Distribuited Intrusion Prevention with XYZ

Gianluca Stringhini

A.A 2007/2008

Contents

Introduction

Internet born with quite no security.
Moreover, a network used by researchers and hobbysts (although originally born for military purposes), and animated by knowledge sharing purposes, had as a natural implementation a complete openess, rather than a partial closure.
Although "hacker" culture had, in years '70s and '80s, already spread, security wasn't perceived as a problem. This lack of attention was reasoned by the fact, that the few intrusions usually ended without consequences, with attackers who gained an access to remote machines in order to study them or to get informations who where accademical or, if private property of someone, hardly soldable.
The situation became slightly different when not only millions of people came to populate the network, but also it became critical for political and economical interests of the entire world. A so huge source of informations could have been used to achieve illegal gains, and it wasn't unnoticed. So, together with people who gained access to systems for fun or for slightly harmless purposes, came those who saw this practice as a money source. Quite well-known attacks became more frequent: arbitrary code execution, "bruteforce", "Denial of Service"...
Dangerousness of computer viruses, once limited, because the only way to propragate was via floppy disks got by friends, increased a lot, and millions of computers were infected by malicious software who turned them into "zombies", which are machines who can obey to attackers' will at a given time, to complete a lethal attack.
Computer science developed several devices to fight these menaces.
Antivirus softwares where distribuited, in order to eradicate digital infections, as well as firewall softwares, to deny non authorized connections to hosts. These softwares, however, aren't the point of this job, because we will talk about network security, which deals with everything that stays between two endpoints, and not with the enpoinds themselves.
As regards network security, two elements were mainly developed to watch traffic.
The first element is the firewall. Usually located on the edge of a network, to separate a company's intranet from the rest of the internet, it manages to decide which packets are allowed to be forwarded and which, on the other hand, have to be blocked. At the beginning, the filters they were based on were rather rudimentary, and basically allowed only to check packet headers, and they could have been bypassed easily. Later the so called *stateful firewalls* were developed. These kind of firewalls were able to analyze the whole packet, and even packet streams, recognizing the different kind of traffic.

The second element is the Intrusion Detection System, or IDS. Located inside an Intranet, it analyzes the whole traffic, which usually is replicated by a switch on a given port. Its function is completely passive. It controls the traffic, on behalf of some rules and signals which packets, even if they passed through the firewall, could be part of an attack. It's an useful tool network administrator can use to test their networks' security, and to take countermeasures when security problems are found.
Intrusion Prevention Systems, or IPS, are the evolution of the IDSs. Differently from IDSs, they have an active function. An IPS is located in a transparent way in the network, acting as a bridge through which the whole network traffic passes. On behalf of rules similar to those an IDS bases its analisys, it blocks packets it thinks are dangeous.
This is the current approach used to protect networks. It's an approach that has many limits and many problems, That we're going to explain briefly. First of all, the defense of dozens or thousands of computers is delegated to few machines, usually 2 or 3. A consequence of this fact is that the machines used to act as a firewall or as a IDS usually have to be more powerful than the client machines, in order to be able to manage the whole traffic generated by them. This is why the machines used for this purposes are really expensive. Moreover, it's necessary to have the whole traffic passing through few given paths, in order to heve it analyzed. This has the disadvantage of being a single point of failure: in case a firewall or an IPS ceases to work, the whole network goes down with it.
The last flaw is packet processing time. The majority of firewalls and IPS controls each incoming packet agaiuns a list of rules. This means that the time a packet will have to stay inside the machine increases in a linear way with the amount of the rules to be controlled. Moreover, the signatures of an IPS increase with the proceiding of time, because new attacks are discovered and the old ones are not removed for sake of retrocompatibility.
The model we propose to solve this problems is totally different. Instead of focalize on few important hosts, we put our attention to the numerous switchs and routers a packet passes through during its path. All these "hosts" (term which is seldom associated to a switch, but in the following paragraphs it will be appropriate to use this term for them) have a limited computing capacity, which is insufficient to execute the whole analisys of a packet, but it's enough to execute only a part of that.
Let's imagine to have all routers and switchs a packet encounters analyzing it, each doing a small part of the analysis. They would decide, in a certain number of stadies, each with a small computational effort, whether the packet is malicious or not. And let's imagine to use as a vehicle of information used to inform the following hosts about the part of analysis already completed by the previous ones the packet itself, using an added field in the IP header.
Bringing this idea to ist extreme consequences we would have the whole routers and switchs inside internet to speak this protocol. Each hosts would control the packets for a tiny part, having a processing time that wouldn't impact at all the normal network performance. In such a scenario, firewalls, IDSs and IPSs wouldn't be necessary anymore, because the whole network could be seen as a macro-host providing all these functionalities.
The purpose of this thesis is to put the foundations for a totally distribuited IPS model among hosts inside an intranet, to evaluate if it's realizable or not.

