Healthcare Data Lake

KENpAL , JosEpH University of Bristol jkl7246@bristol .ac .uk

## BENsoN, LuKE

University of Bristol

wrl96〇6@bristol .ac .uk

SHERREp , JAco

University of Bristol

j .sherred .2〇l9@bristol .ac .uk

## L1u, ANNA

University of Bristol

gfl99l6@bristol .ac .uk

C1sMARu , ARMANp

University of Bristol

fzl9792@bristol .ac .uk

## November l5, 2020

**Ａbstract**

Digital healthcare provided by the NHS in England typically operates in silos. GPs have electronic systems to manage patient care which are distinct from hospital systems which are distinct from the ambulance service, lll, mental health services etc. Each data owner has a wealth of data that, if combined, would generate a more valuable resource than it does in isolation. Ｗhile there are solutions to integrate this data for direct care purposes, there is no centralised solution to use this data to inform future care or service provisioning. This project is designed to explore the benefts of cloud technologies to produce a prototype secure , scalable health data storage platform that can underpin local healthcare analytics.

# **0verview**

## **Client**

Dr. Philip D Harfeld, Health Data Scientist (Informatics) at NIHR Bristol Biomedical Research Centre, University of Bristol.

## **Doｍain**

The domain of our Healthcare Data Lake project is the Healthcare Analytics Environment team who will design, implement and test a set of candidate cloud-hosted analytics environments that provide su 伍 cient functionality while also maintaining a secure data environment.

The overall domain of the three teams is NHS Healthier Together Sustainability Transformation Partnership Bristol, North Somerset, and South Gloucestershire (BNSSG) & Bristol Biomedical Research Centre, University of Bristol Ｍedical School.

## **Project**

The project entails combining a wealth of data from data owners(GPs Patient data, ambulance services, 111, mental health services .etc) into a data lake. This will be used to inform clinical

decisions making by providing more advanced insights into the longitudinal health of the patient on arrival and understand the merits of previous clinic decisions taken.

The project will explore the benefts of cloud technologies and produce a prototype secure, scalable health data storage platforms that can underpin local healthcare analytics.

The project is one of three designed an end-to-end proof of concept to the local NHS. Ｗe will be working alongside the "Healthcare Data Visualisation" and "Healthcare Analytics Environment" teams.

## **Ｖision**

The Healthcare Data Lake Project is envisioning a future integrated data storage solution, one that is scalable and portable, while using the latest cloud-based technologies. Starting with a prototype, the fnal scope is to create, alongside the Simulation and Analytics Projects, a system that is going to change how data is handled and used in the healthcare system. There is the real possibility that this three projects will represent the cornerstone of a future solution used and developed extensively by the NHS, which will bring immediate aid to the average medical worker and improve the quality of the service provided to the patients.

# **Stakeholder and Requireｍents Analysis**

## **Priｍary Stakeholder and User Story**

Philip Harfeld at Bristol, North Somerset and South Gloucestershire CCG (BNSSG). Philip Harfeld is our client, and this software is being developed for him at BNSSG. The user story for this stake- holder is that BNSSG require a piece of software that can be used to allow long-term healthcare data analytics from multiple data sources to inform clinical and strategic decisions. The reason for this is understanding the longitudinal health of a patient allows understanding of the merits of previous clinical decisions taken. In addition it can be used to inform strategic commissioning decisions using data on how efective services ofered to patients have been.

* + 1. **Flow steps**

Considering the user story above, we can breakdown the story into a sequence of steps of user ow.

* + - 1. Local healthcare services provide data to the data lake.
      2. Data is transformed and loaded into the data lake.
      3. The data is catalogued in the data lake and stored in structured data marts as required.
      4. The data lake allows access to a data analytics environment.
      5. An analytics environment can run queries on the data lake and report results to analytics environment.
      6. Clinical and strategic decisions can be taken based on analysis.

Ｗe can also identify alternative or exceptional ows.

