Architettura collaborativa per la rilevazione e l'analisi di malware

Daniele Gozzi
Università degli Studi di Modena e Reggio Emilia
danielegozzi@weblab.ing.unimo.it

Mirco Marchetti
Università degli Studi di Modena e Reggio Emilia
mirco.marchetti@unimore.it

Saverio Verrascina
Università degli Studi di Modena e Reggio Emilia
saverio@weblab.ing.unimo.it

Abstract

Il continuo aumento nel numero e nella pericolosità di worm e di altro malware autoreplicante in Internet rappresenta di per sé una chiara dimostrazione dell'inefficacia delle misure di sicurezza comunemente adottate. Infatti, le contromisure necessarie per impedire la diffusione di malware vengono normalmente adottate solo dopo aver contratto un'infezione, a volte con notevole ritardo. In questo articolo proponiamo una architettura distribuita per l'analisi del malware e la diffusione delle informazioni ricavate basata sulla presenza di molteplici sensori installati in reti cooperanti. L'utilizzo di sensori distribuiti geograficamente in reti eterogenee consente di abbassare notevolmente i tempi di rilevazione del malware, permettendo quindi di catturare un esemplare del malware e di effettuarne l'analisi non appena una delle reti cooperanti rilevi un tentativo di infezione. Inoltre, le informazioni rilevati, determinate in seguito all'analisi di esemplari di malware realmente circolanti in rete, vengono inviate a tutte le reti cooperanti, anche a quelle che non sono ancora state raggiunte dal malware, favorendo la creazione di contromisure preventive. Le problematiche relative al trasferimento di informazioni sensibili tra reti eterogenee sono state risolte adottando comunicazioni cifrate e mutua autenticazione tra tutti i componenti dell'architettura. La fattibilità della soluzione proposta è infine dimostrata mediante un prototipo completo e funzionante, realizzato utilizzando software Open Source e validato sperimentalmente.

Keywords

Worm, Sistemi distribuiti, Honeypot, Difese attive

INTRODUZIONE

I worm, cioè i malware in grado di provvedere autonomamente alla propria diffusione attraverso le reti, hanno senza dubbio trovato il proprio ambiente ideale nella rete globale di Internet. È ormai superfluo riportare considerazioni riguardanti il fatto che la rete tende a crescere costantemente e che gli host tendono ad essere sempre più strettamente connessi (ad esempio con il passaggio dalle connessioni PSTN dialup alla banda larga garantita dalle soluzioni xDSL o cable). Inoltre, la crescente diffusione degli Internet worm è dovuta alla costante evoluzione nei meccanismi utilizzati per la diffusione e la replicazione, che hanno attraversato vari stadi:

1. Infezione automatizzata di servizi vulnerabili su server.
2. Infezione automatizzata di computer desktop attraverso vulnerabilità di servizi collaterali del sistema operativo.
3. Trasmissione via email, con infezione attivabile da un utente incauto.
4. Trasmissione via email, con attacco basato su vulnerabilità del MUA (non richiede l'intervento diretto di un utente).
5. Infezione di applicazioni web vulnerabili, spesso effettuata con tecniche di XSS. La ricerca di bersagli avviene tramite query dirette ai search engine [1].
6. Infezione dell'ambiente di esecuzione del codice lato client (tipicamente, javascript o flash) in applicazioni web ricche, come quelle attuate dai worm *lOrdOfthenOOse* e *EricAndrew* [2]

Indipendentemente dal vettore di infezione utilizzato, la strategia utilizzata dal malware autoreplicante per diffondersi attraverso le reti è rimasta sostanzialmente invariata. Il primo passo consiste nello sfruttare una o più vulnerabilità di uno specifico software applicativo o di un sistema operativo, in modo da garantirsi la possibilità di eseguire codice malevolo senza che l'utente ne sia a conoscenza. Per assicurare il maggiore tasso di infettività possibile, vengono normalmente bersagliate le vulnerabilità di applicazioni molto diffuse tra l'utenza domestica, solitamente meno attenta alle problematiche di sicurezza e all'aggiornamento del software vulnerabile. Una volta ottenuto il controllo di un nuovo host, il worm lo infetta eseguendo il *payload*, codice malevolo che può essere trasferito sull'host infetto contestualmente allo sfruttamento della vulnerabilità o che, più comunemente, viene scaricato da una apposita rete di host infetti, utilizzati come server. In entrambi i casi il payload del worm è largamente indipendente dalla vulnerabilità che utilizza per diffondersi (lo stesso payload può diffondersi grazie a vulnerabilità differenti e viceversa), ed è possibile isolare il payload dal codice di exploit vero e proprio. La "mutazione" del contenuto del worm è solo uno degli aspetti che secondo molti accomunano i virus informatici (in senso lato, intendendo tutte le categorie di malware) e i virus biologici. Ad esempio, è noto che la diffusione dei worm segue cicli temporali analoghi a quelli di alcuni virus

biologici: i worm possono infettare solo le macchine accese, quindi la loro diffusione segue prevalentemente cicli giornalieri e avviene da est verso ovest grazie alla differenza di fuso orario; analogamente, il virus dell'influenza segue cicli annuali e viene portato dai venti da est verso ovest.

Eventuali meccanismi di cifratura e polimorfismo inseriti nel worm possono renderne problematica la classificazione, ma questa è una operazione che la maggior parte dei software antivirus riesce ad effettuare senza difficoltà anche in presenza di tecniche di occultamento. La classificazione del malware che impiegheremo sarà condotta in due stadi proprio per prendere in considerazione anche il malware che impiega polimorfismo o cifratura del payload. Nel primo stadio viene effettuata una analisi basata su firme, usando diversi antivirus. Nel caso in cui il payload non possa essere identificato con questo metodo, si procede ad una analisi mediante sandbox: il malware viene inserito all'interno di un ambiente di esecuzione realistico e ne vengono analizzati gli effetti.

