IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2019-2020

MSc in Computing Science (Specialist)
MSc in Advanced Computing
MEng Honours Degrees in Computing Part IV
MEng Honours Degree in Mathematics and Computer Science Part IV
MEng Honours Degree in Electronic and Information Engineering Part IV
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute

PAPER C408

PRIVACY ENGINEERING

Friday 13th December 2019, 14:00
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators not required

1a Construct the garbled table for an XOR gate with input wires *w1* and *w2* and output wire *w3* with *p*-bits 1, 1, 1 for wires *w1*, *w2* and *w3* respectively. Clearly show your solution using a diagram for the XOR gate, labelling the wires with their keys and permutation bits as well as listing the garbled table entries.

b Assume Alice constructs a garbled circuit with one XOR gate is constructed as in part 1a. Wire *w1* is for Alice's input. Wire *w2* is for Bob's input. Show the steps that Bob takes to evaluate the circuit when Alice's input bit is 1 and Bob's input bit is 0.

c Consider the following *1*-from-*2* oblivious transfer protocol based on the well-known ElGamal encryption scheme in an *honest-but-curious* setting. All operations and random numbers are for a suitable group $\mathbb{Z}/q\mathbb{Z}$ of prime order q and generator g.

Alice's messages are M_0 and M_1 of length n. Bob's message selection bit is b.

1. Alice generates random numbers x and k.

   Alice sends x to Bob.

2. Bob generates a random number y and computes $H_b = g^y$ and $H_{1-b} = x/H_b$.

   Bob sends H_0 and H_1 to Alice.

3. Alice computes $D=g^k$, $C_0 = M_0 \oplus Hash(H_0^k)$, $C_1 = M_1 \oplus Hash(H_1^k)$
   Hash is a cryptographic hash function that produces values of length n.

   Alice sends D, C_0 and C_1 to Bob.

4. Bob computes $M_b = C_b \oplus Hash(D^y)$

For this protocol:

i) Explain why Bob's output equals M_b.

ii) Explain why Alice learns nothing about b and why Bob learns nothing about M_z for $z \neq b$.

d How does Gentry's fully homomorphic encryption scheme control noise?

The four parts carry, respectively, 20%, 20%, 40% and 20% of the marks.

2a Briefly describe 4 weaknesses of the Tor anonymity network.

b Show that a cascade of mixes can be used to anonymously send a message and anonymously receive a reply with 4 mixes *t1*, *t2*, *t3* and *t4*. The message flow for an Alice with Bob communication is Alice → *t1* → *t2* → Bob for the sent message and Bob → *t3* → *t4* → Alice for the corresponding reply message.

c Consider the following 2-round anonymous veto protocol (AV-net) that allows any participant to cast a veto anonymously. There are n participants and they all agree on a suitable group $\mathbb{Z}/q\mathbb{Z}$ of prime order q and generator g. All operations and random numbers are on the group.

Round 1. Each participant P_i generates a random number X_i and broadcasts its public key g^{X_i}.

On completion of round 1 each participant P_i computes:

$$g^{Y_i} = \prod_{j=1}^{i-1} g^{X_j} \Big/ \prod_{j=i+1}^{n} g^{X_j}$$

Recall that the product of a sequence is defined to be 1 if its *lowerbound* > *upperbound*

Round 2. Each participant P_i now broadcasts the value $g^{Y_i V_i}$ using $V_i = X_i$ for "*no veto*" otherwise generating a random number V_i for "*veto*".

On completion of round 2 each participant P_i computes the product of all the values broadcast in round 2. If the product is equal to 1 then no veto has been cast. If the product is not equal to 1 then at least one veto has been cast.

For this protocol:

i) Show that the protocol works for 3 participants where no veto is cast.

ii) Why does the protocol work when no veto is cast? How can an AV-Net be used to solve the Dining Cryptographers problem?

The three parts carry, respectively, 20%, 30% and 50% of the marks.

3 Data anonymization

a i) Give an example of how technology has challenged informal privacy in the past (short answer expected).

ii) Define t-closeness. How does it improve the privacy guarantees of l-diversity?

iii) Write a small dataset that is "well anonymized" according to t-closeness, but from which an attacker can learn information about one user with certainty. What auxiliary information is needed for this attack?

b You and your friend Mark have been asked by the NHS to pseudonymize a large medical dataset to share with a company. You need to figure out how to generate pseudonyms (IDs) for each patient's record based on their concatenated first and last name in upper case (e.g. DONALDTRUMP).

