On the Cryptographic Strength of Symmetric Ciphers Suitable for Power-Line Communications

Reiner Dojen[*]
Tom Coffey[**]

Department of Electronic and Computer Engineering
University of Limerick, Ireland

Abstract

Power-line communications (PLC) use publicly accessible power-lines for the transmission of data. The power-line should be viewed as a hostile channel requiring special security services, if secure communication is desired. This paper introduces cryptography and related services. The operation and characteristics of some symmetric block ciphers for PLC use are reviewed. These characteristics include security, speed and flexibility. Exhaustive key search is suggested as a first yardstick for the security of ciphers and a minimal key length to provide adequate security is proposed. Differential and linear cryptanalysis are presented as powerful attacks on the security of symmetric ciphers. The paper concludes with a tabulated list of potential symmetric ciphers for PLC applications.

Keywords: Power-Lines, Cryptography, Cryptanalysis, Keysearch, Cipher

1. Introduction

Power-line communication (PLC) uses publicly accessible power-lines for the transmission of data. The power-line should be viewed as a hostile channel and special security services are required, if secure communication is desired. Typical data security services for PLCs include data confidentiality, data integrity and authentication of the source and destination. These services can be implemented using cryptographic algorithms. Then the security of the services relies on the cryptographic strength of the algorithms used.

Cryptographic algorithms that are suitable for PLCs need to be strong, fast and flexible in terms of implementation. One group of algorithms that potentially satisfy these requirements are symmetric block ciphers. In recent years, a large number of such algorithms have been published. However, some of these algorithms have security flaws and must be considered insecure. Further, recent developments in cryptanalysis have resulted in new techniques for attacking cryptographic systems. Some algorithms that have previously been considered as secure, are susceptible to these new techniques and must be considered insecure.

A first yardstick for the security of a cipher is its strength against an exhaustive key search. Another way to evaluate the cryptographic strength of the ciphers is the application of cryptanalysis. This evaluation is a "destructive process". A cipher is regarded as broken if a technique is found that reveals the key faster than an exhaustive key search. Two powerful attacks to break a cipher are differential and linear cryptanalysis.

This paper introduces cryptography, which is the art of information hiding, and cryptanalysis, which is the art of revealing information. The operation of widely used symmetric block ciphers, as well as newly proposed ciphers, is reviewed. Their significant properties, like their cryptographic strength, speed and the flexibility, are analysed. Furthermore, an overview of exhaustive key search times is given and a minimum key length for block ciphers to ensure an adequate security is proposed. Differential and linear cryptanalysis, as well as some of their variations, are introduced and applied to the presented ciphers. The paper concludes with a tabulated list of potential symmetric ciphers for PLC applications.

2. Cryptography

Cryptography is the science of keeping information secure. A cryptographic cipher is an algorithm that transforms a message or *plaintext* into an unreadable form called ciphertext. Modern ciphers base their security on keeping the key secret, rather than the algorithm, which is assumed to be publicly known. There are two distinct categories of cipher. The first category, called symmetric ciphers, use the same key for encryption and decryption. This key must be agreed upon by the participating parties before messages can be exchanged. The

[*] E-Mail: reiner.dojen@ul.ie
[**] E-Mail: tom.coffey@ul.ie

second category, called public key ciphers, use two distinct keys for each party. One of these keys is kept private, i.e. it is not shared, while the other is made publicly available. Both keys are complementary in that any plaintext encrypted with one key can only be decrypted with the other key.

Cryptography can be used to provide the following services:

- **Data Confidentiality** is the service of keeping information hidden, i.e. it ensures that only authorised entities can access the information. This can be achieved by encrypting all messages before they are transmitted. Then the information cannot be accessed without the key, although the encrypted message may be available.
- **Data Integrity** is the service of keeping information in its original state, i.e. it ensures that the transmitted information arrives at its destination without any modifications, such as: insertion, deletion or substitution, since it left the source. This can be ensured by using a message authentication code (MAC).
- **Entity Authentication** is the service of identifying entities, i.e. it ensures that parties cannot be impersonated by other parties. This service can be implemented by a challenge-respond scheme.
- **Data Origin Authentication (Non-Repudiation)** is the service of identifying the source of information. It ensures that parties cannot deny being the source of information and hence can be held responsible for their actions/messages. This can be implemented by using a public-key scheme.

