Lecture: IT-Security

Anonymization & Secure Multi-Party-Computation

Markus Krämer

kraemerm@cs.uni-bonn.de

University of Bonn | Institute of Computer Science 4

| Bonn | 16. Januar 2025

UNIVERSITÄT BONN

About me

2013-2017	Bachelor Computer Science in Bonn
2017-2020	Master Computer Science in Bonn
2016-2020	Tutor „Logik und diskrete Strukturen“
	Tutor „Reaktive Sicherheir
	Tutor „IT-Sicherheit“
Since 2020	Researcher WG IT-Security
	Project „WACHMANN“
	Project „BNTrAInee“
	Project „DARIA“
	Project „PEKI“

Privacy

Definition The right of individuals to protect their personal lifes & matters from the outside world and to determine which information about itself should be known to others

(Westin & Kasem-Madani)

UNIVERSITÄT BONN

Personal Data

Definition Any information relating to an identified or identifiable (Name, Address, ...) natural person

Principles of personal data processing

- Lawfulness, fairness and transparency
- Purpose limitation
- Data minimization
- Accuracy of processed data
- Storage limitation
- Integrity and confidentiality

Attribute Types

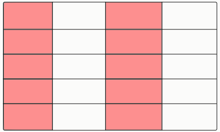
(Direct) Identifiers

- Identify an in individual directly
- Examples
    - Name
    - Address
    - Identity number

Quasi-Identifiers

- Population U
- Table T with attributes A_i: $T(A_1, \dots, A_n)$
- Forward function: $f_v : U \rightarrow T$
- Backward funcion: $f_r : T \rightarrow U', U' \subseteq U$
- **Definition**

  A Quasi-Identifier Q_T is a subset of attributes $(A_j, \dots, A_k) \subseteq (A_1, \dots, A_n)$, which are sufficient to identify an individual: $f_r(f_v(p_i[Q_T])) = p_i$



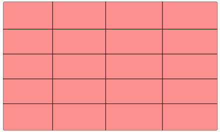
Sensitive Attributes

- Attributes whose values are considered sensitive somehow and are worth to be protected
- Examples:
  - Salary
  - Disease
  - ...

Privacy Threats

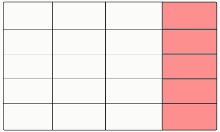
Membership disclosure

- The attacker gains information about whether data concerning an individual is contained in a data set
- Means *not* necessarily leakage of more information (e.g. sensitive attributes)
- Threat relies on leakage of meta information
  - E.g. data set is about cancer patients



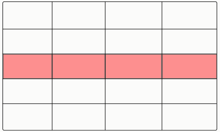
Attribute disclosure

- Leakage of sensitive attributes
- May cause discomfort or harm to affected individuals
  - Illnesses or political convictions
  - Attribute disclosure is possible without matching an individual to a record (We will see this later)



Identity disclosure

- The attacker is able to match an individual to a record of the data set
- **Result**: Leakage of all attribute values



Anonymization: Attacker Models

- **Prosecutor**
  - Intends to gain information about a specific individual
  - Assumption: Knows that the individual is contained in the dataset
- **Journalist**
  - Intends to gain information about a specific individual
  - Assumption: No background knowledge whether the individual is contained in the dataset
- **Marketer**
  - Intends to re-identify a large number of individuals
  - Constraint: An Attack is only considered successful if a larger fraction of the individuals is re-identified

Privacy enhancing technologies

UNIVERSITÄT BONN

Pseudonymization

- Personal data may no longer be attributed to individuals
  - without the use of additional information
  - which must be kept separately
  - where technical or organizational measures ensure that no reidentification is possible
- Effort for reidentification is relatively high

Anonymization

- „information which does not relate to an identified or identifiable natural person“
- „personal data rendered anonymous in such a manner that the data subject is not or no longer identifiable“
- Effort for reidentification is excessively high
- Anonymous data is not personal data anymore, thus the GDPR needs not to be accounted

