Outsourced Pattern Matching

Daniele Venturi

40th International Colloquium on Automata, Languages and Programming (ICALP 2013)
Latvia, Riga 09-07-2013

AARHUS UNIVERSITY

Joint work with Sebastian Faust and Carmit Hazay

Delegatable computation

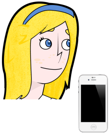
- Outsourcing computation of a public function f to a possibly malicious cloud provider (a.k.a. the server)

Delegatable computation

- Outsourcing computation of a public function f to a possibly malicious cloud provider (a.k.a. the server)

Weak client

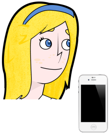
input x



Delegatable computation

- Outsourcing computation of a public function f to a possibly malicious cloud provider (a.k.a. the server)

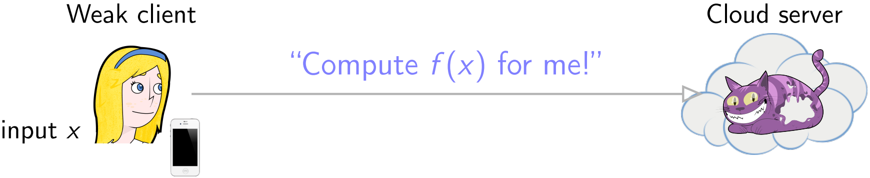
Weak client

input x 

Cloud server

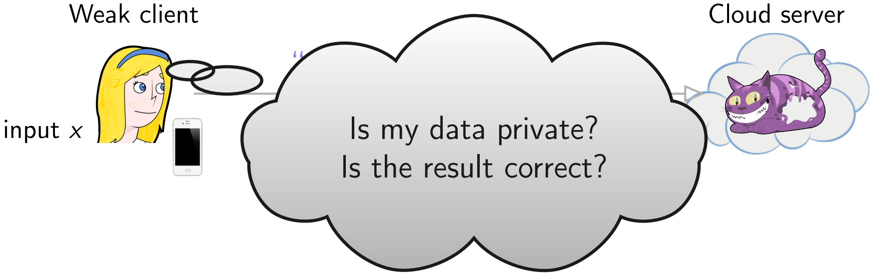
Delegatable computation

- Outsourcing computation of a public function f to a possibly malicious cloud provider (a.k.a. the server)



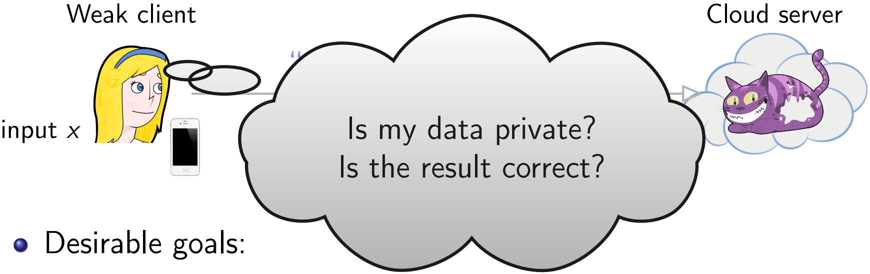
Delegatable computation

- Outsourcing computation of a public function f to a possibly malicious cloud provider (a.k.a. the server)



Delegatable computation

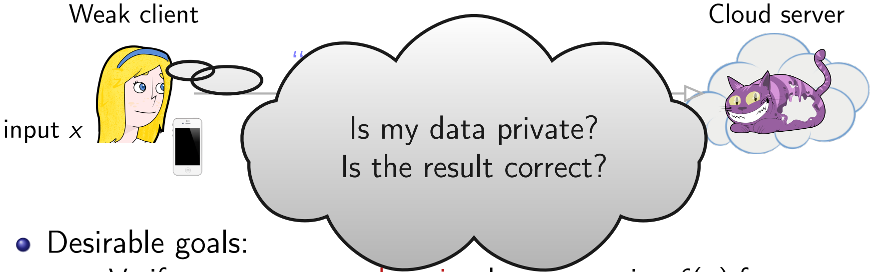
- Outsourcing computation of a public function f to a possibly malicious cloud provider (a.k.a. the server)



- Desirable goals:

Delegatable computation

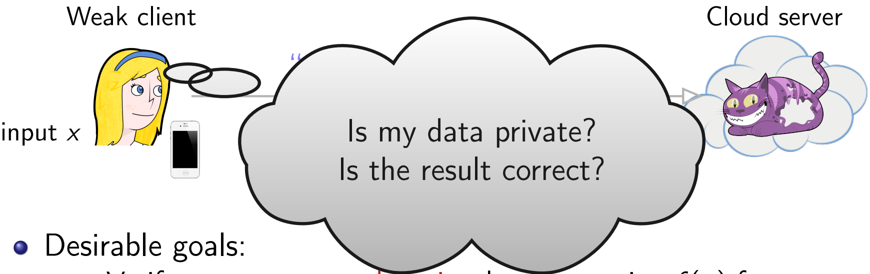
- Outsourcing computation of a public function f to a possibly malicious cloud provider (a.k.a. the server)



- Desirable goals:
  - Verify correctness much easier than computing $f(x)$ from scratch

Delegatable computation

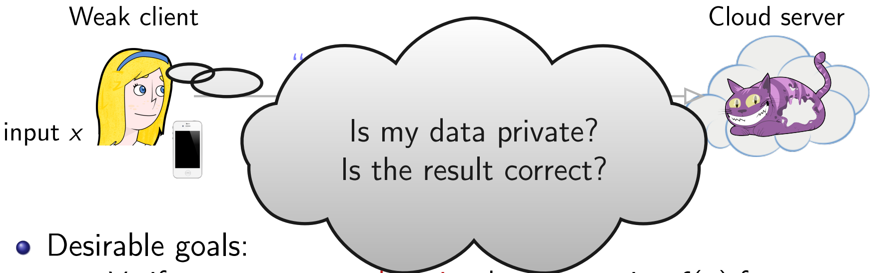
- Outsourcing computation of a public function f to a possibly malicious cloud provider (a.k.a. the server)



- Desirable goals:
  - Verify correctness much easier than computing $f(x)$ from scratch
  - Minimize communication complexity

Delegatable computation

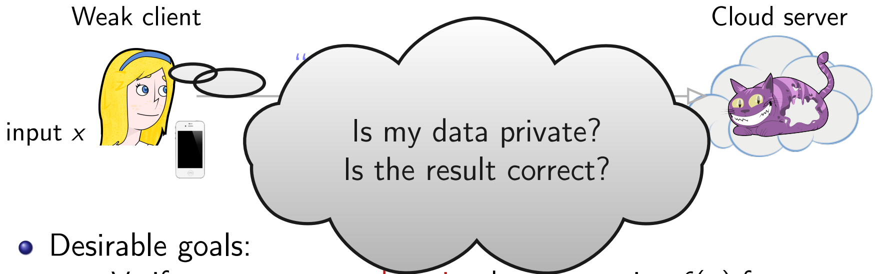
- Outsourcing computation of a public function f to a possibly malicious cloud provider (a.k.a. the server)



- Desirable goals:
  - Verify correctness much easier than computing $f(x)$ from scratch
  - Minimize communication complexity
- Two main lines of work:

Delegatable computation

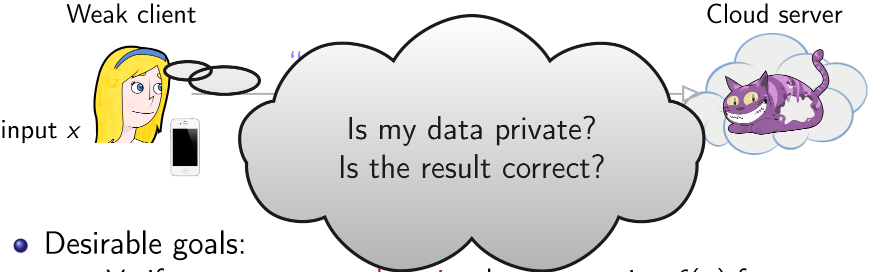
- Outsourcing computation of a public function f to a possibly malicious cloud provider (a.k.a. the server)



