Smarter Balanced Reporting

Runbook, Deployment, Administration

AWS Version

Prepared for: 

# 

by: 

Approvals

|  |  |  |  |
| --- | --- | --- | --- |
| **Representing** | **Date** | **Author** | **Status** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Revision History

|  |  |  |
| --- | --- | --- |
| **Revision Description** | **Author/Modifier** | **Date** |
| Initial Release (DRAFT) | Anna Grebneva (Amplify) | 2014.07.09 |
| Added Test Score Batcher | Doris Ip (Amplify) | 2014.11.03 |
| Added AWS | Clinton Wolfe | 2015.05.08 |

Table of Contents

[1 Summary](#h.dj7grn26ie4r)

[2 Data Capacity Planning Assumptions](#h.8cq6y1louf7h)

[3 Hardware Minimum Requirements](#h.aa93okewjnpy)

[4 Required Infrastructure Components](#h.d248ikpy8tt8)

[Access Control and Secrets Management](#h.r7w7olfqqnv0)

[Monitoring and Logging](#h.k9kcl333ut4c)

[Automation](#h.1f252d1exulm)

[Network Configuration](#h.5u77uacg2az5)

[Subnet Configuration](#h.o81jrvvpg11e)

[Security Group Configuration](#h.papyw4n6qx2p)

[Other Needs](#h.xpycirsckc8c)

[5 Software Requirements](#h.67f1o2htgeld)

[5 Planning for Availability](#h.bx7agbq4h229)

[6 Archival, Backup and Recovery](#h.79edby99gwef)

[7 Installation & Configurations](#h.yfeznwtk3g52)

[7.1 Reporting Web Servers](#h.4rh6s1wzp5iz)

[7.1.1 Installation](#h.iz87h079gnc7)

[7.1.2 Apache Configuration](#h.xu8y2p966bir)

[7.1.2.1 MPM Mode](#h.qnraufrjq6zs)

[7.1.2.2 Configuration Files to Add](#h.d1t74hsm8rdv)

[7.1.2.3 Configuration Files to Remove](#h.n3zdhcaig3bg)

[7.1.3 Smarter Application](#h.rlcpcydvgs7y)

[7.1.3.1 Generating smarter.ini](#h.ki3mskb1afgm)

[7.1.3.2 Configuration File for Authentication between OpenAM SAML and Smarter](#h.6tm0s9euoy9a)

[7.1.3.3 Adding Custom Metadata Configuration](#h.o2fgvyda38k4)

[7.1.3.4 Configuring syslog](#h.pagjs3rs4bm5)

[7.1.3.5 Creating a Database Schema for Production Database](#h.1ykeljtpghpc)

[7.2 Load Balancer](#h.fx0g01inomxi)

[7.3.1 Installation](#h.22urj4l615b5)

[7.3.2 Configuration](#h.2hqcoe7s51rj)

[7.4 PDF Worker [Ansible name: reporting-worker-pdf]](#h.slkrbquqbv4c)

[7.4.1 Installation](#h.a9cm003zx56q)

[7.3.2 Configuration](#h.hrusrg3lgvr9)

[7.3.2.1 GlusterFS](#h.8r1qropz2h02)

[7.3.2.3 celeryd-services](#h.7z7pxwwl41qb)

[7.4 PDF Pre-Generator [Ansible name: reporting-generator-pdf]](#h.1ddi01dlcj1d)

[7.4.1 Installation](#h.8jsw6vl01tyx)

[7.4.2 Configuration](#h.z83yjvfcxmga)

[7.4.2.1 smarter](#h.hhc0egruyqdo)

[7.4.2.2 WSGI](#h.l7owde1jkzcq)

[7.5 Extract Messenger [Ansible name: reporting-rabbit-extract]](#h.ftl4oyvim8u3)

[7.5.1 Installation](#h.fdxxdk3pc9sq)

[7.5.2 Configuration](#h.4c55yzrkxdl6)

[7.6 Extract Worker [Ansible name: reporting-worker-extract]](#h.fmr5u543s7dv)

[7.6.1 Installation](#h.naqsyzaikztv)

[7.6.2 Configuration](#h.daw2cwe62blf)

[7.7 Cache [Ansible name: memcached]](#h.niq3hqy3718o)

[7.7.1 Installation](#h.d3cfaho3cj65)

[7.7.2 Configuration](#h.beunftc1g7k2)

[7.8 Cache Warmer [ansible name: reporting-cache-warmer]](#h.44fuftdu1mya)

[7.8.1 Installation](#h.5kwlt1bq1591)

[7.8.2 Configuration](#h.ajy0uwchqxag)

[WSGI](#h.ayfyualtgfg4)

[Configuration for Smarter application specific WSGI configuration](#h.uzh973j1hj5)

[smarter](#h.lzbre8d5vqt0)

[Generating Smarter.ini](#h.tdi9hv8tumma)

[Configurations specific to Cache Warmer in Smarter.ini](#h.j80soe2f5tl4)

[Configuration for Demographics Filters](#h.l4mh6k3cebjr)

[7.9 Database Master [ansible name: db-master]](#h.92fx825yt7ie)

[7.9.1 Installation](#h.sad7q158kyc9)

[postgres](#h.60bbqg2b9ib3)

[Configuring postgres master](#h.dwvppxm8genr)

[7.10 Database Replica](#h.upfj3y9608b)

[7.10.1 Installation](#h.5tkw3t4he71l)

[postrgres](#h.ey9qqw6jcrud)

[celeryd-edmigrate](#h.35jb9ltgm8az)

[7.11 Database Load Balancer](#h.o2s9vn9fy9sp)

[7.11.1 Installation](#h.lxku5xqp8dug)

[7.12 Database Pool [ansible name: reporting-db-pgpool]](#h.yejo3lknh3g3)

[7.12.1 Installation](#h.ezjcd4k6rsp0)

[pgpool for Smarter Balanced Reporting web frontend](#h.g6vu9opvzn0h)

[pgpool for Extracts](#h.tf4d8jv9x0vs)

[7.13 Smoke test for smarter functioning](#h.wgp6ruvhltpb)

[7.14 Landing Zone](#h.j71i4s791m92)

[7.14.1 Installation](#h.r5uvi0mihsb1)

[7.14.2 Configuration](#h.jsdhb7ysc3x)

[edsftp](#h.l3mh6m8djdbc)

[Configuring chroot](#h.4u06a97pvrl)

[Generating Smarter .ini file](#h.ps9bxhwm4ohx)

[Creating Groups for SFTP users](#h.85gad03jw2r)

[Creating Tenant Accounts](#h.inuv7v4a25tl)

[Starting EdSFTP watcher service](#h.wudmsvn3srsq)

[7.15 Loader](#h.ui8uwr21vv0f)

[7.15.1 Installation](#h.bq4ie6qeplhr)

[7.15.2 Configuration](#h.x3uyu2lefx79)

[celeryd-udl2](#h.kwc96jlj548a)

[Generate udl2\_conf.ini](#h.brsmlz3mvl7c)

[Generate smarter.ini](#h.s1tf0uu4m90)

[Initialize the UDL2 database](#h.tcgkrmz6u4l0)

[Ensure GPG keys are copied to /opt/edware/keys](#h.q359r7rcbclw)

[Mount Work Zones directories from Gluster](#h.aiesgkuqvf8f)

[Ensure Outgoing HTTP and HTTPs ports are opened](#h.aoieogs2w5s2)

[Start edudl2-file-grabber and edudl2-trigger service to watch for incoming files being copied to work zone](#h.7yjvt8s19iq6)

[7.16 Loader Messenger](#h.8zk1h46ngmwv)

[7.16.1 Installation](#h.vk8yycihd77z)

[7.16.2 Configuration](#h.r0p5kjgiicq1)

[7.17 Loader Database](#h.wyexf5ueg6rd)

[7.17.2 Configuration](#h.22ony8xf3l9y)

[postgres](#h.3nlk2xplswf3)

[Allow username/password based Authentication](#h.jgux8ln3ak0o)

[Allow client configuration](#h.m5fhdfey9y98)

[Create Database User](#h.lr5ylib53svj)

[Create UDL database](#h.59pt7gm4vq96)

[Create udl2 User and Group](#h.r4t6wb3mrr49)

[Prepare Work Zone Directory](#h.84pkwo7r3nx)

[7.18 Database Staging](#h.s4m379nzoeyx)

[7.18.2 Configuration](#h.6gmkpamrtzr4)

[postgres](#h.nncp6fwrb8zp)

[Allow username/password based Authentication](#h.teg8lsaut847)

[Allow client configuration](#h.2j0sdgfygcjz)

[Create Database User](#h.8mai9nhbjnhs)

[Create the edware database](#h.hp6tc6mmk9rg)

[7.19.1 Installation](#h.p3q64axlia5v)

[7.20 Migrator](#h.40o8fv7fobjv)

[7.20.1 Installation](#h.rly333xavw1o)

[7.20.2 Configuration](#h.mzfvg2pyxmeq)

[RabbitMQ](#h.g8abd56fnxw)

[edmigrate](#h.3i8nn3j9kxw0)

[Setting up the .ini file](#h.id6a2depazoe)

[7.21 HTTPS Pickup Zone Web Server](#h.no7tk7y5lg7h)

[7.21.1 Installation](#h.3hf4crlllvqi)

[7.21.2 hpz-web Configuration](#h.duvb6hkpluwe)

[7.21.2.1 Configure apache](#h.a56rk6jt97l7)

[7.21.2.2 glusterfs](#h.n2ch3v9yobst)

[7.21.2.3 encfs](#h.j4h2vbjx109v)

[7.21.2.4 Application Configuration](#h.cml92ylk40a8)

[7.22.2 hpz-db Configuration](#h.cavha5k2rrr6)

[7.22.3 Heartbeat](#h.eu5bie3xyrj9)

[7.23 Score Batcher Web Server](#h.e0x11yuqfv5j)

[7.23.1 Installation](#h.ebsna7rim865)

[7.22.2 Configuration](#h.x62tskmtbkrt)

[7.23.2.1 Apache Configuration](#h.pazoe4ptxjuq)

[7.23.2.2 Application Configuration](#h.i2uh4hqqbiz8)

[7.24 Score Batcher Messenger](#h.5siy20ymwblc)

[7.24.1 Installation](#h.16nw7ld1arim)

[7.24.2 Configuration](#h.pkpj9yh853ap)

[7.25 Score Batcher Worker](#h.atred89g9spd)

[7.25.1 Installation](#h.t5rrzhtvobou)

[7.25.2 Configuration](#h.12acx6u7ng6d)

[Custom XML Metadata Directory Structure](#h.tw9pedwsu3k3)

[7.26 Score Batcher Trigger](#h.l92z0gix1bt8)

[7.26.1 Installation](#h.ldzcnhm57i1v)

[7.26.2 Configuration](#h.pltvq8djyi6w)

[7.26 Score Batcher Database Server](#h.4qxvvy7joeot)

[Score Batcher Database Server is a temporary storage that saves assessment information in received XML. This information will be processed by Score Batcher Trigger and then archived and encrypted into a batch.](#h.z5cxmudhlkze)

[7.26.1 Installation](#h.7thfzj64q3et)

[7.26.2 Installation Creating a Database Schema](#h.nmr7c1zdqezn)

[7.26.3 Configuration](#h.wj9pxqksq7wa)

[Once the database is ready, one needs to update the database connection configuration in smarter\_score\_batcher.ini on each smarter score batcher worker server. Update the below two directives with the actual values of database connection:](#h.8ommd0yzrgga)

[8 Starting Applications](#h.txzduy2hsvbj)

[9 Logging & Monitoring](#h.hef4oksbmeab)

[9.1 Web Server](#h.2moa06ow1cur)

[9.2 HTTP Pickup Server](#h.5bvh93abwdiu)

[9.3 PDF Messenger [Ansible name: reporting-rabbit-services]](#h.rer6yw9prcyr)

[9.4 PDF Worker [Ansible name: reporting-worker-pdf]](#h.br5irvz7jeld)

[9.5 PDF Pre-Generator [Ansible name: reporting-generator-pdf]](#h.o9k4yfrpyuuz)

[9.6 Extract Messenger [Ansible name: reporting-rabbit-extract]](#h.px628l8rgu1h)

[9.7 Extract Worker [Ansible name: reporting-worker-extract]](#h.chs7xvxx529v)

[9.8 Cache [Ansible name: memcached]](#h.53loudgyb6li)

[9.9 Cache Warmer [ansible name: reporting-cache-warmer]](#h.l9gjdve7gxzh)

[9.10 Database Master](#h.twkqsgumrlx4)

[9.11 Database Replica](#h.yxyq1pd2nruv)

[9.12 Database Load Balancer](#h.x9eoznqscz3p)

[9.13 Database Pool](#h.rgbomu79tnao)

[9.14 Landing Zone](#h.1cq3by1f5l88)

[9.15 Loader](#h.rsaebz2v445u)

[9.16 Loader Messenger](#h.27no8uu9enpw)

[9.17 Loader Database](#h.2wf7bshm39zo)

[9.18 Database Staging](#h.otrksdd2clsr)

[9.19 Migrator](#h.h4q3lf7g4bl)

[9.20 Score Batcher Web Server](#h.7b4rw2191hs7)

[9.21 Score Batcher Messenger](#h.y7ijwrhhdlje)

[9.22 Score Batcher Worker](#h.7pw3lc1xtl0p)

[10 Troubleshooting](#h.svslmpuvt6tm)

[11 Maintenance](#h.gmx9eoj1sq2p)

[11.1 Cache](#h.fwcbbqwnocdd)

[11.2 Database](#h.cnly8d230bdi)

[11.3 HTTP Pickup Zone](#h.h58lu39y77cw)

[Appendix A: Apache Configuration Files](#h.a56b65wr9udj)

[conf/httpd.conf](#h.2qbf62fxiyvm)

[conf.d/edware\_worker\_mpm.conf](#h.uk2zo92ds3f5)

[conf.d/rewrite\_reporting\_https.conf](#h.ezxn19yyl0pj)

[conf.d/rewrite\_reporting\_slash.conf](#h.nnhhsvjj42d5)

[conf.d/rewrite\_tsb.conf](#h.owy771ifxwwz)

[conf.d/ssl\_main\_server.conf](#h.nty2j6jmsyvh)

[conf.d/ssl\_on.conf](#h.f3181tvin1t7)

[conf.d/wsgi\_edware.conf](#h.yj8r6lu3u9pf)

[conf.d/wsgi\_frs.conf](#h.tf9bt0k03dim)

[conf.d/wsgi\_swi.conf](#h.6wowh27ot0bc)

[conf.d/wsgi\_tsb.conf](#h.bwg65nyg2qya)

[conf.d/xsendfile.conf](#h.a0bn7lgtayuv)

[Appendix B: Service Provider Metadata](#h.8x1s61es2g29)

# 1 Summary

This document is for System Administrators who will be operating the System. It contains instructions on how to install, scale, and maintain the Smarter Balanced Data Warehouse and Reporting instance.

# 2 Data Capacity Planning Assumptions

1. Smarter Balanced Reporting Database Storage:
   1. Student Registration: 0.5kb per student.
   2. Student Assessments: 2kb per assessment and 0.5kb per student.
   3. Allocate similar size to the whole postgresql database for postgresql point in time recovery.
   4. Keep disk usage less than 50% to help Database Administrators to perform postgres vacuum operations efficiently.
2. PDF file Storage:  
   The Smarter web application will pre-generate grayscale PDF’s, and color PDF’s will be generated when a user requests. Therefore, it is good practice to estimate disk space for 100% disk storage for grayscale PDF’s files and 50% disk storage for color PDF’s.
   1. color - 85kb/assessment outcome.
   2. grayscale - 75kb/assessment outcome.
3. Historical Assessment Archive:
   1. 50 mb assessment csv for 100k per assessment.
4. Individual Item Response (a.k.a. Item-Level) Data Storage:
   1. Average size of file after compression is 200kb.
   2. A student generates around 5 item-level files per school year.
   3. SBAC requires storing of 10 years of student item level xml raw data.
5. HTTP Pickup Zone Storage:
   1. [PLACEHOLDER: performance and load testing scheduled as part of M6].
   2. Database for pickup zone operation.
6. Landing Zone Storage:
   1. 0.5kb per student per assessment.
7. Loader Database Storage:
   1. [PLACEHOLDER: performance and load testing scheduled as part of M6].

# 3 Hardware Minimum Requirements

The number of instances required for each machine type is correlated to three independent criterias: the number of tenants/states in the system, the number of users accessing the system, and the number of students in the tenant/state.



Table 1 - Sample configuration for data warehouse and reporting for 1 tenant containing 2M students and 100K users.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Application** | **Avg Usage (3% user concurrency) instances** | **Peak Usage (10% user concurrency) instances** | **Cores (vCPU per instance)** | **Memory (GB per instance)** | **AWS instance type** | **non-OS portion Storage (GB)** |
| reporting-web (main web application server)2 | 9 | 30 | 4 | 8 | m3.large |  |
| load balancers2 | 2 | 2 | 2 | 2 | ELB |  |
| db-master1 | 1 | 1 | 8 | 64 | r3.2xlarge | 1000 |
| reporting-db-slave (replicas for web front end)2 | 2 | 7 | 8 | 64 | r3.2xlarge | 1000 |
| extracts-db-slave (replicas for extraction)2 | 1 | 3 | 8 | 64 | r3.2xlarge | 1000 |
| reporting-db-pool2 | 2 | 3 | 4 | 8 | m3.xlarge |  |
| memcache3 | 2 | 2 | 8 | 64 | m3.2xlarge |  |
| reporting-rabbit-services2 | 2 | 2 | 4 | 8 | m3.xlarge |  |
| reporting-worker-pdf3 | 2 | 2 | 8 | 64 | r3.2xlarge |  |
| reporting- generator-pdf1 | 1 | 1 | 2 | 8 | m3.large |  |
| reporting-rabbit-extract2 | 2 | 2 | 4 | 8 | m3.xlarge |  |
| reporting-worker-extract2 | 2 | 7 | 8 | 32 | m3.2xlarge |  |
| reporting-cache-warmer1 | 1 | 1 | 2 | 8 | m3.xlarge |  |
| lz (landing zone)1 | 2 | 2 | 4 | 8 | m3.xlarge | 50 |
| hpz-web (https pickup zone web app)5 | 3 | 9 | 4 | 8 | m3.xlarge |  |
| hpz-db5 | 1 | 1 | 4 | 8 | m3.2xlarge | 10 |
| udl (universal data loader)3 | 2 | 2 | 4 | 8 | m3.2xlarge |  |
| udl-rabbit1 | 2 | 2 | 4 | 8 | m3.xlarge |  |
| udl-db1 | 1 | 1 | 8 | 32 | m3.2xlarge | 1000 |
| migrate-rabbit |  |  |  |  | m3.xlarge |  |
| loader database (udl-db) | 1 | 1 | 8 | 32 | m3.2xlarge | 1000 |
| migrate-db (database staging)1 | 1 | 1 | 8 | 32 | m3.2xlarge | 1000 |
| gluster-server3 |  |  |  |  | m3.medium | 26480.4 |
| tsb-web (Test Score Batch web server)3 | 2 | 3 | 4 | 8 | m3.xlarge |  |
| tsb-rabbit3 | 2 | 3 | 4 | 8 | m3.xlarge |  |
| tsb-worker3 | 4 | 4 | 8 | 64 | m3.xlarge |  |
| tsb-trigger1 | 1 | 1 | 4 | 8 | m3.large |  |
| tsb-db | 1 | 1 | 8 | 32 | m3.2xlarge |  |
| file-grabber | 1 | 1 | 2 | 8 | m3.large |  |
| file-trigger | 1 | 1 | 2 | 8 | m3.large |  |
| stats-db | 1 | 1 | 8 | 32 | m3.2xlarge |  |

1 The number of instances for this machine type is proportional to every Tenant/State.

2 The number of instances for this machine type is proportional to every 100,000 Users.

3 The number of instances for this machine type is proportional to every 2,000,000 Students.

4  The number of storage for this is proportional to 10 years of item level data storage plus 1 year of pdf.

5 The number of storage for this is proportional to every 100,000 Users.

# 4 Required Infrastructure Components

In addition to the components provided, several infrastructure components are required to successfully deploy Smarter Reporting Live. These components are not included and may be implemented as the operator sees fit.

