



Article

Evaluation of a New-Concept Secure File Server Solution

Gábor Arányi ¹, Ágnes Vathy-Fogarassy ^{2,*} and Veronika Szücs ¹

¹ Department of Electrical Engineering and Information Systems, University of Pannonia, Egyetem u. 10, 8200 Veszprém, Hungary; aranyi.gabor@mik.uni-pannon.hu (G.A.); szucs.veronika@mik.uni-pannon.hu (V.S.)

² Department of Computer Science and Systems Technology, University of Pannonia, Egyetem u. 10, 8200 Veszprém, Hungary

* Correspondence: vathy.agnes@mik.uni-pannon.hu

Abstract: Ransomware attacks are becoming increasingly sophisticated, and defensive solutions must necessarily evolve. Unfortunately, automated backup management through validation—critical for data recovery after an attack—and strengthening file server protection are not sufficiently addressed in current protection strategies. To focus on this area, an architectural model has been developed that complements traditional central data storage with an additional high-level file server protection solution. However, in addition to the functional adequacy of file server protection solutions, the measurability of operational performance is also important. In order to validate and evaluate the developed ransomware-proof file server protection, a dynamic performance metric is introduced for comparability, by which the performance results measured on the tested client–server architectures are presented, together with the methodology and results of the measurements. Our results show that the investigated model does not cause any performance degradation when moving sensitive data files and their backups during operation and even shows performance improvements in some frequently used configurations. The result proves that the developed real-time approach provides a solution to this critical problem in terms of the time required to restore key data from backups and to ensure file availability and continuity of accessibility. Based on a review of the literature and available solutions, it is concluded that there is no integrated solution for implementing a similar concept in practice, and therefore, the developed model is a gap-filling in this priority area.

Keywords: file server security; anti-ransomware system; safe data storage; ransomware attack recovery; dynamic reference-based performance rate



Citation: Arányi, G.; Vathy-Fogarassy, Á.; Szücs, V. Evaluation of a New-Concept Secure File Server Solution. *Future Internet* **2024**, *16*, 306. <https://doi.org/10.3390/fi16090306>

Academic Editor: Paolo Bellavista

Received: 25 June 2024

Revised: 23 August 2024

Accepted: 23 August 2024

Published: 26 August 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Due to the rapid advances in technology, not only smart devices, computers, and software have evolved at an enormous speed, but also malware (computer viruses, key-loggers, data manipulators, etc.) [1]. It is well-known that anti-malware applications have always been one step behind malicious software since the former was developed based on the mass data produced by computer viruses [2]. If a computer falls victim to a new(er) kind of malicious software, the anti-malware protection applications can detect the incident and take the necessary steps only with some delay. Obviously, the elapsed time interval of some days between detection and response can be critical in any system. The question is whether the malicious software was able to cause any damage to the system during this period.

Malicious software can have many purposes, such as crippling a system, stealing and manipulating data, or simply causing damage [3]. In order to reduce the damage caused by data loss, most of today's non-user systems back themselves up every hour, day, or week, or in any other specified time interval. That way, if malicious software gets into the system and goes live, the compromised system can be restored to the previous backup point. As described earlier, the time factor is crucial here, i.e., how quickly the antivirus software or other data protection application detects the problem within its system. Among the various

viruses, there is a growing number of data-stealing or data-access-blocking viruses that not only cripple the users' systems but can also cause serious damage [3]. A ransomware virus works by (1) gaining access to the target system, (2) optionally stealing data and encrypting the files, and (3) finally demanding a ransom from the victim. Although the details of the execution may vary from one ransomware variant to another, these three basic steps are the same for all of them. In relation to ransomware intrusions, Natalie Paskoski, Marketing Communications Manager at RH-ISAC [4], has also described that Unit42 incident response and data breach reports show that insecure Microsoft Remote Desktop Protocol (RDP) connections account for more than half of all ransomware attacks [4], followed by email phishing, which accounts for about a quarter of all ransomware infections, followed by software vulnerability exploits, which account for 12% [4].

Ransomware is still a real threat today, and according to the FBI's Cybercrime Report 2021, it caused more than \$42.9 million in losses in 2021 [5]. Beyond the headlines, this also means that in the event of an attack, a dedicated and well-established response protocol should be ready in every organization's security and defense toolkit. While a ransomware attack required years of development, cryptography, and penetration testing expertise and yielded only modest financial gains in the past, Ransomware-as-a-Service (RaaS) platforms are now becoming more common and widely available. Various illegal Internet forums allow would-be attackers to easily and cheaply associate with the authors of ransomware. Moreover, these RaaS programs are already well-developed and often come with user manuals and technical support.

However, despite rapidly evolving technology and high levels of digitization, many IT systems are outdated, which poses a high risk. For example, healthcare systems are a vulnerable area [6], for if a healthcare provider loses access to their data because of a successful ransomware attack, they have no other choice but to pay the ransom.

With ransomware spreading at an alarming rate and infrastructure being at risk all the time, it is essential to realize that while complex, resource-intensive hardware and software components are already available to provide the specified parts of protection, and however good the protection, prevention solutions and a rapid recovery from malicious events are goals that can really ensure business continuity in affected systems even in the event of a successful attack. One such task is to ensure file availability, meaning the capability of restoring files from backups. However, this requires a file system protection solution that provides adequate protection for the file servers. The solution presented in this paper aims to provide both file system protection and backup protection.

Our goal is to show that the proposed solution does not significantly degrade the user experience and transfer speed while providing protection. It offers adequate file operation speed and network transfer speed depending on the software configuration and can indeed be an integral part of any IT system where file protection and continuous availability are crucial factors.

The structure of the paper is as follows: Section 2 of the paper describes the evolution of ransomware, the changes in their working methods, some previous research results in the international literature focused on file server protection, and some of the advanced technologies in the field of protection solutions that can be applied today. Section 3 presents the authors' previously published complex file server protection solution, whose file server protection module is the subject of the current research. Furthermore, the measurement methodology for the file server protection performance test is also detailed in this section. Section 4 discusses the results of the tests. Section 5 discusses the results and presents the answers to the research questions. Finally, Section 6 summarizes the paper.

2. Background

2.1. Review of File System-Related Security Solutions and Research

The consequence of ransomware attacks, as described in the introduction, is mainly the loss of data availability. The evolution of protection techniques follows the evolution of ransomware, and the aim has always been to provide file systems with effective protection,

which prevents the restriction of file accessibility during an attack. In the early days (~2006), the primary solution was to restore the file system as a process. The file system restoration (FSR) recovery method enables users to go back to a previous state of the file system. In their study, Liang et al. [7] introduced a virtual disk environment (VDE) for undoing data changes on storages and restoring the undamaged files. VDE offers an effective restoring solution to a previous file system state, even in case of a system crash or a boot failure. VDE is similar to virtual disks used in virtualized infrastructures; however, they can be mounted without any hypervisor. Based on their implementation, the authors have analyzed the speed and the system load metrics of the VDE and concluded that, compared to the FSR-based solutions, the overhead is less. At the same time, recovery occurs quicker [7].

Dealing with early solutions, Gaonkar et al. [5] found that the financial burden connected to losing data and their inaccessibility might be huge, and for this reason, companies apply different data protection solutions, like snapshotting, systematic backups, and maintaining remote mirrors, to protect their data integrity. The authors also found that choosing the proper combination of such strategies can be complicated since there are many different methods for keeping data safe and for resource management. Furthermore, during the design of storage systems, it often happens that the mixture of the applied technologies leads to oversized or undersized, often costly, inappropriate solutions. In addition, the authors have presented a theoretical, automated approach to designing reliable storage systems [8].

