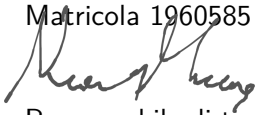
SAPIENZA
UNIVERSITÀ DI ROMA

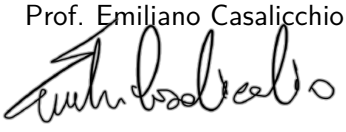
# Un Confronto di Strumenti Per L' Analisi dei Malware


Facoltà di Ingegneria dell'informazione, informatica e statistica
Laurea in Informatica

**Nicoló Madia**
Matricola 1960585



Responsabile di tirocinio
Prof. Emiliano Casalicchio




Anno Accademico 2023/2024

Tesi non ancora discussa

---

**Un Confronto di Strumenti Per L' Analisi dei Malware**
Sapienza Università di Roma



Questa tesi è stata composta con LaTeX e la classe Sapthesis.

Email dell'autore: email autore: madia.1960585@studenti.uniroma1.it

## Sommario

Lo scopo del tirocinio è stato quello di approfondire le tecniche di analisi dei malware più utilizzate dall' analisi statica a quella dinamica. Inoltre si è cercato di approfondire i vari metodi che i malware utilizzano per eludere l' analisi e la rilevazione. In seguito vengono esposti i metodi di classificazione dei malware ed il metodo d' azione di ciascuna famiglia. Scegliere uno strumento per l' analisi puo' risultare difficile, durante il tirocinio è stata effettuata un analisi delle prestazioni di vari strumenti disponibili in rete con versioni gratuite. L'obiettivo principale è stato fornire un'analisi esaustiva e critica dei vari strumenti disponibili per la rilevazione del malware, identificandone punti di forza e limitazioni.

# Indice

**1 Introduzione** **1**
1.1 Il problema della Detection dei Malware . . . . . 1
1.2 Obiettivo del tirocinio . . . . . 2
1.3 Outline . . . . . 2

**2 Background e Related work** **3**
2.1 Tecniche di analisi dei Malware . . . . . 3
2.1.1 Analisi Statica . . . . . 3
2.1.2 Analisi Dinamica . . . . . 7
2.2 Rilevazione di Malware . . . . . 13
2.2.1 Anomaly-based detection . . . . . 13
2.2.2 Specification-based detection . . . . . 14
2.2.3 Signature-based detection . . . . . 15

**3 Analisi delle prestazioni dei tool per analisi dei malware** **17**
3.1 Descrizione del problema . . . . . 17
3.1.1 Obiettivi della ricerca . . . . . 17
3.1.2 Metodologia . . . . . 17
3.1.3 Dataset . . . . . 18
3.2 Selezione dei Campioni dal Dataset . . . . . 19
3.2.1 Campioni Maligni . . . . . 19
3.2.2 Campioni Benigni . . . . . 19

**4 Strumenti per l' analisi** **21**
4.1 Panoramica degli strumenti . . . . . 21
4.1.1 Any.Run . . . . . 21
4.1.2 Comodo: Valkyrie . . . . . 22
4.1.3 Hybrid Analysis . . . . . 22
4.1.4 Joe Sandbox . . . . . 23
4.1.5 Metadefender . . . . . 23
4.1.6 Virus Total . . . . . 23
4.1.7 YARA . . . . . 24

**5 Malware e tipologie** **25**
5.1 Classificazione dei Malware . . . . . 25
5.1.1 Propagazione: Virus . . . . . 26
5.1.2 Propagazione: Worm . . . . . 28

5.1.3 Propagazione: Ingegneria Sociale . . . . . 30
5.2 Payload - Corruzione del sistema . . . . . 31
5.2.1 Distruzione dei dati e ransomware . . . . . 32
5.3 Payload - agenti di attacco . . . . . 33
5.4 Payload - furto di informazioni - keylogger, phising, spyware . . . . . 33
5.4.1 Furtro di credenziali, keylogger e spyware . . . . . 34
5.5 Payload - Stealthing - backdoor, rootkit . . . . . 34
5.5.1 Backdoor . . . . . 34
5.5.2 Rootkit . . . . . 34

**6 Esperimenti** **36**
6.1 Test degli Strumenti . . . . . 36
6.2 Risultati degli Esperimenti . . . . . 36
6.2.1 Analisi comparativa degli strumenti . . . . . 37
6.2.2 Analisi comparativa basata sulle famiglie di malware . . . . . 38

**7 Conclusioni** **46**

**Bibliografia** **47**

# Elenco delle figure

2.1 Struttura del Virus cryptato . . . . . 4
2.2 Rilevatore dei comportamenti . . . . . 14
2.3 Sequenza di byte in assembly . . . . . 15
2.4 Esempio firma in formato Yara . . . . . 16

5.1 Tassonomia dei Malware . . . . . 26

6.1 Sensibilità per ogni strumento . . . . . 37
6.2 Specificità per ogni strumento . . . . . 37
6.3 Accuratezza per ogni strumento . . . . . 38
6.4 Accuracy per Any.Run . . . . . 38
6.5 Sensitivity per Any.Run . . . . . 39
6.6 Accuracy per Comodo . . . . . 39
6.7 Sensitivity per Comodo . . . . . 40
6.8 Accuracy per Hybrid Analysis . . . . . 40
6.9 Sensitivity per Hybrid Analysis . . . . . 41
6.10 Accuracy per Joe Sandbox . . . . . 41
6.11 Sensitivity per Joe Sandbox . . . . . 42
6.12 Accuracy per Meta Defender . . . . . 42
6.13 Sensitivity per Meta Defender . . . . . 43
6.14 Accuracy per Virus Total . . . . . 43
6.15 Sensitivity per Virus Total . . . . . 44
6.16 Accuracy per Yara . . . . . 44
6.17 Sensitivity per Yara . . . . . 45

# Elenco delle tabelle

3.1 Terminologia per la valutazione delle prestazioni . . . . . 18
3.2 Composizione del Dataset . . . . . 20

4.1 Metodo e tipo di rilevamento degli strumenti di analisi dei malware . 24

# Capitolo 1

# Introduzione

## 1.1 Il problema della Detection dei Malware

Nel contesto degli ultimi anni, quasi ogni membro della società è costretto ad utilizzazare dispositivi elettronici giornalmente. Ciò è dovuto alla quasi impossibilità di svolgere qualsiasi attività senza Internet. Le interazioni sociali, le transazioni bancarie online, prenotazione di appuntamenti medici ed attività di marketing, attività che prima veniva svolte di persona, si sono spostate sulle piattaforme digitali. Con la rapida crescita di Internet, quindi, i criminali hanno iniziato a commettere reati online piuttosto che nel mondo reale, utilizzando software malevoli chiamati "malware" per lanciare attacchi informatici alle macchine delle vittime. Negli ultimi anni, gli attacchi malware hanno coinvolto ospedali, sistemi scolastici, agenzie governative locali, statali e federali, nonché altre infrastrutture critiche, tra cui i settori dell'acqua e dell'energia[(3)]. Nel 2021, gli attacchi ransomware hanno coinvolto almeno 2.323 governi locali, scuole e fornitori di assistenza sanitaria negli Stati Uniti. Secondo il World Economic Forum, gli attacchi ransomware sono aumentati del 435 percento nel 2020 e "superano la capacità delle società di prevenirli o rispondere ad essi in modo efficace". Molti di questi attacchi hanno generato perdite e danni significativi per le vittime. Un confronto triennale del numero di denunce di ransomware presentate all'FBI tra il 2018 e il 2020, dimostra un aumento del 65,7 percento nel conteggio delle vittime e un incredibile aumento del 705 percento nelle perdite corrette. Nel 2021, l'agenzia ha ricevuto 3.729 denunce di ransomware con perdite corrette di oltre 49,2 milioni di dollari.
Nel panorama sempre più complesso della cybersecurity, la rilevazione efficace dei malware rappresenta una sfida cruciale per la sicurezza dei sistemi informatici. L'aumento esponenziale delle minacce informatiche e la costante evoluzione delle tecniche utilizzate dai cybercriminali pongono in primo piano la necessità di sviluppare strumenti e metodologie in grado di identificare e contrastare queste minacce in modo tempestivo ed efficace.

In questo contesto, la ricerca e lo sviluppo di strumenti avanzati per la rilevazione dei malware riveste un'importanza fondamentale. Questi strumenti devono essere in grado di identificare in modo accurato ed efficiente le minacce informatiche, consentendo agli amministratori di sistema e agli utenti finali di adottare le misure necessarie per proteggere i propri dispositivi e reti.

## 1.2 Obiettivo del tirocinio

Lo scopo del tirocinio è stato quello di approfondire le tecniche di analisi dei malware, partendo dalle metodologie più consolidate dell'analisi statica fino ad esplorare le possibilità offerte dall'analisi dinamica. Inoltre, si è focalizzato sull'individuazione dei diversi stratagemmi adottati dai malware per eludere i sistemi di analisi e rilevazione.

Si è investigata la vasta gamma di metodi di classificazione dei malware, illustrando il modus operandi di ciascuna famiglia. Si è evidenziato il ruolo cruciale di comprendere il comportamento e le caratteristiche distintive dei malware per sviluppare strategie di difesa adeguate.

Partendo da questi presupposti è stata, infine, condotta un'analisi approfondita delle prestazioni di vari strumenti disponibili online, privilegiando le versioni gratuite. L'obiettivo principale è stato quello di fornire un'analisi esaustiva e critica dei diversi strumenti utilizzati per la rilevazione dei malware, identificandone i punti di forza, le limitazioni e le eventuali aree di miglioramento.

## 1.3 Outline

# Capitolo 2

# Background e Related work

## 2.1 Tecniche di analisi dei Malware

Le tecniche di analisi dei malware comprendono un insieme diversificato di approcci volti a comprendere il comportamento e le caratteristiche delle minacce informatiche. In generale, l'analisi dei malware può essere suddivisa in due categorie principali: **analisi statica** e **analisi dinamica**. Analizzare un sample utilizzando entrambi gli approcci é fondamentale per avere una panoramica quanto piú completa del modo in cui il malware andrá ad agire sulla macchina infetta.

### 2.1.1 Analisi Statica

L'analisi statica dei malware permette di esaminare il codice sorgente o l' eseguibile del malware senza eseguirlo effettivamente nel sistema. Questo approccio fornisce informazioni cruciali sugli attributi e sul comportamento del malware ispezionando il codice alla ricerca di particolari caratteristiche che possano suggerire un comportamento malevolo.

#### Funzionamento

Durante l'analisi statica, gli analisti esaminano il file sospetto per identificare firme uniche, chiamate di sistema particolari (API), chiavi crittografiche, frequenza di *opcode*, control flow graphs ed N-grams.

#### Vantaggi e Svantaggi

La capacità di rilevare la struttura del codice analizzato senza eseguire il programma è una strategia facile e facile e veloce per l' analisi dei malware. Tuttavia porta con se anche degli svantaggi, come discusso in [10], l' analisi statica puo‘ non rilevare codice crittografato.

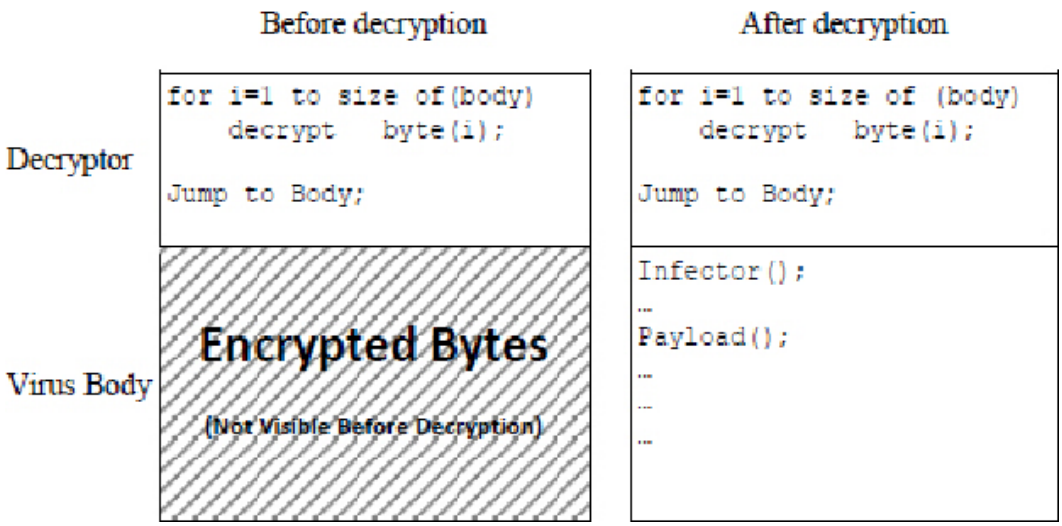



**Figura 2.1** Struttura del Virus cryptato

Crittografando il codice, infatti, è possibile eludere l' analisi statica utilizzando varie tecniche:

1. **Oligomorfismo:** Il primo virus Oligomorfico è apparso nel 1990, era un virus DOS chiamato Whale[16]. Mentre nella normale encryption, il decryptor (la chiave) rimane costante per ogni infezione, nell'oligomorfismo si utilizza una chiave univoca ad ogni replica. Nonostante questa tecnica utilizzi diverse chiavi prendendole da una lista, è comunque possibile rilevarne la presenza cercando i *match* per le suddette chiavi in un database con chiavi note.

2. **polimorfismo**: Nel 1990, il primo virus polimorfico 1260 è stato sviluppato da Mark Washburn. I virus polimorfici sono una combinazione di encryption e oligomorfismo, ma sono più complessi degli altri virus. Sono molto difficili da rilevare in un analisi statica signature based poiché cambiano aspetto con ogni copia. Inoltre siccome il numero di decryptor che possono generare non è limitato, non ha senso andare a ricercare match per le chiavi. Questo tipo di virus utilizza diverse tecniche di offuscamento per cambiare il proprio aspetto. Utilizzando un componente chiamato *Engine di Mutazione*.

3. **Metamorfismo**: I virus metamorfici non fanno parte di quelli che utilizzano la crittografia; in questa strategia, il contenuto del malware viene alterato ad ogni replica quindi non c'è bisogno di utilizzare crittografia. Implementano anche essi un engine di mutazione come nel polimorfismo, ma si va a modificare l' intera struttura del codice. L'idea di base consiste nel modificare la sintassi ad ogni nuova copia mantenendo, però, la stessa semantica, in sostanza il virus cambia apparentemente ad ogni infezione ma il significato o il funzionamento rimane lo stesso. Il primo virus metamorfico ACG è stato sviluppato nel 1998 per DOS.

### Tecniche di offuscamento

Le teniche utilizzate dai malware per renderne difficile la compresione e la lettura sono dette di offuscamento[16]:

1. **Dead-Code Insertion**: l' inserimento di codice *morto* consiste nell' aggiungere istruzioni inutili al codice, mantenendo, quindi, il suo comportamento. Un esempio è l' utilizzo di istruzioni *nop*. In ogni caso questa strategia è facilmente rilevabile.

2. **Riassegnazione dei registri**: il malware cambia il ruota il contenuto dei registri al fine di eludere la rilevazione.

3. **Riordinamento delle sub-routine**: E' possibile cambiare l' ordine di funzioni indipendenti senza modificare il comportamento del malware.

4. **Sostituzione di istruzioni**: alcune istruzioni possono essere modificate con una simile. Per esempio, *xor* può essere sostituita utilizzando *sub* e *mov* utilizzando *push/pop*.

### Tecniche Principali di analisi statica

Le tecniche principali utilizzate nell'analisi statica includono[7]:

- **Analisi delle Stringhe** Questa tecnica si focalizza sull'identificazione e sull'esame di sequenze di caratteri presenti nel codice sorgente o eseguibile del malware. Le stringhe possono assumere diverse forme, inclusi testi in chiaro, stringhe codificate o cifrate, che spesso contengono informazioni cruciali sul comportamento e sull'intenzione del malware.

