Working with S3

Introduction

We use garage, an open-source server compatible with Amazon's S3 API, hosted on our own infrastructure. Currently we store both documentation and binaries there; it may be used for other things down the line.

Configuring a client

You probably want to use rclone; it's friendly (to people who like the terminal) and not tightly bound to any specific storage service. You can also use the Amazon first-party S3 tooling, but this guide does not attempt to explain how.

To follow these steps, you will need to already have ssh access to the server garage runs on, which is `s3.lix.systems`. As of this writing, there is no guide about how to do that, but take a look at `services/ssh.nix` in the `web-services` repo and see whether it makes sense to you. Please feel free to write said guide and add it to this wiki. :)

Generating S3 credentials

Once you have ssh access, you will need to make s3 credentials. You can do it like this:

```
$ ssh root@s3.lix.systems
[root@cache:~]# garage key create some-key-name
Key name: some-key-name
Key ID: GKa653da6819c4140c3db9dfc5
Secret key: ab2b6106fbb7681517cba875c26c8ea99e281f113e2fd809decd6e524ebbc639

Can create buckets: false

Key-specific bucket aliases:

Authorized buckets:
```

(Don't worry - those aren't real keys there, nor were they ever! They're synthetic examples so you know what they look like.)

You'll want to choose a key name that helps the rest of us know whose it is and what it's used for. Don't just create a key called `some-key-name` by copying the example verbatim, it will be confusing clutter!

The most important criterion for a key name is that reading the name should let you answer the questions "is anyone still using this?" and "what will break if this key is deleted?" If you need naming inspiration you can see other people's key names with `garage key list`; in particular, keys meant for individual use should probably start with your username.

Before you sign out of the server, also make sure to grant the key the permissions you need. For example, if you need to work with the `docs` bucket, do:

```
[root@cache:~]# garage bucket allow --read --write docs --key irenes-temp-delete
New permissions for GKa653da6819c4140c3db9dfc5 on docs: read true, write true, owner false.
```

You can see what buckets exist by doing `garage bucket list`.

The "Key ID" and "Secret key" values from the key you generated are what you'll need in the next step. Make sure you have them; there's no way to look up the secret part later.

Configuring your client (probably rclone)

You may find it useful to reference the garage documentation on this.

There are two ways to configure rclone, either of which will work. The one Irenes recommend is to put the credentials directly into the rclone configuration (it has its own tooling for securing them, which you can set up if you want). The other way is to let rclone read them out of the config file used by Amazon's first-party tooling. Either will work; using the AWS config file is a little harder to figure out what you did later, if you happen to forget. Also, if Lix infrastructure isn't the only S3 service you use on a regular basis, the rclone config is probably a better place to keep track of everything because AWS profiles are a pain to use.

If you're doing it the Irenes way, you can either run `rclone config` and go through the prompts, or just prepare a config file by hand.

Here's a sample rclone.config:

```
[lix]
type = s3
provider = Other
env_auth = false
endpoint = s3.lix.systems
region = garage
access_key_id = GKa653da6819c4140c3db9dfc5
```

```
secret_access_key = ab2b6106fbb7681517cba875c26c8ea99e281f113e2fd809decd6e524ebbc639
```

For more information about where to put this config file, see `man rclone`; it's likely that `~/.config/rclone/rclone.conf` is the right place.

Please notice that this example file uses `lix` as the name of the rclone "remote". That means that, when interacting with it, you'll use paths like `lix:` to refer to the entire thing, or `lix:docs/` to refer to the root of the bucket named `docs`, and so on. You can use any name you find convenient for the remote, it doesn't have to be `lix`, but this document will assume it's that. If you think you might have done this configuration already but don't remember what you called the remote, do `rclone listremotes`.

If you're going through the interactive configuration, choose the generic S3-compatible service as the type of service. For the endpoint, write in `s3.lix.systems`, and for the region, write in `garage`. If you leave region blank you'll get weird errors about us-east-1, but we're not Amazon and we don't have a global network of highly-redundant data centers, so don't leave it blank. :)

If you're storing credentials in the AWS config file, everything is pretty similar except you'll need to prepare `~/.aws/credentials` yourself, and tell rclone to use it; the `rclone config` wizard has options for that. The easy way is to use the `default` profile in the AWS credentials file; otherwise you'll have to make sure your environment sets `AWS_PROFILE`, since rclone has no option to manage that itself.

Copying files into and out of s3

If you're using `rclone`, you may find it useful to do `rclone ls lix:` to get a sense of what's there. This will probably become increasingly bad advice as our usage of S3 grows! :)

Notice that the first path component in this output is the bucket name, so ie. a file named `index.html` at the root of the `docs` bucket is listed as `docs/index.html` in this view. That is also how you will refer to it from the command line. Other S3 clients have different conventions in this regard, so if you're using something else, check its upstream documentation.

If you have a local file `index.html` and you want to overwrite the remote `docs/index.html` with it, do `rclone copy index.html lix:docs/`. You have to give a directory prefix, not a filename, for the second part.

In general, the `rclone` CLI lets you intermingle local and remote paths, so pay close attention to the colons. `lix:something` is a remote path, `something` is a local one. If you lose track of this you will end up sad.

For any other `rclone`-related questions, `rclone --help` and `man rclone` are good references.

Happy filing!

Revision #2
Created 5 July 2024 22:10:16 by irenes
Updated 5 July 2024 22:52:09 by irenes