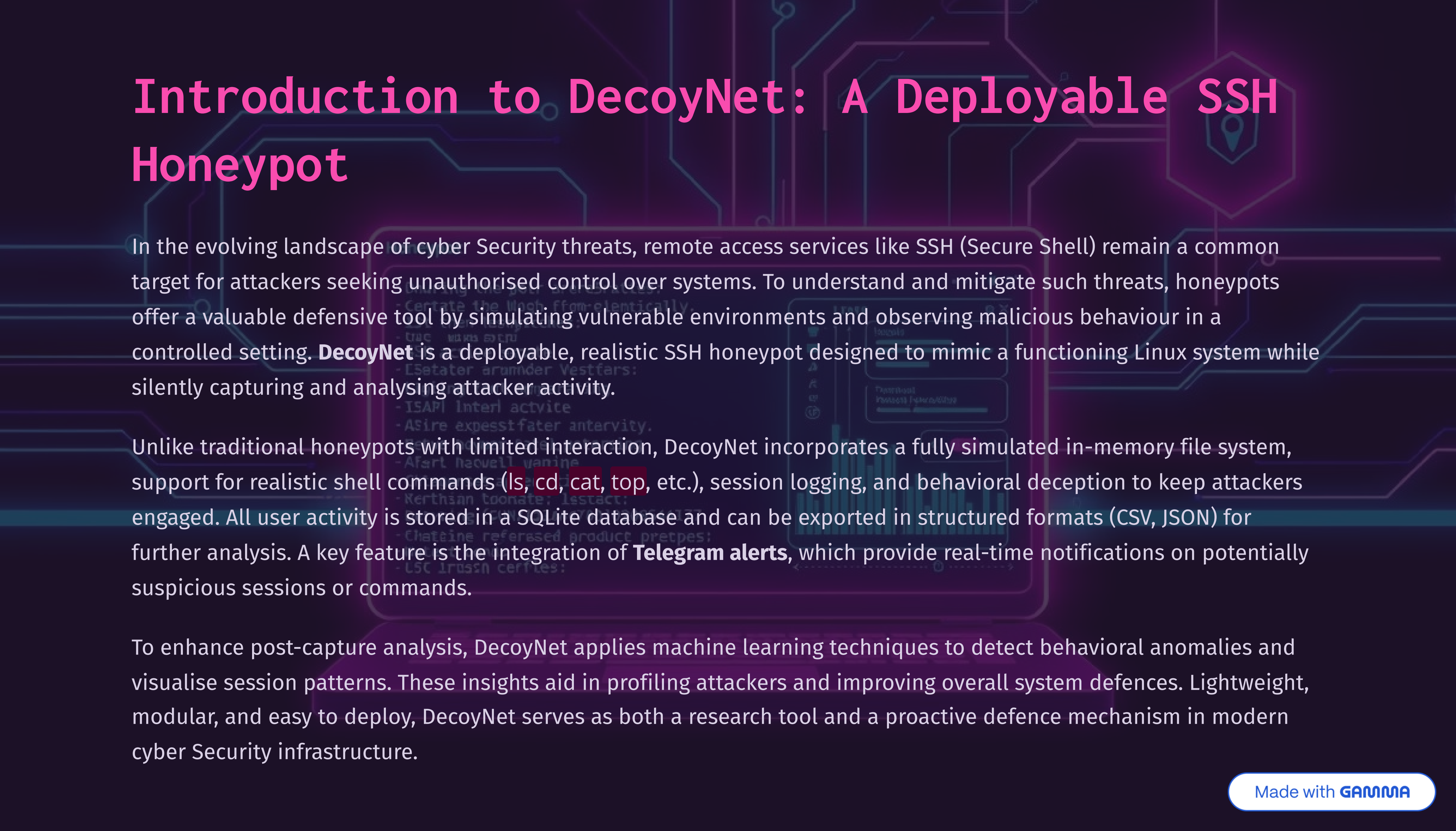

Introduction to DecoyNet: A Deployable SSH Honeypot

In the evolving landscape of cyber Security threats, remote access services like SSH (Secure Shell) remain a common target for attackers seeking unauthorised control over systems. To understand and mitigate such threats, honeypots offer a valuable defensive tool by simulating vulnerable environments and observing malicious behaviour in a controlled setting. **DecoyNet** is a deployable, realistic SSH honeypot designed to mimic a functioning Linux system while silently capturing and analysing attacker activity.

Unlike traditional honeypots with limited interaction, DecoyNet incorporates a fully simulated in-memory file system, support for realistic shell commands (`ls`, `cd`, `cat`, `top`, etc.), session logging, and behavioral deception to keep attackers engaged. All user activity is stored in a SQLite database and can be exported in structured formats (CSV, JSON) for further analysis. A key feature is the integration of **Telegram alerts**, which provide real-time notifications on potentially suspicious sessions or commands.

To enhance post-capture analysis, DecoyNet applies machine learning techniques to detect behavioral anomalies and visualise session patterns. These insights aid in profiling attackers and improving overall system defences. Lightweight, modular, and easy to deploy, DecoyNet serves as both a research tool and a proactive defence mechanism in modern cyber Security infrastructure.

