Metodi bypass e come trovarli

Guida approfondita su ogni metodo detect e bypass di cui la community è a conoscenza

[ATTENZIONE]

QUESTA PARTE DELLA GUIDA É INCOMPLETA E SARÀ CONTINUATA CON IL TEMPO. PRENDETE LE INFORMAZIONI QUI TROVATE CON UN MINIMO DI SCETTICISMO E NEL CASO TROVIATE INCONGRUENZE, RIFERITEVI A ME (@bestemmies)

Avviso del 07/06/2023

In questo capitolo della guida, abbiamo un elenco accurato con ogni metodo detect e bypass mai stato usato nella community dagli inizi ad oggi. Questa parte non sarebbe stata possibile senza l'aiuto di alcuni collaboratori, tutti elencati con i crediti rispettivi nella conclusione. Questa parte della guida andrebbe visualizzata solo e soltanto se i concetti base sono stati ben assimilati. É importante perché questa parte è molto pratica e le spiegazioni non possono essere troppo lunghe, quindi è necessario che le tecniche che si vorranno usare in controllo, siano almeno capite al massimo delle loro potenzialità, ciò richiede dunque uno studio attento della teoria generale. Detto questo dividerò questo capitolo in due grandi categorie:

- Metodi detect generici
- Metodi bypass (con rispettivi detect)

Senza perderci in ulteriori argomenti, buona lettura. P.S. sono tutti metodi per sistemi Windows.

Metodi detect generici

In questa parte, troviamo metodi detect che non sono strettamente correlati a particolari metodi bypass. I detect specifici per i bypass si trovano nella stessa sezione, se esistono …. É tutto diviso in sottocategorie quindi non dovrebbe essere un problema orientarsi.

Cartelle particolari

Ci sono cartelle molto utili che noi possiamo andare a controllare per avere delle prove di utilizzo cheats, o degli artefatti da analizzare. Per viaggiare a queste cartelle è suggerito l'utilizzo dell'utility di windows, run (tasto win + r). Ecco un elenco di queste directories:

- C:\$Recycle.bin
    - É la cartella dove i file del cestino si trovano. Si utilizza per controllare l'ultima eliminazione e svuotamento del cestino, ciò si va con l'attivazione in primo luogo della visualizzazione di elementi nascosti e, per andare sul sicuro, opzioni > visualizzazione >

nascondi file protetti di sistema deve essere off come opzione nell'explorer. Fatto questo possiamo visualizzare l'ultimo svuotamento del cestino.

- Shell:recent
    - É la cartella degli oggetti recenti di windows, una funzione che ci permette di visualizzare, se attiva la stessa, alcuni file con cui windows e l'utente ha avuto a che fare. A volte ci si potevano trovare .dll etc…, ma rimane al giorno d'oggi un metodo detect deprecato.
- %temp%
    - É la cartella dei file temporanei di windows e si utilizza principalmente per trovare il file .dll che rilasciano gli autoclicker in java che utilizzano la libreria con il medesimo nome: JnativeHook. I metodi per il Jnativehook li vedremmo nella parte successiva, ma per ora ci basta sapere che viene utilizzata principalmente come detect per questo artefatto.
- Prefetch
    - Prima che qualcuno lo dica, no non è un errore scrivere sulla casella di run "prefetch", questo infatti ci apre la cartella che windows utilizza per immagazzinare i file prefetch. Vedremmo più avanti metodi bypass che vengono applicati a questa cartella e come averne le prove.

Macro e debounce time

I mouse sono un grande argomento di dibattito spesso, io stesso molte volte mi trovo a discutere con alcuni player di varie ragioni per cui le loro impostazioni sono bannabili etc… Chiaramente ho fatto un po' di ricerche quindi oggi lascio in questa sottocategoria tutte le nozioni sui mouse, il detect di varie cose etc.. Partirei proprio dall'argomento più semplice, ovvero le macro. Partendo dal fatto che programmi come Xmouse e altri simili sono bannabili (Timer resolution non rientra in questa categoria), i software proprietari dei vari mouse salvano le loro impostazioni delle macro nelle seguenti cartelle:

- Logitech
    - %localappdata%\Logitech\LGhub
- Red Dragon
    - %homepath%\Documents\M--- Gaming Mouse\MacroDB
    - I trattini indicano la possibilità di numeri diversi in base al modello del mouse
- Glorius
    - %appdata%\BY-COMBO2
- Roccat
    - // TODO

- Steel Series
    - %localappdata%\steelseriesengine-3-client\Local Storage\LevelDB
- Cooler Master
    - // TODO
- Bloody
    - // TODO
- Razer
    - // TODO
- Mars Gaming
    - // TODO

Queste erano le cartelle dei mouse fin ora analizzati. Ora vi parlo due secondi di un argomento molto scottante al momento, sto parlando del debounce time. Partiamo da qualche nozione base prima: Il debounce time è una feature dei mouse che è stata implementata, originariamente, per prevenire un difetto dei mouse: Il tasto dei mouse è essenzialmente una molla, e come tutti noi sappiamo le molle rimbalzano. Se il mouse viene cliccato in un modo particolare, questa molla rimbalza così vigorosamente che il mouse registra più click. Questo problema è noto da tempo come double clicking e sottolineo è un problema nei mouse. Minecraft è uno dei pochi giochi, se non l'unico, in cui la velocità di click fa la differenza tra vincere o perdere, perciò i player cercavano sempre più modi per cliccare più velocemente. Un giorno dei player con mouse difettosi, non disposti di DC prevent (la feature generale che implementa il debounce time in modo da evitare il double click), hanno iniziato a cliccare in modi particolari, e hanno scoperto che in questi modi il mouse registrava più click di quante volte il tasto veniva realmente premuto. Questi metodi fanno parte della categoria che oggi è conosciuta come mouse abuse. Questi metodi vengono accumunati dal fatto che abusano questo problema dei mouse, aka il double click. Il debounce time, non è altro che un timer, che quando il mouse registra un click, parte e durante il tempo impostato su questo timer (generalmente o 10ms o 16ms), il mouse blocca ogni altro input registrato. Alcuni mouse hanno la possibilità di modificare il debounce timer, in modo da abbassare il threshold e quindi aumentare la possibilità di double clicking. Solo quindi con i metodi di click mouse abuse, di cui butterfly click e drag click fanno parte, i player possono abusare di questo difetto del mouse. Venendo a noi, su alcuni server è bannabile il debounce time sotto 10ms. In questo caso per controllare che il player abbia il deobunce time a 10ms e non l'abbia modificato, si procede con un controllo simile a quello delle macro. Prima si controllano i file raw, dopo si apre il software. I mouse che possono modificare tramite software il debounce time sono i seguenti: P.S. Il debounce time è una feature che ogni manifattura di mouse implementa in modo proprietario, perciò non esiste nessuno programma generico per modificare tale funzionalità.

- Glorius
    - I mouse glorius hanno il debounce time modificabile a livello software tramite uno slider, questo implica che la modifica di BYCOMBO-2 come abbiamo visto prima per le macro è

tutto quello necessario per trovare qualsiasi modifica prima del controllo delle impostazioni

- Cooler Master
    - Stesso discorso per i glorius, i cooler master implementano il debounce time modificabile tramite slider, e tutto questo si verifica nello stesso modo delle macro
- Roccat
    - I roccat sono dei mouse che implementano il debounce time con uno switch (zero debounce time on/off). Il problema dei roccat è che sono tutti, dal primo all'ultimo, difettosi, e indipendentemente dalla funzionalità indicata, hanno un debounce time base inferiore a 10ms. Se si mette in controllo un player perché cliccava 20+ cps, ha un roccat e sul server il debounce time sotto 10ms è bannabile, allora si può direttamente bannare. Una casistica dove non andrebbe bene è quando, per esempio, metto sotto controllo uno per es. reach, e trovo una situazione simile, ma non ho visto che cliccava 20+ cps. In quel caso non è detto che stava mouse abusando, quindi la prova non può essere ritenuta valida per un ban. Massimo massimo un aggravamente ma finita li.
- Bloody
    - I mouse bloody implementano il debounce time modficabile con uno slider, ma hanno lo stesso problema dei roccat, sono tutti difettosi, e anche se lo slider è a 16ms, il debounce time vero è sotto 10ms. Stesso discorso dei roccat in quanto a motivazioni per ban etc..
- Red Dragon
    - I red dragon implementano il debounce time su alcuni mouse, tramite slider, e il metodo di controllo è il medesimo di quello delle macro
- Mad Catz
    - I mouse mad catz, per quanto rari, hanno questa peculiarità di non avere direttamente nessun tipo di debounce timer e/o double click prevent. Stesso discorso quindi per i mouse roccat e bloody, viene applicato anche qua.
- Mars Gaming
    - I mouse della mars gaming implementano il debounce time tramite slider, e il metodo di controllo è uguale a quello per le macro

