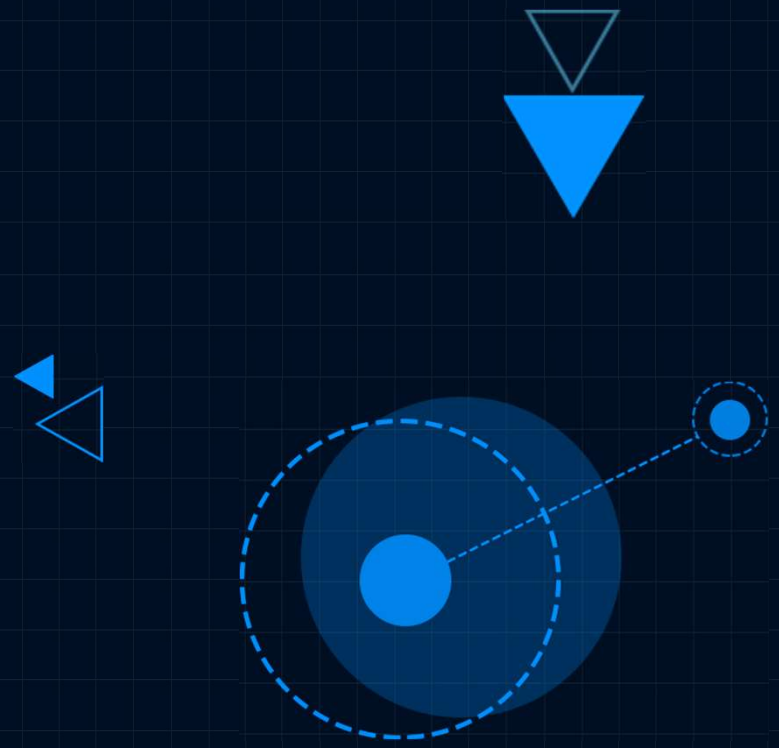
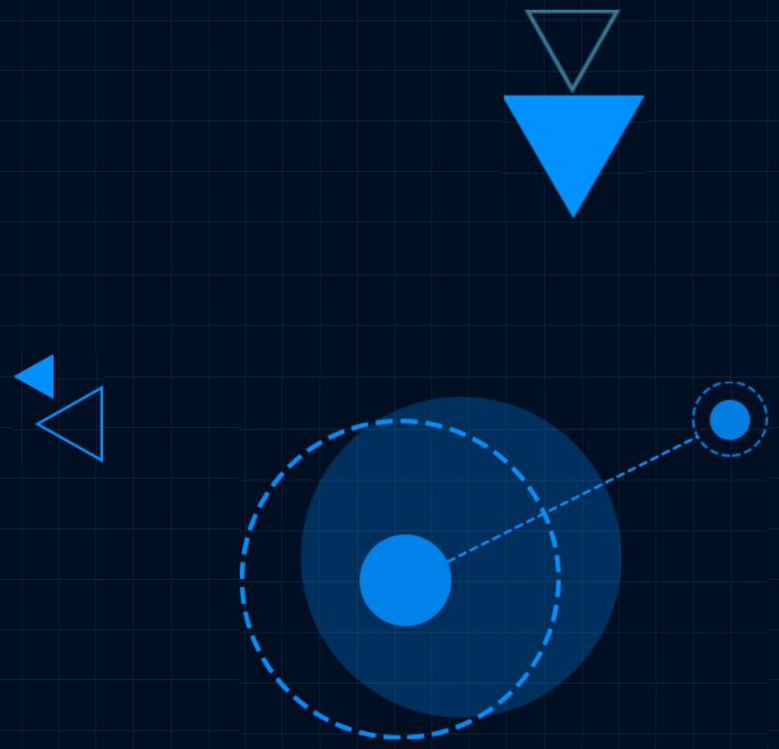
KVM故障预处理

