WEB HOSTING COMPANIES' CLIENT SOLUTIONS: A STUDY OF A STRATEGIC STANDPOINT

Alexander Chizhov *, Andriy Fesenko **

* *Corresponding author*, State University "Kyiv Aviation Institute", Kyiv, Ukraine
Contact details: National Aviation University, Liubomyra Huzara Avenue, 1, Kyiv 03058, Ukraine
** State University "Kyiv Aviation Institute", Kyiv, Ukraine

How to cite this paper: Chizhov, A., & Fesenko, A. (2025). Web hosting companies' client solutions: A study of a strategic standpoint [Special issue]. *Corporate & Business Strategy Review, 6*(1), 421–429.
https://doi.org/10.22495/cbsrv6i1siart18

Copyright © 2025 The Authors

This work is licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0).
https://creativecommons.org/licenses/by/4.0/

ISSN Online: 2708-4965
ISSN Print: 2708-9924

Received: 08.12.2024
Revised: 24.02.2025; 05.03.2025
Accepted: 17.03.2025

JEL Classification: L860
DOI: 10.22495/cbsrv6i1siart18

Abstract

Websites, as the primary unit of information on the Internet, consume a significant amount of computing resources for their operation. Web hosting companies play a key role in ensuring the digital presence of businesses by managing millions of websites and optimizing their performance, reliability, and security. This work analyzes existing hosting architectures, such as traditional shared hosting, virtual private servers (VPS), and cloud solutions with clustering. It identifies the shortcomings of current solutions in terms of resource efficiency as well as security and quality of service (QoS). The study examines the key criteria influencing the strategic decisions of hosting providers, including cost-effectiveness, security level, resource consumption, and service quality. A hybrid solution is proposed that combines the benefits of cloud hosting and distributed architecture, thereby reducing costs and enhancing service quality.

Keywords: Web Hosting Companies, Hosting Strategy, SaaS, Cloud Hosting, Server, Internet, Website, Uptime, Downtime, Optimization, Cybersecurity, Resource Management

Authors' individual contribution: Conceptualization — A.C. and A.F.; Methodology — A.C.; Formal Analysis — A.C.; Investigation — A.C.; Resources — A.C.; Data Curation — A.C.; Writing — Original Draft — A.C. and A.F.; Writing — Review & Editing — A.C.; Visualization — A.C.; Supervision — A.F.; Project Administration — A.C.

Declaration of conflicting interests: The Authors declare that there is no conflict of interest.

1. INTRODUCTION

Modern Internet is a collection of various services, with websites remaining the primary service and source of information since its inception to the present day. According to NJ (2025) and Haan (2024), as of December 2023, the total number of websites worldwide is approximately 1.1 billion. A significant portion of this number, specifically 82%, is inactive or not maintained (NJ, 2025; Haan, 2024). By inactive we mean those that contain a placeholder page or advertising materials for the sale of the domain name on which the page is hosted. The remaining 18% of resources are almost 202 million websites that are maintained and actively visited by Internet users. This number of informational Internet resources requires substantial computational resources to maintain operational functionality. Considering the fact that this number is growing rapidly, the problem of optimization is extremely relevant. Every hour, 10,500 new web resources are created, resulting in a daily growth of 252,000 (NJ, 2025). This is the scale of an average hosting company with all its clients.

According to data from the Global Digital Report (Kemp, 2023) and Statista (Petrosyan, 2025), as of January 2023, there were 5.44 billion Internet users worldwide, reflecting an increase of 98 million compared to 2022. These users represent potential website visitors, and each visit contributes to the computational load on data center resources. A key trend in recent years has been mobility. Users

are spending more time online, which means they visit websites more frequently, further increasing the strain on computing resources.

Shared hosting is one of the most popular website hosting models, especially for small projects, blogs, and startups. This is due to its affordability, ease of setup, and the lack of need for specialized technical skills for administration. The market share of this type of website hosting has been growing year after year and, by the end of 2024, accounted for nearly 38% (Tomkevičiūtė, 2024). Given that shared hosting is the most cost-effective option and holds the largest market share, it is safe to say that the majority of websites worldwide rely on this hosting model.

Analysis of web traffic indicates that only 31% of websites have more than 50,000 unique visitors per month (Fitzgerald, 2023), signifying that 69% of active websites, or 139 million, are lightly loaded (low-traffic).

Shared hosting has been and remains the optimal choice for small and medium-sized businesses, as well as low-budget projects that cannot afford more expensive solutions available on the market, such as virtual servers, dedicated servers, or cloud solutions.

Shared hosting is considered an outdated solution by modern standards and, as a result, has several drawbacks, with resource optimization being the most significant. The uneven distribution of traffic depending on the time of day and day of the week leads to fluctuations in data center computational loads. Hosting companies face a dilemma: if they allocate resources sufficient for average traffic loads, their servers may experience excessive strain during peak hours, potentially leading to partial website downtime. This negatively impacts uptime and lowers the overall quality of service (QoS) (Lorido-Botran et al., 2014). On the other hand, if hosting providers allocate enough resources to handle peak loads, servers remain underutilized for most of the time, consuming electricity while performing little to no useful computational work. This inefficiency contributes to unnecessary energy consumption and operational costs.

