CYVERSE®

Accessing iRODS in Kubernetes

Go-iRODSClient, iRODS FUSE Lite, and iRODS CSI Driver

Illyoung Choi
CyVerse
iychoi@cs.arizona.edu

John H. Hartman
Dept. of Computer Science
University of Arizona
jhh@cs.arizona.edu

Edwin Skidmore
CyVerse
edwin@cyverse.org

June 8, 2021
iRODS User Group Meeting

Kubernetes + iRODS

Kubernetes is a *de facto standard* platform for cloud native architecture

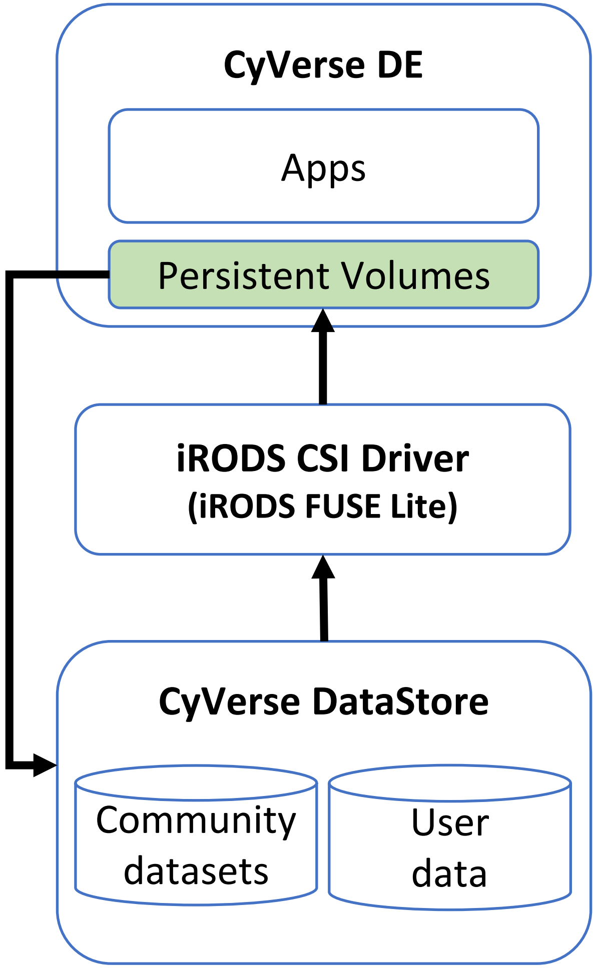
Existing iRODS access methods in Kubernetes?

- Download/upload data before/after jobs
  via *iCommands (iget/iput) or Python-iRODSClient*
- Mount iRODS data in Pods
  via *iRODS FUSE, WebDAV(davfs2), or NFS*

iRODS Integration with CyVerse DE (Discovery Environment)

- Provide DE Apps with access to scientific reference datasets and user data on DataStore (iRODS-based storage service)

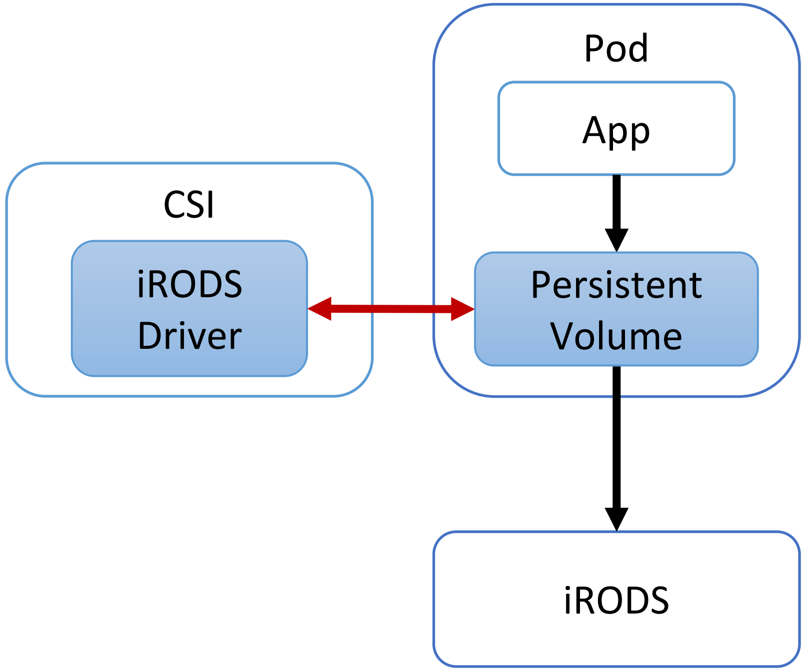
- Via a **Container Storage Interface (CSI) Driver**



