Protocollo xyz per Intrusion Prevention Distribuita

Giorgio Ravera
Gianluca Stringhini

A.A 2007/2008

Indice

Ringraziamenti

Grazie a tutti:D

Obiettivi

Laurearsi con il massimo voto possibile, nel minor tempo possibile, scrivendo una bella tesi da nerd:D

Introduzione

Internet è nata in una quasi totale assenza di sicurezza.
D'altra parte, una rete utilizzata da appassionati e ricercatori (sebbene originariamente nata con finalità militari), e animata dal desiderio di condivisione della conoscenza, aveva come naturale implementazione un'apertura totale delle porte, piuttosto che una loro parziale chiusura.
Sebbene la cultura hacker fosse, negli anni '70 e '80, già consolidata, il problema della sicurezza non era percepito come tale. Disattenzione, questa, in parte giustificata dal fatto che le poche intrusioni indebite terminavano nella maggior parte dei casi senza conseguenze, con gli attaccanti che riuscivano a ottenere un accesso alle macchine remote per poterle studiare o per venire in possesso di informazioni accademiche o, qualora proprietà privata di qualcuno, difficilmente vendibili.
Ben diversa divenne la situazione quando a popolare la rete arrivarono non solo milioni di persone, ma anche gli interessi politici ed economici del mondo intero. Una simile fonte di informazioni costituiva anche una sorgente di guadagni indebiti che non potè passare inosservata. Così, alla schiera di persone che entrava nei sistemi per divertimento o con finalità quasi innocue, accresciuta dal proliferare di programmi atti allo scopo pubblicati sul neonato World Wide Web, si aggiunsero coloro che vedevano in questa pratica una possibile fonte di guadagno. Attacchi già noti, ma dall'impatto fino ad allora trascurabile, divennero all'ordine del giorno: l'esecuzione di codice arbitrario, il bruteforce, il Denial of Service...
La virosità dei virus informatici, un tempo limitata dal fatto che l'unico veicolo di propagazione fosse costituito dai Floppy Disk scambiati tra amici, aumentò a dismisura, al punto che sempre più computer vennero infettati da programmi maligni in grado di trasformarli in zombie, in macchine, cioè, che al momento opportuno avrebbero obbedito agli ordini di un altro, e non del loro proprietario, per portare a termine un attacco.
Il mondo informatico, prontamente, mise a punto diversi dispositivi per combattere queste minacce.
Furono messi in commercio, o distribuiti gratuitamente, software antivirus per debellare le infezioni informatiche, e firewall da installare sugli host per bloccare le connessioni non autorizzate. Di queste due categorie di programmi, tuttavia, non ci occuperemo, perchè questo testo si occupa di sicurezza delle reti, intese come tutto ciò che sta tra due endpoint connessi tra loro, e non degli endpoint stessi.
Per quel che riguarda le reti, due elementi fondamentali furono sviluppati a guardia del

traffico.
Il primo elemento è il firewall. Collocato, in genere, sul limitare di una rete, a dividere cioè una intranet di un'azienda o di un ente dal resto di Internet, esso si occupa di decidere quali pacchetti sono autorizzati a passare e quali, invece, devono venire bloccati. In principio i filtri su cui si basavano erano abbastanza rudimentali, e consentivano in sostanza di controllare soltanto gli header del pacchetto, permettendo un aggiramento piuttosto semplice. Successivamente furono implementati i firewall cosiddetti stateful, in grado, cioè, di analizzare l'intero pacchetto e di riconoscere i diversi tipi di traffico.
Il secondo elemento è l'Intrusion Detection System, o IDS. Collocato all'interno di una Intranet, esso analizza tutto il traffico di rete, in genere replicato da uno switch su una porta prestabilita. La sua funzione è del tutto passiva. Esso, infatti, controlla il traffico, in genere sulla base di regole, e segnala quali pacchetti, pur essendo passati indenni attraverso il firewall, potrebbero costituire un attacco. E' un utile strumento, dunque, offerto all'amministratore di sistema per controllare la sicurezza della propria rete.
L'evoluzione dell'IDS è l'Intrusion Prevention System, o IPS. Esso, al contrario del suo parente, ha funzione attiva. Viene collocato, cioè, in maniera trasparente nella rete, come bridge su un cavo attraverso il quale passi tutto il traffico. Sulla base di regole analoghe a quelle dell'IDS, blocca i pacchetti che reputa dannosi.
Questo è l'approccio usato per difendere le reti al giorno d'oggi. Un approccio che ha molti limiti e molti difetti, che andremo ad enunciare brevemente. Innanzi tutto, si delega a poche macchine, tipicamente 2 o 3, la difesa di una rete di decine o centinaia di computer. Una conseguenza di questo è che le macchine incaricate di fare da IDS o da firewall (specialmente nel caso questo sia stateful) debbano essere più potenti dei client che devono difendere, per poter tener testa al traffico generato da e verso questi. Per questo motivo, le macchine usate per questi scopi sono molto costose.
Oltre a questo, si ha la necessità di far passare il traffico attraverso pochi percorsi predeterminati, affinchè questo sia analizzato (a meno di disporre di sistemi di backup, scelta molto costosa). Questo comporta lo svantaggio che, nel caso un firewall o un IPS cessi di funzionare, tutta la rete perda la funzionalità insieme ad esso.
L'ultimo difetto è il tempo di processing dei pacchetti. La maggior parte dei firewall e degli IPS controlla ogni pacchetto che arriva con una serie di regole salvate in una lista. Perciò il tempo durante il quale il pacchetto dovrà star fermo all'interno della macchina cresce linearmente con la quantità delle regole da controllare. Se a questo si aggiunge che le firme di un IDS crescono nel tempo, in quanto nuovi attacchi vengono scoperti e quelli vecchi non vengono rimossi per retrocompatibilità, questo problema assume un'importanza rilevante.
Il modello da noi proposto per superare questi problemi è del tutto diverso. Anzichè focalizzarci su pochi host di particolare importanza, ci soffermiamo sugli innumerevoli switch e router che vengono attraversati da un pacchetto durante il suo tragitto. Tutti questi host (termine raramente associato a uno switch, ma nella trattazione che segue sarà appropriato chiamarli così) hanno una limitata capacità di calcolo, insufficiente per far loro svolgere da soli la funzione di firewall o di IPS, ma ampiamente sufficiente per svolgere una parte dell'analisi compiuta per intero da essi.

