Crypto and Infosec – Semester Project **Due on APRIL 8, 2014**

This is a group project. Min. group size = 2 persons. Max group size = 4 persons.

> **"Coming together is a beginning.**
> **Staying together is progress.**
> **Working together is success."**
> **-- Henry Ford, Founder, Ford Motor Company**

Project Option #1 Implementation of a Symmetric Block Cipher

Implement one of the more straightforward block ciphers, such as RC5 or Tiny Encryption Algorithm (http://www.cl.cam.ac.uk/ftp/papers/djw-rmn/djw-rmn-tea.html). Verify that the implementation works, then perform some experiments, such as timing trials, exhaustive key search, tests versus known good answers, randomness checks, etc.

Grading: A grade of "A" will be given to a project report that adequately implements the following:

- Detailed written technical report
- Research that includes literature search
- Complete listings and output (as applicable) of any custom code developed for this project.
  - There must be a design plan for the code.
  - Code must be formatted and well-commented.
- Include all relevant references, figures tables, diagrams, etc.

Your can "delight the customer" (i.e., receive extra credit) through providing additional analysis, handsome reporting, etc.

Project Option #2: Old Skewl Firewall Implementation

The purpose of this project is to provide a prototype implementation of a firewall, based on RFC 1928 (SOCKS Protocol Version 5). The FireWall ToolKit (FWTK), which is a set of proxies which you can use to build your own firewall, can be used as a starting point. The firewall should handle forwarding of Telnet, HTTP, FTP, and SMTP traffic. This project requires some background in C coding and knowledge of UNIX/Linux security. URL: http://www.fwtk.org/ (may need to find a different copy somewhere else. There are many out there).

Project requirements:

1. Download the FWTK.
2. Develop a firewall using the toolkit.
3. Provide complete listings and output (as applicable) of any custom code developed for this project.
   a. There must be a design plan for the code.
   b. Code must be formatted and well-commented.
4. Demonstrate attacks against the firewall and record how well it protects against these.
5. Write up a report with your findings:
   a. Executive summary
   b. Materials and Methods
   c. Overview of Development
   d. Conclusions

Project Option #3 Implementation/proof of RSA

Purpose: To demonstrate understanding of RSA and its resistance to cryptanalytic attacks

Requirements: Implement RSA using C/C++. Show that a 200-digit (667-bit) RSA crypto-system can be implemented on a home computer. You will need to find two large (at least 100-digit) prime numbers p and q to generate the modulus for RSA. Also, show that a 50-digit (167-bit) key can be broken within a week. Since n is known to be the product of two prime numbers, factoring this is sufficient for cracking an RSA secret key. What are the average and worst-case times to factor some (e.g., a few thousand) large (e.g., 40-digit) integers? What makes the times better or worse? Determine how long key needs to be to make the crypto-system secure. You will need to implement the following functions:

- CreateKey - Generate RSA public and secret keys, using a 100-digit prime number generator.
- Encrypt - Take a plaintext perform RSA encrypt on it.
- Decrypt - Take a ciphertext and perform RSA decrypt on it.
- Factor - Implements a factoring heuristic to factor large (in excess of 50-digit) numbers.
- Grading: A grade of "A" will be given to a project report that adequately implements the following:
    1. Finding and multiplying two hundred-digit prime numbers to generate a 200-digit RSA public key and a corresponding secret key.
    2. Encrypting, using the public key, and decrypting, using the secret key, a sample message. To show that the algorithm is implemented properly, it is sufficient to show that E(D(m)) = D(E(m)) = m.
    3. Write a program for factoring large numbers using some factoring heuristic. Determine the average and worst-case times for factoring 40-digit keys. Tell what variables affected these times.
    4. Tell how secure the RSA public-key crypto-system is. That is, applying Moore's Law and extrapolating, determine how long RSA keys must be for crypto-systems of the future. What key size is needed to be safe from a hacker with a home computer? What key size is need to be safe from thousands of crackers working together over the Internet?
    5. Supply any other conclusions you can make from this exercise.
- Your can "delight the customer" (i.e., receive extra credit) through providing additional analysis, handsome reporting, etc.

Project Option #4 Implementation/Analysis of an Intrusion Detection System

Purpose: To demonstrate understanding of network traffic and intrusion detection.

Requirements: SNORT is an open-source Network Intrusion Detection System (NIDS). You will want to download it (See http://www.snort.org), install it, and run it in NIDS mode. Then, you should fine tune the rules over the course of a few weeks, attempt to launch some covert attacks on your system, analyze the data received form the IDS, develop a rule for a particular attack, and demonstrate that it works; and write up a report.

Your **Report** should include the following: 1. Overall summary of the project (1 page or less); 2) Description of materials and methods used (1 page); 3) Analysis of traffic data -- What were the predominant packet type(s)? What vulnerabilities were noted? What fine-tuning was needed? (1-2 pages); 4) Attach appendices of sample traffic data summary, or maybe screen shot graphics, properly *labeled and explained* (no more than 10 pages); 5) Explanation of covert attacks used: what they were, how you launched them, were they detected by SNORT? (1-2 pages); 6) Explanation of rule developed and how it performed (1 page); and finally, an overall summary of how SNORT performed, what it is lacking, what you learned, recommendations of enhancements. (1-2 pages).

Grading: A grade of "A" will be given to a SNORT NIDS project which properly implements the IDS, monitors it for several weeks (at least two), implements a new rule, and has a well-written report with all the required elements.

Extra credit: I give extra points for "delight the customer" type things. Extra nice reporting, going above and beyond, etc. are the types of things that impress me.

