CAB340 - Cryptography
Queensland University of Technology
Semester 2, 2018

Assignment 2

Student: John Santias (N9983244)
Due: Friday 21th September 2018

Contents

LSFRs

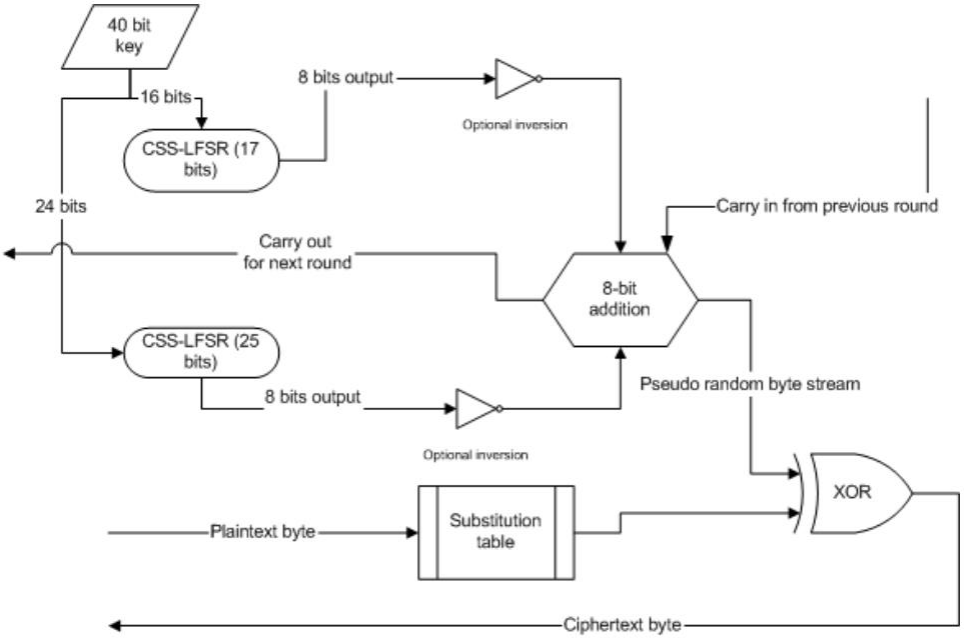

Figure 1: Content scrambling system

a. The maximum possible period for each LFSR can be calculated using the formula: $T = 2^n - 1$ (Linear-feedback shift register - Wikipedia. 2018.)

Where n Is the number of bits (Linear-feedback shift register - Wikipedia. 2018.). The maximum period for a 17-bit LFSR is:

$$T = 2^{17} - 1$$
$$= 131071$$

The maximum period for a 25-bit LFSR is:

$$T = 2^{25} - 1$$
$$= 33554431$$

b. CSS was chosen to have different lengths for the two LFSRs. One outputting 17-bits, the other 25-bits (Linear-feedback shift register - Wikipedia. 2018.). If both LFSRs outputted the same sized bits, then the system is weak. For example, if both LFSRs output 8-bits of output, both outputs from each one is added to form an output byte (Content Scrambling System (CSS): Introduction. 2018.). In this addition, the carry out is used as the carry in for the addition yielding the next output byte (Content Scrambling System (CSS): Introduction. 2018.). Philip Koopman (Linear-feedback shift register - Wikipedia. 2018.9) calculated the period for LFSR sizes from 4 to 32. If both LFSRs had the same size of 8, then the period is 16. Size 17 has a period of 7,710 and size 25 has a period of 1,296,000.

