UNIVERSITÀ DI PADOVA  FACOLTÀ DI INGEGNERIA

CORSO DI LAUREA IN INGEGNERIA INFORMATICA
TESI DI LAUREA MAGISTRALE

INTRUSION DETECTION SYSTEM

A DECENTRALIZED APPROACH

Relatore: **Prof. Carlo Ferrari**

Laureando: **Andrea Busato**

Anno Accademico 2010-2011

. . . a mio Nonno

Indice

Capitolo 1

Introduzione

La complessità delle reti moderne e la loro grande diffusione, aprono scenari sulla gestione della sicurezza dei pc interconnessi a queste reti. Oggi più che mai, Internet ha pervaso ogni aspetto della nostra vita: ci consente di lavorare, di tenerci in contatto con amici e parenti, disporre pagamenti, e molto altro ancora. Gli stessi telefoni cellulari si sono trasformati da semplici strumenti per chiamare e mandare messaggi a veri e propri PC portatili delle dimensioni di un palmo di mano, connessi ad Internet.

Questa breve panoramica ci fa capire come la varietà di cose che possiamo fare attraverso le reti, abbia aumentato le informazioni sensibili che in esse circolano. Basti pensare all'acquisto di un bene online o ad una semplice ricarica telefonica; queste operazioni oggi sono comuni e di "routine", ma richiedono informazioni personali come il numero della carta di credito, il numero di cellulare, ecc. Queste informazioni devono assolutamente essere protette da terzi che potrebbero usarle contro di noi, causandoci danni, non solo economici.

Contestualmente, negli ultimi anni si è assistito anche ad un vertiginoso aumento delle minacce provenienti dalle stesse reti, minacce identificate principalmente come malware. Questo fenomeno non è passato inoservato, ma anzi ha stimolato la nascita di nuovi sistemi per la prevenzione e protezione contro gli attacchi informatici. Ci si è resi conto, infatti, che uno dei metodi più efficaci per proteggersi era quello di "unire le forze". Ecco quindi che nascono i primi sistemi di protezione collaborativi, chiamati *Intrusion*

Detection System (IDS) basati principalmente su di un approccio gerarchico. Questo nuovo metodo introduce dei benifici, quali l'aumento delle macchine monitorate, oltre che la capacità e prontezza nell'individuazione ed analisi di nuovi attacchi. Questi benefici, sono però seguiti da aspetti negativi, primo fra tutti la stessa struttura gerarchica del sistema. Infatti il verificarsi di guasti ai livelli superiori comporta danni tanto più elevati quanto più alto è il livello del nodo compromesso, fino ad inibire completamente il sistema di rilevamento intrusioni quando il problema affligge il nodo radice. Inoltre, un ulteriore svantaggio di questa struttura gerarchica è che all'aumentare del numero di nodi gestiti, aumenta il traffico generato e si fa più pesante l'impossibilità di bilanciare il carico computazionale tra i nodi della rete.

Dalle problematiche evidenziate, nasce questa tesi con l'obiettivo di realizzare un sistema di IDS con un architettura di tipo peer-to-peer, sfruttando quelli che sono i benefici della collaborazione tra nodi. Come mostrerò nei capitoli successivi, con un architettura collaborativa paritaria quale è il peer-to-peer, siamo in grado di risolvere i problemi rilevati in precedenza, eliminando i punti critici del sistema, e introducendo inoltre un'elevata tolleranza ai guasti. Inoltre, grazie alla natura paritaria di questo sistema, è possibile utilizzare in modo più efficiente le risorse disponibili grazie ad un efficace bilanciamento del carico computazionale tra tutti i nodi aderenti al sistema, riducendo così anche le repliche delle informazioni in esso presenti.

Attraverso la realizzazione di un prototipo e la sua messa in opera, sarà possibile dimostrare, anche con dati sperimentali, come il sistema realizzi un'efficace bilanciamento delle risorse e, suo punto di forza, un'elevata tolleranza ai guasti.

Capitolo 2

Rilevazione e prevenzione del malware

In questo capitolo verrà presentata una panoramica sulle principali minacce provenienti dalla rete, con particolare attenzione per il malware. Successivamente verranno illustrate due delle principali armi di difesa a livello di Network, più precisamente gli Honeypots e gli Intrusion Detection System, con relative varianti.

2.1 Malware e minacce dalla rete

Il termine malware deriva dalla contrazione di due termini inglesi, rispettivamente MALicious e softWARE, e viene utilizzato per indicare tutti quei programmi realizzati per danneggiare le macchine che li eseguono, da qui il nome di software malevolo. I tre obiettivi alla base di tutti i programmi all'interno di questa categoria sono:

- Installarsi sul dispositivo (es. computer o smartphone) al verificarsi di un certo evento. Per avere una maggior probabilità di successo è necessario che questi programmi abbiano la più alta compatibilità possibile con le piattaforme esistenti.
- Nascondersi dall'utente, in modo da poter sopravvivere il più a lungo possibile. Per fare questo vengono utilizzate spesso tecniche di mascheramento molto sofisticate che li rendono praticamente invisibili.

- Propagarsi il più possibile aumentando in questo modo il numero di successi. A tal fine si sono da sempre appoggiati ai principali mezzi di comunicazione informatici e se in origine tali mezzi si riconducevano principalmente ai floppy disk, ora il ventaglio è ben più ampio e partendo da email e pagine web modificate arriva alle sempre più diffuse piattaforme di file sharing o di instant messaging.

Prendendo in considerazione le differenti caratteristiche di questi programmi è possibile classificarli come segue:

Virus Si tratta sicuramente della classe più conosciuta. Una volta lanciato in esecuzione, il programma è in grado di replicarsi infettando altri file. Uno dei metodi è quello di copiare il proprio codice all'interno del file eseguibile di un'altra applicazione, questo è possibile inserendosi o all'inizio, col rischio però di sovrascrivere l'header, o in coda in modo da essere eseguito dopo l'esecuzione del programma originale. Altri file particolarmente vulnerabili sono i documenti di applicazioni che permettono di utilizzare linguaggi di programmazione più o meno complessi per la realizzazione di alcune operazioni (Macro). La loro diffusione avviene prevalentemente per mezzo di supporti fisici (floppy disk prima, chiavette USB oggi), email e programmi di file sharing, ma finché non vengono caricati in memoria non arrecano alcun danno. Per sopravvivere tendono a replicarsi in zone nascoste del sistema, alcuni arrivano fino al boot sector dell'hard disk, e cercano di disabilitare eventuali software antivirus attivi su quel sistema. Per favorire la prolificazione invece, sono nati i virus polimorfici e metamorfici. Queste nuove generazioni sono in grado di criptare il proprio codice binario all'atto della replicazione in modo da sfuggire ai sistemi antivirus basati su firme e mentre i primi mantengono comunque una prima parte in chiaro (la routine di decriptazione) i secondi sono in grado di mutare completamente il proprio codice arrivando anche a scomporlo in più parti da inserire in zone separate del file infettato.

Worm A differenza dei virus, i worm non hanno bisogno di infettare altri file per propagarsi, poiché sono in grado di modificare il sistema operativo che li ospita in modo da venire eseguiti ogni volta che la mac-

china si avvia e rimanere attivi fino al suo spegnimento o fino ad un intervento esterno che termini il relativo processo. Spesso questi programmi non arrecano danni direttamente, il loro scopo infatti è passare inosservati, ma possono avere effetti collaterali quali consumo di risorse e introduzione di ulteriore malware. I principali mezzi di trasmissione sono email, file sharing e sfruttamento dei bug di software diffusi. Per diffondersi tramite email possono integrare al loro interno tutto il necessario per generarle autonomamente (con un motore SMTP) oltre ovviamente ad implementare i metodi di difesa dagli antivirus visti in precedenza: disabilitazione delle suite si sicurezza, polimorfismo e metamorfismo. Inoltre procedendo autonomamente possono portare ad un notevole traffico di email che spesso riportano un indirizzo del mittente falsificato con la conseguenza che i provvedimenti presi dai vari server di posta, all'atto della rilevazione del worm, risultano inefficaci se non addirittura controproducenti (es. inserimento in black-list di indirizzi estranei ai fatti). Un'alternativa molto comune e anche più difficile da contrastare è la diffusione mediante lo sfruttamento di bug, spesso noti, di programmi molto diffusi. In questi casi l'utente non ha nessuna possibilità di controllo poiché il meccanismo è completamente automatico e l'unica soluzione possibile è l'aggiornamento costante sia del sistema operativo che dei software installati. Di notevole rilievo è il ruolo di questa classe di malware nella proliferazione di botnet, ovvero di reti di computer infetti alle dipendenze di gruppi criminali e alla base di alcune categorie di attacchi informatici.

Trojan Diminutivo di Trojan Horse ovvero Cavallo di troia. Il nome deriva dalla somiglianza del suo funzionamento con quello dell'antico dono che permise ai greci di espugnare Troia. Questo tipo di malware infatti si presenta come un programma, solitamente gratuito, che mette a disposizione dell'utente funzioni più o meno utili invogliandolo ad eseguire il codice, celando però le vere finalità. L'attrazione delle funzionalità offerte ha un ruolo molto importante, questi programmi infatti spesso non sono in grado di auto replicarsi, di conseguenza le uniche vie per una larga diffusione sono l'inserimento del codice all'interno di worm o con-

vincere il maggior numero di persone a scaricare il programma dal web. Questo malware solitamente è costituito da due file, quello che viene diffuso è il file server, mentre quello che rimane in mano all'attaccante è il file client che permette di comandare a distanza le macchine vittime del file server. Le principali finalità sono la creazione di backdoor o l'installazione di keylogger, con il fine ultimo di carpire informazioni sensibili.

Backdoor Letteralmente porta sul retro o più correttamente porta di servizio, permettono di aggirare, completamente o in parte, le procedure di sicurezza del sistema su cui sono installate. In alcuni casi possono essere già presenti nei programmi originali o per dimenticanza (spesso vengono utilizzate in fase sviluppo) o volutamente (in forma non sempre legale), ma spesso vengono installate da altri malware, come virus, worm o trojan, per garantire un accesso da remoto all'attaccante che ne conosca l'indirizzo IP.

Rootkit Il nome deriva dall'utente amministratore di sistemi Unix, chiamato root, e può assumere due significati. Il primo, e anche il più appropriato, è quello di un pacchetto di funzioni che permettano di avere il completo controllo sulla macchina su cui viene installato, il secondo viene utilizzato invece per indicare altre tipologie di malware in grado di usare avanzate tecniche di mascheramento. Entrambi i casi hanno in comune l'utilizzo di moduli del kernel, librerie o driver di sistema che rendono molto diffcile sia l'individuazione che la rimozione. La loro tecnica è semplice quanto efficace, ogni software installato su un sistema, compresi gli antivirus, basano il proprio funzionamento sulle chiamate a funzioni messe a disposizione dal sistema operativo ospitante. Il rootkit interviene modificando o sostituendo i moduli che mettono a disposizione le suddette funzioni, in questo modo possono assicurarsi che la loro presenza non venga rivelata e bloccare eventuali chiamate che volessero porre termine alla loro esistenza.

Keylogger Sono strumenti il cui scopo è quello di registrare tutto ciò che un utente digita sulla tastiera del proprio computer con la speranza di

intercettare informazioni preziose come password e numeri di carte di credito. Possono essere sia hardware che software, ma ovviamente solo i secondi rientrano nella categoria dei malware. Per svolgere il loro compito usano alcune tecniche tipiche dei rootkit, come la modifica dei driver delle tastiere o la modifica delle specifiche librerie del S.O. e i più avanzati permettono di registrare anche alcuni screenshot dello schermo in modo da agevolare l'operato dell'attaccante. Nonostante siano di difficile rilevazione esistono due tecniche di difesa. La prima è quella di installare plugin in alcuni dei software più critici (come i browser) che vanno ad installare un secondo driver in grado di creare un collegamento sicuro tra tastiera e browser mediante l'utilizzo di comunicazioni crittografate. La seconda tecnica, che si sta diffondedo alcune suite di sicurezza, è quella di utilizzare una tastiera virtuale presentata a schermo e utilizzabile mediante il mouse, in questo modo non vengono premuti tasti della tastiera fisica e il keylogger non registrerà alcuna attività.

Spyware Esistono due categorie identificabili con questo termine che deriva dall'unione dei termini SPY softWARE. La prima, e probabilmente più innocua, raccoglie informazioni riguardanti l'attività online di un utente (siti web visitati, iscrizioni a servizi, acquisti etc) senza il suo consenso, trasmettendole tramite Internet ad un'organizzazione che le utilizzerà per trarne profitto, solitamente attraverso l'invio di pubblicità mirata. La seconda, e più pericolosa, oltre alle consuetudini dell'utente, tiene traccia delle informazioni inserite nei diversi moduli come password e numeri di carte di credito. Quest'ultima categoria è possibile trovarla accoppiata ai keylogger descritti sopra, ma non si tratta di un vincolo. In entrambi i casi si tratta di software non in grado di propagarsi autonomamente e per installarsi richiedono l'intervento dell'utente, quindi un funzionamento simile a quello dei trojan, con la differenza che in alcuni casi vengono aggiunti appositamente dalle softwarehouse per ottenere un ritorno economico. In questi casi è possibile che l'utente venga informato in modo più o meno trasparente prima di procedere con l'installazione, col vincolo che nel caso rifiuti questa componente non è possibile proseguire con l'installazione del resto del programma. Dove

esiste quest'informazione non è più possibile parlare di malware quanto piuttosto di Adware.

