International Journal of
Computational Learning & Intelligence
Vol. 02, Issue 04 (October) 2023

RESEARCH ARTICLE OPEN ACCESS

Network Troubleshooting Simulator Using Wireshark

Varsha . G R Nandita . Aishwarya . Usha Rani . Naveen Chandra Gowda

School of Computer Science and Engineering,
REVA University, Bengaluru, India

DOI: **10.5281/zenodo.10254248**
Received: 28 October 2023 / Revised: 05 November 2023 / Accepted: 02 December 2023
©Milestone Research Publications, Part of CLOCKSS archiving

Abstract – In the realm of information technology, network troubleshooting is a crucial ability that allows experts to locate and fix problems that might impair network functionality. An essential tool in this process is the popular network protocol analyzer Wireshark, which records and examines network traffic. An overview of a network troubleshooting simulator that uses Wireshark to teach and train IT professionals, students, and amateurs in identifying and fixing network issues. A potent teaching tool that provides a realistic and interactive environment for learning and perfecting network troubleshooting techniques is the Network Troubleshooting Simulator Using Wireshark.

Index Terms – Network Troubleshooting Simulator, Network Diagnostic Tool, Wireshark-Based Simulator, Network Protocol Analyzer, Simulation Tool for Network Troubleshooting

I. INTRODUCTION

Computer networks are the backbone of our linked world in the current digital landscape. For personal connectivity and business continuity, these networks must run well whether they are in a data centre, home, or workplace environment. But networks are not impervious to problems and disturbances that might obstruct communication, jeopardize security, and hinder efficiency. The process of locating and fixing these problems is known as network troubleshooting, and it calls for a mix of expertise, equipment, and knowledge. One such essential tool is the potent, free, and open-source. Packet analysis programme Wireshark. with the help of Wireshark, network administrator, IT specialities, and Security specialists may record, examine and debug network traffic providing detailed information about troubleshooting simulator is an excellent tool for anyone who what to practice troubleshooting network with the Wireshark in a real-world setting. The purpose of the simulator is to offer a secure and regulated setting for acquiring and referring the ability required to properly handle network simulators, it provides users with the knowledge and expertise necessary to confidently address network issues in the real world. Key feature of the Network Troubleshooting Simulator using.

Wireshark include

- Hands-On Experience: With Wireshark, users may actively interact with the simulator to record, examine, and analyze network data. This hands-on learning opportunity facilitates skill development in a regulated setting.
- Scalability Both novices and experts may have the simulator tailored to their skill levels. It is appropriate for anybody interested in network troubleshooting because it supports a variety of network complexity levels.
- Risk-Free Learning: Users may safely experiment in the Network Troubleshooting Simulator without worrying about upsetting or harming actual network setups. It promotes learning and experimentation without worrying about unforeseen repercussions.
- Performance Assessment: Users may get performance data and assess their network troubleshooting abilities. This data is useful for monitoring development and enhancing abilities.

Perfecting network troubleshooting techniques is the Network Troubleshooting Simulator Using Wireshark. It includes the following essential elements:

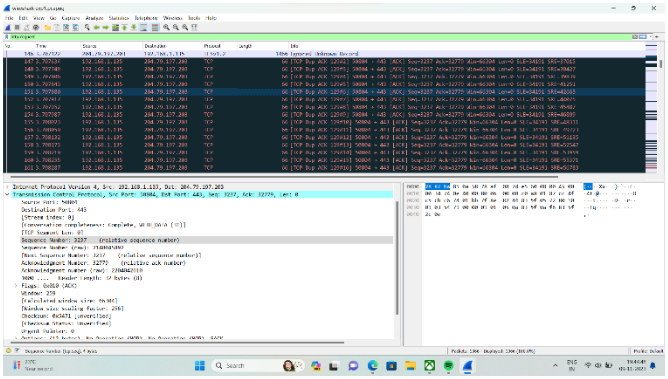
- Realistic Network Scenarios: A wide range of pre-configured network scenarios are available in the simulator; these scenarios depict various network difficulties, including poor performance, connection issues, security breaches, and more. By simulating actual network issues, these scenarios allow users to hone their problem-solving abilities in a safe setting
- Integrated Wireshark Capture: Users may collect, inspect, and analyses network packets in real time using the fully integrated version of Wireshark available to them. To identify the underlying source of problems whether they are due to misconfigurations, security risks, or performance bottlenecks, they can probe deeply into the network traffic.
- Step-by-Step Guidance. The simulator provides users with step-by-step instructions and tips to assist them in traversing the troubleshooting procedure It helps with the formulation of hypotheses, the use of troubleshooting techniques, and the comprehension of the implications of possible solutions.
- Assessment and Feedback: Through the use of the scenarios, users mayevaluate their troubleshooting abilities and receive instant performance feedback. The simulator encourages an iterative learning process by offeringscores, in-depth analysis, and suggestions for development.
- Customization and Real-time Feedback Network configurations may be changed by users, who can then evaluate the results instantly. Additionally, the simulator offers feedback on the effects of these modifications, assisting students in experimenting with various troubleshooting techniques.

