Secret-sharing based secure multiparty computation with malicious adversaries

05.11.2019
Error-correcting codes

- An \((n, t, d)\)-code over a set \(X\) is a mapping \(C : X^t \rightarrow X^n\), such that for all \(x_1, x_2 \in X^t\), \(x_1 \neq x_2\) implies that \(C(x_1)\) and \(C(x_2)\) differ in at least \(d\) positions.
- An element \(x \in X^t\) is encoded as \(y = C(x) \in X^n\) and transmitted. During transmission, errors may occur in some positions of \(y\).
- A \((n, t, d)\)-code can detect at most \(d - 1\) errors.
- A \((n, t, d)\)-code can correct at most \((d - 1)/2\) errors.
- Efficiency is another question, though.
Error-correcting codes

- An \((n, t, d)\)-code over a set \(X\) is a mapping \(C : X^t \rightarrow X^n\), such that for all \(x_1, x_2 \in X^t\), \(x_1 \neq x_2\) implies that \(C(x_1)\) and \(C(x_2)\) differ in at least \(d\) positions.

- An element \(x \in X^t\) is encoded as \(y = C(x) \in X^n\) and transmitted. During transmission, errors may occur in some positions of \(y\).

- A \((n, t, d)\)-code can detect at most \(d - 1\) errors.

- A \((n, t, d)\)-code can correct at most \((d - 1)/2\) errors.

- Efficiency is another question, though.

- In a linear code, \(X\) is a field and \(C\) is a linear mapping between vector spaces \(X^t\) and \(X^n\).

- For linear codes, \(d \leq n - t + 1\).
Reed-Solomon codes

- Reed-Solomon codes are linear codes over some finite field $\mathbb{F}$.
- To encode $t$ elements of $\mathbb{F}$ as $n$ elements of $\mathbb{F}$, fix $n$ different elements $c_1, \ldots, c_n \in \mathbb{F}$.
- Interpret the source word $(f_0, \ldots, f_{t-1})$ as a polynomial $p(x) = \sum_{i=1}^{t-1} f_i x^i$.
- Encode it as $(p(c_1), \ldots, p(c_n))$.
- For Reed-Solomon codes, $d = n - t + 1$.
- Hence they can correct up to $(n - t)/2$ errors.
Decoding Reed-Solomon codes

- Suppose that the original codeword was \((s_1, \ldots, s_n)\), corresponding to the polynomial \(p\).
- But we received \((\tilde{s}_1, \ldots, \tilde{s}_n)\).
  - We assume it has at most \((n - t)/2\) errors.
- Find the coefficients for polynomials \(q_0\) and \(q_1\), such that
  - Degree of \(q_0\) is at most \((n + t - 2)/2\). Degree of \(q_1\) is at most \((n - t)/2\).
  - For all \(i \in \{1, \ldots, n\}\): \(q_0(c_i) - q_1(c_i) \cdot \tilde{s}_i = 0\).
  - \(q_0\) and \(q_1\) are not both equal to 0.
- Then \(p = q_0 / q_1\).
- In general, there are more equations than variables, but \(\tilde{s}_i\) are not arbitrary.
Correctness of decoding

Such polynomials \( q_0, q_1 \) exist:

- \((s_1, \ldots, s_n), (\tilde{s}_1, \ldots, \tilde{s}_n)\) — original and received codewords. Let \( E \) be the set of \( i \), where \( s_i \neq \tilde{s}_i \). Then \( |E| \leq (n - t)/2 \).
- Let \( k(x) = \prod_{i \in E} (x - c_i) \). Then \( \deg k \leq (n - t)/2 \).
- Take \( q_1 = k \) and \( q_0 = p \cdot k \). Then \( \deg q_0 \leq (n + t - 2)/2 \).
- For all \( i \in \{1, \ldots, n\} \) we have

\[
q_0(c_i) - q_1(c_i) \cdot \tilde{s}_i = k(c_i)(p(c_i) - \tilde{s}_i) = k(c_i)(s_i - \tilde{s}_i) = \\
\begin{cases} 
  k(c_i)(s_i - s_i) = 0, & i \notin E \\
  0 \cdot (s_i - \tilde{s}_i) = 0, & i \in E
\end{cases}
\]
Correctness of decoding

