a method to transform a linear-based Fluid MPC scheme with Weak privacy into one with security with abort, where a linear protocol means every operation of the protocol is linear.

A.3 Dynamic Arithmetic Black Box (DABB)

We adapt the dynamic arithmetic black box (DABB) ideal functionality $\mathcal{F}_{\mathsf{DABB}}$ shown in [1,24], which is based on finite field $\mathbb{F}_p$. To further enhance the generality of $\mathcal{F}_{\mathsf{DABB}}$, we modify the $\mathcal{F}_{\mathsf{DABB}}$ definition from the works of [1,24] by revising 'Add' to 'Linear Combination'. This addresses the issue of vague definitions for scalar multiplication and scalar addition in previous works. It is worth noting that the 'relay gate', which transmits computed values to a later layer, can be implemented using a 'Linear combination gate' by simply setting $\boldsymbol{a} = (1, \cdots, 1)$. The $\mathcal{F}_{\mathsf{DABB}}$ we present includes 'Parameters, Init, Input, Next-Committee, Linear Combination, Multiply and Output', which is suitable for more common situations than the Fluid MPC model proposed in [7]. Similar to the traditional MPC, the designing of realizing Multiply will be the key part of the protocol. The formal definition of $\mathcal{F}_{\mathsf{DABB}}$ is shown following.

Furthermore, we use the functionality $\mathcal{F}_{\mathsf{commit}}$ to determine the communication between parties, which is also used and shown in [1].

B Supplementary Constructions and Proofs Omitted from Main Text

B.1 Three Possible Situations for PPT Adversary When $t < d$

This subsection is the supplementary discussion of section 4.3 for PPT adversary.

Functionality 5: $\mathcal{F}_{\mathsf{DABB}}$

Parameters: Finite field $\mathbb{F}_p$, universe $\mathcal{U}$ of parties, and set of clients $\mathcal{C}_{\text{clnt}} \subseteq \mathcal{U}$. The functionality assumes that all parties have agreed upon public identifiers id_x, for each variable x used in the computation.

Init: On input $(\mathsf{Init}, \mathcal{C})$ from every party $P_i \in \mathcal{C}^{\text{clnt}}$, where each P_i sends the same set $\mathcal{C} \subseteq \mathcal{U}$, initialize $j = 1, \mathcal{C}^1 = \mathcal{C}$ as the first active committee.

Input: On input $(\mathsf{Input}, \text{id}_x, x)$ from some $P_i \in \mathcal{T}^{\text{clnt}}$, and $(\mathsf{Input}, \text{id}_x)$ from all other parties in $\mathcal{C}orr^{\text{clnt}}$, store the pair (id_x, x).

Next-Committee: On input $(\mathsf{Next-Committee}, \mathcal{C})$ from every party $P_i \in \mathcal{C}^j$, where each P_i sends the same set $\mathcal{C} \subseteq \mathcal{U}$, update $j = j + 1, \mathcal{C}^j = \mathcal{C}$.

Linear Combination: On input $(\mathsf{LC}, \text{id}_z, \mathbf{id}_x, \boldsymbol{a})$ from every party $P_i \in \mathcal{C}^j$, where $\mathbf{id}_x$ is a vector of $(\text{id}_{x_1}, \cdots \text{id}_{x_\ell})$ and $\boldsymbol{a}$ is a ℓ-dimensional coefficient vector, compute $z = (x_1, \cdots, x_\ell) \cdot \boldsymbol{a}^T$ and store (id_z, z).

Multiply: On input $(\mathsf{Mult}, \text{id}_z, \text{id}_x, \text{id}_y)$ from every party $P_i \in \mathcal{C}^j$, compute $z = x \cdot y$ and store (id_z, z).

Output: On input $(\mathsf{Output}, \{\text{id}_{z_m}\})$ from every party $P_i \in \mathcal{C}^{\text{clnt}} \cup \mathcal{C}^j$, where a value z_m for each id_{z_m} has been stored previously, retrieve $\{(\text{id}_{z_m}, z_m)\}$ and them to adversary. Wait for input from the adversary, if it is Deliver, send the output to every $P_i \in \mathcal{C}^{\text{clnt}}$. Otherwise, abort.

Functionality 6: $\mathcal{F}_{\mathsf{commit}}$

The functionality runs between a set of parties $\mathcal{P}$ and an adversary $\mathcal{A}$.

Commit: On input $(\mathsf{commit}, P_i, x, \tau_x)$ from P_i, where τ_x is a previously unused identifier, store (P_i, x, τ_x) and sent (P_i, τ_x) to all parties.
Open: On input $(\mathsf{open}, P_i, x, \tau_x)$ from P_i, retrieve x and send (x, i, τ_x) to all parties.

No information. First of all, we consider the setting that the adversary does *not know any information* about the transport ways from $\mathcal{C}^1$ to $\mathcal{C}^2$. In this setting, we can demonstrate the probability that the *adaptive* adversary executes completely random corruption and discovers in the hand-off phase that the corrupted result $\mathcal{C}orr^1 \subset \mathcal{C}^1$ contains at least $\lfloor \frac{t}{\lambda} \rfloor$ of the $\mathcal{S}_i^1 \in \mathcal{S}^1$ by

$$\epsilon_{\text{ad}} = \Pr\left[\exists \mathcal{S}_{\text{ch}}^1 \subset \mathcal{S}^1, \left|\mathcal{S}_{\text{ch}}^1\right| \geq \lfloor \tfrac{t}{\lambda} \rfloor, \bigcup \mathcal{S}_{\text{ch}}^1 \subset \mathcal{C}orr^1\right],$$

where $\mathcal{S}_{\text{ch}}^1$ denotes the possible choice of including at least $\lfloor \frac{t}{\lambda} \rfloor$ parties' inputs. It should be noted that different selections of $\mathcal{S}_{\text{ch}}^1$ are not mutually independent, and different specific distribution strategies for $\mathcal{S}^1$ yield varying probability results. Therefore, calculating an accurate value for ϵ_{ad} is challenging. However, we can provide a very loose lower bound.