Part I

Phenomenology of the attack

Part II

State of the art

Part III

Formalization of a distribuited IPS model

Chapter 1

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.

1.1 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.

1.2 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 1.1). 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 1.2).
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

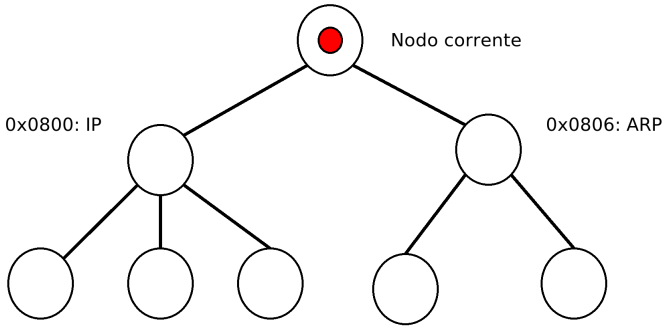

Figure 1.1: Esempio di analisi di un pacchetto - passo 1

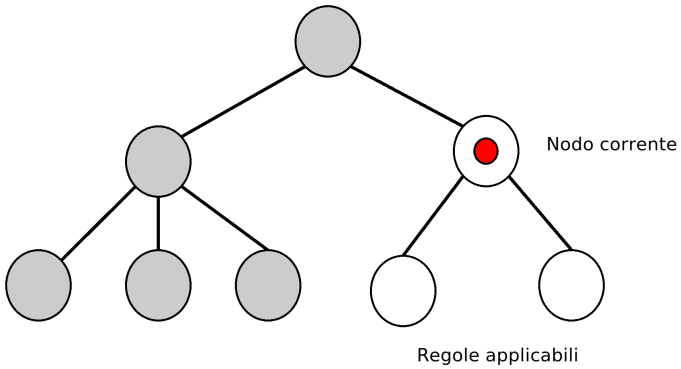

Figure 1.2: Esempio di analisi di un pacchetto - passo 2

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.

1.3 Formalizzazione del modello ad albero

Bibliography

[1] **Peng Ning, Sushil Jajodia, X. Sean Wang**, *Intrusion detection in distribuited systems, an abstraction-based approach*, Kluwer Academic Publishers, 2004.

[2] **Richard Bejtlich**, *Extrusion Detection: Security Monitoring for Internal Intrusions*, 2006.

[3] **Stephen Northcutt, Judy Novak** *Network Intrusion Detection*, New Riders, 2001.

[4] **Michal Zalewski** *Silence on the Wire: A Field Guide to Passive Reconnaissance and Indirect Attacks*, 2006.

[5] **Jon Erickson** *Hacker: the art of exploitation*, 2005

[6] **Giovanni Vigna**, *A topological characterization of TCP/IP security*, 2003.

[7] **Eric Y. Chen**, *Detecting Dos Attacks on SIP Systems*, NTT Corporation, 2006

[8] **D. Katz**, *IP Router Alert Option*, RFC 2113, 1997.

[9] **H. Debar, D. Curry, B. Feinstein**, *The Intrusion Detection Message Exchange Format (IDMEF)*, RFC 4765, 2007.

[10] **Jon Postel**, *Internet Protocol*, RFC 792, 1981.

[11] **Jon Postel**, *Transmission Control Protocol*, RFC 793, 1982.

[12] **Jon Postel**, *User Datagram Protocol*, RFC 793, 1981.

[13] **David C. Plummer**, *Ethernet address resolution protocol*, RFC 826, 1982.