The maximal period would be the same if the sizes are the same. This can be proven using the formula,

$$\frac{pq}{\gcd(p,q)}$$

where p is one of the LFSRs, q is the other, and both have the same size of 8-bits.

$$\frac{8 \cdot 8}{\gcd(8,8)} = 8$$

It would be easier for someone to determine the size of the LFSR.

c. The content scambling system uses a 40-bit key to protect the data. 2 bytes from the 17-bit LFSR, 3 bytes from the 25-bit LFSR. The adversary can use a brute-force attack to find the key, having a complexity of 2^{40} (Limit Exceeded. 2018).

d. In a similar cipher that uses the same method of loading the key and XORs the output of each LFSR to produce the keystream. You can leak the key's information if the mangling step isn't performed during encryption. In figure 1, the substitution table is used before the XOR operation. The table-based substitution is the mangling step to try prevent plaintext attacks (see figure 2). Without this step, the key can easily be retrieved by brute force.

When a plaintext attack is launched, the 5-byte output of the LFSRs can be recovered (Content Scrambling System (CSS): Introduction. 2018.). An attacker and reverse the mangling function by guessing LFSR's 4th-byte key, work backward then verify the input byte, make 2^8 plaintext attack, repeat the process for all 5 bytes (Content Scrambling System (CSS): Introduction. 2018.).

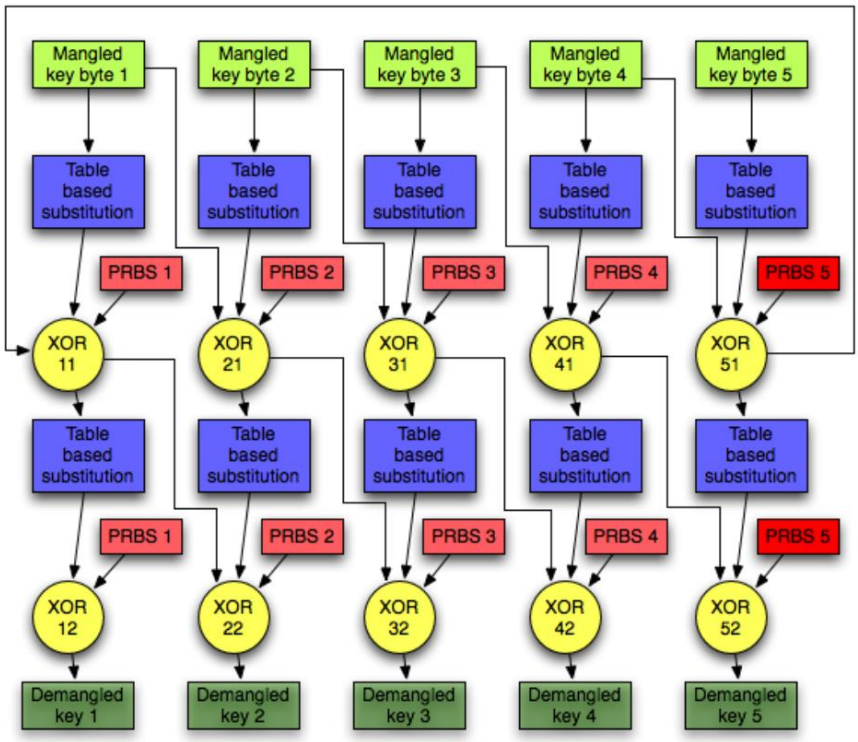

Figure 2: Mangling function of the Content Scrambling System

e. CSS was unable to prevent people from copying the DVD's content onto a disk because it was an epic security failure. The decision to have a 40-bit key was the maximum length the creators could implement by law. The key length was short thus making it feasible to brute-force attacks. The 40-bit CSS encryption uses a combination of two LSFRs, where it takes 2-bytes (16 bits) from one and 3-bytes (24 bits) from another. The system always sets the 4^{th} bit of an LFSR as 1 to prevent a null-cycle. Cryptanalysts can obtain the 40-bit key by guessing the initial state of an LFSR and finding identical bytes or brute-force attack. If the attacker has the initial state, then it can help them find the key. The decision to have a 40-bit key was the maximum length the creators could implement. If the key lengths were longer, it would have made it harder for attackers to find the key.

