

UNIVERSITÀ POLITECNICA DELLE MARCHE

FACOLTÀ DI INGEGNERIA



*Corso di Laurea Magistrale in
Ingegneria Informatica e dell'Automazione*

Progettazione e implementazione di un'infrastruttura sicura

Studenti:

DANIELE PALLINI 1107326

MATTEO ABBRUZZETTI 1108842

Docenti:

LUCA SPALAZZI

ANNO ACCADEMICO 2022-2023

Indice

| | | |
|----------|--|-----------|
| 1 | Introduzione | 3 |
| 1.1 | Specifiche ed obiettivi | 3 |
| 1.2 | Strumenti utilizzati | 3 |
| 1.2.1 | Iptables | 4 |
| 1.2.2 | Squid | 4 |
| 1.2.3 | Snort | 4 |
| 1.2.4 | Tripwire | 5 |
| 2 | Progettazione | 6 |
| 2.1 | Architettura | 6 |
| 2.2 | Definizione delle policies di sicurezza | 7 |
| 3 | Implementazione | 9 |
| 3.1 | Implementazione dell'architettura | 9 |
| 3.1.1 | Prima macchina virtuale: Dual-homed Host | 9 |
| 3.1.2 | Seconda macchina virtuale: rete interna | 13 |
| 3.2 | Applicazione delle policies di sicurezza | 15 |
| 4 | Installazione e Test | 19 |
| 4.1 | Installazione | 19 |
| 4.2 | Test | 19 |
| 4.2.1 | Dual-homed Host | 20 |
| 4.2.2 | Macchina "Client" | 28 |
| 5 | Conclusioni | 37 |
| | Elenco delle figure | 39 |

Capitolo 1

Introduzione

1.1 Specifiche ed obiettivi

L'obiettivo di questo elaborato è la progettazione e l'implementazione di un'infrastruttura sicura attraverso l'utilizzo di macchine virtuali. L'implementazione sarà realizzata tramite la configurazione di un firewall e di sistemi di rilevamento delle intrusioni. Sarà necessaria anche l'installazione e la configurazione di un software in grado di filtrare correttamente i pacchetti in entrata e in uscita secondo le regole e le policies definite. Inoltre, sarà configurato un software finalizzato al monitoraggio e al registro delle modifiche effettuate ai file critici presenti nel sistema. Dovrà essere possibile accedere dall'esterno alle funzionalità fornite dal sistema, attraverso un meccanismo di autenticazione. L'infrastruttura dovrà essere robusta, ossia in grado di prevenire e resistere a diversi tipi di attacchi comuni.

1.2 Strumenti utilizzati

Per soddisfare tutti i requisiti presentati nella Sezione precedente, l'infrastruttura seguirà lo schema rappresentato in Figura 1.1:

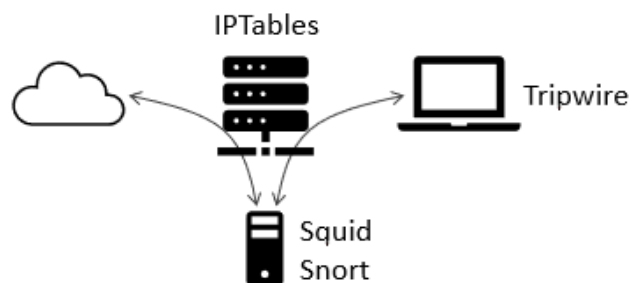


Figura 1.1: Schema dell'infrastruttura sicura

Di seguito saranno presentati singolarmente gli strumenti specificati in essa.

1.2.1 Iptables

Iptables è un firewall per sistemi operativi basati su Linux. È uno strumento essenziale per la gestione delle reti di Linux ed è utilizzato per controllare il traffico di rete, consentendo o bloccando la comunicazione tra il sistema e la rete o tra diverse parti della rete stessa. Iptables può essere integrato con strumenti aggiuntivi, come Ufw e Gufw, per renderlo più intuitivo e semplice da utilizzare.

Ufw

Ufw, acronimo di Uncomplicated Firewall, è un'applicazione di interfaccia a riga di comando per la gestione semplificata delle regole del firewall su sistemi basati su Linux. Ufw è progettato per semplificare la configurazione di Iptables rendendo più facile per gli utenti meno esperti creare e gestire le regole del firewall.

Gufw

Gufw, acronimo di "Graphical Uncomplicated Firewall," è un'interfaccia grafica per Ufw. Gufw offre un'interfaccia grafica semplice e intuitiva che consente agli utenti di configurare le regole del firewall utilizzando un ambiente visuale anziché la riga di comando.

1.2.2 Squid

Squid è un server proxy open source ampiamente utilizzato che funge da intermediario tra le richieste dei client e i server di destinazione su Internet. Questo proxy può migliorare le prestazioni, la sicurezza e il controllo del traffico di rete consentendo il caching delle risorse web, il filtraggio dei contenuti, l'autenticazione degli utenti e altre funzionalità avanzate. In pratica, Squid è uno strumento potente per ottimizzare l'accesso a Internet all'interno di reti aziendali, educative e di altro tipo, consentendo di gestire il traffico in modo efficiente e sicuro.

1.2.3 Snort

Snort è un sistema di rilevamento delle intrusioni (Intrusion Detection System) e di prevenzione delle intrusioni (Intrusion Prevention System) open source. È ampiamente utilizzato per monitorare il traffico di rete alla ricerca di comportamenti sospetti o anomalie che potrebbero indicare tentativi di violazione della sicurezza della rete. Snort può, inoltre, generare registri dettagliati dei pacchetti e delle attività rilevate, consentendo agli amministratori di sistema di esaminare e analizzare gli eventi di sicurezza per comprendere meglio le minacce e prendere le misure necessarie. La comunità di utenti di Snort contribuisce con regole

personalizzate e pacchetti di aggiornamento delle regole, mantenendo il sistema aggiornato per affrontare nuove minacce.

1.2.4 Tripwire

Tripwire è un software di sicurezza informatica ampiamente utilizzato per il monitoraggio dell'integrità dei file su sistemi informatici. La sua principale funzione è quella di rilevare e segnalare le modifiche non autorizzate ai file di sistema e ai dati critici. È spesso utilizzato per rilevare intrusioni, attività malevole o cambiamenti indesiderati all'interno di un sistema. Il funzionamento di Tripwire si può sintetizzare in un'iniziale creazione della baseline o snapshot dei file critici, successivamente in un monitoraggio continuo di questi file che vengono confrontati con la baseline e, infine, in generazione di report contenenti informazioni dettagliate sulle modifiche apportate ai file. Questi report consentono agli amministratori di valutare se la modifica sia legittima o rappresenti una minaccia alla sicurezza. Se la modifica è stata autorizzata, è possibile aggiornare la baseline e i relativi stati dei file.

Capitolo 2

Progettazione

La prima fase è la progettazione. Al fine di realizzare l'infrastruttura sicura desiderata, a seguito dello studio delle specifiche e degli obiettivi, è stato necessario progettare l'architettura alla base del sistema e definire alcune policies di sicurezza.

2.1 Architettura

L'architettura firewall che si è ritenuta essere più adatta, in relazione alle richieste da soddisfare, è la *Dual-homed Gateway* (Figura 2.1). Questa architettura è caratterizzata da un gateway con due interfacce di rete che si comporta come un proxy server.

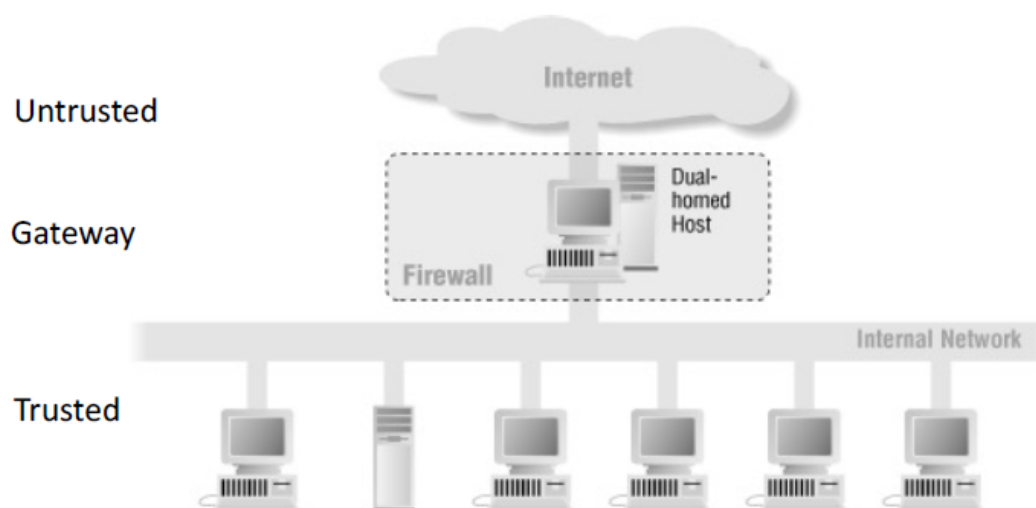


Figura 2.1: Struttura di un'architettura di tipo Dual-homed Gateway

In questa particolare architettura, una macchina, chiamata *Dual-homed Host*, fungerà da mezzo di comunicazione tra la rete interna (trusted) e la rete esterna, o Internet (untrusted). Sia i sistemi dentro il firewall che quelli fuori (su Internet) possono comunicare con il Dual-homed Host, ma essi non possono comunicare direttamente tra loro. Il traffico di pacchetti tra di loro è completamente bloccato. L'adozione di un'architettura di questo tipo comporta dei vantaggi e degli svantaggi.

I vantaggi della Dual-homed Gateway sono:

- La struttura di rete interna è mascherata e gli host interni possono essere resi invisibili alla rete esterna, poiché tutta la comunicazione passa attraverso il gateway.
- Fail-safe: se qualcosa fallisce, l'accesso viene negato e il sistema resta sicuro.
- Rispetto ad un'architettura Screening Router, Dual-homed Gateway ha un sistema che riesce a mantenere più facilmente i logs di sistema.

Gli svantaggi della Dual-homed Gateway sono invece:

- Un gateway non configurato correttamente, o soggetto a violazioni, potrebbe diventare un Single Point Of Failure (SPOF) per la sicurezza dell'intera rete.
- I proxies potrebbero risultare non disponibili per alcuni servizi.
- La centralità della macchina Dual-homed Host all'interno dell'infrastruttura può costituire un collo di bottiglia e causare rallentamenti nel sistema.

2.2 Definizione delle policies di sicurezza

La definizione e l'applicazione di policies di sicurezza appropriate sono fondamentali per garantire l'integrità e la protezione dell'intero sistema. Le policies definite sono le seguenti:

- **Policy di default "deny all".** Implementare una politica di default che scarta tutto il traffico che non corrisponde a regole specifiche costituisce una buona pratica di sicurezza. Ciò significa che, a meno che un pacchetto non corrisponda a una regola specifica, verrà scartato dal gateway. Questo aiuta a proteggere la rete da pacchetti indesiderati o dannosi.
- **Gestione delle connessioni.** Si configurerà una regola per la gestione delle connessioni impostando un limite massimo sul numero di connessioni simultanee per prevenire attacchi di tipo DoS (Denial of Service) o DDoS.

- **Logging e monitoraggio.** Verrà configurato il logging delle attività di sicurezza in modo da registrare gli eventi importanti, come i tentativi di accesso non autorizzati o i rilevamenti di intrusioni. In questo modo, si potranno monitorare regolarmente i registri di sicurezza per individuare eventuali anomalie o attività sospette.
- **Filtraggio del traffico in base al livello di applicazione.** Si implementeranno regole di filtraggio basate sul livello di applicazione per consentire o bloccare specifici protocolli o servizi. Sarà consentito solo il traffico SSH per l'accesso remoto sicuro e si bloccheranno determinati protocolli P2P che possono consumare una larghezza di banda significativa.
- **Politiche di autenticazione e accesso.** Verranno implementate politiche di sicurezza per l'autenticazione degli utenti e il controllo degli accessi, utilizzando regole per consentire solo l'accesso da indirizzi IP e reti specifici o tramite l'utilizzo di chiavi crittografiche.
- **Protezione da attacchi di scansione di porte.** Si configureranno regole per rilevare e bloccare attacchi di scansione di porte impostando limiti sul numero di richieste di connessione in un determinato intervallo di tempo o bloccando gli indirizzi IP sospetti che eseguono attacchi di scansione. L'implementazione di questa politica permetterà al sistema di prevenire e mitigare attacchi di tipo brute force.

