AWS SUMMIT

Deep Dive on Amazon S3

Julien Simon, Principal Technical Evangelist, AWS
julsimon@amazon.fr - @julsimon

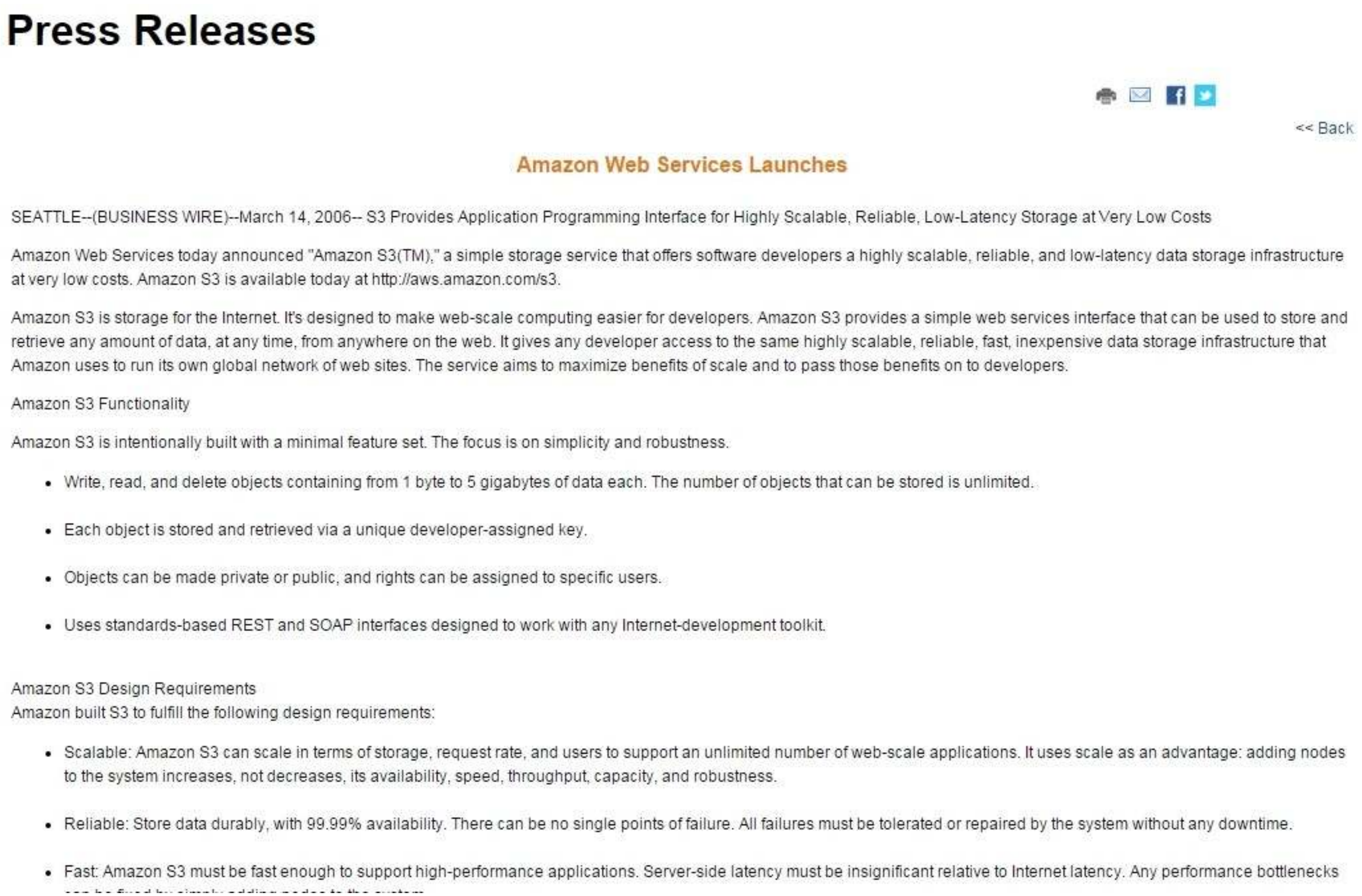
Loke Dupont, Head of Services, Xstream A/S
loke@xstream.net

amazon web services

Agenda

- Introduction
- Case study: Xstream A/S
- Amazon S3 Standard-Infrequent Access
- Amazon S3 Lifecycle Policies
- Amazon S3 Versioning
- Amazon S3 Performance & Transfer Acceleration

Happy birthday, S3



Press Releases

<< Back

Amazon Web Services Launches

SEATTLE--(BUSINESS WIRE)--March 14, 2006-- S3 Provides Application Programming Interface for Highly Scalable, Reliable, Low-Latency Storage at Very Low Costs

Amazon Web Services today announced "Amazon S3(TM)," a simple storage service that offers software developers a highly scalable, reliable, and low-latency data storage infrastructure at very low costs. Amazon S3 is available today at http://aws.amazon.com/s3.

Amazon S3 is storage for the Internet. It's designed to make web-scale computing easier for developers. Amazon S3 provides a simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web. It gives any developer access to the same highly scalable, reliable, fast, inexpensive data storage infrastructure that Amazon uses to run its own global network of web sites. The service aims to maximize benefits of scale and to pass those benefits on to developers.

Amazon S3 Functionality

Amazon S3 is intentionally built with a minimal feature set. The focus is on simplicity and robustness.

- Write, read, and delete objects containing from 1 byte to 5 gigabytes of data each. The number of objects that can be stored is unlimited.
- Each object is stored and retrieved via a unique developer-assigned key.
- Objects can be made private or public, and rights can be assigned to specific users.
- Uses standards-based REST and SOAP interfaces designed to work with any Internet-development toolkit.