Project Option #5 Implementation/Analysis of an Exploit Framework

You will install, use, and understand the **Metasploit Framework**. You will write an exploit that can be launched/loaded via the framework. You will write up a report detailing your findings. The framework will ***only*** be used on and directed at a system **for which you have proper prior authorization**.

The Metasploit Framework is an advanced open-source platform for developing, testing, and using exploit code. The project initially started off as a portable network game and has evolved into a powerful tool for penetration testing, exploit development, and vulnerability research. The Framework was written in the RUBY scripting language and includes various components written in C, assembler, and Python.

Project requirements:

1) Download and install the Metasploit framework from http://www.metasploit.com
2) Write or adapt an exploit that can be used in/ launched from the framework and demonstrate its use.
3) Write up a report of your findings and analysis. The report will include the following:
    a. Overall executive summary (abstract) of the project (1 page);
    b. Description of materials and methods used [HW, OS, etc.] (1 page);
    c. Explanation of how Metasploit works in your words (not copied from the project web page)—include an overview of functions, command set, HMI, your analysis of the installation, use, and efficacy of the tool (4+ pages);
    d. An explanation of the exploit you developed and how to perform and how well it worked with screenshots or live demo for the instructor if applicable (3-4 pages);
    e. An overall summary of how Metasploit performed, what it is lacking, what you learned, and recommendations for enhancements of the tool. (1-2 pages).

Project Option #6 Implementation of a secure E-Commerce Web site

Requirements: Implement a secure purchase order system that allows a customer to securely enter a purchase request (i.e., with encryption between client and server), and securely routes it to a supervisor for signature and then to the purchasing department. [Or, an alternative application: A request for computer account, routed to academic advisor and then to Academic Computing.]

Your project needs to explain the type(s) of security chosen and why. I should be able to access your Web site URL and enter a purchase order, then receive a confirmation by email for the order. You should have some means for simulating the supervisory approval of the transaction (credit card verifier, encrypted email routed to supervisor, etc.)

Please include some high-level diagrams or data flow descriptions to aid in understanding of your design. If possible, the program should be in compilable form, written in HTML, C, C++, Java, Perl, other scripting languages as needed for the Web-based front end with adequate commenting. The project and/or build files should be included, along with a ReadMe for building the program. If the program requires multiple clients (i.e., a local and remote machine) to run, please indicate this.

Grading: A grade of "A" will be given to a project which contains a <u>**Report**</u> with an executive summary explaining the problem, the program, and the approach used. The report should also include output listings from the program, screen shots, diagrams, and/or any other data applicable to explaining the design approach and methods used.
Note: You need not "re-invent the wheel." This project can be implemented by integrating off-the-shelf components; in which case, you would still need to provide some front end user interface, a secure transaction facility, and secure communications. Perhaps you will use some scripting language to accomplish this. If you borrow someone else's script, make sure you properly attribute it.

Sample project components needed:

1. Internet access
2. IIS (or other) Web server set up, script files loaded
3. Web form (Index.html)

4. Script to parse form
5. Script to decrypt
6. Script to encrypt
7. Script to email for supervisor approval
8. Script to email confirmation to client?

Project Option #7 Development/Implementation of custom Steganography program

You will determine a way to hide a file of indeterminate type within another file (either audio such as MP3; video, such as MP4, AVI, or other; or image, such as JPG, GIF or other); in such a manner that it does not significantly alter the appearance of the primary file and secretly conceals the contents of the hidden file. The steg system should also implement an authentication system – i.e., a means to retrieve the file should only be available to those who have the proper authentication code (e.g., UserID and password). Finally, you should analyze your steg program to see how well it hides data without being noticeable, while not also significantly altering the quality of the sound or image file.

Grading: A grade of "A" will be given to a project report that adequately implements the technical requirements and all of the following:

- Detailed written technical report
- Research that includes literature search
- Complete listings and output (as applicable) of any custom code developed for this project.
    - There must be a design plan for the code.
    - Code must be formatted and well-commented.
- Include all relevant references, figures tables, diagrams, etc.
- Analysis of the steg embedment for security and file degradation.

Your can "delight the customer" (i.e., receive extra credit) through providing additional analysis, handsome reporting, etc.

Project Option #8 Development/Analysis of a Pseudorandom Number Generator (PRNG)

John von Neumann once cautioned about the misinterpretation of a PRNG as a truly random generator, and joked: "Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin."

However, you will develop a deterministic random bit generator – i.e., a program based on an algorithm for generating a sequence of numbers that approximates the properties of random numbers. Of course, the sequence is not truly random in that it is determined by an initial state based on a seed. You can base your PRNG on a popular scheme such as Blum Blum Shub, Linear Feedback Shift Register, Fortuna, or Mersenne Twister -- however, you must put your own "twist" on it and add your own modifications or methods for seed generation.

Then, you must take output from your PRNG program, and run statistical analysis (e.g., chi-square test, kappa test, other tests) to show how "good" your PRNG is.

Grading: A grade of "A" will be given to a project report that adequately implements the technical requirements and all of the following:

- Detailed written technical report
- Research that includes literature search
- Complete listings and output (as applicable) of any custom code developed for this project.
    - There must be a design plan for the code.
    - Code must be formatted and well-commented.
- Include all relevant references, figures tables, diagrams, etc.
- Analysis of the PRNG for statistical soundness in the following areas:
    - http://en.wikipedia.org/wiki/Pearson%27s_chi-squared_test
    - http://en.wikipedia.org/wiki/Friedman_test

Your can "delight the customer" (i.e., receive extra credit) through providing additional analysis (perhaps for cryptographically "strong" tests), handsome reporting, etc.