Protocol Composition; The p-Party GMW Protocol

CS 507, Topics in Cryptography: Secure Computation

David Heath

Fall 2025

To review, we have now constructed two general-purpose, semi-honest secure MPC protocols:

- Our first protocol achieved security in the three-party setting, assuming no two parties collude. That protocol was *perfectly* secure. That is, it required no assumptions on the computational capabilities of the parties, and we were able to construct per-party simulators that each produce views identically-distributed to the party's real-world view.
- Our second protocol achieve security in the two-party setting by leverage a 1-out-of-2 Oblivious Transfer (OT) protocol. Recall that we showed one way to instantiate OT, in particular by leveraging presumed hardness of the *Decisional Diffie-Hellman* (DDH) problem. At a high level, we assumed some particular computational task is hard, and we used this to construct OT.

Thus, our two protocols are formally *incomparable*; the former protocol makes an assumption on collusions, the latter an assumption on the computational power of an adversary.

Today, we will clean up our understanding of these protocols.

- First, we will discuss how the second protocol can be proved secure. While intuitively, it is clear, we have not yet formally showed a proof.
- Second, we will discuss about how to extend our second to any number of parties p. This p-party protocol is called the GMW protocol, after Goldreich, Micali, and Wigderson who discovered the protocol [GMW87].
- Third, we will discuss fundamental relationships between our protocols and OT.

Protocol Composition

Let's recall our definition of semi-honest security:

Definition 1 (Semi-Honest Security)**.** *Let f be a (possibly-randomized) n-party functionality running between parties $P_0, ..., P_{n-1}$. Let P_i have input x_i We say that a protocol Π securely computes f in the presence of a semi-honest adversary if for each subset of parties $C \subseteq \{0, ..., n-1\}$, there exists a probabilistic polynomial time algorithm $\texttt{Sim}_C$—called a simulator—such that for all inputs $x_0, ..., x_{n-1}$, the following hold:*

$$\left\{ \bigcup_{i \in C} \texttt{View}_{P_i}^{\Pi(r)}(x_i, y_i), \texttt{Output}^{\Pi(r)}(x_0, ..., x_{n-1}) \right\}$$
$$\stackrel{c}{=} \left\{ \texttt{Sim}_C\left(\bigcup_{i \in C} (x_i, y_i) \right) \quad \textit{where} \quad (y_0, ..., y_{n-1}) \leftarrow f(x_0, ..., x_{n-1}) \right\}$$

Above, $\Pi(r)$ denotes to run Π with fixed randomness r s.t. the corrupted subsets' view and the output are the result of the same protocol execution.

As our first goal, we would like to sketch how to show security of our OT-based 2PC protocol. The problem is that our protocol relies is built in terms of an underlying OT protocol; somehow, our proof (and namely our simulators) must account for both the messages sent as a result of the underlying OT protocol, and the messages sent due to the top-level protocol.

Namely, we have some protocol Π, say our 2PC protocol, which is built *in terms of* some second protocol ρ, say our OT protocol.

Heres one ways we might prove security of Π:

> **The monolithic approach.** To prove security of Π, we inline the code of all calls to ρ, then provide simulators that describe both the view internal to calls to ρ, and the view in parts of Π that are not calls to ρ.

This approach would technically work, and indeed it even has the advantage that the approach is extremely simple—we don't need any new tools to take this approach!

On the other hand, this monolithic has some serious and glaring flaws.

- First, the proof of security for Π is not "reusable". In particular, suppose that tomorrow, someone devises a better 1-out-of-2 OT protocol σ. If we wish to reimplement Π in terms of σ instead of ρ, we will need a new proof of security – after all, many of the messages sent in the protocol will have changed!

- Second, the proof of security for Π is "complex". When writing the monolithic proof of Π's security, we must simultaneously keep in mind what the party sees as a result of Boolean gate handling, and as a result of OT messages (e.g., DDH group elements). This makes writing the proof harder and more error-prone.

What would preferable is clearly a **modular approach** to security. We would like to reason about Π without concerning ourselves with the details of ρ. In particular, we explained our 2PC protocol, we used OT in a black-box manner, making statements like "the parties call 1-out-of-2". Thus our description of Π is modular, and we would like our proof of security to *inherit* that modularity.

