Chapter 6
Encryption Methods for QKD and RKD and XOR Operation

This chapter and https://cryptography.study/phys/modes introduce four new encryption methods and the one-time pad, all of which are based on the XOR operation. The new methods/modes limit the required key length and are therefore particularly interesting for QKD and RKD. Because hash functions are already used in QKD and RKD, no additional mathematical methods are required for data encryption. One of these four methods also generates a MAC for the security objectives of integrity and authenticity. Because these four methods are very simple in structure—they contain only the XOR function for data encryption (like the 100% secure one-time pad) and a hash algorithm/MAC—cryptanalysis is also easily possible.

The XOR function is not based on the laws of physics, but on Boolean algebra [WP–Boo].[1] Under certain circumstances, it can guarantee provable 100% security and is easy to implement in hardware. The XOR function plays an important role in cryptography. This is especially true

- in data encryption (security objective: confidentiality) when using the one-time pad, which consists solely of an XOR function, whereby each data bit is XORed with a key bit during encryption, resulting in a bit of encrypted data, and,
- in MAC calculation (security objectives: integrity and authenticity), where the XOR function is used in leading MACs such as CBC-MAC, CCM-MAC, and GCM-MAC.

6.1 Connection between the XOR Function and Physics

The XOR function is a logical operation of Boolean algebra. Boolean algebra is a mathematical model that can be used, among other things, to describe the behavior of digital circuits. Such circuits do not contain any software themselves, but are

[1] https://en.wikipedia.org/wiki/Boolean_algebra

© The Author(s) 2026

E. Piller and H. Schölnast, *Data Encryption at the Intersection of Mathematics and Physics*, SpringerBriefs in Information Security and Cryptography,
https://doi.org/10.1007/978-3-032-24764-3_6

purely hardware components and, as such, physical systems. They form the hardware basis of many technical applications. Boolean algebra is used directly in classical mechanics and electrical engineering. It is a fundamental part of the development of digital electronics, where it is applied as switching algebra. Boolean algebra is the basis for digital circuits and computer technology.

Logical operations of Boolean algebra such as AND (written as $a \wedge b$), OR (written as $a \vee b$), NOT (written as $\neg a$) can be implemented in practice by combining transistors. The XOR function (eXclusive OR, written as $a \oplus b$) is a logical operation that compares two binary inputs and only outputs 1 if exactly one of the two inputs is 1. If both inputs are the same (both 0 or both 1), the result is 0. The following formula shows how the XOR operation is defined and how it can be formed from the elementary Boolean functions AND, OR, and NOT.

$$s = x \oplus y = x\ XOR\ y = (\neg x \wedge y) \vee (x \wedge \neg y)$$

Table 6.1 Truth table for the XOR operation.

Figure 6.1 shows how the XOR operation can be constructed from circuits for elementary Boolean operations. Figure 6.2 is a black box representation for the same function.

Table 6.1 provides an overview of all possible value combinations

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

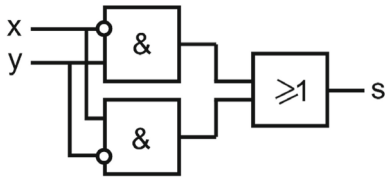

Fig. 6.1. XOR as the result of elementary operations

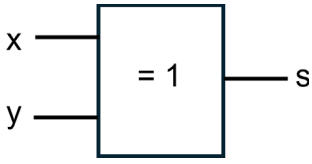

Fig. 6.2. Simplified circuit symbol for XOR

Table 6.2 shows an example of an XOR operation. Each bit from the unencrypted data is linked to the corresponding bit of the key. If both bits are the same, the value of the corresponding bit in the encrypted data is 0; if they are different, the value is 1

Unencrypted data	010011101010
Key	110011010111
Encrypted data	100000111101

6.2 One-Time Pad

The one-time pad was first mentioned in 1882 by Frank Miller and in 1885 by Joseph Mauborgne. However, it was only encrypted character by character. In 1917, Gilbert Vernam registered the one-time pad in its current form as a patent, which was commercialized by AT&T from the 1920s onward [Rij22][2]. As early as 1923, the German Foreign Ministry introduced the one-time pad in conjunction with physical distribution of the keys (see Chap. 5; MKD) for all diplomatic correspondence.

