# Templado System

Empowering Payments with ARM-Based Payment Gateway Technology

Table of Contents

## Abstract

This white paper explores the benefits and advantages of using ARM-based technology in payment gateway systems. We will analyze the performance, security, scalability, and efficiency aspects of ARM architecture and how it can be suitable for payment processing landscape.

## Background

In today's digital age, online payments have become an integral part of businesses across various industries. Payment gateways play a crucial role in securely processing transactions, ensuring seamless customer experiences, and facilitating efficient business operations. This white paper focuses on the background and purpose of utilizing ARM-based technology in payment gateway systems.

Over the past decade, and especially over the past five years, ARM processors have come to dominate the smartphone and tablet landscape, and they had a lot going for them. They offered a lot of performance-per-watt, they were cheap to design, produce and deploy. Processors based on the ARM architecture, an alternative to the mainstream x86 architecture, is gradually making the leap from mobile devices to servers and data centers. ARM servers represent an important shift in server-based computing.

A traditional x86-class server with 12, 16, 24 or more cores delivers increased performance by scaling up the speed and sophistication of each processor, using brute force speed and power to handle demanding computing workloads. By comparison, an ARM server employs many smaller, less sophisticated, low-power processors that share processing tasks across hundreds of processors rather than channeling the workload through just a few processors. The method of increasing performance is sometimes referred to as "scaling out."

All Major Cloud Providers (AWS, Azure, Oracle and Google) offers ARM instances.
Amazon Instance based on Graviton ARM CPU, while the other offering Ampere ARM CPU

The purpose of this white paper is to provide an in-depth understanding of ARM-based payment gateway technology. It aims to educate businesses, payment service providers, and developers about the benefits and implications of adopting ARM architecture in their payment processing infrastructure. By exploring the performance, security, scalability, and cost-effectiveness aspects, this white paper aims to empower decision-makers to make informed choices for their payment gateway systems.

# The Advantages of ARM: From Smartphones to Supercomputers and Beyond

### Higher core count, better performance per watt

While the maximum number of cores in each CPU is not wholly dependent upon the architecture, mainstream x86 CPUs tend to house a smaller number of larger, more complex cores. ARM CPUs compete by having more cores inside a single processor; for example, the Ampere® Altra® Max series can contain up to 128 cores in each CPU. The workload is distributed to a large number of smaller, more efficient cores, rather than having a few powerful cores handle every task. As a result, the cores of an ARM processor can offer better performance per watt of power.

### Lower TCO, better performance per dollar

ARM servers can indeed offer better cost per dollar compared to traditional x86 servers.
ARM CPUs are known for their energy efficiency. They typically consume less power compared to x86 CPUs, resulting in lower electricity costs over time. This is especially beneficial in server rooms or data centers where power consumption is a significant expense. The reduced power consumption also leads to lower cooling requirements, further contributing to cost savings.
ARM processors are generally cheaper to design and manufacture compared to x86 processors. The licensing model of ARM architecture allows multiple companies to design and produce ARM-based chips, increasing competition and driving down costs. This cost advantage can make ARM servers more affordable, especially in large-scale deployments.
ARM servers offer scalability and flexibility, making them suitable for various workloads and applications. They are commonly used in cloud computing, web hosting, and edge computing environments. The ability to optimize hardware configurations and leverage ARM's architecture for specific workloads can result in better performance per dollar compared to x86 servers.

### Cloud-nativity

Price for performance is a deciding factor for companies that choose Arm, and for good reason. Arm is just as efficient as x86 for cloud computing while still offering attractive prices.
Organizations can save money while processing a broad spectrum of workloads, truly putting Arm's performance into competition with x86 architectures. In this area, its rivals simply cannot surpass Arm's price-to-performance ratio. AWS Graviton processor, Amazon's first AWS Arm offering. Arm can save AWS users as much as 40 percent in costs for performance equal to that of Amazon's previous x86 cloud processors.

### Reduced on-premise infrastructure costs.

The infrastructure costs associated with payment gateway systems can be a significant expense for businesses. ARM-based payment gateways offer cost-effective solutions by minimizing infrastructure requirements. Upfront and ongoing infrastructure costs can be brought down by as much as 33 percent per core, which has a positive trickle-down effect on the cost of cloud-based offerings.

### Performance

ARM servers can offer excellent performance for certain workloads, particularly those that are highly parallelizable or require high memory bandwidth. ARM processors have built-in hardware security features, such as Trust Zone technology, which can enhance the security of payment processing systems by protecting against malware and other security threats.

## The Reference Architecture

Templado System Payment platform as shown in Figure 1. The platform can be deployed on-premises or in the cloud. As a cloud-native platform it can run on AWS Graviton-2 and Graviton-3 A RM CPU. For Azure cloud it can run on Ampere Ultra ARM CPU.

