הצפנה בתוכנה חופשית

עבודה סמינריונית בקורס תקשורת מחשבים ובטיחות מידע (35-747)

ליאור קפלן kaplanlior@gmail.com

תוכן עניינים

הצפנה בתוכנה חופשית....1
1.מבוא ועקרונות....2
1.1 תוכנה חופשית....2
1.2 הצפנה....2
1.3 הצפנה סימטרית....3
1.4 הצפנה א-סימטרית – הצפנת המפתח הציבורי....4
1.5 הצפנה א-סימטרית – ההבדל בין הצפנה לחתימה דיגיטלית....5
2.מימושים בתוכנה חופשית....6
2.1 OpenSSH....6
2.2 GNU Privacy Guard....8
2.3 OpenSSL / GnuTLS / Network Security Services....15
2.4 DM-Crypt ו-TryeCrypt....16
3.סיכום....17
4.מקורות....17
5.רישיון....18

1. מבוא ועקרונות

1.1 תוכנה חופשית

תוכנה חופשית[1] היא תוכנה המוגדרת על פי רישיון השימוש שלה. רישיון השימוש של תוכנה חופשית הוא מתירני מאוד בהשוואה לרישיון השימוש הסטנדרטי ("כל הזכויות שמורות"). מטרת הרישיון, ומה שהופך את התוכנה לחופשית היא הדאגה לזכויות המשתמשים ובעיקר לזכויות להפיץ את התוכנה ולשנות אותה. כל אלה כמובן בתנאי שניתן קרדיט הולם לבעל זכויות היוצרים.

אופי הפיתוח של תוכנה חופשית, באמצעות מגוון גדול של אנשים מכל העולם, רובם ללא הכרות פנים מול פנים, הציבה דרישות גבוהות לנושא האימות והאבטחה. כך שניתן למצוא מימושים רבים לאלגוריתמים שונים ואף המצאה של אלגוריתמים חדשים בעקבות צרכים חדשים.

הרישיון המתירני של תוכנה חופשית גם מאפשר לכל פרוייקט חדש להתבסס על פרוייקטים קיימים ללא צורך להמציא את הגלגל מחדש כל פעם, וכך פרוייקטים מסויימים נהפכים לאבני בניין עבור פרוייקטים אחרים. השילוב של השניים, שימוש באבני בניין ודרישה לאימות ואבטחה הובילה לסינון מספר המימושים הרב לכדי מספר אבני בניין כאלה בתחום ההצפנה. כיום, ניתן למצוא מימוש פתוח לתקני הצפנה רבים באמצעות תוכנה חופשית. בחלק מהמקרים, אבני הבניין משמשות גם את הקהילה המסחרית.

1.2 הצפנה

העיסוק בהעברת מידע יצרה שני תחומים דומים שלעיתים מתבלבלים ביניהם אך לכל אחד מהם מטרה שונה. סטגנוגרפיה[2] היא התחום שמתעסק בהסתרת קיומו של המסר מבלי לשנות את המסר עצמו. כלומר, בהינתן מידע על קיום המסר והשיטה בה הוא הוסתר, קל לקרוא ו/או להבין אותו. לעומתה, קריפטוגרפיה[3] מתעסקת בהסתרת תוכנו של המסר, בעוד עצם קיומו ידוע לכל. במקרים רבים אפשר להתייחס לסטגנוגרפיה כאמנות בעוד הקריפטוגרפיה הינה מדע (כיום מדובר בענף של מתמטיקה ומדעי המחשב).

לאורך ההיסטוריה, השתנתה מהותית צורת ההצפנה, החל מצופן אתב"ש בספר ירמיהו[4], דרך צופן קיסר[5] באימפריה הרומית ועד לשיטות הצפנה המכניות (מכונת האניגמה לדוגמה) והאלקטרוניות (מחשבים וציוד תקשורת של ימינו). עם השיפורים בתקשורת, והעליה בשימוש באמצעי התקשורת החדשים נדרשו דרכים מהירות יותר להצפין שדרים ומסרים. הצפנה באמצעות מכונות דרשה שכלול של שיטות ההצפנה, והמעבר להצפנה אלקטרונית דרשה שכלולים רבים עוד יותר. על עובדות אלה יעידו ההתקדמות המשמעותית של נושא ההצפנה מאז מלחמת העולם השניה ועד היום.

במקביל לפיתוח צפנים חדשים, גם התפתח תחום פיצוחם, והדבר בא לידי ביטוי בתקופת מלחמת העולם השניה כאשר הבריטים פיצחו את הצפנת האניגמה (הצפנה מכנית) באמצעות גילוי כשלים במבנה מכונת ההצפנה וטעויות של מפעילי המכונה.

ישנם מספר תחומים בהם נושא התוכנה החופשית ונושא ההצפנה דומים ואף משיקים. כיום, בשניהם השיטה העדיפה לפיתוח והתקדמות היא שקיפות וביקורת מתמדת. כתוצאה מכך גם קצב ההתקדמות שלהם הוא מהיר.

> "בקריפטוגרפיה מודרנית, אלגוריתמים מוצלחים הם כאלו שזכו לביקורת הציבור, במיוחד לעיון מדוקדק של פרטיהם בידי אנליסטים ומומחי הצפנה מסביב לעולם. אלגוריתם AES הוא דוגמה טובה לאלגוריתם שנבחר בקפידה מבין מספר מועמדים טובים על ידי מומחים מרחבי העולם, לאחר תחרות שאורגנה על ידי NIST ונמשכה כמספר שנים.
>
> אלגוריתם הצפנה טוב צריך להישען אך ורק על סודיות המפתח. כך שאם ייחשף המפתח, אין צורך להחליף את האלגוריתם כולו אלא רק להחליף מפתח. באנלוגיה למנעול צירופים, כאשר בעל המנעול חושד כי עוד מישהו מלבדו מכיר את הצירוף הנכון לפתיחת המנעול, כל שעליו לעשות הוא להחליף צירוף, אין צורך לרכוש מנעול חדש."

---

1 http://www.gnu.org/philosophy/free-sw.he.html
2 http://he.wikipedia.org/wiki/סטגנוגרפיה
3 http://he.wikipedia.org/wiki/קריפטוגרפיה
4 http://avrahamtobias.org/writings/2009/על-כתב-סתרים-צופן-וחידות-בספרותנו-העתי/
5 http://he.wikipedia.org/wiki/צופן_קיסר

(קריפטוגרפיה. (2009, אוגוסט 13). *ויקיפדיה, האנציקלופדיה החופשית*.)

1.3 הצפנה סימטרית

עד שנת 1976 כל שיטות ההצפנה היו סימטריות, כלומר שיטת ההצפנה ושיטת הפיענוח הן זהות ומבוצעות בכיוון ההפוך. גם המפתח שלהן זהה, ובמילים אחרות, מי שיכול להצפין הודעה בעזרת מפתח מסויים, יכול גם לפענח הודעה שהוצפנה עם המפתח הזה.

הבעיה העיקרית בהצפנה סימטרית היא הדרך בה שני צדדים מסכימים על מפתח ההצפנה. התשובה הפשוטה היא מפגש פיזי, אך הדבר לא תמיד אפשרי, בעיקר שמדובר בתקשורת בין גורמים מרוחקים. בקצרה, מתוארת הבעיה כפרדוקס שכדי ששני צדדים יחלקו סוד (הודעה מוצפנת) עליהם כבר לחלוק סוד (מפתח הצפנה). כלומר אם צד אחד שמחזיק מנעול ומפתח, צריך למצוא דרך להעביר את שניהם לצד השני כדי שניתן יהיה לבצע את העברת המידע. כך שהבעיה העיקרית של הצפנה סימטרית היא בעיית תיאום וחלוקה של מפתחות הצפנה.

ישנה חידת ילדים מוכרת, ששואלת כיצד ניתן להעביר חבילה בדואר, כאשר ידוע כי עובדי הדואר פותחים כל חבילה ללא מנעול. כאשר ברור מאליו כי לא ניתן לשלוח את המנעול בדואר. הפתרון הוא שצד א' שולח את החבילה עם מנעול עליה, צד ב' מוסיף את המנעול שלו ושולח חזרה. צד א' מסיר את המנעול ושולח חזרה. לבסוף, צד ב' מקבל חבילה שיש עליה רק את המנעול שלו ויכול לפתוח אותה ולגלות את תוכנה.

בעוד הרעיון נשמע פשוט ביותר בחידות ילדים, בפועל הוא לא אפשרי מאחר ופונקציות הפיענוח חייבות להעשות בסדר ההפוך לפונקציות ההצפנה. ההצפנה האחרונה צריכה להיפתח ראשונה. ובדוגמה של ההמנעולים סדר זה לא נשמר - המנעול הראשון נפתח ראשון במקום אחרון.

בשנת 1976 פורסם מאמר של דיפי והלמן[6] המתאר כיצד להתגבר על הבעיה של חלוקת סוד משותף על פני טווח לא מוצפן. שיטה זאת נקראת פרוטוקול דיפי-הלמן[7]. הפרוטוקול מתחיל בהסכמה על שני מספרים כלשהם עבור הצבה בנוסחה מתמטית. בנוסף, כל צד בוחר מספר פרטי משלו, מציב בנוסחה ביחד עם המספרים שנבחרו במשותף ושולח את התוצאה לצד השני. כעת חוזרים הצדדים על התהליך, רק הפעם מציבים את התשובה שהתקבלה מהצד השני במקום את המספר המקורי. באופן מפתיע – התוצאה זהה אצל שני הצדדים. גורם הרואה את החלפת המספרים אינו יכול לשחזר את התהליך מאחר וחסר לו המספר הפרטי של כל אחד מהצדדים.

אני אוותר על המתמטיקה של העניין ועל תיאור הפונקציות עצמן, ואתן דוגמה פשוטה (מתוך הספר סודות ההצפנה) שתבהיר את הרעיון מאוחרי הפונקציות היחודיות האלה. שני הצדדים בוחרים צבע כלשהו במשותף וכל אחד בוחר צבע אישי משלו. כל צד מוסיף ליטר מהצבע האישי שלו לליטר של הצבע המשותף ושולח את הדלי עם הצבע לצד השני.