La strategia di infezione dei worm si sta dimostrando estremamente efficace. Sono state recentemente diffuse stime secondo cui Storm Worm ha ormai infettato 2 milioni di calcolatori, raggiungendo una potenza di calcolo teoricamente superiore a quella dei migliori supercomputer [3]. Altri dati relativi alla diffusione di worm e all'estensione delle relative botnet possono essere reperiti dal sito Shadowsever [4]. Volendo verificare il numero di tentativi di compromissione subiti da un normale personal computer domestico connesso ad internet, è possibile installare il software Open Source *Nepenthes* [5]. Da alcuni test da noi condotti, utilizzando un normale computer connesso a Internet mediante un collegamento ADSL flat, abbiamo rilevato 10464 tentativi di infezione in un periodo di osservazione di 178 ore, registrando quindi una media di circa un tentativo di attacco al minuto.

In un simile contesto emerge chiaramente l'inefficacia delle metodologie di difesa attualmente adottate. Una delle principali cause delle infezioni da parte di worm consiste nel ritardo con cui vengono attivate alcune semplici contromisure, sufficienti per eliminare le vulnerabilità utilizzate dal worm per diffondersi e per evitare la connessione degli host infetti ai *Command and Control* (C&C) server. Normalmente, tali contromisure vengono attuate solo dopo l'infezione, e non su base preventiva. Inoltre, reti distinte adottano queste contromisure in modo del tutto indipendente, senza propagare utili informazioni ad altre reti. Raccogliendo dati sulla diffusione di malware costantemente in tutto il mondo, diventerebbe possibile predisporre delle contromisure prima che la giornata lavorativa abbia inizio in un ampio intervallo di fusi orari, limitando fortemente la diffusione dei malware nelle reti non ancora infette.

In questo articolo proponiamo un'architettura distribuita e cooperativa per la rilevazione e l'analisi di malware (con particolare attenzione al malware autoreplicante) in grado di sfruttare i numerosi benefici derivanti dalla cooperazione tra reti eterogenee e geograficamente distribuite.

La soluzione proposta si basa sull'installazione di sensori posti all'interno di reti cooperanti e distribuite geograficamente. In particolare, i sensori utilizzati sono degli honeypot a bassa interazione, in grado di catturare una copia del payload veicolato con i tentativi di infezione rilevati. Questo approccio consente di isolare il codice malevolo, e di sottoporlo ad analisi comportamentali (ad esempio basate su sandbox) in grado di determinare le attività svolte dal malware. L'esecuzione in un ambiente realistico del malware rende l'analisi immune alle tecniche di polimorfismo e metamorfismo comunemente utilizzate.

La cooperazione tra diverse reti, congiuntamente all'utilizzo di honeypot, comporta numerosi benefici:

- **Diminuzione dei tempi di rilevazione.** Solitamente, i worm attaccano gli host connessi ad Internet generando una lista di indirizzi IP di future vittime, a cui propagare l'infezione. Al fine di massimizzare la probabilità di generare l'indirizzo IP di un host realmente esistente, una strategia comunemente utilizzata consiste nel concentrare un numero maggiore di tentativi verso indirizzi "vicini" a quelli dell'host già compromesso, con l'obiettivo di colpire host appartenenti alla medesima rete. Avendo a disposizione un numero elevato di sensori distribuiti e cooperanti, è possibile rilevare tentativi di attacco molto più velocemente rispetto ad un approccio tradizionale basato su un solo sensore, o su più sensori installati all'interno della stessa rete.
- **Analisi comportamentale del malware**. Rispetto ad altre soluzioni proposte in precedenza, in cui i sensori impiegati sono dei tradizionali Network IDS [6], l'utilizzo di honeypot consente di raccogliere esemplari del malware realmente circolanti in rete. Questi esemplari possono essere trasferiti tra gli elementi dell'architettura proposta, e sottoposti ad analisi comportamentale. In questo modo, è possibile determinare le attività che il malware svolge all'interno degli host infetti, producendo quindi una caratterizzazione dettagliata del malware resistente a tecniche (comunemente utilizzate) di polimorfismo e metamorfismo. Da questo tipo di analisi possono inoltre essere estratte utili informazioni relative

alle attività di rete del malware, quali il tipo di vulnerabilità utilizzata come vettore di infezione e gli indirizzi IP dei server C&C a cui il malware tenta di connettersi.

- **Adozione di contromisure preventive**. Lo scambio di informazioni tra le reti cooperanti può essere una misura particolarmente efficace nel limitare (o nell'impedire definitivamente) la propagazione di un particolare esemplare di malware. Distribuendo le informazioni ottenute mediante analisi comportamentale a tutte le reti cooperanti, le reti che non hanno ancora ricevuto nessun tentativo di attacco possono predisporre efficaci contromisure preventive. Conoscendo i particolari relativi al vettore di infezione utilizzato dal malware è possibile correggere preventivamente le relative vulnerabilità, rendendo gli host immuni all'infezione. Nel caso in cui vengano identificati i server C&C, è inoltre possibile predisporre delle apposite regole di firewalling per identificare e bloccare eventuali tentativi di connessione. In questo modo si limita l'efficacia di tutti i worm che fanno riferimento alla stessa botnet, in più, grazie alla cooperazione di reti diverse diventa semplice redigere blacklist di nodi C&C .
- **Approfondimento dell'analisi**. Avendo a disposizione informazioni dettagliate relative al comportamento del malware, è possibile effettuare un'analisi approfondita. In questo modo le reti cooperanti possono effettuare una ricerca particolarmente approfondita al proprio interno, in grado di far emergere infezioni già contratte che sfuggono alle normali tecniche di analisi, basate su firme (inefficaci contro malware polimorfici e metamorfici) o su logiche euristiche (in grado di rilevare malware sconosciuti, ma con una attendibilità limitata).
- **Miglioramento nell'efficienza dell'analisi**. Le metodologie di analisi comportamentale precedentemente nominate, basate su tecnologie di *sandboxing*, richiedono una elevata disponibilità di risorse. L'utilizzo di una architettura cooperativa, in cui tutti i sensori distribuiti geograficamente inviano ad una sandbox centralizzata (o ad un insieme di sandbox condiviso) *tutti e soli* gli esemplari di malware sconosciuti, consente di razionalizzare il processo di analisi, aumentandone nel contempo l'efficacia e l'efficienza.
- **Raccolta di informazioni statisticamente significative**. Avendo a disposizione un numero sufficientemente elevato di sensori distribuiti geograficamente, è possibile raccogliere statistiche utili sia a fini pratici che di ricerca.