Immaginiamo allora che tutti i router e gli switch che un pacchetto incontra lo analizzino poco alla volta, decidendo, in una serie di stadi, ognuno dal basso carico computazionale, se lasciarlo proseguire o no. E immaginiamo che il veicolo usato per informare gli host successivi della parte di analisi svolta dai precedenti sia il pacchetto stesso, aumentato da un opportuno campo nell'header IP. Portando quest'idea alle estreme conseguenze, avremmo la totalità dei router e degli switch che compongono internet a parlare questo protocollo. Ogni host controllerebbe i pacchetti per un'infinitesima parte, rendendo il tempo di analisi quasi ininfluente su quello di percorrenza totale. In uno scenario simile, firewall, IDS ed IPS non sarebbero più necessari, perchè la rete nella sua interezza sarebbe un unico macro-host dotato di tutte queste funzionalità.
Scopo di questo lavoro di tesi è quello di gettare le basi per un modello di IPS totalmente distribuito tra gli host interni alla rete (router e switch), al fine di valutarne la realizzabilità.

Parte I

Stato dell'arte

Capitolo 1

Intrusion Detection System

1.1 Funzionamento

1.2 Software Applicativi

1.2.1 Snort

1.2.2 Cisco

Capitolo 2

Intrusion Prevention System

Parte II

Formalizzazione di un modello di IPS distribuito

Capitolo 3

Un modello a stadi per l'analisi del traffico

Come detto, questa tesi ha come oggetto la formalizzazione di un modello di Intrusion Prevention distribuito. Al fine di far questo è necessario, come prima cosa, capire se e come sia possibile separare l'analisi in più parti, delegandole a host diversi. In questo capitolo non ci preoccuperemo di definire come i diversi host dovranno cooperare, nè come faranno a passarsi le informazioni frutto dell'analisi parziale. Questi aspetti verranno trattati in seguito.

3.1 E' possibile usare IPS già esistenti?

La prima domanda da porsi è se sia possibile usare IPS già esistenti, adattandoli a un analisi dei pacchetti di tipo sequenziale effettuata da più host.
Prenderemo come caso di studio snort, che, oltre ad essere uno dei migliori IDS/IPS sulla piazza, è anche il miglior esponente della sua categoria nel parco dei programmi liberi e, come tale, meglio si presta allo studio.
Come già detto nel capitolo relativo allo stato dell'arte, snort fa uso di firme per rilevare gli attacchi. Tali firme sono, per loro stessa natura, atomiche, e non si prestano a venire spezzate. Una tipica firma è la seguente:

```
drop tcp $EXTERNAL_NET any -> $HOME_NET 110 (msg:"POP3 DELE negative
argument attempt"; flow:to_server,established;
content:"DELE"; nocase; pcre:"/^DELE\s+-\d/smi";
metadata:service pop3; reference:bugtraq,6053; reference:bugtraq,7445;
reference:cve,2002-1539; reference:nessus,11570; classtype:misc-attack;
sid:2121; rev:11;)
```

