System Security

George Cojocar

October 1, 2015

General remarks

Here are a few remarks that are always a good to remember when you are doing the exercises. These remarks apply to all the exercises but will just be included here as a friendly advise.

- Be specific. What is important? It is better to explain the "cause" of a problem in detail, than to only mention a "consequence".
- Answer all questions as thoroughly as you can.
- Google is your friend. Cite the sources you used in the bibliography boxes.
- Many spelling/formulation errors give a careless impression.

Collaboration between students is not allowed.

1 Cryptography Basics

This section will review some basic security concepts.

A. Explain the terms *Integrity*, *Confidentiality*, *Authentication*, *Authorization*, and *Denial of Service*.

Solution:

1. *Integrity:* A method of ensuring that information has not been altered by unauthorised or unknown means.
2. *Confidentiality:* The act of keeping something secret and private from all but those who are authorised to see it.
3. *Authentication:* The process of verifying the identity claim of an entity. It binds the principal to an identity. The authentication criteria can be:
   - Something an entity knows (e.g. password, PIN)
   - Something an entity has (e.g. key, card)
   - Something an entity is (e.g. bio-metric characteristic)
   - Where an entity is (e.g. location)
4. *Authorization:* The rights or permissions that are granted to entities to access system resources.

5. *Denial of Service:* It is an attack which aims to prevent legitimate users from accessing a specific service. Techniques used:
   - Resource starvation
   - Spoofing
   - Amplification
   - Reflector attack
   - Distributed attack

B. Why do we need both symmetric and asymmetric cryptography? Think of what primitives they enable, as well as a comparison between their execution times.

Solution:

There are several reasons why both encryption methods are in use. One aspect is the performance of encryption, as well as the key size. The key sizes are not comparable between the two approaches. A 128-bit symmetric key might be equivalent in strength to a 3000-bit public key. The speed of encryption is also significantly different, namely on a 1GHz processor, a symmetric encryption algorithm performs roughly 1,000,000 ops/s, while an asymmetric one (e.g.RSA) 100 signatures/s and 1000 verify/s. The other aspect is that the asymmetric encryption largely solves the key distribution problem, excepting the public key authenticity. Furthermore, the digital signature provided by asymmetric cryptography enables the receiver to verify the origin of the message, and it can convince a third party of origin as well (non-repudiation). The non-repudiation property can not be achieved using message authentication implemented with symmetric cryptography.

2 Proper usage of cryptography

A. Data compression and encryption are two techniques often used in storage systems and in communication over the network. What of the following is the best way to use the two? Think of efficiency, as well as security. Justify your answer and clearly state your assumptions.

(a) First compress, then encrypt

(b) First encrypt, then compress

(c) Both previous options achieve similar results

Solution:

The correct order is first compress and then encrypt (a). The most compression algorithms work by looking for similar patterns in order to reduce the amount of data. This means that if the data is randomised, there are no such patterns and the compression has no effect. This kind of randomised data is produced by a properly applied encryption.

B. It is often the case that data is both encrypted and authenticated, because encryption protects only against passive attackers, while authentication detects active attacks. What is the best way to use both of them? Read the options carefully and justify your answer.

(a) Authenticate-then-encrypt: first authenticate the cleartext, then concatenate the authenticator to the cleartext and encrypt this tuple

(b) Encrypt-then-authenticate: first encrypt the cleartext, then authenticate the ciphertext, and append the authenticator to the ciphertext

(c) Encrypt-and-authenticate: encrypt the cleartext, authenticate the cleartext, then append the authenticator to the ciphertext

Observation There exist primitives for authenticated encryption, which perform authentication and encryption in one traversal of the message. AES-GCM (Galois Counter Mode) is one such example. These primitives are not the purpose of this exercise.

Solution:

The proper approach is Encrypt-then-authenticate (b). This has the following properties:

- Ciphertext integrity. This ensures that the ciphertext is not altered in transit.
- Plaintext integrity.
- Protects against malleable cipher schemes by filtering out the modified ciphertext.
- The authentication code does not reveal any information about plaintext, if assuming the that the ciphertext appears random.

The other two methods do not provide integrity of ciphertext and do no protect against malleable cipher schemes. The Encrypt-and-authenticate may provide information about plaintext in the authentication code.

C. An example of a ciphertext-only attack would be a wireless device in a cafe capturing the packets of an HTTPS connection, afterwards trying to either find the plaintexts corresponding to the encrypted packets, or find the encryption key.

(a) Give similar real-world examples for the following attacks, and explain the purpose of the attacker: *Known plaintext*, *Chosen plaintext*, *Chosen ciphertext*, *Chosen cipher- and plaintext*.