The primary goal

iRODS CSI Driver

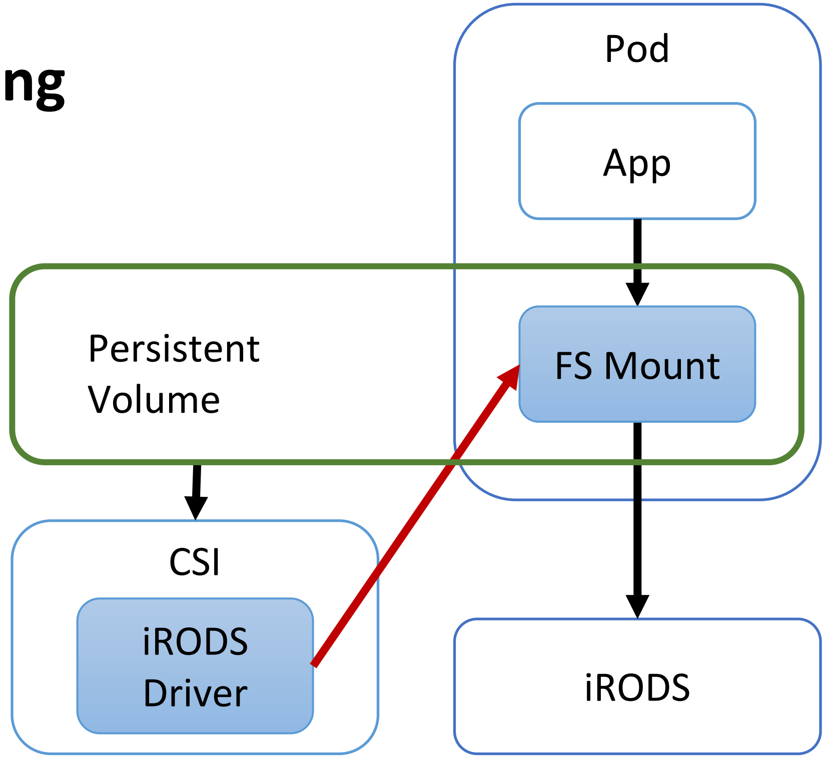
- Provides access to iRODS in Kubernetes in a standard way (via ***Persistent Volumes***)
- Extends storage supports via **Container Storage Interface (CSI)**



The secondary goal

iRODS FUSE Lite: Re-implementation of iRODS FUSE in Golang

- CSI relies on **Linux file system mount**
- Three clients available for iRODS data mount:
  - **Davfs2**: supports web-caches, requires DavRODS
  - **NFS client**: require NFSRODS, host authentication (less secure)
  - **iRODS FUSE**: not maintained, **communicates with iRODS directly, customizable**
- Written in **Golang** for rapid development



The tertiary goal

Go-iRODSClient

- An **iRODS API library** for **Golang**
- Written in pure Golang
- **More Portable** than an existing library (*GoRODS*)
  - GoRODS: a Golang binding for iRODS C API library

Why Golang?

Portability

- Easy to build executables for different architectures
- No dependency issues if written in pure Golang