Capitolo 3

Implementazione

In questo capitolo verranno illustrate in dettaglio le scelte fatte in fase di implementazione dell'infrastruttura sicura. Verranno inoltre presentate le modalità con cui sono state applicate le policies di sicurezza stabilite nel Capitolo 2. In generale, si è cercato di rendere l'intero sistema il più sicuro possibile, mantenendo allo stesso tempo un buon livello di usabilità.

3.1 Implementazione dell'architettura

Come definito in fase di progettazione, l'infrastruttura implementata corrisponde a una di tipo Dual-homed Gateway. Essa è costituita da due attori principali, corrispondenti ad altrettante macchine virtuali. Entrambe le virtual machines (VMs) hanno installato come sistema operativo *Ubuntu* alla versione 22.04 LTS.

3.1.1 Prima macchina virtuale: Dual-homed Host

Si è scelto di racchiudere in un'unica macchina virtuale le funzionalità dello Screening Router e del Bastion Host. Si è ottenuto in questo modo un Dual-homed Host. Questa macchina virtuale ha il compito di interfacciarsi e fare da tramite tra l'interno e l'esterno della rete. Le VMs all'interno non possono comunicare direttamente con l'esterno, devono sempre passare per la Dual-homed Host. Infatti, come si vedrà più in dettaglio in seguito, il traffico IP tra l'interno e l'esterno è completamente bloccato. Per questa duplice natura si è reso necessario avere due interfacce di rete: una dedicata al traffico nella rete interna e l'altra per la comunicazione con l'esterno. Per semplicità sono stati assegnati indirizzi IP statici ad entrambe le interfacce, in modo da poter essere raggiunti sempre allo stesso modo. Le due interfacce hanno le seguenti caratteristiche:

- Interfaccia interna: denominata **enp0s3**, è assegnata all'IP `192.168.0.190`. L'interfaccia interna gestisce il traffico di rete tra il dispositivo gateway e gli altri dispositivi presenti nella rete locale.

- Interfaccia esterna: denominata **enp0s8**, è assegnata all'IP **192.168.0.182**. L'interfaccia esterna è responsabile della comunicazione tra le macchine locali e le risorse esterne, come server remoti su Internet. Essa gestisce il traffico di rete che entra ed esce dal dispositivo ed è coinvolta nel processo di instradamento dei pacchetti di rete tra la rete locale e quella esterna, assicurando che questi siano inviati alla destinazione corretta.

La macchina Dual-homed Host è costituita da diverse componenti fondamentali, ognuna con un compito specifico. Di seguito verranno analizzate individualmente.

Iptables

Iptables è il software firewall di tutta l'infrastruttura. Racchiude tutte le policies di sicurezza e le rotte abilitate o vietate. Si è scelto di utilizzare Ufw (Uncomplicated Firewall), un framework che semplifica la scrittura e la gestione delle regole Iptables attraverso un'interfaccia a riga di comando. Ufw è l'applicazione predefinita di Ubuntu per la configurazione del firewall. Esso mette a disposizione anche un'interfaccia grafica, Gufw, per visualizzare e gestire ancora più facilmente le regole. In Figura 3.1 è rappresentato un esempio di una scheda di Gufw con alcune regole attive, le quali verranno presentate più nel dettaglio in seguito.

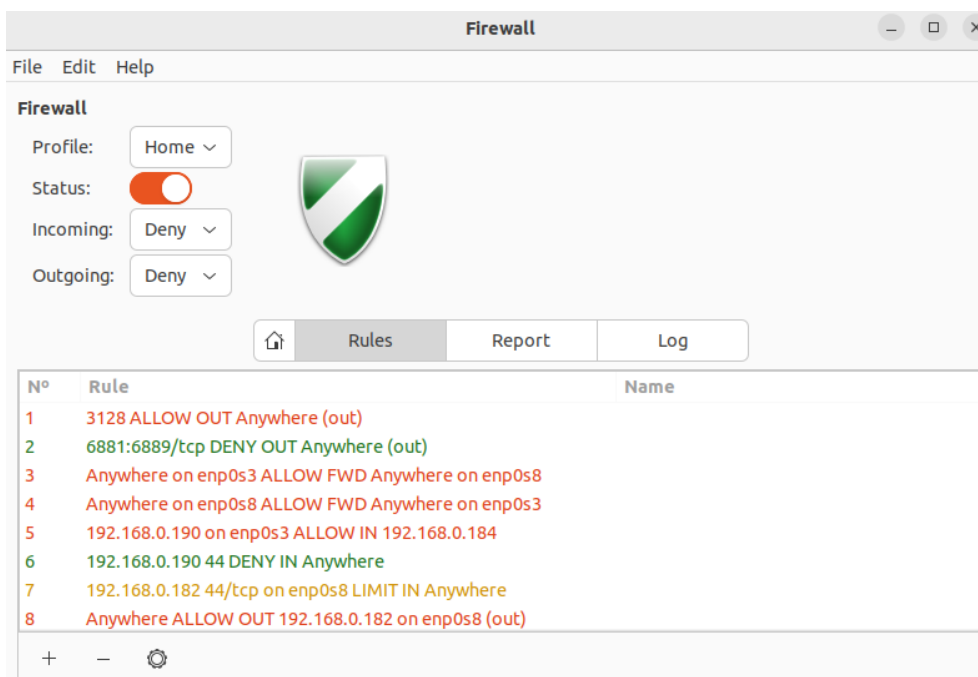


Figura 3.1: Esempio di una scheda di Gufw con alcune regole attive

Snort

Snort è un sistema di rilevamento delle intrusioni (IDS). Si tratta di uno strumento di sicurezza informatica che monitora e analizza il traffico di rete in tempo reale alla ricerca di attività sospette o potenzialmente dannose. Snort può identificare tentativi di attacchi informatici, intrusioni e attività anomale all'interno della rete. A questo proposito, la community di Snort ha stilato una serie di regole standard, note come "community rules", le quali possono essere scaricate e utilizzate come base per permettere a Snort di avere una copertura ampia delle potenziali minacce attualmente note. A queste regole possono esserne aggiunte altre personalizzate, le quali prendono il nome di "local rules". Sono state configurate due istanze di Snort, una per l'interfaccia interna e una per quella esterna. In questo modo si riesce ad avere una serie di vantaggi, tra cui:

1. **Migliore visibilità:** Avendo due istanze separate, è possibile ottenere una visione più chiara del traffico in entrata e in uscita da ciascuna interfaccia. Ciò può aiutare nell'identificazione di pattern di traffico anomali o potenzialmente dannosi.
2. **Isolamento del traffico:** Le interfacce interna ed esterna di una Dual-homed Host Gateway spesso gestiscono tipi di traffico diversi. Utilizzando istanze separate di Snort per ciascuna interfaccia, è possibile concentrarsi sui tipi di minacce specifici a ciascuna area della rete. Ad esempio, l'interfaccia esterna potrebbe essere più esposta a minacce provenienti dall'esterno, mentre l'interfaccia interna potrebbe affrontare minacce interne.
3. **Configurazioni differenziate:** Le interfacce interna ed esterna potrebbero richiedere diverse configurazioni di rilevamento e filtraggio. Utilizzando istanze separate di Snort, è possibile personalizzare le regole e le politiche di rilevamento per adattarle alle esigenze specifiche di ciascuna interfaccia.

Entrambe le istanze di Snort sono configurate in modo da iniziare a controllare sin dall'avvio della macchina. Sono stati predisposti due file di log, rispettivamente al percorso `/var/log/snort/InternalNetwork/alert_fast.txt` per l'interfaccia interna e `/var/log/snort/ExternalNetwork/alert_fast.txt` per quella esterna. Ogni riscontro di Snort viene riportato nel file specifico. Si è scelto di utilizzare Snort in *fast mode*. In questa modalità gli alert di Snort riportano il timestamp, l'indirizzo IP di partenza e la porta, l'indirizzo IP di destinazione e la porta. A questi viene aggiunto il nome e la classificazione della violazione riscontrata e il livello di priorità annesso. Nella Sezione 4.2.1 si presenteranno esempi di violazioni riscontrate da Snort relative a entrambe le interfacce.

Squid

Il proxy utilizzato all'interno dell'infrastruttura è *Squid*. Esso funge da intermediario tra le richieste dei client e i server di destinazione su Internet. Ciò consente di gestire il traffico in modo efficiente e sicuro. È stata predisposta una procedura di autenticazione tramite credenziali. In sostanza, nel momento in cui un utente effettua una richiesta di connessione a un server su Internet, il proxy chiede di inserire username e password (Figura 3.2). Se le credenziali risultano corrette la richiesta è autorizzata e la connessione va a buon fine, altrimenti essa viene bloccata. Come si può notare anche dalla Figura 3.2, Squid è stato assegnato alla porta 3128 (la porta di default) e all'indirizzo IP corrispondente all'interfaccia interna.

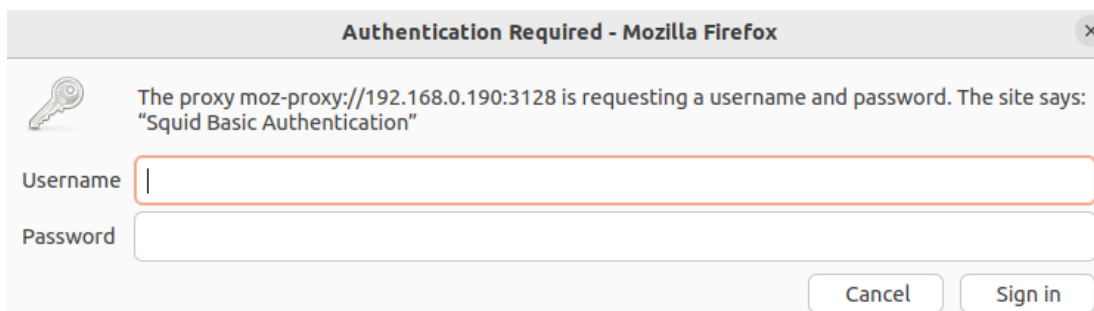


Figura 3.2: Richiesta di credenziali di accesso da parte di Squid

Sono stati predisposte due coppie di credenziali valide. L'amministratore di sistema può comunque aggiungerne altre attraverso il comando:

```
$ sudo htpasswd /etc/squid/passwd nome_utente
```

(sono necessari i permessi di root). Squid è stato inoltre configurato per impedire l'accesso a determinati siti web. I domini in lista nera sono specificati nel file all'indirizzo `/etc/squid/blocked_domains.txt` (Figura 3.3).



Figura 3.3: Lista dei domini web in lista nera

3.1.2 Seconda macchina virtuale: rete interna

La seconda macchina virtuale rappresenta una possibile macchina appartenente alla rete interna dell'infrastruttura, la quale può contenere risorse critiche che devono essere monitorate e protette e alla quale si può accedere dall'esterno tramite SSH. Inoltre, deve essere possibile comunicare con l'esterno e accedere a Internet. Tutto il traffico in entrata e uscita deve passare necessariamente per la macchina Dual-homed Host (Sezione 3.1.1), che lo autorizzerà o scarterà in base alle regole implementate. La macchina è costituita da un'unica interfaccia interna, corrispondente all'indirizzo IP 192.168.0.184. In questa macchina virtuale viene installato *Tripwire*, il quale verrà descritto in dettaglio di seguito.

