

Design, Implementation, and Performance Evaluation of a Low-Cost Edge Cloud Drive and Automated Storage System Based on Raspberry Pi

Thanatip Chompupuen¹, Borwornwatchara Khamhanphon¹, Kietirat Yongpawisai¹, Chitapong Wechtaisong^{1*}, Wichai Srisuluck¹, Thanaphon Sakonphattharasorn²

¹Institute of Engineering, Suranaree University of Technology, Nakhon Ratchasima, Thailand

²Faculty of Industrial Technology, Nakhon Rachasima Rajabhat University, Nakhon Ratchasima, Thailand

Corresponding Author: chitapong@g.sut.ac.th

Abstract

Public cloud storage services impose constraints regarding limited free storage, recurring subscription costs, privacy risks, and restricted system customizability. This study presents a compact personal Network-Attached Storage (NAS) system built on a Raspberry Pi 4 Model B with 4GB RAM, integrated with an Argon ONE M.2 Case and a FIKWOT FN203 512GB M.2 SATA III SSD. The software architecture utilizes OpenMediaVault (OMV) with an EXT4 file system, Tailscale Zero-Config Mesh VPN for encrypted remote access without port forwarding, and n8n for event-driven workflow automation to extract file metadata and deliver instant email notifications upon file uploads. Network bandwidth evaluation using iperf3 over Gigabit LAN yielded average upload and download rates of 950 Mbit/s and 945 Mbit/s, respectively. Raw sequential storage I/O benchmarks with fio using a 20GB workload achieved average read and write throughput of 339 MB/s and 260 MB/s, respectively. Local SMB file transfers peaked at 111.1 MB/s, reaching approximately 88.8% of the theoretical Gigabit Ethernet throughput limit. SLC cache analysis showed sustained write performance of 250 MB/s up to 50GB before declining to 85–90 MB/s beyond 80GB due to direct writing to NAND flash. Remote SMB file transfers via Tailscale VPN attained throughput between 17.5 MB/s and 20.0 MB/s during remote operations. The proposed system offers a cost-effective, secure, and performant personal edge cloud solution operating near hardware throughput limits

Keywords: Raspberry Pi 4; Network-Attached Storage (NAS); OpenMediaVault; Tailscale VPN; n8n Automation; Storage Benchmarking; Edge Cloud Drive.

Date of Submission: 22-09-2026

Date of acceptance: 02-10-2026

I. INTRODUCTION

Digital data management across academic, professional, and personal environments relies heavily on commercial public cloud storage services, such as Google Drive and Dropbox, to facilitate centralized file storage, synchronization, and multi-platform accessibility. Despite their widespread commercial adoption, public cloud platforms present major operational constraints, including restricted free storage capacity, recurring subscription expenses, privacy risks associated with third-party remote servers, and limited system customizability. Furthermore, proprietary public cloud architectures prevent administrative control over access control mechanisms, file-sharing protocols, and automated workflow processing.

Digital data management across academic, professional, and personal environments relies heavily on commercial public cloud storage services, such as Google Drive and Dropbox, to facilitate centralized file storage, synchronization, and multi-platform accessibility. Despite their widespread commercial adoption, public cloud platforms present major operational constraints, including restricted free storage capacity, recurring subscription expenses, privacy risks associated with third-party remote servers, and limited system customizability. Furthermore, proprietary public cloud architectures prevent administrative control over access control mechanisms, file-sharing protocols, and automated workflow processing.

To overcome these technical obstacles, this study presents the design, implementation, and empirical performance evaluation of an automated, low-cost edge cloud drive system based on the Raspberry Pi platform. The physical architecture pairs a Raspberry Pi board with an Argon One M.2 case and an M.2 SATA Solid-State Drive to eliminate storage interface bottlenecks while providing active and passive thermal cooling. The software environment utilizes OpenMediaVault on an EXT4 filesystem to manage centralized file storage, user

authentication, and Server Message Block file sharing. Secure off-site remote access is established without complex router configuration or open firewall ports by integrating a Zero-Configuration Mesh Virtual Private Network powered by Tailscale and the WireGuard cryptographic protocol. Furthermore, an event-driven automation framework using the n8n platform is integrated to perform real-time storage directory monitoring, metadata extraction, and automated email notification dispatches.

The primary objective of this investigation is to demonstrate the feasibility, security, and high throughput of a fully automated personal cloud drive hosted on consumer microcomputing hardware. The principal engineering contributions of this work encompass developing a thermally managed hardware assembly using an M.2 expansion case, evaluating raw storage and network throughput metrics near hardware transmission limits, assessing Single-Level Cell cache exhaustion under continuous heavy write workloads, establishing secure zero-configuration remote access without exposing public network ports, and deploying an automated event-driven workflow notification pipeline.

The remainder of this manuscript is organized systematically to detail the design, implementation, and empirical evaluation of the edge cloud drive system. Section two presents foundational theories and related research covering single-board computing, Network-Attached Storage protocols, OpenMediaVault administration, WireGuard mesh networking, and n8n workflow automation. Section three outlines the hardware interconnects, thermal enclosure assembly, software environment setup, and automation pipeline implementation. Section four presents the empirical experimental methodology, network bandwidth evaluation, storage throughput benchmarks, cache exhaustion analysis, remote access testing, and workflow response validation. Section five provides an engineering discussion regarding system efficiency, hardware bottlenecks, operational challenges, and practical recommendations. Section six concludes the manuscript and summarizes future research directions.

