BankLiteX_AWS_Solution_Deployment_Guideline

Last edited by **czerniga-gft** 8 months ago

BankLiteX AWS Solution Deployment Guideline

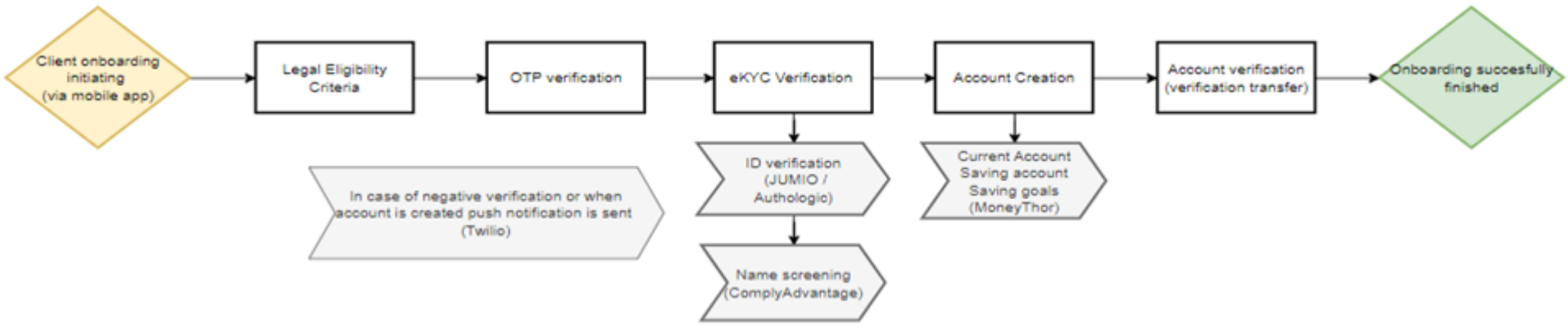
BankLiteX is officially certified by AWS Solution at 2023-06 after passing https://aws.amazon.com/partners/foundational-technical-review/ which is valid till 2025-06 and required for Advanced Partner status for GFT.

Introduction

Usecase for the software

The scope of the solution is

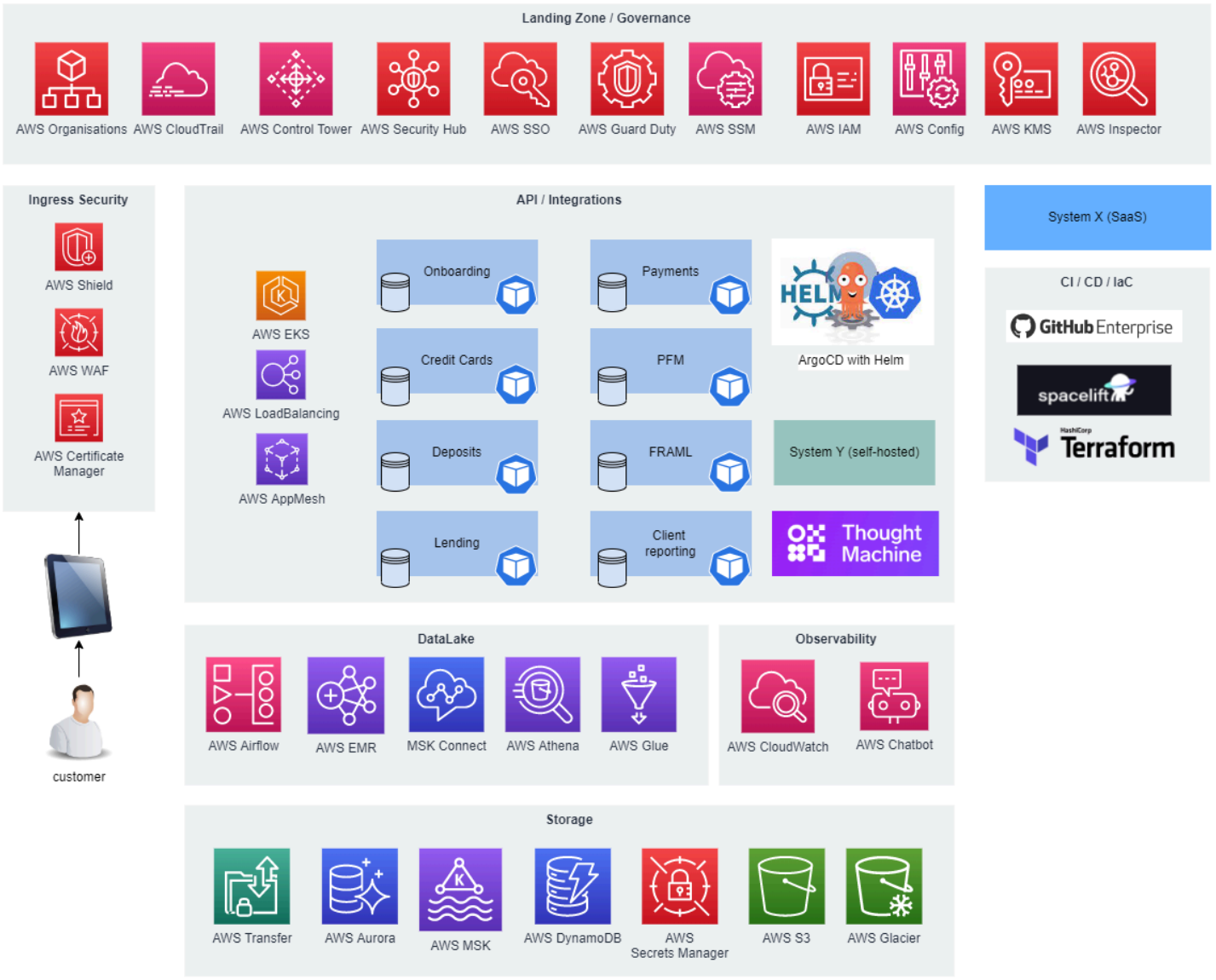
- Deliver AWS platform with fully functional end-to-end solution for further development for banks operating with ThoughtMachine Vault and potentially other core banking systems
- Example mobile android application
- Data lake solution
- Deposits functionalities
    - Savings account with savings goals
    - current accounts
    - periodical pdf statements
    - Transactions lists and insights
- Microservice template
    - a bootstrap repository to kickstart microservice platform development, following the best practices and microservice patterns
    - Written in Kotlin, Spring Boot, Gradle and using technologies like Docker, Helm, OpenApi Spec (Swagger), Kafka
    - Contains all necessary elements for containerization, Kubernetes deployment, CI/CD pipelines based on Github Actions, REST API development, asynchronous communication using Kafka, database connectivity to AWS Aurora, unit testing and integration testing
- Microservice libraries
    - Set of libraries that provides functionality common to various microservices across the platform, all libraries are integrated in microservice template
    - Handles common nonfunctional requirements including exception handling, logging, messaging, observability, data storage and replication (using transactional outbox pattern), contract first REST API development, Kafka connectivity
    - Out of the box connectivity to AWS Aurora and DynamoDB using AWS IAM
    - Support integration testing providing a set of starters for common components used across the project
- Full onboarding user journeys covering identity check, eKYC and creation of Customer entries in 3rd parties system