- Desirable goals:
  - Verify correctness much easier than computing $f(x)$ from scratch
  - Minimize communication complexity
- Two main lines of work:
  - Protocols for any function (e.g. [GGP10])

Delegatable computation

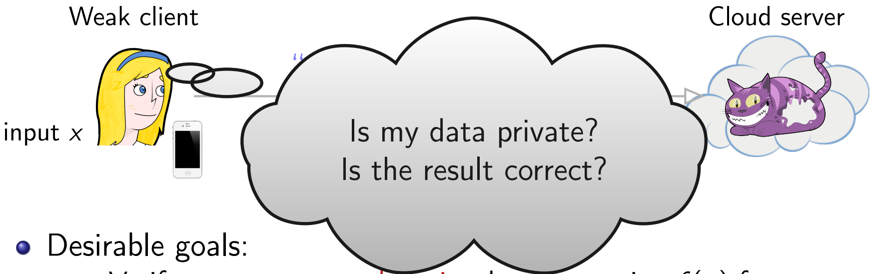
- Outsourcing computation of a public function f to a possibly malicious cloud provider (a.k.a. the server)



- Desirable goals:
  - Verify correctness much easier than computing $f(x)$ from scratch
  - Minimize communication complexity
- Two main lines of work:
  - Protocols for any function (e.g. [GGP10])
  - Protocols for a specific function (e.g. [BGV11])

Delegatable computation

- Outsourcing computation of a public function f to a possibly malicious cloud provider (a.k.a. the server)



- Desirable goals:
  - Verify correctness much easier than computing $f(x)$ from scratch
  - Minimize communication complexity
- Two main lines of work:
  - Protocols for any function (e.g. [GGP10])
  - Protocols for a specific function (e.g. [BGV11])

This talk!

Pattern matching

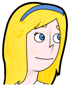
- Input: A text $T \in \Sigma^n$ and a pattern $p \in \Sigma^m$ (e.g., $\Sigma = \{0, 1\}$)

Pattern matching

- Input: A text $T \in \Sigma^n$ and a pattern $p \in \Sigma^m$ (e.g., $\Sigma = \{0, 1\}$)
- Output: The set of positions where p appears in T (if any)

Pattern matching

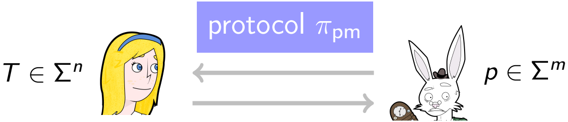
- Input: A text $T \in \Sigma^n$ and a pattern $p \in \Sigma^m$ (e.g., $\Sigma = \{0, 1\}$)
- Output: The set of positions where p appears in T (if any)

$T \in \Sigma^n$ 

 $p \in \Sigma^m$

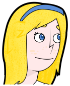
Pattern matching

- Input: A text $T \in \Sigma^n$ and a pattern $p \in \Sigma^m$ (e.g., $\Sigma = \{0, 1\}$)
- Output: The set of positions where p appears in T (if any)



Pattern matching

- Input: A text $T \in \Sigma^n$ and a pattern $p \in \Sigma^m$ (e.g., $\Sigma = \{0, 1\}$)
- Output: The set of positions where p appears in T (if any)

$T \in \Sigma^n$ 

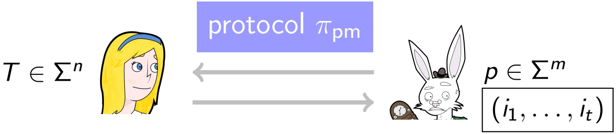
protocol π_{pm}

 $p \in \Sigma^m$

$\boxed{(i_1, \ldots, i_t)}$

Pattern matching

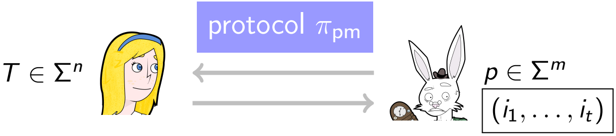
- Input: A text $T \in \Sigma^n$ and a pattern $p \in \Sigma^m$ (e.g., $\Sigma = \{0, 1\}$)
- Output: The set of positions where p appears in T (if any)



-  doesn't lean anything and  doesn't learn anything beyond $(i_1, \ldots, i_t)$

Pattern matching

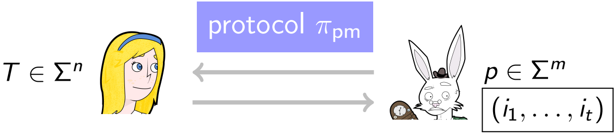
- Input: A text $T \in \Sigma^n$ and a pattern $p \in \Sigma^m$ (e.g., $\Sigma = \{0, 1\}$)
- Output: The set of positions where p appears in T (if any)



-  doesn't lean anything and  doesn't learn anything beyond $(i_1, \dots, i_t)$
- Broad set of applications: text retrieval, music retrieval, computational biology, data mining, network security...

Pattern matching

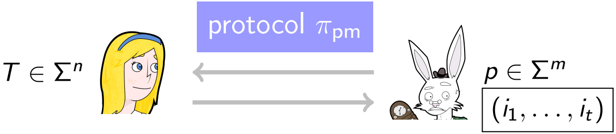
- Input: A text $T \in \Sigma^n$ and a pattern $p \in \Sigma^m$ (e.g., $\Sigma = \{0, 1\}$)
- Output: The set of positions where p appears in T (if any)



-  doesn't lean anything and  doesn't learn anything beyond $(i_1, \ldots, i_t)$
- Broad set of applications: text retrieval, music retrieval, computational biology, data mining, network security...
- Solutions for the 2-party case not applicable to the cloud setting

Pattern matching

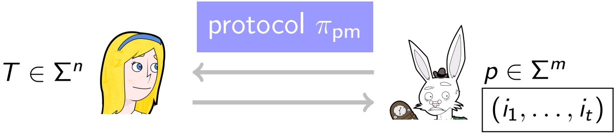
- Input: A text $T \in \Sigma^n$ and a pattern $p \in \Sigma^m$ (e.g., $\Sigma = \{0, 1\}$)
- Output: The set of positions where p appears in T (if any)



-  doesn't lean anything and  doesn't learn anything beyond $(i_1, \ldots, i_t)$
- Broad set of applications: text retrieval, music retrieval, computational biology, data mining, network security...
- Solutions for the 2-party case not applicable to the cloud setting
  - Overhead per search query grows linearly in n

Pattern matching

- Input: A text $T \in \Sigma^n$ and a pattern $p \in \Sigma^m$ (e.g., $\Sigma = \{0, 1\}$)
- Output: The set of positions where p appears in T (if any)



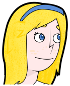
-  doesn't lean anything and  doesn't learn anything beyond $(i_1, \ldots, i_t)$
- Broad set of applications: text retrieval, music retrieval, computational biology, data mining, network security...
- Solutions for the 2-party case not applicable to the cloud setting
  - Overhead per search query grows linearly in n
  - Text holder cannot control the content of the server's responses

Pattern matching in the cloud

1. Setup phase:  outsources an encoding $\tilde{T}$ of T to 

Pattern matching in the cloud

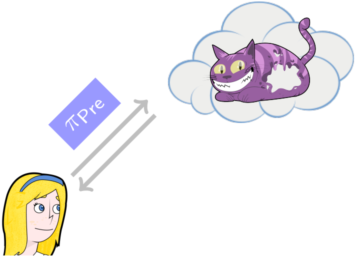
1. Setup phase:  outsources an encoding $\tilde{T}$ of T to 