גורם מן הצד רואה החלפה של שני דליים בצבעים שונים, אך אינו יכול לדעת מה הם הצבעים המקוריים (במיוחד כאשר נתייחס גם לגווני הצבעים ולא רק לצבע עצמו). כעת, כל צד מוסיף ליטר שלו לדלי שהתקבל מהצד השני. ושוב, בשני הצדדים מתקבלת תצורה זהה כתוצאה מעברבוב של הצבע המשותף והצבע הפרטי של שני הצדדים וזאת ללא יכולת של הגורם המתבונן לשחזר את התהליך.

חסרונה העיקרי של שיטה זאת, הוא הצורך בעבודה בו זמנית או המתנה של זמן ארוך לצורך התהליך. אם אני רוצה להעביר למישהו הודעה מוצפנת, עלי קודם כל ליצור איתו קשר לטובת פרוטוקול דיפי-הלמן ורק אז אוכל להעביר לו הודעה מוצפנת.

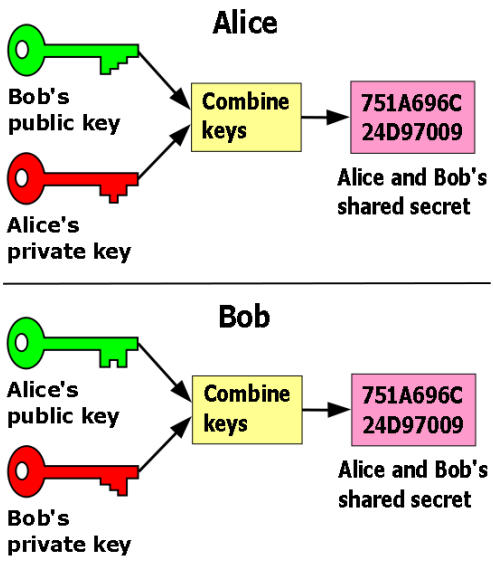
1.4 הצפנה א-סימטרית – הצפנת המפתח הציבורי

בנוסף לפרוטוקול שלהם, מאמרם של דיפי והלמן הכיל גם תיאור של רעיון הצפנת המפתח הציבורי, ותיאור השימוש בו עבור חתימה דיגיטלית. הרעיון המפכני הוא שיטת הצפנה בה דרך הפיענוח שונה מדרך ההצפנה. שוני זה מתבטא באי היכולת של כל צד לבצע את הפעולה ההפוכה ומכאן האי-סימטריות של התהליך בניגוד לשיטה הסימטרית שתוארה לפני כן.

6 http://crypto.csail.mit.edu/classes/6.857/papers/diffie-hellman.pdf
7 http://he.wikipedia.org/wiki/פרוטוקול_דיפי-הלמן

אחד היתרונות של שיטה זאת היא העובדה כי ישנו מפתח פומבי המפורסם ברבים, כלומר כבר אין צורך לבצע את פרוטוקול דיפי-הלמן על מנת לתאם בין שני הצדדים. כל מה שצריך לעשות כדי לשלוח הודעה היא להשתמש במפתח הציבורי של נמען ההודעה.

חוזק ההצפנה נובע מהעובדה כי למרות שמפתח ההצפנה שונה ממפתח הפיענוח, הם צמד יחודי (יחס חד חד ערכי), כך שעל מנת לפענח הודעה שהוצפנה בעזרת מפתח מסויים, יש להשתמש בבן הזוג הספציפי שלו. חלק חשוב נוסף בפתרון הוא שלא ניתן להסיק (באמצעים סבירים) את הקשר בין הצמד. בהניתן מפתח הצפנה לא ניתן לדעת אם מפתח פיענוח כלשהו הוא הבן זוג שלו, ללא בדיקה בפועל. תכונה זאת, מקשה על פעולות של פענוח ההצפנה באמצעות כוח גס[8] (Brute Force).



שימוש במפתח ציבורי ופרטי כרכיבים לפרוטוקול דיפי-הלמן ליצירת סוד משותף.

ב-1977, כשנה לאחר הפרסום של דיפי והלמן, הומצא RSA[9] – מימוש ראשון לרעיון המפתח הציבורי. המימוש הוא מציאת פונקציה העונה על הדרישה להצפנה א-סימטרית וניסוח אלגוריתם המהווה מימוש מלא של רעיון המפתח הציבורי של דיפי והלמן.

עם זאת, יש להזכיר כי למרות יתרונותיה הרבים של הצפנה א-סימטרית יש לה חיסרון עיקרי אחד והוא מהירות ההצפנה. הצפנה א-סימטרית איטית בצורה משמעותית מהצפנה סימטרית וקשה יותר למימוש בצורה פשוטה. הדבר מהווה מגבלה לגבי שימוש בהצפנה באופן יום יומי ובאמצעות משאבים ביתיים. כמו כן, מורכבות האלגוריתם מקשה על מימושו באמצעות חומרה.

כדי להתגבר על בעיה זאת, השימוש העיקרי היום של הצפנה א-סימטרית היא כדי לשתף מפתח עבור הצפנה סימטרית. כלומר, לבצע בהתחלה הצפנה האיטית כדי לאפשר הצפנה מהירה. ניתן להשתמש במפתחות שכבר נוצרו כפרמטרים לביצוע פרוטוקול דיפי-הלמן וכך לחשב את הסוד המשותף בלא צורך בתקשורת אינטראקטיבית.

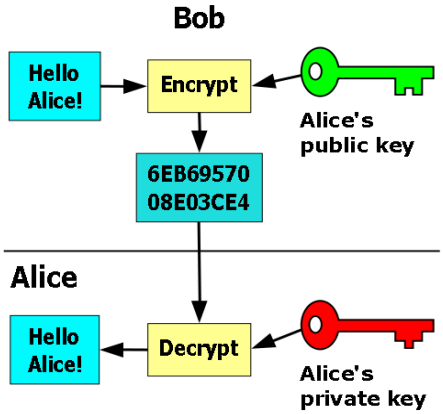
1.5 הצפנה א-סימטרית – ההבדל בין הצפנה לחתימה דיגיטלית

רעיון המפתח הציבורי מכיל שני שימושים עיקריים:

- הצפנה
- אימות וחתימה דיגיטלית

בשניהם ישנו שימוש בעקרון המפתח הציבורי אך בשיטות שונות ולמטרות שונות. כדי להבהיר את הנושא כדאי לחזור לבסיס. לכל אדם יש מפתח פרטי ומפתח ציבורי. הראשון הוא סודי (וזהו חלק מהותי בהצפנה) בעוד שהשני פומבי ומפורסם ברבים.

הצפנה - נזכור כי בצמד המפתחות הציבורי והפרטי כל מפתח עושה את העבודה ההפוכה של הצד השני. ולכן אם אשתמש במפתח הציבורי שלי כדי להצפין, רק המפתח הפרטי שלי יוכל לפענח את ההצפנה. בשיטה זאת, ניתן להצפין מידע באופן בו רק אני אוכל לקרוא אותו. אך שימו לב כי גם אני וגם כל גורם אחר יכול להשתמש במפתח הציבורי שלי. כלומר שאין הבדל בין הצפנה שאני עושה עבור עצמי לבין הצפנה שגורם אחר עושה עבורי.



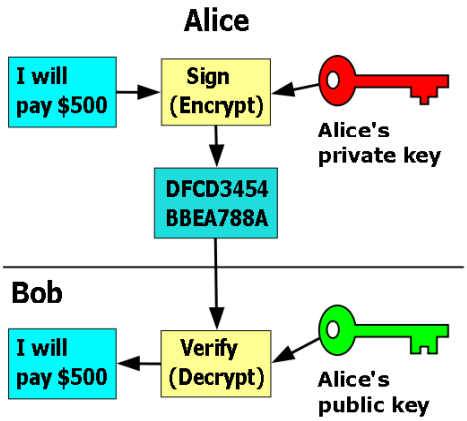
הצפנה באמצעות מפתח ציבורי

8 http:
9 http:

אימות וחתימה דיגטלית – כאן מדובר על התהליך ההפוך להצפנה. המילה הפוך נועד כדי להבהיר כי השימוש בהצפנה הוא במטרה שכל מי שרוצה יפענח, ולכן ההצפנה הינה אמצעי לאימות ולא כלי לצורך סודיות.

הרעיון הוא שימוש במפתח הפרטי על מנת להצפין דברים. בשיטה זאת, רק שימוש במפתח הציבורי שלי יפענח את המידע. מאחר והמפתח הציבורי שלי זמין לכולם, כל אחד יכול לבצע את הפענוח בעזרתו. אך בצורה זאת, מאחר ונדרש המפתח הציבורי שלי, ניתן לאמת כי אני הוא מקור ההודעה.

במקרים רבים, נהוג לבצע שימוש בשני מפתחות במשלוח הודעה מוצפנת, הראשון הוא המפתח הציבורי של הנמען על מנת להצפין את ההודעה כך שרק הוא יכול לפתוח אותה. המפתח השני הוא המפתח הפרטי של השולח וזאת על מנת לאמת את מקור הודעה.



חתימה דיגיטלית באמצעות מפתח פרטי

חתימה דיגיטלית בצורה זאת היא בעייתית עבור נפחי מידע גדולים, מאחר וזמן ההצפנה והפענוח הם ארוכים. דוגמה פשוט היא הרצון של יוצר סרט לחתום דיגיטלית שזהו אכן סרטו – האם עליו להצפין גיגות של מידע לשם כך?

הפתרון הוא תהליך שנקרא הצפנה עם נספח, בו מתבצעת החתימה על קובץ נפרד מהקובץ המכיל את המידע. הנספח מהווה תמצית של הקובץ המקורי בצורה בה ניתן לאמת שזאת אכן התמצית. כך שבפועל מבוצעים שני אימותים: שהחתימה על הנספח היא אכן מבעל החתימה ושהנספח אכן מתמצת את המסמך המקורי.

תמצית המסמך מבוצעת ע"י פונקציית גיבוב[10] (hash), המקבלת את המסמך כקלט ומוציא פלט ייחודי. ניתן להתייחס לתמצית גם כטביעת אצבע של המסמך מאחר והסיכוי לשני מסמכים בעלי אותה טביעת אצבע הוא נמוך מאוד.

