Wikström's Commitment-Consistent Proof of a Shuffle

Rolf Haenni

http://e-voting.bfh.ch

Seminar, E-Voting Group, BFH

September 5th, 2012

Outline

Introduction

Review of Cryptographic Primitives

Batch Re-Encryption and Exponentiation Proofs

Proof of Knowledge of Permutation Matrix

Conclusion

Outline

Introduction

Review of Cryptographic Primitives

Batch Re-Encryption and Exponentiation Proofs

Proof of Knowledge of Permutation Matrix

Conclusion

Motivation

Proof of Re-Encryption Shuffle: given

1. Public key pk
2. Input encryptions $u_1, \ldots, u_n$
3. Output encryptions $u'_1, \ldots, u'_n$

prove knowledge of

1. Permutation π
2. Randomizations $r_1, \ldots, r_n$

such that $u'_i = u_{\pi(i)} \cdot E_{pk}(1, r_{\pi(i)})$

Motivation

Proof of Exponentiation Shuffle: given

1. Input values $u_1, \ldots, u_n$
2. Output values $u'_1, \ldots, u'_n$
3. Commitment $c = C(\alpha, s)$

prove knowledge of

1. Permutation π
2. Exponent α, randomization s

such that $c = C(\alpha, s)$ and $u'_i = u^{\alpha}_{\pi(i)}$

General Proof Strategy

The prover

1. Commits to a permutation matrix of π
2. Proves that this commitment contains a permutation matrix
3. Proves that this permutation has been used in the shuffle

References

D. Wikström.
A Commitment-Consistent Proof of a Shuffle.
ACISP'09, 14th Australasian Conference on Information Security and Privacy, Brisbane, Australia, 2009.

B. Terelius and D. Wikström.
Proofs of Restricted Shuffles.
AFRICACRYPT'10, 3rd International Conference on Cryptology in Africa, Stellenbosch, South Africa, 2010.

D. Wikström.
A sender verifiable mix-net and a new proof of a shuffle.
ASIACRYPT'05, 11th International Conference on the Theory and Application of Cryptographic Techniques, Chennai, India, 2005.

Related Work

J. Furukawa and K. Sako.
An efficient scheme for proving a shuffle.
CRYPTO'01, 21st Annual International Cryptology Conference on Advances in Cryptology, Santa Barbara, USA, 2001

C. A. Neff.
A verifiable secret shuffle and its application to e-voting.
CCS'01, 8th ACM Conference on Computer and Communications Security, Philadelphia, USA, 2001.

J. Groth.
A verifiable secret shuffle of homomorphic encryptions.
Journal of Cryptology, 23(4):546–579, 2010.

Outline

Introduction

Review of Cryptographic Primitives

Batch Re-Encryption and Exponentiation Proofs

Proof of Knowledge of Permutation Matrix

Conclusion

Pedersen Commitment

- Let g, h be independently chosen generators of G_q.
- Let $m \in \mathbb{Z}_q$, then

$$C(m, s) = g^s \cdot h^m$$

  is a Pedersen commitment of m for $s \in_R \mathbb{Z}_q$ is chosen uniformly at random
- Perfectly hiding, computationally binding
- Homomorphic
  - → $C(m_1, s_1) \cdot C(m_2, s_2) = C(m_1 + m_2, s_1 + s_2)$
  - → $C(m, s)^e = C(e \cdot m, e \cdot s)$

Generalized Pedersen Commitment

- Let $g, h_1, \ldots, h_n$ be independently chosen generators of G_q
- Let $\overline{m} = (m_1, \ldots, m_n) \in \mathbb{Z}_q^n$, then

  $$C(\overline{m}, s) = g^s \cdot h_1^{m_1} \cdots h_n^{m_n}$$

  is a generalized Pedersen commitment of $\overline{m}$, where $s \in_R \mathbb{Z}_q$ is chosen uniformly at random
- Perfectly hiding, computationally binding
- Homomorphic
  - → $C(\overline{m}_1, s_1) \cdot C(\overline{m}_2, s_2) = C(\overline{m}_1 + \overline{m}_2, s_1 + s_2)$
  - → $C(\overline{m}, s)^e = C(e \cdot \overline{m}, e \cdot s)$

