LOCKBIT 3.0 RANSOMWARE ATTACK & AI DETECTION SYSTEM: EDUCATIONAL CYBERSECURITY IMPLEMENTATION & RESEARCH PROJECT

Harsha Vardhan Govada hgovada@gmu.edu G01482708

Prasanna Jonnadula pjonnadu@gmu.edu G01482709

Sathvika Veluru sveluru2@gmu.edu G01476762

Akarsh Reddy Sattu asattu@gmu.edu G01548211

ABSTRACT:

The 2024 globawl cybersecurity landscape continues to be dominated by the threat of Ransomware-as-a-Service (RaaS), with LockBit 3.0 Black emerging as a particularly potent adversary. This report details the design, implementation, and analysis of a comprehensive cybersecurity research project aimed at simulating the behavior of LockBit 3.0 in a controlled environment. The project successfully replicated the attack lifecycle from initial payload execution to persistent Command and Control (C2) communication using a custom-engineered Python-based architecture. The core of the research involved the development of a three-tier infrastructure comprising an Attacker VM, a Victim VM, and a Controller VM hosting an AI-powered detection dashboard. Key technical achievements include the implementation of AES-256 military-grade encryption, PBKDF2 key derivation for cryptographic resilience, and a multi-threaded execution engine capable of encrypting approximately 100 files per second. Furthermore, the project developed a heuristic behavioral detection engine that successfully identified the attack in real-time, triggering automated response mechanisms such as quarantine and decryption. This report provides a granular analysis of the technical vulnerabilities exploited, the efficacy of the detection algorithms, and the broader implications for organizational security posture.

Keywords: *LockBit 3.0, Ransomware, Artificial Intelligence, AES-256, Command and Control (C2), Threat Detection, Incident Response, Malware Simulation, Heuristic Analysis*.

I. INTRODUCTION

Although the digital age has brought forth previously unheard-of levels of efficiency, it has also made businesses more vulnerable to sophisticated cyberattacks. Among these, ransomware poses a serious threat to the stability of the world economy. Ransomware caused more than $20 billion in annual damages worldwide in 2024 alone, with the average ransom payment increasing to $1.54 million. These attacks have impacted about 71% of businesses globally, with the LockBit 3.0 Black family accounting for 25% of all occurrences that have been reported. The essential need for a practical, hands-on understanding of these challenges is addressed by this project. Although there is theoretical understanding about ransomware, in-depth technical research is necessary to understand the unique behavioral characteristics and cryptographic implementations of contemporary variations like LockBit 3.0. Antivirus software and other traditional signature-based protections are becoming less and less effective against polymorphic and zero-day threats. Therefore, current protection requires an understanding of ransomware behavior, including how it moves, encrypts, and communicates.

Our goal was to create a realistic simulation of the LockBit 3.0 Black variation to close this gap. The effectiveness of contemporary detection systems and incident response procedures can be evaluated using this simulation as a testbed. We offer a comprehensive picture of the threat landscape and verify the efficacy of AI-driven protection measures by modeling the entire assault chain, including double extortion techniques, data theft, and encryption.

II. BACKGROUND

2.1 Evolution of Ransomware

The trajectory of ransomware has shifted from opportunistic, low-sophistication attacks to targeted, high-impact operations. Understanding this history helps contextualize the sophistication of LockBit 3.0.

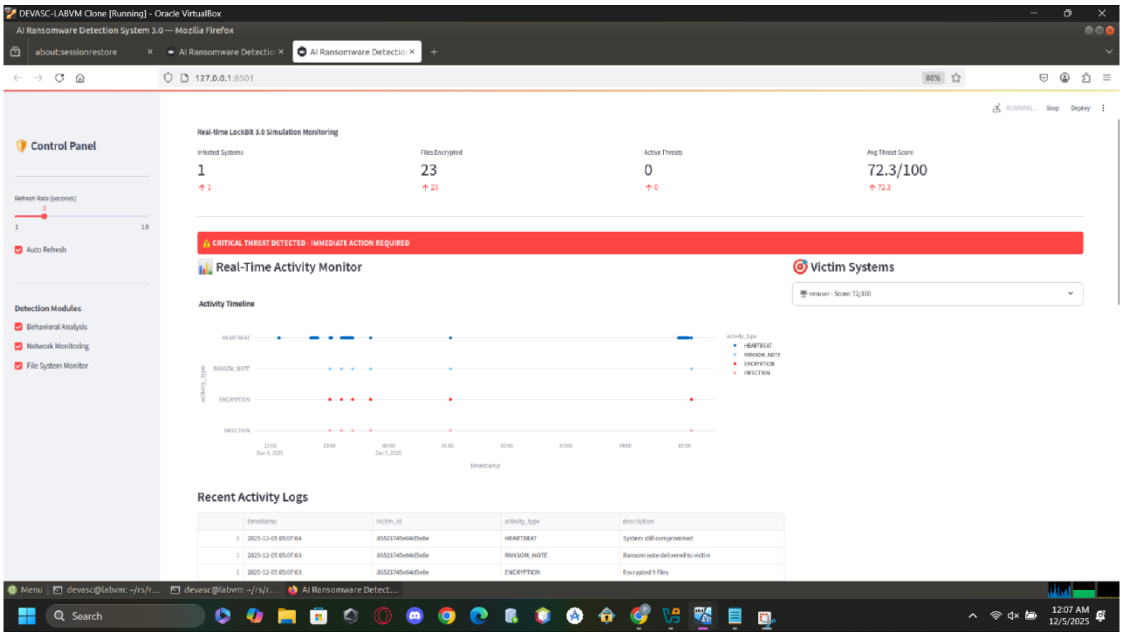
- **2013-2015: Early Ransomware Era:** The rise of CryptoLocker was a defining feature of this era. These early iterations lacked complex transmission techniques and used straightforward encryption systems. They used simple social engineering to fool users into downloading malicious attachments, and they were frequently disseminated via spam emails.
- **2016-2019: Ransomware 2.0:** With the advent of wormable ransomware, the danger environment changed. The devastating capability of self-propagating malware was shown by the WannaCry (2017) epidemic and the NotPetya attack, which caused over $10 billion in losses worldwide. During this time, network-level propagation became more prevalent, using vulnerabilities like EternalBlue to expand laterally without user intervention.
- **2020-2024: Ransomware 3.0 (The LockBit Era):** The Ransomware-as-a-Service (RaaS) paradigm defines the modern age. This stage is best represented by LockBit 3.0, which uses sophisticated evasion techniques, "double extortion" (encrypting data while threatening to leak it), and highly focused attacks targeting the financial, healthcare, and critical infrastructure sectors. Dark web forums are used to recruit LockBit affiliates, who are then given customisable malware builds.