Adware Il termine deriva dalla contrazione inglese di ADvertising-supported softWARE, che può essere tradotto come Software supportato dalla pubblicità, ed indica quei programmi che durante il loro utilizzo mostrano avvisi pubblicitari permettendone la distribuzione gratuita o comunque a prezzi ridotti. Successivamente l'utente potrà poi decidere di rimuovere tali avvisi mediante il pagamento di una licenza. Di per se questi software non rientrerebbero nella categoria dei malware se non fosse che spesso implementano alcune caratteristiche tipiche degli spyware per poter presentare all'utente avvisi pubblicitari focalizzati sui suoi interessi.

Hijacker Sono software in grado di modificare la homepage dei browser con lo scopo di indirizzarli su pagine contenenti altri tipi di malware in grado di diffondersi mediante i bug dei browser stessi o dei plugin in essi installati.

Ransomware Si tratta di una classe di malware abbastanza recente e che può essere suddivisa ancor più nello specifico in cryptovirus, cryptotrojan or cryptoworm. Come si può intuire dai nomi si tratta di particolari versioni di virus, trojan e worm in grado di criptare il contenuto dei più diffusi tipi di file contenuti sull'hard disk dell'utente. Lo scopo è quello di poter successivamente ricattare la vittima chiedendo un riscatto in cambio della password per decriptare i documenti. L'utente sprovvisto di copie di backup dei file più importanti che intendesse pagare come da richiesta, si troverebbe comunque nella spiacevole condizione tipica di tutti i casi di ricatto, quella cioè di non avere nessuna garanzia che la disavventura termini dopo il pagamento della somma pattuita.

Dialer Questa categoria sta lentamente scomparendo grazie all'avvento delle linee xDSL e fibra ottica. In origine il loro unico scopo era quello di creare connessioni telefoniche per potersi connettere ad internet, quindi un atteggiamento lecito, ma in seguito cominciano a diffondersi versioni programmate per contattare numeri di particolari servizi telefonici

caratterizzati da costi molto elevati e distinguibili dal loro prefisso, in origine 144, poi 166 e infine 899 e 892. Grazie all'utilizzo di modem ADSL, il computer non ha più la possibilità di effettuare tali tipi di connessioni, di conseguenza il bacino di potenziali vittime si sta riducendo rapidamente.

Rabbit Anche questa classe di malware sta perdendo popolarità, ma a differenza dei dialer non per particolari innovazioni tecnologiche, ma a causa di un cambiamento radicale nelle motivazioni che portano alla diffusione di malware. Oggi infatti la stragrande maggioranza di programmi malevoli in circolazione ha lo scopo di far ottenere un guadagno alle organizzazioni criminali, mentre i Rabbit hanno il solo scopo di riprodursi in continuazione fino a saturare le risorse del sistema e a differenza dei Virus non infettano alcun file.

Analizzato quindi quelle che sono le principali tipologie di malware oggi conosciute, vediamo quali possono essere i lor principali impieghi nell'ambito di attacchi più complessi.

Man in the middle In questo caso l'attaccante si intromette nella comunicazione tra due terminali intercettandone il contenuto. In base alle caratteristiche della comunicazione quest'attività può essere più o meno complessa, ma la procedura standard prevede che l'attaccante faccia da intermediario nella comunicazione tra i due terminali senza che quest'ultimi se ne accorgano e per fare questo deve riuscire ad ingannare entrambi spacciandosi per l'utente all'altro terminale grazie a credenziali false. In questo caso un malware potrebbe servire per reindirizzare le connessioni verso la macchina dell'attaccante senza che l'utente vittima se ne possa accorgere.

Denial of Service (DoS) Si traduce in Negazione di Servizio e si può dividere in due categorie: sfruttamento di bug e esaurimento delle risorse. Nella prima categorie rientrano diverse tecniche che si basano su bug software, protocolli insicuri o cattiva implementazione. Ultimamente però si tende ad utilizzare il termine DoS per indicare la seconda categoria, cioè quella in cui, lavorando su uno dei parametri d'ingresso, si

cerca di portare il funzionamento di un sistema informatico che fornisce un servizio, ad esempio un sito web, al limite delle prestazioni fino a renderlo non più in grado di erogare il servizio. Solitamente i due parametri su cui si lavora sono la saturazione della banda o il riempimento delle code di gestione delle richieste. L'aumento esponenziale del traffico sulla rete Internet, ha portato ad un proporzionale aumento della banda disponibile, soprattutto lato server, visto che la maggior parte dei collegamenti domestici sono di tipo asimmetrico (Assymmetric-DSL), cioè con una limitata banda in upload. Questa evoluzione ha reso irrealizzabili attacchi DoS lanciati da singoli terminali o da piccoli gruppi di utenti. Nascono così gli attacchi DDoS.

Distribuited Denial of Service (DDoS) Come si intuisce dal nome, si tratta dello stesso tipo di attacco sopra definito, ma realizzato in modo distribuito. Esistono alcuni attacchi DDoS basati su vulnerabilità dei protocolli, ma solitamente per realizzare questo tipo di attacco è prima necessario procurarsi un vasto bacino di zombie, termine usato per indicare macchine infette da malware e sotto il controllo degli attaccanti. Come visto in precedenza esistono vari tipi di malware, in questi casi si fa spesso riferimento a worm che, grazie alla loro capacità di diffondersi molto rapidamente, sono incaricati di aprire backdoor e creare così una botnet, cioè una rete di zombie pronti ad eseguire i comandi che gli verranno impartiti dai server C&C (Command & Control). Spesso quest'ultimi sono dei semplici server IRC a cui i malware si andranno a connettere una volta installati e che possono essere gestiti direttamente dai diffusori del worm o indirettamente da bande criminali che ne facessero richiesta.

Distribuited Reflection Denial of Service (DRDoS) Rappresenta l'evoluzione degli attacchi DDoS. Gli effetti finali sono gli stessi, ma per rendere più difficoltoso il rintracciamento delle macchine attaccanti (che potrebbero quindi essere bloccate), ci si appoggia a server che mettono a disposizione servizi pubblici. Lo scopo è di far riflettere i propri attacchi da questi server, da qui il nome Reflection, utilizzando la tecnica dello spooffing. In poche parole l'attaccante, o gli zombie a sua

disposizione, mandano pacchetti TCP ai server pubblici utilizzando come indirizzo sorgente spoofato (contraffatto) quello della vittima che si vuol colpire, in questo modo i server pubblici risponderanno a quest'ultima che si ritroverà improvvisamente sommersa di richieste. Inoltre per evitare di sovraccaricare i reflection server, l'attaccante utilizzerà quest'ultimi a turno in modo da non superare eventuali livelli di soglia che porterebbero a notificare l'evento ai rispettivi amministratori.

Antisocial network Questo termine rappresenta semplicemente una provocazione ed è stato coniato da alcuni ricercatori che hanno messo in evidenza come la notevole diffusione dei social network possa essere un potenziale rischio per la sicurezza. Il titolo della ricerca è eloquente: "Antisocial Networks: Turning a Social Network into a Botnet", la ricerca infatti propone un punto di vista critico su queste nuove piattaforme già note per la loro vulnerabilità a codice maligno. Cosa accadrebbe infatti, se una delle applicazioni più diffuse, inserite dagli utenti nelle proprie pagine, fosse in realtà studiata per sovraccaricare di richieste i server della vittima prescelta? Una risposta la danno i ricercatori stessi che pubblicano i risultati di un test da loro effettuato. Dopo aver scelto facebook.com come piattaforma di partenza, hanno messo a disposizione un programma che prometteva di visualizzare la foto del giorno del National Geographic sulla pagine personale dell'utente che l'avrebbe installato. Nella realt à il software nascondeva altre 4 richieste che andavano a prelevare un'immagine da 600KB presso un host vittima senza che l'utente se ne accorgesse. Nel caso in questione, l'host vittima era una macchina controllata dai ricercatori che nell'arco di pochi giorni hanno visto crescere il traffico generato dalle mille installazioni fino a raggiungere il picco di 300 richieste/ora e di 6Mbit al secondo di banda. Se a prima vista questi numeri potrebbero non sembrare così impressionanti, si sottolinea come il programma sia stato rimosso dopo pochi giorni e come le richieste fossero limitate a 4, mentre un ipotetico malware potrebbe generare centinaia di richieste e rimanere online molto più tempo con conseguenze devastanti, alcuni degli applicativi più diffusi di Facebook oltrepassano infatti il milione di utenti giornalieri.

Phishing Questo tipo di attacchi, non si basa direttamente sulla distribuzione di malware, ma sullo Spam che questo può generare. L'interesse crescente per questa categoria è legato alla sua diffusione in continua crescita. Lo scopo è quello di ottenere i dati relativi ai conti correnti, sia bancari che postali, delle vittime. Per raggiungerlo, viene inizialmente creata una pagina web che rispecchi fedelmente la pagina di login della banca scelta come esca e che sia in grado di registrare i dati inseriti dagli utenti, dopo di ché si pubblica ad un indirizzo che contenga alcuni nomi che richiamino l'url della pagina originale, anche se ovviamente il dominio sarà diverso. A questo punto parte una campagna di Spam focalizzata il più possibile sull'area geografica interessata all'ente utilizzato come esca. Il termine Spam sta ad indicare email indesiderate, spesso contenenti pubblicità, ma che nel caso del phishing contengono messaggi appositamente studiati per allarmare l'utente invitandolo poi a risolvere eventuali problemi dopo aver effettuato il login. Dopo questi avvisi viene riportato il link alla pagina web creata precedentemente, mascherato da sotto il titolo dell'url originale. L'utente sprovveduto che dovesse cadere nella trappola si ritroverebbe con le credenziali rubate e ben presto col contocorrente svuotato. In questi casi è facile difendersi, innanzitutto è sempre bene diffidare di certi tipi di email, spesso la lingua utilizzata presenta diversi errori grammaticali (soprattutto in Italia), poi è buona abitudine diffidare dei link proposti, quindi aprire un nuova finestra dove poter digitare l'indirizzo che già si conosce.

Pharming Molto simile al phishing con cui condivide le finalità e anche tutta la parte di preparazione. Fino alla realizzazione della pagine web contraffatta infatti, le tue tecniche di attacco sono identiche, la differenza sta nel come si attirano le potenziali vittime. A differenza del phishing, nel pharming non si mandano email di spam, ma si attaccano i server DNS sparsi per la rete. Questi server sono di fondamentale importanza e servono per fornire ai computer la traduzione delle url in indirizzi IP. L'attacco a questi server è molto più complesso rispetto all'inoltro di spam e ha come scopo sostituire l'indirizzo IP originale della banca con quello dei propri server. A questo punto l'utente che tentasse di connet-

tersi per esigenze personali (e non più su richieste via email) verrebbe indirizzato sulla pagina contraffatta senza accorgersene. Questi attacchi sono di difficile realizzazione, ma se portati a termine mieteranno vittime anche tra gli utenti più esperti, questo perché non esiste modo di rendersi conto di quello che sta accadendo, salvo quello di far precedere l'autenticazione da un'approfondita navigazione, in questo caso infatti potrebbero uscire allo scoperto lacune nella contraffazione del sito originale.

2.2 Honeypot

Gli honeypot sono strumenti preziosissimi per la raccolta del malware ed il tracciamento delle attività malevole svolte da eventuali aggressori. La base comune di queste applicazioni è quella di mettere a disposizione dell'utente uno strumento sicuro ed isolato da utilizzare come trappola in cui circoscrivere le attività degli aggressori. Proprio dallo scopo che si prefigge nasce il suo nome, che dalla lingua inglese può essere tradotto letteralmente come "vasetto del miele", usato anche in ambiti non informatici, come quello poliziesco ad esempio, dove viene associato a tutti gli stratagemmi messi in atto per cogliere i malviventi in flagranza di reato. Inoltre uno dei principali vantaggi derivante dalla struttura e dall'implementazione degli honeypot è che qualsiasi attività registrata può essere considerata a priori come ostile, riducendo al minimo, o perfino azzerando, il numero di falsi positivi.

Dopo questa introduzione si può procedere con la loro classificazione. Il primo passo è quello di distinguerli in base al deployment (spiegamento) ottenendo così gli honeypot di produzione e quelli di ricerca.

- Gli honeypots di produzione sono più semplici da usare, ma hanno anche una limitata capacità di raccolta di informazioni. Vengono impiegati principalmente presso aziende e grandi compagnie che li dispongono nella propria rete affianco ai propri server. L'obiettivo è quello di aumentare la sicurezza della rete aziendale, cercando di rallentare l'operato degli aggressori. A tale scopo vengono utilizzati honeypot a bassa interazione su cui reindirizzare i tentativi di attacco, che nonostante

le limitate informazioni raccolte permettono di raggiungere l'obiettivo prefissato.

- Gli honeypots di ricerca vengono impiegati principalmente da istituzioni accademiche, enti no-profit ed enti governativi. Si tratta di sistemi complessi, che richiedono notevoli conoscenze tecniche sia per l'installazione che per la manutenzione, ma che permettono di raccogliere informazioni molto dettagliate su tutte le operazioni eseguite al loro interno. Lo scopo è quello di poter identificare nuove tecniche d'attacco, mediante l'analisi dei dati raccolti per poi studiare e divulgare le relative contromisure atte a garantire il maggior livello di sicurezza possibile.

L'utilizzo di questo strumento ha diversi vantaggi.