II. RELATED WORK AND THEORETICAL FOUNDATION

2.1 SINGLE-BOARD COMPUTER (SBC) STORAGE ARCHITECTURE

The architectural evolution of Single-Board Computers (SBCs) has positioned compact microcomputing devices as viable candidates for continuous edge server deployments. The Raspberry Pi Foundation specifications [1] detail how the Raspberry Pi four Model B represents a major architectural upgrade over prior generations by featuring a Quad-core ARM Cortex-A72 processor, up to eight Gigabytes of system memory, and a dedicated Gigabit Ethernet controller. Unlike earlier single-board iterations where network traffic and USB storage shared a constrained USB bus, the modern board incorporates an independent PCIe-backed USB host controller alongside a physical network controller. This architectural decoupling eliminates historical bus multiplexing bottlenecks, enabling simultaneous high-speed disk operations and full-duplex network communications. Practical evaluations by Geerling [2] demonstrate that modern microcomputing platforms possess sufficient computational capabilities to support Network-Attached Storage workloads effectively.

2.2 NAS OPERATING SYSTEMS AND STORAGE PROTOCOLS

Centralized network storage relies on specialized management systems to govern physical storage media, user authentication, and network file-sharing protocols. OpenMediaVault [3] provides an open-source administration solution built upon Debian Linux that offers a lightweight web-based administrative interface for storage management. Within Network-Attached Storage environments, file sharing is governed by distinct application-layer network communications protocols:

- **Server Message Block (SMB):** Developed originally for Windows environments, SMB provides transparent file and folder access across local networks while supporting granular authentication, Linux Access Control Lists, and transport-layer encryption. Through open-source implementations such as Samba, SMB enables cross-platform compatibility across Windows, macOS, and Linux workstations.
- **Network File System (NFS):** Developed originally for Unix platforms, NFS offers a low-overhead, high-throughput protocol tailored primarily for Linux environments. While NFS delivers superior throughput for computational clusters, its client-side deployment on native Windows machines requires specialized subsystem installation.
- **File Transfer Protocol (FTP/SFTP):** Standard FTP facilitates client-server file transfers but transmits credentials in cleartext format. Secure FTP addresses this vulnerability by encapsulating transactions within an encrypted Secure Shell tunnel, making it well-suited for batch file transfers and automated remote backups.

Storage management in OpenMediaVault [3] natively supports the Fourth Extended File System, which is widely favored in Linux edge environments due to its battle-tested journaling capabilities, operational efficiency, and low microprocessor overhead.

2.3 OVERLAY MESH VPN AND CRYPTOGRAPHIC REMOTE ACCESS

Establishing secure remote access to local network storage across wide area networks traditionally requires router port forwarding, static public IP provisioning, or Dynamic DNS configuration. However, security guidelines from NSTDA [4] emphasize that exposing open inbound ports to public IP space incurs severe security risks from automated port scanning, brute-force intrusions, and unauthorized access attempts. Modern edge architectures mitigate these threats through Zero-Configuration Mesh Virtual Private Networks. Tailscale [5] implements an overlay network architecture based on the WireGuard cryptographic protocol. By utilizing Interactive Connectivity Establishment and NAT Traversal algorithms, Tailscale [5] establishes point-to-point encrypted tunnels between authenticated nodes without requiring firewall port modifications or complex network configurations.

2.4 EVENT-DRIVEN WORKFLOW AUTOMATION

To enhance user experience and operational visibility, contemporary storage platforms integrate event-driven workflow automation frameworks. Documentation from n8n [6] describes an open-source, node-based automation engine operating on a fair-code model. The system architecture relies on two primary node classifications:

- **Trigger Nodes:** These nodes serve as workflow initiators that execute upon registering specific system events, such as local filesystem changes, incoming webhooks, or scheduled time intervals.
- **Regular Nodes:** These processing units execute sequential operations, including data parsing, shell command execution, JSON field formatting, and external email dispatching.

In an edge storage context, binding directory-watching triggers to automated processing pipelines [6] allows real-time file inspection, metadata extraction, and multi-channel administrator alerting without requiring continuous manual intervention.

2.5 REVIEW OF BENCHMARKING FRAMEWORKS

Evaluating the performance of an edge storage system requires standardized, repeatable benchmarking methodologies across both network and storage subsystems:

- **Network Bandwidth Testing (iperf3):** Technical guidelines from NOC RMUTP [7] explain how the iperf3 benchmarking framework operates in client-server mode to measure maximum achievable network bandwidth across physical interfaces. By analyzing throughput rates across local area networks, iperf3 quantifies network interface saturation and identifies protocol overhead losses.
- **Storage I/O Testing (fio):** Technical documentation provided by Axboe [8], Red Hat [9], and Nutanix [10] establishes the Flexible I/O Tester as the industry standard utility for evaluating storage hardware performance. It enables precise workload customization, including sequential read and write throughput, random operations, block size allocation, thread concurrency, and direct memory buffering. Employing fio benchmarks allows engineers to evaluate hardware throughput limits, measure queue depth responses, and identify Single-Level Cell cache exhaustion points during sustained write operations.

III. SYSTEM ARCHITECTURE AND IMPLEMENTATION

3.1 HARDWARE INTERCONNECT AND THERMAL DESIGN

The physical foundation of the proposed edge cloud drive system is engineered to deliver sustained data throughput while mitigating thermal throttling under continuous disk operations. The central processing unit is a Raspberry Pi 4 Model B equipped with four Gigabytes of Low-Power Double Data Rate 4 system memory and a Quad-core ARM Cortex-A72 microprocessor. The single-board computer incorporates an onboard RJ-45 Gigabit Ethernet controller for high-speed local network communication alongside USB 3.0 host ports connected via an internal PCIe bus.