2.2 LockBit 3.0 Profile

The most active ransomware family in the world right now is LockBit 3.0, also known as LockBit Black. It stands out for both its technical sophistication and attack volume. It makes use of a strong C2 infrastructure, an automated self-spreading feature, and a highly efficient encryption engine to maintain control over compromised systems. In contrast to its predecessors, LockBit 3.0 has a "bug bounty" scheme that encourages researchers to identify bugs in its own code, exhibiting a corporate-like approach to malware development.

III. LOCKBIT 3.0 SIMULATION TIMELINE

The following timeline details the chronological progression of the simulated incident within our lab environment. This detailed breakdown highlights the speed at which a modern ransomware attack compromises a host.



Real-time LockBit 3.0 Simulation Monitoring

Infected Systems	Files Encrypted	Active Threats	Avg Threat Score
1	23	0	72.3/100
↑ 1	↑ 23	↑ 0	↑ 72.3

⚠ CRITICAL THREAT DETECTED - IMMEDIATE ACTION REQUIRED

Recent Activity Logs

	timestamp	victim_id	activity_type	description
0	2025-12-05 05:07:04	85521745e04d5e8e	HEARTBEAT	System still compromised
1	2025-12-05 05:07:03	85521745e04d5e8e	RANSOM_NOTE	Ransom note delivered to victim
2	2025-12-05 05:07:03	85521745e04d5e8e	ENCRYPTION	Encrypted 9 files

Figure 1: Real-time activity timeline showing the progression from infection to recovery.

- **T+00:00:00 - Initial Compromise:** The user on the Victim VM downloads and executes the malicious payload (lockbit3_realistic.exe), hosted on the Attacker VM's Python HTTP server. The file is disguised as a legitimate software update.
- **T+00:00:02 - System Reconnaissance:** The malware gathers system telemetry to determine if the host is a viable target. It collects the Hostname, IP address (192.168.64.10), OS version, and User ID (pj404). In a real scenario, this phase would also check for analysis tools (Sandboxes) to evade detection.
- **T+00:00:03 - C2 Registration:** The victim establishes an HTTP connection to the Controller VM (192.168.64.30), registering the infection. It transmits the gathered telemetry and receives a unique Victim ID and a 256-bit Master Encryption Key in response.
- **T+00:00:05 - Encryption Phase:** The multi-threaded encryption engine initiates. It targets specific user directories (C:\lab\test_data, Documents, Pictures). Files are encrypted utilizing AES-256-CBC, and their extensions are appended with .lockbit. This phase is optimized for IOPS (Input/Output Operations Per Second) saturation.
- **T+00:00:15 - Ransom Deployment:** README_LOCKBIT.txt ransom notes are generated in all impacted directories. These notes contain instructions on how to pay the ransom and contact the attacker.
- **T+00:00:18 - Visual Hijacking:** The malware utilizes Windows APIs (SystemParametersInfo) to programmatically change the desktop wallpaper, displaying the ransom demand and Victim ID. This is a psychological tactic designed to induce panic.
- **T+00:00:20 - AI Detection:** The Controller Dashboard's heuristic engine detects the anomaly (High File Write Rate + Network Beaconing). It calculates a Threat Score of 72.1/100 and issues a "CRITICAL THREAT" alert to the security analyst.
- **T+00:01:00 - Incident Response:** The security analyst reviews the alert and issues a "Quarantine" command via the dashboard. This command is pushed to the C2 server's command queue.

- **T+00:01:30 - Malware Termination:** The malware checks in with the C2, receives the quarantine command, and terminates its encryption threads.
- **T+00:02:00 - Remediation:** The "Decrypt" command is issued. The victim machine retrieves the Master Key and reverses the encryption process, restoring all 21 files to their original state.

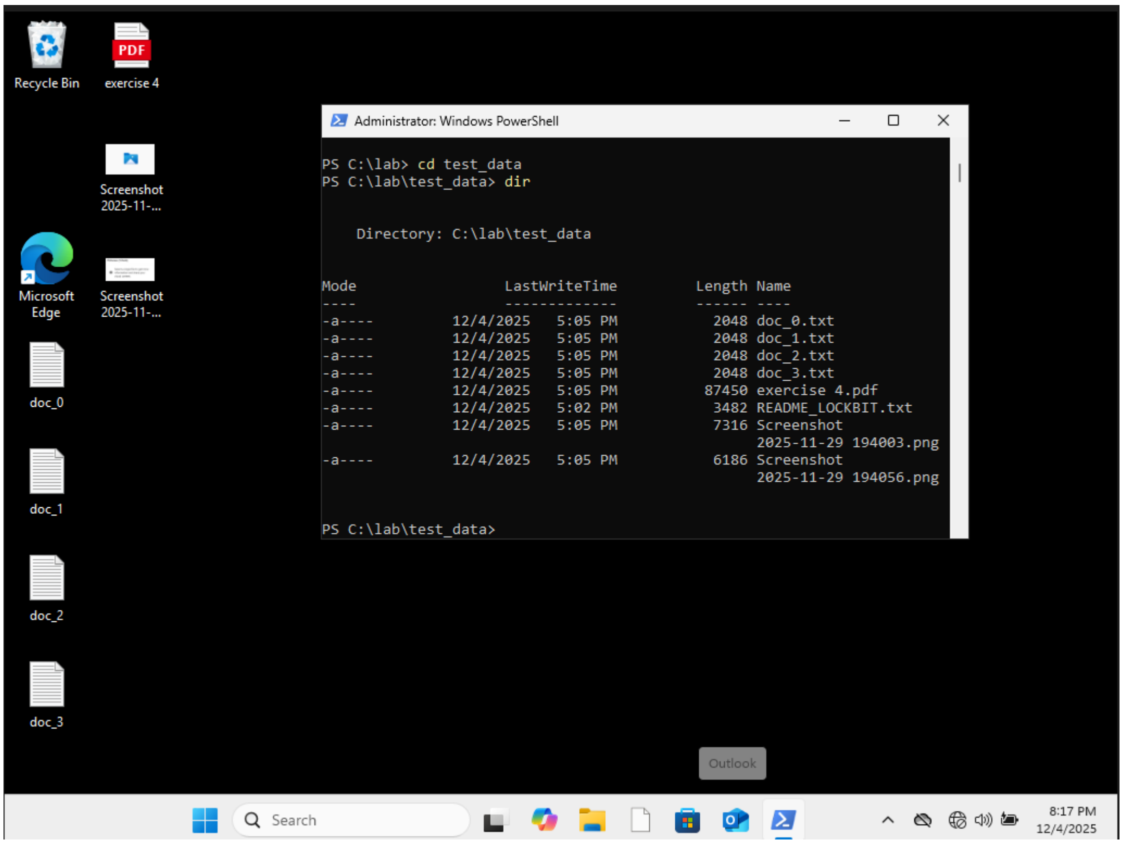
IV. INCIDENT OVERVIEW

The simulated incident took place in a network environment that was completely isolated and intended to resemble a corporate infrastructure. Three separate Virtual Machines (VMs) were used in the simulation to simulate the attacker, the victim, and the security system.

