Live migration

Reducing downtime with multichassis port bindings

Ihar Hrachyshka

Red Hat OpenStack Networking Team

What is live migration?

Live migration [is a] process of moving a running virtual machine [...] between different physical machines without disconnecting the client or application. Memory, storage, and network connectivity [...] are transferred from the original guest machine to the destination.

https://en.wikipedia.org/wiki/Live_migration

Why?

- evacuate for maintenance
- deprovision unused hosts
- balance the load

How is a VM migrated?

How is a VM migrated?

- Copy memory
- Pause original VM
- “Unpause” new VM
- Move port bindings

How is a VM migrated?

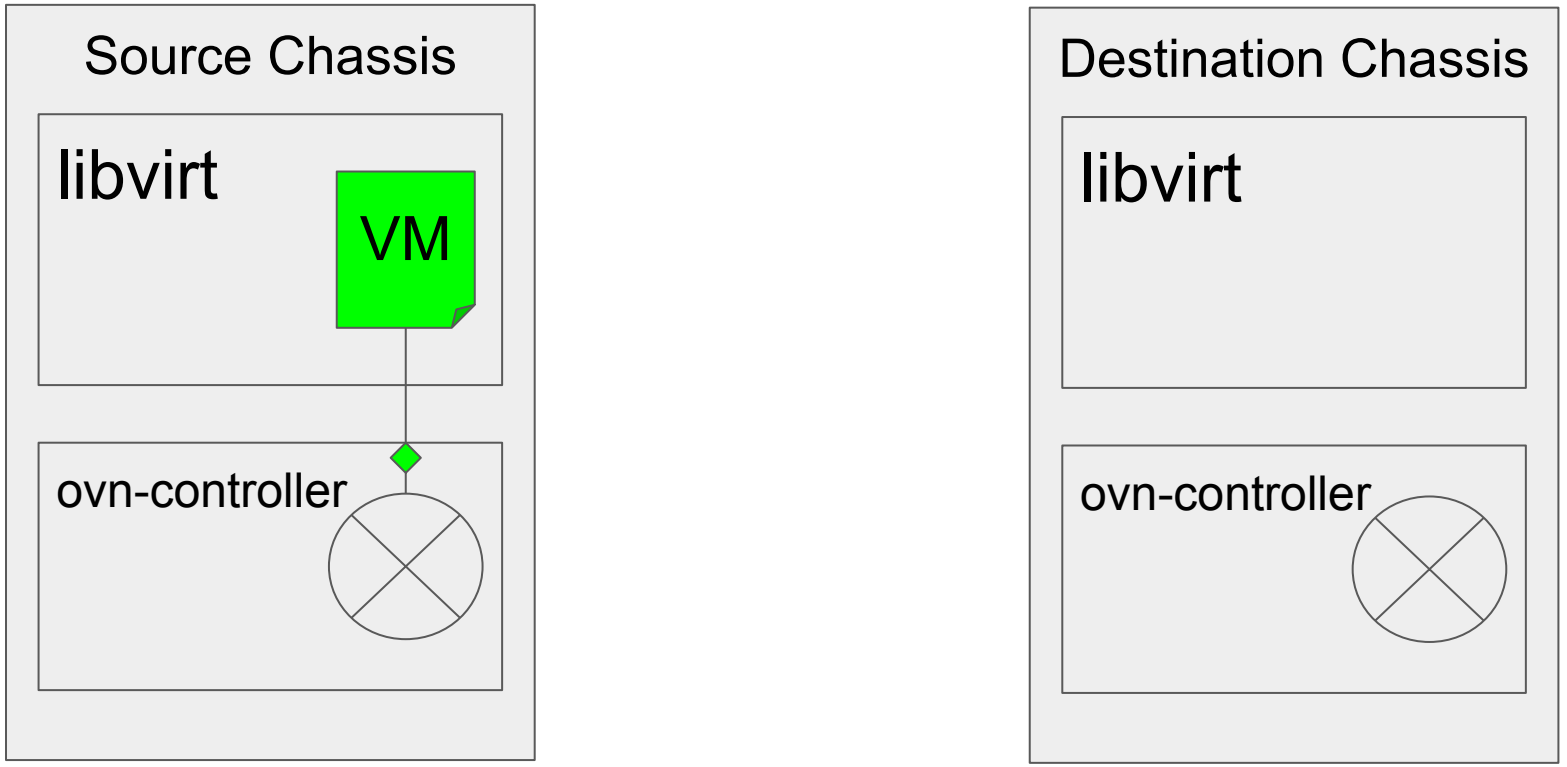
- Copy memory
- Pause original VM
- “Unpause” new VM
- Move port bindings

How is a VM migrated?

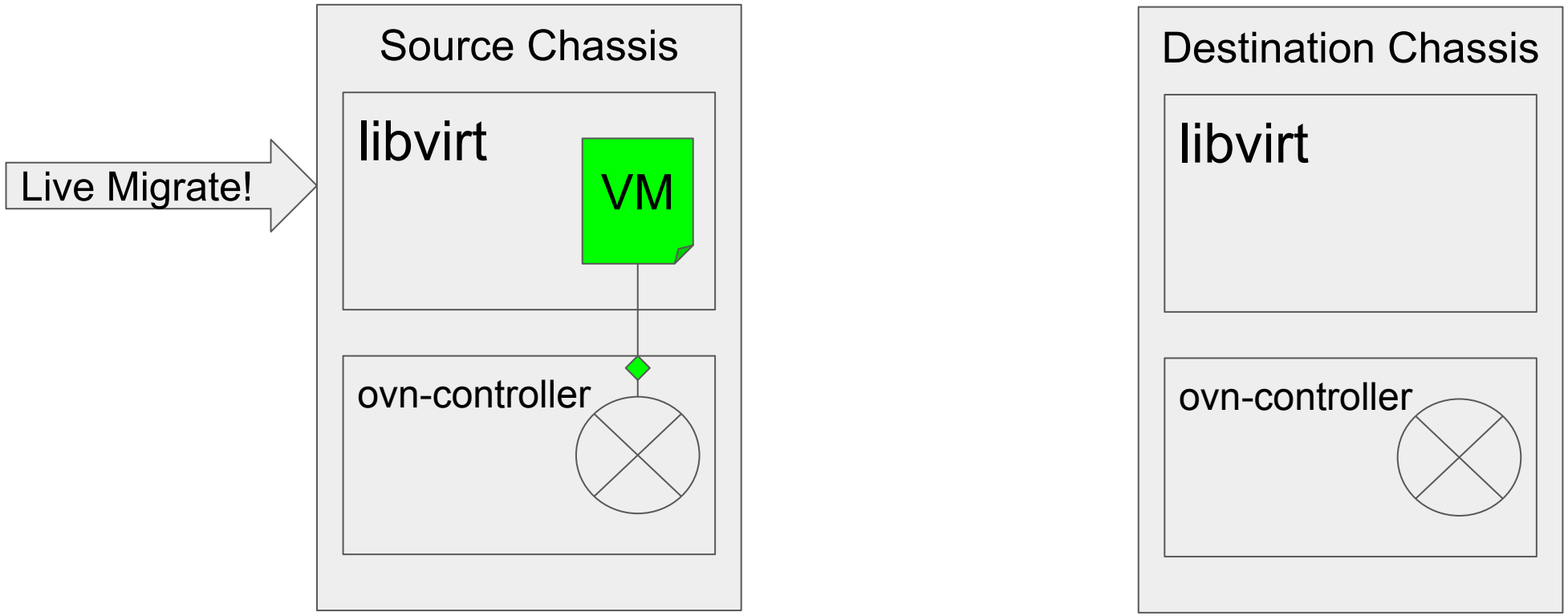
- Copy memory
- Pause original VM
- “Unpause” new VM
- Move port bindings

How is a VM migrated?