  Durante l'analisi delle stringhe, gli analisti ricercano parole chiave, URL sospetti, indirizzi IP noti, o qualsiasi altra sequenza di caratteri che possa rivelare funzionalità malevole o tentativi di comunicazione con server di comando e controllo. Questa tecnica permette di individuare firme uniche associate a specifiche famiglie di malware o campagne di attacco.

  Questo tipo di analisi fornisce agli analisti un *insight* preliminare sulle caratteristiche e sugli obiettivi del malware, consentendo una categorizzazione rapida e l'identificazione di potenziali minacce. Tuttavia, è essenziale notare che gli autori di malware spesso impiegano tecniche di occultamento, cifratura o encoding delle stringhe per sfuggire all'individuazione automatica, richiedendo una continua evoluzione delle metodologie di analisi.

  Uno dei problemi di tale approccio è la necessità di bilanciare la precisione con la gestione dei falsi positivi, in quanto alcune stringhe possono essere presenti anche in software legittimi.

- **Control Flow Graph**: Un CFG è un grafo diretto che illustra il flusso di controllo di un programma, in cui blocchi di codice sono rappresentati da nodi e i percorsi del flusso di controllo da archi. Nella rilevazione di malware, il CFG può essere utilizzato per catturare il comportamento di un file PE ed estrarre la struttura del programma.

- **OPcodes**: Gli opcodes sono la prima parte di un'istruzione in linguaggio macchina (detto anche linguaggio macchina) che identifica quale operazione deve essere eseguita dalla CPU. Un'istruzione completa in linguaggio macchina è composta da opcode e, facoltativamente, uno o più operandi (e.g. "mov eax 7", "add eax ecx" e "sub ebx 1"). Gli opcode possono essere impiegati come caratteristica nella rilevazione di malware mediante il test della frequenza degli opcode o il calcolo della similarità tra sequenze di opcode diversi.

- **N-grams**:[17]Gli n-grams sono sequenze contigue di n elementi, come caratteri, parole o token, all'interno di un testo o di un insieme di dati. Questa tecnica è ampiamente utilizzata per estrarre informazioni semantiche e strutturali da documenti, codice sorgente, dataset e altri tipi di dati testuali.

  Gli n-grams forniscono una rappresentazione dettagliata della struttura linguistica e delle distribuzioni dei termini all'interno di un testo. Ad esempio, gli n-grams di parole possono rivelare la frequenza delle espressioni linguistiche e dei pattern presenti in un documento.

  Una delle applicazioni principali degli n-grams è la loro utilizzazione per la classificazione e la categorizzazione automatica di documenti. Analizzando la frequenza e la distribuzione degli n-grams, è possibile identificare le caratteristiche salienti dei documenti e distinguere tra categorie diverse.

  Nell'analisi del codice sorgente, gli n-grams di token sono preziosi per individuare pattern ricorrenti, sequenze di istruzioni o strutture di controllo. Queste informazioni sono fondamentali per identificare potenziali vulnerabilità, codice sospetto o comportamenti anomali nel codice.

- **Disassemblaggio**

  Il disassemblaggio consente la traduzione del codice macchina del malware in un linguaggio assembly leggibile dall'uomo. Questo processo si rivela utile poiche permette la visualizzazione della struttura interna del programma e ne permette la comprensione.

  Durante il disassemblaggio, il codice eseguibile del malware viene analizzato istruzione per istruzione per identificare sequenze di comandi, salti condizionali, chiamate di funzioni e altre strutture di controllo del flusso. Il codice disassemblato puó, poi, essere utilizzato per individuare pattern di comportamento sospetti, funzionalità dannose e potenziali vulnerabilità nel software.

  Tra i vantaggi del disassemblaggio vi è la possibilità di identificare e analizzare in dettaglio le tecniche di evasione, le backdoor nascoste, le vulnerabilità di sicurezza e altre caratteristiche del malware che potrebbero non essere rivelate mediante altre tecniche di analisi statica. Tuttavia, il disassemblaggio può essere un processo laborioso e richiedere competenze specializzate per interpretare correttamente il codice assembly e riconoscere pattern di comportamento malevolo.

- **Analisi delle Risorse**

  L'analisi delle risorse si concentra sull'esame delle risorse incorporate all'interno dei file eseguibili o dei pacchetti. Queste risorse possono comprendere immagini,

icone, file di configurazione, o altri elementi che contribuiscono al funzionamento complessivo del malware.

Durante l'analisi delle risorse, si esplorano le componenti multimediali e testuali integrate nel malware al fine di identificare pattern distintivi, informazioni nascoste o configurazioni cruciali per il comportamento del malware. Questa tecnica permette di ottenere una comprensione approfondita delle caratteristiche e delle funzionalità del malware, contribuendo alla creazione di firme di rilevamento e alla comprensione delle sue potenziali azioni dannose.

Infatti, le immagini o gli elementi grafici potrebbero essere utilizzati per nascondere codice dannoso, mentre i file di configurazione potrebbero contenere indirizzi IP o URL che il malware utilizza per la comunicazione con server di comando e controllo.

- **Estrazione del Payload**

  L'estrazione del payload è una tecnica di analisi statica dei malware che si concentra sull'individuazione e sull'isolamento del codice dannoso o delle funzionalità nocive all'interno del malware. Il payload rappresenta la parte del codice malware che esegue le azioni dannose o danneggia il sistema ospite.

  Si di identificare e isolare il codice eseguibile o le risorse dannose presenti all'interno del malware. Questo processo può coinvolgere l'esame dei file eseguibili, la decodifica di stringhe criptate o la ricerca di risorse incapsulate nel malware, come moduli aggiuntivi o script dannosi.

  L' estrazione del payload permette di rivelare le funzionalità dannose del malware e di consentire agli analisti di comprendere le azioni specifiche che il malware esegue sul sistema ospite. Questa informazione è cruciale per sviluppare contro-misure efficaci e proteggere i sistemi informatici dall'attività dannosa del malware. Questo metodo, però si rivela inefficace se non si hanno conoscenze di low-level programming.

- **Machine Learning**: Hashemi e Hamzeh hanno presentato un nuovo approccio che estrae opcode unici dal file eseguibile e li converte in immagini digitali. Le caratteristiche visive vengono poi estratte dall'immagine utilizzando il Local Binary Pattern (LBP), uno dei metodi di estrazione delle texture più famosi nel campo dell'elaborazione delle immagini. Infine, vengono utilizzati metodi di machine learning per rilevare il malware. La tecnica di rilevamento proposta ha ottenuto un tasso di accuratezza del 91,9% [6].

  Shaid e Maarof hanno inoltre suggerito di visualizzare il malware sotto forma di immagini. La loro tecnica cattura le chiamate API del malware e le converte in riferimenti visivi o immagini. Queste immagini vengono poi utilizzate per identificare varianti di malware [14].

### 2.1.2 Analisi Dinamica

L'analisi dinamica dei malware consiste nell'esecuzione del file sospetto in un ambiente controllato al fine di osservare il suo comportamento e le sue interazioni con il sistema ospite. Questa tecnica fornisce informazioni dettagliate sulle azioni effettive del

malware durante l'esecuzione e permette di comprendere meglio le sue funzionalità e gli obiettivi [5].

Durante l'analisi dinamica, il malware viene eseguito in un ambiente sandbox o in una macchina virtuale isolata dal resto del sistema. Si monitorano attentamente le attività del malware, inclusi i file creati o modificati, le connessioni di rete stabilite, i processi avviati e le modifiche al registro di sistema. Questo permette di identificare comportamenti sospetti o dannosi e di comprendere le tecniche utilizzate dal malware per eludere la rilevazione.

### Tecniche di analisi

Di seguito una panoramica dei metodi utilizzati per l' analisi dinamica:

1. **Analisi delle Chiamate di Funzione**: Ogni processo si basa sulle chiamate di funzione per svolgere i propri compiti, sia che queste funzioni siano interne al processo o esterne (ad esempio, funzioni esportate da altri processi, chiamate di sistema). Il comportamento del malware analizzato può essere compreso meglio tenendo traccia delle varie funzioni chiamate dal malware e dei parametri correlati a queste funzioni. Tramite un processo chiamato ***hooking***, letteralmente *aggancio*, consistente nell applicare un overhead alle funzioni, è possibile monitorarne in modo dettagliato l' esecuzione. Le tecniche di *hooking* comuni includono il meccanismo di aggancio del sistema operativo Windows (che notifica il processo di analisi quando viene chiamata una funzione), e varie tecniche di iniezione di codice. Il codice iniettato viene principalmente aggiunto all'inizio della funzione e, quando eseguito, notifica al processo di analisi che la funzione agganciata è stata chiamata. Una volta che una funzione agganciata è chiamata, il processo di analisi può quindi accedere allo stack del processo e ottenere informazioni sulla funzione chiamata e sui parametri passati ad essa. Altre informazioni utili possono essere raccolte dal sistema operativo per aumentare la comprensione del contesto in cui è stata chiamata la funzione. Vedremo in seguito come alcuni malware sono in grande di rilevare questa tecnica e quindi di non eseguire il payload se la riconoscono.

2. **Execution Control** : L' analisi dinamica dovrebbe essere in grado di incorporare meccanismi per fermare il malware in esecuzione, per controllare in che stato si trova il processo in esecuzione. Questo permette di ottenere informazioni utili sui comportamenti del malware.

   Il **Debugging** (noto anche come single stepping) è una tecnica di analisi affidabile, sviluppata originariamente per aiutare i programmatori a trovare errori nel proprio codice. Utilizzando il trap flag della CPU per generare un'interruzione dopo ogni istruzione opcode, un debugger può consentire al malware di eseguire solo un'istruzione opcode prima di forzare un cambio di contesto tornando al processo di analisi, che può quindi esaminare lo stato sia del malware che del sistema operativo. Dopo l'analisi dell'operazione, il trap flag viene nuovamente impostato e il malware continua la sua esecuzione. Questa è una tecnica molto dispendiosa in termini di risorse a causa dei frequenti cambi di contesto tra il malware e il processo di analisi. Un altro svantaggio del debugging è che è molto facile da rilevare da parte del malware.

L'**Instrumentazione Binaria** è una tecnica che aggiunge codice per analisi al codice malware originale a *run-time*. Ciò viene fatto decompilando il codice originale fino a quando si trova un opcode di trasferimento di controllo (come CALL, RET, JMP), che segnala la fine di un blocco di esecuzione. Il codice aggiuntivo viene quindi "strumentato" nel blocco, introducendo la tecnica di analisi desiderata. L'indirizzo di ritorno del blocco viene modificato in modo che ritorni al tool di analisi, per consentire l'esecuzione del blocco successivo. Il blocco strumentato risultante viene quindi eseguito. Un nuovo blocco viene ottenuto ripetendo il processo sopra descritto, a partire dal punto di esecuzione precedente.

L'**esecuzione simbolica in avanti dinamica** è utilizzata per valutare l'esecuzione del malware utilizzando input o ambienti diversi. Per bypassare qualsiasi bomba logica utilizzata dal malware, vengono creati simboli per sostituire l'input. Ad esempio, quando il malware interroga il sistema operativo sul timestamp corrente, non viene restituito un valore concreto. Invece, il simbolo creato lo sostituisce. Inizialmente, il simbolo non ha vincoli e rappresenta qualsiasi possibile valore. Ogni condizione testata contro questo simbolo crea un nuovo vincolo (o ramo). Tenere traccia dell'input e dei suoi effetti sul malware viene principalmente fatto utilizzando il tracciamento dei dati. Questa tecnica può essere utilizzata per mappare il codice malware e le condizioni necessarie per eseguire ogni sezione di codice.

In base alle condizioni testate dal malware, viene eseguito un codice diverso a seconda che le condizioni siano soddisfatte o meno. L'**esplorazione di branch multipli** è un meccanismo utilizzato per forzare l'esecuzione di tutto il codice del malware, basato sul concetto di eseguire entrambi i *branch* di una condizione selezionata. Questo metodo richiede un modo per memorizzare lo stato del sistema prima di iniziare il primo percorso (i.e. utilizzando uno snapshot come opzione) e analizzare il comportamento del malware. Una volta terminata l'esecuzione, il sistema viene ripristinato allo stato precedente e viene esplorato il secondo percorso. Questa tecnica ha diversi limiti: questo processo non può essere applicato per ogni condizione (a causa dell'aumento della complessità), potrebbe richiedere molto tempo per analizzare l'intero campione e richiede molto spazio di archiviazione per mantenere tutte le copie dello stato del sistema.

3. **Flow tracking:** *Fine Grained Analysis* viene utilizzata per tracciare il flusso delle informazioni attraverso il codice eseguito dal malware (e.g., quando il risultato di una funzione viene utilizzato come parametro per chiamare un'altra funzione). Il Data Tainting è una tecnica che "etichetta" le informazioni nella loro forma binaria. Le informazioni interessanti (come l'input dall'utente o i dati ricevuti tramite la rete) vengono etichettate (tainted) per indicare la loro origine. Quando il malware esegue un'istruzione opcode che manipola o elabora dati tainted, il taint viene applicato a anche alla regione di memoria interessata. Quando i dati tainted raggiungono un pezzo di codice o memoria predefinito (chiamato anche sink di tainting), il processo di analisi può risalire al flusso delle informazioni. Il tracciamento dei dati tainted fornisce una

profonda comprensione del modo in cui il malware interagisce con il sistema operativo e l'utente. È importante notare che la complessità dell'analisi cresce esponenzialmente man mano che più dati fluiscono nel sistema e più taints devono essere tracciati e applicati contemporaneamente. Il Data Tainting è stato originariamente utilizzato per la rilevazione delle intrusioni per fornire un meccanismo per la rilevazione degli exploit. L'etichettatura dei dati ricevuti da fonti non attendibili (come l'input da parte dell'utente) ha reso possibile rilevare l'occorrenza di buffer overflow (un'indicazione di un possibile exploit). Seguendo il percorso di tainting si può individuare dove e come l'input ha influenzato il sistema, e qualsiasi modifica effettuata a seguito dell'overflow sarà facilmente rilevata.

4. **Tracing**: La raccolta delle informazioni residue dopo l'esecuzione di determinati codici è chiamata tracciamento. Le connessioni di rete e la memoria allocata al malware lasciano delle tracce relative al comportamento del malware.

   **Analisi della memoria volatile** - Analizzare gli effetti del malware dai file di dump di memoria richiede una comprensione di come il sistema operativo tenga traccia di processi, file, utenti e configurazioni. Tutte queste strutture dati esistono nel dump di memoria in forma binaria. Ricostruire lo stato originale del sistema operativo da un dump di memoria si dice "colmare il divario semantico", e può essere fatto mediante reverse engineering della struttura del sistema operativo o utilizzando uno strumento di terze parti. Un dump di memoria fornisce una prospettiva migliore dello stato del sistema in un dato momento rispetto a qualsiasi altro layout di analisi. L'analisi della memoria volatile viene eseguita su una copia statica del sistema senza interrogare il sistema operativo sulle sue strutture interne di processi o driver. Una volta acquisito il dump di memoria, l'analisi può essere eseguita in modo sicuro e senza il timore che il malware possa manipolare il risultato dell'analisi. Oltre al codice della modalità utente, il dump di memoria contiene anche il codice kernel e tutte le sue strutture, rendendo l'analisi della memoria volatile la scelta principale per analizzare rootkit, firmware dannosi e virus di vario genere.