If $q_0$ and $q_1$ satisfy the equalities and upper bounds on degrees, then $p = q_0 / q_1$:

- Let $q'(x) = q_0(x) - q_1(x)p(x)$. Degree of $q'$ is at most $(n + t - 2)/2$.
- For each $i \notin E$,
  
  $q'(c_i) = q_0(c_i) - q_1(c_i)p(c_i) = q_0(c_i) - q_1(c_i)\tilde{s}_i = 0$.
  
  $1 \leq i \leq n$.

- The number of such $i$ is at least $n - (n - t)/2 = (n + t)/2$.
- Thus the number of roots of $q'$ is larger than its degree. Hence $q' = 0$.

- $q_0 - q_1 \cdot p = 0$. 

05.11.2019
MPC with no errors

- The number of corrupted players is at most $t - 1 < n/3$.
- To distribute inputs, each party first commits to his input and then shares the commitment.
- Shamir’s scheme is used for both committing and sharing.
  - Hence the commitments are homomorphic.
  - For a value $a$, let $[a]_i$ denote the commitment of $P_i$ to $a$. The commitment is distributed, hence $[a]_i = ([a]_i^1, \ldots, [a]_i^n)$, with $P_j$ holding the piece $[a]_i^j$. 
Commitments

We need the following functionalities:

- **Commit**: $P_i$ commits to a value $a$.
  - $[a]_i$ is a sharing of $a$ using $(n, t)$-secret sharing.
  - Followed by a proof that the degree of the polynomial is $\leq (t - 1)$.

- **Open** and **OpenPrivate**: opens a commitment.
  - Everybody broadcasts his share or sends it privately to the party that is supposed to open it.
  - Errors can be corrected.

- **Linear Combination**: several commitments of the same party (or different parties) are linearly combined.
  - Everybody performs the same combination on the shares he’s holding.
Commitments

- **Transfer**: turns $P_i$’s commitment $[a]_i$ into $P_j$’s commitment $[a]_j$. Party $P_j$ learns $a$.
  - OpenPrivate $a$ for $P_j$.
  - $P_j$ Commits $a$, giving $[a]_j$.
  - Find the Linear Combination $[a]_i - [a]_j$ and Open it; check that it is 0.

- **Share**: applies Shamir’s secret sharing to a committed value $[a]_i$.
  - $P_i$ generates the values $a_1, \ldots, a_{t-1}$ and Commits to them.
  - $s_i = a + \sum_{j=1}^{t-1} a_j i^j$. These Linear Combinations of $[a]_i$ and $[a_1]_i, \ldots, [a_{t-1}]_i$ are computed, resulting in commitments $[s_1]_i, \ldots, [s_n]_i$.
  - Commitment $[s_j]_i$ is Transfered to $[s_j]_j$. 
Commitments

- **Multiply.** Given \([a]_i\) and \([b]_i\), the party \(P_i\) causes the computation of \([c]_i\), where \(c = a \cdot b\).
- Compute \(c\) and Commit to it.
- Share \([a]_i\) and \([b]_i\), giving \([s^a]_1, \ldots, [s^a]_n\) and \([s^b]_1, \ldots, [s^b]_n\).
  - Let the polynomials be \(f^a\) and \(f^b\).
- Let \(f^c(x) = f^a(x) \cdot f^b(x) = c + \sum_{j=1}^{2t-2} c_j x^j\). Party \(P_i\) Commit to \(c_1, \ldots, c_{2t-2}\).
- Compute \([f^c(1)]_i, \ldots, [f^c(n)]_i\) as Linear Combinations of \([c]_i\) and \([c_1]_i, \ldots, [c_{2t-2}]_i\).
- OpenPrivate \([f^c(j)]_i\) to \(P_j\). He checks that \(s^a_j \cdot s^b_j = f^c(j)\). If not, broadcast complaint and Open \([s^a]_j, [s^b]_j\).
- If \(P_j\) complains then \(P_i\) Opens \([f^c(j)]_i\). Either \(P_i\) or \(P_j\) is disqualified.

