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

Handout 15
October 23, 2003

Take-Home Midterm

You are to do this take-home midterm ***by yourself*** — you may use your own notes, books, libraries, the Internet, etc., but you should not consult anyone other than course staff (including by email) about this exam. We repeat: *no collaboration is allowed.*

If you need a problem clarified, email `6.857-tas@mit.edu` or ask in person. We may occasionally send clarifications or bug fixes to the `6.857-students-public@mit.edu` list, so please make sure you are on that list, and check your email frequently!

Give citations for any material (other than your lecture notes and class handouts) that you use. Keep in mind that citations are not always correct or sufficient for justification. It is best to rely on your own reasoning and the material presented in class.

This midterm is due *on paper,* in room 6-120 on *Thursday, October 30* at the beginning of class.

Your answers must be typed! Each problem answer must appear on separate sheets of paper. Mark the top of each sheet with your name, the course number (6.857), the problem number, and the date. **Answers must be typed and clear.** We have provided templates for LaTeX and Microsoft Word on the course website.

Grading and Late Policy: Each problem is worth 10 points, except where noted. Late midterms will not be accepted without prior approval. Midterms submitted via email will not be accepted without prior permission.

With the author's permission, we will distribute our favorite solution to each problem as the "official" solution – this is your chance to become famous! If you do not wish for your answers to be used as an official solution, or if you wish that it only be used anonymously, please note this on your write-up.

Problem M-1. Protocol Design

Here is a protocol used for mutual authentication in a private peer-to-peer network. All members of the network know a common secret key, K. When two peers connect to each other, they both want to verify that the other is a member of the network.

A uses K to encrypt a nonce n, and sends the ciphertext to B. B decrypts the nonce, adds one to it, and re-encrypts it, sending the ciphertext back to A. A then decrypts the nonce, and verifies that it is the correct value.

Schematically, the protocol looks like this:

$$A \rightarrow B \quad : \quad A, \{n\}_K \tag{1}$$
$$B \rightarrow A \quad : \quad B, \{n+1\}_K \tag{2}$$

(Recall $\{\cdot\}_K$ denotes symmetric encryption under key K. The particular mode of encryption, if relevant, will be specified below.)

Because the protocol is designed to be used in a peer-to-peer network, another instance of this protocol is run concurrently, but with the roles reversed (i.e., B chooses a nonce, and A decrypts it). There is no coordination between the messages of the two instances, other than the fact that the initiation of one instance stimulates the initiation of the other, after some delay.

(a) If one of the parties is controlled by an adversary (who does not know the key K), describe how it can authenticate itself with the other party (who does know K). Your attack should be independent of the particular mode of operation used in the implementation, and should involve only two parties. Describe a possible fix that does not change the contents of the messages.

Now consider a client-server model, in which the client C and server S share a secret key K, and S wants to authenticate C (but not vice-versa). Their identities (also called C and S) are both 64-bit numbers, and are publicly-known.

To perform the authentication, S appends a randomly-chosen 64-bit nonce n to its identity, encrypts the result, and sends the ciphertext to C. C decrypts the message, checks the identity of S, and increments the nonce. C then appends the (incremented) nonce to its own identity, encrypts, and sends the ciphertext back to S. Finally, S decrypts the message and verifies the identity of C and the correct value of the nonce. In contrast to the previous scenario, only one instance of the protocol is run at a time.

Schematically, the protocol looks like this:

$$S \to C \quad : \quad \{S \circ n\}_K \tag{1}$$
$$C \to S \quad : \quad \{C \circ (n+1)\}_K \tag{2}$$

(As usual, $\circ$ denotes concatenation.)

(b) Suppose the encryption is done under AES in CBC mode. (That is, an encryption of a 128-bit value m consists of a 128-bit IV, followed by $\text{AES}_K(m \oplus IV)$. Describe how an adversarial client (who does not know K) can successfully authenticate itself with the server, with some reasonable probability.

(c) Suggest a possible fix to the bug you found (the resulting protocol need not be totally secure; it simply has to fix the bug). If at all possible, encryption should remain under AES in CBC mode, and the nonce should remain secret to an eavesdropper. Feel free to modify the protocol, but try to make the smallest changes possible.

