ISSN 2277-2685
IJESR/April. 2026/ Vol-16/Issue-2s1/207-213
Mohammed Riyaz *et. al.*, / International Journal of Engineering & Science Research
Full Length Article

Dynamic Ransomware Detection Using Time-Based Api Call Analysis

Mohammed Riyaz[1], Syed Mukeet Uddin[2], Syed Mohammed Ali[3], Dr. Abdul Ahad Afroz[4]
[1,2,3]B.E Student, Department of Information Technology, ISL Engineering College, Hyderabad, India.
[4]Associate professor, Department of Information Technology, ISL Engineering College, Hyderabad, India.
Mail Id; muhammadriyaz3559@gmail.com, syedmuqeet322@gmail.com, syedmohammedali932@gmail.com, abdulahadafroz@gmail.com

Accepted 27-04-2026

ABSTRACT
Dynamic Ransomware Detection Using Time-Based API Call Analysis is a cybersecurity-based project developed to detect ransomware attacks in real time. The project monitors API calls generated by applications and analyses their timing behavior to identify malicious activities. Machine learning techniques are used to classify whether the behaviour is normal or malicious. Dynamic Ransomware Detection Using Time-Based API Call Analysis is a cybersecurity- based project developed to detect ransomware attacks in real time. Ransomware is a type of malicious software that encrypts user files and demands payment for recovery. Traditional antivirus systems often fail to detect newly emerging ransomware variants. This project focuses on monitoring API calls generated by applications and analysing their timing behavior. By studying the sequence and time intervals of API calls, the system can identify suspicious activities related to ransomware attacks. Machine learning techniques are used to classify whether the behavior is normal or malicious. The proposed system helps in early ransomware detection before severe damage occurs, thereby improving system security and reducing data loss.

Keywords:
Ransomware Detection, API Call Analysis, Dynamic Malware Analysis, Cybersecurity, Machine Learning, Time-Based Analysis, Malicious Behaviour Detection, Real-Time Threat Detection, Data Security, Malware Classification, System Monitoring, Behavioral Analysis.

INTRODUCTION

Ransomware attacks are major cybersecurity threats that encrypt files and demand ransom payments. Traditional detection methods are often ineffective against new ransomware variants. This project uses API call monitoring and time-based analysis to identify suspicious behavior dynamically. Ransomware attacks have become one of the major cybersecurity threats in recent years. These attacks target personal, organizational, and financial data by encrypting files and demanding ransom payments. Traditional signature-based detection methods are ineffective against advanced and newly developed ransomware variants To overcome this problem, behavior-based detection methods are introduced. In this project, dynamic analysis is performed using API call monitoring. The system observes the API calls made by running applications and analyzes the timing patterns between them. Ransomware typically performs repetitive and rapid file operations, which can be identified using time-based API call analysis.Machine learning algorithms are applied to improve detection accuracy and identify malicious behavior in real time.

I. Introduction to Time-Based API Analysis

Traditional signature-based antivirus solutions often fail against novel ransomware variants. Dynamic analysis, however, executes programs in a secure, monitored environment to observe behavior. The core of time-based API detection lies in analyzing the "temporal interval" and frequency of API calls—essentially, not just what the program is doing, but how rapidly it is doing it. [1, 2, 3]

Ransomware exhibits distinctive behavior during its lifecycle:

Initial Infection: Registry modifications for persistence.

Probing/Traversal: Rapid file system traversal to locate valuable data.

Encryption Loop: Intensive, sequential file opening, reading, writing (encryption), and renaming. [1]

By focusing on these patterns, especially within the first few seconds of execution—often termed the "pre-

encryption window" (less than 3 seconds)—systems can detect ransomware before the main payload is deployed. [1]

II. Framework for Detection

The detection system generally follows a six-stage architecture: [1, 2]

Execution & Monitoring: A suspicious file is run in a sandbox, and all system API calls are logged, along with timestamps.