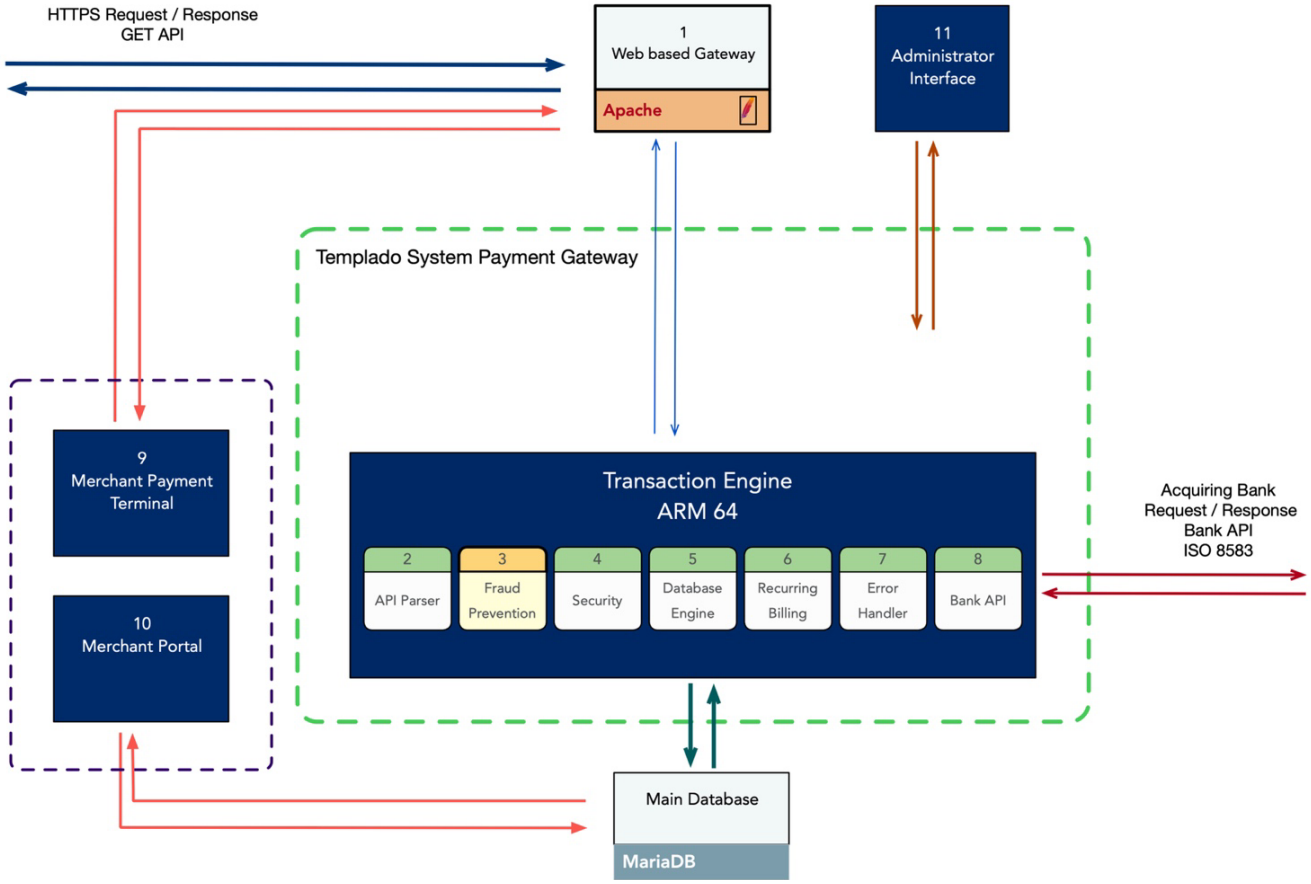



Figure 1. Payment platform reference architecture

The goal was to design the system free from any third-party licenses such as Microsoft Server, or SQL Server. As a result, the system uses all open-source platform. The system uses Linux AARCH64 platform as a server (Ubuntu, Fedora ARM) and MariaDB / MySQL as its main database. For security system utilized OpenSSL to perform necessary cryptographic functions such as SSL API, AES encryption to protect customer data and key management.

Customer are free to use any other Key management to protect data and can use Amazon KMS function or Virtual HSM in case of cloud-base installation, or real HSM Modules (such as Talos, Futurex) in the case of on-premises installation.

# Platform Components

Is where the key payment processing functions resides. It includes payment gateway, transaction engine, data storage and merchant related modules.

### Gateway

Web gateway implemented on Apache web server. Using Apache as the web server for your payment gateway offers several advantages. Apache is an open-source web server, which means it is freely available to use. This can be particularly advantageous for small businesses or startups with limited budgets, as it helps reduce infrastructure costs. It is compatible with multiple operating systems, including Linux, Unix, Windows, and macOS. This flexibility allows you to deploy your payment gateway on a wide range of platforms, depending on your specific requirements. Apache provides a range of security features to protect your payment gateway. It supports SSL/TLS encryption, allowing you to secure the communication between clients and your server. Additionally, Apache offers numerous security modules and configurations to help mitigate common web server vulnerabilities. Yes, Apache HTTP Server can be run on ARM-based systems. The Apache software is designed to be cross-platform and is compatible with various operating systems, including those running on ARM architectures.

The purpose of an Apache gateway, often referred to as an API gateway, is to act as an intermediary between clients and backend services. It serves as a central entry point for all incoming requests and provides various functionalities to manage, secure, and optimize API communication. When it comes to receiving HTTPS transactions, an Apache gateway can serve as a reverse proxy that accepts incoming HTTPS requests from clients. It acts as the initial point of contact for these requests, providing a secure endpoint for clients to connect to. Once gateway received API Request, it forwards the processed request to the Transaction Engine for further handling or processing.

### Transaction Engine

Transaction engine is responsible to authenticate transaction, verifies merchant, fraud detection, selecting appropriate acquiring partner bank and select network protocol / API for bank authentication. Transaction engine submits request to acquiring bank and upon receiving response provide result to the merchant using merchant's API. Additional, transaction engine performs all necessary database functions.

To fully take advantage of ARM Architecture, transaction engine designed and implemented as low lever program combine C / C++ and Assembler languages. It's implemented to be scalable and take advantages of multi-cores ARM architecture. Transaction engine can run on 4 core ARM Cortex 70xx and up to 128 Cores Ampere Ultra CPU.

Transaction Engine has a thread pool with 20 threads available and if the CPU has more than 20 cores, there are a few possibilities for optimizing thread utilization and achieving more efficient processing:

Thread per Core (1:1): In this approach, each available core is assigned a single thread from the thread pool. This maximizes parallelism, allowing each core to execute a thread independently. If there are more cores than threads, some cores may remain idle.

Multi-threading: The transaction engine is designed to support multi-threading, it can spawn multiple threads per core to fully utilize the available resources. For example, if the CPU has 40

cores, the engine can spawn additional threads, resulting in a total of 40 threads and so on. Ampere Ultra ARM CPU has family of 32, 64, 128 cores. This approach can further enhance parallelism and take advantage of the additional computational power.

**API Parser (2)**
An API parser module is responsible for parsing and extracting relevant information from incoming API requests.

**Fraud Prevention (3)**
A payment fraud prevention module is a component designed to detect and prevent fraudulent activities related to payment transactions. Its primary purpose is to identify and mitigate fraudulent behavior. The module utilizes a set of predefined rules and logic to detect specific fraud patterns or known fraud indicators. These rules include velocity checks, IP blacklisting, address verification, and card verification value (CVV) validation.

**Security (4)**
OpenSSL was used to perform several security tasks, such as Hash, AES encryption / decryption and SSL / TLC transport.

**Database Engine (5)**
Transaction Engine uses C++ Library to communicate with database. Each thread has its own SQL connection to parallel database request.

