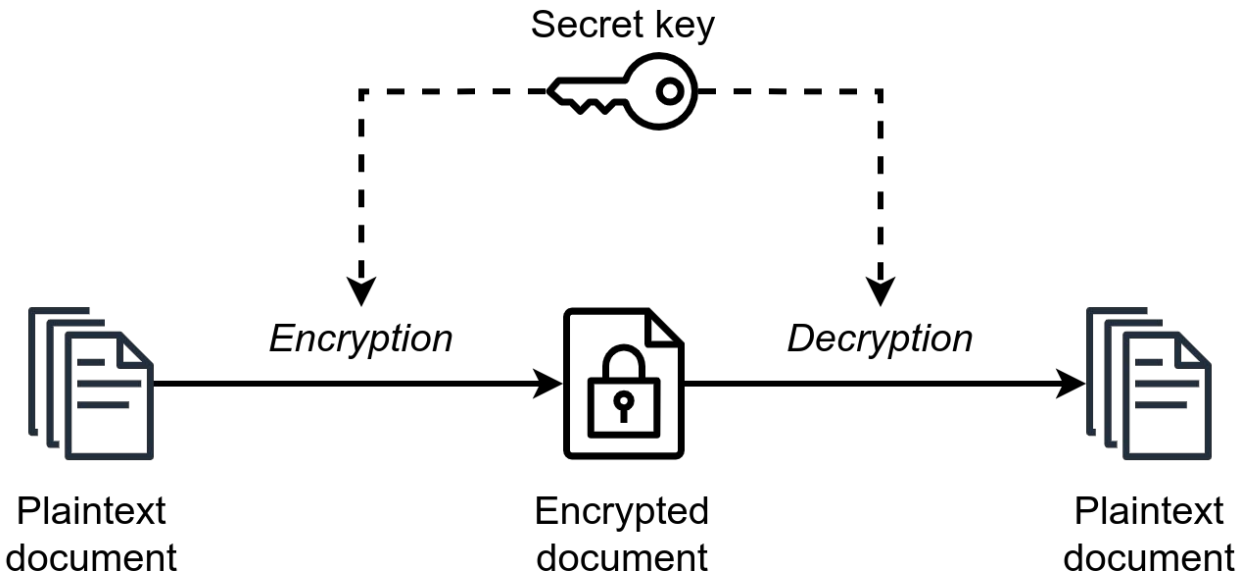
Συμμετρική Κρυπτογραφία

Συμμετρική κρυπτογραφία ⇨ ΚΟΙΝΟ κλειδί

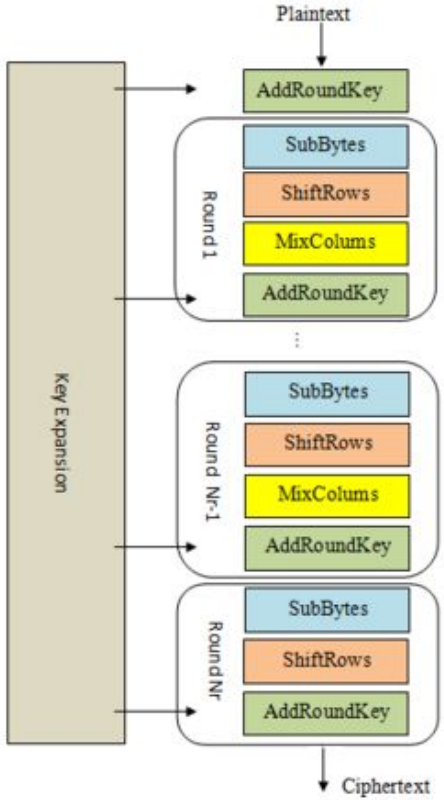


AES(Advanced Encryption Standard)