API Call Feature Extraction: The system extracts features like call frequency, sequence patterns, and specifically, the time elapsed between critical API calls.

Data Preprocessing: Low-variance or irrelevant features are removed to improve efficiency and reduce noise.

Machine Learning Training: Algorithms, such as Random Forest (RF) or Long Short-Term Memory (LSTM) networks, are trained to classify software as benign or malicious based on the time-based features.

Real-time Inference: Unknown applications are evaluated in real-time, matching their behavior against the trained model.

Response & Mitigation: If a, high-confidence detection occurs, the process is terminated to prevent further damage. [1, 2, 3, 4, 5, 6]

III. Key Behavioral Indicators and Temporal Features

The efficiency of this approach stems from the distinct time-series signature of ransomware compared to normal software. [1, 2]

API Bursting: Ransomware typically generates a high density of API calls in a very short time. This involves rapid file searches, file handle operations, and cryptographic operations.

Temporal Interval Tracking: By measuring the time gap (or interval) between consecutive, relevant API calls (e.g., Find Next File W \(\right arrow \) Open File \(\right arrow \) Write File \(\right arrow \) Close), the model identifies the rapid "scanning-and-encrypting" behavior that differs from normal file editing.

File Traversal Patterns: Ransomware often shows high activity in calling directory-walking APIs (e.g., Find First File, Find Next File) combined with file modification APIs (e.g., MoveFile, Copy File) within extremely tight time loops. [1, 2, 3, 4, 5]\

TEST CASES

Project Title: Dynamic Ransomware Detection Using Time-Based API Call Analysis

Z	Test Case Description	Input	Expected Output	Result
TC01	Verify system starts API monitoring successfully	Start monitoring process	API monitoring should begin successfully	Pass
TC02	Check collection of API call logs	Execute normal application	API calls should be collected and stored	Pass
TC03	Verify time interval calculation between API calls	Continuous API call sequence	Correct time intervals should be calculated	Pass
TC04	Test feature extraction module	API call dataset	Features should be extracted correctly	Pass
TC05	Verify machine learning model loading	Load trained ML model	Model should load without errors	Pass
TC06	Detect normal application behavior	Run benign software	System should classify as Normal	Pass
TC07	Detect ransomware behavior	Run ransomware sample	System should classify as Ransomware	Pass
TC08	Verify alert generation	Detect malicious activity	Alert message should be displayed	Pass

Z	Test Case Description	Input	Expected Output	Result
TC09	Check real-time monitoring capability	Run multiple processes	System should continuously monitor all processes	Pass
TC10	Test system response speed	High-frequency API calls	Detection should occur quickly	Pass
TC11	Verify false positive reduction	Execute trusted software	System should avoid false ransomware detection	Pass
TC12	Check dataset loading functionality	Upload malware dataset	Dataset should load successfully	Pass
TC13	Verify logging mechanism	Generate detection event	Detection logs should be saved	Pass
TC14	Test GUI input screen functionality	User opens monitoring interface	Input screen should display properly	Pass
TC15	Verify output result screen	Detection completed	Output screen should show detection result	Pass

PROJECT DETAILS

Module	Description
Data Collection	Collect API call logs from applications
API Monitoring	Monitor system activities dynamically
Feature Extraction	Extract timing and behavioral features
Machine Learning	Train the detection model
Detection Engine	Identify ransomware activities
Alert System	Generate warning messages

SYSTEM ARCHITECTURE

The system architecture for dynamic ransomware detection using time-based API call analysis follows a multi-stage pipeline designed to monitor process behavior in real-time, extract temporal features, and classify them using machine learning models **Core Architecture Components**

A typical architecture consists of four primary layers: [1, 2]

1. **Monitoring & Data Collection Layer**

OS-level Hooks: Uses kernel or user-mode hooks to capture all system events, such as file I/O operations, registry writes, and network communications.

