Verifiable Secret Sharing Voting

Threshold Secret Sharing

Zelda has a **secret** $s \in \{0,1\}^n$.

Def: Let $1 \le t \le m$. (t, L)-**secret sharing** is a way for Zelda to give strings to $A_1, \ldots, A_L$ such that:

1. If any t get together than they can learn the secret.
2. If any $t-1$ get together they cannot learn the secret.

Cannot learn the secret Last lecture this was Info-Theoretic. This lecture we consider info-theoretic and comp-theoretic.

A Scenario

1. $(5,9)$ Secret Sharing.
2. The secret is s. $p \sim s$. Zelda picks rand $r_4, r_3, r_2, r_1 \in \mathbb{Z}_p$, forms the poly $f(x) = r_4x^4 + r_3x^3 + r_2x^2 + r_1x + s$.
3. For $1 \le i \le 9$ Zelda gives A_i $f(i)$.

A Scenario

1. $(5, 9)$ Secret Sharing.
2. The secret is s. $p \sim s$. Zelda picks rand $r_4, r_3, r_2, r_1 \in \mathbb{Z}_p$, forms the poly $f(x) = r_4x^4 + r_3x^3 + r_2x^2 + r_1x + s$.
3. For $1 \le i \le 9$ Zelda gives A_i $f(i)$.

A_2, A_4, A_7, A_8, A_9 get together. BUT the do not trust each other!

1. A_2 thinks that A_7 is a traitor!
2. A_7 thinks A_4 will confuse them just for the fun of it.
3. A_8 and A_9 got into a knife fight over who proved that the muffin problem always has a rational solution. (Used same knife that was used to cut the muffins in $\frac{5}{12}$:$\frac{7}{12}$ ratio.)
4. The list goes on

A Scenario

1. $(5, 9)$ Secret Sharing.
2. The secret is s. $p \sim s$. Zelda picks rand $r_4, r_3, r_2, r_1 \in \mathbb{Z}_p$, forms the poly $f(x) = r_4x^4 + r_3x^3 + r_2x^2 + r_1x + s$.
3. For $1 \le i \le 9$ Zelda gives A_i $f(i)$.

A_2, A_4, A_7, A_8, A_9 get together. BUT the do not trust each other!

1. A_2 thinks that A_7 is a traitor!
2. A_7 thinks A_4 will confuse them just for the fun of it.
3. A_8 and A_9 got into a knife fight over who proved that the muffin problem always has a rational solution. (Used same knife that was used to cut the muffins in $\frac{5}{12}$:$\frac{7}{12}$ ratio.)
4. The list goes on

Hence we need to VERIFY that everyone is telling the truth. This is called VERIFIABLE secret sharing, or VSS.

First Attempt at (t, L) VSS

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1 \in Z_p$
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \le i \le L$ Zelda gives A_i $f(i)$, g, g^s.
   (We think discrete log is HARD so s not revealed.)

Recover: The usual – any group of t can determine the polynomial f and hence the constant term.

Verify: Once a group has s they compute g^s and see if it matches.

First Attempt at (t, L) VSS

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1 \in Z_p$
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \le i \le L$ Zelda gives A_i $f(i)$, g, g^s.
   (We think discrete log is HARD so s not revealed.)

Recover: The usual – any group of t can determine the polynomial f and hence the constant term.

Verify: Once a group has s they compute g^s and see if it matches. If so then they **know** they have the correct secret.

First Attempt at (t, L) VSS

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1 \in Z_p$
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \le i \le L$ Zelda gives A_i $f(i)$, g, g^s.
   (We think discrete log is HARD so s not revealed.)

Recover: The usual – any group of t can determine the polynomial f and hence the constant term.

Verify: Once a group has s they compute g^s and see if it matches.
If so then they **know** they have the correct secret.
If no then they **know** someone is a **stinking rotten liar**

First Attempt at (t, L) VSS

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1 \in Z_p$
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \le i \le L$ Zelda gives A_i $f(i)$, g, g^s.
   (We think discrete log is HARD so s not revealed.)

Recover: The usual – any group of t can determine the polynomial f and hence the constant term.

