POLITECNICO DI TORINO
Repository ISTITUZIONALE

Anatomia del malware

Original
Anatomia del malware / Mezzalama, Marco; Lioy, Antonio; Metwalley, H.. - In: MONDO DIGITALE. - ISSN 1720-898X. - ELETTRONICO. - 47(2013).

Availability:
This version is available at: 11583/2517684 since:

Publisher:
AICA

Published
DOI:

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

19 April 2024

Anatomia del malware

M. Mezzalama, A. Lioy, H. Metwalley

L'esistenza di malware (virus, worm, cavalli di Troia) è uno degli aspetti negativi più rilevanti della rivoluzione digitale, con risvolti penali e economici. Il fenomeno coinvolge anche il settore dei dispositivi mobili (smartphone, tablet, …) in cui si è passati da circa 2000 malware nel 2011, a più di 13000 nel 2012. Questo articolo analizza le tecniche per realizzare malware e permettere allo stesso di introdursi nei sistemi, rendersi residente, prenderne il controllo ed attivarsi a fronte di certi eventi, nascondendosi contemporaneamente ai programmi anti-virus. L'analisi considera sia i normali personal computer sia i più recenti dispositivi mobili.

Keywords: *Computer virus, Worm, Computer security for mobile devices, Antivirus*

1. Introduzione

L'informatica individuale e l'espansione delle comunicazioni Internet sono fenomeni di portata mondiale: circa il 35% della popolazione ha una connessione Internet, con picchi del 70-80% nelle nazioni più sviluppate [7]. Come ogni tecnologia pervasiva, anche l'informatica presenta "lati oscuri", e tra essi il "malware" è il fenomeno più rilevante, presente nel 90% dei cyber attacchi mondiali e che, solo nel 2004, ha provocato danni per oltre diciassette miliardi di dollari [3]. Il codice malevolo, però, è un fenomeno in continua crescita che ha ormai raggiunto numeri significativi: dal primo virus "Brain" (nel 1986) fino ad oggi sono stati creati oltre trenta milioni di programmi malevoli, con una crescita media di oltre il 100% negli ultimi anni.

Il termine *malware* deriva dalla contrazione di due parole inglesi, “malicious” e “software”, e indica programmi realizzati per danneggiare i sistemi su cui vengono eseguiti o per sottrarre informazioni sensibili. Nonostante spesso i programmi malevoli siano identificati genericamente come “virus informatico”, in realtà esistono tre grandi tipologie di malware: i *virus* sono programmi realizzati per creare danni e, successivamente, propagarsi inserendosi all’interno di altri file; i *worm*, invece, non hanno lo scopo di danneggiare direttamente il sistema ma cercano di auto replicarsi per saturare le risorse di sistema e di rete e, di conseguenza, creare danno; infine i *trojan horse* sono malware nascosti all’interno di legittime applicazioni software ma non possiedono codice dedicato alla propria replicazione e propagazione.

2. Struttura e organizzazoione del malware

Nonostante ogni tipologia di codice virale usi strumenti, tecnologie e tattiche diverse, tutti possiedono un unico modello strutturale, basato sulle quattro fasi che il codice virale percorre (figura 1):

- **Infezione.** Il malware s’introduce all’interno del sistema, superando eventuali barriere di sicurezza, e s’installa al suo interno. In seguito il malware modifica le impostazioni del sistema adattandole alle proprie necessità, in primis quella di non essere rilevato.
- **Quiescenza.** Il codice virale rimane residente in memoria in attesa che si realizzi una determinata condizione, a seguito della quale si attiva (esecuzione delle azioni malevole e replicazione). Queste ultime possono essere ripetute più volte, quindi la fase si protrae fino a un’eventuale eliminazione da parte del codice stesso o di un software anti-malware.
- **Replicazione e propagazione (solo per virus e worm**). Al determinarsi di certi eventi o condizioni, il malware si replica e seleziona i bersagli verso cui propagarsi, infettando altri sistemi.
- **Azioni malevole.** Al verificarsi di certi eventi o condizioni, il codice virale esegue i propri compiti malevoli, come distruzione o furto dei dati del sistema. Se il sistema non viene compromesso definitivamente, il software ritorna nella fase di quiescenza.

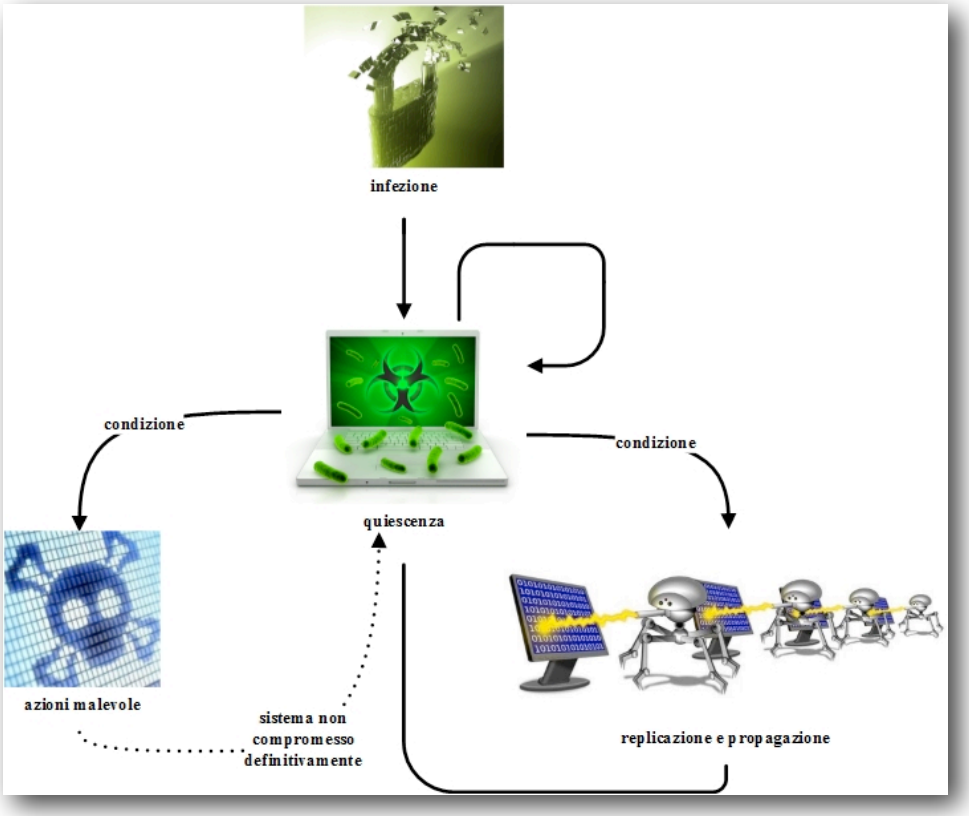

Figura 1
Ciclo di vita di un malware

2.1. Infezione

Il malware deve prima di tutto introdursi all'interno del sistema bersaglio e, se necessario, indurre la vittima a eseguirlo. La penetrazione passa in molti casi attraverso l'esecuzione di un file contenente, in modo diretto o indiretto, il codice virale. È ovvio che un codice malevolo non deve presentarsi in forma esplicita e, quindi, in questa fase spesso la parte malevola viene mascherata all'interno di programmi apparentemente legittimi attivati dall'utente in modo esplicito. Il primo compito di un malware è di inserirsi all'interno del sistema bersaglio e ciò avviene tramite tre principali canali:

1. **Trasferimento fisico.** Questa modalità, molto diffusa in passato ma ancora oggi frequente, prevede l'uso di un supporto di memorizzazione (come floppy disk, CD o unità USB) per il trasporto e l'inserimento del malware nei sistemi.

