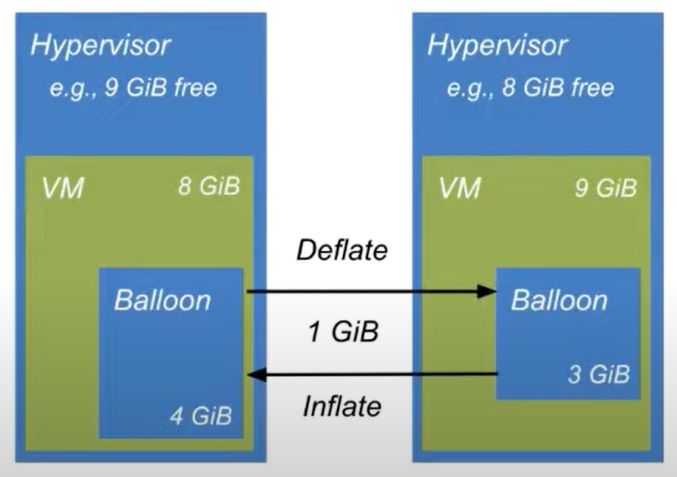
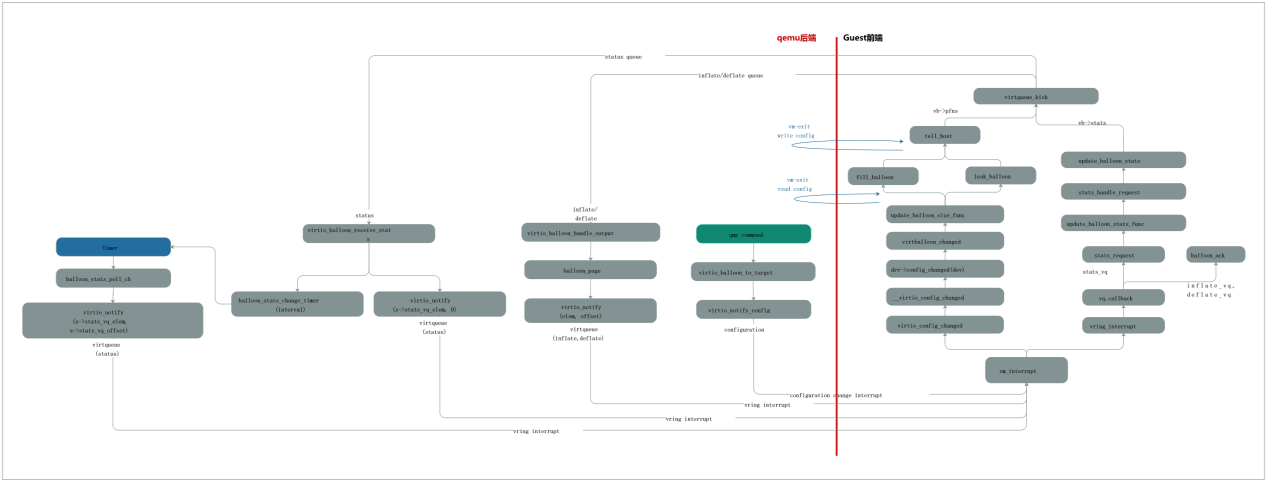
# Auto-Ballooning设计与实现

## Memory Ballooning

Linux virtio-balloon设备允许KVM虚拟机降低内存大小(通过放弃内存给host主机)以及增加内存大小(从host主机获取内存)。这个balloon功能主要是为了支持KVM主机内存的超卖(over-committing memory)。也就是host可以运行所有VM总内存可以大于物理主机实际内存。balloon设备对于内存超卖(memory over-commitment)非常重要，因为它可以在需要时降低guest的内存。如果guest需要运行一个消耗更多内存的应用程序，它有可以增加内存。除了内存超卖，内存ballooning也可以在虚拟机热迁移中收益，较少的内存可以加速热迁移速度。



### virtio-balloon的实现



### 存在的问题

1. 内存调整对GuestOS不透明

Virtio-balloon对guest内存进行inflated之后，从guest角度看，总的内存使用量减少，如果原本虚拟机申请了2G内存，inflate 1G内存之后，从guest内部查询，内存使用情况时，总内存变为1G，可用内存，空闲内存均减少了1G。这种实现机制本身是没有问题的，但对于云原生场景下，虚拟机通常提供了动态资源扩展的能力。例如，虚拟机集群提供了云上K8S服务，虚拟机作为worker节点，内存作为K8S关键资源，称为资源调度的关键因素，如果inflated memory之后，那么report到k8s的可用资源必然减少了，从而影响了k8s服务资源调度。

1. 无法感知guest内存负载

Memory ballooning机制的核心是通过降低guest实际使用内存，从而满足内存超卖的需求。当前的ballooning是通过qemu qmp协议，主动从hypervisor出发去inflate/deflate 虚拟机的内存，无法感知guest内存使用情况，不能从guest内存使用情况出发满足内存超卖。

1. 没有自动化管理机制

目前没有方便的自动化机制管理ballooning

## Auto-ballooning设计与实现

### 核心设计思想

设置内存inflate高水位线，低水位线，内存调整粒度。高水位线用于判断，当前guest free pages 数量，如果高于高水位线，则意味着guest的内存使用负载较轻，可以释放free pages给宿主机。低水位线，则意味着guest 内存压力变大，需要还给guest 真实物理内存。内存调整粒度为每次inflate 内存的尺寸，降低inflate期间对guest系统负载的影响。