To maintain structural rigidity and efficient heat dissipation, the board is housed within an Argon ONE M.2 enclosure manufactured from aluminum alloy. The enclosure acts as a passive heatsink in direct contact with the central processor and wireless chipset using high-performance thermal interface pads. Active thermal management is provided by an integrated software-controlled cooling fan that dynamically adjusts its rotational speed based on real-time microprocessor temperature readings. System storage capacity is supplied by a 512-Gigabyte FIKWOT FN203 M.2 SATA III Solid-State Drive installed into the M.2 expansion board at the base of the chassis. Data transmission between the Raspberry Pi host board and the M.2 SATA expansion board is established through an external USB 3.0 bridge adapter, bypassing traditional micro-SD card speed limits. System power is supplied by a dedicated 5-Volt 3-Ampere USB-C adapter to maintain stable electrical current during simultaneous drive activity and peak computation.

The step-by-step physical assembly procedure is executed as follows:

- **Interface Board Attachment:** The secondary HDMI-Audio daughterboard is mounted onto the Raspberry Pi 4 board to align video and audio physical output ports with the rear casing.
- **Thermal Pad Application:** High-conductivity thermal pads are placed directly onto the top surfaces of the central processing unit and Broadcom chipset components.
- **Top Enclosure and GPIO Coupling:** The Raspberry Pi 4 mainboard is seated into the top half of the Argon ONE casing, aligning external physical ports while coupling the forty-pin GPIO adapter board to the host GPIO header pins.
- **Board Fastening:** The primary circuit board is secured firmly to the aluminum chassis using small retention screws.
- **SSD Installation:** The FIKWOT M.2 SATA III Solid-State Drive is inserted into the M.2 slot on the lower expansion board at a thirty-degree angle and secured horizontally with a mounting screw.
- **Chassis Consolidation:** The upper aluminum housing and lower M.2 expansion assembly are mated together and secured using exterior chassis screws.
- **Bridge Interconnection:** The physical data link between the host Raspberry Pi 4 USB 3.0 port and the M.2 SATA expansion board USB input is established by attaching the USB 3.0 bridge connector.

The physical wiring and power architecture establishes a clear electrical and data flow. Power enters through the 5-Volt 3-Ampere USB-C power supply into the Argon ONE power control board, which distributes electrical current to the cooling fan, GPIO pins, Raspberry Pi mainboard, and the M.2 SATA expansion board. Data transactions move from the FIKWOT Solid-State Drive through the SATA controller, across the USB 3.0 bridge adapter, into the Raspberry Pi host processor, and out through the Gigabit Ethernet port over a CAT5e network cable connected to a local network router.

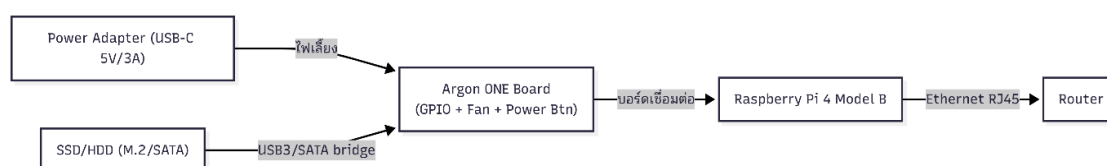


Figure 1: System Interconnect and Electrical Power Wiring Diagram

3.2 SOFTWARE ENVIRONMENT AND STORAGE SETUP

The software environment is constructed on top of a 64-bit headless Raspberry Pi Operating System derived from Debian Linux. Operating system deployment is performed using the official Raspberry Pi Imager desktop utility. During the image flashing process, pre-configuration parameters are specified within the utility settings:

- Setting the system network hostname to raspberrypi for localized domain identification.
- Defining administrative username and password credentials for initial authentication.
- Configuring wireless local area network credentials, including network SSID, passphrase, and regional wireless parameters set to Thailand.
- Configuring regional localization parameters, including setting the system time zone to Asia/Bangkok time.
- Enabling the Secure Shell (SSH) background daemon to permit headless remote terminal access.

Upon completing image creation, the target storage medium is inserted into the server, and power is applied. Initial administrative access is established from a Windows client workstation using the Command Prompt command `ssh username@IP-Address`. Upon verifying the Secure Shell host security key, administrative credentials are entered to obtain interactive command-line interface control. The underlying operating system packages are subsequently upgraded to current baselines using the command `sudo apt update && sudo apt upgrade -y` within the terminal environment.

Storage administration is orchestrated using OpenMediaVault (OMV), an open-source Network-Attached Storage management platform. OpenMediaVault is installed onto the running operating system by executing the official installation script using the command `wget -O - https://github.com/OpenMediaVault-Plugin-Developers/installScript/raw/master/install | sudo bash` in the command prompt. Following installation completion, the administrative web interface is accessible via web browser at the assigned local IP address using default administrative credentials.

System storage configuration within the OpenMediaVault dashboard proceeds through structured setup phases:

- **Disk Identification:** Navigating to the Disks administrative menu to verify physical detection of the connected FIKWOT Solid-State Drive.
- **Filesystem Creation:** Initializing the drive by creating a Fourth Extended File System (EXT4) partition, selected for its journaled integrity and operational stability within Linux server environments.
- **Filesystem Mounting:** Mounting the formatted EXT4 partition into the system hierarchy to enable active directory storage operations.
- **Shared Folder Provisioning:** Creating designated shared storage directories, such as the Inbox folder, and applying access permissions using Linux Access Control Lists (ACLs).
- **User and Group Management:** Defining individual user accounts and assigning access permissions to dictate read and write capabilities across shared directories.
- **Protocol Service Activation:** Enabling the Server Message Block (SMB/CIFS) network file sharing service within OpenMediaVault settings and adding the created shared folders to the active SMB shares list.

Client access from local Windows workstations is established through Windows File Explorer by entering the network UNC path `\\IP-Address\ShareName`. Authenticating with valid OpenMediaVault credentials mounts the remote directory transparently, enabling users to copy, edit, and delete files as if stored locally.

