softScheck
we identify vulnerabilities others don't

info@softScheck.com
www.softScheck.com

The most important IT security basics for cryptography

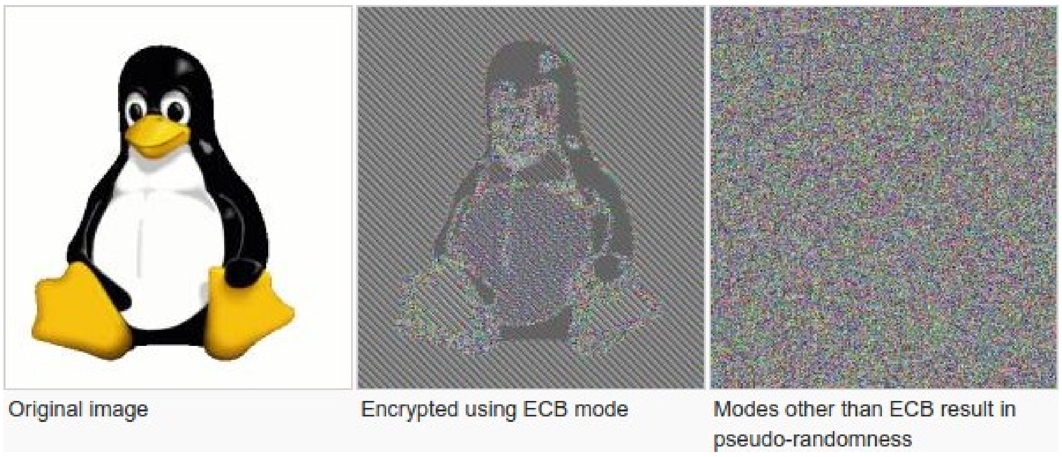
Cryptography is an important part of todays IT world. It is essential to ensure the protection goals of confidentiality, integrity and authenticity. Current cryptography algorithms are considered mathematically secure. Even if NSA, Google and Amazon join forces, it would take them billions of years to crack algorithms like AES-128. If those algorithms are so secure, why is it then that you always read about security vulnerabilities related to cryptography?

The problem mostly lies in the implementation and configuration. More specifically, the most common problems can be divided into the following categories:

Configuration and choice of parameters

Configuration errors or poor parameter selection can weaken a cryptosystem. Here are two examples of symmetric and asymmetric cryptosystems:

1. AES (Advanced Encryption Standard — symmetric cryptography) has several encryption modes. One of these modes is the Electronic Code Book (ECB) mode. ECB encrypts each block with the same key. This allows the recognition of patterns, here using the example of an image file.



2. There are many pitfalls with RSA (Rivest Shamir Adleman — asymmetric cryptography). Care should be taken to ensure that the message m to the power of e is larger than the modulus n, otherwise it has no effect. In this case, a ciphertext can be decrypted by simply taking the "e"-th root. Since e is part of the public key, e is usually known.

Use of weak cryptographic algorithms

Old cryptographic algorithms are often still used to be compatible with old devices and software. Mozilla and the BSI recommend the use of TLS 1.2 and 1.3. However, many still use TLS 1.0 or even SSLv3 for backward compatibility. But even TLS 1.0 can still offer an acceptable level of security and still great compatibility if the unsafe cipher suites are removed. Ciphers that use hash methods such as MD5, SHA1 or cryptographic algorithms such as DES, RC4 and AES in ECB mode should not be used. Under no circumstances may so-called null, anonymous or export ciphers be used. Null ciphers are only for test purposes, there is no encryption. Anonymous ciphers do not authenticate the server. An attacker could take advantage of this to force himself unnoticed into the connection (man-in-the-middle) and transmit his public keys to the other side. He can now read and change the content of the messages without noticing. Export ciphers have drastically limited key sizes and can be cracked in a matter of minutes with the appropriate hardware.

Bad pseudo random numbers

Modern symmetric cryptography builds on the encryption idea of the one-time pad. Since it is not practical to exchange key material in the length of the message with the recipient, pseudo random number generators (PRNG) are used. However, random numbers that are not sufficiently good can lead to the encryption being cracked, as happened in the Debian OpenSSH debacle. Even if good random numbers are generated on the system, corresponding functions still have to be used. Most programming languages offer several methods for generating random numbers. Only functions that access the entropy pool of the operating system are suitable for cryptography. In Java, for example, the java.security.SecureRandom function must be used instead of java.util.Random.

The probably best known example of unused random numbers is the PlayStation 3. To sign executable files, it uses the ECDSA (Elliptic Curve Digitial Signature Algorithm) based on elliptic curve cryptography. A nonce (random number, which should only be used once), k, is included in the calculation of the signature. However, this random number k was constant on the PlayStation 3. If an attacker manages to get two signatures with the same k, he can determine the value of k. If he knows k, he can calculate the private key. This allows the attacker to sign their own programs and, in the case of the PlayStation, circumvent the restrictions on the use of official software.

