

Contents

Purpose.....	4
System Overview	4
Component Software	6
Development/Operation Software Packages	6
Software Packages by Component	7
Deployment Assumptions.....	10
Deployment Configurations.....	10
Test Delivery Unit	11
Elastic Load Balancer	13
Web Server Instance Type	13
AWS ElastiCache – Redis Cluster	16
Spring Cloud Configuration Service	16
RabbitMQ For Spring Cloud Configuration Service	17
S3 Storage	17
Amount of Data Served	17
Database Server Instance Type	17
Database Server Engine Type	18
Database Server Setup and Configuration	19
Database Server Persistent Storage and Provisioned IOPS	19
Database Server Storage and Performance	20
Centralized Logging	20
Elasticsearch service setup	21
Kibana setup with nginx	21
Logstash installation and configuration	22
Long term Elastic Search data maintenance	22
Log File Storage/Archival in S3	22
Test Integration and Scoring Deployment Unit	23
Test Registration and Administration Deployment Unit	23
Assessment Creation and Management Deployment Unit	24
SSO Deployment Unit	26
Monitoring and Alerting Deployment Unit	27
Shared Services Deployment Unit	28

Alternative Deployment Scenario	28
Cost Calculation Spreadsheet.....	30

Figures

Figure 1. Smarter Balanced Contract 11 Components	5
Figure 2: Test Delivery Unit Kubernetes Cluster (note: some services omitted for clarity)	12
Figure 3: RDS Instance deployment for low and high traffic scenarios	19
Figure 4. SSO Configuration	26
Figure 5. Alternative Approach Using Vendor-Specific Test Delivery Platform	29

Tables

Table 1. Development/Operation Software Packages	6
Table 2. Component Categories	7
Table 3. Components and Development/Operation Software Packages	9
Table 4. Summary of Deployment Groups	11
Table 6. EC2 Instance Classes.....	18
Table 7. Test Integration and Scoring Deployment Unit Summary.....	23
Table 8. Test Registration and Administration Deployment Unit Summary.....	24
Table 9. Assessment Creation and Management Deployment Unit Summary	25
Table 11. Monitoring and Alerting Deployment Unit	28
Table 12. Shared Services Deployment Unit Summary	28
Table 14. Example Cost Summary from Spreadsheet	32

Purpose

This document is to assist states, software vendors and systems integrators in planning for delivery of Smarter Balanced assessments. The first year of operational testing will be in the 2014-2015 school year.

The Smarter Balanced interim and summative test system is composed of the following components:

- Assessment Creation & Management
- Assessment Delivery
- Assessment Reporting
- Shared Services

In addition to these components, Smarter Balanced is developing a Digital Library that will support teachers in planning formative assessment activities and in using the results of Smarter Balanced assessments to inform their practice.

The Smarter Balanced Assessment Consortium will host all the above components except Assessment Delivery. The consortium will release an open source implementation of the assessment delivery system. States will be responsible for procuring assessment delivery from vendors who are certified to deliver Smarter Balanced assessments or for deploying their own instance of the open source system.

This document describes the hosting requirements of the open source implementation of the Test Delivery System (TDS) provided by Smarter Balanced.

System Overview

The following diagram is Figure 4.2 from the [Smarter Balanced System Architecture and Technology Report](#) dated 21 March 2012. It depicts the components in the Smarter Balanced system. The components marked with an X are not part of the Test Delivery System and are developed by other Smarter Balanced vendors. Only the Smarter Balanced TDS components and supporting shared services (not marked with an X) will be analyzed in this document.

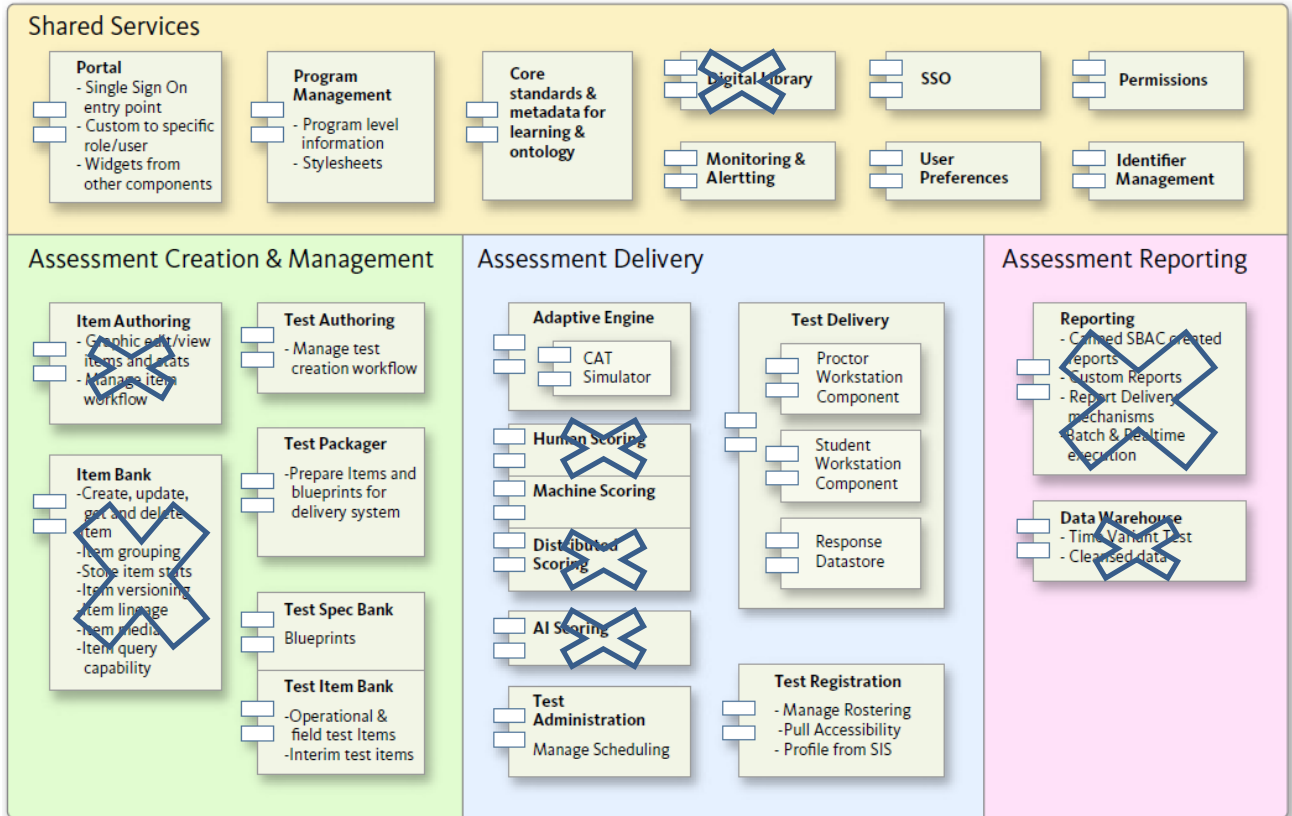


Figure 1. Smarter Balanced Test Delivery System and Shared Services Components

In addition to the above components, Smarter Balanced may authorize the development of two additional components: Test Integration and Test Scoring. Test Integration is responsible for receiving assessments from Test Delivery once the student completes real-time interactive portion of the assessment. It is responsible for sending items, rubrics and responses to various scoring engines including Hand Scoring and Distributed Scoring, and integrating these item scores with the items scored during the student assessment. The Test Scoring component is responsible for computing final test scores including scale scores.

Component Software

Development/Operation Software Packages

The following table summarizes the software technologies used by each component.

Software	Description
Oracle Java JDK	Software development language/platform
WordPress	WordPress Content Management System
Apache	Web server
Tomcat	Java Application Server
Undertow	High-performance web server
MySQL	Relational database engine
PHP	Software development language/platform
Spring Framework	Java development framework
Javascript	Client side scripting language
MongoDB	Non-relational database engine
Hyperic Server	Enterprise server monitoring
OpenOffice Calc	Open source spreadsheet application
OpenAM	Identity Management
OpenDJ	LDAP directory
Kubernetes	Deployment/orchestration software
Docker	Linux container management
S3	Cloud storage service provided by AWS

Table 1. Development/Operation Software Packages



Software Packages by Component

Tables 2 summarizes the component categories as described by Figure 1. Smarter Balanced Test Delivery System and Shared Services Components. The category identifiers will be used in the table below for brevity.