   **Tracciamento di rete** - Poiché nella maggior parte dei casi è richiesta una connessione Internet per consentire al malware di eseguire le proprie azioni, la natura esatta del malware potrebbe non essere rivelata senza accesso a Internet. Tuttavia, concedere al malware pieno accesso a Internet talvolta è rischioso o addirittura non possibile. Limitare l'accesso di rete da parte del malware e analizzare le connessioni di rete può rivelare il C&C del malware e i comandi ricevuti da esso. Le tracce di rete lasciate dal malware sono utili per comprendere i modelli di comunicazione che presenta.

5. **Side-channel Analysis**: Le tecniche di analisi presentate finora si basano sull'estrazione di dati dal sistema operativo, dalla memoria volatile o dallo stato della Virtual machine. Tuttavia, qualsiasi tipo di dispositivo computazionale può essere preso di mira dal malware. Tali dispositivi includono schede PCI, dispositivi IoT, hard disk, dispositivi medici e così via. L'analisi e il rilevamento dei malware in esecuzione su tali dispositivi è difficile, poiché nella maggior parte dei casi questi dispositivi non contengono un sistema

operativo che possa supportare le tecniche di analisi tradizionali. Invece di tracciare il comportamento del sistema dal punto di vista del sistema operativo (o dal livello binario), è possibile analizzare il comportamento dei componenti fisici tramite il consumo di energia, le emissioni EM o gli eventi interni della CPU. I dati acquisiti vengono divisi in "comportamento normale" e "comportamento maligno". Utilizzando metodi statistici e algoritmi di apprendimento automatico, il rilevamento di una deviazione dal comportamento normale potrebbe indicare un comportamento anomalo della CPU (ad esempio, la presenza di un criptominer o di un rootkit). L'analisi del canale laterale non può fornire conoscenze approfondite sugli eventi interni del sistema operativo, della rete o dei file modificati. Infine nessun *report* viene fornito all'utente che riceve semplicemente il verdetto finale (Maligno o meno).

Tra i vantaggi dell'analisi dinamica vi è la capacità di rilevare comportamenti dinamici del malware e di identificare azioni dannose non evidenti durante l'analisi statica. Tuttavia, l'analisi dinamica può essere più complessa e rischiosa, in quanto comporta l'esecuzione del malware in un ambiente controllato che potrebbe essere compromesso, inoltre i Malware più moderni cercano di evadere dai *sandbox* utilizzando diverse tecniche per il rilevamento dei sandbox. Quando un malware rileva un ambiente sandbox, può adattare il suo comportamento ed eseguire solamente azioni non malevole, dando quindi la falsa impressione che sia un software benigno e nascondendo il suo vero comportamento quando entrerà in contatto con il sistema target. Fra i vari metodi che il malware utilizza vi sono [12]:

1. **Stack Sbilanciata**: I sandbox spesso aggiungo un overhead di memoria nello stack per tenere traccia della azioni delle funzioni. Il malware sapendo esattamente la memoria necessaria per eseguire le funzioni, utilizzano questa informazione per accorgersi degli overhead utilizzando manipolazioni dello stack pointer.

2. **sleep skipping detection**: spesso gli ambienti sandbox non eseguono funzioni di sleep con parametri alti, il programma potrebbe crashare.

3. **Rilevazione dell' agent**: per gestire l' ambiente un agent di solito si mette in ascolto su una determinata porta. Il malware è in grado di enumerare tutte le porte inviando dati generati a posta per generare un responso che esporrebbe un agent.

4. **Presenza di artifacts dei monitor**: I malware possono sondare il sistema alla ricerca di artefatti generati dai monitor, file utilizzati per le configurazione e la gestione del sandbox.

5. **Raw firmaware Table detection** : Una "Raw Firmware Table" (Tabella Firmware Grezza) è una struttura dati utilizzata nei sistemi embedded e nei dispositivi hardware per memorizzare informazioni critiche sul firmware, come le informazioni di configurazione, i parametri di avvio, le tabelle di partizione e altre informazioni necessarie per l'avvio e il funzionamento del dispositivo.

   Questa tabella è chiamata "grezza" perché contiene i dati in un formato non elaborato o non interpretato, il che significa che le informazioni al suo

interno potrebbero essere codificate in un formato specifico del dispositivo o del firmware, non ancora processate o interpretate dal sistema operativo o dall'applicazione. I malware possono ricercare l' interno di queste tabelle cercando stringhe associate alla virtualizzazione di ambienti.

6. **Debugger**: La presenza di debugger è un altro indicatore che il file è sottoanalisi (e.g. utilizzando l'istruzione opcode PUSHF, che spinge il valore del trap flag nello stack). Controllando se il flag di debug è impostato su uno, il malware può rilevare se è sotto analisi e nascondere la sua attività dannosa).

7. **Tracce dell' attività utente**: Quando un malware infetta un *personal computer*, è in grado di verificare se un utente ha usato il sistema da un pò di tempo siccome l' uso frequente di un sistema crea tracce. Spesso gli ambienti di sandbox effettuano un *fresh start* del sistema eliminando, effittivamente, qualsiasi traccia di utilizzo di un utente reale. Fra le varie tracce che il malware cerca troviamo la cronologia internet, un installazione pulita, controllare il numero di schermi connessi, se la finestra dei processi è in focus, il movimento del mouse e tastiera.

Per nascondersi dall' analisi, alcuni autori di malware utilizzano un approccio **low level** utilizzando funzioni kernel non documentate, si tratta di funzioni sviluppate per il kernel Windows ma che possono essere invocate anche in user mode. Le funzioni kernel sono progettate per essere usate dal sistema operativo, infatti, spesso non sono documentate. Un altro metodo per evitare l' analisi è quello di manipolare direttamente i registri di sistema senza chiamare le API che di solito si utilizzano per farlo. Questo richiede una profonda conoscenza di come il sistema operativo gestisce i dati. In ogni caso una strategia del genere, sebbene molto difficile, se applicata correttamente è molto difficile da rilevare.

In aggiunta vi sono vari metodi che gli attaccanti utilizzano per evadere al di fuori dei tool di analisi:

1. **code injection**: per evitare di essere rilevato, il malware può eseguire codice utilizzando altri processi. Inserendo codice malevolo all'interno di altri processi, il malware può eseguire il suo payload come se facesse parte di un processo leggittimo. La maggior parte dei tool di rilevazione contengono una lista di processi *sicuri* e quindi presenti su una whitelist. Il malware cerca di eseguire l' injection su uno di questi processi al fine di eludere il rilevamento. Alcuni esempi sono: (1) Iniezione di DLL: utilizzando l'API di Windows, il malware può allocare memoria per conto di un altro processo e iniettarlo con un file DLL precompilato. Questo file DLL può avviare la sua esecuzione come un nuovo thread all'interno del processo remoto, e ogni istruzione eseguita dal DLL farà parte del processo attaccato. Questa tecnica viene anche utilizzata per ottenere privilegi più elevati.(2) Dirottamento dell'ordine di ricerca delle DLL: quando un processo tenta di caricare un file DLL per nome, il sistema operativo cerca quel file in posizioni specifiche (come la cartella del processo, c:/windows/system32/ e altre). Posizionare una DLL dannosa con lo stesso nome in una di queste posizioni può costringere il sistema operativo a caricare un codice dannoso invece della DLL effettiva.(3) Scavatura dei processi: una

tecnica che prende il controllo di un intero processo cancellando la mappa del codice originale dalla memoria e iniettando un codice dannoso nel processo "scavato". La struttura del processo originale rimane intatta, il che significa che l'identità e tutti i permessi del processo sono di proprietà del codice dannoso. Altre tecniche per l'iniezione di codice includono la bomba atomica, il dirottamento dell'esecuzione del thread e il dirottamento di COM.

2. **Infezione del host**: Se viene trovata una vulnerabilità nel sistema che permette al malware di evadere dal guestOS ed eseguire istruzioni sull host potrebbero esserci gravi conseguenze. Chiaramente stiamo parlando dello scenario peggiore.

3. **Nested virtualization**: A volte il malware crea una macchina virtuale all'interno di un sistema infetto ed esegue il codice dannoso all'interno della VM, al di là della distinzione semantica. Questo consente al malware di sfuggire alla rilevazione e all'analisi.

## 2.2 Rilevazione di Malware

Le tecniche per rilevare la presenza di software malevolo in un sistema possono essere divise in due grandi categorie[8]: **Anomaly-based detection** e **Signature-based detection**. La rilevazione tramite anomalie utilizza la conoscenza dei comportamenti considerati *normali* per rilevare comportamenti che deviano, appunto, da tale normalità. Un tipo particolare di Anomaly-based detection é chiamato **Specification-based detection**.

### 2.2.1 Anomaly-based detection

Di solito il rilevamento basato sulle anomalie avviene in due fasi: una fase di *addestramento* (apprendimento) ed una fase di *rilevamento* (monitoraggio). Durante la fase di apprendimento il sistema impara a riconoscere i comportamenti considerati *normali*, per poi andare a monitorare il sistema e, quindi, rilevare ogni tipo di comportamento **anomalo** rispetto a quelli osservati nella fase di apprendimento. Un vantaggio chiave del rilevamento basato sulle anomalie è la sua capacità di rilevare attacchi zero-day. Similmente alle vulnerabilità zero-day, gli **attacchi zero-day** sono attacchi precedentemente sconosciuti al rilevatore di malware.

Il rilevatore di comportamenti utilizzato nella tecnica basata sulle anomalie è composto dai seguenti tre componenti di base[7]:

1. **Raccolta dati** : Come suggerisce il nome, questo componente si occupa della raccolta dei dati, statici o dinamici.

2. **Interpretazione**: Questo componente interpreta i dati raccolti dal componente di raccolta dati e li converte in una forma intermedia.

3. **Algoritmo di corrispondenza:** Questo componente viene utilizzato per confrontare la firma comportamentale con le informazioni convertite nel componente di interpretazione. Il rilevatore di comportamenti è mostrato nella Figura

2.2, che spiega il funzionamento di come tutti questi componenti lavorano insieme.

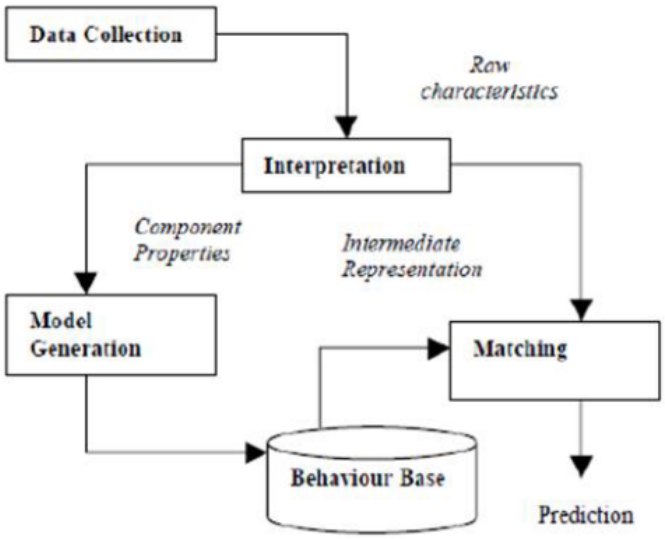



**Figura 2.2** Rilevatore dei comportamenti

Le due limitazioni fondamentali di questa tecnica sono il suo alto tasso di **falsi allarmi** e la complessità nel determinare quali caratteristiche dovrebbero essere apprese nella fase di addestramento. Poiché nella fase di addestramento si fa un approssimazione dei comportamenti *validi*, un comportamento valido potrebbe essere segnalato come malevolo per il semplice fatto che non é stato osservato nella fase di addestramento.

Questo contribuisce all'alto tasso di falsi positivi comunemente associato alle tecniche di rilevamento basate sulle anomalie. Lo sviluppo di migliori approssimazioni al comportamento *normale* di un sistema informatico è un problema aperto nella scienza informatica.

### 2.2.2 Specification-based detection

Il rilevamento basato sulle specifiche è un tipo di rilevamento basato sulle anomalie che cerca di risolvere il tipico alto tasso di falsi allarmi associato alla maggior parte delle tecniche di rilevamento basate sull'anomalia. Invece di tentare di approssimare l'implementazione di un'applicazione o di un sistema, il rilevamento basato sulla specifica cerca di approssimare i requisiti per un'applicazione o un sistema. Nel rilevamento basato sulla specifica, la fase di addestramento consiste nell'ottenere un insieme di regole, che specifica tutto il comportamento valido che qualsiasi programma può esibire per il sistema in fase di protezione o il programma in fase di ispezione. La principale limitazione del rilevamento basato sulla specifica è che spesso è difficile specificare completamente e accuratamente l'intero insieme di comportamenti validi che un sistema può esibire. Si può immaginare che anche per un

sistema moderatamente complesso, specificare completamente i set di comportamenti *validi* diventa un *task* estremamente complesso e laborioso.

### 2.2.3 Signature-based detection

I malware come detto in precedenza hanno delle caratterische utili per generare delle firme digitali uniche. I provider di tool antimalware utilizzano algoritmi meta-euristici can riescono ad analizzare in modo efficiente il file in questione estraendone la firma e, poi, determinandone la natura malevola o benigna confrontandole con un database di firme note[13].

Dopo aver identificato l'oggetto dannoso, la firma rilevata viene aggiunta al database esistente come malware noto. Le *entry* del database includono un gran numero di firme diverse che classificano i file dannosi. Nella *Signature-based detection*, ci sono numerosi vantaggi, tra cui una rapida identificazione e facilità di esecuzione.[11].

Come detto in precedenza questo tipo di approccio non riesce a identificare efficacemente i malware polimorfici. Di conseguenza, la rilevazione basata sulla firma non fornisce protezione da attacchi *zero-day*. Inoltre, poiché si ha bisogno di una firma separata per ogni variante di malware, il database delle firme cresce in modo esponenziale.

#### processo di generazione delle firme

Durante la generazione delle firme, prima vengono estratte le caratteristiche dagli eseguibili. Successivamente, il motore di generazione delle firme genera le firme e le memorizza nel database. Quando un programma un sample deve essere contrassegnato come malware o benigno, la firma del sample correlato viene estratta allo stesso modo di prima e confrontata con le firme presenti nel database. Sulla base del confronto, il sample viene contrassegnato come malware o benigno. Esistono molte tecniche diverse per creare una firma come la scansione delle stringhe, la scansione iniziale e finale, la scansione del punto di ingresso e il controllo dell'integrità[4].

1. **Scansione delle stringhe**: Confronta la sequenza di byte nel file analizzato con le sequenze di byte precedentemente salvate nel database. Le firme di byte sono state ampiamente utilizzate dagli scanner antivirus per molti anni. Sono spesso utilizzate per rilevare malware che appartengono alla stessa famiglia con firme diverse. Ad esempio, '90FF1683EE0483EB0175F6' è la rappresentazione esadecimale di una sezione di codice mostrata nel linguaggio assembly nella figura 2.3.Nella figura 2.4 è mostrata la firma in byte nel formato Yara.

```
Start: 0x401A2E length: 0xC
90 nop
FF 16 call dword ptr [esi]
83 EE 04 sub esi, 4
83 EB 01 sub ebx, 1
75 F6 jnz short loc_401A30
```

**Figura 2.3** Sequenza di byte in assembly

```
Rules
{
   strings:
   signature = {66 90 FF 16 83 EE 04 83 EB 01 75 F6}
   condition:
   signature
}
```

**Figura 2.4** Esempio firma in formato Yara

2. **Scansione iniziale e finale**: Invece dell'intero file, vengono presi solo i punti iniziali e finali del file e vengono create le firme. È un metodo di firma molto comodo e veloce per rilevare virus che si attaccano all'inizio e alla fine dei file.

