Safety First

Using clickhouse-backup for ClickHouse® Backup and Restore

Eugene Klimov
Robert Hodges
https://altinity.com

The Altinity Safety Cat

A brief message from our sponsor…

Robert Hodges

Database geek with 30+ years on DBMS. Kubernaut since 2018. Day job: Altinity CEO

Eugene Klimov

Clickhouse-backup maintainer with 20+ years in software. Day job: Cloud Engineer

ClickHouse support and services including Altinity.Cloud
Authors of Altinity Kubernetes Operator for ClickHouse, Altinity clickhouse-backup and other open source projects

Why do we back up databases?

Backups solve a number of important problems

- Catastrophic failures that delete all data
- Accidental deletion of a database or table
- Debugging problems using production data
- Upgrade testing prior to schema or version change
- Loading schema and configuration for new installations

Welcome to ClickHouse, a real-time analytic database

Understands SQL

Runs on bare metal to cloud

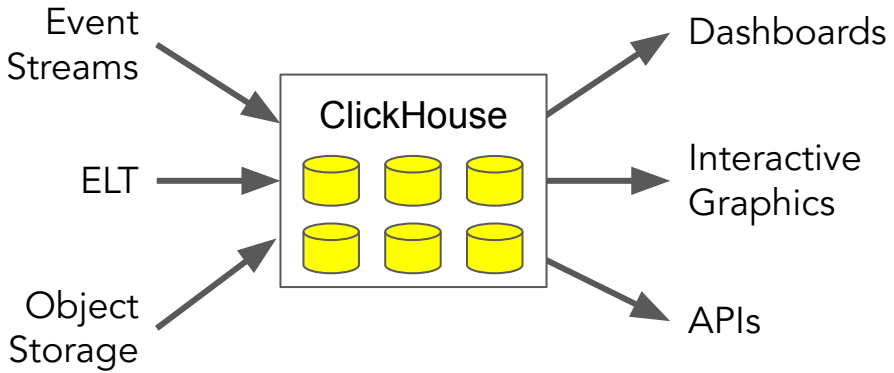
Shared nothing architecture

Stores data in columns

Parallel and vectorized execution

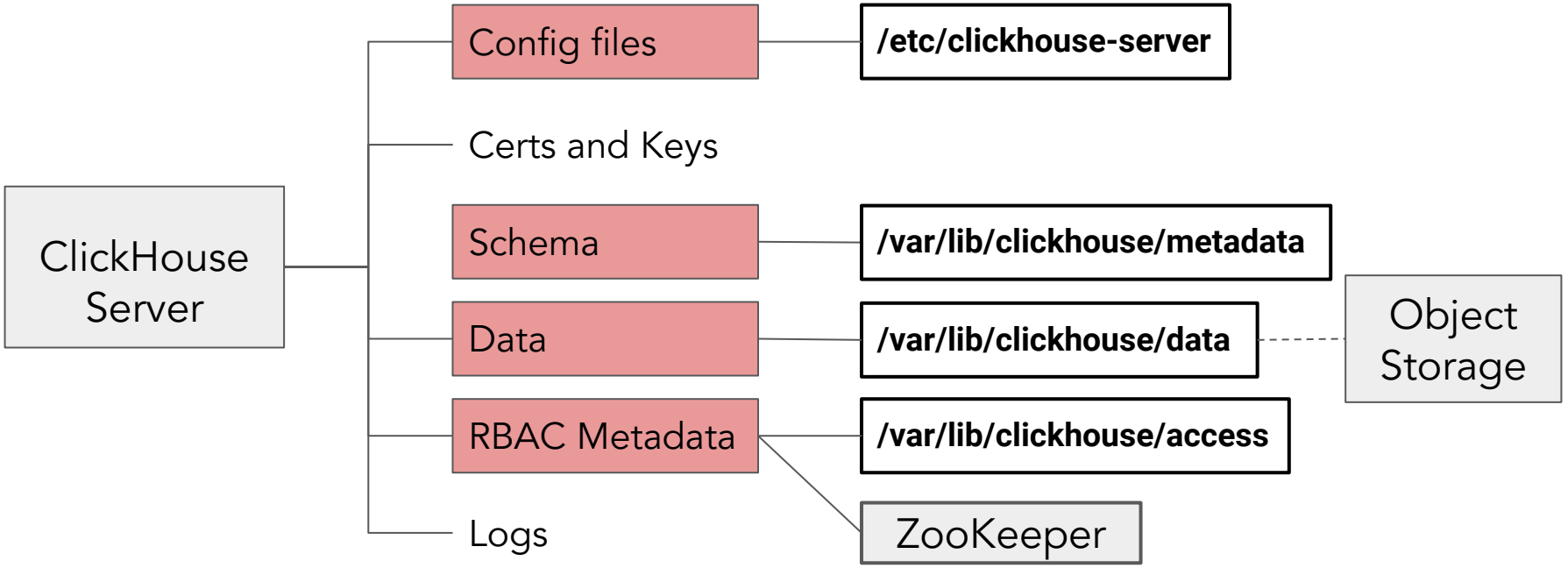
Scales to many petabytes

Is Open source (Apache 2.0)

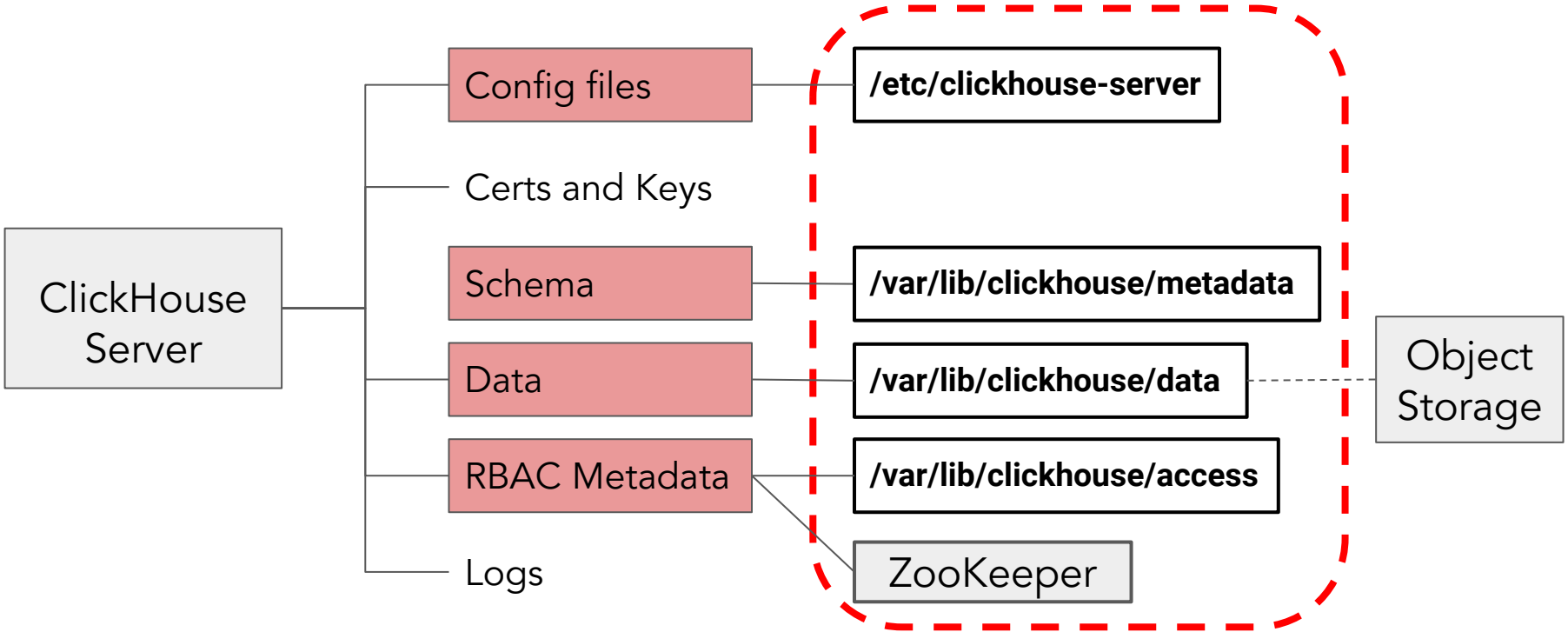


It's the core engine for low-latency analytics

What do we need to protect in ClickHouse?