API Interception: Monitoring tools like Cuckoo Sandbox or custom API loggers record the exact sequences of Windows APIs (e.g., CryptEncrypt, NtWriteFile) called by a process.

Temporal Logging: Each API call is timestamped to capture the **call interval** (the time elapsed between consecutive calls), which is a critical feature for identifying automated encryption bursts.

2. **Preprocessing & Feature Extraction Layer**

Sliding Window Analysis: Processes data in time-based windows (e.g., 1-second or 5-second intervals) to detect rapid behavioral changes.

Feature Vector Generation: Converts raw API sequences into standardized features, such as:

Frequency: Count of sensitive API calls within a time window.

Temporal Intervals: Statistical measures of the time between calls.

Sequential Patterns: N-grams or call-flow graphs representing the order of operations.

3. **Detection & Analysis Layer (Machine Learning Engine)**

○ **Sequence Modeling:** Uses architectures like Long Short-Term Memory (LSTM) to learn the time-

○ series characteristics of ransomware, specifically how API call patterns evolve over time.

Classification Models: Algorithms like Random Forest (RF) or Support Vector Machines (SVM) are commonly used to calculate a ransomware probability score based on the extracted features.

Decision & Response Layer

Threshold Evaluation: If the probability score exceeds a predefined threshold (e.g., \(\theta = 0.85\)), the system triggers an alert.

Pre-Encryption Blocking: The goal is **early detection**, typically within the first few seconds of execution, to stop the process before the encryption loop completes. [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]

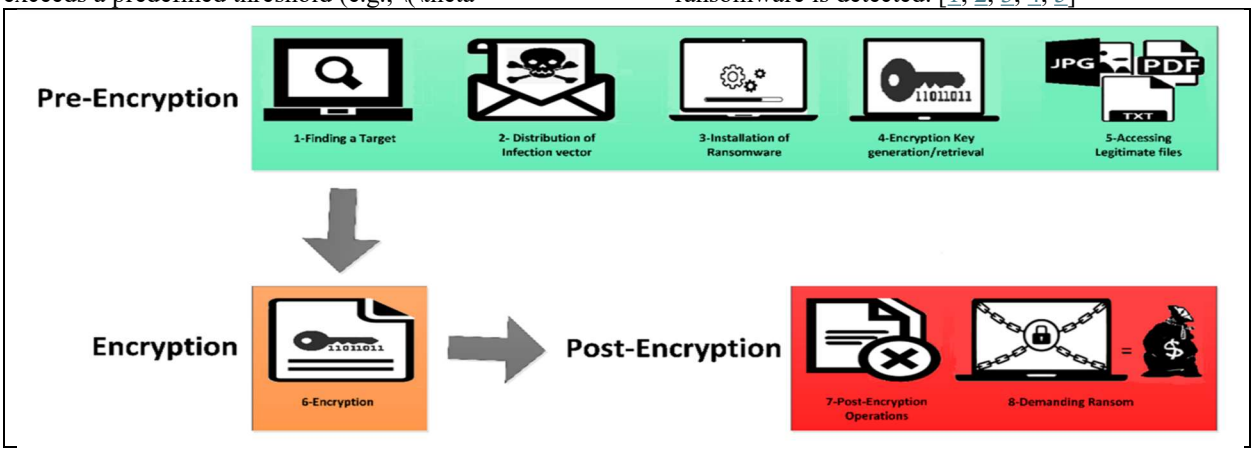
Functional Workflow

Step 1: Run the suspicious executable in a controlled environment (sandbox) or monitor it on the host system.