Oorschot and colleagues [9] worked on a similar mechanism to counteract the harmful effects of ransomware attacks. In their research, they investigated the problem of malicious modification of digital objects. As a result, they presented a protection mechanism to protect digital objects against unauthorized replacement or modification while allowing for authorized updates transparently. In their presented solution, they used digital signatures without the requirement for a centralized public key infrastructure. To test the feasibility of their proposal, they implemented a Linux prototype, and their proposed solution protected the binary files of the file systems, the operating system, and applications on the disk. During testing, they showed that the prototype of the proposed protection solution provided protection against various kernel-modification-based rootkits (available and known at the time of development) while its implementation had minimal costs [9].

The IoT, cloud, fog, and edge computing architectures represent a number of risks: can data be kept confidential, and can integrity and availability be maintained? How can information loss, prolonged denial of data access, information leakage, or technical failures be avoided? In their study [10], Chervyakov et al. proposed a customizable, robust, redundant storage system which is able to verify the encrypted data and validate operations. Their system used the redundant residual number system (RRNS) and a new process for error correcting and key sharing. In their work, they introduced the notion of approximate ranking number value (AR), which contributed to the simplification of operations and decreased the size of the coefficients used during the RNS-to-binary conversion. Based on the approximate value and the arithmetic properties of RNA, the AR-RRNA method was introduced to detect and correct errors and verify computational results. They also provided a theoretical basis for estimating data loss probability and proper management of redundancy, coding rates and computing at various objective preferences, operational loads, and properties. They concluded that with the proper set of RRNS parameters, their suggested scheme both improved the security and reliability of storages and enabled them to reduce the processing time of encrypted data while requiring fewer resources.

Verma et al. [11] presented the design of a file system mechanism in their 2015 paper that protects the integrity of application data from errors such as process crashes, kernel panics, and power failures. In their solution, a simple interface guarantees for the applications that the application data in a file always reflect the last successful fsync or msync operation performed on the file. In addition, their proposed file system provides a new sync mechanism that captures changes in multiple files in case of failure. Error injection

tests have demonstrated that their file system protects the integrity of application data from crashes, and performance measurements have confirmed that the implementation is efficient. Their file system runs on traditional hardware and an unmodified Linux kernel and is commercially available. Verma et al.'s solution can be implemented in any file system that supports write(able) snapshots per file. They have also shown that a mechanism for Consistently Modifying Persistent Data of the Application (CMADD) can be easily implemented by file-by-file cloning, a feature already available in AdvFS and in many other modern FSs. Their implementation of O_ATOMIC provides a simple interface for applications, makes file modifications error-atomic by both write and map, avoids duplicate writes, and supports higher volume and frequent transactional updates of application data. Furthermore, their sync implementation supports error-atomic updates of application data in multiple files. Their empirical results showed that the O_ATOMIC implementation of AdvFS preserves the integrity of application data during updates in the case of both crashes following external interventions and accidental crashes (e.g., power outages). Their performance evaluation showed that the cost of introducing O_ATOMIC would be tolerable, especially for large updates. The implementation of the CMADD mechanism into the file system contributes to better acceptance as it requires neither special hardware nor a modified operating system kernel. The authors believed that file systems should implement simple, generic, robust CMADD mechanisms, as this would allow for many applications to take advantage of this feature, which would then be widely available. They further found that O_ATOMIC and syncv are convenient interfaces to use. In Table 1, the advanced data recovery solutions discussed above are compared based on their features.

Table 1. Summary of proposed major file system protection methods in the literature.

	Solution/Working Environ.	Capabilities/Services of Proposed Solutions					
		Full File System Recovery	Single File Recovery	Operation Validation	Error Correction/Detection	Overhead	Recovery Speed
Liang, J. et al. [7]	FSR VDE	yes yes	no yes	no no	no no	high low	low high
Gaonkar, S. et al. [8]	Automated approach	no	yes	no	no	medium	config. depend.
Van Oorschot, P.C. et al. [9]	Rootkit resistant mechanism	no	no	yes	yes	medium	no data
Chervyakov N. et al. [10]	RRNS—AR-RRNA	no	yes	yes (file op./enc.)	yes	medium	high
Verma, R. et al. [11]	AdvFS—CMADD solution	no	yes	yes	error detection	medium	tolerable

Newberry et al. [12] worked on bandwidth optimization of the Hadoop Distributed File System (HDFS). HDFS is a networked file system used to support several widely used big data processing frameworks, and it can scaled across large clusters. In their 2019 study, they evaluated the effectiveness of using in-network caching on HDFS-supported clusters to reduce in-network bandwidth consumption on switches. They discovered that some applications contained large amounts of data requested by multiple clients and that by caching data reads within the network, the average bandwidth usage of read operations per link in these applications could be more than halved. It was also found that the cache replacement policy chosen can have a significant impact on cache efficiency in this environment, with Low Inter-reference Recency Set (LIRS) and Adaptive Replacement Cache (ARC) generally performing best for the largest and smaller cache sizes. Furthermore, taking into account the structure of HDFS write operations, a mechanism was developed to

reduce the total bandwidth consumption of HDFS write operations per link by replacing write pipelining with write multicast. To evaluate the caching potential within the network, a simulator was developed to replay real pathways over a fat tree network, simulating the caching architecture used in the Named Data Networking (NDN) Information Centered Networking (ICN) architecture. Their results suggested that caching within an ICN-style network can provide significant benefits for HDFS-supported big data clusters, justifying future efforts to apply ICN architectures in cluster environments.

Makris et al. [13] addressed the issue of limited resources in their research on the management of large amounts of data and large data sizes. They found that edge computing is a promising paradigm for managing and processing the huge amount of data generated by IoT devices. Data and computation are brought closer to the client, enabling the effective use of latency and bandwidth-sensitive applications. However, the distributed and heterogeneous nature of the perimeter and its limited resource capabilities present a number of challenges when selecting or implementing a suitable perimeter storage system. A wide range of storage systems have been evaluated, so their results showed significant variance. Performance evaluation was conducted using resource utilization and Quality of Service (QoS) metrics. Each storage system was deployed on a Raspberry Pi, which served as a perimeter device and was able to optimize overall efficiency with minimal power consumption and at a minimal cost. Their experimental results showed that MinIO has the best overall performance in terms of query response times, RAM consumption, disk I/O time, and transaction rate [13].

It can be seen that the main thrust of the research involves improvements in both the performance of data movement over the network and the security of storage, both at the operating system kernel level and at the file structure level. Sizing file servers and optimizing their operational parameters in heterogeneous network environments is a major challenge. Often, new methods and new technologies are the way to go, but sometimes, the desired results can be achieved by reorganizing already well-known technologies. While meeting the requirements placed on file servers, like ensuring data availability and accessibility in the presence of large amounts of data, is a complex task, the effective protection of data requires special support and attention to ward off the increasing number of more and more sophisticated ransomware attacks.

2.2. State-of-the-Art Malware Prevention Techniques

There are many complex protection solutions on the market today, many of which are also antivirus solutions. They aim to protect both the server environment and the various endpoint clients from ransomware attacks. There are also several solutions that focus more on a proper backup strategy and try to minimize the damage caused in the event of a successful attack by using multiple, automated, often cloud-based backups. Currently, available high-end solutions include the following key features: signature-based ransomware detection and prevention, behavior-based threat prevention, depth knowledge-based AI capabilities, real-time memory protection, critical component filtering, real-time file filtering, plant decoy files to detect ransomware, automated server and workstation backup, IoT discovery and control, cloud security, and local and remote file encryption.

