Attacking Weak Crypto

All your keys belong to us!!!

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Overview

While most cryptography has been proven sound at a mathematical level, many weaknesses are often introduced in practices due to poor or incorrect implementations.

Understanding some of these weaknesses will greatly aid in your ability to spot problems and potential flaws in security systems.

The rest of these slides look at some of these possible weaknesses.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Key size

Key sizes are perhaps the most obvious source of weakness to look for.

Weak key sizes allow us to launch brute force attacks against the security of a system.

So what is a weak key size?..
It depends on who's asking, and which algorithm.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Meet in the Middle Attack

Key size weaknesses, can also be a attacked a few other ways, effectively weakening the key size.

Why no Double DES?

In systems that use multiple keys, i.e. encrypt with key 1 then encrypt with key 2, we can sometimes use a **Meet in the middle attack.**

The idea is simple enough. Rather than trying to brute force the total key of Key1+Key2 we brute force both separately, starting at different ends.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Weak public key factorisation

AKA common factor attacks.

In February 2012, two groups of researchers (Lenstra et al. and Heninger et al.'s) revealed that large numbers of RSA encryption keys that are actively used on the Internet can be cracked because the random numbers used to generate these keys were not random enough.

So lets have a look at how this works…

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Weak public key factorisation

For RSA we choose two random primes p and q.

We use these to create our public key (n,e)

To crack this someone would need to factor n to find p and q which is an extremely hard problem.

If we look at two separate public keys, (n1,e1) and (n2,e2) we know n1=p1*q1 and n2=p2*q2

But what if p1 or q1 = p2 or q2, i.e. what if we had reused one of the prime numbers???

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Weak public key factorisation

It works out that there is a very ancient and very fast method for calculating gcd, the greatest common divisor.

So we can quickly check n1 and n2 to see if they share a prime factor. And hence quickly solve.

Cryptosense have a good online key tester that will allow you check your own keys
https://cryptosense.com/an-online-rsa-key-tester/

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Wiener's Attack

Is an attack against RSA to expose the private key d when d is small. $D < {}^{1}/_{3}\, N \text{ ^ } {}^{1}/_{4}$

In the RSA cryptosystem, Bob might tend to use a small value of d, rather than a large random number to improve the RSA decryption performance.

However, Wiener's attack shows that choosing a small value for d will result in an insecure system in which an attacker can recover all secret information, i.e., break the RSA system.

Boneh and Durfee is a similar attack for small d.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Coppersmith's attack (Small public exponent attack)

Is a method for attacking RSA when the public exponent e is small or when partial knowledge of the secret key is available.

In order to reduce encryption or signature-verification time, it is useful to use a small public exponent e.

In practice, common choices for e are 3, 17 and 65537. They are chosen because they make the modular exponentiation operation faster.

Hastad's attack is another similar attack.

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Coppersmith's attack (Small public exponent attack)

If the public exponent is small and the plaintext m is very short, then the RSA function may be easy to invert which makes certain attacks possible.

$C = M^e \bmod n$

But what if $M^e < n$??

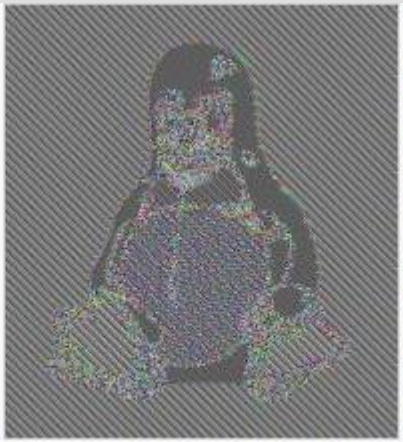
Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Chaining modes

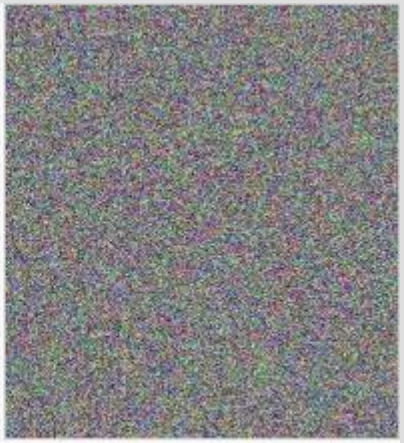
All block ciphers use chaining modes, the incorrect use of certain modes can lead to weaknesses in the encrypted data.