Le operazioni di trasporto e inserimento possono essere svolte dall'utente malevolo o, inconsapevolmente, dalla vittima stessa e prevedono un accesso fisico al PC. Stuxnet, uno dei malware più celebri e devastanti nella storia dei codici virali, usava questa tattica, inserendosi all'interno dei sistemi informatici tramite penne USB.

2. **Posta elettronica.** In questo caso il malware è allegato a messaggi di posta elettronica. L'utente viene così invitato in modo "suadente" ad aprire l'allegato, che può essere un file eseguibile o anche un documento elettronico (come un file Word o PDF contenente una macro pericolosa). "Melissa" è il primo malware che ha usato questo canale di diffusione ed ha infettato in breve tempo oltre ottantamila sistemi, causando danni a livello mondiale per più di un miliardo di dollari.

3. **Web.** Questo è attualmente il canale di diffusione più frequente e trasmette il codice malevolo attraverso un download da una pagina web. In particolare, in questo processo la vittima può svolgere un ruolo attivo o passivo. Nel primo caso, l'utente contrae il malware scaricando un file apparentemente innocente che, invece, si rivela essere malevolo. Nel secondo caso, invece, l'aggressore usa una serie di tecniche (note col nome di "drive-by download") che permettono la trasmissione di un malware con la semplice apertura di una pagina web. Quest'attacco sfrutta una o più vulnerabilità[1] presenti nei browser web ed evita, perciò, l'interazione con la vittima.

Una volta inseritosi nel sistema vittima, il malware deve attivarsi. Ciò richiede che l'utente debba forzatamente aprire il file contenente il codice virale. Questo contenuto, quindi, deve sembrare il più possibile inoffensivo e per questo motivo esistono tecniche di "mascheramento" per includere codice malevolo all'interno di contenuti leciti. Un caso tipico è quello dei virus informatici, che s'introducono autonomamente all'interno di altri file, solitamente eseguibili, modificandone la struttura interna. I file eseguibili possiedono una parte d'intestazione (in inglese, *header*) contenente informazioni destinate al sistema operativo, tra cui anche un puntatore che indica l'inizio del programma, cioè la prima istruzione macchina da eseguirsi all'attivazione del programma. Al momento dell'infezione, il virus si accoda al codice originale dilatando il file e, successivamente, sposta il puntatore originale al codice malevolo e ne crea un altro indirizzato verso l'istruzione di partenza originale, in modo che all'apertura del file il sistema operativo esegua prima il contenuto illecito e, in seguito, quello legittimo. Un esempio di questa tecnica è illustrata in figura 2, che mostra la differente struttura di un file normale e uno infettato [12].

---

[1] *Componente hardware o software, in corrispondenza alla quale le misure di sicurezza sono assenti, ridotte o compromesse. Esso rappresenta un punto debole e consente a un eventuale aggressore di compromettere il livello di sicurezza dell'intero sistema.*

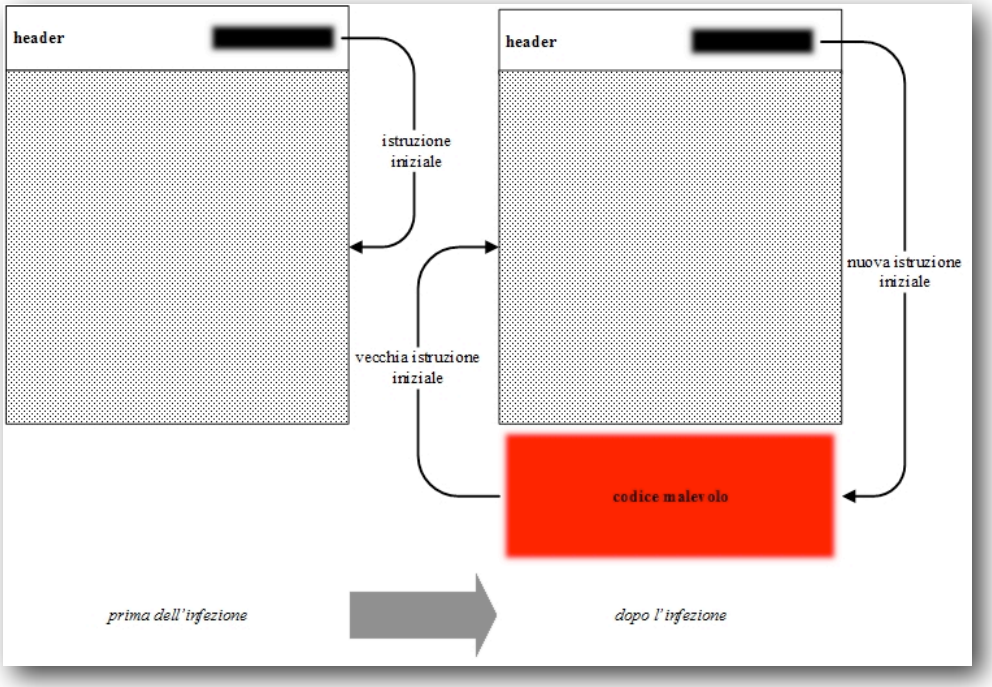

Figura 2
Mascheramento di un virus informatico

I worm, come i trojan, non possiedono la capacità di inserirsi autonomamente in altri file, ma vengono inclusi all'interno di programmi apparentemente legittimi. Per svolgere questa operazione l'utente malevolo usa di norma un *wrapper* (tra cui i più celebri sono "AFX File Lace", "Trojan Man" e "EliteWrap" [12]), uno strumento che consente di combinare due o più programmi all'interno di un solo file eseguibile. In questo modo, la vittima è convinta di aprire un programma apparentemente innocuo che, invece, contiene al proprio interno uno o più malware. Esempi sono il mascheramento in false applicazioni di sicurezza, aggiornamenti software o banner.