3. Symmetric Block Ciphers

Block ciphers operate on blocks of the plaintext or ciphertext. The size of these blocks is usually between 64 and 256 bits. The operation on each block is independent of the operation on all other blocks. Hence, multiple devices can operate in parallel to speed up the encryption/decryption process. Stream ciphers on the other hand, operate on a stream of small pieces of the plaintext or ciphertext. The size of these pieces is usually a bit, a byte or a word. Further, the operation depends not only on the key, but also on the previous operations. Stream ciphers need to be synchronised.

3.1 Feistel Cipher

Feistel ciphers are block ciphers based on the principles of *diffusion* and *confusion*. Both operations are alternately applied a number of times to the plaintext. Diffusion is used to dissipate the statistical structure of the plaintext into the long-range statistics of the ciphertext. This means, that any ciphertext bit is affected by many plaintext bits. In modern ciphers, diffusion is often implemented by applying some permutation function repeatedly. Confusion is used to hide the relationship between the ciphertext and the used secret key. To obtain confusion modern ciphers use complex substitution functions.

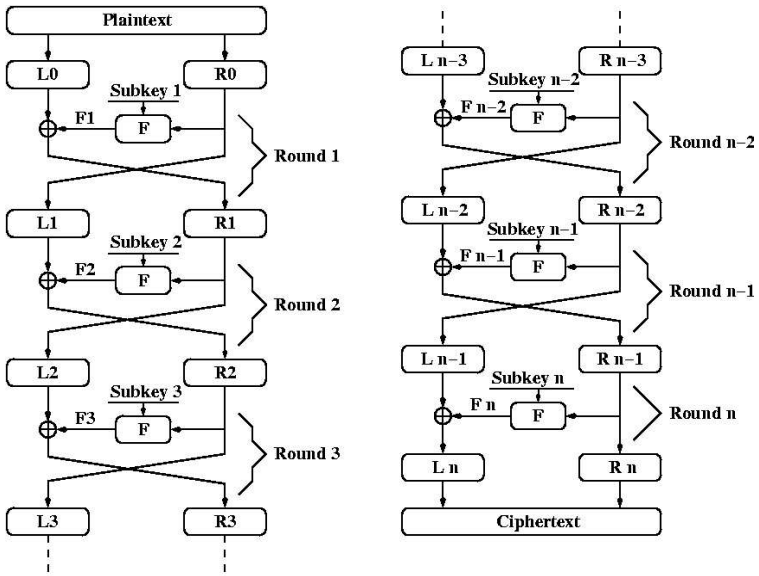

Figure 1: Feistel cipher structure

The structure of a feistel cipher is as follows (cf. Figure 1). Firstly, the plaintext is divided into two halves of equal size, R_0 and L_0. The right half R_0 is input, together with a sub-key (obtained from the overall key by some schedule algorithm), to the function F. The result of the function is then XORed with the left half L_0 to produce the next right half R_1. The right half is taken without any changes as the next left half L_1. This procedure,

including the exchange of the two halves, is called a round. In a feistel cipher these rounds are repeated a number of times. In general, round *r* can be described as follows (⊕ denotes the XOR operation):

$$R_r = L_{r-1} \oplus F(R_{r-1}, SK1)$$
$$L_r = R_{r-1}$$

All rounds, except the last, are the same. In the last round the exchange of the two halves is omitted. This has the advantage that encryption and decryption use the same process. The only difference is, that decryption uses the sub-keys in reverse order to encryption.

Most modern symmetric block ciphers follow the feistel or similar structure. The advantages are:

- Simple operations can provide good security.
- Level of security can easily be scaled by number of rounds.
- Encryption and decryption use the same process.

3.2 Data Encryption Standard (DES)

The Data Encryption Standard (DES) [1] was adopted in 1977 by the National Bureau of Standards (now the National Institute of Standards and Technology NIST) as the standard data encryption algorithm. It has been the "workhorse" of commercial cryptography in the last 30 years. In 1994, NIST reaffirmed DES for another five years as federal standard for applications other than protection of classified information. It has been widely used and analysed. DES is still used as a yardstick to measure the effectiveness of new cryptanalytic attacks.

DES is a feistel cipher with a 56-bit key, a block size of 64 bit and 16 rounds. It has an additional initial permutation before the first round and an inverse of this permutation is applied after the last round. However, these two permutation functions do not improve the security, and, in most analyses these permutations are left out. The round function of DES can be described as follows (cf. Figure 2): Firstly, the 32-bit input of the right half is permutated and expanded to 48 bits. This expanded value is then XORed with the sub-key SK_r. The result of this XOR is fed to eight substitution boxes (S-boxes), each having a 6-bit input and a 4-bit output. The 32-bit output of the S-boxes is then permutated to form the output $F(R_r, SK_r)$ of the F-function in round r.