Anonymization

Anonymization

Anonymization techniques

Masking Methods

- Perturbative Methods
- Non-Perturbative Methods
- Synthetic data

Perturbative Methods

- Alter the attribute values in a way that the new data set may contain erroneous information
- E.g. add noise following a normal distribution
- Age = 23, Noise = 4 $\rightarrow$ Age = 27

Non-Perturbative Methods

- Replace attribute values with less specific than incorrect values
- Detail reduction
- E.g. Interval, statistical values (mean, ...)
- Age = 23 $\longrightarrow$ Age = [20-25]

Synthetic data

- Replace attribute values by artificially created values
- May be based on a model of the real data

Masking Methods

- Perturbative Methods
- Non-Perturbative Methods
- Synthetic data

Masking Methods

- **Perturbative Methods**
- Non-Perturbative Methods
- Synthetic data

Perturbative: Microaggregation

- Replace a group of values with a summary statistic, e.g. mean
- Clustering possible (replace values by centroid)
- Univariate: Microaggregation per attribute (no k-anonymity ensured)
- Multivariate: Microaggregation for multiple (all) attributes at once
  - Ensures k-anonymity for unpartitioned data if all attributes were considered in the process
- For numerical data types

Masking Methods

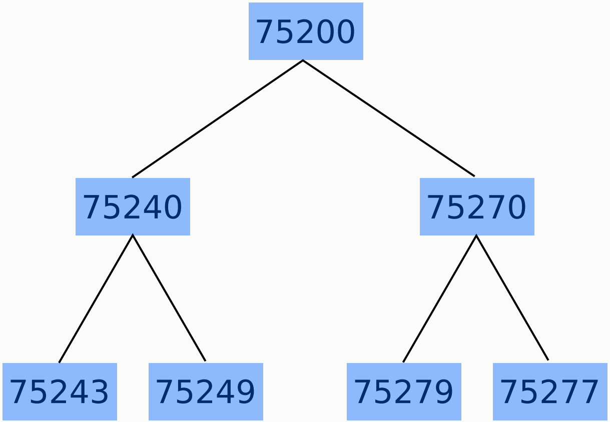
- Perturbative Methods
- **Non-Perturbative Methods**
- Synthetic data

UNIVERSITÄT BONN

Non-Perturbative: Generalization

- In the original table, every attribute value is as specific as possible
  - Values are called to be in the ground domain
- Generalizing attribute values means making them less specific
- Generalization hierarchies may be defined
- Global vs local recoding
  - Global recoding: All occurrences of a value are replaced by the same generalized value
  - Local recoding: Values are replaced by generalizations based on their respective subset they belong to

Generalization hierarchy



Non-Perturbative: Suppression

- Remove data completely from the table
- May be used to adapt the necessary level of generalization
- Outliers may be removed if they would cause a very high generalization level

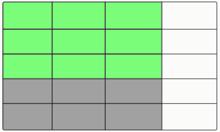
Anonymization

Anonymization - Quality criteria

k-Anonymity

Definition In a released dataset, every unique combination of values of quasi-identifiers must occur for at least k individuals

- These groups of at least k entries are called *equivalence classes*



k-Anonymity

Question How to turn this dataset k-anonymous?

Name	Gender	Age	Zip Code	Illness
Mark	m	31	63457	flu
Tobi	m	33	63455	flu
John	m	20	63455	Corona
Karl	m	22	63447	Cancer
Sophie	f	37	63751	Cancer
Hannah	f	38	63777	Corona

UNIVERSITÄT BONN

k-Anonymity

Question How to turn this dataset k-anonymous?