II. PROPOSED SOLUTIONS

Proposing a work on a network troubleshooting simulator using Wireshark involves outlining the scope, objectives, and methodology of your project. Below is an example of a proposed work for a network troubleshooting simulator using Wireshark. An essential component of managing and keeping up with contemporary computer networks is network troubleshooting. Network engineers and administrators frequently deal with a variety of problems that need for prompt and efficient fixes, such as security breaches and performance concerns. Although it might be difficult for novices, the widely used network protocol analyser Wireshark offers deep insights into network traffic. The goal of this project is to create a network troubleshooting simulator that incorporates Wireshark's features to assist novices and experts in the field of network troubleshooting in gaining hands-on experience in fixing network problems.

- **Packet Losses**

One of the most prevalent issues I find on networks is packet loss. Packet loss occurs when data packets are not correctly delivered from sender to recipient over the internet. When a user visits a website and begins downloading the site’s elements, missed packets cause re-transmissions, increasing the ascertain to download the web files and slowing the overall downloading process.


wireshark exp1.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http.request

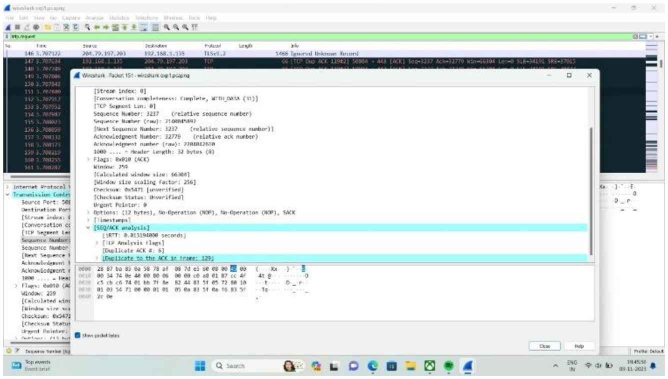
```
No.  Time       Source          Destination     Protocol Length Info
146 3.707122    204.79.197.203  192.168.1.135   TLSv1.2  1466 Ignored Unknown Record
147 3.707634    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#2] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=37015
148 3.707749    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#3] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=38427
149 3.707806    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#4] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=39839
150 3.707843    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#5] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=41251
151 3.707880    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#6] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=42663
152 3.707917    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#7] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=44075
153 3.707952    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#8] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=45487
154 3.707987    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#9] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=46899
155 3.708023    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#10] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=48311
156 3.708059    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#11] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=49723
157 3.708132    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#12] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=51135
158 3.708173    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#13] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=52547
159 3.708219    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#14] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=53959
160 3.708255    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#15] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=55371
161 3.708287    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#16] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=56783
```

```
> Internet Protocol Version 4, Src: 192.168.1.135, Dst: 204.79.197.203
v Transmission Control Protocol, Src Port: 50804, Dst Port: 443, Seq: 3237, Ack: 32779, Len: 0
    Source Port: 50804
    Destination Port: 443
    [Stream index: 0]
    [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 0]
    Sequence Number: 3237    (relative sequence number)
    Sequence Number (raw): 2140045092
    [Next Sequence Number: 3237    (relative sequence number)]
    Acknowledgment Number: 32779    (relative ack number)
    Acknowledgment number (raw): 2204842610
    1000 .... = Header Length: 32 bytes (8)
  > Flags: 0x010 (ACK)
    Window: 259
    [Calculated window size: 66304]
    [Window size scaling factor: 256]
    Checksum: 0x5471 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), SACK
```

```
0000  28 87 ba 85 0a 58 78 af  08 7d e5 60 08 00 45 00
0010  00 34 74 0e 40 00 80 06  00 00 c0 a8 01 87 cc 4f
0020  c5 cb c6 74 01 bb 7f 8e  82 44 83 5f 05 72 80 10
0030  01 03 54 71 00 00 01 01  05 0a 83 5f 0a f6 83 5f
0040  2c 0e
```

