Rust China Conf 2020

Shenzhen, China

2020conf.rustcc.cn

Rust 系统编程在 StratoVirt 中的实践

杨杰
yangjieyj.yang@huawei.com

About Me

- 华为高级工程师
- StratoVirt 主要贡献者
- 具备多年 Linux 系统及虚拟化开发经验

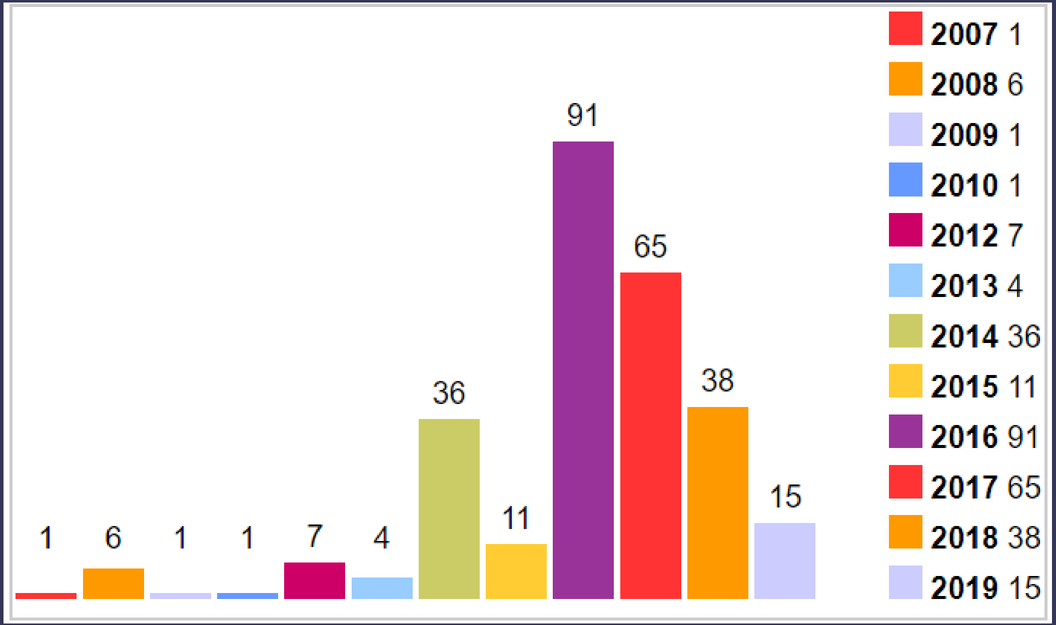
What is system programming?

- 任何不是应用程序的软件
  - BIOS 、固件、引导加载程序、操作系统内核、 hypervisor
- 与硬件交互，直接操作寄存器或内存地址
- 追求高性能和高可靠性
- Assembly -> C

Problems of C

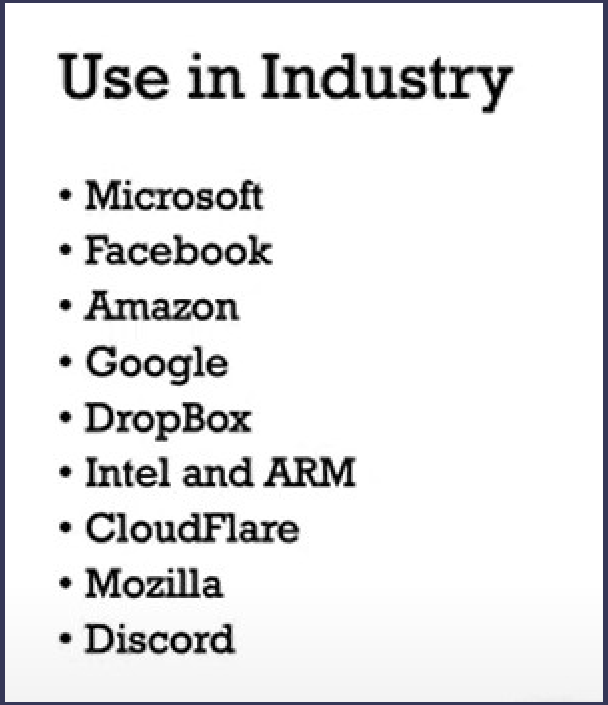
- 无内存管理机制
  - Use-after-free
  - 空指针访问
  - Double-free
  - 内存泄漏
  - 缓冲区溢出
- 语法限制、变量约束不严格
- 不检查数组下标
- 并发访问控制
- 只提供最基础的数据类型和接口