- Ο πιο ευρέως χρησιμοποιούμενος αλγόριθμος συμμετρικής κρυπτογράφησης
- Block cipher(κρυπτογραφεί σε “block” μεγέθους 128 bits, δλδ 16 bytes)
- Κλειδί μήκους 128, 192 ή 256 bits
- Μεγαλύτερο κλειδί = Περισσότερη ασφάλεια ΑΛΛΆ 128 bits υπερασφαλή

Εσωτερική Δομή -
Πως κρυπτογραφείται ένα block

- 10 γύροι με (περίπου) τα ίδια 4 operations
- 4 operations: **AddRoundKey**, **SubBytes**, **ShiftRows, MixColumns**
- **AddRoundKey**: XOR του κλειδιού για τον **εκάστοτε** γύρο με το state
- **SubBytes**: αντικατάσταση κάθε byte με ένα άλλο σύμφωνα με ενα lookup table γνωστό ως “S-BOX”(substitution box)
- **ShiftRows, MixColumns**: κάνουν ένα “ανακάτεμα” των bytes του state

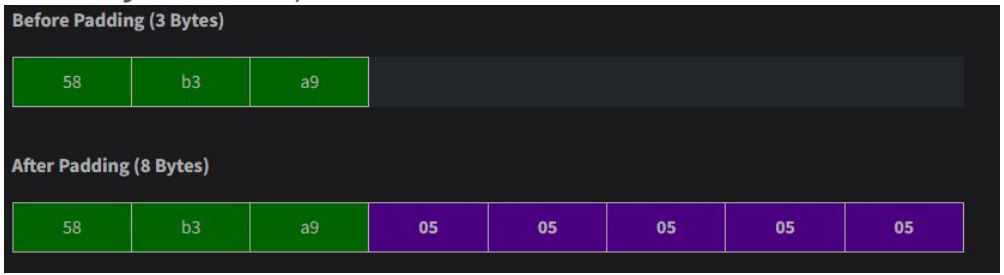


Μας νοιάζει η εσωτερική δομή?

- Ναι … αλλά κυρίως όχι(για τα ctf)
- Είναι **ασφαλής,** και εκτός αν πειράξει κανείς την εσωτερική δομή(πχ αφαιρέσει κάποιο από τα 4 operations τελείως) δε γίνεται να σπάσει
- Όμως, υπάρχουν πολλά vulnerabilities σχετικά με τον AES, αυτά δεν εμφανίζονται όταν κρυπτογραφεί κανείς **ένα** block αλλά στο πως επιλέγει να διαχειριστεί την κρυπτογράφηση **πολλών** blocks.
- Ο τρόπος διαχείρισης πολλών blocks είναι το **mode of operation**, κάποια βασικά ειναι τα ECB, CBC, CTR, ενώ το πιο συχνά απαντώμενο “στη φύση”(i.e. στο διαδίκτυο) είναι το GCM.

AES padding

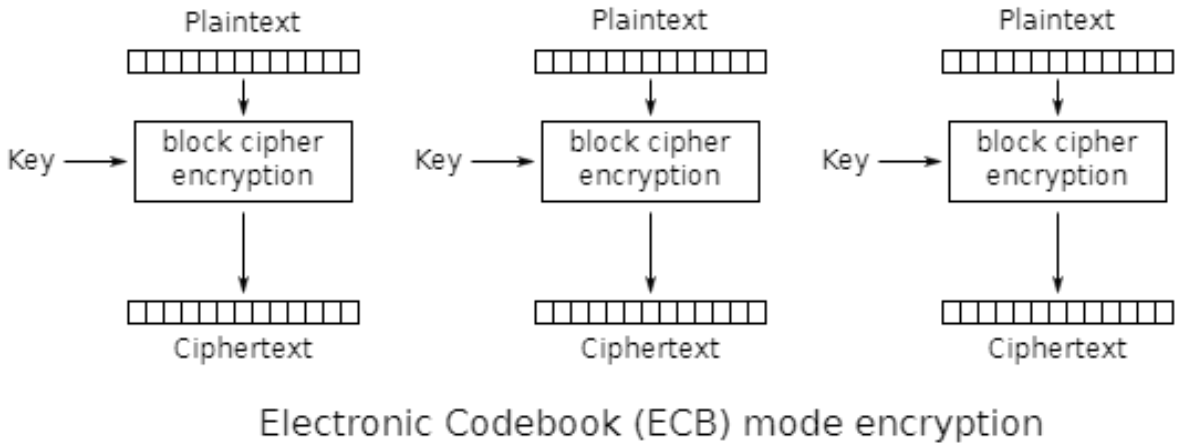
- Ο AES ειναι block cipher με blocksize = 16 bytes(128 bits)
- Αν ένα μήνυμα δεν είναι ακριβώς 16 bytes στο μήκος(η ακριβέστερα, πολλαπλάσιο του 16 σε μήκος) τότε πρέπει να βάλουμε **padding** πριν το κάνουμε encrypt
- Συνηθέστερο padding είναι το PKCS#7 το οποίο αν μας λείπουν x bytes προσθέτει x φορές το byte x.
- Παράδειγμα(για blocksize 8, **όχι** 16 όπως στον AES):