La fattibilità della soluzione proposta è dimostrata mediante la realizzazione di un prototipo funzionante, basato sull'utilizzo di software Open Source. Il prototipo è stato validato sperimentalmente in condizioni controllate di laboratorio, sottoponendone ad analisi i singoli componenti utilizzando del malware noto. Una ulteriore conferma della validità della soluzione proposta è poi stata ottenuta mediante l'installazione di sensori *in the wild*, sottoponendo così l'architettura proposta a scenari di lavoro realistici e a malware precedentemente sconosciuti.

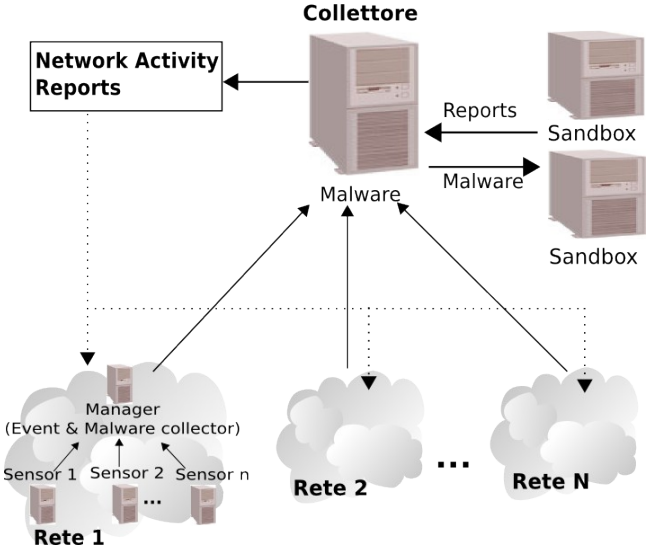

Figura 1: Architettura cooperativa per l'analisi del malware

ARCHITETTURA COOPERATIVA PER L'ANALISI DEL MALWARE

Una descrizione schematica dell'architettura proposta in questo articolo è rappresentata in Figura 1.

Sensori

Il livello più basso dell'architettura è costituito dai sensori. L'architettura proposta consente l'utilizzo di sensori eterogenei, tuttavia è stata ottimizzata per l'utilizzo di honeypot a bassa interazione.

La scelta di utilizzare honeypot, e non un generico NIDS, è motivata dalle particolari caratteristiche dell'analisi che questo tipo di sensore è in grado di garantire.

A differenza dei tipici sistemi di analisi del traffico e di intrusion detection, un honeypot non analizza attivamente tutto il traffico che riesce a ricevere, ma rimane semplicemente in attesa di essere contattato da potenziali attaccanti. Il principio alla base degli honeypot consiste nell'emulare il comportamento di una possibile vittima,

attirando quindi l'attenzione di eventuali attaccanti in cerca di macchine da compromettere.

Una diretta conseguenza di questo approccio consiste nella notevole riduzione (virtualmente, nell'eliminazione completa) dei falsi positivi nei confronti delle altre tipologie di analisi del traffico. Essendo un honeypot una macchina senza una reale utilità produttiva e senza nessun tipo di servizio pubblicizzato, nessun utente legittimo ha la necessità di interagire con tale macchina. Per cui tutte le interazioni registrate tra un qualunque host ed un honeypot non rispondono a delle reali necessità, e sono quindi comportamenti sospetti.

In particolare, nell'architettura descritta proponiamo l'utilizzo di un honeypot a bassa interazione, cioè un honeypot in cui i servizi vulnerabili sono emulati da un apposito software, in contrapposizione ad un honeypot ad alta interazione, in cui i servizi vulnerabili vengono effettivamente installati all'interno della macchina. Mentre gli honeypot ad alta interazione consentono un'analisi molto affidabile delle attività compiute da un eventuale attaccante, l'installazione di questo tipo di honeypot richiede dei costi decisamente superiori rispetto a quelli di una macchina in grado di emulare solo le caratteristiche salienti (in questo contesto, le vulnerabilità) del software. Inoltre un honeypot a bassa interazione consente di emulare le vulnerabilità di numerosi servizi eterogenei, che utilizzando l'approccio ad alta interazione necessiterebbero dell'installazione di altrettante honeypot.

Un'altra caratteristica importante degli honeypot, alla base di alcune delle funzionalità dell'architettura proposta, consiste nella capacità di catturare una copia del payload con cui un host vittima avrebbe dovuto essere infettato in seguito ad un attacco ricevuto. A differenza di un normale network intrusion detection system, che si limita a rilevare un attacco, dando solo una descrizione della vulnerabilità sfruttata dall'attacco ricevuto, un honeypot può emulare il comportamento del software vulnerabile, ottenendo quindi una copia del payload allo stesso modo in cui il medesimo payload sarebbe stato scaricato da una macchina vittima in seguito ad una reale infezione.

La capacità di catturare degli esemplari reali di malware è uno dei principali punti di forza dell'architettura proposta, in quanto consente di analizzare efficacemente il *comportamento* del malware.

