

## ACCESSING STORNEXT SELF-DESCRIBING OBJECTS

Want to run a cloud-based application against data that the StorNext® File System has stored in the cloud? Now you can. Need access to that data from other sites or looking to share files with business partners via the cloud? Now it's possible. Even if you just want insurance that you can always get your data back from the cloud, through disaster or technology obsolescence, it's here.

**NOTICE**

This technology brief contains information protected by copyright. Information in this technology brief is subject to change without notice and does not represent a commitment on the part of Quantum.

Quantum assumes no liability for any inaccuracies that may be contained in this technology brief.

Quantum makes no commitment to update or keep current the information in this document, and reserves the right to make changes to or discontinue this document and/or products without notice.

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or information storage and retrieval systems, for any purpose other than the purchaser's personal use, without the express written permission of Quantum.

Quantum, the Quantum logo, and StorNext are registered trademarks, and FlexTier is a trademark, of Quantum Corporation and its affiliates in the United States and/or other countries. All other trademarks are the property of their respective owners.

## TABLE OF CONTENTS

How StorNext Stores Data in the Cloud.....	3
New in StorNext v6.4 – Self-Describing Objects.....	4
Use Cases.....	4
Scripted Example - Cloud Data Recovery using PowerShell .....	5
Configuring AWS Tools for PowerShell .....	6
The Script .....	7
Conclusion .....	8

StorNext software has been cloud friendly for a long time, making it easy to use public-cloud destinations or on-prem object stores (henceforth simply “the cloud”) to store and protect warm and cold data. But it hasn’t always been easy to access files that StorNext stores in the cloud from elsewhere. This changed dramatically with the release of StorNext v6.4. This Tech Brief explains these changes, describes a few of the many possible use cases, and illustrates how to automate one of them with a sample Microsoft PowerShell script.

## HOW STORNEXT STORES DATA IN THE CLOUD

StorNext solutions store files into the cloud as objects with generated unique identifiers, or “keys.” This technique scales without bound, unlike the other common convention of making the object key equal to the source path and filename. Imagine a folder with tens of millions of files that gets copied to a cloud bucket using the file path and name as the object key. Now imagine someone renames that folder. Every object key must change – and because you can’t change the key once an object has been created, this means every object must be copied to a new object, which is very inefficient.

The tradeoff with the more scalable method the StorNext File System uses is that you can’t simply view the cloud bucket with any tool (e.g., AWS Console, Cyberduck, AWS S3 Browser, etc.) and know which source file corresponds with each object. The StorNext File System keeps track of this mapping, and until v6.4, you had to ask the StorNext File System for the information via the GUI, CLI, or a Web Services API call. This works fine when you have access to the StorNext system, but that’s not always the case.

StorNext File System maintains a one-to-one mapping between files and objects wherever possible. This means that each file is stored as a single object. The only exception is when the file is larger than the allowed maximum object size on the target cloud platform. In that case, the large file is broken into several sequential segments and stored as multiple objects. Maximum object sizes are typically multiple TB (e.g. the limit in AWS is 5TB), so segmentation is rare for most applications.

When StorNext File System stores a file into the cloud, the file content is stored in its original native form. There is no compression, deduplication, sharding, encoding, munging, or other manipulation of the content.

## NEW IN STORNEXT V6.4 – SELF-DESCRIBING OBJECTS

With the release of v6.4, StorNext software can now store additional metadata with objects, utilizing what are known as “user-defined” metadata fields. The new metadata fields are shown in Table 1, below.

Metadata	Description
Path	File path and file name on originating StorNext File System
Segment	For very large files stored as multiple objects, the segment number of this object
Offset	Pointer to start of data within the file
Mod Time	Last modified time of file on originating file system
Copy #	StorNext FlexTier™ copy number
Version	StorNext FlexTier file version
File Key	StorNext internal unique file identifier

Table 1 - StorNext v6.4 Additional Object Metadata

The *Path* and *Segment* data stored provides all the information needed to map an object back to the source file. Because it is stored with the object in a standard way, this makes the objects “self-describing.” There is no need to interrogate the StorNext File System to make use of them.

## USE CASES

Making objects self-describing opens many possibilities for re-use of the files StorNext File System stores in the cloud. For example:

- **Cloud Data Recovery**

If your StorNext system isn’t available due to a disaster, or decades from now you discover data in the cloud stored by your long-obsolete StorNext system, you can recover it.

