Progettazione e sviluppo di un sistema di rilevamento di attacchi zero-day

Tesi di Laurea Magistrale in Ingegneria Informatica

Marco D'Aleo

Relatore: Prof.ssa Alessandra De Paola

Correlatori: Ing. Vincenzo Agate
Ing. Pierluca Ferraro

UNIVERSITÀ DEGLI STUDI DI PALERMO
FACOLTÀ DI INGEGNERIA

LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

PROGETTAZIONE E SVILUPPO DI UN SISTEMA DI RILEVAMENTO DI ATTACCHI ZERO-DAY

Tesi di Laurea di
Marco D'Aleo

Relatore:
Prof.ssa Alessandra De Paola

Correlatore:
Ing. Vincenzo Agate
Ing. Pierluca Ferraro

Sommario

Il panorama delle minacce informatiche è caratterizzato dalla crescente frequenza e complessità degli attacchi, con particolare rilevanza per gli attacchi zero-day. Gli attacchi zero-day sfruttano vulnerabilità software sconosciute, rendendo inefficaci le soluzioni di sicurezza tradizionali. In questo contesto, l'importanza di sviluppare sistemi in grado di identificare tempestivamente attacchi sconosciuti diventa cruciale.

Questa tesi presenta la progettazione, l'implementazione e la valutazione di un sistema di rilevamento delle intrusioni (IDS) con la capacità di rilevare sia attacchi zero-day che attacchi noti. Si introduce il fenomeno degli attacchi zero-day e la necessità di sistemi di rilevamento delle intrusioni in grado di affrontare queste minacce in continua evoluzione. L'efficacia del sistema viene valutata utilizzando il dataset CIC-IDS2017, un benchmark ampiamente riconosciuto nel campo della sicurezza informatica. L'obiettivo principale è migliorare la sicurezza complessiva delle reti fornendo una difesa robusta contro le minacce emergenti.

L'IDS proposto sfrutta una combinazione di tecniche per il rilevamento di attacchi noti e tecniche basate su anomalie per affrontare le limitazioni associate ai sistemi tradizionali. Il rilevamento di attacchi conosciuti si basa su un database completo di modelli di attacco noti, mentre il rilevamento basato su anomalie mira ad identificare deviazioni dal normale comportamento di rete. Integrando questi approcci, il sistema mira a ottenere un tasso di rilevamento più elevato sia per attacchi conosciuti che per quelli mai visti in precedenza.

Il processo di valutazione prevede il test dell'IDS sul dataset CICIDS2017, che comprende una vasta gamma di scenari di traffico di rete. Il dataset è adatto per valutare la capacità del sistema di identificare e categorizzare con precisione gli attacchi. Metriche di prestazione come la *precision* e il *recall* vengono misurate per fornire una valutazione completa. I risultati dimostrano l'efficacia del sistema nel rilevare attacchi noti, evidenziando la sua affidabilità nell'identificare minacce ben consolidate. Inoltre, l'IDS mostra prestazioni promettenti nel riconoscere attacchi zero-day, evidenziando il suo potenziale nel mitigare minacce precedentemente sconosciute. La tesi si conclude con riflessioni sui punti di forza, le limitazioni e le possibilità per lavori futuri per migliorare ulteriormente le sue capacità nel dinamico panorama della sicurezza informatica.

Indice

Introduzione

La cybersecurity è definita come la salvaguardia di dati, reti e sistemi informatici. L'obiettivo è quello di difendere e proteggere le attività di un'organizzazione svolte nel cyber-spazio da qualsiasi attacco informatico volto a corrompere, distruggere, disattivare o prendere il controllo di un sistema informatico, nonché a compromettere l'integrità dei dati o a sottrarre informazioni riservate [1]. Se da un lato il rapido progresso della tecnologia ha offerto a ogni organizzazione enormi opportunità, dall'altro ha portato con sé rischi senza precedenti, che potrebbero rappresentare un serio problema per tutte le organizzazioni. Al giorno d'oggi, implementare misure di sicurezza efficaci risulta essere particolarmente complicato poiché vi sono più dispositivi che persone e gli attacchi stanno diventando sempre più sofisticati. Particolare rilevanza è assunta dai dati di cui si vuole proteggere la confidenzialità, l'integrità e la disponibilità [2]:

- **confidenzialità**: si vuole impedire la divulgazione di informazioni a individui o sistemi non autorizzati;
- **integrità**: si vuole impedire la modifica o la cancellazione di dati tramite accessi non autorizzati;
- **disponibilità**: si vuole garantire che i sistemi che forniscono servizi come l'archiviazione o l'elaborazione delle informazioni siano sempre accessibili da chi ne ha bisogno.

L'enorme quantità di dati generati e che viaggiano su Internet costituisce un obiettivo appetibile per gli attaccanti informatici che utilizzano strumenti e tecniche sempre più avanzate per ottenere il controllo di dispositivi dotati di sensori che possono essere usati per lanciare attacchi massicci e distribuiti contro aree critiche a livello nazionale, infrastrutture e siti web. I criminali informatici possono utilizzare la stessa tecnologia per infiltrarsi in reti pubbliche o private per impiantare malware in pochi minuti. Cisco ha riferito nel suo rapporto annuale del 2020 [3] che la crescita delle violazioni e degli attacchi informatici è aumentata del 776% a livello globale; entro il

2023, il numero di attacchi Distributed Denial of Service (DDoS) raggiungerà i 15,4 milioni (il doppio rispetto al 2018, quando erano 7,9 milioni). Tali attacchi distribuiti su larga scala comportano perdite aziendali, guasti ad apparecchiature industriali e mettono a rischio la vita delle persone. Nel 2016 si è verificato un noto attacco DDoS, "Mirai", contro siti web noti come Netflix, Twitter, GitHub e New York Times, che li ha resi non disponibili. Mirai ha lanciato un attacco DDoS da 665 Gbps che ha infettato oltre 2,5 milioni di dispositivi IoT. Le organizzazioni devono quindi utilizzare delle contromisure che implementino delle misure di sicurezza efficaci per proteggere le proprie risorse. Tra queste contromisure troviamo gli Intrusion Detection System (IDS) [4].

Un Intrusion Detection System (IDS) robusto è un componente chiave all'interno di qualsiasi rete. Una realizzazione efficace di tale sistema richiede il posizionamento strategico all'interno della rete per analizzare tutto il traffico, una rapida capacità di elaborazione e di rilevamento di anomalie presenti nel traffico e la comunicazione delle informazioni sull'intrusione rilevata per decidere come intervenire per limitare i danni. La rapida evoluzione della tecnologia spinge gli attaccanti a sperimentare nuove e avanzate tipologie di attacco e ciò rende difficile la realizzazione di una soluzione che sia in grado di proteggere la rete sia dagli attacchi conosciuti sia da quelli sconosciuti che prendono il nome di attacchi zero-day.

Gli attacchi condotti possono essere mirati verso diversi obiettivi:

- **attacchi verso la rete**: si tratta di attacchi mirati a ledere il funzionamento di rete e sono inizializzati attraverso un flusso di pacchetti trasmessi attraverso la rete. L'attacco più famoso appartenente a questo categoria è l'attacco di *denial of service*, DoS, e la sua variante distribuita, DDoS. Quando un attacco DoS viene eseguito, gli utenti legittimi non riescono ad accedere ad un particolare servizio. Questo accade perché l'attaccante inonda il servizio di richieste rendendo quest'ultimo incapace di soddisfare le richieste degli utenti legittimi.

- **attacchi verso gli host**: in questo caso l'obiettivo dell'attacco è un host. L'attacco viene realizzato attraverso l'esecuzione di software malevolo che va a compromettere le funzionalità di sistema. Tra gli attacchi di questa categoria è possibile osservare i malware, i worms, i virus, gli adware e gli spyware. Due malware particolarmente rilevanti sono i Trojan e i Ransomware. I Trojan permettono ad un attaccante di prendere il controllo di un dispositivo. La loro peculiarità è quella di apparire come applicazioni fidate ingannando di fatto l'utente che ignaro di tutto eseguirà l'applicazione infetta. I Ransomware cifrano

i file di un utente finché quest'ultimo non paga un riscatto. L'attaccante detiene quindi il controllo dei file fino alla ricezione del riscatto dopo il quale rilascia la chiave di cifratura per il recupero dei file.

- **attacchi verso il software**: attraverso questa tipologia di attacco un attaccante riesce ad iniettare del codice malevolo che influenza il flusso di esecuzione di un programma. A questa categoria appartengono gli attacchi di SQL injection e gli attacchi di Cross Site Scripting (XSS). Se eseguita correttamente, una SQL injection permette ad un attaccante di interrogare un database con una query malevola con lo scopo di ottenere dati confidenziali o intaccare l'integrità del database, eliminando righe, colonne o intere tabelle. Gli attacchi XSS sfruttano anch'essi l'iniezione di codice per eseguire codice malevolo in applicazioni web;

- **attacchi verso risorse fisiche**: questa categoria include gli attacchi mirati al danneggiamento di hardware di rete. Questo può essere fatto arrecando del danno fisico ai dispositivi di rete giacché quest'ultimi sono fisicamente esposti.

L'enorme volume del traffico e la velocità con cui esso viene generato rendono impegnativa la rilevazione di intrusioni in tempo reale. Per questa ragione i ricercatori hanno proposto l'impiego di tecniche di Machine Learning (ML) e Deep Learning (DL) nella creazione di sistemi in grado di difendere la rete. Tuttavia, i sistemi tradizionali basati su modelli di Machine Learning tendono a ottenere buoni risultati solamente in presenza di attacchi conosciuti e si osserva un degradamento delle performance in presenza di attacchi zero-day. Infatti, assumendo di avere un modello addestrato e che offre delle ottime prestazioni, quest'ultimo non saprebbe come classificare le istanze appartenenti a categorie di attacco sconosciute, restituendo una classificazione errata e causando di conseguenza una notevole riduzione dell'accuratezza. Migliaia di nuovi attacchi emergono ogni anno e si necessita, quindi, di un Intrusion Detection System in grado di predire con accuratezza sia gli attacchi conosciuti sia gli attacchi che il modello non hai incontrato prima. La realizzazione di un sistema altamente accurato richiede tecniche che differiscono dalle usuali metodologie di addestramento. Molti studi di ricerca sugli IDS utilizzano l'assunzione di mondo chiuso trascurando totalmente la gestione degli attacchi zero-day.

Uno scenario tipico nei problemi di Machine Learning è quello in cui tutte le classi presenti nel dataset vengono utilizzate durante la fase di addestramento e di test del modello. Questo approccio prende il nome di classificazione closed-set ed è quello che sfrutta l'assunzione di

mondo chiuso. Lo scenario opposto è quello in cui i modelli di Machine Learning vengono addestrati con una ipotesi di base più realistica, cioè che l'informazione ad essi fornita è limitata. Questo approccio è quello che viene utilizzato nei problemi di Open-Set Recognition (OSR) e costituisce una classe di problemi più complessi. In questi casi il modello viene valutato anche su istanze appartenenti a classi non presenti nell'insieme di addestramento e che il modello non ha mai visto prima [5]. A causa della difficoltà nell'individuare le classi sconosciute, il classificatore deve essere robusto abbastanza da saper rigettare i nuovi campioni che differiscono da quelli visti durante la fase di addestramento. Dal punto di vista di un classificatore un attacco zero-day può essere definito formalmente come traffico di rete che differisce significativamente dal normale traffico e che non è una variante di uno degli attacchi su cui il modello è stato precedentemente addestrato. In base a quanto detto risulta chiaro che gli approcci comuni per la realizzazione di IDS come i misuse-based IDS e gli anomaly-based IDS sono limitati nella rilevazione di attacchi zero-day. Nonostante i misuse-based IDS producano un basso numero di falsi positivi, non sono in grado di rilevare nuove tipologie di attacco. D'altro canto, gli IDS basati sull'anomalia risultano essere dei buoni candidati alla risoluzione del problema ma producono un alto numero di falsi positivi e negativi. La generazione di falsi positivi richiede una supervisione al fine di stabilire la motivazione per cui l'allarme è stato generato.

Con lo sviluppo tecnologico, nuovi attacchi mirati stanno nascendo e ciò rende la rilevazione di attacchi zero-day un compito arduo. Gli studi proposti dallo stato dell'arte sfruttano approcci basati su tecniche di Machine Learning e Deep Learning. Tuttavia, gli attaccanti possono impiegare strumenti avanzati per generare traffico malevolo in grado simulare e confondersi con il traffico normale, eludendo così i convenzionali IDS. Inoltre, per motivi di privacy, è anche difficile trovare un dataset reale e aggiornato che permetta di valutare questi sistemi. Il traffico di rete genera una quantità di dati massiccia. Il processo di etichettatura di un volume così grande di dati è un compito complesso e ciò costituisce un ostacolo nell'ottenere un dataset etichettato. Quando si sfruttano tecniche di Machine Learning non supervisionate, un dataset etichettato non è necessario ma ciò porta gli studiosi ad effettuare delle assunzioni che spesso si traducono nella generazione di falsi positivi. Un altro problema con i dataset etichettati è che l'operazione di etichettatura potrebbe non essere stata effettuata in maniera accurata. Al momento non vi sono metodologie scientifiche che permettono di validare l'etichettatura di un dataset. Naturalmente la credibilità del dataset viene messa in discussione se l'etichettatura non è accurata, se il dataset contiene dati irrilevanti e se il dataset non è sufficientemente diversificato. Per queste ragioni spesso gli studiosi utilizzano dataset piccoli e simulati in un ambiente controllato. Chiaramente

questi dataset dipendono dall'ambiente e potrebbero non fornire uno scenario completo di traffico di rete. Sulla base di quanto detto, la mancanza di disponibilità di un dataset che sia affidabile e realistico rappresenta una sfida nella realizzazione di IDS. Per mitigare questo problema e poter comunque riuscire a trovare una soluzione a problemi che richiedono la rilevazione di istanze sconosciute, gli studiosi ricorrono all'addestramento di modelli sul solo traffico normale. Questo espediente si basa sulla evidenza che è più semplice ottenere dati sul traffico normale piuttosto che ottenere dati sul traffico malevolo. Tuttavia, in questi casi vengono utilizzati per lo più sistemi che sfruttano soglie e che quindi effettuano una semplice classificazione binaria rigettando tutto ciò che differisce dal modello di traffico normale appreso. Questi sistemi, però, costituiscono una soluzione generale al problema che non fornisce informazioni dettagliate sull'attacco che si verifica. Queste informazioni aggiuntive potrebbero essere necessarie a un professionista che si occupa di intraprendere la corretta soluzione per mitigare i danni subiti a seguito dell'attacco giacché diverse tipologie di attacco necessitano diversi interventi correttivi.

Un'altra sfida che si può evidenziare è quella legata alla complessità del sistema. A causa del continuo emergere di nuove varianti di attacco, i modelli alla base di questi sistemi necessitano di essere periodicamente riaddestrati. Un modello che richiede giorni o settimane per essere addestrato non costituisce una soluzione che si adatta ad un contesto reale, specialmente in un ambiente a risorse limitate come quello IoT. Quindi l'abilità di riaddestrare velocemente un IDS ne aumenterebbe notevolmente le performance. Considerando ad esempio i sistemi basati su tecniche di Deep Learning, notiamo che questi ultimi sono efficaci nella gestione di grandi dataset ma sono particolarmente onerosi computazionalmente e dal punto di vista del consumo energetico. Quindi è auspicabile che un IDS sia non solo robusto ed efficace ma anche leggero.

Gli IDS rappresentano una vasta area di ricerca e sono oggetto di studio grazie alla loro caratteristica di costituire una soluzione a diverse sfide in ambito di sicurezza informatica. Tra i campi di applicazione degli IDS è possibile notare [6]:

- **phishing**: è una violazione di sicurezza attraverso la quale un attaccante cerca di ottenere le informazioni personali o le credenziali di un utente. In genere la vittima viene indotta a cliccare su un link ricevuto tramite un canale di comunicazione come ad esempio la posta elettronica. Un IDS potrebbe rilevare un attacco di phishing andando ad analizzare le feature che è possibile estrarre dal link usato per condurre l'attacco. Ad esempio, è possibile utilizzare feature come l'URL, il dominio e il contenuto della pagina web a cui conduce il link utilizzato per eseguire l'attacco [7].

- **informatica forense**: si tratta di una branca della scienza forense che si occupa di studiare i reperti digitali. Le informazioni contenute all'interno di dispositivi, come computer e smartphone, sono rilevanti per l'informatica forense. Poiché un IDS registra le attività che si verificano all'interno della rete, registrerà anche gli eventi che celano un attacco e l'istante temporale in cui questi vengono effettuati. Pertanto, si può utilizzare un IDS per ottenere delle prove digitali. Tali prove potranno poi essere utilizzate durante un procedimento legale contro l'attaccante.

