

האם הצפנת RSA נשברה?

05/06/2003 | דואר חשמלי | גיליון מספר 14

משה הלוי *

halemo@actcom.co.il

מתמטיקאי מתוסכל ומתכנת מובטל בשם איציק (שם בדוי) חושב שהוא עלה על הדרך לקראת פיצוח ההצפנה החזקה בעולם: הצפנת ה-RSA. הצפנת RSA מבוססת על הרעיון של מפתח ציבורי ומפתח פרטי, המורכבים משני מספרים ראשוניים מאוד גדולים המוכפלים אחד בשני ויוצרים את החלק הארי של מפתח ההצפנה הציבורי. הרעיון מאחורי המפתח הציבורי הוא שלא ניתן לפרק את המפתח לשני גורמיו הראשוניים מכיוון שזו שיטה שלוקחת הרבה מאוד זמן במושגים של שנים, גם אם משתמשים במחשב חזק מאוד.

רעיון המפתח הציבורי והמפתח הפרטי

במשך מאות שנים הוצפנו בשיטות שונות מסרים שהועברו בין אנשים, ממשלות וצבאות. הבעיה העיקרית בהצפנות הללו, שבמשך הזמן הלכו והשתפרו בחוזקם, היתה בעיית הפצת מפתח ההצפנה (שהיה גם מפתח הפענוח).

על מנת שמקבל ההודעה המוצפנת יידע לפענח אותה, עליו היה לדעת מראש מה הוא המפתח שבו הוצפנה ההודעה, ועם אותו מפתח היה עליו לפתוח את ההודעה ולקרוא אותה בצורה מובנת.

הבעיה של הפצת המפתחות נעשתה קריטית כאשר המרחקים היו גדולים בין השולח למקבל, וכאשר היתה סכנה כי המפתח עלול ליפול בידי האויב המאזין, שעלול היה לדעת כיצד לפענח את ההודעות המוצפנות. בין הצדדים היה צורך לתאם מראש את המועדים לקבלת מפתח חדש שהוחלף מדי תקופה, וכאשר מקבלי המפתחות היו רבים מדי (למשל כל האוגדות של הצבא), היתה קיימת בעיה לוגיסטית כבדה.

בשנת 1975, לאחר מספר שנים שבהם מדענים אשר עסקו בהצפנה חיפשו דרך ליצור הצפנה חדשה המתגברת על הבעיה של החלפת מפתחות ראשונית בין שולח ההודעה לזה שמקבל אותה, עלה ויטפילד דיפי (Diffie), מדען אמריקאי על רעיון תיאורטי מבריק: הצפנה אסימטרית (לא סימטרית).

הצפנה אסימטרית היא רעיון שונה מרעיונות ההצפנה שהיו מקובלים עד כה בעולם ההצפנה, בהם תמיד המפתח היה סימטרי. מפתח ההצפנה ומפתח הפענוח הם תמיד שווים. תהליך הפענוח הוא תמיד תהליך ההפוך של ההצפנה. תהליך ההצפנה ותהליך הפענוח תמיד השתמשו באותו מפתח שנקבע מראש.

הרעיון היה ששולח ההודעה משתמש במפתח כלשהו כדי להצפין את ההודעה, ואילו מקבל ההודעה משתמש במפתח אחר על מנת לפענח את ההודעה.

הרעיון של ויטפילד דיפי היה רעיון תיאורטי בלבד לאותה שנה. הרעיון המרכזי היה מתן אפשרות לכל אדם א להצפין הודעה כלשהי המיועדת לאדם ב בעזרת המנעול של אדם ב, כך שמקבל ההודעה, אדם ב, יידע הוא ורק הוא לפענח את ההודעה המוצפנת. המנעול של אדם ב, מקבל ההודעה, יהיה נגיש לציבור וכל אדם המבקש לשלוח לו מסר, יצפין את המסר בעזרת אותו מנעול. כאשר ההודעה תגיע לאדם ב, הוא יוכל לפענח אותה בעזרת מפתח שנמצא אצלו ורק אצלו.

משל למה הדבר דומה:
 אדם א מבקש לשלוח חבילה לאדם ב. את החבילה, שם אדם א בתוך תיבת עץ שאותה הוא
 נועל בעזרת מנעול קפיצי ששייך לאדם ב, ואדם ב הפיץ עותקים מהמנעולים הללו שלו לכל
 הציבור (מפתח ציבורי).

