zArchive - Data Synchronization 4+1 Design

Created by Robert Florstedt(VA,Ctr), last modified on Feb 26, 2025

Introduction

The BIP Data Synchronization (Data Sync) application is a replacement for a legacy capability originally built inside of VBMS Core that supported synchronizing changes that occur to Person or Claim data in the Corporate Database (CorpDB) through an interface with Benefits Gateway Services (BGS). Data Sync extracts the logical elements of the legacy application within VBMS and restructures them as a standalone application for use both in updated VBMS capabilities as well as new capabilities requiring access to Person or Claim change events.

The basic design of Data Sync is similar in approach to that of the existing VBMS Claim Sync, however there are a few notable differences. The first difference is in the removal from scope of who or what is processing the changed data object and closing the synchronization process at the conclusion of determining the changes made to an object. The second difference is in the mechanism of how the data is handled within the solution. The introduction of streams (or more specifically Kafka Topics), in place of messaging queues and the removal of databases as part of the solution is a significant change. The use of Kafka topics provides a foundational element for future phases that can move from pull models of data synchronization to push models. The introduction of these Topics provides the opportunity to remove the need for a separate data store for the object data as these Topics are durable and will persist the lifecycle of changes to an object.

Additional architecture documentation is available from the VBMS DoDAF GitHub repository (master branch) and from the most recent PDF uploaded to VBMS Architecture DoDAF.

Use Case View

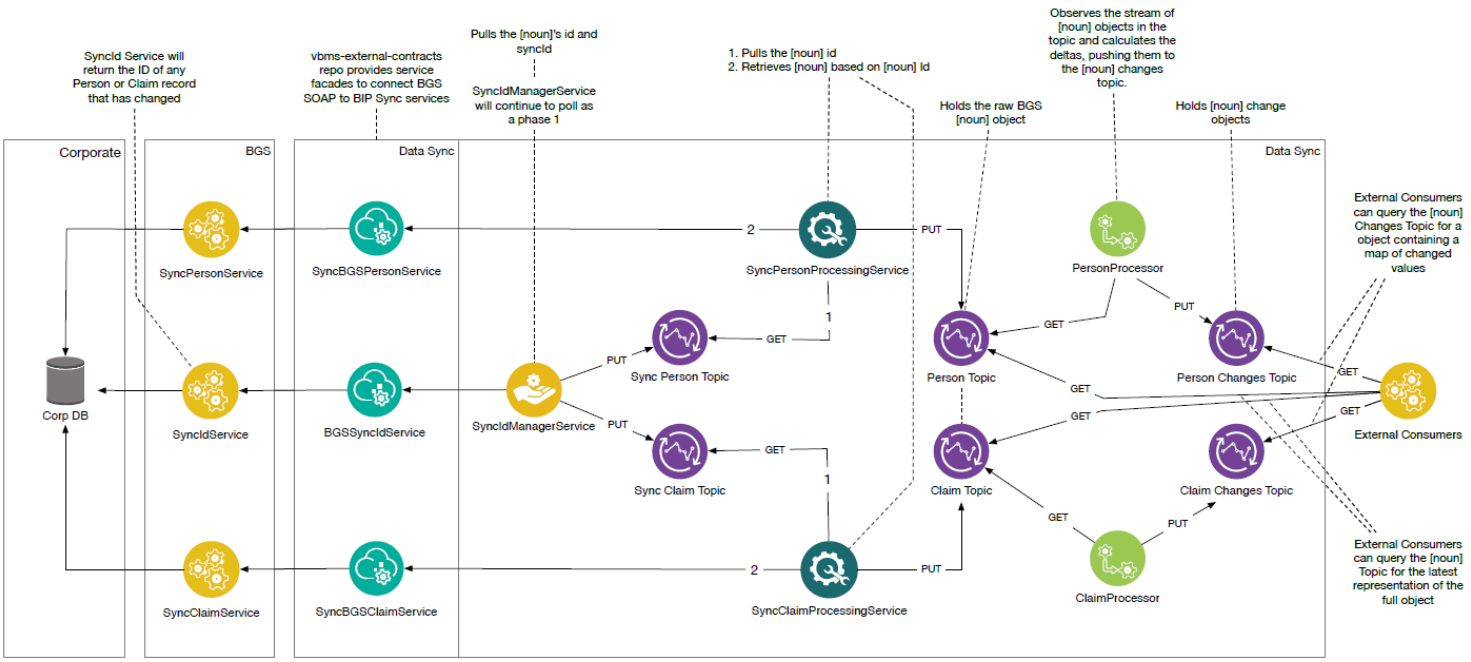
Data Sync is a background system process implemented to retrieve Person and Claim data from CorpDB when changes have occurred; it does not have users or a User Interface (UI).