For the sub-key generation the 56-bit of the key are first permutated and then divided into two halves C_0 and D_0. Each of these halves treated in the following rounds as an entity. In each round, both halves are shifted circularly to the left by either one or two bits, governed by a fixed rotation table. The concatenation of these two values builds the sub-key used in the F-function. The shifted values are passed on to the next round to continue with the generation of the key.

In summary, DES is easy to implement in hardware and software, has insufficient key size and is susceptible to differential and linear cryptanalysis.

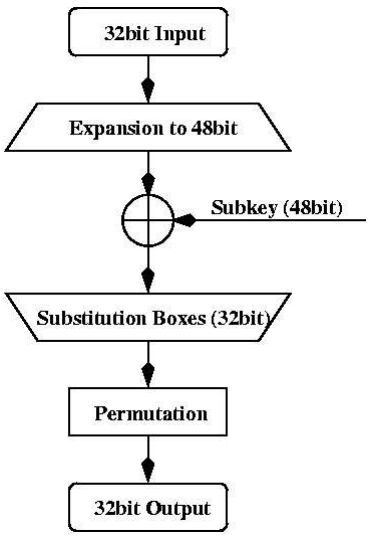

Figure 2: DES F-function

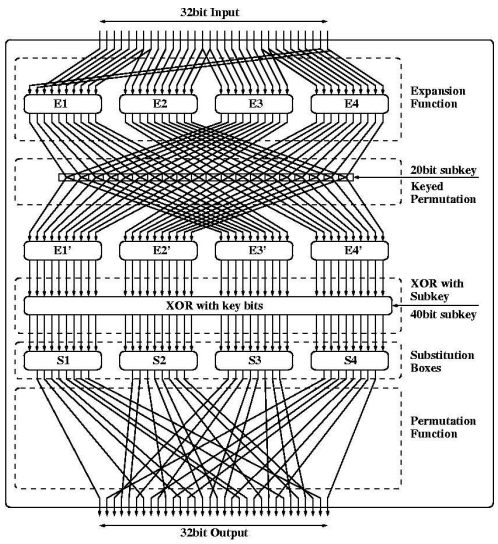

Figure 3: ICE F-function

3.3 Information Concealment Engine (ICE)

The Information Concealment Engine (ICE) [2] was published in 1997 to avoid the weaknesses shown by DES. ICE was designed as a substitute for DES and hence has the same interface as DES. ICE is a feistel cipher with a block size of 64 bits, a 64-bit key and 16 rounds. The main difference to DES is the use of the keyed-permutation. This keyed permutation swaps bits depending on some sub-key bits. This keyed permutation was added to avoid differential and linear cryptanalysis. In addition to standard ICE, there are other versions to allow for a speed/security trade-off.

The F-function of ICE operates as follows (cf. Figure 3): Firstly, the 32-bit right half is expanded to 40 bits. These 40 bits are grouped into four 10-bit values E1 to E4, which are subject to the keyed permutation. In the keyed permutation bits are swapped between E1/E3 and E2/E4. This swapping is governed by a 20-bit sub-key. The result of the keyed permutation is then XORed with a 40-bit sub-key. The output is then fed into the four S-boxes. These S-boxes have a 10-bit input and an 8-bit output, reducing the number of bits to 32. Finally, a permutation function is applied to obtain the 32-bit output $F(R_r, SK_r)$.

The key schedule algorithm of ICE is more complex than that of DES. While the DES key schedule can be performed at encryption time, the ICE key schedule is normally performed prior to encryption/decryption and all the sub-keys are stored in memory. The key schedule algorithm repeatedly shifts and copies bits, or the inverse of bits, into the sub-keys.

In summary, ICE is easy to implement in hardware and software, has fast encryption/decryption rates (3.2 MB/s on a Pentium 200MHz), has a comparably small key. ICE needs a comparably large memory space, due to the complex key schedule. While the algorithm is not broken yet, ThinIce (8 rounds) is susceptible to conditional linear cryptanalysis [3]. ICE with up to 15 rounds can be broken with conditional differential cryptanalysis [4].

