CS4501 Cryptographic Protocols, Homework 4

Response by: Your Name, (Computing ID)

Total points: 40 awarded maximum. 55 available. Points are noted after each problem.

Instructions. For each problem, typeset your solution in the `answer` environment, and if there are sub-problems, mark them clearly. Feel free to use as much space as you require, and be sure to update your name and computing ID above, and the acknowledgments box at the end.

Policies. In short, you are encouraged to think about the problems on your own, and then discuss them and work toward solutions with your classmates. You must write and submit your own solutions. You may also read any published material that helps you come to an understanding of the problems, but you must acknowledge and/or reference any discussion or published material, with the exception of lecture notes and other official class materials, in-class and in-office-hours discussions, and basic LaTeX help or dictionary lookups. It is a violation of the honor code if any of the following occur:

- You copy text directly from any source.
- You use any material or discussion without acknowledgment or citation, excluding the above special cases.
- You are unable to explain your work orally.

See https://jackdoerner.net/teaching/2026/Spring/CS4501/#policies for more details.

Problem 1 (Information Theoretic Garbling (3pts each))**.** In this question we will explore an *information-theoretic* variant of garbled circuits, in which we use one-time-pad as the encryption function for constructing garbled truth tables, instead of a computationally-secure encryption scheme. For simplicity, we will assume that garbled tables are *not* permuted, and thus there is *no need* for the pointer bits or evasive-range encryption techniques that we saw in Lectures 16 and 17.

(a) First, consider a garbled AND gate with input wires u and v and output wire w. Let $\kappa \in \mathbb{N}$ and assume that the keys of the input wires (k_u^0, k_u^1) and (k_v^0, k_v^1) and the keys of the output wire (k_w^0, k_w^1) are independently uniformly distributed κ-bit strings. Given these labels, our garbled table is:

$$
\begin{aligned}
c_{uvw}^1 &= k_w^0 \oplus k_u^0 \oplus k_v^0 \\
c_{uvw}^2 &= k_w^0 \oplus k_u^1 \oplus k_v^0 \\
c_{uvw}^3 &= k_w^0 \oplus k_u^0 \oplus k_v^1 \\
c_{uvw}^4 &= k_w^1 \oplus k_u^1 \oplus k_v^1
\end{aligned}
$$

Show that given that given the above table and the garbled wire values k_u^0 and k_v^0, it is possible to recover both output-wire keys k_w^0 and k_w^1.

(b) Explain how to prevent such an attack by using input-wire keys of length 2κ bits (while the output wire keys remain κ bits long).

(c) Using the idea you developed in the previous sub-problem, garble the circuit $(a \wedge b) \wedge c$. Note that you can begin with $\kappa = 1$ for the key-length of the output wire.

(d) Suppose that we wish for computationally bounded parties to use the information-theoretic garbling technique that you developed in the last two sub-problems. What is the maximal multiplicative depth of circuits (as a function of κ) that your garbling scheme supports?

Problem 2 (No Really, Dishonest-Majority BA is Impossible (5pts))**.** Let $n = 4$ and $t = 2$. Assume that parties have access to an ideal broadcast functionality. In this case, the hexagon argument for the impossibility of Byzantine Agreement when $t \geq n/3$ (which we saw in Lecture 20) no longer holds.

Prove that there is no 4-party 2-secure Byzantine Agreement in this setting.

Hint: We talked informally in class about why this is true. Now I'm asking you to give me a formal proof.

Problem 3 (More on Dolev Strong (11pts))**.** Let $n \in \mathbb{N}$ and $t = n/2$.

(a) (5pts) Consider a variant of the Dolev-Strong protocol that runs for t rounds instead of $t + 1$ rounds. Prove that this variant is not secure. Specifically, fix n and t, and fix an input value for the sender. Then, describe a specific attack that violates consistency or validity.

(b) In the Dolev-Strong protocol no party needs to send more than two messages (the second message is a certificate that the sender is cheating), and an honest party can simply ignore any *malformed* messages it receives. Let the *message complexity* be the number of messages that are either sent or received and processed (i.e. not ignored) by honest parties. Do not double-count a message that one honest party sends to another honest party.

(1) (3pts) What is the message complexity of Dolev-Strong in an honest execution among a full set of honest parties?

(2) (3pts) What is the message complexity of Dolev-Strong when an adversary that can maliciously corrupt t parties?

Problem 4 (Malicious 1-out-of-4 OT in the 1-out-of-2 OT Hybrid Model (5pts))**.** In class, we proved that one can construct perfectly secure 1-out-of-4 OT in the 1-out-of-2 OT Hybrid model when the adversary is semi-honest. Prove that the same result holds when the adversary is malicious.

Hint: You can and should use the exact same protocol as we did in the semi-honest case, and the same assumptions (i.e. none).

