Massachusetts Institute of Technology
6.857: Network and Computer Security
Professor Ronald L. Rivest

Handout 4
September 12, 2002

Problem Set 1

This problem set is due on *Thursday, September 19, 2002* at the beginning of class. Late homeworks will *not* be accepted.

There is both an individual and group component to this problem set. Each member of a group will turn in his or her own solution to problem 1, but the group will turn in one solution collectively for problems 2 and 3. You are to work on the group problems in groups of three or four people. Group problems turned in by individuals, pairs, pentuples, etc. will not be accepted. Be sure that all group members can explain the solutions. See Handout 1 (*Course Information*) for our policy on collaboration. If you do not have a group, seek partners by emailing `6.857-students-public@mit.edu`.

Mark the top of each sheet with your name(s), 6.857, the problem set number and question, and the date. **Type up your solutions**[1] and be clear. **Each solution must begin on a new sheet of paper**. That is, you will turn in problems 1, 2 and 3 in separate piles of paper. Points may be deducted if your TA has problems understanding your solution.

An anonymous TA discovered in his undergraduate classes that by writing problem set solutions in LaTeX, he would receive a good grade even if his solution was wrong. We can't guarantee this feature in 6.857, but writing your solutions in LaTeX will make them more readable. For your convenience, the raw LaTeX version of this problem set is available on the 6.857 Web site.

[1] We will distribute our favorite solution for each problem to the class as the 'official' solution – this is your chance to become famous!

Problem 1-1. PGP

This is an individual problem set question. Each student must complete this problem independently. Submit part (a) via email. Submit parts (b) and (c) on paper.

Do you agree with Alma Whitten's "Why Johnny Can't Encrypt" paper? This assignment will help you form an opinion.

Locate and install a fresh version of PGP or GPG. There are versions for Unix flavors, Windows, and the Macintosh. See `http://web.mit.edu/network/pgp.html` for downloads. If for some reason you are not able to download from this site, `http://www.pgpi.org/` may be of use. Create a new public/private key pair for yourself; you may use an existing key pair if you already have one.

Find the PGP public keys for as many of the 6.857 staff as you can. Part of your assignment is figuring out how to locate PGP keys. Searching the Internet for PGP key servers may be of help. But beware; there may be fake keys out there....

(a) PGP baby steps

Send an email to `6.857-staff@mit.edu` with the subject "PGP is fun". Do not send the mail to staff individually. In the body of the message, (1) tell us what operating system and version of PGP you are using. (2) Show us the public keys you found of the 6.857 staff; PGP fingerprints are sufficient. (3) In a few sentences, explain why you do or do not believe that they do indeed belong to the 6.857 staff. If you do not trust a public key, explain what would convince you otherwise. Finally, (4) include as a MIME attachment an ASCII-armored version of your newly minted public key.

Your mail and attachment should be protected with PGP such that only the 6.857 staff can obtain the plaintext contents. You must also sign the mail with your private key. We will only accept your first message, so make sure to get it right the first time. Are you able to finish the assignment in fewer than 90 minutes as in Whitten's experiment? Remember to cite all your sources (books, friends...) according to the guidelines in Handout 1.

(b) Fingerprints

Explain (1) the purpose of a PGP public key fingerprint; what is it useful for? (2) What primitive discussed in lecture does a fingerprint rely on? (3) If you were to securely obtain a person's PGP fingerprint, what can you assume with a high probability? Limit your solution to two or three paragraphs. Turn in this solution on paper. Remember to cite all your sources.

(c) President who??

Find a PGP key for `president@whitehouse.gov` on a PGP key server. Based on your findings, explain one useful feature and one drawback of PGP key servers. Limit your answer to two paragraphs. Turn in this solution on paper. Remember to cite all your sources.

Problem 1-2. Hotel security

This is a group problem. The length limit is 2–3 pages.

The Singapore Ritz Hotel is considering installing new door locks on all of the hotel rooms. They wish to use the mag-stripe type, where individuals can be issued mag-stripe cards that may cause a particular door lock to open and/or change state. The lock is battery-powered, and has non-volatile memory and a small processor.

The door locks have no communication with the hotel computer system except through the mag-stripe cards. They have no wired or wireless communication capability at all.

The hotel computer system (with the main terminals at the registration desk, and a few others elsewhere) is in control of issuing the cards and updating its local database of hotel registrants, issued cards, etc.

Write a plausible "security policy" that could be used as a model for the hotel to show to the manufacturers who are bidding for this system (which includes both the door locks and relevant portions of the hotel computer system).

Be careful to focus on the security *policy* (what goals should be achieved) rather than on *mechanisms* (how the manufacturer should meet those goals). For example, you might say, "The door locks shall be reasonably resistant to physical tampering, including removal from the door." rather than saying how they are constructed and attached to the door. Similarly, you might say "An 'expired' guest key shall no longer function to operate a door." rather than saying how that is accomplished. Of course, certain notions of the implementation (such as the door lock having changeable state information) may need to be introduced, but you should try not to constrain the manufacturers more than is necessary.

Your policy is illustrative; it doesn't need to be "correct" in any sense. But it should attempt to be realistic, and introduce/discuss various "roles" (e.g. guests, cleaning staff, check-in clerk, sysadmin, etc.) that might need to be handled, and the policies that might be relevant to each role, as well as discuss the security issues relevant to the components of the computer system. Feel free to introduce new terms (e.g. an "expired guest card") and/or make assumptions as appropriate. Try to cover rare but important exceptional situations as well. The idea is that a manufacturer who can meet your specs should have a good chance of meeting the "real" security policy to be drafted by the Singapore Ritz Hotel staff.

Be imaginative, and have some fun with this...

