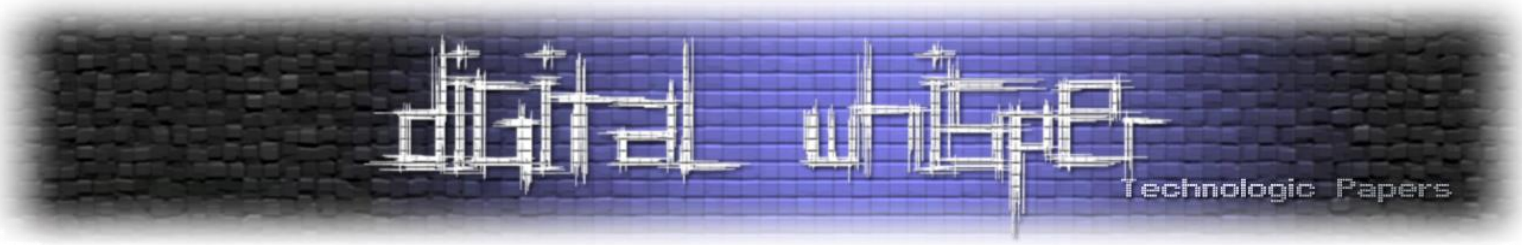


iptables unleashed

מאת עדי מליאנקר (Magen_Gide0n)

הקדמה

במאמר הזה, אני מקווה לסגור, או לפחות להמשיך את היריעה ממאמר מצוין שכתב אפיק קסטיאל בדצמבר 2009, לפני כמעט 10 שנים באשר ל-iptables ולהבין את התהליכים שקורים בטבלאות השונות. הרעיון להמשיך את המאמר הגיע מתוך מאמר אחר שיצא לי לכתוב למגזין (אך עוד לא פורסם) באשר להתקדמות ברשת ע"י port forwarding ומתקשר במעט ל-iptables.

קודם כל ולפני כל, נביא כהקדמה קטנה סקירה של המאמר של אפיק, כי 10 שנים, לך תזכור... (למרות שממליץ בחום לקרוא לפני את המאמר הנ"ל).

סקירה מהעבר

אם כן, iptables הינו firewall מבוסס אירועים המכיל תכונת stateful firewalling, כלומר היכולת לבדוק חבילה בפני עצמה וגם בהקשר לשיחה אליה היא שייכת. הכלי נכתב על ידי Rusty Russel עבור מערכת ההפעלה לינוקס ומגיע מובנה בה. הכלי מכיל טבלה ענקית הנקראת filtering table אשר מחולקת לכמה טבלאות בסיסיות לניהול תעבורת החבילות. הטבלאות הן filter (שהיא הטבלה הדיפולטיבית), טבלת NAT האחראית על פעולות לניתוב הפקטות וטבלת mangle לטיפול מתקדם בחבילות.

כל טבלה מכילה כמה שרשראות, אוסף חוקים. לדוגמא, טבלת filter מכילה 3 סוגי שרשראות: input, output ו-forward. מערכת iptables מנתבת את החבילות לטבלאות המתאימות. כאשר input הן החבילות הנכנסות, output הן החבילות היוצאות ו-forward הן החבילות אשר מיועדות לעבור הלאה... כל טבלה מכילה אוסף של חוקים, כמו כל firewall בסיסי ומתבצעת בדיקה האם ישנו חוק כלשהוא בטבלה המדבר על החבילה, **כאשר סדר החוקים הוא חשוב** ואם נמצא חוק שמתאים לפקטה, יתבצע מה שרשום בחוק ויופסק החיפוש (גם אם יש בהמשך חוקים שעשויים גם להתאים לחבילה).