Fortunately, it is possible to think in this way. Namely, there exists a relatively simple *composition theorem* for semi-honest protocols.

Recall that we define security of protocols with respect to ideal functionalities. I.e., perhaps protocol ρ securely instantiates some functionality g, and we would like to prove that Π—which calls ρ—securely instantiates some functionality f. The composition theorem states that it is sufficient for us to think of Π as calling ρ's ideal functionality g, and composed security falls out for free. This is exactly what we want! We can reason about 1-out-of-2 OT as if it is a black-box functionality.

The protocol Π written in terms of ideal g is sometimes said to be formalized in a *hybrid model*. For instance our 2PC protocol is in the OT-hybrid model: it assumes OT is implemented as an ideal functionality.

When we instantiate the black-box g with a real-world protocol ρ, security is already proved. Moreover, if tomorrow we find a new secure OT protocol σ, we can substitute σ instead, and no new proof of security is necessary.

Theorem 1 (Semi-honest protocol composition)**.** *Let Π be a protocol reducible to ideal functionality g. Suppose Π securely computes ideal functionality f. Let ρ be a protocol that securely computes g. Then $\Pi[\rho/g]$ securely computes f. Namely, if we replace all calls to ideal functionality g by protocol ρ, then the resulting composed protocol also securely computes f.*

Proof Sketch. Recall that semi-honest security requires existence of a simulator. Thus (1) to show this theorem, we must construct a simulator for every subset of corrupted parties, and (2) we can assume that we *already have* simulators (for the same subset of parties) for both Π (written in terms of g) and ρ. Let's call those simulators S_Π and S_ρ. Our goal is to construct a simulator $S_{\Pi[\rho/g]}$.

In short, the simulator $S_{\Pi[\rho/g]}$ is straightforward to construct. $S_{\Pi[\rho/g]}$ first calls S_Π to produce a view, including interactions involving ideal functionality g. In particular, we can extract from this simulated view

the input/output behavior of each interaction with g. Let the input (resp. output) be x (resp. y). For each such call to g, we can call $S_\rho(x, y)$ to obtain a view corresponding to real-world protocol ρ. We now replace the part of the view corresponding to g with newly generated view of ρ. Once all such calls to g are replaced, we have our simulator $S_\Pi[\rho/g]$. This simulator's output is indeed indistinguishable from the real-world view; this can be shown formally by arguing that, if it is not indistinguishable, then one can distinguish the real-world view of ρ from the output of simulator S_ρ, a contradiction.

For a careful proof, see [Gol01] Theorem 7.3.3. □

By using this theorem, we can easily prove security of our OT-based 2PC protocol:

Lemma 1. *Let* Π *be the 2PC protocol described in Lecture 5.* Π *is perfectly semi-honest secure in the OT-hybrid model.*

Note that we can claim Π is perfectly secure. Of course, we lose perfect security once we instantiate the OT functionality by a particular computationally-secure protocol. This is implied by the protocol composition theorem.

Exercise 1. *Write out simulators for* Π *in the semi-honest model. Checkt that you understand how the composition theorem will apply to those simulators.*

The Multiparty GMW Protocol

So far, our OT-based protocol only works for two parties. However, it can be naturally extended to work for any number of parties p. Crucially, even when $p = 3$, this protocol is *qualitatively different* than our previous 3PC protocol. In particular, in this new protocol, each party is individually assured (semi-honest) privacy, even when an *arbitrary subset* of its peers collude. This generalized p-party protocol is called the *GMW protocol* [GMW87].

The transformation of our existing protocol is natural. In particular, recall that our existing protocol simply manipulates XOR secret shares, and recall that XOR secret shares naturally generalize to p parties. Thus, our gate-by-gate handling is exactly the same, with the exception of AND gates.

Recall that we instantiated AND gates by using Beaver triples, and that trick also naturally makes sense for p parties. The only question is as to where these p-party triples come from. In particular, we wish to construct a sharing of form $[\alpha, \beta, \alpha \cdot \beta]$ where α, β are uniform bits, and where each of the p parties holds a share of each bit.