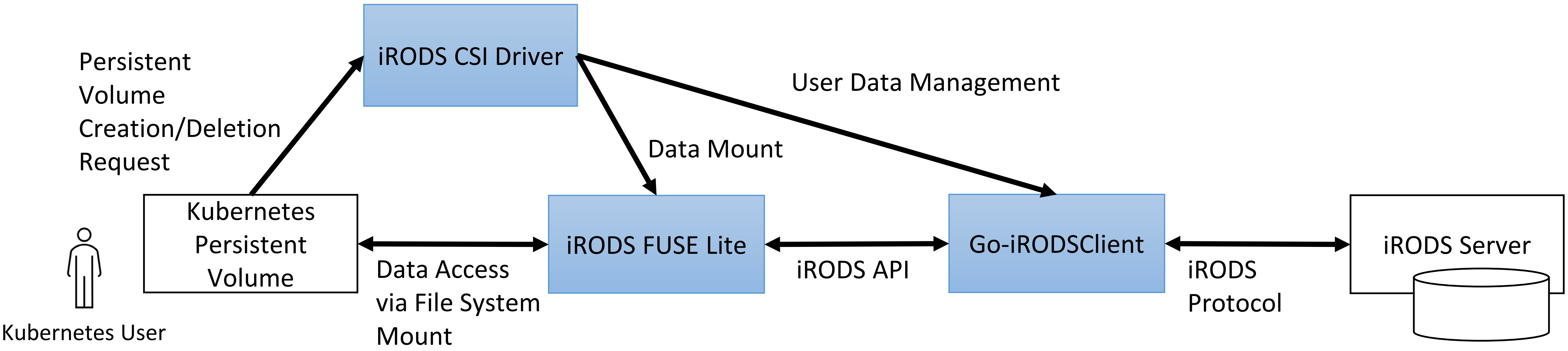
Productivity

- Garbage collection, rich library supports, simple syntax, and ETC

de facto standard

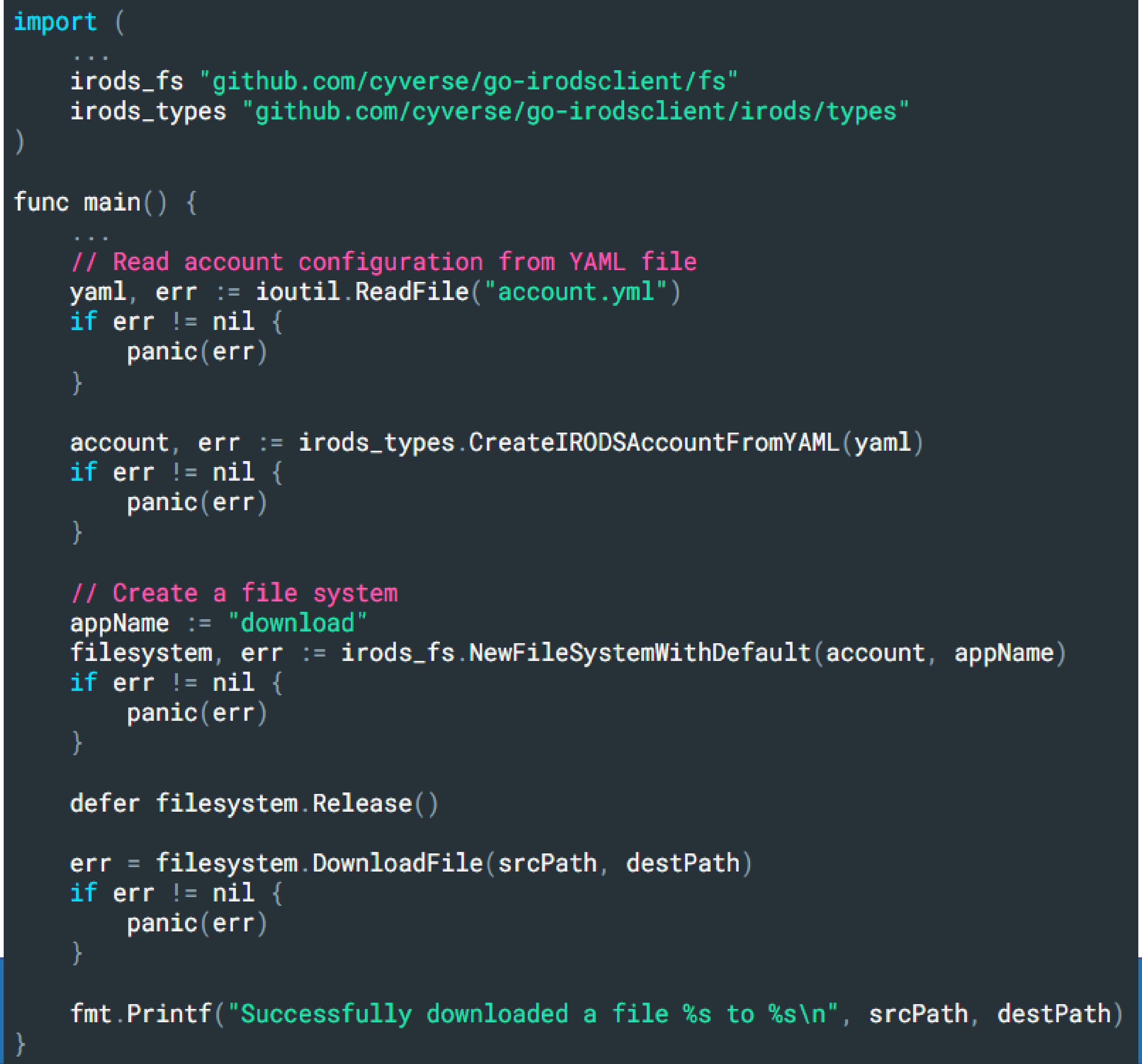
- The most widely used language in Kubernetes