נניח שנתונה לנו הטבלה הבאה:

No.	Protocol	Source IP	Destination IP	Dest. Port	Action
1	TCP	10.1.1.1	20.1.1.1	80	Accept
2	TCP	10.1.1.2	20.1.1.1	80	Deny
3	TCP	10.1.1.0/24	20.1.1.1	80	Deny
4	TCP	10.1.1.3	20.1.1.1	80	Accept
5	TCP	10.2.2.0/24	20.2.2.5	80	Deny
6	TCP	10.2.2.5	20.2.2.0/24	80	Deny
7	TCP	10.3.3.0/24	20.3.3.9	80	Accept
8	TCP	10.3.3.9	20.3.3.0/24	80	Deny
9	IP	0.0.0.0/0	0.0.0.0/0	0-65535	Deny

[מתוך research gate]

במקרה זה, אם תגיע ל-firewall הודעת tcp מ-10.1.1.1 אל 20.1.1.1 בפורט 80, ההודעה מיד תתקבל בחוק מספר 1 (למרות שחוק מספר 3 גם היה עשוי לעניין אותנו, אבל כמו שאמרנו, הסדר חשוב... ולכן לא נגיע בכלל לבדיקת חוק מספר 3).

קיימות 5 אפשרויות להתנהלות אל מול חבילה:

- accept - העברת חבילה
- drop - זריקת חבילה
- reject - כמו drop, אבל תשלח הודעת שגיאה לשולח
- queue - העברת החבילה לשימוש user space
- return - החזרת הקריאה של החוקים לחוק שממנו הופעל החוק הנוכחי).

2 חוקי ברזל חשובים:

- כאשר נוסיף חוקים לטבלה, הם דיפולטיבית, אלא אם ציינו אחרת, יהיו בתחתית הטבלה.
- בדיקת החוקים מתחילה מראש הטבלה את תחתיתה.

עד כאן הסקירה, בואו ניגש לעבודה.

אם כן, כיצד iptables עובד?

הכלי iptables מדבר עם hook-ים של packet filteting ברמת הקרנל, hook-ים השייכים לסביבת netfilter.

מושגים רבותיי, מושגים

מה זה hook?

המושג hook מתייחס להטמנת קוד בסביבת מערכת ההפעלה כך שהקוד יקרא על ידה כאשר תקרה פעולה מסוימת ברמת מערכת ההפעלה. hooking משמש לביצוע debugging, הרחבת פונקציונליות של תוכנית מסוימת וכן כתיבת תוכנות המסוות את עצמן כמו rootkits למיניהן. באחת מבדיקות החדירות שיצא לי לבצע בעבר, יצא לי להשתמש בתוכנית המשתמשת ב-hook-ים על מערכת ההפעלה כדי ליירט הקשות מקשים והקלקות עכבר, keylogger.

NAT - על מה ולמה?

בואו נדבר על NAT , ראשי תיבות של network address translation. כמה פעמים יצא לכם לבצעipconfig ולראות subnet מוזר שדי תמיד חוזר על עצמו בכל בית הישראל- 10.0.0.0/24 או כתובת שמתחילה 192.168, צירוף מקרים? לא ממש. התפקיד של NAT נולד מהצורך ההולך וגובר לאבטחה עבור עמדות הקצה בסביבות פרטיות וכן העודף ההולך וגדל של כתובות ip בעולם. לשם כך, נולד מושג NAT שבו אנו לוקחים סביבות פרטיות-ביתיות או תעשייתיות, מקיפים אותן ב"שומר", שמבצע את הקישור בינן לבין החוץ.

לכל מחשב ברשת הפנימית, ניתנת כתובת פנימית. הכתובת הפנימית רלוונטית רק לרשת הפנימית ולכן ברשתות פנימיות שונות נוכל למצוא כתובות ip זהות. הקישור ממיר בין הכתובת הפנימית אל הרשת החיצונית, האינטרנט. כך, מחשב באינטרנט לא יכול לפנות ישירות אל מחשב ברשת הפנימית (היות והוא לא מכיר ישירות את כתובת ה-ip של המחשב הפנימי והיות ויש הרבה מאוד מחשבים עם הכתובת הזו). מה שהוא כן יכול לעשות, הוא לדבר עם רכיב ה-NAT, השומר, על מנת שיקשר אותו למחשב הפנימי (בדרך כלל הוא גם צריך לחכות שהמחשב הפנימי יפנה אליו כתוצאה מרכיבים כמו firewall הנמצאים בעסק).

אופן הפעולה של iptables

חבילות המגיעות ללינוקס, יעוררו את ה-hook-ים, כאשר הם נכנסים ל-stack החבילות הדורשות פעולה. בפעולה זו, נגרום לכלל התוכנות, kernel modules, הרשומות ל-hook-ים להתעורר ולשמש בתפקיד מפתח בעבודה עם החבילות. ישנם 5 hook-ים ב-netfilter שניתן להירשם אליהם. ה-hook-ים שחבילות יעוררו תלויים באופי החבילה, האם היא נכנסת או יוצאת, מה יעדה, האם היא נפלה או נדחתה לפני כן ועוד.

