Verifying S3 Uploads via Direct Checksum Read from S3 Provider

Justin James
Application Engineer
iRODS Consortium

June 29 - July 2, 2026
iRODS User Group Meeting 2026
Barcelona, Spain

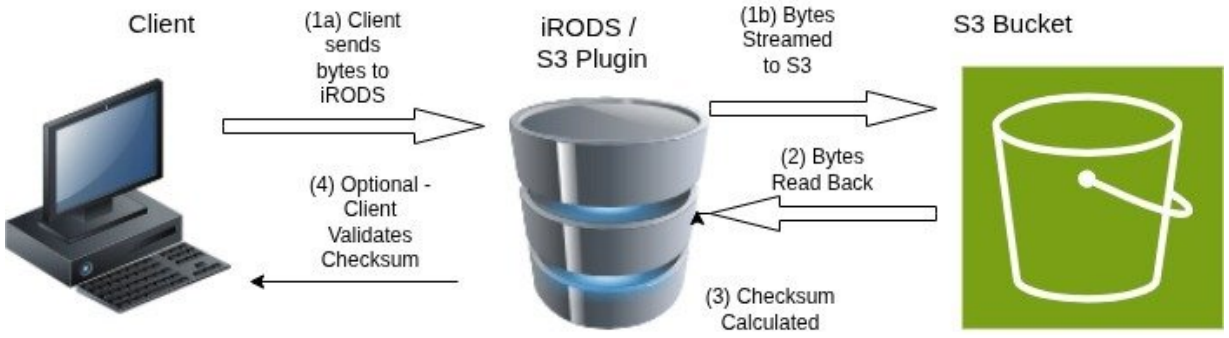
Problem Statement

Note: In this talk I will use the term "checksum" as a generic term for either checksums or hashes. I will use the term "hash" only when I am specifically talking only about hashes.

User writes a large data object to S3 request a checksum:

- iput -K
- iput -k

iRODS writes the entire data object to S3 and then reads the entire data object back to calculate and/or verify the checksum.



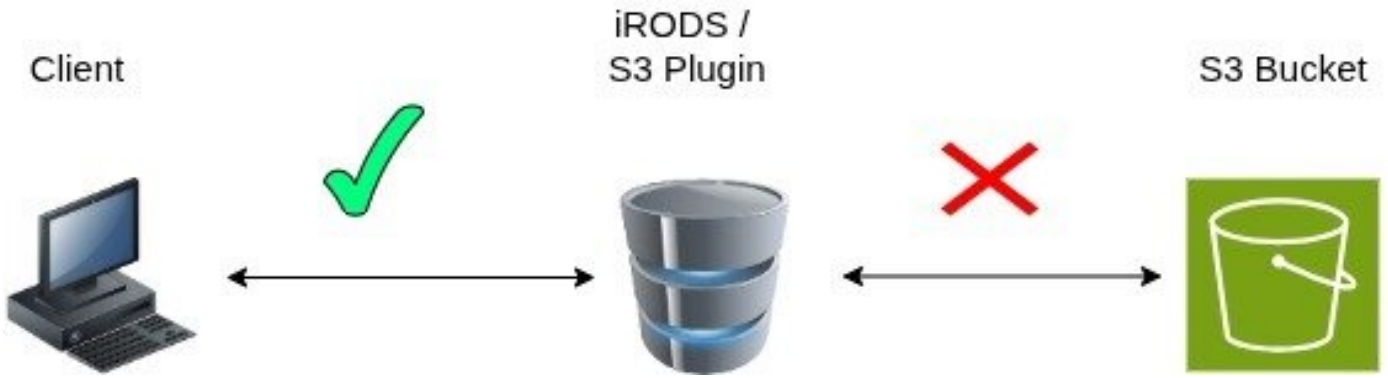
This is obviously a very time consuming activity and reduces throughput considerably.

Possible Solution

Calculate the checksum on the fly as data is being written.

