

Multi-Interface for the Highly-Scalable Data Service (HSDS)

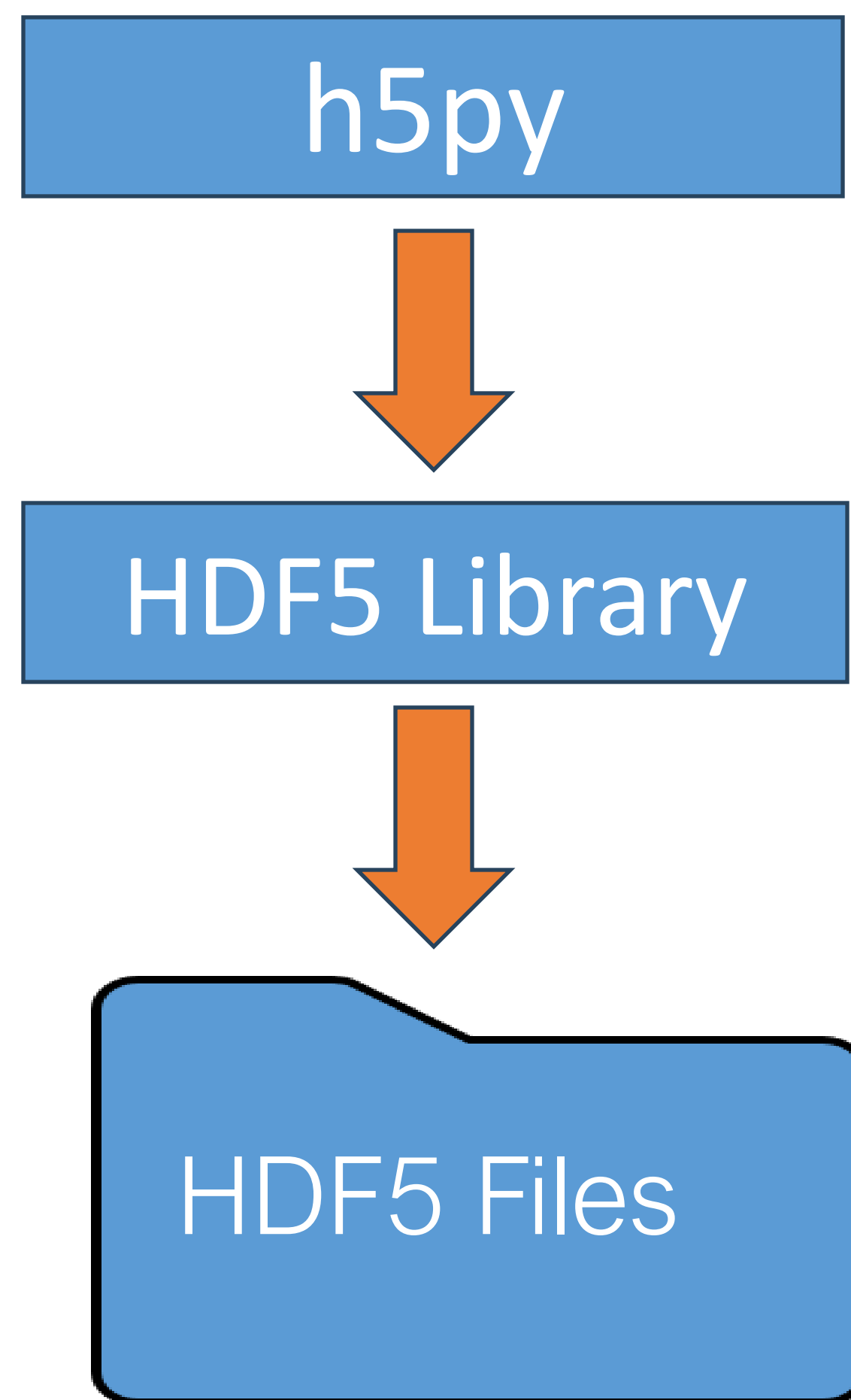


Matt Larson



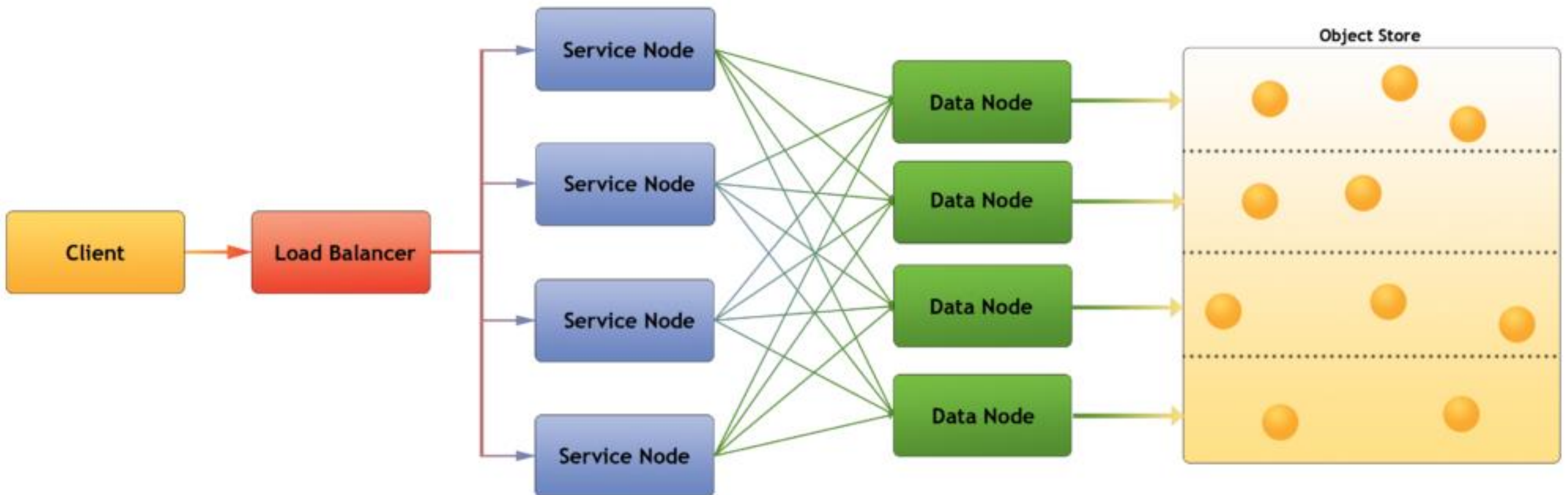
HUG 2024

HDF5 on Your Local Machine

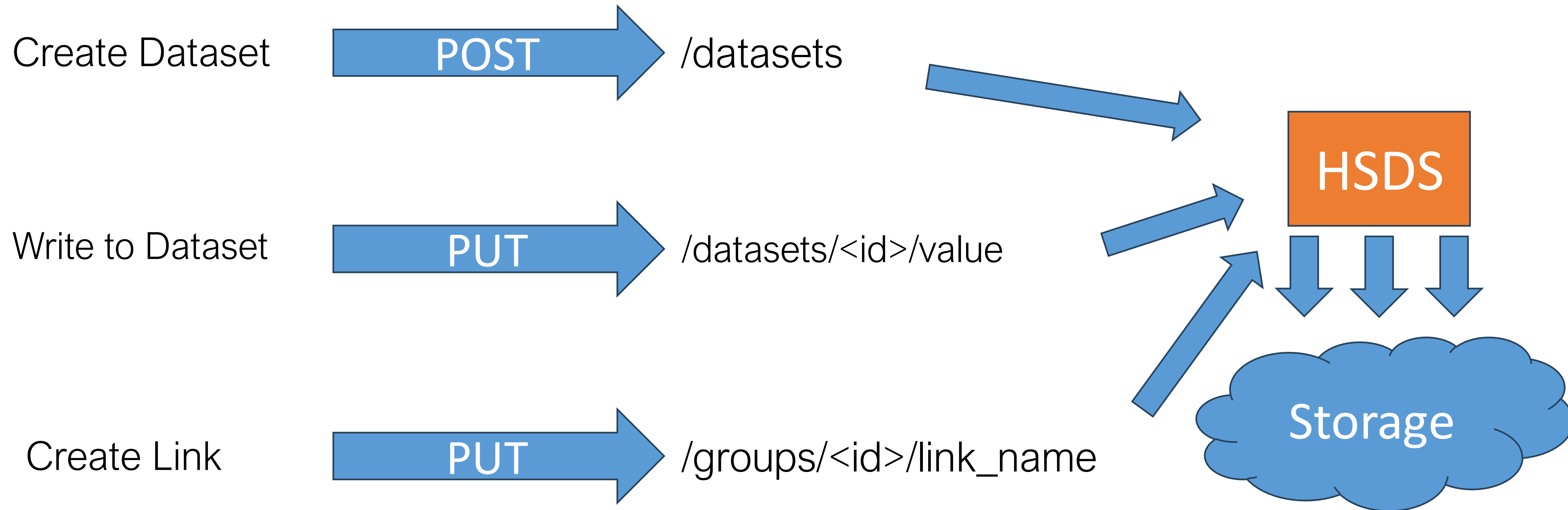