支持inflated 内存从guest 用户态查看时透明的，即从用户态对内存气球工作情况无感知。

支持内存热插拔场景下，水位线的自动调整；

提供管理高水位线，低水位线，内存调整粒度能力。

兼容现有的手动管理内存气球能力。

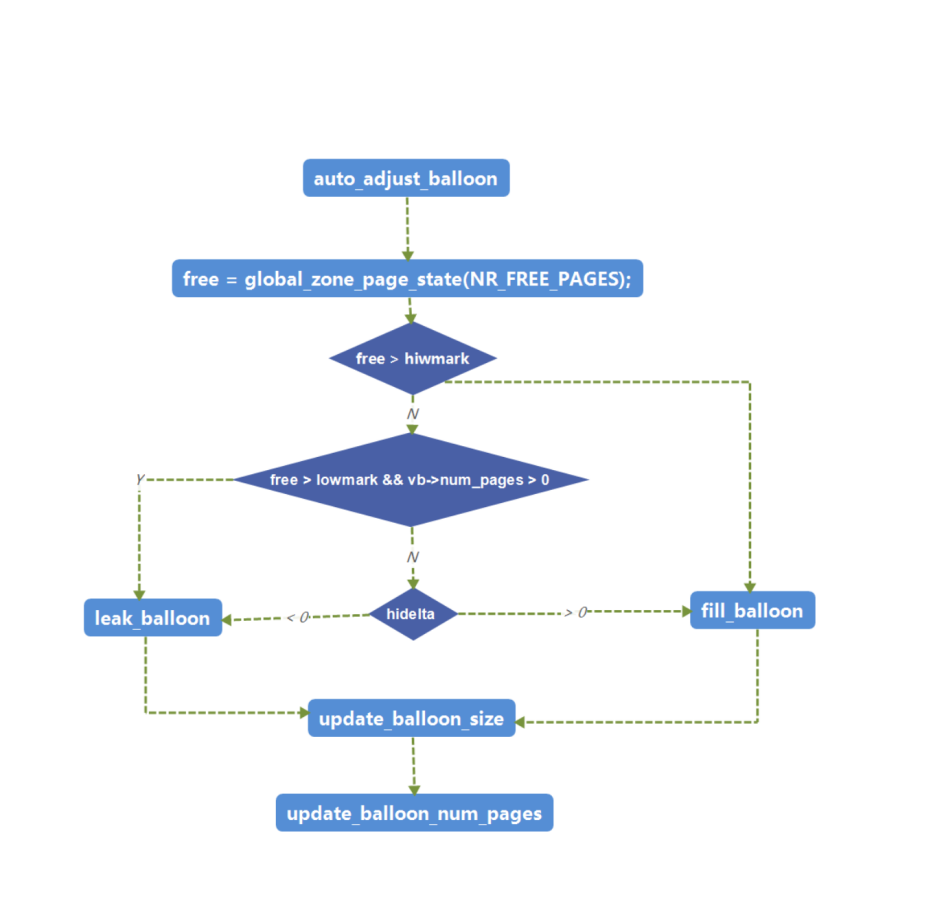
### 功能实现

## Guest kernel 部分

增加VIRTIO\_BALLOON\_F\_AUTO\_BALLOON feature，用于host与guest协商是否支持自动气球管理能力。

增加virtio-balloon pci配置空间参数用于hiwmark lowmark granularity控制气球自动释放内存比例。

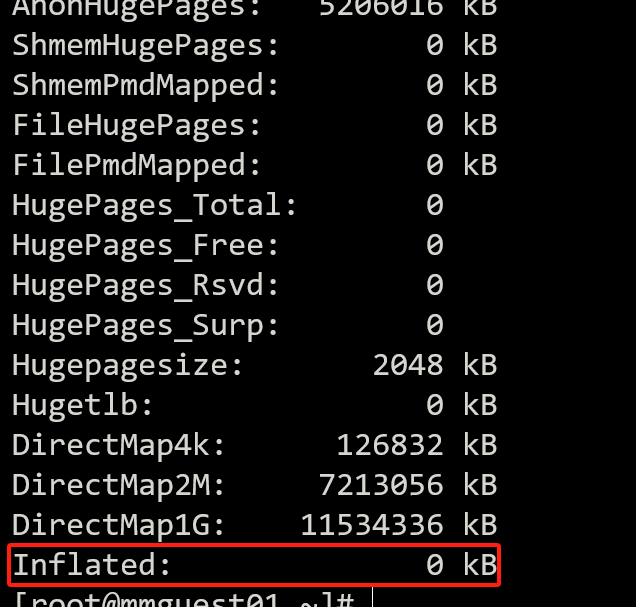
创建自动气球工作线程，工作线程周期性检测free pages 情况，根据水位线对内存进行inflated/deflated操作，处理逻辑如图所示。



支持内存热插拔，通过注册内存热插拔notifier，更新内存水位，达到自动调整的目的。

支持内存shrinker，可通过shrinker机制，及时释放inflated内存，防止内存压力突然增大时，没有及时回收内存。

内存meminfo增加inflated内存显示，MemTotal/MemFree/MemAvailable 默认为用户无感知。



提供 /proc/sys/dev/virtio-balloon/目录下，对自动气球管理调增

show\_adjust\_mem: 是否对用户透明，默认透明

### Host 部分

Qemu

命令支持指定开启自动气球管理

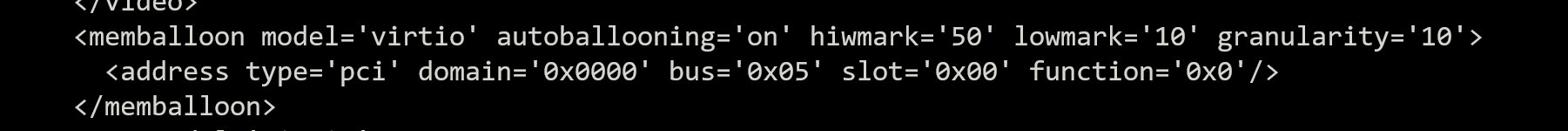
-device virtio-balloon,auto-ballooning=on,hiwmark=50,lowmark=30,granularity=10,autointerval=1000

