Nmap - זמן סיפור / או למה לקרוא תוצאות זה לא מספיק

מאת יובל (tsif) נתיב ואפיק (cp77fk4r) קסטיאל

רקע

הנושא הספציפי הזה מעיק עליי כבר שנים ונראה לי שהגיע הזמן 'לסגור את הפינה'. הרבה נכתב על nmap והרבה מדריכים קמו והלכו. לדעתי בגלל שאנשים לא מבינים את כמות היכולות של nmap ומסתכלים עליו כאל כלי איסוף ולא כלי תקיפה מובהק אנשים מעדיפים להתמקד בכלים כמו metasploit ולא מספיק ב-nmap ולמרות שהכלי הזה נמצא בבסיסה של כמעט כל עבודת Pentest, Security Review, מחקר ועבודת אבטחת מידע כזאת או אחרת. במאמר הזה אנסה לכסות גם את עקרונות היסוד של הכלי וגם שימושים קצת יותר מתקדמים.

> אזהרה לקורא: לא מדובר בעוד Cheat Sheet או מדריך עם פקודות מוכנות. אנסה להתעמק דווקא בתעבורה שרצה מאחורי הכלי ומה המשמעות שלה לסריקה שלנו. כיצד סריקות שונות יניבו תוצאות שונות ומדוע. ידע נדרש להפקת המירב מהמאמר: הבנה של TCP, IP ו-UDP.

שיטות חיבור

ל-nmap כמה שיטות חיבור עיקריות הנקראות Scan Types בשפת הכלי. הוראות אלו יורו לכלי כיצד עליו להתחבר לפורטים מסויימים. נתחיל מלהסתכל על שתי שיטות הסריקה העיקריות של nmap ולעמוד באמת את ההבדלים ביניהן:

TCP Connect

שיטה זאת מסומלת על ידי הארגומנט sT- .על השיטה הזאת ניתן לחשוב כעל השיטה הסטנדרטית שבה אנו חושבים על סריקת פורטים. כאשר אנחנו מדברים על סריקת פורטים של TCP מה שאנו מעוניינים לדעת זה איזה שירותים מציעה מכונה מסויימת. במובן הזה ננסה לראות על איזה פורטים מאזינה איזה מכונה. בתצורה הקלאסית של חיבור TCP רגיל אנחנו בעצם מבצעים שתי שליחות וקבלה אחת. הבקשה הראשונה תהיה בקשת SYN. ה"בקשה" הזאת בעצם רק אומרת שלקחנו את חבילת ה-TCP שייצרנו והדלקנו את הביט שאומר שהבקשה הזאת היא בקשת SYN. במידה ובצד השני יש תוכנה שמאזינה

לבקשות בפורט המדובר היא תשלח חזרה חבילה ותדליק את הביט שאומר SYN\ACK או בעברית: יש כאן מישהו שמוכן לדבר איתך. לאחר קבלת ההודעה אנו נשלח חזרה חבילת ACK (שוב, רק ביט ב-header של החבילה) ולאחר מכן נוכל להתחיל לדבר ב-TCP. זהו בעצם תהליך ה-Three Way Handshake המפורסם שאותו מבצע כל חיבור TCP כדי להתחיל לדבר.

בשיטת סריקת הפורטים TCP Connect אנחנו מבצעים התחברות מלאה כזאת אל כל אחד מהפורטים שאנחנו סורקים. אלה שלא שולחים אלינו חזרה SYN\ACK אנחנו יכולים להניח ש"מתים" ושאין שם מאזינים.

TCP SYN

שיטה זאת מסומלת על ידי הארגומנט sS-. השיטה הזאת הפכה להיות כבר שיטה מאוד סטנדרטית ובעבר נקראה אף שיטת ה-stealth (חשאיות) אך היום הדבר נכון חלקית בלבד. שיטה זאת משתמשת בלחיצת ידיים חלקית ממה שדובר בשיטה הקודמת. כאן, אנו נשלח חבילת SYN אל היעד. שם אם יש תוכנה המאזינה נקבל חזרה SYN\ACK ואם לא אז יהיה 'שקט'. אך את התגובה ACK לא נשלח חזרה. זה נשמע דיי בנאלי אך בפועל המשמעות של הדבר היא שלא נפתח חיבור TCP באמת. אנו נדע אם יש גורם המאזין בצד השני לבקשות ונראה את התגובה אך לא נשלים את לחיצת הידיים.

