University of Tartu, Cryptography Research Seminar

Additive Conditional Disclosure of Secrets And Applications

Sven Laur

Helsinki University of Technology, Finland

Helger Lipmaa

Cybernetica AS and University of Tartu, Estonia

Outline

- Motivation
- Previous Work
- New Construction
- Conclusions

Some Well-Known Public-Key Cryptosystems

- The old one: RSA
  - ⋆ Best known, most used
  - ⋆ Problems with security proofs, has only one instantiation, slow, lacks algebraic clarity
- The almost-as-old one: ElGamal
  - ⋆ Clear security proofs, many instantiations (e.g., ECC), some instantiations are relatively efficient
  - ⋆ Nice algebraic properties

ElGamal: Description

- Let $\mathbf{G}$ be a finite multiplicative group, and $\mathbf{H}$ be its subgroup of prime order, s.t. Decisional Diffie-Hellman Problem in $\mathbf{H}$ is difficult
  - ⋆ For example: $\mathbf{G} = \mathbb{Z}_p$, $|p| = 1024$, $g \in \mathbb{Z}_p$ s.t. $\log_2 \sharp\langle g\rangle \approx 160$; $\mathbf{H} := \langle g\rangle$. Define $q := \sharp \mathbf{H}$
- Fix a generator g of $\mathbf{H}$ as the system parameter
- Receiver generates a random secret key $sk_R \leftarrow \mathbb{Z}_q$, and sets $pk_R \leftarrow g^{sk_R}$
- Encryption: $E_{pk_R}(m; r) = (m \cdot pk_R^r, g^r)$ where $r \leftarrow \mathbb{Z}_q$
- Decryption: given $E_{pk_R}(m; r) = (u, v) = (m \cdot pk_R^r, g^r)$ and sk_R, compute $u/v^{sk_R} = m$

Indistinguishability against Chosen Plaintext Attacks

- Choose a random key pair $(\mathbf{sk}, \mathbf{pk})$, give $\mathbf{pk}$ to Adversary

- Adversary generates two plaintexts $\mathbf{m_0}, \mathbf{m_1}$ and gives them to Bob

- Bob tosses a coin, $\mathbf{b} \leftarrow \{\mathbf{0}, \mathbf{1}\}$, chooses a random $\mathbf{r} \leftarrow \mathbb{Z}_\mathbf{q}$, and sends $\mathbf{E_{pk}(m_b; r)}$ to Adversary

- Adversary outputs $\mathbf{b'}$

- Adversary (τ, ε)-breaks IND-CPA security if it works in time τ and $\Pr[\mathbf{b} = \mathbf{b'}] \geq \varepsilon$

- Fact: ElGamal is IND-CPA secure (given DDH assumption)

ElGamal Is Homomorphic

- A pkc is multiplicatively homomorphic if

$$\mathbf{E_{pk}(m_0; r_0) \cdot E_{pk}(m_1; r_1) = E_{pk}(m_0 m_1; \cdot)}$$

- ElGamal is multiplicatively homomorphic: given

$$\mathbf{E_{pk}(m_0; r_0) = (m_0 \cdot pk^{r_0}, g^{r_0})}$$

and

$$\mathbf{E_{pk}(m_1; r_1) = (m_1 \cdot pk^{r_1}, g^{r_1})} \ ,$$

$$\mathbf{E_{pk}(m_0; r_0) E_{pk}(m_1; r_1) = (m_0 m_1 \cdot pk^{r_0 + r_1}, g^{r_0 + r_1}) = E_{pk}(m_0 m_1; r_0 + r_1)}$$

Why Does Homomorphism Help?

1. Receiver sends $\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r})$ to Sender, who returns $\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r})^\sigma = \mathbf{E}_{\mathbf{pk_R}}(\varrho^\sigma; \cdot)$, Receiver gets back ϱ^σ without knowing σ