10 http://he.wikipedia.org/wiki/פונקציית_גיבוב

2. מימושים בתוכנה חופשית

2.1 OpenSSH

על מנת להבין את החשיבות של SSH (קיצור של Secured Shell) יש להבין מה היה המצב לפני השימוש בו ואת הצורך שעורר את הפיתוח שלו. עד להתחלת השימוש ב-SSH, דרך הכניסה העיקרית לשרתים היתה באמצעות פרוטוקולי rlogin ו-telnet. שני פרוטוקלים אלה אינם מוצפנים, וניתן להאזין להם בקלות ע"י ציתות לתקשורת מצד השרת או מצד הלקוח.

SSH תוכנן ע"י חוקר באוניברסיטת הלסינקי לטכנולוגיה שבפינלנד. מטרת הפיתוח היתה למנוע התקפה על רשת האוניברסיטה במטרה להאזין לסיסמאות שעוברות על גבי הרשת באופן לא מוצפן.

התוכנה חולקה בהתחלה בחינם (freeware), והשימוש בה גדל במהרה. המימוש כלל גם רכיבים של תוכנה חופשית אך לאחר כשנה התוכנה נהפכה למסחרית וקניינית. במקביל, פותחה גרסה חדשה (גרסה 2) לפרוטוקול כדי להתמודד עם בעיות שונות שלו, ובעיקר מספר כשלים באבטחה. בנוסף, הוכנס שימוש בפרוטוקול דיפי-הלמן על מנת לסכם על מפתח הצפנה חדש.

בשנת 1999, כשלוש שנים לאחר מכן החליטו מספרי מפתחי תוכנה חופשית כי הם רוצים מימוש חופשי של SSH. לצורך גם הם חזרו לגרסה ישנה של SSH ששוחררה כתוכנה חופשית והתחילו ממנה את הפיתוח מחדש.

פרוייקט OpenBSD, שמתמקד בפיתוח מערכת הפעלה בטוחה על בסיס מערכת הפעלה BSD (מערכת הפעלה פתוחה נוספת מעבר ללינוקס), לקח חסות על הפיתוח ויצר את OpenSSH. הפיתוח מבוצע עבור פרוייקט OpenBSD, ובמקביל נוצרת גרסה שמיועדת עבור מערכות אחרות כגון לינוקס ו-UNIXים שונים. החל מ-2005, OpenSSH נחשב למימוש הנפוץ ביותר של פרוטוקול SSH, שהחל מ-2006 נהפך לתקן רשמי הקשור באינטרנט[11]. רישיונו הפתוח של המימוש מאפשר התבססות עליו במקרים רבים ועל ידי גופים רבים וכך נהפך להשלמה לתקן ע"י מימוש ותקן דה פקטו.

חשיבותו של SSH אינה רק ביכולת הבסיסית שלו לאפשר חיבור מאובטח לשרת, אלא בכך שהגרסה השניה שלו תוכננה כתשתית לחיבור בטוח ומאפשרת לפרוטוקולים אחרים לרכב על SSH לצורך תקשורת מאובטחת או להציע Tunnel (מנהרה או קשר מנקודה לנקודה) מאובטח. כתוצאה מתכנון זה, ניתן להשתמש בחיבור SSH כדי להוסיף אבטחה לפעולות שונות כגון העברת קבצים (scp), העברת תקשורת ומידע (tunnel), וגישה לשרתים מרוחקים (sshfs).

בזמן ההתקנה של sshd, תוכנת השרת עבור SSH (נקראת גם SSH Daemon), נוצרים זוג מפתחות (ציבורי ופרטי). המפתח הציבורי מתועד לטובת אנשים המתחברים לשרת ורוצים לוודא שזהו אכן השרת. דוגמה ניתן לראות ברשימת המכונות[12] של פרוייקט דביאן. בדף של המכונה alioth.debian.org[13] ניתן לראות תיעוד של המפתח הציבורי של השרת ושל טביעת האצבע של המפתח (בדיקות תקינות למפתח הציבורי).

בצד הלקוח קיימים מספר מנגנונים שנועדו לספק הגנה למשתמש מפני בעיות בזיהוי מול השרת. המנגנון הראשון מיועד לבצע אימות של השרת אליו רוצים להתחבר. השיטה לביצוע אימות זה היא הצגת טביעת האצבע (fingerprint) של המפתח הציבורי של השרת. בהתחברות ראשונה לשרת alioth.debian.org מתקבלת ההודעה הבאה:

```
$ ssh alioth.debian.org

The authenticity of host 'alioth.debian.org (217.196.43.134)' can't be established.

RSA key fingerprint is 99:11:ed:30:03:41:ff:9f:f3:74:bd:7d:e1:8f:04:44.

Are you sure you want to continue connecting (yes/no)?
```

זהו אישור ידני לזהות השרת ולנכונות המפתח שלו, שניתן להשוואה מול דף השרת ברשימה של דביאן. לאחר האישור, מבוצע תיעוד של המפתח בקובץ הנקרא known_hosts.

11 http://tools.ietf.org/html/rfc4252
12 http://db.debian.org/machines.cgi
13 http://db.debian.org/machines.cgi?host=alioth

Warning: Permanently added 'alioth.debian.org,217.196.43.134' (RSA) to the list of known hosts.

בקובץ עצמו, נוספת שורה עבור השרת המכילה את שם השרת, סוג מפתח ההצפנה והמפתח עצמו.

alioth.debian.org ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAxuVlBnTWE9+g5w/uxuk7SmNLEmXPucZz8iE8kE02
zaBxPFdlEKJUhUkkf11qkHp9eWVRMro75IRtOJjVLQNmlKjIw+IncqGvj7bvHcAuqYAwNOhuSt
Pnk/W0jwcs52TkNv7MZprRJOrprJGDMSBhovhBNXYYD8kruhQXJRLV9wBWp9p8VrokBbxl/e
KXVuvJfyZU20JmKbyLUPdB9vfQQr9o3btwM//A61WL8sFnnu7JfetbFNGmnO+AwIew/QLs/8B
Orwk1RwrcuKcs1ULMTgmUK8/QCpM3I9BhLYl/ypxpADiJFSbTRqqzg5xU/UkNQ3NEmXL2G2
A2UWLEuUd22Q==

מנגנון נוסף מאפשר תיעוד של המפתח בהתבסס על שם השרת, כתובת ה-IP שלו או הצירוף של שניהם. מנגנון זה נועד כדי למנוע בעיות התחזות שונות שניתן לעשות כתוצאה מחטיפת כתובת IP או חטיפת כתובת DNS.

Warning: the RSA host key for 'alioth.debian.org' differs from the key for the IP address '217.196.43.134'

Offending key for IP in /home/kaplan/.ssh/known_hosts:4

Matching host key in /home/kaplan/.ssh/known_hosts:3

Are you sure you want to continue connecting (yes/no)?

אך יש לזכור כי מדובר גם בשינויים לגיטימיים שנעשים מידי פעם, ולכן יש לבדוק בחשדנות כל מקרה של חוסר התאמה, אך לא צריך להיבהל כשזה קורה. ניתן גם לערבל את פרטי השרת ברשימה כדי להקשות על דליית פרטים מהקובץ ע"י צד שלישי או כדי להקשות על שיבוש רשומות בקובץ.

עד כאן בוצע אימות לשרת בלבד – נוצר איתו הקשר, נבדק המפתח הציבורי שלו ומאחורי הקלעים גם מבוצע תיאום של ההצפנה לצורך תקשורת מאובטחת. חלק זה נקרא שכבת ה-Transport של פרוטוקול SSH. בסוף הפעולה של שכבה זאת, המשתמש יכול לתקשר באופן מאובטח עם השרת, אך השרת לא יודע מי המשתמש שהתחבר אליו. לצורך כך קיימת שכבת ה-user authentication שאחראית על זיהוי אימות המשתמש.

לצורך האימות ישנן מספר שיטות, הפשוטה ביותר היא שיטת הסיסמה (password). בשיטה זאת שרת ה-SSH מנהל משתמשים וסיסמאות באופן עצמאי. שיטה זאת אינה בשימוש, מאחר ונדרשת השקעה נוספת לצורך ניהול המשתמשים והסיסמאות.

השיטה השניה היא שיטת המפתח הציבורי, בשיטה זאת השרת מזהה את המשתמש באמצעות הצפנה של מידע כלשהו באמצעות המפתח הפרטי של המשתמש ופיענוח ע"י המפתח הציבורי שלו. לצורך כך, נדרש כי בשרת יהיה עותק מראש של המפתח הציבורי והגדרה לאיזה חשבון משתמש (user account) בשרת הוא יישויך.

שיטה זאת ניתנת ליישום רק לאחר הכנות מראש של זוג מפתחות למשתמש והעברת המפתח הציבורי שלו לשרת עבור חשבון ספציפי. יש לשים לב כי מדובר בזוג מפתחות שונה מזוג המפתחות ששרת ה-SSH יוצר בהתקנה שלו, וכל זוג משתמש למטרה שונה - אימות השרת לעומת אימות המשתמש. לצורך אימות השרת אין צורך או תלות בקיומם של זוג מפתחות עבור המשתמש.

מספר יתרונות של שיטה זאת, הן האפשרות להתחבר ללא צורך בהקשת סיסמה ומאחר והאימות מבוצע אך ורק על ידי המפתחות (מתאים לתהליכים שצריכים לרוץ אוטומטית). אך אליה וקוץ בה, השימוש ללא סיסמה הופך את המפתח הפרטי לחמיד ביותר מאחר וניתן להעתיק אותו ולעשות בו שימוש ע"י גורם נוסף.

כדי להתגבר על הבעיה, בזמן יצירת המפתח, ישנה אפשרות ליצירת סיסמה עבור השימוש במפתח הפרטי. כלומר כדי להצפין משהו יש להכניס סיסמה ורק אז ניתן לבצע את ההצפנה. פרוייקטים רבים מאפשרים כניסה לשרתים רק באמצעות מפתח SSH שהועבר מראש ולא באמצעות סיסמה כלשהי על מנת להוריד חלק מהסיכונים הקיימות באימות משתמשים (כולל נסיונות פריצה).