Non-Interactive Basic Preimage Proof

- Let $(X, +, 0)$ and $(Y, \cdot, 1)$ be groups of finite order
- Consider a one-way group homomorphism $\phi : X \to Y$
- Let $b = \phi(a)$ be publicly known
- The prover P proves knowledge of a using the Σ-protocol:
  1. Choose $\omega \in_R X$ uniformly at random
  2. Compute $t = \phi(\omega)$
  3. Compute $c = H(b, t) \bmod q$, for $q = 2^L \leq |\text{image}(\phi)|$
  4. Compute $s = \omega + c \cdot a$
  5. Publish $\pi = (t, s)$
- To verify π, the verifier V computes $c = H(b, t) \bmod q$ and checks $\phi(s) \stackrel{?}{=} t \cdot b^c$

Example 1: Discrete Logarithm (Schnorr)

- Let g be a generator of G_q
- Let $c = g^m$ be a publicly known commitment of $m \in \mathbb{Z}_q$
- P proves knowledge of m using the Σ-protocol for:

$$a = m,$$
$$b = c,$$
$$\phi(x) = g^x,$$

where $\phi : \underbrace{\mathbb{Z}_q}_{X} \rightarrow \underbrace{G_q}_{Y}$

Example 2: Equality of Discrete Logarithms

- Let g_1 and g_2 be generators of G_q
- Let $c_1 = g_1^m$ and $c_2 = g_2^m$ be public commitments of $m \in \mathbb{Z}_q$
- P proves knowledge of m using the Σ-protocol for:

$$a = m,$$
$$b = (c_1, c_2),$$
$$\phi(x) = (g_1^x, g_2^x),$$

where $\phi : \underbrace{\mathbb{Z}_q}_{X} \rightarrow \underbrace{G_q \times G_q}_{Y}$

- Note that $t = (t_1, t_2)$

Example 3: Pedersen Commitment Proof

- Let $c = C(m, s)$ be a publicly known commitment of $m \in \mathbb{Z}_q$
- P proves knowledge of m and s using the Σ-protocol for:

$$\begin{aligned} a &= (m, s), \\ b &= c, \\ \phi(x_1, x_2) &= C(x_1, x_2) = g^{x_2} h^{x_1}, \end{aligned}$$

where $\phi : \underbrace{\mathbb{Z}_q \times \mathbb{Z}_q}_{X} \rightarrow \underbrace{G_q}_{Y}$

- Note that $\omega = (\omega_1, \omega_2)$ and $s = (s_1, s_2)$

Example 4: Commitment Multiplication Proof

- Let $c_1 = C(m_1, s_1)$, $c_2 = C(m_2, s_2)$, and $c_3 = C(m_3, s_3)$ be publicly known commitments of $m_1, m_2, m_3 \in \mathbb{Z}_q$
- P proves knowledge of m_1, m_2, and $m_3 = m_1 m_2$ using the Σ-protocol for:

$$
\begin{aligned}
a &= (m_1, s_1, m_2, s_2, s_3 - m_1 s_2) \\
b &= (c_1, c_2, c_3), \\
\phi(x_1, x_2, x_3, x_4, x_5) &= (C(x_1, x_2), C(x_3, x_4), g^{x_5} c_2^{x_1})
\end{aligned}
$$

where $\phi : \underbrace{\mathbb{Z}_q^5}_{X} \rightarrow \underbrace{G_q^3}_{Y}$

- Note that $\omega = (\omega_1, \ldots, \omega_5)$, $t = (t_1, \ldots, t_3)$, $s = (s_1, \ldots, s_5)$