Vernam was also the first to mechanize the one-time pad. To do this, he used teletype paper tape containing randomly distributed character strings as key strips. The characters of the plaintext in the usual telegraph code were combined with those of the key strips using relays and the XOR function. Special devices were also built to generate these key strips on teletype paper tape. Another important factor here was the easier integration with telegraph traffic. After World War II, the Swiss company Crypto AG offered this solution worldwide as its own version of its C machines and became a market leader in this field. Although many different versions of these so-called "cipher machines" based on the one-time pad were developed and used worldwide, the literature is not very informative in this regard. The reason for this is that these cipher machines were mainly used in high-security areas, especially by secret services, and therefore the research and development results were usually not published.

With the first generations of computers and the representation of characters and numbers in bits, bitwise encryption with the XOR function finally became established [Bor12][3].

In bitwise encryption, each bit of the plaintext is XORed with a bit of the key. (See Table 6.1).

Table 6.2 Example of an XOR operation.

The 100% security of one-time pad encryption can be proven under these three conditions:

1. The key must be completely random, i.e., it must come from a non-deterministic random number generator.
2. The key must be at least as long as the unencrypted data. (If it is longer than the unencrypted data, only part of the key is used.)

[2] https://ciphermachinesandcryptology.com/papers/one_time_pad.pdf

[3] https://ieeexplore.ieee.org/abstract/document/6387923

3. New key bits must be used for each new encryption, i.e., the key must never be reused, not even parts of it.

The one-time pad can be used with all five physical methods / technologies described in this book if a sufficiently large amount of key material is available. If a maximum of a few megabytes of data per day is to be encrypted with a one-time pad, the key material that can be obtained at the same time with QKD devices available today can be used for this purpose, among other sources. However, at the beginning of 2026, the key rates that can actually be achieved by QKD devices will not be sufficient to encrypt significantly larger amounts of data generated daily with a one-time pad. In this case, key generation and distribution must be carried out using faster methods. Of course, established and standardized mathematical methods are available for this purpose, the more recent of which are even considered quantum computer-secure. However, this security classification is not based on rigorous mathematical proof, but "only" on the assessment of a commission of experts. If you also want to rule out the risk of error on the part of the commission of experts, you have to switch to physical methods, and then, due to the key rate, there is currently no way around MKD.

In practice, this means that QKD and RKD can only be used for telecommunications applications involving smaller amounts of data, and not for data storage. MKD is practically independent of the amount of data and the encryption mode, because MKD-capable storage media up to 16 TB are already available today.

In conjunction with a one-time pad, MKD-capable storage media for transporting cryptographic keys only became interesting with the application "secure and simple data backup" because this application ensured correspondingly low prices for storage media for a mass market with [BSI][4] [Gimbut13][5]. This marked the beginning of low-cost and easy-to-use MKD in conjunction with one-time pads and MKD-enabled storage media, followed by cryptography based exclusively on physical processes that covers the security mechanisms of confidentiality, integrity, and authenticity. Absolute cryptographic security, which is secure against computationally powerful adversaries and mathematical attack methods that are still unknown (unpublished) today, is thus also possible in the low-cost sector and can be implemented easily and in a user-friendly way in practice. But in conjunction with MKD, logistics (see Sect. 7.3) must ensure that sufficient key material is always available.