צורה זאת של אימות עדיין סובלת מקושי בניהול המפתחות הציבוריים וכל פרוייקט נדרש לבצע איסוף באופן עצמאי. סיבה נוספת המקשה על הניהול הוא חוסר היכולת לשייך מפתח מסויים לאדם כלשהו רק על סמך המפתח. בשונה למפתחות GPG שניתן (ונהוג) להצמידם לכתובת דואר אלקטרוני (פרטים בהמשך).

השיטה השלישית היא השיטה האינטראקטיבית (*keyboard-interactive*), בשיטה זאת שולח השרת כל מיני בקשות קלט מתוכנת הלקוח שהמשתמש אמור להקליד. בפועל, שיטה זאת משמשת לצורך אימות שם משתמש וסיסמה מול מערכת ההפעלה.

ההבדל בין שימוש בשיטה בזאת לבין השיטה הראשונה של הסיסמה בלבד היא נושא התחזוקה. כאן מבוצע שימוש במנגנון האימות של מערכת ההפעלה, וברשימות המשתמשים והסיסמאות שלה. בשיטה הראשונה יש צורך לבצע ניהול עצמאי של הרשימות וכפילות של הרשימות ביחס למערכת ההפעלה.

השיטה הרביעית נקראת GSSAPI והיא מיועדת כדי לאפשר אימות מול מנגנונים חיצוניים לשרת כגון Kerberos[14]. מנגנונים אלה מתאפיינים בכך שישנה רשות שמאשרת את תקפות המפתחות המשמשים לזיהוי, אימות והצפנה. שיטה זאת מיושמת בעיקר בארגונים גדולים שמחזיקים תשתית שרתים גדולה היכולה לתת שירותי אימות כאלה.

לאחר ששכבת אימות המשתמש מסיימת את תפקידה, נכנסת לשימוש שיטת החיבור (Connection) שתפקידה לנהל את העברת המידע בין הלקוח לשרת ולהפך. שיטה זאת עובדת על ידי הגדרת מספר ערוצי תקשורת, בקשות הקשורות לערוצים ובקשות כלליות. כל ערוץ מבצע העברה של נתונים בין השרת ללקוח או להפך בעוד במקביל מבוצעות בקשות שונות הקשורות להעברת המידע או לפרמטרים כלליים (לדוגמה, החלפת מפתח ההצפנה כעבור זמן מסויים או כמות תעבורה מסויימת).

לסיכום יש לזכור כי SSH הוא דרך התקשורת הטובה ביותר בין שרתים ובין משתמשים לשרתים כאשר נדרשות יכולות הצפנה וגמישות. ניתן להרכיב עליו מספר רב של פרוטוקולים על מנת לאפשר להם לעבוד באופן מאובטח (תקשורת לא מוצפנת על טווח מוצפן). תוכנות ה-SSH מגיעות עם כל מערכת הפעלה המבוססת על לינוקס או על UNIX לסוגיו כך שמדובר בתקן דה פקטו בתחום התקשורת.

2.2 GNU Privacy Guard

בעקבות פרוטוקול דיפי-הלמן והמצאת RSA בשנות ה-70 תחום ההצפנה ביצע קפיצת מדרגה, אך הוא עדיין היה בשימוש של גופים גדולים היכולים לממן את המשאבים הדרושים לצורך רכישת מחשבים היכולים לבצע את החישובים הרבים הקשורים להצפנה.

המחשב האישי בגרסאותיו המקודמות היה זמין החל מתחילת שנות ה-80, אך בימים שלפני האינטרנט התקשורת בין מחשבים היתה באמצעות חיוג במודם ל-BBS, מעיין שרתים שאירחו לוחות אלקטרוניים שאפשרו בין השאר גם העברת קבצים.

עם התפתחות התקשרות בין המחשבים, וגישה גוברת לדואר אלקטורני באמצעות האינטרנט בגרסאותיו המוקדמות, נושא הפרטיות התחיל להיות בעייתי יותר ויותר. בעקבות הצעת חוק בארה"ב שדרשה מכל יצרן ציוד תקשורת מצופנת להשאיר דלת אחורית עבור הממשלה, פיל צימרמן שחרר ב-1991 את PGP, התוכנה הראשונה שהביאה את ההצפנה למשתמש הביתי. הוא הצליח לחבר לתוכנה אחת את השימוש בפרוטוקול דיפי הלמן, הצפנת מפתח ציבורי והצפנה סימטרית לשם מהירות ואף יכולות חתימה מבוססות על אלגוריתם אל-גמאל[15].

מטרת התוכנה היתה לאפשר לאנשים להעביר מידע ולאחסן קבצים באופן מוצפן על גבי BBSים. התוכנה מצאה את דרכה בסופו של דבר לאינטרנט והשימוש בה גדל בצורה משמעותית. הגרסה המקורית הופצה באופן חינמי (freeware), ולאחר מספר שנים נפתחה חברה שסיפקה את המוצר בגרסאות מסחריות. לאחר מספר גלגולים ורכישות, GPG המקורית הפכה למוצר McAfee E-Business Server[16].

14 http://he.wikipedia.org/wiki/קרברוס_(פרוטוקול)
15 http://he.wikipedia.org/wiki/חתימה_דיגיטלית_אל-גמאל
16 http://www.mcafee.com/us/enterprise/products/data_protection/data_encryption/ebusiness_server.html

כתוצאה מתפוצתה ואיכותה של התוכנה, היא נהפכה לתקן דה פקטו בנושא הצפנה ומפתחות ציבוריים. במטרה לייצב ולסדר את עולם תוכנות ההצפנה נוצר בשנת 1997 תקן בשם OpenPGP. על בסיס תקן זה התחיל בשנת 1999 פיתוח של GNU Privacy Guard (או בקיצור GPG). כיום GPG הוא הבסיס העיקרי להצפנת דואר אלקטרוני וקבצים בודדים במערכות הפעלה פתוחות עם מימושים גם עבור Windows[17]. בנוסף למימוש עצמו, נכתבה (בנפרד) תוכנת תשתית עבור ניהול והפצת המפתחות[18]. כיום ידועים ומופצים באופן ציבורי כ-2.7 מיליון[19] מפתחות הצפנה.

בעוד שנושא ההצפנה קשור בד"כ לנושא הפרטיות, אך שימוש בו אינו מתפרסם יתר על המידה. למעט אנשים יש אינטרס לפרסם ברבים כי הם מבצעים הצפנה של החומר שלהם מסיבות שונות. לעומת זאת, נושא החתימה הדיגיטלית זוכה להדים רבים. פעולות רבות הקשורות לתוכנה חופשית משתמש בחתימה דיגיטלית באמצעות מפתחות PGP או מפתחות GPG כפי שמתייחסים אליהם בעולם התוכנה החופשית. מפתחות אלה נמצאים בשימוש רב בכל מצב בו נדרש לאמת את זהות יוצר הקבצים ולוודא שהתוכן שלהם לא השתנה.

הפיתוח של PGP התחיל מרצון להצפין תעבורת דואר אלקטרוני וקבצים. היום, כשדואר אלקטרוני היא תשתית העברת קבצים לכל דבר, הצפנתו היא בפועל הצפנת קבצים, אך עם דרך ליצור קשר עם מקור הקובץ (שולח המייל) גם ללא צורך בפענוח ההודעה. מאחר ורק תוכן המייל מוצפן ולא הכותרים (headers) שלו.

בעניין זה יש להזכיר כי דואר אלקטרוני הוא פרוטוקול בעייתי במיוחד מהיבטי אבטחה ממספר סיבות: הראשונה היא שלא ניתן לאמת את מקור ההודעה. כל אחד, יכול לציין כל כתובות שולח בזמן שליחת ההודעה. השניה היא שלצורך הניתוב של הדואר יש תלות בהרבה מאוד גורמים ובכל שלב אחד מהם יכול להתערב במספר דרכים על מנת שלא להעביר את ההודעה, לשנות את יעדה, את המקור שלה או את תוכנה. בהיבט זה, עצם העובדה שדואר אלקטרוני מגיע ליעדו היא סוג של נס.

בהשוואה גסה, זה כמו להעביר שטר של 200 ש"ח מסוף אוטובוס עמוס דרך העברה מיד ליד דרך כל הנוסעים. אם מישהו יקח את הכסף, האדם בסוף האוטובוס לא ידע וגם לא הנהג. מאחר בעולם הדואר האלקטורני התקשורת היא חד כיוונית (אני שולח דואר אליך ולא יודע אם הוא הגיע) הדבר משול לתשלום נסיעה של 200 ש"ח. כלומר אם מישהו לקח את הכסף, בעל הכסף לא ידע שהכסף לא הגיע והנהג לא יראה כלום כי האוטובוס גם ככה עמוס.

חזרה לעולם התוכנה החופשית. בפועל, הדרך היחידה לאמת כתובת דואר אלקטרוני היא באמצעות שליחת דואר וקבלת תשובה. שיטה זאת אינה חסינה מהתקפות גורם מתווך (man in the middle), אך אין שיטה טובה יותר. הרעיון בדרך אימות זאת היא שבעוד שקל לזייף את מקור המייל, קשה לזייף את יעדו. זיוף היעד מוביל בדרך כלל לאי הגעת ההודעה. זאת בדיוק הסיבה שאתרי אינטרנט שולחים מייל עם תוכן כלשהו על מנת לאמת את כתובת הדואר של הגולשים.

בעולם התוכנה החופשית, כאשר חלק גדול מהתקשורת נעשה באמצעות דואר אלטקורני, שימוש בכתובת הדואר האלקטרוני כאמצעי מזהה היא ההגיונית ביותר. בסופו של דבר, זאת הדרך הכי וודאית ליצירת קשר. אם לא תפרסם א תהכתובת הנכונה, הדואר לא באמת יגיע אליך.

בעית האימות כאן היא בזמן קבלת דואר – איך אני יודע שהדואר שנשלח מהכתובת של אליס אכן הגיע מאליס. לעומת זאת, אני מניח שדואר שנשלח לאליס אכן יגיע אליה באמצעות תשתית הדואר האלקרוני שהיא בפועל אמינה <u>בהעברת</u> הודעות מצד לצד.