- **sicurezza dei sistemi IoT**: i dispositivi IoT sono dotati di sensori e hanno la capacità di comunicare con gli altri dispositivi connessi alla rete. A causa della natura pervasiva di questi dispositivi, si sta assistendo ad un rapido sviluppo e applicazione del mondo IoT in diversi domini. La protezione dei dispositivi IoT dagli attacchi è una sfida a causa dell'eterogeneità degli stessi e della vastità dei protocolli utilizzati [8]. Anche nei contesti IoT, quindi, gli IDS giocano un ruolo fondamentale.

- **violazioni di sicurezza**: i dispositivi e i sistemi usati dagli utenti tendono ad avere delle vulnerabilità. Queste vulnerabilità possono essere sfruttate dagli avversari per causare del danno. Nel peggiore dei casi, il danno non viene arrecato solo ad un singolo utente ma all'intera organizzazione di cui esso fa parte. Di conseguenza vi è una forte richiesta per lo sviluppo di IDS che possano affrontare tali vulnerabilità e che possano anche aiutare a scoprirne di nuove con l'obiettivo finale di proteggere il sistema da attacchi informatici.

- **rilevazione dello spam**: un messaggio di spam può essere definito come un messaggio non desiderato ricevuto da un utente. Un attaccante può inserire un link malevolo all'interno di un messaggio di spam, che se cliccato può compromettere le informazioni sensibili della vittima. Anche in questo caso gli IDS possono correre in soccorso effettuando una classificazione binaria del messaggio per stabilire se il contenuto è legittimo o presenta potenziali minacce.

Allo stato attuale, manca un sistema efficace che blocchi qualsiasi tipo di traffico di natura dannosa e che non abbia un impatto significativo sull'esperienza d'uso degli utenti legittimi. Il lavoro svolto si è quindi concentrato innanzitutto nell'individuare le odierne metodologie adottate in questo ambito e le criticità che da esse derivano. Per fare ciò si è cercato di capire cosa siano gli attacchi zero-day, come simularli e in che modo questi possano rappresentare un problema e una debolezza per le soluzioni proposte dallo stato dell'arte. Successivamente,

tenuto conto delle criticità e delle sfide precedentemente citate, ci si è dedicati alla progettazione e allo sviluppo di un sistema che analizzi il traffico e sia in grado non solo di rilevare gli attacchi conosciuti ma anche quelli sconosciuti. Si vuole inoltre realizzare un sistema che sia robusto abbastanza da rilevare una moltitudine di potenziali attacchi zero-day che differiscano dalle tipologie di traffico che il sistema già conosce. È anche auspicabile che il sistema sia affidabile generando un basso numero di falsi allarmi.

Gli obiettivi progettuali precedentemente elencati hanno portato alla realizzazione di un IDS strutturato su due livelli che offre una solida difesa contro attacchi noti e zero-day. Il primo livello si avvale di una combinazione di modelli e adotta un approccio basato su soglia per individuare le anomalie, concentrandosi specificamente sugli attacchi zero-day. Questo strato iniziale funge da filtro e svolge una classificazione parziale dell'input in tre categorie: benigno, attacco noto o attacco zero-day. Questa fase è fondamentale per apprendere accuratamente le caratteristiche del traffico normale e degli attacchi noti, contribuendo così all'efficace identificazione degli attacchi zero-day.

Le istanze etichettate come attacchi noti vengono inoltrate al secondo livello, il quale impiega un ensemble di classificatori. Questo livello aggiuntivo si occupa di una classificazione più dettagliata, ottimizzando le prestazioni grazie alla notevole riduzione del numero di campioni da analizzare. L'utilizzo di un ensemble di classificatori mira a migliorare la precisione complessiva del sistema, evitando la dipendenza eccessiva da un singolo modello.

L'obiettivo del sistema è garantire non solo una rilevazione efficiente ma anche precisa di varie tipologie di attacchi informatici. L'approccio gerarchico su più livelli consente una gestione ottimale del traffico, migliorando l'efficienza complessiva del sistema e contribuendo a una difesa affidabile.

La tesi è suddivisa in capitoli organizzati come segue:

- il capitolo 1 presenta una panoramica sugli Intrusion Detection System. In particolare, approfondisce le tipologie di IDS e le relative caratteristiche e debolezze. Vengono inoltre mostrati i principali approcci impiegati dallo stato dell'arte per la rilevazione di attacchi zero-day e le loro criticità.
- il capitolo 2 introduce il sistema proposto. Vengono esposte le nozioni necessarie e le considerazioni sulle quali si è basata la progettazione e lo sviluppo del sistema. In seguito,

vengono descritti i singoli componenti del sistema e come questi fanno parte del sistema complessivo.

- il capitolo 3 illustra la valutazione sperimentale. Viene introdotto il dataset utilizzato, le problematiche da questo riportate, le soluzioni intraprese per mitigare queste problematiche e le elaborazioni effettuate. Successivamente vengono presentate le metriche con la quale viene valutato il sistema e i risultati sperimentali ottenuti, effettuando inoltre un confronto con uno studio pertinente nell'ambito del rilevamento di attacchi zero-day.
- il capitolo sulle conclusioni, infine, espone le considerazioni finali. Si riassumono i punti cardine del lavoro svolto e di come questi abbiano permesso il raggiungimento degli obiettivi preposti e si propongono dei miglioramenti al sistema per possibili lavori futuri.

Capitolo 1

Stato dell'arte

Gli Intrusion Detection System (IDS) sono sistemi il cui compito principale è quello di analizzare il traffico di rete con l'obiettivo di rilevare anomalie e intrusioni. Un'anomalia è definita come un'osservazione che devia al punto da destare sospetto [10]. Se non correttamente individuate e gestite, le intrusioni possono andare a violare una o più delle proprietà di confidenzialità, integrità e disponibilità di una risorsa all'interno della rete. Tali risorse possono essere soggette a vulnerabilità e un attaccante può sfruttare tali debolezze, portando ad una violazione indesiderata della sicurezza. Un attacco è detto attivo se l'attaccante tenta di modificare o influenzare il funzionamento del sistema. Un attacco passivo invece, è un attacco in cui l'attaccante cerca di apprendere informazioni sul sistema ma senza influenzare quest'ultimo. Gli attacchi possono anche avere una diversa origine: gli attacchi interni sono quegli attacchi lanciati da un'entità all'interno della rete; viceversa, gli attacchi esterni sono attacchi lanciati da un utente illegittimo del sistema e di conseguenza all'esterno del perimetro di rete.

Le funzionalità di un IDS sono molteplici. Innanzitutto, un IDS deve monitorare il traffico per estrapolare le informazioni di rete contenute nei pacchetti. Sia l'header che il payload del pacchetto possono contenere informazioni utili per rilevare la presenza di un attacco. Successivamente vengono analizzate le informazioni estratte per individuare anomalie o pattern di attacchi già conosciuti. Una volta riconosciuto un pattern, un allarme può essere sollevato per richiamare l'attenzione dell'amministratore di rete. Sarà poi lui a decidere la corretta azione da intraprendere in base all'alert ricevuto.

In base alla sorgente di informazione utilizzata, è possibile classificare gli IDS come host-based IDS e network-based IDS. Un host-based IDS viene installato su uno specifico host.

Vengono quindi monitorate le attività di un singolo host prestando attenzione ad eventi che possano essere sintomo di attività sospette. Questa particolare tipologia di IDS permettere di determinare con esattezza quali processi e utenti sono coinvolti in un attacco. Le attività possono essere monitorate attraverso log di sistema o log di applicazioni facilmente ottenibili dal sistema operativo. Quindi log relativi ad errori di segmentation fault, crash di sistema o accessi non autorizzati possono essere indicatori di un'attività malevola. Un network-based IDS viene installato per monitorare il traffico in entrata e in uscita da un'intera rete o per un particolare segmento di rete. Dispositivi e traffico di rete vengono monitorati alla ricerca di eventi sospetti. Come si può facilmente immaginare, la quantità di traffico da vagliare è enorme ed è probabile che più attività generino degli eventi che indichino la presenza di intrusioni.

Un'ulteriore classificazione per gli IDS si basa sulla strategia utilizzata per il rilevamento delle intrusioni. In questo caso si parla di misuse-based IDS e anomaly-based IDS. Verranno ora approfondite queste due tipologie di sistemi di rilevamento delle intrusioni

1.1 Misuse-based IDS

Questi IDS vengono generalmente progettati basandosi su un database contenente le firme di attacchi conosciuti, ovvero pattern che identificano univocamente l'attacco. Per tale ragione, gli IDS appartenenti a questa categoria sono anche conosciuti come IDS basati su firme. I signature-based IDS, quindi, vanno alla ricerca di pattern nel traffico di rete che combaciano con le firme presenti nel database. I pattern individuati possono poi essere esaminati per verificare la presenza di comportamenti anomali. Questo li rende particolarmente adatti alla rilevazione di attacchi conosciuti, ottenendo buone prestazioni in termini di un basso numero di falsi positivi generati.

1.1.1 Classificatori singoli

La maggior parte dei sistemi proposti appartenenti a questa categoria si avvale di tecniche di Machine Learning come la classificazione. La classificazione è una tecnica che sfrutta un dataset dotato di etichette, le quali rappresentano le classi di appartenenza delle istanze del dataset. Un classificatore assocerà un input ad una delle classi conosciute. Di conseguenza, questa tecnica ben si presta come soluzione al problema.

Alcuni degli algoritmi maggiormente utilizzati sono i seguenti:

- **decision tree (DT)**: viene costruita una struttura ad albero, composta da nodi interni e nodi foglia, attraverso una strategia greedy. Ogni nodo interno viene espanso per generare i nodi successori in base ad un determinato criterio, come ad esempio l'entropia o il coefficiente di Gini. Il processo continua finché non si raggiungono dei nodi per cui non è più possibile generare successori ovvero i nodi foglia che rappresentano l'output del modello. L'obiettivo è quello di ridurre al minimo la diversificazione di istanze appartenenti a classi diverse all'interno delle singole foglie. In altre parole, si vuole che una particolare foglia rappresenti una classe. Le nuove istanze verranno quindi classificate attraverso una sequenza di decisioni. I DT sono una scelta popolare poiché sono modelli semplici e di facile implementazione [11]. Questo modello viene spesso affiancato da un'altra tecnica che migliorarne le performance. È stato realizzato un IDS che raggiunge elevati livelli di accuratezza basato su alberi decisionali e sulla definizione di un set di regole [12].

- **naive Bayes**: questo classificatore si basa sul teorema di Bayes per risolvere il problema di classificazione e si avvale delle probabilità a priori e a posteriori delle istanze del dataset. Il teorema di Bayes afferma che:

$$p(C_k|x) = \frac{p(x|C_k)p(C_k)}{p(x)} \tag{1.1}$$

  dove x è il campione in input, k è il numero totale di classi e C_k è la k-esima classe per cui si vuole calcolare la probabilità a posteriori. Il modello assume che le feature del dataset sono indipendenti (anche se questo non è sempre vero; da questo il nome *naive*) e calcola le probabilità condizionali che ciascun campione del dataset appartenga ad una determinata classe. I nuovi campioni in input verranno classificati come appartenenti alla classe per la quale si è ottenuta la più alta probabilità a posteriori. Uno degli svantaggi di questo modello è la conoscenza necessaria per calcolare le probabilità a priori. Tali probabilità dipendono dal numero di istanze per classe e dal numero di feature, di conseguenza il calcolo delle probabilità può diventare impegnativo al crescere della dimensione del dataset.

- **support vectori machine (SVM)**: è una tecnica che separa i dati in due regioni attraverso un iperpiano. L'addestramento del modello cerca di individuare tale iperpiano massimizzando il margine tra i punti nello spazio e l'iperpiano stesso. Le SVM vengono impiegate anche nel caso in cui si ha a che fare con dati non linearmente separabili poiché è possibile

utilizzare il cosiddetto *kernel trick*, ovvero una funzione che mappa i dati in input in uno spazio a dimensionalità maggiore dove potrebbero essere linearmente separabili. Affinché il classificatore riesca ad essere accurato è necessario che il dataset contenga istanze che siano sufficientemente rappresentative dei dati che si vogliono modellare. È possibile migliorare il funzionamento di una SVM usando tecniche di selezione delle feature come, ad esempio, la Principal Component Analysis (PCA). Sono state confrontate le performance del modello con e senza PCA e si è notato un incremento dell'accuratezza nel caso in cui la selezione delle feature è stata impiegata [13].

- **clustering**: si tratta di un approccio non supervisionato che prevede il raggruppamento dei dati in input sulla base della loro similarità. Il clustering è una tecnica semplice che crea dei cluster omogenei di dati. Attraverso un approccio iterativo che studia più volte i dati, le tecniche di clustering riescono a raffinare la purezza del cluster ottenendo così dei risultati migliori. Uno dei modelli più popolari è il K-means in cui K rappresenta il numero di centroidi. Essendo un centroide il centro del cluster, K rappresenta il numero di cluster. L'addestramento prevede di ridurre al minimo la somma delle distanze tra i punti nello spazio e il centroide del cluster di appartenenza. Il K-means è soggetto al problema dell'assegnamento forzato. Questo problema si verifica quando il valore di k non è scelto in maniera corretta ed è minore rispetto al numero di cluster "richiesti" dal dataset. Sottodimensionando k, istanze aventi caratteristiche diverse e appartenenti a gruppi diversi finiranno per far parte dello stesso cluster. È stata proposta una soluzione a questo problema attraverso la costruzione di un decision tree su ogni cluster che va a risolvere i conflitti interni comportando una riduzione di falsi positivi [14].

Quelli appena elencati sono classificatori singoli ovvero tecniche che sfruttano un solo algoritmo di Machine Learning. Un IDS realizzato esclusivamente tramite un classificatore singolo gioverà dei vantaggi che l'algoritmo offre ma sarà anche soggetto agli svantaggi di cui esso è affetto. Si è visto dunque che la tendenza è quella di affiancare più classificatori e ciò ha permesso un miglioramento delle performance poiché è possibile sfruttare i pregi di diversi algoritmi e mitigare i difetti di ciascuno di essi. Si vengono così a delineare sistemi più complessi come i classificatori ibridi cioè classificatori composti da una combinazione di componenti che permettono di approcciare il problema sotto punti di vista diversi e di fornire una soluzione più accurata.

Come precedentemente accennato, gli IDS basati su firme sono precisi nella rilevazione di attacchi conosciuti ma non si prestano molto bene nell'individuare gli attacchi zero-day, a differenza degli IDS basati sulle anomalie che agiscono rilevando i comportamenti anomali che deviano dal comportamento normale. Si potrebbe quindi creare un sistema ibrido che sfrutti entrambe le tipologie di sistemi per rilevare sia gli attacchi conosciuti sia gli attacchi sconosciuti, avendo un sistema finale più completo.

Considerato quanto osservato circa le capacità dei classificatori singoli e di come questi possano essere migliorati combinando fra loro più modelli si ritiene necessario approfondire tali tematiche così da avere una panoramica più ampia in fase di progettazione del sistema proposto. Una tecnica molto popolare per combinare i modelli di classificazione è quella dell'*ensemble*, pertanto si procede con l'esplorazione di tale approccio.

1.1.2 Classificatori ensemble

L'ensemble è una tecnica di Machine Learning che combina l'output di due o più classificatori base (che prendono il nome di *weak learners*) per produrre l'output finale [15]. La decisione finale restituita dall'ensemble dovrebbe portare a risultati finali migliori rispetto a quelli che si otterrebbero se si utilizzassero solo i classificatori singoli di cui è costituito l'ensemble.

Un ensemble può essere strutturato in due modi. Un ensemble omogeneo è costituito da due o più *weak learners* dello stesso tipo con l'ipotesi di base che aggregare diverse decisioni semplici possa condurre ad una decisione finale più affidabile. Più interessante è l'ensemble eterogeneo, costituito da classificatori di base differenti. Ogni classificatore di base ha caratteristiche e capacità diverse, perciò, approccerà il problema in maniera diversa. Questo punto è fondamentale in quanto, nei contesti che riguardano il rilevamento delle intrusioni, si è notato che non vi è un singolo classificatore in grado di rilevare tutti gli attacchi con accuratezza ma che determinati algoritmi si prestano a rilevare alcuni attacchi meglio di altri. Aggregare le decisioni dei modelli di base può così scaturire in un sistema finale più robusto e completo. Questa robustezza potrebbe riflettersi anche nelle capacità del sistema di rilevare un attacco zero-day giacché un singolo classificatore potrebbe essere ingannato da un attacco sconosciuto mentre un ensemble potrebbe essere in grado di rilevarlo.

