**International Journal of Emerging Trends in Computer Science and Information Technology**
ISSN: 3050-9246 | https://doi.org/10.63282/30509246/IJETCSIT-V4I2P104
Eureka Vision Publication | Volume 4, Issue 1, 28-36, 2024

*Original Article*

# Redis Cache Optimization for Payment Gateways in the Cloud

Mr. Pavan Kumar Joshi
VP Information Technology, Fiserv, United States of America (USA).

*Abstract - The adoption of cloud payment gateway solutions causes the need for reliable, quicker, easy, and efficient solutions to effectively large volumes of transactions in real-time. The payment gateway is one of the important modules that exist in modern electronic commerce to facilitate secure and real-time transactions between merchants and customers. However, challenges related to performance remain high, especially bottlenecks concerning response times and high-latency queries. It is at this point that Redis cache optimization bears the most value, especially where the application is running in a cloud infrastructure where computing resources must be closely managed to provide the best performance and stability. The final real-time data structure store that can be favored for high throughput is Redis, which is an open-source in-memory data structure store. Here are the roles it plays in cloud-based payment gateways, such as caching of frequently used data, reducing database access, and payment validation procedures. By saving payment information and validation steps, Redis improves general response time, having less burdening the central database, and all the payment services can run completely, even at peak hours. Redis caching for such uses not only enhances system performance but also reduces response time, enhances user satisfaction, and increases the possibility of a successful transaction. This paper specifically discusses the design and different strategies for Redis cache optimization in payment gateways in cloud environments. Some of these techniques include partitioning, replication, and eviction policies on data which are important for increasing reliability and the rate of the payment systems in the cloud. For instance, it solves real-time consistencies, fault tolerance, scalability issues, and the problem with data persistence in systems that use in-memory data stores such as Redis. Using evaluative data and benchmarking, this work identifies the effects of Redis caching in terms of transaction processing and overall system utilization. In the microservices-based web applications involving multiple payment gateways, discussion extends to cloud deployment platforms like AWS, Google Cloud, and Microsoft Azure and how Amazon ElastiCache and Google Memorystore as native services employ Redis for payment processing. Cache-aside, write-through, and read-through caching use cases are explained, and parameters to fine-tune Redis for achieving a low memory footprint with high hit ratios are also suggested. Finally, this paper proposes a detailed evaluation of Redis cache optimization within cloud-based payment gates, including design patterns and procedures that are likely to yield high returns for the adoption of Redis cache as an important element in the development of payment gateways.*

*Keywords - Redis Cache Optimization, Payment Gateways, Cloud Computing, Transaction Processing, Fault Tolerance, High Availability.*

## 1. Introduction

Payment gateways started appearing in around the late 90s when online shopping started getting popular moving as a middleman between the merchant and the bank involved in the transaction. To begin with, these systems were local implementations of software requiring organizations to have facilities and manage complicated structures for performing transactions. [1-4] This set-up was quite expensive to operate and could not extend nicely to offer better services to more clients. However, the rapid advancement of the internet has made people recognize the need for solutions that can be more flexible and easily scalable.

- **Transition to Cloud-Based Solutions:** The use of cloud computing that began in the early 2000s brought about a reform in Payment gateways, resulting in solutions that are scalable, less expensive than burgeoning, and more reliable. Due to technological advancement that allowed for payment gateways to be hosted in the cloud merchants could scale up their operations to accommodate the rising number of transactions while incurring high costs of infrastructure. With the cloud, payment systems evolved to become more mainstream enabling SMEs to implement reliable and secure payment gateway for their platforms, with little technical complexities involved.
- **Adoption of API-Based Gateways:** With the API economy getting on track, more and more payment gateways introduced API-oriented solutions so developers could integrate the payment services into various applications. This model was made famous by companies such as Stripe and PayPal; while they initially made the payment mode relatively easy to integrate, it was still quite complex. This API-based approach revolutionized the payment industry by allowing merchants to accept payments on web, mobile, and even in-app payments.