Linearity in block ciphers

a. Imagine a Hill cipher as a block cipher in CTR mode. It is a synchronous stream cipher that generates a keystream by encrypting the value of a counter and initialising it with a nonce N. The formula for generating the keystream is:

$$O_t = E(T_t, K)$$

where $T_t = N||t$ concatenates the nonce and block number t. We assume that d is even and the nonce and counter have length $\frac{d}{2}$, K is the hill cipher key.
The formula for the first three CTR block ciphers are:

$$Encryption: C_t = O_t \oplus P_t$$
$$Decryption: P_t = O_t \oplus C_t$$

The nonce value used in this stream cipher is also known as the initialisation vector.

b. An attacker can launch a plaintext attack against a Hill cipher used in CTR mode working its way back of the encryption process. The attacker can use the XOR operation of the plaintext and ciphertext to get the keystream, then can try brute-force to find the correct key. The adversary will have to determine the correct matrix key. The nonce value would be known by anyone. The nonce is the same as the initialization vector but concatenated with the counter (Content Scrambling System (CSS): Introduction. 2018. 0). The IV usually does not need to be secret (Content Scrambling System (CSS): Introduction. 2018. 0). Once the adversary has obtained the correct encryption inputs, they can verify the values by comparing the outputted results.

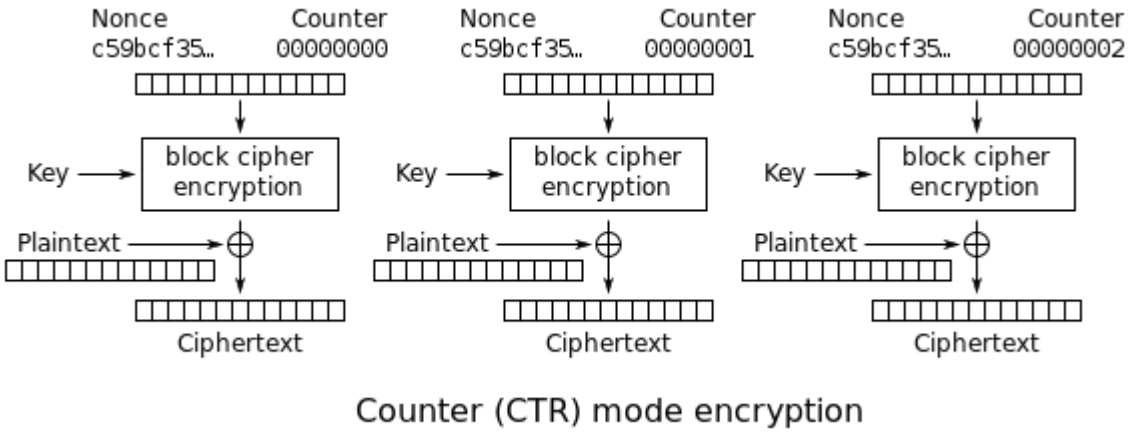

Figure 3: CTR block cipher encryption

Using CTR mode does reduce the known plaintext attack against Hill cipher in ECB mode because it uses a nonce and a counter to generate the keystream, it has three inputs.

In the ECB block cipher operation, it uses inputs a key and a plaintext/ciphertext to encrypt/decrypt. The security for operation only uses the key which an attacker can easily break by brute-force. In CTR mode of operation, it uses a key and a nonce value for encryption then uses the outputted keystream in the XOR operation with

the ciphertext/plaintext. CTR mode is feasible to break if the attacker knows the plaintext.

c. When a hill cipher is used in CBC mode, each block will chain together. The first block uses a random initialisation vector and is XOR'd with the plaintext. The formula for the first CBC block cipher is:

$$Encryption: \ C_t = E(P_t \oplus C_0, K)$$
$$Decryption: \ P_t = D(C_t, K) \oplus C_0)$$

Where P_t is the plaintext, C_0 is the initialisation vector, and K is the hill cipher encryption or decryption key.

