**Padding oracle attack**

בהגשת תרגיל זה מצורף הסקריפט הפייתוני ex1.py המממש Padding oracle attack כפי שנלמד בתרגול.

הסקריפט מקבל את ה cyphertext , את ה key ואת הבלוק IV.

ה cyphertext הוא בעצם הטקסט המוצפן לפי CBC, הטקסט מחולק לבלוקים וכל בלוק מוצפן בעזרת המפתח (ה key שאותו הסקריפט מקבל), ובעזרת XOR עם הבלוק הקודם. לדוגמא הבלוק ה Ci ב cyphertext הוא בעצם E(Pi ⊕ Ci−1) כך ש Pi מייצג את הבלוק בplaintext , כלומר את תוכן הבלוק המקורי, Ci-1 את הבלוק המוצפן הקודם ו E מייצג את ההצפנה (כמובן לפי המפתח). בנוסף נראה שאת הבלוק הראשון אנו נצפין כך C1 = E(P1 ⊕ IV) ,מכיוון שאין בלוק קודם , ו IV מייצג בלוק כלשהו שבעזרתו אנו מצפינים רק את הבלוק הראשון וכאמור גם אותו הסקריפט מקבל בתחילת ריצתו.

כעת אתאר את ריצתו של הסקריפט, בהינתן ה cyphertext, ה key ו IV . תחילה :

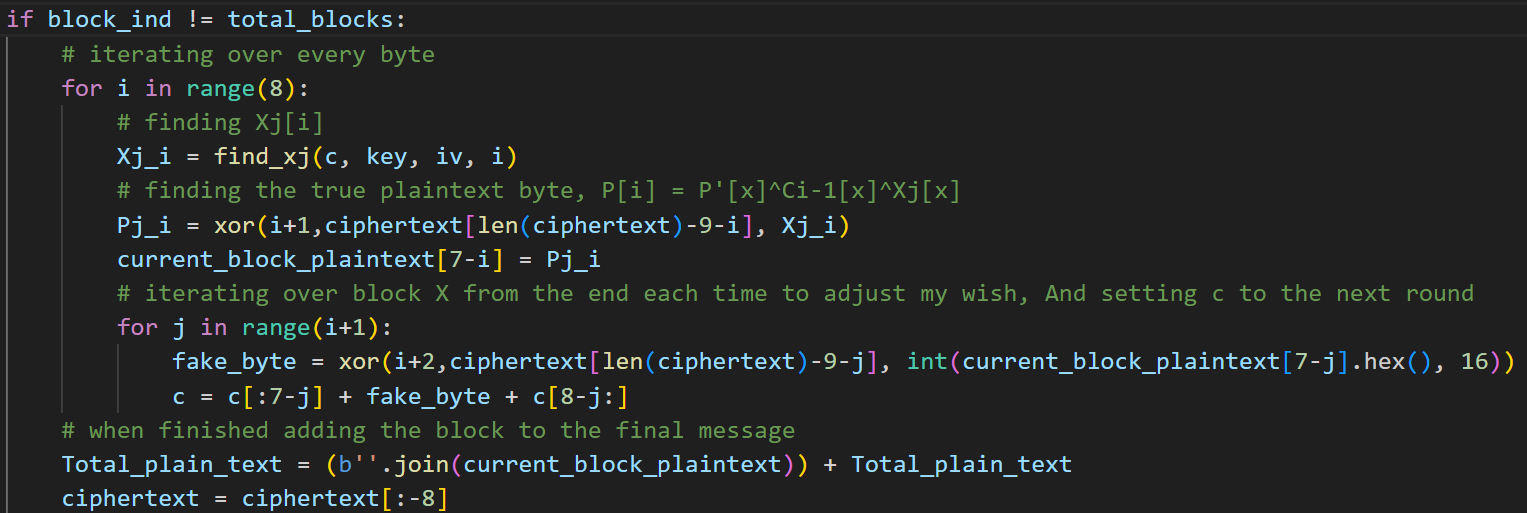
תמונה שמכילה טקסט, צילום מסך, גופן

התיאור נוצר באופן אוטומטי

Total\_plain\_text הוא משתנה שבסוף הריצה יכיל את ערך הPLAINTEXT המקורי בבתים כך שבתחילת הריצה הוא ריק, ובכל איטרציה נשרשר אליו את הבלוק שנחשוף.