ה-hook-ים הם:

- NF_IP_PRE_ROUTING המתעורר כאשר נכנסת חבילה כלשהיא למחסנית הרשת ועדיין לא הוחלט לאן לנתב אותה.
- NF_IP_LOCAL_IN מתעורר כאשר מגיעה חבילה המיועדת למערכת המקומית.
- NF_IP_FORWARD המתעורר כאשר מגיעה חבילה המיועדת למארח אחר.
- NF_IP_LOCAL_OUT המתעורר כאשר חבילה כלשהיא יוצאת מהמכונה המקומית החוצה.
- NF_IP_POST_ROUTING המתעורר כאשר חבילה אמורה להיות מועברת הלאה או לצאת מהמכונה והיא קיבלה כבר החלטת ניתוב מhookים אחרים. ה-hook קורה קצת לפני יציאת החבילה החוצה.

kernel modules המעוניינים להירשם ל-hook-ים, חייבים לספק priority number בכדי לקבוע את סדר התוכנות שירוצו כאשר hook מסוים נקרא. כל תוכנה תרוץ כאשר יגיע תורה ותחזיר את החלטה אל ה-netfilter framework לאחר שהחליטה מה לבצע בחבילה.

כמו שאמרנו, כל טבלה ב-iptables מסודרת בעזרת שרשראות. כל טבלה מייצגת את סוג החוקים שהיא מחזיקה (לדוגמא טבלת NAT תחזיק חבילות המתעסקות עם חבילות NAT, לתרגום כתובת פנימית לחיצונית ולהיפך), שרשרת , לעומת זאת, מייצגת hook מסביבת netfilter framework ומחליטה מתי חוק מסוים יקרא (לדוגמא קיים בטבלת FILTER שרשרת מסוג INPUT המתייחס לחבילות הנכנסות פנימה ול-hook של NF_IP_LOCAL_IN וכן הלאה...). השרשרת היא שמחזיקה בפועל את החוקים. נשים לב לפרט חשוב-לכל אחת משלושת הטבלאות, אין שרשראות עבור כל חמשת ה-hook-ים, אלא רק עבור אלה הרלוונטים בשבילם. עוד פרט חשוב הוא שיש חפיפה בין השרשראות בטבלאות השונות (לדוגמא גם בטבלת FILTER וגם בטבלת NAT ישנה שרשרת של output). כיצד זה עובד, אתם שואלים את עצמכם?

כמו שאמרתי קודם, תוכנות המעוניינות להירשם לhook מסוים שולחות אליו את הpriority number שלהן וכך, לדוגמא כאשר ה-hook של NF_IP_LOCAL_OUT יתעורר, אם ה-priority number של output בטבלת FILTER גבוה מהמספר של output בטבלת NAT, יקראו קודם כל החוקים בטבלת output ב-FILTER לפי הסדר ורק לאחר מכן יקראו החוקים בטבלת output ב-NAT.

הטבלאות הינן:

- טבלת FILTER, הטבלה הדיפולטיבית והנפוצה ביותר, הטבלה שנותנת ל-iptables את הכינוי firewall, אחראית על חבילות היוצאות, נכנסות ומועברות.
- טבלת NAT, המממשת NAT ומבצעת המרה של כתובת ip אחת ופורט אחד לאחרים בכדי להיכנס לרשתות שאין גישה ישירה אליהן.
- טבלת MANGLE משמשת לצורך שינוי ה-header-ים של חבילות של ה-ip בדרכים שונות (שינוי ה-type of service, ttl והגדרות סימון).
- טבלת RAW מתייחסת לחבילות בהקשר של מיקומן ב-session או קשר מסוים.

- טבלת security מאפשרת לסמן חבילות מסוימות עבור SElinux (נגנון בלינוקס המאפשר להגביל כניסה של אובייקטים, כמו לדוגמא תהליכים אל קבצים מסוימים) וטיפולה בהן.

בטבלה הבאה ניתן לראות את השרשראות הנמצאות בכל טבלה.