Considering that $\hat{\mathcal{S}}^1_{\text{ch}}$ contains first $\lfloor \frac{t}{\lambda} \rfloor$ elements of $\mathcal{S}^1$, i.e. $\hat{\mathcal{S}}^1_{\text{ch}} = \{\mathcal{S}^1_1, \mathcal{S}^1_2, \cdots, \mathcal{S}^1_{\lfloor \frac{t}{\lambda} \rfloor}\}$. We denote that $p_{\lfloor \frac{t}{\lambda} \rfloor} = \Pr\big[\bigcup \hat{\mathcal{S}}^1_{\text{ch}} \subset \mathcal{C}orr^1\big]$. Where $\bigcup \hat{\mathcal{S}}^1_{\text{ch}}$ has at most t elements. Thus $p_{\lfloor \frac{t}{\lambda} \rfloor}$ is at least $1/\binom{n}{t}$, which means randomly choose $\mathcal{C}orr^1$ and equal to $\bigcup \hat{\mathcal{S}}^1_{\text{ch}}$ just right. Therefore, we can give the lower bound with $\epsilon_{\text{ad}} \geq p_{\lfloor \frac{t}{\lambda} \rfloor} \geq 1/\binom{n}{t}$. The conditions for equality in this inequality are extremely stringent. It requires $\lfloor \frac{t}{\lambda} \rfloor = n - \mu$ and $\mathcal{S}^1_{\text{ch}} = \mathcal{S}^1$ that $\hat{\mathcal{S}}^1_{\text{ch}}$ becomes the only choice (first equality) while simultaneously demanding that $\bigcup \hat{\mathcal{S}}^1_{\text{ch}}$ contains exactly t parties (second equality). This implies that elements $\mathcal{S}^1_i$ in $\mathcal{S}^1$ are mutually disjoint and each contains λ parties, while also indicating that t/λ is an integer [8].

As previously shown, the adversary must corrupt an additional t honest parties in $\mathcal{C}^2$ to acquire $d+1 = d-t+1+t$ shares. Considering a *static* adversary, it can only randomly and continually select $\mathcal{C}orr^2$. And the probability of $\mathcal{C}orr^2$ meeting the exact condition is difficult to compute due to the varying distribution of $\mathcal{S}^1$.

If we continually consider $\hat{\mathcal{S}}^1_{\text{ch}}$ situation as demonstrated for *adaptive* adversary, then corresponding $\lfloor \frac{t}{\lambda} \rfloor$ parties in $\mathcal{C}^2$ will be deterministic. Therefore, the probability of the adversary corrupting other t honest parties can be computed in a similar way with the proof of Lemma 2: $p_{\text{st}} = \binom{n-\lfloor \frac{t}{\lambda} \rfloor}{t} / \binom{n}{t} \geq \binom{n-\lfloor \frac{t}{\lambda} \rfloor}{d-\lfloor \frac{t}{\lambda} \rfloor+1} / \binom{n}{d-\lfloor \frac{t}{\lambda} \rfloor+1}$. Where the '$\geq$' is due to $\lfloor \frac{t}{\lambda} \rfloor \geq d-t+1$, which means $t \geq d - \lfloor \frac{t}{\lambda} \rfloor + 1$. Based on these, we can give a lower bound of the successful probability of *static* adversary, and the conditions for equality in this inequality are similar to the inequality of ϵ_{ad}.

$$\epsilon_{\text{st}} \geq p_{\lfloor \frac{t}{\lambda} \rfloor} \cdot p_{\text{st}} \geq \frac{\binom{n-\lfloor \frac{t}{\lambda} \rfloor}{d-\lfloor \frac{t}{\lambda} \rfloor+1}}{\binom{n}{t}\binom{n}{d-\lfloor \frac{t}{\lambda} \rfloor+1}} \tag{2}$$

Fully deterministic strategy. This strategy indicates that at the beginning of the hand-off phase, each party should know exactly which parties in the next committee they should send messages to. Moreover, the specific choices for message transmission are known to all parties, meaning that $\mathcal{S}^1$ is known to all parties $P^1_i \in \mathcal{C}^1$. Therefore, the adversary will also know the exact ways of message transportation, even for the honest parties.

This strategy appears to be highly advantageous for the adversary. However, we point out that this is actually the message transmission strategy currently used by all existing Fluid MPC schemes [1, 7, 24], making it the most widely representative. The works of [7,24] need every party to send messages to all parties of the next committee, which are unmistakably deterministic. The strategy of [1] needs every party in $\mathcal{C}^1$ to send messages to $\left\lfloor \frac{n_2}{n_1} \right\rfloor$ or $\left\lfloor \frac{n_2}{n_1} \right\rfloor + 1$ parties in $\mathcal{C}^2$. Considering that every party in $\mathcal{C}^2$ will receive only one message, the selection of every party's transmission targets cannot be randomly chosen by itself. Which will be deterministic when transmission begins.

[8] These conditions actually constraint $\lambda = 1$

Under this strategy, *adaptive* adversary can definitely control at least $\lfloor \frac{t}{\lambda} \rfloor$ $\mathcal{S}_i^1 \in \mathcal{S}^1$. Furthermore, considering p_{st} is a polynomial function of n, thus the *static* adversary can still successfully execute the attack with a non-negligible probability.

Completely random strategy. Under the completely randomized strategy, all $\mathcal{S}_i^1$ in $\mathcal{S}^1$ will be randomly selected from $\mathcal{C}^1$. We can calculate the accurate probability that the adversary executes the attack. Let event A_i represent that all inputs of P_i^2 are controlled, i.e., $\mathcal{S}_i^1 \subset \mathcal{C}orr^1$. Then, the probability of A_i occurring is:

$$p = \Pr[A_i] = \frac{\binom{n-\lambda}{t-\lambda}}{\binom{n}{t}}$$

Since each $\mathcal{S}_i^1$ is selected completely randomly, $\forall i \neq i',\ A_i$ and $A_{i'}$ are mutually independent. Let X be the number of occurrences of event $A_i \in \{A_1, \cdots, A_{n-\mu}\}$. Then X follows a binomial distribution, and we have:

$$\Pr[X = x] = \tbinom{n-\mu}{x} p^x (1-p)^{n-\mu-x}$$

Furthermore, the *adaptive* adversary's successful probability of attack can be computed as:

$$\epsilon_{\text{ad}}^{random} = \Pr[X \geq d - t + 1] = 1 - \sum_{x=0}^{d-t} \tbinom{n-\mu}{x} p^x (1-p)^{n-\mu-x}.$$