Sequence Number (tcp.seq), 4 bytes
Packets: 1366 · Displayed: 1366 (100.0%)
Profile: Default
23°C
Near record
Search
ENG
IN
19:44:48
03-11-2023

Fig. 1: Packet Loss Screenshot - 1


wireshark exp1.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http.request
Wireshark · Packet 151 · wireshark exp1.pcapng

```
No.  Time       Source          Destination     Protocol Length Info
146 3.707122    204.79.197.203  192.168.1.135   TLSv1.2  1466 Ignored Unknown Record
147 3.707634    192.168.1.135   204.79.197.203  TCP      66 [TCP Dup ACK 129#2] 50804 → 443 [ACK] Seq=3237 Ack=32779 Win=66304 Len=0 SLE=34191 SRE=37015
```

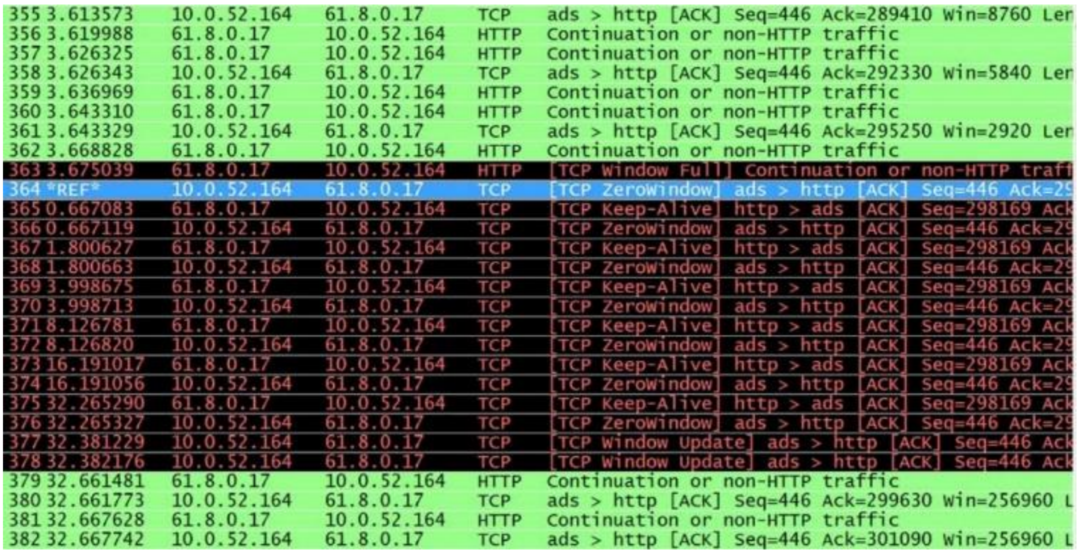
```
    [Stream index: 0]
    [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 0]
    Sequence Number: 3237    (relative sequence number)
    Sequence Number (raw): 2140045092
    [Next Sequence Number: 3237    (relative sequence number)]
    Acknowledgment Number: 32779    (relative ack number)
    Acknowledgment number (raw): 2204842610
    1000 .... = Header Length: 32 bytes (8)
  > Flags: 0x010 (ACK)
    Window: 259
    [Calculated window size: 66304]
    [Window size scaling factor: 256]
    Checksum: 0x5471 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), SACK
  > [Timestamps]
  v [SEQ/ACK analysis]
      [iRTT: 0.013194000 seconds]
    > [TCP Analysis Flags]
      [Duplicate ACK #: 6]
    > [Duplicate to the ACK in frame: 129]
```

```
0000  28 87 ba 85 0a 58 78 af  08 7d e5 60 08 00 45 00   (····Xx· ·}·`··E·
0010  00 34 74 0e 40 00 80 06  00 00 c0 a8 01 87 cc 4f   ·4t·@··· ·······O
0020  c5 cb c6 74 01 bb 7f 8e  82 44 83 5f 05 72 80 10   ···t···· ·D·_·r··
0030  01 03 54 71 00 00 01 01  05 0a 83 5f 0a f6 83 5f   ··Tq···· ···_···_
0040  2c 0e                                              ,·
```

