

[AWS Storage Blog](#)

A look inside how global multimedia agency Reuters uses Amazon Web Services

by Romeo Radanyi | on 11 MAY 2020 | in [Amazon CloudFront](#), [Amazon Route 53](#), [Amazon Simple Storage Service \(S3\)](#), [Customer Solutions](#), [Intermediate \(200\)](#), [Networking & Content Delivery](#), [Storage](#) | [Permalink](#) | [Comments](#) | [Share](#)

My name is Romeo Radanyi and I am a Solutions Architect at Thomson Reuters, where I help teams understand and adopt cloud technologies. I'm involved in setting enterprise-level standards, pushing forward our Artificial Intelligence and Machine Learning plans, and building many Reuters News Agency systems. I have also created and delivered a course at [A Cloud Guru](#) for absolute beginners about Data Science and Jupyter notebooks.

Reuters is the world's largest multimedia news source for the news networks you watch, read, or listen to. We are a wholesaler – meaning we sell raw news in various formats to customers such as the BBC, CNN, The New York Times, The Washington Post, and many other customers around the globe.

In this blog post, I explore how Reuters migrated most of its content onto AWS. I discuss how we use AWS's turnkey object storage solution, [Amazon Simple Storage Service \(Amazon S3\)](#), and its content distribution network (CDN), [Amazon CloudFront](#), to reach billions of people around the globe every day.

Challenges we faced with legacy systems

Five years ago, we chose AWS as our strategic platform for all newly created Reuters systems. You might be asking yourself: "Why?"

Well, Reuters creates and distributes content in a variety of formats: videos, pictures, text, and even live streams during breaking news events. In addition, we had an archive going back to 1896 on physical tapes and a global footprint of more than 200 physical news bureaus, which are effectively mini data centers. Those bureaus are the main source of our edited content after the raw content is shot in the field. Storing, transforming, and serving all these content types around the globe at a moment's notice, in various formats and frame rates, is no easy task!

Initially, we relied on third parties to manage some of these tasks, including storing, transforming, and serving content, buying time for our teams to learn and integrate into our processes. As part of collaborating with a third party, we encountered challenges when troubleshooting outages or service-related issues, and it took time to implement processes to improve them. Even with extensive support from the Reuters team, costs were adding up. With our extensive archive, tapes are cheap and easy to use, but it's much harder to distribute and sell the content to news outlets around the world, as the retrieval is very slow. In the case of our physical bureaus, imagine all of those mini data centers as islands – collaboration is difficult without an intermediary scalable space to store content for editing.

Hence, Reuters was facing three main challenges. First, using a third party to store all of our media assets and using an expensive CDN wasn't the way forward for us anymore. Secondly, with our extensive collection of archived tapes, it was difficult to distribute information quickly. Thirdly, with our physical bureaus, it was complex to collaborate across teams due to lack of interteam connectivity across all of the island-like data centers.

Choosing AWS

As we gained more familiarity with AWS and Amazon S3, we found the wide range of features and ease of usability via its API and HTTPS calls to be the perfect solution to our organization-wide challenges. In addition, we realized the added benefits of AWS and Amazon S3's cost and durability. Remember though – this was in 2015, when the majority of content ingestion to CDN providers was still via File Transfer Protocol (FTP).

At first, we were cautious, like everyone who just starts their cloud journey. There were a number of things we had to pay attention to as a leading global news organization:

1. Security
2. Global coverage
3. Latency and speed of delivery
4. Availability
5. Resiliency
6. Scalability

Initially, we did various test downloads against Amazon CloudFront to prove its speed of delivery and coverage. Reuters has a large footprint of globally distributed physical test clients because one of our main revenue sources is still in the broadcast business. We offer hardware devices and servers for our customers to support their traditional broadcast workflows, allowing a direct feed of news content into their editorial workflow using Serial Digital Interface. The content is delivered via satellites to these servers with an internet delivery-based backup. However, as cloud computing and internet coverage is increasing, our satellite requirements and usage is decreasing. It was ridiculously easy to set up an Amazon S3 test bucket and CloudFront distribution for this, and the results spoke for themselves. Using S3 and CloudFront, our capabilities became cheaper, highly available, secure, and scalable. Additionally, our services were also performing quicker because of the many edge locations AWS has globally (205 Edge Locations and 11 Regional Edge Caches to be exact).