What do we need to protect in ClickHouse?



Common backup/restore options for ClickHouse

Tool	Description	Configs	Schema	Data	RBAC
Replication	Use ReplicatedMergeTree	☐	☐	✅	☐
ClickHouse Copier	Works with ZooKeeper to copy cluster data	☐	☐	✅	☐
Altinity clickhouse-backup project	Standalone backup utility for all ClickHouse versions	✅	✅	✅	✅
ClickHouse BACKUP & RESTORE	Built-in SQL operations in ClickHouse (recent versions)	☐	✅	✅	☐

Introducing clickhouse-backup

The clickhouse-backup utility at a glance

Language
Golang

GitHub Proejct
https://github.com/Altinity/clickhouse-backup

GitHub Stars
1040

License
Apache 2.0

Distributions
• RPM - aarch64, x86_64 • Mac OS X Tarball - amd64 • Linux Tarball - amd64, arm64 • Debian - amd64, arm64 • Docker - amd64, arm64

Original Author
Alex Akulov

Maintainer
Eugene Klimov

Step 1: Install clickhouse-backup on ClickHouse host

```
# Grab the latest release from GitHub.
wget
https://github.com/Altinity/clickhouse-backup/releases/downloa
d/v2.4.2/clickhouse-backup-linux-amd64.tar.gz
# Unpack.
tar -xf clickhouse-backup-linux-amd64.tar.gz
# Install.
sudo install -o root -g root -m 0755 \
 build/linux/amd64/clickhouse-backup /usr/local/bin
# Try it out.
/usr/local/bin/clickhouse-backup -v
```

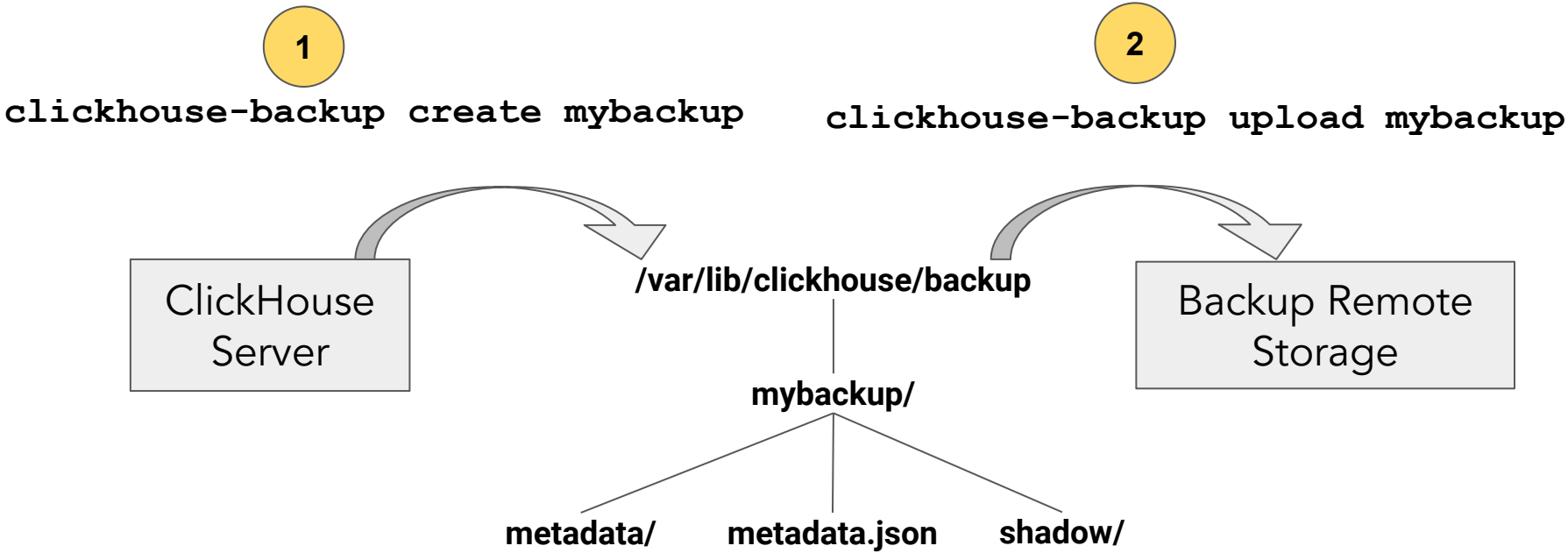
Step 2: Prepare config.yml file

```
# Grab the latest release from GitHub.
sudo -u clickhouse mkdir /etc/clickhouse-backup
sudo -u clickhouse clickhouse-backup \
 default-config > /etc/clickhouse-backup/config.yml
sudo -u vi /etc/clickhouse-backup/config.yml
```