Name	Gender	Age	Zip Code	Illness
~~Mark~~	m	[30-34]	6345*	flu
~~Tobi~~	m	[30-34]	6345*	flu
~~John~~	m	[20-24]	6345*	Corona
~~Karl~~	m	[20-24]	6345*	Cancer
~~Sophie~~	f	[35-39]	637**	Cancer
~~Hannah~~	f	[35-39]	637**	Corona

k-Anonymity

- Is this the end of privacy breaks?
- Unfortunately no

k-Anonymity: Unsorted Matching Attack

- Entries in published datasets have an order
- Different published k-anonymous versions of the original table may be linked
- The linked (combined) table may destroy the k-anonymity
- **Solution:** Mix the entries of a table before publishment

k-Anonymity: Unsorted Matching Attack

Gender	Zip Code
m	63455
d	63455
m	63439
d	63439
f	63455
f	63439

Original table

Gender	Zip Code
*	63455
*	63455
*	63439
*	63439
*	63455
*	63439

Publication one

Gender	Zip Code
m	634**
d	634**
m	634**
d	634**
f	634**
f	634**

Publication two

UNIVERSITÄT BONN

k-Anonymity: Complementary Release Attack

- Previous attack: Only quasi-identifiers in table
- Real world: More attributes than quasi-identifiers
- Every published table may reveal additional information, even when fulfilling k-anonymity
- The linked (combined) table may destroy the k-anonymity
- **Solution:** Consider all attributes of a published table T as a quasi-identifier or base publishments on T

k-Anonymity: Complementary Release Attack

Name	Gender	Age	Zip Code	Illness
Mark	m	31	63457	flu
Tobi	m	33	63455	flu
John	m	20	63455	Corona
Karl	m	22	63457	Cancer
Sophie	f	37	63751	Cancer
Hannah	f	38	63777	Corona

Original table

k-Anonymity: Complementary Release Attack

Gender	Age	Zip Code	Illness
m	[30-34]	6345*	flu
m	[30-34]	6345*	flu
m	[20-24]	6345*	Corona
m	[20-24]	6345*	Cancer
f	[35-39]	637**	Cancer
f	[35-39]	637**	Corona

T_1

k-Anonymity: Complementary Release Attack

Gender	Age	Zip Code	Illness
m	*	63457	flu
m	*	63457	Cancer
f	*	637**	Cancer
f	*	637**	Corona
m	*	63455	flu
m	*	63455	Corona

T_2

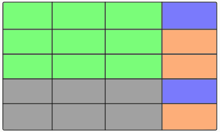
k-Anonymity: Complementary Release Attack

Gender	Age	Zip Code	Illness
m	[30-34]	63457	flu
m	[30-34]	63455	flu
m	[20-24]	63455	Corona
m	[20-24]	63457	Cancer
f	[35-39]	637**	Cancer
f	[35-39]	637**	Corona

Combined table

l-Diversity

Definition: An equivalence class is said to fulfill l-diversity if at least l „well-represented“values for each sensitive attribute occur in it.
A table is said to meet the l-diversity requirement if all its equivalence classes are l-diverse.



l-Diversity: „Well-represented"?

- Distinct l-diversity
- Entropy l-diversity
- Recursive (c, l)-diversity

Distinct l-diversity

For distinct l-diversity „well-represented“ means that there are at least l different values for a sensitive attribute in each equivalence class.

Entropy l-diversity

- Entropy of an equvalence class E is defined as

$$Entropy(E) = -\sum_{s \in S} p(E,s) log(p(E,s))$$

  - S: Domain of sensitive attribute
  - $p(E,s)$: fraction of records in E with sensitive attribute s
- To fulfill the „well-represented“ property, each equivalence class must have $Entropy(E) \geq log(l)$

Recursive (c, l)-diversity

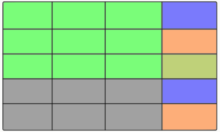
- Most frequent value should not occur too often
- Less frequent values should not occur too rarely
- (c, l)-diversity(E) is fulfilled if the following inequality holds

$$r_1 < c(r_l, r_{l+1}, \ldots, r_m)$$

  - m: number of values in an equivalence class
  - $r_i, 1 \leq r_i \leq m$ number of occurences of the ith most frequent value in E
- (c, l)-diversity holds for the whole table if it holds for all its equivalence classes

t-Closeness

Definition: An equivalence class is said to fulfill t-closeness if the distributions of sensitive attributes in this class and the whole table differ by at least t.
A table is said to meet the t-closeness requirement if all its equivalence classes are t-close.