The formulas for the second and third block is:

$$Encryption: \ C_t = E(P_t \oplus C_{t-1}, K)$$
$$Decryption: \ P_t = D(C_t, K) \oplus C_{t-1})$$

Where C_{t-1} is the ciphertext block generated in the previous iteration.

d. CBC mode is feasible against plaintext attack because the attacker can use the plaintext and XOR it with the IV. As mentioned earlier in question 2b. The IV isn't usually secret because it can be predictable (Content Scrambling System (CSS): Introduction. 2018. 0). The attacker can use the output of the XOR operation and guess the key matrix and its size by brute-force. They can also guess the key repeatedly until the output matches the original ciphertext then be able to use it to break the other block ciphers.

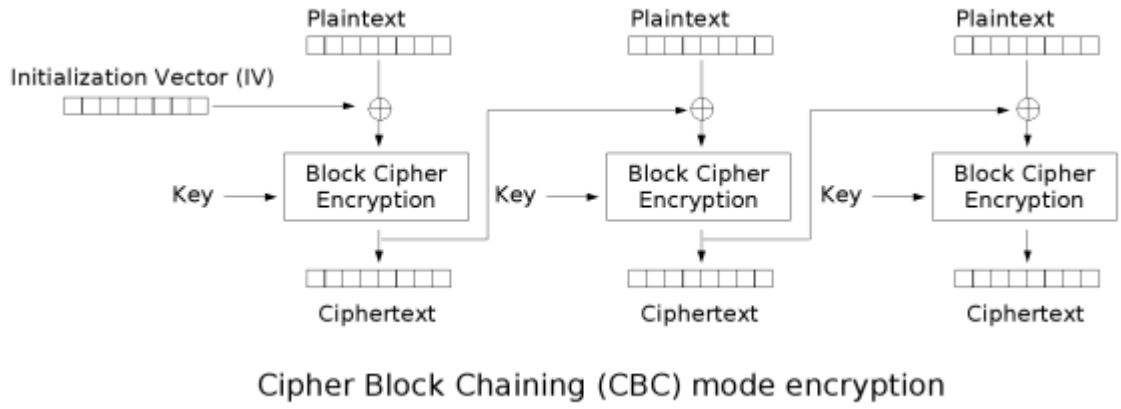

Figure 4: CBC Block cipher encryption

Unlike ECB mode, CBC reduces the known plaintext attack because it uses a binary XOR operation during its encryption/decryption process. The adversary will also have to predict the IV which is usually known to the public and will need to make more computations to break the whole block cipher. The output of the XOR is dependant on the key vice versa.

Using block ciphers in hash functions

a. The simplest way to use AES-128 as a compression function is in CBC mode. The Merkle-Damgard construction splits up the plaintext into small blocks. The first block is XOR'd with the initialisation vector (IV) and sent with a key into the compression function which is the AES-128 block cipher. The compression function will take these inputs to produce a 128-bit output to use as an IV in the next compression function (One-way compression function - Wikipedia.). The message

b. The compression function for AES-128 discussed in the previous question is preimage-resistant (one-way function) because when using an input of 256-bits into AES-128, the output is half the size of the inputs. When you go through the construction backward, you can't obtain the original inputs. However, since we have AES-CBC in this construction, it can be possible to break the construction using a plaintext attack.

   The chained block cipher can be parallelized in decryption meaning that if the two ciphertext blocks are adjacent, the plaintext block can be recovered (Linear-feedback shift register - Wikipedia. 2018.8). If the IV is constant, then confidentiality isn't achieved. However, with this AES-CBC case, it's impossible to decrypt. If an adversary knows the plaintext, they can try to run through the encryption process of this construction to find the matching output. The output can vary if the IV is different.

c. A hash function is a collision resistant if the two inputs hash to the same output but not equally the same (Collision resistance - Wikipedia). For example, $H(a) = H(b)$ but $a \neq b$.

   Collision resistant doesn't mean that there aren't any collisions, it means that they are hard to find. Most modern functions have 256-bit output. Compared to our 128-bit output AES-CBC, it will be harder for an attacker to find any two distinct inputs with the same hash output. A birthday attack is a class of Brute-Force attacks, where it helps the attacker determine the probability of two outputs having the same value. It can be calculated with the following formula (2018. Birthday attack):

$$2^{k/2} \text{ or } 2^{k}$$

   Where k is the number of bits.

