**Lab 1**

**Sniffing & Spoofing**

טל קלדרון:302287974

אלון לידובסקי:313559114

לידור סויסה:316048131

Task 1.1 Sniffing Packets

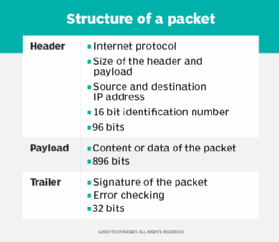
Task 1.1.A

1. מבוא –

Sniffing – תהליך של ניטור ולכידה של כל מנות הנתונים העוברות ברשת נתונה. Sniffers משמשים את מנהל הרשת/מערכת לניטור ופתרון בעיות של תעבורת רשת. תוקפים משתמשים ברחרחנים כדי ללכוד חבילות נתונים אשר מכילות את פרטי השולח, המקבל ומרכיבות חלק מן המידע(פקטות) אשר מכיל חלקים מן המידע העובר ברשת. מידע זה עלול להיות רגיש כמו סיסמאות, פרטי חשבון וכו'.

****

Router



Sniffing

Attacker

server

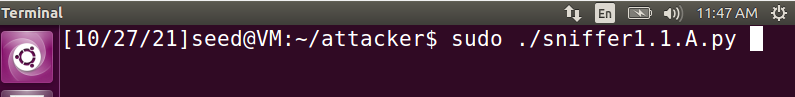
2ביצוע המטלה –

לאחר הקמת שתי מכונות וירטואליות אשר אחת שימשה כתוקפת – Attacker והשנייה כנתקפת – Server.

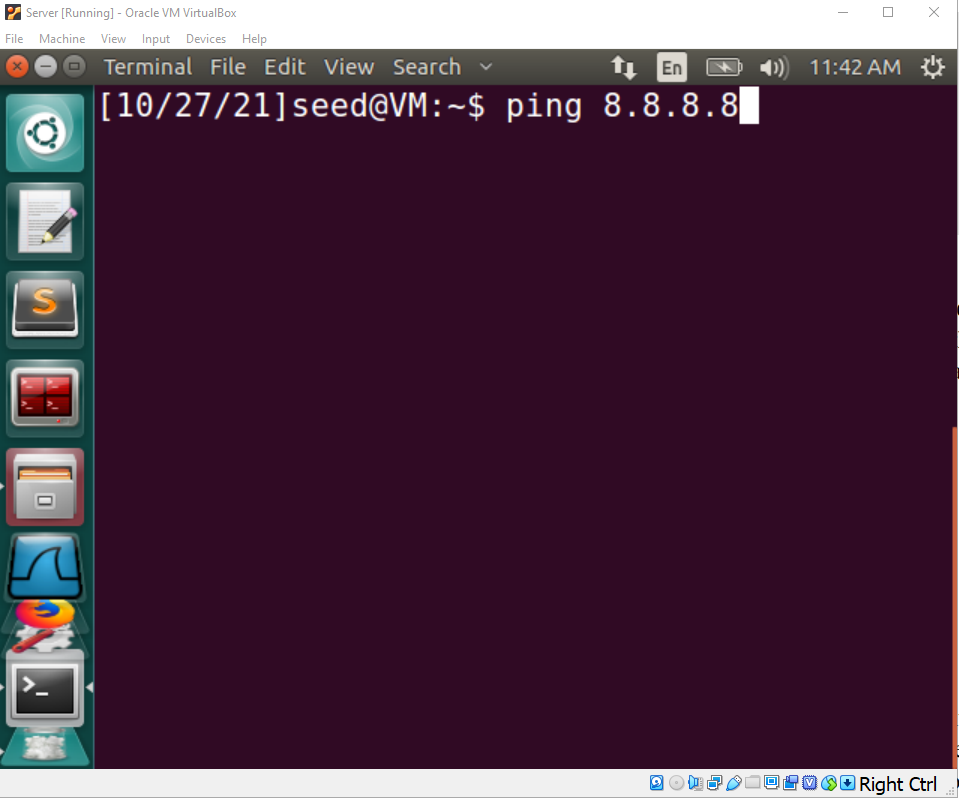
השתמשנו ב- Scapy, תיקייה בפייתון שאפשרה לנו שימוש נוח בממשק עם הרשת ובאמצעותה יכולנו להסניף פקטות ברשת.

