Tamper Proof Quality Data Using Amazon QLDB

Implementation Guide

Tamper Proof Quality Data Using Amazon QLDB : Implementation Guide

Table of Contents

Deploy an Amazon QLDB-based solution to prevent data tampering in your factory

Publication date: *July 2021*

Data tampering is costly for manufacturing companies. Quality data, such as the results of quality tests or the temperature of factory devices, is especially vulnerable. This solution prevents attackers from tampering with quality data by using Amazon Quantum Ledger Database (Amazon QLDB) to maintain an accurate history of data changes. Amazon QLDB implements cryptographic hashing for all documents and revisions of the documents written to the database.

This solution provides three options to register and input the quality data to Amazon QLDB.

1. Exposing REST API endpoints with Amazon API Gateway.
2. Uploading the files to Amazon Simple Storage Service (Amazon S3).
3. Transporting the quality data by MQTT protocol with AWS IoT Core.

This implementation guide describes architectural considerations and configuration steps for deploying the Tamper Proof Quality Data Using Amazon QLDB on the Amazon Web Services (AWS) Cloud. It includes a link to an AWS CloudFormation template and instructions to launch, configure, and run the AWS services required to deploy this solution using AWS best practices for security and availability.

The guide is intended for IT infrastructure architects, administrators, and DevOps professionals who have practical experience architecting on the AWS Cloud.

Cost

You are responsible for the cost of the AWS services used while running this solution. As of July 2021, the cost for running this solution with the default settings in the US East (N.Virginia) is **$128.58 a month** (excluding free tier).

This cost estimate is based on a factory with equipment that does the following:

1. Sends metrics, such as temperature or vibration strength, every second to an Amazon API Gateway endpoint.
2. Captures and sends product image (1 megabyte per image) every second to Amazon S3.
3. Sends inspection results of each individual product every second via AWS IoT Core.

> **Note**
> For additional details about equipment, refer to the Solution components (p. 4) section.

AWS service	Dimensions	Monthly cost
AWS IoT Core	1 message per second published	$ 2.68
Amazon API Gateway	1 request per second received	$ 9.37
Amazon S3	Put 1MB image per second	$75.00
AWS Lambda	Invocations by above three services Average 200ms duration with 128 MB RAM	$ 4.98
Amazon QLDB	Write input/output from Lambda functions Add average 0.5 KB data per write query	$ 6.75
Amazon VPC	S3 and API Gateway VPC Endpoint across 2 availability zones	$29.80
	Total:	**$128.58**

We recommend creating a budget through AWS Cost Explorer to help manage costs. Prices are subject to change. For full details, see the pricing webpage for each AWS service used in this solution.

Architecture overview

Deploying this solution with the default parameters builds the following environment in the AWS Cloud.

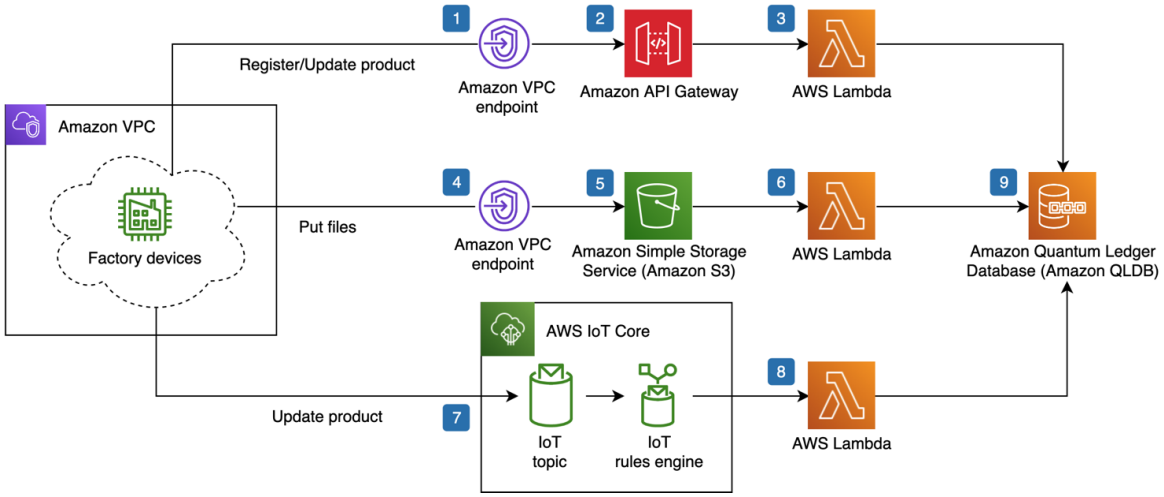

Figure 1: Tamper Proof Quality Data Using Amazon QLDB architecture on AWS

The AWS CloudFormation template deploys the following infrastructure:

1. Amazon Virtual Private Cloud (Amazon VPC) to provide the endpoint for Amazon API Gateway.
2. Amazon API Gateway to provide an API to register or update the product information.
3. An AWS Lambda function to register and update the product information via Amazon API Gateway.
4. Amazon VPC to provide the endpoint for Amazon S3.
5. Amazon S3 to store the product quality data such as images or videos.
6. A Lambda function hooks `s3:ObjectCreated:Put` events to store the metadata of the files.
7. AWS IoT Core connect to the devices to update the product information.
8. A Lambda function is invoked by AWS IoT Core that also allows you to update the production information.
9. Amazon QLDB to maintain an accurate history of data changes.

> **Note**
> The factory devices described in the figure are not deployed by the template. CloudFormation resources are created from AWS Cloud Development Kit (CDK) constructs.

Solution components

This solution provides three input options for registering quality data into Amazon QLDB. These options are designed to demonstrate how you can interact with Amazon QLDB during the product manufacturing process. In a production environment, we recommend customizing these options for your use case.

Important
Before registering the quality data with any of the three input options, you must first register your product information with API Gateway.

Option 1: Amazon API Gateway

APIs are used to register the product serial number (a unique ID for each product) and store aggregated metrics sent from factory equipment, such as temperature or vibration strength.

Option 2: Amazon S3

Amazon S3 is used to store related files of manufactured products, for example, captured images. The checksum of those files are stored in Amazon QLDB for accuracy verification. Also, S3 versioning is turned on to allow tracking of any overwrites of the files.

Option 3: AWS IoT Core

AWS IoT Core is connected to the IoT devices in factories and is used to receive quality data. The quality data is received from the message queue of AWS IoT Core and stored in Amazon QLDB through an AWS Lambda function. In this solution, AWS IoT Core is used to register the results of product inspections.

Security

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This shared model reduces your operational burden because AWS operates, manages, and controls the components including the host operating system, the virtualization layer, and the physical security of the facilities in which the services operate. For more information about AWS security, visit AWS Cloud Security.

IAM policies

AWS Identity and Access Management (IAM) roles allow customers to assign granular access policies and permissions to services and users on the AWS Cloud. This solution creates IAM roles that grant the solution's Lambda functions access to get objects from the solution's Amazon S3 bucket and access to send data to Amazon QLDB. To dispatch S3 event notifications to a Lambda function, its IAM resource policies is configured to accept access from S3. IAM is also used for authentication and authorization of API Gateway, where you must provide a signed HTTP header to access the API. Similarly, the topics of AWS IoT Core requires IAM authentication to publish messages. We recommend that you set the least privileges for the client-side IAM roles or users when you access API Gateway or AWS IoT Core topics in this solution.

Security groups

The security groups created in this solution are designed to control network traffic between the Amazon VPC containing factory devices and Amazon API Gateway. Once the deployment is up and running, we recommend reviewing the security groups and further restrict access as needed.

VPC endpoint policies

This solution uses VPC endpoints for additional security. It ensures the traffic between AWS services, such as Lambda, Amazon S3, and Amazon QLDB, does not leave the Amazon network. VPC endpoint policies allows you to restrict what actions one can perform via the corresponding endpoints. Although the VPC endpoint policies of this solution have already been configured to have the least privileges, you can review and modify them when you need broader access permissions.

Design considerations

Regional deployments

This solution uses the Amazon QLDB service, which is not currently available in all AWS Regions. You must launch this solution in an AWS Region where Amazon QLDB is available. For the most current availability by Region, refer to the AWS Regional Services List.

