CS171: Cryptography

Lecture 19

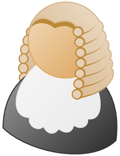
Sanjam Garg

Commitment Schemes

- Bind to a secret value that cannot be later explained with an alternate value.

x

Sender



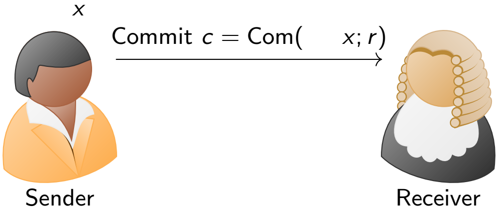
Receiver

Commitment Schemes

- Bind to a secret value that cannot be later explained with an alternate value.

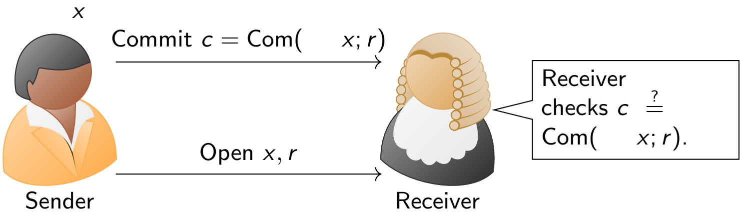
Commitment Schemes

- Bind to a secret value that cannot be later explained with an alternate value.



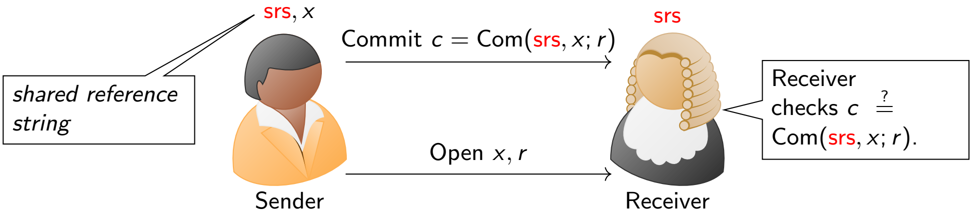
Commitment Schemes

- Bind to a secret value that cannot be later explained with an alternate value.



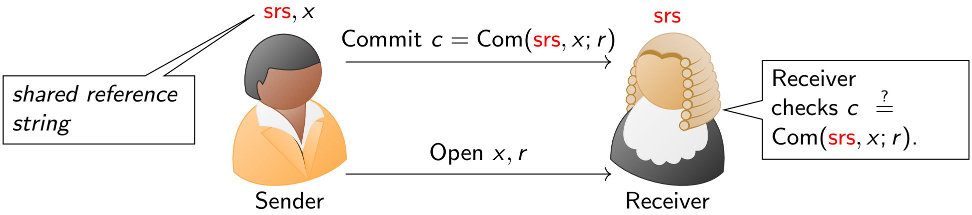
Commitment Schemes

- Bind to a secret value that cannot be later explained with an alternate value.



Commitment Schemes

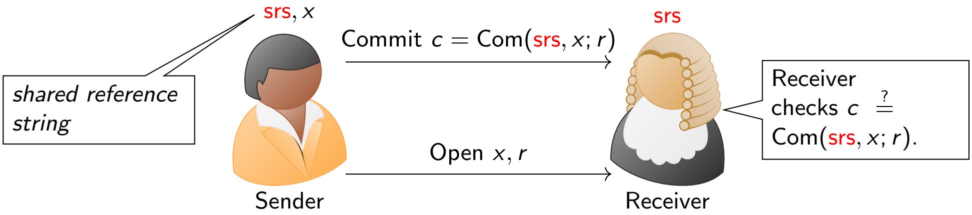
- Bind to a secret value that cannot be later explained with an alternate value.



- Correctness: An sender should be able to convince an honest receiver of the correct opening with *overwhelming* probability. (Easy to see)

Commitment Schemes

- Bind to a secret value that cannot be later explained with an alternate value.

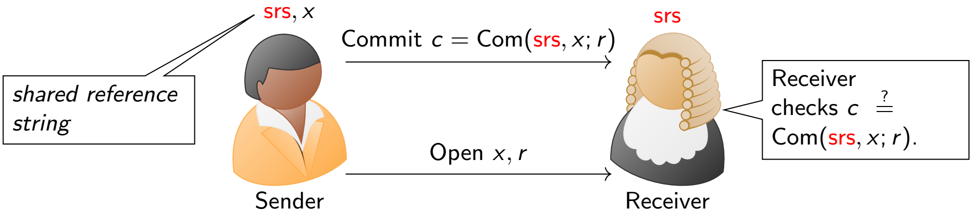


- Correctness: An sender should be able to convince an honest receiver of the correct opening with *overwhelming* probability. (Easy to see)
- Binding: No PPT cheating prover can find two openings for the same commitment. That is, $\forall$ PPT $\mathcal{A}$ we have that

$$\Pr[(x, r, x', r') \leftarrow \mathcal{A}(1^\lambda, \text{srs}) \text{ such that } \text{Com}(\text{srs}, x, r) = \text{Com}(\text{srs}, x', r')] = \text{neg}(\lambda)$$

Commitment Schemes

- Bind to a secret value that cannot be later explained with an alternate value.