Logical View

Main Components

Data Sync Main Components and *Data Sync Component References* depict the Data Sync component architecture.

Data Sync Main Components



Data Sync Component References

Component	Description
vbms-external-contracts repo	Provides service facades to connect BGS SOAP to BIP Data Sync services

SyncBGSClaimService	Interface between SyncClaimProcessingService and BGS SyncClaimService
SyncBGSPersonService	Interface between SyncPersonProcessingService and BGS SyncPersonService
BGSSyncIDService	Interface between SyncIDManagerService and BGS SyncIDService
ClaimProcessor	Returns a list of Claim objects based on list of Claim IDs
SyncClaimProcessingService	Watches the Sync Claim Topic for changed records, then retrieves the full object from BGS SyncClaimService (by way of SyncBGSClaimService) and pushes to the Claim Topic
PersonProcessor	Returns a sample info object based on search by PID
SyncPersonProcessingService	Watches the Sync Person Topic for changed records, then retrieves the full object from BGS SyncPersonService (by way of SyncBGSPersonService) and pushes to the Person Topic
SyncIDManagerService	Retrieves IDs of any Person or Claim record that has changed from BGS SyncIdService (by way of BGSSyncIDService) and pushes to the appropriate Kafka Topic (Sync Person or Sync Claim)
Kafka Topics	• Sync Claim Topic: Holds the changed Claim ID and Sync ID • Sync Person Topic: Holds the changed Person ID and Sync ID • Claim Topic: Holds the raw BGS Claim object • Person Topic: Holds the raw BGS Person object • Claim Changes Topic: Holds Claim change objects • Person Changes Topic: Holds Person change objects

External Systems

Data Sync interacts with CorpDB via BGS, and external consumers can utilize DS to query the Kafka topics to retrieve change information or objects. *Data Sync External Systems* describes Data Sync's relationship to external systems.

Data Sync External Systems

System	Component	Description
Corporate	CorpDB	Database containing Person and Claim records
BGS	SyncIdService	Returns the ID of any Person or Claim record that has changed
BGS	SyncPersonService	Returns the changed Person object
BGS	SyncClaimService	Returns the changed Claim object
External Consumer	Consumer Utilization	• Can query the Person Changes Topic or Claim Changes Topic for an object containing a map of changed values • Can query the Person Topic or Claim Topic for the latest representation of the full object

Logical Data Model

Data Sync provides the claim and person data required to migrate the National Work Queue (NWQ) from VBMS to BIP. The source for this data is the VA CorpDB accessed via BGS.

- **Sync Person Topic** and **Sync Claim Topic** hold the IDs of changed Person or Claim records.
  - SyncIdManagerService pulls the id and syncId from CorpDB via BGS and a service façade and pushes them to Sync Person Topic and Sync Claim Topic.
- **Person Topic** and **Claim Topic** hold the raw BGS objects (Person or Claim)
  - SyncPersonProcessingService and SyncClaimProcessingService pull the IDs from Sync Person Topic or Sync Claim Topic, retrieve the full object from CorpDB via BGS and a service façade, and push them to Person Topic or Claim Topic.
- **Person Changes Topic** and **Claim Changes Topic** hold the corresponding change objects (Person or Claim)
  - PersonProcessor and ClaimProcessor read the changed objects in Person Topic or Claim Topic, create a change object and push it to Person Changes Topic or Claim Changes Topic.

Process View

Business Process Model

Main Processes

The Data Sync solution is divided into three sequences:

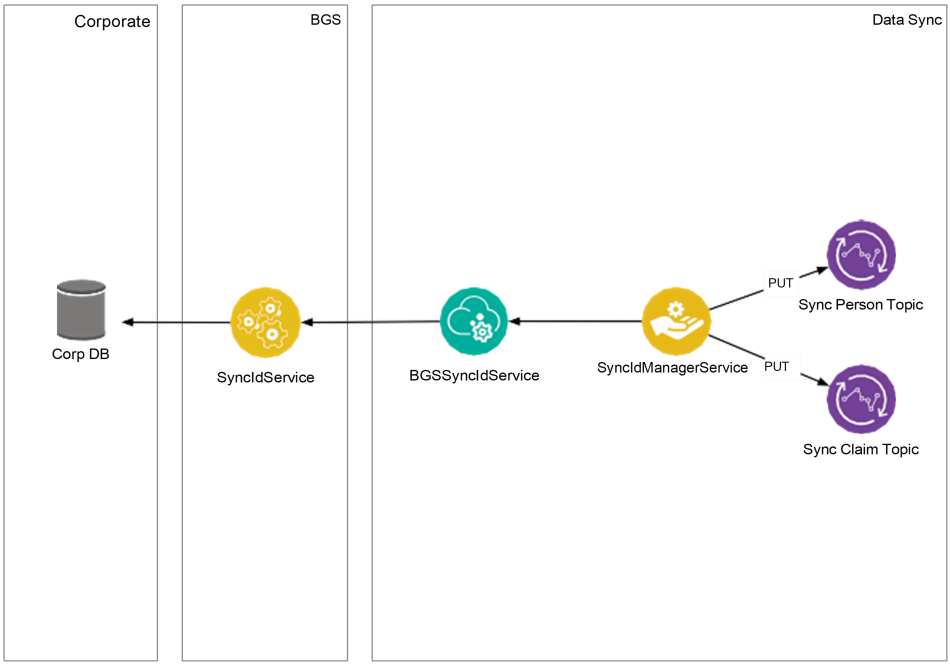
- Syncing from BGS
- Retrieving the Full Object
- Processing and Publishing the Changes

Syncing from BGS

The SyncIdManagerService retrieves IDs of changed Person or Claim objects from the BGS SyncIdService (by way of the Data Sync BGSSyncIDService) and pushes the ID to the appropriate Kafka Topic, Person or Claim.

Syncing from BGS Sequence and *Syncing from BGS Sequence Details* depict the Syncing from BGS sequence.

Syncing from BGS Sequence



Syncing from BGS Sequence Details

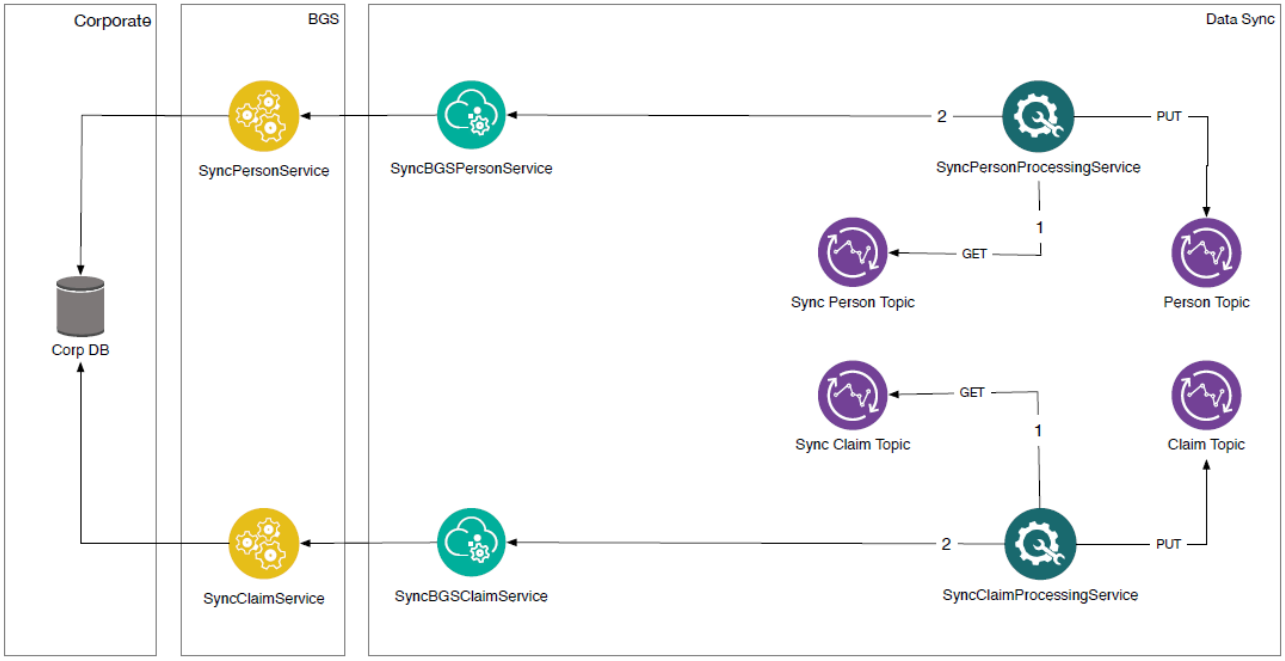
Name	Description
CorpDB	Database containing Person and Claim records
BGS	Includes SyncIdService that returns the ID of any Person or Claim record that has changed
Data Sync	Includes SyncIDManagerService that retrieves IDs of any Person or Claim record that has changed (from BGS SyncIdService by way of the Data Sync BGSSyncIDService) and pushes to the appropriate Kafka Topic (Sync Person or Sync Claim)