- Copy memory
- Pause original VM
- “Unpause” new VM
- Move port bindings

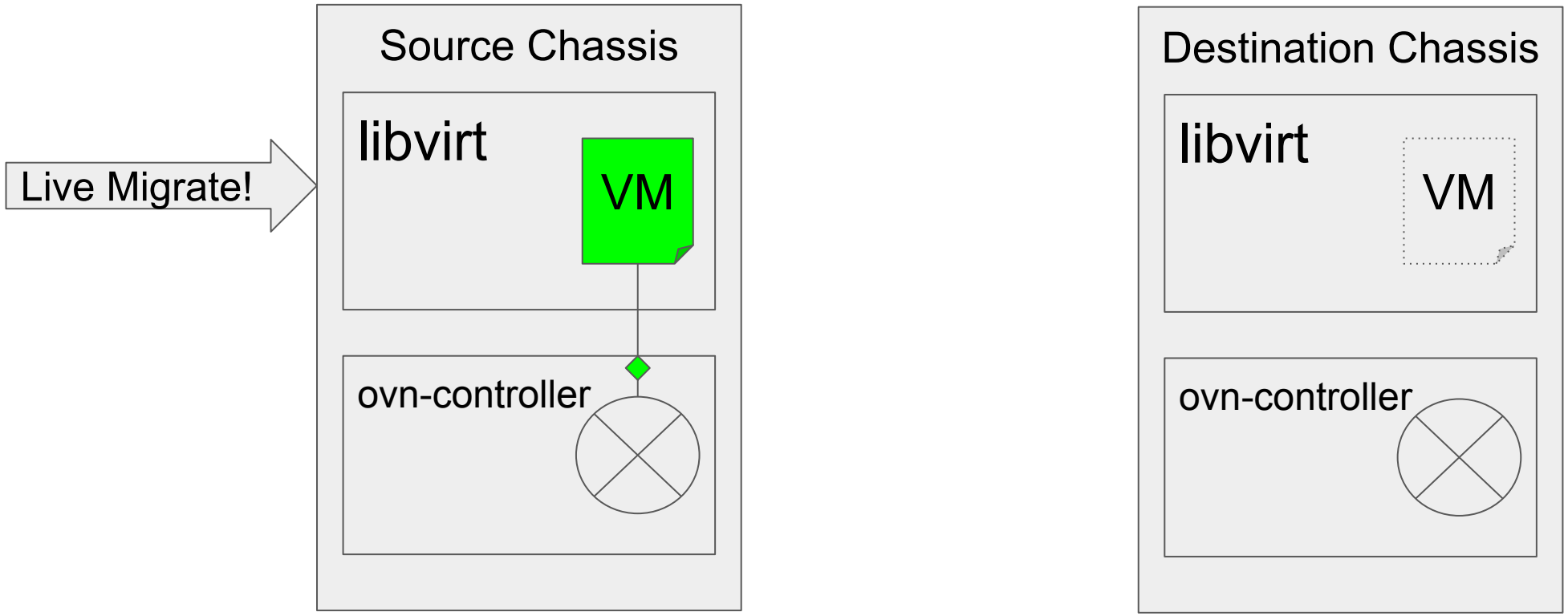


options:requested-chassis=source

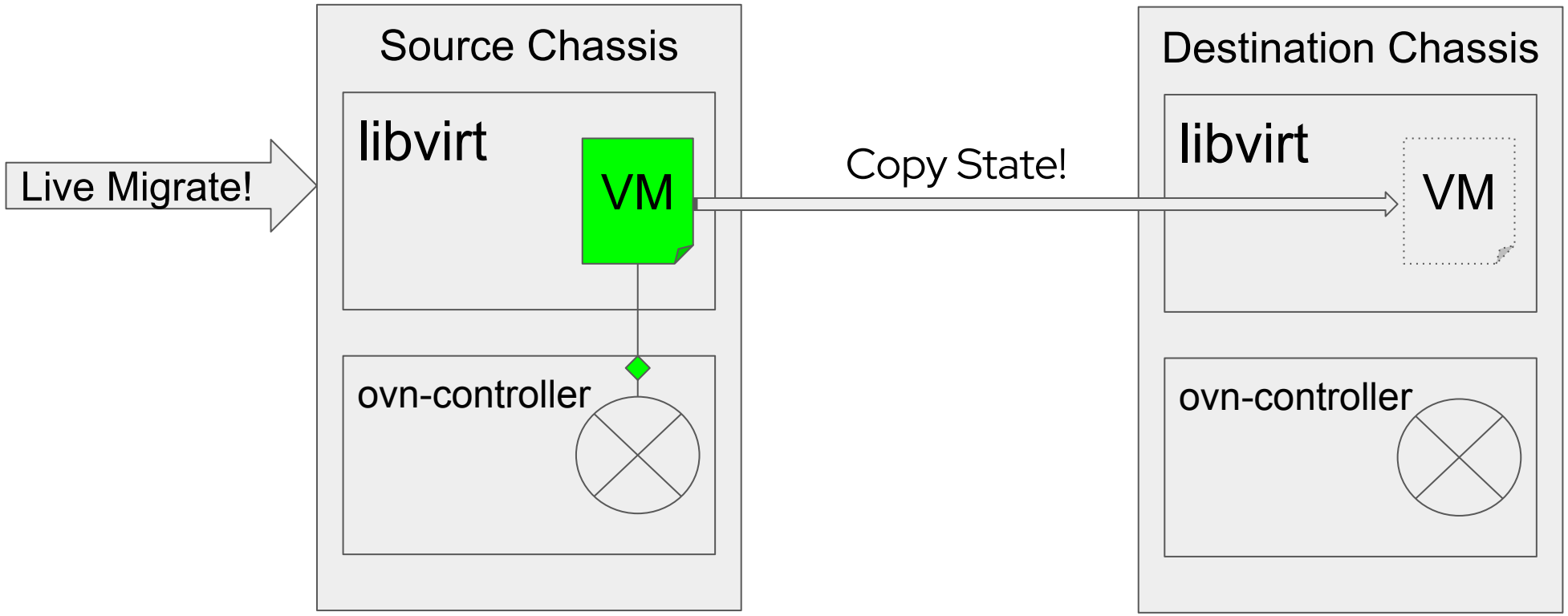


options:requested-chassis=source

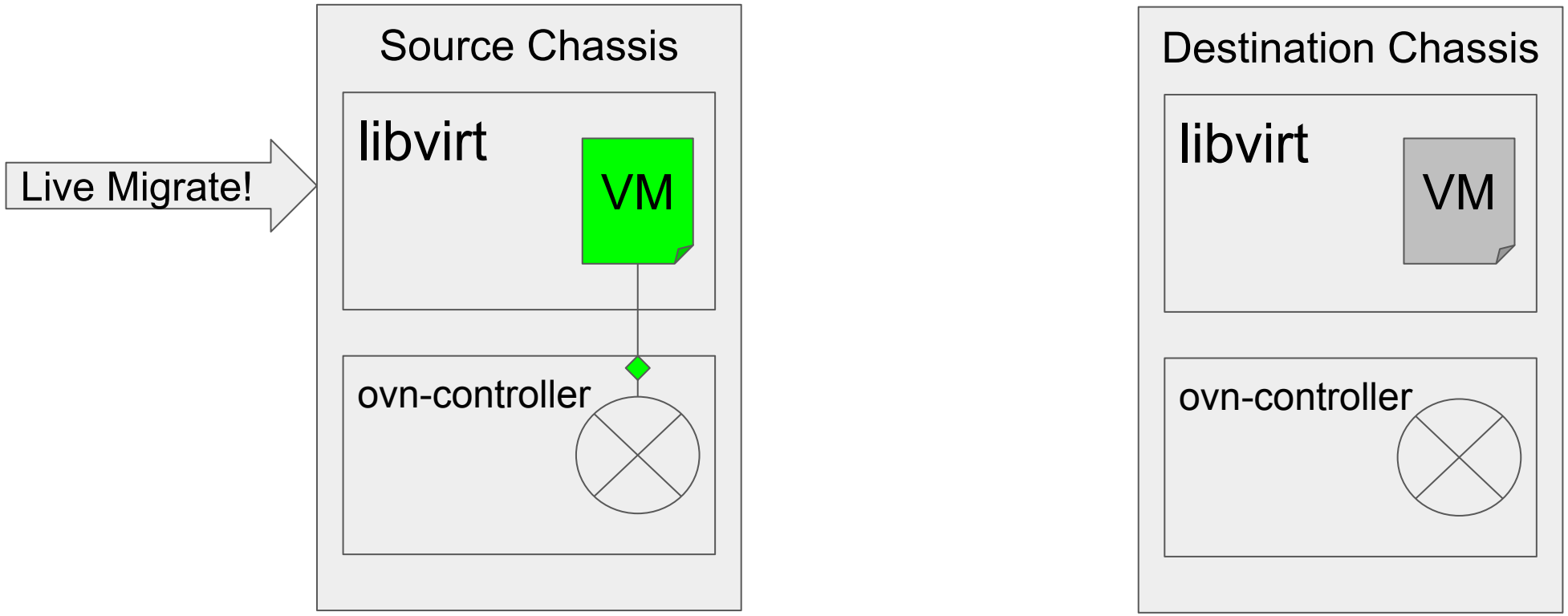
Red Hat



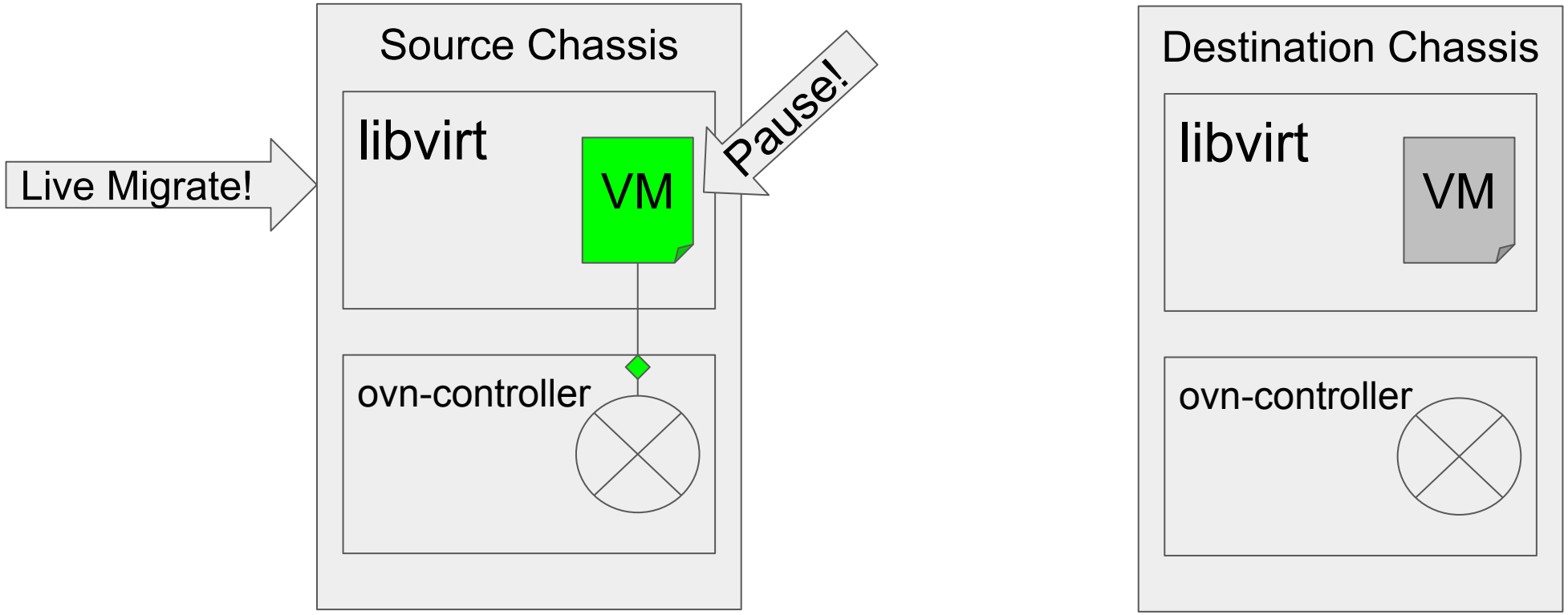
options:requested-chassis=source



options:requested-chassis=source

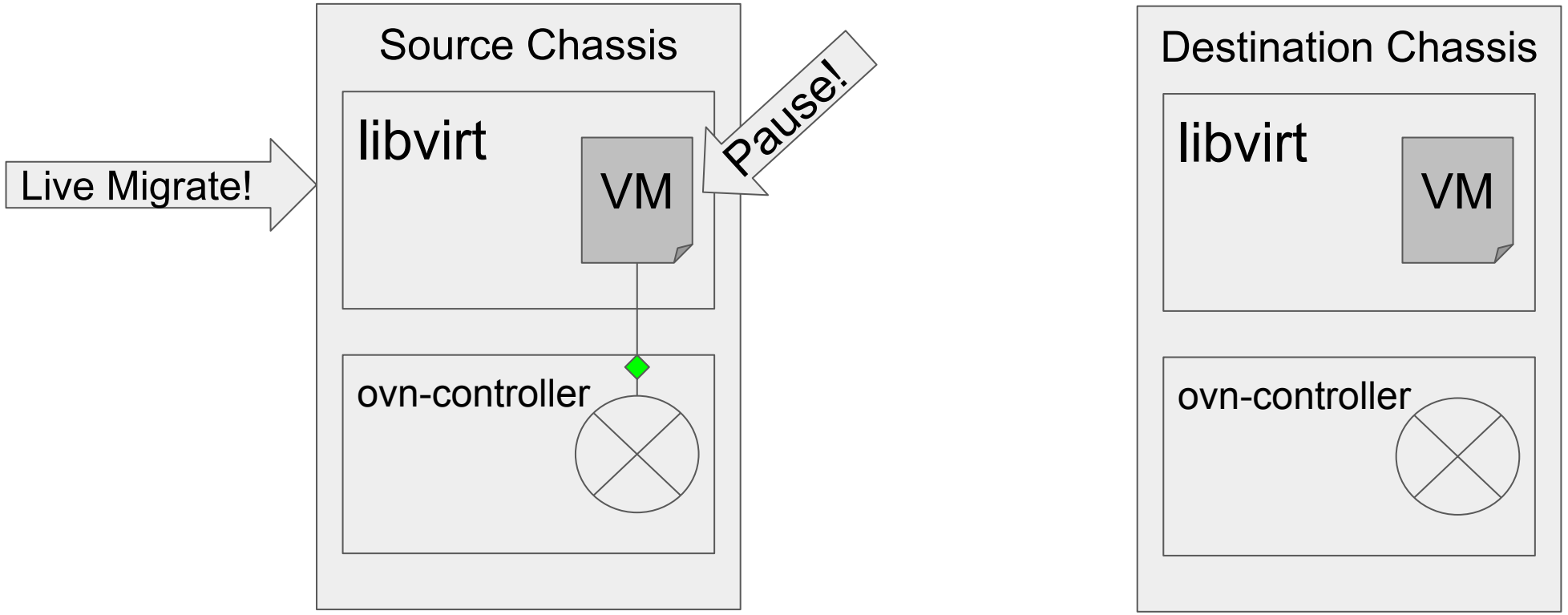


options:requested-chassis=source



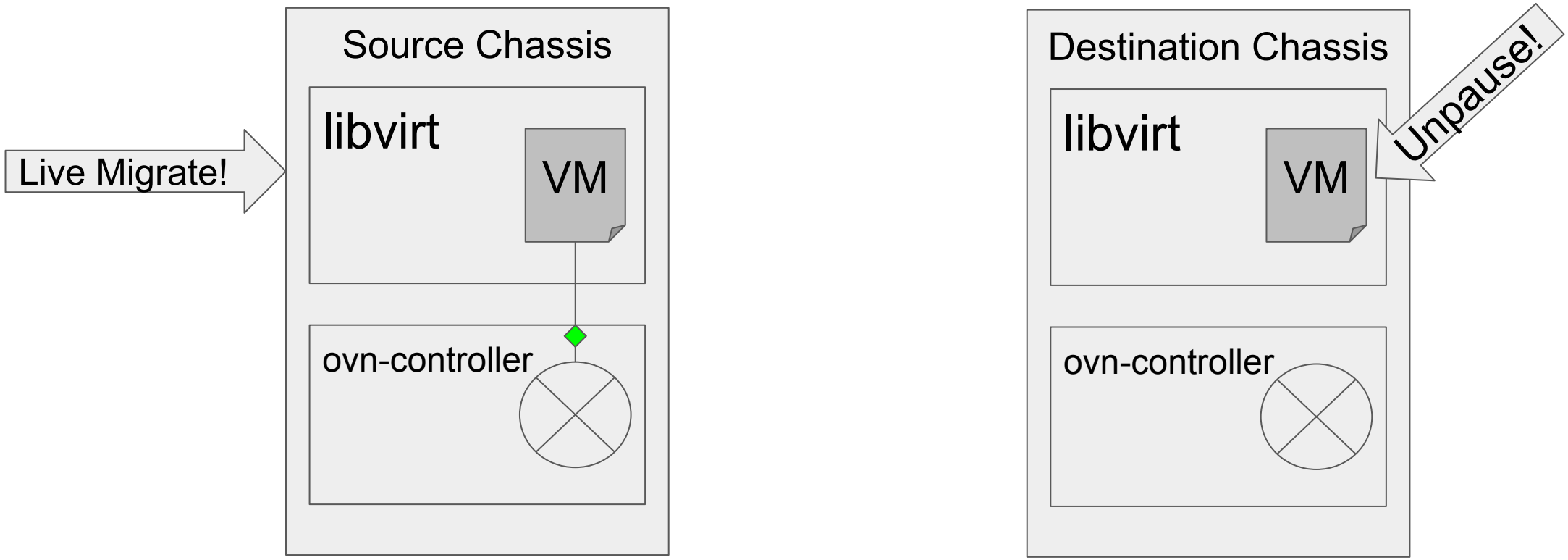
options:requested-chassis=source

Red Hat

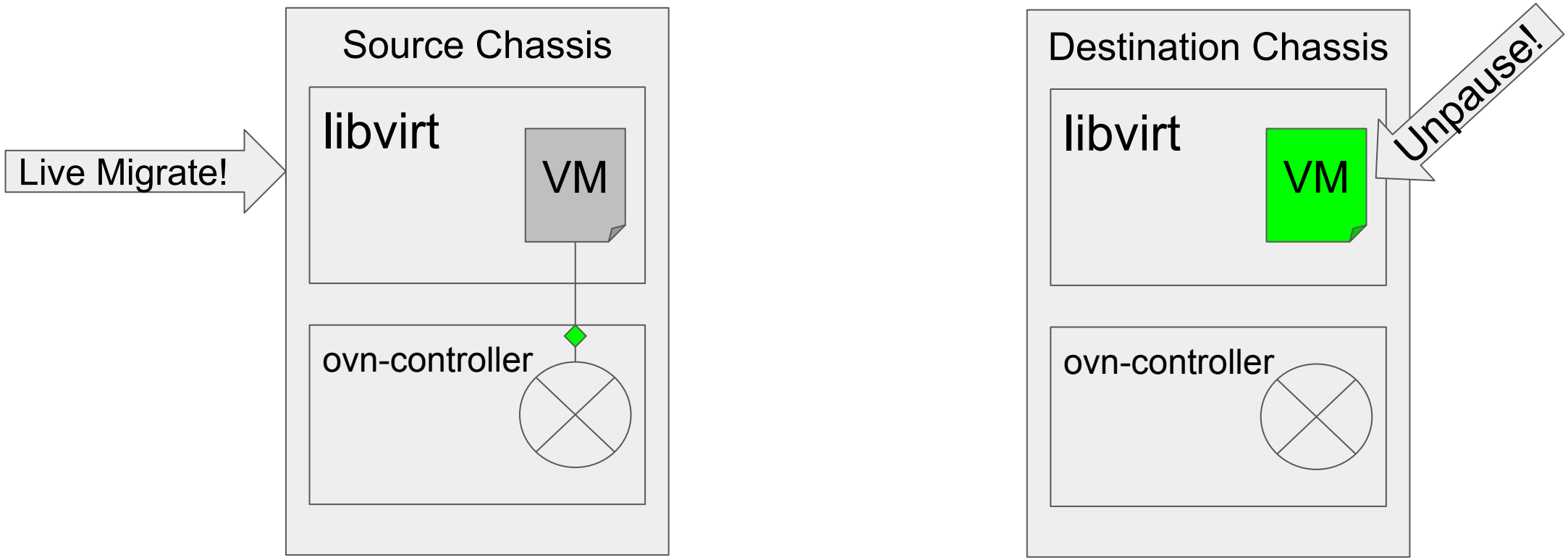


options:requested-chassis=source

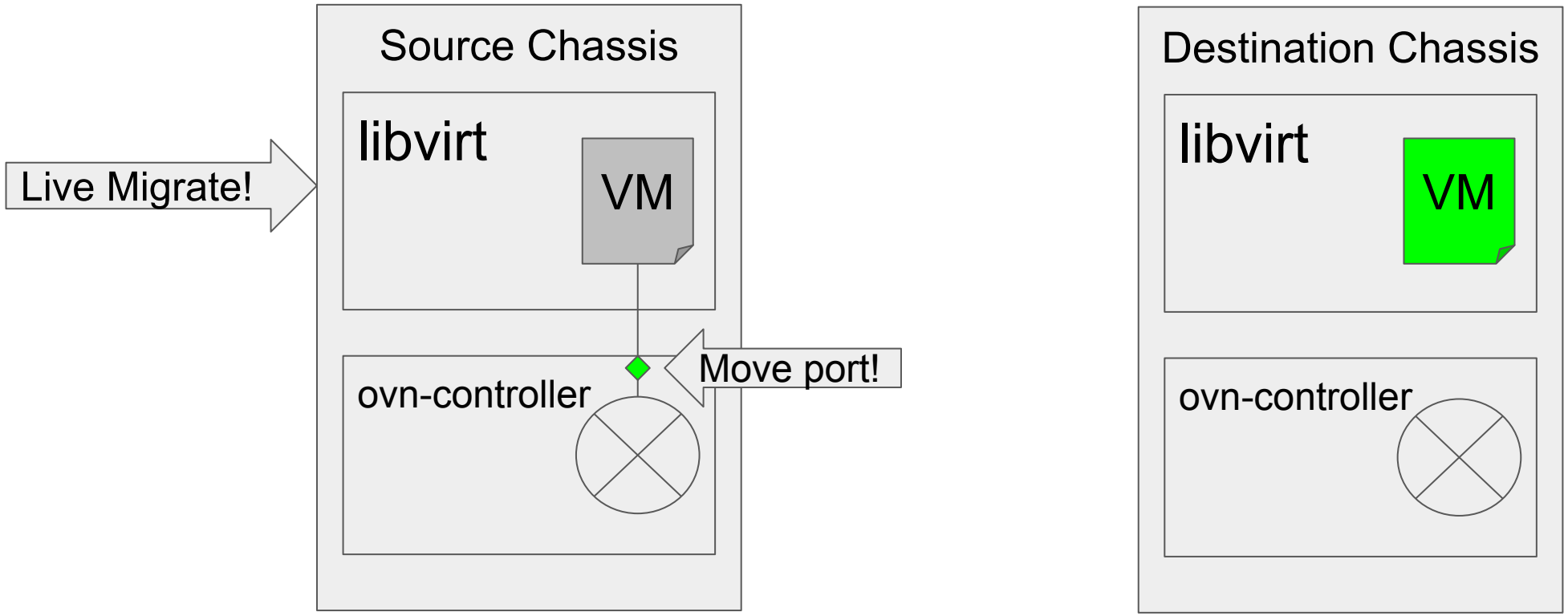
Red Hat



options:requested-chassis=source

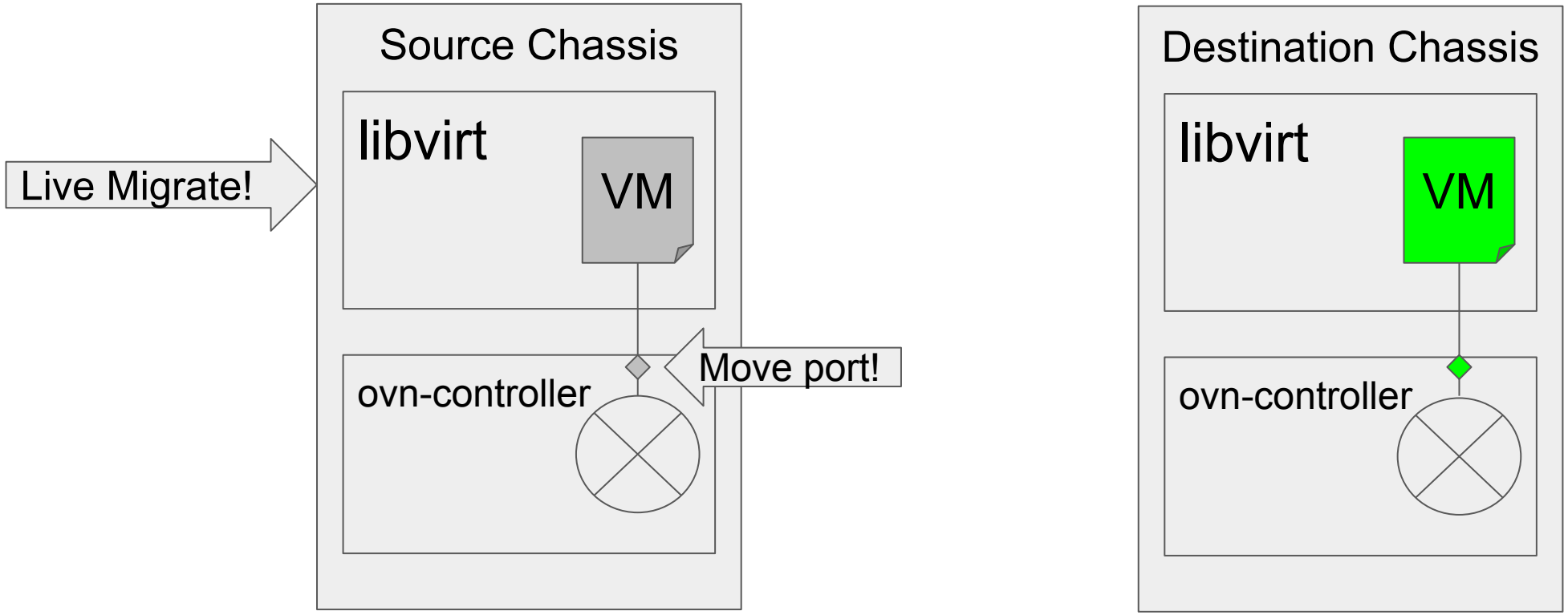