Composition of Preimage Proofs

- Consider n one-way group homomorphism $\phi_i : X_i \to Y_i$
- Let $b_1, \ldots, b_n$ be publicly known, where $b_i = \phi_i(a_i)$
- P proves knowledge of $a_1, \ldots, a_n$ using the Σ-protocol for:

$$\begin{aligned} a &= (a_1, \ldots, a_n), \\ b &= (b_1, \ldots, b_n), \\ \phi(x_1, \ldots, x_n) &= (\phi_1(x_1), \ldots, \phi_n(x_n)), \end{aligned}$$

where $\phi : \underbrace{X_1 \times \cdots \times X_n}_{X} \to \underbrace{Y_1 \times \cdots \times Y_n}_{Y}$

- Note that $\omega = (\omega_1, \ldots, \omega_n)$, $t = (t_1, \ldots, t_n)$, $s = (s_1, \ldots, s_n)$, which implies large proofs of size $O(n)$

Batch Preimage Proof

- Consider a single one-way group homomorphisms $\phi : X \to Y$
- Let $b_1, \dots, b_m$ be publicly known, where $b_i = \phi(a_i)$
- P proves knowledge of $a_1, \dots, a_n$ as follows:
  - → V chooses random seed z
  - → P computes $(e_1, \dots, e_n) = PRG(z)$
  - → P computes $b = \prod_i b_i^{e_i}$ using the fast algorithm from BGR98

$$b = \prod_i b_i^{e_i} = \prod_i \phi(a_i)^{e_i} = \prod_i \phi(e_i a_i) = \phi(\sum_i e_i a_i)$$

  - → P computes basic preimage proof for $b = \phi(a)$ and $a = \sum_i e_i a_i$
- Implies small proofs of size $O(1)$
- Important: verification requires testing $b_1, \dots, b_m \in Y$

Non-Interactive Batch Preimage Proof

- Consider a single one-way group homomorphisms $\phi : X \to Y$
- Let $b_1, \ldots, b_m$ be publicly known, where $b_i = \phi(a_i)$
- P proves knowledge of $a_1, \ldots, a_n$ as follows:
  1. Choose $\omega \in_R X$ uniformly at random
  2. Compute $t = \phi(\omega)$
  3. Compute $e_i = H(b_i, t) \bmod q$, for $q = 2^L \leq |\text{image}(\phi)|$
  4. Compute $a = \sum_i e_i a_i$ and $b = \prod_i b_i^{e_i}$
  5. Compute $c = H(b, t) \bmod q$
  6. Compute $s = \omega + c \cdot a$
  7. Publish $\pi = (t, s)$
- To verify π, V computes $e_i = H(b_i, t)$, $b = \prod_i b_i^{e_i} \bmod q$, and $c = H(b, t) \bmod q$, and checks $b_i \in Y$ and $\phi(s) = t \cdot b^c$

References

U. Maurer
Unifying Zero-Knowledge Proofs of Knowledge
AFRICACRYPT'09, 2nd International Conference on Cryptology in Africa, volume 5580 of *LNCS 5580*, pages 272–286, Gammarth, Tunisia, 2009.

M. Bellare, J. A. Garay, and T. Rabin
Batch verification with applications to cryptography and checking
LATIN'98: 3rd Latin American Symposium on Theoretical Informatics, LNCS 1380, pages 170–191, Campinas, Brazil, 1998.

K. Peng, C. Boyd, and E. Dawson
Batch zero-knowledge proof and verification and its applications
ACM Transactions on Information and System Security, 10(2), 2007.