虚拟化组件	开源代码量
libvirt	560 KLOC
QEMU	1570 KLOC
KVM	30 KLOC



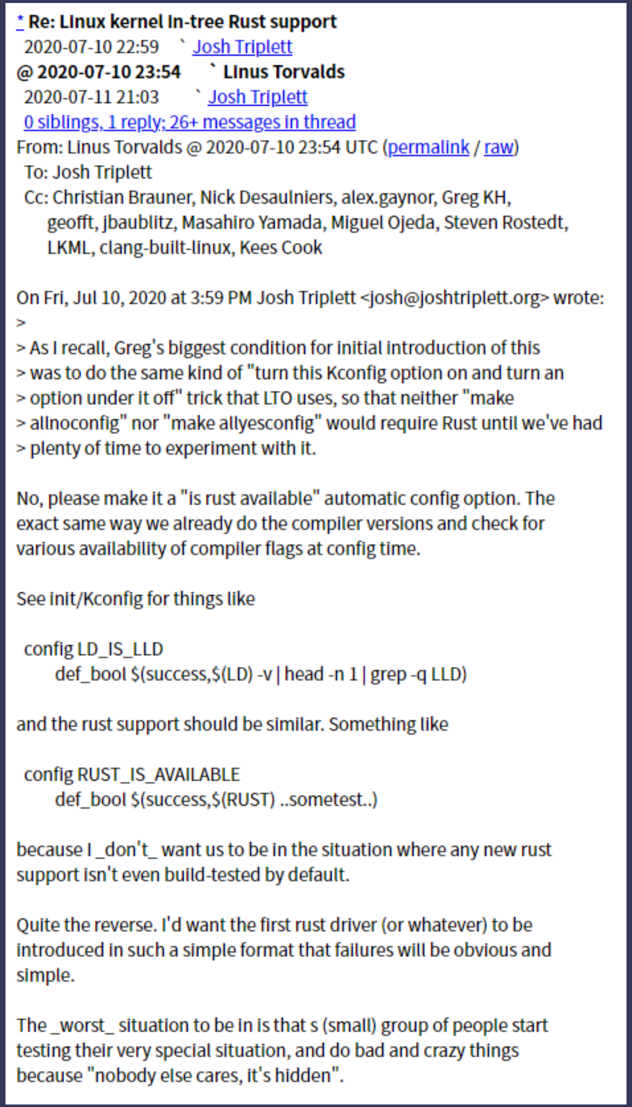
Why Rust?

- 内存安全
- 零运行时的性能
- 友好的工具链
- 充满活力的社区 & 生态
- ……



Use in Industry

- Microsoft
- Facebook
- Amazon
- Google
- DropBox
- Intel and ARM
- CloudFlare
- Mozilla
- Discord



*** Re: Linux kernel in-tree Rust support**
 2020-07-10 22:59 ` Josh Triplett
@ 2020-07-10 23:54 ` Linus Torvalds
 2020-07-11 21:03 ` Josh Triplett
 0 siblings, 1 reply; 26+ messages in thread
From: Linus Torvalds @ 2020-07-10 23:54 UTC (permalink / raw)
 To: Josh Triplett
 Cc: Christian Brauner, Nick Desaulniers, alex.gaynor, Greg KH,
 geofft, jbaublitz, Masahiro Yamada, Miguel Ojeda, Steven Rostedt,
 LKML, clang-built-linux, Kees Cook

On Fri, Jul 10, 2020 at 3:59 PM Josh Triplett <josh@joshtriplett.org> wrote:
>
> As I recall, Greg's biggest condition for initial introduction of this
> was to do the same kind of "turn this Kconfig option on and turn an
> option under it off" trick that LTO uses, so that neither "make
> allnoconfig" nor "make allyesconfig" would require Rust until we've had
> plenty of time to experiment with it.