Quanto detto è ottenibile attraverso un ensemble diversificato. Infatti per ottenere un ensemble che operi in maniera più accurata rispetto ai suoi costituenti, è necessario che i classificatori di base siano non solo accurati ma anche diversi [16]. Due classificatori diversi sono carat-

terizzati da errori non correlati. Si supponga di avere un ensemble costituito da tre modelli (C_1, C_2, C_3). Se i tre modelli non sono sufficientemente diversi e se C_1 produce un risultato errato, probabilmente, in maniera similare, anche C_2 e C_3 produrranno un output errato. Se invece supponiamo che l'ensemble sia diversificato e che gli errori commessi dai costituenti siano non correlati, anche se uno dei tre componenti non dovesse classificare correttamente l'input, i rimanenti potrebbero farlo. Attraverso una tecnica di votazione a maggioranza, l'ensemble restituirebbe il risultato corretto [**Al-AâĂŹaraji_2021**].

Oltre a diversificare l'ensemble impiegando classificatori diversi è possibile individuare altri criteri. Si potrebbero addestrare i singoli modelli su insiemi di feature differenti. Vengono così utilizzate più tecniche indipendenti per la selezione delle feature per individuare quelle più rilevanti, garantendo un maggiore accuratezza e velocità di addestramento ed evitando l'overfitting. Ad esempio, attraverso la Recursive Feature Elimination (RFE), è possibile selezionare in maniera ricorsiva le caratteristiche più importanti per un modello.

Gli ensemble differiscono anche per come sono costruiti. È possibile realizzare un ensemble attraverso i seguenti metodi:

- **bagging**: è un metodo che prevede di addestrare i weak learners su sottoinsiemi del dataset generati in maniera casuale. Per creare gli insiemi di addestramento dei singoli modelli viene usato un campionamento con rimpiazzo in cui alcune delle istanze del dataset iniziale possono comparire in più sottoinsiemi di addestramento. In fase di predizione gli output dei classificatori vengono in genere combinati attraverso un meccanismo basato su votazione. Il vantaggio del bagging è che attraverso tale combinazione il sistema risulta più robusto e produce errori più bassi giacché gli errori commessi dai singoli classificatori vengono in qualche modo mediati. Un'applicazione del bagging é costituita dalle Random Forest (RF). Una RF è composta da un insieme di alberi decisionali, ciascuno dei quali è addestrato su un sottoinsieme casuale del dataset. Oltre a differire per le istanze su cui gli alberi decisionali vengono addestrati, essi possono variare anche il sottoinsieme di feature da considerare per i criteri di scelta necessari per la costruzione dell'albero [17]. I componenti della foresta saranno così meno correlati così come lo saranno i loro errori commessi. La RF è un modello di classificazione ampiamente utilizzato nel mondo dei sistemi di rilevamento delle intrusioni poiché si tratta di un classificatore multiclasse che offre ottime prestazioni.

- **boosting**: si tratta di approccio significativo in cui ad ogni iterazione viene posta maggiore

enfasi sulle istanze che sono risultate difficili da classificare ovvero quelle istanze che non sono state classificate correttamente durante l'iterazione precedente. L'enfasi apportata viene quantificata sotto forma di pesi che vengono associati alle istanze del dataset [18]. Inizialmente tutto i pesi sono identici e ad ogni iterazione, i pesi delle istanze classificate in maniera errata aumentano mentre i pesi delle istanze classificate correttamente diminuiscono. Inoltre, un insieme di pesi viene attribuito ai componenti dell'ensemble in base alla loro accuratezza nel classificare le istanze. Di conseguenza quando una nuova istanza viene valutata, si fa maggiore affidamento alle decisioni prese dai componenti con un peso maggiore. Questo avviene perché si effettua una votazione pesata e l'output finale sarà la classe con la votazione più alta.

- **stacking**: è una tecnica che, come il bagging e il boosting, mira ad ottenere una classificazione più accurata ma, a differenza dei metodi precedenti, sfrutta un approccio basato su più livelli. Generalmente si hanno due livelli, il primo costituito dai componenti di base dell'ensemble e il secondo è costituito da un modello che viene chiamato meta-classificatore [19]. I modelli del primo livello vengono prima addestrati e i successivi output prodotti vengono utilizzati per creare un nuovo dataset costituito da un nuovo insieme di feature basate sulle predizioni dei modelli stessi. A ciascuna istanza del nuovo dataset viene associata la corretta etichetta del dataset originale. A questo punto il nuovo insieme di addestramento viene utilizzato per addestrare il meta-classificatore, il quale fornirà la decisione finale.

Quello che si è appena visto riguarda i metodi che è possibile utilizzare per costruire un ensemble. Sia il boosting che il bagging prendono in considerazione gli output dei classificatori di base per ottenere la decisionale finale. Il problema che sorge adesso è come questi output possano essere combinati.

Vi sono principalmente due approcci che vanno per la maggiore:

- **votazione a maggioranza**: in questo caso la classe che costituisce la decisione finale non è altro che la classe più popolare tra quelle predette dai membri dell'ensemble. Ogni modello di base vota per una particolare classe e i voti ottenuti possono essere combinati in diversi modi:
  - *votazione unanime*: tutti i classificatori devono essere d'accordo sulla stessa classe;

  - *maggioranza semplice*: la decisione finale sarà data dalla classe per cui ha votato almeno la metà più uno dei componenti dell'ensemble;
  - *votazione a pluralità*: la classe vincente deve ottenere un numero alto di votazioni, indipendentemente dal fatto che questi voti provengano da almeno la metà dei partecipanti o meno.

- **votazione pesata**: rappresenta un'estensione della votazione a maggioranza in cui ad ogni membro dell'ensemble viene associato un peso. Ciascun peso associato può variare a seconda di quanto il modello sia accurato. Si può quindi attribuire un peso maggiore ad un classificatore per cui si è stabilito che funzioni meglio degli altri secondo un certo criterio. In fase di predizione ciascun modello fornirà il proprio voto mitigato dal peso ad esso associato e la classe con il punteggio più alto costituirà la decisione finale.

Tralasciando la particolare tecnica di Machine Learning impiegata, uno dei requisiti necessari per rendere i misuse-based IDS efficaci è quello di mantenere il database delle firme sempre aggiornato. Questo processo è oneroso e dispendioso in termini di tempo, in quanto richiede una continua ispezione del sistema alla ricerca di vulnerabilità e un conseguente aggiornamento del database delle firme. Considerando la continua comparsa di attacchi zero-day e il dover fare affidamento a un database di firme sempre aggiornato, risulta evidente che non è possibile affidarsi esclusivamente a un sistema di questo tipo per la realizzazione di un IDS efficace. Si ritiene quindi necessario procedere con lo studio dei sistemi basati sull'anomalia.

1.2 Anomaly-based IDS

A differenza dei sistemi di rilevamento delle intrusioni basati su firme, che si affidano a modelli predefiniti di attacco per rilevare minacce conosciute, i sistemi basati su anomalie adottano un approccio più dinamico, analizzando le deviazioni da un modello di traffico considerato normale, con l'obiettivo di identificare comportamenti anomali che potrebbero indicare una potenziale violazione della sicurezza [20].

Il principio fondamentale alla base degli IDS basati su anomalie si basa sull'ipotesi che l'attività normale di rete segua modelli prevedibili. Le deviazioni da questi modelli possono essere indicative di attività dannose o di accessi non autorizzati. Questo approccio consente a questi sistemi di adattarsi alle minacce emergenti e agli attacchi zero-day, fornendo uno strato di difesa prezioso contro vulnerabilità precedentemente sconosciute.

Uno dei principali punti di forza di questa tipologia di IDS è la loro capacità di rilevare attacchi nuovi che potrebbero eludere i sistemi tradizionali basati su firme. Utilizzando algoritmi di Machine Learning e analisi statistica, questi sistemi apprendono e comprendono il comportamento tipico di reti, applicazioni o utenti nel tempo. Qualsiasi deviazione dal modello appreso genera un allarme, scaturendo in ulteriori indagini da parte degli addetti alla sicurezza informatica. Le anomalie possono essere classificate secondo tre tipologie [21]:

- **anomalie puntuali**: in questo caso una singola istanza è considerata anomala rispetto alla restante parte dei dati e prende il nome di anomalia puntuale. Questo è il caso più semplice di anomalia;
- **anomalie contestuali**: se una istanza è considerata anomala in un particolare contesto ma non in un altro, allora si è in presenza di un'anomalia contestuale. Una anomalia contestuale è definita attraverso due tipi di attributi: gli attributi contestuali e gli attributi comportamentali. La prima tipologia di attributi determina il contesto nel quale l'istanza è considerata anomala. La seconda tipologia definisce in che modo l'istanza è considerata anomala;
- **anomalie collettive**: in questo caso un gruppo di istanze anomale è considerato anomalo rispetto all'intero dataset.

La fase di addestramento implica la raccolta e l'analisi di dati per stabilire modelli e comportamenti normali all'interno della rete o del sistema. Ciò può includere parametri come il traffico di rete, l'utilizzo delle risorse di sistema, il comportamento dell'utente e le interazioni delle applicazioni. Modelli di Machine Learning e deep learning, vengono poi impiegati per creare una rappresentazione di traffico che rifletta la normalità attesa.

Tuttavia, l'efficacia dei sistemi di rilevamento delle intrusioni basati su anomalie non è priva di criticità. Questi sistemi tendono infatti ad essere propensi nel generare un alto numero di falsi positivi in cui attività legittime vengono erroneamente segnalate come anomalie, e falsi negativi in cui attacchi effettivi non vengono rilevati. Trovare il giusto equilibrio è cruciale per ridurre al minimo questi errori e garantire l'affidabilità del sistema.

Il monitoraggio continuo e gli aggiornamenti regolari sono essenziali per mantenere l'accuratezza di tali sistemi, poiché l'ambiente di rete e i comportamenti degli utenti possono cambiare nel tempo.

Una volta definita che cosa è un'anomalia e quali tipologie di anomalie possono verificarsi si procede con una panoramica sulle principali tecniche di rilevamento delle anomalie che possono essere classificate in tre categorie: tecniche statistiche, tecniche basate sulla conoscenza e tecniche di Machine Learning [22]. Tali approcci non sono mutuamente esclusivi infatti è possibile utilizzare algoritmi di Machine Learning per implementare tecniche statistiche.

1.2.1 Tecniche statistiche

Queste tecniche utilizzano modelli statistici, distribuzioni di probabilità e analisi matematiche per definire modelli di base e rilevare anomalie basate su valori anomali. Analizzando le proprietà statistiche del traffico di rete, del comportamento del sistema o delle attività degli utenti, gli IDS che sfruttano tali tecniche possono identificare in modo efficace potenziali minacce alla sicurezza in modo dinamico e adattativo.

In base al modello di analisi utilizzato è possibile effettuare una ulteriore classificazione dei sistemi che si basano su questi approcci:

- **analisi univariata**: in questo caso il modello si basa su una singola variabile per costruire il profilo statistico normale;
- **analisi multivariata**: si analizzano le relazioni tra due o più variabili per comprendere la relazione che lega le suddette [23];
- **analisi basata su serie temporali**: si basa su una serie di osservazioni ottenute in certo intervallo temporale. Una osservazione è considerata anomala se la probabilità di verificarsi in un certo intervallo temporale è bassa.

Una delle tecniche statistiche impiegate si basa sull'uso di distribuzioni di probabilità per modellare il comportamento normale. Le distribuzioni gaussiane sono spesso utilizzate per rappresentare i modelli attesi di vari parametri, come il volume del traffico di rete, le dimensioni dei pacchetti o gli intervalli di arrivo tra i pacchetti di rete. Deviazioni da queste distribuzioni, specialmente quelle che cadono nelle code della gaussiana o che mostrano valori estremi, possono essere indicative di attività anomale che richiedono ulteriori indagini. In sostanza, l'ipotesi di base che i dati normali abbiano un'alta probabilità. Al contrario le anomalie sono meno probabili e, di conseguenza, un evento con bassa probabilità viene indicato come anomalia.

L'analisi delle serie temporali è un'altra tecnica statistica ampiamente utilizzata nella rilevazione delle anomalie. Modellando e analizzando i modelli temporali delle metriche di rete

o di sistema, gli IDS possono identificare deviazioni dal comportamento atteso nel tempo [24]. Variazioni e picchi improvvisi di attività possono essere indicativi di anomalie, e l'analisi delle serie temporali consente la rilevazione di tali deviazioni facilitando l'identificazione di potenziali incidenti di sicurezza.

Un altro approccio per il rilevamento delle anomalie si basa su soglie e costituisce una tecnica statistica semplice ma efficace. Comporta la definizione di soglie per specifiche metriche e la generazione di avvisi quando certi valori superano o scendono al di sotto di queste soglie. Nonostante questo approccio possa essere suscettibile a falsi positivi se le soglie non sono calibrate correttamente, rimane comunque una tecnica utile e ampiamente utilizzata.

Il perfezionamento continuo dei modelli statistici, e l'integrazione di questi con approcci di Machine Learning, rimane essenziale per migliorare l'accuratezza ed efficacia dei sistemi che si basano su queste tecniche, in un panorama della sicurezza informatica in continua evoluzione.

1.2.2 Tecniche basate sulla conoscenza

A differenza degli approcci statistici o di Machine Learning che si concentrano sulla modellazione di pattern e deviazioni, le tecniche basate sulla conoscenza si affidano a regole predefinite e conoscenze derivate da esperti per identificare anomalie nelle attività di rete, nel comportamento del sistema o nelle azioni dell'utente.

I sistemi basati su regole rappresentano approccio basato sulla conoscenza e permettono ai professionisti della sicurezza di definire regole esplicite che riflettano i comportamenti normali nella rete [25]. Queste regole sono generalmente elaborate sulla base della conoscenza degli esperti del sistema. Ad esempio, le regole potrebbero stabilire che un certo numero di tentativi di accesso falliti entro un determinato intervallo di tempo debba generare un avviso, segnalando un possibile attacco a forza bruta. Gli IDS basati su regole offrono flessibilità nell'adattare i meccanismi di rilevamento alle caratteristiche uniche della rete e delle attività degli utenti.

Inoltre, è possibile incorporare l'esperienza umana per definire sistemi esperti e basi di conoscenza necessari alla rilevazione delle anomalie basate sulla conoscenza [26]. I sistemi esperti emulano i processi decisionali degli esseri umani, utilizzando una base di conoscenza di regole e un motore di inferenza per trarre conclusioni sulla legittimità delle attività osservate. Questo ragionamento simile a quello umano consente all'IDS di identificare anomalie che potrebbero richiedere una comprensione o interpretazione contestuale al di là di ciò che gli algoritmi automatizzati possono fornire.

Mantenere aggiornato e perfezionare l'insieme delle firme è fondamentale per tenere conto delle modifiche nell'ambiente di rete e delle attività in corso. La combinazione di tecniche basate sulla conoscenza e di altri approcci di rilevamento, come quelli statistici e di Machine Learning, in un'architettura ibrida di IDS fornisce una strategia di difesa più efficace, sfruttando i punti di forza di ciascuna tecnica per creare un sistema più resiliente e adattabile.

1.2.3 Tecniche di Machine Learning

Uno dei principali punti di forza delle tecniche di rilevamento delle anomalie basate sul Machine Learning risiede nella capacità di imparare autonomamente e adattarsi alle complessità del comportamento di rete e alle attività di sistema. Invece di basarsi su regole o firme predefinite, gli algoritmi analizzano grandi dataset ed estraggono pattern e caratteristiche che caratterizzano il comportamento normale. Questo processo consente all'IDS di evolversi dinamicamente e adattarsi ai cambiamenti dell'ambiente di rete, individuando anomalie che potrebbero sfuggire ai sistemi tradizionali basati su firme.

Gli algoritmi di Machine Learning supervisionati rappresentano una categoria predominante all'interno di questo panorama. In questo approccio, il sistema viene addestrato su un insieme di dati etichettati contenente istanze di comportamento normale. Una volta addestrato, il modello può classificare i nuovi input, identificando anomalie che differiscono dal modello di normalità appreso dal dataset.

Le tecniche di Machine Learning non supervisionate sono ampiamente utilizzate anche nei sistemi di rilevamento delle anomalie basati su IDS [27], in particolare quando i set di dati etichettati sono scarsi o difficili da ottenere come nel caso degli attacchi zero-day. Un esempio è dato dagli algoritmi di clustering, come il K-means [28] che raggruppano dati simili sulla base di somiglianze intrinseche, aiutando a identificare pattern e anomalie senza etichette predefinite. È possibile citare anche gli autoencoder [29], un tipo di rete neurale adatta ad apprendere rappresentazioni complesse dei dati, rendendoli efficaci nel rilevare anomalie in scenari in cui le caratteristiche del comportamento normale sono meno chiaramente definite. Aggregando le uscite di modelli diversi, le tecniche di ensemble migliorano la robustezza dei sistemi di rilevamento delle anomalie, riducendo il rischio di falsi positivi e aumentando la precisione complessiva.