- Correctness: An sender should be able to convince an honest receiver of the correct opening with *overwhelming* probability. (Easy to see)
- Binding: No PPT cheating prover can find two openings for the same commitment. That is, $\forall$ PPT $\mathcal{A}$ we have that

$$\Pr[(x, r, x', r') \leftarrow \mathcal{A}(1^\lambda, \text{srs}) \text{ such that } \text{Com}(\text{srs}, x, r) = \text{Com}(\text{srs}, x', r')] = \text{neg}(\lambda)$$

- Hiding: The commitemnt doesn't leak any information about the committed value x. That is, $\forall$ PPT $\mathcal{A}, x, x'$ we have that

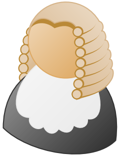
$$\left|\Pr[\mathcal{A}(1^\lambda, \text{srs}, \text{Com}(\text{srs}, x; r)) = 1] - \Pr[\mathcal{A}(1^\lambda, \text{srs}, \text{Com}(\text{srs}, x'; r')) = 1]\right| \leq \frac{1}{2} + \text{neg}(\lambda)$$

Commitment Scheme From Harness Concentration

$f : \{0,1\}^n \to \{0,1\}^n$ be a one-way permutation

$x \in \{0,1\}$

Sender



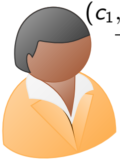
Receiver

Commitment Scheme From Harness Concentration

$f : \{0,1\}^n \to \{0,1\}^n$ be a one-way permutation

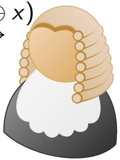
$x \in \{0,1\}$

$r, s \leftarrow \{0,1\}^n$



Sender

$\xrightarrow{(c_1, c_2, c_3) = (f(r), s, \langle r, s \rangle \oplus x)}$



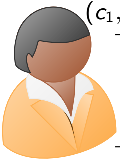
Receiver

Commitment Scheme From Harness Concentration

$f : \{0,1\}^n \to \{0,1\}^n$ be a one-way permutation

$x \in \{0,1\}$

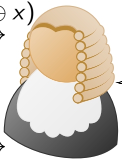
$r, s \leftarrow \{0,1\}^n$



Sender

$\xrightarrow{(c_1, c_2, c_3) = (f(r), s, \langle r, s \rangle \oplus x)}$

$\xrightarrow{\text{Open } x, r}$



Receiver

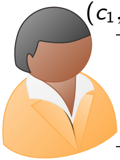
Receiver checks
$c_1 \stackrel{?}{=} f(r)$ and
$c_3 \stackrel{?}{=} \langle r, c_2 \rangle \oplus x$.

Commitment Scheme From Harness Concentration

$f : \{0,1\}^n \to \{0,1\}^n$ be a one-way permutation

$x \in \{0,1\}$

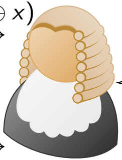
$r, s \leftarrow \{0,1\}^n$



Sender

$$\xrightarrow{(c_1, c_2, c_3) = (f(r), s, \langle r, s\rangle \oplus x)}$$

$$\xrightarrow{\text{Open } x, r}$$



Receiver

Receiver checks
$c_1 \stackrel{?}{=} f(r)$ and
$c_3 \stackrel{?}{=} \langle r, c_2\rangle \oplus x$.

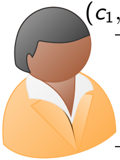
- Binding: Because f is a permutation, given c there is a unique value of r, x such that $c_1{=}f(r)$ and $c_3{=}\langle r, c_2\rangle \oplus x$.

Commitment Scheme From Harness Concentration

$f : \{0,1\}^n \to \{0,1\}^n$ be a one-way permutation

$x \in \{0,1\}$

$r, s \leftarrow \{0,1\}^n$



Sender

$(c_1, c_2, c_3) = (f(r), s, \langle r, s \rangle \oplus x)$ $\longrightarrow$

Open x, r $\longrightarrow$

Receiver

Receiver checks
$c_1 \stackrel{?}{=} f(r)$ and
$c_3 \stackrel{?}{=} \langle r, c_2 \rangle \oplus x$).

- Binding: Because f is a permutation, given c there is a unique value of r, x such that $c_1{=}f(r)$ and $c_3{=}\langle r, c_2 \rangle \oplus x$).
- Hiding: Follows from the hardness concentration property.

Can we use the encryption algorithm of any commitment scheme?