Highly-Scalable Data Service (HSDS)

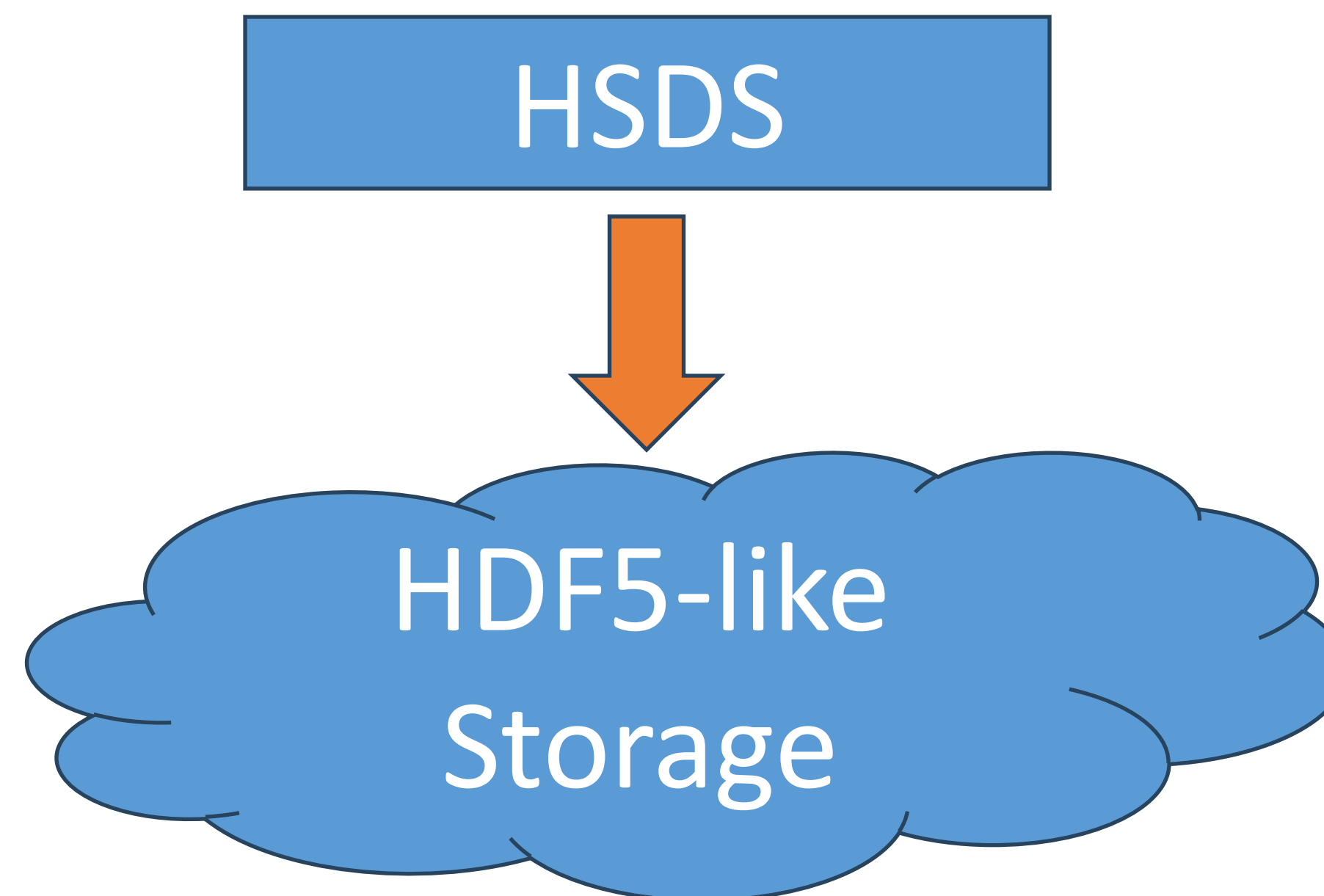
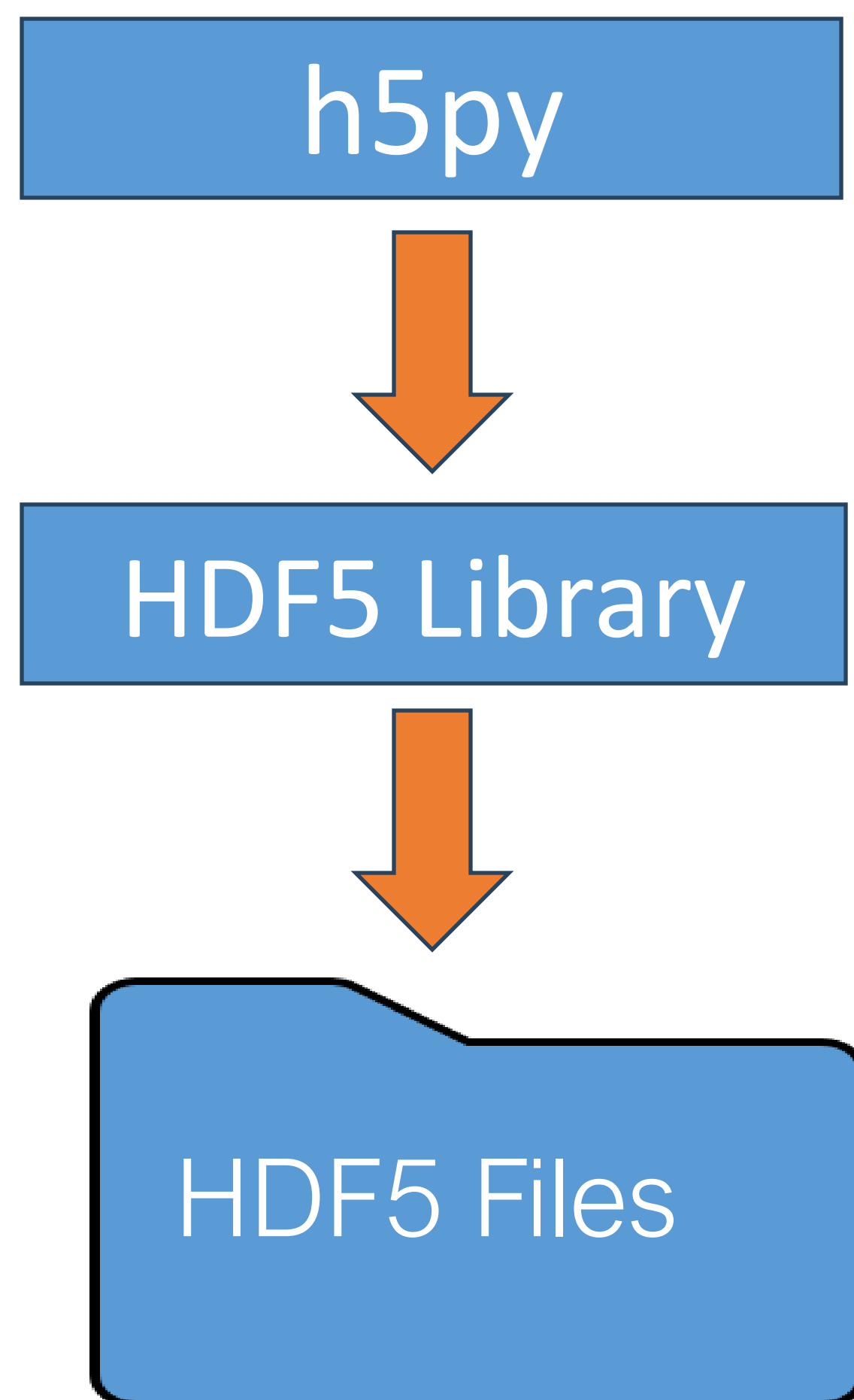
- HDF5 in Cloud via REST API
- Backend: AWS S3, Azure Blob, or OpenIO
- Access: CL tools, REST VOL, h5pyd



