# UNIVERSITA' DEGLI STUDI DI BARI

FACOLTA' DI SCIENZE MM. FF. NN.

*Corso di Laurea in Informatica*


---

# *Honeypot:*

# *un diverso approccio alla sicurezza informatica*


RELATORE:
*Prof. Emanuele COVINO*

LAUREANDO:
*Mauro CICOLELLA*


---


Anno Accademico 2007-2008

# Premessa

La sicurezza rappresenta uno dei più importanti capisaldi dell'informatica, soprattutto da quando la diffusione delle reti di calcolatori e di Internet in particolare ha permesso la realizzazione di un vero villaggio tecnologico globale con tutti i rischi connessi. Le esperienze maturate negli anni hanno dimostrato che la sicurezza assoluta di un sistema informatico è un'utopia, nel senso che non si riuscirà mai ad ottenere una protezione assoluta dovendo bilanciare opportunamente costi e funzionalità; al contrario la sicurezza può essere definita come riduzione del rischio, ovvero l'adozione di tutta una serie di misure volte a limitare la probabilità che un attacco possa andare a segno. Molti sono i pericoli informatici a cui si é esposti: virus, worm, exploit, e molto altro. Finora si è seguito un approccio alla sicurezza di tipo passivo: si innalzano delle barriere di protezione (firewall, antivirus ecc.) nella speranza che siano sufficienti a contenere un attacco e solo a seguito di un'intrusione, si corre ai ripari, lasciando in questo modo l'iniziativa all'aggressore.

L'idea alla base degli honeypot è diametralmente opposta e potremmo sintetizzarla con un'antica massima degli strateghi militari: ***"Conosci il tuo nemico per affrontarlo al meglio"***. In parole semplici si tratta di studiare, capire, conoscere il modus operandi del possibile aggressore per essere pronti ad affrontarlo opportunamente. Ciò può essere ottenuto usando dei sistemi trappola, per l'appunto gli ***honeypot*** (letteralmente "***barattolo di miele***"), appositamente creati per essere compromessi dagli aggressori, monitorando contemporaneamente le loro attività.

Questa tesi si propone di fornire una panoramica dei concetti alla base di tale tecnologia, dei suoi vantaggi e svantaggi, delle applicazioni nel campo della sicurezza. In particolare è descritto il prototipo di un **"Wireless Honeypot"** realizzato esclusivamente con l'impiego di soluzioni open source.

# Ringraziamenti

Al termine di questo lungo cammino di studi è doveroso rivolgere alcuni ringraziamenti:

- in primo luogo ai miei genitori che, pur essendo mancati molti anni fa, spero mi abbiano comunque seguito nel tempo e ai quali dedico il frutto del mio lavoro;
- alla mia famiglia per il sostegno che mi ha consentito di andare avanti anche nei momenti più bui;
- agli amici più cari, pochi ma buoni;
- a tutti i docenti che hanno contribuito alla mia formazione universitaria ed in particolare al relatore, Prof. Emanuele Covino, per avermi consentito di sviluppare questa tesi, lasciandomi ampia autonomia ma fornendomi preziose indicazioni per la sua realizzazione.

# Indice

*“Tutti sanno che una cosa è impossibile*
*da realizzare, finchè arriva uno*
*sprovveduto che non lo sa e la inventa”*
*(A. Einstein)*

# 1. Sicurezza informatica

La sicurezza è da sempre legata al concetto di protezione dei dati che hanno valore per coloro che li utilizzano e la cui perdita può rappresentare un danno economico e d'immagine soprattutto per le aziende che su di essi basano il proprio business.

## 1.1 Definizione

Secondo la norma ISO 17799[1]: "***un sistema informatico é considerato sicuro quando è in grado di garantire determinati requisiti di sicurezza in termini di disponibilità, integrità, riservatezza e autenticità***".

Vediamo il significato di queste quattro parole chiave:

- **disponibilità**: il sistema deve garantire la disponibilità delle informazioni agli utenti autorizzati nei tempi e nei modi previsti dalla policy aziendale;
- **integrità**: il sistema deve impedire e segnalare qualsiasi tentativo di manomissione dei dati da parte di persone non autorizzate o determinato da eventi casuali;
- **riservatezza**: le informazioni devono essere accessibili esclusivamente agli utenti autorizzati;
- **autenticità**: il sistema deve garantire la paternità delle informazioni.

Gli obiettivi della sicurezza sono normalmente descritti dal cosiddetto paradigma ***C.I.D.*** (dalle iniziali di ***confidenzialità***, ***integrità*** e ***disponibilità***). Tale paradigma è di difficile realizzazione, per cui un sistema informatico non sarà mai perfettamente sicuro e nessuna politica di sicurezza potrà impedire il verificarsi prima o poi di un'intrusione, sia che venga condotta dall'esterno sia che venga perpetrata dall'interno in maniera più subdola. I sistemi reali sono assai complessi e costituiti da molte componenti in stretta

[1] http://www.iso.ch

correlazione. Inoltre la sicurezza si scontra con il bisogno di garantire la piena fruibilità delle risorse, per cui occorre raggiungere una sorta di compromesso dal momento che una protezione assoluta implicherebbe delle limitazioni tali da negare i servizi persino agli utenti autorizzati. Non a caso si dice che un sistema realmente sicuro è solo quello spento e scollegato dalla rete. In più nella maggior parte dei casi il software utilizzato (dai sistemi operativi agli applicativi in generale) viene acquistato da terze parti, non potendo sostenere i costi di uno sviluppo interno. In questo modo si introduce un grave rischio perché non disponendo dei sorgenti è praticamente impossibile controllare il comportamento del codice, che può presentare vulnerabilità ed esporre a minacce di ogni tipo. Non da ultimo bisogna considerare che i costi della sicurezza devono essere opportunamente bilanciati in rapporto al valore delle risorse da proteggere.

## 1.2 Tipologia di aggressori e loro motivazioni

La compromissione di un dato può assumere vari aspetti a seconda dei casi e più precisamente si parla di:

- **modifica**: quando ne viene alterata l'***integrità***;
- **generazione**: quando vengono aggiunti dati falsi compromettendo l'***autenticità***;
- **intercettazione**: quando il dato viene intercettato facendo venir meno la ***riservatezza***;
- **interruzione**: quando il dato viene distrutto e ciò ne impedisce la ***disponibilità***.

Generalmente le varie tipologie di attacco sfruttano un elemento comune, la ***vulnerabilità***, cioè un imprevisto o un errore di progettazione e/o configurazione in grado di garantire ad un intruso l'accesso al sistema. Nonostante venga utilizzata erroneamente come sinonimo di pirata informatico, soprattutto dai media, la parola ***hacker*** denota una persona in grado di padroneggiare le tecnologie informatiche senza un fine necessariamente ostile. In realtà occorrerebbe adottare il termine ***cracker*** per indicare il vero pericolo, ovvero coloro che si introducono nei sistemi informatici altrui

per provocare danni. Tuttavia vanno distinti ulteriormente quelli che agiscono per puro divertimento, dai criminali informatici professionisti. In generale possiamo concordare sull'utilizzo di un unico sostantivo, ***attacker***, per denotare chiunque violi un sistema informatico. Un altro aspetto molto importante da valutare riguarda le motivazioni che spingono gli aggressori ad attaccare un sistema: alcuni sono mossi dalla curiosità o da semplice vanteria con gli amici, altri sono orientati al furto di informazioni o al danneggiamento di aziende e/o organizzazioni, altri ancora agiscono per motivazioni di carattere ideologico.

## 1.3 Categorie della sicurezza

Secondo il punto di vista di Bruce Schneier, uno dei massimi esperti in questo campo, "***la sicurezza è riduzione del rischio***" e poggia su tre pilastri fondamentali:

- **prevenzione**: chiudere qualunque breccia nel perimetro di sicurezza prima che venga sfruttata, ad esempio identificando le vulnerabilità in un sistema e adottando delle opportune contromisure come meccanismi di autenticazione robusti (password, smart-card, sistemi biometrici), firewall, crittografia.
- **rilevazione**: scoprire la presenza di una falla di sicurezza o un attacco in corso, identificarne la natura e possibilmente l'identità degli aggressori. In questo caso si può ricorrere ad ***Intrusion Detection System***, analisi dei file di log, honeypot ecc.
- **reazione**: limitare le conseguenze di un attacco reagendo opportunamente ad esempio con il blocco del sistema vittima, o reindirizzando gli attacchi verso opportuni sistemi trappola in cui possano essere contenuti.

Tutto ciò rientra nel processo di analisi dei rischi, dove vengono identificate minacce e vulnerabilità, effettuate delle valutazioni in termini quantitativi e approntate le opportune contromisure. Si tratta di una attività alquanto difficile e costosa che richiede un notevole sforzo in termini di risorse, ma che ripaga certamente nel lungo termine.

## 1.4 Principali attacchi informatici

Vediamo di seguito una breve panoramica dei principali attacchi a cui sono esposti i sistemi informatici. Pur non essendo una trattazione esaustiva possiamo farci un'idea della vastità di problematiche da affrontare.

### 1.4.1 Exploit

Un ***exploit*** è una tecnica in grado di sfruttare un bug o una vulnerabilità di un sistema informatico per acquisire privilegi o addirittura prenderne il controllo. Esistono diversi modi per classificarli tra i quali la modalità con cui viene contattata l'applicazione target: un exploit remoto viene condotto attraverso la rete e non richiede precedenti accessi al sistema, mentre un exploit locale necessita di un accesso fisico alla macchina. Un'ulteriore classificazione tiene conto del tipo di vulnerabilità sfruttata (ad es. buffer overflow, format string attacks, ecc.)

La maggior parte di essi mira all'acquisizione dei privilegi di ***root*** (amministratore) per effettuare operazioni che non sono consentite ai normali utenti. Generalmente un exploit può sfruttare solo una specifica vulnerabilità e nel momento in cui questa viene individuata e corretta attraverso il rilascio di una opportuna patch, l'exploit stesso perde ogni efficacia. Per questo motivo molti pirati informatici cercano di non divulgare le loro scoperte (chiamate in gergo "***zero-day exploit***") per sfruttarle il più a lungo possibile. A parte l'utilizzo illecito che può esserne fatto, bisogna sottolineare che gli exploit sono estremamente utili perché rappresentano la prova evidente dell'esistenza di un potenziale problema o di un difetto di sicurezza all'interno di un sistema informatico e stimolano di conseguenza l'adozione di opportuni correttivi.

### 1.4.2 Overflow

Generalmente si distinguono due tipologie di attacco basate sull'***overflow***: il ***buffer*** e lo ***stack overflow***. Il primo è basato su un difetto del programma, che in presenza di una quantità di dati maggiore di quella attesa, produce una sovrascrittura di aree di memoria contigue. Di conseguenza il programma può dare risultati errati o imprevedibili, bloccarsi o bloccare il computer (in caso di coinvolgimento di driver di periferiche o di componenti del sistema operativo stesso). Questa debolezza dei

programmi è nota da molto tempo, ma solo di recente la sua conoscenza si è diffusa tanto da permettere anche ai dilettanti di sfruttarla per bloccare o prendere il controllo di altri computer collegati in rete.

Un caso del genere si può verificare quando il programma non controlla in anticipo la dimensione dei dati in input, ma si limita a scrivere il loro valore in un buffer di lunghezza prestabilita, affidandosi alla buona fede dell'utente. Quando, per errore o per dolo, la quantità di dati supera la capienza del buffer destinato a contenerli, i dati in eccesso vanno a sovrascrivere le variabili interne del programma o di altri processi in esecuzione. Non tutti i programmi sono vulnerabili a questo tipo di inconveniente. In genere sono necessarie le seguenti condizioni:

- il programma deve prevedere un input di lunghezza variabile e non nota a priori;
- tali dati devono essere allocati in aree di memoria contigue ad altre strutture e risultare di vitale importanza per il programma stesso;
- il programmatore non deve aver previsto alcun meccanismo di validazione dell'input.

Naturalmente è facile verificare la prima condizione, mentre le altre sono strettamente legate alla correttezza del software.

Lo ***stack overflow*** consiste nella sovrascrittura dell'area dati del programma, ma questa volta non è causata dall'input ma dall'attività del programma stesso. In particolare può verificarsi in presenza di funzioni ricorsive che ad ogni chiamata memorizzano le informazioni di stato in un'apposita area di memoria denominata ***stack***. Quando quest'ultimo è completamente riempito, in mancanza di opportuni controlli, vengono sovrascritte altre locazioni di memoria, causando gli stessi effetti visti per il buffer overflow.

### 1.4.3 Backdoor

Le ***backdoor*** ("***porte di servizio***") rappresentano degli accessi privilegiati in grado di superare le procedure di sicurezza attivate in un sistema informatico. La loro creazione da parte degli amministratori consente una più agevole opera di manutenzione dell'infrastruttura informatica, ma spesso è determinata da attacker intenzionati a

penetrare nel sistema senza essere rilevati. Possono anche essere installate autonomamente da alcuni malware (come virus, worm o trojan), in modo da consentire il controllo remoto della macchina senza l'autorizzazione del proprietario. Di solito l'accesso ottenuto in questo modo consente di sfruttare il sistema per il lancio di attacchi di tipo DoS.

### 1.4.4 Port scanning

Il ***port scanning*** è una tecnica utilizzata per raccogliere informazioni sui computer connessi in rete. Letteralmente significa "***scansione delle porte***" e consiste nell'inviare richieste di connessione al computer bersaglio utilizzando pacchetti TCP, UDP e ICMP modellati ad hoc per stabilire quali servizi di rete siano attivi. Una porta si dice "***in ascolto***" ("***listening***") o "***aperta***" quando vi è un servizio o programma che la usa. Più in dettaglio si può stabilire se una porta è:

- **aperta**: l'host ha inviato una risposta indicando che un servizio è in ascolto su quella porta;
- **chiusa**: l'host ha inviato una risposta indicando che le connessioni alla porta saranno rifiutate;
- **filtrata**: non c'è stata alcuna risposta dall'host, ad esempio in presenza di un firewall.

Di per sé il port scanning non è pericoloso per i sistemi informatici, e viene comunemente usato dagli amministratori nelle procedure di controllo. Rivela però informazioni dettagliate che potrebbero essere usate per predisporre un piano di attacco sfruttando la conoscenza dei servizi attivi e documentandosi su eventuali vulnerabilità note. L'utilizzo di un firewall ben configurato può consentire tuttavia il corretto funzionamento dei sistemi impedendo in tutto o in parte la scansione delle porte e privando gli attacker di uno strumento estremamente utile.

### 1.4.5 Sniffing

Il termine ***sniffing*** indica l'attività di intercettazione dei dati che viaggiano attraverso una rete di comunicazione. Tale attività può essere svolta per scopi legittimi (ad esempio l'individuazione di problemi di connessione) ma anche per scopi illeciti

(intercettazione di password o altri dati sensibili). I prodotti software utilizzati per eseguire queste attività sono detti ***sniffer*** e tra le loro funzionalità offrono una completa analisi del traffico stesso, oltre ad operare in modo del tutto trasparente.

### 1.4.6 Spoofing

Un attacco ***spoofing*** (letteralmente "***falsificazione***") si verifica quando una persona o un programma altera la propria identità elettronica spacciandosi per qualcun altro, acquisendo così privilegi in maniera illegittima. Esistono varie forme di spoofing: in primo luogo l'***IP spoofing*** permette di alterare l'indirizzo IP per mascherare la sorgente di un attacco o per superare alcuni sistemi di autenticazione basati su **ACL** (***Access Control List***). Un altro tipo di spoofing è il ***webpage spoofing***, noto più comunemente come ***phishing***. In questo caso un sito legittimo, es. di una banca, viene riprodotto fedelmente su un altro server, con lo scopo di far inserire agli utenti dati sensibili (userid e password) per carpirli in modo del tutto trasparente, sfruttando gli account ottenuti per azioni illecite.

### 1.4.7 Trojan horse

Un ***trojan horse*** ("***cavallo di Troia***") è un software che deve il suo nome al fatto di celarsi all'interno di un programma apparentemente utile, per cui è proprio l'utente inconsapevolmente ad installarlo sulla propria macchina. In questo modo è possibile ottenere il controllo remoto di un sistema in maniera del tutto trasparente. Normalmente un trojan consta di 2 file:

- il ***server***, che viene installato sulla macchina vittima;
- il ***client***, usato per pilotare a distanza il sistema target.

I trojan non sono in grado di diffondersi autonomamente, per cui è necessario ingannare la vittima per spingerla ad installare i programmi fraudolenti. Spesso vengono utilizzati come alternativa ai worm e ai virus per installare delle backdoor o dei keylogger.

### 1.4.8 Virus

Per ***virus informatico*** si intende una porzione di codice in grado, una volta eseguito, di infettare altri file in modo da riprodursi facendo copie di se stesso, senza farsi rilevare dall'utente.

I virus teoricamente potrebbero essere innocui, ma in realtà comportano comunque un certo spreco di risorse in termini di RAM, CPU e spazio sul disco fisso e la loro diffusione nasconde sempre uno scopo ostile. Generalmente un virus può danneggiare in modo diretto solo il software della macchina che lo ospita, anche se può indirettamente provocare danni all' hardware, ad esempio causando il surriscaldamento della CPU mediante overclocking, oppure bloccando la ventola di raffreddamento o ancora agendo sul movimento delle testine degli hard disk.

#### 1.4.8.1 Tipologie di virus

Sebbene accomunati da caratteristiche simili, è possibile denominarli in maniera particolare in base a proprietà specifiche soprattutto in merito alle modalità di diffusione e alla tipologia dei file ospite. In dettaglio parliamo di:

- **virus polimorfici**
  dispongono di una routine di mutazione che permette di modificare il proprio codice ad ogni nuova infezione in modo da renderne più difficoltosa l'individuazione da parte dei software antivirus;
- **virus eseguibili**
  attaccano i file di programma con estensione .EXE e .COM, ma anche altri file necessari per applicazioni specifiche con estensione .SYS, .DLL. Sono ormai abbastanza rari a causa della diffusione dei sistemi Windows che hanno soppiantato il vecchio DOS;
- **boot virus**
  sostituiscono i dati contenuti nel *MBR* (***Master Boot Record***) con il proprio codice, in modo da essere eseguiti automaticamente ad ogni avvio del computer, senza lasciare traccia nella procedura di boot;

- **macro virus**
  possono essere contenuti generalmente in un documento ***Office*** e sono costituiti da una macro[2] in grado di diffondersi a tutti i documenti che vengono aperti con quella particolare applicazione;
- **retrovirus**
  si annidano all'interno dei programmi antivirus, riuscendo a disabilitarli. Il nome deriva dai retrovirus biologici, in grado di attaccare il sistema immunitario (come, ad esempio, l'HIV);
- **virus multipiattaforma**
  sono stati fatti molti tentativi per creare virus capaci di infettare differenti sistemi operativi su una stessa architettura hardware, ma si sono rivelati degli insuccessi o hanno avuto un successo molto limitato.

### 1.4.9 Worm

Il termine ***worm*** (in inglese "***verme***") fu coniato per la prima volta negli anni settanta da John Brunner nel suo romanzo di fantascienza intitolato "***The shockwave raider***", ma occorre arrivare al 3 novembre 1988 per vederlo balzare alla ribalta della cronaca. In quella occasione veniva diffuso un prototipo di worm realizzato dallo studente Robert Morris, che sfruttando tre diversi tipi di vulnerabilità dei sistemi Unix, aveva realizzato in linguaggio C un programma in grado di autoreplicarsi e diffondersi attraverso la rete.

In effetti un worm é simile ad un virus, ma a differenza di quest'ultimo non necessita di un file ospite a cui agganciarsi per agire e diffondersi sebbene provveda a modificare il computer che infetta, in modo da essere eseguito ad ogni avvio rimanendo costantemente attivo. Possiamo paragonarlo ad un agente infettivo capace di diffondersi attraverso le reti sfruttando il TCP/IP oppure altri protocolli. Il vettore di infezione più comune è la posta elettronica: il programma ricerca indirizzi e-mail memorizzati nel computer infetto ed invia una copia di se stesso come file allegato (attachment) a tutti o parte degli indirizzi individuati. I messaggi contenenti il worm utilizzano spesso tecniche di social engineering per indurre il destinatario ad aprire l'allegato, che spesso

[2] sequenza di istruzioni eseguibili utilizzate per automatizzare delle procedure di uso comune.

usa una estensione in grado di camuffare il worm come file non eseguibile. Un altro canale è rappresentato dai circuiti di file sharing dove si nascondono tra i file condivisi dall'utente vittima, spacciandosi per programmi di utilità o per crack di programmi molto costosi o ricercati, in modo da indurre altri utenti a scaricarlo ed eseguirlo. Infine ci sono worm che sfruttano le vulnerabilià del software e si diffondono automaticamente ai pc connessi in rete.

### 1.4.10 DoS e DDoS

Acronimo di ***denial of service***, (letteralmente "***negazione del servizio***"), ovvero l'impossibilità da parte di un server di soddisfare le richieste provenienti dagli utenti legittimi a causa dell'eccessiva mole di richieste generata dall'attacker. In questo modo si provoca la saturazione della banda e l'esaurimento delle risorse di calcolo, causando addirittura il crash del server.

Una variante è il ***DDoS*** (***Distributed Denial of Service***) dal funzionamento identico ma realizzato sfruttando delle macchine di ignari utenti (denominate "***zombie***") controllate attraverso worm o trojan horse per concentrare un attacco congiunto su un unico bersaglio. Le tecniche più comuni per realizzare questi attacchi sono:

- ***smurf***: si invia un falso messaggio a molti sistemi contenente come indirizzo del mittente l'IP dell'obiettivo. In questo modo le macchine contattate rispondono in massa all'indirizzo sovraccaricando il server corrispondente;
- ***banana attack***: si indirizzano i messaggi in uscita da un sistema al sistema stesso, bloccando di fatto l'accesso esterno e provocando un sovraccarico;
- ***flood***: è il più classico e si basa sulla creazione di un flusso di richieste di servizio soprattutto sfruttando i protocolli TCP e ICMP.

### 1.4.11 Hijacking

Consiste nel prendere il controllo di una comunicazione fungendo da intermediario non autorizzato. Si parla in gergo di ***man in the middle*** quando l'aggressore riesce ad intercettare, modificare e trasmettere messaggi tra due nodi in modo del tutto trasparente.

## 1.5 Intrusion detection system

Nonostante gli sforzi profusi non si può rendere un sistema informatico assolutamente sicuro, per cui risulta indispensabile disporre di un meccanismo per la rilevazione degli attacchi che non si è riusciti ad evitare. Il primo ad occuparsi di ***intrusion detection*** fu J.P. Anderson, che negli Anni '80 propose alcuni metodi, ancora oggi utilizzati, basati sull'analisi del comportamento tipico di un utente che interagisce con il computer e che sono basati su due paradigmi fondamentali:

- **anomaly detection**
- **misuse detection**

Nel primo caso si suppone che un attacco consista in un evento anomalo, per cui risulta indispensabile definire esattamente cosa si intenda per eventi normali. In tale approccio si ritiene che gli stessi corrispondano alle attività svolte con maggiore frequenza, assumendo implicitamente l'equazione **eventi rari = attacchi**. In effetti ciò non è sempre vero perché spesso gli eventi rari sono assolutamente innocui mentre le reali minacce possono celarsi dietro attività molto comuni che non destano particolare attenzione.

Viceversa il paradigma ***misuse detection*** individua in partenza l'insieme dei possibili attacchi, riducendo la probabilità di ***falsi positivi***, ma potendo individuare solo quelli effettivamente codificati aumenta il rischio di ***falsi negativi***, cioè attacchi non rilevati.

Confrontando i due approcci possiamo dedurre che nessuno dei due è superiore all'altro bensì sono complementari. L'approccio anomaly detection sembrerebbe il più efficace perché riduce il tasso di falsi negativi, ma paradossalmente l'elevato numero di falsi positivi prodotti può celare gli attacchi reali sia per la mole di dati da analizzare sia perché può indurre psicologicamente ad ignorare tali allarmi perché ritenuti infondati. Oltretutto si rischia di impedire attività poco frequenti ma assolutamente legittime. In conclusione si può affermare che questo approccio è adottabile solo se il costo di un attacco è di gran lunga superiore ai costi di gestione degli IDS. Da un punto di vista pratico si usa il paradigma misuse detection oppure un approccio misto (o ibrido), mentre quasi nessuno adotta l'anomaly detection.

Con il termine di ***IDS (Intrusion Detection System)*** si è soliti raggruppare una varietà di soluzioni tecnologiche che assumono denominazioni differenti in base al tipo di

implementazione adottata, ma accomunate dalla caratteristica di rilevare i segni di una intrusione, cioè di una qualsiasi violazione della politica di sicurezza. Se l'IDS è collocato direttamente sull'host oggetto del monitoraggio si parla di ***Host IDS (HIDS)***; se rappresenta un nodo autonomo sulla rete si parla di ***Network IDS (NIDS)***. Infine in caso di combinazione delle precedenti due soluzioni si parla di ***Hybrid*** o ***Stack-based IDS***. Normalmente gli IDS si limitano all'acquisizione di informazioni da utilizzare successivamente per pianificare delle opportune contromisure. In alcuni casi possono intraprendere delle azioni, affidandosi il più delle volte ad applicazioni esterne, in risposta ai tentativi di aggressione.

## 1.6 La normativa nel campo della criminalità informatica

A margine di questo capitolo ci proponiamo di esaminare brevemente la legislazione in materia di criminalità informatica. Prima di tutto osserviamo che un crimine informatico può manifestarsi in due modalità differenti:

- reati commessi contro un sistema informatico
- reati commessi utilizzando un sistema informatico.

Poiché tali attività possono essere condotte attraverso la rete Internet, che ha una copertura a livello mondiale e gode di una sorta di extraterritorialità, è praticamente impossibile definire una normativa assoluta valida universalmente, bensì occorre definire degli standard a cui adattare ciascuna legislazione nazionale. In questa direzione è stata sottoscritta a Budapest nel 2001 da parte di 29 stati la ***Convenzione sul Cybercrime***, che fornisce delle direttive per le forze dell'ordine in materia di crimini informatici. L'Italia si è interessata al problema modificando le norme già esistenti nel codice penale, ad esempio con l'estensione del concetto di violenza sulle cose (previsto dall'art. 1 della legge 547/93) anche ai beni informatici. Le pene previste a seconda della gravità della violazione includono la reclusione fino a 8 anni e pesanti sanzioni amministrative. Inoltre viene attribuito al supporto informatico lo stesso valore di qualunque altro documento cartaceo o di altra natura, riconoscendone l'inviolabilità del contenuto.

# 2. Honeypot e honeynet

## 2.1 Definizione di honeypot

Il termine ***honeypot*** (letteralmente "***barattolo di miele***") esprime al meglio l'idea di fondo alla base di questa tecnologia, ovvero la realizzazione di un sistema in grado di attrarre i potenziali aggressori per osservarli in azione.

Una definizione ampiamente utilizzata e condivisa è stata fornita da Lance Spitzner[1], massimo esperto in questo campo, secondo il quale

***"un honeypot è una risorsa il cui valore risiede nell'essere rilevata, attaccata e compromessa***".

Il punto non è dunque la specifica implementazione adottata, ma gli obiettivi che si vogliono raggiungere e che in sintesi sono:

- disporre di un sistema in grado di essere rilevato, attaccato e compromesso;
- registrare tutte le attività intrusive per acquisire informazioni sugli strumenti e le strategie impiegate.

Un honeypot può fungere da potenziale obiettivo per un attacco, non avendo alcun valore in quanto privo di informazioni di vitale importanza e i cui servizi in esecuzione non sono parte integrante di un reale ambiente di produzione. Come vedremo in seguito, ciò si traduce in un enorme vantaggio perché tutto il traffico che lo riguarda è da considerarsi potenzialmente ostile ed essendo quantitativamente limitato consente di raccogliere informazioni di grande utilità.

[1] http://www.tracking-hackers.com

## 2.2 Cenni storici

Gli honeypot non costituiscono una novità assoluta di oggi, ma già negli Anni '90 diverse pubblicazioni ne avevano illustrato le idee di base. In particolare nel 1988 fu pubblicato il primo testo sull'argomento intitolato ***The Cuckoo's Egg***. L'autore Clifford Stoll, astrofisico responsabile della gestione dei sistemi presso il Lawrence Berkeley Lab, aveva casualmente scoperto la presenza di un intruso che, dopo essere penetrato nel sistema, cercava di attaccare altre risorse informatiche alla ricerca di segreti militari per conto del KGB (servizio segreto russo). Invece di denunciarlo, Stoll aveva seguito e documentato le sue azioni per ben tre anni acquisendo delle prove tali da farlo arrestare, ma nello stesso tempo imparando molto sulle strategie di attacco utilizzate.