Category Identifier	Component Category
1	Shared Services
2	Assessment Creation & Management
3	Assessment Delivery

Table 2. Component Categories

Table 3 summarizes the software components and the software technologies use in the development and operation of each component.

Category	Component	Java	Spring Framework	Javascript	WordPress	MySQL	MongoDB	PHP	OpenOffice Calc	Hyperic Server	Open AM	Open DJ	Kubernetes	Docker	S3
1	Portal	✓	✓		✓	✓		✓							
1	Program Management	✓	✓				✓								
1	Core Standards	✓	✓			✓			✓						
1	Monitoring and Alerting	✓	✓				✓			✓					
1	Single Sign On	✓	✓								✓	✓			
1	Permissions	✓	✓			✓						✓			
1	User Preferences	✓	✓				✓								



Hosting Requirements
 Smarter Balanced Assessment Consortium
 Test Delivery System

Category	Component	Java	Spring Framework	Javascript	WordPress	MySQL	MongoDB	PHP	OpenOffice Calc	Hyperic Server	Open AM	Open DJ	Kubernetes	Docker	S3
1	Identifier Management	✓	✓				✓								
2	Test Authoring	✓	✓												
2	Test Packager	✓	✓			✓									
2	Test Spec Bank	✓	✓			✓									
2	Test Item Bank	✓	✓				✓								
3	Test Delivery	✓	✓			✓									
3	Proctor Application			✓									✓	✓	
3	Student Application			✓									✓	✓	
3	Adaptive Engine	✓	✓			✓									
3	CAT Simulator	✓	✓			✓									
3	Machine Scoring	✓	✓												
3	Test Registration	✓	✓			✓					✓	✓			

Category	Component	Java	Spring Framework	Javascript	WordPress	MySQL	MongoDB	PHP	OpenOffice Calc	Hyperic Server	Open AM	Open DJ	Kubernetes	Docker	S3
3	Test Administration	✓	✓			✓									
3	Test Integration	✓	✓			✓									
3	Test Scoring	✓	✓			✓									
3	Exam Results Transmitter Service	✓	✓			✓							✓	✓	
3	Exam Service	✓	✓			✓							✓	✓	✓
3	Assessment Service	✓	✓			✓							✓	✓	
3	Config Service	✓	✓			✓							✓	✓	
3	Session Service	✓	✓			✓							✓	✓	
3	Student Service	✓	✓			✓							✓	✓	
3	Spring Cloud Configuration Service	✓	✓			✓							✓	✓	

Table 3. Components and Development/Operation Software Packages

Deployment Assumptions

This document assumes the following:

1. The deployment of the Smarter Balanced open source components being developed as part of various Smarter Balanced contracts, but focuses only on components being developed as part of the Test Delivery System.
2. Deployment on Amazon Web Services (AWS), including (but not exclusive to) AWS services such as Elastic Compute Cloud (EC2), Amazon Relational Database Service (RDS) and ElastiCache. AWS is a robust, scalable and ubiquitous cloud infrastructure that provides virtual computing resources of a range of capabilities and prices.
3. Working familiarity with [Kubernetes](https://kubernetes.io/)¹ and [Docker](https://www.docker.com/)² for interacting with the Test Delivery Unit

Please note that the use of Amazon’s AWS services is not strictly required, and other cloud infrastructures could be used instead of AWS. This document only analyzes the use of AWS as a reference deployment for pricing purposes.

Deployment Configurations

The deployment of the Smarter Balanced open source software is divided into several deployment units. Each deployment unit is meant to isolate several components and provide a computing environment tailored to their needs. Table 4 summarizes the deployment units.

Deployment Unit	Components Deployed	Description
Test Delivery	Test Delivery, Adaptive Engine, Machine Scoring (real-time item scoring only), Exam Service, Exam Results Transmitter Service, Assessment Service, Session Service, Student Service, Cloud Configuration Service	The Test Delivery unit isolates the components that are responsible for interactive real-time management of student test sessions
Centralized Logging	ElasticSearch, LogStash, Kibana ³	A centralized logging mechanism to record log entries from the various services that comprise the Test Delivery Unit
Test Integration and Scoring	Test Integration, Test Scoring	The Test Integration and Scoring unit works closely with Test Delivery to manage student assessments once the interactive real-time aspect of a student assessment is complete

¹ <https://kubernetes.io/>

² <https://www.docker.com/>

³ ElasticSearch, LogStash and Kibana are not listed in the component table above because they are third-party software components; they are not part of the Test Delivery Unit source code

Deployment Unit	Components Deployed	Description
Test Registration and Administration	Test Registration, Test Administration	The Test Integration and Scoring unit manages all aspects of student test registration and administration that are not already managed by the Test Delivery unit
Assessment Creation and Management	Test Authoring, Test Packager, Test Spec Bank, Test Item Bank (Item Authoring and Item Bank from Smarter Balanced contract 07 may also be deployed here)	The Assessment Creation and Management unit provides a computing environment for components that are used to create student assessments
SSO	SSO	This deployment unit isolates components critical to the authentication and authorization function
Monitoring and Alerting	Monitoring and Alerting	Monitoring and Alerting is isolated into its own unit to isolate other components from the significant system logging traffic
Shared Services	Portal, Program Management, Core Standards, Permissions	The Shared Services unit houses all remaining Shared Services components

Table 4. Summary of Deployment Groups

Test Delivery Unit

The primary component of the Test Delivery Unit is a [Kubernetes](#) cluster that hosts the applications necessary to serve assessments to students. The Kubernetes cluster consists of several EC2 instances that host groups of Docker containers (referred to as “[pods](#)”) that comprise various parts of the Test Delivery System (TDS):

- **Exam Service:** Manages data a student works through an exam
- **Exam Results Transmitter (ERT) Service:** Delivers the Test Results Transmission (TRT) file of the completed exam to the Test Integration System (TIS)
- **Assessment Service:** Manages data for an assessment (i.e. a test that has been loaded into TDS)
- **Config Service:** Manages configuration data for the TDS
- **Session Service:** Manages data for the proctored session
- **Student Service:** Manages student-related data
- **Spring Cloud Configuration Service:** Manages configuration properties for the services cited above
- **Student Application WAR:** A Docker container for the TDS Student application (user interface for the students to take exams)
- **Proctor Application WAR:** A Docker container for the TDS Proctor application (user interface for proctoring an exam session)
- **RabbitMQ:** A message queue/broker service for propagating configuration changes to the subscribing services

Supporting the Kubernetes cluster is an AWS ElastiCache instance for storing static data in an in-memory data store. Leveraging AWS ElastiCache allows the TDS quick access to static data, mitigating the need to repeatedly query the TDS databases for the same records.

The diagram in figure 2 represents the components of the Test Delivery deployment unit. We will use the Test Delivery deployment unit to illustrate features of the Amazon EC2 and RDS services.