Recall that to solve this in the two-party setting, we made use of the following identity:

$$\begin{aligned}\alpha \cdot \beta &= (\alpha_0 \oplus \alpha_1) \cdot (\beta_0 \cdot \beta_1) \\ &= \alpha_0 \cdot \beta_0 \oplus \alpha_0 \cdot \beta_1 \oplus \alpha_1 \cdot \beta_0 \oplus \alpha_1 \cdot \beta_1\end{aligned}$$

From here, we used OT to compute each cross term $\alpha_i \cdot \beta j$ for $i \neq j$.

This same identity naturally generalizes to p parties, though the number of required OTs is higher:

$$\alpha \cdot \beta = \left(\bigoplus_{i \in [p]} \alpha_i \right) \cdot \left(\bigoplus_{i \in [p]} \beta_i \right) = \bigoplus_{i,j \in [p]} \alpha_i \cdot \beta_j$$

Thus, to generate a uniformly-shared Beaver triple, (1) each party P_i uniformly samples shares α_i, β_i, then (2) each pair of distinct parties P_i, P_j perform two 1-out-of-2 OTs (as described in Lecture 5) to compute secret shares of $\alpha_i \cdot \beta_j$. By combining the results of OTs, the parties obtain $[\alpha \cdot \beta]$.

By using this triple generation for each AND gate, the parties obtain a semi-honest p-party protocol for arbitrary Boolean circuits.

Theorem 2 (GMW Protocol)**.** *In the OT hybrid model, there exists a p-party MPC protocol that is perfectly secure against a semi-honest adversary corrupting any subset of parties.*

To re-emphasize, we can use the composition theorem to instantiate OT, e.g. using DDH. For instance:

Corollary 1. *Assuming DDH is hard for some family of groups, there exists a p-party MPC protocol that is computationally secure against a semi-honest adversary corrupting any subset of parties.*

Some Basic Results about MPC

Theorem 2 is the first true result we have seen that asserts the feasibility of (semi-honest) MPC *in general*. So it's a good moment to reflect on some basic theory. What Theorem 2 can really be seen as establishing is the following:

> The existence of an OT protocol is sufficient for MPC to exist. Or, more shortly, OT is sufficient to achieve MPC.

With a bit of thought, one can also arrive at the following realization:

> The existence of an OT protocol is *necessary* for MPC to exist.

For those unfamilar with such statements, the above claim might seem out of our current reach to prove, but it is actually straightforward. Indeed, the above claim can be interpreted as the following informal piece of mathematics:

$$\neg\texttt{OT} \implies \neg\texttt{MPC}$$

That is, if OT does not exist, then neither does MPC. Let's take the contrapositive:

$$\texttt{MPC} \implies \texttt{OT}$$

I.e., the existence of MPC is sufficient for OT to exist. But this is easy to show! Indeed, we can use an MPC protocol to implement *any* computable function, and OT is surely such a function. In other words, given any MPC protocol, it is easy to build OT, and hence OT must exist for MPC to exist.

Another observation: We started with a three-party protocol that worked against only one corrupted party, but that made no computational assumptions. I.e., the protocol is information-theoretically and unconditionally secure. Can we hope for such a protocol that does *not* make an assumption about the number of corrupted parties? While we are not ready to prove this now, the answer is unfortunately no:

> An honest majority is both necessary and sufficient to achievE information-theoretically-secure MPC [GMW87].

That is, if we want to forgo cryptographic assumptions like DDH, we must assume that a majority of the parties do not collude. On the other hand, once we make a cryptographic assumption, it becomes possible to achieve MPC for *any* number of parties.

Next Time

So far, all of our protocols incur high *round complexity*. These protocol are based on an interactive trick involving XOR secret sharing of bits. Next time, we will begin studying a different route to MPC protocols call the *Garbled Circuit* (GC) technique. GC has the amazing property that, regardless of the desired computation, we can construct protocols that run in a small constant number of rounds of communication. The trade-off will be an increase in overall bandwidth consumption.

References

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

[Gol01] Oded Goldreich. *Foundations of cryptography: volume 2, basic applications*, volume 2. Cambridge university press, 2001.