- L'honeypot non eroga servizi pubblici, di conseguenza ogni connessione ricevuta può essere considerata malevola senza il rischio di falsi positivi. Un utente comune, infatti, non può connettersi per sbaglio, perché non esistono informazioni pubbliche che riconducano all'indirizzo dell'honeypot, quindi chi riesce a contattare tale servizio, deve averle ottenute grazie ad una qualche tecnica di attacco.
- Considerando che, come appena detto, ogni connessione è malevola, se ne deduce che anche ogni attività svolta all'interno dell'honeypot risulta tale. Questo risultato è molto importante, poiché semplifica notevolmentele operazioni di analisi. Ogni modifica apportata al sistema infatti, è parte di un tentativo di attacco e come tale dev'essere presa in considerazione senza il rischio di appesantire il lavoro da svolgere con informazioni superflue.
- Altro vantaggio importante è che le attività svolte vengono registrate indipendentemente dalle tecniche di offuscamento adottate dall'attaccante per raggiungere l'honeypot. Quest'ultimo infatti rappresenta il punto finale, di conseguenza le operazioni si svolgeranno in chiaro, sia che le connessioni utilizzate siano crittografate (tecnica usata per bypassare i controlli effettuati sul traffico che attraversa la rete locale), sia che provengano da sorgenti multiple (tecnica usata dall'attaccante per operare nell'anonimato).

- Inoltre, nel caso si utilizzino honeypot a bassa interazione, a questi vantaggi si aggiunge la relativa economicità del deployment di questi strumenti. Oltre alla possibilità di appoggiarsi a piattaforme open source senza dover rinunciare alle funzioni fondamentali, si hanno costi ridotti anche per quanto riguarda l'hardware da utilizzare. Un moderno computer è infatti sufficiente per gestire migliaia di indirizzi IP e per implementare svariati honeypots, mentre per singole installazioni ci si può appoggiare anche ad hardware più datato.

Tutti questi vantaggi sono accompagnati da due svantaggi che devono essere tenuti in considerazione all'atto dell'installazione.

1. L'honeypot è in grado di registrare e tenere traccia esclusivamente del traffico ad esso indirizzato. Di conseguenza attacchi rivolti ad altri non verrebbero rilevati. Per questo vanno ben studiati gli indirizzi da associare all'honeypot e soprattutto non bisogna sciarlo "solo" sperando che basti a limitare i danni, ma affiancarlo ad altri strumenti di rilevazione ed analisi.

2. Essendo vulnerabile di sua natura, bisogna prestare attenzione all'eventualità che un aggressore si renda conto di essere finito in un honeypot. In questo caso infatti potrebbe cercare di prendere il controllo della macchina per lanciare ulteriori attacchi agendo stavolta dall'interno della rete.

Per questi due motivi occorre studiare attentamente la disposizione di questi strumenti cercando di mantenerli isolati dalle altre macchine connesse alla rete aziendale, mantenendoli però accessibili in modo da poter dirottare il traffico su di essi. Nella classificazione precedente si è fatto riferimento anche al diverso livello di interazione degli honeypot di produzione da quelli di ricerca e la distinzione fatta era tra alta e bassa.

Bassa interazione Si tratta di software installati su sistemi operativi standard e che spesso impiegano un singolo processo per simulare dal singolo computer alla complessa rete di host. Quest'ultimo è il caso di honeyd, un demone in grado di emulare l'esecuzione di svariati servizi su diversi host a cui vengono associati gli indirizzi IP disponibili all'interno della

rete reale. Questa possibilità permette di simulare un elevato numero di server tra i quali nascondere quelli reali, rendendo più difficili le attività di ricerca di un attaccante. Ha inoltre il vantaggio di poter essere personalizzato mediante script creati dall'utente che potrà in questo modo adattarlo alle proprie esigenze. Altri esempi di honeypots diffusi sono Nepenthes e mwcollect, entrambi in grado di simulare alcuni dei servizi più diffusi e di scaricare il contenuto di eventuali malware proposti dall'attaccante, che verranno poi archiviati per una futura analisi. I vantaggi principali di questa categoria di prodotti risiedono nella semplicità di utilizzo e di installazione (si tratta di semplici programmi), nonché nella bassa probabilità di compromissione delle macchine ospitanti.

Alta interazione Si tratta di sistemi molto complessi basati su computer dedicati a tale scopo o in alternativa su macchine virtuali, anche se questo rappresenta più un vantaggio economico che tecnico. La caratteristica di questa tipologia consiste nella raccolta di una notevole quantità di dati da cui estrapolare informazioni importanti per lo studio di nuove tecniche di attacco. La complessità di tali implementazioni, sia in termini di installazione che di mantenimento, risulta tale da limitare il loro utilizzo a personale esperto in grado di studiare piani di spiegamento che tengano conto dei rischi derivanti da possibili infezioni o dalla possibilità che uno degli host cada nelle mani di un attaccante. Inoltre questi sistemi risultano solitamente statici e di conseguenza identificabili dagli aggressori che scambiandosi informazioni realizzano vere e proprie blacklist. Uno dei progetti più importanti di questa categoria è Honeynet, che appoggiandosi su una grande rete di macchine dedicate è in grado di raccogliere e analizzare un'imponente mole di dati.

Oltre alla raccolta di malware ed al rallentamento degli attacchi subiti, l'utilizzo di honeypots risulta molto utile ai fini di combattere lo spam. Una delle tecniche principali degli spammers è infatti lo sfruttamento di open mail relay, cioè di server SMTP impostati in modo da permettere a chiunque l'invio di email, senza controllare che il mittente o il destinatario siano utenti registrati. Purtroppo questa è l'impostazione di default di molti server, con la

conseguenza che molti di essi vengono sfruttati appunto a fine di spam. La tecnologia honeypot può intervenire simulando uno di questi server, in questo modo nel caso si riesca ad ingannare lo spammer si otterranno due grandi benefici: primo, le email di spam non verranno inoltrate; secondo, sarà possibile risalire allo spammer, mediante le tracce da lui lasciate. Interessante è notare come all'introduzione di questi sistemi la maggior parte degli spammer si collegasse direttamente ai mail server senza alcun timore, mentre col tempo e la presa di coscienza del rischio di cadere in una trappola, il comportamento è cambiato e l'accesso avviene solo mediante l'utilizzo di una catena di sistemi precedentemente compromessi. Questo significa che l'introduzione degli honeypot in questo campo ha reso più difficoltose le operazioni degli spammer.

In conclusione questa tecnologia risulta utile al fine di rilevare e reagire agli attacchi provenienti dalla rete. Offre un ampio ventaglio di opzioni permettendo all'utente di scegliere quelle che meglio si adattano alle proprie esigenze, sia a livello di funzionalità offerte che di capacità richieste.

2.3 Intrusion Detection System

Gli Intrusion Detection System o IDS vengono utilizzati per rilevare accessi non autorizzati alle reti locali o ai computer ad esse collegate. Si tratta di dispositivi software o hardware (a volte la combinazione di tutti e due) che dislocati in punti strategici della rete permettono di fornire dettagli sugli attacchi sfuggiti al controllo del firewall. Le intrusioni rilevate possono essere di varia natura, partendo da quelle prodotte da cracker esperti si passa a quelle di utenti inesperti che utilizzano programmi semiautomatici, fino ad arrivare a tool automatici (vedi malware). Come indica il nome stesso, si tratta di sistemi in grado di segnalare intrusioni già avvenute o ancora in atto, ma non di prevenirle, per cercare di rendere più chiara la sua funzione, è possibile paragonare l'IDS ad un antifurto. Inoltre si possono utilizzare le informazioni raccolte sia per poter aggiornare la propria rete rendendola di volta in volta sempre più sicura, sia a scopo legale, nel caso si volessero intraprendere cause nei confronti degli intrusori.

Un IDS è composto da 4 componenti fondamentali:

- i sensori, utilizzati per ricevere le informazioni dalla reti o dai computer;
- un engine o motore, che analizza i dati prelevati dai sensori e provvede a individuare eventuali anomalie;
- un database, contenente le regole e le firme utilizzate dal engine per l'analisi.

Solitamente i componenti si trovano in un unico pacchetto, ma non è escluso trovarli separatamente, soprattutto nel caso dei sensori che possono essere caratterizzati da funzioni specifiche.

Una prima classificazione di questi strumenti si può fare in base all'obiettivo da monitorare ottenendo le seguenti categorie:

Host Intrusion Detection System (HIDS) consiste in un agente che analizza l'Host alla ricerca di intrusioni mediante l'analisi dei file di log del sistema, delle system call, delle modifiche al file system del computer (modifiche nel file delle password, nel database degli utenti e della gestione dei privilegi, ecc), e anche di altre componenti del computer. Come per gli honeypots, questa classe di IDS ha il vantaggio di essere installata sulla macchina vista dall'aggressore come end-point, di conseguenza le tecniche di offuscamento degli attacchi non hanno alcun effetto. Lo svantaggio è di avere una visione limitata del traffico rivolto alla singola macchina. Un esempio di questa tipologia è Ossec.

Network Intrusion Detection System (NIDS) ha l'obiettivo di analizzare il traffico di rete per identificare le intrusioni e per fare ciò tiene sotto monitoraggio non solo un singolo host ma una rete completa. Si tratta di un sistema che ispeziona (in gergo sniffa) il traffico che passa sul segmento di rete a cui sono connessi i suoi sensori, cercando tracce di attacchi. A tale scopo il dislocamento di quest'ultimi è di fondamentale importanza e dev'essere studiato attentamente per garantire una quantità di traffico adeguata. Questa categoria, anche se soggetta a tecniche di offuscamento degli attacchi, consente una visione generale dello stato della rete. Un esempio di Network Intrusion Detection System è Snort, un prodotto open source ed probabilmente il più diffuso.

Application Protocol Intrusion Detection System (APIDS) analizza il traffico concentrandosi su specifici protocolli applicativi, dei quali conosce le diverse implementazioni ed è in grado di notare eventuali anomalie.

Hybrid Intrusion Detection System Come indica il nome si tratta di un modello ibrido, ovvero implementa più tecniche cercando di sfruttare i vantaggi di ognuna. I modelli più diffusi si basano su HIDS e NIDS e offrono sia una panoramica sullo stato della rete che un'analisi più specifica per ogni singolo host. Un esempio di questa categoria di IDS è rappresentato da Prelude.

Esiste infine un ultima classificazione e viene fatta tra gli IDS passivi e quelli attivi:

passivi sono la versione standard, si limitano alla segnalazione degli eventi rilevati e per loro vale tutto quello scritto in precedenza.

attivi oltre alle funzioni di segnalazione, implementano metodi per intervenire sul sistema in modo da porre fine alle minacce rilevate. Gli interventi più diffusi riguardano la modifiche delle access list dei firewall, in modo da interrompere le connessioni con indirizzi esterni alla rete locale. Ulteriori miglioramenti hanno reso disponibile il blocco anche del traffico interno alla rete, potendo così interrompere anche la diffusione di malware tra host interni. Tutte queste funzionalità hanno portato alla creazione degli IPS (Intrusion Prevention System).

2.3.1 Intrusion Prevention System

Rappresentano l'evoluzione degli IDS e non esiste in realtà una linea di demarcazione netta nei confronti delle versioni attive di quest'ultimi. In alcuni casi si tende ad identificare col termine IDS quelli esclusivamente passivi, mentre con IPS tutti gli strumenti che permettono di intervenire attivamente, in altri si aggira la questione utilizzando il termine IDPS, Intrusion Detection and Prevention System. Hanno molte similitudini col funzionamento dei firewall: sono entrambi in line, quindi fail close, e agiscono entrambi mediante l'utilizzo di liste di controllo degli accessi. Gli IPS, però sono più spostati a livello applicativo e nelle liste citate tendono ad inserire coppie

utente-programma, piuttosto che indirizzo IP-porta. Esistono due categorie principali di questi strumenti.

Host IPS (HIPS) come si deduce dal nome si tratta di implementazioni a livello di singolo host. Il controllo può essere molto dettagliato e per ogni programma che dovrà essere eseguito sul sistema è possibile definire specifiche policy. Se non impostato in modo preciso, può risultare particolarmente invadente per l'utente dell'host.

Network IPS (NIPS) hanno una visione a livello di rete, quindi più ampia, e si dividono in tre principali sottocategorie.

- **Protocol analysis IPS (PIPS)** ispezionano il traffico a livello di protocolli applicativi, come HTTP e FTP, cercando anomalie o violazioni degli standard.
- **Content Based IPS (CBIPS)** analizzano il payload dei pacchetti utilizzando solitamente tecniche basate su signature analysis, con tutti i vantaggi e svantaggi tipici di tale approccio.
- **Rate Based IPS (RBIPS)** si basano principalmente su tecniche di anomaly detection. Esiste una fase iniziale di studio delle statistiche sul traffico relativo alla rete monitorata, in particolare si osservano le percentuali (rate) dei pacchetti divisi per protocolli (TCP, UDP, ARP, ICMP). Finito lo studio iniziale l'IPS entra in funzione affiancato da un sistema di autoapprendimento, in modo da garantire un miglioramento delle prestazioni. Questo approccio risulta particolarmente efficace nel rilevamento di attacchi DoS o DDoS.

La scelta tra un HIPS o un NIDS va effettuata in base alle proprie esigenze, ma occorre tenere in considerazione come la seconda opzione richieda maggiori risorse e risulti un punto critico del sistema (in gergo single point of failure) a causa del suo comportamento fail close citato in precedenza.

2.3.2 Distribuited Intrusion Detection System

Un IDS distribuito (dIDS) consiste nel deployment di più IDS su larga scala, i quali sono in grado di comunicare tra loro oppure con un'unità centrale.

Questa comunicazione facilita il monitoraggio avanzato della rete, l'analisi di incidenti e la raccolta istantanea di dati riguardanti gli attacchi. Inoltre la cooperazione permette agli amministratori di avere una visione più ampia delle proprie reti nel complesso. Un dIDS permette inoltre un'analisi più efficiente degli attacchi ricevuti, grazie alla raccolta centralizzata dei rispettivi dati, e la possibilità di rilevare eventuali trend a livello globale.

