ShadowHound: חלופה ל-SharpHound באמצעות Native PowerShell

מאת יהודה סמירנוב

הקדמה

בעוד ש-SharpHound מהווה אבן יסוד לאיסוף נתונים עבור BloodHound, פריסתו בסביבות מציבה סיכוני זיהוי (אפילו לאחר אובפוסקציה והתאמות אישיות). מערכות EDR משתפרות בזיהוי וסימון של קבצים בינאריים כאלה, בין אם הם נטענים באופן רפלקטיבי ובין שלא. **ShadowHound שואף להימנע מבעיה זו על ידי שימוש ב-Native Powershell או כלים לגיטימיים כמו AD Module.**

יש לציין כי **Domain Controllers עדיין יכולים לסמן שאילתות LDAP חריגות** בסיוע מוצרים כמו Defender for Identity.

עם זאת, על ידי הימנעות מהכנסת קבצים בינאריים זדוניים / אנומליים, אנו יכולים לצמצם את טביעת הרגל שלנו במכונת המטרה ולהימנע מכמה ממסנני ה-LDAP הנפוצים העלולים להפעיל התראות (עוד על כך בהמשך).

ניתן למצוא את הכלי "ShadowHound" כאן.

מהו LDAP?

בבסיסו, פרוטוקול LDAP או בשמו המלא - Lightweight Directory Access Protocol הוא **שפת התקשורת הסטנדרטית לגישה וניהול של סביבות Active Directory**, בהם ארגונים רבים נוטים לנהל את הזהויות, הרשאות, קבוצות ומחשבים שיש בסביבה שלהם. כל אלו נשמרים בסופו של דבר כאובייקטים. בעזרת הפרוטוקול ניתן לקבל מידע אודות האובייקטים, לעדכנם, ליצור אובייקטים חדשים או למחוקם.

המידע מאורגן במבנה של עץ, כאשר כל פריט בתוך העץ הוא אובייקט – משתמש, מחשב, קבוצה או קונטיינר:

```
DC=fndsec,DC=net
│
├── OU=HR,DC=fndsec,DC=net
│   ├── CN=Yehuda Smirnov,OU=HR,DC=fndsec,DC=net
│   └── CN=Hai Vaknin,OU=HR,DC=fndsec,DC=net
│
├── OU=IT,DC=fndsec,DC=net
│   ├── CN=Servers,OU=IT,DC=fndsec,DC=net (Container)
│   │   ├── CN=Server1,CN=Servers,OU=IT,DC=fndsec,DC=net
│   │   └── CN=Server2,CN=Servers,OU=IT,DC=fndsec,DC=net
│   ├── CN=Noam Pomerantz,OU=IT,DC=fndsec,DC=net
│   └── CN=Hoshea Yarden,OU=IT,DC=fndsec,DC=net
│
└── OU=Finance,DC=fndsec,DC=net
    ├── CN=Workstations,OU=Finance,DC=fndsec,DC=net (Container)
    │   ├── CN=WS1,CN=Workstations,OU=Finance,DC=fndsec,DC=net
    │   └── CN=WS2,CN=Workstations,OU=Finance,DC=fndsec,DC=net
```

לכל אובייקט יש כתובת ייחודית – Distinguished Name לדוגמא:

```
CN=Yehuda Smirnov,OU=Users,DC=fndsec,DC=net
```

נקרא משמאל לימין, ה-CN – Common Name מייצג כי שם האובייקט הוא Yehuda Smirnov, אשר נמצא תחת ה-OU – Organizational Unit בשם Users, שנמצא תחת Domain Controller בשם fndsec.net.

כדי להשתמש ב-LDAP יש לייצר שאילתת חיפוש המורכבת **משלושה מרכיבים עיקריים:**

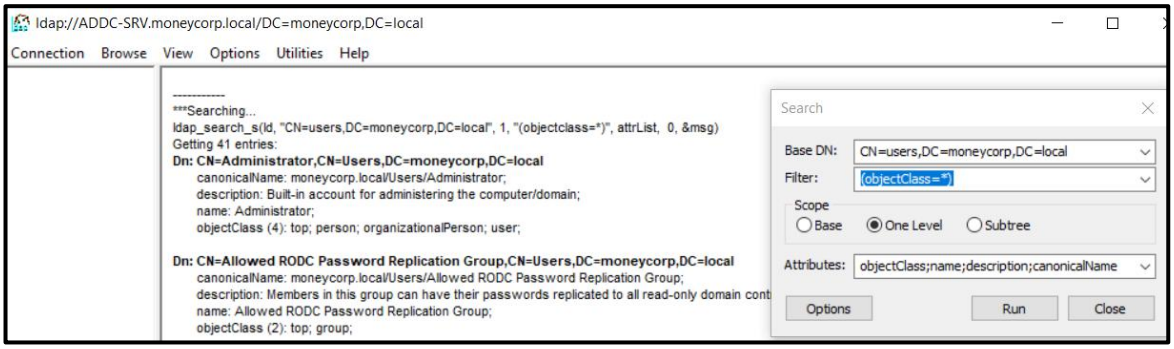
1. **Base**: הנקודה בעץ ממנה מתחיל החיפוש.
2. **Scope**: כמה עמוק בעץ לחפש (ברמה הנוכחית או בכל תתי הענפים).
3. **Filter**: התנאי המדויק של החיפוש.

