650.445: Practical Cryptographic Systems December 8, 2012

Final Exam

Instructor: Matthew Green *Due: 11:59 pm, December 20*

Name: _____

This exam assignment is to be completed *individually*. Except where specifically indicated, you are not to collaborate or otherwise share information with any other person. You *are* permitted to use the Internet and any printed references. Electronically-submitted assignments may be emailed to 650445@gmail.com before midnight on December 20, or submitted via Blackboard.

Problem 1: True or False (10 points)
You do not need to justify your answer.

1. True or False: The HDCP v2.1 protocol is vulnerable to protocol attacks, but there are no practical attacks on the underlying primitives (*e.g.,* block ciphers).
2. True or False: The CBC-mode ciphersuite of the TLS 1.0 protocol (RFC 2246) generates a random Initialization Vector prior to encrypting each TLS record.
3. True or False: Shor's algorithm does not apply to the Diffie-Hellman crypto system (*i.e.,* Diffie-Hellman key agreement remains secure even in the presence of a quantum computer running the algorithm).
4. True or False: A significant advantage of Quantum Key Distribution is its invulnerability to side-channel attacks.
5. True or False: The proof of security for full-domain RSA signatures (discussed in class) is still valid if you implement the hash function $H()$ with a real hash function such as SHA3.
6. True or False: Dan Brown's popular 1999 fiction novel *Digital Fortress* describes an encryption scheme that uses "rotating cleartext" to render brute force key search ineffective. Rotating cleartext concept is a real idea that has a scientific basis.
7. True or False: As of this moment, there are no known quantum algorithms for solving lattice problems (that perform significantly better than classical algorithms).
8. True or False: The recomended technique for verifying RSA signature padding (*e.g.,* RSA-PKCS #1v1.5) is to reconstruct the padding from scratch and apply strcmp().
9. True or False: Hash functions like SHA1 are vulnerable to length-extension attacks.
10. True or False: As far as we know, side-channel attacks are only applicable to the RSA cryptosystem.

Problem 2: Short Answer (10 points)

1. **Shor's Algorithm.** Explain what Shor's algorithm is and what it is used for. What implications might the existence of this algorithm have for our decisions about which cryptosystems to use right now.

Problem 3: CBC-MAC and Proofs (30 points)

A common approach to building a Message Authentication Code (MAC) is to use a block cipher in CBC mode. This MAC, known as CBC-MAC, is computed by running CBC mode encryption on a message using a fixed IV (typically 0) and outputting the final block of ciphertext as the MAC. Let $M = m1||m2||\ldots||m_x$, where each m_i the size of the cipher's block length. The following diagram illustrates the computation of CBC-MAC:

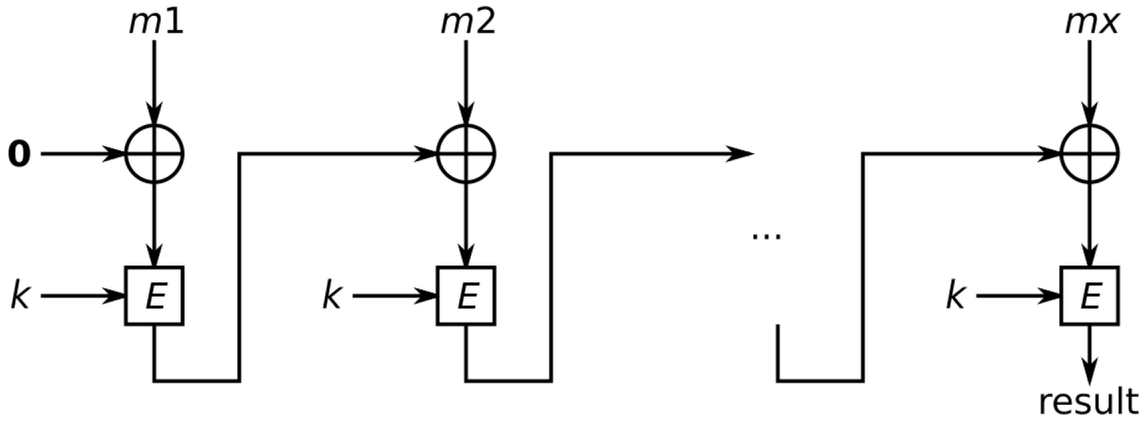

Figure 1: Description of CBC-MAC, courtesy Wikipedia.