The modern web hosting market is evolving under intense competition, where companies must balance pricing, service quality, and security. Web hosting providers implement various resource management and service optimization strategies to remain competitive. One of the primary strategies in this competitive environment is maximizing website density on a single server to minimize idle time, often at the expense of service quality and security. Additionally, the shift from leasing physical data center space to renting cloud resources can help reduce maintenance and infrastructure upgrade costs. However, this transition does not solve the issue of uneven load distribution, as the underlying architecture remains largely the same. Hosting companies require new, more comprehensive strategies to reduce costs, enhance service quality, and mitigate cybersecurity risks.

The core research questions of this study are:

RQ1: How do different web hosting architectural solutions impact economic efficiency, security levels, and service quality?

RQ2: What strategic development directions can be recommended for hosting service providers?

The remaining part of the article is structured as follows. Section 2 provides a literature review on web hosting, resource management strategies, and cybersecurity. Section 3 describes the criteria for analysis and comparison of architectural solutions. Section 4 presents the research findings, including an analysis of different architectural solutions. Section 5 proposes the architectures comparison, while Section 6 concludes the study with strategic recommendations for web hosting companies.

2. LITERATURE REVIEW

The literature review in this study focuses on analyzing existing research in the fields of web hosting, resource management, and cybersecurity. Particular attention is given to strategic approaches for optimizing server resource utilization, ensuring fault tolerance, and reducing vulnerabilities in cloud computing. The review examines key studies on dynamic load balancing, hosting system security, and service quality monitoring methods.

2.1. Resource management in web hosting

Urgaonkar et al. (2002) examined the issue of resource redistribution in shared hosting environments and proposed a dynamic load-balancing mechanism. Their study demonstrated that adaptive server management algorithms can reduce website downtime and improve server utilization efficiency. In contrast, Gul et al. (2019) focused on energy efficiency in servers, analyzing the relationship between central processing unit (CPU) time consumption and random access memory (RAM) usage in cloud systems. Their research confirms that intelligent load distribution not only reduces server maintenance costs but also enhances QoS.

2.2. Cybersecurity in cloud environments

Gruschka and Jensen (2010) analyzed the attack surface of cloud services and identified key vulnerabilities, including cross-site scripting (XSS), structured query language (SQL) injection, and authentication-level attacks. Their study emphasized the importance of multi-factor authentication and dynamic threat monitoring in cloud systems to mitigate these risks. Similar conclusions were drawn by Theisen et al. (2018), who proposed a classification of attacks on cloud infrastructures and methods for their prevention. They found that network segmentation and user privilege limitations significantly reduce the likelihood of cloud server breaches. Additionally, Alghuraibawi et al. (2021) investigated cloud service protection against distributed denial-of-service (DDoS) attacks, highlighting that distributed anomaly detection systems help reduce the risk of server overload. Their work demonstrated that a combination of behavioral analysis and machine learning enables rapid detection and mitigation of DDoS attacks. The traffic filtering models and adaptive scaling mechanisms proposed in their study allow for dynamic load redistribution, preventing service disruptions and enhancing system resilience.

2.3. Monitoring methods and fault tolerance management

Singleton (2002) studied the impact of uptime on website performance and proposed a real-time server availability monitoring method. His research demonstrated that even a slight increase in downtime leads to reduced user satisfaction and customer attrition. Campbell et al. (2015) explored information technology (IT) infrastructure reliability management strategies and proposed predictive server maintenance methods. Their study confirmed that automated fault detection systems significantly reduce the likelihood of hardware failures, ensuring higher service reliability. Luo et al. (2015) investigated update planning in open-source environments and its effect on hosting infrastructure reliability. They developed mathematical models to predict the impact of updates on service uptime, which is crucial for maintaining the stability of web hosting platforms.

The analysis of existing research indicates that the primary challenges in web hosting are related to efficient resource management, enhanced security, and ensuring high uptime. Despite significant advancements in dynamic load balancing and cloud security methods, there remains a need for further development of hybrid solutions. Such solutions should maintain the affordability and accessibility of shared hosting while integrating the fault tolerance, security, and resource efficiency found in enterprise-level hosting solutions.

3. CRITERIA FOR ANALYSIS AND COMPARISON OF ARCHITECTURAL SOLUTIONS

As already noted, during the research, architectural solutions are compared based on criteria such as service quality, efficiency in resource utilization, security, and cost.

3.1. Quality of service (QoS)

The primary criterion for the quality of an online service is its availability or the time of uninterrupted operation (uptime). Essentially, this is the difference between the total time the service is operational and the time it is unavailable (Parr & Larter 1999; Singleton, 2002; Welch, 2018). It is common to express this parameter as a percentage:

$$uptime = \frac{At}{Tt} \tag{1}$$

where,

- At — the time when the website was accessible; in the study, we consider availability within the local data center network, thereby ignoring external factors such as failures in intermediate network equipment;
- Tt — the total time of service provision.

The goal is to achieve this metric not lower than in existing architectures.