$T \in \Sigma^n$ 

Pattern matching in the cloud

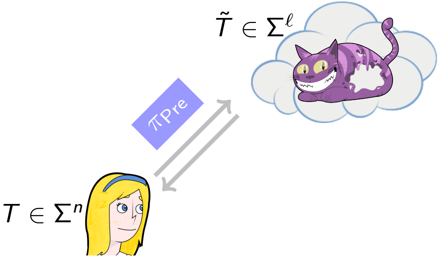
1. Setup phase:  outsources an encoding $\tilde{T}$ of T to 

$T \in \Sigma^n$



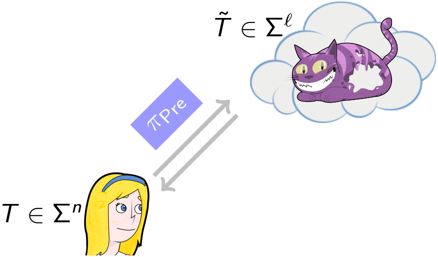
Pattern matching in the cloud

1. Setup phase:  outsources an encoding $\tilde{T}$ of T to 



Pattern matching in the cloud

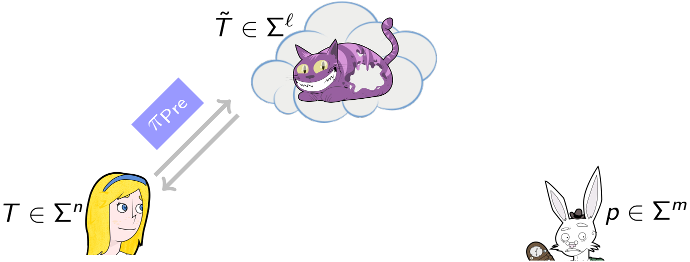
1. Setup phase:  outsources an encoding $\tilde{T}$ of T to 



2. Query phase:  with p interacts with  to get $(i_1, \ldots, i_t)$

Pattern matching in the cloud

1. Setup phase:  outsources an encoding $\tilde{T}$ of T to 



2. Query phase:  with p interacts with  to get $(i_1, \ldots, i_t)$

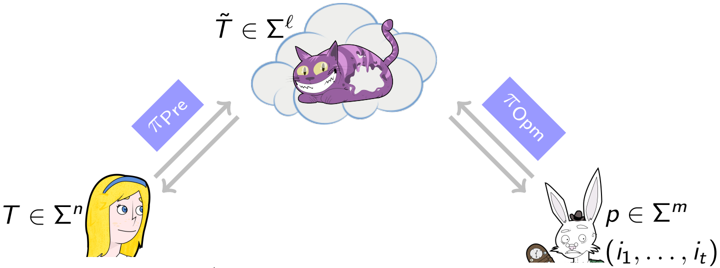
Pattern matching in the cloud

1. Setup phase:  outsources an encoding $\tilde{T}$ of T to 

2. Query phase:  with p interacts with  to get $(i_1, \ldots, i_t)$

Pattern matching in the cloud

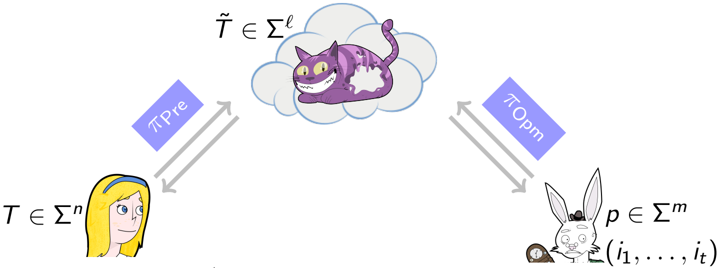
1. Setup phase:  outsources an encoding $\tilde{T}$ of T to 



2. Query phase:  with p interacts with  to get $(i_1, \ldots, i_t)$
   - To avoid disclosure of too much information about T we need to "bind" a search query to p

Pattern matching in the cloud

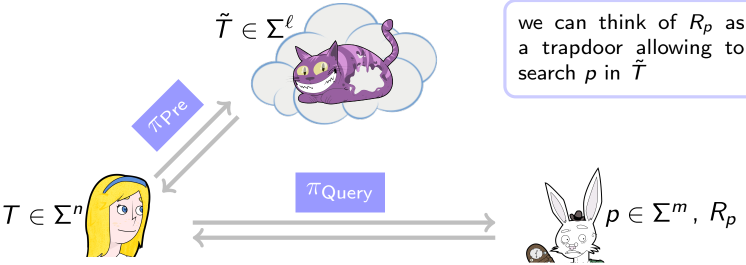
1. Setup phase:  outsources an encoding $\tilde{T}$ of T to 



2. Query phase:  with p interacts with  to get $(i_1, \ldots, i_t)$
   - To avoid disclosure of too much information about T we need to "bind" a search query to p
   - We do so by letting  interact with  before running π_{Opm}

Pattern matching in the cloud

1. Setup phase:  outsources an encoding $\tilde{T}$ of T to 

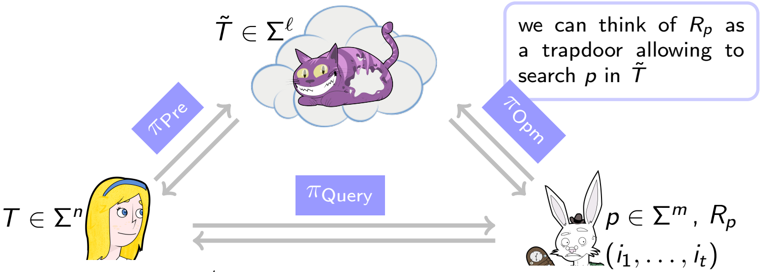


we can think of R_p as a trapdoor allowing to search p in $\tilde{T}$

2. Query phase:  with p interacts with  to get $(i_1, \ldots, i_t)$
   - To avoid disclosure of too much information about T we need to "bind" a search query to p
   - We do so by letting  interact with  before running π_{Opm}

Pattern matching in the cloud

1. Setup phase:  outsources an encoding $\tilde{T}$ of T to 



2. Query phase:  with p interacts with  to get $(i_1, \ldots, i_t)$
   - To avoid disclosure of too much information about T we need to "bind" a search query to p
   - We do so by letting  interact with  before running π_{Opm}

Summary of results

- A precise (simulation-based) security definition for outsourced pattern matching

Summary of results

- A precise (simulation-based) security definition for outsourced pattern matching
- A simple protocol $(\pi_{\mathsf{Pre}}, \pi_{\mathsf{Query}}, \pi_{\mathsf{Opm}})$ with passive security based on the subset sum problem

Summary of results

- A precise (simulation-based) security definition for outsourced pattern matching
- A simple protocol $(\pi_{\mathsf{Pre}}, \pi_{\mathsf{Query}}, \pi_{\mathsf{Opm}})$ with passive security based on the subset sum problem
  - Amortized complexity: Communication/computation linear in n for π_{Pre}, but linear in m during π_{Opm} (optimal)

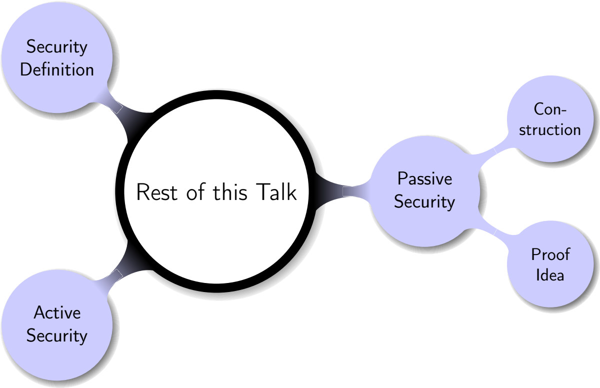
Summary of results

- A precise (simulation-based) security definition for outsourced pattern matching
- A simple protocol $(\pi_{\mathsf{Pre}}, \pi_{\mathsf{Query}}, \pi_{\mathsf{Opm}})$ with passive security based on the subset sum problem
  - Amortized complexity: Communication/computation linear in n for π_{Pre}, but linear in m during π_{Opm} (optimal)
  - The server is allowed to learn the matched positions for each query (this seems necessary if we want sublinear communication in the query phase)