Additionally, it contains 3rd party service integrations

Jumio	e-KYC service used for Identity verification of clients
Twillio	Used for sms / push notifications from back-end events to mobile clients
PingOne	PingOne is a cloud-based identity service that gives users one-click access
Comply Advantage	ComplyAdvantage provides AI-driven financial crime risk data and detection technology
OneSignal	To have a single provider that routes push notifications to all target platforms
MoneyThor	The Moneythor solution offers Personal Financial Management (PFM) features

Typical customer deployment



Architecture summary

General principles

Architecture design is focused on the following principles:

- use AWS-managed services whenever it allows to save costs and shift to a pay-per-use vs pay per subscription model
- use cloud agnostic patterns whenever it is globally recognized standards (i.e. Kubernetes over other container models)
- use services promoting delivering AWS cloud-native security using AWS IAM security model
- prepare for using integrations with external SaaS providers though allow to also support self-hosting models if required

Landing Zone and governance

Architecture aims to use AWS Control Tower to use the best native tool for creating landing zones and deliver fundamental guardrails for aws account structure on top of AWS Organisations. From the beginning, AWS CloudTrail is delivering trail services for auditing while the general security posture is managed using AWS Security Hub using aggregation from the whole account structure and enabling broad range of tooling and services like GuardDuty, Config, Inspector. For human access, AWS SSO is enabled along with typical integration with external Identity Provider.

Ingress security

Bank customer typically operates from Public Internet. To expose services in a controlled and secure manner multiple layer of protection is applied. This starts from using secure DNS services through edge protection which is using DDoS protection along with AWS Web Application Firewall. Additional expansion capabilities exist for additional Network Firewall solutions depending on the organisation's security strategy with the potential use of AWS Network Firewall or 3rd party virtual firewall protections. Full support to end-to-end encryption in transit starts with using AWS Certification Manager with full control over transit protocols and lowering cost of managing certification rotation.

API and integrations

integration and application layer is using AWS EKS which is managed kubernetes service. We recommend using service mesh functionality in API and integration layer which might be delivered using AWS Service Mesh for functionalities likes mtls authentication in cluster along with encryption in transit and retries capabilities and network policies. Native load balancing services are managing traffic and integration between EKS and the rest of services.

Storage

We recommend using aws managed services hence for storage we rely on managed services to provide RDBMS and messaging (kafka) functionalities as long as S3 for object storage. Banks typically operate for transfer with SFTP protocol for which we recommend to use AWS Transfer family which is SFTP managed service. For special usecases DynamoDB is offered as highly scalable key value store.