[4] https://www.bsi.bund.de/EN/Topics/Consumers/Information and recommendations/Cyber security recommendations/Backing up, encrypting, and deleting data/Data encryption/Software and hardware-based encryption/software and hardware-based encryption.html?nn = 921,724#doc921720bodyText2

[5] doi:10.1007/s11623-013-0212-0

6.3 Data Encryption for Data Storage

Data encryption using physical methods for data storage involves different conditions than those for telecommunications. The use of a one-time pad requires a very high number of key bits, because new key bits are required for encryption every time the data is changed (see Sect. 6.4). For example, MS Word constantly performs temporary saves, and user-initiated saves can also occur regularly. MS Word automatically saves 6 times per hour, although the user can reduce this interval to 10 minutes. Working with MS Word for five hours means at least 30 save operations, which, with average file sizes of, say, 200 kB, requires 6 MB of key material. This key material is required for a pure one-time pad. If this very high amount of key material is to be avoided, special encryption modes with a one-time pad are possible, though these do not constitute a pure one-time pad. They are presented in Sect. 6.4 and in the appendix https://cryptography.study/phys/modes and allow for a restriction on the key length. One of these new modes, called OTPM, also incorporates the security objectives of integrity and authenticity and additionally randomizes the entire encryption process. They are also considered highly secure, although no mathematical proof is known for this assessment. One of these new encryption methods is based on the XTS encryption mode, which is briefly introduced below. Since they are structurally simple—they contain only the XOR operation (like the 100% secure one-time pad) and a hash/MAC algorithm for data encryption—cryptanalysis is also easily possible.

6.3.1 XTS Mode

XTS (XEX-based Tweaked CodeBook mode with Ciphertext Stealing) [WP–XTS][6] [Ball12][7] is an encryption mode of operation developed for encrypting data in permanent storage devices such as hard disks, SSDs, USB sticks, virtual storage, etc. It was created by Phillip Rogaway [Rog04][8] with a few minor modifications. XTS mode uses two independent keys. XTS was approved by NIST for data storage protection and standardized in 2007 as quasi-standard IEEE 1619 [WP–IEE] [9] [IEEE18][10].

In XTS mode, the data to be encrypted, referred to below as m, is divided into blocks of 128 bits in length; the last block may be shorter. The variable j indicates the block number, starting with $j = 0$, i.e., the individual data blocks are m_j. The encrypted data blocks are called C_j.

The variable i indicates the location of the data block in the permanent memory (e.g., logical sector number) or the location of the data block within a file or a

[6] https://en.wikipedia.org/wiki/Disk_encryption_theory#XTS

[7] doi:10.1080/01611194.2012.635115

[8] doi:10.1007/978-3-540-30,539-2_2

[9] https://en.wikipedia.org/wiki/IEEE_Security_in_Storage_Working_Group

[10] https://standards.ieee.org/ieee/1619/11552/

database/table of a database, etc. However, linked to the location of the data block, it must also contain the name of the permanent memory or the file or database/table.

- α is a generator polynomial in $GF(2^{128})$, defined by the polynomial j.
- $\otimes$ represents the multiplication of two polynomials (α and AES-result) in the Galois field $GF(2^{128})$, where the multiplication is performed modulo $x^{128} + x^7 + x^2 + x + 1$
- Key_1 is the encryption key (256-bit) of the data m_j (plaintext block j).
- Key_2 is the second key (256-bit) for encrypting the variable i.

The left image shows the data encryption of the individual blocks. The right image shows the data encryption of the last two data blocks.

Figure 6.3 shows that the data is encrypted three times. First with the XOR function with the encrypted variable i (which is also linked to the value of variable j by Galois multiplication), then with the AES algorithm, and finally once again with the encrypted variable i. The encryption therefore corresponds to the following link:

$$C_j = AES_{Key_1}\left(m_j \; XOR \; \left(AES_{Key_2}(i) \otimes j\right)\right) XOR \; \left(AES_{Key_2}(i) \otimes j\right)$$

In XTS mode, identical plaintext blocks P_j result in different ciphertext blocks C_j because the block position is included in the encryption. This makes partial encryption, i.e., the encryption of individual data blocks independently of other data blocks, quick and easy to perform. Figure 6.3 illustrates the processes for the last two data blocks. This shows that no padding (filling the last data block to a length of 128 bits) is required for the last data block (ciphertext stealing). XTS mode also allows for easy parallelization in hardware, which is ideal for large amounts of data and