Outline

Introduction

Review of Cryptographic Primitives

Batch Re-Encryption and Exponentiation Proofs

Proof of Knowledge of Permutation Matrix

Conclusion

Basic Re-Encryption Proof

- Let u and $u' = u \cdot E_{pk}(1, r)$ be publicly known encryptions
- Therefore, $u' \cdot u^{-1}$ is an encryption of 1 with randomization r
- P proves knowledge of r using the Σ-protocol for:

$$\begin{aligned} a &= r, \\ b &= u' \cdot u^{-1} \\ \phi(x) &= E_{pk}(1, x), \end{aligned}$$

- For ElGamal encryptions, we have $\phi(x) = (g^x, pk^x)$, where where $\phi : \underbrace{\mathbb{Z}_q}_{X} \rightarrow \underbrace{G_q \times G_q}_{Y}$

Batch Re-Encryption Proof

- Let $u_1, \ldots, u_n$ and $u'_1, \ldots, u'_n$ be publicly known encryptions, where $u'_i = u_i \cdot E_{pk}(1, r_i)$
- P proves knowledge of $r_1, \ldots, r_n$ as follows:
  - → V chooses random seed z
  - → P computes $(e_1, \ldots, e_n) = PRG(z)$
  - → P computes $u = \prod_i u_i^{e_i}$ and $u' = \prod_i (u'_i)^{e_i}$

$$u' = \prod_i (u'_i)^{e_i} = \prod_i u_i^{e_i} \prod_i E_{pk}(1, r_i)^{e_i} = u \cdot E_{pk}(1, \sum_i e_i r_i)$$

  - → P creates basic re-encryption proof for $u' \cdot u^{-1} = E_{pk}(1, \sum_i e_i r_i)$
- Implies small proofs of size $O(1)$

Batch Re-Encryption Proof under Permutation

- Let $u_1, \ldots, u_n$ and $u'_1, \ldots, u'_n$ be publicly known encryptions, where $u'_i = u_{\pi(i)} \cdot E_{pk}(1, r_{\pi(i)})$
- P proves knowledge of π and $r_1, \ldots, r_n$ as follows:
  - → V chooses random seed z
  - → P computes $(e_1, \ldots, e_n) = PRG(z)$
  - → P computes $u = \prod_i u_i^{e_i}$ and $u' = \prod_i (u'_i)^{e_{\pi(i)}}$

$$u' = \prod_i (u'_i)^{e_{\pi(i)}} = \prod_i u_{\pi(i)}^{e_{\pi(i)}} \prod_i E_{pk}(1, r_{\pi(i)})^{e_{\pi(i)}} = u \cdot E_{pk}(1, \sum_i e_i r_i)$$

  - → P creates basic re-encryption proof for $u' \cdot u^{-1} = E_{pk}(1, \sum_i e_i r_i)$
- Note that V can verify everything except $u' = \prod_i (u'_i)^{e_{\pi(i)}}$

Basic Exponentiation Proof

- Let $c = C(\alpha, s)$ be publicly known
- Let u and $u' = u^\alpha$ be publicly known values
- P proves knowledge of α and s using the Σ-protocol for:

$$a = (\alpha, s),$$
$$b = (c, u'),$$
$$\phi(x_1, x_2) = (C(x_1, x_2), u^{x_1})$$

- Remark: since α is no longer perfectly hidden for $u' = u^\alpha$, we could use $c = g^\alpha$ to commit to α (no randomization)

Batch Exponentiation Proof

- Let $c = C(\alpha, s)$ be publicly known
- Let $u_1, \ldots, u_n$ and $u'_1, \ldots, u'_n$ be publicly known, for $u'_i = u_i^\alpha$
- P proves knowledge of α and s as follows:
  - → V chooses random seed z
  - → P computes $(e_1, \ldots, e_n) = PRG(z)$
  - → P computes $u = \prod_i u_i^{e_i}$ and $u' = \prod_i (u'_i)^{e_i}$

$$u' = \prod_i (u'_i)^{e_i} = \prod_i (u_i^\alpha)^{e_i} = (\prod_i u_i^{e_i})^\alpha = u^\alpha$$

  - → P creates basic exponentiation proof for $u' = u^\alpha$ and c
- Implies small proofs of size $O(1)$