Gli elementi caratterizzanti dei dIDS sono tre.

Server centrale di analisi Come dice il nome stesso rappresenta l'unità centrale. Svolge le operazioni di analisi sui dati raccolti e memorizza le informazioni ottenute nel proprio database, che viene reso disponibile agli agenti distribuiti per eventuali interrogazioni. Lo scopo è quello di condividere le informazioni per migliorare i risultati ottenibili. Spesso viene affiancato da un server web che mette a disposizione degli utenti un interfaccia grafica user friendly.

Agenti cooperanti distribuiti Si tratta di IDS completi ed installati presso le reti che aderiscono al dIDS e il loro incarico è di raccogliere i dati dalle proprie reti per poi inoltrarli al server centrale per farli analizzare. Per ottenere i migliori risultati le reti devono esser distribuite geograficamente in modo da coprire la più vasta area possibile.

Aggregazione dei dati Non si tratta di un componente software o hardware, ma di un processo fondamentale per l'ottenimento degli obiettivi preposti. Questa operazione viene effettuata sul server centrale e ha lo scopo di aumentare il valore dei dati raccolti dai singoli agenti. Infatti è grazie all'aggregazione dei dati che è possibile rilevare la diffusione su larga scala delle minacce identificate a livello di singole reti. Si pensi alla segnalazione di un malware, rilevarlo 100 volte all'interno della stessa rete e rilevarlo una volta in 100 reti distinte ha significati ben differenti. Nel primo caso si tratta di un infezione locale, probabilmente un host compromesso ha infettato le macchine all'interno della propria. I danni risultano circoscritti alla rete in questione e l'intervento potrà solo salvare eventuali macchine ancora pulite. Nel secondo caso invece, esiste una nuova minaccia che ha già raggiunto diverse zone del mondo

e che se non bloccata tempestivamente rischia di portare a conseguenze devastanti. L'intervento dev'essere immediato.

Lo svantaggio nell'utilizzo di dIDS sta nella complessità di tali sistemi. Devono essere studiati attentamente per garantire la massima sicurezza, inoltre richiedono la collaborazione tra enti differenti, quindi la scelta di uno standard comune. D'altra parte i benefici sono notevoli e possono essere paragonati a quelli che si ottengono passando da un HIDS ad un NIDS. Inoltre una visione più ampia migliora i tempi di reazione alle nuove minacce e la gestione più efficiente permette anche il risparmio di risorse. Quest'ultimo punto aumenta d'importanza in considerazione del crescente numero di minacce da cui difendersi e di conseguenza del costante aumento di risorse richieste per difendersi appunto da tali minacce.

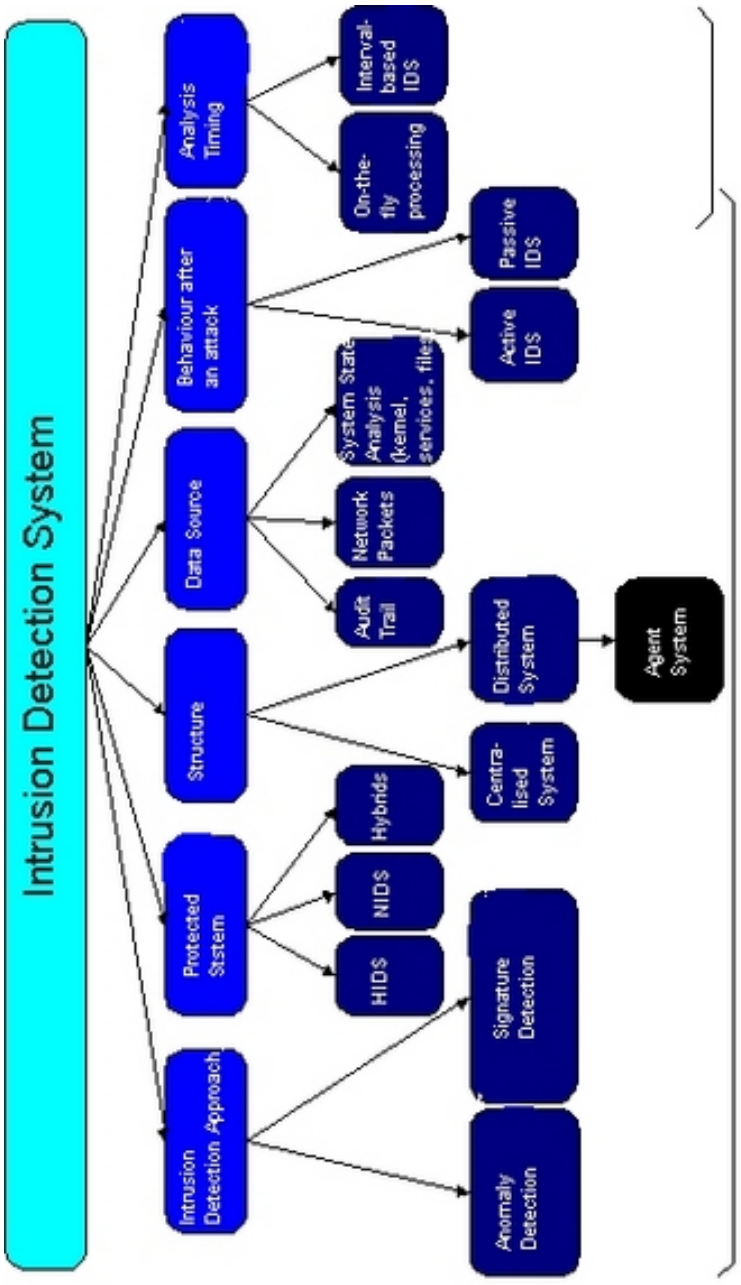

Figura 2.1: *Modello riassuntivo delle varie tipologie di IDS*

Capitolo 3

Modello Architetturale

In questo capitolo ci proponiamo di descrivere il modello architetturale alla base del progetto. Per far questo, iniziamo con una descrizione delle reti peer-to-peer e delle loro caratteristiche, per poi esporre il modello del sistema.

3.1 Reti Peer-To-Peer

Le reti peer-to-peer, indicate anche con la sigla p2p, sono costituite da nodi aventi diverse connessioni dirette verso gli altri partecipanti alla rete al contrario dei tipici sistemi centralizzati dove i collegamenti sono unicamente tra client e server. Per funzionare ogni nodo deve implementare sia le funzioni del server che quelle del client a cui ricorrerà di volta in volta a seconda della situazione. I vantaggi di questa tipologia di rete sono molti.

efficienza di rete L'utilizzo di nodi che svolgono funzioni sia di client che di server, permette una migliore gestione della banda disponibile, poiché si può contare su tutta quella messa a disposizione dai singoli collegamenti. In un sistema centralizzato invece la banda sarebbe limitata alla capacità offerta dal nodo centrale.

maggiore capacità di storage Ogni nodo che partecipa alla rete porta in dote una certa quantità di storage che sarà a disposizione di tutti. Quindi anche se l'investimento per un singolo nodo fosse ridotto, all'aumentare del numero di nodi si raggiungerebbero comunque capacità notevoli.

Oltre al minore costo unitario, vi è anche la possibilità di raggiungere livelli superiori a quelli possibili in caso di unità centralizzata.

maggiore potenza computazionale Oltre alla capacità di storage, ogni nodo della rete rende disponibile una certa potenza computazionale proporzionale alle proprie caratteristiche hardware. Anche ipotizzando che della potenza totale venga resa disponibile una piccola percentuale, è possibile ottenere risorse complessive molto elevate. Ne sono un chiaro esempio i diversi progetti di calcolo distribuito aperti al pubblico, che grazie a questa tecnica sono in grado di ovviare all'utilizzo di potenti e costosi supercomputer.

migliore resistenza ai guasti La possibilità di distribuire il carico di lavoro evita di avere un unico point of failure, di conseguenza la resistenza ai guasti risulta superiore. Ovviamente l'aumento degli elementi in gioco aumenta anche la probabilità di riscontrare malfunzionamenti, ma quest'ultimi risultano circoscritti e non influenzano l'intera rete. Con uno studio accurato dei vari parametri in gioco, sarà inoltre possibile raggiungere valori di affidabilità molto elevati, che in caso di sistemi centralizzati risulterebbero difficilmente ottenibili e molto costosi.

Un altro elemento che trova terreno fertile in questa tipologia di reti, senza però rappresentarne un vantaggio intrinseco, è quello dell'anonimato. A seconda dell'implementazione utilizzata è possibile per i nodi comunicare in forma anonima. Le cause principali si riscontrano nell'assenza di un punto centrale, che possa tenere traccia delle operazioni svolte dai singoli nodi, e dalla possibilità di assegnare ai nodi riferimenti indipendenti dagli indirizzi IP.

In linea di massima le reti peer-to-peer si possono distinguere tra pure e ibride. Le prime sono costituite da nodi tutti considerati allo stesso livello, mentre le seconde, nonostante si basino sugli stessi principi, si appoggiano a nodi che svolgono esclusivamente le funzioni di server.

Una caratteristica importante delle diverse implementazioni di reti p2p è l'overlay routing. L'idea alle sue spalle è quella di astrarre il concetto di routing, in questo modo le risorse cercate possono essere di diversa natura (e non solo indirizzi). L'evoluzione storica dell'overlay routing è la seguente.

Sistemi con directory centralizzata Inizialmente le reti p2p venivano implementate appoggiandosi a server incaricati di tenere traccia dei contenuti resi disponibili da ogni nodo partecipante. In questo modo le ricerche risultavano particolarmente efficienti mentre le reti risultavano di più semplice realizzazione. Lo svantaggio era legato alla vulnerabilità dei nodi centrali che rappresentavano un punto critico per l'intero sistema. Infatti in caso di guasto, i dati sui nodi rimanevano salvi, ma risultavano irreperibili.

Sistemi con ricerca Flood-based Rientra nella categoria di reti p2p pure. Il concetto alla sua base è semplice: ogni nodo inoltra le richieste ai suoi vicini. Il vantaggio principale è quello di non avere single point of failure, ma purtroppo gli svantaggi sono notevoli. Primo tra tutti la scarsa scalabilità, dovuta principalmente a due motivi: crescita esponenziale del numero di messaggi scambiati, col rischio di rimanere vittime di DoS involontari, e crescita lineare del costo di lookup. Questo significa che al crescere della rete aumentano le probabilità di congestione e i tempi di attesa.

Distribuited Hash Tables (DHT) Sfrutta il vantaggio delle tabelle basate su hash, già conosciute da tempo (vengono utilizzate in diversi sistemi tra i quali alcuni database), applicandolo in forma distribuita. Le tabelle hash si basano su funzioni (hash appunto) in grado di associare un indirizzo ai parametri passati (chiave). Questo permette, al momento del recupero, di saltare, o meglio accelerare la fase di ricerca, poiché è possibile ottenere l'indirizzo dell'oggetto semplicemente calcolandone la sua funzione hash. Nel caso distribuito si associa ad ogni nodo un range di chiavi hash delle quali risulta direttamente responsabile (l'associazione avviene in base all'ID del nodo), in questo modo sia per l'inserimento che per il successivo prelievo è possibile indirizzarsi istantaneamente verso il nodo responsabile, se lo si conosce, o verso un suo vicino. Questo sistema fornisce un'ottima scalabilità. Alcune tra le più note implementazioni di DHT sono: Chord, Pastry, Tapestry e CAN. Nonostante questo tipo di reti abbia raggiunto la notorietà presso il grande pubblico grazie agli applicativi di file sharing (a partire dal celeberrimo Napster), le ap-

plicazioni basate su questi protocolli sono parecchie e tra loro troviamo: file system distribuiti, piattaforme VoIP, piattaforme di streaming e i già citati sistemi di calcolo distribuito.

3.2 Il Modello

Come detto nel capitolo precedente, l'obiettivo è quello di realizzare un'architettura distribuita in cui ogni componente collabori all'individuazione di eventuali intrusioni o malware presenti nella rete che ci proponiamo di proteggere.

Per raggiungere questo obiettivo, gli aspetti minimi che ci aspettiamo il nostro sistema IDS ci fornisca, sono:

- individuare il comportamento dei malware
- capire la diffusione dei malware stessi
- attacchi in corso nella rete
- identificazione di indirizzi IP sospetti
- identificazione dei server da cui i malware provengono

La nuova architettura, che ci proponiamo così di realizzare, migliora i precedenti modelli ad organizzazione gerarchica, garantendoci un'elevata scalabilità, fault tollerance, e un carico di lavoro bilanciato, tutti aspetti che nei precedenti modelli non erano presenti.

Per raggiungere questi obiettivi, proponiamo un'architettura ad organizzazione orizzontale, costituita da nodi collaborativi che comunicano tra loro attraverso una rete overlay. Ogni nodo, chiamato *Collaborative Alert Agregator*, svolge le stesse funzioni di alto livello: generare dei local security alert, inoltrare gli alert più rilevanti agli altri nodi collaborativi, analizzare gli eventi ricevuti e comunicare gli esiti delle analisi fatte. Tutte le comunicazioni tra i nodi collaborativi avvengo attraverso una rete overlay peer-to-peer fortemente connessa, garantendoci così il funzionamento senza la necessità di un nodo centrale che svolga funzioni di coodinamento (anche detto supernodo).

