**SGX Environment’s Guide**

September, 2016

SGX Environments are hosted with AWS in two regions.

1. North America (QA)
2. Singapore (Production)

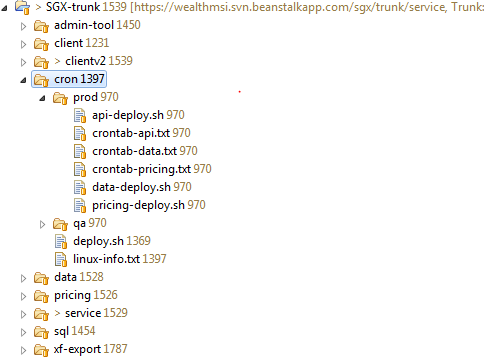
Each SGX environment exists in its own VPC and contains roughly the same makeup (although the Singapore region tends to have more machines to handle load).  All the machines in the environment use Amazon Linux instances of different memory/network configurations.  There is a base image in both locations (US and Singapore) that you can use to spawn up new instances.  Basic requirements for the machines are

1. At least 4G memory
2. Moderate to High network latency
3. JAVA
4. Tomcat 7
5. NGINX

The AMI's saved in each location have optimized configurations for both NGINX and Tomcat as well the configuration parameter in the tomcat7.conf file that denotes the environment (this helps pick up the application database, load balancer and other settings).

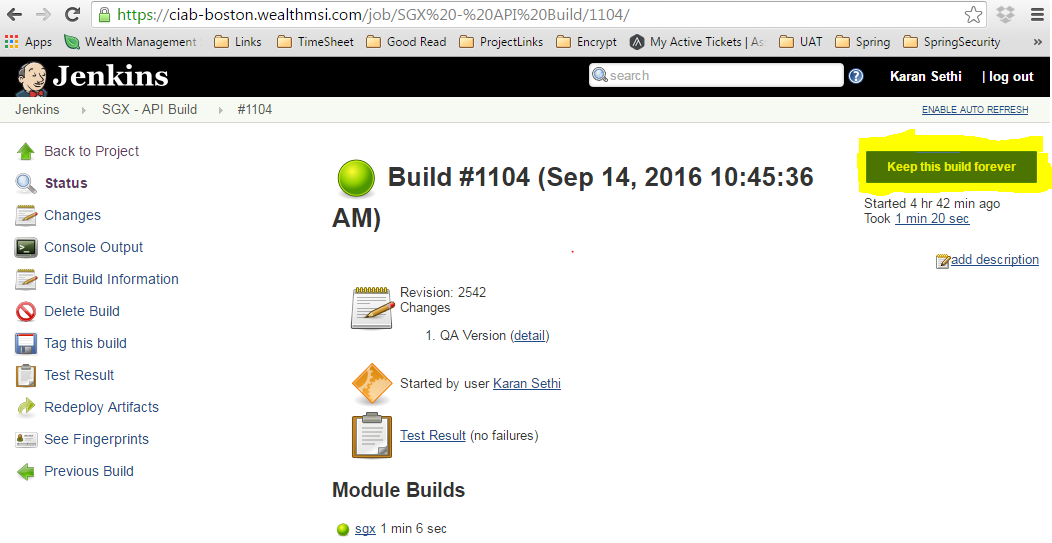
Deployment Scripts:

There are typically 2 cron jobs that run on each machine.  The crons themselves run under root and all code for them is checked into the main SGX repo (under cron).



The first, checks for new builds being available on Jenkins machine and if they are downloads them, deploy and restarts the server.

Deploying to QA works little different than Production. In the case of QA, it pulls the last good build from Jenkins - [https://ciab-boston.wealthmsi.com](https://mail.dstsystems.com/owa/redir.aspx?C=RlRyu1bliUCyaYaJwQwkSbnppGQZ4tMI3KF_Kc5uDf0z8jCW631LD531vy-dNW6E1HkKrp-xtdo.&URL=https%3a%2f%2fciab-boston.wealthmsi.com) – and deploys to the machine, for the production servers it checks for the last good build marked "keep this build forever" and does the same thing.  So If you want to deploy something to production make sure the mark that build as “Keep this build forever” from Jenkins as shown below.