options:requested-chassis=source

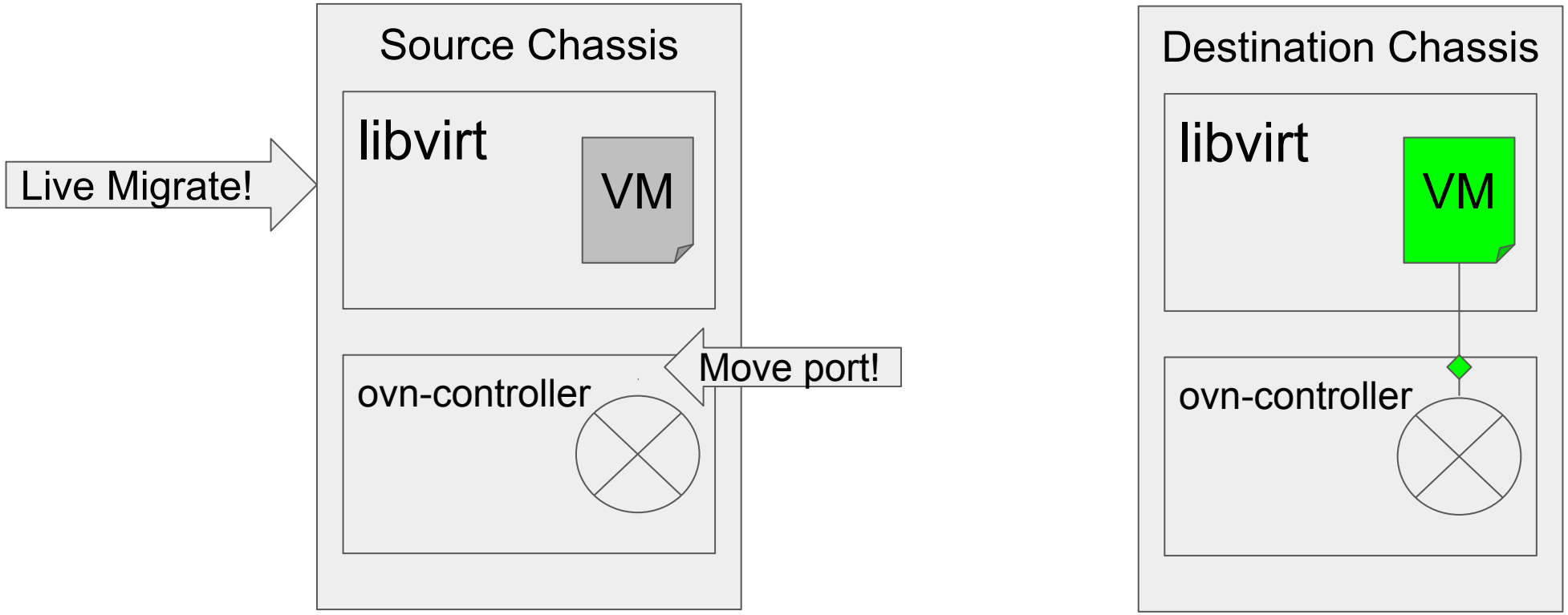


options:requested-chassis=**dst**

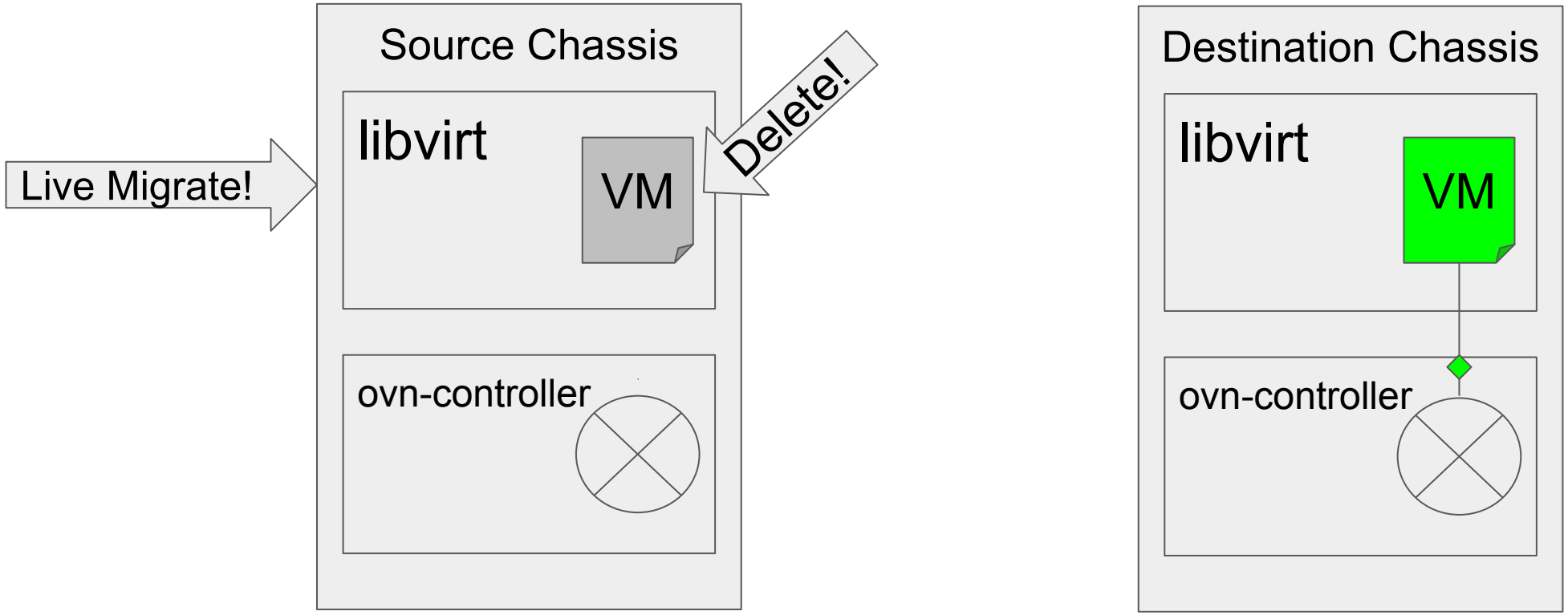
Red Hat



options:requested-chassis=**dst**

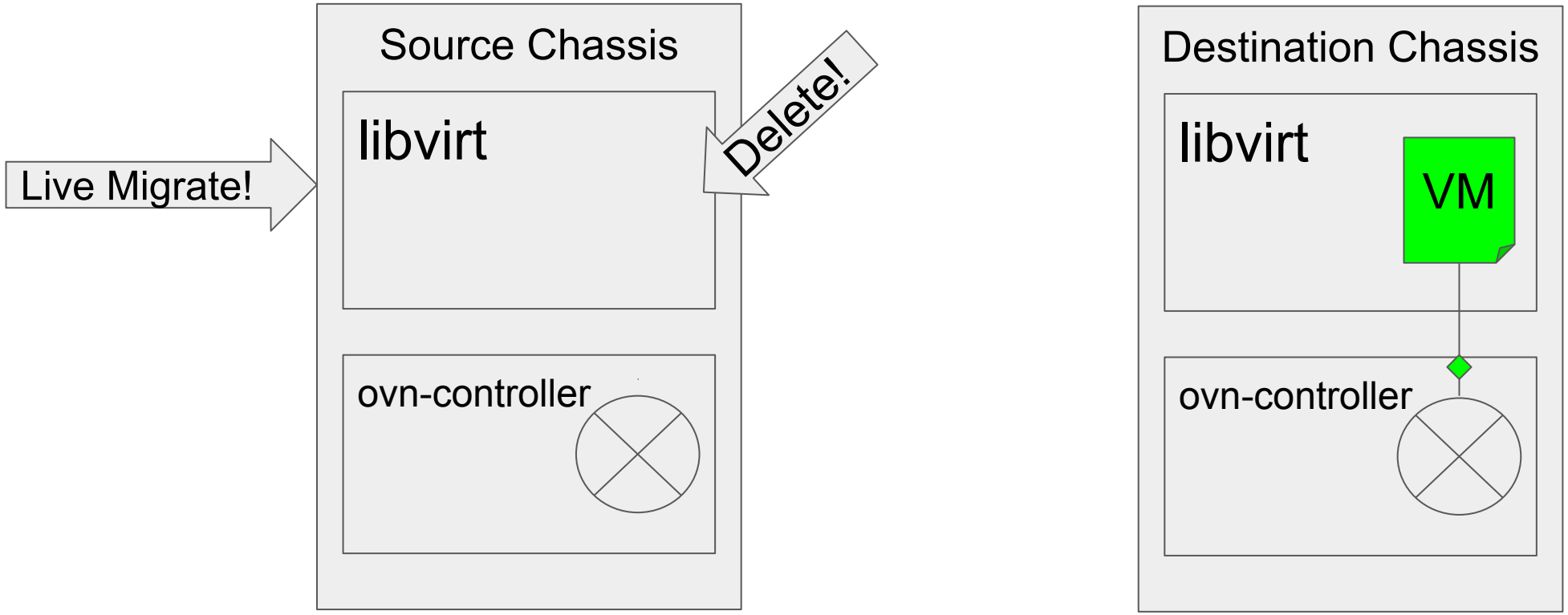


options:requested-chassis=**dst**



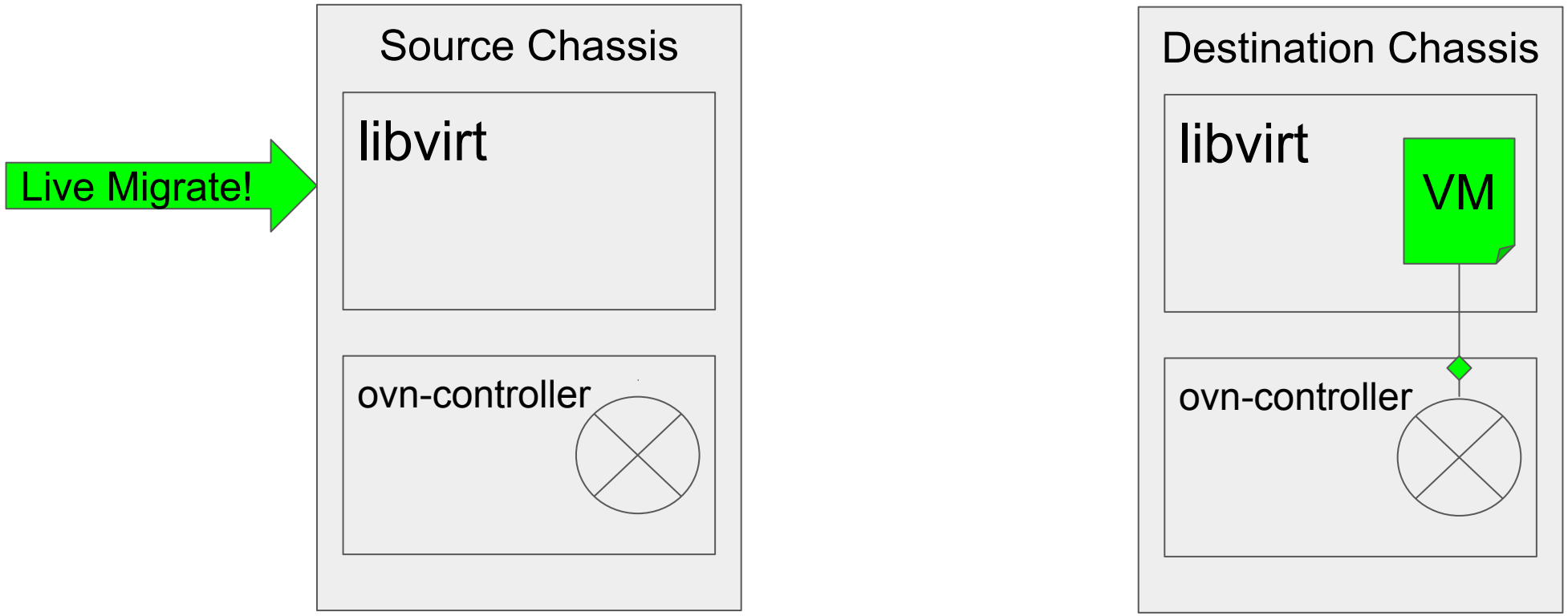
options:requested-chassis=**dst**

Red Hat



options:requested-chassis=**dst**

Red Hat



options:requested-chassis=**dst**

Problem?

Problem?

It takes too long.

Problem?

2-3 seconds too long.

(Or much longer, if any services are down or slow.)

Problem?

```
$ ping -i 0.01 10.0.0.208
…
1607 packets transmitted, 1394
received, 13.2545% packet loss,
time 17975ms
```

Sources of disruption