לשם כך, הנוהג הוא לשייך את מפתחות ההצפנה הנוצרים עם GPG לכתובת דואר אחת לפחות. בדרך זאת, באמצעות החתימה על דואר אלקטרוני באמצעות המפתח, ניתן לוודא כי כתובת המקור היא אכן נכונה מאחר והיא מופיעה במייל עצמו וגם מפתח איתו בוצעה החתימה האלקטרונית.

לאחר ההסבר כיצד משתמשים במפתחות GPG בתוכנה חופשית, אראה כיצד הם באים לידי שימוש בפועל ע"י מפתחים באופן מכוון וע"י משתמשים במנגנונים אוטומטים המובנים במקרים בהם נדרש אימות כלשהו.

חבילות תוכנה. הפצת לינוקס היא בתיאור גס מאוד גוף שעושה אינטרגציה בין תוכנות ממקורות שונים ע"י (בד"כ) מספר רב של אנשים. עבודת האינטגרציה מאפשרת להרכיב על גרעין מערכת ההפעלה את התוכנות הבסיסיות הנדרשות למערכת הפעלה ומעליהן ממשקים ושירותים רבים. לצורך עבודה זאת, ההפצה צריכה לוודא את מקור התכונות, את שלמותן ואת זהות האדם שאחראי להכללתן בהפצה.

17 http://www.gpg4win.org/
18 http://code.google.com/p/sks-keyserver/
19 http://sks-keyservers.net/status/

כל חבר בהפצת לינוקס שרוצה להכניס חבילת תוכנה כלשהי (חדשה או עדכון לחבילה קיימת) נדרש לחתום על קוד המקור של החבילה ועל שינויים שהוא עשה בה. לדוגמה, כאשר העלתי גרסה חדשה של החבילה php-doc לדביאן באוקטובר 2008[20] הייתי צריך לחתום בעזרת המפתח הפרטי שלי על קובץ המכיל את פרטי הקבצים שאני מעלה ואת טביעת האצבע שלהם על פי שלושה אלגוריתמים שונים (sha1, sha256 ו-md5).

בצורה זאת, במקום לחתום דיגיטלית על מספר קבצים, שחלקם יכולים להיות גדולים, מבוצע עליהם חישוב עבור טביעת אצבע ועליה מבוצעת החתימה. ההבדל הוא בכמות המידע שנחתם ובקלות הטיפול בו. הקובץ אינו מוצפן או חתום ישירות בשום צורה שהיא. טביעת האצבע נועדת כדי לוודא את שלמותו ותקינותו בעוד החתימה הדיגיטלית נועדת לוודא את שלמות טביעת האצבע ואת מקור הקבצים. בשיטה זאת, כמות המידע שעליה מתבצעת החתימה כי קטנה יחסית ומושפעת מכמות הקבצים ולא מגודלם.

כאשר ההפצה מבצעת שחרור של גרסה רשמית, נאסף אינדקס של כל החבילות[21] וכל חבילה מכילה גם תיאור של טביעת האצבע שלה. בתיאור יש גם הפניה לשם הקובץ במאגר של ההפצה.

Package: php-doc

Priority: optional

Section: doc

Installed-Size: 53356

Maintainer: Lior Kaplan <kaplan@debian.org>

Architecture: all

Version: 20081024-1

Recommends: php5-cli

Filename: pool/main/p/php-doc/php-doc_20081024-1_all.deb

Size: 5218428

MD5sum: c66a9a8bad1ab613c4823e852b5b465d

SHA1: fbf54ed0a3ca5a8d0f881ab6305482f6798f0d01

SHA256: 6ba946258866c647ed35c7f94e493c554fd2de53d5d0e30ce0b76a7654d6da2f

Description: Documentation for PHP5

מאחר ולמאגר החבילות של מספר חלקים, ישנם לא מעט קבצי אינדקס כאלה. כדי לפשט עניינים, מבוצע גם להם אינדקס[22] הכולל טביעת אצבע לכל קובץ. קובץ זה נחתם דיגטלית באמצעות מפתח שנוצר מראש ע"י ההפצה כחלק מתהליך שחרור. החתימה מצורפת כקובץ נפרד.

ניתן לאמת את החתימה ידנית, אך התהליך קורה באופן אוטומטי בכל פעם שמשתמשים בכלי ההפצה לצורך התקנת תוכנה מהמאגר של דביאן. כלומר בזמן ההתקנה מבוצע תהליך אימות באופן היררכי:

1. הורדת קובץ ה-Release והחתימה שלו.
2. בדיקה כי החתימה תקפה ביחס לקובץ ה-Release ולמפתח של הגרסה.
3. הורדת אינדקס של תיאורי החבילות.
4. בדיקת טביעת אצבע לאינדקס.
5. הורדת קבצי תיאור החבילות.
6. בדיקת טביעת האצבע לקבצי התיאורים.
7. הורדת החבילה המבוקשת.

---

20 http://lists.debian.org/debian-devel-changes/2008/10/msg01402.html

21 האינדקס המכווץ זמין בכתובת http://mirror.isoc.org.il/pub/debian/dists/Debian5.0.2/main/binary-i386/Packages.gz

22 http://mirror.isoc.org.il/pub/debian/dists/Debian5.0.2/Release

8. בדיקת טביעת אצבע של החבילה.

בדוגמה הבאה, ניתן לראות את הפלט של בדיקת אימות ידנית

```
$ gpg -d Release.gpg

Detached signature.

Please enter name of data file: Release

gpg: Signature made Sat 15 Aug 2009 05:41:08 PM IDT using RSA key ID 55BE302B

gpg: Good signature from "Debian Archive Automatic Signing Key (5.0/lenny)
<ftpmaster@debian.org>"

Primary key fingerprint: 150C 8614 919D 8446 E01E  83AF 9AA3 8DCD 55BE 302B

gpg: Signature made Sat 15 Aug 2009 05:45:22 PM IDT using DSA key ID F42584E6

gpg: Good signature from "Lenny Stable Release Key <debian-release@lists.debian.org>"

Primary key fingerprint: 7F5A 4445 4C72 4A65 CBCD  4FB1 4D27 0D06 F425 84E6
```

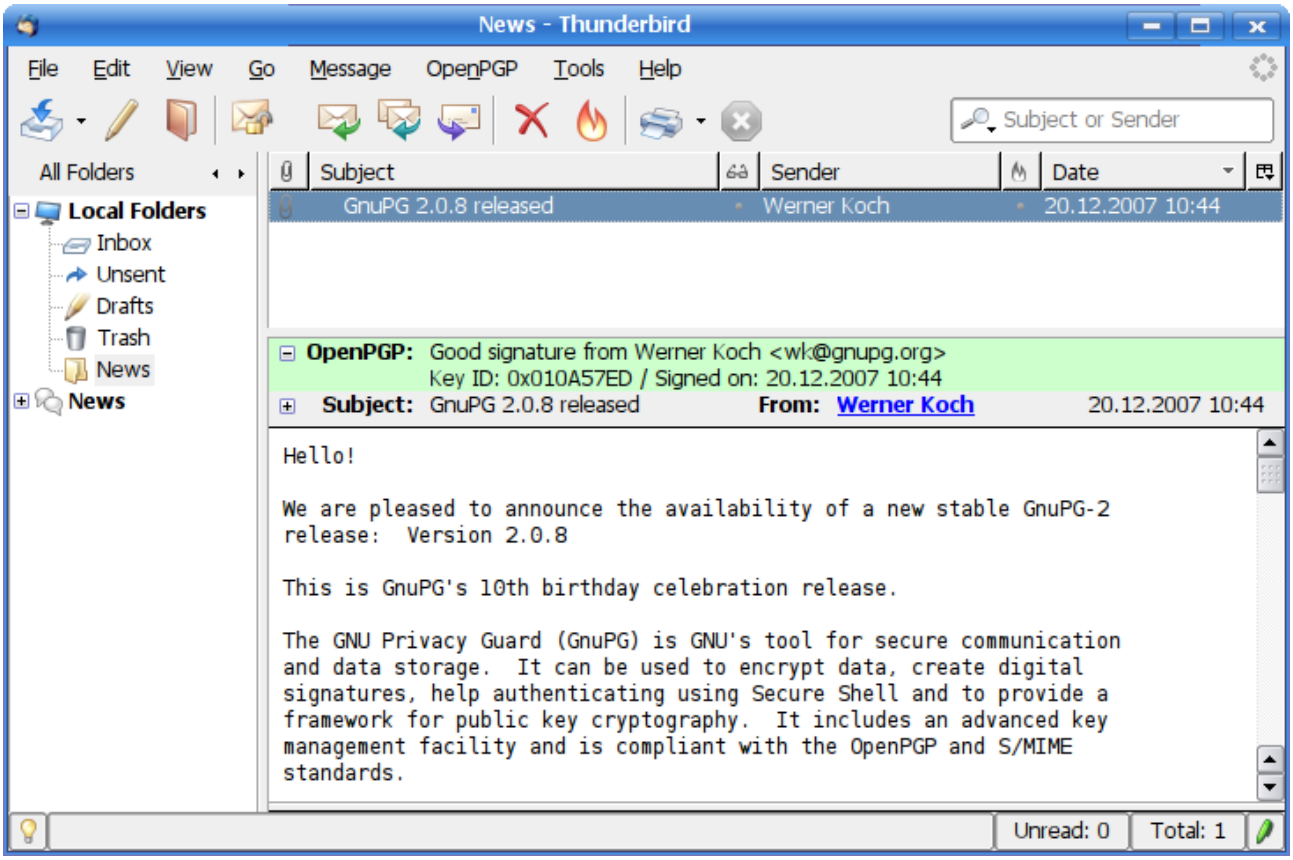
המפתחות, אשר המזהה שלהם מופיע בפלט אכן מופיעים באופן ציבורי[23], ולצורך הבדיקה גם ייבאתי אותם ממקור זה.

בהפצות המבוססות של Red Hat השיטה שונה, ואצלם מבוצעת חתימה על כל חבילת תוכנה בנפרד כך שניתן לאמת כל קובץ באופן עצמאי וללא קשר למאגר הרשמי. ניתן לראות[24] את רשימת המפתחות בשימוש צוות השחרור וצוות האבטחה של Red Hat. מפתחות אלה משמשים גם כדי לחתום על הודעות רשמיות לרשימות התפוצה של Red Hat.