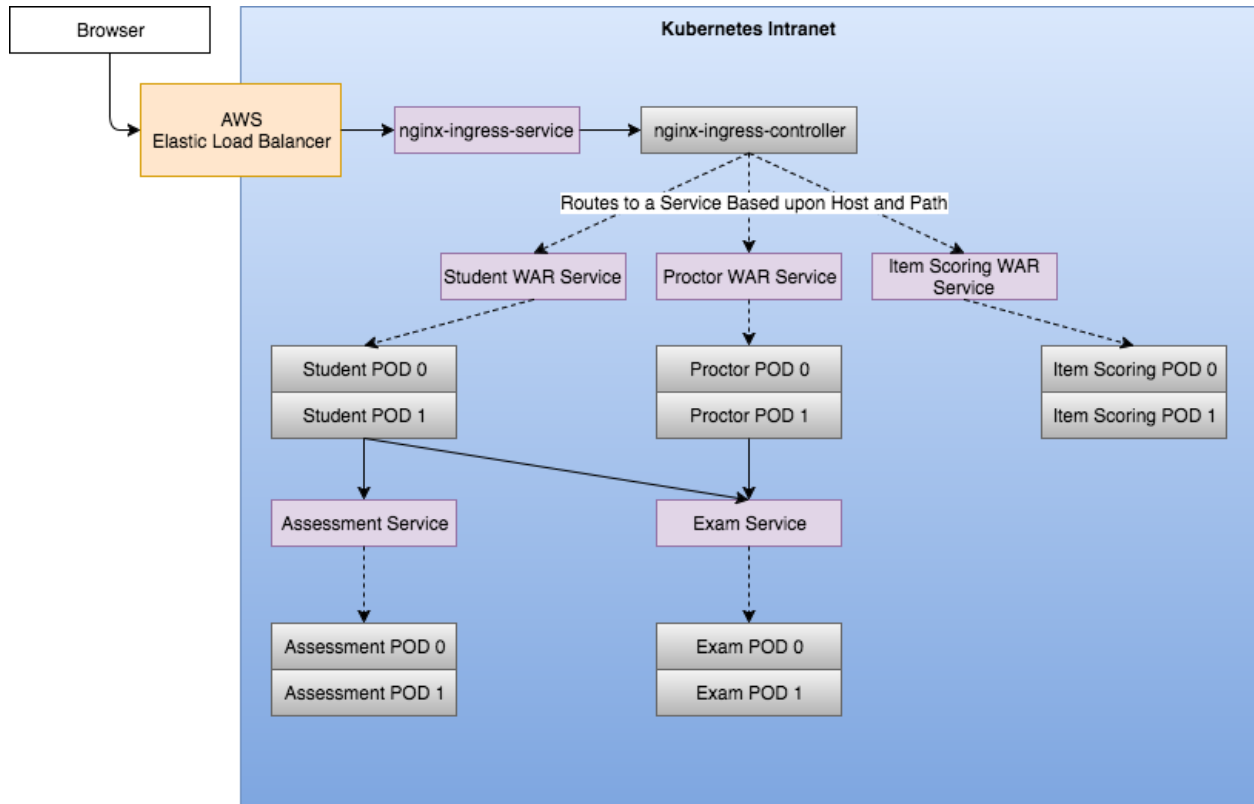


Figure 2: Test Delivery Unit Kubernetes Cluster (note: some services omitted for clarity)

A typical Test Delivery System request flow is as follows:

- A student requests a page of questions in the user interface
- The request is intercepted by an AWS Elastic Load Balancer, which routes the request to an nginx ingress controller
- The [nginx](https://www.nginx.com/resources/wiki/)⁴ ingress controller routes the request to the correct application (student, in this case)
- The service pod (student WAR service in this example) collects the data it needs and composes the response
- The response from the pod is returned through the AWS Elastic Load Balancer to the student's browser

⁴ <https://www.nginx.com/resources/wiki/>

Elastic Load Balancer

Elastic Load balancer costs are determined by the hour and by the amount of data served by the web servers through the load balancer. See [Section 0 Amount of Data Served](#) for more details on calculating the amount of data.

Web Server Instance Type

Each virtual computer created in the cloud is known as an *instance*. Amazon EC2 currently offers the following instance types. Table 5 below describes the EC2 instance types that are available ([source](#)⁵). Kubernetes [makes the following recommendation](#)⁶ when choosing the EC2 instances to host the Test Delivery Unit:

[Kubernetes] generally recommend[s] the m3 instances over the m4 instances, because the m3 instances include local instance storage. Historically local instance storage has been more reliable than AWS EBS, and performance should be more consistent. The ephemeral nature of this storage is a match for ephemeral container workloads also!

*If you use an m4 instance, or another instance type which does not have local instance storage, you may want to increase the **NODE_ROOT_DISK_SIZE** value, although the default value of 32 is probably sufficient for the smaller instance types in the m4 family.*

Instance Type	vCPU	Memory (GiB)	Storage (GB)	Enhanced Networking	Networking Performance	Physical Processor	Clock Speed (GHz)	Intel® AVX†	Intel® AVX2†	Intel® Turbo	EBS OPT
m4.large	2	8	EBS Only	Yes	Moderate	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes
m4.xlarge	4	16	EBS Only	Yes	High	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes
m4.2xlarge	8	32	EBS Only	Yes	High	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes
m4.4xlarge	16	64	EBS Only	Yes	High	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes
m4.10xlarge	40	160	EBS Only	Yes	10 Gigabit	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes
m4.16xlarge	64	256	EBS Only	Yes	10 Gigabit	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes
m3.medium	1	3.75	1 x 4 SSD	-	Moderate	Intel Xeon E5-2670 v2*	2.5	Yes	-	Yes	-
m3.large	2	7.5	1 x 32 SSD	-	Moderate	Intel Xeon E5-2670 v2*	2.5	Yes	-	Yes	-

⁵ <https://aws.amazon.com/ec2/instance-types/>

⁶ <https://kubernetes.io/docs/getting-started-guides/aws/>

Instance Type	vCPU	Memory (GiB)	Storage (GB)	Enhanced Networking	Networking Performance	Physical Processor	Clock Speed (GHz)	Intel® AVX†	Intel® AVX2†	Intel® Turbo	EBS OPT
m3.xlarge	4	15	2 x 40 SSD	-	High	Intel Xeon E5-2670 v2*	2.5	Yes	-	Yes	Yes
m3.2xlarge	8	30	2 x 80 SSD	-	High	Intel Xeon E5-2670 v2*	2.5	Yes	-	Yes	Yes
c4.large	2	3.75	EBS Only	Yes	Moderate	Intel Xeon E5-2666 v3	2.9	Yes	Yes	Yes	Yes
c4.xlarge	4	7.5	EBS Only	Yes	High	Intel Xeon E5-2666 v3	2.9	Yes	Yes	Yes	Yes
c4.2xlarge	8	15	EBS Only	Yes	High	Intel Xeon E5-2666 v3	2.9	Yes	Yes	Yes	Yes
c4.4xlarge	16	30	EBS Only	Yes	High	Intel Xeon E5-2666 v3	2.9	Yes	Yes	Yes	Yes
c4.8xlarge	36	60	EBS Only	Yes	10 Gigabit	Intel Xeon E5-2666 v3	2.9	Yes	Yes	Yes	Yes
c3.large	2	3.75	2 x 16 SSD	Yes	Moderate	Intel Xeon E5-2680 v2	2.8	Yes	-	Yes	-
c3.xlarge	4	7.5	2 x 40 SSD	Yes	Moderate	Intel Xeon E5-2680 v2	2.8	Yes	-	Yes	Yes
c3.2xlarge	8	15	2 x 80 SSD	Yes	High	Intel Xeon E5-2680 v2	2.8	Yes	-	Yes	Yes
c3.4xlarge	16	30	2 x 160 SSD	Yes	High	Intel Xeon E5-2680 v2	2.8	Yes	-	Yes	Yes
c3.8xlarge	32	60	2 x 320 SSD	Yes	10 Gigabit	Intel Xeon E5-2680 v2	2.8	Yes	-	Yes	-
g2.2xlarge	8	15	1 x 60 SSD	-	High	Intel Xeon E5-2670	2.6	Yes	-	Yes	Yes
g2.8xlarge	32	60	2 x 120 SSD	-	10 Gigabit	Intel Xeon E5-2670	2.6	Yes	-	Yes	-