ניתן להשתמש בכלים כמו ldapsearch, pyldapsearch או ldp.exe שניתן להתקין בעזרת הפקודה:

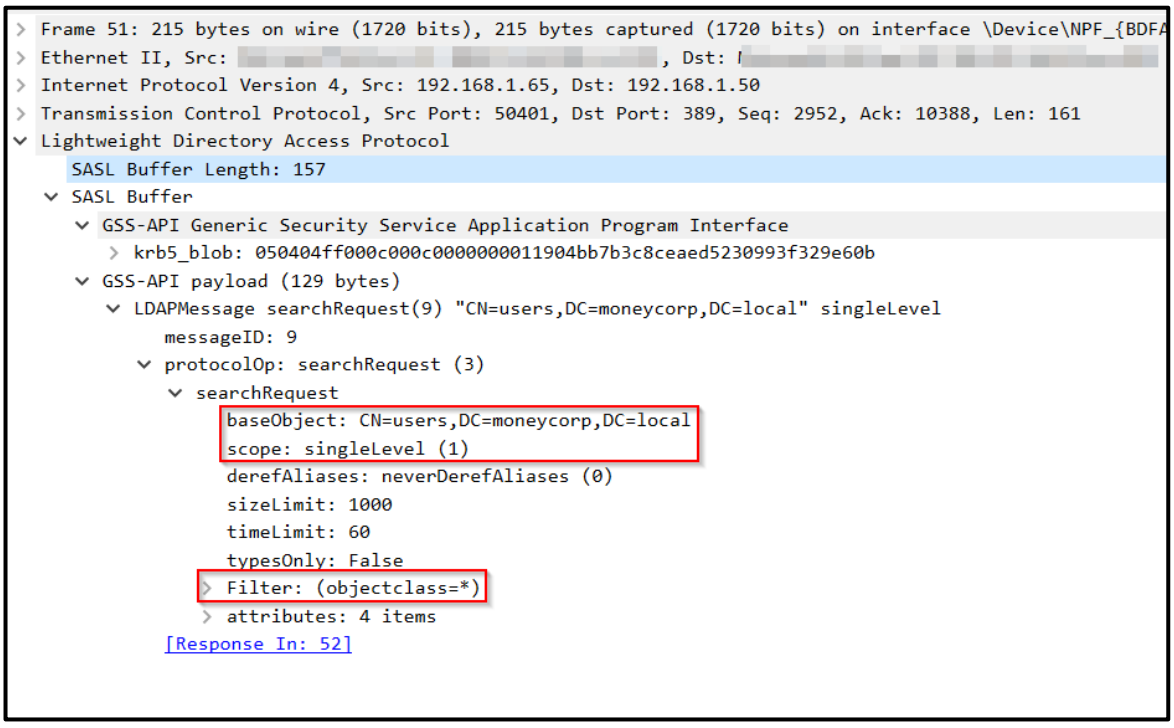
```
Add-WindowsCapability -Name Rsat.ActiveDirectory.DS-LDS.Tools~~~~0.0.1.0 -Online
```

לאחר שנכתוב שאילתא ונשלח אותה, ניתן להתבונן כיצד התעבורה נראית ב-Wireshark, לרבות ה-Base Object, ה-Filter וה-Scope.

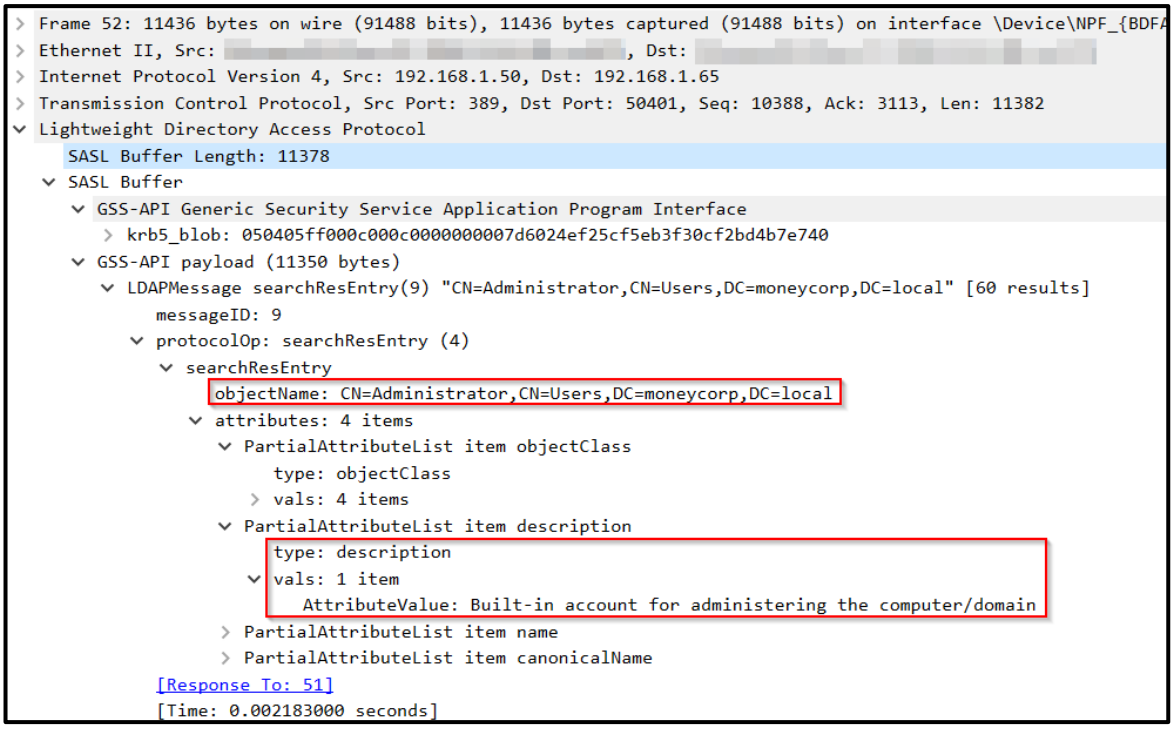
להלן דוגמא לשימוש ב-LDP.exe על מנת לבצע שאילתא וקבלת התשובה:



וכאן נוכל לראות את הבקשה ב-Wireshark:



להלן חלק מהתגובה של ה-DC לבקשה ששלחנו ב-Wireshark:



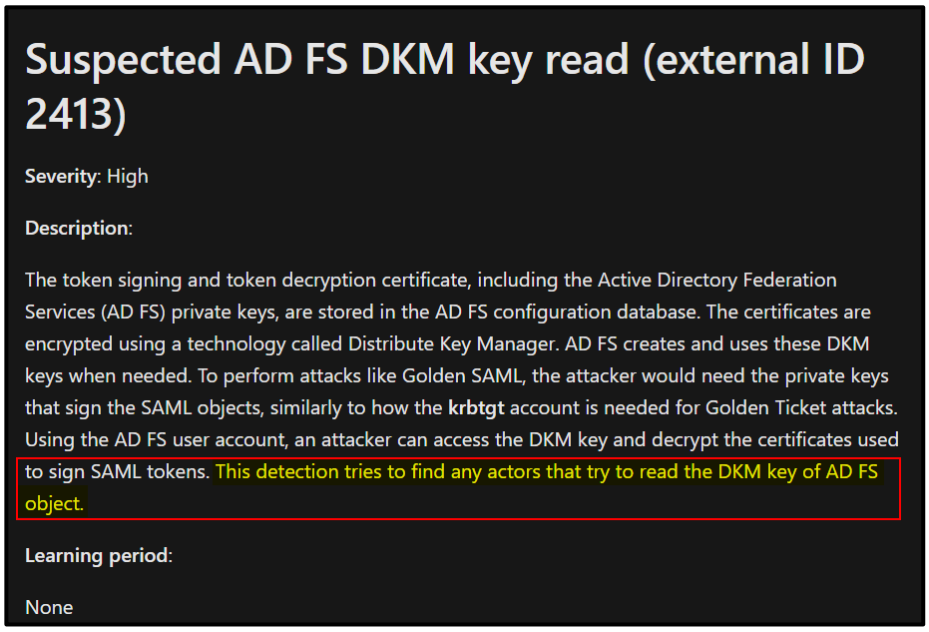
כתוצאה מכך נוכל לקבל מידע מה-DC אודות האובייקטים השונים וההרשאות שלהם. ניתן לבצע את האינומרציה הזו בצורה ידנית, אך ככל שהארגון יהיה גדול יותר, כך הסיבוך שבחיפושי LDAP ידניים עולה.

כיום קיימים כלים שמסייעים לנתח סביבות Active Directory, הבולט שביניהם הוא SharpHound שמהווה Collector המבצע שאילתות LDAP רבות בסביבה. לצידו קיים BloodHound אשר מנתח את הפלטים של SharpHound ומסייע בהבנה מעמיקה של הסביבה. לא ניכנס במסגרת מאמר זה לכלי, אך במידה ואינכם מכירים אותו, אנו ממליצים להכיר ולהתנסות.

זיהויים של ADExplorer Snapshot (בערך)

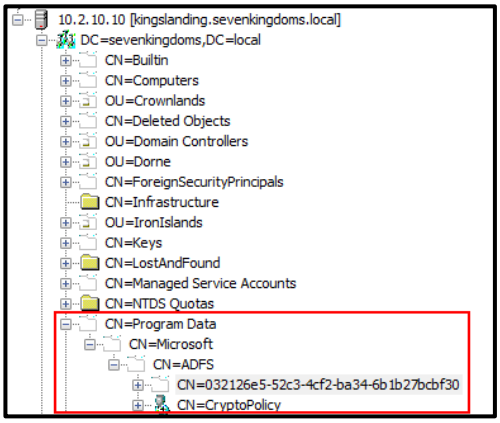
חלופה אחת ל-SharpHound היא שימוש ב-ADExplorer ליצירת snapshots של סביבת Active Directory ולאחר מכן המרתם לקבצי JSON ש-BloodHound יודע לקבל (כמו פלטי SharpHound), באמצעות כלים כמו ADExplorerSnapshot.py. למרות ששיטה זו יכולה להיות יעילה, היא עלולה להיכשל בדומיינים גדולים כאשר מתמודדים עם חיבור ירוד או כאשר ה-VPN מתנתק אוטומטית לאחר פרק זמן מסוים. במקרים כאלה **ADExplorer עלול לייצר שגיאה עקב בעיות חיבור, ואתם תיוותרו ללא נתוני snapshot.**

