Success Story.
DEVELOPING HIGHLY SCALABLE PAYMENT PROCESSING ENGINE LEVERAGING THE CLOUD

About the Client

The client provides payment processing services for merchants and businesses including ACH / eCheck processing, Credit card processing, recurring billing etc. The client also provides tools and API's for developers to easily integrate ACH and eCheck payment processing functionality directly into their software, website, or application.

Business Challenge

The client's existing payment processing platform was built on an older technology stack and had issues in scalability and performance. There were also more serious reliability issues where transactions got lost within the system, subjecting the client to 3rd party audits and inquiries.

The customer base and transaction volumes have increased through a recent acquisition. There was added pressure to develop a brand-new platform that could meet customer expectations.
The client chose Silicus to develop the payment processing platform based on our expertise in developing mission critical modern applications, combined with our domain expertise in payments processing.

Silicus developed a payment processing system leveraging modern application architectures developed and deployed natively in the Cloud.

Web Portal

This was the application front end comprised of several sub-modules for each service offering - ACH processing, Credit card processing, Returns processing, Billing, APIs for integrations etc. The bulk of the business logic resided in a combination of the web portal and backend processing. Merchants and sellers could manage their accounts and view their transactions through the web portal. The traffic and load on each module varied depending on customer requirements. As such, there was a need to optimize the compute power for each module depending on use, rather than have the same compute for all modules.

The key sub-modules within the portal included:

- Web Interface – administration, management, reporting, processing & diagnostics

Each functional sub-module was developed and deployed within containers, with the flexibility to scale each depending on user traffic. Azure Kubernetes Service (AKS) was used for managing and orchestrating the containers. The web portal was developed in PHP.
Transaction Ingestion

This was a real time engine that had to be always available with low latency operations. The service was made available via a web API endpoint that could integrate into multiple front-end systems. The module was architected to ingest 120,000 transactions a day, with the capability to scale 100x in 5 years. Additionally, the ingestion engine had to be extremely elastic to handle burst traffic loads during peak holiday shopping (e.g. Black Friday) without having to invest in upfront infrastructure capacity.

Silicus evaluated two options to architect this engine – Container based architecture vs serverless computing architecture. While containers allowed the flexibility to scale individual modules, these were still within the boundaries of the compute level subscriptions that were purchased. Any compute requirements that went beyond the subscription would develop into a bottleneck.

Azure Functions is a fully managed serverless compute platform that can scale on demand with high reliability and security. This helped the customer to pay only for the additional computing capacity it needed during peak traffic. Silicus recommended Azure Functions as the right approach for seamless scalability and elasticity required by the transaction ingestion engine. Each Azure Function instance could also be deployed across multiple data centres around the world for global geo-redundancy.

Transaction Processing & Storage

This was the transaction processing engine with operations in batch mode – each batch needed to get processes in approx. 2 hours. It was designed for scale and extremely high volumes. The key sub-modules included:
• Processor Facing – ACH processing all Fed windows, CC processing, returns processing, processor integration

• Data – Merchant data, diagnostics, reporting, processor submission data

By deploying the processing module within Azure batch service, it would allow the batch service to assess the batch size and automatically spin up the required VMs needed to process the batch within the 2-hr window. The VMs would be terminated once the batch was executed. This resulted in highly compute power streamlined and optimized processing.

The data storage component was tightly coupled with the transaction processing engine.

CosmosDB was chosen for storing the ingested transaction data in NoSQL format, with the capability to handle high volumes and provide out of the box geo-redundancy. Multiple CosmosDB instances were stood up across geographies to ensure a truly high-availability solution. The Azure traffic manager service would automatically route transactions to the appropriate instance. Each CosmosDB instance would sync with each other instance to ensure a single point of truth.

Once the transactions were processed, a separate master CosmosDB instance would store the information. All configurations and user management data were stored in an Azure SQL relational database.

Security & PCI Compliance

Silicus architected the solution with comprehensive and modern security using Azure key vault for encryption and secure key management, next-gen firewalls, web application firewalls (WAF), log auditing with Splunk, and application monitoring leveraging Azure monitor. Azure AD B2C was
used for authentication and identity management.

The application had to be PCI compliant since it would be handling credit card related information. Silicus leveraged Azure PaaS Services to achieve PCI compliance. Since PaaS components are locked down at the system level, the Azure PCI certification was leveraged out of the box.

**DevOps**

The application was developed to run in a full DevOps driven delivery model. This included continuous delivery and deployment, along with round the clock application support. Silicus built out the entire DevOps toolchain and processes to facilitate this. This included:

- Continuous Integration
- Continuous & Automated Testing
- Continuous Delivery
- Automated deployment management
- Release management orchestration

**Modern Architecture**

Silicus leveraged the latest application architecture paradigms – Containers, serverless less computing and Cloud based “pay-for-use” compute to create a truly agile, scalable, high performance and resilient application.
Technologies Used

- PROGRAMMING: PHP, Java
- STORAGE: CosmosDB, Azure SQL
- CLOUD SERVICES: Azure Functions, Azure Batch Services
- CONTAINER MANAGEMENT: Docker, Azure Kubernetes Services (AKS)
- SECURITY: Azure Key vault, Azure monitor, WAF, Splunk, Azure AD
- DEVOPS TOOLCHAINS: Splunk, JMeter, Bitbucket

Client Benefits

COST EFFICIENT OPERATIONS
Traditionally payment processing was a compute and storage intensive business. By leveraging modern architectural design concepts and services such as Azure Functions, Azure batch service and Azure PaaS, Silicus was able to deliver a payment processing platform that could offer scalability from small scale to internet scale operations at a fraction of the operational and CAPEX required.

TRANSACTION PROCESSING SPEED
Silicus architected a superior and high-speed payment processing system with built-in geo-redundancy and low latency through instance deployments across multiple Azure data centres. Intelligent traffic routing and load balancing ensured that payments would be processed at a data centre that was nearest to the payment location.