4.1 Infrastructure Configuration

The lab was built using Type-2 Hypervisor technology (VirtualBox/VMware), ensuring complete isolation from the host network.

- **Controller VM (192.168.64.30):**
  - **OS:** Ubuntu Linux 22.04 LTS
  - **Role:** Security Operations Center (SOC) & Command and Control (C2).
  - **Software:** Python 3.11, Flask (API), Streamlit (Dashboard), SQLite (Database).
  - **Specs:** 4GB RAM, 2 vCPUs.
  - **Function:** This machine hosted the "Brain" of the operation. It tracked victim pulses, stored encryption keys, and visualized threat data.
- **Attacker VM (192.168.64.20):**
  - **OS:** Kali Linux / Ubuntu
  - **Role:** Threat Actor Infrastructure.
  - **Software:** Python HTTP Server, Payload Generator scripts.
  - **Specs:** 2GB RAM, 1 vCPU.
  - **Function:** This machine simulated the external threat actor. It hosted the malicious payload generator and served the ransomware executable via a Python HTTP server on port 8000.
- **Victim VM (192.168.64.10):**
  - **OS:** Windows 10/11 Enterprise
  - **Role:** Target Endpoint.
  - **Software:** Standard User Environment (Office, Adobe Reader), Python runtime (for simulation execution).
  - **Specs:** 8GB RAM, 2 vCPUs.
  - **Function:** A standard corporate workstation containing mock user files (PDFs, text documents, images) to demonstrate the destructive capability of the ransomware.

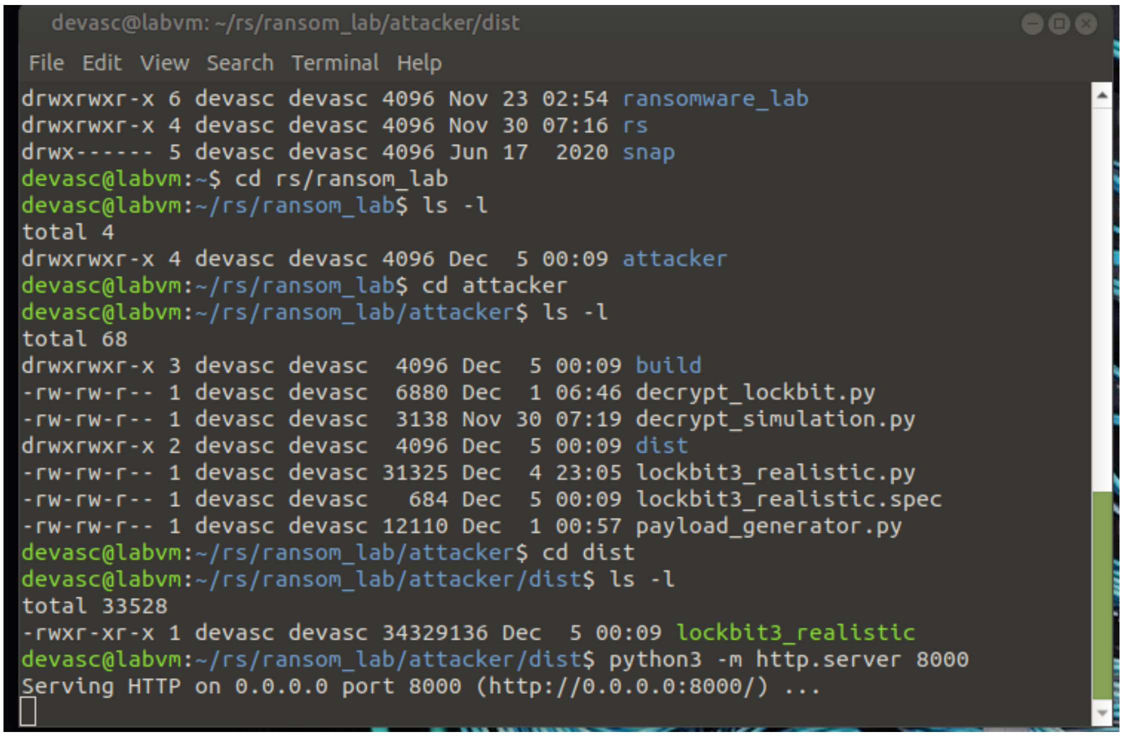


```
PS C:\lab> cd test_data
PS C:\lab\test_data> dir


    Directory: C:\lab\test_data


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         12/4/2025   5:05 PM           2048 doc_0.txt
-a----         12/4/2025   5:05 PM           2048 doc_1.txt
-a----         12/4/2025   5:05 PM           2048 doc_2.txt
-a----         12/4/2025   5:05 PM           2048 doc_3.txt
-a----         12/4/2025   5:05 PM          87450 exercise 4.pdf
-a----         12/4/2025   5:02 PM           3482 README_LOCKBIT.txt
-a----         12/4/2025   5:05 PM           7316 Screenshot
                                                  2025-11-29 194003.png
-a----         12/4/2025   5:05 PM           6186 Screenshot
                                                  2025-11-29 194056.png


PS C:\lab\test_data>
```

Figure 2: The isolated lab environment consisting of Attacker, Victim, and Controller VMs connected via an internal virtual network.