3. **Scansione del punto di ingresso:** Il punto di ingresso di un file indica da dove inizia l' esecuzione dello stesso. I malware possono cambiare il punto di ingresso di un programma, in modo da eseguire il loro payload prima del codice legittimo. Pertanto, alcuni malware possono essere rilevati estraendo la firma dalle sequenze ai punti di ingresso del programma.

4. **Controllo dell'integrità (Firme di hash):** Il controllo dell'integrità genera un checksum crittografico come MD5 e SHA-256 per ciascun file in un sistema a intervalli regolari ed è utilizzato per identificare possibili modifiche che possono essere causate dai malware.

# Capitolo 3

# Analisi delle prestazioni dei tool per analisi dei malware

## 3.1 Descrizione del problema

Il problema specifico affrontato nel tirocinio riguarda l'analisi delle prestazioni degli strumenti e delle tecniche esistenti per il rilevamento dei malware. Sebbene siano disponibili numerosi strumenti e approcci per il rilevamento dei malware, la loro efficacia varia in base alle caratteristiche dei malware e alle condizioni ambientali. È pertanto essenziale valutare le prestazioni di questi strumenti per identificare quelli più efficaci e adatti a fronteggiare le minacce attuali.

### 3.1.1 Obiettivi della ricerca

L'obiettivo principale della ricerca è condurre un'**analisi comparativa** delle prestazioni dei principali strumenti di rilevamento dei malware. Questo implica la valutazione delle capacità di rilevamento, la precisione e l'affidabilità dei vari strumenti nell'identificare.

### 3.1.2 Metodologia

Per raggiungere l' obiettivo sono stati presi in esame vari tool fra i più popolari in grado di eseguire un analisi automatica dei sample. Successivamente, ogni tool è stato preso in esame singolarmente e, quindi, eseguito sul dataset selezionato.

Al fine di valutare nel modo quanto più completo ed oggettivo possibile, sono state considerate delle metriche che forniscono un quadro completo delle prestazioni dei tool di rilevamento dei malware, consentendo una valutazione accurata e obiettiva della loro efficacia e affidabilità nel rilevare minacce informatiche.

#### Metriche per il Confronto

Le metriche utilizzate sono basate su un lavoro precedentemente eseguito nello stesso ambito[15]:

1. **Sensibilità**: La sensibilità indica la capacità di identificare correttamente una condizione. Maggiore è il valore della sensibilità, migliori sono le prestazioni

degli strumenti. La sensibilità può essere definita come:

$$Sensibilità = \frac{\text{Numero di TP}}{\text{Numero di TP} + \text{Numero di FN}}$$

| Terminologia | Descrizione |
|---|---|
| True Positive (TP) | Correttamente identificato |
| True Negative (TN) | Erroneamente identificato |
| False Positive (FP) | Correttamente respinto |
| False Negative (FN) | Erroneamente respinto |

**Tabella 3.1** Terminologia per la valutazione delle prestazioni

2. **Specificità**: La specificità è il concetto di testare la capacità di escludere correttamente una condizione. Minore è il valore della specificità, migliori saranno le prestazioni degli strumenti. Può essere espressa come:

$$Specificità = \frac{\text{Numero di TN}}{\text{Numero di TN} + \text{Numero di FP}}$$

3. **Accuratezza**: Questa metrica rappresenta la proporzione dei risultati veri (sia veri positivi che veri negativi) nel campione dato. Le prestazioni degli strumenti saranno eccellenti se il valore della precisione sarà alto. Può essere espressa come:

$$Accuratezza = \frac{\text{Numero di TP} + \text{Numero di TN}}{\text{Numero di TP} + \text{Numero di TN} + \text{Numero di FP} + \text{Numero di FN}}$$

Dopo aver raccolto i dati relativi ai singoli strumenti, è stato necessario elaborarli per ottenere le informazioni utili al calcolo delle metriche. Per raggiungere questo fine si è reso necessario l' utilizzo di script python per tradurre i risultati, che sono stati spesso in output differenti per ogni tool, ad un formato unico che fosse poi utile per rielaborare i dati. Ho scelto di tradurre tutto in un semplice file *csv*, acronimo di "Comma-Separated Values" (valori separati da virgole), è un formato di file molto comune utilizzato per archiviare e trasferire dati tabellari in modo semplice e leggibile. Un file CSV è essenzialmente un file di testo in cui le righe rappresentano le righe della tabella e i valori all'interno di ciascuna riga sono separati da un carattere delimitatore, spesso una virgola (,), un punto e virgola (;), o una tabulazione.

Successivamente ho utilizzato la potente libreria di manipolazione dei dati **pandas** insieme a **matplolib** per calcolare e generare i grafici che permettono di visualizzare ed analizzare più facilmente le performance di ciascun tool.

### 3.1.3 Dataset

Per condurre un'analisi accurata delle prestazioni degli strumenti di rilevamento dei malware, è fondamentale utilizzare un dataset rappresentativo e diversificato di campioni di malware. In questo contesto, ho utilizzato DikeDataset [1], un dataset

ampiamente utilizzato in vari articoli accademici. Tale dataset è stato ottenuto utilizzando l' API di **MalwareBaazar** una piattaforma online che funge da repository pubblico per campioni di malware e informazioni associate. È gestita da abuse.ch, un'organizzazione senza scopo di lucro che si concentra sulla raccolta e la condivisione di informazioni relative alla sicurezza informatica. MalwareBazaar offre un vasto archivio di campioni di malware provenienti da una varietà di fonti. Questi campioni di malware vengono raccolti, analizzati e resi disponibili al pubblico per scopi di ricerca, analisi forense e difesa informatica.

Campione di Malware: Ogni campione di malware è accompagnato da informazioni dettagliate, come il nome del file, l'hash MD5, l'hash SHA256, la data del caricamento, il tipo di malware e altre informazioni pertinenti.

## 3.2 Selezione dei Campioni dal Dataset

Per condurre un'analisi esaustiva delle prestazioni degli strumenti di rilevamento dei malware, è stata effettuata una selezione oculata dei campioni dal dataset di riferimento. Il dataset utilizzato contiene un vasto assortimento di malware e file benigni, ciascuno con caratteristiche uniche e variazioni all'interno delle rispettive famiglie. DikeDataset è composto da un insieme di file PE e OLE, suddivisi in due categorie principali: benigni e maligni. Questi file sono stati etichettati in base alle loro caratteristiche e comportamenti.

### 3.2.1 Campioni Maligni

Dal dataset sono stati selezionati un totale di 450 campioni di malware per la valutazione delle prestazioni degli strumenti di rilevamento. Questi campioni sono stati scelti in modo da coprire una vasta gamma di famiglie di malware e varianti, garantendo una rappresentazione significativa delle minacce potenziali affrontate dagli utenti finali. In particolare, sono stati estratti 50 campioni per ciascuna delle famiglie di malware esaminate, includendo backdoor, downloader, encrypter, ransomware, rootkit, spyware, trojan, worm e generic (tipologia che non presenta caratteristiche di una sola famiglia di malware).

### 3.2.2 Campioni Benigni

Parallelamente ai campioni di malware, sono stati selezionati anche 300 file benigni per la valutazione comparativa degli strumenti di rilevamento. Questi file benigni rappresentano applicazioni e file legittimi che potrebbero essere comunemente presenti nei sistemi informatici degli utenti, senza alcuna componente dannosa o comportamento sospetto.

La selezione attenta di campioni di malware e file benigni garantisce un'analisi completa e bilanciata delle capacità di rilevamento degli strumenti, consentendo di valutare in modo accurato la loro capacità di distinguere tra minacce reali e file innocui.

**Tabella 3.2** Composizione del Dataset

| **Tipo di File** | **Numero di File** |
|---|---|
| Benigni | 300 |
| Maligni (backdoor) | 50 |
| Maligni (downloader) | 50 |
| Maligni (encrypter) | 50 |
| Maligni (ransomware) | 50 |
| Maligni (rootkit) | 50 |
| Maligni (spyware) | 50 |
| Maligni (trojan) | 50 |
| Maligni (worm) | 50 |
| Maligni (generic) | 50 |
| **Totale** | 750 |

# Capitolo 4

# Strumenti per l' analisi

In questa ricerca sono stati utilizzati sette strumenti per l'analisi del malware: Any.Run, Comodo, Hybrid Analysis, Joe Sandbox, Metadefender, VirusTotal e Yara (tab. 4.1). La selezione di questi strumenti è stata effettuata in base a diverse considerazioni:

- **Funzionalità**: Offrono una combinazione di funzionalità di analisi statica e dinamica per una valutazione completa del malware.
- **Reputazione**: Sono sviluppati da organizzazioni affidabili e ampiamente utilizzati nella comunità della sicurezza informatica.
- **Accessibilità**: Sono disponibili gratuitamente o con versioni gratuite con funzionalità sufficienti per la ricerca.

## 4.1 Panoramica degli strumenti

Di seguito si descrivono gli strumenti nel dettaglio analizzando le loro funzionalità e punti di forza.

### 4.1.1 Any.Run

Any.Run è una piattaforma sofisticata e potente progettata per l'analisi approfondita del malware. Offre agli utenti la possibilità di eseguire file sospetti in un ambiente virtuale sicuro e controllato, fornendo un'ampia gamma di strumenti e funzionalità per comprendere il comportamento e l'impatto del malware.

Una delle caratteristiche distintive di Any.Run è la sua capacità di eseguire i file sospetti all'interno di un ambiente virtuale isolato anche detto "Sandbox". Questo ambiente protetto protegge il sistema dell'utente da danni potenziali causati dal malware, consentendo agli analisti di esplorare in modo sicuro le minacce senza compromettere la sicurezza del proprio sistema.

Durante l'esecuzione del malware, Any.Run monitora attentamente il suo comportamento. Registra ogni azione eseguita dal malware, inclusa la creazione di file, le modifiche al registro di sistema, le connessioni di rete e altro ancora, fornendo informazioni preziose che permettono una comprensione sulla natura e l' origine delle minacce in esame.

Any.Run raccoglie una vasta gamma di informazioni sul malware in esame. Queste informazioni includono stringhe rilevanti, URL utilizzati dal malware e hash univoci che possono essere utilizzati per identificare e classificare il malware. La raccolta di queste informazioni aiuta gli analisti a comprendere meglio la natura e l'origine delle minacce.

Al termine dell'analisi, Any.Run genera report dettagliati che riassumono i risultati dell'analisi del malware. Questi report contengono informazioni essenziali, tra cui screenshot delle attività del malware, dump di memoria, analisi del codice e altro ancora. I report facilitano la comprensione del comportamento del malware e forniscono agli utenti una base solida per prendere decisioni informate sulla sicurezza.

### 4.1.2 Comodo: Valkyrie

Valkyrie è una potente piattaforma di analisi del malware sviluppata per consentire agli utenti di eseguire e analizzare file sospetti in un ambiente sicuro e controllato. Con una vasta gamma di funzionalità e strumenti avanzati, Valkyrie offre agli utenti la possibilità di comprendere in dettaglio il comportamento e l'impatto del malware sul sistema, facilitando l'identificazione e la mitigazione delle minacce.

Una delle caratteristiche distintive di Valkyrie è la sua capacità di condurre un'analisi approfondita del comportamento del malware durante l'esecuzione. La piattaforma monitora attentamente le attività del malware, inclusa la creazione di file, le modifiche al registro di sistema, le connessioni di rete e altro ancora. Questo monitoraggio dettagliato fornisce agli utenti informazioni preziose sulle azioni e le intenzioni del malware.

Valkyrie utilizza un motore avanzato di rilevamento delle minacce per identificare e classificare i file sospetti. Utilizzando una combinazione di tecniche euristico-comportamentali e analisi statica, Valkyrie è in grado di individuare una vasta gamma di malware.

Al termine dell'analisi, Valkyrie genera report dettagliati che riassumono i risultati dell'analisi del malware. Questi report contengono informazioni essenziali sul comportamento del malware, inclusi screenshot delle attività, informazioni sui processi e altro ancora. I report di Valkyrie forniscono agli utenti una panoramica completa del malware, facilitando la comprensione e la mitigazione delle minacce.

Valkyrie offre anche integrazioni con una varietà di strumenti di sicurezza e piattaforme di gestione delle minacce. Queste integrazioni consentono agli utenti di automatizzare il processo di analisi del malware e di integrare i risultati dell'analisi direttamente nei loro flussi di lavoro di sicurezza esistenti.

### 4.1.3 Hybrid Analysis

Hybrid Analysis si presenta come una piattaforma all'avanguardia nel campo dell'analisi del malware, unendo potenza e facilità d'uso in un'unica soluzione.

Hybrid Analysis adotta un approccio combinato di analisi statica e dinamica per offrire una visione completa del comportamento del malware. L'analisi statica esamina il codice del malware per individuare caratteristiche sospette, mentre l'analisi dinamica esegue il file in un ambiente virtuale sicuro per monitorarne le azioni.

La piattaforma estrae informazioni rilevanti dal malware, come stringhe, URL e hash, che possono essere utilizzate per identificare e classificare le minacce. Hybrid Analysis effettua la ricerca di Indicatori di Compromissione (IOC) noti, come hash di file, URL e indirizzi IP, per individuare possibili minacce e correlarle con database di riferimento. Dopo l'analisi, Hybrid Analysis genera report dettagliati che sintetizzano i risultati ottenuti. Questi report includono screenshot, dump di memoria e analisi del codice per offrire una visione approfondita del comportamento del malware.

### 4.1.4 Joe Sandbox

Joe Sandbox è un potente strumento di analisi del malware che consente agli utenti di eseguire file sospetti in un ambiente sicuro e virtuale, attraverso una combinazione di tecniche avanzate, dall'analisi statica alla dinamica, fino all'approccio ibrido e basato su grafi. Joe Sandbox, inoltre, permette l' analisi su qualsiasi sistema operativo utilizzando delle macchine fisiche, offrendo un grande vantaggio nel contrastare i malware che rilevano l' ambiente di esecuzione.

Come gli altri Sandbox considerati Joe Sandbox monitora e registra il comportamento dei malware durante l'esecuzione, fornendo informazioni dettagliate sulle sue attività, come la creazione di file, modifiche ai registri di sistema, chiamate API, e le connessioni di rete. Inoltre combina l' analisi statica con l'analisi dinamica per ottenere una visione più completa della natura del malware, migliorando l' efficacia sui malware che utilizzano tecniche di occultamento, packing e self-modifying code, rendendo possibile una comprensione delle tecniche di evasione come: *logic bombs, system fingerprinting and sleeps*. In aggiunta Joe-sandbox possiede uno dei più grandi Database di comportamenti generici di malware, che consistono in più di 2437+ behavior, 2633+ YARA rules, che coprono più piattaforme fra cui: Windows, Android, macOS e Linux.

### 4.1.5 Metadefender

Metadefender è una piattaforma di analisi del malware avanzata che offre una varietà di servizi per la scansione e l'analisi di file sospetti. Sviluppata da OPSWAT, Metadefender combina diverse tecnologie di scansione antivirus e motori di analisi per fornire una valutazione completa e accurata delle minacce. Con la scansione multi-motore offerta da Metadefender è possibile sottoporre il file a 38 + motori di analisi che eseguono sia analisi statiche che dinamiche. Infine Metadefender genera un report dettagliato che include le informazioni sul file, i risultati della scansione per ciascun motore considerato ed informazioni specifiche sul comportamento del malware.