לאחר מכן נתחיל לעבור על כל בלוק ב ciphertext בנפרד כך שאנו מתחילים מהבלוק האחרון.

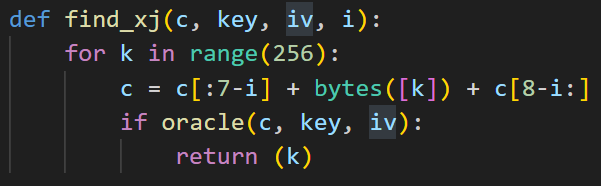
ניצור משתנה c שגודלו יהיה בגודל 2 בלוקים כך שהבלוק הראשון הוא בלוק אפסים, והבלוק השני הוא הבלוק ב cyphertext עליו אנו עוברים באיטרציה הזו.

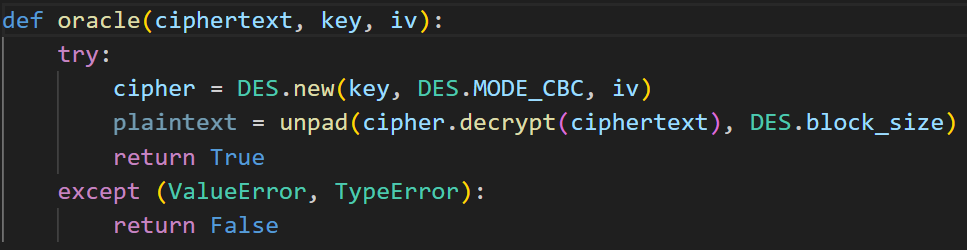
current\_plain\_text יהיה מערך שיכיל את הבתים שמייצגים את הטקסט המקורי בבלוק עליו אנו עובדים. בסוף האיטרציה מערך זה יכיל את ערך הבלוק המקורי בשלמותו.

תחילה, נבדוק האם אנו לא בבלוק הראשון, כזכור הבלוק הראשון מוצפן בעזרת הבלוק הנתון IV ואילו כל בלוק אחר מוצפן בעזרת הבלוק המוצפן שקדם לו – Ci-1 .

נניח שאנו לא בבלוק הראשון ובכך נכנס לתנאי וכעת נעבור בלולאה על כל הבתים בבלוק הנוכחי.

נזכיר ש c = Xj || Ci , כך שהרעיון יהיה למצוא את ערכי Xj כך שההצפנה תהיה חוקית ובכך לחלץ את הplaintext . וכעת נשלח לפונקציה find\_xj את c על מנת למצוא Xj[7-i] כך שc תהיה מוצפנת בצורה חוקית.





נראה שהפונקציה find\_xj בעצם רצה על כל האפשרויות בבית מסוים ב Xj , בהתאמה למיקום הבית אותו אנו מחפשים כעת ב cyphertext. כאמור בפעם הראשונה יהיה על הבית האחרון ובכל פעם שנקרא לפונקציה זו, עקב בניית הקוד, הוא יחפש את כל האפשרויות על בית אחד אחורה.

על כל צרוף שכזה ישאל האורקל האם ההצפנה של c חוקית. נראה שמכיוון ש Ci ב c הוא הבלוק האחרון הוא גם יצטרך להיות "כביכול" מרופד על מנת שההצפנה תהיה חוקית. ולכן באיטרציה הראשונה אנו נצפה שהאורקל יחזיר TRUE במצב שבו לאחר פענוח שלו הבית האחרון יהיה : b'x01' עקב הגדרת ההצפנה כפי שהסברתי קודם לכן.

האורקל פשוט מנסה לפענח את ה ciphertext לפי CBC, מפתח ההצפנה ו IV , במידה והצליח מחזיר True אחרת False . נציין שלאורקל בלבד יש את פונקציית הפענוח decrypt וכמובן שאם גם לנו כתוקפים היה אותה היינו יכולים ישירות לפענח את ה cyphertext .

כעת נתבונן על קטע הקוד הבא :

# finding the true plaintext byte, P[x] = P'[x]^Ci-1[x]^Xj[x]

  Pj\_i = xor(i+1,ciphertext[len(ciphertext)-9-i], Xj\_i)

כפי שהסברתי קודם לכן, מהגדרת ההצפנה המשוואה הזו מתקיימת :

P[7-i] = P'[7-i]^Ci-1[7-i]^Xj[7-i]

כך ש P' בעצם מייצג את מה שאילצנו את הPLAINTEXT להיות על מנת שהאורקל יחזיר TRUE .