במהלך בדיקה שביצענו לאחרונה, נתקלנו גם בזיהויים בעת שימוש ב-ADExplorer באופן ספציפי, **כאשר Active Directory Federation Services (ADFS) פרוס, יצירת snapshot עם ADExplorer הפעילה התראה** מכיוון שהכלי קורא את קונטיינר ה-LDAP של ה-ADFS, כפי שמצוין בסעיף הזיהויים של Microsoft Defender for Identity:



[*Security alerts in Microsoft Defender for Identity - MSDN*]

בהתקנה רגילה של ADFS, הקונטיינר קיים תחת הקונטיינר "Program Data":



לאור אתגרים אלו, הבנו שאנו זקוקים לפתרון שיוכל להימנע הן מבעיות הזיהוי והן להתמודד ביעילות עם דומיינים גדולים. זה הוביל אותנו (איתי ישר, חברי לצוות, ואני) לפתח את ShadowHound, ולהפוך רעיון ממדף המו"פ שלנו לכלי ממשי.

הימנעות מזיהויים של Defender for Identity

במהלך בדיקות וכן בסביבת המעבדה שלנו, שמנו לב **שזיהויי** Defender for Identity מתרחשים עבור שאילתות כגון: `Get-ADUser -Filter *` או בעת שימוש ב-pyLdapsearch עם פילטר LDAP `(objectClass=*)` אשר ידוע בתור השאילתה המבוצעת על ידי ADExplorer (ולכן נחשב פחות opsec):

LDAP search events

23 items | Customize columns | Filter

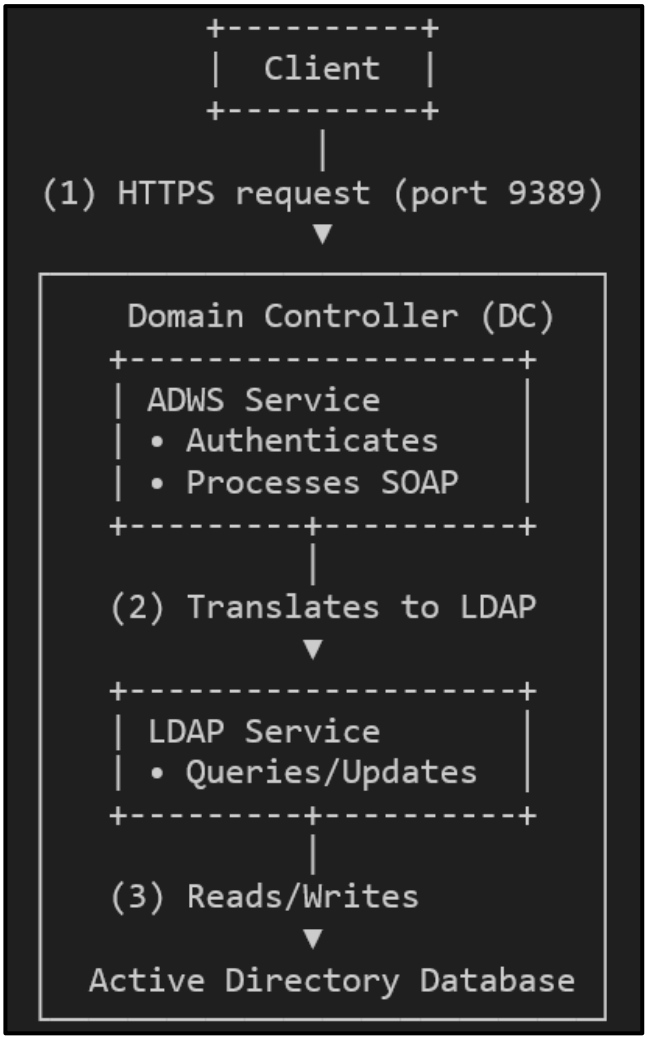
Timestamp ↓	Base object	Search scope	Search filter	Enumeration t...	Sensitive type	Queried groups
Nov 6, 2024 5:00 PM	DC=essos,DC=local	WholeSubtree	(objectClass=*)	AllObjects	None	
Nov 6, 2024 3:39 PM	DC=essos,DC=local	WholeSubtree	(objectClass=*)	AllObjects	None	

לאחר מחקר, גילינו שהשאילתה `(objectGuid=*)` **לא גרמה להפעלת זיהויים שלDefender for Identity** (לעת עתה), ולכן בחרנו להשתמש בפילטר LDAP זה עבור הכלי. כמו כן, נציין כי ShadowHound הוא למעשה סקריפט post-processing אשר ממיר פלט של DirectorySearcher או ADModule לפלט בפורמט של pyLdapSearch. עם זאת pyLdapSearch, אינו תומך ב-ADWS (לפחות נכון לכתיבת שורות אלו).

מהו ADWS?