Tables↓/Chains→	PREROUTING	INPUT	FORWARD	OUTPUT	POSTROUTING
(routing decision)				✓	
raw	✓			✓	
(connection tracking enabled)	✓			✓	
mangle	✓	✓	✓	✓	✓
nat (DNAT)	✓			✓	
(routing decision)	✓			✓	
filter		✓	✓	✓	
security		✓	✓	✓	
nat (SNAT)		✓			✓

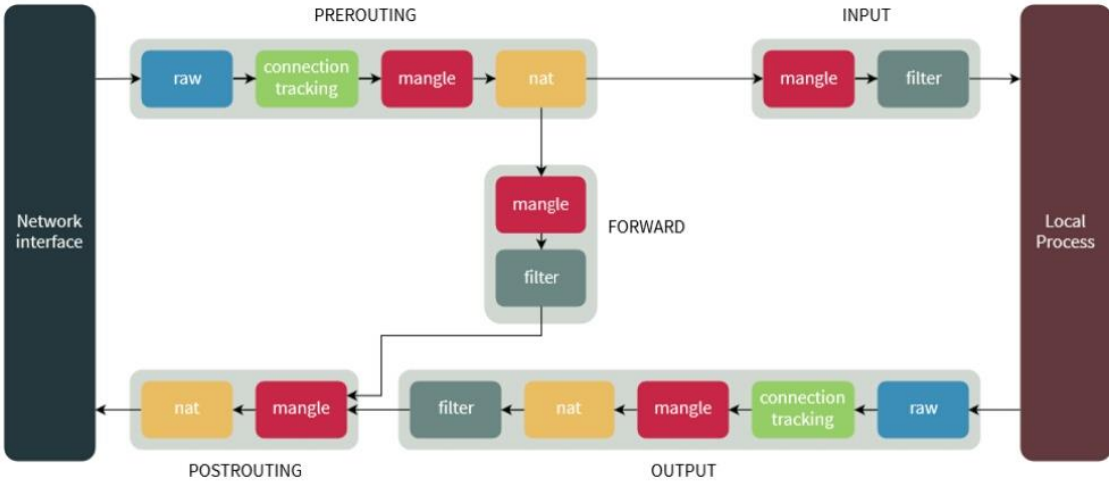
[מתוך: https://www.digitalocean.com/community/tutorials/a-deep-dive-into-iptables-and-netfilter-architecture]

**נשים לב לשוני בין DNAT (מתבצע שינוי כתובת היעד) ל-SNAT (מתבצע שינוי כתובת מקור).

כמו כן, ישנם אירועים מסוימים שיביאו לכך שחלק מהטבלאות לא יקראו, לדוגמא רק עבור החבילה הראשונה בקשר, תתבצע ניתוח של חבילה בטבלת NAT, עם שאר החבילות תתבצע אותה ההחלטה שהתקבלה בראשונה.

מתוך שיקול של גמישות, החליט יוצר iptables לאפשר למשתמש ליצור טבלאות משלו. אבל אנחנו נתקעים פה בבעיה, הלא כן? איך נגיע לטבלאות הללו בזמן ריצת המערכת? כאן מגיע פעולת jump לעזרנו המאפשרת קפיצה מהטבלאות המסודרות המוגדרות ב-iptables לטבלאות שיצר המשתמש.לאחר שקפצנו לטבלאות הללו, נוכל לחזור לטבלה הקודם לאחר שהסתיימה קריאת כל החוקים או אם נתקלנו בפקודת RETURN.

נוכל לראות שסדר ההרצה הוא כך:



[מתוך: https://www.google.com/amp/s/www.booleanworld.com/depth-guide-iptables-linux-firewall/amp]

תכונת Connection tracking

כאמור, iptables היא firewall הבודק חבילות גם מתוך המיקום שלהם בחיבור, לפי ה-contex שלהם ויחסן לחבילות הקודמות בקשר. תכונת connection tracking הנמצאת ב-netfilter framework מאפשרת ל-iptables לסווג חבילות לפי קשרים ולא רק להסתכל על החבילות כאינדיבידואליות. כאשר חבילה רק נכנסת למחסנית הרשת, מתבצע עליה בדיקת ה-connection tracking (למרות שטבלת RAW בודקת את החבילה לפני כן) והחבילה נבדקת אל מול רשימה של חיבורים שתועדו בכדי למצוא שייכות של החבילה. לאחר מכן, מעודכן סטטוס החיבור. כמובן, חבילות שסומנו כ-NO TRACK בלוגיקת טבלת RAW, ידלגו על הבדיקה של connection tracking.

