Deploying Liferay Digital Experience Platform in Amazon Web Services

Table of Contents

Introduction

Liferay Digital Experience Platform (DXP) can be deployed into a variety of infrastructures and cloud-based environments. The Liferay Engineering and Global Services teams have extensive experience deploying and managing infrastructure in cloud environments, including Amazon Web Services (AWS). Their accumulated experiences and best practices are described in this comprehensive document. It provides an initial architecture of AWS deployments and can be altered depending on specific needs for fault tolerance, scalability, and other infrastructure and non-functional requirements. The reference architecture presented in this document is similar to the base reference architecture as described in the Deployment Checklist with AWS-specific technologies applied.

The Liferay Global Services team also offers a specialized Go Live package which can help Liferay users with pre-production tuning and configuration.

Reference Architecture

The selection of an appropriate architecture is one of the first decisions in the deployment path. To select an appropriate architecture, the following must be considered:

- **Information Security**: Securing the hardware and sensitive information against malicious attacks and intrusions
- **Performance**: Supporting the desired number of total users, concurrent transactions, etc.
- **Fault Tolerance**: Maintaining uptime during unexpected failure or scheduled maintenance
- **Flexibility and Scalability**: Designing an expandable architecture to support additional features and users without significant redesign

Overview

The reference architecture depicted in Figure 1 appears complex, but it provides high levels of fault tolerance and flexibility.

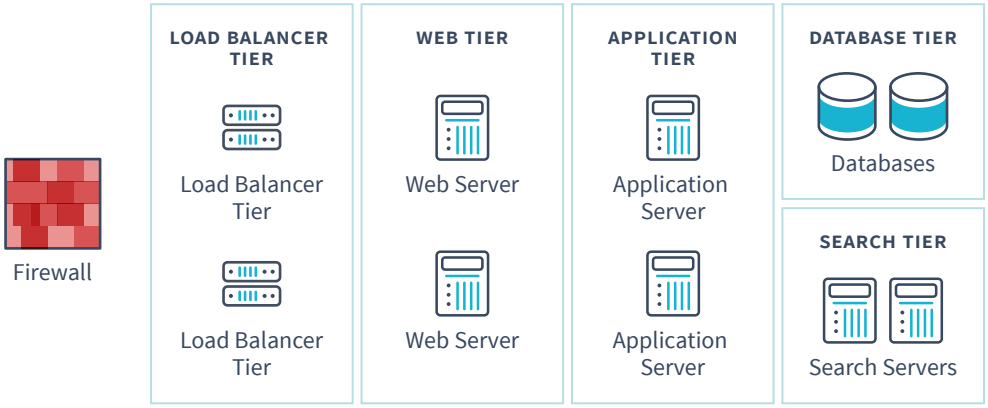

Figure 1 - Liferay DXP Reference Architecture

The architecture contains the following tiers:

Firewall	• Provides intrusion detection and prevention
Load Balancer	• Ensures smooth distribution of load between multiple web server resources (and underlying application servers)
Web Tier	• Delivers static content elements like images, rich media, CSS files, etc. • Provides integration modules to single sign on solutions like CA Netegrity, Oracle Identity, etc. • Handles custom routing needs
Application Tier	• Hosts Liferay DXP on supported application servers like Tomcat, JBoss, Wildfly and IBM WebSphere*
Database Tier	• Hosts Liferay-supported database servers like MySQL, Oracle, IBM DB2 and PostgreSQL*

*Please see Liferay DXP Compatibility Matrix for a complete list of supported application servers and database servers.

Sizing

The hardware deployed within each tier varies depending on the type of transactions. We will use Liferay Engineering's benchmarking environment as a hardware specification guide. Some of the tiers will use AWS services where explicit sizing is not available:

Firewall

- Amazon Virtual Private Cloud (VPC), subnets, security groups, and network access control lists (ACLs)

Load Balancer Tier

- 1 x AWS Elastic Load Balancer (ELB)

Web Tier

- 2 x Amazon EC2 c6g.xlarge instance
  (4 vCPU AWS Graviton2 Processor with 64-bit Arm Neoverse cores, 8 GB memory)
- Amazon CloudFront (CDN) can be utilized for some functions at this tier as well
- Also possible to share EC2 instance with application tier, or not utilize web servers at all (both options are discussed in more detail in the Web Tier section)