**Exercise.** Show that if \(P_i\) cheats then there will be a complaint.
MPC

- For each wire, the value on it is shared and the parties have commitments to those shares.
- Start: each party **Commits** to his input and then **Shares** it.
- Addition gates: **Linear Combination** is used to add the shares of values on incoming wires.
- Multiplication gates: the shares of the values on incoming wires are **Multiplied** together. These products are **Shared** and those shares are recombined into the shares of the product, using **Linear Combination**.
  - i.e. Gennaro-Rabin-Rabin multiplication is performed on committed shares.
- End: the shares of a value that a party is supposed to learn are **Opened Privately** to this party.
Commit: proving the degree of a polynomial

- $P_i$ wants to commit to a value $a$ using a random polynomial $f$, where $\deg f \leq t - 1$ and $f(0) = a$. A party $P_j$ learns $[a]_i^j = f(j)$.
- $P_i$ has to convince others that $f$ has a degree at most $t - 1$. 
Commit: proving the degree of a polynomial

- $P_i$ wants to commit to a value $a$ using a random polynomial $f$, where $\deg f \leq t - 1$ and $f(0) = a$. A party $P_j$ learns $[a]_i^j = f(j)$.
- $P_i$ has to convince others that $f$ has a degree at most $t - 1$.
- $P_i$ randomly generates a two-variable symmetric polynomial $F$, such that $F(x, 0) = f(x)$ and the degrees of $F$ with respect to $x$ and $y$ are $\leq (t - 1)$. I.e.
  - randomly generate coefficients $c_{kl} \in \mathbb{F}$, where $1 \leq l \leq k \leq t - 1$;
  - Let $c_{00} = a$. Let $c_{i0}$ be the coefficient of $x^i$ in $f$.
  - Let $c_{lk} = c_{kl}$ for $l \geq k$.
  - Let $F(x, y) = \sum_{k=0}^{t-1} \sum_{l=0}^{t-1} c_{kl} x^k y^l$.
- $P_i$ sends to $P_j$ the polynomial $F(x, j)$ (i.e. its coefficients). The share $[a]_i^j$ of $P_j$ is $F(0, j) = F(j, 0) = f(j)$. 
Commit: proving the degree of a polynomial

- $P_j$ and $P_k$ compare the values $F(k, j)$ and $F(j, k)$. If they differ, they broadcast a complaint $\{j, k\}$.
- $P_i$ answers to “complaint $\{j, k\}$” by publishing the value $F(j, k)$ (which is the same as $F(k, j)$).
- If $P_j$ (or $P_k$) has a different value then he broadcasts “disqualify $P_i$”.
- $P_i$ responds to that by broadcasting $F(x, j)$.
- All parties $P_l$ check that $F(l, j) = F(j, l)$. If not, broadcast “disqualify $P_i$”. Again $P_i$ responds by broadcasting $F(x, l)$, etc.
- If there are at least $t$ disqualification calls then $P_i$ is disqualified.
- Otherwise the commitment is accepted and parties update their shares with the values that $P_i$ had broadcast.
Soundness and privacy

- **Exercise.** Show that if $P_i$ is honest then the adversary does not learn anything beyond the polynomials $F(x,j)$, where $P_j$ is corrupt.

- **Exercise.** Show that if the commitment is accepted then the shares $[a]^j_i$ of honest parties are lay on a polynomial of degree $\leq (t - 1)$. 
Consistency of shares

Let $B \subseteq \{1, \ldots, n\}$ be the set of indices of honest parties. We must show that there exists a polynomial $g$ of degree at most $t - 1$, such that $g(j) = [a]_i^j = F(0, j)$ for all $j \in B$.

Let $C \subseteq B$ be the indices of honest parties that did not accuse the dealer. **Exercise.** How large must $C$ be?