3.4 Advanced Encryption Standard (AES/Rijndael)

In January 1997, NIST announced the initiation of an effort to develop the AES and made a formal call for algorithms in September 1997. After a long and extensive review of all candidates, Rijndael [5] was announced as AES in October 2000. Rijndael, while not a feistel cipher, is similar as it is an iterated block cipher. The block size and the key length vary both independently between 128, 192 and 256 bit and the number of rounds is either 10,12 or 14, depending on the block size and key length.

Instead of dividing the plaintext into two halves, Rijndael arranges the plaintext in an array of bytes. This array has 4 rows and 4, 6, or 8 columns, depending on the block size. All functions within one round operate on the whole array. Before the first round, a sub-key is XORed to the plaintext. This operation is added, as any key-independent operation before the first or after the last key-XOR can be peeled of without knowledge of the key. In every round, the bytes in the array are first fed to the S-box, where single bytes are substituted by other bytes. There is only a single S-box for the complete cipher. The bytes in the different rows are then shifted by different offsets. To increase diffusion the bytes in one column are linearly combined afterwards. Finally, the sub-key is XORed with the whole array. In the last round the linear combination of columns is omitted. This makes the encryption and decryption process more similar. Even though encryption and decryption follow the same structure, they have different parameters and hence must be viewed and implemented as two different processes/procedures.

The key-schedule is kept simple and can be performed either before the encryption/decryption process or, in cases where memory is scarce, “on the fly” during encryption/decryption. The original key is arranged in a byte-array of 4 rows and 4,6, or 8 columns. The sub-key generation consists of the key expansion and the round-key selection. The total number of sub-key bits required equals the block-length multiplied by the number of rounds plus one. For the key expansion the first 16/24/32 bytes (128/192/256 bits) are simply the key itself. Each following byte is defined in a recursive way from previous bytes, by simply XORing two different previous bytes together. In regularly intervals, one of these two bytes is modified by cyclic shifts, substitution and XOR offset. The round-key selection allocates the first 16/24/32 byte (depending on the block size of 128/192/256 bit) to the first sub-key, the next 16/24/32 to the second sub-key and so on.

In summary, Rijndael is very fast (8.6MByte/sec for 128bit key, 6.3MByte/sec for 256bit key, both on Pentium 200MHz), easy to implement in software. The implementation in hardware takes extra space, as encryption and decryption are different procedures. The key and block size vary independently between 128, 192 and 256 bits. Key schedule can be performed “on the fly”, allowing for a memory/speed trade-off. Rijndael currently appears to be cryptographic strong.

4. Brute Force Attacks and an Estimation for a Minimum Key Length

No matter how secure a cipher is an attacker can always mount a brute force attack, where simply all possible keys are tried. For this an attacker needs only a single pair of plaintext and ciphertext. Often, if the plaintext has an understandable format, even a single ciphertext may suffice. As long as the wrong key is used, the resulting plaintext will appear as random data and once the resulting plaintext makes any sense it can be assumed with a high probability that the correct key has been found.

When choosing a cipher one should ensure that the key length of the selected cipher is long enough to make a brute force attack infeasible. This means that the number of possible keys is large enough to make it computationally infeasible for an attacker to find the correct key in a certain time-span. However, it is important to remember, that a long key itself does not guarantee that the cipher is secure. Sometimes ciphers have a shortcut, that can be used to find the key much faster than by exhaustive key search. In some cases, the long key is obtained from a short user-memorizeable password, thus effectively reducing the size of the key. Overall, a brute force attack is a worst-case scenario for the attacker, that is only applied if no other attack can be found. Hence, the size of the key is an upper bound on the security of the cipher. In this sense, a good cipher is one that has no shortcut and where the key is long enough to make a brute force attack infeasible.

Table 1 was presented by M. Blaze et.al. [6] in January 1996. It shows the expected execution times of brute force attacks and the costs per revealed keys, for 40 and 56 bit ciphers with different technologies and initial investments. Taking Moore's Law ("The computational power doubles every 18 month") into consideration, the given times must be divided by eight to scale them to today's technology (given in the table in braces).

It can be seen that ciphers with up to 56-bit keys provide poor protection. With an investment of $300,000, the key can be found within about 22 minutes if ASICs are used. Considering that the additional computational cost for stronger encryption is not significant, the authors propose a minimum key length of 90 bits for any new cryptographic system. Considering Moore's Law, this will secure data for approximately 15-20 years, if the cipher used shows no weaknesses and exhaustive key search is the only possible attack. Depending on the sensitivity of data that is already encrypted with lesser key lengths, an upgrade of the used cryptosystem should be undertaken.