- Αν έχουμε **ακριβώς** οσα bytes όσο το blocksize τότε **ΠΑΛΙ** μπαίνει padding x φορές το byte x όπου x είναι το blocksize
- Βοηθάει στο unpadding. Γιατί?

Modes of operation: AES-ECB

- Κόβουμε το plaintext σε blocks μήκους 128 bits και encrypt το καθένα ξεχωριστά

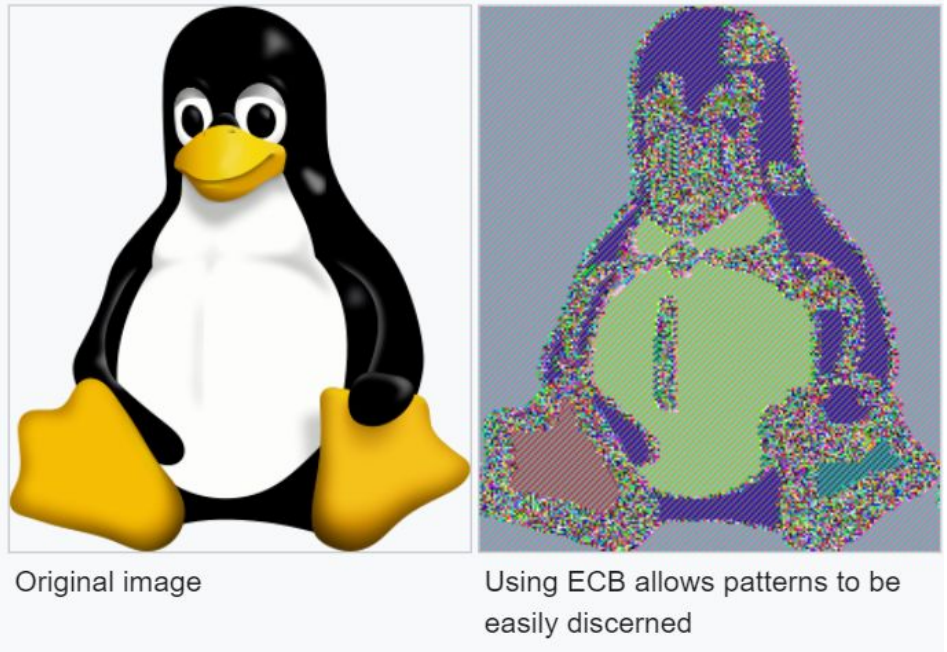


Electronic Codebook (ECB) mode encryption

Γιατί όχι ECB?