AWS CloudFormation template

To automate deployment, this solution uses AWS CloudFormation. It includes the following CloudFormation template, which you can download before deployment:

tamper-proof-quality-data-using-amazon-qldb.template: Use this template to launch the solution and all associated components. The default configuration deploys AWS IoT Core, Amazon QLDB, AWS Lambda, Amazon API Gateway and Amazon VPC.

Note
AWS CloudFormation resources are created from AWS Cloud Development Kit (CDK) constructs.

Automated deployment

Before you launch this solution, review the architecture, network security, and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy this solution into your account.

Time to deploy: Approximately 10 minutes.

Deployment overview

Use the following steps to deploy this solution on AWS. For detailed instructions, follow the links for each step.

Step 1. Launch the stack (p. 8)

- Launch the AWS CloudFormation template into your AWS account.

Step 2. Create Amazon QLDB table and index (p. 9)

- Use the Amazon QLDB console to create a ledger table and its index.

Step 3. Launch an Amazon EC2 instance (p. 10)

- Create an EC2 instance to access the API Gateway endpoint in Amazon VPC.
- Access the EC2 instance via Session Manager.

Step 1. Launch the stack

> **Important**
> This solution includes an option to send anonymous operational metrics to AWS. We use this data to better understand how customers use this solution and related services and products. AWS owns the data gathered though this survey. Data collection is subject to the AWS Privacy Policy.
> To opt out of this feature, download the template, modify the AWS CloudFormation mapping section, and then use the AWS CloudFormation console to upload your template and deploy the solution. For more information, refer to the Collection of operational metrics (p. 20) section of this guide.

This automated AWS CloudFormation template deploys Tamper Proof Quality Data Using Amazon QLDB in the AWS Cloud.

> **Note**
> You are responsible for the cost of the AWS services used while running this solution. For more details, visit to the Cost (p. 2) section in this guide, and refer to the pricing webpage for each AWS service used in this solution.

1. Sign in to the AWS Management Console and select the button to launch the `tamper-proof-quality-data-using-amazon-qldb` AWS CloudFormation template.

Alternatively, you can download the template as a starting point for your own implementation.

2. The template launches in the US East (N. Virginia) Region by default. To launch the solution in a different AWS Region, use the Region selector in the console navigation bar.

   Note
   This solution uses the Amazon QLDB service, which is not currently available in all AWS Regions. You must launch this solution in an AWS Region where Amazon QLDB is available. For the most current availability by Region, refer to the AWS Regional Services List.

3. On the **Create stack** page, verify that the correct template URL is in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your solution stack. For information about naming character limitations, see IAM and STS Limits in the *AWS Identity and Access Management User Guide*.
5. Choose **Next**.
6. On the **Configure stack options** page, choose **Next**.
7. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
8. Choose **Create stack** to deploy the stack.

You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a CREATE_COMPLETE status in approximately 10 minutes.

Note
In addition to the primary `ReflowOvenLambda`, `InspectionCameraLambda`, and `InspectionModelLambda` AWS Lambda functions, this solution includes the `solution-helper` Lambda function, which runs only during initial configuration or when resources are updated or deleted.
When you run this solution, you will notice multiple Lambda functions exist in the AWS console. Even if not regularly active, you must not delete the `solution-helper` Lambda function, because it is necessary to manage associated resources.

Step 2. Create Amazon QLDB table and index

To store your data, you must create a ledger table in Amazon QLDB.

1. Sign in to the Amazon QLDB console.
2. In the navigation pane, choose **Query editor**.
3. Under **Ledger**:
   a. Choose **quality-ledger**.
   b. In the box, enter the following query to create a table:

```
CREATE TABLE QualityData
```

   c. Choose **Run**.
4. Verify a value appears on the **Result** tab under **tableId**.

5. In the box, enter the following query to create an index table for index lookup:

```
CREATE INDEX on QualityData (serialNumber)
```

6. Choose **Run**. A success message appears to confirm that you have created the index.

Step 3. Launch an EC2 instance

Create an Amazon EC2 instance to access the API Gateway endpoint in Amazon VPC.