As for *static* adversary, we need to consider $X = x$ as well as let the adversary additionally corrupt $d+1-x$ parties in $\mathcal{C}^2$. Then the probability can be computed as:

$$\begin{aligned}
&p_{ran_sta}(x) = \Pr[X = x \text{ and the adversary additionally corrupt at least } d+1-x \text{ parties in } \mathcal{C}^2] \\
&= \begin{cases} 0 & 0 \leq x < d-t+1 \\ \Pr(X=x) \cdot \left(1 - \sum_{k=0}^{d+1-x} \frac{\binom{n-x}{k}}{\binom{n}{t}}\right) & d-t+1 \leq x \leq d+1 \\ \Pr[X=x] & d+1 < x \leq n-\mu \end{cases} \quad \text{If } d+1 \leq n-\mu \\
&= \begin{cases} 0 & 0 \leq x < d-t+1 \\ \Pr[X=x] \cdot \left(1 - \sum_{k=0}^{d+1-x} \frac{\binom{n-x}{k}}{\binom{n}{t}}\right) & d-t+1 \leq x \leq n-\mu \end{cases} \quad \text{If } d+1 > n-\mu
\end{aligned}$$

Therefore, the *static* adversary's successful probability of attack can be computed as:

$$\epsilon_{\text{st}}^{random} = \sum_{x=0}^{n-\mu} p_{ran_sta}(x)$$

Note that although we have not been able to provide a lower bound for $\epsilon_{\text{st}}^{random}$ due to its computational complexity, we point out that since $\Pr[X = x]$ follows a binomial distribution, it can be approximated by a normal distribution when n is large. Specifically, if d is half of n, then calculating

$$\epsilon_{\text{st}}^{random} = \sum_{x=0}^{n-\mu} p_{ran_sta}(x) > \sum_{x=d+1}^{n-\mu} p_{ran_sta}(x)$$

is equivalent to integrating nearly half of the normal distribution, which will always yield a non-negligible result, even for PPT adversaries.

B.2 Proof of Theorem 1

Proof. Firstly, we prove the situation when the settings of each committee are pairwise equal. The situation for $t \geq d$ is proved by Lemma 2 and the communication costs are proved in Corollary 3. When we consider $d > t \geq (1 - \frac{1}{\lambda+1})(d+2)$ and $\lfloor \frac{t}{\lambda} \rfloor \leq n - \mu$[9]. There exists a feasible corruption method for the adversary that allows it to compromise the privacy protection security of π_{reshare} with a non-zero probability. As described in section 4.3, for adversaries with different capabilities, there exists a non-zero lower bound on the success rate of this attack.

For an adversary with information-theoretic capabilities, as shown in equation 1, the adversary can always find a corruption method. For an adversary with PPT capabilities. If the adversary has *adaptive* capabilities, then according to section 4.3, a lower bound for its attack success probability can be given as $\epsilon_{\text{ad}} \geq 1/\binom{n}{t}$. If the adversary has *static* capabilities, then according to equation 2, a lower bound for its attack success probability can be given as $\epsilon_{\text{st}} \geq \frac{\binom{n-\lfloor \frac{t}{\lambda} \rfloor}{d-\lfloor \frac{t}{\lambda} \rfloor+1}}{\binom{n}{t}\binom{n}{d-\lfloor \frac{t}{\lambda} \rfloor+1}}$. For the exact choice of n, t, λ, the values of ϵ_{ad} and ϵ_{st} are both nonzero constants.

However, considering that ϵ_{ad} and ϵ_{st} are both exponentially small in terms of n. As n increases, these two probabilities will rapidly decrease and can break through the adversary's attack success probability of $1/p$ commonly seen in existing MPC schemes [21, 23]. Therefore, it is necessary for us to take the size of n into consideration. As shown in Table 1, there does not exist a procedure π_{reshare} that is secure for any number of parties n against a PPT adversary.

Secondly, for situation that committees have different setting. As mentioned in the information-theoretic adversary situation of Section 4.3, the adversary can corrupt $t_{j'-1}$ parties that denoted as $\mathcal{C}orr^{j'-1} \subset \mathcal{C}^{j'-1}$, which have none zero opportunity to encompass at least $\left\lfloor \frac{t_{j'-1}}{\lambda} \right\rfloor$ $\mathcal{S}_i^{j'-1} \in \mathcal{S}^{j'-1}$. Therefore, the

[9] The first condition can be satisfied because $2\lambda < d$. For the second condition, the opposite case $\lfloor \frac{t}{\lambda} \rfloor > n - \mu$ is demonstrated in section 4.3. For the sake of brevity, it will not be elaborated upon in this proof.

Procedure 5: $\pi_{\mathsf{double-rand}}$

Usage: $\mathcal{C}^j$ get Shamir random double sharings $[r]_d^j, [r]_{2d}^j$.

1. $\forall P_i^{j-1} \in \mathcal{C}^{j-1}$, each party locally generates random $s_i \in \mathbb{F}_p$, runs $\Pi_{\mathrm{LSS}}.\mathrm{Share}(s_i)$ to get degree-d and degree-$2d$ sharings $[s_i]_d, [s_i]_{2d}$, and distributes them to $\mathcal{C}^j$.
2. The parties of $\mathcal{C}^j$ receive $\{[s_i]_d, [s_i]_{2d}\}$ for $i \in [n]$, locally compute

$$\begin{pmatrix} [r_1]_d & [r_1]_{2d} \\ \vdots & \vdots \\ [r_{n-t}]_d & [r_{n-t}]_{2d} \end{pmatrix} = M \begin{pmatrix} [s_1]_d & [s_1]_{2d} \\ \vdots & \vdots \\ [s_n]_d & [s_n]_{2d} \end{pmatrix},$$

and get $n-t$ pairs of Shamir random double sharing $[r]_d^j, [r]_{2d}^j$.

Procedure 6: π_{zero}

Usage: $\mathcal{C}^j$ get a degree-d secret sharing $[o]_d^j$, where $o = 0$.