## Access Control and Secrets Management

* Previous deployments have used IAM users, or AWS-SAML integration, to control access to the AWS console.
* Secrets, such as SSL keys, were stored on S3 buckets unique to each application. IAM instance profiles were used to bind each application type to a list of S3 buckets it may access.
* SSH keys were baked into the AMIs that were used.

## Monitoring and Logging

* Metrics were collected using sensu and stored using graphite/carbon.
* Sensu was configured to send alert triggers to pagerduty.
* The application is in its infancy in terms of metric availability. Many metrics have to be collected through indirect or awkward means.
* Log aggregation was performed using rsyslog to send logs to logstash, then indexed using ElasticSearch and queried using Kibana.

## Automation

* Some mechanism to create and configure the nodes will be needed.
* Ansible was used to perform single-node configuration.
* Mass spinup of nodes was implemented in python/boto.
* Previous deployments did not implement auto-scaling.

## Network Configuration



### Subnet Configuration

The basic configuration consists of three subnets. Similar to a traditional AWS layout, it includes a public subnet (which contains an Internet gateway and nodes with public IP addresses) and a private subnet. The private subnet contains nearly all nodes, including web servers, database servers, workers, rabbitMQ servers, and other utility machines. The special LZ subnet contains the landing zone node which is exposed to the public on port 22.

### Security Group Configuration

In all subnets, each host class was restricted to accept only those incoming connection ports that were required for it to operate. These assignments were handled through automation; details are available upon request.

## Other Needs

* NTP service
* A private YUM repo
  + s3iam yum plugin, with repo in S3, worked well.
* Low-volume outbound email (for internal alerts) - SES was previously used.

# 5 Software Requirements

The system has been tested with servers running on CentOS 6.5. A brief summary of software requirements for each machine type is listed in the table below. Please refer to the other sections of this document for more details.

Table 2 - Machine Types with their software requirements

|  |  |  |
| --- | --- | --- |
| **Machine** | **External Software (Available from Linux distributors)** | **Software provided by Amplify** |
| web servers | httpd  xmlsec1  xmlsec1-openssl-devel | python3-3  python3-mod\_wsgi  smarter |
| http landing zone | httpd  mod\_xsendfile  xmlsec1-openssl-devel  xmlsec1  glusterfs  glusterfs-fuse  fuse-encfs | python3-mod\_wsgi3-3  hpz |
| http landing zone database | postgresql92 postgresql92-libs postgresql92-server postgresql92-contrib |  |
| load balancers |  |  |
| database masters | postgresql92 postgresql92-libs postgresql92-server postgresql92-contrib  repmgr |  |
| database replicas for web reporting | postgresql92 postgresql92-libs postgresql92-server postgresql92-contrib  repmgr  xmlsec1  xmlsec1-openssl-devel | python3-3 python3-mod\_wsgi smarter |
| database replicas for extraction | postgresql92 postgresql92-libs postgresql92-server postgresql92-contrib  repmgr |  |
| db pool with failover for web front end | pgpool-II |  |
| db pool without failover for bulk extract | pgpool-II |  |
| cache | memcached |  |
| pdf messenger | rabbitmq-server |  |
| pdf workers | glusterfs  glusterfs-fuse  fuse-encfs  urw-fonts | wkhtmltox-0.12.1-1.x86\_64  pdfunite |
| pdf generator | httpd xmlsec1  xmlsec1-openssl-devel | python3-3 python3-mod\_wsgi smarter |
| extract messaging | rabbitmq-server |  |
| extract worker | glusterfs  glusterfs-fuse  fuse-encfs  xmlsec1  xmlsec1-openssl-devel | python3-3 python3-mod\_wsgi smarter |
| cache warmer | httpd  xmlsec1  xmlsec1-openssl-devel | python3-3 python3-mod\_wsgi smarter |
| landing zone |  | python3-3  edsftp |
| loader | glusterfs  glusterfs-fuse  fuse-encfs | python3-3  edudl |
| loader messenger | rabbitmq-server |  |
| loader database | postgresql92 postgresql92-libs postgresql92-server postgresql92-contrib  glusterfs  glusterfs-fuse  fuse-encfs |  |
| migrator | xmlsec1  xmlsec1-openssl-devel  rabbitmq-server | python3-3 python3-mod\_wsgi smarter |
| database staging | postgresql92 postgresql92-libs postgresql92-server postgresql92-contrib |  |
| gluster (storage) | glusterfs-geo-replication  glusterfs  glusterfs-server  glusterfs-fuse |  |
| score batcher web server | httpd xmlsec1  xmlsec1-openssl-devel | python3-3  python3-mod\_wsgi  smarter\_score\_batcher |
| score batcher messenger | rabbitmq-server |  |
| score batcher worker | glusterfs  glusterfs-fuse  fuse-encfs  xmlsec1  xmlsec1-openssl-devel | python3-3  python3-mod\_wsgi  smarter\_score\_batcher |
| score batcher trigger | glusterfs  glusterfs-fuse  fuse-encfs  xmlsec1  xmlsec1-openssl-devel | python3-3  python3-mod\_wsgi  smarter\_score\_batcher |

The following diagram illustrates the architecture of, and relationships between the components listed above:



# 5 Planning for Availability

The reporting application is designed for high availability and each installation can be setup to provide the service with minimal interruption.

The following architectural diagram is a sample configuration intended to minimize administrative involvement to failover process.

Table 1. Sample Configuration for Smarter Balanced Reporting



Table 2. Sample Configuration for Smarter Data Loader



* Apache  
  All web servers are registered to ELB. When a web server goes down, ELB will not route a request. When a web server comes back online, ELB will resume routing a request.
* DB Connection Pool
  + PgPool-II  
    The version of PgPool-II must be 3.3 or greater. PgPool-II uses “watchdog” to monitor each other. An active PgPool-II server uses virtual IP addresses to receive all database requests and forward to specific PostgreSQL servers by roundrobin. Standby PgPool-II becomes active and takes over the virtual IP when an active PgPool-II goes down.
* Memcached:
  + Memcached supports clustering. By setting up clustering, it provides load balancing and redundancy. Configuration guidance can be found in the following external documentation: <https://code.google.com/p/memcached/wiki/NewConfiguringServer>
* PostgresQL 9.2
  + The Smarter Balanced Data Warehouse uses multiple read-only replication databases to provide redundancy. The failover for database is via PgPool in previous sections.
  + The data loader can be configured with multiple staging databases to parallelize data injection. This also provides failover for the data loading pipeline when a subset of the staging database is not in available.
* Data Loader
  + The data loader can be configured with multiple processing servers to process the data loading job in parallel. This provides redundancy and load balancing in the data loading process.
* Data Loader Messenger
  + The data loader can be configured with RabbitMQ clustering to provide redundancy and failover and load balancing for the data loading process.
* Landing Zone:
  + No provision for redundancy is provided.

# 6 Archival, Backup and Recovery

1. Backup and restore PostgreSQL (**manual process**).

Use the PostgreSQL master database to backup the database.

Depending on the size of the database, it may take several hours.

1. Stop EdMigrate service:  
   /etc/init.d/edmigrate-conductor stop
2. Extract PostgreSQL data (please refer to the man for pg\_dump):  
   pg\_dump -f /path/to/backup.sql.gz -Z 9 -U edware -W edware
3. Resume Edmigrate serivce:  
   /etc/init.d/edmigrate-conductor start

Encrypt & Upload to S3:

1. Create a Python script to automatically encrypt and then upload **backup.sql.gz** to S3, into it’s respective folder, as follows:
   1. S3 bucket: **srl-backups-pg**
   2. GPG encrypt the **backup.sql.gz** file to its respectable filename, as below.
   3. Hourly/Daily/Monthly folders: Python script should detect the time of execution and name the encrypted file like this:
      1. srl-backups-pg/hourly/backup.1400.sql.tar.gz.gpg (2:00pm backup).
      2. srl-backups-pg/daily/backup.28.sql.tar.gz.gpg (28th of the month backup).
      3. srl-backups-pg/monthly/backup.05-May.sql.tar.gz.gpg (monthly backup for May).
      4. Cleanup: Python script should delete the 2 local files to save space:
         1. **backup.28.sql.tar.gz.gpg**
         2. **backup.sql.tar.gz**

To restore the Smarter Balanced Reporting System database:

1. Download the latest backup file from S3.
2. Run the Python script to decrypt the backup file.
3. Stop the EdMigrate service:  
   /etc/init.d/edmigrate-conductor stop
4. Restore PostgreSQL data (please refer to the man for pg\_restore.)  
   pg\_restore -U edware -W -d edware /path/to/backup.sql.gz
5. Resume the Edmigrate service:  
   /etc/init.d/edmigrate-conductor start
6. Please check <http://www.slideshare.net/InesSombra/data-antipatterns-nycdevops> for more information on the following process:
   1. Provisioning enough disk space for database backup.
   2. Proper backup schedule for full backup and incremental backup, see <https://wiki.postgresql.org/wiki/Incrementally_Updated_Backups>, <http://stackoverflow.com/questions/5529603/best-method-for-postgres-incremental-backup> and <http://www.postgresql.org/docs/9.2/static/continuous-archiving.html> for good incremental backup for PostgresQL.
   3. Set up good backup retention time.
   4. Periodically practice restore database backups to detect corrupted backup files.
   5. Always test the restored backup with smarter app to make sure the restoration from backup is proper.
   6. Set up recovery protocols, plans, and practice disaster recovery.
7. Historical Assessments archive:
   1. Periodically backup history assessment files in landing zone history to tapes/optical media/amazon Glacier.
8. Item-Level raw data:
   1. We suggest setting up rsync and a redundant item level storage cluster as a backup plan due to the number of files and size of data.
   2. Periodically backup item-level files to tape/optical media/amazon glacier.
   3. Item level data file may be updated/modified, so it is advisable to plan incremental backup on whole file repository periodically. Make sure the gap for missed item-level raw data is as minimal as possible
   4. Tar may not be usable in backup item-level data. See: http://serverfault.com/questions/329273/store-and-backup-200-million-small-files

# 7 Installation & Configurations

## 7.1 Reporting Web Servers

Reporting Web Servers are responsible for running the Smarter Balanced Reporting Web Application. This application is written in Python, and runs on Apache using WSGI.

### 7.1.1 Installation

The following table lists the RPMs that must be installed on reporting-web servers.

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| Python | 3.3.0\* | python3 | Python 3 RPM is generally not available from Linux distributions, hence Amplify has supplied a customized Python 3 RPM |
| Apache | 2.2.15 | httpd | Apache hosts Smarter Web Application. It can be installed from the standard distributed RPM. We highly recommend using customized configuration. Apache should run on worker MPM mode. |
| WSGI | 3.4\* | python3-mod\_wsgi | WSGI defines a simple and universal interface between web servers and web applications or frameworks for Python. This is a customized RPM supplied by Amplify. |
| Smarter | TBD\* | smarter | Amplify’s Smarter Web Application |
| XMLSec | 1.2.20 | xmlsec1  xmlsec1-openssl-devel | XMLSec can be installed from standard distributed RPM. It is used for verification of SAML Responses. |

\* Provided by Amplify.

### 7.1.2 Apache Configuration

#### 7.1.2.1 MPM Mode

The nature of the application is CPU intensive, and requires Apache to run in worker MPM (Multi-Processing Module) mode.

Modify **/etc/sysconfig/httpd** to set Apache to use worker MPM mode.

*# Configuration file for the httpd service.*HTTPD=/usr/sbin/httpd.worker

##### 7.1.2.2 Configuration Files to Add

See Appendix A for the details of the contents of these files. All paths are under **/etc/httpd**.

The Reporting Web Server requires:

* conf/httpd.conf
* conf.d/edware\_mpm\_worker.conf
* conf.d/rewrite\_reporting\_https.conf
* conf.d/rewrite\_reporting\_slash.conf
* conf.d/ssl\_main\_server.conf
* conf.d/wsgi\_edware.conf

##### 7.1.2.3 Configuration Files to Remove

These files are included with the stock RPMs, and should be removed:

* conf.d/ssl.conf
* conf.d/welcome.conf

#### 7.1.3 Smarter Application

Smarter is the main web application. It provides HTML, CSV and PDF data access to end users. The application runs on Apache web server via WSGI with Python Pyramid Framework.

The Smarter RPM, provided by Amplify, packages an entire Python 3.3 virtual environment and all Python dependencies used by the application. The Smarter RPM installs the virtual environment in **/opt/virtualenv/smarter**. A utility packaged within the RPM is provided to generate the **.ini** configuration file for the Smarter Web Application. The generation of the **.ini** file is the administrator’s/operator’s responsibility.

##### 7.1.3.1 Generating smarter.ini

Smarter needs to read an **.ini** configuration upon start-up. The default location of this file is **/opt/edware/conf/smarter.ini**, and the configuration of the path is specified in **/opt/edware/smarter/smarter.wsgi**.

Each environment is unique, hence, the configuration of each configuration is unique. The operator must generate the **.ini** file for each server/environment. Within the directory, **/opt/edware/conf/**, a **settings.yaml** file exists that defines the default and environment specific key/value pairs of the configuration. For permanent changes to the configuration of an environment, changes should be made in **/opt/edware/conf/settings.yaml**.

Here are the steps to generate the **.ini** file for the “**uat**” environment:

shell> . /opt/virtualenv/smarter/bin/activate

(virtualenv) cd /opt/edware/conf

(virtualenv) python generate\_ini.py -e uat

(virtualenv) mv uat.ini smarter.ini

##### 7.1.3.2 Configuration File for Authentication between OpenAM SAML and Smarter

The Smarter Web Application needs to verify the authentication with OpenAM SAML by IDP Metadata XML file. This metadata file doesn’t come with the RPM. The operator must generate and reference the path of this file inside smarter.ini. By default, Smarter expects this file to be located in **/opt/edware/conf/idp\_metadata.xml.**

##### 7.1.3.3 Adding Custom Metadata Configuration

Every tenant/state has the option to insert custom metadata into their production database’s **custom\_metadata** table. This table has three columns:

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Description** | **Example Value** |
| state\_code | The state code of a particular tenant. | NC |
| asmt\_subject | The subject for a particular assessment | Math |
| asmt\_custom\_metadata | A JSON formatted string that contains the minimum cell size, and colors used for assessment representation within reports. | {"min\_cell\_size":30, "colors":[  {"text\_color":"#ffffff","bg\_color":"#DD514C","start\_gradient\_bg\_color":"#EE5F5B","end\_gradient\_bg\_color":"#C43C35"},  {"text\_color":"#000","bg\_color":"#e4c904","start\_gradient\_bg\_color":"#e3c703","end\_gradient\_bg\_color":"#eed909"},  {"text\_color":"#ffffff","bg\_color":"#6aa506","start\_gradient\_bg\_color":"#3d9913","end\_gradient\_bg\_color":"#65b92c"},  {"text\_color":"#ffffff","bg\_color":"#237ccb","start\_gradient\_bg\_color":"#2078ca","end\_gradient\_bg\_color":"#3a98d1"}]} |

The insertion of custom metadata is optional, as default values will be used instead.

##### 7.1.3.4 Configuring syslog

A optional rsyslog configuration file, **/etc/rsyslog.d/wgen.conf**, can be created by an operator. All regular files in the /etc/rsyslog.d directory are included as additional rsyslog server settings for **/etc/rsyslogd.conf**. If syslog local facilities are not specified, they are sent to the syslog message facility.

$FileCreateMode 0644  
local0.\* /opt/edware/log/audit.log  
local1.\* /opt/edware/log/smarter.log  
local2.\* /opt/edware/log/security\_event.log

##### 7.1.3.5 Creating a Database Schema for Production Database

You must manually create an empty schema for the production database. A script is provided to perform this task. This step requires that the Database Master is installed and configured first.

shell> . /opt/virtualenv/smarter/bin/activate

(virtualenv) cd /opt/virtualenv/smarter/lib/python3.3/site-packages/edschema-0.1-py3.3.egg/edschema

(virtualenv) python metadata\_generator.py -s edware -d edware -m edware --host [dbMastHostName] -p [password]

**Note**: The command is in the format of:   
python metadata\_generator.py -s [schemaName] -d [databaseName] -m edware --host [dbMastHostName] -p [password]

## 7.2 Load Balancer

Load Balancers are used to load balance traffic to web servers. In AWS, Elastic Load Balancers are used for this purpose.

**7.3 PDF Messenger [Ansible name: reporting-rabbit-services]**

PDF Messenger hosts the broker for PDF tasks that are requested by the Smarter Web Application. We have chosen to use RabbitMQ as the message broker. RabbitMQ, written in Erlang, implements the Advanced Message Queuing Protocol (AMQP) standard.

### 7.3.1 Installation

The following table lists the RPMs that must be installed on the PDF Messenger server.