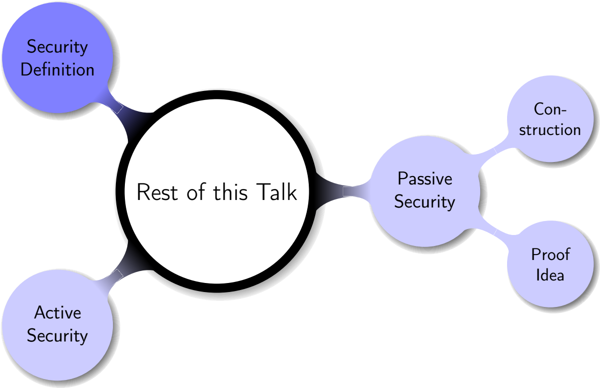
Summary of results

- A precise (simulation-based) security definition for outsourced pattern matching
- A simple protocol $(\pi_{\mathsf{Pre}}, \pi_{\mathsf{Query}}, \pi_{\mathsf{Opm}})$ with passive security based on the subset sum problem
  - Amortized complexity: Communication/computation linear in n for π_{Pre}, but linear in m during π_{Opm} (optimal)
  - The server is allowed to learn the matched positions for each query (this seems necessary if we want sublinear communication in the query phase)
- An extension achieving active security (see the paper)

Roadmap


Security
Definition
Rest of this Talk
Passive
Security
Con-
struction
Proof
Idea
Active
Security

Roadmap


Security
Definition
Rest of this Talk
Passive
Security
Con-
struction
Proof
Idea
Active
Security

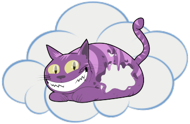
Security of outsourced pattern matching

- We define security following the real/ideal world paradigm

Security of outsourced pattern matching

- We define security following the real/ideal world paradigm

ideal world

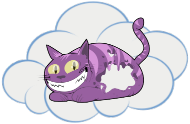


$T \in \Sigma^n$

Security of outsourced pattern matching

- We define security following the real/ideal world paradigm

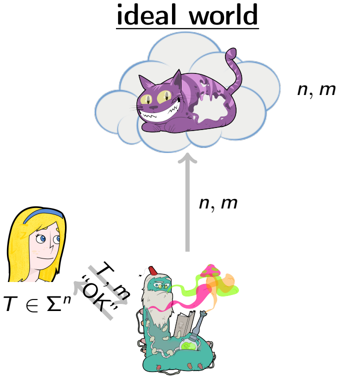
ideal world



$T \in \Sigma^n$

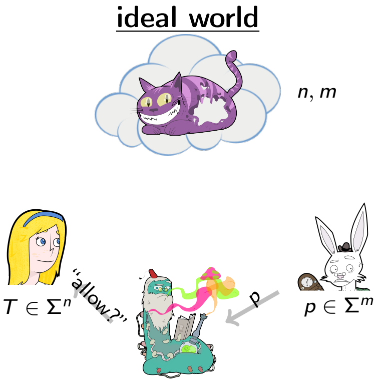
Security of outsourced pattern matching

- We define security following the real/ideal world paradigm



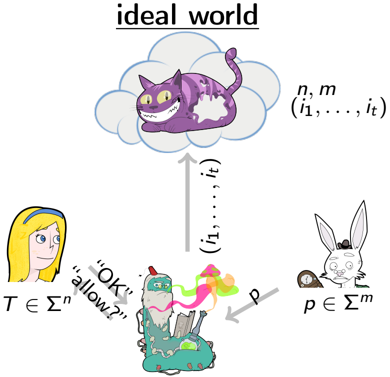
Security of outsourced pattern matching

- We define security following the real/ideal world paradigm



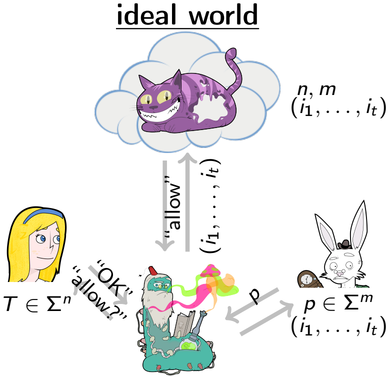
Security of outsourced pattern matching

- We define security following the real/ideal world paradigm



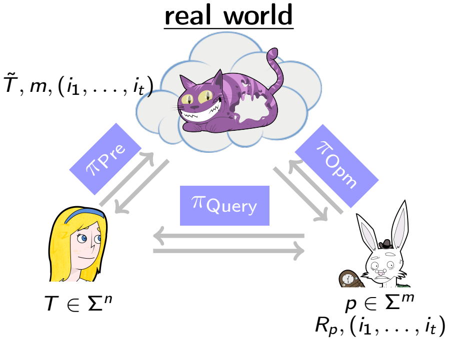
Security of outsourced pattern matching

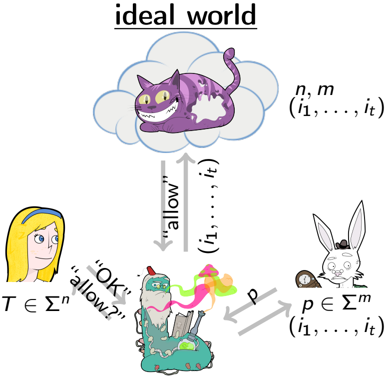
- We define security following the real/ideal world paradigm



Security of outsourced pattern matching

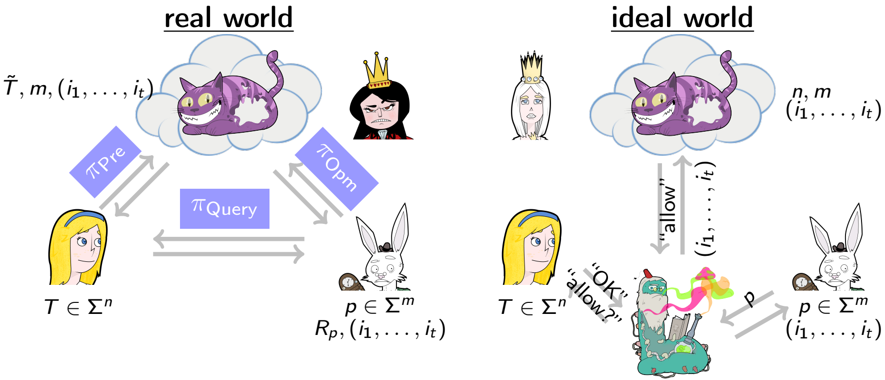
- We define security following the real/ideal world paradigm





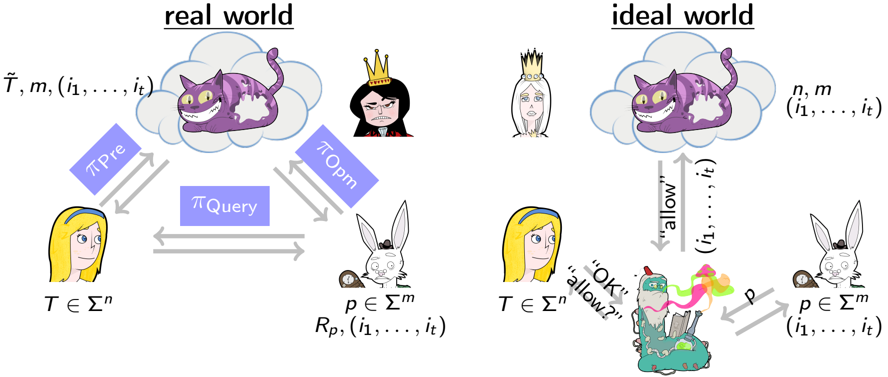
Security of outsourced pattern matching

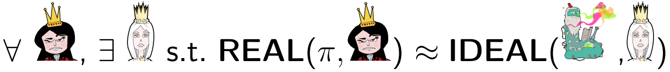
- We define security following the real/ideal world paradigm