Application Tier

- 2 x Amazon EC2 c6g.4xlarge instance
  (16 vCPU AWS Graviton2 Processor with 64-bit Arm Neoverse cores,
  32 GB memory)
- AWS Simple Storage Service (S3) used as document repository
  (discussed in File Storage)
- Separate Elasticsearch cluster used as Liferay's search engine
  (discussed in Search)

Database Tier

- 2 x Amazon RDS db.m6g.2xlarge instance
  (8 vCPU AWS Graviton2 Processor with 64-bit Arm Neoverse cores, 32 GB memory)

Search Tier

- 3 x Amazon EC2 r5.2xlarge instance
  (8 vCPU 3.1 GHz Intel Xeon Platinum processors, 64 GB memory)

Implementation Details

Firewall

VPCs can define one or more subnets, which can be set as private or public. Making a subnet private makes all resources created in this subnet (like EC2 instances or ELBs) inaccessible from the internet. This may be used to isolate the entire infrastructure from the public internet and isolate layers such as the web tier and application tier according to security requirements.

AWS security groups and ACLs further enable users to define access restrictions on inbound and outbound ports on various resources, like EC2 instances or ELBs.

Load Balancer

AWS Elastic Load Balancer (ELB) allows traffic to be distributed in an even manner between EC2 instances. There are two versions of Elastic Load Balancing available - v1 and v2.

ELB V1

Classic Load Balancer (v1) is very basic and only utilizes single source and target IP and port to route traffic. Sticky sessions can be defined, either using a dedicated AWS-generated cookie or by reusing an existing cookie generated by application servers (like `JSESSIONID` in case of servlet containers).

Although somewhat outdated, this type of balancer can still be used with Liferay DXP. Since v2 was introduced, no major improvements were made for the v1 balancer and many new features, like integrations with other AWS services, are only implemented for the v2 balancers. For these reasons, it's recommended to use the v2 balancers for Liferay DXP.

ELB V2

Application Load Balancer (v2) offers more configurability and features out-of-the-box than the v1 balancer. It is recommended to use this type of balancer for Liferay DXP. Just as with ELB v1, sticky sessions can be defined, based on cookies generated by the balancer ("duration-based") or the application itself ("application-based"). Although both options can be used with Liferay DXP, it's recommended to use the second approach and use `JSESSIONID` as the cookie. For more details, please see the AWS documentation on this topic.

Web Tier

The web tier in AWS consists of the EC2 instances with web server installed and possibly Amazon CloudFront (CDN) as well.

Amazon CloudFront is typically responsible for delivering static content from any CDN which usually serves images, CSS, JavaScript, and other files that do not change frequently.

As for the web server implementation being used in the EC2 instances, any common web server such as Apache or Nginx can be deployed and configured to perform the desired work (compression, caching etc.) and then forward requests to the appropriate application server in the next tier. Liferay DXP should be configured accordingly by disabling the unnecessary servlet filters in Liferay (e.g. gzip compression filter).

The reference architecture utilizes separate EC2 instances for this tier, but deploying the web servers on the same EC2 instances as the application servers is also a viable option. Note that such a strategy will result in the web and application tiers sharing computing resources, which may not be ideal for all applications and loads.

You may also choose not to deploy a web tier and instead rely upon a content delivery network (CDN) to deliver your static resources, like CloudFront CDN from AWS. As with most architectural choices, there are advantages and disadvantages to each approach. Load tests will determine the best option for your solution.

Please see the Liferay DXP Compatibility Matrix for a complete list of supported operating systems which can be used in the EC2 instances.

EC2 instances should be created in several availability zones (within one region) to increase the level of resiliency against data center outage.

Application Tier

This tier consists of EC2 instances with an application server where Liferay DXP is deployed. Users can choose from a variety of Java application servers and operating systems when deploying Liferay. The AWS platform offers a wide range of Amazon Machine Images (AMIs) and OS options so users can choose based on what suits their needs. Custom AMIs can also be created in any AWS account and be the base for EC2 instances.