1. Sign in to the Amazon EC2 console.
2. On the **Instances** page, choose **Launch Instances**.
3. Select the **Amazon Linux 2 AMI (HVM), SSD Volume Type** instance.
4. On the **Choose an Instance Type** page, select **t2.micro**, and then choose **Review and Launch**.
5. Select **Edit instance details**, and chose the following configuration:
   a. **Network**: The VPC ID created by the solution.
   b. **Subnet**: `tamper-proof-quality-data-using-amazon-qldb/VPC/VPC/PrivateSubnet1`.
   c. **IAM Role**: Select the IAM Role with `tamper-proof-quality-data-using-amazon-qldb-Role` prefix.

      Note
      The `tamper-proof-quality-data-using-amazon-qldb/VPC/VPC/PrivateSubnet1` subnet and IAM role for the instance are created during the AWS CloudFormation deployment.
6. Choose **Review and Launch**.
7. Review your Instance launch and choose **Launch**.
8. In the **Select an existing key pair or create a new key pair** box, choose **Proceed without a key pair** and check the box to acknowledge that I will not be able to connect to this instance unless you use EC2 Instance Connect or know the password built into the AMI.
9. Choose **Launch Instances**.

   It takes a few minutes to launch the instance.

Access the EC2 Instance via Session Manager

1. Sign in to the Session Manager console.
2. Choose **Start Session**.
3. Select the target instance, and choose **Start Session**.

   Note
   It takes a few minutes to show instances on Session Manager console after launching the EC2 instance.

Register your product information and update your quality data

This solution provides three options to update the quality data of a product. However, first you must use API Gateway to register your product information, such as the product's serial number, in Amazon QLDB.

Important
You must complete the following steps while logged into EC2 via Session Manager.

1. Sign in to the Session Manager console.
2. Choose **Start Session**.
3. Select the target instance, and choose **Start Session**.

Set up $PATH and install jq

We use jq to parse the command results. Run the following commands to install jq in the EC2 instance:

1. Change user:

```
sudo -iu ssm-user
```

2. Add /usr/local/bin to $PATH to allow installation of command line tools:

```
export PATH=/usr/local/bin:$PATH
```

3. Download the jq binary package:

```
wget -O jq https://github.com/stedolan/jq/releases/download/jq-1.6/jq-linux64
```

4. Grant permissions:

```
chmod +x ./jq
```

5. Move jq to the correct path:

```
sudo mv jq /usr/local/bin
```

6. Confirm the installation:

```
jq --help
```

Register product information in Amazon QLDB

We use API Gateway to create a private API. To access the API, you must be inside the Amazon VPC. For more information, refer to How to invoke a private API in the *Amazon API Gateway Developer Guide*. To

access the private API, we deployed an EC2 instance inside the Amazon VPC. For details, refer to Step 3. Launch an EC2 instance (p. 10).

This solution uses Signature Version 4 (SigV4) for prominent API authentication.

Requests to the API Gateway are authorized with IAM. We use the `TamperProofQualityDataClientRole` IAM role assigned to EC2.

Install awscurl

To make an HTTP request to the API Gateway, we use the awscurl OSS command line interface. Run the following commands to install awscurl with pip:

1. Update pip to latest version:

```
sudo pip3 install -U pip
```

2. Install awscurl:

```
pip3 install awscurl
```

   You will use the assumed role for the EC2 instance. awscurl uses the credential to securely issue requests to the API Gateway.

3. Retrieve the credential values:

```
export ROLEARN=`aws iam get-role --role-name "TamperProofQualityDataClientRole" | jq -
r .Role.Arn`
export ASSUMEROLE=`aws sts assume-role --role-arn $ROLEARN --role-session-name
 DeviceClient --duration-second 3600`
export AWS_ACCESS_KEY_ID=`echo $ASSUMEROLE | jq -r .Credentials.AccessKeyId`
export AWS_SECRET_ACCESS_KEY=`echo $ASSUMEROLE | jq -r .Credentials.SecretAccessKey`
export AWS_SESSION_TOKEN=`echo $ASSUMEROLE | jq -r .Credentials.SessionToken`
```

   Note
   `TamperProofQualityDataClientRole` is the role name of the EC2 instance. `DeviceClient` is a session name that is allowed to be assumed to the instance role.

Register your product with API Gateway

1. Set up the environment variables:

```
export AWS_DEFAULT_REGION=<Region>
export API_ENDPOINT=https://<API ID>.execute-api.<Region>.amazonaws.com/prod/product
```

   Replace *<Region>* and *<API ID>* with your specific details.

