NPIS IT

Kubernetes discovery chaotic cache and how to speed up your Velero backups?

DORS/CLUC
29TH OPEN SYSTEMS DAYS
CROATIAN LINUX USERS' CONFERENCE
16 -17 MAY 2024
Zagreb - Visoko učilište Algebra, Gradiščanska 24

Dario Sindičić

Who are we?

- Government IT company with over 50 years of existence
- Ministry of Finance, Tax administration, Customs Administration, Elections (local, presidential, parliamentary)
- In the current transition into the mostly containerized environment
- And those new stuff also needs backup

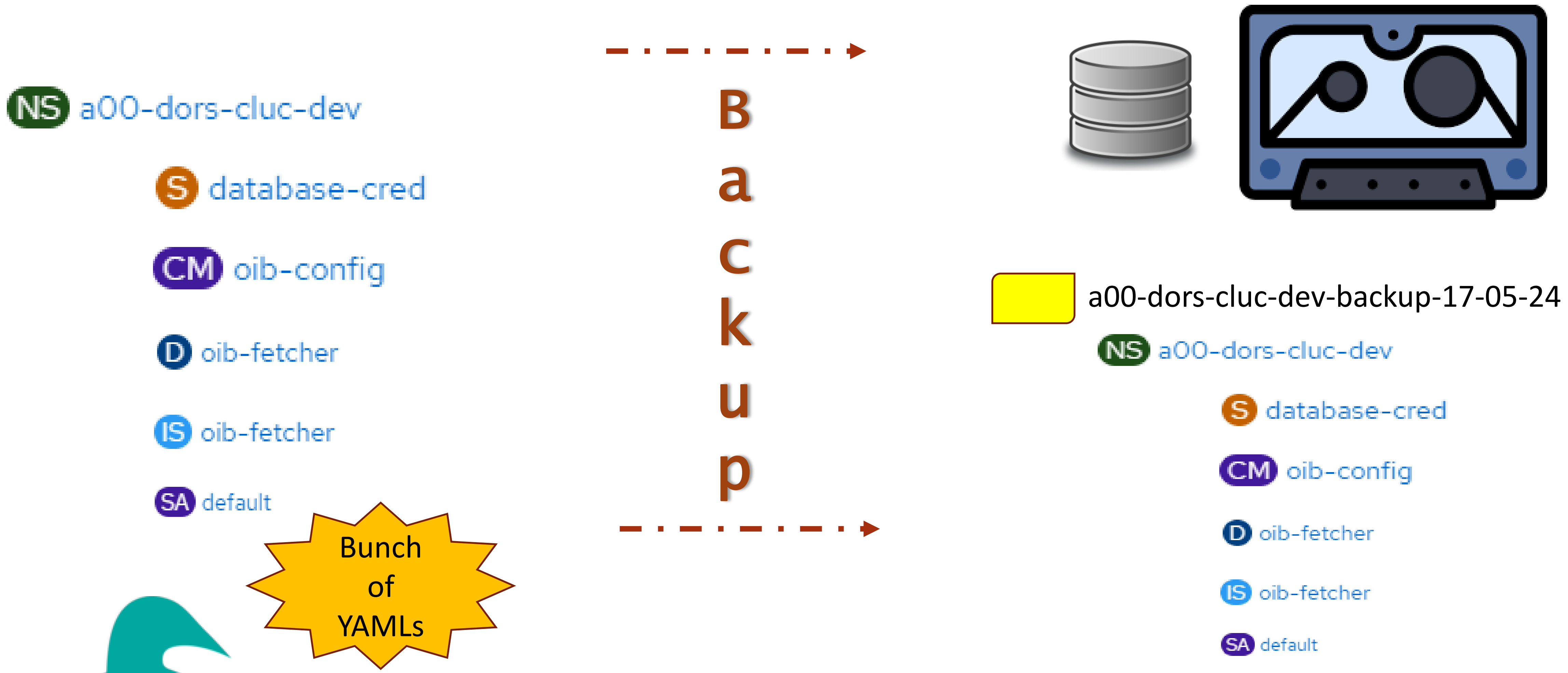
Introduction

```yaml
kind: Namespace
apiVersion: v1
metadata:
  name: a00-dors-cluc-dev
spec:
  finalizers:
    - kubernetes
---
kind: ConfigMap
apiVersion: v1
metadata:
  name: oib-config
  namespace: a00-dors-cluc-dev
immutable: false
data:
  DATABASE: database01.something.com
```

```yaml
kind: Secret
apiVersion: v1
metadata:
  name: database-cred
  namespace: a00-dors-cluc-dev
data:
  DB_PASS: b3BlbnNlc2FtZQ==
  DB_USR: YWRtaW4=
type: Opaque
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: oib-fetch
  namespace: a00-dors-cluc-dev
spec:
  containers:
    - name: oib-fetcher
      image: 'oib-fetcher:latest'
      ports:
        - containerPort: 8080
      envFrom:
        - configMapRef:
            name: oib-config
          secretRef:
            name: database-cred
```

Our project


NS a00-dors-cluc-dev
S database-cred
CM oib-config
D oib-fetcher
IS oib-fetcher
SA default
Bunch of YAMLs
Backup
a00-dors-cluc-dev-backup-17-05-24
NS a00-dors-cluc-dev
S database-cred
CM oib-config
D oib-fetcher
IS oib-fetcher
SA default

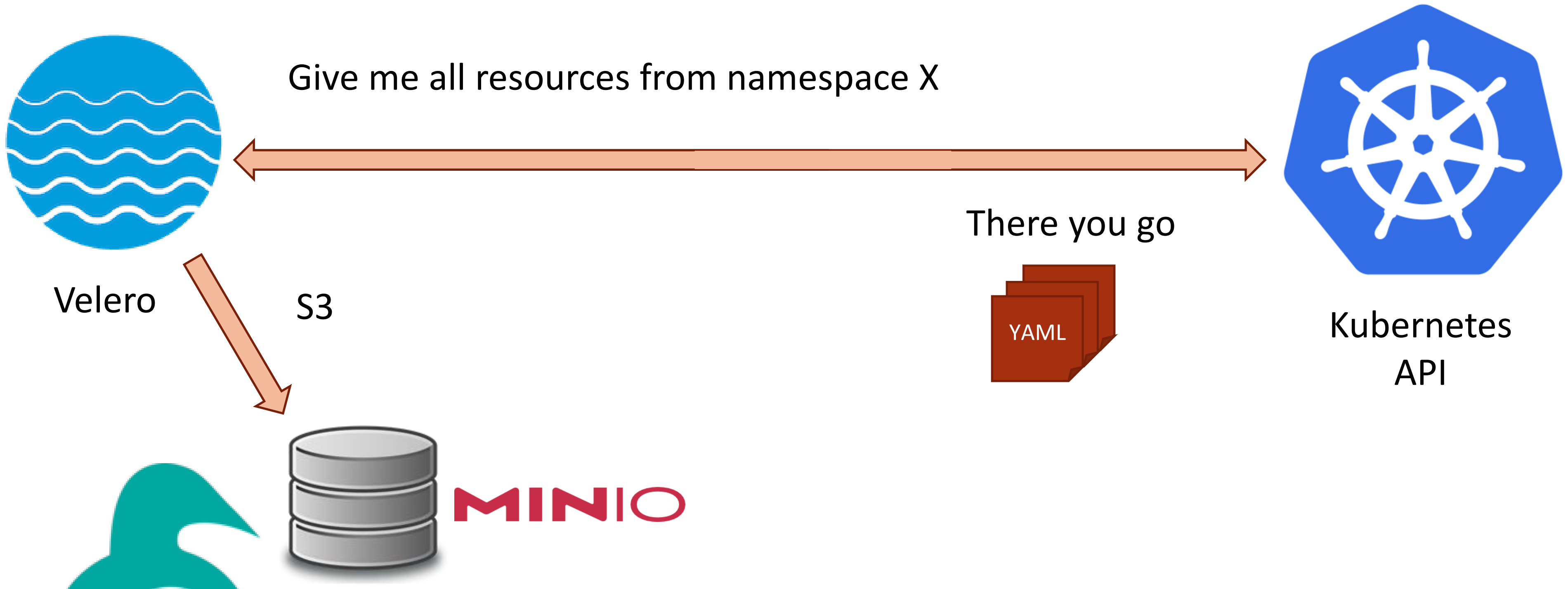
Backup creation

```
sh-4.4$ /velero backup create dors-one \
                             --include-namespaces a00-dors-cluc-dev
Backup request "dors-one" submitted successfully.
Run `velero backup describe dors-one` or `velero backup logs dors-
one` for more details
```

```yaml
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: backup-resources-dors-first-take
spec:
  ...
  includedNamespaces:
    - a00-dors-cluc-dev
  ...
status:
  completionTimestamp: '2024-05-04T15:40:21Z'
  phase: Completed
  progress:
    itemsBackedUp: 45
    totalItems: 45
  startTimestamp: '2024-05-04T15:38:46Z'
```