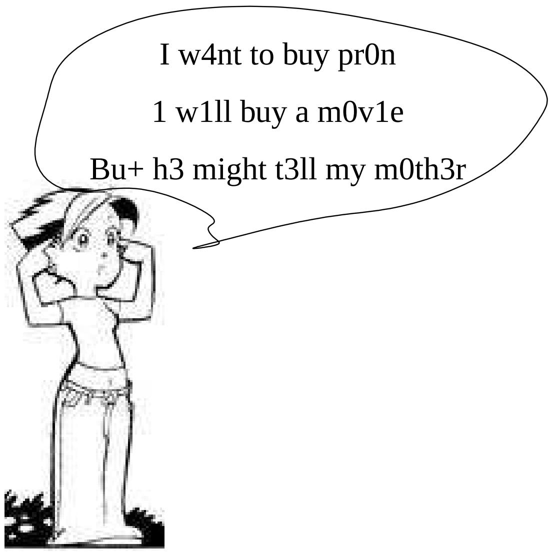
2. Receiver sends $(\mathbf{E}_{\mathbf{pk_R}}(\varrho_\mathbf{i}; \mathbf{r_i}))_{\mathbf{i} \in [\mathbf{N}]}$ to Sender, who returns $\prod \mathbf{E}_{\mathbf{pk_R}}(\varrho_\mathbf{i}; \mathbf{r_i})^{\sigma_\mathbf{i}} = \mathbf{E}_{\mathbf{pk_R}}(\prod \varrho_\mathbf{i}^{\sigma_\mathbf{i}}; \cdot)$

3. Receiver sends $\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r})$ to Sender, who returns ($*$ is a random element)

$$(\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r}) / \mathbf{E}_{\mathbf{pk_R}}(\sigma; \mathbf{anything}))^* \cdot \mathbf{E}_{\mathbf{pk_R}}(1; *) = \mathbf{E}_{\mathbf{pk_R}}((\varrho/\sigma)^*; *)$$
$$= \begin{cases} \mathbf{E}_{\mathbf{pk_R}}(1; *) , & \varrho = \sigma \\ \mathbf{E}_{\mathbf{pk_R}}(*; *) , & \varrho \neq \sigma \end{cases} .$$

$\Rightarrow$ **In general, Receiver and Sender can compute on ciphertexts**

Oblivious Transfer: Vots Dat?



* *Parental advisory: this is not the only application of OT. Stay tuned!*

Oblivious Transfer: Vots Dat?

ϱ

$(\sigma[\mathbf{1}], \ldots, \sigma[\mathbf{n}])$

ϱ →

← $\sigma[\varrho]$

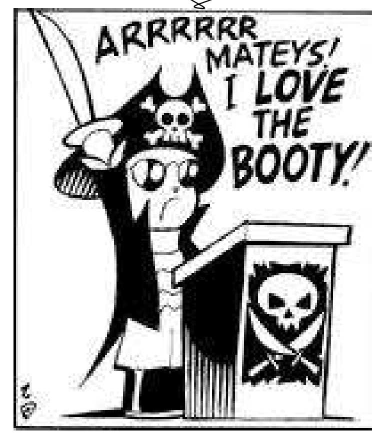
He l00ks fri3ndly but m4y b3 n0t...

ShOuld 1 ask my fr13nd cryp+0graph3r?

Oblivious Transfer: Vots Dat?

ϱ

$(\sigma[1], \ldots, \sigma[n])$



$\mathbf{E_K}(\varrho)$ →

← $(\mathbf{E_K}(\mathbf{f}(\sigma[\mathbf{1}], \varrho)), \ldots, \mathbf{E_K}(\mathbf{f}(\sigma[\mathbf{n}], \varrho)))$

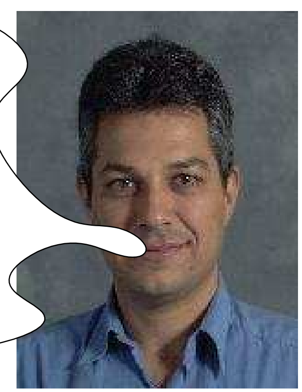
0hn0!
Sl1ck h4s 1,000,000 movies
1ll f1nd an0th3r cryp+3r

Of course you can do that!

It just takes a little work. . .

Like the total length of all movies

. . . times a small constant factor



AIR Oblivious Transfer Protocol

- OT: Receiver has input $\varrho \in [\mathbf{N}]$, Sender has input $\sigma = (\sigma[\mathbf{1}], \ldots, \sigma[\mathbf{N}])$. Receiver obtains $\sigma[\varrho]$ without getting *any extra* information on σ; Sender gets *no* information about ϱ

AIR Oblivious Transfer Protocol

- OT: Receiver has input $\varrho \in [\mathbf{N}]$, Sender has input $\sigma = (\sigma[\mathbf{1}], \ldots, \sigma[\mathbf{N}])$.