Fig. 1: Packet Loss Screenshot - 2

Furthermore, when an application uses TCP, missing packets have a particularly negative impact. When a TCP connection detects a dropped packet, the throughput rate automatically slows down to compensate for network issues. It gradually improves to a more acceptable pace until the next packet is dropped, resulting in a significant reduction in data throughput. Large file downloads, which should otherwise flow easily across a network, suffer significantly from packet loss. What does it look like when a packet is lost? It is debatable. Packet loss can take two forms if the program is operating via TCP. In one example, the receiver monitors packets based on their sequence numbers and detects a missing packet. The client makes three requests for the missing packet (double acknowledgments), resulting in a resend. When a sender observes that a receiver has not confirmed receipt of a data packet, the sender times out and retransmits the data packet.

- **Problem Delay In Communication**



```
355 3.613573   10.0.52.164  61.8.0.17    TCP   ads > http [ACK] Seq=446 Ack=289410 Win=8760 Len
356 3.619988   61.8.0.17    10.0.52.164  HTTP  Continuation or non-HTTP traffic
357 3.626325   61.8.0.17    10.0.52.164  HTTP  Continuation or non-HTTP traffic
358 3.626343   10.0.52.164  61.8.0.17    TCP   ads > http [ACK] Seq=446 Ack=292330 Win=5840 Len
359 3.636969   61.8.0.17    10.0.52.164  HTTP  Continuation or non-HTTP traffic
360 3.643310   61.8.0.17    10.0.52.164  HTTP  Continuation or non-HTTP traffic
361 3.643329   10.0.52.164  61.8.0.17    TCP   ads > http [ACK] Seq=446 Ack=295250 Win=2920 Len
362 3.668828   61.8.0.17    10.0.52.164  HTTP  Continuation or non-HTTP traffic
363 3.675039   61.8.0.17    10.0.52.164  HTTP  [TCP Window Full] Continuation or non-HTTP traff
364 *REF*      10.0.52.164  61.8.0.17    TCP   [TCP ZeroWindow] ads > http [ACK] Seq=446 Ack=29
365 0.667083   61.8.0.17    10.0.52.164  TCP   [TCP Keep-Alive] http > ads [ACK] Seq=298169 Ack
366 0.667119   10.0.52.164  61.8.0.17    TCP   [TCP ZeroWindow] ads > http [ACK] Seq=446 Ack=29
367 1.800627   61.8.0.17    10.0.52.164  TCP   [TCP Keep-Alive] http > ads [ACK] Seq=298169 Ack
368 1.800663   10.0.52.164  61.8.0.17    TCP   [TCP ZeroWindow] ads > http [ACK] Seq=446 Ack=29
369 3.998675   61.8.0.17    10.0.52.164  TCP   [TCP Keep-Alive] http > ads [ACK] Seq=298169 Ack
370 3.998713   10.0.52.164  61.8.0.17    TCP   [TCP ZeroWindow] ads > http [ACK] Seq=446 Ack=29
371 8.126781   61.8.0.17    10.0.52.164  TCP   [TCP Keep-Alive] http > ads [ACK] Seq=298169 Ack
372 8.126820   10.0.52.164  61.8.0.17    TCP   [TCP ZeroWindow] ads > http [ACK] Seq=446 Ack=29
373 16.191017  61.8.0.17    10.0.52.164  TCP   [TCP Keep-Alive] http > ads [ACK] Seq=298169 Ack
374 16.191056  10.0.52.164  61.8.0.17    TCP   [TCP ZeroWindow] ads > http [ACK] Seq=446 Ack=29
375 32.265290  61.8.0.17    10.0.52.164  TCP   [TCP Keep-Alive] http > ads [ACK] Seq=298169 Ack
376 32.265327  10.0.52.164  61.8.0.17    TCP   [TCP ZeroWindow] ads > http [ACK] Seq=446 Ack=29
377 32.381229  10.0.52.164  61.8.0.17    TCP   [TCP Window Update] ads > http [ACK] Seq=446 Ack
378 32.382176  10.0.52.164  61.8.0.17    TCP   [TCP Window Update] ads > http [ACK] Seq=446 Ack
379 32.661481  61.8.0.17    10.0.52.164  HTTP  Continuation or non-HTTP traffic
380 32.661773  10.0.52.164  61.8.0.17    TCP   ads > http [ACK] Seq=446 Ack=299630 Win=256960 L
381 32.667628  61.8.0.17    10.0.52.164  HTTP  Continuation or non-HTTP traffic
382 32.667742  10.0.52.164  61.8.0.17    TCP   ads > http [ACK] Seq=446 Ack=301090 Win=256960 L
```