Tripwire

L'obiettivo principale di Tripwire è rilevare eventuali modifiche non autorizzate ai file di sistema e alle configurazioni, aiutando a identificare potenziali minacce o violazioni della sicurezza. Funziona attraverso la creazione di una "baseline" o snapshot iniziale dei file di sistema critici. Successivamente, controlla periodicamente questi file e confronta le loro firme crittografiche o i valori hash con quelli della baseline. Se vi sono state modifiche non autorizzate, Tripwire segnalerà l'incidente, consentendo agli amministratori di sicurezza di intervenire prontamente. Al fine di ottenere un controllo efficace ed efficiente, Tripwire richiede una configurazione iniziale accurata e una gestione continua per garantire che le modifiche legittime siano adeguatamente autorizzate e che solo le modifiche sospette vengano segnalate. Sono stati definiti tre livelli di severità:

1. **33 (Low)**: File non critici che hanno un impatto minimo sulla sicurezza.
2. **66 (Medium)**: File non critici che potrebbero avere un impatto significativo sulla sicurezza.
3. **100 (High)**: File critici che potrebbero essere punti significativi di vulnerabilità

I livelli sono stati assegnati in base al percorso. Ad esempio, i file all'interno della cartella `/root` sono stati contrassegnati con il livello massimo di severità. Si è scelto di non monitorare tutti i file presenti all'interno della macchina, ma di concentrarsi su quelli più importanti e critici, ignorando di fatto i file soggetti a modifiche frequenti e non significative. Creato il database di partenza, possono essere eseguiti controlli periodici per riscontrare eventuali violazioni di integrità. In Figura 3.4 è rappresentato un report di Tripwire eseguito tramite il comando:

```
$ sudo tripwire --check
```

(sono necessari i permessi di root).

```

client@Client: ~
=====
Report Summary:
=====
Host name:          Client
Host IP address:    127.0.1.1
Host ID:            None
Policy file used:   /etc/tripwire/tw.pol
Configuration file used: /etc/tripwire/tw.cfg
Database file used: /var/lib/tripwire/Client.twd
Command line used:  tripwire --check
=====
Rule Summary:
=====
-----
Section: Unix File System
-----

Rule Name          Severity Level   Added   Removed   Modified
-----
Other binaries      66               0       0         0
Tripwire Binaries   100              0       0         0
Other libraries     66               0       0         0
Root file-system executables 100              0       0         0
Tripwire Data Files 100              0       0         0
System boot changes 100              0       0         0
Root file-system libraries 100              0       0         0
(/lib)
Critical system boot files 100              0       0         0
Other configuration files 66               0       0         0
(/etc)
Boot Scripts        100              0       0         0
Security Control     66               0       0         0
Root config files    100              0       0         0
(/root)
Devices & Kernel information 100              0       0         0
(/dev)
Invariant Directories 66               0       0         0

Total objects scanned: 62046
Total violations found: 0

```

Figura 3.4: Esempio di un controllo di integrità di Tripwire con nessuna violazione riscontrata

Dalla Figura si può notare l'assenza di violazioni. In seguito, all'interno del Capitolo 4.2, si effettuerà una prova di aggiunta di un nuovo file nel sistema e si testerà l'effettivo funzionamento di Tripwire.

3.2 Applicazione delle policies di sicurezza

In questa sezione sarà presentata l'implementazione delle policies di sicurezza definite in fase di progettazione.

Policy di default "deny all"

Impostare una policy che, di base, blocca tutti i pacchetti è una scelta comune poiché garantisce una maggiore sicurezza e precisione nella definizione delle regole rispetto ad una policy di default "allow all". In questo modo tutto il traffico non autorizzato dalle altre regole attive è esplicitamente bloccato. Tale regola è riportata in Figura 3.5.

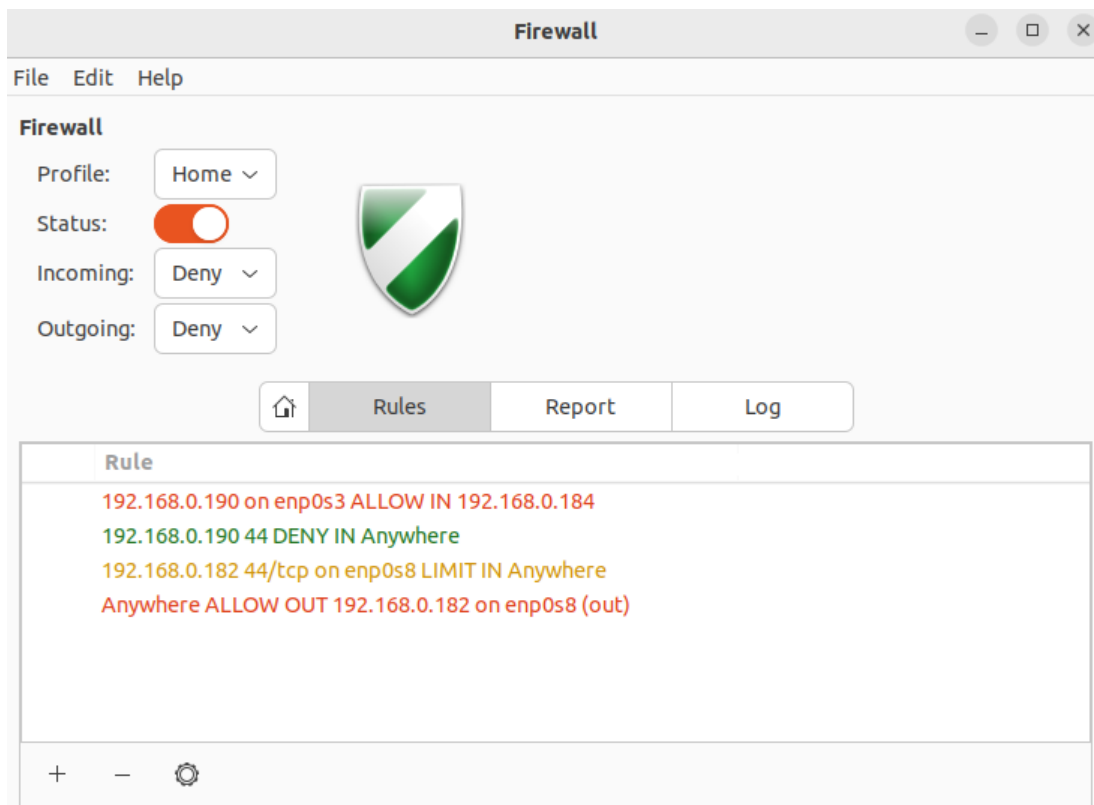


Figura 3.5: Policy di default "deny all"

In essa è possibile notare che il firewall è impostato su *Deny*, sia per quanto riguarda le richieste in entrata (Incoming) sia per quelle in uscita (Outgoing). Andando più nello specifico della sezione "Rule", la regola "ALLOW IN" consente all'interfaccia interna della macchina Dual-homed Host di accettare le richieste provenienti dalla macchina nella rete interna. Al contrario, la regola "DENY IN" rifiuta ogni tipo di richiesta effettuata sulla porta 44 dell'interfaccia interna.

Gestione delle connessioni

Per rendere più difficile ad un attaccante individuare e attaccare il servizio SSH, si è deciso di utilizzare una porta diversa da quella predefinita per le connessioni SSH (la 22). Infatti, è stata configurata la porta 44 per questo tipo di connessioni al posto della 22. La policy di gestione delle connessioni, come specificato nel capitolo precedente, è utile a prevenire attacchi di Denial of Service. Riprendendo la Figura 3.5, la regola *LIMIT IN* è quella che implementa tale policy. Il numero massimo di connessioni SSH (sulla porta 44) che possono essere stabilite in un intervallo di tempo di 30 secondi è stato impostato a 5. Al sesto tentativo di connessione, si riceverà come risposta un messaggio di timeout¹.

Logging e monitoraggio

La policy di logging e monitoraggio è stata implementata tramite l'utilizzo di Snort e Tripwire. Come già presentato all'interno della Sezione 3, Snort è lo strumento in grado di monitorare il traffico all'interno della rete e registrare, su appositi file di log, attività sospette o potenzialmente dannose. Tripwire è, invece, il software utilizzato al fine di verificare l'integrità dei file presenti nel sistema attraverso attività di monitoraggio periodiche.

Filtraggio del traffico in base al livello di applicazione

Per garantire un filtraggio del traffico in base al livello di applicazione, sono state implementate delle regole con la finalità di bloccare connessioni P2P e connessioni SSH per determinate porte e indirizzi IP. Come mostrato in Figura 3.6, sono tre le regole che vanno a definire questa caratteristica. La prima regola blocca in uscita le porte nel range 6881:6889, solitamente associate al software BitTorrent, famoso per il traffico P2P. La seconda regola blocca le connessioni in entrata sull'interfaccia interna (corrispondente all'IP 192.168.0.190) alla porta destinata alle connessioni SSH, ossia la 44. La terza regola consente le connessioni SSH alla porta 44 sull'interfaccia esterna (corrispondente all'IP 192.168.0.182), ma ne limita il numero in un determinato intervallo di tempo (come già affermato nel paragrafo relativo alla gestione delle connessioni).

¹Questo caso verrà analizzato più nel dettaglio nella Sezione 4.2.2

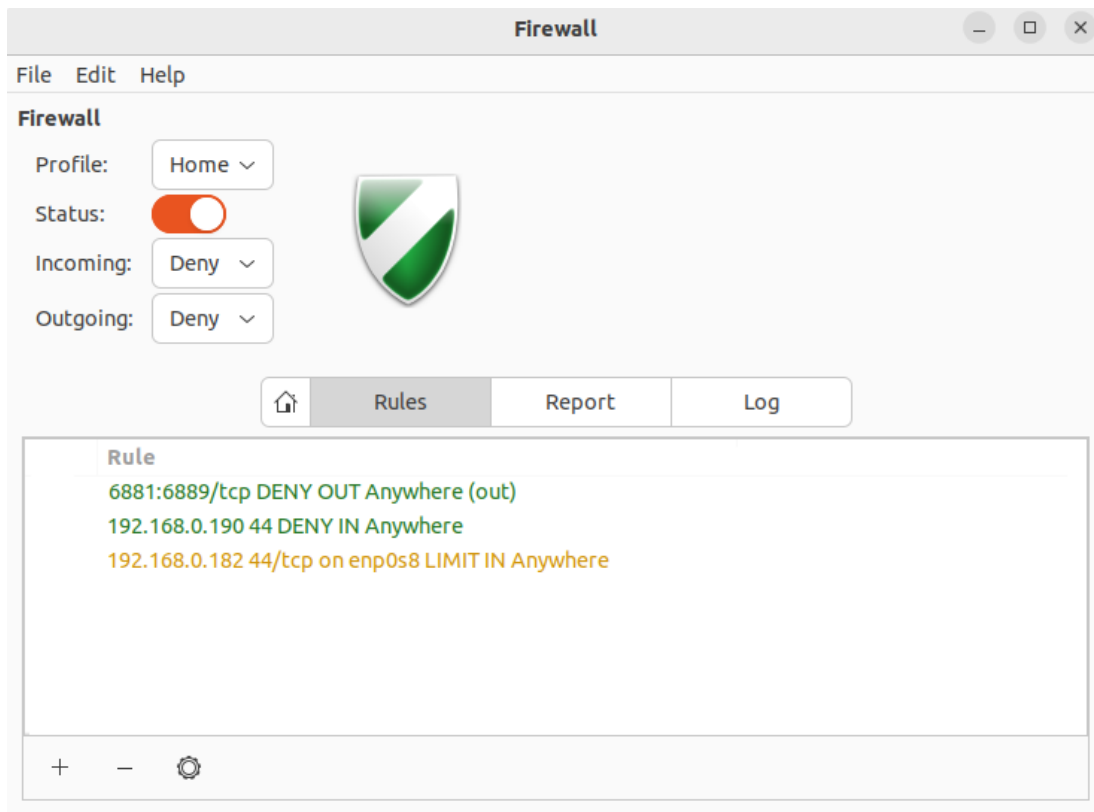


Figura 3.6: Esempio di regole per il filtraggio del traffico

Politiche di autenticazione e accesso

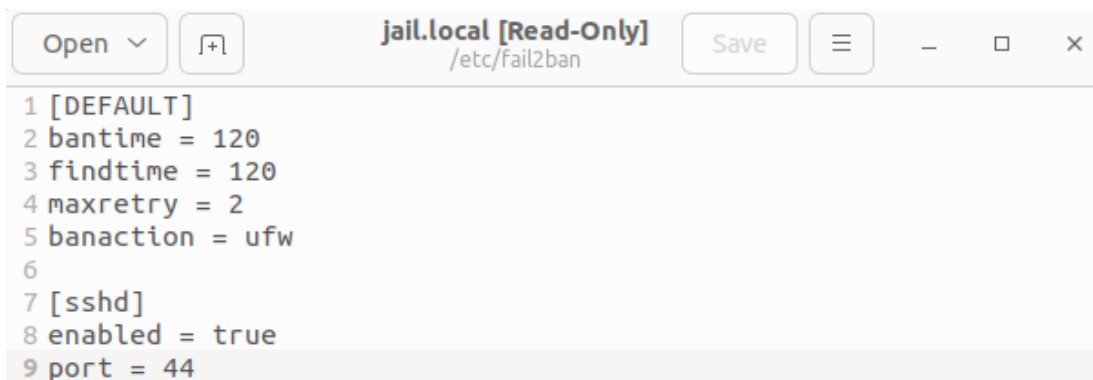
Sono state implementate politiche di sicurezza che permettono l'accesso via SSH tramite l'inserimento di credenziali valide. Si ha anche la possibilità di configurare una connessione SSH tramite chiavi di accesso in modo da semplificare il processo di autenticazione. È stato anche configurato il proxy Squid per consentire la navigazione su Internet esclusivamente dopo aver inserito credenziali valide. Nella Sezione 4.2.1 si presenteranno esempi relativi all'interazione dell'utente con Squid nel processo di autenticazione per poter navigare su Internet.