בעוד פעולות רבות מבוצעות משורת הפקודה או מאחורי הקלעים, ישנן מספר תוכנות המאפשרות שילוב של יכולות ההצפנה והחתימה הדיגיטלית בתוכנות אחרות כדי לחשוף את היכולות הללו למשתמשים בצורה קלה וזמינה. לנושא הדואר האלקטרוני קיים תוסף לתוכנה Mozilla Thunderbird בשם Enigmail[25]. תוסף זה, מאפשר הצפנה ו/או חתימה דגיטלית עבור דואר יוצא בצורה פשוטה מתוכנת הדואר. Enigmail משוחרר ברישיון חופשי, ומתבסס על GPG לצורך הפעולות השונות.

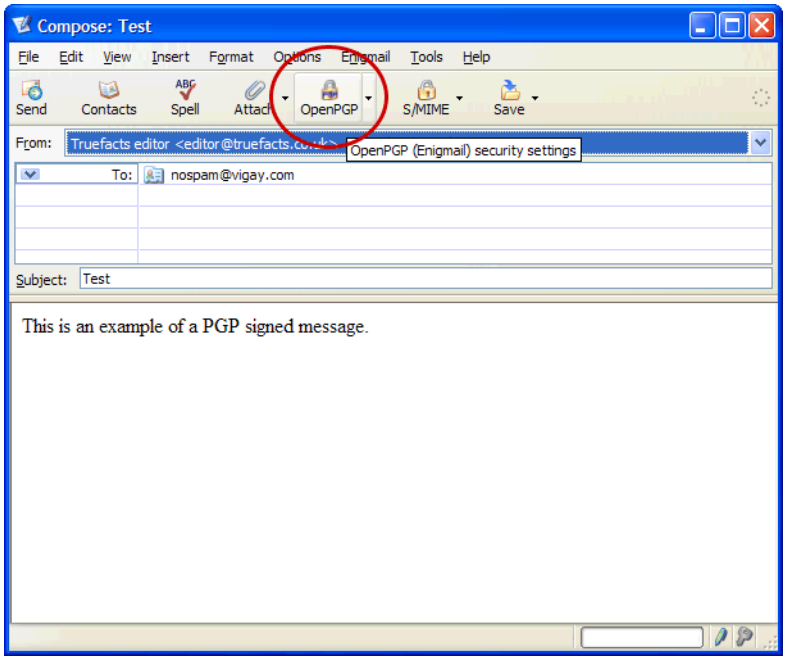
במקביל, עבור דואר נכנס, התוסף יודע לזהות אם מדובר בחתימה תקינה ומציג חיווי מתאים. במקרה ואין את המפתח המתאים במאגר מהפתחות המקומי, ניתן לייבא את המפתח משרת המפתחות לצורך אימות זהות השולח. כל הפעולות מבוצעות תוך התבססות על אוסף המפתחות הקיים בספריית ה-GPG של המשתמש.

23 http://pgp.mit.edu:11371/pks/lookup?op=vindex&search=0x9AA38DCD55BE302B
24 http://www.redhat.com/security/team/key/
25 http://enigmail.mozdev.org/home/index.php



דוגמה לדואר אלקטרוני חתום בתוכנה Mzoilla Thunerbird ביחד עם התוסף Enigmail

בחירת מפתח ההצפנה/חתימה, נעשה בד"כ על פי כתובת הדואר של השולח, כאשר נדרש כמובן נוכחות של המפתח הפרטי המתאים. עבור יכולות ההצפנה, נדרש לייבא את המפתח הציבורי של הנמען. ניתן לבצע חיפוש בשרתי המפתחות על פי כתובת דואר אלקטרוני או שם הנמען. לאחר ייבוא המפתח, ניתן להצפין את ההודעה ולשלוח אותה.



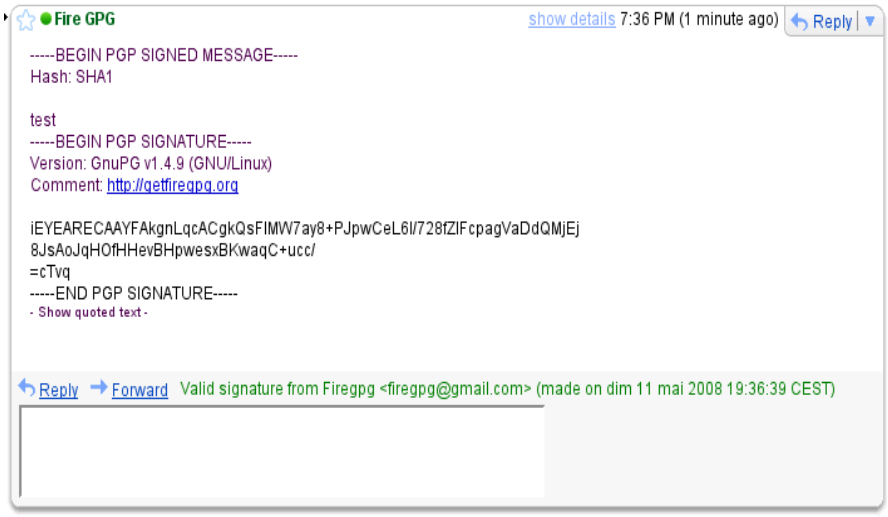
שליחת הודעת דואר אלקטרוני כאשר התוסף Enigmail מותקן

הממשק מספק בחירה פשוטה לגבי אפשרויות אלה, והן שני סימונים בפינה הימנית התחתונה של המסך, אחד עבור הצפנה (סימן מפתח) והשני עבור חתימה (סימן של עט).

בפועל, סימן המנעול אומר שימוש במפתח הציבורי של הנמען. סימן העת אומר שימוש במפתח הפרטי של השולח. שילוב של שני הסימנים אומר שימוש בשנ המפתחות גם יחד (סוד משותף).

כהשלמה ל-Enigmail, קיים תוסף עבור הדפדפן Mozilla Firefox בשם FireGPG המאפשר הצפנה של טקסטים בתיבות טקסט שונות ופענוח של טקסטים מוצפנים או חתומים תוך כדי עבודה של הדפדפן. הרעיון שמאחורי התוסף הוא לאפשר שימוש ביכולות של GPG בכל מקום בו ישנו טקסט שנשלח מהמשתמש לשרת כלשהו ובמקביל גם לאפשר פענוח קל של טקסט מוצפן או חתום.

FireGPG יודע גם להוסיף את יכולותיו מעל הממשק של Gmail, ובכך להפוך את נושא ההצפנה לשקוף עבור משתמשי Gmail. יש להדגיש שלא מדובר בתמיכה מיוחדת מצד האתר, אלא בשינויים שעושה התוסף לקוד שהדפדפן מציג בזמן שגולשים ל-Gmail.



ממשק Gmail כאשר התוסף FireGPG מותקן.

כמו במקרה של Enigmail, גם כאן מדובר בממשק מעל GPG וכדי להשתמש בו יש צורך להתקין את GPG מראש. בהפצות לינוקס דרישת קדם זאת ממולאת באופן אוטומטי במהלך ההתקנה של ההפצה. ב-Windows ניתן להתקין מספר הסבות של GPG לסביבה זאת כדי לספק את הפונקציונליות.

ציינתי מקודם כי בעולם התוכנה החופשית מוצמד כל מפתח הצפנה לכתובת דואר מסויימת. הצמדה זאת מאפשרת לקשר בין המפתח לכתובת הדואר והדבר חשוב לאימות מקור ההודעה, אך בעולם בו התקשורת נעשית בעיקר ע"י דואר אלקטרוני ולא ע"י מפגש פנים מול פנים, קשה מאוד לקשר בין כתובת דואר כלשהי לבין אדם ספציפי.

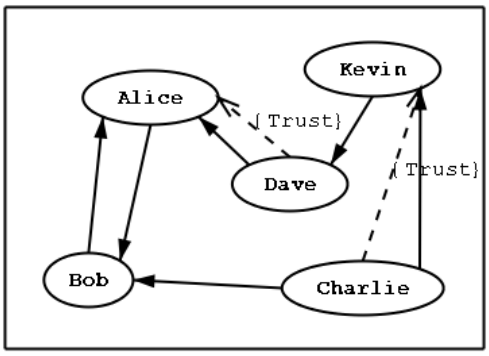
כאשר יצרתי את התיבה שלי ב-gmail זהותי האמיתית אינה נבדקת (וטוב שכך) בשום צורה. הדבר מהווה בעיה כאשר מישהו רוצה לסמוך על דברים המגיעים מכתובת הדואר הזאת. לפי מה שהצגתי עד כה, כל אדם יכול ליצור מפתח הצפנה, להכניס בזמן היצירה את הדואר שלו, ולשלוח ממנו (גם אם בצורה מזוייפת) הודעות חתומות.

כדי לספר יכולות אמון במפתחות הציבוריים שמוצגים בפומבי, יש שתי אפשרויות. הראשונה היא לתת אמון בגוף כלשהו שמאפשר את זהות האדם וגם יכול לאמת אותה על פי הצורך. המושג נקרא Certificate Authority או CA בקיצור. כך לדוגמה עובד נושא מפתחות ה-SSL של אתרי אינטרנט – קונים אותם ממספר גופים מצומצם שמספקים את שירות ה-CA. בחתימתם הם מאשרים את זהות האתר וכתובתו. לצורך כך, אנחנו, וליתר דיוק הדפדפן שלנו צריך לסמוך על אותו CA.

אך סמכותו של ה-CA אינה קיימת במצב בו אף אחד לא סומך עליו, ובפרט כאשר אף אחד לא סומך על מישהו שאינו עצמו. הפתרון במצב זה נקרא רשת אמון (web of trust). רשת אמון זאת בנויה על סמך העובדה שאני יכול לאפשר מספר אנשים שנפגשתי איתם פיזית ובדקתי את זהותם ואת כתובת הדואר שלהם. בכך יצרתי את הקשר בין האדם לבין כתובת הדואר שלו, והשלמתי את הקשר בין האדם למפתח הצפנה שלו.