Estrarre feature rilevanti dai dati grezzi attraverso tecniche come la PCA [30] migliora la capacità del modello di distinguere tra comportamenti normali e anomali [31]. Ciò può implicare

la trasformazione dei dati grezzi in rappresentazioni significative, la selezione delle feature più rilevanti o addirittura la creazione di nuove caratteristiche che catturano informazioni essenziali per il rilevamento delle anomalie.

Nonostante la loro efficacia, i metodi di rilevamento delle anomalie basati sull'apprendimento automatico non sono privi di sfide. L'overfitting, un fenomeno per cui il modello diventa troppo specifico per i dati di addestramento e si comporta in maniera non ottimale sui nuovi dati, è una preoccupazione comune. Inoltre, la natura dinamica degli ambienti di rete richiede un aggiornamento continuo del modello di normalità appreso per adattarsi ai cambiamenti dell'ambiente e alle minacce emergenti.

1.3 Approcci comuni per la rilevazione di attacchi zero-day

Una volta esaminate le principali tecniche impiegate per la realizzazione degli IDS, si ritiene utile approfondire gli approcci comuni proposti dallo stato dell'arte per la progettazione di sistemi in grado di rilevare attacchi zero-day.

1.3.1 Sistemi basati su soglia

Uno degli approcci maggiormente utilizzati per la rilevazione di istanze sconosciute è quello di sfruttare delle soglie. I modelli impiegati vengono prima addestrati e successivamente si cerca di trovare un valore soglia idoneo basandosi su un certo punteggio ottenuto durante la fase di predizione. Il punteggio di predizione assume forme diverse a seconda della tecnica utilizzata.

In genere quello che si fa quando si sfrutta tale tecnica è quello di addestrare il classificatore solo sul traffico normale, trovare un valore soglia consono e classificare come anomale tutte le istanze aventi un punteggio di predizione maggiore rispetto alla soglia precedentemente definita. Ad esempio, si potrebbe addestrare un autoencoder solo sul traffico normale con l'obiettivo di minimizzare l'errore di ricostruzione; questo permette anche di individuare dei candidati da utilizzare come valore soglia. Si potrebbe dunque ipotizzare di utilizzare l'errore di ricostruzione medio delle istanze normali come valore soglia. Tutte le istanze che avranno un errore di ricostruzione maggiore della soglia verranno considerate anomale basandosi sull'ipotesi che l'autoencoder produrrà un valore elevato errore di ricostruzione per le anomalie giacché queste

ultime differiscono sostanzialmente dal traffico normale. Diversi approcci sono stati proposti per il calcolo della soglia. Una delle tecniche più comuni prevede l'utilizzo dell'errore quadratico medio tra l'istanza in input e l'output dell'autoencoder:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x_i})^2 \tag{1.2}$$

Anche La norma L_2 può essere utilizzata, calcolando la differenza tra le feature dell'input e dell'output ricostruito [33] come segue:

$$L_2 = \|X - \hat{X}\|^2 \tag{1.3}$$

Nonostante le tecniche basate su soglia siano ampiamente utilizzate, presentano delle limitazioni. Vi sono delle situazioni in cui le istanze normali sono posizionate in zone ad alta densità e le istanze anomale sono posizionate in zona a bassa densità. In questo caso anche una singola soglia fornisce una buona soluzione. Ma nella maggior parte delle situazioni reali una separazione così netta non si verifica e l'utilizzo della soglia può generare un alto numero di falsi positivi. Inoltre, gli attaccanti potrebbero generare una tipologia di attacco che si presenta come una variazione di istanze legittime, ingannando il sistema e rendendo la soglia inutile. In alcuni contesti una singola soglia può non risultare sufficiente perché l'errore di ricostruzione non è linearmente separabile e questo si verifica specialmente nei casi in cui si ha a che fare con dati ad elevata dimensionalità [33]. Per questa ragione vengono impiegate in maniera congiunta ad algoritmi di Machine Learning per realizzare sistemi più complessi. In figura 1.1 è mostrata l'architettura di un sistema che fa uso di molteplici soglie applicate agli output di algoritmi di Machine Learning supervisionati e non supervisionati per garantire la capacità di riconoscimento sia di attacchi noti che di attacchi zero-day [34]. Le performance di tale sistema verranno in seguito impiegate come termine di confronto durante la fase di valutazione sperimentale.

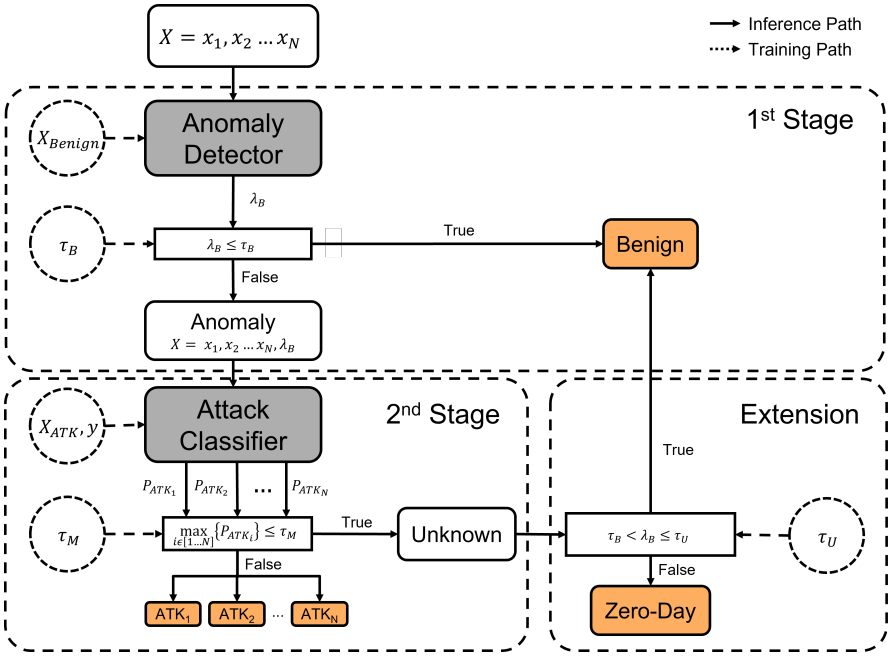

Figura 1.1: Architettura del sistema proposto da [34]

1.3.2 Sistemi basati sul calcolo delle distanze

Gli approcci basati sulle distanze usano una funzione di distanza per rilevare le anomalie. L'idea dietro questo approccio è che le istanze anomale si trovino in zone isolate e che quindi siano distanziate dalla maggior parte delle altre istanze. Alcuni sistemi usano funzioni di distanza più tradizionali. Ad esempio, dei ricercatori hanno sfruttato la distanza di Minkowski per calcolare i K vicini più vicini tra i dati in un sistema basato su un Local Outlier Factor (LOF) e autoencoder [35]. Viene calcolata la distanza tra una istanza e il traffico normale affinché sia proprio la distanza a identificare le istanze anomale da quelle normali. Le istanze anomale saranno maggiormente distanziate da quelle normali.

Il calcolo della distanza di Minkowski tra due vettori X e Y a n viene effettuato come segue:

$$Minkowski(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}} \tag{1.4}$$

Altri sistemi sfruttano approcci meno convenzionali. È stato proposto un sistema che calcola la distanza media per classe nello strato di embedding di una rete neurale, cioè l'ultimo strato

prima dello strato softmax [36]. Il motivo di questa scelta è che lo strato di embedding ha un numero di dimensioni pari al numero delle classi del dataset e che ogni classe in questo layer costituisce un cluster separato. Ogni istanza che si discosta notevolmente dalla media della corrispondente classe verrà etichettata come una istanza sconosciuta. La formula utilizzata per calcolare la distanza di un'istanza dalla media della classe corrispondente alla classe è la seguente:

$$distanza(X) = \min_{1 \leq j \leq K} \|\overline{\mu}_j - \overline{z}\|_2^2 \tag{1.5}$$

dove K è il numero totale di classi conosciute, $\overline{\mu}_j$ è la media del cluster e $\overline{z}$ è la proiezione dell'istanza in input.

Gli approcci basati sulla distanza hanno però delle limitazioni. Spesso può risultare complicato trovare una funzione di distanza appropriata considerato lo spazio di feature in input, di conseguenza il sistema potrebbe non ottenere buoni risultati nella rilevazione di istanze sconosciute [37]. Inoltre, questi approcci necessitano di un numero sufficientemente alto di campioni di attacco per poter modellare correttamente la classe. Un elevato numero di campioni potrebbe non essere sempre a disposizione specialmente in presenza di attacchi che si comportano come piccoli varianti di altri. Per queste ragioni tali sistemi potrebbero non essere efficaci nel rilevare attacchi zero-day [38].

1.3.3 Classificazione a classe singola

Questa tecnica prevede di addestrare un classificatore solo sulle istanze normali. Il classificatore impara così a modellare la normalità e tutto ciò che differisce da tale modello viene considerato come un'anomalia [39]. La classificazione su singola classe è una tecnica che spesso viene utilizzata insieme ad altre tecniche come, ad esempio, tecniche basate sull'uso di soglie o sul calcolo delle distanze. In genere dopo che un modello viene addestrato sulle istanze normali, viene calcolato un punteggio per i nuovi input che definisce il loro grado di anomalia. Qualora il punteggio associato ad un input dovesse superare una certa soglia, l'istanza verrebbe rigettata in quanto non appartenente alla classe normale. Questa tecnica, quindi, effettua una classificazione binaria in quanto separa le istanze di input in due gruppi, costituiti dalle istanze normali e da quelle malevole.

I modelli maggiormente utilizzati impiegati nell'utilizzo di tale tecnica sono gli autoencoder e le one-class support vector machine (OC-SVM). La SVM è uno dei modelli supervisionati maggiormente utilizzati. Quest'ultimo è addestrato con l'obiettivo di trovare un iperpiano

che separi le classi presenti nel dataset. Quando i dati non sono linearmente separabili viene utilizzato un kernel che permette di mappare i data in uno spazio a maggiore dimensionalità in cui è possibile trovare un iperpiano di separazione. La controparte non supervisionata della SVM è la OC-SVM. In questo caso, a differenza del precedente, l'iperpiano costruito è sferico e racchiude le istanze appartenenti all'unica classe su cui il modello è stato addestrato. La OC-SVM è uno degli algoritmi maggiormente utilizzati per effettuare la rilevazione di outlier nei dati [40].

Sono state sollevate delle criticità sui sistemi basati su classificazione a classe singola. Infatti in alcuni contesti questo approccio pure risultare non efficace in quanto gli attaccanti possono utilizzare strumenti sofisticati per generare attacchi molto simili al traffico normale. In questi casi questa tecnica non è una soluzione ottimale in quanto non sarebbe in grado di separare le istanze benigne da quelle malevole. Inoltre, i classificatori addestrati solo sul traffico normale, senza considerare esempi appartenenti alla classe anomala, potrebbe generare un alto numero di falsi positivi. E ancora, questi algoritmi possono essere anche computazionalmente onerosi. Si è visto infatti che un OC-SVM è dispendioso in termini di risorse quando si a che fare con dati con elevata dimensionalità [33].

1.3.4 Sistemi basati sul comportamento

Questa tipologia di IDS si focalizza sull'analisi del comportamento per individuare istanze anomale da etichettare come attacchi zero-day, a differenza degli IDS realizzati attraverso tecniche di Machine Learning che sfruttano un insieme statico di feature.

Sulla base di questa ipotesi è stato realizzato un sistema che analizza i log negli endpoint e crea delle politiche di accesso [35]. Un LOF e un autoencoder vengono utilizzati per analizzare i log alla ricerca di differenze rispetto a quelli attesi con l'obiettivo di calcolare un punteggio. Tale punteggio è indice di quanto il comportamento sia anomalo. Una differenza sostanziale tra un nuovo log e quello atteso produce un punteggio elevato, sintomo di un possibile attacco zero-day.

Altri sistemi invece mirano ad analizzare il codice binario dei file eseguibili presenti in una macchina target. In particolare l'attenzione si focalizza sulle chiamate alle API [41]. Questo perché si suppone che le chiamate alle API siano delle buone feature per un classificatore in grado di predire un comportamento anomalo poiché permetterebbero di analizzare il comportamento di un possibile attacco.

Un approccio simile è quello che si focalizza sulle chiamate di sistema in quanto permettono di comprendere la dipendenza tra le applicazioni e le risorse del sistema [42]. Si suppone infatti che un attacco generi un numero maggiore di chiamate di sistema e che queste differiscano anche da quelle fatte da un normale programma. Dall'analisi delle chiamate di sistema, quindi, è possibile estrarre informazioni inutili sul comportamento di un programma e la loro analisi aiuta l'individuazione di attacchi zero-day.

1.3.5 Sistemi basati su algoritmi di DL

A causa della continua evoluzione della rete e del volume di dati generato, si stanno cercando soluzioni alternative ai tradizionali algoritmi di Machine Learning. Per questa ragione gli algoritmi di Deep Learning stanno riscuotendo un grande successo e vengono utilizzati sempre di più. Infatti, grazie alle loro caratteristiche permettono di risolvere alcuni problemi che emergono con l'utilizzo di approcci di ML tradizionali. Tra queste proprietà è possibile notare la loro capacità di elaborare in maniera efficiente grandi quantità di dati, la capacità di estrarre le feature in maniera automatica senza che questa operazione venga fatta in maniera manuale e la possibilità di generare dataset fittizi per valutare la capacità di generalizzazione del modello.

Gli algoritmi più utilizzati possono essere suddivisi come segue:

- **autoencoder**: la struttura tipica di un autoncoder prevede due parti speculari chiamate encoder e decoder. Sia l'input e l'output hanno la stessa dimensione. Il suo funzionamento prevede l'utilizzo dell'encoder per ridurre le dimensioni dell'input e trasformarlo in uno spazio latente (ottenendo così una codifica dell'input); il decoder tenta di ricostruire l'input originale a partire dalla codifica cercando di minimizzare l'errore di ricostruzione. Gli autoncoder sono spesso utilizzati nella rilevazione degli attacchi zero-day anche a causa delle diverse varianti che è possibile impiegare. Ad esempio gli adversarial autoencoder permettono la generazione di traffico malevolo fittizio per la valutazione del sistema [37]. E ancora, i denoising autoencoder vengono utilizzati per corrompere i dati di input e di ricostruire il dato originale a partire dall'input corrotto [43] permettendo così una maggiore generalizzazione del traffico normale.

- **reti neurali convoluzionali (CNN)**: sono reti specializzate per risolvere problemi complessi come l'elaborazione delle immagini e il riconoscimento di oggetti presenti in un'immagine. Il rilevamento di anomalie per mano di un IDS viene fatto sul traffico di rete

che per sua natura è sequenziale e di conseguenza, per questo tipo di problemi, vengono utilizzate reti convoluzionali con kernel ad una sola dimensione (1D-CNN). Una proprietà particolarmente utile delle CNN, (e per cui trovano un grande impiego nel mondo della ricerca) è la loro capacità di estrarre le feature rilevanti per il problema in esame in maniera automatica senza la necessità di affrontare il problema in maniera esplicita attraverso altri approcci specifici per l'estrazione delle feature. Una tipica CNN è costituita da strati convoluzionali, strati di pooling e strati densi. In maniera molto generale, gli strati convoluzionali si occupano dell'estrazione delle feature a partire dai dati di input; gli strati di pooling sottocampionano i dati e sopprimono le feature non rilevanti rendendo il modello meno suscettibile all'overfitting. I dati vengono infine inoltrati alla parte finale della rete che si occupa della classificazione. In una CNN il numero di connessioni tra i neuroni è ridotto e ciò si traduce in un numero minore di parametri da addestrare aiutando a diminuire il tempo e l'onere computazionale necessario per l'addestramento del modello. Diversi studi impiegano le CNN per la rilevazione di attacchi zero-day. Ad esempio, è stato proposto un sistema di rilevazione dell'anomalia che elabora solo i primi pacchetti di un flusso tra una sorgente e una destinazione. Il sistema sfrutta proprio la capacità di estrazione delle feature tipica delle CNN, è addestrato solo sulle istanze normali di traffico e utilizza una soglia per la rilevazione di attacchi zero-day [44]. Un altro sistema proposto utilizza le CNN per implementare un approccio di rilevazione degli attacchi zero-day basato sul comportamento. In questo sistema la rete convoluzionale viene utilizzata per estrarre feature costituite da sequenze binarie per poi convertirle in un vettore e analizzarle. Da quanto si evince dallo studio le CNN sono in grado di rappresentare le caratteristiche comportamentali in maniera più precisa rispetto ad approcci tradizionali di ML come le SVM [41].

