PROJECT REPORT

**Title:  
Network Traffic Analysis using Wireshark and Zeek**



Submitted by:  
Atulendra Mishra  
B.Tech (Computer Science), 2nd Year  
United College of Engineering & Research, Prayagraj

# ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my guide, Mr. Ayush Kumar , for the valuable support, encouragement, and guidance provided throughout this project. Their experience, constructive criticism, and timely suggestions greatly helped in shaping the course of this project work.  
  
I am equally thankful to the Department of Computer Science & Engineering, United College of Engineering & Research, for providing the necessary facilities and environment for completing this project.  
  
I am grateful to my family and friends who were always a source of inspiration and encouragement, and without whose support this project would not have been successful.

# CERTIFICATE

This is to certify that the project entitled "Network Traffic Analysis using Wireshark and Zeek" is a bonafide record of the project work carried out by Atulendra Mishra, B.Tech (Computer Science), 2nd Year, in partial fulfillment for the award of the degree of Bachelor of Technology under my supervision and guidance.

# ABSTRACT

In today’s digital age, securing computer networks against unauthorized access, data breaches, and malicious activity has become increasingly critical. This project focuses on Network Traffic Analysis using two widely adopted tools: Wireshark and Zeek. The primary objective is to detect anomalies, suspicious communication patterns, and potential security threats by monitoring and analyzing network traffic in a controlled lab environment. The project begins by capturing live traffic data using Wireshark, a powerful packet analyzer, to inspect protocol-level communication. Concurrently, Zeek—a network security monitoring platform—is deployed to log network behavior and extract higher-level event information. Together, these tools allow for a comprehensive multi-layer analysis of the network.

We implemented several tests simulating real-world threats, such as port scanning, malformed packets, and brute-force login attempts. The analysis revealed patterns indicative of abnormal behavior, including irregular packet sizes, unexpected protocols, and unusual IP communication. Zeek’s scripting capabilities further enabled correlation of these eventsto identify actionable threats. Our findings highlight the effectiveness of combining Wireshark's deep packet inspection with Zeek’s behavioral analysis for proactive threat detection. The project underscores the importance of real-time traffic monitoring in maintaining cybersecurity and provides a blueprint for enhancing intrusion detection strategies in enterprise environments.

# INDEX

1. Introduction  
2. Objective  
3. Tools Used  
4. Methodology  
5. Implementation  
6. Observations  
7. Results  
8. Conclusion  
9. Future Scope  
10. References  
11. Appendix

# 1. INTRODUCTION

Network traffic analysis is vital in understanding how data flows through a network. It involves capturing and inspecting packets to uncover usage patterns, detect threats, and ensure optimal performance. This analysis is critical in areas such as cybersecurity, compliance, and IT troubleshooting.  
  
Wireshark and Zeek are two widely used tools in this domain. Wireshark is a GUI-based packet analyzer, while Zeek operates through scripting and logging, providing contextual, behavioral insights into network activity. Together, they offer a robust framework for comprehensive traffic monitoring.  
  
This report presents a detailed study on utilizing these tools for traffic analysis, including objectives, implementation, observations, and results.

# 2. OBJECTIVE

- Analyze live and recorded network traffic.  
- Understand differences in packet-level and log-based analysis.  
- Learn to detect common attacks such as port scanning and spoofing.  
- Explore the strengths of Wireshark in packet dissection.  
- Understand how Zeek can automate traffic inspection and alert generation.  
- Study the logs generated by Zeek and correlate them with Wireshark captures.

# 3. TOOLS USED

- \*\*Wireshark\*\*: A detailed packet-level inspection tool.  
- \*\*Zeek (formerly Bro)\*\*: A network analysis tool for high-level monitoring and alerting.  
- \*\*Ubuntu/Linux OS\*\*: Chosen for its compatibility with Zeek and command-line operations.  
- \*\*VirtualBox / VMware\*\*: For testing in isolated environments.  
- \*\*PCAP Files\*\*: Packet Capture files used for offline analysis.  
- \*\*Command-line Utilities\*\*: Such as tcpdump, ifconfig, and tshark.

