CS255: Cryptography and Computer Security **Winter 2002**

Final Exam

Instructions
– Answer **four** of the following five problems. Do not answer more than four.
– The exam is open book and open notes.
– You have two hours.

Problem 1 Questions from all over.

a. Explain why reusing a stream cipher key for multiple messages is a bad idea.

b. Is the RSA permutation modulo a prime a one-way permutation? In other words, how hard is computing an e'th root in the group $\mathbb{Z}_p^*$ when $\gcd(e, \varphi(p)) = 1$? Either give an algorithm for computing e'th roots in $\mathbb{Z}_p^*$ or briefly explain why the problem is hard.

c. Should the IV in the Merkle-Damgard construction for collision resistant hash functions be randomized on every application of the hash function? Very briefly explain why it should or should not. (there is no need to reprove the Merkle-Damgard lemma).

d. Recall that CRC is a linear checksum. That is, when $M, \Delta \in \{0,1\}^n$ we have that $CRC(M \oplus \Delta) = CRC(M) \oplus F(\Delta)$ for some easy to compute function F. Consider the following MAC defined by $MAC_k(M) = [R,\ AES_k(R) \oplus CRC(M)]$ where a fresh R is chosen at random in $\{0,1\}^n$ for every message. Show that given one Msg/MAC pair an attacker can create the MAC on any message M' of her choice.

e. Give the best upper bound you can on the effective key length of double-DES, Triple-DES, and quadruple-DES. That is, given a small number of plaintext/ciphertext pairs state the running time of the best algorithm you can think of for recovering the secret key (no need to describe the algorithm). Also write down the space requirements for each of your algorithms. Note that quadruple-DES is the cipher defined by $C = DES_{k1}(DES_{k2}(DES_{k3}(DES_{k4}(M))))$.

Problem 2 Let g be an element of prime order q in $\mathbb{Z}_p^*$. Suppose there exists an efficient algorithm that computes the Diffie-Hellman function base g. More precisely, there is an efficient algorithm $\mathcal{A}$ such that $\mathcal{A}(g^x, g^y) = g^{xy}$ for all $x, y \in \{1, \ldots, q\}$. Let $h = g^\alpha$ for some $\alpha \in \{1, \ldots, q-1\}$. Our goal is to show that there is an efficient algorithm $\mathcal{D}$ that is able to compute the Diffie-Hellman function base h, i.e. $\mathcal{D}(h, h^x, h^y) = h^{xy}$.

a. Show that there is an efficient algorithm $\mathcal{B}$ that given α is able to compute the Diffie-Hellman function base h, i.e. $\mathcal{B}(h, \alpha, h^x, h^y) = h^{xy}$.
Algorithm $\mathcal{B}$ may use algorithm $\mathcal{A}$ as a subroutine.

b. Show that there is an efficient algorithm $\mathcal{C}$ such that $\mathcal{C}(g, g^\alpha) = g^{(\alpha^{-1})}$ for all $\alpha \in \{1, \ldots, q-1\}$.
Hint: Recall that $\alpha^{q-2} = \alpha^{-1} \bmod q$. Algorithm $\mathcal{C}$ uses algorithm $\mathcal{A}$ as a subroutine.

c. Show that there is an efficient algorithm $\mathcal{D}$ that given h is able to compute the Diffie-Hellman function base h, i.e. $\mathcal{D}(h, h^x, h^y) = h^{xy}$. Use part (b).

Problem 3 Recall that the UNIX `crypt` function is a hash function that only looks at the first eight bytes of the input message. For example, `crypt(helloworld)` returns the same value as `crypt(hellowor)`.

Some web sites use the following authentication method to authenticate users: (1) the user types in a user-id and a password P into his web browser, (2) the site, upon verification of the password P, computes $T = \texttt{crypt}(userid \| K)$ where $\|$ denotes string concatenation, and K is a k-byte site secret key ($k \leq 8$), (3) the site sends a cookie back to the user containing T, (4) the user can use T to authenticate himself to the site in future connections.

Show that by choosing clever user-id's (of varying length) an attacker can expose the site's secret key K in time approximately $128k$ after $k+1$ successful authentication requests. More concretely, the attacker types in a valid user-id and password and receives the corresponding T. He then types in another valid user-id and password and receives another T. Show that by collecting at most $k+1$ such T's the attacker is able to deduce the site's secret key K (we are assuming there are 128 possible values for each character in a string).

Problem 4 Birthday paradox

Let $x_1, \ldots, x_n$ be randomly sampled integers in the range $[1, B]$. The birthday paradox says that when $n = \lfloor \sqrt{B} \rfloor$ the probability that there is a collision (i.e. exists $i \neq j$ such that $x_i = x_j$) is constant (greater than 1/10). How many samples $x_1, \ldots, x_n$ do we need until the probability that we get k collisions is some non-zero constant?

Hint: define the indicator random variable $I_{j,k}$ to be 1 if $x_j = x_k$ and zero otherwise. Then the expected number of collisions is $\sum_{j,k=1}^{n} E[I_{j,k}]$.

Problem 5 Access control and file sharing using RSA. In this problem $N = pq$ is some RSA modulus. All arithmetic operations are done modulo N.

a. Suppose we have a file system containing n files. Let $e_1, \ldots, e_n$ be relatively prime integers that are also relatively prime to $\varphi(N)$, i.e. $\gcd(e_i, e_j) = \gcd(e_i, \varphi(N)) = 1$ for all $i \neq j$. The integers $e_1, \ldots, e_n$ are public. Let $R \in \mathbb{Z}_N^*$ and suppose each file F_i is encrypted using the key $key_i = R^{1/e_i}$.

Now, let $S \subseteq \{1, \ldots, n\}$ and set $b = \prod_{i \in S} e_i$. Suppose user u is given $K_u = R^{1/b}$. Show that user u can decrypt any file $i \in S$. That is, show how user u using K_u can compute any key key_i for $i \in S$.

This way, each user u_j can be given a key K_{u_j} enabling it to access those files to which it has access permission.

b. Next we need to show that using K_u user u cannot compute any key key_i for $i \notin S$. To do so we first consider a simpler problem. Let d_1, d_2 be two integers relatively prime to $\varphi(N)$ and relatively prime to each other. Suppose there is an efficient algorithm $\mathcal{A}$ such that $\mathcal{A}(R, R^{1/d_1}) = R^{1/d_2}$ for all $R \in \mathbb{Z}_N^*$. In other words, given the d_1'th root of $R \in \mathbb{Z}_N^*$ algorithm $\mathcal{A}$ is able to compute the d_2'th root of R. Show that there is an efficient algorithm $\mathcal{B}$ to compute d_2'th roots in $\mathbb{Z}_N^*$. That is, $\mathcal{B}(X) = X^{1/d_2}$ for all $X \in \mathbb{Z}_N^*$. Algorithm $\mathcal{B}$ uses $\mathcal{A}$ as a subroutine.

c. Show using part (b) that user u cannot obtain the key key_i for any $i \notin S$ assuming that computing e'th roots modulo N is hard for any e such that $\gcd(e, \varphi(N)) = 1$. (the contrapositive of this statement should follow from (b) directly).