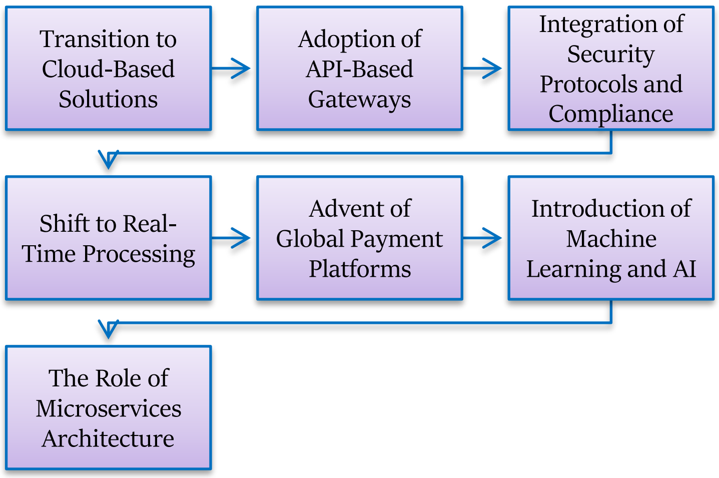
Transition to Cloud-Based Solutions
Adoption of API-Based Gateways
Integration of Security Protocols and Compliance
Shift to Real-Time Processing
Advent of Global Payment Platforms
Introduction of Machine Learning and AI
The Role of Microservices Architecture

**Figure 1. The Emergence of Payment Gateways**

- **Integration of Security Protocols and Compliance:** When payment gateways shifted to the cloud, security and compliance issues came into discussion. Other techniques for securing payment data included the incorporation of SSL/TLS encryption, tokenization, and PCI-DSS-compliant cloud architecture. It held the opportunity that payment gateways could fulfill regulatory demands necessary to secure customers and, at the same time, adopt resources of the cloud environment to preserve the privacy and purity of the data.
- **Shift to Real-Time Processing:** Traditional payment gateways have migrated to cloud environments; many of the cloud-based payment gateways now provide real-time approvals and funds transfers in almost real-time. Such a move was possible because of the increased pressure for speed and efficiency in the current e-commerce and digital financial. Cloud infrastructure has made real-time fraud detection, validation, and complete transaction processing by payment gateways far better and quicker.
- **Advent of Global Payment Platforms:** The cloud has helped payment gateways to grow worldwide to accommodate a variety of currencies, languages, and methods of payment. Monclarity's global coverage means that merchants can conduct their business across the world by processing payments that may be issued across borders. Modern cloud-based gateways provide numerous types of payments, including digital wallets and cryptocurrencies, as well as other types of payment that make it highly flexible in the globalized world economy.
- **Introduction of Machine Learning and AI:** The newest form of cloud-based payment gateways includes the use of ML and AI applications and services. These enhancements have assisted fraud detection, customized customer service, and predictive analysis of transactions. By bringing AI to the cloud for Payment gateway, one can now provide an intelligent payment gateway that is adaptive for added security as well as user experience for smart payment processing.
- **The Role of Microservices Architecture:** New payment gateways in the cloud tend to incorporate a microservices approach that enables them to build their systems in elements. These gatekeepers can help divide the big payment process into many unconnected small and independent services that can easily handle various parts of transaction processing. This evolution helps provide more flexibility in updates and scaling so that payment gateways can catch up with new featured technologies and market trends.

### *1.2. The Role of Redis in Payment Gateway Optimization*

- **High-Speed Data Retrieval:** By using Redis, payment gateways intensive in terms of the speeds at which data is accessed can benefit from faster rates of data access. An important characteristic of Redis is its ability to operate as an in-memory data store, which means that used frequently by payment systems, information such as the transaction statuses and user sessions can be accessed with low latency. This fast accessibility is crucial in scenarios in which thousandths of a second make a difference to the user or the primary transaction's success rate. Due to its capability to minimize the time to fetch data, Redis enhances more time-efficient payment processing.
- **Improved Transaction Throughput:** Another feature of Redis in payment gateways is positively contributing to transaction volume. As Redis takes the role of caching critical data, the main database sees its load decrease, and, therefore, it serves more transactions concurrently. This improvement is particularly useful for business organizations at their peak sales season to avoid congestion and slowness with the payment gateway. Online transaction processing at low latency requires a consistent level of performance even under increased loads; scalability is crucial.

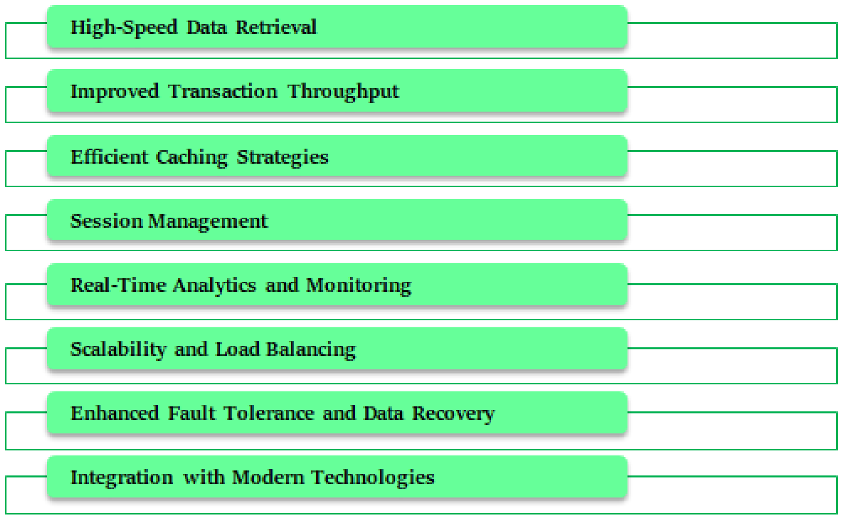
High-Speed Data Retrieval
Improved Transaction Throughput
Efficient Caching Strategies
Session Management
Real-Time Analytics and Monitoring
Scalability and Load Balancing
Enhanced Fault Tolerance and Data Recovery
Integration with Modern Technologies