DataLake

Data lake architecture is focused on lowering the cost of modern analytical and reporting capabilities required for virtual banking and pay per use vs pay per subscription model. For this purpose object storage is used with real time processing using streaming from operational database area. For transformations and aggregations EMR with Athena is used driven by scheduling capabilities delivered by Athena operating using AWS Glue schema registry. This architecture is highly scalable and prepared even for largest organisations and highest volumes of data processed.

Observability

Default observability stack relies on providing services in areas like:

- logs browsing and search from both kubernetes pods and infrastructure
- metrics provided from multiple sources (custom applications, prometheus, infrastructure)
- tracing support
- alerting support
- integration with modern notifications and optional interactive channels including Slack (through AWS ChatBot support) and or other channels like email, PagerDuty or ServiceNow

CI / CD / IaC

Continuous integrations and delivery might be constructed using many frameworks. We assume the most mature standards here are used like github enterprise along with github actions and Terraform potentially using some managed control planes like spacelift for managing IaC. We support the reuse of the libraries along with rich RBAC for infrastructure delivery pipeline. For docker images delivery we aim to operate using static controls and shift left approach for automated testing and static analysis. Vulnerabilities protections are enabled by integration with AWS Inspector scans and reporting.

The expected amount of time to complete deployment

The expected amount of time for whole solution delivery is described below. Please note that this timeline is in weeks and focused on the whole solution.