Please see the Liferay DXP Compatibility Matrix for a complete list of supported application servers and operating systems which can be used in the EC2 instances.

EC2 instances should be created in several availability zones (within one region) to increase the level of resiliency against data center outage. If you do choose to spread your deployment across multiple availability zones, you should co-locate the EC2 instances with your multi-availability zone database and other components of the infrastructure.

Database Tier

AWS offers a managed database solution, Relational Database Service (RDS). Users can select from common database servers and also choose the size of the RDS instances, determining how much power will be available to each database server. AWS RDS service offers a low barrier of entry for users looking to deploy highly resilient and scalable database tiers.

Liferay recommends using RDS instead of managing a set of EC2 instances with database servers deployed. The cost difference is minimal and the added value provided by a managed service outweighs the savings in almost every circumstance.

Assuming the platform is supported according to the Liferay DXP Compatibility Matrix, users are free to choose any database platform.

Search Tier

For this tier, we need to set up an Elasticsearch (ES) cluster compatible with Liferay DXP. It's recommended to use Elasticsearch 7 for Liferay DXP 7.1 and above. Elasticsearch 7 is supported out of the box in Liferay DXP 7.3+, and there is a Liferay Marketplace plugin available here for DXP 7.1 and 7.2.

Liferay and Elastic recommend at least a two-node cluster so as to provide resiliency in case of random failures. For cluster sizes greater than 2, Elasticsearch recommends setting the discovery.zen.minimum_master_nodes property to "2". For more information, please consult the node settings documentation on Elasticsearch.

Since Liferay DXP 7.3+ utilizes the HTTP connection to transfer the data to Elasticsearch (typically the port :9200 in Elasticsearch), it's no longer needed to maintain the matching JDK versions between DXP and Elasticsearch nodes. This was the case only with DXP 7.2 and earlier, which was using the TCP connector, typically on port :9300. It's recommended to use the JDK vendor and version as recommended by the official Elasticsearch documentation for the given Elasticsearch version.

As with the application tier, EC2 instances of ES should be created in several availability zones (within one region) to increase the level of resiliency against data center outage.

For details on how to install and setup ES, please check the official documents provided by its creators: Set up Elasticsearch.

Note: Elastic, the company behind Elasticsearch, offers its own service - the Elastic Cloud. It can be successfully utilized with Liferay DXP running in AWS, as a remote Elasticsearch cluster. This assumes a matching version of Elasticsearch is selected for your DXP, as described in Liferay's Search Engine Compatibility Matrix.

To minimize the networking latency, it's recommended to use AWS as the cloud provider of your Elastic Stack and also use the same AWS region as your DXP nodes run in. Since your Elasticsearch cluster will have a public endpoint, make sure to use encryption wherever possible (in transfer of the data, at rest) and enable authentication on the Elasticsearch side. This is true by default in any elastic Cloud deployment. Then just supply the HTTPS endpoint URL and credentials to your DXP Elasticsearch connector as usual for remote clusters.

AWS also offers a managed ES solution – Amazon Elasticsearch Service. However, Liferay DXP is not compatible with Amazon OpenSearch. For optimal operation, it is required that the Elasticsearch distribution is used as the search cluster for your Liferay DXP cluster.

Liferay Specific Considerations

File Storage

By default, Liferay utilizes disk storage for its document management capabilities. In traditional (non-cloud) deployments, customers tend to provision network attached storage (NAS) or storage area network (SAN) drives mounted via network file system (NFS) or similar technologies.

In AWS, Liferay DXP can leverage either the Simple Storage Service (S3) or Elastic File System (EFS) as its underlying document management storage system. S3 is a common document storage system as it is cost effective, highly durable, and provides file versioning and encryption at rest. While S3 promises high durability, one downside is the lack of a snapshot type of backups, which puts a wrinkle in the ease of Disaster Recovery. To use S3, simply configure Liferay DXP to use S3Store as an implementation of Document Library store. Please see the Liferay User Guide for instructions on how to set up S3Store.