崔凡钦
阿里云虚拟化工程师



目录

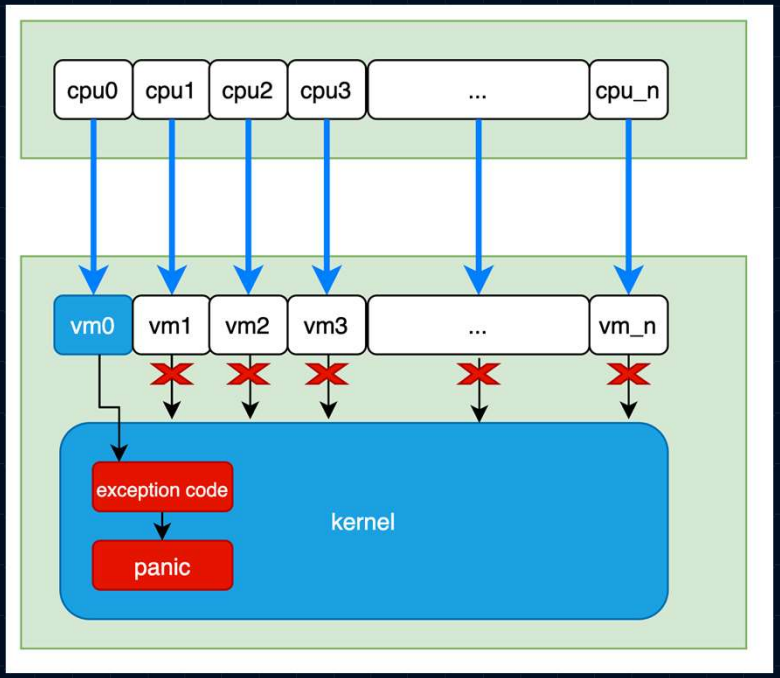
1. 背景



背景和目标

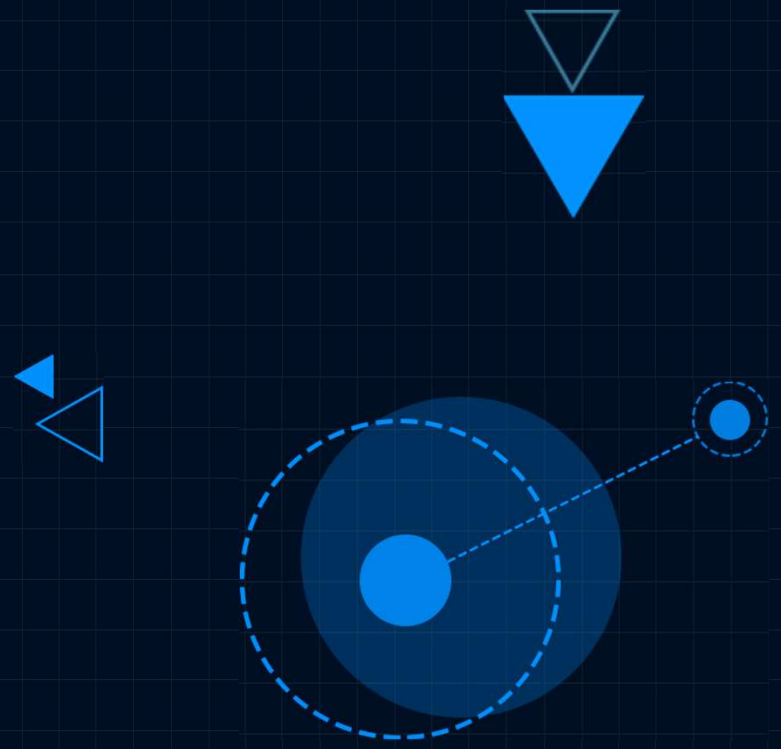
- 背景
  a. 现在server cpu数目越来越多，单台server上同时运行几十台vm甚至上百台vm，一台vm触发异常导致整台server上的所有vm不可用，造成损失严重。

- 目标
  a. 避免单台vm出现的异常导致所有vm不可用，其他vm能够继续提供服务。
  b. 优化代码或者增加测试等手段不太容易覆盖到所有场景，所以我们提出在kernel和kvm之间增加故障预处理逻辑的方案。