Un'altra interessante pubblicazione, considerata una pietra miliare nel campo degli honeypot, fu ***An Evening with Berferd*** dell'esperto informatico Bill Cheswick, in cui veniva descritto un esperimento fatto dall'autore e dai suoi colleghi per attirare i pirati informatici in un sistema trappola appositamente predisposto (chiamato ***roach motel***) per osservare e documentare tutti i loro movimenti.

Vediamo adesso una breve cronologia degli eventi più significativi:

- **1997**

  Viene rilasciato il ***Deception Toolkit***[2], un insieme di script in Perl in grado di riprodurre le più comuni vulnerabilità dei sistemi UNIX. Per la prima volta si utilizza il termine di ***deceptive defense*** (difesa basata sull'inganno) che oggi costituisce un elemento cardine per la tecnologia degli honeypot. L'idea è di simulare l'output atteso da un potenziale aggressore per spingerlo ad interagire con il sistema, ritenendo di averlo bucato, impegnando le sue risorse nel tentativo di sfruttare queste false vulnerabilità e avendo di conseguenza il tempo di individuare l'attacco ed escogitare una opportuna contromossa.

- **1998**

  Vede la luce la prima versione commerciale di un honeypot denominato ***Cybercop Sing*** definito alternativamente come ***decoy server*** per la sua capacità di simulare una rete contenente differenti sistemi inclusi server Windows NT, Unix e anche router. Tutti i dispositivi sono in grado di analizzare e riportare

[2] http://all.net/dtk/dtk.html

qualsiasi attività intrusiva. Tuttavia il prodotto soffre dei problemi tipici delle soluzioni a bassa interazione e cioè la possibilità di simulare un numero limitato di servizi. Un altro prodotto commerciale è ***NetFacade***, che riesce a simulare reti con molti host (un network di classe C fino a 254 host) e ben 7 differenti sistemi operativi con i relativi servizi attivi. Appare sulla scena ***Back Officer Friendly***[3], prodotto gratuito per piattaforma Windows, la cui popolarità nasce dal semplice approccio con cui riesce a spiegare i concetti di base degli honeypot al vasto pubblico.

- **1999**

  Lance Spitzner crea l'***Honeynet Project***[4], un gruppo non-profit impegnato nella ricerca sulla comunità dei pirati informatici e la condivisione dei risultati ottenuti. Viene elaborato il concetto di ***honeynet***.

- **2002**

  Parte una nuova iniziativa denominata ***Honeynet Research Alliance***[5] volta a coinvolgere l'intera comunità di esperti in sicurezza informatica e a cui prendono parte gruppi di tutto il mondo interessati alla ricerca nel campo degli honeypot.

- **2003**

  Vengono rilasciati molti strumenti utili nella gestione di honeypot e honeynet:

  - **Snort-inline**: una variante del noto IDS Snort in grado di bloccare o rallentare gli attacchi invece di limitarsi alla loro rilevazione;
  - **Sebek:** tool per la cattura di informazioni sulle attività intrusive;
  - **Honeynet virtuali:** consentono l'esecuzione di più honeynet su una stessa macchina fisica.

- **2004**

  Grazie alla collaborazione tra Honeynet Project e Honeynet Research Alliance viene sviluppato il CDROM ***Roo***, che consente l'installazione automatizzata di una honeynet completa.

[3] http://www.nfr.com/resource/backOfficer.php
[4] http://www.honeynet.org
[5] http://www.honeynet.org/alliance

- **Dal 2005 ad oggi**

  Si susseguono varie iniziative volte alla ricerca nel campo di honeynet e honeypot e allo sviluppo di nuovi tool sempre più potenti e semplici da utilizzare.

## 2.3 Classificazione

Gli honeypot possono avere molteplici impieghi, ma comunemente si considerano due principali categorie:

- **sistemi di produzione**: utilizzati nelle reti aziendali per incrementare il livello di sicurezza;
- **sistemi di ricerca**: impiegati per acquisire informazioni sulla comunità dei pirati informatici e scoprire eventuali vulnerabilità ignote, nuove tecniche di attacco e di conseguenza per mettere a punto nuovi meccanismi difensivi.

### 2.3.1 Sistemi di produzione

Utilizzati fondamentalmente per la rilevazione di attacchi diretti alla rete interna, possono essere considerati un'estensione degli ***IDS*** (***Intrusion Detection System***) in grado di verificare l'adeguatezza delle protezioni adottate, in quanto in caso di compromissione si può dedurre che sono stati elusi gli altri meccanismi di sicurezza (es. firewall). La consapevolezza di un attacco, ottenuta in questo modo, può consentire la risoluzione del problema, evidenziando addirittura la presenza di falle di sicurezza del tutto ignote. Allo stesso modo può giustificare ulteriori investimenti in sicurezza che altrimenti, in assenza di attacchi documentati, potrebbero essere ridotti o azzerati nella falsa convinzione di essere completamente al sicuro. Attraverso gli honeypot è possibile mantenere traccia degli attacchi e quindi fare delle statistiche circa la loro tipologia, frequenza ed eventualmente individuare nuove minacce che potrebbero sfuggire ad altri strumenti. Gli IDS ad esempio si appoggiano ad un database di firme (signature) delle varie tipologie di attacco che deve essere costantemente aggiornato.

### 2.3.2 Sistemi di ricerca

Rappresentano una piattaforma molto utile per lo studio dei pericoli informatici in quanto consentono di vedere gli aggressori in azione mentre attaccano e si impadroniscono di un sistema, e nel contempo di valutare strategie e strumenti utilizzati. In generale non consentono la riduzione dei rischi, ma indirettamente le informazioni acquisite possono incrementare il livello di sicurezza. Nella maggior parte dei casi si tratta di sistemi gestiti da università, enti governativi e organizzazioni non-profit con finalità di ricerca (vedi ***The Honeynet Project***[6]).

## 2.4 Livelli di interazione

Il livello di interazione offerto dagli honeypot determina le modalità con cui un aggressore può accedere al sistema e di conseguenza la libertà di azione che gli è consentita, influenzando in maniera diretta la complessità e il rischio connessi alle attività di realizzazione, configurazione e gestione. In generale maggiore è il livello di interazione maggiore sarà la probabilità di compromissione e di impiego del sistema per sferrare attacchi verso altri obiettivi. Tuttavia solo questi sistemi risultano veramente efficaci nel campo della ricerca.

### 2.4.1 Livello basso

Gli honeypot di questo tipo si limitano ad emulare alcuni servizi di rete non consentendo alcuna azione diretta per cui il loro impiego è riservato al rilevamento delle intrusioni nei sistemi reali. Per le sue caratteristiche si presenta come una soluzione a basso rischio per l'ambiente in cui è installata, ma anche facilmente identificabile, ad esempio per la mancanza di traffico in uscita.

### 2.4.2 Livello medio

In questa categoria rientrano gli honeypot che utilizzano, al posto dei reali servizi di rete, script o programmi che ne emulano il comportamento, cercando di essere il più possibile realistici. La principale differenza con quelli del primo tipo risiede nella

[6] http://www.honeynet.org

capacità di generare del traffico in uscita, permettendo un'interazione limitata con il sistema.

### 2.4.3 Livello alto

Si tratta degli honeypot più potenti e complessi in grado di consentire all'aggressore l'accesso al sistema operativo, lasciandolo libero di agire e monitorando le sue attività. Il loro impiego principale è nei sistemi di ricerca, ma possono essere utilizzati anche in quelli di produzione con opportune precauzioni. La disponibilità di un ambiente operativo completo implica un livello di rischio molto elevato, permettendo ad un attacker di compromettere tale piattaforma e utilizzarla per sferrare attacchi verso altri sistemi o saturare la rete con una elevata produzione di traffico.

### 2.4.4 Confronto tra i vari livelli

Un confronto tra le diverse soluzioni proposte deve tener conto di vari aspetti come installazione, fingerprinting, valore dei dati, manutenzione e rischi.
La difficoltà di installazione può essere valutata considerando il tempo e le competenze richieste. In genere i sistemi a bassa interazione implicano maggiore semplicità. Per individuare un honeypot bisogna evidenziare le differenze tra tale sistema e quello reale. Anche in questo caso gli honeypot a bassa/media interazione sono più semplici da scoprire per le difficoltà evidenti di emulazione di servizi e applicazioni. L'utilità di un honeypot si misura in base al valore dei dati acquisiti e certamente i sistemi ad alta interazione offrono spunti più interessanti permettendo la creazione di ambienti molto realistici. I sistemi a bassa/media interazione non richiedono grossi interventi manutentivi poiché i servizi sono semplicemente emulati, a differenza degli honeypot ad alta interazione che si basano su un effettivo ambiente operativo.

Infine sul fronte dei rischi dobbiamo considerare che questi aumentano proporzionalmente al livello di interazione.

| | BASSA/MEDIA | ALTA |
|---|---|---|
| | INTERAZIONE | |
| INSTALLAZIONE | Semplice | Maggiore complessità |
| MANUTENZIONE | Semplice | Richiede tempo |
| RISCHIO | Basso | Alto |
| NECESSITA' DI MONITORAGGIO | No | Si |
| RACCOLTA DATI | Limitata | Ampia |
| INTERAZIONE | Emulazione servizi | Ambiente completo |

**Tabella 2.1 Confronto tra i diversi livelli di interazione**

## 2.5 Posizionamento degli honeypot

Un aspetto non trascurabile è rappresentato dal corretto posizionamento all'interno della rete, che può pregiudicare l'efficacia di tali sistemi.

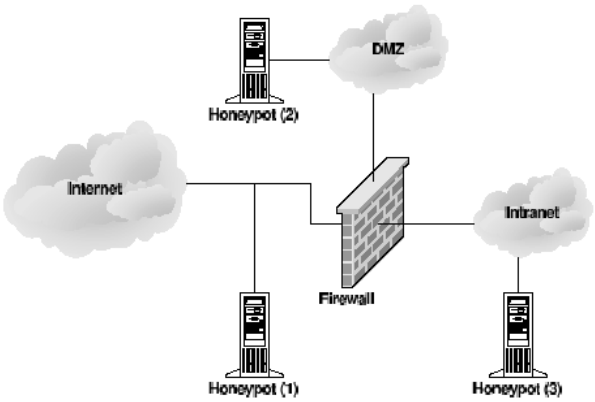



**Figura 2.1 Posizionamento degli honeypot**

Gli honeypot di produzione con funzione di rilevamento agiscono correttamente se posti all'interno del perimetro di sicurezza protetti da un firewall (3). In questo modo possono fungere da sensore di allarme nel caso di attacchi che abbiano aggirato le barriere difensive ed eventualmente possono agire come deterrente per distogliere l'attenzione dalle risorse di valore. Collocarli fuori dal perimetro di sicurezza non avrebbe senso perché li esporrebbe ad un gran numero di attacchi, molti dei quali dovrebbero già essere bloccati dagli altri dispositivi di difesa. Se però gli honeypot di produzione vengono impiegati come meccanismo di reazione, per dirottare gli attacchi provenienti dall'esterno, è conveniente collocarli nella ***DMZ*** (***zona demilitarizzata***,

segmento di rete parzialmente accessibile da Internet) dove normalmente trovano posto i servizi pubblici (ftp, www, smtp ecc.) (2).

Al contrario gli honepot di ricerca vanno posizionati esclusivamente al di fuori del perimetro di sicurezza per acquisire la massima quantità di informazioni sugli attacchi (1). In questo caso, trattandosi di sistemi particolarmente esposti, è molto elevato il rischio di compromissione e di impiego per attaccare altre risorse, per cui è indispensabile adottare delle contromisure adeguate.

## 2.6 Vantaggi e svantaggi

Come per tutte le tecnologie vi sono vantaggi e svantaggi che occorre valutare attentamente affinché tali sistemi abbiano efficacia.

### 2.6.1 Vantaggi

Gli honeypot sono diversi dai comuni tool di sicurezza che risolvono specifici problemi come, ad esempio, i firewall che vigilano sul perimetro di una rete monitorando le connessioni in ingresso e uscita e gestendo attraverso opportune regole il traffico in transito, oppure gli IDS (Intrusion Detection System) che rilevano gli attacchi esaminando in dettaglio i pacchetti trasferiti attraverso la rete.

Al contrario gli honeypot non forniscono servizi di produzione e rappresentano uno strumento altamente flessibile e adattabile a molteplici situazioni. In questo senso non vanno considerati come un surrogato dei classici sistemi difensivi bensì come un loro complemento. Spesso si fa confusione proprio con gli IDS che al contrario degli honeypot ispezionano tutto il traffico di rete confrontando i dati con un database di "***attack signature***" (una sorta di DNA degli attacchi). Quando viene trovata una corrispondenza l'IDS lancia un allarme. Questa metodologia non sempre funziona bene e spesso si generano dei ***falsi positivi***, ovvero un'attività lecita viene erroneamente scambiata per un attacco. Troppi falsi positivi in aggiunta alle dimensioni ragguardevoli dei file di log di sistema e di quelli generati dai firewall rendono estremamente difficoltoso estrapolare dati di un qualche valore per gli addetti alla sicurezza.

Uno dei principali vantaggi degli honeypot è la raccolta di una piccola quantità di dati relativi a interazioni con il sistema. Non fornendo servizi di produzione, tutte le

connessioni sono in teoria non autorizzate e i dati raccolti il risultato di una scansione o di un attacco. In particolare le connessioni in uscita sono il sintomo inequivocabile della compromissione dell'honeypot. Anche il numero di falsi positivi è limitato proprio perché è minore il traffico prodotto. Inoltre un comune IDS può riconoscere solo attacchi già noti e codificati nel database delle signature (in maniera simile a quanto accade per gli antivirus); in tutti gli altri casi si osserva il fenomeno dei ***falsi negativi***, ossia attacchi che sfuggono al riconoscimento.

Gli honeypot al contrario possono servire per la rilevazione di qualsiasi tipo di attacco, inclusi quelli sconosciuti. Inoltre i meccanismi di sicurezza tradizionali devono esaminare tutti i pacchetti che attraversano la rete e in realtà di grandi dimensioni ciò si traduce in un'enorme mole di dati con l'impiego di ingenti risorse. Un honeypot deve catturare solo il traffico ad esso diretto, quindi necessita di una configurazione hardware minima ed ha un costo di implementazione molto basso. Oltretutto è possibile realizzare degli ***honeypot virtuali***, cioè una serie di sistemi che girano su una stessa macchina attraverso l'impiego di software di virtualizzazione (es. VMware, UML) oppure di scripts che simulano una rete di computer (es. Honeyd).

Infine occorre osservare che esistono molte soluzioni open source con tutti i vantaggi che ne derivano e a cui lavora una vasta comunità di ricercatori ed esperti in grado di offrire supporto nelle varie fasi di progettazione, configurazione e gestione.

### 2.6.2 Svantaggi

Il principale svantaggio di un honeypot è rappresentato dal suo "***campo visivo ristretto***". In pratica é in grado di accorgersi esclusivamente degli eventi che lo interessano, perché a differenza di altri dispositivi passivi non può catturare le attività relative ad altri sistemi. Spitzner parla di "***effetto microscopio***", cioè della capacità di ingrandire e mettere a fuoco gli eventi che avvengono al suo interno e ciò va considerato positivamente, sebbene in alcuni casi possa rappresentare un problema. Ipotizziamo due diverse situazioni: nella prima l'aggressore cerca di individuare il potenziale obiettivo adottando le classiche tecniche di scansione della rete. Tale azione viene rilevata dall'honeypot inducendo i responsabili della sicurezza alla verifica degli altri sistemi di produzione che presumibilmente sono stati oggetto del medesimo attacco. In questo caso l'honeypot si dimostra molto valido.

Viceversa ammettiamo che l'aggressore conosca la tipologia della rete e sferri un attacco diretto verso uno specifico obiettivo conoscendone l'IP e quindi senza ricorrere alla scansione della rete. In questa situazione l'honeypot è di fatto tagliato fuori e non aiuta nella rilevazione dell'attacco.

In secondo luogo occorre considerare l'entità del rischio introdotto nella sicurezza di rete, che risulta strettamente legato al livello di interazione offerto di cui abbiamo parlato in precedenza.

Infine bisogna osservare che per risultare veramente efficace un honeypot deve nascondere la propria identità, perché in caso contrario può spingere l'aggressore a trascurarlo per orientarsi verso altri obiettivi impedendo la rilevazione di eventuali attacchi.

| VANTAGGI | SVANTAGGI |
|---|---|
| Limitata quantità di dati da esaminare | Campo visivo ristretto |
| Pochi falsi positivi e negativi | Rischio legato al livello di interazione |
| Configurazione hardware minima | Fingerprinting |
| Costi di implementazione ridotti | |

**Tabella 2.2 Vantaggi e svantaggi degli honeypot**

## 2.7 Impiego degli honeypot nella sicurezza informatica

Vediamo ora il valore che gli honeypot possono aggiungere alle tre aree della sicurezza: prevenzione, rilevamento e reazione.

### 2.7.1 Prevenzione

In quest'area il valore aggiunto é abbastanza limitato, perché la vera prevenzione si ottiene applicando una adeguata policy di sicurezza e cioè disabilitando i servizi inutilizzati, installando tutte le patch di aggiornamento e adottando meccanismi di autenticazione più robusti.

Un possibile utilizzo preventivo è come deterrente verso gli attacchi, pubblicizzandone l'impiego all'interno della rete con lo scopo di far perdere tempo e risorse nel cercare di bucarlo, distogliendo l'attenzione dai sistemi reali. Questo si ottiene attraverso una configurazione adeguata in grado di ingannare i potenziali nemici. Naturalmente questa misura di carattere psicologico (si spera che l'attacker desista dal suo intento sapendo di

poter essere individuato) non funziona con attacchi automatici e worm, ma anche con nemici molto preparati e determinati.

### 2.7.2 Rilevazione

In questa area si ottengono i maggiori vantaggi, in quanto abbiamo già visto che i sistemi di produzione devono fornire servizi a molti utenti generando una grande mole di traffico e log di dimensioni ragguardevoli. Gli honeypot non soffrono di questi problemi perché tutto il traffico che li riguarda può essere considerato ostile e ciò aiuta nella rilevazione di attacchi sconosciuti che potrebbero sfuggire ai comuni IDS in quanto non ancora codificati.

### 2.7.3 Reazione

Anche in questo caso il vantaggio è rilevante e sempre legato alla ridotta quantità di dati da gestire rispetto ad un sistema di produzione. Quando quest'ultimo viene compromesso occorre metterlo offline, rischiando il blocco delle attività. Spesso è difficile isolare le tracce di una compromissione perché inquinate dal traffico di produzione e ciò può rallentare significativamente il riconoscimento di un attacco. Gli honeypot, invece, possono essere disattivati anche immediatamente, esaminati senza interrompere la continuità del servizio e opportunamente programmati per rispondere in maniera attiva a determinati attacchi. Possono fungere da bersaglio e rallentare la diffusione di azioni malevole, consentendo la predisposizione di un'opportuna procedura difensiva.

## 2.8 Limitare i rischi

Qualunque sia la tipologia di honeypot adottata viene comunque introdotto un certo livello di rischio che è auspicabile ridurre al massimo. In primo luogo occorre analizzare a fondo le proprie esigenze per scegliere il livello di interazione più idoneo. Spesso non è necessario ricorrere ad un sistema ad alta interazione, ad esempio se si vogliono semplicemente registrare i tentativi di scansione della propria rete aziendale, limitandosi all'uso di un honeypot a medio-basso livello. La superiorità dei sistemi ad alta interazione risiede nel maggior numero di funzionalità offerte, sebbene occorra

stare molto attenti e quindi è più logico ricorrere ad una honeynet adeguatamente equipaggiata. In secondo luogo ogni segnale di allarme dovrebbe essere inviato ad un responsabile in grado di adottare tempestivamente le opportune contromisure. Inoltre è indispensabile la presenza di una funzione di shutdown da utilizzare come extrema ratio per bloccare qualsiasi attività dannosa – soprattutto attacchi verso l'esterno – qualora sia sfuggita ad ogni altro controllo. Tutti gli accorgimenti devono essere orientati alla riduzione dei rischi sebbene siano richiesti investimenti, tempo e competenze specifiche. D'altro canto produrre danni verso terzi a causa di una leggerezza nella configurazione dei propri sistemi può costare molto caro.

## 2.9 Attrarre i nemici

L'effettiva utilità di un honeypot risiede nella capacità di essere attaccato, quindi deve risultare rintracciabile e attraente per i potenziali aggressori offrendo servizi interessanti. Il problema principale risiede nella decisione su cosa sia da considerare interessante e tale scelta dipende soprattutto dal livello di competenze dell'aggressore. Un dilettante potrebbe limitarsi a cercare la presenza dei servizi più comuni (http, ftp, telnet, ecc) e trascurare quelli più specifici come ssh, finger o applicazioni come ad es. i database. Al contrario questo potrebbe non sfuggire all'occhio attento di un esperto.

## 2.10 Meccanismi di cattura delle informazioni

L'acquisizione di dati su un sistema progettato per essere compromesso deve consentire delle analisi significative delle attività svolte dagli aggressori e deve avvenire in modo del tutto trasparente. In primo luogo sull'***host*** è possibile registrare tutte le connessioni in ingresso e uscita, i comandi impartiti e i processi in esecuzione. Lo svantaggio principale è rappresentato dalla facilità di individuazione sia dei log che dei tool di sicurezza utilizzati che possono essere disabilitati per nascondere le proprie tracce. Una soluzione più complessa consiste nella cattura delle informazioni sull'honeypot e nel loro trasferimento ad un server remoto per un'analisi successiva consentendo una completa archiviazione di tutto il traffico acquisito. Naturalmente

anche tale server deve essere attentamente protetto per evitare che subisca degli attacchi soprattutto in presenza di un evidente flusso dati tra i due sistemi, rendendolo del tutto inefficace. Infine possiamo considerare il gateway la cui funzione principale consiste nel trasferimento dei dati tra gli host della rete e Internet, consentendo la registrazione di tutte le connessioni e del traffico scambiato. Il rischio è rappresentato dal gateway stesso, che essendo visibile in rete, può diventare oggetto di un attacco. Senza opportune tecniche di cattura dei dati la validità degli stessi è estremamente ridotta.

## 2.11 Manutenzione degli honeypot

Per quanto visto finora si può intuire che tali sistemi non andrebbero lasciati alla deriva. Molti svantaggi si possono evitare attraverso un continuo monitoraggio in real-time delle attività in corso. Nonostante tali accorgimenti c'è sempre il rischio che la configurazione adottata non sia sufficiente ad ingannare un intruso e a contenere un attacco. Quando un honeypot viene compromesso deve essere posto offline e analizzato, cosa che può richiedere delle ore per capire cosa sia effettivamente accaduto. Infine occorre ripristinare lo status quo, ovvero resettare il sistema, eliminando tutte le modifiche apportate dagli attacker e sfruttare le informazioni acquisite per migliorarne la configurazione.

## 2.12 Definizione di honeynet

Una honeynet è costituita da una rete di honeypot ad alta interazione che può includere differenti sistemi operativi e applicazioni in grado di riprodurre efficacemente un ambiente di produzione e come tale risulta maggiormente esposta ai rischi di compromissione. Più precisamente diciamo che:

***" una honeynet è una rete di honeypot ad alta interazione che simula una rete di produzione ed è configurata in modo tale da monitorare, registrare e in parte regolare ogni attività svolta al suo interno"***

Anche in questo caso non si tratta di un prodotto, di un software da installare sul proprio computer bensì di una architettura in grado di ricreare un ambiente altamente controllato e il cui fulcro è costituito dalla presenza di un gateway denominato ***honeywall***, che opera in modo simile ad un firewall in una rete convenzionale, ovvero come un filtro che isola la honeynet dal resto della rete in cui è integrata. Mentre in un honeypot il meccanismo di controllo è collocato al suo interno esponendolo al rischio di disattivazione, nelle honeynet tale ruolo è affidato al gateway, risultando del tutto trasparente all'aggressore. L'allestimento di una honeynet, da cui dipende la sua efficacia, prevede che siano soddisfatti almeno due requisiti fondamentali: ***Data Control*** e ***Data Capture***.

## 2.13 Data Control

In primo luogo si richiede l'adozione di opportuni meccanismi per limitare le attività di un intruso, soprattutto riguardo al traffico in uscita a seguito di una compromissione con lo scopo di ridurre i rischi di possibili attacchi verso obiettivi esterni.

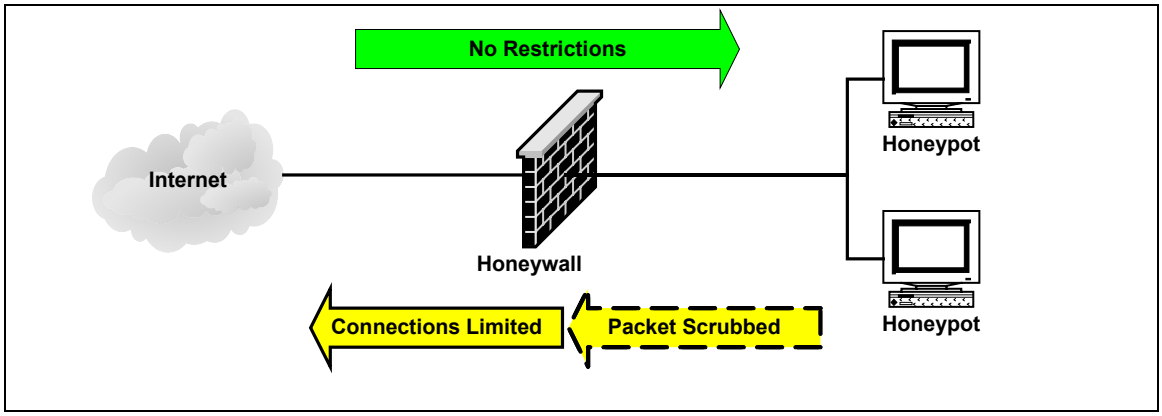



**Figura 2.2 Data Control in una honeynet**

## 2.14 Data Capture

L'acquisizione dei dati relativi alle strategie e ai tool utilizzati dagli attacker deve risultare del tutto trasparente per evitare che il sistema venga smascherato e perda

di conseguenza qualsiasi valore. In secondo luogo bisogna decidere dove memorizzare i dati raccolti. Certamente è sconsigliabile considerare il sistema compromesso perché i pirati informatici possono alterare o addirittura cancellare i log per nascondere le proprie tracce, rendendo di fatto vano tutto il lavoro svolto. La soluzione più logica prevede l'impiego di un server di raccolta verso cui far confluire i dati acquisiti. L'Honeynet Project Alliance ha predisposto un vademecum di suggerimenti da seguire per ottenere buoni risultati.

In primo luogo adottare tutte le precauzioni necessarie per nascondere i processi di data capture. Posizionando l'IDS direttamente sull'honeypot diventa semplice la sua rilevazione e di conseguenza aumenta il rischio che venga disattivato. Per tale motivo nelle honeynet di seconda e terza generazione il meccanismo di data capture viene trasferito su un sistema gateway che opera in modalità bridge, rendendo più difficoltosa la sua individuazione in assenza dello stack IP.

In secondo luogo evitare la contaminazione dei dati raccolti dall'honeypot con qualsiasi attività non-standard definita in gergo ***data-pollution***, ad esempio effettuando dei test interni sul sistema trappola che possano alterare i dati raccolti e falsare eventuali statistiche.

Le informazioni acquisite (log dei firewall, pacchetti di dati, log di sistema) devono essere memorizzate in un formato standard su un sistema separato e sicuro per un periodo di tempo adeguato a consentire le analisi del caso o a preservare le prove digitali in un procedimento giudiziario, per cui risulta conveniente predisporre una procedura di archiviazione automatica da eseguire ad intervalli regolari.

## 2.15 Data Collection

In presenza di varie honeynet logicamente o fisicamente distribuite i dati provenienti da più fonti devono essere raggruppati e memorizzati in modo sicuro in una locazione centralizzata, per essere successivamente analizzati. Sono richiesti degli standard comuni per la loro gestione soprattutto per quanto riguarda la definizione del formato dei file e delle convenzioni su nomi e identificatori univoci per le varie honeynet.

## 2.16 Rischi

Ancora una volta bisogna prestare molta attenzione ai rischi connessi all'uso di questa tecnologia, sebbene sia difficile dare una definizione assoluta di rischio, lasciando a ciascuna organizzazione l'onere di stabilire cosa sia rilevante e quali limiti non debbano essere superati. Nello specifico possiamo individuare quattro aspetti principali di cui tener conto: ***danneggiamento***, ***rilevamento***, ***disabilitazione*** e ***violazione***.

Un aggressore può prendere il controllo della honeynet e utilizzarla per attaccare altri sistemi provocando danni di grave entità, per cui bisogna tenere nel giusto conto questa eventualità, predisponendo vari livelli di Data Control.

Una volta scoperta la vera identità di una honeynet, il suo valore decresce significativamente, in quanto gli aggressori possono ignorarla o bypassarla disabilitandone le funzioni di logging o ancora fornire delle false informazioni.