הרבה מאוד חומות-אש נפלו ועדיין נופלות בסריקה הזאת. הסיבה היא שחומת אש היא ייצור דיי מטומטם (ויעיל!) בבסיסו: היא מונחית חוקים. אם הוגדר חוק היא תעקוב אחריו. אך הם החוק שהוגדר אומר לדוגמא שבמידה ונפתחו חיבורי TCP אל יותר מ-3 פורטים מדובר בסריקה ולכן יש לחסום את התעבורה מאותה כתובת IP הרי שהסריקה הזאת לא תיכנס לחוק הזה. אמנם אנחנו יודעים מי ענה ומי לא, אבל עם אף אחד מהם לא הקמנו חיבור TCP פעיל ולכן הפעולות שעשינו הן לא "matching" לקריטוריונים שהוגדרו בחומת האש. בהמשך נראה איך בשילוב עם סריקת SYN ועוד אפשרויות של NMAP אפשר להגיע לסריקות שהן כמעט בלתי ניתנות ללכידה גם על ידי IPS-ים מודרניים.

FIN, NULL, XMAS המבלבלות

עכשיו שעברנו על הבסיסיות נרד קצת יותר נמוך (עדיין מרחפים מגבוה) ונאחד כמה סריקות שונות ונדבר עליהן כעל סריקה אחת (לאחר שנבין מה המשמעות). סריקת ה-FIN תדליק רק את הביט שאומר סיום התקשורת (FIN). סריקת NULL תשלח את החבילות כאשר כל הביטים המעידים על סוג החבילה מאופסים וסריקת XMAS תדליק את הביטים FIN, URG ו-PUSH. עכשיו שסיכמנו את זה והבנו טכנית מה קורה בואו נבין גם למה. למען הסדר הטוב: FIN = -sF, NULL = -sN, XMAS = -sX.

היום אנחנו מתחילים לראות המון חומות אש ו-IDS-ים מפוזרים כמעט מאחורי כל ארגון שאנחנו מסתכלים עליו מבחוץ. הסוד כאן טמון בהבנת הניאנסים הקטנים של TCP\IP לפי יישומים (אימפלמנטציות) מסויימות. קודם כל, לאחר השימוש הנרחב בסריקות SYN, מערכות IDS "מחפשות" חבילות SYN נכנסות ושלושת סוגי הסריקות האלה אמורות להיות סריקות "משלימות". הרעיון מאחורי סריקות אלה הוא שמערכות אשר מקבלות בקשות TCP לשירותים סגורים אמורות להגיב בחבילת RST בעוד ששירותים פתוחים אשר מקבלות בקשות כאלה (ללא יצירת חיבור ראשוני) מגיבות בזריקת החבילה ולא מגיבות כלל. (מערכות אשר מכבדות (או מיישמות) את TCP\IP לפי RFC 793).

כאן נחמד להזכיר שמערכות ההפעלה Windows אינן מיישמות פרוטוקול TCP\IP לפי RFC793 מה שמאפשר לנו לקבל אינדיקציה על מערכות ההפעלה האלה. במידה וביצעתי סריקת SYN ונראה שיש שירותים פתוחים אך סריקת FIN\NULL\XMAS לא מראות שירותים פתוחים כנראה שמדובר במערכת Windows. שימו לב למילה 'כנראה' מכיוון שיש מערכות נוספות שאינן מיישמות TCP\IP לפי RFC793.

בנוסף חשוב להדגיש שתוצאות אלה אינן יודעות להבדיל בין פורט שהוא filtered לפורט שהוא open ועל זה נרחיב בהמשך.

סריקת UDP

סריקת פורטים ב-UDP הינה סריקה חשובה ביותר שכמות גדולה מידי של בודקים מקפידה בעקביות להתעלם ממנה. כמו פורטים של TCP, פורטים של UDP מספקים לנו הבנה נוספת לגבי המערכת. בחלק מהמקרים בגלל האופי של UDP (stateless) מספק לנו יכולות תקיפה מעניינות או זיהוי של כלים (כגון טרוייאנים) שמשתמשים ב-UDP. בנוסף קיימים שירותים אשר יכולים להיות נקודות קריטיות בבדיקה כמו SNMP אשר מסתמך על פורט 161 ב-UDP. עכשיו לנושא עצמו.

אם UDP הינו stateless, כיצד אני יכול לדעת אם הוא פתוח או לא? ובכן, באמת ב-UDP עצמו אין לנו אינדיקציה אם הפורט פתוח או לא, אך מערכות ההפעלה עוזרות לנו בכך שהן עונות לנו ברמה מסויימת. כאשר אנחנו סורקים עם הארגומנט sU-, אנחנו נשלח הודעות UDP בגודל 0 בתים לפורטים. במידה והפורט פתוח אנחנו יכולים לצפות לחוסר תגובה. במידה והפורט סגור מערכת ההפעלה תגיב ותכתוב לנו הודעת ICMP Port Unreachable חזרה.