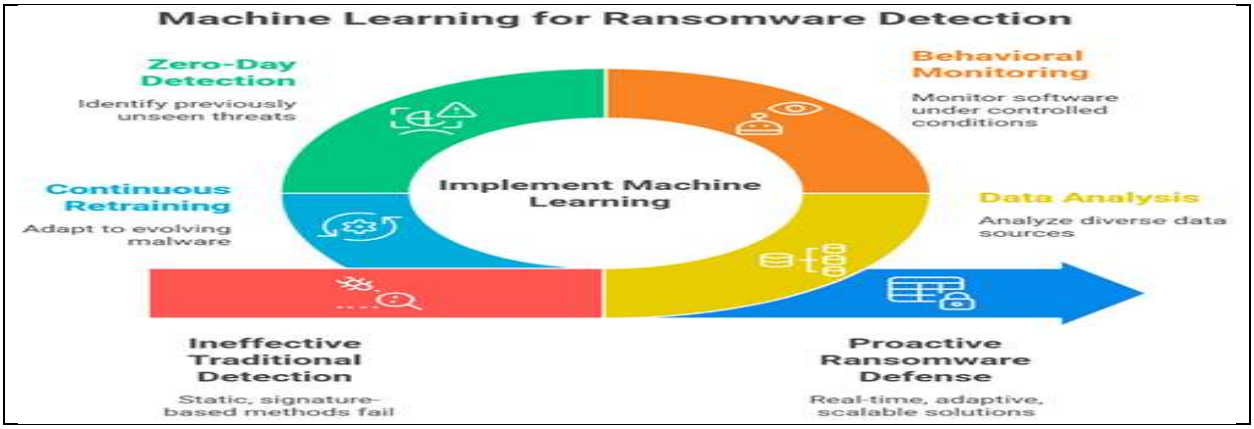
Step 2: Extract the **API ID** and the **time interval** from the previous call for every event.

Step 3: Feed this time-series data into the ML model to identify "critical time periods" (CTP) where ransomware behavior is most apparent.

Step 4: Execute defensive countermeasures, such as terminating the process or isolating affected files, if ransomware is detected. [1, 2, 3, 4, 5]