Elastic File System (EFS) is a cloud storage system that supports concurrent access from multiple sources (e.g. EC2 instances). It is analogous to a traditional NFS but on the cloud. While it is easy to set up, there are some minor drawbacks to EFS. First, Amazon EFS is not available in all AWS regions. Please check the

AWS Regional Table to confirm its availability. Second, the storage of files in EFS is typically several times more costly (per GB) than storing the same files in S3. However, EFS is supported by the AWS Backup service (for more, see Disaster Recovery on page 10).

Another option is to use DBStore, but Liferay strongly discourages its use except for demoing purposes or deployments with zero requirements on storing documents in Liferay DXP.

Search

Liferay DXP ships with either an embedded Elasticsearch search engine (Liferay DXP <=7.2), where the Elasticsearch engine runs in the same JVM as Liferay DXP, or a sidecar (Liferay DXP 7.3+), which automatically manages a separate process in the OS (see docs). **Although these approaches are great for having out-of-the-box search in Liferay, production use of embedded or sidecar Elasticsearch won't be supported by Liferay and any issues will be out of the scope of Liferay support.** For production usage, Liferay recommends and will only support a standalone Elasticsearch search engine that runs outside of the DXPs own JVM (i.e. 1 JVM for Liferay DXP and a separate JVM for the Elasticsearch search engine) and is not managed by DXP.

Search engines benefit heavily from caching and their JVM memory profiles are substantially different from a JVM focused on serving content and web views (e.g. Liferay JVM). For these reasons, the two applications should always be kept separate in production environments.

The search engine JVM can run on the same EC2 instance as the Liferay JVM; however, this will cause the two processes to compete for the same resources. For heavy search usage, Liferay strongly advises deploying not only to a separate process, but to a separate EC2 instance as this will provide dedicated CPU capacity.

Although Apache Solr is an alternative for earlier versions of Liferay DXP, it has been deprecated for some time and eliminated altogether for Liferay DXP 7.4. We recommend using Elasticsearch and avoiding an eventual but necessary migration later.

Cloud Architecture Considerations

Security

As mentioned in the implementation section (see Firewall), the basic means of securing the Liferay deployment in AWS are VPCs, security groups, and ACLs. There are, however, additional measures which can be taken if higher security standards are required.

SSL/TLS

When setting up ELB, by default it listens (and forwards) only the incoming traffic on port 80, which is an unencrypted HTTP connection. It is strongly recommended to also add a listener on port 443, which is an encrypted HTTPS connection. This will make sure that data transferred between the user's browser and ELB will be encrypted and cannot be intercepted. A valid SSL/TLS certificate needs to be provided to ELB, either by choosing one of the user's existing certificates, uploading a new one into IAM, or provisioning one from the AWS Certificate Manager. SSL/TLS certificate on ELB is used to decrypt the incoming traffic and then route it to one of the defined instances. ELB acts as an SSL connection terminator.

After the request is processed by the balancer it needs to be forwarded to EC2 instances. This can be done in unencrypted form, since the traffic is now flowing through VPC's internal network. The connection can be encrypted again by setting up another SSL connection between ELB and EC2 instances (e.g. web server or Tomcat with Liferay DXP deployed).

As for the SSL/TLS certificates, Amazon became a certificate authority (CA) in early 2016 and added a service called AWS Certificate Manager. This service allows all users of AWS to ask for SSL certificates without any additional charges. This certificate can then be deployed on ELB, providing HTTPS capability. Optionally, for users not comfortable with Amazon managing their SSL certificate and any private information it includes, users can opt for getting an SSL certificate using a traditional approach (i.e. generating the SSL certificate and having an external CA sign it upon verifying the user's and server's credentials).

DATA ENCRYPTION

In many of its services, AWS offers encryption of the data at rest as well. This includes, but is not limited to:

- Encrypted EBS volumes, mounted into EC2 instances
- RDS encryption for DB instances
- Encryption of objects in S3 buckets
- Encryption of objects in EFS

All these services use AWS Key Management Service to store the encryption keys. AWS claims there is only a minimal overhead for the encryption, that it does not limit the use of the encrypted resource and that it will be available in the same way as the unencrypted option.

Please see the latest AWS documentation to see if it offers additional encryption options and follow the recommended steps for implementation.