כדי לוודא את הקשר בין האדם שהזדהה מולי לבין כתובת הדואר שמצויינת במפתח שלו, נהוג לשלוח את החתימה על המפתח לכתובת הדואר הזאת. וכך, אם הכתובת אינה נכונה, והוא לא מקבל את החתימה, הוא אינו יכול לעשות בה שימוש. אם הכתובת נכונה והחתימה הגיעה אליו, הוא יכול להעלות אותה לשרת המפתחות כדי שכולם יראו אותה.

בהנחה שאני סומך על מישהו, אני יכול לסמוך גם על אנשים שהוא סומך עליהם. וכך נבנית שרשרת אמון ובסופו של דבר רשת שלמה. השאלה הכמעט מיידית הינה מה קורה אם מישהו שאני סומך עליו עושה טעות או מאשר בצורה עיוורת מישהו אחר. כדי להתגבר על טעויות של בודדים, יש צורך במספר קשרים רב. כלומר כדי שאדם זר יסמוך עלי, הוא צריך מספר שרשראות אמון שמובילות ממנו אלי.



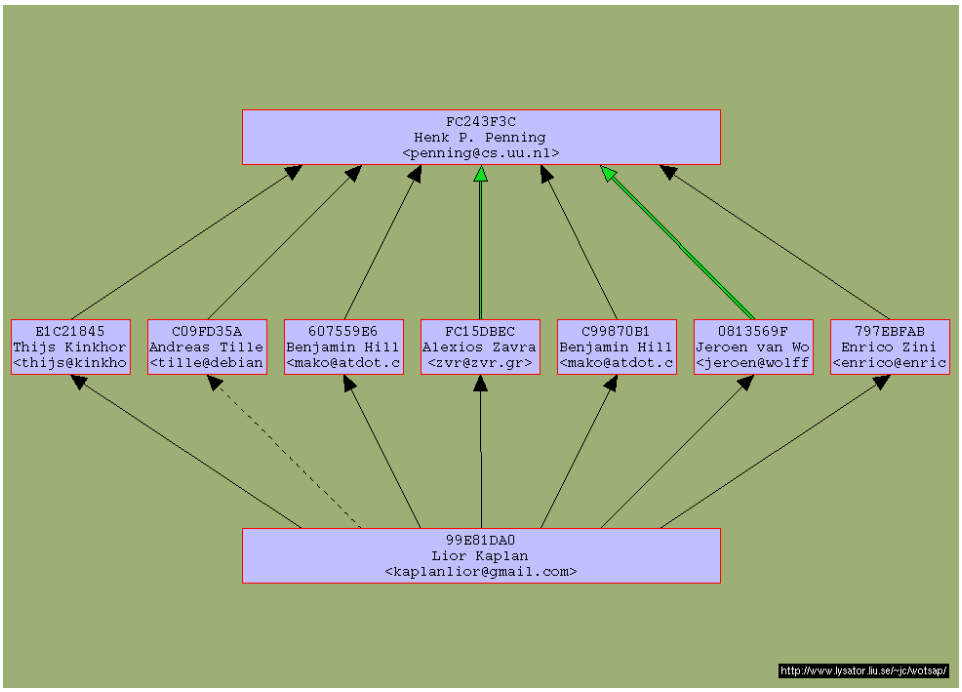
An example of the web of trust model

קשרי אמון ברשת אמון. השרטוט מתוך האתר

http://www.gnu.org/software/gnutls/openpgp.html

על המפתח הצפנה שלי[26], יש לי כ-300 חתימות. כלומר 300 איש שראו תעודה מזהה שלי לפני שאישרו את זהותי ואת השיוך שלי לכתובת הדואר האלקטרוני ומזהה המפתח הציבורי. כך שמישהו שמעולם לא פגשתי ורוצה לדעת אם ניתן לסמוך עלי, יכול לראות שיש 300 איש שכן סומכים עלי, ועכשיו הוא צריך לסמוך על כמה מתוך ה-300 האלה כדי לאשר את שרשרת האמון ביננו.

את שרשרת האמון אפשר למצוא בעזרת אתרים שמנתחים את רשת האמון[27]. לשם הדוגמה, החלטתי לבדוק את נתיבי האמון ביני לבין Henk P. Penning, יוצר תוכנת/אתר הניתוח. על פי תוצאות האתר[28], יש לי 8 נתיבים של אמון אל הנק. באותו אופן אפשר למצוא נתיבים בכיוון ההפוך, כך הנק יכול לדעת אם לסמוך על תוכן שאני סומך עליו. יש לזכור כי החתימות אינן תמיד דו כיווניות, אך רצוי שיהיו כאלה.



שרטוט של נתיבי אמון בכיוון אחד בין שני מפתחות

בהמשך לרעיון של הצפנת תעבורת דואר אלקטרוני, עלה רעיון לשנות לחלוטין את דרך האבטחה של בקשות HTTP. כיום הדרך השלטת היא SSL שיוצרת ערוץ מוצפן עליו מועברות הסיסמאת השונות. בשיטה זאת, האימות היחידי שמבוצע הוא של השרת אך לא של המשתמש. פרטים נוספים אודות שיטה זאת ניתן למצוא בהמשך העבודה.

בשיטה החדשה, הלקוח מצפין או חותם על בקשות ה-HTTP שלו וכך מזדהה מול השרת. כלומר בעצם הבקשה כבר ישנו זיהוי ולכן לא נדרש שום פרט נוסף. השרת יכול להשתמש במפתח הציבורי של המשתמש כדי לשלוח לו תשובה מוצפנת. מאחר והשרת גם יצפין וגם יחתום על הבקשה יכול הלקוח לדעת כי מקור הודעה היא מהשרת ומיועדת אליו. זאת היא רמת הזדהות גדולה יותר מזאת הקיימת היום עם SSL.

כיום קיימים שני חלקים לפתרון והם מודול בשם ModOpenpgp[29] ל-Apache, שרת ה-HTTP הנפוץ בעולם התוכנה החופשית, ותוספת בשם Enigform[30] לדפדפן Firefox שדואג להצפין את הבקשות.

למרות שמדובר ברעיון חדשני, המימוש הבסיסי כבר קיים ועובד אם כי מוגדר להיות בשלב בטא. מימוש הרעיון וקבלתו כסטנדרט תהווה מהפכה בתחום העבודה שלנו עם שרתים, אם זאת כל משתמש ידרש לנהל מפתחות הצפנה, דבר שאינו כזה פשוט עבור המשתמש ההדיוט, ויתכן כי עדיף שימשיך לעבוד עם סיסמאות.

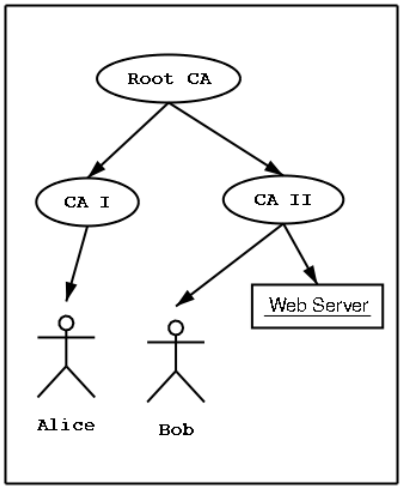
לסיכום, ניתן לראות כי GPG הינה תשתיות מאוד משמעותית בתחום ההצפנה של תוכנה חופשית. בעזרת תוכנות נוספות המתממשקות ל-GPG, הגישה של המשתמשים להצפנה עולה וכך גם קלות השימוש.

26 http://pgp.mit.edu:11371/pks/lookup?op=vindex&search=0x1558944599E81DA0
27 http://pgp.cs.uu.nl/
28 http://pgp.cs.uu.nl/paths/99E81DA0/to/FC243F3C.html
29 http://code.google.com/p/mod-openpgp/
30 https://addons.mozilla.org/en-US/firefox/addon/4531

2.3 OpenSSL / GnuTLS / Network Security Services

SSL (ראשי תיבות של Secure Sockets Layer) ו-TLS (ראשי תיבות של Transport Layer Security) הם שני דורות של פרוטוקולים שמיועדים להצפנה תקשורת. TLS הוא הדור הבא של SSL. יש בניהם הבדלים טכניים, אך מטרתם זהה ולכן אתייחס אליהם כ-SSL לשם הנוחות. SSL הוא רעיון שפותח בחברת Netscape, לימים Mozilla ארגון המפתח תוכנה חופשית.

בצורת תקשורת זאת, מבוצע אימות של השרת בלבד, על סמך חתימה דגיטלית על המפתח הציבורי של השרת. החתימה ניתנת על ידי גורם מאשר (CA). מבחינה טכנית, אין דרך לאמת באמצעות המפתח הפומבי את כתובת השרת אלא אם הכתובת מופיעה בחתימה של ה-CA. אימות כזה מבוצע בד"כ ידנית לאחר האימות הטכני של החתימה הדיגיטלית. הדבר דומה לחתימות הדיגיטליות על מפתחות GPG, רק שכאן ישנה הסתמכות על גורם מאשר במקום רשת אמון.



Two typical X.509 Certification paths

מבנה חתימות הררכי. השרטוט מהאתר http://www.gnu.org/software/gnutls/openpgp.html

כמו רוב יישומי ההצפנה, השימוש ב-SSL נועד לצורך קשר ראשוני בעזרת המפתח הציבורי של השרת ולאחר מכן מבוצע מעבר לשימוש בהצפנה סימטרית לשם המשך התקשורת. מאחר ואין אימות של המשתמש, שכבה זאת מהווה בעצם תשתית לתקשורת מוצפנת עבור יישומים שונים כמו דואר אלקטרוני או פרוטוקול HTTP. לאחר הסבר זה, ניתן להבין בצורה טובה יותר את הרעיון של שימוש ב-PGP כדי לחתום על בקשות לשרתים שונים.

ל-TLS יש גרסה שנקראת TLS-PSK, כאשר PSK הוא קיצור של Pre Shared Key, כלומר מפתח משותף מראש. זאת גרסה מהירה/קלה יותר של TLS המבצעת הצפנה סימטרית בלבד או משתמש במפתח המשותף כדי לזרז את ביצוע פרוטוקול דיפי-הלמן. צורת הצפנה זאת נפוצה יותר במקרים בהם ניהול המערכות נעשה ע"י גורם אחד ו/או קל ליצור קשר כדי לתאם את המפתחת. דוגמה לכך היא הגדרות אינטרנט אלחוטי ביתי – בה כל בני הבית יכולים להעביר בקלות את מפתח ההצפנה בין אחד לשני בעל פה.