支持

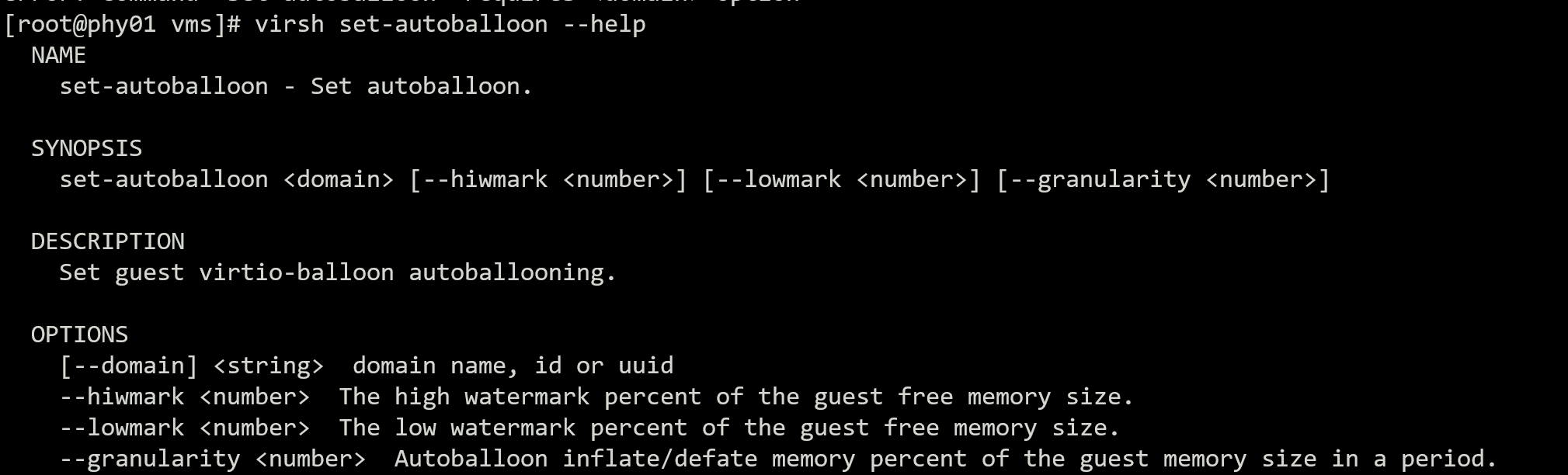
支持通过qmp/hmp方式，修改hiwmark, lowmark, granularity, autointerval, 并生成AUTOBALLOON\_CHANGE event。

Libvirt:

通过xml定义开启自动气球能力



通过virsh命令修改hiwmark, lowmark, granularity



根据AUTOBALLOON\_CHANGE event，更新domain的hiwmark, lowmark, granularity,autointerval

## 功能验证

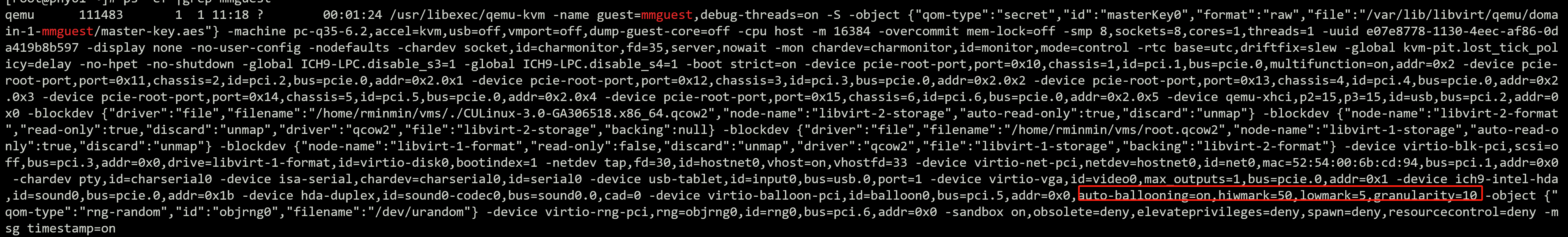
1. 创建开启auto-ballooning的虚拟机

<memballoon model='virtio' autoballooning='on' hiwmark='50' lowmark='10' granularity='10' autointerval='1000'>