背景示意图

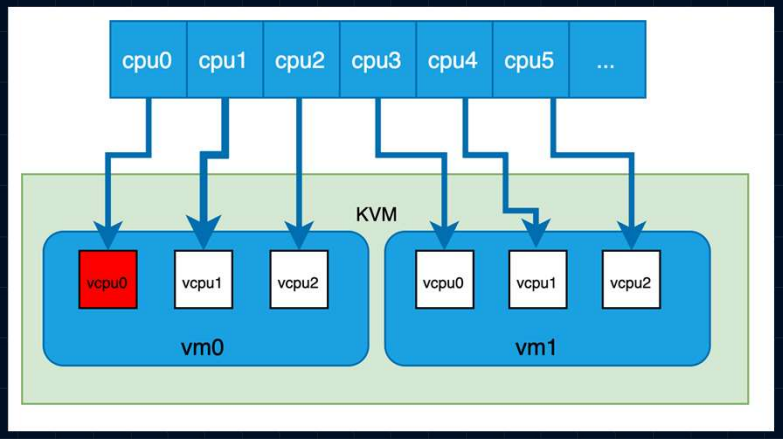
2. 方案介绍



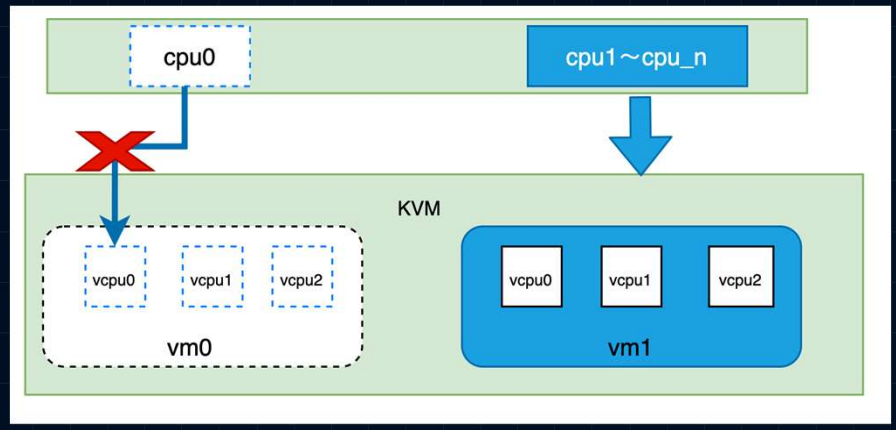
方案简介：

- 基本原理：

一旦某个vm上的vcpu线程运行出现异常时，由对应的kvm处理函数接管异常，stop vcpu， stop vm，并且offline pcpu。其他vm会继续在其他cpu上运行。