Per operare nel sistema, accedendo a tutte le sue risorse, il malware modifica le impostazioni del sistema per prenderne il controllo. Questa fase richiede lo sfruttamento di una vulnerabilità che permetta di ridurre o annullare le misure di sicurezza. Tecnicamente, quest'operazione si traduce nell'aumento dei privilegi di sistema attribuiti al programma malevolo: nell'ambito informatico, il privilegio classifica il livello di autorizzazioni fornite a un determinato programma e, conseguentemente, ad ogni sua azione. Di norma esistono due livelli, denominati "utente" e "amministratore" (o "root", in inglese), a seconda che un'operazione possa modificare componenti importanti del sistema, come le operazioni di I/O o le tabelle di sistema. Per acquisire elevati privilegi di sistema, un codice malevolo sfrutta una o più vulnerabilità. Le più diffuse sono di tipo "memory error", che

sfruttano criticità nella gestione di zone di memoria come stack o heap. La tipologia di attacco più conosciuta e usata di questa categoria è il "buffer overflow". Questa tecnica, usata con successo contro numerosi programmi, come Adobe Reader, Microsoft Internet Explorer o Microsoft SQL Server, si basa sul fatto che, quando si attiva una routine, il suo indirizzo di ritorno viene memorizzato nella struttura dati denominata stack. L'utente malevolo cerca così di posizionare una serie di dati fuori dai limiti di un buffer[2]. Alcuni linguaggi di programmazione, come Java, effettuano automaticamente controlli sul corretto uso del buffer, segnalando eventuali anomalie durante la fase di esecuzione del programma. Altri linguaggi, come il C, non operano alcuna verifica sulla corretta allocazione delle variabili, che quindi possono essere posizionate in indirizzi di memoria occupati da altri processi. Se, quindi, il programmatore non controlla scrupolosamente le dimensioni delle strutture dati prima dello spostamento in memoria, il programma diventa vulnerabile ad attacchi di questo tipo. Per sfruttare tale difetto, un utente malintenzionato (o direttamente un malware) invia all'applicazione una quantità di dati maggiore dello spazio assegnato in memoria, con l'obiettivo di provocarne un overflow della struttura e comprometterne zone critiche. Lavorando con attenzione i dati inviati al bersaglio, un attaccante può quindi eseguire qualsiasi tipo di codice malevolo e compiere facilmente azioni illecite senza l'autorizzazione della vittima [1].

La figura 3 illustra il medesimo stack: a sinistra correttamente allocato, a destra, invece, con un attacco di buffer overflow. Nell'uso appropriato, lo stack è usato per immagazzinare una stringa di dodici caratteri (in cui è inserita la parola "hello"). L'elemento più importante di questa struttura è il "Return Address", che contiene l'indirizzo dell'istruzione da cui un programma deve riprendere la propria esecuzione dopo la chiamata di una eventuale funzione. Nel secondo caso, invece, è stata inserita la stringa "AAAAAAAAAAAAAAAAAAAAx08x35xC0x80" che, superando il limite dei dodici caratteri, provoca l'inserimento di dati illeciti nell'area di memoria dedicata all'elemento precedentemente illustrato. Per un utente malevolo è quindi possibile preparare una stringa per sovrascrivere il campo return address con l'indirizzo di un'area di memoria contenente un codice malevolo, che verrà eseguito dal sistema con i privilegi dell'applicazione chiamante. Per questo motivo, scoprire una vulnerabilità di questo tipo in un programma che possiede privilegi elevati permette ad un utente malevolo di compiere qualsiasi operazione desiderata [1].

[2] Un *buffer* è una sequenza di lunghezza predefinita di celle di memoria, concettualmente molto simile a un vettore.

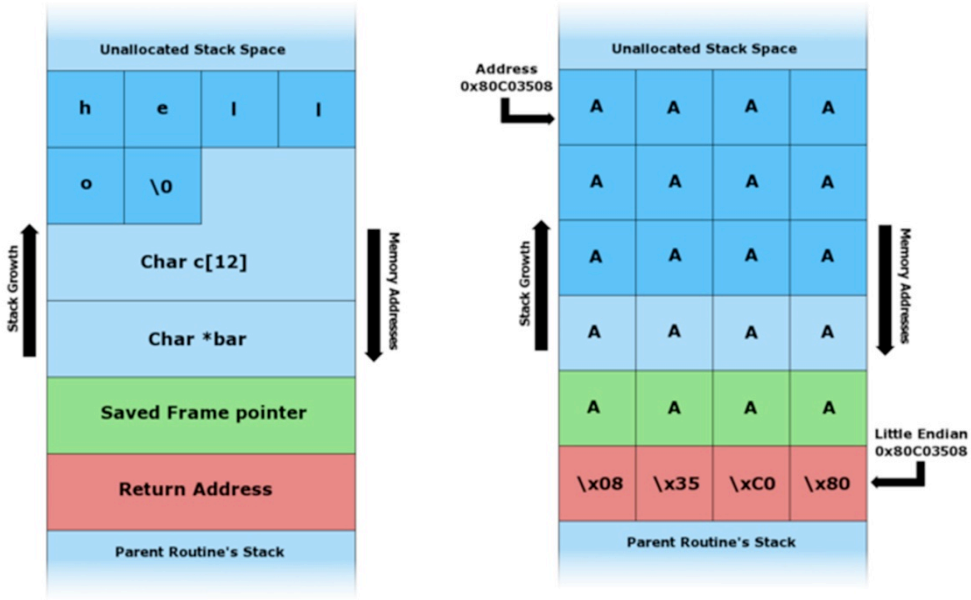

Figura 3
Stack correttamente allocato (a sinistra) e con buffer overflow *(a destra)*

Ottenere i privilegi di sistema permette al malware di eseguire qualsiasi operazione sul sistema vittima, tra cui la più importante – nel caso di Windows – è la modifica del "registro di sistema", una base dati che contiene al proprio interno le impostazioni del sistema operativo e ne regola il funzionamento. I campi di questa base dati (detti anche chiavi di registro), inaccessibili senza privilegi di amministratore, permettono di modificare il comportamento del sistema adattandolo ai bisogni del malware. Per un programma malevolo queste modifiche sono indispensabili perché gli permettono di accedere a risorse di sistema protette e di caricarsi in memoria ad ogni avvio del sistema operativo. Da recenti statistiche della nota azienda di sicurezza F-Secure, infatti, risulta che oltre il 40% dei malware modifica la chiave di registro "Run" che impone al sistema operativo di eseguire un determinato processo al proprio avvio [4]. In questo modo il programma malevolo si assicura la propria esecuzione. Gli esempi di malware che usavano questa chiave di registro sono numerosi: Sasser inseriva all'interno della chiave il valore "avserve.exe=%WinDir% \avserve.exe" (ove %WinDir% indica la cartella di installazione di Windows) mentre Sober inserisce copie di se stesso in tre file e crea tre chiavi di registro del tipo "<nome_casuale>=%System%\<nome_del_file_infetto>" [11].

2.2. Quiescenza

Completata la fase di penetrazione e infezione, il malware rimane residente in memoria, in attesa di una determinata condizione che lo avvii. Durante questa fase, è quiescente ma vigile per auto-proteggersi da eventuali rilevamenti da parte della vittima o di un software di sicurezza.