- **long-short term memory (LSTM)**: si tratta di un particolare tipo di rete neurale ricorrente (RNN) dotata di memoria che ha la capacità di immagazzinare le informazioni rilevanti e risolve il problema del gradiente che svanisce, tipico delle RNN tradizionali. La struttura di una LSTM presenta tre componenti chiamati "*gate*": input, output e forget gate. Questi componenti regolano come l'informazione fluisce all'interno della memoria della rete ed in particolare controllano quali informazioni vengono memorizzate, quali devono rimanere memorizzate e quali informazioni possono essere sovrascritte perché sono ritenute irrilevanti. Il meccanismo di feedback di cui sono dotate fa si che abbiano natura

ciclica e quindi possono elaborare in maniera efficace i dati sequenziali. È stato proposto un sistema per la rilevazione di attacchi zero-day che si basa su LSTM per analizzare le dipendenze sequenziali tra le applicazioni e le risorse di sistema. Questo perché un software malevolo generare chiamate di sistema differenti e in numero maggiore rispetto a un normale programma. L'analisi delle chiamate di sistema può aiutare nella rilevazione di attacchi zero-day [42].

Nonostante gli approcci basati sul Deep Learning siano promettenti presentano delle limitazioni. I modelli di DL necessitano di grandi quantità di dati per l'addestramento. Come detto in precedenza, l'ottenimento di dati da utilizzare per l'addestramento risulta essere uno degli aspetti più critici nell'ambito dell'intrusion detection e ciò si amplifica ulteriormente quando si utilizzano algoritmi di DL che necessitano di dati in quantità. Spesso, inoltre, i dataset che si hanno a disposizione sono sbilanciati in quanto presentano in abbondanza campioni appartenenti alla classe normale e pochi campioni appartenenti alle classi di attacco. I due problemi appena elencati influenzano negativamente le performance, che evidenziano una bassa capacità di generalizzazione dovuto allo scarso numero di campioni di addestramento e una tendenza a preferire la classe maggioritaria dovuta al dataset sbilanciato.

Da non sottovalutare infine sono gli aspetti dovuti al costo computazionale; infatti, questi modelli (soprattutto quelli più grandi e complessi) possono risultare onerosi sia computazionalmente che in termini di tempo. Al fine di avere un sistema sempre efficace e aggiornato potrebbe essere necessario addestrare nuovamente l'intero sistema per adattarsi alle nuove minacce. Questo potrebbe essere un problema quando si ha a che fare con modelli di Deep Learning complessi, specialmente in ambienti a risorse limitate.

Capitolo 2

Sistema proposto

I sistemi di rilevamento delle intrusioni proposti spesso fanno affidamento su un unico modello di machine learning e ciò pone non poche criticità. Tali sistemi sono, infatti, completamente esposti alle problematiche del singolo algoritmo utilizzato. Di conseguenza, come si è visto, si preferisce utilizzare una moltitudine di modelli, il cui utilizzo congiunto porta ad un miglioramento delle performance. Un'altra criticità è che tutti i campioni in input verrebbero analizzati dal singolo modello impiegato. Quanto detto si traduce in un alto onere computazionale e in una elevata latenza di risposta. Infine, la gran parte dei sistemi proposti, ignora totalmente il problema del rilevamento degli attacchi zero-day, non fornendo una soluzione esaustiva. Consci di queste criticità e di quanto visto in precedenza nello stato dell'arte, in termini di informazioni utili e limitazioni, si è provveduto alla progettazione del sistema proposto.

Il lavoro svolto ha condotto alla realizzazione di un IDS completo, dotato di un'architettura su più livelli, che rileva in maniera gerarchica sia gli attacchi conosciuti che gli attacchi zero-day. Il sistema proposto, oltre a fornire un'efficace capacità di rilevazione degli attacchi zero-day, offre prestazioni migliori rispetto a sistemi basati su più livelli già esistenti.

L'IDS proposto è un sistema ibrido che effettua la classificazione delle istanze attraverso due livelli. La sua struttura generale è mostrata in Figura 2.1.

Il primo livello sfrutta una combinazione di due modelli ed effettua la rilevazione dell'anomalia attraverso un approccio basato su soglia. Le anomalie rilevate sono proprio gli attacchi zero-day. In particolare, il primo livello classifica l'input come benigno, attacco conosciuto o attacco zero-day. Questo livello si comporta come un filtro poiché solo quello che viene etichettato come attacco conosciuto viene inoltrato al secondo livello. Questa fase è cruciale poiché

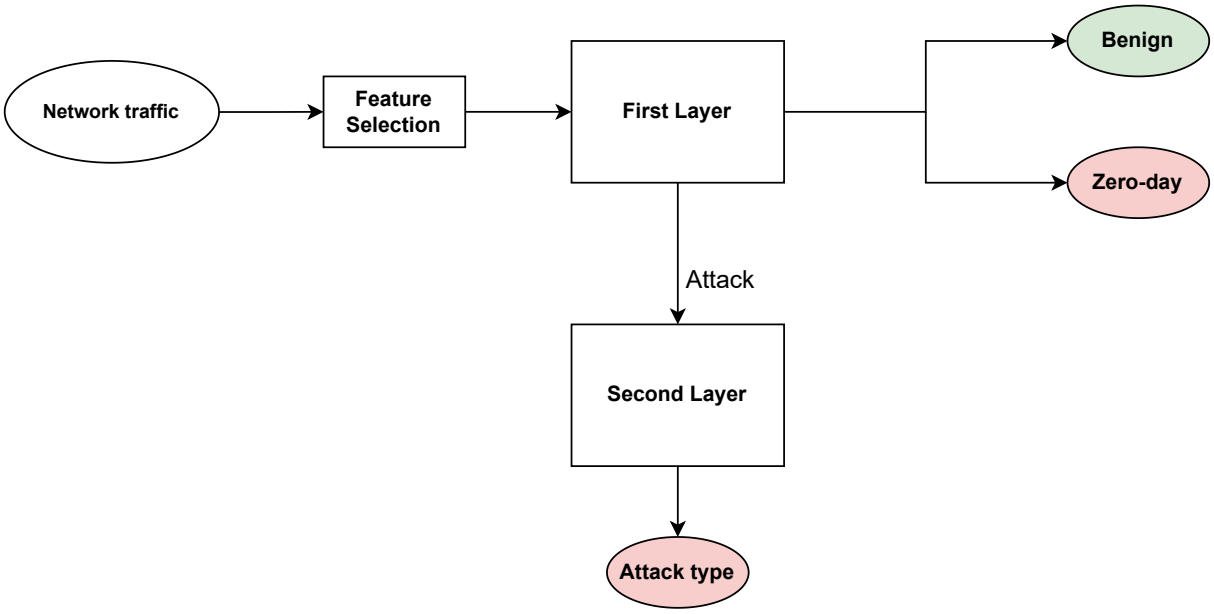

Figura 2.1: Panoramica del sistema proposto

il sistema deve riuscire ad apprendere con precisione le caratteristiche tipiche del traffico benigno e degli attacchi conosciuti cosicché gli attacchi zero-day (che rappresentano le anomalie) possano essere rilevati efficacemente. Qualora ciò non si verificasse, non poche sarebbero le ripercussioni. Un attacco zero-day non rilevato e classificato come benigno riuscirebbe a passare i controlli non venendo bloccato. L'attacco verrebbe quindi eseguito con successo causando anche non pochi danni e verrebbe eventualmente individuato quando è troppo tardi. Un'altra possibilità è che un attacco zero-day venga etichettato come un attacco conosciuto. In questo caso l'input verrebbe inoltrato al secondo livello che cercherà invano di classificarlo in una delle categorie di attacco conosciute. Per quanto questa situazione sia meno spiacevole di quella del caso precedente, potrebbe comportare una risposta non corretta dell'amministratore di rete, in un vano tentativo di bloccare un particolare attacco che in realtà non si sta verificando. Specularmente si vuole anche evitare che il traffico benigno o gli attacchi conosciuti vengano etichettati come attacchi zero-day: se del traffico normale venisse etichettato come attacco zero-day verrebbe bloccato interferendo con l'esperienza d'uso di un utente legittimo e ciò non è auspicabile perché si vuole che questi sistemi operino in maniera trasparente per l'utente; sarebbe da evitare anche il caso in cui un attacco conosciuto venga etichettato come attacco zero-day poiché ciò non permetterebbe a chi di dovere di intervenire in maniera adeguata nonostante la soluzione

corretta sia disponibile. Riassumendo, quindi, è cruciale che questa prima fase di classificazione produca un basso numero di falsi positivi e di falsi negativi. Infine, è necessario che tale livello sia estremamente efficiente in quanto ha l'onere di analizzare tutto il traffico e in caso contrario potrebbe diventare il collo di bottiglia dell'intero sistema.

Il risultato di questa prima fase è che le istanze etichettate come benigne o come attacchi zero-day vengono velocemente valutate e non necessitano di ulteriore analisi. Al contrario invece, gli attacchi conosciuti vengono inoltrati al secondo strato poiché necessitano di ulteriori attenzioni.

Il secondo livello si occupa di associare le istanze in input alla corretta categoria di attacco fornendo una classificazione a grana più fine e precisa. Questo approccio gerarchico su più livelli permette di utilizzare una logica più complessa nel secondo livello poiché il numero di campioni da analizzare è notevolmente ridotto, garantendo delle prestazioni migliori. Per effettuare questa ulteriore classificazione viene impiegato un ensemble di classificatori in grado di rilevare efficacemente gli attacchi conosciuti. L'idea è quella di raggiungere una classificazione finale migliore sfruttando la congiunzione di algoritmi realizzata attraverso l'ensemble piuttosto che fare affidamento sulle capacità di un singolo modello [45].

Il sistema da proteggere può influenzare la scelta dei componenti dell'ensemble. Ciò introduce un fattore di flessibilità in quanto è possibile adattare il sistema al contesto: non tutti gli ambienti mettono a disposizione le stesse risorse computazionali, basti pensare alle reti costituite da dispositivi a risorse limitate come quelli IoT.

L'ensemble è anche facilmente estensibile, in quanto è possibile aggiungere nuovi classificatori che siano in grado di classificare con maggiore accuratezza le eventuali nuove minacce scoperte. Ciò rende il sistema robusto in un ambiente dinamico in cui il profilo degli attacchi cambia rapidamente a causa di nuove minacce che emergono.

La decisione finale viene delegata ad un modulo che tiene in considerazione il risultato restituito dall'ensemble attraverso un meccanismo di votazione pesata e l'output fornito da un meta-classificatore. Il meta-classificatore usa un vettore di feature proveniente dall'ensemble e permette così di sfruttare la tecnica della *stacked generalization*. Il modulo di decisione finale calcola, quindi, un valore euristico per ciascuno di questi due risultati ottenuti e sceglie come decisione finale l'output che ha prodotto il valore euristico maggiore.

Oltre ai vantaggi già elencati è possibile evidenziarne un altro. L'architettura del sistema permette una installazione gerarchica distribuita. Si ipotizzi che sia necessario proteggere un sistema composto da più sottoreti. Si potrebbe pensare di installare il primo livello sulla singola

sottorete. Questo avrebbe diversi vantaggi. Innanzitutto, poiché il primo livello si troverebbe vicino al sistema da monitorare, verrebbero prodotti risultati con una bassa latenza e inoltre si avrebbe una preservazione della privacy perché il traffico benigno non necessita di essere inoltrato al secondo livello. In seconda battuta, sarebbe possibile addestrare il primo strato sui dati benigni locali della sottorete. Questo permette di avere un sistema più flessibile e più accurato poiché viene modellato con precisione il traffico benigno localmente, avendo una maggiore capacità di rilevazione degli attacchi zero-day.

Nella restante parte di questo capitolo si procede con l'approfondimento dei singoli livelli del sistema proposto e delle scelte progettuali.

2.1 Background

2.1.1 Novelty detection

La novelty detection è l'identificazione di dati nuovi o sconosciuti di cui un algoritmo di machine learning non è a conoscenza durante l'addestramento [46]. Il problema della novelty detection rientra nell'ambito della classificazione a singola classe, in cui l'unica classe presente viene considerata come positiva e deve essere distinta da tutto il resto. La novelty detection, come la outlier detection, è una tecnica di anomaly detection in cui si è interessati nel rilevare le osservazioni anomale. Le due tecniche differiscono per le ipotesi di base. Con l'outlier detection si assume che l'insieme di addestramento sia inquinato da valori anomali che si trovano in regioni dello spazio lontane dai valori considerati normali. Un outlier è definito come un dato che non appare coerente con il resto dell'insieme di addestramento. Durante l'addestramento il modello si concentrerà sulle regioni in cui i dati sono maggiormente concentrati cercando di ignorare le eventuali deviazioni dovute ad anomalie. Quando si sfrutta la novelty detection, invece, si suppone che l'insieme di addestramento non sia affetto da valori anomali e si è interessati nel determinare se un nuovo campione è normale o meno. Il termine *novelty* fa riferimento a qualcosa di nuovo e che non somiglia a qualcosa di precedentemente conosciuto o usato. Per queste ragioni spesso ci si riferisce alla outlier detection come anomaly detection non supervisionata mentre ci si riferisce alla novelty detection come anomaly detection semi-supervisionata, giacché il modello è costruito utilizzando dei dati di addestramento che appartengano tutti alla medesima classe, ovvero quella positiva (qualsiasi essa sia).

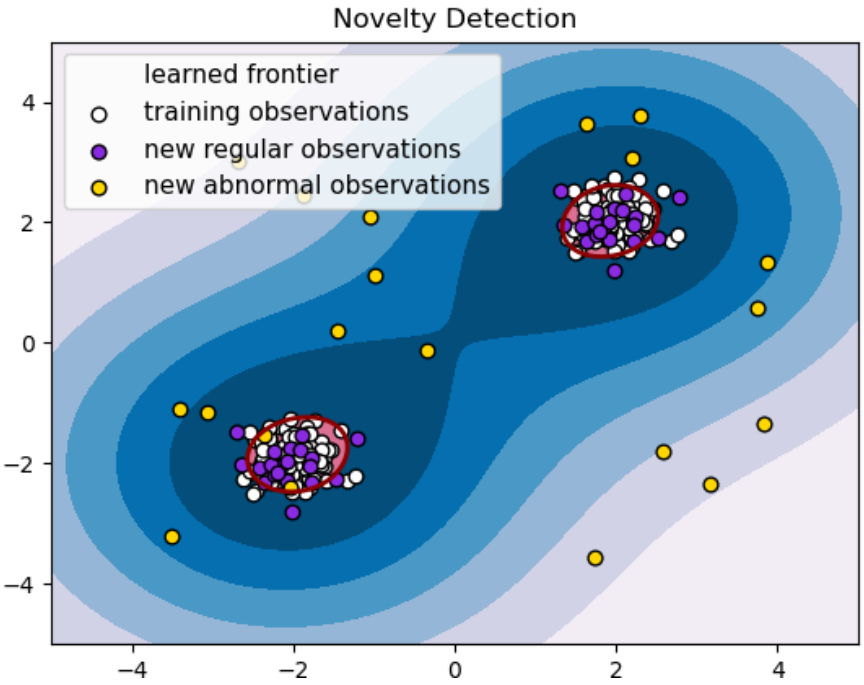

Figura 2.2: Funzionamento Novelty Detection [47]

Solitamente, in questo tipo di problemi, si ha un'abbondanza di campioni appartenenti alla classe positiva e pochi campioni, se non nessuno, appartenente alla classe negativa. La scarsità dei campioni negativi potrebbe essere dovuta alla bassa frequenza con cui questi eventi si verificano o alla difficoltà nell'ottenere questo tipo di campioni. Quanto detto può essere facilmente compreso prendendo come esempio il problema in esame. Infatti, si supponga di voler realizzare un sistema di rilevamento delle intrusioni in grado di rilevare i comportamenti anomali rappresentati dagli attacchi zero-day. Ottenere campioni di traffico normale è semplice e non richiede altro che la raccolta del traffico in un normale scenario di funzionamento della rete. Al contrario, però, è impossibile ottenere campioni che siano rappresentativi della classe anomala perché ciò richiederebbe di avere un insieme di campioni che sia rappresentativo di tutti i possibili attacchi zero-day. Ciò risulta essere impossibile in quanto un attaccante può sempre escogitare una nuova tipologia di attacco.