כאן יש לנו שלושה נושאים להזכיר: הראשון - כאשר חומת אש תחסום הודעות ICMP Port Unreachable אנחנו יכולים לצפות מ-nmap להחזיר לנו תוצאות false-positives. נושא שני - רב מערכות ההפעלה מגבילות את כמות ההודעות הללו שהן מוציאות לפרקי זמן קצובים מה שאומר שאנחנו יכולים לצפות לאיטיות רבה בסריקה מכיוון ש-nmap יתאים את הסריקה לתגובות. נושא שלישי - מערכת ההפעלה חלונות איננה מגבילה את התגובות הללו ולכן נוכל למפות אותה בתחום ה-UDP בקצב מהיר יחסית.

סריקת פרוטוקולי IP

בסוג סריקה זה (-sO) אנחנו מנסים להבין באיזה פרוטוקולים מסט פרוטוקולי ה-IP תומכת המכונה. כדי להשיג את זה תשלח המכונה שלנו הודעות "ריקות" בכל סוגי הפרוטוקולים כדי לראות את התגובה. במידה והתגובה תהיה הודעת ICMP Protocol Unreachable אנחנו נדע שפרוטוקול מסוים איננו נתמך. יש לשים לב שהרבה חומות אש אינן מעבירות הודעות אלה והתוצאה עלולה להיות false-positive שוב.

דוגמא לתוצאת סריקה כזאת על המכונה שלי:

```
tisf ~ > sudo nmap -sO 127.0.0.1
[sudo] password for tisf:
Starting Nmap 6.46 ( http://nmap.org ) at 2014-06-24 19:43 IDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000011s latency).
Not shown: 249 closed protocols
PROTOCOL STATE SERVICE
1 open icmp
2 open|filtered igmp
6 open tcp
17 open udp
103 open|filtered pim
136 open|filtered udplite
255 open|filtered unknown
Nmap done: 1 IP address (1 host up) scanned in 1.25 seconds
```

סריקת ACK

סריקת ACK תשמש אותנו בעיקר להבנה של חומת האש מולה אנחנו מתמודדים ו\או למפות את החוקים שבחומת האש. כמו שכבר אתם מבינים, סריקת ACK אומרת שהחבילות שישלחו ידליקו את ביט ה-ACK. הטריק הוא דיי פשוט, במידה ורכיב מקבל הודעת ACK ישירות, ללא הקמת חיבור TCP לפני, אותו רכיב צריך להגיב בחבילת RST. התוצאות תלויות בלקבל את אותה תגובת RST או לא. שימו לב שתוצאות סריקת ACK לעולם לא יחזירו האם הפורט פתוח או סגור אלא את התגובה filtered או unfiltered.

במידה וחומת האש ערנית לתקשורת או פשוטה (stateful inspection \ stateless inspection) אנו נראה את התגובות. במידה ולא קיבלנו חזרה את חבילת ה-RST אזי משהו חסם אותה ולכן נבין שחומת האש היא stateful. מכיוון שידעה לחסום תקשורת לא מורשית לעומת אם החיבור היה חוזר, אזי היינו יודעים שדבר לא עצר אותה ולכן כל עוד החיבור הגיע מבפנים ונראה תקין, חומת האש אינה מודעת לדבר, ולכן תאשר את העברת החבילה.

מידע משלים

למקרה שזה עוד לא היה ברור, ל-nmap יש עוד אפשרויות רבות. אנסה להביא כאן עוד כמה דוגמאות לדרכים להשתמש באותם ארגומנטים שרובכם השתמשתם בהם כבר אך בשאיפה לספק קצת ערך מוסף לסריקות העתידיות.

תזמון

תזמון הוא נושא קריטי (בכללי ו) במיוחד בעת סריקות. הנושא מורכב מרכיבים רבים. כך לדוגמא כמו החיבורים המקביליים שמותר ל-nmap להחזיק כלפי המכונה הנסרקת, כמות הרכיבים ש-nmap ידבר איתם במקביל, המרווח בין חבילה לחבילה ועוד. כדי להקל עלינו, במקום שנתחיל לשחק עם כל הגדרה והגדרה nmap מכילה תבניות סריקה. התבניות האלה מסומנות כ-T0-5-. כאשר אנחנו יכולים לבחור את הרמה. אפילו יש לרמות אלה שם: מצב פראנויה, חמקמק, מנומס, אגרסיבי, מטורף (בסדר עולה מ-0 עד 5).