Si tratta di una regola relativa alla posta elettronica. Data una determinata porta di destinazione (110, quella tipica del protocollo pop3) e una stringa riconosciuta all'interno del messaggio ("DELE"), l'IPS si preoccupa di bloccare il messaggio. A parte il fatto

che questa regola dipende fortemente dalla porta di destinazione del pacchetto, fatto che la renderebbe inutile nel caso l'amministratore di sistema decidesse di mettere il proprio server di posta in ascolto su un'altra porta, questa regola non è scomponibile in più di due stadi.
Ciò che si potrebbe fare è rilevare in un primo host che il pacchetto è destinato alla porta 110 e comunicarlo all'host successivo, che si preoccuperà di controllare tutte le regole seguenti. In questo modo avremmo separato l'esecuzione della regola in due passi, e nel contempo avremmo ridotto il carico computazionale dei due host, in quanto il primo avrebbe potuto analizzare soltanto l'header TCP/IP, al fine di rilevare la porta di destinazione, e il secondo si sarebbe occupato del payload.
Sorge però un altro problema: snort processa le regole in maniera sequenziale, quindi per poter avere un risparmio di tempo significativo ogni host dovrebbe caricare soltanto le regole che gli servono. Questo punto però porterebbe a un grande onere di gestione, e potrebbe condurre a errori o mancanze da cui nascerebbero falle nella sicurezza della rete.
Un altro difetto di snort è che, alla versione 2.6.1 che è quella stabile al momento della stesura di questa tesi, è un applicazione monolitica, in cui non è possibile istanziare più thread che processino ognuno un set di regole differente.
L'ultimo punto in sfavore di snort è che la nostra idea sarebbe quella di sviluppare un sistema quanto più indipendente dalla piattaforma, che possa, eventualmente, venire implementato da qualsiasi produttore di apparati di rete. In tale ottica, legarsi a un applicativo specifico non sarebbe stata una strada percorribile.

3.2 Panoramica sul modello di analisi ad albero

Come si sarà capito, il problema che ha un impatto maggiore sulle prestazioni di un IPS come snort è quello di avere una gestione delle regole fatta a lista, che impone una visita di tutte le regole ad ogni pacchetto che attraversa l'apparato.
Il modello che presentiamo utilizza un approccio ad albero, notoriamente più performante di quello sequenziale (Figura 3.1). L'idea è quella che, mano a mano che l'analisi del pacchetto procede, dall'header ethernet in poi, il campo delle possibili regole che si dovranno controllare diminuisca notevolmente. Nel momento in cui, ad esempio, si arriva al campo *EtherType* del frame ethernet e si rileva che questo campo è impostato a 0x0806 per il pacchetto in questione, il modello sa che il pacchetto è una richiesta o una risposta ARP e, agli stadi successivi, processerà soltanto le regole relative a questo protocollo (Figura 3.2).
Il nostro modello, dunque, consiste in un grande albero, in cui, durante la visita, la discesa in un nodo comporta, dal punto di vista dell'analizzatore, la scomparsa di tutte le regole che non sono "figlie" del nodo corrente.
Se si riesce a inoltrare, insieme al pacchetto, l'informazione relativa al nodo dell'albero in cui l'analizzatore si trovava al momento di terminare la sua parte di analisi, ecco che l'host successivo potrà riprendere l'analisi da quel punto, senza neanche visitare tutti i nodi precedenti.
Inoltre, ogni nodo avrà un'etichetta, "Good" o "Bad". Qualora l'analisi di un pacchetto

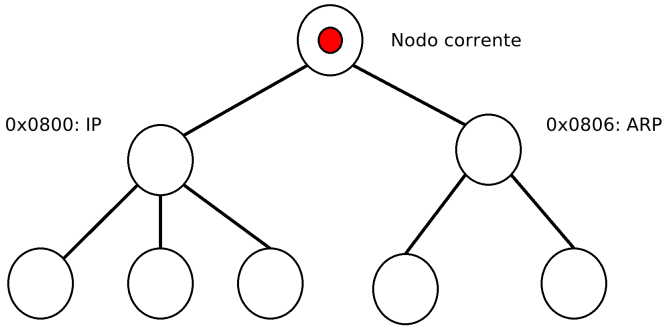

Figura 3.1: Esempio di analisi di un pacchetto - passo 1

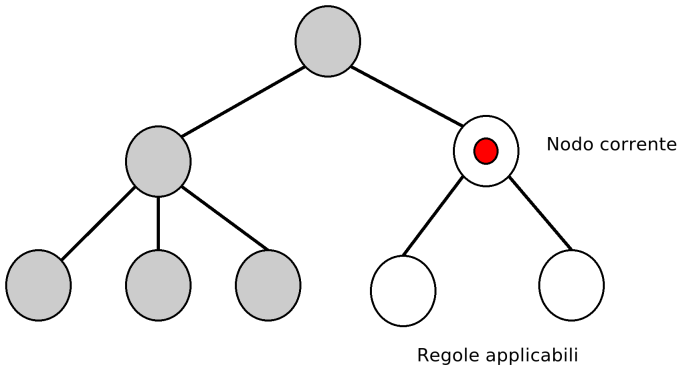

Figura 3.2: Esempio di analisi di un pacchetto - passo 2

porti l'analizzatore in un nodo marcato "Bad", significherà che quel pacchetto è stato riconosciuto come malizioso, e l'analizzatore provvederà a scartarlo, senza inoltrarlo all'host successivo.

3.3 Formalizzazione del modello ad albero

Indice analitico