No, please make it a "is rust available" automatic config option. The
exact same way we already do the compiler versions and check for
various availability of compiler flags at config time.

See init/Kconfig for things like

```
 config LD_IS_LLD
     def_bool $(success,$(LD) -v | head -n 1 | grep -q LLD)
```

and the rust support should be similar. Something like

```
 config RUST_IS_AVAILABLE
     def_bool $(success,$(RUST) ..sometest..)
```

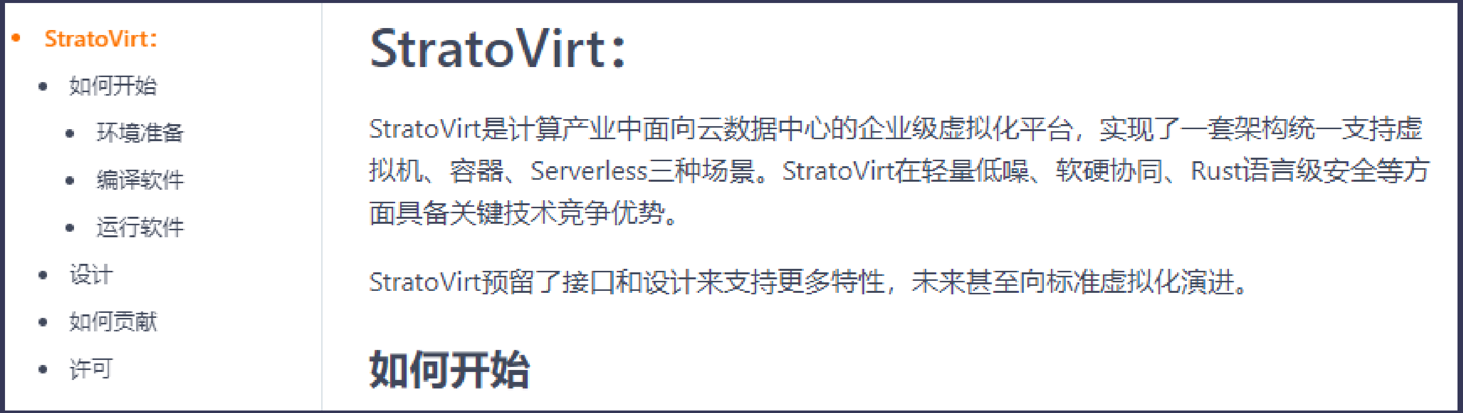
because I _don't_ want us to be in the situation where any new rust
support isn't even build-tested by default.

Quite the reverse. I'd want the first rust driver (or whatever) to be
introduced in such a simple format that failures will be obvious and
simple.

The _worst_ situation to be in is that s (small) group of people start
testing their very special situation, and do bad and crazy things
because "nobody else cares, it's hidden".

What is StratoVirt?

- 基于 Rust 开发的面向云数据中心的企业级虚拟化平台
- 轻量、安全，实现一套架构统一支持虚拟机、容器、 Serverless 三种场景
- 支持 x86_64 和 aarch64



- StratoVirt：
  - 如何开始
    - 环境准备
    - 编译软件
    - 运行软件
  - 设计
  - 如何贡献
  - 许可

StratoVirt：

StratoVirt是计算产业中面向云数据中心的企业级虚拟化平台，实现了一套架构统一支持虚拟机、容器、Serverless三种场景。StratoVirt在轻量低噪、软硬协同、Rust语言级安全等方面具备关键技术竞争优势。