Retrieving the Full Object

The second sequence involves a pair of services, SyncPersonProcessingService and SyncClaimsProcessingService, that watch both the Sync Person Topic and Sync Claim Topic and react to additions to these topics. As IDs are added to these topics, the appropriate processing service retrieves the changed object (Person or Claim) from the appropriate BGS services and puts the fully hydrated object on either the Person or Claim Topic accordingly.

Retrieving the Full Object Sequence and *Retrieving the Full Object Sequence Details* depict the Retrieving the Full Object sequence.

Retrieving the Full Object Sequence



Retrieving the Full Object Sequence Details

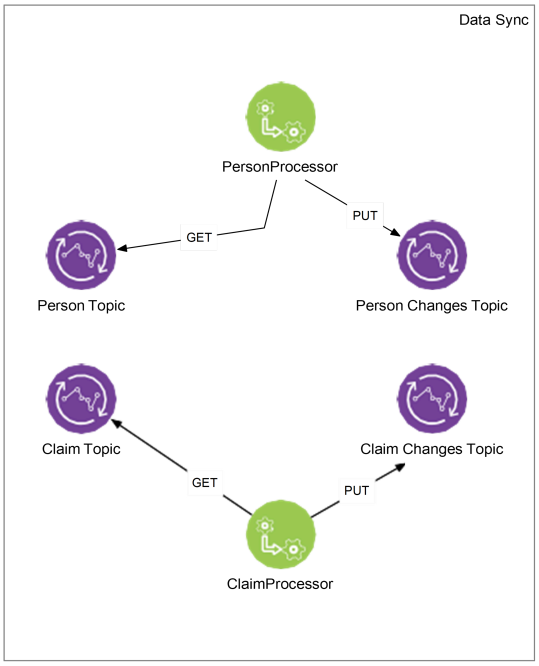
Name	Description
CorpDB	Database containing Person and Claim records
BGS	Includes SyncPersonService and SyncClaimServices that return the changed object (Person or Claim)
Data Sync	Includes SyncPersonProcessingService and SyncClaimProcessingService that watch the Sync Person Topic / Sync Claim Topic for changed records, then retrieve the full object from BGS SyncPersonService / SyncClaimService (by way of the SyncBGSPersonService / SyncBGSClaimService) and push to the appropriate Kafka Topic (Person or Claim)

Processing and Publishing the Changes

The third and final sequence in Data Sync is where each object topic, the Person or Claim Topics, is watched and when a newly updated object has been added to the topic, the appropriate processor reviews the object, and the previous state of the object on the topic, and creates a “change” object that is pushed onto the Person Changes Topic or Claim Changes Topic, as appropriate.

Processing and Publishing the Changes Sequence and *Processing and Publishing the Changes Sequence Details* depict the Processing and Publishing the Changes sequence.

Processing and Publishing the Changes Sequence



Processing and Publishing the Changes Sequence Details

Name	Description
PersonProcessor	Watches the Person Topic for newly updated objects to be added, reviews each updated object, creates a change object, and pushes the change object to the Person Changes Topic
ClaimProcessor	Watches the Claim Topic for newly updated objects to be added, reviews each updated object, creates a change object, and pushes the change object to the Claim Changes Topic

Implementation View

This section describes the layer and model separation design for applications implemented on the BIP Platform. The separation of layers (or “application tiers”) is enforced by the BIP Framework which defines three layers: Provider, Domain, and Partner. Separation of concerns is of primary importance for ensuring encapsulation of functional requirements. It is important that this separation of concerns also be reflected in the packages created for any application.

Application Layers

BIP Platform applications can be thought of as being comprised of three layers (or "tiers"), each of which encapsulates functionality for the layer:

- Provider layer (or Web Tier)
- Domain layer (or Service Tier)
- Partner layer (or Client Access Layer)

The Provider layer is where the "public" API for the application is exposed.

