Practice Problem Set 2: Integrity (MACs & Signatures)

March 19, 2017

MAC security
The following is the security game for message authentication codes (MACs).

- The game master chooses a random $k$ to the MAC.
- The adversary has access to a $MAC_k()$ oracle, that computes MACs on messages of the adversary’s choice.
- The adversary has access to a $VER_k(,)$ oracle, that Verifies that a tag $t$ is a valid MAC on a message $m$; both $m$ and $t$ can be chosen by the adversary.
- The adversary wins if outputs $m^*, t^*$ such that $m^*$ has not been queried to the $MAC_k()$ oracle and $VER_k(m^*, t^*) = 1$.

We say the MAC is secure if no (polynomial time) adversary can win this game with probability better than about $\frac{1}{2^\ell}$, where $\ell$ is the length of the MAC tag.

Signature security
The following is the security game for digital signatures.

- The game master chooses a random asymmetric key $(PK, SK)$ for the signature and gives $PK$ to the adversary.
- The adversary has access to a $Sign_{SK}()$ oracle, that computes signatures on messages of the adversary’s choice.
- The adversary wins if outputs $m^*, \sigma^*$ such that $m^*$ has not been queried to the $Sign_{SK}()$ oracle and $VER_{PK}(m^*, \sigma^*) = 1$.

We say the digital signature is secure if no (polynomial time) adversary can win this game with non-negligible probability.

Questions.

Exercise 1. Show that $MD5(k||m)$ is not a secure MAC. That is, present an attack that allows the adversary to win the MAC security game described above.

(Hint: Recall the length extension attack from Lab 1.)
**Exercise 2.** On February 23, 2017, researchers announced that they found a collision in SHA1. The collision was two files $f_1$ and $f_2$ such that $SHA1(f_1) = SHA1(f_2)$. See shattered.io.

Consider PKCS #1 v1.5 RSA digital signatures. To sign a message $m$, the message is hashed and padded as shown below to obtain the padded value $p(m)$:

```
00 01 FF···FF 00 3021009060520E03021A05000414 XX···XX
k/8 − 38 bytes wide        ASN.1 “magic” bytes         SHA1(m) (20-bytes)
```

Then, the signature is

$$p(m)^d \mod N$$

where $N$ is the RSA modulus, $d$ is the secret RSA decryption exponent, and $e$ is the public encryption exponent. Thus, the public key is $(e, N)$ and the secret key is $(d, N)$.

Present an attack that proves that PKCS #1 v1.5 RSA is not a secure digital signatures when SHA1 is used as the hash function. You must use the two files $f_1$ and $f_2$ in your attack.

**Exercise 3.** Dr. Snakeoil markets a new product that he claims protects the integrity of messages.

This product requires Alice and Bob to share a secret key 128-bit key $k$ that they will use to authenticate every message they send.

Then, if Alice wants to send a message $m$ to Bob, she breaks the message $m$ up into blocks $m_1, m_2, ..., m_n$ and outputs the tag $t_1, t_2, ..., t_i, ..., t_n$ where each $t_i = HMAC_k(m_i)$.

Alice then sends $m_1, m_2, ..., m_n, t_1, t_2, ..., t_n$ to Bob.

1. Write down the verification algorithm for this scheme.

2. Prove that this scheme is not a secure MAC.

**Exercise 4.** (Key exchange). Consider the following diffie-helman key-exchange protocol. Recall that the shared key is $k = g^{xy}$, and that $SIG_A(m)$ is the (public-key) digital signature on message $m$ signed by the secret key of $A$. Suppose that $A$, $B$ and $E$ all know each other’s correct public keys.

After this protocol runs, Alice and Bob send each other messages encrypted and authenticated under the key $k$.

Suppose there is a man-in-the-middle adversary $E$ that can intercept, add, drop, and the modify the traffic that $A$ sends to $B$.

1. Suppose that Alice and Bob are running software that has the following implementation flaw: it forgets to validate digital signatures and just accepts any messages it receives as valid.

Show how Eve $E$ can launch an man-in-the-middle attack, where she can read any of the encrypted and authenticated messages that Alice sends Bob.

2. Suppose that Alice and Bob are running software that has the following implementation flaw: it forgets to validate digital signatures and just accepts any messages it receives as valid.
2. Now suppose \( E \) can launch an “identity misbinding attack” where she convinces \( B \) that he shares the key \( k = g^{xy} \) with \( E \), while convincing \( A \) that she shares \( k = g^{xy} \) with \( B \). Explain exactly how \( E \) does this. (What messages does she send, and to who?) [Note, with this attack, \( E \) doesn’t know \( k = g^{xy} \) but \( B \) considers anything sent by \( A \) as coming from \( E \)]

3. Give an example of a scenario where your identity misbinding attack might create problems.