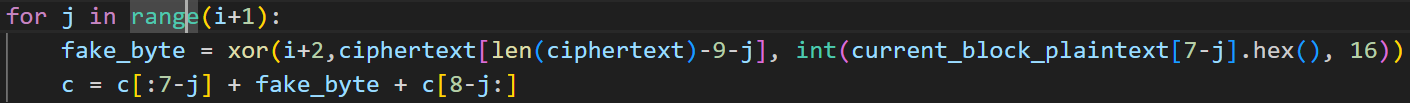
כאמור את P'[7-i] אנו יודעים בהתאם לאיטרציה, באיטרציה הראשונה נרצה שיהיה 1 מכיוון שאנו מסתכלים רק על הבלוק האחרון ונצפה שרק הוא ירופד ב1, באיטרציה השנייה נרצה ששני הבתים יהיו 2 וכן הלאה...

Ci-1[7-i] נתון לנו, ו Xj[7-i] זה הערך שהפונקציה find\_xj החזירה לנו. וכעת, מכיוון ששלושת הערכים הללו נתונים לנו, אנו יכולים להשתמש בפונקציית הXOR שבנינו ולחלץ את הPLAINTEXT המקורי במקום ה 7-i.

בנוסף, נכניס את Pj\_i למערך current\_block\_plaintext במקום ה 7-i וכעת נרצה להכין את c לסיבוב הבא. נשים לב למשוואה הבאה:

P ′ 2 [x] = Pi [x] ⊕ Ci−1[x] ⊕ Xj [x]

זו בעצם אותה משוואה מלפני כמה שורות רק שהחלפנו בין P ל P' , חילוף זה הוא חוקי עקב חוקי XOR . נתבונן בלולאה זו :



נראה שאנו מממשים את המשוואה שהצגתי כעת ומטרתנו היא בעצם להכין את c לסיבוב הבא כך שיהיה חוקי עד לבית אותו אנו מחפשים.

לדוגמא, אם אנו מנסים לפענח את הבית החמישי, נראה שההצפנה שתהיה חוקית היא ההצפנה שאם נפענח אותה עד המקום החמישי היא תהיה מרופדת ב b'x03' בכל שלושת הבתים האחרונים.

מיד לאחר מכן נחזור בלולאה לבית הבא אותו אנו רוצים לפענח, שהוא בעצם הבית הקודם, וכמובן רצף הפעולות יתבצע באופן דומה

בסיום מעבר על כל הבתים בבלוק הנוכחי, המערך current\_block\_plaintext יהיה מלא בכל שמונת ערכיו ויכיל את הplaintext המקורי ואותו נוסיף לTotal\_plain\_text/.

בנוסף נמחק את שמונת הבתים האחרונים ב cyphertext מכיוון שכבר פענחנו אותם ובכל איטרציה c נבנית בעזרת שמונת הבתים האחרונים של ה cyphertext ונרצה שכעת הם יהיו הבלוק הבא.

לאחר מכן נחזור בלולאה לעבור על שאר הבלוקים וכמובן התהליך יתבצע באופן דומה עבור כל בלוק.

נשים לב שבאיטרציה האחרונה, שבה בעצם אנו מפענחים את הבלוק הראשון, יתבצע תהליך זהה לחלוטין רק בכל פעם שביצענו פעולת XOR עם Ci-1 נחליף אותו ב IV מהסיבות המפורטות קודם לכן.

פענוח הבלוק הראשון כך שהשינויים מפענוח בלוק שאינו הראשון מסומנים :

תמונה שמכילה טקסט, צילום מסך, תוכנה, גופן

התיאור נוצר באופן אוטומטי



ובנוסף, אין סיבה למחוק את שמונת הבתים האחרונים של ה cyphertext מכיוון שסיימנו לעבור על כולו.

שורת הקוד האחרונה תהיה :

print(unpad(Total\_plain\_text, DES.block\_size).decode())

והיא תדפיס כמבוקש את הPLAINTEXT המקורי בתצורה טקסטואלית, ובנוסף תבצע unpad שיוריד את הריפוד המלאכותי שהתווסף לפני ההצפנה.