מבוא לקריפטוגרפיה / תרגיל בית 2#

אריאל סטולרמן
ודים סטוטלנד

(1)

להלן תיאור אלגוריתם לפענוח $DES^{V}_{k,k_1}(M)$:

יהי k_j ניחוש עבור k ו-(P_i, C_i) זוגות *known plaintext-ciphertext* נתונים. עבור כל ניחוש k_j נעבור על כל הזוגות (P_i, C_i) ונבצע את התהליך הבא:

- נצפין את P_i עם המפתח k_j בהצפנת DES רגילה, ונקבל $DES_{k_j}(P_i)$.
- נחשב את $DES_{k_j}(P_i) \oplus C_i$ ונקבל k_1' שהוא מועמד עבור k_1, שכן $DES_{k_j}(P_i) \oplus C_i = DES_{k_j}(P_i) \oplus (DES_{k_j}(P_i) \oplus k'_1) = k'_1$.
- עבור אותו k_j, אם כל ה-k_1' (המועמדים ל-k_1) זהים, ניתן לקבוע בהסתברות גבוהה (יפורט בהמשך) כי k_j, k_1' הם ה-k, k_1 הנכונים.

ישנם סה"כ 2^{56} ניחושים עבור k_j; אם נשתמש בשני זוגות (P_i, C_i) שונים, כיוון שניתן להניח כי DES היא פונקציה פסאודו-ראנדומית, אז הסיכוי בהינתן $k_j \neq k$ לקבל k_1' זהה עבור שני הזוגות הוא $\frac{1}{2^{64}}$. כמובן שאם $k_j = k$ אז כל k_1' המתקבלים אמורים להיות זהים. אם ניקח l זוגות (P_i, C_i) אז הסיכוי לטעות הוא $\frac{1}{2^{64(l-1)}}$, וזהו סיכוי קטן מאוד כבר עבור l קטן.

להלן אלגוריתם לפענוח $DES^{W}_{k,k_1}(M)$:

תחת אותם סימונים מקודם, עבור כל ניחוש של k_j נעבור על כל הזוגות (P_i, C_i) ונבצע את התהליך הבא:

- נפעיל את $DES^{-1}_{k_j}(C_i)$.
- נחשב את $DES^{-1}_{k_j}(C_i) \oplus P_i$ ונקבל k_1' שהוא מועמד עבור k_1, שכן $DES^{-1}_{k_j}(C_i) \oplus P_i = P_i \oplus k_1' \oplus P_i = k_1'$.
- עבור אותו k_j, אם כל ה-k_1' זהים, ניתן לקבוע בהסתברות גבוהה כי k_j, k_1' הם ה-k, k_1 הנכונים.

משיקולים דומים לאלגוריתם הקודם, ההסתברות שבהינתן $k_j \neq k$ ו-l זוגות (P_i, C_i) ההסתברות להגיע למסקנה ש-$k_1' = k_1$ היא $\frac{1}{2^{64(l-1)}}$.

(2)

נסמן k_1 המפתח שאיתו נצפין ו-k_2 המפתח שאיתו נפענח. נחלק את כל האפשרויות עבור k_2 שהם 2^{56} לבלוקים של 2^{40}, סה"כ 2^{16} בלוקים. עבור כל בלוק וזוג *plain-text, ciphertext* נבצע את התהליך הרגיל של *Meet in the middle*:

- עם כל המפתחות k_2 בבלוק נפענח את C_i ונשמור בזיכרון שלרשותינו.
- נעבור על כל האפשרויות עבור k_1, סה"כ 2^{56}, נצפין את P_i ונחפש התאמה לאחד מהפענוחים השמורים.
- אם מצאנו התאמה נשמור את זוג המועמדים עבור k_1, k_2.

הנכונות נובעת מנכונות ההתקפה הישירה של *meet in the middle*, שכן התהליך נשאר אותו דבר למעט סיבוכיות הזמן והמקום שהשתנתה. סה"כ *decryptions* בתהליך: $2^{56} = \underset{=\#blocks}{2^{16}} \cdot 2^{40}$; סה"כ *encryptions*: $2^{56} \cdot 2^{16} = 2^{72}$. כלומר, סה"כ מספר ההצפנות שצריך לבצע בתהליך גדל, בעוד מספר הפענוחים נשאר כמו בהתקפה המקורית. מספר זוגות ה-(P_i, C_i) הנדרשים לא שונה מההתקפה המקורית, בגלל שימור התהליך כמו בהתקפה המקורית (רק פרסנו אותו על גבי יותר פעולות).