Ci vuole, come in tutti i controlli e situazioni in generale, sempre un minimo di contesto e bisogna capire in primo luogo, se quella persona è da sanzionare o meno. Usate umanità quando fate controlli non siate dei robot per farmare ban.

Process Hacker

Process hacker è il tool che in assoluto è il più utilizzato dagli staffer nel tempo. Pur essendo sconsigliato basarsi principalmente su prove provenienti dalle memorie dei processi, non si può

negare che siano incredibilmente utili e se utilizzate con destrezza, possono essere un gran time saver durante un controllo, sia per lo staffer che per l'utente. Come tutte le cose ci vuole prudenza, ma in questo caso più che mai, visto che spesso il concetto di memoria volatile e cosa fa process hacker viene spesso percepito nel modo scorretto, o non capito proprio. Ipotizzando che voi abbiate letto attentamente la guida ai processi e la memoria ram, oltre alla guida dei tools e quindi sull'utilizzo di process hacker, senza altri indugi partire con l'elencare i processi e i rispettivi pattern di ricerca, con relative spiegazione al perché le facciamo in primo luogo:

explorer.exe

Il primo processo che si va a controllare, essendo quello che ha il rate di volatibilità più alta, ecco l'elenco di pattern usati:

- Parola (case insensitive) > pcaclient
    - in output ci dà più risultati a volte, ma quello che ci importa è una stringa di output che ha come contenuto, parte della memoria del pca client stesso (Program Compatibility Assistant). Questo ci da una lista degli ultimi 10 programmi eseguiti. Questo è un metodo vecchio e deprecato ma nonostante la sua inefficacia contro tutti i cheat moderni, rimane un ottimo artefatto iniziare un controllo
- Parola (case insensitive) > file:///
    - in output ci ritorna la lista di file che sono stati visualizzati dall'esplora risorse di windows. Questo è molto utile per avere una overview generale, ma bisogna stare attenti a non concentrarsi troppo su questi risultati. Rimangono pur sempre dei risultati con utilizzo deprecato, ciò significa che non sono sempre così tanto utili e spesso la gente si perde nel controllare tali risultati. Chiaramente questa ricerca ritorna una mole di file enorme, perciò su process hacker si può sempre eseguire una ricerca di qualsiasi tipo su un subset di risultati, per chi non lo sapeva.
- Regex (case insensitive) > ^[A-Z]:\\.+:
    - Questo da in output vari risultati, che bisogna analizzare attentamente, ma essenzialmente è un vecchio modo per detectare un metodo bypass (WMIC, lo vediamo dopo). Questa è una stringa deprecata e se ne sconsiglia l'utilizzo

csrss.exe

Questo è forse il processo simultaneamente più utile e inutile che c'è. Utile perché è un servizio di livello molto basso, più basso perfino di cose come svchost o altro, ed è inoltre proprio il servizio che si occupa del paradigma client - server, perciò nella sua memoria possiamo trovare ogni eseguibile mai avviato, da .dll a .exe a .pif etc .. Il problema è che essendo un servizio di livello basso, è molto spesso protetto da antivirus esterni, rendendo la sua memoria inacessibile, o ancora peggio lockato nei sistemi come windows 11 e superiori. Nonostante ciò su sistemi come windows 7 è perfettamente utilizzabile e all'interno della sua memoria con alcuni semplici filtri regex, possiamo trovare tantissime informazioni:

- Regex (case insensitive) > ^[A-Z]:\\.+.exe$

- Questo filtro trova ogni singolo path degli eseguibili .exe che sono stati avviati nel sistema