Problem M-2. SSL Certs

This problem takes an in-depth look at SSL certificates.

(a) Internet Explorer 6 on Windows XP comes with a VeriSign Class 4 Primary CA certificate installed as a trusted root certificate.

```
Signature algorithm:   SHA1-RSA
Issuer:                VeriSign Trust Network
                       (c) 1998 VeriSign, Inc. - For authorized use only
                       Class 4 Public Primary Certification Authority - G2
                       VeriSign, Inc.
                       US
Subject:               VeriSign Trust Network
                       (c) 1998 VeriSign, Inc. - For authorized use only
                       Class 4 Public Primary Certification Authority - G2
                       VeriSign, Inc.
                       US
Valid from:            Sunday, May 17, 1998 8:00:00 PM
Valid to:              Tuesday, August 01, 2028 7:59:59 PM
Public key:            RSA (1024 Bits)
Thumbprint algorithm:  SHA1
```

How do users of Internet Explorer know that they can trust this certificate? Discuss the security of this certificate today and over the certificate's intended lifespan.

(b) The Palm Store at `https://store.palm.com/checkout/index.jsp?process=login` is an SSL-enabled website. The SSL certificate has the following fields:

```
Signature algorithm:  SHA1-RSA
Issuer:               Equifax Secure Certificate Authority
                      Equifax
                      US
Subject:              store.palm.com
                      Domain Control Validated - Organization Not Validated
                      See www.geotrust.com/quickssl/cps (c)02
                      store.palm.com
                      Santa Clara
                      California
                      US
Valid from:           Monday, August 26, 2002 11:40:56 AM
Valid to:             Wednesday, September 08, 2004 11:40:56 AM
Public key:           RSA (1024 Bits)
Thumbprint algorithm: SHA1
```

This certificate was issued by Equifax Geotrust; a root certificate for Equifax Geotrust is pre-installed in Internet Explorer and Netscape Navigator.

Review the Certificate Practice Statement at `http://www.geotrust.com/quickssl/cps`.

How is this certificate different from the VeriSign certificate? What does "Domain Control Validated - Organization Not Validated" mean?

What is involved in getting a Equifax Geotrust certificate for `store.palm.com`? How could you get an Equifax Geotrust certificate for a computer in the `mit.edu` domain?

(c) MIT operates a certificate authority that is used, among other things, to certify the public keys of MIT's SSL-enabled websites. The public keys are all signed with a private key, whose corresponding public key is attached to a certificate available at `http://web.mit.edu/is/topics/certificates/`. On that page, there is a link to download the actual MIT CA key, at `http://bs.mit.edu/mitca.ca`. MIT instructs people to get this certificate and install it as a trusted root certificate before getting a personal certificate.

What technical mechanisms are you trusting when you install the MIT certificate? Name two attacks that you are not protected against.

(d) The MIT CA actually has several root certificates. One is used to certify SSL server public keys, another is used to certify SSL client public keys. Client certificates are issued to any student who has a valid Kerberos username, Kerberos password, and MIT student ID number.

One of the things that you can do with an SSL client certificate is access the Stellar online course management system. You can also access the MIT Libraries' proxy server, to get off-campus access to MIT-licensed materials.

What is an the advantage of having Stellar and the MIT Libraries use the MIT client certificate, rather than having them simply ask for each student's Kerberos username and password? What is a disadvantage?

Problem M-3. Patch Management

You are designing a patch management system for a Major Software Company. MSC, which might be Apple, Microsoft, Red Hat, or some other company, distributes operating systems that are used by desktop computers. The first version of this system has these characteristics:

- The desktop operating system automatically checks with MSC to see if new OS software is ready to be installed.
- When new updates are ready, they are automatically downloaded and installed.

- The system authenticates the website that has the updates, the updates themselves, and the computer initiating the download request.

(a) Why is it important for the security of the desktop computer to authenticate both the website and the software update? If it is not necessary, state why not.

(b) Why might it be important to MSC that the individual desktop computers be authenticated?

(c) This system is deployed and is quite successful. It is so successful, in fact, that third-party software developers want to be able to use this system for their own programs.