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| RabbitMQ | 2.6.1 | rabbitmq-server | RabbitMQ RPM is available from Extra Packages for Enterprise Linux (EPEL) |

### 7.3.2 Configuration

RabbitMQ

The RabbitMQ RPM comes with default built-in configurations which are sufficient for running the service effectively. We recommend using RabbitMQ cluster to prevent a single point of failure. Please see <http://www.rabbitmq.com/clustering.html> for more details.

The main configuration file for RabbitMQ is located in **/etc/rabbitmq/rabbitmq.config**. Please refer to [http://www.rabbitmq.com/configure.html for more details.](http://www.google.com/url?q=http%3A%2F%2Fwww.rabbitmq.com%2Fconfigure.html&sa=D&sntz=1&usg=AFQjCNHnXaDawyanlReNV9jxIKfZ008MEg)

Please remember that any configurations set here (such as user name, passwords, etc.) needs to be reflected back in your **smarter.ini** file in your web server, pdf worker, and pdf generator machines.

Adding a New User

A user needs to be created to authenticate with applications with RabbitMQ. To add a new user, execute the commands below with root access, please refer to **smarter.ini** for the user name and password. We default to the user **‘edware’** and password **‘edware1234’**.

shell> rabbitmqctl add\_user <user> <password>

Adding a New vhost

A new virtual host needs to be added so that PDF Generator has a separation with other applications that may utilize the same broker. By default, Smarter Web Applications expects this virtual host to be named **services**. To add a new virtual host, execute the following command below with root access:

shell> rabbitmqctl add\_vhost services

Granting User Permission

The user that you have created above needs to have permission to your virtualhost. To grant permission, execute the following command below, replacing <user> with the one with proper permission:

shell> rabbitmqctl set\_permissions -p services <user> ".\*" ".\*" ".\*"

Enabling SSL (Optional)

If you decide to run RabbitMQ on SSL, you will need to enable Celery on SSL in your PDF Worker (as described later).

*Generate self signed certificate and key with OpenSSL*

You will need to generate the certificate with the following commands:

shell> cd ~/

shell> mkdir testca

shell> mkdir certs private

shell> chmod 700 private

shell> echo 01 > serial

shell> touch index.txt

shell> vi testca/openssl.cnf

[ ca ]

default\_ca = testca

[ testca ]

dir = .

certificate = $dir/cacert.pem

database = $dir/index.txt

new\_certs\_dir = $dir/certs

private\_key = $dir/private/cakey.pem

serial = $dir/serial

default\_crl\_days = 7

default\_days = 365

default\_md = sha1

policy = testca\_policy

x509\_extensions = certificate\_extensions

[ testca\_policy ]

commonName = supplied

stateOrProvinceName = optional

countryName = optional

emailAddress = optional

organizationName = optional

organizationalUnitName = optional

[ certificate\_extensions ]

basicConstraints = CA:false

[ req ]

default\_bits = 2048

default\_keyfile = ./private/cakey.pem

default\_md = sha1

prompt = yes

distinguished\_name = root\_ca\_distinguished\_name

x509\_extensions = root\_ca\_extensions

[ root\_ca\_distinguished\_name ]

commonName = hostname

[ root\_ca\_extensions ]

basicConstraints = CA:true

keyUsage = keyCertSign, cRLSign

[ client\_ca\_extensions ]

basicConstraints = CA:false

keyUsage = digitalSignature

extendedKeyUsage = 1.3.6.1.5.5.7.3.2

[ server\_ca\_extensions ]

basicConstraints = CA:false

keyUsage = keyEncipherment

extendedKeyUsage = 1.3.6.1.5.5.7.3.1

*Generate self-signed CA*

You will need to generate a self-signed certificate authority with the following command:

shell> openssl req -x509 -config openssl.cnf -newkey rsa:2048 -days 365 -out cacert.pem -outform PEM -subj /CN=MyTestCA/ -nodes

*Generate Server Certificate*

You will need to generate a server certificate with the following commands:

shell> cd ~/

shell> mkdir server

shell> cd server

shell> openssl genrsa -out key.pem 2048

shell> openssl req -new -key key.pem -out req.pem -outform PEM -subj /CN=$(hostname)/O=server/ -nodes

shell> cd ../testca

shell> openssl ca -config openssl.cnf -in ../server/req.pem -out ../server/cert.pem -notext -batch -extensions server\_ca\_extensions

*Generate Client Certificate*

You will need to generate a client certificate, which be installed by clients such as PDF Worker:

shell> cd ~/

shell> mkdir client

shell> cd client

shell> openssl genrsa -out key.pem 2048

shell> openssl req -new -key key.pem -out req.pem -outform PEM -subj /CN=$(hostname)/O=client/ -nodes

shell> cd ../testca

shell> openssl ca -config openssl.cnf -in ../client/req.pem -out ../client/cert.pem -notext -batch -extensions client\_ca\_extensions

*Configure and Enable SSL on RabbitMQ*

1. Update /etc/rabbitmq/rabbitmq.config:  
     
   [

{rabbit, [

{tcp\_listeners, []},

{ssl\_listeners, [5671]},

{ssl\_options, [{cacertfile,"/etc/rabbitmq/testca/cacert.pem"},

{certfile,"/etc/rabbitmq/server/cert.pem"},

{keyfile,"/etc/rabbitmq/server/key.pem"},

{verify,verify\_peer},

{fail\_if\_no\_peer\_cert,false}]}

]}]

1. Restart the rabbitmq server:  
     
   /etc/rc.d/init.d/rabbitmq-server restart
2. Configure the Firewall, change **/etc/sysconfig/iptables**, add the following lines before the REJECT rules:

-A INPUT -p tcp -m state --state NEW -m tcp --dport 5671 -j ACCEPT

-A INPUT -p tcp -m state --state NEW -m tcp --dport 15672 -j ACCEPT

## 7.4 PDF Worker [Ansible name: reporting-worker-pdf]

PDF Worker is responsible for producing PDF versions of reports. The worker picks up messages/tasks from the PDF Messenger via RabbitMQ. PDF Worker is written using a Python framework, Celery. Celery is an asynchronous task queue/job queue based on distributed message passing. PDFs are stored on a volume on glusterFS encrypted with EncFS.

### 7.4.1 Installation

PDF Messenger, internally known as celeryd-services, is packaged inside the Smarter RPM, hence, the prerequisites for Smarter is also required in PDF Worker servers.

The following table lists the RPMs that must be installed on PDF Worker servers:

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| Python | 3.3 | python3-3.3.0 | Python 3 RPM is generally not available from Linux distributions, hence Amplify has supplied a customized Python 3 RPM |
| WSGI | 3.4 | python3-mod\_wsgi | WSGI defines a simple and universal interface between web servers and web applications or frameworks for Python. This is a customized RPM supplied by Amplify. |
| Smarter | TBD | smarter | Amplify’s Smarter Web Application |
| XMLSec | 1.2.16 | xmlsec1  xmlsec1-openssl-devel | XMLSec can be installed from standard distributed RPM. It is used for verification of SAML Responses. |
| wkhtmltopdf | 0.12.1 | wkhtmltox | A third party utility for PDF generation of reports using webkit rendering engine and Qt. |
| PDFUnite | 0.26.1 | pdfunite | A third party utility for merging PDFs.    Amplify provides a customized RPM for PDFUnite. |
| urw-fonts | 2.4 | urw-fonts | Free versions of 35 standard PostScript fonts.    This RPM is readily available from Linux Distributions. |
| PDF Fonts | <placeholder> | <placeholder> | Fonts for other languages are required so that PDFs are generated properly for such languages. |
| glusterfs | 3.3.1 | glusterfs | We suggest installing a newer version of the RPM from GlusterFS’ site other than the RPM from Linux distributors.    See below for more details. |
| glusterfs-fuse | 3.3.1 | glusterfs-fuse | It provides support to FUSE based clients.    We suggest installing a newer version of the RPM from GlusterFS’ site other than the RPM from Linux distributors.    See below for more details. |
| encfs | 1.7.4 | fuse-encfs | encFS client to encrypt a volume    This RPM is readily available from Linux Distributions.    Not Required for AWS |

Installing GlusterFS

GlusterFS is a clustered file-system capable of scaling to several petabytes. As noted above, we recommend installing a newer version of the glusterfs RPM directly from GlusterFS’ website.

Setting up yum repo configuration for GlusterFS

Please execute the following command:

shell> wget -P /etc/yum.repos.d http://download.gluster.org/pub/gluster/glusterfs/LATEST/EPEL.repo/glusterfs-epel.repo

### 

### 7.3.2 Configuration

#### 7.3.2.1 GlusterFS

At this step, it’s expected that you have a gluster that is ready to be mounted. You will need to configure PDF worker servers to automount the gluster.

Automount the Gluster

Modify **/etc/fstab** to automount when the OS boots up.

glusterServer.example.net:/gv0 /mnt/gluster glusterfs defaults 1 2

There is a network timing issue with automount when the OS boots up. To work around this issue, use **rc.local** to mount GlusterFS. Append the following line to **/etc/rc.d/rc.local**:

mount -a

#### 7.3.2.3 celeryd-services

Similar to the Smarter Web Application, PDF Workers read the same **.ini** file (located in /opt/edware/conf/smarter.ini). Please refer [here](#h.ki3mskb1afgm) to generate the **.ini** file.

Changing Celery process’ user/group (Optional)

If you need to run the celery process with a different user or group other than the default, you will need to modify **/opt/edware/conf/celeryd-services.conf**.

You’ll need to modify the values for CELERYD\_USER and/or CELERYD\_GROUP.

Changing PDF Worker’s Celery Configuration (Optional)

In **/opt/edware/conf/smarter.ini**, the following configurations are relevant and most likely needs to be changed in PDF Worker:

|  |  |  |
| --- | --- | --- |
| **Configuration Name** | **Description** | **Example Value** |
| services.celery.BROKER\_URL | The Broker URL that PDF Worker will bind to | amqp://edware:edware1234@pdfMessengerHost/services |

Enabling SSL (Optional)

This section is required only if you’ve enabled SSL on your PDF Messenger.

*Copy Client Certificates*

Copy the client certificates and **testca** generated from [here](#h.t6l1h8n3i7sb) and place the CA to **/opt/edware/conf/testca** and certificates to **/opt/edware/conf/client**.

*Configure Celery to use SSL*

1. Update broker URL to RabbitMQ with the SSL port number:  
     
   services.celery.BROKER\_URL = amqp://user:pwd@broker:5671/services
2. Add the following lines into **smarter.ini**:  
     
   services.celery.BROKER\_USE\_SSL = {'ca\_certs': '/opt/edware/conf/testca/cacert.pem', 'keyfile': '/opt/edware/conf/client/key.pem', 'certfile': '/opt/edware/conf/client/cert.pem', 'cert\_reqs': True }
3. Restart celeryd:

shell> /etc/init.d/celeryd-services restart

## 7.4 PDF Pre-Generator [Ansible name: reporting-generator-pdf]

PDF Pre-Generator is used to pre-generate PDFs when new data batches are loaded into the system. This mechanism is done via a cron job that is scheduled to run and check the database for newly ingested data. We can consider this mechanism as a backend Smarter Web Application. Its sole responsibility is to trigger PDF pregeneration on a scheduled basis.

PDF Generator needs to write and update the **edware\_stats** database, therefore the database connection must be directly to the database master. We recommend this database to be on the Loader Database server.

### 7.4.1 Installation

Please refer to [here](#h.iz87h079gnc7) for installing PDF Pre-Generator server. The install is almost identical to installing and configuring web servers, except that wsgi needs to be configured to be single process. This is described in the configuration section.

### 7.4.2 Configuration

#### 7.4.2.1 smarter

Generating Smarter.ini

Similarly, PDF Generator needs to read from the **.ini** file. Please refer to [here](#h.ki3mskb1afgm) for generating this file.

Note: In **settings.yaml**, you can check if there exists a section dedicated to this server type. To generate the **.ini** for PDF Generator, you can/should set the environment value as ‘**uat.pdf\_generator**’.

ex. python generate\_ini.py -e uat.pdf\_generator

Configurations specific to PDF Generator in Smarter.ini

|  |  |  |
| --- | --- | --- |
| **Configuration** | **Description** | **Example Value** |
| cache.session.expire | The special pdf generation batch user’s session expiration in seconds. This should be set to the value that you expect the duration of the PDFs to be generated for the current batch of data. | 300000 |
| batch.user.session.timeout | The special pdf generation batch user’s cookie expiration in seconds. This should align with the session expiration. | 300000 |
| trigger.pdf.enable | Enable pdf pre-generation | True |
| trigger.pdf.schedule.cron.hour | The hour in which to schedule pdf-generation | 20 |
| trigger.pdf.schedule.cron.minute | The minute in which to schedule pdf-generation | 15 |

#### 7.4.2.2 WSGI

The configuration for wsgi is almost identical to the configuration of wsgi in web servers. Please refer to that section for the configuration needed. The only configuration difference is described below.

Configuration for Smarter application specific WSGI configuration

Modify **/etc/httpd/conf.d/wsgi\_edware.conf** to add the WSGI configuration for PDF Pre-Generator. The notable difference compared to web server is that we’re running on a single process.

WSGIApplicationGroup %{GLOBAL}

WSGIPassAuthorization On

WSGIDaemonProcess pyramid user=apache group=apache processes=1 threads=30 python-path=/opt/virtualenv/lib/python3.3/site-packages

WSGIScriptAlias / /opt/edware/smarter/smarter.wsgi

WSGIImportScript /opt/edware/smarter/smarter.wsgi process-group=pyramid application-group=%{GLOBAL}

WSGISocketPrefix run/wsgi

<Directory /opt/virtualenv>

WSGIProcessGroup pyramid

Order allow,deny

Allow from all

</Directory>

WSGIPythonPath /opt/virtualenv/lib/python3.3/site-packages

## 7.5 Extract Messenger [Ansible name: reporting-rabbit-extract]

Extract Messenger hosts a message system between Smarter Web Application and Extract Worker. It uses RabbitMQ as its message broker and extract tasks are sent from Web Servers and are received by Extract Workers. The installation and configuration is very similar to PDF Messenger.

### 7.5.1 Installation

Please refer to the installation of RabbitMQ from [here](#h.22urj4l615b5).

**Note:**  The only configuration difference is that the name of the virtual host for extracts should be different than the one defined for PDF Messenger. By default, we expect and recommend the virtual host for Extract Messenger to be named, **edextract**.

### 7.5.2 Configuration

Please refer to the configuration of RabbitMQ from [here](#h.2hqcoe7s51rj).

## 7.6 Extract Worker [Ansible name: reporting-worker-extract]

Extract worker is responsible for receiving extract tasks from the queue and generating bulk raw extracts in CSV format. The installation and configuration is very similar to PDF Worker.

There are a few key differences for Extract Worker:

* Celery service name is **celeryd-edextract**.
* Celery configuration file is **celeryd-edextract.conf**.
* Mount EncFS to **/opt/edware/extraction** (be sure to mount it as celery user).
* **extract.celery.BROKER\_URL** is the broker URL used by Extract Worker.

### 7.6.1 Installation

Please refer to the installation of Extract Worker from [here](#h.a9cm003zx56q).

**Note:**  Please remember the notable installation differences between PDF and Extract Worker, namely, the celery service.

### 7.6.2 Configuration

Please refer to the configuration of Extract Worker from [here](#h.hrusrg3lgvr9).

**Note:**  Please remember the notable configuration differences between PDF and Extract Worker, namely, the celeryd configuration file, the EncFS mount point, and the broker URL configuration in smarter.ini.

## 7.7 Cache [Ansible name: memcached]

The Smarter Web Application uses cache servers to persist data for reports and user sessions for some configurable amount of duration. It uses memcached, an in-memory key-value store for small chunks of arbitrary data (strings, objects).

### 7.7.1 Installation

The following table lists the RPMs that must be installed on cache servers.

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| memcached | 1.4.4 | memcached | Available on standard Linux Distributions |

### 7.7.2 Configuration

memcached

The standard install of memcached should be sufficient.

Main configuration for memcached

Modify **/etc/sysconfig/memcached** for main configuration for memcached:

**PORT="11211"  
USER="memcached"  
MAXCONN="1024"  
CACHESIZE="64"  
OPTIONS=""**

Please update MAXCOMM and CACHESIZE based on your memchaced server configuration.

## 7.8 Cache Warmer [ansible name: reporting-cache-warmer]

The Cache Warmer re-populates memcached used by the Application Server for results of popular data requests. The Cache Warmer is triggered on a schedule. The data requests are configurable for the types of demographics filters to apply to the data requests.

Similar to PDF Pre-Generator, Cache Warmer is considered as an offline Smarter Web Application. It’s a dedicated server used to handle the flushing and re-caching of reports data when new data has been ingested into the system.

### 7.8.1 Installation

Please refer to the installation guide from [here](#h.8jsw6vl01tyx). The install is almost identical to installing and configuring web servers, except that the wsgi needs to be configured to be a single process. This is described in configuration section.

### 7.8.2 Configuration

#### WSGI

The configuration for wsgi is almost identical to the configuration of wsgi in web servers. Please refer to that section for the configurations needed. The only configuration difference is described below.

##### Configuration for Smarter application specific WSGI configuration

Modify **/etc/httpd/conf.d/wsgi\_edware.conf** to add WSGI configuration for the Cache Warmer. The notable difference compared to web server is that we’re running on a single process.

WSGIApplicationGroup %{GLOBAL}

WSGIPassAuthorization On

WSGIDaemonProcess pyramid user=apache group=apache processes=1 threads=30 python-path=/opt/virtualenv/lib/python3.3/site-packages

WSGIScriptAlias / /opt/edware/smarter/smarter.wsgi

WSGIImportScript /opt/edware/smarter/smarter.wsgi process-group=pyramid application-group=%{GLOBAL}

WSGISocketPrefix run/wsgi

<Directory /opt/virtualenv>

WSGIProcessGroup pyramid

Order allow,deny

Allow from all

</Directory>

WSGIPythonPath /opt/virtualenv/lib/python3.3/site-packages

#### smarter

##### Generating Smarter.ini

Similarly, Cache Warmer needs to read from **.ini** file. Please refer to [here](#h.ki3mskb1afgm) for generating this file.

Note: In settings.yaml, you should check if there exists a section dedicated to this server type. To generate **.ini** for Cache Warmer, you can/should set the environment value as ‘uat.cache.

ex. python generate\_ini.py -e uat.cache

##### Configurations specific to Cache Warmer in Smarter.ini

|  |  |  |
| --- | --- | --- |
| **Configuration** | **Description** | **Example Value** |
| trigger.recache.enable | Enable cache warmer | True |
| trigger.recache.schedule.cron.hour | The hour in which to schedule cache warmer | 20 |
| trigger.recache.schedule.cron.minute | The minute in which to schedule cache warmer | 15 |

##### Configuration for Demographics Filters

Cache Warmer re-caches comparing populations reports, and the demographic filters for the report configuration. The state and district comparing populations reports are both candidates for caching, the configuration file contains state and district specific sections.