La maggior parte dei malware rimangono residenti in memoria e quindi qualsiasi utente esperto può individuarlo tramite un'applicazione *task manager* che visualizza i processi in esecuzione nel sistema operativo, come "Gestione attività" in Windows. Per questo motivo, spesso gli attaccanti assegnano al processo malevolo il nome di un programma presente nel sistema. Per esser sicuri, inoltre, che essi siano realmente esistenti, vengono usati i nominativi di processi presenti di default nel sistema operativo, come "iexplorer" (processo di Microsoft Internet Explorer) o "notepad" (editor di testo di Microsoft Windows), o di applicativi che non desterebbero sospetto, come "win" (processo inesistente ma con un nome plausibile in un sistema Windows). Anche se la vittima scovasse la presenza del malware, non saprebbe quale tra i due processi (quello legittimo e quello illegittimo) eliminare. Per questo motivo, difficilmente l'utente elimina il processo malevolo, anche perché sbagliando la scelta verrebbe compromessa la stabilità del sistema operativo, in quanto spesso i malware assumono le sembianze di processi indispensabili per il corretto funzionamento [1].

Oltre a proteggere il malware dal rilevamento dell'utente, gli sviluppatori adottano anche tecniche di mascheramento per eludere i controlli di un eventuale anti-virus. Questi controlli possono sfruttare tecniche statiche o dinamiche.

2.3. Tecniche di rilevamento statiche

I metodi statici prevedono la rilevazione dei malware ricercando codice malevolo all'interno di file presenti su unità di memoria o flussi dati trasmessi in rete. L'idea è di attribuire ad ogni file una "firma" (basata su algoritmi di hash) che lo identifica univocamente e perde validità qualora il file venga modificato. La rilevazione di uno specifico malware si basa sulla rilevazione di sequenze univocamente associate al codice virale [1], anch'esse codificate con algoritmi di hash e note come *virus signature*. Il software di sicurezza scandisce il file e ricerca eventuali corrispondenze tra i dati e le sequenze dei codici virali noto. L'uso di questa tecnica implica la conoscenza e l'analisi di tutti i malware. Essa è quindi efficace contro gli attacchi noti ma è inutile contro i malware *zero-day*: nei giorni successivi alla creazione di un nuovo codice virale, esso non sarà ancora presente nel database degli anti-virus e sarà quindi in grado di superare i controlli di sicurezza. I produttori di anti-virus affrontano questo problema rilasciando periodicamente degli aggiornamenti del database per riconoscere i nuovi codici malevoli.

Per proteggere il sistema dal giorno del rilascio del malware fino all'aggiornamento del database vengono usate tecniche euristiche statiche, che permettono di riconoscere un codice malevolo con un alto livello di accuratezza senza, però, avvalersi delle signature. Questo tipo di tecnica prevede una prima fase di raccolta dati, durante la quale viene usato un certo numero di euristiche per esaminare il file in questione. Ogni singola euristica fa riferimento ad una particolare caratteristica virale, come la presenza di codice spazzatura o cifrato, l'uso di librerie insolite o la presenza di istruzioni non generate solitamente dai

compilatori o che modificano il vettore degli interrupt. Dopo aver raccolto questi dati, essi vengono analizzati, associando ad ogni euristica un grado di pericolosità e stilando un report che raccoglie le statistiche riguardanti il file scansionato. A seconda dei valori raggiunti dai parametri del report, il contenuto viene classificato come pulito, sospetto o infetto.

Per nascondersi dai controlli statici degli anti-virus, gli utenti malevoli adottano diverse tecniche di "offuscamento" del codice, per renderlo illeggibile o mutevole nel tempo. L'idea di base prevede che il corpo del malware (cioè la sequenza di istruzioni binarie che lo costituiscono) venga cifrato in modo che il codice virale non possa essere identificato mediante il rilevamento delle signature predefinite. La parte cifrata non può tuttavia essere eseguita se prima non viene decifrata; pertanto devono essere presenti anche istruzioni binarie e non cifrate eseguite all'avvio del malware. L'operazione di cifratura prevede l'uso di un'apposita chiave, nota come "*stub*", per cifrare e decifrare il contenuto malevolo. Esistono diverse tecniche di cifratura ma quella più usata e semplice sfrutta una chiave segreta secondo i principi della crittografia simmetrica [3]. Ovviamente tale chiave non è statica, ma viene modificata di volta in volta per rendere il codice malevolo cifrato sempre diverso. Ad esempio, assumendo come chiave il contenuto di un'apposita area di memoria, è possibile mutare ad ogni infezione la chiave e, insieme ad essa, il corpo del codice virale.

Oltre alla cifratura, viene spesso usato un algoritmo di compressione per ridurre la dimensione del codice malevolo. Quest'operazione è spesso necessaria per evitare che la vittima venga insospettita da un'occupazione anomala di memoria. Questa tecnica, nota come "*pacchettizzazione*" (figura 4), permette di comprimere e cifrare il codice malevolo inserendogli in testa una sequenza di istruzioni per lo spacchettamento. Quest'approccio è ampiamente usato nei codici virali più recenti: secondo statistiche Symantec, oltre l'80% dei malware la usa.

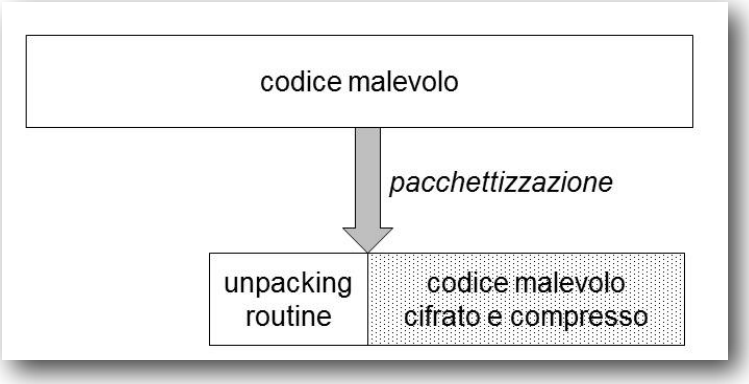

Figura 4
Pacchettizzazione di un codice malevolo

Il pregio di questa tecnica rappresenta però anche la sua principale debolezza: infatti, il codice binario che svolge le operazioni di cifratura e compressione (e viceversa) rimane identico ad ogni infezione e può perciò essere facilmente rilevato dagli anti-virus. Per risolvere questo problema, i malware più sofisticati e innovativi integrano al loro interno algoritmi in grado di variare le routine di cifratura e compressione. Ad esempio, per mutare una sequenza di istruzioni possono essere inserite istruzioni che non alterano il funzionamento del programma. Si consideri l'esempio in figura 5, dove le due serie di istruzioni hanno diverse sequenze binarie e quindi signature, ma svolgono le medesime operazioni. Infatti inserendo l'istruzione NOP (che non svolge alcuna operazione), il codice cambia ma il risultato è invariato. A seconda delle tecniche usate per la mutazione dei componenti di cifratura e compressione, si parla di malware "*oligomorfi*" (varie routine di cifratura ma in numero limitato), "*polimorfi*" (infinite varianti di cifratura) e "*metamorfi*" (infinite varianti del codice malevolo ma senza fare uso della cifratura). Collettivamente si parla di malware di tipo "*stealth*" (invisibile) per tutte queste tecniche di offuscamento [2].