**Exceptional Flow:**

1. Local healthcare services provide data to the data lake.
2. Data is not provided in an acceptable format to the API.
3. The healthcare service receives an error message to provide data in standard format.
   * 1. **Atoｍic Ｒequireｍents**

Ｗe can breakdown these steps into atomic requirements of the software.

* + - 1. An API takes in data from local healthcare providers as a HL7 FHIR message. Ｗe have chosen HL7 FHIR as it is the [UK standard for transferring healthcare messages.](https://digital.nhs.uk/services/fhir-uk-core)
      2. These data messages are stored in a cloud solution in a well structured data model.
      3. Ingested data is catalogued.
      4. An ETL tool is used to curate data marts.
      5. These data marts can be queried by the analytics environment. There are also additional requirements specifed for the software:

1. Ｍedical data is to be stored independently from pseudonymised patient identifers.
2. Provide a user console to monitor automated ETL jobs.
3. Provide full audit trails.

## **Additional stakeholders and User Stories**

This software will provide services to a number of local healthcare organisations such as NHS trusts and the Healthier Together STP and such all these additional users are secondary stakeholders to this project. These organisations will need to be able to provide healthcare information to the software which will need to be able to load and store the data for future analysis.

* + 1. **Flow Steps**

Considering the user story above, we can breakdown the story into a sequence of steps of user ow.

* + - 1. Healthcare provider (e.g. Hospital Trust) provide data to the data lake by a HL7 FHIR message.
      2. The data is stored in the data lake.

# **Personal Data, Privacy, Security and Ethics Ｍanageｍent**

## **GDPR**

The data lake solution developed by the “Healthcare Data Lake" project team is an integrating part of the larger prototype system, alongside "Healthcare Data Visualisation" and "Healthcare Analytics Environment" teams *(section 1.3)*. The proposed system is going to be used in compli-

ance with the *[NHS Digital GDPR](https://digital.nhs.uk/about-nhs-digital/our-work/keeping-patient-data-safe/gdpr)* compliance implementation and the liability for the personal

data stored falls onto the respective primary stakeholder, Philip Harfeld at Bristol, North Som- erset and South Gloucestershire CCG (BNSSG) *(section 2.1)*. The team developing the data lake infrastructure is responsible for creating a prototype secure, robust and scalable health data storage platform used for ingesting and interrogating patient's data under the HL7 FHIR standard *(section 4.1)*. As a client-specifed requirement, medical data is going to be stored independently from the pseudonymised patient identifers, thus protecting the identity and integrity of the patients whose data is stored in the data lake. The security of the data ingested as described in *section 3.2* is aligned with the *[NHS Digital GDPR](https://digital.nhs.uk/about-nhs-digital/our-work/keeping-patient-data-safe/gdpr)* compliance and practice.

## **Security**

The data lake solution provided Amazon Simple Storage Service(Amazon S3) as the object storage service of the data lake with advanced scalability, data availability, security and performance. It helps stores users data and secure it from unauthorized access with encryption features and access management tools. Amazon S3 offers flexible security features to block the unauthorized users from accessing users data. VPC endpoints connect S3 from Amazon Virtual Private Cloud(Amazon VPC), support server-side and client-side encryption for uploading data and use S3 Inventory to check encryption status of S3 objects. S3 Block Public Access ensures S3 buckets and objects do not have public access. Access Analyzer for S3 monitors users’ bucket access policies and ensure that the policies provide only the intended access to S3 resources. Users can also use Amazon Macie to make to protect sensitive data which stored in Amazon S3. All Amazon S3 resources defaultly have buckets, objects and related subresources, which can only be accessed by owner and AWS account that created it.

