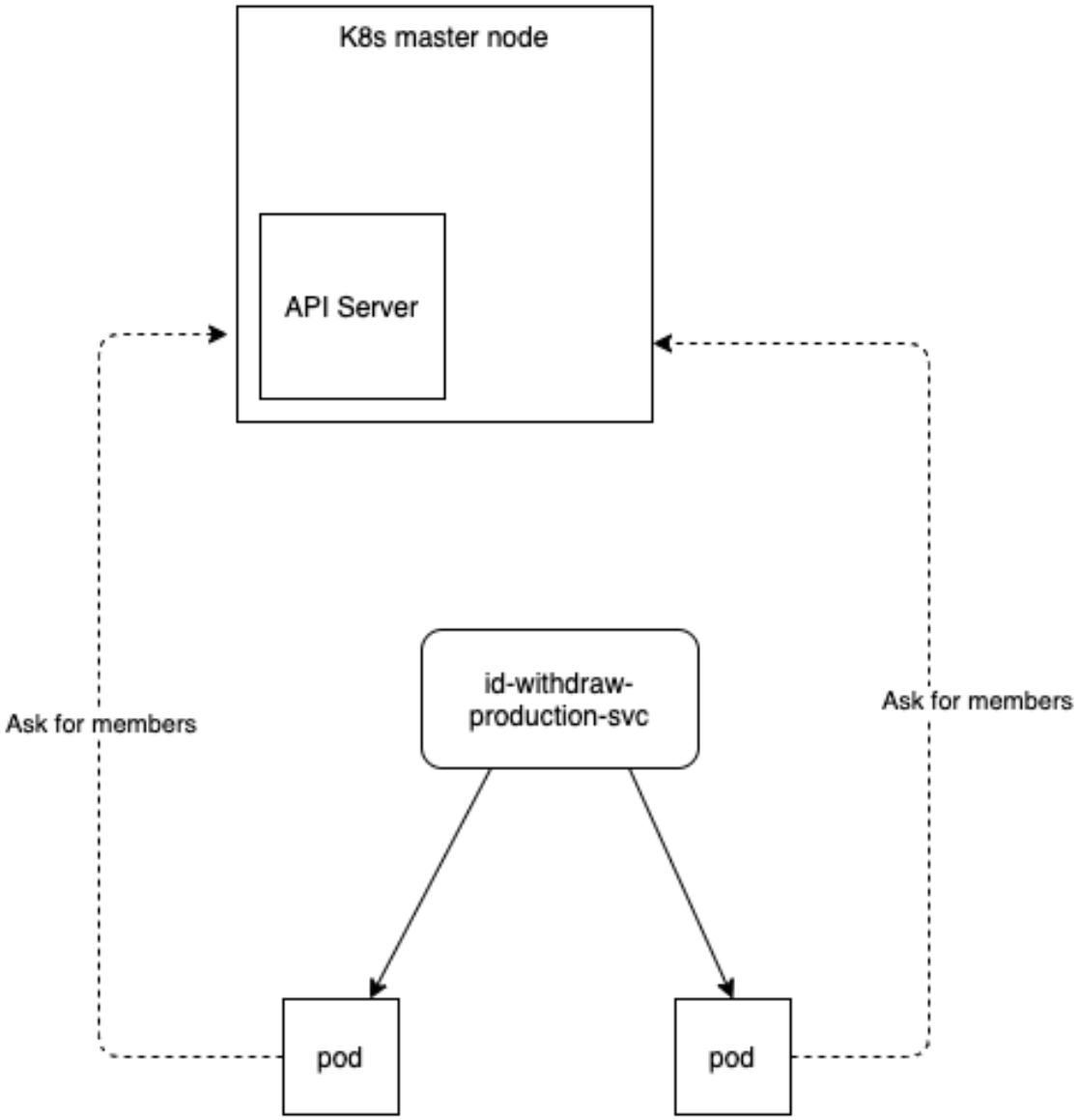
Hazecast Adv

K8s discovery

Mô hình triển khai các pods của kubernetes

- Pod : một instance của một service. Tương đương một node trong môi trường vật lí mà ZaloPay đang sử dụng
- Service: một group của những pod.
- VD: Triển khai service withdraw lên môi trường k8s với 5 instance sẽ có:
  - 1 service với tên là: td-withdraw-production
  - 5 pods ứng với 5 instance của service withdraw
- Tại 1 pod sẽ không biết được sự tồn tại của các pods khác
- Service sẽ nắm giữa toàn bộ danh sách của các pods bao gồm thông tin như là: ip, port, ... để khi nhận được request sẽ tiến hành loadbalancing để forward request vào 1 pod.