Instance Type	vCPU	Memory (GiB)	Storage (GB)	Enhanced Networking	Networking Performance	Physical Processor	Clock Speed (GHz)	Intel® AVX†	Intel® AVX2†	Intel® Turbo	EBS OPT
x1.32xlarge	128	1,952	2 x 1,920 SSD	-	10 Gigabit	Intel Xeon E7-8880 v3	2.3	Yes	Yes	Yes	Yes
r3.large	2	15.25	1 x 32 SSD	Yes	Moderate	Intel Xeon E5-2670 v2	2.5	Yes	-	Yes	-
r3.xlarge	4	30.5	1 x 80 SSD	Yes	Moderate	Intel Xeon E5-2670 v2	2.5	Yes	-	Yes	Yes
r3.2xlarge	8	61	1 x 160 SSD	Yes	High	Intel Xeon E5-2670 v2	2.5	Yes	-	Yes	Yes
r3.4xlarge	16	122	1 x 320 SSD	Yes	High	Intel Xeon E5-2670 v2	2.5	Yes	-	Yes	Yes
r3.8xlarge	32	244	2 x 320 SSD	Yes	10 Gigabit	Intel Xeon E5-2670 v2	2.5	Yes	-	Yes	-
i2.xlarge	4	30.5	1 x 800 SSD	Yes	Moderate	Intel Xeon E5-2670 v2	2.5	Yes	-	Yes	Yes
i2.2xlarge	8	61	2 x 800 SSD	Yes	High	Intel Xeon E5-2670 v2	2.5	Yes	-	Yes	Yes
i2.4xlarge	16	122	4 x 800 SSD	Yes	High	Intel Xeon E5-2670 v2	2.5	Yes	-	Yes	Yes
i2.8xlarge	32	244	8 x 800 SSD	Yes	10 Gigabit	Intel Xeon E5-2670 v2	2.5	Yes	-	Yes	-
d2.xlarge	4	30.5	3 x 2000	Yes	Moderate	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes
d2.2xlarge	8	61	6 x 2000	Yes	High	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes
d2.4xlarge	16	122	12 x 2000	Yes	High	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes
d2.8xlarge	36	244	24 x 2000	Yes	10 Gigabit	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes

M3 instances may also launch as an Intel Xeon E5-2680 v2 (Ivy Bridge) Processor running at 2.5 GHz.

Table 5. EC2 Instance Types

The general purpose m3.large instance type is selected for Kubernetes master and worker nodes based on the recommendation cited above. The m3.large instance represents a reasonable trade-off of compute capacity, memory, storage, and cost. EC2 instances incur costs for each hour (or fraction thereof) the virtual computer is powered up and operating.

In addition to selecting the web server instance type, the web server operating system is also selected. Amazon EC2 offers preconfigured Amazon Machine Instances (AMI) to select from. AMIs come in many operating systems and various flavors of each operating system. Smarter Balanced components are intended to run on the Linux operating system, and Linux comes prepackaged in

various flavors known as distributions. Some of these distributions are enterprise versions that require paid support (Red Hat and SUSE Enterprise, for example) and some are community versions that are free to download and use (Fedora, CentOS, for example). The Ubuntu Server distributions are the same for enterprise and community users but offer paid support as an option.

Some enterprise distributions place restrictions on changing any of the software components provided in the distribution, and these are updated only when the distribution is updated in periodic release cycles. This means that certain components such as the Java Virtual Machine (JVM) may be older and may lack desirable features available in later versions. With Red Hat and SUSE Enterprise Linux, there is no option to upgrade components without voiding the support agreement.

Canonical Inc., the company that packages the Ubuntu Linux distribution, allows modifications to the preloaded components and provides cloud support for up to 100 virtual cloud servers for a reasonable cost. For this reason, Ubuntu Linux distribution is recommended for this project.

The Docker containers for the components described in Table 3 with a Docker dependency are based on a lightweight Docker image built on the [Alpine Linux](#)⁷ distribution with a Java 8 JDK.

AWS ElastiCache – Redis Cluster

The applications that comprise the Test Delivery Unit rely heavily on cached data to maximize performance. AWS ElastiCache provides a simple mechanism for creating a [Redis](#)⁸ cluster for caching data. The Docker containers within the Kubernetes cluster communicate with the Redis cluster instead of repeatedly querying the database for data that rarely (if ever) changes.

Spring Cloud Configuration Service

The Spring Cloud Configuration service provides configuration information for the other services that participate in the Kubernetes cluster. The configuration values are YAML files describing the properties required by the TDS services. These files are securely stored in source control via a private repository. [GitLab](#)⁹ offers free private repositories, [GitHub](#)¹⁰ offers private repositories with a paid subscription. Other version control systems/software (e.g. BitBucket) may be available; refer to the [Spring Cloud](#)¹¹ documentation for additional details.

After a provider is chosen, an account with sufficient privileges must be created and configured within the Spring Cloud Configuration service so the property values can be fetched. The services will query the Spring Cloud Configuration Service to get the configuration properties they need when they start up.

For storing sensitive values (e.g. database connection passwords), the [Spring Cloud CLI](#)¹² program can be used for encryption.

⁷ <https://alpinelinux.org/about/>

⁸ <https://redis.io/>

⁹ <https://about.gitlab.com/>

¹⁰ <https://github.com/>

¹¹ http://cloud.spring.io/spring-cloud-static/spring-cloud.html#_features

¹² <https://cloud.spring.io/spring-cloud-cli/>

RabbitMQ For Spring Cloud Configuration Service

The Spring Cloud Configuration Service depends upon a RabbitMQ instance for handling changes to the configuration files. Changes to configuration files are broadcast to the dependent services by means of the [Spring Cloud Bus](#)¹³, which leverages RabbitMQ for delivering messages.

S3 Storage

The assessments served up to students by the TDS consist of a series of items comprised of files. Each item has at least one XML file; many items consist of multiple files. An S3 bucket is required for storing these assessment items. The Exam service application will fetch item content from the S3 bucket, which will ultimately be served to the browser for display.

Amount of Data Served

For the Test Delivery deployment unit, the data served is calculated as follows:

Total Data Served = Number of concurrent students X average number of items server per student per hour X average size of item served X Overhead factor

For example:

Total Data Served = 15,000 concurrent students X 20 items per student per hour X 50KB per item X 125% / 1024² KB per GB
 = 17.88 GB per hour
 ≈ 18 GB/hour

Data served is charged separately as a standalone expense and as part of the cost of running an Elastic Load Balancer. The amount of data served should be planned based on the peak data demands and pro-rated for the number of hours and days of peak data demand.

Database Server Instance Type

Database server instances are selected from Amazon’s Relational Database Service (RDS). These are high-performance instance types that are preconfigured with appropriate database engines and provide high availability features. For example, RDS instances are available in “Multi-Availability Zone Deployments.” Availability Zones (AZ) are regions where Amazon maintains data centers such as the Easter region in Northern Virginia. These types of instances are provided with a second standby instance in a different AZ in case of AZ failure.

Table 6 describes the DB instance classes that are available ([source](#)¹⁴):

Instance Class	vCPU	ECU	Memory (GiB)	EBS Optimized	Network Performance
Micro Instances					

¹³ http://cloud.spring.io/spring-cloud-static/spring-cloud.html#_push_notifications_and_spring_cloud_bus

¹⁴ <http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Concepts.DBInstanceClass.html>

Instance Class	vCPU	ECU	Memory (GiB)	EBS Optimized	Network Performance
db.t1.micro	1	1	.615	No	Very Low
db.m1.small	1	1	1.7	No	Very Low
Standard - Current Generation					
db.m4.large	2	6.5	8	450 Mbps	Moderate
db.m4.xlarge	4	13	16	750 Mbps	High
db.m4.2xlarge	8	25.5	32	1000 Mbps	High
db.m4.4xlarge	16	53.5	64	2000 Mbps	High
db.m4.10xlarge	40	124.5	160	4000 Mbps	10 GBps
Memory Optimized – Current Generation					
db.r3.large	2	6.5	15	No	Moderate
db.r3.xlarge	4	13	30.5	500 Mbps	Moderate
db.r3.2xlarge	8	26	61	1000 Mbps	High
db.r3.4xlarge	16	52	122	2000 Mbps	High
db.r3.8xlarge	32	104	244	No	10 Gbps

Table 6. EC2 Instance Classes

The Multi-AZ High-Memory Quadruple Extra Large DB Instance is selected for the Test Delivery deployment unit for high performance and high availability.

Database Server Engine Type

The TDS is written to support a MySQL-compliant database server (e.g. MySQL, [MariaDB](https://mariadb.org/)¹⁵, Aurora). When selecting the database engine for the RDS instance(s) that will support the TDS environment, consider how much traffic (i.e. how many concurrent students) are expected. [Amazon Aurora](https://aws.amazon.com/rds/aurora/)¹⁶ is more performant than MySQL at high volume (i.e. over 200,000 concurrent students), thus consider choosing Aurora for deployments that expect many concurrent students.