3.2. Efficiency of computational resource utilization

The criterion for the efficiency of computational resource utilization is the average usage of processor time and RAM during a monitoring period (Gul et al., 2019, 2020). In real systems, it is recommended to use a period equal to a calendar week to account for varying numbers of visitors and their activity at different times of the day and on different days of the week. For a test environment, a shorter time period can be used, provided a load profile is created to emulate peaks during working hours and declines during nighttime and weekends.

$$CPU(t) = 1 - \frac{1}{t}\int_0^t CPU \tag{2}$$

$$RAM(t) = 1 - \frac{1}{t}\int_0^t RAM \tag{3}$$

where,

- CPU — the percentage of CPU idle time at a specific moment in time;
- $CPU(t)$ — the percentage of CPU time utilization during a test interval;
- RAM — the percentage of free memory at a specific moment in time;
- $RAM(t)$ — the percentage of memory utilization during a test interval.

Based on the server's load at a specific moment in time, we can draw conclusions about the availability of websites on this server and thus determine their uptime. The criterion for website unavailability is the exceeding of the server load beyond an acceptable threshold:

$$CPU < CPUmin$$
$$RAM < RAMmin$$

In this way:

$$downtime = \sum_0^t (CPU < CPUmin) + \sum_0^t (RAM < RAMmin) \tag{4}$$

$$uptime = \frac{t - downtime}{t} \tag{5}$$

Since we are dealing with a multi-server architecture, the final result should consider the sum of metrics from all servers. To assess the overall efficiency of the solution, it is necessary to provide a relative indicator of server resource utilization that takes into account not only the load on each server but also their quantity. The number of utilized servers represents the most indicative economic criterion. The economic indicator of the used equipment can be expressed through the total number of processor cores and gigabytes of RAM across all servers in the cluster required to serve a single website. To obtain a single numerical indicator for the utilized equipment, we introduce weighting coefficients as follows:

- $w1$ — the weighting coefficient for the total number of processors;
- $w2$ — the weighting coefficient for the total amount of memory.

$$w1 + w2 = 1 \tag{6}$$

$$EQ = \frac{w1\sum_0^n Cn + w2\sum_0^n Rn}{w} \tag{7}$$

where,

- n — the total number of servers that serve the websites;
- Cn — the number of cores on server n;
- Rn — the amount of RAM in gigabytes on server n;
- w — the total number of all websites on all servers;
- EQ — the ratio of required server hardware per website.

In the majority of servers used for web services and websites, the amount of memory (in gigabytes) is typically four times greater than the number of processor cores (DigitalOcean, 2024; "Google compute engine", n.d.). Therefore, we assume that $w1 = 0.8$ and $w2 = 0.2$.

In systems employing an architecture with a dynamic number of servers, it is imperative to take into account the fact that the quantity of engaged hardware may fluctuate at different temporal instances.

$$EQ(t) = \frac{1}{t}\int_0^t EQ \tag{8}$$

3.3. The cost of owning

The cost of initial setup and maintenance for each solution for the end-user is considered as a relative quantity. If the client configures the infrastructure and server software independently, the cost of initial setup is taken as 1; otherwise, it is considered as 0. Similarly, the cost of maintenance is accounted for. In the case of self-maintenance, the cost is 1; otherwise, it is 0.

$$\begin{aligned} Ps &= \{0,1\} \\ Pm &= \{0,1\} \end{aligned} \tag{9}$$

where,

- Ps — setup price;
- Pm — maintenance price.

The final ownership cost (P):

$$P = Ps + Pm \tag{10}$$

3.4. Security

We subjectively assess the security of solutions based on their attack surface and potential vulnerabilities (Gruschka & Jensen, 2010; Manadhata & Wing, 2004; Theisen et al., 2018; Gnatyuk et al., 2020). The attack surface of websites and servers refers to the potential points of entry that malicious actors can exploit to compromise the security of a system. It includes all the avenues, interfaces, and interactions through which an attacker might attempt to gain unauthorized access, disrupt services, or extract sensitive information. Here are some aspects to consider regarding the attack surface that we should consider when analyzing it and comparing existing architectures against the proposed solution:

- *Server-side vulnerabilities*: unpatched software, outdated operating systems, and misconfigurations can provide entry points for attackers;
- *Network infrastructure*: open ports, insufficient firewalls, or poor network segmentation, can be exploited;
- *User authentication and authorization*: flaws in authentication mechanisms, session management, or insufficient authorization controls can be exploited;
- *DDoS attacks*: DDoS attacks can overwhelm servers and disrupt services, making them unavailable to legitimate users.