(3)

(a) תהי A קבוצה ריקה. נבצע את התהליך הבא:

- נגריל מחרוזת m_i באורך 128 ביטים.
- נצפין את h_0 תחת m_i כמפתח: $h_1^{m_i} := AES_m(h_0)$.
- נבדוק האם קיים j כך ש-$h_1^{m_j} \in A$ וגם $h_1^{m_j} = h_1^{m_i}$ (ו-$m_i \neq m_j$ כמובן). אם כן, אז m_i, m_j הן הודעות המתנגשות עבור H. אם לא, נוסיף את $h_1^{m_i}$ ל-A ונמשיך.

מספר האיטרציות שנדרש אליהם עד מציאת התנגשות בהסתברות גבוהה הוא כשורש מרחב ההודעות (בדומה לפרדוקס היומולדת), כלומר $\sqrt{2^{128}} = 2^{64}$.

(b) בהינתן הודעה m באורך n בלוקים, נסמן $H(m) = h_n$ כאשר h_n היא מחרוזת בגודל בלוק (128 ביטים). נמצא הודעה $m' := m_1 m_2$ באורך 2 בלוקים המקיימת $H(m') = H(m)$ באמצעות *meet-in-the-middle attack*, באופן הבא:

- נאתחל קבוצה A וקבוצה B, שתיהן ריקות.
- נגריל m' הודעה באורך שני בלוקים עד שנקבל $m' \neq m$ כזו.
- נבדוק האם $AES_{m_1}\left(\vec{0}\right) = AES_{m_2}^{-1}(h)$, או האם $AES_{m_1}\left(\vec{0}\right)$ מתאים לאיבר מ- B או האם $AES_{m_2}^{-1}(h)$ מתאים לאיבר מ- A. אם כן, סיימנו. אחרת נוסיף את $AES_{m_1}\left(\vec{0}\right)$ לקבוצה A ואת $AES_{m_2}^{-1}(h)$ לקבוצה B ונמשיך.

הגרלת הודעה m' באורך שני בלוקים שקולה להגרלת שתי הודעות בנות 128 ביטים מתוך מרחב בגודל 2^{128} הודעות. בדומה לפרדוקס היומולדת, לאחר $\sqrt{2^{128}} = 2^{64}$ ניסיונות נמצא בסיכוי גבוה שתי הודעות המקיימות זאת (תחת הנחה ש- AES היא פרמוטציה פסאודו ראנדומית התלויה במפתח לפיו פועלת). הודעה m' המקיימת זאת כמובן מקיימת $H(m') = AES_{m_2}\left(AES_{m_1}\left(\vec{0}\right)\right) = h$, ולכן m' מהווה התנגשות עם m עבור H.

(4)

(a) להלן אלגוריתם למציאת $\left(w, MAC_k(w)\right)$ עבור $w \notin \{z_1, \dots, z_s\}$ בזמן פולי' בגודל הבלוק n:

עבור z_1, z_2 מילים בגודל בלוק יחיד (n ביטים), בהינתן $\left(z_1, MAC_k(z_1)\right)$ ו-$\left(z_2, MAC_k(z_2)\right)$ כאשר $z_2 = MAC_k(z_1)$, הזוג $\left(w, MAC_k(z_2)\right)$ הוא זוג טוב כאשר $w = z_1 \circ \vec{0}$ ($\vec{0}$ הוא בלוק אפסים בגודל n ביטים):

- נשים לב כי $C_1 = MAC_k(z_1) = E_k\left(z_1 \oplus \vec{0}\right)$.
- כמו כן $C_2 = MAC_k(z_2) = MAC_k(C_1) = E_k\left(C_1 \oplus \vec{0}\right)$.
- מתקיים: $MAC_k\left(z_1 \circ \vec{0}\right) = C_2$ כיוון שהבלוק הראשון הוא z_1, ומופעל $E_k\left(z_1 \oplus \vec{0}\right)$ שהוא ידוע – C_1, ועל הבלוק השני, שהוא $\vec{0}$, מופעל $E_k\left(C_1 \oplus \vec{0}\right)$ שהוא גם ידוע – C_2. לכן הזוג $\left(w, MAC_k(z_2)\right)$ הוא זוג חוקי ו-$w \notin \{z_1, z_2\}$.