Nella novelty detection, quindi, un insieme di istanze X appartenenti alla classe positiva viene utilizzato per apprendere un modello $M(\theta)$ che rappresenta la normalità, in cui θ costituisce i parametri appresi. Il modello M viene utilizzato per assegnare un punteggio $s(x)$ ai nuovi input, non precedentemente osservati. Un punteggio più alto si traduce in una maggiore possibilità

che l'input sia anomalo. Viene definita una soglia k tale che se l'input x ottiene un punteggio $s(x) \leq k$ allora viene classificato come normale, viceversa viene classificato come anomalo.

Vi sono diversi metodi per determinare i parametri e la soglia del modello M da apprendere. Le tecniche di novelty detection possono essere classificate come segue [48]:

- **probabilistiche**: vengono utilizzati metodi probabilistici che mirano a stimare la densità delle istanze positive. In altre parole, questi metodi assumono che le regioni a bassa densità hanno una minore probabilità di contenere istanze normali;
- **basate sulla distanza**: in questi casi si fa affidamento a concetti come il calcolo dei vicini più vicini e all'analisi di cluster di dati. L'assunzione è che i dati appartenenti alla classe positiva hanno la tendenza a formare dei cluster mentre i dati anomali si trovano in regioni lontane dai cluster formati;
- **basate sull'errore di ricostruzione**: il nuovo input viene passato al modello il quale calcola un errore di ricostruzione. L'errore calcolato viene utilizzato come punteggio $s(x)$ e se quest'ultimo differisce notevolmente dall'usuale errore di ricostruzione delle istanze positive, indica la presenza di input anomalo;
- **basate sul dominio**: queste tecniche mirano a descrivere un dominio che contiene i dati normali attraverso la definizione di un confine che contiene i dati stessi e ne segue la distribuzione;
- **basate sulla teoria dell'informazione**: questi approcci utilizzano delle misure della teoria dell'informazione, come l'entropia, per misurare il contenuto informativo dei dati di addestramento. L'idea è che i valori anomali causino un'alterazione del contenuto informativo dei dati.

2.1.2 Entropia

Nella teoria dell'informazione, l'entropia è una misura di incertezza e casualità associata ad una variabile causale. La teoria dell'informazione è una branca della matematica che si occupa di quantificare il contenuto informativo di un messaggio trasmesso [49]. I campi di applicazione sono molteplici e spaziano vari campi come la teoria della probabilità, l'informatica e le telecomunicazioni.

L'informazione può essere definita come l'interpretazione o il significato che viene attribuito ad un insieme di dati. L'informazione può essere definita in un altro modo osservandola da un altro punto di vista. Essa può essere espressa come la riduzione di incertezza. Meno probabile è un evento, maggiore informazione fornisce quando si verifica. Shannon introdusse il concetto di "bit", l'unità base dell'informazione, per rappresentare una decisione binaria. L'entropia può quindi essere utilizzata come misurare di incertezza legata alla realizzazione di una variabile aleatoria.

Data la probabilità p_i di un evento, l'entropia può essere calcolata come segue:

$$H = -\sum_i p_i \cdot \log_b p_i \tag{2.1}$$

La precedente formula corrisponde all'equazione dell'entropia di Shannon.

2.1.3 Stacked generalization

La stacked generalization [50], comunemente nota come stacking, è una sofisticata tecnica di apprendimento progettata per migliorare le prestazioni predittive dei modelli di machine learning. Questo approccio prevede l'uso congiunto di una collezione di modelli di base diversi, seguita dall'integrazione delle loro previsioni attraverso un meta-modello, spesso indicato come modello di stacking. L'essenza dello stacking risiede nella sua capacità di sfruttare i punti di forza unici dei singoli modelli e di compensare i loro limiti imparando a pesare efficacemente i loro contributi. Creando una struttura gerarchica in cui il meta-modello apprende dai risultati dei modelli di base, lo stacking può catturare relazioni più intricate all'interno dei dati, trascendendo le capacità dei singoli modelli. Questa combinazione gerarchica delle previsioni consente allo stacking di migliorare la generalizzazione, la robustezza e l'accuratezza delle previsioni. Ampiamente applicabile in vari scenari di machine learning, la stacked generalization rappresenta una tecnica versatile, che fornisce una strategia dinamica ed efficace per costruire modelli predittivi più resistenti e accurati.

Per aumentare la capacità predittiva dei modelli, questa tecnica opera attraverso un processo a più fasi. Inizialmente, una serie di modelli di base diversificati viene addestrata sullo stesso insieme di dati, ognuno dei quali cattura diverse sfaccettature nei dati sottostanti. Questi modelli di base generano previsioni individuali. Nella fase successiva, un meta-modello, spesso un algoritmo più complesso (ma non necessariamente), viene addestrato per combinare e perfezionare i

risultati dei modelli di base. Questo meta-modello impara a valutare efficacemente i contributi di ciascun modello di base, riconoscendone i punti di forza e compensandone le debolezze. La previsione finale viene quindi effettuata dal meta-modello, che ha imparato a sfruttare l'intelligenza collettiva dei modelli di base. Questo approccio gerarchico permette alla stacked generalization di adattarsi a relazioni complesse all'interno dei dati, portando a prestazioni di generalizzazione migliori e a un modello più robusto in grado di gestire input e scenari diversi.

L'addestramento del meta-modello è un passo fondamentale per realizzare il pieno potenziale di questa tecnica di apprendimento. Una volta che i modelli di base hanno generato individualmente previsioni sul dataset di addestramento, queste previsioni servono come feature di input per il meta-modello [51]. Quest'ultimo viene quindi addestrato su questo nuovo insieme di addestramento, dove ogni istanza è associata alle previsioni dei corrispondenti modelli di base. Durante l'addestramento, il meta-modello impara a discernere le relazioni e gli schemi nelle previsioni dei modelli di base e a combinarli in modo ottimale per generare un risultato più preciso. Questo processo di addestramento prevede in genere tecniche come la cross-validation per garantire la robustezza ed evitare l'overfitting. Di conseguenza, il meta-modello diventa abile nel fornire una previsione finale che sintetizza le diverse intuizioni raccolte dai modelli di base.

2.2 Primo Livello: filtraggio del traffico

OMISSIS

2.3 Secondo Livello: classificatore attacchi

OMISSIS

Capitolo 3

Valutazione sperimentale

Una volta descritto il sistema proposto e i dettagli progettuali che hanno portato alla sua realizzazione, si procede con la valutazione sperimentale. Verrà introdotto il dataset utilizzato e le relative elaborazioni applicate. Successivamente si definiranno le metriche impiegate per valutare il sistema. Infine, si mostreranno i risultati ottenuti con le dovute osservazioni e si effettuerà un'analisi comparativa con un sistema di confronto.

3.1 Dataset

Nell'ambito degli IDS, diversi dataset sono stati proposti per la valutazione delle prestazioni. Uno dei primi fu il DARPA98, ormai caduto in disuso poiché ritenuto antiquato e non rappresenta scenari di traffico reali. Da esso è stato derivato il KDD99 [52], che presenta nuove varianti di attacco. Anche questo dataset è affetto da criticità come un elevato numero di campioni ridondanti e corrotti. Il NSL-KDD è stato realizzato appositamente per sopperire ad alcune lacune del KDD99. Nonostante questi due dataset vengano ancora impiegati, il loro utilizzo è ampiamente criticato e sconsigliato poiché sono ritenuti obsoleti. Le limitazioni degli attuali dataset hanno portato il Canadian Institute for Cybersecurity (CIC) alla realizzazione di un dataset più recente che meglio si presti alla valutazione degli IDS.

Affinché un dataset sia considerato affidabile è necessario che goda di diverse proprietà:

- il traffico di rete contenuto dovrebbe essere reale e valido affinché rappresenti scenari di traffico che si possano verificare in un ambiente reale;
- i campioni benigni e malevoli devono essere etichettati correttamente;

- dovrebbe essere sufficientemente diversificato in termini di comportamento degli utenti e profili di attacco;
- la sua realizzazione dovrebbe avvenire in maniera corretta e permettere un confronto coerente tra diversi approcci;
- non deve contenere informazioni private affinché sia facilmente condivisibile;
- dovrebbe essere fornita una documentazione a supporto che descriva i dettagli delle infrastrutture di rete e degli scenari simulati.

Basandosi su questi criteri, il CIC ha realizzato CIC-IDS2017 [53], un dataset per il rilevamento delle intrusioni aggiornato e contenente le più comuni tipologie di attacco.

Il traffico di rete raccolto è costituito da comunicazioni che coprono diversi protocolli come HTTP, FTP e SSH, in modo da genere un traffico benigno diversificato e attendibile. Il dataset include anche una varietà di attacchi come Botnet, Brute Force, (D)DoS, Heartbleed, Infiltration e attacchi di tipo Web. L'acquisizione del traffico ha avuto una durata di 5 giorni, dalle ore 09:00 di lunedì 3 luglio 2017 fino alle ore 17:00 di venerdì 7 luglio 2017. Il lunedì comprende solo traffico benigno. Nei giorni successivi sono stati condotti i vari attacchi e generata la restante parte del traffico benigno.

Basandosi su quanto detto, quindi, per gli esperimenti condotti si è deciso di adottare il dataset CIC-IDS2017, o meglio, una sua versione corretta come si vedrà a breve.

Nonostante gli sforzi impiegati per la sua realizzazione, tale dataset non è esente da problematiche. Si procede quindi alla rassegna delle criticità riscontrate e alle soluzioni intraprese per poi procedere con la descrizione dei profili di attacco presenti nel dataset.

3.1.1 Problematiche del dataset

Il dataset CIC-IDS2017 viene spesso utilizzato per valutare le prestazioni degli IDS grazie alle sue proprietà precedentemente descritte. Nonostante ciò, a seguito di alcune analisi condotte sul dataset è emerso che sono presenti diverse criticità. Se non gestite correttamente, queste imperfezioni possono portare ad una errata interpretazione dei risultati ottenuti attraverso una valutazione sperimentale [54]. Infatti, si è visto che alcune di queste problematiche possano causare in maniera subdola l'overfitting degli algoritmi di machine learning.

Gli autori del dataset forniscono sia i dati di rete grezzi sottoforma di file .pcap, sia una descrizione dei flussi di rete sottoforma di file .csv ottenuti attraverso l'estrazione di feature dal traffico di rete. Le caratteristiche estratte rappresentano informazioni rilevanti sul flusso di rete avvenuto tra una sorgente e una destinazione. Tra queste possiamo osservare l'indirizzo IP sorgente e destinazione, così come il numero di porta, il numero di pacchetto e la durata del flusso.

Le criticità introdotte dipendono da diversi fattori [55]. L'estrazione delle feature a partire dai dati grezzi è avvenuta attraverso il software CICFlowMeter. La versione del software utilizzata era affetta da bug e ha influenzato il processo di estrazione delle feature scaturendo in una scorretta descrizione dei flussi di rete, timestamp incoerenti, duplicazione di flussi e l'omissione di un attacco. Altre imperfezioni sono sorte a causa di un impreciso processo di assegnazione delle etichette ai flussi di rete. Gli autori del dataset forniscono la relativa documentazione in cui descrivono le tipologie di traffico in corso durante le varie fasce orarie. Si è visto che gli intervalli temporali forniti non sono precisi e ciò ha portato ad associare alcuni flussi malevoli alla tipologia di attacco sbagliata. Il caso peggiore è quello in cui alcuni flussi malevoli vengono etichettati come benigni e viceversa. Si evince, quindi, che un processo di etichettatura basato su fasce orarie può risultare problematico se non viene effettuato con cura.

Gli autori di uno studio in letteratura [56] hanno applicato delle modifiche correttive al software CICFlowMeter seguite da una fase di riassegnazione delle etichette corrette ai vari flussi. Per la valutazione sperimentale del sistema proposto si è deciso di adottare la versione corretta più recente del dataset [57] giacché le critiche sollevate risultano valide e potrebbero influenzare negativamente gli esperimenti condotti.

3.1.2 Descrizione degli attacchi

Vengono ora descritti i profili di attacco presenti nel dataset. Questi profili sono stati realizzati considerando la lista delle famiglie di attacco più comuni e forniscono scenari di attacco sufficientemente diversificati.

Botnet attack

Gli attacchi botnet rappresentano una minaccia significativa per il panorama della sicurezza informatica, in quanto sfruttano reti di computer compromessi per svolgere attività dannose gestite da un'unica entità. Queste reti, note come botnet, vengono create attraverso l'infezione

di computer attraverso i malware, che consentono il controllo remoto da parte dell'attaccante. Le macchine compromesse, o bot, comunicano con un server centrale di comando e controllo (C&C) [58], che ne dirige le azioni. Le reti bot sono strumenti versatili, in grado di eseguire attacchi DDoS (distributed denial-of-service), spam, furto di credenziali e altro ancora. Il miglioramento di questi attacchi spesso comporta un continuo adattamento per eludere il rilevamento, sottolineando la necessità di solide misure di sicurezza. La lotta alle minacce delle botnet richiede un approccio su più fronti, che comprende aggiornamenti regolari dei sistemi, solidi protocolli di sicurezza e sforzi collaborativi all'interno della comunità della sicurezza informatica per identificare, tracciare e smantellare queste reti dannose.

Brute force attack

Gli attacchi a forza bruta rappresentano un metodo semplice ma potente utilizzato dai criminali informatici per ottenere l'accesso non autorizzato a sistemi, account o reti. In un attacco a forza bruta, un aggressore tenta sistematicamente tutte le possibili combinazioni di password [59] o chiavi crittografiche fino a scoprire quella corretta. Questo metodo si basa sul presupposto che le misure di sicurezza del sistema preso di mira non siano sufficienti a rilevare o impedire tentativi ripetuti di accesso. Questi attacchi possono richiedere molto tempo, ma sono efficaci contro password deboli o facilmente indovinabili. Per aumentare le possibilità di successo, gli aggressori possono utilizzare strumenti automatici che generano e provano rapidamente numerose combinazioni di password. Per difendersi dagli attacchi a forza bruta è necessario implementare politiche di password robuste, impiegare meccanismi di blocco dell'account dopo un certo numero di tentativi di accesso falliti e incorporare livelli di sicurezza aggiuntivi, come l'autenticazione a più fattori, per rafforzare la resilienza complessiva dei sistemi contro i tentativi di accesso non autorizzati.

Denial of Service attack

Gli attacchi Denial-of-Service (DoS) costituiscono tentativi malevoli di interrompere il regolare funzionamento di un sistema informatico, di una rete o di un servizio online sovraccaricandolo di una quantità eccessiva di traffico. In un attacco DoS, l'aggressore cerca di esaurire le risorse dell'obiettivo o di sfruttarne le vulnerabilità, causando un'interruzione del servizio e rendendolo inaccessibile agli utenti legittimi. A differenza degli attacchi DDoS (Distributed Denial-of-Service), che coinvolgono più sistemi compromessi, un attacco DoS tradizionale è

in genere condotto da un'unica fonte. Le tecniche più comuni prevedono l'inondazione del bersaglio di traffico, lo sfruttamento dei punti deboli della rete o l'utilizzo di altri mezzi per esaurire le risorse del sistema. L'impatto di un attacco DoS eseguito con successo può variare da interruzioni temporanee a tempi di inattività prolungati, sottolineando l'importanza di implementare misure di sicurezza, come firewall e sistemi di rilevamento delle intrusioni, per difendersi da questi tentativi deliberati di interrompere i servizi online.

Distributed Denial of Service attack

Gli attacchi Distributed Denial-of-Service (DDoS) rappresentano una forma pervasiva di minaccia informatica in cui più computer compromessi, organizzati in una botnet, vengono manipolati per inondare una rete, un sistema o un sito web bersaglio con un volume di traffico eccessivo. Questa tipologia di attacco è sostanzialmente una variante degli attacchi DoS. L'obiettivo è quello di esaurire le risorse e la larghezza di banda della vittima, rendendo i suoi servizi online inaccessibili agli utenti legittimi [60]. Gli attacchi DDoS si presentano in varie forme, tra cui attacchi volumetrici che inondano le reti con un traffico massiccio, attacchi al protocollo che sfruttano le vulnerabilità dei protocolli di rete e attacchi di livello applicativo che prendono di mira applicazioni o servizi specifici. Questi attacchi possono avere gravi conseguenze, causando perdite finanziarie, danni alla reputazione e interruzioni di servizi online critici. La mitigazione degli attacchi DDoS comporta l'implementazione di solide misure di sicurezza, come il filtraggio del traffico, i sistemi di prevenzione delle intrusioni e le reti di distribuzione dei contenuti, per rilevare e deviare il traffico dannoso. Inoltre, la collaborazione tra professionisti della sicurezza informatica e fornitori di servizi Internet è essenziale per contrastare efficacemente la portata e la complessità degli attacchi DDoS.