HSDS API



HDF5 on the Cloud with HSDS



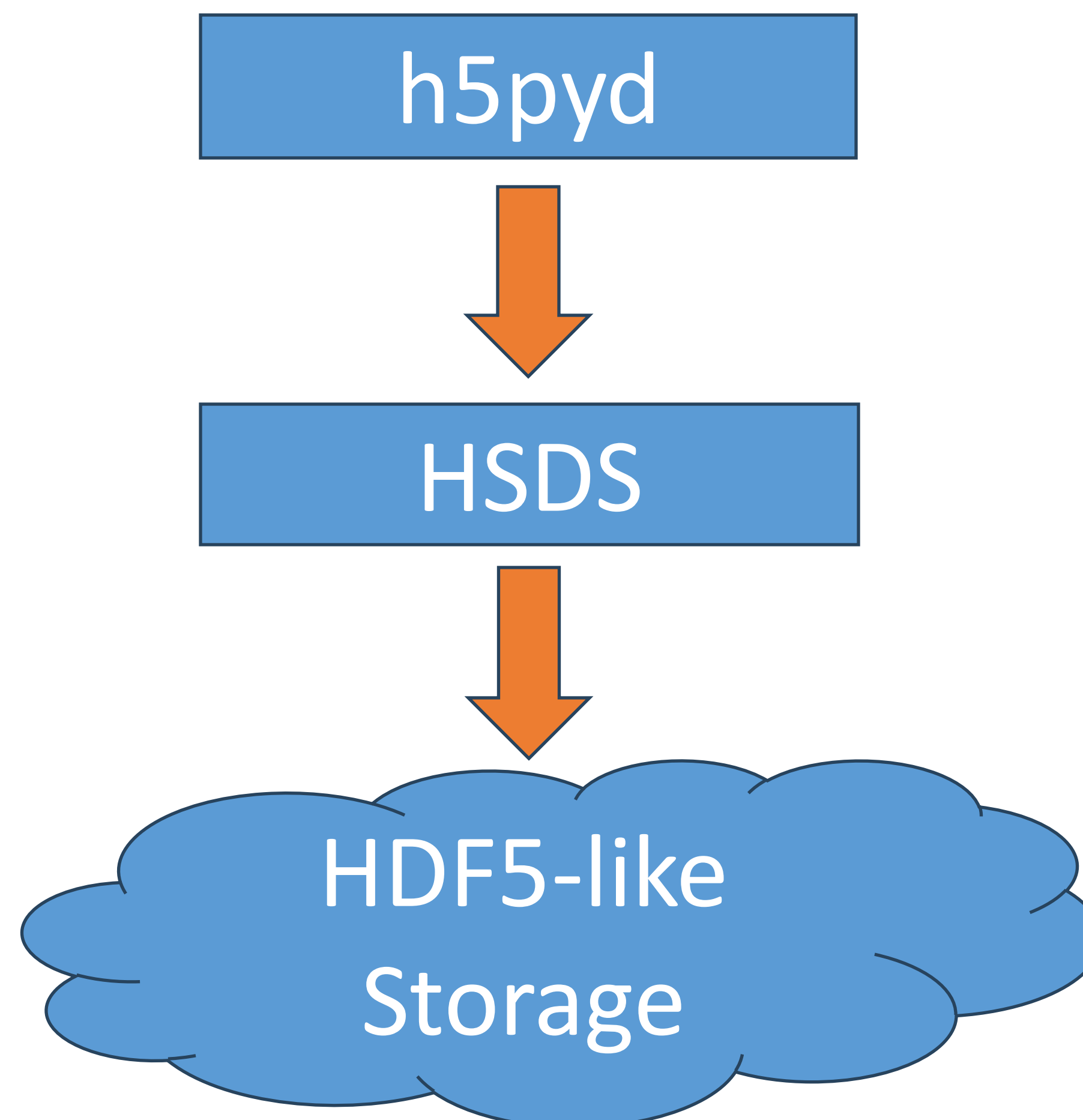
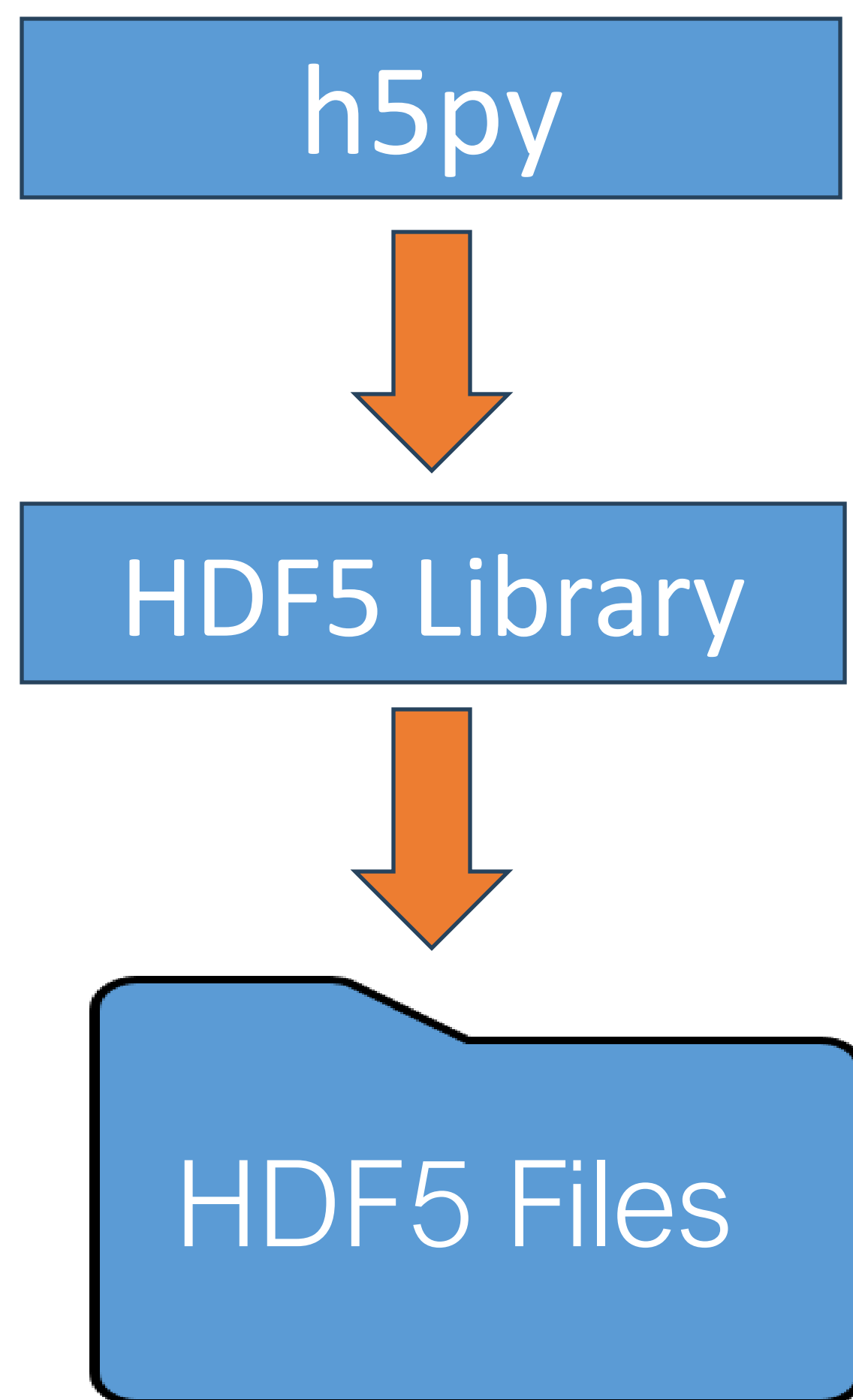
"I like the h5py interface."

"I don't want to rewrite
everything."



h5py-distributed
(h5pyd)

