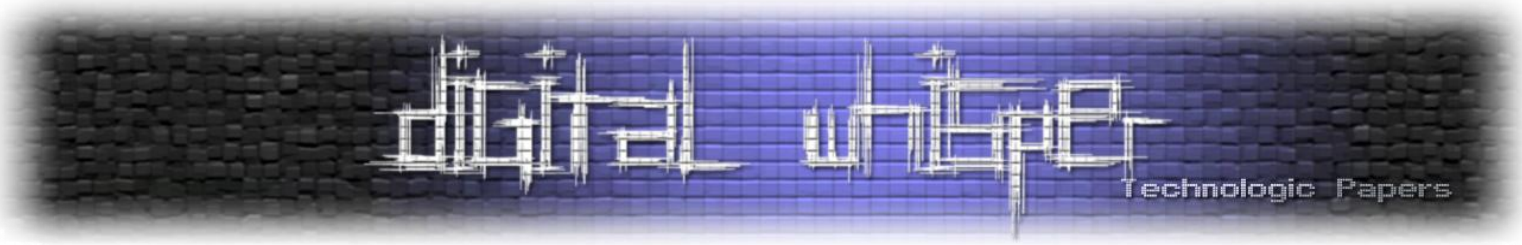


תנועה רוחבית בין תחנות ושרתים בסביבת Azure

מאת בן זמיר

הקדמה

בסביבה ארגונית היברידית, תחנות עבודה ושרתים יכולים להיות מחוברים ל-Azure במתכונתEntra Joined או Hybrid joined.

תנועה רוחבית (Lateral Movement), בייחוד בין תחנות לא היברידיות (Entra Joined), עלולה להציב אתגרי אבטחה ייחודיים, מאחר ומנגנוני האימות "המסורתיים", כמו NTLM או Kerberos, אינם פועלים כמו במתכונת הרגילה בסביבות Domain / Hybrid Joined.

בניגוד לסביבות Domain קלאסיות, בהן נעשה שימוש במשתמשים מקומיים או מבוססי Active Directory, בסביבת Entra ID נעשה שימוש בזהויות ענניות. ניהול הזדהויות מתבצע בענן, כשמערכת ניהול ההרשאות המרכזית שונה.

היעדר הזדהות תוך שימוש ב-NT Hash והמעבר לשימוש ב-Primary Refresh Token (PRT) במקום, משנים לחלוטין את מנגנוני האימות והזדהות בין תחנות ושרתים.

מטרת המאמר, היא להתמקד ולנתח את תהליך ההזדהות בין תחנות בסביבת Entra ID, לבחון את אופן פעולת הפרוטוקולים המשמשים להזדהות ולהציג את האיומים האפשריים על אבטחת המידע במתאר זה.

הזדהות מול Entra Joined Device

לתחנה אשר מצורפת ל-Entra, ניתן להתחבר (בדומה לסביבת Domain) באמצעות זהות עננית. במצב זה, תהליך הזיהוי מתבצע מול מערכת מרכזית חיצונית.

המצב הפשוט יותר הוא במקרה ומדובר בתחנה היברידית. שם ניתן כמובן להשתמש בזהויות מה-Domain, תוך שימוש בפרוטוקולים Kerberos ו-NTLM, אשר נשארים בשימוש ויכולים לאפשר תנועה רוחבית.

חשוב לשים לב, שניתן לנצל תחנות היברידיות, תוך שימוש בזהות דומיינית בעלת הרשאת ניהול מקומית, לחילוץ פרטי הזדהות עננים, כגון PRT ו-Session Key, כפי שמתואר בהרחבה במאמרים רבים. מומלץ לקרוא את המאמר של Dirk-Jan בנושא: Digging further into the Primary Refresh Token - dirkjanm.io

הפוטנציאל במצב זה טמון ביכולת לנצל הרשאות גבוהות ב-Domain, לגנוב session של זהות עננית, ובכך לעקוף את מנגנון ה-MFA. קיימות מספר שיטות לביצוע פעולה זו, וכל אחת מהן יכולה לשמש בסיס למאמר שלם בפני עצמו. המודעות לסיכון במצב זה חשובה במיוחד, שכן נפוץ למצוא תחנות עבודה או שרתים היברידיים שבהם נעשה במקביל שימוש בזהויות עננים בעלות הרשאות גבוהות. תרחיש המהווה יעד אטרקטיבי במיוחד לתוקפים.

אך מה קורה כשהתחנה לא מצורפת לדומיין כלל, אלא רק לסביבת Azure, ולא ניתן לנצל הרשאות ב-Domain לביצוע תנועה רוחבית?

כפי שהוזכר, כאשר משתמש Entra ID מחובר לתחנה אשר מצורפת ל-Azure (Entra ID Joined) בלבד, אין שימוש ב-NTLM או Kerberos. במקום זאת, התחנה משתמשת בפרטי ההזדהות של המשתמש, ומקבלת PRT ו-Session Key מ-Azure, אשר משמשים את התחנה לביצוע הזדהות SSO ונשמרים בתהליך ה-LSASS.

ווקטור תקיפה מוכר בסביבות Entra ID, כולל חילוץ של ה-PRT וה-Session Key מהתחנה, באמצעות הרשאות ניהול מקומיות. תקיפה זו מזכירה את מתקפות ה-Dump הקלאסיות, שבהן תוקפים היו מחלצים NT Hash של משתמשי דומיין שהיו מחוברים לתחנה, תוך ניצול גישה לתהליך ה-LSASS. בדומה לאותן מתקפות מיתולוגיות, גם כאן חילוץ המידע המאוחסן בתחנה מאפשר לתוקף להתחזות לזהויות העננית הפעילות, ובכך לגשת למשאבים ארגוניים שונים ללא צורך באימות נוסף, שכן לרוב ה-PRT בתחנות העבודה כוללים MFA Claim, ומאפשרים שימוש בו לקבלת סשן פעיל.