Heartbleed attack

Heartbleed [61] è una vulnerabilità di sicurezza critica scoperta nel 2014 nella libreria software OpenSSL. Questa falla permetteva agli aggressori di sfruttare un bug nell'implementazione del protocollo Transport Layer Security (TLS), consentendo loro di recuperare dati sensibili dalla memoria di un server, tra cui nomi utente, password e chiavi crittografiche. Il nome "Heartbleed" si riferisce all'estensione heartbeat di OpenSSL che era suscettibile a questo exploit. Questa vulnerabilità ha colpito una parte significativa di Internet, poiché molti siti web e servizi online utilizzano OpenSSL per proteggere le loro comunicazioni. La natura diffusa di Heartbleed ha

sottolineato l'importanza di affrontare e correggere tempestivamente le vulnerabilità di sicurezza. In seguito alla sua scoperta, i siti web e le organizzazioni colpite hanno rapidamente aggiornato i loro sistemi e l'incidente ha evidenziato la necessità di migliorare le pratiche di sviluppo del software open-source e di sicurezza informatica per evitare che vulnerabilità simili compromettano la sicurezza digitale.

Infiltration attack

Gli attacchi di infiltrazione sono una categoria di minacce informatiche caratterizzate dall'accesso non autorizzato e dalla penetrazione in sistemi, reti o applicazioni informatiche con l'intento di compromettere la riservatezza, l'integrità o la disponibilità dei dati. Questi attacchi possono assumere varie forme, tra cui lo sfruttamento di vulnerabilità nel software come le backdoor, l'utilizzo di malware per prendere il controllo o l'impiego di tecniche di ingegneria sociale per ingannare le persone e far loro rivelare informazioni sensibili [62]. L'obiettivo principale degli attacchi di infiltrazione è quello di entrare clandestinamente in un sistema bersaglio, spesso rimanendo inosservati per un lungo periodo, ed estrarre dati preziosi o stabilire una presenza persistente per ulteriori attività dannose. Una difesa efficace contro gli attacchi di infiltrazione prevede l'implementazione di solide misure di cybersecurity, l'aggiornamento periodico del software, la conduzione di audit di sicurezza e la formazione del personale in materia di cybersecurity per aumentarne la consapevolezza e la resilienza contro l'evoluzione delle tattiche di infiltrazione impiegate dagli avversari informatici.

Portscan attack

Gli attacchi portscan sono una tecnica di ricognizione comune utilizzata dai criminali informatici per identificare i punti di ingresso vulnerabili in un sistema o in una rete. Durante un portscan, l'aggressore esegue una scansione sistematica di una serie di porte di rete su un sistema bersaglio per scoprire quali sono aperte e potenzialmente accessibili [63]. Identificando le porte aperte, gli aggressori possono ottenere informazioni sui servizi e sulle applicazioni in esecuzione sul sistema preso di mira, il che aiuta a pianificare ulteriori attacchi. Gli attacchi portscan possono essere furtivi o aggressivi: alcuni attaccanti utilizzano scansioni lente per evitare il rilevamento, mentre altri optano per scansioni più rapide per identificare rapidamente i punti deboli. Per difendersi dagli attacchi portscan è necessario implementare firewall, sistemi di rilevamento delle intrusioni e monitorare regolarmente l'attività di rete per rilevare e rispondere

a comportamenti di scansione sospetti, contribuendo a rafforzare la sicurezza generale di un sistema o di una rete.

Web attack

Gli attacchi Web comprendono un'ampia gamma di minacce informatiche specificamente progettate per sfruttare le vulnerabilità dei sistemi, delle applicazioni e dei servizi basati sul Web. Questi attacchi prendono di mira vari livelli dello stack web, tra cui il livello applicativo, i protocolli di rete e l'infrastruttura del server. Gli attacchi web più comuni includono le SQL Injection, in cui gli aggressori manipolano i campi di input per eseguire query di database non autorizzate, e il cross-site scripting (XSS), che consente l'esecuzione di script dannosi nel contesto del browser dell'utente [64]. Altre tecniche prevedono lo sfruttamento di upload di file non sicuri, la compromissione dei meccanismi di autenticazione o il lancio di attacchi DDoS (Distributed Denial of Service) contro i server Web. Gli attacchi Web comportano rischi significativi in quanto possono portare ad accessi non autorizzati, violazioni dei dati e interruzioni del servizio. La mitigazione di queste minacce richiede un approccio completo, che preveda pratiche di codifica sicure, valutazioni regolari della sicurezza, firewall per applicazioni web e un monitoraggio diligente per rilevare e rispondere tempestivamente a potenziali incidenti di sicurezza basati sul web.

Zero-day attack

A causa della loro natura, non si è in possesso di dati che rappresentino specificatamente attacchi zero-day poiché non è possibile stabilire a priori quale sarà un attacco escogitato da un attaccante. D'altro canto, qualsiasi tipologia di attacco non precedentemente osservata dal sistema durante la fase di addestramento è considerabile un attacco zero-day.

Un modo per simulare un attacco zero-day, quindi, è quello di escludere le istanze appartenenti ad una classe di attacco dall'insieme di addestramento e proporle al sistema solo durante la fase di testing. Questo approccio risulta convincente poiché si valuta il sistema su traffico che sappiamo essere attendibile e reale. Ciò permette di valutare le reali capacità di rilevamento del sistema degli attacchi zero-day, a differenza di quanto avverrebbe se si pensasse di valutare il sistema su traffico generato in maniera casuale. In altre parole, i valori generati per il traffico potrebbero non essere possibili in un contesto reale.

Classe	N. campioni
Benign	1582566
Botnet	736
FTP-Patator	3972
SSH-Patator	2961
DDoS	95144
DoS GoldenEye	7567
DoS Hulk	158468
DoS Slowhhtptest	1740
DoS Slowloris	3859
Heartbleed	11
Infiltration	36
Portscan	230833
Web attack - Brute force	73
Web attack - SQL Injection	13
Web attack - XSS	18

Tabella 3.1: Numero di campioni per classe del dataset

Per ottenere risultati più significativi si è deciso di ripetere più volte gli esperimenti. In ogni esperimento si esclude una classe di attacco dall'insieme di addestramento. Questo permette di avere una valutazione complessiva più significativa in quanto permette di apprendere anche come il sistema si comporta in presenza di diverse tipologie di attacchi zero-day.

3.1.3 Preprocessing del dataset

Una volta che sono state individuate e corrette le criticità del dataset si procede alla fase di preprocessing dei dati, attraverso la quale è possibile filtrare gli stessi e riorganizzarli in modo tale che meglio si prestino ad essere utilizzati per la valutazione sperimentale.

Il dataset corretto fornito da [57] è costituito da 2087997 campioni e da un insieme di 90 feature. La tabella 3.1 mostra le classi del dataset e il relativo numero di campioni.

Si procede innanzitutto con una fase di filtraggio dei dati che permette la rimozione di campioni duplicati o contenenti valori mancanti. Il dataset presenta dimensionalità elevate e alcune delle feature risultano essere ridondanti o non informative. Ci si è quindi dedicati alla

fase di selezione delle feature più significative. In prima battuta sono state eliminate le feature relative all'id del flusso, all'indirizzo IP sorgente e destinazione, al numero di porta sorgente e destinazione e al timestamp. Questo viene fatto per prevenire che l'IDS associ un timestamp o un utente specifico ad una tipologia di attacco senza apprendere il problema sottostante. Questo fenomeno è in genere chiamato *shortcut learning*. Successivamente si procede con la fase di selezione delle feature vera a propria. A tal proposito viene utilizzato un approccio basato sulla mutua informazione [65]. In particolare, per ciascuna feature viene calcolato un valore che descrive quanto quella feature sia discriminante per individuare l'etichetta corretta. Una volta ottenuti e analizzati i valori, sono state selezionate le feature con i valori più alti. L'insieme delle caratteristiche è ora costituito da 30 feature.

Il dataset risulta essere anche notevolmente sbilanciato mostrando una predominanza di campioni appartenenti alla classe benigna con 1582566 istanze. Inoltre, come si può vedere dalla tabella, alcune classi di attacco presentano un numero irrisorio di campioni. Questo sbilanciamento è problematico per diverse ragioni:

- in situazioni in cui una classe supera notevolmente le altre, gli algoritmi di apprendimento possono mostrare un bias, favorendo la classe maggioritaria [66];
- quando il numero di campioni per una classe tende ad aumentare esageratamente, le prestazioni dei modelli di machine learning peggiorano a causa della loro tendenza all'overfitting;
- un numero troppo ridotto di campioni non permette al sistema di apprendere con esattezza le caratteristiche di una specifica classe e a generalizzare correttamente.

Il problema è stato affrontato su diversi fronti. Innanzitutto, sono state raggruppate alcune tipologie di attacco così da avere delle macro-classi di attacco. In particolare le classi *FTP-Patator* e *SSH-Patator* sono state sostituite dalla classe *Brute force*; le classi relative agli attacchi di denial of service *DDoS*, *DoS GoldenEye*, *DoS Hulk*, *DoS Slowhttptest*, *DoS Slowloris* sono state raggruppate in un'unica classe denominata *(D)DoS*; lo stesso passaggio è stato effettuato per le classi *Web attack - Brute force*, *Web attack - SQL Injection*, *Web attack - XSS* che vengono ora rappresentate dalla classe *Web attack*.

Successivamente le classi maggioritarie sono state sotto-campionate e le classi minoritarie sono state sovra-campionate. Il sottocampionamento è stato effettuato attraverso un campionamento casuale dei campioni di ciascuna classe. Per il sovra-campionamento è stato utilizzato

Classe	N. campioni
Benign	40000
Botnet	20000
Brute force	20000
(D)DoS	20000
Heartbleed	20000
Infiltration	20000
Portscan	20000
Web attack	20000

Tabella 3.2: Numero di campioni per classe dopo la fase di preprocessing

SMOTE [67]. L'acronimo SMOTE sta per Synthetic Minority Over-sampling Technique ed è una tecnica fondamentale nell'ambito del machine learning progettata per mitigare le sfide poste dallo sbilanciamento delle classi in un dataset. SMOTE affronta questo problema generando istanze sintetiche per le classi minoritarie. Questo processo aiuta a ottenere una distribuzione più bilanciata, migliorando alla fine la capacità del modello di generalizzare e fare previsioni accurate sia per le classi maggioritarie che per quelle minoritarie. Sia per l'operazione di sottocampionamento che per quella di sovracampionamento, il numero di campioni per le classi di attacco è stato fissato a 20000. Tale valore viene scelto per non penalizzare eccessivamente le classi maggioritarie rischiando di escludere troppi campioni utili. Per il traffico benigno il numero di campioni viene fissato a 40000 poiché il traffico benigno è molto diversificato quindi ha senso mantenere un numero maggiore di campioni che permettono di avere un traffico più diversificato. La tabella 3.2 mostra la struttura del dataset dopo il tentativo di bilanciamento.

Infine le feature vengono normalizzate, di conseguenza ciascuna feature ha valori compresi nell'intervallo $[0, 1]$ e contribuisce allo stesso modo [68]. Si preferisce la normalizzazione alla standardizzazione poiché quest'ultima sfrutta la media per effettuare lo scaling, notoriamente suscettibile a valori anomali. La normalizzazione effettuata lo scaling di ciascun campione in maniera indipendente dagli altri evitando che i campioni anomali possano influenzare negativamente l'operazione.

3.2 Metriche

Una delle metriche maggiormente utilizzate per la valutazione dei sistemi e l'*accuracy* definita come:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{3.1}$$

dove TP, TN, FP, e FN sono rispettivamente i veri positivi, i veri negativi, i falsi positivi e i falsi negativi. In altre parole, l'accuracy è il rapporto tra il numero di predizioni corrette e il numero totale di predizioni. Per quanto l'accuracy sia una metrica semplice e di facile intuizione si ritiene di dover utilizzare ulteriori metriche per fornire un'analisi più dettagliata delle prestazioni del sistema. A tal proposito vengono impiegate la *precision* e il *recall* (definito anche come il tasso di veri positivi) definite come segue:

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

Le due metriche possono essere utilizzate per calcolare l'F_1 score:

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall} \tag{3.4}$$

con $\beta = 1$. L'F_1 score costituisce la media armonica tra precision e recall ed è un buon indicatore del bilanciamento tra le due metriche. Questa metrica è molto rilevante ed è quella a cui si fa maggiore affidamento in questa tesi. Infine, vengono anche calcolate le matrici di confusione così da poter essere mostrate e avere un'interpretazione visuale dei risultati.

3.3 Esperimenti e analisi comparativa

OMISSIS

Conclusioni

Questa tesi ha affrontato con successo la sfida cruciale di realizzare un sistema di rilevamento delle intrusioni per identificare in modo efficace sia attacchi zero-day che attacchi noti. La prima fase del lavoro svolto è stata dedicata all'approfondimento dei tradizionali approcci per la realizzazione di IDS e alle tecniche utilizzate per individuare gli attacchi zero-day, cercando di evidenziare i punti di forza e le corrispondenti limitazioni.

Questa fase preliminare ha condotto alla realizzazione di un sistema in linea con le necessità progettuali con le seguenti proprietà:

- il sistema è costituito da due livelli il cui utilizzo in maniera sequenziale permette una classificazione più efficiente dell'input;
- viene utilizzato un approccio basato su soglia congiuntamente ad un LOF per distinguere in maniera efficace gli attacchi zero-day dal traffico benigno;
- un ensemble costituito da un Decision Tree, da una Random Forest e da una rete neurale si occupa della classificazione degli attacchi noti;
- un meccanismo basato su votazione pesata e meta-classificatore viene impiegato per migliorare le prestazioni di classificazione.

Le caratteristiche elencate permettono al sistema non solo di essere efficace nel rilevare gli attacchi conosciuti ma anche una varietà di possibili attacchi zero-day. Si è visto come la simulazione di attacchi zero-day attraverso l'utilizzo di attacchi noti fosse un buon approccio per la valutazione del sistema contro attacchi sconosciuti e ha mostrato come esso fosse anche affidabile al variare della natura del possibile attacco zero-day in corso.

L'utilizzo del dataset CIC-IDS2017 ha fornito un ambiente realistico e diversificato per valutare il sistema IDS proposto. La rappresentazione completa di scenari di traffico di rete del

dataset ha permesso una valutazione robusta delle capacità del sistema nel distinguere minacce precedentemente non viste e modelli di attacco ben noti. Questo dataset è più moderno e aggiornato rispetto alle alternative utilizzate dagli studi in letteratura ma non è esente da criticità. È stata quindi necessaria una fase di consultazione di possibili correzioni al dataset seguite dalle usuali procedure di preprocessing necessarie alla eliminazione di campioni non necessari, al bilanciamento del dataset e all'estrazioni di feature significative.

I risultati del lavoro svolto rivelano l'efficacia del sistema sviluppato nel fronteggiare le sfide poste dagli attacchi zero-day. La capacità del sistema di adattarsi e identificare minacce emergenti ne dimostra il potenziale nel rafforzare la sicurezza delle reti in ambienti dinamici ed in evoluzione. Inoltre, le prestazioni nel rilevare attacchi noti confermano l'affidabilità e la resistenza del sistema contro minacce consolidate. Le prestazioni del sistema sono state confrontate con quelle di uno studio individuato in letteratura i cui obiettivi progettuali erano in linea con quelli di questa tesi. Il sistema proposto supera lo studio esaminato e tale confronto ha permesso di mettere in luce alcuni aspetti critici dei sistemi proposti in letteratura e che una metodologia di sviluppo superficiale possa non rispecchiare le vere capacità del sistema. Attraverso una ricerca e sperimentazione rigorose, gli obiettivi stabiliti all'inizio di questo studio sono stati raggiunti.

Gli avanzamenti compiuti in questa ricerca possono essere utili nel migliorare la sicurezza di reti e sistemi, specialmente di fronte a minacce informatiche in costante evoluzione.

Nonostante l'IDS proposto offra delle prestazioni soddisfacenti è possibile individuare un paio di aree in cui si potrebbe intervenire per eventuali lavori futuri. Durante la fase di manipolazione del dataset si è visto come alcune classi di attacco siano state unificate per formare delle macrocategorie di attacco. Si potrebbe dunque pensare di aggiungere un ulteriore livello classificazione che permetta di ottenere la tipologia di attacco specifica per una particolare macro-classe. Negli esperimenti il sistema è stato in grado di rilevare una vasta quantità di attacchi con una percentuale di falsi positivi accettabile. Si potrebbe pensare di intervenire in questo ambito cercando di ridurre ulteriormente, per quanto possibile, i falsi allarmi.