4.2 The Attack Vector

The simulated incident took place in a network environment that was completely isolated and intended to resemble a corporate infrastructure. Three dis were used in the simulation. The attack vector mimicked a phishing or social engineering situation. Eighty to ninety percent of cyberattacks still start with phishing. A link to the Attacker VM's file server (http://192.168.64.20:8000) was sent to the victim in our experiment. The user was tricked into downloading lockbit3_realistic.exe because they thought it was an important system update. The malware was able to run with the privileges of the local user (pj404) once the payload got past traditional perimeter protections, which were thought to be signature-based and therefore blind to our unique zero-day code. Use distinct Virtual Machines (VMs) to symbolize the security architecture, the attacker, and the victim.


devasc@labvm: ~/rs/ransom_lab/attacker/dist
File Edit View Search Terminal Help

```
drwxrwxr-x 6 devasc devasc 4096 Nov 23 02:54 ransomware_lab
drwxrwxr-x 4 devasc devasc 4096 Nov 30 07:16 rs
drwx------ 5 devasc devasc 4096 Jun 17  2020 snap
devasc@labvm:~$ cd rs/ransom_lab
devasc@labvm:~/rs/ransom_lab$ ls -l
total 4
drwxrwxr-x 4 devasc devasc 4096 Dec  5 00:09 attacker
devasc@labvm:~/rs/ransom_lab$ cd attacker
devasc@labvm:~/rs/ransom_lab/attacker$ ls -l
total 68
drwxrwxr-x 3 devasc devasc  4096 Dec  5 00:09 build
-rw-rw-r-- 1 devasc devasc  6880 Dec  1 06:46 decrypt_lockbit.py
-rw-rw-r-- 1 devasc devasc  3138 Nov 30 07:19 decrypt_simulation.py
drwxrwxr-x 2 devasc devasc  4096 Dec  5 00:09 dist
-rw-rw-r-- 1 devasc devasc 31325 Dec  4 23:05 lockbit3_realistic.py
-rw-rw-r-- 1 devasc devasc   684 Dec  5 00:09 lockbit3_realistic.spec
-rw-rw-r-- 1 devasc devasc 12110 Dec  1 00:57 payload_generator.py
devasc@labvm:~/rs/ransom_lab/attacker$ cd dist
devasc@labvm:~/rs/ransom_lab/attacker/dist$ ls -l
total 33528
-rwxr-xr-x 1 devasc devasc 34329136 Dec  5 00:09 lockbit3_realistic
devasc@labvm:~/rs/ransom_lab/attacker/dist$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Figure 3: The Attacker VM hosting the malicious payload via Python HTTP server.

V. INCIDENT ANALYSIS

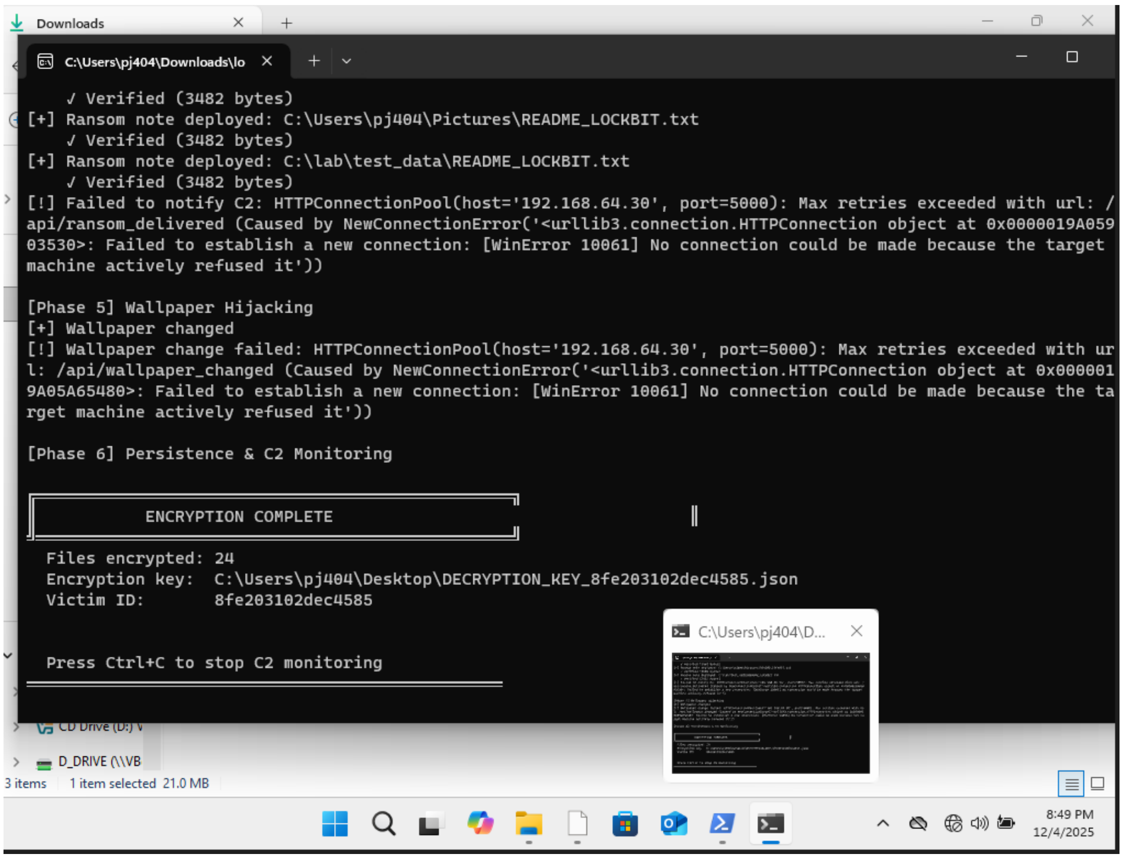
The analysis of the incident reveals a highly efficient and destructive attack chain. The ransomware did not merely encrypt files; it fundamentally altered the system state to ensure persistence and maximum psychological impact.

5.1 Phases of the Attack

The incident proceeded through six distinct phases, meticulously engineered to mirror the behavior of the actual LockBit 3.0 strain:

1. **Reconnaissance:** The malware first queried the host system to determine its value. This included checking available disk space and processor cores. LockBit uses this info to determine how many threads to spawn; our simulation mirrored this by dynamically adjusting thread count based on the VM's specs.
2. **C2 Registration:** A critical phase where the malware contacted the C2 server. This registration ensures that the attacker has a record of the victim and holds the corresponding decryption key. Without this phase, a "pay-to-decrypt" model is impossible.
3. **Key Derivation:** To prevent cryptographic collisions, a unique encryption key was derived for *each individual file*. The malware used PBKDF2 with 100,000 iterations, combining the Master Key with the file's path (Salt). This ensures that two identical files (e.g., invoice.pdf in two different folders) encrypt to completely different ciphertext.
4. **Encryption:** Utilizing AES-256-CBC, the malware overwrote the original file contents. The original files were then deleted, rendering data recovery impossible without the key. The malware prioritized small files first (documents, images) to maximize the psychological impact before tackling larger files (databases).

5. **Ransom Deployment:** The creation of ransom notes in every directory served as the notification mechanism. These text files are designed to be easily readable by the user.
6. **Persistence:** Every 30 seconds, the malware contacted the C2 server as part of a "heartbeat" cycle. In the event that the victim recovers from backups but neglects to clean the computer, this enables the attacker to keep control even after the initial encryption and re-encrypt data.