1. $\forall P_i^{j-1} \in \mathcal{C}^{j-1}$, each party runs $\Pi_{\mathrm{LSS}}.\mathrm{Share}(0)$ locally to get random degree-d sharing $[o']_d$, and sends them to $\mathcal{C}^j$.
2. The parties of $\mathcal{C}^j$ receive $\{[o'_i]_d\}$ for $i \in [n]$, locally compute

$$\begin{pmatrix} [o_1]_d \\ \vdots \\ [o_{n-t}]_d \end{pmatrix} = M \begin{pmatrix} [o'_1]_d \\ \vdots \\ [o'_n]_d \end{pmatrix},$$

and get $n-t$ pairs of Shamir random double sharings $[o]_d^j$.

adversary can gain $\left\lfloor \frac{t_{j'-1}}{\lambda} \right\rfloor$ parties' shares of secret v in committee $\mathcal{C}^{j'}$ without corruption. And corrupting $d_{j'} - \left\lfloor \frac{t_{j'-1}}{\lambda} \right\rfloor + 1$ more parties in $\mathcal{C}^{j'}$ will be enough for the adversary to obtain the secret v. To ensure this attack, $t_{j'-1}, t_{j'}$ and $d_{j'}$ have to meet the condition $t_{j'} \geq d_{j'} - \left\lfloor \frac{t_{j'-1}}{\lambda} \right\rfloor + 1$. Considering that $\left\lfloor \frac{t_{j'-1}}{\lambda} \right\rfloor > \frac{t_{j'-1}}{\lambda} - 1$, $t_{j'} + \frac{t_{j'-1}}{\lambda} \geq d_{j'} + 2$ is sufficient.

B.3 Procedures for Random Sharings Generation

The procedure for generating $[r]_d, [r]_{2d}$ with $O(n)$ communication complexity is proposed in Procedure $\pi_{\mathsf{double-rand}}$. This procedure is mainly developed from DN07 [8] and is suitable for the Fluid MPC. Here, M denotes a $(n-t) \times n$ hyper-invertible matrix known by every party in $\mathcal{U}$. The main property of M is that any $(n-t) \times (n-t)$ sub-matrix of M is invertible.

Table 3. Proof strategy for Lemma 2

Steps	Situation	Adversary ability	Proof strategy
Part 1	$t > d$	PPT	Adversary can easily compute the secret with $t > d$ shares
Part 2	$t = d$	Information-theoretic	The state of a special party $P^2_{\text{ch}} \in \mathcal{C}^2$ can be controlled by the adversary without corruption.
Part 3	$t = d$	PPT/Statistical	Introducing three parameters μ, λ, γ to find more parties in $\mathcal{C}^2$ whose state can be controlled by the adversary without corruption.

By simply modifying $\pi_{\mathsf{double-rand}}$, a pair of generated double sharings can be transformed into a single sharing, making it easy to obtain a procedure for generating a single random share. Here, we provide the definition of π_{zero}:

Lemma 4 *Procedures $\pi_{\mathsf{double-rand}}$ and π_{zero} can securely compute functionalities $\mathcal{F}_{\mathsf{double-rand}}$ and $\mathcal{F}_{\mathsf{zero}}$ separately in the presence of a malicious R-adaptive adversary controlling at most t corrupted parties in each committee.*

Due to space limitations, the detailed proof of this lemma is not provided here. Both $\pi_{\mathsf{double-rand}}$ and π_{zero} are simulatable. This is because, considering that honest parties generate $n - t$ pairs (or individual) of completely random shares, the $n-t$ pairs (or individual) shares produced through a linear combination of these shares will also be completely random. The guarantee of complete randomness is similar to that in schemes such as DN07 [8].

B.4 Proof of Lemma 2

Proof. Assuming that there exists a π_{reshare} that can securely compute $\mathcal{F}_{\mathsf{reshare}}$ with at most $O(n)$ field elements communication cost, as shown in Table 3, we separate the proof into three parts.

Part 1. When $t > d$, the adversary can easily reconstruct the secret v based on shares held by corrupted parties, which means no π_{reshare} can securely compute $\mathcal{F}_{\mathsf{reshare}}$.

Part 2. As for $t = d$, we begin with an *information-theoretic adversary*, then we have the following claim:

Claim 1 *There must exist a party $P^2_{\text{ch}} \in \mathcal{C}^2$, which can receive messages from at most t parties in committee $\mathcal{C}^1$. Otherwise, π_{reshare} will have at least $\Omega(n{\cdot}(t{+}1)) = \Omega(n^2)$ communication cost.*

Adaptive adversary. For the situation of the *adaptive* adversary, denote the set of parties in $\mathcal{C}^1$ who send messages to P^2_{ch} as $\mathcal{S}^1_{\text{ch}}, |\mathcal{S}^1_{\text{ch}}| = t$ (If $|\mathcal{S}^1_{\text{ch}}| < t$, add a sufficient amount of random parties from $\mathcal{S}^1_{\text{ch}}$ to make up for it). Then, the

adversary will have probability of ϵ_{base} to corrupt all parties in $\mathcal{S}^1_{\text{ch}}$ exactly right, which means $\mathcal{C}orr^1 = \mathcal{S}^1_{\text{ch}}$. The probability is at least:

$$\epsilon_{\text{base}} = \Pr[\mathcal{C}orr^1 = \mathcal{S}^1_{\text{ch}}] = 1/\binom{n}{t} > 0.$$

Here $1/\binom{n}{t}$ is computed in a completely random situation in which the adversary has no information about the transmission strategy. [10]

When the adversary corrupts all parties in $\mathcal{S}^1_{\text{ch}}$ with non-zero probability ϵ_{base}, denote the messages received by P^2_{ch} from $P^1_i \in \mathcal{S}^1_{\text{ch}}$ as $\boldsymbol{m}_i = \{m_i^{(1)}, m_i^{(2)}, \cdots\}$. According to the Fluid model, the share v^2_{ch} held by P^2_{ch} will be determined by the messages received from parties in $\mathcal{S}^1_{\text{ch}}$ and the random vector $\boldsymbol{z}$ generated by itself. That is, there exists a function f such that $v^2_{\text{ch}} = f(\bigcup_{P^1_i \in \mathcal{S}^1_{\text{ch}}} \boldsymbol{m}_i, \boldsymbol{z})$. Moreover, we point out that $v^2_{\text{ch}} = f(\bigcup_{P^1_i \in \mathcal{S}^1_{\text{ch}}} \boldsymbol{m}_i, \boldsymbol{z}) = f(\bigcup_{P^1_i \in \mathcal{S}^1_{\text{ch}}} \boldsymbol{m}_i, \boldsymbol{0})$.

Considering that shares $\{v^2_i\}$ held by $P^2_i \notin \mathcal{C}^2 \backslash P^2_{\text{ch}}$ is uniquely determined. Therefore, regardless of the value of the random vector $\boldsymbol{z}$, the calculated v^2_{ch} will be the same. As a result, the adversary can acquire v^2_{ch} by calculating $f(\bigcup_{P^1_i \in \mathcal{S}^1_{\text{ch}}} \boldsymbol{m}_i, \boldsymbol{0})$. It is worth noting that the *adaptive* adversary can do the corruption at any time during the committee's lifetime. Thus, it can corrupt other $t = d$ parties in $\mathcal{C}^2$, know $t + 1 = d + 1$ shares $\{v^2_{\text{ch}}\} \cup [v]_{\mathcal{C}orr^2}$ of v, and break the privacy of the protocol.