יצרנו קובץ sniffer 1.1.a.py אשר בתוכו כתבנו תכנית בשפת פייתון, התוכנית מסניפה פקטות מסוג פרוטוקול ICMP (בדוגמה שלנו מאבחנת ניתוב בין ה-server לIP שרירותי) על ידי שימוש בפילטר ומדפיסה לנו אותן.

הרצנו את התוכנית שכתבנו ,sniffer 1.1.A.py לאחר שהפכנו את הקובץ ל (chmod a+x) Executable, והרצנו אותה עם root privilege (הרשאה גישה של superuser) (התוקף)



במכונת ה-server הרצנו פקודת Ping לשרת של גוגל. (הנתקף)



בתמונה כאן ניתן לראות כי הסניפר התחיל לתפוס את הפקטות שעברו ב-ICMP כמו שהגדרנו מראש בתכנית שלנו. ניתן לראות כי יש מענה חזרה מגוגל –

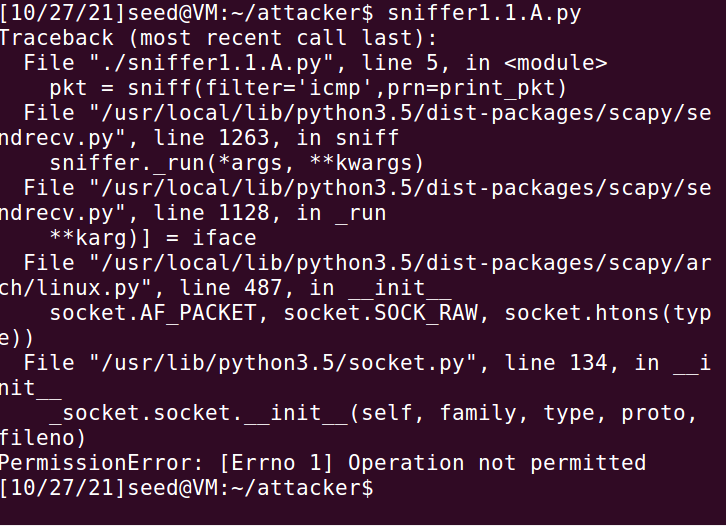
(echo-reply) – חזרה לIP של ה-server, סוג של הודעת ״האם נמצא״ ותשובה.

Text

Description automatically generated with medium confidence

כאן אפשר לראות את התיעוד של הפקטות שנשלחות הלוך וחזור מהנתקף לגוגל במכונת התוקף

לאחר מכן הרצנו פעם נוספת את התוכנית ללא הroot privilege (בלי ה-sudo), וקיבלנו שגיאה מכיוון שלא הייתה לנו הרשאה מתאימה לפעולה. (התוקף)



2.סיכום המשימה –

במעבדה זו היינו סוג של ‘man in the middle’, ממכשיר אחד שלחנו פקטות לשרת של גוגל ומהמכשיר השני הצלחנו לאתר אותם ולהוציא עליהם מידע , למשל זמן תגובה, סוג פרוטוקול , source Ip , Destination Ip. הראנו כי המכשיר תפס את הפקטות על ידי ה’echo-reply’ .

המעבדה בסך הכל תאמה לציפיות שלנו, למדנו לראשונה לכתוב תוכנית ולהשתמש בלינוקס וב – ‘VM’.

Task 1.1B

1. מבוא –

במשימה הזאת שינינו את הפילטר בפונקציית sniffing בקובץ ‘sniffer 1.1.B.py‘ וזאת על מנת לבצע מעקב אחר סוגים שונים של פקטות.

הגדרנו 3 סוגים שונים של פקטות:

* Capture only the ICMP packet. בוצע במשימה הקודמת -
* Capture any TCP packet with a destination port number 23.
* Capture packets comes from or to go to Subnet 128.230.0.0/16.