Modify **/opt/edware/conf/comparing\_populations\_precache\_filters.json** to configure filters specific to your tenant/state. You can override a particular tenant’s configuration by adding a section in the JSON file by prepending the tenant name in front of state or district.

Ex. The following configuration has 2 filters for State View, 1 filter for District View, and overwrites ES tenant State View with a different filter.

{“state”:

[{“grade: [“3”], “dmgPrgIep”: [“Y”]},

{“grade”: [“4”]}],

“district”: [{“grade”: [“5”]}],

“ES.state”: [{“grade”: [“6”]}]}

## 7.9 Database Master [ansible name: db-master]

Database Master server is used for storing all reporting data. Database Master allows insert/update/delete/select queries. Master sends replication data to one or more standby servers.

In order to protect PII, database data should be encrypted. Please see the [Encrypting](#h.b9liv75bsgd0) section to prepare to encrypt your data in PostgreSQL.

### 7.9.1 Installation

The following lists the RPM required by Database Master

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| Database | 9.2.8 | postgresql  postgresql92-server  postgresql92-libs  postgresql92-contrib  postgresql92-devel | These RPMs are available from most Linux distributions |
| Replication Manager | 2.0 | repmgr | RPM is available from most Linux distributions. |

#### 

**7.9.2 Configuration**

#### postgres

##### Configuring postgres master

1. If this is the first time you install postgres, run:

shell> service postgresql-92 initdb

1. We need to set up an archive directory for the WAL log for replication, and put the right archive directory into archive\_command options. For example in the runbook we have **/var/lib/pgsql/9.2/archive** as archive directory.
2. Generate an ssh key pair on master. Executing the following commands as the **postgres** user. (repmgr program will copy the database with rsync and ssh for user postgres, so we need to set up ssh login with password between master and slave servers).  
   shell> ssh-keygen  
   Generating public/private rsa key pair.  
   Enter file in which to save the key (/var/lib/pgsql/.ssh/id\_rsa):   
   Created directory '/var/lib/pgsql/.ssh'.  
   Enter passphrase (empty for no passphrase):   
   Enter same passphrase again:   
   Your identification has been saved in /var/lib/pgsql/.ssh/id\_rsa.  
   Your public key has been saved in /var/lib/pgsql/.ssh/id\_rsa.pub.  
   The key fingerprint is:  
   58:66:3b:d1:97:f4:b9:fc:1f:66:dc:28:3f:8f:bf:8f postgres@dbpgdw0.qa.dum.edwdc.net  
   The key's randomart image is:  
   +--[ RSA 2048]----+  
   | . |  
   | . . o . |  
   | = . o o |  
   | = o . . . |  
   | . S o |  
   | . .o.|  
   | . .=o|  
   | o+oo|  
   | E=B|  
   +-----------------+
3. Distribute public key to postgresql database replica servers:  
   Append generated public key(**/var/lib/pgsql/.ssh/id\_rsa.pub**) to replica servers (**/var/lib/pgsql/.ssh/authorized\_keys**).
4. Modify **/var/lib/pgsql/9.2/data/postgresql.conf**

Update the following options. Keep the other options untouched.

|  |  |  |
| --- | --- | --- |
| **Configuration** | **Description** | **Example Value** |
| listen\_addresses | address for postgres server, we use all IP on the server. so ‘\*’ | ‘\*’ |
| shared\_buffers | This depends on your server memory. See postgres manual for details. | 8192MB |
| work\_mem | work\_mem for each connection, | 100MB |
| log\_filename | file name for log file | ‘postgresql-%a.log’ |
| log\_truncate\_on\_rotation |  | on |
| log\_rotation\_size |  | 0 |
| log\_timezone | Time Zone for log file. We use ‘UTC’ | ‘UTC’ |
| timezone | Time Zone | ‘UTC’ |
| wal\_level | We are doing replication. So host\_standbay | hot\_standby |
| checkpoint\_segments |  | 30 |
| archive\_mode | We need to run replication. so we set it on | on |
| max\_wal\_senders | number of processes to send WAL log to remote | 10 |
| wal\_keep\_segments | WAL files that are kept under pg\_xlog before it is archived. | 5000 |
| hot\_standby |  | on |
| ssl | enable ssl | on |
| archive\_command | command to be executed by celery to move WAL log to archive directory. We prefer keep pg\_xlog small and move compress WAL log to archive and delete the original WAL log under pg\_xlog | 'gzip -9 < %p > /var/lib/pgsql/9.2/archive/%f && rm %p' |

1. Update **/var/lib/pgsql/9.2/data/pg\_hba.conf**, add the following lines into pg\_hba.conf, keeping the original pg\_hba.conf content:

local all all peer

# IPv4 local connections:

hostssl all all 127.0.0.1/32 trust

# Replication:

hostssl replication all ###.###.###.###/# trust

hostssl all all ###.###.###.###/# md5

# IPv6 local connections:

hostssl all all ::1/128 ident

###.###.###.### is your postgresql slave server IP/network.

1. Create the **edware** user and database at the psql prompt:  
   CREATE DATABASE edware with encoding 'UTF-8';

CREATE DATABASE edware\_stats with encoding 'UTF-8';  
CREATE USER edware WITH PASSWORD 'edware2013';  
GRANT ALL PRIVILEGES ON DATABASE edware to edware;

GRANT ALL PRIVILEGES ON DATABASE edware\_stats to edware;

1. Create **/var/lib/pgsql/repmgr.conf**:

|  |  |  |
| --- | --- | --- |
| **Configuration** | **Description** | **Example Value** |
| cluster | group of cluster name. Use the same name for all servers. | edware\_pg\_cluster |
| node | node number must be unique for each servers. | 1 |
| node\_name | value must be its own server hostname | fully qualified domain name of the machine |
| conninfo | repmgr to connect to PostgreSQL server, host value must be its server host name, and the proper user name and database name for replicated database | 'host=<my\_FQDN\_.com> user=repmgr dbname=edware' |
| pg\_bindir | path to pgsql installation | /usr/pgsql-9.2/bin |

1. Create the **repmgr** user for PostgreSQL at the psql prompt:  
   CREATE ROLE repmgr SUPERUSER LOGIN;
2. Register **repmgr**:

PATH=$PATH:/usr/pgsql-9.2/bin repmgr -f /var/lib/pgsql/repmgr.conf master register

1. Restart PostgreSQL.

## 7.10 Database Replica

Database Replica servers are responsible for replicating data from the Database Master. The server is a read-only server, therefore, only SELECT queries can be executed.

In order to protect PII, database data should be encrypted. Please see the [Encrypting](#h.b9liv75bsgd0) section to prepare to encrypt your data in PostgreSQL.

In Smarter Balanced Reporting, we allocated ⅓ of the replica servers for data extract use and ⅔ replica servers for web frontend reporting. The data extraction replica server doesn’t require smarter to be installed.

### 7.10.1 Installation

The following lists the RPMs required by Database Replica for web frontend reporting:

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| Database | 9.2.8 | postgresql  postgresql92-server  postgresql92-libs  postgresql92-contrib  postgresql92-devel | These RPMs are available from most Linux distributions. |
| Replication Manager | 2.0 | repmgr | RPM is available from most Linux distributions. |
| smarter | <TBD> | smarter | An Amplify supplied RPM. edmigrate-celerdy is package within smarter RPM. |

The following lists the RPMs required by the Database Replica for extract:

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| Database | 9.2.8 | postgresql  postgresql92-server  postgresql92-libs  postgresql92-contrib  postgresql92-devel | These RPMs are available from most Linux distributions. |
| Replication Manager | 2.0 | repmgr | RPM is available from most Linux distributions. |

**7.10.2 Configuration**

#### postrgres

1. Generate an ssh key pair on all slave servers. Executing commands must be the **postgres** user. (The repmgr program will copy the database with rsync and ssh for the postgres user, so we need to set up ssh login with password between the master and slave servers).  
   shell> ssh-keygen  
   Generating public/private rsa key pair.  
   Enter file in which to save the key (/var/lib/pgsql/.ssh/id\_rsa):   
   Created directory '/var/lib/pgsql/.ssh'.  
   Enter passphrase (empty for no passphrase):   
   Enter same passphrase again:   
   Your identification has been saved in /var/lib/pgsql/.ssh/id\_rsa.  
   Your public key has been saved in /var/lib/pgsql/.ssh/id\_rsa.pub.  
   The key fingerprint is:  
   58:66:3b:d1:97:f4:b9:fc:1f:66:dc:28:3f:8f:bf:8f postgres@dbpgdw0.qa.dum.edwdc.net  
   The key's randomart image is:  
   +--[ RSA 2048]----+  
   | . |  
   | . . o . |  
   | = . o o |  
   | = o . . . |  
   | . S o |  
   | . .o.|  
   | . .=o|  
   | o+oo|  
   | E=B|  
   +-----------------+
2. Copy the public key to the postgresql Master Server:  
   Append the generated public key (**/var/lib/pgsql/.ssh/id\_rsa.pub**) to the Master Server (**/var/lib/pgsql/.ssh/authorized\_keys**).
3. Stop PostgreSQL.
4. Make sure the data directory is empty. Delete all files and directories:  
   cd /var/lib/pgsql/9.2/data  
   rm -rf \*
5. Clone the master database:
   1. This must be the **postgres** user. To execute, replace **master.server.com** to your master database’ fully qualified domain name:  
      shell> su - postgres  
      shell> PATH=$PATH:/usr/pgsql-9.2/bin repmgr -D /var/lib/pgsql/9.2/data -d edware -p 5432 -R postgres --verbose standby clone master.server.com
6. After the clone, modify **recovery.conf**. To add the following two lines, you need to change the archive path to what you have on your replica server:

restore\_command = 'gunzip < /var/lib/pgsql/9.2/archive/%f.gz > %p'

archive\_cleanup\_command = 'pg\_archivecleanup /var/lib/pgsql/9.2/archive %r'

1. After the clone, if there are files under the archive directory in the master database - **scp** them under the **postgres** user to the replicated servers. Here is an example copy from master to slave replica’s archive directory:

shell> scp \* postgres@[fqdn of replica]:/var/lib/pgsql/9.2/archive/

1. Update **pg\_hba.conf**, make sure the following line is in **pg\_hba.conf**:

host all all 127.0.0.1/32 trust

1. Start the slave database after modifying **postgresql.conf**.
2. Create **/var/lib/pgsql/repmgr.conf**:

|  |  |  |
| --- | --- | --- |
| **Configuration** | **Description** | **Example Value** |
| cluster | group of cluster name. Use the same name for all servers. | edware\_pg\_cluster |
| node | node number must be unique for each servers. | 1 |
| node\_name | value must be its own server hostname | fully qualified domain name of the machine |
| conninfo | repmgr to connect to PostgreSQL server, host value must be its server host name, and the proper user name and database name for replicated database | 'host=<my\_FQDN\_.com> user=repmgr dbname=edware' |
| pg\_bindir | path to pgsql installation | /usr/pgsql-9.2/bin |

1. Make sure **/var/log/repmgrd** and **/var/run/repmgrd** exist and are owned by the **postgres** user.
2. Register **repmgr** for **postgres** use:  
   PATH=$PATH:/usr/pgsql-9.2/bin repmgr -f /var/lib/pgsql/repmgr.conf standby register
3. Start replication:
   1. If using **repmgr 1.2** then ***sudo*** as **postgres**:  
      PATH=$PATH:/usr/pgsql-9.2/bin repmgrd -f /var/lib/pgsql/repmgrdx.conf
   2. Add **repmgrd**:  
      chkconfig --add repmgrd
   3. If using **repmgr** - then run as ***root***:  
      service repmgrd start
4. Grant privilege to **repmgr\_edware\_pg\_cluster** to the **edware** user:  
   shell> su - postgres  
   shell> psql -d edware  
   edware=# grant usage on schema repmgr\_edware\_pg\_cluster to edware;

edware=# set search\_path to repmgr\_edware\_pg\_cluster;

edware=# GRANT ALL PRIVILEGES ON TABLE repl\_monitor to edware;

edware=# GRANT ALL PRIVILEGES ON TABLE repl\_nodes to edware;

edware=# GRANT ALL PRIVILEGES ON TABLE repl\_status to edware;

#### celeryd-edmigrate

**celeryd-edmigrate** is a service used for our migration process. The role of replicas in the migration process is known as a player. We only need to enable **celeryd-edmigrate** for Database replicas that are used for the Smarter Balanced Reporting Web Front End.

**Configure iptables for celeryd-edmigrated**

1. Reset **iptables** rules.  
   shell> service iptables stop
2. Create the minimum **iptables** rules:  
   shell> /sbin/iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT  
   shell> /sbin/iptables -A INPUT -p tcp --dport 22 -j ACCEPT  
   shell> /sbin/iptables -A INPUT -p tcp --dport 5432 -j ACCEPT  
   shell> /sbin/iptables -P INPUT DROP
3. Create new user-defined custom chain “**EDMIGRATE\_PGSQL**”:  
   shell> /sbin/iptables -N EDMIGRATE\_PGSQL
4. Create a new rule for the chain:  
   shell> /sbin/iptables -A EDMIGRATE\_PGSQL -p tcp -d 127.0.0.1 --dport 5432 -j ACCEPT  
   shell> /sbin/iptables -A EDMIGRATE\_PGSQL -p tcp --dport 5432 -j REJECT
5. Save the **iptables** rules. Caution: when you execute the following command - it will overwrite any existing **iptables** saved settings.  
   shell> service iptables save  
   or  
   shell> iptables-save > /etc/sysconfig/iptables

**Configure sudoers for celeryd-edmigrate**

* + - 1. Run ***visudo***:  
         shell> visudo
      2. Add the following lines. Caution: the second line is a single line:

Defaults:celery !requiretty  
celery ALL=NOPASSWD: /sbin/iptables -t filter -I INPUT -j EDMIGRATE\_PGSQL, /sbin/iptables -t filter -D INPUT -j EDMIGRATE\_PGSQL, /sbin/iptables -t filter -I OUTPUT -j EDMIGRATE\_PGSQL, /sbin/iptables -t filter -D OUTPUT -j EDMIGRATE\_PGSQL, /sbin/iptables-save

**Configure Player**

* + - 1. Register the **celery** task:

shell> chkconfig --add celeryd-edmigrate

* + - 1. Start the service:

shell> service celeryd-edmigrate start

* + - 1. Stop the service:

shell> service celeryd-edmigrate stop

## 7.11 Database Load Balancer

**PgBouncer** is a lightweight connection pooler for PostgreSQL. The Smarter web application uses **PgBouncer** in order to handle massive database connections by web service requests efficiently.

### 7.11.1 Installation

The following lists the RPMs required by the Database Load Balancer:

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| Database Middleware | 1.5.4 | pgbouncer | RPM is available from most Linux distributions. |

**7.11.2 Configuration**

1. Edit **/etc/pgbouncer/pgbouncer.ini**.Modify the following options and keep the others unchanged:  
   [databases]  
   \* = host=dwrouter1.qa.dum.edwdc.net port=9999 user=edware password=edware2013 pool\_size=450 connect\_query='select 1'  
     
   [pgbouncer]  
   logfile = /var/log/pgbouncer/pgbouncer.log  
   pidfile = /var/run/pgbouncer/pgbouncer.pid  
   listen\_addr = \*  
   listen\_port = 6432  
   auth\_type = md5  
   auth\_file = /etc/pgbouncer/userlist.txt  
   admin\_users = user2, postgres  
   stats\_users = stats, postgres  
   pool\_mode = session  
   server\_reset\_query = DISCARD ALL  
   ignore\_startup\_parameters = client\_min\_messages   
   max\_client\_conn = 500  
   default\_pool\_size = 100  
   reserve\_pool\_size = 10  
   log\_connections = 0  
   log\_disconnections = 0  
   log\_pooler\_errors = 1  
   query\_timeout = 180
2. Create **/etc/pgbouncer/userlist.txt**  
   PgBouncer requires its own userlist file. The client authenticates with PgBouncer service first, this is independent of the PostgreSQL Database Authentication. The authentication file contains pairs of double-quote enclosed username and passwords that a client application uses to access PgBouncer. The location of **userlist.txt** is specified at **auth\_file** in **/etc/pgbouncer/pgbouncer.ini**.
   1. "postgres" "md5cf8e80c6852c634a6e00613455d34189"  
      "edware" "md50927d04170fc5ebc2a14e662d5425c9c"
   2. Encrypting the password  
      The authentication file takes both clear text passwords and MD5-encrypted passwords by settings in pgbouncer.ini (**auth\_type**). For security reasons, we strongly encourage to use of MD5-encrypted passwords.
      1. How to generate MD5-encrypted password  
         From PostgreSQL Query Prompt  
         select 'md5' || md5 ('edware2013' || 'edware');

Copy the resulting string into the userlist.txt file:  
“edware" "md5d305b538896b9a9ea5086cc126bcc09f"

1. Log file  
   The location of log file can be specified in pgbouncer.ini (**logfile**).

## 7.12 Database Pool [ansible name: reporting-db-pgpool]

Pgpool-II is a middleware that works between PostgreSQL servers and PostgreSQL database client. Pgpool-II is mainly used by Smarter to load balance the distribution of SELECT queries among multiple servers, thus improving the system’s overall throughput.

In Smarter Balanced Reporting. We have two sets of Database Pool. One set of Database Pool is for Smarter Balanced Reporting Web Front End. Another set of Database Pool is for Smarter Balanced Bulk Extract and Bulk Printing.

### 7.12.1 Installation

The following lists the RPMs required by Database Pool

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| Database | 9.2.8 | postgresql92-server  postgresql92-libs  postgresql92 | RPM for postgres user for pgpool |
| Database Middleware | 3.3.1 | pgpool-II | RPM is available from most Linux distributions. |

**7.12.2 Configuration**

#### pgpool for Smarter Balanced Reporting web frontend

1. Make sure postgres user exists
2. Make sure firewall is opened for port 9999 for pgpool.
3. Disable postgresql server running

shell> service postgresql-9.2 stop

1. Make sure /var/run/pgpool is owned by postgres user
2. Edit **/etc/pgpool-II/pool\_hba.conf,** Add following to allow access to pgpool.

host all all ###.###.###.###/## trust  
host replication all ###.###.###.###/## trust

host all all 127.0.0.1/32 trust

###.###.###.### is your pgbouncer server IP/network.