Static adversary. As discussed before, adversary has to corrupt other t parties in $\mathcal{C}^2$ except P^2_{ch}. However, the *static* adversary can only decide the corruption at the beginning of the hand-off phase, when $\mathcal{C}^2$ is determined by the environment $\mathcal{E}$. At this point, the adversary may have no idea of the specific transfer methods that will be used during the hand-off phase. Thus, the *static* adversary needs to design the corruption strategy according to the transfer strategy of π_{reshare}[11]. Note that no matter how π_{reshare} is designed, the adversary can choose corrupted set $\mathcal{C}orr^2$ randomly, which has at least a p_{st} probability to meet $\mathcal{C}orr^2 \subset \mathcal{C}^2 \backslash P^2_{\text{ch}}$. In which p_{st} is:

$$p_{\text{st}} = \Pr\big[\mathcal{C}orr^2 \subset \mathcal{C}^2 \backslash P^2_{\text{ch}}\big] = \frac{\binom{n-1}{t}}{\binom{n}{t}} = 1 - \frac{t}{n} \geq 1 - \sigma,$$

which is greater than a non-zero constant. Therefore, the *static* adversary has at least

$$\epsilon_{\text{st_loose}} \geq \epsilon_{\text{base}} \cdot p_{\text{st}} = \frac{\binom{n-1}{t}}{\binom{n}{t}^2} > 0$$

[10] This situation is suitable with the *adaptive* adversary when the strategy is like that every $P^1_i \in \mathcal{C}^1$ randomly sends a message to a party in $\mathcal{C}^2$. In this strategy, before sending the message, there is no way to acquire the exact transmission choice. After sending the message, the parties in $\mathcal{C}^1$ will be permanently inactive.

[11] This is meaning that every strategy of π_{reshare} will have a 'Best Strategy' for adversary, which is not needed in the proof of lemma 2. However, this will be discussed in a later section, especially for the situation of the PPT adversary and $t < d$.

probability to successfully compromise the privacy security of π_{reshare}, which means informational security is impossible.

Part 3. As for a PPT or statistical adversary (For the sake of simplicity, we will only mention PPT adversary later), if we consider that the parameter n is a constant number, then $\epsilon_{\text{st_loose}}$ will be a non-zero constant probability, and the proof before is *enough*.

Nevertheless, in the Fluid MPC setting, n is a variable number. This means n is included in the security parameter σ and contributes to the success rate of the adversary. In this situation, the success rate $\epsilon_{\text{base}}, \epsilon_{\text{st_loose}}$ of the adversary is inversely proportional to the n-th power, which is secure with the PPT adversary. As far as we know, this is the *first* work to connect the number of parties with the security probability in the MPC with dynamic parties.

Considering that p_{st} is greater than a non-zero constant, we have to enhance the ϵ_{base} to be a polynomial function of n. Recall the claim in Part 2, the t restriction is much more than needed. We have a tighter claim. Assuming that there exists a π_{reshare} that can securely compute $\mathcal{F}_{\mathsf{reshare}}$ with at most $O(n)$ field elements communication cost, we have:

Claim 2 *Committee $\mathcal{C}^2$ can be divided into two sets $\mathcal{H}$ and $\mathcal{L}$ according to the received data volume measured in field elements. In which $|\mathcal{H}| = \mu = o(n)$ is asymptotically negligible to n and $|\mathcal{L}| = |\mathcal{C}^2| - \mu = n - \mu$. And, $\forall P_i^2 \in \mathcal{L}$, P_i^2 can receive single message from no more than λ parties separately in the committee $\mathcal{C}^1$ and every message has at most γ field elements, where $\lambda > 0, \gamma > 0$ are constant integers.*

Here, we firstly show why this separation exists. As shown in Fig. 2, we let any party that receives $\omega(1)$ field elements belong to 'heavy' set $\mathcal{H}$. Then $|\mathcal{H}| = \mu$ must be asymptotically negligible to n (i.e., $o(n)$), otherwise communication costs of $\mathcal{H}$ will be $\omega(1) \cdot \Omega(n) = \omega(n)$. We use $T(\mu, n) = O(n)$ to describe the total communication costs of $\mathcal{H}$, notes that $T(0, n) = 0$. The last $n - \mu$ parties are denoted as 'light' set $\mathcal{L}$. Considering that $\mu = o(n)$, thus $|\mathcal{L}| \sim n$. Every party in $\mathcal{L}$ must be restricted to constant communication costs (i.e., $\lambda \cdot \gamma$), otherwise communication costs of $\mathcal{L}$ will be $\omega(1) \cdot \Omega(n) = \omega(n)$.

Furthermore, if $\lambda = 0$ or $\gamma = 0$, all $n - \mu$ parties in $\mathcal{L}$ would not receive any input. Consequently, the shares held by these parties could only be random values or 0 without loss of generality. Which is unacceptable to recover the secret v. To sum up, we have $\mu = o(n) \geq 0, \gamma \geq 1$ and $\lambda \geq 1$.

Denote the set of parties in $\mathcal{C}^1$ who send messages to $P_i^2 \in \mathcal{L}$ as $\mathcal{S}_i^1$(without loss of generality, we assume $\mathcal{L} = \{P_1^2, \cdots, P_{n-\mu}^2\}$). We set $|\mathcal{S}_i^1| = \lambda$, which will be filled up with other parties when $\mathcal{S}_i^1$ does not have enough parties. Furthermore, we denote that $\mathcal{S}^1 = \{\mathcal{S}_i^1 | P_i^2 \in \mathcal{L}\}$. The adversary-controlled set of parties $\mathcal{C}orr^1$ only needs to contain any $\mathcal{S}_i^1 \in \mathcal{S}^1$ to gain all input messages for party $P_i^2 \in \mathcal{C}^2$. This enables the computation of the share held by P_i^2. Subsequently, by selecting $\mathcal{C}orr^2 \subset \mathcal{C}^2 \backslash P_i^2$, the adversary can obtain $t + 1 = d + 1$ shares, ultimately compromising the privacy of the secret.