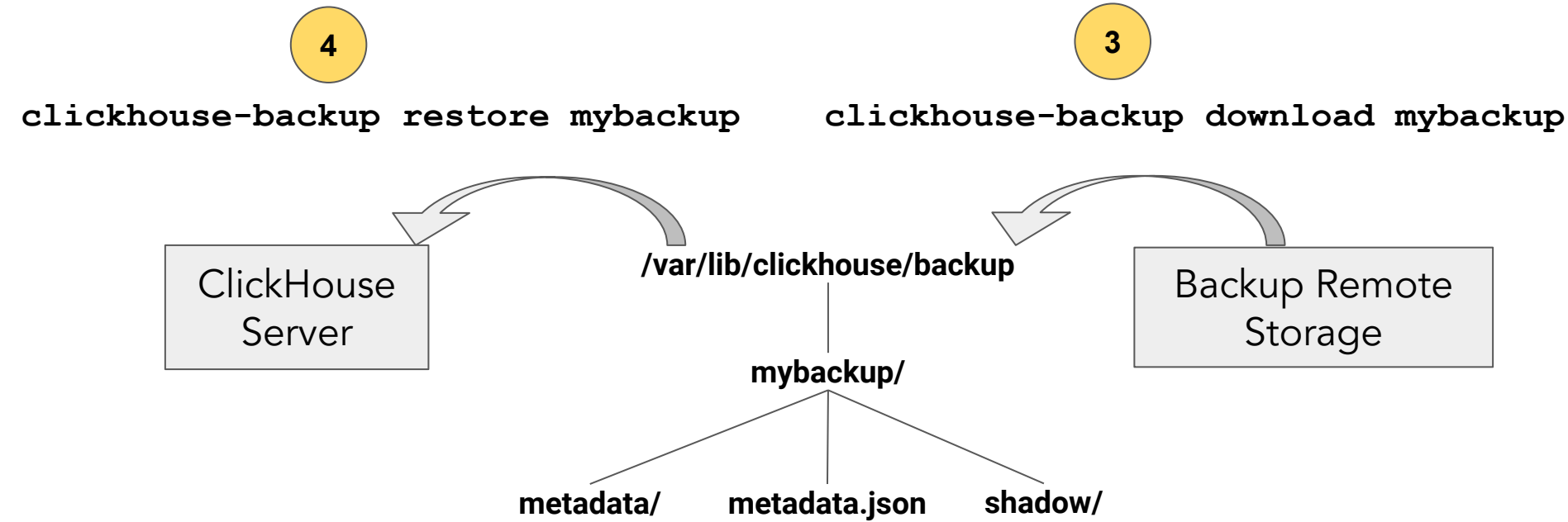
Fill in values in sections:

- general:
- clickhouse:
- s3:

Step 3: Now let's learn how to create a backup



Step 4: And how to restore it



Backing up and restoring with clickhouse-backup

DEMO TIME!