<address type='pci' domain='0x0000' bus='0x05' slot='0x00' function='0x0'/>

</memballoon>

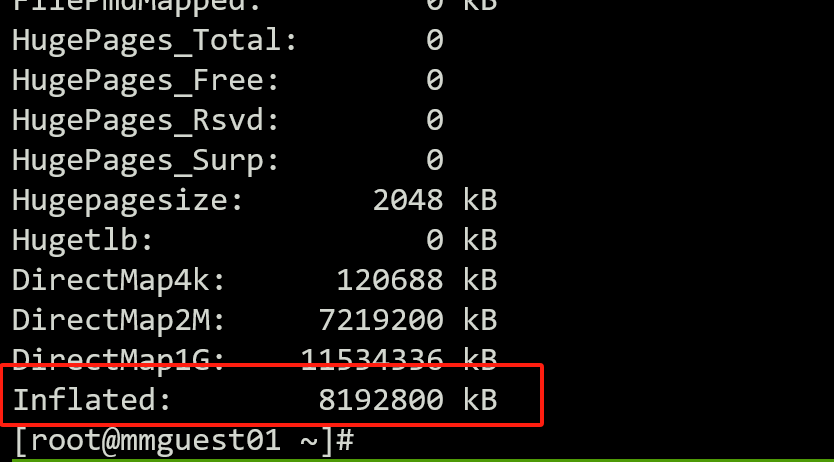
1. 确认auto-ballooning 开启



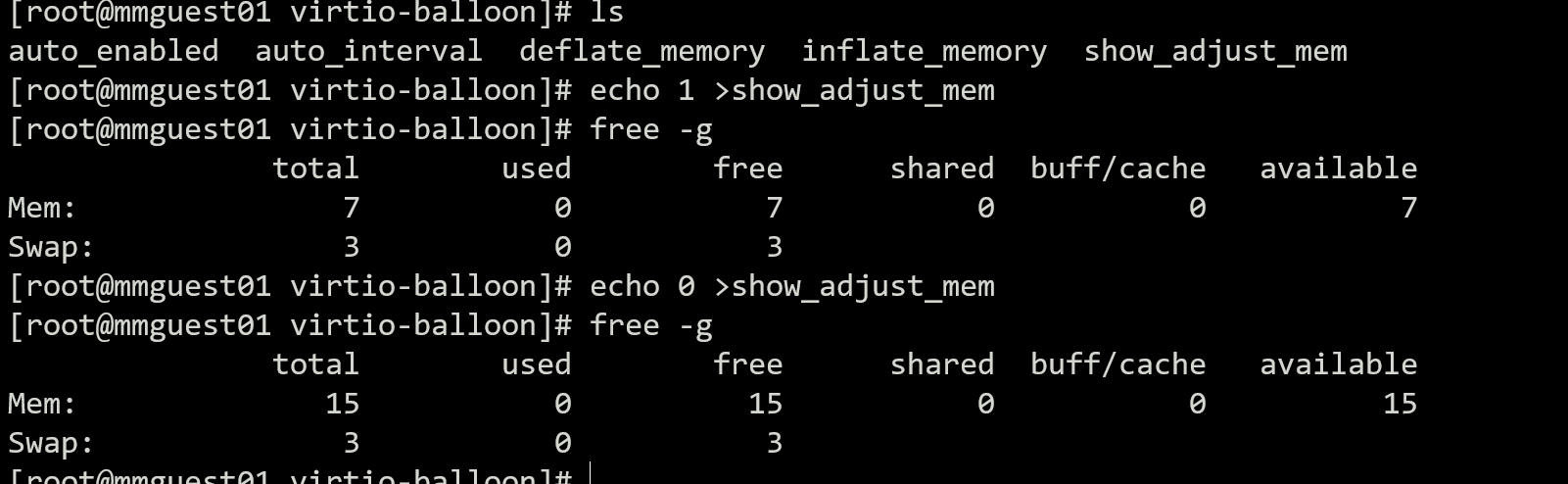
1. 登录guest查看内存使用情况

默认情况下，inflated memory透明

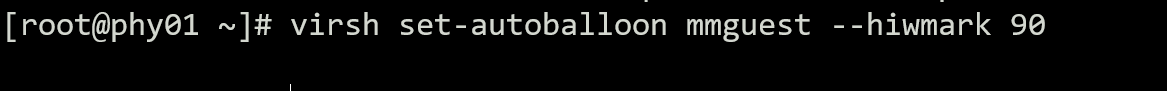


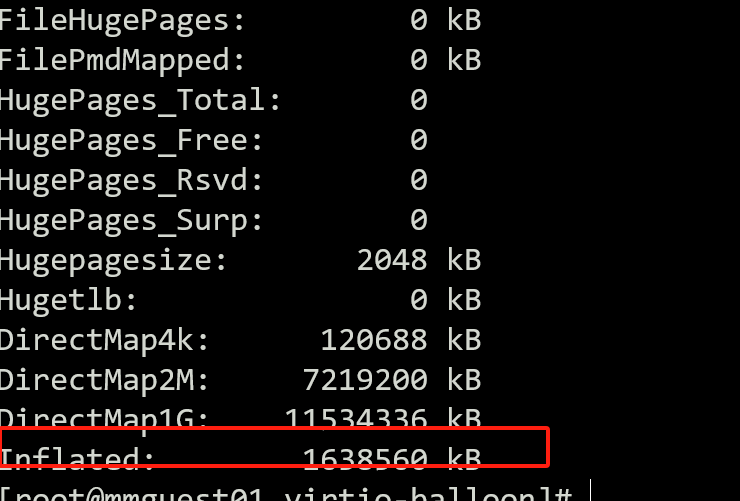


