REVOLUTIONIZING NETWORK PERFORMANCE TESTING WITH A NEXT-GEN TOOL

Poornika Sri G [1], Swetha S [2], Harini P [3], Nithya R [4]

[1] *Bachelor of Engineering, Computer Science And Engineering, Bannari Amman Institute of Technology, Erode, India*

[2] *Bachelor of Technology, Information Technology, Bannari Amman Institute of Technology, Erode, India*

[3] *Bachelor of Technology, Computer Technology, Bannari Amman Institute of Technology, Erode, India*

[4] *Assistant Professor, Computer Science And Engineering, Bannari Amman Institute ofTechnology, Erode, India*

ABSTRACT

This project explores in developing a network performance measurement tool that supports multiple clients concurrently. Computer networks have influenced the software industry by providing enormous resources distributed around the globe and interactions among people working anywhere in the world that the world today seems too small. Networks themselves have undergone a radical change in the last few decades starting from ARPANET to the Inter-Continental data cables that we see today. The amount of data that is carried on the Information Super Highway has been increasing everyday prompting for efficient management of the Trans-Continental Super Highway of data. The growing dependence on networks for everyday tasks has created the demand for high performance; reliable networks thereby making companies invest a lot on research on improving the networks and new designs. Part of achieving the goal of high performance is active monitoring of networks to help in the identification and prevention of network errors. Many tools have emerged to aid in performance monitoring of networks. The most common class of tools is based on the Simple Network Management Protocol (SNMP), a protocol for sending and transmitting network performance information on IP networks. Other types of network performance monitoring tools include packet sniffers, flow monitors and application monitors. Examples of the various monitoring tools are SolarWind's Orion SNMP monitoring platform, WireShark packet capture tool, Webmetrics' GlobalWatch and Cisco's NetFlow flow monitoring tools.Finally , in a laboratory environment, the performance of four network traffic generators are compared. Two computers with Windows operating systems were connected via a 100 Mbps link and for various payload sizes, ranging from 128 Bytes to 1408 Bytes, the TCP traffic on the link was measured using the various monitoring tools mentioned above. The results indicate that these tools can produce significantly different results. In the Windows environment, the bandwidth that the tools measure can vary as much as 16.5 Mbps for a TCP connection over a 100 Mbps link.

Keywords*: Network management protocol, internet protocol, monitoring tools, performance, tool, multiclient, traffic generator, performance tool, performance analysis.*

1. INTRODUCTION

Network performance testing plays an important role in ensuring the efficiency and reliability of data transfer in a computer network. It involves monitoring various parameters such as bandwidth, latency, and all devices to ensure the best possible connection. An important part of this testing is measuring the network's ability to handle data transfer between devices. A common approach to such measurements involves the use of client-server architecture, where the client initiates the transfer of data to the server.

1.1 Background of the project:

During this test, data is sent to the network so that its capacity and performance can be evaluated. This process helps identify potential vulnerabilities and ensure the network can maintain performance levels. This test is especially important in a network environment where factors such as cloud services, virtualization, connection of different devices can completely disrupt all network connections. Finally, network performance testing helps identify problems before they occur, improve configuration, and improve overall data transfer within the network infrastructure. The revolution in network performance measurement includes a shift to more flexible and intelligent tools that can meet the challenges of modern collaboration. These next-generation tools are essential for organizations that want to improve network performance, increase security, and provide a great end-user experience in the business environment.

1.2 Motivation (Scope of the Proposed Initiative)

The plan to revolutionize network performance measurement represents a forward-looking approach designed to address evolving challenges in today's IT environment. This approach is based on innovation and flexibility and aims to improve the accuracy, efficiency and understanding of network performance measurement. At the core of this approach is the integration of technology and strategic approaches to revise the way organizations evaluate, optimize and manage cyberinfrastructure.

Key elements of the plan include technology, machine learning and artificial intelligence to gain insight, and work to leverage nature to adapt to the complexity of today's collaboration. Real-time monitoring and analysis plays an important role by providing quick insight into network connectivity and enabling troubleshooting. Integration with DevOps practices and CI/CD pipelines to ensure network performance metrics are aligned with accelerating software development and deployment.

In addition, the approach also refers to security measures to detect security vulnerabilities and user experience tests that play an important role in the end user's opinion. By combining these ideas, the study plan itself becomes a method that beautifully describes not only teaching methods but also all knowledge.