In questo caso l'attacco si concentra direttamente sulle componenti di Data Control e Data Capture con l'obiettivo di disabilitarle possibilmente senza essere individuati.

La honeynet può essere sfruttata per svolgere attività illegali, ad esempio distribuendo software copiato, materiale pornografico, numeri di carte di credito ecc. Il rischio è che tali azioni vengano addebitate al responsabile della honeynet, lasciandogli l'onere di dimostrare la sua estraneità.

In tutti i casi un'adeguata azione di contenimento prevede un presidio continuo dei sistemi ed una accurata personalizzazione. La presenza di personale qualificato può consentire l'adozione di opportune contromisure in caso di attacchi andati a buon fine. La personalizzazione, al contrario, può creare dei problemi. Le soluzioni più ampiamente adottate rientrano nella filosofia dell'open source per cui chiunque ha accesso alle informazioni, inclusi gli attacker che possono sviluppare delle opportune contromisure. L'ideale sarebbe introdurre dei meccanismi altamente dinamici in grado di adattarsi alle varie situazioni senza ricorrere a procedure standard.

## 2.17 Classificazione

In letteratura solitamente si adotta una classificazione che caratterizza l'evoluzione delle honeynet parlando in termini di generazioni.

### 2.17.1 Generazioni di honeynet

Le honeynet di prima generazione furono sviluppate dalla Honeynet Alliance nel 1999, con l'obiettivo di migliorare la tecnologia degli honeypot, già ampiamente utilizzata negli ambienti di ricerca, soprattutto in merito all'emulazione dei servizi.
I principali vantaggi rispetto agli honeypot sono rappresentati dalla maggiore capacità di acquisizione dei dati e dalla possibilità di cattura di attacchi sconosciuti grazie all'impiego di sistemi reali in luogo di software di emulazione ed alla presenza di una infrastruttura di rete che consente la dislocazione di più sensori di rilevamento.

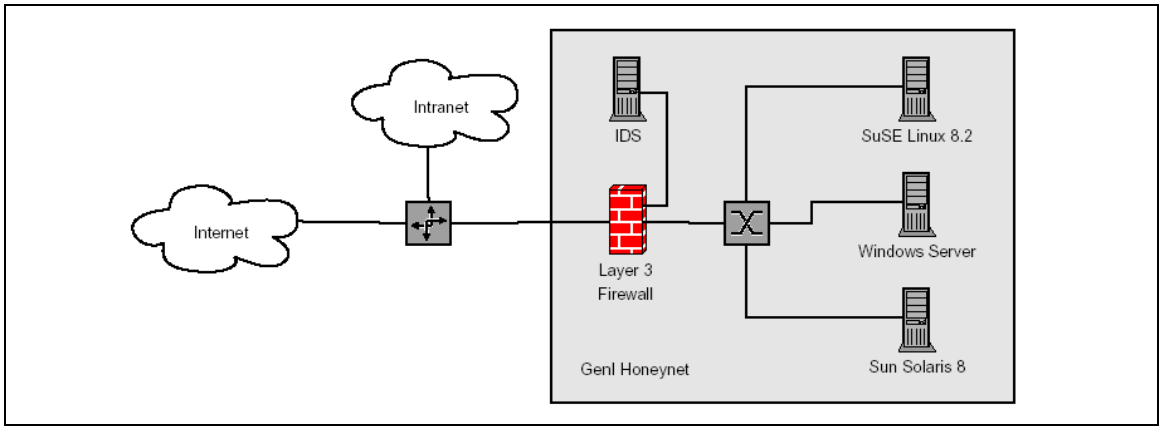



**Figura 2.3 Esempio di honeynet di prima generazione.**

La funzione di ***data control*** è svolta da un classico firewall al quale sono applicate delle specifiche regole che in primo luogo cercano di prevenire l'eccessiva connettività in uscita, autorizzando tutte le connessioni in ingresso ma applicando un rigido controllo sulle altre con un blocco in caso di superamento del limite massimo consentito. Lo scopo è di impedire l'utilizzo del sistema compromesso per sferrare attacchi esterni alla honeynet e contemporaneamente rallentare le attività intrusive, favorendo la reazione dei responsabili della sicurezza.
La cattura delle informazioni deve avvenire in modo del tutto trasparente e si ottiene operando su più livelli con tecniche complementari. Il firewall acquisisce i dati sulle connessioni in ingresso e uscita, lo stato delle porte, gli indirizzi IP, data e ora

dell'attacco. Il secondo livello, costituito da un IDS, serve ad identificare gli attacchi noti e ad acquisire i pacchetti per una successiva analisi. L'ultimo livello è rappresentato dagli honeypot stessi, opportunamente equipaggiati per tracciare al meglio le azioni svolte dagli aggressori. Infatti l'honeywall può controllare il traffico in ingresso e uscita, ma non le azioni svolte sul sistema attaccato, soprattutto in presenza di connessioni criptate.

Le honeynet di prima generazione risultano efficaci nella identificazione di attacchi automatici o di basso livello. Gli svantaggi principali riguardano i limiti nel data control (operante a livello 3 della pila OSI) che ne permettono agevolmente l'identificazione. Sono poco interessanti per i più smaliziati perché normalmente gli honeypot corrispondono ad installazioni di default di vari sistemi operativi.

Le honeynet di seconda generazione (***GenII***) furono introdotte nel 2001 e quelle di terza (***GenIII***) nel 2004. Fondamentalmente hanno la medesima architettura sebbene le GenIII forniscano un supporto migliore nella gestione (attraverso una interfaccia web). La principale evoluzione rispetto alla prima generazione riguarda l'introduzione di un unico dispositivo con funzione di data control e data capture separato dagli honeypot stessi, orientando gli sforzi nella direzione di rendere le honeynet più difficili da rilevare e riconoscere, permettendo di monitorare più a lungo l'attività degli intrusi.

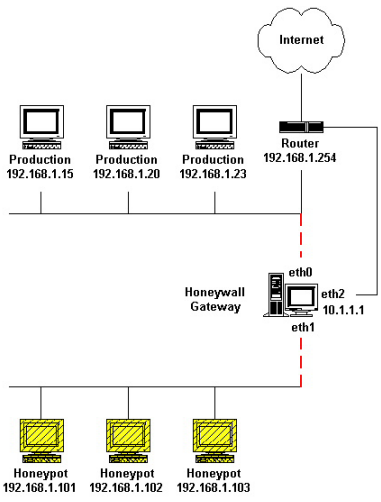



**Figura 2.4 Diagramma di una honeynet di seconda generazione**

Le funzionalità di IDS e firewall sono integrate in un unico dispositivo, l'***honeywall gateway***, facilitando lo sviluppo e la gestione.

Il Data Control è ottenuto adottando una serie di accorgimenti quali la limitazione del numero di connessioni consentite e l'impiego delle modalità ***packet-drop*** e ***packet-replace*** nella gestione dei pacchetti. Si utilizza un dispositivo ibrido firewall/IDS per sfruttare le funzionalità del firewall in modo da limitare il numero di connessioni in uscita (per l'Honeypot Project il valore previsto è di 50 connessioni all'ora).

Le modalità ***packet-drop*** e ***packet-replace*** sono il nucleo principale di questa tipologia di honeynet. L'IDS permette di distinguere il traffico potenzialmente ostile, utilizzando il database delle signature per confrontare e processare tutti i pacchetti in uscita. Attraverso la modalità packet-drop vengono scartati i pacchetti per i quali è stata trovata una corrispondenza nel database, in quanto parte di un tentativo di attacco conosciuto. Tale compito viene affidato a prodotti come **Snort-Inline**[7], in grado di modificare i pacchetti "particolari" per renderli inoffensivi, operando di fatto come un ***IPS*** (***Intrusion Prevention System***). Anche in questo caso, come per le GenI, tutto il traffico viene filtrato da un nodo che agisce nella duplice veste di sniffer e logger, catturando i pacchetti in transito. La principale differenza è rappresentata dal fatto che questo sensore (***IDS gateway***) opera a livello 2 della pila OSI come un ***bridge***, per cui non dispone di un indirizzo IP e ciò lo rende più difficile da rilevare in quanto non svolge funzioni di routing, non viene modificato il TTL (time to live) dei pacchetti ad ogni hop ed infine non è previsto un indirizzo IP.

Il firewall si occupa, come per le GenI, di registrare i dati relativi a tutte le connessioni in ingresso e uscita affidando all'IDS il compito di catturare tutti i pacchetti in ingresso e uscita per analisi successive. L'ultimo livello rappresenta una novità perché aggiunge ai classici log di sistema, in grado di evidenziare i processi in esecuzione durante l'attacco, la capacità di keystroke attraverso cui è possibile acquisire le sequenze di comandi impartite dagli attacker. La sua utilità si manifesta soprattutto in presenza di connessioni cifrate basate su SSH, dove la funzione di sniffing si rivela inefficace.

[7] http://snort-inline.sourceforge.net

### 2.17.2 Honeynet distribuite

Una ***honeynet distribuita*** è costituita da una serie di honeynet dislocate in locazioni diverse che inviano i loro dati verso un sistema di raccolta centralizzato. Un esempio è rappresentato dall'Honeynet Research Alliance. In tale contesto risulta necessario garantire i requisiti di automazione, flessibilità e trasparenza nell'ambito del Data Control e Data Capture a cui si aggiunge la necessità di Data Collection per fare in modo che i dati possano essere aggregati ed analizzati correttamente. Per questo motivo risulta molto utile ricorrere alla tecnologia delle ***macchine virtuali*** che approfondiamo nel prossimo capitolo.

### 2.17.3 Honeynet virtuali

Accanto alle classiche honeynet fisiche trovano posto quelle ***virtuali*** che permettono di installare una honeynet completa su un singolo computer, grazie all'utilizzo di software di virtualizzazione che garantiscono l'esecuzione di più sistemi operativi contemporaneamente sfruttando un solo hardware. I vantaggi sono essenzialmente riconducibili alla riduzione dei costi ed alla semplicità di gestione dovendo occuparsi di un unico sistema. Tra gli svantaggi annoveriamo il numero limitato di piattaforme supportate dal software di virtualizzazione. Inoltre non bisogna trascurare il maggior livello di rischio, rappresentato dalla compromissione del software di virtualizzazione che può pregiudicare il funzionamento dell'intera honeynet. Infine ottenuto il controllo del sistema ospite è possibile scoprire che si tratta di un ambiente virtuale. Spitzner classifica le honeynet virtuali in due categorie: ***self-contained*** (le più comuni) e ***ibride***.

Nelle honeynet virtuali ***self-contained*** l'intera struttura è collocata su un singolo sistema fisico dove trovano posto sia il gateway che gli honeypot stessi. I vantaggi principali di questa soluzione sono la portabilità, la facilità di impiego e l'economicità. L'honeynet può essere installata ad esempio su un notebook senza richiedere l'acquisto di hardware specifico e trasportata ovunque per essere immediatamente operativa semplicemente collegandola ad una rete. Tra gli svantaggi dobbiamo osservare l'esistenza di un singolo punto di controllo che può mettere fuori uso l'intera honeynet in caso di compromissione del sistema host. Inoltre sul fronte della sicurezza bisogna tenere in conto i rischi legati allo stesso software di virtualizzazione.

Le honeynet virtuali ibride combinano una classica honeynet e il software di virtualizzazione. Data Capture (firewall) e Data Control (IDS) si trovano su due sistemi separati e isolati, in modo da ridurre i rischi di compromissione. Tuttavia gli honeypot virtuali possono girare su un singolo sistema. I principali vantaggi sono la ***sicurezza*** determinata dalla separazione tra le componenti di logging e gli honeypot con conseguente riduzione del rischio di compromissione e disattivazione e l'elevata ***flessibilità*** che permette l'adozione di varie soluzioni hardware e software per l'implementazione di Data Control e Data Capture.

D'altronde ci sono svantaggi come la ***scarsa portabilità*** determinata dall'impiego di più macchine fisiche che ne rende complesso il trasporto e i ***costi*** dovuti all'impiego di più energia, spazio e denaro per la gestione delle varie macchine della rete.

# 3. Honeyd e Roo

Di seguito analizzeremo in dettaglio due soluzioni molto complete ed efficienti messe a disposizione dalla comunità open source. ***Honeyd*** è un esempio di honeypot a medio livello di interazione con cui si può creare un ambiente di rete molto complesso su di una singola macchina fisica, mentre ***Roo*** permette di installare in maniera del tutto automatica una completa honeynet di seconda generazione.

## 3.1 Honeyd

Progetto open source[1] sviluppato da Niels Provos dell'Università del Michigan in grado di simulare la presenza di host virtuali all'interno di una rete. Pur essendo nato specificamente per sistemi Unix-based, esiste anche un porting per Windows realizzato da Michael Davis. In dettaglio si tratta di un software capace di assumere la "personalità" di qualsiasi sistema operativo e che può essere opportunamente configurato per offrire vari servizi basati sul protocollo TCP/IP come http, smtp, ssh, ecc. Pur rientrando nella categoria degli honeypot a basso/medio livello di interazione si distingue per la sua versatilità nel simulare una completa infrastruttura di rete su di un'unica macchina fisica, consentendo l'impostazione di vari parametri come numero di hops, larghezza di banda, perdita di pacchetti e latenza.

## 3.2 Caratteristiche di Honeyd

Le principali caratteristiche di Honeyd si possono riassumere in:

- simulazione di reti di grosse dimensioni
- elevata configurabilità
- supporto di router multipli per la gestione di più reti
- integrazione tra reti fisiche e reti virtuali

[1] http://www.honeyd.org

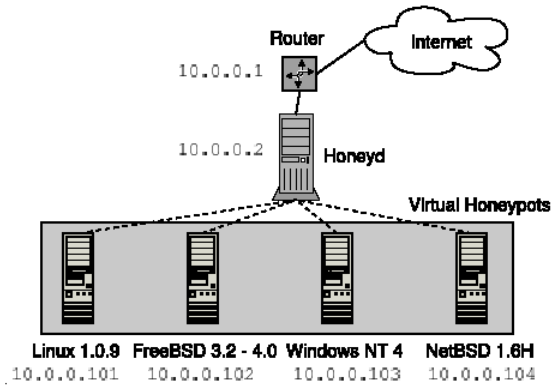



**Figura 3.1 Honeypot virtuali generati da Honeyd**

Lo scopo principale di Honeyd è il rilevamento di attività non autorizzate all'interno di una rete, attraverso il monitoraggio degli indirizzi IP non utilizzati di uno specifico intervallo (denominato "***dark space***"). Ogni connessione verso uno di tali indirizzi viene considerato come un possibile attacco o comunque come il risultato di una scansione. Viene esaminata qualsiasi attività relativa alle porte TCP e UDP nonché al traffico ICMP. In aggiunta si possono emulare dei servizi, utilizzando script realizzati in Perl, shell o altro in grado di interagire con l'aggressore secondo una modalità opportunamente stabilita.

## 3.3 Architettura di Honeyd

L'architettura di Honeyd consiste di varie componenti:

- un database delle configurazioni
- un packet dispatcher
- un gestore di protocolli
- un personality engine
- un componente di routing opzionale

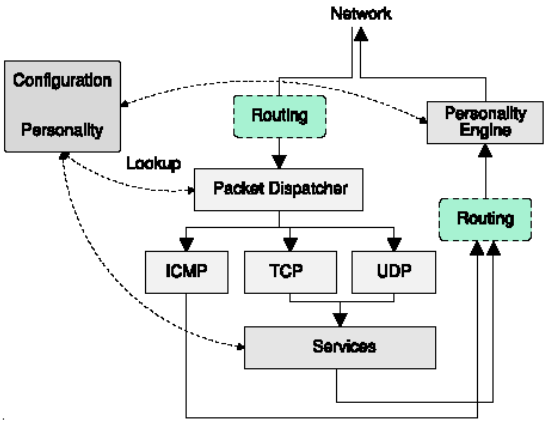



**Figura 3.2 Architettura di Honeyd**

I pacchetti in arrivo vengono processati dal packet dispatcher che controlla la lunghezza dell'indirizzo IP e ne verifica l'integrità tramite il relativo checksum. Sono supportati i principali protocolli : TCP, UDP e ICMP. Tutti gli altri pacchetti vengono scartati. A questo punto viene interrogato il file di configurazione alla ricerca di un modello associato all'indirizzo IP di destinazione, in mancanza del quale si utilizza il modello di default. Nel caso dei protocolli UDP e TCP è possibile interagire con applicazioni esterne capaci di gestire il flusso dati. Alla ricezione di una richiesta il framework verifica se il pacchetto riguarda una connessione già stabilita e in questo caso lo inoltra al corrispondente servizio, altrimenti viene creato un nuovo processo per la sua esecuzione. Oltre ai servizi locali è supportato un meccanismo di reindirizzamento delle connessioni grazie al quale è possibile girare una richiesta di servizio ad un sistema reale. Prima che un pacchetto sia immesso sulla rete viene processato dal personality engine che apporta delle modifiche al suo contenuto per renderlo compatibile con lo stack utilizzato dal sistema operativo simulato.

## 3.4 Honeyd: personality engine

I pirati informatici utilizzano strumenti come ***Xprobe***[2] e ***Nmap***[3], che esaminando lo stack IP del sistema bersaglio riescono ad identificare con sufficiente precisione il sistema operativo in esecuzione. Le informazioni così ricavate sono utili per ricercare

[2] http://xprobe.sourceforge.net
[3] http://insecure.org/nmap

eventuali vulnerabilità e approntare un attacco mirato. Honeyd riesce a simulare il comportamento dello stack di rete di un determinato sistema operativo, ovvero può assumerne la personalità, modificando gli header dei pacchetti in uscita in modo da garantire una corrispondenza con le caratteristiche del sistema operativo desiderato. Per default Honeyd sfrutta il ***fingerprinting database*** di Nmap come riferimento per le personalità TCP e quello di Xprobe per quelle ICMP. In questo modo cerca di ingannare gli attacker con le loro stesse armi.

```
Fingerprint IRIX 6.5.15m on SGI O2
TSeq(Class=TD%gcd=<104%SI=<1AE%IPID=I%TS=2HZ)
T1(DF=N%W=EF2A%ACK=S++%Flags=AS%Ops=MNWNNTNNM)
T2(Resp=Y%DF=N%W=0%ACK=S%Flags=AR%Ops=)
T3(Resp=Y%DF=N%W=EF2A%ACK=O%Flags=A%Ops=NNT)
T4(DF=N%W=0%ACK=O%Flags=R%Ops=)
T5(DF=N%W=0%ACK=S++%Flags=AR%Ops=)
T6(DF=N%W=0%ACK=O%Flags=R%Ops=)
T7(DF=N%W=0%ACK=S%Flags=AR%Ops=)
PU(Resp=N)
```

**Figura 3.3 Esempio di fingerprint del database di Nmap**

## 3.5 Installazione di Honeyd

Sono possibili diverse alternative, dalla compilazione dei sorgenti (disponibili per varie piattaforme), all'utilizzo di file binari precompilati (starter-kit), fino all'impiego di pacchetti RPM già pronti.

### 3.5.1 Compilazione dei sorgenti

Il file README incluso evidenzia la dipendenza dalle seguenti librerie che occorre installare prima della compilazione dei sorgenti:

- libdnet
- libevent
- libpcap
- libdnsres

I file indicati di seguito corrispondono alle ultime versioni disponibili al momento della stesura della tesi.

1. **LIBDNET**[4]

   Fornisce una semplice interfaccia per l'accesso a varie routine di basso livello per il networking.

```
libdnet-1.11.tar.gz
   1. tar xzf libdnet-1.11.tar.gz
   2. cd libdnet-1.11
   3. ./configure
   4. make
   5. sudo make install
   6. sudo /sbin/ldconfig
```

- **LIBEVENT**[5]

  API per l'esecuzione di funzioni specifiche in presenza di determinati eventi relativi ad un file oppure alla scadenza di un timeout.

```
libevent-1.1.tar.gz
   1. tar xzf libevent-1.1.tar.gz
   2. cd libevent-1.1
   3. ./configure
   4. make
   5. sudo make install
   6. sudo /sbin/ldconfig
```

- **LIBPCAP**[6]

  Libreria di cattura dei pacchetti di rete.

```
libdnet-1.11.tar.gz
   1. tar xzf libdnet-1.11.tar.gz
   2. cd libdnet-1.11
   3. ./configure
   4. make
   5. sudo make install
   6. sudo /sbin/ldconfig
```

- **LIBDNSRES**[7]

  Libreria per la risoluzione dei nomi DNS.

[4] http://www.monkey.org/~provos/libdnet
[5] http://www.monkey.org/~provos/libevent
[6] http://winpcap.polito.it
[7] http://www.monkey.org/~provos/libdnsres

```
libdnsres-0.1a.tar.gz
   1. tar xzf libdnsres-0.1a.tar.gz
   2. cd libdnsres-0.1
   3. ./configure
   4. make
   5. sudo make install
   6. ln –s /usr/local/lib/libdnet.1.0.1
      /usr/local/lib/libdnet.so
   7. sudo /sbin/ldconfig
```

- **HONEYD**

```
honeyd-1.5a.tar.gz
   1. tar xzf honeyd-1.5a.tar.gz
   2. cd honeyd-1.5a
   3. ./configure
   4. make
   5. mkdir /usr/local/share/honeyd
   6. sudo make install
   7. chown –R nobody /usr/local/share/honeyd
   8. sudo /sbin/ldconfig
```

#### 3.5.1.1 Problemi segnalati

In base alla distribuzione Linux utilizzata possono essere richiesti alcuni pacchetti aggiuntivi non installati di default, sebbene le indicazioni fornite dai comandi **`configure`** e **`make`** durante il processo di compilazione non siano molto esplicative. In sintesi i problemi principali riscontrati e le relative soluzioni sono:

- se richieste le librerie ***libedit*** o ***libreadline*** installare il pacchetto **readline-devel**
- in caso di errori relativi al supporto del linguaggio ***python*** installare **zlib-devel**
- per la compilazione di ***libpcap*** sono richiesti i pacchetti **flex** e **bison.**

Normalmente gli errori sono generati dalla mancanza dei file header .h inclusi nei sorgenti utilizzati dagli sviluppatori.

### 3.5.2 Starter-kit

Lo scopo del kit[8] è rendere disponibili tutti gli strumenti necessari per l'esecuzione di Honeyd su architettura i386 Linux, senza preoccuparsi del processo di compilazione, grazie ad una serie di file binari precompilati. Rispetto alla versione standard non sono supportate tutte le funzionalità avanzate come l'interfaccia web. Dopo aver scompattato il file compresso procediamo come segue:

1. Impostare il proprietario dei file a "nobody"

```
% chown -R nobody honeyd-dir
```

2. Creare la directory per i file di log

```
% mkdir /var/log/honeyd
% chown -R nobody /var/log/honeyd
```

3. Eseguire gli script di shell

```
% ./start-arpd.sh
% ./start-honeyd.sh
```

Onde evitare problemi di sicurezza il demone honeyd dovrebbe essere eseguito con il minor numero possibile di privilegi, per cui conviene creare un utente ad hoc che non possa accedere al sistema tramite il classico login. Lo script **`start-arpd.sh`** viene avviato per prima in modo da disporre dell'***arp spoofing*** per la gestione del traffico indirizzato verso i sistemi fantasma. In entrambi gli script si fa riferimento alla rete 192.168.1.0/24, ma nulla vieta di personalizzarli opportunamente in base alle proprie esigenze. Per default Honeyd ricerca i file di configurazione (e gli script di emulazione) nella directory **`/usr/local/share/honeyd`**, mentre nel kit si fa riferimento alla directory locale di installazione.

### 3.5.3 Honeyd su Windows

Il porting per la piattaforma Windows[9], pur non disponendo di tutte le funzionalità avanzate previste nella versione per Linux, consente di provare il software senza dover affrontare le problematiche legate alla compilazione del codice e alla

[8] http://www.citi.unich.edu/u/provos/honeyd/honeyd-kit-1.0c-a.tgz
[9] http://www.winhoneyd.org

gestione di un sistema Linux non sempre alla portata di tutti. L'unico requisito è la presenza della libreria ***WinpCap*** per la gestione del traffico di rete.

## 3.6 Configurazione di Honeyd

L'implementazione di Honeyd necessita di un tool specifico denominato ***Arpd***, che svolge funzioni di ***ARP Spoofing***, cioé controlla costantemente lo spazio IP inutilizzato relativo ad un dato intervallo e inoltra tutte le richieste di connessione verso un unico sistema, in questo caso l'honeypot. La configurazione si ottiene molto semplicemente attraverso un file di testo all'interno del quale occorre specificare gli honeypot virtuali e le relative topologie di rete. Honeyd offre tre diverse modalità per aggiungere servizi di rete agli host emulati:

- invocazione di servizi esterni, operanti come processi autonomi, ai quali trasferisce i dati in ingresso e da cui riceve una risposta che viene girata al client;
- utilizzo di plug-in interni scritti in Python, un linguaggio di programmazione di alto livello, e integrati direttamente in Honeyd che vengono eseguiti all'interno di tale processo;
- reindirizzamento delle connessioni in ingresso verso applicazioni reali eventualmente residenti su host fisici differenti.

I nuovi template vengono creati attraverso la primitiva ***create***, mentre il comando ***set*** assegna al template una delle personalità del file di Nmap, determinando di fatto il comportamento dello stack di rete. Occorre specificare esattamente la stringa corrispondente al sistema operativo prescelto per non incorrere in errori, visto che viene effettuato il parsing del file delle personalità. In aggiunta ***set*** definisce il comportamento di default da tenere in relazione a ciascuno dei protocolli supportati, scegliendo tra le seguenti opzioni:

- **block**: scarta i pacchetti
- **reset**: simula una porta chiusa
- **open**: simula una porta aperta

```
create windows
set windows personality "Windows XP Professional SP1"
```

Attraverso il comando ***add*** si stabiliscono i servizi accessibili in remoto specificando protocollo, porta e comando da eseguire per ciascuno di essi.

```
add tcp port 23 open "/usr/bin/perl scripts/router-telnet.pl"
add tcp port 25 open
set windows default tcp action reset
```

Nell'esempio supponiamo che le porte 23 e 25 siano aperte, per simulare la presenza dei servizi telnet ed smtp. In particolare lo script **`router-telnet.pl`**, eseguito ad ogni richiesta indirizzata alla corrispondente porta, riproduce il comportamento classico del servizio telnet su un host Unix-like. Viceversa qualunque altra porta TCP risulterà chiusa. La direttiva ***proxy*** consente l'inoltro di una connessione verso un host differente oppure verso un servizio reale.
Ad esempio volendo dirottare le chiamate verso un web server reale (in questo caso collocato sulla stessa macchina che ospita Honeyd) possiamo utilizzare la direttiva:

```
add windows tcp port 80 proxy localhost:80
```

Naturalmente localhost può essere sostituito da un qualsiasi altro indirizzo IP valido.
Infine con ***bind*** si assegna un indirizzo IP al template. Tutti gli indirizzi IP a cui non corrisponde uno specifico template vengono associati a quello di default.

```
bind 192.168.1.50 windows
```

Una caratteristica estremamente interessante è rappresentata dai template dinamici che consentono ad Honeyd di modificare il suo comportamento in base a varie condizioni:

- **indirizzo sorgente**: da cui viene originata la connessione verso l'honeypot;
- **sistema operativo**: relativo all'host che ha originato la connessione ed individuato tramite la tecnica del ***passive fingerprinting***;
- **intervallo temporale**: il template viene attivato e disattivato ad intervalli prestabiliti per simulare un ambiente realistico.

Ad esempio scrivendo

```
dynamic host
add host use windowsxp if source os = windows
add host use linux if source ip = 192.168.0.0/16
add host otherwise use default
```

si stabilisce che il template host sarà definito in termini di altri template in base alle condizioni specificate:

- se il sistema operativo dell'host remoto è windows l'honeypot farà riferimento al template **`windowsxp`**;
- viceversa se la connessione proviene dalla rete 192.168.0.0/16 verrà utilizzato il template **`linux`**;
- in tutti gli altri casi si farà riferimento al modello di **`default`**.

## 3.7 Honeyd: esecuzione

Prima di eseguire Honeyd bisogna fare in modo che tutte le richieste di connessione rivolte all'intervallo di indirizzi da controllare siano dirette verso la scheda di rete sulla quale è in ascolto il software. Allo scopo utilizziamo l'applicativo gratuito ***arpd***. Ammesso di voler monitorare la rete 192.168.1.0/24 attraverso l'interfaccia di rete eth1 basta eseguire il comando (modificando opportunamente l'intervallo di indirizzi che si intende utilizzare):

```
% arpd –i eth1 192.168.1.0/24
```

Quindi possiamo lanciare Honeyd con il seguente comando:

```
% honeyd –f honeyd.conf –p /usr/local/share/nmap.prints –x \
/usr/local/share/xprobe2.conf 192.168.1.0/24
```

Il comando deve essere scritto su una singola riga. L'opzione **–f** specifica la posizione del file di configurazione, mentre **–p** quella del file contenente le personalità utilizzate da Nmap per la simulazione dei vari stack IP. Infine con **–x** si specifica il file (lo stesso di Xprobe) per determinare la risposta da adottare nel caso di scansioni basate sul fingerprinting del protocollo ICMP.