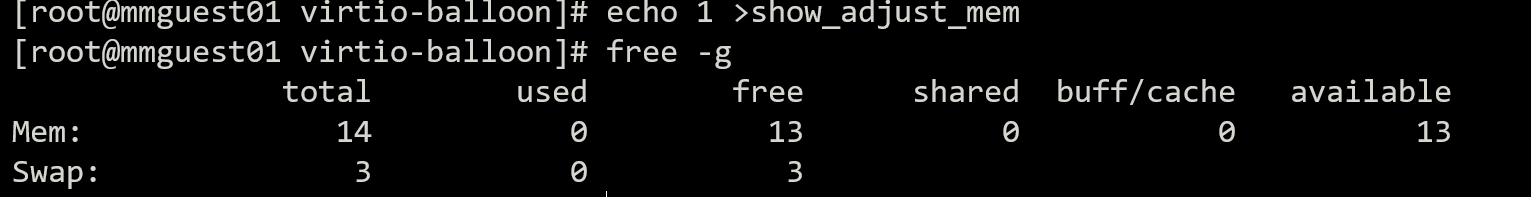
1. 用户态接口调整inflated 内存是否可见

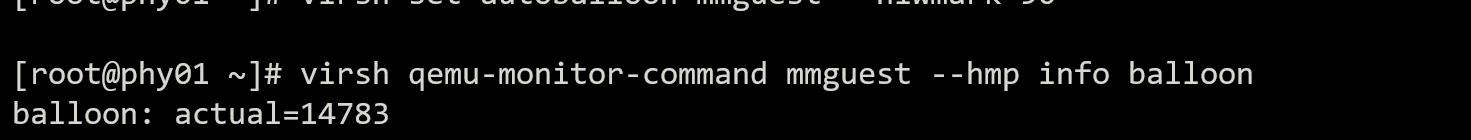


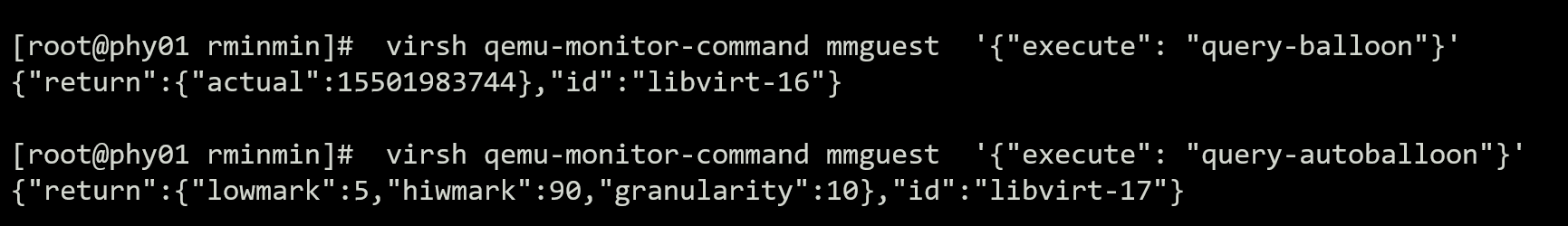
1. 调整高水位线，降低inflated 内存



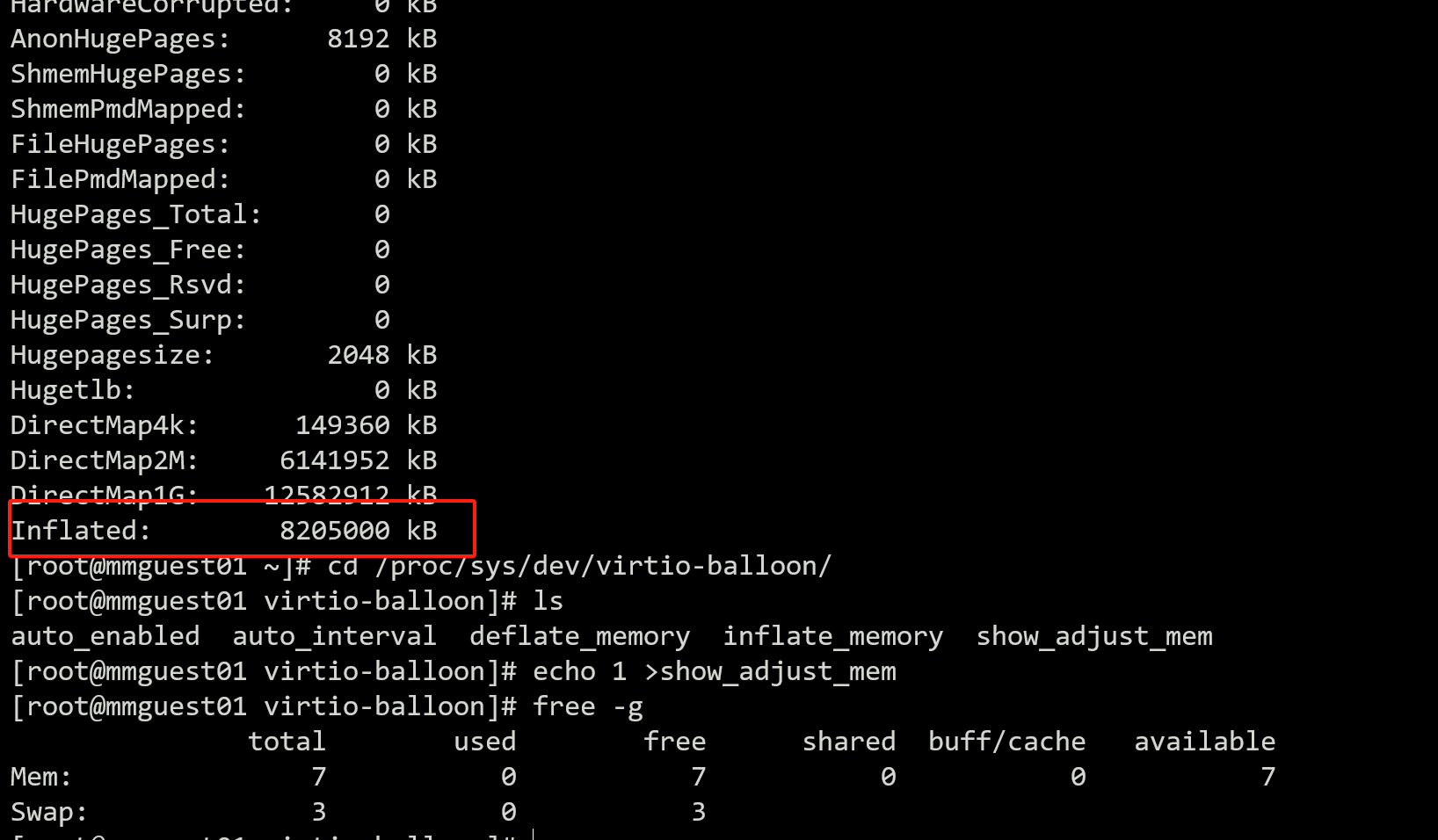


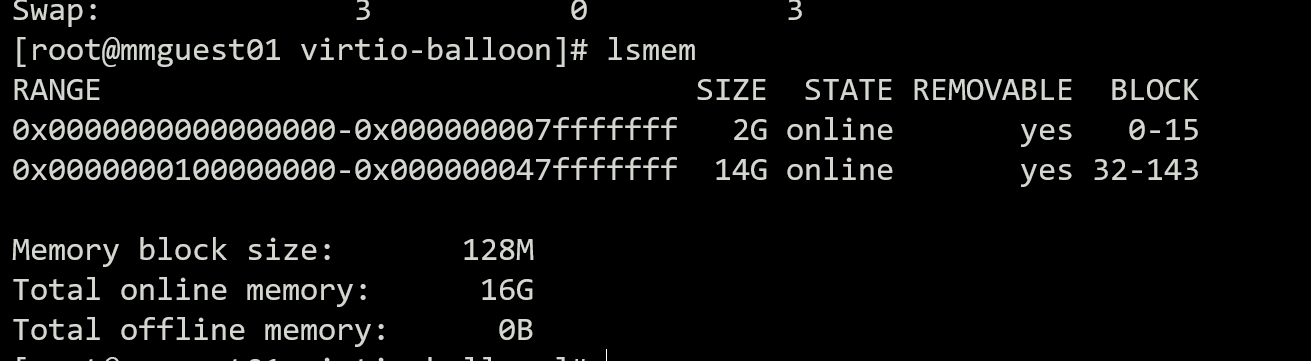






1. 内存热插拔





添加8G内存