- $block_1 = block_2 \Rightarrow$

  $ENC(block_1, key) = ENC(block_2, key)$

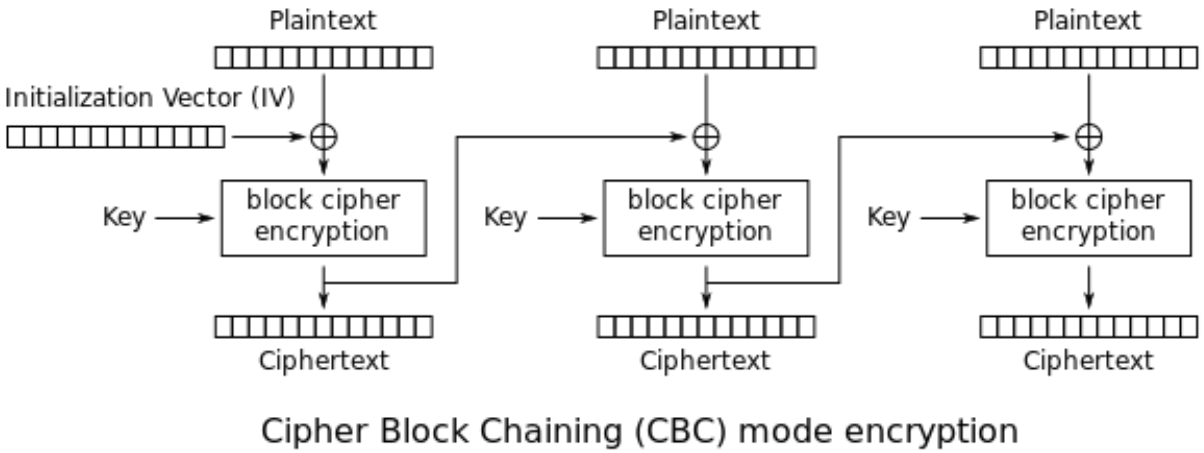


Original image

Using ECB allows patterns to be easily discerned

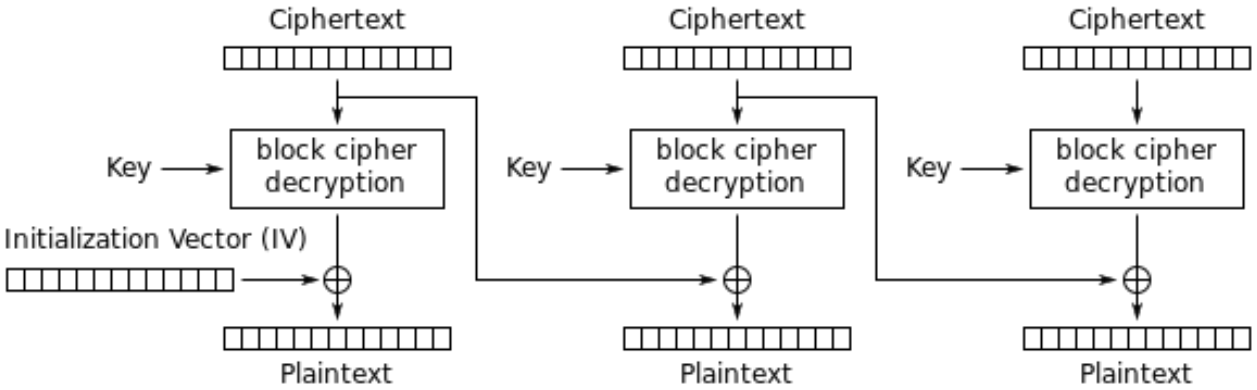
AES-CBC

- Extra δημόσια τιμή **IV**
- Κάθε block επηρεάζει το επόμενο!!!