**Exercise.** Show that for all $j \in B$ and $k \in C$ we have $F(j, k) = F(k, j)$ at the end of the protocol.

Let $r_k$, where $k \in C$ be the Lagrange interpolation coefficients for polynomials of degree $\leq t - 1$. I.e. $h(0) = \sum_{k \in C} r_k h(k)$ for all such polynomials $h$. **Exercise.** Why do such $r_k$ exist?

**Exercise.** Show that $g(x) = \sum_{k \in C} r_k \cdot F(x, k)$ is the polynomial we’re looking for.
Linear Secret Sharing Schemes (LSSS)

Monotone Span Programs (MSP)

MSP for parties in set $\mathcal{P}$ consists of
- $M \in \mathbb{F}^{m \times d}$ with $m \geq d$; $\varepsilon \in \mathbb{F}^d$; $\psi : \{1, \ldots, m\} \to \mathcal{P}$

Secret-sharing $s \in \mathbb{F}$ among $\mathcal{P}$ using a MSP

- Pick $x \xleftarrow{\$} \mathbb{F}^m$ subject to $\langle x, \varepsilon \rangle = s$
- Let $s = M \cdot x$. Give $i$-th component of $s$ to party $\psi(i)$
- $\mathcal{Q} \subseteq \mathcal{P}$ is qualified if $\varepsilon$ is in the span of the rows of $M$ with indices in $\psi^{-1}(\mathcal{Q})$
  - Let $\lambda^\mathcal{Q}$ be such that $(M_{\psi^{-1}(\mathcal{Q})})^\top \lambda^\mathcal{Q} = \varepsilon$
  - Recovery: compute $\langle \lambda^\mathcal{Q}, s_{\psi^{-1}(\mathcal{Q})} \rangle$

Examples: Shamir’s sharing. The any-access-structure scheme.
Multiplicative LSSSs

Local tensor

Let \( s, s' \in \mathbb{F}^m \) and \( \psi : \{1, \ldots, m\} \rightarrow \mathcal{P} \). Define \( s \otimes_1 s' \) as the vector of all values \( s_i \cdot s_j \), where \( i, j \in \{1, \ldots, m\} \), \( \psi(i) = \psi(j) \).

Multiplicative LSSS

LSSS is multiplicative if

- There is a vector \( v \),
- Such that for all values \( s, s' \in \mathbb{F} \),
- For all sharings \( s \) of \( s \) and \( s' \) of \( s' \),
- We have \( \langle v, s \otimes_1 s' \rangle = s \cdot s' \)

Examples: Shamir’s sharing. Replicated sharing (“any-access-structure”, where formula is in CNF)
Honest majority $\Rightarrow$ LSSS detects errors

- Let MSP allow only a majority of parties to recover the secret.
- Then a minority $\mathcal{Q}$ cannot encode an error:
  - Let $s^\mathcal{Q}$ be such that $s_i \neq 0$ only if $\psi(i) \in \mathcal{Q}$. Then $s$ encodes 0 or $s$ is erroneous.
- With multiplication triples, can do MPC with security against active abort.
Proactive secret sharing

- Let $D$ be a secret that is distributed with Shamir’s secret sharing scheme, using the polynomial $f_\circ$ of degree $\leq t - 1$.
- Recomputing shares: change the polynomial to $f_\bullet$ with $f_\circ(0) = f_\bullet(0)$ in a random manner.
- Passive adversary:
  - each party $P_i$ generates a random polynomial $h_i$ with zero free term; sends $h_i(j)$ to $P_j$.
  - parties add the values they got to their current shares.
  - Thus $f_\bullet = f_\circ + h_1 + \cdots + h_n$.
- Active adversaries: use VSS. Only use $h$-s from honest parties.
- A party relieved from adversarial control needs to be repaired.
  - To repair $P_r$, construct a polynomial $f_\bullet + h$ where $h$ is a random polynomial with $h(r) = 0$.
  - Send to $P_r$ the shares corresponding to that polynomial.