virsh attach-device --live mmguest add-mem.xml

<maxMemory slots='2' unit='KiB'>33554432</maxMemory>

<memory model='dimm'>

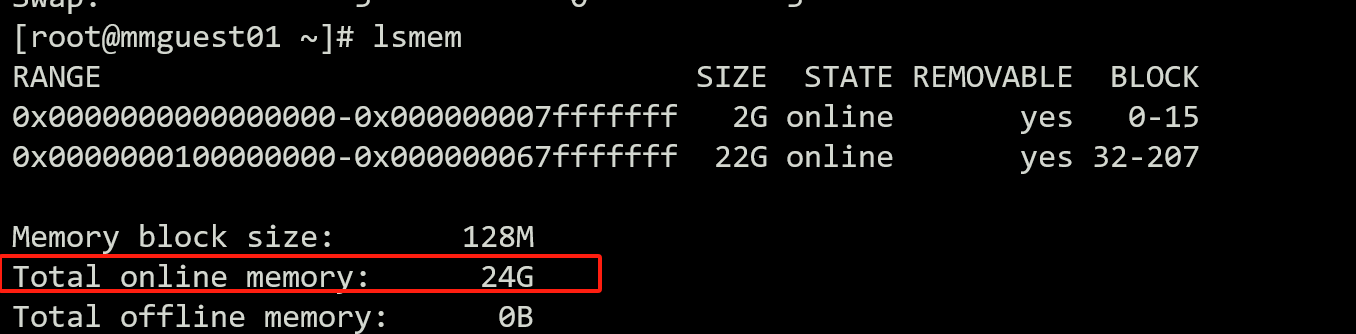
<target>

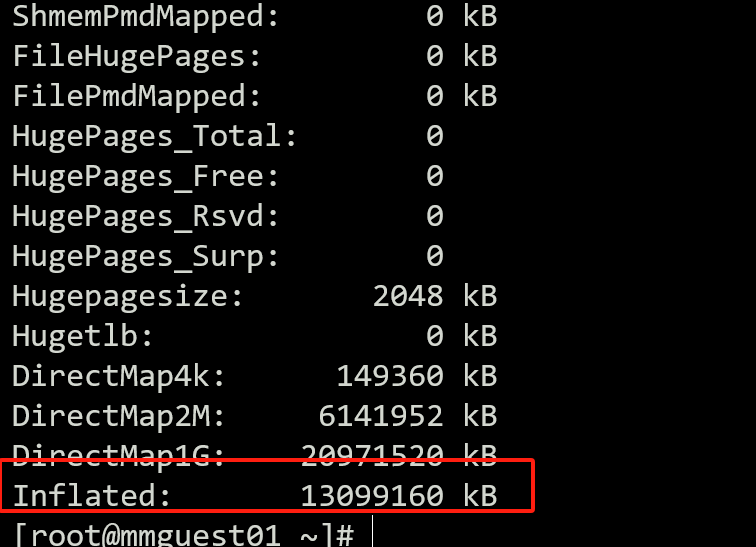
<size unit='KiB'>8388608</size>

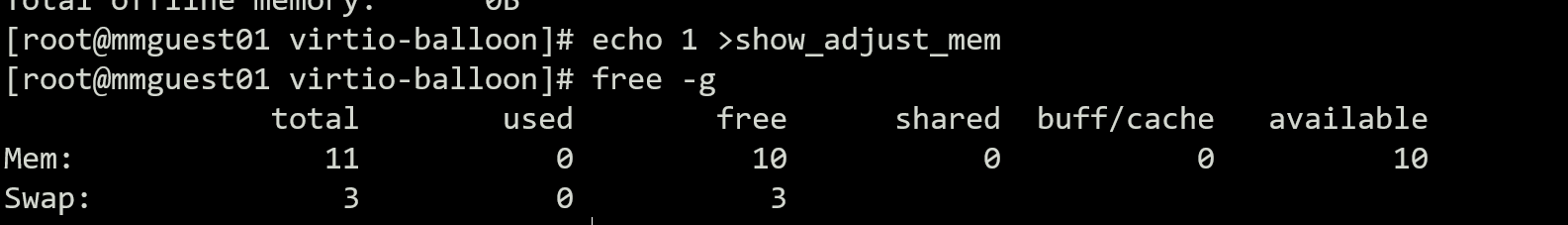
<node>0</node>

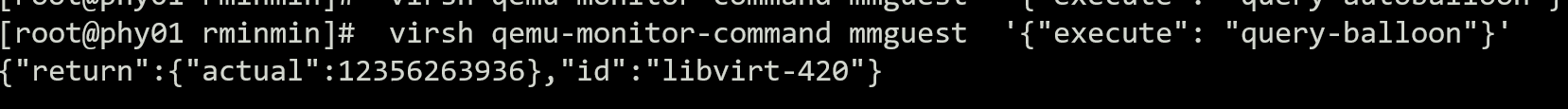