Manager Locale

Una rete cooperante è una rete in cui è stato installato almeno un sensore, in grado quindi di fornire informazioni relative agli attacchi rilevati al suo interno al resto dell'architettura. A seconda della dimensione e della topologia della rete, un singolo sensore può non essere sufficiente per garantire una adeguata copertura della rete stessa. In questo caso risulta utile installare più di un singolo sensore, ad esempio installandone uno in ogni sottorete.

Qualora in una delle reti cooperanti vengano installati più sensori, le comunicazioni con gli altri componenti dell'architettura vengono mediate da un elemento intermedio, il *manager locale*. La scelta di utilizzare un manager locale risponde a varie necessità, in quanto consente di semplificare, ed al contempo di ottimizzare, lo scambio di informazioni tra le reti cooperanti e l'architettura proposta.

In seguito alla rilevazione ed alla cattura di un esemplare di malware, ogni sensore ne invia una copia al manager locale. Il manager locale ha in gestione una collezione di tutti i malware precedentemente catturati all'interno della rete dai sensori, pertanto è in grado di verificare se gli esemplari di malware appena ricevuti sono già stati rilevati in precedenza o se sono stati catturati per la prima volta. Qualora il malware risulti sconosciuto, viene inviato al livello superiore dell'architettura per sottoporlo ad ulteriori analisi. Questo schema di comunicazione consente di propagare verso l'esterno esclusivamente le informazioni strettamente indispensabili, quelle relative al malware non ancora analizzato, riducendo quindi il traffico e migliorando l'efficienza dell'architettura nel suo complesso.

Inoltre, il mantenimento di un singolo punto di contatto tra una generica rete cooperante semplifica la gestione delle varie reti, e consente di prescindere dalle loro caratteristiche interne (dimensione e topologia).

Qualora le caratteristiche di una rete cooperante siano tali da richiedere la presenza di un solo sensore, è inoltre possibile accorpare i ruoli di sensore e di manager locale su di un'unica macchina fisica, mantenendo quindi un alto livello di flessibilità e riducendo al minimo i costi di installazione.

Collettore centrale

Il collettore centrale è l'elemento posto al vertice dell'architettura proposta. Ogni rete cooperante deve, come operazione preliminare, registrare il proprio manager locale presso il collettore, a cui il manager locale invia tutti gli esemplari di malware sconosciuti. In questo modo, tutte le reti cooperanti contribuiscono ad allargare la collezione di malware gestita dal collettore, che arriva a comprendere tutti gli esemplari di malware rilevati all'interno delle reti.

Nel caso il malware appena ricevuto dal collettore non sia già stato ricevuto ed analizzato, tale malware viene aggiunto alla collezione ed inviato a strumenti esterni per effettuarne una analisi approfondita. I risultati di queste analisi vengono poi trasformati in rapporti contenenti informazioni utili sul malware catturato. Tali rapporti sono inviati dal collettore a tutte le reti cooperanti, anche a

quelle che non sono ancora state coinvolte da quel tipo di infezione.

Sandbox

Sebbene l'architettura proposta sia in grado di utilizzare tecniche di analisi eterogenee, una sandbox risulta lo strumento più adatto al fine di determinare con certezza il comportamento di un malware. Generalmente, una sandbox consiste in un ambiente di esecuzione emulato, all'interno del quale vengono caricati ed eseguiti i file che si vogliono analizzare. Essendo l'ambiente di esecuzione perfettamente conosciuto e controllato, è possibile isolare con certezza tutte le attività eseguite dal malware, sia al livello delle modifiche apportate al filesystem (file alterati, chiavi di registro modificate) che all'ambiente di esecuzione (processi creati o terminati). Analizzando il traffico di rete generato dalla macchina virtuale infetta è inoltre possibile isolare le attività di rete generate dal malware. Tipicamente, in questo modo è possibile identificare quali sono i servizi vulnerabili bersagliati dal malware per infettare altri host. Inoltre, nel caso il malware installi un bot all'interno degli host infetti, è possibile identificare quali sono le macchine verso cui il bot tenta di connettersi per ricevere ordini, identificando i relativi C&C server.

I risultati delle analisi effettuate dalle sandbox vengono poi inviati automaticamente al collettore.

Activity report

Potendo utilizzare diversi servizi di analisi del malware, i relativi risultati saranno forniti in formati differenti tra loro, spesso non strutturati e, comunque, non direttamente confrontabili.

Al fine di semplificare l'utilizzo di queste informazioni da parte delle reti cooperanti, i risultati delle analisi vengono elaborati dal collettore, che genera degli activity report omogenei e in cui vengono evidenziate le informazioni di maggiore utilità. In particolare, viene dato notevole risalto alle attività di rete, identificando chiaramente gli end-point delle eventuali connessioni che possono essere messe in relazione con il malware analizzato.

I report così generati vengono poi inviati a tutte le reti cooperanti, anche a quelle che non sono ancora state raggiunte dal malware. In questo modo a tutte le reti cooperanti vengono messe a disposizione le informazioni necessarie per attuare delle contromisure preventive volte ad immunizzare la rete da quel particolare esemplare di malware, eliminando le vulnerabilità che sono state identificate o bloccando il traffico generato dal malware mediante apposite regole di firewall.

Opportune regole di filtraggio del traffico possono inoltre essere utilizzate per bloccare l'accesso ai server C&C, inibendo le attività di eventuali bot già presenti all'interno della rete ed impedendo il corretto funzionamento anche ad eventuali malware ancora del tutto sconosciuti che facciano comunque riferimento alla stessa botnet.

Sicurezza delle comunicazioni

Raccogliendo dati da un grande numero di fonti, geograficamente distribuite e facenti capo a reti eterogenee, è essenziale impedire che una delle fonti possa (volontariamente o meno) inquinare l'insieme dei dati raccolti. Per prevenire questa eventualità, è necessario instaurare una forma di tracciabilità delle fonti di informazioni. A questo scopo, si è deciso di utilizzare la proprietà di non ripudiabilità delle transazioni a chiave pubblica.