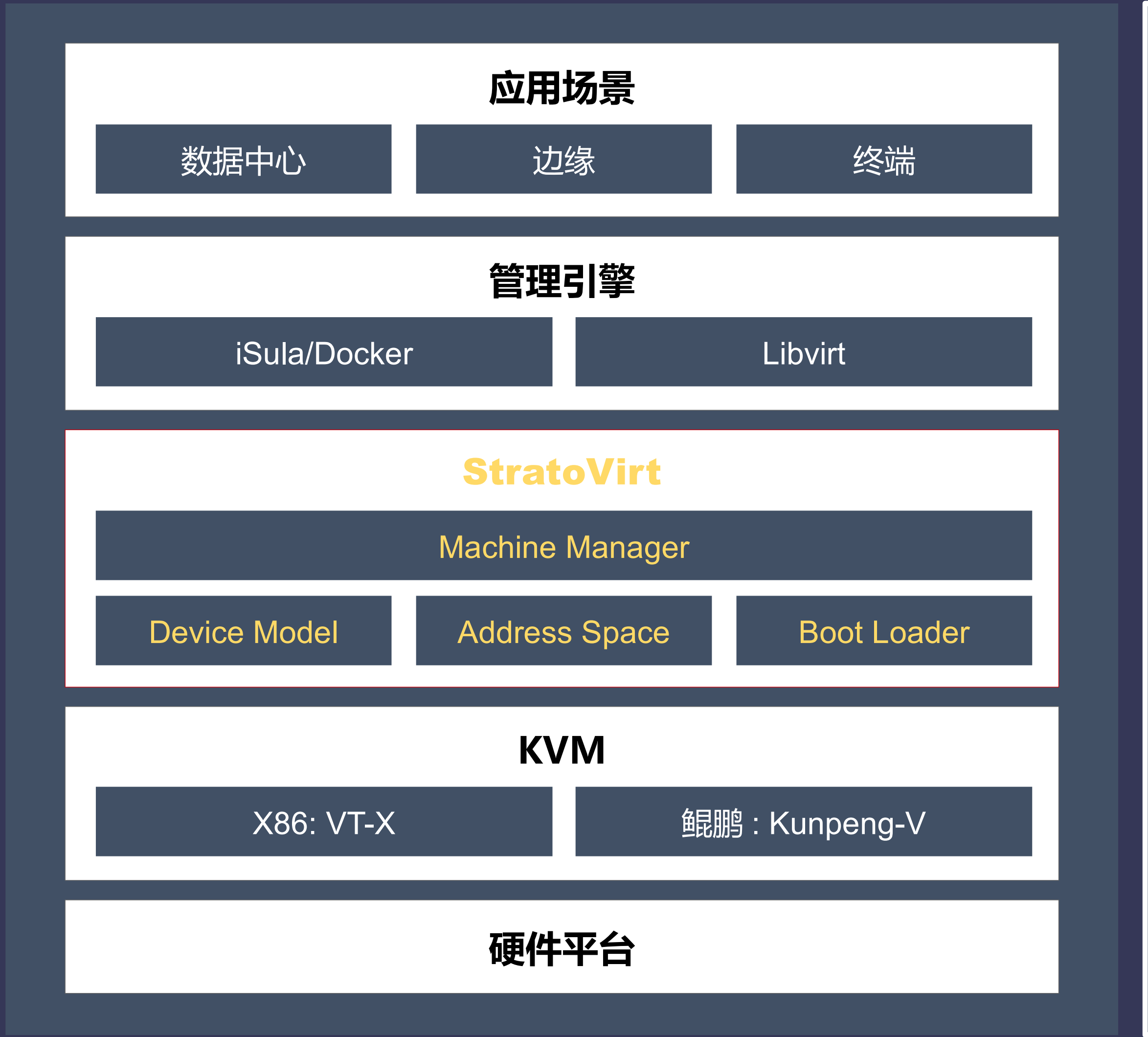
StratoVirt预留了接口和设计来支持更多特性，未来甚至向标准虚拟化演进。

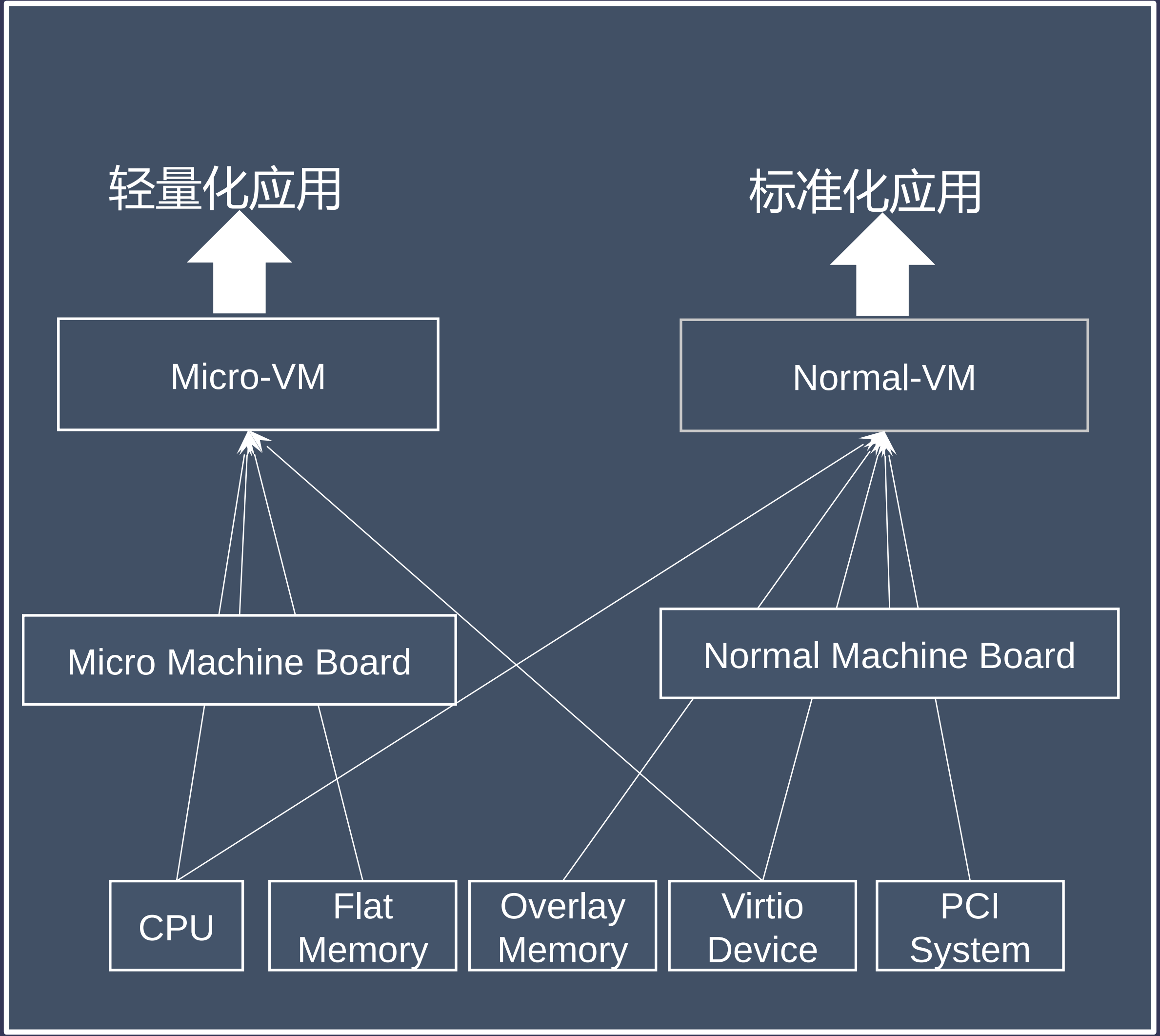
如何开始

https://gitee.com/openeuler/stratovirt

StratoVirt