Amazon S3 Design Requirements
Amazon built S3 to fulfill the following design requirements:

- Scalable: Amazon S3 can scale in terms of storage, request rate, and users to support an unlimited number of web-scale applications. It uses scale as an advantage: adding nodes to the system increases, not decreases, its availability, speed, throughput, capacity, and robustness.
- Reliable: Store data durably, with 99.99% availability. There can be no single points of failure. All failures must be tolerated or repaired by the system without any downtime.
- Fast: Amazon S3 must be fast enough to support high-performance applications. Server-side latency must be insignificant relative to Internet latency. Any performance bottlenecks

S3: our customer promise

Durable

99.999999999%

Available

Designed for 99.99%

Scalable

Gigabytes → Exabytes

Using Amazon S3 for video ingestion

Loke Dupont, Head of Services, Xstream A/S
loke@xstream.net

What does Xstream do?

Xstream is an online video platform provider. We sell OVP’s to broadcasters, ISP’s, cable companies etc.

What we are trying to provide is a ”white-label Netflix” that our customers can use to provide video services to end users.

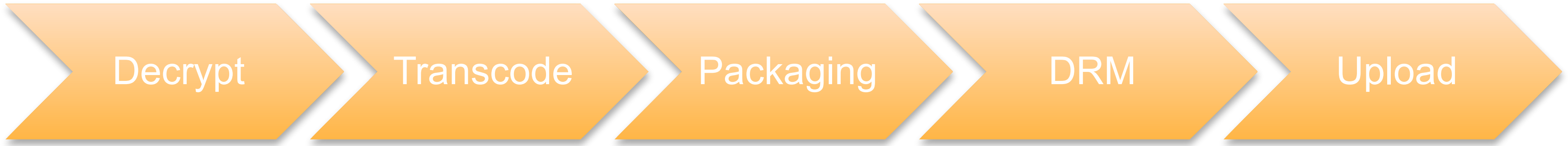
As part of that delivery, we ingest large amounts of video.

Challenges of ingesting video

Challenges of premium video ingestion

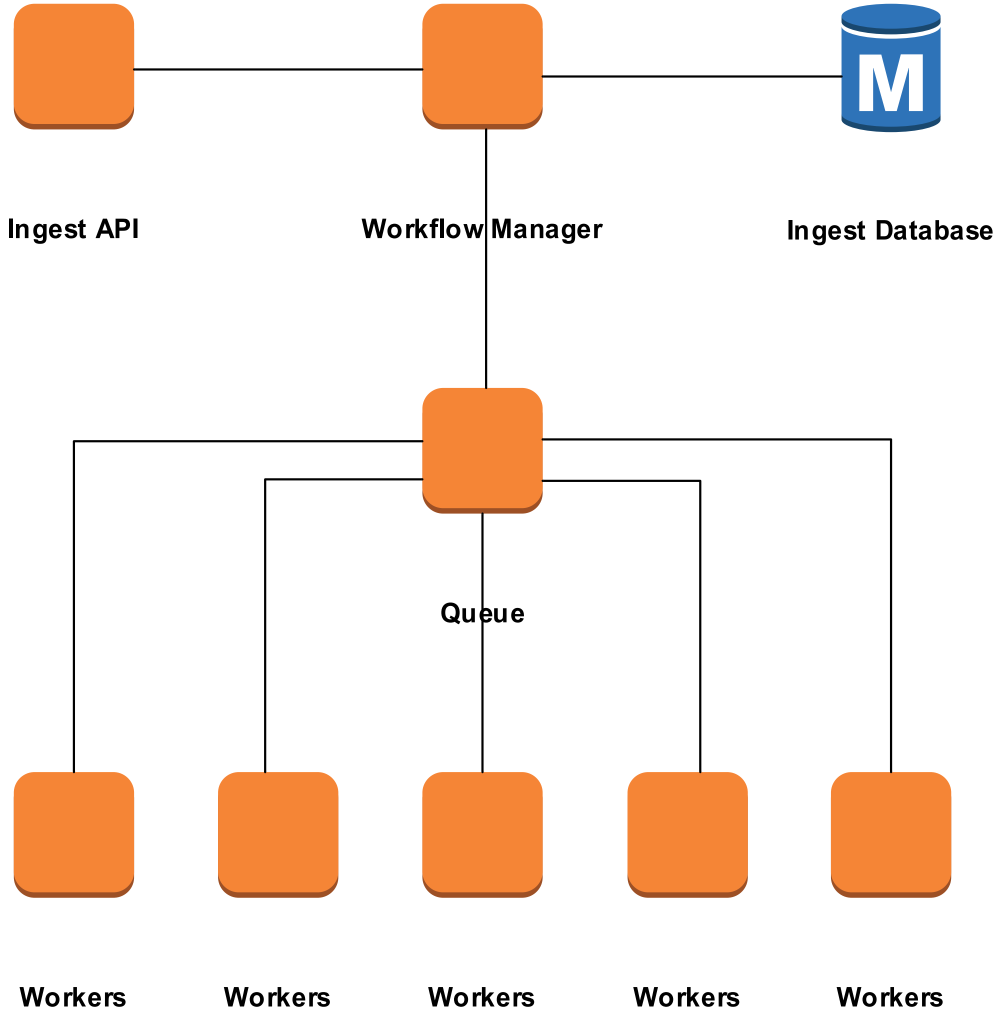
- Very large files (upwards of several hundred GB)
- Content security is extremely important
- Content integrity is very important (no video corruption)
- Content often arrive in batches of 1000+ videos
- Content needs to be available to all ingest processes

Ingest workflow



Ingest architecture

- Amazon RDS MySQL instance for data
- Running 100% on Amazon EC2 instances
- Planning to replace EC2 with AWS Lambda and Amazon SQS



How does Amazon S3 help?

