

IT-Security Tutorübung 04

Dorian Zedler

13. November 2023

Technische Universität München

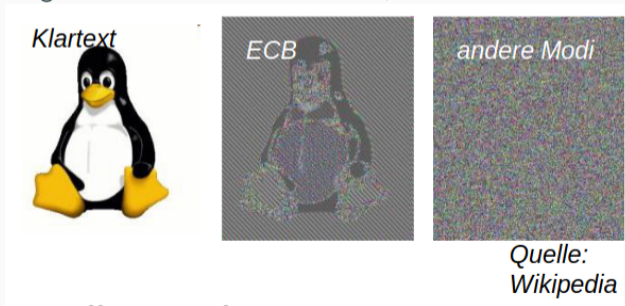
Aufgaben

Aufgaben

Aufgabe 1a - Betriebsmodi für Blockchiffren

a) Erklären Sie, warum Betriebsmodi für Blockchiffren notwendig sind!

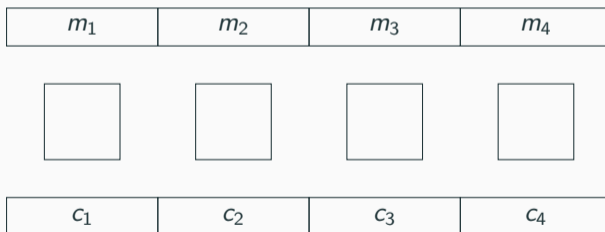
- Blockchiffren können nur einzelne Blöcke verschlüsseln
- Bei der unabhängigen Verschlüsselung mehrerer Blöcke kommt es zu Problemen (siehe TU03)
- Es gibt verschiedene Betriebsmodi, die diese Probleme lösen



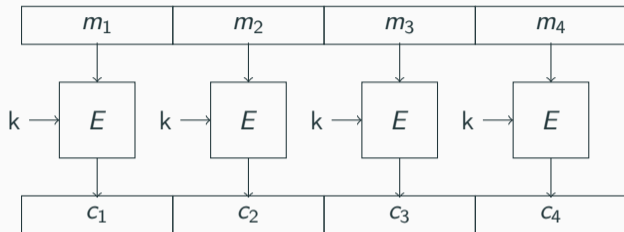
Aufgabe 1b - Betriebsmodi für Blockchiffren

b) Skizzieren Sie nun die Verschlüsselung für folgenden Betriebsmodi und nennen Sie jeweils einige Eigenschaften:

- Electronic Code Book (ECB)
- Cipher Block Chaining (CBC)
- Counter (CTR)



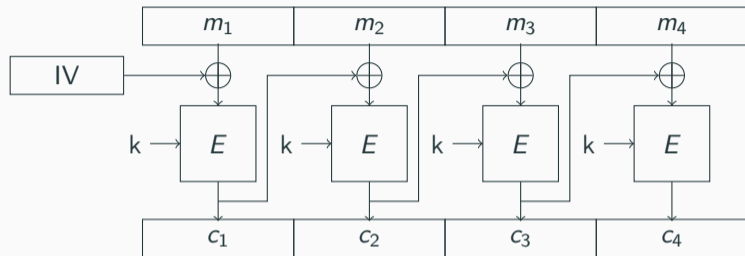
Aufgabe 1b - Electronic Code Book (ECB)



Typische Probleme mit ECB:

- Muster sichtbar
- Teile entschlüsselbar falls Klartext - Ciphertext Paare bekannt
- Ciphertext blockweise vertauschbar

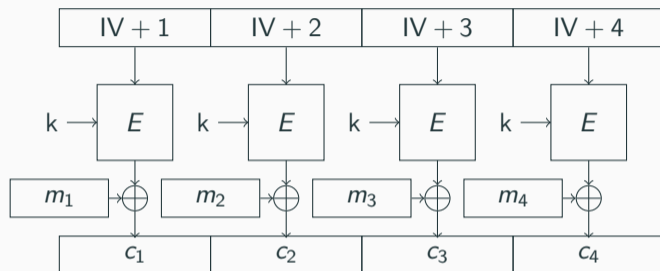
Aufgabe 1b - Cipher Block Chaining (CBC)



Eigenschaften CBC

- Muster nicht erkennbar
- Keine parallele Verarbeitung von Blöcken mehr möglich, aber auch kein Ändern der Reihenfolge
- Bitfehler in Chiphertextblock i wirken sich an der gleichen Stelle in Klartextblock $i+1$ aus (Siehe später: Padding Oracle)

Aufgabe 1b - Counter (CTR)



Eigenschaften CTR

- Blockchiffre ist nur zur Erzeugung von Schlüsselstrom
- Schlüsselstrom kann schon im Vorhinein berechnet werden
- Man muss nicht beim ersten Block anfangen

Aufgabe 2 - WEP

- WEP wurde früher für die WLAN-Verschlüsselung eingesetzt
 - Stromchiffre RC4 erzeugt Schlüsselstrom für jede Nachricht: $RC4(IV|K)$
 - K: symmetrischer Schlüssel, IV: Initialisierungsvektor mit 24 Bit
 - Verschlüsselung geschieht mit $c = m \oplus RC4(IV|K)$
 - bei Verschlüsselung von vielen Nachrichten wird irgendwann der IV wiederverwendet!