Sia il collettore centrale che tutti i manager locali installati in ciascuna delle reti cooperanti sono dotati di una coppia di chiavi asimmetriche, che vengono utilizzate per autenticare (mediante firma digitale) tutte le informazioni scambiate tra i diversi componenti dell'architettura. In questo modo è possibile garantire la completa tracciabilità delle comunicazioni, assieme alla certezza che solo i manager locali correttamente registrati presso il collettore centrale possono inviare esemplari di malware per l'analisi.

Tutte le informazioni scambiate tra i componenti dell'architettura sono cifrate, al fine di garantirne la riservatezza.

Occorre inoltre considerare il fatto che lo schema di comunicazione proposto consente di raggiungere i benefici della cooperazione senza richiedere lo scambio diretto di informazioni tra tutte le reti cooperanti, in quanto l'unico flusso di dati (e quindi l'unico rapporto di fiducia) necessario per il funzionamento dell'architettura è rappresentato dalle comunicazioni tra un manager locale e il collettore. Questa proprietà è particolarmente importante, in quanto non rappresenta un limite alla scalabilità dell'architettura. Infatti, alle singole reti cooperanti non è richiesta la conoscenza dei manager locali delle altre reti coinvolte nell'architettura.

REALIZZAZIONE DELL'ARCHITETTURA PROPOSTA

Al fine di verificare la fattibilità della soluzione proposta, abbiamo realizzato un prototipo sperimentale, ma completamente funzionante, utilizzando principalmente componenti software Open Source liberamente disponibili in rete. Alcune delle funzionalità necessarie all'architettura proposta, ma non fornite da tali software, sono state ottenute mediante la realizzazione di appositi moduli ad-hoc, integrati con il resto dell'architettura.

Raccolta del malware

La raccolta del malware viene condotta da delle installazioni della honeypot Nepenthes [7]. Tale software

consiste di un demone dalla struttura modulare che ha 5 funzioni principali:

- *binding* di socket per l'attesa di connessioni in ingresso
- identificazione della vulnerabilità bersagliata
- analisi del codice di exploit per l'estrazione delle informazioni necessarie al download del payload
- recupero del payload
- aggiornamento del registro delle attività, eventualmente su server remoti

Per ciascuna di queste funzioni esistono appositi moduli, che nei sorgenti del software hanno nomi prefissati da una descrizione del proprio tipo: `dnsresolve-`, `download-`, `module-`, `log-`, `shellcode-`, `shellemu-`, `sqlhandler-`, `submit-`, `vuln-`. Ad esempio, i moduli aventi nomi che iniziano per "vuln" contengono la logica che occorre per simulare le vulnerabilità e rispondere ai tentativi di attacco in modo da potere raccogliere le informazioni necessarie a ottenere il payload. Esaminando il funzionamento di Nepenthes, appare subito chiaro che l'emulazione di vulnerabilità diverse sulla stessa porta TCP o UDP non è triviale: occorre interpretare il tentativo di exploit ricevuto per simulare una risposta corretta. Nepenthes risolve questo problema con i Dialog, classi a cui viene delegata l'interpretazione delle comunicazioni relative all'attacco. Ogni Dialog che è stato registrato per una certa porta TCP/UDP esamina i primi pacchetti giunti alla honeypot e determina se si tratta di una comunicazione che è in grado di sostenere. Nepenthes scarta i dialog che hanno riferito di non potere gestire l'attacco e delega la comunicazione al Dialog corretto, grazie al quale vengono raccolti i dati necessari per compiere il download del codice virale.

Il payload viene reperito con diversi metodi, a seconda del tipo di attacco; ad esempio vi sono alcuni tipi di shellcode che aprono shell per l'immissione di comandi da indirizzi remoti, altre volte viene avviato un download via TFTP, FTP, HTTP o trasferendo uno stream di byte direttamente via TCP.

Nepenthes calcola un hash SHA512 o MD5 del malware, ed archivia una copia del suo codice. I file binari così raccolti possono essere caricati su server remoti con diversi metodi o essere sottoposti (anche tramite il protocollo G.O.T.E.K.) all'attenzione di enti che ricercano nuovi malware, come mwcollect.org o Norman [8]. Il calcolo dell'hash semplifica la procedura di archiviazione delle copie del malware, perché permette di scartare i duplicati dello stesso codice binario, dato che l'hash calcolato sulle sequenze identiche di byte risulta uguale.

Nepenthes fornisce anche un modulo per il log che consente di registrare presso un manager Prelude [9] gli eventi rilevati.

Infrastruttura di comunicazione

Uno dei componenti fondamentali dell'architettura proposta (ed in generale, di tutti i sistemi distribuiti) è rappresentato dall'infrastruttura utilizzata per la comunicazione tra tutti i suoi elementi. In questo particolare contesto, i requisiti fondamentali di tale infrastruttura si possono riassumere nel seguente elenco:

- trasferimento degli allarmi generati e dei malware catturati dai sensori locali al manager locale (nel caso siano presenti più sensori in una stessa rete);
- trasferimento degli esemplari di malware sconosciuti dai manager locali al collettore centrale;
- autenticazione di tutti i messaggi scambiati tra gli elementi dell'architettura;
- riservatezza delle comunicazioni mediante l'utilizzo di primitive crittografiche.

Oltre a questi requisiti minimi, necessari per il corretto funzionamento dell'architettura, è importante che l'infrastruttura di comunicazione utilizzata preveda lo scambio di messaggi seguendo un formato standard, in modo da non porre limiti alla tipologia di sensori utilizzati e di fornire la maggiore interoperabilità possibile con altre tecnologie di intrusion detection.

Tutte queste considerazioni hanno portato alla scelta di Prelude [9] come software ideale per implementare le funzionalità richieste.