Security of outsourced pattern matching

- We define security following the real/ideal world paradigm



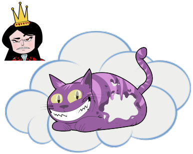


Technical challenges in the simulation

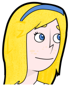
- Let's look at the case of a passively corrupted server

Technical challenges in the simulation

- Let's look at the case of a passively corrupted server

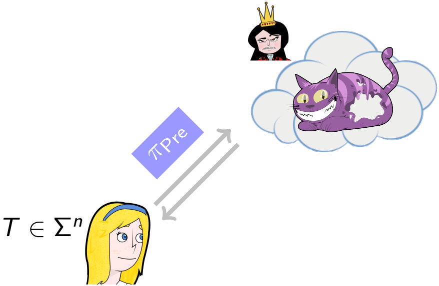


$T \in \Sigma^n$



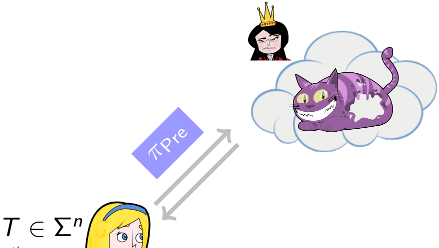
Technical challenges in the simulation

- Let's look at the case of a passively corrupted server



Technical challenges in the simulation

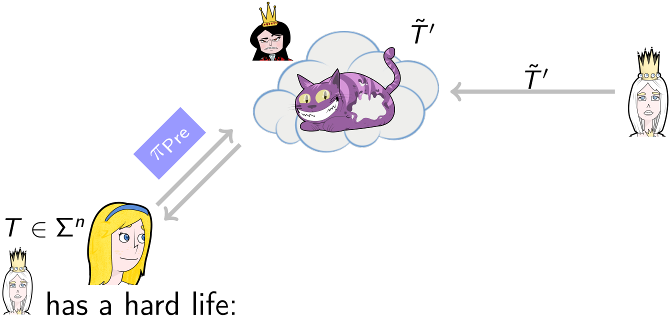
- Let's look at the case of a passively corrupted server



-  has a hard life:

Technical challenges in the simulation

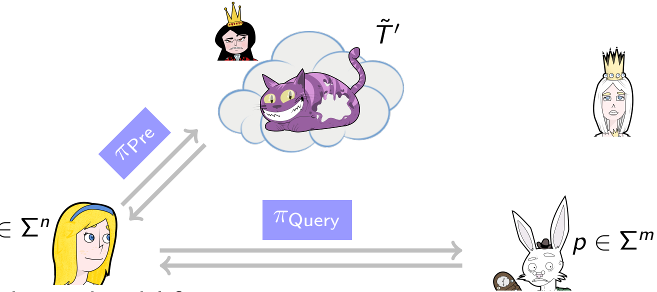
- Let's look at the case of a passively corrupted server



- has a hard life:
  - It has to simulate $\tilde{T}'$ as output of π_{Pre}

Technical challenges in the simulation

- Let's look at the case of a passively corrupted server

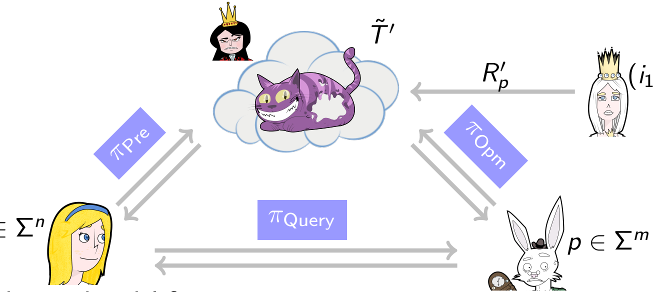


-  has a hard life:
  - It has to simulate $\tilde{T}'$ as output of π_{Pre}
  - When later p is searched, it gets to know $(i_1, \ldots, i_t)$ and has to cook up a consistent R'_p such that

$$(\tilde{T}', R'_p) \approx (\tilde{T}, R_p)$$

Technical challenges in the simulation

- Let's look at the case of a passively corrupted server

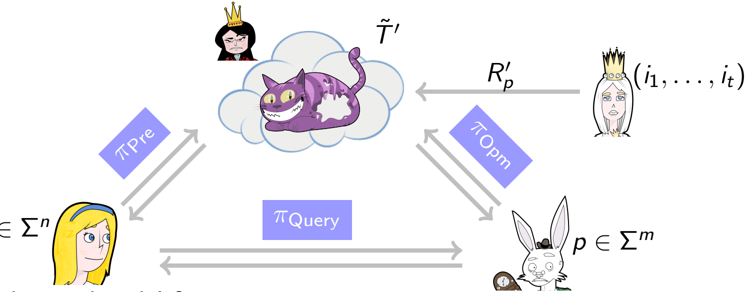


-  has a hard life:
  - It has to simulate $\tilde{T}'$ as output of π_{Pre}
  - When later p is searched, it gets to know $(i_1, \ldots, i_t)$ and has to cook up a consistent R'_p such that

$$(\tilde{T}', R'_p) \approx (\tilde{T}, R_p)$$

Technical challenges in the simulation

- Let's look at the case of a passively corrupted server

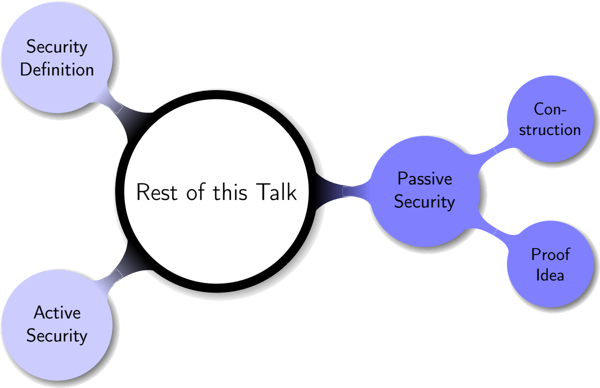


-  has a hard life:
  - It has to simulate $\tilde{T}'$ as output of π_{Pre}
  - When later p is searched, it gets to know $(i_1, \ldots, i_t)$ and has to cook up a consistent R'_p such that

$$(\tilde{T}', R'_p) \approx (\tilde{T}, R_p)$$

  - However p was not known when $\tilde{T}'$ has been computed!

Roadmap



Security Definition
Rest of this Talk
Passive Security
Con-struction
Proof Idea
Active Security

Subset sum

- Let ℓ and M be integers

Subset sum

- Let ℓ and M be integers

$$\underbrace{a_1 \; \ldots \; a_\ell}_{\in \mathbb{Z}_M^\ell}$$

Subset sum

- Let ℓ and M be integers

$$\underbrace{a_1 \; \ldots \; a_\ell}_{\in \mathbb{Z}_M^\ell} \cdot \begin{matrix} s_1 \\ \vdots \\ \underbrace{s_\ell}_{\in\{0,1\}} \end{matrix}$$

Subset sum

- Let ℓ and M be integers

$$\underbrace{\boxed{a_1} \ldots \boxed{a_\ell}}_{\in \mathbb{Z}_M^\ell} \cdot \underbrace{\begin{matrix} \boxed{s_1} \\ \vdots \\ \boxed{s_\ell} \end{matrix}}_{\in \{0,1\}} \mod M = R$$

Subset sum

- Let ℓ and M be integers

$$\underbrace{a_1 \ldots a_\ell}_{\in \mathbb{Z}_M^\ell} \cdot \underbrace{\begin{matrix} s_1 \\ \vdots \\ s_\ell \end{matrix}}_{\in\{0,1\}} \bmod M = R$$