2. Register the product:

```
awscurl --access_key $AWS_ACCESS_KEY_ID --secret_key $AWS_SECRET_ACCESS_KEY --
session_token $AWS_SESSION_TOKEN -H 'Content-Type: application/json' -XPOST $API_ENDPOINT
 --data '{ "serialNumber": "serial1", "factoryId": "factory1", "lineId": "line1" }'
```

   The API returns the document ID to confirm success:

```
[
```

```
  {
    "documentId": "<Document ID>"
  }
]
```

Update your quality data

After registering the serial number with API Gateway, there are three options to update the product information.

1. Use API Gateway to register values such as temperature and vibration of the product.
2. Store the product quality data in Amazon S3 and register the metadata in Amazon QLDB.
3. Register the results of the product quality test with AWS IoT Core.

Option 1: Update the quality data in Amazon QLDB with API Gateway

Proceed with this procedure in your open Session Manager session. If your session is not open, refer to Access the EC2 Instance via Session Manager (p. 10) to open a new session.

1. In Session Manager, enter the following command to update a product:

```
awscurl --access_key $AWS_ACCESS_KEY_ID --secret_key $AWS_SECRET_ACCESS_KEY --
session_token $AWS_SESSION_TOKEN -H 'Content-Type: application/json' -XPUT $API_ENDPOINT/
serial1 --data '{ "temperature": 1.9, "voltage": 1.2, "vibration": 1.8 }'
```

2. The API returns the document ID to confirm success:

```
[
  {
    "documentId": "<Document ID>"
  }
]
```

The `documentId` is used to query the change history on Amazon QLDB.

Check your result

To confirm that your product is updated in Amazon QLDB, use the following query:

```
SELECT * FROM QualityData WHERE serialNumber = 'serial1'
```

Note
The above query works efficiently because `serialNumber` is a high cardinality index.

Check the update in Amazon QLDB

1. Sign in to the Amazon QLDB console.
2. In the navigation pane, choose **Query editor**.
3. Under **Ledger**:

a. Choose **quality-ledger**.

b. In the **Query editor** box, enter the following query to check the update:

```
SELECT * FROM history(QualityData) AS h where h.metadata.id = '<document-ID>';
```

c. Choose **Run**.

4. Review the history of the data changes.

Option 2: Update quality data in Amazon QLDB with AWS Lambda and Amazon S3

This option uses AWS Lambda and Amazon S3 to update the quality data in QLDB. When you upload a file to the S3 bucket, an S3 event notification invokes a Lambda function which records the information in Amazon QLDB.

Proceed with this procedure in your open Session Manager session. If your session is not open, refer to Access the EC2 Instance via Session Manager (p. 10) to open a new session.

1. In Session Manager, enter the following command to set up environment variables:

```
export BUCKET_NAME=<Your-bucket-name>
```

   Replace *<Your-bucket-name>* with the S3 bucket that uses the following prefix: `qldb-inspectioncamerainspectioncamerabucket`.

2. Create a 1 MB dummy file to be uploaded to the S3 bucket:

```
dd bs=1024 count=1024 </dev/urandom > serial1
```

3. Upload the dummy file to S3:

```
aws s3 cp serial1 s3://$BUCKET_NAME/serial1 --endpoint-url=https://bucket.<vpce-xxxxxxxx-xxxxxxx>.s3.<Region>.vpce.amazonaws.com
```

   <vpce-xxxxxxxx-xxxxxxx> is not the same as the VPC endpoint ID.

4. Confirm the endpoint in the Amazon VPC console.

   If successful, you will receive the following response:

```
upload: ./serial1 to s3://BUCKET_NAME/serial1
```

After issuing the commands to invoke the Lambda function, you can run a QLDB query to find the data that you stored.

Find the data stored in Amazon QLDB

1. Sign in to the Amazon QLDB console.
2. In the navigation pane, choose **Query editor**.
3. Under **Ledger**:
   a. Choose **quality-ledger**.
   b. In the **Query editor** box, enter the following query to confirm the result:

```
SELECT data.inspectionCamera FROM QualityData WHERE serialNumber = 'serial1';
```

c. Choose **Run**.