בשנים האחרונות טבעת האבטחה התהדקה באופן משמעותי מול מתקפות חילוץ פרטי הזדהות מה-LSASS. למשל, במידה וה- TPM (Trusted Platform Module) פעיל בתחנה, נעשה בו שימוש לאחסון מאובטח של המפתחות הקריפטוגרפיים. זה בשילוב הגנות נוספות כמו Credential Guard ו-PPL וניטור התנהגותי, הקשו

באופן משמעותי על גניבה ושימוש ב-PRT וה-Session Key. אז איך מתבצעת הזדהות מול תחנות אשר משוייכות ל-Entra בלבד בארגון? איך ניתן להזדהות בפרוטוקולים כגון SMB ו-RDP?

NegoEx PKU2U

תהליך ההזדהות ב-Entra Identity נעשה תוך שימוש בפרוטוקול NegoEx, אשר בעצם משמש כפרוטוקול תיווך לבחירת פרוטוקול ההזדהות, והינו הרחבה של GSSAPI SPNEGO (Simple and Protected Negotiation Mechanism). פרוטוקול זה פועל כברירת מחדל בתחנות Windows אשר מצורפות ל-Entra.

במערכות Windows, NegoEx נועד לתמוך בתהליך אימות זהות גמיש, והוא מאפשר משא ומתן (Negotiation) בין לקוח (נקרא Initiator), לשרת (נקרא Accpetor), לבחירת פרוטוקול האימות המתאים ביותר מבין כמה אפשרויות נתמכות.

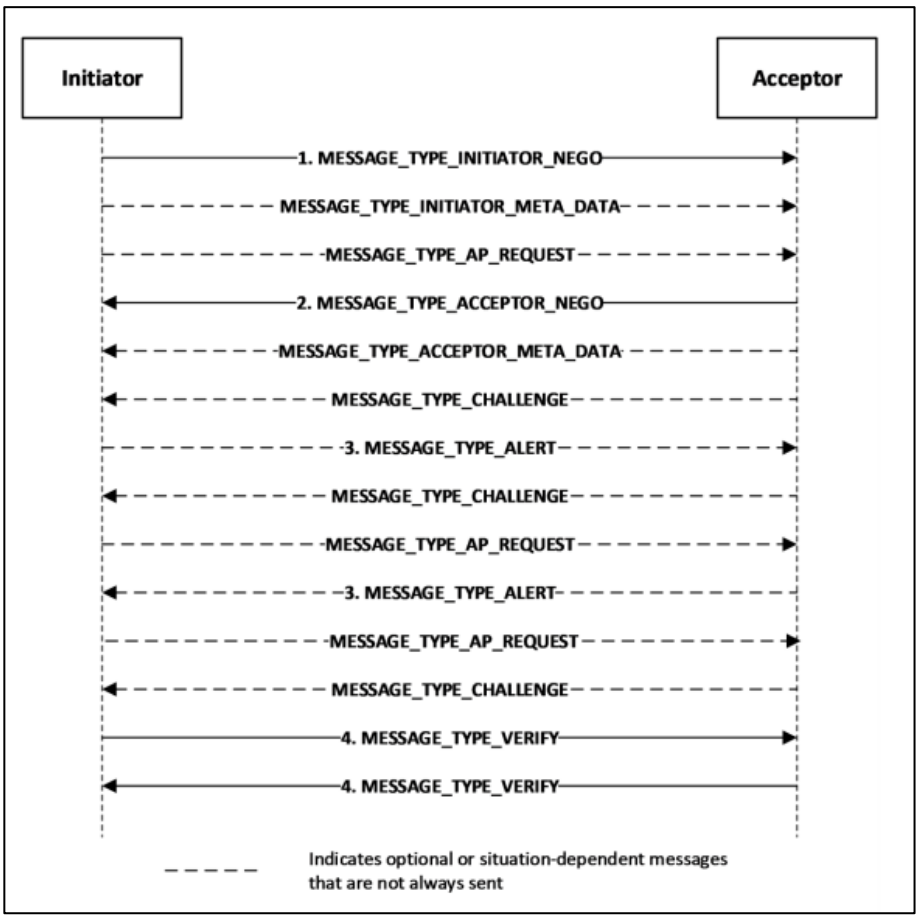
מגרסאת Windows 2008 R2 ו-Windows 7, נתמכת שיטת ההזדהות PKU2U על ידי ה-SSP (בפשטות, מדובר בסיפריות המשמשות לתמיכה בתהליכי הזדהות), תוך שימוש בסיפריה negoexts.dll. הזדהות זו משמשת כתחליף ל-Kerberos ו-NTLM בסביבה מבוססת Azure.

אז מה זה בעצם PKU2U?

PKU2U (Public Key User to User) הוא פרוטוקול הזדהות Peer to Peer, אשר נעשה בו שימוש בהזדהות ללא Domain Controller. הפרוטוקול מבוסס על Kerberos PKINIT, ומאפשר הזדהות בין שני הצדדים (P2P) בפרוטוקול Kerberos, תוך שימוש בתעודה (Certificate).

בפשטות, הזדהות של זהות Entra ID מול תחנה, תעשה על ידי הזדהות ישירה (Peer To Peer), בתיווך NegoEx ושימוש ב-PKU2U, הזדהות מבוססת תעודה.

המאמר של מור רובין, שהוצג ב-Blackhat, מסביר לעומק את התהליך. מומלץ מאוד לקרוא אותו (קישור מצורף בסוף המאמר) אבל בכל מקרה הנה הסבר מקוצר ותרשים שנלקח מהמאמר:



א. התהליך מתחיל כאשר ה-Initiator (הלקוח) שולח ל-Acceptor (השרת) הודעת התחלה, שכוללת מחרוזת רנדומלית, שהיא בעצם המזהה של ה-Session.