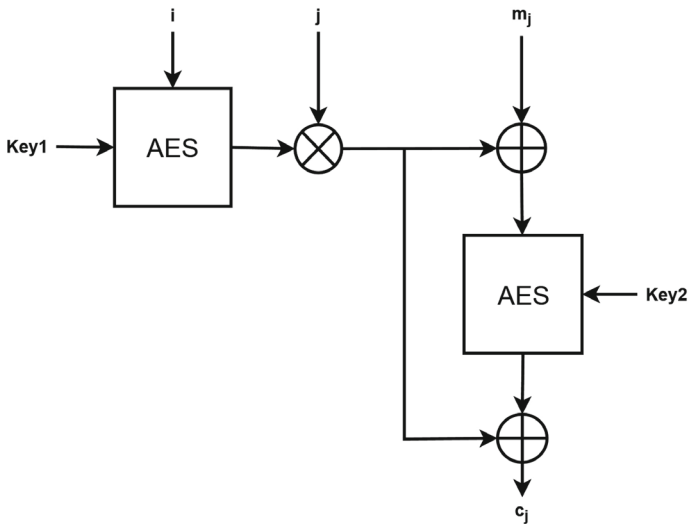

Fig. 6.3 XTS with one block

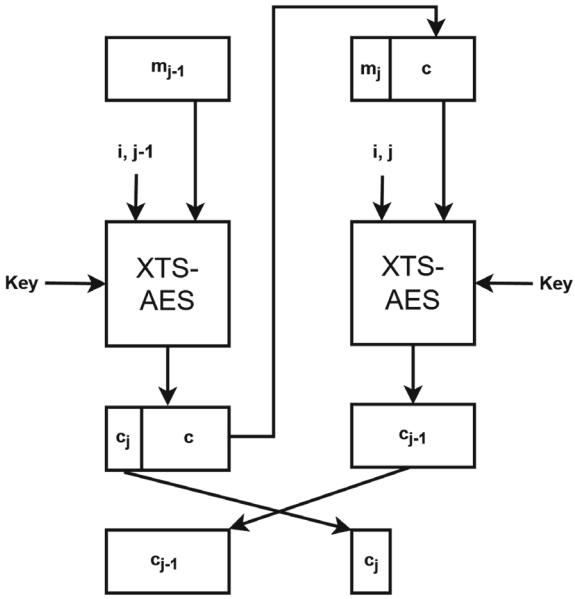

Fig. 6.4 XTS with the last two blocks

hardware acceleration (e.g., AES-NI). The double application of the AES encryption function requires double the key length and half the speed. XTS mode is used today, for example, in TrueCrypt, FreeBSD, OpenBSD softraid disk encryption software, Mac OSX Lion's FileVault, SPYRUS Hydra PC Digital Attaché, and Kingston DataTraveler.

6.4 Encryption Methods/Modes for QKD and RKD

6.4.1 OTPH Encryption Method

In the OTPH the upper section of Fig. 6.5 containing the two AES ciphers and the first XOR operation is omitted. For OTPH, only the area in Fig. 6.5 marked with a dotted line applies. The OTPH extends the One-Time Pad with a cyclic buffer (ring buffer) and a HMAC, based on RFC5869 Key Derivation Function (HKDF). The choice of HMAC is flexible in OTPH. We use the standardized HMAC-SHA3-512 (ISO/IEC 9707 und 10188). In OTPH, the non-deterministic—i.e., completely random—key is treated as a cyclic buffer. This means that the Key 3 (see Fig. 6.5) is considered a "cyclic buffer with modulo addressing" [WP–Cir],[11] [NSCL][12].