אדם א יודע לסגור את המנעול הקפיצי בעזרת קליק בודד, אבל הוא ואחרים אינם יכולים
 לפתוח אותו בחזרה כי אין להם את המפתח (המפתח הפרטי). כלומר, מי שירצה לפענח את
 המסר המוצפן, לא יוכל כי אין לו את המפתח הפותח את המנעול. גם לא זה שנעל את
 המנעול (שולח ההודעה).

אדם ג שירצה לנסות ולהסתכל על תכולת התיבה הנעולה, חייב לפרוץ את המנעול.

גדולתו של רעיון המפתח הציבורי והמפתח הפרטי היה בכך שכעת, בניגוד לכל הרעיון של
 ההצפנה עד לאותה תקופה, אין צורך להצפין את מפתח ההצפנה, מכיוון שלפענוח יש מפתח
 אחר, פרטי ולא ידוע לציבור, מלבד למקבל ההודעה.

ההיסטוריה של הצפנת ה RSA

הבעיה העיקרית ברעיון של דיפי היא שלא היתה ידועה עד 1977 פונקציה מתמטית שיכלה
 לתמוך וליישם את הרעיון של המפתח הציבורי והמפתח הפרטי. עד כה, כל ההצפנות
 הסימטריות עבדו והסתמכו על דרך מתמטית כלשהי להצפנה ולפענוח המסרים.

בשנת 1976, צוות של שלושה מדענים, מהמכון הטכנולוגי של מסצ'וסטס (MIT), כמו מדעני
 מחשב רבים בעולם, שמע על הרעיון של דיפי, שאותו דאג דיפי להפיץ בטייטה שפרסם
 לראשונה בקיץ 1975.

רון ריווסט, ליאונרד אדלמן, אמריקאים, ועדי שמיר, ישראלי, שלושה מדעני מחשב
 ומתמטיקאים, התלהבו מהרעיון והחליטו גם הם כמו רבים, לחפש את הפונקציה המתמטית
 הנכספת שתאפשר הצפנתו של מספר כלשהו ללא אפשרות פענוחו ע"י אותו מפתח הצפנה
 אלא ע"י מפתח פענוח שונה.

בחודש אפריל 1977, לאחר כשנה של חיפוש מתיש אחרי הפונקציה המתמטית, בערב חג
 הפסח של אותה שנה, עלו שלושה מדעני מחשב, על דרך מתמטית להצפין מספר כלשהו
 בצורה כזו שמפתח הפענוח יהיה שונה ממפתח ההצפנה.

באותו הלילה, עלה ריווסט על הפונקציה המבוקשת, והחל לכתוב מאמר מדעי המפרט את
 דרך הביצוע של ההצפנה והפענוח. באותו בוקר, מסר את המאמר לאדלמן על מנת שזה
 יבדוק האם לא נפלה שגיאה כלשהי. אדלמן לא מצא שגיאה כזו. ההצפנה האסימטרית
 המעשית הראשונה היתה כתובה כמאמר מדעי שלם.

ריווסט הכניס למאמר את שמו ואת שמות שני עמיתיו אדלמן ושמיר. כך נולדה למעשה שמה
 של ההצפנה, הצפנת RSA, ראשי תיבות של שלושת המדענים שמצאו את הפונקציה
 המתמטית המאפשרת הצפנה אסימטרית. האות R מייצג את רון ריווסט, S מייצגת את עדי
 שמיר ו A מייצגת את ליאונרד אדלמן.

הפונקציות המתמטיות העומדות בבסיסה של ההצפנה הן פונקציות של חשבון מודולרי.
 איציק מסביר שכרגע אין זה רלוונטי לפרט אותן כאן כדי להבין את הרעיון שהוא יסביר
 בהמשך.

הרעיון המרכזי של הצפנת RSA

הרעיון המרכזי בהצפנת RSA הוא שההצפנה מתבססת על כך שהמפתחות מבוססים על שני מספרים ראשוניים, מספרים המתחלקים רק בעצמם ובמספר אחד, המוכפלים אחד בשני ויוצרים את מפתח הצפנה (המפתח הציבורי). נניח שמספרים ראשוניים אילו יהיו p ו q . מכפלתם תיתן את המספר N שזכרנו לחלק מהמפתח הציבורי, ובעזרתו ניתן להצפין כל מסר שרוצים לשלוח לבעל אותו מפתח N .