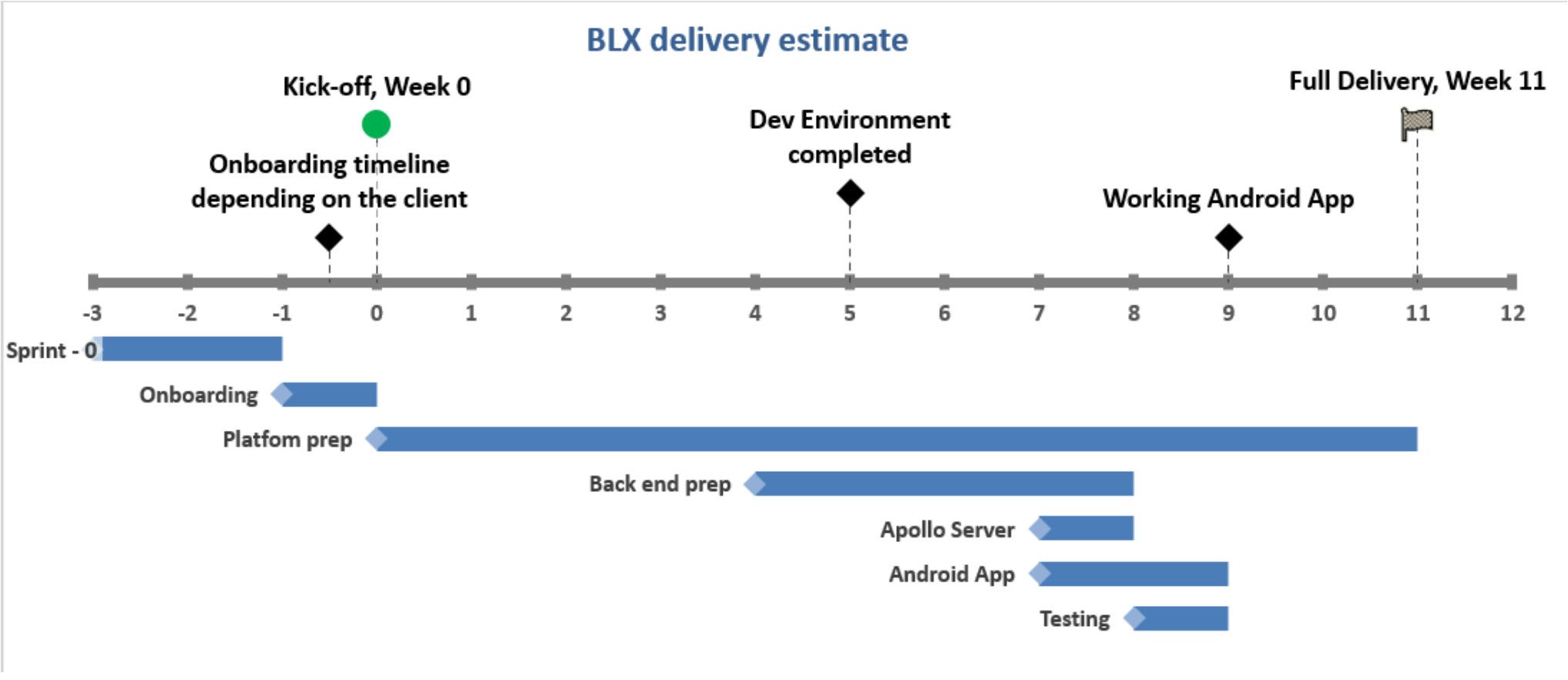
Example platform squad timelines

			W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W...n
Core team	Infrastructure Delivery	Landing zone	x	x	x								
50% of Cloud Solution Architect		Control Tower	x	x	x								
1 DevOps / Cloud Team Lead		Base guardrails	x	x	x								
2 DevOps / Cloud Engineers		Base account structure	x	x	x								
		Networking account			x	x							
		First non-prod environment (DEV)							x	x			
		Each next env									x	x	
		Integrations (ingress, egress)							x	x	x	x	x
	Application operations												
		CI pipelines (microservices)					x	x					
		CD pipelines (microservices)					x	x	x				
		Shared Service account					x	x					
		CI / CD pipeline for IaaC				x	x						
		CI / CD pipelines for supporting components (i.e. kafka, db)					x	x					
		gitops for github / teams managemnet			x	x	x	x					
		Core Observability (monitoring & alerting)							x	x			
		Core Compliance reporting				x	x	x					
Augmentation / Operational management team minimum 2 of engineers (recommended 5)	Cloud operating model												
		Disaster Recovery & BCP testing								x	x	x	x
		Chaos Engineering								x	x	x	x
		Cost management				x	x						
		Incident Management							x	x	x	x	x
	Operational efficiency												
		Continous Compliance				x	x	x	x	x	x	x	x
		Cost optimisation								x	x	x	x
		Capacity optimistations								x	x	x	x
		Perfomance / load testing									x	x	x
		Extended observability					x	x	x	x	x	x	x

Example source file: <a href="attachments/331498394/337806045.xlsx"

data-linked-resource-id="337806045" data-linked-resource-version="1" data-linked-resource-type="attachment" data-linked-resource-default-alias="blx2-platform-estimate-timelines.xlsx" data-nice-type="Excel Spreadsheet" data-linked-resource-content-type="application/vnd.openxmlformats-officedocument.spreadsheetml.sheet" data-linked-resource-container-id="331498394" data-linked-resource-container-version="37">blx2-platform-estimate-timelines.xlsx

Example full project timelines



Some additional information: [Implementation Estimates]

Regions supported

The application was tested in EU-1 (Frankfurt) region though should be supported in all the regions containing:

- EKS
- MSK
- Aurora (Postgresql)
- ALB with WAF
- Code Artifact and ECR
- (optionally) Airflow with EMR and Glue

Prerequisites and Requirements

All the prerequisites are listed in [HOWTO setup Spacelift and provision Apps & TM infra].

Skills required include:

- terraform skill
- optionally spacelift
- bash / shell in linux
- kubernetes / helm
- aws services

Architecture Diagrams

All diagrams are contained within the corresponding architectures and using AWS Architecture icons.

Security

Using AWS root account and IAM users

The project does not require using a root account and was fully tested using only IAM roles using SSO.

Least privilege guidance

The project follows the least privilege:

- using IRSA for EKS roles with granular roles
- not using instance nodes permissions
- introducing minimum roles required for a given role

Public Resources

The following resources are aimed to be public:

- ALB with WAF load balancers for public-hosted zone

IAM roles

Name	Context	Purpose
chatbot-sandbox-apps-role	SA for chatbot	apps
sandbox-(accountid)-mwaa-role	SA for MWAA	apps
sandbox-apps-apollo-eks-irsa-role	IRSA EKS for apollo	apps
sandbox-apps-appmesh-controller-eks-irsa-role	IRSA EKS for appmesh controller	apps
sandbox-apps-aurora-automations-apps-eks-irsa-role	IRSA EKS for aurora automations	apps
sandbox-apps-aws-load-balancer-controller-eks-irsa-role	IRSA EKS for aws loadbalancer	apps
sandbox-apps-cert-manager-eks-irsa-role	IRSA EKS for cert manager	apps
sandbox-apps-cloudwatch-agent-eks-irsa-role	IRSA EKS for cloudwatch agent	apps
sandbox-apps-cluster-autoscaler-eks-irsa-role	IRSA EKS for cluster autoscaler	apps
sandbox-apps-complyadvantage-mock-eks-irsa-role	IRSA EKS for complyadvantage mock	apps
sandbox-apps-crr-calc-service-eks-irsa-role	IRSA EKS for crr cacl service	apps
sandbox-apps-customer-cbs-service-eks-irsa-role	IRSA EKS for customer cbs service	apps
sandbox-apps-customer-iam-gateway-eks-irsa-role	IRSA EKS for customer iam gateway	apps
sandbox-apps-customer-iam-service-eks-irsa-role	IRSA EKS for customer iam service	apps
sandbox-apps-customer-identity-service-eks-irsa-role	IRSA EKS for customer identity service	apps
sandbox-apps-customer-phone-verification-service-eks-irsa-role	IRSA EKS for customer phone verification service	apps
sandbox-apps-customer-service-eks-irsa-role	IRSA EKS for customer service	apps
sandbox-apps-customer-transaction-service-eks-irsa-role	IRSA EKS for customer transaction service	apps
sandbox-apps-dcr-service-eks-irsa-role	IRSA EKS for dcr service	apps
sandbox-apps-deposit-account-service-eks-irsa-role	IRSA EKS for deposit account service	apps
sandbox-apps-deposit-balance-service-eks-irsa-role	IRSA EKS for deposit balance service	apps
sandbox-apps-deposit-portfolio-service-eks-irsa-role	IRSA EKS for deposit portfolio service	apps
sandbox-apps-deposit-transfer-service-eks-irsa-role	IRSA EKS for deposit transfer service	apps
sandbox-apps-eks-common-worker-role	Role for common eks worker node	apps
sandbox-apps-external-dns-private-eks-irsa-role	IRSA EKS for external-dns private service	apps
sandbox-apps-external-dns-public-eks-irsa-role	IRSA EKS for external-dns public service	apps
sandbox-apps-external-secrets-eks-irsa-role	IRSA EKS for external secrets service	apps
sandbox-apps-fluent-bit-eks-irsa-role	IRSA EKS for fluentbit service	apps
sandbox-apps-idfc-mongo-api-eks-irsa-role	IRSA EKS for mongo	apps
sandbox-apps-jumphost-admin-role	SA for jumphost admin	apps
sandbox-apps-kafka-automations-apps-eks-irsa-role	IRSA EKS kafka automations	apps
sandbox-apps-kafka-ui-eks-irsa-role	IRSA EKS kafka ui	apps
sandbox-apps-microservice-template-eks-irsa-role	IRSA EKS microservice-template	apps
sandbox-apps-onboarding-service-eks-irsa-role	IRSA EKS onboarding service	apps
sandbox-apps-private-worker-role	SA private worker	apps

sandbox-apps-prometheus-blackbox-exporter-eks-irsa-role	IRSA EKS prometheus blackbox exporter	apps
sandbox-apps-statement-service-eks-irsa-role	IRSA EKS statement service	apps
sandbox-apps-temporal-eks-irsa-role	IRSA EKS temporal	apps
sandbox-apps-temporal-test-eks-irsa-role	IRSA EKS temporal test	apps
sandbox-apps-temporal-ui-eks-irsa-role	IRSA EKS temporal ui	apps
sandbox-apps-temporal-ui-test-eks-irsa-role	IRSA EKS temporal ui test	apps
sandbox-apps-transaction-limit-service-eks-irsa-role	IRSA EKS transaction limit service	apps
sandbox-apps-transaction-service-eks-irsa-role	IRSA EKS transaction service	apps
sandbox-apps-twilio-otp-gateway-eks-irsa-role	IRSA EKS twilio otp gateway	apps
sandbox-apps-workflow-service-eks-irsa-role	IRSA EKS workflow service	apps
sandbox-apps-xray-daemon-eks-irsa-role	IRSA EKS xray daemon	apps
sandbox-data-cert-manager-eks-irsa-role	IRSA KES cert manager (data eks)	apps
sandbox-data-cloudwatch-agent-eks-irsa-role	IRSA EKS cloudwatch agent (data eks)	apps
sandbox-data-cluster-autoscaler-eks-irsa-role	IRSA EKS cluster autoscaler (data eks)	apps
sandbox-data-eks-common-worker-role	SA common worker node data eks	apps
sandbox-data-emr-on-eks-job-role	SA emr on eks	apps
sandbox-data-external-secrets-eks-irsa-role	IRSA EKS dat external secrets	apps
sandbox-data-fluent-bit-eks-irsa-role	IRSA EKS data fluent bit	apps
sandbox-data-jumphost-admin-role	IRSA EKS data jumphost admin	apps
sandbox-data-prometheus-blackbox-exporter-eks-irsa-role	IRSA EKS data prometheus blackbox exporter	apps
sandbox-data-xray-daemon-eks-irsa-role	IRSA EKS data xray daemon	apps
system-eks-sandbox-apps-control-plane-role	SA control place apps	apps
system-eks-sandbox-data-control-plane-role	SA control place data	apps
chatbot-shared-services-ci-role	SA chatbot	shared
shared-service-ci-private-worker-role	SA private worker	shared
shared-services-ci-cert-manager-eks-irsa-role	IRSA EKS cert-manager	shared
shared-services-ci-cloudwatch-agent-eks-irsa-role	IRSA EKS cloudwatch agent	shared
shared-services-ci-cluster-autoscaler-eks-irsa-role	IRSA EKS cluster autoscaler	shared
shared-services-ci-eks-common-worker-role	SA worker node	shared
shared-services-ci-external-secrets-eks-irsa-role	IRSA EKS external secrets	shared
shared-services-ci-fluent-bit-eks-irsa-role	IRSA EKS fluentbit	shared
shared-services-ci-github-actions-runners-eks-irsa-role	IRSA EKS github actions runner	shared
shared-services-ci-jumphost-admin-role	SA jumphost admin	shared
shared-services-ci-prometheus-blackbox-exporter-eks-irsa-role	IRSA EKS prometheus blackbox exporter	shared
shared-services-ci-xray-daemon-eks-irsa-role	IRSA EKS xray daemon	shared
system-eks-shared-services-ci-control-plane-role	SA control plane	shared