**Recurring Billing (6)**
Templado payment gateway has capability to perform recurring billing based on merchant request. Such capability serves the subscription-based merchants.

**Error Handler (7)**
Error handle module responsible to handle situation such as Timed Out transaction response, and any other errors on the system related to incorrect API, failed merchant verification, failed fraud result, etc.

**Acquiring Bank communication (8)**
Acquiring partner bank API was incorporated directly to the engine. Since there is not standard on acquiring bank API, like ISO8583, we decided to incorporate API directly to transaction engine. This gives additional advantage on speed, but if platform change the partner bank it would require to change API inside the transaction engine. Some other payment gateway creates additional layer, responsible to bank API, that will not require to change / recompile the engine, we decided to keep it simple and fast.

### Merchant Portal

A merchant portal for a payment gateway is a web-based application that allows merchants to manage their payment processing activities. It provides a centralized platform for merchants to view and manage transaction data, track sales and revenue, and perform other important tasks related to payment processing.

### Payment Terminal

A web-based payment terminal is a payment processing solution that is accessible through a web browser. This type of payment terminal allows merchants to accept online payments without the need for a physical point of sale (POS) system. With a web-based payment terminal, customers can make payments by entering their payment information directly into a secure web form. The payment information is then processed by the payment gateway.

### Admin Interface

A payment processor administrative interface is a web-based application that allows payment processing companies to manage their payment processing activities. It provides a centralized platform for payment processors to monitor and manage payment transactions, configure payment settings, add merchants, and perform other important tasks related to payment processing

### Note on standard

While banking system operate under international standard (ISO8583), for online payment processor there is no such thing as standard. Every payment processor implements their own API, delivery transport, and security. During last 20 years of booming web commerce, VISA and MasterCard did not introduce any standard API for processing online transactions. The only requirement for online payment processing is be certified under PCI authority. Not having standard, creates possibility for payment processing to develop their own API based on their internal decisions that suited their goals.

## Merchant Model

Templado System platform support the following merchant models:

### Direct Merchant

The Direct Merchant ID (DMID) model is a payment processing model in which a merchant has a direct relationship with a payment processor. Under the DMID model, the merchant is typically required to obtain their own merchant account with a payment processor. The merchant account is linked to the DMID, which serves as a unique identifier for the merchant when processing transactions and settling funds.

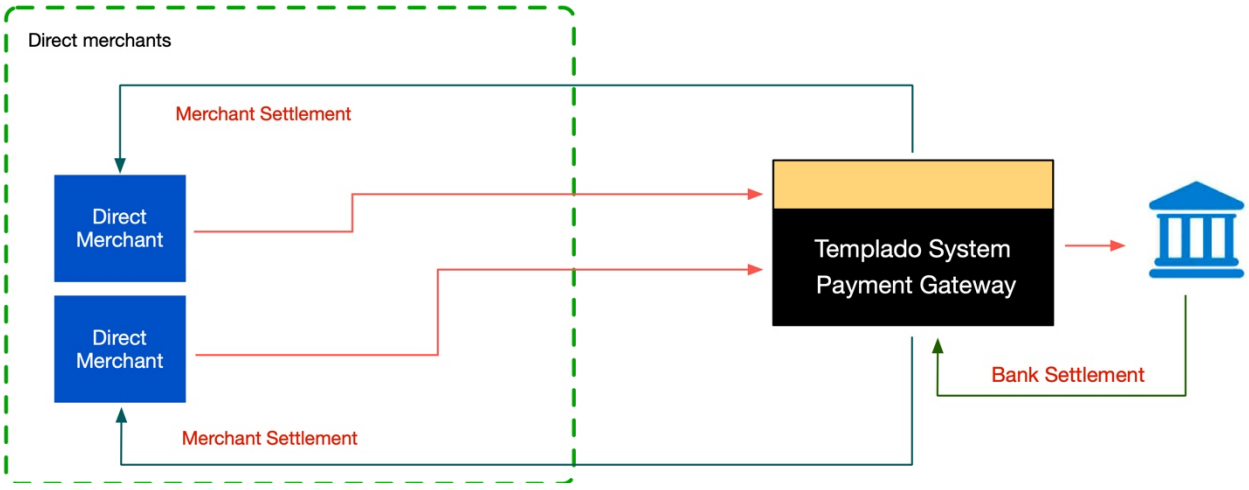



Figure 2. Direct Merchant Model

## Aggregation Merchant

The merchant aggregation model is a business model in which a company acts as an intermediary between multiple merchants and payment processor. A Master Merchant ID (MMID) is a unique identifier assigned to an aggregator and all of its sub-merchants or merchants, who are working under the aggregator's umbrella. The MMID is used by the aggregator to manage the payment processing for all of its sub-merchants. When a customer makes a payment to one of the sub-merchants, the payment is collected by the aggregator, who then disburses the funds to the appropriate sub-merchant account.

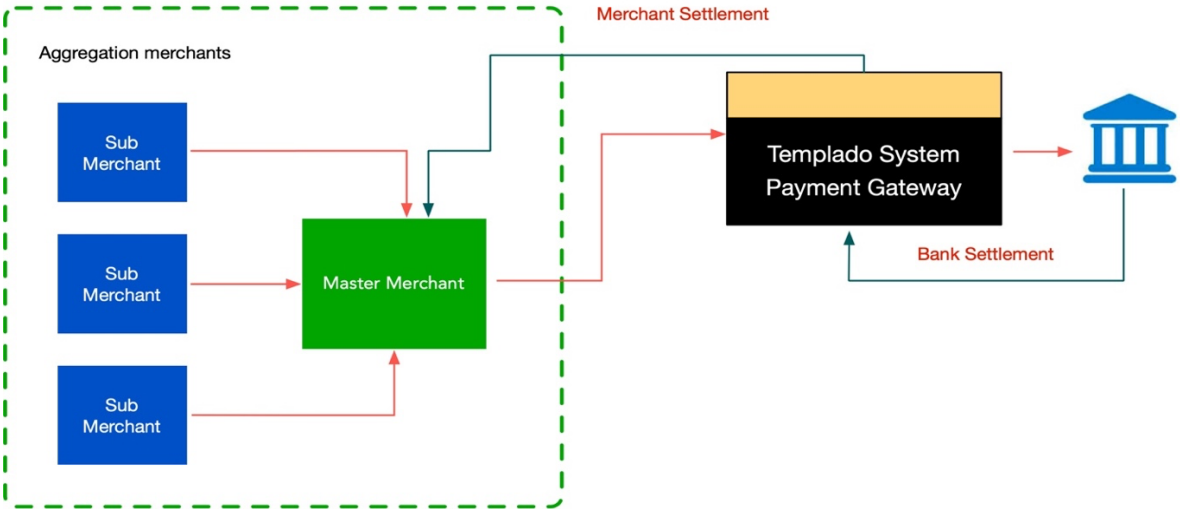



Figure 3. Aggregation Merchant Model

### Hybrid Merchant

In Hybrid model, Templado System allows to mix direct merchants and aggregation merchants