Protezione da attacchi di scansione di porte

La politica di protezione da attacchi di scansione di porte è stata implementata tramite la configurazione del software *Fail2Ban*, il quale blocca temporaneamente gli indirizzi IP che inseriscono ripetutamente credenziali errate in un determinato lasso di tempo. La policy è stata impostata in modo da bloccare l'indirizzo IP sospetto per due minuti nel momento in cui vengono inserite credenziali errate per 3 volte consecutive. Infatti, al successivo tentativo di connessione nei due minuti di ban si riceverà un messaggio di timeout. In questo modo il sistema riesce a

diminuire drasticamente l'efficacia degli attacchi di tipo brute force. In Figura 3.7 è rappresentato il file di configurazione locale di Fail2Ban. La configurazione impostata ha le seguenti caratteristiche:

- Il tempo di ban e quello di ricerca corrispondono entrambi a 120 secondi.
- I tentativi massimi consentiti equivalgono a 3 (due retry dopo il primo errore).
- Il sistema si interfaccia con le azioni del firewall Ufw.
- La porta da controllare per le connessioni SSH (44).



```
1 [DEFAULT]
2 bantime = 120
3 findtime = 120
4 maxretry = 2
5 banaction = ufw
6
7 [sshd]
8 enabled = true
9 port = 44
```

Figura 3.7: Configurazione di Fail2Ban

Capitolo 4

Installazione e Test

In questo capitolo si guiderà il lettore nell'installazione dei componenti dell'infrastruttura. Successivamente, si effettuerà il test di tutte le funzionalità implementate e si verificherà l'effettivo rispetto di tutte le regole stabilite. Allo stesso tempo, si controllerà la robustezza di tutta l'infrastruttura attraverso una serie di controlli di sicurezza.

4.1 Installazione

Per installare le macchine virtuali che sono alla base dell'infrastruttura sicura, è necessario installare sul proprio computer un software per l'esecuzione di macchine virtuali, come VirtualBox.

I file OVA delle due macchine virtuali sono disponibili al seguente indirizzo OneDrive: [Link](#).

Se si dovessero riscontrare problemi all'avvio delle macchine, verificare che le interfacce di rete siano state correttamente rilevate e che le risorse assegnate siano sufficienti. Una volta installate, è possibile avviare le due macchine e testare l'intero sistema.

4.2 Test

Come già descritto nel Capitolo 3, sono state implementate due diverse macchine virtuali appartenenti alla rete interna:

1. Dual-homed Host
2. Macchina "Client"

4.2.1 Dual-homed Host

All'avvio della macchina virtuale Dual-Homed Host, sono disponibili due diversi account:

1. **screening**: account con permessi di *root*, necessario per l'implementazione e la gestione delle policy e della sicurezza di tutta l'infrastruttura. È riservato all'amministratore di sistema. La password per poter accedere a tale account è: "univpm2223".
2. **guest**: account senza permessi di root, predisposto per l'accesso via SSH degli utenti. La password per poter accedere a tale account è: "guestscreening".

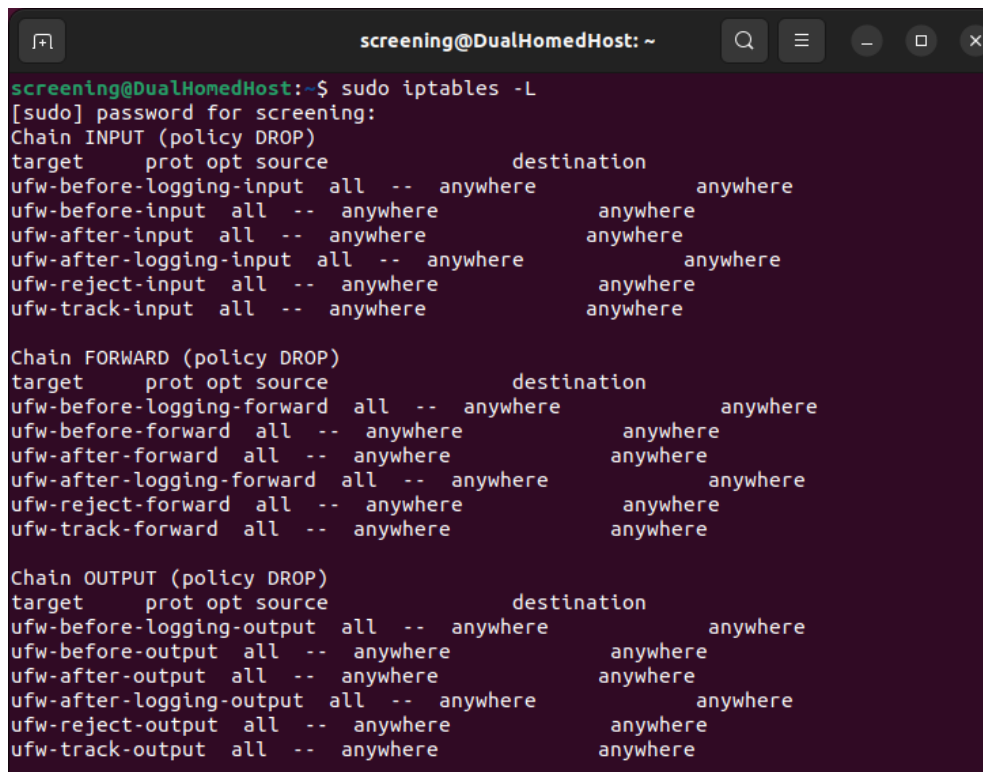
Una volta effettuato l'accesso all'account **screening**, l'amministratore di sistema ha il completo controllo sui programmi per la sicurezza.

Iptables

Sono disponibili diverse modalità con cui si possono implementare nuove policies di sicurezza. La prima modalità prevede la scrittura di regole con la sintassi di Iptables, direttamente dal terminale. Per visualizzare la lista di tutte le regole implementate e attive si può utilizzare il comando:

```
$ sudo iptables -L
```

In Figura 4.1 è rappresentato un esempio dell'utilizzo di questo comando e il risultato ottenuto, ossia un estratto di tutte le regole di Iptables implementate.

A terminal window titled 'screening@DualHomedHost: ~' with search, menu, and window control icons. The user has run 'sudo iptables -L'. The output shows three chains: INPUT, FORWARD, and OUTPUT, all with a policy of DROP. Each chain lists several rules with target names like 'ufw-before-logging-input', 'ufw-after-input', etc., and their parameters for protocol, options, source, and destination.

```
screening@DualHomedHost:~$ sudo iptables -L
[sudo] password for screening:
Chain INPUT (policy DROP)
target     prot opt source                destination
ufw-before-logging-input  all  --  anywhere                anywhere
ufw-before-input          all  --  anywhere                anywhere
ufw-after-input           all  --  anywhere                anywhere
ufw-after-logging-input   all  --  anywhere                anywhere
ufw-reject-input          all  --  anywhere                anywhere
ufw-track-input           all  --  anywhere                anywhere

Chain FORWARD (policy DROP)
target     prot opt source                destination
ufw-before-logging-forward all  --  anywhere                anywhere
ufw-before-forward        all  --  anywhere                anywhere
ufw-after-forward         all  --  anywhere                anywhere
ufw-after-logging-forward all  --  anywhere                anywhere
ufw-reject-forward        all  --  anywhere                anywhere
ufw-track-forward         all  --  anywhere                anywhere

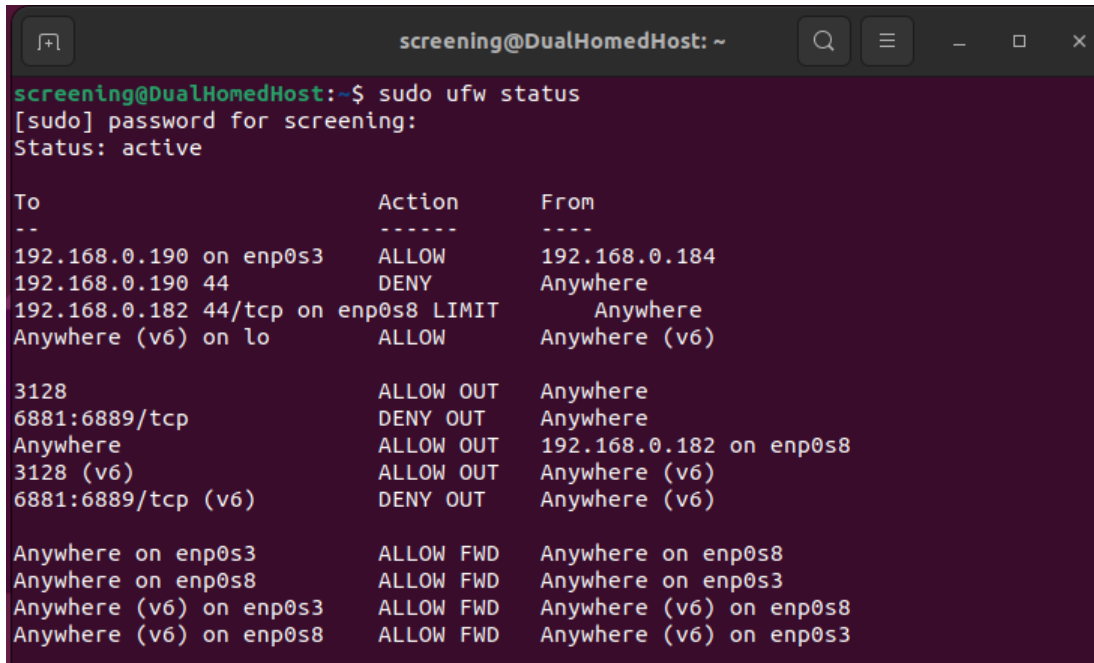
Chain OUTPUT (policy DROP)
target     prot opt source                destination
ufw-before-logging-output all  --  anywhere                anywhere
ufw-before-output         all  --  anywhere                anywhere
ufw-after-output          all  --  anywhere                anywhere
ufw-after-logging-output  all  --  anywhere                anywhere
ufw-reject-output         all  --  anywhere                anywhere
ufw-track-output          all  --  anywhere                anywhere
```

Figura 4.1: Estratto di tutte le regole di Iptables implementate

Come si può notare, le regole di Iptables non risultano facilmente comprensibili. A questo proposito, come già accennato in precedenza, Ufw permette una visualizzazione e una scrittura più agevole di regole rispetto a Iptables. Si può utilizzare il comando:

```
$ sudo ufw status
```

In questo modo, il risultato ottenuto è simile a quello in Figura 4.2.



```
screening@DualHomedHost:~$ sudo ufw status
[sudo] password for screening:
Status: active

To Action From
--
192.168.0.190 on enp0s3 ALLOW 192.168.0.184
192.168.0.190 44 DENY Anywhere
192.168.0.182 44/tcp on enp0s8 LIMIT Anywhere
Anywhere (v6) on lo ALLOW Anywhere (v6)

3128 ALLOW OUT Anywhere
6881:6889/tcp DENY OUT Anywhere
Anywhere ALLOW OUT 192.168.0.182 on enp0s8
3128 (v6) ALLOW OUT Anywhere (v6)
6881:6889/tcp (v6) DENY OUT Anywhere (v6)

Anywhere on enp0s3 ALLOW FWD Anywhere on enp0s8
Anywhere on enp0s8 ALLOW FWD Anywhere on enp0s3
Anywhere (v6) on enp0s3 ALLOW FWD Anywhere (v6) on enp0s8
Anywhere (v6) on enp0s8 ALLOW FWD Anywhere (v6) on enp0s3
```

Figura 4.2: Esempio elenco delle regole implementate con Ufw

Il miglioramento rispetto alla visualizzazione proposta direttamente con il comando di Iptables è evidente. Tuttavia, è disponibile anche un'interfaccia grafica tramite il software Gufw (Figura 4.3). L'utilizzo di Gufw consente di aggiungere ed eliminare rapidamente nuove regole, attraverso i pulsanti posizionati in basso nella schermata. Inoltre, nella scheda "Rules" sono presenti tutte le regole attualmente attive, con colorazioni diverse in base alla tipologia (rosso, verde e giallo rispettivamente per ALLOW, DENY e LIMIT. La scheda "Report" include dettagli come le porte e le applicazioni coinvolte, i protocolli utilizzati e gli indirizzi IP di destinazione. La scheda "Log" registra ed elenca tutte le operazioni effettuate relative al firewall.

