**Cyber Lab Ex1 – SYN Flood**

הסבר כללי: **במטלה זו קיבלנו Docker שמכיל 3 מכונות – Server, Attacker, Monitor.**

**התבקשנו לבצע התקפת SYN Flood באמצעות מכונת ה-Attacker על ה-Server.**

**רעיון ההתקפה:**

**מכונה אחת (Attacker) שולח הודעות SYN רבות לשרת (Server).**

**השרת מחזיר הודעת SYN-ACK לכל הודעה כזו, ומקצה משאבים ע"מ להמשיך תקשורת עם המכונה.**

**מכיוון שכמות הודעות ה-SYN גדולה – השרת מקצה משאבים רבים לצורך התקשורת, מה שמוביל למצב בו הוא לא קיימים לא משאבים מספיקים בשביל לתקשר עם מכונות אחרות. וכך בעצם אנו מבצעים Denial Of Service לשרת.**

**בפועל מבוצע Distributed Denial Of Service, שכן אנו מזייפים כתובות IP כדי שהשרת לא ידע איזה IP לחסום. וכך בעצם אנו מונעים מהשרת לעצור את ההתקפה.**

**מטרת העל של כל Image:**

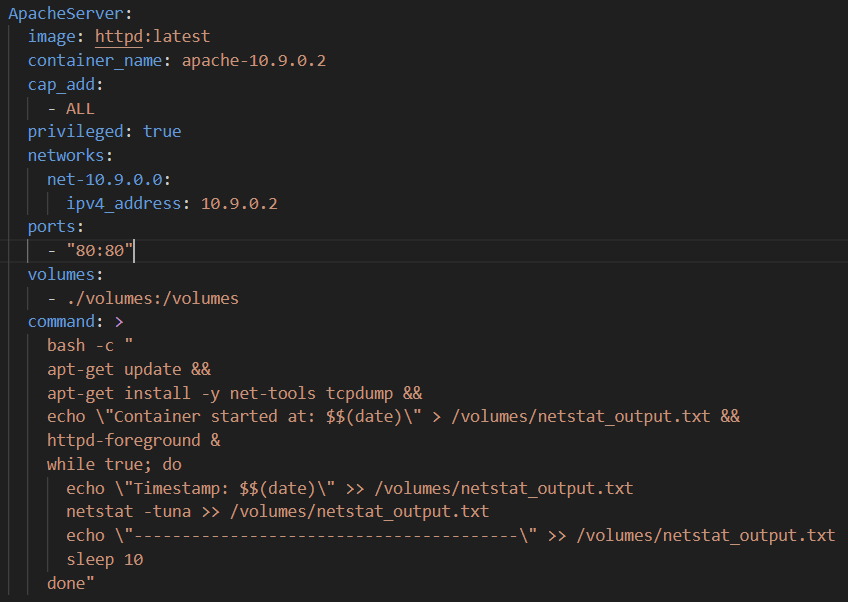
1. **Server – שרת Apache2. משמש בתור מטרה לתוקף.**
2. **Attacker – לבצע שליחה של חבילות SYN מרובות על גבי פרוטוקול TCP לפורט 80**

**ב-Server.**

1. **Monitor – שליחת PING לשרת ע"מ לנתר את העומס על השרת בזמן המתקפה. זאת ע"י בדיקת זמני RTT לפני ההתקפה, במהלך ההתקפה ואחריה.**

**הסבר על כל Image:**

**Server**

****

**שרת HTTP בכתובת 10.9.0.2 שמבצע חלק פסיבי בהתקפה בכך שהוא המטרה של התוקף.**

**כאשר ה-Image של השרת עולה. השרת מתקין כלי שנקרא tcpdump, מה שמאפשר לנו לנתר את כמות החיבורים במצב SYN RECV (חיבורים מהם השרת קיבל הודעת SYN וממתין להודעת ACK על הודעת ה-SYN ACK אותה שלח).**

**אנו מדפיסים לקובץ את תוצאת netstat -tuna כל 10 שניות. כך אנו מתעדים את השינוי בחיבורים הפתוחים בשרת בין זמן ההתקפה לזמן שלפניה ואחריה.**