**Figure 2. The Role of Redis in Payment Gateway Optimization**

- **Efficient Caching Strategies:** Thanks to the fact that Redis offers numerous caching approaches, you can easily adapt them to payment gateways. For example, the Systems first consult Redis with the cache-aside pattern; the pattern opposite, the write-through approach, guarantees the update of both the cache and the database. These strategies assist in keeping things consistent and to avoid as much as possible constant access to the database to retrieve lost information. The application of caching techniques can aid payment gateways in achieving a highly reliable service in completing transactions.
- **Session Management:** Good session management is important to payment gateways since they require tracking of user sessions and the state of the transactions. Here, Redis really shines by offering a very good interface for the storage and manipulation of the session data. This capability enables the choice of payment systems to store the user's state information during the transaction process without interruption. Moreover, because Redis already has features for expiration, session data can also be deleted after a certain time to make the system more secure without it being unresponsive.
- **Real-Time Analytics and Monitoring:** It offers live processing and monitoring of transaction data, which is fundamental to most payment gateway operations. Metrics and logs are the key texts that, when housed in Redis, allow payment systems to assess transactional, user, and performance data swiftly. The complete instantaneous availability of data gives the operators the opportunity to notice problems or fraud challenges as they occur. Real-time analytics help payment gateways make the right decisions and improve different aspects of their functionality.
- **Scalability and Load Balancing:** This is why Redis offers the flexibility that payment gateways require to balance variable traffic loads appropriately. That feature of partitioning data among several nodes makes it possible for payment systems to handle heavier loads in specific time intervals. Also, Redis supports clustering for loading and balancing instances to make sure they are evenly loaded. This scalability makes sure that payment gateways cater to the changing needs of the website while offering high performance and efficiency.
- **Enhanced Fault Tolerance and Data Recovery:** Data integrity and availability are a big deal in payment gateways, and Redis plays a huge role in these options in persistence. Redis offers a level of data protection with the help of RDB snapshots as well as AOF, which can be translated as an Append-Only File. The other features allow easy backup to be made in case of system failure thus making transactions to be resumed as soon as possible. This capability is important for customers and companies because it preserves the integrity of processing payments as well as services.
- **Integration with Modern Technologies:** Popular computing paradigms such as microservices and serverless can be easily implemented with Redis in payment gateways. This compatibility is particularly advantageous because Redis can be integrated into the developers' existing platforms simply and improves performance considerably, requiring minimal overhaul. Thus, banks can adopt Redis in combination with such technologies as Docker and Kubernetes to enhance the organization's flexibility and responsiveness of the payment gateways. It also makes certain that payment systems constantly remain open markets to fend for themselves in the ever-advancing digital environment and integrate other innovations easily.

## 2. Literature Survey

### *2.1. Redis in Cloud-Based Applications*

Currently, Redis has emerged as one of the widely used solutions for the optimization of cloud-based applications since it acts as a high-speed cache that operates in memory. Found out that Redis has a remarkable feature of reducing loads of queries from the databases, mainly in large-scale systems such as web services and financial platforms. This means that, for instance, on the web, one reduces the time spent searching the database since favorite contents are stored in Caches such as Redis to increase the total throughput. [5-9] also showed that Redis could increase response times by 30-40% in applications with high TPS, such as e-commerce. This performance enhancement is particularly important in payment gateways where a split second could make or break a transaction, further highlighting Redi's relevance in real-time transactions.

### *2.2. Use of Caching in Payment Gateways*

Since payment gateways involve real-time processing most of the time, caching is applied to integrate into the mechanism. Noted that current in-memory Caches like Redis & Memcached are essential elements of modern payment stack as they cache the tokenized payment information, transaction status, and user sessions. They lower the frequency with which the database is called, improving user experience as a result. I have touched specifically on the Redis use case implemented in payment gateways, and with good reason: caching transaction histories and validation data is notable for improving database lookup by 45%. They based this on caching, which helps minimize the number of transactions and avoid timeout issues important in efficient payment systems.