- Given $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ let sender execute $\mathsf{Com}(x; r)$ as follows. Use randomness r to execute Gen and then encrypt x using Enc and the obtained key k.

Can we use the encryption algorithm of any commitment scheme?

- Given $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ let sender execute $\mathsf{Com}(x; r)$ as follows. Use randomness r to execute Gen and then encrypt x using Enc and the obtained key k.
- No!
- While this commitment offers hinding, it doesn't give binding.

Can we use the encryption algorithm of any commitment scheme?

- Given $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ let sender execute $\mathsf{Com}(x; r)$ as follows. Use randomness r to execute Gen and then encrypt x using Enc and the obtained key k.
- No!
- While this commitment offers hinding, it doesn't give binding.
- Shouldn't binding come from the correctness of encryption?

Can we use the encryption algorithm of any commitment scheme?

- Given $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ let sender execute $\mathsf{Com}(x; r)$ as follows. Use randomness r to execute Gen and then encrypt x using Enc and the obtained key k.
- No!
- While this commitment offers hinding, it doesn't give binding.
- Shouldn't binding come from the correctness of encryption?
- The encrypter may not choose their random coins honestly.

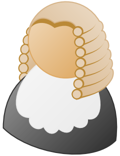
Pederson Commitment Schemes

$$\text{srs} = (G, g, q, h)$$

srs,x

Sender

srs



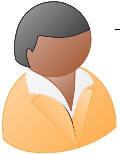
Receiver

Pederson Commitment Schemes

$\mathsf{srs} = (G, g, q, h)$

srs, x

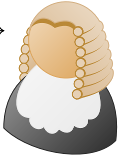
$r \leftarrow \mathbb{Z}_q$



Sender

Commit $c = g^x \cdot h^r$ $\longrightarrow$

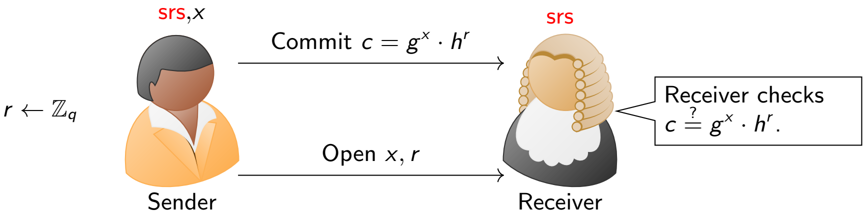
srs



Receiver

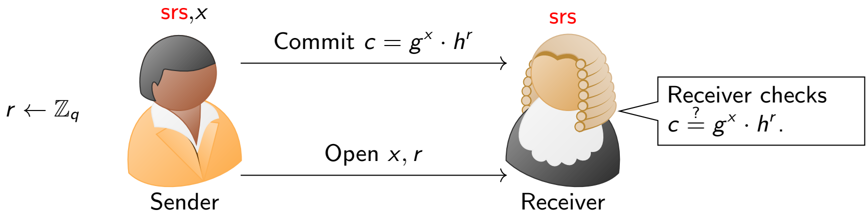
Pederson Commitment Schemes

$$\textcolor{red}{\mathsf{srs}} = (G, g, q, h)$$



Pederson Commitment Schemes

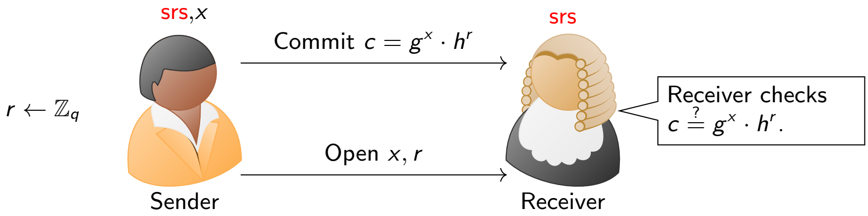
$\textsf{srs} = (G, g, q, h)$



- Binding: Given x, x', r, r' such that $g^x \cdot h^r = c = g^{x'} \cdot h^{r'}$ we can compute $dlog_g(h)$.

Pederson Commitment Schemes

$\text{srs} = (G, g, q, h)$



- Binding: Given x, x', r, r' such that $g^x \cdot h^r = c = g^{x'} \cdot h^{r'}$ we can compute $dlog_g(h)$.
- Hiding: For every $c = g^x h^r$ and x' there exists $r' = r + \frac{x' - x}{dlog_g(h)}$.

Commitment to a vector $\mathbf{x} = (\mathbf{x_0}, \ldots \mathbf{x_{n-1}})$

Commitment to a vector $\mathbf{x} = (\mathbf{x_0}, \ldots \mathbf{x_{n-1}})$

Send $c_i = \mathsf{Com}(x_i; r_i)$ for each i.