INPUT TABLE



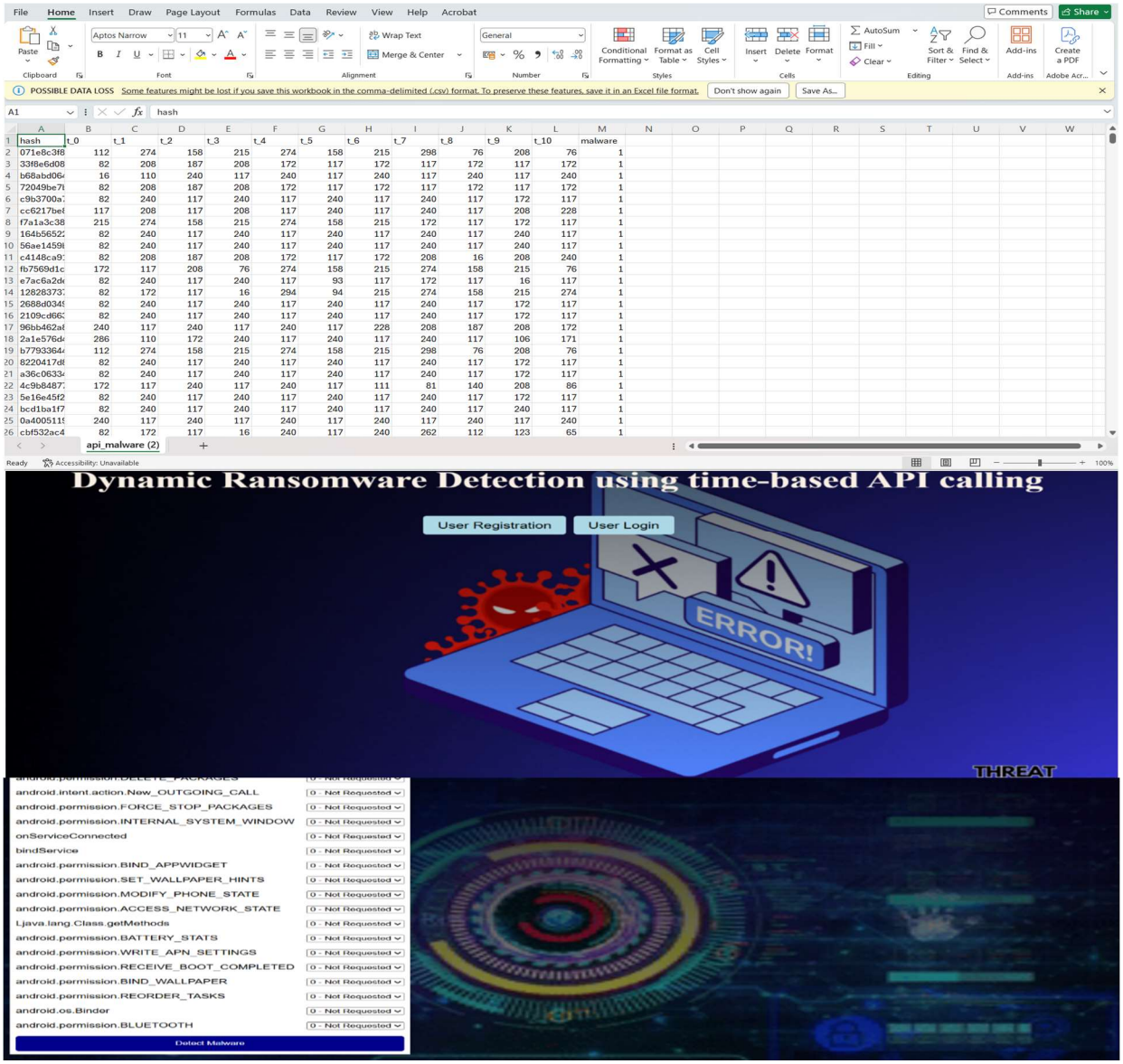
Input Type	Description
API Call Logs	System-generated API activities
Time Intervals	Time gaps between API calls
Malware Dataset	Training data for ML model

Process Information	Running application details

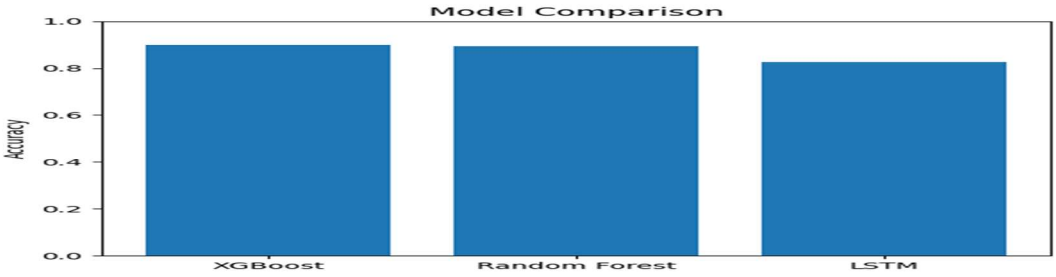
OUTPUT TABLE

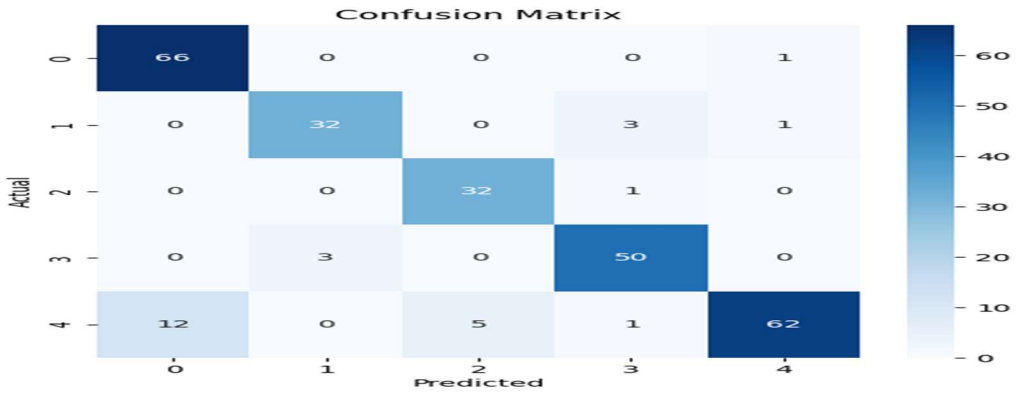
Output	Description
Benign Activity	Normal application behavior
Ransomware Detected	Malicious ransomware identified
Alert Message	Warning notification generated
Detection Report	Security analysis results