If successful, you will receive the following response:

```
{
url: "s3://[BUCKET_NAME]/serial1",
hash: "<new hash value>"
}
```

When the contents of the file are changed and uploaded, the data will be updated with a new hash value.

Check update history

To confirm the update history of the file, use the following query in Amazon QLDB:

```
SELECT data.data.inspectionCamera FROM history(QualityData) WHERE data.serialNumber =
 'serial1';
```

Check the update in Amazon QLDB

To check the update history in Amazon QLDB, run the following query in Amazon QLDB:

```
SELECT * FROM history(QualityData) AS h where h.metadata.id = '<document-ID>';
```

Option 3: Update quality data in Amazon QLDB with Lambda and AWS IoT Core

This option uses Lambda and AWS IoT Core to update the quality data in QLDB. The AWS IoT Core publish event invokes a Lambda function which records the information in Amazon QLDB. To publish data to AWS IoT, you must use AWS CLI version 2. For details, refer to Update the AWS CLI version 2 on Linux in the *AWS Command Line Interface User Guide*.

1. Download the AWS Command Line Interface (AWS CLI) zip file:

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
```

2. Unzip the downloaded file:

```
unzip awscliv2.zip
```

3. Install AWS CLI version 2:

```
sudo ./aws/install --bin-dir /usr/local/bin --install-dir /usr/local/aws-cli --update
```

4. Remove the downloaded files:

```
rm -rf ./aws
```

Publish data to AWS IoT

To publish data to AWS IoT, run the following command:

```
/usr/local/bin/aws iot-data publish --cli-binary-format raw-in-base64-out --topic
 'iot/inspectionmodel' --payload '{ "serialNumber": "serial1", "data": {"isGood": true,
 "accuracy": 0.8, "modelRevision": "model"} }'
```

You will not receive any status messages if the command runs successfully.

Check history in Amazon QLDB

After publishing through AWS IoT Core and Lambda, you can run a QLDB query to find the data you stored.

1. Sign in to the Amazon QLDB console.
2. In the navigation pane, choose **Query editor**.
3. Under **Ledger**:
   a. Choose **quality-ledger**.
   b. In the **Query editor** box, enter the following query:

   ```
   SELECT * FROM QualityData WHERE serialNumber = 'serial1';
   ```

   c. Choose **Run**.
4. Review the quality data on the **Results** tab.

Check the update in Amazon QLDB

To check the update history in Amazon QLDB, run the following query:

```
SELECT * FROM history(QualityData) AS h where h.metadata.id = '<document-ID>';
```

Verify the data in Amazon QLDB

You can efficiently verify the integrity of a document in your ledger's journal by using cryptographic hashing with SHA-256.

Request a digest

1. Sign in to the Amazon QLDB console.
2. In the navigation pane, choose **Ledgers**.
3. Under **Ledgers**, select **quality-ledger** and choose **Get digest**. The **Get digest** dialog box displays the following digest details:
   - **Digest**: The SHA-256 hash value of the digest that you requested.
   - **Digest tip address**: The latest block location in the journal covered by the digest that you requested. An address has the following two fields:
     - `strandId`: The unique ID of the journal strand that contains the block.
     - `sequenceNo`: The index number that specifies the location of the block within the strand.
   - **Ledger**: The ledger name for which you requested a digest.
   - **Date**: The timestamp when you requested the digest.

4. Review the digest information. Then choose **Save**. You can keep the default file name, or enter a new name. This step saves a plaintext file with contents in Amazon Ion format. The file has a `.ion.txt` file name extension and contains all of the digest information that was listed on the preceding dialog box.

   The following code snippet displays an example of a digest file. The order of the fields can vary depending on your browser.