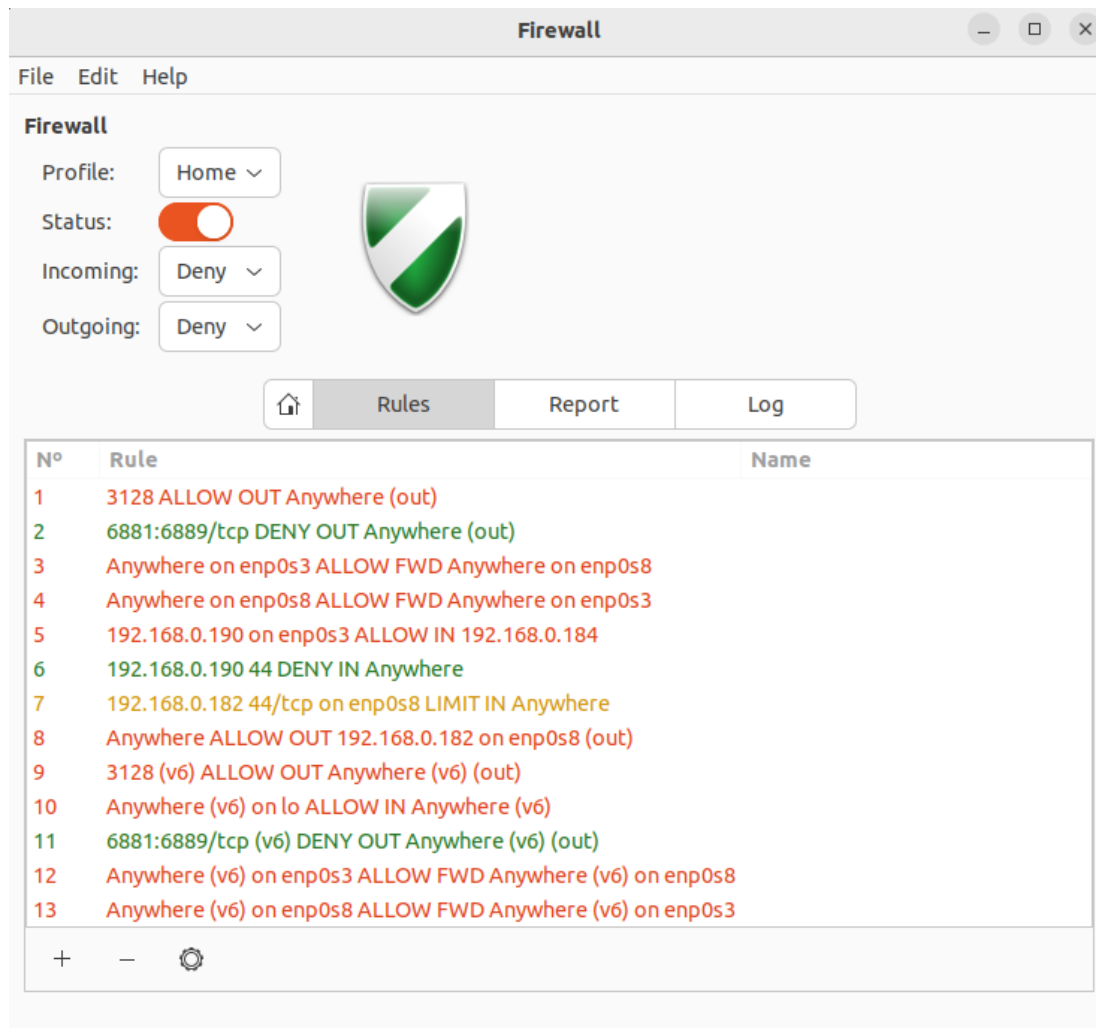


Figura 4.3: Esempio elenco delle regole implementate visualizzate con Gufw

Come già accennato, cliccando il pulsante "+" posizionato in basso è possibile aggiungere una nuova regola. In Figura 4.4 si riporta un esempio di questo tipo di operazione.

The screenshot shows the 'Add a Firewall Rule' window with the 'Advanced' tab active. The 'Name' field is empty and has a red border. The 'Insert' field contains '0'. The 'Policy' is 'Allow', 'Direction' is 'In', 'Interface' is 'All Interfaces', 'Log' is 'Do not Log', and 'Protocol' is 'Both'. The 'From' and 'To' fields are both set to 'IP' and 'Port', with red 'X' icons indicating they are required or invalid. The 'Close' and 'Add' buttons are at the bottom right.

Figura 4.4: Esempio di aggiunta di una nuova regola con Gufw

È possibile selezionare ed utilizzare policies preconfigurate (nella scheda "Pre-configured"), aggiungere policies semplici e con pochi dettagli (nella scheda "Simple") oppure specificare tutti i parametri all'interno della scheda "Advanced" (come mostrato in Figura 4.4).

Snort

Come già accennato nella Sezione 3.1.1, le due istanze di Snort sono configurate in modo da iniziare a controllare sin dall'avvio della macchina. Sono stati predisposti due file di log. Ogni riscontro di Snort viene riportato nel file specifico. Nelle Figure 4.5 e 4.6 si rappresentano due esempi relativi a violazioni riscontrate rispettivamente sull'interfaccia esterna e interna. Il primo esempio è caratterizzato da tre diverse violazioni: due derivano dalle "community rules" e riguardano operazioni di PING tra l'interfaccia esterna e un server di *Google* (corrispondente all'IP 216.58.204.131), una è invece stata inserita per testare

il funzionamento delle "local rules" con lo stesso tipo di violazione PING e un messaggio personalizzato (in Figura 4.7 è rappresentata la definizione della local rule corrispondente). Allo stesso modo, il secondo esempio riguarda nuovamente un'operazione di PING, questa volta nella rete interna. Si hanno quindi due messaggi provenienti dalle "community rules", il primo che specifica anche il sistema operativo (Unix in questo caso), il secondo che si limita a riportare il generico PING. Anche in questo esempio è presente la "local rules" corrispondente, a dimostrazione del fatto che entrambi i file contenenti regole siano correttamente utilizzati da *Snort*.

```

alert_fast.txt [Read-Only]
/var/log/snort/ExternalNetwork

210 08/07-12:20:25.647195 [**] [1:29456:3] "PROTOCOL-ICMP Unusual PING detected" [**]
[Classification: Information Leak] [Priority: 2] [AppID: ICMP] {ICMP} 192.168.0.182 ->
216.58.204.131
211 08/07-12:20:25.672676 [**] [1:1000001:1] "ICMP connection test" [**] [Priority: 0] [AppID:
ICMP] {ICMP} 216.58.204.131 -> 192.168.0.182
212 08/07-12:20:25.672676 [**] [1:408:8] "PROTOCOL-ICMP Echo Reply" [**] [Classification: Misc
activity] [Priority: 3] [AppID: ICMP] {ICMP} 216.58.204.131 -> 192.168.0.182

```

Figura 4.5: Esempio di una violazione riscontrata da Snort sull'interfaccia esterna

```

alert_fast.txt [Read-Only]
/var/log/snort/InternalNetwork

742 07/30-11:26:27.950944 [**] [1:366:11] "PROTOCOL-ICMP PING Unix" [**] [Classification: Misc
activity] [Priority: 3] [AppID: ICMP] {ICMP} 192.168.0.184 -> 192.168.0.190
743 07/30-11:26:27.950944 [**] [1:1000001:1] "ICMP connection test" [**] [Priority: 0] [AppID:
ICMP] {ICMP} 192.168.0.184 -> 192.168.0.190
744 07/30-11:26:27.950944 [**] [1:384:8] "PROTOCOL-ICMP PING" [**] [Classification: Misc activity]
[Priority: 3] [AppID: ICMP] {ICMP} 192.168.0.184 -> 192.168.0.190

```

Figura 4.6: Esempio di una violazione riscontrata da Snort sull'interfaccia interna

```

local.rules [Read-Only]
/usr/local/etc/rules

1 alert icmp any any -> $HOME_NET any (msg:"ICMP connection test"; sid:1000001; rev:1;)|

```

Figura 4.7: Esempio di una local rule di Snort

Squid

Squid è il proxy utilizzato per l'autenticazione degli utenti, necessaria per la navigazione su Internet. All'apertura del browser, l'utente visualizzerà un prompt per l'immissione di credenziali (Figura 4.8).

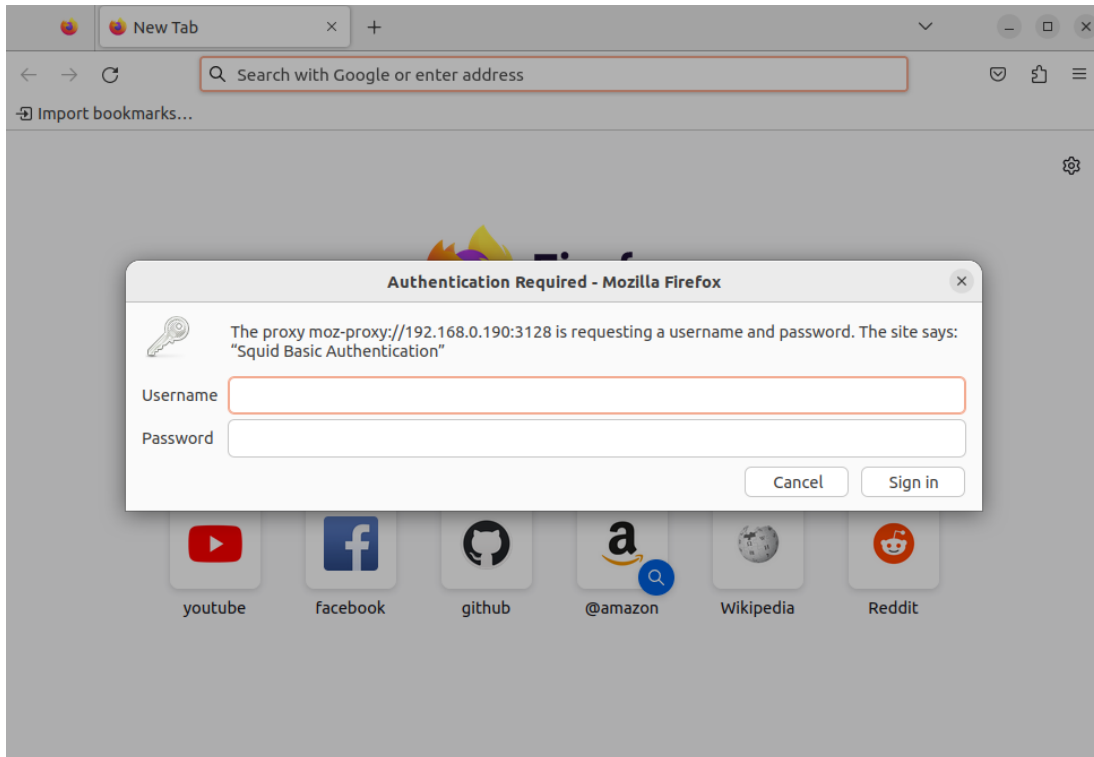


Figura 4.8: Richiesta di credenziali all'apertura del browser

Sono state predisposte due coppie di credenziali di accesso:

1.
 - **Username:** univpm
 - **Password:** advanced
2.
 - **Username:** univpm2
 - **Password:** cybersecurity

L'amministratore di sistema può aggiungere nuove coppie di credenziali tramite il comando:

```
$ sudo htpasswd /etc/squid/passwd nome_utente
```

Un esempio di questa operazione è rappresentato in Figura 4.9.

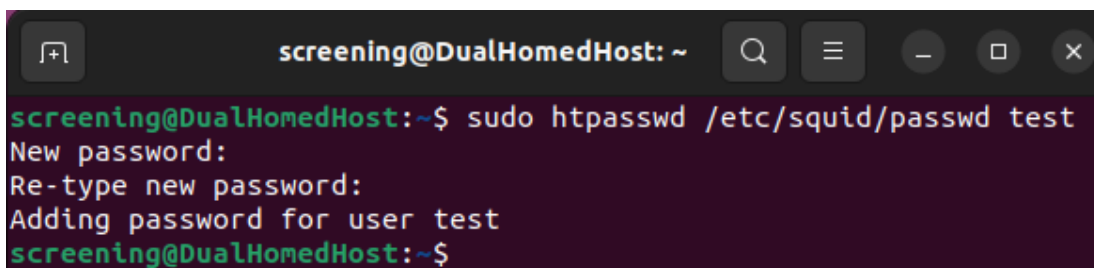


Figura 4.9: Aggiunta di una nuova coppia di credenziali

Una volta inserita una coppia di credenziali valida, il sistema autorizza la navigazione su Internet. Tuttavia, se un utente autorizzato effettua una richiesta per uno dei domini in lista nera (ad esempio *Youtube*), il risultato ottenuto sarà quello rappresentato in Figura 4.10.

