Università degli Studi di Modena

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E INFORMATICHE
Corso di Laurea in Informatica

Tesi di laurea

Titolo tesi

Candidato:
Michele Murgolo
Matricola 101851

Relatori:
Prof. Mirco Marchetti
Prof. Giovanni Apruzzese

Anno Accademico 2017-2018

La pagina della dedica

[**?**]

Sommario

Stratosphere Testing Framework (stf) è una framework di ricerca sulla sicurezza della rete per analizzare i modelli comportamentali delle connessioni di rete nel Progetto Stratosphere. Il suo obiettivo è aiutare i ricercatori a trovare nuovi comportamenti malware, etichettare tali comportamenti, creare i loro modelli di traffico e verificare gli algoritmi di rilevamento. Stf funziona utilizzando algoritmi di apprendimento automatico sui modelli comportamentali. L'obiettivo di Stratosphere Project è creare un IPS comportamentale (Intrusion Detection System) in grado di rilevare e bloccare i comportamenti dannosi nella rete. Come parte di questo progetto, stf viene utilizzato per generare modelli altamente attendibili di traffico dannoso consentendo una verifica automatica delle prestazioni di rilevamento. Il framework genera questi modelli da file in formato binetflow, il DIEF salva il traffico internet in file formato flows. Si è scritto un programma in python3 che esegue la conversione batch da flows a binetflow. I file che il programma deve convertire sono numerosi e di grandi dimensioni, ogni giorno di traffico ha una dimensione media pari a 150Mb. Per effettuare una conversione efficiente si è utilizzato un approccio multicore che ha permesso di ottenere uno speed up lineare della conversione.

Indice

Capitolo 1

Introduzione

La continua digitalizzazione nel mondo sta mettendo le aziende a rischio di attacchi informatici più che mai. Negli ultimi anni, grazie alla crescente adozione di servizi cloud e mobili, la sicurezza delle informazioni ha subito un profondo cambio di paradigma dai tradizionali strumenti di protezione verso l'individuazione di attività dannose all'interno delle reti aziendali.

I metodi di attacco sempre più sofisticati utilizzati dai criminali informatici in diverse recenti violazioni della sicurezza su larga scala indicano chiaramente che gli approcci tradizionali alla sicurezza delle informazioni non possono più tenere il passo.

L'analisi dei dati è l'elemento chiave per sfruttare la resilienza informatica. Con attacchi sempre più avanzati e persistenti e il semplice fatto che ogni organizzazione deve proteggersi da tutte le varietà di attacchi mentre un aggressore ha bisogno solo di un tentativo riuscito, le organizzazioni devono ripensare ai propri concetti di sicurezza informatica: Devono andare oltre la pura prevenzione.

big data security analytics è l'approccio alla base di questo miglioramento del rilevamento. Il rilevamento deve essere in grado di identificare i modelli di utilizzo che cambiano ed eseguire analisi complesse su una varietà di fonti di dati che vanno dai registri di server e applicazioni agli eventi di rete e alle attività degli utenti. Ciò richiede di eseguire analisi su grandi quantità di dati correnti e storici.

Negli ultimi anni è emersa una nuova generazione di soluzioni di analisi della sicurezza, in grado di raccogliere, archiviare e analizzare enormi quantità di dati. Questi dati vengono analizzati utilizzando vari algoritmi di correlazione per rilevare le anomalie e quindi identificare possibili attività dannose. L'industria ha finalmente raggiunto il punto in cui gli algoritmi di intelligenza artificiale per l'elaborazione di dati su larga scala sono diventati accessibili utilizzando framework prontamente disponibili.

Ciò consente di combinare analisi storiche e in tempo reale e identificare nuovi incidenti che potrebbero essere correlati ad altri che si sono verificati in passato. Insieme a fonti di intelligence di sicurezza esterne che forniscono informazioni aggiornate sulle ultime vulnerabilità, ciò può facilitare notevolmente l'identificazione di attacchi informatici in corso sulla rete.

È con l'intenzione di utilizzare queste tecnologie che viene presentato in questa tesi un software per la cattura, l'archiviazione e l'analisi di enormi quantità di dati. Come mostrerò nei seguenti capitoli, l'utilizzo di questi software comporta una organizzazione dei dati notevole e verranno evidenziati in special modo le difficoltà nella gestione dei diversi formati che questi tipi di tecnologie comportano. Questa tesi metterà in evidenza l'eterogeità dei software che hanno sempre contraddistinto l'informatica e le soluzioni scelte per risolvere tali problemi.

Nel secondo capitolo verranno in primo luogo presentate le minacce provenienti dalla rete, con particolare enfasi sui tipi di attacchi su larga scala. Successivamente verranno presentato lo stato dell'arte dei sistemi di difesa utilizzati ad oggi per contrastare questi tipi di attacchi. In conclusione del capitolo verrà discussa la scelta delle tecnologie utilizzate in questa tesi.

Nel terzo capitolo verrà introdotto uno speciale tipo di file che sarà il principale punto di enfasi in questa tesi. Dopo di chè verranno presentati i differenti tipi di formati che questo file può assumere e le problematiche dovute ai diversi standard in uso oggi. Dopo una descrizione dettagliate delle varie differenze tra i formati verrà introdotto il software che farà uso di questi file e si presenterà l'utilizzo che questi file hanno in relazione alle tecnologie utilizzate. Infine troverà spazio la presentazione del problema da affrontare.

Nel quarto capitolo sarà descritto in dettaglio l'installazione del software di cui si farà uso. In seguito si descriveranno le scelte effettuate per risolvere il problema nell'uso di diversi formati di file e l'automatizzazione di tale soluzione. Infine saranno presentati i diversi metodi atti a perfezionare l'automatizzazione per renderla efficiente e la scelta che è stata effettuata.

Nel quinto capitolo verranno mostrati i risultati dei test e i benchmark effettuati con lo scopo di confermare al livello pratico quanto mostrato sotto forma teorica nel capitolo precedente. Saranno descritte in modo dettagliato le condizioni sotto le quali sono stati effettuati i test e limiti della scalabilità della soluzione. Il tutto verrà seguito da grafici esplicativi.

Nel sesto capitolo infine, verrà fatto un riassunto della tesi ribadendo l'obiet-

tivo, cosa si è svolto in questa tesi e i risultati ottenuti.

Capitolo 2

Stato dell'arte

In questo capitolo verranno discussi i principali lavori che sono stati fatti durante l'ultimo decennio in letteratura. Si procede col fornire un'introduzione preliminare sui concetti che verranno utilizzati durante questa tesi.

2.1 Malware

Il termine malware è una combinazione delle parole *malicious* e *software*. Il malware rappresenta quei programmi software progettati per danneggiare o effettuare azioni indesiderate su un sistema informatico. [3]

Gli obiettivi che può avere un malware sono molteplici e sono in continua evoluzione. Il malware, a seconda dello scopo per cui è stato creato e alle sue caratteristiche, viene classificato nei seguenti modi:

Virus Prendono il nome dai virus in campo biologico e si comportano in modo analogo, sono programmi che si replicano sul computer che hanno infettato e si predispongono ad infettare nuovi computer mediante mezzi di trasmissione quali email e chiavette USB. [5]

Spyware Il termine Spyware è una combinazione delle parole *Spy* e *Software*. È un software che viene installato sul computer della vittima a sua insaputa e che raccoglie informazioni. Uno spyware è oggetto di controversia perchè può essere utilizzato negli ambienti lavorativi per controllare le ricerche dei dipendenti o per controllare l'attività dei propri figli su internet. Anche se utilizzato per scopi più innocui può comunque violare la privacy dell'utente. [6]

Usi più scorretti di programmi spyware prevedono di tracciare la cronologia internet di un utente per inviare pubblicità mirata, accedere alle password degli account in uso sul computer infetto e/o alle informazioni bancarie. Le informazioni raccolte attraverso l'uso di spyware possono essere utilizzate in vari modi, l'uso più frequente e più remunerativo ad oggi è quello di rivendere tali informazioni a dei terzi. [9]

Backdoor Tradotto letteralmente come *porta sul retro*, è un metodo utilizzato per avere un accesso privilegiato e spesso segreto che aggira il sistema di autenticazione previsto. Lo scopo di una backdoor è quello di permettere una connessione in remoto al computer vittima per prenderne il controllo.

