**מהו Shellcode Injection?**

Shellcode Injection, או בעברית – קוד ייעודי, הוא קוד קצר המשמש כ-Payload בניצול חולשה מבוססת זיכרון, לדוגמה: Buffer Overflow. מושג זה נקרא Shellcode מכיוון שבדרך כלל הקוד המדובר יוזם Command Shell שבאמצעותו יכול התוקף לבצע את כל העולה על רוחו בעמדת הקורבן. בדרך כלל הקוד יהיה כתוב בשפת מכונה. קוד זה יהיה מתומצת ויכיל פקודות HEX ספציפיות הנקראות opcodes.

בפרק זה נלמד מה הם סוגי מתקפות ה-Shellcode Injection וכיצד ניתן לכתוב קוד ייעודי בעצמנו.

**סוגי Shellcode Injection**

באופן כללי קוד ייעודי מתחלק לשניים – קוד ייעודי מקומי וקוד ייעודי מרוחק. נפרט אודות כל סוג:

1. **Local Shellcode – קוד ייעודי מקומי**

קוד זה משמש במצב בו לתוקף יש גישה מוגבלת לתחנת הקורבן, אך באפשרות לנצל בה חולשה כזו או אחרת בהרשאות גבוהות, למשל כמו שכבר ראינו "Buffer Overflow". Shellcode מסוג זה מאפשר לתוקף להסלים הרשאות.

1. **Remote Shellcode – קוד ייעודי מרוחק**

קוד זה משמש את התוקף כאשר מטרתו לתקוף תהליך פגיע אשר רץ במחשב אחר ברשת מהמחשב בו הוא אוחז, וכך להשיג גישה אליו ולהתקדם ברשת. בדרך כלל קוד ייעודי מרוחק יזום חיבורי TCP/IP Socket על מנת להשיג את הגישה אל היעד, אמנם תוקף יעדיף שלא ליזום חיבור חדש (מתוך החשש כי ייתקל בחומת אש אשר תחסום את התקשורת), אלא יעדיף "לרכוב" על תקשורת קיימת, אשר מוגדרת לגיטימית עבור אותה תחנת יעד.

קוד זה יסווג על פי טיב החיבור שהוא יוצר –

* אם הקוד הייעודי יוזם את התקשורת, הוא ייקרא "Reverse Shell" או "Connect-Back Shellcode", מאחר שה-Shell מתחבר חזרה לעמדת התוקף.
* אם התוקף יוזם את התקשורת מול העמדה הנתקפת, ה-Shellcode ייקרא "Bindshell" מאחר שה-Shellcode "נקשר" (מלשון Bind בשפה האנגלית) ל-port מסוים בעמדת הקורבן.
* כאשר חולשה מסוימת יוזמת את ה-connection לתהליך הפגיע בעמדת הקורבן, מדובר ב-"Socket-Reuse Shellcode". גישה זו מורכבת יותר מאחר שהקוד הייעודי נדרש להבין באיזה חיבור קיים להשתמש, ולעמדת הקורבן עשויים להיות מספר חיבורים פתוחים.

1. **Download and Execute – "הורד והרץ"**

טכניקה זו מנחה את עמדת היעד הנתקפת להוריד תכנה מן הרשת / מאתר אינטרנט מסוים, כלומר פוגען מסוים, וכך לאחר שתופעל יוכל ה-Shellcode לרוץ. שיטה נוספת הינה להוריד ספריות תכנה מהרשת. היתרון של טכניקה זו הוא שהקוד קטן יחסית, מאחר שזה לא דורש מן ה-shellcode ליצור processes חדשים בעמדת הנתקפת ואין צורך בניקוי ראיות.

1. **Staged – קוד ייעודי רב שלבי**

שיטה זו מייצגת מצב שבו הקוד הייעודי גדול מכדי להיות מוזרק לאותו process שהתוקף מעוניין לנצל בעמדת היעד. לכן הגישה של התוקף תהיה להריץ את הקוד הייעודי בשלבים – בשלב הראשון יריץ רק חלק קטן מן הקוד הייעודי, אשר יוריד חלק גדול יותר מן הקוד הייעודי אשר יוזרק לזיכרון ה-process ויורץ, וזהו למעשה השלב השני.

1. **Egg-Hunt – ביצה חבויה**

סוג זה של Shellcode הוא מעין קוד ייעודי רב שלבי. גישה זו משמשת את התוקף כאשר הוא מעוניין להזריק shellcode גדול יחסית לתוך process בעמדת הקורבן, כך שאינו יודע לקבוע היכן בזיכרון הריצה תסתיים. לכןקוד ייעודי קטן מוזרק לתוך ה-process הרצוי ומורץ. קוד זה יחפש את הכתובת של ה-process בזיכרון על מנת שהקוד הגדול יותר יוכל לרוץ (ה-"Egg").

1. **Omelette – חביתה**

Shellcode זה פועל בדומה ל-Egg-Hunt, אולם הוא מזריק לתוך ה-process מספר חבילות קוד שונות והקוד "המחפש" יידע למצוא אותן, לחבר אותן מבחינה לוגית ולהריצן יחד באותו ה-process המנוצל.

**כיצד נכתוב Shellcode?**

בכתיבת shellcode נתחשב בכמה גורמים חשובים לריצתו.

כתיבת ה- shellcode תעשה בשפת המכונה assembly מאחר וחשוב לנו שה- shellcode יהיה כמה שיותר מצומצם על מנת שלא נתקל בבעיות בעת ניצול חולשה מבוססת זיכרון. שפות תכנות אחרות בסופו של דבר מתורגמות לפקודות באסמבלר, כלומר כל פקודה בשפה תתורגם ולכן לא נרצה להגיע למצב בו הקוד יתורגם ליותר שורות ממה שצריך. זאת אומרת נשאף לכתוב את ה- shellcode במינימום פקודות.

בנוסף כתיבת קוד פולימורפי כדי להתחמק מזיהוי של ה- shellcode בעת הרצה. הכוונה לכתוב פקודות שונות שיבצעו אותו דבר וכך אם AV חתם כבר shellcode דומה נצליח לחמוק מזיהוי על ידו.

בכתיבת ה- shellcode צריך להיזהר מתווים בקוד העלולים לפגוע בתהליך הרצת ה- shellcode. התו NULL לדוגמא עלול לגרום להפסקת ריצת ה- shellcode.

בסיום כתיבת ה- shellcode נוודא שהוא כן הכי קצר שיכולנו ואם נמצא שלא נצמצם אותו ככל שנוכל על מנת לנסות ולהבטיח את רציתו.

לאחר סיום כתיבת ה- shellcode נרצה לוודא את יעילותו ע"י ניתוחו בעזרת דיסאסמבלר.

בנוסף נרצה לאמת שה- shellcode אכן פונות לפונקציות מערכת (system calls) המתאימות ונצטרך לכתוב קוד שיריץ אותו. פונקציות מערכת משמשות להסלמת הרשאות, לעצירת הקוד ועוד.

**ביבליוגרפיה**

<https://en.wikipedia.org/wiki/Shellcode>

גיליון 49 - Shellcoding 101, Digital Whisper.