## 3.8 Logging di Honeyd

In primo luogo diamo uno sguardo al file **/var/log/messages**, in cui Honeyd registra tutte le attività rilevate in maniera semplice e leggibile.

```
/var/log/messages
Feb 12 23:06:33 laptop honeyd[30948]: Connection to closed port: udp
(210.35.128.1:1978 - 192.168.1.101:1978)
Feb 12 23:23:40 laptop honeyd[30948]: Connection request: tcp
(66.136.92.78:3269 - 192.168.1.102:25)
Feb 12 23:23:40 laptop honeyd[30948]: Connection established: tcp
(66.136.92.78:3269 - 192.168.1.102:25) <-> sh scripts/smtp.sh
Feb 12 23:24:14 laptop honeyd[30948]: Connection dropped with reset:
tcp (66.136.92.78:3269 - 192.168.1.102:25)
Feb 12 23:34:53 laptop honeyd[30948]: Killing attempted connection:
tcp (216.237.78.227:3297 - 192.168.1.102:80)
Feb 12 23:39:14 laptop honeyd[30948]: Connection: udp (10.5.5.71:1026 -
192.168.1.101:137)
```

**Figura 3.4 Contenuti del log di sistema riferiti a Honeyd**

Si noti la presenza di molte informazioni utili come la data e l'ora della connessione stabilita o del semplice tentativo con il relativo esito; gli indirizzi IP del mittente e del destinatario con le relative porte, nonché il tipo di traffico TCP o UDP. Viceversa nel file **/var/log/honeyd** possiamo ritrovare le stesse informazioni ma in un formato differente, progettato espressamente per essere processato da script o tool automatici. Accanto alle informazioni raccolte direttamente da Honeyd si aggiungono quelle relative agli eventuali script utilizzati per emulare i servizi e a quelle ottenute attraverso l'impiego di IDS.

## 3.9 Roo

L'***Honeynet Project***[10] ha sviluppato una soluzione, chiamata ***ROO*** e basata sul sistema operativo Linux, in grado di implementare una Honeynet di seconda generazione in modo semplice e gratuito, partendo da un CD di boot.

I requisiti minimi prevedono la presenza di almeno due computer con il ruolo rispettivamente di ***honeypot*** e di ***honeywall***. Tutte le connessioni in ingresso e uscita all'honeypot vengono controllate da un firewall e le attività sospette segnalate grazie alla presenza di un IDS opportunamente configurato. L'accesso alla rete si ottiene tramite un bridge di livello 2, privo di indirizzo IP, che permette di ridurre la possibilità

[10] http://www.honeynet.org

di identificazione dell'honeywall stesso. L'attuale versione **1.39** è basata su una distribuzione **Fedora Core 3**.

## 3.10 Roo: componenti

### 3.10.1 Snort

***Snort***[11] è un IDS network-based, sviluppato sotto licenza GPL (GNU public license) e distribuito gratuitamente su Internet in formato open source e con il supporto di una vasta comunità di sviluppatori, tanto da essersi conquistato sul campo il ruolo di standard de facto per l'intrusion detection.

Le sue caratteristiche principali possono riassumersi in:

- **leggerezza** (l'eseguibile è circa 250kb)
- **portabilità** (Unix, Linux, Windows, *BSD, HP-UX, IRIX)
- **velocità di esecuzione**
- **configurabilità** (grazie alla semplicità del linguaggio delle regole e alle varie opzioni per il log)
- **modularità** (integrazione di plug-in)
- **licenza GPL/Open Source software**

L'architettura di Snort comprende tre sottosistemi primari:

- un packet sniffer/decoder molto efficiente;
- un motore di gestione delle regole basato sul pattern matching;
- un sottosistema per il logging, la creazione di report, e l'invio di avvisi all'amministratore di rete.

[11] http://www.snort.org

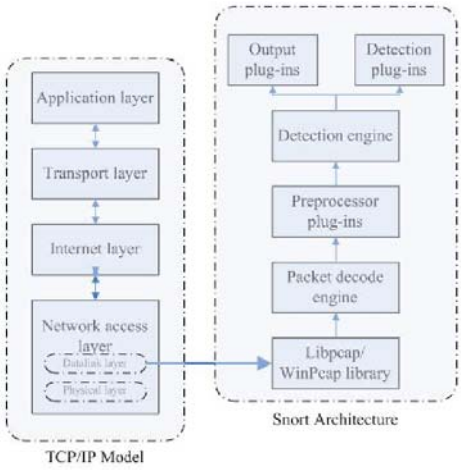



**Figura 3.5 Architettura di Snort**

I pacchetti catturati vengono sottoposti a Snort affinché possa decodificarli, esaminarli e all'occorrenza lanciare degli alert.

In particolare i passi eseguiti sono:

- il motore di decodifica dei pacchetti agisce in base al protocollo utilizzato a livello datalink (Ethernet, Token Ring, PPP);
- il preprocessore si occupa di riassemblare i pacchetti ricevuti, decodificando i protocolli;
- il motore di rilevamento confronta i pacchetti con le regole disponibili per scoprire eventuali comportamenti sospetti;
- ulteriori plug-in possono eseguire altri test;
- infine in caso di riconoscimento di un attacco viene chiamato l'output plug-in per formattare e presentare gli alert. Sono supportati vari formati come UNIX syslog, XML oltre alla possibilità di indirizzare l'output verso i più comuni database relazionali come Oracle, MySQL e PostgreSQL.

L'insieme delle regole (***knowledge base***) é memorizzato in vari file di testo, ciascuno dei quali costituisce un ***ruleset*** che raccoglie una lista di regole simili. E' molto importante la loro frequenza di aggiornamento per essere pronti a individuare nuovi attacchi. Sul sito ufficiale sono disponibili gli aggiornamenti con cadenza quasi giornaliera. La sintassi utilizzata è molto semplice ed efficace. Ad esempio:

```
log tcp any any -> 192.168.1.0/24 79
```

indica una regola che viene attivata in presenza di traffico TCP, proveniente da qualsiasi sorgente sia in merito all'indirizzo IP che alla porta e indirizzato verso la porta 79 di un host della rete 192.168.1.*.

Snort è in grado di analizzare il traffico TCP, UDP e ICMP. Gli indirizzi IP vengono scritti in formato decimale puntato seguiti dalla netmask in CIDR. Per le porte viene usato il numero decimale ad esse corrispondente, oppure si può anche indicare un intervallo utilizzando l'operatore ":" (es. 1:1024). In entrambi i casi il numero può essere sostituito dalla parola ***any*** per indicare che sono accettati tutti i valori possibili.

Le azioni che Snort può intraprendere sono:

- **alert**: genera un allarme secondo la modalità selezionata effettuando il log del pacchetto;
- **log**: effettua il log del pacchetto;
- **pass**: ignora il pacchetto;
- **activate**: genera un alert e poi attiva una regola dynamic;
- **dynamic**: equivalente al log ma viene attivata da una regola activate.

Si può inoltre specificare il contenuto e la tipologia dei pacchetti per cui far scattare l'allarme, ad esempio:

```
alert tcp any any -> 10.0.0.0/24 80 (content: "/cgi-bin/phf"; msg: "Probe del PHF!";)
```

permette di individuare qualunque tentativo di sfruttare la vecchia vulnerabilità del PHF[12], ricercando la stringa "/cgi-bin/phf" nel traffico tcp. Tale ricerca può riguardare anche specifiche sequenze in formato binario:

```
alert tcp any any -> 10.10.10.0/24 111 (content: "|00 01 86 a5|";msg: "Accesso al mountd";)
```

[12] Storico exploit per i sistemi Unix basato su uno script CGI di esempio chiamato appunto PHF che utilizza la funzione escape_shell_cmd() per verificare i dati in ingresso. La procedura di validazione dell'input non riesce a rilevare il carattere di ritorno a capo ("++", o "0x0a" in esadecimale), per cui questo carattere può essere utilizzato per forzare la fine dell'esecuzione dello script e fare in modo che il programma esegua, localmente sul server, tutto quello che segue il carattere di ritorno a capo. In questo modo si può visualizzare il file delle password oppure ottenere l'accesso ad una shell remota.

Infine si possono specificare ulteriori condizioni relative alla frammentazione dei pacchetti, al TTL, a specifici flag del TCP ecc.

### 3.10.2 Classificazione delle regole

Tutte le regole di Snort sono classificate in base alla loro priorità, dal livello critico (1) al basso rischio (3). Quando si scrivono delle nuove regole è possibile specificare tale valore a seconda del tipo di rischio considerato.

### 3.10.3 Snort_inline

Versione[13] modificata di Snort che, dopo aver acquisito i pacchetti da IPTables attraverso un modulo del kernel chiamato ip_queue in luogo di libpcap, stabilisce quali di essi devono essere scartati, modificati o accettati in base ad un nuovo insieme di regole di Snort, fungendo in sostanza da IPS (Intrusion Prevention System) e impedendo la diffusione di attacchi all'esterno della honeynet. Non tutto il traffico viene filtrato: l'interesse riguarda solo quello in uscita perché potenzialmente dannoso. Le nuove regole appositamente predisposte sono:

- **reject**: scarta il pacchetto riconosciuto e invia un segnale di reset icmp indicando un host irraggiungibile;
- **drop**: scarta il pacchetto riconosciuto e invia un alert;
- **sdrop**: scarta il pacchetto riconosciuto senza effettuare il log.

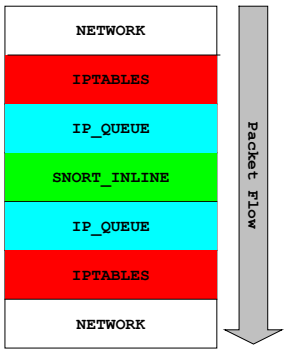



**Figura 3.6 Struttura di Snort-inline**

[13] http://snort-inline.sourceforge.net

Come si può notare nell'esempio seguente il payload dei pacchetti relativi ad attacchi conosciuti viene modificato in modo che falliscano senza arrecare danni.

```
reject tcp $HONEYNET any <>
              $EXTERNAL_NET 80 (msg: "REJECT";)
drop tcp $HONEYNET any <>
     $EXTERNAL_NET 80 (msg: "DROP TCP";)
sdrop tcp $HONEYNET any <>
     $EXTERNAL_NET 80 (msg: "SDROP";)
alert tcp $HONEYNET any <>
     $EXTERNAL_NET 80 (msg: "Modifying HTTP GET";
                           content:"GET"; replace:"BET";)
```

**Figura 3.7 Esempio di regole di filtraggio di Snort_Inline**

### 3.10.4 Firewall pf

Versione modificata del firewall ***pf***[14] di OpenBSD utilizzato per limitare il numero di sessioni contemporanee consentite.

### 3.10.5 Sebek

***Sebek***[15] nasce come tool di cattura delle informazioni con il preciso obiettivo di ricostruire gli eventi accaduti sull'honeypot ed è basato sul paradigma client/server: l'honeypot esegue la componente client che acquisisce i dati in modo trasparente, sostituendo la classica chiamata di sistema **`read()`** con una propria versione che copia tutti i dati letti mediante tale primitiva in un buffer specifico, inviandoli successivamente ad un server di raccolta, generalmente una macchina dedicata allo scopo, per le successive analisi. Il server è costituito a sua volta da tre componenti utilizzate per la cattura dei dati. Il primo ***sbk_extract*** è un programma C che acquisisce i pacchetti a partire da un file in formato tcpdump o direttamente in modalità live. Gli altri due sono script Perl: ***sbk_ks_log.pl*** estrae le sequenze di caratteri digitati dall'intruso inviandole allo standard output mentre ***sbk_upload.pl*** importa i pacchetti in un database per consentire ulteriori elaborazioni. In assenza di cifratura è possibile acquisire tutta l'attività di rete in forma di pacchetti da esaminare con appositi tool come Ethereal, recuperando il contenuto di un'intera sessione e di fatto visualizzando non solo l'input digitato dall'attacker ma anche le risposte del sistema. Il vero problema riguarda le comunicazioni cifrate per le quali è necessario recuperare il contenuto in chiaro. Piuttosto che impiegare ingenti risorse, talvolta inutilmente, per spezzare la

[14] http://www.openbsd/faq/pf
[15] http://www.honeynet.org/papers/sebek.pdf

chiave sono stati ricercati metodi alternativi per eludere la cifratura. La soluzione adottata consiste nell'acquisire i dati direttamente dal kernel del sistema operativo dopo la regolare fase di decrittazione.

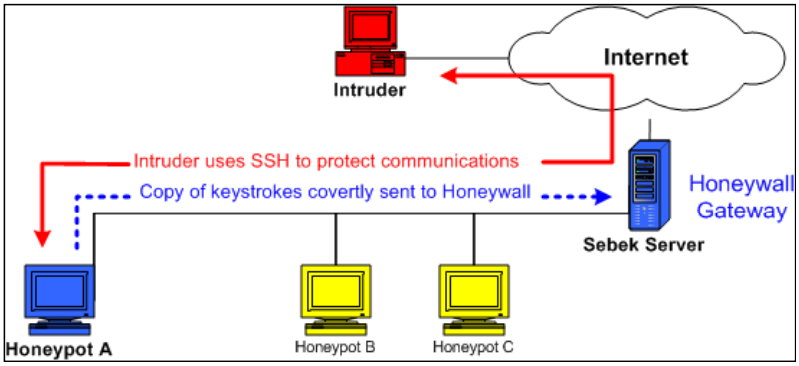



**Figura 3.8 Architettura di Sebek**

Il server Sebek può acquisire i dati in modalità live oppure partendo da un file in formato tcpdump. Lo scambio dei dati avviene utilizzando il protocollo UDP che, non essendo orientato alla connessione, ne rende più difficile l'individuazione. Il client risiede nello spazio di memoria riservato al kernel sull'honeypot e registra tutti i dati a cui un utente ha accesso tramite la system call read(). Per sortire questo effetto Sebek sostituisce il puntatore alla funzione all'interno della Tabella delle System Call in modo che faccia riferimento ad una nuova funzione appositamente predisposta. Quest'ultima provvede a chiamare quella originale, ne copia il contenuto in uno specifico buffer, aggiunge un header e invia il pacchetto al server.

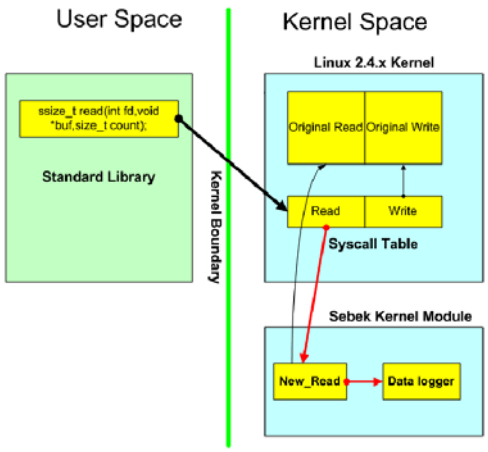



**Figura 3.9 Schema di redirezione delle chiamate alla read()**

A questo punto Sebek dispone di una visione completa di tutti i dati letti tramite la chiamata di sistema read(). La stessa tecnica si potrebbe utilizzare per qualunque altra funzione da monitorare. Per nascondere la sua presenza Sebek sfrutta le tecniche utilizzate dai comuni rootkit riuscendo a sparire dalla lista dei moduli installati e quindi rendendo difficile la sua individuazione.

### 3.10.6 Menu

Digitando la parola chiave **`menu`** si accede ad un menu testuale che consente la configurazione iniziale del sistema in maniera interattiva attraverso una serie di domande a cui rispondere per impostare i vari parametri e successivamente per la modifica degli stessi.


**Figura 3.10 Menu principale Honeywall CD**


**Figura 3.11 Menu di configurazione**

Per ciascuna categoria sono previsti una serie di sottomenu che consentono una gestione completa e dettagliata di tutte le componenti.

### 3.10.7 Walleye

L'interfaccia web costituita da Walleye permette l'amministrazione remota dell'Honeywall, mettendo a disposizione tutte le funzionalità del menu visto in precedenza oltre a fornire un ottimo supporto per l'analisi dei dati. Al momento della prima connessione occorre acquisire un certificato SSL in grado di attestare l'identità di Walleye, oltre alla richiesta di supporto per i cookie e JavaScript. A questo punto aprendo il browser e digitando:

***https://indirizzo-ip-gestione***

si accede alla schermata di login. Da notare l'uso di ***https*** trattandosi di una connessione cifrata. L'account di default è ***roo*** con password ***honey***, ma viene automaticamente richiesto il cambio della password alla prima connessione. Per ragioni di sicurezza la scelta viene assistita da un meccanismo di controllo che si basa sui seguenti requisiti:

- lunghezza minima 8 caratteri
- almeno una lettera maiuscola
- almeno una cifra
- almeno un simbolo

Dopo tre fallimenti successivi in fase di autenticazione l'utente è bloccato per 15 minuti, trascorsi i quali può nuovamente ritentare. Rispetto al Menu non è supportato il meccanismo interattivo di configurazione del sistema, bensì occorre riferirsi direttamente alle varie opzioni che si intende modificare. Un'altra differenza é la presenza di un modulo di gestione degli utenti che ne permette la creazione, modifica o eliminazione. In questo modo si può stabilire chi ha accesso a Walleye e con quali privilegi. Infatti sono previste tre categorie:

- **User**: hanno accesso in lettura alla sezione di analisi dei dati.
- **Admin Read-Only**: hanno accesso alle sezioni di analisi dei dati e allo status del sistema, ma senza possibilità di modifica.
- **Admin**: hanno accesso in lettura e scrittura a tutte le sezioni.

### 3.10.8 Pcap

API (Application Programming Interface) creata originariamente dal Politecnico di Torino[16] per la cattura dei pacchetti di rete. In ambito Unix-like è più nota con il nome di ***libpcap***, mentre per Windows si parla di ***WinPcap***. Pur essendo sviluppata per un utilizzo con i linguaggi C e C++, attualmente sono molti i software open source e commerciali che ne fanno uso per l'acquisizione del traffico di rete.

[16] http://winpcap.polito.it

### 3.10.9 Apache

***Apache***[17] é un HTTP server open source che rappresenta lo standard di riferimento per gli ambienti Unix-like. La sua presenza all'interno di Roo è richiesta per l'esecuzione dell'interfaccia di gestione web, denominata Walleye, che consente l'amministrazione in remoto dell'honeywall.

### 3.10.10 P0f (Passive OS Fingerprinting)

Tool[18] sviluppato da Michal Zalewski per l'acquisizione di informazioni su un sistema operativo (tipo e versione) in esecuzione su un host remoto. P0f esamina nello specifico i pacchetti scambiati durante il three-way handshake del TCP o semplicemente si limita a monitorare tutti quelli che passano attraverso la rete. Questa tecnica chiamata ***passive fingerprinting*** non altera né aggiunge pacchetti in una comunicazione per cui è in teoria non rilevabile. Le informazioni sono ottenute tramite il confronto tra i flag impostati nei pacchetti TCP e un database di fingerprint relativo ai vari sistemi operativi, dal momento che ciascuno di essi utilizza una propria implementazione dello stack TCP/IP. P0f riesce a determinare lo stato delle varie connessioni stabilite con la macchina remota. Le connessioni in ingresso sono individuate dalla presenza di un singolo pacchetto ***SYN*** originato dall'host remoto e destinato alla rete locale. Quelle in uscita al contrario richiedono un ***SYN*** dalla macchina locale a cui corrisponde un pacchetto ***ACK*** di risposta da quello remoto. ***RST+mode*** identifica i sistemi che hanno rifiutato una connessione, infine ***ACK+mode*** permette di ottenere informazioni dalle connessioni TCP già stabilite. In aggiunta P0f dispone di altre caratteristiche, non strettamente legate al passive fingerprinting, come l'individuazione della presenza di più host dietro un gateway o di vari dispositivi come ethernet, wireless ecc. Infine è in grado di salvare in vari formati i pacchetti di rete esaminati.

### 3.10.11 Argus

***Argus***[19] consente la creazione in tempo reale di report dettagliati circa lo stato e le prestazioni di tutte le transazioni in rete. Sono supportate varie metriche come

[17] http://www.apache.org
[18] http://lcamtuf.coredump.cx/p0f.shtml
[19] http://argus.tcp4me.com

connettività, capacità, perdita di pacchetti, ritardo ecc. Può essere utilizzato per analizzare e creare dei report sul contenuto dei pacchetti catturati oppure agire in modalità live esaminando tutti i dati in transito.

### 3.10.12 IPTables

La funzione di Data Control serve a contenere l'attività di un intruso all'interno e all'esterno di una honeynet. Generalmente vengono autorizzate tutte le connessioni in ingresso ma limitate quelle in uscita utilizzando ***IPTables***, un firewall integrato direttamente in Linux. Le caratteristiche principali di IPTables sono:

- limitazione della connettività
- Network Address Translation
- funzione di logging

IPTables definisce nella tabella ***filter*** tre gruppi di regole di controllo dette ***catene*** :

- **INPUT**: contiene le regole da utilizzare per i pacchetti in arrivo al firewall e destinati all'host locale;
- **OUTPUT**: contiene le regole per i pacchetti in uscita dal firewall e originati dall'host locale;
- **FORWARD**: contiene le regole per i pacchetti in transito verso altri host.

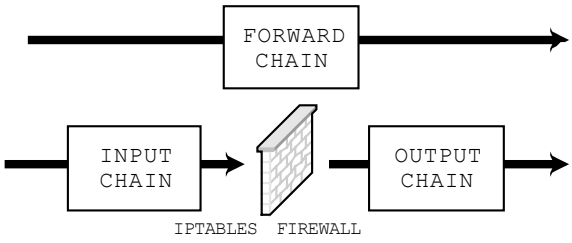



**Figura 3.12 Struttura del firewall IPTables**

Quando un pacchetto viene processato da una catena é soggetto alle regole specificate al suo interno in base al relativo ordine di inserimento. Una regola può decidere se scartare (***DROP***), rifiutare (***REJECT***) o accettare (***ACCEPT***) un pacchetto in base a molteplici parametri:

- interfaccia di rete

- indirizzi IP di origine e destinazione
- protocolli
- porte (TCP o UDP) di origine e destinazione

Se il pacchetto non soddisfa alcuna regola si applica quella di default prevista per la catena. Nell'ambito di ROO viene utilizzato come filtro per il sistema host, imponendo un limite al numero di connessioni in uscita, raggiunto il quale scatta un blocco per ogni ulteriore tentativo. Tutte le impostazioni sono contenute nello script **`rc.firewall`**[20], configurabile opportunamente. Tra le varie impostazioni possibili vi è la scelta della modalità operativa del gateway: routing (livello 3 della pila OSI) o bridging (livello 2). Nel secondo caso non vi è attività di routing né alterazione del TTL, per cui diventa difficile il suo rilevamento.

```
### Set the connection outbound limits for different protocols.
SCALE="day"
TCPRATE="15"
UDPRATE="20"
ICMPRATE="50"
OTHERRATE="15"
iptables -A FORWARD -p tcp -i $LAN_IFACE -m state --state NEW
         -m limit --limit ${TCPRATE}/${SCALE} --limit-burst
         ${TCPRATE} -s ${host} -j tcpHandler
iptables -A FORWARD -p tcp -i $LAN_IFACE -m state --state NEW
         -m limit --limit 1/${SCALE} --limit-burst 1 -s ${host}
         -j LOG --log-prefix "Drop TCP after ${TCPRATE} attempts"
iptables -A FORWARD -p tcp -i $LAN_IFACE -m state --state NEW
         -s ${host} -j DROP
```

**Figura 3.13 Esempio file di configurazione firewall IPTables**

### 3.10.13 Swatch

***SWATCH***[21] , ovvero "***The Simple WATCHer and filter***" è uno script in Perl realizzato da Todd Atkins per monitorare i file di log in tempo reale. A seguito del riconoscimento di uno specifico pattern, viene effettuata la notifica in base alla modalità prevista, comunemente facendo ricorso all'email. Il cuore del programma è rappresentato dal file di configurazione ***swatchrc***. La sintassi utilizzata è alquanto semplice: consiste di 4 campi di cui i primi 2 obbligatori e gli altri opzionali. Il primo campo **`/pattern/pattern`** indica l'espressione regolare oggetto della ricerca. Il secondo **`Action, Action`** specifica le azioni da eseguire in caso di riconoscimento del pattern. Sono previste varie opzioni tra cui l'invio di email di avviso o l'esecuzione di

[20] http://www.honeynet.org/tools/dcontrol/rc.firewall
[21] http://swatch.sourceforge.net

un qualsiasi altro programma specificato. Il terzo campo nella forma **HH:MM:SS** corrisponde ad un intervallo di tempo espresso in ore, minuti e secondi durante il quale SWATCH ignorerà eventuali pattern ripetuti a prescindere dal numero di occorrenze. L'ultimo, obbligatorio in presenza del precedente, rappresenta un timestamp nella forma **start:lenght** riportato nel messaggio di notifica. Vediamo un esempio concreto: supponiamo di voler monitorare il nostro server di posta, ricevendo un alert quando un utente non riesce a spedire una email. Scriviamo la seguente regola:

```
watchfor /Relaying denied|expn/
         echo=normal
         mail=abuse@ourcompany.net,subject=--- Sendmail Alert! ---
         throttle 5:00 0:16
```

La prima riga istruisce SWATCH a ricercare la stringa corrispondente all'impossibilità di inviare una mail e in caso di matching ad inoltrare un alert. La seconda stabilisce che venga visualizzata la riga del log in cui è stato individuato il pattern, mentre la quarta provvede all'invio di un messaggio di notifica. Sull'ultima riga troviamo ulteriori 2 campi (opzionali). Il primo impedisce la notifica di ulteriori alert nel caso di pattern identici rilevati in un intervallo di 5 minuti. L'ultimo indica la lunghezza del timestamp. Predisposto il file di configurazione resta da vedere la sintassi con cui lanciare il programma:

```
/usr/local/bin/swatch -c /var/log/swatchrc -t /var/log/syslog &
```

L'opzione **–c** permette di specificare la posizione del file di configurazione, mentre **–t** indica il monitoraggio dei log in tempo reale. Infine con **&** si richiede l'esecuzione del processo in background.

## 3.11 Requisiti di sistema

Fondamentalmente sono gli stessi per l'installazione della distribuzione di default di ***Fedora Core 3*** su cui ROO è basato. La configurazione minima richiede: una CPU basata su architettura Intel Pentium x86 (o superiore); 256MB (valore minimo) o 512MB (raccomandati) di memoria RAM e 550 MB di spazio libero per l'installazione di base. In un ambiente reale occorrono almeno 10 GB tenuto conto della quantità di dati da memorizzare (file di log, pacchetti di dati ecc.). L'ideale dal punto di vista delle

prestazioni sarebbe l'utilizzo di un disco SCSI. Infine é richiesta come minimo la presenza di una scheda di rete, numero che sale a 3 se si vuole disporre dei servizi di gestione e logging remoti.
L'uso di un sistema più potente può risultare vantaggioso sul fronte delle prestazioni.

## 3.12 Installazione

L'installazione, che cancella completamente il contenuto del disco di destinazione, prevede i seguenti passi:

- predisporre un sistema con i requisiti minimi visti in precedenza;
- scaricare l'immagine ISO del CD direttamente dal sito ufficiale[22];
- utilizzando un qualsiasi software di masterizzazione procedere alla creazione del disco di boot;
- modificare le impostazioni del BIOS per effettuare il bootstrap direttamente dal CDROM;
- avviare il sistema da CD. L'installazione sarà eseguita automaticamente, senza alcun intervento da parte dell'utente e una volta terminata verrà effettuato il reboot per poi proseguire con la configurazione.

## 3.13 Configurazione

Dopo aver effettuato l'installazione del cdrom ed eseguito il reboot si dispone di un sistema operativo basato su Fedora Core 3 dotato di tutte le funzionalità di un Honeywall. Il sistema operativo è stato ridotto al minimo (in totale 233 pacchetti RPM) e blindato attraverso l'esecuzione del file di configurazione

***/usr/local/bin/lockdown-hw.sh***

predisposto dal ***Center for Internet Security (CIS)***[23] e dal ***National Institute of Standards and Technology (NIST)***[24]. Tra le diverse opzioni vengono impostate le

[22] http://www.honeynet.org/tools/cdrom/roo/ISO/current/
[23] http://www.cisecurity.org
[24] http://www.nist.gov

caratteristiche delle password come lunghezza, scadenza, presenza di specifici caratteri. Viene negato l'accesso diretto al sistema dell'utente ***root*** sia localmente che in remoto e definiti i permessi relativi ai file di sistema. All'avvio viene proposto il prompt di login in modalità testo, non essendo supportato l'ambiente grafico per contenere le risorse. Il login è necessario per procedere al setup iniziale richiesto dall'Honeywall.

