

Omnichannel Transaction Collector

Arquitectura de Despliegue

Índice de Contenidos

[Índice de Contenidos 2](#_Toc501460657)

[Introducción 3](#_Toc501460658)

[Arquitectura de Despliegue 3](#_Toc501460659)

Introducción

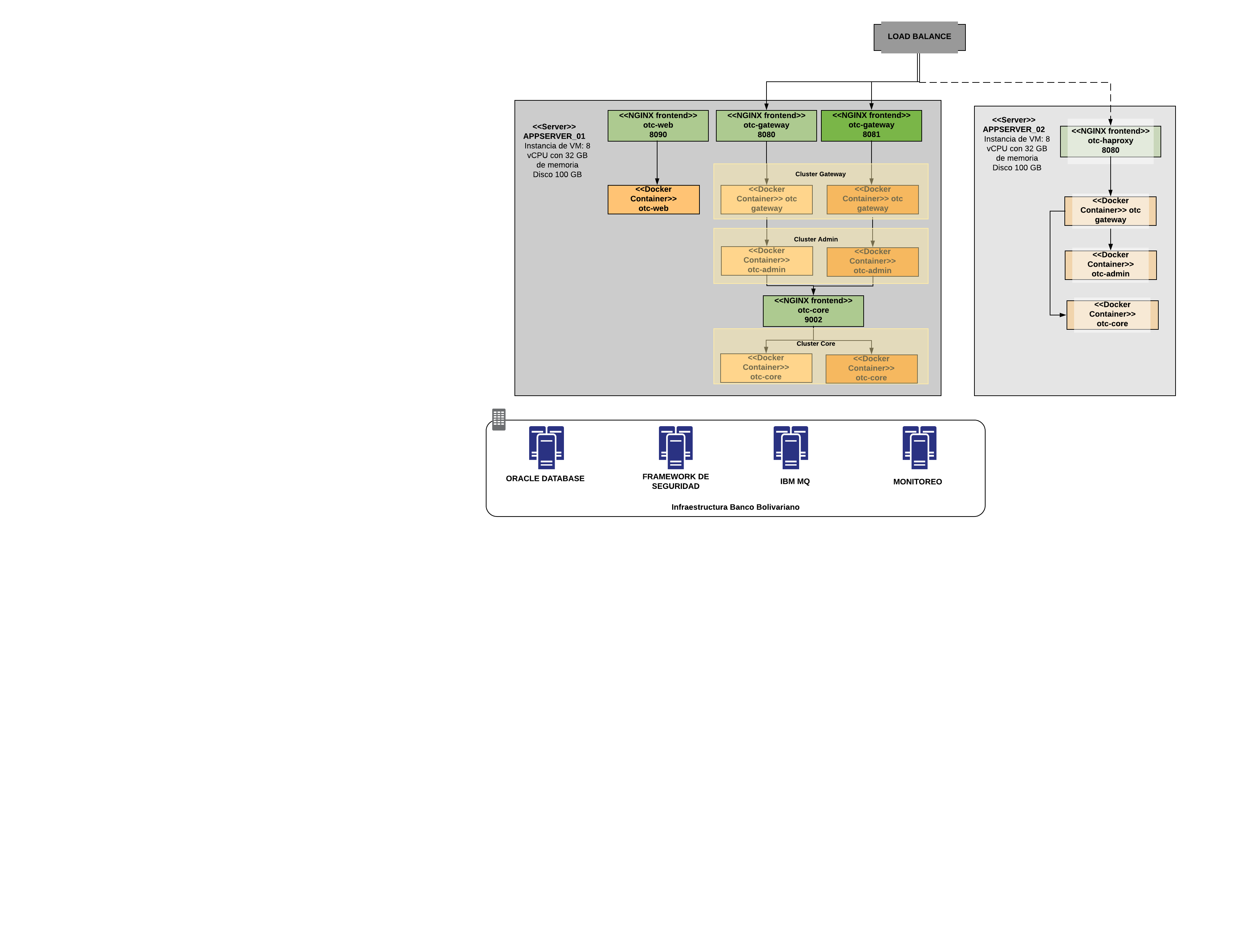
El Banco Bolivariano se encuentra en la implementación del proyecto Cash Management, para el cual ha adquirido una solución tecnología que le permite brindar los servicios de Cash Management a sus clientes por medio de la Banca en línea. Este producto permite consultar y ejecutar servicios de recaudo expuesto con por el Banco, sin embargo, requiere de un servicio (SOAP/XML con un contrato definido) que centralice todos los servicios de recaudos del Banco.