Bits Possible outputs (H)		Desired probability of random collision				
		0.10%	**1%**	**25%**	**50%**	**75%**
16	2^{16} (~6.5 x 10^{4})	11	36	190	300	430
32	2^{32} (~4.3 × 10^{9})	2900	9300	50,000	77,000	110,000
64	2^{64} (~1.8 × 10^{19})	1.9×10^{8}	6.1×10^{8}	3.3×10^{9}	5.1×10^{9}	7.2×10^{9}
128	2^{128} (~3.4 × 10^{38})	8.3×10^{17}	2.6×10^{18}	1.4×10^{19}	2.2×10^{19}	3.1×10^{19}
256	2^{256} (~1.2 × 10^{77})	1.5×10^{37}	4.8×10^{37}	2.6×10^{38}	4.0×10^{38}	5.7×10^{38}
384	2^{384} (~3.9 × 10^{115})	2.8×10^{56}	8.9×10^{56}	4.8×10^{57}	7.4×10^{57}	1.0×10^{58}
512	2^{512} (~1.3 × 10^{154})	5.2×10^{75}	1.6×10^{76}	8.8×10^{76}	1.4×10^{77}	1.9×10^{77}

Table 1: Probability of a random collision using the outputs

With a 256-bit output, the attacker will have to make 4.0×10^{38} attempts to get a collision using brute force. With a 128-bit output, the attacker will make 2.2×10^{19} attempts for a collision. Therefore, having a 256-bit is more infeasible than 128-bits.

Message authentication codes

a. i. In principle, it is possible to create a hash function by using CBC-MAC with a constant, public key. CBC-MAC is used for integrity but not confidentiality. Attackers can find the single-block message m that hashes to the given digest d if the key is reused for encryption and authentication (Linear-feedback shift register - Wikipedia. 2018.0). Usually the IV is set to zero and in this case, we have a key that is public.

The attacker can find collisions. the attacker can break the CBC-MAC operation by finding collisions. This function is not pre-image resistant as having a key to create a different message still hashes to the same tag (Linear-feedback shift register - Wikipedia. 2018.0) which it can be used to create a new message. This function is vulnerable to a MITM attack which the attacker can modify the ciphertext in transit but still produce the same output.

ii. When an attacker receives the length, they can find the message. The output of the CBC-MAC is used as the MAC allowing attackers to forge it. This is done by getting the MAC T on message 1, then XOR'ing the tag T to the second message, then getting the MAC from the modified message 2. The resulting tag is still a valid MAC. They can use the length of the message in the tag formula, $T = MAC(M, K)$ where the key will have to be retrieved. The attacker can work its way backward of the encryption process to obtain the plaintext message.

b. i. Padding is a solution for small data to match the block size. It is added to the end of the data. Suppose that an adversary obtains a message that is not an integer number of blocks in length and the CBC-MAC tag for that message assuming the padding scheme. It is possible to easily construct another message that gives the same tag for the same key.

When a message is padded, it basically adds 0's to the last block until it becomes a one full block size. The problem with add just 0's, the tag will be the same.

$$pad(m) = pad(m||0)$$

This zero-padding does not make any difference but allows a forgery attack (Linear-feedback shift register - Wikipedia. 2018.1). It is possible to easily construct another message when the attacker obtains the tag.

```
... | 1011 1001 1101 0100 0010 0111 0000 0000 |
```

Figure 5: Padding messages

ii. When an adversary is given a message which is an integer number of blocks in length. They can use the length to calculate the hash like question 4bi. In a similar way, the attack can use the length of the message, and can calculate the hash of the message blocks. This allows the attacker to include more information at the end of the message. Like forgery attack but doesn't modify the original information, they can append information. Length-extension attack.