Examples of backup commands

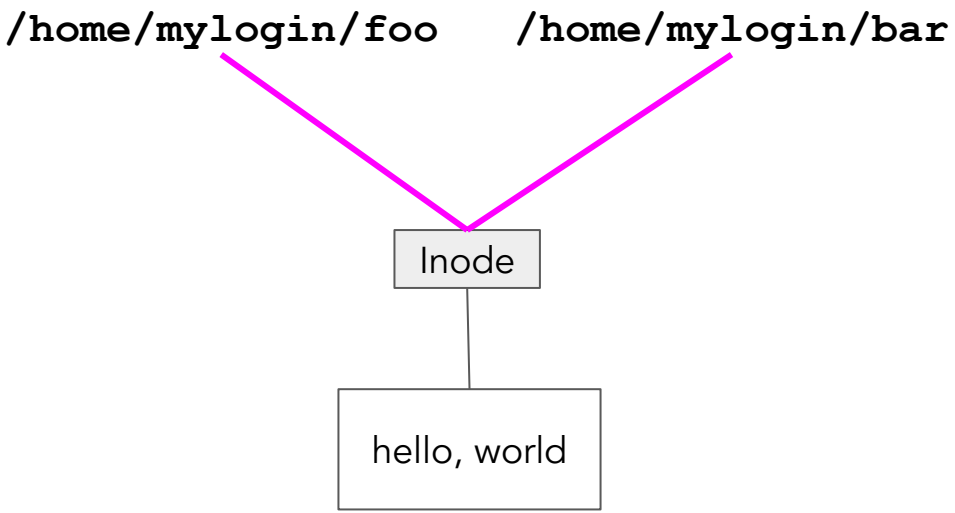
```
# Back up everything locally.
sudo -u clickhouse clickhouse-backup create mybackup \
 --rbac --configs

# Back up a single table locally.
sudo -u clickhouse clickhouse-backup create \
 mybackup_table_local -t default.ex2

# Back up and upload a database to remote backup storage.
sudo -u clickhouse clickhouse-backup create_remote \
 mybackup_database_remote -t 'default.*'
```

Quick primer on hard links

```
$ echo "hello, world" > foo
$ ln foo bar
$ ls --inode foo bar
4206300 bar  4206300 foo
$ cat bar
hello, world
$ rm foo
$ ls --inode bar
4206300 bar
```