Amazon S3 real world usage

In April, in just one region, we had:

- **300 TB/month** of short term storage in S3
- **62 million PUT/COPY/POST/LIST** requests
- **55 million GET** requests

In the same region we had **848 TB** of Amazon Glacier long term archive storage

Previous workflow vs. Amazon S3

Previous workflow

- Large files moved between machines
- Managing access had to be done pr. machine
- Disk space had to be managed carefully
- Encryption at rest was tricky
- Constant file integrity checks

Amazon S3

- Files always accessible on Amazon S3
- Managing access for bucket using policy and Amazon IAM
- Running our of space, practically impossible
- Encryption is easy
- S3 checks integrity for us, using checksums

What else do we get for free?

Versioning, which allows us to retrieve deleted and modified objects.

Easy Amazon Glacier integration for long term content archiving of “mezzanine” assets. Alternatively Amazon S3-IA could be used.

Event notifications using Amazon SNS, Amazon SQS and AWS Lambda

Demo

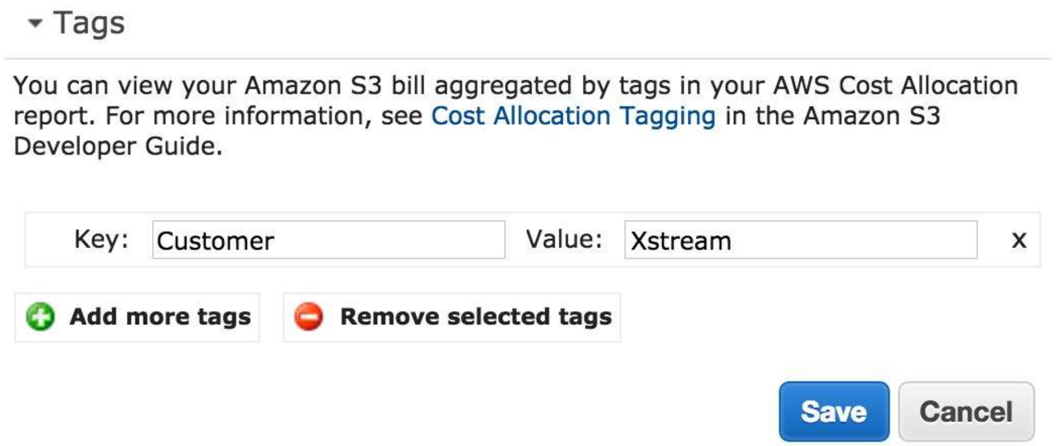
Amazon S3 events & AWS Lambda

Sample code: http://cloudvideo.link/lambda.zip

Lesser known Amazon S3 features – Bucket tagging

Bucket tagging is a great feature for cost allocation.

Assign custom tags to your bucket and they can be used to separate cost pr. customer or pr. project.



You can view your Amazon S3 bill aggregated by tags in your AWS Cost Allocation report. For more information, see Cost Allocation Tagging in the Amazon S3 Developer Guide.

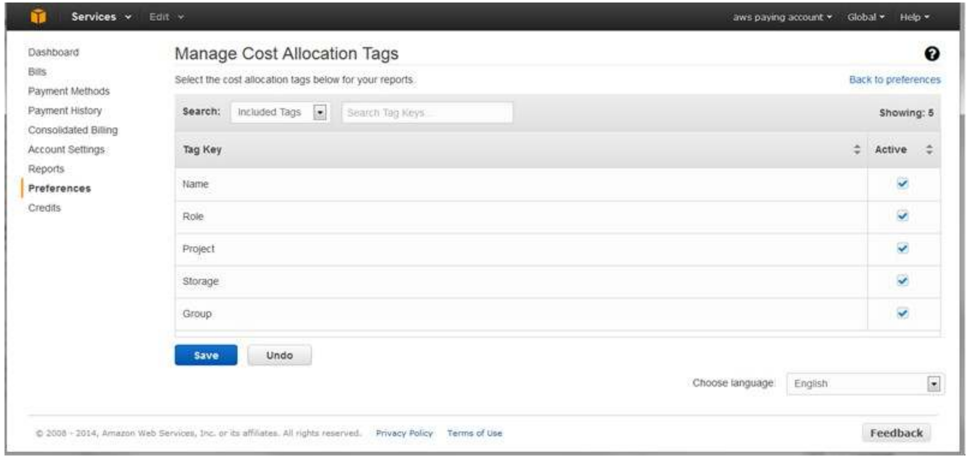
Key: Customer Value: Xstream

Getting cost with tags

Setup Cost allocation tags in preferences.

Using the AWS Cost Explorer to create a new report.

Select filter by “tags” and select the tag you want to filter by.



Manage Cost Allocation Tags

Select the cost allocation tags below for your reports.

Search: Included Tags | Search Tag Keys... | Showing: 5

Tag Key	Active
Name	✓
Role	✓
Project	✓
Storage	✓
Group	✓

Choose language: English

© 2008 - 2014, Amazon Web Services, Inc. or its affiliates. All rights reserved.