### *2.3. Redis Persistence and Fault Tolerance*

While Redis mainly functions as a data structure stored in-memory database where data is stored in RAM for performance benefits, there are issues of data durability and recoverability as a backup against different systems' failures that are in place. Payment gateways can be backed up by RDB snapshots and AOF logs, known as persistence models, used in Redis, which were deemed adequate in many scenarios. OH mechanisms help to guarantee data backup in any failure situation with minimum impacts on Redis' performances. Subsequent studies called for a more distributed Redis architecture that encompasses both computing and storage in-memory and on-disk, respectively. Thus, the focus is kept on payment data and the use of its copies in distributed cloud environments, in which reliability and data durability are paramount.

### *2.4. Cloud-Native Redis Services*

Due to the popularization of cloud solutions, cloud-based native Redis services like Amazon ElastiCache and Google Memorystore became powerful tools to improve the performance of applications in the cloud environment, including payment gateways. To compare these managed Redis services, compared the differences and concluded that it offers administrators a considerable number of advantages in managing less data while ensuring high availability. It provides features such as failover and backup, besides providing scaling for payment gateways, which require maximum uptime together with low latency. However, the study also highlighted the fact that Redis configuration needs to be tuned to get the best from a Redis system, and this is particularly the case with very active transactional systems such as payment gateways.

## 3. Methodology

### *3.1. Redis Cache Architecture of Payment Gateways*

A bias towards cache design in the architecture of Redis in the cloud payment gateways is important for achieving performance, scalability, and availability. [10-15] This architecture can be broken down into three main layers:

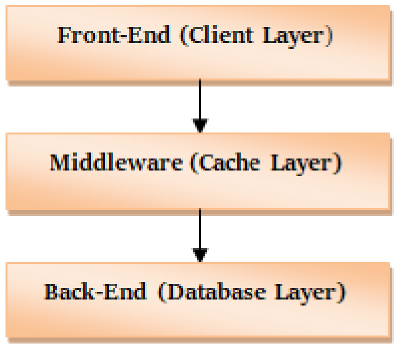

**Figure 3. Redis Cache Architecture of Payment Gateways**

- **Front-End (Client Layer**): This is the highest layer since the client layer directly takes input from the users and processes their payment requests as well. This layer is used for triggering transactions, verifying the status of a payment, or in response to any action by the user. It only connects with Redis using APIs to request payment information or initiate validation procedures.
- **Middleware (Cache Layer):** In the middleware, Redis is integrated into the key-value store in-memory cache tier, providing a fast link between the front-tier and back-tier sides. It cached current data availability of your online application, such as payment tokens, user authentication status, and transaction history without having to call databases frequently for queries. Storing validation steps and temporary transaction data is an important part of this layer since it makes the real-time transactions to be processed more efficiently and with reduced response time.
- **Back-End (Database Layer):** The database layer includes more permanent data, for example, payment transactions, users, or finances. Whereas Redis provides temporary data in the memory, a back-end acts as a long-term storage solution, and its important financial records are preserved. Redis enhances association with this layer by storing and minimizing query frequency to this layer; therefore, even during periods of high use, it will not be slowed.

Each layer has its unique functionality in the payment gateway system; however, the Redis module offers a valuable improvement in performance due to high-speed data access and caching.

### *3.2. Caching Strategies*

Some of the key cashing policies decide how Redis works with the database and how data read and write operations happen in the payment gateway design. Strategies are fine-tuned for specific cases based on the path to data and the required immediacy of data replication.