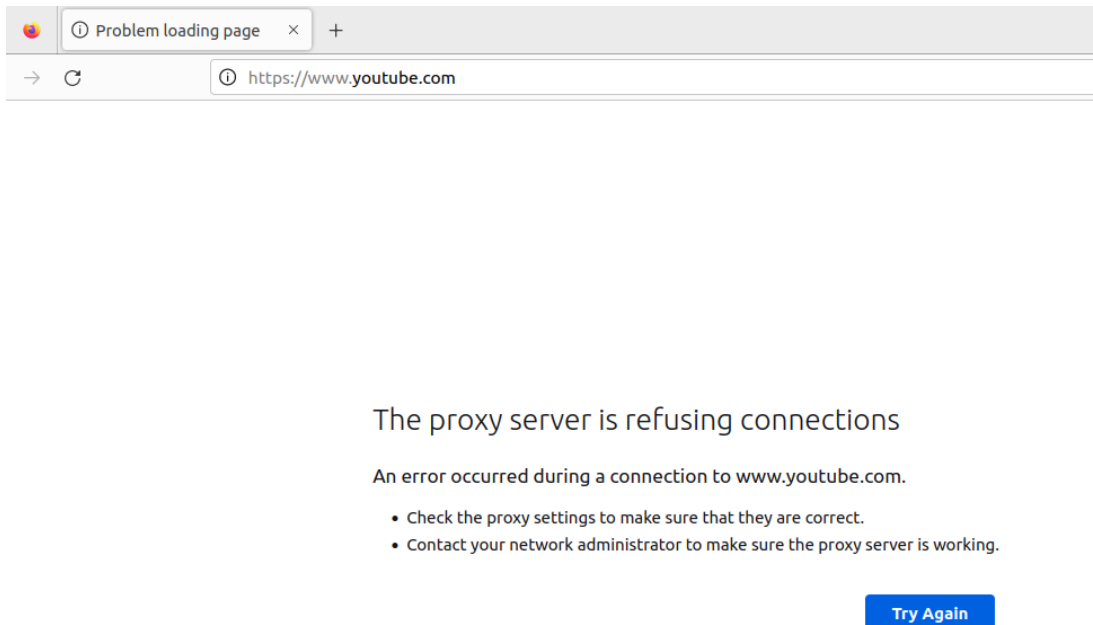


Figura 4.10: Esempio di una richiesta per uno dei domini in lista nera

Non è possibile navigare in assenza di credenziali valide. Infatti, se un utente prova a effettuare richieste HTTP via terminale tramite il comando *curl*, vedrà restituirsi un messaggio di errore come quello in Figura 4.11.

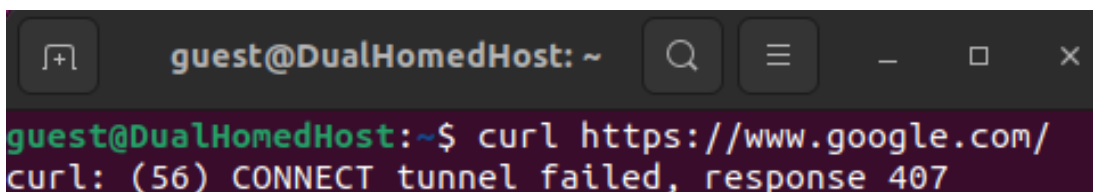


Figura 4.11: Esempio di curl senza l'inserimento di credenziali e relativo errore

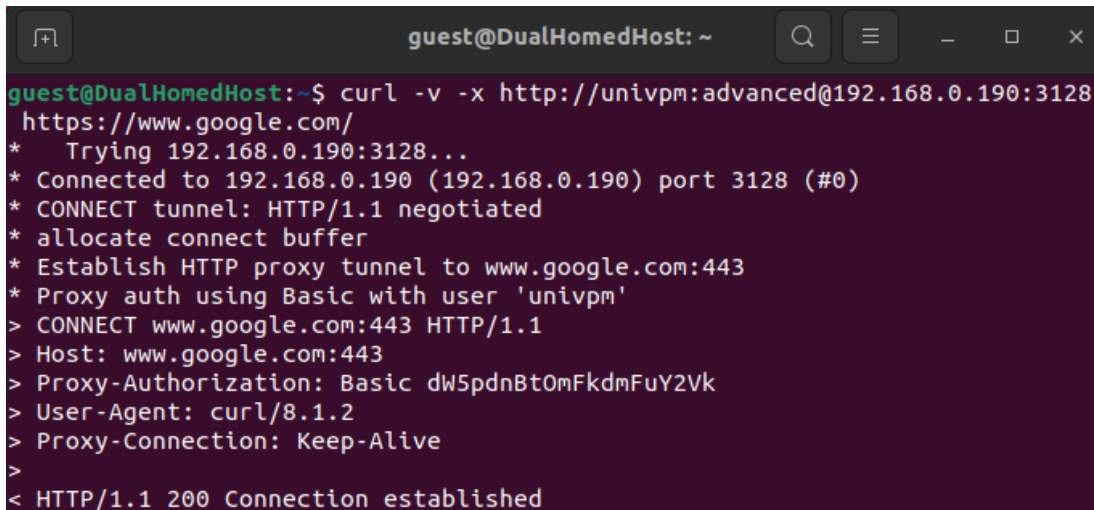
L'immissione delle credenziali è possibile anche per richieste via terminale. Il comando da utilizzare è il seguente:

```
$ curl -v -x http://*username*:~password*@192
.168.0.190:3128 *url*
```

Ad esempio, un comando valido potrebbe essere:

```
$ curl -v -x http://univpm:advanced@192.168.0.190:3128  
https://www.google.com/
```

In Figura 4.12 è rappresentato proprio questa casistica: l'utente che inserisce le credenziali correttamente verrà autenticato dal proxy Squid e riceverà come risposta la pagina richiesta. In questo caso l'utente corrisponde a "univpm".



```
guest@DualHomedHost: ~  
guest@DualHomedHost:~$ curl -v -x http://univpm:advanced@192.168.0.190:3128  
https://www.google.com/  
* Trying 192.168.0.190:3128...  
* Connected to 192.168.0.190 (192.168.0.190) port 3128 (#0)  
* CONNECT tunnel: HTTP/1.1 negotiated  
* allocate connect buffer  
* Establish HTTP proxy tunnel to www.google.com:443  
* Proxy auth using Basic with user 'univpm'  
> CONNECT www.google.com:443 HTTP/1.1  
> Host: www.google.com:443  
> Proxy-Authorization: Basic dW5pdnBtOmFkdmFuY2Vk  
> User-Agent: curl/8.1.2  
> Proxy-Connection: Keep-Alive  
>  
< HTTP/1.1 200 Connection established
```

Figura 4.12: Esempio di curl con inserimento di credenziali valide per l'utente "univpm"

4.2.2 Macchina "Client"

All'avvio della macchina virtuale "Client", sono disponibili due diversi account:

1. **client**: account con permessi di *root*, necessario per la gestione e il controllo dei file presenti sulla macchina. È riservato all'amministratore di sistema. La password per poter accedere a tale account è: "gruppootto".
2. **guest**: account senza permessi di root, predisposto per l'accesso via SSH degli utenti. La password per poter accedere a tale account è: "guestclient".

Una volta effettuato l'accesso all'account **client**, l'amministratore di sistema ha il completo controllo sui file presenti sulla macchina.

Tripwire

Ogni aggiunta, modifica o cancellazione di file in zone specifiche (critiche) del file system viene registrata da Tripwire. Come già affermato nella sezione 3.1.2, il comando da utilizzare per effettuare un controllo con Tripwire è il seguente:

```
$ sudo tripwire --check
```

Partendo da una situazione con 0 violazioni riscontrate, quindi con database di Tripwire aggiornato alle ultime modifiche, si possono effettuare dei test di modifica di file in diverse cartelle. Ad esempio, si aggiunge il file *add.txt* e si elimina *remove.txt* nella cartella `/etc`, mentre si modifica il file *modify.txt* nella cartella `/dev`. A questo punto si effettua un controllo di integrità con Tripwire. In Figura 4.13 è mostrato l'esito della verifica.

```

=====
Rule Summary:
=====

-----
Section: Unix File System
-----

Rule Name                Severity Level   Added   Removed   Modified
-----
Other binaries           66              0       0         0
Tripwire Binaries        100             0       0         0
Other libraries           66              0       0         0
Root file-system executables 100             0       0         0
Tripwire Data Files      100             0       0         0
System boot changes      100             0       0         0
Root file-system libraries 100             0       0         0
(/lib)
Critical system boot files 100             0       0         0
* Other configuration files 66              1       1         1
(/etc)
Boot Scripts             100             0       0         0
Security Control          66              0       0         0
Root config files         100             0       0         0
(/root)
* Devices & Kernel information 100             0       0         1
(/dev)
Invariant Directories     66              0       0         0

Total objects scanned: 62048
Total violations found: 4

```

Figura 4.13: Esempio di un controllo di integrità di Tripwire con alcune violazioni riscontrate

Si nota immediatamente la presenza di violazioni all'interno delle cartelle `/etc` e `/dev`. Andando più nello specifico, il report prosegue elencando singolarmente le violazioni riscontrate (Figura 4.14).

```

client@Client: /
=====
Object Summary:
=====
# Section: Unix File System
-----
Rule Name: Other configuration files (/etc)
Severity Level: 66
-----
Added:
"/etc/add.txt"
Removed:
"/etc/remove.txt"
Modified:
"/etc"
-----
Rule Name: Devices & Kernel information (/dev)
Severity Level: 100
-----
Modified:
"/dev/modify.txt"
=====
Error Report:
=====
No Errors
-----
*** End of report ***

```

Figura 4.14: Dettagli delle violazioni riscontrate da Tripwire

Tripwire ha correttamente registrato e segnalato tutte le modifiche effettuate. Infatti, sono riportate tutte le operazioni effettuate: nella cartella `/etc` il file `add.txt` risulta essere tra gli "Added" e `remove.txt` tra i "Removed", in `/dev` il file `modify.txt` risulta essere tra i "Modified". Il report viene salvato al percorso¹ `/var/lib/tripwire/report/<name>.twr`. Il comando per approvare le violazioni riscontrate e aggiornare il database di Tripwire è il seguente:

```
$ sudo tripwire --update --twrfile /var/lib/tripwire/
report/<name>.twr
```

¹<name>.twr indica il report da utilizzare; è costituito dall'hostname della macchina e dal timestamp del report

Questa operazione richiede l'inserimento di una local passphrase, equivalente a: "advancedcybersecurity?". Una volta eseguito questo comando, si torna ad una situazione di base con nessuna violazione presente; Tripwire registrerà tutte le modifiche ai file monitorati da quel momento in poi.

Connessione via SSH

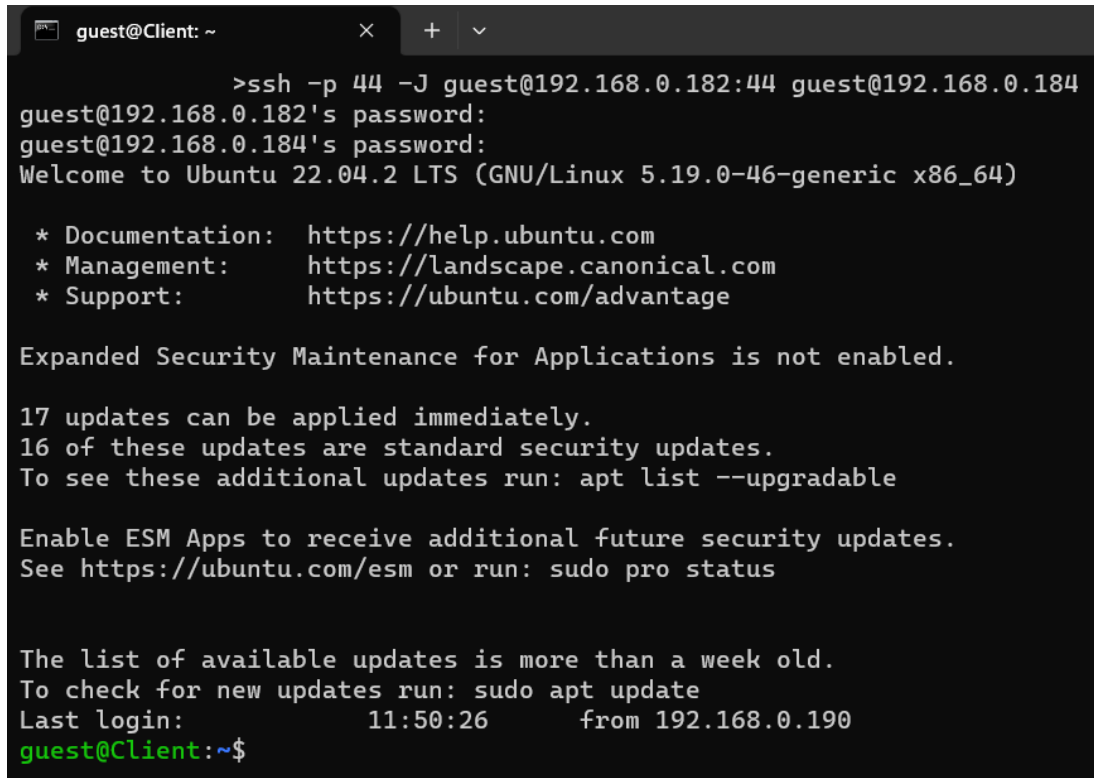
La macchina "Client" è stata predisposta in modo da poter essere raggiunta da remoto tramite connessioni SSH sulla porta 44. Tutte le richieste di connessione devono passare necessariamente per il Dual-Homed Host; infatti, è quest'ultimo che si occupa eventualmente di accettare la richiesta e di fungere da "ponte" tra l'esterno e l'interno della rete. A questo proposito, nel comando deve essere inserito il parametro **-J** seguito dall'indirizzo dell'interfaccia esterna della macchina Dual-Homed Host. In questo modo si specifica che essa deve essere utilizzata per l'operazione di "jumping" tra l'esterno e l'interno della rete. I due account della macchina "Client" sono entrambi raggiungibili via SSH. Per raggiungere l'account con privilegi di root il comando da utilizzare è il seguente:

```
$ ssh -p 44 -J screening@192.168.0.182:44 client@192.168.0.184
```