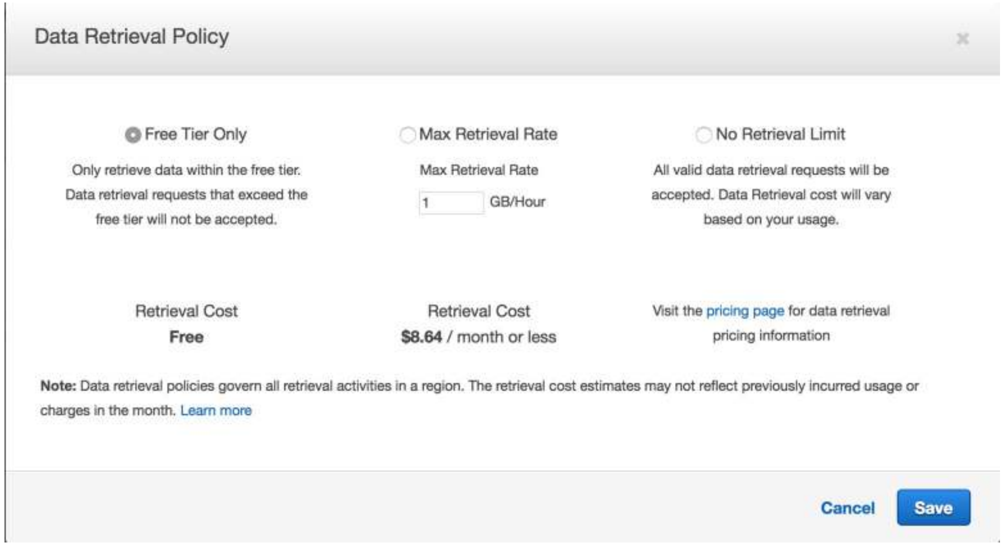
Lessons learned

Lessons learned from Amazon Glacier

Verify archive creation before deleting data.

Retrieval is priced by “peak rate” – spread it out

Retrieval has several hours latency


Data Retrieval Policy

[✓ Free Tier Only]
Only retrieve data within the free tier. Data retrieval requests that exceed the free tier will not be accepted.

Retrieval Cost
Free

Max Retrieval Rate
Max Retrieval Rate
1 GB/Hour

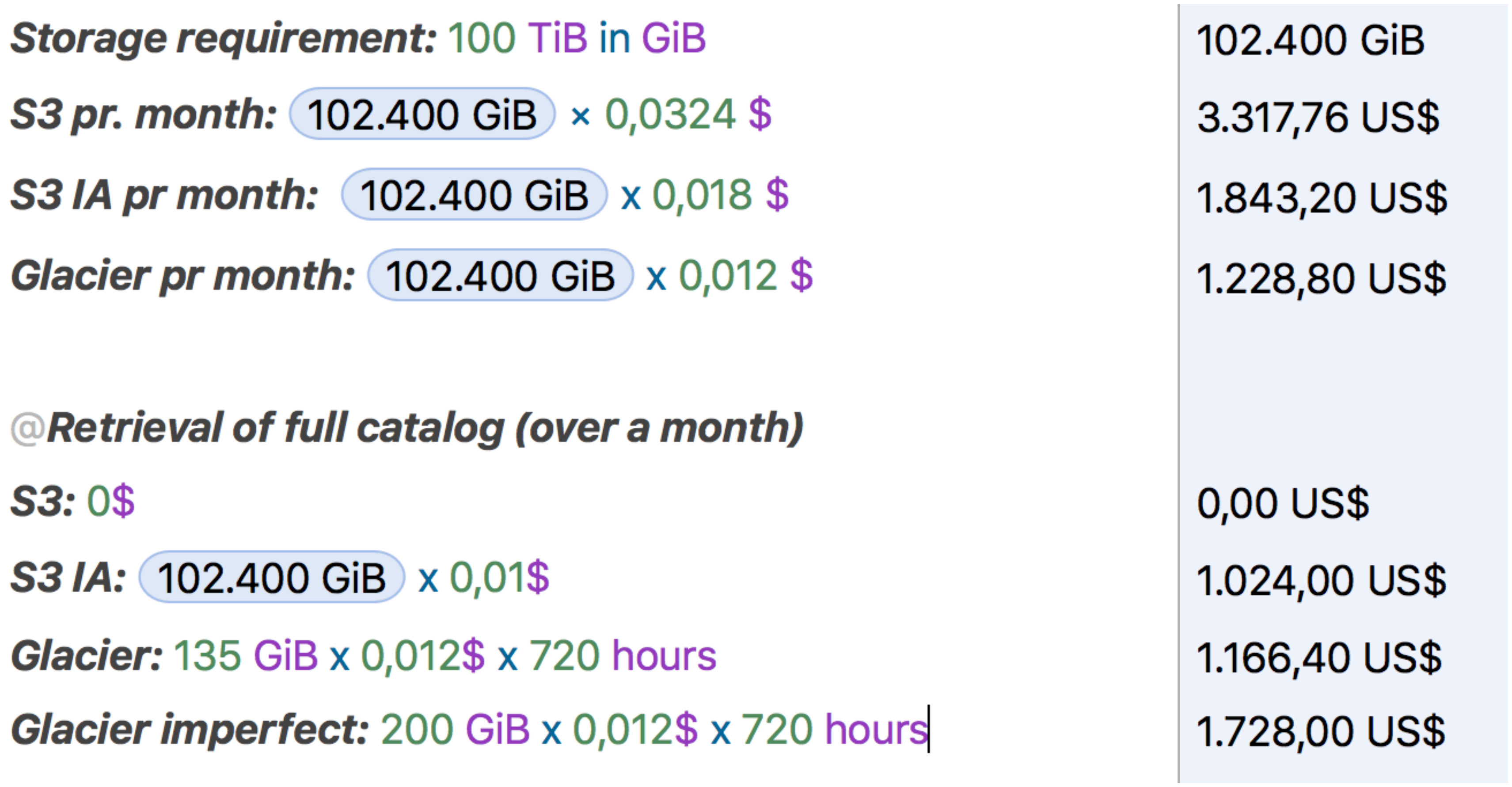
Retrieval Cost
$8.64 / month or less

No Retrieval Limit
All valid data retrieval requests will be accepted. Data Retrieval cost will vary based on your usage.

