
KubeCon
CloudNativeCon
North America 2019

Decentralized Distribution Strategy in Dragonfly

Ben Ye
@dragonfly_oss

Agenda

- Dragonfly Overview
- Dragonfly Architecture
- Decentralization Mode
- Roadmap

What and Why

- P2P based, highly reliable, image distribution system
- Bottleneck in pulling images in DCs at scales

Native vs Dragonfly

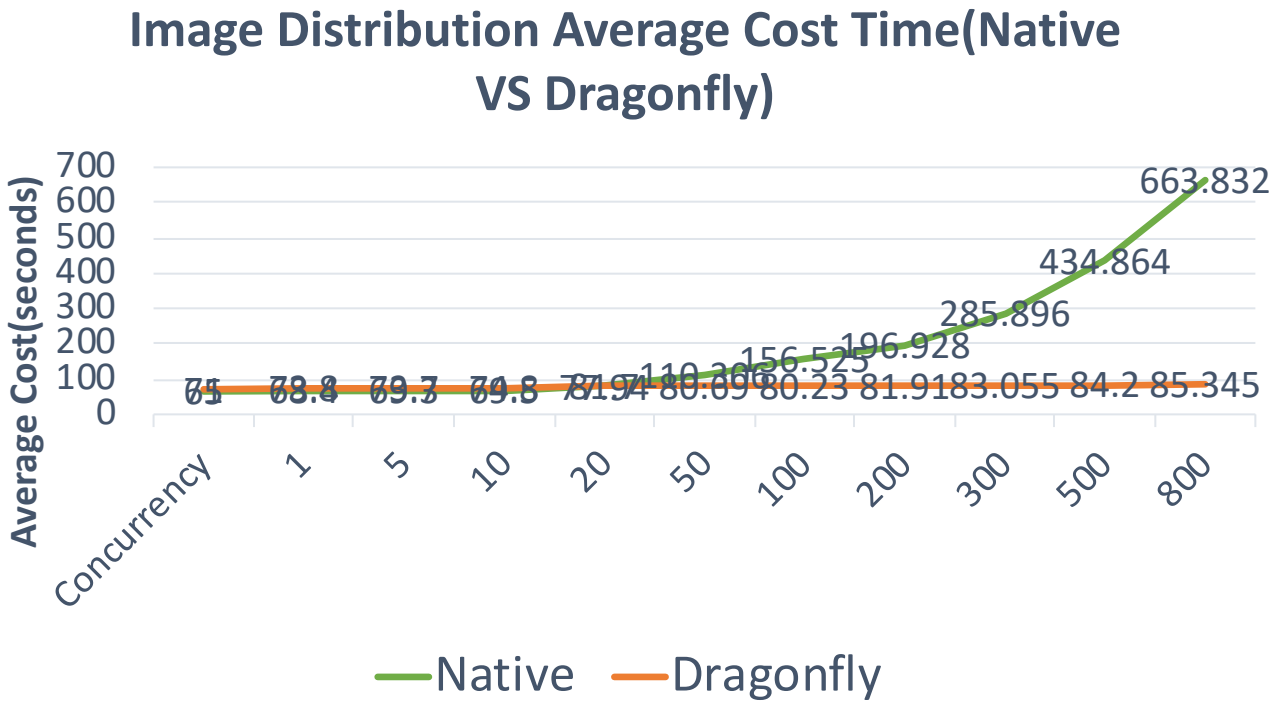
KubeCon CloudNativeCon North America 2019

Testing conditions:

- Server: 2 VMs(24C64G, SSD, 1000Mb/s)
- Client: 200 VMs(4C8G, 100Mb/s)