- From pause to “unpause” (**<0.01 ms**)
- From “unpause” to port moved (**2+ seconds**)

2 seconds, sometimes longer.

2 seconds, sometimes longer.

- Maybe CMS is too slow...

2 seconds, sometimes longer.

- Maybe CMS is too slow...
- Maybe northd is too slow...

2 seconds, sometimes longer.

- Maybe CMS is too slow...
- Maybe northd is too slow...
- Maybe ovn-controller is too slow...

2 seconds, sometimes longer.

- Maybe CMS is too slow...
- Maybe northd is too slow...
- Maybe ovn-controller is too slow...

while VM apps wait for connectivity to restore

Solution?

Solution?

- Pre-configure destination port
- Don’t wait on CMS to enable new port

How?

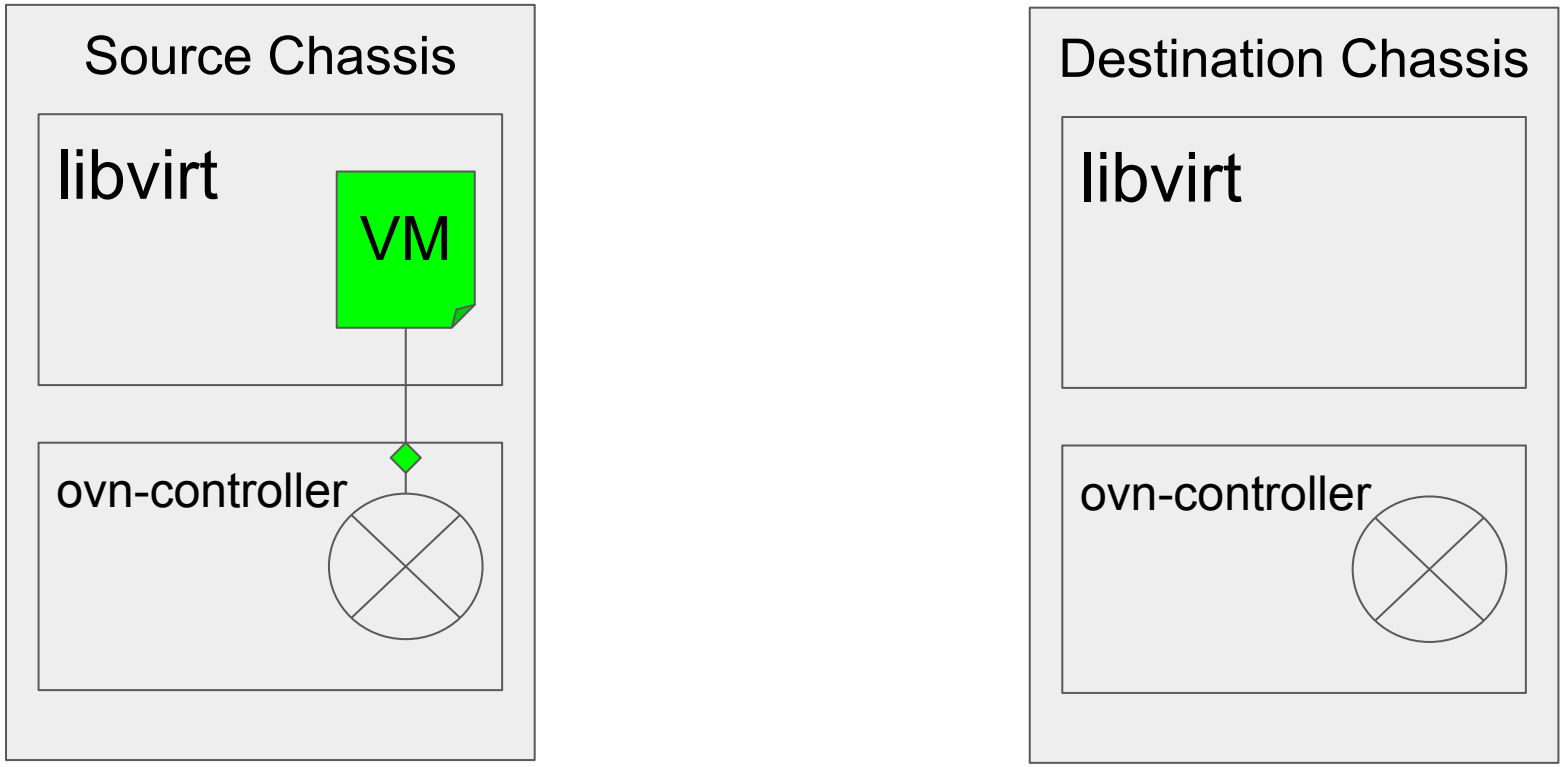
"Multi-chassis" port bindings.