Verify: Once a group has s they compute g^s and see if it matches.
If so then they **know** they have the correct secret.
If no then they **know** someone is a **stinking rotten liar**

1. If verify s there may still be two liars who cancel out.
2. If do not agree they do not know who the liar was.
3. Does not serve as a deterrent.

Second Attempt at (t, L) VSS

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1 \in \mathbb{Z}_p$.
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \leq i \leq L$ Zelda gives A_i $f(i)$.
5. Zelda gives to EVERYONE the values $g^{f(1)}$, ..., $g^{f(L)}$, g.
   (We think discrete log is HARD so $f(i)$ not revealed.)

Recover: The usual – any group of t can blah blah.

Verify: If A_i says $f(i) = 17$, they can all then check of g^{17} is what Zelda said $g^{f(i)}$ is.

Second Attempt at (t, L) VSS

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1 \in \mathbb{Z}_p$.
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \le i \le L$ Zelda gives A_i $f(i)$.
5. Zelda gives to EVERYONE the values $g^{f(1)}$, ..., $g^{f(L)}$, g.
   (We think discrete log is HARD so $f(i)$ not revealed.)

Recover: The usual – any group of t can blah blah.

Verify: If A_i says $f(i) = 17$, they can all then check of g^{17} is what Zelda said $g^{f(i)}$ is.
If so then they **know** A_i is truthful.

Second Attempt at (t, L) VSS

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1 \in \mathbb{Z}_p$.
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \le i \le L$ Zelda gives A_i $f(i)$.
5. Zelda gives to EVERYONE the values $g^{f(1)}$, ..., $g^{f(L)}$, g.
   (We think discrete log is HARD so $f(i)$ not revealed.)

Recover: The usual – any group of t can blah blah.

Verify: If A_i says $f(i) = 17$, they can all then check of g^{17} is what Zelda said $g^{f(i)}$ is.
If so then they **know** A_i is truthful.
If not then they **know** A_i is a **stinking rotten liar**.

Second Attempt at (t, L) VSS

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1 \in \mathbb{Z}_p$.
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \le i \le L$ Zelda gives A_i $f(i)$.
5. Zelda gives to EVERYONE the values $g^{f(1)}$, ..., $g^{f(L)}$, g.
   (We think discrete log is HARD so $f(i)$ not revealed.)

Recover: The usual – any group of t can blah blah.

Verify: If A_i says $f(i) = 17$, they can all then check of g^{17} is what Zelda said $g^{f(i)}$ is.
If so then they **know** A_i is truthful.
If not then they **know** A_i is a **stinking rotten liar**.

1. **PRO:** If someone lies they know right away.
2. **PRO:** Serves as a deterrent.
3. **CON:** L public strings A LOT!, may need to update.

Third Attempt at (t, L) VSS

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1 \in Z_p$,
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \le i \le L$ Zelda gives A_i $f(i)$.
5. Zelda gives to EVERYONE the values g^{r_1}, ..., $g^{r_{t-1}}, g^s, g$.
   (We think discrete log is HARD so r_i not revealed.)

Recover: The usual – any group of t can blah blah.
Verify: A_i reveals $f(i) = 17$. Group computes: g^{17} and:
$$(g^{r_{t-1}})^{i^{t-1}} \times (g^{r_{t-2}})^{i^{t-2}} \times \cdots \times (g^{r_1})^{i^1} \times (g^s)^{i^0}$$
$$= g^{r_{t-1}i^{t-1} + r_{t-2}i^{t-2} + \cdots + r_1 i^1 + s} = g^{f(i)}$$

Third Attempt at (t, L) VSS

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1 \in Z_p$,
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \le i \le L$ Zelda gives A_i $f(i)$.
5. Zelda gives to EVERYONE the values $g^{r_1}, \ldots, g^{r_{t-1}}, g^s, g$.
   (We think discrete log is HARD so r_i not revealed.)

Recover: The usual – any group of t can blah blah.

Verify: A_i reveals $f(i) = 17$. Group computes: g^{17} and:
$(g^{r_{t-1}})^{i^{t-1}} \times (g^{r_{t-2}})^{i^{t-2}} \times \cdots \times (g^{r_1})^{i^1} \times (g^s)^{i^0}$
$= g^{r_{t-1}i^{t-1} + r_{t-2}i^{t-2} + \cdots + r_1 i^1 + s} = g^{f(i)}$
If this is g^{17} then A_i is truthful.