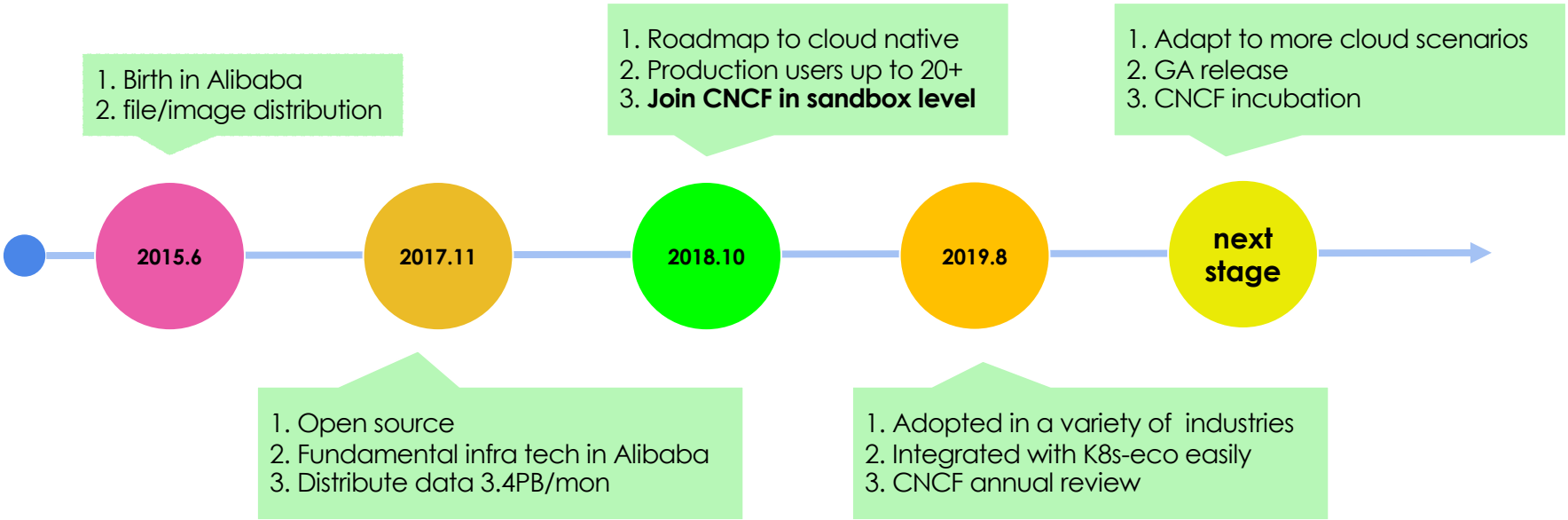
Conclusion:

- Dragonfly is more smoothly than native
- With the expansion of the scale, dragonfly's advantage becomes more and more obvious