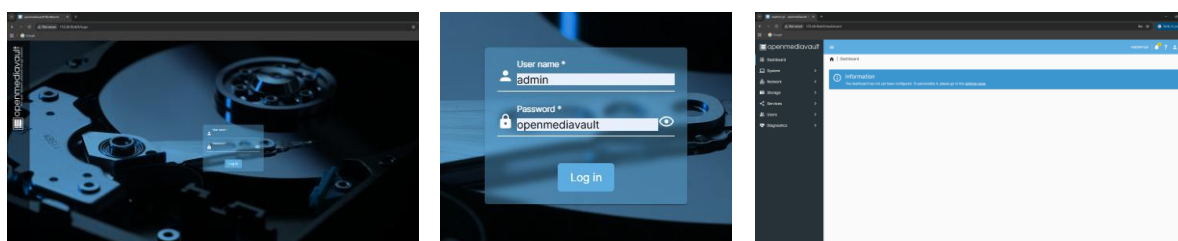


Figure 2: OpenMediaVault Web Administrative GUI and Disk Management Interface

3.3 SECURE REMOTE GATEWAY CONFIGURATION

To enable secure remote file access across external public networks without exposing internal storage endpoints, the system integrates Tailscale mesh VPN software. Tailscale implements an encrypted overlay network utilizing the WireGuard protocol to establish peer-to-peer tunnels.

The Tailscale software daemon is installed onto the Raspberry Pi operating system command line interface using official distribution package repositories. Service initialization and network node registration are executed by running the terminal command `sudo tailscale up` within the shell environment. The command outputs a secure Single Sign-On (SSO) authentication URL, allowing the administrator to authorize the node using Google or Microsoft identity providers.

Upon successful authentication, the Raspberry Pi node automatically joins the user's private overlay network and receives a dedicated, static Tailscale IPv4 address. The background service utilizes Interactive Connectivity Establishment (ICE) algorithms and Network Address Translation (NAT) traversal techniques to

establish direct encrypted peer-to-peer connections with authenticated client devices, such as remote laptops and smartphones running the Tailscale client application. This configuration bypasses complex router port forwarding, eliminates public static IP requirements, and shields the storage server from public internet exposure. Remote clients can access the OpenMediaVault web management interface and mount SMB shared folders over cellular networks or public Wi-Fi connections using the dedicated Tailscale IP address.

3.4 WORKFLOW AUTOMATION INTEGRATION

Event-driven workflow processing and real-time administrator alerting are implemented using n8n, an open-source node-based workflow automation engine running within the host software environment. The automation architecture is designed to continuously monitor inbound storage directories, extract relevant file metadata, parse raw system metrics, and dispatch formatted email notifications upon file upload completion.

The node-based automation pipeline consists of five interconnected functional nodes:

- **Local File Trigger Node:** Acts as the workflow initiator by polling the designated target directory (e.g., the Inbox shared folder) for filesystem creation events triggered when a new file arrives.
- **Execute Command Node:** Spawns system utility commands upon event detection to collect file metadata, capturing attributes such as exact filename, file size in bytes, modification timestamp (mtime), and remaining storage capacity.
- **Code Node:** Receives raw command line text outputs and executes custom JavaScript logic to parse, sanitize, and convert raw text strings into structured JSON objects.
- **Edit Fields Node:** Maps parsed JSON attributes into standardized notification fields, organizing parameters including user identifier, upload timestamp, filename, and human-readable file size metrics.
- **SMTP Send Email Node:** Connects securely to an external Simple Mail Transfer Protocol server to dispatch formatted HTML email alerts containing full upload summaries directly to registered user email addresses.

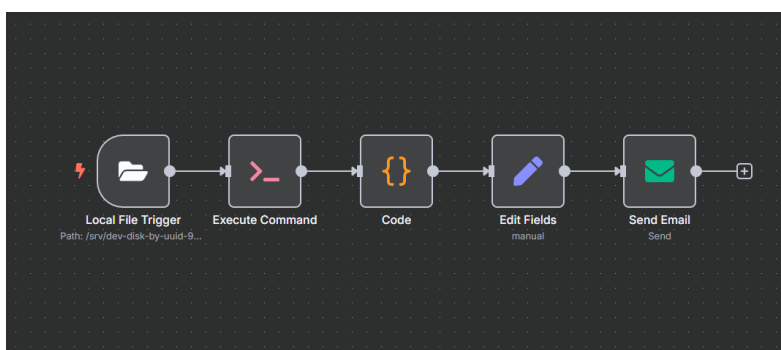


Figure 3: Event-Driven Node Automation Workflow Diagram in n8n

3.5 End-to-End System Workflow

The comprehensive operational sequence of the edge cloud drive system integrates hardware components, storage services, secure networking layers, and event-driven automation pipelines into a unified functional lifecycle.

When power is applied, the Raspberry Pi boots the 64-bit operating system, automatically mounts the EXT4 partition on the FIKWOT Solid-State Drive, and initializes the OpenMediaVault management daemon alongside SMB file-sharing services. Concurrently, the Tailscale background service establishes an encrypted mesh tunnel to the overlay network, enabling remote connection readiness. Local users connect to shared folders over local Gigabit LAN, while remote users connect securely via Tailscale VPN.

When an authorized user copies or uploads a file into the monitored Inbox shared folder, the Linux kernel filesystem layer registers the write transaction. The n8n automation engine detects the new file via its Local File Trigger node, launching the metadata processing pipeline. The Execute Command node retrieves file size, timestamp, and remaining disk capacity details, which are structured into JSON by the Code node and formatted

by the Edit Fields node. Finally, the Send Email node transmits a detailed notification message over SMTP to the user's inbox, while system execution logs record the completed transaction for administrative auditing purposes.