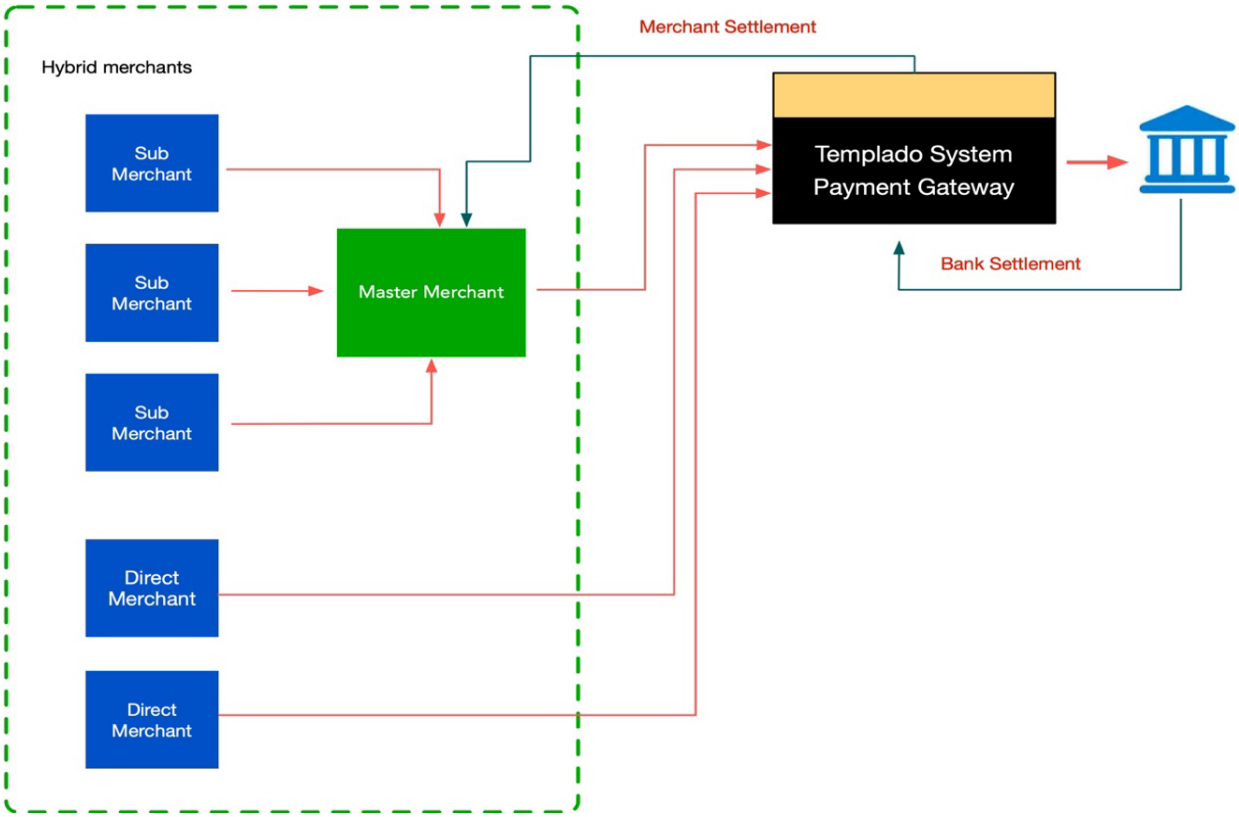



Figure 4. Hybrid Merchant Model

# Testing

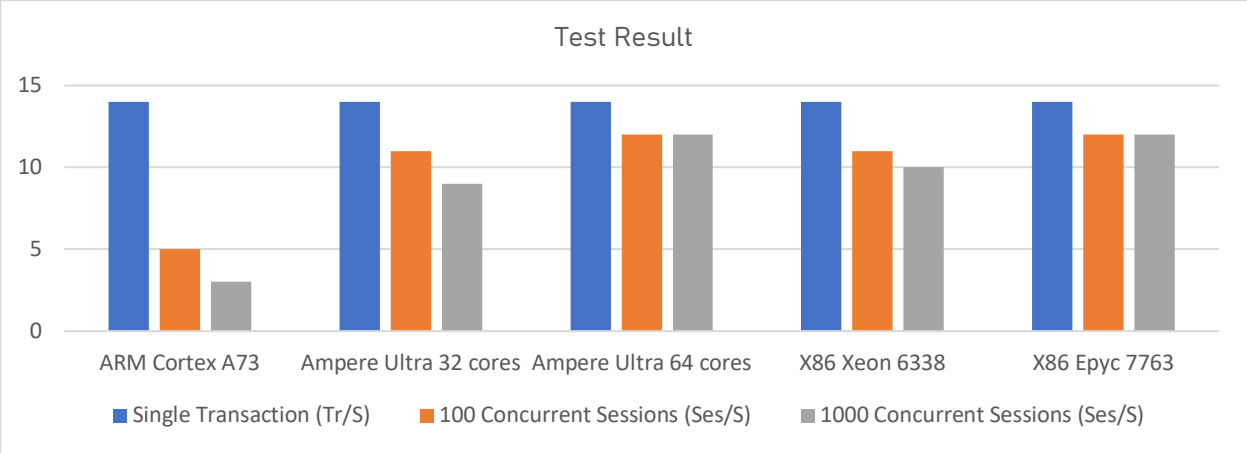
### On premises systems

1. ARM Cortex A73 CPU, 8GB RAM
2. Ampere Ultra Ampere Altra 32 Core, 32GB RAM, Samsung 970 NVMe, Ubuntu 22.04 Server
3. Ampere Ultra Ampere Altra 64 Core, 32GB RAM, Samsung 970 NVMe, Ubuntu 22.04 Server
4. Dell Power Edge 650. Intel Xeon 6338, 32GB RAM, SATA SSD RAID 5, Microsoft 2019 Server
5. EPYC 7763, 32GB RAM, SATA SSD RAID 5, Microsoft 2019 Server6

We use Ampere Ultra developed workstation from Adlink technology.
https://www.ipi.wiki/products/ampere-altra-developer-platform?variant=41995037868194

Cost:

1. Ampere Ultra 32 Core $3,250
2. Ampere Ultra 64 Core $4,200
3. Intel Xeon $6,029
4. AMD Epyc $7,890




Single Session / Single transaction takes 64ms to completed on any system. Gateway take around 64ms to process single transaction. All systems perform the same.