Trojan Il cavallo di Troia fu una macchina da guerra che, secondo la leggenda, fu usata dai greci per espugnare la citta di Troia. Questo termine è entrato nel linguaggio comune per indicare uno stratagemma con cui penetrare le difese. Nell'ambito dei malware il trojan è un software che si nasconde all'interno di un altro programma all'apparenza innocuo e che, se eseguito, esegue anche il codice del trojan [7]. Oggi col termine trojan ci si riferisce principalmente ai malware ad accesso remoto. Spesso vengono utilizzati per installare backdoor sui sistemi bersaglio. [8]

I malware erano inizialmente usati per compiere azioni dolose sia da hacker malintenzionati che dai governi per sottrarre informazioni personali, inviare spam e commettere frodi. [2] [4]

L'evoluzione e lo sviluppo di internet hanno portato ad un incremento degli utenti connessi sempre maggiore. Questa crescita di internet ha spostato l'obiettivo dei malware che vengono usati sempre di meno per compiere azioni dolose. Fin dal 2003 la maggior parte dei malware sono stati creati per prendere il controllo dei computer dell'utente vittima per scopi illeciti [4]. Vengono usati computer zombie per l'invio di email di spam o per effettuare attacchi distribuiti Denial of Service (DDoS).

2.2 Botnet

Nella sua forma più semplice una Botnet è un gruppo di computer che sono stati infettati da un malware che consente al suo controller, detto anche master, di avere il controllo sulle macchine infettate. Le Botnet sono usate dal master per compiere operazioni illecite ad insaputa della vittima. Una volta infetto, il computer della vittima prende il nome di zombie. [1]

Per aggiungere un computer ad una botnet si infetta il computer vittima con un malware che installa una backdoor in grado di consentire al master di avere accesso remoto al computer infettato dal malware. Il master ha così accesso ad un sistema gigantesco di computer zombie pronti ad essere attivati ed eseguire i suoi ordini. Le botnet rilevate e studiate nella storia dimostrano come questi sistemi possano arrivare a contenere anche milioni di computer infetti. [1]

I componenti di una botnet sono i seguenti:

Master Il master è il computer che ha creato la botnet e che ne ha il controllo remoto. È detto C&C (Command and Control) il programma che dà i comandi da eseguire tramite un canale nascosto sul computer della vittima.

Control protocol Il protocollo utilizzato dal master per comunicare con i computer zombie.

Computer zombie Computer connesso ad internet che è stato infettato attraverso dei virus o trojan e che può essere utilizzato in modo remoto.

Nella figura seguente viene rappresentato una possibile architettura di una botnet. Il master è il computer che controlla in modo remoto i computer zombie attraverso dei server C&C.

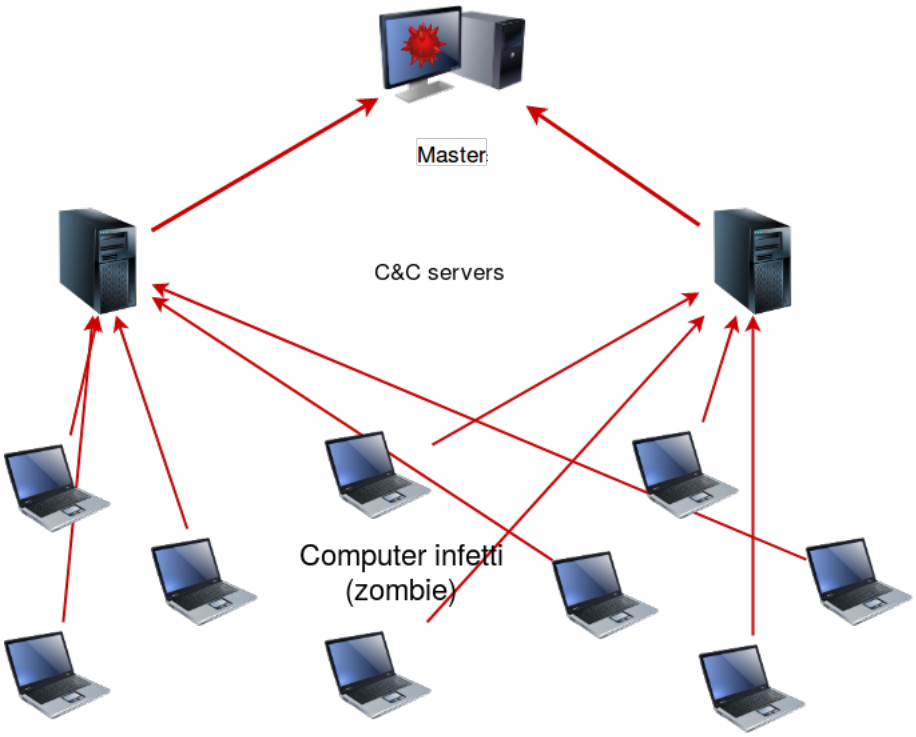


Figura 2.1: Esempio architettura botnet

I principali attacchi informatici effettuati attraverso l'utilizzo delle botnet sono vari: possono essere spam, DDoS, click fraud e bitcoin mining per citarne alcuni. È quindi evidente come l'utilizzo delle botnet sia al fine di un guadagno economico.

Spam È l'invio tramite posta elettronica di messaggi ad alta frequenza contenenti truffe o pubblicità.

DDoS Acronimo di Distributed Denial of Service, è un tipo di attacco in cui si mira a far esaurire le risorse di un sistema informatico affinchè non riesca più a fornire servizio. Per fare ciò si effettuano moltissime richieste al sistema per farlo rallentare o nei casi più estremi rendere inutilizzabile.

Click fraud Questo tipo di frode si verifica sulla pubblicità su internet di tipo pay per click. Questo tipo di pubblicità genera quantità di denaro in base al numero di click effettuati sull'annuncio pubblicitario. Le botnet sfruttano questo tipo di pubblicità utilizzando computer zombie per cliccare in massa gli annunci pubblicitari.

Bitcoin mining I bitcoin sono una criptovaluta che a differenza dei soldi, che vengono stampati da un governo centrale, vengono creati risolvendo operazioni matematiche complesse. Una botnet utilizza i computer zombie come unità di calcolo a cui far eseguire queste operazioni complesse.

2.3 Intrusion Detection and Prevention System

Un'intrusione si verifica quando un aggressore tenta di entrare o interrompere le normali operazioni di un sistema informativo, quasi sempre con l'intento di fare del male [12]. In questa sezione verranno presentati dei dispositivi utilizzati per identificare accessi non autorizzati ai computer o alle reti locali. Questi dispositivi possono essere utili per il rilevamento di botnet [13].

I sistemi di rilevamento delle intrusioni sono gli "allarmi antifurto" del campo della sicurezza informatica. L'obiettivo è difendere un sistema con un allarme che viene emesso ogni volta che la sicurezza del sito è stata compromessa [10].