Cipher Block Chaining (CBC) mode encryption

CBC - decryption



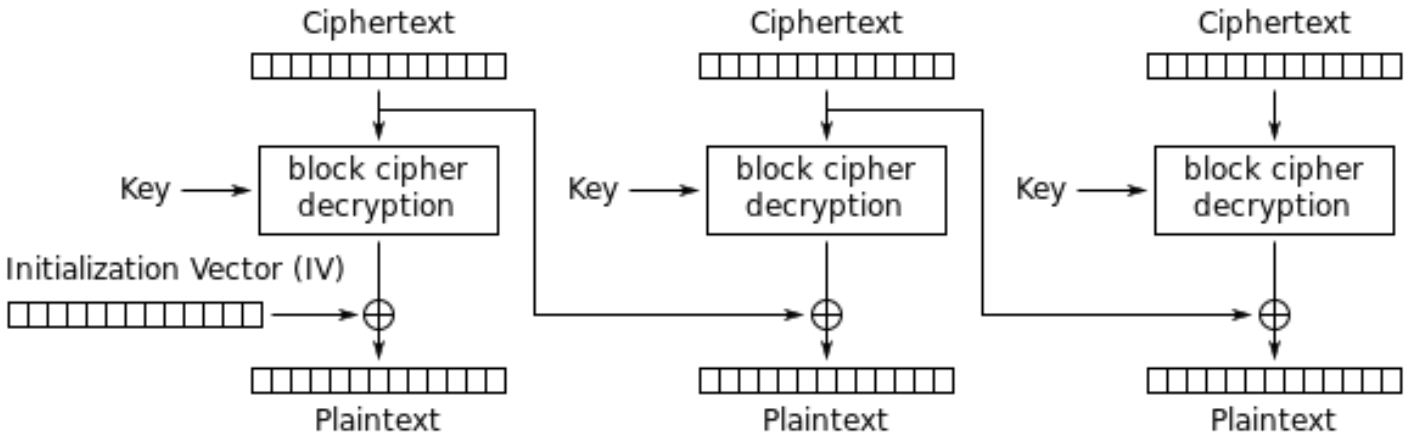
Cipher Block Chaining (CBC) mode decryption

Προβλήματα με CBC

Τα πιο συχνά attacks:

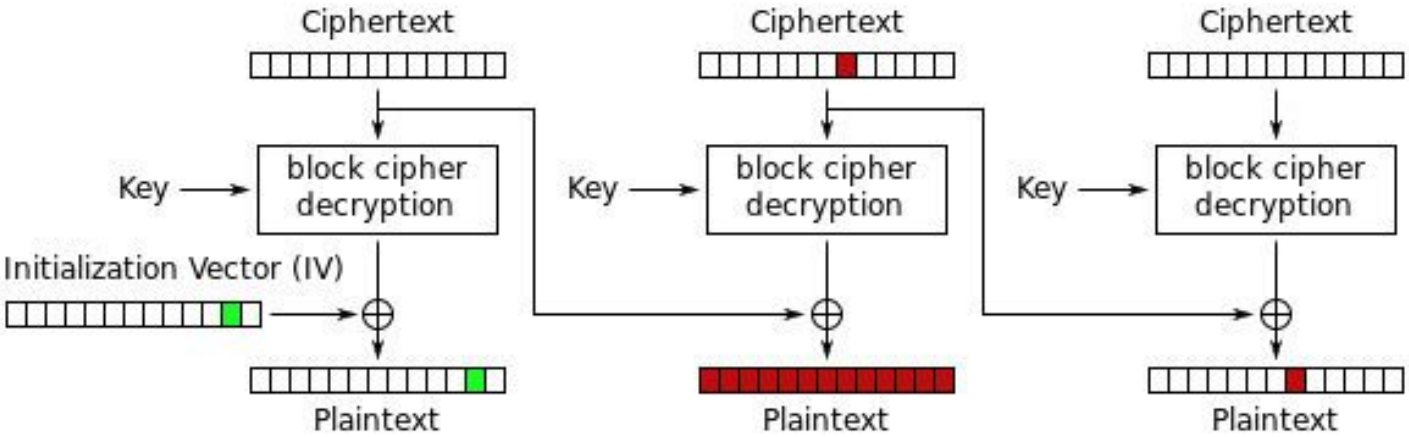
- CBC bit flipping
- CBC padding oracle

CBC bit flipping



Cipher Block Chaining (CBC) mode decryption

CBC bit flipping



Cipher Block Chaining (CBC) mode decryption

CBC bit flipping

- Όπως φαίνεται και στις εικόνες μπορούμε αν έχουμε ένα ciphertext encrypted με AES-CBC να το “πειράξουμε” με τέτοιο τρόπο ώστε ΌΤΑΝ γίνει decrypt να αλλάξει το plaintext από το αρχικό με **συγκεκριμένο**(και πιθανώς malicious hehehe) τρόπο.
- π.χ. Ένας server μας δίνει ένα cookie που είναι το αποτέλεσμα του **encryption** του json:

  `'{"admin": false, "username": "Bobos"}'` <- μήκος 37 bytes = 2*16 + 5, λείπουν 11 bytes για να έχουμε πολλαπλάσιο του 16 άρα με το **padding** το μήνυμα που γίνεται encrypt είναι:

  `'{"admin": false, "username": "Bobos"}\x0b\x0b\x0b\x0b\x0b\x0b\x0b\x0b\x0b\x0b\x0b'`

- Το cookie θα εχει το ciphertext που είναι 3 blocks, και το IV(είναι public parameter και το χρειάζεται ο server για να κάνει το decryption).
- Παρατηρούμε ότι το “σημαντικό” πεδίο του json(η τιμή του “admin”) είναι στο πρώτο block, άρα πειράζοντας με XOR το IV στο cookie μπορούμε εύκολα να θέσουμε “admin” = True!!!

Κώδικας για το παράδειγμα

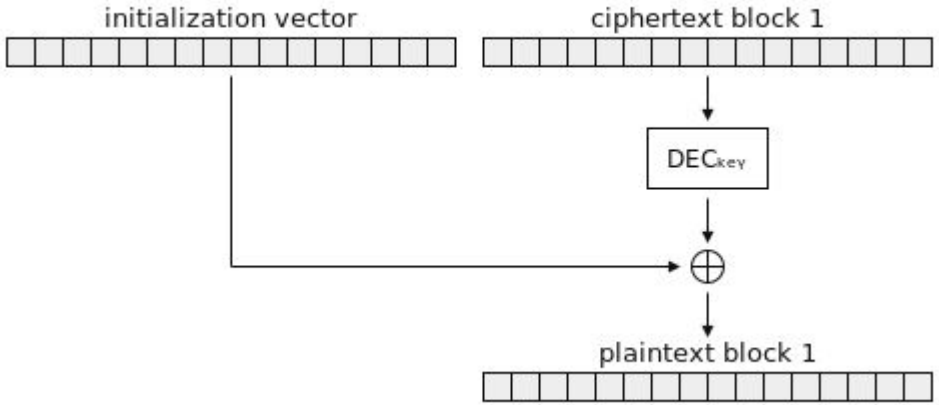
Βλ. κώδικα

CBC padding oracle

- Έστω ότι έχουμε ένα ciphertext που θέλουμε να κάνουμε decrypt και εναν server που του στέλνουμε κάποιο(οποιοδήποτε) ciphertext και μας απαντάει με το αν το corresponding plaintext έχει λάθος padding.
- Με αυτο το “oracle” και την τεχνικη του bit flipping μπορούμε να κάνουμε σταδιακά decrypt το ciphertext.