The Domain layer is where the business logic of the application takes place. Functions undertaken in this layer include:

- Business validation of data received from the Provider layer.
- Acquire additional data from Partner services (e.g., other services external to the application).
- Convert data to the Partner's object model.
- Request data through the Partner layer client.
- Convert the Partner response back to the domain model.
- Aggregate or manipulate data to the required state.
- Return resulting data to the Provider layer, communicating any issues encountered during processing of the service request.

The Partner Layer is comprised of a helper class and any other support classes needed to interact with a Partner Client module.

The BIP Platform development approach separates code into layers to achieve the following:

- Allow for service evolution (bug fixes, etc.) without requiring API version increments
- Allow for consumer API evolution without requiring service major-version increments
- Help avoid implementation changes in one layer from polluting implementation details in other layers
- Maintain clear separation of concerns, reducing developer confusion between layer responsibilities
- Reduce long-term maintenance burdens, thus saving funds and improving VA work processes

Data Sync is secured through a Java Web Token (JWT) using the VA trust store for its generation. All files are secured using an authorization framework provided by the BIP Security Services (BSS). The scope of how BSS works is beyond this document, but Veteran files are secured using BSS in conjunction with the JWT token.

Deployment View

When functionality is ready to be delivered, the developer opens a pull request in GitHub which gets approved by the tech lead. After it is merged into the development branch and built, it is promoted to the DEV environment. Once testing is complete at the end of the sprint, it is promoted to the STAGE cluster for further testing. After a successful UAT period has been completed, the build is identified for production readiness and promoted to the PERF environment. In PERF environment, tests are run to identify any performance issues due to any changes in the code.

Environments

Data Sync Environments lists and describes the environments for Data Sync.

Data Sync Environments

Environment	Description
DEV	Current development version, dev cluster
TEST	Sprint release version, dev cluster

INT	Environment where OBPI authors schedules, dev cluster
IVV/SQA	Sprint release version, stage cluster
UAT	Sprint release version, BGS link test dependency, stage cluster
PAT	Sprint release version, stage cluster
PDT	Sprint release version, stage cluster
DEMO	Previous release version, stage cluster
PERF	Performance testing, latest release ready for production, prod cluster
COLA	Previous release
PrePROD	Next production release
ProdTest	Next production release
Prod	Current production environment

Connectivity to External Systems

Data Sync does not connect to external systems outside of the VA network.

Primary System Technologies (SV-9) (VASI Technology Components)

Software Type	Software Vendor	Software Name	Software Version
Development Tool	Akuity	ArgoCD	2.8.6
Development Tool	GitHub	CodeQL	2.16.2
Development Tool	Confluent	Confluent Platform	7.3.0
Middleware: Systems Management Tool	HashiCorp	Consul	1.13.9
Development Tool	The Linux Foundation	Dex	2.37.0
Cloud Services/Server Virtualization	Amazon	Elastic Kubernetes Service (EKS)	N/A
Development Tool	The Linux Foundation	Flux	1.25.4
Development Tool	Broadcom	Gangway	3.2.0
Development Tool	N/A - Open Source	Git2Consul	0.12.13
Development Tool	The Linux Foundation	Helm	3.11.3
Development Tool	Oracle	Java	8
Development Tool	N/A - Open Source	Jenkins Continuous Integration Server	2.387.1
Development Tool	The Linux Foundation	Kubernetes	2.7.0
Development Tool	SonaType	Nexus	3.61.0
Development Tool	OAuth2 Proxy	OAuth2-proxy	7.4.0
Cloud Services/Server Virtualization	Amazon	OpenSearch	N/A

Development Tool	Palo Alto Networks	Prisma Cloud	21.08.514
Operating System	Red Hat	Red Hat Enterprise Linux (RHEL)	7.x
Development Tool	SonarSource SA	SonarQube	8.9.7
Development Tool	Broadcom	Spring Boot	2.x
Middleware: Systems Management Tool	HashiCorp	Vault	1.12.9
Middleware: Systems Management Tool	N/A - Open Source	Velero	1.11.1

System Configuration

The Data Sync network infrastructure resides within the AWS VPC. Network administration is managed by AWS and VAEC/NSOC. This includes:

- NAT
- Firewall policies
- New subnet requests
- Resolution of physical infrastructure issues

Via the AWS console, ProdOps can access the VPC for server management activities. These include:

- Configuring security groups
- Configuring the application and network load balancers
- NACL

System Monitoring and Metrics

Data Sync's availability will be ensured by actively monitoring the PROD environment to detect and resolve minor incidents before they become serious problems. Standard industry and application-integrated tools are used to alert PRODOps. These tools include AppDynamics, Oracle Enterprise Manager (OEM), Nagios, Hazelcast, and Splunk, as well as custom scripts. These tools allow PRODOps to resolve potential issues and to contact end users before these issues become outages. They also allow for real-time monitoring by providing alerts to both PRODOps and VA business managers.