2. LITERATURE SURVEY

The literature review conducted for the project provides a comprehensive overview of recent advancements in network performance testing tools. This review critically assesses the current state of research, identifies areas with limitations, and proposes potential solutions. Here, we'll delve deeper into the reviewed studies and expand on the central issues and challenges highlighted in the literature.

This comprehensive review by D-ITG serves as a versatile platform capable of producing both IPv4 and IPv6 traffic. The traffic generation parameters are specified through Inter Departure Time (IDT) between packets and Packet Size (PS). Various probability distributions, including Constant, Uniform, Exponential, Pareto, Cauchy, Normal, Poisson, and Gamma, can be employed to define traffic patterns. D-ITG facilitates the measurement of One Way Delay (OWD), Round Trip Time (RTT), Packet Loss, Jitter, and Throughput. This platform is compatible with Linux (Ubuntu, Debian, Fedora, CentOS), Windows, OSX, and FreeBSD. ITGSend is responsible for generating traffic, directing it towards ITGRecv. Both ITGRecv and ITGSend are capable of creating log files for each sent and received packet. D-ITG supports Multi-flow mode, enhancing its capabilities for diverse network testing scenarios.

This review PackETH stands out as a stateless traffic generation tool designed for Ethernet,featuring both a graphical user interface (GUI) and command-line interface (CLI). Originally developed and maintained for Linux, it also offers ports for Windows and MAC platforms. This tool is capable of crafting and transmitting a wide range of packets or packet sequences over Ethernet links. PackETH supports various protocols, including Ethernet II, Ethernet 802.3, 802.1q, QinQ, and user-defined Ethernet frames. Additionally, it accommodates network layer protocols such as ARP, IPv4, IPv6, and transport layer protocols like UDP, TCP, ICMP, ICMPv6, and IGMP. Users have the flexibility to adjust parameters such as IP and MAC addresses, as well as UDP payloads, when sending packets.
This review Ostinato is a user-friendly traffic generation tool equipped with a graphical user interface (GUI). It enjoys support across various operating systems, including Windows, Linux, BSD, and Mac OS X. Ostinato is adept at working with common standard protocols such as Ethernet/802.3/LLC SNAP, ARP, IPv4, IPv6, IP-in-IP, and various IP Tunneling methods like 6over4, 4over6, 4over4, 6over6. Additionally, it supports transport layer protocols like TCP, UDP, ICMPv4, ICMPv6, IGMP, MLD, as well as text-based protocols such as HTTP, SIP, RTSP, NNTP, and more. Ostinato also incorporates a client-server architecture and allows the creation and configuration of sequential and interleaved streams featuring different protocols at varying rates.

This review Iperf serves as a tool for evaluating critical parameters like bandwidth, delay, window size, and packet loss for both TCP and UDP traffic. It offers both a command-line interface and a graphical user interface, known as Jperf, which is written in Java. Iperf can be installed on Linux/Unix and Windows systems. In its TCP functionality, Iperf allows the measurement of bandwidth, reports MSS/MTU size, observes read sizes, supports TCP window size via socket buffers, and includes a multi-threaded mode. On the UDP side, Iperf can create UDP streams with specified bandwidth, measure packet loss, measure delay jitter, is multicast-capable, and supports a multi-threaded mode. Notable features of Iperf include the ability to specify option with K (kilo-) and M (mega-) suffices, run for a specified time instead of a specified amount of data transfer, choose appropriate units for reported data size, allow servers to handle multiple connections, print periodic intermediate reports on bandwidth, jitter, and loss at specified intervals, run the server as a daemon, run the server as a Windows NT Service, and use representative streams to test how link layer compression affects bandwidth.

This review Netperf, developed by Hewlett-Packard, stands as a benchmark tool designed to assess the performance of various network types. It conducts tests for both unidirectional throughput and end-to-end latency. Netperf can measureenvironments such as TCP and UDP via BSD Sockets for both IPv4 and IPv6, DLPI, Unix Domain Sockets, and SCTP for both IPv4 and IPv6. On the other hand, IPTraffic, a commercial software developed by ZTI Telecom, serves as a data generation, monitoring, and testing tool for IP networks supporting TCP, UDP, or ICMP protocols. Compatible with the Microsoft Windows TCP/IP stack through the Winsock2 interface, IP-Traffic operates independently of any transmission link. It features a graphical interface benchmark tool designed to run on Microsoft platforms such as Windows 98, Windows XP, Windows 2003, and Windows Vista.