* 1. **Ethics**

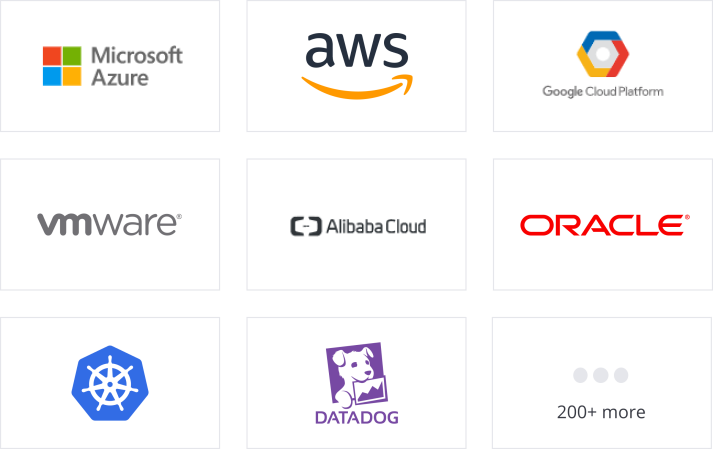
# **Architecture**

## **Introduction**

Ｗe propose a modular, (cloud)platform-independent solution that ofers high scalability and per- formance at a low cost for maintenance, development and deployment. The key to achieving this is leveraging the practises of infrastructure-as-code (IaC), serverless architectures and open standards. Therefore, development expense is focused on providing the most value, security and exibility to the user.



**IaC** By building infrastructure through extensible confguration fles, we make it easy to build, test, secure, update and rollback versions of architecture which can combine multiple cloud or on-premise services. Terraform is an IaC framework that supports all major cloud providers in addition to self-hosted options such as Kubernetes. This makes it a popular choice for a modern infrastructure team that seeks to avoid vendor lock-in and easily protect the security of it with robust tests and auditing.



**Serverless** Serverless computing is a cloud computing execution model in which the cloud provider runs the server, and dynamically manages the allocation of machine resources. Pricing is based on the actual amount of resources consumed by an application, rather than on pre-purchased units of capacity "[1](#_bookmark0)]. The benefts of this approach are the huge reduction in infrastructure and development expense. Engineers can focus on shipping their microservices and have secure, scalable infrastruc- ture taken care of for them.

The Serverless Framework is also an example of IaC and provides developers with the ability to develop and deploy their serverless application to diferent cloud providers or simply Kubernetes (using **Ｋnative**) if they wanted to run it on-premise or avoid code exposure to a cloud native service such as AＷS Lambda or Google Cloud Functions.

This also provides quicker developer on-boarding and exibility as a broad array of popular lan- guages are supported. Given that the Function as a Service (FaaS ) model is centred around the

principles of loosely coupled microservices, this enables easy migration out of serverless architecture to a VＭ or container-based deployment. This may make sense in scenarios where there are cost savings to be made which typically applies to long-running, memory intensive tasks as opposed to short-lived and resource-light stateless services.

Provider Supported Languages

AＷS Lambda Python, Java, Go, Node, .NET, Ruby or Custom runtime

Google Cloud Functions Node, Python, Go and Java Azure Functions .NET, Node, Java, PowerShell

and Python

IBＭ Cloud Functions Node, Python, Swift, PHP, Go,

Ruby, Java and .NET

Cloud are Ｗorkers JavaScript, TypeScript or Ｗasm-supported

**Open standards** Open Standards allow people to share all kinds of data freely and with perfect fdelity. They prevent lock-in and other artifcial barriers to interoperability, and promote choice between vendors and technology solutions. "[2](#_bookmark1)]

**OpenAPI** The OpenAPI Specifcation (OAS) defnes a standard, language-agnostic interface to RESTful APIs which allows both humans and computers to discover and understand the ca- pabilities of the service without access to source code, documentation, or through network tra 伍 c inspection. Ｗhen properly defned, a consumer can understand and interact with the remote ser-

vice with a minimal amount of implementation logic. An OpenAPI defnition can then be used by documentation generation tools to display the API, code generation tools to generate servers and clients in various programming languages, testing tools, and many other use cases. "[3](#_bookmark2)]

