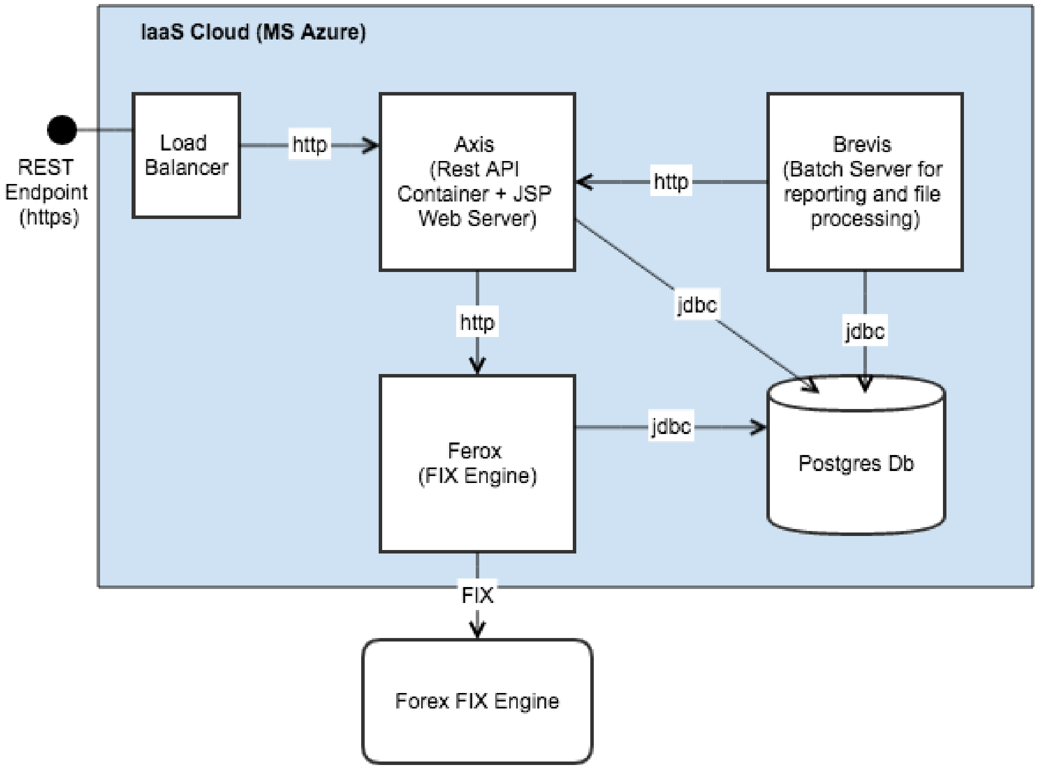
Cambrist Platform Deployment

This document describes Cambrist platform in the context of deploying it in a production cloud environent. The document is mean to assist the IT consultants that Cambrist will employ in order to accomplish the initial platform setup with ongoing monitoring and support later on.

Platform Overview

The platform Aequus's purpose is to provide a service that other financial companies can use in order to facilitate currency conversion in the conext of payment card processing.

Fugure 1 – Platform Architecture



The diagram above shows high level of Cambrist Platform named **Aequus**. The main components are:

1. **Axis** (Rest API Container + Web Server) – This is the core business layer API which gets exposed to the outside partners via REST. The service runs inside of Jetty container.
2. **Ferox** (FIX Engine) – This component facilitates communication with a 3rd party Forex provider. It also exposes a simple REST API to hide any complexity of the underlying protocol used by a particular forex provider. The service runs inside of Jetty container.
3. **Brevis** (Batch Server for reporting and file processing) – This service is responsible for running various business batch jobs. The service runs inside of Jetty container.
4. **Postgres Db** – This is the primary db. It contains separate schemas for Axis and Ferox. In real production environent the database will need to be configured with some sort of replication, where the primary instance is used for read-write operaions, while secondary instance (hot