Milestone

image/file distribution system in cloud native era



Industry Adoptions

telecom&communication

e-commerce

cloud service providers

live streaming

public life service

artificial Intelligence

Integration

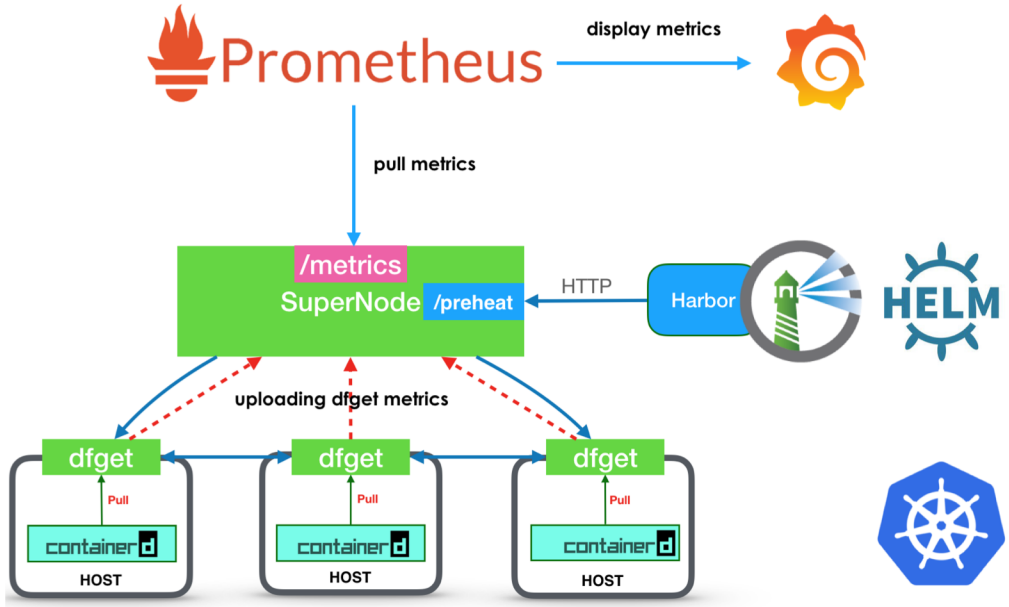
Prometheus: metrics monitoring

Containerd: container engine integration

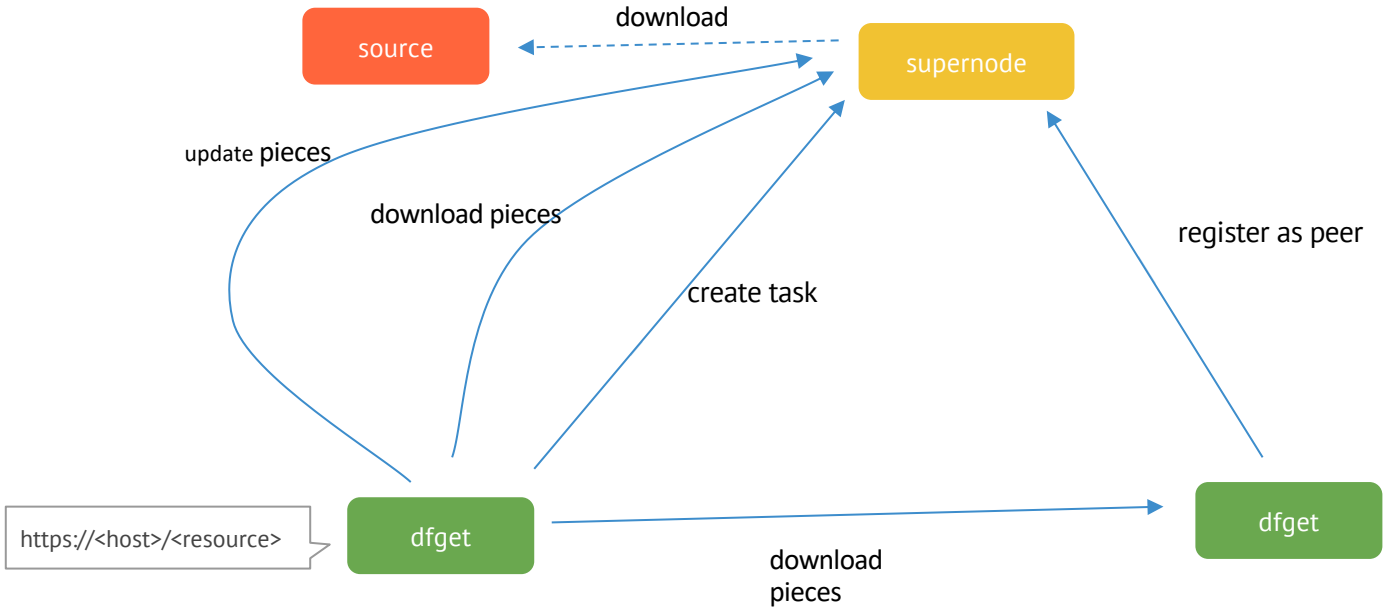
Harbor: collaborate in image prefetch

Helm: dragonfly/supernode installation

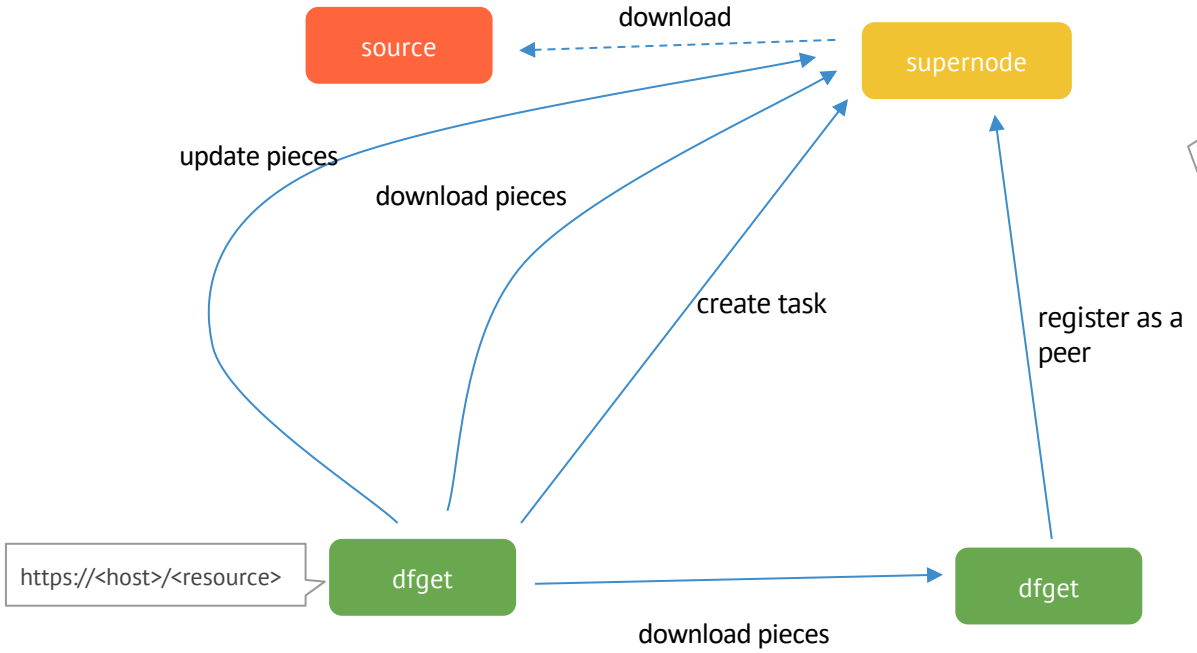
K8s: dragonfly/dfget deployment



Architecture