Cách hazelcast sử dụng k8s discovery



- Khi 1 node join vào cluster thì sẽ gọi lên master để lấy danh sách members của nó
  - API: https://kubernetes.default.svc/api/v1/namespaces/<name-space-name>/endpoints/<service-name>
  - Có 2 thông tin cần cung cấp cho master: **name-space-name & service-name**
- Sau khi có danh sách members sẽ tiến hành tạo cluster gồm những member đó
- Lưu ý: default thì pod không có quyền gọi lên API master để lấy danh sách members. Mình có thể allow quyền với command sau:
  - kubectl apply -f https://raw.githubusercontent.com/hazelcast/hazelcast-kubernetes/master/rbac.yaml

Data loss problem

Mất data do không đủ dữ liệu backup để khôi phục

Kịch bản test:

- Gồm có 3 instance và 2 server.
- Server 1 chứ 1 instance

- Server 2 chứ 2 instance
- Tiến hành kill server 2

Dữ liệu tại 3 instance như sau:

Map Statistics (In-Memory Format: BINARY)

Now | Now

Member	Entries	Gets	Puts	Removals	Sets	Entry Memory	Backups	Backup Memory	Events	Hits	Locks	Dirty Entries
127.0.0.1:5701	17	0	0	0	0	10.79 kB	14	8.97 kB	0	34	0	0
127.0.0.1:5702	18	0	0	0	0	11.41 kB	17	10.77 kB	0	36	0	0
127.0.0.1:5703	15	0	0	0	0	9.59 kB	19	12.05 kB	0	30	0	0
TOTAL	50	0	0	0	0	31.79 kB	50	31.79 kB	0	100	0	0

- Total data là : 50
- Đang chạy chế độ backup-count : **1**
- Với mỗi 1 primary data sẽ được lưu **1** backup ở máy khác

Tiến hành kill 2 instance 5701 và 5703 (kill -9 để giả lập trường hợp server shutdown bất ngờ như là: mất mạng, mất điện, lỗi không mong muốn, ...)

Dữ liệu sau khi tiến hành kill 2 instance

Map Statistics (In-Memory Format: BINARY)

Now | Now

Member	Entries	Gets	Puts	Removals	Sets	Entry Memory	Backups	Backup Memory	Events	Hits	Locks	Dirty Entries
127.0.0.1:5701	31	0	0	0	0	19.76 kB	0	0 B	0	62	0	0
TOTAL	31	0	0	0	0	19.76 kB	0	0 B	0	62	0	0

- Chỉ còn lại 31 dữ liệu. Bị mất = 50 -31 = 19 dữ liệu

Nhận xét:

- Việc phân chia partition chỉ là hạn chế trường hợp mất dữ liệu chứ không đảm bảo 100%
- Đối với các service của mình cần dữ liệu đảm bảo 100% vì hiện tại không có cơ chế warn-up
- Cách sử dụng hzc của minh hiện tại chỉ dùng để lưu trữ config và share sự thay đổi giữa các instance
- Chủ yếu 99.99% là thao tác read, chỉ write lúc start instance & reload config

Giải pháp khắc phục: Sử dụng backup-count

- "**backup-count**: This setting backs up map entries to other Hazelcast nodes in the cluster. The default value is 1, which means it backs up to one other node. Setting this to 2 backs up a map entry to two other Hazelcast nodes. The maximum backup count is 6. The backup occurs synchronously, so the backup has priority and will block other operations." - https://docs.hazelcast.com/imdg/latest-dev/data-structures/map.html

Demo sử dụng backup count: 6

hzc.config

```
Config hazelCastConf = new Config();
...
mapConfig.setName("default").setBackupCount(6);
...
```

Cluster có 1 instance

Map Statistics (In-Memory Format: BINARY)

Now | Now

Member	Entries	Gets	Puts	Removals	Sets	Entry Memory	Backups	Backup Memory	Events	Hits	Locks	Dirty Entries
127.0.0.1:5701	50	0	0	0	0	31.79 kB	0	0 B	0	0	0	0
TOTAL	50	0	0	0	0	31.79 kB	0	0 B	0	0	0	0

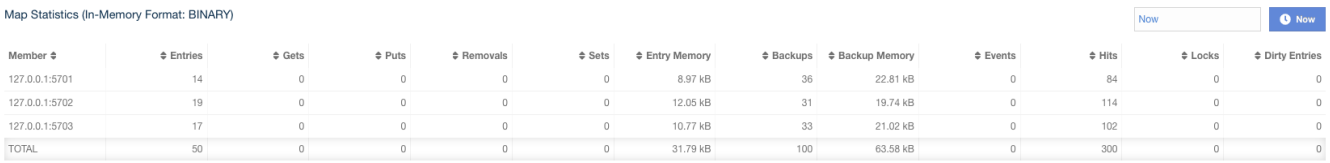
Cluster có 2 instances

Map Statistics (In-Memory Format: BINARY)