1. ביצוע המטלה –

במחשב ה-Attacker שינינו את פונקציית ההסנפה כך שיתקבלו פקטות לפי הפילטר הבא –

Capture any TCP packet that comes with a destination port number 23.

בנוסף הגדרנו את כתובת ה-IP של ה-Server.

(במכונת התוקף)

Graphical user interface, text, application

Description automatically generated

לאחר מכן הפעלנו את הפונקציה עם הפילטר המעודכן. (במכונת התוקף)

Text

Description automatically generated

במכשיר ה-Server ביצענו netcat - זאת על מנת להתחיל תקשורת בפרוטוקול TCP עם IP שהמצאנו ב-port 23. במידה והיינו שולחים פקודת Ping היינו שולחים פקטות ICMP ולא היינו יכולים להסניף אותן. (במכונת הנתקף)



בתמונה המצורפת מה-Terminal של ה-Attacker מתחת .ניתן לראות שהצלחנו להסניף פקטות של TCP/IP מכתובת ה-IP שהגדרנו מראש ועל ה-Port שהגדרנו מראש. (מכונת התוקף)

Text

Description automatically generated

מאיזה

IP

לאיזה

סוג הפקטה

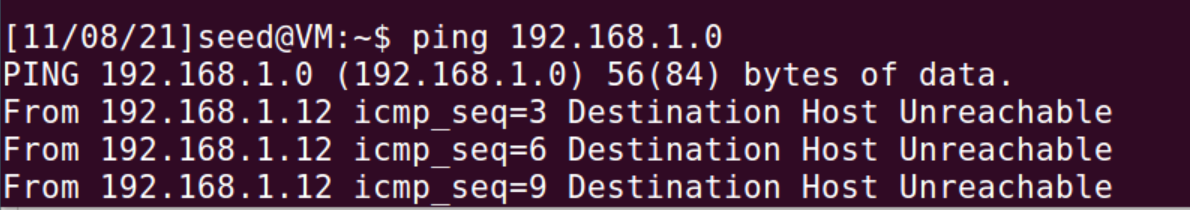
בשלב השלישי שינינו את הפילטר בפונקציה של ה-sniffing כמו שנדרש בסעיף – ‘Capture packets comes from or to go to a particular subnet. You can pick any subnet.’

לכן בחרנו כתובת subnet שרירותית (24).