Firstly, you need to decide between hashing the names to produce an ID, and generating random IDs kept in a correspondence table.

i) Is a hash function protecting the privacy of individuals in the dataset better than a correspondence table? Explain.

ii) Give one reason why hash functions are more popular than correspondence tables?

After long debates, you decide to use a hash function with one salt to encode the concatenated name.

iii) For your system to be secure, what do you need to keep secret? What can you disclose publicly?

iv) Mark has decided to add a salt of 4 digits (between 0 and 9) at the end of each name before hashing. To show him that this might not be secure, compute the time it would take to find Mark (whose last name is Lambeth) in such an pseudonymized dataset (assume that computing the hash function takes 1 millisecond, that it has no collision, and that there is only one Mark Lambeth in the dataset).

v) Mark, unwilling to concede, decides that to complicate the task: the salt will be added at a specific (secret) position in the name rather than at the end. How does that increase the time taken to find Mark?

vi) Compute how many digits could you would need to add to the 4-digits salt (while keeping it at the end of the name) to make it more robust than Mark's solution?

You have finally convinced Mark that the salt should be very long. But he has an idea to improve the method: he proposes to use an additional function f to make attacks harder

$$PSEUDOID = f(hash(ID+salt))$$

He chooses the (secret) function $f(x) = x^2 - 3000\,x + 2000000$.

vii) Prove to him that this is a bad pseudonymization method.

The two parts carry, respectively, 35%, and 65% of the marks.

4 Query-based systems and differential privacy

a Define what an ε-differentially private mechanism is.

b Suppose you have a dataset with only two columns: date of birth and salary (1K, 2K, 3K, … up to 100K). Suppose that John is the only person in the dataset born on 1-1-1984. Suppose that we give access to this dataset via a query-based system (QBS) that supports only count queries of the form

count(dob □ … AND salary ○ ...) where □ and ○ can be either = or ≠ (you can have either one or two conditions).

So for example you can have queries such as count(dob = 2-10-1991 AND salary = 31K), count(dob = 2-10-1991 AND salary ≠ 31K), count(salary = 31K), etc. We here assume that date of birth is a quasi-identifier and salary a secret attribute.

For now, suppose that the QBS always responds with the true value to all count queries.

i) Explain what a uniqueness attack is and how a malicious analyst would perform one to find out John's salary.

ii) Suppose that the QBS adds random Gaussian noise to each query. That is, for each query, we have:

Output = true_count + noise

where noise is a random value drawn from a Gaussian distribution N(0,1) generated independently every time an analyst sends a query.

How could an attacker circumvent this to find John's salary? What is the name of this attack?

Suppose you now have a new QBS that adds (condition-specific) static noise, where each noise layer is drawn from a Gaussian distribution N(0,1). This means that each condition in the query yields a fixed noise value that depends on the condition itself.

For example, consider the queries

Q1: count(salary=20K)

Q2: count(dob=1-1-1994 AND salary=20K)

The outputs of these queries would be:

$$\text{Output}_{Q1} = \text{true\_count}_{Q1} + \text{noise}_{\text{salary=20K}}$$

$$\text{Output}_{Q2} = \text{true\_count}_{Q2} + \text{noise}_{\text{salary=20K}} + \text{noise}_{\text{dob=1-1-1994}}$$

where $\text{noise}_{\text{salary=20K}}$ and $\text{noise}_{\text{dob=1-1-1994}}$ are drawn from a Gaussian distribution N(0,1).

iii) Would this new QBS prevent the attack from part b(i) and why? How about the attack from part b(ii) and why?

Another way to attack this system would be as follow. We will now design an attack on such a system. Consider the following two queries:

Q1: count(salary=20K)

Q2: count(dob≠1-1-1994 AND salary=20K)

iv) Using the notation introduced above, write down the output of Q1, Q2, and Q1-Q2 (i.e. counts and noise layers).

v) Similarly, write down the output of Q3 - Q4 (i.e. counts and noise layers).

Q3: count(salary=21K)

Q4: count(dob≠1-1-1994 AND salary=21K)

vi) How can you use this to find John's salary?

Hint: Look at the query sets for each query. How do they differ depending on John's salary?

The two parts carry, respectively, 15% and 85% of the marks.