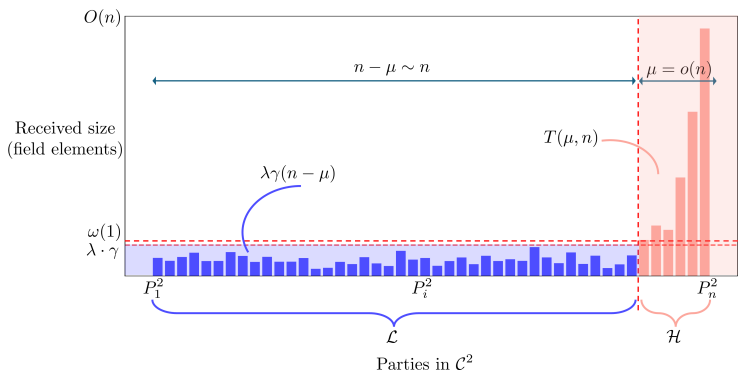

Fig. 2. Data received by committee $\mathcal{C}^2$ under an $O(n)$ communication bound. If a protocol implements the functionality $\mathcal{F}_{\mathsf{reshare}}$ to transmit a share to $\mathcal{C}^2$ with total communication cost $O(n)$, then the final-round communication pattern must follow the structure illustrated in the figure. Without loss of generality, let the last $\mu = o(n)$ *heavy* parties (denoted by $\mathcal{H}$) each receive up to $O(n)$ field elements, yielding a total communication cost $T(\mu, n) = O(n)$ (with $T(0, n) = 0$). The remaining $n - \mu \approx n$ *light* parties, denoted by $\mathcal{L}$, each receive at most $\lambda \cdot \gamma$ field elements, where $\lambda \geq 0$ and $\gamma \geq 0$ are fixed constants. Hence, the total data received by all parties in $\mathcal{L}$ is $\lambda\gamma(n - \mu)$.

Adaptive adversary. Thus, we consider the probability of the adversary having $\mathcal{C}orr^1$ contain at least one $\mathcal{S}_i^1 \in \mathcal{S}^1$, which is

$$\epsilon_{\text{ad}} = \Pr\left[\exists \mathcal{S}_i^1 \in \mathcal{S}^1, \mathcal{S}_i^1 \subset \mathcal{C}orr^1\right]$$

However, ϵ_{ad} is a variable according to the distribution of $\mathcal{S}^1$. Thus, it is hard to compute an exact value of ϵ_{ad}. Fortunately, we can give a sufficiently large lower bound of ϵ_{ad}. The worst strategy for adversary is choosing the corruption $\mathcal{C}orr^1$ randomly. Then, let event A_i denote $\mathcal{S}_i^1 \subset \mathcal{C}orr^1$, whose probability can be computed as: $p_i = \Pr[A_i] = \binom{n-\lambda}{t-\lambda} / \binom{n}{t}$. It is worth noting that p_i is a polynomial function of n. At the same time,

$$\epsilon_{\text{ad}} = \Pr\left[\bigcup_{i=1}^{n-\mu} A_i\right] \geq p_i.$$

Therefore, ϵ_{ad} is none-negligible for a PPT adversary. As mentioned before, ϵ_{ad} is the success probability of the *adaptive* adversary executing the attack.

Static adversary. The *static* adversary additionally needs to meet the condition $\mathcal{C}orr^2 \subset \mathcal{C}^2 \backslash P_i^2$ to execute the attack while having $\exists\ \mathcal{S}_i^1 \in \mathcal{S}^1,\ \ \mathcal{S}_i^1 \subset \mathcal{C}orr^1$. Therefore, considering that the choices of $\mathcal{C}orr^1$ and $\mathcal{C}orr^2$ are uncorrelated when corruption is random, we can also give a lower bound for the success probability of *static* adversary to execute the attack as follows

$$\epsilon_{\text{st}} \geq p_i \cdot p_{\text{st}} = \frac{\binom{n-\lambda}{t-\lambda}\binom{n-1}{t}}{\binom{n}{t}^2}.$$

This is computed by only considering the inputs of one party $P_i^{j'}$ controlled by the adversary. Noting that $\binom{n-\lambda}{t-\lambda}\binom{n-1}{t}/\binom{n}{t}^2$ is a polynomial function of n, the attack strategy is feasible for PPT adversaries. ■

B.5 Proof of Lemma 3

Proof. The proof will be separated into two steps. First, we will prove that $\pi_{\mathsf{rs-two}}$ can securely compute the functionality $\mathcal{F}_{\mathsf{rs-two}}$. Then, we will continue to the π_{reshare} and $\mathcal{F}_{\mathsf{reshare}}$.

Simulation of $\mathcal{F}_{\mathsf{rs-two}}$: For the $\pi_{\mathsf{rs-two}}$, a simulator $\mathcal{S}_1$ will be constructed to simulate the behaviors of honest parties.

Step 1: $\mathcal{S}_1$ emulates $\mathcal{F}_{\mathsf{zero}}$.
Step 2: $\forall i \in [n]$:
- If P_i^j is honest and P_i^{j+1} is corrupted, $\mathcal{S}_1$ randomly generates ${v'}_i^{j+1}$ and sends it as the share of $[v']_d^{j+1}$ held by P_i^{j+1}.
- If P_i^j and P_i^{j+1} are both corrupted, notice that we do not care about the currect of the output, $\mathcal{S}_1$ randomly generates ${v'}_i^{j+1}$ and considers it as the share hold by P_i^{j+1}.
- Otherwise, $\mathcal{S}_1$ does nothing.

Then, $\mathcal{S}_1$ knows all the shares of $[v']_d^{j+1}$ held by corrupted parties in $\mathcal{C}^{j+1}$ and sends them to $\mathcal{F}_{\mathsf{rs-two}}$.

Hybrid Arguments for $\mathcal{S}_1$. Now, we show that $\mathcal{S}_1$ can perfectly simulates the behaviors of honest parties. Consider the following hybrids.

Hybrid$_0$: The execution in the real world.

Hybrid$_1$: In this hybrid, $\mathcal{S}_1$ computes the shares of corrupted parties in $\mathcal{C}^{j+1}$ as Step 2 described above. Noting that $[v']_d^j$ and $[v']_d^{j+1}$ are identical, the adversary can obtain up to $2t \leq d$ (without loss of generality, $2t = d$) shares of $[v']_d^{j+1}$. According to the (d, n)-threshold secret sharing scheme, although all shares output by $\mathcal{S}_1$ are random, adversary can not tell the difference. The distribution of **Hybrid**$_1$ is identical to the distribution of **Hybrid**$_0$.