כמובן שכולנו היינו רוצים להשתמש אך ורק במצב פראנויה אך מצב זה אומר ש-nmap תחכה כ-5 דקות בין הודעה להודעה. חישוב מהיר אומר לנו שסריקה של רכיב אחד עם 65,000 פורטים תיקח כ-255 ימים. סריקה של T5 אמנם קיימת אך אינני ממליץ עליה כלל. אתם עלולים לאבד כמויות מידע גדולות מידע, לקבל יותר מידי false-positives וצריכים להיות עם חיבור מאוד מהיר את המכונה הנסרקת. סריקה במצב T0 מאפשרת "התחמקות מגילוי". הביטוי מופיע במרכאות מכיוון שזוהי סריקה רגילה אך מתפרסת על יותר זמן. כנראה שסריקה זאת תתפרש על פני לוגים רבים וכל חומת אש או IDS יתייחסו אליה כאל רעש רקע זניח.

במהלך הסריקות היומיים אני ממליץ להתחיל מכיוון ה-T2 לסריקות חיצוניות אשר מוקצה זמן רב יותר ועד ל-T4 למכונות אשר איתכם באותו הסגמנט בבדיקות פנימיות.

גילוי שירותים

גילוי שירותים הוא תהליך מאוד רועש שעלול לעשות הרבה מאוד "בלאגן" ברשת. יש לשים לב שגם כאשר נעשה סריקות "רגילות" כמו סריקות SYN נראה תוצאות שאומרות לנו שפורט מסויים משוייך לשירות מסוים. כך לדוגמא אם פורט 80 יהיה פתוח nmap יכתוב לנו ליד http. דבר זה אינו אומר שבהכרח יש שירות WEB על פורט 80 אלא ש-nmap זיהה שהוא פתוח והשלים אותו מרשימה מסודרת מראש שבה פורט 80 משויך לשרת web. יכול להיות שעל הפורט הזה פתוחים שירותים אחרים בכלל.

כאשר נדליק את הדגל sV- אנו נורה ל-nmap לגשת את הפורטים הפתוחים שהוא מצא והפעם הוא לא יסתפק בלזהות שהפורט פתוח. הוא יבצע חיבור מלא וינסה להבין באמת איזה שירות עומד מאחורי

הפורט. זה יתחיל מניסיון לבצע "Banner Grabbing" שבוא הוא "יבקש" מהשירות לשלוח אליו את סוג הגרסה או לנסות למצוא את זה באחת ההוראות החוזרות. לאחר מכן, במידה ולא הצליח, ינסה nmap לברר באיזה סוג שירות מדובר. תחילה ישלח בקשות לפי סוג הפורט. כך לדוגמא אם מדובר על פורט 80 nmap ינסה לשלוח בקשה לפי פרוטוקול HTTP ולראות איזה תגובות יקבל. במידה ולא יקבל את התשובה שציפה לה, יתחיל nmap לשלוח בקשות בפרוטוקולים אחרים עד אשר יצליח לזהות.

כמה דגשים לגבי הדגל sV. קודם כל, במידה ו-nmap לא זיהה את סוג הפרוטוקול אנו נראה סימן שאלה קטן לידו. כך לדוגמא במקרה הבא נראה סריקה שבה nmap לא הצליח לזהות את השירות בפורט 23 ולכן כתב לנו שלפי מספר הפורט הוא אמור להיות מוקצה לשירות מסוג telnet אך בעת תקשורת עם הפורט לא הגיב כפי ששירות telnet אמור להגיב.

```
tisf ~/ > sudo nmap -sV 192.168.6.254

Starting Nmap 6.46 ( http://nmap.org ) at 2014-07-06 20:44 IDT
Nmap scan report for 192.168.6.254
Host is up (0.041s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
23/tcp open  telnet?
80/tcp open  http  Cisco IOS http config
MAC Address: 00:11:5C:B9:50:00 (Cisco Systems)
Service Info: OS: IOS; Device: router; CPE: cpe:/o:cisco:ios

Service detection performed. Please report any incorrect results at
http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.15 seconds
```

דגש נוסף הינו שילוב סריקות. סריקת sV חייבת להשלים את ה-TCP handshake. מה שאומר שבמידה והדלקנו את הדגל sV ו-sS הוא יהיה דיי חסר משמעות מכיוון שבכל מקרה נשלים את כל לחיצות הידיים.