- small project without any volume
- Total time = 95 seconds
- 412 projects in production ➔ 10 hours and 38 minutes minimal

Architecture



Disk

Velero log: Backup is in memory

```
time="2024-05-05T12:00:45Z" level=info msg="Backed up 45 items out of an estimated total of 45
```

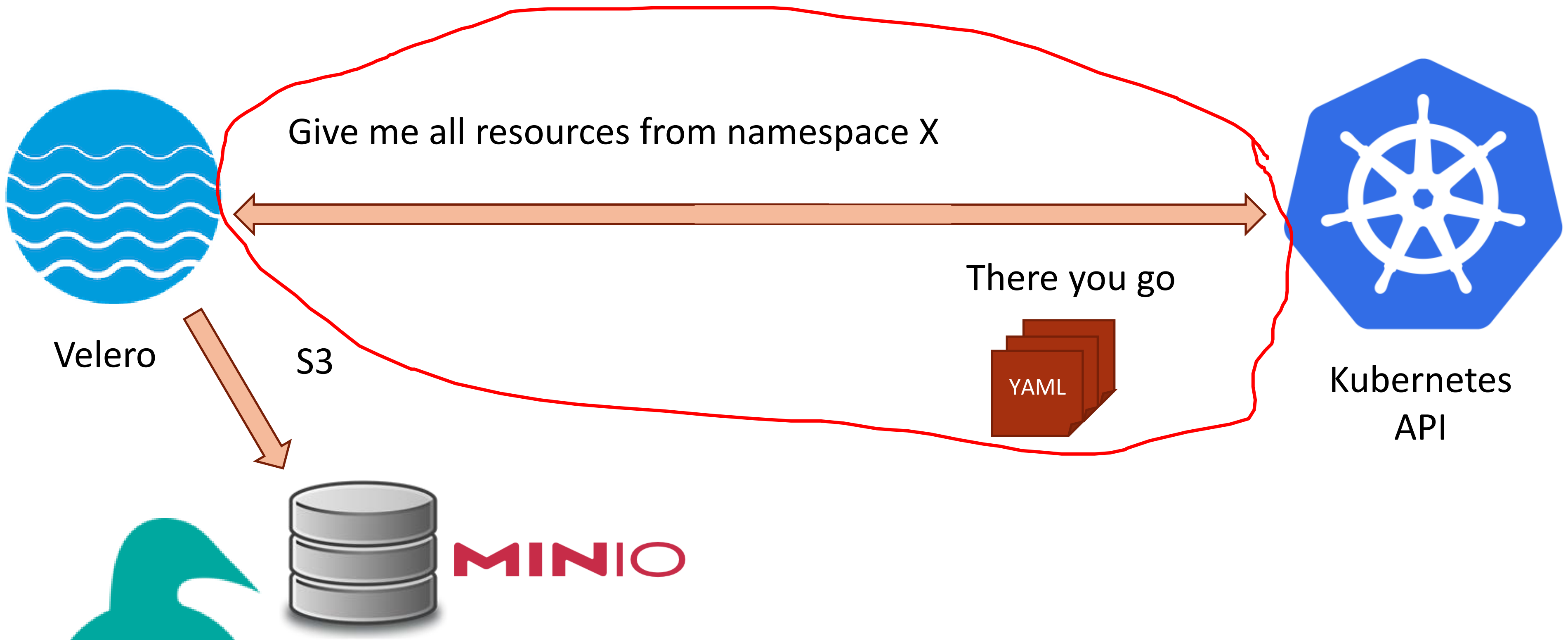
Strace from Velero: TLS handshake ➔ saving data on Minio

```
12:00:45 connect(12, {sa_family=AF_INET, sin_port=htons(9000), sin_addr=inet_addr("172.30.62.27")})
12:00:45 write(12,”\26\3\1\1.\......n\300\24\0\234\0\235\0/\0005\300\22\0\n\23\1\23\2\23\3\1\0\0\273\0\0\0.\0,\0\0)guardian-minio-svc.apis-backup-
restore.svc\0\5\0\5\1\0\0\0\0\0\n\0\n\0\10\0\35\0\27\0\30\0\31\0\v\0\2\1\0\0\r\0\32\0\30\10\......", 307) = 307 <0.002225>
```

Velero log: Backup is saved on Minio

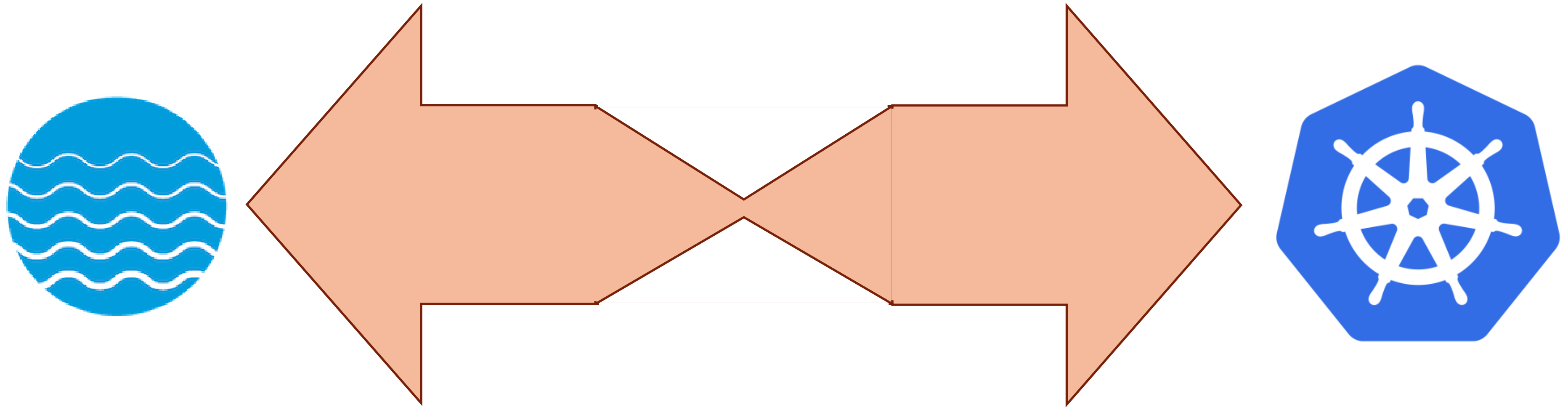
```
time="2024-05-05T12:00:46Z" level=info msg="Backed up 45 items out of an estimated total of 45
```

Architecture



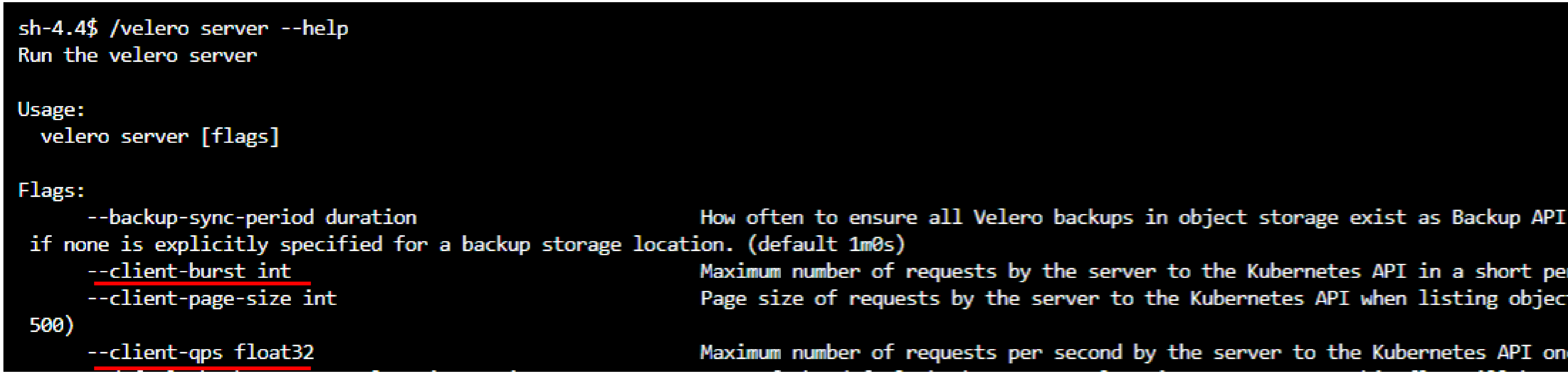
```
9706 request.go:601] Waited for 21.595097708s due to client-side
throttling, not priority and fairness, request:
GET:https://172.30.48.1:443/apis/application.isf.com

9706 request.go:601] Waited for 11.395595197s due to client-side
throttling, not priority and fairness, request:
GET:https://172.30.48.1:443/apis/config.io/v1
```