vcpu0发生异常

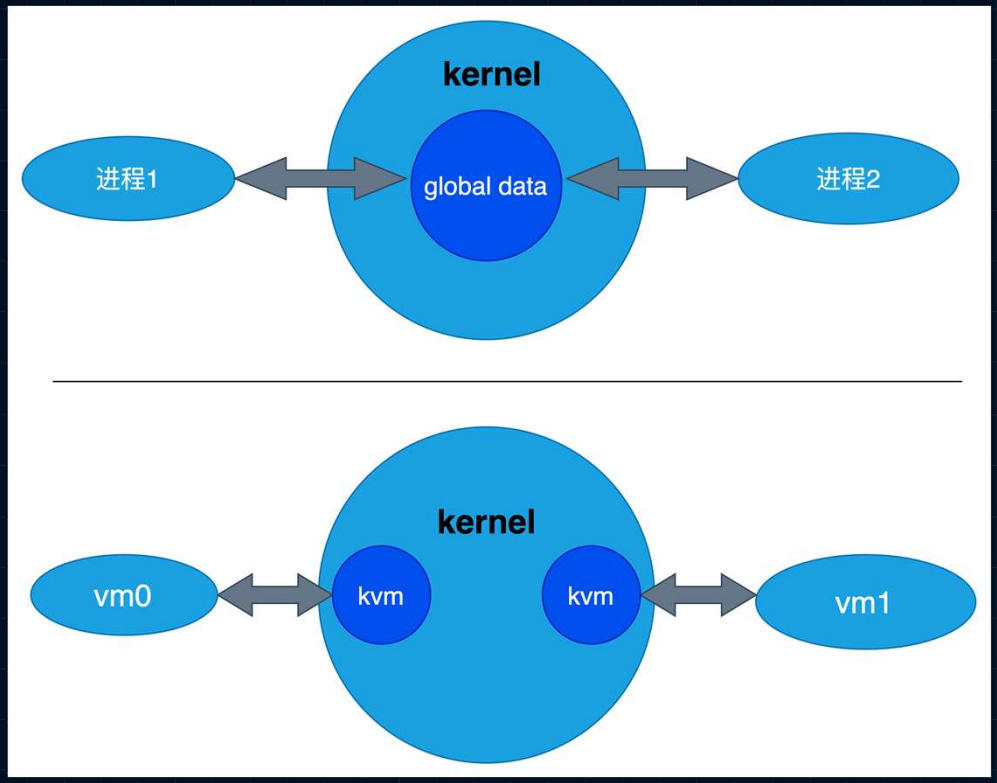


异常处理

方案可行性:

a. Kvm设计时是一个per-vm系统，虽然使用同一套代码，但每个vm彼此执行逻辑和数据区域都是隔离的。每个vm之间以及vm与kernel 几乎不存在耦合性。

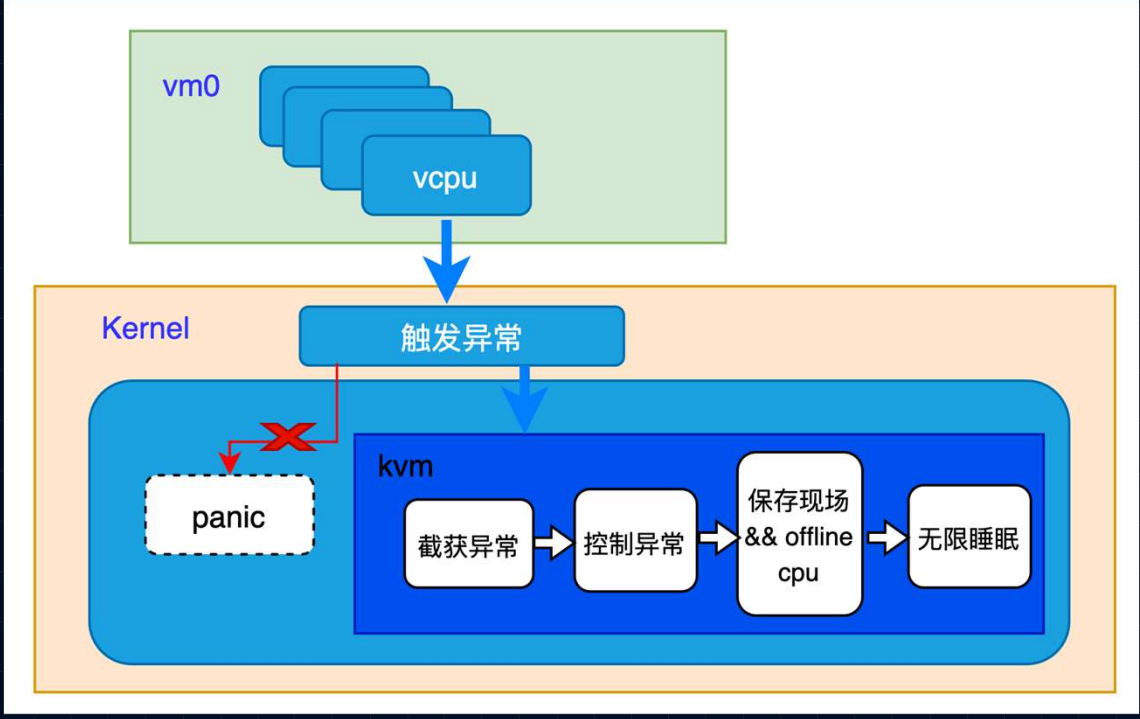
b. 从历史数据来看，bugzilla有158个kvm相关bug, 18个可能会导致host 不正常，所有都是单vcpu出问题，导致整个系统无法服务。



可行性说明

方案处理流程：

- 异常处理流程：



异常处理流程

截获异常

- 截获哪些异常?