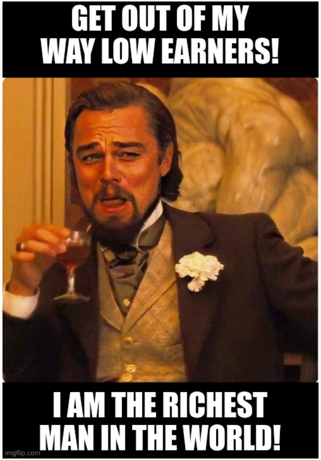


Beyond t-Closeness

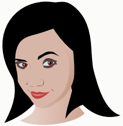
- k-map
- Average Risk
- Population Uniqueness
- Sample Uniqueness
- δ-disclosure privacy
- β-likeness
- β-prensence
- Profitability
- Differential privacy

Secure Multiparty Computation

Millionaire's Problem



Millionaire's Problem

- 2 millionaires
  - Alice 
  - Bob 
- **Question:** Who is richer?
- **Constraint:** No additional information about their wealth should be leaked

Secure Multiparty Computation (MPC)

- m parties P_i would like to compute a joint function $f(x_1, x_2, ..., x_m)$
- P_i knows its own value x_i
- **Goal:** Jointly compute the function by communicating among each other without learning private information of other parties
- **Secure Two-Party Computation (2PC)**: $m = 2$

Millionaire's Problem: Yao's solution

- Alice has a millions & Bob has b millions
- $1 < a, b < 10$
- M: Set of all non-negative integers
- Q_N: Set of all 1-1 onto functions from M to M
- E_a: Public key of Alice, generated by choosing a random element from Q_N
- Protocol for deciding whether $a < b$?

Millionaire's Problem: Yao's solution

- Pick a random N-bit Integer x
- $k = E_a(x)$

$\xleftarrow{k-b+1}$

- $y_u = D_a(k-b+u), u = 1, 2, ..., 10$
- p= random prime of $\frac{N}{2}$ bits
- $\forall u \; z_u = y_u$ mod p
  - If all z_u differ by at least 2 break
  - Else start again with another p

$m_i = z_1, z_2, ..., z_a, z_a + 1, z_{a+1} + 1, ...z_{10} + 1$, p, all numbers are mod p $\longrightarrow$

- If $m_b = x$ mod p, $a \geq b$
- Else $a < b$

Millionaire's Problem: Yao's solution

- Specially fitted solution
- Solutions for other MPC problems?
- Flexibility?

UNIVERSITÄT BONN

Attacker Models

- **Semi-honest**
  - Outputs, computations and sent messages are as given by the original protocol
  - May use its own & received knowledge to compute more than required
  - Tries to infer more knowledge than it should gain
  - $\Rightarrow$Violates mainly privacy constraints
- **Malicious**
  - May arbitrarily deviate from the given protocol
  - $\Rightarrow$ May violate privacy constraints & alter the outcome of the protocol

1-out-of-2 Oblivious transfer

- Alice owns two secret messages M_1, M_2
- Bob should receive exactly one, either M_1 or M_2 such that
- Alice does not know which message M_i Bob received
- 1-out-of-2 Oblivious transfer is denoted as OT_2^1
- Further protocols for OT_n^1 and OT_n^k exist

OT_2^1 protocol based on RSA

Messages M_0, M_1
RSA Keypair $d, (N, e)$

- Generate two random messages m_0, m_1

$$\xrightarrow{(N, e), m_0, m_1}$$

- Choose $b \in 0, 1$
- Generate random message k
- $x = (m_b + k^e)$ mod N

$$\xleftarrow{x}$$

- $k_0 = (x - m_0)^d$ mod N
- $k_1 = (x - m_1)^d$ mod N
- $M'_0 = (M_0 + k_0)$ mod N
- $M'_1 = (M_1 + k_1)$ mod N