Active Directory Web Services (או ADWS) הוא שירות Web שמייקרוסופט הציגה כדי לספק דרך אלטרנטיבית לתקשר עם Active Directory. במקום לדבר ישירות בפרוטוקול LDAP, **לקוחות יכולים לשלוח בקשות Web** סטנדרטיות (SOAP על גבי HTTP/S) ל-Domain Controller, אשר מתרגם אותן מאחורי הקלעים לפעולות LDAP. השירות פועל כברירת מחדל מעל port 9389.

להלן תרשים המתאר את ה-Flow של בקשה:



ה-Flow בגדול מתבצע כך:

1. תחילה, הלקוח שולח בקשת HTTP ל-Port 9389.
2. לאחר מכן, ה-DC מקבל את הבקשה ומבצע אימות.
3. ה-SOAP מתפרסר מתוך הבקשה ומתורגם לשאילתת LDAP.
4. לאחר מכן שירות LDAP על ה-DC מתשאל את עצמו ומחזיר תשובה ללקוח.

הלקוח הנפוץ ביותר של ADWS הוא לא אחר מאשר **מודול ה-Active Directory של PowerShell. כל פקודה** שתריצו, כמו Get-ADUser, Get-ADObject **משתמשת ב-ADWS** (כלומר בקשות HTTP עם הודעות SOAP) מאחורי הקלעים **כדי לתקשר עם ה-DC** ולבצע את שאילתות ה-LDAP.

להלן דוגמא ל-Body של בקשת SOAP:

```
<soap:Body>
  <SearchRequest xmlns="http://schemas.microsoft.com/ldap/operations">
    <baseObject>CN=Users,DC=fndsec,DC=net</baseObject>
    <scope>sub</scope>
    <derefAliases>neverDerefAliases</derefAliases>
    <sizeLimit>0</sizeLimit>
    <timeLimit>0</timeLimit>
    <typesOnly>false</typesOnly>
    <filter>
      <present>
        <attributeDesc>objectClass</attributeDesc>
      </present>
    </filter>
    <attributes>
      <attribute>distinguishedName</attribute>
      <attribute>cn</attribute>
      <attribute>sAMAccountName</attribute>
    </attributes>
  </SearchRequest>
</soap:Body>
```

בבקשה הנ"ל תחת התגיות filter ניתן לראות את התגית present שמשעותה קיום. כלומר, קיום של attribute בשם objectClass.

לאחר מכן, ניתן לראות כי אנו מבקשים לאחזר את ה-attributes הבאים: distinguishedName, cn ו-sAMAccountName. כל זאת, מתוך ה-container הבא: CN=Users,DC=fndsec,DC=net.

עכשיו שיש לנו הבנה בסיסית בפרוטוקול ה-ADWS, אפשר לעבור ולדבר על כלי תקיפה.

Shoutout ל-Soaphound

בהקשר של ADWS, תחילה ראוי להזכיר את SoapHound של **FalconForce**. **הם היו הראשונים לפרסם בפומבי על השימוש ב-ADWS** לאינומרציית Active Directory ויצרו כלי למשימה.

אף על פי ש-SoapHound מספק פונקציונליות חשובה ומחקר מדהים, הוא מהווה קובץ בינארי נוסף שאנו צריכים להכניס לנקודות קצה מנוטרות, ולצערנו נתקלנו בקשיים מסוימים בשימוש בו אל מול דומיין גדול מאוד (ולכן יצרנו את הדגל SplitSearch **אשר מפצל את האינומרציה לקונטיינרים** וכן שאלנו מהם את רעיון פיצול החיפוש עם הדגל LetterSplitSearch כדי לפצל עוד יותר את החיפוש תחת קונטיינר לאובייקטים המתחילים באותיות a ולאחר מכן b ואז c וכן הלאה).

כיצד להשתמש ב-ShadowHound

ShadowHound מגיע בשתי גרסאות:

1. **מבוסס ADModule:** סקריפט זה ממנף את Active Directory Module בעזרת ה-cmdlet Get-ADObject אשר עושה שימוש באמצעות פרוטוקול ה-Active Directory Web Services מעל port 9389 אל מול Domain Controller.
2. **מבוסס DirectorySearcher:** סקריפט זה משתמש במחלקת ה-DirectorySearcher, ומבצע שאילתות LDAP ישירות.

שימוש בכלי ShadowHound הוא פשוט. להלן דוגמאות:

שימוש בגרסה מבוססת ADModule:

```
Import-Module .\ShadowHound-ADM.ps1

# דוגמה 1: שימוש בסיסי עם פרמטר חובה
ShadowHound-ADM-OutputFilePath "C:\Results\ldap_output.txt"

# דוגמה 2: ציון Domain Controller ומסנן LDAP מותאם אישית
ShadowHound-ADM -Server "dc.domain.local" -OutputFilePath "C:\Results\ldap_output.txt" -
LdapFilter "(objectClass=user)"
```