Al contrario, per raggiungere l'account "ospite", senza privilegi di root, il comando è:

```
$ ssh -p 44 -J guest@192.168.0.182:44 guest@192.168.0.184
```

192.168.0.182 è l'indirizzo IP dell'interfaccia di rete esterna della macchina "Dual-Homed Host". Quando essa riceve la richiesta di connessione SSH e la approva, la richiesta viene passata all'interfaccia interna (IP 192.168.0.190) che la inoltra alla macchina "Client" (IP 192.168.0.184). Per effettuare la connessione SSH è dunque necessario autenticarsi ad entrambe le macchine, inserendo le password dei rispettivi account (Figura 4.15).



```

guest@Client: ~
>ssh -p 44 -J guest@192.168.0.182:44 guest@192.168.0.184
guest@192.168.0.182's password:
guest@192.168.0.184's password:
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.19.0-46-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

17 updates can be applied immediately.
16 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Last login:          11:50:26          from 192.168.0.190
guest@Client:~$

```

Figura 4.15: Esempio di connessione SSH

Per rendere la procedura di connessione SSH più veloce, è possibile configurare delle chiavi SSH da scambiare tra le varie macchine virtuali. In questo modo, il terminale che effettua la richiesta verrà riconosciuto immediatamente ed autorizzato ad accedere senza l'inserimento della password. Innanzitutto, bisogna generare la coppia di chiavi pubblica e privata. Per far questo, bisogna utilizzare il comando da terminale *ssh-keygen* per creare la coppia di chiavi. Questo comando genererà una chiave pubblica (*id_rsa.pub*) e una chiave privata (*id_rsa*) nella directory home (generalmente */.ssh/*).

```
$ ssh-keygen -t rsa
```

Verrà chiesto dove salvare la chiave, si può lasciare il percorso predefinito e semplicemente premere "Invio" per accettarlo. Verrà anche chiesto di inserire una passphrase opzionale per proteggere la chiave privata. Si può inserire una passphrase o lasciarla vuota. Generata la coppia di chiavi SSH, bisogna copiare la chiave pubblica (*id_rsa.pub*) sul server remoto (la macchina su cui autenticarsi). Il comando da utilizzare varia in base al sistema operativo. Per Windows sono necessari due comandi per questa operazione:

```
$ scp -P 44 .ssh/id_rsa.pub guest@192.168.0.182:~/\
ssh/authorized_keys
```



```
$ scp -P 44 .ssh/id_rsa.pub guest@192.168.0.182:~/.  
ssh/id_rsawin.pub
```

Si può ora accedere alla macchina Dual-homed Host tramite SSH:

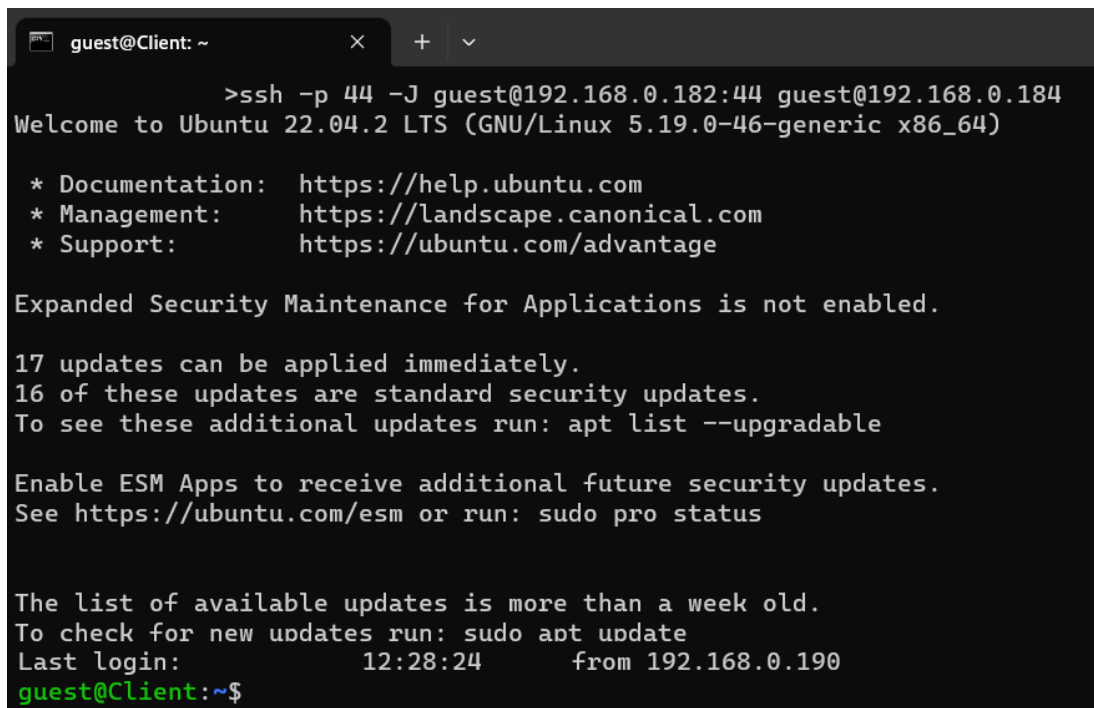
```
$ ssh -p 44 guest@192.168.0.182
```

Si può notare che il sistema non ha richiesto l'inserimento della password, in quanto ha utilizzato la chiave SSH per l'autenticazione.

A questo punto, bisogna scambiare la chiave anche con la macchina "Client". Per far ciò, una volta entrati via SSH sulla macchina "Dual-homed Host", il comando da utilizzare è:

```
$ ssh-copy-id -f -p 44 -i ~/.ssh/id_rsawin.pub  
guest@192.168.0.184
```

Se tutto il processo è andato a buon fine, da questo momento in poi è possibile connettersi dall'esterno via SSH sulla macchina "Client" senza dover inserire nessuna password, come mostrato in Figura 4.16.



```
guest@Client: ~  
>ssh -p 44 -J guest@192.168.0.182:44 guest@192.168.0.184  
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.19.0-46-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
Expanded Security Maintenance for Applications is not enabled.  
  
17 updates can be applied immediately.  
16 of these updates are standard security updates.  
To see these additional updates run: apt list --upgradable  
  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
Last login:      12:28:24      from 192.168.0.190  
guest@Client:~$
```

Figura 4.16: Esempio di connessione SSH con l'utilizzo delle chiavi da macchina Windows

Se il sistema operativo della macchina esterna è invece basato su Linux, la procedura richiede meno passaggi. Infatti, dalla macchina esterna a "Dual-homed Host" è sufficiente il comando:

```
$ ssh-copy-id -p 44 guest@192.168.0.182
```

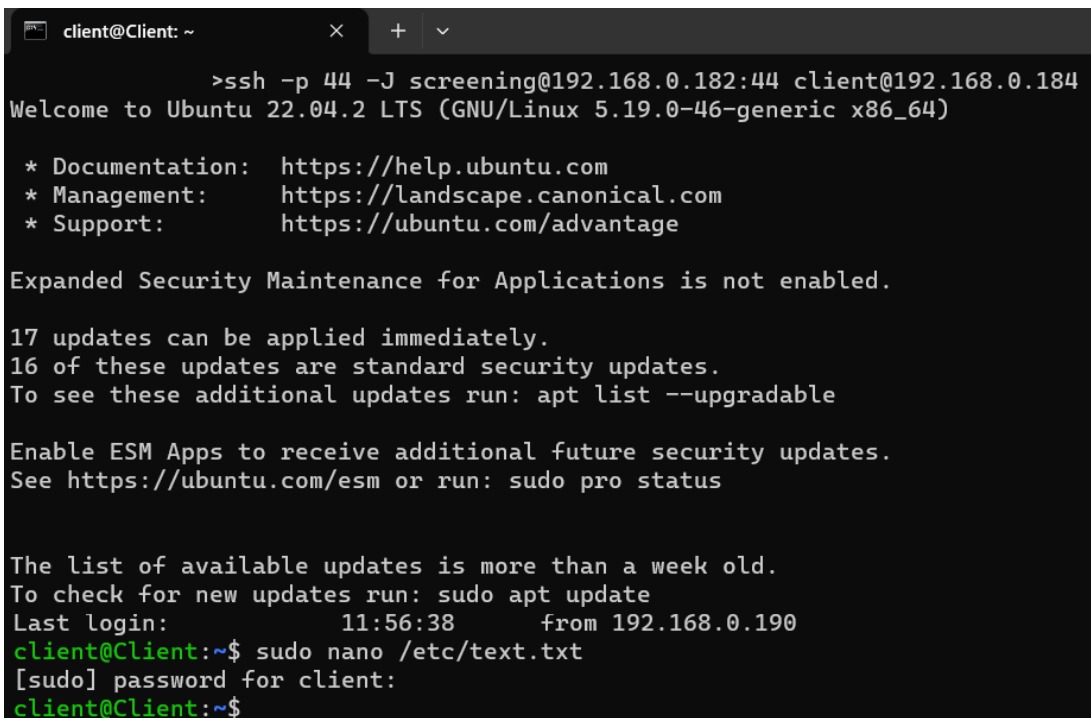
e da "Dual-Homed Host" a "Client":

```
$ ssh-copy-id -p 44 guest@192.168.0.184
```

Tramite SSH è possibile anche aggiungere, modificare e rimuovere file. A questo proposito, è importante che Tripwire registri queste modifiche. Si effettuerà un test per verificarne l'effettivo funzionamento. Come rappresentato in Figura 4.17, si effettua una connessione SSH all'account **client** della macchina "Client" (l'account con i permessi di root) e si esegue il comando:

```
$ sudo nano /etc/test.txt
```

che permette di creare il file di testo *test.txt* all'interno della cartella `/etc`.



```
client@Client: ~
>ssh -p 44 -J screening@192.168.0.182:44 client@192.168.0.184
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.19.0-46-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

17 updates can be applied immediately.
16 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

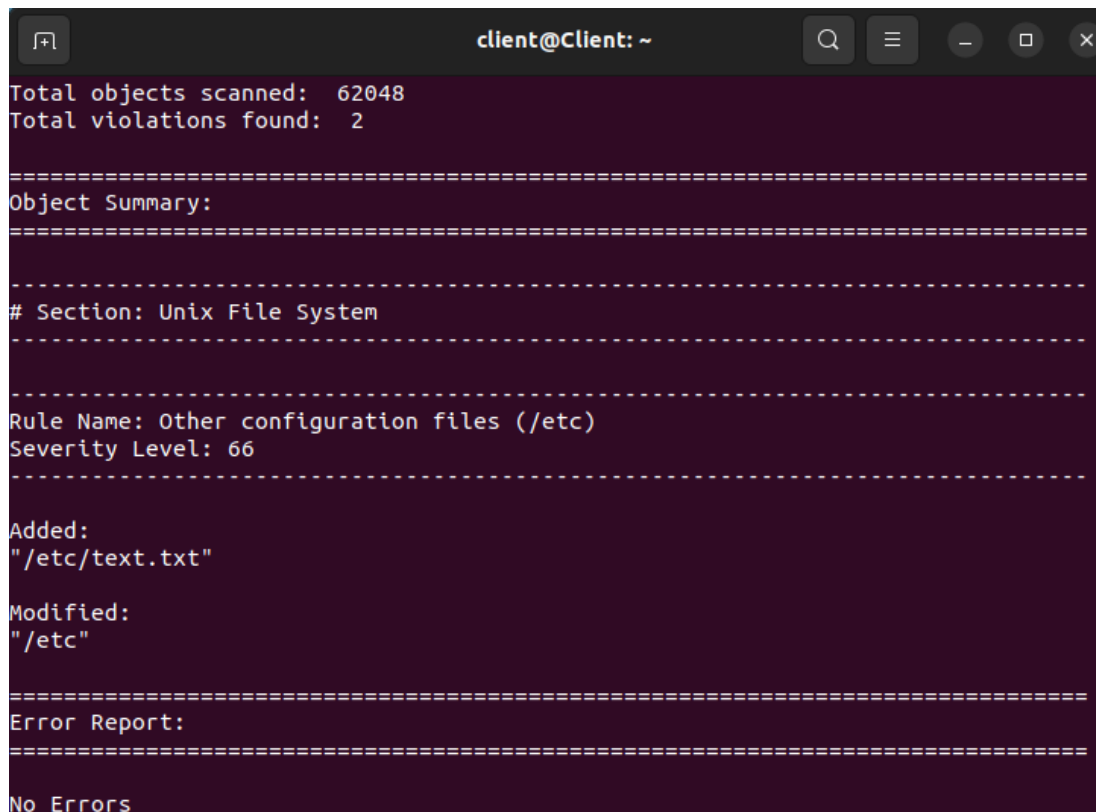
The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Last login:      11:56:38      from 192.168.0.190
client@Client:~$ sudo nano /etc/test.txt
[sudo] password for client:
client@Client:~$
```

Figura 4.17: Esempio di connessione SSH e creazione di un file di testo

A questo punto, eseguendo il comando:

```
$ sudo tripwire --check
```

per effettuare il controllo di integrità, il risultato è quello in Figura 4.18. Come si può notare, l'unica violazione riscontrata è quella relativa all'inserimento del file *test.txt* nella cartella `/etc`. Si può affermare che il test ha avuto esito positivo e che, quindi, Tripwire registra anche le modifiche ai file effettuate via SSH.