- Receiver sends $\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r})$ to Sender

- For any $\mathbf{j} \in [\mathbf{N}]$, Sender returns

$$\mathbf{c_j} \leftarrow (\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r}) / \mathbf{E}_{\mathbf{pk_R}}(\mathbf{j}; \mathbf{anything}))^* \cdot \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *)$$
$$= \begin{cases} \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *) \ , & \varrho = \mathbf{j} \\ \mathbf{E}_{\mathbf{pk_R}}(*; *) \ , & \varrho \neq \mathbf{j} \end{cases} .$$

- Receiver decrypts $\mathbf{c}_\varrho$, obtaining $\sigma[\varrho]$

- Note that if $\varrho \notin [\mathbf{N}]$ then $\mathbf{D}_{\mathbf{sk_R}}(\mathbf{c_j})$ is a random element of $\mathbf{H}$ for all $\mathbf{j} \in [\mathbf{N}]$

Need More!

- Consider scalar product computation: $\mathbf{answer} = \sum_{\mathbf{i}} \varrho_{\mathbf{i}} \sigma_{\mathbf{i}}$

- Recall a previous protocol:

  ⋆ Receiver sends $(\mathbf{E}_{\mathbf{pk_R}}(\varrho_{\mathbf{i}}; \mathbf{r_i}))_{\mathbf{i} \in [\mathbf{N}]}$ to Sender

  ⋆ Sender returns $\prod \mathbf{E}_{\mathbf{pk_R}}(\varrho_{\mathbf{i}}; \mathbf{r_i})^{\sigma_{\mathbf{i}}} = \mathbf{E}_{\mathbf{pk_R}}(\prod \varrho_{\mathbf{i}}^{\sigma_{\mathbf{i}}}; \cdot)$

- Not yet SP but close but no cigar…

Need More!

- Receiver sends $(\mathbf{E_{pk_R}(g^{\varrho_i}; r_i)})_{\mathbf{i \in [N]}}$ to Sender
  Sender returns $\prod \mathbf{E_{pk_R}(g^{\varrho_i}; r_i)^{\sigma_i} = E_{pk_R}(g^{\sum \varrho_i \sigma_i}; \cdot)}$

- Receiver obtains SP by computing discrete logarithm of $\mathbf{g^{\sum \varrho_i \sigma_i}}$

- Might be useful if $\sum \varrho_\mathbf{i} \sigma_\mathbf{i}$ is *small* which is the case *sometimes*

- DL is a lift from a multiplicative group $\mathbf{H}$ to an additive group $\mathbb{Z}_\mathbf{q}$

- We often need a cryptosystem that is *additively homomorphic*: $\mathbf{E_{pk}(m_0; r_0) E_{pk}(m_1; r_1) = E_{pk}(m_0 + m_1; \cdot)}$, i.e., that works directly in $\mathbb{Z}_\mathbf{q}$ (without a lift)

Paillier Cryptosystem

- For random large primes $\mathbf{p}$ and $\mathbf{q}$, set $\mathbf{n} \leftarrow \mathbf{pq}$. $\mathbf{n}$ is public key, $(\mathbf{p}, \mathbf{q})$ is secret key.

- Encryption: $\mathbf{c} = \mathbf{E_{pk}(m; r)} := (\mathbf{1} + \mathbf{mn})\mathbf{r^n} \mod \mathbf{n^2}$

- Additive homomorphism:

$$\begin{aligned}&(\mathbf{1} + \mathbf{m_0 n})(\mathbf{1} + \mathbf{m_1 n})\mathbf{r_0^n r_1^n} \\ &\quad = (\mathbf{1} + (\mathbf{m_0} + \mathbf{m_1})\mathbf{n})(\mathbf{r_0 r_1})^{\mathbf{n}} \mod \mathbf{n^2}\end{aligned}$$

- Paillier PKC is IND-CPA secure if given a random $\mathbf{y} \in \mathbb{Z}_{\mathbf{n^2}}$ it is hard to decide if $\mathbf{y}$ is an $\mathbf{n}$th residue (DCRP)

Scalar Product With Additive Homomorphism

- Consider scalar product computation: $\mathbf{answer} = \sum_{\mathbf{i}} \varrho_{\mathbf{i}} \sigma_{\mathbf{i}}$

- Multiplicative homomorphism: Receiver sends $(\mathbf{E}_{\mathbf{pk_R}}(\varrho_{\mathbf{i}}; \mathbf{r_i}))_{\mathbf{i} \in [\mathbf{N}]}$ to Sender
  Sender returns $\prod \mathbf{E}_{\mathbf{pk_R}}(\varrho_{\mathbf{i}}; \mathbf{r_i})^{\sigma_{\mathbf{i}}} = \mathbf{E}_{\mathbf{pk_R}}(\prod \varrho_{\mathbf{i}}^{\sigma_{\mathbf{i}}}; \cdot)$

- Additive homomorphism: Receiver sends $(\mathbf{E}_{\mathbf{pk_R}}(\varrho_{\mathbf{i}}; \mathbf{r_i}))_{\mathbf{i} \in [\mathbf{N}]}$ to Sender
  Sender returns $\prod \mathbf{E}_{\mathbf{pk_R}}(\varrho_{\mathbf{i}}; \mathbf{r_i})^{\sigma_{\mathbf{i}}} = \mathbf{E}_{\mathbf{pk_R}}(\sum \varrho_{\mathbf{i}} \sigma_{\mathbf{i}}; \cdot)$

- Receiver recovers $\sum \varrho_{\mathbf{i}} \sigma_{\mathbf{i}}$ by decrypting the result

- Security?