The second cron pushes the logs to an s3 bucket (logs-wmsi) so they can be backed up and reviewed without logging in.

On our API/UI servers (they do dual duty) there is typically another cron that synchs the content of the matching S3 bucket to the /var/www directory.  This is allows our UI team to post the HTML for a build to an S3 bucket and the server pulls the most updated files every 2 minutes or so.

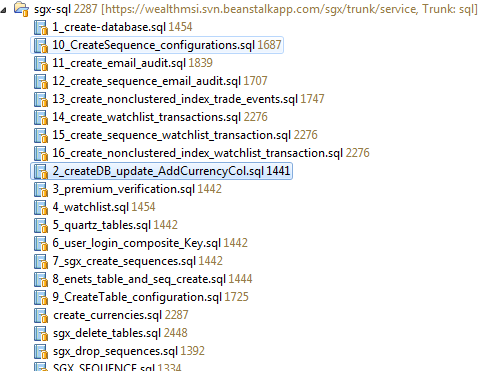
Main Components of SGX Environments

All of the components mentioned below are hosted on different machines.

1. **RDS (MS Sql Server) database**

Each SGX Environment has one AWS maintained SQL Server Database. This database stores all the user account information along with the user auditing tables. The scripts to recreate this DB are placed in the SGX repo (under /sql/).

DB scripts in the SGX repo (under /sql) are numbered and they must be executed in that order. For example 1\_create-database.sql must run before 2\_createDB\_update\_AddCurrencyCol.sql and so on. No special configurations for this DB is required.



1. **API Servers:**

Each SGX Environment has multiple instances of these servers based on the load requirements. These servers serve all the api calls generated from client side to retrieve data. This is the middle layer of the application. These are served by tomcat7 server.

1. **Data Loader:**   
     
   Each SGX Environment has one AWS hosted Linux instance.  This machine takes the successful build from the repo SGX/data.  It has cron scripts associated with it in repo which runs every day at 1:50 PM EST for Production and 2:50 PM EST for QA to load and store the daily data feed into Elastic Search machines. This machine does not have any special configurations.  This machine is controlled by Jenkins and cron scripts.
2. **Elastic Search Instances:**   
     
   Each SGX environment has three load balanced instances. These use the standard Amazon Linux images described earlier.  Only special configuration for these are the custom node name (so they can discover each other).  There are no cron scripts associated with these machines.  ES instances are set to start on boot. Elastic search loads all the stock data received from our data provider S&P. This data is saved every day at a fixed time by the above mentioned data loader cron jobs.
3. **Pricing Server:**   
     
   It uses same standard Amazon Linux instance mentioned in other machines.  This would run on shared servers, except that S&P requires all requests come from a unique IP address.  It is important to note that the most important part of this machine is the IP address, as we can't get pricing info from a non-registered address. We run this on its own server because we can't scale like the data load or API/UI servers.  It takes the build from the repo SGX/pricing.
4. **Data Export Job:**   
     
   In addition, we use what S&P calls the ExpressFeed package.  This system is a combination of two SQL Server databases (one as a control database and another that stores all the S&P data), a loader program (provided by S&P) and an export program (.NET written by us and the SGX/xfloader repo).  These DB's are run on two different windows machines (we couldn't use Amazon RDS because of special configuration requirements) with the target DB, loader and exporter running on one and the large S&P database running on the other.  Currently this runs in the US SGX VPC.  The environment is not backed up as we can spin up machines pretty easily and just startup the S&P loader program which will recreate the databases.  The exporter written by us is a scheduled task that kicks of the EXE at 12:30pm EST every day.  The configuration and details on that program can be found in the readme in its repo.
5. **UI Code**: All the UI code is hosted at AWS S3 bucket named as sgx-premium.wealthmsi.com for production environment.