A terminal window titled 'client@Client: ~' with standard window controls. The output of a Tripwire integrity check is displayed. It shows that 62048 objects were scanned and 2 violations were found. The 'Object Summary' section is followed by a dashed line separator. The next section is '# Section: Unix File System', also followed by a dashed line. Below this, the 'Rule Name' is 'Other configuration files (/etc)' and the 'Severity Level' is '66'. The 'Added' files list includes '/etc/text.txt' and the 'Modified' files list includes '/etc'. Another dashed line separates this from the 'Error Report' section, which shows 'No Errors' at the bottom.

```
client@Client: ~
Total objects scanned: 62048
Total violations found: 2

=====
Object Summary:
=====

-----
# Section: Unix File System
-----

Rule Name: Other configuration files (/etc)
Severity Level: 66
-----

Added:
"/etc/text.txt"

Modified:
"/etc"

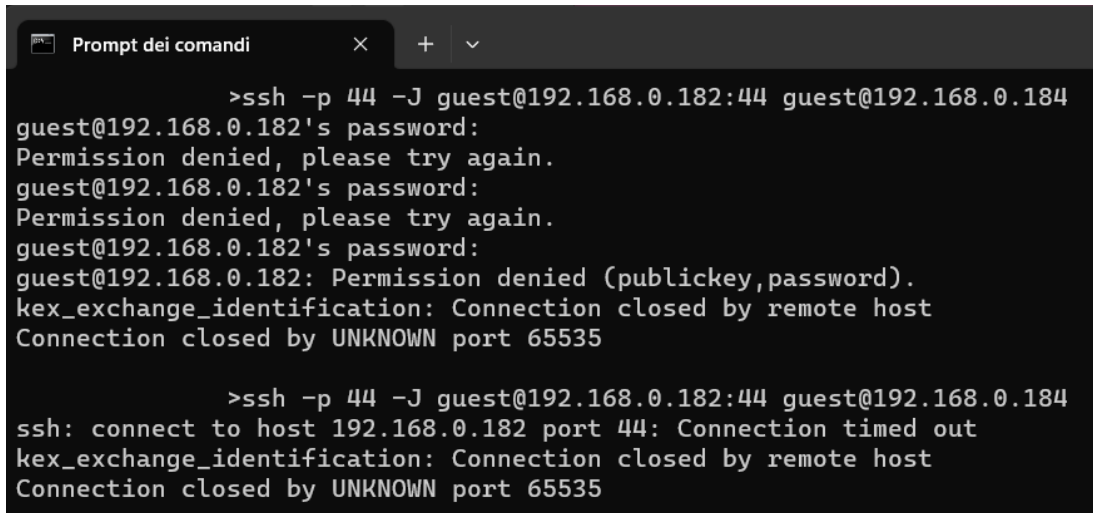
=====
Error Report:
=====

No Errors
```

Figura 4.18: Controllo di integrità di Tripwire in seguito al test effettuato

Fail2Ban

Il software Fail2Ban protegge il sistema da attacchi di tipo brute force. Infatti, un utente che si connette via SSH ad una delle due macchine dovrà inserire la password dell'account corrispondente. Se la password inserita è errata per 3 volte consecutive nell'arco di 2 minuti, l'IP da cui proviene la richiesta verrà bloccato per i successivi 2 minuti. In questo modo la probabilità di successo di un attacco di tipo brute force viene fortemente ridotta, perché computazionalmente troppo oneroso. In Figura 4.19 è rappresentato un esempio di ban dopo 3 tentativi errati di inserimento password. Come si può notare, il sistema interrompe la connessione e risponde con un messaggio di timeout alla successiva richiesta di connessione SSH, senza dare la possibilità di inserire la password. L'utente è infatti in una situazione di ban per i seguenti 2 minuti.



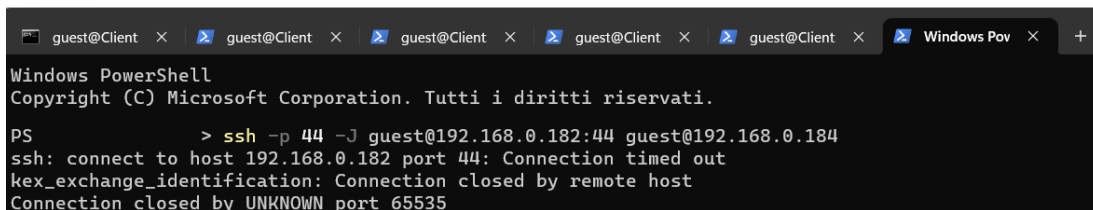
```
>ssh -p 44 -J guest@192.168.0.182:44 guest@192.168.0.184
guest@192.168.0.182's password:
Permission denied, please try again.
guest@192.168.0.182's password:
Permission denied, please try again.
guest@192.168.0.182's password:
guest@192.168.0.182: Permission denied (publickey,password).
kex_exchange_identification: Connection closed by remote host
Connection closed by UNKNOWN port 65535

>ssh -p 44 -J guest@192.168.0.182:44 guest@192.168.0.184
ssh: connect to host 192.168.0.182 port 44: Connection timed out
kex_exchange_identification: Connection closed by remote host
Connection closed by UNKNOWN port 65535
```

Figura 4.19: Esempio di ban dopo 3 tentativi errati di inserimento password

Test di resistenza del sistema ad attacchi di tipo DoS

Il firewall Iptables protegge il sistema da attacchi di tipo Denial of Services (DoS) attraverso la regola LIMIT IN presentata nella Sezione 3.2. Il numero massimo di connessioni SSH (sulla porta 44) che possono essere stabilite in un intervallo di tempo di 30 secondi è stato impostato pari a 5. Al sesto tentativo di connessione, si riceverà come risposta un messaggio di timeout. In Figura 4.20 è rappresentato proprio questo caso: si hanno 5 connessioni SSH aperte all'account guest della macchina "Client" (come si può notare dal nome delle schede del terminale); il sesto tentativo di connessione viene bloccato dal sistema e, come risposta, l'utente riceve un messaggio di timeout.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. Tutti i diritti riservati.

PS
> ssh -p 44 -J guest@192.168.0.182:44 guest@192.168.0.184
ssh: connect to host 192.168.0.182 port 44: Connection timed out
kex_exchange_identification: Connection closed by remote host
Connection closed by UNKNOWN port 65535
```

Figura 4.20: Esempio di violazione della regola LIMIT IN del firewall alla sesta connessione SSH

Capitolo 5

Conclusioni

In conclusione, la progettazione e l'implementazione di un'architettura di rete sicura basata su macchine virtuali, con l'utilizzo di un Dual-homed Host (il quale unisce le funzionalità di uno Screening Router e di un Bastion Host) e di una combinazione di strumenti avanzati come Iptables, Snort, Tripwire e Squid, rappresenta un passo fondamentale verso la protezione e la sicurezza delle risorse digitali di un'organizzazione.

Iptables permette di definire regole di filtraggio del traffico a livello di pacchetto, consentendo di autorizzare o bloccare specifiche connessioni in base alle esigenze di sicurezza.

Snort, come sistema di rilevamento delle intrusioni (IDS), sorveglia costantemente il traffico di rete alla ricerca di comportamenti sospetti o firme di attacchi noti, consentendo una risposta tempestiva alle minacce.

Tripwire offre un meccanismo affidabile per il monitoraggio dell'integrità dei file di sistema e delle configurazioni, rilevando qualsiasi modifica non autorizzata.

Squid fornisce un proxy web avanzato con funzionalità di caching e filtraggio, consentendo un controllo granulare sull'accesso a risorse esterne e la protezione dalle minacce web.

L'uso di macchine virtuali e l'implementazione di un Dual-homed Host consentono di suddividere la rete in segmenti isolati, riducendo la superficie di attacco. È essenziale monitorare costantemente il traffico di rete, mantenere aggiornati i sistemi e le firme di sicurezza e condurre revisioni regolari per garantire un ambiente resiliente alle minacce in continua evoluzione.

In definitiva, l'architettura proposta rappresenta un solido approccio alla sicurezza delle reti, garantendo l'integrità, la confidenzialità e la disponibilità delle risorse digitali. La sua corretta implementazione e manutenzione sono fondamentali per proteggere le attività e i dati critici dell'organizzazione in un mondo digitale sempre più complesso e pericoloso.

Elenco delle figure

| | | |
|------|---|----|
| 1.1 | Schema dell'infrastruttura sicura | 3 |
| 2.1 | Struttura di un'architettura di tipo Dual-homed Gateway | 6 |
| 3.1 | Esempio di una scheda di Gufw con alcune regole attive | 10 |
| 3.2 | Richiesta di credenziali di accesso da parte di Squid | 12 |
| 3.3 | Lista dei domini web in lista nera | 12 |
| 3.4 | Esempio di un controllo di integrità di Tripwire con nessuna violazione riscontrata | 14 |
| 3.5 | Policy di default "deny all" | 15 |
| 3.6 | Esempio di regole per il filtraggio del traffico | 17 |
| 3.7 | Configurazione di Fail2Ban | 18 |
| 4.1 | Estratto di tutte le regole di Iptables implementate | 21 |
| 4.2 | Esempio elenco delle regole implementate con Ufw | 22 |
| 4.3 | Esempio elenco delle regole implementate visualizzate con Gufw | 23 |
| 4.4 | Esempio di aggiunta di una nuova regola con Gufw | 24 |
| 4.5 | Esempio di una violazione riscontrata da Snort sull'interfaccia esterna | 25 |
| 4.6 | Esempio di una violazione riscontrata da Snort sull'interfaccia interna | 25 |
| 4.7 | Esempio di una local rule di Snort | 25 |
| 4.8 | Richiesta di credenziali all'apertura del browser | 26 |
| 4.9 | Aggiunta di una nuova coppia di credenziali | 26 |
| 4.10 | Esempio di una richiesta per uno dei domini in lista nera | 27 |
| 4.11 | Esempio di curl senza l'inserimento di credenziali e relativo errore | 27 |
| 4.12 | Esempio di curl con inserimento di credenziali valide per l'utente "univpm" | 28 |
| 4.13 | Esempio di un controllo di integrità di Tripwire con alcune violazioni riscontrate | 29 |
| 4.14 | Dettagli delle violazioni riscontrate da Tripwire | 30 |
| 4.15 | Esempio di connessione SSH | 32 |

| | |
|---|----|
| 4.16 Esempio di connessione SSH con l'utilizzo delle chiavi da macchina Windows | 33 |
| 4.17 Esempio di connessione SSH e creazione di un file di testo | 34 |
| 4.18 Controllo di integrità di Tripwire in seguito al test effettuato . . . | 35 |
| 4.19 Esempio di ban dopo 3 tentativi errati di inserimento password . . | 36 |
| 4.20 Esempio di violazione della regola LIMIT IN del firewall alla sesta connessione SSH | 36 |