Commitment to a vector $\mathbf{x} = (\mathbf{x_0}, \ldots \mathbf{x_{n-1}})$

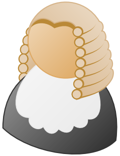
Send $c_i = \mathsf{Com}(x_i; r_i)$ for each i.

Can we do it succinctly?

Merkle Commitment Schemes

$x_0, \ldots x_{n-1}$

Sender

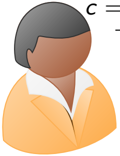


Receiver

Merkle Commitment Schemes

$x_0, \dots x_{n-1}$

$r_0 \dots r_{n-1} \leftarrow \mathbb{Z}_q$



Sender

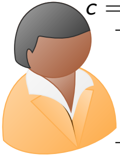
$$\xrightarrow{c \,=\, H\left(g^{x_0} \cdot h^{r_0} \dots H(g^{x_{n-1}} \cdot h^{r_{n-1}})\right)}$$

Receiver

Merkle Commitment Schemes

$x_0, \dots x_{n-1}$

$r_0 \dots r_{n-1} \leftarrow \mathbb{Z}_q$



Sender

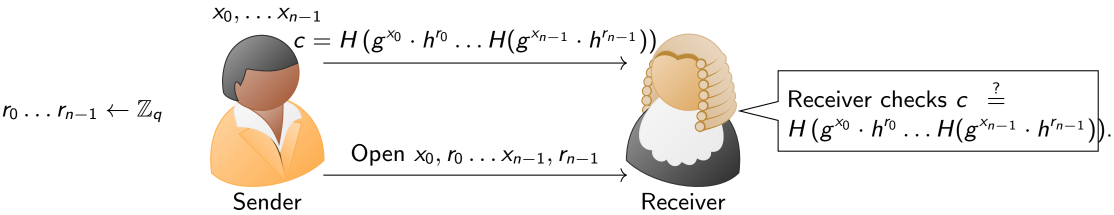
$$\xrightarrow{c = H\left(g^{x_0} \cdot h^{r_0} \dots H(g^{x_{n-1}} \cdot h^{r_{n-1}})\right)}$$

$$\xrightarrow{\text{Open } x_0, r_0 \dots x_{n-1}, r_{n-1}}$$

Receiver

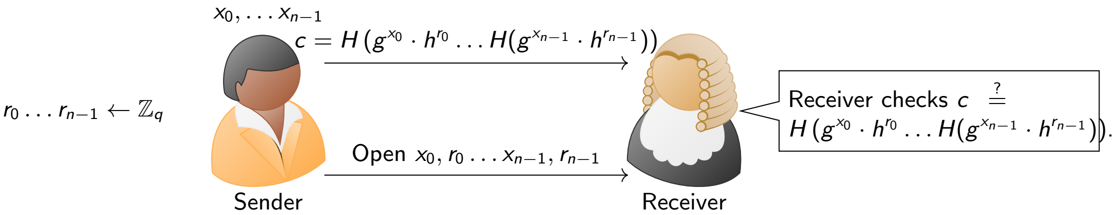
Receiver checks $c \stackrel{?}{=} H\left(g^{x_0} \cdot h^{r_0} \dots H(g^{x_{n-1}} \cdot h^{r_{n-1}})\right)$.

Merkle Commitment Schemes



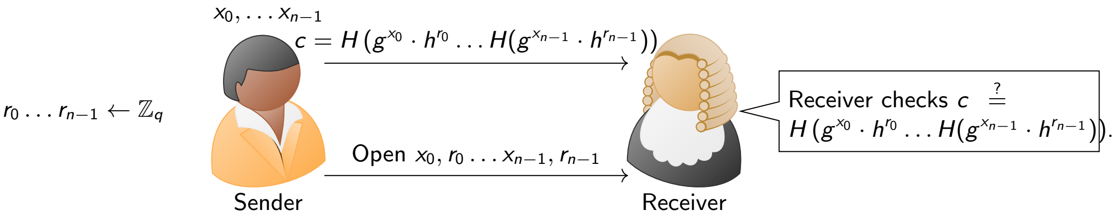
- **Hashing in More Detail ($n = 2^\ell$):** For every $i \in \{0, n-1\}$, $c_i^0 = g^{x_i} h^{r_i}$. For all $j \in \{0, \ldots \ell - 1\}, i \in \{0 \ldots 2^j - 1\}$ set $c_{i/2}^{j+1} = H(c_i^j || c_{i+1}^j)$. Finally, $c = c_0^\ell$.