Un'attuale estensione degli IDS sono i sistemi di prevenzione delle intrusioni detti IPS, che possono rilevare un'intrusione e impedire che l'intrusione attacchi con successo tramite una risposta attiva. Gli IPS usano diverse tecniche di risposte attive, che possono essere suddivise nei seguenti gruppi [11]:

- **terminare** la connessione internet o la sessione utente che sta eseguendo l'attacco
- **bloccare** l'accesso all'obiettivo e agli obiettivi simili all'utente o indirizzo IP che sta tentando un'intrusione
- **cambiare** l'ambiente di sicurezza modificando la configurazione di altri controlli di sicurezza per interrompere l'attacco, come la riconfigurazione delle regole di un firewall. Alcuni IPS possono anche applicare della patch di sicurezza a degli host se l'IPS rileva che l'host presenta delle vulnerabilità.

Poiché i due sistemi coesistono spesso, il termine combinato sistema di rilevamento e prevenzione delle intrusioni (IDPS) viene generalmente utilizzato per descrivere le attuali tecnologie anti-intrusione [11].

2.3.1 Perchè usare IDPS

Dispositivi di tipo IDPS sono utili per le difese di sistemi informatici per molteplici ragioni:

- **defense in depth**, l'utilizzo di un dispositivo IDPS unito a firewall, controlli di accesso e autenticazione e antivirus permette la realizzazione di un meccanismo di protezione multi-livello
- **documentazione**, i dati acquisiti sono utili anche per il miglioramento continuo della qualità, gli IDPS raccolgono costantemente informazioni sugli attacchi che hanno compromesso con successo gli strati esterni dei controlli di sicurezza, per esempio di un firewall. Queste informazioni possono essere utilizzate per identificare e riparare le vulnerabilità esposte dagli attacchi, questo aiuta l'organizzazione ad accelerare il processo di risposta agli incidenti e ad apportare miglioramenti continui. Inoltre, nei casi in cui un IDPS non riesce a prevenire un'intrusione, può ancora assistere nella revisione fornendo informazioni su come si è verificato l'attacco, i dettagli su cosa ha fatto l'intruso e i metodi utilizzati. Gli IDPS possono anche fornire informazioni forensi che possono essere utili per motivi legali qualora l'attaccante venga catturato [11]
- **insider threat**, gli attacchi non sempre arrivano dall'esterno, con questi dispositivi è possibile difendersi anche da intrusioni che avvengono all'interno della rete

Il miglior motivo per installare un IDPS è per la loro funzione di deterrente. Se gli utenti esterni e interni sanno che un'organizzazione ha un sistema di rilevamento e prevenzione delle intrusioni sono meno propensi a sondare o tentare di comprometterla [11].

2.3.2 Tipi di IDPS

Gli IDPS possono essere di due tipi in base al target su cui vengono applicati: possono essere basati su rete o su host. Un IDPS basato su rete è focalizzato sulla protezione delle risore di rete. Due sottotipi di IDPS basati su rete sono [11]:

- **wireless IDPS**, si concentra sulle reti wireless
- **network behavior analysis**, analizza i flow di traffico sulla rete nel tentativo di riconoscere dei modelli comportamentali anomali

Gli IDPS basati su host sono focalizzate sulla protezione dei server o dei singoli host.

Network-Based IDPS Un IDPS basato sulla rete, detto NIDPS, è installato su un computer o dispositivo collegato ad un segmento

della rete e ne monitora il traffico alla ricerca di indicazioni di attacchi. Un IDPS basato su rete può rilevare molti più attacchi rispetto ad un IDPS basato su host, ma richiede una configurazione e manutenzione più complessa.

Host-Based IDPS Un IDPS basato su host, detto HIDPS, risiede su un particolare computer o server e monitora l'attività solo su quel sistema.

2.3.3 Metodi di rilevamento

Le regole utilizzate per identificare le violazioni possono essere [10]:

Signature detection La decisione di rilevamento delle intrusioni è formata sulla base di un modello del processo intrusivo e di quali tracce dovrebbe lasciare nel sistema osservato. Questo metodo si basa sul presupposto che in qualsiasi caso riusciamo a definire un comportamento legale o illegale e confrontare di conseguenza il comportamento osservato.

Un vantaggio di questo approccio è che avendo un modello con cui confrontare il comportamento che si sta osservando non esistono falsi positivi, ovvero quando si considera dannoso un comportamento innocuo generando un falso allarme, questo perchè i modelli comportamentali all'interno del database sono comportamenti per certo dannosi. È inoltre veloce e semplice in quanto si tratta soltanto di effettuare confronti con modelli illegali già noti.

Uno svantaggio di questo approccio è che si possono verificare molti falsi negativi, cioè i comportamenti dannosi che non vengono segnalati. Questo si verifica perchè questi sistemi sfruttano regole per rilevare le intrusioni salvate su di un database, è quindi estremamente importante tenere aggiornato il database con tutti i possibili comportamenti dannosi.

Anomaly detection In questo tipo di rilevamento si parte con il presupposto che qualcosa di anormale sia molto probabilmente sospetto, si cercano quindi anomalie nel traffico. È quindi necessario uno studio anticipato della rete per capire cosa sia normale per il soggetto osservato. Si decide poi quanto è possibile discostarsi dal tipo di attività normale. Questo tipo di rilevamento guarda comportamenti che sono improbabili che si verifichino dallo studio effettuato sul traffico normale.

Il vantaggio di questo metodo è che indipendente dal tipo di intrusione ed è possibile utilizzare algoritmi di *machine learning* che

apprendono da soli cosa è normale osservando il traffico per un lungo periodo di tempo.

Gli svantaggi sono la difficoltà di creare modelli e l'imprecisione che hanno questi modelli che danno vita a molti falsi positivi e falsi negativi.

Un IDPS è un dispositivo software o hardware utilizzato per identificare intrusioni a computer o reti locali. Ci sono varie classi di intrusione che vanno rilevate: si possono verificare situazioni in cui un utente ruba una password, utenti legittimi che abusano dei loro privilegi o hacker che usano script trovati in rete per attaccare il sistema. Le intrusioni sono varie e non è possibile elencarle tutte.

Un IDPS è composto da quattro componenti [12]:

- **Sensori**, sono utilizzati per ricevere informazioni dalla rete o dai computer.
- **Console**, utilizzata per monitorare lo stato della rete e dei computer.
- **Motore**, analizza i dati prelevati dai sensori e provvede a individuare eventuali falle nella sicurezza.
- **Database**, memorizza le regole utilizzate per identificare violazioni di sicurezza.

Di seguito è presente un'immagine che descrive i componenti di un IDS. I pacchetti in entrata e uscita dalla rete sono ricevuti da un sensore che raccoglie il traffico dati, successivamente il motore analizza i dati prelevati dai sensori con le regole presenti nel database.