The Version 2 system requires that individual software vendors obtain some kind of "software publisher registration" from the Major Software Company. MSC is not willing to host notices of software updates on its web server.

Describe a way for the Version 2 system to allow Big Graphics Company to use the software update feature.

Problem M-4. Sharing Secrets with Deities

Hercules has some pretty dirty secrets (involving his wife, kids, and temporary insanity) in his diary that he'd like to keep under wraps, until such time as is appropriate for making awful B-movies about his many exploits.

He plans to share the contents of his diary with Zeus, Hera, Apollo, Dionysus, Athena, Poseidon, Aphrodite, ... (all n of the Greek Gods) so that any t of them can reconstruct it at the appropriate time. At first, he thinks of Shamir's secret sharing. However, this being ancient Greece, it's very difficult for him to do computation modulo an enormous prime (and he has some big secrets to keep!). Despair befalls him.

Then, in a true "Eureka!" moment, he remembers Krawczyk's scheme. But, alas — the Gods are all-powerful, and can crack symmetric encryption like the shell of a walnut. Again, despair.

However, Hercules remembers that most of the Gods are honorable, and decides that no more than $m \ll t$ of them would ever try to maliciously discover his secrets. Therefore, he only desires perfect secrecy against groups of at most m (it is acceptable for groups of more than m deities to learn some partial information). Can you help Hercules devise a secret-sharing scheme that will fulfill all his goals, without requiring the Sisyphean task of extra-large-modulus computation?

(a) Suppose the diary is b bits long. Devise a scheme that allows Hercules to share its contents, by giving each deity an (approximately) $\frac{b}{t-m}$-bit share. Arithmetic should be done modulo a prime p of length (approximately) $\frac{b}{t-m}$ bits. *Hint:* think about something "in-between" the Shamir and Krawczyk schemes.

(b) Describe how any t shares can be used to reconstruct the secret.

(c) Argue (as formally as you can) that any group of at most m deities learns nothing about the secret, in an information-theoretic sense.

Problem M-5. Authenticated Key Exchange

Recall the problem of *authenticated key exchange:* Alice and Bob want to generate a fresh session key such that only Alice and Bob know the key, and such that Alice believes she is talking to Bob and vice-versa. In order to do this, we presuppose a public key infrastructure: there is an authenticated public key available for every party.

Consider the following key exchange protocol (which is similar to, but slightly different from, the "unified" one shown in class): there is a large, publicly-known prime p and generator g. All arithmetic is done in the group $\mathbb{Z}_p^*$. Alice and Bob have secret keys x_A and x_B (respectively), and public keys $y_A = g^{x_A}$ and $y_B = g^{x_B}$ (respectively). To establish a session key, they do the following:

1. Both parties state their identities and send each other their public keys y_A and y_B.

2. Alice chooses a random r_A, $1 \leq r_A \leq p-1$, and sends $t_A = g^{r_A}$ to Bob.
3. Bob chooses a random r_B, $1 \leq r_B \leq p-1$, and sends $t_B = g^{r_B}$ to Alice.
4. They each compute the session key $K = g^{x_A r_B + x_B r_A}$.

(a) Explain how both Alice and Bob can compute K efficiently, using the information known to them individually.

(b) The protocol has a flaw: a malicious Bob can impersonate being Alice, to Alice (thereby letting Alice believe that she is talking to herself; this could be a security risk!). Here's how:

1. In step 1, Bob claims to be Alice, and sends y_A as his public key.
2. In step 2, Bob receives t_A, which was generated correctly by Alice.
3. In step 3, Bob chooses a random r_B as above, but sends some t_B value (different from g^{r_B}) to Alice.
4. In step 4, they each compute the session key K. Alice does so honestly (according to your solution to the previous part), but Bob may do so in a different way.

What value should Bob send in step 3, and how should he compute K in step 4, so that Alice and Bob compute the same key K? Remember: K must be efficiently computable by both parties (given what they individually know).

(c) Is the protocol forward-secure? Explain why you believe it is or is not.

Problem M-6. Academic Honesty [3 points]

Write a couple of sentences to testify that you have not collaborated with anyone on this midterm and that you have cited all your sources. Affix your signature to this statement.