- **Cloud or Remote Site Data Processing**

Objects StorNext File System stores to the cloud may be processed with cloud-native applications, or applications at another site, with no need for any information from or contact with the originating StorNext system.

- **Sharing and Collaborative Workflows**

Given the proper credentials and access, files stored in the cloud may be copied and used by others, such as subcontractors and business partners.

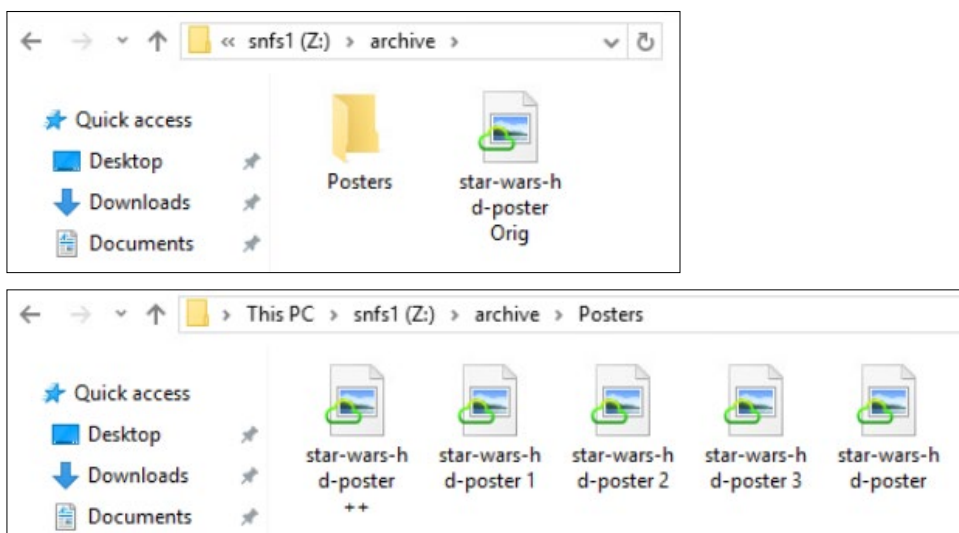
Combined with other StorNext features, complex workflows may be designed to span on-prem and cloud applications. For example, files generated locally and stored by StorNext solutions in the cloud may be analyzed by a cloud-native application, with results written to a separate bucket. The results bucket may be imported back into the StorNext file system namespace using the FlexTier Bucket Import feature, and visualization of the results performed on local workstations.

## SCRIPTED EXAMPLE - CLOUD DATA RECOVERY USING POWERSHELL

You can see the new StorNext v6.4 File System metadata for an object using network automation and file transfer tools like the AWS Console, AWS S3 Browser, or Cyberduck. This is useful if you only have a few objects to manipulate, but if you need to handle a lot of objects, it's better to write a script. This section includes a sample script that implements the first use case mentioned above – cloud data recovery. The script copies every object from a StorNext File System attached AWS bucket down to the local machine, starting at the specified directory root and using the paths and filenames stored by StorNext File System with the objects.

For simplicity, the example uses PowerShell on Windows. Amazon has something called AWS Tools for PowerShell that provides easy access to AWS APIs from PowerShell. PowerShell is available for Linux and MacOS too, and the basic concepts illustrated below are easily translated to other development environments.

For this example, a few files have been stored into an AWS S3 bucket by the StorNext system. Some of the files are in a subfolder. In Windows Explorer, the structure looks like this:



If you view the files as they appear on the StorNext system itself, the view is similar:

```

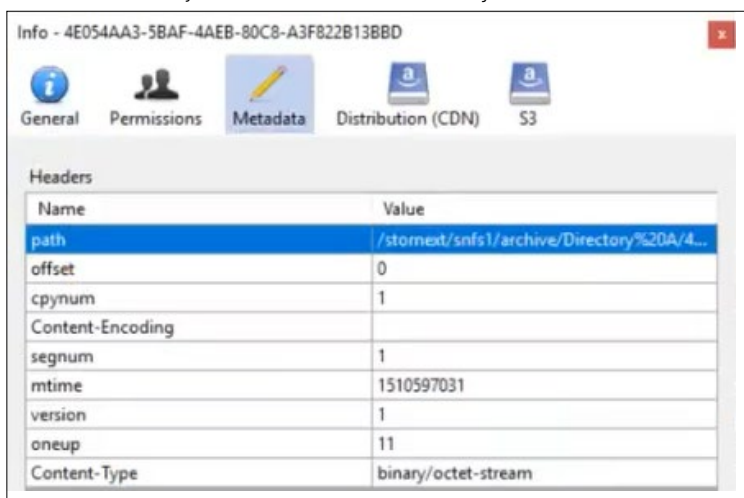
root@xcellis:/stornext/snfs1/archive/Posters
Using username "root".
Last login: Mon Oct 5 09:34:57 2020 from adc.talabs.local
[root@xcellis ~]# cd /stornext/snfs1/archive
[root@xcellis archive]# ls
star-wars-hd-poster Orig.jpg
[root@xcellis archive]# cd Posters
[root@xcellis Posters]# ls
star-wars-hd-poster 1.jpg  star-wars-hd-poster 3.jpg  star-wars-hd-poster.jpg
star-wars-hd-poster 2.jpg  star-wars-hd-poster ++.jpg
[root@xcellis Posters]#

```

If you look at the cloud bucket using Cyberduck (or a similar file transfer application), you see something else entirely:

Filename
14CFF606-7864-41CD-87BC-AE6CD04DD375
1953C869-3184-4781-9094-1A32C81A16DE
2D6A14A2-2DA5-4B5F-847D-981CAF96899C
4C8380E7-B145-427C-9173-F9BEBF866C6D
685A706C-9673-44DE-B00D-7C427DDF591F
70085D44-61A4-4EF8-A7EE-947021C4DAF8

Everything has been collapsed into the flat S3 namespace, so all that is visible are the generated object keys. If you select an object and view the metadata, you can see the source path and filename that the StorNext File System stored with that object:



Name	Value
path	/stornext/snfs1/archive/Directory%20A/4...
offset	0
cpynum	1
Content-Encoding	
segnum	1
mtime	1510597031
version	1
oneup	11
Content-Type	binary/octet-stream

The goal of the script is to take the objects shown in this flat view and copy them back down to local files on another system, recreating the source directory structure in the process, using the path information the StorNext File System has stored in the object metadata. But first, it is necessary to set up AWS Tools for PowerShell.

## CONFIGURING AWS TOOLS FOR POWERSHELL

Complete documentation for AWS Tools for PowerShell is available [here](#), but the steps below will enable a quick start.

1. Start PowerShell
2. Install the AWS Tools Installer:  
`Install-Module -Name AWSPowerShell.NetCore`
3. Import the AWS PowerShell module into the PowerShell session:  
`Import-Module AWSPowerShell.NetCore`

```
PS C:\Users\Administrator> install-module -name awspowershell.netcore
Untrusted repository
You are installing the modules from an untrusted repository. If you trust this repository, change its
InstallationPolicy value by running the Set-PSRepository cmdlet. Are you sure you want to install the modules from
"PSGallery"?
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): y
PS C:\Users\Administrator> import-module awspowershell.netcore
```

4. Verify that the PowerShell Execution Policy is set to RemoteSigned using the command `Get-ExecutionPolicy`. If it isn't, use `Set-ExecutionPolicy RemoteSigned` to set it.

```
PS C:\Users\Administrator> Get-ExecutionPolicy
RemoteSigned
PS C:\Users\Administrator> .
```

5. Set up the AWS credentials needed to access the bucket:

```
Set-AWSCredential -AccessKey YourAccessKeyHere -SecretKey YourSecretKeyHere
```

```
PS C:\Users\Administrator> Set-AWSCredential -AccessKey AKIARYAPYAIZ7HMWDZ7J -SecretKey CvsitocAaxmKOMLsHF8Ch/D5jd+yKP6hW
PS C:\Users\Administrator> .
```

