12. Module - Attacking Crypto

Verschlüsselung:

die Umwandlung von Klartext in Chiffretext

Symmetrische Verschlüsselung:

Hier wird ein einziger Schlüssel für die Verschlüsselung und Entschlüsselung verwendet.

Asymmetrische Verschlüsselung:

Hier wird 2 Schlüsseln verwendet, also **public Key** für die Entschlüsselung und die **private key** für die Verschlüsselung.

Ciphers / Chiffren

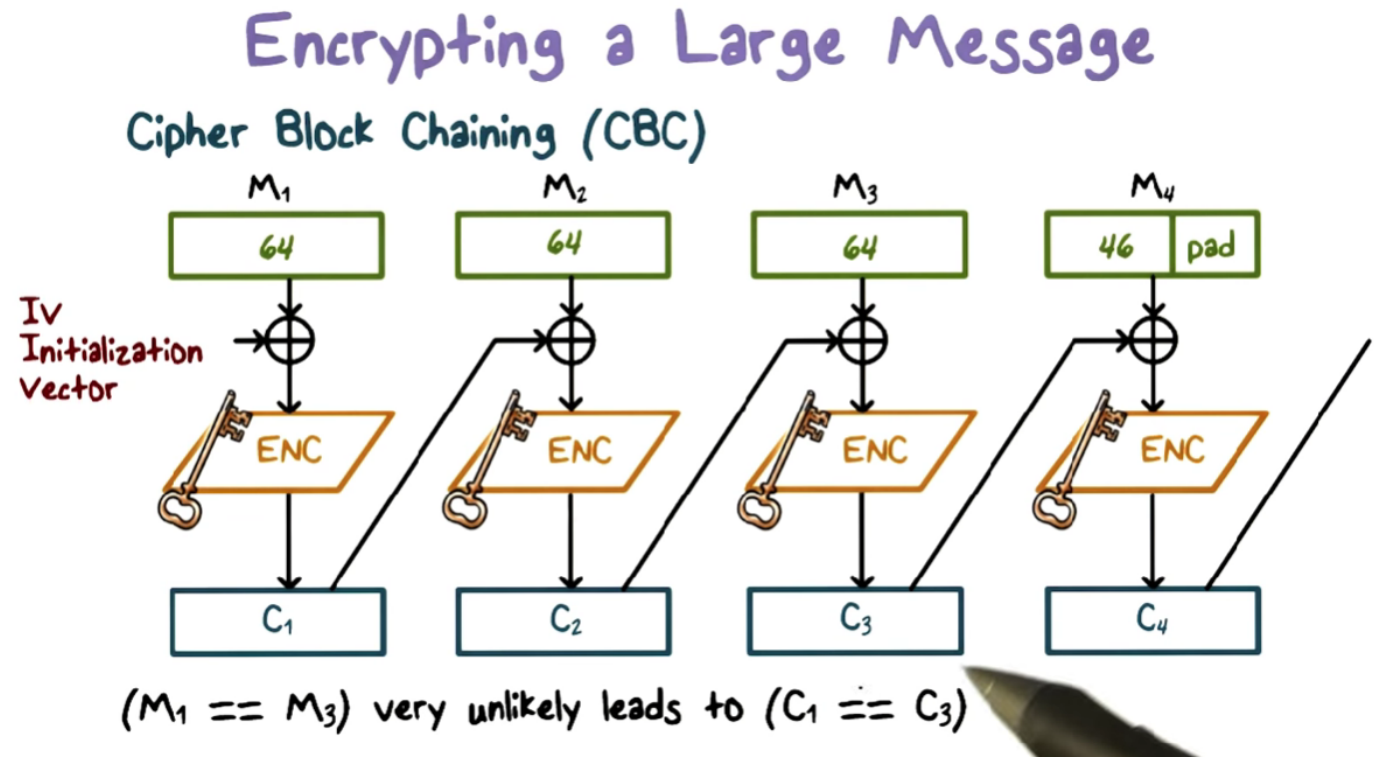
Cipher ist ein Algorithmus zum Durchführen von Verschlüsselung oder Entschlüsselung von Daten.
Hier gibt es 2 Arten:
• **Stream Ciphers**: Daten werden einzeln verschlüsselt. Also die Daten werden in blöcken geteilt, dann jede blöcke wird mit einem Schlüssel verschlüsselt.
• **Block Ciphers**: Daten werden in Blöcken verschlüsselt.