Prelude: un IDS ibrido

Prelude è un software Open Source, liberamente scaricabile, nato per facilitare la creazione di IDS ibridi, in grado cioè di fondere i benefici di diversi tipologie di analisi, tra loro complementari. Ad esempio, è possibile utilizzare Prelude per raccogliere ed aggregare allarmi generati da sistemi di intrusion detection eterogenei, quali Network IDS e Host IDS. L'eterogeneità delle fonti supportate da prelude è gestita mediante lo scambio di allarmi nel formato standard *IDMEF* (Intrusion Detection Message Exchange Format) [10] supportato nativamente da Nepenthes.

Un componente essenziale di Prelude, e di cui si è fatto ampiamente uso all'interno dello scenario applicativo descritto, è la libreria di comunicazione Libprelude. Tale libreria fornisce tutte le primitive necessarie per il trasferimento di messaggi nel formato IDMEF tra i sensori utilizzati ed i manager locali. Un'altra funzionalità molto utile messa a disposizione dalla libreria *Libprelude* consiste nella possibilità di configurare i manager locali in modalità relaying. In questo modo, tutti gli alert ricevuti dal relaying manager possono essere inoltrati ad un altro manager, consentendo quindi la propagazione delle informazioni. Nella nostra implementazione, tutti i manager locali

eseguono il relay dei messaggi IDMEF ricevuti verso il collettore centrale, che è quindi informato della diffusione del malware all'interno delle reti cooperanti.

Sfruttando Libprelude come infrastruttura di comunicazione, sia al livello delle reti locali che tra manager locali e collettore, è possibile utilizzare le primitive di autenticazione e crittografia (realizzate mediante chiavi asimmetriche) in grado di garantire tracciabilità e riservatezza delle comunicazioni, soddisfando quindi tutti i requisiti di sicurezza delle comunicazioni.

Trasferimento del malware catturato

Un requisito fondamentale per l'architettura proposta, ma non supportato nativamente dall'IDS ibrido Prelude, consiste nella capacità di inoltrare al livello superiore (da sensore a manager locale e da manager locale a collettore) tutti e soli gli esemplari di payload sconosciuti. La sola propagazione dei messaggi IDMEF, pur essendo necessaria, non è infatti sufficiente per consentire la raccolta dei file infetti e la relativa analisi da parte del collettore.

Questo requisito è stato soddisfatto progettando e realizzando il software *MWSurfer*, implementato con una serie di script, in grado di confrontare il malware raccolto da Nepenthes con la collezione di file ostili già rilevati in precedenza e, solo qualora il malware raccolto risulti non già presente all'interno della collezione, inviarlo al livello superiore dell'architettura.

Il confronto tra i vari esemplari di malware avviene confrontando una signature, ottenuta eseguendo l'hash MD5 del file catturato. In questo modo è possibile confrontare in modo efficiente un nuovo file malevolo con una collezione, anche estesa, di malware già conosciuti, senza richiedere un confronto tra i singoli file.

Strumenti di analisi del malware

Uno dei punti di forza dell'architettura proposta, consiste nella possibilità di utilizzare strumenti di analisi eterogenei, ottenendo il meglio dai diversi approcci che tali strumenti possono implementare. In particolare, è possibile utilizzare sia strumenti di analisi interni (quali antivirus locali) che strumenti di analisi esterni, come motori antivirus e sandbox pubblicamente accessibili tramite Internet.

Per dimostrare la versatilità della soluzione proposta, il prototipo da noi realizzato si avvale di tre diversi strumenti di analisi, tutti esterni. Tali strumenti sono:

- Virustotal [11]
- Norman Sandbox [8]
- CW Sandbox [12]

L'invio dei malware raccolti, e la ricezione delle relative risposte, è stata gestita da un software appositamente realizzato, chiamato *MWSender*. Tale software può essere corredato da un numero arbitrario di plugin, ognuno dei quali è responsabile dell'invio del malware ad uno specifico servizio di analisi.

Virustotal

Virustotal è un servizio di analisi di file potenzialmente infetti gratuito e liberamente utilizzabile. Il pregio principale di questo servizio consiste nell'utilizzo di un elevato numero (32, al momento della scrittura di questo articolo) di motori antivirus, prodotti da diversi vendor e costantemente aggiornati.

Nonostante questo tipo di analisi non permetta di trarre conclusioni sul comportamento del malware analizzato, rappresenta comunque un ottimo strumento per ottenere numerose informazioni sul file, come ad esempio la diversa denominazione che i vari vendor utilizzano per classificare lo stesso tipo di malware. Inoltre, l'utilizzo di un elevato numero di motori antivirus consente di eliminare la maggior parte dei falsi negativi (ovvero dei file infetti che non sono correttamente riconosciuti come tali) rispetto ad una analisi locale condotta con un singolo antivirus. Occorre infatti evidenziare che, secondo le statistiche fornite da Virustotal, su un totale di 13800 file infetti, solo 2 sono stati correttamente rilevati da tutti i motori utilizzati, mentre in 13798 casi almeno uno dei motori antivirus utilizzati ha generato dei falsi negativi, pur essendo perfettamente aggiornato[1].

Al fine di automatizzare l'utilizzo di Virustotal, abbiamo realizzato un apposito plugin di MWSender, chiamato *Mail_to_virustotal*, in grado di inviare un malware a Virustotal sfruttando la relativa interfaccia di consegna tramite e-mail. Tale software è scritto in C++, e utilizza la libreria jwSMTP [12].

Norman Sandbox

La sandbox sviluppata dai laboratori Norman consente di ottenere una analisi comportamentale di malware sconosciuti monitorandone l'esecuzione in un ambiente controllato. In particolare, l'ambiente emulato utilizzato dalla Norman Sandbox consente di rilevare le eventuali attività di rete, in quanto viene simulata anche la presenza di una rete locale a cui sono connesse le macchine virtuali.