¹⁵ <https://mariadb.org/>

¹⁶ <https://aws.amazon.com/rds/aurora/>

Database Server Setup and Configuration

For a deployment that expects a small amount of traffic (i.e. few concurrent students), hosting the database for the Exam service on the same RDS instance as the other TDS database may be sufficient. For a Test Delivery unit that expects high volume (i.e. many concurrent students), the database for the Exam service should be deployed to a dedicated RDS instance. A student taking an exam is a data-intensive process, thus allowing the Exam service database a dedicated RDS instance ensures enough resources are available for the processes to execute quickly.

Figure 3 is a high-level diagram describing how a low-traffic and high-traffic TDS environment would be set up

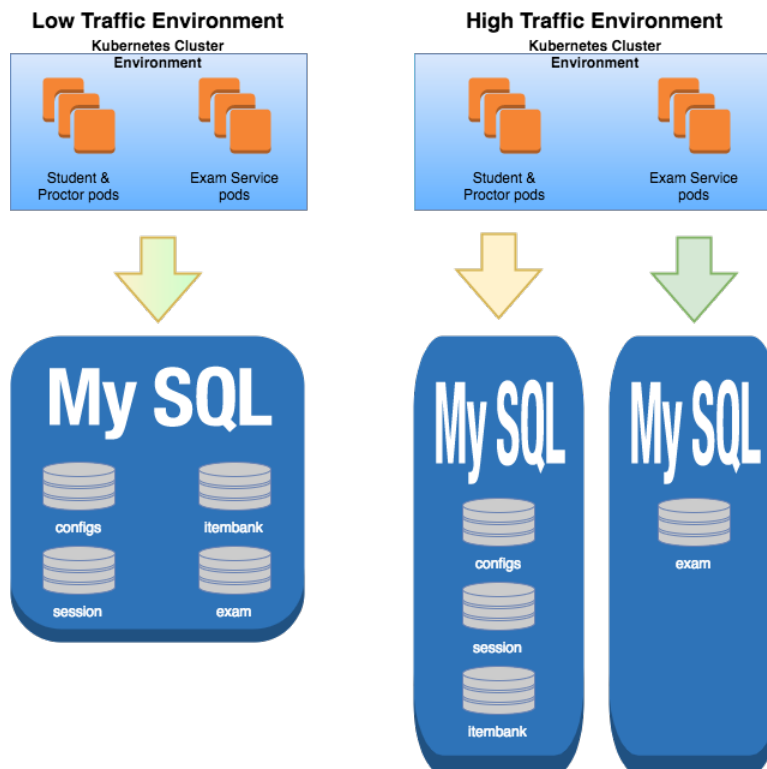


Figure 3: RDS Instance deployment for low and high traffic scenarios

Database Server Persistent Storage and Provisioned IOPS

Database server persistent storage can be selected in sizes between 100GB and 6TB, and with up to 30,000 provisioned IOPS. The amount of provisioned IOPS for MySQL should be selected to be within a 3:1 and 10:1 ratio of IOPS to storage size. For example, a 100 GB storage size should be provisioned with an IOPS between 300 (3/1) and 1,000 (10/1) IOPS.

When selecting database instances for use with Provisioned IOPS, Amazon recommends the following:

If you are using Provisioned IOPS storage, we recommend that you use the M4, M3, R3, and M2 DB instance classes. These instance classes are optimized for Provisioned IOPS storage; other instance classes are not.

http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Storage.html

For this reason, only the four instance types are selected for database servers when high transactional performance is required. When the highest performance is required, only one of the four top instances mentioned above is selected.

Each of these database instances provides high performance and scale, but the scale is limited by the speed of the persistent database storage. The following applies to MySQL:

The following table shows the page size and the theoretical maximum IOPS rate for each DB engine. IOPS rates are based on the m2.4xlarge instance class with full duplex and a workload that is perfectly balanced between reads and writes. The SQL Server limit of 10,000 is due to the current storage limit of 1 TB and the current maximum IOPS to storage ratio of 10:1.

DB Engine	Page Size	Maximum IOPS Rate
MySQL	16 KB	30,000
Oracle	8 KB	25,000
SQL Server	8 KB	20,000

http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Storage.html

Database Server Storage and Performance

In the high-traffic deployment scenario (i.e. where the Exam service database is deployed to a dedicated RDS instance), the following metrics were recorded during a 200,000-concurrent student load test:

- The MySQL RDS instance hosting the configs, itembank and session databases peaked at 1055 write IOPS per second
- The MySQL RDS instance hosting the exam database peaked at 2483 IOPS per second

To support 300,000 students, the RDS was switched to use Aurora, which does not provide metrics for write IOPS per second. The RDS instance hosting the configs, itembank and session databases remained on MySQL and peaked at 1003 write IOPS per second.

Centralized Logging

Due to horizontal scaling, finding log messages is problematic. It is difficult to find which instance of an application contains the relevant log entry of an issue. To simplify troubleshooting, log entries from multiple applications and application instances can be sent to a single, searchable location.

The "[ELK](#)¹⁷" stack, which stands for Elasticsearch (ES), Logstash, and Kibana was leveraged for centralized logging. AWS has a packaged ES service, which includes Kibana as a plugin. Logstash was installed manually with the instructions below. On AWS, ES was configured with IP-based security to only talk to our logstash hosts. The Nginx reverse-proxy forwards kibana web requests over to the Kibana plugin.

An easy way to run a load balanced ES cluster of any size is by using AWS' Elasticsearch service. AWS has wrapped Elasticsearch with a user interface that allows a user to easily create, re-configure, snapshot and monitor Elasticsearch. When setting up a Test Delivery Unit deployment decide how many nodes of what class are desired and how much disk space is required. These values can vary greatly depending on expected traffic. After being provided with configuration values, AWS conducts the setup. When setup is complete, AWS provides a single endpoint to communicate with the cluster.

The cluster can be resized and/or reconfigured while operating with no service interruption. AWS will create a new cluster with the updated configuration, migrate the data over and decommission the old cluster without any downtime or loss of data.

Elasticsearch service setup

To set up the AWS ES service, navigate to the 'Amazon Elasticsearch Service dashboard' in the AWS account dashboard and click 'Create a new domain'. Fill in the form by providing the required information. In our config, we're using IP-based security. To do this, select the 'allow access from specific IP' template during setup and put your LogStash EC2 host IP addresses in the form. Elasticsearch version 5.3 is recommended - it is the first version that supports the curator tool, described below.

An ES cluster consists of two types of nodes: data nodes and master nodes. Having dedicated master nodes is highly recommended by AWS for performance. Three is the default number of master nodes, and a good value to start with. For a high-volume deployment, Fairway recommends the m4.large.elasticsearch node type for all master and data nodes, but experimentation could be done to save cost or increase performance in your cluster. Another option is to leverage c4.large for the masters, as they need good compute and network speed but less storage performance than the data nodes.

For a high-volume deployment, each data node can be assigned 300GB, giving a 1.2TB total data size for the cluster. Storage values can be configured at runtime. Changes can require 15-30 minutes to process, depending on the data sizes being migrated. The cluster remains live and responsive throughout the migration process.

Any time the cluster configuration is changed, it will show a status of 'processing' until the new cluster is set up. Once complete the cluster will show 'active'. At this point you can access the endpoint and Kibana URL's that are given on the Cluster Details page in the dashboard

Kibana setup with nginx

As the ES cluster will only listen to the provided IP, a proxy server (e.g. nginx) should be set up to route Logstash requests to the kibana plugin in the ES cluster. The nginx server should be configured with security practices that comply with the organization's standards.

¹⁷ <https://qbox.io/blog/welcome-to-the-elk-stack-elasticsearch-logstash-kibana>

Logstash installation and configuration

LogStash can be installed via the operating system's package management software (e.g. apt for Ubuntu or yum for RHEL/CentOS). When configuring LogStash for the Test Delivery Unit deployment, ensure the LogStash configuration does not directly interact with the Elasticsearch master nodes. The LogStash nodes should be configured to communicate with the Elasticsearch data nodes. Refer to the documentation in [this link](#)¹⁸ for more details.