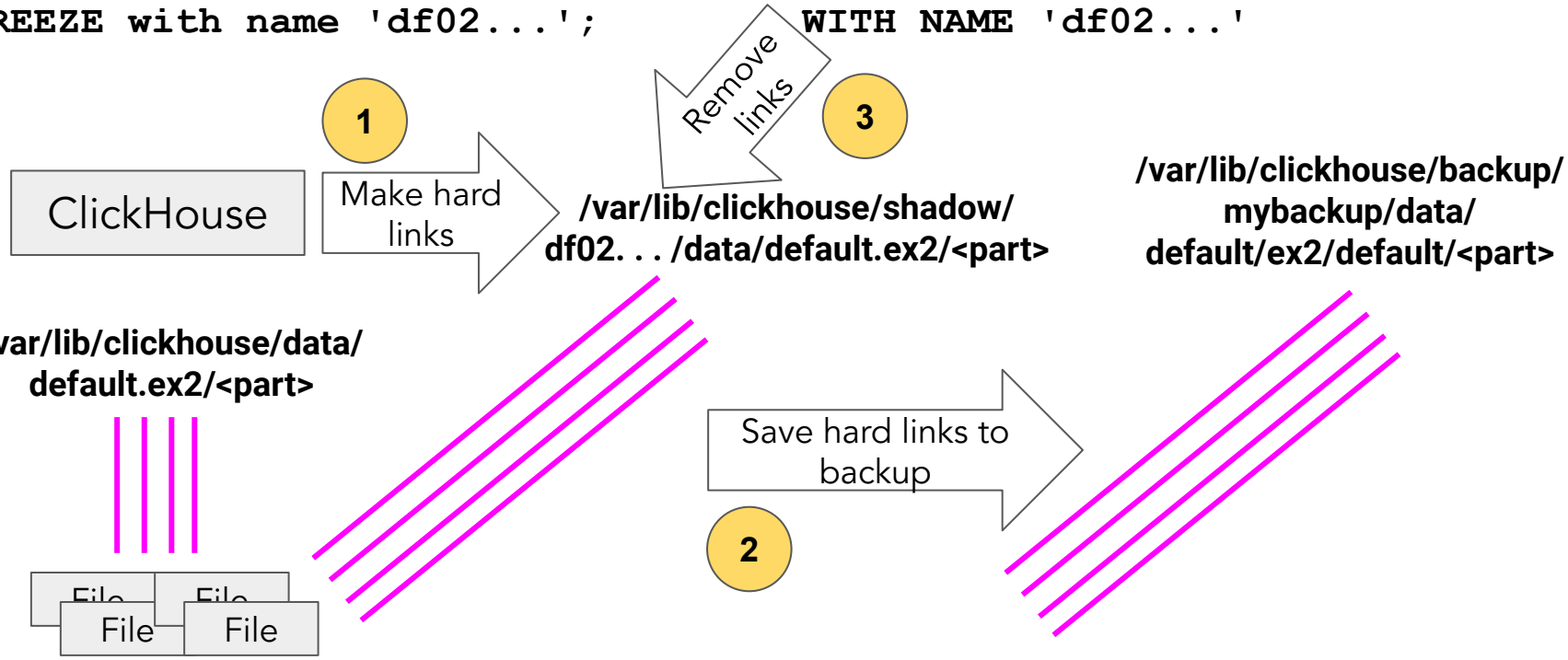


Tip: Cross device and remote hard links are not possible. Hard links only work within a single file system.

How does the backup command work under the covers?

```
ALTER TABLE default.ex2
FREEZE with name 'df02...';
```

```
ALTER TABLE default.ex2 UNFREEZE
WITH NAME 'df02...'
```



Examples of restore commands

```
# Restore all data from already downloaded backup.
sudo -u clickhouse clickhouse-backup restore mybackup

# Restore a single table from local backup.
sudo -u clickhouse clickhouse-backup restore \
 mybackup -t default.ex2

# Download and restore a single database.
sudo -u clickhouse clickhouse-backup restore_remote \
 mybackup -t 'default.*'
```

So how does restore work?

