MCS 425 Final Exam Study Guide (Spring, 2008)

Chapters and Sections refer to the textbook for the course: Trappe and Washington, Introduction to Cryptography with Coding Theory, 2nd Ed. Sections with as asterisk will be covered only a limited way; please see the text below for more details.

Chap 1: Secs 1.1–1.2 and extra material covered in class.

You should be familiar with the basic definitions and concepts.

You should also be familiar with computations regarding the probability of various error patterns in the presence of white noise (binary code).

Chap 2: Not covered on final.

Chap 3: Secs 3.1–3.7 and 3.9

Definitions and properties.

Z_n = the integers mod n.

$\varphi(n)$ and $\Lambda(n)$, meaning and formulas for computing ($\varphi(n)$ especially) given factorization of n.

Pseudo-primes, Carmichael numbers and absolute pseudo-primes.

Major theoretical results.

Prime Number Theorem (sec 3.1.2).

Fermat's Little Theorem.

Euler's Theorem.

Existence of primitive roots in a prime (actually prime power) modulus.

The relationship between factoring an integer and finding all of its square roots.

Critical Algorithms.

Euclid's Extended Algorithm (gcd and lcm, modular inverses, solving other congruences).

Chinese Remainder Theorem.

Fast Modular Exponentiation.

The Rabin-Miller Primality Test (actually in Chap 6 of text).

Other important algorithms.

Determining if an integer a is a quadratic residue mod p (Is $a^{(p-1)/2} \equiv 1 \pmod{p}$?)

Finding square roots mod p (or mod p^e). You will need to know this only when $e = 1$ and $p \equiv 3 \pmod 4$, but please understand that the problem is *not* hard in any case.

"Easy" vs difficult computations.

You should know which calculations are relatively easy (say for 1000-bit integers) and which are extremely difficult, except in special cases, as far as we know. Difficult problems include:

i) factoring large integers (two or more distinct prime divisors).

ii) computing square roots mod n when we cannot factor n.

iii) computing $\varphi(n)$ where we cannot factor n.

iv) computing discrete logarithms in a prime modulus, and the (possibly simpler) Diffie-Hellman computational problem (Given α, α^x, and α^y (mod p), compute α^{xy} (mod p).

v) Possibly finding a primitive element mod p if we cannot factor $p-1$. (As far as I know, there is no known way to do this efficiently.)

Chap 4: Secs 4.1–4.2 and 4.4*–4.5*

Components used to design cryptographic circuits

S-boxes
P-boxes (possibly expanding or contracting)
XOR-boxes
Shift boxes

Feistel Ciphers

How they work.
Relation between encryption and decryption subkeys.

DES (Data Encryption Standard)

You should know that

DES was an official standard for many years.
DES has block size 64, 16 rounds, effective key length 56, and subkey length 48.
The security in DES comes primarily from eight 6×4 S-boxes.
Except for the choice of too short a key length, DES appears to have been well designed.
The key length is too short, and today DES can be broken, although great effort is required.
Triple DES was adopted as a temporary fixup, and a new algorithm (Rijndael) was chosen as the Advanced Encryption Standard (AES) to replace DES.

You don't need to memorize any details about DES, except as noted above.

Modes of operation:

You should be somewhat familiar with the concept of modes of operation (specifically ECB, CBC, and CFB), but you don't need to memorize how they work.

Chap 5: Not covered.

Chap 6: Secs 6.1, 6.3, and 6.4*

The RSA algorithm: The most important public-key algorithm. How it works, and what its security rests on.

Primality testing via the Fermat, Euler, and Rabin-Miller algorithms, mentioned in 3 above. (The Solovay-Strassen test, good but inferior to Rabin-Miller, was not covered in class and will not be covered on the final.)

Factoring: You should be somewhat familiar with the quadratic sieve, and be able to apply it in simple cases.

Chap 7: Secs 7.1, 7.4, and 7.5

The discrete log problem: definition and presumed difficulty

The Diffie-Hellman key exchange algorithm

The ElGamal Encryption Algorithm.

Chap 8: Secs 8.1, 8.4*

Properties that a cryptographic (or one-way) hash function should have.

You do *not* need to memorize details of any hash function, such as SHA-1.

Birthday attacks on one-way hash functions and certain cryptographic algorithms.

(You should be aware that if each of r people choose an object from an n-element set, with replacement, the probability some pair of people choose the same object is very low if $r << \text{sqrt}(n)$ and very high if $r >> \text{sqrt}(n)$. Likewise, if we have two groups each containing r people, and each person in each group chooses one object from the set (with replacement), the same conclusion holds for the probability that some person in the second group chooses the same object as some person in the first group.)

Chap 9: Sec 9.1

You should know the purpose of a digital signature.

You should know how to digitally sign a message using RSA, either with or without using a one-way hash function. Digital signature with El Gamal will *not* be included.

Chap 18: Secs 18.1–18.2, 18.4

Definitions related to error-correcting codes

Calculating probabilities of error patterns for a communication line with white noise. Advantages of longer codes if error rate is low.

Linear codes, the generator and parity-check matrices, syndromes and syndrome decoding. You should know how to compute syndromes and apply syndrome decoding.