מפתח הפענוח (המפתח הפרטי) גם הוא מתבסס על אותם שני מספרים ראשוניים p ו q . קיימת פונקציה נוספת המחשבת את מפתח הפענוח, אבל מכיוון שרק מפיץ המפתח יודע מה הם אותם שני מספרים ראשוניים, רק הוא יכול ליצור את מפתח הפענוח (המפתח הפרטי), מכיוון שכדי לחשב את שני הגורמים (המספרים) הראשוניים p ו q , המרכיבים את המפתח הציבורי N , יש צורך בזמן חישוב רב מאוד, אם המספרים הראשוניים שנבחרו הם גדולים מאוד.

כאן בעצם טמונה הבעיה המתמטית ויפיה של הצפנת RSA: כדי לחשב את שני הגורמים הראשוניים p ו q שמכפלתם יוצרת את המספר N , יש לבצע את תהליך בדיקה שיכול לקחת הרבה מאוד זמן. תהליך הבדיקה מורכב משני שלבים: מתחילים במספר האי זוגי הראשון הראשון 3 ובודקים האם הוא מחלק את המפתח N ללא שארית. המספר הבא שנבדק הוא מספר הגדול ב 2 מהמספר הקודם (מספרים ראשוניים גדולים מ 3 הם תמיד אי זוגיים ולכן מוסיפים 2 ולא 1), וגם עליו בודקים האם הוא מחלק את N ללא שארית.

מבצעים את הלולאה הזו עד למספר שקטן משורש של N (כי שורש של N כפול שורש של N ייתן לנו את N , ואין צורך לבדוק מספרים גדולים משורש של N).

מכיוון שאנו יודעים ש N הוא מכפלת שני מספרים ראשוניים, אין צורך לבדוק האם המספר שמצאנו הוא ראשוני, מכיוון ש N אינו יכול להיות מכפלה של שני מספרים אחרים, מכיוון שמכפלת מספרים ראשוניים מרכיבה אותו.

הבעיה נוצרת כאשר N הוא ממש, אבל ממש גדול. נניח בגודל של מאה בחזקה מאה. כדי לבדוק גורמים ראשוניים שמכפלתם יוצרת מספר כזה, יש לבדוק את כל המספרים מ 3 עד שורש N , כלומר עד מאה בחזקת חמישים.

כדי לייעל את האלגוריתם של הבדיקה, אפשר לנסות לדלג על כל המספרים בלולאה שאינם ראשוניים. אבל כדי לדעת מתמטית האם מספר הוא ראשוני או לא, יש צורך באלגוריתם הבדוק האם מספר כזה הוא ראשוני. יש לבדוק את כל המספרים היכולים לחלק אותו.

בשנת 2002 פרסמו קבוצת מתמטיקאים הודים כי הצליחו למצוא אלגוריתם פשוט לגילוי מספרים ראשוניים. הפרסום עשה גלים מכיוון שנטען בצורה לא נכונה כי אם אכן נמצא אלגוריתם כזה, הוא מפשט את הבדיקה של מספרים ראשוניים, ומקטין את הזמן ביצוע של האלגוריתם להפקת הגורמים הראשוניים p ו q שמכפלתם יוצאת את N , המפתח הציבורי.

כתבה: מתמטיקאים הודים מצאו אלגוריתם למציאת מספרים ראשוניים
<http://computers.walla.co.il/ts.cgi?tsscript=item&path=4&id=270246>

כתבה: הפרכת הטענה של המתמטיקאים ההודים כי גרמו לפיצוח ה RSA
<http://computers.walla.co.il/ts.cgi?tsscript=item&path=4&id=273287>

איציק חושב בצורה פרקטית

איציק הוא מתמטיקאי מתוסכל. למרות שאהב את המקצוע, הוא לא המשיך לעסוק במתמטיקה טהורה. איציק סיים את לימודיו האקדמיים בפקולטה למדעי מחשב באחת האוניברסיטאות בישראל.