Hybrid$_2$: In this hybrid, $\mathcal{S}_1$ uses random elements as shares of $[v]_d^j$ of honest parties in $\mathcal{C}^j$. Recall that these shares are not used in **Hybrid**$_1$, the distribution of **Hybrid**$_2$ is identical to the distribution of **Hybrid**$_1$.

Hybrid$_3$: In this hybrid, $\mathcal{S}_1$ emulates $\mathcal{F}_{\mathsf{zero}}$ as described in Step 1. Note that these shares are not used in **Hybrid**$_2$ ($\mathcal{F}_{\mathsf{zero}}$ is used for building π_{reshare}). Therefore, the distribution of **Hybrid**$_3$ is identical to the distribution of **Hybrid**$_2$.

Note that **Hybrid**$_3$ is the execution in the ideal world, and the distribution of **Hybrid**$_3$ is identical to the distribution of **Hybrid**$_0$, the execution in the real world. Thus, the simulation of $\mathcal{F}_{\mathsf{rs-two}}$ is done.

Simulation of $\mathcal{F}_{\mathsf{reshare}}$: For the π_{reshare}, a simulator $\mathcal{S}_2$ can be constructed to simulate the behaviors of honest parties. Which can be built by modifying $\mathcal{S}_1$ slightly and repeating its execution. Notice that $\mathcal{F}_{\mathsf{zero}}$-hybrid model is actually not used in $\mathcal{S}_1$, we presect $\mathcal{S}_1'$ by just showing the difference from $\mathcal{S}_1$. Intuitively, $\mathcal{F}_{\mathsf{zero}}$ prevents an adversary from knowing the transmitted shares in the previous

committees, which is acceptable between two committees but becomes insecure in a multi-committee setting.

Step 1: $\mathcal{S}_1'$ emulates $\mathcal{F}_{\mathsf{zero}}$ and receives the shares of random zero sharing $[0]_d^j$ held by corrupted parties. Then $\mathcal{S}_1'$ randomly generates $d-t$ numbers as shares of $[0]_d^j$ hold by $d-t$ honest parties in $\mathcal{C}^j$. Furthermore, $\mathcal{S}_1$ can reconstruct and generates all random shares hold by honest parties in $\mathcal{C}^j$.

Step 2: Same with $\mathcal{S}_1$.

And only the $\mathbf{Hybrid}_1$ needs modification.

$\mathbf{Hybrid}_1$: In this hybrid, $\mathcal{S}_1'$ computes the shares of corrupted parties in $\mathcal{C}^{j+1}$ as Step 2 described above. Consider that shares of $[0]_d^j$ hold by honest parties in $\mathcal{C}^j$ are random, so adversary can not tell the difference between the generated $[v']_d^j = [v]_d^j + [0]_d^j$ and random number as shown in Step 2. Noting that $[v']_d^j$ and $[v']_d^{j+1}$ are identical, the adversary can obtain up to $2t \leq d$ (without loss of generality, $2t = d$) shares of $[v']_d^{j+1}$. According to the (d, n)-threshold secret sharing scheme, although all shares output by $\mathcal{S}_1'$ are random, adversary can not tell the difference. The distribution of $\mathbf{Hybrid}_1$ is identical to the distribution of $\mathbf{Hybrid}_0$.

By repeatly using $\mathcal{S}_1'$, the distribution of execution π_{reshare} in the real-world hybrid is identical to the distribution of the ideal-world hybrid. ■

B.6 Proof of Theorem 3

Proof. A simulator $\mathcal{S}$ will be constructed to simulate the behaviours of honest parties and honest clients except for the last communication round (Output phase). The Simulator $\mathcal{S}$ maintains awareness of the active committee through the reception of $(\mathsf{Init}, \mathcal{C})$ and $(\mathsf{Next}-\mathsf{Committee}, \mathcal{C})$ inputs from $\mathcal{F}_{\mathsf{DABB}}$, thereby enabling simulation of inter-committee communication corresponding to the circuit gates by using $\mathcal{F}_{\mathsf{commit}}$. The simulation proceeds as follows:

Simulation for Input:

For an input belonging to an honest client, $\mathcal{S}$ randomly chooses x as the input and generates the shares of it. Then, $\mathcal{S}$ sends t of these shares to the corrupted parties.

Note that, for each input of every gate, $\mathcal{S}$ knows $t+1$ shares then learns the shares held by corrupted parties by reconstructing whole shares.

Simulation for Linear Combination: On input $(\mathsf{LC}, \mathrm{id}_z, \mathbf{id}_x, \boldsymbol{a})$, considering that the computation of 'Linear combination' is local, $\mathcal{S}$ only needs to simulate π_{reshare}. According to Lemma 3, $\mathcal{S}$ can invoke $\mathcal{F}_{\mathsf{reshare}}$ to simulate it.

Simulation for Multiplication: On input $(\mathsf{Mult}, \mathrm{id}_z, \mathrm{id}_x, \mathrm{id}_y)$, $\mathcal{S}$ simulates the behavior of honest parties in π_{mult} for each gate and committee. Recall that for each gate with all the input sharings prepared, $\mathcal{S}$ will know the shares held by corrupted parties in $\mathcal{C}^j$. We describe the strategy of $\mathcal{S}$ step by step.

1. According to Lemma 4, $\mathcal{S}$ can simulate $\pi_{\mathsf{double-rand}}$ by emulating $\mathcal{F}_{\mathsf{double-rand}}$. Furthermore, $\mathcal{S}$ knows the shares $[x]_d^j, [y]_d^j, [r]_d^j$ and $[r]_{2d}^j$ held by corrupted parties in $\mathcal{C}^j$.
2. $\mathcal{S}$ computes the shares of $[e]_{2d}^j = [x]_d^j \cdot [y]_d^j + [r]_{2d}^j$ held by corrupted parties. Then, $\mathcal{S}$ samples random e as the computing result and samples random elements as shares of $[e]_{2d}^j$ of honest parties.
3. According to Lemma 3, $\mathcal{S}$ can simulate π_{reshare} by emulating $\mathcal{F}_{\mathsf{reshare}}$. Moreover, $\mathcal{S}$ knows the shares of $[r]_d^{j+2}$ of corrupted parties in $\mathcal{C}^{j+2}$.
4. Depending on whether P_{king} is honest, there are two cases:
   - If P_{king} is corrupted, $\mathcal{S}$ sends the randomly generated shares of $[e]_{2d}^j$ of honest parties to P_{king} and receives e' sent by P_{king} ($e' := 0$ if nothing is received).
   - If P_{king} is honest, $\mathcal{S}$ sends e to corrupted parties in $\mathcal{C}^{j+2}$ and set $e' = e$.
5. Last, $\mathcal{S}$ computes the shares of $[z]_d^{j+2} = e' - [r]_d^{j+2}$ held by corrupted parties. Then $\mathcal{S}$ sends shares of z to the $\mathcal{F}_{\mathsf{DABB}}$.