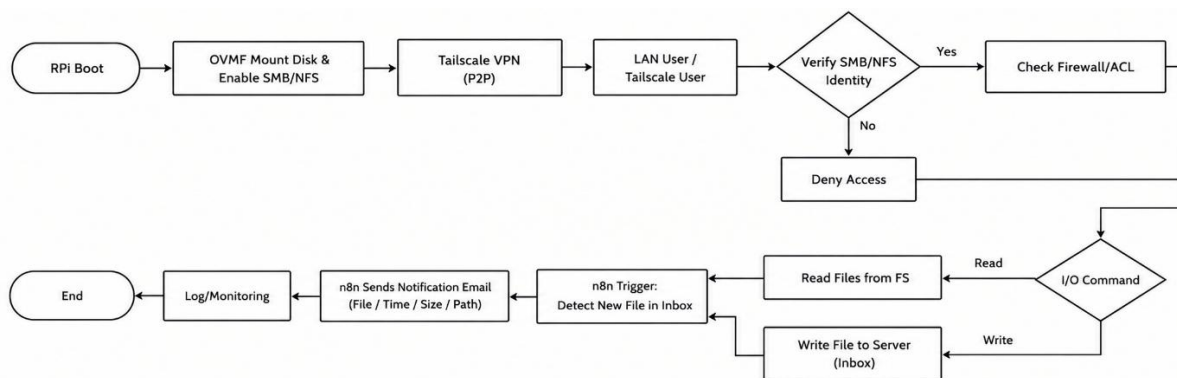


Figure 4: End-to-End System Operational Lifecycle Flowchart

IV. PERFORMANCE EVALUATION AND EMPIRICAL RESULTS

4.1 EXPERIMENTAL METHODOLOGY AND SETUP

To rigorously evaluate the operational capabilities, throughput limits, and system stability of the proposed edge cloud drive, a standardized experimental testbed environment was established. The physical testing infrastructure comprised the core Raspberry Pi 4 Model B server equipped with four Gigabytes of LPDDR4 memory, an Argon ONE M.2 thermal enclosure, and a 512-Gigabyte FIKWOT FN203 M.2 SATA III Solid-State Drive connected via an external USB 3.0 bridge adapter. The client side consisted of a dedicated Windows workstation connected to the edge server across a local area network using CAT5e Ethernet cabling and a Gigabit Ethernet router. Remote network testing was conducted across public internet connections utilizing Wi-Fi and 5G cellular networks.

The empirical evaluation framework was organized into three primary operational categories:

- **Raw Network Throughput Measurement:** Executed using the iperf3 benchmarking framework operating in a client-server model to quantify physical network interface saturation and TCP packet overhead.
- **Raw Storage Block I/O Performance:** Conducted using the Flexible I/O Tester utility, to measure sequential read and write speeds, file fragmentation impacts, and Single-Level Cell cache exhaustion behavior under varying workload volumes.
- **Application-Layer File Transfers and Automation Responsiveness:** Evaluated by timing real-world Server Message Block file transfers over local Gigabit LAN and remote Tailscale VPN connections, while monitoring event response times of the n8n workflow engine.

4.2 RAW NETWORK BANDWIDTH EVALUATION (iperf3 BENCHMARK)

Network bandwidth benchmarking was conducted using iperf3 to establish the baseline throughput of the Raspberry Pi 4 Gigabit Ethernet controller prior to testing storage protocol overheads. The iperf3 server daemon was executed on the Raspberry Pi host while the client application initiated five sequential ten-second TCP stream iterations from the connected Windows workstation.

The empirical results from the five testing iterations demonstrated consistent network interface saturation:

- **Upload Bandwidth (Client Workstation to Edge Server):** Achieved individual iteration speeds of 947 Mbit/s, 950 Mbit/s, 955 Mbit/s, 948 Mbit/s, and 952 Mbit/s, resulting in an average upload throughput of 950 Mbit/s.
- **Download Bandwidth (Edge Server to Client Workstation):** Yielded individual iteration speeds of 940 Mbit/s, 945 Mbit/s, 950 Mbit/s, 943 Mbit/s, and 948 Mbit/s, resulting in an average download throughput of 945 Mbit/s.

An engineering analysis of these results indicates that the physical network link operates at approximately ninety-five percent of the theoretical one thousand Megabit per second Gigabit Ethernet limit. The remaining five percent discrepancy is attributed to standard TCP/IP protocol encapsulation overhead, frame preamble headers, and packet inter-frame spacing rather than processing bottlenecks on the Raspberry Pi board. These metrics confirm that the network controller and physical CAT5e medium provide a high-bandwidth conduit capable of supporting unthrottled storage operations.

4.3 STORAGE I/O BENCHMARKING (fio BENCHMARK) AND SLC CACHE EXHAUSTION ANALYSIS

Storage hardware performance was evaluated using the fio benchmark utility to measure direct block-level read and write throughput on the Fourth Extended File System partition,. To analyze the impact of file fragmentation, benchmarks were executed across two distinct twenty-Gigabyte workload structures: a single continuous twenty-Gigabyte file and a batch of one hundred and two smaller files totaling twenty Gigabytes.

The empirical metrics derived from five testing cycles revealed high block-level performance:

- **Single Continuous File Workload (1 x 20GB):** The Solid-State Drive achieved an average sequential read speed of 339 MB/s with a peak reading of 347 MB/s, and an average sequential write speed of 260 MB/s with a peak writing of 271 MB/s.
- **Multiple Fragmented Files Workload (102 x 20GB Total):** The drive achieved an average sequential read speed of 342 MB/s with a peak reading of 345 MB/s, and an average sequential write speed of 254 MB/s with a peak writing of 268 MB/s.