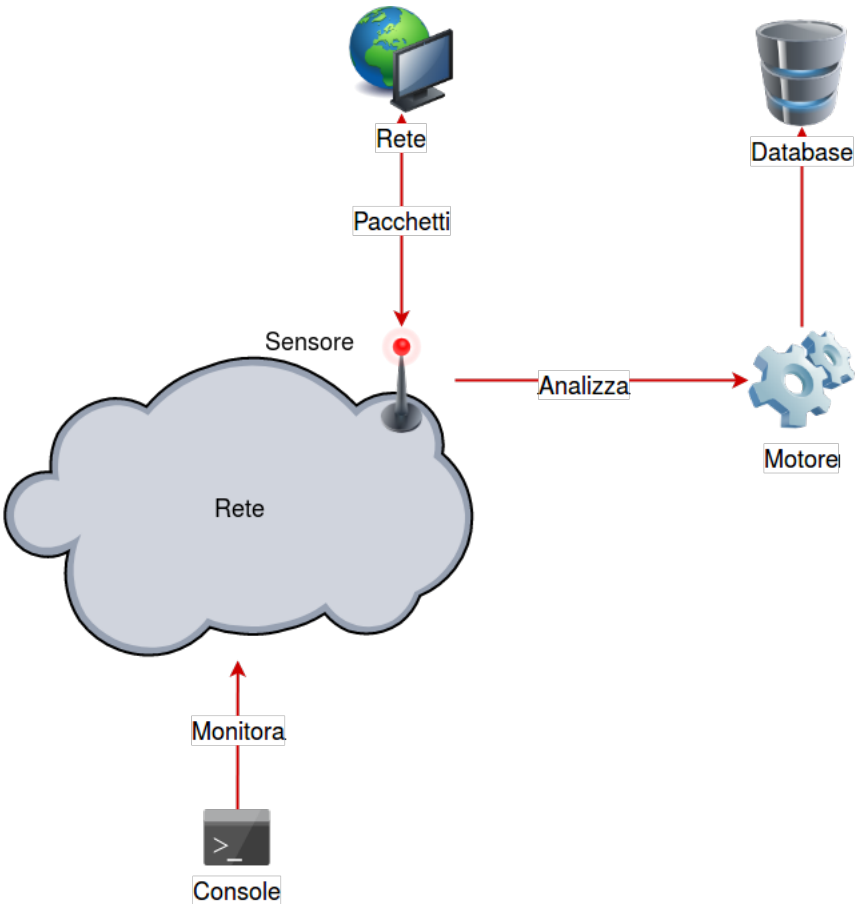


Figura 2.2: Esempio di un IDS

Esistono diversi tipi di tecniche per rilevare le intrusioni. In questa tesi si è fatto uso di un *IDS* che utilizza algoritmi di *machine learning*.

Machine learning può essere definito come la capacità di un programma per computer di apprendere e migliorare le prestazioni su una serie di attività nel tempo. Le tecniche di *machine learning* si concentrano sulla costruzione di un modello, *behavioral model*, di sistema che migliora le sue prestazioni in base ai risultati precedenti.

L'utilizzo di *IDS* è un approccio utile per fare *botnet detection* sul traffico di rete, si osserva il traffico di dati nella rete e si cercano comunicazioni sospette che possono essere fornite da *bot*.

Capitolo 3

Analisi del problema

3.1 Analisi traffico di rete

Nell'era digitale il traffico di rete è aumentato notevolmente e con esso gli attacchi di tipo informatico, per questo motivo è necessario trovare soluzioni per analizzare questo grande quantitativo di pacchetti che attraversano i dispositivi di rete delle aziende di grosse dimensioni. Nel mondo informatico di oggi è molto importante poter determinare rapidamente e con precisione l'origine e la portata di un potenziale attacco su una rete al fine di poterlo contrastare in modo efficace. Per fare ciò viene costruito un *audit trail* di informazioni collezionate dal traffico di rete usando una combinazione di network flow e PCAP.

Un audit trail è un file che contiene una registrazione cronologica di attività relative alla sicurezza per consentire la ricostruzione e l'esame di eventi.

3.1.1 Network Flow

Un flow è una sequenza di pacchetti inviati da una sorgente ad una destinazione che hanno degli attributi in comune:

- indirizzo IP sorgente
- indirizzo IP destinazione
- porta sorgente
- porta destinazione
- protocollo

Se i pacchetti che attraversano un dispositivo di rete hanno questi attributi in comune possono essere raggruppati in un flow.

3.1.2 Packet Capture

Per packet capture (PCAP) si intende la cattura di traffico internet che attraversa un dispositivo di rete. Un packet capture intercetta i singoli pacchetti e li archivia. Nei sistemi operativi Unix è utilizzata la libreria **libpcap** mentre nei sistemi Windows si fa utilizzo di **WinPcap**

3.1.3 NetFlow

NetFlow è un protocollo di analisi di rete —- proprietario — di Cisco che offre la possibilità di raccogliere informazioni dettagliate sul traffico mentre attraversa un'interfaccia. I dispositivi di rete — conformi —- a NetFlow possono raccogliere statistiche sul traffico ed esportarle come record verso un NetFlow collector, un server che esegue l'analisi del traffico.

Cisco definisce un flow come una sequenza unidirezionale di pacchetti che condividono tutti i seguenti 7 valori:

- interfaccia di ingresso
- indirizzo IP sorgente
- indirizzo IP destinazione
- protocollo IP
- porta sorgente TCP o UDP, 0 per altri protocolli
- IP Type of Service

— fare sezione netflow vs pcap ——

3.1.3.1 Componenti di NetFlow

—- Una architettura NetFlow (network flow) ha i seguenti componenti —- introdurre la figura

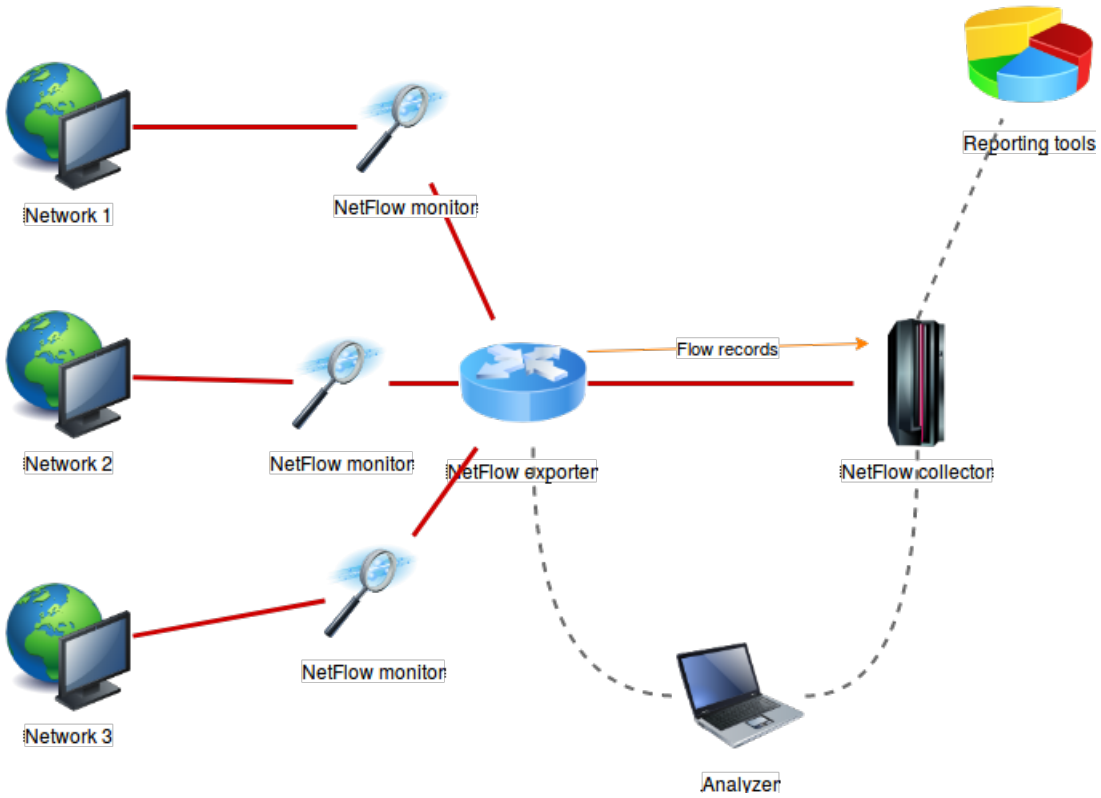


Figura 3.1: Architettura NetFlow

NetFlow monitor un componente applicato a un'interfaccia che raccoglie informazioni sui flow. I NetFlow monitor sono costituiti da un record e una cache.

NetFlow exporter Aggrega i pacchetti in flows e ne esporta i record verso uno o più *flow collectors*. Quando dei pacchetti arrivano al NetFlow exporter, vengono ispezionati singolarmente per uno o più attributi che vengono utilizzati per determinare se il pachetto è univoco o è simile agli altri pacchetti. Se il pacchetto presenta attributi simili viene classificato nello stesso flow.