### 4.1.6 Virus Total

VirusTotal è una piattaforma di analisi dei malware gratuita e ampiamente utilizzata che offre un'analisi completa di file sospetti. Acquisita da Google nel 2012, VirusTotal combina diverse tecnologie di scansione antivirus e motori di analisi per fornire una valutazione accurata e affidabile delle minacce informatiche. Virus Total utilizza un approccio di scansione multi-motore - inclusi sandbox - per analizzare i file utilizzando più di 70 scanner antivirus confrontandoli con un vasto database di

firme digitali. Per ogni file analizzato, VirusTotal fornisce un report dettagliato che include: informazioni sui file, risultati della scansione per ogni motore utilizzato, informazioni sulle minacce. VirusTotal ha anche una vasta community di analisti di sicurezza che collaborano per identificare e classificare nuove minacce. Questa community fornisce informazioni e analisi avanzate sui file sospetti, arricchendo i dati di VirusTotal e consentendo agli utenti di beneficiare dell'esperienza di altri esperti di sicurezza. Per gli sviluppatori, VirusTotal offre un'API gratuita che consente di integrare le sue funzionalità nei propri strumenti di sicurezza. Questo permette di automatizzare l'analisi dei file sospetti e implementare diverse funzionalità di sicurezza, pur rispettando i limiti di utilizzo dell'API gratuita.

### 4.1.7 YARA

YARA rappresenta uno degli strumenti più potenti nel campo della sicurezza informatica per l'identificazione e la categorizzazione dei malware, lo ritroviamo infatti utilizzato da tutti gli altri strumenti presi in esame come tool base per il riconoscimento di firme. È una tecnologia flessibile e potente che consente agli analisti di sicurezza di creare regole personalizzate per rilevare specifici modelli di comportamento o firme all'interno dei file.

Questo linguaggio di regole, versatile ed estensibile, permette agli utenti di definire criteri precisi per individuare e classificare le minacce digitali. Le regole YARA possono essere create per identificare non solo singoli malware, ma anche intere famiglie di minacce o comportamenti maligni specifici.

Con YARA, gli analisti possono eseguire analisi approfondite dei file sospetti, includendo documenti, eseguibili, script e molti altri tipi di file. Questo strumento è particolarmente utile in scenari in cui è necessario identificare rapidamente e accuratamente nuove varianti di malware o adattarsi a minacce emergenti.

YARA è uno strumento molto potente ma che dipende fortemente dal set di regole utilizzato per l' analisi. Nel corso di questo tirocinio è stato notato come, infatti, cambiando il set di regole[(2)] si ottengono risultati diversi per le prestazioni, e che quindi, utilizzando set di regole open-source potrebbero essere meno efficaci rispetto a quelli utilizzati da grandi organizzazioni.

**Tabella 4.1** Metodo e tipo di rilevamento degli strumenti di analisi dei malware

| **Strumento** | **Metodo di Rilevamento** | **Tipo di Rilevamento** |
|---|---|---|
| Any.Run | Analisi dinamica | Anomaly-based, Signature-based |
| Comodo | Analisi statica e dinamica | Anomaly-based, Signature-based |
| Hybrid Analysis | Analisi statica e dinamica | Anomaly-based, Signature-based |
| Joe Sandbox | Analisi statica e Analisi dinamica | Anomaly-based, Signature-based |
| Metadefender | Analisi statica | Signature-based |
| VirusTotal | Analisi statica e dinamica | Signature-based |
| Yara | analisi statica | Signature-based |

# Capitolo 5

# Malware e tipologie

Il NIST Special Publication 800-83 descrive un **malware** come[9]:

Il malware, abbreviazione di "*software dannoso*" (malicious software), è un termine generico utilizzato per descrivere software progettato per danneggiare, interrompere, rubare informazioni o ottenere l'accesso non autorizzato a sistemi informatici. Il malware può assumere molte forme, tra cui virus, worm, trojan, ransomware, spyware, adware e altri.

## 5.1 Classificazione dei Malware

Sebbene sia possibile considerare molteplici aspetti, un approccio pratico classifica il malware in due categorie generali, basate in primo luogo su come si diffonde o si propaga per raggiungere i target designati e in secondo luogo sulle azioni o payload che esegue una volta raggiunto un target.

Fra i meccanismi di propagazione vi sono l' infezione da parte di virus di eseguibili esistenti o contenuti interpretati che successivamente vengono diffusi ad altri sistemi, lo sfruttamento di vulnerabilità del software di vulnerabilità del software sia localmente che in rete da parte di worm o di drive-by-download per consentire al malware di replicarsi e l' utilizzo di attacchi di ingegneria sociale che convincono gli utenti a eludere meccanismi di siurezza al fine di installare Trojoan o a rispondere ad attacchi di phishing. I primi approcci alla classificazione dei malware distinguevano tra quelli che necessitano di un programma host, ovvero i codici parassiti, quali i virus e quelli costituiti da programmi indipendenti e autonomi eseguiti sul sitema, quali i worm, i Trojan e i bot. Un' altra distinzione in uso era tra i malware che si replicano , quali i Trojan e le email di spam, e quelli che si replicano, che comprendono i Virus e i Worm. Le azioni del payload eseguite dai malware una volta raggiunto un sistema target possono comprendere la corruzione di file di sistema o di dati, la compromissione di servizi per rendere il sistema un agente zombie con cui sferrare attacchi come parte di una botnet, il furto di informazioni dal sistema , soprattutto login, password o altri dettagli personali, utilizzando programmi di keylogging o spyware e il comportamento furtivo con cui il malware nasconde la propria presenza nel sistema quando vi sono tentativi di rilevarlo o di bloccarlo. Mentre i primi malware tendevano a utilizzare un singolo meccanismo di propagazione per trasmettere un singolo payload, nel corso della loro

evoluzione si nota una crescita di malware ibridi che incorporano più meccanismi contemporaneamente, massimizzando la velocità di propagazione e la gravità dell' attacco.

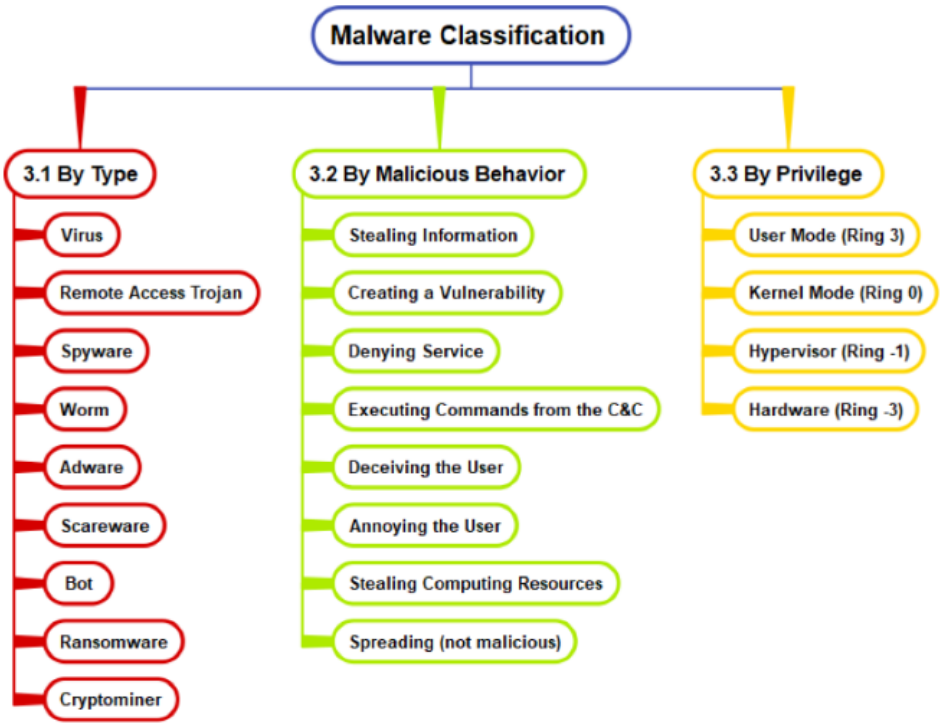



**Figura 5.1** Tassonomia dei Malware

### 5.1.1 Propagazione: Virus

La prima categoria di propagazione dei malware riguarda frammenti di sofware parassita che si legano a qualche contenuto eseguibile esistente. Il termine "virus" è cosi comune che viene spesso utilizzato per riferirsi al malware in generale. Un Virus informatico è un frammento di software che può "infettare" altri programmi, o comunque qualsiasi tipo di contenuto eseguibile, tramite la loro modifica. La modifica comprende l' inserimento all' interno del codice originale di una procedura per la creazione di copie del codice del virus, che possono a loro volta andare a infettare altri contenuti. Analogamente ai virus biologici, un virus informatico racchiude nella propria sequenza di istruzioni la procedura per produrre copie di se stesso. Un virus che si inserisce in un programma eseguibile è in grado di fare tutto ciò che il programma è autorizzato a fare. Viene eseguito segretamente quando il programma ospite viene lanciato. In seguito si sono evoluti in Macro viru, che sfruttano i conenuti attivi supportati da alcuni tipi di documenti, come i file Word, Excel o i PDF. Tali documenti sono facilmente modificabili e condivisibili dagli utenti come parte del loro normale utilizzo del sistema. Attualmente la modalità di infezione virale costituisce uno dei principali meccanismi di propogazione.

Un Virus è costituito da tre componenti principali:

1. **Payload**: Il payload è la parte del virus che esegue l'azione dannosa sul sistema infetto. Può includere funzionalità come la cancellazione di file, la modifica del registro di sistema, il furto di informazioni sensibili o la creazione di una porta di accesso per il controllo remoto.

2. **Meccanismo di Infezione**: Il mezzo con cui un virus di diffonde o si propoaga. Il meccanismo viene anche detto **vettore di infezione.**

3. **Trigger**: Il trigger è l'evento o la condizione che attiva il virus. Può essere un'azione specifica dell'utente, come l'apertura di un file infetto, o un evento automatico, come il collegamento a una rete.

4. **Meccanismi di Autoprotezione**: Alcuni virus sono dotati di meccanismi di autoprotezione per evitare la rimozione o la disinfezione da parte degli strumenti di sicurezza. Possono criptare parti del loro codice, modificare costantemente la loro struttura o nascondersi in aree protette del sistema.

Nel corso della sua esistenza, un virus ripicamente attraversa le seguenti quattro fasi:

1. **Fase Dormiente**: Durante questa fase, il virus è inattivo e dormiente nel sistema ospite, aspettando il momento opportuno per entrare in azione senza dare segni evidenti della sua presenza. Verrà poi attivato da qualche evento particolare.

2. **Fase di Propagazione**: Durante questa fase, il virus cerca attivamente di diffondersi ad altri sistemi o risorse, utilizzando varie tecniche come la copia in file eseguibili, l'infezione di dispositivi di archiviazione rimovibili o la propagazione attraverso reti di computer.

3. **Fase di Esecuzione**: È il momento in cui il virus viene attivato e inizia a eseguire il suo codice dannoso. Questo può essere scatenato da eventi specifici o condizioni predefinite nel sistema.

4. **Fase di Attivazione**: È il momento in cui il virus entra completamente in azione, eseguendo il suo payload dannoso o svolgendo altre attività dannose.

**Metodi di Evasione**

1. **Virus criptato**: I virus informatici possono crittografare il proprio codice utilizzando algoritmi crittografici complessi. Questo rende il codice dannoso illeggibile agli strumenti di sicurezza e rende più difficile l'analisi del malware. La crittografia può includere tecniche di cifratura simmetrica o asimmetrica, rendendo il codice inaccessibile senza la chiave corretta. Una porzione del virus crea una chiave crittografica e cifra il resto del virus. Quando il programma infetto viene eseguito il virus utilizza la chiave per decifrare il codice criptato e, al momento della replicazione, viene generata una chiave diversa in modo da evitare il riconoscimento dei pattern.

2. **Polimorfismo**: I virus polimorfici sono in grado di modificare la propria struttura e il proprio comportamento ad ogni infezione. Utilizzano algoritmi complessi per generare varianti del proprio codice senza cambiare la funzionalità essenziale del malware. Questa tecnica rende difficile la creazione di firme di rilevamento fisse e richiede un'analisi più approfondita per identificare la minaccia.

3. **Polimorfismo e Metamorfismo**: I virus metamorfici vanno oltre il polimorfismo e riscrivono completamente il proprio codice ad ogni infezione. Mantengono la stessa funzionalità del malware ma modificano la sua struttura, rendendo difficile la rilevazione basata sulla firma. Questa tecnica richiede un'analisi avanzata per identificare e comprendere il comportamento del virus. Per ottenere queste variazioni , e quindi modificare la propria firma, il virus al momento della replica può aggiungere istruzioni inutili oppure scambiare l' ordine di istruzioni indipendenti. Un altro approccio utilizza la crittografia, seguendo la strategia del virus criptato. Spesso si muta anche il metodo usato per mutare ad ogni iterazione.

4. **Steganografia**: La steganografia è una tecnica che consiste nell'incorporare il codice dannoso all'interno di file multimediali, come immagini o video. Il malware è nascosto all'interno dei dati del file, rendendolo difficile da individuare. Questa tecnica sfrutta la natura binaria dei file multimediali per nascondere il codice dannoso in modo efficace ed eludere i sistemi antivirus.

### 5.1.2 Propagazione: Worm

In questa categoria di propagazione affrontiamo gli exploit utilizzati dai worm. Un worm è un programma che ricerca attivamente altre macchine da infettare per poi utilizzarle come *rampe di lancio* per sferrare altri attacchi ad altre macchine. Se il worm trova una vulnerabilità comune l' infenzione può essere estremamente veloce in quanto può seguire un andamento geometrico. Il worm sfrutta vulnerabilità sofware sia lato client che lato server per condurre i suoi attacchi. Dopo la propagazione, di solito, il worm porta con se anche qualche sorta di payload dannoso. Uno dei primi esempi di worm su larga scala è il Morris worm. Creato da Robert Tappan Morris nel 1988, il Morris Worm si diffuse rapidamente attraverso Internet, causando disagi e danni significativi agli ospiti infetti.

Il Morris Worm fu progettato per sfruttare diverse vulnerabilità presenti nei sistemi Unix dell'epoca. In particolare, sfruttava le debolezze nella gestione delle password e nelle routine di autenticazione dei sistemi Unix per diffondersi attraverso la rete. Il worm sfruttava anche una vulnerabilità nel servizio Sendmail, utilizzato per l'invio di email su Unix.

Il funzionamento del Morris Worm era relativamente semplice ma estremamente efficace. Una volta introdotto in un sistema Unix vulnerabile, il worm si copiava in una directory del sistema e tentava di ottenere l'accesso a nuovi sistemi attraverso una serie di tecniche, inclusa l'esecuzione di attacchi di password deboli. Una volta ottenuto l'accesso, il worm si installava sul nuovo sistema e ripeteva il ciclo di infezione.

Il problema principale del Morris Worm era la sua aggressività nell'autoreplicazione. Ogni istanza del worm avrebbe cercato di infettare il massimo numero possibile di sistemi, generando un carico di traffico di rete e un sovraccarico dei sistemi che contribuivano a paralizzare Internet.

Il Morris Worm non era progettato per scopi dannosi, ma per dimostrare le vulnerabilità della rete e dei sistemi informatici dell'epoca. Tuttavia, a causa della sua aggressività e della sua rapida diffusione, il worm causò danni significativi, interrompendo i servizi di rete e richiedendo sforzi considerevoli per rimuoverlo e ripristinare la normale operatività dei sistemi.

Di norma un worm presenta le sesse fasi di un virus con una differenza nella fase di propagazione:

1. Cerca sugli altri sistemi gli opportuni meccanismi di accesso da infettare esaminando le tabelle delgi host, le rubriche, gli elnchi di amici, i peer attendibili e altri archivi simili contenenti dettagli di accesso al sistema remoto; esamindo possibili indirizzi di host target; o cercando dispositivi di supporto rimovibili inonei all' utilizzo.