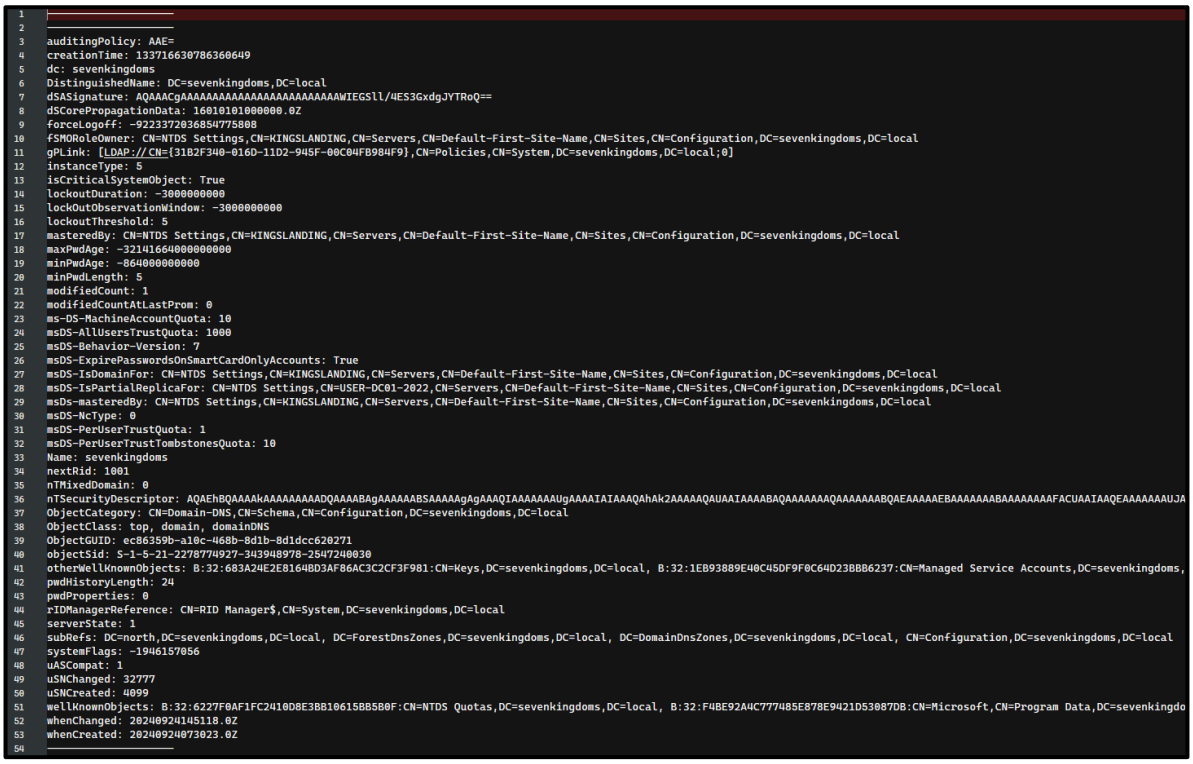
```
# דוגמה 3: שימוש ב-credential חלופיים וציון Search Base
$cred = Get-Credential
ShadowHound-ADM -OutputFilePath "C:\Results\ldap_output.txt" -Credential $cred -SearchBase "CN=Infrastructure,DC=sevenkingdoms,DC=local"

# דוגמה 4: פיצול החיפוש בין קונטיינרים ברמה העליונה עם פיצול לפי אותיות
ShadowHound-ADM -OutputFilePath "C:\Results\ldap_output.txt" -SplitSearch -LetterSplitSearch

# דוגמה 5: אינומרציית תעודות
ShadowHound-ADM -OutputFilePath "C:\Results\output.txt" -Certificates

# דוגמה 6: החרגת קונטיינר (כמו קונטיינר ה-ADFS) מהאינומרציה
ShadowHound-ADM -OutputFilePath "C:\Results\output.txt" -SplitSearch -ParsedContainers "./parsed.txt"
# הקובץ parsed.txt מכיל רשימה של Distinguished Names של קונטיינרים, מופרדים בשורה חדשה
# CN=Program Data,DC=sevenkingdoms,DC=local
```

שימוש בגרסה מבוססת DirectorySearcher:

```
Import-Module .\ShadowHound-DS.ps1

# דוגמה 1: שימוש בסיסי עם פרמטר חובה
ShadowHound-DS -OutputFile "C:\Results\ldap_output.txt"

# דוגמה 2: ציון Domain Controller
ShadowHound-DS -Server "dc.domain.local" -OutputFile "C:\Results\ldap_output.txt"

# דוגמה 3: שימוש במסנן LDAP מותאם אישית
ShadowHound-DS -OutputFile "C:\Results\ldap_output.txt" -LdapFilter "(objectClass=computer)"

# דוגמה 4: ציון Search Base
ShadowHound-DS -OutputFile "C:\Results\ldap_output.txt" -SearchBase "CN=Infrastructure,DC=sevenkingdoms,DC=local"
```

```
# דוגמה 5: אינומרציית אובייקטים הקשורים לתעודות
ShadowHound-DS -OutputFile "C:\Results\cert_output.txt" -Certificates

# דוגמה 6: שימוש ב-credentials חלופיים
$cred = Get-Credential
ShadowHound-DS -OutputFile "C:\Results\ldap_output.txt" -Credential $cred
```

