crypt@b-it 2018: Problem Set 1

1. Sometimes it is not clear whether certain behavior is an "attack" against a protocol. To decide whether something is an attack (i.e., whether it violates security), we have to determine whether such behavior is possible in the ideal world.

   For each of the following, determine whether this behavior is possible in the ideal world.

   (a) Alice & Bob each hold an input in $\{0, \ldots, N-1\}$ and wish to compute their sum modulo N. A malicious adversary corrupts Alice and learns Bob's input in its entirety.

   (b) Alice & Bob each hold an input in $\{0, \ldots, N-1\}$ and wish to compute their sum modulo N. A malicious adversary corrupts Alice and forces Bob to always output zero.

   (c) Alice holds x and Bob holds y, where $x, y \in \{0, \ldots, 7\}$, and wish to compute whether $x < y$ (i.e., they get output 1 if $x < y$ and output 0 otherwise). A malicious adversary corrupts Alice and learns the most significant bit of y.

   (d) Alice holds x and Bob holds y, where $x, y \in \{0, \ldots, 7\}$, and wish to compute whether $x < y$ (i.e., they get output 1 if $x < y$ and output 0 otherwise). A malicious adversary corrupts Alice and causes Bob to output the least significant bit of his input y.

   (e) Alice holds x and Bob holds y, where $x, y \in \{0,1\}^n$. They wish to compute the inner product of those strings modulo 2: $\sum_{i=1}^{n} x_i y_i \pmod 2$. A malicious adversary corrupts Alice and learns whether the y string has a majority of 0s or a majority of 1s (assume that n is odd).

   (f) Alice and Bob each hold a single bit and wish to compute the boolean-AND of their inputs. A malicious adversary corrupts Alice and learns Bob's input in its entirety.

2. Suppose Alice has an input $x \in \{0, 2, 4, \ldots, 8\}$ and Bob has an input $y \in \{1, 3, 5, \ldots, 9\}$. Here is a protocol that computes the function $f(x, y) = \max\{x, y\}$:

   - If Bob has input $y = 9$, he announces "yes" and both parties output 9 and halt. Otherwise he announces "no" and the protocol continues.
   - If Alice has input $x = 8$, she announces "yes" and both parties output 8 and halt. Otherwise, she announces "no" and the protocol continues.
   - ...

   The protocol continues until some party says "yes", at which point the output is determined and the protocol is finished.

   Show that this protocol is secure against semi-honest adversaries by describing appropriate simulators.

3. Consider a variant of the above protocol, where $x, y \in \{0, \ldots, 9\}$. It still computes $f(x, y) = \max\{x, y\}$:

   - If Bob has input $y = 9$, he announces "yes" and both parties output 9 and halt. Otherwise he announces "no" and the protocol continues.
   - If Alice has input $x = 9$, she announces "yes" and both parties output 9 and halt. Otherwise, she announces "no" and the protocol continues.
   - ... repeat for $y = 8$ then $x = 8$, and so on...

   Show that this protocol is *not* secure against semi-honest adversaries.

4. We often assume that both parties learn the output of the function from 2PC. Suppose we modify the ideal world so that the functionality gives separate outputs to both parties. In more detail, suppose there are two functions f_A and f_B, and in the ideal world, the functionality receives x from Alice and y from Bob, then gives (only) $f_A(x, y)$ to Alice and (only) $f_B(x, y)$ to Bob.

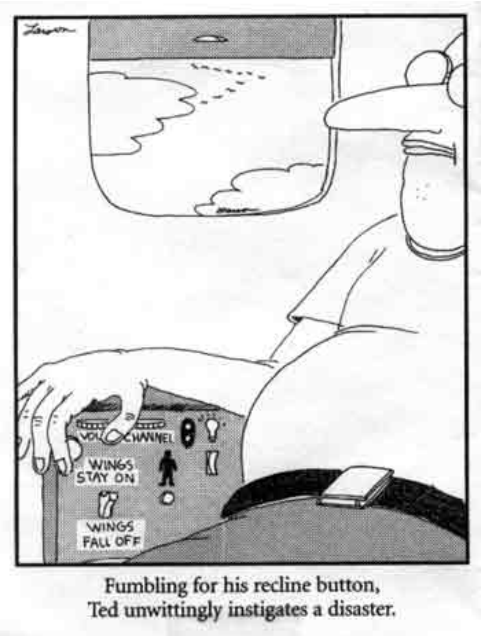
   (a) Give an example f_A and f_B where it is demonstrably insecure (i.e., less secure than the ideal world described above) if *both* parties learn $f_A(x, y)$ and $f_B(x, y)$.