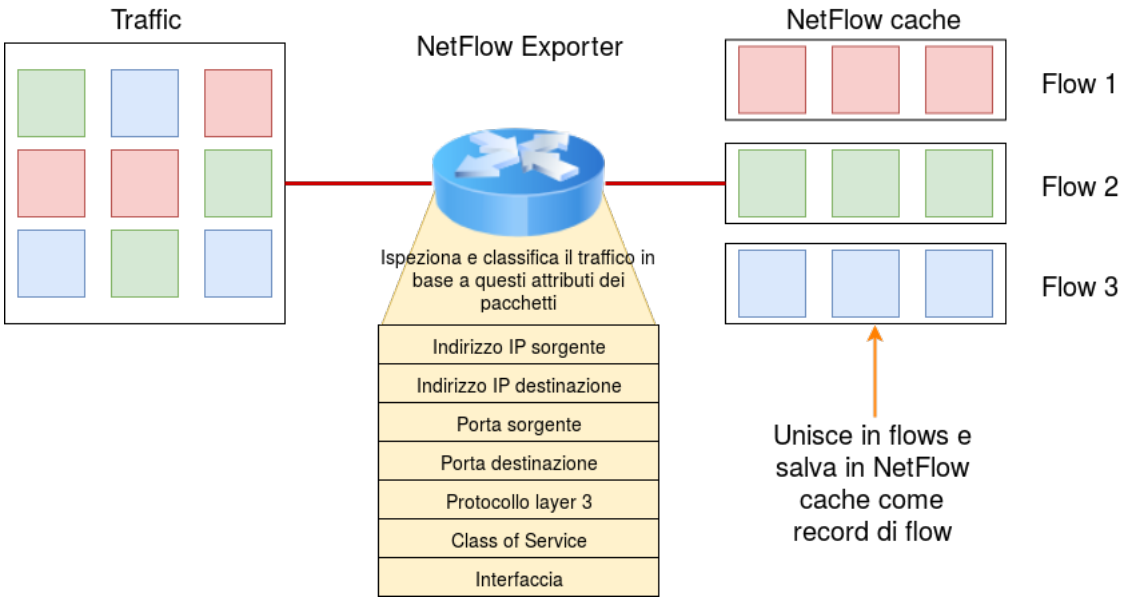


Figura 3.2: NetFlow exporter

Dopo aver esaminato questi attributi, il NetFlow exporter li aggrega in record di flow e li salva in un database che può essere una cache NetFlow o un NetFlow collector.

NetFlow collector Responsabile della ricezione, conservazione e pre-elaborazione dei dati di un flow ricevuti da un *flow exporter*. Solitamente è un software separato in esecuzione su un server di rete. I record NetFlow vengono esportati in un NetFlow collector tramite protocollo UDP.

—- si possono utilizzare ids

Analysis application Analizza i dati dei flows ricevuti nel contesto del rilevamento delle intrusioni o del profilo di traffico. ——

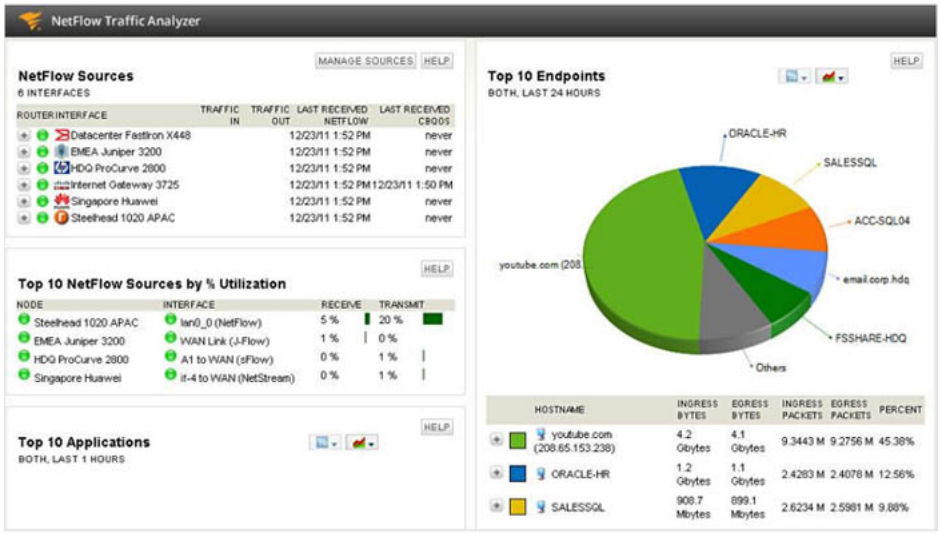


Figura 3.3: SolarWinds NetFlow Traffic Analyzer

—- il nome della sezione va cambiato —

3.2 Strumenti software utilizzati

—- riscrivere meglio —- Verranno ora descritti i software utilizzati in questa tesi. I software utilizzati sono multipiattaforma, in questa tesi sono stati adoperati su sistema operativo basato su una distribuzione GNU/Linux.

3.2.1 Audit Record Generation and Utilization System

Argus (Audit Record Generation and Utilization System) è stato la prima implementazione del monitoraggio dei flow, è un progetto open source e multipiattaforma. Quest'ultima particolarità lo rende molto interessante poichè supportando molti sistemi operativi, tra i quali Windows, MaxOSX, Linux, Solaris, FreeBSD, OpenBSD, IRIX e OpenWrt, può essere adoperato in quasi tutte le reti comprendendo la maggior parte degli host. La sua architettura è di tipo server/client. Il server recupera i pacchetti ricevuti da una o più interfacce di rete disponibili su una una macchina, argus assembla poi questi pacchetti in dati binari che rappresentano dei flow. Lo scopo dei client è di quello di leggere i dati dei flow.

Argus viene utilizzato da molte università e aziende per registrare dei flow che vengono utilizzati sia nell'analisi immediata dell'utilizzo della rete, sia nell'analisi storica. —- I record Netflow di Argus offrono un rapporto fino a 10.000:1 dalla dimensione del pacchetto al record scritto sul disco, che consente alle installazioni

di salvare i record per molto più tempo rispetto alle acquisizioni di pacchetti completi.
——

La seguente tabella descrive i campi dei flow che argus salva su file. —- introdurre tabella —-

campo	**descrizione**
StartTime	record start time
Dur	record total duration
Proto	transaction protocol
SrcAddr	source IP address
Sport	source port number
Dir	direction of transaction
DstAddr	destination IP address
Dport	destination port number
State	transaction state
sTos	source TOS byte value
dTos	destination TOS byte value
TotPkts	total transaction packet count
TotBytes	total transaction bytes
SrcBytes	src ->dst transaction bytes
srcUdata	source user data buffer
dstUdata	destination user data buffer
Label	metadata label

3.2.2 nProbe

Negli ambienti commerciali, NetFlow è probabilmente lo standard de facto per la —- contabilità e la fatturazione —- del traffico di rete. nProbe è un software in grado di reccogliere, analizzare ed esportare report sul traffico di rete utilizzando il formato standard Cisco NetFlow. È disponibile per la maggior parte dei sistemi operativi sul mercato.