iii. A more secure method is if the padding is invertible such that,

$$m_0 \neq m_0 \Rightarrow pad(m_0) \neq pad(m_1)$$

disallowing collisions on the padding functions.
This is achieved if padding starts with a '1' and followed by n-1 zeros until it becomes a multiple of the block length (Linear-feedback shift register - Wikipedia. 2018.2). If the length of the original message has the length of the block size, an extra dummy block is added. This whole system then becomes a one to one function where the messages are distinct and it would be hard for the adversary to start the forgery attack with the correct tags (Linear-feedback shift register - Wikipedia. 2018.3).

c. i. CBC-MAC is secure is the following is true:

- The IV used is always the same.
- Messages are always same length.

If it doesn't hold any one of those conditions, then CBC-MAC is insecure. In a given block of two n-block messages,

$$P = P_1|0P_2|..|0P_n \text{ and } P' = P'_1|0P'_2|..|0P'_n$$

And tags T and T' such that

$$T = CBC - MAC(P, K) \text{ and } T' = CBC - MAC(P', K)$$

You can produce a new block message without knowing K. When an attacker obtains two messages P and P' and its corresponding MAC tags (T, T'). The adversary can get MAC tag T' by combining the messages together, then concatenate P and P' to form a combined message. However, the first block P' is manipulated so that when the attacker inputs it to the encryption algorithm, it will be the same as when the MAC is computed for P' (Linear-feedback shift register - Wikipedia. 2018.4).

XCBC defeats this attack because it uses three keys. The keys are XORed before the last block is encrypted. The following below is a representation of XCBC.

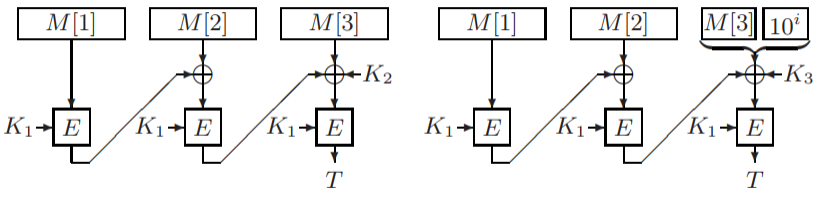

Figure 6: Illustration of XCBC

In the same scenario of knowing the two block messages and its corresponding tags. The MACs can still be computed if the padding is not applied to the blocks. MACs can be computed using the key K_2 (Linear-feedback shift register - Wikipedia. 2018.6). If there's no padding, the length of the messages has the length $(n + 1)q$ bits.

The number of messages or MAC pairs is $2^{\frac{n}{2}+1}$ (Linear-feedback shift register - Wikipedia. 2018.6). Chris J. Mitchell (Linear-feedback shift register - Wikipedia. 2018.6) argues that the probability of a MAC from the first block message is equal to the MAC from the second block message is around 0.63. The formula is defined (Linear-feedback shift register - Wikipedia. 2018.6):

$$e_{K_1}(Q \oplus X^* \oplus K_3) = e_{K_1}(Q \oplus Y^* \oplus K_2)$$

If we assume the key used in the encryption is fixed, the permutation set of all n-bit blocks uses the formula (Linear-feedback shift register - Wikipedia. 2018.6):

$$Q \oplus X^* \oplus K_3 = Q \oplus Y^* \oplus K_2$$

This allows the attacker to learn the value of the second and third key K_2 and K_3 enabling forgery attack. If padding was used (satisfying $(q-1)n < \ell < qn$ which MAC is known), then the unpadded message also has MAC M (Linear-feedback shift register - Wikipedia. 2018.6).

The construction uses three keys which makes it hard for the adversary to distinguish each block cipher. It is more secure because the attacker will need to find the message' MAC equalling the MAC from the second set with a probability of 63% chance. If the padding is not applied and the keys are fixed, potentially the attacker can break the construction.

ii. The chance of an attacker generating a valid message and MAC pair is very likely if the second and third keys are equally the same. Regardless whether there is padding, the attacker can find the MAC of the message with a valid tag T. As described in the previous question, when the attacker knows the second and third key, they can start forgery attacks with the same MACs.