```
{"digest":"xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx","digestTipAddress":"{strandId:"yyyyyyyyyyyy
ledger","date":"yyyy-mm-ddTxx:xx:xx.xxxZ"}
```

5. Save this file where you can access it later. Use this file to Verify a document revision (p. 17).

Verify a document revision

Query your ledger for the `id` and `blockAddress` of the document revision that you want to verify. These fields are included in the document's metadata, which you can query in the committed view. The `id` is a system-assigned unique ID string. The `blockAddress` is an Ion structure that specifies the block location where the revision was committed.

1. In the navigation pane of the QLDB console, choose **Query editor**.
2. Under **Ledger**:
   a. Choose **quality-ledger**.
   b. In the **Query editor** box, enter the following query:

```
SELECT r.metadata.id, r.blockAddress FROM _ql_committed_QualityData AS r WHERE
 r.data.serialNumber = 'serial1'
```

   c. Choose **Run**.
3. Copy and save the `id` and `blockAddress` values that your query returns. Omit the double quotes for the `id` field. In Amazon Ion, string data types are delimited with double quotes. Only copy the alphanumeric text.

   Now that you have a document revision selected, you can start the process of verifying it.
4. In the Amazon QLDB navigation pane, choose **Verification**.
5. On the **Verify document** form:
   a. Under **Specify the document that you want to verify**, enter the following input parameters:
      - **Ledger**: The ledger in which you want to verify a revision.
      - **Document ID**: The `id` value returned by the query.
      - **Block address**: The `blockAddress` value returned by the query.
   b. Under **Specify the digest to use for verification**, select **Choose digest** and select the digest that you previously saved. If the file is valid, this auto-populates all the digest fields on your console. Alternatively, you can manually copy and paste the following values directly from your digest file:
      - **Digest**: The `digest` value from your digest file.
      - **Digest tip address**: The `digestTipAddress` value from your digest file.
   c. Review your document and digest input parameters, and then choose **Verify**.

   The console displays the results of your request in the **Verification results** card. If successfully verified, this means that the recalculated digest is the same as what you specified. It proves the following.
   - The document exists at the same location in your journal where it was first written.
   - The document hasn't been altered in any way since it was written.

   For more details about data verification, refer to Data verification in Amazon QLDB.

Additional resources

AWS services:

- AWS CloudFormation
- AWS IoT Core
- AWS Lambda
- Amazon API Gateway
- Amazon QLDB

Uninstall the solution

You can uninstall the Tamper Proof Quality Date using Amazon QLDB solution from the AWS Management Console or by using the AWS Command Line Interface (AWS CLI)

Using the AWS Management Console

1. Sign in to the AWS CloudFormation console.
2. Select this solution's installation stack.
3. Choose **Delete**.

Using AWS Command Line Interface

Determine whether the AWS Command Line Interface (AWS CLI) is available in your environment. For installation instructions, refer to What Is the AWS Command Line Interface in the *AWS CLI User Guide*. After confirming that the AWS CLI is available, run the following command.

```
aws cloudformation delete-stack --stack-name <installation-stack-name> --region <your-Region-name>
```

Collection of operational metrics

This solution includes an option to send anonymous operational metrics to AWS. We use this data to better understand how customers use this solution and related services and products. When invoked, the following information is collected and sent to AWS:

- **Solution ID:** The AWS solution identifier
- **Solution version:** The solution version

AWS owns the data gathered though this survey. Data collection is subject to the AWS Privacy Policy. To opt out of this feature, complete the following steps before launching the AWS CloudFormation template.

1. Download the AWS CloudFormation template to your local hard drive.
2. Open the AWS CloudFormation template with a text editor.
3. Modify the AWS CloudFormation template mapping section from:

```
AnonymousData:
    SendAnonymousData:
      Data: Yes
```

   to:

```
AnonymousData:
    SendAnonymousData:
      Data: No
```

4. Sign in to the AWS CloudFormation console.
5. Select **Create stack**.
6. On the **Create stack** page, **Specify template** section, select **Upload a template file**.
7. Under **Upload a template file**, choose **Choose file** and select the edited template from your local drive.
8. Choose **Next** and follow the steps in Launch the stack (p. 8) in the Automated deployment section of this guide.

Source code

Visit our GitHub repository for the solution to download the source files for this solution and to share your customizations with others. The Tamper Proof Quality Data Using Amazon QLDB templates are generated using the AWS Cloud Development Kit (CDK). Refer to the README.md file for additional information.

Revisions

Date	Change
July 2021	Initial release

Contributors

- Taichiro Suzuki
- Daiki Kuriyama
- Masashi Tomooka
- Satoshi Suzuki
- Takuya Mizuma
- Yukinobu Mine
- Kei Toda
- Yunhe Wang

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents AWS current product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. AWS responsibilities and liabilities to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.