Disc 3b

Andy

UC Berkeley

July 3, 2018

Secret Sharing

1. Objective: Secret is unknown unless "enough" people come together

Secret Sharing

1. Objective: Secret is unknown unless "enough" people come together
2. Think nuclear launch codes

Secret Sharing

1. Objective: Secret is unknown unless "enough" people come together
2. Think nuclear launch codes
3. Polynomial of degree d is defined on $d + 1$ points (even in modspace)

Secret Sharing

1. Objective: Secret is unknown unless "enough" people come together
2. Think nuclear launch codes
3. Polynomial of degree d is defined on $d+1$ points (even in modspace)
4. Let $P(0)$ be the secret

Secret Sharing

1. Objective: Secret is unknown unless "enough" people come together
2. Think nuclear launch codes
3. Polynomial of degree d is defined on $d+1$ points (even in modspace)
4. Let $P(0)$ be the secret
5. Distribute $P(1), P(2), ..., P(k)$ st $k \geq d+1$

Secret Sharing

1. Objective: Secret is unknown unless "enough" people come together
2. Think nuclear launch codes
3. Polynomial of degree d is defined on $d+1$ points (even in modspace)
4. Let $P(0)$ be the secret
5. Distribute $P(1), P(2), ..., P(k)$ st $k \geq d+1$
6. $d+1$ people need to come together to solve for $P(0)$

Secret Sharing

1. Objective: Secret is unknown unless "enough" people come together
2. Think nuclear launch codes
3. Polynomial of degree d is defined on $d+1$ points (even in modspace)
4. Let $P(0)$ be the secret
5. Distribute $P(1), P(2), ..., P(k)$ st $k \geq d+1$
6. $d+1$ people need to come together to solve for $P(0)$
7. Kind of like the 2 man rule on nuclear submarines

Packet Loss

1. If original message is size n and we drop k packets, we need $n + k$ packets.

Packet Loss

1. If original message is size n and we drop k packets, we need $n + k$ packets.
2. Proof: pretend the n points code for points on a polynomial of degree $n - 1$

Packet Loss

1. If original message is size n and we drop k packets, we need $n + k$ packets.
2. Proof: pretend the n points code for points on a polynomial of degree $n - 1$
3. We just need n independent points!

Packet Loss

1. If original message is size n and we drop k packets, we need $n+k$ packets.
2. Proof: pretend the n points code for points on a polynomial of degree $n-1$
3. We just need n independent points!
4. Thus if we drop k from $n+k$, we are still good!

Berlekamp Welch

1. Invented by Lloyd Welch, and Elwyn Berlekamp

Berlekamp Welch

1. Invented by Lloyd Welch, and Elwyn Berlekamp
2. Berlekamp is a former UC Berkeley Professor (emeritus)

Berlekamp Welch

1. Invented by Lloyd Welch, and Elwyn Berlekamp
2. Berlekamp is a former UC Berkeley Professor (emeritus)
3. We need a way to recover from errors in messages

Berlekamp Welch

1. Invented by Lloyd Welch, and Elwyn Berlekamp
2. Berlekamp is a former UC Berkeley Professor (emeritus)
3. We need a way to recover from errors in messages
4. If original message is size n, and we have k errors. We need $n + 2k$ packets

Berlekamp Welch

1. $P(x)$ is true polynomial, we are trying to solve for this

Berlekamp Welch

1. $P(x)$ is true polynomial, we are trying to solve for this
2. $E(x)$ is error polynomial. $E(i) = 0$ iff ith packet was wrong

Berlekamp Welch

1. $P(x)$ is true polynomial, we are trying to solve for this
2. $E(x)$ is error polynomial. $E(i) = 0$ iff ith packet was wrong
3. r_i is received message (possibly with error)

Berlekamp Welch

1. $P(x)$ is true polynomial, we are trying to solve for this
2. $E(x)$ is error polynomial. $E(i) = 0$ iff ith packet was wrong
3. r_i is received message (possibly with error)
4. $P(i)E(i) = r_i * E(i)$

Berlekamp Welch

1. $P(x)$ is true polynomial, we are trying to solve for this
2. $E(x)$ is error polynomial. $E(i) = 0$ iff ith packet was wrong
3. r_i is received message (possibly with error)
4. $P(i)E(i) = r_i * E(i)$
5. $P(i)E(i)$ has $(n-1)+k$, or $n+k$ unknown coeffs.

Berlekamp Welch

1. $P(x)$ is true polynomial, we are trying to solve for this
2. $E(x)$ is error polynomial. $E(i) = 0$ iff ith packet was wrong
3. r_i is received message (possibly with error)
4. $P(i)E(i) = r_i * E(i)$
5. $P(i)E(i)$ has $(n-1)+k$, or $n+k$ unknown coeffs.
6. $E(i)$ has degree k. First coeff is 1, thus it has k unkown coeffs.

Berlekamp Welch

1. $P(x)$ is true polynomial, we are trying to solve for this
2. $E(x)$ is error polynomial. $E(i) = 0$ iff ith packet was wrong
3. r_i is received message (possibly with error)
4. $P(i)E(i) = r_i * E(i)$
5. $P(i)E(i)$ has $(n-1)+k$, or $n+k$ unknown coeffs.
6. $E(i)$ has degree k. First coeff is 1, thus it has k unkown coeffs.
7. $P(i)E(i) = r_i * E(i)$ has $n+2k$ unknowns that require $n+2k$ equations

Problem 3

1. Determine the form of $Q(x)E(x) = r_i * E(x)$

Problem 3

1. Determine the form of $Q(x)E(x) = r_i * E(x)$
2. ie $Q(0)E(0) = d = 3(0 + e) = r_0 * E(0)$

Problem 3

1. Determine the form of $Q(x)E(x) = r_i * E(x)$
2. ie $Q(0)E(0) = d = 3(0 + e) = r_0 * E(0)$
3. ie $Q(1)E(1) = a + b + c + d = 7(1 + e) = r_1 * E(0)$

Problem 3

1. Determine the form of $Q(x)E(x) = r_i * E(x)$
2. ie $Q(0)E(0) = d = 3(0 + e) = r_0 * E(0)$
3. ie $Q(1)E(1) = a + b + c + d = 7(1 + e) = r_1 * E(0)$
4. Then solve for a, b, c, d, e

Problem 3

1. Determine the form of $Q(x)E(x) = r_i * E(x)$
2. ie $Q(0)E(0) = d = 3(0 + e) = r_0 * E(0)$
3. ie $Q(1)E(1) = a + b + c + d = 7(1 + e) = r_1 * E(0)$
4. Then solve for a, b, c, d, e
5. Solve for $E(x)$

Problem 3

1. Determine the form of $Q(x)E(x) = r_i * E(x)$
2. ie $Q(0)E(0) = d = 3(0 + e) = r_0 * E(0)$
3. ie $Q(1)E(1) = a + b + c + d = 7(1 + e) = r_1 * E(0)$
4. Then solve for a, b, c, d, e
5. Solve for $E(x)$
6. Solve for $P(x) = \frac{Q(x)}{E(x)}$

Problem 3

1. Determine the form of $Q(x)E(x) = r_i * E(x)$
2. ie $Q(0)E(0) = d = 3(0 + e) = r_0 * E(0)$
3. ie $Q(1)E(1) = a + b + c + d = 7(1 + e) = r_1 * E(0)$
4. Then solve for a, b, c, d, e
5. Solve for $E(x)$
6. Solve for $P(x) = \frac{Q(x)}{E(x)}$
7. Get back the original message!!

Problem 3