Client-side throttling



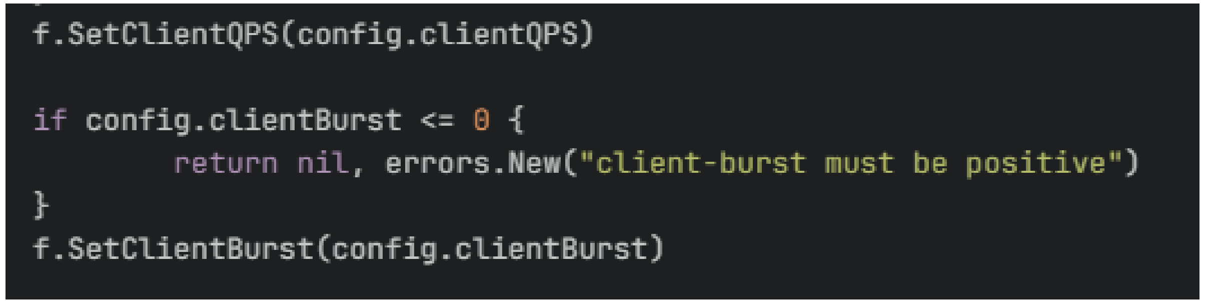
```
const (
        DefaultQPS   float32 = 5.0
        DefaultBurst int     = 10
)
```

client-go/rest/config.go

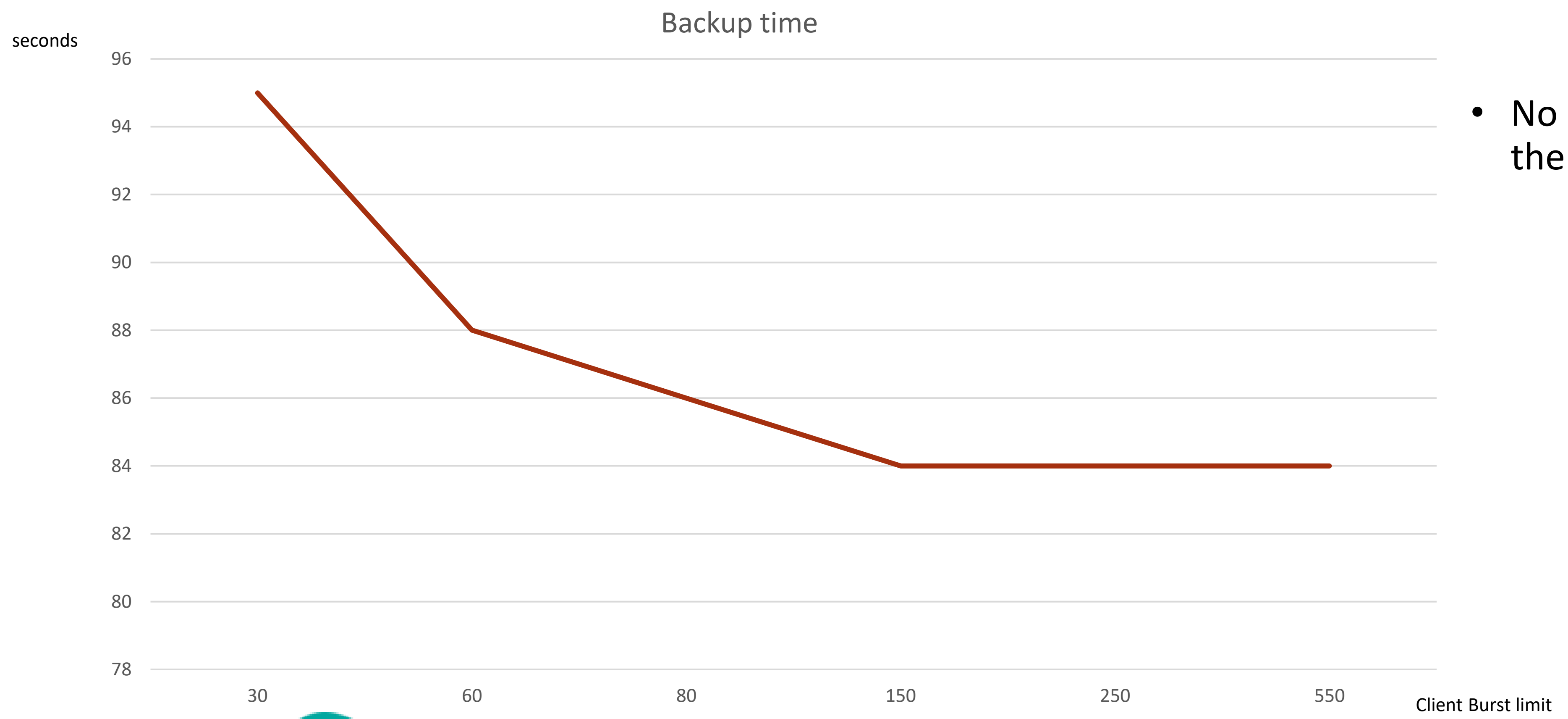


```
sh-4.4$ /velero server --help
Run the velero server

Usage:
  velero server [flags]

Flags:
      --backup-sync-period duration                    How often to ensure all Velero backups in object storage exist as Backup API
 if none is explicitly specified for a backup storage location. (default 1m0s)
      --client-burst int                               Maximum number of requests by the server to the Kubernetes API in a short pe
      --client-page-size int                           Page size of requests by the server to the Kubernetes API when listing objec
 500)
      --client-qps float32                             Maximum number of requests per second by the server to the Kubernetes API on
```

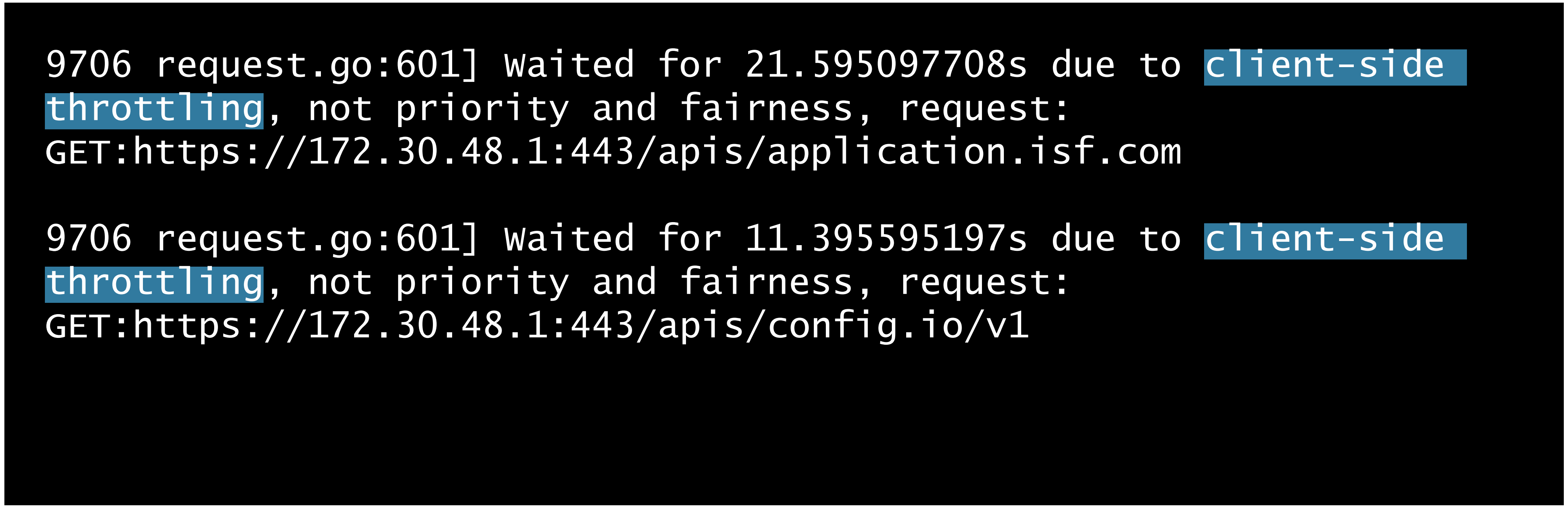
pkg/cmd/server/server.go



```
f.SetClientQPS(config.clientQPS)

if config.clientBurst <= 0 {
        return nil, errors.New("client-burst must be positive")
}
f.SetClientBurst(config.clientBurst)
```

