

HW #3: Hash Functions, Message Authentication Code and Key Distribution

CS 392/681: Computer Security
Fall 2006

[100pts] DUE 10/09/2006 (midnight)

Problem 1 [10+10+10=30pts]

Alice and Bob share a (160-bit long) private key $K=(K1,K2)$ with each other. Alice sends a message m along with message authentication code $MAC=MAC(K,m)$ to Bob. Eve intercepts (m,MAC) . Is it possible for Eve to produce a forgery, (*in fewer than 2^{160} operations required for brute forcing the key K*), if $MAC(K,m)$ is constructed as follows ($H()$ denotes a iterated hash function (like SHA-1), with 160-bit long output)? In each case, explain your forgery attack, give the forged message(s), and the number of operations to perform the forgery.

- (1) $MAC(K,m) = H(K,m)$
- (2) $MAC(K,m) = H(m,K)$ (we discussed one attack on this in the class – I am looking for a faster attack than the one discussed)
- (3) $MAC(K,m) = H(K1,m,K2)$

Problem 2 [5+5+5+5pts]

- (1) Schnorr signature scheme was explained in the last lecture (look under “some questions” in the slides). Alice needs to sign (using Schnorr signature) two messages $m1, m2$. She chooses a random number $k1$ in Zq , computes $r1 = g^{k1} \text{ mod } p$, $c1 = H(m1,r1)$, and $s1 = k1 + c1x \text{ (mod } q)$, and outputs $(r1, s1)$. Alice becomes lazy and uses the same random number $k1$ (as random number generation is costly) while signing the message $m2$. She computes $c2 = H(m2,r1)$, and $s2 = k1 + c2x \text{ (mod } q)$, and outputs $(r2, s2)$. Is this secure? Explain why or why not. Note that (p,q,g) are discrete-log parameters and are publicly known. You don't need to worry about what $H()$ function is, just assume that it is a random function.
- (2) Bob downloaded a 30GB tar.gz file from Alice's server today. Bob needs to know if he downloaded the correct file and that there were no errors in the transmission. Unfortunately, Alice and Bob do not share any keys nor do they have a common CA. How can Bob ensure the correctness of the file? Alice and Bob know each other personally, and they are scheduled to meet in a couple of days.
- (3) Is $H(m1 \text{ xor } m2) = H(m1) \text{ xor } H(m2)$? Why or why not? Assume $H()$ is MD-5.

- (4) Does HMAC exhibit a complementation property, i.e., does $h = \text{HMAC}(K,m) \rightarrow h^c = \text{HMAC}(K^c,m^c)$? Why or why not? Assume $H()$ is MD-5.

Problem 3 [10+20pts]

In this exercise, you will get some hands on experience with hashing, signing/verification using RSA and HMAC.

- Use the same code that you used in the last exercise: downloadable from: <http://www.funet.fi/pub/crypt/cryptography/asymmetric/rsa/rsaref2.tar.gz>. Get familiarized with MD-5 hashing function, signing, and verification functions and the sample code.
 1. execute the key generation function to generate the public key (e,n) and private key d
 2. choose any large message M and execute the signing function to sign M using (e,n) , and obtain the signature S
 - compute the execution time for MD5 hashing
 - compute the execution time for signing (this includes hashing)
 3. execute the verification function to verify (M, S) using (e,n) .
 - compute the execution time (this includes hashing too)

Repeat (2) and (3) an appropriate number of times and give the average execution time for each cases above. To get the timing for MD-5, you need to iterate a large number of times to be able to record something. List the type and speed of the processor, and the memory (RAM) of the machine you execute the code on.

- I don't think the above code implements HMAC (please let me know if it does). Assuming it doesn't, you have to do so. It should be very easy – you have the code for MD-5, and the structure of HMAC was explained in the class. So implement the HMAC function that uses a random key of any length (length is user input). Time this function as before, for a key of length 160-bit, for messages as long as you used above (for RSA signing/verification). How fast is HMAC as compared to RSA signing/verification for authenticating messages of same sizes. How does HMAC timing compare to the timing of the underlying MD-5 hash?

Problem 4 [4+4+4+4+4=20pts]

We discussed public key cryptography, and “certification by a trusted authority” as a means of key distribution in public key cryptography. There is also an emerging paradigm called identity-based cryptography, which works as follows. Each user sets its public key as its “identity” ID (such as an email address); obtains a signature of a trusted authority on its public key and set this signature as its private key. The users can use the

public key and private key pairs for secure communication with each other (everyone needs to know the public key of the trusted authority to do so, as in the public-key cryptography).

Compare the identity-based cryptography with the public-key cryptography (say which one is better and explain why), in terms of the following:

1. Security of the “joining” process (which is the process of obtaining a certificate in public key crypto, and a signature in identity-based crypto)
2. Cost of key generation on the user
3. Level of trust on the trusted authority
4. Usability of secure communication (both for confidentiality and authentication)
5. Revocation