SP: Privacy

- Sender only sees random encryptions of some elements
- Thus, if Sender can break Receiver's privacy (guess which elements he sees) then he can also break IND-CPA security of PKC

$\Rightarrow$ Protocol is computationally Receiver-private, if PKC is IND-CPA secure

- What about Sender's privacy?

SP: Sender's Privacy

- If we are computing SP of Boolean values, then Sender's privacy is protected given that Receiver inputs correct values
  - ⋆ If $\varrho_i \in \{0, 1\}$ then the only value Receiver sees is $\sum \varrho_i \sigma_i$, the scalar product
- If $\varrho_i \notin \{0, 1\}$ then Receiver recovers more information:
  - ⋆ Take $\varrho_i \leftarrow 2^{i-1}$
  - ⋆ $1\sigma_1 + 2\sigma_2 + 4\sigma_3 + \ldots$ reveals Sender's input!

SP: Security (2)

- We established: Sender's privacy is guaranteed if Receiver's inputs belong to *valid input sets*, $\varrho_{\mathbf{i}} \in \mathbf{Valid(i)}$

- Standard way to guarantee Sender's privacy: Receiver proves in zero-knowledge that her inputs are correct

  ⋆ E.g., Receiver proves that $\varrho_{\mathbf{i}} \in \{\mathbf{0}, \mathbf{1}\}$ for all $\mathbf{i} \in [\mathbf{N}]$

- Unfortunately, zero-knowledge protocols take $\mathbf{3+}$ rounds

- *Non-interactive zero-knowledge* requires non-standard assumptions (random oracle, ...)

- We would like to stick to the minimum assumption that PKC is IND-CPA secure *and* have a one-round protocol

Recall: AIR Oblivious Transfer Protocol

- OT: Receiver has input $\varrho \in [\mathbf{N}]$, Sender has input $\sigma = (\sigma[\mathbf{1}], \dots, \sigma[\mathbf{N}])$.
- Receiver sends $\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r})$ to Sender // pkc is mult. homomorphic
- For any $\mathbf{j} \in [\mathbf{N}]$, Sender returns

$$\mathbf{c_j} \leftarrow (\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r})/\mathbf{E}_{\mathbf{pk_R}}(\mathbf{j}; \mathbf{anything}))^{*} \cdot \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *)$$
$$= \begin{cases} \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *) \ , & \varrho = \mathbf{j} \\ \mathbf{E}_{\mathbf{pk_R}}(*; *) \ , & \varrho \neq \mathbf{j} \end{cases} .$$

- Receiver decrypts $\mathbf{c_j}$, obtaining $\sigma[\mathbf{j}]$
- Note that if $\varrho \notin [\mathbf{N}]$ then $\mathbf{D}_{\mathbf{sk_R}}(\mathbf{c_j})$ is a random element of $\mathbf{H}$

AIR Protocol for Arbitrary Index Ranges

- OT: Receiver has input $\varrho \in \mathbf{S}$, Sender has input $\sigma = (\sigma[\mathbf{j}])_{\mathbf{j} \in \mathbf{S}}$. Receiver obtains $\sigma[\varrho]$ without getting any extra information on σ; Sender gets no information about ϱ

- Receiver sends $\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r})$ to Sender // pkc is mult. homomorphic

- For any $\mathbf{j} \in \mathbf{S}$, Sender returns

$$\mathbf{c_j} \leftarrow (\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r}) / \mathbf{E}_{\mathbf{pk_R}}(\mathbf{j}; \mathbf{anything}))^{*} \cdot \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *)$$
$$= \begin{cases} \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *) \ , & \varrho = \mathbf{j} \\ \mathbf{E}_{\mathbf{pk_R}}(*; *) \ , & \varrho \neq \mathbf{j} \end{cases} .$$

- Receiver decrypts $\mathbf{c_j}$, obtaining $\sigma[\mathbf{j}]$

- Note that if $\varrho \notin \mathbf{S}$ then $\mathbf{D}_{\mathbf{sk_R}}(\mathbf{c_j})$ is a random element of $\mathbf{H}$