One such example is the use of ECB with highly structured data.

Original image



Encrypted using ECB mode



Modes other than ECB result in pseudo-randomness

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Padding Methods

Block ciphers often use padding schemes to fill out the blocks size for the last block.

Most block ciphers use PKCS7 for padding.

PKCS7 says that the value to pad with is the number of bytes of padding that are required. So, if the blocksize is 8 bytes and we have the string "ABC", it would be padded "ABC\x05\x05\x05\x05\x05". If we had "ABCDEFG", with padding it would become "ABCDEFG\x01".

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Padding Methods

In most encryption schemes the system will produce an error if the padding is incorrect. We can use these error messages to out advantage.

One such attack is the **Padding Oracle Attack**.

How it works:

When the block will be deciphered there will be a verification to check if the padding is good or not,

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Padding Oracle Attack

S|E|C|R|E|T| |M|E|S|S|A|G|E|03|03 => Wrong padding
S|E|C|R|E|T| |M|E|S|S|A|G|E|02|02 => Good padding

Now imagine we can know when we have a bad padding and a good padding (the server send an "error padding" or "404 not found" when the padding is wrong etc).

We will call this our **Oracle**.

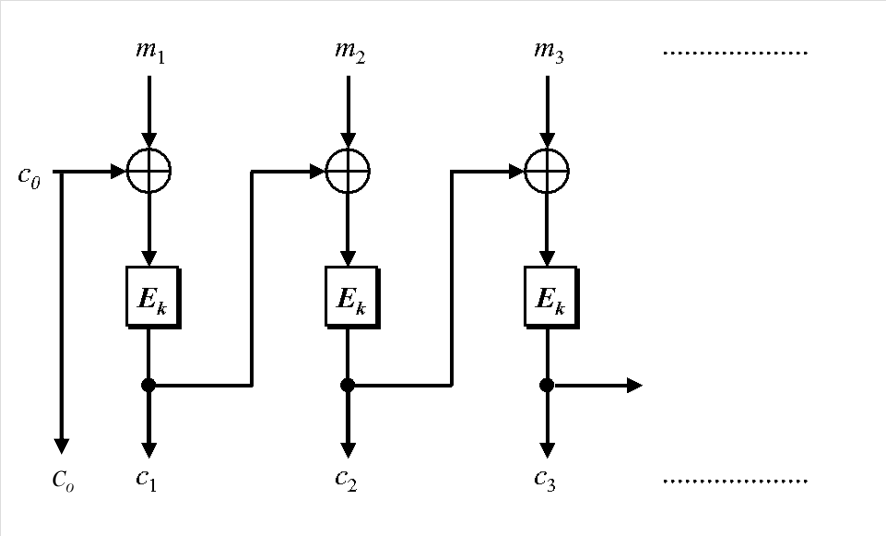
The answers he will give us will be :

- good padding
- bad padding

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Padding Oracle Attack

So we need to understand how CBC works.