Long term Elastic Search data maintenance

To prevent the Elasticsearch cluster from reaching capacity, old data must be removed. One tool that makes this easy is called 'curator', a Python tool supported by Elasticsearch. Shown below are simple curator actions that can help with Elasticsearch cluster maintenance. Documentation for the curator application can be found [here](#)¹⁹, and documentation for configuring curator to suit environment needs can be found [here](#)²⁰.

For example, curator can be configured to delete all indices over 90 days old, and close all indices over 45 days old. These values can be configured in the action file. Closed indices have virtually no runtime cost to Elasticsearch, but can easily be opened again if required. Deleted indices are completely removed and not recoverable.

After curator is configured, it should be set up to run at a regular interval (e.g. a daily crontab) from an Elasticsearch master, Kibana server, or other host with access to your Elasticsearch cluster.

Alternately, Elasticsearch can expire stale data via a time-to-live (TTL) value for each index. Elasticsearch TTL setup/configuration is outside the scope of this document, but easily accomplished and is automatic once set. The curator method described here is preferred; it takes effect regardless of the individual metadata settings inside each elastic search index, thus is considered to be more reliable in bulk. The values in the curator cleanup action file can also be changed at will and made effective immediately by executing the curator. The TTL can be configured to a long period as a failsafe against very old data remaining in the data store in case the curator crontab fails to run.

Log File Storage/Archival in S3

LogStash Log entries are stored in an S3 bucket for permanent archival, including all instrumented event data (see the logstash configuration section above). Although S3 can hold an unlimited number of files, it may be desirable to change their storage class to STANDARD_IA as they age, or to move them to Glacier or another service. Changing the storage class and moving files to Glacier are described [here](#)²¹. Such an operation is outside the scope of this document.

¹⁸ <https://www.elastic.co/guide/en/logstash/5.0/deploying-and-scaling.html>

¹⁹ <https://www.elastic.co/guide/en/elasticsearch/client/curator/current/index.html>

²⁰ <https://www.elastic.co/guide/en/elasticsearch/client/curator/current/configfile.html>

²¹ <http://docs.aws.amazon.com/AmazonS3/latest/dev/storage-class-intro.html>

Test Integration and Scoring Deployment Unit

The Test Integration and Scoring deployment unit supports the Test Delivery deployment unit with test integration and scoring services. This unit receives the student assessment once the interactive real-time highly transactional part of the student assessment is complete. This unit receives results from hand scoring and other scoring engines and integrates these scores with the other assessment items scored in real time during the student assessment. Once a student assessment is completely integrated, the assessment is scored (including scale scores) according to the configured scoring rules.

The following is a summary of the Test Integration and Scoring deployment unit selections.

Deployment Selection	Description
Elastic Load Balancer	An Elastic Load Balancer is required for this deployment unit.
Web Server Instance Type	An EC2 general purpose m1.xlarge instance is selected (64 bit, 4 virtual CPUs, 15GB of memory, 4x420GB of ephemeral storage, available EBS storage, and high network performance).
Number of Baseline Web Servers	Same scaling model as Test Delivery: two baseline web servers are selected for high availability and performance.
Number of On-Demand Web Servers	Same scaling model as Test Delivery: one on-demand web server is selected for increased load.
Amount of Data Served	Based on the number of concurrent students, the number of assessments completed per hour, the size of each assessment unit and an overhead factor. For example, for 10,000 concurrent students, one student assessment per hour, an assessment size of 20KB and an overhead factor of 25%, we have = 10,000 students X 50KB per student X 125% = 0.60GB/hour, or about 1 GB per hour.
DB Server Instance Type	A Multi-AZ Extra Large RDS database instance is selected. This server is not as large as the DB server selected for Test Delivery but reflects the lower activity expected for this server as compared to Test Delivery. It is optimized for Provisioned IOPS.
DB Server Storage	This storage size is selected based on the total number of students, the size of each test opportunity, the number of test opportunities, and an overhead figure. For example, = 500,000 students X 50KB per test opportunity X 3 opportunities per student X 125% overhead = 89.4GB Select a 200GB persistent store. Please note that this size will vary with the expected size of the deployment. A deployment with more students will require additional storage.
DB Server Provisioned IOPS	A 3/1 ratio of provisioned IOPS is selected. 200GB X 3 = 600 Provisioned IOPS

Table 7. Test Integration and Scoring Deployment Unit Summary

Test Registration and Administration Deployment Unit

The 5.3 Test Registration and Administration deployment unit provides services for upload of entity, user and student files and a user interface for ad hoc modification of user roles and student demographics and PNP information. The following selections are made for this deployment unit.

Deployment Selection	Description
Elastic Load Balancer	An Elastic Load Balancer is required for this deployment unit.
Web Server Instance Type	An EC2 general purpose m1.xlarge instance is selected (64 bit, 4 virtual CPUs, 15GB of memory, 4x420GB of ephemeral storage, available EBS storage, and high network performance).
Number of Baseline Web Servers	Two baseline web servers are selected for high availability and performance.
Number of On-Demand Web Servers	One on-demand web server is selected for periods of increased load during student registration and testing time.
Amount of Data Served	Test Registration and Administration functions consume a significant amount of data from various sources, but sends a similar amount of data to SSO, Test Delivery and Data Warehouse. For example, = 500,000 students X 50KB per student (leveled over 6 hours per day and 60 days per year) * 125% overhead for users and entities * 125% overhead
DB Server Instance Type	A Multi-AZ Extra Large RDS database instance is selected (IOPS optimized).
DB Server Storage	The database server storage size is proportional to the total number of students, the size of an average student record, an overhead factor to account for users and entities as a proportion of students, and the database overhead of storing a student record. For example: = 500,000 students X 50KB per student record X 125% user/entity factor X 125% database overhead factor = 37.25 GB, assume a safe size of database storage of 200GB
DB Server Provisioned IOPS	600 Provisioned IOPS are required for a factor of 3 to 1 Provisioned IOPS to storage size.

Table 8. Test Registration and Administration Deployment Unit Summary

Assessment Creation and Management Deployment Unit

This deployment unit is only required for entities that wish to have their own item authoring, test authoring and test packaging capabilities. The following selections are made for this deployment unit.

Deployment Selection	Description
Elastic Load Balancer	An Elastic Load Balancer is required for this deployment unit.
Web Server Instance Type	An EC2 general purpose m1.xlarge instance is selected (64 bit, 4 virtual CPUs, 15GB of memory, 4x420GB of ephemeral storage, available EBS storage, and high network performance).
Number of Baseline Web Servers	Two baseline web servers are selected for high availability and performance.
Number of On-Demand Web Servers	One on-demand web server is selected for periods of increased load during student registration and testing time.
Amount of Data Served	Predicting the amount of data served by this deployment unit is TBD at this time. Assume 5 GB/hour for the time being.

Deployment Selection	Description
DB Server Instance Type	A Multi-AZ Extra Large RDS database instance is selected (IOPS optimized).
DB Server Storage	<p>DB server storage for this deployment unit varies based on the number of items and the average size of each item. Various multipliers apply such as:</p> <ul style="list-style-type: none"> • A multiplier to account for various versions of an item • Number of test packages items are used in (items are replicated into each test package) • Database overhead <p>For example, = 200,000 items X 50 KB per item X 5 item versions X 10 copies in Test Packages * 125% DB OH / 1024² KB per GB ≈ 597 KB. Assume 1,000 GB.</p>
DB Server Provisioned IOPS	3,000 IOPS for a ratio of 3 to 1 Provisioned IOPS to storage size.

Table 9. Assessment Creation and Management Deployment Unit Summary

SSO Deployment Unit

The SSO deployment unit is different than other deployment units in that it utilizes an open source identity management system, OpenAM. Each instance of OpenAM includes the OpenDJ LDAP directory server. Therefore, no separate DB server instance is required. However, each web server instance requires persistent storage. This is different from web servers in other deployment groups.

The following diagram illustrates an SSO deployment unit in a high-availability configuration using three OpenAM and three OpenDJ servers. AIR recommends two servers each for most installations, and three servers each for a very high demand installation. The Field Test configuration uses three servers each for maximum performance.