4. ANALYSIS OF POPULAR ARCHITECTURES

4.1. Classic shared hosting

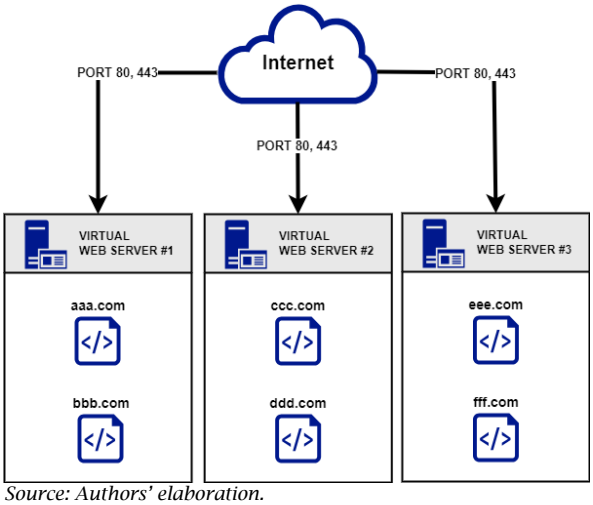
The classic shared hosting is an architecture in which numerous websites are hosted on a single virtual server, sharing its resources evenly (Urgaonkar et al., 2002; Mirheidari et al., 2012). In this architecture solution, incoming requests to the websites are placed in a First In, First Out (FIFO) queue (Castillo at al., 2012; Xiao et al., 2008), which can be expressed as a tuple Q:

$$Q = (S, S_{max}, enq, deq, first, isEmpty) \tag{11}$$

where,

- S — is the set representing the elements in the queue;
- S_{max} — maximum allowable number of elements in the queue;
- *enq:* $S \times E \rightarrow S$ is the enqueue function that adds an element to the queue; it takes the current state of the queue S and an element E to produce a new state of the queue;
- *deq:* $S \rightarrow S$ is the dequeue function that removes the front element from the queue, it takes the current state of the queue S and produces a new state of the queue;
- *first:* $S \rightarrow E$ is the function that returns the front element of the queue without removing it; it takes the current state of the queue S and returns an element E;
- *isEmpty:* $S \rightarrow \{True, False\}$ is a predicate that checks whether the queue is empty; it takes the current state of the queue S and returns either *True* or *False*.

Figure 1. Classic shared hosting architecture



Source: Authors' elaboration.

When it is necessary to host a considerable number of websites, multiple servers are deployed, and the websites are distributed among the servers with a specific limit. Thus, each server represents an independent queue.

$$Q(n) = \{Q_1, Q_2, \ldots, Q_n\} \tag{12}$$

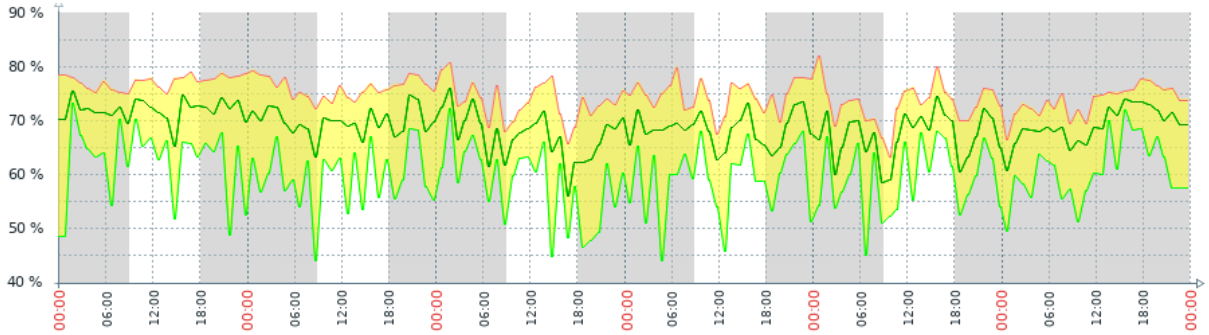
An evident drawback of such an architecture is that, during increased loads on websites hosted on a single server, there is no possibility to borrow computational resources from neighboring, less burdened servers (Urgaonkar et al., 2002; Mirheidari et al., 2012). The isolation of queues from each other, coupled with the interdependence of websites on the same server, introduces another disadvantage. In the event of excessive load on one or several websites on a single server, the QoS for all websites on that server will be compromised. When the server exceeds its permissible load and becomes non-functional, all websites on that server cease operation for the duration necessary to restore server functionality. Similarly, websites lose functionality during scheduled server maintenance activities, such as software updates and the need for rebooting (Luo et al., 2015). All the aforementioned factors prevent achieving a website uptime equal to 1, even in theory.

In real-world scenarios, hosting companies are often forced to provide inflated hardware requirements to prevent service disruptions caused by sudden load increases on individual websites. As a result, servers tend to remain moderately loaded for a significant portion of the time.

To ensure performance stability and accommodate unexpected traffic spikes, it is a common industry practice to maintain a safety margin of at least 20% of server resources. Consequently, a portion of CPU and memory capacity remains intentionally underutilized.

The following CPU and RAM utilization patterns illustrate typical behavior observed in production environments of modern shared hosting platforms. The data has been generalized based on common usage scenarios and real-world operational experience.