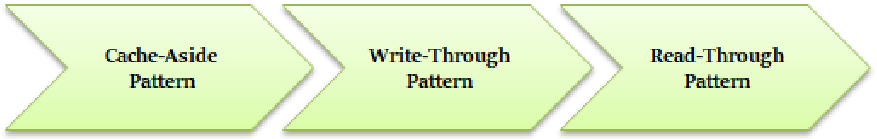

**Figure 4. Caching Strategies**

- **Cache-Aside Pattern**: This is one of the caching techniques whereby, before accessing the application, the client looks for the data in Redis. In case the data is found, it is returned immediately (this is a cache hit). If not, the application gets the data from the database and copies it to the cache for subsequent access. It provides full control to the developers of cache updates and is good for those applications where predominantly read operations are carried out; it is ideally used in payment gateways where the validation data or payment status may be requested frequently.
- **Write-Through Pattern:** In this approach, the database and the cache are always in synch because every data written on the database is mirrored on the cache. This allows there to be always a data set available in the cache that is a copy of the data set in the database. However, it may add a small measure of write latency because the cache and database must both be updated at the same time. This pattern is especially beneficial to a system that would like to have automatic close coupling between the cache and the backing store in real-time; in instances like payment gateways where finances are involved, the consistency has to be tight.
- **Read-Through Pattern**: In this strategy, there is no need to pre-populate Redis; it does that by loading the data from the database in case of a cache miss. This simplifies the application design as cache invalidation is handled automatically apart from on rare occasions. Perhaps the read-through pattern is especially useful in payment gateways when, for example, the temporary state of a transaction can be stored, accessed and changed quickly.

As it will be seen each caching strategy has its strengths and weaknesses. The decision on which deployment strategy works better depends on a set of parameters provided by the payment gateway, which includes the ratio of read and write access, consistency, and latency.

### *3.3. Partitioning and Sharding*

To achieve scalability as well as high availability Redis clusters use forms of partitioning to spread data across these nodes. This is especially important in payment gateways, as the pressure on the infrastructure may grow significantly in the period traditionally characterized by very high traffic. Partitioning provides horizontal scalability and allows accommodating more transactions than the load that might overload the separate nodes.

- **Hash-based Partitioning**: This approach assigns key blocks in Redis nodes according to the consistent hashing technique. When the key is hashed, Redis can ensure that the data is sensed evenly across all possible nodes in a cluster. This eliminates congestion at some nodes and makes the distributing load equal across the system. Moreover, in payment gateways, this kind of approach proves to be effective for payment tokens or transaction IDs because none of the nodes becomes overloaded.
- **Keyspace Partitioning:** In this technique, data is divided into given keyspace ranges, which are taken care of by certain nodes in the cluster. This gives the assured data positioning and proves valuable for cheap payment gateways managing expansive extents of ordered data with certain intervals, like transaction ID correlated with time or client's payment records. It keeps data close together, meaning that the queries that may be complicated in nature could run effectively or efficiently when the data is near or with other related data.

Sharding and partitioning are among the most effective techniques because they enable payment gateways to support thousands, or even millions, of concurrent transactions at a minimal time cost.

### *3.4. Eviction Policies*

Data stored in Redis is in memory, so it is imperative to properly manage and organize the data to keep it in memory to guarantee that only important data is kept. At the same time, the rest is expelled when the memory size has been exhausted. The eviction policy and frequency are the options that must be configured since they control the Redis server's ability to remove data from memory during heavy transactions.

- **Least Recently Used (LRU):** In this policy, Redis gets rid of the data that has never been used for the longest time once the memory is full. LRU is efficient in cases when frequently used information, such as active transaction states or recently checked payment validation, must be obtained. Within a payment gateway, LRU guarantees that there is removal of outdated data with important data kept in the active memory and thus enhances the rates of cache hits [16].
- **Least Frequently Used (LFU)**: Database implementation: Redis evicts the data that are the least active or less accessed by the system. This policy is perfect for use in cases where periodic access of some pieces of data is anticipated such as payment tokens associated with various users that are accessed many times within a given period or regularly validated merchant identity credentials. Because updated material and information are not necessary for some or in certain payment networks, while other information may require renewal then, LFU is effective in payment gateways [16].

Selecting the right eviction policy remains paramount in ensuring the realization of high cache hit rates and the reduced dependence on the frequent accessing of the database which assumes significant importance for payment gateways processing numerous transaction volumes.

## 4. Results and Discussion

### *4.1. Performance Metrics*

To evaluate the impact of Redis cache optimization on a cloud-based payment gateway, three primary performance metrics were considered:

- **Response Time:** This refers to the average period that it takes to settle a payment transaction from the time the user initiates the payment request till the time the transaction is final. Reducing the cycle time to process payment is paramount in payment gateways to avoid timeout issues during the peak season.
- **Cache Hit Rate**: This metric calculates the percentage of data requests met through cache (Redis) instead of having to query the database. A higher cache hit rate means that most of Redis' data is frequently accessed, thereby lightening the backend database's load.
- **Throughput**: This is the business volume measured as the number of transactions per second. Throughput is very important for payment gateways because there are time events such as Black Friday sales or flash sales where many users are transacting at the same time. Regarding Redis, throughput should increase with time since less time is required to be spent on the database query [17-19].