Conditional Disclosure of Secrets: Idea

- Any AH one-round protocol: Receiver has inputs $(\varrho_1, \ldots, \varrho_M) \in \mathbf{Valid(1)}$ $\times \cdots \times \mathbf{Valid(M)}$, Sender has input $(\mathbf{S[1]}, \ldots, \mathbf{S[N]})$. Receiver obtains $(\mathbf{f_1}(\varrho, \mathbf{S}), \ldots, \mathbf{f_L}(\varrho, \mathbf{S}))$ without getting any extra information on $\mathbf{S}$; Sender gets no information about ϱ

- Protocol goal:

  ⋆ For any $\mathbf{j} \in [\mathbf{M}]$, Receiver sends $\mathbf{E_{pk_R}}(\varrho_\mathbf{j}; \mathbf{r})$ to Sender

  ⋆ For any $\mathbf{j} \in [\mathbf{L}]$, Sender returns $\mathbf{c_j} = \begin{cases} \mathbf{E_{pk_R}}(\mathbf{f_j}(\varrho, \mathbf{S}); *), & \varrho_\mathbf{j} \in \mathbf{Valid(j)} \\ \mathbf{E_{pk_R}}(*; *), & \varrho_\mathbf{j} \notin \mathbf{Valid(j)} \end{cases}$.

  ⋆ For any $\mathbf{j} \in [\mathbf{L}]$, Receiver decrypts $\mathbf{c_j}$, obtaining $\mathbf{f_j}(\varrho, \mathbf{S})$

- But how to construct such a protocol?

CDS: La Technique

- For any $\mathbf{j} \in [\mathbf{M}]$, Receiver sends $\mathbf{E}_{\mathbf{pk_R}}(\varrho_\mathbf{j}; \mathbf{r})$ to Sender

- For any $\mathbf{i} \in [\mathbf{M}]$, $\mathbf{j} \in [\mathbf{L}]$: Sender generates a new random string $\mathbf{t_{ij}}$, and performs AIR OT on a database $\mathbf{S_{ij}}$, where $\mathbf{S_{ij}[k] = t_{ij}}$ for $\mathbf{k} \in \mathbf{Valid(i)}$. Receiver gets back encryptions of $\mathbf{t_{ij}}$, $\mathbf{j} \in [\mathbf{L}]$, iff $\varrho_\mathbf{i} \in \mathbf{Valid(i)}$

- For any $\mathbf{j} \in [\mathbf{L}]$, Sender computes $\mathbf{c_j}$, a random encryption of $\mathbf{f_j}(\varrho, \sigma)$. Sender sends $\mathbf{c'_j} \leftarrow \mathbf{c_j} \cdot \mathbf{E}_{\mathbf{pk_R}}(\sum_{\mathbf{i} \in [\mathbf{M}]} \mathbf{t_{ij}}; *)$ to Receiver.

- If Receiver's *all* inputs were valid then she knows all values $\sum \mathbf{t_{ij}}$ and thus can obtain $\mathbf{f_j}(\varrho, \sigma)$ for *all* $\mathbf{j}$. If *any* input was invalid, she obtains *no* answer

- Thus, this compound protocol is Sender-private!

Done? Not Yet!

- AIR OT uses multiplicatively homomorphic PKC — not important, can work with an additive one

- AIR OT runs in a group $\mathbf{H}$ of prime order, while Paillier plaintexts belong to $\mathbb{Z}_{\mathbf{pq}}$! — problem

- Thus, CDS-transformed protocols are Sender-private if
  - ⋆ AIR OT is secure (=ElGamal is IND-CPA secure=DDH is hard) and
  - ⋆ Paillier is IND-CPA secure (=DCRP is hard)

- Can we use AIR over a composite modulus $\mathbf{n}$?

Recall: AIR Oblivious Transfer Protocol

- OT: Receiver has input $\varrho \in [\mathbf{N}]$, Sender has input $\sigma = (\sigma[\mathbf{1}], \ldots, \sigma[\mathbf{N}])$. Receiver obtains $\sigma[\varrho]$ without getting any extra information on σ; Sender gets no information about ϱ

- Receiver sends $\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r})$ to Sender // Additively homomorphic pkc

- For any $\mathbf{j} \in [\mathbf{N}]$, Sender generates $\mathbf{r_j} \leftarrow \mathbb{Z}_\mathbf{q}$ and returns

$$\mathbf{c_j} \leftarrow (\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r}) / \mathbf{E}_{\mathbf{pk_R}}(\mathbf{j}; \mathbf{0}))^{\mathbf{r_j}} \cdot \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *)$$
$$= \begin{cases} \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *), & \varrho = \mathbf{j} \\ \mathbf{E}_{\mathbf{pk_R}}(*; *), & \varrho \neq \mathbf{j} \end{cases} .$$

- Receiver decrypts $\mathbf{c_j}$, obtaining $\sigma[\mathbf{j}]$

- Wrong! $\mathbf{n}$ is composite, and Sender does not know $\mathbf{q}$!