Now | Now

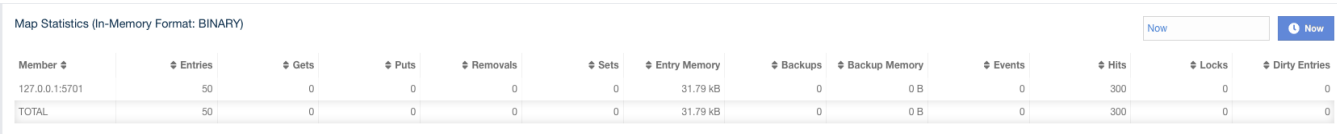
Member	Entries	Gets	Puts	Removals	Sets	Entry Memory	Backups	Backup Memory	Events	Hits	Locks	Dirty Entries
127.0.0.1:5701	24	0	0	0	0	15.29 kB	26	16.50 kB	0	24	0	0
127.0.0.1:5702	26	0	0	0	0	16.50 kB	24	15.29 kB	0	26	0	0
TOTAL	50	0	0	0	0	31.79 kB	50	31.79 kB	0	50	0	0

Cluster node có 3 instances



Member	Entries	Gets	Puts	Removals	Sets	Entry Memory	Backups	Backup Memory	Events	Hits	Locks	Dirty Entries
127.0.0.1:5701	14	0	0	0	0	8.97 kB	36	22.81 kB	0	84	0	0
127.0.0.1:5702	19	0	0	0	0	12.05 kB	31	19.74 kB	0	114	0	0
127.0.0.1:5703	17	0	0	0	0	10.77 kB	33	21.02 kB	0	102	0	0
TOTAL	50	0	0	0	0	31.79 kB	100	63.58 kB	0	300	0	0

Tiến hành kill 2 instances: 5702 & 5703



Member	Entries	Gets	Puts	Removals	Sets	Entry Memory	Backups	Backup Memory	Events	Hits	Locks	Dirty Entries
127.0.0.1:5701	50	0	0	0	0	31.79 kB	0	0 B	0	300	0	0
TOTAL	50	0	0	0	0	31.79 kB	0	0 B	0	300	0	0

Nhận Xét:

- Khi số lượng backup count tăng lên thì dữ liệu sẽ được đảm bảo không mất
- Nếu ta thiết lập backup count : 6 thì quy luật hoạt động của backup count theo công thức như sau
  - Gọi n là số lượng instance
  - Nếu số lượng instance <= 6 thì số lượng primary data được backup sẽ là: n -1
  - Nếu số lượng instace > 6 thì số lượng primary data được backup sẽ là const = 6

High performance

Enabling Backup Reads

- "By default, Hazelcast has one sync backup copy. If `backup-count` is set to more than 1, then each member will carry both owned entries and backup copies of other members. So for the `map.get(key)` call, it is possible that the calling member has a backup copy of that key. By default, `map.get(key)` always reads the value from the actual owner of the key for consistency." - https://docs.hazelcast.com/imdg/latest-dev/data-structures/map.html
- "To enable backup reads (read local backup entries), set the value of the `read-backup-data` property to **true**. Its default value is **false** for consistency. Enabling backup reads can improve performance but on the other hand it can cause stale reads while still preserving monotonic-reads property." - https://docs.hazelcast.com/imdg/latest-dev/data-structures/map.html

hzc.config

```
Config hazelCastConf = new Config();
....
mapConfig.setName("default").setBackupCount(6).setReadBackupData(true);
....
```

Async Backups

- "Asynchronous backups, on the other hand, do not block operations. They are fire & forget and do not require acknowledgements; the backup operations are performed at some point in time."

hzc.config

```
Config hazelCastConf = new Config();
....
mapConfig.setName("default").setBackupCount(0).setAsyncBackupCount(6)
....
```