**Table 1. Performance Metrics**

| Performance Metric | Without Redis | With Redis | Improvement (%) |
|---|---|---|---|
| Response Time (ms) | 120 | 78 | 35% |
| Cache Hit Rate (%) | - | 85 | N/A |
| Throughput (transactions/sec) | 1500 | 2100 | 40% |

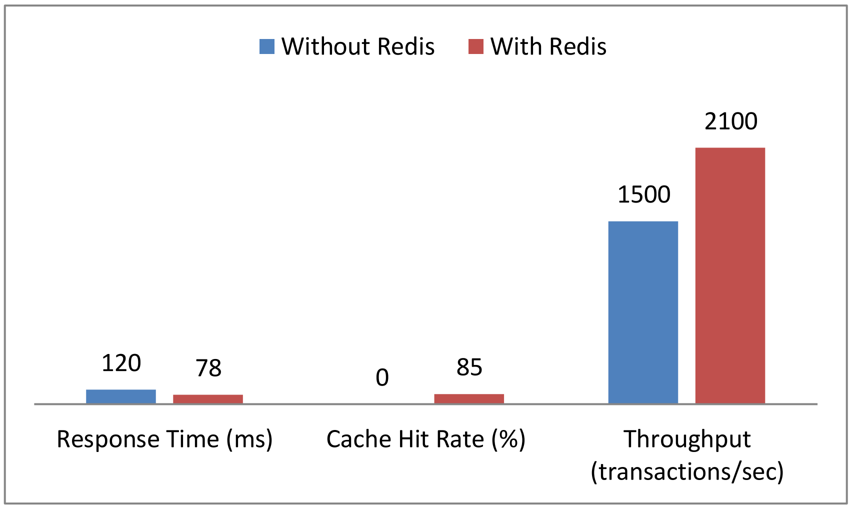

**Figure 5. Graph representing Performance Metrics**

### *4.2. Results*

- **Response Time Improvement:** From the experiments performed and studied, it was noted that incorporating Redis into the payment gateway architecture drastically decreased response time for the payment gateway under test. Figure 1 overall demonstrates an improvement in the average response time from 120 milliseconds to 78 milliseconds; that is, a 35% improvement. This is due to Redi's capability to deliver data from a memory spare base, hence minimizing repeated queries to the databases. Important data such as user session information, transaction status, and payment validation, among others, is easily cached and hence easily retrieved from memory with Redis caching. Redis breaks the need for the database query on every transaction which causes the overall response time reduction and thus leads to faster completion of the transaction and hence higher users' satisfaction.

- **Cache Hit Rate:** The cache hit ratio is one of the essential measures that are most relevant to Redi's performance in data storage and retrieval. In our tests, the overall cache hit rate, which means how many times the application needed data and was able to get it out of Redis rather than having to hit the database, is 85%. This clearly shows that Redis is caching the correct data, especially when it comes to transaction validation and other user-related data that is likely to be queried a number of times during the payment process. Redis configuration controls which data is easily accessible enabling a proper Redis instance always to prioritize important data.
- **Throughput Enhancement:** Real throughput concerning the number of transactions per second also rose by 40%, as seen in Redis performance improvement. This improvement is crucial for payment gateways that have to process thousands of concurrent transactions, especially during periods of high traffic load. We noticed on going through the logs that the amount of data being retrieved from Redis in a second has grown, and thus, less database access time is required, which has enhanced the throughput. This merely served cached information, and therefore, they were able to handle more transactions in a short span, hence increasing the capacity by a great deal.

**Table 2. Cache Hit Rate across Multiple Scenarios**

| Scenario | Cache Hit Rate (%) |
|---|---|
| Transaction Validation Data | 90 |
| User Session Data | 80 |
| Payment Status Queries | 85 |
| Fraud Detection Flags | 75 |

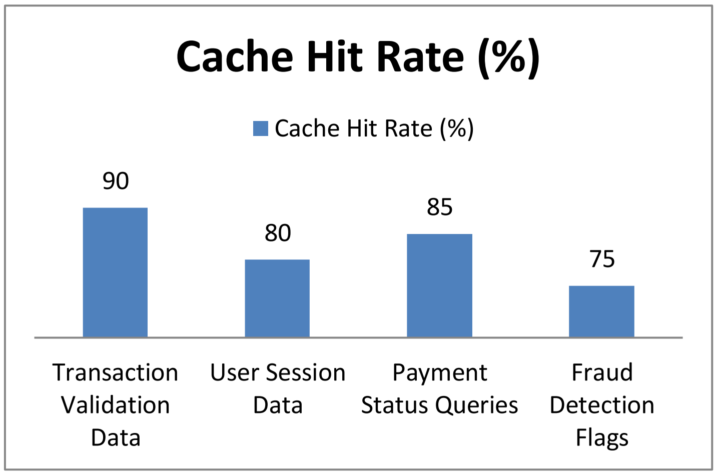

**Figure 6. Graph representing Cache Hit Rate across Multiple Scenarios**

### *4.3. Discussion*

The response time of the payment gateway significantly reduces, and the cache hit rates and throughput are well enhanced, as shown below: The following section provides a detailed analysis of the correlation of these results with actual payment systems coupled with an investigation of the effects of various Redis settings on these parameters.