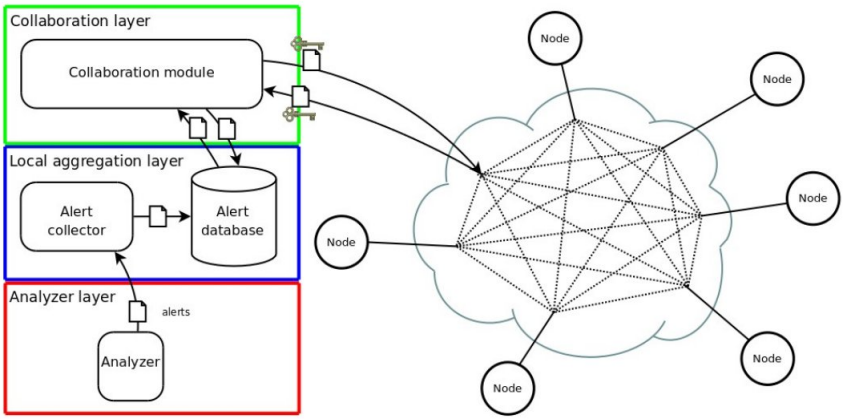

Figura 3.1: *Modello di rete P2P e relativi livelli del nodo collaborativo*

Nel definire questo modello, possiamo identificare tre livelli in cui organizzare ciascun nodo collaborativo della nostra rete: sensor layer, local aggregation layer, collaborative layer.

3.2.1 Sensor Layer

Questo livello costituisce il più basso nella scala dei tre livelli in cui è stato diviso ciascun nodo collaborativo. Di fatto, questo livello è costituito da uno o più tipi di sensori che raccolgono gli alerts provenienti dalla rete o dai pc, e li inoltra al livello superiore.

I sensori, utilizzati nei sistemi IDS più comuni, si possono suddividere nelle seguenti categorie:

- **Host IDS**: questi sensori vanno a monitorare le attività svolte nel PC in cui sono installati, verificando possibili intrusioni o la presenza di malware in esso (ad esempio, carico di lavoro della CPU spropositato rispetto ai processi in esecuzione).
- **Network IDS**: sono sensori posti in posizioni strategiche della rete, che catturano tutto il traffico in essa circolante, per identificare situazioni anomale (ad esempio, traffico elevato verso un sito considerato malevolo), intrusioni o circolazione di malware.
- **Honeypot**: sono una categoria di sensori più soffisticati rispetto ai due precedenti, in grado di individuare i malware e tracciarne le attività,

consentendo poi un intervento più mirato nell'eliminazione del malware stesso.

Non è necessario installare i sensori sulla stessa macchina fisica ove risiedono il local aggregation e il collaborative layer; infatti, questi possono interfacciarsi con gli altri due livelli attraverso una comunicazione da remoto (sfruttando la rete, quindi).

3.2.2 Local Aggregation Layer

Questo livello è responsabile della collezzione, filtraggio e aggregazione degli alerts provenienti da tutti i sensori del livello sottostante.

Il compito fondamentale che il Local Aggregation Layer deve eseguire, è il pre-processing delle informazioni che gli giungono dai sensori. Infatti, come visto precedentemente, nella rete possiamo avere più sensori e di differente tipo; questi, invieranno alerts completamente diversi tra loro (ad esempio, il Network IDS notificherà un'intrusione sulla rete mentre l'Host IDS un malware agente in un PC), che dovranno essere in ugual modo notificati e resi noti al livello superiore, il Collaboration Layer. Il pre-processing ha quindi il compito di elaborare queste informazioni eterogenee, che verranno poi salvate nell'*alert database*, una normalissima base di dati nella quale il Collaboration Layer andrà ad attingere le informazioni elaborate per prendere decisioni su come gestire la situazione di intrusione.

3.2.3 Collaboration Layer

Il Collaboration Layer è l'unico elemento connesso alla rete overlay collaborativa, e come tale sono tre i compiti fondamentali che deve svolgere:

- apprendere nuovi eventi presenti nell'alert database del livello sottostante. Questa operazione può essere eseguita in due modi: *pull mode*, in cui è lo stesso Collaboration Layer a verificare, ad intervalli regolari, la presenza di nuovi eventi nel database; *push mode*, in cui è il Local Aggregation Layer che notifica i nuovi eventi presenti al nostro livello collaborativo.

In fase di progettazione, sarà necessario definire bene quale delle due modalità utilizzare per evitare un elevato numero di messaggi inviati al Collaboration Layer (modalità push), o una frequenza di verifica errata dell'alert database che non consente di individuare i nuovi eventi in tempi corretti (modalità pull).

- ricevere i messaggi inviati dagli altri nodi connessi alla rete overlay.
- è responsabile della diffusione delle informazioni di cui dispone agli altri nodi delle rete overlay.

Gli ultimi due punti, in particolare, sono fondamentali in quanto consento a tutti i nodi della rete overlay di avere lo stesso livello di conoscenza di ciò che sta avvenendo nella rete monitorata, oltre che per accordarsi sulle adeguate contromisure da applicare per gestire la situazione di intrusione che si sta verificando.

Capitolo 4

P2P o Gerarchia?

Nel capitolo precedente è stato definito un possibile modello architetturale che vogliamo utilizzare nella realizzazione del nostro sistema di Intrusion Detection System. Ma siamo davvero sicuri che sia quello più corretto o quello che ci da le migliori prestazioni in termini di gestione e reazione agli attacchi informatici?

Confrontiamo quindi il nostro sistema basato su architettura peer-to-peer con quelli di altri sistemi che utilizzano un'organizzazione gerarchica. I fattori su cui concentriamo il nostro confronto sono: tolleranza ai guasti, carico di lavoro, scalabilità del sistema, e numero di copie di messaggi salvate, necessarie per le comunicazioni tra i nodi.

4.1 Tolleranza ai guasti

La natura distribuita del modello proposto è sicuramente tollerante ai guasti, e privo di un singolo punto di fallimento che è tipico delle architetture gerarchiche e centralizzate, dove tutte le funzionalità sono concentrate nel nodo radice. Questo nodo rappresenta il punto di fallimento del modello gerarchico in quanto, quando questo risulta guasto o non raggiungibile, l'intero sistema non riesce a funzionare. Nel nostro caso, il verificarsi di questa situazione, porta alla completa perdita di protezione delle rete e dei nostri dati. Nella migliore delle ipotesi, a non funzionare può essere un nodo interno del nostro

albero gerarchico, che comporta la mancata protezione della sottorete che a quel nodo fa riferimento.

L'architettura proposta, con la sua natura distribuita, risulta essere più adatta ai nostri scopi. Infatti, ogni nodo è responsabile dell'aggregazione ed elaborazione di un piccolo sottoinsieme di alerts. Inoltre, se un nodo si guasta o diventa irragiungibile, vengono perse solo le informazioni relative a quel nodo, senza compromettere o impedire il corretto funzionamento di tutti gli altri. Una funzionalità in più che ci è garantita dall'uso di questo modello collaborativo, è quella che ogni nodo, individuata la non raggiungibilità di un vicino, può modificare la propria tabella di routing per creare un percorso più diretto con gli altri nodi della rete, escludendo quello non funzionante. Con l'utilizzo di questa nuova funzionalità, la rete è quindi in grado di riorganizzarsi autonomamente senza l'intervento dell'uomo.

Per colmare una delle poche carenze del modello collaborativo, e cioè la perdita di informazioni relative al nodo guasto, possiamo utilizzare un modello di replicazione dei messaggi. Se definiamo quindi un valore k, chiamato *costante di replicazione*, possiamo creare k copie dello stesso messaggio e distribuire le *k-1* nei vicini del reale destinatario; così facendo, ci garantiamo che se un nodo fallisce, le informazioni che esso custodiva, possono essere recuperate dai suoi vicini. Ovviamente questa soluzione aumenta la quantità di messaggi che devono essere scambiati tra i nodi della rete e la quantità di informazioni che ogni singolo nodo dovrà memorizzare. Un'opportuna definizione del valore k consentirà di giungere ad un compromesso tra overhead di messaggi e la tolleranza ai guasti che la replicazione ci garantisce.

4.2 Carico di lavoro

Un'architettura gerarchica concentra la propria analisi dei malware e gestione degli alert nel proprio nodo radice, evitando così la replicazione delle informazioni, vista precedentemente. Se da un lato questo può essere visto come un fattore positivo, dall'altro mostra come il carico computazionale della radice sia fortemente maggiore rispetto a quello di tutti gli altri nodi dell'architettura, che non solo dimostra una cattiva gestione delle risorse disponibili,

ma anzi richiede un forte investimento economico per la realizzazione di un nodo radice altamente prestante e funzionale.

Se andiamo a vedere il modello collaborativo proposto, è intrinseco nella natura del peer-to-peer un uso sapiente ed equilibrato di tutte le risorse disponibili, consentendone un uso bilanciato. Allo stesso modo, non solo la fase di elaborazione delle informazioni, ma anche la fase di scoperta e raccolta degli alert risulta migliore. Infatti, nel caso di un'organizzazione gerarchica, solo i nodi a stretto contatto con quello infettato si accorgono dell'intrusione e lo stesso attaccante, sovracaricando i loro canali di comunicazione, può impedire loro di comunicare la scoperta. Ciò non può avvenire nel modello collaborativo, in cui tutti i nodi della rete partecipano alla scoperta di nuovi alert e in cui il numero di canali con cui ogni nodo comunica è talmente elevato da impedire all'attaccante di sovracaricarli (dovrebbe bloccare l'intera rete, caso pressochè impossibile).

4.3 Scalabilità

In un modello architetturale di tipo gerarchico, come quello visto, ogni nodo comunica con un nodo padre, che a sua volta avrà un ulteriore nodo padre, e così via fino al nodo radice. Con questa struttura, ogni nodo è in grado di collezzionare un numero finito di alerts provenienti dai nodi del livello sottostante, ed è in grado di gestire un numero finito di questi nodi. Questa struttura la possiamo quindi modellare attraverso un albero n-ario, il cui numero di nodi intermedi cresce in maniera logaritmica all'aumentare del numero di foglie; ciò significa che più sono i pc utilizzati per monitorare la rete, più nodi intermedi con la radice sono necessari per coordinare l'intera infrastruttura.

Nell'altra architettura proposta, la rete collaborativa, i sistemi di raccolta e analisi degli alerts sono distribuiti nella rete e non vi è la necessità, come nell'architettura precedentemente esposta, di una struttura intermedia di gestione. Questo rappresenta un enorme vantaggio in termini di scalabilità, in quanto non è necessario riconfigurare la rete ogni volta che si aggiungono nuovi nodi al nostro sistema di rilevamento intrusioni, come invece è necessario fare nell'organizzazione gerarchica.

4.4 Numero di copie salvate

Un ulteriore vantaggio delle architetture collaborative è rappresentato dall'esiguo numero di copie di ciascun alert che ogni nodo deve memorizzare e conservare. Infatti, come visto precedentemente, la tecnica di replicare nei vicini di un nodo destinatario la copia dei messaggi ad esso indirizzati, consente al sistema di rilevamento intrusioni un funzionamento ottimale anche in caso di fallimento di uno o più nodi.

In un'architettura gerarchica, una copia di ogni alert viene mantenuta in ogni nodo della fascia intermedia di gestione da cui l'alert ha avuto origine. Consideriamo ad esempio un albero con n nodi di gestione, l foglie e altezza $h = \log_n(l)$. Se indichiamo con c il numero di copie in cui ogni alert viene salvato nei nodi della struttura gerarchica, abbiamo:

$$h \leq c \leq \sum_{i=0}^{h-1} n^i \tag{4.1}$$

Questo significa che il numero di copie c è compreso tra il numero di nodi gestori nel percorso tra la foglia che ha generato l'alert e la radice (h) e il numero totale di nodi gestori quando lo stesso alert viene rilevato da tutte le foglie dell'albero ($\sum_{i=0}^{h-1} n^i$). Dato che h cresce in modo proporzionale al logaritmo delle foglie, il numero di copie di ogni alert che la struttura gerarchica necessita di mantenere cresce in maniera uguale seguendo il logaritmo del numero di foglie.

Nell'altro caso invece, la struttura collaborativa richiede che il numero di copie di ciascun alert sia determinato dal fattore di replicazione k, che è invece un parametro configurabile e indipendente dal numero di nodi che costituiscono la rete che vogliamo proteggere.

Nella tabella qui riportata, vi è un confronto tra le due architetture proposte. Nella prima riga è riportato il numero di copie necessarie alla rete collaborativa: il numero dipende da un solo parametro, il fattore di replicabilità k. Le altre rige contengono il numero di copie memorizzate dall'architettura gerarchica con differenti numeri di nodi. Ad esempio, la seconda riga mostra come in una rete gerarchica con *l=1000* nodi, il numero di copie salvate stra

Architettura	Minimo	Massimo
Collaborativa, k=5	5	5
Gerarchica, l=1000, n=10	3	111
Gerarchica, l=10000, n=10	4	1111
Gerarchica, l=100000, n=10	5	11111
Gerarchica, l=8000, n=20	3	421
Gerarchica, l=1600000, n=20	4	8421

Tabella 4.1: *Numero di copie di messaggi memorizzate nella struttura gerarchica e in quella collaborativa*

tra 3 e 111, evidenziando come sia fortemente dipendente dal numero di foglie che individuano lo stesso alert.

Capitolo 5

Realizzazione di un prototipo

Dopo aver discusso dal punto di vista teorico il modello collaborativo proposto, e dopo averlo confrontato con il suo opposto, il modello gerarchico, andiamo a dimostrarne la validità realizzando concretamente un prototipo.

In accordo con il modello proposto nel Capitolo 3, ogni *Collaborative Alert Aggregator* consiste di differenti moduli software che possono essere divisi in tre classi. Le prime due classi includono dei tipici sistemi di difesa per le reti; la terza include i moduli per le comunicazioni tra i nodi. Come ipotesi di lavoro, è stato deciso di implementare in ciascun nodo del sistema IDS ciascuno dei tre livelli definiti nel modello visto, ritenendo che l'intero sistema IDS può essere più solido se ciascuno dei nodi che lo compone dispone localmente di tutti gli strumenti necessari ad operare.