options:requested-chassis is now a list

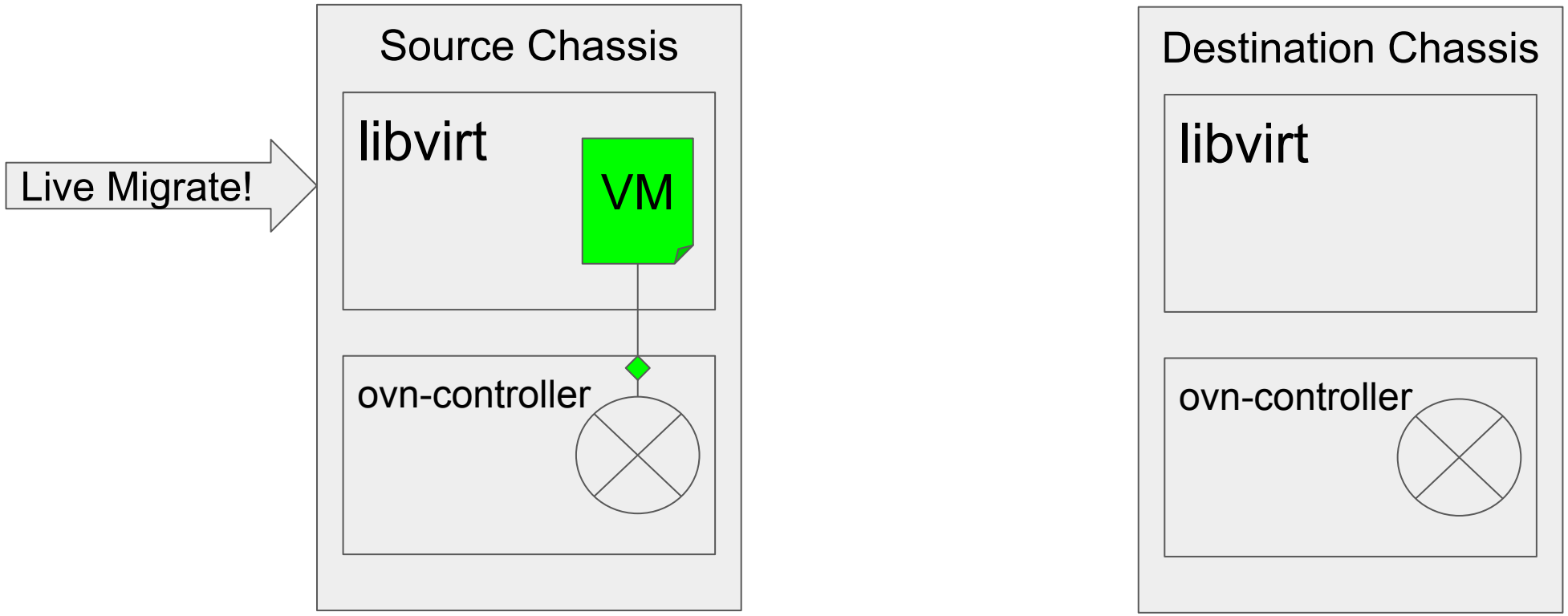
- destination chassis can pre-configure port binding during live migration

deliver packets to both chassis

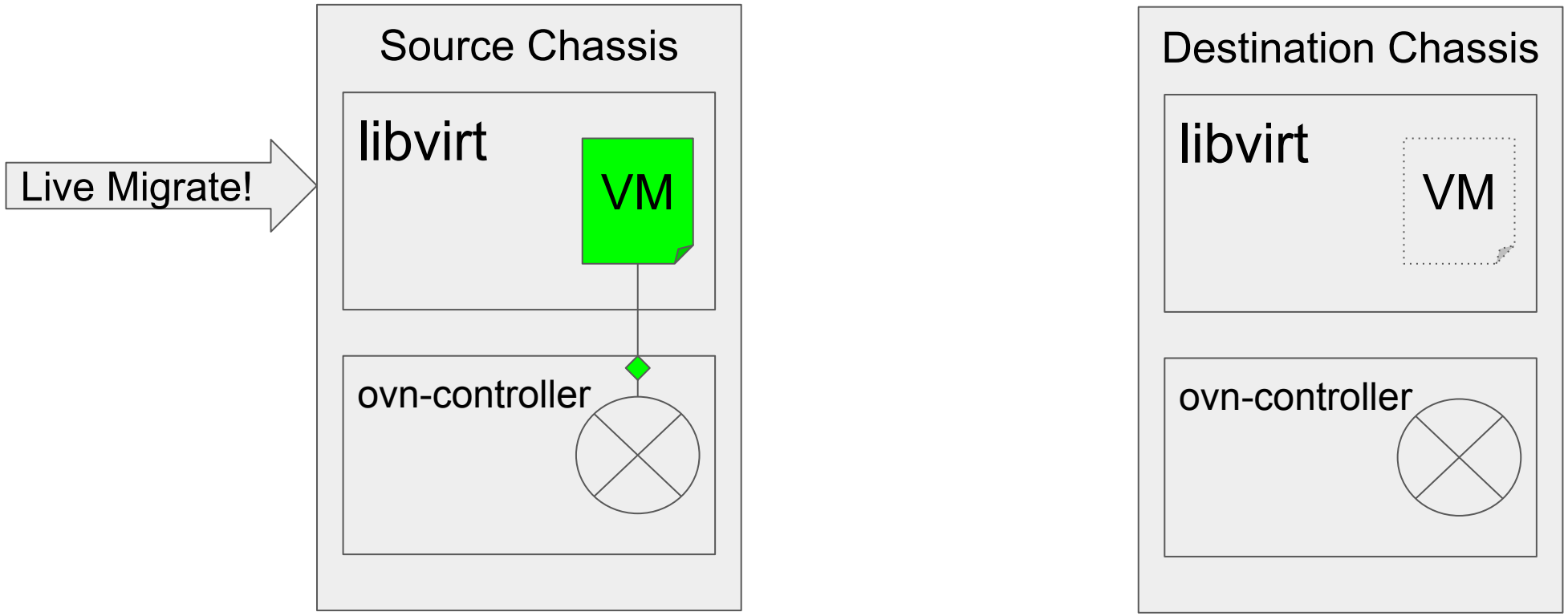
- packets are cloned to tunnels
- tunneling is enforced