AIR Oblivious Transfer Protocol with Composite Modulus?

- OT: Receiver has input $\varrho \in [\mathbf{N}]$, Sender has input $\sigma = (\sigma[\mathbf{1}], \ldots, \sigma[\mathbf{N}])$. Receiver obtains $\sigma[\varrho]$ without getting any extra information on σ; Sender gets no information about ϱ

- Receiver sends $\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r})$ to Sender // Additively homomorphic pkc

- For any $\mathbf{j} \in [\mathbf{N}]$, Sender generates $\mathbf{r_j} \leftarrow \mathbb{Z}_\mathbf{n}$ and returns

$$\mathbf{c_j} \leftarrow (\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r}) / \mathbf{E}_{\mathbf{pk_R}}(\mathbf{j}; \mathbf{0}))^{\mathbf{r_j}} \cdot \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *)$$
$$= \begin{cases} \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *), & \varrho = \mathbf{j} \\ \mathbf{E}_{\mathbf{pk_R}}(*; *), & \varrho \neq \mathbf{j} \end{cases} .$$

- Receiver decrypts $\mathbf{c_j}$, obtaining $\sigma[\mathbf{j}]$

- Better?

Still Wrong!

- For any $\mathbf{j} \in [\mathbf{N}]$, Sender generates $\mathbf{r_j} \leftarrow \mathbb{Z}_\mathbf{n}$ and returns

$$\mathbf{c_j} \leftarrow (\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r}) / \mathbf{E}_{\mathbf{pk_R}}(\mathbf{j}; \mathbf{0}))^{\mathbf{r_j}} \cdot \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *)$$
$$= \begin{cases} \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *), & \varrho = \mathbf{j} \\ \mathbf{E}_{\mathbf{pk_R}}(*; *), & \varrho \neq \mathbf{j} \end{cases} .$$

- Attack: Suppose ϱ is such that $\varrho \equiv \mathbf{i_1} \pmod{\mathbf{p}}$ and $\varrho \equiv \mathbf{i_2} \pmod{\mathbf{q}}$, for $\mathbf{i_1} \neq \mathbf{i_2} \in [\mathbf{N}]$

- Receiver obtains $\mathbf{b_1} \leftarrow (\mathbf{i_1} - \varrho)^{\mathbf{r_{i_1}}} + \sigma[\mathbf{i_1}] \mod \mathbf{pq}$ and $\mathbf{b_2} \leftarrow (\mathbf{i_2} - \varrho)^{\mathbf{r_{i_2}}} + \sigma[\mathbf{i_2}] \mod \mathbf{pq}$

- Now, $\mathbf{b_1} \equiv \sigma[\mathbf{i_1}] \mod \mathbf{p}$ and $\mathbf{b_2} \equiv \sigma[\mathbf{i_2}] \mod \mathbf{q}$, thus Receiver got information about both $\sigma[\mathbf{i_1}]$ and $\sigma[\mathbf{i_2}]$!

New OT Protocol

- Fix "suitable" ℓ ($\ell \approx \mathbf{433}$ is sufficient)

- Receiver sends $\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r})$ to Sender // Additively homomorphic pkc

- For any $\mathbf{j} \in [\mathbf{N}]$, Sender generates $\mathbf{r_j} \leftarrow \mathbb{Z}_\mathbf{n}$ and returns

$$\mathbf{c_j} \leftarrow (\mathbf{E}_{\mathbf{pk_R}}(\varrho; \mathbf{r})/\mathbf{E}_{\mathbf{pk_R}}(\mathbf{j}; \mathbf{0}))^{\mathbf{r_j}} \cdot \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}] + \mathbf{2}^\ell \cdot *; *)$$
$$= \begin{cases} \mathbf{E}_{\mathbf{pk_R}}(\sigma[\mathbf{j}]; *), & \varrho = \mathbf{j} \\ \mathbf{E}_{\mathbf{pk_R}}(\mathbf{almost}\ *; *), & \varrho \neq \mathbf{j} \end{cases} .$$