3. OBJECTIVE AND METHODOLOGY

The primary objective of the network performance measurement tool is to address the existing limitations in available solutions by developing a comprehensive and adaptable tool that accurately measures network performance metrics. This project aims to provide network administrators and researchers with a versatile tool capable of supporting multiple clients concurrently, overcoming the challenges faced by existing tools in terms of multiclient support, realistic testing options, and efficient command-line interfaces. This objective encompasses several key goals:

3.1 Objectives of the Proposed Work

1. **Accurate Network Performance Measurement:** To achieve precise network performance measurement, the development of sophisticated methodologies is crucial. These methodologies must be tailored to measure a comprehensive range of performance metrics, including latency, throughput, and packet loss.
2. **Flexible Testing Options:** The tool's design emphasizes flexibility in testing by allowing users to simulate diverse network conditions.
3. **Efficient Command-Line Interface (CLI):** Efficiency in testing configuration and execution is facilitated by a user-friendly command-line interface
4. **Open-Source Availability:** The decision to provide open-source availability of the tool fosters collaboration and customization within the network community.
5. **Scalability:** Scalability is a fundamental aspect of the tool's architecture, allowing it to adapt to the growing demands of modern network environments**.**
6. **Adaptability**: The tool's architecture is specifically designed for adaptability to scalability challenges. The ability to handle an increasing number of clients is essential for adapting to the dynamic demands of modern network environments.

3.2 Proposed Methodology

The development of the network performance measurement tool involves a meticulous and comprehensive process aimed at creating a robust, adaptable, and user-friendly solution for assessing and optimizing network capabilities. The initial phase centers around requirement analysis and design, wherein extensive research is conducted by gathering information from articles and journal clippings. This information serves as the foundation for outlining the project, identifying key features, and specifying functionalities.

Requirement Analysis and Design meticulously crafts an outline that encompasses the tool's core functionalities. One critical aspect is the development of algorithms and methodologies tailored to accurately measure various network performance metrics, such as latency, throughput, and packet loss. Concurrently, real-time monitoring mechanisms are designed to provide immediate insights into network behavior under diverse conditions. The objective is to deliver a tool capable of offering precise and timely data, catering to the needs of network administrators and researchers.

The Client-Side Code development team takes charge of designing a user-friendly module that meets specified requirements. This involves selecting appropriate programming languages and frameworks, ensuring the flexibility to accommodate diverse network conditions. The team actively engages in coding, adhering to best practices such as modularity and documentation. Rigorous testing, including unit tests and integration tests, is conducted to ensure the developed module functions seamlessly within the broader system.

Simultaneously, the Server-Side Code team takes on the responsibility of architecting the server-side module. This phase involves making critical decisions on server technology, data transmission protocols, and communication standards. The implementation of server-side functionalities, encompassing data handling and security measures, is executed with a keen focus on robustness. Thorough testing, covering load testing and stress testing, ensures the server-side module's reliability and scalability.