In addition to the above, some large enterprise solutions (e.g., IBM Security QRadar XDR [14], Bitdefender GravityZone Business Security Premium [15], Sophos Intercept X with XDR [16], Trend Micro Vision One [17]) monitor network traffic (C2, outbound traffic, DNS queries, etc.), include detailed logging and alerting services (SIEM), collect and automatically correlate data across multiple security layers (email, endpoint, server, cloud storage, network), and include features to enable data recovery.

3. Research Methodology

3.1. The ARDS System and Study Questions

The fundamental problem is that competitive protection solutions require advanced hardware tools, high-level professional support, and continuous financial investment,

which is often not a viable option for the masses of small or medium-sized companies and educational or healthcare institutes. However, due to the often outdated software environment, the overloaded administrator teams, and the sensitivity of critical services, these systems are increasingly targeted by ransomware attacks. Since the primary objective is to protect critical data, it is important to create an easily manageable, resource-efficient file server solution that runs smoothly on earlier-generation hardware devices as well. The proposed solution should not only store and back up the data assets but should also be able to protect data and detect and report an attack. The problems described above require complex, multi-layered solutions. As a result of the authors' earlier research, a comprehensive protection solution [18] was designed and developed, which is introduced briefly for easier understanding of the contents of the new results.

During the designing and development of the system, the main goal was to focus on the most important attack vectors (phishing [19], remote desk protocol (RDP) exploitation [20]) in order to be able to minimize the attack surface. Moreover, it was important to design the file server solution with the assumption that malware is already in the network, meaning that the central file server is the last possible point of protection. The solution developed is based on the previously published ARDS (Anti-Ransomware Defense System) model [18]. The model and its features can be seen in Figure 1.

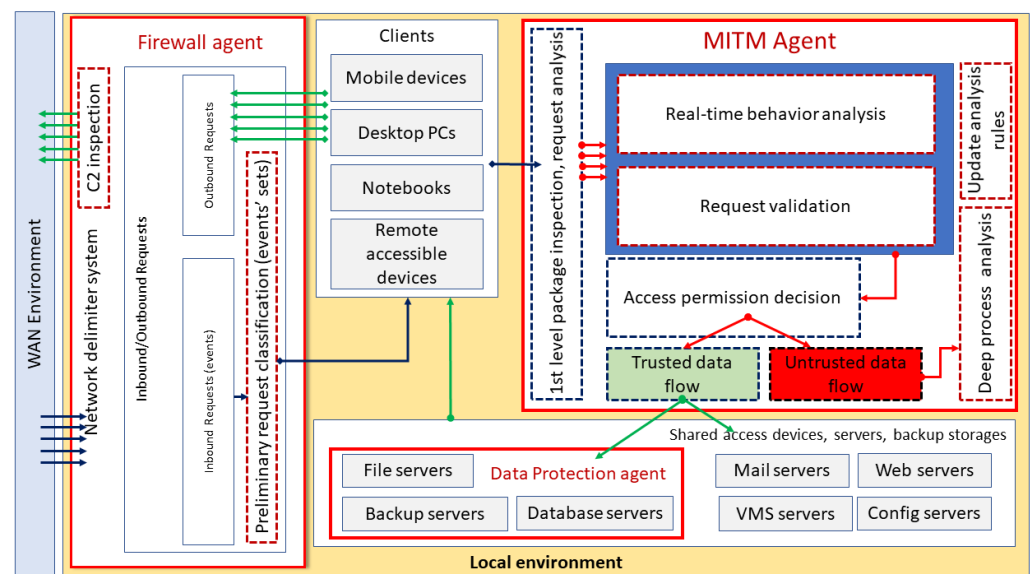


Figure 1. ARDS model concept and its features [18].

The ARDS model served as the test environment in our study. It has features that guarantee basic security, of which the local, point-to-point, IP-less backup system, scheduled snapshots on volume level, synchronized, encrypted backup to an off-site remote file server, and honeypot drive monitoring capabilities represent the basis for the protection. For more details, please see article [18].

The question (RQ-1: Research Question 1) that arises is whether the proposed file server protection solution, in terms of functionality, achieves the expected protection in a way that the measurable parameters (transfer speed, network capacity utilization) do not fall short of the conventional file server architectures built on traditional platforms, compared to currently accepted and practiced solutions. Therefore, the main aim of the present study was to present the comparative performance test and its results for the proposed server environment. To answer this question, performance measurements were performed on different client–server architectures with a clean installation and default service environment settings.

Additionally, the following research questions were raised during the study: RQ-2: Can the performance of systems with different service environments be compared using

default settings? RQ-3: If valid comparability is possible, is there currently a generally accepted method and metric that is not an individual performance indicator of each system but can quantitatively characterize heterogeneous systems with a well-defined metric as an aggregated performance indicator of functionally constructed architectures?

3.2. Methodology of the Performance Tests

For an easier understanding of the goal of the examination, we should highlight that the main goal was to compare an often used, typical, clean-installation Windows server solution with a customized (i.e., software-based RAID, scheduled snapshotting on file system level, encryption) FreeBSD instance. In the present study, we ran the tests with the default settings available on clean installs of the operating systems; no uniform settings were customized, and no additional third-party anti-virus software was installed. The possible additional impact of an installed anti-virus solution was not a part of the present study. The transfer time performance of the clean installed systems was investigated in the client–server context, and in addition to the most often used Windows OS traditional server solutions, we also investigated a custom-built FreeBSD-based test server solution.

As a starting point, services and software configurations required for building the client–server architectures under investigation were defined. Based on the information gathered about business processes related to general daily routine tasks, typical file sizes were also defined for the measurements. Finally, metrics to be used for the performance measurements were also defined.

3.2.1. Examined Structures

Preliminary surveys suggested that in typical client–server architectures used in small and medium-sized enterprise (SME) business environments, the file servers are mostly installed with Windows Server operating systems or Unix-like operating systems. Looking at the composition of the clients' OS types, it was stated in our previous survey [18] that the MS Windows 7 x64 SP1 (6.1.7601), MS Windows 10 x64 Pro 21H2, and Ubuntu 22.04.4 LTS operating systems are present in the networks in a huge volume.

Based on the statements mentioned above, the following measurement structures (TE—Test Environments) were defined for the performance testing, where the ARDS TestServer is a custom-built FreeBSD 13.0 server environment:

- {TE-01}: Windows 7 client—ARDS TestServer
- {TE-02}: Windows 7 client—Windows 2019 Server
- {TE-03}: Windows 10 client—ARDS TestServer
- {TE-04}: Windows 10 client—Windows 2019 Server
- {TE-05}: Ubuntu 22.04 client—ARDS TestServer
- {TE-06}: Ubuntu 22.04 client—Windows 2019 Server

In order to obtain the most accurate results, it was important to keep the hardware configurations unchanged on both the server and client sides throughout the tests. The direction of data flow was determined from the client side, and this was where the transfer time was measured for both the read and write tests. The testing of Windows 7 clients was motivated by the fact that, unfortunately, such outdated operating systems are also present in large numbers in the computer pools of SMEs and other service providers.

The measurements were carried out on the same file server configuration, which typically performs adequately in SME business environments: Intel Xeon 4Core E-2224G 3.5 GHz processor, 8GB DDR4 ECC RAM, Apacer AS350X 128 GB SSD (system drive), Gigabyte GP-GSTFS31240GNTD 240 GB SSD (data drive), and Intel X520-DA2 SFP+ 10GBE PCI-E dual-port network interface card.