IAM policies

TODO:

Name	Context	Purpose
shared-services-ci-cloudwatch-notifications-policy	cloudwatch notifications policy	shared
shared-services-ci-deny-log-access-policy	deny access to logs (for chatbot)	shared
shared-services-ci-jumphost-admin-additional-policy	jumpst addition policy	shared
shared-services-ci-jumphost-admin-eks-describe-cluster-policy	jumphost describe cluster policy	shared
shared-services-ci-jumphost-admin-kms-s3-access-policy	jumphost access s3 kms policy	shared
sandbox-apps-aurora-secret-access-policy	access to aurora secret policy	apps
sandbox-apps-cloudwatch-notifications-policy	cloudwatch notifications policy	apps
sandbox-apps-deny-log-access-policy	deny logs access policy	apps
sandbox-apps-glue-access-policy	glue access policy	apps
sandbox-apps-jumphost-admin-additional-policy	jumpstho additional access policy	apps
sandbox-apps-jumphost-admin-eks-describe-cluster-policy	jumphost eks describe cluster policy	apps
sandbox-apps-jumphost-admin-kms-s3-access-policy	jumphost ksm s3 access policy	apps
sandbox-apps-jumphost-msk-connection-policy	jumphost msk connection policy	apps
sandbox-apps-kafka-ui-policy	kafka ui policy	apps
sandbox-apps-pingone-token-policy	pingone token policy	apps
sandbox-apps-statement-service-policy	statement service policy	apps
sandbox-data-jumphost-admin-additional-policy	jumphost admin additional policy	apps
sandbox-data-jumphost-admin-eks-describe-cluster-policy	jumphost admin eks describe policy	apps
sandbox-data-jumphost-admin-kms-s3-access-policy	jumphost admin msk s3 acces plicy	apps

CMK KMS keys

Name	Context	Purpose
shared-service-default-ebs	shared services	EBS default
shared-service-flowlog-s3	shared services	Flow logs S3
shared-service-spacelift-cloudwatch	shared services	Cloduwatch
shared-service-spacelift-ecr	shared services	ECR
shared-service-spacelift-ssm	shared services	SSM for spacelift
shared-service-tf-state-dynamodb	shared services	DynamoDB state for spacelift / terraform
shared-service-tf-state-s3	shared services	S3 state for spacelift / terraform
shared-services-ci-cloudwatch	shared services	CloudWatch CI
shared-services-ci-codeartifact	shared services	CodeArficat
shared-services-ci-eks	shared services	Code Artifact
sandbox-apps-cloudwatch	apps	Cloudwatch

sandbox-data-secrets-manager	apps	Secrets Manager for data apps
sandbox-observability-sns	apps	sns for apps
sandbox-data-eks	apps	sandbox eks data
sandbox-apps-eks	apps	sandbox eks apps
sandbox-apps-ssm	apps	sandbox ssm apps
sandbox-data-sns	apps	sandbox data sns
sandbox-apps-secrets-manager	apps	sandbox apps secrets manager
sandbox-apps-platform-secrets	apps	sandbox apps paltform secrets
sandbox-default-ebs	apps	sandbox apps default ebs

Maintaining stored secrets

HOWTO configure AWS Secrets for microservices

Customer-sensitive data storage

Please note that blx itself is designed to be adopted within customer infrastructure and the following places might be (but do not have to be) a place where customer-sensitive data might be stored:

- Aurora Database
- MSK kafka storage
- S3
- CloudWatch application logs
- DynamoDB

Data encryption for KMS

Customer-managed CMK with KMS generated key material will be used as general approach for CMK keys with auto-rotation enabled.

As best tradeoff between cost of build / maintain key per account-service (i.e. dev-apps-s3-kms) will be used.

More details in Encryption at rest with KMS

Network configuration

network configuration is driven by terraform scripts in skeletons.