Visit the pricing page for data retrieval pricing information

Note: Data retrieval policies govern all retrieval activities in a region. The retrieval cost estimates may not reflect previously incurred usage or charges in the month. Learn more

AWS Storage cost comparison



Storage requirement: 100 TiB in GiB	102.400 GiB
S3 pr. month: 102.400 GiB × 0,0324 $	3.317,76 US$
S3 IA pr month: 102.400 GiB x 0,018 $	1.843,20 US$
Glacier pr month: 102.400 GiB x 0,012 $	1.228,80 US$
@***Retrieval of full catalog (over a month)***	
S3: 0$	0,00 US$
S3 IA: 102.400 GiB x 0,01$	1.024,00 US$
Glacier: 135 GiB x 0,012$ x 720 hours	1.166,40 US$
Glacier imperfect: 200 GiB x 0,012$ x 720 hours	1.728,00 US$

Things we wish we new earlier

- Don’t use Amazon S3 filesystem wrappers
- Use Amazon IAM roles whenever possible
- If there is an AWS service for it, use that!
- Auto scaling, auto scaling, auto scaling

Continuous Innovation for Amazon S3

Amazon S3 Standard-IA

September 2015

Lifecycle policy & Versioning

16/3 New

Expired object delete marker

16/3 New

Incomplete multipart upload expiration

Performance

Object naming

Multipart operations

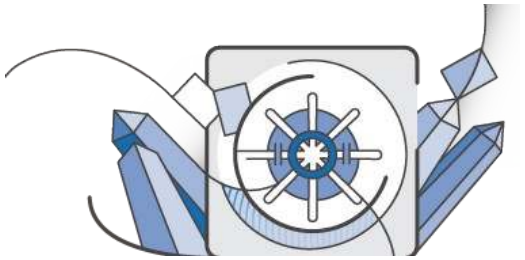
19/4 New Transfer Acceleration

S3 Infrequent Access

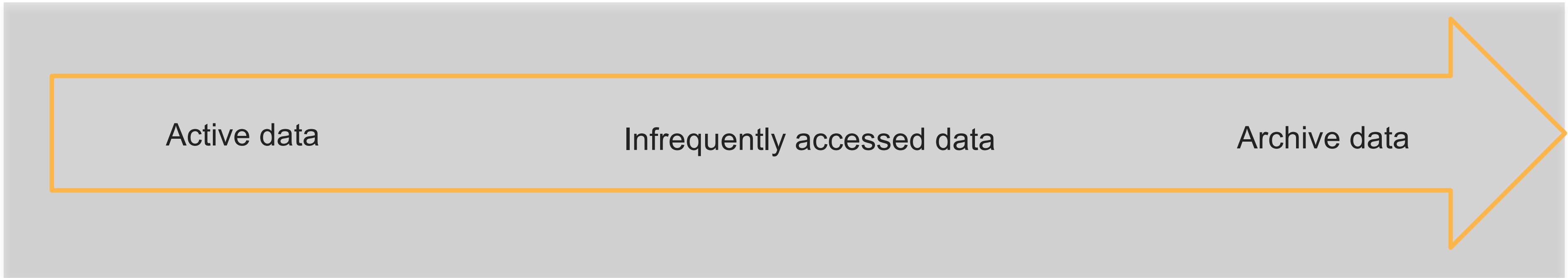
Choice of storage classes on Amazon S3

Standard

Standard - Infrequent Access



Amazon Glacier



Standard-Infrequent Access storage

Durable

11 9s of durability

Available

Designed for **99.9**% availability

High performance

Same throughput as Amazon S3 Standard storage

Easy to use

- No impact on user experience
- Simple REST API
- Single bucket

Secure

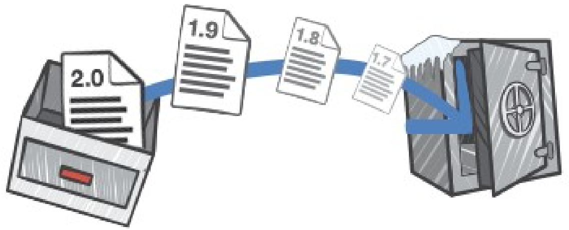
- Server-side encryption
- Use your encryption keys
- KMS-managed encryption keys

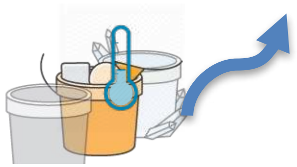
Integrated

- Lifecycle management
- Versioning
- Event notifications
- Metrics

Management policies

Lifecycle policies





- Automatic tiering and cost controls
- Includes two possible actions:
  - Transition: archives to Standard-IA or Amazon Glacier after specified time
  - Expiration: deletes objects after specified time
- Allows for actions to be combined
- Set policies at the prefix level

```
aws s3api put-bucket-lifecycle-configuration
--bucket BUCKET_NAME
--lifecycle-configuration file://LIFECYCLE_JSON_FILE
```

Standard → Standard-IA

```
"Rules": [
        {
            "Status": "Enabled",
            "Prefix": ”old_files",
            "Transitions": [
                {
                    "Days": 30,
                    "StorageClass": "STANDARD_IA"
                },
                {
                    "Days": 365,
                    "StorageClass": "GLACIER"
                }
            ],
            "ID": ”lifecycle_rule",
        }
]
```