Aufgabe 2a - Probleme mit WEP

- a) Welche Folgen hat die Wiederverwendung des IV für die Sicherheit der Verschlüsselung?
- Key ist statisch (WLAN-Passwort)
→ bei identischem IV wird für verschiedene Pakete der gleiche Schlüsselstrom verwendet

Aufgabe 2b - Angriff auf WEP

- Ein Angreifer hat die verschlüsselte Nachricht c_1 abgehört
- Später hat er die ihm bekannte Nachricht M_2 zu c_2 verschlüsselt
- Mit WEP gilt:
 - $c_1 = M_1 \oplus RC4(IV|K)$
 - $c_2 = M_2 \oplus RC4(IV|K)$

b) Wie kann der Angreifer nun M_1 berechnen?

- XOR ist selbstinvers, $\rightarrow A = A \oplus B \oplus B$
- Schlüsselstrom lässt sich mit c_2 und M_2 berechnen:
$$c_2 \oplus M_2 = (M_2 \oplus RC4(IV|K)) \oplus M_2 = RC4(IV|K)$$
- Mit dem Schlüsselstrom kann man c_1 zu M_1 entschlüsseln:
$$c_1 \oplus RC4(IV|K) = (M_1 \oplus RC4(IV|K)) \oplus RC4(IV|K) = M_1$$

Aufgabe 3 - PKCS#7

- Padding wird verwendet, um die Länge des Klartextes auf ein Vielfaches der Blocklänge zu bringen
- Wenn k Bytes gepaddet werden müssen, werden k Bytes mit dem Wert k angehängt
- Beispiel:

I	T	-	S	i	c	h	e	r	h	e	i	t	\x03	\x03	\x03
I	T	-	S	i	c	h	e	r	h	e	i	\x04	\x04	\x04	\x04

- Wird beim Padding-Oracle zur Schwachstelle

Aufgabe 3a - PKCS#7

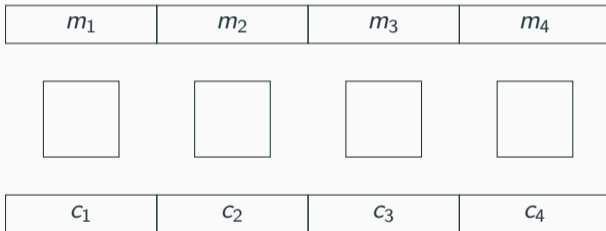
- a) Die Nachricht $M = \text{"Foobar"}$ soll mit AES (Blockgröße 16 Byte) verschlüsselt werden. Wie sieht diese Nachricht mit *PKCS#7*-Padding aus?
- Die Nachricht hat Länge 6, somit müssen 10 Bytes gepadded werden.
 - ASCII: foobar\n\n\n\n\n\n\n\n\n\n
 - Hex: 666f66261720a0a0a0a0a0a0a0a0a

b) Wie sieht die Nachricht $M = \text{"Das ist ein Test"}$ mit *PKCS#7*-Padding aus? Wie kann ein Empfänger entscheiden, ob es sich am Ende der Nachricht um Padding handelt oder nicht?

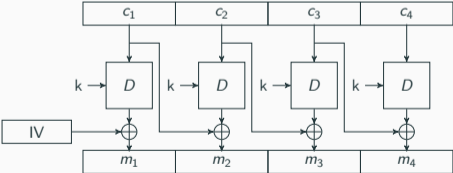
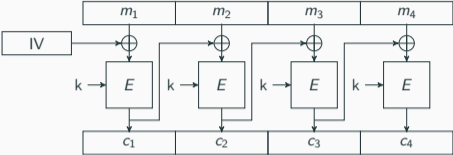
- ASCII: Das ist ein Text\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10
- Hex: 446173206973742065696e20546578741010101010101010101010101010101010

Aufgabe 3c - Entschlüsselung von CBC

- c) Mit der Verschlüsselung im CBC-Modus haben Sie sich bereits vertraut gemacht. Skizzieren Sie nun auch die Entschlüsselung!