HDF5 on the Cloud with HSDS

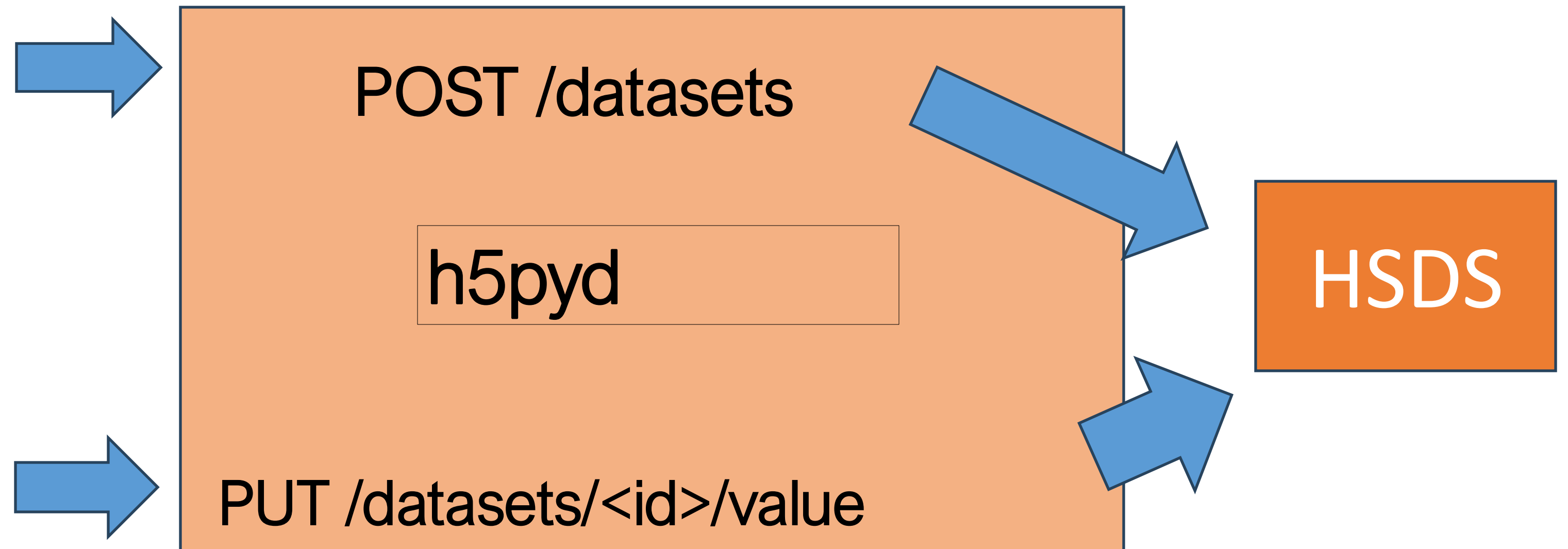


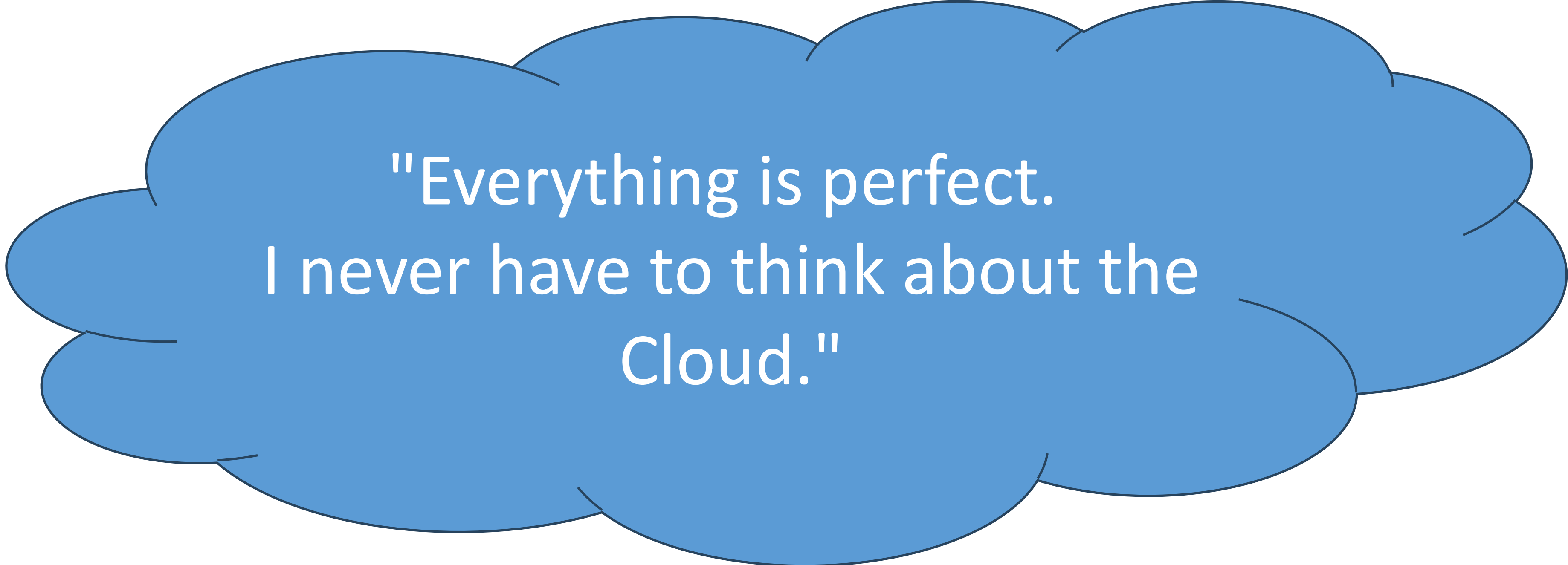
h5py-like Cloud Access with h5pyd

```
import h5pyd as h5py
```

```
grp.create_dataset()
```

```
dset[...] = values
```

