CS 6431: Security and Privacy Technologies
Homework #4

Due: Before class on November 4, 2015

Instructors: Ari Juels, Tom Ristenpart, and Vitaly Shmatikov

Problem 1: Recall that one option for generating plausible honeywords is to use the passwords of other users in the same system as honeywords. This approach offers a nice, practical method to selecting honeywords likely to resemble those of the user (thus achieving a “flat” honeyword generation process). Let’s call such a system a secondhand honeyword system.

1. Consider a system with \( n \) users in which user \( i \) has true password \( P_i \). One might consider a secondhand honeyword scheme that takes all \( n \) passwords as sweetwords for every user, i.e., lets \( \{ P_j \}_{j=1}^{n} \) be the sweetwords for user \( i \). What is the benefit of this approach? What drawback(s) might it have?

2. In any secondhand honeyword system, an adversary that can observe the state of the password database / computer system (and not the honeychecker) prior to and immediately after the addition of a user can trivially deduce the true password of this user. In general, an adversary capable of periodically observing the password database in a secondhand honeyword system can learn information about true passwords from state changes. Suppose an attacker can periodically (say, once a week) observe the state of the password database in a large organization. How might you embellish or modify a secondhand honeyword scheme to minimize leakage of information about true passwords while still benefitting from the use of other users’ passwords as honeywords?

Problem 2: Consider the DTE-then-encrypt honey encryption construction (DTE encoding followed by the application of a conventional encryption scheme). Explain how an attacker capable of mounting a chosen-message attack, i.e., obtaining
a ciphertext on a message $M$ of her choice, might feasibly recover a (low-entropy) encryption key $K$ used in such a scheme. For simplicity, you may assume very large (e.g., of size $> 2^{256}$) seed and message spaces.

**Problem 3:** Recall that padding oracle attacks arise when decryption leaks information about unauthenticated values.

1. Explain the role of the number 13 in the Lucky 13 paper.

2. Bellare and Namprempre in a paper in 2000 argue that one should build symmetric encryption via the encrypt-then-MAC construction: one first encrypts using a mode of operation such as CBC, and then applies a MAC to the resulting ciphertext using, e.g., HMAC. So a ciphertext is $IV||C||T$ where $IV$ is the random initialization vector of CBC mode, $C$ is the remaining portion of the CBC ciphertext, and $T$ is the HMAC output. Here $\|$ denotes concatenation of strings.

To build a secure channel out of this encrypt-then-MAC using CBC and HMAC, suppose we encode a plaintext message $M$ of length $\ell$ bytes by $\langle \ell \rangle_4 || M || \langle P \rangle_4^P$ where:

- $\langle x \rangle_m$ means encode in some canonical way the numerical value $x$ as an $m$ byte string.
- $\|$ denotes concatenation of byte strings
- $P$ is the number of bytes needed to ensure that the encoded string is a multiple of the block-size $n$ (in bytes) of the block-cipher underlying CBC. This is the same padding mechanism as used in TLS: so for example, if $\ell = 10$ and $n = 16$ bytes, then one appends the padding string “01 01”.

A server receiving a streaming ciphertext sent over TCP proceeds as follows. Decrypt the first $2n$ bytes which here will be the IV of CBC mode and the first ciphertext block. Then, one decodes the first 4 bytes of plaintext to get the length $\ell$ of the plaintext, determines the number of further bytes of ciphertext blocks and tag bytes, and then proceeds by checking the tag (recomputing HMAC over $IV||C$ and comparing with the received tag) and then ultimately decrypting the rest of the CBC ciphertext and releasing it to the application. If the MAC tag fails, the server sends an error message to the client.
Describe an attack against this scheme that recovers some amount of plaintext from a target ciphertext without violating the security of the block cipher or MAC underlying the mechanism.

Discuss the implications of your attack — why do Bellare and Nampempre’s about the strong authenticated-encryption security of Encrypt-then-MAC not apply to the scheme?