Third Attempt at (t, L) VSS

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1 \in Z_p$,
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \le i \le L$ Zelda gives A_i $f(i)$.
5. Zelda gives to EVERYONE the values g^{r_1}, $\ldots$, $g^{r_{t-1}}, g^s, g$.
   (We think discrete log is HARD so r_i not revealed.)

Recover: The usual – any group of t can blah blah.

Verify: A_i reveals $f(i) = 17$. Group computes: g^{17} and:

$(g^{r_{t-1}})^{i^{t-1}} \times (g^{r_{t-2}})^{i^{t-2}} \times \cdots \times (g^{r_1})^{i^1} \times (g^s)^{i^0}$

$= g^{r_{t-1}i^{t-1}+r_{t-2}i^{t-2}+\cdots+r_1 i^1+s} = g^{f(i)}$

If this is g^{17} then A_i is truthful.

If not then A_i is dirty stinking liar.

Third Attempt at (t, L) VSS

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1 \in Z_p$,
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \le i \le L$ Zelda gives A_i $f(i)$.
5. Zelda gives to EVERYONE the values $g^{r_1}, \ldots, g^{r_{t-1}}, g^s, g$.
   (We think discrete log is HARD so r_i not revealed.)

Recover: The usual – any group of t can blah blah.

Verify: A_i reveals $f(i) = 17$. Group computes: g^{17} and:

$(g^{r_{t-1}})^{i^{t-1}} \times (g^{r_{t-2}})^{i^{t-2}} \times \cdots \times (g^{r_1})^{i^1} \times (g^s)^{i^0}$

$= g^{r_{t-1}i^{t-1} + r_{t-2}i^{t-2} + \cdots + r_1 i^1 + s} = g^{f(i)}$

If this is g^{17} then A_i is truthful.

If not then A_i is dirty stinking liar.

1. **PRO:** If someone lies they know right away.
2. **PRO:** Serves as a deterrent.
3. **PRO:** t public strings, never need to update.
4. **CAVEAT:** Security – see next slide.

Security and References

The scheme above for VSS is by Paul Feldman.

A Practical Scheme for non-interactive Verifiable Secret Sharing
28th Conference on Foundations of Computer Science (FOCS)
1987

They give proof of security based on zero-knowledge protocols which are themselves based on blah blah.

Upshot: Pretty good Hardness Assumption.

Electronic Voting Using Public Key Crypto And Secret Sharing

Math Needed For Paillier Public Key Encryption

- $N = pq$ where p, q are primes.
- Let $m \in \mathbb{Z}_N$.
- Let $r \in \mathbb{Z}_N^*$ picked at random.
- Let $c = (1 + N)^m r^N \pmod{N^2}$. (NOTE mod N^2 not N)

1. Given c, p, q, determining m is EASY. (We omit proof but its not hard. In Katz's book.)
2. Given c, N, determining m is believed to be hard

The Paillier Public Key Encryption

n is a security parameter.

1. Alice picks p, q primes length n, let $N = pq$, broadcasts N.
2. To send $m \in \mathbb{Z}_N$ Bob picks random $r \in \mathbb{Z}_N^*$, broadcasts $(1+N)^m r^N \pmod{N^2}$
3. As noted in last slide, Alice can decode.
4. As noted in last slide, we think Eve cannot.

Hardness Assumption: The following is hard: given $a \in \mathbb{Z}_{N^2}$, is it an Nth power. (That this is equivalent to breaking the scheme is not obvious. Not hard – it is in Katz's book.)

Nice Property of Paillier Encryption

Alice broadcasts N to B_1, B_2.
B_1 broadcasts $c_1 = ENC(m_1) = (1+N)^{m_1} r_1^N$.
B_2 broadcasts $c_2 = ENC(m_2) = (1+N)^{m_2} r_2^N$.