2. sfrutta le vulnerabilià trovate per tasferire una copia di se stesso al sistema remoto ed eseguire la copia effettuato nel nuovo sistema.

**Ricerca del Target**

1. **Scansione di Range IP**: I worm possono eseguire una scansione di range di indirizzi IP per individuare e identificare potenziali sistemi vulnerabili. Questo metodo permette loro di individuare un'ampia gamma di host all'interno di una rete specifica e di identificare quelli che possono essere facilmente compromessi.

2. **Analisi di Servizi di Rete**: I worm analizzano i servizi di rete esposti su un determinato host per individuare vulnerabilità e punti di accesso potenziali. Possono esaminare porte aperte, protocolli di rete attivi e configurazioni di servizio per identificare le opportunità di accesso non autorizzato.

3. **Hit-list**: l' attaccante redice anzitutto una lunga lista di macchine potenzialmente vulnerabili. Poi la utilizza per trovare i bersagli vulnerabili per il worm.

4. **Attacchi di Forza Bruta**: Alcuni worm eseguono attacchi di forza bruta contro i sistemi target, cercando di indovinare le credenziali di accesso o le password deboli per ottenere l'accesso non autorizzato. Utilizzano liste di password predefinite o generano combinazioni casuali per tentare di accedere ai sistemi target.

5. **Esplorazione di Database di Nomi di Dominio (DNS)**: I worm possono esplorare i database di nomi di dominio (DNS) per individuare indirizzi IP associati a nomi di dominio specifici. Questo metodo consente loro di individuare gli host all'interno di una rete utilizzando nomi di dominio conosciuti e di identificare i sistemi che possono essere bersagliati per l'infezione.

6. **Utilizzo di Backdoor e Exploit**: Alcuni worm sfruttano backdoor e exploit già presenti nei sistemi target per ottenere l'accesso non autorizzato. Utilizzano vulnerabilità di sicurezza precedentemente installate o porte aperte per infiltrarsi nei sistemi target e propagare il loro codice dannoso.

#### State of the Art

Al momento lo State of the art relativo ai worm comprende i seguenti aspetti:

1. **Multipiattaforma**: i worm più recenti non rimangono circoscritti alle macchine Windows, ma possono attaccare una varietà di piattaforme, e in particolare le varianti più diffuse di UNIX, o sfruttare macro o linguaggi di scipting supportati dai documenti più diffusi.

2. **Diffusione rapida**: si sfruttano varie tecniche per ottimizzare il tasso di diffusione di un worm al fine di massimizzare la sua probabilità di individuare il maggiore numero di macchine nel minor periodo di tempo.

3. **Polimorfismo**: per eludere il rilevamento, i worm utilizzano le tecniche poliformiche tipiche dei virus. Ogni copia del worm presenta un nuovo codice generato sul momento utilizzando istruzioni crittografiche.

4. **Zero-day exploit**: per massimizzare il fattore sorpresa e la propagazione un worm dovrebbe sfruttare una vulnerabilità non nota che verrà poi scoperta solo al lancio del worm. La maggiore parte delle vulnerabilità si trova in software comuni per computer e smartphone, altre si possono trovaree all' interno di librerie pubbliche o package di sviluppo.

### 5.1.3 Propagazione: Ingegneria Sociale

L' ultima categoria di propagazione che consideriamo implica l' utilizzo di tecniche di ingegneria sociale, ovvero "ingannare" gli utenti per favorire la compromissione dei loro sistemi o delle loro informazioni personali. Questo approccio sfrutta la naturale propensione umana a fidarsi, curiosare o agire per cortesia, per convincere le vittime a compiere azioni che favoriscono gli obiettivi dell'attaccante. Gli attaccanti cercano di convincere gli utenti a scaricare e installare malware sui propri dispositivi, spesso mascherandoli da software legittimi o da contenuti interessanti.

#### Spam

Le email spam rappresentano una delle forme più comuni di comunicazione indesiderata e invadente su Internet. Si tratta di messaggi non richiesti che vengono inviati in grandi quantità a destinatari che non hanno espresso interesse o consenso nel riceverli. Questi messaggi sono spesso concepiti per promuovere prodotti, servizi o contenuti che non hanno alcuna rilevanza per il destinatario, o peggio, per truffare o ingannare gli utenti.

Il contenuto delle email spam può variare ampiamente, ma spesso include promozioni di prodotti farmaceutici, offerte finanziarie discutibili, proposte per servizi online dubbi, promesse di guadagni facili o richieste di informazioni personali e

finanziarie. Molte di queste email sono progettate per giocare sulle emozioni degli utenti, cercando di suscitare la curiosità, la paura o l'avidità al fine di ottenere una risposta. Lo spam è anche un importante vettore di malware. Infatti l' email potrebbe avere un documento allegato, che, se aperto potrebbe sfruttare vulnerabilità del software per installare software malevolo. Oppure, potrebbe avere in allegato un Trojan Horse o un codice di scripting che, se eseguito installa anch' esso malware sul sistema.

#### Trojan Horse

Un Trojan horse è un programma o una procedura utile, o in apparenza utile contenente codice nascosto che, se invocato, svolge alcune operazioni indesiderate o dannose. Per esempio al fine di acquisire l' accesso alle informazioni sensibili, un attaccante potrebbe creare un programma Trojan horse - spacciato come programma leggittimo - che, una volta eseguito, analizza i file dell' utente alla ricerca delle informazioni sensibili desiderate e ne invia una copia all' attaccante stesso tramite un web form, un email o un messaggio di testo. Gli autori inducono gli utenti a eseguire il programma incorporandolo in un gioco o in un comodo programma di utilità rendondolo accessibili nei vari siti di distribuzione.

I trojan possono rientrare in uno di questi tre modelli:

1. Proseguire nell' esecuzione della funzione del programma originale e in più svolgere un' attività malevola separata.

2. Proseguire nell'esecuzione della funzione del programma orginale, ma madoficare la funzione per svolgere attività dannose( e.g inserire un modulo per raccogliere le password e username) o per maschrarne altre ( e.g. una versione Trojan horse di un programma che elnca i processi che non mostra determinati processi dannosi).

3. Svolgere una funzione dannosa che sostituisce in completamente la funzione del programma originale.

A differenza dei worm e dei virus, i trojan horse non si replicano. Inoltre i trojan rappresentano un problema significativo anche per gli smartphone, specialmente quando si tratta di dispositivi con sistema operativo Android. A differenza di iOS, il sistema operativo di Apple, che controlla strettamente le applicazioni attraverso l'App Store, Android consente agli utenti di installare applicazioni da fonti esterne, come store di terze parti o direttamente da file APK scaricati da Internet. Questo rende gli smartphone Android più vulnerabili agli attacchi da parte di trojan e altri tipi di malware.

## 5.2 Payload - Corruzione del sistema

Una volta che il malware è attivo sul sistema target, la preoccupazione successiva riguarda quali azioni intraprenderà su tale sistema. Ossia, quale payload trasporta? Alcuni malware presentano un payload inesistente o non funzionale. Il suo unico obiettivo, intenzionale causato da un rilascio anticipato accidentale, è quello di

diffondersi. La maggior parte delle volte vengono trasportati da uno o più payload che svolgono attività segrete a beneficio dell' attaccante. Uno dei primi payload riscontratto in diversi virus e worm provocava la distruzione dei dati sul sistema infetto al verificarsi di certe condizioni di attivazione. Un payload correlato è quello che mostra messaggi indesiderati sul sistema dell' utente al momento dell' attivazione. Un' altra variante, piè seria, cerca di provocare danni al sistema nel mondo reale. Tutte queste azioni hanno come obiettivo quello di compromettere l' intergrità del software.

### 5.2.1 Distruzione dei dati e ransomware

Il virus Chernobyl, noto anche come CIH, apparso per la prima volta nel 1998 è un primo esempio di virus parassita distruttivo, residente in memoria, per sistemi Windows 95 e 98. Prende il nome dalla centrale nucleare di Chernobyl, in quanto è stato progettato per attivarsi in modo distruttivo il 26 aprile, la data del disastro nucleare di Chernobyl avvenuto nel 1986. Una volta attivato, il virus Chernobyl si diffonde rapidamente attraverso la rete e le unità di archiviazione, infettando file cruciali del sistema operativo. Una delle sue caratteristiche più dannose è la capacità di sovrascrivere parti vitali del BIOS del computer, rendendo il sistema completamente inutilizzabile. Questo ha reso il virus particolarmente temibile, poiché poteva causare danni irreversibili ai computer infetti.

In alternativa alla mera distruzione dei dati, alcuni malware criptano i dati dell' utente e richiedono di pagare un riscatto per ottenere al chiave necessaria al recupero di tali informazioni. Un esempio importante e degno di nota è il ransomware WannaCry, apparso nel maggio del 2017, ha infettato numerosi sistemi in diversi paesi. Una volta installato sui sistemi infettati, criptava un elevato numero di ile corrispondenti a specicifici tipi e poi richiedeva il pagamento di un riscatto in Bitcoin per il loro recupero. A pagamento effettuato, il recupero di dette informazioni era generalmente possibile soltanto se l' organizzazione disponeva di validi backup e di un adeguato piano di incident responde e disaster recovery.

#### Bombe Logiche

La bomba logica è un componente chiave dei malware che danneggiano i dati. Si tratta di un codice incorporato nel malware configurato per *esplodere* quando si verificano determinate condizioni:

1. **Data Specifica:** Una logic bomb potrebbe essere programmata per attivarsi in una data specifica, come il compleanno di un individuo, una data di scadenza o un anniversario particolare.

2. **Numero di Esecuzioni:** La logic bomb potrebbe essere attivata dopo un certo numero di esecuzioni del programma o del processo in cui è incorporata.

3. **Cambiamenti nei Dati:** La logic bomb potrebbe essere attivata in risposta a determinati cambiamenti nei dati o nelle condizioni dell'ambiente informatico.

4. **Eventi di Rete:** La logic bomb potrebbe essere innescata da eventi di rete, come la connessione o la disconnessione da una rete specifica.

5. **Cambiamenti di Autorizzazione:** Una logic bomb potrebbe essere attivata in risposta a cambiamenti nelle autorizzazioni o nei privilegi di accesso.

6. **Variabili di Tempo:** La logic bomb potrebbe essere innescata in base a variabili di tempo, come l'intervallo di tempo trascorso dall'installazione del programma.

## 5.3 Payload - agenti di attacco

Nel caso in cui il malware sovverta le risorse computazionali del sistema e di rete per fornirle, poi, all' attaccante allora si parla di bot, drone o zombie. Il bot in genere viene in genere installato su centinaia e migliaia di computer appartenenti a terzi ignari. I sistemi compromessi non sono solo i personal computer, ma comprendono anche server e ultimamente dispositivi embedded quali router o videocamere di sorveglianza. Spesso l' insieme dei bot è in grado di agire in modo coordinato; in questo caso si parla di **botnet**.

Utilizzi di botnet comprendono:

1. **Attacchi Distribuiti Denial of Service (DDoS):** Le botnet sono ampiamente utilizzate per condurre attacchi DDoS, che mirano a sovraccaricare un servizio online con un enorme flusso di richieste di traffico, rendendolo inaccessibile agli utenti legittimi.

2. **Spam e Phishing:** Le botnet vengono utilizzate per inviare enormi quantità di spam e e-mail di phishing. I bot possono generare e inviare e-mail fraudolente o indesiderate a una vasta gamma di destinatari.

3. **Furti di Informazioni:** Le botnet possono essere utilizzate per raccogliere informazioni sensibili dagli utenti, come dati personali, credenziali di accesso, informazioni finanziarie e altro ancora.

4. **Mining di Criptovalute:** Le botnet possono essere sfruttate per il mining di criptovalute, sfruttando le risorse di elaborazione delle macchine compromesse per generare valuta digitale.

5. **Attacchi Mirati:** Le botnet possono essere utilizzate per condurre attacchi mirati contro organizzazioni o individui specifici, sfruttando le risorse della rete di computer compromessi per infiltrarsi nei sistemi o danneggiare l'infrastruttura informatica.

6. **Manipolazione delle Reti Sociali:** Le botnet possono essere utilizzate per manipolare le reti sociali online, generando falsi account e interazioni per diffondere disinformazione o influenzare opinioni pubbliche.

## 5.4 Payload - furto di informazioni - keylogger, phising, spyware

I creatori di malware possono essere interessati alla raccolta di dati memorizzati sul sistema infetto. Un tipico obiettivo è rappresentato dalle credenziali di login

e password dell' utente su siti bancari. Anche se meno frequente il payload può colpire documenti o dettagli di configurazione del sistema a scopo di ricognizione o spionaggio.

### 5.4.1 Furtro di credenziali, keylogger e spyware

In genere gli utenti inviano le proprie credenziali di login e password tramite canali di comunicazione cifrati che li proteggono dalla loro intercettazionemonitorando i pacchetti di rete. Per aggirare questo problema, un attaccante può installare un **keylogger**, che cattura le sequenze di caratteri digitati su tastiera sulla macchina infetta per raccogliere queste informazioni sensibili. Ancora più sofisticati sono gli **spyware**, che permettono il monitoraggio di una vasta gamma di attività sul sistema. Fra queste possono esserci la cronologia di navigazione, il reindirizzamento di alcune richieste web a siti contraffatti.

## 5.5 Payload - Stealthing - backdoor, rootkit

Per ultima analizziamo la categoria di payload che tratta le tecniche utilizzate dai malware per celare la propria prenseza al sistema infetto e per ottenere un accesso clandestino a tale sistema.

### 5.5.1 Backdoor

Una **backdoor** è un punto di ingresso segreto in un programma che permette a qualcuno, che ne è a conoscenza, di acquisire l'accesso al sistema evitando le consuete procedure di autenticazione. Le backdoor vengono utilizzate in modo legittimo quando, nel testing di un software che richiede lunghe procedure di autenticazione, vengono invece inserite procedure per *bypassare* tali procedure. Quando vengono inserito in modo fraudolento, però, le backdoor possono diventare una vera minaccia per un sistema. Una delle strategie più recenti viene implementata come servizio di rete in ascolto su una certa porta non standard a cui l' attaccante può connettersi e inviare comandi da eseguire sul sistema compromesso. Il ransomware WannaCry presentava una backdoor di questo tipo.

### 5.5.2 Rootkit

Un rootkit è un insieme di programmi installati su un sistema per mantenere un accesso segreto ad esso con privilegi di admin. Ciò fornisce l' accesso a tutte le funzioni e i servizi del sistema operativo. Tramite l' accesso come root un attaccante può dispone del controllo completo del sistema. Un rootkit può apportare diverse modifiche al sistema per mantenere segreta la propria esistenza. Un rootkit può essere classificato nei seguenti modi:

1. **Persistenza:** Un rootkit è progettato per rimanere attivo sul sistema anche dopo un riavvio. Utilizza tecniche sofisticate per garantire la sua persistenza nel tempo, come l'installazione di servizi nascosti o la modifica dei file di avvio del sistema.

2. **Residenza in Memoria:** Alcuni rootkit risiedono interamente in memoria, evitando di lasciare tracce sui dischi rigidi. Questa caratteristica rende la loro individuazione più difficile, in quanto non lasciano file o tracce evidenti sul sistema di archiviazione, di contro non sopravviveranno un riavvio.

3. **Modalità Utente:** Possono modificare le chiamate API di sistema ad esempio alterando i risultati di listing delle directory.

4. **Modalità Kernel:** Altri rootkit operano a livello del kernel del sistema operativo, consentendo loro di avere un controllo completo sul sistema e di nascondere le loro attività in modo più efficace. Questi rootkit sono particolarmente difficili da individuare e rimuovere.