La tabella seguente descrive i campi dei flow che il software nProbe salva su file.

campo	descrizione
IPV4_SRC_ADDR	IPv4 source address
IPV4_DST_ADDR	IPv4 destination address
IPV4_NEXT_HOP	IPv4 next hop address
INPUT_SNMP	input interface SNMP idx
OUTPUT_SNMP	output interface SNMP idx
IN_PKTS	incoming flow packets (src ->dst)
IN_BYTES	incoming flow bytes (src ->dst)
FIRST_SWITCHED	SysUptime (msec) of the first flow pkt
LAST_SWITCHED	SysUptime (msec) of the last flow pkt
L4_SRC_PORT	IPv4 source port
L4_DST_PORT	IPv4 destination port
TCP_FLAGS	cumulative of all flow TCP flags
PROTOCOL	IP protocol byte
SRC_TOS	Type of service byte
SRC_AS	source BGP AS
DST_AS	destination BGP AS
IPV4_SRC_MASK	IPv4 source subnet mask
IPV4_DST_MASK	IPv4 dest subnet mask
L7_PROTO	layer 7 protocol (numeric)
BIFLOW_DIRECTION	1=initiator, 2=reverseInitiator
FLOW_START_SEC	seconds (epoch) of the first flow packet
FLOW_END_SEC	seconds (epoch) of the last flow packet
OUT_PKTS	outgoing flow packets (dst ->src)
OUT_BYTES	outgoing flow bytes (dst ->src)
FLOW_ID	serial flow identifier
FLOW_ACTIVE_TIMEOUT	activity timeout of flow cache entries
FLOW_INACTIVE_TIMEOUT	inactivity timeout of flow cache entries
IN_SRC_MAC	source MAC address
OUT_DST_MAC	destination MAC address

—- argus vs nprobe —-

3.3 Stratosphere IPS

In questa tesi si è utilizzato Stratosphere IPS, un software open source disponibile per sistemi operativi Windows e distribuzioni GNU/Linux. In questa tesi si è fatto utilizzo esclusivo della versione per distribuzioni GNU/Linux. Stratosphere Linux IPS (slips), al —corrente—-pubblicato come alpha, è un sistema di rilevazione e prevenzione delle intrusioni comportamentale che utilizza algoritmi di machine learning per rilevare comportamenti dannosi. L'obiettivo di Strato-

sphere IPS è quello di creare un Intrusion Prevention System che può rilevare e bloccare comportamenti malevoli all'interno di una rete.

—- introdurre nello stato dell'arte e descrivere qui in dettaglio il tipo —

Algoritmi di machine learning insieme di metodi di calcolo che utilizzano l'esperienza per migliorare le prestazioni o per effettuare delle previsioni accurate.

—- riscrivere —-

3.3.1 Architettura

Slips si occupa della parte di rilevazione attraverso machine learning. Il traffico di rete viene letto da e salvato da Argus che rimane in ascolto su una porta. Slips legge poi i flows passati da standard input dal client di Argus. In questo modo è possibile analizzare il traffico del proprio computer o di qualsiasi altra rete su cui sia in esecuzione Argus.

— mettere un disegno dell'architettura —-

3.3.2 Modelli comportamentali

Slips fa utilizzo di modelli comportamentali creati da Stratosphere Testing Framework.

— dire all'inizio della sezione la divisione del software in slips e stf —-

Stratosphere Testing Framework è un framework di ricerca sulla sicurezza della rete per analizzare i modelli comportamentali delle connessioni di rete. Il suo obiettivo è aiutare i ricercatori a trovare nuovi comportamenti malware, etichettare tali comportamenti, creare i loro modelli di traffico e verificare gli algoritmi di rilevamento. Una volta creati e verificati i migliori modelli comportamentali di malware, questi verranno utilizzati da slips per il rilevamento. Stratosphere Testing Framework usa algoritmi di machine learning sui modelli comportamentali.

Il nucleo di *Stratosphere IPS* è composto dai modelli comportamentali di reti e algoritmi di rilevamento. I modelli comportamentali rappresentano ciò che una connessione specifica fa nella rete durante la sua vita. Il comportamento è costuito analizzando la sua periodicità, le dimensioni e la durata di ciascun

flusso. Sulla base di queste caratteristiche a ciascun flusso viene assegnata una lettera e il gruppo di lettere caratterizza il comportamento della connessione.

— dire i campi che influenzano i modelli —

Prendiamo come esempio una connessione generata da una botnet che ha il seguente modello comportamentale

88∗y∗y∗ i ∗H∗H∗H∗y∗0yy∗H∗H∗H∗y∗y∗y∗y∗y∗H∗h∗y∗h∗h∗H∗H∗h∗H∗y∗y∗y∗H∗

In questo caso ci dice che i flussi sono altamente periodici (lettere h, i), con qualche periodicità persa vicino all'inizio (lettere y). I flussi hanno anche una grande dimensione con una durata media. I simboli tra le lettere sono correlati al tempo trascorso tra i flussi. In questo caso il simbolo '*' significa che il flusso è separato da meno di un'ora. Con l'utilizzo di questo tipo di modelli siamo in grado di generare le caratteristiche comportamentali di un gran numero di azioni dannose. L'immagine seguente mostra i criteri di assegnazione delle lettere per i modelli comportamentali

— introdurre l'immagine —-

	Size Small			Size Medium			Size Large		
	Dur. Short	Dur. Med.	Dur. Long	Dur. Short	Dur. Med.	Dur. Long	Dur. Short	Dur. Med.	Dur. Long
Strong Periodicity	a	b	c	d	e	f	g	h	i
Weak Periodicity	A	B	C	D	E	F	G	H	I
Weak Non-Periodicity	r	s	t	u	v	w	x	y	z
Strong Non-Periodicity	R	S	T	U	V	W	X	Y	Z
No Data	1	2	3	4	5	6	7	8	9

Symbols for time difference:

Between 0 and 5 seconds:	.
Between 5 and 60 seconds:	,
Between 60 secs and 5 mins:	+
Between 5 mins and 1 hour:	*
Timeout of 1 hour	0

Figura 3.4: Tabella modelli comportamentali

3.4 Problematiche dovute all'utilizzo di due diversi formati

I file prodotti da nProbe e Argus, oltre ad avere formati diversi, hanno anche campi eterogenei. Stratosphere IPS si appoggia ad Argus per la cattura di file

e di conseguenza può leggere file solo in formato binetflow. Se una azienda si appoggia a Cisco utilizzando nProbe non può quindi fare utilizzo di questo software, poichè non riesce a leggere file in formato differente.

Questa incompatibilità ha portato alla necessità di una conversione: i file prodotti da nProbe devono essere convertiti in file con formato usato da Argus. Questa conversione deve essere precisa ed efficiente.

3.5 Presentazione del problema

I file prodotti da nProbe hanno una struttura gerarchica fissa ben definita: ci sono 4 livelli di subdir, in cui il primo livello indica l'anno, il secondo il mese, il terzo il giorno e il quarto l'ora. All'interno dell'ultima subdir, quella delle ore, ci sono 60 file uno per ogni minuto della giornata.

— introdurre l'immagine —-

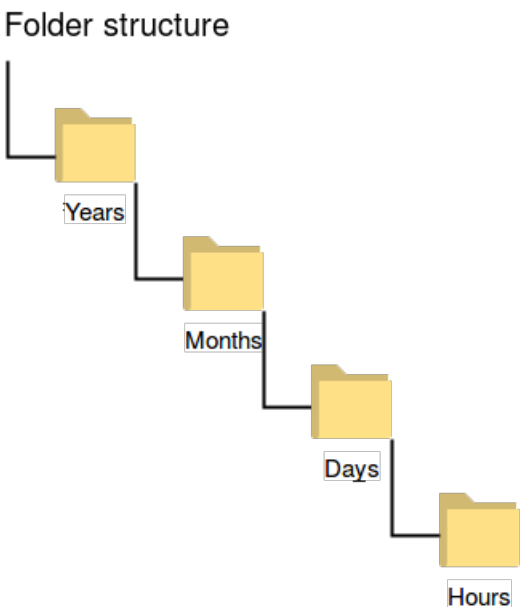


Figura 3.5: Folder Structure

—- togliere —- I file dei minuti sono compressi usando il programma *gzip*, pertanto c'è da tenerne conto nella soluzione per la conversione.