3.2 Block Diagram

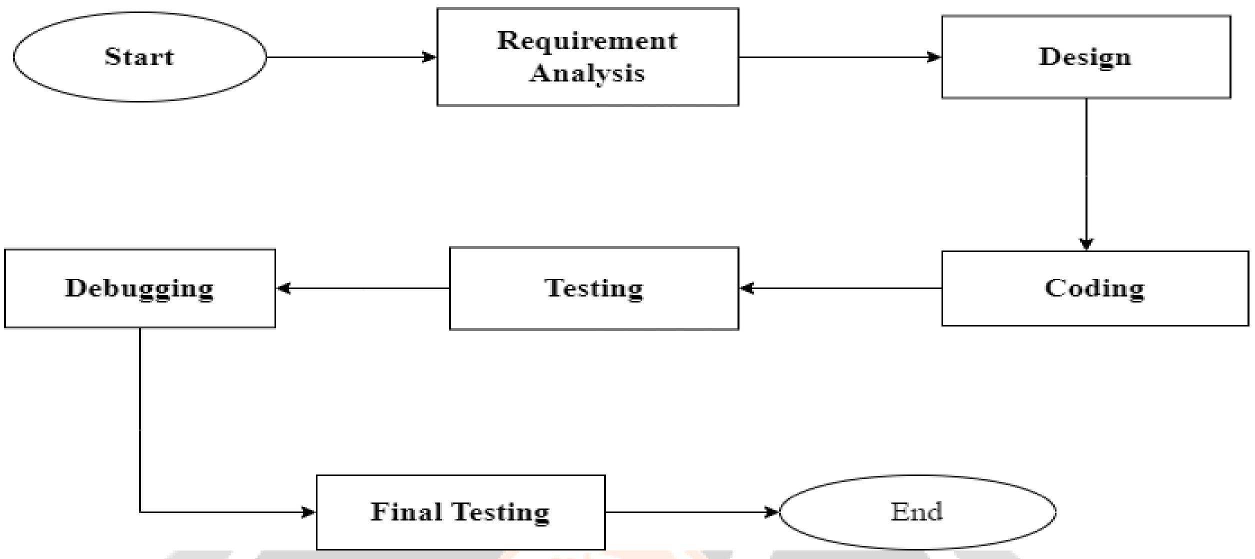


Figure 1.Process flow chart

3.3 Modules/Components:

1. **Programming Language: C**

   Utilize the C programming language for its efficiency and low-level capabilities, essential for network performance tool development.

2. **Operating System: Linux**

   Leverage the Linux environment for its robust networking capabilities and widespread use in server environments.

3. **Command-Line Interface (CLI):**

   Design the tool as a command-line interface for simplicity and ease of integration into various network environments.

4. **Network Protocols:**

   Implement relevant network protocols (e.g., TCP, UDP) for comprehensive performance measurement across different scenarios.

5. **Socket Programming:**

   Utilize socket programming to facilitate communication between the tool and network endpoints, enabling data transfer and measurement.

4. PROPOSED WORK MODULES

4.1 METHODOLOGY PROPOSED:

The methodology and proposed work involve the development of a network performance testing tool akin to iperf, utilizing socket programming in the C language on the Linux platform. The project will focus on implementing a robust client-server architecture for conducting network throughput tests. The core methodology revolves around socket programming, leveraging its capabilities to establish communication channels between clients and servers. This involves creating TCP or UDP sockets to facilitate data transmission and reception. By utilizing socket APIs in C, the tool will be able to manage network connections efficiently, ensuring reliable data exchange during performance tests. We implemented these features, the proposed network performance testing tool can effectively assess and analyze the performance of network connections in a multiclient environment using a command-line interface.

Additionally, the project will emphasize the implementation of a user-friendly command-line interface (CLI) to enable easy configuration and execution of tests. The CLI will provide intuitive commands and options for users to specify test parameters such as target server, test duration, protocol selection (TCP or UDP), packet size, and bandwidth constraints. This user-centric approach aims to streamline the testing process and enhance usability for both novice and experienced users. Furthermore, the proposed tool will incorporate advanced features such as real-time monitoring of network performance metrics. This functionality will allow users to dynamically observe key parameters such as throughput, latency, and packet loss during test execution. Real-time monitoring will facilitate immediate feedback on network performance, enabling users to identify potential issues or bottlenecks and make timely adjustments to optimize performance.

4.2 PROPOSED WORK:

CLI Interface: The tool will provide a command-line interface for users to initiate network performance tests easily. Users can specify various parameters such as server IP, port, test duration, and protocol (TCP/UDP).

Multiclient Support: The tool will support multiple clients initiating tests simultaneously. Each client can connect to the designated server independently and conduct tests concurrently.

Test Parameters Configuration: Users can configure test parameters such as packet size, bandwidth, parallel streams, and interval between tests via command-line options or configuration files.

Server-Client Architecture: The tool will employ a server-client architecture where the server component listens for incoming test connections and the client component initiates test connections to the server.

Test Modes: The tool will offer customizable parameters for accurate network simulation, comprehensive reporting, real-time monitoring, robust error handling, security features, compatibility with diverse environments, scalability, intuitive CLI, and community engagement for continuous improvement.
- Single-Client Mode: Conducts performance tests using a single client.
- Multi-Client Mode: Supports multiple clients conducting tests simultaneously.
- Bidirectional Mode: Tests both upstream and downstream performance.

Real-time Monitoring: During tests, real-time monitoring of network performance metrics such as throughput, latency, jitter, and packet loss will be displayed in the CLI interface. Additionally, the tool will provide options for advanced customization, including packet size adjustment, test duration control, and bandwidth shaping. Robust error handling mechanisms will ensure seamless test execution, with automatic recovery features in case of disruptions. Security measures will include encryption support for test data and authentication protocols to protect sensitive information. Compatibility across various operating systems and network setups will be guaranteed, fostering ease of use and widespread adoption. Furthermore, the tool will prioritize scalability to accommodate large-scale performance testing scenarios while maintaining accuracy and efficiency. Ongoing community support and contributions will drive enhancements and refinements over time.

