# Introduction

This document outlines the Fingerprint Architectural components and describes how the system fits together.

There are several components that make up the full platform and each is detailed in this document:

[Introduction 1](#_Toc26178905)

[Ingest & Archive 2](#_Toc26178906)

[Email 2](#_Toc26178907)

[Bloomberg 3](#_Toc26178908)

[Slack 5](#_Toc26178909)

[Voice 6](#_Toc26178910)

[Data Replication 7](#_Toc26178911)

[Web Front End 8](#_Toc26178912)

# Ingest & Archive

Ingest is the process of listening for or downloading data from a source, processing the contents of the data, uploading it to *ElasticSearch* for indexing and saving an original copy of the data in an Archive.

This work is undertaken in a set of Serverless *AWS* components.

## Email

A close up of a map

Description automatically generated

1. The system accepts email from any system that has journaling capabilities. *O365* and *GSuite* are very popular cloud-based systems, along with an on premise or private cloud *Exchange Server*. Using a ruleset on the local email system allows for only a subsection of email to flow to the Fingerprint platform if required. IP Sentinel provide a unique journal address for each client.
2. We use the *Amazon Simple Email Serve* to process inbound mail. The gateway uses TLS to secure the traffic. It has a maximum capacity of 30MB size which is more than adequate for business email. When a journal message is delivered to *SES* it is parsed through a set of rules to determine to which client it belongs. The email is then stored into your dedicated s3 bucket (3) and an *AWS* Lambda task (4) is instantiated to process the email.
3. Each client has a dedicated storage bucket. The bucket has a lifecycle policy that keeps data in S3 for 7 days and then migrates the data to S3 Glacier for long term storage. The bucket has a data retention of five years pre-set with data being automatically deleted when the retention period expires. Retention periods can easily be extended beyond five years by request. All data is encrypted with a 256Bit key. The S3 bucket has a set of ‘directories” where data is moved though the download/enrichment/upload and archive cycle.
4. The email processing lambda takes the raw MIME encoded email from S3, decodes the contents and populates the Fingerprint email schema and the relevant client and message metadata. Any text-content attachments are prepared for upload, the lambda then uploads the content of the email to the *Elasticsearch* cluster (5) and finally tidies up any temporary files in S3 and moves the email to the archive once *Elasticsearch* has confirmed the upload has succeeded.
5. The ElasticSearch cluster extracts text from any attachments and then indexes email content for search and retrieval in the Web Front end.

## Bloomberg

A close up of a map

Description automatically generated

1. The *Bloomberg* task is scheduled daily using *AWS CoudWatch* events. Each client is scheduled separately. Heuristically we have found the best time to run the tasks is between 9:00 and 11:00 the day after the data was captured by *Bloomberg*. The CloudWatch event instantiates an AWS Step Function (2).
2. As the *Bloomberg* process is multistep and requires careful orchestration, we use the AWS Step Function service. The step function triggers a series of lambdas that perform specific tasks. If the tasks error out then the Step Function retries them, also if a lambda gets close to its execution time limit the step function captures its state and continues processing in a fresh lambda. The lambdas’ the step function orchestrates are:
   1. **Bloomberg Download**  
      Using a fixed EC2 IP Address (3) and a dedicated ssh key (4) for the sFTP session the lambda downloads the encrypted Bloomberg daily download files to S3 (6)
   2. **Download Decode**  
      Using the specific client encryption key the daily download files are decrypted for parsing.
   3. **BBG IB Process & Upload**  
      The IB XML file is processed to re-construct each chat so that the Fingerprint system can show who actually received a given IB message or file. Each conversation generates multiple records to upload to Elasticsearch (7). Each conversation is batch uploaded. As it is compute heavy to process the BBG IB conversations the Lambdas often reach their 15 min execution time limit. Each Lambda is self-aware in terms of how long it has left to live. If the processing of the file is incomplete, the lambda will hand back a pointer to the Step function as to where it has got to within the BBG XML file. The Step Function will then start another Lambda which will pick up where the previous lambda left off.
   4. **BBG MSG Process & Upload**

As BBG MSG is the same as email each message is self-contained, so it is processed and uploaded in one action rather than implementing a batch upload process. As with the BBG IB Lambdas the BBG MSG lambda are self-aware and will hand a pointer to the Step Function prior to expiring and the Step function will again start another Lambda which will know where to pick up processing again.

* 1. **File clean-up**  
     Once the BBG MSG and IB XML files have been processed the file clean-up moves the relevant files to the Archive and removes any temporary working files.

1. Bloomberg requires all downloads to be requested from a fixed IP address which is whitelisted on their system. We use an Elastic IP Address on an EC2 instance that can be mapped to the sFTP Lambda. This means that we can satisfy the BBG requirements.
2. Each customer can have their own ssh Key and decryption certificate. We store these as parameters tn AWS Systems Manager on a client by client basis
3. We only access Bloomberg via sFTP. This is compatible with archiving to *Global Relay* in parallel if required.
4. Each client has a dedicated storage bucket. The bucket has a lifecycle policy that keeps data in S3 for seven days and then migrates the data to S3 Glacier for long term storage. The bucket has a data retention of five years pre-set with data being automatically deleted when the retention period expires. The retention period can be extended if required. All data is encrypted with a 256Bit key. The S3 bucket has a set of ‘directories” where data is moved though the download/enrichment/upload and archive cycle.
5. The ElasticSearch cluster extracts text from any attachments and then indexes Bloomberg content for search and retrieval in the Web Front end.