Next, we simulate 100 Concurrent Sessions from Apache passing transaction to Transaction Engine While Ampere 32 core perform on pair with Intel Xeon, but slower than AMD Epyc.
Ampere 64 core slightly outperform Intel Xeon and pretty much the same as AMD Epyc.

Finally, we simulate 1000 Concurrent Sessions from Apache passing transaction to Transaction Engine. This time Ampere 32 slower than both Xeon and AMD, but Ampere 64 clearly beat Intel Xeon and match the AMD Epyc.

Ampere 128-core ARM-based server was not available for testing in out scenario, as well as new Ampere One (192 Cores).

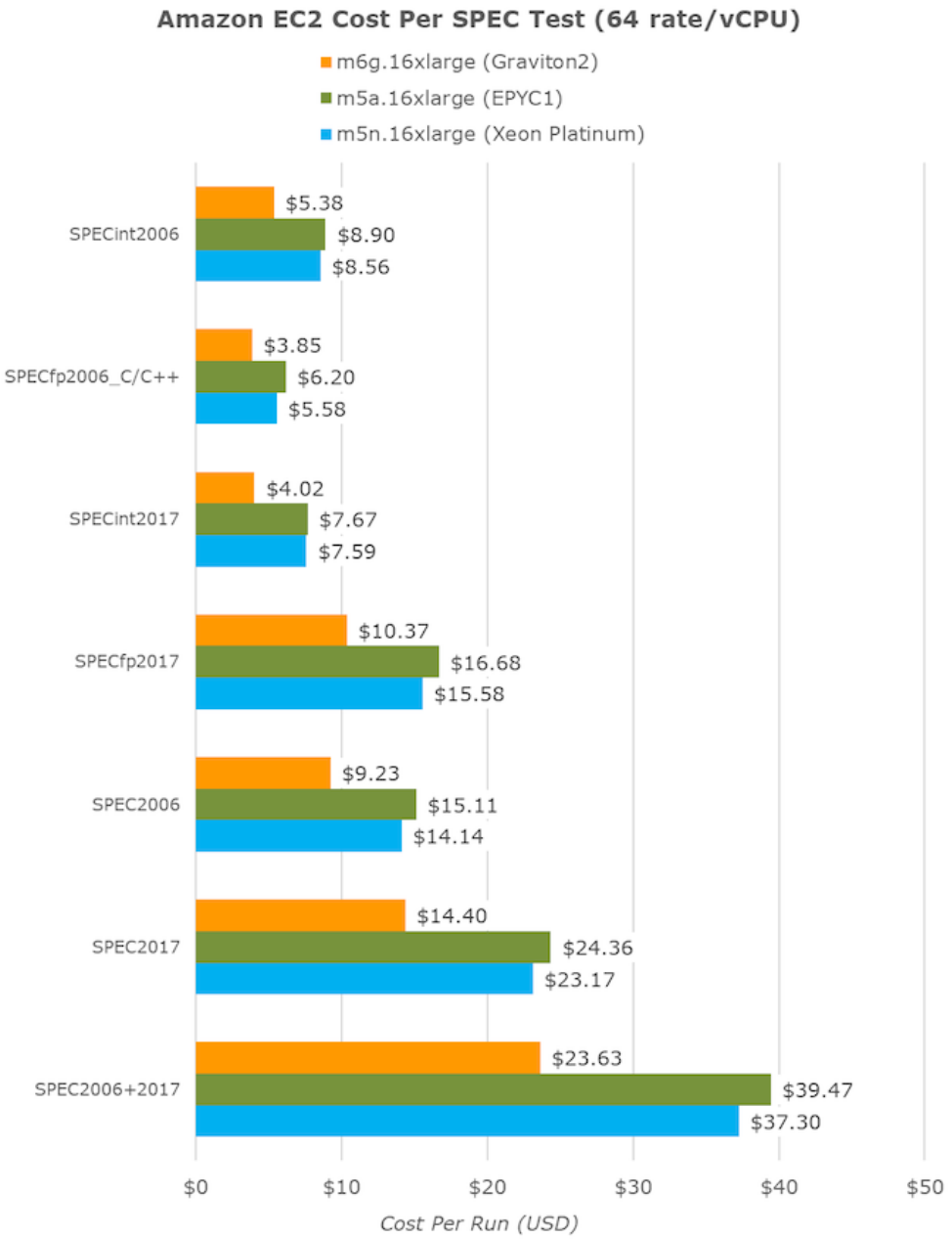
### Amazon Graviton

AWS Graviton processors are designed by AWS to deliver the best price performance for your cloud workloads running in Amazon EC2. Since Amazon does not provide Graviton CPU to build the server, the test was conducted using AWS Instance. For the purpose of the testing the minimum Graviton configuration was chosen. EC2 t4g.small with only 2 cores and 1GB of memory.
Instance were capable to run Ubuntu 22.04 with Apache-2 Web, where engine receive real live external transaction to simulate real live merchant experience. Engine was connected to Maria DB, running on different instance, for the purpose of simulating separation database server from engine server.