</target>

</memory>







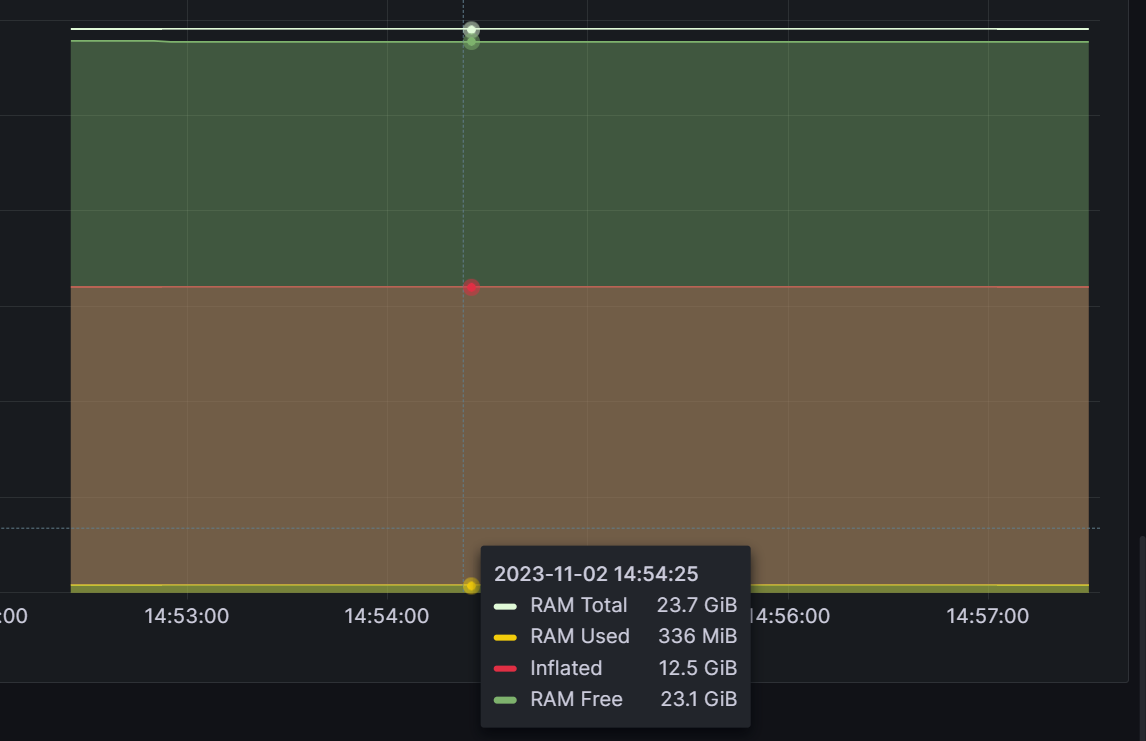




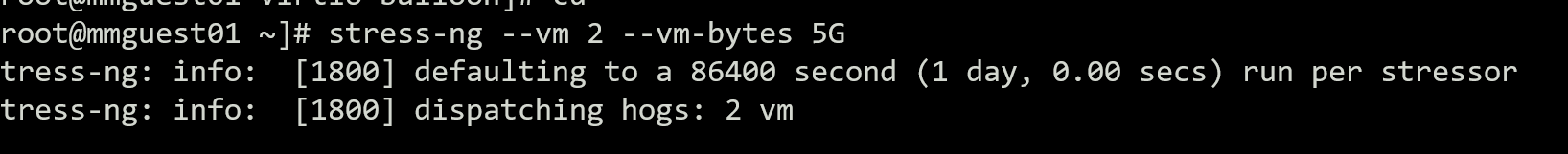
1. 内存压力测试

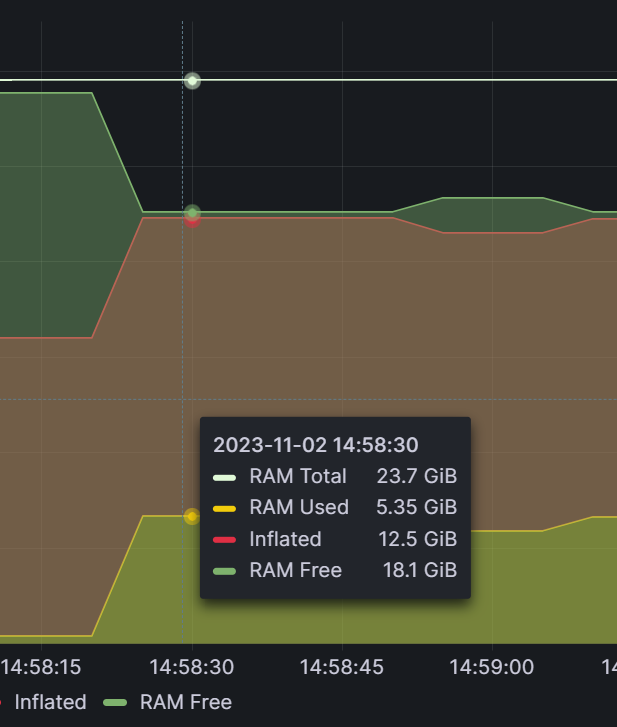
高水位：50% 低水位：5% 粒度：10%

初始情况如下：

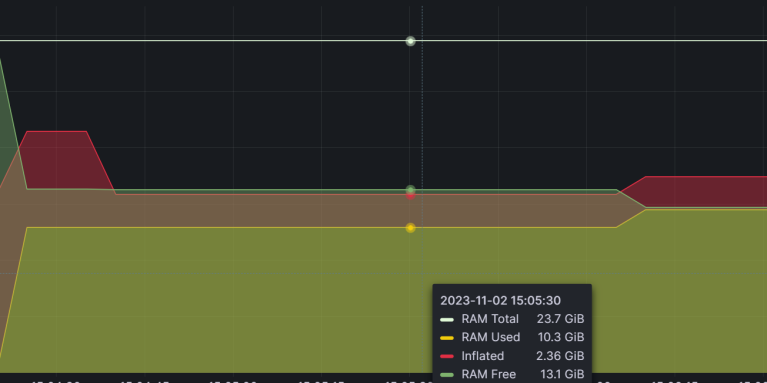


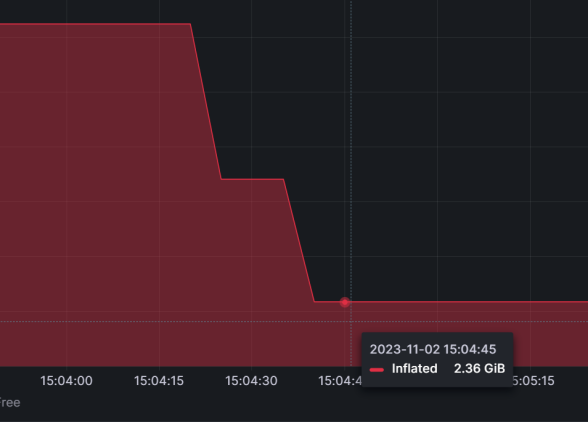
运行内存5G压力测试，在稳定状态区