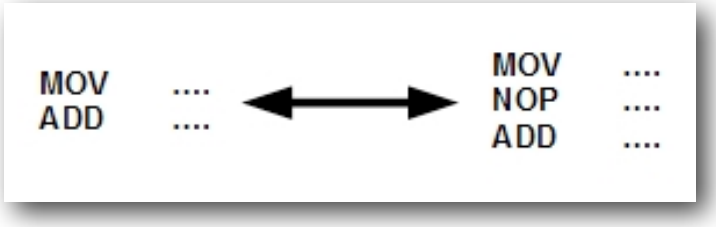

Figura 5
Esempio di istruzioni con uguale funzionamento ma signature differente

2.4. Tecniche di rilevamento dinamiche

Gli anti-virus usano anche tecniche dinamiche per rilevare il malware eseguendone il codice o controllandone il comportamento.

Il monitoraggio del comportamento (conosciuto anche come "*behaviour blocker*") è un componente dell'anti virus, solitamente residente in RAM, che controlla in tempo reale i programmi in esecuzione cercando comportamenti sospetti. In pratica, questi programmi analizzano tutte le operazioni svolte nel sistema, come quelle di lettura e scrittura su disco o gli accessi ad aree di memoria particolari. Sulla base di questi sintomi, il software di sicurezza riesce a rilevare programmi che eseguono azioni sospette e che, pertanto, sono potenzialmente dei malware. Se viene rilevata una azione allarmante, il software può bloccare l'operazione corrente, terminare il programma sospetto o chiedere all'utente di scegliere quale azione intraprendere [1]. Il pregio del monitor rappresenta però anche un suo punto debole, perché richiede l'esecuzione di un codice potenzialmente dannoso sul sistema reale. Per evitare questo problema, alcuni anti-virus emulano un

ambiente apposito per controllare il comportamento di un malware senza danneggiare il sistema reale: ad esempio, si analizza un codice cifrato eseguendolo su una macchina virtuale dedicata

Anche in questo caso gli scrittori di malware hanno messo in atto contromisure. Esistono infatti diversi strumenti, tra cui uno dei più famosi è "*red-pill*" [10], che rilevano se il programma è eseguito su una CPU reale o emulata. Integrando nel malware questi strumenti, è possibile eseguire il codice malevolo solo in presenza di un ambiente reale, evitando comportamenti illeciti in presenza di emulazione e superando i controlli imposti dal software di sicurezza.

I virus più recenti hanno iniziato a sfruttare anch'essi la virtualizzazione, eseguendosi ad un livello inferiore rispetto ai software di sicurezza. Questi programmi malevoli s'inseriscono all'interno del PC e convertono di nascosto un sistema fisico in uno virtuale, pienamente controllato dal malware che risulta così invisibile ai software di sicurezza che operano ad un livello superiore.

Oltre a nascondere il malware, durante questa fase il supervisor regola l'avvio delle azioni malevole e delle routine di replicazione e propagazione. Le fasi successive sono regolate da un componente aggiuntivo, il "*trigger*", che le attiva al raggiungimento di una particolare condizione impostata dal realizzatore del malware. Gli eventi che scatenano il trigger possono essere molteplici, come ad esempio una data (ad esempio il virus "Jerusalem" si attivava ogni venerdì 13) o una particolare operazione eseguita dalla vittima.

Oltre ai malware elencati fino a questo punto, che operavano ad un livello strettamente locale, infettando un solo sistema per volta, esiste una particolare categoria di codice virale, conosciuta come "malware distribuito", in cui ogni sistema infetto entra a far parte di una "botnet", una rete formata da dispositivi collegati a Internet e controllati da un'unica entità, il "botmaster". Una botnet può essere formata da milioni di sistemi infetti (una delle più grandi, chiamata "Bredolab", era composta da oltre 30 milioni di PC), che vengono chiamati "bot" o "zombie". I malware distribuiti possiedono un kernel che, oltre ad eseguire le normali operazioni, è in grado di contattare uno o più server remoti per scambiare informazioni. Per svolgere questa operazione, sfruttano i servizi di chat, connettendosi a un determinato canale protetto da una password. Per controllare e impartire ordini ai sistemi infetti vengono, inoltre, usati gli stessi canali di chat o reti peer-to-peer appositamente realizzate. Le botnet vengono usate per compiere attacchi "distributed denial of service" (DDoS) o campagne di spam: i DDoS mirano a compromettere il funzionamento di un sistema informatico inviando numerosi messaggi fino a esaurirne le risorse; le campagne di spam, invece, puntano a reclamizzare un determinato prodotto, inviando molte mail pubblicitarie.

2.5. Replicazione e propagazione

Parallelamente all'esecuzione delle azioni malevole, il malware deve anche replicarsi e selezionare nuovi obiettivi verso cui propagarsi. Questa fase è presente nei virus informatici e nei worm ma non nei trojan horse, che non si replicano automaticamente. Virus e worm possiedono, però, tattiche e comportamenti diversi: il primo punta a inserirsi all'interno di altri file e poi sfruttare la vittima per propagarsi, mentre il secondo si moltiplica e sfrutta la rete per propagarsi autonomamente [3].

I virus usano diverse tecniche per introdursi all'interno di altri file, solitamente eseguibili, e diffondersi. I virus di tipo "*parasitic*" inseriscono il proprio componente malevolo all'interno di porzioni di codice del file eseguibile da infettare. A seconda di dove venga inserito il codice virale, esistono diverse tipologie di questo tipo di malware: il *prepending*, che include il codice prima di quello originale, l'*appending*, che si salva dopo i contenuti originali, e il *cavity*, che si inserisce all'interno di eventuali spazi inusati nei settori di disco contenenti il file originale. I virus informatici, però, non compromettono solamente i file eseguibili ma si introducono anche in altre tipi di file, come boot sector o documenti. I boot sector virus infettano il settore di avvio presente nelle unità disco rimovibili (come unità USB o i vecchi floppy disk) o rigidi (hard disk) e ne assumono il controllo. Questo tipo di malware viene eseguito a ogni avvio del PC infetto prima del caricamento del sistema operativo e rimane in memoria fino allo spegnimento. Durante l'infezione, il virus si copia all'interno delle unità inserite nel lettore, che diventano un veicolo di diffusione se usati per avviare altri dispositivi, o danneggia i dati contenuti all'interno del sistema vittima.

Una particolare tipologia di virus informatici è rappresentata dai "macro virus", che non infettano file contenenti codice eseguibile ma file dati, inserendosi all'interno di documenti contenenti macro definizioni, sequenze di istruzioni scritte in linguaggio "Visual Basic for Application" e usate nei programmi della suite Microsoft Office per aumentare le potenzialità dei documenti di testo. Generalmente, per assicurarsi di essere eseguiti, essi infettano i modelli standard (nel caso di Word, il file "Normal.dot") che vengono presentati all'apertura di un programma di Office. In questo caso, ogni nuovo documento creato è automaticamente infetto. Un'altra tecnica usata da questi virus consiste nel modificare le macro associate alle voci di menu (ad esempio "Apri", "Salva" o "Salva con nome"): in questo modo, ogni volta che la vittima seleziona una di queste voci, il sistema esegue anche il codice malevolo, dando automaticamente inizio all'infezione di nuovi documenti [3].