Standard Storage -> Standard-IA

Standard-IA → Amazon Glacier

```
"Rules": [
        {
            "Status": "Enabled",
            "Prefix": ”old_files",
            "Transitions": [
                {
                    "Days": 30,
                    "StorageClass": "STANDARD_IA"
                },
                {
                    "Days": 365,
                    "StorageClass": "GLACIER"
                }
            ],
            "ID": ”lifecycle_rule",
        }
]
```

Standard Storage -> Standard-IA

Standard-IA -> Amazon Glacier

Versioning S3 buckets

- Protects from accidental overwrites and deletes
- New version with every upload
- Easy retrieval and rollback of deleted objects
- Three states of an Amazon S3 bucket
  - No versioning (default)
  - Versioning enabled
  - Versioning suspended

```
aws s3api put-bucket-versioning
--bucket BUCKET_NAME
--versioning-configuration
  file://VERSIONING_JSON_FILE
```

```
{
    "Status": "Enabled",
    "MFADelete": "Disabled"
}
```

Restricting deletes

- For additional security, enable MFA (multi-factor authentication) in order to require additional authentication to:
  - Change the versioning state of your bucket
  - Permanently delete an object version

- MFA delete requires both your security credentials and a code from an approved authentication device

Lifecycle policy to expire versioned objects

```
"Rules": [
{
        …

    "Expiration": {
        "Days": 60
    },
      "NoncurrentVersionExpiration": {
        "NoncurrentDays": 30
      }

    ]
}
```

Current version will expire after 60 days.

Older versions will be permanently deleted after 30 days.

Delete markers

- Deleting a versioned object puts a delete marker on the current version of the object

- No storage charge for delete marker
- No need to keep delete markers when all versions have expired (they slow down LIST operations)

- Use a lifecycle policy to automatically remove the delete marker when previous versions of the object no longer exist

Lifecycle policy to expire delete markers

```
"Rules": [
{
        …

    "Expiration": {
        "Days": 60,
        "ExpiredObjectDeleteMarker" :
true
    },
      "NoncurrentVersionExpiration": {
        "NoncurrentDays": 30
      }


    ]
}
```

Current version will expire after 60 days. A delete marker will be placed and expire after 60 days.

Older versions will be permanently deleted after 30 days.

Performance optimization

Distributing key names

Use a key-naming scheme with randomness at the beginning for high TPS

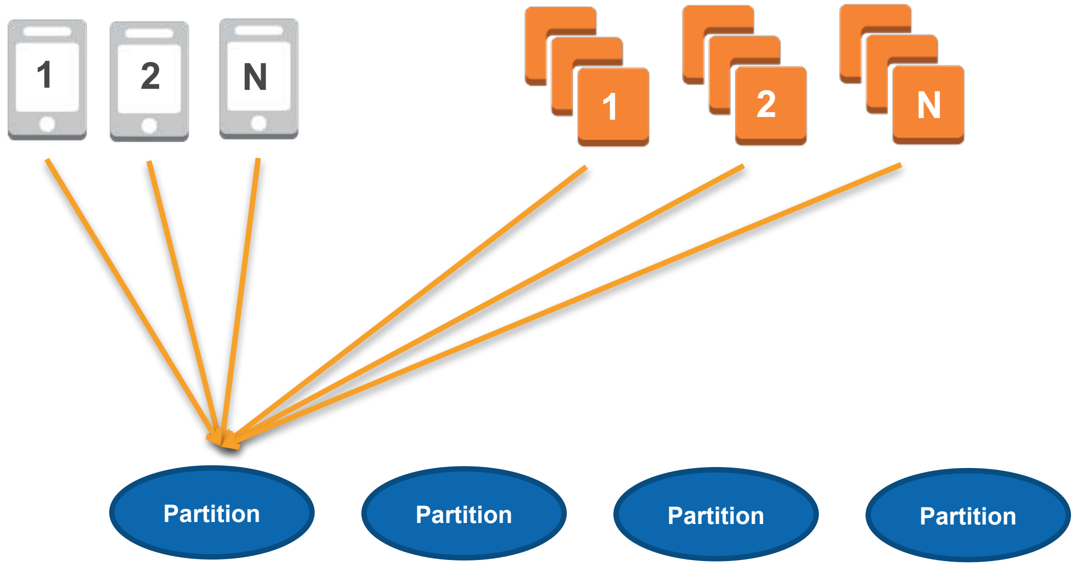
- Most important if you regularly exceed 100 TPS on a bucket
- Avoid starting with a date
- Avoid starting with sequential numbers

Don't do this…

```
<my_bucket>/2013_11_13-164533125.jpg
<my_bucket>/2013_11_13-164533126.jpg
<my_bucket>/2013_11_13-164533127.jpg
<my_bucket>/2013_11_13-164533128.jpg
<my_bucket>/2013_11_12-164533129.jpg
<my_bucket>/2013_11_12-164533130.jpg
<my_bucket>/2013_11_12-164533131.jpg
<my_bucket>/2013_11_12-164533132.jpg
<my_bucket>/2013_11_11-164533133.jpg
<my_bucket>/2013_11_11-164533134.jpg
<my_bucket>/2013_11_11-164533135.jpg
<my_bucket>/2013_11_11-164533136.jpg
```

Distributing key names

…because this is going to happen



Distributing key names

Add randomness to the beginning of the key name…

```
<my_bucket>/521335461-2013_11_13.jpg
<my_bucket>/465330151-2013_11_13.jpg
<my_bucket>/987331160-2013_11_13.jpg
<my_bucket>/465765461-2013_11_13.jpg
<my_bucket>/125631151-2013_11_13.jpg
<my_bucket>/934563160-2013_11_13.jpg
<my_bucket>/532132341-2013_11_13.jpg
<my_bucket>/565437681-2013_11_13.jpg
<my_bucket>/234567460-2013_11_13.jpg
<my_bucket>/456767561-2013_11_13.jpg
<my_bucket>/345565651-2013_11_13.jpg
<my_bucket>/431345660-2013_11_13.jpg
```