מצב קשר יכול להיות אחד מהבאים:

- NEW - כאשר מגיע חבילה שלא קשורה לשום חיבור, נוצר חיבור חדש. הדבר רלוונטי הן ל-tcp והן ל-udp.
- established - כאשר עובר בחיבור מסוג NEW, חבילה בכיוון תשובה. עבור tcp מדובר בsyn+ack, עבור udp ו-icmp מדובר בחבילה שבה כתובת המקור והיעד הפוכות מאלו שבחבילה הראשונה בקשר.
- RELATED - כאשר מגיעה חבילה שאינה חלק מקשר קיים, אבל קשורה אליו (לדוגמה תשובת icmp על פרוטוקול אחר).
- INVALID - כאשר חבילה אינה חלק מקשר קיים ואינה תקינה לפתוח קשר חדש / אינה ניתנת לניתוב / אינה ניתנת לזיהוי.
- UNTRACKED - דיברנו על זה קודם, כאשר טבלת RAW יכולה לסמן חבילה במצב זה ולדלג על בדיקת ה-connection tracking.

- SNAT -סטטוס המציין כי בחבילה שהגיעה אלינו שונתה כתובת המקור. מיידעת את
  - ה-connection tracking שבהעברת החבילה מהמכונה המקומית יש לשנות כתובת המקור.
- DNAT -סטטוס המציין כי בחבילה שהגיעה אלינו שונתה כתובת היעד. מיידעת את
  - ה-connection tracking שבהעברת החבילה מהמכונה המקומית יש לשנות טת כתובת היעד בחבילה.

היכולת של ה-connection tracking מאפשרת למשתמש לייצר חוקים המתייחסים לסטטוס החיבור.

קיימים מודולים שונים היכולים להצטרף ל-iptables, אחד מהם הוא connection tracking, אחרים נוספים הם tcp המשמש לבדיקת דגלי tcp בחבילות, limit לצורך הגבלת מספר החבילות העוברות, owner המשמש לחסימת חבילות פר משתמש ומארח ועוד. טעינת המודול בעזרת דגל m, לדוגמא:

```
iptables -A INPUT -p tcp -m tcp --tcp-flags ALL FIN,PSH,URG -j DROP
```

כאן, נזרקות חבילות tcp המכילות את כלל הדגלים.

קמצוץ של אבטחה לעסק

כעת, אחרי שהבנו כיצד להשתמש ב-iptables (מאמר הקודם) ומה עומד מאחורי iptables, אני מעוניין לתת כמה עצות להגן על המחשב שלנו בבית על בסיס iptables המבוססות על best practices. חשוב לציין שיכול להיות שחלק מהדרישות לא יתאימו לשימושים מיוחדים שאתם עושים במחשב, אבל הדרישות עצמן, נועדו להיות גנריות ולהתאים לרוב המוחלט של המשתמשים. כמו כן, חשוב לי לציין שהגנה על מחשב היא מורכבת מאוד ואינה בנויה על רכיב אחד בלבד ולעיתים, גם הממשקים שבין הכלים השונים יכולים להיות קריטיים למערכת. אז בואו נתחיל:

- לכל טבלה ישנה מדיניות ברירת מחדל, המדיניות **אינה** החוק האחרון שיש בטבלה אלא היא כלל חיצוני. ההמלצה היא שלא לשים מדיניות דיפולטיבית כלהיא ב-DROP. מדוע? אם בטעות מנהל רשת אחר ימחק את כלל חוקי הטבלה, או אם לא שמרתם את החוקים, אתם עשויים להביא לכך שתמיד תזרקו את כלל החבילות המגיעות. לכן, ההמלצה היא לשים את מדיניות ברירת המחדל ב-ACCEPT ובחוק הכי אחרון בטבלה, לשים באופן מפורש חוק של DROP אם לא נמצאו התאמות.