Problem 1-3. Cookie security

This is a group problem.

After several years of secret research after graduating from MIT, Kevin Nguyen finally came up with a revolutionary block cipher, which he calls KNC (as in Kevin Nguyen's Code). This code takes as input a 64-bit plaintext input and a set $\mathcal{K}$ of four 64-bit secret keys to output a 64-bit ciphertext output. It is surprisingly cryptographically secure.

Following this tremendous breakthrough, Kevin Nguyen decided to market a KN Super Chip that can compute the KNC algorithm efficiently, along with the usual basic arithmetic and logic operations. To keep the costs down to a minimum, Kevin made the chip work only with 64-bit words. The set of secret keys is initialized by the user to any set of values to his or her choice, and is only used in the KNC algorithm, so that it never leaks.

Damien Merrin runs xorcisms.com, a site for online XORcisms of demonic code. Damien decides to integrate the new KN Super Chip on his existing Web servers, and to update the authentication scheme to take advantage of the features of the KN Super Chip. He assigns Thien-Loc Fu to design the new authentication scheme.

Following the best cookie recipe he knows[2], Thien-Loc decides to encode in the cookie the expiration date e and the username u as optional data, along with a digest $\delta(u, e)$. Having missed the warnings about homebrew security in lecture 2, he decides to build his own MAC using KNC. Remember that $\text{KNC}_{\mathcal{K}}$ has the signature $\{0,1\}^{64} \to \{0,1\}^{64}$.

Fortunately, by restricting the username u to be 8 characters or less, u can be encoded in 64 bits or less. By representing e as the number of seconds since January 1, 1970, e can also be encoded in 64 bits or less. When the username or expiration date is less than 64 bits, it is padded on the *left* with null bytes (0x00).

The Web site will allow you to create new accounts and set passwords for any username not already created. The Web site allows both printable and non-printable characters in usernames. The account login process can be modeled as a function α which takes as input a username and password to produce a cookie-based authenticator:

$$\alpha(u, password) \to \begin{cases} (u, e = t + \tau, d), & \text{if the password is correct for the username } u \\ & \text{where } t \text{ is the current server date/time} \\ & \text{and } \tau = 1 \text{ hour} \\ & \text{and } d = \delta(u, e) \text{ is to be specified later} \\ \text{INVALID}, & \text{otherwise} \end{cases}$$

After logging in and receiving a cookie-based authenticator, one can model the verification routine on the XORcism Web server as:

$$\omega(\text{cookie}) \to \begin{cases} \text{true}, & \text{if cookie can be unmarshalled to } (u, e, d) \\ & \text{and if } e > \text{the current time} \\ & \text{and } \delta(u, e) = d \\ \text{false}, & \text{otherwise} \end{cases}$$

Note that the Web server is robust and strong enough to handle *multiple* queries *simultaneously*. You can issue multiple queries at the exact same time and also expect the server to respond to the queries at the exact same time.

[2] `Dos and Don'ts of Client Authentication on the Web`

In the following, $\oplus$ denotes an exclusive-OR, $+$ denotes the addition mod 2^{64} and $-$ denotes the subtraction mod 2^{64} in 2's complement.

(a) Thien-Loc, trying to KISS[3], first proposes to Damien the following digest:

$$\delta = \mathrm{KNC}_{\mathcal{K}}(u \oplus e)$$

Explain using the notation above how an interrogative adversary can make an existential forgery for this user authentication scheme. That is, implement an adaptive chosen message attack. State your assumptions.

(b) Following Damien's remarks on the weakness of his first trial, Thien-Loc then proposes to Damien the following digest:

$$\delta = \mathrm{KNC}_{\mathcal{K}}(u) \oplus \mathrm{KNC}_{\mathcal{K}}(e)$$

Explain how an interrogative adversary can still make an existential forgery for this new user authentication scheme.

(c) Not anyhow discouraged by his numerous failures, Thien-Loc decides to refine the digest he previously proposed:

$$\delta = \mathrm{KNC}_{\mathcal{K}}(u + e) \oplus \mathrm{KNC}_{\mathcal{K}}(u - e)$$

Explain how an interrogative adversary can now make a selective forgery for this new user authentication scheme. Don't forget to state your assumptions.

(d) Eventually, Thien-Loc decides to use two KN Super Chips, with two different sets of keys for this last proposal:

$$\delta = \mathrm{KNC}_{\mathcal{K}_\infty}(u) \oplus \mathrm{KNC}_{\mathcal{K}_\in}(e)$$

Damien is finally convinced of the security of this scheme against an interrogative adversary.

Explain however how an eavesdropping adversary can impersonate any user whose communication to the server he eavesdrops. More precisely, explain how this adversary can mint at any time in the future valid authenticators for any user whose communication to the server he intercepts once.

Although this adversary cannot hijack any communication, you can suppose he is powerful enough to automatically query the XORcism Web server in response to intercepted communications.

[3] Keep It Simple, Stupid...

General remarks valid for all the questions:

For forging an authenticator for a user u, the better solutions will never have to query α with u.

Note that a forgery does not include the exact replay of an entire cookie previously returned by α. That is, if you ever query for $(u, e, d) = \alpha(u, \text{password})$, you cannot simply replay (u, e, d) as a forgery.

You can query the XORcism Web server a reasonable number of times. You can send to the XORcism Web server multiple queries at the same time.

An existential forgery is a forgery of an authenticator for some username, not necessarily of your choosing or having any meaning. A selective forgery is a forgery of an authenticator for any user of your choosing.

Hints:

Although you have no control over e (which is set by the server), e is rather easy to predict. Explain how.

Don't forget you can send multiple queries at the same time to the server!