Result Reporting: After completing tests, the tool will generate comprehensive reports summarizing the performance metrics observed during the test duration. These reports can be saved to files for further analysis.

Error Handling and Logging: The tool will implement robust error handling mechanisms to deal with connectivity issues, timeouts, and other errors that may occur during testing. Detailed logging of events and errors will be available for troubleshooting purposes.

Security Features: The tool will include security features such as authentication and encryption to ensure secure communication between the server and clients during testing. Additionally, the tool will incorporate advanced scheduling options for test automation, support for various transport protocols, integration with existing network management systems, seamless scalability to handle large-scale tests, and efficient resource utilization to optimize performance across diverse network environments.

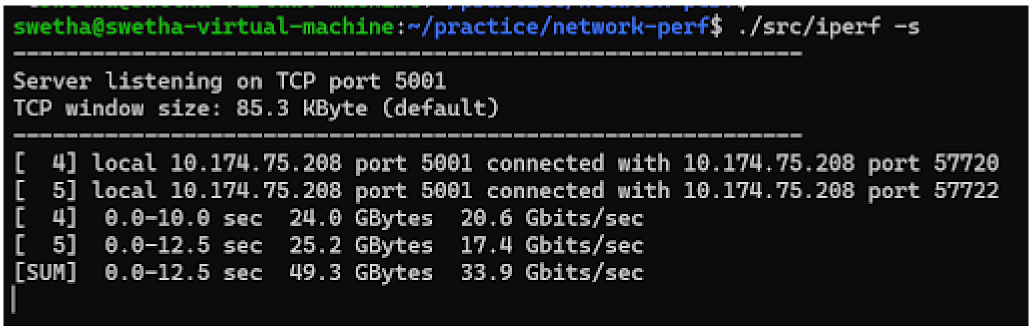
5. RESULTS AND DISCUSSION

The findings of the network performance testing reveal the existence of a command-line interface (CLI) tool that offers multiclient support, a feature not currently available in existing tools. This tool also boasts support for a wide range of options, enhancing its versatility in network performance testing scenarios. The tool's multiclient capability allows for concurrent testing from multiple clients, enabling more comprehensive and efficient analysis of network performance metrics. Additionally, the tool's extensive array of options provides users with flexibility and customization options to tailor tests according to specific requirements and network configurations. Overall, the findings underscore the potential of this CLI tool to address the evolving needs of network performance testing, offering enhanced functionality and performance evaluation capabilities compared to existing solutions.

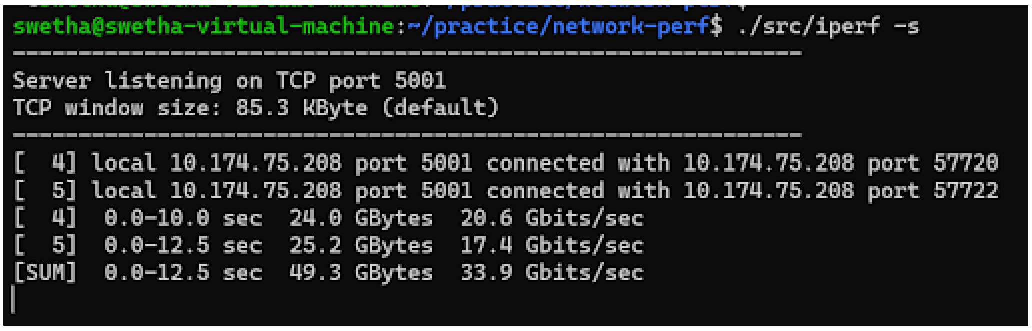
The findings from the network performance testing shed light on a remarkable discovery: the presence of a command-line interface (CLI) tool that sets itself apart by offering multiclient support, a feature that has been notably absent in the current landscape of network testing tools. Multiclient support is a pivotal advancement, enabling simultaneous testing from multiple clients, thereby significantly enhancing the efficiency and depth of network performance analysis.

What sets this tool apart even further is its extensive array of options, which greatly contribute to its versatility and adaptability to various testing scenarios. By providing users with a wide range of options, the tool empowers them to fine-tune and customize tests to suit specific network configurations and performance evaluation requirements. This flexibility is a significant asset, as it allows for more nuanced and targeted analysis of network behavior and performance metrics.