So the blocks sent to the oracle will be C_0, C_1, etc

When decrypting $M_1 = C_0 \oplus D_k(C_1)$ or $M_n = C_{n-1} \oplus D_k(C_n)$

Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Padding Oracle Attack

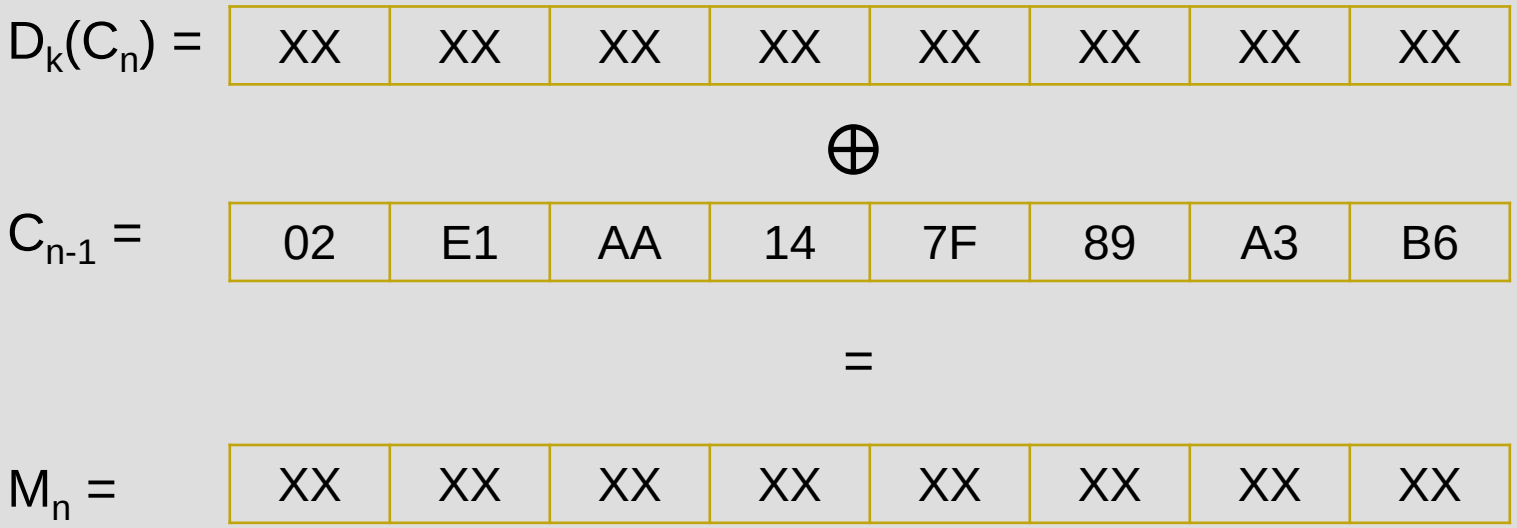
So when decrypting the oracle will check if the padding format for M_n is correct or else return an error.

We know the format of the last block message + padding and we can use this to our advantage.

So what we attempt to do is change the second last block, 1 byte at a time, so it doesn't cause an error, and we can use this method to rebuild the original message 1 byte at a time.

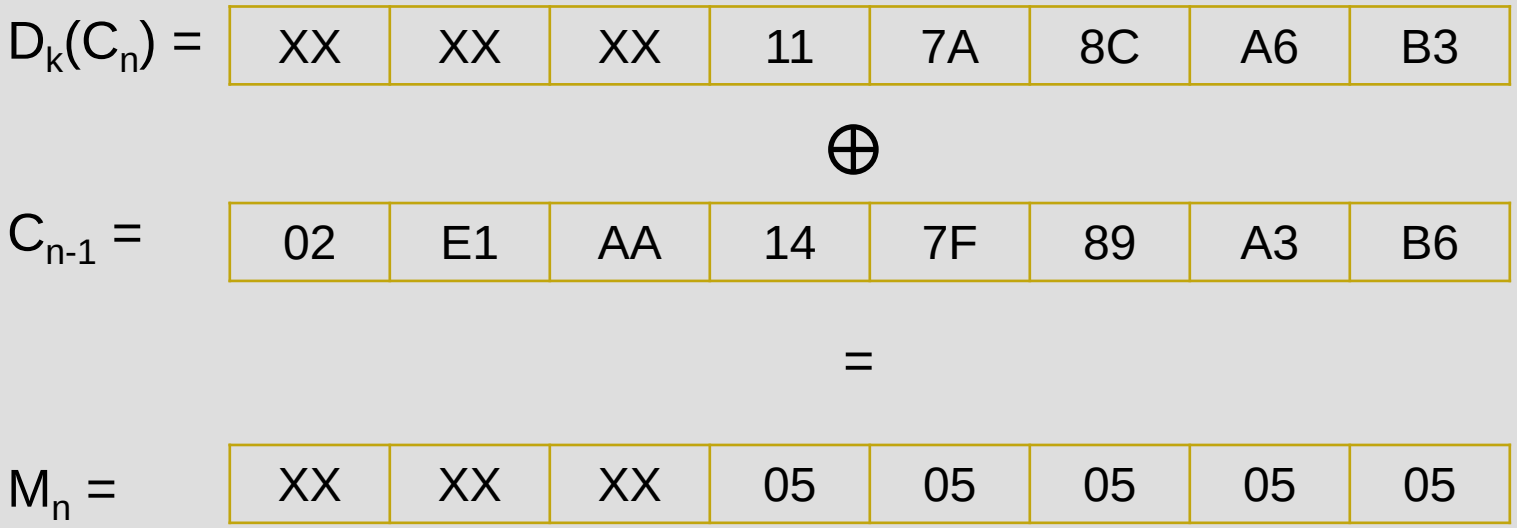
Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Padding Oracle Attack