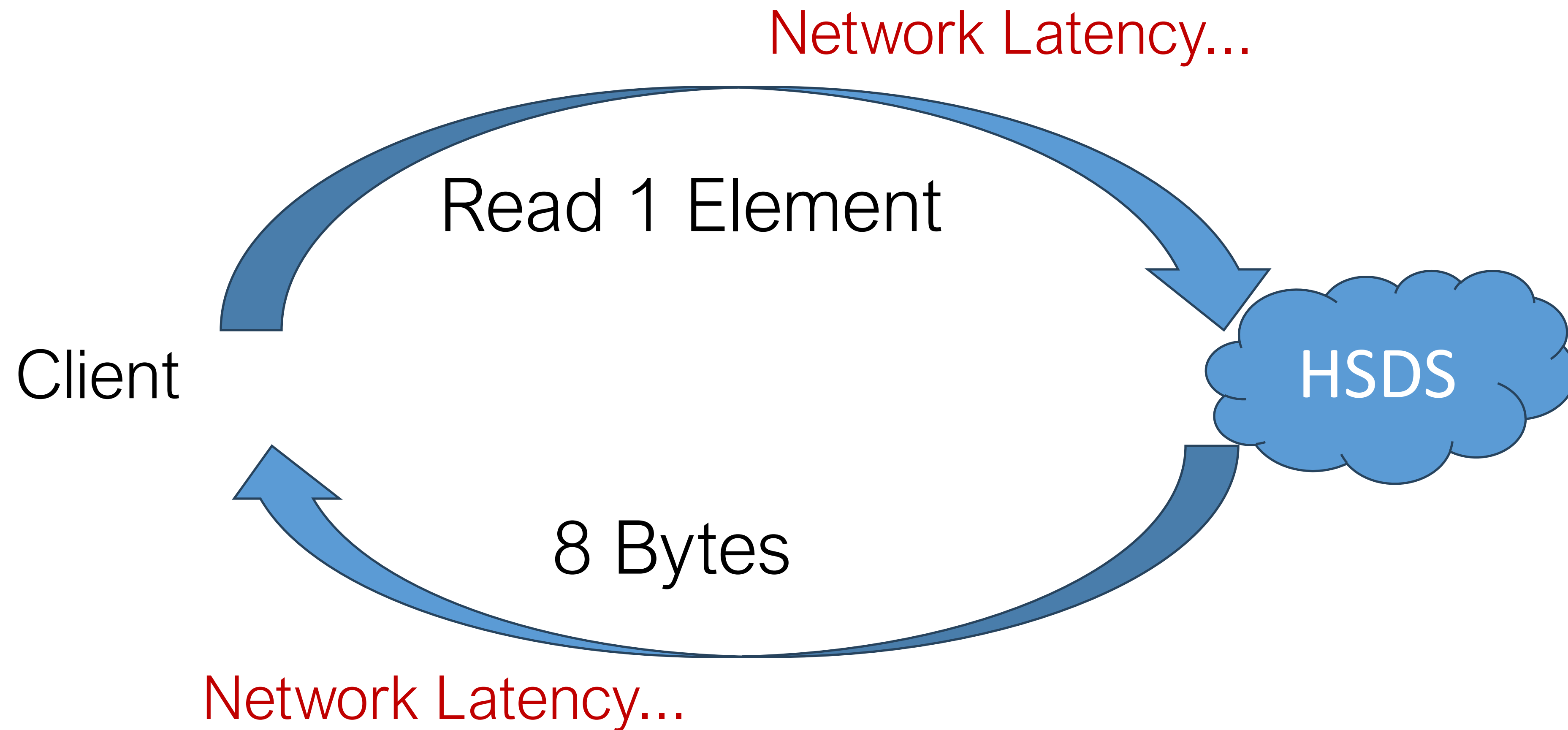




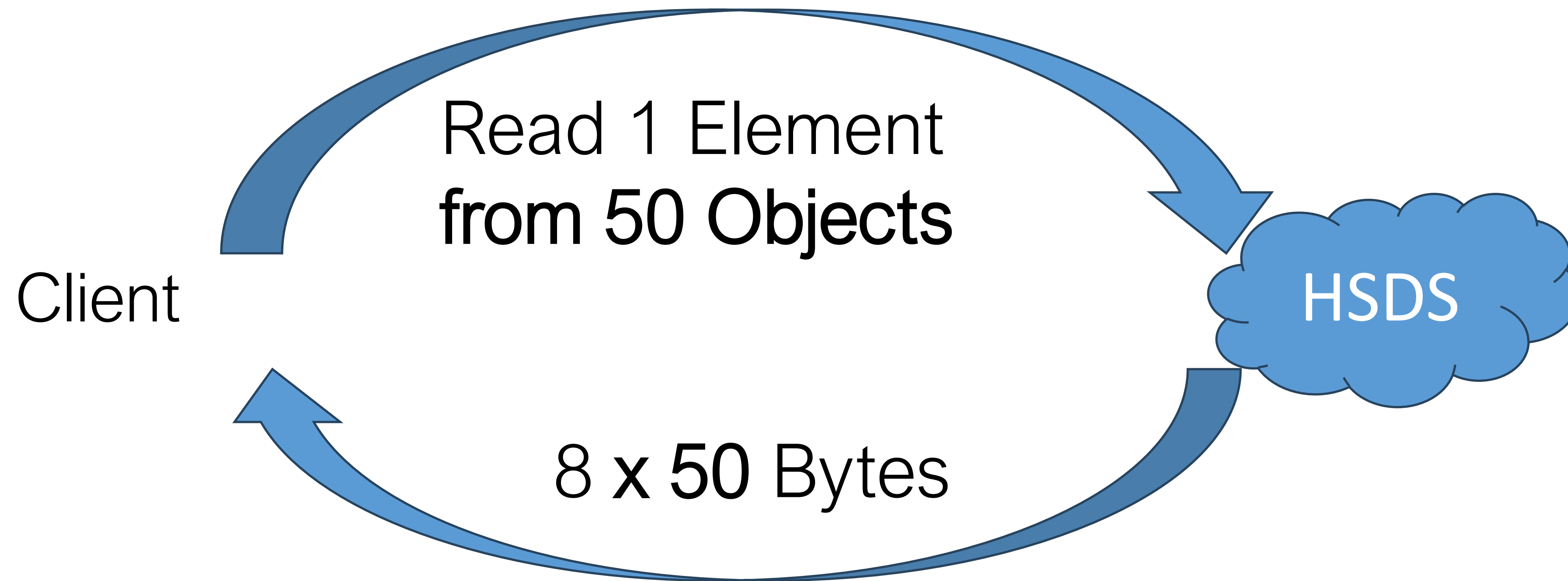
"Everything is perfect.
I never have to think about the
Cloud."

Not exactly...

The Small Packet Problem



Solution: Batching



Multi-Attribute, Multi-Link h5pyd Interface

Multi-Link: Creation



```