- Regex (case insensitive) > ^[A-Z]:\\.+.dll$
  - Questo filtro trova ogni singolo path degli eseguibili .dll che sono stati avviati nel sistema

- Regex (case insensitive) > ^[A-Z]:\\((?!Exe|dll|jar|ini).)*$
  - Questo filtro trova ogni singolo path degli eseguibili che hanno un estensione diversa da .exe .dll e così via, e sono stati avviati. Se qualcosa appare qua, è un ban quasi sicuro.

svchost.exe (-s dps)

Il -s dps è un servizio particolare che si fa parte dell'svchost, e tramite regex troviamo anche qui alcune informazioni utili:

- Regex (case insensitive) > ^!!(?!svchost|dwm|csrss|explorer|taskhostw|ctfmon|rundll32|conhost|lsass|usoclient|sihost|dashost|nissrv|smss|sc|servicehost|settingsynchost|consent|dllhost|sppsvc|wermgr).+.exe
  - Questo regex di dimensioni comicamente grandi, trova ogni file .exe eseguito di cui il dps tiene traccia, escludendo dalla lista i nomi dei processi che si trovano tra le parentesi.

- Regex (case insensitive) > ^!![A-Z]+(.*)[A-Z]:
  - Questo regex trova tracce dell'esecuzione di un metodo bypass (WMIC) che vedremmo più tardi. Come metodo è deprecato

- Regex (case insensitive) > ^!![A-Z](.)((?!Exe|dll).)*$
  - Questo regex trova ogni file che è stato eseguito, ma non ha l'estensione di un eseguibile (metodo bypass estensioni spoofate, le vediamo dopo)

altri processi

Questo spazio mi serve riempirlo con le vostre stringhe magiche collaboratori, non ho il full knowledge di tutte le stringhe anche quelle inutili purtroppo

Journal (fsutil)

Il modo più comune per interagire con il $USNJrnl:$J journal è senza ombra di dubbio tramite la utility cli fornita da windows fsutil. Il problema degli staffer è inseriscono i comandi, ma non sanno il loro funzionamento, o non sanno inventare comandi specifici loro stessi. Inventare pattern di ricerca per process hacker è complicato, e serve tanta pazienza, ma per il journal, tutto si limita nel filtrare per bene i vari risultati con il tool findstr.exe. Molto spesso i comandi journal che troviamo in giro sono suddivisi in questo modo:

1. fsutil usn readjournal c: csv
2. |

3. findstr /….

4. (opzionale) > output.txt

Ecco alcune nozioni sui comandi: la prima parte è un comando tutto a se, e essenzialmente quello che fa è tradurre i dati contenuti in modo compresso all'interno del file $USNJrnl:$J, in un contenuto leggibile, per l'esattezza gli specifichiamo di mettere in output, i dati sotto forma di un file csv, di cui dovremmo già sapere tanto, ma per ripassare, è essenzialmente una versione primitiva delle tabelle excel. La seconda parte del comando, è una cosa geniale, perché quella barretta è un alias di nome "pipe" e come suggerisce il nome, ogni output del primo comando viene intubato, fornito come input al comando successivo. La terza parte è sempre un comando, questa volta però filtriamo i dati del csv, con l'utility findstr.exe, fornita anche questa dal sistema. Quando facciamo il findstr, ci sono degli argomenti, come /i /R /c e così via, ecco a cosa servono:

- /i
    - indica il filtraggio dei risultati per una determinata stringa
- /R
    - indica che il filtro accade in maniera case sensitive se presente, ciò significa che ogni risultato dovrà matchare maiuscole e minuscole della stringa data come filtro.
- /C
    - indica la stringa effettiva per cui noi filtriamo. Segue infatti subito dopo un :"filtro"

Alcune volte, in comandi molto lunghi, si effettua più volte il pipe con il | e un altro findstr, che quindi filtra tutti i risultati del comando precendente. Tutto questo può terminare normalmente, o con la sintassi indicata nel punto 4. Tale sintassi indica che l'output deve finire nel file indicato dopo il simbolo maggiore (>). Li si può specificare un path relativo o assoluto, attenzione solo a dove siete nel file system con il command prompt quando fate il comando, specialmente con questa parte finale. Una cosa da sottolineare è che per questo comando, è obbligatorio avviare il cmd in modalità amministratore. Detto questo, ecco alcuni comandi già scritti, molto utili per risparmiare tempo, ma la cosa che suggerisco è inventare sul momento i comandi del journal. Fatevi furbi, poi semplificatevi la vita copiando questi comandi ma mi raccomando prendeteci destrezza.