Fig. 3: Picture Displays A 32-Second Delay in Network Communications Due To A Zero-Window Scenario

The TCP stack's receiver window is a buffer space. When data is received, it is stored in this buffer space until an application picks it up. The receiver window fills up when an application does not keep up with the receive rate, eventually leading to a "zero window" scenario. All data transmission to the host must come to a halt when a receiver announces a zero-window condition. The rate of throughput falls to zero. A method known as Window Scaling (RFC 1323) allows a host to increase the receiver window size and lower the likelihood of a zero-window scenario.

III. IMPLEMENTATIONS

Using Wireshark to analyze network traffic and simulate network problems is the process of building a network troubleshooting simulator. Here is a broad overview of how to put such a simulator into practice:

- **Requirements**

  - Wireshark: Ensure you have Wireshark installed on your system. You can download it from the official website (https://www.wireshark.org/download.html).
  - Virtual Machines or Network Simulation Tools: You may need a network of virtual machines or use network simulation tools like GNS3 or Packet Tracer to simulate various network configurations and issues.
  - Scenario Design: Decide on the network scenarios and issues you want to simulate. Common network issues include:
    - Packet Loss: Simulate packet loss due to network congestion or misconfigured routers.
    - Latency: Introduce latency to mimic a slow network.
    - DNS Issues: Simulate DNS resolution problems.
    - Firewall Rules: Create scenarios to test firewall rules and traffic filtering.
    - IP Conflicts: Introduce IP address conflicts.
    - Routing Problems: Simulate routing issues.
    - Broadcast Storms: Generate excessive broadcast traffic.

- **Implementation Steps**

  - Set Up the Network: Use tools for network simulation or virtual machines to create your own network environment. Plan your network based on the circumstances you wish to replicate.
  - Start Wireshark: On the machine where Wireshark is installed, launch the application.
  - Select Capture Interface: Select the network interface for traffic collection. The simulated network ought to be linked to this interface.
  - Apply Filters: In Wireshark, use capture filters to focus on the traffic you wish to examine. You may filter, for instance, based on protocol, port, or source/destination IP.
  - Simulate Network Issues: Describe the planned network difficulties. For instance, you may use a programme to create jitter or limit the bandwidth on a particular link in order to mimic packet loss.
  - Capture Traffic: In Wireshark, click the "Start" button to start recording network traffic. Give it some time to gather sufficient data.
  - Analyze Packets: Once the network data has been recorded, packet analysis may be used to determine the problems. Examine the recorded data for any irregularities, mistakes, or trends.
  - Interpret Results: To understand the findings and determine the underlying cause of the simulated network problems, apply your Wireshark skills.
  - Document Findings: Keep a journal of your findings, mentioning the problem, any probable causes, and any prospective fixes. You can use this material for troubleshooting or for training.
  - Repeat: For further network situations and problems

- Provide Training: In the event that the simulator is intended for instructional use, network administrators or students can receive training from you using the data that was collected and your conclusions.

IV. RESULTS

Utilising resources like Wireshark in conjunction with network troubleshooting simulators is crucial for developing practical expertise in identifying and fixing network problems. A useful tool for diagnosing network issues, Wireshark is a powerful network protocol analyzer that records and examines network data. When utilising Wireshark in a network troubleshooting simulator, the following outcomes and advantages are anticipated:

- Real-world experience: Simulating different network scenarios in a controlled environment is made possible by network troubleshooting simulators. These simulations, which mimic actual network issues, provide you the opportunity to practise troubleshooting in a secure and regulated setting using Wireshark.
- Identifying network anomalies: By capturing network packets, Wireshark enables you to analyse network traffic. You can find abnormalities like excessive latency, packet loss, misconfigurations, or security risks by examining the recorded data.
- Isolating issues: With the aid of Wireshark, you may identify the precise cause of a network issue. Tracing network traffic can help you determine whether a problem or bottleneck is at the application, transport, or network layer.
- Verifying configurations: Verifying the settings of the router, switch, and firewall is a common step in network troubleshooting. You may verify whether the network devices are setup correctly and that the packets are acting as expected with the aid of Wireshark.
- Protocol analysis: For many different types of network protocols, Wireshark offers protocol decodes. It may be used to examine problems that are unique to a given protocol, including DNS resolution difficulties, DHCP faults, or HTTP errors.
- Security analysis: By examining network traffic, you may also practise spotting security problems in a network troubleshooting simulator. You can use Wireshark to identify unauthorised activity, malware communication, and intrusion attempts.
- Learning opportunities: When paired with Wireshark, network troubleshooting simulators provide a wealth of educational resources for network managers, security experts, and students. You can acquire useful abilities that you can use in actual network contexts.
- Documentation: You may record network traffic with Wireshark and store it for a later study. Sharing information with coworkers, working with other teams, and keeping track of changes over time all depend on this documentation.
- Improved problem resolution: Through the use of Wireshark in network troubleshooting simulators, you may improve your ability to swiftly and efficiently resolve network problems. This enhances overall network performance and lowers downtime.