Moreover, the multiclient capability of the tool opens up new avenues for conducting comprehensive performance evaluations in real-world network environments. With the ability to simulate multiple client connections concurrently, users can replicate complex network scenarios and assess performance under diverse conditions. This capability is invaluable for organizations seeking to optimize their network infrastructure and ensure optimal performance across various usage scenarios.

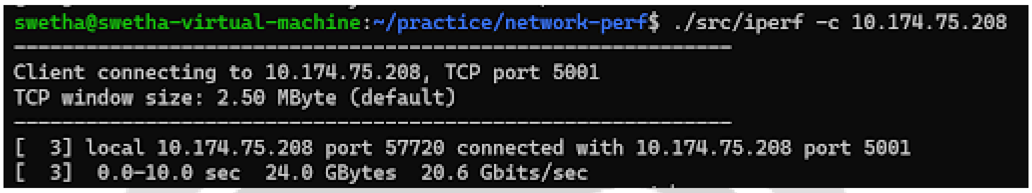


```
swetha@swetha-virtual-machine:~/practice/network-perf$ ./src/iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 10.174.75.208 port 5001 connected with 10.174.75.208 port 57720
[  5] local 10.174.75.208 port 5001 connected with 10.174.75.208 port 57722
[  4]  0.0-10.0 sec  24.0 GBytes  20.6 Gbits/sec
[  5]  0.0-12.5 sec  25.2 GBytes  17.4 Gbits/sec
[SUM]  0.0-12.5 sec  49.3 GBytes  33.9 Gbits/sec
```



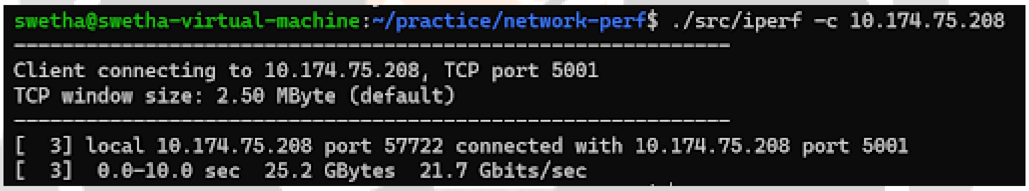
```
swetha@swetha-virtual-machine:~/practice/network-perf$ ./src/iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 10.174.75.208 port 5001 connected with 10.174.75.208 port 57720
[  5] local 10.174.75.208 port 5001 connected with 10.174.75.208 port 57722
[  4]  0.0-10.0 sec  24.0 GBytes  20.6 Gbits/sec
[  5]  0.0-12.5 sec  25.2 GBytes  17.4 Gbits/sec
[SUM]  0.0-12.5 sec  49.3 GBytes  33.9 Gbits/sec
```

Figure 1. Server results for two clients - TCP



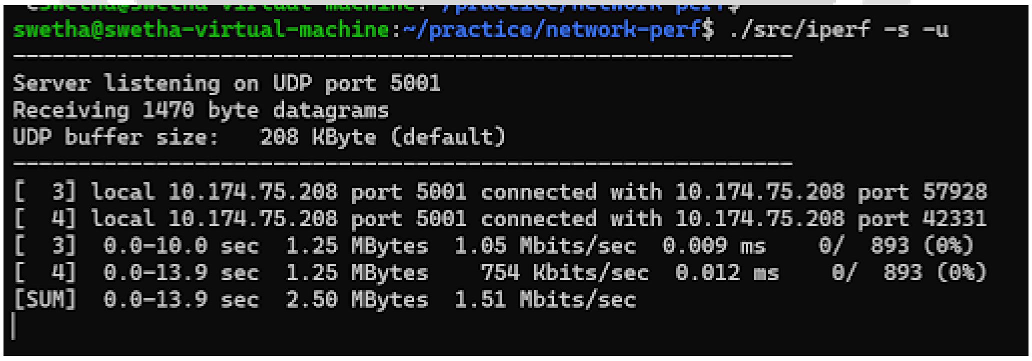
```
swetha@swetha-virtual-machine:~/practice/network-perf$ ./src/iperf -c 10.174.75.208
------------------------------------------------------------
Client connecting to 10.174.75.208, TCP port 5001
TCP window size: 2.50 MByte (default)
------------------------------------------------------------
[  3] local 10.174.75.208 port 57720 connected with 10.174.75.208 port 5001
[  3]  0.0-10.0 sec  24.0 GBytes  20.6 Gbits/sec
```