System Auditing

Build audits are performed to verify the product matches. The Configuration Items (Cis) described in the specification, other documents, and the package is reviewed before release. The audits and reviews vary in complexity and formality. At a minimum, each named build goes through the following audits:

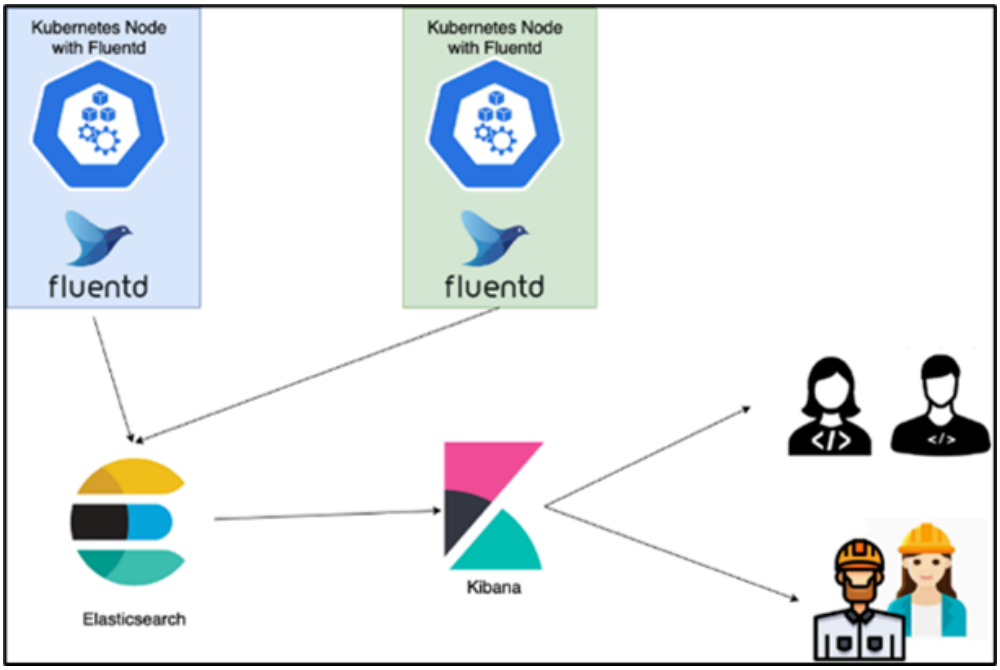
- A comparison of the Version Description Document (VDD) and the build change log
- A comparison of the VDD and query of Jira or GitHub work items tagged with the build label
- A review of the change log and work items contained in the build to ensure that code commits were targeted to the appropriate repository stream
- A review of Jira or GitHub work items to ensure appropriate detail and work item state and traceability to requirements and test cases are captured

Configuration audits are broken into functional and physical configuration audits. They occur either at delivery or now of effecting the change. Configuration audits are functional and physical. A functional configuration audit ensures functional and performance attributes of a CI are achieved. A physical configuration audit ensures a CI is installed in accordance with the requirements of its detailed design documentation.

System Logging

Logging follows practices of the BIP Platform. For detailed logging on the clusters and the tenant’s applications, the BIP Platform leverages an Elasticsearch, Fluentd and Kibana (EFK) stack to aggregate and visualize logs. Fluentd captures the logs and forwards them to AWS S3 and Elasticsearch. Elasticsearch has similar functionality to a database and stores these logs and indexes for fast retrieval. Kibana is then used by developers and operations engineers to see the logs stored in Elasticsearch. Logs are duplicated into sS3 and retained there for a certain period of time (30 days non-PII environments, 90 days PII environment) after which they are lifecycled from S3 Standard to S3 Glacier, which is a cheaper storage class for objects that have to be accessed infrequently. *BIP Platform Logging* shows the components of BIP Platform logging.

BIP Platform Logging


Kubernetes Node
with Fluentd
fluentd
Kubernetes Node
with Fluentd
fluentd
Elasticsearch
Kibana

No labels