Important Note:

$$c_1 c_2 = (1+N)^{m_1} r_1^N (1+N)^{m_2} r_2^N = (1+N)^{m_1+m_2} (r_1 r_2)^N$$

$$= ENC(m_1 + m_2)$$

Scenario: If B_1 broadcasts c_1, B_2 broadcasts c_2, and Alice doesn't see it, but does see $c_1 c_2$, then Alice can determine $m_1 + m_2$.

Nice Property of Paillier Encryption-II

Alice broadcasts N to $B_1, B_2, \ldots, B_S$.
B_1 broadcasts $c_1 = ENC(m_1)$.
B_2 broadcasts $c_2 = ENC(m_2)$.
$\vdots$
B_S broadcasts $c_S = ENC(m_S)$.

Important Note:

$$c_1 \cdots c_S = (1+N)^{m_1} r_1^N \cdots (1+N)^{m_S} r_S^N = (1+N)^{m_1+\cdots+m_S}(r_1 \cdots r_S)^N$$

$$= ENC(m_1 + \cdots + m_S)$$

Scenario: If B_1 broadcasts c_1, ..., B_S broadcasts c_S, and Alice doesn't see $c_1, \ldots, c_S$, but does see $c_1 \cdots c_S$, then Alice can determine $m_1 + \cdots + m_S$.

Application to Voting

A and B supervise voting. $B_1, \ldots, B_S$ vote NO (0) or YES (1).

1. Alice picks p, q primes length n, let $N = pq$, broadcasts N.
2. B_i votes $m_i \in \{0, 1\}$ and prepares c_i.
3. B_i send vote to Bob (NOT to Alice).
4. Bob computes $c = c_1 c_2 \cdots c_S$.
5. Bob gives c to Alice.
6. Alice can find $m_1 + \cdots + m_S$. If $< \frac{S}{2}$ then NO, otherwise YES.

Is there a problem with this? **Discuss**

Application to Voting

A and B supervise voting. $B_1, \ldots, B_S$ vote NO (0) or YES (1).

1. Alice picks p, q primes length n, let $N = pq$, broadcasts N.
2. B_i votes $m_i \in \{0, 1\}$ and prepares c_i.
3. B_i send vote to Bob (NOT to Alice).
4. Bob computes $c = c_1 c_2 \cdots c_S$.
5. Bob gives c to Alice.
6. Alice can find $m_1 + \cdots + m_S$. If $< \frac{S}{2}$ then NO, otherwise YES.

Is there a problem with this? **Discuss**

Problem: If $S > N^2$ then sum might overflow and go back to 0.

Solution: Make sure $N^2 > S$. Duh.

Security: Neither Alice nor Bob knows how anyone voted.

Application to Voting

A and B supervise voting. $B_1, \ldots, B_S$ vote NO (0) or YES (1).

1. Alice picks p, q primes length n, let $N = pq$, broadcasts N.
2. B_i votes $m_i \in \{0, 1\}$ and prepares c_i.
3. B_i send vote to Bob (NOT to Alice).
4. Bob computes $c = c_1 c_2 \cdots c_S$.
5. Bob gives c to Alice.
6. Alice can find $m_1 + \cdots + m_S$. If $< \frac{S}{2}$ then NO, otherwise YES.

Is there a problem with this? **Discuss**

Problem: If $S > N^2$ then sum might overflow and go back to 0.

Solution: Make sure $N^2 > S$. Duh.

Security: Neither Alice nor Bob knows how anyone voted.

Problem: Alice could lie to make **The All Night Party** win.

Application to Voting

A and B supervise voting. $B_1, \ldots, B_S$ vote NO (0) or YES (1).

1. Alice picks p, q primes length n, let $N = pq$, broadcasts N.
2. B_i votes $m_i \in \{0, 1\}$ and prepares c_i.
3. B_i send vote to Bob (NOT to Alice).
4. Bob computes $c = c_1 c_2 \cdots c_S$.
5. Bob gives c to Alice.
6. Alice can find $m_1 + \cdots + m_S$. If $< \frac{S}{2}$ then NO, otherwise YES.

Is there a problem with this? **Discuss**

Problem: If $S > N^2$ then sum might overflow and go back to 0.

Solution: Make sure $N^2 > S$. Duh.

Security: Neither Alice nor Bob knows how anyone voted.

Problem: Alice could lie to make **The All Night Party** win.