Due to our initial wins, we decided to migrate all of our core news content to Amazon S3 and move away from our CDN provider, which was expensive in comparison to Amazon CloudFront. Additionally, breaking news – and news in general – has diminishing value as time passes. For this reason, we categorize content based on whether it was created within 30 days, after which the content should become part of our archive. This categorization and archiving was easy after we came up with a prefix strategy on Amazon S3, based on our existing content structure and our requirements around content retention periods. We simply use S3 Lifecycle rules to “expire” our content after 30 days, saving us money on storage costs.

Uploading objects using Multipart Upload API

As we deliver large video files (from 100 MB to ~2 GB), we had to ensure we had resilient retry logic built in to smooth out potential errors in transmission. To accomplish this, we rewrote our applications, which were hosted on-premises to use the AWS SDK Multipart Upload API. We also use the Amazon S3 multipart upload API to achieve the best performance. This is because multiple concurrent Transmission Control Protocol (TCP) connections can potentially make better use of your available bandwidth. This is the result of how TCP window sizes allocate

bandwidth. The Multipart Upload API enables you to upload large objects in parts, a perfect capability when we are frequently sending such large video files.

Uploading videos via the Multipart Upload API is a simple 3-step process:

1. Initiate the upload.
2. Upload the object parts.
3. After uploading all the parts, complete the multipart upload.

Once S3 receives the complete multipart upload request, it constructs the entire object from the uploaded parts. You can then access the object just as you would any other object in your bucket.

When using multipart upload API, make sure to create lifecycle rules to delete incomplete multipart uploads that haven't finished within the specified number of days after being initiated. This is an important step to save on storage cost. We use the following rule on nearly all of our buckets where we use multipart upload API:

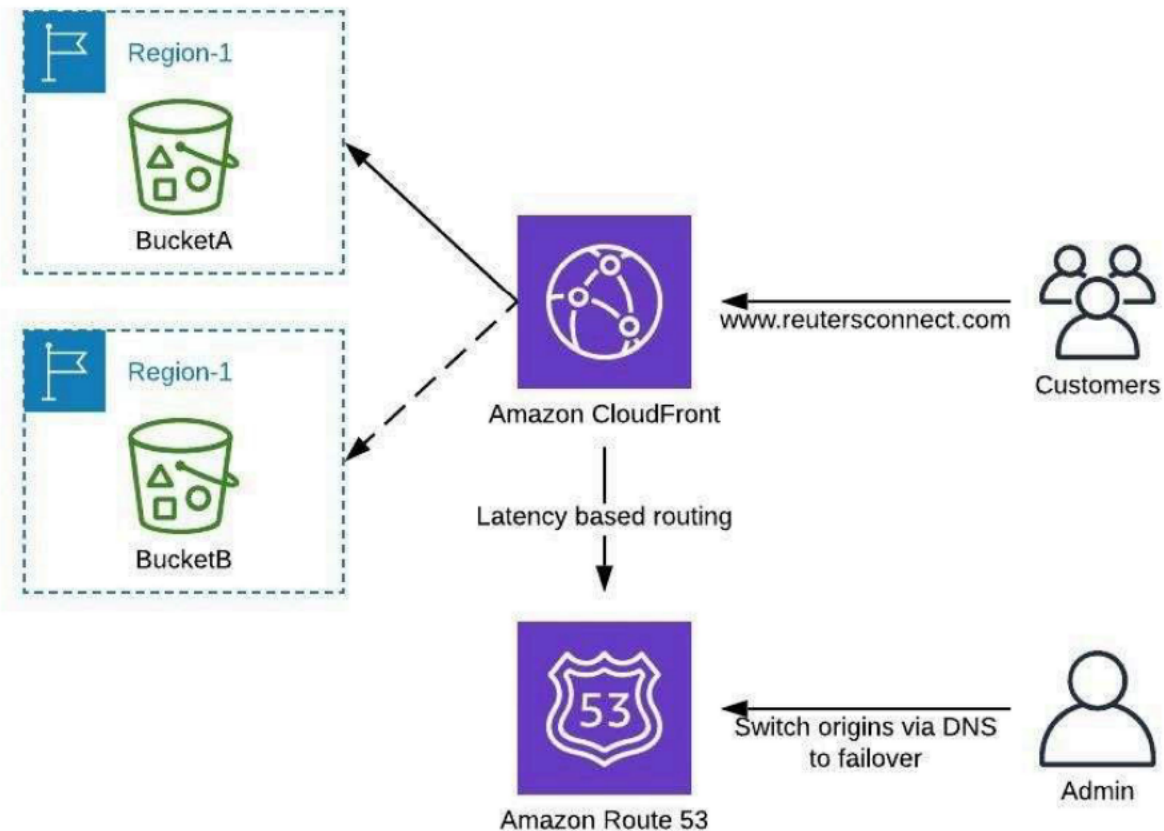
XML

```
<LifecycleConfiguration>
  <Rule>
    <ID>multipart-cleanup-rule</ID>
    <Prefix/>
    <Status>Enabled</Status>
    <AbortIncompleteMultipartUpload>
      <DaysAfterInitiation>7</DaysAfterInitiation>
    </AbortIncompleteMultipartUpload>
  </Rule>
</LifecycleConfiguration>
```