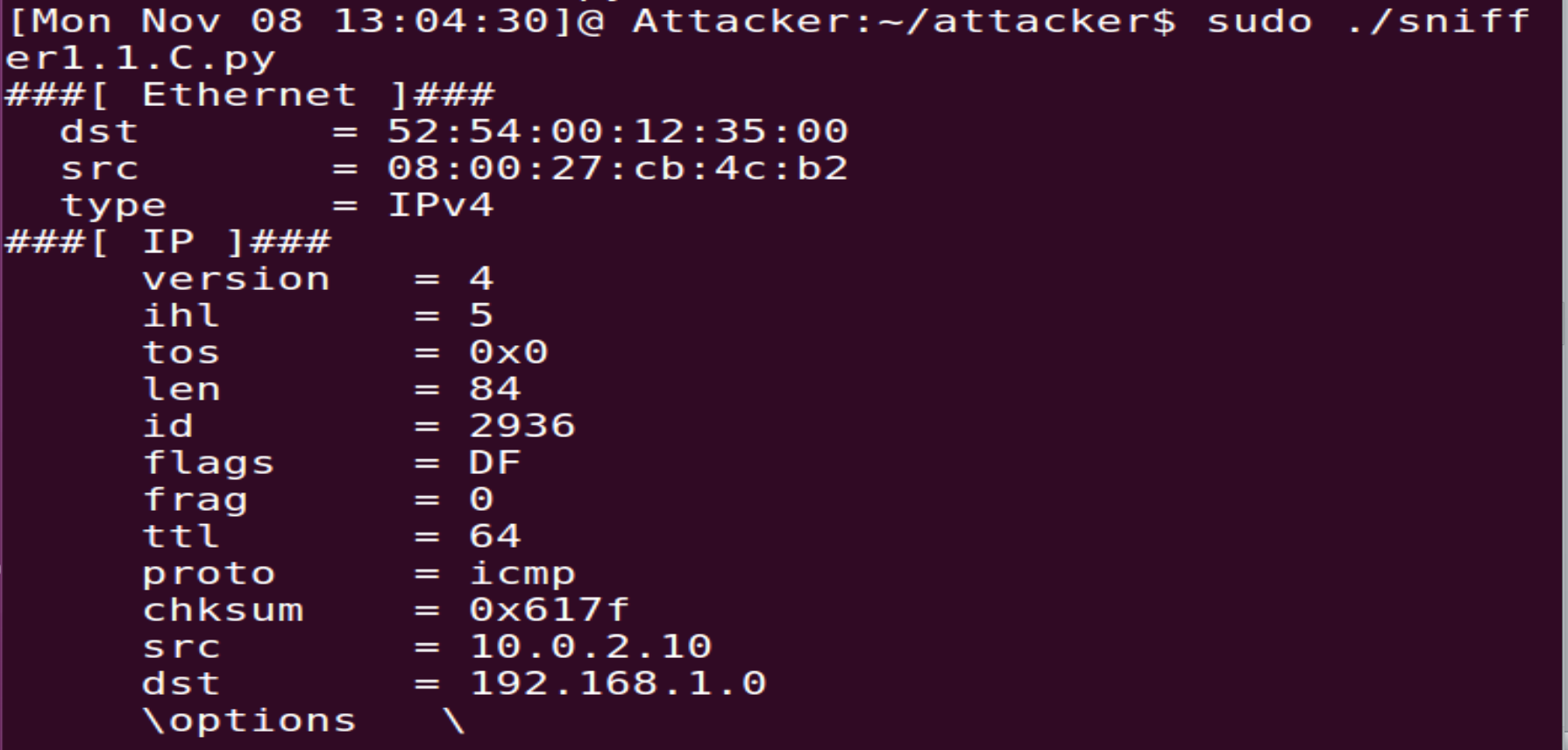


הפהרצנו שוב את פונקציית ה-sniffing שלנו אצל ה-Attacker, מיד לאחר מכן הפעלנו פעולת ping במחשב ה-server עם כתובת ה-subnet שהגדרנו על מנת להסניף את הפקטות שנשלחו לשם.

מכונת הנתקף מבצעת את הפינג



בתמונה ניתן לשים לב כי הצלחנו לתפוס ולראות את הפקטות מסוג ICMP.



**Destination Ip**

**של ה**

**PING**

מכונת התוקף מבצעת את הסניפינג

**IP**

של הנתקף

1. סיכום משימה –

במטלה זו ניסינו לתפוס סוגים שונים של פקטות על ידי שימוש בפילטרים שונים בפונקציית ה-sniffing.

הראנו את הפקטות שנתפסו בכל סעיף.

במהלך המטלה למדנו את ההבדלים בין סוגי הפקטות וסוגי התקשורת, למדנו כי אפשר לתפוס פקטות מכל מיני סוגים על ידי שימוש נכון בפילטר.

למדנו גם על ההבדל בין netcat לבין ping ובאיזה סוג תקשורת כל פקודה כזאת פועלת.

Task 1.2 Spoofing ICMP Packets

1. מבוא –

Spoofing הוא תהליך שבו התוקף מציג תעבורת תקשורת מזויפת ומתחזה למישהו אחר. הוא עושה את זה על ידי שליחת פקטות מכתובת IP שגויה, הקורבן לא מודע לכך שנתוניו נחשפו, לדוגמא כאשר נשלח אימייל מכתובת מזויפת ומבקש מהנתקף לספק מידע רגיש.

במטלה זו אנו זייפנו פקטות עם כתובת IP מומצאת ושלחנו אותן למכונת ה-server.

Text, chat or text message

Description automatically generated

1. ביצוע המשימה –

בתור התחלה יצרנו קובץ פייתון שקראנו לו ‘spoofing2.1ttl1.py’, התוכנית שכתבנו יצרה פקטה מסוג ICMP עם כתובת IP מקור מומצאת ושלחה אותה לIP של מכונת ה-server. ה-TTL מוסבר בהמשך.