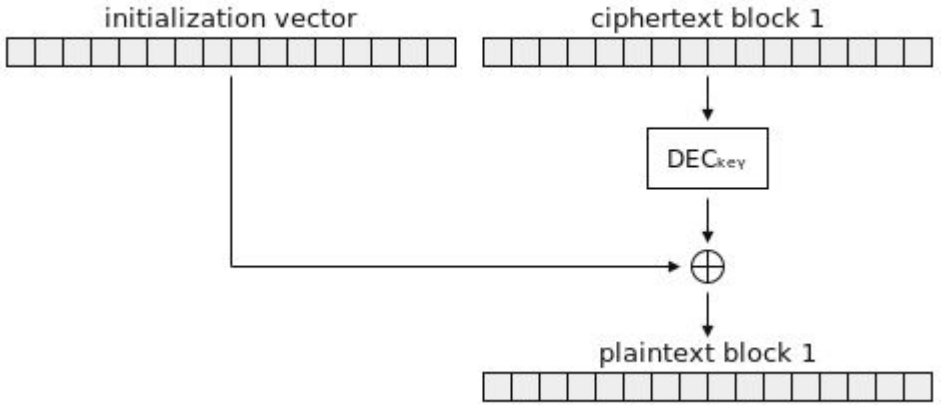
CBC padding oracle - single block case (1)

- Πειράζουμε το **τελευταίο** byte του IV δοκιμάζοντας όλες τις πιθανές 256 τιμές και στέλνουμε το newIV, ctxt block 1 στο σέρβερ που μας απαντάει αν το padding ειναι σωστό.
- Το plaintext θα προκύψει από το XOR του αποτελέσματος του AES decrypt και του newIV



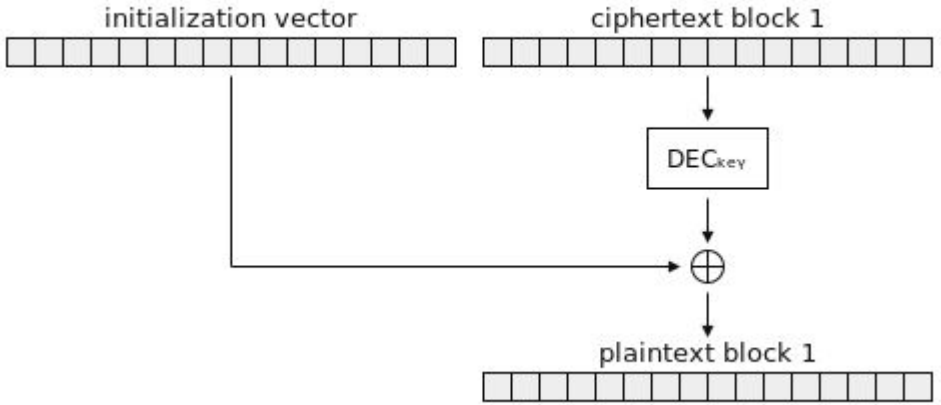
CBC padding oracle - single block case (2)

- Αν “IV_{15}” είναι το τελευταιο byte του IV και “D_{15}” το τελευταίο byte του αποτελέσματος του AES decrypt, τότε το τελευταίο byte του πειραγμένου plaintext θα είναι $P_{15}=(IV_{15}$ XOR $D_{15})$
- Δοκιμάζοντας όλες τις 256 τιμές τιμές για το IV_{15} κάποια στιγμή θα γίνει P_{15} = b’\x01’ το οποίο είναι valid PKCS#7 padding! Άρα απο το error msg μαθαίνουμε ότι (IV_{15} XOR D_{15}) = 1



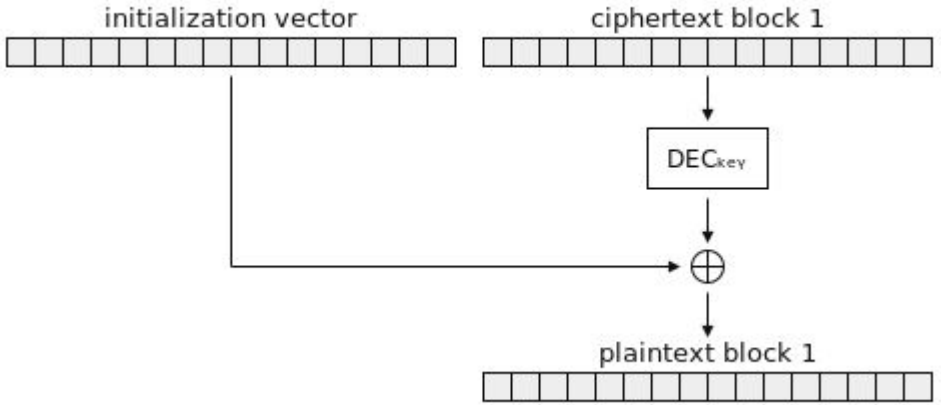
CBC padding oracle - single block case (3)