Autoscaling

Failure scenarios should be planned (e.g. application not responding, loss of server, loss of data center, etc.) and factored into the high availability and fault-tolerance plan for the application.

Autoscaling is a common strategy used in cloud environments to improve fault tolerance and provide dynamic resource allocation. AWS offers Auto Scaling Groups as part of its EC2 service. It allows administrators to manage sets of EC2 instances (pools) that will automatically adjust its size. New instances can be added or existing ones removed, based on various factors including load or failures.

We will not discuss how to set up auto-scaling groups in AWS as official AWS documentation offers excellent guidelines with step-by-step instructions. The AWS console also contains wizards to guide users through all necessary steps when defining autoscaling.

Liferay is compatible with autoscaling architectures provided that new instances are added to replace others that have shut down, for example due to unexpected failures. Liferay customers deploying on AWS must ensure that the number of running instances does not exceed the number of purchased subscriptions.

Disaster Recovery

Disaster recovery is the process of restoring a system to its normal running condition from a failure. At a high level, restoring a typical DXP system may include rebuilding the infrastructure, reinstalling the applications and restoring data. Ideally, the recovery process should be automated and AWS offers many features that can assist in implementing a proper disaster recovery operation. In this section, we'll provide a high-level overview of the features that relate to running DXP.

AUTOMATED SETUP

A major part of recovering a DXP system from a disaster is the recreation of the entire DXP system, including the underlying system infrastructure and all its applications. This can be done by carefully documenting all steps in the set up of the infrastructure and applications. However, Liferay recommends fully

automating the construction of the entire DXP system, as well as the on-going deployment of application development. With little to no manual intervention, disaster recovery is then simply executing the automated infrastructure reconstruction along with deploying the latest application. This approach in building infrastructure automatically is known as "infrastructure as code" or "configuration as code."

There are tools that can help with automating the building of infrastructure and deploying applications, such as:

- CloudFormation (AWS service)
- Terraform
- Sceptre

It is important to perform all successive updates (i.e. done after initial installation) using the automated tool, to make sure the stack is recovered exactly to the configuration from before the failure.

BACKUP SCHEDULE AND REPLICATION

Backing up data (or content) properly is an important piece in planning for a successful data recovery procedure. DXP has three sets of data which need to be backed up and recovered after a failure:

- Database
- Document library file store
- Elasticsearch index

AWS Backup is a service that automates scheduled backups, defines retention policies and removes the need of custom backup scripts. It supports the backup of RDS (database), EFS (file storage if chosen), and EBS (search index). As a best practice, plan on synchronizing the backups to all three sets of data, so that they remain as consistent as possible.

S3 Backed Document Library

If S3 is chosen (and not EFS) as the storage implementation for the DXP document library, the following setup can be used as a backup strategy for the document library file store.

- Enable versioning on the bucket
    - This ensures that any deleted files will only be marked for deletion and will be kept until a dedicated file-version API is used to permanently delete that particular version.

- Liferay DXP uses the regular S3 API to delete the files (never file versions), so all files deleted from Liferay DXP can still be looked up and restored in S3.

- Enable cross-region replication into secondary S3 bucket
  - For example, all files from the main bucket in us-east-1 region into a second one in us-east-2 region.

DATA RECOVERY

For database, assuming RDS was used, backup snapshots can be restored directly into the same RDS instance. The endpoint then stays exactly the same in the eyes of DXP JDBC configuration.

For the document library (file storage), if EFS was chosen, restoring from its backups can be done directly in the AWS console. If S3 is used to store files for the document library, the restoration is a bit more complicated. Unfortunately, AWS does not provide a way to restore the whole bucket into a point-in-time (with versioning enabled on the bucket), only individual objects can be restored. The only way to revert all the files in a bucket to a previous state is to implement a custom script to iterate over all the files and revert them individually.

Lastly, for the search index, if the EBS storage for the Elasticsearch nodes are backed up, then restoring the index from the backup can be done in the AWS console, very similar to restoring the RDS backup and EFS backup. However, if the search index is not backed up, a full reindex can be performed once the entire DXP system is restored.