Batch Exponentiation Proof u. Permutation

- Let $c = C(\alpha, s)$ be publicly known
- Let $u_1, \ldots, u_n$ and $u'_1, \ldots, u'_n$ be publicly known, for $u'_i = u^{\alpha}_{\pi(i)}$
- P proves knowledge of π, α, and s as follows:
  - → V chooses random seed z
  - → P computes $(e_1, \ldots, e_n) = PRG(z)$
  - → P computes $u = \prod_i u_i^{e_i}$ and $u' = \prod_i (u'_i)^{e_{\pi(i)}}$

$$u' = \prod_i (u'_i)^{e_{\pi(i)}} = \prod_i (u^{\alpha}_{\pi(i)})^{e_{\pi(i)}} = (\prod_i u^{e_{\pi(i)}}_{\pi(i)})^{\alpha} = u^{\alpha}$$

  - → P creates basic exponentiation proof for $u' = u^{\alpha}$ and c
- Note that V can verify everything except $u' = \prod_i (u'_i)^{e_{\pi(i)}}$

What Remains?

Great, batch proofs almost work under permutation for both re-encryptions and exponentiations, but how can P prove the correct form of

$$u' = \prod_i (u_i')^{e_{\pi(i)}}$$

without revealing any information about π?

Necessity of Blinding u'

- Suppose that $u' = \prod_i (u'_i)^{e_{\pi(i)}}$ has been formed correctly
- V may then brute-force search for π, especially if n is small
- Let G be the group under consideration and $\{h_1, \ldots, h_k\}$ a generating set of G
  - → ElGamal Re-Encryption: $\{(g, 1), (1, g)\}$ for $G_q \times G_q$
  - → Exponentiation: $\{g\}$ for G_q
- P blinds u' as follows:
  1. Choose random exponents $\bar{t} = (t_1, \ldots, t_k) \in \mathbb{Z}_q^k$
  2. Let $b = \prod_i h_i^{t_i}$ be the blinding factor
  3. Compute $u'' = b \cdot u' = \prod_i h_i^{t_i} \prod_i (u'_i)^{e_{\pi(i)}}$

Blinded Batch Re-Encryption Proof

- Compute $(e_1, \ldots, e_n) = PRG(z)$ for seed z
- Compute $u = \prod_i u_i^{e_i}$
- Let $b = (g, 1)^{t_1} \cdot (1, g)^{t_2} = (g^{t_1}, g^{t_2})$ for $(t_1, t_2) \in_R \mathbb{Z}_q^2$
- Compute $u'' = (g^{t_1}, g^{t_2}) \cdot \prod_i (u'_i)^{e_{\pi(i)}}$
- Create basic re-encryption proof for

$$u'' \cdot u^{-1} = (g^{t_1}, g^{t_2}) \cdot E_{pk}(1, \sum_i e_i r_i)$$

Blinded Batch Exponentiation Proof

- Compute $(e_1, \ldots, e_n) = PRG(z)$ for seed z
- Compute $u = \prod_i u_i^{e_i}$
- Let $b = g^t$ for $t \in_R \mathbb{Z}_q$
- Compute $u'' = g^t \cdot \prod_i (u_i')^{e_{\pi(i)}}$
- Create basic exponentiation proof for $u'' = g^t \cdot u^\alpha$ and c

Outline

Introduction

Review of Cryptographic Primitives

Batch Re-Encryption and Exponentiation Proofs

Proof of Knowledge of Permutation Matrix

Conclusion

Permutation Matrix

- A permutation matrix is a square 0/1-matrix with exactly one 1 in each row and each column
- Let M be a permutation matrix and $\overline{x} = (x_1, \ldots, x_n)$, then

$$M \cdot \overline{x} = (x_{\pi(1)}, \ldots, x_{\pi(n)})$$

- Example: $\pi(1) = 2$, $\pi(2) = 3$, $\pi(3) = 1$

$$M = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}, \overline{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}, \text{ and therfore } M \cdot \overline{x} = \begin{pmatrix} x_2 \\ x_3 \\ x_1 \end{pmatrix}$$