names = ["link" + str(i) for i in range(100)]
```

```
links = [h5py.SoftLink("dset" + str(i)) for i in range(100)]
```

```
group[names] = links
```

One request, many links created

Multi-Link: Retrieval

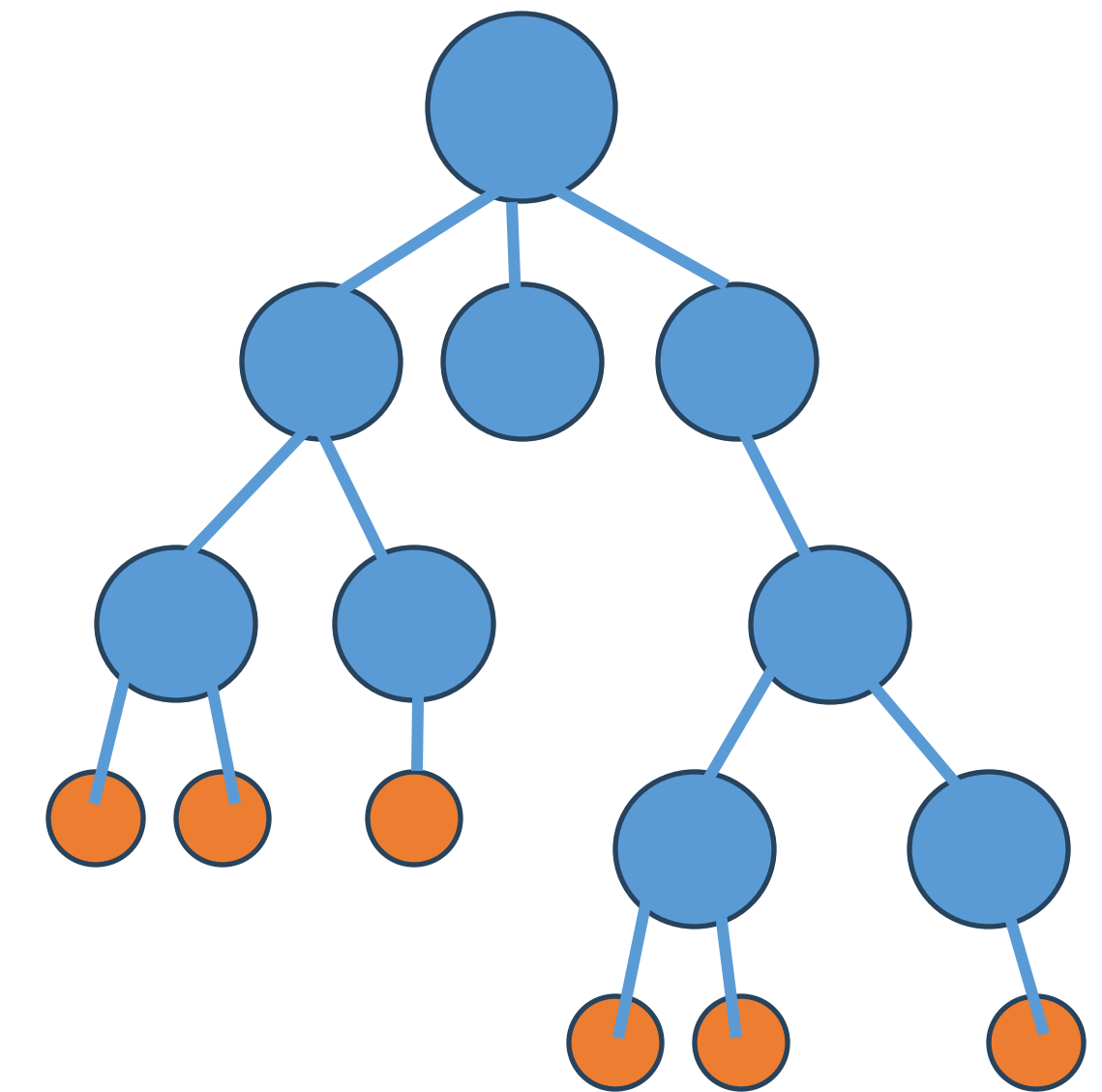
No target name → get all links in group

```
links = group.get(None, getlink=True)
```