כמו רבים מחבריו, גם הוא מצא את עצמו בהייטק הישראלי עובד כמהנדס תוכנה באחת החברות. הרקע המתמטי היה לו לעזר רב, אבל בהייטק כמו בהייטק, עושים גם דברים פרקטיים עוקפי מתמטיקה.

כך למשל, בבניית מערכת תוכנה שמצריכה חישובים מהירים מאוד, והמעבד (מיקרופרוססור) בתוך אותה מערכת לא חזק דיו כדי לבצע חישוב מהיר, נעשתה פניה לטבלה בזיכרון שהכילה תוצאות של חישובים מוכנים. הגישה לזיכרון היתה מהירה הרבה יותר מאוסף פעולות חישוב שביצע המעבד, וזו נוצלה על מנת לשפר את העבודה.

לדוגמה, מודול התקשורת במערכת מקבל נתונים דו ספרתיים בצורה הפוכה בקו התקשורת ועליו היה להמיר את הנתונים על מנת שהתוכנה תבין אותם. למשל, מודול התקשורת מקבל מספר דו ספרתי בצורה הפוכה, נניח 57. המספר האמיתי שהמערכת צריכה לקבל הוא המספר 75, שספרותיו הפוכות מהמספר 57.

תכנת רגיל עם רקע מתמטי בסיסי היה כותב פונקציה שלוקחת את המספר 57, מפרקת אותו לספרותיו, מרכיבה מספר חדש בעזרת חישוב מתמטי ושולחת את התוצאה 75 ביציאה מן הפונקציה.

כלומר, אם המספר הדו ספרתי N והוא במבנה כמו XY, צריך להמיר אותו למספר YX על ידי חישוב מתמטי. את התוצאה נשים במשתנה F.

כך זה יראה:

$$\begin{aligned} X &= N \text{ div } 10 \\ Y &= N \text{ mod } 10 \\ F &= 10 * Y + X \end{aligned}$$

בחישוב בוצעו הפעולות המתמטיות הבאות:

פעולת div היא פעולה של חלוקת שלמים. למשל $34 \text{ div } 10$ תיתן את התוצאה 3.
פעולת mod היא פעולת של שארית חלוקה: למשל $34 \text{ mod } 10$ תיתן את התוצאה 4.
פעולת * היא פעולת כפל בין שני מספרים וקודמת לפעולת חיבור או חיסור.
פעולת + היא פעולת חיבור בין שני מספרים.
פעולת = היא פעולת הצבה של תוצאה למשתנה ששומר את התוצאה.

עכשיו, נניח כי למעבד שלנו לוקח שניה אחת לבצע פעולה מתמטית. בדוגמה לעיל שכוללת ארבע פעולות חשבון ועוד שלוש פעולות הצבה, כלומר שבע פעולות לביצוע, ייקח למעבד לקבל תוצאה תוך שבע שניות.

ועכשיו נניח שבאותה מערכת פעולת גישה לזיכרון לוקחת גם כן שניה אחת. כיצד ניתן לשפר את זמן ההמרה של המספר הדו ספרתי שהתקבל מ 7 שניות לשניה אחת?

ובכן, בהנחה שאין בעיה של מחסור בזיכרון, והבעיה הקריטית כאן היא רק זמן החישוב, נשים בזיכרון רשימה בת 100 שורות. בכל שורה ברשימה נרשום מספר. למשל בשורה 00 נרשום את המספר 00. בשורה 01 נרשום את המספר 10. בשורה מספר 34 נרשום את המספר 43. בשורה מספר 57 נרשום את המספר 75. בשורה 99 האחרונה, כמובן ירשם המספר 99.

כלומר, בכל שורה נרשום את המספר ההפוך למספר השורה. רשימת המספרים תהיה קבועה ולא תשתנה לעולם.

כאשר נקבל בתקשורת מספר כלשהו, הוא יסמן עבורנו את מספר השורה בזיכרון שצריך לגשת אליה ולשלוף את המספר שכתוב שם.

אם למשל המספר שהתקבל הוא 34, ניגש לשורה 34 בזיכרון ונשלוף את המספר הרשום שם: המספר 43. זמן הגישה שווה לשניה אחת. כך למעשה ביצענו "חישוב" שלקח לנו שניה אחת ולא שבע שניות.