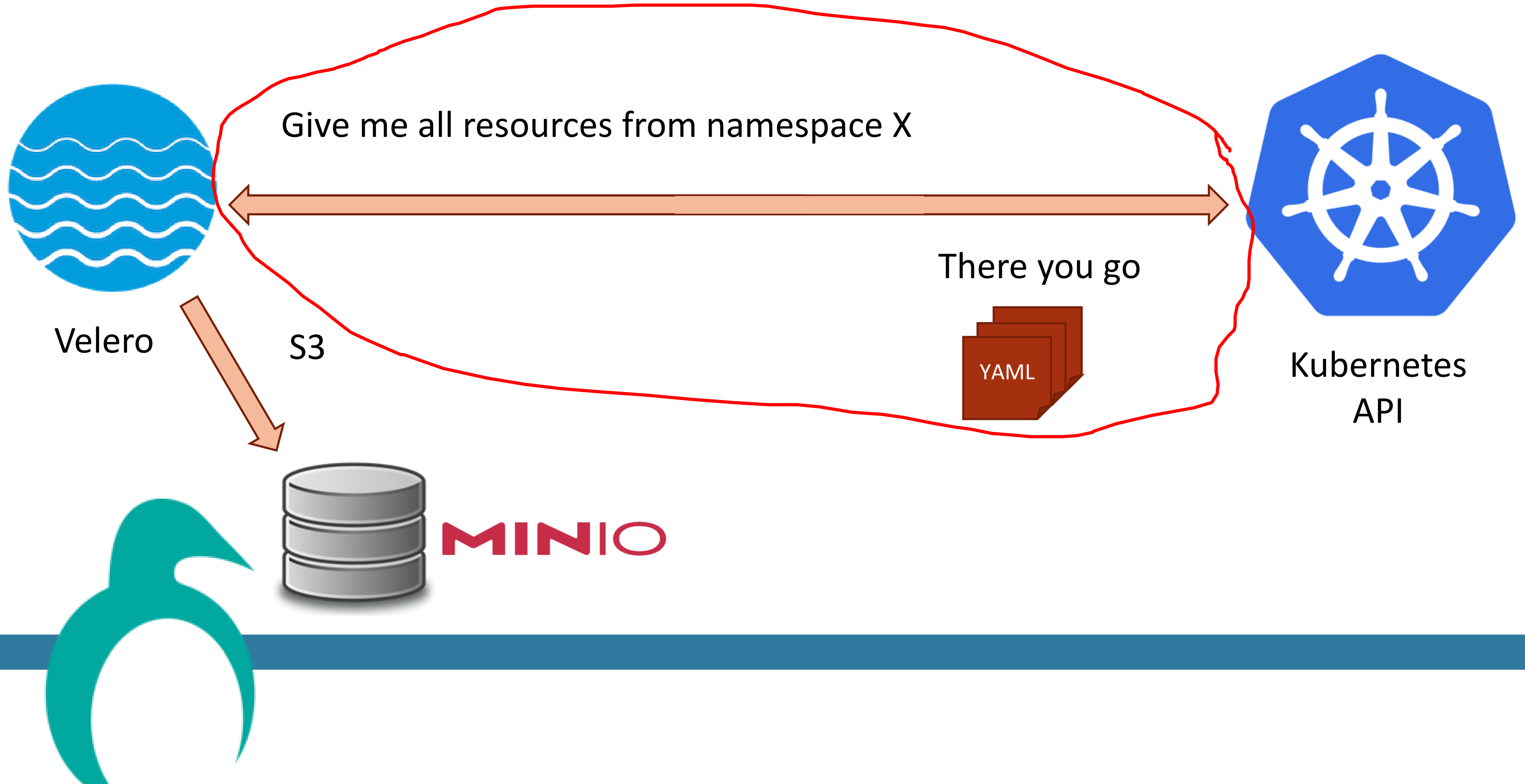


- No significant changes in the backup speed



```
9706 request.go:601] Waited for 21.595097708s due to client-side
throttling, not priority and fairness, request:
GET:https://172.30.48.1:443/apis/application.isf.com

9706 request.go:601] Waited for 11.395595197s due to client-side
throttling, not priority and fairness, request:
GET:https://172.30.48.1:443/apis/config.io/v1
```

Architecture



```yaml
kind: Namespace
apiVersion: v1
metadata:
  name: a00-dors-cluc-dev
spec:
  finalizers:
    - kubernetes
---
kind: ConfigMap
apiVersion: v1
metadata:
  name: oib-config
  namespace: a00-dors-cluc-dev
immutable: false
data:
  DATABASE: database01.something.com
```

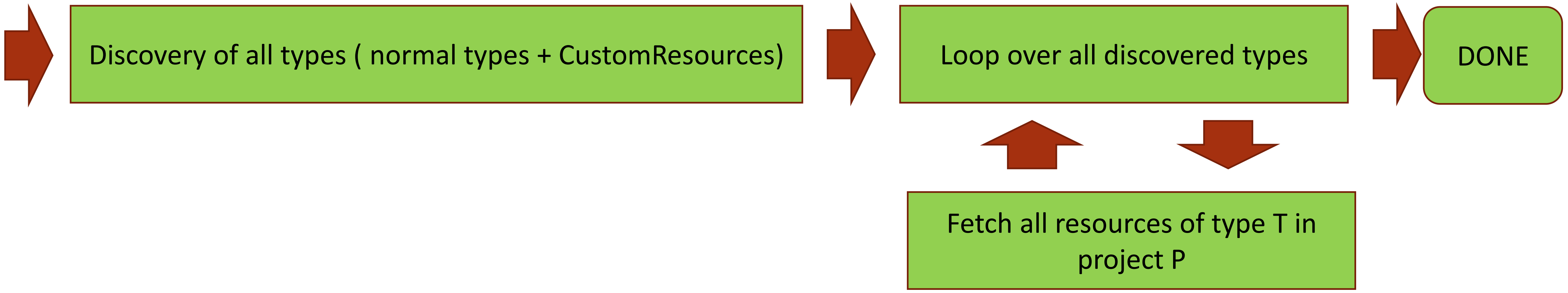
CUSTOM RESOURCE

```yaml
kind: DorsCluc
apiVersion: v1
metadata:
    name: dors
    namespace: cluc
spec:
    events:
      - year: 2024
        location: Algreba
      - year: 2023
        location: FER
      - year: 2013
        location: Hotel International
```

```yaml
v1

fetch
 a00-dors-cluc-dev

:
oib-fetcher
 'oib-fetcher:latest'

ntainerPort: 8080
m:
  - configMapRef:
      name: oib-config
    secretRef:
        name: database-cred
```

Velero steps

VELERO


Discovery of all types (normal types + CustomResources)
Loop over all discovered types
DONE
Fetch all resources of type T in project P