Comparative analysis indicates that sequential read performance remains virtually identical at approximately 340 MB/s regardless of whether data is structured as a single continuous file or distributed across multiple files. Conversely, write throughput experienced a minor reduction from 260 MB/s to 254 MB/s during multi-file operations. This slight decrease is caused by the administrative overhead of creating, updating, and closing multiple filesystem inode pointers within the Linux kernel during active write cycles.

Table 1: Consolidated Baseline Hardware Performance Metrics for Network Bandwidth (iperf3) and Raw Storage I/O (fio)

Subsystem / Evaluation Metric	Test Workload / Parameter	Unit	Test 1	Test 2	Test 3	Test 4	Test 5	Mean
Network Bandwidth (iperf3)	Upload (PC Client → Raspberry Pi)	Mbit/s	947	950	955	948	952	950
	Download (Raspberry Pi → PC Client)	Mbit/s	940	945	950	943	948	945
Storage Sequential Read (fio)	Single Continuous File (1 x 20 GB)	MB/s	344	347	341	322	341	339
	Multiple Fragmented Files (102 x 20 GB Total)	MB/s	339	343	344	345	341	342
Storage Sequential Write (fio)	Single Continuous File (1 x 20 GB)	MB/s	271	266	244	263	259	260
	Multiple Fragmented Files (102 x 20 GB Total)	MB/s	268	258	245	252	249	254

To investigate hardware limitations under sustained write stress, a dedicated Single-Level Cell cache exhaustion test was conducted using fio across incremental file sizes ranging from ten Gigabytes to one hundred Gigabytes. Solid-State Drives employ high-speed pseudo-SLC cache buffers to accelerate initial write operations before committing data to primary Multi-Level or Triple-Level Cell NAND flash memory.

The experimental observations revealed three distinct operational phases:

- **SLC Cache Buffered Phase (10GB to 40GB Workloads):** Write speeds remained consistently high, recording 260 MB/s at 10GB, 259 MB/s at 20GB, 245 MB/s at 30GB, and 252 MB/s at 40GB, demonstrating active cache buffering.
- **Cache Exhaustion Transition Phase (50GB to 70GB Workloads):** As write volumes exceeded fifty Gigabytes, write throughput dropped to 178 MB/s at 50GB, 168 MB/s at 60GB, and 160 MB/s at 70GB, indicating partial cache saturation.
- **Direct NAND Writing Phase (80GB to 100GB Workloads):** Beyond eighty Gigabytes, the pseudo-SLC buffer became fully exhausted, causing write throughput to stabilize at the native NAND flash write speed of 87 MB/s at 80GB, 90 MB/s at 90GB, and 85 MB/s at 100GB.

Throughout the write cache exhaustion trial, read throughput remained exceptionally stable between 321 MB/s and 347 MB/s across all workload sizes, demonstrating that read operations are entirely decoupled from write cache saturation states.

4.4 REAL-WORLD SMB FILE TRANSFER BENCHMARKS

While synthetic benchmarks establish hardware boundaries, practical utility depends on application-layer file transfer efficiency. The maximum theoretical throughput of a standard Gigabit Ethernet connection operating under IEEE standards is calculated by converting Megabits to Megabytes per second, yielding a theoretical cap of one hundred and twenty-five Megabytes per second. Real-world application benchmarks were executed by copying twenty-Gigabyte test files between the Windows client workstation and the OpenMediaVault SMB share over local Gigabit LAN.

Table 2: Real-World Application-Layer SMB File Transfer Performance over Gigabit Ethernet (20 GB Total Workload)

Transfer Operation / Direction	File Structure & Workload	Total Data Volume	Elapsed Time (mm:ss.ms)	Average Throughput (MB/s)	Throughput Efficiency (% of 125 MB/s Cap) ¹²
Write (PC Client → Edge NAS)	Single Continuous File (1 File)	20 GB	03:04.40	111.1	88.88%
	Multiple Fragmented Files (102 Files)	20 GB	03:11.79	104.4	83.52%
Read (Edge NAS → PC Client)	Single Continuous File (1 File)	20 GB	03:33.35	95.8	76.64%
	Multiple Fragmented Files (102 Files)	20 GB	03:34.23	95.3	76.24%

The real-world file transfer metrics demonstrated high network utilization across all operational modes:

- **Write Operations (PC to Edge NAS):** Transferring a single continuous twenty-Gigabyte file required three minutes and four seconds, achieving an average transfer speed of 111.1 MB/s. Transferring one hundred and two files required three minutes and eleven seconds, achieving an average transfer speed of 104.4 MB/s.
- **Read Operations (Edge NAS to PC):** Downloading a single continuous twenty-Gigabyte file required three minutes and thirty-three seconds, yielding an average transfer speed of 95.8 MB/s. Downloading one hundred and two files required three minutes and thirty-four seconds, yielding an average transfer speed of 95.3 MB/s.

The upload transfer speed of 111.1 MB/s represents approximately eighty-eight point eight percent of the theoretical Gigabit Ethernet limit. These empirical results prove that the system eliminates computational bottlenecks, allowing client machines to write data near the maximum physical limit of local network infrastructure.

4.5 REMOTE ACCESS PERFORMANCE VIA TAILSCALE VPN

Remote file access performance was evaluated across external public networks utilizing the Tailscale Zero-Configuration Mesh VPN. The test environment linked a remote client laptop connected over cellular data networks to the edge storage node operating behind an institutional firewall. Initial network layer benchmarks conducted using iperf3 over the encrypted Tailscale tunnel recorded an average upload rate of 289 Mbit/s and an average download rate of 140 Mbit/s. Application-layer remote file transfers executed over SMB yielded the following transfer metrics:

- **Remote Read Performance (Edge NAS to Client):** Transferring a single twenty-Gigabyte file required nineteen minutes and one second, achieving an average throughput of 17.5 MB/s. Transferring multiple files achieved an average throughput of 17.6 MB/s.