A secure MAC should satisfy the definition of *Existential Unforgeability under Chosen-Message Attack* (EU-CMA). Briefly, this means that no Adversary should win the following game played against an honest Challenger except with negligible probability:

1. The Challenger picks a random λ-bit key k.
2. As many times as the Adversary likes, s/he can send a "query" M_i to the Challenger. The Challenger computes $T_i = \mathsf{CBCMAC}(k, M_i)$ and returns T_i to the Adversary.
3. The Adversary wins if s/he can output any pair (M^*, T^*) such that $T^* = \mathsf{CBCMAC}(k, M^*)$ and for all i, $(M^*, T^*) \neq (M_i, T_i)$ (*i.e.,* the adversary has not previously issued a query on M^* and received T^* in response).

Consider the following questions:

1. For this question, *assume that all messages are of a pre-determined fixed length* (exactly N blocks long, for some arbitrary N).

   Let E be an ideal cipher with a λ-bit key and ℓ-bit input/output. Using the ideal cipher model heuristic, give an informal argument as to why CBC-MAC might be a secure MAC under the EU-CMA definition above. Note that this does not need to be a proof, but it should be complete enough to convince me that CBC-MAC satisfies the informal definition above.

   (Hint: The structure of your argument should be as follows. First, consider an Adversary playing the game with a Challenger. Assume that both parties are calling out to an ideal cipher oracle [the "gnome"] each time they call the E function on some key and plaintext. Explain how the Challenger will produce each MAC requested by the Adversary. For the pair (M^*, T^*) output in step 3, explain the properties of the tag T^* and convince me of the Adversary's probability of finding such a tag.)

2. CBC-MAC is not secure when the messages are of *variable-length*. To forge a message, the adversary can query on two messages M, M' to obtain MAC tags T, T' respectively. Let us represent M' as $\{m'_1, \ldots, m'_N\}$. The Adversary can now compute a third message M'' as follows:

   $$M'' = M||(m'_1 \oplus T)||m'_2|| \ldots ||m'_N$$

   What is the correct MAC on the message M''. How does the Adversary find it?

3. Bonus (5 points): Some CBC-MAC variants prepend the *length* of the message M to the message before computing the MAC. Explain how this foils the attack described above.

Problem 4: Key privacy (20 points)

Recall that the RSA encryption algorithm is defined as $C = m^e \ mod \ N$, where (N, e) form the recipient's public key. As discussed in class, when the value m has been formatted using an appropriate randomized padding scheme (such as OAEP), RSA encryption is semantically secure.

However, imagine that the attacker's goal is not to learn *what* the message is, but rather to *whom* is it addressed. Suppose there are two possible recipients Alice and Bob with keys (N_A, e_A) and (N_B, e_B) respectively. Both N_A and N_B are distinct RSA moduli of approximately 1024-bits (more formally, they're integers in the range 0 to $2^{1025} - 1$).

Part 1 (10 points). A sender is transmitting many different ciphertexts to only one of those parties. Assume that the adversary can observe an unlimited number of these. If the attacker knows the two public keys, is there a simple technique he can use to determine which of the two parties the ciphertexts are destined for? (Hint: it might take many, many ciphertexts.)

Part 2 (10 points). Recall that the OAEP padding function randomizes the RSA encryption process. Thus, even when encrypting the same message, the value C obtained from the RSA encryption is likely to be quite different each time it is encrypted. Can you think of a way that the sender might use this property to thwart the technique you found above?

Problem 5: ZRTP and Implementation Review (30 points)

Note: for this question *only*, you are permitted to collaborate with one other student. If you do this, please indicate which student you worked with. They should also give your name.