זיהוי מערכת הפעלה (O-)

אחת מתבניות הפעולה הכי פחות מובנות ועדיין הכי נפוצות בסריקת NMAP. לפני שנכנס לאופן הפעולה בואו ננסה להבין קודם את התוצר של הסריקה המדוברת. כאשר נוסיף את הדגל O- נראה כמה שדות נוספים אשר מתווספים לפלט הסריקה. שדה ראשון יש את סוג המכשיר (Device Type). בסוג זה נוכל לראות את סוג המכשיר הכללי (הרבה פעמים לא מדוייק) כגון: router, firewall, server, embedded וכו'. שדה נוסף (היותר מדוייק) יהיה שדה ה"מריץ" (Running). בשדה זה NMAP יציג לנו את בסיס מערכת ההפעלה שהתגלה מהסריקה. לא מדובר על מערכת הפעלה מדוייקת, אך NMAP ינסה להתאים לז'אנר של קרנל לדוגמא. מה שאומר שלא יהיה לנו הבדל לדוגמא בין Windows Vista, Windows Server 2008+

ו-Windows7. מבחינת NMAP הוא יתייחס לאופן התפקוד הבסיסי של הקרנל של מערכת ההפעלה עם ה-TCP\IP Stack שלה.

השדות OS CPE ו-OS Details ינסו להחזיר תוצאה מדוייקת יותר. אני ממליץ להזהר מאוד עם התוצרים האלה. מניסיון, תוצאות אלו לא מדוייקות לרוב אך מספקות הבנה וכיוון. לדוגמא אם אנו סורקים מכונה שאנחנו יודעים שהיא בסגמנט השרתים ותוצאות הסריקה מראים לנו שמדובר על מערכת הפעלה Windows Vista כנראה שהקשת המסקנות של NMAP לא מדוייקת. יש לזכור שמעבר לרמת הקרנל קשה מאוד לזהות את סוג מערכת ההפעלה המדוייק ואנו נבנה על סריקות "משלימות" שנדבר עליהן בהמשך.

Uptime

אחת השאלות שאני מקבל המון היא כיצד nmap יודעת להעריך בדיוק טוב כל כך את זמן הריצה של המחשב בעוד דברים כמו גרסת מערכת הפעלה מדויקת זה דבר מסובך כל כך. הרבה מאוד מערכות הפעלה "מאפסות" את המונה המשמש לחותמת הזמן בתוך חבילות ה-TCP כחלק מה-Header. סריקה הינו תהליך המייצר הרבה מאוד חבילות כאלה אל היעד ומקבל גם הרבה חזרה. על ידי בדיקה מדגמית של האינקרמנטציה בין חבילה לחבילה ובניה של מדגם כזה, nmap מסוגלת להעריך בדיוק טוב מאוד את המרחק בין הזמן שהוא המונה היה 0 ועל ידי כך להעריך את זמן ה-uptime של המכונה.

בלבול האויב

סריקת פיתיונות (Decoy Scanning)

ידוע גם בשם Decoy Scanning ומסומל בארגומנט D-. הסריקה הזאת מבצעת זיוף של סריקה מכתובות נוספות. הסריקה לא תסתיר את הכתובת האמיתית שלכם אלא רק תעמיס בכתובות נוספות רבות כך שלאנשי ההגנה שיושבים מול הלוגים יהיה קשה יחסית לאתר מהיכן מגיעה הסריקה. מוזמנים להשתמש בכלי עזר שהכנתי לסריקה זאת שמחולל כתובות IP רנדומליות.

פרגמנטציה

פרגמנטציה היא איננה פונקציה של nmap אלא טכניקה של פיצול חבילות ברשת. הכוונה היא שישנן רשתות אשר גודל ה-MTU ביניהן שונות (Maximum Transfer Unit). ה-MTU מייצג את גודל החבילה המקסימלית שהרשת מוכנה להעביר. כאשר חבילה עוברת בין רשתות גודל ה-MTU עלול להשתנות בין רשת לשת בעוד גודל החבילה יקבע לפי ה-MTU ברשת ממנסה יצאה החבילה. לאחר מכן, כאשר תגיע החבילה לרשת שבה ה-MTU קטן יותר החבילה תתפצל לשתי חבילות כאשר בראשונה ידלק הביט שאומר IP Fragmentation שמעיד על כך שיש עוד חבילות בדרך. אחת הטכניקות הנחמדות יותר בסריקה

היא פיצול של חבילות לחבילות קטנות יותר. הרבה מאוד מכשירי הגנה כאלה ואחרים לא יודעים לחבר IP Fragmentation אלא רק להסתכל חבילה חבילה ולכן לא יבינו את המשמעות של החבילה.

```
nmap -f (fragment packets); --mtu (using the specified MTU)
```