Solution:

1. *Known plaintext:* The attacker knows at least one sample of ciphertext and its corresponding plaintext. This can be obtained by recording the encrypted communication and deriving the plaintext from some distribution that the adversary does not control. The purpose of this attack is to reveal the encryption key, for example if the XOR cipher is used, $key = plaintext \oplus ciphertext$.
2. *Chosen plaintext:* The attacker can specify his own plaintext and has access to a machine which can encrypt or sign it. The goal of this attack is to learn

the characteristics of the algorithm used for encryption and ultimately the key. A real example is the analysis of the Japanese naval code JN-25 by the US army during the World War II, Pacific battle of Midway. The US army sent fake messages containing the word "Midway". The messages were encrypted by the Japanese army using the code book JN-25 and then were communicated to Japan. The US army intercepted the communication and learned the ciphertext of the word "Midway".

3. *Chosen ciphertext:* The attacker has access to a machine which can decrypt arbitrary ciphertext and sends the plaintext back to him. The intention of the attacker is to determine the private key, even though the attacker is capable to decrypt the intercepted messages by using the decryption machine. In 1998, Bleichenbacher demonstrated a chosen-chiphertext attack against the PKCS 1 standard, which queried the decryption device (e.g. an SSL-equipped web server) on a large number of specifically crafted ciphertexts, which gradually revealed information to the attacker. In practical terms, this means that an SSL session key could be exposed in a reasonable amount of time, perhaps a day or less.
4. *Chosen chiphertext and plaintext:* The attacker can choose both the ciphertext and the plaintext, and also has the ability to query an encryption/decryption machine. Sometime the chosen-ciphertext attack is called like this, because the attacker gets to choose both plaintext values and ciphertext values.

(b) What is the weakest attack and what is the strongest attack? Justify your answer.

Solution:

Ciphertext only attack is the weakest since the adversary only knows the ciphertext, which was produced by the owner of the key. The strongest is chosen ciphertext and plaintext attack because the adversary has access to a decryption/encryption machine. This allows him to query this machine with arbitrary ciphertext or plaintext and to learn corresponding plaintext, respectively ciphertext.

3 Locking System

A car manufacturer wants to equip his cars with a new locking system which works in the following way: the system is composed of a small (tamper-proof) sender embedded in the car key and a receiver in the car. When the sender goes close to the receiver, it sends a radio frequency signal. The signal consists of 128-bit string which is the actual cryptographic key and which will be checked by the receiver. The range of the sender is about $20m$.

The cryptographic key length renders a brute force attack impossible and the sender can be considered tamper-proof.

(a) The system seems to be as secure as a classical lock. Do you see another possibility to break the security of the system?

Solution:

A replay attack is possible, namely the 128-bit string, representing that the cryptographic key can be recorded in one communication session between the sender and the receiver. The recorded value can be sent to the receiver in a later session without it to observe that the key is not fresh.

(b) What can be done to avoid the attack you just described?

Solution:

A challenge-response can be utilised together with the shared key as follows:

1. The key sends a predefined string, e.g. "open" to the car.
2. The car generates a "challenge" and sends it to the key.
3. The key encrypts the challenge with the shared cryptographic key of 128-bit and sends it back to the car. The car decrypts the message and verifies the challenge: if it is correct, it unlocks the doors.

(c) Would your solution solve the "relay attack" problem as described in Francillion A. et. al. in their work "Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars" (`http://eprint.iacr.org/2010/332.pdf`)?

Solution:

Yes. The challenge-response mechanism ensures the aliveness of both communication parties, and the shared key provides the authenticity. The man in the middle needs both, the challenge and the key in order to be successful, but the shared key is never sent out.

4 Side Channel Attacks Basics

This section will review some important concepts underlying the side channel attacks described in the lecture.

(a) What is a side channel attack?

Solution:

A side channel attack is a technique that allows the attacker to gain information about the encryption/decryption from the hardware of the cryptosystem, rather then exploiting the weakness in the cryptographic algorithms.

(b) What is the difference between simple and differential side channel (e.g., power, timing) analysis?

Solution:

The simple side channel attack exploits the side-channel output mainly depending on the performed operations (executed instructions). Typically, a single trace is used in a simple side channel attack analysis, and therefore the secret key can be directly read from the side channel trace. The side-channel information of the attacked instructions needs to be larger than the side-channel information of the unrelated instructions (the noise). When the noise level is too high, the simple side channel attack is not feasible. In that case, the differential side channel attack is more suitable because it is based on statistical methods. It exploits the correlation between the processed data and the side-channel output (e.g. different data consumes different power even if the operations are the same). Usually, many traces and statistical methods are used to deduce the possible secret key. For instance, it analyses the power consumption when normal non-cryptographic operations are being done, then further analyses during the cryptographic operations. The resulted statistical models are compared, in order to extract the noise from the signal.

(c) Computers and screens emit electromagnetic waves which can be measured and analyzed from a distance. This might allow an attacker to draw conclusions from the processed data. On which parameters does the maximum "attack–distance" from the receiver to the emission source depend?

Solution:

The maximum "attack–distance" depends on the intensity of the emitted radiation. This decreases with distance in a non-linear fashion as follows:

$S = \frac{I_i}{d^2}$

Where I_i is the initial intensity and d the distance.