Other ideas

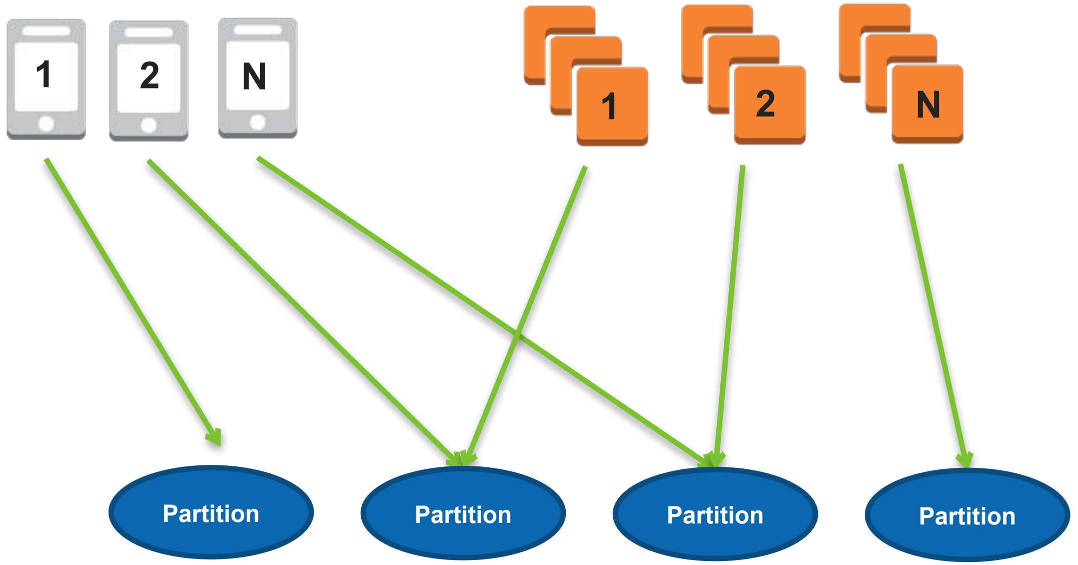
- Store objects as a hash of their name and add the original name as metadata

  `“deadbeef_mix.mp3” → 0aa316fb000eae52921aab1b4697424958a53ad9`

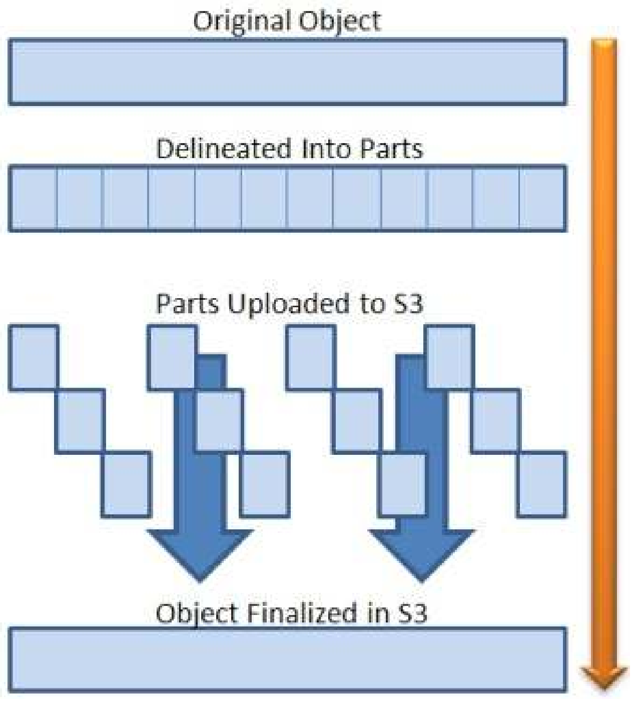
- Reverse key name to break sequences

Distributing key names

…so your transactions can be distributed across the partitions



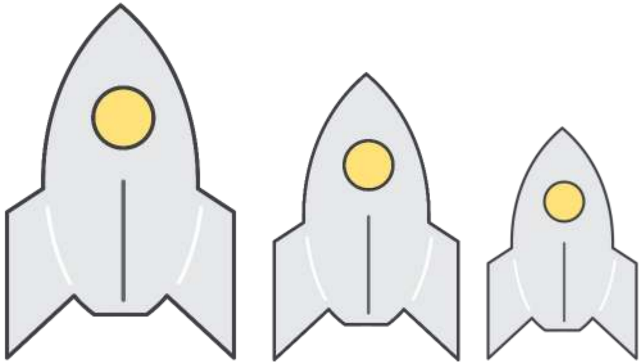
Parallelizing PUTs with multipart uploads



- Increase aggregate throughput by parallelizing PUTs on high-bandwidth networks
- Move the bottleneck to the network where it belongs
- Increase resiliency to network errors; fewer large restarts on error-prone networks

https://aws.amazon.com/fr/premiumsupport/knowledge-center/s3-multipart-upload-cli/

Choose the right part size



- Maximum number of parts: 10,000
- Part size: from 5MB to 5GB

- Strike a balance between part size and number of parts
  - Too many small parts
    → connection overhead
    (TCP handshake & slow start)
  - Too few large parts
    → not enough benefits of multipart

Incomplete multipart upload expiration policy

Incomplete multipart upload expiration

- Multipart upload feature improves PUT performance
- Partial upload does not appear in bucket list
- Partial upload does incur storage charges
- Set a lifecycle policy to automatically expire incomplete multipart uploads after a predefined number of days