ב. ההודעה כוללת פרטי זיהוי של מנגנוני האימות המוצעים. במקרה הזה, יוצע פרוטוקול PKU2U, אשר מאפשר הזדהות על ידי P2P Certificate.

ג. לאחר מכן, הצד השני, ה-Acceptor, שולח הודעה בעלת מבנה דומה, רק עם מזהה Session משלו.

ד. בשלב הבא, ישלחו הודעות AP_REQ ו-AP-REP, הכוללות קביעה של תהליך ההזדהות על ידי PKU2U, תוך שימוש ב-Kerberos PKINIT.

ה. במידה ותהליך האימות הצליח, השלב הבא הוא הודעות Verify של שני הצדדים, במטרה לבחון את מהימנות הבקשות (Message Integrity).

אז איך ה-Initiator מקבל את התעודה המשמשת להזדהות ומה הדרישות באופן כללי כדי לבצע הזדהות מול תחנה המצורפת ל-EntraID?

כדי לבצע חיבור בתצורה הלגיטימית צריכים להתקיים מספר תנאים:

1. התנאי הראשון הוא ששתי התחנות, צריכות להיות Entra Joined לאותו ה-Tenant. כלומר, אם ננסה להתחבר מתחנה אשר איננה מצורפת לשום Tenant, או לחילופין מצורפת ל-Tenant אחר, החיבור ייכשל (גם אם פרטי הזדהות המלאים ידועים לנו וישנה הרשאה מתאימה)

הסיבה הטכנית מתחלקת לשני מרכיבים:

א. כדי לבצע את תהליך ההזדהות לקבלת תעודת ה-P2P, צריכים להיות מוגדרים בתחנה הפרטים הנדרשים (ה-Tenant, ה-Endpoints המשמשים להזדהות ועוד).

ב. לתחנה אשר מצורפת ל-Entra מונפקת תעודה המשמשת לזיהוי מול Azure. תהליך בקשת התעודה כולל שימוש בתעודה לאימות זהות התחנה עצמה (בהמשך אציג את התהליך עצמו).

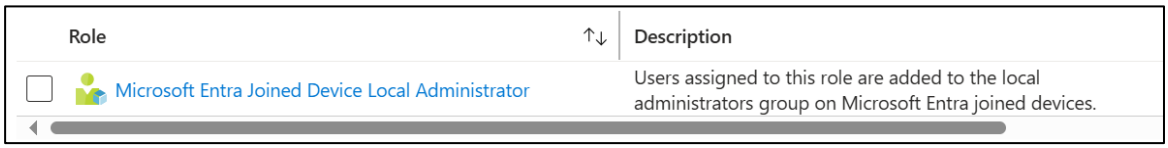
יש בלוג מאוד מעניין על גניבת זהות והתחזות למכונה המשויכת ל-Tenant, היישום של המתקפה כבר מיושם ב-AADINTERNALS: Stealing and faking Azure AD device identities

2. נדרשת זהות EntraID בעלת הרשאות ניהול מקומיות בתחנת היעד.

איך עובד מערך הרשאות לתחנות קצה ושרתים בסביבת Azure?

בראש ובראשונה, קיים Role ניהולי, המקנה הרשאות ניהול מקומיות על כלל התחנות והשרתים המצורפים ל-Azure (Entra Joined).

צפייה בשם ותיאור ה-Role מבהירים באופן ברור את התכלית שלו:



משתמש בעל ה-Role הנ"ל, יהיה בעל הרשאת ניהול מקומית **בכל התחנות והשרתים** המצורפים ל-Entra.

הערה: אם יש שימוש ב-PIM, רק כאשר המשתמש יבצע הפעלה (Activate) ל-Role, ולמשך הזמן המוגדר, הרשאות אלו יהיו בתוקף.

בנוסף, ניתן להגדיר משתמשי Entra בתוך קבוצות מקומיות:

```
C:\Windows\system32>net localgroup administrators azuread\localadmin@benzamir1234outlook.onmicrosoft.com /add
The command completed successfully.


C:\Windows\system32>net localgroup administrators
Alias name     administrators
Comment        Administrators have complete and unrestricted access to the computer/domain

Members

-------------------------------------------------------------------------------
Administrator
AzureAD\localadminuser
User
The command completed successfully.
```

התייחסות למשתמש Entra יהיה או על ידי שימוש ב-UPN המלא, לדוגמא User@tenant.com או על ידי שימוש בקידומת AzureAD, לדוגמא: AzureAD\User.

אז יש לנו אחיזה בתחנה בתוך ה-Tenant, ובמשתמש בעל הרשאות ניהול מקומיות בתחנת היעד, אשר גם היא משויכת לאותו ה-Tenant, מה עכשיו?

