CSC 152 – Cryptography

Meetings: This course will have synchronous and asynchronous components. All synchronous activities will be scheduled during our allocated time: Tuesday and Thursday 12:40 - 4:10.

Final Exam Time: The final exam will occur during our final class meeting, Thursday July 8.

Instructor: Ted Krovetz. Email: tdk@csus.edu. Web: https://krovetz.net/csus. Office Hours: See https://krovetz.net/oh for information about office hours.

Texts: Ferguson, Schneier, Koho; *Cryptography Engineering*, Wiley Publishing, 2010. About $40 at Amazon.

Webpage: Canvas

Catalog Description: Introduction to design and analysis of cryptographic systems. Symmetric cryptography: Block ciphers and secure hash functions. Asymmetric cryptography: Key exchange and public-key systems. Authentication and encryption in an adversarial model. Simple cryptanalysis. Protocol design and analysis.

Prerequisite: CSC 60, CSC 130; and STAT 50 or ENGR 115. These courses provide C programming, data structures, algorithm analysis and probability.

Announcements, Questions and Feedback: This course does not use Canvas for course communication. Electronic communication will take place using Piazza. You will use Piazza to access announcements and ask questions. On Piazza, you should not post anything that gives away too much of a homework solution, because this deprives others of finding the solution themselves, but otherwise students are encouraged to answer each other's questions. Course feedback is also encouraged. If something about the course frustrates you, please let me know. Piazza allows anonymous posting, if you like.

You may access Piazza via Canvas or directly at https://piazza.com/csus/csc152.

Instructor Interaction: This course does not use Canvas for course communication. If you need to contact me for something personal, send me an email. If your question may be of general interest to the class, ask it on Piazza. Questions can be asked during live Zoom sessions too.

Zoom Netiquite: When attending a group Zoom meeting, keep your microphone on mute. If you wish to speak unmute and say "professor". Once I acknowledge you, speak and then go back to mute.

Homework: Homework will be assigned via Canvas. There are two types of homework, graded and ungraded. Graded homework will be mostly or entirely programming assignments and will directly effect your grade whereas ungraded homework will only effect your grade via the quiz and exam assessments.

Exams & Quizzes: Each module will have a quiz that you do twice: once alone and once in a group. There will be two midterms and one final exam. If an unforseen and unavoidable happening keeps you from a quiz or exam, contact me as soon as possible.

Grading: Each of the following will be worth 20% of your grade: Quizzes, graded homework, midterm 1, midterm 2, and final exam. Your weighted average will determine your grade: A for 90+, A- for 85+, B+ for 80+, B for 75+, B- for 70+, C+ for 65+, C for 60+, and C- for 55+.

Regrading: If you believe you lost points on some work even though your solution was correct, contact me, with a note describing your concern, within a week of when it was returned to the class.

Approximate Schedule of Topics: Approximately one-fifth of course time on each of the following:

1. **Permutation functions and C programming.** Manipulating memory via pointers, bitwise manipulation, file IO. Intro to OpenSSL. Constructing permutation functions: AXR and Feistel. Model security as indistinguishability from random, adversarial advantage. Implement large permutation (eg, Chacha core).
2. **Symmetric encryption.** Block and stream ciphers made from permutations (eg, AES, Chacha), modes of operation, introduction to security reductions. Reimplement large permutation using vectorization.
3. **Hashing and authentication.** Cryptographic and universal hashing. Applications, including authentication and authenticated encryption. Security definitions. Implement sponge construction with large permutation.
4. **Asymmetric cryptography and algorithms on large numbers.** RSA encryption and signatures. Prime and elliptic curve groups, and using each for encryption and key exchange. O(log n) algorithms for exponentiation, finding multiplicative inverses, and finding large random prime numbers.
5. **Cryptographic systems.** Example applications of cryptography. Possibilities include random generation, secure communication protocols, blockchain, cryptocurrencies, electronic voting.

Course Outcomes: Students completing this course will be able to

1. Apply cryptography to secure data;
2. Explain cryptographic primitives, algorithms and their security; and
3. Use a cryptographic software package to perform cryptographic tasks.