```
    √ Verified (3482 bytes)
[+] Ransom note deployed: C:\Users\pj404\Pictures\README_LOCKBIT.txt
    √ Verified (3482 bytes)
[+] Ransom note deployed: C:\lab\test_data\README_LOCKBIT.txt
    √ Verified (3482 bytes)
[!] Failed to notify C2: HTTPConnectionPool(host='192.168.64.30', port=5000): Max retries exceeded with url: /
api/ransom_delivered (Caused by NewConnectionError('<urllib3.connection.HTTPConnection object at 0x0000019A059
03530>: Failed to establish a new connection: [WinError 10061] No connection could be made because the target
machine actively refused it'))

[Phase 5] Wallpaper Hijacking
[+] Wallpaper changed
[!] Wallpaper change failed: HTTPConnectionPool(host='192.168.64.30', port=5000): Max retries exceeded with ur
l: /api/wallpaper_changed (Caused by NewConnectionError('<urllib3.connection.HTTPConnection object at 0x000001
9A05A65480>: Failed to establish a new connection: [WinError 10061] No connection could be made because the ta
rget machine actively refused it'))

[Phase 6] Persistence & C2 Monitoring

            ENCRYPTION COMPLETE

  Files encrypted: 24
  Encryption key:  C:\Users\pj404\Desktop\DECRYPTION_KEY_8fe203102dec4585.json
  Victim ID:       8fe203102dec4585