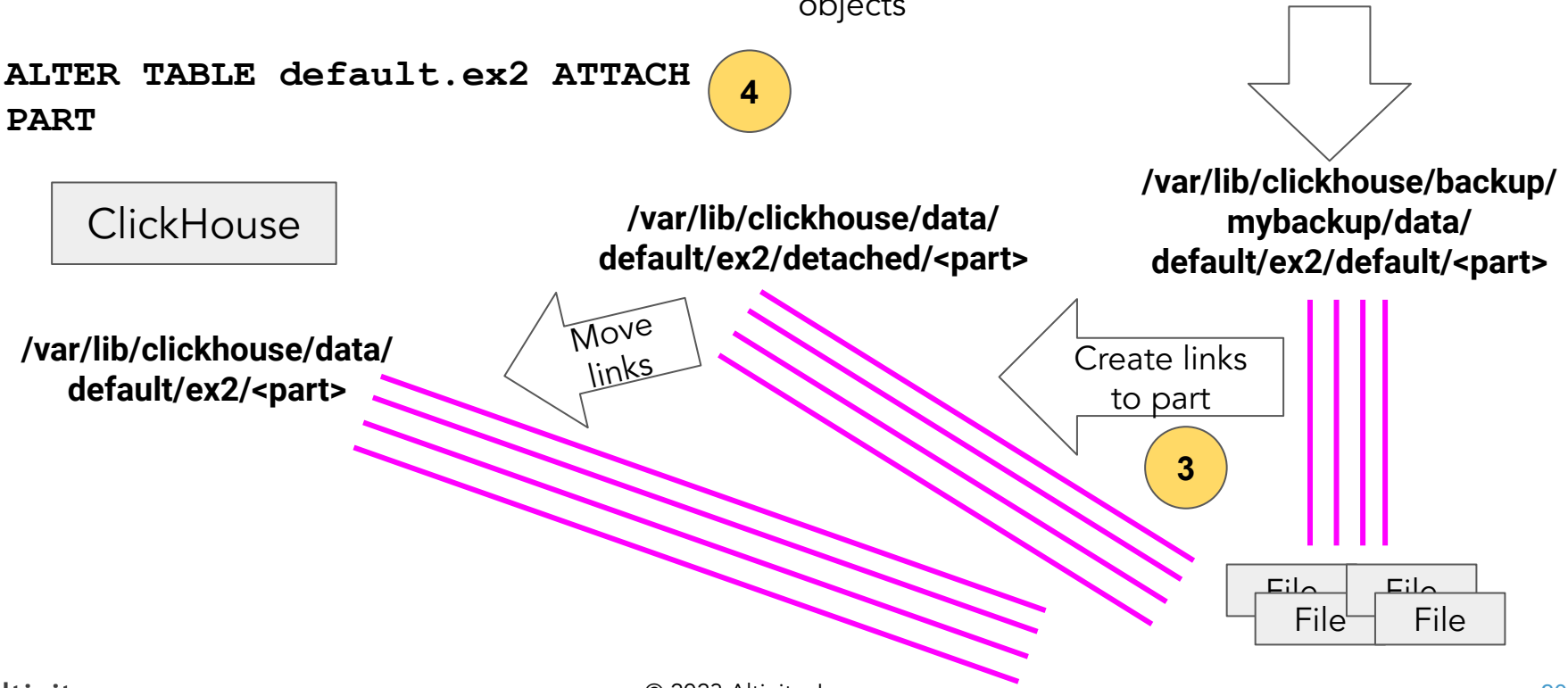
```
CREATE TABLE default.ex2 ...

ALTER TABLE default.ex2 ATTACH
PART
```

2 Create the tables and other schema objects

1 Download backup and write files

4



More restore commands

```
# Restore everything: schema, data, users, config files.
sudo -u clickhouse clickhouse-backup \
  restore mybackup --rbac --configs

# Restore only configuration files.
sudo -u clickhouse clickhouse-backup restore \
 mybackup --configs-only

# Restore only RBAC metadata.
sudo -u clickhouse clickhouse-backup restore \
 mybackup --rbac-only
```

Tip: Server restart required for these commands

Managing backups

```
# Listing your backups.
sudo -u clickhouse clickhouse-backup list
sudo -u clickhouse clickhouse-backup list local
sudo -u clickhouse clickhouse-backup list remote

# Deleting backups.
sudo -u clickhouse clickhouse-backup delete local mybackup
sudo -u clickhouse clickhouse-backup delete remote mybackup
```

Creating an incremental backup

```
# Create a full backup to get things started.
sudo -u clickhouse clickhouse-backup create_remote \
  full_backup -t 'default.*'
sudo -u clickhouse clickhouse-backup delete \
  local full_backup

# Now create an incremental backup.
sudo -u clickhouse clickhouse-backup \
  create_remote --diff-from-remote=full_backup \
  incremental_backup1 -t 'default.*'
sudo -u clickhouse clickhouse-backup delete \
  local incremental_backup1
```

Only the differences with remote backup are uploaded!