Analysis of API calls

Total number of API calls: 284

Number of API calls related to the backup 82 187 15

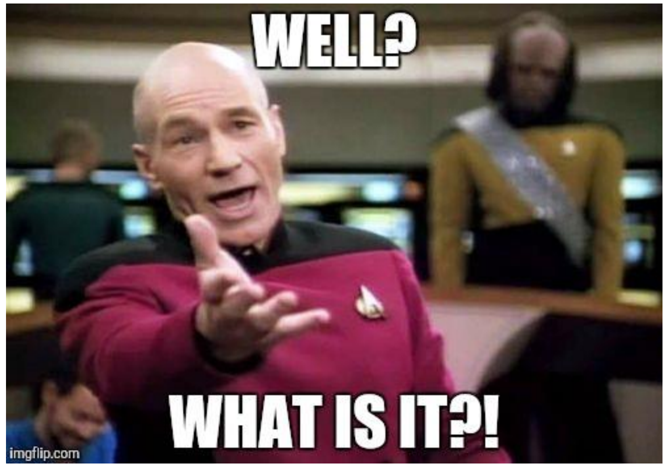
14:20:54
Start

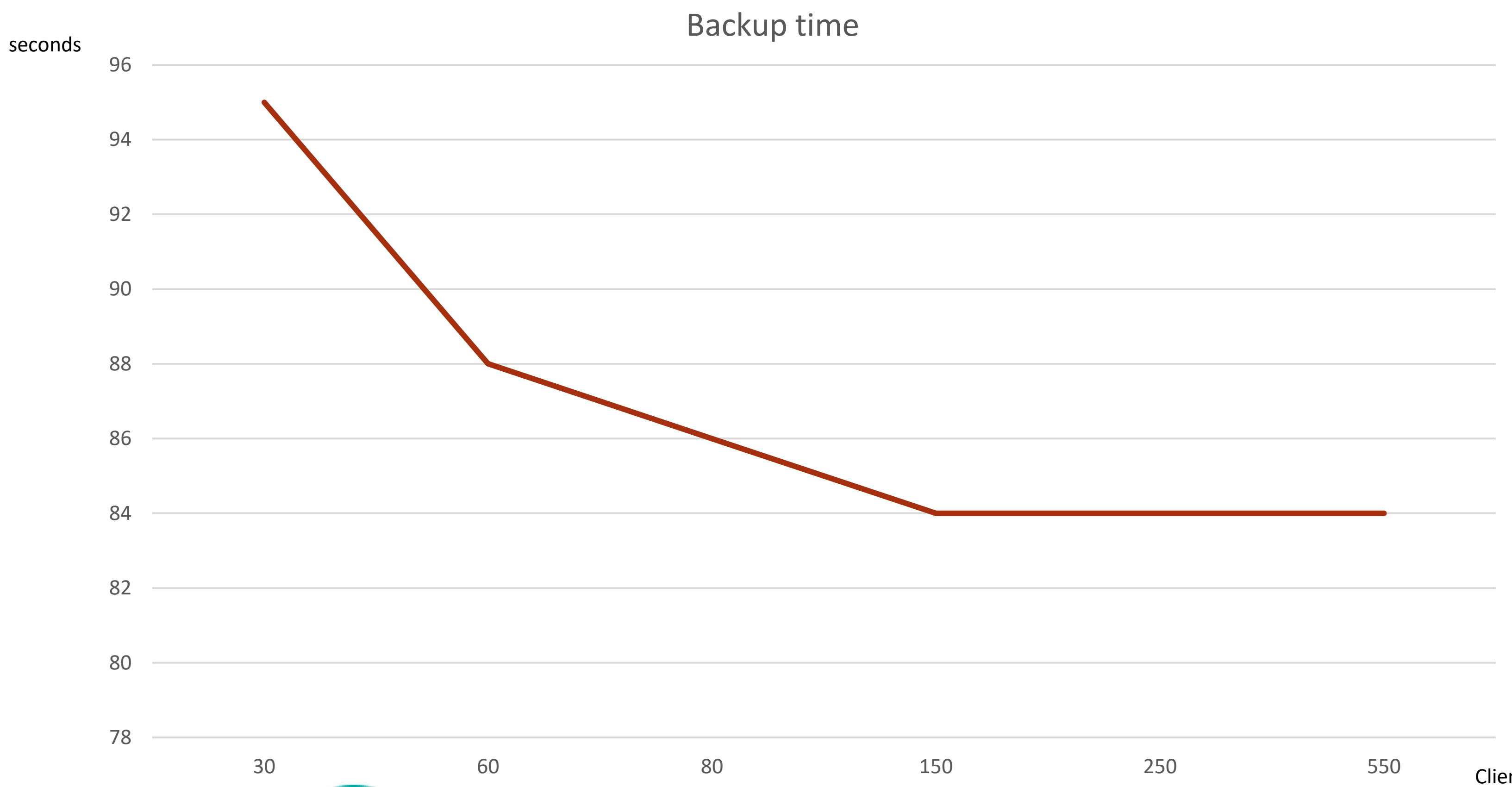
14:22:16

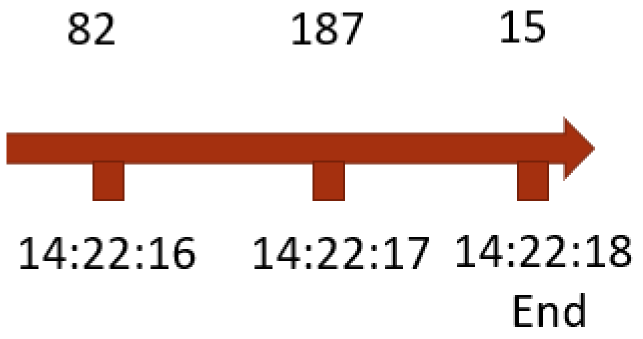
14:22:17

14:22:18
End

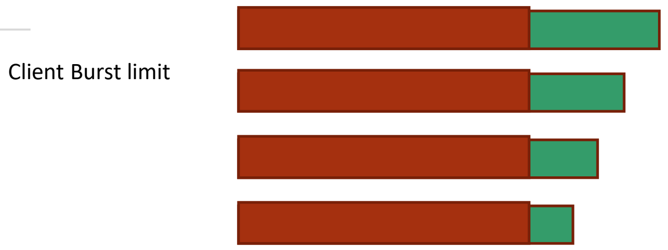
82 seconds of doing "something"



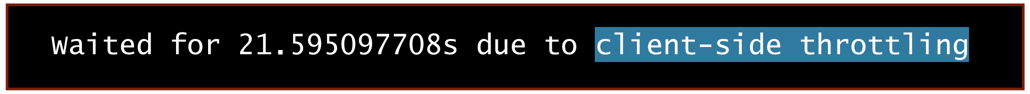




- 284 calls
- 30 r/s ➔ 284 calls ➔ 10 sec
- 60 r/s ➔ 284 calls ➔ 4.5 sec
- 150 r/s ➔ 284 calls ➔ 2 sec
- 250 r/s ➔ 284 calls ➔ 1.5 sec

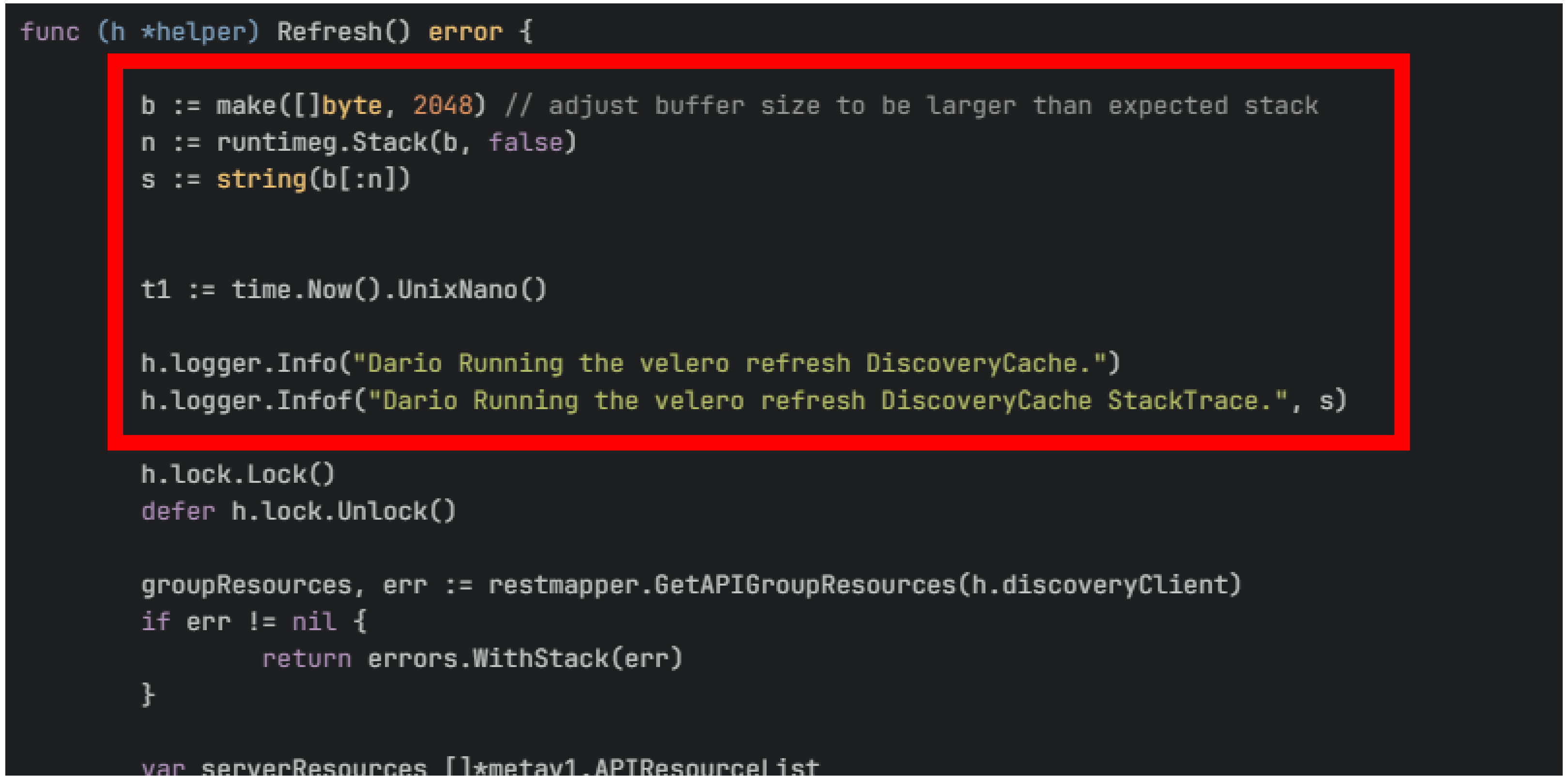


Doing „something”