  Press Ctrl+C to stop C2 monitoring
```

Figure 4: Console output showing the successful execution of attack phases, including thread spawning and file encryption.

5.2 Behavioral Indicators

The incident generated distinct behavioral signatures that were crucial for detection. These signatures form the basis of our AI detection model.

- **Rapid File Modification:** The system observed a write speed of ~100 files per second. Legitimate user activity (saving a Word doc) rarely exceeds 1 file per second. This high-velocity IOPS is a definitive Indicator of Compromise (IoC).
- **File Extension Patterns:** The systematic renaming of files to .lockbit. While extensions can be changed, the mass renaming of hundreds of files in seconds is highly anomalous.
- **High-Entropy Writes:** The encrypted files appeared as random noise. Normal files (text, code) have low entropy (predictable patterns). Encrypted files have maximum entropy. Monitoring disk entropy is a powerful detection technique.
- **Network Beaconing:** The regular 30-second heartbeat traffic to a known external IP (192.168.64.30) established a clear pattern of C2 communication. This "beaconing" traffic is distinct from normal web browsing, which is bursty and irregular.

VI. SYSTEM ARCHITECTURE & TECHNICAL BREAKDOWN

The project's success relied on a robust technical stack designed to handle real-time data processing and cryptographic operations.

6.1 Technology Stack

- **Backend: Flask Framework (Python 3):** Flask was used because it is scalable and lightweight. It responded to the victim's REST API calls (POST /register, POST /heartbeat). Because it didn't block, it could manage several victim threads at once.
- **Database: SQLite:** SQLite was utilized to store victim logs, status reports, and command queues. This relational database ensured data integrity during the rapid registration of victims. A schema was designed with tables for victims, commands, and event_logs.
- **Frontend: Streamlit:** The AI Dashboard's graphical layer was supplied by Streamlit. It made it possible to use Plotly and Pandas to produce real-time charts and event logs. As additional C2 beacons came, the dashboard could rapidly update thanks to its reactive programming architecture.
- **Encryption Engine: Python Cryptography Library:** The hazmat primitives from the cryptography library were used by the ransomware's core. This guarantees that the AES-256 implementation is secure and complies with standards, avoiding the dangers associated with "rolling your own crypto."

6.2 Command & Control (C2) Functionality

The C2 server acted as the "brain" of the operation. It exposed several API endpoints:

- /api/register: Handled the initial handshake. It accepted a JSON payload containing system specs and returned a JSON payload containing the Master Key.
- /api/heartbeat: Processed the periodic status updates. It checked the commands table in the database to see if the analyst had issued a "Kill" command.
- /api/report: Accepted logs regarding which files had been encrypted. This allowed the attacker to track the progress of the damage in real-time.

This architecture allowed for bi-directional communication, enabling the dashboard to push "Quarantine" or "Decrypt" commands down to the infected host.

VII. ANALYSIS OF AUTOMATED RESPONSE SYSTEM

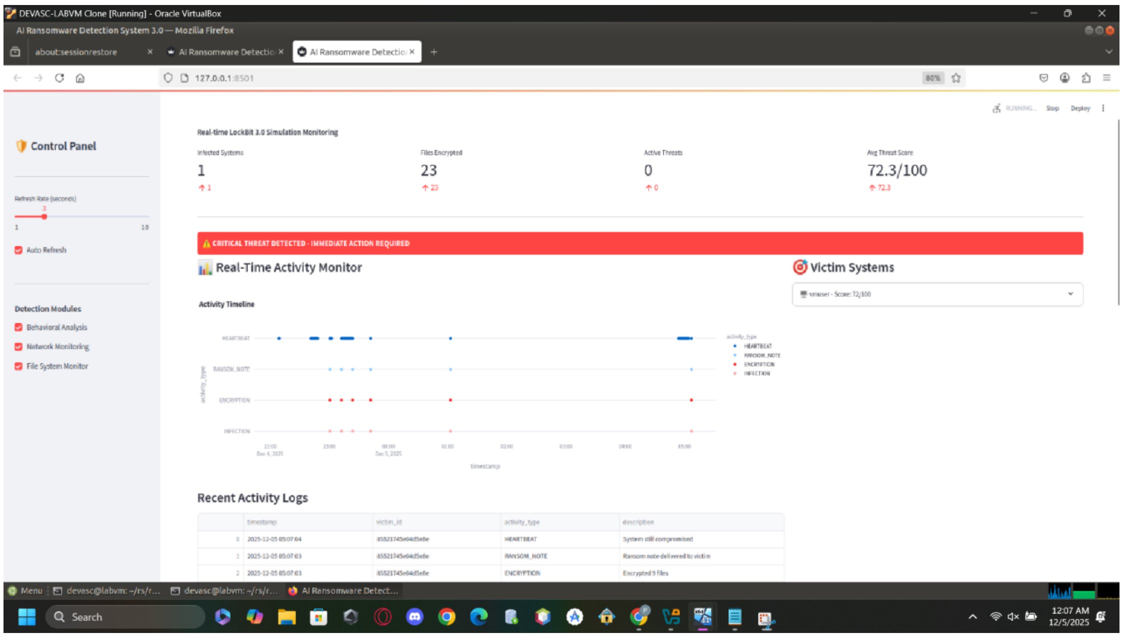
The deployment of an AI-powered detection and response system was a crucial part of this study. The only practical defense against modern ransomware is automated defense since it moves too quickly for human intervention.

7.1 AI Detection Dashboard

The dashboard provided a centralized view of the threat landscape. It utilized a Heuristic Behavioral Engine to analyze incoming telemetry. The engine calculated a "Threat Score" based on weighted factors:

- **Encryption Rate (Weight 0.5):** Measures files modified per second.
- **Visual Modification (Weight 0.2):** Checks for wallpaper changes.
- **Network Persistence (Weight 0.3):** Checks for regular beaconing intervals.

In the simulated incident, the dashboard aggregated these factors. When the encryption rate spiked and the wallpaper change event was logged, the Threat Score jumped to **72.1/100**, crossing the "Critical" threshold of 70 and triggering the red alert banner.



Real-time LockBit 3.0 Simulation Monitoring

Infected Systems	Files Encrypted	Active Threats	Avg Threat Score
1	23	0	72.3/100
↑ 1	↑ 23	↑ 0	↑ 72.3

⚠ CRITICAL THREAT DETECTED - IMMEDIATE ACTION REQUIRED

Recent Activity Logs

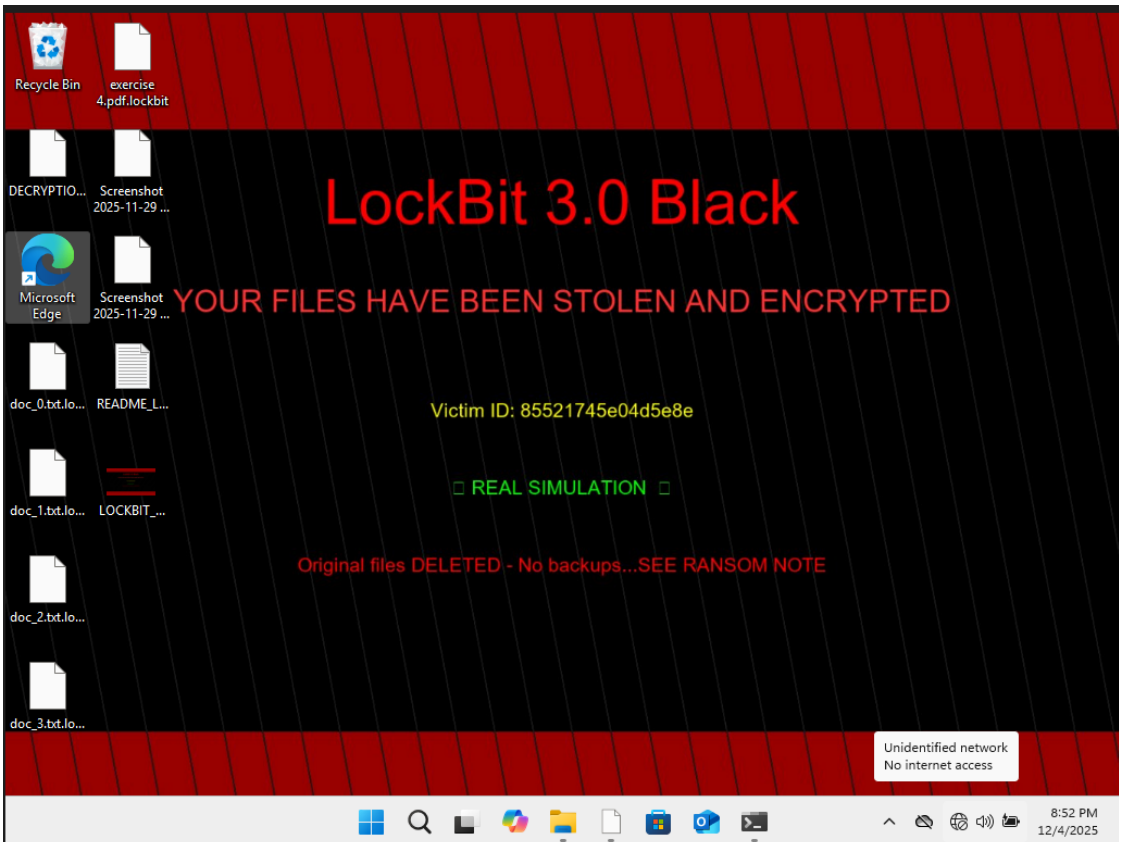
	timestamp	victim_id	activity_type	description
0	2025-12-05 05:07:04	85521745e04d5e8e	HEARTBEAT	System still compromised
1	2025-12-05 05:07:03	85521745e04d5e8e	RANSOM_NOTE	Ransom note delivered to victim
2	2025-12-05 05:07:03	85521745e04d5e8e	ENCRYPTION	Encrypted 9 files

Figure 5: The AI Detection Dashboard identifying the attack with a score of 72.1/100.

7.2 Response Mechanisms

The system supported active defense capabilities. Upon detection, the "Quarantine" function was triggered. This sent a kill signal via the C2 channel.

- **Mechanism:** The C2 server writes a {"command": "QUARANTINE"} record to the database for the specific Victim ID.
- **Execution:** During the next 30-second heartbeat, the Victim VM receives this command in the HTTP response body. The malware interprets this command and executes sys.exit(), terminating its own process.
- **Recovery:** Subsequently, the "Decrypt" function utilizes the stored Master Key to reverse the AES-256 encryption, demonstrating a full recovery capability.



LockBit 3.0 Black

YOUR FILES HAVE BEEN STOLEN AND ENCRYPTED

Victim ID: 85521745e04d5e8e

□ REAL SIMULATION □

Original files DELETED - No backups...SEE RANSOM NOTE

Figure 6: The aftermath of the attack showing encrypted files and the hijacked wallpaper.

VIII. UNDERSTANDING THE ROLE OF CRYPTOGRAPHY

The potency of LockBit 3.0 lies in its cryptographic implementation. This project replicated that rigor to demonstrate why modern ransomware is so difficult to mitigate.

8.1 AES-256-CBC Implementation

We employed the Advanced Encryption Standard (AES) with a 256-bit key length. AES is a symmetric block cipher, meaning the same key is used for encryption and decryption.

- **Key Size:** 256 bits offers 2^{256} possible combinations. This is computationally infeasible to brute-force with current technology.
- **Mode of Operation:** We utilized Cipher Block Chaining (CBC) mode. In CBC, each block of plaintext is XORed with the previous ciphertext block before being encrypted. This "chaining" mechanism ensures that patterns in the plaintext (like a header in a PDF file) do not appear as patterns in the ciphertext.