Goal

given $(R, a_1, \ldots, a_\ell)$, find $(s_1, \ldots, s_\ell)$

$s_i = 1$ means a_i contributes to the summation

Subset sum

- Let ℓ and M be integers

$$\underbrace{a_1 \; \dots \; a_\ell}_{\in \mathbb{Z}_M^\ell} \cdot \underbrace{\begin{matrix} s_1 \\ \vdots \\ s_\ell \end{matrix}}_{\in\{0,1\}} \bmod M = R$$

Goal

given $(R, a_1, \dots, a_\ell)$, find $(s_1, \dots, s_\ell)$

$s_i = 1$ means a_i contributes to the summation

- Observation: For random s, a the probability that a random s' shares the same R with s is $\leq 2^\ell / M$

Subset sum

- Let ℓ and M be integers

$$\underbrace{a_1 \ldots a_\ell}_{\in \mathbb{Z}_M^\ell} \cdot \underbrace{\begin{pmatrix} s_1 \\ \vdots \\ s_\ell \end{pmatrix}}_{\in\{0,1\}} \mod M = R$$

Goal

given $(R, a_1, \ldots, a_\ell)$, find $(s_1, \ldots, s_\ell)$

$s_i = 1$ means a_i contributes to the summation

- Observation: For random s, a the probability that a random s' shares the same R with s is $\leq 2^\ell / M$
- Hardness of subset sum as a function of $\Delta = \ell / \log M$

Subset sum

- Let ℓ and M be integers

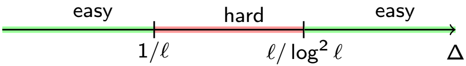
$$\underbrace{a_1 \ \dots \ a_\ell}_{\in \mathbb{Z}_M^\ell} \cdot \underbrace{\begin{matrix} s_1 \\ \vdots \\ s_\ell \end{matrix}}_{\in\{0,1\}} \mod M = R$$

Goal

given $(R, a_1, \dots, a_\ell)$, find $(s_1, \dots, s_\ell)$

$s_i = 1$ means a_i contributes to the summation

- Observation: For random s, a the probability that a random s' shares the same R with s is $\leq 2^\ell / M$
- Hardness of subset sum as a function of $\Delta = \ell / \log M$



The basic idea...

- The π_{Pre} protocol:

The basic idea...

- The π_{Pre} protocol:

$$T = \boxed{0\ 0}\ \underbrace{\boxed{1\ 1\ 0}}_{\substack{p \in \{0,1\}^3 \\ (t\ \text{matches})}}\ \boxed{0\ 1\ 1\ 0}$$

The basic idea...

- The π_{Pre} protocol:

$$T = \boxed{0\ 0}\ \underbrace{\boxed{1\ 1\ 0}}_{\substack{p\in\{0,1\}^3 \\ (t \text{ matches})}}\ \boxed{0\ 1\ 1\ 0}$$

$\forall p \subseteq T$

choose random R_p, $(a_1, \ldots, a_t)$
s.t. $\sum_{i=1}^{t} a_i = R_p \bmod M$

The basic idea...

- The π_{Pre} protocol:

$$T = 0\ 0\ \underbrace{1\ 1\ 0}_{\substack{p \in \{0,1\}^3 \\ (t \text{ matches})}}\ 0\ 1\ 1\ 0 \Rightarrow \tilde{T} = a_1 \ \ldots\ a_\ell \quad (\ell = n - m + 1)$$

$\forall p \subseteq T$

choose random R_p, $(a_1, \ldots, a_t)$
s.t. $\sum_{i=1}^{t} a_i = R_p \bmod M$

The basic idea...

- The π_{Pre} protocol:

$$T = \boxed{0\ \ 0}\ \underbrace{\boxed{1\ \ 1\ \ 0}}_{\substack{p \in \{0,1\}^3 \\ (t \text{ matches})}}\ \boxed{0\ \ 1\ \ 1\ \ 0} \Rightarrow \tilde{T} = \boxed{a_1}\ \ldots\ \boxed{a_\ell}\ (\ell = n - m + 1)$$

$\forall p \subseteq T$

choose random R_p, $(a_1, \ldots, a_t)$
s.t. $\sum_{i=1}^{t} a_i = R_p \bmod M$

(In practice compute $R_p = f(\kappa, p)$ for PRF f and store only κ.)

The basic idea...

- The π_{Pre} protocol:

$$T = \boxed{0\ \ 0}\ \underbrace{\boxed{1\ \ 1\ \ 0}}_{\substack{p \in \{0,1\}^3 \\ (t \text{ matches})}}\ \boxed{0\ \ 1\ \ 1\ \ 0} \Rightarrow \tilde{T} = \boxed{a_1}\ \ldots\ \boxed{a_\ell}\ (\ell = n - m + 1)$$

$\forall p \subseteq T$

choose random R_p, $(a_1, \ldots, a_t)$
s.t. $\sum_{i=1}^{t} a_i = R_p \bmod M$

(In practice compute $R_p = f(\kappa, p)$ for PRF f and store only κ.)

- Protocol π_{Query}: Any two-party protocol for oblivious evaluation of $f(\kappa, p)$

The basic idea...

- The π_{Pre} protocol:

$$T = \boxed{0\ \ 0}\ \underbrace{\boxed{1\ \ 1\ \ 0}}_{\substack{p \in \{0,1\}^3 \\ (t \text{ matches})}}\ \boxed{0\ \ 1\ \ 1\ \ 0} \Rightarrow \tilde{T} = \boxed{a_1}\ \ldots\ \boxed{a_\ell}\ \ (\ell = n - m + 1)$$

$\forall p \subseteq T$

choose random R_p, $(a_1, \ldots, a_t)$
s.t. $\sum_{i=1}^{t} a_i = R_p \bmod M$

(In practice compute $R_p = f(\kappa, p)$ for PRF f and store only κ.)

- Protocol π_{Query}: Any two-party protocol for oblivious evaluation of $f(\kappa, p)$

- Protocol π_{Opm}:  gives R_p to  and the latter solves subset sum instance $(R_p, \tilde{T})$

… and its limitation

- The above simple protocol can be proven secure, but suffers from two limitations:

… and its limitation

- The above simple protocol can be proven secure, but suffers from two limitations:
  - Communication complexity is $O(n^2 + \lambda n)$ in the setup phase and proportional to n in the query phase (λ is security parameter)

… and its limitation

- The above simple protocol can be proven secure, but suffers from two limitations:
  - Communication complexity is $O(n^2 + \lambda n)$ in the setup phase and proportional to n in the query phase (λ is security parameter)
  - Limited use, as the text has to be very short

… and its limitation

- The above simple protocol can be proven secure, but suffers from two limitations:
  - Communication complexity is $O(n^2 + \lambda n)$ in the setup phase and proportional to n in the query phase (λ is security parameter)
  - Limited use, as the text has to be very short
    - To keep the collision probability ($= 2^\ell / M$) low we shall set $M = 2^{\lambda+n}$

… and its limitation

- The above simple protocol can be proven secure, but suffers from two limitations:
  - Communication complexity is $O(n^2 + \lambda n)$ in the setup phase and proportional to n in the query phase (λ is security parameter)
  - Limited use, as the text has to be very short
    - To keep the collision probability ($= 2^\ell / M$) low we shall set $M = 2^{\lambda + n}$
    - This yields $\ell < \sqrt{\lambda}$ if we want the subset sum problem to be solvable in polynomial time

… and its limitation

- The above simple protocol can be proven secure, but suffers from two limitations:
  - Communication complexity is $O(n^2 + \lambda n)$ in the setup phase and proportional to n in the query phase (λ is security parameter)
  - Limited use, as the text has to be very short
    - To keep the collision probability ($= 2^\ell/M$) low we shall set $M = 2^{\lambda+n}$
    - This yields $\ell < \sqrt{\lambda}$ if we want the subset sum problem to be solvable in polynomial time
    - Even $\lambda = 10^4$ (i.e., subset sum elements of size $\approx$ 10 KByte) allows to process texts of less than 100 bits