   (b) Suppose Alice and Bob know a way to securely compute *any* function $f(x, y)$, but only in a way where they *both* learn the output.

   Suggest a way for them to use this ability to securely compute different functions of the same inputs. Given f_A and f_B, Alice and Bob should securely compute a related function f^* (with outputs to both parties), which should allow only Alice to learn $f_A(x, y)$ and only Bob to learn $f_B(x, y)$.

   Hint: f^* should use a one-time pad to hide some of the output from one of the parties.

5. Describe a 2PC protocol that is secure against semi-honest adversaries but is completely insecure against malicious adversaries.



Fumbling for his recline button,
Ted unwittingly instigates a disaster.

crypt@b-it 2018: Problem Set 2

1. In light of Free-XOR, it is desirable to minimize the number of AND gates in a boolean circuit.

   (a) Show a circuit for equality of n-bit strings (output 1 if and only if $x = y$) using only $n - 1$ AND gates.

   ★ (b) A full adder has inputs a, b, c (c = carry-in) and computes outputs $s = a \oplus b \oplus c$ and $c' = (a \wedge b) \vee (a \wedge c) \vee (b \wedge c)$ (carry out). Show how to write a full adder using only 1 AND gate.

2. Consider garbling an AND gate with standard point-and-permute, where we use nested one-time pad for the encryption. For example (after point-and-permute), the garbled gate might consist of $(G_{00}, \ldots, G_{11})$ where:

$$\begin{aligned} G_{00} &= A_0 \oplus B_0 \oplus C_0 \\ G_{01} &= A_0 \oplus B_1 \oplus C_0 \\ G_{10} &= A_1 \oplus B_0 \oplus C_1 \\ G_{11} &= A_1 \oplus B_1 \oplus C_0 \end{aligned}$$

   (A_0, A_1 and B_0, B_1 are the input wire labels, and C_0, C_1 are the output wire labels) Argue that this is an insecure garbling scheme. Which security property/properties (privacy, obliviousness, authenticity) can you break?

3. Suppose for simplicity that H is a random oracle. We have presented standard point-and-permute garbling as in the left example below:

$$\begin{aligned} G_{00} &= H(A_0, B_0) \oplus C_0 \\ G_{01} &= H(A_0, B_1) \oplus C_0 \\ G_{10} &= H(A_1, B_0) \oplus C_1 \\ G_{11} &= H(A_1, B_1) \oplus C_0 \end{aligned} \qquad \begin{aligned} G_{00} &= H(g, A_0, B_0) \oplus C_0 \\ G_{01} &= H(g, A_0, B_1) \oplus C_0 \\ G_{10} &= H(g, A_1, B_0) \oplus C_1 \\ G_{11} &= H(g, A_1, B_1) \oplus C_0 \end{aligned}$$

   However, in practice it should be implemented as in the right example, where g is a *unique index for each gate.* Show that the left example is actually insecure! (The problem is maybe a bit easier if you assume free-XOR wire labels.) Which security property/properties (privacy, obliviousness, authenticity) can you break? How does including the gate index g fix the problem?

   Hint: Consider garbling a circuit consisting of two different gates that have the same input wires.

★ 4. In the Free-XOR scheme, no cryptographic operations are required to evaluate an XOR gate. However, cryptographic operations are still required to evaluate a garbled AND gate.

   Suppose we try to avoid or reduce cryptographic operations for AND gates, with a scheme that has the following properties:

   - It is compatible with Free-XOR; i.e., all wires have labels of the form $W, W \oplus \Delta$ for some global Δ.
   - It uses standard point-and-permute; i.e., every wire label has a "color bit" that is independent of its truth value and visible to the evaluator.
   - The evaluation procedure for an AND gate works like this. Suppose the input labels of this gate are A^* and B^*. If the "color bits" are (0,0), then the output label is $A^* \oplus B^*$.

So in this scheme, at least in some situations (when the color bits are 0,0), you can evaluate an AND gate with no cryptographic operations. In other situations, some cryptographic operations may be required.

Show that such a scheme cannot be secure. Again, try to be specific about which security property is violated.

5. When discussing garbling of *boolean* circuits, we have mostly discussed only XOR & AND gates. What about NOT gates? What is the cost of garbling a NOT gate (with and without free-XOR)?