ב-SSL, בניגוד ל-SSH, אין בשרת שירות מיוחד המאזין לבקשות וגם אין צורך בניהול משתמשים. השימוש ב-SSL מבוצע ע"י היישומים בשני הצדדים, והם צריכים להכיל תמיכה בסוג תקשורת זה.

בעולם התוכנה החופשית ישנם שלושה מימושים לנושא השימוש ב-SSL. הראשון הוא Network Security Services[31] או NSS בקיצור. זה מימוש של ארגון Mozilla, שהתחיל עוד בימים של Netscape כאשר היא המציאה את SSL. כיום זהו מימוש שנמצא בשימוש בעיקר ע"י גורמים מסחריים שונים וזאת כנראה בעקבות הבשלות שלו, כמו גם הרישוי שלו במספר רב של רישיונות חופשיים המאפשרים גמישות.

המימוש השני הוא OpenSSL[32], פתרון וותיק המתבסס על ספריה חופשית קודמת. זהו מימוש נפוץ מאוד בעולם התוכנה החופשית, אם כי בשנים האחרונות ישנו מעבר ל-GnuTLS[33], המימוש השלישי, בעקבות נושאי רישיון ובפרט תאימות מול רישיון ה-GPL.

בפועל, כל המימושים תואמים לאותם סטנדרטים הנוגעים ל-SSL, אך GnuTLS מתקדם יותר מבחינת תמיכה בגרסאות עדכניות של TLS ועבודה עם מפתחות GPG בנוסף למפתחות על פי התקן של SSL. כמו כן, GnuTLS אינו תומך ב-SSLv2 מאחר ואינו נחשב מאובטח, בעוד שאר המימושים תומכים בו בכל זאת.

בסופו של דבר, אפשרות השימוש ב-SSL נמצאת ברוב היישומי הרשת בעולם התוכנה החופשית. לכל אחד מהמימושים יתרונות משלו, אך המגוון מבטיח כי הם יתחרו ביניהם וידאגו להמשיך ולהשתפר בעתיד. כמו כן, מספר המימושים מוריד את הסיכון במקרה ומתגלה באג באחד המימושים, כי אף אחד מהם לא נמצא בשימוש ב-100% מהתוכנות.

31 http://www.mozilla.org/projects/security/pki/nss/
32 http://www.openssl.org/
33 http://www.gnu.org/software/gnutls/

2.4 DM-Crypt ו-TryeCrypt

בחלקים הקודמים, דובר בעיקר על הצפנת תקשורת (למעט הצפנת קבצים עם GPG), אך כאן אני רוצה להציג נושא אחר לחלוטין והוא הצפנה של מידע על גבי הדיסק הקשיח (כולו או חלקו).

הצפנה זאת, חייבת להיות הצפנה סימטרית מסיבה של יעילות ומהירות תגובה. יש לזכור כי דיסק קשיח נחשב לבעל זמני תגובה איטיים (ביחס למעבד), והשקעת זמן רב מידי בהצפנה / פיענוח תהפוך את הגישה אליו ללא הגיונית מבחינת זמני תגובה.

מעבר לעניין הביצועים, יש להניח כי כל הדיסק הקשיח זמין לגורם אשר ינסה לפרוץ את ההצפנה, ולכן אין בשום מצב לאפשר שמירה של מפתח ההצפנה על הדיסק עצמו. המחמירים גם ידאגו להצפנה של שטח ה-SWAP, למקרה שמערכת ההפעלה תדפדף החוצה מהזיכרון חלקים בעלי מידע רגיש. את מפתח ההצפנה ניתן לשמור על גבי התקן חיצוני או לשנן אותו בעל פה.

נשאלת השאלה – את מה להצפין? האם להצפין קבצים בודדים כמו עם GPG, האם להצפין ספריות, מערכות קבצים או את כל הדיסק. בסופו של דבר נבנתה תשתית המאפשרת למנהל המערכת לבצע הצפנה ברמת ה-block device. המשמעות היא שיכולות הצפנה אינם מוגבלות לדיסקים שלמים אלה ניתן להצפין כל רכיב שמחזיק עליו מערכת קבצים, גם אם הוא חלק מדיסק פיזי, דיסק לוגי, שטח בזיכרון וכו'. יכולת זאת מבוצעת באמצעות שילוב של dm-crypt[34] עם device mapper או dm בקיצור. זאת אותה תשתית המאפשרת יצירה של דיסקים וירטואליים בעלי מנגנונים שרידות כמו RAID לסוגיו.

הסיסמה לפתיחת ההצפנה ניתנת בזמן עליית המערכת, או באופן ידני כאשר רוצים לבצע mount. כך שעל הדיסק ישנו מידע מוצפן בלבד, ללא המפתח ההצפנה באופן מפורש.מרגע שנוצר הרכיב המוצפן, הוא נהיה שקוף לחלוטין למערכת וזמין באופן זהה לכל שאר הדיסקים הוירטואלים. כתוצאה משקיפות זאת, ניתן להצפין כל block device, כולל זה שעליו יושב שטח ה-SWAP ואף שטח הבסיס של מערכת ההפעלה (/ root).

ניתן גם להשתמש ב-TrueCrypt[35] לשם יצירת דיסקים וירטואלים, מחיצות בדיסקים והתקני USB כדי להוסיף ליכולות הצפנת הדיסקים הקיימות בלינוקס. תוכנה זאת זמינה גם ל-Windows ומכילה מספר אפשרויות יחודיות עבורה. התוכנה גם מנסה לספק יכולות סטגנוגרפיה עבור הקבצים המוצפנים, כלומר שמירתם בצורה בהם הם יראו כסתם "רעש" ולא כמידע מוצפן.

3. סיכום

עולם התוכנה החופשית מציע לכל מתכנת ספריות למימוש יכולות הצפנה וחתימה אלקטרונית ברמות שונות. פתרונות אלה נולדו ברובם מצרכים של אנשים או גופים ספציפיים והתפתחו לפרוייקטים שלמים. לדוגמה SSH ו-PGP שהפכו ל-OpenSSH ו-GPG החופשיים.

המימוש בתוכנה חופשית מכיל שלושה יתרונות גדולים – הראשון הוא כי המימוש שקוף לחלוטין ואין שום מידע חסוי שלא ניתן לבדוק ("קופסה שחורה"). השני שניתן להשתמש בקוד באופן חופשי ולשלב אותו בתוכנות אחרות (במקרה של בעיית רישוי, ראינו שאפשר להשתמש במימושים אחרים). והיתרון השלישי הוא שהמימוש עובר בקרה רבה, ואף מעורר עניין בחוגים אקדמיים, מסחריים וממשלתיים. חלק ממימושי האלגוריתמים להצפנה (NSS ו-OpenSSL) אף הוסמכו כעומדים בתקנים של הממשל בארה"ב בנוגע להצפנה (FIPS 140-2).

באמצעות מספר דוגמאות, הראיתי כי נושא ההצפנה נמצא בבסיסה של העבודה עם תוכנה חופשית, באמצעות מנגנונים שנועדו לוודא כי הקוד המועבר להפצות הלינוקס חתום דיגיטלית. זה אינו מאפשר לדעת מי כתב כל שורה בקוד, אבל מאפשר לדעת כי הקוד שמופץ למשתמשי ההפצה הוכנס אליה ע"י חבר הפצה ולא גורם זר. בצד השני, נבדק כי הקבצים המתקבלים מההפצה הם אכן הקבצים החתומים ומנגנון זה דומה קיים בכל ההפצות הגדולות.

בצד המשתמש, הראיתי כי תוכנות הצפנה אינן משהו תיאורטי ולא נגיש, אלא להפך – הן זמינות וניתנות לשילוב ביישומים קיימים ומאפשרות גישה ליכולות הצפנה כמעט בכל מצב בו ישנה תקשורת.

---

34 http://en.wikipedia.org/wiki/Dm-crypt

35 http://www.truecrypt.org/

בסיס העבודה הוא ידע וניסיון אישי, אך במהלך כתיבת העבודה נדרשתי לוודא ולאמת חלק מהמוכר לי בתחום ההצפנה של התוכנה החופשית. בדרך זאת גם העמקתי חלקים רבים מהידע שלי בנושא זה. אני מקווה כי הקורא ילמד מקריאת העבודה באותה מידה שאני למדתי מהכנתה. אשמח לקבל תוספות, הערות והארות לכתובת הדואר kaplanlior@gmail.com .

4. מקורות

- סטגנוגרפיה. (2008, דצמבר 5). *ויקיפדיה, האנציקלופדיה החופשית*.
- קריפטוגרפיה. (2009, אוגוסט 13). *ויקיפדיה, האנציקלופדיה החופשית*.
- על כתב-סתרים, צופן וחידות בספרותנו העתיקה. (2009, פברואר 8). *כתבים מאת אברהם טוביאס*.
- מפתח ציבורי. (2009, יולי 15). *ויקיפדיה, האנציקלופדיה החופשית*.
- סודות ההצפנה. (2003). סיימון סינג.
- חתימה דיגיטלית. (2009, מאי 14). *ויקיפדיה, האנציקלופדיה החופשית*.
- Secure Shell. (2009, August 27). In *Wikipedia, The Free Encyclopedia*.
- Pretty Good Privacy. (2009, August 25). In *Wikipedia, The Free Encyclopedia*.
- Transport Layer Security. (2009, August 24). In *Wikipedia, The Free Encyclopedia*.
- TLS-PSK. (2009, August 31). In *Wikipedia, The Free Encyclopedia*.

5. רישיון

- מסמך זה זמין ברישיון CC-BY-SA 3.0[36].
- זכויות היוצרים על התמונות מהאתר http://www.gnu.org/software/gnutls/openpgp.html שייכות ל-Copyright © 2009 Free Software Foundation, Inc.
- צילומי המסך של התוכנות Enigmail ו-FireGPG נלקחו מאתרי התוכנות.
- השרטוטים המתארים את מפתחות ההצפנה ופרוטוקול דיפי-הלמן נלקחו מויקיפדיה ושייכים לנחלת הכלל.

36 http://creativecommons.org/licenses/by-sa/3.0/