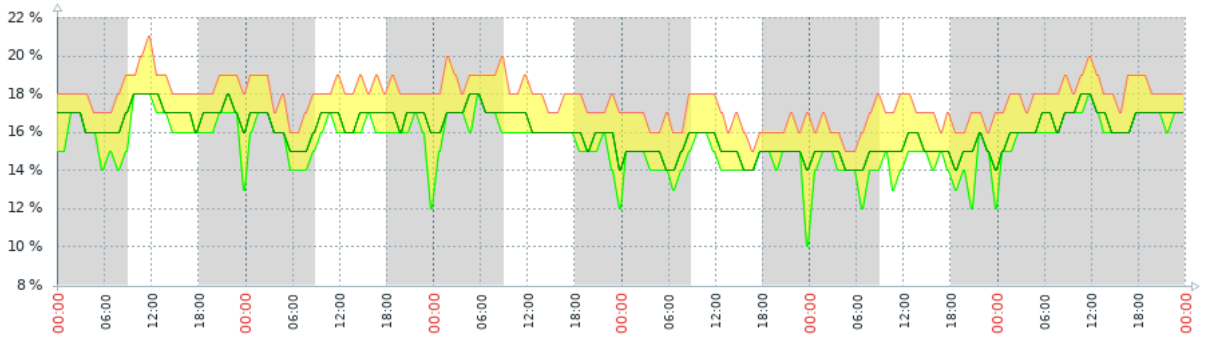
Figure 2. CPU idle time during a week (5 min average)



Note: The aggregated data were collected from multiple hosting servers across various hosting providers. The authors assume full responsibility for the accuracy and reliability of the presented data.

It is evident that, on average, approximately 68% of CPU time remains unused.

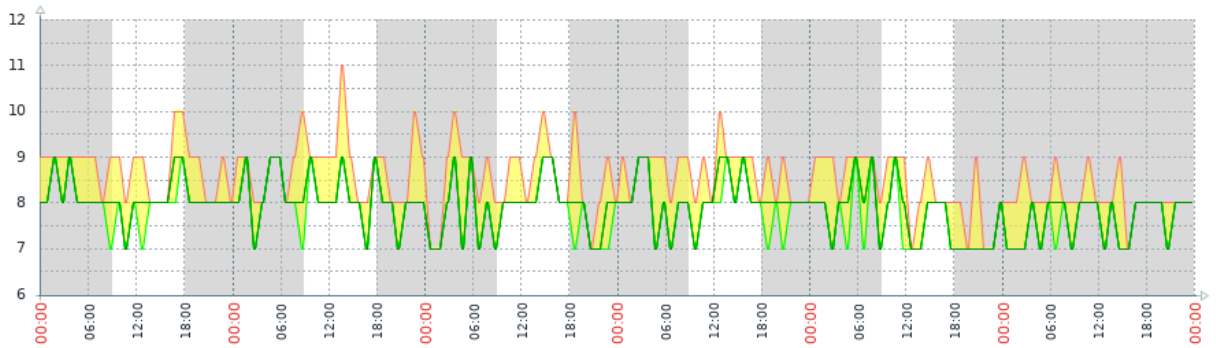
Figure 3. RAM availability during a week (in %)



Note: The aggregated data were collected from multiple hosting servers across various hosting providers. The authors assume full responsibility for the accuracy and reliability of the presented data.

It is apparent that, typically, around 16% of the available RAM is not in use.

Figure 4. Requests queue size: Number of Apache processes



Note: The aggregated data were collected from multiple hosting servers across various hosting providers. The authors assume full responsibility for the accuracy and reliability of the presented data.

The number of requests in the server queue is not significant, indicating lightly loaded websites. Despite having surplus resources on each server, there is still a risk of increased load on a particular website leading to server overload and, consequently, downtime for all websites on that server.

From a security standpoint, this architecture is highly vulnerable to various types of attacks due to servers being exposed to the public Internet and directly handling requests, significantly expanding the attack surface (Gruschka & Jensen, 2010; Manadhata & Wing, 2004; Theisen et al., 2018; Gnatyuk et al., 2020).

From the perspective of rental and maintenance costs for end-users, such an architecture is the most cost-effective. It does not require expenditures for initial setup and ongoing maintenance. All costs related to the initial configuration of infrastructure and server software are absorbed by the hosting company. The hosting company is also responsible for server maintenance, repairs, and server software updates.

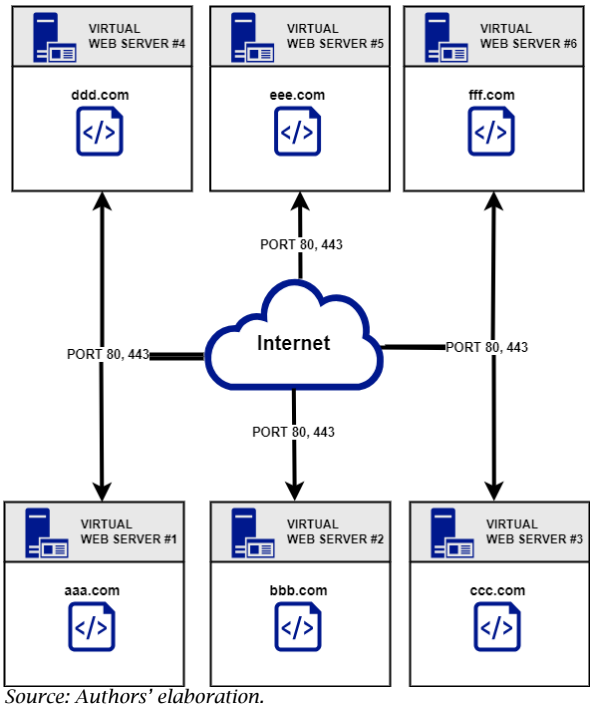
4.2. Virtual private server

A VPS is a solution wherein the client rents a dedicated virtual server exclusively for their website (Almurayh, 2010; Westfall, 2021). The advantages of this architecture lie in the fact that all the computational resources of the server are allocated solely for one website, and its operation is independent of other Internet resources hosted by the provider.