Si può scegliere tra due opzioni: creare manualmente un file di configurazione **`honeywall.conf`** oppure utilizzare il Menu di configurazione.

La seconda è più utilizzata perché consente di lavorare in remoto sfruttando una connessione SSH. Sono supportati due utenti di default: ***roo*** (user ID 501) e ***root*** (user ID 0), entrambi con associata la password ***honey***. Non é previsto il login diretto come root per cui bisogna ricorrere alla funzione ***su –*** che consente all'utente roo di acquisire i privilegi di amministratore. Il file di configurazione ***honeywall.conf*** non é altro che un file ASCII contenente tutti i valori da associare alle variabili utilizzate dal sistema operativo e dall'Honeywall.

## 3.14 Aggiornamenti

Per mantenere aggiornato il sistema si può ricorrere all'utilità ***yum*** specifica di Fedora, in grado di scaricare ed installare tutti gli upgrade dei pacchetti. I file di configurazione di yum sono collocati in ***/etc/yum.repos.d/****

Dopo una nuova installazione basta eseguire il comando (in qualità di root) ***yum update*** per aggiornare l'intero honeywall, scaricando i file necessari direttamente dal sito di Fedora[25].

## 3.15 Analisi dei log

La raccolta di informazioni da parte di Roo si basa su differenti file di log e sulla cattura del traffico in transito attraverso l'honeywall.

[25] http://www.fedora.org

I ***log del firewall IPtables*** riportano per ciascuna connessione data/ora, protocollo, indirizzi IP sorgente e destinazione e ulteriori dettagli relativi alle intestazioni IP

```
Jan  8 09:52:43 honeywall user.warn klogd: INBOUND ICMP: IN=br0
OUT=br0 PHYSIN=eth0 PHYSOUT=eth1 SRC=10.10.10.3 DST=10.10.10.10 LEN=84
TOS=0x00 PREC=0x00 TTL=64
```

**Figura 3.15 Esempio di log di IPTables**

Gli ***alert di Snort*** permettono di identificare la tipologia di attacco (se già noto) e il suo livello di criticità.

```
[**] [111:10:1] (spp_stream4) STEALTH ACTIVITY (XMAS scan) detection
[**]
01/08-10:06:09.729583 10.10.10.3:46271 -> 10.10.10.10:1
TCP TTL:52 TOS:0x0 ID:29436 IpLen:20 DgmLen:60
**U*P**F Seq: 0x452BBA60  Ack: 0x0  Win: 0x400  TcpLen: 40  UrgPtr: 0x0
TCP Options (4) => WS: 10 NOP MSS: 265 TS: 1061109567 0
```

**Figura 3.16 Esempio di alert di Snort**

I numeri presenti sulla prima riga hanno puro scopo identificativo: il primo indica il detector ossia il motore di parsing che ha rilevato l'alert. Il secondo è l'identificatore dell'alert (Snort-ID) mentre l'ultimo indica la revisione della regola. Successivamente viene data una descrizione testuale dell'alert. La riga successiva riporta la classificazione e il livello di priorità. Sulla terza linea sono riportate la data, l'ora, l'indirizzo IP sorgente e destinatario dell'alert. Quindi vengono indicati il protocollo di trasporto e vari dettagli relativi all'intestazione IP. I flag inutilizzati sono marcati da asterischi, mentre quelli impostati sono indicati dalla corrispondente lettera identificativa.

Infine il traffico catturato sotto forma di pacchetti completi (header + payload) può essere letto da vari tool come Ethereal, Snort o tcpdump[26].

Queste ed altre informazioni dettagliate sono accessibili nella sezione ***Data Analysis*** di Walleye (che presenta anche una serie di grafici molto interessanti) oppure attraverso il menu di configurazione.

[26] http://www.tcpdump.org

# 4. Tecniche anti-honeypot e analisi forense di un attacco

Finora abbiamo visto come sia possibile utilizzare la tecnologia di honeypot e honeynet per cercare di contenere gli attacchi informatici. Naturalmente sull'altro fronte i nostri nemici non restano a guardare ma si impegnano nella ricerca di contromisure adeguate. Si parla in gergo di tecniche ***anti-honeypot*** o anche di ***honeypot discovering***.

L'approccio migliore per nascondere la vera identità di un honeypot resta senza dubbio la personalizzazione. Molti tool in grado di evidenziarne la presenza ricercano, come nel caso di antivirus e IDS, file e directory specifiche oppure determinati processi in esecuzione. Rinominare i file binari o installare il software in locazioni diverse da quelle di default può talvolta rivelarsi una soluzione sufficientemente efficace.

## 4.1 Problemi di tempo

I principali interventi adottati nella costruzione degli honeypot riguardano il potenziamento dei meccanismi di logging o l'utilizzo di software di virtualizzazione per creare degli ambienti protetti. In entrambi i casi l'esecuzione di qualunque attività da parte di un intruso necessita di tempi più lunghi rispetto ad un "sistema normale", in quanto sono richieste più operazioni per il log, oppure è necessaria la loro emulazione nei sistemi a basso livello di interazione.

Attraverso la stima dei tempi di esecuzione è possibile individuare la presenza di un ambiente sospetto.

## 4.2 Honeypot virtuali

I sistemi a basso/medio livello di interazione consentono di creare degli ambienti particolarmente realistici attraverso un attento lavoro di personalizzazione.

La parola chiave da tenere presente è ***fingerprinting***, ovvero l'individuazione dell'impronta digitale di tale sistema da confrontare con quello reale onde evidenziare eventuali discrepanze. Comunemente gli attacker interagiscono in remoto concentrando i loro sforzi sul protocollo di comunicazione a vari livelli della pila ISO/OSI.

- **Interazione a livello datalink**

  Quando l'aggressore si trova sullo stesso segmento di rete dell'honeypot (es. nel caso di attacchi provenienti dall'interno della LAN) può identificarlo attraverso un semplice esame della propria cache arp nella quale sono riportate le associazioni tra indirizzi IP e MAC delle schede di rete. Infatti per simulare la presenza di molteplici host su una singola macchina si ricorre alla tecnica dell'***arp spoofing***, che sostanzialmente consiste nel reindirizzare il traffico diretto ad un determinato range di indirizzi IP verso un'unica scheda di rete. In questo modo è facile stabilire che si tratta di host fantasma dal momento che risultano tutti associati ad uno stesso MAC.

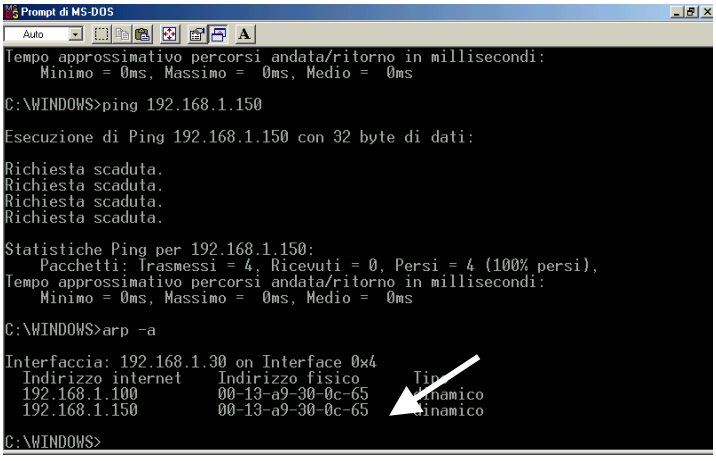



**Figura 4.1 Contenuto della cache arp**

Per risolvere il problema Honeyd, ad esempio, permette di associare un indirizzo MAC differente ad ogni sistema emulato tramite la direttiva:

```
set template ethernet <vendor | mac address >
```

- **Interazione a livello rete**

  In questo caso l'interesse si sposta sul protocollo IP e le relative intestazioni dei pacchetti, attraverso la cui analisi si possono scoprire elementi interessanti. Ogni sistema operativo utilizza in maniera particolare lo stack di rete (TTL, Window size, TOS ecc.) per cui è facile evidenziare eventuali errori in fase di emulazione.

- **Interazione a livello di applicazione**

  Nello specifico riguarda la capacità di riprodurre il comportamento delle varie applicazioni attraverso opportuni script, inclusi i messaggi prodotti dalle stesse. Talvolta anche un piccolo errore di battitura in una stringa può svelare la vera natura dell'honeypot.

Infine non bisogna dimenticare il buon senso e lasciarsi prendere dalle manie di grandezza. E' poco realistico avere in una rete molti sistemi operativi differenti, se non altro per i loro costi di gestione. Normalmente si tende ad avere ambienti omogenei per sfruttare al meglio le proprie competenze così come è altrettanto improbabile che una piccola azienda possa permettersi il lusso di avere apparecchiature molto costose (come alcuni modelli di router Cisco) destinate a realtà di maggiori dimensioni. Anche le applicazioni emulate devono risultare compatibili con l'ambiente in cui sono integrate: è più sensato avere IIS su Windows e Apache su Linux che non il contrario.

### 4.2.1 Software di virtualizzazione: VMware

Esistono vari metodi per rilevare le macchine virtuali create da VMware, come l'analisi delle informazioni di sistema o gli indirizzi MAC associati ai dispositivi di rete.

- **Informazioni di sistema**

Accedendo alle informazioni sul bios è possibile stabilire la presenza di macchine virtuali. Ad esempio visualizzando il file **dmesg** in Linux si notano delle tracce evidenti di VMware nei nomi attribuiti ai dispositivi hardware.

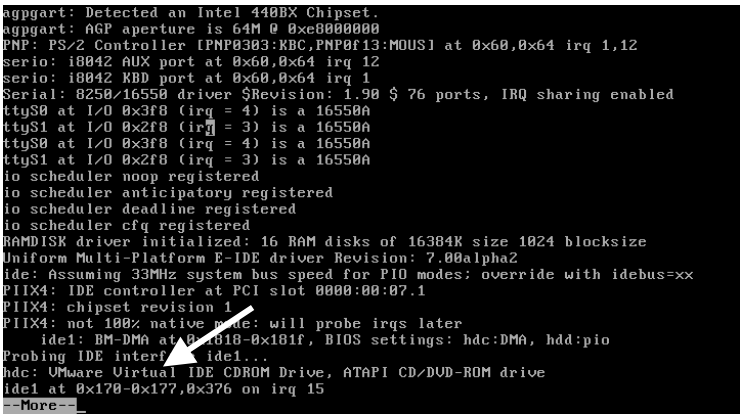



**Figura 4.2 File dmesg di una macchina virtuale Linux**

- **Indirizzi MAC**

Un altro metodo per rilevare VMware consiste nel guardare gli indirizzi MAC associati ai dispositivi di rete, che appartengono nello specifico a 3 classi non impiegate normalmente dai produttori:

**00-0C-29-XX-XX-XX**
**00-05-69-XX-XX-XX**
**00-50-56-XX-XX-XX**

Per scoprire l'indirizzo basta digitare in Windows il comando ***ipconfig /all***.

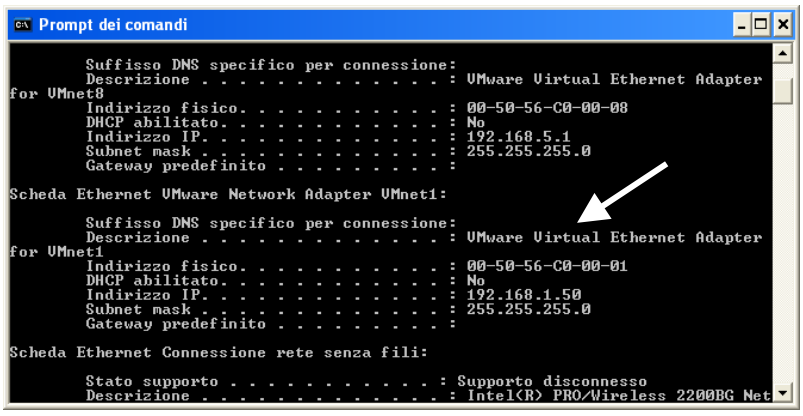



**Figura 4.3 Dispositivi virtuali su un sistema Windows**

Nel caso di un sistema basato su Linux occorre utilizzare il comando ***ifconfig –a***.

### 4.2.2 VMware: Contromisure

In ambiente Windows, per ovviare ad alcuni dei suddetti inconvenienti si può ricorrere ad un editor esadecimale come Ultraedit per modificare le stringhe identificative dei dispositivi virtuali all'interno del file binario di VMware: ***vmware-vmx.exe***. Dopo aver caricato tale file all'interno dell'editor ricerchiamo la stringa ASCII corrispondente ad es. al CDROM virtuale.

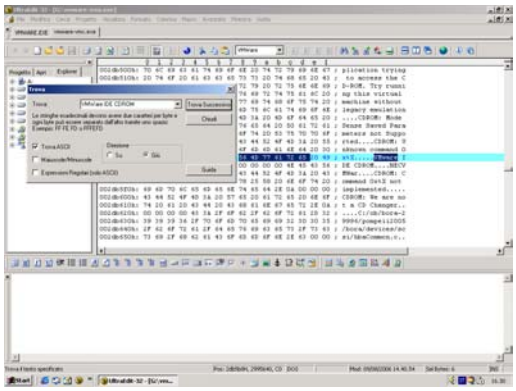


**Figura 4.4 Schermata di Ultraedit**

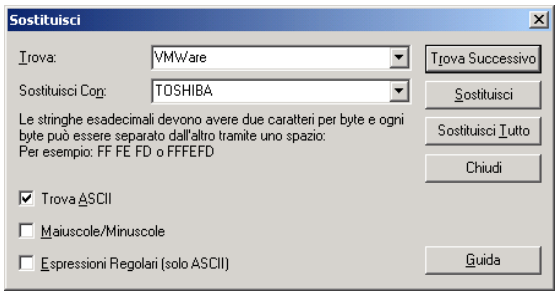



**Figura 4.5 Dialog box per la ricerca di stringhe**

Sostituiamo la stringa VMware con quella relativa al nome di un costruttore di componenti, ad es. TOSHIBA. Dopo aver salvato il file con le modifiche, riavviamo la macchina virtuale e a questo punto saranno scomparse le tracce relative a VMware almeno per quanto riguarda le stringhe identificative dei dispositivi hardware. In alternativa sui sistemi Linux/Unix si possono applicare direttamente delle patch come quella scaricabile dal sito French HoneyNet Project[1], che permette di risolvere alcune problematiche di Vmware:

1) **I/O backdoor**

   Viene disabilitata impedendo ad un eventuale attacker il suo utilizzo per assumere il controllo del sistema.

[1] http://honeynet.rstack.org/tools.php

2) **MAC address**

   Viene consentita la modifica del valore di default OUI 00:0c:29 con uno qualsiasi a scelta (Si consiglia di usare un MAC valido appartenente ad una scheda in proprio possesso).

3) **Adattatore video**

   La patch provvede alla sostituzione integrale del bios della scheda video utilizzata nell'ambito della macchina virtuale oltre a modificarne la stringa identificativa.

4) **CDROM**

   Non viene applicata alcuna patch, ma semplicemente creato un link al dispositivo fisico presente sul sistema.

D'altro canto la modifica dell'indirizzo MAC risulta estremamente semplice. Anche in questo caso si consiglia di utilizzare un valore corrispondente ad una delle schede in proprio possesso in modo da renderlo realistico. In Windows sono disponibili dei tool specifici come SMAC[2] che permettono la modifica di tali indirizzi, oltre ad essere in grado di risalire al costruttore della scheda stessa.

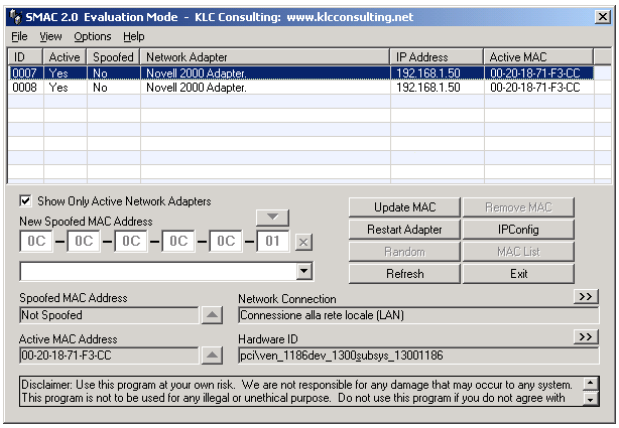



**Figura 4.6 SMAC Tool**

Mentre in Linux basta eseguire i seguenti comandi:

```
% ifconfig eth0 down hw ether 00:00:00:00:00:01
% ifconfig eth0 up
```

[2] http://www.klcconsulting.net/smac

### 4.2.3 Software di virtualizzazione: UML User-Mode Linux

Esistono varie tecniche per identificare la presenza di una macchina virtuale UML. Il modo più diretto consiste nel ricercare delle tracce evidenti all'interno dei file di configurazione del sistema, ad esempio esaminando **`/proc/cpuinfo`**.

```
# cat /proc/cpuinfo
processor               :0
vendor_id               :User Mode Linux
model name              :UML
mode                    :tt
host                    :Linux default 2.4.23
bogomips                :725.81
```

**Figura 4.7 Cartella /proc/cpuinfo di una macchina UML**

Le risorse UML sono identificate da un tag specifico **`udb`**. Un altro modo consiste nel verificare l'indirizzo del puntatore alla locazione corrispondente all'***end of stack***. In un kernel "tradizionale" tale valore è **`0xC0000000`** mentre in UML risulta modificato. A tal proposito basta esaminare il file **`/proc/self/maps`** che mostra le informazioni circa le varie aree di memoria mappate e i permessi di accesso dei vari processi in esecuzione.

### 4.2.4 UML: Contromisure

L'impiego di UML nell'ambito degli honeypot ha richiesto l'introduzione di funzionalità specifiche, non originariamente previste, orientate a rendere sempre più trasparenti le attività di data capture incrementando allo stesso tempo il livello di sicurezza. In particolare si tratta di:

- **HPPFS Honeypot Proc Filesystem**
  Sostituisce le informazioni contenute nella directory **`/proc`** di una macchina UML per evitare che possa essere identificata come tale.
- **SKAS Separate Kernel Address Space**
  Modalità di esecuzione del kernel di UML in uno spazio di memoria riservato per impedirne qualsiasi modifica da parte dei processi utente.
- **TTY logging**

Consente di acquisire il traffico in transito attraverso i terminali (tty teletype) su file all'interno dell'host in modo da rendere l'operazione trasparente.

#### 4.2.4.1 SKAS Separate Kernel Address Space

Per impedire che un aggressore possa scrivere nell'area di memoria riservata al kernel di UML, ottenendo così l'accesso all'host é stato introdotto ***skas***, ovvero una patch che applicata al kernel dell'host consente al kernel UML e ai suoi processi di girare in un differente spazio di indirizzi. In questo modo i dati del kernel UML diventano invisibili agli altri processi e quindi anche allo stesso aggressore. Tra i principali effetti visibili si nota una riduzione del numero di processi visualizzati attraverso il comando **`ps`** e un incremento delle prestazioni.

#### 4.2.4.2 HPPFS Honeypot Proc Filesystem

Legge la cartella **`/proc`** sul sistema host e ne copia il contenuto nella directory **`/proc`** del sistema guest (quello che gira in UML) consentendo all'amministratore di modificare le informazioni in essa contenute per riprodurre un ambiente molto più realistico. La directory **`/proc`** consente agli utenti di vedere e modificare le impostazioni del kernel attraverso una serie di file virtuali. Un aggressore che abbia compromesso il sistema può stabilire che si tratta di un honeypot esaminando tale directory e le informazioni relative a kernel e hardware. Quindi l'obiettivo di hppfs è creare l'illusione di un sistema reale popolando la directory **`/proc`** di UML con false informazioni costruite ad hoc dall'amministratore del sistema oppure copiando quelle presenti sul sistema host.

#### 4.2.4.3 TTY logging

Uno dei problemi principali del data capture è l'acquisizione dei comandi digitati dall'attacker, soprattutto in presenza di connessioni cifrate, che rendono del tutto inutile qualunque attività di sniffing del traffico di rete. UML cerca di risolvere il problema applicando una patch al driver tty in modo da salvare i dati acquisiti non più all'interno della macchina virtuale, ma direttamente sull'host. In questo modo l'attività non viene rilevata all'interno dell'honeypot e non viene generato alcun

traffico di rete. E' anche possibile salvare i dati in una tabella del database MySQL per effettuare delle analisi più dettagliate.

## 4.3 Rilevare la presenza di un honeywall

Nelle honeynet la presenza di un ***honeywall*** è indispensabile per svolgere un'azione di contenimento degli attacchi che possono sfruttare il sistema compromesso come testa di ponte e allo stesso tempo per acquisire maggiori informazioni in maniera trasparente. L'adozione di particolari misure di sicurezza può rivelarsi un'arma a doppio taglio in grado di facilitare il lavoro di riconoscimento di un sistema trappola da parte di un aggressore particolarmente preparato.

### 4.3.1 Restrizione del traffico in uscita

Questa strategia orientata ad evitare il rischio di attacco di ulteriori sistemi a partire da un honeypot compromesso può essere facilmente rilevata attivando un elevato numero di connessioni in uscita e osservando la presenza di una qualche limitazione.

### 4.3.2 Snort_inline

La sua presenza può essere facilmente rilevata costruendo dei pacchetti ad hoc contenenti un payload dannoso che sicuramente comporterebbe la loro eliminazione da parte di Snort_inline. Inviando pacchetti "maliziosi" verso un sistema target di cui si ha il controllo è possibile stabilire se tali pacchetti sono stati alterati in qualche modo o addirittura scartati.

### 4.3.3 Identificazione e disabilitazione di Sebek

Abbiamo già visto che Sebek è un'applicazione di tipo client/server utilizzata come strumento fondamentale per la cattura di informazioni sulle attività intrusive

all'interno di un honeypot. Basato sulla tecnologia dei rootkit[3] permette l'acquisizione dei dati a cui si ha accesso tramite la primitiva ***read()***. Sebek opera nello spazio riservato al kernel per cui è in grado di accedere alla maggior parte delle comunicazioni in maniera non criptata. In questo modo può acquisire le sessioni SSH (Secure Shell), recuperare i file copiati attraverso SCP (Secure Copy) e registrare le password impiegate dagli intrusi. Questi dati vengono inviati attraverso il protocollo UDP verso il server di raccolta che ne costituisce l'altra componente. La trasmissione avviene modificando il kernel in modo da nascondere tali pacchetti in uscita, per evitare che l'aggressore si accorga del flusso dati. In più vengono opportunamente alterati i contatori e le strutture utilizzate nella comunicazione in modo da renderla estremamente difficile da individuare. Resta comunque possibile rilevare la presenza di Sebek riferendosi al livello di rete; poiché Sebek registra qualunque attività correlata alla primitiva ***read()*** e invia i dati attraverso la rete si evidenzia un notevole incremento del traffico in uscita. Si può procedere generando migliaia di chiamate alla ***read()*** per creare un effettivo congestionamento della rete e utilizzare un banalissimo comando **`ping`** per osservare il valore medio di RTT. Vari studi al riguardo hanno evidenziato mediamente un incremento da 0.7 ms a 4800 ms.

Una possibilità concreta di disabilitazione consiste nell'effettuare il rebooting del sistema, visto che Sebek non può essere caricato automaticamente allo startup, a meno che non si proceda ad una personalizzazione degli script di shell, lasciando in questo modo una traccia evidente della sua presenza e facilitando la vita all'aggressore. Il metodo più utilizzato per mettere fuori gioco Sebek consiste nell'evitare l'uso della primitiva ***read()***, sostituendola con la funzione ***mmap()*** che mappa i file direttamente in memoria, aggirando di fatto le attività di logging, sebbene non tutti i file, come i device file o quelli contenuti nella directory **`/proc`**, possano essere gestiti in questo modo. Infine è stata proposta una soluzione più elegante che rappresenta una sorta di suicidio di Sebek. Infatti riuscendo ad ottenere l'indirizzo della funzione ***cleanup()*** si può ripristinare lo stato iniziale della tabella delle system call disabilitando ogni attività di logging.

[3] Insieme di strumenti in grado di conservare i privilegi acquisiti in una precedente intrusione nascondendo file, processi e creando delle backdoor di accesso al sistema che l'aggressore è in grado di sfruttare senza essere scoperto.

# 4. Tecniche anti-honeypot e analisi forense di un attacco

## 4.4 Analisi forense di un attacco

Data la facilità di alterazione del materiale digitale occorre prestare maggiore attenzione e seguire una metodologia di lavoro scientifica. Oltretutto i risultati dell'analisi possono servire in un contesto giudiziario e devono avere valore probatorio. Per questo motivo parleremo in primo luogo di ***informatica forense*** ( o ***computer forensics*** per gli anglosassoni) esaminandone gli aspetti caratterizzanti, gli strumenti utilizzati e le metodiche.

## 4.5 Informatica forense

Con questo termine viene definita la disciplina che si occupa di tutte le attività rivolte all'analisi e alla soluzione dei casi di criminalità informatica, comprendendo sia i reati commessi attraverso l'uso di strumenti informatici che quelli perpetrati verso gli stessi. Gli obiettivi principali che si propone sono l'acquisizione, conservazione, documentazione e analisi dei dati memorizzati su un qualsiasi supporto informatico. Più in dettaglio si cercano le strategie migliori per l'acquisizione delle prove informatiche senza produrre alcuna alterazione delle stesse o del contesto in cui si trovano, garantendone l'integrità e provvedendo all'analisi del materiale acquisito. Spesso viene erroneamente considerata come una nuova branca della sicurezza informatica, essendo venuta alla ribalta negli ultimi anni in conseguenza della crescita dei crimini informatici. In realtà la sua data di nascita risale al 1984 quando il laboratorio scientifico dell'FBI decise di sviluppare dei programmi specifici per l'esame dei supporti digitali. Un'altra data di fondamentale importanza è il 1994 quando il Dipartimento di Giustizia americano pubblicò un insieme di linee guida che per la loro accuratezza e completezza sono diventate uno standard di riferimento nel settore.

## 4.6 Processo forense e catena di custodia

Il processo di analisi forense comincia nel momento stesso in cui ci si interroga circa gli eventi accaduti su un computer o un generico dispositivo digitale.

# 4. Tecniche anti-honeypot e analisi forense di un attacco

La prima attività critica che merita una particolare attenzione è la stesura di un rapporto dettagliato che illustri accuratamente tutte le azioni svolte e che rappresenta a sua volta una prova di valore legale. La preservazione delle prove informatiche è fondamentale per non invalidare tutto il lavoro di indagine a causa di una banale leggerezza. Per questo motivo bisogna attenersi scrupolosamente ad alcuni principi cardine dettati dal buon senso e maturati in anni di esperienza.

- **Principio di rapidità**
  Le prove devono essere acquisite nel minor lasso di tempo possibile dall'evento che si vuole ricostruire, per evitare un qualsiasi inquinamento o addirittura la loro perdita.

- **Principio del congelamento**
  Tutti i supporti informatici oggetto di indagine, cioè contenenti dati potenzialmente rilevanti per gli investigatori, devono essere "congelati", ossia va impedita qualsiasi loro alterazione volontaria o involontaria.

- **Principio della catena di custodia**
  Bisogna garantire e documentare tutte le fasi di gestione delle prove informatiche dalla loro acquisizione fino alla presentazione in tribunale.

- **Principio della controllabilità e ripetibilità**
  Tutte le attività di indagine eseguite devono essere ripetibili da periti e consulenti di parte producendo i medesimi risultati.

### 4.6.1 Acquisizione dei dati

Le indagini informatiche possono svolgersi direttamente sul "luogo del delitto" oppure in laboratorio. In entrambi i casi una delle prime attività da svolgere è l'isolamento della scena del crimine per evitare l'accesso a persone non autorizzate che possano inquinarle e nello stesso tempo ricercare prove e descrivere l'ambiente servendosi anche di fotografie e riprese filmate. Andrebbero ricercati appunti, agende

e qualunque altro elemento utile all'indagine, ad esempio per ricavare password, codici, ecc. Il contesto ideale sarebbe costituito dalla possibilità di accedere al sistema informatico ancora attivo in modo da poterne esaminare la memoria fisica riuscendo a ricavare molte informazioni utili come gli ultimi comandi eseguiti, i processi in esecuzione, gli accessi alla rete, tutte informazioni critiche che si perdono quando il sistema viene spento e che bisogna salvare su supporti permanenti. In generale andrà deciso volta per volta se accedere alla macchina accesa in loco o togliere direttamente la corrente in modo da creare una fotografia istantanea del sistema, procedendo all'analisi in un secondo momento. Al riguardo esiste un acceso dibattito tra gli esperti soprattutto tenendo presente che un normale spegnimento (procedura di ***shutdown***) comporta comunque una alterazione o cancellazione di molti dati (es. file di swap, file temporanei ecc). Già in fase di sequestro le macchine e i supporti devono essere opportunamente imballati per preservarli da urti, fonti di calore, umidità. Devono essere apposte etichette e sigilli che potranno essere rimossi solo dal personale autorizzato. Va inoltre redatto un verbale di sequestro riportando tutti i dati identificativi come numero di serie, marca e modello dei supporti.