(b) להלן אלגוריתם למציאת $\left(w, MAC(w)\right)$ עבור $w \notin \{z_1, \dots, z_s\}$ בזמן פולי' בגודל הבלוק n:

נסמן $MAC_k(m) = t_m$. בהינתן הזוגות:

- $\left(z_1, t_{z_1}\right)$
- $\left(z_2, t_{z_2}\right)$
- $\left(z', t_{z'}\right)$ עבור $z' = z_1 \circ \underbrace{0 \dots 01}_{n\ bits} \circ z_3$

כאשר z_i מילים שרירותיות בגודל n ו-$z_1 \neq z_2$, נראה כיצד למצוא את (w, t_w) עבור $w = z_2 \circ \underbrace{0 \dots 01}_{n\ bits} \circ \left[t_{z_1} \oplus t_{z_2} \oplus z_3\right]$. נפתח תחילה את $t_{z'}$:

1. $\vec{0}, z_1 \to E_k\left(\vec{0} \oplus z_1\right) = E_k(z_1)$
2. $E_k(z_1), 0 \dots 01 \to E_k(0 \dots 01 \oplus E_k(z_1))$
3. $E_k(0 \dots 01 \oplus E_k(z_1)), z_3 \to E_k(z_3 \oplus E_k(0 \dots 01 \oplus E_k(z_1)))$
4. *Adding block size (3):* $E_k(z_3 \oplus E_k(0 \dots 01 \oplus E_k(z_1))), 0 \dots 011 \to E_k(0 \dots 011 \oplus E_k(z_3 \oplus E_k(0 \dots 01 \oplus E_k(z_1))))$
   $= E_k\left(0 \dots 011 \oplus E_k\left(z_3 \oplus t_{z_1}\right)\right) = t'$

כמו כן מתקיים כי $t_{z_1} = E_k(0 \dots 01 \oplus E_k(z_1))$ ו-$t_{z_2} = E_k(0 \dots 01 \oplus E_k(z_2))$.

כעת נפתח את t_w:

1. $\vec{0}, z_2 \rightarrow E_k\left(\vec{0} \oplus z_2\right) = E_k(z_2)$
2. $E_k(z_2), 0 \dots 01 \rightarrow E_k(0 \dots 01 \oplus E_k(z_2)) = t_{z_2}$
3. $t_{z_2}, \left[t_{z_1} \oplus t_{z_2} \oplus z_3\right] \rightarrow E_k\left(t_{z_1} \oplus t_{z_2} \oplus z_3 \oplus t_{z_2}\right) = E_k\left(t_{z_1} \oplus z_3\right)$
4. *Adding block size (3):* $E_k\left(t_{z_1} \oplus z_3\right), 0 \dots 011 \rightarrow E_k\left(0 \dots 011 \oplus E_k\left(t_{z_1} \oplus z_3\right)\right)$

כאשר הביטוי האחרון שקיבלנו הוא t', כלומר כבר ברשותנו מהנתונים. כיוון ש-$z_1 \neq z_2$ אז $w \notin \{z_1, z_2, z'\}$, וברשותינו $\left(w, MAC_k(w)\right)$, כנדרש.

(*c*) להלן אלגוריתם למציאת $\left(w, MAC(w)\right)$ עבור $w \notin \{z_1, \dots, z_s\}$ בזמן פולי' בגודל הבלוק n:

נסמן $t_m = MAC_{k,k'}(m)$. יהיו z_1, z_2 שתי מילים שרירותיות באורך n (גודל בלוק יחיד) שלא שתיהן $\vec{0}$. בהינתן זוגות המילים וה-*MAC* שלהן למילים:

- $z_1 \circ z_1$
- z_2
- $z_1 \circ z_1 \circ \left[t_{z_2} \oplus t_{z_1 \circ z_1}\right]$