References

Linear-feedback shift register - Wikipedia. 2018. Linear-feedback shift register - Wikipedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/Linear-feedback_shift_register. [Accessed 21 September 2018].

Content Scrambling System (CSS): Introduction. 2018. Content Scrambling System (CSS): Introduction. [ONLINE] Available at: https://www.cs.cmu.edu/~dst/DeCSS/Kesden/. [Accessed 21 September 2018].

Zhou, Yan, Liu, Liu, Black, BZ, PY, GL, ZL, MB, 2004. Content Scramble system (CSS). Content Scramble system (CSS), 1, 17.

Correlation attack - Wikipedia. 2018. Correlation attack - Wikipedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/Correlation_attack. [Accessed 21 September 2018].

Pedersen, Munk, Andersen, LP, CM, LA, 2007. Cryptography – The Rise and Fall of DVD Encryption. Cryptography – The Rise and Fall of DVD Encryption, 1, 25.

Download Limit Exceeded. 2018. Download Limit Exceeded. [ONLINE] Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.118.6103&rep=rep1&type=pdf. [Accessed 21 September 2018].

One-way compression function - Wikipedia. 2018. One-way compression function - Wikipedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/One-way_compression_function. [Accessed 21 September 2018].

Collision resistance - Wikipedia. 2018. Collision resistance - Wikipedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/Collision_resistance. [Accessed 21 September 2018].

Birthday attack - Wikipedia. 2018. Birthday attack - Wikipedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/Birthday_attack. [Accessed 21 September 2018].

CBC-MAC - Wikipedia. 2018. CBC-MAC - Wikipedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/CBC-MAC. [Accessed 21 September 2018].

Coursera. 2018. MAC Padding - Message Integrity | Coursera. [ONLINE] Available at: https://www.coursera.org/lecture/crypto/mac-padding-vTbf0. [Accessed 21 September 2018].

Padding (cryptography) - Wikipedia. 2018. Padding (cryptography) - Wikipedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/Padding_(cryptography). [Accessed 21 September 2018].

David Ireland, DI Management Services Pty Limited, Australia, www.di-mgt.com.au. 2018. Padding schemes for block ciphers. [ONLINE] Available at: https://cryptosys.net/pki/manpki/pki_paddingschemes.html. [Accessed 21 September 2018].

SpringerLink. 2018. Stronger Security Bounds for OMAC, TMAC, and XCBC | SpringerLink. [ONLINE] Available at: https://link.springer.com/content/pdf/10.1007%2F978-3-540-24582-7_30.pdf. [Accessed 21 September 2018].

Iwata, Kurosawa, TI, KK, 2003. Stronger Security Bounds for OMAC, TMAC and XCBC. Stronger Security Bounds for OMAC, TMAC and XCBC, 1, 30.

Mitchell, CM, 2003. On the security of XCBC, TMAC and Royal Holloway University of London OMAC. On the security of XCBC, TMAC and Royal Holloway University of London OMAC, 1, 16.

SpringerLink. 2018. CBC MACs for Arbitrary-Length Messages: The Three-Key Constructions | SpringerLink. [ONLINE] Available at: https://link.springer.com/chapter/10.1007/3-540-44598-6_12. [Accessed 21 September 2018].

Block cipher mode of operation - Wikipedia. 2018. Block cipher mode of operation - Wikipedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation. [Accessed 21 September 2018].

Maximal Length LFSR Feedback Terms. 2018. Maximal Length LFSR Feedback Terms. [ONLINE] Available at: https://users.ece.cmu.edu/~koopman/lfsr/index.html. [Accessed 21 September 2018].

Medium. 2018. Why are nonces important on CTR mode ciphers – Code Fighters – Medium. [ONLINE] Available at: https://medium.com/code-fighters/why-are-nonces-important-on-ctr-mode-ciphers-6aba7503bd3c. [Accessed 21 September 2018].