While designing the physical client–server network, it was taken into account that in SME business environments, data are typically transmitted over Ethernet and SFP interfaces. Consequently, the measurements were run using both 1 Gbit/s and 10 Gbit/s connections. Therefore, it was important that the network card was natively supported by the BIOS and OS, so in this case, the traditional workstation configuration was not used. For the client,

the hardware configuration set up was as follows: Intel Core i7-3520M 2.9 GHz processor, 16 GB DDR3 non-ECC RAM, Kingston SSD now V300 120 GB (system+data), Intel 82579LM Gigabit NIC, Intel X520-DA2 SFP+ 10GBE PCI-E Dual port NIC.

The test environment was not designed to scale the network but to investigate the temporal performance of file transfers between different operating systems. The one server–one client architecture is justified by the fact that the proposed file server protection solution focuses primarily on protecting the file server. Having redundant file servers in a network was not considered viable, even approaching the number of workstations communicating with the backup server. For the hardware configuration, we also aimed for a minimal operating environment, which also means that with higher-capacity hardware, performance is certainly no worse than currently under investigation.

Tests were performed on entry-level business-class hardware because these are typical in the SMA sector. From the aspect of service performance in the comparison phase, hardware platforms do not play significant roles. The performance rates would be approximately the same if we would use high-end blade cluster architectures.

3.2.2. Defining the Data Set to Be Used for Measurement

In order to make valid comparisons between the different server environments, we needed to ensure not only identical hardware and network infrastructure but also identical sample data. The data transfer tests were carried out on both 1 Gbit/s and 10 Gbit/s capacity direct copper-based cable connections, measuring both transmission directions (read/write).

When designing the measurement plan, it was important to take into account that users typically work with different file sizes in different productive environments. In order to ensure that the compression procedures used by the file transfer protocols, various CIFS/SMB (Common Internet File System—CIFS, Server Message Block- SMB) versions and configurations in the test process, did not notably affect the results, the files used for testing were generated by using random bit patterns. For both the read and write tests, 1 GiB of data were transferred in the form of the required number of 500 KiB, 4 MiB, 256 MiB, and 1 GiB files, as follows:

- 1×1 GiB
- 4×256 MiB
- 256×4 MiB
- 2097×500 kiB

File operations were deliberately carried out by imitating normal user practice, using native command-line solutions (copy, del, cp, rm), as tools such as robocopy or rsync could have greatly affected the results of the tests—not to mention that the use of such tools would not reflect real-life scenarios. Each read and write test was followed by a short deletion process when the target directory was emptied. This was, of course, not an actual and safe wipe but a logical wipe and was not counted as part of the transfer time. The maximal throughput of the network connections was tested using Iperf3 (network bandwidth test application) prior to the transfer tests.

3.2.3. Reducing the Uncertainty of the Measurements and Filtering Out Measurement Errors

Reducing measurement uncertainty was essential during the tests, as measurements can be affected by both random and systematic errors. Random errors are difficult to identify and may, for example, result from an anomaly in the functioning of the electronic equipment used for the measurement or the devices providing the physical link between them. Systematic errors may result from shortcomings in the measurement method or from the normal operation of the system under test.

Our aim was to detect these errors and eliminate their distorting effects on the measurement results. To measure the performance of the server environments under test, we needed to reduce measurement uncertainty as much as possible by reducing the number of

external factors that cause uncertainty. To reach this goal, we used a direct cable connection (crossover CAT6 UTP, SFP + DAC) instead of the traditional switched and possibly routed network infrastructure, and we also disabled the possible native real-time virus protection on both sides during the measurements.

Despite all our efforts, some anomalies might still have occurred in the study, which could be attributed to the following factors, among others:

- Read/write buffer management (filling up, emptying);
- Initial indexing of files;
- The network throughput sometimes exceeded the theoretical maximal bandwidth of the SATA3 interface (10 Gbit/s vs. 6 Gbit/s), which could lead to uncertainty due to different caching mechanisms;
- Minor fluctuations in read/write speed of SSD drives;
- The performance impact of other running processes.

In order to reduce measurement uncertainty and the impact of measurement errors, all read and write tests were conducted ten times consecutively in each test environment. The average transfer speeds were calculated based on the time needed for the transmission. We determined the lowest and maximum transfer time values, the average transfer time, and the standard deviation values for both transfer times and speeds. The network load was continuously checked to prevent transmission errors, and the hash calculation check of the data was conducted after each transmission.

3.2.4. Ensuring Reproducibility

The replicability of the tests also helps to eliminate measurement uncertainty. The measurements were carried out according to the measurement plan, and the transfer times—which, of course, did not include the time needed to delete temporarily stored files—were logged separately for each test. The specification of both the hardware and software environments was thoroughly documented. The files used for the measurements were generated by our script, which can be found in Appendix A.1, and the scripts responsible for file transfer on clients are in Appendices A.2 and A.3.

3.2.5. Methodology for Evaluating the Results

While processing measurement results, the measured values were analyzed using statistical methods. In repeated measurements, transmission rate values showing a transient state were both detected and excluded; from the remaining analyzable data, the measurements were then evaluated by treating them as normally distributed probability variables, and in this way, the characteristic values for each architecture were determined. Based on the differences between the measurements, characteristic deviation values for the statistical distribution were determined, allowing for the results to be compared.

In the case of different hardware and software configurations, the average transfer times are unique characteristics; they are not suitable for comparing the effectiveness of the systems. However, it is important to see the performance values compared to an arbitrary, selected reference system. To solve this problem, the following metric was calculated:

Let T_a be the transfer time of the current (examined) environment, and T_r the transfer time of the reference environment. The reference-based performance indicator (PR_{ar}) can be calculated as follows:

$$PR_{ar} = \frac{T_a}{T_r} \quad (1)$$

If $PR_{ar} > 1$, then the current (tested) environment's performance is better than the performance of the reference environment; if $PR_{ar} = 1$, then the performance of the T_r and T_a environments is almost on the same level, and if $PR_{ar} < 1$, then the performance of the reference environment is better than the currently tested environment.

4. Results

Initially, the transfer time of different client–server platforms (defined in Section 3.2.1) was tested using a 1 Gbit/s point-to-point network connection. This was important partly because it can be regarded as typical for today’s non-profit organizations and SMEs, and partly because it provided transfer time characteristics for each system, thereby serving as a basis of comparison for evaluating the results of subsequent tests.

This was followed by a 10 Gbit/s point-to-point network connection data transmission, which provided a good point of reference for “enterprise class” network infrastructures. The Microsoft Windows 7 × 64 operating system was not tested here, as it does not support the Intel X520-DA2 network card, and it would have been unrealistic to use an EOL (End of Life) operating system while providing a modern network infrastructure for the measurements.

It was also important to increase the throughput of the network following the tests at 1 Gbit/s connection speeds because tests at 1 Gbit/s were, in most cases, close to the upper limit of the theoretical maximum throughput (125 MiB/s), making performance differences less detectable. At that bandwidth, it was not possible to model workstations and servers being placed under significant workload.

When evaluating the measurement results, we found that the results of the first measurements differed from the results of the subsequent nine measurements in terms of transfer time, especially for smaller files (500 KiB, 4 MiB), typically in the case of Windows operating systems. However, since the standard deviations calculated from the results of the subsequent nine measurements did not show significant differences anywhere, the results could be considered representative based on the average of the measurements. In the case of the 1 Gbit/s network connection, the average of the transfer speeds of the ten measurements was 846 Mbit/s (100.85 MiB/s), while for the 10 Gbit/s connection, the average transfer speed was 9.38 Gbit/s (1118.18 MiB/s).