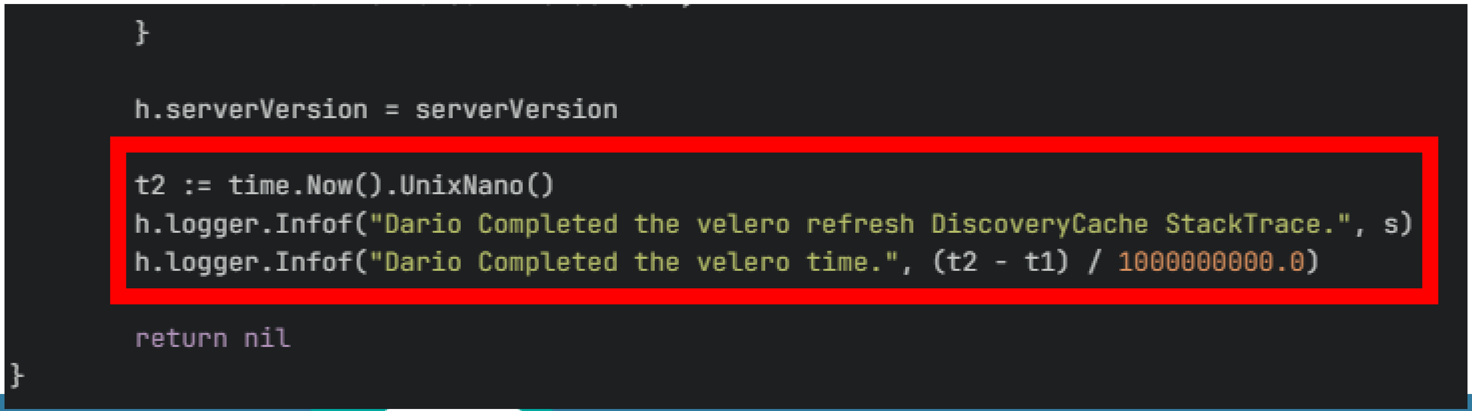

```
Waited for 21.595097708s due to client-side throttling
```

- Probbably client-side throtling are caused by discovery of all types
- Let’s fork the code and see what to do with it

https://github.com/vmware-tanzu/velero



```go
func (h *helper) Refresh() error {
        b := make([]byte, 2048) // adjust buffer size to be larger than expected stack
        n := runtimeg.Stack(b, false)
        s := string(b[:n])


        t1 := time.Now().UnixNano()

        h.logger.Info("Dario Running the velero refresh DiscoveryCache.")
        h.logger.Infof("Dario Running the velero refresh DiscoveryCache StackTrace.", s)
        h.lock.Lock()
        defer h.lock.Unlock()

        groupResources, err := restmapper.GetAPIGroupResources(h.discoveryClient)
        if err != nil {
                return errors.WithStack(err)
        }

        var serverResources []*metav1.APIResourceList
```



```go
        }

        h.serverVersion = serverVersion

        t2 := time.Now().UnixNano()
        h.logger.Infof("Dario Completed the velero refresh DiscoveryCache StackTrace.", s)
        h.logger.Info("Dario Completed the velero time.", (t2 - t1) / 1000000000.0)

        return nil
}
```

pkg/discovery/helper.go

- Let’s add some debug prints to see what is going on

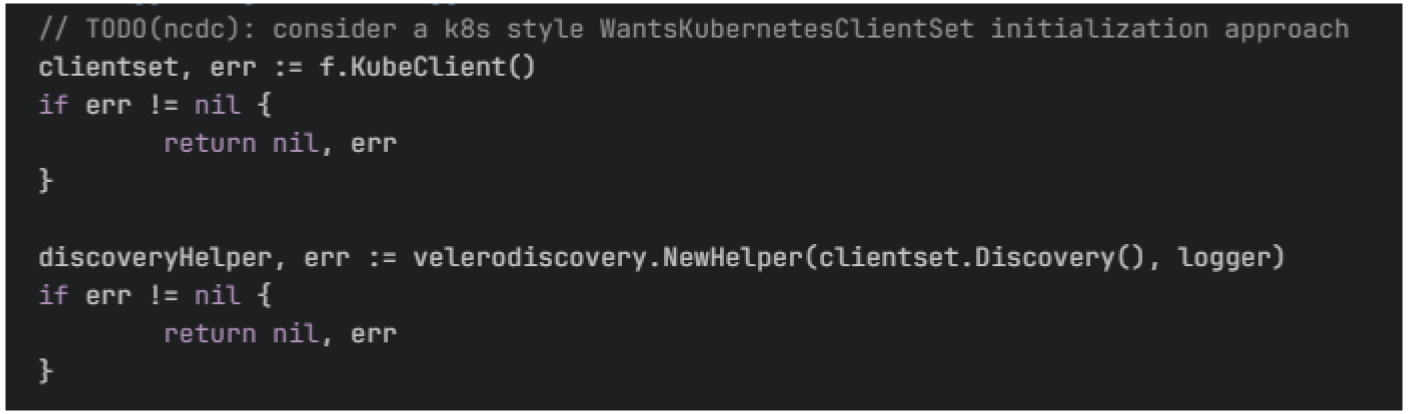
```
14:20:54Z" level=info msg="Dario Running the velero refresh
DiscoveryCache.

14:21:35Z" level=info msg="Dario Completed the velero time 41 seconds"
```

```
14:21:35Z" level=info msg="Dario Running the velero refresh
DiscoveryCache.


14:22:16Z" level=info msg="Dario Completed the velero time 41 seconds"
```

82 seconds of doing "something„
41 seconds each

- 2 discovery searches was run by the Velero
- 457 api resources => ~ 10 qps

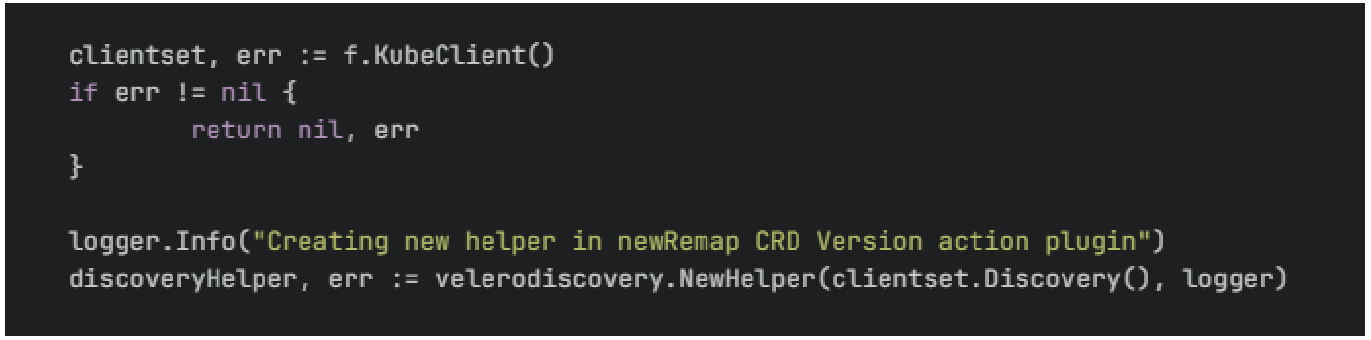
Plugin



```go
// TODO(ncdc): consider a k8s style WantsKubernetesClientSet initialization approach
clientset, err := f.KubeClient()
if err != nil {
        return nil, err
}

discoveryHelper, err := velerodiscovery.NewHelper(clientset.Discovery(), logger)
if err != nil {
        return nil, err
}
```