Kubernetes integration with iRODS



Go-iRODSClient: Code Examples

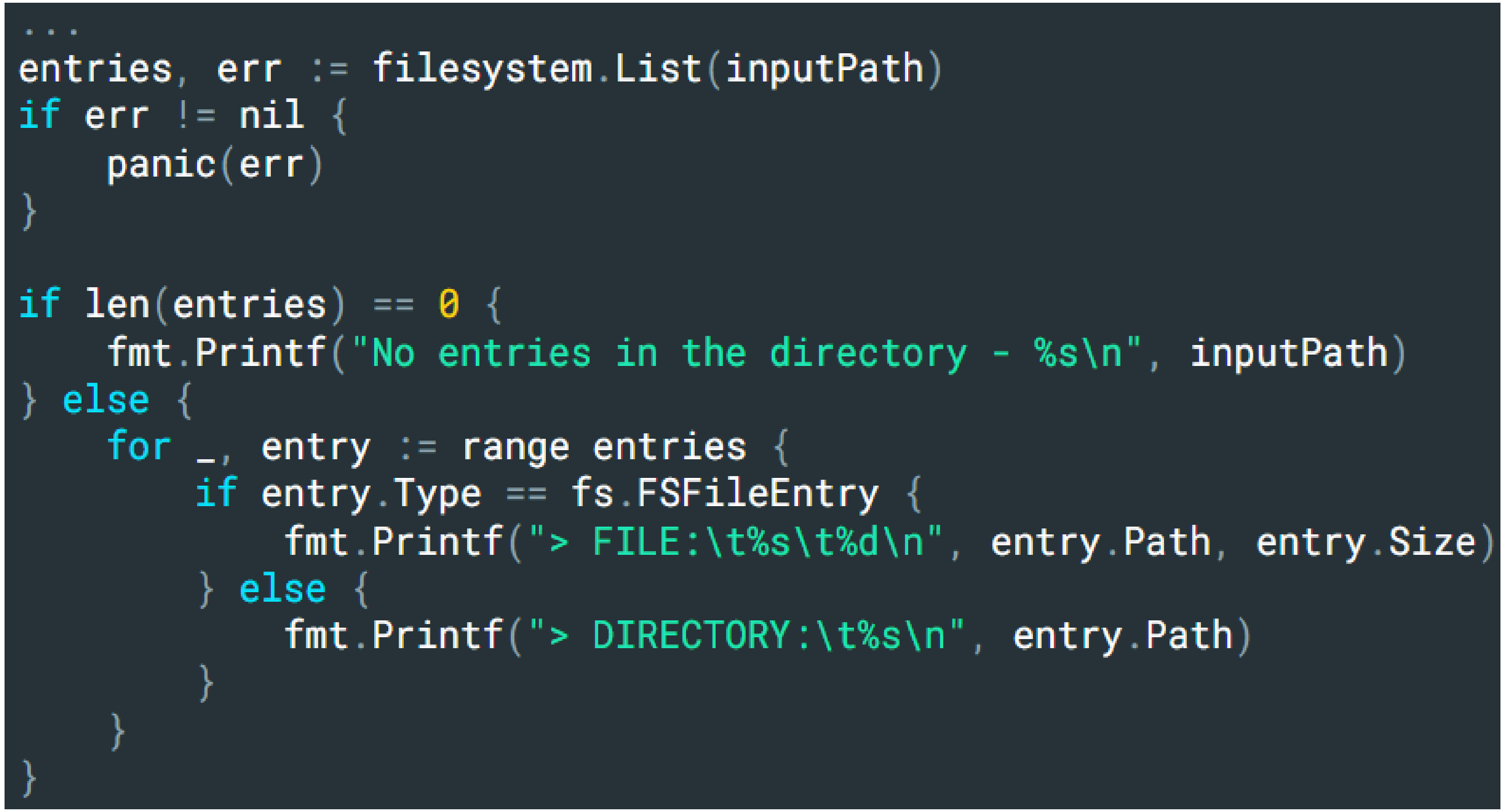
File Download



```go
import (
    ...
    irods_fs "github.com/cyverse/go-irodsclient/fs"
    irods_types "github.com/cyverse/go-irodsclient/irods/types"
)

func main() {
    ...
    // Read account configuration from YAML file
    yaml, err := ioutil.ReadFile("account.yml")
    if err != nil {
        panic(err)
    }

    account, err := irods_types.CreateIRODSAccountFromYAML(yaml)
    if err != nil {
        panic(err)
    }

    // Create a file system
    appName := "download"
    filesystem, err := irods_fs.NewFileSystemWithDefault(account, appName)
    if err != nil {
        panic(err)
    }

    defer filesystem.Release()

    err = filesystem.DownloadFile(srcPath, destPath)
    if err != nil {
        panic(err)
    }

    fmt.Printf("Successfully downloaded a file %s to %s\n", srcPath, destPath)
}
```