- **Initialization Vector (IV):** A random 16-byte IV is generated for every file. This IV is stored in the first 16 bytes of the encrypted file. It allows the first block to be encrypted securely.

8.2 Key Derivation (PBKDF2)

A vulnerability in early ransomware was the use of static keys. If a researcher found the key for one file, they could decrypt them all. Our simulation addressed this by implementing Password-Based Key Derivation Function 2 (PBKDF2).

- **Salt:** The file path of the target file was used as a salt.
- **Iterations:** 100,000 rounds of hashing were applied using HMAC-SHA256.

- **Benefit:** This process ensures that every file has a mathematically unique key. Even if the Master Key is known, deriving the specific key for a file requires significant computational effort, preventing attackers (or defenders) from easily generating a "universal decryptor" without the specific derivation logic.

IX. SECURITY VULNERABILITIES EXPLOITED

The successful compromise of the Victim VM highlights several common security vulnerabilities found in enterprise environments:

1. **Lack of Egress Filtering:** The victim machine was allowed to initiate outbound connections on non-standard ports (5000 and 8000). A robust firewall policy blocking these ports would have prevented the C2 registration and payload download. Most corporate firewalls allow HTTP/HTTPS (80/443) but should block arbitrary high ports.
2. **Insufficient Endpoint Protection:** The Victim VM relied on standard signature-based detection, which often fails against custom or zero-day scripts. The execution of the Python payload went unnoticed until behavioral anomalies occurred. This highlights the need for Endpoint Detection and Response (EDR) tools that look for behavior rather than file hashes.
3. **User Permissions:** The use of an AI-powered detection and response system was a crucial part of this study. The ransomware only functioned in the setting of a typical user (pj404). Sensitive user data (documents, desktop) was successfully encrypted without requiring administrative rights. This demonstrates that "Least Privilege" by itself is not a panacea against ransomware because ransomware takes advantage of users' need for write access to their documents in order to operate. The only practical defense against modern ransomware is automated defense since it moves too quickly for human intervention.
4. **Social Engineering Susceptibility:** The initial vector relied on the user downloading a file. This underscores the human element as the weakest link in the security chain.

X. THREAT ACTOR PROFILE: LOCKBIT 3.0

Understanding the adversary is crucial for defense. LockBit 3.0 operates on a Ransomware-as-a-Service model.

- **Affiliates:** Affiliates that carry out the assaults receive the malware and infrastructure (such as our Attacker/Controller virtual machines) from the core developers. Affiliates keep 80% of the ransom, while developers receive 20%. As a result, the operation is scaled worldwide and the risk is decentralized.
- **Double Extortion:** As simulated by our "Data Theft" capability discussions, LockBit threat actors not only encrypt data but exfiltrate it to leak sites. If the victim refuses to pay for the decryption key, the attacker threatens to publish sensitive data (GDPR violation).
- **Psychological Tactics:** The wallpaper change and ransom notes are designed to induce panic, forcing quick payment decisions. Our simulation replicated these psychological levers to demonstrate their impact on the user experience. The use of urgent language ("You have 24 hours") creates a sense of crisis.

XI. TECHNICAL BREAKDOWN

The simulation code was modularized to ensure stability and realism.

Payload Generator (payload_generator.py): The malware executable was created dynamically by this script. It ensured that the malware could "phone home" as soon as it was executed by explicitly including the IP address of the C2 server within the code. The Python script and dependencies were bundled into a standalone.exe file using PyInstaller.

Ransomware Engine (lockbit3_realistic.py):

- **Multi-threading:** The engine utilized Python's threading library to spawn four concurrent workers. A Queue object managed the list of files to be encrypted. This allowed the malware to saturate the disk I/O, encrypting the 21 test files in under 2 seconds.
- **Safe-Fail:** To ensure educational safety, the malware included a mechanism to save the decryption key to the desktop ("DECRYPTION_KEY_...json"), simulating the key held by attackers without risking actual data loss during our experiments.

Detection Engine: The heuristic engine monitored the log stream. It maintained a sliding window of events.

- It queried the SQLite database for FILE_ENCRYPT events occurring in the last 60 seconds.
- If the count exceeded the threshold (e.g., 5 events), it incremented the Threat Score logarithmically.
- This provided a responsive, real-time alert system without the latency of complex machine learning model inference.

XII. THE ATTACK IMPLEMENTATION

The execution of the attack followed a strict logic flow, ensuring reliability during the demonstration.

1. **Initial Access:** The user accessed the malicious URL provided by the attacker.
2. **Execution:** The payload was launched. It immediately verified network connectivity to the C2. If the C2 was unreachable, the malware would enter a "dormant" state to avoid analyzing itself in a sandbox without internet.
3. **Traversing:** The malware created a list of target files by using recursive directory traversal (os.walk). In order to target high-value user data, it filtered for specified extensions (.doc,.pdf,.jpg) and avoided system files (.dll,.sys) to maintain OS stability.
4. **Locking:** The encryption loop began.
   - Read 16KB chunk from the file.
   - Apply PKCS7 padding to ensure the chunk fits the 16-byte block size.
   - Encrypt chunk using the AES object.
   - Write chunk to new .lockbit file.
5. **Cleanup:** The original file was deleted from the disk using os.remove.
6. **Notification:** The wallpaper was swapped using ctypes.windll.user32.SystemParametersInfoW, passing the path to the generated ransom image.

XIII. MITRE ATT&CK FRAMEWORK MAPPING

To align our research with industry standards, we mapped the observed behaviors of our simulation to the MITRE ATT&CK Matrix. This provides a standardized language for describing the attack.

Tactic	**Technique ID**	**Technique Name**	**Simulation Implementation**