## Slack

A close up of a map

Description automatically generated

1. The Slack task is scheduled hourly using AWS CoudWatch events. Each client is scheduled separately. The CloudWatch event instantiates a Lambda task (2).
2. The lambda task logs into Slack and uses the Slack REST API (3) to download content since the previous login, content is saved to S3 (4). The content is processed by slack type and Bulk uploaded to *ElasticSearch* (5). Once The content has been successfully uploaded the Lambda cleans up any temporary files in S3.
3. The Slack REST API credentials are stored on a client by client basis in the AWS Systems Manager parameters store.
4. Each client has a dedicated storage bucket. The bucket has a lifecycle policy that keeps data in s3 for seven days and then migrates the data to S3 Glacier for long term storage. The bucket has a data retention of five years pre-set with data being automatically deleted when the retention period expires. All data is encrypted with a 256Bit key. The S3 bucket has a set of ‘directories” where data is moved though the download/enrichment/upload and archive cycle.
5. The ElasticSearch cluster extracts text from any attachments and then indexes slack content for search and retrieval in the Web Front end.

## Voice

A close up of a map

Description automatically generated

1. The Voice task is scheduled to the client’s request using AWS CoudWatch events. Each client is scheduled separately. The CloudWatch event instantiates a Lambda task (4) to perform the transcoding of the calls.
2. The Redbox system is set up to send calls via the Redbox Data Consumer (Export Broker)

A screenshot of a cell phone

Description automatically generated

1. A CloudFormation front end is set up to provide HTTPS, authentication and IP Whitelisting for the Data Consumer which is ultimately hosted in an S3 bucket.
2. The lambda loads data from the S3 Bucket. The call audio and the call metadata. The call audio is transcribed to text using the Amazon Transcribe (5) and then merged with the call metadata before the record is uploaded to ElasticSearch (7).
3. Amazon Transcribe uses advanced machine learning technologies to recognize speech in audio files and transcribe them into text.
4. Each client has a dedicated storage bucket. The bucket has a lifecycle policy that keeps data in s3 for 7 days and then migrates the data to S3 Glacier for long term storage. The bucket has a data retention of 5 years pre-set with data being automatically deleted when the retention period expires. Retention periods can be extended upon request. All data is encrypted with a 256Bit key. The S3 bucket has a set of ‘directories” where data is moved though the download/enrichment/upload and archive cycle.
5. The ElasticSearch cluster indexes the text extracted from the Audio conversation and the metadata about the call content for search and retrieval in the Web Front end.

# Data Replication

A close up of a map

Description automatically generated

Data is replicated from Amazon S3 to an Azure storage blob on a defined client schedule. The schedule is managed by an Azure Data Factory and is set to 24 hours. This period can be set as low as 5 minutes.

The data is stored in an encrypted Azure file Blob on a client by client basis. This provides a fully redundant data backup for all archive data.

# Web Front End

A close up of a map

Description automatically generated

1. Systems Users are all manually created by IP Sentinel staff. This is a security protocol that prevents unauthorised access to the system. The system itself is also aware of a user’s location and only permits access from whitelisted IP addresses.
2. The web front end is designed to be highly resilient and scalable (in a similar way to the Lambda technology). The Application Load Balancer ensures that users remain attached to the appropriate back end components.
3. The web site is hosted in a set of Docker instances in an ECS cluster (4) this is split across availability zones. The clusters resources are provided by the AWS VPC service.
4. The application is hosted on Docker instances which are automatically scaled to provide massive horizontal capacity (more user access capacity).
5. The application itself is based on five major functional areas;
   1. **Authentication & Permission**

There is a multi level security system that provides for Chief Compliance Officer -> Manager -> Compliance Staff. Each level has specific view and read permissions that need to be managed.

* 1. **User Front End**  
     The system framework is written in Django and the front end is in Vue. There are many plug in components and a large amount of systems plumbing to implement the Fingerprint Platform Search and Case Management systems.
  2. **Case Management & Search**  
     The case management and search functions are what ties the entire Fingerprint Platform together. The data and processes are stored in the RDS Database (7) and referenced to the ElasticSearch data store (9).
  3. **Reporting**
  4. **REST API**

1. The multifactor authentication for users is provided by *Cisco Duo Security* which is integrated with the standard *Django* security layer.
2. All data that is non-Supervision is stored in the *Amazon RDS* data service emulating a PostGres data source. This is resilient across multiple geo locations and is automatically archived
3. Each client has a dedicated storage bucket. The bucket has a lifecycle policy that keeps data in S3 for seven days and then migrates the data to S3 Glacier for long term storage. The bucket has a data retention of five years pre-set with data being automatically deleted when the retention period expires. Retention periods can be extended upon request. All data is encrypted with a 256Bit key. The S3 bucket has a set of ‘directories” where data is moved though the download/enrichment/upload and archive cycle.
4. The *ElasticSearch* provides a relevance search across the indexed content of all ingested data types. The front-end discovery agents use the *ElasticSearch* cluster as the discover mechanism for Suspicious terms and Ad-Hoc searches.