Restoring from an incremental backup.

```
# Restore test1 from the latest incremental backup.
sudo -u clickhouse clickhouse-backup \
  restore_remote incremental_backup1 \
  -t 'default.test1'
```

Command traverses all backups to find data

Advanced Topics

Managing backup storage

How to clean up orphan data in /var/lib/clickhouse/shadow:

```
sudo -u clickhouse clickhouse-backup clean
```

How to clean up a broken remote backup (missing or bad metadata.json file):

```
sudo -u clickhouse clickhouse-backup clean_remote_broken
```

How to keep backups from accumulating using automatic retention:

```
general:
  allow_empty_backups: false
  backups_to_keep_local: 1
  backups_to_keep_remote: 1
```

Tips for managing remote storage

Handling remote storage in sharded clusters:

- Use macros (e.g. {shard}) in `path` section of remote storage settings to avoid deleting backup from other shards by accident when computing backup retention

How to enable parallel upload and download for object storage in config.yml:

```
general:
  download_concurrency: 3
  upload_concurrency: 3
```

More on incremental backups

- A minimal increment item for calculation of increment is data part name
- Increment will grow if you frequently use OPTIMIZE … FINAL or ALTER TABLE … UPDATE / DELETE
  - They make a lot of new data parts for exists data
- Increment calculates only in upload stage
- Create command always create full backup
  - Parts which are present in base backup marked as required in metadata/db/table.json
- During download required parts will download from base remote backup to local disk.
  - ClickHouse-backup creates hard links in backup_name/shadow folder to make it complete

Enabling the REST API for backups

It's easy! clickhouse-backup can work as a daemon with REST API

```
clickhouse-backup server
```

Check out the api: section in config.yml. Tips:

- **`enable_metrics: true`** - /metrics endpoint with Prometheus format
- **`enable_pprof: true`** - /debug/pprof endpoints for memory heap and CPU profiling
- **`create_integration_tables: true`** - create system.backup_list system.backup_actions

An example REST request

```
$ curl http://localhost:7171/backup/list |jq
{
  "name": "my_backup",
  "created": "2023-10-25 02:48:25",
  "size": 828848,
  "location": "remote",
  "required": "",
  "desc": "tar, regular"
}
```

Working with REST API via SQL

Check out the **`api:`** section in config.yml
`create_integration_tables: true` will create **`system.backup_list`**
`system.backup_actions` tables

INSERT INTO system.backup_actions(comand)
VALUES('create_remote backup_name'),('delete local backup_name');

SELECT * FROM system.backup_actions;

SELECT * FROM system.backup_list;

More things to learn about with clickhouse-backup

- Backup and restore on a sharded cluster
  - See Examples.md#how-to-make-backup--restore-sharded-cluster
- Converting MergeTree to ReplicatedMergeTree
  - See Examples.md#how-to-convert-mergetree-to-replicatedmergetree
- Using shell scripts for (list, upload, download, delete) to integrate any remote storage type **`remote_storage: custom`**
  - See https://github.com/Altinity/clickhouse-backup/tree/master/test/integration/

And a final tip for health and happiness…

Test your backups

before you need them!

Wrap-up

Roadmap

- Better Support for incremental backups
- Backing up MergeTree on S3 object storage (now in beta)
- Add support for embedded BACKUP/RESTORE incremental backups

Current detailed backlog:

https://github.com/Altinity/clickhouse-backup/milestones

Help us to make the clickhouse-backup project better!!!

https://github.com/Altinity/clickhouse-backup

Try it out!

Tell your friends!

Log issues!

Send us pull requests!

Summary

- Backups solve problems from disaster recovery to making test copies
- Clickhouse-backup is well tested and rich in features:
  - Full server backups including schema, data, RBAC metadata, and config files
  - Incremental backups
  - Many remote storage options
  - Retentions
  - Server API
- Clickhouse-backup uses hard linking tricks to back up and restore MergeTree
- Future releases will handle backup of S3-backed MergeTree (in beta)

Talk sample code: https://github.com/Altinity/clickhouse-sql-examples/tree/main/clickhouse-backup

Thank you!

Eugene Klimov - Robert Hodges
https://altinity.com

Altinity.Cloud
Altinity Stable Builds
Altinity Kubernetes Operator for ClickHouse

The Altinity
Safety Cat