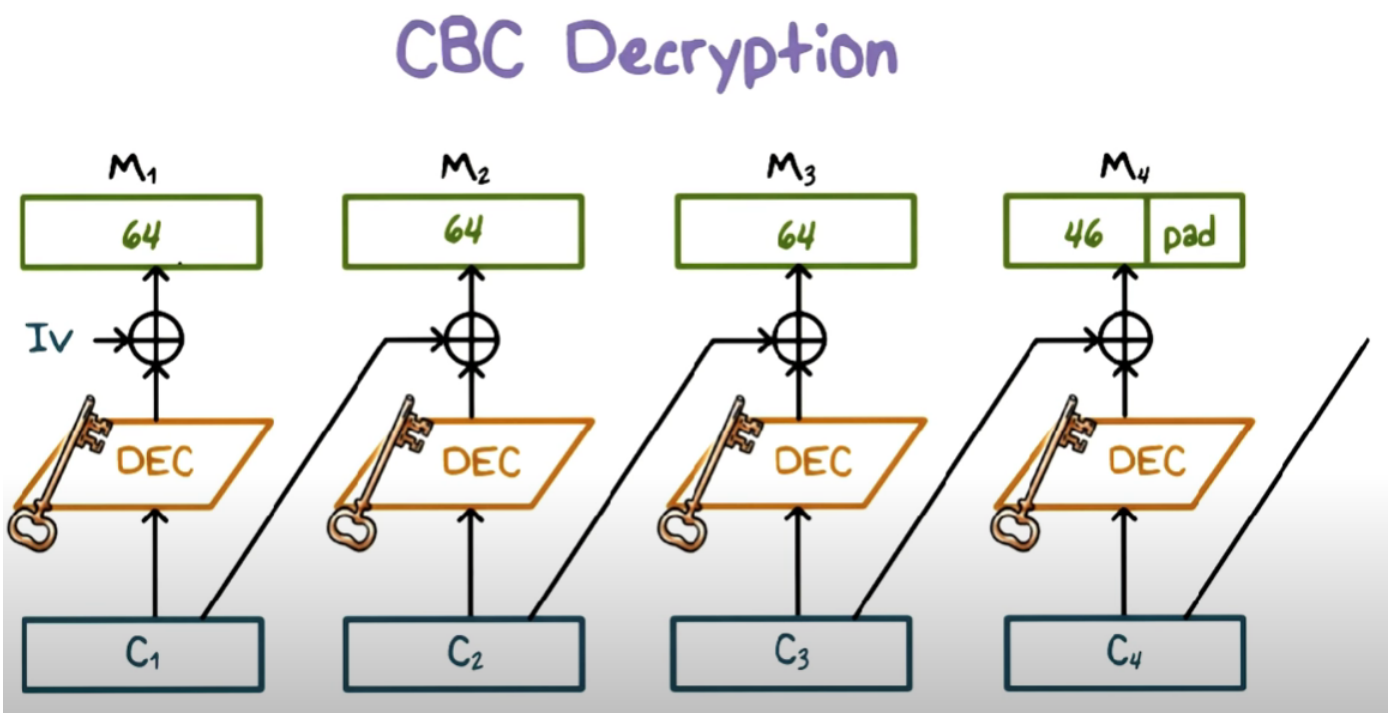
Electronic Code Book (ECB):?

Cipher Block Chaining (CBC):

is a block cipher mode of encryption. This means that it encrypts plaintext by passing individual block of bytes (each character is a byte) of a fixed length through a “block cipher”, which uses a secret key to pretty much mess up the block beyond recognition. So if you were encrypting the sentence:
`This is a sentence of a carefully chosen length.`
you would encrypt the first block of 16 characters using your chosen block cipher algorithm, then the next block, then the final block. If the final block does not have exactly 16 characters then you add padding until it does.