Per automatizzare l'utilizzo di questa sandbox, abbiamo realizzato un apposito modulo per MWSender, chiamato *Send_to_norman*. Tale modulo si occupa della generazione di una richiesta HTTP valida, contenente il malware da

[1] Questa statistica si riferisce a tutti i file infetti analizzati in un periodo di 24 ore, tra le ore 11 di giovedì 27 settembre e le ore 11 di venerdì 28 settembre.

analizzare, inviata direttamente verso l'interfaccia web della Norman Sandbox.

CW Sandbox

Anche in questo caso il comportamento del malware viene studiato osservandone le attività all'interno di un ambiente emulato.

CWSandbox consente tuttavia una analisi più approfondita delle attività di rete del malware, in quanto, a differenza di quanto accade nella Norman Sandbox, l'ambiente di rete utilizzato da CWSandbox non è isolato. Ad esempio, se un particolare malware necessita di scaricare un file eseguibile dalla rete (ad esempio un altro tipo di payload) tale operazione fallisce all'interno della Norman Sandbox, in quanto la rete emulata non è connessa ad Internet. Norman Sandbox è comunque in grado di rilevare il tentativo di connessione ad un particolare host. L'ambiente di rete emulato da CWSandbox, invece, consente al malware di reperire ulteriori componenti da Internet, i quali vengono poi automaticamente sottoposti ad analisi.

Un'altra caratteristica utile di CWSandbox è rappresentata dalla possibilità di ottenere immediatamente i risultati delle analisi effettuate su un particolare esemplare di malware, qualora questo sia già stato analizzato in precedenza. Infatti, i risultati di tutte le analisi vengono memorizzati come pagina web statica, il cui URL è facilmente determinabile in base all'hash MD5 del file analizzato. Pertanto, se il malware è già conosciuto dalla CWSandbox, il reperimento delle relative informazioni è pressoché immediato.

Anche in questo caso, è possibile sottoporre dei file inviandoli alla sandbox all'interno di una opportuna richiesta HTTP, operazione effettuata dal modulo per MWSender chiamato *Send_to_cwsandbox*.

Generazione e invio dei report

I risultati delle analisi generate dai diversi strumenti utilizzati sono eterogenei, difficilmente confrontabili, e in forma non strutturata (tipicamente dei semplici file di testo). Prima di inviare tali informazioni alle reti cooperanti occorre quindi elaborare i risultati ottenuti, al fine di generare una risposta in un formato standard, in cui siano facilmente identificabili tutte le informazioni richieste. Ad esempio, tutte le informazioni relative alle eventuali attività di rete del malware vengono identificate ed evidenziate, in modo da semplificare l'implementazione di eventuali regole di firewall agli amministratori delle reti cooperanti.

RISULTATI SPERIMENTALI

Il prototipo descritto è stato validato sperimentalmente in ambiente di laboratorio, mediante l'utilizzo di tre macchine come illustrato in Figura 2.

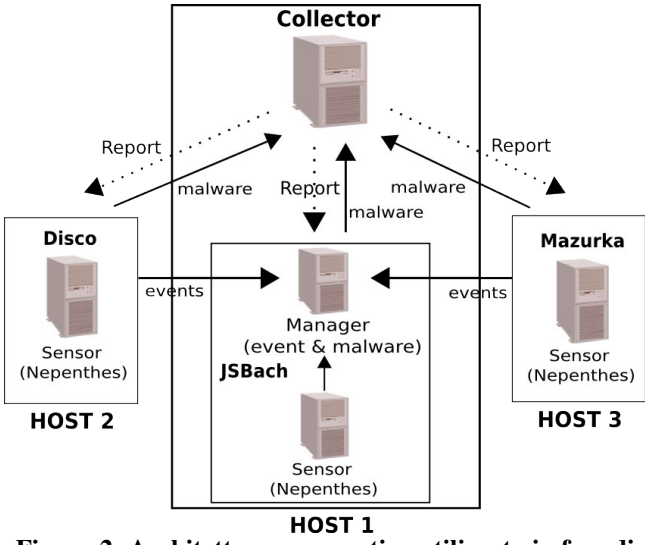

Figura 2: Architettura cooperativa utilizzata in fase di sperimentazione

Per semplicità, l'host *JSBach* svolge contemporaneamente le funzioni di sensore, manager locale e collettore, mentre i rimanenti due host fungono solamente da sensore; in questo modo è stato possibile simulare:

1. una singola rete, con tre sottoreti distinte, in modo da verificare l'invio degli eventi e del malware al manager locale presente su *JSBach*;
2. tre reti distinte dal punto di vista del collettore, in modo da verificare la corretta ricezione degli Activity Report.

La validazione dei singoli componenti è avvenuta mediante il riconoscimento di malware noti inviati ai sensori.

I software *Nepenthes* e *Prelude* gestiscono la raccolta degli esemplari di codici binari dei malware, utilizzando i quali è stato possibile verificare il corretto comportamento del collettore, e l'implementazione della struttura gerarchica sensore-manager desiderata.

In particolare, in occasione della ricezione di malware sconosciuti, il collettore ha provveduto a sottoporre ad analisi presso apposite sandbox remote (*Norman Sandbox* [8]) ogni payload e in ogni caso ha prodotto dei report contenenti tutte le informazioni raccolte.

Questi report sono stati inviati a reti cooperanti appositamente simulate, verificando la rapidità nel fornire informazioni agli amministratori di rete riguardo a eventuali malware circolanti in rete e alle loro caratteristiche peculiari, utili per creare una prima linea di difesa.

Per verificare l'efficacia del sistema in una situazione realistica si è proceduto a una seconda fase di esperimenti, nei quali alcuni sensori sono stati installati in corrispondenza di collegamenti ADSL domestici.