למעשה, עקפנו כאן את המתמטיקה ויצרנו כביש עוקף מתמטיקה. דרך עוקפת חישובים מתמטיים. השיטה המתוארת לעיל נקראת גם lookup table והיא פרקטיקה ידועה בתכנות במערכות משובצות מחשב (Embedded Systems)..

כביש עוקף מתמטיקה לשבירת צופן ה RSA

הבעיה העיקרית כיום בפריצת צופן ה RSA הוא בכך שכדי לבצע חישוב מתמטי הבודק מה הם גורמיו הראשוניים p ו q של מפתחי ציבורי N , יש לבצע כל כך הרבה חישובים, שזמן החישוב המצטבר שלהם יכול לקחת מספר שנים.

אין כיום אלגוריתם מתמטי יעיל המסוגל לבדוק מה הם גורמיו הראשוניים של מספר הוא תוצאה של שני מספרים ראשוניים גדולים מאוד. יש לבדוק עבור כל מספר ראשוני p הקטן משורש N , האם הוא מחלק בשלמות ללא שארית את המספר N .

אבל, זו כמובן חשיבה מתמטית ולא חשיבה פרקטית.

איציק טוען שכאשר יוצרים את המספר N , משתמשים בטבלה של מספרים ראשוניים גדולים ידועים, או מחשבים אותם בעזרת אלגוריתם כלשהו. כך למעשה יש בפועל רשימה של כל המספרים הראשוניים בעולם, מ 3 עד p כלשהו.

כדי לבדוק האם מספר הוא ראשוני בזמן יעיל, אין צורך להפעיל אלגוריתם מתמטי, אלא לחפש את המספר ברשימה שחושבה מראש של כל המספרים הראשוניים הידועים. אם המספר נמצא ברשימה, הרי הוא ראשוני. אם הוא לא ברשימה, אז הוא לא ראשוני.

כדי לחפש מספר כלשהו ברשימת מספרים ממוינת, אפשר להפעיל חיפוש בינארי, כמו שמפעילים כשמחפשים מספר טלפון של אדם הרשום בספר טלפונים. חיפוש יעיל ברשימה ממוינת הוא חיפוש שמתחיל מאמצע הרשימה. אם המספר שהעלנו (מהאמצע) הוא מספר גדול מהמספר שאנחנו מחפשים, אז נחפש בחצי העליון. אם המספר שהעלנו הוא מספר קטן יותר מהמספר שאנחנו מחפשים, נחפש את המספר בחצי התחתון. חצי הרשימה תהיה הרשימה החדשה לחיפוש.

נחפש שוב את המספר החל מהחצי של הרשימה החדשה. שוב על פי הכללים בפסקאות הקודמות. נמשיך בכך עד שנגיע ונקבל רשימה שמספר איבריה הוא 1.

בחיפוש בינארי משתמשים בחיפוש בבסיסי נתונים ממוינים, ברשימות ממוינות ובכל טבלה ממוינת.

חיפוש בינארי הוא כל כך יעיל, עד שחיפוש של פריט (או מספר) בטבלה המכילה 1000 פריטים, יתבצע בסך הכול בפחות מ 10 צעדים. חיפוש בטבלה של מליון פריטים, יתבצע בפחות מ 20 צעדים. חיפוש בטבלה בגודל של 2 בחזקת N יתבצע ב N צעדים בלבד. החיפוש יתבצע בזמן לוגריתמי $O(\log(n))$ יעיל ולא בזמן ליניארי $O(n)$ לא יעיל.

בהצפנת RSA הבעיה העיקרית היא למצוא את הגורמים הראשוניים p ו q שכפולתם נותנת את המספר N המשמש כמפתח ציבורי שבעזרתו כל אדם א יכול להצפין מסר לאדם ב בעזרת מפתח ציבורי N שהפיץ לעולם אדם ב.

כיום, רק אדם ב יכול לפענח את המסר אם N הוא מספר מספיק גדול, ואף אדם אחר, כולל אדם א שהצפין את המסר אינו יכול.

אבל, אם היתה קיימת טבלה של כל מספרי ה N האפשריים שהן כפולות של כל המספרים הראשוניים אחד בשני, לא היה צורך בניסיון להפעיל אלגוריתם מתמטי כדי למצוא את המספרים הראשוניים p ו q המרכיבים את N.