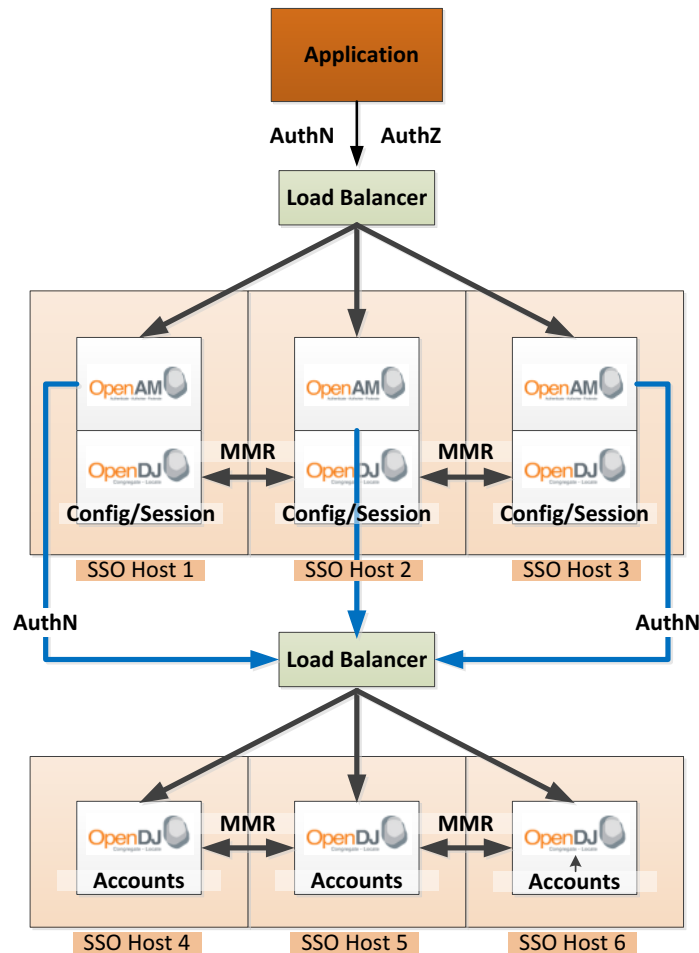


Figure 4. SSO Configuration

Deployment Selection	Description
Elastic Load Balancer	Two Elastic Load Balancers are required for this deployment unit. SSL terminates at the load balancer exposed to the internet, and the second load balancer balances traffic for the OpenDJ LDAP servers.

Deployment Selection	Description
Web Server Instance Type	An EC2 general purpose 2nd generation double extra large instance is selected (64 bit, 8 virtual CPUs, 30GB of memory, EBS storage only, and high network performance).
Number of Baseline Web Servers	Two baseline web servers are selected for high availability and performance.
Number of On-Demand Web Servers	One on-demand web server is selected for periods of increased load during student registration and testing time.
Amount of Data Served	This varies according to the peak number of authentications per hour, the size in KB of each authentication, and an overhead factor. For example, = 10,000 authentications per hour X 50 KB per authentication X 150% overhead / 1024 ² GB per MB ≈ 1 GB/s
DB Server Instance Type	No DB server is required
DB Server Storage	Although no DB server storage is required, storage for the LDAP data store is required in each web server instance. For example, = 50,000 users X 50 KB per user X 150% DB overhead ≈ 4GB. Assume 50GB.
DB Server Provisioned IOPS	500 IOPS for a ratio of 10 to 1 Provisioned IOPS to storage size for peak LDAP performance. Also include 200GB of Amazon S3 storage for snapshots for backup purposes (this is always included in the RDS database instance storage but is extra in standard EBS storage).

Table 10. SSO and Permissions Deployment Unit Summary

Monitoring and Alerting Deployment Unit

Monitoring and Alerting is similar to SSO and Permissions in that it uses and off the shelf open source monitoring solution called Hyperic for its function. It also has a custom component for storing logs and alerts. Therefore, no RDS database instance is required. Also, since Hyperic doesn't require multi-instance scaling (one instance is adequate for the number of servers in question), only one web server with persistent storage is required.

Deployment Selection	Description
Elastic Load Balancer	No load balancer required.
Web Server Instance Type	An EC2 general purpose m1.xlarge instance is selected (64 bit, 4 virtual CPUs, 15GB of memory, 4x420GB of ephemeral storage, available EBS storage, and high network performance).
Number of Baseline Web Servers	One baseline web server is required.
Number of On-Demand Web Servers	No additional on-demand web servers are required.
Amount of Data Served	This component is primarily a data sink rather than a data source. Assume 1GB/hour out for nominal usage.

Deployment Selection	Description
DB Server Instance Type	A standard EC2 instance with EBS block storage will be used instead of a dedicated RDS database server instance. This choice is made because Hyperic has an embedded database and does not rely on RDS virtual hardware dedicated to a MySQL database.
DB Server Storage	Although no DB server storage is required, storage for Hyperic and custom log and alert data is required. This is difficult to predict, assume 250GB of standard EBS storage (as opposed to RDS DB server storage).
DB Server Provisioned IOPS	750 IOPS for a ratio of 3 to 1 Provisioned IOPS to storage size. Also include 350GB of Amazon S3 storage for snapshots for backup purposes (this is always included in the RDS database instance storage but is extra in standard EBS storage).

Table 11. Monitoring and Alerting Deployment Unit

Shared Services Deployment Unit

This deployment unit is used to house the remaining shared services other than SSO and Monitoring and Alerting that do not require the same level of high availability, scalability and performance.

Deployment Selection	Description
Elastic Load Balancer	No load balancer required.
Web Server Instance Type	An EC2 general purpose m1.xlarge instance is selected (64 bit, 4 virtual CPUs, 15GB of memory, 4x420GB of ephemeral storage, available EBS storage, and high network performance).
Number of Baseline Web Servers	Two baseline web servers are required.
Number of On-Demand Web Servers	One on-demand web server is required.
Amount of Data Served	Assume 1GB per hour, primarily for standards information stored in Core Standards.
DB Server Instance Type	A Multi-AZ Large RDS database instance is selected (IOPS optimized).
DB Server Storage	Assume minimum of 100GB of database storage.
DB Server Provisioned IOPS	300 IOPS for a ratio of 3 to 1 Provisioned IOPS to storage size.

Table 12. Shared Services Deployment Unit Summary

Alternative Deployment Scenario

Smarter Balanced is aware that some vendors may choose to certify alternative delivery systems rather than deploy the open source code base. To support that approach, Smarter Balanced will be releasing a Certification Package composed of all of the requirements, specifications, sample items and test harnesses necessary to certify a system for delivery of Smarter Balanced assessments. As Smarter Balanced is presently contracting out development of the certification package, delivery dates are not yet fixed. However, we expect the first specifications to become available before the end of Calendar Year 2013. One convenient way to adapt an existing test delivery system is to

combine a vendor’s existing test delivery server with the Smarter Balanced front-end. The following diagram illustrates this approach.