# 4. METHODOLOGY

The methodology included the following steps:  
  
1. \*\*Setup\*\*: Installed Wireshark and Zeek on a Linux machine.  
2. \*\*Traffic Capture\*\*: Collected real-time network traffic using Wireshark and tcpdump.  
3. \*\*Zeek Analysis\*\*: Ran Zeek on the captured PCAP files to generate logs (conn.log, http.log, etc.).  
4. \*\*Interpretation\*\*: Compared the deep packet information from Wireshark with summarized insights from Zeek.  
5. \*\*Anomaly Detection\*\*: Identified scans and simulated attacks.  
6. \*\*Reporting\*\*: Prepared logs and screen captures for evidence-based reporting.

# 5. IMPLEMENTATION

The implementation process was carried out as follows:  
  
- Wireshark was installed and configured to capture on selected network interfaces.  
- Captures were saved in `.pcapng` format and opened for analysis.  
- Zeek was set up and verified with sample PCAPs for correct log generation.  
- Traffic was generated using browser activity and simple simulations (e.g., ping, HTTP requests).  
- Logs were parsed and checked for suspicious behavior such as SYN floods or malformed packets.  
- A comparison was made between Zeek's high-level insights and Wireshark's packet-level data.

# 6. OBSERVATIONS

Some key observations include:  
  
- Wireshark provided accurate and detailed layer-by-layer breakdown of each packet.  
- Zeek logs offered fast, filterable summaries ideal for scanning large traffic captures.  
- The conn.log revealed unexpected connection attempts and failed handshakes.  
- DNS log showed queries made by background processes.  
- Anomalies such as port scans were detected efficiently by Zeek.  
- Combining the tools improved validation and understanding of network behavior.

# 7. RESULTS

The following results were noted:  
  
- Detection of SYN scan and malformed packet injection from simulated attacks.  
- Successful parsing of DNS, HTTP, and SSL protocols by Zeek.  
- Wireshark was able to inspect payloads in detail.  
- Zeek provided fast, scriptable insights ideal for automation.  
- Logs generated (http.log, dns.log, conn.log) were consistent and useful for further ML-based inspection.

# 8. CONCLUSION

The project achieved its aim of exploring both low-level and high-level traffic analysis tools. Wireshark provided deep visibility, while Zeek added automation and intelligence to traffic review. The integration of both allows for comprehensive monitoring, faster detection, and rich insights required for modern cybersecurity challenges.  
  
These tools empower administrators to manage large-scale networks efficiently and respond proactively to emerging threats.

# 9. FUTURE SCOPE

- Integration with Elasticsearch and Kibana for dashboarding.  
- Use of Zeek logs in SIEM tools for enterprise security.  
- Incorporation of machine learning models to predict and classify threats.  
- Real-time alerting systems via Zeek scripting.  
- Encryption-level analysis using TLS fingerprinting.  
- Training modules based on live traffic for educational institutions.

# 10. REFERENCES

1. Wireshark Official Documentation - https://www.wireshark.org/docs/  
2. Zeek Network Security Monitor - https://zeek.org/  
3. Practical Packet Analysis - Chris Sanders  
4. Network Security Essentials - William Stallings  
5. Cybersecurity and Infrastructure Security Agency (CISA) guidelines  
6. GitHub Repositories: Wireshark samples, Zeek scripts

# 11. APPENDIX

Sample Zeek conn.log Output:  
----------------------------------  
ts uid id.orig\_h id.orig\_p id.resp\_h id.resp\_p proto service duration orig\_bytes resp\_bytes conn\_state local\_orig missed\_bytes history orig\_pkts orig\_ip\_bytes resp\_pkts resp\_ip\_bytes tunnel\_parents  
1629823937.12345 C8XY1q... 192.168.1.2 12345 93.184.216.34 80 tcp http 4.230012 350 2345 SF - 0 ShADadf 5 455 6 534 -  
  
Sample Wireshark Screenshot:  
[Insert Screenshot of Packet View with HTTP Headers]