כאשר ננסה לבצע חיבור בפרוטוקול כגון SMB, יתרחש התהליך הבא:

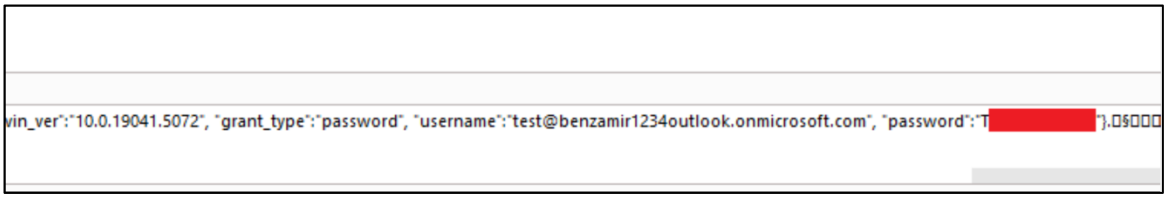
1. כמו שאמרנו, הלקוח (מזכיר, Initiator ב-NegoEx) פונה לשרת היעד, אשר מציג את שיטות ההתחברות הנתמכות:

NegoEx → PKU2U → P2P Certificate

2. אם ללקוח אין תעודה בתוקף, הוא פונה ל-Entra להזדהות וקבלת התעודה.
3. אם המשתמש הפעיל מחובר כבר עם זהות Entra ID, התחנה תשתמש אוטומטית ב-PRT, וה-SessionKey לצורך אימות.
4. אם ל-EntraID Identity אין הרשאות מתאימות, יופיע חלון הזדהות הדורש שם משתמש וסיסמה (כמו בדומיין), מה שיוביל לאימות ישיר מול Entra ID תוך שימוש בשם משתמש וסיסמה.

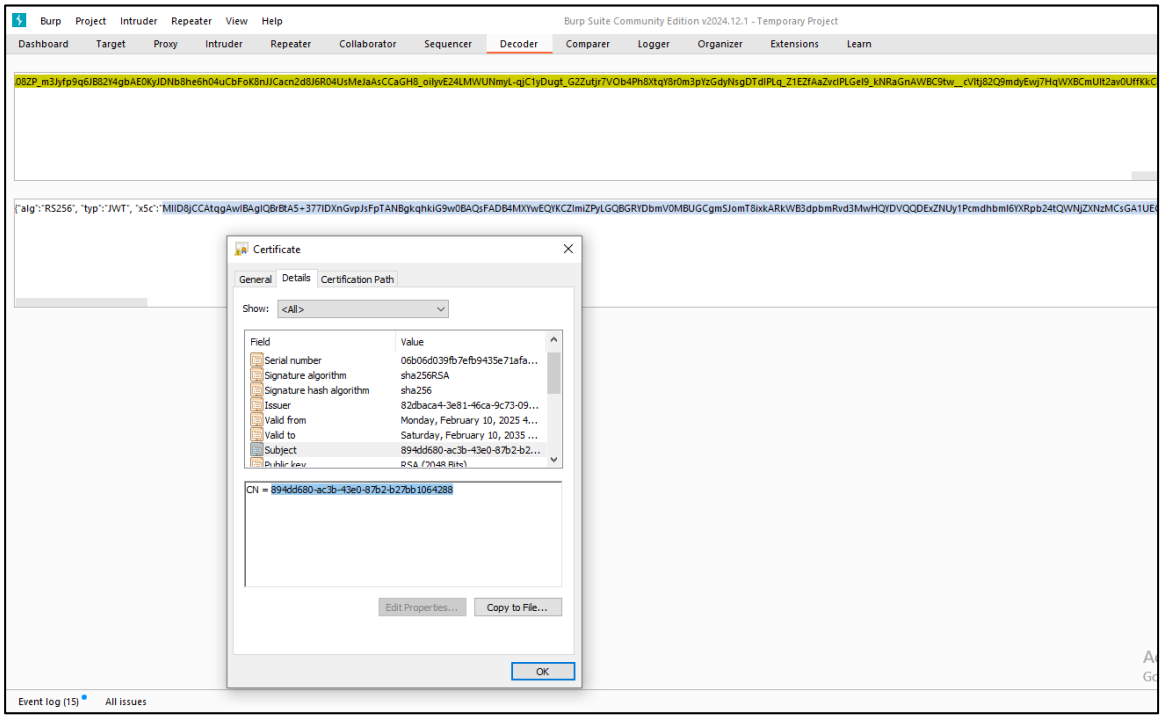
בפשטות, תהליך ההזדהות עובד בצורה הבאה:

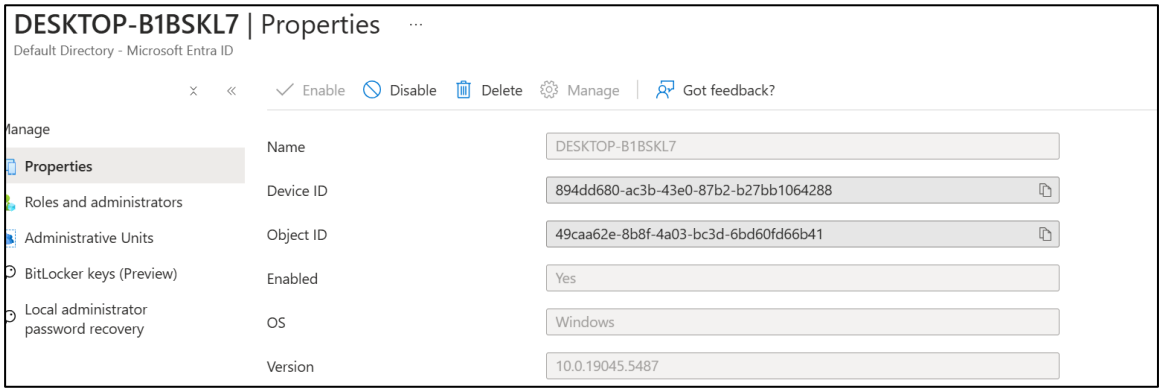
א. מתבצעת הזדהות מול Entra תוך שימוש בשם משתמש וסיסמה (במידה ומתבצעת הזדהות בתצורה של RunAs) או על ידי שימוש ב-PRT (אם המשתמש מחובר לתחנה וקיים PRT בתוקף).
דוגמא להזדהות עם שם משתמש וסיסמה:



בנוסף, נשלח המפתח הציבורי של תעודת המכונה כ-Base64.