- fsutil usn readjournal c: csv | findstr /i /C:"0x80000200" /i /C:"0x00001000" /i

  /C:"0x00002000" | findstr /i /C:.exe\^" /i /C:.pf\^" /i /C:.com\^" /i /C:.cmd\^" /i /C:".jar" /i

  /C:.pif\^" /i /C:.bat\^" /i /C:"?"

    - Questo mastodontico comando, indica ogni file di tipo eseguibile e le loro azioni, quindi eliminazione, spostamento o modifica.
- fsutil usn readjournal c: csv | findstr /i /C:"0x00000800 /i /C:"0x80000800" | findstr /i

  /C:"Prefetch"

    - Filtra risultati per un metodo bypass (CACLS), ma essenzialmente cerca modifiche di permessi alla cartella prefetch

- fsutil usn readjournal c: csv | findstr /i /C:"0x--------" /i /C:"0x--------"

    - Trova tracce dell'esecuzione del metodo bypass (WMIC). Trova delle modifiche al flusso di dati di un file.

Powershell

La history dei comandi dati al cmd non è salvata da nessuna parte, ma non si può dire lo stesso di powershell. Per trovare tutti i comandi eseguiti da powershell, basta fare il seguente comando su di essa:

- cat (get-PSReadlineoption).Historysavepath

Cmd

Alcuni comandi da riga di comando sono molto utili per avere informazioni fondamentali. Ad inizio controllo, per esempio, bisognerebbe sempre fare un veloce:

- sc query dps
- sc query sysmain

Questo comando, sc query "nome servizio", controlla lo stato del servizio, e ci indica se sono stati riavviati o no. Se uno di questi risulta stoppato o riavviato, si può facilmente bannare. L'utilizzo del comando dir è poi fondamentale per trovare metodi bypass con i permessi dei file. Per trovare ogni file nascosto, anche ai massimi livelli, basta fare nella cartella di interesse:

- dir /ar

Regedit

I registri di windows sono pieni di informazioni importanti, quindi ecco alcune chiavi di registro e raccolte di chiavi da analizzare durante un controllo

- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management\PrefetchParameters

    - Questa chiave di registro indica se il prefetch è attivo o meno (0 = inattivo)

- HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\AppCompatFlags\Compatibility Assistant\Store

    - Questa cartella indica un po' di programmi che sono stati loggati dall'utility di windows per la retro compatibilità (non indica l'esecuzione di essi, solo la loro presenza sul disco).

- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\bam\State\UserSettings\
    - Una delle detect per il metodo bypass (WMIC). Deprecato
- HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\OpenSavePidlMRU\
    - Indica gli ultimi 20 file aperti e salvati
- HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\FileExts\
    - Tutte le estensioni dei file che sono presenti sul pc
- HKEY_CURRENT_USER\SOFTWARE\WinRAR\ArcHistory
    - Tutti gli archivi che winrar ha aperto / utilizzato (Chiaramente presente solo se l'utente ha installat winrar)

Event viewer

Nel visualizzatore eventi possiamo trovare tanti eventi generici che ci indicano azioni malevole e quindi possibile ban:

- Registri windows > Sicurezza > ----
    - Indica un cambio di orario
- Registri windows > Applicazione > ----
    - Eliminazione del journal
- Registro windows > Sicurezza >-----
    - Eliminazione di eventi dall'eventvwr

altre cose

Vedete contribuitori se c'è altro da inserire (si c'è tanto altro, ecco perché siete quà)

Metodi bypass (con rispettivi detect)

Ora in questa sezione ci concentriamo nella parte dei metodi bypass, con tanto di metodi detect etc… Sapere il nome comune in community di questi metodi, oltre al loro funzionamento è fondamentale per una buona conoscenza per iniziare, e in questa sezione miriamo ad elencare ogni metodo bypass più o meno conosciuto che siamo riusciti a trovare. Ancora una volta grazie ai collaboratori, crediti alla conclusione finale.