pkg/cmd/server/plugin/plugin.go

newServiceAccountBackupItemAction function

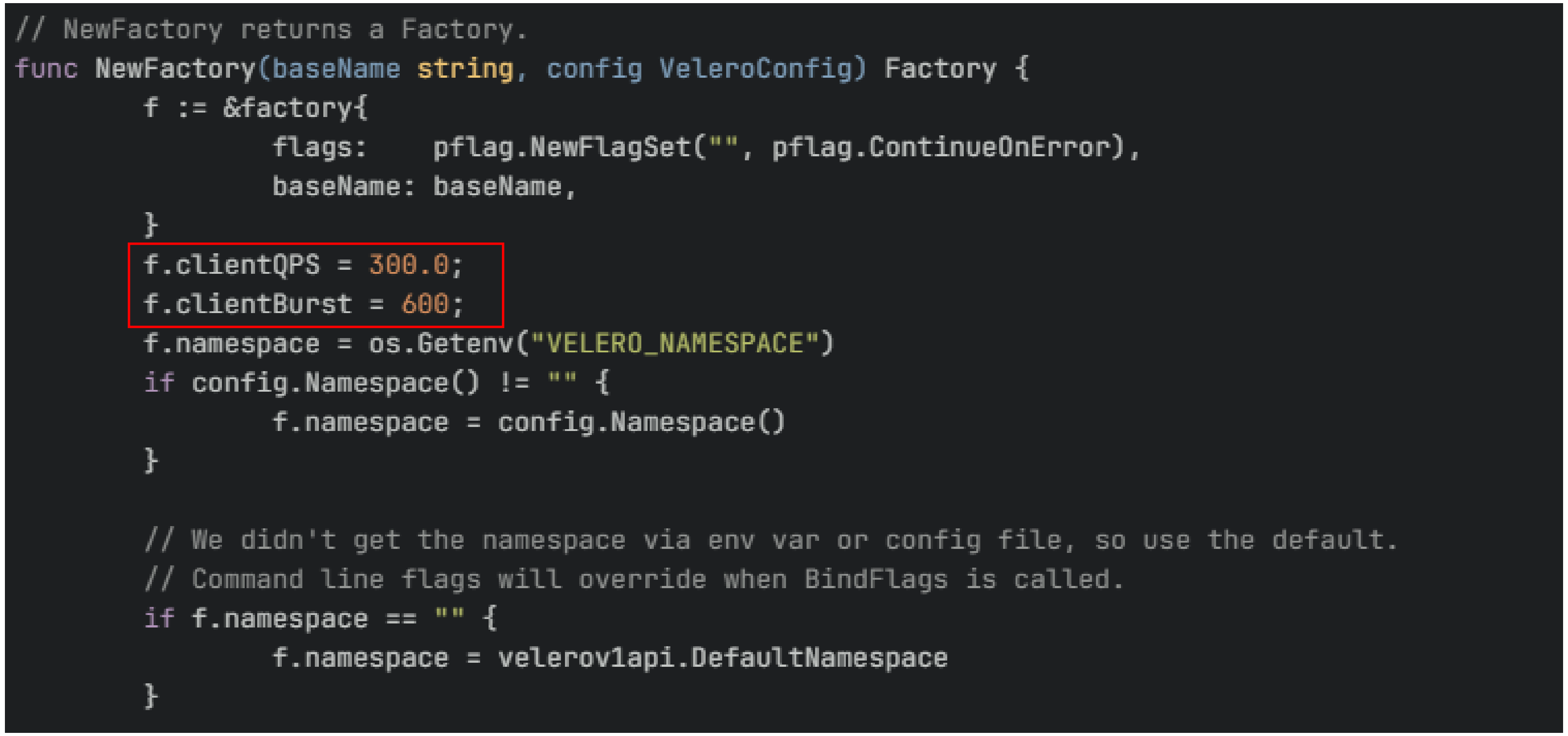


```go
clientset, err := f.KubeClient()
if err != nil {
        return nil, err
}

logger.Info("Creating new helper in newRemap CRD Version action plugin")
discoveryHelper, err := velerodiscovery.NewHelper(clientset.Discovery(), logger)
```

newRemapCRDVersionAction function

- Plugin is ran in seperate process so this factory has different settings than default Velero one
- It is using really low QPS and burst parameters

pkg/client/factory.go



```go
// NewFactory returns a Factory.
func NewFactory(baseName string, config VeleroConfig) Factory {
	f := &factory{
		flags:    pflag.NewFlagSet("", pflag.ContinueOnError),
		baseName: baseName,
	}
	f.clientQPS = 300.0;
	f.clientBurst = 600;
	f.namespace = os.Getenv("VELERO_NAMESPACE")
	if config.Namespace() != "" {
		f.namespace = config.Namespace()
	}

	// We didn't get the namespace via env var or config file, so use the default.
	// Command line flags will override when BindFlags is called.
	if f.namespace == "" {
		f.namespace = velerov1api.DefaultNamespace
	}
```

End results

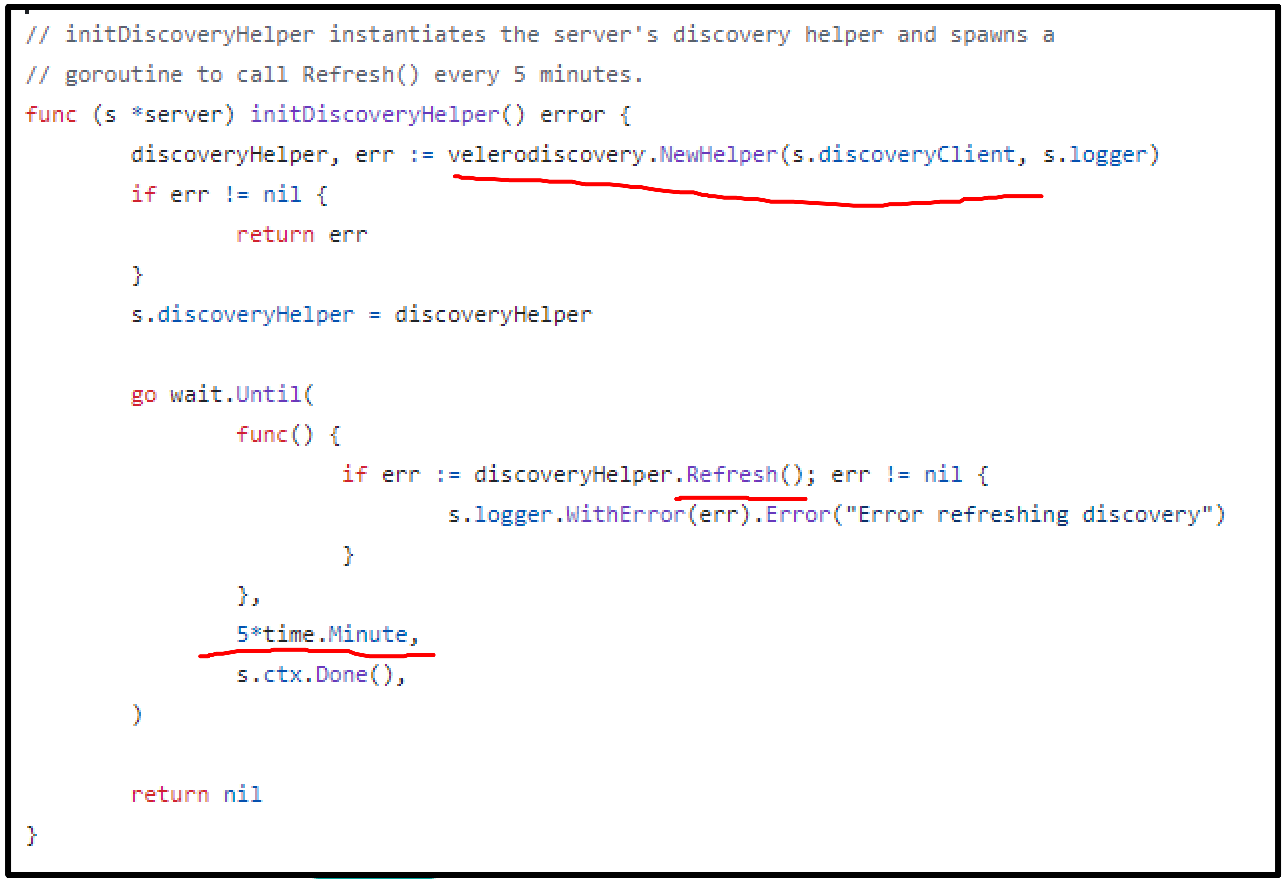
- Backup time reduced to only 2 seconds
- Total backup of production with only one instance of Velero = 14 minutes