## THE SCRIPT

The text in the shaded box below is the script that will perform the copy from the StorNext cloud bucket down to the target system. Embedded comments explain what each line does.

```
# Credentials must have been established prior to running this script using 'Set-AWSCredential'
# directly or recalling a stored profile

# Set source bucket and destination local path root

$bucket = '228868asn64tapeaws'

$localPath = 'C:\test'

# Scan all objects in bucket, load metadata into array 'objects'

$objects = Get-S3Object -BucketName $bucket

# Loop through object metadata, pull object keys into array 'fileName'

foreach ($object in $objects) {
    $fileName = $object.Key
}

# Loop through array 'fileName', & for each object key do a few things:

foreach ($file in $fileName) {
    # Get object metadata

    $Metadata = Get-S3ObjectMetadata -Bucketname $bucket -Key $file -Select 'Metadata'

    # Set 'localFileName' to SN stored path

    $localFileName = $Metadata.Item('x-amz-meta-path')

    # Create valid local path including filename

    $localFilePath = Join-Path $localPath $localFileName

    # URL Decode local path

    $localFilePath = [System.Web.HttpUtility]::UrlDecode($localFilePath)

    # Download object to local path & name

    Read-S3Object -BucketName $bucket -Key $file -File $localFilePath }}
```

When the script runs, it lists the target filenames of the objects as they are copied:

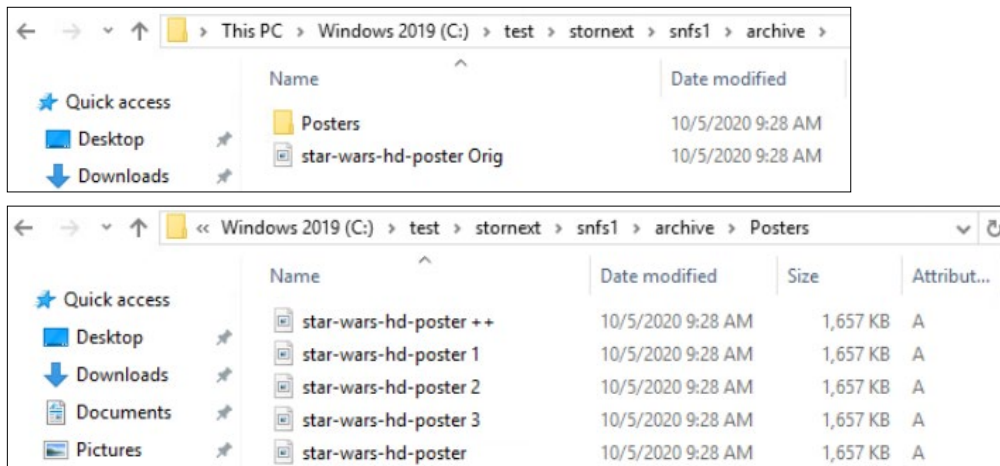
```

Mode                LastWriteTime         Length Name
-----
-a----            10/5/2020   9:28 AM         1696399 star-wars-hd-poster 1.jpg
-a----            10/5/2020   9:28 AM         1696399 star-wars-hd-poster Orig.jpg
-a----            10/5/2020   9:28 AM         1696399 star-wars-hd-poster 2.jpg
-a----            10/5/2020   9:28 AM         1696399 star-wars-hd-poster .jpg
-a----            10/5/2020   9:28 AM         1696399 star-wars-hd-poster 3.jpg
-a----            10/5/2020   9:28 AM         1696399 star-wars-hd-poster ++.jpg

PS C:\Users\Administrator>

```

Viewing the copied files from Windows Explorer, you can see that the filenames have been preserved, and the directory structure has been recreated, appended to the target directory root specified in the script (c:\test in this example).



This is a very simple example with less than a dozen lines of code, but hopefully it makes clear how easy it is to get started leveraging the StorNext File System's self-describing objects.

## CONCLUSION

With StorNext v6.4, self-describing objects make it possible to easily access and re-use data the StorNext File System has stored in the cloud. Objects may be accessed ad-hoc using tools such as AWS Console, Cyberduck, and AWS S3 Browser. Automated and batch operations are easily scripted using common development tools. Use cases ranging from simple cloud DR to complex hybrid and multi-cloud workflows are further enabled by this new feature.

### Credits and Disclaimer

The information in this document is for illustration purposes only, and is offered 'as-is', without any express or implied warranty. Quantum makes no representation or warranty as to the accuracy of the information and methods described in this document. Reliance and use of the information presented in this document is at the sole risk of the user; Quantum disclaims any and all liability.

Thank you to everyone who shares their PowerShell knowledge on the 'Net. A Perficent blog post by Omar Abuzaher (Mar 13, 2020) provided critical inspiration for a non-programmer to create the PowerShell script contained in this Tech Brief.