Graphical user interface, application

Description automatically generated

הרצנו את תוכנית ה- spoofingממכונת ה-Attacker. ניתן לראות בתמונה כי הפקטה נשלחה. (התוקף)

Text

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated with medium confidence

IP

המומצאת על ידי מוכנת התוקף

IP

של הנתקף

ב-wireshark אנו יכולים לראות את ה-echo request מכתובת ה-IP הממוצאת 1.1.1.1, שהיא בעצם הפקטה שנשלחה כשהרצנו את ה-spoofing וכמובן את ה-echo reply שקיבלנו בחזרה ממכונת ה-server שאליו הגיעה הפקטה ששלחנו.



. 3 סיכום המטלה –

במטלה זו כתבנו תוכנית בפייתון ששולחת פקטות ממכונת ה-Attacker מכתובת IP לא אמיתית למכונת ה-server. הצלחנו לבצע את המשימה, אפשר לראות ב-wireshark, שקיבלנו echo reply בחזרה ממכונת ה-server. למדנו את עקרון הspoofing, מה זה ואיך זה עובד. לא נתקלנו בבעיות והתוכנית עבדה לנו בדיוק כמו בהוראות.

Task 1.3 Traceroute

1. מבוא –

במטלה זו אנו בדקנו את הדרך, מספר הראוטרים שבין המכונה שלנו ליעד כלשהו ע״י שימוש ב-TTL שהגדרנו מראש. בכל מעבר של פקטה בראוטר ערך ה-TTL יורד באחד. כאשר ערך ה-TTL מגיע ל-0 אנחנו מקבלים חזרה הודעת שגיאה עם כתובת ה-IP של הראוטר שבו הפקטה שלנו נזרקה.

דרך נוספת למדוד TTL היא על ידי זמן ולא על ידי מספר קפיצות.

כך בעצם אנחנו יכולים ללמוד את הדרך שהפקטה שלנו עברה עד ליעד מסוים.

1. ביצוע המטלה –

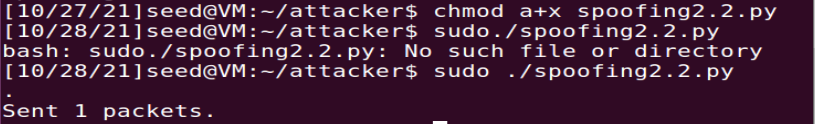
בתור התחלה יצרנו קובץ פייתון אשר קראנו לו ‘spoofing2.2‘. התוכנית שיצרנו מכינה פקטות ICMP ושולחת אותם לכתובת שאנחנו המצאנו 19.4.5.11.

בהתחלה הגדרנו את ה-ttl ל-1 ובדקנו האם הגיע ליעד הסופי שלו.

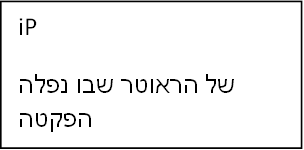
Graphical user interface, application

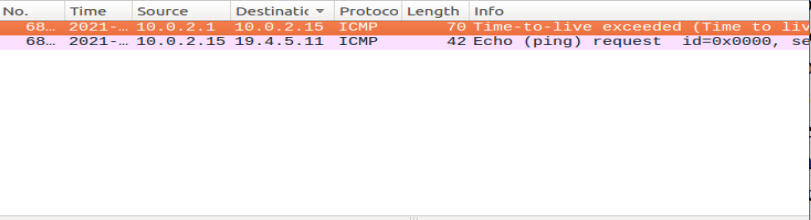
Description automatically generated

הרצנו את התוכנית וראינו שנשלחה פקטה. (ממכונת התוקף)



במקביל יכלנו לראות ב-wireshark כי הפקטה לא הספיקה להגיע לכתובת היעד הסופית שלה עם TTL=1 וקיבלנו חזרה הודעת שגיאה time to live exceeded. בנוסף להודעת השגיאה קיבלנו חזרה את כתובת ה-IP של הראוטר בו ערך ה-TTL של הפקטה הגיע ל-0.