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: backup-resources-dors-n-th-take
spec:
  ...
  includedNamespaces:
    - a00-dors-cluc-dev
  ...
status:
  completionTimestamp: '2024-05-10T16:49:46Z'
  phase: Completed
  progress:
    itemsBackedUp: 45
    totalItems: 45
  startTimestamp: '2024-05-10T16:49:44Z'
```

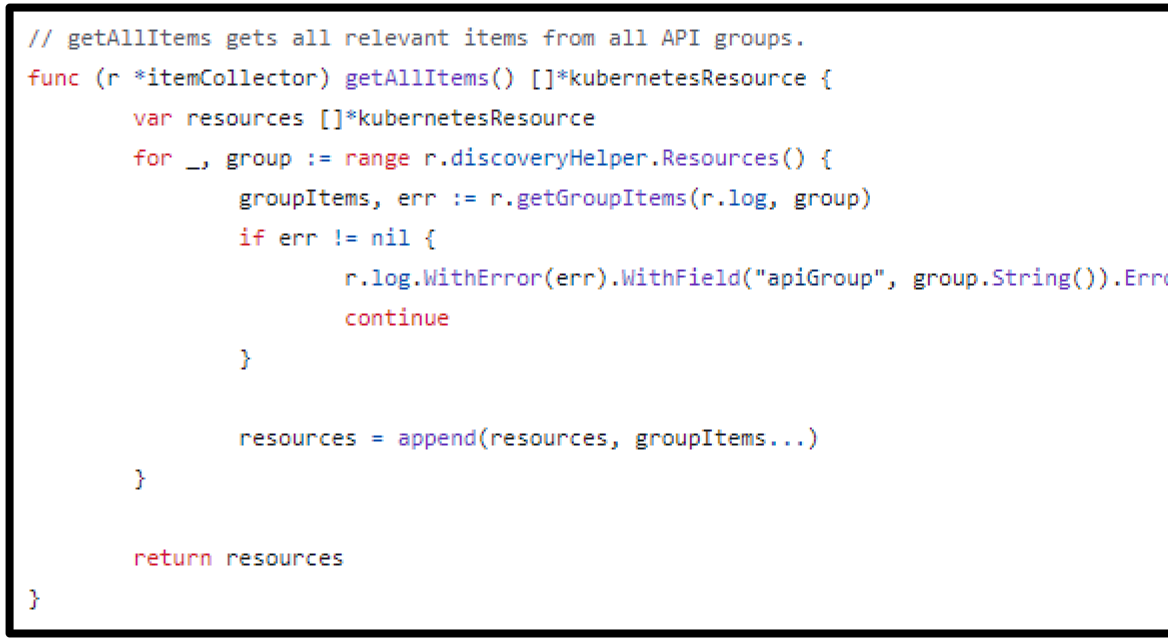
You can't have two backups!
Of course I can.
It's just good sense
to back up your backup.

Thank you !
☺

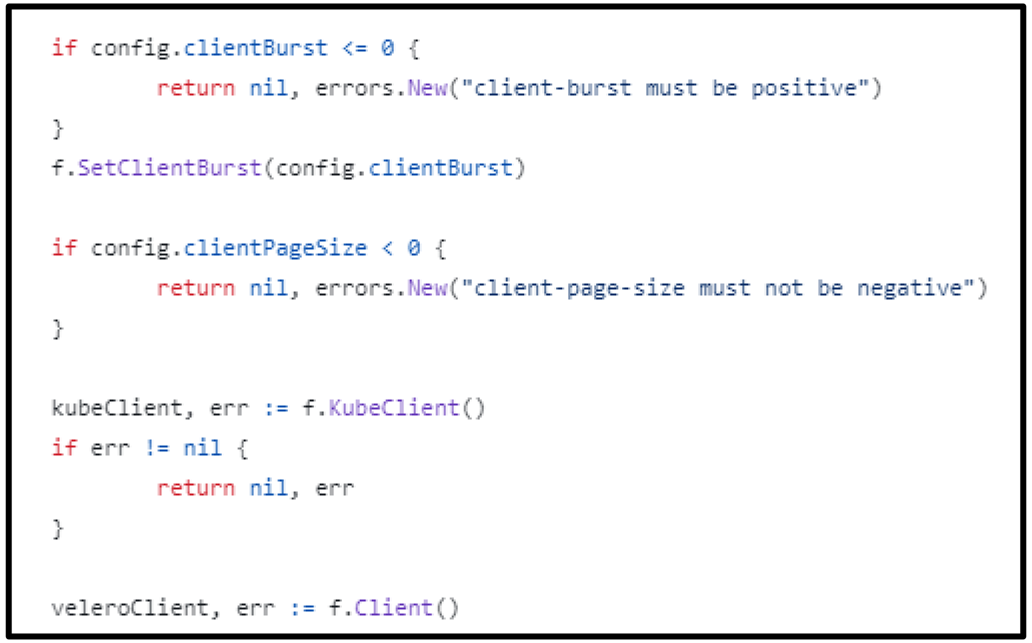
Appendix



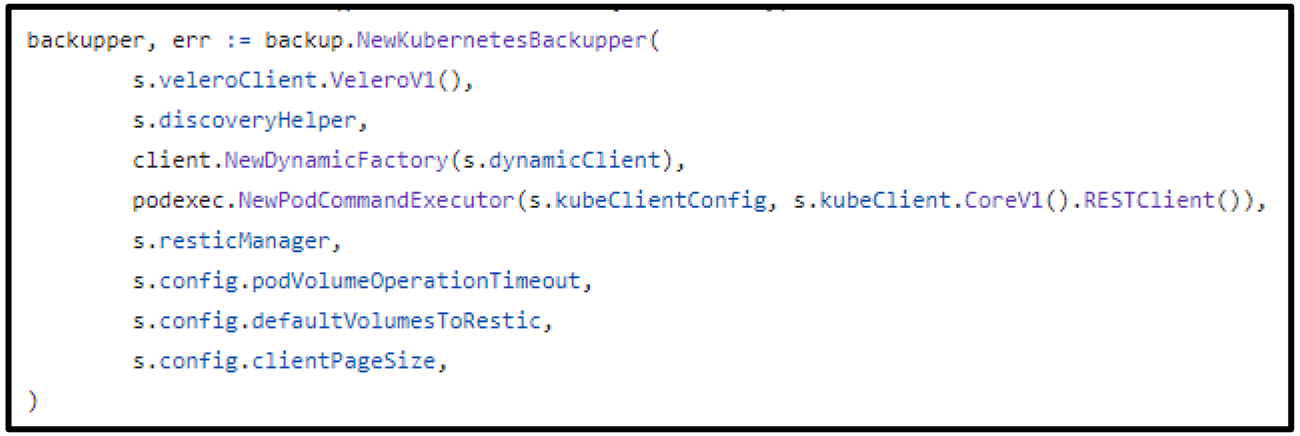
```go
// initDiscoveryHelper instantiates the server's discovery helper and spawns a
// goroutine to call Refresh() every 5 minutes.
func (s *server) initDiscoveryHelper() error {
	discoveryHelper, err := velerodiscovery.NewHelper(s.discoveryClient, s.logger)
	if err != nil {
		return err
	}
	s.discoveryHelper = discoveryHelper

	go wait.Until(
		func() {
			if err := discoveryHelper.Refresh(); err != nil {
				s.logger.WithError(err).Error("Error refreshing discovery")
			}
		},
		5*time.Minute,
		s.ctx.Done(),
	)

	return nil
}
```



```go
// getAllItems gets all relevant items from all API groups.
func (r *itemCollector) getAllItems() []*kubernetesResource {
	var resources []*kubernetesResource
	for _, group := range r.discoveryHelper.Resources() {
		groupItems, err := r.getGroupItems(r.log, group)
		if err != nil {
			r.log.WithError(err).WithField("apiGroup", group.String()).Erro
			continue
		}

		resources = append(resources, groupItems...)
	}

	return resources
}
```



```go
if config.clientBurst <= 0 {
        return nil, errors.New("client-burst must be positive")
}
f.SetClientBurst(config.clientBurst)

if config.clientPageSize < 0 {
        return nil, errors.New("client-page-size must not be negative")
}

kubeClient, err := f.KubeClient()
if err != nil {
        return nil, err
}

veleroClient, err := f.Client()
```



```go
backupper, err := backup.NewKubernetesBackupper(
        s.veleroClient.VeleroV1(),
        s.discoveryHelper,
        client.NewDynamicFactory(s.dynamicClient),
        podexec.NewPodCommandExecutor(s.kubeClientConfig, s.kubeClient.CoreV1().RESTClient()),
        s.resticManager,
        s.config.podVolumeOperationTimeout,
        s.config.defaultVolumesToRestic,
        s.config.clientPageSize,
)
```