Permutation Matrix Test

- Let M be an arbitrary square matrix over $\mathbb{Z}_q$
  - → $\overline{m}_i = (m_{i,1}, \ldots, m_{i,n})$ denotes the i-th row vector of M
  - → $\langle \overline{m}_i, \overline{x} \rangle = \sum_j m_{ij} \cdot x_j$ denotes the inner product of $\overline{m}_i$ and $\overline{x}$
- Theorem 1: M is a permutation matrix if and only if
  1. $\prod_i \langle \overline{m}_i, \overline{x} \rangle = \prod_i x_i$
  2. $M \cdot \overline{1} = \overline{1}$
- Counter-example: only the first condition holds

$$M = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -x_2 \\ -x_1 \end{pmatrix}, \text{ i.e., } \prod_i \langle \overline{m}_i, \overline{x} \rangle = x_1 \cdot x_2$$

Committed Permutation Matrix Test (1)

- Let $\widehat{m}_i = (m_{1,i}, \ldots, m_{n,i})$ denote the i-th column vector of M
- P commits column-wise to M by computing

$$C(M, \overline{s}) = (C(\widehat{m}_1, s_1), \ldots, C(\widehat{m}_n, s_n)) = (c_1, \ldots, c_n)$$

- P performs a batch proof to prove knowledge of M and $\overline{s}$
  1. V chooses random seed z
  2. P computes $(e_1, \ldots, e_n) = PRG(z)$
  3. P computes

$$c = \prod_i c_i^{e_i} = \cdots = C(\overline{e}', \sum_i e_i s_i), \text{ for } \overline{e}' = (e_{\pi(1)}, \ldots, e_{\pi(n)})$$

  4. P creates Pederson commitment proof for $c = C(\overline{e}', \sum_i e_i s_i)$

Committed Permutation Matrix Test (2)

To prove that M is a permutation matrix, Theorem 1 need to be demonstrated under the commitment $C(M, \bar{s})$

- First condition: P proves $\prod_i e'_i = \prod_i e_i$
  1. Compute commitments $c'_i = C(e'_i, s'_i)$ for $i = 2, \ldots, n$
  2. Compute commitments $c''_i = C(e'_1 \cdots e'_i, s''_i)$ for $i = 1, \ldots, n$
  3. Create commitment multiplication proofs for all (c''_{i-1}, c'_i, c''_i) (using a batch proof for $i = 2, \ldots, n$)
  4. Create Pedersen commitment proof for $c''_n = C(\prod_i e_i, s''_n)$
- Second condition: P proves $M \cdot \bar{1} = \bar{1}$
  1. Compute $d = \prod_i c_i = \cdots = C(\bar{1}, \sum_i s_i)$
  2. Create Pedersen commitment proof for $d = C(\bar{1}, \sum_i s_i)$

Outline

Introduction

Review of Cryptographic Primitives

Batch Re-Encryption and Exponentiation Proofs

Proof of Knowledge of Permutation Matrix

Conclusion

Recapitulation: Re-Encryption Shuffle (1)

- Common input: $u_1, \ldots, u_n, u'_1, \ldots, u'_n, (c_1, \ldots, c_n) = C(M, \overline{s})$
- Private input: π, $r_1, \ldots, r_n$, $\overline{s} = (s_1, \ldots, s_n)$
- V chooses random seed z
- P computes the following:
  1. $(e_1, \ldots, e_n) = PRG(z)$
  2. $u = \prod_i u_i^{e_i}$
  3. $u'' = (g^{t_1}, g^{t_2}) \cdot \prod_i (u'_i)^{e_{\pi(i)}}$ for $t_1, t_2 \in_R \mathbb{Z}_q$
  4. $c = \prod_i c_i^{e_i}$
  5. $c'_i = C(e'_i, s'_i)$ for $s'_i \in_R \mathbb{Z}_q$ and $i = 2, \ldots, n$
  6. $c''_i = C(e'_1 \cdots e'_i, s''_i)$ for $s'_i \in_R \mathbb{Z}_q$ and $i = 1, \ldots, n$
  7. $d = \prod_i c_i$