During the measurements, neither client nor server CPU usage exceeded 47%, and system memory utilization did not exceed 52%. However, the SSD drives repeatedly showed 100% utilization in the faster, 10 Gbit/s bandwidth tests.

The detailed results of the tests are presented in the following subsections.

4.1. Results on 1 Gbit/s Connection

In the following sections, the reading (Section 4.1.1) and writing (Section 4.1.2) test results are presented for the 1 Gbit/s connection. The raw measurement results are available in the Supplementary Material for more analyses.

4.1.1. Results of Reading Performance Tests on 1 Gbit/s Connection

The detailed results of the reading tests conducted on the network with 1 Gbit/s throughput are illustrated in Table 2 and Figure 2. The used notation convention of files involved in the operation is as follows: *the number of files* × *file size*.

For files of 256 MiB and 1 GiB, all six client–server platform pairs performed very similarly (min. 101.01 MiB/s, max. 111.21 MiB/s), as it can be seen in Table 2, but for files of 4 MiB, a significant performance drop was observed for Windows clients. However, the results also showed that the transfer speeds for both server platforms were nearly the same for Windows 7 and Windows 10. It was also striking that, regardless of the file server, the Ubuntu 22.04 client was able to achieve 30–40 percent faster transfer speeds. When reading files up to 500 kiB, the speed dropped by almost a fifth for Windows 7 and Windows 10, regardless of the server-side platform, while the Ubuntu client was roughly three times faster in both scenarios.

Table 2. Reading test results on 1 Gbit/s bandwidth.

Test ID	Files			
	1 × 1 GiB	4 × 256 MiB	256 × 4 MiB	2097 × 500 kiB
Average Transfer Speed in MiB/s				
Windows 7—ARDS TestServer	101.01	110.01	50.47	19.31
Windows 7—Windows 2019 Server	107.99	108.58	48.95	17.90
Windows 10—ARDS TestServer	108.43	111.21	59.81	20.87
Windows 10—Windows 2019 Server	107.90	110.74	59.08	19.48
Ubuntu 22.04—ARDS TestServer	109.82	109.10	90.95	59.51
Ubuntu 22.04—Windows 2019 Server	110.38	109.90	88.68	69.71

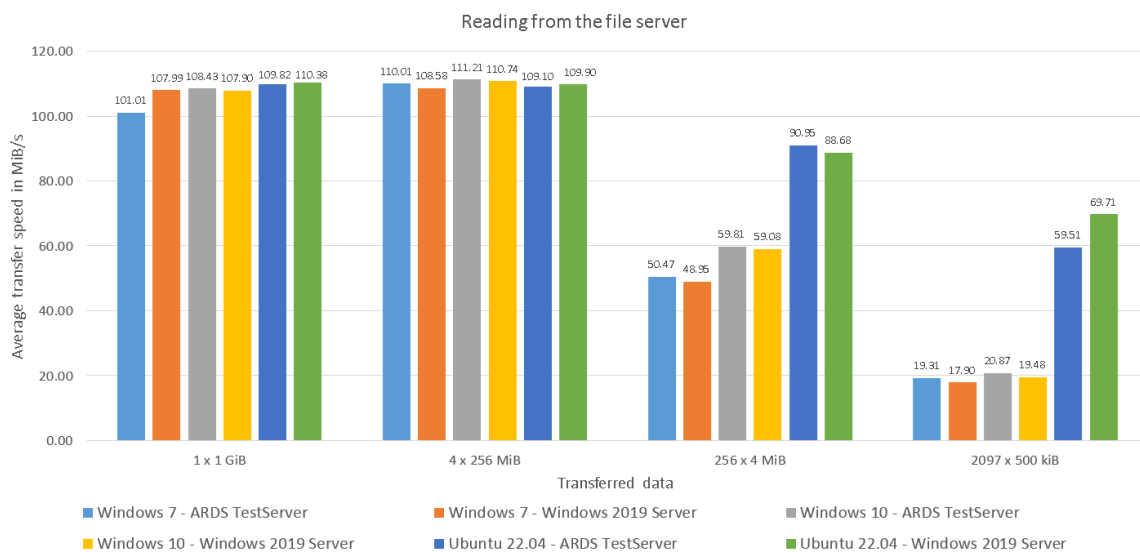


Figure 2. Measurement results of reading on 1 Gbit/s bandwidth.

4.1.2. Results of Writing Performance Tests on 1 Gbit/s Connection

In the case of file writing, the average transfer speeds calculated for the 1 Gbit/s bandwidth network are illustrated in Table 3 and Figure 3.

Table 3. Writing test results for 1 Gbit/s bandwidth.

Test ID	Files			
	1 × 1 GiB	4 × 256 MiB	256 × 4 MiB	2097 × 500 kiB
Average Transfer Speed in MiB/s				
Windows 7—ARDS TestServer	110.01	107.84	42.73	11.67
Windows 7—Windows 2019 Server	111.14	109.39	46.46	9.91
Windows 10—ARDS TestServer	27.26	8.35	0.11	0.03
Windows 10—Windows 2019 Server	111.14	110.44	73.46	28.15
Ubuntu 22.04—ARDS TestServer	107.45	104.87	92.29	45.15
Ubuntu 22.04—Windows 2019 Server	108.38	105.73	92.05	61.80

The results show that for all file sizes, the Windows 10—ARDS TestServer client–server configuration showed a spectacular drop in performance, with speeds for smaller files (4 MiB, 500 KiB) approaching zero. It was also observed that during the data transfer of smaller files, the transfer speed was lower for all other client–server pairings, but the two servers provided almost identical performances. The drastic drop in transfer time for Windows 10 in a non-Microsoft Windows 2019 Server environment is a widely known and well-documented phenomenon [21–24], which can be solved by fine-tuning server-side

configurations, client-side system parameter settings, and registry changes, which can be implemented centrally using group policy objects in a domain environment.

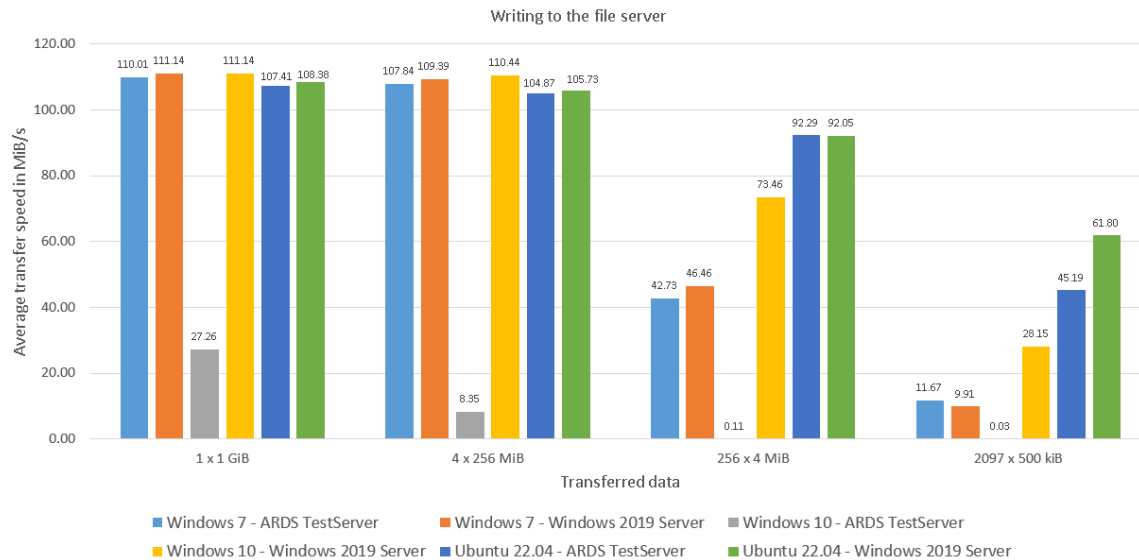


Figure 3. Measurement results of writing on 1 Gbit/s bandwidth.