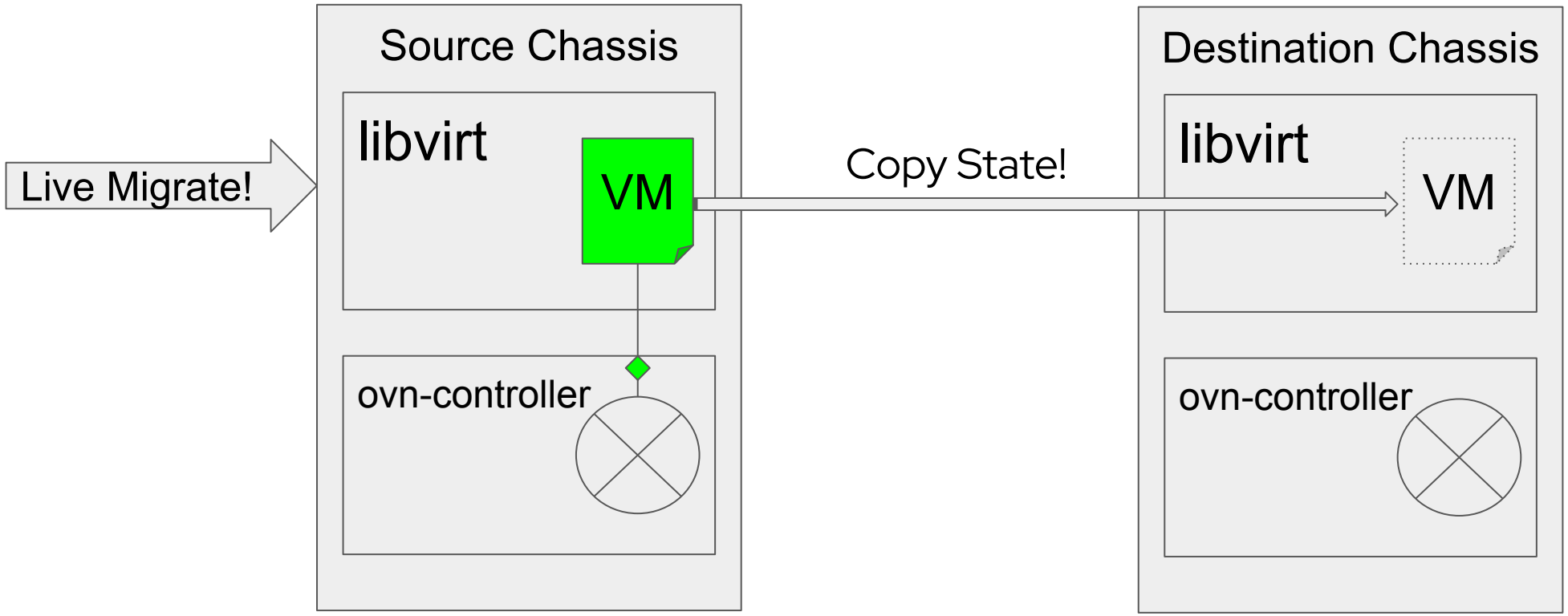
options:requested-chassis=source



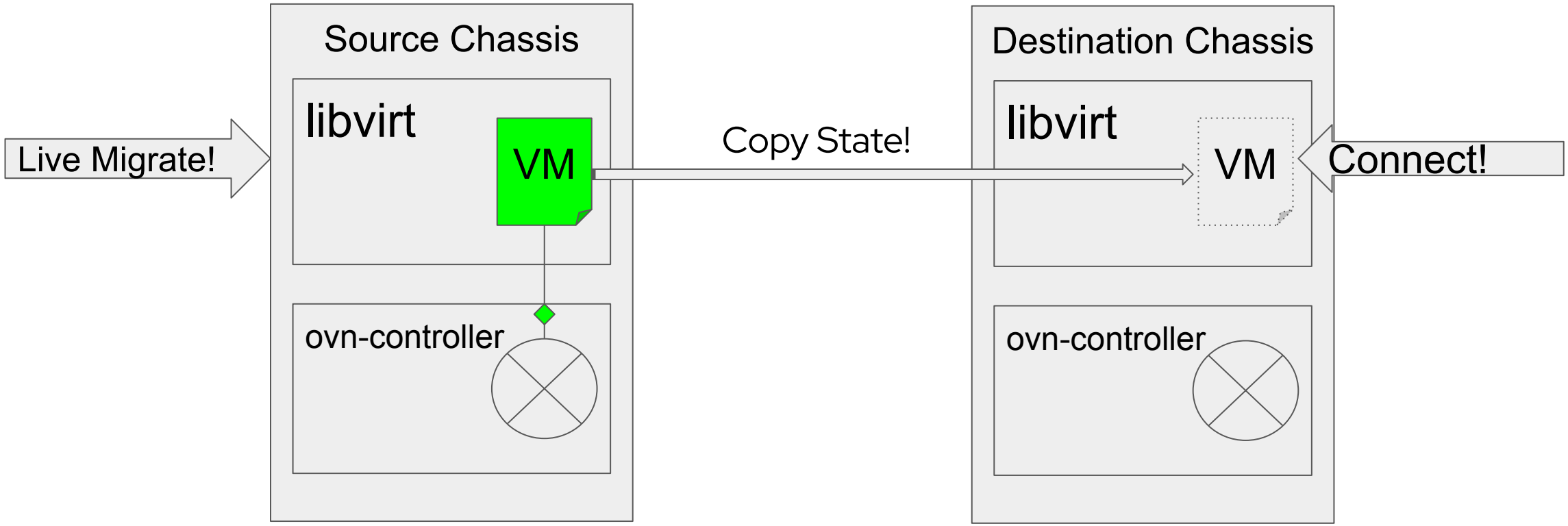
options:requested-chassis=source



options:requested-chassis=source

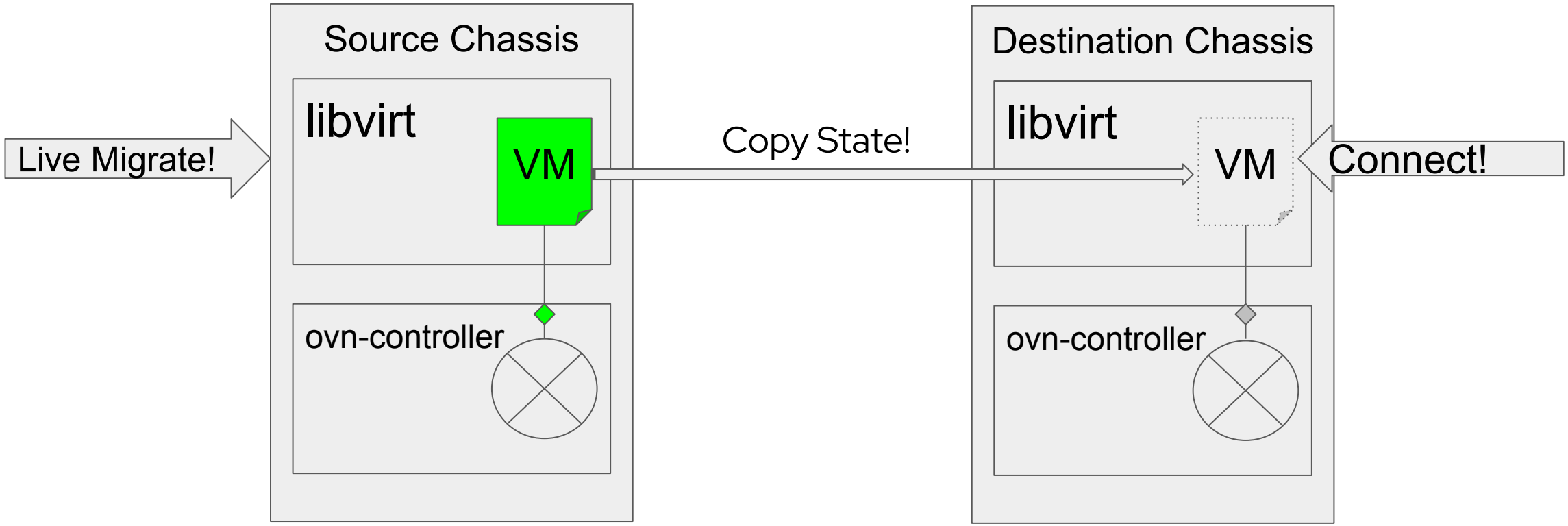


options:requested-chassis=source



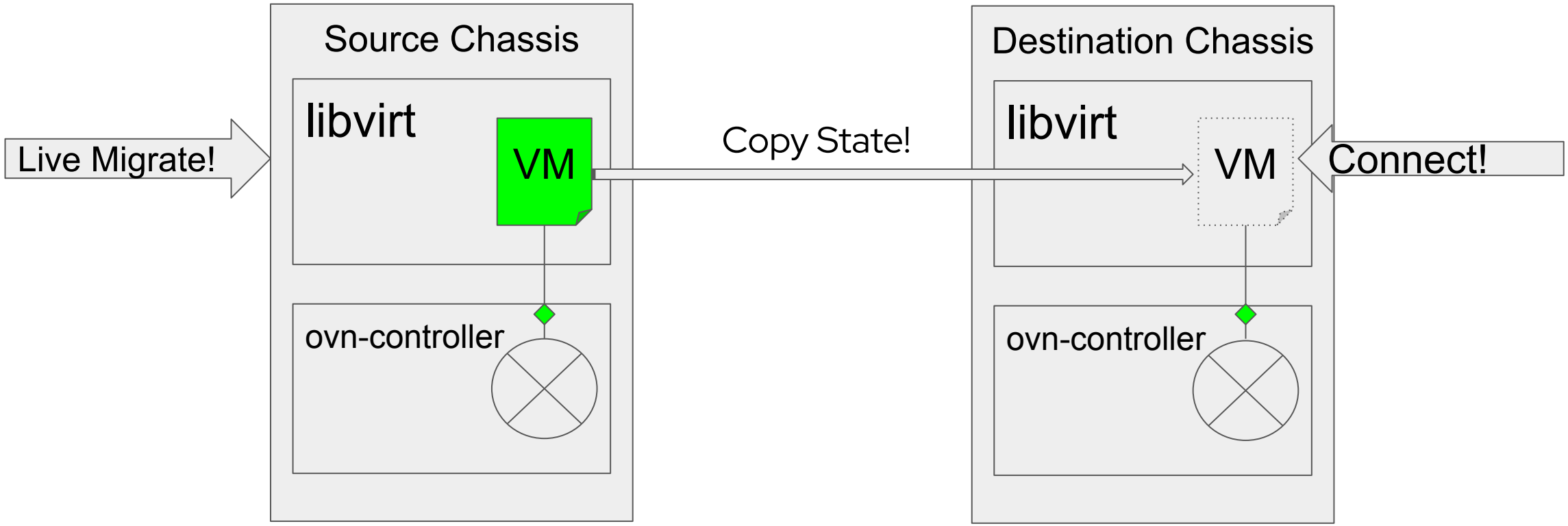
options:requested-chassis=source,**dst**

Red Hat



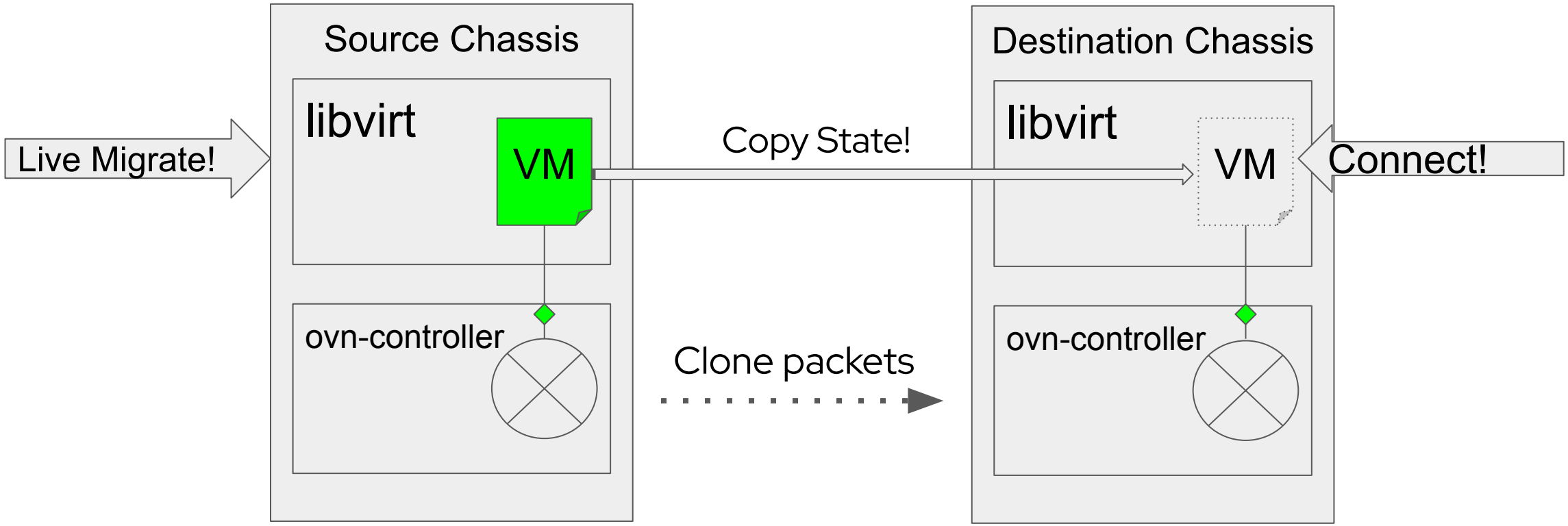
options:requested-chassis=source,**dst**

Red Hat



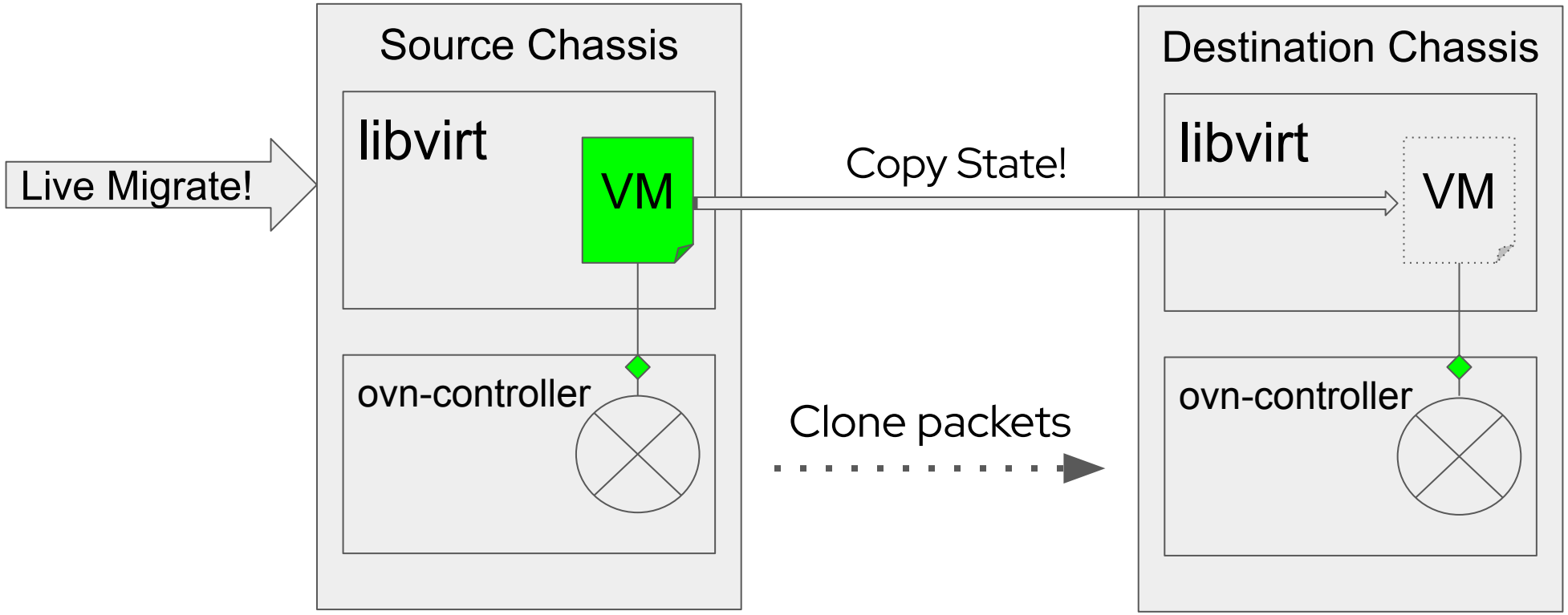
options:requested-chassis=source,**dst**