Having a well-defined prefix strategy and a set of lifecycle rules to keep our bucket clean, is important. It is also important to ensure reliability with a good multi-region strategy to protect our most critical systems. This is crucial to making sure that our customers can access unbiased breaking news at all times – constant uptime is vital for our business.

Reuters's modern global data architecture

The following architecture diagram conceptually shows how we use [Amazon Route 53](#), CloudFront, and Amazon S3 to protect against S3 API-related problems or regional issues with S3:

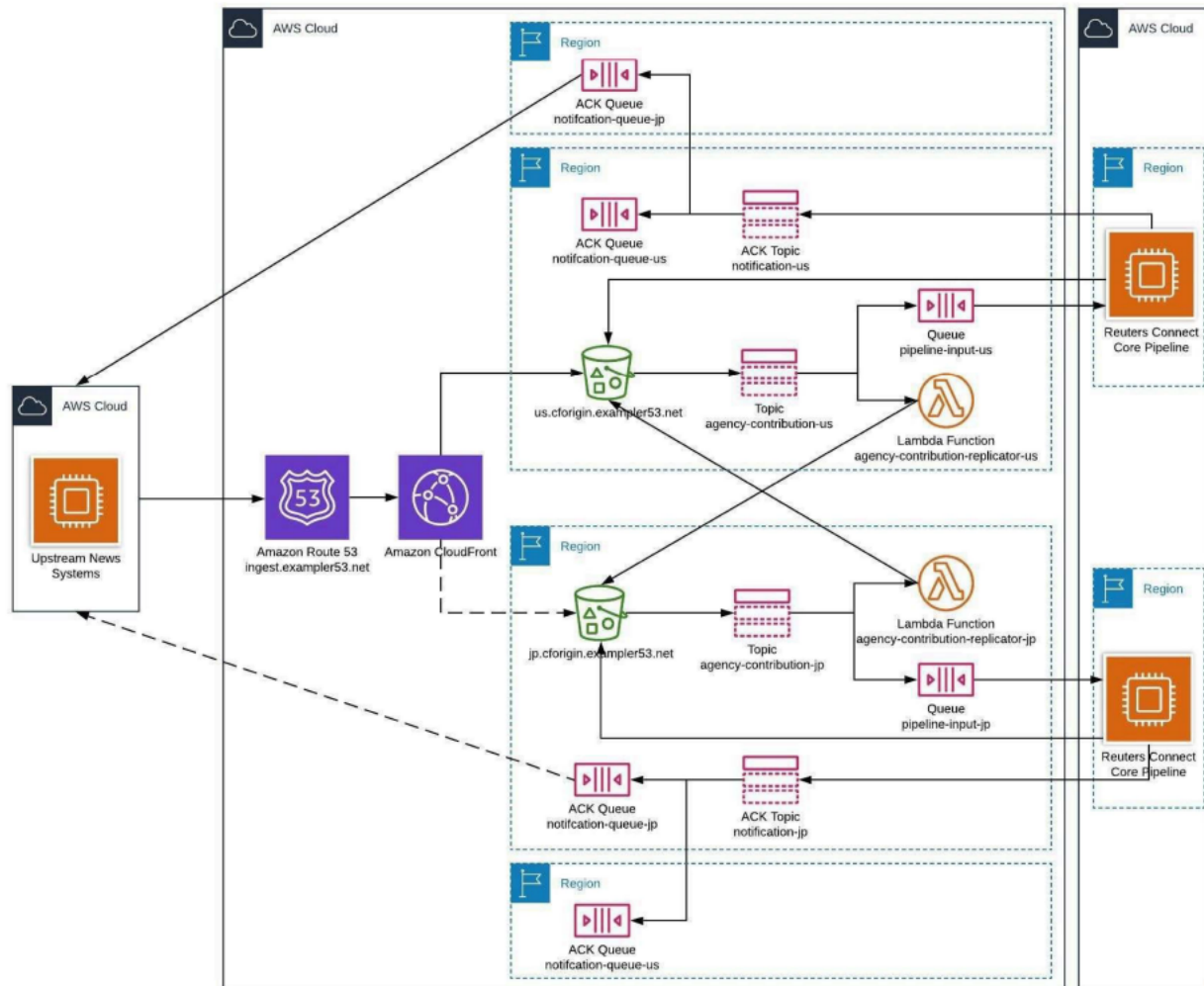


- The web service has a smart logic to detect content availability on S3 and can serve download from local storage if there is a regional problem.
- The web portal runs in live-live mode across two Regions so only some customers are impacted if one Region has a problem.
- The whole system is duplicated across two Regions and DNS-based failover is easy and fast using Route 53.
- The content published during the failure period is queued up and automatically re-uploaded after S3 recovers in a Region.

We also use a similar setup to have a resilient and flexible content ingestion capability for all of our upstream systems. These are the systems that produce videos, text, pictures, and so on.

Reuters's multi-region ingestion setup

The following architecture diagram shows our multi-region ingestion setup:



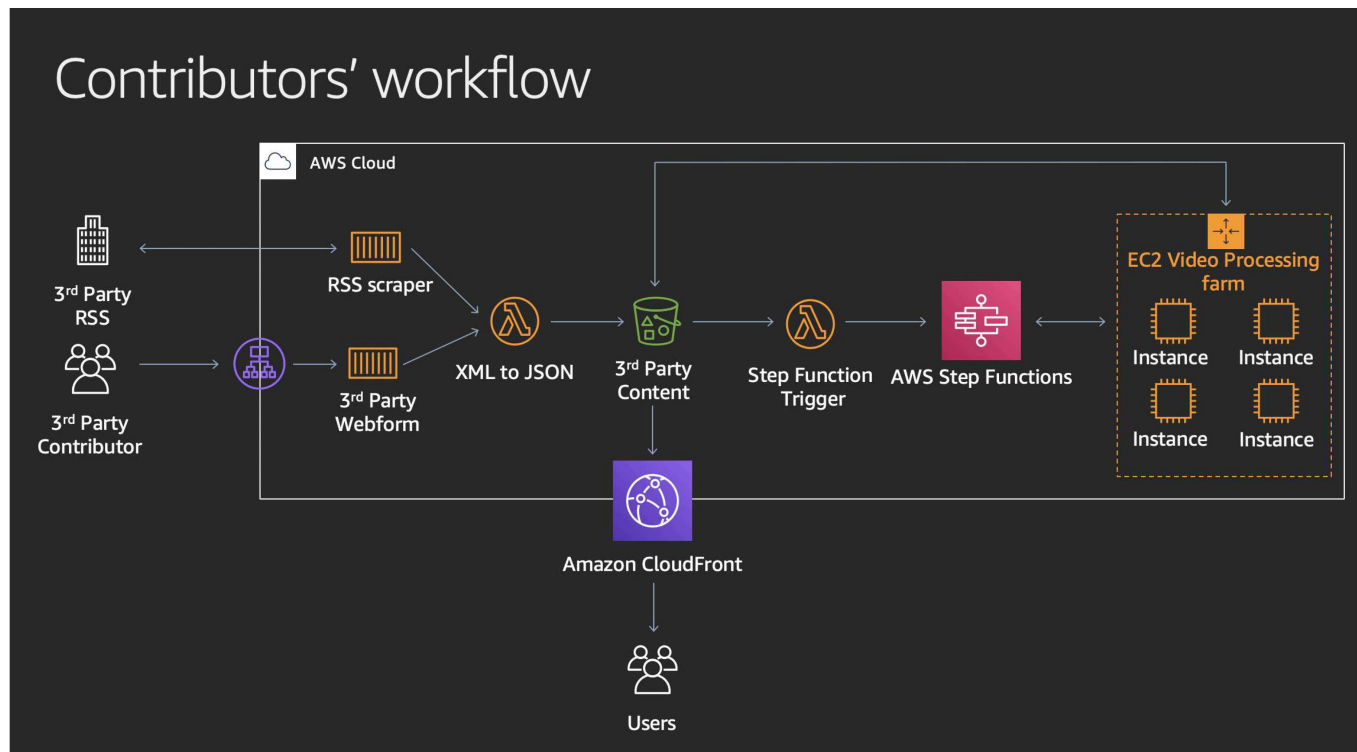
1. Reuters Connect has an Amazon S3 bucket in each of its two Regions for upstream systems to deliver content into. A single domain name is provided to the upstream systems, which must only upload content once (instead of dual uploading to both buckets in two Regions).
2. The single domain registered via Route 53 is pointed to a CloudFront distribution, in which one of the S3 buckets is the origin. Our system handles the S3 bucket failover by changing the CloudFront origin, either manually or via Lambda.
3. The upstream system's AWS account is granted permissions to sign CloudFront URLs, which are used to PUT objects into S3.
4. A replicator Lambda function is used to listen on PutObject events and copy the object to another bucket. The replicator function also adds a flag in the replicated object metadata to identify this object is replicated from the other bucket, so it shouldn't be replicated back.
5. The PutObject events are also sent to an SQS queue that our system listens to. Once our application receives the message for a new item coming in, it downloads the item from S3 and processes it.

6. After the processing is completed, it will send a notification to an SNS topic. The topic then publishes the same notification to an SQS “acknowledgment queue” in both Regions – and verifies that only if the original item was processed and not a replica (S3 object metadata helps here).
7. Upstream systems can listen to one of the SQS “acknowledgment queues” close to their location and get notified of the processing results of each item. If SQS fails in one of the Regions, the upstream systems must handle the failover between the two Regions if SQS fails in one of the Regions.

Third-party contributor's workflow

At Reuters, we primarily produce our own content on a large scale, with more than 2500 journalists in 200+ worldwide locations. However, we cannot be everywhere every time there is a breaking news story, so we also rely on trusted third parties and user-generated content (UGC). There are many ways we receive and ingest content into our systems for processing:

- FTP/SFTP
- RSS
- Web Services
- Webform/web portals



Starting from the left in the preceding diagram, this is our workflow breakdown:

1. Third parties can provide an RSS page, which we are constantly checking for a news story via our application. They can also contribute using our scalable, secure, MFA protected custom website, managed in an ECS cluster, to supply their metadata, scripts, and/or videos after they log in.

2. We then complete metadata standardization via a Lambda function from XML to JSON.
3. We upload both the metadata and the videos to S3.
4. This then triggers a Lambda function that triggers a Step Function job to start the video processing.
5. We then make the content available via the same CloudFront distribution we use to serve content from our Reuters Connect platform.

Securing our content

Finally, it is critical that we secure our content on Amazon S3 to protect against accidental or malicious deletion, suspicious activities, intrusion, or stolen access rights. Amazon S3 bucket policies, Multi-Factor Authentication (MFA) delete protection, and CloudTrail and IAM permissions provide various ways to protect our data.