Type of Attacker	Budget	Tool	Time/Cost for 40bit	Time/Cost for 56bit	Minimal Length
Pedestrian Hacker	Tiny	Scavenged computer time	1 week (<1 day)	Infeasible	45
	$400	FPGA	5 hours/$0.08 (38.5 min)	38 years /$5000 (5 years)	50
Small Business	$10,000	FPGA	12 min/$0.08 (1.5 min)	18 month/$5000 (2.5 month)	55
Corporate Department	$300K	FPGA or	24 sec/$0.08 (3 sec)	19 days/$5000 (2.4 days)	60
		ASIC	.18 sec/$0.001 (2.5 msec)	3 hours /$38 (23 min)	
Big Company	$10M	FPGA or	.7 sec/$0.08 (87.5 msec)	13 hours /$5000 (1.6 hours)	70
		ASIC	5 msec/$0.001 (.6 msec)	6 min/$38 (45 sec)	
Intelligence Agency	$300M	ASIC	.2 msec/$0.001 (.025 msec)	12 sec/$38 (1.5 sec)	75

Table 1: Overview of brute force attack times and costs

5. Cryptanalysis

The goal of cryptanalysis is to reveal information from the ciphertext without knowledge of the key and/or to reveal the key itself in order to decrypt further messages. In order to achieve these goals, the cipher is analysed to establish any weaknesses in its design. If some weaknesses are found, they may be exploitable to enable an attack that is faster than exhaustive key search. The verification of ciphers by applying cryptanalysis is a destructive process. A cipher is regarded as broken if a technique is found that reveals the key faster (i.e. with less complexity) as the brute force attack. However, it should be kept in mind, that even this faster technique may be still infeasible. For example, a cipher with a key length of 128 bits can be broken with brute force with a

complexity of 2^{127}. If a new technique is found, that can find the key with a complexity of 2^{100}, the cipher is regarded as broken. Nevertheless, the execution of a technique of complexity 2^{100} is still not feasible with today's computing power. Hence, even though the cipher is broken, the attack cannot be mounted practically.

Consider a scenario of a cryptographic attack, shown in Figure 4, where a sender wishes to send a message P in confidence to the receiver over an insecure communication channel. In order to keep the message confidential the sender and the receiver use a cipher and therefore must agree on a key before the communication commences. The plaintext message is encrypted to form ciphertext, which is transmitted over the channel. The receiver decrypts the ciphertext with the same cipher and key, thus revealing the original plaintext. It is assumed that an adversary who does not possess the key is unable to decipher the message.

However, in this scenario it must be assumed that the adversary has powerful abilities and has access to the communication channel. The adversary also has complete knowledge of the cipher and has sizeable computational resources only bound to current technological limits. Further, it is assumed that the adversary has access to a huge number of plaintext/ciphertext pairs, all encrypted with the same key.

Two powerful attacks to break a symmetric cipher are differential and linear cryptanalysis. Both techniques have been successfully applied to a number of symmetric block ciphers.

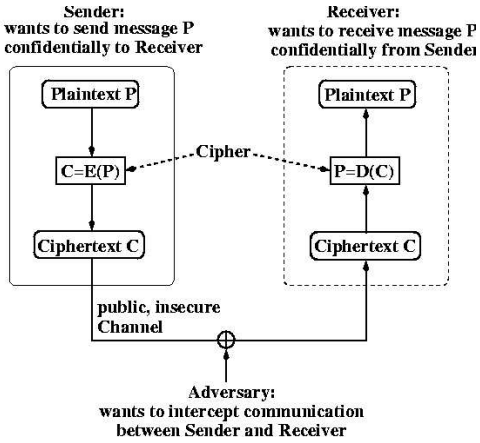

Figure 4: Scenario of a cryptographic attack

5.1 Differential Cryptanalysis

Differential cryptanalysis [7] analyses the evolution of differences (XOR) between two plaintexts as they progress through the encryption process. This analysis is based on statistical properties of the cipher. To mount such an attack, multiple plaintext pairs, all with the same difference and the corresponding ciphertexts are required. As the attacker has to choose the plaintexts, this is a *chosen plaintext attack.* This attack deduces part of the key. The remaining key-bits can be obtained by an exhaustive key search. Ciphers that have shown weaknesses against differential cryptanalysis include DES, ICE, Lucifer and LOKI.