Red Hat



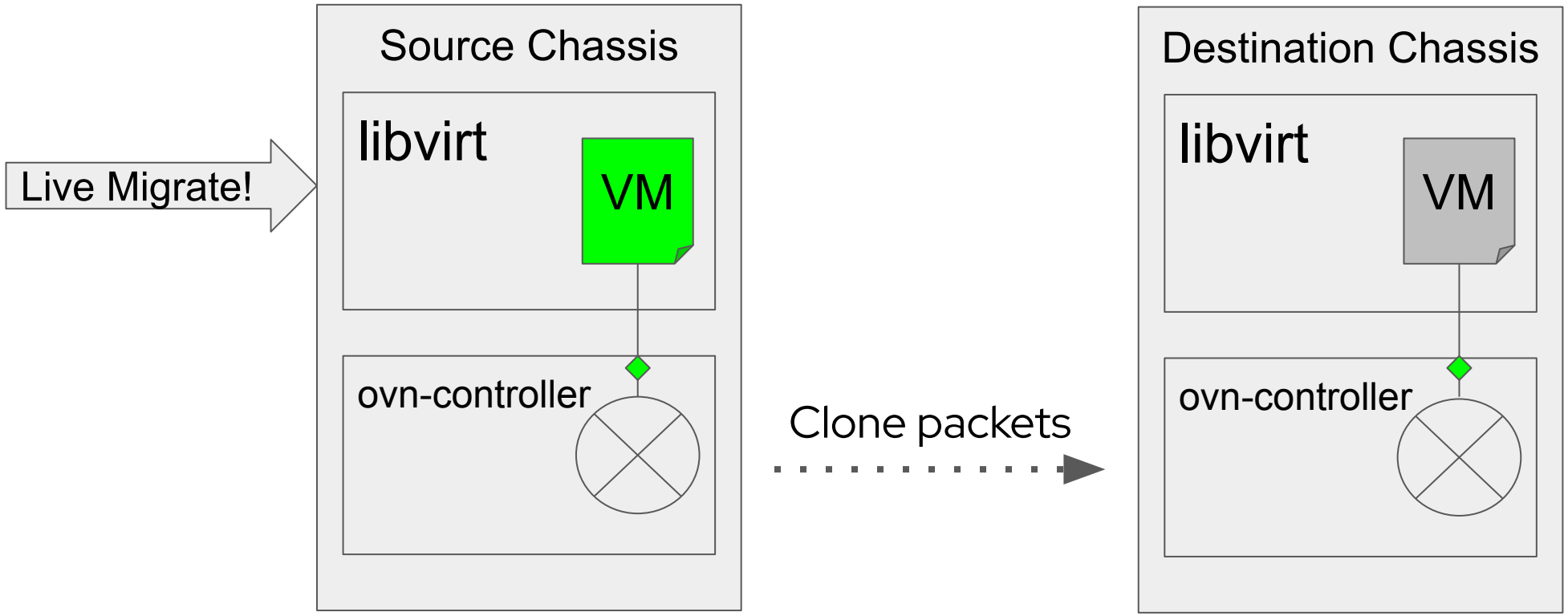
options:requested-chassis=source,**dst**

Red Hat



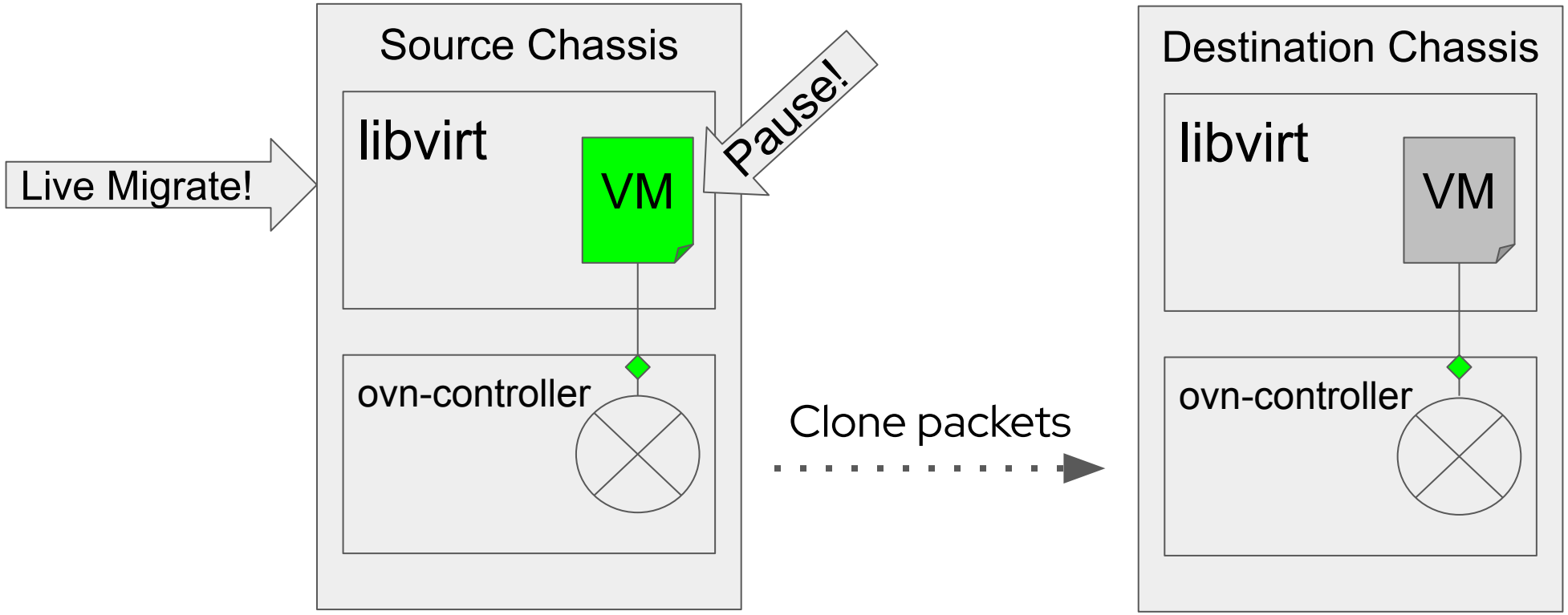
options:requested-chassis=source,**dst**

Red Hat



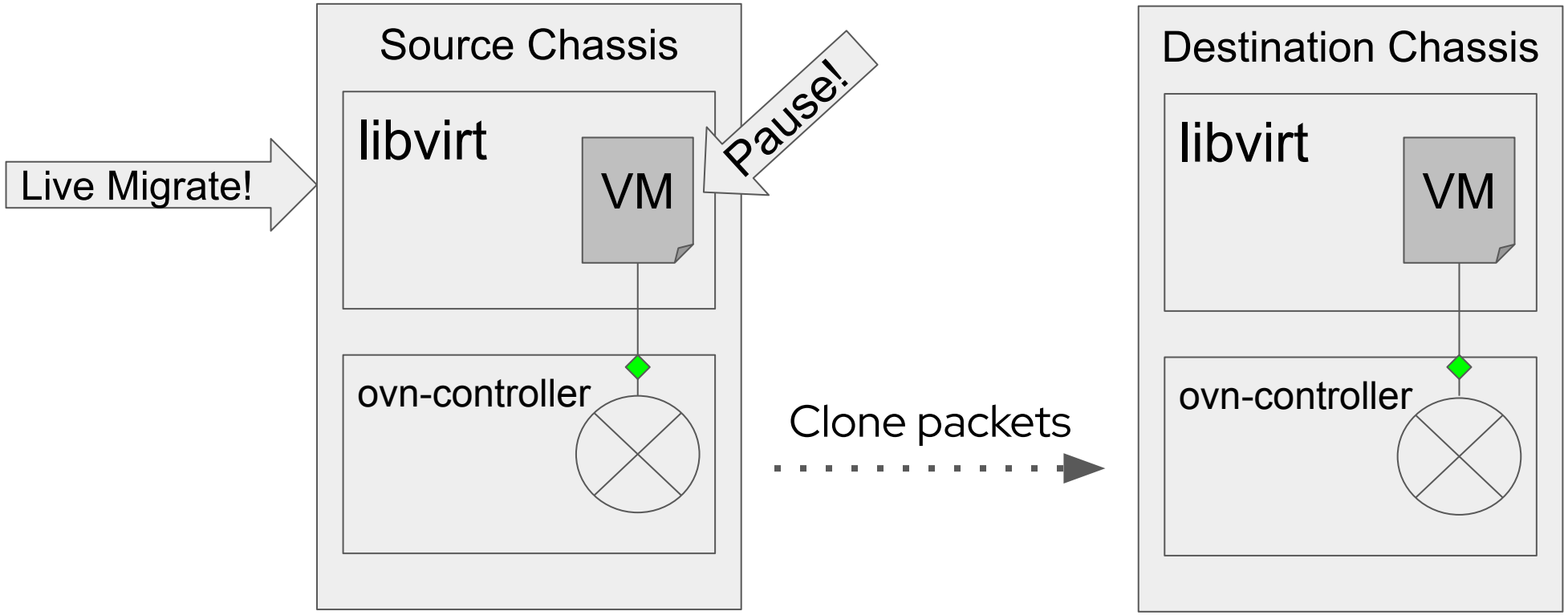
options:requested-chassis=source,**dst**