Hybrid Arguments. Now, we show that $\mathcal{S}$ perfectly simulates the behaviors of honest parties except for the last communication round. Consider the following hybrids.

Hybrid$_0$: The execution in the real world.

Hybrid$_1$: In this hybrid, Simulator $\mathcal{S}$ computes the input of corrupted clients. The distribution of **Hybrid**$_1$ is identical to **Hybrid**$_0$.

Hybrid$_2$: In this hybrid, Simulator $\mathcal{S}$ simulates all the π_{reshare} by invoking $\mathcal{F}_{\mathsf{reshare}}$. According to Lemma 4, the distribution of **Hybrid**$_2$ is identical to **Hybrid**$_1$.

Hybrid$_3$: In this hybrid, $\mathcal{S}$ simulates π_{mult} by dividing it into several hybrids:

Hybrid$_{3,1}$: In this hybrid, $\mathcal{S}$ computes the shares of corrupted parties as described above. Then $\mathcal{S}$ sends the shares of $[z]_d^{j+2}$ held by corrupted parties to $\mathcal{F}_{\mathsf{DABB}}$. The distribution of **Hybrid**$_{3,1}$ is identical to the distribution of **Hybrid**$_2$.

Hybrid$_{3,2}$: In this hybrid, $\mathcal{S}$ computes according to the situations of P_{king}:

- When P_{king} is corrupted, $\mathcal{S}$ sends random elements as shares of $[e]_{2d}^j$ of honest parties. Consider that $[r]_d^j$ and $[r]_{2d}^j$ are uniformly random. The distribution of $[e]_{2d}^j$ is identical to **Hybrid**$_{3.1}$. Moreover, $\mathcal{S}$ can learn the value e' reconstructed by P_{king} (*malicious* adversary may broadcast wrong value). Consider $\mathcal{S}$ will not send anything after receiving the e', the distribution of **Hybrid**$_{3,2}$ is identical to that of **Hybrid**$_{3.1}$.
- When P_{king} is honest, corrupted parties in $\mathcal{C}^j$ can change the multiplication result by sending incorrect shares of $[e]_{2d}^j$ to P_{king}. However, e received by corrupted parties in $\mathcal{C}^{j+2}$ is totally random. Consider adversary can only know t shares of $[e]_{2d}^j$ and has no information of e, the distribution of **Hybrid**$_{3,2}$ is identical to **Hybrid**$_{3.1}$.

Therefore, the distribution of **Hybrid**$_{3,2}$ is identical to the distribution of **Hybrid**$_{3,1}$.

Table 4. Comparison of communication time cost for one million multiplication gates between our protocol and state-of-art traditional protocol

Protocol	Depth	Number of parties				
		5	9	13	17	21
Ours	1k	2.5307	2.3412	2.3945	2.3465	2.3219
	5k	6.2470	6.0429	6.3347	6.2161	6.1634
	10k	11.2083	10.9456	11.2602	11.0487	11.0017
ATLAS [20]	1k	1.5807	1.6117	1.6199	1.6293	1.6330
	5k	5.2721	5.4500	5.5021	5.5556	5.5784
	10k	9.8868	10.2480	10.3551	10.4639	10.5104

Note: One million multiplication gates are computed on a LAN and measured in seconds.

Hybrid$_{3,3}$: In this hybrid, $\mathcal{S}$ emulates $\mathcal{F}_{\mathsf{double-rand}}$ to simulate $\pi_{\mathsf{double-rand}}$ according to Lemma 4 and does not generate the shares of honest parties. Note that these shares are not used in **Hybrid**$_{3,2}$. Therefore, the distribution of **Hybrid**$_{3,3}$ is identical to the distribution of **Hybrid**$_{3,2}$.

All in all, the end **Hybrid**$_{3,3}$ of **Hybrid**$_3$ is identical to the distribution of **Hybrid**$_2$.

Hybrid$_4$: In this hybrid, $\mathcal{S}$ simulates the input phase. For every input of honest parties, $\mathcal{S}$ randomly chooses a number as the input and generates the shares of it. Note that the honest parties' inputs are not used in **Hybrid**$_3$. Therefore, the distribution of **Hybrid**$_4$ is the same as **Hybrid**$_3$.

Note that **Hybrid**$_4$ is the execution in the ideal world, and the distribution of **Hybrid**$_4$ is identical to the distribution of **Hybrid**$_0$, the execution in the real world. ■

C Efficiency Comparison with traditional MPC

To emphasize the efficiency advantages of the protocol proposed in this paper, we conduct a simulation comparison with ATLAS [20], the state-of-the-art traditional MPC scheme, within a local area network (LAN) environment under a semi-honest adversary[12]. The results, shown in Table 4, are expressed in seconds. These simulations are performed using the event simulator OMNeT++ [26]. We assume each pair of participants communicates over channels with latency ranging from 0.4 ms to 0.5 ms and a channel rate of 100 Mbps. Computations are based on a finite field of size $2^{31} - 1$, with each field element represented using 32 bits for convenience. We evaluate the communication time required to compute a circuit consisting of $1,000,000$ multiplication gates at various circuit depths. Additionally, since our protocol does not consider on-the-fly committee establishment, we assume that committee formation is completed instantly for simulation purposes.

[12] The code for our protocol is publicly accessible at `https://github.com/stillalive-HaHaHa/Fluid-protocol`.

In practical computing environments, the time required for a single message transmission is composed of communication delay and data transfer time. The communication delay is determined solely by network conditions and remains unaffected by the communication complexity of the protocol. Therefore, it can be observed that the actual communication overhead of our proposed Fluid MPC protocol is only marginally greater than that of ATLAS. This demonstrates that the functionality of Fluid MPC can be achieved with relatively low cost, underscoring its practical applicability and value.