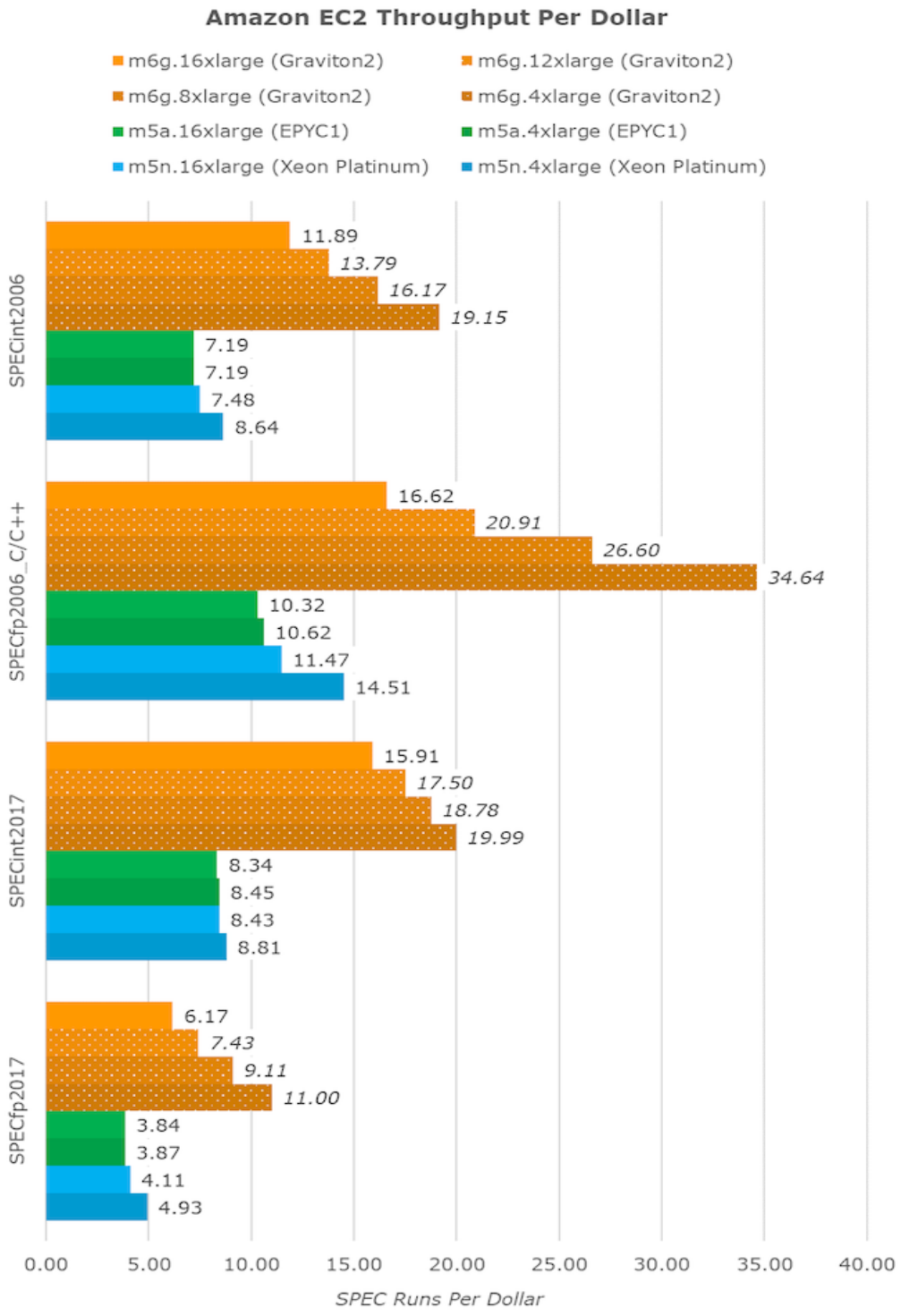
Graviton-2 result
8 trans/seconds on EC2 t4g.small ARM configuration.

## ARM On Cloud

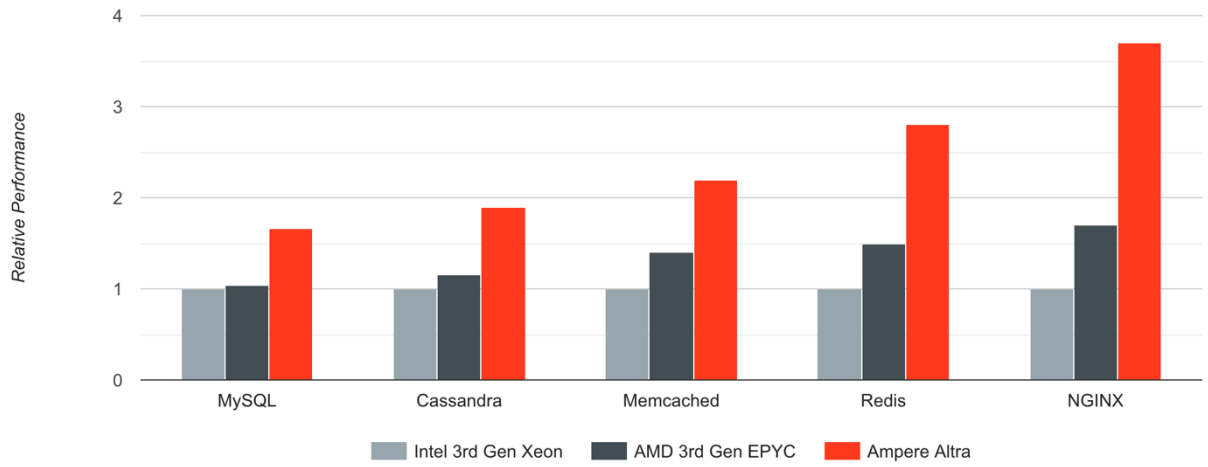
All Major Cloud Providers (AWS, Azure, Oracle and Google) offers ARM instances.
Amazon Instance based on Graviton ARM CPU, while the other offering Ampere ARM CPU



The Graviton2 showcased that it can keep up extremely well in terms of performance and throughput, even beating the competition in a lot of the tests. However sometimes you don't care too much about performance, and you just want to get some workload completed in the cheapest way possible, at which point value comes into play.



Amazon does allude to that, stating that the new chip is able to achieve 40% better performance per dollar than its competition. As covered in the introduction, for the 64-vCPU count 16xlarge instances the m6g (Graviton2), m5a (EPYC1), and m5n (Xeon Cascade Lake) are priced at an hourly cost of $2.464, $2.752 and $3.808 respectively.
An aggregate of all workloads summed up together, which should hopefully end up in a representative figure for a wide variety of real-world use-cases, we do end up seeing the Graviton2 coming in 40% cheaper than the competing platforms, an outstanding figure.