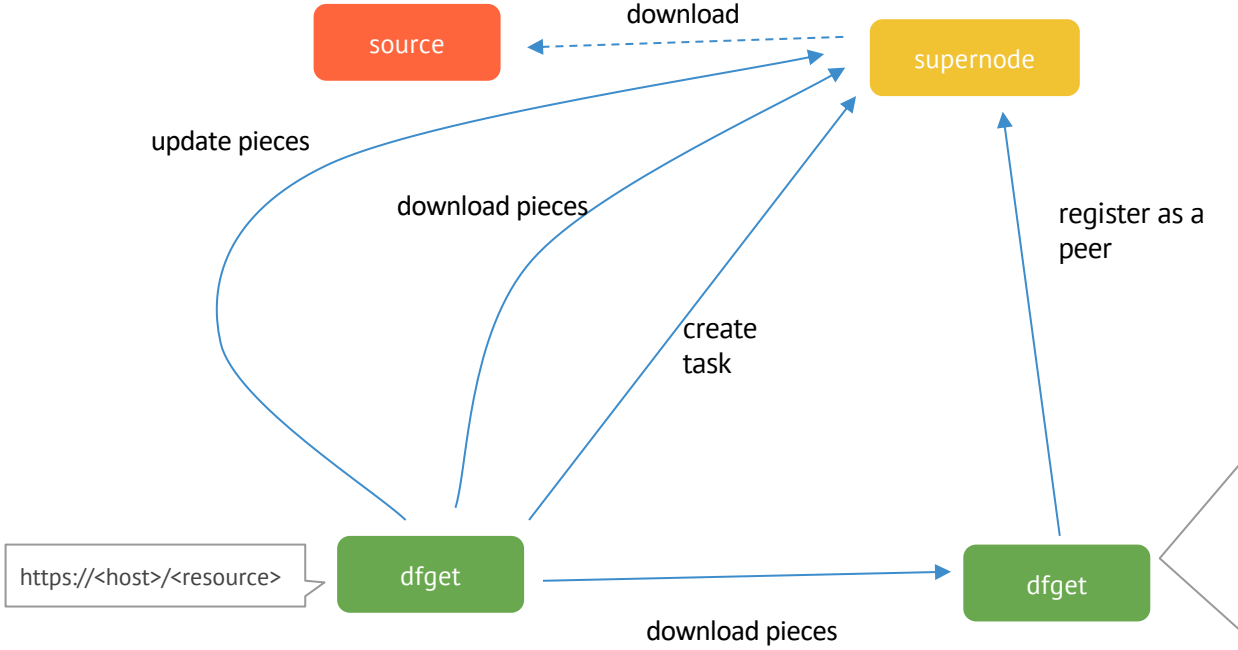


Supernode



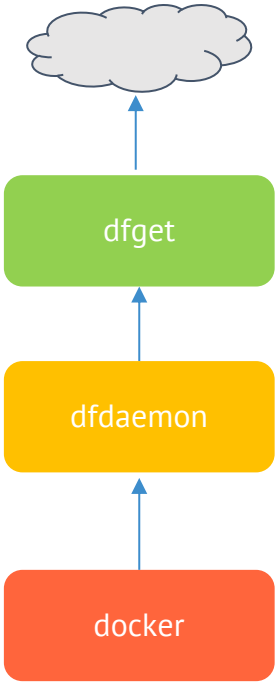
- **API**
  - peers
  - tasks
  - pieces
- **Scheduling**
  - select peers
- **Seeding**
  - initial seeding
- **Metrics**

Dfget



- **CLI**
- **Rate Limit**
  - Per Task
  - Per Host
- **Checksum**
  - integrity check
- **Seeding**

Dfdaemon



- **Reverse Proxy**

```
reverseProxy := httputil.NewSingleHostReverseProxy(target)
  reverseProxy.Transport = NewDFRoundTripper(reg.TLSConfig())
  reverseProxy.ServeHTTP(w, r)
```

- **Delegate to dfget**

```
 func RoundTrip(req *http.Request) (*http.Response, error)
{
    dstPath, err := rt.DFGetter.Download(req)
    ...
    fileReq, err := http.NewRequest("GET",
"file:///"+dstPath, nil)
    rt.RoundTripper.RoundTrip(fileReq)
}
```

Configuration

- **Registry Mirrors**: /etc/docker/daemon.json

```
# docker pull busybox:latest
{
        "registry-mirrors" : ["http://127.0.0.1:65001"]
}
```

- **HTTP_PROXY**: /etc/systemd/system/docker.service.d/http-proxy.conf

```
# docker pull [<registry-host>/]busybox:latest
[Service]
Environment="HTTPS_PROXY=http://127.0.0.1:65001"
```