Merkle Commitment Schemes



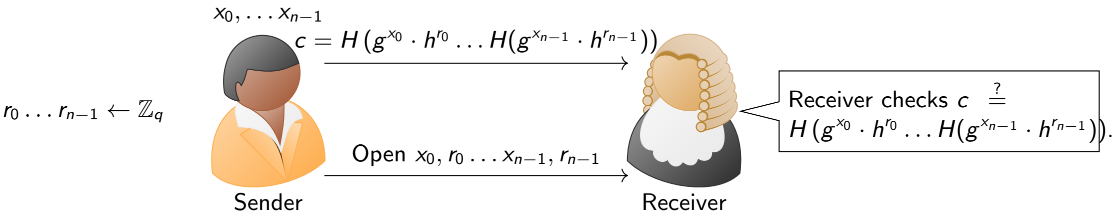
- Hashing in More Detail ($n = 2^\ell$): For every $i \in \{0, n-1\}$, $c_i^0 = g^{x_i} h^{r_i}$. For all $j \in \{0, \ldots \ell - 1\}, i \in \{0 \ldots 2^j - 1\}$ set $c_{i/2}^{j+1} = H(c_i^j || c_{i+1}^j)$. Finally, $c = c_0^\ell$.
- Binding: An attacker that outputs distinct $x_1, r_0, \ldots x_{n-1}, r_{n-1}$ and $x_1', r_1', \ldots x_n', r_n'$ such that the receiver check pass on both either (i) break CRHF, or (ii) can comute $dlog_g(h)$.

Merkle Commitment Schemes



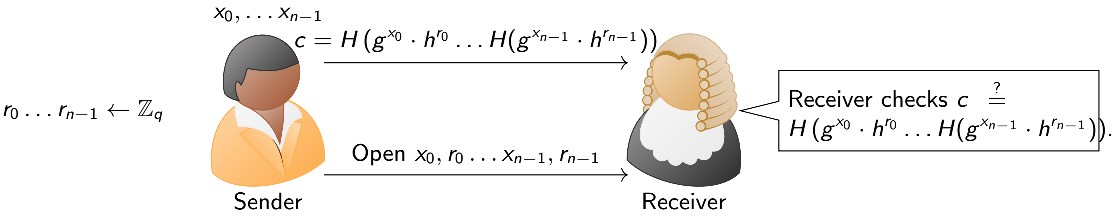
- Hashing in More Detail ($n = 2^\ell$): For every $i \in \{0, n-1\}$, $c_i^0 = g^{x_i} h^{r_i}$. For all $j \in \{0, \ldots \ell - 1\}, i \in \{0 \ldots 2^j - 1\}$ set $c_{i/2}^{j+1} = H(c_i^j || c_{i+1}^j)$. Finally, $c = c_0^\ell$.
- Binding: An attacker that outputs distinct $x_1, r_0, \ldots x_{n-1}, r_{n-1}$ and $x_1', r_1', \ldots x_n', r_n'$ such that the receiver check pass on both either (i) break CRHF, or (ii) can comute $dlog_g(h)$.
- Hiding: For every $c_i^0 = g^{x_i} h^{r_i}$ that is hashed and x_i' there exists $r_i' = r_i + \frac{x_i' - x_i}{dlog_g(h)}$.

Merkle Commitment Schemes



- Hashing in More Detail ($n = 2^\ell$): For every $i \in \{0, n-1\}$, $c_i^0 = g^{x_i} h^{r_i}$. For all $j \in \{0, \dots \ell - 1\}, i \in \{0 \dots 2^j - 1\}$ set $c_{i/2}^{j+1} = H(c_i^j || c_{i+1}^j)$. Finally, $c = c_0^\ell$.
- Binding: An attacker that outputs distinct $x_1, r_0, \dots x_{n-1}, r_{n-1}$ and $x_1', r_1', \dots x_n', r_n'$ such that the receiver check pass on both either (i) break CRHF, or (ii) can comute $dlog_g(h)$.
- Hiding: For every $c_i^0 = g^{x_i} h^{r_i}$ that is hashed and x_i' there exists $r_i' = r_i + \frac{x_i' - x_i}{dlog_g(h)}$.
- Partial Opening (Location k): Opening c_k^0, x_k, r_k and

Merkle Commitment Schemes

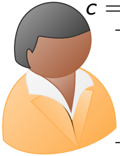


- Hashing in More Detail ($n = 2^\ell$): For every $i \in \{0, n-1\}$, $c_i^0 = g^{x_i} h^{r_i}$. For all $j \in \{0, \dots \ell - 1\}, i \in \{0 \dots 2^j - 1\}$ set $c_{i/2}^{j+1} = H(c_i^j || c_{i+1}^j)$. Finally, $c = c_0^\ell$.
- Binding: An attacker that outputs distinct $x_1, r_0, \dots x_{n-1}, r_{n-1}$ and $x_1', r_1', \dots x_n', r_n'$ such that the receiver check pass on both either (i) break CRHF, or (ii) can comute $dlog_g(h)$.
- Hiding: For every $c_i^0 = g^{x_i} h^{r_i}$ that is hashed and x_i' there exists $r_i' = r_i + \frac{x_i' - x_i}{dlog_g(h)}$.
- Partial Opening (Location k): Opening c_k^0, x_k, r_k and

Merkle Commitment Schemes

$x_0, \dots x_{n-1}$

$r_0 \dots r_{n-1} \leftarrow \mathbb{Z}_q$



Sender

$\xrightarrow{c = H\left(g^{x_0} \cdot h^{r_0} \dots H(g^{x_{n-1}} \cdot h^{r_{n-1}})\right)}$

$\xrightarrow{\text{Open } x_0, r_0 \dots x_{n-1}, r_{n-1}}$

Receiver

Receiver checks $c \stackrel{?}{=} H\left(g^{x_0} \cdot h^{r_0} \dots H(g^{x_{n-1}} \cdot h^{r_{n-1}})\right)$.