- $(IV_{15}$ XOR $D_{15}) = 1$
- Όμως το IV_{15} το διαλέξαμε εμείς, το γνωρίζουμε και άρα μπορούμε να βρούμε το $D_{15} = (IV_{15}$ XOR $1)$!!!
- Στη συνέχεια αφού ξέρουμε το D_{15} μπορούμε να θέσουμε με bit flipping το τελευταίο byte του plaintext block σε **όποια** τιμή θέλουμε πειράζοντας κατάλληλα το **τελευταίο** byte του IV



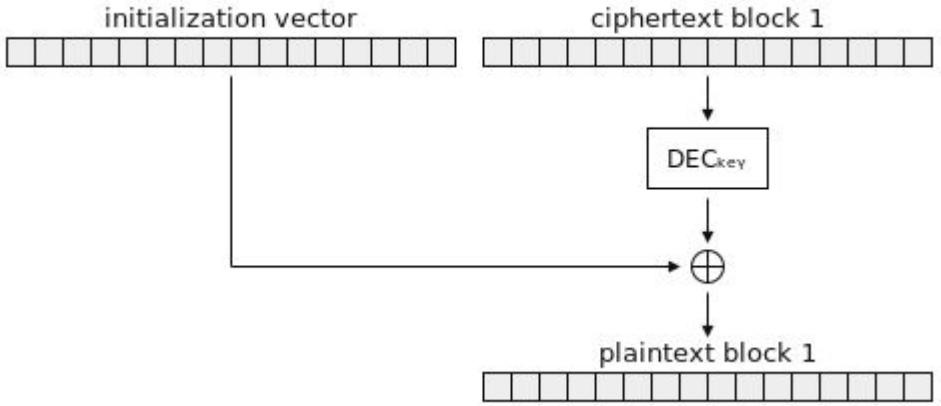
CBC padding oracle - single block case (4)

- Για το επόμενο βήμα θέτουμε κατάλληλο IV_{15} ώστε P_{15}= b”\x02”
- Ξαναδοκιμάζουμε όλες τις 256 πιθανές τιμές για το $\mathbf{IV_{14}}$ αυτή τη φορά.
- Κάποια στιγμή θα γίνει P_{14}= (IV_{14} XOR D_{14}) = 2
- Τότε **και τα δύο** τελευταία byte του plaintext block θα εχουν τιμη b”\x02”
- Όμως η κατάληξη b”\x02\x02” είναι πάλι valid PKCS#7 padding!!!



CBC padding oracle - single block case (5)

- Άρα θα λάβουμε διαφορετικό error message και **τότε** θα καταλάβουμε ότι ισχύει P_{14}= (IV_{14} XOR D_{14}) = 2
- Πάλι το IV_{14} το θέσαμε εμείς άρα μπορούμε να βρούμε το D_{14}=(2 XOR IV_{14})
- …
- Επαναλαμβάνουμε όλη αυτή τη διαδικασία και σιγά σιγά, byte-by-byte βρίσκουμε όλο το plaintext!!!
- Σημείωση: γίνεται και για περισσότερα blocks



Some resources

- Cbc padding oracle: https://research.nccgroup.com/2021/02/17/cryptopals-exploiting-cbc-padding-oracles/
- AES Course: https://cryptohack.org/courses/symmetric/
- Wikipedia modes of operation(παραδόξως τα λέει πολύ καλά): https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

That’s it!