PUTTING IT ALTOGETHER

With both the scheduled backups and data recovery procedures in place, it is important to perform a disaster recovery test. The end goal of this test is to reconstruct a fully functional DXP system from the various backups, using the data recovery procedures. Of course, this should be done in an environment closely matching the production environment, but not the actual production environment (unless downtime is acceptable, or if the project has not been debuted). Additionally, commit to testing disaster recovery regularly.

Networking (Liferay Clustering)

When Liferay DXP is deployed in AWS, it will run inside Amazon's VPC network. The virtualized networking infrastructure is very similar to physical networks. However, it is important to note that multicast is not available in Amazon VPC.

Liferay implements clustering within its Cluster Link feature. By default, Cluster Link uses multicast for discovery and communication between cluster members. In AWS, Cluster Link can be configured to use either S3 or JDBC for member discovery and TCP for communication between cluster members.

Cloud vs. Bare Metal Performance

Special attention should be paid to the vocabulary used when describing AWS performance, especially when instance-sizing choice is required from the user for services like EC2 or RDS.

With bare metal machines, a CPU refers to the physical chip which typically contains multiple cores. Each core may contain one or more hardware threads and modern processors typically have two hardware threads per core. AWS uses the term vCPU (virtual CPU) to mean one physical hardware thread.

For example, c4.4xlarge instances in EC2 are providing 16 vCPUs from Intel® Xeon® E5-2666 v3 chips. This means the instance can utilize 16 hardware threads. Based on official Intel documentation, each physical E5-2666 v3 chip contains 10 hyper-threaded cores or 20 hardware threads. Therefore, c4.4xlarge instances provide roughly the equivalent CPU performance of 80% of a physical E5-2666 v3 CPU.

Virtualization will also have a certain degree of overhead when compared to bare metal. For instance, while the c4.4xlarge instances theoretically provide 80% of the threads from a physical CPU, in reality, they offer less than 80% of bare metal performance. This should be factored into the calculations when performing capacity planning.

Summary

The preceding sections outlined the steps and considerations to design a fault-tolerant Liferay DXP deployment for the Amazon Web Services cloud platform. The described architecture provides a solid foundation for future growth.

Disclaimer

Liferay can only give you an initial tuning recommendation based on benchmarks that have been performed on the core Liferay DXP product. It is up to you as system architects and business analysts to come up with the utilization scenarios that your system will need to service the required amount of users.

It is your responsibility to run the appropriate load tests on your system before production deployment, so that you can identify significant bottlenecks due to custom applications/portlets or other unforeseen system and network issues, and implement appropriate configurations.

Please use this document and the Liferay DXP Deployment Checklist as guides in sizing your system and procuring your hardware.

AWS service features and compatibility may change over time. Please test your system, as well as the disaster recovery plan regularly to ensure it is operational.

Moving Forward

Liferay Experience Cloud

If your team would rather focus on business critical needs instead of infrastructure maintenance, Liferay Experience Cloud is a secure and scalable platform tailored for Liferay DXP. For more information, visit liferay.com/products/liferay-experience-cloud or contact sales@liferay.com.

Liferay Global Services

Learn how Liferay's Global Services team can support your Liferay DXP project with a Go Live consultation. Contact sales@liferay.com for more information.

References

1. Liferay DXP Compatibility Matrix
   liferay.com/compatibility-matrix
2. Liferay DXP Deployment Checklist
   liferay.com/resources/product-info/Liferay+DXP+7.4+Deployment+Checklist
3. Set up Elasticsearch
   elastic.co/guide/en/elasticsearch/reference/7.0/setup.html
4. Elastic File System Supported Regions
   aws.amazon.com/about-aws/global-infrastructure/regional-product-services
5. CloudFormation
   aws.amazon.com/cloudformation
6. Terraform
   terraform.io
7. Sceptre
   github.com/cloudreach/sceptre
8. AWS Backup
   aws.amazon.com/backup
9. AWS Documentation
   docs.aws.amazon.com

Liferay makes software that helps companies create digital experiences on web, mobile and connected devices. Our platform is open source, which makes it more reliable, innovative and secure. We try to leave a positive mark on the world through business and technology. Hundreds of organizations in financial services, healthcare, government, insurance, retail, manufacturing and multiple other industries use Liferay. Visit us at liferay.com.

220425