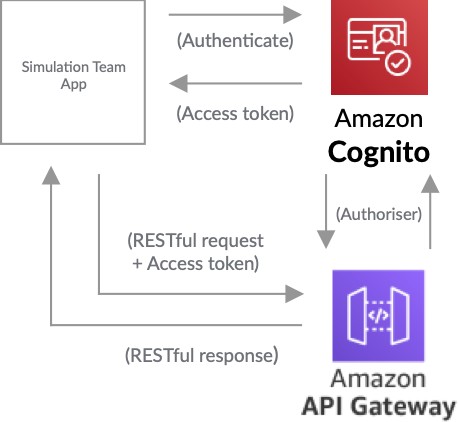
**ＨL7 FＨIＲ** *Fast Healthcare Interoperability Resources* (FHIR , pronounced "fre") is a standard describing data formats and elements (known as "resources") and an "API] for exchanging electronic

health records (EHR ). The standard was created by the Health Level Seven International (HL7 ) health-care standards organization. One of its goals is to facilitate interoperation between legacy health care systems, to make it easy to provide health care information to health care providers and individuals on a wide variety of devices from computers to tablets to cell phones, and to allow third-party application developers to provide medical applications which can be easily integrated into existing systems. "[4](#_bookmark3)]

## **Data ingestion**

* + 1. **API ｍanageｍent Authentication**

**Gateway**



Example: AＷS API Gateway

* + 1. **Ｍicroservices**



Example: AＷS Lambda

* + 1. **Ｍetadata cataloguing Glue**
    2. **Scheｍa-less object store S3**

## **Data warehousing**

* + 1. **Federated querying**
    2. **Coｍｍon ｍodels**

## **ETL & data ｍarts**

* + 1. **Scheduling**
    2. **Developers**
    3. **Console**

## **Secure access**

* + 1. **Encryption**

Encryption is the process of turning plain text into a cipher, then using an encryption key to decrypt the cipher back to plain text.

* + 1. **Concept**

Encryption is the process of turning plain text into a cipher, then using an encryption key to decrypt the cipher back to plain text.

Ｗhen data is received on the server to be encrypted it will require an encryption key to encrypt the data, this key is typically generated by either a key manager. Ｗhen data is being retrieved or queried it will then use the same key to decrypt the data to be used.

A key manager is used to generate, exchange, use, replace and delete cryptographic keys. It's able to deal with asymmetric and symmetric keys. Some key managers encrypt keys after generation to add another layer of security.

Client-side encryption is an alternative option is to encrypt the data locally in the application before it gets to the server. This means the server is not involved in the cryptographic process of the data, as the application will use its own encryption key which the server cannot access. This is benefcial for the users as it provides them peace of mind.

* + 1. **Solutions**

In cloud computing services “Ｍicrosoft Azure Data Lake Storage Gen1", “Google Cloud" and “AＷS" all provide options for the client and server-side encryption.

* + 1. **Exaｍple**

Ｗe will be looking at AＷS S3, AＷS Redshift and AＷS KＭS for an example of the implementation of encryption of data lakes.

**AＷS S3**

Server-side encryption uses three diferent modes of server-side encryption: SSE-S3, SSE-C, or SSE- KＭS. The following will be an example of SSE-KＭS.

SSE-KＭS enables the choice of a customer-managed CＭK (customer master key) or the AＷS managed CＭK for Amazon S3 in the account. AＷS KＭS and Amazon S3 perform the following actions on encryption:

* Amazon S3 requests a plaintext data key and a copy of the key encrypted under the specifed CＭK.
* AＷS KＭS generates a data key, encrypts it under the CＭK and sends both the plaintext data key and the encrypted data key to Amazon S3.
* Amazon S3 encrypts the data using the data key and removes the plaintext key from memory as soon as possible after use.
* Amazon S3 stores the encrypted data key as metadata with the encrypted data.

Amazon S3 and AＷS KＭS perform the following actions when data is requested to be decrypted:

* Amazon S3 sends the encrypted data key to AＷS KＭS.
* AＷS KＭS decrypts the key by using the same CＭK and returns the plaintext data key to Amazon S3.
* Amazon S3 decrypts the ciphertext and removes the plaintext data key from memory as soon as possible.

Client-Side Encryption with AＷS can use Amazon S3 Encryption Client in the AＷS SDK of an application to encrypt objects and upload them to Amazon S3. This method ensures its security as it passes to the Amazon S3 service. The Amazon S3 service receives the encrypted data; it does not play a role in encrypting or decrypting it.