The ZRTP protocol was designed to perform authenticated key exchange for voice-over-IP sessions. It's used in a variety of systems, including WhisperSystems' RedPhone software, which was recently acquired by Twitter. The ZRTP protocol is described in RFC6189.[1]

The Java-based source for RedPhone can be found on GitHub at `https://github.com/WhisperSystems/RedPhone`, and the ZRTP implementation is located in the subdirectory `src/org/thoughtcrime/crypto/zrtp`. You can browse the source via the web, or download it using `git`.

You have been hired by Twitter to analyze RedPhone's ZRTP specification and source code. Since this code may be used in conditions where operational secrecy means the difference between life or death, even theoretical vulnerabilities should be treated very seriously. Using the specification and source, answer the following questions:

1. Section 3.1.1 of the specification (Figure 2) provides an overview of a typical ZRTP interaction. At a high level, explain the purpose of (a) the Commit, DHPart1, DHPart2, and (b) Confirm1, Confirm2 and Conf2ACK messages.

2. ZRTP does not use certificates or a PKI to authenticate the Diffie-Hellman key exchange. As discussed in class, this could make the protocol vulnerable to a Man-in-the-Middle (MitM) attack in which the attacker alters protocol messages or replaces them with his own. Explain (at a high level) the mechanism that ZRTP uses to detect and prevent such attacks.

3. ZRTP detects tampering in the handshake messages by computing a value `total_hash` over the handshake messages, and folding this value into the calculation of the secret key. In principle, any tampering with the messages should create a key mismatch.

   In the RedPhone source code, identify the files/lines where `total_hash` is calculated.[2] Are any messages omitted from `total_hash`? Describe what impact (if any) this might have on RedPhone specifically?

4. Look at the code for computing a Short Authentication String (SAS). How many possible SAS strings are there?

[1] See `http://zfone.com/docs/ietf/rfc6189.html`.

[2] Don't forget that it's calculated on both sides of the connection!

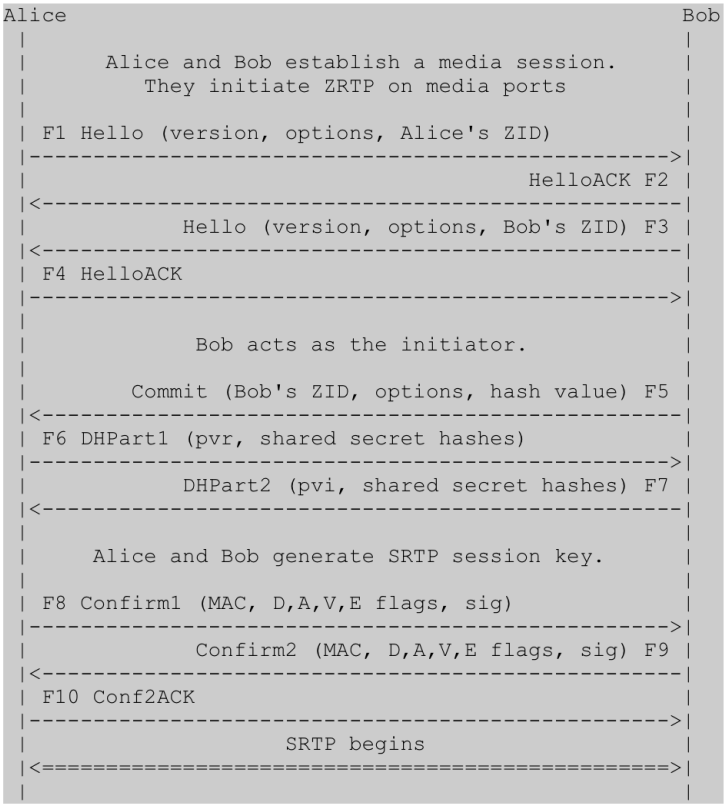

Figure 2: Sample ZRTP protocol run.