Capitolo 4

Soluzione proposta

— outline del capitolo —-

Per lo sviluppo del programma per la conversione dei file e per l'utilizzo di Stratosphere IPS è stata creata una piattaforma dedicata alla Security Analytics tramite l'utilizzo di *VirtualBox* con una installazione del sistema operativo Ubuntu 16.04 LTS.

— mettere nei software utilizzati —

VirtualBox è un software gratuito e open source per l'esecuzione di macchine virtuali che supporta Windows, GNU/Linux e macOS come sistemi operativi host ed è in grado di eseguire Windows, GNU/Linux, OS/2 Warp, BSD come ad esempio OpenBSD, FreeBSD e infine Solaris e OpenSolaris come sistemi operativi guest.

— mettere nella sezione 5 —-

4.1 Installazione Stratosphere IPS

Sulla macchina virtuale è stato installato il framework di Stratospehere IPS. Per l'installazione si sono seguiti i seguenti passaggi:

- Installazione del programma git 2.7.4

  ```
  $ sudo apt install git
  ```

- Clonazione repository github del framework

```
$ git clone https://github.com/stratosphereips/StratosphereTestingFrame
```

- Installazione del programma python-pip

  ```
  $ sudo apt install python-pip
  ```

- prettytable 0.7.2-3

  ```
  $ sudo apt install python-prettytable
  ```

- transaction 1.4.3-3

  ```
  $ sudo apt install python-transaction
  ```

- persistent 4.1.1-1build2

  ```
  $ sudo apt install python-persistent
  ```

- zodb 5.4.0

  ```
  $ sudo pip install zodb
  ```

- sparse 1.1-1.3build1

  ```
  $ sudo apt install python-sparse
  ```

- dateutil 2.4.2-1

  ```
  $ sudo apt install python-dateutil
  ```

4.2 Installazione di Argus

- libpcap 1.7.4-2

  ```
  $ sudo apt install libpcab-dev
  ```

- bison 3.0.4

  ```
  $ sudo apt install bison
  ```

- flex 2.6.0-11

  ```
  $ sudo apt install flex
  ```

- Installazione dell'ultima versione di argus 3.0.8.2 dal sito http://qosient.com/argus/dev/argus-latest.tar.gz
- Installazione dell'ultima versione di argus-client 3.0.8.2 dal sito http://qosient.com/argus/dev/argu clients-latest.tar.gz

4.3 Utilizzo del programma stf

Per eseguire il programma lo si esegue con

```
./stf.py
```

```
    Stratosphere Testing Framework
            _    __
           | |  / _|
        ___| |_| |_
       / __| __|  _|
       \__ \ |_| |
    ... |___/\__|_|  ...
    0.1.2alpha


[*] Amount of experiments in the DB so far: 0
[*] Amount of datasets in the DB so far: 0
[*] Amount of groups of connections in the DB so far: 0
[*] Amount of groups of models in the DB so far: 0
[*] Amount of notes in the DB so far: 0
stf >
```

Per caricare un dataset si utilizza il comando

```
datasets -c /absolute/path/file.binetflow
```

Per generare la connessione si utilizza il comando

```
connections -g
```

Infine, per generare i modelli, il comando

```
models -g
```

Per visualizzare il behavioral model si utilizza il comando

```
models -L [id]
```

```
test: stf > models -l
[] Groups of Models
+------------------+------------------+------------+--------------+
| Group of Model Id | Amount of Models | Dataset Id | Dataset Name |
+------------------+------------------+------------+--------------+
| 0-1              | 446              | 0          | test         |
+------------------+------------------+------------+--------------+
test: stf >
```

4.4 Conversione

Per convertire i file di nProbe nel formato utilizzato da Argus si è fatto per prima cosa uno studio sui campi degli header per individuarne le differenze. Alcuni campi sono presenti in entrambi i formati seppure con nome diverso, altri sono stati ottenuti come combinazione, mentre quelli che nProbe utilizza in più rispetto ad Argus sono stati scartati.

Di seguito è rappresentata una tabella che mostra sulla colonna di sinistra i campi dell'header di Argus, mentre sulla destra la scelta dei campi di nProbe per rappresentare i dati nello stesso formato scelto da Argus.

Tabella 4.1: Tabella di conversione

binetflow	flow
start time	first switched
duration	last switched - first switched
protocol	protocol
source address	ipv4 source address
source port	source port
direction	biflow direction
destination address	ipv4 destination address
destination port	destination port
state	-
source tos	source tos
destination tos	-
tot packets	input packets + output packets
tot bytes	input bytes + output bytes
source bytes	input bytes
source data	-
destination data	-
label	-

nProbe e Argus, oltre le differenze degli header presentano anche differenze nel formato con cui sono scritti i file.

4.5 Automatizzazione conversione

La conversione è stata automatizzata con la scrittura di uno script in Python. Si è scelto di scrivere un programma usando questo linguaggio per la facilità di utilizzo nel lavorare con i file e per le performance.

Il problema richiede lo sviluppo di un programma che converte file in modalità batch. I file hanno una struttura gerarchica per data organizzata per sottocartelle, si esegue quindi un ciclo *for* che prende tutti i file in modo ricorsivo.

Dopodichè si apre il file in lettura leggendo una riga per volta, si converte il contenuto e lo si salva in una lista. Una volta terminata la lettura del file si scrive la lista contente i dati convertiti su un nuovo file.

Si è scelto di leggere il file una riga alla volta perchè le grandi dimensioni dei file non permettono un approccio diverso. Un approccio più veloce sarebbe stato quello di leggere i file per intero nella memoria principale ma non è possibile per le grandi dimensioni dei file.

Pseudocodice del programma

Algorithm 1 Single core version

```
1: procedure HYDRA
2:     for all file in path do
3:         read data from file
4:         convert data into new format
5:         append data into new file
```

4.6 Rendere efficiente la conversione

Nel programma descritto in precendenza viene generato un solo file di output in cui vengono convertiti tutti i file dati in input. Questa soluzione è comoda perchè da migliaia di file si ha un solo file con i dati convertiti, ma presenta il problema di creare un file con dimensioni enormi e di difficile gestione (il file può

raggiungere dimensioni tali da rendere difficile anche solo aprirlo in lettura) su cui crearci i modelli comportamentali.

Il programma è inoltre inefficiente poichè è single core e ha come collo di bottiglia la scrittura su un unico file.

La soluzione proposta seppure sia teoricamente corretta non può avere un'applicazione nel mondo reale. Bisogna cambiare quindi strategia per rendere la conversione più veloce sfruttando le macchine multi core e per avere in output file di dimensioni accettabili.

4.6.1 Possibili soluzioni

Si possono pensare diverse soluzioni che migliorerebbero in modo significativo il programma visto in precedenza.

- Lavorare su chunk di file
- Meccanismi di lock
- Scrittura su file 1:1

Lavorare su chunk di file Per velocizzare il programma e sfruttare i processori disponibili si potrebbe assegnare ad ogni processore un file da leggere e convertire. Quando il processore termina la conversione dei dati scrive sul file in output. In questo modo si dividrebbe il tempo di esecuzione sul numero di processori disponibili. Questa soluzione rende il più veloce possibile la conversione dei file ma i processori finiscono per scrivere sullo stesso file senza avere nessuna regola di precedenza, questo crea problemi perchè le scritture in output non sono ordinate e non è possibile creare modelli comportamentali affidabili. Le scritture su file devono essere ordinate e sequenziali. Inoltre questa soluzione ha il problema di scrivere sempre su unico file e come detto in precedenza questo tipo di soluzione non è possibile.

Meccanismi di lock un modo per risolvere i problemi precedenti è quello di utilizzare il concetto di semaforo.