On a single round, differential cryptanalysis works as follows: Assume the left halves of the plaintexts have a difference of a and the right halves have a difference of b. The difference of b in the right half causes with a certain probability p a difference of c in the outcome of the F-function. The right halves in the next round are obtained by XORing the left halves with the outcome of the F-function. Hence, the difference in the right halves in the next round will be $a \oplus c$. As the left halves in the next round are equal to the right halves in the current round, they still have a difference of b. The propagation of the difference through one round is called a 'round characteristic'.

These 'round characteristics' can be combined to obtain multiple 'round characteristics'. The goal is to obtain information about the difference in the left halves before the last round. Then, the difference in the outcome of the F-function can be calculated, as it is equal to the XOR of the difference in the left halves before the last round and the difference in the left halves of the ciphertext (cf. Figure 5). The right halves in the last round are also known, as they are equal to the right halves of the ciphertext. Therefore, the attacker can also calculate the difference in the outcome of the F-function for all possible sub-keys in the last round. These values are then compared to the previously obtained difference. This is done for all acquired plaintext pairs. Counters are maintained recording the number of times the differences are equal. The sub-key, which corresponds to the counter with the highest value, is likely to be the correct sub-key. Information about the key can be deduced by

reversing the key schedule algorithm, using the knowledge gained about the sub-key. The remaining key-bits are then revealed by brute force. The higher the probability of the characteristic and the more plaintext pairs that

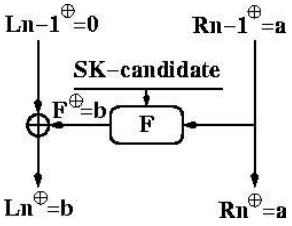

Figure 5: Differential attack

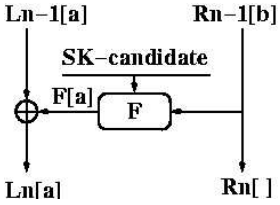

Figure 6: Linear Attack

are available, the higher is the success rate of the attack.
DES is susceptible to differential cryptanalysis. It can be broken with 2^{47} chosen plaintext/ciphertext pairs [7]. There is no publicly known differential attack against the standard version of ICE. However, ICE with up to 15 rounds can be broken by using a derived attack, called conditional differential cryptanalysis [4]. There are no known weaknesses of Rijndael against differential cryptanalysis.

5.2 Linear Cryptanalysis

Linear cryptanalysis [8] uses linear approximations to describe one round of a block cipher. This means that if some plaintext bits and ciphertext bits are XORed together, then the result equals the XOR of some bits of the key with a probability *p*, for the key and any random plaintext/ciphertext pair. These linear approximations are based on statistical properties of the cipher. Linear cryptanalysis is a *known plaintext attack*, as the adversary only has to know, rather than to choose, the plaintext/ciphertext pairs. With this method, the adversary can deduce part of the key. The remaining key bits can be obtained by an exhaustive search. Ciphers that have shown weaknesses against linear cryptanalysis include DES, FEAL, ICE, SAFER.

Linear cryptanalysis works as follows on a single round: Assume that a pattern *a* is applied on the left half of the plaintext, a pattern of *b* on the right half and a pattern *c* on the sub-key. Then the pattern *a*, applied to the left half, is regarded as causing a pattern *d* before the F-function with a certain probability *p*. Hence, in the next round the pattern *a* must be applied on the right half (taken unchanged from the previous left half). The pattern *b* ⊕ *d* must be applied on the left half. A set of patterns with a probability not equal to ½ is called a linear characteristic.

Linear characteristics can be combined to multiple round characteristics. The basic method is the same as for the differential attack. Information about the left half before the last round is obtained with a certain probability. This enables the attacker to get information about the result of the F-function (cf. Figure 6). Then for each plaintext/ciphertext pair the result of the F-function is calculated for all possible sub-keys. Counters are kept that count the occurrence of equality of the calculated result and the result gathered from the characteristics. The sub-key that corresponds to the counter with the highest distance to half the number of used pairs is likely to be the correct sub-key. Reversing the key schedule enables an attacked to deduce part of the key. The remaining key bits can be obtained by brute force. The higher the distance between the probability of the used characteristic and ½ and the more plaintext/ciphertext pairs that are available, the higher is the success rate of the attack.