Javaedit

I javaedit, sono delle versioni di minecraft apparentemente vanilla, ma che sono state modificate apposta in modo da contenere, per esempio, una reach integrata, difficile da individuare in primo

luogo, ma molto facile da controllare. Essendo i javaedit, molto vicini all'esperienza vanilla, essi provano in tutti i modi a fingersi una versione vanilla, ma una cosa le tradisce tutte: la SHA256. Per trovare un javaedit, bisogna confrontare la SHA256 del client utilizzato dal player, con la SHA256 della stessa identica versione che il player pretende di utilizzare. Esempio: player ha la forge-1.7.10-29174, noi dobbiamo confrontarlo con la sha 256 della forge-1.7.10-29174, non con la forge-1.7.10-27865 o la forge-1.8.9 o la vanilla 1.7.10 e così via, ci vuole la esatta versione, non una a random sennò si ottengono dei falsi positivi.

Java -jar

Tramite il comando java-jar, noi possiamo avviare qualsiasi file che ha al suo interno, del java bytecode. Questo comando ci evita di utilizzare l'explorer, cosa molto utile ma sopratutto nel prefetch è meno evidente l'esecuzione di un .jar. Il jar può essere rinominato anche con un estensione spoofata, ma l'importante è che il contenuto sia del java bytecode. Non posso avviare un vero eseguibile con java-jar, ma un 7clicker.jar rinominato tagliani.exe lo possiamo avviare. Per trovarlo, basta controllare i file prefetch del java.exe, e nei vari argomenti, controllare per file sospetti.

WMIC

Il wmic è un metodo bypass che consiste nel utilizzare due comandi e una funzionalità particolare di windows. I file possiedono diverse stream (flussi) di dati. Questi flussi possono essere creati a piacimento e possono contenere diverse informazioni. Ciò può essere un problema perché tramite questo metodo noi possiamo mettere cheat come perfino la vape all'interno di un file .txt, senza che esso sembri strano allo staffer o niente, tutti questo perché si trova all'interno di un altra stream di dati, che in primo luogo è invisibile allo staffer. Il metodo bypass sfrutta il comando type, per scrivere tutti i bytes di un file, nella stream alternativa di un altro, e il comando wmic per richiamare i bytes della stream alternativa. Questo metodo si può detectare in due modi principali:

- Analizzare il disco per file con più stream alternative (AlternativeStreamView di NirSoft)
- Controllare il journal, per la modifica anomala di data stream per alcuni file sospetti

Tutti questi due metodi sono molto efficaci, diversi in esecuzione ma pursempre efficaci. Ci sono vari metodi deprecati che includono stringhe su process hacker e noi li sconsigliamo, ma intanto sapete che esistono.

CACLS

Il cacls è una utility per modificare in modo particolare, i permessi delle cartelle e dei file. Questa utility può essere utilizzata per mettere in sola lettura la cartella prefetch e quindi far si che i file prefetch non si registrino, rimanendo però apparentemente indetectata come azione. Piccola premessa prima del detect, il cacls gioca con i permessi di sistema, rendendo il tutto estremamente pericoloso da effettuare, specialmente se non sapete cosa fate. State attenti se lo provate. Il metodo detect però non rompe il sistema operativo, e consiste nel controllare tramite il journal, la modifica di permessi di determinate cartelle, tutto alla fine relativamente semplice.

Altri metodi bypassa

Non mi vengono tutti in mente quindi mi servite voi contributors ..

Conclusioni

In questa ultima sezione della guida controlli, abbiamo visto metodi bypass e detect reali, che vengono utilizzati tutt'ora. Tutto questo faceva parte della parte di pratica, e si spera che arrivati a questa conclusione, voi abbiate una conoscenza già approfondita dei capitoli precedenti sulla teoria. Questa parte è inutile senza la teoria e le basi quindi mi raccomando. Per il resto buona lettura e buon proseguimento.