5. **Modalità Esterna:** Alcuni rootkit possono operare in modalità esterna, interagendo con il sistema da remoto attraverso reti o altri mezzi di comunicazione. Questo consente agli attaccanti di controllare il sistema compromesso da remoto e di eseguire operazioni dannose senza essere rilevati.

# Capitolo 6

# Esperimenti

Per valutare le prestazioni dei vari strumenti per il rilevamento dei malware, è stata pianificata e condotta una serie di esperimenti in un ambiente controllato. Gli esperimenti sono stati progettati per testare l'efficacia e l'accuratezza dei tool nella rilevazione di un insieme di malware selezionati dal dataset di riferimento.

## 6.1 Test degli Strumenti

Ciascuno strumento di rilevamento dei malware è stato eseguito in sequenza sul dataset ottenuto. Ogni strumento è stato configurato con le impostazioni predefinite e i parametri consigliati dal produttore per garantire una valutazione equa delle sue capacità di rilevamento. Fatta eccezione per Yara strumento che dipende fortemente dalle regole utilizzate e su cui sono stati condotti più esperimenti con regole diverse, osservando una grande variazione nella accuretezza a seconda delle qualità delle regole applicate.

## 6.2 Risultati degli Esperimenti

Qui di seguito si riportano i risultati degli esperimenti condotti per valutare le prestazioni dei vari strumenti di rilevamento dei malware. Per prima cosa, vengono presentate le metriche relative all'intero dataset per ciascun Tool.

Considerando la sensibilità, la specificità e l'accuratezza come metriche chiave per valutare l'efficacia dei tool, i risultati forniti offrono una panoramica completa delle performance di ciascuno strumento. Tali metriche consentono di comprendere l'efficacia complessiva dei tool nel rilevamento dei malware e di confrontarne le capacità in modo approfondito.

Successivamente, saranno presentati tre grafici che illustrano rispettivamente la sensibilità, la specificità e l'accuratezza per ogni Tool, fornendo un'analisi visiva delle loro prestazioni. Questi grafici offrono una chiara rappresentazione delle differenze tra i vari strumenti e forniscono un valido supporto alla valutazione delle loro capacità.

### 6.2.1 Analisi comparativa degli strumenti

Dopo aver testato ogni strumento sul dataset, ho applicato le metriche descritte in precedenza per analizzare le prestazioni, possiamo osservare che Virus Total ottiene un punteggio elevato sia in sensibilità che in accurettezza, ed ha ottenuto un basso livello di specificità dimostrandosi il tool più performante fra quelli presi in esame. Notiamo invece come yara sia risultato il peggiore fra tutti, questo è dovuto soprattutto al set di regole open source che sono state utilizzate, infatti, le performarce di yara è molto dipende dal set di regole utilizzate che possono essere più o meno specifiche a seconda dei casi ed è molto soggetto ai falsi positivi. (2)

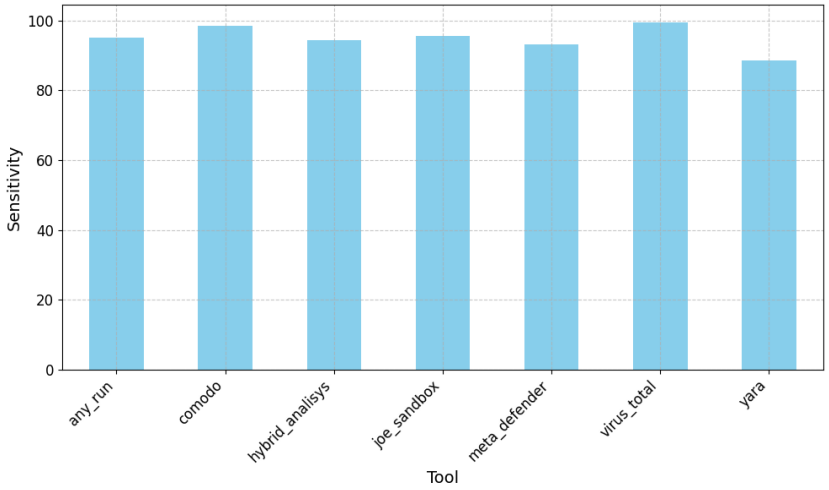



**Figura 6.1** Sensibilità per ogni strumento

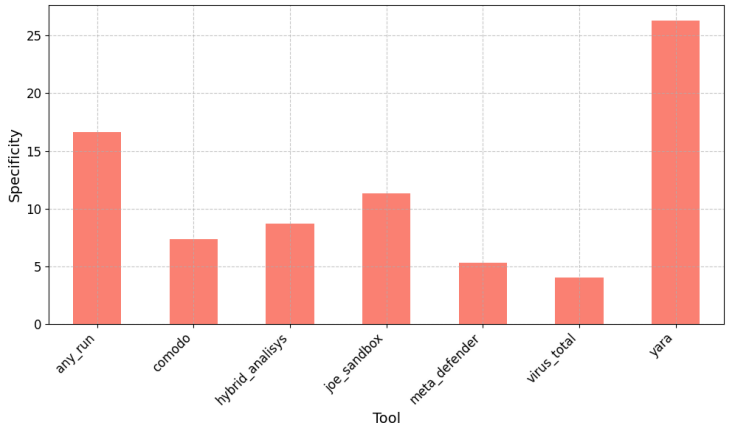



**Figura 6.2** Specificità per ogni strumento

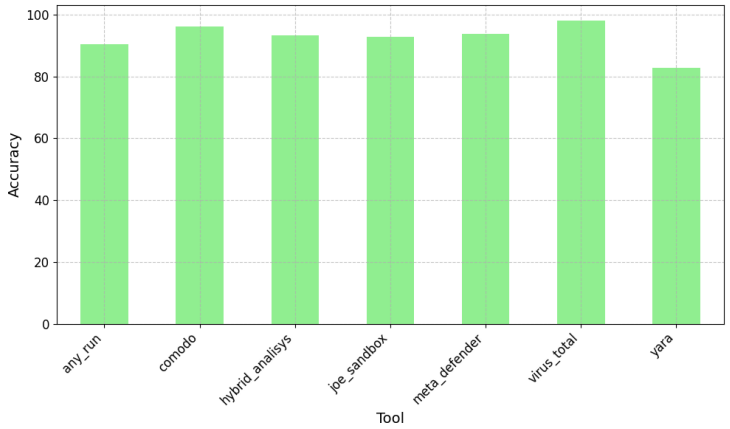



**Figura 6.3** Accuratezza per ogni strumento

### 6.2.2 Analisi comparativa basata sulle famiglie di malware

Infine si è voluta analizzare la performance di ogni tool, calcolando l' accuratezza e la sensibilità, sulle differenti famiglie di malware, cercando di rivelare se alcuni tool funzionassero meglio di altri su determinati tipi di malware.