I worm, al contrario dei virus informatici, non hanno capacità autonome di inserirsi in quanto, come detto in precedenza, sono gli stessi utenti malevoli ad inserirli in programmi leciti tramite i "wrapper". I worm, quindi, costituiscono un file a se stante e non hanno bisogno di infettarne altri. Per questo motivo, i worm possiedono tattiche di replicazione poco sofisticate, in quanto essi puntano

semplicemente a realizzare copie di se stessi mirando, quindi, ad infettare i PC e non i file. Questo tipo di malware adotta un sistema di propagazione particolarmente complesso costituito da due componenti, il "Target Selection Algorithm" e lo "Scanning Engine". Il primo modulo seleziona gli indirizzi dei nodi collegati alla rete che costituiscono potenziali bersagli. Per realizzare questo operazione i worm usano diversi stratagemmi [3]:

- **Esplora risorse di rete.** Il malware esplora la rete simulando le richieste di un qualsiasi utente in cerca degli indirizzi dei sistemi vicini.
- **Richiesta DNS.** Il worm si collega al server DNS associato al sistema vittima e ottiene gli indirizzi di rete di altri nodi.
- **Contatti mail.** Il malware legge la lista dei contatti di posta elettronica dell'utente infetto. In questa modo, chiunque abbia scambiato messaggi di posta con la vittima diventa un potenziale bersaglio.

Lo scanning engine, invece, ha il compito di contattare gli indirizzi selezionati dal target selection algorithm e verificare se siano presenti le condizioni per una eventuale infezione. I worm, infatti, si propagano autonomamente sfruttando vulnerabilità presenti nei sistemi vittima. Lo scanning engine, quindi, inoltra verso il bersaglio una serie di pacchetti creati appositamente per verificare se il sistema sia vulnerabile agli exploit del malware e, se questa condizione si verifica, penetra al suo interno e lo infetta. L'efficacia e la velocità di questo componente è fondamentale per un worm, che ha come prerogativa la velocità di diffusione (Slammer infettò oltre 75000 server in soli dieci minuti) [3].

3. Il malware per dispositivi mobili

L'interesse degli utenti malevoli verso i dispositivi mobili nasce dagli anni 2000. Come si vede dalla figura 6, che illustra il numero totale di malware esistenti per dispositivi mobili secondo statistiche McAfee [9], in quel periodo i codici virali (consistenti in poche decine di unità) erano ancora poco presenti e non rappresentavano un problema. Le motivazioni di questa tendenza erano da ricercare, principalmente, nelle basse possibilità di guadagno (per gli utenti malevoli) e nelle limitate potenze di calcolo degli smartphone e della rete telefonica.

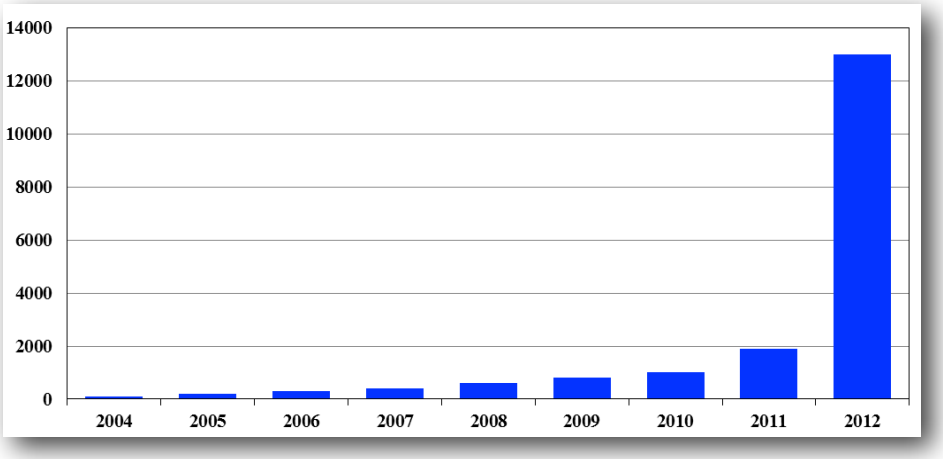

Figura 6
Tasso di crescita del malware per dispositivi mobili

La situazione cambia radicalmente a partire dal 2010, quando si realizzano una serie di condizioni che aumentano l'interesse degli utenti malevoli e quindi anche la creazione di malware. Infatti, attualmente esistono numerose congiunture che inducono gli aggressori a dedicarsi in maniera quasi esclusiva a questo settore:

- **Aumento delle vendite.** Le vendite dei soli smartphone hanno superato quelle di desktop e notebook nel 2011, con i tablet che effettueranno il sorpasso a breve. L'incremento vertiginoso della base di installazione di sistemi operativi mobili rappresenta uno dei principali fattori di interesse per gli attaccanti, attirati dalla possibilità di diffondere i propri malware su milioni di dispositivi.

- **Incremento delle prestazioni.** La potenza di calcolo degli smartphone è aumentata. Basti pensare, infatti, che le ultime generazioni di dispositivi possiedono processori quad core e elevati quantitativi di memoria RAM. Questa condizione permette di concepire malware potenti e devastanti quanto quelli esistenti su sistemi non mobili.

- **Aumento della velocità di rete.** Anche le reti dati hanno subito un incremento di prestazioni ragguardevole. Le ultime generazioni di collegamenti, infatti, possono raggiungere la velocità di un Gigabit per secondo. Le reti telefoniche, in accoppiata a quelle wireless, rappresentano un ambiente ideale per la diffusione e la realizzazione di botnet.

- **Sistemi operativi.** Le ultime generazioni di dispositivi mobili dispongono di sistemi operativi complessi ed avanzati come per PC, dove qualsiasi sviluppatore può creare liberamente applicazioni e, quindi, concepire codici virali senza alcuna restrizione.

- **Presenza di nuovi sensori.** Le ultime generazioni di smartphone e tablet possiedono sensori inesistenti su PC, come GPS, microfono o fotocamera, utili per ottenere preziose informazioni circa l'utente.

La prerogativa dei dispositivi mobili, quindi, è quella di avere sempre a portata di mano le informazioni e i contatti personali dell'utente, accessibili da qualsiasi luogo appoggiandosi a reti audio e dati senza fili. Gli smartphone e, recentemente, i tablet, contengono dati non solo personali ma anche di terzi, come quelli di amici e colleghi (i loro contatti, messaggi, appuntamenti, appunti condivisi e posizioni). Per questo motivo, gli obiettivi dei malware per dispositivi mobili sono mutati rispetto alla controparte per PC. Infatti, mentre i codici virali tradizionali miravano a realizzare una gamma molto variegata di attacchi, come denial of service, distributed denial of service, controllo del sistema da remoto oppure cancellazione o furto di dati sensibili, le applicazioni malevole per dispositivi mobili si sono focalizzate sull'ultima tipologia di attacchi.