**קובץ זה נועד לתיעוד שלנו שהמתקפה אכן עשתה את שנדרשה ואינו חלק מהמטלה ולכן אינו מצורף.**

**Monitor**

**A screen shot of a computer

Description automatically generated**

**מכונת ה-Monitor שולחת הודעת Ping יחידה לשרת ה-Apache כל 5 שניות.**

**תוצאת כל הודעה (הדפסת הסטטיסטיקות) נכתבת בקובץ ייעודי.**

**לכל שורה בקובץ זה ביצענו Parsing ויצרנו את הגרפים הבאים המתארים את כמות הודעות ה-PING שנשלחו עבור כל זמן RTT:**

**A graph with blue lines

Description automatically generated**

**A graph with blue lines

Description automatically generated**

**Attacker:**

**A screen shot of a computer

Description automatically generated**

**A screen shot of a computer program

Description automatically generated**

**ראשית, נשים לב שהתוקף רץ על תמונה מבוססת GCC/Python. זאת מכיוון שהתמונה של ubuntu לא מכילה את ה-Compiler/Interpreter ומתוך כך לא היה ניתן להריץ/לקמפל את קובץ ההרצה מבלי להוריד את ה-Compiler/Interpreter המתאימים.**

**השתמשנו בתמונות אלו כדי לייעל את תהליך ההתקפה (בפועל לא משפיע על זמן הריצה, שכן אנו ממתינים 20 שניות לפני ההתקפה).**

**ב-C: נקמפל את הקובץ המתאים שנמצא בתיקיית ה-volumes. נמתין 20 שניות ולאחר מכן נריץ אותו.**

**ב-Python: נוריד את התוספים שהקוד שכתבנו דורש. נמתין 20 שניות ולאחר מכן נריץ את קובץ ההתקפה.**

**המטרה שהתוקף מחכה 20 שניות היא בשביל שהשרת וה-Monitor יעלו באופן מלא, כלומר שיסיימו את ההתקנות הנדרשות עבורן ויפעלו כרגיל.**

**קוד:**

**הרעיון המרכזי שעומד בבסיס הקוד ב-C וב-Python, הוא יצירת RAW Socket, יצירת פאקטת SYN באופן ידני, לשלוח אותה כמספר הפעמים שנרצה.**

**בנוסף, ב-2 התוכניות אנו מבצעים IP & Port Spoofing כדי שהנתקף לא יוכל לחשוף את זהותנו ולחסום אותנו.**

**קוד ב-C:**

**A computer screen shot of a program code

Description automatically generated**

**נגדיר משתנים בהם נשתמש בהמשך הקוד ע"מ לתעד את הזמן שלקח לשלוח כל פאקטה בנפרד וביחד.**

**תיעוד זה יתבצע לתוך הקובץ syn\_flood\_log.txt.**

**A computer screen shot of a program code

Description automatically generated**

**נפתח Raw Socket ונגדיר אותו כך שיוכל לקבל Headers "מבחוץ".**

**A screen shot of a computer

Description automatically generated**

**נגדיר פאקטה ונאפס אותה.**

**נגדיר מצביעים למיקומים המתאימים ל-TCP Header ו-IP Header בפאקטה (כך ששינוי שלהם ישנה ערכים בפאקטה).**

**A computer screen shot of a program

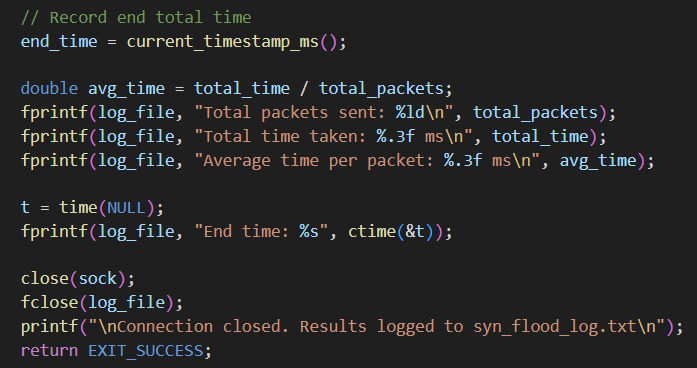
Description automatically generated**

**נתחיל במתקפה – ראשית, נבצע השמה של ערכים לתוך ה-IP Header & TCP Header באמצעות פונקציה ייעודית (אסביר עליה בהמשך).**

**לאחר מכן נגדיר לאן הפאקטה תישלח (sendto מקבל כתובת בתצורת struct sockaddr. לכן ההשמות שביצענו ל-Headers אינן רלוונטיות כאן).**

**נשלח את הפאקטה ונחשב את הזמנים המתאימים.**

**נשים לב ש-NUM\_OF\_TRIES מוגדר ל-10,000 ו-NUM\_OF\_ITERATIONS מוגדר ל-100 כנדרש.**

****

**לבסוף, נתעד את כלל הסטטיסטיקות בקובץ ה-log ונסיים את התוכנית.**

**תיעוד handle\_packet:**

**A computer screen shot of a program

Description automatically generated**

**מטרת פונקציה זו היא לבצע את ההשמות הנדרשות ב**