Get links recursively from subgroups

```
all_links = group.get(...follow_links=True)
```

Hierarchy traversed in one request



Multi-Link: Benchmark

Seconds to complete benchmark

10,000 Links

Create:
95.0s → 0.5s

Retrieve:
160.8s → 0.2s

200

150

100

50

0

Create
Retrieve

Single

Multi

Multi-Attribute: Create/Retrieve

```
names = ['attr' + str(i) for i in range(100)]
```

```
values = [np.arange(i) for i in range(100)]
```

```
group.attrs.create(names, values)
```

One request, many attrs created

```
...
```

```
values = group.attrs.get_attributes()
```

One request, many attrs read

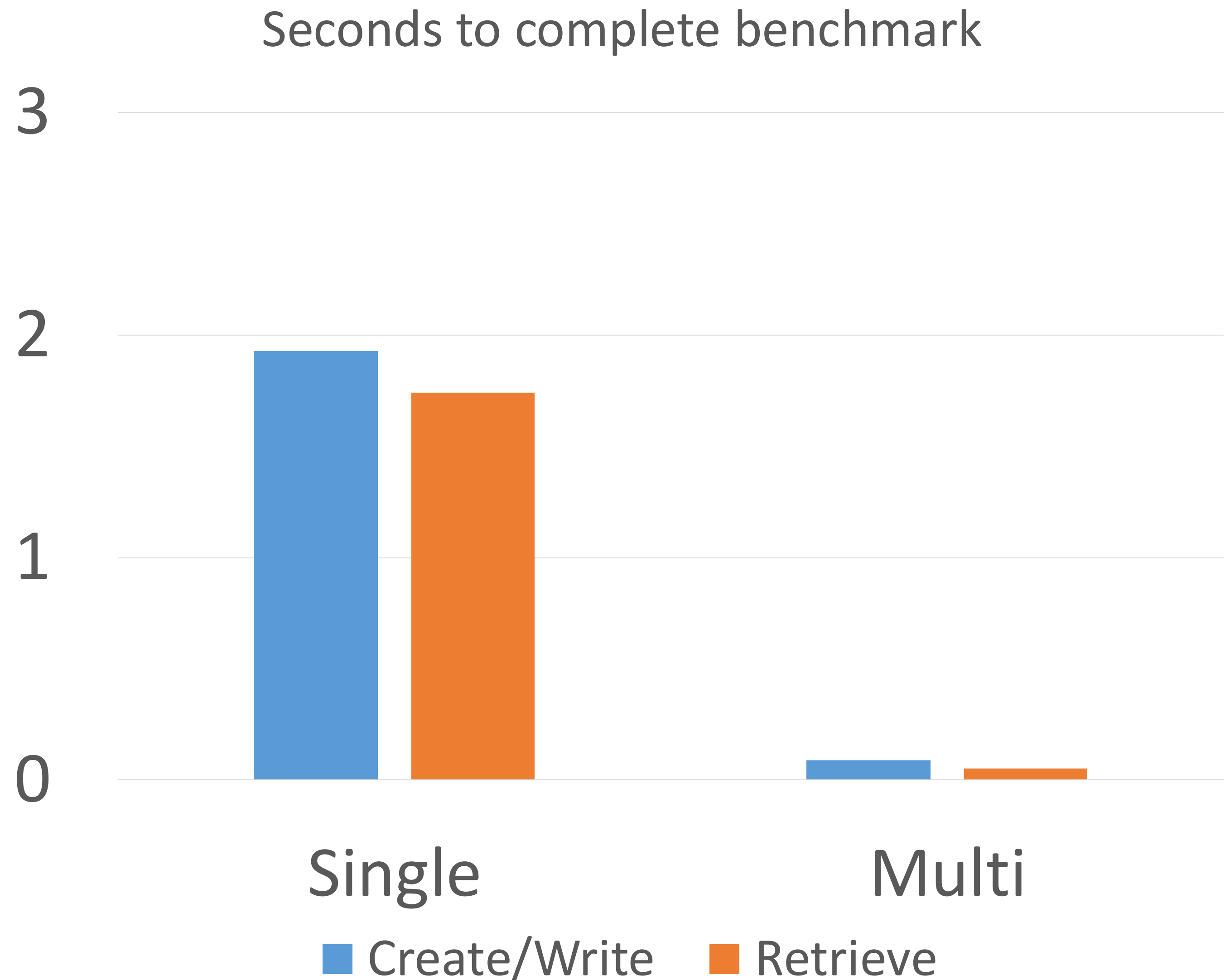
Multi-Attribute: Benchmark



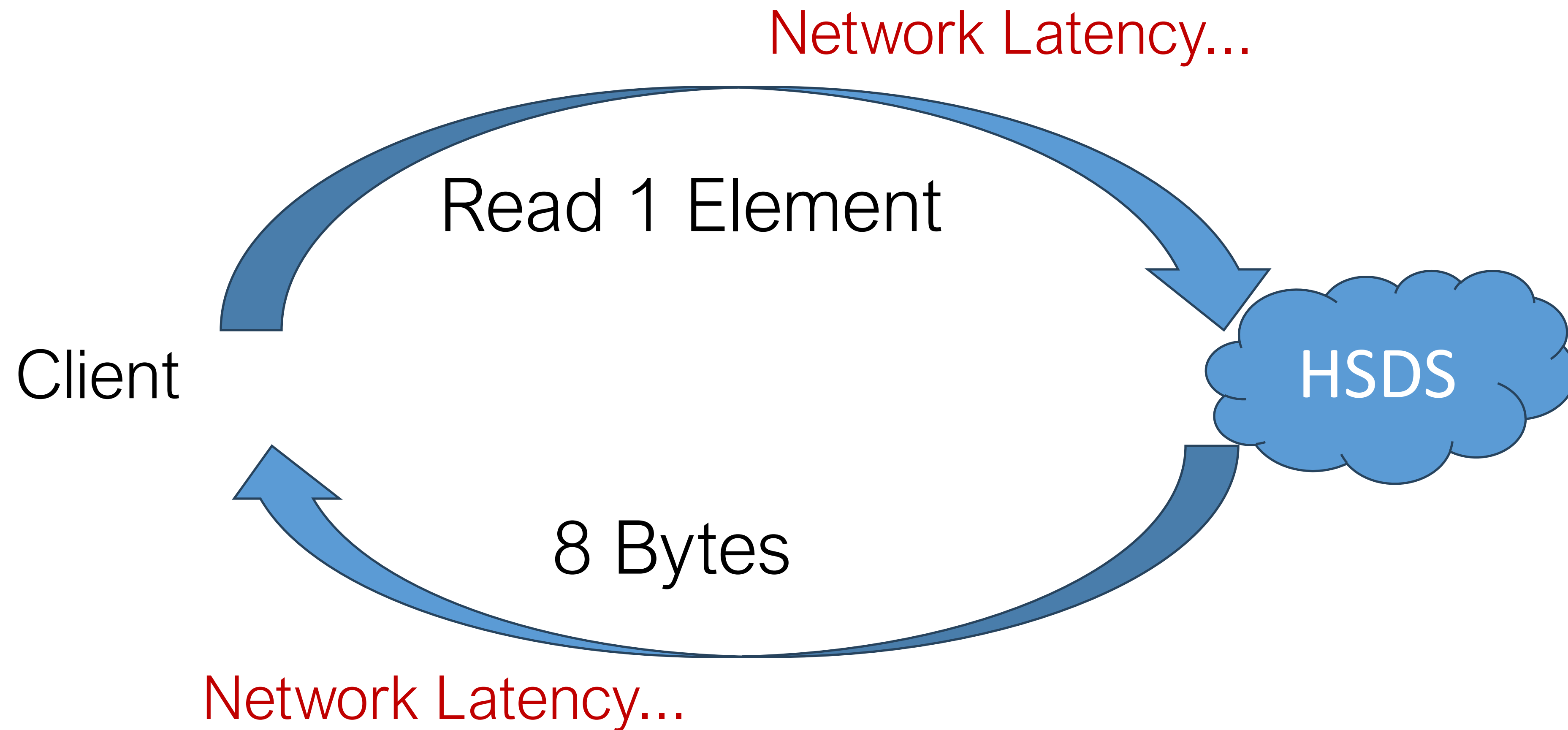
100 x 1000 Integer Attributes

Create/Write:
1.93s → 0.10s

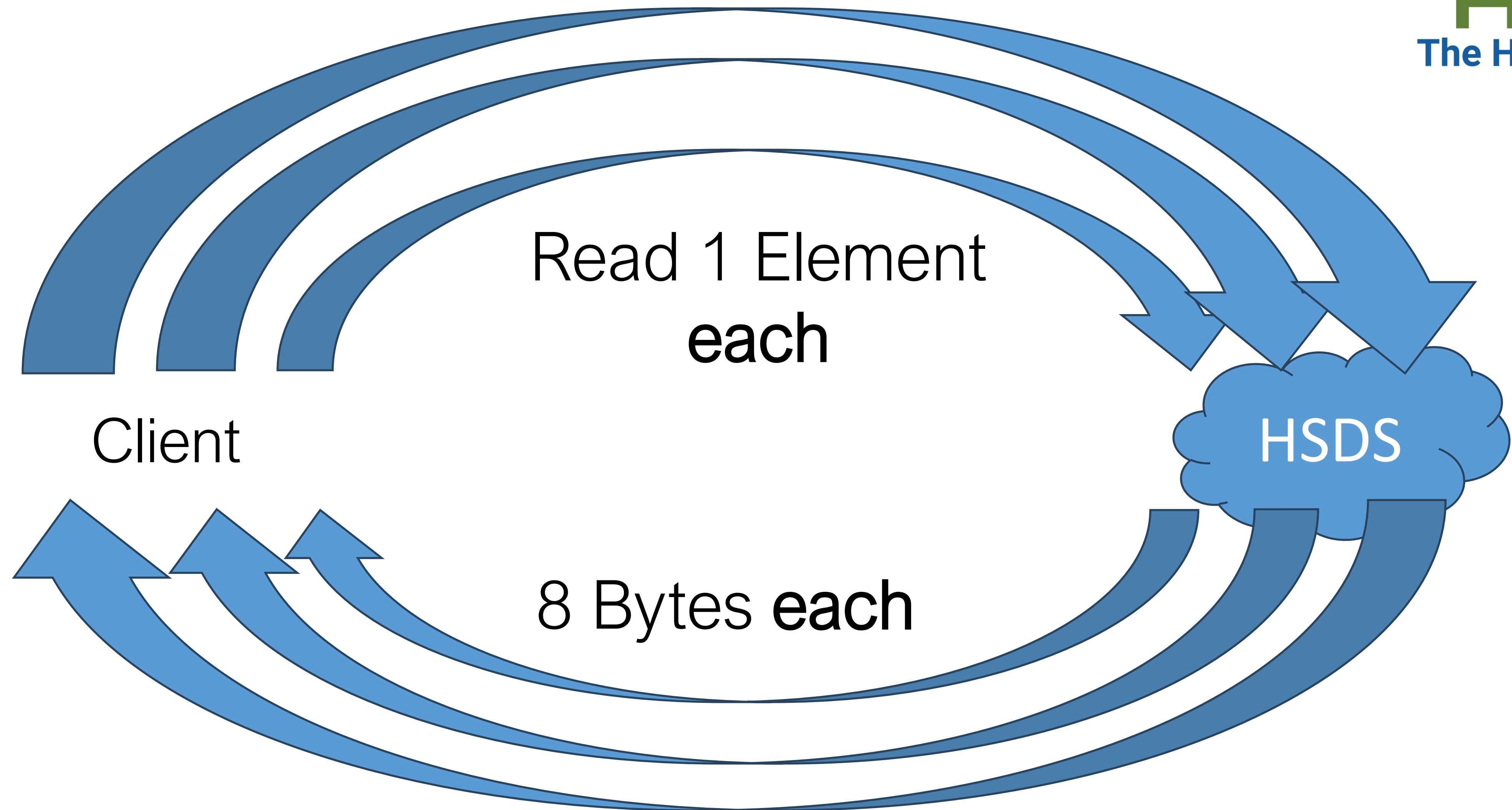
Retrieve:
1.74s → 0.05s



The Small Packet Problem



Solution: Parallelism



Multi-Dataset h5pyd Interface

MultiManager for Datasets

```
mm = MultiManager([dset1, dset2, dset3])
```

```
data_out = mm[...]
```

Read from all datasets in parallel

```
mm[...] = [data1, data2, data3]
```

Write to all datasets in parallel

MultiManager: Selections

```
slice1 = np.s_[0:100, 0:100]  
slice2 = np.s_[6000:7000, 6000:7000]  
selections = [slice1, slice2]
```

```
Mm = MultiManager([dset1, dset1])  
data_out = mm[selections]
```

Read disjoint regions of one dataset

MultiManager: Benchmark



Seconds to complete benchmark

Reading Full Datasets
0.97s → 0.69s

Writing Full Datasets
0.99s → 0.64s

Random Access
3.74s → 0.45s



Closing Remarks

- New interfaces not in h5pyd/HSDS release versions
- Multi-Link/Attribute ops are h5pyd only
 - h5py has MultiManager in progress!
 - Library users: REST VOL: multi_read/write
- MultiManager is not a free lunch; HSDS needs resources

Links



HSDS - <https://github.com/HDFGroup/hsds>

h5pyd - <https://github.com/HDFGroup/h5pyd>

HSDS Forum - <https://forum.hdfgroup.org/c/hsds/>

h5py - <https://github.com/h5py/h5py>

THANK YOU!

Questions & Comments?