- Receiver decrypts $\mathbf{c_j}$, obtaining $\sigma[\mathbf{j}]$

- Note that if $\varrho \notin [\mathbf{N}]$ then $\mathbf{D}_{\mathbf{sk_R}}(\mathbf{c_j})$ is an almost random element of $\mathbf{H}$

Applications I: Private SP

- Consider scalar product computation: $\mathbf{answer} = \sum_{\mathbf{i}} \varrho_{\mathbf{i}}\sigma_{\mathbf{i}}$

- Receiver sends $(\mathbf{E}_{\mathbf{pk_R}}(\varrho_{\mathbf{i}}; \mathbf{r_i}))_{\mathbf{i} \in [\mathbf{N}]}$ to Sender
  Sender returns $\prod \mathbf{E}_{\mathbf{pk_R}}(\varrho_{\mathbf{i}}; \mathbf{r_i})^{\sigma_{\mathbf{i}}} = \mathbf{E}_{\mathbf{pk_R}}(\sum \varrho_{\mathbf{i}}\sigma_{\mathbf{i}}; \cdot)$

- Receiver recovers $\sum \varrho_{\mathbf{i}}\sigma_{\mathbf{i}}$ by decrypting the result

- If Receiver is malicious, then this is not Sender-private

- Private SP protocol is very popular in PPDM, see [GLLM04]

- Many other similar protocols (linear algebra, PPDM)

Scalar Product With Additive Homomorphism + CDS

- Consider scalar product computation: $\mathbf{answer} = \sum_{\mathbf{i}} \varrho_{\mathbf{i}} \sigma_{\mathbf{i}}$

- Receiver sends $(\mathbf{E}_{\mathbf{pk_R}}(\varrho_{\mathbf{i}}; \mathbf{r_i}))_{\mathbf{i} \in [\mathbf{N}]}$ to Sender
  Sender returns $\prod \mathbf{E}_{\mathbf{pk_R}}(\varrho_{\mathbf{i}}; \mathbf{r_i})^{\sigma_{\mathbf{i}}} = \mathbf{E}_{\mathbf{pk_R}}(\sum \varrho_{\mathbf{i}} \sigma_{\mathbf{i}}; \cdot)$

- Receiver recovers $\sum \varrho_{\mathbf{i}} \sigma_{\mathbf{i}}$ by decrypting the result

- Add CDS: Receiver recovers $\sum \varrho_{\mathbf{i}} \sigma_{\mathbf{i}}$ only if her inputs were from correct sets

- Thus, we get Sender-privacy! (without any computational assumptions)

Applications II: Communication-Efficient OT

- CPIR: Receiver has input $\varrho \in [N]$, Sender has input $\sigma = (\sigma[1], \ldots, \sigma[N])$.
  Receiver obtains $\sigma[\varrho]$ with possibly getting more information about σ; Sender gets no information about ϱ

- Lipmaa's CPIR [2005]: based on AH PKC, one round, secure if PKC is IND-CPA secure, communication $\Theta(\log^2 N)$

Applications II: Communication-Efficient OT

CPIR$\to$OT:

(1) Receiver sends first message of CPIR to Sender, (1') Receiver sends first message of the new OT to Sender

(2) Sender applies the new OT protocol to σ, getting database $\mathbf{c} = (\mathbf{c_1}, \dots, \mathbf{c_N})$, *but does not send* $\mathbf{c}$ *to Receiver*. Instead, (2') Sender applies the CPIR protocol to $\mathbf{c}$, sending some values back to Receiver

- Receiver obtains some ciphertexts, and recovers $\sigma[\varrho]$. In the original CPIR she also might have obtained more information, but due to use of the OT protocol "inside", this additional information will be gargage

- Result: OT protocol, based on AH PKC, one round, secure if PKC is IND-CPA secure, communication $\Theta(\log^2 N)$

Applications III: Millionaire's Problem

- Receiver has $0 \leq \varrho < 2^{\ell}$, Sender has $0 \leq \sigma < 2^{\ell}$. Receiver gets only to know if $\varrho > \sigma$, Sender obtains no information

- Write $\varrho = \sum_{k=0}^{\ell-1} \varrho_k 2^k$, $\sigma = \sum_{k=0}^{\ell-1} \sigma_k 2^k$. Then $\varrho > \sigma$ iff

$$
\begin{aligned}
&[\varrho_{\ell-1} = 1 \wedge \sigma_{\ell-1} = 0] \vee \\
&([\varrho_{\ell-1} = \sigma_{\ell-1}] \wedge [\varrho_{\ell-2} = 1 \wedge \sigma_{\ell-2} = 0]) \vee \\
&\ldots \\
&([\varrho_{\ell-1} = \sigma_{\ell-1}] \wedge [\varrho_1 = \sigma_1] \wedge [\varrho_0 = 1 \wedge \sigma_0 = 0]) \ .
\end{aligned}
$$