#### Anyrun

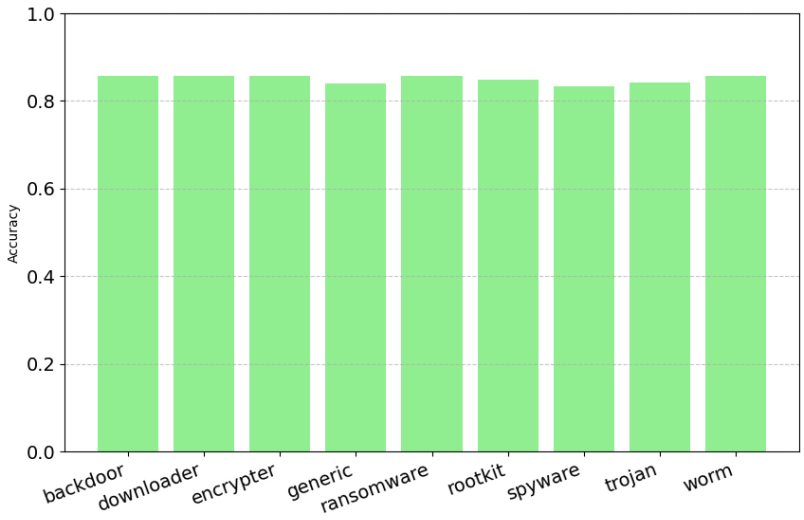



**Figura 6.4** Accuracy per Any.Run

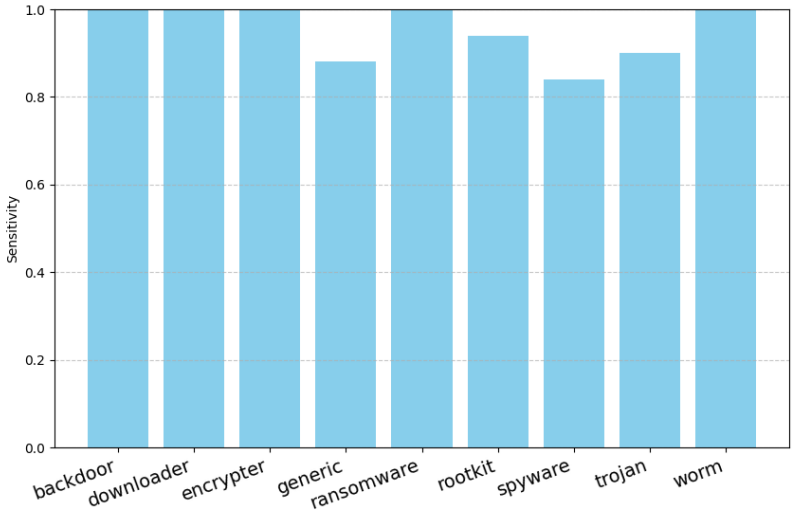



**Figura 6.5** Sensitivity per Any.Run

### Comodo

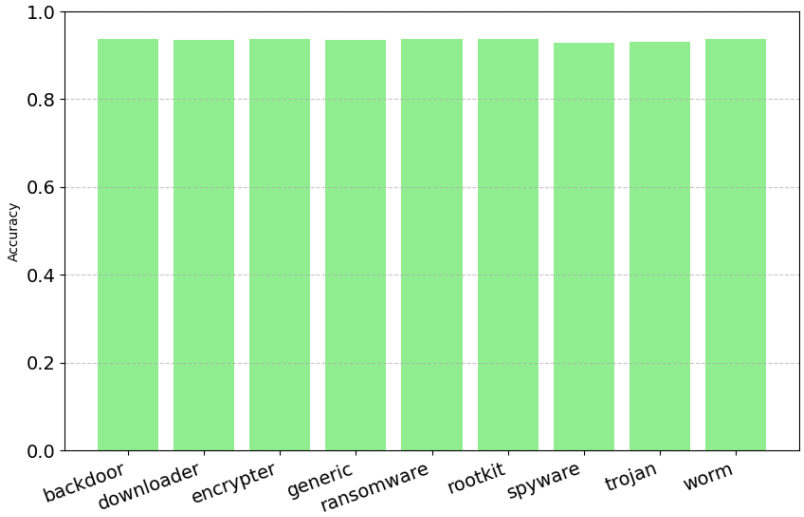



**Figura 6.6** Accuracy per Comodo

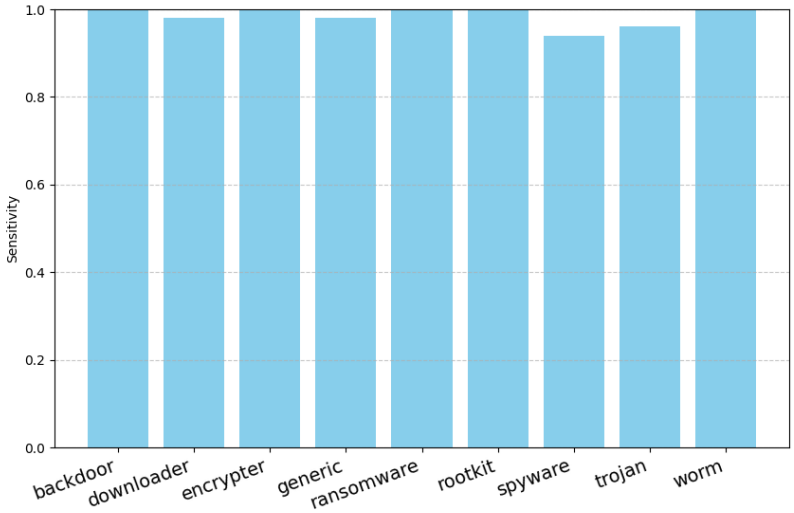



**Figura 6.7** Sensitivity per Comodo

### Hybrid Analysis

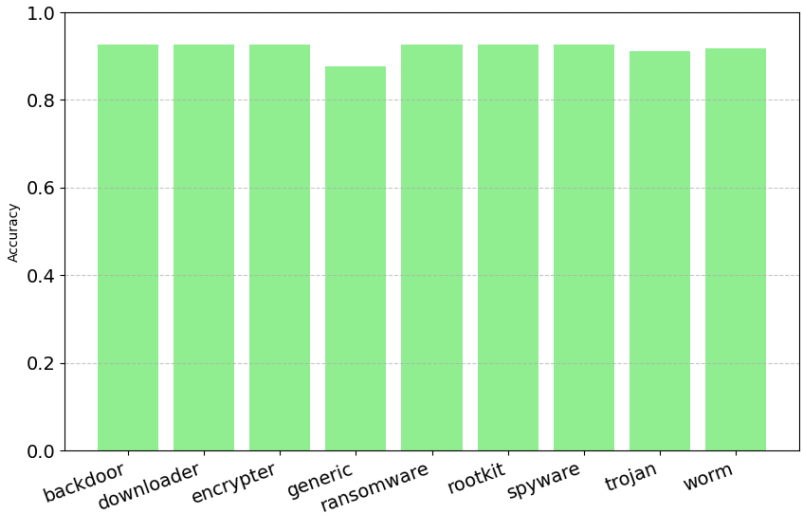



**Figura 6.8** Accuracy per Hybrid Analysis

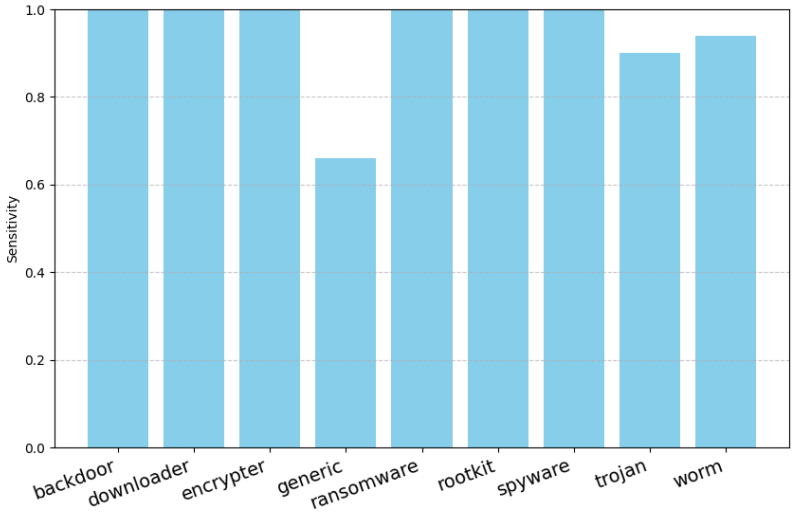



**Figura 6.9** Sensitivity per Hybrid Analysis

### Joe Sandbox

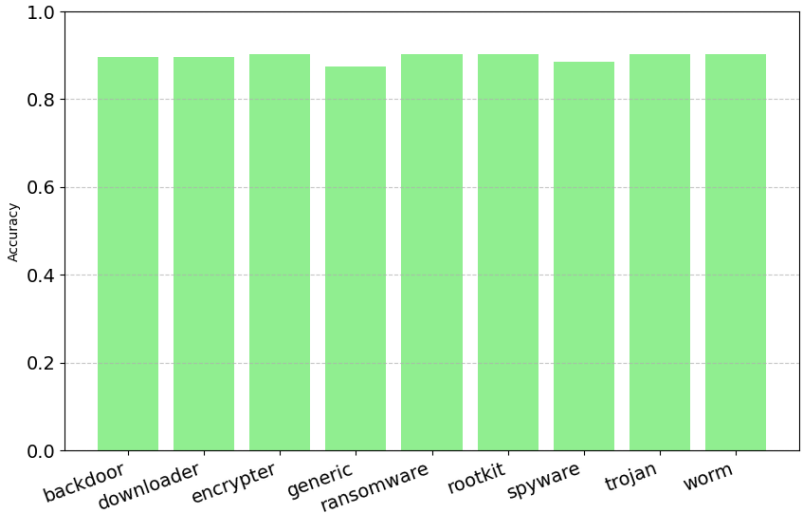



**Figura 6.10** Accuracy per Joe Sandbox

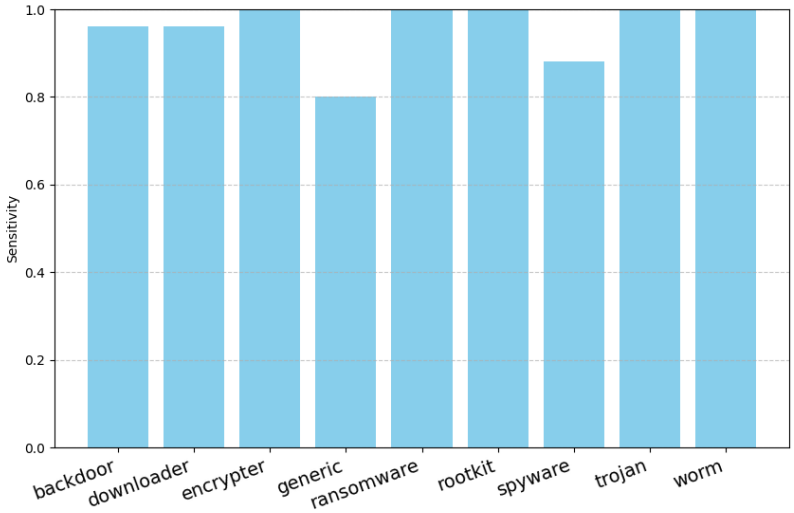



**Figura 6.11** Sensitivity per Joe Sandbox

### Meta Defender

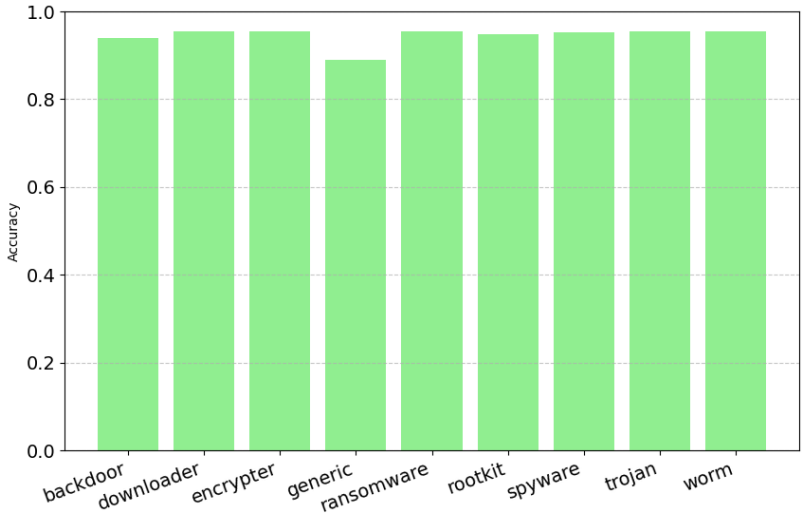



**Figura 6.12** Accuracy per Meta Defender

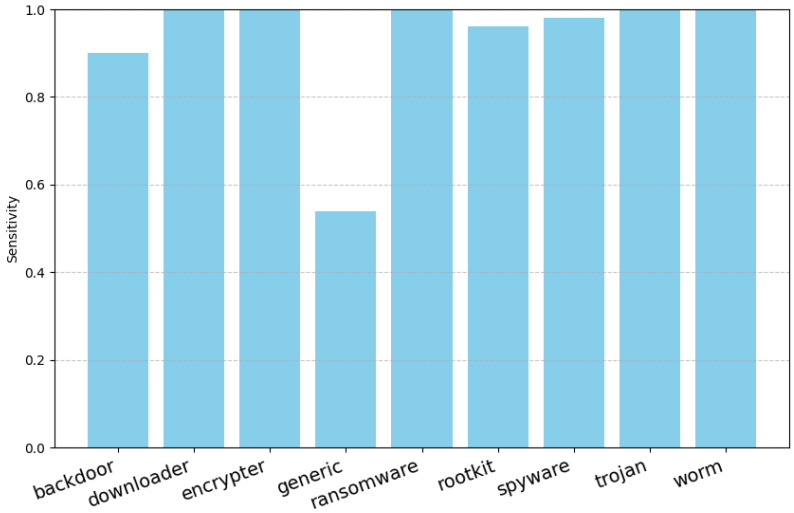



**Figura 6.13** Sensitivity per Meta Defender

### Virus Total

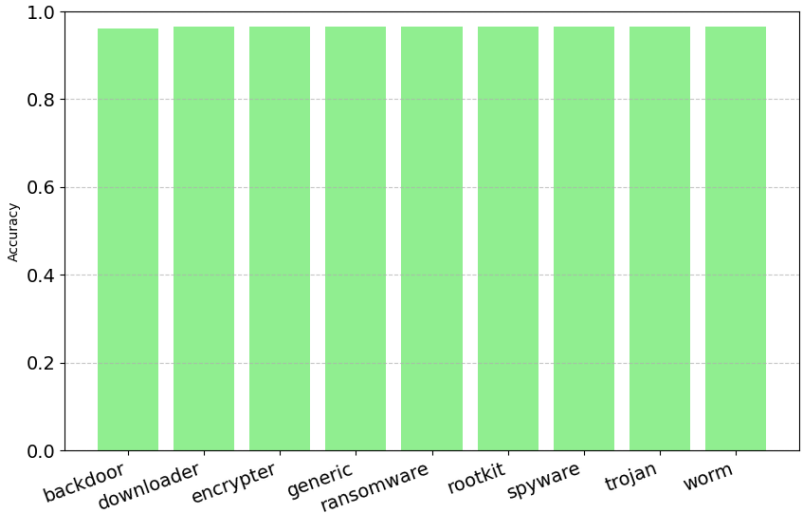



**Figura 6.14** Accuracy per Virus Total

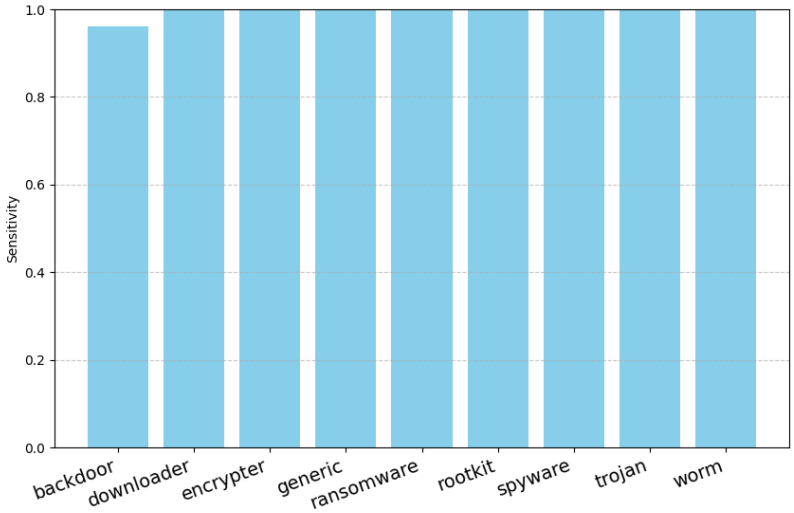



**Figura 6.15** Sensitivity per Virus Total

### Yara

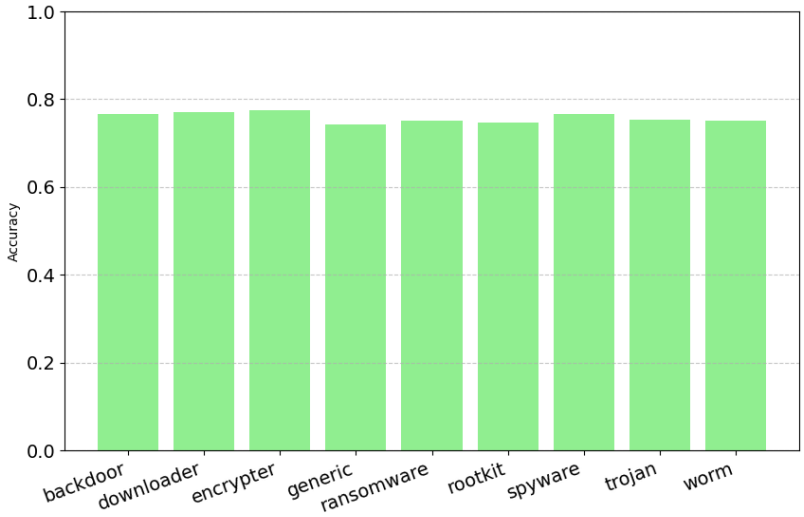



**Figura 6.16** Accuracy per Yara

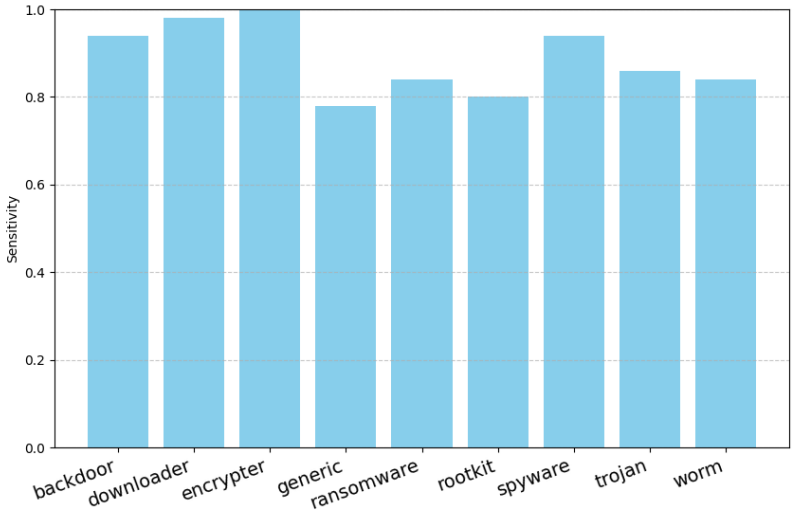



**Figura 6.17** Sensitivity per Yara

# Capitolo 7

# Conclusioni

Uno degli aspetti più significativi di questo tirocinio è stata l'analisi dei dati ottenuti dai test condotti su un ampio dataset di malware e file benigni. Grazie a questo processo, si è potuto valutare le prestazioni dei vari strumenti di rilevamento in termini di sensibilità, specificità e accuratezza. Questi risultati forniscono informazioni preziose sulle capacità e sui limiti di ciascun strumento.

Dall'analisi condotta emerge chiaramente che l'utilizzo di strumenti che adottano un approccio multi motore, combinando analisi dinamica e statica, si rivela il più efficace nel contesto dell'analisi del malware. Questo approccio permette di ottenere una visione completa e dettagliata del comportamento del malware, fornendo informazioni preziose per la sua identificazione e la sua classificazione. L'integrazione di più motori di scansione e l'analisi multi livello consentono di rilevare un'ampia gamma di minacce, garantendo una maggiore precisione e affidabilità nei risultati.

Al contrario, gli approcci che si basano su un singolo motore di scansione o che utilizzano esclusivamente analisi statica o dinamica hanno dimostrato limitazioni significative. Questi approcci potrebbero non essere in grado di rilevare determinate varianti di malware o di fornire informazioni sufficienti sul suo comportamento, compromettendo l'efficacia complessiva dell'analisi.

Inoltre, è emerso che l'adozione di strumenti on-premise, sebbene possa offrire un maggiore controllo sui processi di analisi, comporta rischi significativi per la sicurezza del sistema utilizzato. Gli strumenti on-premise possono esporre il sistema a potenziali minacce e vulnerabilità, oltre a richiedere risorse e competenze dedicate per la gestione e la manutenzione dell'infrastruttura.

In conclusione, per ottenere risultati ottimali nell'analisi del malware, è consigliabile adottare strumenti basati su un approccio multi motore che combinano analisi dinamica e statica. Questi strumenti offrono una maggiore precisione, affidabilità e copertura delle minacce, garantendo nel contempo un livello adeguato di sicurezza per l'infrastruttura informatica.

# Bibliografia

[1] Iosif Achim and Bogdan Ionescu. Dikedataset: A dataset of network traffic for ddos attack detection, 2023.

[2] Yara Project Team. Yara rules, 2023. Accessed on 2023-11-16.

[3] United States Senate Committee on Homeland Security and Governmental Affairs. Hsgac majority cryptocurrency ransomware report, 2022.

[4] Ömer Aslan Aslan and Refik Samet. A comprehensive review on malware detection approaches. *IEEE Access*, 8:6249–6271, 2020.

[5] Ori Or-Meir, Nir Nissim, Yuval Elovici, and Lior Rokach. Dynamic malware analysis in the modern era—a state of the art survey. *ACM Comput. Surv.*, 52(5), sep 2019.

[6] Hashem Hashemi and Ali Hamzeh. Visual malware detection using local malicious pattern. *Journal of Computer Virology and Hacking Techniques*, 15, 03 2019.

[7] Rami Sihwail, Khairuddin Omar, and Khairul Akram Zainol Ariffin. A survey on malware analysis techniques: Static, dynamic, hybrid and memory analysis. 8:1662, 09 2018.

[8] Alireza Souri and Rahil Hosseini. A state-of-the-art survey of malware detection approaches using data mining techniques. *Human-centric Computing and Information Sciences*, 8(1):3, 2018.

[9] William Stallings and Lawrie Brown. *Computer Security: Principles and Practice*. Pearson, Global Edition, 4th edition, 2018.

[10] Rabia Tahir. A study on malware and malware detection techniques. *International Journal of Education and Management Engineering*, 8:20–30, 03 2018.

[11] Hao Sun, Xiaofeng Wang, Rajkumar Buyya, and Jinshu Su. Cloudeyes: Cloud-based malware detection with reversible sketch for resource-constrained internet of things (iot) devices. *Software: Practice and Experience*, 47(3):421–441, 2017.

[12] Alexander Chailytko and Stanislav Skuratovich. Defeating sandbox evasion. *Virus Bulletin Conference*, 2016.

[13] Mingshen Sun, Xiaolei Li, John C.s Lui, Richard Ma, and Zhenkai Liang. Monet: A user-oriented behavior-based malware variants detection system for android. *IEEE Transactions on Information Forensics and Security*, PP, 12 2016.

[14] Syed Zainudeen Mohd Shaid. Malware behavior image for malware variant identification. 04 2015.

[15] Sudhir Kumar Pandey and B.M. Mehtre. Performance of malware detection tools: A comparison. In *2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies*, pages 1811–1817, 2014.

[16] Ilsun You and Kangbin Yim. Malware obfuscation techniques: A brief survey. pages 297–300, 11 2010.

[17] T. Abou-Assaleh, N. Cercone, Vlado Keselj, and R. Sweidan. N-gram-based detection of new malicious code. volume 2, pages 41– 42 vol.2, 10 2004.