Abstract

Cybersecurity threats targeting remote access services like SSH have increased rapidly, making intrusion detection and proactive monitoring critical for system administrators and researchers. Honeypots play a pivotal role in this domain by acting as decoy systems that attract and record malicious activity. **DecoyNet** is a deployable, interactive SSH honeypot developed to simulate a realistic Linux environment while logging attacker behavior for security analysis and research.

DecoyNet supports a wide range of fake yet believable shell commands (`cd`, `ls`, `cat`, `top`, etc.), maintains a fake in-memory filesystem, and offers session logging through a structured SQLite database. It provides export options in CSV and JSON formats for offline inspection. Real-time **Telegram alerts** are integrated to notify administrators of suspicious or unauthorized activity. Unlike simplistic honeypots, DecoyNet offers high interactivity and deception, making it more effective at engaging attackers.

To gain deeper insights, collected data is analyzed using machine learning techniques for anomaly detection and behavioral clustering. Visualizations and graphs are generated to reveal usage patterns, detect outliers, and enhance understanding of attack vectors. With its extensible design, DecoyNet serves as a practical tool for both learning and defending against SSH-based cyber threats.

Background on SSH Attacks

Secure Shell (SSH) is a widely used protocol for secure remote access and administration of servers. Due to its privileged access capabilities, SSH is a frequent target for brute-force attacks, credential stuffing, and automated exploit tools. Attackers often scan the internet for open SSH ports and attempt to gain unauthorised access using weak or default credentials. Once inside, they may deploy malware, exfiltrate data, or establish persistent backdoors. These attacks are often stealthy and leave minimal traces, making them hard to detect. Understanding SSH attack behaviour is crucial for building effective defences and improving network security posture.

Objectives of DecoyNet

The primary objective of **DecoyNet** is to simulate a realistic SSH environment that attracts, deceives, and logs the behavior of potential attackers. It aims to provide a high-interaction honeypot that captures session data, command histories, and behavioral patterns without compromising the real system. DecoyNet also seeks to support real-time alerting via Telegram, export detailed logs for analysis, and apply machine learning for behavioral clustering and anomaly detection. Ultimately, DecoyNet is designed to serve as both a research platform and an early-warning tool to help security teams better understand threats and strengthen SSH-based security measures.

Literature Review

Honeypots have long been used in cybersecurity as decoy systems to monitor and analyze malicious activity. Early honeypots like **Honeyd** and **Kippo** offered basic interaction and served as effective traps for low-skill attackers. Kippo, in particular, was known for simulating SSH environments and logging attacker sessions. However, it lacked extensibility and realism, making it less effective against advanced adversaries. Projects like **Cowrie** extended Kippo's capabilities by adding more realistic file systems and command emulation.

In recent years, the need for more **high-interaction honeypots** has grown as attackers use increasingly sophisticated methods. Research has emphasized the importance of behavioral logging, real-time alerting, and integration with data analysis tools to improve detection and response capabilities. Papers exploring machine learning on honeypot data highlight its utility in identifying attack patterns and anomalies.

Despite advancements, many existing solutions are either too complex to deploy or lack modularity and real-time capabilities. **DecoyNet** aims to bridge this gap by offering a lightweight, customizable, and realistic SSH honeypot with support for live notifications and ML-based analysis. It builds on the lessons of past tools but incorporates modern features needed for today's cybersecurity landscape.

Proposed Work: DecoyNet Features and Purpose

The proposed system, **DecoyNet**, is a high-interaction SSH honeypot designed to imitate a real Linux system while silently monitoring and recording the behavior of unauthorized users. Its primary purpose is to attract attackers, keep them engaged, and log their activities for further analysis without exposing any real vulnerabilities. Unlike traditional low-interaction honeypots, DecoyNet offers an immersive shell experience by supporting common commands such as `ls`, `cd`, `cat`, `top`, `whoami`, and more.

A key feature of DecoyNet is its **in-memory fake filesystem**, allowing attackers to navigate directories, view fake files, and even manipulate the environment, all without affecting the host system. This makes the experience convincing while maintaining full isolation. All commands issued by the attacker are logged into an SQLite database along with timestamps and session IDs.

Another major component is **real-time alerting via Telegram**, enabling instant notification of suspicious logins or actions. Additionally, DecoyNet includes functionality to export logs in CSV and JSON formats, making data analysis easy. To gain deeper insights, collected data can be fed into machine learning models for **anomaly detection and behavioral clustering**, helping identify patterns and deviations.

Overall, DecoyNet blends realism, security, and analytics into a single deployable honeypot platform.