L'esperimento ha provocato la generazione da parte del manager locale di 3866 messaggi di allerta, la cattura da parte dei sensori di 52 binari, di cui 7 sconosciuti al collettore (già in possesso di una estesa raccolta). Questi 7 sono stati scaricati dal collettore centrale, che ha provveduto al loro invio a sistemi di scanning e a sandbox on line, ottenendo dettagliati rapporti sulla pericolosità. Grazie all'utilizzo di sandbox, questi report presentano sempre informazioni, anche a fronte di malware mai incontrato prima e quindi non rilevabile tramite firme dagli antivirus. È molto importante sottolineare questo vantaggio, che consente di approntare contromisure immediate come il filtraggio delle connessioni verso gli host C&C delle botnet.

Di seguito vengono presentati alcuni esempi di rapporti prodotti dalla sandbox a cui sono stati inoltrati i campioni di malware raccolti da Nepenthes.

```
0995104827bee951abc4fcc93cdf85ee :
INFECTED with W32/Malware
(Signature: W32/Malware.LNH)
    * Connects to "j4m4lz.B3D3RPIERO.INFO"
      on port 6137 (TCP).
    * Connects to IRC Server.
    * Possible backdoor functionality
      [Authenticate] port 113.
Network Activity:
  Opened listening TCP connection on port: 113
    * C&C Server: 69.64.36.188:6137
    * Server Password:
```

In questo caso il malware raccolto si collega ad un server C&C e in più apre una backdoor sulla porta 113 TCP.

```
13ff667bebcc58253faba2313dce7b89 :
INFECTED with W32/Kut.gen1
(Signature: W32/Poebot.ADT)
  * C&C Server: 140.116.199.57:8998
Network activity
  * Server Password: PING
```

In questo caso è stato possibile anche intercettare la password usata dal malware per collegarsi al proprio server C&C (questi server utilizzano una logica opposta alle honeypot: le connessioni casuali devono essere impedite per minimizzare il rischio di esporre la propria attività, quindi l'accesso è spesso protetto da password).

```
03fb1ecf2cbcfb74ab5c29dcd247e132 :
INFECTED with W32/Endom.A (Signature:
Allaple.gen1)
    * Sends data stream (76 bytes) to remote
        address "124.86.6.4",
      port 139.
    * Connects to "124.86.6.4" on port 445 (TCP).
    * Sends data stream (76 bytes) to remote
        address "124.86.8.6",
      port 139.
    * Connects to "124.86.8.6" on port 445 (TCP).
      * Sends data stream (76 bytes) to remote
        address "124.86.10.8", port 139.
    * Connects to "124.86.10.8" on port 445 (TCP).
    * Connects to "124.86.6.4" on port 9988 (TCP).
       * Sends data stream (255 bytes) to remote
         address "124.86.6.4", port 9988.
```

Questo malware utilizza una rete di server C&C complessa, probabilmente multi-livello, che serve ad aggirare le contromisure volte a interrompere le comunicazioni tra gli host infetti e i server di controllo. I tentativi di connessione registrati dalla sandbox infatti riguardano tre host distinti.

CONCLUSIONI

In questo articolo è stato descritto un approccio alla raccolta automatizzata di malware che ha lo scopo di trarre beneficio dalla cooperazione tra honeypot posizionate in reti distribuite geograficamente, ottenendo di migliorare i tempi di risposta nella messa in atto di contromisure nei confronti di nuove infezioni. Una delle prime applicazioni che un simile sistema può trovare è la minimizzazione degli effetti collaterali di negazione del servizio dovuti all'esplosione di epidemie di worm, prevenendo situazioni come quella causata dal worm Sasser nel 2004.

L'analisi operata sul malware raccolto permette di aggirare vari meccanismi di occultamento comunemente usati dai virus writer, contemporaneamente evitando analisi ripetute su campioni identici. La mutua autenticazione che avviene tra i nodi del sistema è la migliore garanzia possibile nei confronti dell'attendibilità dei dati raccolti e dell'accuratezza dei risultati delle analisi, dato che permette di rintracciare l'origine di ogni segnalazione di malware e lascia aperta la possibilità di escludere dalla rete di honeypot cooperanti i singoli nodi sulla base di comportamenti errati o malevoli.

L'assenza di meccanismi automatici di reazione ai nuovi malware impedisce che il sistema possa essere usato contro se stesso, ad esempio immettendo malware preparato in modo tale da impedire le comunicazioni tra i nodi honeypot e i nodi collettori (eventualità possibile se esistesse una blacklist aggiornata automaticamente contenente gli indirizzi dei server C&C rilevati nei payload degli attacchi).

Al fine di dimostrare la fattibilità dell'approccio da noi proposto, e di consentirne una validazione sperimentale, abbiamo realizzato un prototipo (minimale ma perfettamente funzionante) utilizzando software Open Source. I software utilizzati nel setup del prototipo precedentemente descritto sono altamente modulari e facilmente integrabili grazie a standard dalle specifiche pubbliche.

Bibliografia

1: http://www.securityfocus.com/bid/10701

2: http://en.wikipedia.org/wiki/Timeline_of_notable_computer_viruses_and_worms#2006

3: Sharon Gaudin , Storm Worm botnet more powerful than top supercomputers, Information Week, 2007

4: http://www.shadowserver.org

5: http://nepenthes.mwcollect.org/

6: When-Yi Hsin, Shian-Shiong Tseng, Shun-Chieh Lin, A study of alert based collaborative defense, Proceedings of the 8th International Simposium on Parallel Architectures, Algorithms and Networks (ISPAN05), 2005

7: Qinghua Zhang, Douglas S. Reeves, Peng Ning, S. Purushotaman Iyer, Analyzing Network Traffic to Detect Self-Decrypting Exploit Code, Proceedings of the 2007 ACM Symposium on InformAtion, Computer and Communications Security (ASIACCS'07), 2007

8: http://sandbox.norman.com

9: http://www.prelude-ids.org

10: http://www.ietf.org/rfc/rfc4765.txt

11: http://www.virustotal.com/

12: http://www.cwsandbox.org/

13: http://johnwiggins.net/jwsmtp/