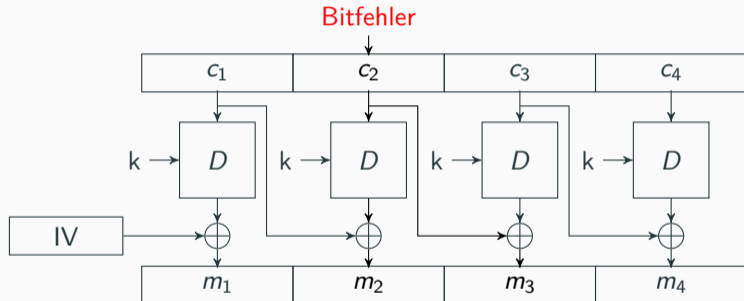


Aufgabe 3c - Entschlüsselung von CBC



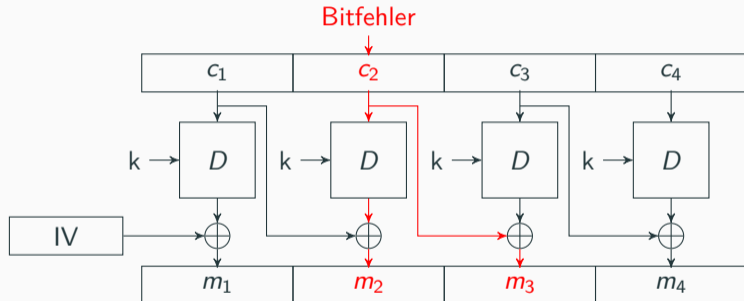
Aufgabe 3d - Bitfehler im CBC-Modus

- d) Wie wirken sich im CBC-Modus Bitfehler im Ciphertext auf den entschlüsselten Klartext aus?



Aufgabe 3d - Bitfehler im CBC-Modus

d) Wie wirken sich im CBC-Modus Bitfehler im Ciphertext auf den entschlüsselten Klartext aus?



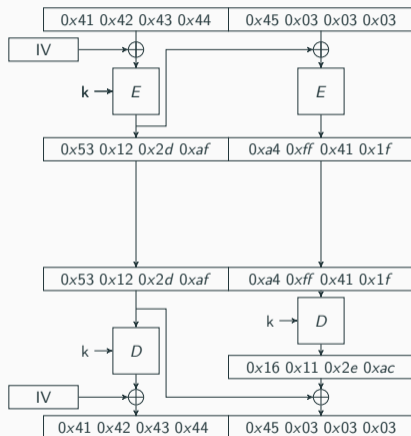
- Wegen Konfusion und Diffusion ist m_2 komplett verändert
- Bei m_3 ist nur das in c_2 veränderte Bit verändert

Aufgabe 3e - Padding-oracle

- e) Skizzieren Sie nun einen Padding-Oracle-Angriff: Wie könnte ein Angreifer die Fehlerausbreitung im CBC-Modus nutzen, um beliebige Nachrichten von einem *Padding-Oracle* entschlüsseln zu lassen?

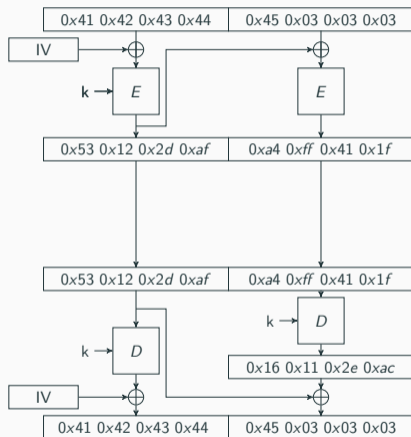
Aufgabe 3e - Padding-oracle

Verschlüsselung und
Entschlüsselung im
CBC-Modus



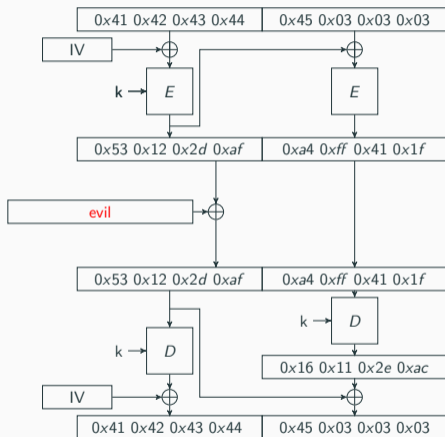
Aufgabe 3e - Padding-oracle

Blockgröße: 4 Byte
Klartext: "ABCDE"