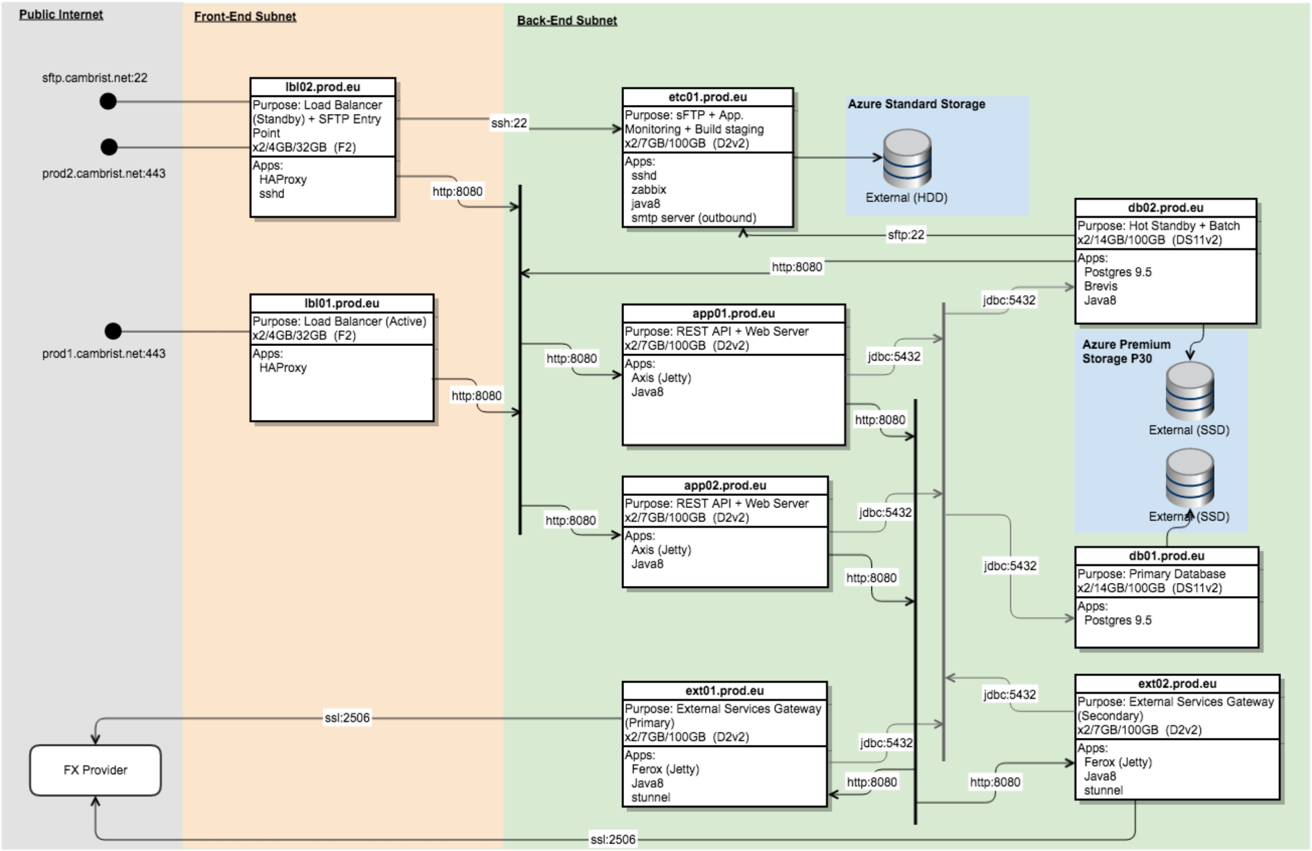
standby) is used to read-only queries (i.e. for business reporting).

5. **Forex FIX Engine** – a 3rd party service for currency trading.
6. **Load Balancer** – a front-end load balancer whose primary goal is to be able to achieve zero downtime deployments.

Platform Deployment

The platform will be deployed in Azure cloud using CentOS 7.2 Virtual Machines. The proposed deployment scheme is outlined below.

Figure 2 – Aequus Deployment Diagram



Virtual Machines

Please note the diagram above represents a deployment scheme from the application developer point of view. Each rectangle represents a VM where the title contains a suggested machine name (in bold); the section below contains the primary purpose of the machine and its expected sizing according to Azure vm sizing scheme. And the third section contains the list of required applications to be installed on that machine. Each VM is a Linux CentOS 7.2 (or whichever latest stable version of CentOS is available in Azure). The machine naming convention is defined as the following:
`<purpose#index>.<environment>.<region>`

Subnets

There are two subnets – a front-end and back-end. All machines on the back-end subnet should not be accessible from the outside internet. Any required access to the back end machines should be accomplished via a reverse proxy running on the front-end subnet (see *Load Balancing* section below).

Database

Postgres 9.5 database is configured with two nodes: Primary Active and Hot Standby. The former is used for read-write operations, and the latter is for read-only query. Postgres supports many forms of replication. It's suggested we go ahead with an asynchronous streaming replication. Both db01.prod.eu and db02.prod.eu are rather powerful DS11v2 instances. In order to keep costs low, the db02.prod.eu is also used to run the batch server Brevis. Both db instances will utilize external disks via Azure Premium Storage P30 (SSD based).

File Server

The etc01.prod.eu is designated as a file server where business partners will deposit various files using SSH. The server is accessible from outside via reverse proxy HAProxy running on lbl02.prod.eu. However within the backend subnet, the Brevis app (on db02.prod.eu) will have a direct access to the file server. The file server will have an external disk via Azure's standard storage account.

SMTP Server

The etc01.prod.eu is also designated as a outbound SMTP server for various applications to be able to send an outgoing email. The smtp server is not meant to receive any inbound email.

Endpoints

There will be following endpoints publicly accessible from the Internet:

prod1.cambrist.net:443 – primary https access to the RESTful API

prod2.cambrist.net:443 – secondary https access to the RESTful API

sftp.cambrist.net:443 – ssh based access to the file server. This end point can also be used to ssh into etc01.prod.eu which turns this host into a jumpbox. Please note that ssh authentication should be resticted to public/private key method; while username/password access should be disabled.

Load Balancing

The load balancing of the incoming RESTful calls is achieved via HAProxy service running on lbl01.prod.eu. HAProxy is capable of handling traffic at TCP and/or HTTP level. It is also used for SSL termination. lbl02.prod.eu will be used as a backup load balancer. The client application will be responsible for implementing the failover logic, if needed.

Monitoring

The etc01.prod.eu will run a monitoring server (zabbix?) which will collect various health data from the applications. In order to get external web access to the monitoring server, the user can establish an ssh port forwarding to etc01.prod.eu via login to the sftp.cambrist.net endpoint.