ה-Subject הוא ה-Object ID של ה-Device:

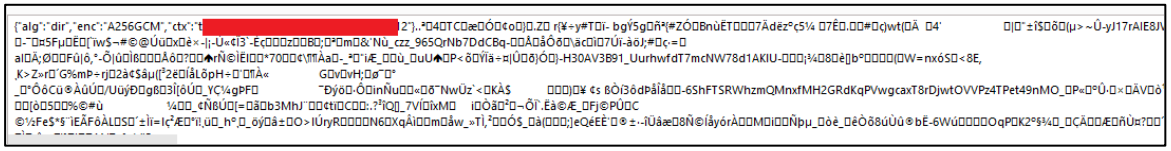




מתקבל מפתח הצפנה של JWE, אשר מוצפן בעצמו, תוך שימוש בתעודה של ה-Device, כדי להוכיח שייכות לסביבה.

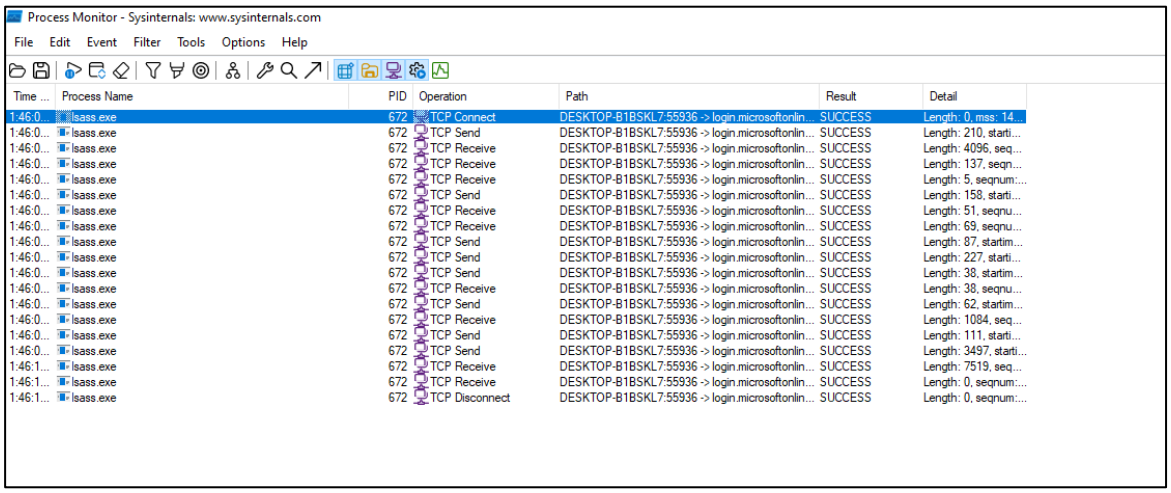
בנוסף, רק ה-Device עצמו יכול לפענח אותו ולהשתמש בו:

5. לאחר האימות, הלקוח יקבל תעודת P2P חתומה ע"י Azure. חשוב לציין שבוצע שינוי בתהליך קבלת התעודה מאז המאמר של מור. בדיקה של התהליך נכון לעכשיו, הראתה שהשינוי שבוצע במעבר והאכיפה לשימוש ב-KDF בגרסא 2 שינה גם את אופן קבלת התעודה. כעת, התעודה נשלחת מוצפנת:



מור כתב ביחד עם המחקר שעשה מספר כלים. אחד מהם, שימש לקבלת התעודה מתוך תחנה לא משוייכת תוך שימוש ב-PRT ו-Session Key. השינוי בתהליך "שבר" את הכלי שכעת לא פועל. בהמשך נסקור חלופות ופתרונות אפקטיביים.

הבקשה של התעודה, נעשית על ידי תהליך ה-LSASS:

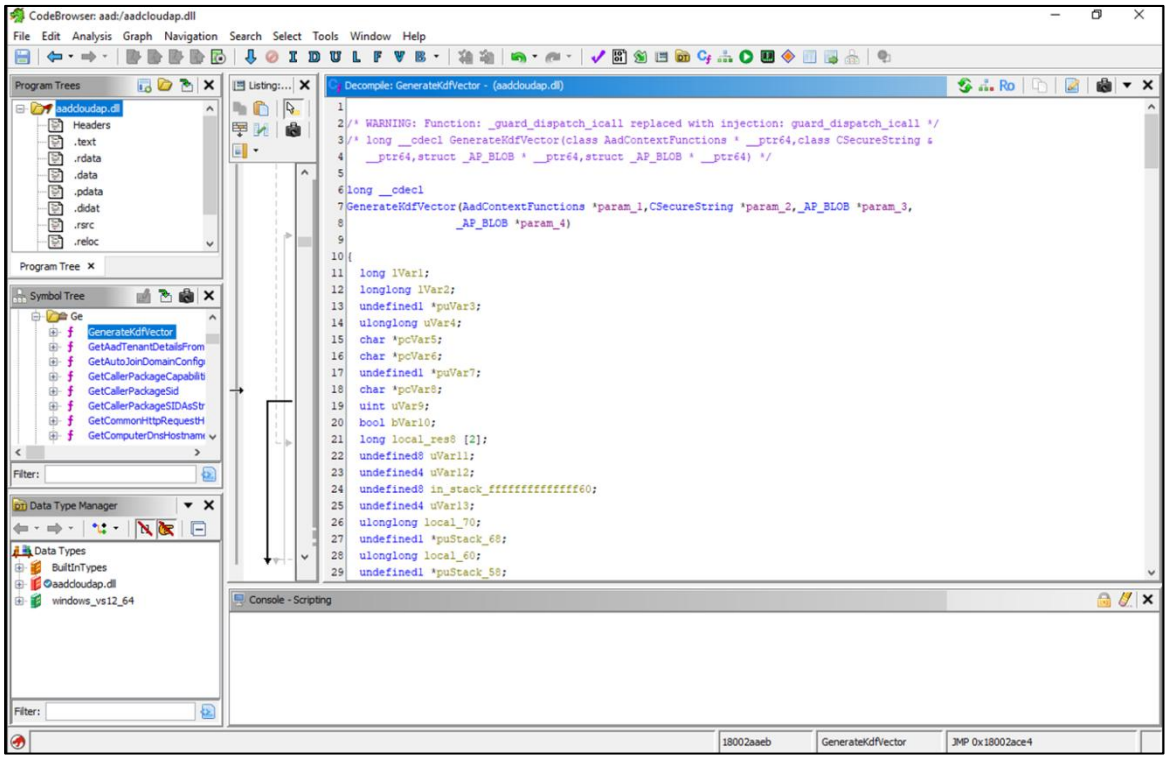


אשר טוען ומשתמש במודול AADCLOUDAP.DLL למטרת הזדהות בסביבה ארגונית מבוססת Entra.

בתוך המודול, קיימת הפונקציה DeviceP2PCertificateReqeuest, אשר אחראית לביצוע תהליך בקשת התעודה:

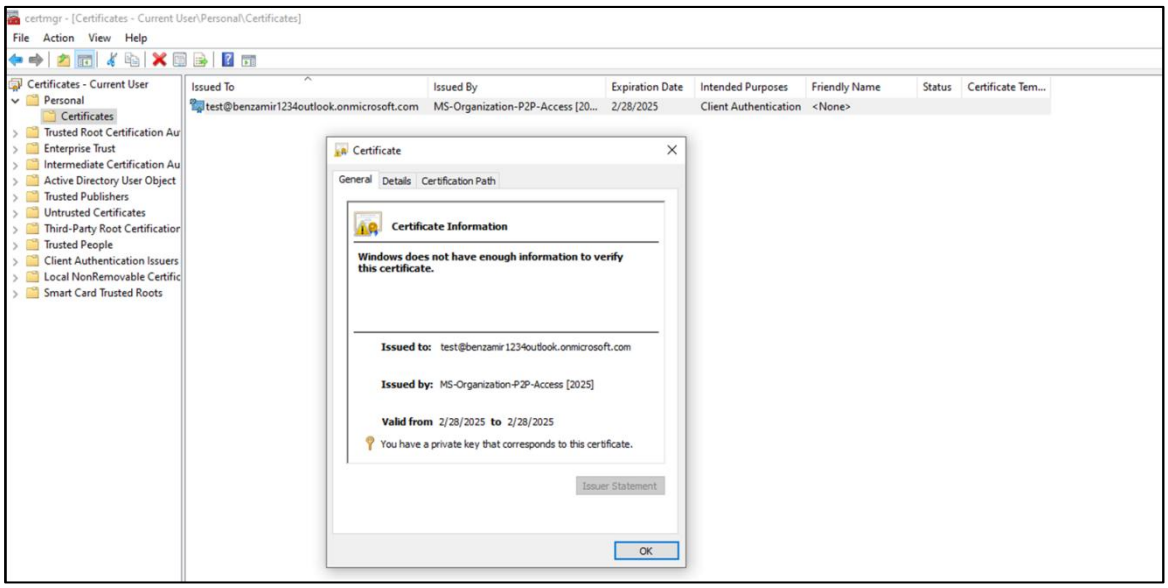
```
DeviceP2PCertificateRequest::DeviceP2PCertificateRequest(
```

במהלכה, הפונקציה GenerateKdfVector מבצעת את החתימה על ההודעות ב-KDFv2:



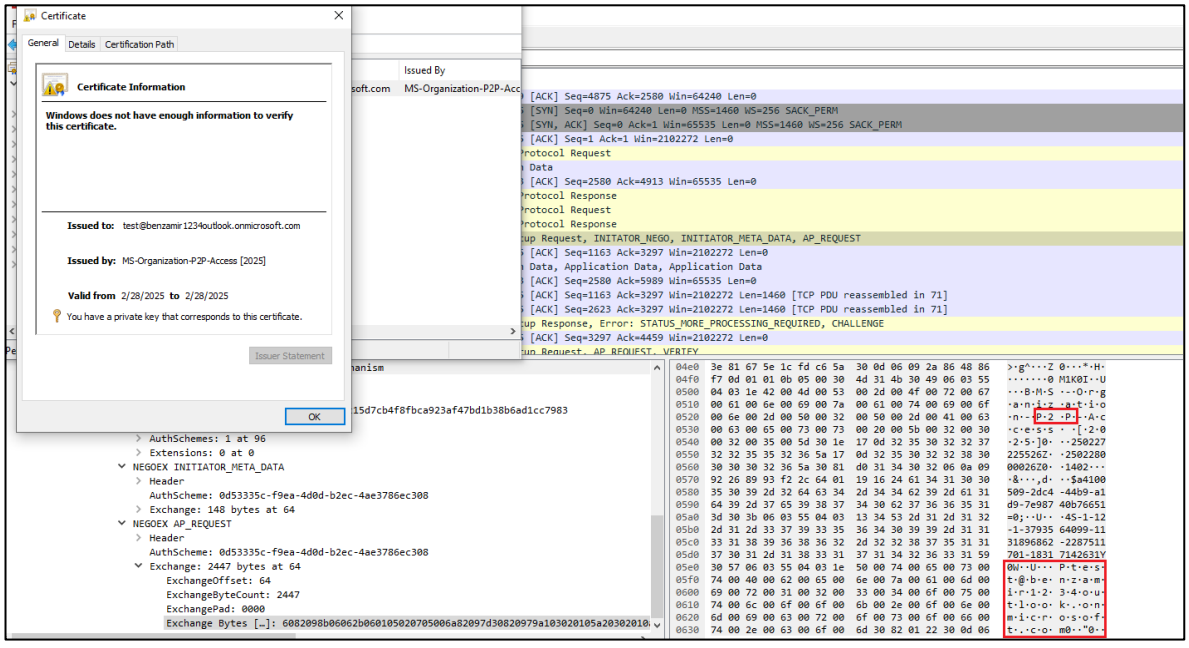

```
LAB_18002a996:
  if (bVar10) {
    pcVar8 = pcVar6;
  }
  local_48 = "GenerateKdfVector";
  local_40 = local_res8;
  local_38 = 0;
  uVar12 = 1;
  local_50 = pcVar8;
  WPPTraceLogA(4,0,"%s:%s enter",pcVar8,"GenerateKdfVector");
  if (((param_1 == (AadContextFunctions *)0x0) || (param_3 == (_AP_BLOB *)0x0)) ||
     (param_4 == (_AP_BLOB *)0x0)) {
    local_res8[0] = -0x7ff8ffa9;
    do {
      pcVar6 = pcVar5;
      pcVar5 = pcVar6 + -1;
      bVar10 = *pcVar5 == '\\';
      if (bVar10) goto LAB_18002ac99;
    } while ("onecoreuap\\ds\\ext\\aad\\aadcloudap\\login.cpp" < pcVar5);
    bVar10 = *pcVar5 == '\\';
LAB_18002ac99:
    if (bVar10) {
```