Shared Services

Configuration in file: tf-shared-services-networking / skeleton.tf. Example values

```
module "main" {
  source  = "spacelift.io/gft-blx/skeleton-networking/aws"
  version = "1.1.9"

  environment = var.environment
  region      = var.region
  component   = var.component

  cidr_block        = "172.21.8.0/21"
  public_subnets    = ["172.21.8.0/26", "172.21.8.64/26", "172.21.8.128/26"]
  protected_subnets = ["172.21.8.192/26", "172.21.9.0/26", "172.21.9.64/26"]
  tgw_subnets       = ["172.21.9.128/28", "172.21.9.144/28", "172.21.9.160/28"]
  private_subnets   = ["172.21.10.0/23", "172.21.12.0/23", "172.21.14.0/23"]

  kms_s3_arn         = module.kms.kms_s3_arn
  kms_ebs_arn        = module.kms.kms_ebs_arn
  kms_cloudwatch_arn = module.kms.kms_cloudwatch_arn

  eip_allocation_ids   = values(aws_eip.this)[*].allocation_id
  spacelift_ec2_ami_id = var.spacelift_ec2_ami_id
```

```
  L4_tags = var.L4_tags
}
```

Application layer

Configuration in file: tf-sandbox-apps-networking/ skeleton.tf. Example values

```
module "main" {
  source  = "spacelift.io/gft-blx/skeleton-networking/aws"
  version = "1.2.1"

  environment = var.environment
  region      = var.region
  component   = var.component
  domain      = var.domain

  cidr_block        = "172.21.32.0/21"
  public_subnets    = ["172.21.32.0/26", "172.21.32.64/26", "172.21.32.128/26"]
  protected_subnets = ["172.21.32.192/26", "172.21.33.0/26", "172.21.33.64/26"]
  tgw_subnets       = ["172.21.33.128/28", "172.21.33.144/28", "172.21.33.160/28"]
  private_subnets   = ["172.21.34.0/23", "172.21.36.0/23", "172.21.38.0/23"]

  kms_s3_arn         = module.kms.kms_s3_arn
  kms_ebs_arn        = module.kms.kms_ebs_arn
  kms_cloudwatch_arn = module.kms.kms_cloudwatch_arn

  eip_allocation_ids          = values(aws_eip.this)[*].allocation_id
  spacelift_ec2_ami_id        = var.spacelift_ec2_ami_id
  spacelift_ec2_instance_type = "t3.medium"

  L4_tags = var.L4_tags
}
```

Instance Metadata Service Version 1 (IMDSv1)

Status of Instance Metadata Service

EC2 area	IMDSv2 status
Jumphost	IMDSv2 required
EKS worker nodes	IMDSv2 required
EKS system nodes	IMDSv2 required
space lift worker nodes	IMDSv2 required
GitHub self-hosted runners	IMDSv2 required

Costs

Billable services

The solution is to deploy below services

EC2	yes
EKS	yes
MSK	yes
RDS Aurora	yes
CloudWatch logs	no
CLoudWatch Container Insights	no

CloudWatch metrics	yes (though part of metrics generated i.e. for application might be optional)
Cloudwatch alerts	yes (though part of alerts maintained as part of the application might be optional)
ELB	yes
IAM roles, policies	yes
WAF	yes
KMS	yes
S3	yes
Route 53	yes
SNS	yes
CodeArtifact	yes
VPC	yes
Apache Airflow	no
EMR	no
Glue	no
ECR	yes
Secrets manager	yes

Cost model and licensing cost

The licensing model is incorporated within the master service agreement contract and individually negotiable as part of the deployment of the virtual bank.

Sizing

Information about sizing: BankLiteX aws infra sizing recommendations

Deployment Assets

Deployment instruction

All deployment instructions are provided within HOWTOs

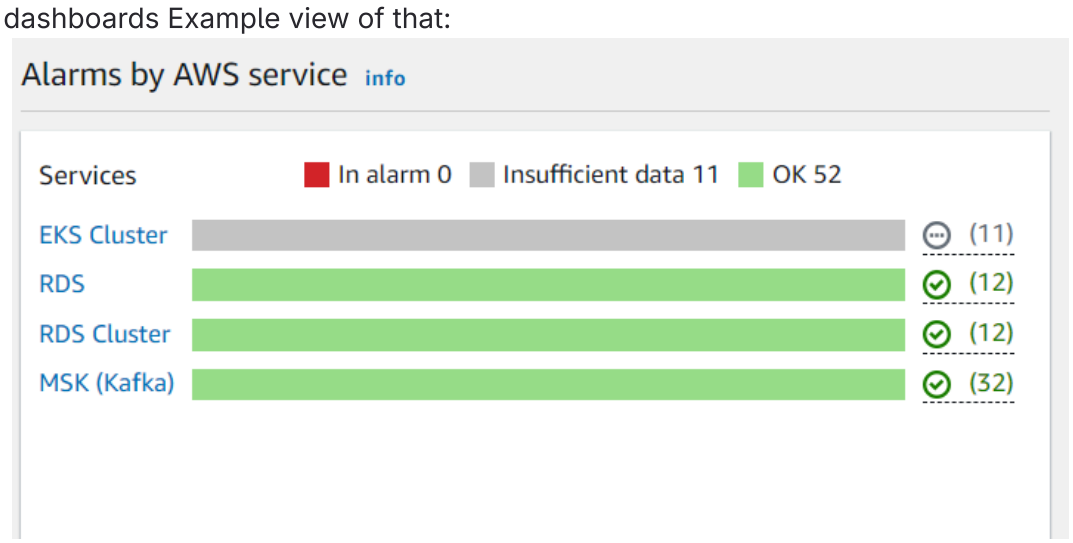
Troubleshooting

All troubleshooting is embedded within HOWTOs

HealthCheck

The functionality of the solution is determined by deploying the CloudWatch stack which is divided into two layers. More info in the table below and link: Architecture for monitoring and observability