Vediamo in dettaglio come realizzare ciascuna delle tre classi sopra definite.

5.1 Sensori (Livello 1)

Come già detto in precedenza, a questa classe appartengono tutti quei software che abbiamo definito *sensori*, il cui obiettivo è quello di monitorare la rete e i pc, e raccogliere informazioni su questi ultimi da passare ai livelli superiori. In particolare, sono tre i principali processi di monitoraggio che questa fase del sistema IDS deve implementare:

- monitoraggio della rete, in termini di traffico in transito

- monitoraggio accessi fraudolenti locali, in termini di tentativi di accessi non autorizzati nel singolo PC, o tentativi di modifica di file di sistema
- monitoraggio malware, in termini di infezione da parte di virus e loro possibile diffusione

Per realizzare queste tre principali funzioni, ci serviamo di software opensource liberamente disponibili che, opportunamente installato e configurato, svolgerà un'attività di monitoraggio completa ed esaustiva. I software di cui ci serviamo, sono:

- **Snort:** è un programma di *packet sniffing* che consente di rilevare eventuali tentativi di intrusione in un sistema, tramite il confronto delle "impronte" dei pacchetti di rete in arrivo e quelle conservate in un database (rules). Rilevata un'attività sospetta può svolgere sia compiti di registrazione che di avviso all'amministratore. Non svolge però compiti di difesa attiva, a differenza dei firewall, che quindi rimangono a carico di altre applicazioni.

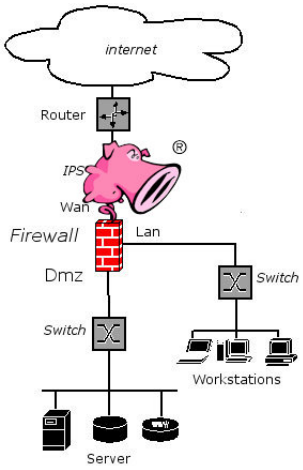

Figura 5.1: *Esempio modello architetturale in cui inserire Snort*

- **OSSEC:** è un *Host-based Intrusion Detection System* (HIDS) opensource ovvero un software in grado di monitorare il funzionamento di un sistema "dal suo interno" anziché ricorrere all'uso delle interfacce di rete (come fanno invece i network-based intrusion detection system, quali Snort). I software HIDS si occupano di sorvegliare costantemente il

comportamento del sistema oggetto d'esame e lo stato in cui sta operando. Mentre i NIDS ispezionano il contentuto di ogni singolo pacchetto dati in transito, un HIDS qual è OSSEC può stabilire a quali risorse tentano di accedere i programmi installati bloccando tempestivamente operazioni sospette. Un word-processor non deve, ovviamente, modificare le password di sistema o la configurazione di aree vitali: se lo fa significa che, con buona probabilità, sta eseguendo del codice maligno, collegato all'azione di un malware. Un HIDS, inoltre, controlla lo stato del sistema verificando le informazioni memorizzate, sia in RAM che sul file system, i file di log e così via. L'obiettivo è quello di rilevare tempestivamente anomalie. In particolare, OSSEC si incarica di analizzare i log di sistema, verificare l'integrità dei file memorizzati su disco, controllare la presenza di rootkit e tenere traccia delle performance di ogni macchina collegata alla rete locale nella quale è installato. Come si può notare anche dalla figura, OSSEC è multipiattaforma e disponibile per tutti i principali sistemi operativi presenti in commercio.

Figura 5.2: *Modello architetturale OSSEC*

- **Nepenthes:** è un sistema honeypot il cui obiettivo è individuare eventuali vulnerabilità a cui è soggetto il sistema e che possono essere oggetto di attacco da parte di malware. Per svolgere questo compito, il software emula i principali servizi normalmente attivi nei pc e soliti oggetti ad attacco; Nepenthes quindi individua l'attacco scaricando il codice malevolo e l'indirizzo da cui questo attacco proviene, ovviamente impedendo che l'attacco vado a buon fine (essendo il servizio emulato) e che il pc subisca danni. Inoltre, invia le caratteristiche dell'attacco a Norman Sandbox (uno dei principali centri mondiali di analisi malware), che restituisce una reportistica completa sull'intrusione in atto e le rispettive contromisure da adottare per proteggersi da futuri attacchi simili.

Attraverso questi semplici strumenti e la loro configurazione, siamo in grado di soddisfare alle esigenze di monitoraggio di cui il nostro sistema IDS necessità. Ogni volta che questi sensori che abbiamo disseminato nella rete individuano delle situazioni anomale, inviano delle notifiche, chiamate *alert*, al livello superiore. Vediamo come questo livello viene realizzato.

5.2 Manager (Livello 2)

In questo livello vengono raccolti tutti gli alert provenienti dai sensori posti a livello sottostante; il sistema, per come viene realizzato, è asincrono verso il livello sottostante, cioè si limita a riceve gli alert dai sensori, senza interagire con questi ultimi. Queste informazioni ricevute, sono strutturate secondo lo standard IDMEF (Intrusion Detection Message Exchange Format), standard comune a tutti gli strumenti di intrusion detection, e poi memorizzate all'interno di una base di dati.

Lo standard IDMEF è nato per la rappresentazione dei messaggi d'allarme prodotti dagli IDS, con lo scopo di favorire l'interoperabilità tra differenti sistemi di rilevamento delle intrusioni e facilitare la comunicazione con i moduli di gestione e correlazione. Tale standard adotta XML (eXtendible Markup Language) per la creazione degli alert, in quanto permette di definire, con una sintassi semplice e direttamente comprensibile, le grammatiche

di diversi linguaggi derivati e al tempo stesso può venire impiegato per la rappresentazione e lo scambio di dati gerarchicamente strutturati.

Nella realizzazione di questo livello intermedio, ci serviamo di software opensource preesistente. In particolare, utilizziamo il software Prelude che fungerà da aggregatore attraverso il suo modulo Prelude Manager. Il compito di questo modulo software è abbastanza semplice e può risultare banale, ma di fondamentale importanza per il livello superiore. Questo riceve gli alert dai sensori sottostanti, strutturati secondo lo standard IDMEF, e li inserisce all'interno di un DBMS. La motivazione che ci spinge a salvare le informazioni in una base di dati, trova radice in due principi:

- il primo è fondamentalmente di log, mantengo cioè traccia di tutto quanto è avvenuto nel sistema e le segnalazioni che sono giunte al singolo nodo
- il secondo, e più importante, è che il livello superiore sarà costituito da un modulo software da me progettato e realizzato, che avrà bisogno di un elevato bacino di informazioni da cui attingere per prendere delle decisioni "informate" su quelli che posso essere gli attacchi in corso e quali contromisure adottare per arginare il fenomeno.

A tal proposito, in ciascuno dei nodi costituenti il sistema IDS, sarà installato il DBMS opensource MySQL che localmente avrà il compito di mantenere il patrimonio informativo degli alert che giungono nel sistema. Questa base di dati, verrà periodicamente interrogata, in modo asincrono, dal livello superiore, con una cadenza opportunamente studiata per:

- evitare interrogazioni troppo frequenti: se il sistema interroga troppo frequentemente la base di dati, questo si traduce in un carico di lavoro esagerato ed inutile per il DBMS, oltre al fatto che il livello superiore passerà più tempo ad interagire con il manager piuttosto che eseguire le funzioni per cui è stato progettato
- evitare interrogazioni troppo poco frequenti: se il livello superiore interroga la base di dati raramente, non avrà conoscenza di eventuali attacchi in corso e sarà quindi più difficile contrastarli tempestivamente

Fatte queste considerazioni, andiamo quindi a vedere quale sarà il compito del terzo livello definito nel nostro modello architetturale.

5.3 Comunicatore (Livello 3)

Questo livello è il più importante, in quanto deve interagire con il livello sottostante prelevando gli alert raccolti dai sensori, deve ricevere ed inviare notifiche provenienti dagli altri nodi del sistema IDS, oltre che attuare le adeguate contromisure al problema rilevato. Facendo una similitudine con il corpo umano, possiamo dire che questo livello è il cervello e cuore dell'intero nostro progetto.

Data l'importanza all'interno del sistema, questo comunicatore è stato completamente progettato e realizzato dal principio. Data la natura distribuita del sistema IDS e l'eterogeneità dei sistemi operativi presenti nei nodi, si è scelto di sviluppare questo livello attraverso il linguaggio di programmazione Java. Questo linguaggio si presta bene a tale scopo in quanto orientato agli oggetti, indipendente dalla piattaforma, contiene strumenti e librerie per il networking, e può anche essere progettato per eseguire codice da sorgenti remote in modo sicuro.

Una seconda motivazione che ha contribuito alla scelta dello sviluppo attraverso Java, è il framework FreePastry. Questo framework è la versione gratuita del sistema commerciale Pastry, sviluppato da alcune università americane, che realizza un substrato efficiente e scalabile per applicazioni peer-to-peer. FreePastry non fa altro che realizzare una rete overlay, cioè una rete virtuale atta a fornire meccanismi non nativamente supportati dalla rete sottostante, quali soluzioni avanzate di instradamento, particolari forme di routing, multicast, indirizzamento e caching di contenuti, servizi di sicurezza e segretezza, gestione a priorità del traffico, tecniche di compressione e diffusione di flussi multimediali, ecc. Ma l'aspetto a noi più utile al fine del progetto, è che attraverso la rete overlay siamo in grado da dare una qualunque topologia al sistema IDS, indipendentemente da quella reale con cui i nodi sono interconnessi; questo solo grazie alle funzioni avanzate di routing che FreePastry implementa.

La topologia scelta per il sistema IDS è una peer-to-peer fortemente connessa con DHT (Distributed Hash Table). Ciò significa che ogni nodo del sistema è collegato ad un insieme di altri nodi, in modo che ciascuno abbia almeno un vicino (non esistono quindi nodi isolati). Ma che ruolo ha la DHT? Le ta-

belle di hash distribuite sono una classe di sistemi distribuiti decentralizzati che partizionano l'appartenenza di un set di chiavi tra i nodi partecipanti, e possono inoltrare in maniera efficiente i messaggi all'unico proprietario di una determinata chiave, tipicamente progettate per gestire un vasto numero di nodi, anche nei casi in cui ci siano continui ingressi o improvvisi guasti di alcuni di essi. Data quindi la natura distribuita del sistema e la scalabilità che vogliamo presenti, l'utilizzo di DHT è fondamentale per soddisfare questi due requisiti.

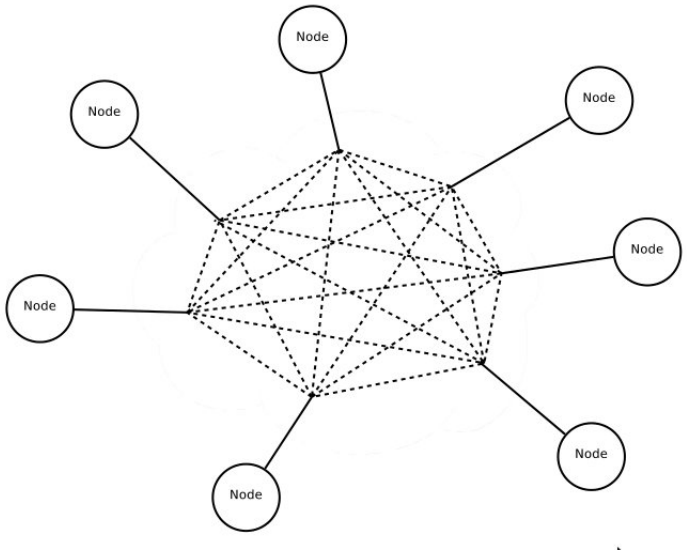

Figura 5.3: *Rete peer-to-peer fortemente connessa con DHT*

Prima di procedere con la descrizione dell'implementazione, è bene definire il formato dei messaggi che i nodi aderenti al sistema IDS si scambiano. Dato che tutti i messaggi di controllo atti a mantenere la topologia della rete overlay sono completamente gestiti dal framework ed a noi trasparenti, il formato che qui definiamo è quello degli alert scambiati dai nodi. Come si può ben capire da quanto descritto fino ad ora, sono due le informazioni di fondamentale importanza per il nodo: un identificativo dell'attacco in corso e da dove questo attacco proviene, sorgente descritta attraverso il suo indirizzo IP. Come mostra la figura che segue, al messaggio decidiamo di allegare altre tre informazioni ritenute importanti: l'identificativo dell'alert, formato da una codifica esadecimale, in modo tale che il nodo sia in grado di discriminare gli

alert già esaminati e duplicati da quelli nuovi e mai visti; il mittente di questo messaggio, cioè l'indirizzo IP del vicino da cui quest'informazione proviene; il grado di confidenza o veridicità che il mittente ha attribuito a tale alert. Queste ultime due informazioni sono di particolare importanza in quanto il nodo ricevente dispone di informazioni sullo stato dei suoi vicini (e quindi del mittente) e capisce se ed in che misura considerare veritiera l'informazione che gli è giunta.

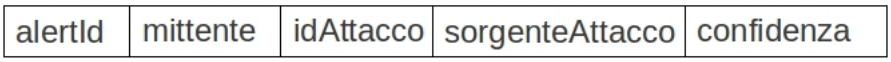

Figura 5.4: *Formato messaggi scambiati tra i nodi*

Il framework FreePastry realizza e mantiene autonomamente la rete overlay, senza alcun nostro intervento dall'esterno ed in modo totalmente trasparente; l'unico nostro compito è stato quello di realizzare le classi per l'interfacciamento con i livelli sottostanti e gli altri nodi, alla scoperta di nuove intrusioni. A tal scopo, andiamo ad analizzare nel dettaglio le classi Java implementate.