Problem 5 (A Conundrum Within a Conundrum (7pts))**.** Consider again a function $f : \{0,1\}^* \times \{\lambda\} \to \{0,1\}^* \times \{0,1\}^*$ such that $f(x, \lambda) = (g(x), h(x, g(x)))$ where $g(x)$ is some randomized process and h is deterministic.

In Problem 2c of Homework 1, you proved that there exists a *one-message* protocol that *perfectly*

securely computes any such f (under Definition 2 of Lecture 3) in the presence of a semi-honest adversary that statically corrupts at most one party.

In Problem 5 of Homework 3, you proved that if the same one-message protocol also *computationally* securely computes any f (under Definition 7 of Lecture 14) in the presence of a semi-honest adversary that statically corrupts at most one party, then one-way functions *do not exist*.

(a) (5pts) Let R be any binary *relation* of your choice, represented as a two-input function that outputs a single bit that is equal to 1 if and only if the relation holds for the inputs. So for example the "is-a-factor-of" relation $R_{\mathsf{factor}}(x, w)$ outputs 1 if and only if w evenly divides x.

Next, consider the *zero-knowledge* functionality $\mathcal{F}_{\mathsf{ZK}}^{R}$ which is parameterized by an arbitrary relation R. $\mathcal{F}_{\mathsf{ZK}}^{R}$ interacts with two parties. It receives a *statement* x and a *witness* w from P_1, and P_2 provides no input. It outputs $(x, R(x, w))$ to P_2, and provides no output to P_1. In other words P_1 can use $\mathcal{F}_{\mathsf{ZK}}^{R}$ to convince P_2 that it knows some secret w such that the chosen relation holds on w and a public x.

Construct a *new* one-message protocol in the $\mathcal{F}_{\mathsf{ZK}}^{R}$-hybrid model (where R can depend upon any of f, g, h, even in a non-black-box way). Your one-message protocol should involve exactly one message from P_1 to P_2, plus exactly one invocation of $\mathcal{F}_{\mathsf{ZK}}^{R}$, where P_1 supplies the inputs and P_2 gets the output. Prove that your new protocol securely computes any f (of the form specified at the beginning of this problem) in the presence of a *malicious* adversary that statically corrupts at most one party.

(b) (2pts) Does your new protocol also achieve semi-honest security, or does the same proof you wrote for homework 3 still apply? You do not need to reprove anything - just give some intuition.

Problem 6 (Arithmetic Garbling (15pts))**.** In class, we explored Yao's Garbled Circuits, which is used to garble boolean circuits (i.e. circuits over $\mathbb{F}_2$). In this problem, we will see how Yao's Garbled Circuits can be generalized to larger rings. Recall that a finite *commutative ring* is like a finite field, except that multiplicative inverses do not necessarily exist for every element.[1] The integers modulo a *non-prime* value form a finite commutative ring. For example, $\mathbb{Z}_{15}$ is a ring. For this problem, assume that Garbled Circuits are *never* optimized and evasive-range encryption is always necessary.

(a) (5pts) Describe how Yao's Garbled Circuits can be *directly* generalized to the ring $\mathbb{Z}_m$ for any m. That is, consider circuits in which wires can take any value in $\mathbb{Z}_m$ for some arbitrary but fixed (and possibly non-prime) modulus m. Describe how wires with m distinct values are represented in your garbling scheme, and how the gates $+_m$ and $\times_m$ (that is, modular addition and multiplication with respect to m) can be garbled. You do not need to prove security for your garbling scheme, but it should be clear that the proof of the classic Yao protocol can be generalized to cover yours. For the rest of this problem, we will refer to the scheme you have developed here as the "basic arithmetic garbling scheme"

(b) (1pts) Assuming that you use evasive-range encryption like we did for our initial construction of Yao's Garbled Circuits in class, what is the bandwidth cost of garbling a single two-input gate using the basic arithmetic garbling scheme from the previous sub-problem, as a function of the modulus m and security parameter κ?

[1] A commutative ring also does not necessarily have a multiplicative identity that is different from its additive identity. However, for all of the rings we consider in this homework problem, we have $0 \neq 1$.

(c) (5pts) Suppose that $\mathbb{X}$ and $\mathbb{Y}$ are two finite commutative rings with operations denoted $(+_{\mathbb{X}}, \cdot_{\mathbb{X}})$ and $(+_{\mathbb{Y}}, \cdot_{\mathbb{Y}})$ respectively. The *direct product* of $\mathbb{X}$ and $\mathbb{Y}$, denoted $\mathbb{W} = \mathbb{X} \times \mathbb{Y}$ is a finite commutative ring over the underlying set $\{(x, y) : x \in \mathbb{X}, y \in \mathbb{Y}\}$. If $(x_1, y_1) = w_1 \in \mathbb{W}$ and $(x_2, y_2) = w_2 \in \mathbb{W}$, then we have $w_1 +_{\mathbb{W}} w_2 = (x_1 +_{\mathbb{X}} x_2, y_1 +_{\mathbb{Y}} y_2)$ and $w_1 \cdot_{\mathbb{W}} w_2 = (x_1 \cdot_{\mathbb{X}} x_2, y_1 \cdot_{\mathbb{Y}} y_2)$. In other words, each element of a direct-product ring is a tuple containing exactly one element from each of the constituent rings, and operations in a direct-product ring are performed element-wise. Note that the notion of a direct product can be generalized to any number of constituent rings.