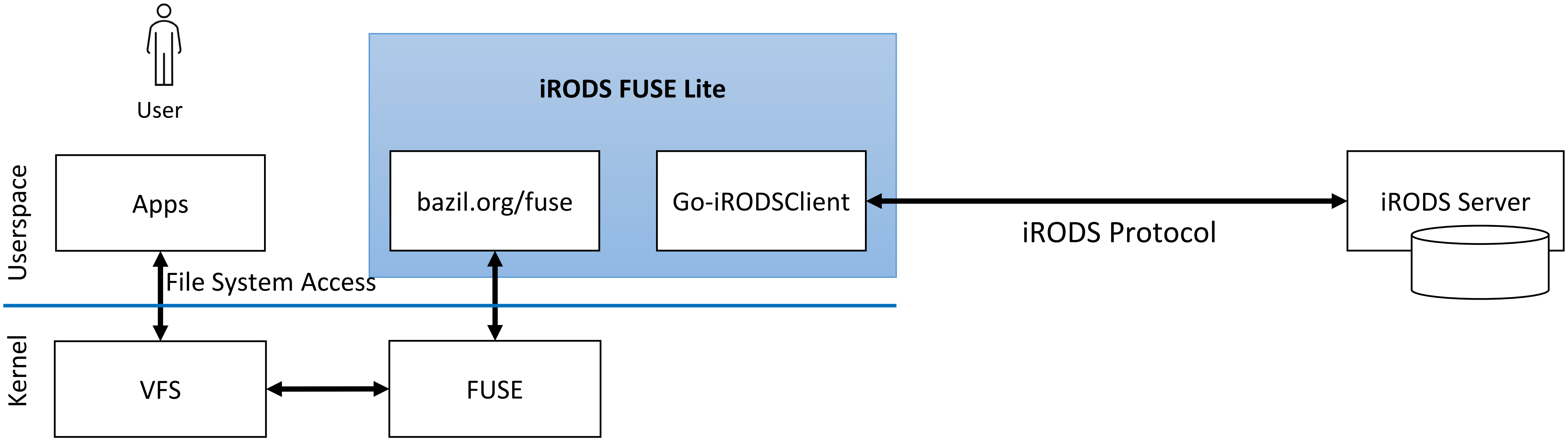
Go-iRODSClient: Code Examples

Directory Listing



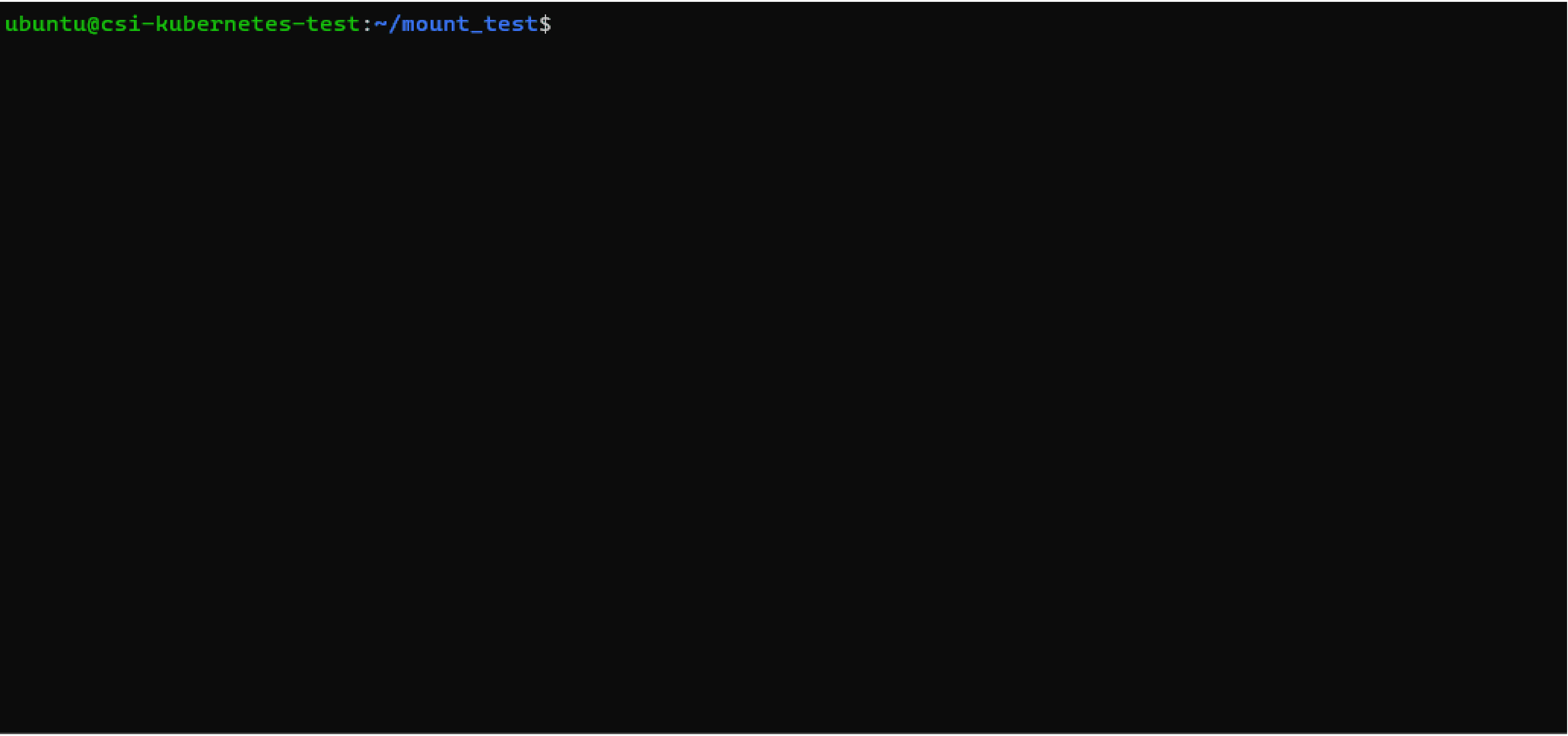
```
...
entries, err := filesystem.List(inputPath)
if err != nil {
    panic(err)
}

if len(entries) == 0 {
    fmt.Printf("No entries in the directory - %s\n", inputPath)
} else {
    for _, entry := range entries {
        if entry.Type == fs.FSFileEntry {
            fmt.Printf("> FILE:\t%s\t%d\n", entry.Path, entry.Size)
        } else {
            fmt.Printf("> DIRECTORY:\t%s\n", entry.Path)
        }
    }
}
```