Poiché il panorama digitale continua a evolversi, la necessità di sistemi di rilevamento delle intrusioni robusti e adattabili diventa sempre più fondamentale. Ricerche future potranno costruire su questi risultati, esplorando vie per ulteriori miglioramenti, scalabilità e integrazione con le tecnologie emergenti.

In conclusione, i successi di questa tesi sottolineano l'importanza dell'innovazione continua nel campo della rilevazione delle intrusioni, fornendo una base per misure di sicurezza

informatica più resilienti ed efficaci nella continua battaglia contro le minacce informatiche.

Elenco delle figure

Elenco delle tabelle

Bibliografia

[1] E. Custodio e G. Tumibay. «Cybersecurity for higher education institutions: adopting regulatory framework». In: *Global Journal of Engineering and Technology Advances* 2 (mar. 2020), pp. 016–021.

[2] J. Jang-Jaccard e S. Nepal. «A survey of emerging threats in cybersecurity». In: *Journal of Computer and System Sciences* 80 (ago. 2014).

[3] *Cisco Annual Internet Report - Cisco Annual Internet Report (2018–2023) White Paper — cisco.com*. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html. [Accessed 08-02-2024].

[4] R. Ahmad, I. Alsmadi, W. Alhamdani e L. Tawalbeh. «Zero-day attack detection: a systematic literature review». In: *Artificial Intelligence Review* 56.10 (feb. 2023), pp. 10733–10811. URL: http://dx.doi.org/10.1007/s10462-023-10437-z.

[5] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota e T. E. Boult. «Toward Open Set Recognition». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.7 (2013), pp. 1757–1772.

[6] A. Thakkar e R. Lohiya. «A survey on intrusion detection system: feature selection, model, performance measures, application perspective, challenges, and future research directions». In: *Artificial Intelligence Review* 55.1 (lug. 2021), pp. 453–563. URL: http://dx.doi.org/10.1007/s10462-021-10037-9.

[7] R. Islam e J. Abawajy. «A multi-tier phishing detection and filtering approach». In: *Journal of Network and Computer Applications* 36.1 (2013), pp. 324–335. URL: https://www.sciencedirect.com/science/article/pii/S1084804512001397.

[8] K. Sai Kiran, R. K. Devisetty, N. P. Kalyan, K. Mukundini e R. Karthi. «Building a Intrusion Detection System for IoT Environment using Machine Learning Techniques». In: *Procedia Computer Science* 171 (2020). Third International Conference on Computing and Network Communications (CoCoNet'19), pp. 2372–2379. URL: https://www.sciencedirect.com/science/article/pii/S1877050920312497.

[9] V. Agate, A. De Paola, P. Ferraro, G. Lo Re e M. Morana. «SecureBallot: A secure open source e-Voting system». In: *Journal of Network and Computer Applications* 191 (2021), p. 103165.

[10] U. S. Musa, M. Chhabra, A. Ali e M. Kaur. «Intrusion Detection System using Machine Learning Techniques: A Review». In: *2020 International Conference on Smart Electronics and Communication (ICOSEC)*. 2020, pp. 149–155.

[11] N. Haq, M. Avishek, F. Shah, A. Onik, M. Rafni e D. Md. «Application of Machine Learning Approaches in Intrusion Detection System: A Survey». In: *International Journal of Advanced Research in Artificial Intelligence* 4 (mar. 2015).

[12] A. Ahmim, L. Maglaras, M. A. Ferrag, M. Derdour e H. Janicke. «A Novel Hierarchical Intrusion Detection System Based on Decision Tree and Rules-Based Models». In: *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. 2019, pp. 228–233.

[13] I. S. Thaseen e C. Kumar. «Intrusion detection model using fusion of PCA and optimized SVM». In: *2014 International Conference on Contemporary Computing and Informatics (IC3I)*. 2014, pp. 879–884.

[14] A. P. Muniyandi, R. Rajeswari e R. Rajaram. «Network Anomaly Detection by Cascading K-Means Clustering and C4.5 Decision Tree algorithm». In: *Procedia Engineering* 30 (2012). International Conference on Communication Technology and System Design 2011, pp. 174–182. URL: https://www.sciencedirect.com/science/article/pii/S1877705812008594.

[15] L. Rokach. «Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography». In: 53 (ott. 2009), pp. 4046–4072.

[16] L. Hansen e P. Salamon. «Neural network ensembles». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12.10 (1990), pp. 993–1001.

[17] R. Polikar. «Polikar, R.: Ensemble based systems in decision making. IEEE Circuit Syst. Mag. 6, 21-45». In: *Circuits and Systems Magazine, IEEE* 6 (ott. 2006), pp. 21–45.

[18] T. Dietterich. «An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization». In: *Mach. Learn.* 40 (dic. 2000).

[19] M. A. Ganaie, M. Hu, A. Malik, M. Tanveer e P. Suganthan. «Ensemble deep learning: A review». In: *Engineering Applications of Artificial Intelligence* 115 (2022), p. 105151.

[20] V. Agate, S. Drago, P. Ferraro e G. Lo Re. «Anomaly Detection for Reoccurring Concept Drift in Smart Environments». In: *2022 18th International Conference on Mobility, Sensing and Networking (MSN)*. IEEE. 2022, pp. 113–120.

[21] A. Nassif, M. Abu Talib, Q. Nasir e F. Dakalbab. «Machine Learning for Anomaly Detection: A Systematic Review». In: *IEEE Access* PP (mag. 2021), pp. 1–1.

[22] A. Khraisat, I. Gondal, P. Vamplew e J. Kamruzzaman. «Survey of intrusion detection systems: techniques, datasets and challenges». In: *Cybersecurity* 2 (dic. 2019).

[23] N. Ye, S. Emran, Q. Chen e S. Vilbert. «Multivariate statistical analysis of audit trails for host-based intrusion detection». In: *IEEE Transactions on Computers* 51.7 (2002), pp. 810–820.

[24] Q. Wu e Z. Shao. «Network Anomaly Detection Using Time Series Analysis». In: nov. 2005, pp. 42–42.

[25] Y. Laarouchi, M. Kaaniche, V. Nicomette, I. Studnia e E. Alata. «A language-based intrusion detection approach for automotive embedded networks». In: *International Journal of Embedded Systems* 10 (gen. 2018), p. 1.

[26] L. Bouzar-Benlabiod, L. Meziani, A. Chebieb, N.-E. Rim e Z. Mellal. «Experts' Knowledge Merging to Reduce IDS Alerts Number». In: *2016 International Conference on Collaboration Technologies and Systems (CTS)*. 2016, pp. 418–423.

[27] T. Zoppi, A. Ceccarelli, T. Capecchi e A. Bondavalli. «Unsupervised anomaly detectors to detect intrusions in the current threat landscape». In: *ACM/IMS Transactions on Data Science* 2.2 (2021), pp. 1–26.

[28] Z. Muda, W. Mohamed, m. n. Sulaiman e N. Udzir. «K-Means Clustering and Naive Bayes Classification for Intrusion Detection». In: *Journal of IT in Asia* 4 (apr. 2016), pp. 13–25.

[29] Z. Chen, C. K. Yeo, B. S. Lee e C. T. Lau. «Autoencoder-based network anomaly detection». In: *2018 Wireless Telecommunications Symposium (WTS)*. 2018, pp. 1–5.

[30] A. George. «Anomaly Detection based on Machine Learning Dimensionality Reduction using PCA and Classification using SVM». In: *International Journal of Computer Applications* 47 (giu. 2012), pp. 5–8.

[31] K. El-Khatib. «Impact of Feature Reduction on the Efficiency of Wireless Intrusion Detection Systems». In: *IEEE Trans. Parallel Distrib. Syst.* 21 (ago. 2010), pp. 1143–1149.

[32] V. Agate, A. De Paola, S. Drago, P. Ferraro e G. Lo Re. «Enhancing IoT Network Security with Concept Drift-Aware Unsupervised Threat Detection». In: *2024 IEEE Symposium on Computers and Communications (ISCC)*. IEEE. 2024.

[33] M. Elsayed, N.-A. Le-Khac, S. Dev e A. Jurcut. «Network Anomaly Detection Using LSTM Based Autoencoder». In: nov. 2020.

[34] M. Verkerken, L. D'hooge, D. Sudyana, Y.-D. Lin, T. Wauters, B. Volckaert e F. De Turck. «A Novel Multi-Stage Approach for Hierarchical Intrusion Detection». In: *IEEE Transactions on Network and Service Management* 20.3 (2023), pp. 3915–3929.

[35] S. Kim, C. Hwang e T. Lee. «Anomaly Based Unknown Intrusion Detection in Endpoint Environments». In: *Electronics* 9.6 (2020). URL: https://www.mdpi.com/2079-9292/9/6/1022.

[36] M. Hassen e P. K. Chan. «Learning a Neural-network-based Representation for Open Set Recognition». In: *Proceedings of the 2020 SIAM International Conference on Data Mining (SDM)*, pp. 154–162. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611976236.18. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611976236.18.

[37] M. Hassen e P. K. Chan. «Unsupervised Open Set Recognition using Adversarial Autoencoders». In: *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 2020, pp. 360–365.

[38] Z. Zhang, Q. Liu, S. Qiu, S. Zhou e C. Zhang. «Unknown Attack Detection Based on Zero-Shot Learning». In: *IEEE Access* 8 (2020), pp. 193981–193991.

[39] A. De Paola, S. Drago, P. Ferraro e G. Lo Re. «Detecting Zero-Day Attacks under Concept Drift: An Online Unsupervised Threat Detection System». In: *CEUR Workshop Proceedings - ITASEC 2024*. Vol. 3731. 2024.

[40] H. Hindy, R. Atkinson, C. Tachtatzis, J.-N. Colin, E. Bayne e X. Bellekens. «Utilising Deep Learning Techniques for Effective Zero-Day Attack Detection». In: *Electronics* 9.10 (2020). URL: https://www.mdpi.com/2079-9292/9/10/1684.

[41] *ICSCA '19: Proceedings of the 2019 8th International Conference on Software and Computer Applications*. Penang, Malaysia: Association for Computing Machinery, 2019.

[42] W. Xie, S. Xu, S. Zou e J. Xi. «A System-Call Behavior Language System for Malware Detection Using A Sensitivity-Based LSTM Model». In: CSSE '20. Beijing, China: Association for Computing Machinery, 2020, pp. 112–118. URL: https://doi.org/10.1145/3403746.3403914.

[43] R. C. Aygun e A. G. Yavuz. «Network Anomaly Detection with Stochastically Improved Autoencoder Based Models». In: *2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)*. 2017, pp. 193–198.

[44] R.-H. Hwang, M.-C. Peng, C.-W. Huang, P.-C. Lin e V.-L. Nguyen. «An Unsupervised Deep Learning Model for Early Network Traffic Anomaly Detection». In: *IEEE Access* 8 (2020), pp. 30387–30399.

[45] V. Agate, F. M. D'Anna, A. De Paola, P. Ferraro, G. Lo Re e M. Morana. «A Behavior-Based Intrusion Detection System Using Ensemble Learning Techniques.» In: *CEUR Workshop Proceedings - ITASEC 2022*. 2022.

[46] M. Markou e M. Singh. «Novelty detection: A review—Part 1: Statistical Approaches». In: *Signal Processing* 83 (dic. 2003), pp. 2481–2497.

[47] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot e E. Duchesnay. «Scikit-learn: Machine Learning in Python». In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[48] M. A. Pimentel, D. A. Clifton, L. Clifton e L. Tarassenko. «A review of novelty detection». In: *Signal Processing* 99 (2014), pp. 215–249. URL: https://www.sciencedirect.com/science/article/pii/S016516841300515X.

[49] R. M. Gray. *Entropy and Information Theory*. Springer US, 2011. URL: http://dx.doi.org/10.1007/978-1-4419-7970-4.

[50] D. H. Wolpert. «Stacked generalization». In: *Neural networks* 5.2 (1992), pp. 241–259.

[51] F. k. Nakano, S. Martiello Mastelini, S. Barbon e R. Cerri. «Stacking Methods for Hierarchical Classification». In: *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 2017, pp. 289–296.

[52] D. Protic. «Review of KDD Cup '99, NSL-KDD and Kyoto 2006+ datasets». In: *Vojnotehnicki glasnik* 66 (lug. 2018), pp. 580–596.

[53] I. Sharafaldin, A. Habibi Lashkari e A. Ghorbani. «Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization». In: gen. 2018, pp. 108–116.

[54] M. Lanvin, P.-F. Gimenez, Y. Han, F. Majorczyk, L. Mé e É. Totel. «Errors in the CICIDS2017 Dataset and the Significant Differences in Detection Performances It Makes». In: *Risks and Security of Internet and Systems*. A cura di S. Kallel, M. Jmaiel, M. Zulkernine, A. Hadj Kacem, F. Cuppens e N. Cuppens. Cham: Springer Nature Switzerland, 2023, pp. 18–33.

[55] A. Rosay, E. Cheval, F. Carlier e L. Pascal. «Network Intrusion Detection: A Comprehensive Analysis of CIC-IDS2017». In: feb. 2022.

[56] G. Engelen, V. Rimmer e W. Joosen. «Troubleshooting an Intrusion Detection Dataset: the CICIDS2017 Case Study». In: mag. 2021, pp. 7–12.

[57] L. Liu, G. Engelen, T. Lynar, D. Essam e W. Joosen. «Error Prevalence in NIDS datasets: A Case Study on CIC-IDS-2017 and CSE-CIC-IDS-2018». In: ott. 2022, pp. 254–262.

[58] J. Liu, Y. Xiao, K. Ghaboosi, H. Deng e J. Zhang. «Botnet: Classification, Attacks, Detection, Tracing, and Preventive Measures». In: *EURASIP J. Wireless Comm. and Networking* 2009 (dic. 2009).

[59] M. Najafabadi, T. Khoshgoftaar, C. Kemp, N. Seliya e R. Zuech. «Machine Learning for Detecting Brute Force Attacks at the Network Level». In: nov. 2014, pp. 379–385.

[60] F. Lau, S. Rubin, M. Smith e L. Trajkovic. «Distributed denial of service attacks». In: vol. 3. Feb. 2000, 2275–2280 vol.3.

[61] Z. Durumeric, M. Payer, V. Paxson, J. Kasten, D. Adrian, J. Halderman, M. Bailey, F. Li, N. Weaver, J. Amann e J. Beekman. «The Matter of Heartbleed». In: nov. 2014, pp. 475–488.

[62] Y. Boshmaf, I. Muslukhov, K. Beznosov e M. Ripeanu. «Design and analysis of a social botnet». In: *Computer Networks* 57 (feb. 2013), pp. 556–578.

[63] M. Vivo, L. Ke, G. Isern e G. Vivo. «A review of port scanning techniques». In: *Computer Communication Review* 29 (apr. 1999), pp. 41–48.

[64] O. Al-Khurafi e D.-M. Alahmad. «Survey of Web Application Vulnerability Attacks». In: dic. 2015, pp. 154–158.

[65] B. Ross. «Mutual Information between Discrete and Continuous Data Sets». In: *PloS one* 9 (feb. 2014), e87357.

[66] S. Kotsiantis, D. Kanellopoulos e P. Pintelas. «Handling imbalanced datasets: A review». In: *GESTS International Transactions on Computer Science and Engineering* 30 (nov. 2005), pp. 25–36.

[67] A. ALFRHAN, R. ALHUSAIN e R. Khan. «SMOTE: Class Imbalance Problem In Intrusion Detection System». In: set. 2020, pp. 1–5.

[68] D. Singh e B. Singh. «Investigating the impact of data normalization on classification performance». In: *Applied Soft Computing* (mag. 2019), p. 105524.

[69] V. Agate, F. Concone, A. De Paola, P. Ferraro, G. Lo Re e M. Morana. «Bayesian Modeling for Differential Cryptanalysis of Block Ciphers: A DES Instance». In: *IEEE Access* 11 (2023), pp. 4809–4820.

[70] L. Breiman. «Random Forests». In: *Machine Learning* 45 (ott. 2001), pp. 5–32.

[71] K. Ting e I. Witten. «Issues in Stacked Generalization». In: *Journal of Artificial Intelligence Research* 10 (apr. 2002).

[72] L. Todorovski e S. Džeroski. «Combining Multiple Models with Meta Decision Trees». In: *Principles of Data Mining and Knowledge Discovery*. A cura di D. A. Zighed, J. Komorowski e J. Żytkow. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 54–64.

[73] V. Agate, P. Ferraro, G. Lo Re e S. K. Das. «BLIND: A privacy preserving truth discovery system for mobile crowdsensing». In: *Journal of Network and Computer Applications* (2023), p. 103811.