- אם יצרתם באחת הטבלאות קבוצת חוקים שעשויה להיות מסובכת, שיקלו להעביר את החוקים לשרשרת חדשה משלכם ושימו JUMP אל מיקום זה. הדבר עשוי לעזור לכם לשמור על סדר ולנהל טת הטבלה בקלות.

- נסו להיות כמה שיותר ספציפיים בחוקים שלכם בכדי לא לאפשר חבילה שלא רציתם לאפשר. לדוגמא, בכדי לאפשר כניסת ping-ים למכונה, אני אעדיף את החוק הבא, שמאפשר icmp מסוג 8 בלבד (פינג):

```
iptables -A INPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT
```

על פני החוק הבא שמאפשר כל icmp:

```
iptables -A INPUT -p icmp -m icmp -j ACCEPT
```

- זכרו תמיד לשמור את החוקים שלכם, אין פה שמירה אוטומטית לאחר כיבוי המחשב!
- כמו שחשוב לדעת ולתעד מי נכנס למדינה בכל רגע, ככה חשוב לתעד את החבילות המתאימות לחוק מסוים, זה יעזור לתחקורים מתקדמים יותר במחשב...
- מדיניות DROP היא תמיד מומלצת בסיום טבלאות.
- שימו חוקים ספציפיים יותר בראש הטבלה וחוקים גנרים יותר בראשה.
- אפשרו למשתמשים את מינימום השירותים הנדרשים לעבודתם (שירותים כמו http ו-dns הם הכרחיים לעבודת המערכת), אבל האם לעבודת המערכת, אתם רוצים לאפשר לדוגמא שירותים כמו telenet ו-ssh? ומה לגבי smb פנימה והחוצה?
- חסמו את מספר הקישורים שמחשב כלשהו יכול לבצע אליכם כדי למנוע DOS, לדוגמא בעזרת מודול של multiports. מראה את החוק הבא המאפשר להגביל את מספר החיבורים החדשים ל-20 עבור https ו-http:

```
iptables -A INPUT -p tcp -syn -m multiport --dport http,https -m
connlimit --connlimit-above 20 -j REJECT
```

- הפקטה לא טובה להעברה הלאה? העדיפו לעשות לה DROP ולא REJECT. בעזרת REJECT, אנו מדווחים לצד השני שהחבילה נפלה. נעדיף להשתמש ב-DROP בכדי להימנע מעיבוד נוסף למערכת, אשר עשוי לעדכן גם גורם שעשוי להיות תוקף למערכת.

סיכום

במהלך המאמר הצגתי כיצד עובד הכלי של iptables מאחורי הקלעים, הצגתי את המנגנונים שבהם iptables משתמש אל מול מערכת ההפעלה ומדוע. דיברנו קצת על hook-ים ו-NAT והקשר שלהם ל-iptables. לסיום, הבאתי כמה best practices לעבודה עם iptables כמנהלים רשתות וכאנשים פרטיים. מקווה שעכשיו הכלי הקטן הזה שנמצא על כל מחשב, מובן לכם יותר. אז לסיום,

Be ethnical, be safe and dont be a script kiddies

על הכותב

עדי מליאנקר, 22 חוקר אבטחה עצמאי, מתעסק ב-bug bounty וייעוץ לחברות. כותב חיצוני בבלוג אבטחה בעניינים הפנימיים של מערכת ההפעלה windows (תבוא לבקר!).

זמין לשאלות, הערות, ויותר מאשמח לתוספות לדברים ששכחתי במאמר או סתם memes במייל (temporarynothing0gmail.com).

ביבליוגרפיה

- https://www.jensd.be/343/linux/forward-a-tcp-port-to-another-ip-or-port-using-nat-with-iptables
- https://www.digitalocean.com/community/tutorials/a-deep-dive-into-iptables-and-netfilter-architecture
- https://www.booleanworld.com/depth-guide-iptables-linux-firewall/amp/
- https://major.io/2010/04/12/best-practices-iptables/
- https://opensource.com/article/18/10/iptables-tips-and-tricks
- https://www.lammertbies.nl/comm/info/iptables.html&ved=2ahUKEwiSj9bhzOviAhVOZJoKHcU4Ad0QFjADegQIARAB&usg=AOvVaw3xy_wCJ2mEIxoKTUU1OJC5