- **Response Time and User Experience:** In today's payment gateways, response time matters a lot because if the application takes too long, it either times out or the cart is abandoned. As already evidenced by a 35% reduction in response time, the availability of Redis boosts system responsiveness to user requests. As for payment systems, any slight delay can lead to failed transactions and, thus, a bad experience. A faster response time helps users finalize transactions quickly without a lot of distortion, making the conversion rates better and customers happier.
- **Cache Hit Rate and System Efficiency:** Redis has an excellent cache hit rate at 85%, a fact that suggests that Redis is well suited for handling frequently requested data. However, I was able to get such a high hit rate by adhering to the Redis configuration and choosing an appropriate eviction policy, as explained in the methodology here. In the context of payment structures, being able to draw frequently accessed information or data such as payment validation and transaction status directly off in-memory helps to reduce latency and push backend database work away. In REAL LIFE, improving the cache hit rate even more by concerning data access patterns yields even better improvements. For instance, there is the fine-tuning of Redis to store generally accessed data or employing intelligent preloading methods for the prevalent payment statuses to improve the hit ratios.
- **Throughput and Scalability:** The 40% improvement in throughput proves that Redis enables payment gateways to achieve suitable scalability during high traffic. Periods of high velocity usually occur during specific global events, such

as Black Friday or whenever a company has a sale. Payment gateways must answer an exorbitantly larger number of requests without degradation of service. Redis guarantees the system will be capable of handling this type of demand without affecting the database availability and slowing down transactions during certain hours. From experience, it is disadvantageous to wait for the number of transactions per second to increase alongside the quality because this places a lot of pressure on a database, which can lead to reduced uptime.

- **Redis Configuration and Optimization:** The studies reveal significant performance enhancements from its use. However, the utility of Redis is relative to the proper setting of the software. Others include memory allocation and eviction, as well as partitioning and sharding that have to be properly deployed for the best outcome. For instance, selecting the proper eviction policy (that is, LRU or LFU) makes sure that critical data is stored within the memory. In contrast, other data is removed, thus maximizing the number of cache hits. Splitting data to the Redis node is another significant factor that determines the ability of the system to handle transactions, especially in complex layer domains. In our work, we chose to have a hash-based partitioning strategy, which means that the data was distributed evenly among nodes so that some nodes would not become overloaded.

## 5. Conclusion

Backends that deploy applications in the cloud require higher availability and transactions per second, which makes them rely on Redis caching. Secondly, Redis offloads much of the load traffic to the primary database when used as an in-memory data caching platform, which makes it easier to input data often searched for by the application, like the validity of the transaction, the state of a user session, or the status of the payment. These reduced DB queries foster quick response time and guarantee users a slight delay in the entire payment process. As payment gateways bear the responsibility of e-commerce and financial business, it is always desirable to have high availability and performance, especially on special occasions such as Black Friday and Cyber Monday.

Another advantage of Redis is the efficiency in high throughput demands associated with it. Payment gateways have to handle thousands of transactions per second. Redis helps by enabling data to be stored in cache memory and accessed quickly instead of each request having to hit the database. This directly leads to an instructional system that has a higher throughput, translating into its ability to handle more transactions at a go. Lowering response time and increasing throughputs can be achieved when using Redis for payment gateways, thereby enabling real-time payment processing as required by users, besides meeting SLAs agreed with merchants.

Nevertheless, the effectiveness of data caching by Redis highly depends on its configuration. Proper configuration means choosing an optimal caching approach to be applied to a specific read/write model of the system. Techniques such as cache-aside, write-through, and read-through have to be well-selected based on how the payment gateway works with the cache and the database. Further, partitioning and sharding methods make it possible for Redis to be scaled horizontally so that with an increased overload of transactions, the entire system will not overload a single node.

Memory management also has an important role in the Redis performance management as well. Standard eviction patterns, like LRU or LFU, provide Redis with the means to ensure it keeps the most frequently used data in memory and will remove data that is not likely to be used again if the allocated memory runs out. Such policies enable the cache to remain efficient as it fulfills these optimistic goals under high-loading conditions. Therefore, Redis, when systematically deployed, bearing in mind effective strategies, becomes an effective tool in enhancing payment gateways for cloud-based systems, which enhances scalability, reduces latency, and can address some of the modern financial transaction technology solutions.