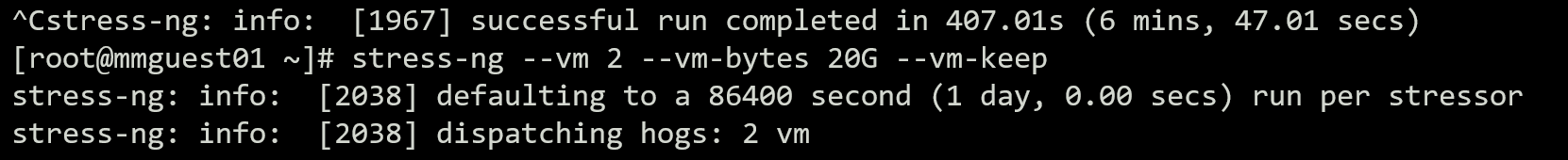


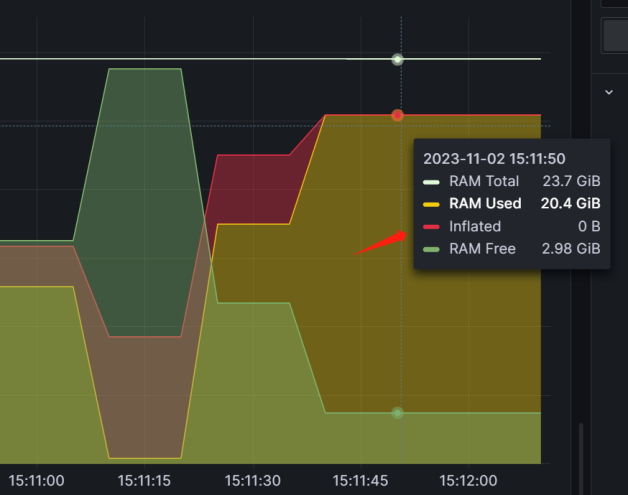
增加内存压测10G内存，触发低水位线释放内存，最终inflated 内存稳定在2G左右

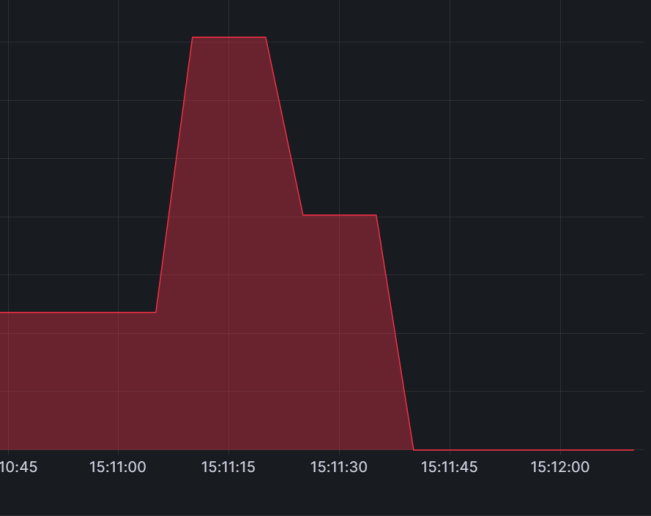




再次加压20G内存，释放所有inflated 内存给guest。图中inflated，是因为退出了压测程序，因为设置的自动调整周期是500ms，所以是比较敏感的。







参考链接：

https://maggie262.github.io/img/in-post/post-vb-virtio-balloon.svg