$$\xrightarrow{M'_0, M'_1}$$

- $M_b = (M'_b - k)$ mod N

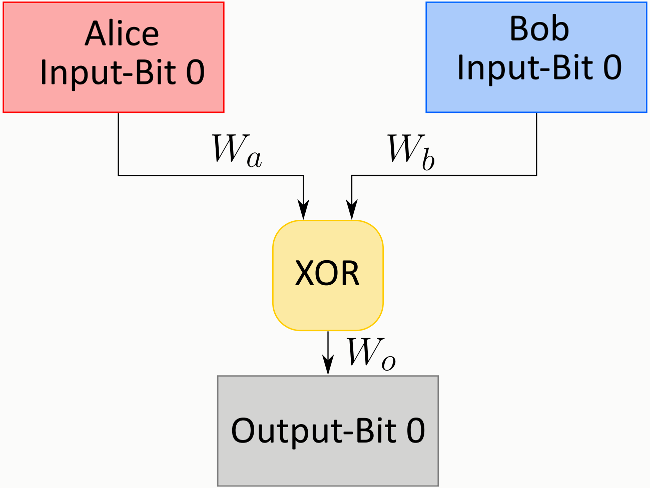
Secure Multiparty Computation

Garbled Circuits

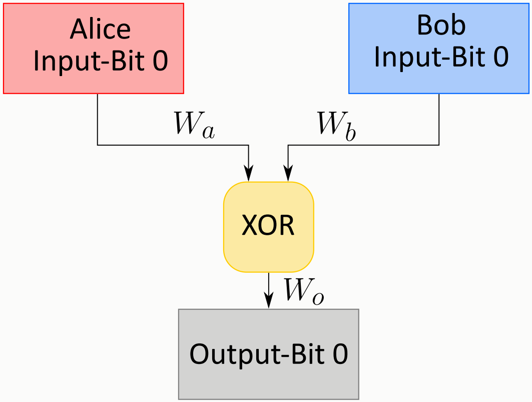
Garbled Circuits

- 2 parties: Alice & Bob (called Garbler & Evaluator)
- Would like to compute a shared function
- With secret inputs each
- $\rightsquigarrow$ Model the function as boolean circuit
- $\rightsquigarrow$ Gates connected by wires
- $\rightsquigarrow$ Circuit is mutually known

Garbled Circuits: Construction

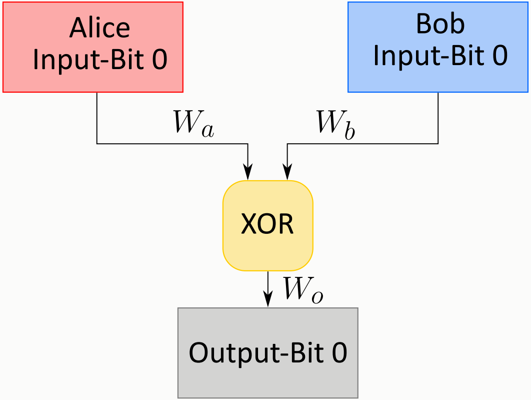


Garbled Circuits: Truth table



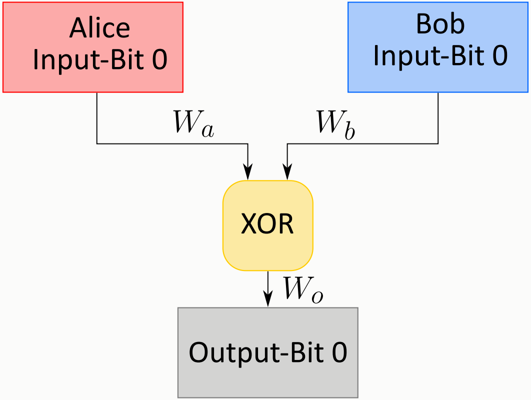
W_a	W_b	W_o
0	0	0
0	1	1
1	0	1
1	1	0

Garbled Circuits: Labelling



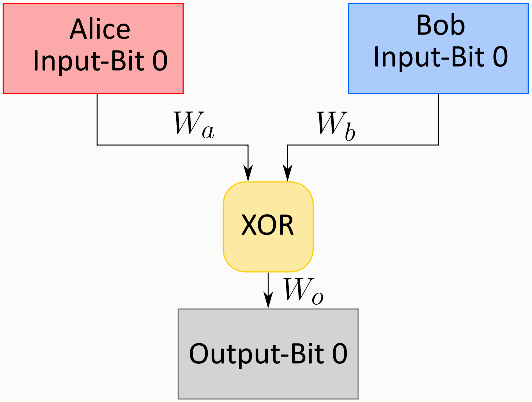
W_a	W_b	W_o
L_a^0	L_b^0	L_o^0
L_a^0	L_b^1	L_o^1
L_a^1	L_b^0	L_o^1
L_a^1	L_b^1	L_o^0