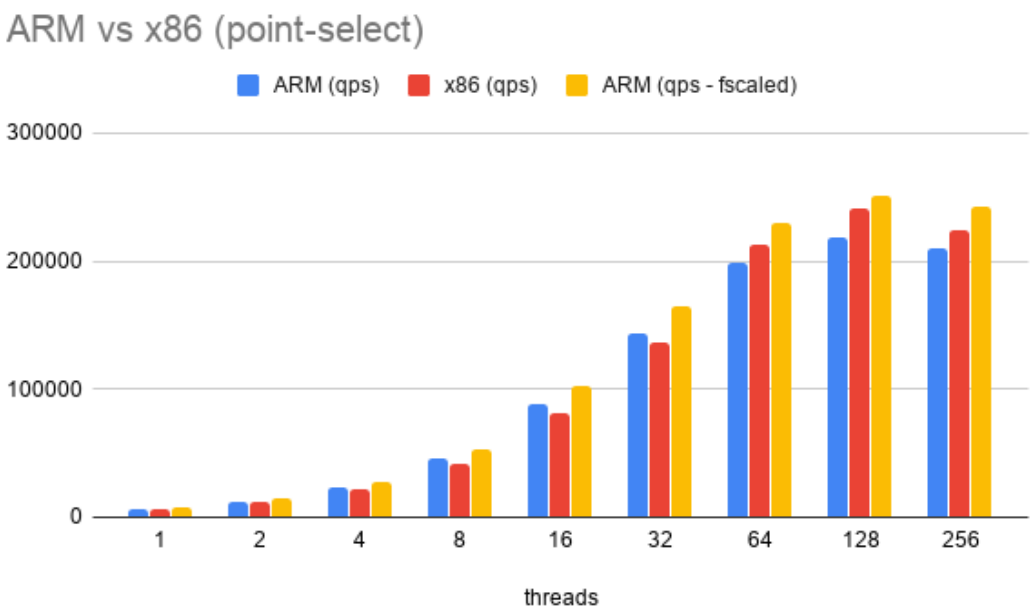

*Comparisons between the major legacy X86 architectures*
*and the Ampere Altra Max Cloud Native Processor running on five of the most popular cloud native workloads.*

## Database on ARM

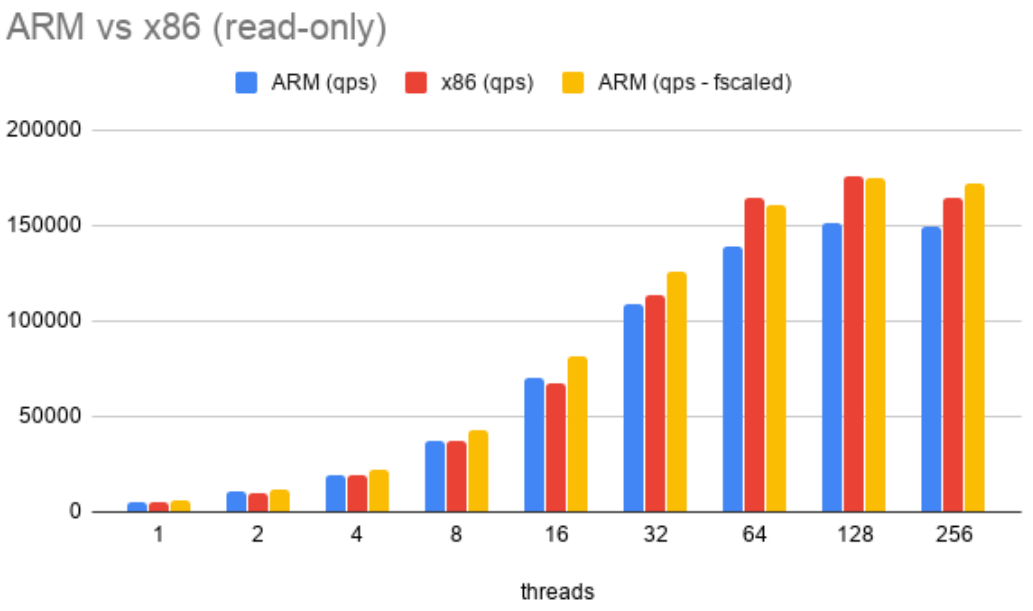
Platform uses MariaDB as main database. MariaDB is designed for high performance and can handle large volumes of data and transactions efficiently. This makes it a good choice for payment processing applications that require fast and reliable performance. MariaDB supports features such as replication and clustering, which can help ensure high availability and uptime for payment processing applications. This can help minimize downtime and ensure that payment processing is always available to customers. It is open source and free to use, which can help reduce the cost of payment processing for businesses. MariaDB includes built-in security features such as SSL/TLS encryption, secure password storage, and access controls. This can help ensure that sensitive payment data is protected from unauthorized access and theft.

Performance
24 vCPU/48 GB Intel(R) Xeon(R) Gold 6266C CPU @ 3.00GHz for running MySQL on x86.
24 vCPU/48 GB ARM @ 2.60GHz for running MySQL on ARM