**AＷS Ｒedshift**

Amazon Redshift uses a four-tier, key-based architecture for encryption. The architecture consists of data encryption keys, a database key, a cluster key and a master key.

Data encryption keys encrypt data blocks in the cluster. Each data block is assigned a randomly generated AES-256 key. These keys are encrypted by using the database key for the cluster.

The database key encrypts data encryption keys in the cluster. The database key is a randomly generated AES-256 key. It is stored on disk in a separate network from the Amazon Redshift cluster and passed to the cluster across a secure channel.

The cluster key encrypts the database key for the Amazon Redshift cluster. AＷS KＭS, AＷS CloudHSＭ, or an external hardware security module (HSＭ) can be used to manage the cluster key.

The master key encrypts the cluster key. AＷS KＭS customer master key (CＭK) can be used as the master key for Amazon Redshift.

**AＷS ＫＭS** AＷS Key Ｍanagement Service (KＭS) allows you to create and manage crypto-

graphic keys and control their use across a wide range of AＷS services and in the applications. AＷS KＭS uses hardware security modules that have been validated under FIPS 140-2, or are in the process of being validated, to protect keys. AＷS KＭS is integrated with AＷS CloudTrail to provide logs of all key usage to help meet applicable regulatory and compliance needs.

Ｗhat is going on when S3 handles the process of server-side encryption:

* Ｗhen an object is upload into S3 and specify SSE, a unique 256-bit key (a data key) is gen- erated and the data is encrypted using that key with AES-256.

images/KＭSDiagram1.png

* The data key is then encrypted using a master key, which S3 retrieves from KＭS using the master key ID that was supplied in the s3 cp command.

images/KＭSDiagram2.png

* The encrypted data key is then stored with the encrypted data in the S3 bucket. images/KＭSDiagram3.png

**Overview**

images/EncryptionOverview.png The major steps in this process are

* + - 1. You create a master key in AＷS KＭS
      2. You load the data into S3.
      3. S3 retrieves the key from KＭS and encrypts the data.
      4. You run the COPY command in Amazon Redshift to load the data from S3.
    1. **IAＭ**
    2. **Concept**

Identity and Access management (IＭA) is used to make sure authorised users have the right to services while preventing access to non-authorised users guarantying secure access. It does this by defning and managing the roles, the access privileges of individual users and the circumstance users are granted those privileges.

IAＭ allows access management over computing, storage, database and application services. It gives one identity per individual. Once that identity has been introduced, it must be maintained, modifed and monitored throughout each user's access lifecycle.

* + 1. **Solutions**

Each cloud service provider ofers a built an associated IAＭ. Independent IAＭ includes Okta Identity Cloud, JumpCloud and Auth0 which are all compatible with a range of cloud service providers.

* + 1. **Exaｍples**

Ｗe will be examining AＷS Identity and Access Ｍanagement. IAＭ is a feature of an AＷS account ofered at no additional charge.

The use cases of AＷS IAＭ include fnite-grained access control to AＷS resources, Ｍultifactor authentication for high privilege users, analyse access across the AＷS environment and ability to integrate with the corporate directly (integration of current IAＭ system).

**Ｈow it works**

AＷS IAＭ allows us to:

* Ｍanage IAＭ users and their access - Create users in IAＭ, assign them individual security credentials (access keys, passwords and multi-factor authentication devices), or request tem- porary security credentials to provide users access to AＷS services and resources. Ｍanage permissions in order to control which operations a user can perform.
* Ｍanage IAＭ roles and their permissions - Create roles in IAＭ and manage permissions to control which operations can be performed by the entity, or AＷS service, that assumes the role. Defne which entity can assume the role. In addition, you can use service-linked roles to delegate permissions to AＷS services that create and manage AＷS resources automatically.
* Ｍanage federated users and their permissions - Enable identity federation to allow existing identities (users, groups and roles) in the enterprise to access the AＷS Ｍanagement Console, call AＷS APIs and access resources, without the need to create an IAＭ user for each identity. Enable the use of any identity management solution that supports SAＭL 2.0.
  + 1. **Logging**
    2. **Concept**

A Log is data that can describe application information, system performance and user activity, for example, might include Timestamp, Application, User, Session ID .etc.

