Microsoft Azure Blob Storage

Paths are specified as `remote:container` (or `remote:` for the `lsd` command.) You may put subdirectories in too, e.g.`remote:container/path/to/dir`.

Here is an example of making a Microsoft Azure Blob Storage configuration. For a remote called `remote`. First run:

```
rclone config
```

This will guide you through an interactive setup process:

```
No remotes found - make a new one
n) New remote
s) Set configuration password
q) Quit config
n/s/q> n
name> remote
Type of storage to configure.
Choose a number from below, or type in your own value
[snip]
XX / Microsoft Azure Blob Storage
   \ "azureblob"
[snip]
Storage> azureblob
Storage Account Name
account> account_name
Storage Account Key
key> base64encodedkey==
Endpoint for the service - leave blank normally.
endpoint>
Remote config
--------------------
[remote]
account = account_name
key = base64encodedkey==
endpoint =
--------------------
y) Yes this is OK
e) Edit this remote
d) Delete this remote
y/e/d> y
```

See all containers

```
rclone lsd remote:
```

Make a new container

```
rclone mkdir remote:container
```

List the contents of a container

```
rclone ls remote:container
```

Sync `/home/local/directory` to the remote container, deleting any excess files in the container.

```
rclone sync -i /home/local/directory remote:container
```

--fast-list

This remote supports `--fast-list` which allows you to use fewer transactions in exchange for more memory. See the rclone docs for more details.

Modified time

The modified time is stored as metadata on the object with the `mtime` key. It is stored using RFC3339 Format time with nanosecond precision. The metadata is supplied during directory listings so there is no overhead to using it.

Restricted filename characters

In addition to the default restricted characters set the following characters are also replaced:

Character	Value	Replacement
/	0x2F	／
\	0x5C	＼

File names can also not end with the following characters. These only get replaced if they are the last character in the name:

Character	Value	Replacement
.	0x2E	．

Invalid UTF-8 bytes will also be replaced, as they can't be used in JSON strings.

Hashes

MD5 hashes are stored with blobs. However blobs that were uploaded in chunks only have an MD5 if the source remote was capable of MD5 hashes, e.g. the local disk.

Authenticating with Azure Blob Storage

Rclone has 3 ways of authenticating with Azure Blob Storage:

Account and Key

This is the most straight forward and least flexible way. Just fill in the `account` and `key` lines and leave the rest blank.

SAS URL

This can be an account level SAS URL or container level SAS URL.

To use it leave `account`, `key` blank and fill in `sas_url`.

An account level SAS URL or container level SAS URL can be obtained from the Azure portal or the Azure Storage Explorer. To get a container level SAS URL right click on a container in the Azure Blob explorer in the Azure portal.

If you use a container level SAS URL, rclone operations are permitted only on a particular container, e.g.

```
rclone ls azureblob:container
```

You can also list the single container from the root. This will only show the container specified by the SAS URL.

```
$ rclone lsd azureblob:
container/
```

Note that you can't see or access any other containers - this will fail

```
rclone ls azureblob:othercontainer
```

Container level SAS URLs are useful for temporarily allowing third parties access to a single container or putting credentials into an untrusted environment such as a CI build server.

Standard Options

Here are the standard options specific to azureblob (Microsoft Azure Blob Storage).

--azureblob-account

Storage Account Name (leave blank to use SAS URL or Emulator)

- Config: account
- Env Var: RCLONE_AZUREBLOB_ACCOUNT
- Type: string
- Default: ""

--azureblob-service-principal-file

Path to file containing credentials for use with a service principal.

Leave blank normally. Needed only if you want to use a service principal instead of interactive login.

```
$ az sp create-for-rbac --name "<name>" \
  --role "Storage Blob Data Owner" \
  --scopes "/subscriptions/<subscription>/resourceGroups/<resource-group>/providers/Microsoft
  > azure-principal.json
```

See Use Azure CLI to assign an Azure role for access to blob and queue data for more details.

- Config: service_principal_file
- Env Var: RCLONE_AZUREBLOB_SERVICE_PRINCIPAL_FILE
- Type: string
- Default: ""

--azureblob-key

Storage Account Key (leave blank to use SAS URL or Emulator)

- Config: key
- Env Var: RCLONE_AZUREBLOB_KEY
- Type: string
- Default: ""

--azureblob-sas-url

SAS URL for container level access only (leave blank if using account/key or Emulator)

- Config: sas_url
- Env Var: RCLONE_AZUREBLOB_SAS_URL
- Type: string
- Default: ""

--azureblob-use-msi

Use a managed service identity to authenticate (only works in Azure)

When true, use a managed service identity to authenticate to Azure Storage instead of a SAS token or account key.

If the VM(SS) on which this program is running has a system-assigned identity, it will be used by default. If the resource has no system-assigned but exactly one user-assigned identity, the user-assigned identity will be used by default. If the resource has multiple user-assigned identities, the identity to use must be explicitly specified using exactly one of the msi_object_id, msi_client_id, or msi_mi_res_id parameters.

- Config: use_msi
- Env Var: RCLONE_AZUREBLOB_USE_MSI

- Type: bool
- Default: false

--azureblob-use-emulator

Uses local storage emulator if provided as 'true' (leave blank if using real azure storage endpoint)

- Config: use_emulator
- Env Var: RCLONE_AZUREBLOB_USE_EMULATOR
- Type: bool
- Default: false

Advanced Options

Here are the advanced options specific to azureblob (Microsoft Azure Blob Storage).