### 4.6.2 Gestione delle prove

Molto importante risulta la gestione delle prove raccolte, il loro trasporto e archiviazione per evitare che vengano alterate, perse o che comunque si possa mettere in discussione la loro integrità. Prima di procedere con l'analisi occorre partire da una copia integrale bit a bit del supporto oggetto dell'indagine con l'intento di evitare qualsiasi rischio di perdita dei dati originali. Anche un semplice boot del sistema può alterarli in maniera significativa facendo perdere il loro valore probatorio. Dunque qualsiasi futura attività verrà svolta esclusivamente sulle copie preservando l'originale. Riguardo alla creazione dei file immagine va evidenziato che i classici strumenti di backup si limitano a copiare i file e non i dati di ambiente che al contrario possono risultare estremamente utili ai fini dell'indagine. In Windows tali dati risiedono nel ***file di swap*** (file di scambio) mentre in Unix esiste una partizione ad hoc analogamente denominata. Inoltre molte informazioni interessanti si possono trovare nello ***slack space***, costituito dallo spazio inutilizzato

all'interno dei cluster. Esistono dei tool che consentono di impiegare questo spazio per occultare dei dati. Andrebbe esaminato anche lo spazio non allocato in cui si possono ritrovare file cancellati o altro materiale interessante. Questi sono alcuni dei motivi che giustificano la necessità di una copia bit a bit dell'intera superficie del disco. Pur avendo assunto che le analisi vanno effettuate sulle copie e non sui dati originali, gli strumenti impiegati devono comunque garantire che le copie stesse non siano oggetto di alterazione: quindi bisogna disporre di un meccanismo di verifica che assicuri la genuinità dei dati originali (le prove acquisite) e delle loro copie. Normalmente si ricorre ad un hash dell'immagine del supporto. Un ***algoritmo di hashing*** parte da una sequenza arbitraria di dati, ad esempio il contenuto del disco e produce una sequenza molto più breve (detta appunto ***hash***) che è in stretta correlazione con quella di partenza. In caso di modifica dei dati la relativa sequenza hash verrà a sua volta modificata, per cui confrontandola con quella originale (da conservare accuratamente) sarà possibile evidenziare la presenza di eventuali alterazioni. Si parla nello specifico di ***catena di custodia***, cioè della descrizione dettagliata di tutti i passaggi subiti dalle prove durante la loro gestione in modo da poter risalire a coloro che le hanno maneggiate e garantire che non si siano prodotte alterazioni dal momento del loro sequestro fino alla presentazione in tribunale. Naturalmente al materiale dovrebbe accedere solo chi dispone di una adeguata autorizzazione e lo stesso dovrebbe essere conservato in ambienti sicuri e controllati, in modo da evitare qualsiasi eccezione in fase dibattimentale tale da invalidare il lavoro svolto. Riguardo all'autenticazione delle prove bisogna dimostrare che tutte le operazioni sono state eseguite senza modificare il sistema, certificando la sequenza temporale e le modalità con cui sono state condotte.

### 4.6.3 Analisi

Successivamente si procede all'analisi di tutto il materiale acquisito per ricostruire accuratamente gli eventi accaduti. Come illustrato in precedenza, la ricerca all'interno dei supporti informatici deve riguardare sia lo spazio allocato sia quello inutilizzato, in quanto la cancellazione di un file normalmente lo rende inaccessibile ma non elimina effettivamente i dati fino ad una successiva

sovrascrittura, permettendo il suo ripristino (nella maggior parte dei casi) con opportuni strumenti. L'analisi delle prove comporta una serie di operazioni che richiedono vari tool e devono essere eseguite secondo una opportuna sequenza. Quindi risulta utile preparare una checklist dei compiti da svolgere in modo da eseguirli e documentarli dettagliatamente. In particolare occorre:

- Inventariare accuratamente tutto il materiale sequestrato
- Esaminare lo stato di ogni supporto
- Ispezionare documenti, agende, cassetti o altro alla ricerca di eventuali password
- Ricostruire le attività svolte in Rete
- Ricercare la presenza di eventuali malware
- Verificare la sequenza di operazioni eseguite (ultimi file aperti o cancellati, registrazioni nei log di sistema ecc.)
- Ripetere le analisi per accertarsi della correttezza dei risultati ottenuti

Seguendo una metodologia scientifica si prevengono eventuali perdite di dati ma allo stesso tempo si attribuisce alle prove una credibilità tale da poter essere presentate in giudizio.

### 4.6.4 Gli strumenti

La complessità dei sistemi informatici da esaminare comporta l'impiego di soluzioni hardware e software adeguate a supportare l'indagine forense.
Le funzionalità principali richieste a tali strumenti sono:

- ricerca rapida all'interno dei supporti digitali (dischi, CD, DVD, ZIP, schede di memoria) comprese le aree inutilizzate;
- produzione di copie dei dischi a basso livello (settore per settore);
- supporto di vari filesystem;
- analisi dei dati con varie modalità di codifica (ASCII, esadecimale, binario);
- recupero file cancellati;
- stampa di report delle analisi sulla base di opportuni parametri.

Dal punto di vista hardware si potrebbe pensare all'utilizzo di un notebook per la sua portabilità e manegevolezza, sebbene nella maggior parte dei casi non si possa considerare la scelta migliore, soprattutto per le limitate capacità di collegamento con dispositivi esterni. Una "generica soluzione" dovrebbe essere dotata almeno delle seguenti caratteristiche:

- elevata disponibilità di memoria (> 1 GB);
- ampia gamma di connessioni (Firewire, USB 2.0, SCSI, ecc.);
- disco rigido di grandi dimensioni per lavorare con le immagini dei sistemi acquisiti;
- connessione di rete;
- sistema di backup basato su CD o DVD.

In ambito professionale si utilizzano delle stazioni di lavoro appositamente progettate allo scopo e in grado di soddisfare tutte le necessità dell'investigatore.

Sul fronte software esiste una varietà di soluzioni ancora più ampia. Accanto ai prodotti commerciali è possibile trovare molti progetti open source. Ancora una volta bisogna sottolineare il contributo che Linux può apportare in questo campo. Le principali caratteristiche che lo rendono idoneo ad essere una piattaforma di base su cui costruire dei prodotti molto complessi sono:

- gestione degli oggetti in forma di file, compresi i dispositivi hardware;
- supporto di vari filesystem;
- analisi dei sistemi senza produrre alterazioni potendo montare le partizioni in modalità read-only;
- concatenazione dei comandi;
- monitoraggio delle attività e dei processi in esecuzione;
- possibilità di creare e configurare opportunamente dei supporti avviabili (CD bootable).

Oltretutto i tool di Linux sono open source e questo contribuisce alla riduzione dei costi. Senza entrare in merito alla diatriba tra i sostenitori di tale filosofia e quelli del software commerciale occorre dire che i primi ne sostengono il primato nell'ambito dell'indagine forense in quanto la disponibilità del codice sorgente può fugare ogni dubbio circa eventuali manipolazioni effettuate in fase di analisi. Esiste una gran

varietà di distribuzioni dedicate alla computer forensics e liberamente scaricabili da Internet.

## 4.7 Analisi di una macchina compromessa (honeypot)

I dati acquisiti da un honeypot (log, pacchetti di dati, file, ecc.) hanno realmente valore solo se si riesce a tradurli in informazioni utili, servendosi dell'ampia varietà di strumenti visti in precedenza. In questo modo si possono ricavere motivazioni, strumenti, strategie utilizzate dagli attacker.

I ***log dei firewall*** permettono di tracciare tutte le connessioni in ingresso e uscita dal nostro honeypot. Già sappiamo che il traffico riguardante tali sistemi è potenzialmente sospetto, ma una particolare attenzione va riservata alle connessioni in uscita in quanto sintomo evidente di una compromissione del sistema.

Dai log degli IDS si possono ottenere informazioni circa i tentativi di attacco di cui è oggetto il sistema. Inoltre, in funzione delle caratteristiche del prodotto utilizzato, è possibile raggruppare alert simili, traffico di rete ed eventi in ordine cronologico, ricostruendo la sequenza degli eventi accaduti, generando report dettagliati e statistiche sugli attacchi.

Tutte le attività di un sistema vengono registrate localmente in vari modi a seconda del sistema operativo utilizzato. Quasi tutte le piattaforme sono in grado di svolgere questa attività di logging in remoto, permettendo di scoprire le modalità con cui un aggressore ha ottenuto l'accesso al sistema, la provenienza dell'attacco, i servizi attivati o interrotti. Inoltre è possibile confrontare il sistema attaccato con quello originale in modo da individuare eventuali modifiche o cancellazioni relative ai file locali.

Esistono molti altri tool quali il già citato ***Sebek*** che consentono l'acquisizione delle sequenze di comandi impartiti dall'attacker e le risposte generate dal sistema oltre a creare delle copie dei file caricati sull'honeypot. Infine l'analisi dei pacchetti di rete catturati rivela moltissime informazioni circa i protocolli utilizzati, i tentativi di scansione, gli attacchi condotti e le vulnerabilità sfruttate.

# 5. Laboratorio: mobile wireless honeypot

Le tecnologie wireless hanno avuto un grande sviluppo negli ultimi anni e sono ormai parte integrante della nostra vita quotidiana visto che abbiamo a che fare continuamente con dispositivi che comunicano senza fili: dai pc, alle stampanti, alle fotocamere, ai telefoni cellulari ecc. Si pone quindi il problema di garantire la sicurezza di tali comunicazioni. Nonostante l'adozione di protocolli specifici (Wep, WPA) sono sempre possibili varie tipologie di attacco in grado di raggiungere il loro scopo a seconda del livello di protezione adottato e delle competenze del potenziale intruso. Sfortunatamente per incuria o superficialità molte reti sono addirittura lasciate aperte o con le impostazioni di fabbrica consentendo a chiunque di accedervi senza difficoltà per rubare dati preziosi o sfruttare l'accesso ad Internet. A questo va aggiunta una opinione alquanto diffusa circa la minore pericolosità di tali attacchi rispetto a quelli sferrati attraverso Internet dovendo trovarsi fisicamente in prossimità della wlan a differenza di quanto accade con la Rete, dove un aggressore può agire comodamente sulla poltrona di casa dall'altro capo del mondo. Effettivamente la realtà è molto diversa: basta armarsi di un notebook dotato di scheda wifi, un software di sniffing (***Netstumbler*** o ancora meglio ***Kismet***), eventualmente di un ricevitore GPS e fare un giro in macchina per scoprire che siamo letteralmente circondati da decine di reti wireless più o meno sicure.

Nel seguito è descritta una indagine conoscitiva condotta allo scopo di studiare il livello di diffusione del wifi e il comportamento tipico degli utenti.

Purtroppo o per fortuna (a seconda dei punti di vista) gli access point più moderni sono dotati di potenze emissive molto elevate e spesso dispongono di più antenne omnidirezionali che diffondono il segnale in un'area molto vasta e difficilmente delimitabile. Inoltre i wardrivers hanno dalla loro la possibilità di utilizzare schede molto potenti (fino a 100 mW e oltre) dotate di attacchi per antenne esterne. In commercio ne esistono diversi modelli a prezzi più che accessibili, oltre alla descrizione in Internet di progetti autocostruiti. Con questa attrezzatura è semplicissimo riuscire a

potenziare il livello di ricezione lavorando a distanza di sicurezza dal possibile obiettivo.

I ***wireless honeypot*** possono rivelarsi utili in questo campo per acquisire informazioni ed elaborare statistiche sugli attacchi determinando il livello di competenza degli aggressori e le loro strategie, oltre a rappresentare un ottimo deterrente in grado di distogliere l'attenzione dai sistemi reali.

## 5.1 Il progetto

L'idea alla base del "***Mobile Wireless Honeypot***" è la realizzazione di un sistema capace di trasformare un notebook in una piattaforma per lo studio delle attività di wardriving, applicando i principi degli honeypot ed utilizzando esclusivamente strumenti open-source dalle caratteristiche paragonabili ai prodotti commerciali. Questi ultimi si basano su soluzioni proprietarie che non consentono alcuna modifica, personalizzazione ed estensione in base alle proprie esigenze.

## 5.2 Obiettivi

Accanto al già citato impiego nell'ambito dello studio del wardriving, per cui il wireless honeypot può rientrare nella categoria dei sistemi di ricerca, esiste un'ulteriore opportunità di impiego nei sistemi di produzione, sfruttando l'idea dei ***rogue ap*** (ap non autorizzati). In sintesi creando un honeypot capace di simulare una rete aziendale è possibile utilizzarlo sia per coprire il segnale della rete "autorizzata", amplificando quello esterno, sia monitorarlo per scoprire eventuali falle nei meccanismi di sicurezza adottati senza mettere a rischio le risorse aziendali.

## 5.3 Indagine conoscitiva

Preliminarmente è stata condotta un'indagine conoscitiva sulle reti wifi presenti nell'area di residenza per raccogliere informazioni più dettagliate sulle abitudini degli utenti, individuando le aree a maggiore densità di traffico wireless, in cui di

conseguenza è elevata la probabilità di trovare dei wardriver. Principalmente la ricerca si è focalizzata sui seguenti aspetti:

- tipologia delle reti
- canali utilizzati
- SSID
- impiego della cifratura

Armati di un notebook dotato di sistema operativo Linux, scheda wireless PCMCIA D-link DWL-G650 e software di sniffing Kismet, girando in auto per circa 2 ore sono state individuate 222 reti.

| Totale | Canale |
|---|---|
| 64 | 1 |
| 63 | 6 |
| 40 | 11 |
| 25 | 0 |
| 6 | 3 |
| 6 | 10 |
| 3 | 2 |
| 3 | 5 |
| 3 | 9 |
| 3 | 12 |
| 2 | 4 |
| 2 | 8 |
| 1 | 7 |

| Totale | Tipologia rete |
|---|---|
| 195 | infrastructure |
| 22 | probe |
| 4 | ad-hoc |

La maggior parte di esse opera sui canali 1, 6 e 11, mentre quelli meno utilizzati sono 4, 8 e 7. Il canale 0 corrisponde alle probe request (ricerca di reti). La quasi totalità delle reti individuate è configurata come infrastructure per cui utilizza un access point.

| Totale | Cifratura |
|---|---|
| 79 | WEP,TKIP,WPA |
| 70 | WEP |
| 67 | nessuna |
| 4 | WEP,TKIP,WPA,AES-CCM |
| 1 | WEP,TKIP,WPA,PSK,AES-CCM |

Sul fronte della sicurezza l'uso della cifratura è sicuramente più diffuso rispetto al passato, ma ancora 1/3 delle reti risultano aperte e l'adozione di algoritmi più robusti

come AES è abbastanza limitata. Proprio in corrispondenza di 3 reti aperte è stato possibile rilevarne l'indirizzo IP (in tutti i casi 192.168.0.1).

| Numero | ESSID |
|---|---|
| 20 | <no ssid> |
| 11 | divittorio.wifi |
| 9 | default |
| 5 | DLINK_WIRELESS |
| 5 | linksys |
| 4 | NETGEAR |
| 4 | -w |
| 3 | 3Com |
| 3 | G604T_WIRELESS |
| 3 | THSW |
| 3 | wlan-ap |

| Numero | ESSID |
|---|---|
| 3 | ZyXEL |
| 2 | AK2471joy |
| 2 | belbridge |
| 2 | belkin54g |
| 2 | Casa |
| 2 | CCSWLESS |
| 2 | conexant |
| 2 | linksys_SES_26069 |
| 2 | PSWL |
| 2 | Rete Supersonica |
| 1 | Altre 131 reti |

Analizzando gli SSID utilizzati per l'identificazione delle reti si scoprono delle cose interessanti: più di 2/3 sono gli identificatori distinti, mentre alcuni risultano ripetuti. Circa il 10% corrisponde a cognomi o denominazioni di aziende e studi professionali. Un altro 10% corrisponde agli SSID impostati in fabbrica, per cui è ipotizzabile che siano mantenute tutte le altre impostazioni di default.
Le informazioni così ricavate sono risultate utili per le successive impostazioni del wireless honeypot.

## 5.4 Architettura del wireless honeypot

Un wireless honeypot necessita di almeno tre componenti ciascuna delle quali svolge un ruolo fondamentale e complementare rispetto alle altre:

- access point
- honeypot
- sniffer

L'***access point*** rappresenta il nodo centrale della rete wlan a cui i client devono associarsi per accedere alle risorse disponibili. L'***honeypot*** costituisce l'ambiente simulato e deve risultare particolarmente interessante per attrarre i potenziali intrusi, mantenendo soprattutto una elevata coerenza dei servizi con il contesto in cui sono

integrati. Infine lo ***sniffer*** è indispensabile per la cattura del traffico scambiato tra i client (potenziali intrusi) e l'access point per una successiva analisi.

## 5.5 Configurazione hardware/software

L'intero sistema è installato su un notebook basato su processore Intel Pentium IV 1.73 Ghz dotato di 1 GB di memoria RAM e hard disk da 30 GB, pur essendo sufficienti solo 256 MB di RAM e 8-10 GB di spazio.

La scelta del sistema operativo è ricaduta sulla distribuzione ***Linux Suse 10.1*** non solo per la sua stabilità ma anche per la ricca dotazione di pacchetti disponibili (in particolare quelli utili allo sviluppo del progetto) e l'efficiente gestore dell'installazione ***YAST***. Altro punto di forza è la capacità di riconoscimento dell'hardware più recente. In questo modo è stato possibile predisporre un prototipo in tempi più ridotti rispetto alla compilazione manuale dei sorgenti.

### 5.5.1 Access Point

Rendere il sistema facilmente trasportabile ha richiesto la simulazione di un access point tradizionale (che per giunta avrebbe richiesto una fonte di alimentazione esterna) con una diversa soluzione. Fortunatamente questo compito può essere svolto direttamente dalla scheda wireless purché sia possibile impostarla in modalità master, cosa che dipende esclusivamente dal driver utilizzato.

Il progetto di riferimento in questo senso è ***HostAp***[1] che purtroppo supporta solo i chipset della serie Prism, attualmente abbastanza rari da trovare nei prodotti più recenti. Un'attenta ricerca su Internet ha consentito l'individuazione di un'altra soluzione open source denominata ***MadWifi***[2] (***Multiband Atheros Driver for Wifi***) per chipset Atheros, che si è rivelata estremamente utile, avendo a disposizione una scheda di rete D-Link equipaggiata con tale chipset. L'access point virtuale viene creato con i seguenti comandi:

```
% wlanconfig ath0 create wlandev wifi0 wlanmode ap
% iwconfig ath0 essid "StudioAzzurra" channel 1
```

[1] http://hostap.epitest.fi
[2] http://madwifi.org

```
% ifconfig ath0 192.168.1.50 netmask 255.255.255.0
```

In dettaglio si impostano i seguenti parametri:

- ESSID
- Canale
- Indirizzo IP e subnet mask

Per facilitare l'accesso alla rete attiviamo il servizio ***DHCP*** (***Dynamic Host Configuration Protocol***) così da fornire automaticamente un indirizzo IP ai client che si associano. In questo modo otteniamo anche la lista degli indirizzi MAC utilizzati dai client, informazione utilissima per analizzare meglio il traffico catturato. In primo luogo dobbiamo accedere al file **`/etc/dhcpd`** per abilitare il servizio sull'interfaccia di rete utilizzata come AP (nel nostro caso "**ath0**"). Quindi procediamo alla configurazione dei parametri in **`/etc/dhcpd.conf.`**

```
ddns-update-style none;
ddns-updates off;
option domain-name "studiolegaleazzurra.it";
subnet 192.168.1.0 netmask 255.255.255.0 {
  default-lease-time 3600;
  max-lease-time 3600;
  option subnet-mask 255.255.255.0;
  option broadcast-address 192.168.1.255;
  option routers 192.168.1.1;
  option domain-name-servers 212.216.112.112;
  option domain-name-servers 212.216.172.62;
  range 192.168.1.100 192.168.1.200;
}
```

Fissiamo il range di indirizzi da assegnare in modo da escludere a priori quelli staticamente associati agli host virtuali per evitare eventuali conflitti; impostiamo l'indirizzo del router che fungerà da gateway per l'accesso ad Internet e il lease-time ovvero l'intervallo di tempo durante il quale il client potrà "ricevere in affitto" l'indirizzo IP. Sul server DHCP all'interno del file **`/var/lib/dhcp/db/dhcpd.leases`** viene memorizzato automaticamente il database degli indirizzi assegnati. Le informazioni includono la durata dell'affitto, l'assegnatario, la data inziale e finale del periodo di affitto e l'indirizzo MAC della scheda di rete usata per l'acquisizione dell'IP.

### 5.5.2 Honeypot

Il primo problema ha riguardato l'adozione del livello di interazione più adeguato allo scopo. La decisione è ricaduta sul livello medio soprattutto per la maggiore sicurezza che è in grado di offrire anche se a discapito della facilità di configurazione dell'ambiente di lavoro e con tutte le limitazioni relative ai servizi utilizzabili.

L'honeypot è realizzato sfruttando Honeyd e una serie di script realizzati dal ***French Honeynet Project***[3] opportunamente modificati in base alle specifiche esigenze.

La rete simulata è costituita da:

- 1 Access point
- 1 Router per l'accesso ad Internet
- 1 Server Linux per la Intranet
- 3 Client Windows (XP Professional SP1, Home, 98 SP1)

Di seguito é riportata la configurazione di tutti gli apparati, per ciascuno dei quali è indicata la personalità secondo le specifiche del file di Nmap, i servizi attivi e la modalità di implementazione attraverso file script oppure proxy verso servizi reali. Sebbene l'access point sia simulato dalla scheda di rete, per renderlo più realistico è stata creata una personalità in Honeyd attivando un servizio http per il sito di amministrazione.

[3] http://honeynet.rstack.org

| Apparato | IP | Personalità | Servizi attivi | Implementazione |
|---|---|---|---|---|
| **Router** | 192.168.1.1 | `Zyxel Prestige 642R-11 ADSL router running ZyNOS` | tcp 23 / telnet | router-telnet.pl |
| **Server Intranet** | 192.168.1.10 | `Linux 2.4.18` | tcp 21 / ftp<br>tcp 23 / telnet<br>tcp 25 / sendmail<br>tcp 80 / http | proxy vsftpd<br>telnetd.sh<br>sendmail.sh<br>proxy Apache |
| **Client 1** | 192.168.1.20 | `Microsoft Windows XP Professional` | tcp 21 / ftp<br>udp 137 / netbios name service<br>udp 138 / netbios datagram service | msftp.sh<br>FakeNetbiosNS<br><br>FakeNetBiosDGM |
| **Client 2** | 192.168.1.30 | `Microsoft Windows XP Home Edition` | tcp 21 / ftp<br>tcp 4444<br>udp 137 / netbios name service<br>udp 138 / netbios datagram service | msftp.sh<br>cmdexe.pl<br>FakeNetbiosNS<br><br>FakeNetBiosDGM |
| **Client 3** | 192.168.1.40 | `Microsoft Windows 98 SP1` | tcp 139 / netbios session service<br>udp 137 / netbios name service<br>udp 138 / netbios datagram service | -<br><br>FakeNetbiosNS<br><br>FakeNetBiosDGM |
| **Access Point** | 192.168.1.50 | `Linksys WAP11 or D-Link DWL-900+ wireless AP` | tcp 80 / http<br>udp 67 / bootstrap protocol server<br>udp 69 / bootstrap protocol client | fakelinksys.sh<br>-<br>- |

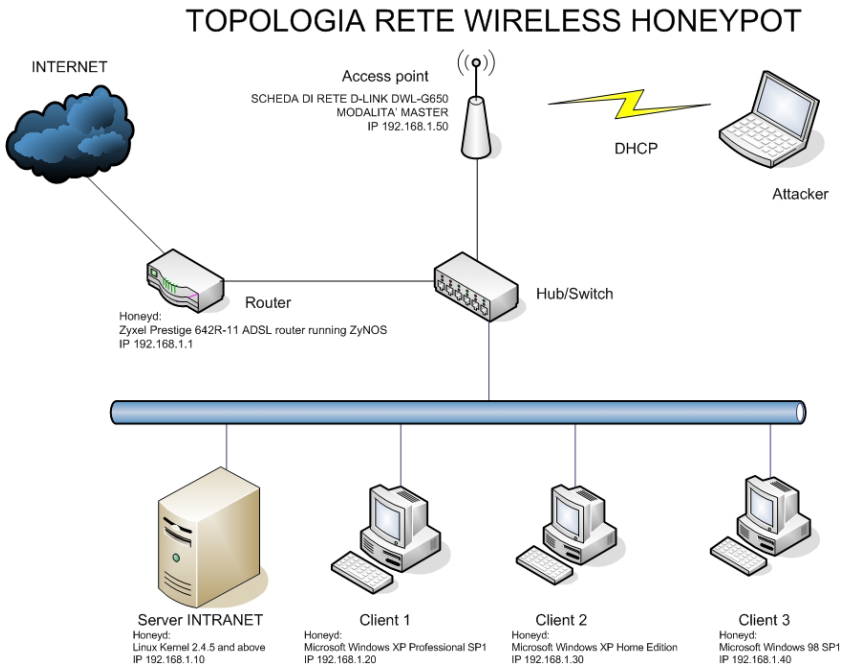



**Figura 5.1 Topologia rete wireless honeypot**

**Server Linux**

La Intranet aziendale simulata dispone di un server al quale sono affidati i principali servizi erogati agli utenti:

- server web
- server ftp
- server posta elettronica

Riguardo al server web si è deciso di non ricorrere ad uno script di emulazione per alcuni problemi di fingerprinting riscontrati durante i test preliminari condotti con Nmap. In particolare Honeyd offre la possibilità di reindirizzare le richieste di un servizio ad una applicazione reale, nello specifico un server Apache in esecuzione sul notebook. Inoltre è più semplice gestire le pagine del sito e personalizzarle in base alle varie esigenze. Quindi è stato creato una sorta di ***captive portal*** per acquisire userid e password utilizzati dagli attacker per accedere alla rete intranet attraverso la procedura di autenticazione. Una opportuna regola di Snort cattura i pacchetti relativi ai form compilati dagli utenti.

```
#Regola per il captive portal
alert tcp $EXTERNAL_NET any -> $HOME_NET 80 (content: "login.cgi";
msg:"Tentativo login";)
```

La procedura di login consta di due pagine: ***index.html*** collocata in **`/srv/www/htdocs`** che presenta il form da compilare e ***login.cgi*** collocata in **`/srv/www/cgi-bin`** che processa i dati restituendo un messaggio di autenticazione fallita.

**Servizio FTP**

La distribuzione Suse mette a disposizione il pacchetto ***VSFTPD*** che implementa un server ftp.

```
% mkdir /var/ftp
% useradd –d /var/ftp ftp
% chown root.root /var/ftp
% chmod og-w /var/ftp
```

Il file di configurazione si trova in **`/etc/vsftp.conf`**.