1. Edit **/etc/pgpool-II/pcp.conf**Configuration for pcp.conf user. We need to use pcp\_attach\_node command to recover pg\_pool connection during edmigrate.
   1. generate md5 password  
      # USERID:MD5PASSWD  
      postgres:{the md5 passphrase from postgres}
2. Edit **/etc/pgpool-II/pgpool.conf**  
   Main configuration file for Pgpool-II. Default configurations that comes with the RPM is almost sufficient for Smarter. There are few changes need in the file.
   1. Updating pgpool.conf, keep other options unchanged.  
      Assuming load balancing 2 servers

port = 9999  
listen\_addresses = '\*'  
backend\_hostname0 = 'dbpgdwr0s0.qa.dum.edwdc.net'  
backend\_port0 = 5432  
backend\_weight0 = 1

backend\_data\_directory0 = ‘/mnt/san-postgres/9.2/data’

backend\_flag0 = ‘ALLOW\_TO\_FAILOVER’  
backend\_hostname1 = 'dbpgdwr0s1.qa.dum.edwdc.net'  
backend\_port1 = 5432  
backend\_weight1 = 1

backend\_data\_directory1 = ‘/mnt/san-postgres/9.2/data’

backend\_flag1 = ‘ALLOW\_TO\_FAILOVER’  
load\_balance\_mode = on  
replication\_mode = off

* 1. Update pgpool.conf to add failover\_command for edmigrate, if the slaves that pgpool points doesn’t need edmigrate. then no need for this step and step 8.

failover\_command = '/usr/local/bin/pgpool\_failover.sh %d %h %p'

8. Create **/usr/local/bin/pgpool\_failover.sh** script by copying following text

#!/bin/bash  
node\_id=$1

host\_name=$2

port=$3

pcp\_user=postgres # replace this with your pcp.conf user name if necessary

pcp\_pass=postgres # replace this with your pcp.conf password if necessary

pcp\_host=localhost

pcp\_port=9898 # replace this with your pcp.conf pgpool.conf pcp port if necessary

attach\_timeout=100

pgpool\_status\_file=/var/log/pgpool-II/pgpool\_status # replace with your pgpool\_status file location

# command to check pgpool parent is running or not

COMMAND\_PS="ps -C pgpool|wc -l"

# command to check failover host name

COMMAND\_TEST="nc -w 1 $host\_name $port"

# command to reattach node to pgpool. change port number and user name, password

COMMAND\_ATTACH="/usr/bin/pcp\_attach\_node $attach\_timeout localhost $pcp\_port $pcp\_user $pcp\_pass $node\_id"

wait\_for\_host\_to\_recover() {

eval $COMMAND\_TEST

while [ $? -ne 0 ]; do

PARENT=`eval $COMMAND\_PS`

if [ $PARENT -eq "1" ]; then

rm $pgpool\_status\_file

exit

fi

eval $COMMAND\_TEST

done

eval $COMMAND\_ATTACH

}

wait\_for\_host\_to\_recover &

9. Logfile

create /var/log/pgpool if it doesn’t exist, and change ownership to postgres.postgres

* 1. /var/log/pgpool.log
  2. /var/log/pgpool/pgpool\_status.

#### pgpool for Extracts

1. Make sure postgres user exists
2. Make sure firewall is opened for port 9999
3. Disable postgresql server running

shell> service postgresql-9.2 stop

1. Make sure /var/run/pgpool is owned by postgres user
2. Edit **/etc/pgpool-II/pool\_hba.conf,** Add following to allow access to pgpool.

host all all ###.###.###.###/## trust  
host replication all ###.###.###.###/## trust

host all all 127.0.0.1/32 trust

###.###.###.### is your pgbouncer server IP/network.

1. Edit **/etc/pgpool-II/pgpool.conf**  
   Main configuration file for Pgpool-II. Default configurations that comes with RPM is almost sufficient for Smarter. There are few changes need in the file.
   1. Updating pgpool.conf

Assuming load balancing 2 servers

port = 9999  
listen\_addresses = '\*'  
backend\_hostname0 = 'dbpgdwr0s2.qa.dum.edwdc.net'  
backend\_port0 = 5432  
backend\_weight0 = 1

backend\_data\_directory0 = ‘/mnt/san-postgres/9.2/data’

backend\_flag0 = ‘ALLOW\_TO\_FAILOVER’  
backend\_hostname1 = 'dbpgdwr0s3.qa.dum.edwdc.net'  
backend\_port1 = 5432  
backend\_weight1 = 1

backend\_data\_directory1 = ‘/mnt/san-postgres/9.2/data’

backend\_flag1 = ‘ALLOW\_TO\_FAILOVER’  
load\_balance\_mode = on  
replication\_mode = off

1. Logfile  
   create /var/log/pgpool if it doesn’t exist, and change ownership to postgres.postgres
   1. /var/log/pgpool.log
   2. /var/log/pgpool/pgpool\_status.

## 7.13 Smoke test for smarter functioning

After setting up above software, there is a smoke test to test the whole system is functioning correctly.

1. open a browser
2. pick a web server for smarter. for example: web1.example.com
3. type following url into browser tab: <http://web1.example.com/services/heartbeat>
4. if it shows 200 Ok. Then the whole Smarter Balanced Reporting is running correctly now. If not. you need to troubleshooting the whole chain of reporting.

## 7.14 Landing Zone

Landing Zone is a tenant-based drop-off zone for incoming batched data files. Files are dropped into its dedicated tenant space, and are transferred to the Loader server for processing. A File Watcher sits on this machine and watches for newly arrived files

### 7.14.1 Installation

The following lists the RPMs required by Landing Zone

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| Python | 3.3 | python3-3.3.0 | Python 3 RPM is generally not available from Linux distributions, hence Amplify has supplied a customized Python 3 RPM |
| EdSFTP | TBD | edsftp | Amplify’s EdSFTP RPM for Landing Zone server. |
| smarter |  | smarter | to generate ini file |

### 7.14.2 Configuration

#### edsftp

EdSFTP is a RPM supplied by Amplify that contains scripts for setting up the Landing Zone server for tenants.

##### Configuring chroot

1. Add the following section in **/etc/ssh/sshd\_config**. Make sure the content is placed at the very bottom of the file.  
     
   Match Group edwaredataadmin  
    X11Forwarding no  
    AllowTcpForwarding no  
    ForceCommand internal-sftp  
    ChrootDirectory /sftp/%h
2. Replace existing system override in **/etc/ssh/sshd\_config**override default of no subsystems  
   #Subsystem sftp /usr/libexec/openssh/sftp-server  
   Subsystem sftp internal-sftp
3. Restart sshd service  
     
   shell> service sshd restart

##### Generating Smarter .ini file

EdSFTP also reads its configuration from the **.ini** file. You can find the instructions of how to generate the **.ini** file from [here](#h.ki3mskb1afgm).

##### Creating Groups for SFTP users

By default, all the users will belong to the group: **edwaredataadmin**. If you want to change the group that the SFTP users belong to, you can modify **/opt/edware/conf/smarter.ini** and change the value for **sftp.group**

To create the group, you will need to execute the following commands,

shell> source /opt/virtualenv/edsftp/bin/activate

(virtualenv) cd /opt/virtualenv/edsftp

(virtualenv) sftp\_driver.py --init

##### Creating Tenant Accounts

For each tenant, you will need to set up an account. Each tenant may have more than one SFTP user if required.

# adding tenant “ca”

(virtualenv) sftp\_driver.py -s -t ca

# adding user “ca\_user1”

(virtualenv) sftp\_driver.py -a -u ca\_user1 -t ca -r sftparrivals

# set password for the user

(virtualenv) passwd ca\_user1

You can test whether SFTP works with jailroot for the above user

Shell> sftp [ca\_user1@l](mailto:ca_user1@lzsftpudl0.qa.dum.edwdc.net)andingZoneServer

ftp> cd /

ftp> cd file\_drop

ftp> put </path/to/test/file/to/be/sftped>

ftp> cd /etc # access will be restricted

##### Starting EdSFTP watcher service

On the Landing Zone machine, you will need to run the SFTP watcher service to monitor incoming files that are being dropped off.

The following instructions requires the Loader server to be installed and configured (udl2 user must be created in Loader machine). Please proceed if that is completed.

As root user,

1. start service  
   service edsftp-watcher start
2. Verify the service is running

ps -ef | grep sftp\_driver

## 7.15 Loader

The Data Loader is responsible for processing newly arrived data and loading it into a Staging database.

The Loader has the ability to call a callback URL for notification purposes. Please make sure that the server is able to make outgoing HTTP and/or HTTPS calls to such URLs (port 80 and 443).

### 7.15.1 Installation

The following lists the RPMs required by Loader

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| Python | 3.3 | python3-3.3.0 | Python 3 RPM is generally not available from Linux distributions, hence Amplify has supplied a customized Python 3 RPM |
| EdUDL | TBD | edudl2 | Amplify’s RPM for Data Loader |
| smarter | TBD | smarter | Amplify’s RPM for Smarter. We currently need this installed in the Loader for logging purposes. |

### 7.15.2 Configuration

#### celeryd-udl2

##### Generate udl2\_conf.ini

This configuration file is needed and read by **celeryd-udl2**.

shell> . /opt/virtualenv/udl2/bin/activate

(virtualenv) /opt/edware/conf

(virtualenv) python generate\_ini.py -i udl2\_conf.yaml -o /opt/edware/conf/ud2\_conf.ini

This file contains configurations for tenants that are supported, the UDL database, the Staging and Production Database servers.

##### Generate smarter.ini

Currently, we require smarter.ini for configuring logging in the Loader machine.

Please follow the instructions from [here](#h.ki3mskb1afgm).

##### Initialize the UDL2 database

You will need to manually initialize the Loader Database schema:

shell> . /opt/virtualenv/udl2/bin/activate

(virtualenv) cd /opt/virtualenv/udl2

(virtualenv) python -m edudl2.database.database --action setup

##### Ensure GPG keys are copied to /opt/edware/keys

Please make sure that the encryption keys are copied to **/opt/edware/keys**

##### Mount Work Zones directories from Gluster

On every Loader server, please mount the gluster as the **root** user. Make sure the id of **udl2** user and group are the same across the central UDL DB and the pipeline machines (use group id 501 and user id of 502):

usermod -G fuse udl2

chown -R udl2.udl2 /opt/edware

To mount encrypted zones from gluster as the **udl2** user, execute the following:

encfs /mnt/gluster/udl/zones /opt/edware/zones -o umask='007'

##### Ensure Outgoing HTTP and HTTPs ports are opened

Loader makes GET requests to a configurable URL specified in the data files for notifications. Please make sure ahead of time that the URLs are reachable.

##### Start edudl2-file-grabber and edudl2-trigger service to watch for incoming files being copied to work zone

**Note:** If multiple Loader servers exist, only one instance needs to be running the **edudl2-file-grabber** and **edudl2-trigger** services.

**edudl2-file-grabber** is to move files (except **.partial** file extension) from the Landing Zone to the Loader by rsync for the **edudl2-trigger** service.

To start **edudl2-file-grabber**:

service edudl2-file-grabber start

**edudl2-trigger** is used to monitor and watch for files that have arrived in the Loader machine and triggering the Loader to process the new data file.

To start **edudl2-trigger**:

service edudl2-trigger start

The logs can be found based on the logging configs defined in **/opt/edware/conf/smarter.ini**.

## 7.16 Loader Messenger

Loader Messenger hosts the broker for the Loader tasks that are requested by Data Loader. We have chosen to use RabbitMQ as the message broker. RabbitMQ, written in Erlang, implements the Advanced Message Queuing Protocol (AMQP) standard.

### 7.16.1 Installation

Please refer to the installation of RabbitMQ from [here](#h.22urj4l615b5).

**Note:**  By default, we expect and recommend the virtual host for Loader Messenger to be named, **edudl**. Please make the appropriate configuration changes.

### 7.16.2 Configuration

Please refer to the configuration of RabbitMQ from [here](#h.2hqcoe7s51rj).

## 7.17 Loader Database

The Loader Data hosts an internal centralized database used for temporary storage used by the Loader. When data files get dropped off in the Landing Zone, the data gets transformed and is temporarily stored in the Staging Database waiting to be migrated. The Loader Database is shared amongst all tenants.

In order to protect PII, all database data should be encrypted. Please see the [Encrypting](#h.b9liv75bsgd0) section to prepare to encrypt your data in PostgreSQL.

**7.17.1 Installation**

The following lists the RPMs required by Loader Database

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| Database | 9.2.8 | postgresql  postgresql92-server  postgresql92-libs  postgresql92-contrib  postgresql92-devel | RPMs for postgres are readily available from Linux distributions |
| glusterfs | 3.3.1 | glusterfs | We suggest installing a newer version of the RPM from GlusterFS’ site other than the RPM from Linux distributors.    See below for more details. |
| glusterfs-fuse | 3.3.1 | glusterfs-fuse | It provides support to FUSE based clients.    We suggest installing a newer version of the RPM from GlusterFS’ site other than the RPM from Linux distributors.    See below for more details. |
| encfs | 1.7.4 | fuse-encfs | encFS client to encrypt a volume    This RPM is readily available from Linux Distributions. |

### 7.17.2 Configuration

#### postgres

##### Allow username/password based Authentication

Update **/var/lib/pgsql/9.2/data/pg\_hba.conf**:

host all all ###.###.###.###/## trust  
host replication all ###.###.###.###/## trust

###.###.###.### is your postgresql server IP/network

### Allow client configuration

Update **/var/lib/pgsql/9.2/data/postgresql.conf**:

listen\_addresses = '\*'# what IP address(es) to listen on;

port = 5432

max\_connections = 100

Restart PostgreSQL after this configuration change:

shell> service postgresql-9.2 restart

### Create Database User

We recommend creating a dedicated user for database operations.

shell> sudo -u postgres createuser -s -e -E -d -P udl2

### Create UDL database

You will need to create a database to host the data and grant the above created user permission to the database. We recommend that you name this database, udl2:

shell> su - postgres

shell> createdb -e -E utf-8 -O udl2 -W udl2

### Create udl2 User and Group

The creation of the udl2 user and group is required for the Loader Database to change the files in the GlusterFS as the same user being used in the Loader. We recommend using 501 id for group and 501 id for the user, though you just need to make sure it’s consistent with the user created in the Loader server(s).

To add the group and user, please execute the following commands:

groupadd udl2 -f -g 501

useradd udl2 -g udl2 -u 501

### Prepare Work Zone Directory

Incoming files into the Landing Zone are copied over to an encrypted volume sitting on a gluster. You will set this up following the steps below:

1. Mount gluster onto Loader Database server  
     
   mkdir /mnt/gluster

mount -t glusterfs <glusterServer>:/gv0 /mnt/gluster

1. Mount work zone directory from gluster (this is needed for Foreign Data Wrapper to locate the same path for zones folder as defined in INI). Please execute the following commands as root:  
     
   usermod -G fuse udl2

echo "user\_allow\_other" | sudo tee -a /etc/fuse.conf

usermod -G udl2 postgres

mkdir /mnt/gluster/udl/zones /opt/edware/zones

chown -R udl2.udl2 /mnt/gluster/udl/zones/ /opt/edware/zones/

service postgresql-9.2 restart

1. As udl2 user, mount the encfs root:

encfs /mnt/gluster/udl/zones /opt/edware/zones -o allow\_other -o umask='007'

1. Create udl arrivals directories under the encfs root /opt/edware/zones  
   mkdir -p /opt/edware/zones/landing/arrivals  
   mkdir -p /opt/edware/zones/landing/work  
   mkdir -p /opt/edware/zones/landing/history
2. as udl2 user, generate ssh key pair if one does not exist.  
   Alternatively, you can specify the file location of a private key in **udl2\_rsync.args.private\_key**.
3. Copy “udl2” user public key to LZ server “root” authorized\_keys (/root/.ssh/authorized\_keys)
4. Try to ssh to LZ server without password as root user.

## 7.18 Database Staging

The Database Staging machine hosts a staging database for each tenant. The staging database gets populated by the Loader, and contains the data delta only (data since the last migration from staging to production database).

In order to protect PII, all database data should be encrypted. Please see the [Encrypting](#h.b9liv75bsgd0) section to prepare to encrypt your data in PostgreSQL.

**7.18.1 Installation**

The following lists the RPMs required by Database Staging

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| Database | 9.2.8 | postgresql  postgresql92-server  postgresql92-libs  postgresql92-contrib  postgresql92-devel | RPMs for postgres are readily available from Linux distributions |

### 7.18.2 Configuration

#### postgres

##### Allow username/password based Authentication

Update **/var/lib/pgsql/9.2/data/pg\_hba.conf**:

host all all ###.###.###.###/## trust  
host replication all ###.###.###.###/## trust

###.###.###.### is your postgresql server IP/network.

### Allow client configuration

Update **/var/lib/pgsql/9.2/data/postgresql.conf**:

listen\_addresses = '\*'# what IP address(es) to listen on;

port = 5432

max\_connections = 100

Restart PostgreSQL after this configuration change:

shell> service postgresql-9.2 restart

### Create Database User

We recommend creating a dedicated user for database operations. We recommend creating a user named **edware**:

shell> sudo -u postgres createuser -e -E -d -P edware

### Create the edware database

You will need to create a database to host the data and grant the above created user permission to the database. We recommend that you name this database **edware**:

shell> su - postgres

shell> createdb -e -E utf-8 -O edware -W edware

You can validate that the user has access by running the following commands:  
 shell> su - postgres

shell> psql -U edware -W

postgres#> \l

postgres#> \du

**7.19 Gluster (Storage)**

GlusterFS is a distributed file system capable of scaling to several petabytes and handling thousands of clients. We use GlusterFS to store a large number of files including PDFs, CSVs, and landing zone files.

The installation and configuration for GlusterFS is relatively standard, but we have listed our installation recommendation below.

### 7.19.1 Installation

The following lists the recommended RPMs required by the Gluster machine.

Note: Even though Linux distributors provide an RPM for GlusterFS server, Amplify suggests to use a newer version of the RPM directly from the GlusterFS site.

You will need to set up the **yum** repo configuration for GlusterFS:

wget -P /etc/yum.repos.d http://download.gluster.org/pub/gluster/glusterfs/LATEST/EPEL.repo/glusterfs-epel.repo

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| glusterfs | 3.3.1 | glusterfs-geo-replication glusterfs glusterfs-server glusterfs-fuse | RPMs for GlusterFS server |

## 7.20 Migrator

Migration is a batch process to move records from the Staging Database (pre-prod) to Database Master (prod). The process is designed to provide minimal interruption to user reporting.

It’s designed to have two roles - a conductor and player(s). The conductor orchestrates the process of removing players from the Database Pool such that they can sync with the latest data to be migrated.