MyProject.java E' la classe che contiene il main, e quindi l'unica eseguibile. Tale classe esegue tre compiti fondamentali: attraverso il framework, crea un nuovo nodo ed esegue il join nella rete (overlay) del sistema IDS; esegue un thread per la verifica periodica di nuovi alert presenti nel database del livello sottostante (verifica locale di intrusioni); esegue un thread per la ricezione di alert provenienti dagli altri nodi.

Tabella Visione del Mondo Questa non è una classe Java, ma bensì una tabella che dinamicamente viene costruita da ciascun nodo e di fondamentale importanza. In essa, sono contenuti i livelli di confidenza che ciascun nodo attribuisce agli alert che analizza. Questa tabella è stata realizzata in quanto ogni nodo può disporre di informazioni differenti dagli altri nodi del sistema; in questo modo, ad ogni alert che analizza (sia locale che ricevuto dai vicini) può essere assegnato un peso differente in termini di quanto veritiero risulta essere sulla base delle infor-

mazioni di cui il nodo dispone. Questa tabella viene aggiornata dai due thread InvioAlert e RicAlert.

InvioAlert.java E' uno dei due thread attivati, che si occupa dalla verifica locale al nodo di eventuali intrusioni in atto nel nodo stesso o nei vicini. Infatti, i NIDS sono in grado di rilevare eventuali attacchi rivolti ad altri pc connessi alla rete, analizzando i pacchetti in transito. Ecco quindi che il modulo Java implementato, una volta individuata la presenza di un nuovo alert nel database, diversifica il suo comportamento a seconda che l'attacco sia rivolto al nodo in cui è in esecuzioni o ad un suo vicino. Se l'attacco è locale, viene creato un messaggio con il formato precedentemente definito e inviato ai vicini, per portarli a conoscenza della situazione individuata. Se viceversa l'attacco è rivolto ad un altro nodo, viene aggiornata la tabella di visione del mondo locale al nodo, e costruito il messaggio da inviare ai vicini con anche il livello di confidenza stimato per l'alert rilevato. Se con un livello di confidenza stimato superiore o uguale al 50%, l'alert viene propagato ai vicini, viceversa nessun messaggio verrà inoltrato; questa scelta è stata presa per evitare di innondare la rete di messaggi.

RicAlert.java Questa classe implementa il secondo dei due thread realizzati, occupandosi di ricevere le informazioni dai nodi vicini (non va quindi in alcun modo ad interagire con i livelli sottostanti locali al nodo). Ogni volta che il nodo riceve un messaggio dai vicini, viene aggiornata la tabella di visione del mondo stimando il grado di confidenza che il nodo può attribuire all'alert ricevuto. Come avviene nell'altro thread, se il livello di confidenza stimato risulta essere superiore o uguale al 50%, il nodo si preoccuperà di propagare ai vicini l'informazione appena giuntagli; viceversa, nessuna informazione verrà propagata ed il nodo rimarrà in attesa di nuovi alert dai vicini.

MyMsg.java Questa classe non fa altro che utilizzare i costruttori del framework FreePastry per comporre i messaggi che i nodi si scambiano secondo il formato da me definito.

MyApp.java Questa classe implementa i metodi definiti nel framework, in particolare per quanto riguarda l'invio e la ricezione dei messaggi, limitandosi a definire quello che è il messaggio da inviare ed il suo destinatario che le classi del framework, concretamente si preoccuperanno di inviare e ricevere.

Database.java In questa classe vengono definiti tutti quei metodi necessari alla connessione ed interrogazione del database MySql che contiene tutti gli alert che localmente i sensori hanno individuato.

Una volta terminata la fase di scoperta dell'intrusione e individuato l'attacco, ogni nodo può attuare localmente delle adeguate contromisure, che cambiano a seconda del tipo di attacco:

- impostare una regola nel firewall per impedire la comunicazione con un dato nodo infetto
- attivare l'antivirus per la presenza di malware
- cambiare permessi ad un file di sistema che stanno cercando di modificare, impedendo di portare a termine la modifica
- impedire l'installazione di una dato programma (rootkit)
- nella peggiore delle ipotesi, contattare l'amministratore di sistema.

5.4 Problematiche del prototipo proposto

Il prototipo realizzato come sopra, potrebbe presentare qualche piccola debolezza che, sfruttata da un'attaccante, può compromettere la sicurezza dell'intera rete e rendere vano lo sforzo di realizzare una struttura di questo tipo per la sua protezione.

Andiamo ad analizzare queste possibili debolezze nel dettaglio.

Livello 1 Come posso garantire che le segnalazioni provenienti dai sensori siano reali ed affidabili e non provenienti da un'attaccante? Ogni sensore, per poter comunicare con il nostro manager Prelude, deve avere una chiave privata RSA con la quale codifica ciascun messaggio che invia all'aggregatore, il quale sarà l'unico in grado di leggerli. Nel caso siano corrotti o codificati con una chiave errata, il sistema è subito

in grado di rilevare i sensori compromessi ripristinandoli (grazie ai file di configurazione precedentemente salvati) o richiedendo l'intervento dell'amministratore di sistema per la loro sostituzione.

Livello 2 Considerando il worst case scenario, può accadere che il nostro aggregatore di alert Prelude o il database che quest'ultimo utilizza per memorizzare le informazioni vengano compromessi; come mi accorgo di questa situazione? Localmente il singolo host non è in grado di capire se ciò che è memorizzato nel database sia corretto e affidabile o meno; dobbiamo quindi affidarci agli altri nodi che compongono il sistema IDS. In particolare, se N è il numero di nodi del sistema IDS attivi al momento, e $N/2 + 1$ segnalano al nodo che in esso è presente un'intrusione (e non vi è riscontro nei dati in suo possesso), il nodo decide di "fidarsi" degli altri nodi e applicare le adeguate contromisure (nell'ipotesi che la rete sia talmente grande che $N/2+1$ nodi siano difficilmente tutti compromessi).

Livello 3 Questo livello è costituito dal modulo, scritto in Java, che realizza la comunicazione tra i nodi; cosa succede se l'attaccante sostituisce questo modulo con un proprio sistema, che quindi comunica false informazioni ai nodi? Per evitare questa situazione, è possibile utilizzare un certificato digitale per le comunicazioni tra i nodi del sistema IDS, certificando così l'identità degli interlocutori. Però, così facendo, come posso essere sicuro che l'informazione che questi si stanno scambiando sia corretta e affidabile? A questo livello, basta certificare la comunicazione, in quanto l'affidabilità dell'informazione viene risolta ai livelli sottostanti (garantendo l'affidabilità dei sensori e dell'aggregatore di alert). Il problema che ora si presenta, utilizzando il certificato digitale nelle comunicazioni, è individuare un'entità nel sistema che certifichi e mantenga le chiavi utilizzate dai certificati. Data la delicatezza delle informazioni che questa entità dovrà gestire (compromesse le chiavi dei certificati, è compromesso l'intero livello di comunicazione), queste funzioni verranno accentrante in un unico nodo il cui unico compito sarà quello di KDC (Key Distribution Center), fortemente protetto, disponibile H24 e 365 gg/anno. Tale soluzione è stata pensata e progettata, ma non implemen-

tata nel prototipo per lasciar posto al tema centrale di questa tesi, ossia il rilevamento di intrusioni.

Capitolo 6

Risultati Sperimentali

In questo capitolo ci occupiamo ora di validare, con i risultati sperimentali ottenuti, il prototipo realizzato. Vediamo quindi in dettaglio il comportamento dei singoli componenti del sistema IDS realizzato e come questi hanno contribuito al comportamento generale dell'intera rete.

6.1 Validazione Componenti

Il progetto è stato completamente sviluppato in ambiente Linux, in particolare utilizzando Ubuntu 10.04. Questa scelta di utilizzare un singolo sistema operativo, è stata dettata dalla poca disponibilità di risorse hardware a disposizione per la validazione pratica. Infatti, se pensiamo ad uno scenario concreto e reale, tale scelta sarebbe sicuramente riduttiva e di scarso interesse pratico.

Nella realizzazione pratica del progetto è stato realizzato un singolo nodo in tutte le sue componenti (sensori, manager, comunicatore), opportunamente configurate, e poi, attraverso il tool Remastersys, è stata realizzata un'immagine dell'intero nodo. Attraverso questo tool, è stato possibile realizzare una distribuzione Linux Live del sistema realizzato e, attraverso più copie, realizzare una rete con più PC per la fase di testing. Quest'ultima, è avvenuta in ambito domestico, in una rete posta dietro ad un firewall in cui gli eventi venivano appositamente generati per verificare il comportamento dell'intero progetto a seconda delle situazioni.

Vediamo ora nel dettaglio il comportamento delle singole componenti di cui il nostro sistema si compone.

OSSEC Come descritto nel capitolo precedente, il compito di questo sensore è di HIDS, cioè monitorare il singolo nodo nel quale è installato con l'obiettivo di individuare comportamenti anomali. Dato l'ambito di simulazione, in cui tutti gli utenti del sistema sono conosciuti e fidati, le notifiche che da questo sensore giungevano erano principalmente informative. Le principali giunte al nostro aggregatore sono state il corretto login da parte di un particolare utente, piuttosto che il corretto inserimento della password di amministratore per l'esecuzione di qualche istruzione con privilegi di root. Per verificare che OSSEC si comportasse correttamente in tutte le situazioni, si è simulata un'intrusione cercando di modificare il file di sistema contenente gli utenti, senza averne il permesso. Non solo OSSEC ha immediatamente segnalato tale situazione, ma rispetto alle notifiche precedenti, vi ha attribuito una priorità massima, indicata con il termine inglese *high*, permettendo così la sua validazione.

Snort Il compito di questo sensore, per come è stato concepito, era quello di sniffare la rete alla ricerca potenziali minacce racchiuse nei pacchetti in transito. Dato che nella rete domestica di testing, per di più protetta da firewall, non vi erano pacchetti "pericolosi" in transito, si è scelto di porre almeno un nodo al di là del firewall, completamente esposto ad Internet. In soli dieci minuti in cui il nodo è stato nella DMZ (Demilitarized Zone), Snort ha generato ben 980 notifiche, di cui circa la metà con la massima priorità. Tra queste, una è spiccata tra tutte: il sensore ha individuato il tentativo, da parte di un worm, di accedere ed infettare il sistema, dimostrandone così il suo corretto funzionamento anche in termini di riconoscimento di differenti tipologie di attacco.

Nepenthes Per testare quest'ultimo sensore, il cui compito è simulare dei servizi attivi nel sistema per individuare eventuali attacchi a questi ultimi, ci siamo serviti di Nessus. Tale strumento è ben noto nella cerchia degli amministratori di sistema, in quanto è in grado di testare tutti i

PC connessi alla rete provando tra una vasta schiera di vulnerabilità continuamente aggiornate. In questa fase, non solo Nepenthes ha individuato il tentativo di attacco in corso, ma ha anche indicato la vulnerabilità che l'attaccante stava cercando di sfruttare, grazie all'appoggio a Norman SandBox. Quest'ultimo è un sito web che raccoglie notifiche provenienti da sensori come Nepenthes, le analizza, e grazie alla sua base di dati di vulnerabilità note, è in grado di restituire l'attacco in corso e le contromisure da adottare perchè tale attacco non vada a buon fine.

Prelude e MySQL La validazione di questi due strumenti, che realizzano il livello intermedio del modello architetturale proposto, è avvenuta indirettamente con le notifiche che i sensori hanno inviato. Infatti, il compito di questi due tool era quello di raccogliere e memorizzare tali informazioni, e renderle disponibili al livello superiore. Per avere anche un riscontro visivo, ci siamo serviti del tool Prewikka che, attraverso una finestra del browser, visualizza le notifiche giunte all'aggregatore Prelude.

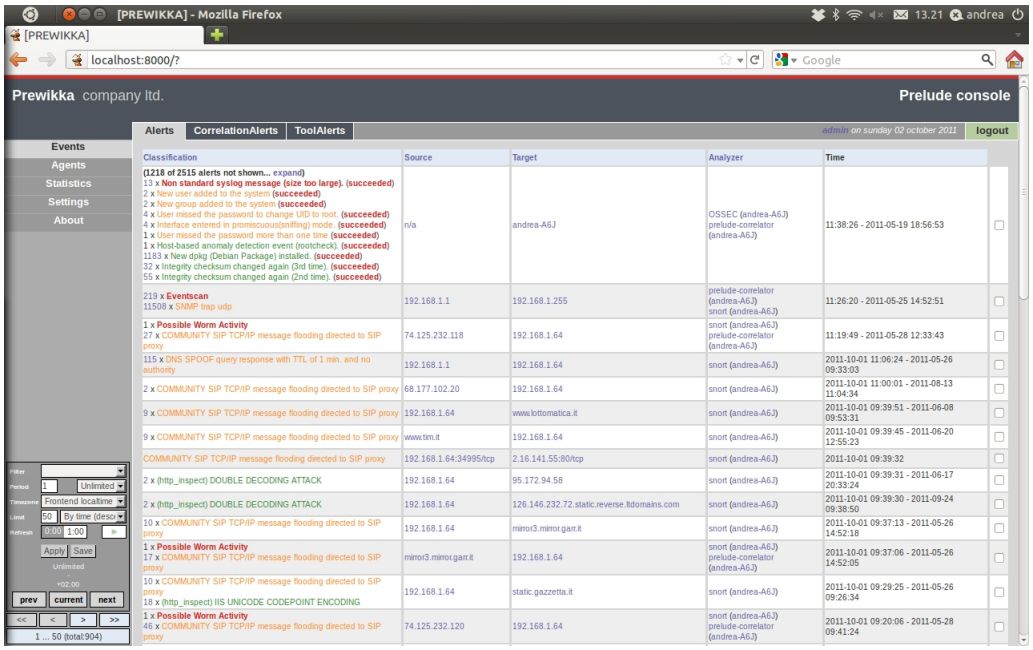

Figura 6.1: *Prewikka - Visione degli alert giunti dai sensori*

Comunicatore L'unico testing che localmente al nodo possiamo fare per questo componente è verificare che esegua correttamente l'interfacciamento con il livello sottostante. Infatti, la validazione del comunicatore è descritta nel paragrafo successivo quando descriveremo il comportamento globale della rete. Per quanto riguarda l'interazione locale, il pacchetto Java realizzato, attraverso il driver ODBC di MySQL, si interfaccia perfettamente con il database contenente gli alert rilevati, effettuando una verifica periodica secondo le tempistiche definite in 5.2.