El Banco Bolivariano requiere que el servicio de recaudo centralizado pueda ser reutilizado por otros canales (omnicanal), adicional que sea administrable.

Omnichannel Transaction Collector (OTC) permite centralizar la administración y ejecución de los servicios de recaudo del Banco, desde una sola interfaz o API por medio de acceso. Los servicios de recaudaciones son configurados por empresa y tipo de recaudación (ejemplo: recaudación de Agua), y habilitados por canal (ejemplo: Banca en línea); cada canal este asociado a un medio de acceso (ejemplo: soap/xml), sobre el cual se expone un API con las operaciones de consulta, pago y reverso; cada una de las operaciones representan un flujo de ejecución que puede ser parametrizado para invocar los diferentes servicios de recaudo expuesto por el Banco.

Arquitectura de Despliegue

El siguiente grafico describe la arquitectura propuesta para el despliegue de los componentes de TI bajo un enfoque de arquitectura de microservices.



Los servidores propuestos son los siguientes:

* **APPSERVER\_01**: servidor de cluster microservices.
  + **Caratectisticas**: Instancia de VM: 8 vCPU con 32 GB de memoria Disco 100 GB.
  + **Requerimiento:** 
    - Se requiere Docker, Docker Compose, Docker Swarm.
    - Todo los componentes se instalaran como contenedor Docker. Los componentes (y productos) son:
      * Servidor http: NGINX Server
      * OTC Web: Microservicio Java Spring Boot/Angular
      * OTC Gateway: Microservicio Java Spring Boot/Apache Camel
      * OTC Admin: Microservicio Java Spring Boot
      * OTC Core: Microservicio Java Spring Boot/Apache Camel. Principal componente de la solución. Ocupa el 70% de recursos aisgnados.
      * Cache: Redis
    - Se recomiendo Internet en su instalación.
    - Se recomienda tener un filesystem separado por cada micro-servicio. La distribucion seria la siguiente:
      * /otc 60GB (incluye BDD)
      * /logs 20GB
      * /cache 10GB
      * / 10GB
* **APPSERVER\_02 (SERVIDOR BAJO DEMANDA)**: servidor de cluster microservices.
  + **Caratectisticas**: Instancia de VM: 8 vCPU con 32 GB de memoria Disco 100 GB.
  + **Requerimiento:** 
    - Se requiere Docker, Docker Compose, Docker Swarm.
    - Todo los componentes se instalaran como contenedor Docker. Los componentes (y productos) son:
      * Servidor http: NGINX Server
      * OTC Web: Microservicio Java Spring Boot/Angular
      * OTC Gateway: Microservicio Java Spring Boot/Apache Camel
      * OTC Admin: Microservicio Java Spring Boot
      * OTC Core: Microservicio Java Spring Boot/Apache Camel
      * Cache: Redis
    - Se recomiendo Internet en su instalación.
    - Se recomienda tener un filesystem separado por cada micro-servicio. La distribucion seria la siguiente:
      * /otc 60GB (incluye BDD)
      * /logs 20GB
      * /cache 10GB
      * / 10GB

**Observaciones:**

* Para el dimencionamiento se considero una carga de 100TPM para los servicios expuestos. Sin embargo, se debe considerar como un estimado ya que la solución no se ha sometido a pruebas de carga.
* Los considera que los servidores de monitoreo y administración ya estan disponibilizados en el Banco, por lo cual no se los incluye en la arquitectura.
* Se definio el minimo de requerimiento de hardware para las herramientas de continuidad, y administracion, con el fin de ir redimiencionando, de acuerdo su uso.
* En un ambiente cloud, se podria iniciar con la mitad de recursos de almacenamiento definido e ir redimencionando de acuerdo a su utilización.
* El ambiente definido proporciona caracteristicas HA (Alta Disponibilidad).