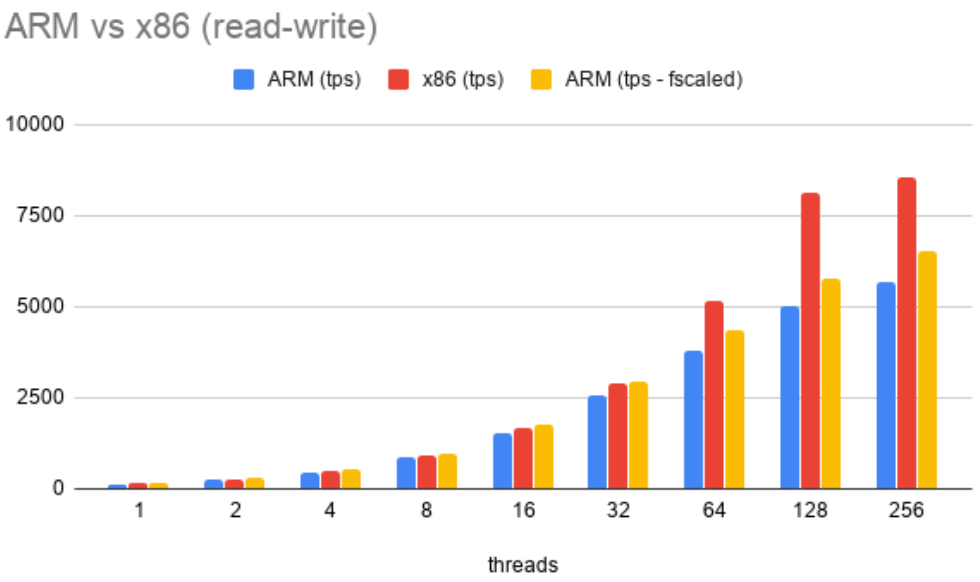
Point Select



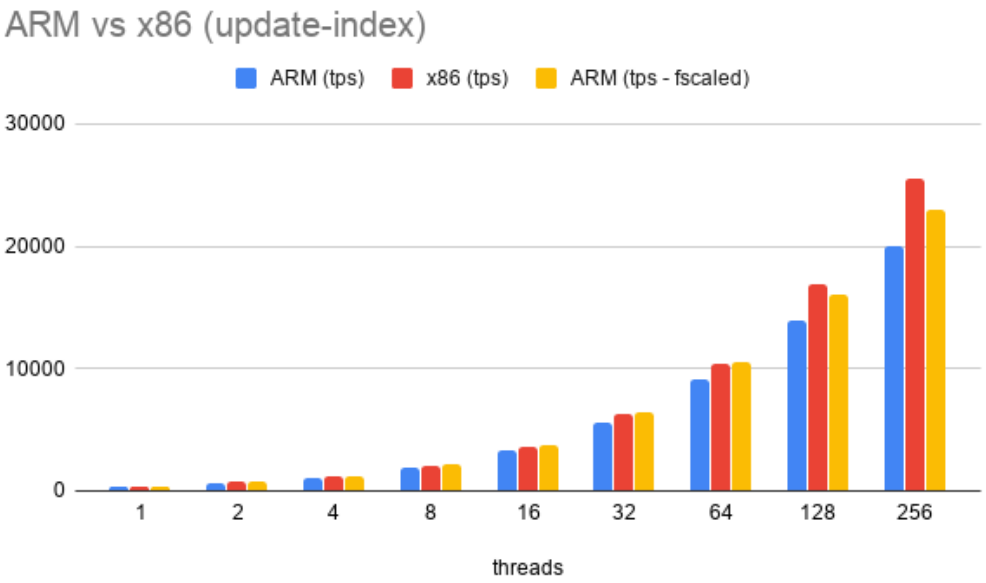

Read Only



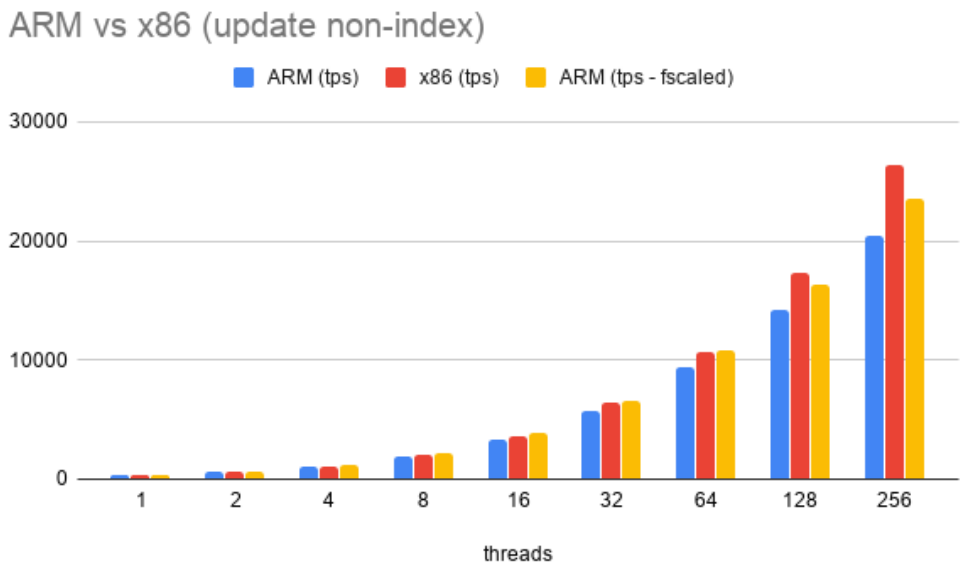
Read Write




Update Index



Update Non-Index




## Conclusion

There are some important observations
For read only workload MySQL on ARM continues to perform on-par with MySQL on x86.
For write involving workload MySQL on ARM starts lagging a bit but if we consider frequency scaling things start getting better.
Frequency scaling is not a real-life parameter so we should consider the price-per-performance ratio. This could be a topic in itself but just a quick fact: ARM instance is 66% cheaper than x86 (24U48G same one we used).

There is a pattern that we can observe. ARM workloads are very well scalable till it hits the CPU limits. With increasing scalability, contention increases and ARM starts lagging. This is expected since mutexes / contention hot-spots were all tuned for x86 (for example: spin-lock). But now that MySQL officially supports ARM and the growing ARM community and interest, it would be tuned for ARM too.

To summarize, MySQL on ARM is a worth exploring option from a cost and performance perspective.

The cost-effectiveness and resource optimization advantages of ARM-based payment gateways make them a compelling choice for businesses seeking to maximize their return on investment. By adopting ARM architecture, organizations can reduce infrastructure costs, lower energy consumption, and optimize resource utilization, leading to significant cost savings. The ARM server ecosystem has been growing steadily, with an increasing number of software vendors and operating systems providing support for ARM-based servers. This expanding ecosystem ensures compatibility and access to a wide range of software, further enhancing the value proposition of ARM servers.

This white paper has highlighted the benefits of ARM-based technology in achieving cost-effectiveness and resource optimization in payment gateway systems.

# About Templado System

The Templado Payment Platform is a payment platform built on ARM architecture.
It is designed for data processing environments that require the highest throughput and lowest latency.

Templado System is headquartered in 80 Corporate drive Suite 309, Scarborough, Ontario M1H 3GH Canada. To learn more about Templado System visit https://templadosystem.com or contact us at info@templadosystem.com to get more information.