… and its limitation

- The above simple protocol can be proven secure, but suffers from two limitations:
  - Communication complexity is $O(n^2 + \lambda n)$ in the setup phase and proportional to n in the query phase (λ is security parameter)
  - Limited use, as the text has to be very short
    - To keep the collision probability $(= 2^\ell / M)$ low we shall set $M = 2^{\lambda + n}$
    - This yields $\ell < \sqrt{\lambda}$ if we want the subset sum problem to be solvable in polynomial time
    - Even $\lambda = 10^4$ (i.e., subset sum elements of size $\approx$ 10 KByte) allows to process texts of less than 100 bits
- To overcome the above problems, we define an extension of the previous solution based on packaging

Packaging

- Divide the original text T into blocks of length $2m$ (overlapping in the last m bits)

Packaging

- Divide the original text T into blocks of length $2m$ (overlapping in the last m bits)

$$T = \boxed{001100110}$$

Packaging

- Divide the original text T into blocks of length $2m$ (overlapping in the last m bits)

$$B_1 = \boxed{0011}$$

$$T = \boxed{001100110} \qquad B_2 = \boxed{1100}$$

$$B_3 = \cdots$$

Packaging

- Divide the original text T into blocks of length $2m$ (overlapping in the last m bits)

$$B_1 = \boxed{0011}$$

$$T = \boxed{001100110} \qquad B_2 = \boxed{1100}$$

$$B_3 = \cdots$$

- Process each of the blocks separately as before

Packaging

- Divide the original text T into blocks of length $2m$ (overlapping in the last m bits)

$$T = \boxed{001100110} \qquad \begin{array}{l} B_1 = 0011 \\ B_2 = 1100 \\ B_3 = \cdots \end{array}$$

- Process each of the blocks separately as before
  - Avoid using in each block the same trapdoor for some pattern p

Packaging

- Divide the original text T into blocks of length $2m$ (overlapping in the last m bits)

$$B_1 = 0011$$

$$T = 001100110 \qquad B_2 = 1100$$

$$B_3 = \cdots$$

- Process each of the blocks separately as before
  - Avoid using in each block the same trapdoor for some pattern p
  - To do so we encode $R_p = \mathcal{H}(f(\kappa, p)||b)$ where $\mathcal{H}$ is a random oracle and b is the block number

Packaging

- Divide the original text T into blocks of length $2m$ (overlapping in the last m bits)

Now $\ell < \sqrt{\lambda} \Rightarrow m \approx \sqrt{\lambda}$ which upper bounds the length of the pattern (not of the text)

$$T = 001100110$$

$$B_1 = 0011$$

$$B_2 = 1100$$

$$B_3 = \cdots$$

- Process each of the blocks separately as before
  - Avoid using in each block the same trapdoor for some pattern p
  - To do so we encode $R_p = \mathcal{H}(f(\kappa, p)||b)$ where $\mathcal{H}$ is a random oracle and b is the block number

Packaging

- Divide the original text T into blocks of length $2m$ (overlapping in the last m bits)

Now $\ell < \sqrt{\lambda} \Rightarrow m \approx \sqrt{\lambda}$ which upper bounds the length of the pattern (not of the text)

$T = 001100110$

$B_1 = 0011$

$B_2 = 1100$

$B_3 = \cdots$

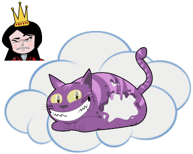
- Process each of the blocks separately as before
  - Avoid using in each block the same trapdoor for some pattern p
  - To do so we encode $R_p = \mathcal{H}(f(\kappa, p)||b)$ where $\mathcal{H}$ is a random oracle and b is the block number
- The communication complexity is $O(mn + \lambda n)$ in the setup phase and $O(\lambda m)$ in the query phase

How the simulator works

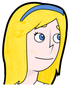
- Let's look again at the case of a passively corrupted server

How the simulator works

- Let's look again at the case of a passively corrupted server



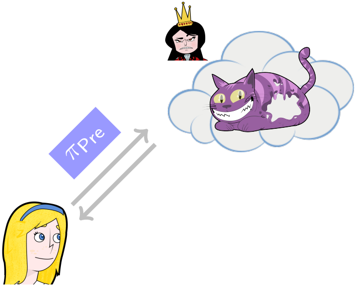
$T \in \Sigma^n$



How the simulator works

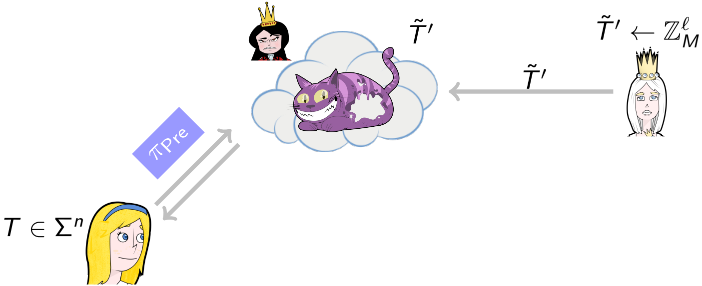
- Let's look again at the case of a passively corrupted server