Same key encrypt (like AES) for all blocks





Note: The IV should be known to the sender & reciever

Padding

in **Block Ciphers** findet die Verschlüsselung in Blöcken mit fester Größe statt, und das Padding wird verwendet, um sicherzustellen, dass die Klartextdaten (von beliebiger Größe) genau in den Blöcken mit

fester Größe passen. Die bevorzugte Methode zum Auffüllen von Blockchiffretexten ist **PKCS7**

	Block 1								Block 2							
Byte	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8
A	A	0x07	0x07	0x07	0x07	0x07	0x07	0x07								
Sunil	S	u	n	i	l	0x03	0x03	0x03								
Sudhanshu	S	u	d	h	a	n	s	h	u	0x07	0x07	0x07	0x07	0x07	0x07	0x07

Angriffe auf Krypto

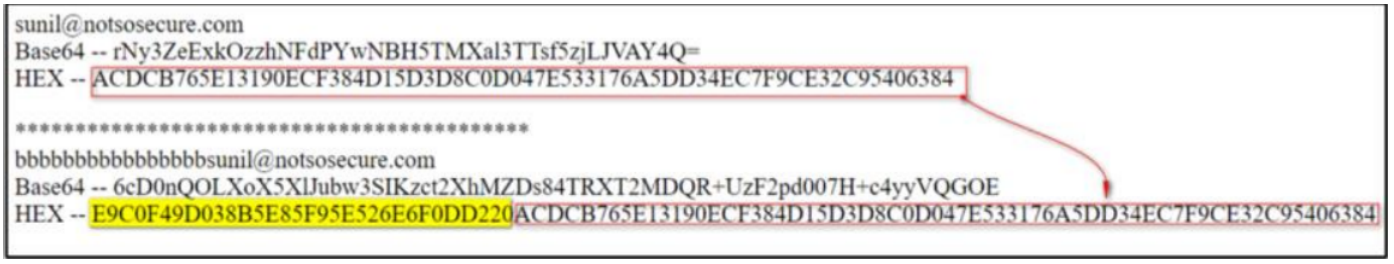
• Ciphertext-Only Attack (COA)
-> Der Angreifer hat Zugriff auf einen Satz/Teil von Chiffretext(e)
• Chosen-Ciphertext Attack (CCA)
-> The attacker can choose different ciphertexts to be decrypted and obtain the corresponding plain text
• K

Padding Oracle Attack

-> Der Angreifer kennt den Klartext und seine verschlüsselte Version (Chiffretext)
• Chosen Plaintext Attack (CPA)
-> Der Angreifer kann Klartexte seiner Wahl verschlüsseln

Insecure Password Reset

Mithilfe einer unsicheren Implementierung der Verschlüsselung können Angreifer Konten übernehmen.
Beispiel:
Eine normale E-Mail **sunil@notsosecure.com**. Angreifer hier kann E-Mail-Adressen wie **bbbbbbbbbbbbbbsunil@notsosecure.com** und **xxxxxxxxxxxxxxxxsunil@notsosecure.com** registrieren und dann eine Kennwortzurücksetzung anfordern.
Der Angreifer nimmt **den gemeinsamen Teil** der empfangenen Token, der ein gültiger Token zum Zurücksetzen des Passworts für sunil@notsosecure.com ist, und setzt das Passwort des Targetkontos erfolgreich zurück :)



aaaaaaaaaaaaibrahim@gmail.com -> token -> asdasdasdmnbvmnbv
sssssssssssibrahim@gmail.com -> token -> qweqweqwemnbvmnbv
we realize that this part is the same in both token mnbvmnbv
so the token for the ibrahim@gmail.com is mnbvmnbv
-> we performed a Known Plaintext Attack (KPA), because we know the plain text which is the email and the ciphertext which is the token!

Padding Oracle Attack

Was ist Oracle?

An oracle ist jede Anwendungsfunktionalität, Fehlermeldung oder jedes Verhalten, das wertvolle Informationen preisgeben kann

Was ist padding oracle Attack ?