V. CONCLUSIONS

When used with programmers like Wireshark, network troubleshooting simulators provide a strong and efficient platform for refining troubleshooting abilities and resolving a variety of network-related problems. To sum up, here are some important lessons learned:

- Realistic Practice: Simulating network issues in a controlled and realistic setting is made possible by network troubleshooting simulators. Network administrators, IT specialists, and students may all benefit greatly from this practical experience in order to develop their practical knowledge.
- Effective Learning: With Wireshark, users may record and examine network data as a network protocol analyzer. This study improves users' troubleshooting skills by assisting in the identification and understanding of network abnormalities, misconfigurations, and security concerns.
- Problem Isolation: With Wireshark, you can precisely isolate problems by tracking network traffic and locating the source of problems. This helps identify and fix issues at different network tiers, ranging from network devices to applications.
- Skill Development: These simulators provide chances for training and skill development that are transferable to actual network settings. Users may learn more effective techniques for fixing network issues, enhancing network functionality, and reducing downtime.

All things considered, network troubleshooting simulators, when used in conjunction with Wireshark, provide an indispensable toolkit for improving troubleshooting abilities, which in turn results in more effective problem solving and superior network administration. Gaining hands-on experience with these tools enhances one's comprehension of network behavior and strengthens one's capacity to preserve network security and integrity. In conclusion, employing Wireshark in network troubleshooting simulators may improve your troubleshooting abilities, make it easier for you to identify and fix problems with networks, and give you a secure setting in which to practice.

REFERENCES

1. "Wireshark Official Website": Wireshark's official website offers documentation, user guides, and tutorials on how to use the tool effectively. (https://www.wireshark.org/)
2. "Wireshark User's Guide": Wireshark provides an extensive user guide that covers various aspects of using the software for network analysis and troubleshooting. (https://www.wireshark.org/docs/wsug_html_chunked/)
3. "Network Troubleshooting and Maintenance Guide" by Neal Allen and Darrel Raynor: This book offers practical guidance on network troubleshooting and includes information on using Wireshark. It's a valuable resource for IT professionals. Book Link
4. "Practical Packet Analysis: Using Wireshark to Solve Real-World Network Problems" by Chris Sanders: This book provides practical examples and case studies on using Wireshark for network troubleshooting. It's a great resource for learning how to use Wireshark effectively. Book Link
5. "Wireshark Network Analysis: The Official Wireshark Certified Network Analyst Study Guide" by Laura Chappell: This book focuses on preparing for the Wireshark Certified Network Analyst (WCNA) certification and includes in-depth information on using Wireshark for network analysis. Book Link
6. "NetSimK - Network Simulation Tool": NetSimK is a network simulation tool that allows you to simulate complex network scenarios for training and troubleshooting. It can be used in conjunction with Wireshark for practical experience. (https://www.netsimk.com/)

7. "Packet Tracer" by Cisco: Packet Tracer is a network simulation tool offered by Cisco that is designed for learning and practicing networking concepts. It can be a valuable tool for network troubleshooting practice, alongside Wireshark. (https://www.netacad.com/courses/packet-tracer)
8. "GNS3 - Graphical Network Simulator": GNS3 is an open-source graphical network simulator that allows you to create complex network topologies for learning and troubleshooting. You can integrate Wireshark into GNS3 for packet analysis. (https://www.gns3.com/)
9. Online Forums and Communities: Online platforms like Stack Overflow, Reddit (r/Wireshark), and Wireshark's own community forums are great places to ask questions, seek help, and share knowledge about using Wireshark and network troubleshooting simulators.