[11] https://en.wikipedia.org/wiki/Circular_buffer

[12] https://docs.frib.msu.edu/daq/newsite/nscldaq-10.2/c5.html

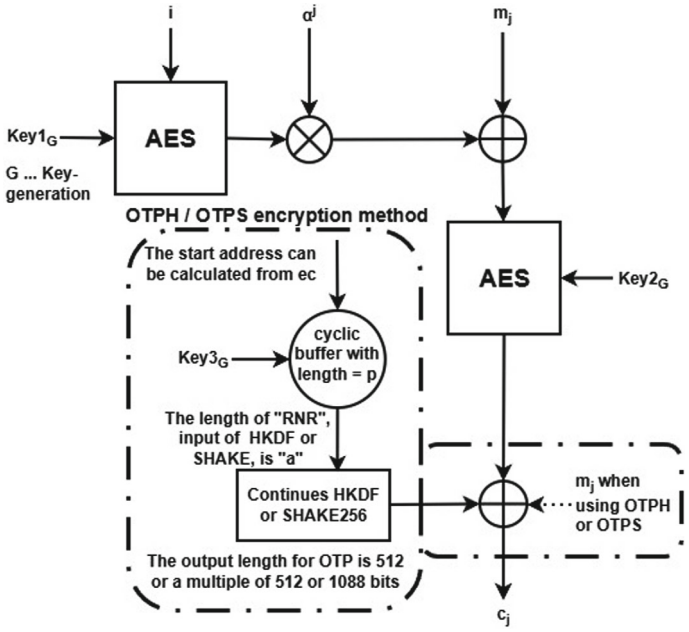

Fig. 6.5 OTPH, OTPS and XTSO

The key bits of Key 3 are read cyclically (in a loop), and the address pointer when reading Key3 is calculated modulo "p". The length of the cyclic buffer must be a prime number, hereinafter referred to as "p". In the first step, the key bits of Key 3 are read bit by bit with a spacing of "d=1" (d denotes bit spacing)—that is, the bits are read right next to each other—and in blocks (block length is "a") from the cyclic buffer, starting from an address derived from "ec" (encryption counter) using a simple algorithm (see website). These blocks are subsequently referred to as "RNR" and are extended using the HKDF-Expand function. This results in T_1, T_2, T_3, T_4, etc., each of which is 512 bits long due to SHA3-512. right next to each other.

The following function is used for this: $T_1 = HMAC_{RNR}(\text{„First Part“} \parallel 0x01)$, $T_2 = HMAC_{RNR}(T_1 \parallel \text{„Second Part“} \parallel 0x02)$, $T_3 = HMAC_{RNR}(T_2 \parallel \text{„Third Part“} \parallel 0x03)$, and so on. This results in the Key for the One-Time Pad, which provides the actual key bits for the XOR operation used in data encryption.

Key for One-Time Pad $= T_1 \parallel T_2 \parallel T_3 \parallel T_4 \parallel T_5 \parallel T_6 \parallel \ldots\ldots\ldots$

For the last data block, which may of course be shorter, correspondingly fewer bits from the Key for the One-Time Pad are used, and thus no padding is required. After the first "p" rounds, the value of "d" is changed, i.e., the interval between the bits read from the cyclic buffer. The new value of "d" is specified by the first three bytes in the cyclic buffer. However, "p" must be greater than 16,777,216; otherwise, two bytes must be used. After another "p" rounds with the new "d", a new "d" is determined, which is specified in the second three bytes in the cyclic buffer, and so on. For example, if "d=10," only every tenth bit is read sequentially from the cyclic

buffer. This means that the three bytes in the cyclic buffer specify the value of "d," and all "d" values used must be different. If this is not the case, the corresponding three-byte block is ignored and the next one is used. A further separation occurs when the values of "d" are not part of the key bits in the cyclic buffer, i.e., they are stored separately. The combination of HKDF and "d-variation" allows the key length—i.e., the number of key bits—to be significantly increased, e.g., from 100 KByte to TBytes.

The block length for the data encryption is 512 bits. No padding is required for the last block, meaning it may be shorter. If there is only one block that is shorter, there is no security issue. OTPH allows partial encryption, i.e., the encryption/decryption of individual data blocks, and identical unencrypted data blocks $m_i = m_j$ result in different encrypted data blocks $c_i \neq c_j$. These four characteristics are important advantages of the encryption method OTPH.