לאחר הרצת הסקריפט, תקבלו קובץ פלט המכיל תוצאות חיפוש LDAP בפורמט מוכן לעיבוד:



```
auditingPolicy: AAE=
creationTime: 133716630786360649
dc: sevenkingdoms
DistinguishedName: DC=sevenkingdoms,DC=local
dSASignature: AQAAACgAAAAAAAAAAAAAAAAAAAAAAAAAWIEGSll/4ES3GxdgJYTRoQ==
dSCorePropagationData: 16010101000000.0Z
forceLogoff: -9223372036854775808
fSMORoleOwner: CN=NTDS Settings,CN=KINGSLANDING,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=sevenkingdoms,DC=local
gPLink: [LDAP://CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policies,CN=System,DC=sevenkingdoms,DC=local;0]
instanceType: 5
isCriticalSystemObject: True
lockoutDuration: -3000000000
lockOutObservationWindow: -3000000000
lockoutThreshold: 5
masteredBy: CN=NTDS Settings,CN=KINGSLANDING,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=sevenkingdoms,DC=local
maxPwdAge: -32141664000000000
minPwdAge: -864000000000
minPwdLength: 5
modifiedCount: 1
modifiedCountAtLastProm: 0
ms-DS-MachineAccountQuota: 10
msDS-AllUsersTrustQuota: 1000
msDS-Behavior-Version: 7
msDS-ExpirePasswordsOnSmartCardOnlyAccounts: True
msDS-IsDomainFor: CN=NTDS Settings,CN=KINGSLANDING,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=sevenkingdoms,DC=local
msDS-IsPartialReplicaFor: CN=NTDS Settings,CN=USER-DC01-2022,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=sevenkingdoms,DC=local
msDS-masteredBy: CN=NTDS Settings,CN=KINGSLANDING,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=sevenkingdoms,DC=local
msDS-NcType: 0
msDS-PerUserTrustQuota: 1
msDS-PerUserTrustTombstonesQuota: 10
Name: sevenkingdoms
nextRid: 1001
nTMixedDomain: 0
nTSecurityDescriptor: AQAEhBQAAAAkAAAAAAAAADQAAAABAgAAAAAABSAAAAAgAgAAAQIAAAAAAAUgAAAAIAIAAAQAhAk2AAAAAQAUAAIAAAABAQAAAAAAAQAAAAAAAABQAEAAAAAEBAAAAAAAABAAAAAAAAFACUAAIAAQEAAAAAAAUJA
ObjectCategory: CN=Domain-DNS,CN=Schema,CN=Configuration,DC=sevenkingdoms,DC=local
ObjectClass: top, domain, domainDNS
ObjectGUID: ec86359b-a10c-468b-8d1b-8d1dcc620271
objectSid: S-1-5-21-2278774927-3439489978-2547240030
otherWellKnownObjects: B:32:683A24E2E8164BD3AF86AC3C2CF3F981:CN=Keys,DC=sevenkingdoms,DC=local, B:32:1EB93889E40C45DF9F0C64D23BBB6237:CN=Managed Service Accounts,DC=sevenkingdoms,
pwdHistoryLength: 24
pwdProperties: 0
rIDManagerReference: CN=RID Manager$,CN=System,DC=sevenkingdoms,DC=local
serverState: 1
subRefs: DC=north,DC=sevenkingdoms,DC=local, DC=ForestDnsZones,DC=sevenkingdoms,DC=local, DC=DomainDnsZones,DC=sevenkingdoms,DC=local, CN=Configuration,DC=sevenkingdoms,DC=local
systemFlags: -1946157056
uASCompat: 1
uSNChanged: 32777
uSNCreated: 4099
wellKnownObjects: B:32:6227F0AF1FC2410D8E3BB10615BB5B0F:CN=NTDS Quotas,DC=sevenkingdoms,DC=local, B:32:F4BE92A4C777485E878E9421D53087DB:CN=Microsoft,CN=Program Data,DC=sevenkingdo
whenChanged: 20240924145118.0Z
whenCreated: 20240924073023.0Z
```

התמודדות עם דומיינים גדולים באמצעות ADWS

אחד האתגרים בהתמודדות עם סביבות Active Directory נרחבות הוא הפוטנציאל לשאילתות להיתקל ב-timeout בעת שימוש ב-ADWS (מה שגורם לשגיאת "invalid enumeration context").

גרסת ה-ADModule של ShadowHound מתמודדת עם כך באמצעות הפונקציונליות של Recurse, SplitSearch ו-LetterSplitSearch.

- **שוב תודה ל-FalconForce על המחקר ועל מציאת הפתרון!**

- **SplitSearch**: כאשר אפשרות זו מופעלת ShadowHound, יפצל את החיפוש בין קונטיינרים ברמה העליונה ב-root של הדומיין או בתוך ה-distinguished name של קונטיינר שסופק (במידה ויש קונטיינרים ענקיים ומעצבנים או שתרצו להיות ספיצפיים וממוקדים), ויריץ שאילתה על כל אחד בנפרד. גישה זו מסייעת בניהול מערכי נתונים גדולים על ידי פירוקם לחלקים קטנים יותר.