### 7.20.1 Installation

**EdMigrate** is part of the smarter package. Installing the Smarter RPM is required. Also:

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| RabbitMQ | 2.6.1 | rabbitmq-server | RabbitMQ RPM is available from Extra Packages for Enterprise Linux (EPEL) |

### 7.20.2 Configuration

#### RabbitMQ

Please refer to this [section](#h.voh71frl14a) in RabbitMQ configuration. For the virtual host name, we recommend that you name it ‘**edmigrate**’.

#### edmigrate

##### Setting up the .ini file

The **.ini** file from Smarter will be used. Please see [here](#h.ki3mskb1afgm) for more details.

|  |  |  |
| --- | --- | --- |
| **Configuration Name** | **Description** | **Example Value** |
| migrate.broadcast.queue | Name of Queue used by Celery from the Conductor to the Players | edmigrate\_players |
| migrate.celery.BROKER\_URL | URL of the Message Queue server used by celery | amqp://edware:edware1234@edwappsrv1.poc.dum.edwdc.net/edmigrate |
| migrate.celery.CELERYBEAT\_SCHEDULE | celery task scheduling | [{'seconds': 100, 'task': 'task.edmigrate.master.prepare\_edware\_data\_refresh', 'schedule': 'timedelta', 'name': 'prepare-migration'}, {'seconds': 200, 'task': 'task.edmigrate.master.start\_edware\_data\_refresh', 'schedule': 'timedelta', 'name': 'start-migration'}] |
| migrate.celery.CELERY\_QUEUES | Celery queue used by EdMigrate | [{'exchange': 'fanout', 'key': 'edmigrate\_players', 'durable': False, 'name': 'edmigrate\_players'}] |
| migrate.celery.CELERY\_RESULT\_BACKEND | Type of Queue server | amqp |
| migrate.celery.CELERY\_ROUTES | name of celery route for EdMigrate | [{'edmigrate.tasks.player': {'queue': 'edmigrate\_players'}}] |
| migrate.conductor.enable | Set to True if the server is conductor | True |
| migrate.conductor.find\_player.timeout | How long in second the conductor waits to find the players | 5 |
| migrate.conductor.schedule.cron.day | How often conductor runs | \*/1 |
| migrate.iptables.chain | Name of chain to use for iptable | EDMIGRATE\_PGSQL |
| migrate.iptables.command | path to iptables | /sbin/iptables |
| migrate.iptables.mock | for testing purpose, mock iptable for the players | False |
| migrate.iptables.sudo | path to sudo | /usr/bin/sudo |
| migrate.master.hostname | hostname of PostgreSQL master server | edwdbsrv1.poc.dum.edwdc.net |
| migrate.pgpool.hostname | hostname of PgPool server | edwdbsrv4.poc.dum.edwdc.net |
| migrate.replication\_monitor.admin.apply\_lag\_tolerance | replication tolerance | 100 |
| migrate.replication\_monitor.admin.check\_interval |  | 1000 |
| migrate.replication\_monitor.admin.replication\_lag\_tolerance | replication tolerance | 100 |
| migrate.replication\_monitor.admin.time\_lag\_tolerance | replication tolerance | 100 |
| migrate.replication\_monitor.apply\_lag\_tolerance | replication tolerance | 100 |
| migrate.replication\_monitor.monitor\_timeout | replication tolerance | 28800 |
| migrate.replication\_monitor.replication\_lag\_tolerance | replication tolerance | 100 |
| migrate.replication\_monitor.time\_lag\_tolerance | replication tolerance | 100 |
| migrate.timeout |  | 5 |
| migrate\_dest.db.[tenant].schema\_name | schema name production server | edware\_prod |
| migrate\_dest.db.[tenant].url | production PostgreSQL server | postgresql+psycopg2://edware:edware2013@localhost:5432/edware |
| migrate\_source.db.[tenant].url | pre-production PostgreSQL server | postgresql+psycopg2://edware:edware2013@localhost:5432/edware |
| edware\_stats.db.schema\_name | The schema name of the stats database server | edware\_stats |
| edware\_stats.db.url | the database url of the stats database server | postgresql+psycopg2://edware:edware2013@localhost:5432/edware\_stats |

**Configure Conductor**

1. Register startup service  
   shell> chkconfig --add edmigrate-conductor
2. Start service  
   shell> service edmigrate-conductor start
3. Stop service  
   shell> service edmigrate-conductor stop

**Diagrams to illustrate migration process**







# 

## 7.21 HTTPS Pickup Zone Web Server

# 

### 7.21.1 Installation

The following RPMs are required for installing **hpz-web**:

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| apache module | 0.12 | mod\_xsendfile | RPMs for apache server to process X-SENDFILE headers |
| xmlsec1 | 1.2.16 | xmlsec1  xmlsec1-openssl | RPMs for XML security |
| encfs | 1.7.4 | fuse-encfs | RPMS for encrypt/decrypt hpz data on glusterfs |
| fuse | 2.8.3 | fuse  fuse-libs | RPMs for fuse file system interface to be used by encfs and glusterfs |
| gluster | 3.3.1 | glusterfs  glusterfs-fuse | RPMs for hpz to use gluster file server |
| mod\_wsgi | 3.4.0 | python3-mod\_wsgi | RPMs for apache server uses Python wsgi server |
| python | 3.3.0 | python3-3.3.0  python3-libs  python3-psycopg2 | RPMs for python runtime |
| postgresql | 9.2.8 | postgresql92  postgresql92-libs | RPMs for hpz server to contact its working database |
| hpz | 0.1 | hpz | RPM for http pickup zone |

### 7.21.2 hpz-web Configuration

#### 7.21.2.1 Configure apache

1. Update the apache config file **/etc/sysconfig/httpd**:

|  |  |  |
| --- | --- | --- |
| **Configuration Name** | **Description** | **Example Value** |
| HTTPD | *Configuration file for the httpd service.* | /usr/sbin/httpd.worker |

Using Appendix A, add:

* conf/httpd.conf
* conf.d/ssl\_on.conf
* conf.d/xsendfile.conf
* conf.d/wsgi\_frs.conf
* conf.d/wsgi\_swi.conf

Remove:

* conf.d/ssl.conf
* conf.d/welcome.conf

#### 7.21.2.2 glusterfs

Configure glusterfs

1. Mount **glusterfs** as root:

shell> mount -t glusterfs glstrpdfstr0.qa.dum.edwdc.net:/gv0 /mnt/gluster

1. Create the hpz uploads directory:

shell> cd /mnt/gluster

shell> mkdir hpz

shell> cd hpz

shell> mkdir uploads

1. Make the **apache** user owner of the **hpz** directory:

shell> cd /mnt/gluster/

shell> chown -R apache:apache hpz

#### 7.21.2.3 encfs

Configure encfs

1. Create **/opt/edware/hpz/uploads**, and ensure it is owned by **apache**:

shell> mkdir /opt/edware/hpz/uploads

shell> chown apache.apache /opt/edware/hpz/uploads

1. Make the **apache** user a member of the **fuse** group:

shell> usermod -G fuse apache

1. Mount **encfs** as the **apache** user, and give a password for **encfs**:

shell> sudo su -s /bin/sh apache -c "encfs /mnt/gluster/hpz/uploads /opt/edware/hpz/uploads"

#### 7.21.2.4 Application Configuration

Generate the .ini file:

$ cd /opt/edware/conf

$ source /opt/virtualenv/hpz/bin/activate

$ python generate\_ini.py -e qa -o /opt/edware/conf/hpz.ini

Install the IDP Metadata File

1. Grab the IDP’s metadata file from the OpenAM server in your installation for example <https://auth1.qa.dum.edwdc.net/openam/saml2/jsp/exportmetadata.jsp?entityid=https://auth1.qa.dum.edwdc.net/openam> and save it as **/opt/edware/conf/idp\_metadata.xml**.

You can adjust **hpz**’s number of connections by changing the following options in:

* /opt/edware/conf/hpz.ini
* /etc/httpd/conf.d/wsgi\_frs.conf
* /etc/httpd/conf.d/wsgi\_swi.conf

hpz.ini

|  |  |  |
| --- | --- | --- |
| **Configuration Name** | **Description** | **Example Value** |
| hpz.db\_pool\_size | For optimal configuration, the pool\_size should be equal to number of threads | 5 |

wsgi\_swi.conf

|  |  |  |
| --- | --- | --- |
| **Configuration Name** | **Description** | **Example Value** |
| WSGIDaemonProcess | number of process and threads for external facing endpoint | swi user=apache group=apache processes=2 threads=30 python-path=/opt/virtualenv/hpz/lib/python3.3/site-packages |
| MaxClients | must be at least as large  as ThreadsPerChild | 90 |

wsgi\_frs.conf

|  |  |  |
| --- | --- | --- |
| **Configuration Name** | **Description** | **Example Value** |
| WSGIDaemonProcess | number of process and threads used by internal facing endpoint | frs user=apache group=apache processes=2 threads=30 python-path=/opt/virtualenv/hpz/lib/python3.3/site-packages |
| MaxClients | must be at least as large  as ThreadsPerChild | 90 |

**7.22.2 hpz-db installation**

The following RPMs are required for installing the **hpz** database server:

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| postgresql | 9.2.8 | postgresql92  postgresql92-libs  postgresql92-server  postgresql92-contrib  postgresql92-devel | RPMS for hpz database to keep track of files. |

### 7.22.2 hpz-db Configuration

Configure Postgresql

1. Update the config file: **/var/lib/pgsql/9.2/data/postgresql.conf**:

|  |  |  |
| --- | --- | --- |
| **Configuration Name** | **Description** | **Example Value** |
| listen\_address |  | ‘\*’ |
| shared\_buffers |  | 8192MB |
| work\_mem |  | 100MB |
| log\_filename |  | ‘postgresql-%a.log’ |
| log\_truncate\_on\_rotation |  | on |
| log\_rotation\_size |  | 0 |
| log\_timezone |  | ‘UTC’ |
| max\_connections | max number of db connections for postgres server | 100 |
| timezone |  | ‘UTC’ |

1. Update **/var/lib/pgsql/9.2/data/pg\_hba.conf**:

local all all peer

# IPv4 local connections:

hostssl all all 127.0.0.1/32 trust

# Replication:

hostssl replication all 10.0.0.0/8 trust

hostssl all all 10.0.0.0/8 md5

# IPv6 local connections:

hostssl all all ::1/128 ident

###.###.###.### is your postgresql slave server IP/network.

1. Start the Postgres Database:

shell> services postgresql-9.2 start

1. If postgresql was not used, initialize postgres:

shell> services postgresql-9.2 initdb

1. If the HPZ database is not created yet, create the HPZ Database:

shell> su - postgres

shell> psql

psql> CREATE DATABASE hpz

1. If the HPZ user is not created yet, create the DB user:

psql> CREATE USER hpz WITH PASSWORD 'hpz2015';  
psql> GRANT ALL PRIVILEGES ON DATABASE hpz to hpz;

1. Teardown any old HPZ Schemas:

shell> python3.3 -m hpz.database.metadata\_generator --metadata hpz -a teardown -s hpz -d hpz --host=dbpghpz0.qa.dum.edwdc.net:5432 -u hpz -p hpz2014

1. Initialize the HPZ Schema:

shell> python3.3 -m hpz.database.metadata\_generator --metadata hpz -s hpz -d hpz --host=dbpghpz0.qa.dum.edwdc.net:5432 -u hpz -p hpz2014

Smoke Test For HPZ

1. Register a file:
   1. curl -X PUT -H "Content-Type: application/json" -d '{"uid":"shall"}' <https://hpzweb0.qa.dum.edwdc.net:8443/registration> -k
   2. Capture the registration id and download url from the response.
2. Copy a file to HPZ:
   1. curl -X POST -H "File-Name: test.txt" -F "file=@<Path to file>" <https://hpzweb0.qa.dum.edwdc.net:8443/files/><registration id> -k
3. Download the file from HPZ:
   1. Use the download url captured in the first step to login as shall and download the test.txt file.

### 7.22.3 Heartbeat

HPZ provides a heartbeat service to monitor service health.

URL: **http[s]://hostname/services/heartbeat**

Response:

200 - OK

500 - Database connection is down and/or Disk is not writable for 1024 bytes (disk maybe full).

## 7.23 Score Batcher Web Server

Score Batcher Web Servers are responsible for running the Web Application for handling XMLs for Score Batching.

## 7.23.1 Installation

The following table lists the RPMs that must be installed on web servers:

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Tested Version** | **RPM Name** | **Description** |
| Python | 3.3 | python3 | Python 3 RPM is generally not available from Linux distributions, hence Amplify has supplied a customized Python 3 RPM |
| Apache | 2.2.15 | httpd | Apache hosts Smarter Score Batcher Web Application. It can be installed from the standard distributed RPM. We highly recommend using customized configuration. Apache should run on worker MPM mode. |
| WSGI | 3.4 | python3-mod\_wsgi | WSGI defines a simple and universal interface between web servers and web applications or frameworks for Python. This is a customized RPM supplied by Amplify. |
| Smarter\_score\_batcher | TBD | smarter\_score\_batcher | Amplify’s Smarter Score Batcher Web Application |
| XMLSec | 1.2.16 | xmlsec1  xmlsec1-openssl-devel | XMLSec can be installed from standard distributed RPM. It is used for verification of SAML Responses. |

### 7.22.2 Configuration

#### 7.23.2.1 Apache Configuration

Using Appendix A, add:

* conf/httpd.conf
* conf.d/wsgi\_tsb.conf
* conf.d/ssl\_main\_server.conf
* conf.d/rewrite\_tsb.conf

Remove:

* conf.d/welcome.conf
* conf.d/ssl.conf

#### 7.23.2.2 Application Configuration

Score Batcher Web Server uses an **.ini** file to configure the application. The path of this file is **/opt/edware/conf/smarter\_score\_batcher.ini**  This is a manual process to generate this file, and the steps to generate this file is similar to Smarter Reporting Web Server’s **.ini** file.

smarter\_score\_batcher.ini

|  |  |  |
| --- | --- | --- |
| **Configuration Name** | **Description** | **Example Value** |
| smarter\_score\_batcher.metadata.performance | Location of performance xml metadata | /opt/edware/resources/meta/performance |
| smarter\_score\_batcher.metadata.static | Location of static metadata | /opt/edware/resources/meta/static |

## 7.24 Score Batcher Messenger

Score Batcher Messenger is a message system between the Smarter Score Batcher Web Application and Score Batcher Worker. It uses RabbitMQ as its message broker and score batcher tasks are sent from Web Servers and are received by Score Batcher Workers. The installation and configuration is very similar to PDF Messenger.

### 7.24.1 Installation

Please refer to the installation of RabbitMQ from [here](#h.22urj4l615b5).

**Note:**  The only configuration difference is that the name of the virtual host for score batcher should be different than the one defined for PDF Messenger. By default, we expect and recommend that the virtual host for Score Batcher Messenger should be named ‘**smarter\_score\_batcher**’.

### 7.24.2 Configuration

Please refer to the configuration of RabbitMQ from [here](#h.2hqcoe7s51rj).

## 7.25 Score Batcher Worker

The Score Batcher worker is responsible for receiving score batcher tasks from the queue and saving raw XML to storage, and generating item level files, and landing zone format CSV/JSON files. The installation and configuration is very similar to PDF Worker.

There are a few key differences for Score Batcher Worker:

* The Celery service name is **celeryd-smarter\_score\_batcher**.
* The Celery configuration file is **celeryd-smarter\_score\_batcher.conf**.
* Mount EncFS to **/opt/edware/item\_level**, **/opt/edware/raw\_data** and **/opt/edware/tsb** (be sure to mount it as the **celery** user.)
* **smarter\_score\_batcher.celery.BROKER\_URL** is the broker URL used by the Test Score Batcher Worker.

### 7.25.1 Installation

Please refer to the installation of Score Batcher Worker from [here](#h.a9cm003zx56q).

**Note:**  Please remember the notable installation differences between PDF and Score Batcher Worker, namely, the celery service.

### 7.25.2 Configuration

Please refer to the configuration of Score Batcher Worker from [here](#h.hrusrg3lgvr9).

**Note:**  Please remember the notable configuration differences between PDF and Test Score Batcher Worker, namely, the **celeryd** configuration file, the EncFS mount point, and the broker URL configuration in **smarter\_score\_batcher.ini**.

#### Custom XML Metadata Directory Structure

Score Batcher Worker relies on static custom metadata files that describe assessments. These files must be manually placed in the directories specified in the **.ini** file. These files can be stored on a gluster volume and all the workers can access these files over the network.

Performance metadata XML must be placed in the following directory structure:

<smarter\_score\_batcher.metadata.performance>/<year>/<assessment type>/<grade>/<subject>.xml

Static Metadata will be used to merge with Custom XML Metadata. Static Metadata should be placed in the following directory structure, <smarter\_score\_batcher.metadata.static>/<subject>.json

Here is a sample format:  
{

"Content": "assessment",

"Identification": {

"Subject": "ELA"

},

"PerformanceLevels": {

"Level1": {

"Name": "Level 1"

},

"Level2": {

"Name": "Level 2"

},

"Level3": {

"Name": "Level 3"

},

"Level4": {

"Name": "Level 4"

}

},

"Claims": {

"Claim1": {

"Name": "Reading",

"Mapping": "myClaim1"

},

"Claim2": {

"Name": "Writing",

"Mapping": "Claim2Writing"

},

"Claim3": {

"Name": "Listening",

"Mapping": "Claim3Listening"

},

"Claim4": {

"Name": "Research & Inquiry",

"Mapping": "Claim4ReadingResearch and Inquiry"

}

},

"ClaimsPerformanceLevel": {

"Level1": {

"Name": "Below Standard"

},

"Level2": {

"Name": "At/Near Standard"

},

"Level3": {

"Name": "Above Standard"

}

}

}

Score Batcher also provides default metadata files as fallback for performance metadata. The default metadata files have similar structure as static metadata files, and are mutually exclusive with performance metadata. When Score Batcher runs, it first tries to get performance metadata; if no such data is available, it looks for default metadata as a last resort before raising an error. Usually default metadata files should be transparent to an operator, but in case that providing performance metadata is impossible while still needing to provide additional information to static metadata, one can place default metadata files under <smarter\_score\_batcher.metadata.default>/<assessment type>/<subject>.json.

To create default metadata, follow the below steps. In this example we use “**Interim Assessment Blocks**” as assessment type and “**Math**” as subject. Please replace the variables enclosed by `<`, `>` with values that are specific to your environment.