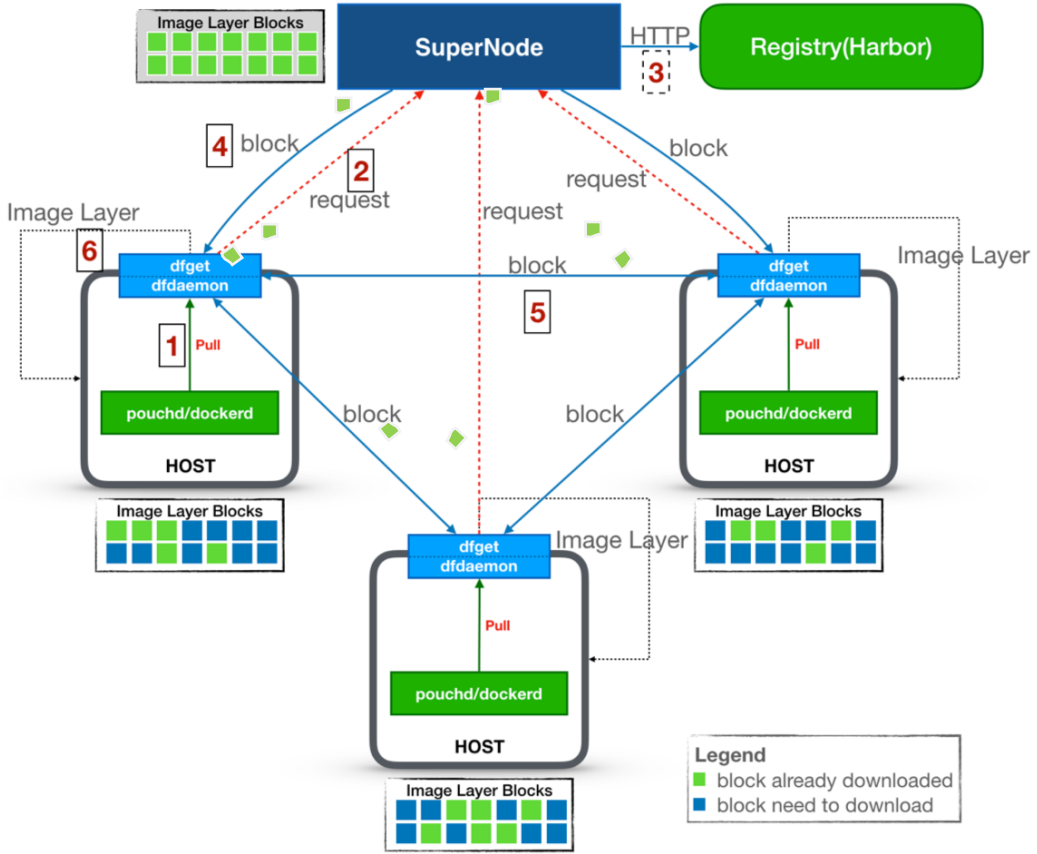
- **HTTPS_PROXY**: /etc/systemd/system/docker.service.d/http-proxy.conf

```
# docker pull [<registry-host>/]busybox:latest
[Service]
Environment="HTTPS_PROXY=http://127.0.0.1:65001"
```

