Network Security (Spring 2022) Due: Thursday, Feb 24 (11:59 pm)

Assignment II

Aniket Kate Purdue University

Please read the questions carefully. There are 45 points to earn in total.
Your answers **must** be typed and uploaded as a PDF document on gradescope. **Only** answer one question/subquestion per page; i.e., start every (sub)question on a different page. If you answer every question (including bonus questions), your written part submission pdf must include at least 13 pages. Include the question number (e.g., **Question 1.a** or **Question 4**) in a bigger font at the top of the page. While submitting on gradescope, use gradescope's answer assignment tool to assign one or more pages to every question. Include and assign an empty page for all unanswered questions.
There is *no* programming part in this assignment.

Question 1: Brute-force Attacks [2 pt, 1 page]

Assume that a powerful organization is doing an exhaustive key search attack and can check 2^{60} keys per second. How long would it take for the organization to search through all keys while attacking ciphers of the following key lengths: 80, 128? You must use year, month, day, and hour for time units. For simplicity, we assume that the attacker's capabilities do not improve over the time. Approximate, low-precision answers are sufficient.

Question 2: Security Principles [3+3 pt, 1 page]

Assume that an attacker has just discovered a buffer overflow vulnerability present in a slightly outdated version of SSH. By sending well-crafted input at the password prompt, the exploit allows the attacker to gain a root shell.

- The attacker searches for victims by connecting to TCP port 22 (default port for SSH) of the main web server for several sites whose servers he would like to compromise. Upon successfully connecting, the SSH server sends a string identifying what version it is running, which enables the attacker to determine whether the server has the vulnerability. After some time (and with fruitless results), the attacker realizes that the main web server of one of the sites, *www.attackme.com*, does not have an SSH server listening on port 22 at all. He "port scans" the target and finds that SSH is indeed running, but on a different, non-standard port, and that server is in fact running the vulnerable version of SSH! Identify the key relevant security principle missed by *www.attackme.com* the and justify it in a sentence or two. (2 point)
- The attacker sends the evil input and obtains a root shell. As the attacker pokes around the compromised system, he realizes that *www.attackme.com* is also used as a staging machine to deploy source code to several other web application servers. The deployment script copies the files to target machines via SSH. To keep the copying process quick and easy, the web application developer

team has enabled password-less access to the other servers, even though the site's security officers have set a policy that all access between machines must use two separate forms of authentication. The additional servers will accept any incoming SSH connection for user root that comes from *www.attackme.com* without requiring authentication. The attacker is delighted: his compromise of the staging server will now gain him root access to the several other servers as well! Identify a relevant security principle and justify it in a sentence or two. (2 point)

Question 3: Hash Function Applications [3+3 pt, 2 pages]

There are three desirable properties for cryptographic hash functions: Pre-image resistant (or onewayness), Second pre-image resistant, and Collision-resistant. For each of the following applications of hash functions, explain which of these three properties are needed and which are not.

a) Alice poses to Bob a tough math problem and claims she has solved it. Bob would like to try it himself, but would yet like to be sure that Alice is not bluffing. Therefore, Alice writes down her solution, prepends and appends some random bits to it, computes the cryptographic hash of the result and tells Bob the hash output (keeping the solution secret). This way, when Bob comes up with the solution himself a few days later, Alice can verify his solution but still be able to prove that she had a solution earlier.

b) A system administrator is concerned about possible breakins in her system. Therefore, she computes the hash of important system binaries and stores the hash values in a read-only file. A program periodically recomputes the hash values of the files containing the system binaries, and compares them to the stored values. A malicious user who is able to overwrite one of the "protected" files should not be able to change the file without detection.

Question 4: IP Fragmentation [6+3=9 pt, 2 pages]

IP fragmentation allows oversized IP packets to be split to fit on a smaller network. They are re-assembled on the destination machine.

As the packet fragments can overlap, we need to decide data acceptance policy for overlapping segments. In our operating system Zumba, if we find that the current fragment's `fp->offset` to be inside the end (`prev->end`) of a previous fragment (`prev`), then we align it such that old data is *not* replaced. The current segment can only start writing (decided by `ptr`) at the end of the previous segment.