In the virtual private server architecture (Figure 5), the evident drawbacks include the cost of initial setup and support, as the client bears the full expenses for these services (Westfall, 2021). Payment for the services of a specialist for configuring and systematically maintaining a dedicated server makes such a solution unjustifiably expensive for a lightly loaded website.

The security level can be considered conditionally lower than that of classic shared hosting, as this solution shares similar challenges with a larger attack surface. Additionally, it is noteworthy that the maintenance of a dedicated server on demand may be somewhat less efficient and less prompt compared to the services provided by in-house system administrators in a hosting company, who typically manage and monitor equipment operation 24/7.

Figure 5. Virtual private server architecture



Source: Authors' elaboration.

4.3. SaaS cloud shared hosting

As an alternative architecture, it is proposed to combine two technologies, where a website shares a server with others, as implemented in classic shared hosting, but the servers are organized into a cluster. This approach allows the client to benefit from the simplicity and cost-effectiveness of classic shared hosting setup and maintenance, along with additional advantages related to website resilience and security due to the clustering.

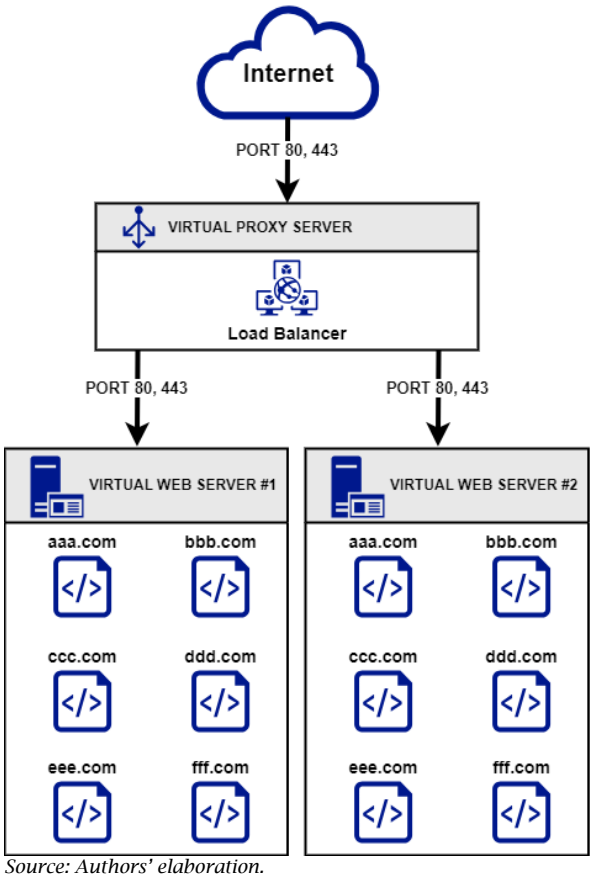
Let's examine this architecture from the perspective of the efficiency criteria established within the framework of this study.

From the client's perspective, the efficiency of the website is enhanced as it is independent of the load on other websites in the cluster. In case of resource shortages, automatic horizontal scaling is implemented by adding additional servers to the cluster. Each website receives the necessary amount of resources, minimizing the probability of overloading a single server and leading to increased uptime for all websites in the cluster. During

planned maintenance, rebooting one server in the cluster does not disrupt the operation of any website. These factors theoretically allow achieving uptime = 1 (Chizhov et al., 2024).

From the hosting company's standpoint, this architecture brings significant economic benefits. Dynamic resource allocation enables the efficient utilization of CPU time and RAM for extended periods, reducing the required amount of hardware to serve the same number of web resources. Each website, in perspective, will consume a smaller amount of CPU time and RAM, leading to an overall increase in QoS.

Figure 6. SaaS cloud shared hosting architecture



Source: Authors' elaboration.

Due to the company using fewer computational resources to serve the same number of websites, the cost of services for end-users may be lower, concurrently with an increase in the company's profit (Chizhov et al., 2024). In this architecture, the hosting company assumes the responsibility for server maintenance and infrastructure setup, relieving the client from incurring any expenses for initial configuration and ongoing maintenance.

5. ARCHITECTURES COMPARISON

5.1. Classic shared hosting

Advantages:

- cost savings for end-users as the hosting company handles initial setup and maintenance.

Disadvantages:

- low-security level due to a shared attack surface;
- risk of server overload during high load on one or multiple websites;
- not optimal utilization of computing resources.

5.2. Virtual private server

Advantages:

- isolated computational resources for the website.

Disadvantages:

- high costs for clients for initial setup and maintenance;
- security risk due to wide attack surface and potential service delays;
- not optimal utilization of computing resources.