- **Remote Write Performance (Client to Edge NAS):** Uploading a single twenty-Gigabyte file required sixteen minutes and thirty-nine seconds, achieving an average throughput of 20.0 MB/s. Uploading multiple files achieved an average throughput of 18.6 MB/s.

Table 3: Real-World Application-Layer SMB File Transfer Performance over Remote Tailscale Mesh VPN
(20 GB Total Workload)

Transfer Operation / Direction	File Structure & Workload	Total Data Volume	Elapsed Time (mm:ss)	Average Throughput (MB/s)	Equivalent Bandwidth (Mbit/s)
Write (Remote PC → Edge NAS)	Single Continuous File (1 File)	20 GB	16:39	20.0	160.0
	Multiple Fragmented Files (102 Files)	20 GB	17:55	18.6	148.8
Read (Edge NAS → Remote PC)	Single Continuous File (1 File)	20 GB	19:01	17.5	140.0
	Multiple Fragmented Files (102 Files)	20 GB	18:56	17.6	140.8

The observed throughput reduction from 111.1 MB/s on local LAN to 20.0 MB/s over remote VPN is primarily governed by wide area network bandwidth constraints, internet service provider throttling, cellular signal variation, and the cryptographic encapsulation overhead imposed by WireGuard packet encryption. Despite WAN bandwidth limits, a transfer rate of 20.0 MB/s remains highly practical for remote document management and media streaming.

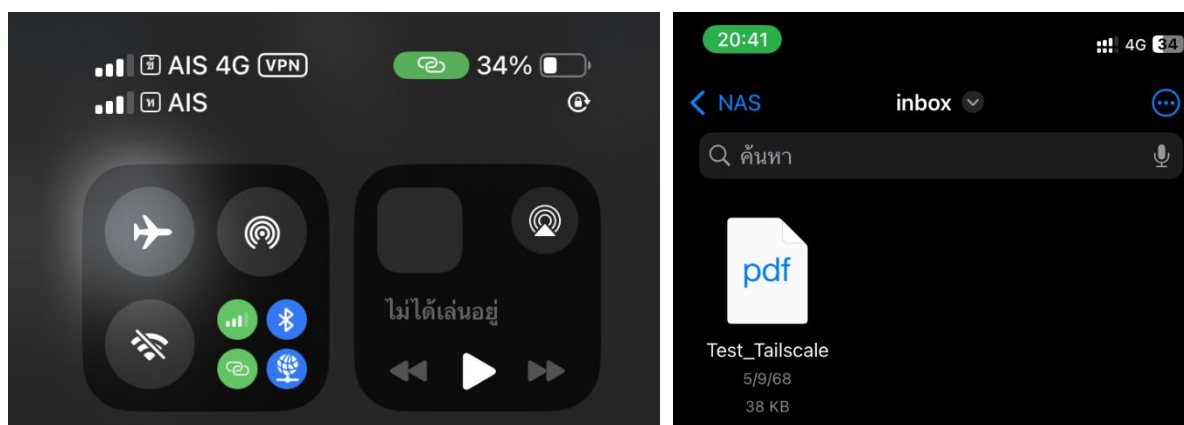


Figure 5: Remote Shared Directory Access via Tailscale Overlay Network on Mobile Client.

4.6 EVENT-DRIVEN AUTOMATION PERFORMANCE

The final testing phase evaluated the operational accuracy and response latency of the n8n workflow automation pipeline. Upon copying a test file into the monitored Inbox shared directory via Windows File Explorer, the n8n Local File Trigger node immediately detected the filesystem event.

The automation pipeline successfully executed its complete processing sequence in under two seconds:

1. Detecting file arrival via filesystem directory polling.
2. Executing system shell commands to capture exact file attributes and storage metrics.
3. Sanitizing and parsing raw text outputs into structured JSON data objects.
4. Mapping parsed parameters into a formatted email message payload.
5. Transmitting the automated alert email to the administrator inbox over SMTP protocols.

System monitoring verified one hundred percent notification delivery accuracy without dropped events or parsing failures during repeated file creation testing cycles.



Figure 6: Automated Email Notification Payload Dispatched by n8n Engine upon File Upload

V. DISCUSSION AND COMPARATIVE ANALYSIS

The empirical findings demonstrate that deploying an edge cloud storage server based on a low-power single-board computer achieves an exceptional balance between performance, capital expenditure, and operational energy efficiency. Operating under a maximum power profile of five volts and three amperes, the complete Raspberry Pi and solid-state drive assembly consumes approximately fifteen watts under peak processing load and substantially less during idle states. In comparison, traditional enterprise server hardware or commercial multi-bay Network-Attached Storage appliances continuously draw between thirty and one hundred watts of electrical power. Over multi-year deployment horizons, this dramatic reduction in energy consumption yields substantial financial savings in electricity overhead. From a financial capital perspective, the total hardware procurement expense requires a modest one-time investment. Commercial public cloud storage services require recurring monthly or annual subscription fees that accumulate significantly over extended timeframes. While public cloud providers offer tier-based pricing models, users who require several terabytes of personal storage face escalating long-term costs without acquiring physical hardware ownership. Furthermore, commercial off-the-shelf storage appliances entail higher initial hardware costs while offering redundant features that exceed the operational requirements of individual users or small research teams. The proposed edge cloud drive delivers a cost-effective alternative that achieves financial break-even within months of operation while preserving complete administrative data sovereignty.