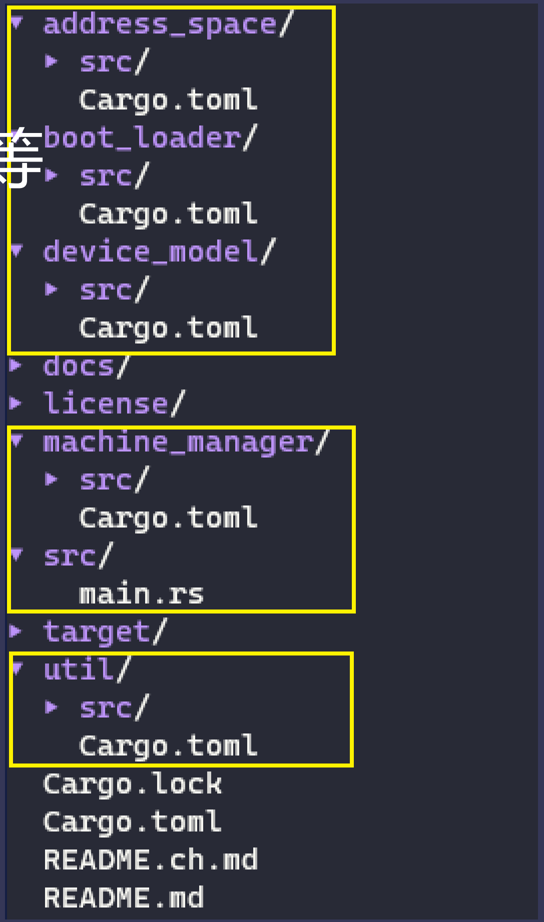
StratoVirt Architecture





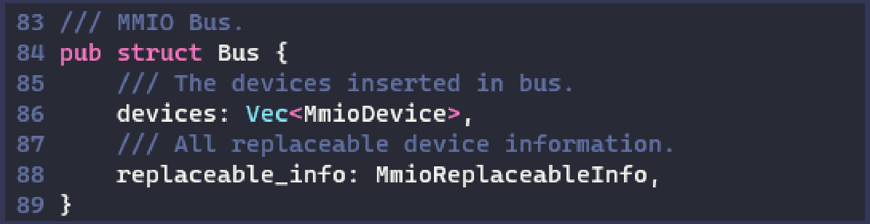
StratoVirt Realization

- 模块化： 1 个 bin + 5 个 lib crate
- 引用第三方 Rust 库 11 个，涵盖日志、错误处理等