6.2 Validazione della rete

Dopo aver verificato il corretto comportamento dei singoli componenti che compongono il nostro sistema, vediamo come si comporta la rete IDS risultante. Il testing di questa fase avverrà in due differenti modalità: una modalità reale, in cui a ciascun nodo corrisponderà un reale PC; una modalità simulata, in cui verranno virtualizzati più nodi nei singoli PC. Data la limitata disponibilità di risorse hardware a disposizione, la prima modalità sarà costituita da un numero esiguo di nodi rispetto alla seconda. Si è però deciso di eseguire ugualmente questo test per vedere come realmente si può comportare il sistema progettato in un ambito reale, seppur di limitate dimensioni. Ovviamente anche la seconda modalità di testing è doverosa, in quanto il progetto nasce per essere applicato a qualunque tipo di rete e dimensione, e come tale una verifica anche su nodi virtuali può essere rappresentativa del comportamento tenuto dal sistema.

6.2.1 Nodi Reali

Come descritto precedentemente, in questa modalità di testing del sistema, ad ogni nodo IDS corrisponde un reale PC. In particolare, nelle prove effettuate, sei è stato il numero massimo di nodi testati.

L'aspetto che più ci premeva testare in questo ambito, non era tanto la scalabiltà del prototipo proposto o la sua tolleranza ai guasti, ma il traffico di informazioni generato nella rete. Infatti, tolleranza ai guasti e scalabilità in un ambito così ristretto di al più sei nodi, risultano poco significativi; vice-

versa, sei nodi reali connessi ad una piccola rete domestica, possono già dare un buon indice del traffico di messaggi generato. Utilizzando un settimo PC connesso alla rete, con un programma di sniffing come Snort, ci siamo preoccupati di monitorare il traffico di rete quando il sistema IDS è in funzione. Sono emerse due componenti di traffico:

- una componente costante che risulta presente dal momento in cui viene attivato il sistema IDS al momento in cui viene disattivato (quindi quando anche l'ultimo nodo viene terminato). Tale componente è costituita da messaggi utilizzati dal framework per mantenere la topologia della rete overlay, cioè mantenere aggiornate le tabelle di routing dei singoli nodi.
- una seconda componente che si presenta come un "escalation" di messaggi quando un'intrusione è individuata e inizia la comunicazione tra i nodi. In questa situazione, si vede aumentare il numero di messaggi in circolo abbastanza rapidamente, ma non raggiunge mai livelli preoccupanti o la saturazione della rete stessa. In più, la durata del fenomeno è breve e la rete riprende velocemente la normale attività.

Se pensiamo ad un caso reale di un'organizzazione con centinaia di PC, la normale rete utilizzata per la loro connessione può risultare sufficiente a sostenere anche il traffico generato dal sistema IDS; probabilmente potrà presentare qualche piccolo rallentamento nella fase di propagazione degli alert quando è individuata un'intrusione. Se vogliamo evitare anche questo piccolo "intoppo" e, allo stesso tempo, aumentare l'affidabilità del mezzo comunicativo, potremmo utilizzare una rete dedicata per le comunicazioni tra i nodi del sistema IDS.

6.2.2 Nodi Virtuali

Questa seconda modalità di validazione del sistema IDS proposto, risulta sicuramente più utile ed interessante della precedente. Infatti, attraverso questa modalità siamo in grado di vedere come il sistema si comporta in un ambito reale, dove il numero di nodi è considerevole.

Nelle prove effettuate, progressivamente venivano aumentati il numero di nodi che componevano il sistema IDS fino ad un massimo di 50 nodi totali

(limite fisico delle risorse disponibili). La prima cosa che è saltata all'occhio è che all'aumentare del numero di nodi, aumentava la velocità con cui l'alert di un possibile attacco in corso si propagava nella rete e raggiungeva tutti i nodi. Un buon contributo a questo fatto è dato dalla virtualizzazione e dalla presenza su un singolo nodo fisico di più nodi virtuali, ma questo conferma anche le ipotesi fatte nei capitoli precedenti riguardo alla scalabilità. Infatti, questo dimostra che, a differenza di un'architettura gerarchica, non si verificano "colli di bottiglia" all'aumentare del numero di partecipanti al sistema IDS, ma bensì aumenta l'efficienza e la velocità di propagazione delle informazioni. All'atto pratico, questo è importantissimo in quanto si traduce in una tempestiva individuazione dell'intrusione in atto.

Un ulteriore prova che in questo ambiente simulato si è voluto testare, è stata la resistenza ai guasti. Infatti, una volta generato un evento che ha stimolato i sensori nella produzione di alert (nello specifico, si è provato a modificare i file di sistema di un host), sono stati terminati via via un numero di nodi sempre maggiore. Confortando le ipotesi fatte durante lo studio del modello, il simulato guasto di alcuni nodi non ha impedito al sistema di funzionare; a tutti i nodi attivi sono giunte le notifiche dell'attacco in corso, con tempistiche leggermente superiori alle precedenti dovute al ripopolamento delle tabelle di routing della rete overlay.

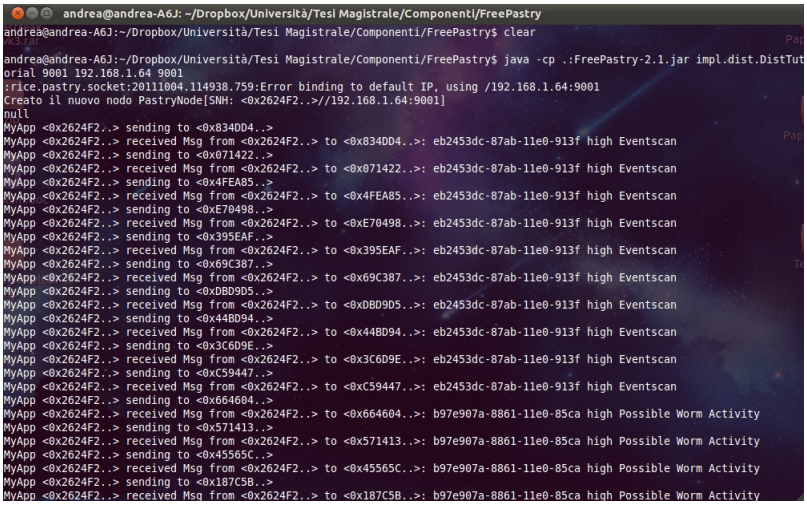

Figura 6.2: *Testing della rete in corso*

Nella figura precedente, possiamo vedere questa modalità di testing in atto. In particolare:

- le prime due righe rappresentano la fase di join nella rete overlay e l'avvenuta creazione di un nuovo nodo del sistema IDS
- le righe successive rappresentano la fase in cui è stata rilevata una possibile intrusione o attacco, e quindi i vari messaggi che il singolo nodo mostrato invia e riceve dai vicini

Possiamo quindi concludere che, dalle prove effettuate in entrambe le modalità, il prototipo realizzato conferma con una buona approssimazione le ipotesi e le aspettative emerse durante la fase progettuale.

Capitolo 7

Conclusioni

In questa tesi si è cercato di dare una visione il più possibile completa, attraverso anche la realizzazione di un prototipo, di un'aspetto della sicurezza informatica. Il progetto realizzato non è da intendersi come sostitutivo agli attuali sistemi in uso, ma porta un nuovo punto di vista che, integrato con le misure attualmente presenti, può rafforzare ulteriormente la nostra sicurezza e quella delle nostre informazioni.

Se pensiamo che fino ad un decennio fa, di sicurezza informatica ne parlavano solo gli enti governativi e le grandi aziende, come la NASA, da allora la situazione è fortemente cambiata. La diffusione del Personal Computer prima, e il successivo avvento di Internet poi, hanno trasformato l'informatica e con essa le nostre vite. I PC sono oggi il principale strumento di comunicazione e come tale, le informazioni in essi custodite contengono traccia della vita di ciascuno dei suoi utilizzatori. Ritengo che a nessuno di noi faccia piacere se i dettagli della propria vita privata vengano resi pubblici; e nella storia, non appena questa convinzione si è radicata negli utilizzatori dei PC, è nata la sicurezza informatica.

Da i primi exploit e attacchi DOS in rete ad oggi, il livello di sicurezza si è estremamente alzato; con questo non vogliamo dire che i sistemi informatici siano tutti ben protetti, ma con la tecnologia e le conoscenze oggi a nostra disposizione, possiamo mettere a dura prova le capacità di un attaccante, magari dissuadendolo nei suoi intenti.

Con l'obiettivo di dare un personale contributo alla sicurezza informati-

ca, ed il profondo interesse per questa disciplina, nasce questa tesi. La sua realizzazione ha richiesto mesi di lavoro, durante i quali sono stati analizzati i punti deboli dei differenti approcci presenti in letteratura e dei prototipi realizzati sulla base di questi ultimi. Dopo di chè, si è cercato di capire come risolvere tali debolezze con le conoscenze acquisite durante gli studi universitari, proponendo l'architettura collaborativa peer-to-peer descritta nei capitoli precedenti. La tesi sfocia infine nella realizzazione di un prototipo con due obiettivi: dimostrare a noi stessi la validità pratica dello studio eseguito, ma soprattutto testare sul campo cosa significa realizzare un sistema per la sicurezza delle nostre informazioni.

Lo studio di tale progetto e le differenti sfaccettature che di volta in volta emergevano, anche nell'analisi del più semplice dei componenti, hanno dimostrato la vastità della sicurezza informatica. E' emersa quindi la coscienza dell'impossibilità di protezione totale; proprio per questo motivo, sono stati scritti e saranno scritti libri e libri su tale argomento. Diffidate da chi vi assicura la protezione totale e certa per i vostri dati, perchè la sicurezza non la si compra, ma la si conquista passo passo, migliorando le proprie conoscenze sui sistemi continuamente in evoluzione. Per citare uno dei più famosi hacker della storia americana, Kevin Mitnick: "Non investite miliardi del vostro denaro in apparecchiature hardware o consulenze informatiche, perchè anche un semplice ragazzino nella propria stanza potrà superarle. E sapete perchè? Perchè lui è mosso dalla curiosità e dalla passione, i due motori principali della conoscenza umana".

Bibliografia

[1] Snort - open source network intrusion prevention and detection system, http://www.snort.org

[2] Nepenthes - vulnerability collection, http://nepenthes.mwcollect.org

[3] OSSEC - open source host-based intrusion detection system, http://www.ossec.net

[4] Prelude - alert aggregator, http://www.prelude-technologies.com/en/welcome/index.html

[5] IDMEF standard, http://www.ietf.org/rfc/rfc4765.txt

[6] FreePastry - framework for p2p overlay network, http://www.freepastry.org

[7] Norman SandBox Information Center, http://sandbox.norman.com

[8] "Sistemi Distribuiti", Andrew Tanunbaum, Maarten Van Steen, PEARSON Edizioni

[9] IETF - The Internet Engineering Task Force, http://www.ietf.org

[10] MySQL, http://www.mysql.it

[11] VMware - virtualizzazione di host http://www.vmware.com/it

[12] "Unix & Internet Security", Simon Garfinkel, Gene Spafford, Alan Schwartz, O'REILLY Edizioni

[13] "L'arte dell'intrusione", Kevin Mitnick, FELTRINELLI Edizioni

[14] “L’arte dell’inganno”, Kevin Mitnick, FELTRINELLI Edizioni

[15] “Il rumore dell’hacking”, Michal Zalewski, APOGEO Edizioni

Ringraziamenti

Al termine di questo lungo percorso, sono d'obbligo dei ringraziamenti.

In primo luogo, desidero ringraziare la mia famiglia, i miei genitori e mio fratello, che durante questi cinque anni mi hanno sempre sostenuto, confortato e, alle volte, anche sfidato per spingermi a dare il meglio di me in ogni momento. Grazie.

Doverosi sono i ringraziamenti a Linda, che in questi ultimi anni mi ha supportato e sopportato, nei momenti più o meno difficili della carriera universitaria.

Come poi non ringraziare gli amici di sempre, Angela, le tre Federiche, Vittorio, Silvia, Marco, Davide, Luca, Enrico, Cristopher, Giulia, Matteo, Elena, Francesca, Vanio e Alessandro, con i quali ho condiviso i momenti di svago e divertimento, utilissimi per ricaricarsi dalle fatiche quotidiane e per affrontare la vita con una buona dose di ironia.

Voglio sentitamente ringraziare il Prof. Carlo Ferrari, che mi ha seguito e guidato duranto lo svolgimento di questa tesi; un grazie particolare alla sua infinita pazienza e disponibilità, che lo ha portato ad incontrarmi anche quando le strutture universitarie erano chiuse per le vacanze estive.

Infine, un grazie di cuore a tutti quelli che durante la mia vita ho incontrato nelle più disparate situazioni, che hanno contribuito a formare l'uomo che sono oggi. Grazie.