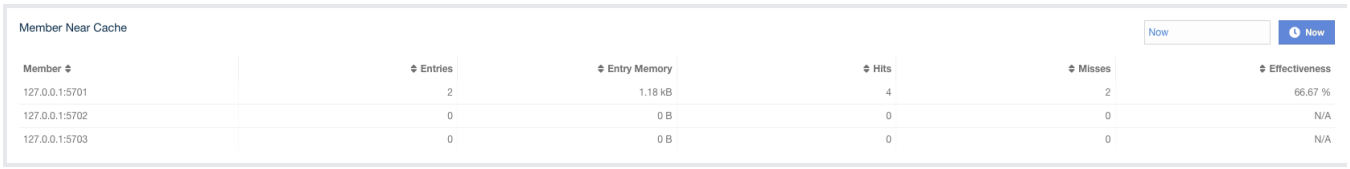
Near Cache

- "Map or Cache entries in Hazelcast are partitioned across the cluster members. Hazelcast clients do not have local data at all. Suppose you read the key `k` a number of times from a Hazelcast client or `k` is owned by another member in your cluster. Then each `map.get(k)` or `cache.get(k)` will be a remote operation, which creates a lot of network trips. If you have a data structure that is mostly read, then you should consider creating a local Near Cache, so that reads are sped up and less network traffic is created."

hzc.config

```
...
MapConfig mapConfig = new MapConfig();
mapConfig.setName("default");
NearCacheConfig nearCacheConfig = new NearCacheConfig();
nearCacheConfig.setTimeToLiveSeconds(900);
mapConfig.setNearCacheConfig(nearCacheConfig);
...
```

Sau thực hiện 2 lần map.get(key) với 2 key khác nhau thì cache đã có dữ liệu

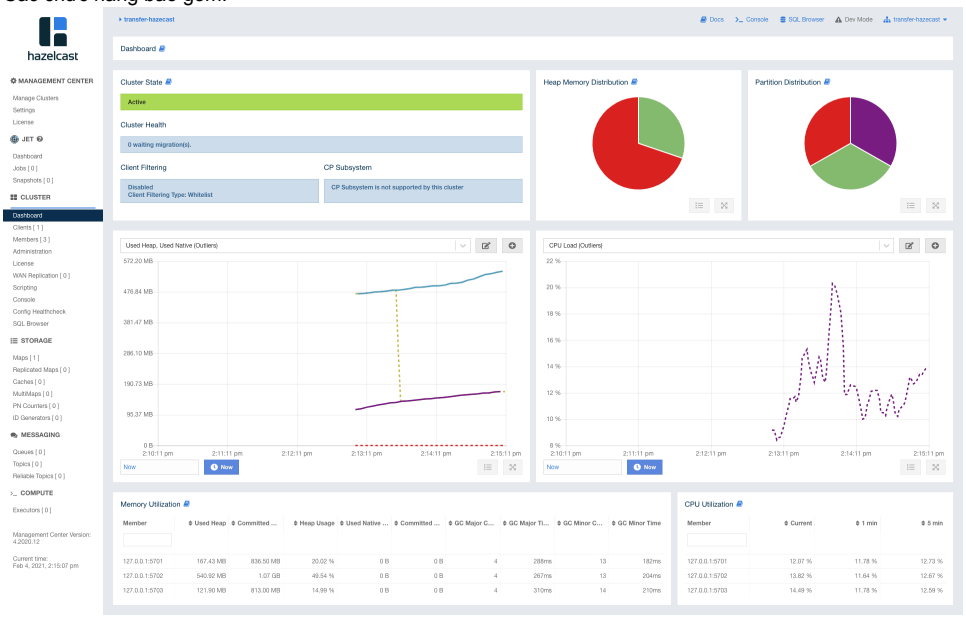


Lưu ý:

- "Near Cache works only when you access data via `map.get(k)` or `cache.get(k)` methods. Data returned using a predicate is not stored in the Near Cache."
- Vì vậy thao tác đọc tất cả dữ liệu trong map sẽ không sử dụng cache:
  - **VD: dataResponse.bankConfigMap = GsonUtils.toJsonString(withdrawDbConfig.getBankConfigMap());**

Hazelcast Monitoring cluster

Hazelcast management center

- Sử dụng hazelcast management center để monitor cluster
- Link: https://docs.hazelcast.org/docs/management-center/4.2020.12/manual/html/index.html#managing-clusters
- Các chức năng bao gồm:
- 

Hazelcast prometheus TBD

- Hướng tiếp cận: export prometheus với các metrics
  - Heap Memory
  - Partition Distribution
  - Data storage: entries, entries memory, backup, backup memory

Kết Luận

Các chức năng nên apply khi sử dụng hazelcast của nhóm Transfer

- K8s discovery: để triển khai được hzc trên k8s
- Backup count 0 để không thực hiên đồng bộ backup sync
- Async backup count 6 để tránh mất dữ liệu khi node chết bất thường & không lock các operations khác khi tiến hành sync dữ liệu giữa các node
- Enable read back up để giao roud trip khi đọc dữ liệu
- Có cơ chế monitor sử dụng grafana (TBD)