6. The **parity** of a boolean gate is the parity of the number of 1s in its truth table. Hence, XOR and XNOR gates have even parity; AND, OR, NAND gates have odd parity.

   Recall the Gueron-Lindell-Nof-Pinkas garbling scheme for 2-ciphertext AND gates. Its main idea is to choose output wire labels so that (1) the first ciphertext is all zeroes, (2) the XOR of the other 3 ciphertexts is all zeroes.

   (a) Does the scheme always work for arbitrary odd-parity gates? Why / why not? Be sure to consider both cases for the "payload" of the first ciphertext.

   (b) Does the scheme always work for arbitrary even-parity gates? Why / why not?

7. The Ball-Malkin-Rosulek garbling scheme supports free addition mod m and supports non-free unary gates.

   (a) Show how to write a boolean AND-gate (of fan-in k) as a composition of addition mod-m and a unary gate. What is the cost to garble such a gate? Be careful about the choice of modulus m.

   (b) Suggest a way to garble an AND gate (fan-in 2) using just 1 ciphertext (combining these techniques with others from the lecture). Discuss the drawbacks of this scheme, and why it has not made half-gates obsolete.

8. Here is a different approach for garbling an AND gate for 2 ciphertexts. View $\{0,1\}^\lambda$ as the finite field $GF(2^\lambda)$. Recall that the evaluator will be able to compute at most one of the following values:

$$K_1 = H(A_0, B_0); \qquad K_2 = H(A_0, B_1); \qquad K_3 = H(A_1, B_0); \qquad K_4 = H(A_1, B_1)$$

   (a) Given arbitrary values $K_1, \ldots, K_4$ (i.e., you have no control over them), describe how to compute two polynomials P and Q over $GF(2^\lambda)$ such that:

   - $P(5) = Q(5)$ (I write "5" to refer to the $GF(2^\lambda)$ element whose binary representation is $\cdots 000101$)
   - $P(6) = Q(6)$
   - 3 of the points $(1, K_1), (2, K_2), (3, K_3), (4, K_4)$ lie on polyomial P and the other one lies on polynomial Q.

   (b) Suggest a way to use this observation to garble an AND-gate for 2 ciphertexts. The evaluator will construct such polynomials P and Q. What values should comprise the garbled gate? What should the output wire labels be? What should the evaluator do?

   Hint: The garbled gate should consist of just $P(5)$ and $P(6)$

crypt@b-it 2018: Problem Set 3

1. Generalize the random-OT-derandomization protocol. In 1-out-of-n OT, the sender has strings $m_0, \ldots, m_{n-1}$; the receiver has selection value $c \in \{0, \ldots, n-1\}$ and learns m_c.

   Suppose parties have already performed an instance of 1-out-of-n **random OT** (where all the values $m_0, \ldots, m_{n-1}$ and c are uniform in their respective domains). Show how to efficiently derandomize for a 1-out-of-n OT instance on chosen inputs.

2. If you want to use IKNP to get 1 million extended OTs, you need 128 base OTs on 1-million-bit strings.

   Give a construction that achieves 1-out-of-2 OT of N-bit strings, using an instance of 1-out-of-2 OT of λ-bit strings ($N \gg \lambda$).

3. The dual-execution protocol of Mohassel-Franklin leaks one bit to an adversary. Show that for **any** efficiently-computable predicate $P : \{0,1\}^* \to \{0,1\}$, there is indeed an attack on the protocol where the adversary learns $P(x)$, where x is the honest party's input.

   Can you suggest a change to the dual-execution protocol to restrict the class of possible P in some way?

4. IKNP OT extension requires a function $H : \{0,1\}^\lambda \to \{0,1\}^n$ such that $t \mapsto H(t \oplus s)$ is a PRF with seed s (actually IKNP only requires this to be a *weak* PRF).

   You can prove that a random oracle H satisfies this property using the H-coefficient technique. Consider the following two systems:

Initialize: $H \leftarrow$ random function $s \leftarrow \{0,1\}^\lambda$ QueryH(x): return $H(x)$ QueryCons(t): return $H(t \oplus s)$	Initialize: $H \leftarrow$ random function $F \leftarrow$ random function QueryH(x): return $H(x)$ QueryCons(t): return $F(t)$

   Prove a concrete bound on the maximum distinguishing advantage of these two systems.

   Hint: Add something extra to the transcript, and use it to characterize a bad event.