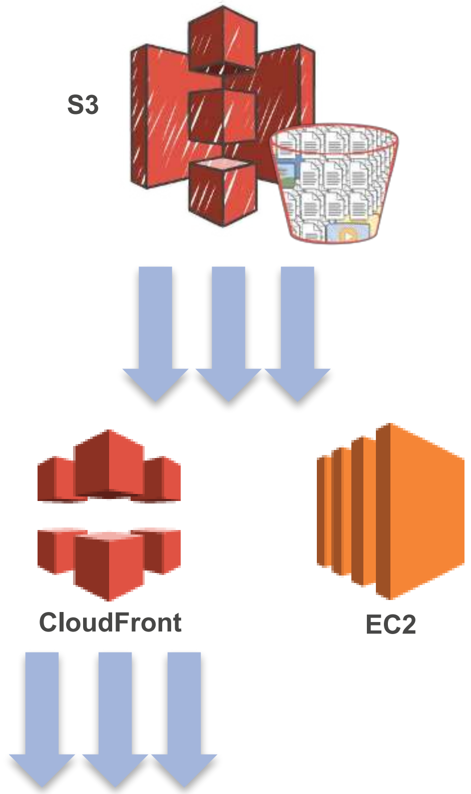
Lifecycle policy to expire multipart uploads

```
"Rules": [
{
      …

    "AbortIncompleteMultipartUpload": {
        "DaysAfterInitiation": 7
    }
    ]
}
```

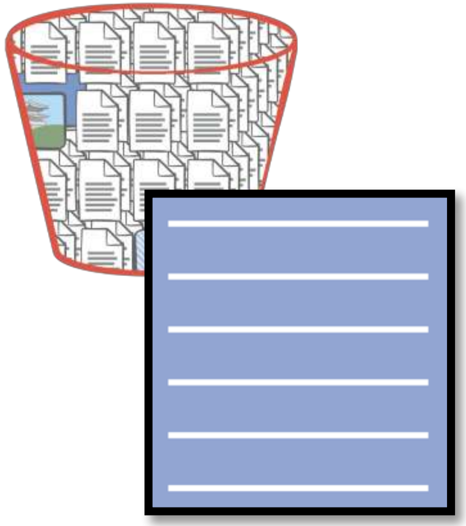
Incomplete multipart uploads will expire seven days after initiation

Parallelize your GETs



- Use Amazon CloudFront to offload Amazon S3 and benefit from range-based GETs
- Use range-based GETs to get multithreaded performance when downloading objects
- Compensates for unreliable networks
- Benefits of multithreaded parallelism

Parallelizing LIST



- Parallelize LIST when you need a sequential list of your keys
- Secondary index to get a faster alternative to LIST
    - Sorting by metadata
    - Search ability
    - Objects by timestamp

"*Building and Maintaining an Amazon S3 Metadata Index without Servers*"
AWS blog post by Mike Deck on using Amazon DynamoDB and AWS Lambda

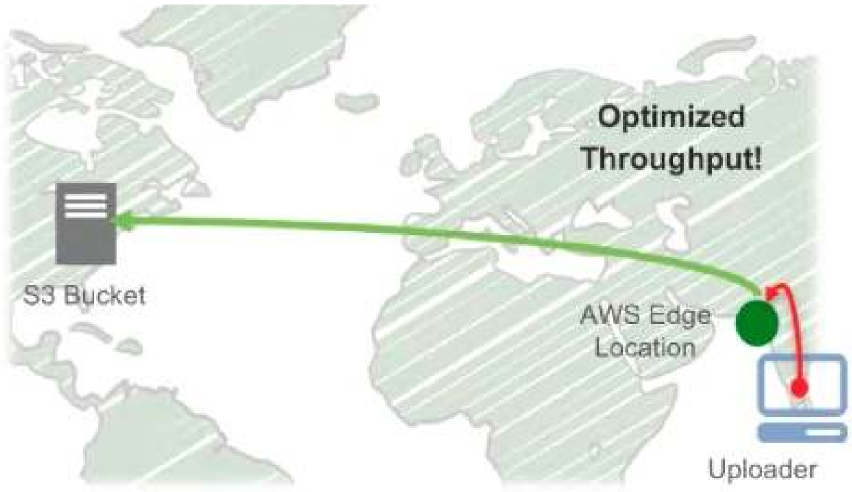
Amazon S3
Transfer Acceleration

Amazon S3 Transfer Acceleration

- Designed for long distance transfers
- Send data to Amazon S3 using the 54 AWS Edge Locations
- Up to 6 times faster thanks to the internal AWS network
- No change required (software, firewalls, etc.)

- Must be explicitely set by customers, on a per-bucket basis
- Pay according to volume : from $0.04 / GB
- You’re only charged if transfer is faster than using Amazon S3 endpoints

Amazon S3 Transfer Acceleration



```
aws s3api put-bucket-accelerate-configuration
--bucket BUCKET_NAME
--accelerate-configuration
  file://ACCELERATE_JSON_FILE
```

```
{
      "Status": "Enabled"
}
```

AWS Snowball

- New version: 80 Terabytes (+60%)
- Available in Europe (eu-west-1)
- All regions available by the end of 2016
- $250 per operation
- 25 Snowballs → 2 Petabytes in a week for $6250

Recap

- Case study: Xstream A/S
- Amazon S3 Standard-Infrequent Access
- Amazon S3 Lifecycle Policies
- Amazon S3 Versioning
- Amazon S3 Performance & Transfer Acceleration

AWS SUMMIT

Thank You!

amazon web services