החלפת פורט המקור

לכל חומת אש יש חוקי חריגה. חבילות מכתובות מסוימות או מאחרות מוגדרות כחבילות שניתן להעביר ושאין איתם בעיה. אחד החוקים שמכניסים הרבה מאוד פעמים כדי למנוע תקלות היא שחבילות שנכנסות בתקשורת HTTP הן חבילות בסדר שחוזרות מאתר מסוים. ה"כיף" שלנו זה שהרבה פעמים אנשים מתעצלים לגבי ההגדרה (או שאין יכולת במכשיר) להבין מתי משהו באמת חוזר בתקשורת HTTP והחוק שיופעל בפועל הוא חוק שאומר שהם החבילה מגיעה מפורט 80 או 443 כנראה שהיא מגיעה משירות HTTP או HTTPS וניתן להעביר אותה הלאה. הודות לזה נוכל לאמר ל-NMAP להוציא את הסריקות שלנו מפורט 80 או כל פורט אחר בעזרת:

```
nmap --source-port <portnumber>; -g <portnumber>
```

הרחבות ותוספים ל-NMAP

ב-NMAP קיים המתג "script--" ו-"script-args" אשר מטרתם הינה לאפשר להריץ הרחבות ותוספים ל-NMAP. לדוגמא, נניח וסרקנו מטרה מסויימת וגילינו כי פורט 1433 פתוח? ברב המקרים, השלב ההגיוני הבא יהיה לנסות לאתר שם משתמש וסיסמא שבעזרתם נוכל להתחבר אליו. יהיה זה פשוט לבצע את הפעולה עבור שרת או מספר בודד של שרתים, אך אם נסרוק subnet שלם - נניח את ה-subnet של השרתים בארגון, קיים סיכוי רב שנאתר לא מעט שרתים המריצים MsSQL ויענו לפורט 1433, כנראה שלבדוק שרת אחרי שרת יהיה בלתי אפשרי. ולכן, במקרים מסוג זה - נוכל להורות ל-NMAP לבצע את הבדיקה בשבילנו, בפרק זה נראה איך לעשות זאת, ועל מנת להרחיב את הדוגמא, נראה כיצד ניתן להתמודד בדרך זו עם מספר מקרים דומים.

אז, ספציפית, לגבי הדוגמא הנ"ל, נוכל להשתמש בסקריפט בשם "ms-sql-brute" שנכתב בדיוק למטרה זו, נריץ אותו באופן הבא:

```
nmap -p 1433 --script ms-sql-brute --script-args userdb=users_list.txt
passdb=pass_list.txt ip-range
```

הסקריפט ינסה, עבור כל משתמש בקובץ users_list.txt את אחת מהסיסמאות מהקובץ pass_list.txt, הסקריפט מסוגל לזהות אם חשבון מסויים ננעל (בשרתי MSSQL 2005 ומעלה יש תמיכה בנעילת

חשבונות במקרים של מספר ניסיונות התחברות כושלים). ניתן לקרוא אודות הסקריפט הספציפי בקישור הבא:

http://nmap.org/nsedoc/scripts/ms-sql-brute.html

סקריפט זהה הינו ה-ftp-brute - שמנסה לזהות שמות משתמשים וסיסמאות לשרתי ftp. סקריפטים נוספים שמבצעים מתקפות brute force על שירותים כאלה ואחרים ניתן למצוא בקישור הבא:
http://nmap.org/nsedoc/scripts/ftp-brute.html

עבר לסריקת פורטים רגילה, שאותה NMAP מסוגלת לבצע באופן יוצא מהכלל, ניתן לגלות מטרות ברשת ע"י ניצול שירותי Discovery שונים, הרעיון כאן הוא שבמקום להעיר ולזעזע את רכיבי ה-IDS וה-IPS הקיימים ברשת ע"י כך שנשלח מספר רב של חבילות ברשת. נוכל לשלוח חבילת Discovery עבור שירות מסוים כ-Broadcast (שזאת חבילה לגיטימית ברוב המקרים), ואם אכן יש שרתים אשר תומכים בשירות הספציפי הנ"ל ומאזינים לחבילות Broadcast, נוכל לזהותם.