Using logs to monitor the use of the system provides security to check for violations and unpredicted behaviour from users and the system. Analysing, searching and fltering logs discovers and alerts these incidents. It enables access to historic information for debugging and forensic investigations.

* + 1. **Exaｍples**

The example which will show the use of logs is Amazon CloudＷatch Logs. Amazon CloudＷatch Logs can be used to monitor, store and access the log fles from AＷS Services. AＷS Identity and Access Ｍanagement (IAＭ) and AＷS Lambda used in conjunction with CloudＷatch Logs.

CloudＷatch Logs enables the centralization of the logs from all over the systems, applications and AＷS services that are used, in a single, highly scalable service. You can then easily view them, search them for specifc error codes or patterns, flter them based on specifc felds, or archive them securely for future analysis.

**Features**

Query the Log Data - CloudＷatch Logs Insights can be used to interactively search and analyse log data. CloudＷatch provides sample queries, command descriptions, query autocompletion and log feld discovery to help get started.

Ｍonitor AＷS CloudTrail Logged Events - You can create alarms in CloudＷatch and receive noti- fcations of particular API activity as captured by CloudTrail and use the notifcation to perform troubleshooting.

Log Retention - By default, logs are kept indefnitely and never expire. You can adjust the retention policy for each log group, keeping the indefnite retention, or choosing a retention period between 10 years and one day.

Archive Log Data - CloudＷatch Logs can be used to store log data in highly durable storage. The CloudＷatch Logs agent makes it easy to quickly send both rotated and non-rotated log data of of a host and into the log service. You can then access the raw log data when needed.

# **Developｍent Testing**

## **Local environｍent**

* + 1. **API developｍent**

**Serverless** The Serverless Framework enables developers to run an emulated environment locally instead of needing to deploy that serverless app for testing. This comes with over 1,000 plugins that can emulate services such as AＷS Lambda and API Gateway.

**Postｍan** Postman is a very popular tool used by developers that want to conveniently test and build APIs in a collaborative way.

As we are implementing the OpenAPI specifcation, Postman enables the import of Swagger fles to automatically generate a Postman environment. This makes developer on-boarding much easier as versioned mock endpoints and documentation are hosted for us as part of the DevOps deployment.

## **Dev0ps pipeline**

* + 1. **GitＨub**
    2. **CircleCI**
    3. **CodeDeploy**

# **Release Testing**

## **Staging**

* 1. **Production**

# **00 Design & UＭL**

**References**

"1] Ron Ｍiller. *AＷS Lambda Ｍakes Serverless Applications A Reality*. TechCrunch. 24. uRL :

[https://techcrunch .com/2015/ 11 /24/aws-lamda-makes- seryerless-applications-a-](https://techcrunch.com/2015/11/24/aws-lamda-makes-serverless-applications-a-reality/)

[reality/](https://techcrunch.com/2015/11/24/aws-lamda-makes-serverless-applications-a-reality/).

"2] *Overview of Open Standards*. Free Software Foundation Europe. uRL : [https://fsfe .org/](https://fsfe.org/freesoftware/standards/def.en.html)

[freesoftware/standards/def .en .html](https://fsfe.org/freesoftware/standards/def.en.html) .

"3] *OpenAPI Speci五cation*. Version 3.0.3. SmartBear Software. uRL : [https : / / swagger .io /](https://swagger.io/specification/)

[specification/](https://swagger.io/specification/).

"4] *Fast Healthcare Interoperability Resources*. Ｗikipedia Foundation. Nov. 2016. uRL : [https:](https://en.wikipedia.org/wiki/Fast_Healthcare_Interoperability_Resources)

[//en .wikipedia .org/wiki/Fast-Healthcare-Interoperability-Resources](https://en.wikipedia.org/wiki/Fast_Healthcare_Interoperability_Resources).