Il seguente file ***honeyd.conf*** riporta la configurazione completa dei sistemi Honeyd.

```
############################################################################
### Default template.  Viene utilizzato in assenza di uno specifico  ###
### honeypot associato all'indirizzo IP richiesto.                   ###
############################################################################

create default
set default personality "Microsoft Windows XP Home Edition"
set default default tcp action block
set default default udp action block
set default default icmp action block


####################################
### Configurazione access point ####
####################################

create dlink
set dlink personality "Linksys WAP11 or D-Link DWL-900+ wireless AP"
set dlink default tcp action block
add dlink tcp port 80 "sh scripts/fakelinksys.sh"
add dlink udp port 67 open
add dlink udp port 69 open
set dlink ethernet "D-Link"
bind 192.168.1.50 dlink

####################################
### Configurazione router #########
####################################

create router
set router personality "Zyxel Prestige 642R-11 ADSL router running ZyNOS"
set router default tcp action block
set router default udp action block
add router tcp port 23 "perl scripts/unix/telnet/router-telnet.pl"
set router ethernet "Zyxel"
bind 192.168.1.1 router


############################################
### Configurazione Server INTRANET Linux  ##
############################################

create serverlinux
set serverlinux personality "Linux 2.4.18"
set serverlinux default tcp action reset
set serverlinux default udp action reset
set serverlinux default icmp action open

add serverlinux tcp port 21 proxy 127.0.0.1:21
add serverlinux tcp port 23 "sh scripts/unix/telnetd.sh $ipsrc $sport $ipdst
$dport"
add serverlinux tcp port 25 "sh scripts/unix/sendmail.sh $ipsrc $sport $ipdst
$dport"
add serverlinux tcp port 80 proxy 127.0.0.1:80
set serverlinux ethernet "D-Link"
bind 192.168.1.10 serverlinux


############################################################
### Configurazione Client1 Microsoft Windows XP Professional ##
```

```
#################################################################

create client1
set client1 personality "Microsoft Windows XP Professional"
set client1 default tcp action reset
set client1 default udp action reset
set client1 default icmp action reset

add client1 tcp port 21 "sh scripts/win2k/msftp.sh $ipsrc $sport $ipdst
$dport"
add client1 udp port 137 "scripts/win2k/FakeNetbiosNS -d 192.168.1.255 -D
Mshome -N Client1 -H"
add client1 udp port 138 "scripts/win2k/FakeNetbiosDGM -d 192.168.1.255  -H -n
3"
set client1 ethernet "3Com"
bind 192.168.1.20 client1


#################################################################
# Configurazione Client2 Microsoft Windows XP Home Edition ##
#################################################################

create client2
set client2 personality "Microsoft Windows XP Home Edition"
set client2 default tcp action reset
set client2 default udp action reset
set client2 default icmp action reset

add client2 tcp port 21 "sh scripts/win2k/msftp.sh $ipsrc $sport $ipdst
$dport"
# blaster
add client2 tcp port 4444 "scripts/win2k/cmdexe.pl -p winxp -l
/materiale/honeyd/logs/cmdexe"
add client2 udp port 137 "scripts/win2k/FakeNetbiosNS -d 192.168.1.255 -D
Mshome -N Client2 -H"
add client2 udp port 138 "scripts/win2k/FakeNetbiosDGM -d 192.168.1.255  -H -n
3"
set client2 ethernet "3Com"
bind 192.168.1.30 client2


#################################################################
# Configurazione Client3 Microsoft Windows 98 SP1          ##
#################################################################

create client3
set client3 personality "Microsoft Windows 98 SP1"
set client3 default tcp action reset
set client3 default udp action reset
set client3 default icmp action reset

add client3 tcp port 139 open
add client3 udp port 137 "scripts/win2k/FakeNetbiosNS -d 192.168.1.255 -D
Mshome -N Client3 -H"
add client3 udp port 138 "scripts/win2k/FakeNetbiosDGM -d 192.168.1.255  -H -n
3"
set client3 ethernet "3Com"
bind 192.168.1.40 client3
```

### 5.5.3 Sniffer

L'impiego di uno sniffer si dimostra utile nell'acquisizione dei pacchetti di rete scambiati tra l'intruso e l'honeypot per una successiva e completa ricostruzione degli eventi. Dal momento che non è prevista l'adozione della cifratura le comunicazioni avvengono in chiaro e possono essere esaminate con un comune analizzatore di protocollo. Trattandosi di comunicazioni wireless la ricerca dello strumento più idoneo da usare si è concentrata sui ***wifi sniffer***.

Inizialmente si è pensato a Kismet per l'elevata capacità di acquisizione dei pacchetti e di altre informazioni utili, ma in seguito ci si è orientati verso un'altra soluzione open source: ***Snort-Wireless***.

Tale progetto estende le funzionalità della versione standard di Snort per supportare il protocollo 802.11, attraverso l'introduzione di nuove regole nella sezione "***wifi***" per rilevare specifici frame relativi alle attività di autenticazione, deautenticazione e disassociazione, rogue AP, reti Ad-hoc e sniffer come NetStumbler.

I vantaggi rispetto a Kismet sono:

- operatività come IDS (Intrusion Detection System)
- output verso differenti database

A questo si aggiunge la capacità, condivisa con Kismet, di salvataggio del traffico acquisito in formato tcpdump gestibile con i più comuni analizzatori di protocollo.

### 5.5.4 Installazione

Al momento il progetto Snort-Wireless è stato abbandonato e l'unica versione di Snort che supporta tali estensioni è la 2.1.1[4], da estendere attraverso le estensioni ***Snort-Wireless-Patch***[5] e ***Wireless-Database-Patch***[6], quest'ultima necessaria nel caso in cui si desideri memorizzare alert e log in un database.

In alternativa è possibile scaricare tutto l'occorrente direttamente dal seguente indirizzo:

**`http://www.wireless-bern.ch/downloads/snort-wireless-db-2.1.1.tar.gz`**

Procediamo con l'installazione, scompattando in primo luogo il codice di Snort 2.1.1 nella cartella Snort-2.1.1.

[4] http://www.snort.org/dl/old/snort-2.1.1.tar.gz
[5] http://snort-wireless.org/files/snort-2.1.1-wireless.patch.gz
[6] http://www.wireless-bern.ch/downloads/snort-2.1.1-wireless-db.patch

```
% tar xvfz snort-2.1.1.tar.gz
```

Applichiamo le patch per integrare il supporto wireless e quello dei database.

```
% patch -p0 < snort-2.1.1-wireless.patch.gz
% patch -p0 < snort-2.1.1-wireless-db.patch
```

Abilitiamo il supporto per il database MySQL in fase di configurazione:

```
% ./configure --enable-wireless --with-mysql
% make
% make install
```

Procediamo alla creazione del database servendoci del file **`create_mysql`**. In primo luogo aggiungiamo l'utente **`snortusr`** al quale verranno concessi in esclusiva (per ragioni di sicurezza) i privilegi di lettura/scrittura sul database snort.

```
% adduser snortusr

% mysql –h localhost –u root

   mysql> create database snort;
   mysql> grant CREATE, INSERT, DELETE, UPDATE, SELECT on snort.* to
   snortusr@localhost;
   mysql> quit;

% mysql –h localhost –u  snortusr snort < create_mysql
```

Riguardo all'esecuzione di MySQL occorre precisare che Snort-Wireless segnala un errore non trovando il socket di comunicazione nella posizione richiesta (di default MySQL usa **`/tmp/mysql.sock`**) per cui occorre specificare il percorso direttamente sulla riga di comando:

```
% mysqld --user=mysql –socket=/var/lib/mysql/mysql.sock
```

A questo punto va predisposta una seconda interfaccia wireless da abilitare in modalità promiscua (monitor) per catturare tutto il traffico scambiato con l'honeypot, incanalandolo verso Snort. Una caratteristica interessante di Madwifi è la possibilità di creare più interfacce virtuali (denominate **`athX`**, con **`X`** numero progressivo) appoggiandosi ad un solo hardware. Quindi eseguiamo:

```
% wlanconfig ath1 create wlandev wifi0 wlanmode monitor
% ifconfig ath0 up
```

Rispetto all'altra interfaccia non assegnamo alcun indirizzo IP in modo da renderla invisibile, nascondendo la presenza dello stesso Snort.

Quindi procediamo alla configurazione predisponendo il file ***snort.conf***, all'interno del quale sono indicati tra l'altro gli indirizzi IP degli host della rete interna e di quella esterna, i file delle regole e le modalità di importazione dei dati nel database MySQL.

```
# File di configurazione di Snort snort.conf

# Step #1: Impostazione delle reti da controllare
# Vengono specificati gli IP degli host della honeynet
var HOME_NET 192.168.1.0/24

# Rete esterna da cui proviene il traffico corrispondente agli indirizzi
diversi dagli host
var EXTERNAL_NET ![192.168.1.1/32,192.168.1.50/32,192.168.1.10/32,
192.168.1.20/32,192.168.1.30/32,192.168.40/32]

# Configurazione dell'access point da monitorare (viene indicato il MAC della
scheda di rete)
var ACCESS_POINTS 00:0F:3D:AF:BA:EB
var CHANNELS 1

# Path delle regole
var RULE_PATH rules

# Configurazione del decoder per la gestione degli alert

config disable_decode_alerts


# Step #2: Preprocessori per la gestione dei pacchetti

preprocessor flow: stats_interval 0 hash 2
preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy first detect_anomalies
preprocessor stream4: disable_evasion_alerts
preprocessor stream4_reassemble
preprocessor http_inspect: global \
    iis_unicode_map unicode.map 1252
preprocessor http_inspect_server: server default \
    profile all ports { 80 8080 8180 } oversize_dir_length 500
preprocessor rpc_decode: 111 32771
preprocessor bo
preprocessor telnet_decode
preprocessor sfportscan: proto  { all } \
                         memcap { 10000000 } \
                         sense_level { low }

# AntiStumbler
#---------------------------------------
# AntiStumbler detects possible Netstumbler activity
#
preprocessor antistumbler: probe_reqs 90, probe_period 30, expire_timeout 3600


# Step #3: Configurazione output
#Traffico in formato tcpdump e log nel database Mysql
```

```
output log_tcpdump: tcpdump.log
output database: log, mysql, dbname=snort user=snortusr host=localhost
include classification.config
include reference.config


#Step #5: Definizione delle regole

include $RULE_PATH/wifi.rules
```

Le nuove regole per la gestione del traffico wifi sono incluse all'interno del file ***wifi.rules***.

```
# Regole per il traffico wifi

alert wifi any -> any (msg:"Association Request"; stype:STYPE_ASSOCREQ;)
alert wifi any -> any (msg:"Association Response"; stype:STYPE_ASSOCRESP;)
alert wifi any -> any (msg:"Reassociation Request"; stype:STYPE_REASSOC_REQ;)
alert wifi any -> any (msg:"Reassociation Response";
stype:STYPE_REASSOC_RESP;)
log wifi any -> any (msg:"Probe Request"; stype:STYPE_PROBEREQ;)
log wifi any -> any (msg:"Probe Response"; stype:STYPE_PROBERESP;)
alert wifi any -> any (msg:"Beacon"; stype:STYPE_BEACON;)
alert wifi any -> any (msg:"Disassociation"; stype:STYPE_DISASSOC;)
alert wifi any -> any (msg:"Authentication"; stype:STYPE_AUTH;)
alert wifi any -> any (msg:"Deauthentication"; stype:STYPE_DEAUTH;)


#Regole aggiuntive per la distinzione dei flussi in ingresso e uscita


log tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"Traffico TCP in";
session:printable;)
log tcp $HOME_NET any -> $EXTERNAL_NET any (msg:"Traffico TCP out";
session:printable;)

log udp $EXTERNAL_NET any -> $HOME_NET any (msg:"Traffico UDP in";
session:printable;)
log udp $HOME_NET any -> $EXTERNAL_NET any (msg:"Traffico UDP out";
session:printable;)

log icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"Traffico ICMP in";
session:printable;)
log icmp $HOME_NET any -> $EXTERNAL_NET any (msg:"Traffico ICMP out";
session:printable;)
```

Snort è in grado di operare sia come sniffer che come IDS: nella prima modalità può catturare tutto il traffico in transito ma non può salvare le informazioni nel database, mentre nella seconda cattura esclusivamente il traffico corrispondente alle regole impostate. In questo progetto occorre catturare tutto il traffico memorizzandolo nel database e allo stesso tempo poter sfruttare le capacità di IDS. Per questo motivo sono state introdotte due nuove regole (***Traffico IN*** e ***Traffico OUT***) per ciascun protocollo gestibile (***tcp***, ***udp*** e ***icmp***) in modo da differenziare il flusso in ingresso proveniente dai

client (potenziali aggressori) e quello in uscita dai sistemi che costituiscono l'ambiente honeypot. In particolare sono esclusi dalla rete esterna gli indirizzi IP corrrispondenti agli host simulati. Infine la direttiva ***session:printable*** permette di salvare in formato leggibile l'intera sessione in modo da poterla analizzare successivamente.

```
#Regole aggiuntive per la distinzione dei flussi in ingresso e uscita

log tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"Traffico TCP in";
session:printable;)
log tcp $HOME_NET any -> $EXTERNAL_NET any (msg:"Traffico TCP out";
session:printable;)

log udp $EXTERNAL_NET any -> $HOME_NET any (msg:"Traffico UDP in";
session:printable;)
log udp $HOME_NET any -> $EXTERNAL_NET any (msg:"Traffico UDP out";
session:printable;)

log icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"Traffico ICMP in";
session:printable;)
log icmp $HOME_NET any -> $EXTERNAL_NET any (msg:"Traffico ICMP out";
session:printable;)
```

## 5.6 Test di valutazione dell'ambiente virtuale

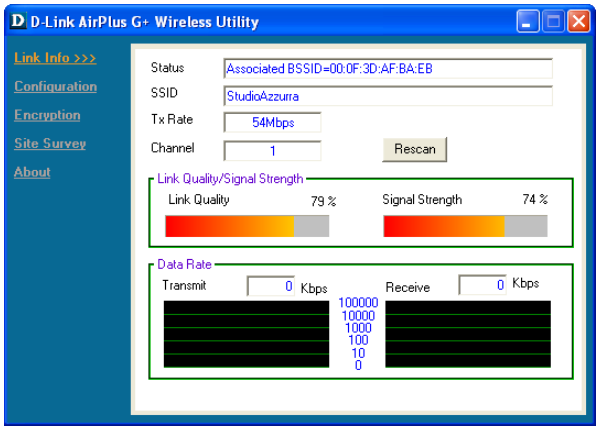
Sono stati condotti dei test in laboratorio per verificare il livello di realismo dell'ambiente creato con Honeyd facendo ricorso ad ***Nmap***, uno degli strumenti più utilizzati dagli attacker e facilmente reperibile in Internet. Le informazioni raccolte hanno permesso di fare un tuning dell'applicazione.

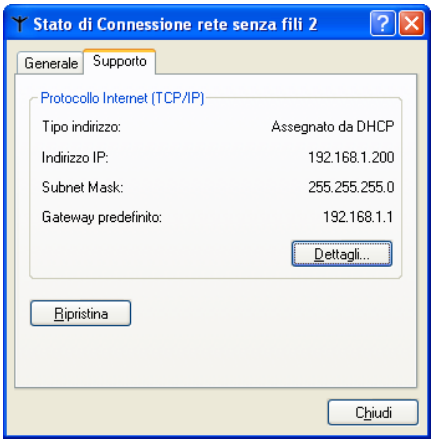
Dopo aver avviato lo script ***./start-honeypot.sh*** il sistema è pienamente operativo.

```
#!/bin/sh
# Script di avvio wireless honeypot

# Impostazioni access point

SSID="StudioAzzurra"
CHANNEL="1"
IP_ACCESS_POINT="192.168.1.50"
NETMASK="255.255.255.0"
GATEWAY="192.168.1.1"
INTERFACE="ath0"


# 1. Avvio database MySQL
echo
echo "Avvio Mysql in corso ..."
mysqld --user=mysql --socket=/var/lib/mysql/mysql.sock &
echo "Operazione effettuata."
sleep 5


# 2. Avvio access point
echo
wlanconfig $INTERFACE create wlandev wifi0 wlanmode ap
iwconfig $INTERFACE essid  $SSID channel $CHANNEL
ifconfig $INTERFACE $IP_ACCESS_POINT netmask $NETMASK
ifconfig $INTERFACE up

#Impostazione parametri DHCP
echo -n "Creazione file dhcpd.conf ..."
cp dhcp/dhcpd.conf.src  /etc/dhcpd.conf
echo "Operazione effettuata."

#Riavvio servizio DHCP
echo -n "Riavvio server DHCP ..."
/etc/init.d/dhcpd restart
echo "Operazione effettuata."

# 3. Avvio Honeyd
# Parametri:
# --disable-webserver : disabilitazione webserver interno
# --disable-update : disabilitazione aggiornamenti
# -f : file di configurazione
```

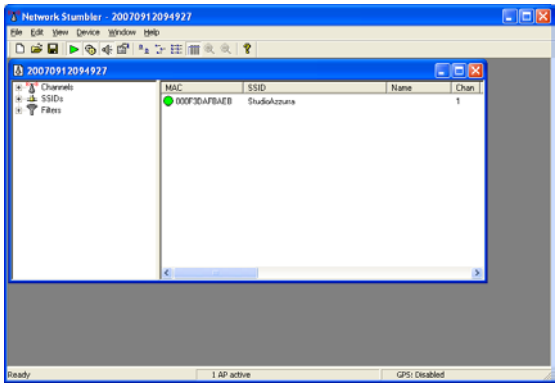
```
# -p : file fingerprint di Nmap
# -x : file fingerprint di Xprobe
# -a : file associazioni di Nmap
# -O : file passive fingerprinting
# -l : file di log
# -s : file di log dei servizi
# -i : interfaccia di rete

echo
echo "Avvio Honeyd in corso ..."
./honeyd_kit --disable-webserver --disable-update -f honeyd.conf -p
nmap.prints -x xprobe2.conf -a nmap.assoc -0 pf.os -l logs/log_honeyd
-s logs/service_honeyd  -i ath0 192.168.1.0/24 &
echo "Operazione effettuata."
sleep 3

# 4. Avvio Snort

wlanconfig ath1 create wlandev wifi0 wlanmode monitor
ifconfig ath1 up
sleep 3
echo
echo "Avvio Snort in corso ..."
snort -c /materiale/snort/snort.conf -l /materiale/snort/logs -i ath1
&
echo "Operazione effettuata."
sleep 3

# Avvio ulteriori servizi di supporto
echo "Avvio VSFTP in corso..."
vsftpd &
echo "Operazione effettuata."
```

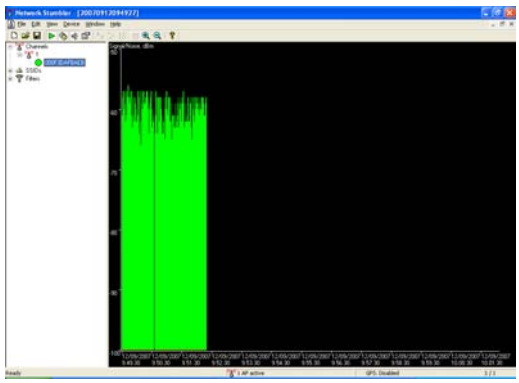
Il client rileva la presenza della rete a cui può accedere immediatamente non essendo attivata alcuna cifratura e grazie al servizio DHCP che permette automaticamente l'impostazione di tutti i parametri relativi al TCP.



Effettuiamo una verifica con Netstumbler per assicurarci che tutto funzioni correttamente (SSD, Canale e MAC), oltre ad osservare l'intensità del segnale per stabilire l'eventuale area di copertura.



Esaminando il file di log in formato tcpdump possiamo individuare le informazioni relative all'associazione del client e alla relativa assegnazione dell'indirizzo IP tramite DHCP.

```
No.   Time        Source          Destination      Protocol   Info
35    67.576104   192.168.1.50    192.168.1.200    DHCP       DHCP Offer
- Transaction ID 0x8e747c18
Frame 35 (504 bytes on wire, 504 bytes captured)
Prism Monitoring Header
IEEE 802.11
Logical-Link Control
Internet Protocol, Src: 192.168.1.50 (192.168.1.50), Dst:
192.168.1.200 (192.168.1.200)
User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)
Bootstrap Protocol

No.   Time        Source          Destination      Protocol   Info
36     67.843728   192.168.1.50     192.168.1.200     DHCP         DHCP ACK
- Transaction ID 0x8e747c18
```

Procediamo all'analisi della rete effettuando in primo luogo una scansione per individuare gli host presenti.

```
% nmap -v -sP 192.168.1.0/24
```

Nmap invia una richiesta PING verso l'host destinatario: se riceve una risposta il sistema risulta attivo e i pacchetti ICMP non vengono bloccati. In caso contrario tenta

un TCP Ping per stabilire se esiste un blocco sul protocollo ICMP oppure l'host è realmente offline.

```
Starting Nmap 4.11 ( http://www.insecure.org/nmap ) at 2007-09-12 19:50 ora
legale Europa occidentale
Initiating ARP Ping Scan against 200 hosts [1 port/host] at 19:50
The ARP Ping Scan took 3.00s to scan 200 total hosts.
DNS resolution of 6 IPs took 13.00s.
Host 192.168.1.0 appears to be down.
Host 192.168.1.1 appears to be up.
MAC Address: 00:A0:C5:0C:DC:80 (Zyxel Communication)
Host 192.168.1.10 appears to be up.
MAC Address: 00:05:5D:B2:71:8E (D-Link Systems)
Host 192.168.1.20 appears to be up.
MAC Address: 08:00:4E:76:FF:CF (3com Europe)
Host 192.168.1.30 appears to be up.
MAC Address: 00:40:B4:3F:E5:36 (Nextcom K.K.)
Host 192.168.1.40 appears to be up.
MAC Address: 00:50:99:F3:2C:3D (3com Europe)
Host 192.168.1.50 appears to be up.
MAC Address: 00:80:C8:AC:B9:81 (D-link Systems)
DNS resolution of 1 IPs took 13.00s.
Host 192.168.1.200 appears to be up.
Initiating ARP Ping Scan against 55 hosts [1 port/host] at 19:51
The ARP Ping Scan took 1.33s to scan 55 total hosts.
Nmap finished: 256 IP addresses (7 hosts up) scanned in 30.641 seconds
               Raw packets sent: 504 (21.168KB) | Rcvd: 13 (546B)
```

Ottenuto l'elenco degli host (per ragioni di spazio non sono riportati anche quelli offline), verifichiamo i servizi attivi su ciascuno di essi confrontandoli con la configurazione di Honeyd. Si può notare che vengono rilevati 6 host come previsto a cui si aggiunge il client che si è associato alla rete (IP 192.168.1.200).

```
% nmap –sS -sU -A -v 192.168.1.x
```

Spieghiamo brevemente il significato dei parametri specificati rimandando tutti gli approfondimenti al sito ufficiale di ***Nmap***[7].

***-sS SYN Stealth Scan***: viene inviato un pacchetto SYN di inizio connessione e in caso di ricezione di un pacchetto SYN/ACK si deduce che la porta contattata è aperta e il sistema remoto sta tentando di stabilire una connessione. A questo punto Nmap invia un pacchetto RST per troncare la connessione (in effetti non ancora stabilita) .

***-sU***: individua le porte UDP aperte attraverso l'invio di pacchetti 0-byte. Se viene ricevuto un messaggio ICMP di Port Unreachable (porta non raggiungibile) significa

[7] http://www.insecure.org/nmap

che quest'ultima è chiusa. Da osservare che questo tipo di scansione può richiedere tempi di esecuzione molto elevati.

***-A***: abilita l'OS fingerprinting (identificazione del sistema operativo in esecuzione)

***-v***: rileva la versione del servizio

I risultati ottenuti dai test suggeriscono alcune osservazioni e miglioramenti da apportare alla configurazione.

La scelta di abilitare il servizio DHCP sull'AP obbliga ad assegnare staticamente un indirizzo IP (nel nostro caso 192.168.1.50). Purtroppo in questo modo Honeyd non risponde correttamente alle richieste relative a tale indirizzo.

```
Warning:  OS detection will be MUCH less reliable because we did not find at
least 1 open and 1 closed TCP port
All 1680 scanned ports on 192.168.1.50 are filtered
MAC Address: 00:80:C8:8D:EA:EF (D-link Systems)
Too many fingerprints match this host to give specific OS details
TCP/IP fingerprint:
SInfo(V=4.11%P=i686-pc-windows-windows%D=6/9%Tm=466AE3CB%O=-1%C=-1%M=0080C8)
```

Come riportato da Nmap tutte le porte relative all'AP risultano chiuse compresa la tcp/80 lasciata intenzionalmente aperta per simulare il server web di amministrazione. Questo impedisce l'identificazione del sistema operativo e della tipologia di dispositivo. Al contrario rinunciando all'uso del DHCP le cose vanno a posto.

```
Interesting ports on 192.168.1.1:
Not shown: 1679 filtered ports
PORT   STATE SERVICE    VERSION
23/tcp open  tcpwrapped
MAC Address: 00:A0:C5:48:7A:46 (Zyxel Communication)
Device type: broadband router
Running: ZyXel ZyNOS
OS details: ZyXel Prestige 642R-11 ASDL router running ZyNOS
TCP Sequence Prediction: Class=trivial time dependency
                         Difficulty=8 (Trivial joke)
IPID Sequence Generation: Incremental
```

Il router viene riconosciuto correttamente, riportando come unica porta aperta la 23 a cui corrisponde il servizio di amministrazione remota via telnet.

Passiamo ad esaminare gli host della rete, osservando come vengano rilevate le porte secondo la configurazione prevista. Qualche problema si presenta nell'identificazione corretta della versione del servizio. Di seguito si riportano i risultati generati direttamente da Nmap.

**SERVER : 192.168.1.10**

```
Starting Nmap 4.11 ( http://www.insecure.org/nmap ) at 2007-09-12 19:52 ora
legale Europa occidentale
Initiating ARP Ping Scan against 192.168.1.10 [1 port] at 19:52
The ARP Ping Scan took 0.11s to scan 1 total hosts.
DNS resolution of 1 IPs took 13.00s.
Initiating SYN Stealth Scan against 192.168.1.10 [1680 ports] at 19:52
Discovered open port 21/tcp on 192.168.1.10
Discovered open port 25/tcp on 192.168.1.10
Discovered open port 80/tcp on 192.168.1.10
Discovered open port 23/tcp on 192.168.1.10
The SYN Stealth Scan took 1.91s to scan 1680 total ports.
Initiating UDP Scan against 192.168.1.10 [1487 ports] at 19:52
The UDP Scan took 2.88s to scan 1487 total ports.
Initiating service scan against 5 services on 192.168.1.10 at 19:52
The service scan took 50.00s to scan 5 services on 1 host.
For OSScan assuming port 21 is open, 1 is closed, and neither are firewalled
Host 192.168.1.10 appears to be up ... good.
Interesting ports on 192.168.1.10:
Not shown: 3162 closed ports
PORT   STATE         SERVICE    VERSION
21/tcp open          tcpwrapped
23/tcp open          tcpwrapped
25/tcp open          tcpwrapped
80/tcp open          http       Apache httpd 2.2.0 ((Linux/SUSE))
53/udp open|filtered domain
MAC Address: 00:05:5D:B2:71:8E (D-Link Systems)
Device type: general purpose
Running: Linux 2.4.X
OS details: Linux 2.4.18
TCP Sequence Prediction: Class=random positive increments
                         Difficulty=7144679 (Good luck!)
IPID Sequence Generation: All zeros

Nmap finished: 1 IP address (1 host up) scanned in 70.141 seconds
               Raw packets sent: 3227 (118.318KB) | Rcvd: 3180 (151.534KB)
```

**CLIENT 1 : 192.168.1.20**

```
Starting Nmap 4.11 ( http://www.insecure.org/nmap ) at 2007-09-12 19:58 ora
legale Europa occidentale
Initiating ARP Ping Scan against 192.168.1.20 [1 port] at 19:58
The ARP Ping Scan took 0.11s to scan 1 total hosts.
DNS resolution of 1 IPs took 13.00s.
Initiating SYN Stealth Scan against 192.168.1.20 [1680 ports] at 19:59
Discovered open port 21/tcp on 192.168.1.20
The SYN Stealth Scan took 1.84s to scan 1680 total ports.
Initiating UDP Scan against 192.168.1.20 [1487 ports] at 19:59
The UDP Scan took 1.84s to scan 1487 total ports.
Initiating service scan against 3 services on 192.168.1.20 at 19:59
Discovered open port 137/udp on 192.168.1.20
Discovered open|filtered port 137/udp on 192.168.1.20 is actually open
Discovered open port 138/udp on 192.168.1.20
Discovered open|filtered port 138/udp on 192.168.1.20 is actually open
The service scan took 50.00s to scan 3 services on 1 host.
For OSScan assuming port 21 is open, 1 is closed, and neither are firewalled
```

```
Host 192.168.1.20 appears to be up ... good.
Interesting ports on 192.168.1.20:
Not shown: 3164 closed ports
PORT    STATE SERVICE      VERSION
21/tcp  open  ftp          Microsoft ftpd 5.0
137/udp open  netbios-ns   Microsoft Windows netbios-ssn (workgroup: Mshome)
138/udp open  netbios-dgm?
1 service unrecognized despite returning data. If you know the
service/version, please submit the following fingerprint at
http://www.insecure.org/cgi-bin/servicefp-submit.cgi
MAC Address: 08:00:4E:76:FF:CF (3com Europe)
Device type: general purpose
Running: Microsoft Windows NT/2K/XP
OS details: Microsoft Windows XP Pro

TCP Sequence Prediction: Class=random positive increments
                         Difficulty=96292 (Worthy challenge)
IPID Sequence Generation: Incremental
Service Info: Host: Client11; OS: Windows

Nmap finished: 1 IP address (1 host up) scanned in 67.922 seconds
               Raw packets sent: 3194 (117.202KB) | Rcvd: 3180 (151.050KB)
```

**CLIENT 2 : 192.168.1.30**

```
Starting Nmap 4.11 ( http://www.insecure.org/nmap ) at 2007-09-12 20:00 ora
legale Europa occidentale
Initiating ARP Ping Scan against 192.168.1.30 [1 port] at 20:00
The ARP Ping Scan took 1.13s to scan 1 total hosts.
DNS resolution of 1 IPs took 13.02s.
Initiating SYN Stealth Scan against 192.168.1.30 [1680 ports] at 20:00
Discovered open port 21/tcp on 192.168.1.30
Discovered open port 4444/tcp on 192.168.1.30
The SYN Stealth Scan took 1.94s to scan 1680 total ports.
Initiating UDP Scan against 192.168.1.30 [1487 ports] at 20:00
The UDP Scan took 3.41s to scan 1487 total ports.
Initiating service scan against 4 services on 192.168.1.30 at 20:00
Discovered open port 137/udp on 192.168.1.30
Discovered open|filtered port 137/udp on 192.168.1.30 is actually open
Discovered open port 138/udp on 192.168.1.30
Discovered open|filtered port 138/udp on 192.168.1.30 is actually open
The service scan took 50.00s to scan 4 services on 1 host.
For OSScan assuming port 21 is open, 1 is closed, and neither are firewalled
Host 192.168.1.30 appears to be up ... good.
Interesting ports on 192.168.1.30:
Not shown: 3163 closed ports
PORT     STATE SERVICE      VERSION
21/tcp   open  ftp          Microsoft ftpd 5.0
4444/tcp open  tcpwrapped
137/udp  open  netbios-ns   Microsoft Windows netbios-ssn (workgroup: Mshome)
138/udp  open  netbios-dgm?
1 service unrecognized despite returning data. If you know the
service/version, please submit the following fingerprint at
http://www.insecure.org/cgi-bin/servicefp-submit.cgi
MAC Address: 00:40:B4:3F:E5:36 (Nextcom K.K.)
Device type: general purpose
Running: Microsoft Windows NT/2K/XP
OS details: Microsoft Windows XP Home Edition

TCP Sequence Prediction: Class=random positive increments
```

```
                          Difficulty=73662 (Worthy challenge)
IPID Sequence Generation: Incremental
Service Info: Host: Client21; OS: Windows

Nmap finished: 1 IP address (1 host up) scanned in 70.875 seconds
               Raw packets sent: 3235 (118.526KB) | Rcvd: 3180 (151.058KB)
```

```
CLIENT 3 : 192.168.1.40
═══════════════════════════════════════════════════════════════════════

Starting Nmap 4.11 ( http://www.insecure.org/nmap ) at 2007-09-12 20:01 ora
legale Europa occidentale
Initiating ARP Ping Scan against 192.168.1.40 [1 port] at 20:01
The ARP Ping Scan took 0.09s to scan 1 total hosts.
DNS resolution of 1 IPs took 13.00s.
Initiating SYN Stealth Scan against 192.168.1.40 [1680 ports] at 20:02
Discovered open port 139/tcp on 192.168.1.40
The SYN Stealth Scan took 1.97s to scan 1680 total ports.
Initiating UDP Scan against 192.168.1.40 [1487 ports] at 20:02
The UDP Scan took 1.75s to scan 1487 total ports.
Initiating service scan against 3 services on 192.168.1.40 at 20:02
Discovered open port 137/udp on 192.168.1.40
Discovered open|filtered port 137/udp on 192.168.1.40 is actually open
Discovered open port 138/udp on 192.168.1.40
Discovered open|filtered port 138/udp on 192.168.1.40 is actually open
The service scan took 113.50s to scan 3 services on 1 host.
For OSScan assuming port 139 is open, 1 is closed, and neither are firewalled
Host 192.168.1.40 appears to be up ... good.
Interesting ports on 192.168.1.40:
Not shown: 3164 closed ports
PORT    STATE SERVICE      VERSION
139/tcp open  netbios-ssn?
137/udp open  netbios-ns   Microsoft Windows netbios-ssn (workgroup: Mshome)
138/udp open  netbios-dgm?
1 service unrecognized despite returning data. If you know the
service/version, please submit the following fingerprint at
http://www.insecure.org/cgi-bin/servicefp-submit.cgi
MAC Address: 00:50:99:F3:2C:3D (3com Europe)
Device type: general purpose
Running: Microsoft Windows 95/98/ME
OS details: Microsoft Windows 98 SP1

TCP Sequence Prediction: Class=trivial time dependency
                         Difficulty=14 (Easy)
IPID Sequence Generation: Incremental
Service Info: Host: Client31; OS: Windows

Nmap finished: 1 IP address (1 host up) scanned in 132.171 seconds
               Raw packets sent: 3208 (117.818KB) | Rcvd: 3181 (151.054KB)
```

Dopo aver individuato la presenza di un server http attivo all'indirizzo 192.168.1.10 proviamo ad aprirlo nel browser.