Garbled Circuits: Encryption



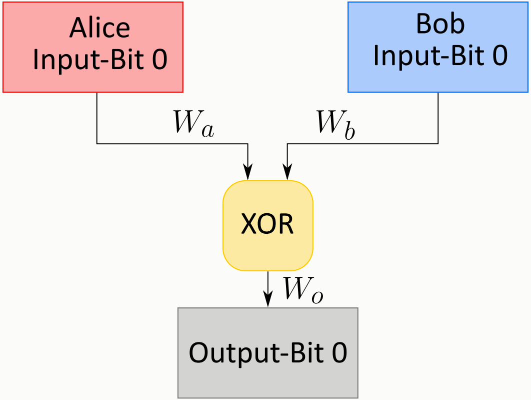
Encrypted Table
$Enc_{L_a^0, L_b^0}(L_o^0)$
$Enc_{L_a^0, L_b^1}(L_o^1)$
$Enc_{L_a^1, L_b^0}(L_o^1)$
$Enc_{L_a^1, L_b^1}(L_o^0)$

Garbled Circuits: Encryption



Encrypted Table
$Enc_{L_a^0, L_b^0}(L_o^0)$
$Enc_{L_a^0, L_b^1}(L_o^1)$
$Enc_{L_a^1, L_b^0}(L_o^1)$
$Enc_{L_a^1, L_b^1}(L_o^0)$

Garbled Circuits: Garbling

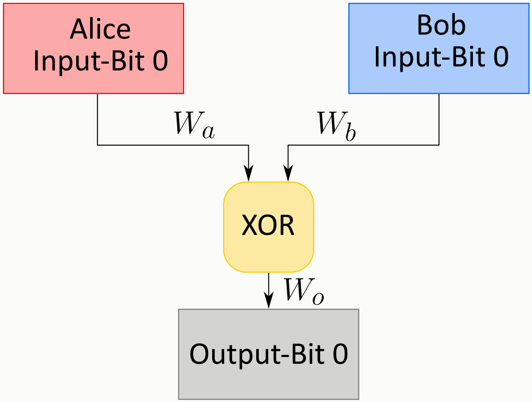


Garbled Table
$Enc_{L_a^1,L_b^0}(L_o^1)$
$Enc_{L_a^1,L_b^1}(L_o^0)$
$Enc_{L_a^0,L_b^1}(L_o^1)$
$Enc_{L_a^0,L_b^0}(L_o^0)$

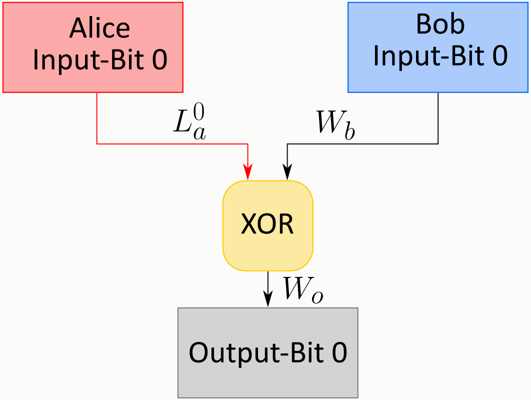
Garbled Circuits: Setup process in a nutshell

- Construct the function as a boolean circuit
- Construct the truth tables for each gate
- Generate a label for each possible state at each wire
- Encrypt the output-wire labels with the input-wire labels
- Garble the encrypted tables
- Share the garbled circuit & the Garbler's input labels with the Evaluator