Il mercato dei dispositivi mobili è attualmente dominato da due sistemi operativi: "Google Android" e "Apple iOS". Nonostante siano molto diversi tra loro, essi usano politiche di sicurezza simili:

1. **protezione contro accessi fisici non autorizzati –** l'utente può inserire meccanismi di protezione basati su codici PIN o segreti più complessi, che impediscono ad un utente malevolo di avere accesso al dispositivo senza l'autorizzazione del proprietario;

2. **sandbox –** ogni applicazione viene eseguita in una macchina virtuale dedicata (detta anche "sandbox"), che impedisce ad un codice malevolo di interagire con altri programmi in esecuzione o con lo stesso sistema operativo;

3. **cifratura dei contenuti –** vengono usati algoritmi per proteggere i contenuti personali dell'utente rendendoli illeggibili in caso di furto del dispositivo;

4. **firma delle applicazioni –** ogni applicazione è accompagnata da una firma digitale, per impedire che un malware possa inserirsi al suo interno.

Nonostante queste misure di sicurezza siano simili, la quasi totalità dei malware viene sviluppata per Android, interessando solo in minima parte iOS. Secondo le statistiche, il sistema operativo di Google raccoglie più dell'80% delle nuove minacce, mentre solo l'1% delle nuove famiglie di malware interessa iOS [5, 9]. Le motivazioni di questa tendenza sono da ricercare nelle diverse filosofie adottate da Google e Apple: Android, infatti, è un sistema operativo open-source in cui il proprietario del dispositivo ne ha un completo controllo, mentre iOS è una piattaforma chiusa che lascia molte meno possibilità di personalizzazione all'utente. Il diverso livello di apertura di un sistema operativo non si traduce solo in una diversa possibilità di personalizzazione, ma anche in un diverso modello di sicurezza. Apple, infatti, controlla meticolosamente ogni sorta di applicazione

presente nel proprio canale di diffusione ufficiale, l'*App Store*, verificando l'identità degli sviluppatori e accertando la legittimità del codice da essi realizzato. Google, invece, effettua controlli più laschi sugli sviluppatori e sulle applicazioni disponibili sul proprio canale *Google Play*. Inoltre su qualsiasi dispositivo Android è possibile installare programmi esterni a Google Play e perciò un utente poco istruito circa le tematiche di sicurezza è esposto a tutti i pericoli provenienti dal malware. Queste motivazioni hanno portato allo sviluppo di numerosi codici virali per questa piattaforma, con un trend che, a partire dal 2010, ha subito una incremento continuo: ogni anno si assiste ad un crescita del malware per Android vicina al 100% [5].

I codici virali esistenti per Android possiedono un ciclo di vita del tutto identico a quelli presenti su PC, anche se la fase che ha subito maggiori cambiamenti è l'infezione. Anche in questo caso, quindi, la prima fase rappresenta un passaggio delicato per il malware, che deve adottare tecniche atte a non farsi trovare dall'utente, nel caso non ne sia richiesta l'interazione, o a convincerlo della propria legittimità, nel caso sia indispensabile che la vittima autorizzi l'installazione dell'applicazione. Nel passaggio da PC a mobile, i codici virali hanno però mutato i propri comportamenti e le proprie tattiche e, nel caso dell'inserimento iniziale nel dispositivo, questo cambiamento si è tradotto nella perdita di alcuni canali tradizionali, come il trasferimento fisico e la posta elettronica, a favore del web o dei "negozi" di applicazioni (detti anche comunemente "market"). Queste due tipologie di canali trasmissivi ben classificano le tattiche disponibili per i malware per dispositivi mobili: lo sfruttamento di vulnerabilità o l'inclusione di un codice virale in un'applicazione.

La prima possibilità prevede lo sfruttamento di una o più vulnerabilità per l'introduzione del malware e l'acquisizione dei privilegi a lui necessari. Solitamente vengono usate almeno due debolezze, una presente nel software di navigazione, per realizzare un attacco drive-by download e introdurre il programma malevolo, ed una seconda, presente direttamente nel sistema operativo, per acquisire i privilegi di amministratore e permettere al malware di installarsi silentemente senza richiedere l'autorizzazione della vittima. L'uso di vulnerabilità permette di realizzare codici virali efficaci e invasivi ma, di contro, poco compatibili. Infatti, le applicazioni malevole di questo tipo riescono ad installarsi solo sulla ristretta gamma di dispositivi mobili che ancora soffrono della debolezze scelte e che, quindi, non possiedono software del tutto aggiornato. La politica di rilascio degli aggiornamenti, inoltre, rappresenta un serio problema per Google: più del 45% dei dispositivi Android nel mondo possiede una versione obsoleta del sistema operativo ed ha quindi falle di sicurezza [6]. Le radici di questo problema sono da ricercarsi nell'elevata frequenza di rilascio di nuove versioni del sistema operativo e nella politica di aggiornamenti dei produttori di dispositivi. Ogni anno, infatti, viene rilasciata almeno una nuova versione di Android, ma essa non è subito installabile su qualsiasi dispositivo. Il sistema operativo di Google, infatti, è open-source ed è usabile da qualsiasi produttore

che, però, solitamente la adatta al proprio prodotto. Questo processo di personalizzazione, oltre ad essere lento, diventa progressivamente costoso e, quindi spesso smartphone o tablet relativamente recenti non vengono più aggiornati dal produttore, rimanendo fermi ad una versione vecchia di Android. Questa frammentazione permette di sfruttare vulnerabilità già largamente corrette da aggiornamenti che, in realtà, non vengono installati sulla maggior parte dei dispositivi. Inoltre, per aumentare la compatibilità delle proprie applicazioni, gli utenti malevoli integrano codice per sfruttare diverse debolezze. Un esempio è rappresentato da "DroidDream" e "DroidKungFu", due celebri codici virali che sfruttano due diverse vulnerabilità per acquisire i privilegi di sistema in due diverse versioni di Android. La debolezza in questione, conosciuta come "Exploid" coinvolge il gestore di dispositivi per il kernel Linux. Fino alla versione 2.2 di Android, questo componente non controllava l'attendibilità della fonte dei messaggi di cambiamento di stato di una qualsiasi interfaccia fisica. In questo modo, se un'applicazione inviava un falso messaggio di connessione, il sistema operativo non controllava se esso provenisse dallo spazio kernel o utente, assegnando automaticamente i privilegi di root al programma che ha inviato il messaggio. Simulando l'inserimento di una periferica o attivando a ripetizione l'interfaccia wireless del dispositivo, quindi, era possibile ottenere facilmente i privilegi di amministratore.