התעודה תתקבל מוצפנת תוך שימוש ב-JWE. אם המשתמש בעל session חיבור פעיל תוך שימוש בהזדהות העננית, התעודה תאוחסן ב-User Certificate Store, ויעשה בה שימוש חוזר ללא צורך בהזדהות מחודשת מול-Azure:



הערה: בשום שלב לא מתבצעת דרישה להזדהות כפולה. גם אם נעשה שימוש בשם משתמש וסיסמה (למשל תוך שימוש ב-RunAs) וגם אם התהליך בוצע תוך שימוש ב-PRT אשר לא מכיל MFA Claim, הבקשה לתעודת P2P תצליח והשרת יקבל את החיבור.

בשלב זה, הלקוח ישתמש בתעודה בתצורה שדיברנו עליה (NegoEx PKU2U), והשרת יאמת את מהימנותה:



6. אם למשתמש יש הרשאה מתאימה ,תהליך ההזדהות יושלם, והתקשורת בפרוטוקול המבוקש תתחיל.

לפי זה יוצא שאנחנו מוגבלים לביצוע תנועה רוחבית רק מתחנות ארגוניות. אז זהו, שלא!

מור כתב שני כלים נוספים (מצורפים קישורים בסוף המאמר):

Pass The Certificate בסביבת Azure (פייתון). הכלי מקבל כאגומנט תעודת P2P וכתובת יעד, משתמש בסיפריות מוכרות כגון Impacket ו-Minikerberos ובנוסף יישום של מור להזדהות ב-NegoEx PKU2U. הכלי מבצע הזדהות ומשתמש ב-Remcomsvc כמו impacket-psexec.

במילים אחרות, אם יש ברשותנו תעודת P2P של משתמש בעל הרשאות גבוהות, למשל משתמש בעל Entra Joined Device Administrator role פעיל, אפשר לנצל את התעודה גם תוך שימוש בתחנה לא ארגונית ולא משוייכת ולהשיג הרשאת הרצת קוד על כלל התחנות והשרתים בסביבה.
אז איך משיגים את התעודה? שימוש בכלים כמו SharpDPAPI המאפשרים חילוץ תעודות מהתחנה יאפשרו להשיג את התעודה ולהשתמש בה מחוץ לתחנה.

אבל התעודה תקפה רק שעה, מה שהופך את חלון ההזדמנויות לקטן מאוד. אלא אם כן, נאלץ את התחנה לבקש תעודה!

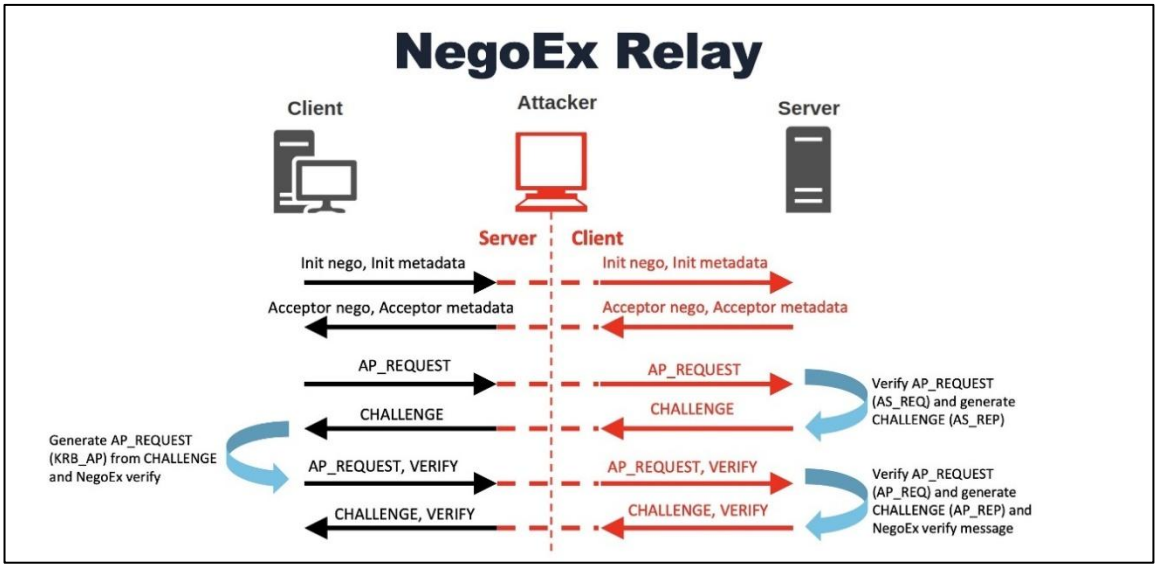
הרי אמרנו שבמידה ומדובר בסשן פעיל של זהות Entra, הבקשה של התעודה נעשית מאחורי הקלעים ברגע שהמשתמש מנסה לבצע חיבור בפרוטוקול כגון SMB לתחנה אחרת. אפשר לנצל הרשאות ניהול מקומיות במספר דרכים. לדוגמא, הזרקה של Shellcode אשר יגרום ל-Process לגיטימי במערכת שרץ תחת ה-Context של המשתמש, לבצע פקודת Dir לכתובת של תחנה אחרת בסביבה. בדרך זו תוקף יוכל לאלץ את השרת לבקש תעודה, לגנוב אותה ולהשתמש בה להשגת גישה לשרתים ותחנות נוספים.