When performing cryptanalysis on OTPH, one can refer to the HKDF (RFC5869). This applies fully to the first round with "d=1". Because the value of "d" is also completely random in subsequent rounds, the selection of bits in the cyclic buffer is non-deterministic and there are no deterministic dependencies.

6.4.2 OTPS Encryption Method

For OTPS, only the area in Fig. 6.5 marked with a dotted line applies. OTPS (OTP with SHAKE256) is the same as OTPH, except that SHAKE256 is used instead of HKDF. SHAKE256 has been standardized as an SHA-3 variant. SHAKE256 has an extendable output function (XOF), where the length of the hash value is not fixed, but any number of hash data can be extracted. The input length is "a = 1088," and the output length is a multiple of 1088, e.g., 4352 bits, which corresponds to seventeen 256-bit data blocks in data encryption. That is, the key is extended as early as the first round of OTPS. If, after the end of the first round, there is still insufficient key material available, further rounds are performed as described above.

The block length for the data encryption is 512 bits. No padding is required for the last block, meaning it may be shorter. If there is only one block that is shorter, there is no security issue. OTPS allows partial encryption, i.e., the encryption/decryption of individual data blocks, and identical unencrypted data blocks $m_i = m_j$ result in different encrypted data blocks $c_i \neq c_j$. These four characteristics are important advantages of the encryption method OTPS. When performing cryptanalysis on OTPS, one can refer to the SHAKE256. This applies fully to the first round with "d=1". Because the value of "d" is also completely random in subsequent rounds, the selection of bits in the cyclic buffer is non-deterministic and there are no deterministic dependencies.

For every "d" where "$0 < d \leq p\text{-}1$" the following applies to OTPH and OTPS: During sequential reading of a constant amount "a" of bits from the cyclic buffer, "p" different input blocks "RNR" can be read. According to the rules of combinatorics, this results in "$p^2 - p$" different input blocks "RNR" which can be read from the cyclic

buffer. Key expansion is primarily based on the HKDF or SHAKE256. The rounds are intended only in the event that too few key bits are available for the One-Time Pad after the first round. Due to the theoretical maximum of "p^2 - p" rounds, in practice the OTPH and OTPS always provides a sufficient number of key bits if "p" is greater than one hundred thousand. This is a significant (quadratic) key expansion that, for example, increases the key length from the original 100 KByte to the TByte range when HKDF or SHAKE256 is included.

For OTPH and OTPS: An encryption counter "ec" is also required, which is stored for subsequent encryption operations. Using this counter "ec", a simple algorithm can determine the current bit position within the cyclic buffer and the parameters "r" (round counter) and "d" (read distance / spacing between two bits in the cyclic buffer) at any time (see https://cryptography.dot/phys/modes). The value "a" represents the length of the input to the HKDF or SHAKE256, while the value "ol" (output length) represents the length of the output of the HKDF or SHAKE256 (see Fig. 6.5). This output is used sequentially to encrypt the bits of the data blocks using the XOR operation. The value "ol" is a multiple of 512 (or 1088). Whenever all bits from this output have been used up—i.e., after "ol / 512" encrypted data blocks—the value "ec" (encryption counter) must be incremented by one, and "a" bits must be read from the cyclic buffer and processed with HKDF or SHAKE256.

When a new key (Key 3) of length "p" is loaded into the cyclic buffer, "ec" is reset to one ("ec = 1"). The parameters "a", "ol", and "p" must remain constant at least until a new key is loaded into the cyclic buffer. Due to the multiple use of Key 3, an HSM or another security module is required. Since modern HSMs have storage capacities in the GByte range, a specific (first) memory area within the HSM can be used to store the new key bits—such as those currently being generated via QKD or RKD—while the current key in the cyclic buffer is used for data encryption in a separate memory area. As soon as the first memory area in the HSM is filled with new key bits, a switch can be made, thereby reducing the number of rounds required.