- Hashing in More Detail ($n = 2^\ell$): For every $i \in \{0, n-1\}$, $c_i^0 = g^{x_i} h^{r_i}$. For all $j \in \{0, \dots \ell - 1\}, i \in \{0 \dots 2^j - 1\}$ set $c_{i/2}^{j+1} = H(c_i^j || c_{i+1}^j)$. Finally, $c = c_0^\ell$.
- Binding: An attacker that outputs distinct $x_1, r_0, \dots x_{n-1}, r_{n-1}$ and $x_1', r_1', \dots x_n', r_n'$ such that the receiver check pass on both either (i) break CRHF, or (ii) can comute $dlog_g(h)$.
- Hiding: For every $c_i^0 = g^{x_i} h^{r_i}$ that is hashed and x_i' there exists $r_i' = r_i + \frac{x_i' - x_i}{dlog_g(h)}$.
- Partial Opening (Location k): Opening c_k^0, x_k, r_k and $\forall j \in \{0, \ell\}$ send $c^j_{\frac{k}{2^j}}$ and $c^j_{\frac{k}{2^j}+1}$.

Commitment to a Polynomial $f(x)$ of degree $n-1$

Succinctly

Polynomial Interpolation

Problem: Given $a_0 ... a_{n-1}$ **(evaluation representation) find** **<u>the</u>** **degree-**$n-1$ **polynomial** $f(x) = b_0 + b_1 x + ... b_{n-1} x^{n-1}$ **(coefficient representation), i.e.** $b_0, b_1 ... b_{n-1}$**, such that for all** $i \in H = \{0, ... n-1\}$ **we have** $f(i) = a_i$**.**

Polynomial Interpolation

Problem: Given $a_0 \ldots a_{n-1}$ **(evaluation representation) find <u>the</u> degree-**$n-1$ **polynomial** $f(x) = b_0 + b_1 x + \ldots b_{n-1} x^{n-1}$ **(coefficient representation), i.e.** $b_0, b_1 \ldots b_{n-1}$**, such that for all** $i \in H = \{0, \ldots n-1\}$ **we have** $f(i) = a_i$**.**

- Let $L_i(x)$ be the degree-$n-1$ polynomial such that $L_i(i) = 1$ and for all $j \in H \backslash \{i\}$ $L_i(j) = 0$

$$L_i(x) = \frac{\prod_{j \in H \backslash \{i\}} (x - j)}{\prod_{j \in H \backslash \{i\}} (i - j)}.$$

Polynomial Interpolation

Problem: Given $a_0 \dots a_{n-1}$ **(evaluation representation) find the degree-**$n-1$ **polynomial** $f(x) = b_0 + b_1 x + \dots b_{n-1} x^{n-1}$ **(coefficient representation), i.e.** $b_0, b_1 \dots b_{n-1}$**, such that for all** $i \in H = \{0, \dots n-1\}$ **we have** $f(i) = a_i$**.**

- Let $L_i(x)$ be the degree-$n-1$ polynomial such that $L_i(i) = 1$ and for all $j \in H \backslash \{i\}$ $L_i(j) = 0$

$$L_i(x) = \frac{\prod_{j \in H \backslash \{i\}} (x - j)}{\prod_{j \in H \backslash \{i\}} (i - j)}.$$

- Next, we have

$$f(x) = \sum_{i \in H} a_i \cdot L_i(x)$$

Polynomial Interpolation

Problem: Given $a_0 \ldots a_{n-1}$ (evaluation representation) find <u>the</u> degree-$n-1$ polynomial $f(x) = b_0 + b_1 x + \ldots b_{n-1} x^{n-1}$ (coefficient representation), i.e. $b_0, b_1 \ldots b_{n-1}$, such that for all $i \in H = \{0, \ldots n-1\}$ we have $f(i) = a_i$.

- Let $L_i(x)$ be the degree-$n-1$ polynomial such that $L_i(i) = 1$ and for all $j \in H \backslash \{i\}$ $L_i(j) = 0$

$$L_i(x) = \frac{\prod_{j \in H \backslash \{i\}} (x-j)}{\prod_{j \in H \backslash \{i\}} (i-j)}.$$

- Next, we have

$$f(x) = \sum_{i \in H} a_i \cdot L_i(x)$$

- L_is can be cached for efficiency. DIY: Prove that the constructed polynomials are correct and unique.