אם בדוגמא קודמת איתרנו שרתי MsSQL, ועל מנת לעשות זאת נאלצנו לסרוק טווח שלם עבור מספר פורט ספציפי, בעזרת העובדה כי שרתי MsSQL תומכים ב-SSRP (שירות SQL Server Resolution Protocol), על מנת של-wizard-ים של מיקרוסופט תהיה היכולת לזהות בקלות ולאתר שרתים אלו, נוכל לבצע את אותה הפעולה (כאשר מדובר באותו ה-Subnet בו אנו נמצאים) באופן שקט הרבה יותר. באופן הבא, נוכל להורות ל-NMAP לשלוח חבילת CLNT_BCAST_EX ברשת, ולהאזין לחבילות החוזרות:

```
nmap --script broadcast-ms-sql-discover
```

דוגמא לפלט (במקור: http://nmap.org/nsedoc/scripts/broadcast-ms-sql-discover.html):

```
| broadcast-ms-sql-discover:
|   192.168.100.128 (WINXP)
|     [192.168.100.128\MSSQLSERVER]
|       Name: MSSQLSERVER
|       Product: Microsoft SQL Server 2000
|       TCP port: 1433
|       Named pipe: \\192.168.100.128\pipe\sql\query
|     [192.168.100.128\SQL2K5]
|       Name: SQL2K5
|       Product: Microsoft SQL Server 2005
|       Named pipe: \\192.168.100.128\pipe\MSSQL$SQL2K5\sql\query
|   192.168.100.150 (SQLSRV)
|     [192.168.100.150\PROD]
|       Name: PROD
|       Product: Microsoft SQL Server 2008
|_      Named pipe: \\192.168.100.128\pipe\sql\query
```

[מידע נוסף אודות SSRP - ניתן לקרוא בקישור הבא: http://download.microsoft.com/download/B/0/.../%5bMC-SQLR%5d.pdf]

דוגמא לשירותים נוספים התומכים בשירותי Discovery שונים שאותם ניתן לזהות ע"י הרחבות של NMAP הם מדפסות רשת, שרתי DHCP, שירותי DNS, מחשבים המוגדרים כ-Master Browser ועוד.

במידה ונרצה לאתר מטרות ברשת בצורה שקטה לחלוטין נוכל לבצע "Passive Scan" - הרעיון הוא להריץ את NMAP שיאזין לחבילות הנשלחות כ-Broadcast ברשת, ובסופו של דבר יציג לנו את כלל הרכיבים שזוהו. ההיתרון בסריקה מסוג זה היא שמדובר בפעולה פאסיבית לחלוטין, כך שרכיבי IDS שונים לא יוכלו לאתר אותה. לטובת ביצוע סריקה זו, נכתבה ההרחבה broadcast-listener, ונריץ את NMAP באופן הבא על מנת להפעילה:

```
nmap --script broadcast-listener --script-args timeout=<seconds>
```

כעת, NMAP תאסוף את כלל החבילות אשר נשלחות מרכיבי הרשת ב-Broadcast ותנתח אותן (חבילות כגון CDP, HSRP, DHCP, ARP ועוד), ולאחר ה-timeout, תציג כפלט את כלל הרכיבים שנמצאו.

כאמור, ההיתרון של סריקה זו הוא שהיא בלתי-נראת, אך החסרון שלה הוא שברשתות קטנות - נאלץ להריץ את הסריקה לא מעט זמן על מנת לאתר את הרכיבים בה, ומה גם שלא נוכל לאתר רכיבים שאינם מפרסמים אודותם.