This is ensured by the following `C` code segment.

```
if (prev != NULL && fp->offset < prev->end)
// overlapping fragments
{
     i = prev->end - fp->offset;
     fp->offset += i;    /* offset ptr into datagram */
     ptr += i;        /* ptr into fragment data */
     //shift to the end of the previous fragment
}
```

After the above structure, the `offset` and `ptr` values are passed to another function, where the length of the current fragment `fp` is computed as follows:

```
                    fp->len = fp->end - fp->offset;
```

a) What can go wrong with this? Demonstrate a possible vulnerability that can arise due to the above code?

b) How can you fix this vulnerability? Propose a fix to the problem.

Question 5: Symmetric Encryption Modes [4+8 pt, 2 pages]

PlzAttackMe Bank protects money-transfer orders digitally sent between branches, by encrypting them using AES in the ECB (electronic code book) mode for 128-bit blocks. Money-transfer orders (m) have the following structure: $m = \{f \| r \| t \| x \| y \| p\}$, where f,r are each 40 bits long and represent the payer (from) and the payee (recipient), t is a 48-bit field encoding the timestamp, x is a 64-bit field representing the amount, y is a 256-bit comment field defined by the payer and p is 64-bit parity fields, computed as the bitwise-XOR of the preceding 64-bit words. Transfer orders with incorrect parity, outdated or repeating time field, or unknown payer/payee are not processed. Notice that PlzAttackMe bank do not used any MAC for integrity protection, and instead it uses parity field.

a) An Man-in-the-middle adversary IStealMoney captures ciphertext message C (between two branches) containing money-transfer order of 1$ from Alice to his account. Assume that IStealMoney has tricked Alice into including a comment field y of his choice. Can IStealMoney cause transfer of larger amount to his account from Alice, and how?

b) Now assume that PlzAttackMe Bank was using CBC mode instead. Is it still possible for IStealMoney cause transfer of larger amount to his account from Alice? If yes, how?

Question 6: Public Key Cryptography (10 points, 5 pages)

In the following, we use

- k_A^- denotes user A's private/signing key,
- k_B^+ denotes user B's public (encryption) key,
- $\mathtt{sign}(m)_k$ for a message m signed with the private/signing key k. Here the notation $\mathtt{sign}(m)_k$ includes both the signed message m and a signature on it
- $\mathtt{enc}(m)_k$ for a public key encryption of message m with public key k

Additionally, n is a fresh nonce generated by B and m a message that A wants to authenticate with B. Do the following protocols ensure that whenever B authenticates m as coming from A, A has really started a session with B to authenticate m and the authentication request is fresh)? For each protocol, either argue how it ensures the security or show an attack (4 x 2 Points).

a)

$$\begin{tikzcd}
A & B \\
{} \arrow[r, "Hello"] & {} \\
{} & {} \arrow[l, "n"'] \\
{} \arrow[r, "\mathtt{sign}(B{,}m{,}n)_{k_A^-}"] & {}
\end{tikzcd}$$

b)

$$\begin{tikzcd}[column sep=huge]
A & B \\
{} \arrow[r, "Hello"] & {} \\
{} & {} \arrow[l, "n"'] \\
{} \arrow[r, "{m,\mathtt{sign}(B,n)_{k_A^-}}"] & {}
\end{tikzcd}$$

c)

$$\begin{tikzcd}[column sep=huge]
A & B \\
{} \arrow[r, "Hello"] & {} \\
{} & {} \arrow[l, "n"'] \\
{} \arrow[r, "{\mathtt{enc}(m)_{k_B^+},\mathtt{sign}(B,n)_{k_A^-}}"] & {}
\end{tikzcd}$$

d)

$$\begin{tikzcd}[column sep=huge]
A & B \\
{} \arrow[r, "Hello"] & {} \\
{} & {} \arrow[l, "n"'] \\
{} \arrow[r, "{\mathtt{enc}(m,n)_{k_B^+},\mathtt{sign}(B)_{k_A^-}}"] & {}
\end{tikzcd}$$

e)

$$\begin{tikzcd}[column sep=huge]
A & B \\
{} \arrow[r, "Hello"] & {} \\
{} & {} \arrow[l, "n"'] \\
{} \arrow[r, "{\mathtt{enc}(\mathtt{sign}(m,n)_{k_A^-})_{k_B^+}}"] & {}
\end{tikzcd}$$