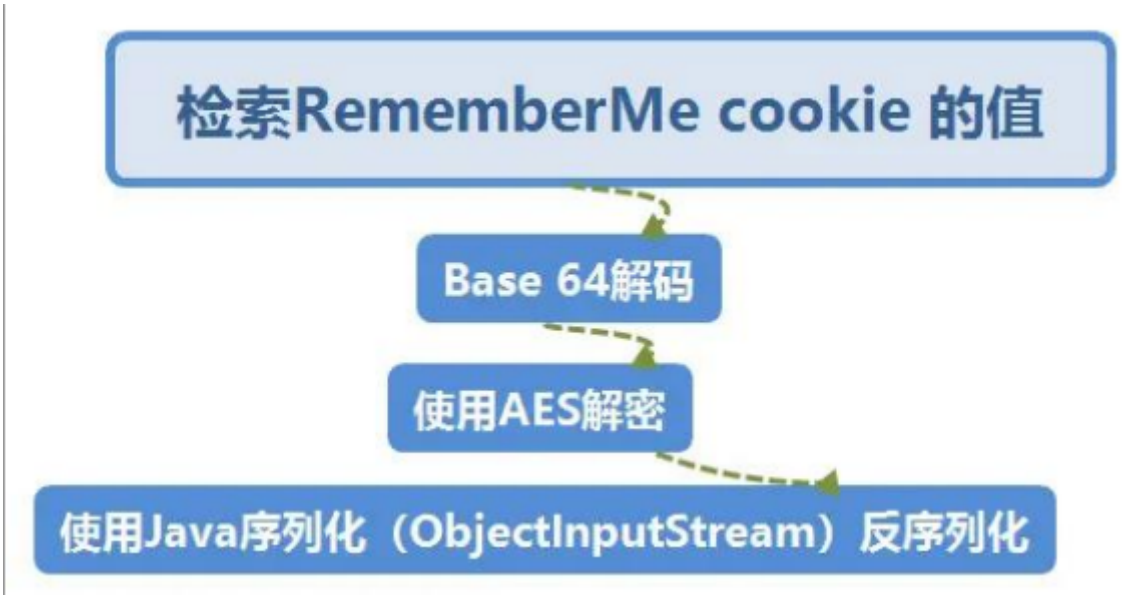
Angreifer können Informationen entschlüsseln, ohne den Verschlüsselungsschlüssel zu kennen
Intermediate Values == XOR result
Wenn wir **Intermediate Values** finden, ist die Entschlüsselung von Ciphertext einfach.
https://robertheaton.com/2013/07/29/padding-oracle-attack/

http://seffyvon.github.io/cryptography/2014/08/20/CBC-Padding-Oracle-Attacks/

Attack senerio: ??

Target: Apache Shiro
Older Shiro versions suffered from a Padding Oracle vulnerability, that when chained with a another deserialization-based vulnerability could result in remote code execution.

Shiro used the **AES-128-CBC** mode to
encrypt cookies enabling Padding Oracle attacks. It also used **CookieRememberMeManager** by default, which serialized, encrypted, and encoded the user's identity for later retrieval.



The Padding Oracle vulnerability can result in an attacker creating a malicious object, serializing it, encoding it and finally sending it as a cookie. Shiro will then decode and deserialize it. the deserialization implementation of the affected Shiro versions was also insecure, so
Padding Oracle vulnerability + deserialization-based vulnerability = RCE
-> page 23 ????

How to exploit on Apache shiro?

1. Capture the cookie
2. create a Payload with ysoserial with .class extension
3. Download the public exploit for the shiro version
4. command > `python shiro_exp.py http://targe.com/home.jsp the_capture_cookie ysoserial_payload.class`
5. After some time (minutes/hours) we get cookie contains our paylaod

Note: The Padding Oracle attack enabled the attack. Without it, the attack is not possible

Hash Length Extension Attack

Some Web App prepend a secret value to data. They hash this secret value with a flawed algorithm and the user the hashed value + the data, but not the secret value.

Attacker can calculate a valid hash for a
message without knowing the value of the secret by only guessing its length. Hashes are calculated in blocks and the hash of one block is the state for the next block.

if we identify the length of padding, we can calculate the new hash.

Leveraging machineKey

Subverting HMAC in Node.js