1. Modify the **.ini** file and modify <smarter\_score\_batcher.metadata.default> to the location where default metadata should be located.
2. Change current work directory to <smarter\_score\_batcher.metadata.default> using the command line:

$ cd <smarter\_score\_batch.metadata.default>

1. Execute below command to make a directory with the name of corresponding assessment type:

$ mkdir interim\ assessment\ blocks & cd interim\ assessment\ blocks & touch MATH.default\_asmt\_metadata.json

1. Paste below sample JSON file into `MATH.default\_asmt\_metadata.json`:

{

"Content": "assessment",

"Identification": {

"Subject": "Math",

"Period": "" },

"PerformanceLevels": {

"Level1": {

"Name": "",

"CutPoint": "" },

"Level2": {

"Name": "",

"CutPoint": "" },

"Level3": {

"Name": "",

"CutPoint": "" },

"Level4": {

"Name": "",

"CutPoint": "" },

"Level5": {

"Name": "",

"CutPoint": "" }

},

"Overall": {

"MaxScore": 0,

"MinScore": 0 },

"Claims": {

"Claim1": {

"Name": "",

"Mapping": "",

"MinScore": 0,

"MaxScore": 0 },

"Claim2": {

"Name": "",

"Mapping": "",

"MinScore": 0,

"MaxScore": 0 },

"Claim3": {

"Name": "",

"Mapping": "",

"MinScore": 0,

"MaxScore": 0 },

"Claim4": {

"Name": "",

"Mapping": "",

"MinScore": 0,

"MaxScore": 0 }},

"ClaimsPerformanceLevel": {

"Level1": {

"Name": "" },

"Level2": {

"Name": "" },

"Level3": {

"Name": "" }

}

}

Claim Name Mapping

Claims.Claim1.Mapping key is used as Claim alias name.  
e.g.  
<Score measureOf="**myClaim1**" measureLabel="ScaleScore" value="1285" standardError="13" />

<Score measureOf="**myClaim1**" measureLabel="PerformanceLevel" value="1" />

In this example, myClaim1 is used to identify as the Claim1 entry.

If no claim name mapping exists, Score Batcher will map “Claim1” from the value of the first Score element whose *measureLabel* attribute is “*ScaleScore*” and *measureOf* attribute is not “Overall”. All other claims are left with empty value in this case.

## 7.26 Score Batcher Trigger

Score Batcher Trigger is responsible for encrypting and archiving CSV and JSON files into a batch to be consumed by the Loader . The installation and configuration is very similar to Score Batcher Worker.

### 7.26.1 Installation

Please refer to the installation of Score Batcher Trigger from [here](#h.t5rrzhtvobou).

### 7.26.2 Configuration

Please refer to the configuration of Score Batcher Trigger from [here](#h.12acx6u7ng6d).

Score Batcher Trigger needs to have the following settings enabled in its **smarter\_score\_batcher.ini**:

|  |  |  |
| --- | --- | --- |
| **Configuration Name** | **Description** | **Example Value** |
| trigger.assessment.enable | Enabling the worker to batch up CSV and JSON files to be sent to Loader | true |
| trigger.assessment.schedule.cron.[minute|day] | Cron scheduling of how frequent to batch up the files. | \*/5 |

## 7.26 Score Batcher Database Server

# Score Batcher Database Server is a temporary storage that saves assessment information in received XML. This information will be processed by Score Batcher Trigger and then archived and encrypted into a batch.

### 7.26.1 Installation

Database installation is the same as installing . See [7.9 Database Master](#h.92fx825yt7ie) for detail.

##### **7.26.2 Installation** Creating a Database Schema

You must manually create an empty schema for **smarter score batcher**. A script is provided to perform this task. This step requires that Database is installed and configured first.

shell> . /opt/virtualenv/smarter\_score\_batcher/bin/activate

(virtualenv) cd /opt/virtualenv/smarter\_score\_batcher/lib/python3.3/site-packages/smarter\_score\_batcher-0.1-py3.3.egg/smarter\_score\_batcher/database

(virtualenv) python metadat.py -s edware -d edware -u edware --host [dbMastHostName] -p [password]

**Note**: The command is in the format of:   
python metadata\_generator.py -s [schemaName] -d [databaseName] -u [userName] --host [dbMastHostName] -p [password]

### 7.26.3 Configuration

# Once the database is ready, one needs to update the database connection configuration in **smarter\_score\_batcher.ini** on each smarter score batcher worker server. Update the below two directives with the actual values of database connection:

smarter\_score\_batcher.db.schema\_name = edware\_tsb

smarter\_score\_batcher.db.url = postgresql+psycopg2://edware:edware2013@localhost:5432/edware

# 8 Starting Applications

The relevant services for each machine type are listed below. You can use the commands below to start and/or stop the services.

|  |  |  |
| --- | --- | --- |
| **Type** | **Component** | **Command** |
| Web Servers | Smarter | /etc/init.d/httpd [start|stop] |
| HTTPS Pickup Zone Servers | https pickup zone server | /etc/init.d/httpd [start|stop] |
| HTTPS Pickup Zone Servers | encfs | sudo su -s /bin/sh apache -c "encfs /mnt/gluster/hpz/uploads /opt/edware/hpz/uploads" |
| PDF Messenger | RabbitMQ | /etc/init.d/rabbitmq-server [start|stop] |
| PDF Worker | celeryd-services | /etc/init.d/celeryd-services [start|stop] |
| PDF Pre-Generator | Smarter | /etc/init.d/httpd [start|stop] |
| Extract Messenger | RabbitMQ | /etc/init.d/rabbitmq-server [start|stop] |
| Extract Worker | celeryd-edextract | /etc/init.d/celeryd-services [start|stop] |
| Cache | memcached | /etc/init.d/memcached [start|stop] |
| Cache Warmer | Smarter | /etc/init.d/httpd [start|stop] |
| Database Master | PostgreSQL | /etc/init.d/postgres-9.2 [start|stop] |
| Database Replica | PostgreSQL | /etc/init.d/postgres-9.2 [start|stop] |
| Database Load Balancer | PgBouncer | /etc/init.d/pgbouncer [start|stop] |
| Database Pool | Pgpool-II | /etc/init.d/pgpool [start|stop] |
| Landing Zone | edsftp-watcher | service edsftp-watcher [start|stop] |
| Loader | edudl2-trigger | service edudl2-trigger start |
| Loader | edudl2-file-grabber | service edudl2-file-grabber [start|stop] |
| Loader | edudl | service celeryd-edudl [start|stop] |
| Loader Messenger | RabbitMQ | /etc/init.d/rabbitmq-server [start|stop] |
| Loader Database | PostgreSQL | /etc/init.d/postgres-9.2 [start|stop] |
| Migrator | Conductor | /etc/init.d/edmigrate-conductor [start|stop] |
| Migrator | Player | /etc/init.d/celeryd-edmigrate [start|stop] |
| Database Staging | PostgreSQL | /etc/init.d/postgres-9.2 [start|stop] |
| GlusterFS | glusterfsd | service glusterd [start|stop] |
| Score Batcher Web Server | smarter\_score\_batcher | /etc/init.d/httpd [start|stop] |
| Score Batcher Messenger | RabbitMQ | /etc/init.d/rabbitmq-server [start|stop] |
| Score Batcher Worker | celeryd-smarter\_score\_batcher | /etc/init.d/celeryd-smarter\_score\_batcher [start|stop] |
| Score Batcher Trigger | celeryd-smarter\_score\_batcher | /etc/init.d/celeryd-smarter\_score\_batcher [start|stop] |

# 9 Logging & Monitoring

## 9.1 Web Server

**Apache**

The Apache server provides very comprehensive and flexible logging capabilities.

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/httpd/access\_log | Refer to access\_log for requests being served by Apache |
| /var/log/httpd/error\_log | Refer to error\_log for any errors that surface up to Apache |

**Smarter**

Smarter has three log files - **smarter.log**, **audit.log** and **security\_event.log**. Smarter utilizes the syslog service for logging and uses the local syslog facilities. By default, **local0** is used by audit.log, **local1** is used by smarter.log, and **local2** is used by audit.log. Using syslog facilities, log levels can be changed inside the **.ini** file. The location of the log files is managed by rsyslogd.

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /opt/edware/log/smarter.log | This log contains application level logging. |
| /opt/edware/log/audit.log | This log contains information on reports being accessed. |
| /opt/edware/log/security\_event.log | This log contains information on security events in the system related to login, logout, forbidden access, and SAML2 responses. |

## 9.2 HTTP Pickup Server

**Apache**

|  |  |
| --- | --- |
| **Log File Directory** | **Description** |
| /var/log/httpd/access.log | The log contains information for http pickup server’ access records |
| /var/log/httpd/error.log | The log contains information for errors in hpz wsgi server. |

**HPZ**

|  |  |
| --- | --- |
| **Log File Directory** | **Description** |
| /var/log/hpz/log | The log contains information on hpz working status and errors. |

## 9.3 **PDF Messenger** [Ansible name: reporting-rabbit-services]

**RabbitMQ**

|  |  |
| --- | --- |
| **Log File Directory** | **Description** |
| /var/log/rabbitmq/ | All RabbitMQ logging are saved into this directory |

## 9.4 PDF Worker [Ansible name: reporting-worker-pdf]

**celeryd-services**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/celery-services/default\_worker.log | Logs for PDF Worker tasks |
| /var/log/celery-services/batch\_worker.log | Logs for PDF Pre-Generator tasks |
| /var/log/celery-services/health\_check\_worker.log | Logs for health check worker. This worker dequeues from our health check queue. |

## 9.5 PDF Pre-Generator [Ansible name: reporting-generator-pdf]

**Smarter**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /opt/edware/log/smarter.log | This log contains application level logging. |

## 9.6 Extract Messenger [Ansible name: reporting-rabbit-extract]

**RabbitMQ**

|  |  |
| --- | --- |
| **Log File Directory** | **Description** |
| /var/log/rabbitmq/ | All RabbitMQ logging are saved into this directory |

## 9.7 Extract Worker [Ansible name: reporting-worker-extract]

**celeryd-edextract**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/celery-edextract/extract\_sync\_worker.log | Logs for synchronous extraction requests. |
| /var/log/celery-edextract/extract\_worker.log | Logs for asynchronous extraction requests. |
| /var/log/celery-edextract/extract\_archive\_worker.log | Logs for archiving step of extracting requests |

## 9.8 Cache [Ansible name: memcached]

**memcached**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/memcached.log | Logs related to memcached. By default, logging is disabled |

## 9.9 Cache Warmer [ansible name: reporting-cache-warmer]

**Smarter**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /opt/edware/log/smarter.log | This log contains application level logging. |

## 9.10 Database Master

**PostgreSQL**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/postgres/postgres.log | PostgreSQL application level logs |

## 9.11 Database Replica

**PostgreSQL**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/postgres/postgres.log | PostgreSQL application level logs |

## 9.12 Database Load Balancer

**PgBouncer**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/pgbouncer/pgbouncer.log | Logging of PgBouncer status |

## 9.13 Database Pool

**Pgpool-II**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/pgpool.log | Logging of Pgpool |
| /var/log/pgpool/pgpool\_status | Logging of Pgpool status |

## 9.14 Landing Zone

**edsftp-watcher**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /opt/edware/log/smarter.log | This log contains application level logging. |

## 9.15 Loader

**edudl2-trigger  
edudl2-file-grabber**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /opt/edware/log/smarter.log | This log contains application level logging. |

**edudl**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/celeryd-udl2/udl2\_worker.log | Logging of edudl service |

## 9.16 Loader Messenger

**RabbitMQ**

|  |  |
| --- | --- |
| **Log File Directory** | **Description** |
| /var/log/rabbitmq/ | All RabbitMQ logging are saved into this directory |

## 9.17 Loader Database

**PostgreSQL**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/postgres/postgres.log | PostgreSQL application level logs |

## 

## 9.18 Database Staging

**PostgreSQL**

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/postgres/postgres.log | PostgreSQL application level logs |

## 9.19 Migrator

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /opt/edware/log/smarter.log | This log contains application level logging. |

## 9.20 Score Batcher Web Server

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /opt/edware/log/smarter.log | This log contains application level logging. |

## 9.21 Score Batcher Messenger

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/rabbitmq/ | All RabbitMQ Logs are saved in this directory. |

## 9.22 Score Batcher Worker

|  |  |
| --- | --- |
| **Log File** | **Description** |
| /var/log/celery-smarter\_score\_batcher/smarter\_score\_batcher\_sync\_worker.log | This log is for the workers handing synchronous requests. |
| /var/log/celery-smarter\_score\_batcher/smarter\_score\_batcher\_async\_worker.log | This log is for the workers handling asynchronous requests. |

# 

# 

# 10 Troubleshooting

**How do I know if the smarter web application is up and running?**

When Smarter is up and running, you can navigate to /services/heartbeat endpoint and make sure a 200 OK is returned.  
  
Ex. http://[hostname]/services/heartbeat  
  
A 200 OK is returned only if smarter is able to connect to all the databases that are configured in smarter.ini and that it’s heartbeat task is processed in the queue.

**How do I tell if memcached is up and running?**

telnet [server] 11211  
stats items

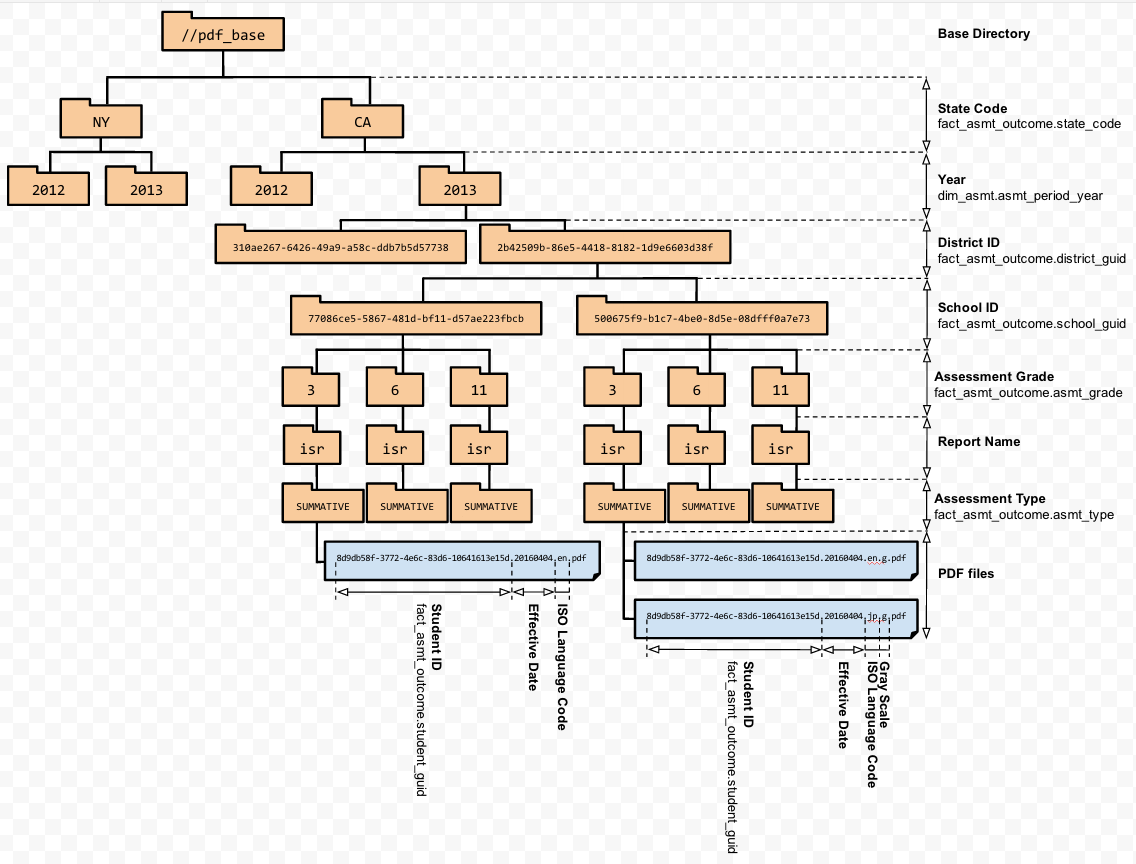
**How do I clear the contents in memcache?**

The quickest way is to restart memcached service on cache server.

**A bad PDF was generated. How do I remove it?**

Login as celery user and then change to the **/opt/edware/pdf** directory. Find the PDF and delete it.

The diagram below describes the directory structure in which PDFs are stored.



**How do I delete all the PDF files?**

The quickest way to delete all the PDF files is to delete the directory directory within the volume.

rm -rf /export/brick1/vdb1/smarter/[dir]

**Important Warning:**  You will see a “**/export/brick1/vdb1/smarter/.encfs6.xml**” file. **DO NOT DELETE THE FILE**. It contains EncFS information. Deleting this file causes PDF files to become unrecoverable by EncFS.

**How do I clear all queues in RabbitMQ?**

Restarting RabbitMQ does not clear all queues. Executing the following commands will clear all the queues.

rabbitmqctl stop\_app  
rabbitmqctl reset  
rabbitmqctl start\_app

**How do I monitor Celery Tasks?**

Flower is a real-time web based monitor and administration tool for Celery. It provides graphs and statistics on task details, as well as remote control. An important note is that Celery Flower runs on Python 2.7 (Reference: [https://github.com/mher/flower)](https://github.com/mher/flower).

**How do I list the queues or users configured in RabbitMQ?**

To get the list of users, you can execute the following as root user:

rabbitmqctl list\_users

To get the list of queues, you can execute the following as the root user:

rabbitmqctl list\_queues [-p vhostpath]

# 

# 

# 11 Maintenance

## 11.1 Cache

**Clearing Cache**

Cache can be flushed only by a user with super admin rights. Using a Rest Client (ex. Chrome Advanced Rest Client), the super admin user can send REST API calls to interface with the cache services.  
  
To delete all data in memcached:  
Send a DELETE request to http://[hostname]/services/flush/all  
  
To delete sessions in memcached:  
Send a DELETE request to http://[hostname]/services/flush/session  
  
To delete reports data in memcached:  
Send a DELETE request to http://[hostname]/services/flush/data

## 11.2 Database

**Cleanup Maintenance [Database, Repmgr and pg\_xlogs]**

Vacuum and Analyze

Vacuum can be performed on the Loader Database, Staging Database and Production Database Master to compact and release space back to the OS. The frequency of doing this can be different for each database.