Garbled Circuits: Evaluation

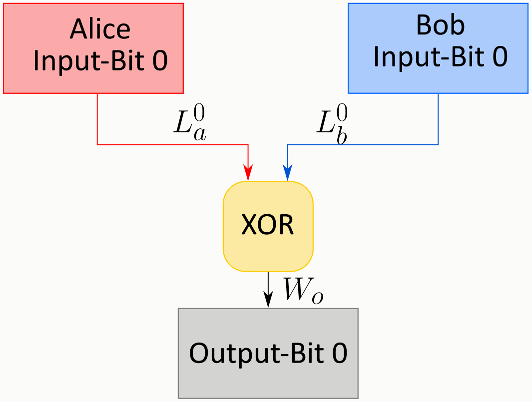


Garbled Circuits: Evaluation



- Bob knows the input label of Alice L_a^0 but not the real value as it is randomly generated

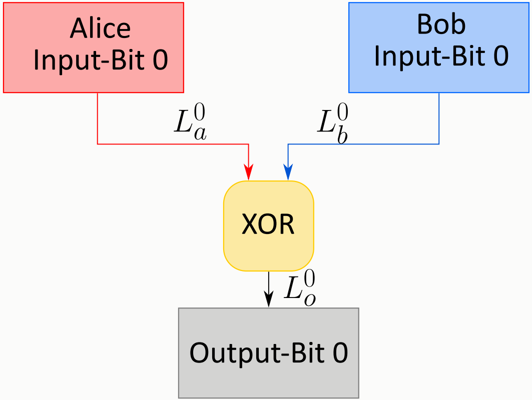
Garbled Circuits: Evaluation



- Bob gets to know the label of its own input from Alice via oblivious transfer

UNIVERSITÄT BONN

Garbled Circuits: Evaluation



- Bob decrypts the garbled table from top to bottom (whereby incorrect decryptions may be detected) until he has decrypted the correct row and gets to know the resulting label

Garbled Table
$Enc_{L_a^1,L_b^0}(L_o^1)$
$Enc_{L_a^1,L_b^1}(L_o^0)$
$Enc_{L_a^0,L_b^1}(L_o^1)$
$Enc_{L_a^0,L_b^0}(L_o^0)$

UNIVERSITÄT BONN

Garbled Circuits: Results

- The result can either be
  - Taken as an input into another gate or
  - Taken as an output bit of the circuit
- Bob shares the output bits with Alice who knows the assignment to the real values
- As specified by the semi-honest model, Alice shares the result with Bob

Secure Multiparty Computation

Performance Optimizations

UNIVERSITÄT BONN

Garbled Circuits: Performance

Technique	Rows			Enc-calls					
				Garbler			*Evaluator*		
	AND	XOR	NOT	AND	XOR	NOT	AND	XOR	NOT
classical	4	4	2	4	4	2	4	4	2

Point & Permute

- Problem: In the worst case, the correct label can be found in the last row of the garbled table
- Inefficient due to up to 3 useless decryption operations
- How to make the decryption process more efficient?
- $\rightsquigarrow$ Point & Permute

Point & Permute

- For each Wire W_w two labels will be generated: L_w^0 & L_w^1
- In point & permute the labels are genarated as follows L_w^{0,s_w^0} & L_w^{1,s_w^1}
- $s_w^0 \in \{0,1\}$ is called the sorting bit & is chosen randomly
- $s_w^1 = s_w^0 \oplus 1$
- Instead of shuffling the table it will be sorted by the random bits
- Bob may immediately find the correct line to decrypt
- No security loss as the sorting bit is independent of a real value

UNIVERSITÄT BONN

Garbled Circuits: Performance

Technique	Rows			Enc-calls					
				Garbler			*Evaluator*		
	AND	XOR	NOT	AND	XOR	NOT	AND	XOR	NOT
classical	4	4	2	4	4	2	4	4	2
point-and-permute	4	4	2	4	4	2	1	1	1

Garbled Row Reduction

- Based on point & permute
- Top row of a garbled table is known as it is sorted by the sorting bits
- Choose the output label in the top row such that it results in a zero-bitstring with length N
- As the top row of the resulting garbled table is known to be 0 it does not need to be transmitted to Bob
- Transmitted table size is reduced by 1 for each gate