נראה כיצד למצוא את (w, t_w) עבור $w = z_2 \circ \underset{n\ bits}{\vec{0}}$. נפתח תחילה את ה-*MAC* של המילים הנתונות:

עבור $z_1 \circ z_1$:

1. $\vec{0}, z_1 \rightarrow E_k\left(\vec{0} \oplus z_1\right) = E_k(z_1)$
2. $E_k(z_1), z_1 \rightarrow E_k(z_1 \oplus E_k(z_1))$
3. *Adding k':* $E_k(z_1 \oplus E_k(z_1)) \oplus k'$

עבור z_2:

1. $\vec{0}, z_2 \rightarrow E_k\left(\vec{0} \oplus z_2\right) = E_k(z_2)$
2. *Adding k':* $E_k(z_2) \oplus k'$

עבור $z_1 \circ z_1 \circ \left[t_{z_2} \oplus t_{z_1 \circ z_1}\right]$:

1. $\vec{0}, z_1 \rightarrow E_k\left(\vec{0} \oplus z_1\right) = E_k(z_1)$
2. $E_k(z_1), z_1 \rightarrow E_k(z_1 \oplus E_k(z_1))$
3. $E_k(z_1 \oplus E_k(z_1)), \left[t_{z_2} \oplus t_{z_1 \circ z_1}\right] \rightarrow E_k\left(\left[t_{z_2} \oplus t_{z_1 \circ z_1}\right] \oplus E_k(z_1 \oplus E_k(z_1))\right) =$
   $E_k(E_k(z_2) \oplus k' \oplus \boldsymbol{E_k(z_1 \oplus E_k(z_1))} \oplus k' \oplus \boldsymbol{E_k(z_1 \oplus E_k(z_1))}) = E_k(E_k(z_2) \oplus \boldsymbol{k'} \oplus \boldsymbol{k'}) = E_k\left(E_k(z_2)\right)$
4. *Adding k':* $E_k\left(E_k(z_2)\right) \oplus k'$

נראה כי עבור $w = z_2 \circ \vec{0}$ מתקיים כי t_k גם כן שווה ל-$E_k\left(E_k(z_2)\right) \oplus k'$:

1. $\vec{0}, z_2 \rightarrow E_k\left(\vec{0} \oplus z_2\right) = E_k(z_2)$
2. $E_k(z_2), \vec{0} \rightarrow E_k\left(\vec{0} \oplus E_k(z_2)\right) = E_k\left(E_k(z_2)\right)$
3. *Adding k':* $E_k\left(E_k(z_2)\right) \oplus k'$

כיוון ש-z_1, z_2 אינן שתיהן $\vec{0}$ אז $w \notin \left\{z_1 \circ z_1, z_2, z_1 \circ z_1 \circ \left[t_{z_2} \oplus t_{z_1 \circ z_1}\right]\right\}$ וברשותינו (w, t_w), כנדרש.

(6)

(a) לכל $p > 2$ ראשוני מתקיים כי p הוא אי זוגי ולפיכך לכל $k > 1$ גם p^k אי זוגי. מכאן ש-$p^k - 1$ הוא זוגי, ונסמן $p^k - 1 = 2r$ כלומר $r = \frac{p^k-1}{2}$.

לכל $x \in GF^*(p^k)$ מתקיים $x^{p^k-1} \equiv 1$ ב-$GF^*(p^k)$ ולכן: $x^{p^k-1} = x^{2r} = (x^2)^r \equiv 1$, וכיוון שסדר כל איבר מחלק את סדר החבורה, הסדר של (x^2) הוא לכל היותר $r = \frac{p^k-1}{2} < p^k$ ל-p, k בתחום המבוקש. לפיכך $(x^2) \in GF^*(p^k)$ אינו פרימיטיבי.

(b) אם נסתכל על ייצוג $GF^*(p^k)$ מעל שדה הפולינומים, אז כל פולינום מדרגה 0 שיסומן $f(x) = i \cdot x^0$ עבור $2 \leq i \leq p-1$ הוא איבר לא פרימיטיבי, כיוון שמתקיים: $(a \cdot x^0)^{p-1} = a^{p-1} = 1 (\mathrm{mod}\, p)$ ב-$\mathbb{Z}_p^*$. כיוון שאיברי $GF^*(p^k)$ הם פולינומים מעל $GF(p)$, העלאת $p-2$ הפולינומים הנ"ל ב-$p-1$ תתן את הפולינום הקבוע 1. לפיכך סדר אותם איברים הוא $p-1 < p^k - 1$ ולכן איברים אלו אינם פרימיטיביים.