זיהוי מערכת הפעלה במדוייק על ידי סקריפטים

יש ל-nmap המון תוספות סריקה שונות. כמובן שלא נספיק לכתוב כאן על כולם אך תוכלו למצוא רבות עוד בקישור הזה. יש כמה תוספות שאני מוצא שימושיות במיוחד. לדוגמא במידה ואתם מבצעים סריקה פנימית או לחילופין מנסים לסרוק דרך מכונה אחרת שנמצאת בתוך ה-LAN נרצה להגיע לגרסת מערכת הפעלה מדוייקת. תראו את תוצאות הסריקה הבאה:

```
tisf ~/ > sudo nmap -A 192.168.6.209

Starting Nmap 6.46 ( http://nmap.org ) at 2014-07-06 20:48 IDT
Nmap scan report for 192.168.6.209
Host is up (0.038s latency).
Not shown: 986 closed ports
PORT        STATE SERVICE          VERSION
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn
443/tcp   open  ssl/http      VMware VirtualCenter Web service
|_http-methods: No Allow or Public header in OPTIONS response (status
code 501)
|_http-title: Site doesn't have a title (text; charset=plain).
| ssl-cert: Subject: commonName=VMware/countryName=US
| Not valid before: 2014-06-08T07:15:19+00:00
|_Not valid after:  2015-06-08T07:15:19+00:00
445/tcp   open  netbios-ssn
902/tcp   open  ssl/vmware-auth VMware Authentication Daemon 1.10 (Uses
VNC, SOAP)
912/tcp   open  vmware-auth   VMware Authentication Daemon 1.0 (Uses
VNC, SOAP)
```

```
3389/tcp  open  ms-wbt-server?
5405/tcp  open  netsupport    NetSupport PC remote control (Name IITC-
501)
49152/tcp open  msrpc         Microsoft Windows RPC
49153/tcp open  msrpc         Microsoft Windows RPC
49154/tcp open  msrpc         Microsoft Windows RPC
49158/tcp open  msrpc         Microsoft Windows RPC
49159/tcp open  msrpc         Microsoft Windows RPC
49160/tcp open  msrpc         Microsoft Windows RPC
MAC Address: 00:27:0E:09:49:83 (Intel Corporate)
Device type: general purpose
Running: Microsoft Windows 2008|7
OS CPE: cpe:/o:microsoft:windows_server_2008::sp2
cpe:/o:microsoft:windows_7::- cpe:/o:microsoft:windows_7::sp1
cpe:/o:microsoft:windows_8
OS details: Microsoft Windows Server 2008 SP2, Microsoft Windows 7 SP0 -
SP1, Windows Server 2008 SP1, or Windows 8
Network Distance: 1 hop
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_nbstat: NetBIOS name: MARVIN, NetBIOS user: <unknown>, NetBIOS MAC:
00:27:0e:09:49:83 (Intel Corporate)
| smb-os-discovery:
|   OS: Windows 7 Ultimate 7600 (Windows 7 Ultimate 6.1)
|   OS CPE: cpe:/o:microsoft:windows_7::-
|   Computer name: MARVIN
|   NetBIOS computer name: MARVIN
|   Workgroup: WORKGROUP
|_  System time: 2014-07-06T20:49:26+03:00
| smb-security-mode:
|   Account that was used for smb scripts: guest
|   User-level authentication
|   SMB Security: Challenge/response passwords supported
|_  Message signing disabled (dangerous, but default)
|_smbv2-enabled: Server supports SMBv2 protocol

TRACEROUTE
HOP RTT     ADDRESS
1   37.70 ms 192.168.6.209

OS and Service detection performed. Please report any incorrect results
at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 70.23 seconds
```

שימו לב שבסריקת מערכת ההפעלה nmap לא הצליחה לקבל זיהוי מדויק מכיוון שהיא זיהתה קרנל ולכן היא פלטה את התוצאות הבאות:

```
Running: Microsoft Windows 2008|7
OS CPE: cpe:/o:microsoft:windows_server_2008::sp2
cpe:/o:microsoft:windows_7::- cpe:/o:microsoft:windows_7::sp1
cpe:/o:microsoft:windows_8
OS details: Microsoft Windows Server 2008 SP2, Microsoft Windows 7 SP0 -
SP1, Windows Server 2008 SP1, or Windows 8
```

מה שאומר שיכול להיות שמדובר במערכת הפעלה חלונות 7 או חלונות 2008. שתי גרסאות שונות מאוד אחת מהשניה. יחד עם זאת נוכל לראות מתחת שרצו סקריפטים נוספים. בואו נסתכל על התוצאות שלהם:

```
Host script results:
|_nbstat: NetBIOS name: MARVIN, NetBIOS user: <unknown>, NetBIOS MAC:
00:27:0e:09:49:83 (Intel Corporate)
| smb-os-discovery:
|   OS: Windows 7 Ultimate 7600 (Windows 7 Ultimate 6.1)
|   OS CPE: cpe:/o:microsoft:windows_7::-
|   Computer name: MARVIN
|   NetBIOS computer name: MARVIN
|   Workgroup: WORKGROUP
|_  System time: 2014-07-06T20:49:26+03:00
```

ובמקרה זה נוכל לשים לב ששני הסקריפטים nbtstat ו-smb-os-discovery החזירו לנו תוצאות מדויקות מאוד של מערכת ההפעלה. ישנם עוד סקריפטים רבים כגון אלו ואני ממליץ בחום לפחות לעבור על רשימת הסקריפטים הקיימים.

סיכום

הכלי nmap הוא כלי גדול ועצום. רובנו משתמשים בו בלי להבין את הפוטנציאל שלו. קריאה לא נכונה של סריקות עלולה להוביל לנפילה של מבדק ועלולה בהחלט להוביל גם לחסימות ברכיבים שונים. כולי תקווה שכל קוראי המאמר יכול לאחר מכן לאתר הרשמי של nmap ויקראו את שאר המדריכים הנהדרים ואת שאר החומרים הקיימים שם כדי ללמוד יותר טוב את הכלי. בהצלחה במבדקים הקרובים ולהתראות בגיליונות הבאים. :)