השימוש ב-remcomsvc בכלי של מור אשר חתום כזדוני, גורר חסימה כמעט אוטומטית על ידי רוב מוצרי אבטחת המידע. עם זאת, החלק הקריטי בתהליך- הזדהות המשתמש באמצעות NegoEx PKU2U, כבר ממומש באופן מלא ונותר שמיש. גמישות המימוש מאפשרת לבצע שינויים ולהתאים את הכלי למגוון רחב של סיטואציות.

לדוגמה, ביצוע Fork לכלי של מור, ושינוי קטן בקוד, אפשר להתאים את פעולתו לשימוש בסימולציה ספציפית במהלך מבדק חוסן, בסביבה בעלת תחנות המצורפות ל-Entra בלבד. בסימולציה זו, בוצעה תנועה רוחבית בין תחנות בארגון, תוך שימוש בזהויות עננית, במטרה להסלים הרשאות בסביבת Azure. כך נבחנו יכולות הניטור והתגובה של מערכות האבטחה הקיימות. למי שמעוניין, העלתי את התבנית לGitHub:-

```
C:\Users\test\AzureADJoinedMachinePTC-master>python.exe  Main.py --usercert p3p.pfx --certpass Aa123456 -
-remoteip 192.168.47.193 --command "net user test1233 /del"
StringBinding ncacn_np:192.168.47.193[\pipe\svcctl]
Creating service testsvc on the remote machine...
Starting service testsvc...
Got expected ERROR_SERVICE_REQUEST_TIMEOUT, Command ran successfully!
Stopping and deleting service testsvc...
Service seems already stopped, deleting
Service stopped and deleted.
```

NegoEx Relay - כלי המבצע פעולה דומה מאוד ל-Ntlmrelayx, ומאפשר שידור הזדהות מול שירות SMB אשר לא אוכף Signing, ובכך פגיע למתקפת Relay. התרשים מהמאמר של מור מסביר את התהליך באופן כללי:



סיכום

במאמר זה בחנו וקטורי תקיפה אפשריים לביצוע תנועה רוחבית (Lateral Movement) בסביבה ארגונית, שבה התחנות מנוהלות באמצעות Entra בלבד. הדגשנו כיצד ניתן לנצל הרשאות ניהול מקומיות ללא צורך באימות רב-שלבי (MFA) ב-Machines המחוברים ל-Entra, מה שמעניק לתוקפים אפשרות לגשת למגוון תחנות ושרתים בסביבה הארגונית.

אחת מנקודות התורפה המשמעותיות, היא תעודת ה-P2P, אשר בתוקף למשך כשעה, ומספקת גישה רחבה לכל המשאבים בסביבה תחת הרשאות המשתמש. תכונה זו הופכת את התעודה ליעד נחשק במיוחד עבור תוקפים, שכן היא מאפשרת לבצע פעולות רבות ללא צורך באימות נוסף.

עוד ציינו, כי במידה והתחנה מבקשת את התעודה כחלק מסשן של משתמש פעיל, התעודה נשמרת ב-Store של המשתמש. אופציה יעילה לחמוק ממנגנוני זיהוי מודרניים, שמנטרים פעולות חשודות מול תהליכים רגישים כגון LSASS, היא לגרום לתחנה לבצע פעולה תוך שימוש בזדהות המשתמש הרצוי, שתגרור בקשה לגיטימית לקבלת תעודת P2P.

כאשר למשתמש הפעיל קיימת תעודת P2P בתוקף בזמן ניסיון הזדהות, המערכת תשתמש בה אוטומטית, ללא שליחה מחודשת של בקשת תעודה ל-Azure. גניבת התעודה תוך שימוש ב-session של המשתמש יכולה לאפשר גניבת זהות והרשאות ניהול מקומיות בצורה שקטנה יחסית.

על המחבר

בן זמיר, בודק חוסן וחבר בצוות אדום. עוסק בעיקר במחקר וביצוע מתקפות בסביבות ענן (GCP, Azure ו-AWS) ובסביבות היברידיות. ליצירת קשר, אפשר לפנות אלי באימייל או דרך לינקדאין

מקורות מידע

- https://medium.com/@mor2464/azure-ad-pass-the-certificate-d0c5de624597
- https://i.blackhat.com/USA-22/Wednesday/US-22-Rubin-AAD-Joined-Machines-New-Lateral-Movement.pdf
- https://aadinternals.com/post/deviceidentity/
- https://github.com/morRubin/NegoExRelay
- https://github.com/morRubin/AzureADJoinedMachinePTC