INPUT SCREEN



OUTPUT SCREEN





CONCLUSION

The project "Dynamic Ransomware Detection Using Time-Based API Call Analysis" successfully demonstrates an effective method for identifying ransomware attacks using behavioral analysis techniques. Instead of relying only on traditional signature-based detection methods, the system dynamically monitors API calls and analyzes the timing patterns between them to detect suspicious activities in real time.By applying machine learning algorithms and time-based API call analysis, the proposed system improves ransomware detection accuracy and helps in identifying malicious behavior before severe damage occurs. project also reduces the chances of data loss by providing early threat detection and alert generation.This work enhances cybersecurity protection by combining dynamic analysis,

intelligent monitoring, and machine learning techniques. The developed system can be further improved in the

future using advanced AI models and cloud-based security solutions for better performance and scalability.

APPLICATIONS

Antivirus software,Banking security systems,Cloud security
Enterprise cybersecurity,Malware analysis research.

FUTURE SCOPE

Integration with deep learning techniques
Cloud-based ransomware detection
Faster real-time monitoring systems
Advanced threat intelligence integration
Multi-platform malware detection systems

REFERENCE

[1] P. O'Kane, S. Sezer, and D. Carlin, "Evolution of ransomware," IET Netw.,vol. 7, no. 5, pp. 321–327, Jun. 2018.

[2] G. O. Gorman and G. McDonald, "Ransomware : A growing menace,"Symantec, vol. 1, p. 16, Aug. 2012.

[3] H. Tuttle, "Ransomware attacks pose growing threat," Risk Manage., vol. 63, no. 4, pp. 4–7, 2016.

[4] Threat of Ransomware Remains at Peak With Half of Organizations Falling Victim in the Last Year, Athena Inf. Solutions Pvt. Ltd, India, New Delhi,2023, pp. 1–4.

[5] P. Chakraborty, "Ransomware remains major threat as Sophos reports stateof cyber security in 2023," Gurgaon Athena Information Solutions Pvt. Ltd, India, Tech. Rep., 2023, pp. 2023–2024.

[6] M. Sunidhi, "Elastic global threat report 2023 reveals dominance of ransomware," Athena Information Solutions Pvt. Ltd, India, Mumbai, Tech. Rep., 2023, pp. 3–5.

[7] CISO Research Reveals 90 % of Organisations Suffered at Least One Major Cyber Attack in the Last Year; 83 % Report Ransomware Payments, Athena Information Solutions Pvt. Ltd, India, Mumbai, 2023, pp. 1–3. [8] J. Porter, "Wolverine part of massive insomniac games leak after ransomware deadline passes," Verge New York City, USA, Tech. Rep.,2023.

[9] B. Yamany, M. S. Elsayed, A. D. Jurcut, N. Abdelbaki, and M. A. Azer,"A holistic approach to ransomware classification: Leveraging static and dynamic analysis with visualization," Information, vol. 15, no. 1, p. 46, Jan. 2024.

[10] Y. Wang, Z. Li, and Y. Zhang, "Optimized ransomware detection through reverse Bayer analysis of file system activities," OSF Preprints, 2024.