We use the following IAM user policy to make sure no IAM users (even admin) can delete Amazon S3 objects or buckets. This policy must be attached to all users in the account to be effective, and that can be done simply by organizing users into groups and attaching the policy to those groups.

```
JSON
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1523630470225",
      "Action": [
        "s3:DeleteBucket",
        "s3:DeleteBucketPolicy",
        "s3:DeleteBucketWebsite",
        "s3:DeleteObject",
        "s3:DeleteObjectTagging",
        "s3:DeleteObjectVersion",
        "s3:DeleteObjectVersionTagging"
      ],
      "Effect": "Deny",
      "Resource": [
        "arn:aws:s3:::tr-agency-video-content-eu"
      ]
    }
  ]
}
```

We also programmatically create and add the following bucket policy to enable partners to consume our content directly from S3:

```
JSON
{
  "Version": "2012-10-17",
  "Statement": [
```

```
{
  "Sid": "$PARTNER1 read only",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::$PARTNER1_AWS_ACCOUNT_ID:root"
  },
  "Action": "s3:GetObject",
  "Resource": "arn:aws:s3:::tr-agency-video-content-eu/archive/*"
},
{
  "Sid": "$PARTNER2 read only",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::$PARTNER2_AWS_ACCOUNT_ID:root"
  },
}
```

Lastly, we also selectively enable S3 Block Public Access feature on our buckets that are used for anything other than web hosting.

Conclusion

AWS has changed the way our news agency works. It has made computing power and storage more available, while offloading the heavy-duty tasks like provisioning those resources. Reuters has been using Amazon S3 for years, and we use the AWS service portfolio as much as we can. Building new projects on the AWS platform has completely changed how we deliver trusted news content to our customers. Its rich feature set, security, and its overall cost of management and delivery has helped us cut costs and latency, not just for ourselves but for our customers too. This leaves us with more time to focus on key business objectives and delivering news faster. This is better for us, our customers, and the public consumers of our news—needless to say, speed for a news agency is critical.

AWS has also removed obstacles to innovation like high costs and complexity. Moreover, it set us up for future success throughout the domain of Artificial Intelligence and Machine Learning. S3 enabled us to serve our content more efficiently and reliably, both for human consumptions and also for machine consumption via [AWS Data Exchange](#). If you need data for speech-to-text training, visual analysis training, or for machine translation, [Reuters Content on AWS Data Exchange](#) is a great place to start.

I hope sharing our experience helps you to see the flexibility and build on AWS and manage content efficiently and resiliently to provide the best service for your customers. If you are interested in learning more about how Reuters uses AWS services, stay tuned for another blog post about our video archive capability. In that post, I plan on discussing how we use serverless technology to deliver content from the last 124 years. Thanks for reading, and please comment with any questions you may have! The following two video recordings of my presentations at AWS re:Invent 2019 offer deep dives into our use of Amazon S3 and best practices:

How Thomson Reuters manages content in Amazon S3:

How Thomson Reuters manages content in Ama

AWS Events



Watch on

AWS re:Invent 2019: Best practises for Amazon S3 ft. Thomson Reuters (STG302-R2) – Long version:

AWS re:Invent 2019: Best practises for Amazon

AWS Events



Watch on

The content and opinions in this post are those of the third-party author and AWS is not responsible for the content or accuracy of this post.

TAGS: [Amazon CloudFront](#), [Amazon Route 53](#), [Amazon Simple Storage Service \(Amazon S3\)](#), [AWS Cloud Storage](#), [AWS Data Exchange](#), [Thomson Reuters](#)



Romeo Radanyi

Romeo is a London-based Solutions Architect specializing in Artificial Intelligence and Machine Learning for Thomson Reuters. He is also a Data Science instructor at A Cloud Guru. He focuses on design, delivering large scale enterprise solutions, and pragmatic ways to apply machine learning to products. In his free time, he loves to do mixed martial arts, read, and travel.



Comments

Log in to comment