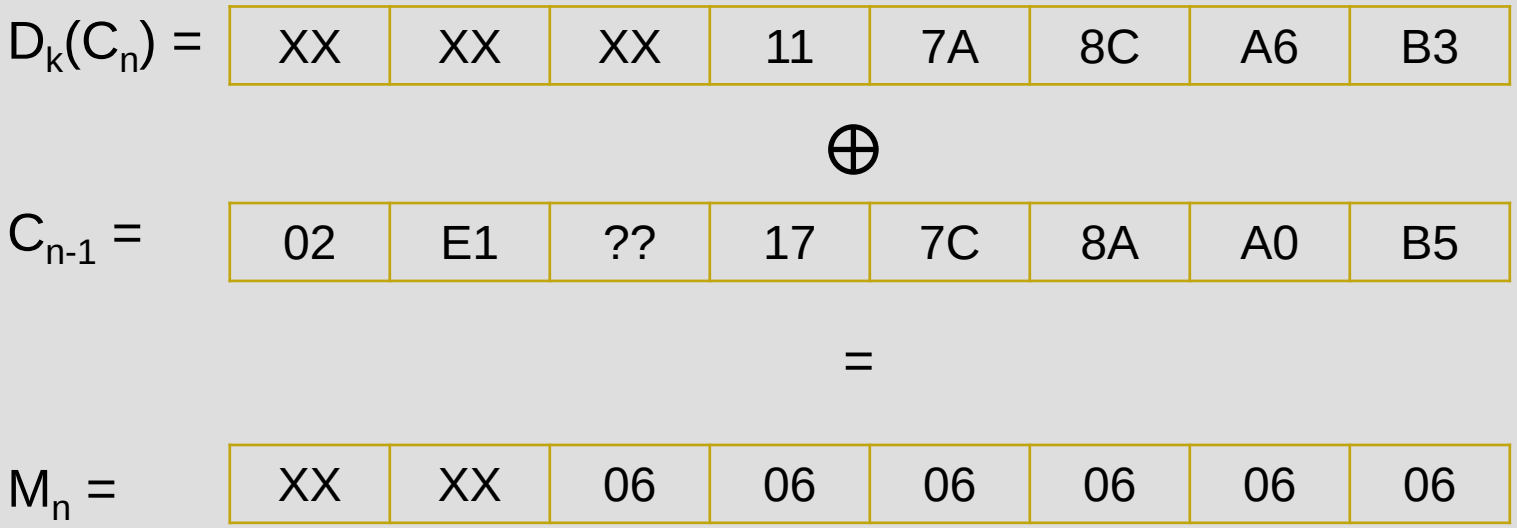
Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Padding Oracle Attack



Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156

Padding Oracle Attack



Mark Cummins
Institute of Technology Blanchardstown
mark.cummins@itb.ie
+353 1 885 1156