Polynomial Commitment/Pairing Curve BLS12-381

- Gives groups $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$ and G_T (of the same prime order p) along with a bilinear pairing operation e.
- For every $\alpha, \beta \in \mathbb{Z}_p^*$, we have that $e(g_1^\alpha, g_2^\beta) = e(g_1, g_2)^{\alpha\beta}$.

Polynomial Commitment/Pairing Curve BLS12-381

- Gives groups $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$ and G_T (of the same prime order p) along with a bilinear pairing operation e.
- For every $\alpha, \beta \in \mathbb{Z}_p^*$, we have that $e(g_1^\alpha, g_2^\beta) = e(g_1, g_2)^{\alpha\beta}$.
- **Setup:** srs generation that supports committing to degree $d-1$ polynomials:
  - Sample $\tau \leftarrow \mathbb{Z}_p^*$.
  - $\text{srs} = (h_0 = g_1, h_1 = g_1^\tau, g_1^{\tau^2}, \ldots . h_d = g_1^{\tau^{d-1}}, g_2, h' = g_2^\tau)$

Polynomial Commitment/Pairing Curve BLS12-381

- Gives groups $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$ and G_T (of the same prime order p) along with a bilinear pairing operation e.
- For every $\alpha, \beta \in \mathbb{Z}_p^*$, we have that $e(g_1^\alpha, g_2^\beta) = e(g_1, g_2)^{\alpha\beta}$.
- **Setup:** srs generation that supports committing to degree $d-1$ polynomials:
  - Sample $\tau \leftarrow \mathbb{Z}_p^*$.
  - $\mathsf{srs} = (h_0 = g_1, h_1 = g_1^\tau, g_1^{\tau^2}, \ldots h_d = g_1^{\tau^{d-1}}, g_2, h' = g_2^\tau)$
- **Homomorphic Commitment:** Given srs and a polynomial $f(x) = c_0 + c_1 x + \ldots c_{d-1} x^{d-1}$ of degree $d-1$, we can compute $\mathsf{Com}(f)$ as:

$$F = \mathsf{Com}(f) = g_1^{f(\tau)} = \prod_{i=0}^{d-1} h_i^{c_i}$$

Polynomial Commitment/Pairing Curve BLS12-381

- Gives groups $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$ and G_T (of the same prime order p) along with a bilinear pairing operation e.
- For every $\alpha, \beta \in \mathbb{Z}_p^*$, we have that $e(g_1^\alpha, g_2^\beta) = e(g_1, g_2)^{\alpha\beta}$.
- **Setup:** srs generation that supports committing to degree $d-1$ polynomials:
  - Sample $\tau \leftarrow \mathbb{Z}_p^*$.
  - srs $= (h_0 = g_1, h_1 = g_1^\tau, g_1^{\tau^2}, \ldots . h_d = g_1^{\tau^{d-1}}, g_2, h' = g_2^\tau)$
- **Homomorphic Commitment:** Given srs and a polynomial $f(x) = c_0 + c_1 x + \ldots c_{d-1} x^{d-1}$ of degree $d-1$, we can compute $\mathrm{Com}(f)$ as:

$$F = \mathrm{Com}(f) = g_1^{f(\tau)} = \prod_{i=0}^{d-1} h_i^{c_i}$$

- **Opening:** Show that $f(z) = s$. In this case, $g(x) = f(x) - s$ is such that $g(z) = 0$. Or, $x - z$ divides $f(x) - s$.

Polynomial Commitment/Pairing Curve BLS12-381

- Gives groups $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$ and G_T (of the same prime order p) along with a bilinear pairing operation e.
- For every $\alpha, \beta \in \mathbb{Z}_p^*$, we have that $e(g_1^\alpha, g_2^\beta) = e(g_1, g_2)^{\alpha\beta}$.
- **Setup:** srs generation that supports committing to degree $d-1$ polynomials:
  - Sample $\tau \leftarrow \mathbb{Z}_p^*$.
  - srs $= (h_0 = g_1, h_1 = g_1^\tau, g_1^{\tau^2}, \ldots h_d = g_1^{\tau^{d-1}}, g_2, h' = g_2^\tau)$
- **Homomorphic Commitment:** Given srs and a polynomial $f(x) = c_0 + c_1 x + \ldots c_{d-1} x^{d-1}$ of degree $d-1$, we can compute $\mathrm{Com}(f)$ as:

$$F = \mathrm{Com}(f) = g_1^{f(\tau)} = \prod_{i=0}^{d-1} h_i^{c_i}$$

- **Opening:** Show that $f(z) = s$. In this case, $g(x) = f(x) - s$ is such that $g(z) = 0$. Or, $x - z$ divides $f(x) - s$.
- Sender computes $T(x) = \frac{f(x)-f(z)}{x-z}$ and sends $W = \mathrm{Com}(T)$.