Viene richiesta la compilazione di un form per l'autenticazione. Proviamo ad inserire la coppia ***prova/prova*** ricevendo un messaggio di autenticazione fallita. Esaminando i log di Snort rileviamo la presenza di un alert in corrispondenza del tentativo di login.

```
[**] [1:0:0] Tentativo login [**]
[Priority: 0]
09/20-09:31:13.644539 192.168.1.200:1041 -> 192.168.1.10:80
TCP TTL:128 TOS:0x0 ID:191 IpLen:20 DgmLen:595 DF
***AP*** Seq: 0xDD1D7A70  Ack: 0x70D25689  Win: 0x4470  TcpLen: 32
TCP Options (3) => NOP NOP TS: 980 417200
```

Procediamo a testare il servizio FTP osservando come Snort consenta di recuperare tutte le informazioni associate alla sessione, compresi i dati di login e i comandi impartiti.

```
No.  Time        Source          Destination   Protocol Info
587  199.719580  192.168.1.200   192.168.1.10  FTP      Request: opts
utf8 on
590  200.222110  192.168.1.200   192.168.1.10  FTP      Request: syst
600  201.932101  192.168.1.200   192.168.1.10  FTP      Request: site
help
602  202.435021  192.168.1.200   192.168.1.10  FTP      Request: PWD
603  203.008182  192.168.1.200   192.168.1.10  FTP      Request: noop
604  203.556856  192.168.1.200   192.168.1.10  FTP      Request: CWD /
605  204.046634  192.168.1.200   192.168.1.10  FTP      Request: TYPE A
606  204.547493  192.168.1.200   192.168.1.10  FTP      Request: PORT
192,168,1,200,4,19
643  238.218680  192.168.1.200   192.168.1.10  FTP      Request: USER
ftp
644 240.989889  192.168.1.200  192.168.1.10  FTP   Request: PASS prova
```

Molte altre informazioni sul client si possono rilevare grazie al protocollo ***NETBIOS***: ad esempio il nome dell'host, il dominio di appartenenza, eventuali altri servizi attivi.

```
No.  Time        Source          Destination    Protocol  Info
271  96.654779  192.168.1.200  192.168.1.255  NBNS      Registration
NB <01><02>__MSBROWSE__<02><01>
617  219.655202 192.168.1.200  192.168.1.255  BROWSER   Domain /
Workgroup Announcement MSHOME, NT Workstation, Domain Enum
1359 676.584620 192.168.1.200   192.168.1.255   BROWSER    Local Master
Announcement MAURO, Workstation, SQL Server, Print Queue Server, NT
Workstation, Potential Browser, Master Browser
```

## 5.7 Gestione e analisi dei dati

Le fonti di dati sono numerose ed eterogenee:

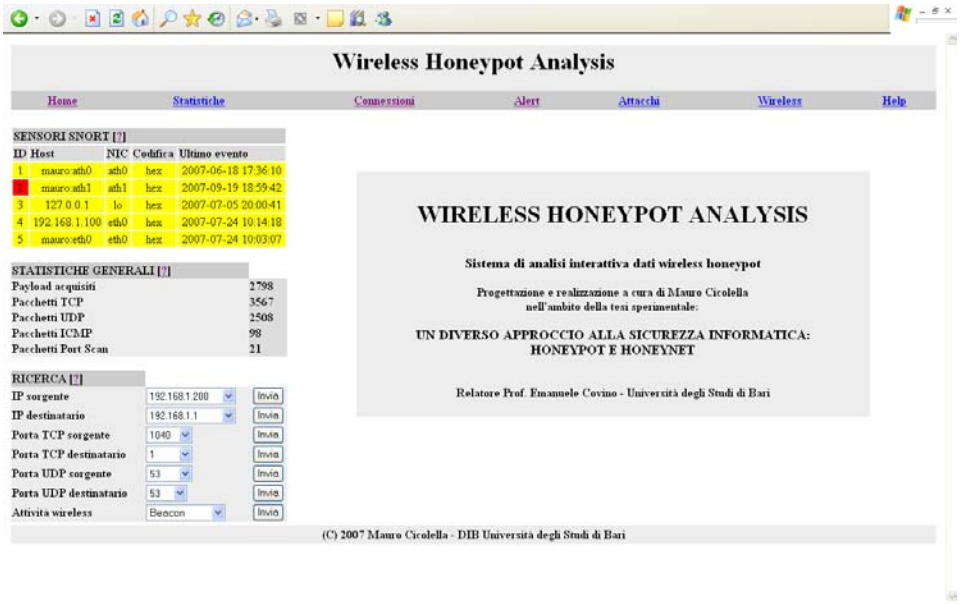
- log DHCP
- log Honeyd
- log degli script di emulazione dei servizi
- alert di Snort
- traffico catturato

Considerata tale varietà e consistenza diventa necessario predisporre una loro centralizzazione e correlazione. In questa ottica si è deciso l'utilizzo di un database in quanto:

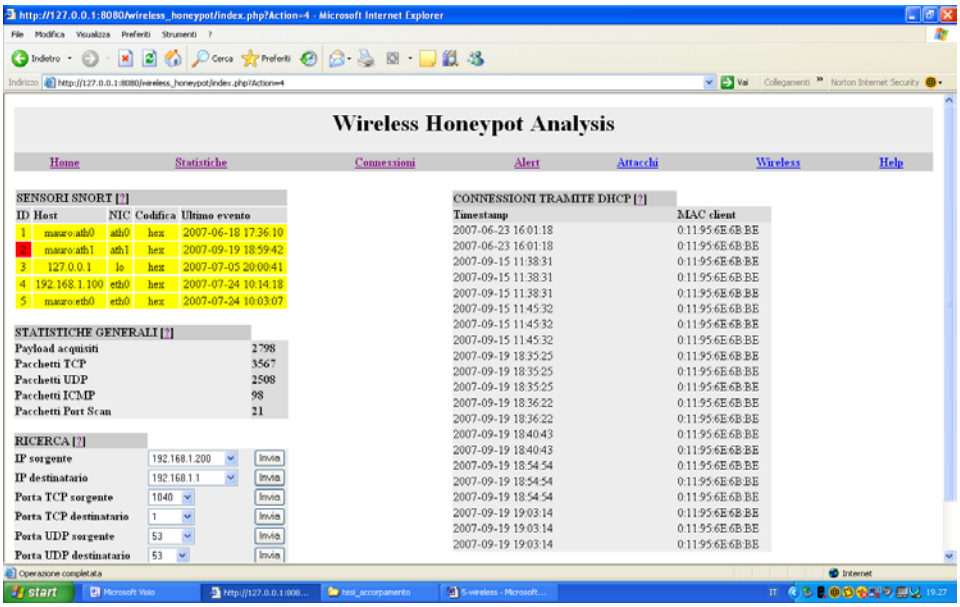
- permette di concentrare tutti i dati facilitando le operazioni di backup;
- risulta molto flessibile in fase di analisi consentendo attraverso query ad-hoc la correlazione di tutti i dati.

Per semplificare il processo di analisi dei dati è stata creata una semplice applicazione web in php per l'interfacciamento con il database di Snort, in modo da ricostruire tutti gli eventi accaduti sull'honeypot. Naturalmente non si sostituisce ai file di log (soprattutto quelli in formato tcpdump) che restano la fonte primaria dei dati, ma consente di guidare più agevolmente le successive operazioni. Uno dei vantaggi è rappresentato dalla possibilità di operare autonomamente rispetto all'honeypot.
Per illustrarne le diverse componenti vediamo una tipica sessione di analisi, procedendo per passi successivi. Dopo aver avviato l'applicazione dal browser ci troviamo di fronte alla schermata principale. Sulla sinistra è possibile scorgere una serie di box:

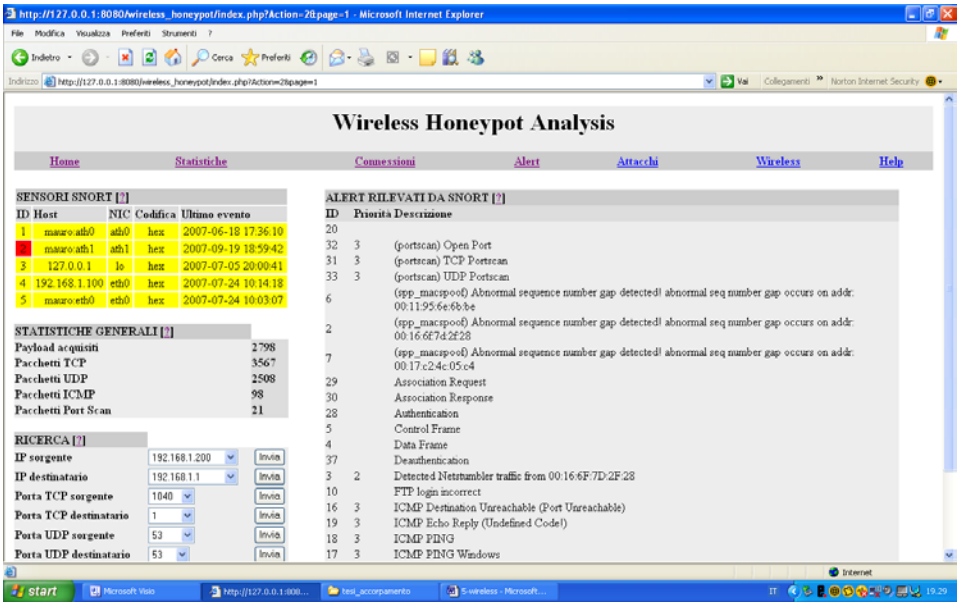
- il primo riporta sinteticamente i dati relativi ai vari sensori Snort da cui sono stati acquisiti i dati. In particolare viene indicato in rosso quello attualmente attivo e sul quale è consentito effettuare le analisi. Modificando il file di configurazione è possibile scegliere qualunque altro tra quelli disponibili;
- il secondo riporta le statistiche riguardanti i pacchetti catturati, dettagliandole in base allo specifico protocollo;
- l'ultimo consente invece di fare molteplici ricerche nel database impostando criteri come il protocollo, il numero di porta, l'indirizzo IP.



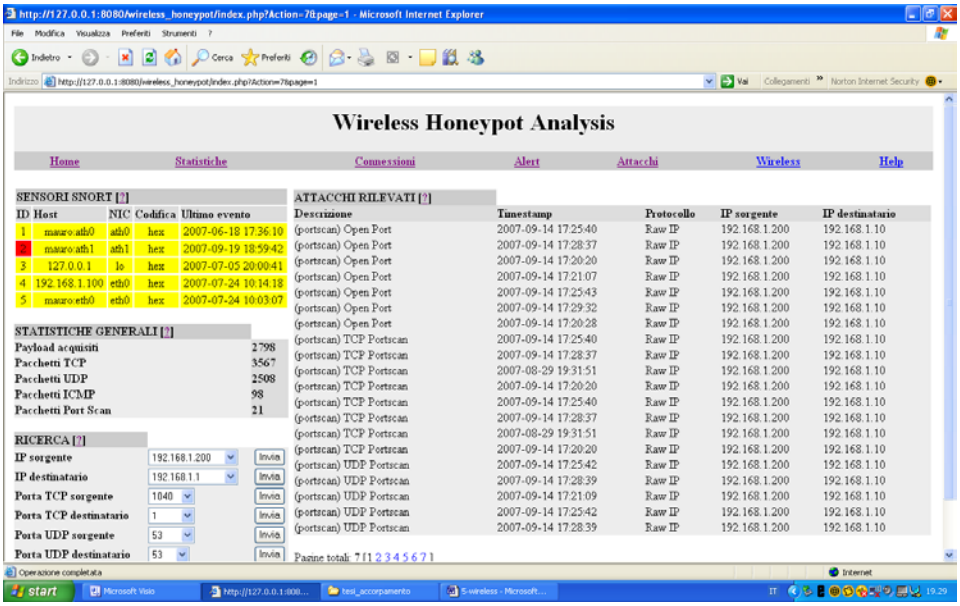

Attraverso la barra superiore è possibile accedere alle diverse sezioni disponibili. Partiamo esaminando le richieste di connessione ricevute dal DHCP. Per ciascuna di esse è indicato il timestamp e l'indirizzo MAC dei vari client, fondamentale per consentire una loro distinzione dal momento che un IP può essere assegnato a client diversi in momenti differenti.



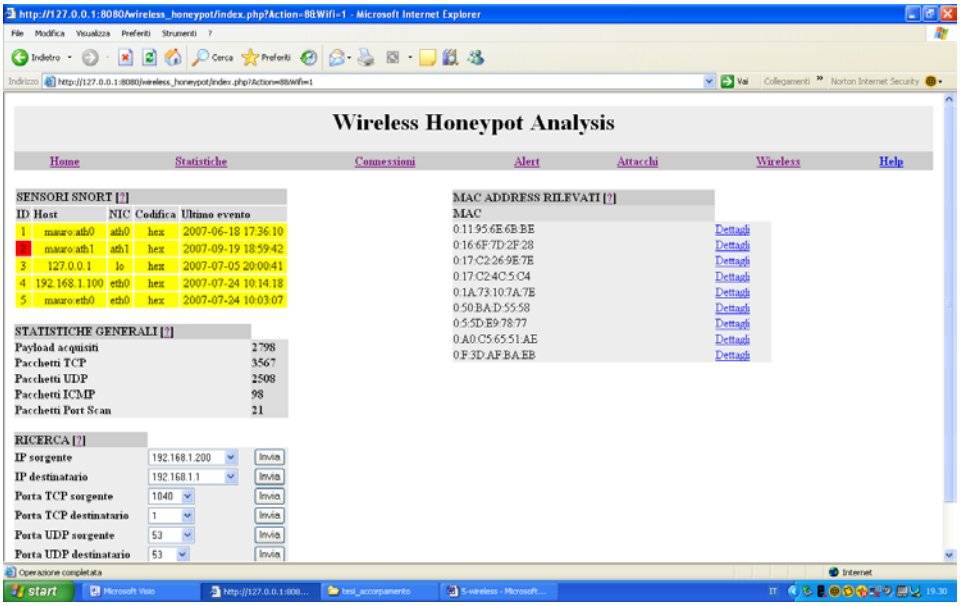

A questo punto si può dare un'occhiata agli "Alert" prodotti da Snort in modo da rendersi conto di quale tipologia di traffico potenzialmente "ostile" sia stata rilevata.



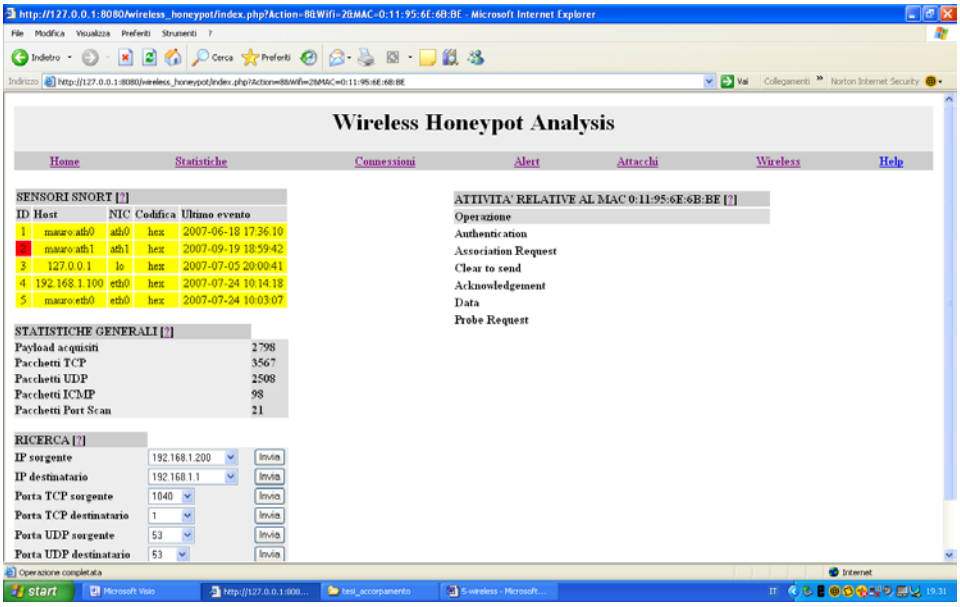

Più in dettaglio si possono esaminare gli attacchi individuati in base alle regole impostate. Per ciascuno di essi è possibile conoscere data e ora, protocollo interessato e indirizzi sorgente e destinatario. Queste informazioni saranno utili in seguito per ricerche più approfondite.



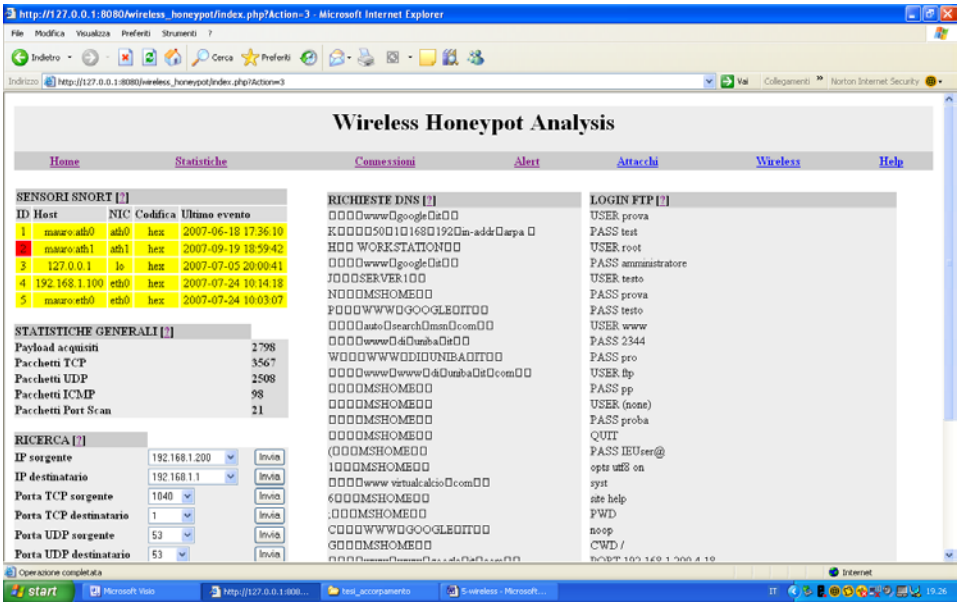

E’ inoltre possibile dare uno sguardo all’attività wireless rilevata nella zona in cui è posto l’honeypot.




In particolare cliccando sul singolo MAC vengono illustrate le tipologie di pacchetti catturati da tale sorgente. Ciò aiuta ad individuare le autenticazioni andate a buon fine o semplicemente le richieste di associazione oppure la presenza di eventuali access point.




Passando alla sezione statistiche è possibile avere un riepilogo delle richieste DNS registrate con relativi indirizzi web, nonché delle sessioni FTP potendo individuare rapidamente le credenziali di autenticazione utilizzate per l'accesso al sistema.




Con le informazioni ricavate in questo modo risulta alquanto semplice condurre delle ricerche più approfondite sui file di log tracciando in maniera completa le varie attività svolte sull'honeypot.

## 5.8 Sviluppi futuri

Il progetto descritto in questa tesi mostra un possibile impiego dei concetti alla base di honeypot e honeynet nel campo della sicurezza delle reti wireless.

Naturalmente molte delle scelte sono state condizionate dai tempi e dalle risorse disponibili nell'ottica di un lavoro puramente didattico, lasciando comunque ampi spazi a futuri sviluppi e miglioramenti.

In primo luogo si potrebbe lavorare ad una distribuzione Linux minimale, in sostituzione di quelle commerciali, permettendo l'utilizzo dell'intero sistema con risorse limitate e migliorando la sicurezza attraverso un'attenzione particolare per le componenti installate.

In secondo luogo si potrebbe adottare un honeypot ad alta interazione in grado di raccogliere un numero più consistente di informazioni anche se con maggiori investimenti in termini di tuning, oppure lavorare ad un ulteriore livello di personalizzazione degli script utilizzati per l'ambiente simulato.

Infine una ricerca su larga scala (aree metropolitane) per un periodo di tempo considerevolmente lungo consentirebbe di acquisire informazioni statisticamente rilevanti.

In conclusione possiamo affermare che honeypot e honeynet rappresentano un interessante campo di ricerca (ancora poco esplorato) nell'ambito della sicurezza informatica con molteplici applicazioni pratiche e possibili sviluppi di carattere commerciale.

# Bibliografia


| | | |
|---|---|---|
| **[1]** | ***K. Ealy*** | **A New Evolution in Hack Attacks**<br>Sans Institute, 2003 |
| **[2]** | ***M. Pickett*** | **A guide to the honeypot concept**<br>Sans Institute, 2003 |
| **[3]** | ***L. Spitzner*** | **Know your enemy: honeynets** |
| **[4]** | ***L. Spitzner*** | **Honeypots: Catching the Insider Threat** |
| **[5]** | ***R.Baumann, C.Plattner*** | **White Paper: Honeypots**<br>2002 |
| **[6]** | ***R.Talabis*** | **Honeypots 101: What's in it for me? (Advantages of Honeypots)**<br>The Philippine Honeynet Project, 2005 |
| **[7]** | ***R.Talabis*** | **Honeypots 101: Uses of Honeypots in IT Security**<br>The Philippine Honeynet Project, 2005 |
| **[8]** | ***R.Talabis*** | **Honeynets: A Honeynet Definition**<br>The Philippine Honeynet Project, 2005 |
| **[9]** | ***J. P. Anderson*** | **Computer Security Threat Monitoring and Surveillance**<br>Technical Report Fort Washington P.A., 1980 |
| **[10]** | ***R.Talabis*** | **The Gen I Honeypot Architecture**<br>The Philippine Honeynet Project, 2005 |
| **[11]** | ***R.Talabis*** | **The GenII & Gen III Honeynet Architecture**<br>The Philippine Honeynet Project, 2005 |
| **[12]** | ***N. Provos*** | **A Virtual Honeypot Framework**<br>CITI University of Michigan, 2003 |
| **[13]** | ***R. Chandran, S. Pakala*** | **Simulating Networks with Honeyd**<br>Paladion, 2003 |
| **[14]** | ***R. Baumann*** | **Honeyd – A low involvement Honeypot in Action**<br>GCIA |
| **[15]** | ***X. Fu, W. Yu, D. Cheng, X. Tan, S. Graham*** | **On recognizing virtual honeypots and countermeasures** |
| **[16]** | ***M. Wolfgang*** | **Host Discovery with Nmap**<br>2002 |

| | | |
|---|---|---|
| **[17]** | ***A. J. Bennieston*** | **NMAP – A Stealth Port Scanner**<br>http://www.nmap-tutorial.con |
| **[18]** | ***M. Dornseif, T. Holz, C.N. Klein*** | **NoSEBrEaK – Attacking Honeynets**<br>Workshop on Information Assurance and Security, United States Military Academy West Point, 2004 |
| **[19]** | ***D. Jeff*** | **User-mode Linux**<br>West Virginia University, 2002 |
| **[20]** | ***AA.VV.*** | **The Honeypot procfs**<br>http://user-mode-linux.sourceforge.net/hppfs.html |
| **[21]** | ***AA.VV.*** | **Skas Mode**<br>http://user-mode-linux.sourceforge.net/skas.html |
| **[22]** | ***C.Carella, J.Dike, N.Fox, M.Ryan*** | **UML Extensions for Honeypots in the ISTS Distributed Honeypot Project**<br>Institute for Security Tecnology Studies, Dart Mouth College Hanover, 2004 |
| **[23]** | ***P. Pozzi*** | **Introduzione all'informatica forense**<br>Franco Angeli, 2004 |
| **[24]** | ***Avv. P. Perri*** | **La computer and network forensics e leindagini informatiche - Strategie di indagine e best practice** |
| **[25]** | ***D. Farmer, W. Venema*** | **Forensic Computer Analysis: An Introduction – Reconstructing past events**<br>Dr Dobb's Journal, Settembre 2000 |
| **[26]** | ***M. Pierce*** | **Detailed Forensic Procedure for Laptop Computers**<br>Sans Institute, 2003 |
| **[27]** | ***T.Nguyen*** | **Wireless Networking Techniques Security Issues**<br>High Performance Technology Jsc, InWent, Akademic der Polizei DW, 2003 |
| **[28]** | ***R. Neumerkel, S. GroB*** | **A Sophisticated Solution for Revealing Attacks on Wireless Lan**<br>Institute for System Architecture, Techische Universitata Dresden, 2006 |
| **[29]** | ***Stefano Bendandi*** | **Honeypot: inganno o realtà?**<br>Hakin9, Luglio 2007 |