Problem:

- With multipart, the S3 plugin can do this on a part-by-part basis but for most hashes there is no way to combine those parts back into a full file checksum.
- Even if you can, this is only calculating the checksum between the client and the iRODS server. This would not detect data corruption going to the S3 appliance.



Partial Solution

Have the S3 appliance report the checksum back.

- AWS calculates and stores the CRC64/NVME checksum automatically.
- **GetObjectAttributes** will retrieve this checksum.

iRODS core modifications:

- CRC64/NVME added as a valid checksum
- Add a resource plugin operation (**RESOURCE_OP_READ_CHECKSUM_FROM_STORAGE_DEVICE**) that checks the storage to see if it supports the direct read of a checksum.
- If the operation is not defined for the resource type, an error is returned. Control will continue as normal with a full read/calculate cycle. This is the case for unixfilesystem.
- If the operation does not return the desired checksum control continues with a full read/calculate cycle.
- If the operation returns the requested checksum, this checksum is returned and the full read/calculate cycle is skipped.

iRODS S3 resource plugin modifications:

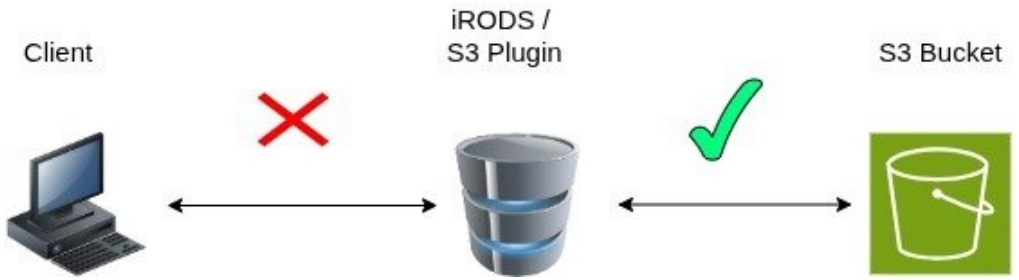
- Implement the operation to read the checksum via **GetObjectAttributes**
- If this returns an error or the specified checksum is not available, the operation returns an error causing the server to do a full read/calculate cycle.
- Otherwise return the checksum back to the iRODS core code.

This is sufficient for AWS.

Not all S3 Providers Calculate CRC64/NVME Automatically

Other appliances may support the checksum but may not store it unless it is sent in the upload. This is true for MinIO.

Note: The S3 appliance will validate the checksum sent and reject the put if it is not valid. This itself is not sufficient to verify the file upload as it does not detect corruption between the client and iRODS.



Solution

S3 plugin calculates the checksum on the fly and sends it with the upload.

Multipart issues:

- This can only be done with a CRC type checksum due to the inability to combine hashes into a full object hash. (For hashes, S3 specifications are to calculate a hash of hashes which is not what we want.)
- This can't be sent as a header because that must be sent prior to all of the data being received.
- Can use trailing checksums but libs3 does not support them.
- Added support for trailing checksums in libs3.

With this all in place, we can get the CRC64/NVME directly from MinIO.

Putting it All Together

A couple of configuration options have been added to the S3 resource context string:

ENABLE_DIRECT_CHECKSUM_READ=1

- This instructs the S3 resource plugin to attempt to read the checksum via the **GetObjectAttributes** call.
- Try this with your appliance. If you are lucky the checksum will automatically be calculated and returned.
- If this API is not enabled, just disable this by removing it from the context string. You will have to continue with a full object read for checksums. (Note that keeping this enabled will not cause a failure as the fallback will be a full object read.)
- If **GetObjectAttributes** succeeds but does not return the desired checksum, try enabling trailing checksum on upload.

ENABLE_TRAILING_CHECKSUM_ON_UPLOAD=1

- This sends the trailing checksum when objects are uploaded.
- This is not necessary for AWS as the CRC64/NVME is automatically stored.
- This is necessary for MinIO.
- I am unsure about other S3 appliances.

For this to work for multipart uploads, the desired checksum must be CRC64/NVME. This can be set via the client or server defaults.

Questions