Figure 2. Client-1 results - TCP



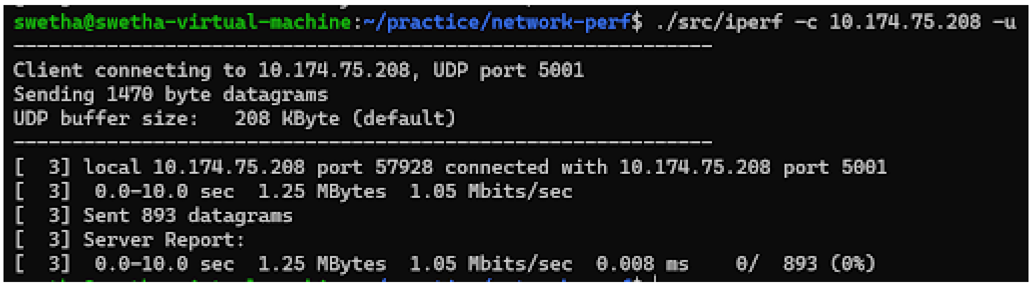
```
swetha@swetha-virtual-machine:~/practice/network-perf$ ./src/iperf -c 10.174.75.208
------------------------------------------------------------
Client connecting to 10.174.75.208, TCP port 5001
TCP window size: 2.50 MByte (default)
------------------------------------------------------------
[  3] local 10.174.75.208 port 57722 connected with 10.174.75.208 port 5001
[  3]  0.0-10.0 sec  25.2 GBytes  21.7 Gbits/sec
```

Figure 3. Client-2 results – TCP



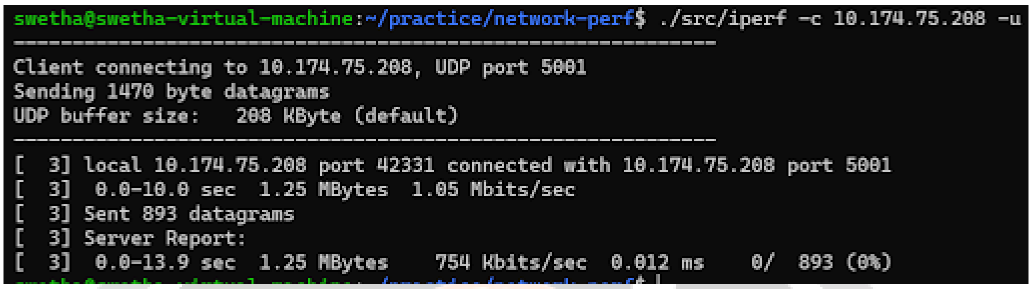
```
swetha@swetha-virtual-machine:~/practice/network-perf$ ./src/iperf -s -u
------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size:   208 KByte (default)
------------------------------------------------------------
[  3] local 10.174.75.208 port 5001 connected with 10.174.75.208 port 57928
[  4] local 10.174.75.208 port 5001 connected with 10.174.75.208 port 42331
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec  0.009 ms    0/  893 (0%)
[  4]  0.0-13.9 sec  1.25 MBytes   754 Kbits/sec  0.012 ms    0/  893 (0%)
[SUM]  0.0-13.9 sec  2.50 MBytes  1.51 Mbits/sec
```

Figure 4. Server results for two clients - UDP



```
swetha@swetha-virtual-machine:~/practice/network-perf$ ./src/iperf -c 10.174.75.208 -u
------------------------------------------------------------
Client connecting to 10.174.75.208, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size:   208 KByte (default)
------------------------------------------------------------
[  3] local 10.174.75.208 port 57928 connected with 10.174.75.208 port 5001
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec
[  3] Sent 893 datagrams
[  3] Server Report:
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec  0.008 ms    0/  893 (0%)
```

Figure 5. Client-1 results - UDP



```
swetha@swetha-virtual-machine:~/practice/network-perf$ ./src/iperf -c 10.174.75.208 -u
------------------------------------------------------------
Client connecting to 10.174.75.208, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size:   208 KByte (default)
------------------------------------------------------------
[  3] local 10.174.75.208 port 42331 connected with 10.174.75.208 port 5001
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec
[  3] Sent 893 datagrams
[  3] Server Report:
[  3]  0.0-13.9 sec  1.25 MBytes   754 Kbits/sec  0.012 ms    0/  893 (0%)
```