Analyzing the empirical performance metrics reveals several critical throughput constraints across physical network and storage hardware layers. The primary physical constraint governing local file transfer throughput is the onboard Gigabit Ethernet interface. While raw storage block benchmarks confirm that the M.2 SATA solid-state drive achieves sequential read speeds up to three hundred and forty Megabytes per second, real-world application-layer SMB file transfers peak at one hundred and eleven point one Megabytes per second. This transfer rate saturates approximately eighty-eight point eight percent of the theoretical Gigabit Ethernet link capacity. Consequently, the local network interface acts as the primary throughput bottleneck, preventing the system from utilizing the full read potential of the underlying solid-state storage medium. Under continuous heavy write workloads, storage throughput is constrained by internal flash memory architecture. Experimental workload evaluations show that write speeds remain elevated between two hundred and forty-five and two hundred and sixty Megabytes per second for file sizes up to fifty Gigabytes due to high-speed pseudo-SLC cache buffering. However, when handling continuous write operations between sixty and one hundred Gigabytes, the cache buffer becomes fully saturated. This cache exhaustion forces write operations to transition directly to native NAND flash memory, causing sustained write throughput to decline to eighty-five to ninety Megabytes per second.

In addition to local interface limits, remote file management via Tailscale VPN experiences throughput reductions compared to local area network transfers. While raw local network transfers reach one hundred and eleven point one Megabytes per second, remote SMB transfers over encrypted VPN tunnels yield transfer speeds between seventeen point five and twenty point zero Megabytes per second. This performance reduction is influenced by external internet service provider bandwidth limitations, variable cellular network latency, packet encryption overhead from the WireGuard cryptographic protocol, and intermediate NAT traversal routing paths across public infrastructure. Despite these wide area network constraints, a remote transfer rate of twenty Megabytes per second remains highly practical for everyday remote document management and media streaming.

Deploying an edge storage node within institutional or enterprise environments introduces distinct operational challenges and architectural limitations. Network security policies enforced by corporate or academic

firewalls frequently block specific User Datagram Protocol ports or restrict encrypted Virtual Private Network traffic. Although Tailscale utilizes advanced NAT traversal mechanisms to establish peer-to-peer tunnels, strict firewall rules or symmetric NAT configurations can force network traffic through intermediate relay nodes, resulting in elevated latency and reduced transfer bandwidth. Furthermore, a primary technical limitation of the current physical architecture is the absence of hardware data redundancy. Because the system utilizes a single M.2 SATA solid-state drive in a non-RAID configuration, physical disk failure or file system corruption represents a critical single point of failure. Unlike commercial multi-disk storage appliances that implement RAID configurations or automated array rebuilding to protect data integrity, the single-drive architecture requires scheduled off-site backup routines to prevent catastrophic data loss.

VI. CONCLUDING SUMMARY

This study successfully demonstrates the design, deployment, and empirical validation of a low-cost, automated edge cloud drive system built on the Raspberry Pi 4 microcomputing platform. By integrating OpenMediaVault for storage administration, Tailscale for zero-configuration VPN access, and n8n for workflow automation, the proposed architecture achieves robust operational stability and high performance. Experimental evaluations confirm that local file transfers reach throughput levels near physical hardware limits, achieving SMB transfer speeds up to 111.1 MB/s over Gigabit LAN and maintaining secure remote connectivity over public networks. Additionally, the event-driven automation engine processes file metadata and dispatches notification alerts with negligible response latency. Ultimately, these findings prove that low-power single-board computing hardware offers an exceptionally practical, private, and cost-effective alternative to proprietary commercial cloud storage services for personal and small-scale edge environments.

Building upon these foundational results, future research should focus on several key hardware, network, and software enhancements. From a storage hardware perspective, upgrading to solid-state drives equipped with larger pseudo-SLC cache buffers or superior sustained-write NAND controllers will prevent throughput degradation during massive continuous file transfers. To overcome local network bandwidth limits, future system iterations can adopt higher-bandwidth network adapters, such as USB-to-2.5GbE interfaces, combined with CAT6 or CAT6A network cabling. In terms of data protection and operational resilience, implementing automated off-site backup pipelines and multi-disk RAID 1 or RAID 5 arrays will eliminate single points of failure and safeguard critical data against physical drive corruption. Finally, to ensure continuous remote accessibility across restrictive institutional firewalls that block Tailscale relay traffic, deploying secondary encrypted tunnels such as direct WireGuard or OpenVPN configurations will provide reliable cross-network connectivity.

REFERENCES

- [1] Raspberry Pi Foundation (2024) “Raspberry Pi 4 Model B Specification” Available online: <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>
- [2] OpenMediaVault (2024) “The Open Network Attached Storage Solution” Available online: <https://www.openmediavault.org/>
- [3] Tailscale Inc. (2024) “Tailscale · Zero Config VPN” Available online: <https://tailscale.com/>
- [4] n8n.io (2024) “n8n – Workflow Automation Tool” Available online: <https://n8n.io/>
- [5] National Science and Technology Development Agency (NSTDA) (2023) “VPN and Security Guidelines” Available online: <https://www.nstda.or.th/home/vpn-security/>
- [6] NOC RMUTP (2022) “Using iperf3 for Network Bandwidth Benchmarking” Available online: <https://noc.rmudp.ac.th/iperf/>
- [7] Axboe, J. “fio – Flexible I/O Tester rev. 3.38 Documentation” Available online: https://fio.readthedocs.io/en/latest/fio_doc.html
- [8] Geerling, J. (2021) “Raspberry Pi 2.5 Gbps 16 TB OMV NAS – Setup and Performance” Available online: <https://www.jeffgeerling.com/blog/2021/raspberry-pi-25-gbps-16-tb-omv-nas-setup-and-performance>
- [9] Red Hat Documentation (2024) “How to Install the Flexible I/O Tester (fio) Performance Benchmarking Tool” Available online: <https://access.redhat.com/solutions/3780861>
- [10] Nutanix (2025) “Performance Benchmarking with Fio on Nutanix” Available online: <https://portal.nutanix.com/kb/12075>