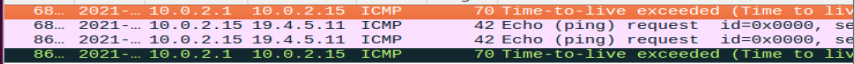




לאחר מכן העלנו את ערך ה-TTL מ-1 ל-2 והרצנו את אותה פעולה שוב.



שוב ראינו שהפקטה לא הגיעה ליעדה הסופי וקיבלנו פעם נוספת שגיאת time to live exceeded.



בניסיון הבא כבר העלנו את ערך ה-TTL ל-50 ושינינו את כתובת היעד לשרת של גוגל, עשינו שוב את אותה הרוטינה על מנת לבדוק האם הפקטה הגיעה ליעדה.



הפעם ראינו ב-wireshark כי באמת נשלחה פקטה לגוגל ע״י ‘echo request’ ואנו יודעים גם שהיא הגיעה ליעד מכיוון שהתקבלה פקטה חזרה משרת גוגל ‘echo reply’.



1. סיכום המטלה –

במטלה זו השתמשנו ב-TTL על מנת לזהות את הדרך שהפקטה שלנו עוברת ממכונת ה-Attacker ועד היעד הסופי שקבענו, על ידי שימוש נכון ב-TTL אפשר להבין בכל פעם מאיזה ראוטר נזרקה הפקטה בדרך ולקבל את כתובת ה-IP. המטלה פעלה לנו כמעט כפי שמתואר בהוראות שלה. למדנו על השימוש בTTL-, ואיך מגלים איזה דרך עוברת הפקטה מהמוצא ועד היעד הסופי שלה.

Task 1.4 Sniffing and-then Spoofing

1. מבוא –

במטלה האחרונה נדרשנו לשלב בין ה-sniffing וה-spoofing

ביצענו sniffing של פקטות ICMP שנשלחות דרך הרשת מכתובת ה-IP של מכונת ה-server שלנו.

לאחר ה-sniffing יצרנו פקטה בה הכנסנו בכתובת היעד את כתובת המקור שהופיעה בפקטה שלה עשינו sniffing. בכתובת המקור הכנסנו את היעד של הפקטה שאותה שלח ה-server. לאחר שיצרנו את הפקטה המזויפת שלחנו אותה.

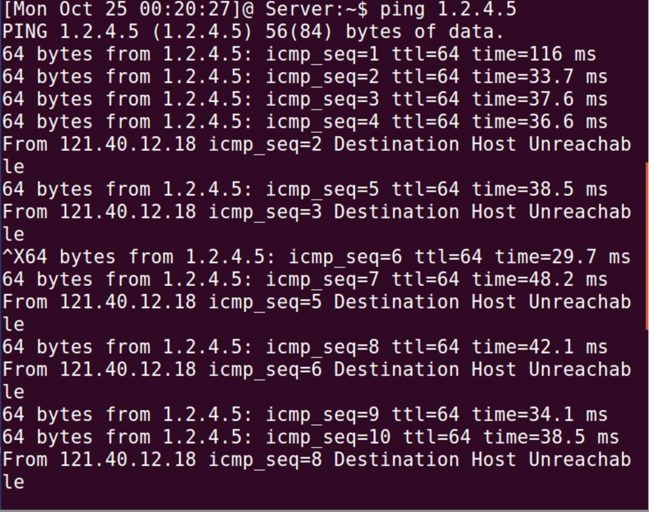
1. ביצוע המטלה –

בהתחלה יצרנו קובץ פייתון ‘sniffing\_spoofing.py’, בקובץ זה כתבנו תוכנית שתבצע sniffing, יצרנו פקטת ICMP וביצענו שינויים בכתובת המקור והיעד של הפקטה.

Text

Description automatically generated

הרצנו במקביל את התוכנית ‘sniffing\_spoofing’ ב-Attacker ופקודת ping לכתובת -1.2.4.5 במכונת ה-server. אפשר לראות כי במכונת ה-server תשובות reply שמגיעות מכתובת ה-IP שאליה שלח ה-server ping. אך התשובות הגיעו מהמכונה של ה-Attacker.