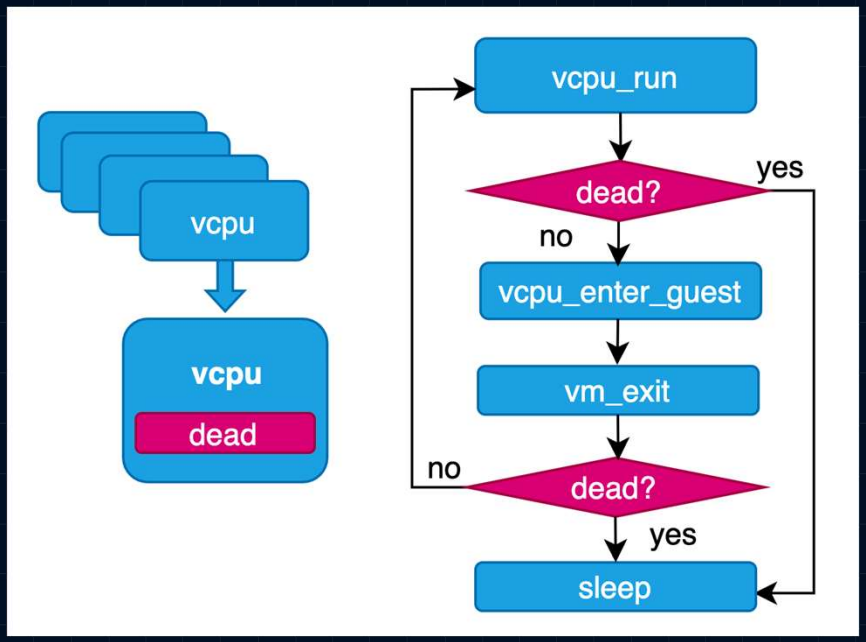
  Hardlockup 、GP、 NULL指针异常;

- 判断是否能够截获

  a. 判断当前线程是否是vm 某个vcpu线程;
  b. 检查是否处于kvm module;
  c. 次数限制，如果多次触发截获的话，认为处理失败。

控制异常扩散:

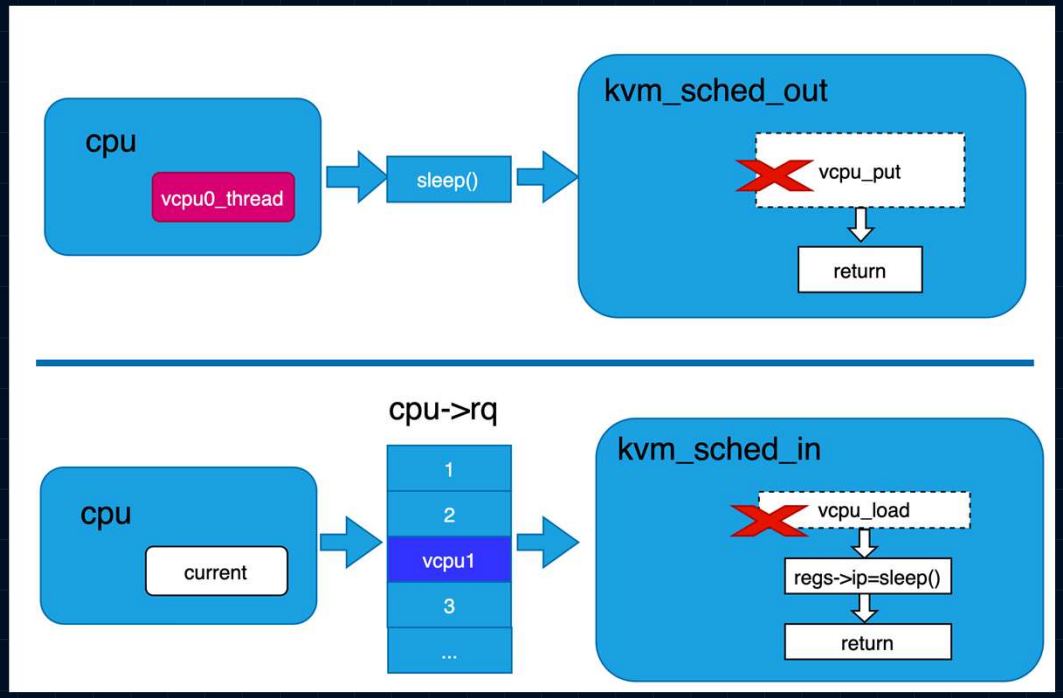
- 出现异常后，阻塞同vm内的vcpu线程，防止再次发生异常



添加阻塞点

控制异常扩散:

- 阻止其他vcpu调度

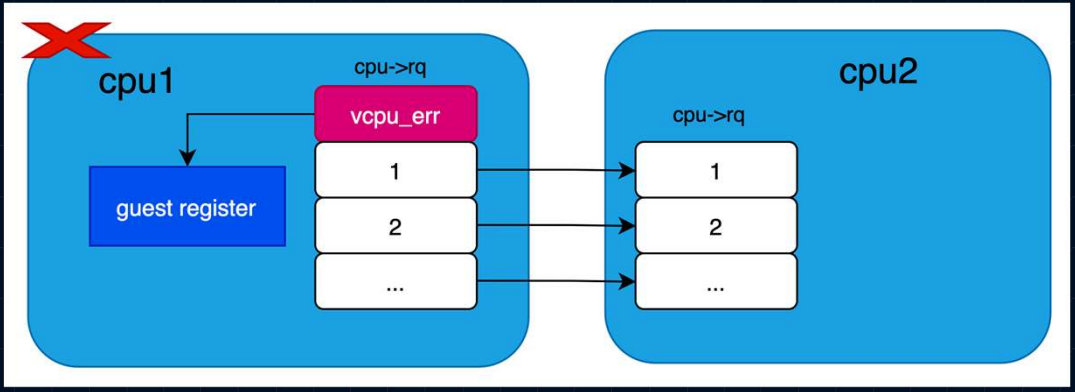


防止vcpu调度

控制异常扩散:

- Offline PCPU

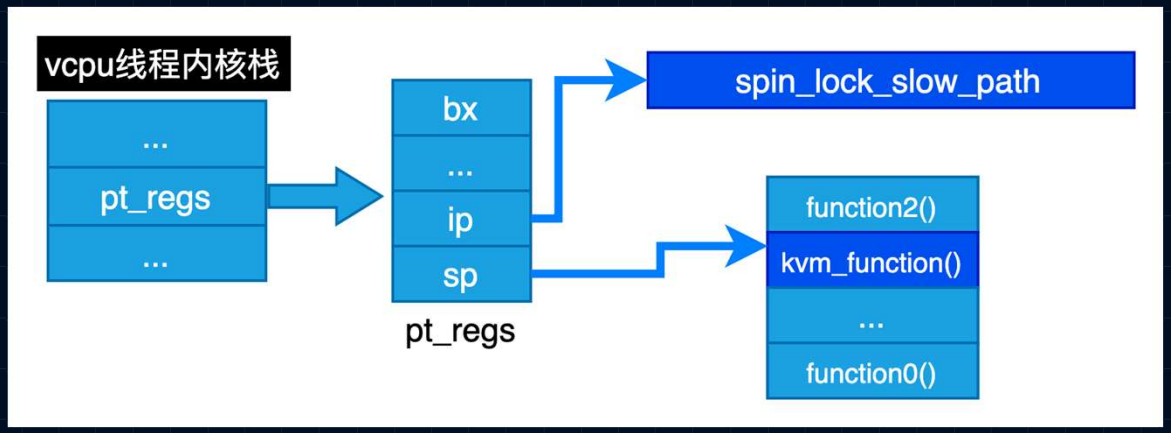
  a: 因为有些寄存器并不是立即恢复到host state，所以需要把当前pcpu offline掉。



- 共享数据vm_list, 不适用这套方案

异常特殊性:

- 死锁上下文



- 防止触发异常时，仍然持有锁的情况，进行处理。

系统链路:

- 释放vfio、后端资源等
- 主动运维，热迁移其它vm

Thanks_