iRODS FUSE Lite: File System Mount



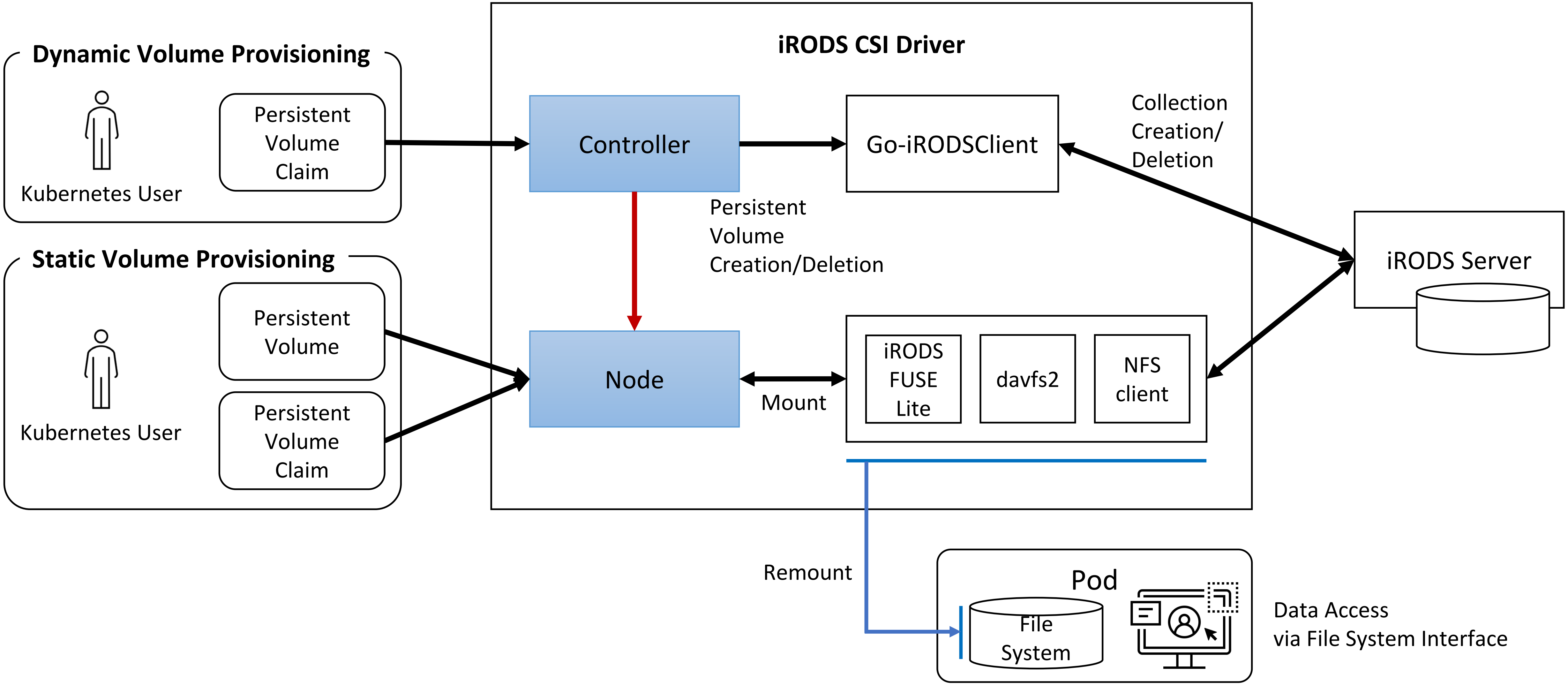
iRODS FUSE Lite: Quick Demo

Mounting an iRODS Collection

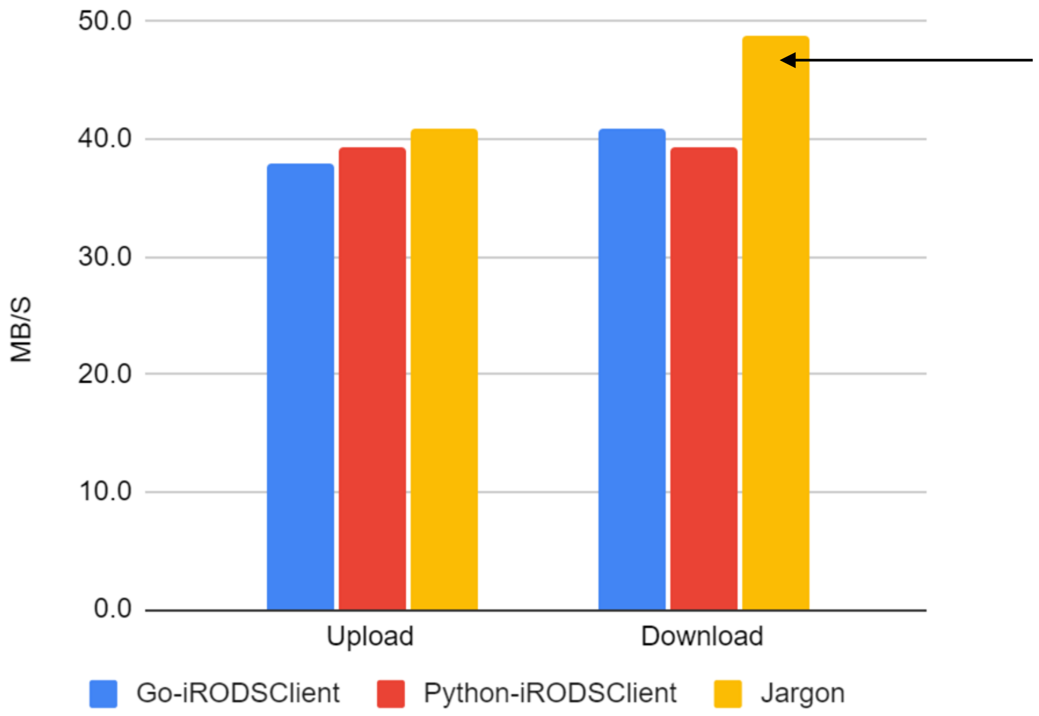


```