--azureblob-msi-object-id

Object ID of the user-assigned MSI to use, if any. Leave blank if msi_client_id or msi_mi_res_id specified.

- Config: msi_object_id
- Env Var: RCLONE_AZUREBLOB_MSI_OBJECT_ID
- Type: string
- Default: ""

--azureblob-msi-client-id

Object ID of the user-assigned MSI to use, if any. Leave blank if msi_object_id or msi_mi_res_id specified.

- Config: msi_client_id
- Env Var: RCLONE_AZUREBLOB_MSI_CLIENT_ID
- Type: string
- Default: ""

--azureblob-msi-mi-res-id

Azure resource ID of the user-assigned MSI to use, if any. Leave blank if msi_client_id or msi_object_id specified.

- Config: msi_mi_res_id
- Env Var: RCLONE_AZUREBLOB_MSI_MI_RES_ID
- Type: string
- Default: ""

--azureblob-endpoint

Endpoint for the service Leave blank normally.

- Config: endpoint
- Env Var: RCLONE_AZUREBLOB_ENDPOINT
- Type: string
- Default: ""

--azureblob-upload-cutoff

Cutoff for switching to chunked upload (<= 256MB). (Deprecated)

- Config: upload_cutoff
- Env Var: RCLONE_AZUREBLOB_UPLOAD_CUTOFF
- Type: string
- Default: ""

--azureblob-chunk-size

Upload chunk size (<= 100MB).

Note that this is stored in memory and there may be up to "--transfers" chunks stored at once in memory.

- Config: chunk_size
- Env Var: RCLONE_AZUREBLOB_CHUNK_SIZE
- Type: SizeSuffix
- Default: 4M

--azureblob-list-chunk

Size of blob list.

This sets the number of blobs requested in each listing chunk. Default is the maximum, 5000. "List blobs" requests are permitted 2 minutes per megabyte to complete. If an operation is taking longer than 2 minutes per megabyte on average, it will time out (source). This can be used to limit the number of blobs items to return, to avoid the time out.

- Config: list_chunk
- Env Var: RCLONE_AZUREBLOB_LIST_CHUNK
- Type: int
- Default: 5000

--azureblob-access-tier

Access tier of blob: hot, cool or archive.

Archived blobs can be restored by setting access tier to hot or cool. Leave blank if you intend to use default access tier, which is set at account level

If there is no "access tier" specified, rclone doesn't apply any tier. rclone performs "Set Tier" operation on blobs while uploading, if objects are not modified, specifying "access tier" to new one will have no effect. If blobs are in "archive tier" at remote, trying to perform data transfer operations from remote will not be allowed. User should first restore by tiering blob to "Hot" or "Cool".

- Config: access_tier
- Env Var: RCLONE_AZUREBLOB_ACCESS_TIER
- Type: string
- Default: ""

--azureblob-archive-tier-delete

Delete archive tier blobs before overwriting.

Archive tier blobs cannot be updated. So without this flag, if you attempt to update an archive tier blob, then rclone will produce the error:

```
can't update archive tier blob without --azureblob-archive-tier-delete
```

With this flag set then before rclone attempts to overwrite an archive tier blob, it will delete the existing blob before uploading its replacement. This has the potential for data loss if the upload fails (unlike updating a normal blob) and also may cost more since deleting archive tier blobs early may be chargable.

- Config: archive_tier_delete
- Env Var: RCLONE_AZUREBLOB_ARCHIVE_TIER_DELETE
- Type: bool
- Default: false

--azureblob-disable-checksum

Don't store MD5 checksum with object metadata.

Normally rclone will calculate the MD5 checksum of the input before uploading it so it can add it to metadata on the object. This is great for data integrity checking but can cause long delays for large files to start uploading.

- Config: disable_checksum
- Env Var: RCLONE_AZUREBLOB_DISABLE_CHECKSUM
- Type: bool
- Default: false

--azureblob-memory-pool-flush-time

How often internal memory buffer pools will be flushed. Uploads which requires additional buffers (f.e multipart) will use memory pool for allocations. This option controls how often unused buffers will be removed from the pool.

- Config: memory_pool_flush_time
- Env Var: RCLONE_AZUREBLOB_MEMORY_POOL_FLUSH_TIME
- Type: Duration

- Default: 1m0s

--azureblob-memory-pool-use-mmap

Whether to use mmap buffers in internal memory pool.

- Config: memory_pool_use_mmap
- Env Var: RCLONE_AZUREBLOB_MEMORY_POOL_USE_MMAP
- Type: bool
- Default: false

--azureblob-encoding

This sets the encoding for the backend.

See: the encoding section in the overview for more info.

- Config: encoding
- Env Var: RCLONE_AZUREBLOB_ENCODING
- Type: MultiEncoder
- Default: Slash,BackSlash,Del,Ctl,RightPeriod,InvalidUtf8

Limitations

MD5 sums are only uploaded with chunked files if the source has an MD5 sum. This will always be the case for a local to azure copy.

`rclone about` is not supported by the Microsoft Azure Blob storage backend. Backends without this capability cannot determine free space for an rclone mount or use policy `mfs` (most free space) as a member of an rclone union remote.

See List of backends that do not support rclone about See rclone about

Azure Storage Emulator Support

You can test rclone with storage emulator locally, to do this make sure azure storage emulator installed locally and set up a new remote with `rclone config` follow instructions described in introduction, set `use_emulator` config as `true`, you do not need to provide default account name or key if using emulator.