כל שצריך הוא לייצר טבלה כזו. איציק קורא לה "לוח הכפל של המספרים הראשוניים".

פעם, כשהצפינו מסרים שלא בעזרת הצפנת RSA, המפתח לא היה ציבורי, אלא המפתח היה נסתר. כיום, המפתח של הצפנת RSA הוא מפתח ציבורי ונגיש לכל אחד, גם למצפין המבקש לשלוח מסר, וגם לאדם שמנסה לשבור את המסר המוצפן. וכאן החולשה של הצפנת המפתח הציבורי.

איציק מסביר שהבעיה כיום היא שאין מחשב חזק דיו כדי לבצע חישובים בפרק זמן סביר, אבל זיכרון יש בשפע, וכיום הוא זול מאוד בהשוואה לשנת 1977, השנה בה המציאו את הצפנת ה RSA. איציק יודע גם להסביר שאם מפתח ההצפנה N הוא למשל בגודל 1024 ביט (שמתאים למספר עשרוני בין יותר מ 300 ספרות), אז כמות הזיכרון שצריך כדי לאחסן את המפתח הוא בסך הכל 128 בתים (מחלקים 1024 ביטים ב 8 כי בכל בית יש 8 ביטים). בנוסף יש צורך ב 128 בתים נוספים על מנת להחזיק את המספר הראשוני p וכן 128 בתים נוספים כדי להחזיק את המספר הראשוני q , שניהם מרכיבים את מפתח ההצפנה N.

איציק, מתמטיקאי מתוסכל ותכנת מובטל טוען שכולם חושבים בצורה מתמטית ולא בצורה פרקטית כמו מתכנתים, ולכן הצפנת RSA עדיין לא נשברה.

לדעתו של איציק, כשם שילדים בבית הספר היסודי משננים בעל פה את לוח הכפל והחרוצים שבהם יודעים עבור כל מספר בלוח הכפל מי הם גורמיו שהוכפלו אחד בשני (ללא ביצוע פעולה מתמטית כלשהי), כך גם מחשבים יכולים למצוא את הגורמים הראשוניים של המספר N, על ידי פנייה לטבלה מוכנה השוכנת בזיכרון המחשב.

השלכות אפשריות

תיאור האפשרות לפצח מסרים מוצפנים ב RSA על ידי חיפוש המפתח הציבורי ב"לוח הכפל של המספרים הראשוניים" יכול לגרום למצב שארגוני ביון של מדינות גדולות ייצרו לוח כפל ענק של כל מכפלות המספרים הראשוניים בעולם, וכל שיצטרכו על מנת לגלות את המפתח

הפרטי מתוך המפתח הציבורי הוא לבדוק היכן נמצא המפתח הציבורי בלוח הכפל של המספרים הראשוניים.

אם ישנם מליון מספרים ראשוניים ידועים, אז ישנם מליון בחזקת 2 מפתחות ציבוריים אפשריים. מספר הצעדים המקסימלי הנדרשים למצוא את הגורמים הראשוניים הוא פחות מ 40 צעדים (חישוב לוגריתם בבסיס 2 של מליון בחזקת 2).

$$\text{Log}_2(1,000,000^2) = 2 * \log_2(1,000,000) < 2 * \log_2(2^{20}) = 2 * 20 = 40$$

יתרה מזאת: אם ישנם 10 בחזקת 100 מספרים ראשוניים ידועים, אז ישנם 10 בחזקת 200 מפתחות ציבוריים אפשריים. מספר הצעדים המקסימלי הנדרשים למצוא את הגורמים הראשוניים הוא פחות מ 800 צעדים.

$$\text{Log}_2(10^{200}) = 200 * \log_2(10) < 200 * \log_2(2^4) = 200 * 4 = 800$$

על פי המיקום של המספר N בלוח הכפל של המספרים הראשוניים, נוכל לחפש ולמצוא ביעילות את הגורמים הראשוניים q ו p שאותם אין צורך לחפש בעזרת אלגוריתם מתמטי לא יעיל, ובעזרתם לפענח את המסר המוצפן ע"י יצירת מפתח הפענוח (המפתח הפרטי).

* הכותב הוא בעל תואר ראשון במדעי המחשב של האוניברסיטה הפתוחה