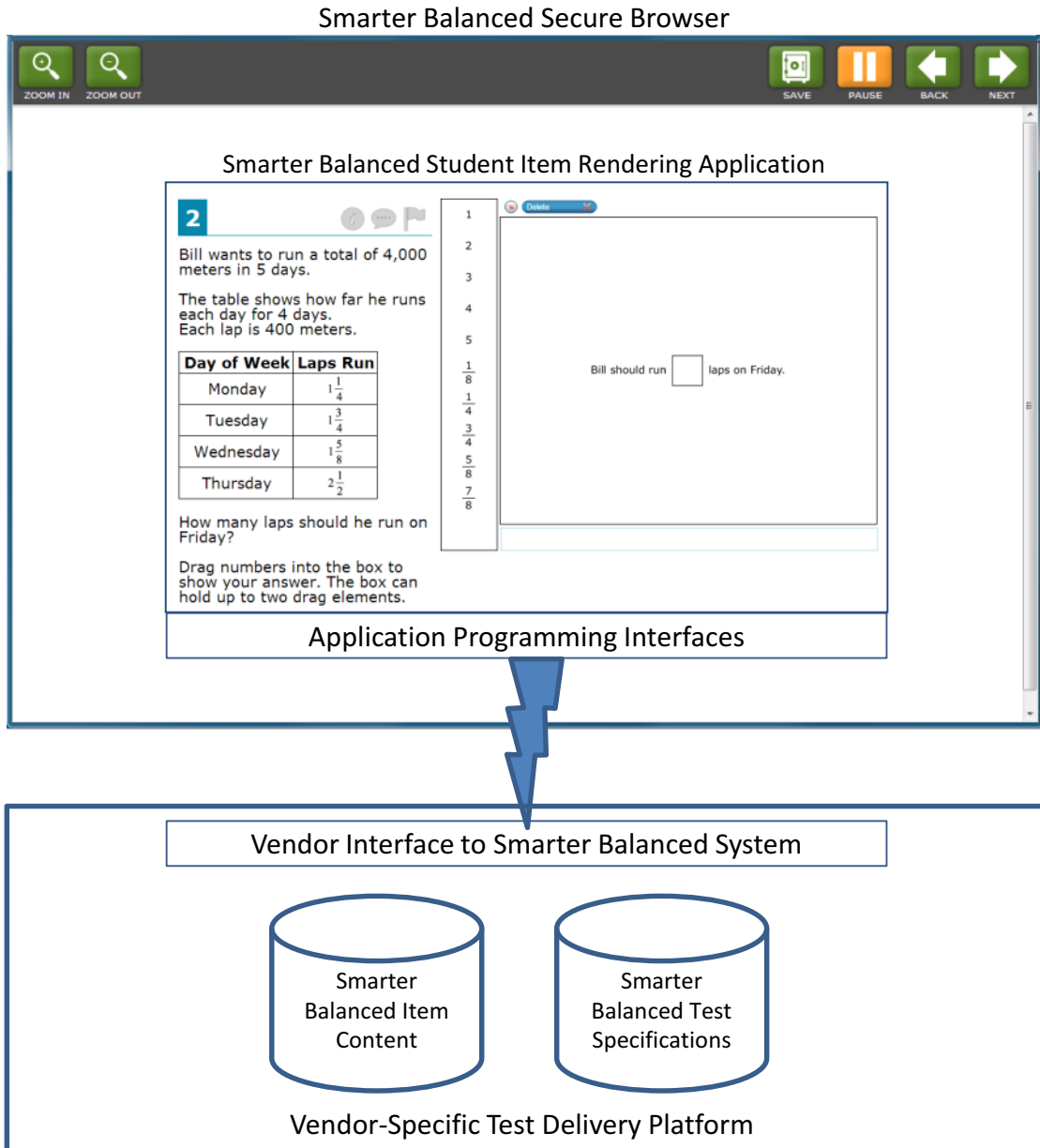


Figure 5. Alternative Approach Using Vendor-Specific Test Delivery Platform

In this scenario, test delivery vendors develop a Smarter Balanced-specific set of server-side interfaces that satisfy the application programming interface (API) requirements of Smarter Balanced Student Item Rendering Application and Secure Browser.

The Smarter Balanced Student Item Rendering Application is a Javascript client-side application that enables navigation and accurate item rendering for Smarter Balanced assessments. The Secure Browser is a special version of browser that limits student interactions outside of the assessment

and provides certain accessibility features. Together, these two applications comprise the part of the assessment platform that runs on the student computer and that interacts with the student.

Not shown in this diagram is the Smarter Balanced Proctor Application, a client-side application similar to the Student application that is used by test administrators to create and manage test sessions that students can join. It also requires certain server side interfaces that are part of the vendor-specific interface.


This approach has the distinct advantage that the test delivery platform remains the same as the vendor’s current proprietary test delivery platform. In order to take advantage of this alternative, a vendor will have to take the following steps:

1. Implement vendor interfaces to the Smarter Balanced Student and Proctor applications and the Secure Browser. This is necessary to present and render Smarter Balanced assessments that are indistinguishable to the test administrator and student. The uniformity of navigation and rendering provides a uniform assessment experience, maintaining the integrity of the Smarter Balanced assessments.
2. Consume Smarter Balanced items: Assessment item content and item assets would be made available by Smarter Balanced. Vendors will need to perform the appropriate translation between the item content and assets and provide the Student Application with correctly formatted XML. The Student Application will correctly render the items on the student’s computer and provide the necessary student-facing navigation and features that the vendor interface satisfies.
3. Consume Smarter Balanced test specifications: Assessment packages will be provided that includes blueprints, scoring rules, reporting parameters and adaptive algorithm configuration. These test packages provide sufficient information for the vendor’s proprietary assessment platform to deliver the Smarter Balanced assessments as designed.

Smarter Balanced is working on a complete set of specifications including APIs, item specifications and test specifications that vendors can use to adapt their assessment delivery platforms in this scenario. Smarter Balanced will announce the completion of the specifications and provide documentation and implementation guidelines as appropriate.

Cost Calculation Spreadsheet

A spreadsheet is provided embedded in the table below that calculates expected monthly and yearly costs. The spreadsheet accepts inputs that capture assumptions the deployment and produce total costs based on the AWS prices at the time of this writing.

Cost Model Spreadsheet	 Smarter Balanced Contract 11 Hosting F
------------------------	---

The following is an example cost summary produced by this spreadsheet when key assumptions shown above are plugged in. This is a worse-case example using California student population numbers and assuming a 7.8% concurrent student load.

TOTAL COSTS PER MONTH

Monthly Costs	Test Delivery	Test Int & Scoring	Test Reg & Adm	SSO	Mon & Alerting	Shared Services	Assmt Creation & Mgmt	Total
Load balancer	1,678.60	293.58	263.64	51.27	0.00	19.97	31.64	2,338.70
Data transfer out	1,219.20	32.00	2,944.00	176.00	0.00	20.00	160.00	4,551.20
Baseline web server	5,738.88	5,738.88	409.92	819.84	204.96	409.92	409.92	13,732.32
On-demand web server	943.46	1,048.32	74.88	0.00	0.00	93.59	74.88	2,235.13
Web server storage	0.00	0.00	0.00	156.86	139.50	0.00	0.00	296.36
Database server	27,054.72	7,583.52	541.68	0.00	0.00	0.00	241.56	35,421.48
Database server storage	63,000.00	3,076.50	223.50	0.00	0.00	85.00	850.00	67,235.00
CloudWatch Monitoring	245.00	147.00	10.50	14.00	3.50	14.00	10.50	444.50
Subtotal	99,879.86	17,919.80	4,468.12	1,217.97	347.96	642.48	1,778.50	126,254.69
AWS Support Costs								12,625.47
Ubuntu Cloud Support Costs								6,250.00
Total								145,130.16

TOTAL COSTS PER YEAR

Monthly Costs	Test Delivery	Test Int & Scoring	Test Reg & Adm	SSO & Perm	Mon & Alerting	Shared Services	Assmt Creation & Mgmt	Total
Load balancer	20,143.20	3,522.96	3,163.68	615.24	0.00	239.64	379.68	28,064.40
Data transfer out	14,630.40	384.00	35,328.00	2,112.00	0.00	240.00	1,920.00	54,614.40
Baseline web server	68,866.56	68,866.56	4,919.04	9,838.08	2,459.52	4,919.04	4,919.04	164,787.84
On-demand web server	11,321.52	12,579.84	898.56	0.00	0.00	1,123.08	898.56	26,821.56
Web server storage	0.00	0.00	0.00	1,882.32	1,674.00	0.00	0.00	3,556.32
Database server	324,656.64	91,002.24	6,500.16	0.00	0.00	0.00	2,898.72	425,057.76
Database server storage	756,000.00	36,918.00	2,682.00	0.00	0.00	1,020.00	10,200.00	806,820.00

CloudWatch Monitoring	2,940.00							5,334.00
	0	1,764.00	126.00	168.00	42.00	168.00	126.00	0
Subtotal	1,198,558.32	213,273.60	53,491.44	14,447.64	4,133.52	7,541.76	21,216.00	1,515,056.28
AWS Support Costs								151,505.64
Ubuntu Cloud Support Costs								75,000.00
Total								1,741,561.92

Table 13: Example hosting cost calculator output

PLEASE NOTE THAT AMAZON SUPPORT COSTS OF 10% OF THE SUBTOTAL AND UBUNTU YEARLY SUPPORT COSTS OF \$75,000 PER YEAR ARE FACTORED IN.