Red Hat

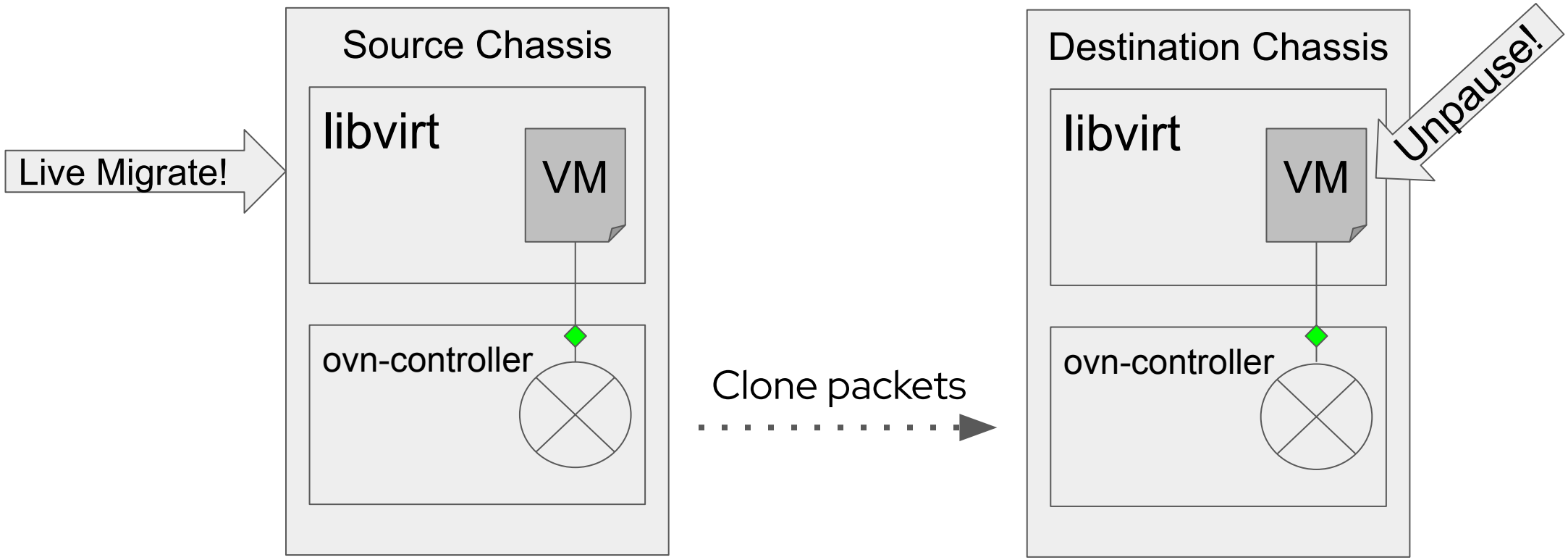


options:requested-chassis=source,**dst**

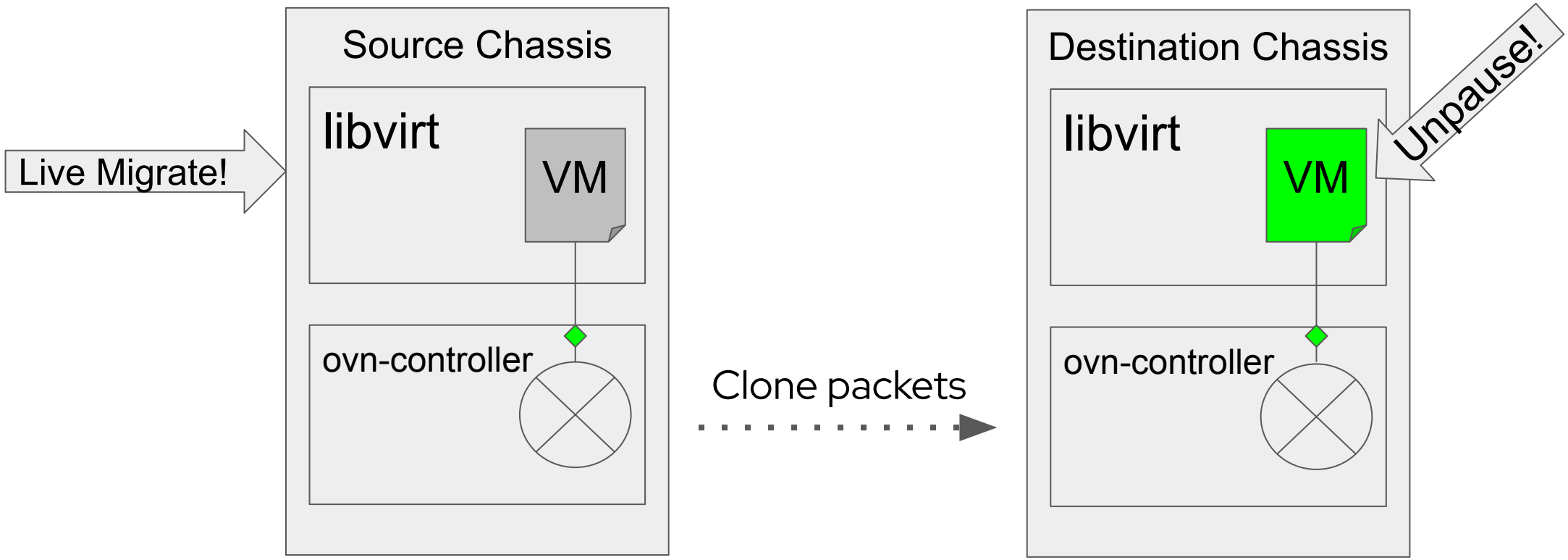
Red Hat



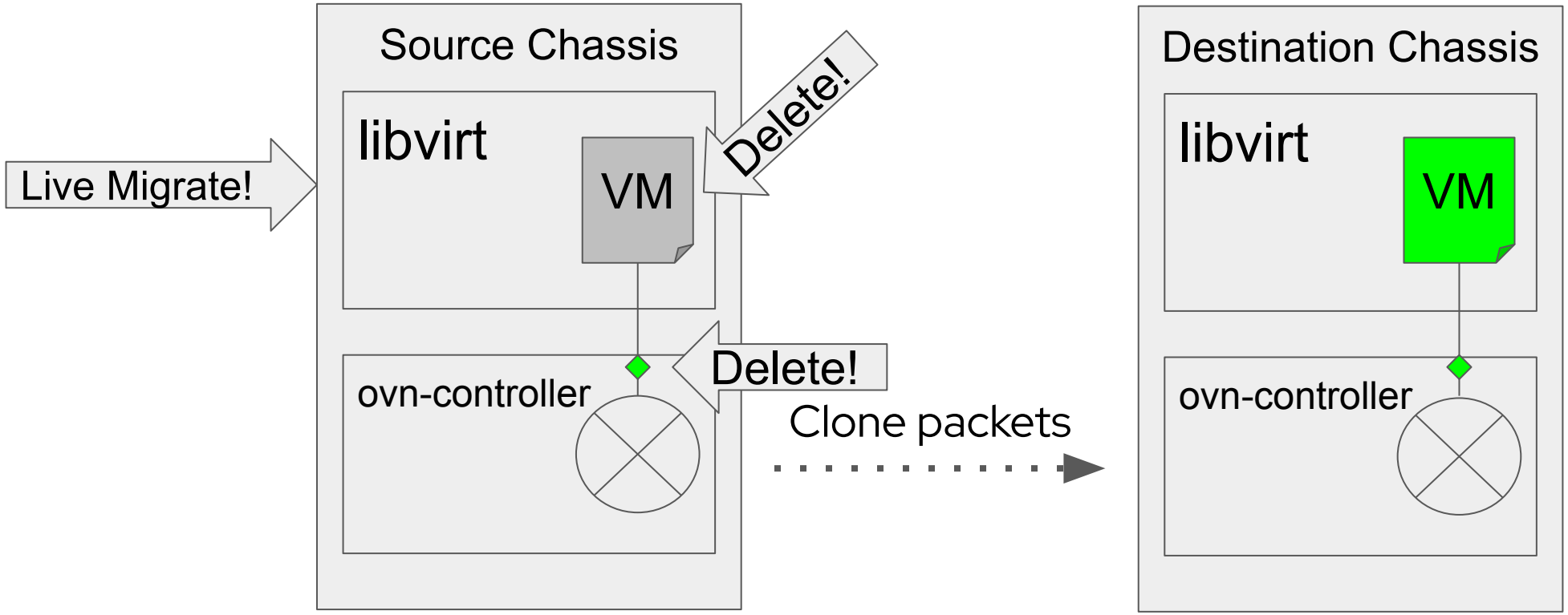
options:requested-chassis=source,**dst**



options:requested-chassis=source,**dst**

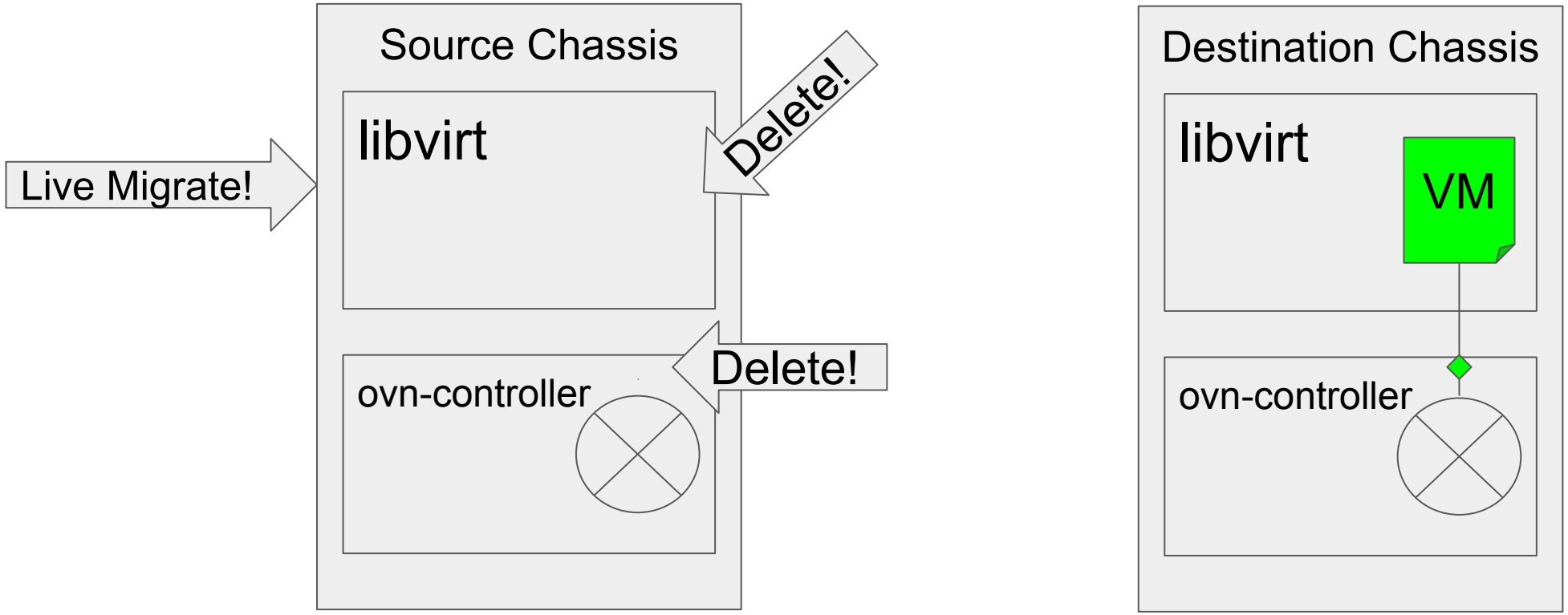


options:requested-chassis=source,**dst**



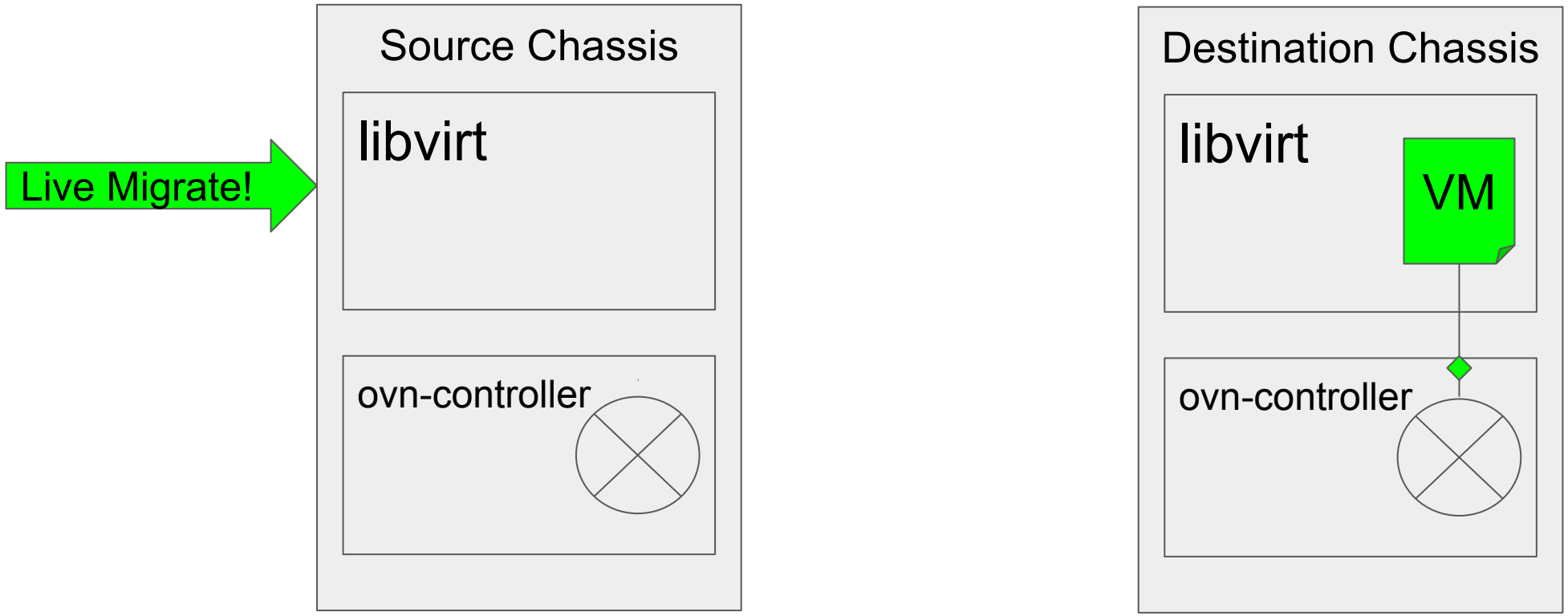
options:requested-chassis=**dst**

Red Hat



options:requested-chassis=**dst**

Red Hat



options:requested-chassis=**dst**

Sources of disruption

- From pause to “unpause” (**<0.01 ms**)
- ~~From “unpause” to port moved (**2+ seconds**)~~

Let’s repeat the test…

It works!

```
1153 packets transmitted, 1153 received, +94 duplicates, 0% packet loss, time 24239ms
```

It works! (with a caveat)

```
1153 packets transmitted, 1153
received, +94 duplicates, 0%
packet loss, time 24239ms
```

Both VMs captured the same packets. We need to wait.

Libvirt will tell us when it’s ready, with RARP.

options:activation-strategy=rarp

- drop all traffic

options:activation-strategy=rarp

- drop all traffic
- ...except ingress RARP

options:activation-strategy=rarp

- drop all traffic
- ...except ingress RARP
- ...handled by controller()

options:activation-strategy=rarp

- drop all traffic
- ...except ingress RARP
- ...handled by controller()
- ...which removes drop rules

options:activation-strategy=rarp

- drop all traffic
- ...except ingress RARP
- ...handled by controller()
- ...which removes drop rules
- ...activating the new port location

Let’s try again…

Voila!

```
1235 packets transmitted,
1231 received, 0.323887%
packet loss, time 21252ms
```

4 packets lost

~

4 ms

4 ms << 2s+

… all without CMS action involved

Thank you. Questions?