In informatica un semaforo è un tipo di dato astratto gestito da un sistema operativo multitasking per sincronizzare l'accesso a risorse condivise tra processi. È composto da una variabile intera e da una coda di processi. Quando un processo apre il file per scriverci viene impostato un semaforo che segnala che la risorsa è occupata, se un altro processore prova ad aprire lo stesso file per scriverci gli sarà negato l'accesso dal semaforo fino a quando l'altro processo non rilascerà la risorsa.

In questo modo si risolve il problema delle scritture ordinate, ma c'è da tener conto che i semafori riducono la velocità di esecuzione dell'algoritmo poichè mentre un processore occupa la risorsa tutti gli altri processori devono mettersi in coda per aspettarne il rilascio. Questa soluzione sebbene rallenti l'esecuzione del programma rispetto alla soluzione precedente è comunque molto più veloce della versione single core perchè si guadagna comunque molto tempo nella lettura e conversione dei dati che viene effettuata alla massima velocità possibile. Questa soluzione risolve anche il problema dell'ordinamento delle scritture.

Rimane il problema della scrittura su un unico file che però può essere risolto facilmente decidendo di scrivere su un nuovo file quando raggiunge una dimensione specificata.

Se si implementa la divisione del file di output questa soluzione può considerarsi efficiente anche se rimane il collo di bottiglia introdotto dai semafori che costringe i processi ad aspettare in coda quando una risorsa è impegnata.

Scrittura su file 1:1 in questa soluzione ogni processore apre un file, lo legge. lo converte e scrive la conversione su un proprio file. Questa soluzione è decisamente la più semplice tra quelle proposte ed è anche la più efficiente poichè i processori non entrano mai in conflitto tra di loro cosicchè da effettuare letture, conversioni e scritture alla massima velocità possibile dalla macchina. Un problema che può creare questa soluzione è la produzione di una grande quantità di file in output. Si pensi che una sola settimana di traffico di rete, sono circa 10 mila file.

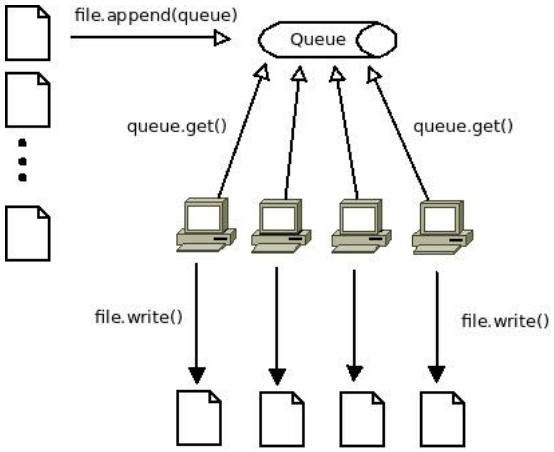
4.6.2 Scelta effettuata

Si è scelto di procedere con l'ultima soluzione presentata perchè data la grande quantità di informazioni da convertire si preferisce l'approccio più veloce a discapito dell'ordinamento finale.

Pseudocodice del programma multi core

Algorithm 2 Multi core version

```
1: procedure HYDRA
2:     for all file in path do
3:         Queue[] ← file
4:         read data from file
5:         spawn 4 process
6:         while Queue[] not empty do
7:             filename ←Queue.get()
8:             convert data into new format
9:             append data into new file
```



Il programma, dato in input la root directory dei file, cerca ricorsivamente tutti i file e li inserisce in una coda, chiama poi una funzione a cui assegna 4 processori. La funzione ha un ciclo while che si ripete fino a quando la coda popolata da tutti i file non è vuota. Ogni processore che entra in questa funzione toglie un file dalla coda e lo elabora.

Capitolo 5

Esperimenti e risultati

5.1 benchmark single core

Sono stati effettuati 10 test in entrambe le modalità. I risultati vengono riportati a seguire. tutti i benchmark sono stati effettuati con */usr/bin/time*

1. 2:28:03 98%CPU
2. 2:27:98 98%CPU
3. 2:28:86 98%CPU
4. 2:28:97 95%CPU
5. 2:24:52 99%CPU
6. 2:34:21 94%CPU
7. 2:28:36 98%CPU
8. 2:35:02 93%CPU
9. 2:25:59 99%CPU
10. 2:28:75 96%CPU

La media calcolata è quindi di **2:28:38 97%CPU**

5.2 Benchmark multi core

Sono stati effettuati 10 test in entrambe le modalità. I risultati vengono riportati a seguire. Tutti i benchmark sono stati effettuati con */usr/bin/time*

1. 0:54:82 265%CPU
2. 0:36:13 375%CPU
3. 0:38:24 367%CPU
4. 0:40:25 377%CPU
5. 0:40:14 368%CPU
6. 0:44:28 351%CPU
7. 0:41:67 368%CPU
8. 0:40:72 383%CPU
9. 0:40:83 381%CPU
10. 0:41:19 382%CPU

La media calcolata è quindi di **0:41:83 362%CPU**

Capitolo 6

Conclusioni

6.1 Prestazioni

Sia *T(p)* il tempo di esecuzione in secondi di un certo algoritmo su *p processori*. Di conseguenza sia *T(1)* il tempo di esecuzione del codice parallelo su 1 processore. La *misura di scalabilità* o *speedup* relativo di un algoritmo parallelo eseguito su p processori si calcola come:

$$S(p) = \frac{T(1)}{T(p)}$$

Con i risultati ottenuti si ha uno *speedup relativo* di

$$S(p) = \frac{T(148)}{T(40)} = \mathbf{3{,}7}$$

In un sistema ideale, in cui il carico di lavoro potrebbe essere perfettamente partizionato su p processori, lo speedup relativo dovrebbe essere uguale a p. In questo caso si parla di **speedup lineare**.

Si definisce *efficienza* il rapporto

$$E(p) = \frac{S(p)}{p}$$

Idealmente, se l'algoritmo avesse uno speedup lineare, si avrebbe $E(p) = 1$

Più l'efficienza si allontana da 1, peggio stiamo sfruttando le risorse di calcolo disponibili nel sistema parallelo.

$$E(p) = \frac{S(3{,}7)}{4} = \mathbf{0{,}925}$$

Bibliografia

[1] Anatomy of a botnet. https://web.archive.org/web/20170201233855/https://www.scribd.com/document/179124526/Anatomy-of-a-Botnet-WP-pdf. Accessed: 2017-02-01.

[2] Federal trade commission- consumer information, "malware". https://www.consumer.ftc.gov/articles/0011-malware. Accessed: November 2015.

[3] Malware definition. https://techterms.com/definition/malware. Accessed: 27-04-2016.

[4] Malware revolution: A change in target. https://docs.microsoft.com/en-us/previous-versions/tn-archive/cc512596(v=technet.10). Published: 05/20/2008.

[5] Sicurezza: Virus, worm, trojan... http://www.bloomriot.org/91/sicurezza-virus-worm-trojan.html.

[6] spyware. https://searchsecurity.techtarget.com/definition/spyware. Accessed: september 2016.

[7] Tojan horse definition. https://techterms.com/definition/trojanhorse.

[8] What is a trojan virus? - definition. https://usa.kaspersky.com/resource-center/threats/trojans#.VvD3eOLhDtQ.

[9] What is spyware? - definition. https://www.kaspersky.com/resource-center/threats/spyware.

[10] Stefan Axelsson. Intrusion detection systems: A survey and taxonomy. Technical report, University of Technology Goteborg Sweden, 14 March 2000.

[11] Herbert J. Mattord Michael E. Whitman. *Principles of Information Security*. Course Technology, 4 edition, 2011.

[12] Robert Newman. *Computer Security: Protecting Digital Resources*. 2009.

[13] David Dagon Wenke Lee, Cliff Wang. *Botnet Detection: Countering the Largest Security Threat (Advances in Information Security)*. Springer, softcover reprint of hardcover 1st ed. 2008 edition, 2010.