5.3. Clustered approach

Advantages:

- efficient resource utilization with automatic horizontal scaling;
- high-security level due to isolation;
- potential to achieve theoretical uptime = 1;
- economic efficiency for the hosting company.

Disadvantages:

- more complex setup.

Table 1. Architectures comparison

Assessment criterion	*Classic shared hosting*	*Virtual private server*	*SaaS cloud hosting*
Rent costs	Low	High	Low
Setup costs	n/a	High	n/a
Maintenance costs	n/a	High	n/a
Performance	Low	High	High
Reliability	Low	Middle	High
Security level	Low	Low	High
Resources utilization	Middle	Low	High

6. CONCLUSION

The study examined various web hosting architectural solutions, including traditional shared hosting, VPS, and cloud-based solutions with dynamic scaling. The evaluation considered the qualitative characteristics of each model from both the consumer's and the service provider's perspectives. The analysis revealed that clustered cloud architectures provide the highest efficiency in resource management, fault tolerance, and security compared to traditional approaches. The research also confirmed that the implementation of automated load management systems significantly reduces server maintenance costs while improving QoS metrics.

Each year, Internet security requirements continue to increase, and the existing hosting solutions on the market no longer fully align with modern demands. Architectural limitations prevent the resolution of identified challenges through internal upgrades alone, necessitating a fundamental

overhaul of hosting architectures. A hybrid solution has been proposed, integrating the strengths of various hosting strategies to effectively address key issues, specifically:

- enhancing resource efficiency while reducing the overall cost for both consumers and providers, thereby increasing market competitiveness and maximizing vendor profitability;
- improving service quality by maximizing uptime and reliability of the hosting infrastructure;
- strengthening security by minimizing the attack surface and introducing advanced protection mechanisms.

This study is based on theoretical model analysis and comparisons with existing practical solutions. However, there are some limitations to this research:

- lack of empirical data from real-world testing of the proposed hybrid architecture;
- limited focus on DDoS protection, as mechanisms for detection, filtering, and prevention of such attacks were not examined in detail;
- no in-depth analysis of load balancer fault tolerance, despite its crucial role in ensuring the stability of web hosting environments.

Future research should address these limitations by conducting real-world testing, exploring DDoS mitigation techniques, and evaluating the resilience of load-balancing systems to enhance the robustness of hybrid hosting architectures. To advance the proposed model, experimental testing in real-world web hosting environments is necessary. Specifically, the following areas require further development and validation:

- development of an efficient and fault-tolerant load balancing system; since the load balancer represents a critical bottleneck in the architecture, special attention should be given to its resilience, redundancy, and failover mechanisms;
- testing of clustered architecture on real servers to evaluate its performance under high-load conditions and determine its scalability and reliability.
- integration of AI for DDoS protection, incorporating anomaly detection, traffic analysis, and automated threat response to enhance security;
- implementation of automated scaling modules based on real-time monitoring, ensuring adaptive resource allocation in existing cloud solutions;
- development of user-friendly hosting management systems for end-users, enabling easy configuration of secure sockets layer (SSL) certificates, code deployment, and database integration.

The analysis of existing research reveals that the primary challenges in web hosting revolve around efficient resource management, enhanced security, and ensuring high uptime. Despite significant advancements in dynamic load balancing and cloud security measures, these solutions remain largely accessible only to large enterprises, while low-budget web resources struggle to adopt such innovations due to high costs and infrastructure limitations. Thus, this study establishes a theoretical foundation for the future development of innovative web hosting strategies that aim to enhance reliability, security, and economic efficiency in digital infrastructures. By bridging the gap between cost-effective hosting models and enterprise-grade solutions, these advancements could make scalable and resilient architectures more accessible to a broader range of users.

REFERENCES

Alghuraibawi, A. H. B., Abdullah, R., Manickam, S., & Alyasseri, Z. A. A. (2021). Detection of ICMPv6-based DDoS attacks using anomaly-based intrusion detection system: A comprehensive review. *International Journal of Electrical and Computer Engineering, 11*(6), 5216–5228. https://doi.org/10.11591/ijece.v11i6.pp5216-5228

Almurayh, A. (2010). *VPS: Virtual private server.* https://www.ssra2022.org/-cs526/studentproj/projS2010/aalmuray/doc/Almurayh_VPS.pdf

Campbell, J. D., Reyes-Picknell, J. V., & Kim, H. S. (2015). *Uptime: Strategies for excellence in maintenance management* (3rd ed.). Productivity Press.