$T \in \Sigma^n$



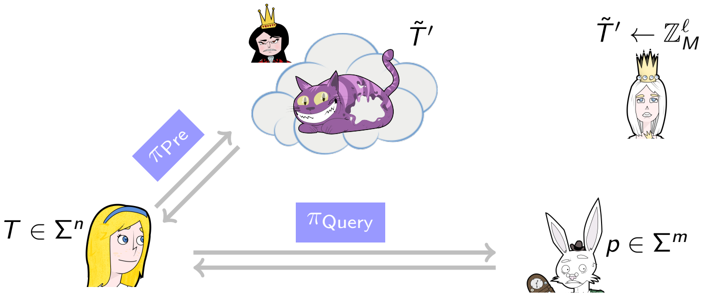
How the simulator works

- Let's look again at the case of a passively corrupted server



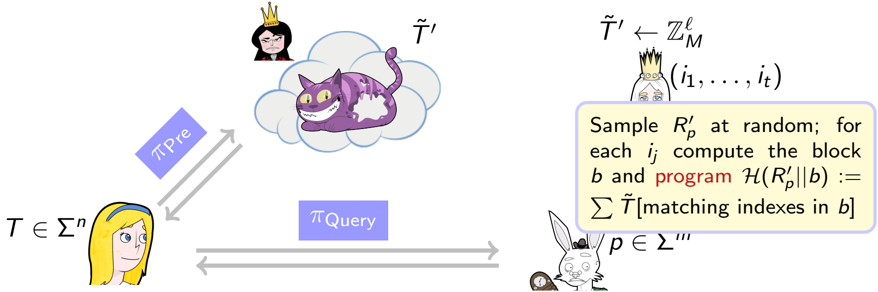
How the simulator works

- Let's look again at the case of a passively corrupted server



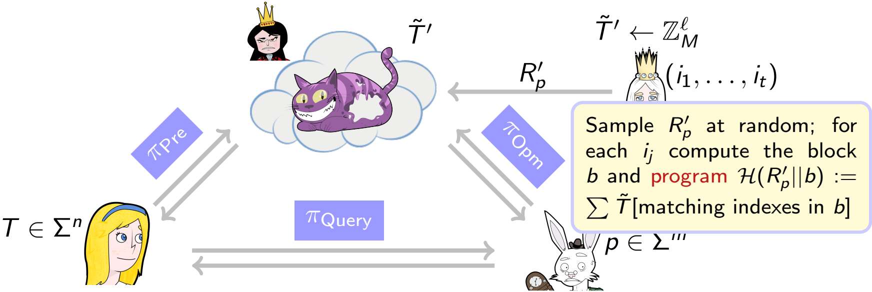
How the simulator works

- Let's look again at the case of a passively corrupted server



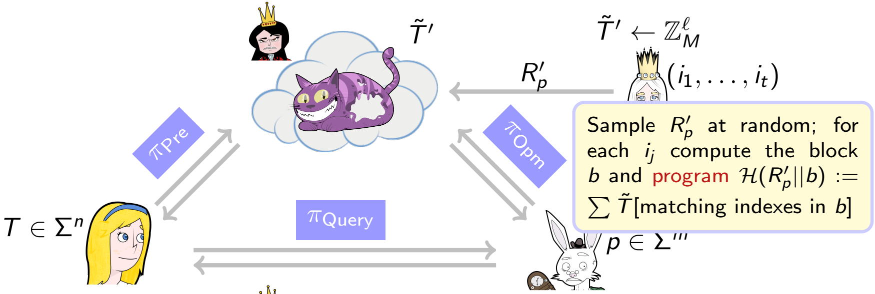
How the simulator works

- Let's look again at the case of a passively corrupted server



How the simulator works

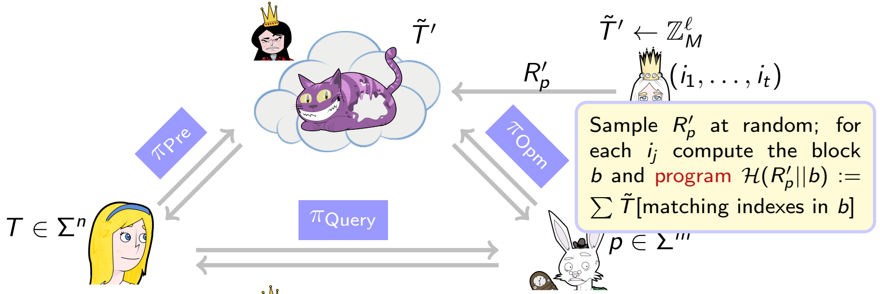
- Let's look again at the case of a passively corrupted server



- Define **HYB**(π, ) to be the same as the real distribution but such that R_p is random

How the simulator works

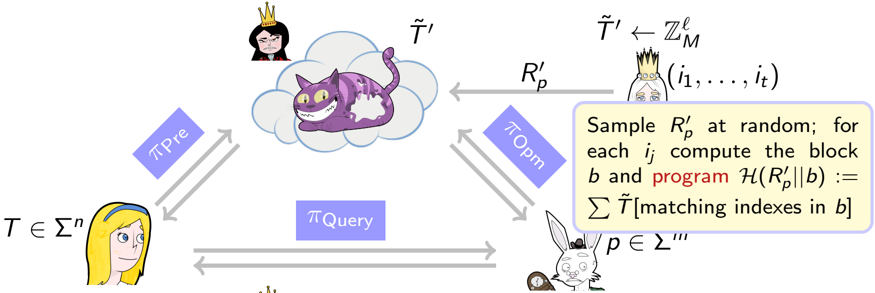
- Let's look again at the case of a passively corrupted server



- Define **HYB**(π, ) to be the same as the real distribution but such that R_p is random
  - **HYB**(π, ) $\approx_c$ **REAL**(π, ) (by security of PRF)

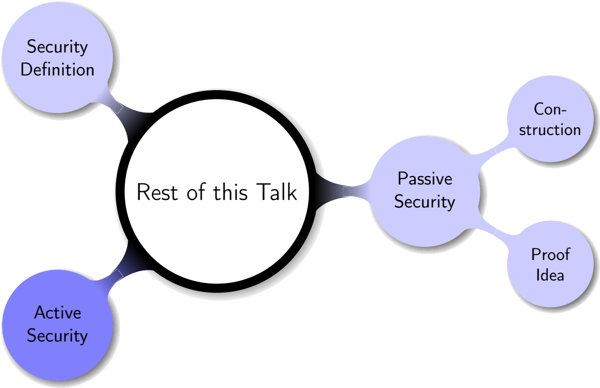
How the simulator works

- Let's look again at the case of a passively corrupted server



- Define **HYB**(π, ) to be the same as the real distribution but such that R_p is random
  - **HYB**(π, ) $\approx_c$ **REAL**(π, ) (by security of PRF)
  - **HYB**(π, ) $\approx_s$ **IDEAL**(, ) (programming can fail)

Roadmap


Security Definition
Rest of this Talk
Passive Security
Con- struction
Proof Idea
Active Security

Active security

- What about active corruption?

Active security

- What about active corruption?
- We need to verify correctness of 's answers

Active security

- What about active corruption?
- We need to verify correctness of 's answers
  - To do so we let  outsource a succinct commitment to $\tilde{T}$ and ask the server to open the values for the matching locations

Active security

- What about active corruption?
- We need to verify correctness of 's answers
  - To do so we let  outsource a succinct commitment to $\tilde{T}$ and ask the server to open the values for the matching locations
  - Note that  could still cheat by always declaring a "no match" (we avoid this via zero-knowledge sets)

Active security

- What about active corruption?
- We need to verify correctness of 's answers
  - To do so we let  outsource a succinct commitment to $\tilde{T}$ and ask the server to open the values for the matching locations
  - Note that  could still cheat by always declaring a "no match" (we avoid this via zero-knowledge sets)
- We need to ensure  can search only p's for which it has a trapdoor

Active security

- What about active corruption?
- We need to verify correctness of  's answers
  - To do so we let  outsource a succinct commitment to $\tilde{T}$ and ask the server to open the values for the matching locations
  - Note that  could still cheat by always declaring a "no match" (we avoid this via zero-knowledge sets)
- We need to ensure  can search only p's for which it has a trapdoor
  - For this π_{Query} must have active security

Active security

- What about active corruption?
- We need to verify correctness of 's answers
  - To do so we let  outsource a succinct commitment to $\tilde{T}$ and ask the server to open the values for the matching locations
  - Note that  could still cheat by always declaring a "no match" (we avoid this via zero-knowledge sets)
- We need to ensure  can search only p's for which it has a trapdoor
  - For this π_{Query} must have active security
- We need to ensure that the text $\tilde{T}$ computed by  has associated a well defined text T

Active security

- What about active corruption?
- We need to verify correctness of 's answers
  - To do so we let  outsource a succinct commitment to $\tilde{T}$ and ask the server to open the values for the matching locations
  - Note that  could still cheat by always declaring a "no match" (we avoid this via zero-knowledge sets)
- We need to ensure  can search only p's for which it has a trapdoor
  - For this π_{Query} must have active security
- We need to ensure that the text $\tilde{T}$ computed by  has associated a well defined text T
  - This requires expensive cut-and-choose techniques, which we avoid by a smart "on-the-fly" verification trick

Take-home message

- We give a simulation-based security definition for outsourced pattern matching

Take-home message

- We give a simulation-based security definition for outsourced pattern matching
- We construct a protocol with passive security (in the RO model) and sublinear communication complexity in the query phase (which is optimal)

Take-home message

- We give a simulation-based security definition for outsourced pattern matching
- We construct a protocol with passive security (in the RO model) and sublinear communication complexity in the query phase (which is optimal)
- We explain how to modify the basic protocol to tolerate active adversaries

Take-home message

- We give a simulation-based security definition for outsourced pattern matching
- We construct a protocol with passive security (in the RO model) and sublinear communication complexity in the query phase (which is optimal)
- We explain how to modify the basic protocol to tolerate active adversaries
- Open problems for future work:

Take-home message

- We give a simulation-based security definition for outsourced pattern matching
- We construct a protocol with passive security (in the RO model) and sublinear communication complexity in the query phase (which is optimal)
- We explain how to modify the basic protocol to tolerate active adversaries
- Open problems for future work:
  - An efficient construction in the standard model

Take-home message

- We give a simulation-based security definition for outsourced pattern matching
- We construct a protocol with passive security (in the RO model) and sublinear communication complexity in the query phase (which is optimal)
- We explain how to modify the basic protocol to tolerate active adversaries
- Open problems for future work:
  - An efficient construction in the standard model
  - Extensions (pattern matching with wildcards, approximate pattern matching, hiding the length of the text/pattern)

Thank you!

entirely written in $\LaTeX 2_\varepsilon$ using Beamer and Tik Z, drawings by Andrea Chronopoulos