Una seconda strategia di attacco è quella in cui l'utente malevolo tenta di realizzare un'applicazione che deve apparire il più possibile legittima, per convincere la vittima ad installarla e a dare inizio all'infezione. Questa tattica, quindi, non si basa sull'uso di debolezze software, ma sfrutta la relativa disinformazione degli utenti circa le tematiche di sicurezza. Le tecniche usate in questo caso sono molteplici ma si differenziano a seconda che si scelga di diffondere il malware attraverso Google Play o canali di diffusione alternativi. Questi ultimi rappresentano l'ambiente ideale per diffondere malware sotto forma di programmi piratati, ovvero versioni normalmente a pagamento diffuse, però, gratuitamente. Gli esempi di questo tipo di applicazione malevola sono numerosi e coinvolgono spesso giochi particolarmente celebri e desiderati, come "Angry Birds" o "Angry Birds Star Wars". La tecnica più usata per realizzare un malware di questo tipo è il *repackaging*: tramite strumenti reperibili in rete (tra cui il celebre "apktool") un utente malevolo può facilmente modificare e personalizzare l'applicazione originale inserendovi codice virale. Una volta che l'inclusione della componente malevola è completata, è possibile ricompilare i file ed ottenere una nuova applicazione identica all'originale, che però non possiede alcuna firma digitale, perché quella originale è stata compromessa durante il processo di decompilazione. Esistono però strumenti che permettono di firmare la nuova applicazione con un certificato "auto firmato" (noto anche come certificato *self-signed*), realizzabile da qualsiasi utente e che non necessita di alcuna validazione da parte di autorità di certificazione, cosa invece indispensabile per i certificati delle applicazioni ufficiali per Android. Infatti per la diffusione su Google Play è necessario che lo sviluppatore effettui una registrazione in cui deve

inserire i propri dati personali e, a seguito della quale, viene rilasciato un certificato da Google stessa. Se invece un utente desidera diffondere la propria applicazione attraverso canali non ufficiali, può usare un certificato auto firmato. In quest'ultimo caso, però, l'applicazione può solo distribuita fuori dallo store ufficiale, e quindi da una fonte sconosciuta: le impostazioni predefinite del sistema operativo non permettono l'installazione di questi contenuti ma, attraverso una semplice modifica, l'utente può consentirla. Completato il processo di firma, è possibile diffondere l'applicazione riconfezionata su qualsiasi canale di diffusione non ufficiale. Secondo statistiche redatte dall'azienda di sicurezza Lookout, questo tipo di tecnica coinvolge circa l'11% delle applicazioni presenti sui market alternativi [8].

4. Conclusioni

Nell'articolo si è evidenziato come il malware sia una minaccia quanto mai attuale per qualsiasi utente che possieda un dispositivo con accesso a Internet. I malintenzionati, infatti, sviluppano in continuazione nuove tattiche in grado di aggirare anche i più moderni sistemi di protezione, continuando la continua lotta tra "guardie e ladri" cominciata negli anni '90 con la diffusione di massa dei sistemi informatici. Inoltre, i dispositivi mobili, che fino a qualche anno fa erano solo in minima parte coinvolti da infezioni da parte di codici virali, stanno sempre di più diventando gli obiettivi preferiti degli utenti malevoli, grazie a caratteristiche introvabili sui PC tradizionali. Inoltre, la diffusione sempre più capillare dei dispositivi mobili ha fatto sì che sempre più utenti poco istruiti circa le tematiche di sicurezza compiano operazioni rischiose senza essere consapevoli dei pericoli. Prestando più attenzione nel compiere determinate azioni, infatti, sarebbe possibile diminuire notevolmente le possibilità di contrarre un malware. Nonostante che negli ultimi anni si siano fatti decisi passi avanti in tema di cultura della sicurezza informatica, questo campo risulta ancora poco noto. Recenti ricerche [13] evidenziano che quasi la metà degli utenti non effettua gli aggiornamenti software o addirittura non conosce l'esistenza di software di sicurezza appositi per dispositivi mobili. Risulta quindi indispensabile sensibilizzare gli utenti circa la sicurezza informatica, anche perché negli usi normali, soprattutto con smartphone e tablet, comportamenti adeguati sono più efficaci di molti software di sicurezza.

Riferimenti bibliografici

[1] Aycock J., "*Computer viruses and malware*", Springer, 2006

[2] Bashari B., Masrom M., Ibrahim S., "*Camouflage in Malware: from Encryption to Metamorphism*", International Journal of Computer Science and Network Security, Vol. 12 No. 8, Agosto 2012, pp. 74–83

[3] Computer Economics, Inc, "*2007 Malware Report: The Economic Impact of Viruses, Spyware, Adware, Botnets, and Other Malicious Code*", Giugno 2007.

[4] F-Secure Corporation, "*Top10 malware registry launchpoints*", http://www.f-secure.com/weblog/archives/00001207.html

[5] F-Secure2 Corporation, "*Mobile Threat report Q1 2013*", 2013, http://www.f-secure.com/static/doc/labs_global/Research/Mobile_Threat_Report_Q1_2013.pdf

[6] Google Android Developers, "*Platform Versions*", http://developer.android.com/about/dashboards/index.html

[7] InternetWorld Stats, "*World Internet Users and Population Stats*", http://www.internetworldstats.com/stats.htm

[8] Lookout Inc. "*Alternative App Markets: Increase Choice of Apps, but Some Have a Higher Propensity for Security Risks*", App Genome report, febbraio 2011, https://www.lookout.com/appgenome/

[9] McAfee Labs, "*McAfee threats report: Third quarter 2012*", 2012, http://www.mcafee.com/us/resources/reports/rp-quarterly-threat-q3-2012.pdf

[10] Paleari R., Martignoni L., Roglia G.F., Bruschi D., "*A fistful of red-pills: How to automatically generate procedures to detect CPU emulators*", 3rd USENIX Workshop on Offensive Technologies (WOOT), Agosto 2009

[11] SecurityNET, *Virus informatici, internet worm e malware*. Edicred, Novembre 2005

[12] Skoudis E., Zeltser L., "*Malware: Fighting Malicious Code*", Prentice Hall Ptr, 2004

[13] Symantec Corp., *"2012 NORTON CYBERCRIME REPORT",* Luglio 2012

Biografia

Marco Mezzalama, Professore Ordinario di Sistemi di Elaborazione, ha svolto la propria attività didattica e scientifica presso il Dipartimento di Automatica e Informatica del Politecnico di Torino, del quale è stato Pro Rettore Vicario e Vice Rettore. Membro dell'Accademia delle Scienze, è stato revisore di progetti scientifici nazionali e internazionali. È autore di numerosi articoli scientifici e di alcuni libri. Ha interessi di ricerca nel settore della sicurezza informatica, delle architetture di elaborazione e dell'e-learning.

Email: marco.mezzalama@polito.it

Antonio Lioy, è Professore Ordinario di Sistemi di Elaborazione presso il Dipartimento di Automatica e Informatica del Politecnico di Torino. Attualmente la sua ricerca si concentra sulla sicurezza ICT, con particolare attenzione all'identità digitale (e-ID), ai sistemi informativi *trusted* ed ai sistemi di protezione basati su metodi formali (policy e ontologie). Agisce frequentemente come valutatore e revisore di progetti per la Commissione Europea. Autore di oltre 100 pubblicazioni, è stato Coordinatore del progetto POSITIF (IST-FP6) ed attualmente lo è del progetto SECURED (IST-FP7).

Email: antonio.lioy@polito.it

Hassan Metwalley, laureato in computer engineering presso il Politecnico di Torino nel 2013, ha svolto la propria tesi studiando i codici virali per i sistemi informatici. Attualmente lavora presso Syskoplan Reply occupandosi di applicazioni per sistemi operativi Android e iOS.

Email: hassan.metwalley@gmail.com