Castillo, E., Menendez, J. M., Nogal, M., Jimenez, P., & Sanchez-Cambronero, S. (2012). A FIFO rule consistent model for the continuous dynamic network loading problem. *IEEE Transactions on Intelligent Transportation Systems, 13*(1), 264–283. https://doi.org/10.1109/TITS.2011.2169668

Chizhov, A., Fesenko, A., Ziuziun, V., & Bashykyzy, D. (2024). Cloud shared hosting DDoS resistance and potential ways of protection. In *Proceedings of the Third International Conference on Cyber Hygiene & Conflict Management in Global Information Networks (CH&CMiGIN 2024)* (pp. 13–23). https://ceur-ws.org/Vol-3925/paper02.pdf

DigitalOcean. (2024). *Choosing the right droplet plan.* https://docs.digitalocean.com/products/droplets/concepts/choosing-a-plan/

Fitzgerald, A. (2023, June 19). How many visitors should your website get? *HubSpot*. https://blog.hubspot.com/blog/tabid/6307/bid/5092/how-many-visitors-should-your-site-get.aspx

Gnatyuk, S., Sydorenko, V., Polozhentsev, A., Fesenko, A., Akatayev, N., & Zhilkishbayeva, G. (2020). Method of cybersecurity level determining for the critical information infrastructure of the state. In *Proceedings of the 2nd International Workshop on Control, Optimisation and Analytical Processing of Social Networks (COAPSN 2020)* (pp. 332–341). https://ceur-ws.org/Vol-2616/paper28.pdf

Google compute engine. (n.d.). In *Wikipedia*. https://en.wikipedia.org/wiki/Google_Compute_Engine

Gruschka, N., & Jensen, M. (2010). Attack surfaces: A taxonomy for attacks on cloud services. In *2010 IEEE 3rd International Conference on Cloud Computing*. IEEE. https://doi.org/10.1109/CLOUD.2010.23

Gul, B., Khan, I. A., Mustafa, S., Khalid, O., & Khan, A. u. R. (2019). CPU–RAM-based energy-efficient resource allocation in clouds. *The Journal of Supercomputing, 75*, 7606–7624. https://doi.org/10.1007/s11227-019-02969-5

Gul, B., Khan, I. A., Mustafa, S., Khalid, O., Hussain, S. S., & Dancey, D. (2020). CPU and RAM energy-based SLA-aware workload consolidation techniques for clouds. *IEEE Access, 8*, 62990–63003. https://doi.org/10.1109/ACCESS.2020.2985234

Haan, K. (2024, June 4). *Top website statistics today.* Forbes Advisor. https://www.forbes.com/advisor/business/software/website-statistics/

Kemp, S. (2023, October 19). *Digital 2023 October global Statshot report.* Datareportal. https://datareportal.com/reports/digital-2023-october-global-statshot

Lorido-Botran, J., Miguel-Alonso, J., & Lozano, J. A. (2014). A review of auto-scaling techniques for elastic applications in cloud environments. *Journal of Grid Computing, 12*(4), 559–592. https://www.proquest.com/docview/2259771596?sourcetype=Scholarly%20Journals

Luo, C., Okamura, H., & Dohi, T. (2015). Optimal planning for open source software updates. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability, 230*(1), 44–53. https://doi.org/10.1177/1748006X15586507

Manadhata, P., & Wing, J. M. (2004). *Measuring a system's attack surface* (Report, School of Computer Science, Carnegie Mellon University). DTIC. https://doi.org/10.21236/ADA458115

Mirheidari, S. A., Arshad, S., & Khoshkdahan, S. (2012). Performance evaluation of shared hosting security methods. In *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*. IEEE. https://doi.org/10.1109/TrustCom.2012.219

NJ. (2025). *How many websites are there in the world?* Siteefy. https://siteefy.com/how-many-websites-are-there/#How-Many-Active-Websites-Are-There

Parr, J. D., & Larter, P. C. (1999). Standardization of reliability/maintainability/availability metrics for US AFSCN common user element. In *Annual Reliability and Maintainability Symposium 1999 Proceedings* (pp. 13–18). IEEE. https://doi.org/10.1109/RAMS.1999.744089

Petrosyan, A. (2025, February 13). *Worldwide digital population 2025.* Statista. https://www.statista.com/statistics/617136/digital-population-worldwide/

Singleton, D. (2002). Website performance monitoring. In D. Meyerhoff, B. Laibarra, R. P. Kraan, & A. Wallet (Eds.), *Software quality and software testing in Internet times* (pp. 199–216). Springer. https://doi.org/10.1007/978-3-642-56333-1_13

Theisen, C., Munaiah, N., Al-Zyoud, M., Carver, J. C., Meneely, A., & Williams, L. (2018). Attack surface definitions: A systematic literature review. *Information and Software Technology, 104*, 94–103. https://doi.org/10.1016/j.infsof.2018.07.008

Tomkevičiūtė, A. (2024, September 27). Web hosting market share analysis 2025. *Cybernews.* https://cybernews.com/best-web-hosting/web-hosting-market-share/

Urgaonkar, B., Shenoy, P., & Roscoe, T. (2002). Resource overbooking and application profiling in shared hosting platforms. *ACM SIGOPS Operating Systems Review, 36*(SI), 239–254. https://doi.org/10.1145/844128.844151

Welch, N. (2018). *Real-world SRE: The survival guide for responding to a system outage and maximizing uptime.* Packt Publishing.

Westfall, J. (2021). *Set up and manage your virtual private server: Making system administration accessible to professionals.* Apress. https://doi.org/10.1007/978-1-4842-6966-4

Xiao, Y., & Zhou, R. (2008). Low latency high throughout circular asynchronous FIFO. *Tsinghua Science and Technology, 13*(6), 812–816. https://doi.org/10.1016/S1007-0214(08)72205-3