ubuntu@csi-kubernetes-test:~/mount_test$
```

iRODS CSI Driver: Persistent Volume Mount



I/O Performance of API libraries



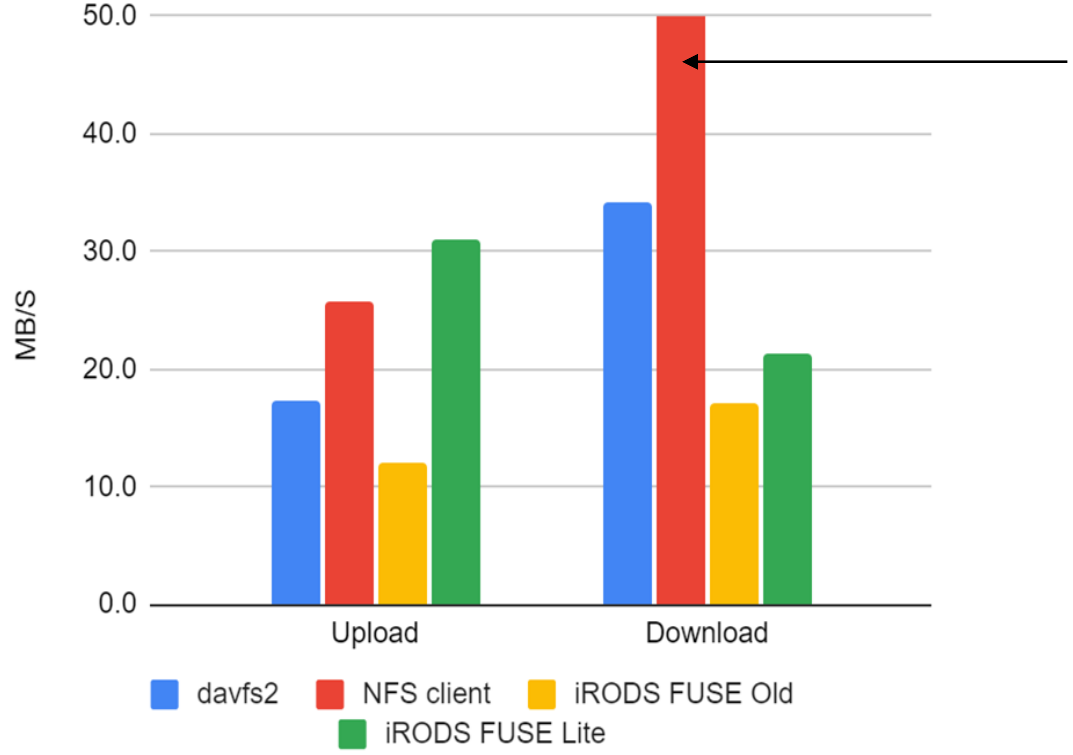
Stream read

Future Work:

- Implement Stream Read
- Implement Parallel Data Transfer API

Note: did not use parallel put/get APIs

I/O Performance of FS Clients



Stream read via Jargon

Note: tested using "cp"

Future Work:

- Implement Stream Read
- Make it as fast as NFS client

Integration with CyVerse services

Go-iRODSClient

- Used by **DataWatch** – a data monitoring service for DataStore (iRODS)
- To retrieve detailed information of Data Objects/Collections updated

iRODS CSI Driver

- Integrated with **Discovery Environment (DE)**
- To provide DE Apps with access to scientific reference datasets and user data stored on DataStore (iRODS)

Conclusion

Working on a project in Golang?

⇒ **Go-iRODSClient**

Want to mount iRODS data on file system?

⇒ **iRODS FUSE Lite**

Want to access iRODS in Kubernetes Pods?

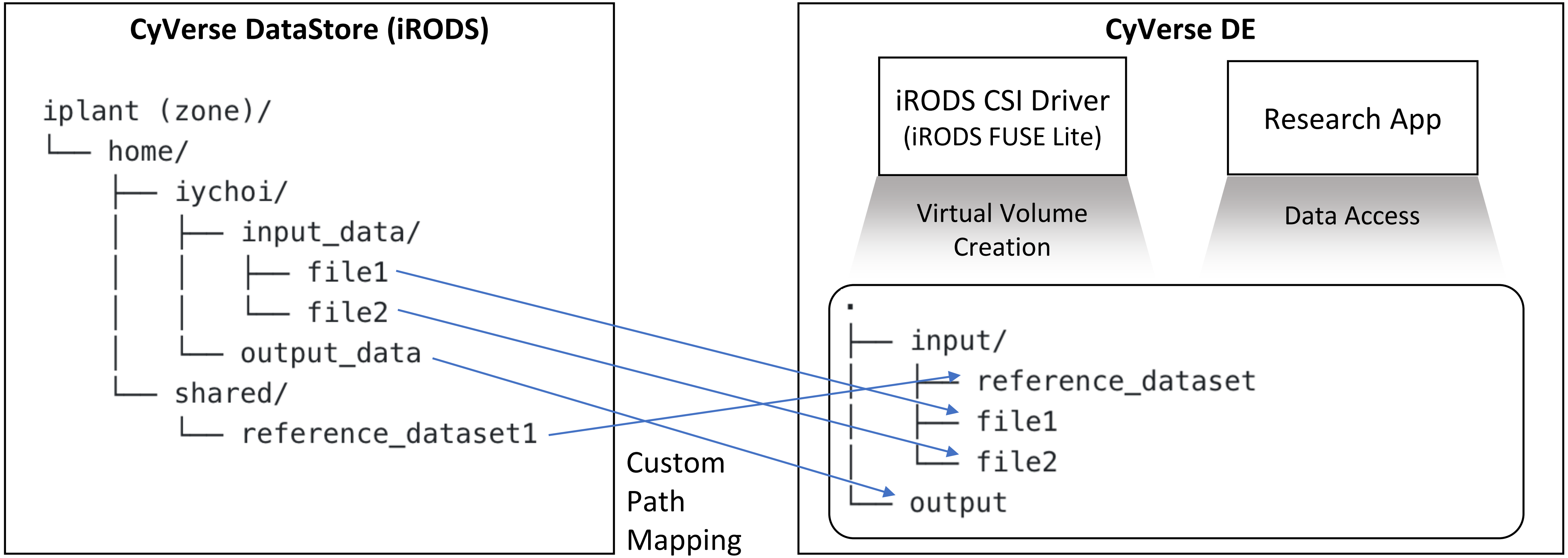
⇒ **iRODS CSI Driver**

Questions?

Go-iRODSClient: https://github.com/cyverse/go-irodsclient
iRODS FUSE Lite: https://github.com/cyverse/irodsfs
iRODS CSI Driver: https://github.com/cyverse/irods-csi-driver

Appendix

CyVerse DE Integration – Custom Path Mapping



CyVerse DataStore (iRODS)

```
iplant (zone)/
└── home/
    ├── iychoi/
    │   ├── input_data/
    │   │   ├── file1
    │   │   └── file2
    │   └── output_data
    └── shared/
        └── reference_dataset1
```

CyVerse DE

```
.
├── input/
│   ├── reference_dataset
│   ├── file1
│   └── file2
└── output
```