Problem: If Alice obtains c_i then she could find out how B_i voted.

Application to Voting

Alice and Bob joined by reps from each party $Q_1, \ldots, Q_t$.

1. Alice picks p, q primes length n, let $N = pq$, broadcasts N.
2. B_i votes $m_i \in \{0, 1\}$ and prepares c_i.
3. B_i sends vote to Bob (NOT Alice, $Q_1, \ldots, Q_t$).
4. Bob computes $c = c_1 c_2 \cdots c_S$ and broadcasts c.
5. Alice: VSS (t, t) – secret p, people $Q_1, \ldots, Q_t$.
6. $Q_1, \ldots, Q_t$ have p, q. They compute $DEC(c)$.
7. $Q_1, \ldots, Q_t$ agree on the winner.

Security: Neither Alice nor Bob knows how anyone voted.

Application to Voting

Alice and Bob joined by reps from each party $Q_1, \ldots, Q_t$.

1. Alice picks p, q primes length n, let $N = pq$, broadcasts N.
2. B_i votes $m_i \in \{0, 1\}$ and prepares c_i.
3. B_i sends vote to Bob (NOT Alice, $Q_1, \ldots, Q_t$).
4. Bob computes $c = c_1 c_2 \cdots c_S$ and broadcasts c.
5. Alice: VSS (t, t) – secret p, people $Q_1, \ldots, Q_t$.
6. $Q_1, \ldots, Q_t$ have p, q. They compute $DEC(c)$.
7. $Q_1, \ldots, Q_t$ agree on the winner.

Security: Neither Alice nor Bob knows how anyone voted.
Security: The outcome is correct since all $Q_1, \ldots, Q_t$ verify.

Application to Voting

Alice and Bob joined by reps from each party $Q_1, \ldots, Q_t$.

1. Alice picks p, q primes length n, let $N = pq$, broadcasts N.
2. B_i votes $m_i \in \{0, 1\}$ and prepares c_i.
3. B_i sends vote to Bob (NOT Alice, $Q_1, \ldots, Q_t$).
4. Bob computes $c = c_1 c_2 \cdots c_S$ and broadcasts c.
5. Alice: VSS (t, t) – secret p, people $Q_1, \ldots, Q_t$.
6. $Q_1, \ldots, Q_t$ have p, q. They compute $DEC(c)$.
7. $Q_1, \ldots, Q_t$ agree on the winner.

Security: Neither Alice nor Bob knows how anyone voted.
Security: The outcome is correct since all $Q_1, \ldots, Q_t$ verify.
Problem: If any Q_j obtains c_i then Q_j could find out how B_i voted.

Application to Voting

Alice and Bob joined by reps from each party $Q_1, \ldots, Q_t$.

1. Alice picks p, q primes length n, let $N = pq$, broadcasts N.
2. B_i votes $m_i \in \{0, 1\}$ and prepares c_i.
3. B_i sends vote to Bob (NOT Alice, $Q_1, \ldots, Q_t$).
4. Bob computes $c = c_1 c_2 \cdots c_S$ and broadcasts c.
5. Alice: VSS (t, t) – secret p, people $Q_1, \ldots, Q_t$.
6. $Q_1, \ldots, Q_t$ have p, q. They compute $DEC(c)$.
7. $Q_1, \ldots, Q_t$ agree on the winner.

Security: Neither Alice nor Bob knows how anyone voted.
Security: The outcome is correct since all $Q_1, \ldots, Q_t$ verify.
Problem: If any Q_j obtains c_i then Q_j could find out how B_i voted.
Problem: This can be solved. Omitted. In Katz's book.

For More on Secret Sharing

Google Scholar is a website of all papers (or at least most)
I went there and googled

"Secret Sharing"

How many papers are on it?
VOTE

1. between 1 and 100
2. between 100 and 1000
3. between 1000 and 10,000
4. between 10,000 and 20,000
5. over 20,000

For More on Secret Sharing

Google Scholar is a website of all papers (or at least most)
I went there and googled

"Secret Sharing"

How many papers are on it?
VOTE

1. between 1 and 100
2. between 100 and 1000
3. between 1000 and 10,000
4. between 10,000 and 20,000
5. over 20,000

58,000.