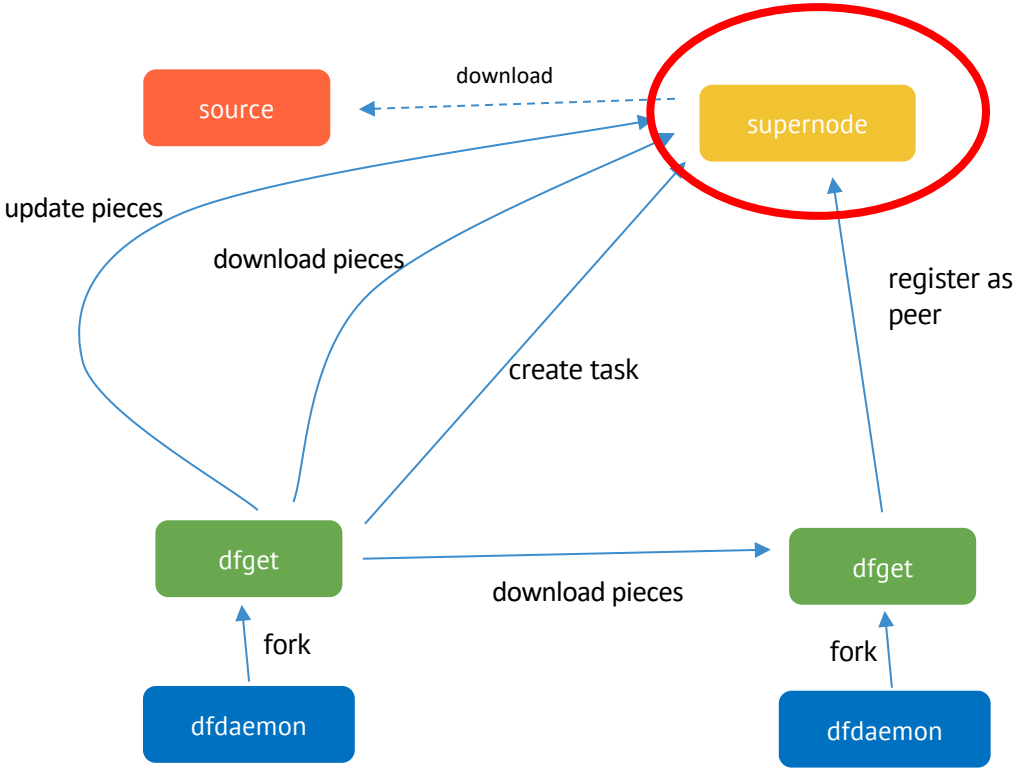
Pull image

docker pull [registry/]nginx:latest

1. Pull image from node proxy(dfdaemon)
2. Send pulling requests to SuperNode
3. Cache image layers from Registry if non-exist
4. Reply to peers which have pieces
5. Transport pieces among all peers
6. Finish whole pulling when all pieces are downloaded



Problems



- **Can't scale easily**
  - No HA
- **High demand of hardware**
  - Disk IO
  - Network Bandwidth
- **Increase the cost of maintenance**

Problems

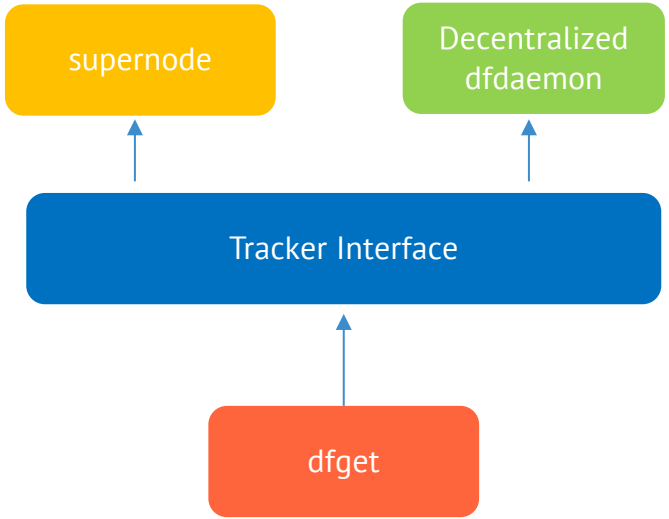
Is there a way to run Dragonfly without supernode?

Gossip

- **Membership**
- **Failure detection and recovery**
- **Custom event propagation**

https://github.com/hashicorp/serf

Tracker Interface




```
type Tracker interface {
 Register(node string, req *types.RegisterRequest)
 PullPiece(node string, req *types.PullPieceRequest)
 ReportPiece(node string, req *types.ReportPieceRequest)
 ServiceDown(node string, taskID string, cid string)
 ReportError(node string, req *types.ClientErrorRequest)
}
```

Custom Events

- Progress Events
  - Download Start Event
  - Download Progress Event
  - File Remove Event
- Length Event
- Recover Event
  - Very brief review of cluster API

Peer Discovery

Kubernetes

Nodes / Endpoints

Coredns

Service Discovery

Serf

Dummy Node

Further Work

- Extract dfget as library
- Scheduling in DfDaemon
- Metrics for decentralized mode
- Dfdaemon disk gc

Roadmap

features

supernode HA
decentralized scheduling
flexible plugin framework
enhanced encryption
more file transfer protocol
Dragonfly UI
……

scenarios

physical machine -> cloud disk
performance optimization
IoT scenarios
ARM and more computing arch
……

ecosystem

opentracing support
operator support
file distribution support in k8s
……

See more: https://github.com/dragonflyoss/Dragonfly/blob/master/ROADMAP.md

https://github.com/dragonflyoss/Dragonfly

THANKS!