The *Chinese Remainder Theorem* (CRT) states that for any $a, b \in \mathbb{N}$ such that a and b are coprime, there is an isomorphism between the ring $\mathbb{Z}_{a \cdot b}$ and the direct-product ring $\mathbb{Z}_a \times \mathbb{Z}_b$. So for example, if we consider $a = 5, b = 3$, the CRT tells us that there exists a bijective map $\phi : \mathbb{Z}_5 \times \mathbb{Z}_3 \to \mathbb{Z}_{15}$ such that for every $x_1, x_2 \in \mathbb{Z}_5$ and $y_1, y_2 \in \mathbb{Z}_3$, we have $\phi(x_1 \cdot_5 x_2, y_1 \cdot_3 y_2) = \phi(x_1, y_1) \cdot_{15} \phi(x_2, y_2)$. To put it another way, the CRT tells us that we can perform a multiplication in $\mathbb{Z}_{15}$ by performing one multiplication in $\mathbb{Z}_5$ and one multiplication in $\mathbb{Z}_3$. This notion can be generalized to any set of coprime moduli.

Suppose I give you a composite modulus $m \in \mathbb{N}$ and a set X containing the coprime factors of m (that is, $m = \prod_{x \in X} x$). Describe how you can use the map ϕ guaranteed by the Chinese Remainder Theorem together with the basic arithmetic garbling scheme you constructed in the first-sub problem to construct a second arithmetic garbling scheme for $\mathbb{Z}_m$, with better efficiency properties than the basic one.[2]

(d) (4pts) Suppose that you wish to multiply two 32-bit integers *without overflow*. If you were writing code for an ordinary, physical computer with a 64-bit CPU, you would probably achieve this by performing a 64-bit multiplication. Using Yao's Garbled Circuits, you could likewise construct a 64-bit boolean multiplication circuit. The naive schoolbook multiplication circuit[3] contains $6\ell^2 - 8\ell$ AND and XOR gates, given ℓ-bit inputs. Since each gate requires 12κ bits to garble, where κ is the security parameter, the bandwidth cost of a 64-bit multiplication is 36,962,304 bits, or 4.62 MB, assuming $\kappa = 128$.

As an alternative, you could choose $m \geq 2^{64}$ and garble a *single* multiplication gate over $\mathbb{Z}_m$. Using your basic arithmetic garbling scheme from the first sub-problem, what is the bandwidth cost of a single multiplication gate over $\mathbb{Z}_{2^{64}}$, when $\kappa = 128$?

On the other hand, suppose that you choose m to be a so-called *primorial*[4] modulus that is sufficiently large to prevent overflow. Specifically, if we let $X = \{2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 32, 37, 41, 43, 47, 53\}$ be the set containing the 16 smallest primes, then $m = \prod_{x \in X} x = 32589158477190044730 > 2^{64}$. Using your second arithmetic garbling scheme from the third sub-problem, what is the bandwidth cost of a single multiplication gate over $\mathbb{Z}_m$, when $\kappa = 128$?

Given the task of overflow-free 32-bit multiplication, which of the three approaches[5] is best, from the perspective of minimizing bandwidth?

[2] You do not need to discuss the efficiency of your second scheme yet; we will explore it in the next sub-problem.

[3] Also known as a *Braun Multiplier*.

[4] A primorial number is the product of sequential primes, starting from 2. In other words, a primorial is like a factorial over the sequence of primes instead of the sequence of natural numbers.

[5] That is, representing the multiplication as a boolean circuit and garbling over $\mathbb{F}_2$, or garbling over $\mathbb{Z}_{2^{64}}$ using the basic arithmetic garbling scheme, or garbling over $\mathbb{Z}_m$ where $m > 2^{64}$ using the second arithmetic garbling scheme.

Acknowledgments

In this box, you should acknowledge your collaborators and the resources you used, if any. For example:

Problem 1: I discussed this problem with Alice and Bob. In addition, I asked Carol for help understanding Conditional Probability, but we did not discuss the problem further.

Problem 2: I asked ChatGPT "What is a turing machine?", and it gave me the following transcript: https://chatgpt.com/share/68b4dba7-e19c-8013-a137-e8db901493b7.

Problem 3: It helped me to read the proof of [Vad12, Theorem X, Page Y].

Instructor's Acknowledgments

Problems in this homework have been borrowed from or inspired by a number of sources. Citations can be provided on request, after the homework is completed.

References

[Vad12] Salil P. Vadhan. *Pseudorandomness*. 2012. https://people.seas.harvard.edu/~salil/pseudorandomness/pseudorandomness-published-Dec12.pdf.