- **Recurse**: אם קונטיינר לא מצליח להחזיר תוצאות בתוך חלון של כ-30 דקות (הנקודה שבה ה-enumeration context פג עקב מגבלות ADWS אזי ShadowHound יפצל אוטומטית את אותו קונטיינר לתתי-הקונטיינרים שלו וינסה לאחזר את הנתונים שוב. גישה רקורסיבית זו מבטיחה (בתקווה) שלא נאבד מידע חשוב.

- **LetterSplitSearch**: בדומה ל-SoapHound אם דגל זה מסופק, השאילתות יפוצלו ל-(cn=a*), (cn=b*), וכו'. אם שאילתה כזו נכשלת, היא תפוצל פעם נוספת - (cn=aa*), (cn=ab*), (cn=ac*). ניתן לשלב דגל זה עם SplitSearch כדי לפצל כל קונטיינר לשאילתות קטנות עוד יותר.

המרת הנתונים עם BofHound

לאחר שאספתם את הנתונים (שאמורים להיראות כמו פלט של ldapsearch), השלב הבא הוא להמיר אותם לקבצי JSON תואמי BloodHound באמצעות BofHound, שעודכן לאחרונה:

```
python3 bofhound.py -i ldap_output.txt -p All --parser ldapsearch
```

BofHound יעבד את פלט ה-LDAP וייצר את קבצי ה-JSON שניתן לייבא ל-BloodHound CE לצורך ניתוח.

פיצול קבצי ה-JSON

בהתאם לגודל הקובץ (>100MB), ייתכן שתרצו לפצל את קובץ ה-JSON באמצעות כלים כמו ShredHound או snippet של mgeeky.

מניסיוננו, עבור דומיינים גדולים, גם כאשר BloodHound CE מדווח שהנתונים נטענו, כדאי לבדוק שוב ולנסות לשאול נתונים מה-JSONs כדי לוודא שהם אכן נטענו כראוי.

אם הנתונים לא נטענו כראוי, יש להשתמש באחד הכלים לעיל כדי לפצל אותם, גם אם נראה שמדובר בכמות קטנה של מחשבים.

אסטרטגיות זיהוי

השיטה היעילה לזהות את ShadowHound (וכלים דומים המבצעים אינומרציית AD) היא **לעקוב אחר זהויות אשר מבצעות מספר גבוה במיוחד של קריאות של רשומות LDAP בפרק זמן מסוים**, מכיוון שהתנהגות זו יכולה להצביע על אינומרציה זדונית הדומה לפעילות של SharpHound. כלל הזיהוי שלElastic, למשל, מסמן חריגות כאלה על ידי זיהוי בקשות מופרזות ל-object attributes. זה צוין גם בבלוג של FalconForce תחת הסעיף "Detecting SOAPHound".

רעיון מעניין נוסף עשוי להיות **הקמת honeypots בקונטיינרים של LDAP המופעלים בעת גישת קריאה**. זה יכול להוות אינדיקטור לתקיפה.

סיכום

ShadowHound מציע גישה חלופית לאיסוף נתוני Bloodhound על ידי ביטול הצורך לפרוס אתSharpHound או קבצים בינאריים אחרים בנקודת הקצה של המטרה. על ידי מינוף AD Module או מחלקת ה-DirectorySearcher, ניתן להפחית את סיכוי לזיהוי בנקודת הקצה עצמה.

בעוד ש-ShadowHound עשוי למזער את הזיהוי ברמת נקודת הקצה, פעילות הרשת צריכה תמיד להישאר שיקול. כדאי לשים לב לשאילתות המבוצעות ולהיות מודע לכך ש-DCs (בסיוע MDI ודומיו) יכולים לסמן פעילות LDAP חריגה או חשודה (ועם הפיכת ה-Machine Learning לדבר נפוץ יותר, אנו צופים שזה יקרה במוקדם או במאוחר).

נסו את ShadowHound וראו כיצד הוא עובד עבורכם. אם יש לכם שאלות, נתקלתם בבעיות, או שיש לכם משוב - בין אם זה על משהו שאינו מדויק או רעיונות לשיפור - אל תהססו ליצור קשר. תוכלו למצוא אותי ב-X (לשעבר Twitter) תחת ה-handle @yudasm_, ב-Linkedin או במייל info@fndsec.net.

על המחבר

המאמר נכתב במקור לטובת בלוג המחקר "Friends & Security" המנוהל על ידי צוות של חוקרים – **הושע, חי, יהודה ונעם**, שהם קודם כל חברים (ועל כן השם של הבלוג). הבלוג מתמקד במחקרים טכניים בתחומי התקיפה, ההגנה, ומחקר Low-Level בסביבות Windows וענן.

מטרתנו היא להתחיל ולשתף את מחקרינו עם הקהילה בשפה העברית. אנו שמחים על ההזדמנות לפרסם לראשונה בבלוג DigitalWhisper, הנחשב בעיננו לאבן יסוד בקהילת הסייבר הישראלית.

מקורות מידע

- ShadowHound Github Repo
- ADExplorer on Engagements - TrustecSec
- ADExplorerSnapshot.py Github Repo
- Security alerts in Microsoft Defender for Identity - MSDN
- SoapHound – tool to collect Active Directory data via ADWS - FalconForce
- BofHound Github Repo
- ShrepHound Github Repo or mgeeky's bh_split2.py snippet
- Elastic's LDAP volume detection rule