4.2. Results for 10 Gbit/s Connection

The following sections present the reading (Section 4.2.1) and writing (Section 4.2.2) test results for the 10 Gbit/s connection. The raw measurement results are available in the Supplementary Material for more analyses.

4.2.1. Results of Reading Tests on 10 Gbit/s Connection

The detailed results of the reading tests conducted on the network with a 10 Gbit/s throughput are illustrated in Table 4 and Figure 4.

Table 4. Reading test results for 10 Gbit/s bandwidth.

Test ID	Files			
	1 × 1 GiB	4 × 256 MiB	256 × 4 MiB	2097 × 500 kiB
Average Transfer Speed in MiB/s				
Windows 10—ARDS TestServer	961.41	796.50	144.62	39.97
Windows 10—Windows 2019 Server	759.98	689.00	184.00	42.87
Ubuntu 22.04—ARDS TestServer	1020.51	998.14	620.99	208.60
Ubuntu 22.04—Windows 2019 Server	630.24	634.34	304.59	154.75

In the reading tests, ARDS TestServer was typically faster. However, when reading files of 4 MiB and 500 kiB using Windows 10 clients, the Windows 2019 server performed better; it outperformed the open-source file server by 27.3 percent in the case of the former size and by 7.2 percent for the 500 kiB file sizes. It is important to note that the fully open-source client-server combination was faster than any other operating system version in all cases. At this network speed, the client is subject to a significant I/O workload, which, especially for the two smaller file sizes, meant a spectacular performance degradation due to the intensive writing activity to the disk [25]. Also, it could be clearly observed that the file transfer time exceeded the theoretical throughput of the SATA III interface (750 MiB/s) in several cases due to the different caching mechanisms (SSD, CPU data cache, internal network cache, NIC data cache, etc.) operating in various manners and at different rates.

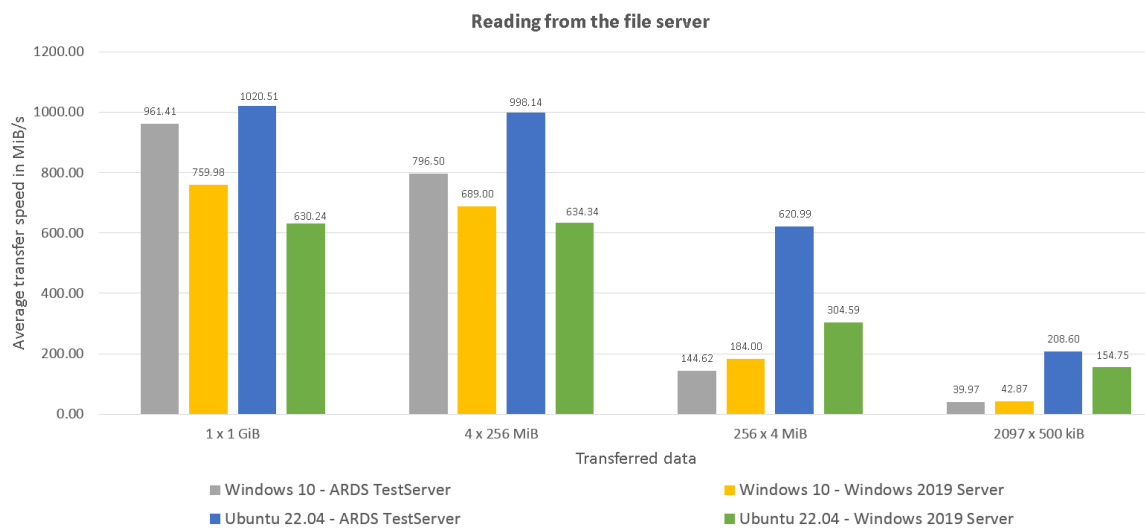


Figure 4. Measurement results of reading on 10 Gbit/s bandwidth.

4.2.2. Results of Writing Tests on 10 Gbit/s Connection

The average transfer speeds calculated for file writing for the 1 Gbit/s bandwidth network are illustrated in Table 5 and Figure 5.

Table 5. Writing test results for 10 Gbit/s bandwidth.

Test ID	Files			
	1 × 1 GiB	4 × 256 MiB	256 × 4 MiB	2097 × 500 kiB
Average Transfer Speed in MiB/s				
Windows 10—ARDS TestServer	268.08	252.94	180.35	75.64
Windows 10—Windows 2019 Server	810.08	816.61	234.18	59.35
Ubuntu 22.04—ARDS TestServer	303.69	328.22	233.43	150.84
Ubuntu 22.04—Windows 2019 Server	843.81	844.27	558.02	322.65

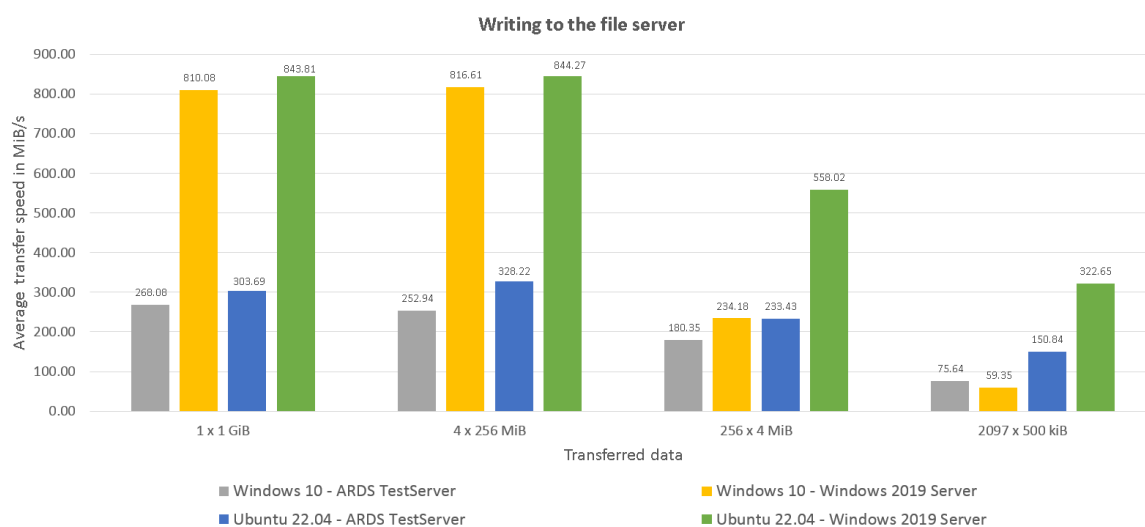


Figure 5. Measurement results of writing on 10 Gbit/s bandwidth.

The Ubuntu–Windows 2019 Server configuration performed best in the measurements for all four file sizes. It is clear from the results that ARDS TestServer was, on average, three times slower than the Windows server for files of 1 GiB and 256 MiB at this transfer rate because of the use of snapshotting techniques. In the case of using the 4 MiB files, the speed difference was smaller, and what is more, when transferring the smallest files

using Windows 10, the transfer was already more than 20 percent faster when we tested the open-source file server. Furthermore, it is important to realize that the Windows server did not have shadow copy service (VSS) enabled, which would have meant a similar extra load when writing to the server as the automated snapshotting in ARDS TestServer.