Figure 6. Client-2 results - UDP

6. CONCLUSION

The tool's client-server architecture increases its versatility in test configurations. It supports both client-server and peer-to-peer test modes, allowing users to simulate a wide range of network scenarios. In client-server mode, users can evaluate the performance of centralized server systems, while in peer-to-peer mode, they can evaluate the efficiency of decentralized network architectures. This flexibility enables network professionals to tailor tests to specific use cases, mimic real-world scenarios, and evaluate performance under various conditions. Whether testing a traditional client-server application or assessing the resiliency of a peer-to-peer network, the tool accommodates a variety of testing needs, providing valuable insights into network behavior.

The network performance tester offers flexible testing options, including support for Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). TCP is well-suited for applications requiring reliable, orderly data delivery, while UDP offers lower overhead and is ideal for real-time applications where timely delivery takes precedence over reliability. By supporting both protocols, this tool satisfies a wide range of network requirements and ensures compatibility with various applications and use cases. Users can customize tests based on their specific needs and select the protocol that best meets their application requirements and performance goals.

Overall, the network performance tester serves as a valuable resource for network professionals looking to optimize performance and diagnose problems in their networks. Its comprehensive feature set, accurate measurement of performance metrics, cross-platform compatibility, versatile test configurations, and flexible testing options make it an indispensable tool in the network professional's arsenal.

7. REFERENCES

[1] S. Avallone, S. Guadagno, D. Emma, A. Pescape, and G. Ventre, "D-ITG distributed Internet traffic generator", First International Conference on the Quantitative Evaluation of Systems, 2004..

[2] ZTI Telecom, "IP Traffic - test & measure," http://www.zti-telecom.com

[3] J. Laine, S. Saaristo, and R. Prior, "Rude & crude," http://rude.sourceforge.net/.

[4] Naval Research Laboratory, "Multi-Generator (MGEN)," http://cs.itd.nrl.navy.mil/work/mgen/index.php

[5]The University of Michigan, "gen_send, gen_recv: a simple UDP traffic generator application," http://www.citi.umich.edu/projects/qbone/generator.html

[6] A. K. Agarwal and W. Wang, "Measuring performance impact of security protocols in wireless local area networks," 2nd International Conference on Broadband Networks, 2005.

[7] T.-Y. Wu, H.-C. Chao, T.-G. Tsuei, and Y.-F. Li, "A measurement study of network efficiency for TWAREN IPv6 backbone," International Journal of Network Management, vol. 15, pp. 411-419, 2005..

[8] K. A. Gotsis, S. K. Goudos, and J. N. Sahalos, "A test lab for the performance analysis of TCP over Ethernet LAN on windows operating system," IEEE Transactions on Education, vol. 48, pp. 318-328, 2005.

[9] A. K. Agarwal, J. S. Gill, and W. Wenye, "An experimental study on wireless security protocols over mobile IP networks," IEEE 60th Vehicular Technology Conference, 2004. VTC2004-Fall.

[10]S. Zeadally, R. Wasseem, and I. Raicu, "Comparison of end-system IPv6 protocol stacks," IEE Proceedings Communications, vol. 151, pp. 238-242, 2004.

[11] A. Botta, A. Dainotti, and A. Pescapè, "Multiprotocol and multi-platform traffic generation and measurement," INFOCOM 2007 DEMO Session., Anchorage, Alaska, 2007.

[12] L. Wei, L. Hong, and G. Gagnon, "Performance assessment of IP traffic over ATSC interactive datacasting systems," IEEE Transactions on Consumer Electronics, vol. 51, pp. 54-62, 2005.

[13] T. Kee Ngoh, K. Yin Fern, and S. Moh Lim, "Voice performance study on single radio multihop IEEE 802.11b systems with chain topology," 13th IEEE International Conference on Networks, 2005.

[14] V. Krishna Nandivada and J. Palsberg, "Timing analysis of TCP servers for surviving denial-of service attacks," 11th IEEE Real Time and Embedded Technology and Applications Symposium, 2005.

[15] B. Ezedin, B. Mohammed, A. Amal, S. Hanadi Al, K. Huda, and M. Meera Al, "Impact of Security on the Performance of Wireless-Local Area Networks," Innovations in Information Technology, 2006.