Polynomial Commitment/Pairing Curve BLS12-381

- Gives groups $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$ and G_T (of the same prime order p) along with a bilinear pairing operation e.
- For every $\alpha, \beta \in \mathbb{Z}_p^*$, we have that $e(g_1^\alpha, g_2^\beta) = e(g_1, g_2)^{\alpha\beta}$.
- **Setup:** srs generation that supports committing to degree $d-1$ polynomials:
  - Sample $\tau \leftarrow \mathbb{Z}_p^*$.
  - srs $= (h_0 = g_1, h_1 = g_1^\tau, g_1^{\tau^2}, \ldots. h_d = g_1^{\tau^{d-1}}, g_2, h' = g_2^\tau)$
- **Homomorphic Commitment:** Given srs and a polynomial $f(x) = c_0 + c_1 x + \ldots c_{d-1} x^{d-1}$ of degree $d-1$, we can compute $\mathrm{Com}(f)$ as:

$$F = \mathrm{Com}(f) = g_1^{f(\tau)} = \prod_{i=0}^{d-1} h_i^{c_i}$$

- **Opening:** Show that $f(z) = s$. In this case, $g(x) = f(x) - s$ is such that $g(z) = 0$. Or, $x - z$ divides $f(x) - s$.
- Sender computes $T(x) = \frac{f(x)-f(z)}{x-z}$ and sends $W = \mathrm{Com}(T)$.
- **Receiver Accepts if:** $e\left(\frac{F}{g_1^s}, g_2\right) = e\left(W, \frac{h'}{g_2^z}\right)$.

Optimizing Opening by Batching — Warmup

Often we want to check multiple pairing equations:

$$e(F_0, g_2) = e(W_0, h_2)$$

$$e(F_1, g_2) = e(W_1, h_2)$$

$$e(F_2, g_2) = e(W_2, h_2)$$

A faster way to check?

Optimizing Opening by Batching — Warmup

Often we want to check multiple pairing equations:

$$e(F_0, g_2) = e(W_0, h_2)$$

$$e(F_1, g_2) = e(W_1, h_2)$$

$$e(F_2, g_2) = e(W_2, h_2)$$

A faster way to check? The receiver samples a random γ and checks:

$$e\left(\prod_{i=0}^{2} F_i^{\gamma^i}, g_2\right) = e\left(\prod_{i=0}^{2} W_i^{\gamma^i}, h_2\right)$$

Need only 2 pairings instead of 6.

Optimizing Opening by Batching

- **Problem:** Consider the setting where sender commits to polynomials $f_1 ... f_t$ as $F_1 ... F_t$ and wants to show that for all i we have that $f_i(z) = s_i$.

Optimizing Opening by Batching

- **Problem:** Consider the setting where sender commits to polynomials $f_1 \dots f_t$ as $F_1 \dots F_t$ and wants to show that for all i we have that $f_i(z) = s_i$.
- **Opening:** Receiver sends random $\gamma \in \mathbb{F}$. Sender computes $T(x) = \sum_{i=1}^{t} \gamma^{i-1} \cdot \frac{f_i(x) - f_i(z)}{x - z}$ and sends $W = \mathrm{Com}(T)$.

Optimizing Opening by Batching

- **Problem:** Consider the setting where sender commits to polynomials $f_1 \ldots f_t$ as $F_1 \ldots F_t$ and wants to show that for all i we have that $f_i(z) = s_i$.
- **Opening:** Receiver sends random $\gamma \in \mathbb{F}$. Sender computes $T(x) = \sum_{i=1}^{t} \gamma^{i-1} \cdot \frac{f_i(x) - f_i(z)}{x - z}$ and sends $W = \mathrm{Com}(T)$.
- **Receiver Accepts if:** $e\left(\prod_{i=1}^{t} \left(\frac{F_i}{g_1^{s_i}}\right)^{\gamma^{i-1}}, g_2\right) = e\left(W, \frac{h'}{g_2^{z}}\right)$. (only two pairings)

KZG Commitment is Homomorphic

- Given commitments c_1, c_2 to polynomials $f_1(x)$ and $f_2(x)$ find a commitment to the polynomial $g(x) = f_1(x) + f_2(x)$?

KZG Commitment is Homomorphic

- Given commitments c_1, c_2 to polynomials $f_1(x)$ and $f_2(x)$ find a commitment to the polynomial $g(x) = f_1(x) + f_2(x)$?

KZG Commitment is Homomorphic

- Given commitments c_1, c_2 to polynomials $f_1(x)$ and $f_2(x)$ find a commitment to the polynomial $g(x) = f_1(x) + f_2(x)$?
- Output Commitment as $c_1 \cdot c_2$.