4.3. Reference-Based Performance Evaluation

In order to compare the performance of the two server-side software environments in an exact manner, the reference-based performance indicator presented in Section 3.2.5 was also calculated. The metric was calculated by dividing the average of the transfer times for the same clients and the same file sizes with each other for different server environments. Since we are investigating the performance of the ARDS TestServer, the average transfer times of the ARDS Server were divided by the Windows 2019 Server results. It follows that if the quotient is less than one, the proposed solution (ARDS TestServer) performs better, and if it is greater than one, it indicates poorer performance. The calculated reference-based performance indicators are summarized in Tables 6 and 7.

Table 6. Reference-based performance evaluation for reading and writing tests on 1 Gbit/s bandwidth.

	Files			
	1 × 1 GiB	4 × 256 MiB	256 × 4 MiB	2097 × 500 kiB
Reading test results				
Windows 7—ARDS TestServer/ Windows 7—Windows 2019 Server	1.0723 (=)	0.9872 (+)	0.9703 (+)	0.9078 (+)
Windows 10—ARDS TestServer/ Windows 10—Windows 2019 Server	0.9958 (+)	0.9957 (+)	0.9873 (+)	0.9363 (+)
Ubuntu 22.04—ARDS TestServer/ Ubuntu 22.04—Windows 2019 Server	1.0043 (=)	1.0075 (−)	0.9749 (+)	1.1715 (−)
Writing test results				
Windows 7—ARDS TestServer/ Windows 7—Windows 2019 Server	1.0109 (=)	1.0150 (=)	1.0947 (=)	0.8491 (+)
Windows 10—ARDS TestServer/ Windows 10—Windows 2019 Server	4.0782 (−)	13.2265 (−)	683.8709 (−)	878.7343 (−)
Ubuntu 22.04—ARDS TestServer/ Ubuntu 22.04—Windows 2019 Server	1.0085 (=)	1.0072 (=)	0.9973 (+)	1.3681 (−)

(+) ARDS TestServer performed better. (=) Windows 2019 Server performed better, but the difference in speed was under 10 percent. (−) Windows 2019 Server performed better and the difference in speed was higher than 10 percent.

The tests provided a comprehensive picture of the performance of the proposed server environment compared to a commonly used solution. Examining the different transfer time ratios obtained from the read tests, it is clear that in the vast majority of cases (14), the proposed solution performed better; in 3 cases, there were minor differences; and in only 3 cases was it slower during the transmission. In the writing tests, the transfer time in most cases (12) was low compared to the Windows 2019 Server solution. In five cases, it showed similar results, and in three cases, it was higher.

Table 7. Reference-based performance evaluation for reading and writing tests on 10 Gbit/s bandwidth.

	Files			
	1 × 1 GiB	4 × 256 MiB	256 × 4 MiB	2097 × 500 kiB
Reading test results				
Windows 10—ARDS TestServer/ Windows 10—Windows 2019 Server	0.7809 (+)	0.8672 (+)	1.2733 (−)	1.0728 (=)
Ubuntu 22.04—ARDS TestServer/ Ubuntu 22.04—Windows 2019 Server	0.6134 (+)	0.6199 (+)	0.4947 (+)	0.7620(+)
Writing test results				
Windows 10—ARDS TestServer/ Windows 10—Windows 2019 Server	3.0263 (−)	3.2113 (−)	1.2903 (−)	0.7893 (+)
Ubuntu 22.04—ARDS TestServer/ Ubuntu 22.04—Windows 2019 Server	2.8602 (−)	2.7480 (−)	2.4110 (−)	2.1825 (−)

(+) ARDS TestServer performed better. (=) Windows 2019 Server performed better, but the difference in speed was under 10 percent. (−) Windows 2019 Server performed better, and the difference in speed was higher than 10 percent.

5. Discussion

Considering the reading test results of the reference-based performance evaluation, the proposed solution outperformed the performance of the Windows 2019 Server in most cases and resulted in only a few cases of similar or slightly slower performance. The methods for increasing reading speed are well-documented in the relevant literature. In particular, server-side service parameters need to be fine-tuned or defined depending on the specific client operating systems.

In the writing tests, the proposed solution performed mostly worse than the Windows server. This is mainly due to two important factors that caused considerable workload in the FreeBSD-based environment. One is the file system-level encryption to prevent physical data theft, and the other is volume-based automatic snapshotting, which only imposes a significant challenge when writing data. Integrating these two services, of course, was a handicap compared to the Microsoft system, but since most live Windows servers do not have file encryption (BitLocker) enabled by default and shadow copy (VSS) is only enabled on the system partition, we wanted the analysis to be based on a typical user practice.

To widely deploy the proposed solution, it is important to ensure that services such as scheduled snapshotting, honeypot network drive monitoring, IDS/IPS, virus protection (real-time or scheduled virus scanning depending on hardware capacity), the automated backup system, or AI-based log monitoring should not result in significant performance degradation compared to the commonly used solutions.

The results show that while only minimal parameter correction is needed for performing read operations efficiently, significant speed-ups are essential for write operations. For this purpose, the following “best practice” solutions can help: client-specific configuration fine-tuning on the server side, increasing the buffer size, adjusting caching parameters, using an extra SSD for caching, and client-side optimization (registry, GPO, and SMB parameters).

Our answer to the first research question (RQ-1) is a definitive yes. At the current configuration level, the evaluation of the measurement results confirms that the proposed system does not lag behind the file servers operating on traditional platforms in terms of measurable performance parameters.

The conducted study was able to partially answer the second research question (RQ-2): whether the performance of systems set up with different default service environments can be compared. In order to detect truly significant differences in performance, further studies are required in environments built with coordinated services.

In connection with the third research question (RQ-3), in which we sought a reliable methodology and metric of evaluation in the course of this research, we concluded that it is possible to quantitatively characterize and compare the performance of heterogeneous systems with the reference-based performance indicator.

6. Summary

In this article, the performance of a new-concept secure file server solution designed to protect against ransomware without causing significant performance degradation was evaluated. The article introduced the ARDS (Anti-Ransomware Defense System) model, which integrates various security features such as IP-less local backup, scheduled snapshots, and encrypted remote backups. During the performance tests, this solution was compared with a traditional file server architecture, revealing that the ARDS system generally performed better, particularly in reading speed, though there was a performance drop in writing operations due to encryption and snapshotting. In more detail, the results showed that ARDS TestServer performed better or equal in 66.7% of the read tests compared to the Windows 2019 Server environment at the maximum throughput limit of 1 Gbit/s, while in the write tests, the figure was 58.3%. At the 10 Gbit/s maximum throughput limit, the same tests showed that the ARDS TestServer read performance was better than the Windows 2019 environment in 83.3% of the tests, but in the write tests, Windows 2019 Server performed better in 87.5% of the tests. The research findings confirm that the ARDS solution offers effective protection without significantly impacting user experience or data transfer speeds by reading. The transfer speed drops that occurred during some writing test cases can be effectively eliminated according to best practice solutions and official documentation.

Supplementary Materials: The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/fi16090306/s1>.

Author Contributions: Conceptualization, G.A. and V.S.; methodology, G.A., V.S. and Á.V.-F.; software, G.A.; validation, G.A.; writing—original draft preparation, G.A., V.S. and Á.V.-F.; writing—review and editing, G.A., V.S. and Á.V.-F.; visualization, G.A. and V.S.; supervision, Á.V.-F. and V.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the TKP2021-NVA-10 project with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the 2021 Thematic Excellence Programme funding scheme.

Data Availability Statement: Data is contained within the article or Supplementary Materials.