Example of Running vacuum on Loader Database:

Determine DB size

> psql -h localhost -U udl2 -d udl2 -c "select pg\_size\_pretty(pg\_database\_size('udl2'))"

Run vacuum command while udl is running. This frees up space of the deleted records,

but will not release space to OS. This command is non-blocking

> **psql -h localhost -U udl2 -d udl2 -c “VACUUM (VERBOSE, ANALYZE)”**

Run vacuum command while udl2 is not running. This command blocks the database but

releases the space to OS and is more efficient than previous

> **psql -h localhost -U udl2 -d udl2 -c “VACUUM (VERBOSE, FULL)”**

<http://www.postgresql.org/docs/9.2/static/sql-vacuum.html>

<http://www.postgresql.org/docs/9.2/static/sql-analyze.html>

Transaction log Cleanup

Transaction logs under pg\_xlog can be archived continuously by the following settings:

# To prevent the primary server from removing the WAL segments required for  
# the standby server before shipping them, set the minimum number of segments  
# retained in the pg\_xlog directory. At least wal\_keep\_segments should be  
# larger than the number of segments generated between the beginning of  
# online-backup and the startup of streaming replication. If you enable WAL  
# archiving to an archive directory accessible from the standby, this may  
# not be necessary.  
**wal\_keep\_segments = 32**  
# Enable WAL archiving on the primary to an archive directory accessible from  
# the standby. If wal\_keep\_segments is a high enough number to retain the WAL  
# segments required for the standby server, this is not necessary.  
**archive\_mode = on  
archive\_command = 'gzip -9 < %p > /mnt/server/archivedir/%f && rm %p'**

The archive file path should be accessible to the master and the slaves for recovery. In this case we can keep the value of **wal\_keep\_segments** to minimal such as **32**.

We also need to set up recovery.conf to enable postgres recover from gzipped archive wal files.

In recovery.conf:

**restore\_command = 'gunzip < /mnt/server/archivedir/%f > %p'**

**archive\_cleanup\_command = 'pg\_archivecleanup /mnt/server/archivedir %r'**

<http://www.mkyong.com/database/postgresql-point-in-time-recovery-incremental-backup/>

<http://wiki.postgresql.org/wiki/Streaming_Replication>

## 11.3 HTTP Pickup Zone

**Cleanup Maintenance**

HPZ will store files internally for the number of days listed on ‘/opt/edware/conf/hpz.ini’ by ‘hpz.record\_expiration’ . You can also cleanup manually via the pickup\_zone\_cleanup.py script:

Cleanup via script

**Requirement**

1. Make sure pickup\_zone\_cleanup.py is installed on the hpz server.
2. The user that the script runs as needs to be able to delete both entries in HPZ's internal database and the files in HPZ's filesystem.
3. Python and SQLAlchemy are installed.

**Usage**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Description** | **Required?** |
| -c, --config | The local path to HPZ's config file. If not provided, will default to /opt/edware/conf/hpz.ini | No |
| -e, --expiration | Files (and their related metadata) older than the number of days passed in (as an integer) will be removed | Yes |

For example, when HPZ's config file is located at "/the/path/to/my/file.ini" and all files older than a week that HPZ is storing should be removed:

python pickup\_zone\_cleanup.py -c "/the/path/to/my/file.ini" -e 7

HPZ is meant to be used with extract files that can take a lot time to generate. As a general rule, expiration values should be in the range of 3 to 10 days. Anything less, end users may not always be able to receive the file that they requested. Anything more, large files may be hanging around long after they've been used.

The clean up script can be triggered via **crond**.

Clean up manually

1. Check the download url that corresponds to that file.
2. The url contains an id that corresponds to that file.
3. The format of the url is <hpz host>/download/<file id>.
4. On the machine that is running HPZ, navigate to the location where HPZ stores files. (Refer to HPZ's config file if uncertain where HPZ stores them). Remove the file that has the same name as the file id.

# Appendix A: Apache Configuration Files

## conf/httpd.conf

|  |
| --- |
| ServerTokens Prod  ServerRoot "/etc/httpd"  PidFile run/httpd.pid  Timeout 120  KeepAlive Off  MaxKeepAliveRequests 100  KeepAliveTimeout 15  Listen 80  LoadModule authz\_host\_module modules/mod\_authz\_host.so  LoadModule log\_config\_module modules/mod\_log\_config.so  LoadModule logio\_module modules/mod\_logio.so  LoadModule env\_module modules/mod\_env.so  LoadModule deflate\_module modules/mod\_deflate.so  LoadModule headers\_module modules/mod\_headers.so  LoadModule mime\_module modules/mod\_mime.so  LoadModule status\_module modules/mod\_status.so  LoadModule vhost\_alias\_module modules/mod\_vhost\_alias.so  LoadModule dir\_module modules/mod\_dir.so  LoadModule alias\_module modules/mod\_alias.so  LoadModule setenvif\_module modules/mod\_setenvif.so  LoadModule rewrite\_module modules/mod\_rewrite.so  Include conf.d/\*.conf  User apache  Group apache  ServerAdmin root@localhost  UseCanonicalName Off  TypesConfig /etc/mime.types  DefaultType text/plain  <IfModule mod\_mime\_magic.c>  MIMEMagicFile conf/magic  </IfModule>  HostnameLookups Off  ErrorLog logs/error\_log  LogLevel warn  LogFormat "%h %l %u %t \"%r\" %>s %b %>D \"%{Referer}i\" \"%{User-Agent}i\"" combined\_with\_duration  LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined  LogFormat "%h %l %u %t \"%r\" %>s %b" common  LogFormat "%{Referer}i -> %U" referer  LogFormat "%{User-agent}i" agent  CustomLog logs/access\_log combined env=!dontlog  ServerSignature Off  AddDefaultCharset UTF-8  AddType application/x-compress .Z  AddType application/x-gzip .gz .tgz  AddType application/x-x509-ca-cert .crt  AddType application/x-pkcs7-crl .crl  AddHandler type-map var  BrowserMatch "Mozilla/2" nokeepalive  BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0  BrowserMatch "RealPlayer 4\.0" force-response-1.0  BrowserMatch "Java/1\.0" force-response-1.0  BrowserMatch "JDK/1\.0" force-response-1.0  BrowserMatch "Microsoft Data Access Internet Publishing Provider" redirect-carefully  BrowserMatch "MS FrontPage" redirect-carefully  BrowserMatch "^WebDrive" redirect-carefully  BrowserMatch "^WebDAVFS/1.[0123]" redirect-carefully  BrowserMatch "^gnome-vfs/1.0" redirect-carefully  BrowserMatch "^XML Spy" redirect-carefully  BrowserMatch "^Dreamweaver-WebDAV-SCM1" redirect-carefully |

## conf.d/edware\_worker\_mpm.conf

|  |
| --- |
| <IfModule worker.c>  ThreadLimit 120  StartServers 1  ServerLimit 4  MaxClients 120  MinSpareThreads 30  MaxSpareThreads 120  ThreadsPerChild 30  MaxRequestsPerChild 0  MaxMemFree 256  </IfModule> |

## conf.d/rewrite\_reporting\_https.conf

|  |
| --- |
| RewriteEngine On  RewriteCond %{HTTP:X-Forwarded-Proto} !https  RewriteCond %{HTTP\_USER\_AGENT} !^ELB-HealthChecker  RewriteRule (.\*) https://reporting.smarterbalanced.org%{REQUEST\_URI} [L,R=301] |

## conf.d/rewrite\_reporting\_slash.conf

|  |
| --- |
| RewriteEngine on  RewriteRule ^/$ https://%{HTTP\_HOST}/assets/public/landing.html |

## conf.d/rewrite\_tsb.conf

|  |
| --- |
| RewriteEngine On  RewriteCond %{HTTP:X-Forwarded-Proto} !https  RewriteCond %{HTTP\_USER\_AGENT} !^ELB-HealthChecker  RewriteRule (.\*) https://reportingintake.smarterbalanced.org%{REQUEST\_URI} [L,R=301] |

## conf.d/ssl\_main\_server.conf

|  |
| --- |
| LoadModule ssl\_module modules/mod\_ssl.so  Listen 443  SSLEngine on  SSLCertificateFile /etc/pki/tls/certs/{{ ssl\_key\_name }}.crt  SSLCertificateKeyFile /etc/pki/tls/private/{{ ssl\_key\_name }}.key  SSLProtocol All -SSLv2 -SSLv3  SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE  -DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-E  CDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA256:DHE-RSA-AES256-SHA256:DHE-DSS-AES2  56-SHA:DHE-RSA-AES256-SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!3DES:!MD5:!PSK  SSLSessionCache none |

## conf.d/ssl\_on.conf

|  |
| --- |
| LoadModule ssl\_module modules/mod\_ssl.so  SSLSessionCache none |

## conf.d/wsgi\_edware.conf

|  |
| --- |
| WSGIApplicationGroup %{GLOBAL}  WSGIPassAuthorization On  WSGIDaemonProcess pyramid user=apache group=apache processes=4 threads=30 python-path=/opt/virtualenv/smarter/lib/python3.3/site-packages  WSGIScriptAlias / /opt/edware/smarter/smarter.wsgi  WSGIImportScript /opt/edware/smarter/smarter.wsgi process-group=pyramid application-group=%{GLOBAL}  WSGISocketPrefix run/wsgi  <Directory /opt/virtualenv/smarter>  WSGIProcessGroup pyramid  Order allow,deny  Allow from all  </Directory>  WSGIPythonPath /opt/virtualenv/smarter/lib/python3.3/site-packages |

## conf.d/wsgi\_frs.conf

|  |
| --- |
| WSGISocketPrefix run/wsgi  WSGIPythonPath /opt/virtualenv/hpz/lib/python3.3/site-packages  WSGIDaemonProcess frs user=apache group=apache processes=2 threads=30 python-path=/opt/virtualenv/hpz/lib/python3.3/site-packages  XSendFile on  <Directory "/opt/edware">  XSendFile on  XSendFilePath /opt/edware  </Directory>  NameVirtualHost \*:443  <VirtualHost \*:443>  WSGIApplicationGroup %{GLOBAL}  WSGIPassAuthorization On  WSGIScriptAlias / /opt/edware/hpz/frs.wsgi  WSGIImportScript /opt/edware/hpz/frs.wsgi process-group=frs application-group=%{GLOBAL}  RewriteEngine On  RewriteCond %{HTTP:X-Forwarded-Proto} !https  RewriteCond %{HTTP\_USER\_AGENT} !^ELB-HealthChecker  RewriteRule (.\*) https://reportdownload.smarterbalanced.org%{REQUEST\_URI} [L,R=301]  SSLEngine on  SSLCertificateFile /etc/pki/tls/certs/{{ ssl\_key\_name }}.crt  SSLCertificateKeyFile /etc/pki/tls/private/{{ ssl\_key\_name }}.key  SSLProtocol All -SSLv2 -SSLv3  SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256  :DHE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECD  HE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA256:DHE-RSA-AES256-SHA256:DHE-DSS-  AES256-SHA:DHE-RSA-AES256-SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!3DES:!MD5:!PSK  <LocationMatch ^/registration>  Order Allow,Deny  Deny from all  </LocationMatch>  <LocationMatch ^/files/(.\*)>  Order Allow,Deny  Deny from all  </LocationMatch>  <Directory /opt/virtualenv/hpz>  WSGIProcessGroup frs  Order allow,deny  Allow from all  </Directory>  LogLevel info  </VirtualHost>  Listen 443 |

## conf.d/wsgi\_swi.conf

|  |
| --- |
| WSGISocketPrefix run/wsgi  WSGIPythonPath /opt/virtualenv/hpz/lib/python3.3/site-packages  WSGIDaemonProcess swi user=apache group=apache processes=2 threads=30 python-path=/opt/virtualenv/hpz/lib/python3.3/site-packages  <VirtualHost \*:8443>  WSGIApplicationGroup %{GLOBAL}  WSGIPassAuthorization On  WSGIScriptAlias / /opt/edware/hpz/swi.wsgi  WSGIImportScript /opt/edware/hpz/swi.wsgi process-group=swi application-group=%{GLOBAL}  SSLEngine on  SSLCertificateFile /etc/pki/tls/certs/{{ ssl\_key\_name }}.crt  SSLCertificateKeyFile /etc/pki/tls/private/{{ ssl\_key\_name }}.key  SSLProtocol All -SSLv2 -SSLv3  SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:D  HE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE  -ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA256:DHE-RSA-AES256-SHA256:DHE-DSS-AE  S256-SHA:DHE-RSA-AES256-SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!3DES:!MD5:!PSK  <LocationMatch ^/download/(.\*)>  Order Allow,Deny  Deny from all  </LocationMatch>  <Directory /opt/virtualenv/hpz>  WSGIProcessGroup swi  Order allow,deny  Allow from all  </Directory>  </VirtualHost>  Listen 8443 |

## conf.d/wsgi\_tsb.conf

|  |
| --- |
| WSGIApplicationGroup %{GLOBAL}  WSGIPassAuthorization On  WSGIDaemonProcess pyramid user=apache group=apache processes=2 threads=30 python-path=/opt/virtualenv/smarter\_score\_batcher/lib/python3.3/site-packages  WSGIScriptAlias / /opt/edware/smarter\_score\_batcher/smarter\_score\_batcher.wsgi  WSGIImportScript /opt/edware/smarter\_score\_batcher/smarter\_score\_batcher.wsgi process-group=pyramid application-group=%{GLOBAL}  WSGISocketPrefix run/wsgi  <Directory /opt/virtualenv/smarter\_score\_batcher>  WSGIProcessGroup pyramid  Order allow,deny  Allow from all  </Directory>  WSGIPythonPath /opt/virtualenv/smarter\_score\_batcher/lib/python3.3/site-packages |

## conf.d/xsendfile.conf

|  |
| --- |
| LoadModule xsendfile\_module modules/mod\_xsendfile.so |

# Appendix B: Service Provider Metadata

|  |
| --- |
| **<?xml version='1.0' encoding='UTF-8'?>**  **<ns0:EntityDescriptor xmlns:ns0="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:ns1="http://www.w3.org/2000/09/xmldsig#" entityID="{{ ENTITY\_NAME }}">**  **<ns0:SPSSODescriptor AuthnRequestsSigned="false" WantAssertionsSigned="true" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">**  **<ns0:KeyDescriptor use="signing">**  **<ns1:KeyInfo>**  **<ns1:X509Data>**  **<ns1:X509Certificate>MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQC6dlOjWkiMqgnI**  **FHDTFRu2VwWawiqRAdyNDtGOjy2TGNvH76cXKpb267Au3gua4bwRDYcK7moUgat+**  **tQh5+ouowNfZPgrRqzDnIoxyi1Pr2dG/1f1pJk36gYwVu0T3iWFqyZLdzNwfrze7**  **3q5re/RIbDn9G7QQq/w1VygAnIuk+3ZxUYvVJYim+Kc0/yDw2WmQqBh+MtSA6UYK**  **6Y1af3Nm8dxllGiqocT/UxEXfoSnwQBzzVeLKswr9fTBSfuodugpMoJPQm/iYWzq**  **70PYdq4iwPPqwSDQ3PrawABylJGR6CY5rS80ehQhLZg3UqveyyJqYZpHaSI3vHTE**  **d8yX3GkJAgMBAAECggEAamcXd0IP4GUvy8MOVwXWmIi0JAnk7HaZes0X/DfsE+K9**  **mCxDt06QczsvgK8pBCsIfNqxUyWV1znfuSBpQ+IbTCMviUKEPf1onJz0dHDZZN8Z**  **eFUOlLBNE0RVwhjpVDBLLPr2GifK/TGppkV+VsuXLLTFqo/igxkBhRbFNwQeE/iH**  **8Z86yNIuiDLc77SEDDsT9Y5VXokyWlk01eAdMJb0vJUjeX4IhtXq5lfnqI8h8LBO**  **Y3ASaJHoOrSyoKf7Nenn2xy10b+ZzICVPBlQXeI42LB/ncBj6RcsDdMm7M/oEZkX**  **Ra7wytJ7CSXbihtLZD2yizY9ZzHBQ5M8QojYhzdH2QKBgQD3Gzqbom0reMpi2zTe**  **PRrsiKDzSJIozn4ZqrEnH0q04ng+9WCid0iEAce7K8oz//S2gHvy/WHpFJ3X2iiq**  **2dzCtLJh+w0HRNt1VsAXLooJAfNTOqfXwZpPsFSsHZTJcZI+Ox/KHiNPvpsoqPmC**  **Ftrc3oDsfQ1pFS8dRNVBM01B6wKBgQDBLFbJzzXiMgSq0h7Jr6EUatguK705H6ii**  **yrn5KthE3cCLot/Z07CId5OXTcA1cJiO+uY83EjHZ5eLvwM5hSUC6CQQg2Bp8U6X**  **Lcnqcq3M4RsBNGeHgYA2rfM0BcQgOoEGi+DudcfoMD1GlJXiY0qtx7o7Rc/bbzdF**  **MiOEH2jP2wKBgQCtWhTLii88CAkzQMTeA9AXj+IZyhd/OR2NtAmDgJhQPXBN+qZI**  **U5YzxWvxCs9Xm4B5Z7mMxeUiNqxk+U7+TMhqb4mU4yJIry2mK62mlaeQmiWxhbyp**  **x9ARh4pfFIGEsH//2Ep86JetIfKHm8MuWApCWHXpcfEufwYGanuQugTA8QKBgCbY**  **fr9ofuJ0woGDbZOLUO4lKkHmo8+n8SaDc/CKUCFRIpYx86f/LOHUggaMKRnWAafH**  **BqO4fJ4008HTcJeRkcbK3e3ALZRAZVh7ab5Ov2qS3fZZDnSucxUL1/tfKXh9q1Kg**  **sGPjpYaopLnbyWRUKUFXB7lBqbHr5IZytbetBUu/AoGAcrU6d+ellgGmMkJMQ6kP**  **iEoiFfgxp7+o5rDPEQBznmnIzgVIA8n81kdkYXcZKOUDemZ72WGcHO3iSVpZyIDW**  **0k5Ar1nBSCr0bZn35WBcFB58es9yY75F0P+tN3HcQ5uzUg7LeF/LIvkK4D39Ol+g**  **NdB8BihIpUAP54CkV1T+WXw=**  **</ns1:X509Certificate>**  **</ns1:X509Data>**  **</ns1:KeyInfo>**  **</ns0:KeyDescriptor>**  **<ns0:SingleLogoutService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="http://{{ HOST\_NAME }}/logout\_redirect" />**  **<ns0:AssertionConsumerService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="http://{{ HOST\_NAME }}/saml\_post" index="1" />**  **</ns0:SPSSODescriptor>**  **</ns0:EntityDescriptor>** |

please update {{ ENTITY\_NAME }} and {{ HOST\_NAME }}. {{ ENTITY\_NAME }} can be any name, but it must be unique in SSO name entries.