DES is susceptible to linear cryptanalysis and can be broken with 2^{43} known plaintext/ciphertext pairs [8]. There is no public known linear attack against the standard version of ICE. However, up to 8 rounds (ThinICE) can be broken with a derived attack, called conditional linear cryptanalysis [3]. There are no public known weaknesses of Rijndael against linear cryptanalysis.

6. Conclusion

This paper introduced cryptography and related security services. It reviewed the operation of a general feistel cipher, DES, ICE and Rijndael (AES). Important properties of ciphers for PLC use, such as security, speed and flexibility, have been explained. Table 2 summarises the properties of the symmetric block ciphers DES, FEAL, ICE, 3-WAY, SHARK, IDEA, RC6 and Rijndael (AES).

Exhaustive key search is a first yardstick for the security of ciphers. This brute force attack can always be mounted. The only way to counter an exhaustive key search is to make the key space large enough, so that the attack cannot finish within an acceptable time. The expected execution times and costs of this type of attack

have also been shown. A minimal key length of 90 bits for symmetric ciphers has been proposed, to ensure security for the next 15-20 years. However, some ciphers have weaknesses, which allow an attacker to deduce the key faster than by exhaustive key search. Two techniques to deduce the key by exploiting such weaknesses are differential and linear cryptanalysis. The known weaknesses of the reviewed ciphers against these attacks have also been outlined.

Cipher	Security	Speed (Pentium 200MHz)	Flexibility
DES	Insufficient key size (56bit) Can be broken with differential and linear cryptanalysis	2.2Mbyte/s	Easy to implement in software (SW) and hardware (HW).
FEAL	Small key (64 bit) Can be broken with differential and linear cryptanalysis	4.27Mbyte/s	Easy to implement in SW and HW.
ICE	Small key (64 bit) Weaknesses against differential and linear cryptanalysis	3.2Mbyte/s	Easy to implement in SW and HW. Requires comparably large memory due to complex key schedule.
3-Way	Adequate key (92 bit) Has some weak keys	1.3Mbyte/s	Easy to implement in SW and HW.
SHARK	Large key (128 bit) Weaknesses against interpolation attack	5.3Mbyte/s	Easy to implement in SW and HW.
IDEA	Large key (128 bit) Has some weak keys	6.4Mbyte/s	Easy to implement in SW and HW.
RC6	Large key sizes (128,192 or 256 bit) No known weaknesses	5.2Mbyte/s (any key size)	Easy to implement in SW and HW.
Rijndael (AES)	Large key sizes (128, 192 or 256 bit) No known weaknesses	8.6Mbyte/s (128 bit key) 6.3Mbyte/s (256 bit key)	Easy to implement in software. Hardware implementation needs extra space for decryption process. Speed/memory trade-off possible.

Table 2: Properties of some symmetric block ciphers

7. References

[1] Data Encryption Standard, *Federal Information Processing Standard 46-1*, U.S. Department of Commerce/National Institute of Standards and Technology, 1977

[2] Kwan, M.: The Design of the ICE Encryption Algorithm, *Proceedings of Fast Software Encryption - Fourth International Workshop, Haifa, Israel*. Springer Verlag, 1997, pp.69-82

[3] Coffey, T. and Dojen, R.: Conditional Linear Cryptanalysis and its Application to ICE, *Submitted for Publication to IEE Proceedings – Communications*, 2000

[4] Rompay, B., Knudsen, L.R. and Rijmen, V.: Differential Cryptanalysis of the ICE Encryption Algorithm, *Proceedings of Fast Software Encryption - FSE98, LNCS~1372*, Springer Verlag, 1998, pp.270-283

[5] Daemen, J. and Rijmen, V.: The Rijndael Block Cipher, *AES Proposal*, Document Version 2, 1999

[6] Blaze, M., Diffie, W., Rivest, R.L., Schneier, B., Shimomura, T., Thompson, E. and Wiener, M.: Minimal Key Lengths for Symmetric Ciphers to Provide Adequate Commercial Security, *A Report by an Ad Hoc Group of Cryptographers and Computer Scientists*, 1996

[7] Biham, E. and Shamir, A.: Differential Cryptanalysis of the full 16-round DES, *Advances of Cryptology – CRYPTO'92*, Springer Verlag, 1991, pp.487-496

[8] Matsui, M.: The First Experimental Cryptanalysis of the Data Encryption Standard. *Advances in Cryptology - CRYPTO'94*, Springer Verlag, 1994, pp.1-11