Acknowledgments: We are grateful to our colleague, Ferenc Markó, who helped us by providing the hardware infrastructure and shared many useful pieces of advice to help us achieve our goals. In addition, we would like to give special thanks to Ákos Dávid for the technical support.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

AI	Artificial Intelligence
ARDS	Anti-Ransomware Defense System
CPU	Central Processing Unit
DAC	Direct Attached Cable
ECC-RAM	Error Correction Code Random Access Memory
GPO	Group Policy Object
HDFS	Hadoop Distributed File System
ICN	Information Centered Networking
LIRS	Low Inter-reference Recency Set
NDN	Named Data Networking
NIC	Network Interface Card
OS	Operating System
RaaS	Ransomware-as-a-Service

SATA	Serial Advanced Technology Attachment
SFP	Small Form-factor Pluggable network interface
SMB	Server Message Block
SME	Small and Medium-sized Enterprises
SSD	Solid State Drive
VSS	Volume Shadow Copy Service

Appendix A. Scripts

Appendix A.1. Script to Generate Sample Files for Measurements

```
for i in {1..2};
do head -c 256M < /dev/urandom > demo256M_${i}.bin;
done
```

Appendix A.2. Script to Manage File Transfer on Windows Clients

```
@echo off
setlocal enabledelayedexpansion
for /l
del /s /q c:\temp\*. * > NUL
@echo !time: =!
copy r:\temp\256M c:\temp\ > NUL
@echo !time: =!
)
```

Appendix A.3. Script to Manage File Transfer on Ubuntu Clients

```
for ((n=0;n<10;n++)); do rm -f /temp/256M/*;
time cp -f /home/user/Desktop/Bench_demo/256M/* /temp/256M;
done
```

References

1. Ferdous, J.; Islam, R.; Mahboubi, A.; Islam, M.Z. A Review of State-of-the-Art Malware Attack Trends and Defense Mechanisms. *IEEE Access* **2023**, *11*, 121118–121141. [\[CrossRef\]](#)
2. Ngo, F.T.; Agarwal, A.; Govindu, R.; MacDonald, C. Malicious Software Threats. In *The Palgrave Handbook of International Cybercrime and Cyberdeviance*; Holt, T.J., Bossler, A.M., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 793–813. [\[CrossRef\]](#)
3. Vanness, R.; Chowdhury, M.M.; Rifat, N. Malware: A Software for Cybercrime. In Proceedings of the 2022 IEEE International Conference on Electro Information Technology (eIT), Mankato, MN, USA, 19–21 May 2022; pp. 513–518. [\[CrossRef\]](#)
4. Microsoft. Troubleshooting Slow File Copying in Windows. 2022. Available online: <https://learn.microsoft.com/en-us/troubleshoot/windows-client/performance/troubleshooting-slow-file-copying-in-windows> (accessed on 28 February 2023).
5. FBI. Ransomware. 2023. Available online: <https://www.fbi.gov/how-we-can-help-you/safety-resources/scams-and-safety/common-scams-and-crimes/ransomware> (accessed on 28 February 2023).
6. Gabrielle, H.; Mikiann, M. Healthcare Ransomware Attacks: Understanding the Problem and How to Protect Your Organization. *LogRhythm*. **2024**.
7. Liang, J.; Guan, X. A virtual disk environment for providing file system recovery. *Comput. Secur.* **2006**, *25*, 589–599. [\[CrossRef\]](#)
8. Gaonkar, S.; Keeton, K.; Merchant, A.; Sanders, W. Designing dependable storage solutions for shared application environments. In Proceedings of the International Conference on Dependable Systems and Networks (DSN'06), Philadelphia, PA, USA, 25–28 June 2006; pp. 371–382. [\[CrossRef\]](#)
9. Van Oorschot, P.C.; Wurster, G. Reducing Unauthorized Modification of Digital Objects. *IEEE Trans. Softw. Eng.* **2012**, *38*, 191–204. [\[CrossRef\]](#)
10. Chervyakov, N.; Babenko, M.; Tchernykh, A.; Kuchеров, N.; Miranda-López, V.; Cortés-Mendoza, J.M. AR-RRNS: Configurable reliable distributed data storage systems for Internet of Things to ensure security. *Future Gener. Comput. Syst.* **2019**, *92*, 1080–1092. [\[CrossRef\]](#)
11. Verma, R.; Mendez, A.; Park, S.; Mannarswamy, S.; Kelly, T.; Morrey, C.B., III. Failure-Atomic Updates of Application Data in a Linux File System. In Proceedings of the 13th USENIX Conference on File and Storage Technologies (FAST '15), Santa Clara, CA, USA, 16–19 February 2015; pp. 606–614.

12. Newberry, E.; Zhang, B. On the Power of In-Network Caching in the Hadoop Distributed File System. In Proceedings of the 6th ACM Conference on Information-Centric Networking, New York, NY, USA, 24–26 September 2019; ICN '19. pp. 89–99. [CrossRef]
13. Makris, A.; Kontopoulos, I.; Psomakelis, E.; Xyalis, S.N.; Theodoropoulos, T.; Tserpes, K. Performance Analysis of Storage Systems in Edge Computing Infrastructures. *Appl. Sci.* **2022**, *12*, 8923. [CrossRef]
14. IBM. IBM Security QRadar XDR. 2023. Available online: <https://www.ibm.com/qradar> (accessed on 28 February 2023).
15. Bitdefender GravityZone Business Security Premium. 2023. Available online: <https://www.bitdefender.com/business/products/gravityzone-premium-security.html> (accessed on 28 February 2023).
16. The Industry's Most Sophisticated Endpoint Security Solution. Available online: <https://www.sophos.com/en-us/products/endpoint-antivirus> (accessed on 22 August 2024).
17. Trend Micro Incorporated. Propel Security Operations Forward. 2023. Available online: https://www.trendmicro.com/en_us/business/products/security-operations.html (accessed on 28 February 2023).
18. Szűcs, V.; Arányi, G.; Dávid, Á. Introduction of the ARDS—Anti-Ransomware Defense System Model—Based on the Systematic Review of Worldwide Ransomware Attacks. *Appl. Sci.* **2021**, *11*, 6070. [CrossRef]
19. KnowBe4. What Is Phishing? 2022. Available online: <https://www.phishing.org/> (accessed on 28 February 2023).
20. Vaas, L. Widespread, Easily Exploitable Windows RDP Bug Opens Users to Data Theft. 2022. Available online: <https://threatpost.com/windows-bug-rdp-exploit-unprivileged-users/177599/> (accessed on 28 February 2023).
21. GeekPage. Fix Slow File Copy Speed in Windows 10/11. 2022. Available online: <https://thegeekpage.com/fix-slow-file-copy-speed-in-windows-10/> (accessed on 28 February 2023).
22. AOME. Windows 10 File Copy/Transfer Very Slow. 2022. Available online: <https://www.ubackup.com/windows-10/windows-10-file-copy-slow-1021.html> (accessed on 28 February 2023).
23. Natalie, P. Remote Desktop Protocol Use in Ransomware Attacks. 2021. Available online: <https://rhisac.org/ransomware/remote-desktop-protocol-use-in-ransomware-attacks/> (accessed on 28 February 2023).
24. Synology. What Can I Do When the File Transfer via Windows (SMB/CIFS) Is Slow? 2022. Available online: https://kb.synology.com/hu-hu/DSM/tutorial/What_can_I_do_when_the_file_transfer_via_Windows_SMB_CIFS_is_slow (accessed on 28 February 2023).
25. Ye, X.; Zhai, Z.; Li, X. ZDC: A Zone Data Compression Method for Solid State Drive Based Flash Memory. *Symmetry* **2020**, *12*, 623. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.