Architecture

1. **SSH Listener** This is the entry point for any attacker. It mimics a standard SSH login prompt and accepts incoming SSH connections on a specified port. Once the connection is established, it initiates a fake shell session.
2. **Fake Shell Interface** This component simulates a real Linux shell. It supports common Unix commands (ls, cd, cat, etc.) and provides realistic command outputs. It also manages the fake in-memory file system structure that attackers can interact with.
3. **Logging Engine** (SQLite) Every session and command executed by the attacker is timestamped and stored in a local SQLite database. This structured logging allows efficient querying and post-analysis.
4. **Export Engine** Supports exporting logged data in CSV and JSON formats. This makes it easier to integrate with external tools or perform offline analysis using Excel or Jupyter notebooks.
5. **Real-Time Alert System (Telegram)** A Telegram bot is integrated to notify administrators instantly when suspicious activity occurs—such as login attempts, sensitive command execution (rm, passwd, etc.), or repeated brute-force access.
6. **Machine Learning Module** The stored data is later analyzed using Python-based Jupyter notebooks to apply anomaly detection and clustering algorithms, revealing attacker behavior patterns and anomalies.

Technologies Used

1. Python: The core language used for developing the DecoyNet honeypot. Python provides simplicity, cross-platform compatibility, and rich libraries for networking, database management, and system simulation.
2. Socket Programming: Used to emulate an SSH server by listening for incoming TCP connections and handling interactive sessions with attackers.
3. SQLite: A lightweight, file-based relational database used to store session data, command logs, and timestamps efficiently without requiring external database servers.
4. Telegram Bot API: Integrated for real-time alerts. The Telegram Bot API is used to send notifications to the administrator upon detecting suspicious behavior or unauthorized access.
5. Jupyter Notebook (Python): Used for post-capture data analysis. Includes visualizations, anomaly detection, and clustering through ML libraries like Scikit-learn and Matplotlib.
6. Matplotlib & Seaborn: Python libraries used to generate graphs and charts for visual analysis of attacker behavior.
7. Pandas: Used for data manipulation, exporting session logs to CSV/JSON, and preparing datasets for machine learning analysis.
8. Fake FileSystem & Command Parser (Custom): A Python-based custom module that mimics realistic file structures and shell command responses to fool attackers.

Machine Learning Analysis

DecoyNet incorporates Machine Learning to analyze attacker behavior from captured SSH sessions. Using Python and Jupyter Notebooks, session logs are exported from the SQLite database into CSV format, which are then processed using Pandas for cleaning and feature extraction. Each session is transformed into a numerical representation based on command usage frequency, session duration, and command sequences.

To uncover hidden patterns, **K-Means clustering** is used to group similar attack sessions. This helped identify categories such as reconnaissance, file exploration, privilege escalation attempts, and command injection behavior. In addition, **Isolation Forest**, an anomaly detection algorithm, was applied to highlight suspicious sessions that deviate significantly from normal patterns. These include sessions with very few or highly dangerous commands, indicating automated scripts or unique attacker behavior.

Visualizations are generated using **Matplotlib** and **Seaborn**, such as heatmaps of command usage, bar charts for top commands, cluster scatter plots, and anomaly score graphs. These graphs provide clear insights into how attackers interact with the honeypot and which tactics are most common.

By integrating ML, DecoyNet transforms static logs into dynamic intelligence, aiding in both real-time threat detection and long-term behavioral profiling of attackers.

Graphical Results, Observations, and Insights

The command flow graphs reveal that most attackers initiate interactions with reconnaissance commands like **ls** to list directory contents, followed by **cat** to inspect files. The frequent use of **clear** and **exit** commands suggests attackers may be cautious, attempting to erase traces or abort suspicious sessions rapidly.

Clustering analysis identified distinct groups of sessions: automated brute force attempts characterized by repetitive sequences and short session durations, and exploratory sessions with diverse commands and longer engagement times. Anomaly detection highlighted a small set of sessions exhibiting unusual command patterns, potentially signaling novel attack techniques or manual intrusion attempts.

These insights validate DecoyNet’s ability to capture meaningful attacker behavior data and support proactive cybersecurity measures through detailed profiling and alerting mechanisms.

Conclusion and Future Work

Conclusion

DecoyNet successfully demonstrates the power of high-interaction honeypots in cyber security defence and research. By simulating a realistic Linux shell environment with a fake file system and command responses, it effectively captures and logs attacker behaviour without exposing real vulnerabilities. The integration of a real-time alert system via Telegram and an exportable logging mechanism further enhances its utility. Machine learning analysis of the session data adds another layer of intelligence, uncovering patterns and anomalies that may indicate malicious intent. Overall, DecoyNet provides a powerful, lightweight, and extensible platform for monitoring unauthorised access and conducting threat intelligence.

Future Work

While DecoyNet performs well in its current form, several enhancements are planned. First, integrating **IP geolocation** and **whois lookups** can provide more context about attackers. Second, deploying **Dockerized versions** for rapid deployment across multiple nodes can improve scalability. Future updates may also include **web dashboard integration** for real-time visualisations, **support for additional protocols** like HTTP or FTP to widen the attack surface, and **deeper deception mechanisms** like fake vulnerabilities or services. Additionally, using deep learning for behaviour classification could improve accuracy in attacker profiling and threat prediction.