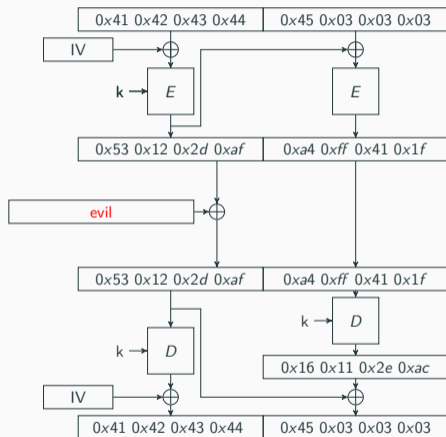
Aufgabe 3e - Padding-oracle

Blockgröße: 4 Byte
Klartext: "ABCDE"



Aufgabe 3e - Padding-oracle

Korrektes Padding, falls
 $m'_2 = ?$



Aufgabe 3e - Padding-oracle

Korrektes Padding, falls

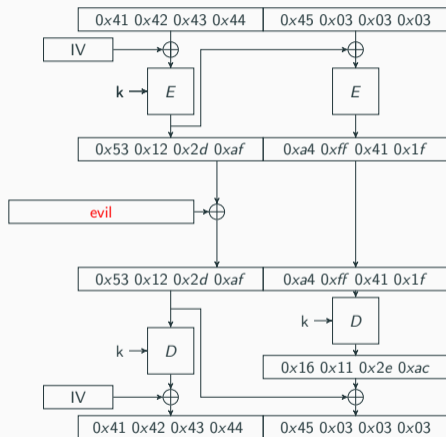
$m'_2 = ?$

0x04 0x04 0x04 0x04

0x__ 0x03 0x03 0x03

0x__ 0x__ 0x02 0x02

0x__ 0x__ 0x__ 0x01



Aufgabe 3e - Padding-oracle

Korrektes Padding, falls

$m'_2 = ?$

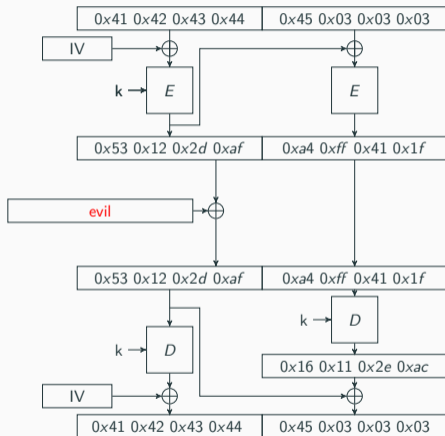
0x04 0x04 0x04 0x04

0x__ 0x03 0x03 0x03

0x__ 0x__ 0x02 0x02

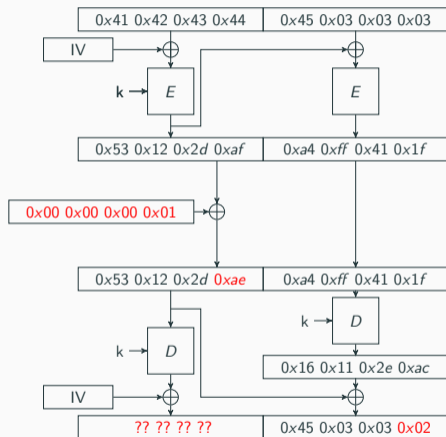
0x__ 0x__ 0x__ 0x01

⇒ **Bruteforce**



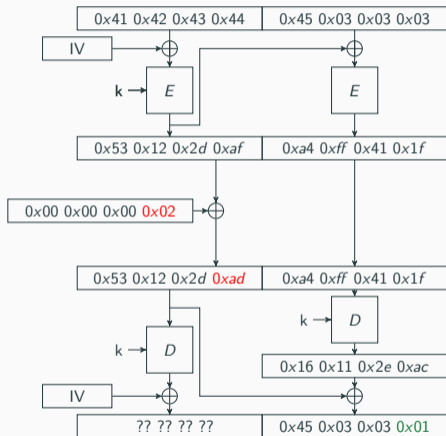
Aufgabe 3e - Padding-oracle

Bruteforce:
 $m'_2[3] = 0x01$



Aufgabe 3e - Padding-oracle

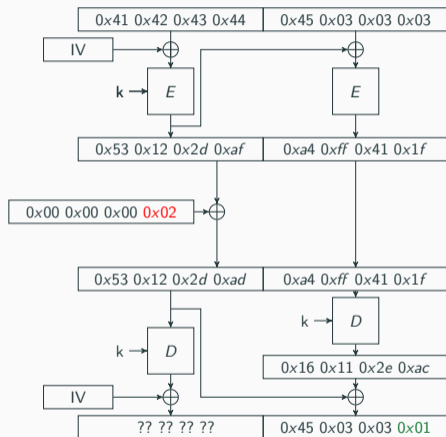
Bruteforce:
 $m'_2[3] = 0x01$



Aufgabe 3e - Padding-oracle

Padding korrekt:

$$m'_2[3] = 0x01$$

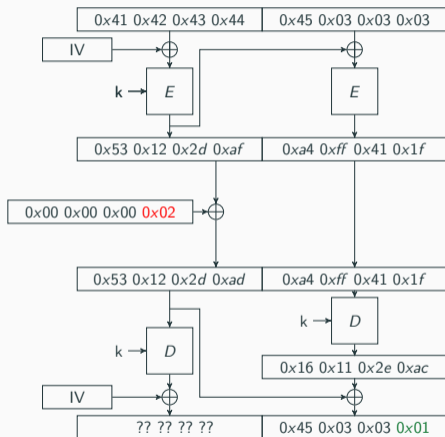


Aufgabe 3e - Padding-oracle

Padding korrekt:

$$m'_2[3] = 0x01$$

$$m_2[3] = 0x01 \oplus 0x02 = 0x03$$



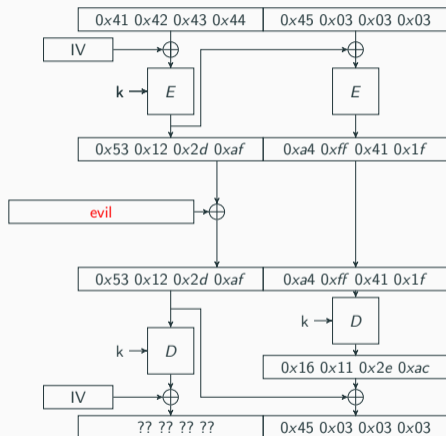
Aufgabe 3e - Padding-oracle

Bruteforce:

$$m'_2[2 : 3] = 0x02 \ 0x02$$

Wann gilt

$$m'_2[3] = 0x02?$$



Aufgabe 3e - Padding-oracle

Wann gilt

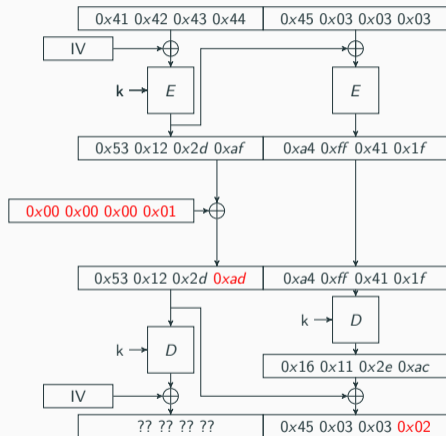
$$m'_2[3] = 0x02?$$

XOR propagiert:

$$c'_1 = c_1 \oplus e$$

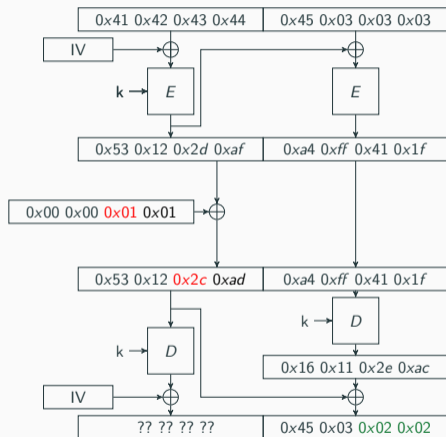
$$\rightarrow m'_2 = m_2 \oplus e$$

$$\begin{aligned} \rightarrow e[3] &= m_2[3] \oplus m'_2[3] \\ &= 0x03 \oplus 0x02 \\ &= \mathbf{0x01} \end{aligned}$$



Aufgabe 3e - Padding-oracle

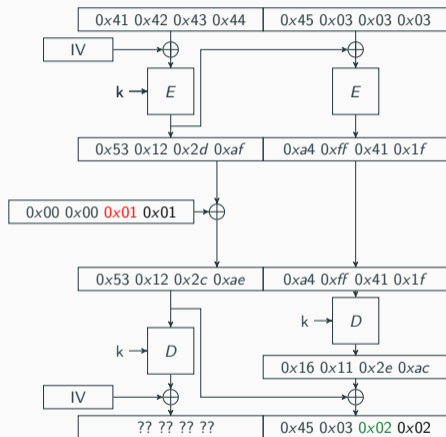
Bruteforce:
 $m'_2[2] = 0x02$



Aufgabe 3e - Padding-oracle

Padding korrekt:

$$m'_2[2] = 0x02$$

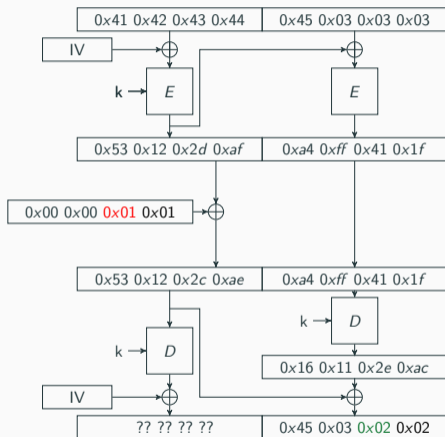


Aufgabe 3e - Padding-oracle

Padding korrekt:

$$m'_2[2] = 0x02$$

$$m_2[2] = 0x02 \oplus 0x01 = 0x03$$



Aufgabe 3e - Padding-oracle

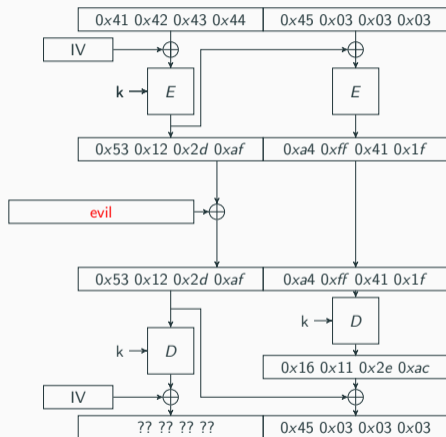
Bruteforce:

$$m'_2[1 : 3] = 0x03 \ 0x03 \ 0x03$$

Wann gilt

$$m'_2[3] = 0x03?$$

$$m'_2[2] = 0x03?$$



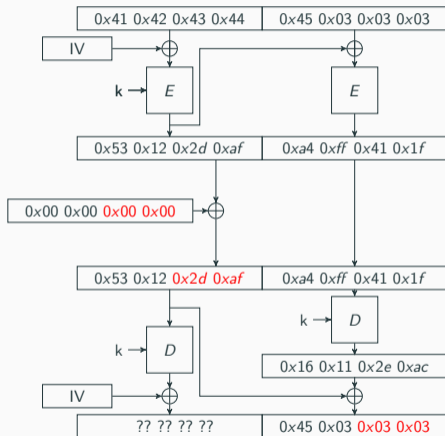
Aufgabe 3e - Padding-oracle

$$e[3] = 0x03 \oplus 0x03$$

$$= 0x00$$

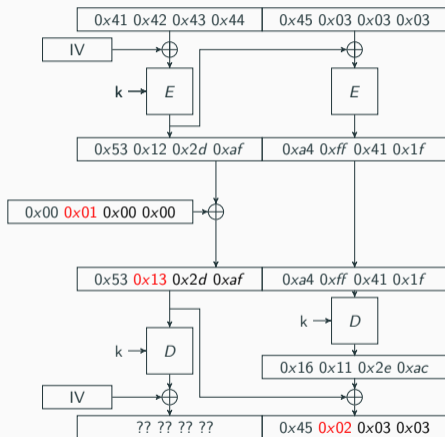
$$e[2] = 0x03 \oplus 0x03$$

$$= 0x00$$



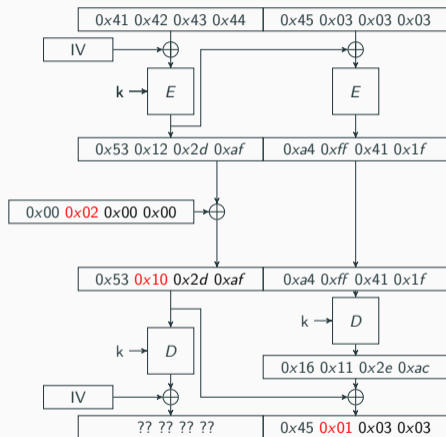
Aufgabe 3e - Padding-oracle

Bruteforce: $m'_2[1] = 0x03$



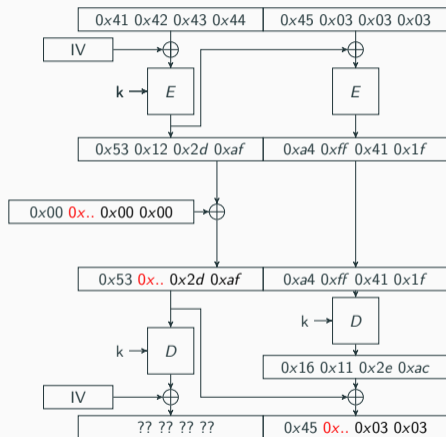
Aufgabe 3e - Padding-oracle

Bruteforce: $m'_2[1] = 0x03$



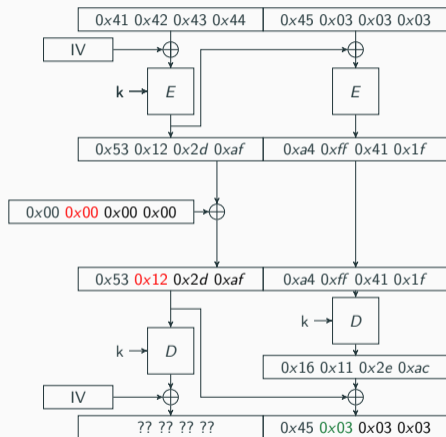
Aufgabe 3e - Padding-oracle

Bruteforce: $m'_2[1] = 0x03$



Aufgabe 3e - Padding-oracle

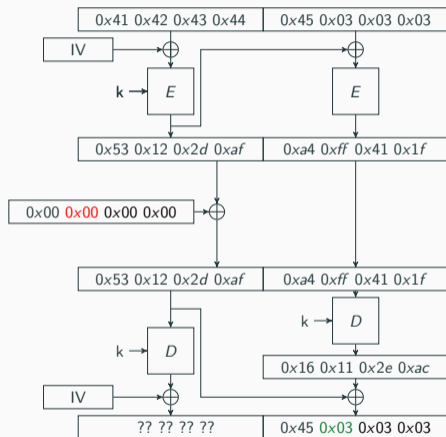
Bruteforce: $m'_2[1] = 0x03$



Aufgabe 3e - Padding-oracle

Padding korrekt:

$$m'_2[1] = 0x03$$

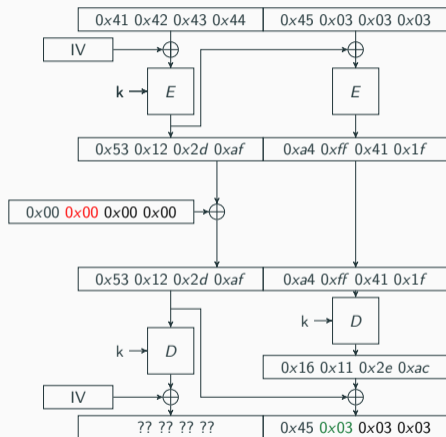


Aufgabe 3e - Padding-oracle

Padding korrekt:

$$m'_2[1] = 0x03$$

$$m_2[1] = 0x03 \oplus 0x00 = 0x03$$



Aufgabe 3e - Padding-oracle

Bruteforce:

$m'_2[0 : 3] =$

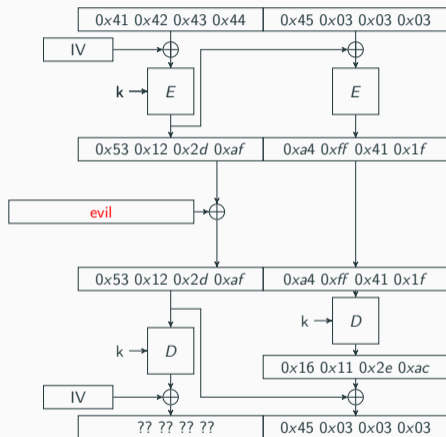
0x04 0x04 0x04 0x04

Wann gilt

$m'_2[3] = 0x04?$

$m'_2[2] = 0x04?$

$m'_2[1] = 0x04?$



Aufgabe 3e - Padding-oracle

$$e[3] = 0x03 \oplus 0x04$$

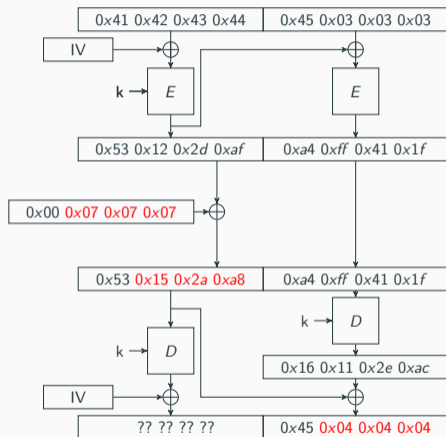
$$= 0x07$$

$$e[2] = 0x03 \oplus 0x04$$

$$= 0x07$$

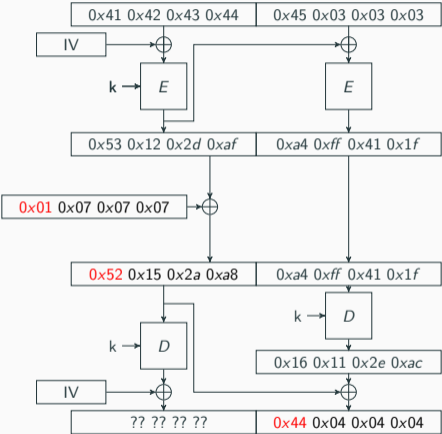
$$e[1] = 0x03 \oplus 0x04$$

$$= 0x07$$



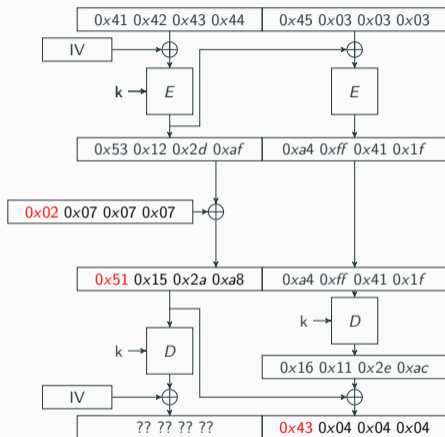
Aufgabe 3e - Padding-oracle

Bruteforce: $m'_2[0] = 0x04$



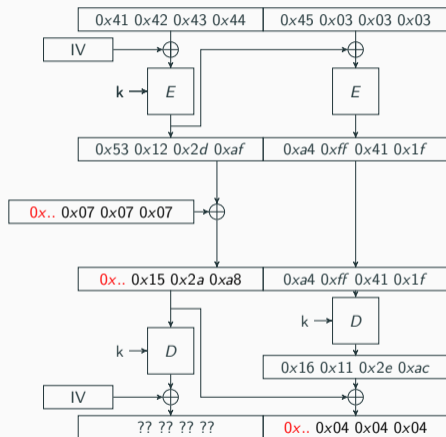
Aufgabe 3e - Padding-oracle

Bruteforce: $m'_2[0] = 0x04$



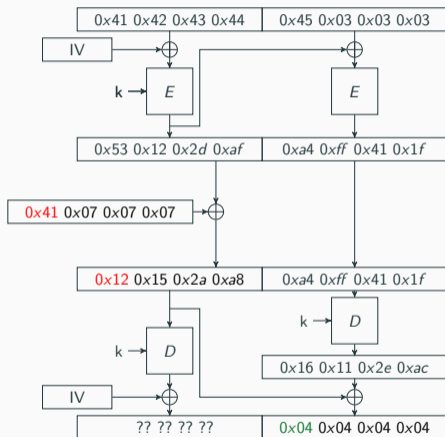
Aufgabe 3e - Padding-oracle

Bruteforce: $m'_2[0] = 0x04$



Aufgabe 3e - Padding-oracle

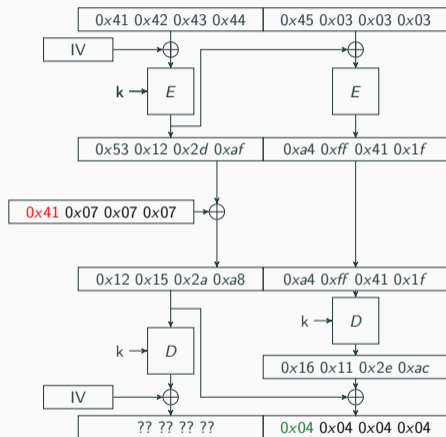
Bruteforce: $m'_2[0] = 0x04$



Aufgabe 3e - Padding-oracle

Padding korrekt:

$$m'_2[0] = 0x04$$

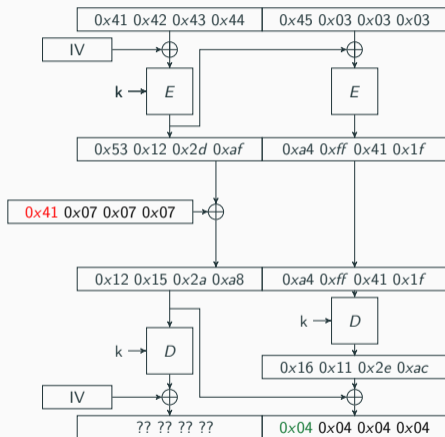


Aufgabe 3e - Padding-oracle

Padding korrekt:

$$m'_2[0] = 0x04$$

$$m_2[0] = 0x04 \oplus 0x41 = 0x45$$



Aufgabe 3e - Padding-oracle

- Wenn der n -te Block fertig entschlüsselt ist, wird er weggelassen
→ So kann der Angreifer den $n - 1$ -ten Block entschlüsseln, indem er den $n - 2$ -ten manipuliert
- So wird immer weiter gemacht, bis alle Blöcke entschlüsselt sind
- Beim 0-ten Block wird der IV manipuliert