6.4.3 XTSO Encryption Mode

XTSO (XTS with One-Time Pad, see Fig. 6.5) corresponds exactly to the XTS mode of operation (see Fig. 6.3 and 6.4), with the following difference: The second XOR function receives its input from an OTPH or OTPS, i.e., Key 3 with a cyclic buffer and HKDF or SHAKE256.

$$C_j = AES_{Key_1}\left(m_j \; XOR \; \left(AES_{Key_2}(i) \otimes j\right)\right) XOR \; OTPH/OTPSresult$$

The representation in Fig. 6.4 for the last two blocks therefore applies unchanged to this encryption mode as well, that means no padding is required, as with the XTS. Compared to XTS, the new XTSO encryption mode is much more secure due to the integration of the OTPH or OTPS encryption method.

Key generations in data storage application: The keys Key 1, Key 2, and Key 3 can be changed together, e.g., if a subject leaves a role-based authorization system and therefore loses read authorization (decryption authorization). These keys therefore also include the specification of the respective key generation G ($Key1_G$, $Key2_G$, $Key3_G$). For reasons of clarity, however, this has been deliberately omitted from the above description. Further details on OTPH, OTPS and XTSO can be requested from the first author or can be found at https://cryptography.study/phys/modes, which describes another encryption mode that includes a MAC for the security objectives of integrity and authenticity and is based on OTP, OTPH and OTPS.

References

[WP-Boo] Wikipedia contributors, "Boolean algebra," Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Boolean_algebra

[Rij22] D. Rijmenants, The complete guide to secure communications with the one time pad cipher, in *Self-Published Manuscript*, 8.1 edn., (2022). https://ciphermachinesandcryptology.com/papers/one_time_pad.pdf

[Bor12] S. Borowski, M. Lesniewicz, Modern usage of old one-time pad, in Military Communications and Information System Conference (IEEE, 2012), pp. 1–5. https://ieeexplore.ieee.org/abstract/document/6387923

[BSI] Bundesamt für Sicherheit in der Informationstechnik, "Verschlüsselung mit Software & Hardware; Hardware-based encryption," Online. https://www.bsi.bund.de/EN/Themen/Verbraucherinnen-und-Verbraucher/Informationen-und-Empfehlungen/Cyber-Sicherheitsempfehlungen/Daten-sichern-verschluesseln-und-loeschen/Datenverschluesselung/Soft-und-hardwaregestuetzte-Verschluesselung/soft-und-hardwaregestuetzte-verschluesselung.html?nn=921724#doc921720bodyText2

[Gimbut13] L. Gimbut, Datensicherheit: Was leisten externe verschlüsselte Festplatten? Datenschutz und Datensicherheit—DuD **37**(8), 526–529 (2013). https://doi.org/10.1007/s11623-013-0212-0

[WP-XTS] Wikipedia contributors, "Disk encryption theory; XTS," Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Disk_encryption_theory#XTS

[Ball12] Ball M.V, C. Guyot, J.P. Hughes, L. Martin, L. C. Noll, The XTS-AES disk encryption algorithm and the security of ciphertext stealing. Cryptologia (2012). https://doi.org/10.1080/01611194.2012.635115

[Rog04] P. Rogaway, Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC, in *International Conference on the Theory and Application of Cryptology and Information Security*. (Berlin, Heidelberg: Springer Berlin Heidelberg, 2004). https://doi.org/10.1007/978-3-540-30539-2_2

[WP-IEE] Wikipedia contributors, "IEEE Security in Storage Working Group; XTS," Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/IEEE_Security_in_Storage_Working_Group

[IEEE18] IEEE. IEEE Std 1619-2018: "IEEE Standard for Cryptographic Protection of Data on Block-Oriented Storage Devices. IEEE ndards Association (IEEE SA)" standard web page. (2018). https://standards.ieee.org/ieee/1619/11552/

[WP-Cir] Wikipedia contributors, "Circular buffer," Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Circular_buffer

[NSCL] Facility for Rare Isotope Beams (FRIB), Michigan State University. "NSCL DAQ Software Documentation (NSCLDAQ 10.2): Chapter 1 Introduction," no date. https://docs.frib.msu.edu/daq/newsite/nscldaq-10.2/c5.html

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.