The electromagnetic wave intensity depends on the energy density of electric and magnetic fields. The energy of these fields is equal for waves travelling through free space.

$S = c \cdot u$

c represents the speed of the electromagnetic wave in vacuum, and u represents the total energy density in an electromagnetic wave

5 Simple Power Analysis Attacks on RSA

(a) The "Square and Multiply" algorithm is commonly used to implement modular exponentiation. The algorithm computes the exponentiation by a series of squarings and multiplications. It processes the exponent bitwise, and for each bit, a squaring is executed. If the current bit of the exponent is $e_i = 1$, the intermediate result is multiplied with the base a.
What makes RSA implementations that use square and multiply (in its native form that we just described) vulnerable to a side channel attack?

Solution:

The "Square and Multiply" instructions have different power consumption signatures when the RSA plaintext m ($m \equiv \frac{c}{d} \pmod{n}$) is computed using "Square and Multiply" algorithm. The power consumption signature of each instruction can be easily visualised on an oscilloscope. In this way, the bits of private key exponent d can be recovered by observing when a squaring instruction is followed by a multiplication instruction, representing a bit of 1, respectively when a squaring instruction is followed by another squaring instruction, representing a bit of 0.

(b) Which platforms are most vulnerable to power-based side-channel attacks and why?

Solution:

The power-based side-channel attacks rely on the attacker ability to analyse the power consumption of the devices performing the cryptographic primitives. A successful attack requires that the power drawn by the device can be examined with the accuracy of its processor cycle. This vulnerability can be exploited in smart cards, which:

- use an external power supply
- run on a low frequency clock, facilitating the analysis
- use an external clock, which can be controlled by the attacker on his behalf to make the analysis possible
- do not incorporate isolation circuitry which would hide the power consumption per-cycle

6 Cache-timing Attack on AES

(a) What is the vulnerability exploited in the attack described at the lecture?

Solution:

The AES scrambles a 16-byte input n using a 16-byte key k by means of lookup tables, whose indexes are dependent on secret information $n \oplus k$. The lookup time of each such index is different. This leads to a correlation between timing and the secret information, which finally allows the recovering of the private key. This vulnerability is exploited by the attack presented in the lecture.

(b) On a general-purpose computer, would it be easy to overcome the problem while still having an efficient implementation? Justify your answer.

Solution:

Typically, it is very difficult to write constant-time high-speed AES software using S-boxes for general-purpose computers. Some of the challenges are:

- Skipping an operation is faster than doing it

- Cache is faster than DRAM
- L1 cache is faster than L2 cache
- Cache associativity is limited
- Code execution can be interrupted
- Stores can interfere with loads
- Cache-bank throughput is limited

Intel has recently introduced an AES instruction set in order to overcome these problems, beginning with 2010 Intel Core processor family (Westmere).

On the other hand, it would be relatively easy to write AES software that takes constant time with respect to the AES key and input. This can be achieved by replacing the two tables S and $S\prime$ with formulas using constant-time bit operations (e.g. xor, constant-distance, shift, etc). The resulting implementation would be resistant to timing attacks, but unfortunately much slower than the AES implementation using S-boxes.

(c) For a k byte key, how many messages would you require in theory for this attack? You may assume a noiseless/ideal measurement system.

Solution:

key size = k byte

input size = k byte

Assumption: 2^{22} messages are sent for each value of an input byte.

Then, the number of messages sent to recover one byte from the key is:

$2^8 \cdot 2^{22}$

Finally, the total number of messages to recover the entire key is:

$k \cdot 2^8 \cdot 2^{22} = k \cdot 2^{30}$

References

[1] Diffie, W. and Hellman, M.E., *New directions in cryptography*, IEEE Transactions on Information Theory, 1976.

[2] Srdjan Capkun, David Basin, *Information Security*, ETH, SS2015

[3] Bernhard Plattner, Thomas Duebendorfer, Stefan Frei, Adrian Perrig *Network Security*, ETH, AS2014

[4] M. Bellare and C. Namprempre, *Authenticated Encryption: Relations among notions and analysis of the generic composition paradigm*, Advances in Cryptology, ASIACRYPT 2000

[5] YongBin Zhou, DengGuo Feng, *Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testing*

[6] Doug Davis, *Electromagnetic Waves*, http://www.ux1.eiu.edu/~cfadd/1160/Ch22AC/EMWaves.html, 2002

[7] Manfred Aigner and Elisabeth Oswald, *Power Analysis Tutorial*, https://www.iaik.tugraz.at/content/research/implementation_attacks/introduction_to_impa/dpa_tutorial.pdf

[8] Daniel J. Bernstein, *Cache-timing attacks on AES*

[9] *Intel Advanced Encryption Standard(AES) New Instructions Set* https://software.intel.com/sites/default/files/article/165683/aes-wp-2012-09-22-v01.pdf