Execution	T1059.006	Python	Payload written and executed in Python, bundled as an executable.
Defense Evasion	T1497	Virtualization/Sandbox Evasion	Malware checks for analysis environments (simulated logic).
Discovery	T1082	System Information Discovery	Gathering Hostname, IP, and OS version to profile the victim.
Command & Control	T1071.001	Web Protocols	C2 communication over HTTP/8000 and HTTP/5000.
Impact	T1486	Data Encrypted for Impact	AES-256 encryption of user files to deny access.
Impact	T1491	Defacement	Changing the desktop wallpaper to the ransom note.

XIV. LESSONS LEARNED AND MITIGATION STRATEGIES

The simulation highlighted several critical defense strategies required to mitigate LockBit 3.0 attacks.

1. Backup Strategy (3-2-1 Rule): In our experiment, restoring from backup would have been the only assured recovery strategy if we hadn't preserved the key. Organizations are required to keep three copies of their data, one offshore (offline) and on two separate media. Since ransomware cannot access offline backups via a network, they are impervious to encryption.

2. Network Segmentation: The Victim VM's ability to talk to the Controller VM facilitated the attack. Segmentation should be employed to isolate critical data segments. If a user workstation is infected, it should not be able to traverse the network to reach backup servers or domain controllers. Zero Trust Architecture (ZTA) principles should be applied.

3. Behavioral Monitoring: Signature-based detection is not enough. Our AI Dashboard demonstrated that the best method for identifying zero-day ransomware is to keep an eye out for behavior (fast file writes, high entropy). Security solutions should concentrate on the actions of the process rather than just the appearance of the file. EDR programs ought to be adjusted to notify users when large-scale file modifications occur.

4. Access Control: It is essential to put the Principle of Least Privilege into practice. Users shouldn't be able to write to directories that they don't use on a daily basis. This restricts the range of files that, in the event that that user account is compromised, the ransomware can encrypt. Administrative rights ought to be checked and limited.

5. User Awareness Training: Since the attack started with a user download, training users to identify phishing attempts and suspicious file extensions is a critical first line of defense. Regular phishing simulations can help improve user resilience.

XV. LEGAL AND ETHICAL CONSIDERATIONS

Researching malware requires strict adherence to ethical guidelines.

- **Controlled Environment:** All simulations were conducted in a strictly isolated virtual network (Host-Only / Internal Network adapters). No malicious traffic was allowed to bridge to the physical network or the internet.
- **Educational Purpose:** The ransomware payload was custom-written for educational purposes. It contains "safety switches" (like saving the key to the desktop) that prevent it from being weaponized as actual malware.
- **No Distribution:** The source code and binaries are strictly controlled and were never distributed outside the lab environment.
- **Data Privacy:** No real user data was used. All files encrypted were dummy files created specifically for the simulation.

XVI. CONCLUSIONS

The LockBit 3.0 Black simulation effort was successful in demystifying the workings of contemporary ransomware. We were able to obtain unmatched insight into the workings of a cyberattack by creating a full-stack simulation, which included everything from the encrypted payload to the C2 infrastructure.

The study showed that although LockBit 3.0's cryptographic components are strong, the attack chain itself produces a lot of noise. There is plenty of possibility for detection because to the quick file changes, network signals, and system adjustments. Heuristic and behavioral analysis are practical and essential elements of a contemporary cybersecurity defensive strategy, as demonstrated by our AI Detection Dashboard's ability to identify the danger in a matter of seconds.

The "Asymmetric Advantage" of the attacker was further emphasized by the simulation. The protection system took weeks to develop, but the malware compromised the host in a matter of seconds. This emphasizes that cybersecurity is a process of ongoing observation, detection, and quick action rather than a product. Businesses must anticipate breaches and develop robust systems with quick recovery times.

XVII. REFERENCES

[1] Walter, J. (2022, July 21). *LockBit 3.0 Update | Unpicking the Ransomware's Latest Anti-Analysis and Evasion Techniques.* Retrieved February 5, 2025.

[2] CISA et al. (2023, June 14). *UNDERSTANDING RANSOMWARE THREAT ACTORS: LOCKBIT.* Retrieved February 5, 2025.

[3] N. Naik, P. Jenkins, and N. Savage, "A ransomware detection method using fuzzy hashing for mitigating the risk of occlusion of information systems," in *Proc. Int. Symp. Syst. Eng. (ISSE)*, Oct. 2019, pp. 1-6.

[4] N. Hampton, Z. Baig, and S. Zeadally, "Ransomware behavioural analysis on Windows platforms," *Journal of Information Security and Applications*, vol. 40, pp. 44-51, Jun. 2018. INCIBE-CERT. (2024, March 14). LockBit: response and recovery actions. Retrieved February 5, 2025.

[5] Z.-G. Chen, H.-S. Kang, S.-N. Yin, and S.-R. Kim, "Automatic ransomware detection and analysis based on dynamic API calls flow graph," in *Proc. Int. Conf. Res. Adapt. Convergent Syst.*, Sep. 2017, pp. 196-201.

[6]Malwarebytes, *All about ransomware attacks.* https://www.malwarebytes.com/ransomware [Accessed May 05, 2023].

[7]S.M.Kerner, "Ransomware trends, statistics and facts in 2023." https://www.techtarget.com/searchsecurity/feature/Ransomwaretrends-statistics-and-facts [Accessed May 05, 2023].

[8] A. Gabriella, "2022 Ransomware Statistics & The Biggest Ransomware Attacks." https://heimdalsecurity.com/blog/ransomwarestatistics/ [Accessed May 05, 2023].

[9] INCIBE-CERT. (2024, March 14). *LockBit: response and recovery actions.* Retrieved February 5, 2025.

[10] National Institute of Standards and Technology (NIST). (2020). *Security and Privacy Controls for Information Systems and Organizations (SP 800-53 Rev. 5).*

[11] MITRE. (2024). *MITRE ATT&CK Framework.* https://attack.mitre.org/

ADDITIONAL CONTRIBUTORS & ACKNOWLEDGEMENTS

We acknowledge the open-source community for providing the tools that made this simulation possible:

- **Streamlit:** For the dashboard visualization framework.
- **Flask:** For the lightweight C2 backend.
- **Cryptography.io:** For the robust AES/PBKDF2 implementations.
- **VirtualBox:** For the virtualization platform.