(7)

(a)

```
p = 249*(2^249)-1
is_prime(p)
> True
factor(p-1)
> 2 * 708211533214392631 * 4932122799315549593711974 3 * 3224344192787689425940157396942471
```

(b)

```
def is_primitive_root(p, g):
    if is_prime(p) == False:
        print "p is not prime"
    else:
        f = factor(p-1)
        for i in range(len(f)):
            exp = (int)((p-1)/f[i][0])
            pow_mod = power_mod(g,exp,p)
            if pow_mod == 1:
                return False
        return True

# print is_primitive_root(p,g)
# mod(g,p).multiplicative_order()
```

הפונקציה נבדקה ע"י שתי השורות האחרונות בקוד לעיל עבור p, g שונים. כאשר פלט הפונקציה היה $True$, הסדר שהוחזר היה $p-1$, וכאשר פלט הפונקציה היה $False$, הסדר שהוחזר היה קטן ממנו.

(c)

```
p = 249*(2^249)-1
g = randint(10^7+1,10^8)
print(g)
f = factor(p-1)

def is_primitive_root(p, g, f):       # adding factor(p-1) to the input
    if is_prime(p) == False:
        print "p is not prime"
    else:
        for i in range(len(f)):
            exp = (int)((p-1)/f[i][0])
            pow_mod = power_mod(g,exp,p)
            if pow_mod == 1:
                return False
        return True

is_primitive_root(p,g,f)
is_primitive_root(p,g+1,f)
print p
print mod(g,p).multiplicative_order()
print mod(g+1,p).multiplicative_order()
> 34845892     # = g
> True         # g is a primitive element of Z*p
> False        # g+1 is not a primitive element of Z*p
> 22525179859446666140991543177471319574581426704487890973300733139039351000268 7
> 22525179859446666140991543177471319574581426704487890973300733139039351000268 6  # = p-1, as expected
> 11262589929723333070495771588735659787290713352243945486650366569519675500134 3  # != p-1, as expected
```

(*d*)

```
q1 = next_prime(2^400)
p1 = 2*q1 + 1
while is_prime(p1) == False:
    q1 = next_prime(q1)
    p1 = 2*q1 + 1
print q1
print p1
>
2582249878086908589655919172003011874329705792829223512830659356540647622016841194629645353280137831435903171972
747524733
>
5164499756173817179311838344006023748659411585658447025661318713081295244033682389259290706560275662871806343945
495049467

q2 = next_prime(q1)
p2 = 2*q2 + 1
while is_prime(p2) == False:
    q2 = next_prime(q2)
    p2 = 2*q2 + 1
print q2
print p2
>
2582249878086908589655919172003011874329705792829223512830659356540647622016841194629645353280137831435903171972
747631671
>
5164499756173817179311838344006023748659411585658447025661318713081295244033682389259290706560275662871806343945
495263343
```

(8)

(*a*) להלן הוכחה כי עבור $f_0(x) \coloneqq g^x \bmod p\,, f_1(y) = a \cdot g^y \bmod p$ כאשר $a, g, x, y \in \mathbb{Z}_p^*$, g יוצר של $\mathbb{Z}_p^*$ ו-p ראשוני, מתקיים כי f_0, f_1 הן *claw-free permutations*:

ידוע כי קשה למצוא z כך ש-$a = g^z$. נניח בשלילה כי f_0, f_1 אינן *claw-free*, אזי קל למצוא $x, y \in \mathbb{Z}_p^*$ כך ש-$g^x \equiv a \cdot g^y (mod\ p)$. כיוון ש-$g$ יוצר של $\mathbb{Z}_p^*$ אז קיים z כך ש-$a = g^z \bmod p$ אזי: $g^x \equiv g^z \cdot g^y (mod\ p)$ ולכן $g^x \equiv g^{y+z} (mod\ p)$. נכפיל את האגפים ב-$g^{(p-1)-x}$ ונקבל: $1 \equiv g^{(p-1)-x+y+z} (mod\ p)$. כיוון ש-$g^{\phi(p)} = g^{p-1} = 1$ אז אפשר למצוא z ע"י פתרון המשוואה $(p-1) - x + y + z = p - 1$ כלומר $z = x - y$. אם מספר זה יוצא שלילי, ניתן להוסיף לו $p-1$ כיוון ש-$g^{p-1+l} = g^{p-1} \cdot g^l \equiv g^l (\mathrm{mod}\, p)$ (מספיק להוסיף כפולה אחת של $p-1$ אם יוצא z שלילי, כי $x, y \in \mathbb{Z}_p^*$ ולכן $\min\{z\} = 1 - (p-1) = 2 - p$ ואז $2 - p + (p-1) = 1$, שכבר אינו שלילי).

כיוון שקשה לחשב את z כך ש-$a = g^z$ אז f_0, f_1 הן כן *claw-free permutations*.

(*b*) יהיו f_0, f_1 *claw-free permutations*, וידוע כי קשה למצוא z כך ש-$f_0(z) = IV \in D$ או $f_1(z) = IV \in D$. נניח בשלילה כי H אינה עמידה בפני התנגשויות, אזי נוכל למצוא בקלות $m_1 \neq m_2$ כך ש-$H(m_1) = H(m_2)$. נראה כי ניתן בקלות למצוא זוג $x, y \in D$ כך ש-$f_0(x) = f_1(y)$ (כלומר f_0, f_1 אינן *claw-free*) או $z \in D$ כך ש-$f_0(z) = IV \in D$ או $f_1(z) = IV \in D$, בסתירה לנתונים.

יהיו m_1, m_2 כך ש-$m_1 \neq m_2$ ו-$m_1 = b_1 b_2 \dots b_n$ ו-$m_2 = c_1 c_2 \dots c_k$ ביטים ובה"כ $k \leq n$). עבור $1 \leq i \leq k$ נבצע את התהליך הבא:

- כל עוד $b_i = c_i$ נחשב את $H(b_{i+1} \dots b_n)$ ואת $H(c_{i+1} \dots c_k)$, שהן כמובן שוות.
- אם הגענו ל-i כך ש-$b_i \neq c_i$, בה"כ $b_i = 0, c_i = 1$, אזי נחשב את $x \coloneqq H(b_{i+1} \dots b_n)$ ו-$y \coloneqq H(c_{i+1} \dots c_k)$ ולמעשה מתקיים $f_0(x) = f_1(y)$, בסתירה ל-*claw-freeness* של f_0, f_1.
- אם לא נתקלנו ב-i עבורו $b_i \neq c_i$ אז למעשה "הפשטנו" את m_2 עד למחרוזת $IV \in D$, ואז מתקיים ש-$H(b_{k+1} \dots b_n) = IV$, כלומר $f_{k+1}\big(H(b_{k+2} \dots b_n)\big) = IV$, כלומר מצאנו $z = H(b_{k+2} \dots b_n)$ המקיים $f_{k+1}(z) = IV \in D$ עבור $k+1 \in \{0,1\}$, וזה בסתירה לקושי במציאת z כזה.

(5) חלקי בלבד

(*a*) הוכחה כי $\gcd(a,m)=1 \Leftrightarrow ord_m(a)<\infty$:

יהי $m=p_1^{k_1}\cdot\ldots\cdot p_l^{k_l}$. נניח תחילה כי $\gcd(a,m)=1$ אז קיימים x,y כך ש-$ax+my=1$ ו-x הוא ההופכי הכפלי של a ב-$\mathbb{Z}_m^*$, כלומר $a\in\mathbb{Z}_m^*$.

ידוע כי לכל $x\in\mathbb{Z}_m^*$ מתקיים $ord_m(x)|ord(\mathbb{Z}_m^*)=\phi(m)$ כאשר $\phi(m)=\Pi_{i=1}^{l}\left(p_i^{k_i}-p_i^{k_i-1}\right)\in\mathbb{N}$, כלומר מספר סופי. מכאן $ord_m(a)<\infty$.

נניח כעת כי $ord_m(a)=k$ עבור $k\in\mathbb{N}$ כלשהו.

...

- (*b*)