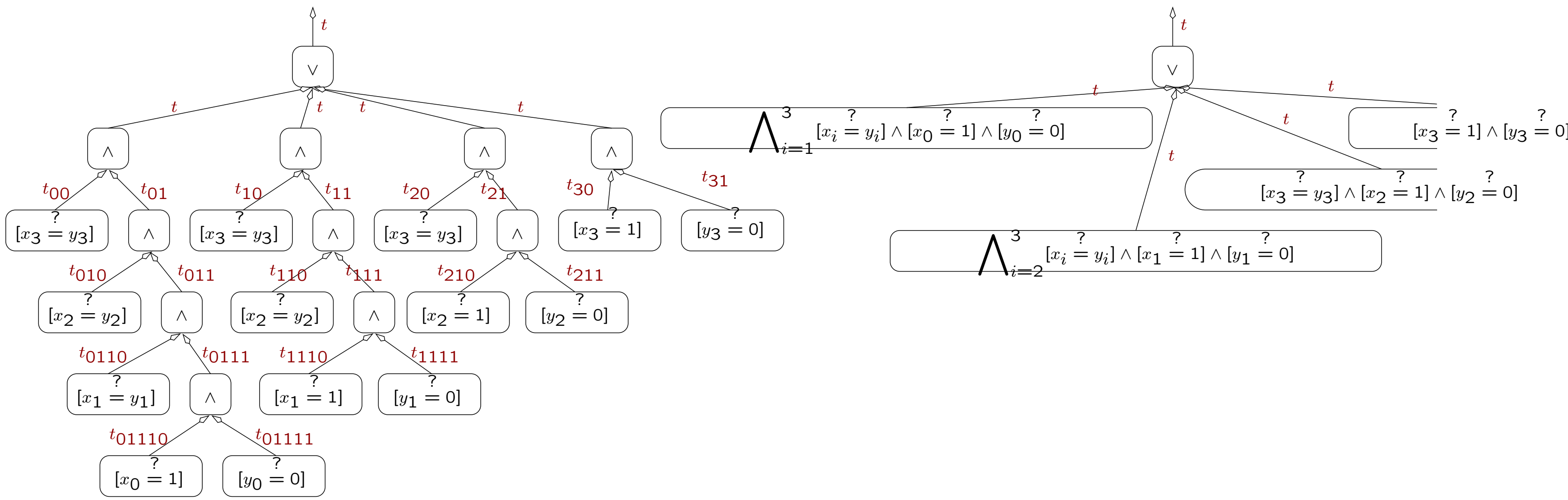
- Write down a circuit where internal nodes correspond to $\vee$ and $\wedge$ gates and leaves correspond to affine equality tests $\sum \gamma_{ij} \varrho_i + \delta_j = 0$

- Use CDS on circuits; Receiver gets answer if some Boolean formula holds on her inputs

Applications III: Circuit Evaluation

- Assign a random secret to the output wire of the circuit
- $\forall \vee$ gate ψ: assign the output secret $\mathbf{t}_\psi$ of ψ to every input wire of ψ
- $\forall \wedge$ gate ψ: Generate random $\mathbf{t_1}$ and $\mathbf{t_2}$ s.t. $\mathbf{t_1} + \mathbf{t_2} = \mathbf{t}_\psi$, assign $\mathbf{t_1}$ and $\mathbf{t_2}$ to input wires
- Receiver transfers $\mathbf{E_K}(\varrho_\mathbf{j}; \cdot)$ for $\mathbf{j} \in [\ell]$, Sender sends back $\mathbf{E_K}(\mathbf{value}_\psi; \cdot)$ for every conjunctive affine equality test
- Receiver obtains secrets, corresponding to tests that are consistent with her inputs
- Receiver recursively obtains inner secrets, finally receiving the output secret of the secret if her inputs were correct

Applications III: Circuit Evaluation



- Circuit on the left: protocol with communication $\Theta(\ell^2)$
- Circuit on the right: protocol with communication $\Theta(\ell)$

Conclusions

- CDS is a powerful tool, especially when coupled with AH PKC
- Goal 1: Popularise CDS
- Goal 2 (and a mean for goal 1): Propose efficient protocols for specific interesting problems
- OT, millionaire's, scalar product: can find efficient protocols for others, too
- All protocols are one-round, private if PKC is IND-CPA secure, and quite efficient

Any questions?

Caveat: This presentation is based on a draft version of the paper! Paper will be available in 1-2 weeks