REPLY

מה

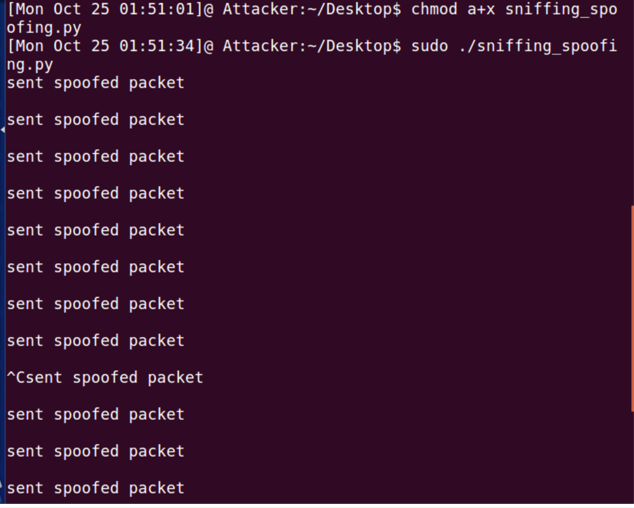
IP

שהמכונה הנתקפת שלחה

REQUEST

PING

ממוכנת הנתקף



ה

REPLY

ממכונת התוקף

לנתקף

הפעלת התוכניות ממכונת התוקף

1. סיכום משימה –

במטלה זו שילבנו בין שתי הטכניקות שנלמדו במעבדה הראשונה, ה-sniffing וה-spoofing.

ביצענו תפיסה של כתובת IP של משתמש זר ושליחה מאותה כתובת IP מה-Attacker.

ניתן להסיק כי טכניקה זו משמשת תוקפים לשם התחזות למשתמש תמים למטרות זדוניות ללא יכולת מעקב מן הצד הנתקף.

סיכום מעבדה –

במעבדה זו השתמשנו בכלים בסיסיים של תקיפה בשתי טכניקות חדשות שלמדנו ה-sniffing וה-spoofing.

ביצענו מספר מטלות שונות, בהתחלה ביצענו sniffing לסוגים שונים של פקטות מפרוטוקולים ופורטים שונים על ידי שימוש בפילטרים.

לאחר מכן ביצענו מספר מטלות spoofing, שלחנו פקטות מכתובת IP מזויפת למחשב הנתקף וגילינו כיצד ניתן לזהות את תעבורת הפקטה מראוטר לראוטר בעזרת שימוש נכון ב-TTL.

בנוסף למדנו לעבוד ולכתוב פקודות בטרמינל של Linux והשתמשנו לראשונה בWireShark שעזר לנו להבין בצורה ויזואלית את תעבורת הרשת בין המכונות שיצרנו.

במהלך העבודה נחשפנו בצורה מעמיקה יותר לפרוטוקולי הרשת כמו IP (UDP, TCP) ו-ICMP, למדנו על השימוש בPING וב-NC.

ניתן להסיק כי שימוש בכלים אלה בידי הידיים הלא נכונות עלול להוות איום ממשי ברשת, כלים מסוג sniffing ו-spoofing יכולים לאפשר לאנשים עם כישורים יחסית מועטים וללא הרבה מיומנות לפרוץ ולעקוב אחרי תעבורת רשת ולהתחזות למשתמשים לגיטימים, על ידי כך תיהיה להם האפשרות לגנוב סיסמאות ומידע פרטי אחר.

עם זאת לאחר שחקרנו מעט את הנושא, גילינו כי שימוש נכון בפרוטוקולים מאובטחים יוריד את רמת הפגיעות של משתמשים ברשת לדוגמא על ידי שימוש בTLS (אבטחת שכבת התעבורה ברשת HTTPS,FTPS), אבטחת פרוטוקולים באינטרנט (IPSec) ו-(SSH). על ידי שימוש נכון פרוטוקלים אלה מאמתים את האפליקציה או המכשיר שאליהם המשתמש מתחבר ומצפינים נתונים במעבר, באמצעות כך הם מפחיתים את הסבירות לפגיעה.