UNIVERSITÄT BONN

Garbled Circuits: Performance

Technique	Rows			Enc-calls					
				Garbler			*Evaluator*		
	AND	XOR	NOT	AND	XOR	NOT	AND	XOR	NOT
classical	4	4	2	4	4	2	4	4	2
point-and-permute	4	4	2	4	4	2	1	1	1
GRR_3	3	3	1	4	4	2	1	1	1

Free-XOR

- Altered generation of wire labels
- XOR gates are evaluated without a garbled table (for free)
- $L_a^0, L_b^0, R \in_R \{0,1\}^N$
- $L_o^0 = L_a^0 \oplus L_b^0$
- $L_i^1 = L_i^0 \oplus R \forall i \in \{a, b, o\}$

UNIVERSITÄT BONN

Free-XOR: Proof

- $L_o^0 = L_a^0 \oplus L_b^0 = (L_a^0 \oplus R) \oplus (L_b^0 \oplus R) = L_a^1 \oplus L_b^1$
- $L_o^1 = L_o^0 \oplus R = L_a^0 \oplus (L_b^0 \oplus R) = L_a^0 \oplus L_b^1 = (L_a^0 \oplus R) \oplus L_b^0 = L_a^1 \oplus L_b^0$

Garbled Circuits: Performance

Technique	Rows			Enc-calls					
				Garbler			*Evaluator*		
	AND	XOR	NOT	AND	XOR	NOT	AND	XOR	NOT
classical	4	4	2	4	4	2	4	4	2
point-and-permute	4	4	2	4	4	2	1	1	1
GRR_3	3	3	1	4	4	2	1	1	1
free-XOR	4	0	0	4	0	0	1	0	0

UNIVERSITÄT BONN

Garbled Circuits: Performance

Technique	Rows			Enc-calls					
				Garbler			*Evaluator*		
	AND	XOR	NOT	AND	XOR	NOT	AND	XOR	NOT
classical	4	4	2	4	4	2	4	4	2
point-and-permute	4	4	2	4	4	2	1	1	1
GRR$_3$	3	3	1	4	4	2	1	1	1
free-XOR	4	0	0	4	0	0	1	0	0
GRR$_3$ + free-XOR	3	0	0	4	0	0	1	0	0

Secure Multiparty Computation

Implementation

Secure Hardware Definition Language (SHDL)

- Format for representing garbled circuits
- All gate types supported
- Each lines describes a wire of types input, gate, output
- All non input wires: gate arity, truth table, inputs []

Fairplay library

- No performance improvements
- Implemented in Java
- Circuits can be defined by a Secure Function Definition Language (SFDL) Program which has been developed for use in Fairplay and is C-like
- A SFDL Compiler compiles the program to an optimized Secure Hardware Definition Language (SHDL) Circuit
- Fairplay offers 4 different OT implementations

Fairplay library

- Alice and Bob must be started in separate consoles
- Both get the SHDL circuit and a random seed, Alice get the hostname of Bob additionally, Bob the OT variants indicator number
- For both, the input must be read in via the console
- Second file describing inputs and outputs

UNIVERSITÄT BONN

Bristol Fashion Format

- Format for representing garbled circuits
- AND, XOR, NOT and more supported
- **First row:** No of gates & no of wires
- **Second row:** No of input values & no of input wires for each value
- **Third row:** No of output values & no of output wires per value
- **Gates:** No of inputs & no of outputs & numbers of input wires & number of output wire & gate type

JIGG Library

- JIGG implements point-and-permute & free-XOR
- Circuits are represented in the Bristol-format
- Macro-Assembler allows to reuse small circuits in bigger ones

JIGG Library

- Three parties must be executed: Server, Alice & Bob
- Server must be started once and will run in a listening mode
- Alice and Bob will connect to it, so all of them need to get a common port
- Alice and Bob get their inputs + the circuit

Survey

Lightning Surveys

Markus Krämer
University of Bonn | Institute of Computer Science 4
kraemerm@cs.uni-bonn.de