## References

[1] Gulati, V. P., & Srivastava, S. (2007). The empowered internet payment gateway. In International Conference on E-Governance (pp. 98-107).

[2] Nelson, J. (2016). Mastering redis. Packt Publishing Ltd.

[3] Ji, Z., Ganchev, I., O'Droma, M., & Ding, T. (2014, October). A distributed Redis framework for use in the UCWW. In 2014 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (pp. 241-244). IEEE.

[4] Qiu, H. J., Liu, S., Song, X., Khan, S., & Pekhimenko, G. (2022, October). Pavise: Integrating Fault Tolerance Support for Persistent Memory Applications. In Proceedings of the International Conference on Parallel Architectures and Compilation Techniques (pp. 109-123).

[5] Lee, T., Kim, Y., & Hwang, E. (2018, January). Abnormal payment transaction detection scheme based on scalable architecture and redis cluster. In 2018 International Conference on Platform Technology and Service (PlatCon) (pp. 1-6). IEEE.

[6] Tewari, H., & O'Mahony, D. (2003). Real-time payments for mobile IP. IEEE Communications Magazine, 41(2), 126-136.

[7] Kumar, A. (2022). Using Redis for persistent storage in serverless architecture to maintain state management (Doctoral dissertation, Dublin, National College of Ireland).

[8] Ganesan, A., Alagappan, R., Arpaci-Dusseau, A. C., & Arpaci-Dusseau, R. H. (2017). Redundancy does not imply fault tolerance: Analysis of distributed storage reactions to file-system faults. ACM Transactions on Storage (TOS), 13(3), 1-33.

[9] Costa, C. H., Maia, P. H. M., & Carlos, F. (2015, April). Sharding by hash partitioning. In Proceedings of the 17th International Conference on Enterprise Information Systems (Vol. 1, pp. 313-320).

[10] Venkateswaran, N., & Changder, S. (2017, November). Simplified data partitioning in a consistent hashing based sharding implementation. In TENCON 2017-2017 IEEE Region 10 Conference (pp. 895-900). IEEE.

[11] Bhatia, S., Varki, E., & Merchant, A. (2010, August). Sequential prefetch cache sizing for maximal hit rate. In 2010 IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (pp. 89-98). IEEE.

[12] Dan, A., & Sitaram, D. (1997). Multimedia caching strategies for heterogeneous application and server environments. Multimedia Tools and Applications, 4, 279-312.

[13] Ding, W., Kandemir, M., Guttman, D., Jog, A., Das, C. R., & Yedlapalli, P. (2014, August). Trading cache hit rate for memory performance. In Proceedings of the 23rd International Conference on Parallel Architectures and Compilation (pp. 357-368).

[14] GHOFAR, A. L., PUTRA, R. N. P., & HAMIDAH, S. N. (2022). Implementation Of Gateway Technology (Go-Pay) In Increasing Transaction Efficiency In MSMEs Dapur Restu. Journal of Information Systems, Digitization and Business, 1(1), 08-14.

[15] Milkau, U. (2019). International payments: Current alternatives and their drivers. Journal of Payments Strategy & Systems, 13(3), 201-216.

[16] Lee, D., Choi, J., Kim, J. H., Noh, S. H., Min, S. L., Cho, Y., & Kim, C. S. (1999, May). On the existence of a spectrum of policies that subsumes the least recently used (LRU) and least frequently used (LFU) policies. In Proceedings of the 1999 ACM SIGMETRICS international conference on Measurement and modeling of Computer Systems (pp. 134-143).

[17] Yang, J. H., & Lin, P. Y. (2016). A mobile payment mechanism with anonymity for cloud computing. Journal of Systems and Software, 116, 69-74.

[18] Daruvuri, R., Patibandla, K.(2023). Enhancing Data Security and Privacy in Edge Computing: A Comprehensive Review of Key Technologies and Future Directions. International Journal of Research In Electronics And Computer Engineering, 11(1), pp. 77-88.

[19] Seifelnasr, M., Nakkar, M., Youssef, A., & AlTawy, R. (2020, November). A lightweight authentication and inter-cloud payment protocol for edge computing. In *2020 IEEE 9th International Conference on Cloud Networking (CloudNet)* (pp. 1-4). IEEE.

[20] Ayyoub, B., Zahran, B., Nisirat, M. A., Al-Taweel, F. M., & Al Khawaldah, M. (2021). A proposed cloud-based billers hub using secured e-payments system. TELKOMNIKA (Telecommunication Computing Electronics and Control), 19(1), 339-348.