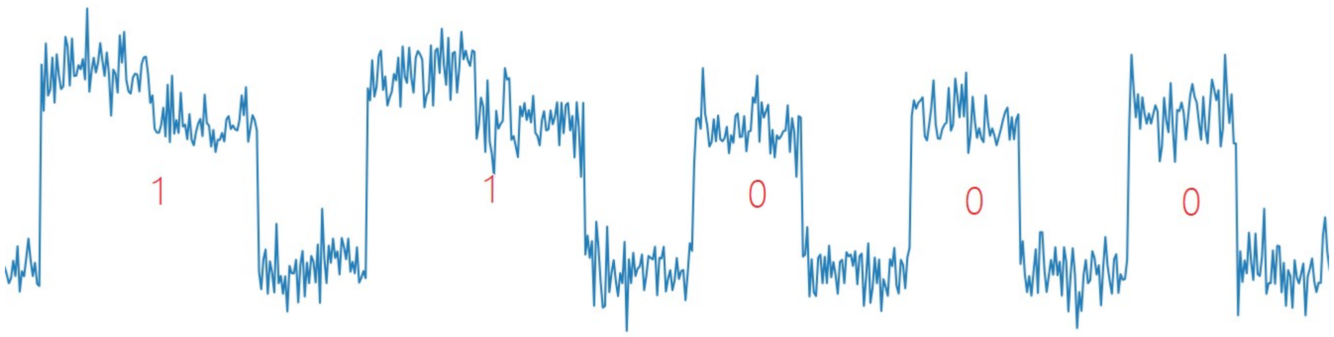
Unsuitable algorithm for the application

Another pitfall is the abundance of different algorithms that exist and from which to choose carefully. For example, passwords should be saved hashed and salted. There are hash algorithms such as Argon2 or Scrypt that have been specially developed for securely hashing passwords. However, due to their slow speed, they are unsuitable for hashing larger amounts of data.

Due to the large number of possible security vulnerabilities in the area of cryptography and the potentially great damage that can occur, it is advisable to consult an expert who examines the application of cryptography in the product.

Side channel attacks

If an attacker can physically access the hardware in which cryptographic operations take place, he can attempt side-channel attacks. In the event of a side channel attack, it is not the algorithm itself that is attacked, but its physical implementation: the attacker tries to draw conclusions about the key used or the plain text from data that the chip generates during cryptography operations. These data can be the duration (timing attack), the power consumption or the electromagnetic radiation during the operation. It is even possible to draw conclusions based on the noise that the chip generates. The following image file shows the power consumption for an exponential function.



In order to calculate this as quickly as possible, binary exponentiation is often used. Here the exponent is broken down into ones and zeros and then iterated through step by step. In this picture, the exponent 11000 is 24 in the decimal system. Every 1 is squared and multiplied. With a 0, however, only multiplied. This requires different amounts of electricity or takes different lengths of time. With a trained eye, the exponent can be derived from the power consumption analysis. For example, exponents play a major role at RSA. Textbook RSA encrypts a message m as follows: $c = m^e \mod n$. The decryption is done by $m = c^d \mod n$. An attacker can use the analysis to find out e or d. If he can carry out a current analysis during decryption, he gets the private key d and can decrypt the message.

Differential power analysis is even more effective. This is based on the fact that the processing of numbers uses different amounts of electricity. The more ones are included, the higher the power consumption. However, this effect is so minimal that the fluctuations make it impossible to recognize it. However, if the measurement is carried out several hundred times, the fluctuations can be calculated out. This makes it possible, for example, to recover the 16 byte key from AES.

The presented side channel attacks can be made relatively difficult by random dummy operations. There are also a number of side channel attacks based on error induction, as well as mitigations against them.

Conclusion

Despite secure cryptosystems, errors occur due to many things that should be observed, but are often unknown to developers. Under no circumstances should developers redesign and develop their own cryptography systems, as experience shows that these are often more susceptible to errors. Current, secure crypto algorithms must be used and implemented according to best practice guidelines. If you want to be on the safe side, it is advisable to have this tested later by external penetration testers and cryptographers.

Authors

Wilfried Kirsch has been working as a consultant in the field of IT security at softScheck GmbH since 2016. His professional focus is on IT security audits of networks, software and hardware products.

Prof. Dr. Hartmut Pohl is the managing partner of IT security consulting and responsible for tactical and strategic security consulting based on BSI-Grundschutz, ISO 27000 family, COBIT, NIST SP 800, ITIL etc. including forensics.