Recapitulation: Re-Encryption Shuffle (2)

- P creates the following composition of preimage proofs:
  1. Blinded re-encryption: $u'' \cdot u^{-1} = (g^{t_1}, g^{t_2}) \cdot E_{pk}(1, \sum_i e_i r_i)$
  2. Generalized Pederson commitment: $c = C(\bar{e}', \sum_i e_i s_i)$
  3. Commitment multiplications: c''_{i-1}, c'_i, c''_i (using a batch proof for $i = 2, \ldots, n$)
  4. Pedersen commitment: $c''_n = C(\prod_i e_i, s''_n)$
  5. Generalized Pedersen commitment: $d = C(\bar{1}, \sum_i s_i)$
- Note that if n is given, everything except u, u'', and the corresponding proof can be pre-computed in advance (offline)

Recapitulation: Exponentiation Shuffle (1)

- Common input:
  $u_1, \ldots, u_n, u'_1, \ldots, u'_n, c, (c_1, \ldots, c_n) = C(M, \bar{s})$
- Private input: π, α, s, $\bar{s} = (s_1, \ldots, s_n)$
- V chooses random seed z
- P computes the following:
  1. $(e_1, \ldots, e_n) = PRG(z)$
  2. $u = \prod_i u_i^{e_i}$
  3. $u'' = g^t \cdot \prod_i (u'_i)^{e_{\pi(i)}}$ $t \in_R \mathbb{Z}_q$
  4. $c = \prod_i c_i^{e_i}$
  5. $c'_i = C(e'_i, s'_i)$ for $s'_i \in_R \mathbb{Z}_q$, $i = 2, \ldots, n$
  6. $c''_i = C(e'_1 \cdots e'_i, s''_i)$ for $s'_i \in_R \mathbb{Z}_q$, $i = 1, \ldots, n$
  7. $d = \prod_i c_i$

Recapitulation: Exponentiation Shuffle (2)

- P creates the following composition of preimage proofs:
  1. Pedersen commitment: $c = C(\alpha, s)$
  2. Generalized Pederson commitment: $c_i = C(\widehat{m}_i, s_i)$ (using batch proof for $j = 1, \ldots, n$)
  3. Blinded exponentiation: $u'' = g^t \cdot u^\alpha$
  4. Generalized Pederson commitment: $c = C(\overline{e}', \sum_i e_i s_i)$
  5. Commitment multiplications: c''_{i-1}, c'_i, c''_i (batch proof for $i = 2, \ldots, n$)
  6. Pedersen commitment: $c''_n = C(\prod_i e_i, s''_n)$
  7. Generalized Pedersen commitment: $d = C(\overline{1}, \sum_i s_i)$
- Note that if n is given, everything except u, u'', and the corresponding proof can be pre-computed in advance (offline)

Open Quesions

- Can we make the proof non-interactive?
  - → Using non-interactive batch proofs (Fiat-Shamir)
  - → How secure is this?
  - → Does it affect pre-computations?
- Can we skip some commitments?
  - → The paper contains a commitment to $\bar{t}$, but this seems not to be necessary (already skipped)
  - → In the chained commitment multiplication proof, the output of one proof is one of the inputs of the next proof

Conclusion

- The proof is a composition of several basic preimage and batch preimage proofs
- The size of the proof is $O(n)$
- A large portion of the proof can be computed offline
  - → Ok, if n is known in advance
  - → If n is unknown, the pre-computation can be done for an upper bound $N \geq n$, and when the input data arrives, it is "filled up" with trivial values
- The proof can be generalized to incorporate:
  - → Restrictions on π (e.g., that π is a rotation)
  - → Any "shuffle-friendly map" (re-encryptions, exponentiations, partial decryptions, or combinations thereof)
- Great job, Douglas!!!