Area	Approach

infra	infra deployment health monitoring is done as part of infra creation (cloudwatch is deployed along with modules). Independently built-in dashboards Example view of that: 
app	app deployment for health monitoring is delivered through terraform modules applying app level monitoring.

Backup And Recovery

The following approach for Backup and recovery is assumed in the solution

Area	Solution
Aurora	Solution by default apply Point In Time recovery with retention defined on deployment by parameter backup_retention_period
MSK	MSK is provided in HA mode. No built-in backup solution for MSK. Customers might add solutions based on mirroring or CDC on topics to add Backup functionality.
EC2 instances / EKS worker nodes	No stateful instances, and no provided in-solution backup option though the customer might implement a backup solution outside of the solution.

Routine Maintenance

Rotating static keys

PAT for GitHub integration and GitHub

Regarding DevOps perspective GitHub PAT is used by ArgoCD and GitHub actions runner controller. Both of them are stored in AWS Secrets Manager on appropriate accounts:

- ArgoCD -**digibank/argo/repo-creds** - every AWS account that runs EKS cluster with Argo (Apps, TM, Shared Service)
- actions-runner-controller - **digibank/actions-runner-controller/github-token **- Shared Service Account

HOWTO bootstrap github enterprise

Private key CA for AWS Service Mesh

Private key for AWS Service Mesh is stored in AWS Secrets Manager under digibank/appmesh/root-ca-bundle

All the secrets required by microservices are managed withing AWS Secrets Manager with automatic refresh in pod HOWTO configure AWS Secrets for microservices

Software patches and upgrades

Area	Instructions
EKS nodes AMI patches	HOWTO upgrade AMI latest version for EKS managed node group
BLX terraform upgrades	BLX terraform upgrades are distributed as source code packages to be built and installed by customers.
BLX application upgrades	BLX application upgrades are distributed as source code packages to be built & installed by the customer.
Aurora DB upgrade	HOWTO upgrade Aurora db

Managing licenses

The solution does not provide any sublicensing hence there is no explicit information about managing licenses. For products that are incorporated please rely on the following additional information:

Area	License instructions
Spacelift	https://spacelift.io/
Github	https://github.com/
CIS benchmark for Amazon Linux 2 AMI Marketplace subscription (optional for jumphosts)	<a href="https://aws.amazon.com/marketplace/pp/prodview-wm36yptaecjnu"

```
                                                                rel="nofollow">https://aws.amazon
```

Managing AWS service limits

Solution does not require by default changing default service limits.

Emergency Maintenance

Handling fault conditions

Depending on the specific area of fault following areas might be used as help

General troubleshooting microservices and blx specific code	Please see support section
SpaceLift	https://docs.spacelift.io/product/support/
ArgoCD	https://argo-cd.readthedocs.io/en/stable/SUPPORT/
AWS resources	https://docs.aws.amazon.com/awssupport/latest/user/troubleshooting.html

Software recovery

Main audit and history of changes can be tracked under HOWTO audit deployments history.

For reference about instruction

Area	**Notes**
undeployment of software artefacts (microservices)	These are the steps reversing the HOWTO setup Spacelift and provision Apps infra - 5 - microservices
undeployment of other components deployed in k8s	These are the steps reversing the HOWTO setup Spacelift and provision Apps infra - 5 - microservices
undeployment of infrastructure	This is about reversing actions in: HOWTO setup Spacelift and provision Apps & TM infra - 4 - application platform HOWTO setup Spacelift and provision Apps & TM infra - 3 - networking layer HOWTO setup Spacelift and provision Apps infra - 2 - configure infrastructure prerequisites HOWTO setup Spacelift and provision Apps & TM infra - 1 - org and accounts

Support

Support for customers is provided based on MSA and contract-level agreements. This includes:

- contact channels
- tiers of technical support
- defined SLA

Attachments:

image-2023-6-19_8-5-44.png (image/png) cloud-platform-asset-high-level-arch-examplev3.drawio.png (image/png) image-2023-6-19_16-40-12-1.png (image/png) image-2023-6-19_16-46-3.png (image/png) image-2023-6-19_17-28-26.png (image/png) blx2-platform-estimate-timelines.xlsx (application/vnd.openxmlformats-officedocument.spreadsheetml.sheet)