- 支持 libc 的静态链接 (musl) 及动态链接
- 动态链接第三方 C 库
- 30 余个 Linux 系统调用

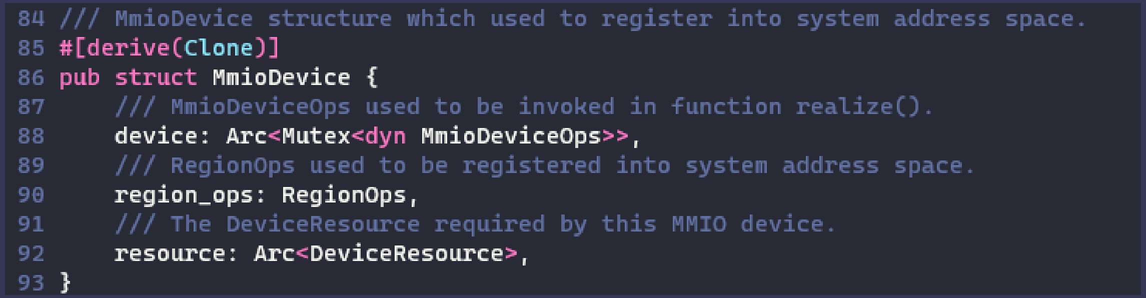


```
▾ address_space/
  ▸ src/
    Cargo.toml
▾ boot_loader/
  ▸ src/
    Cargo.toml
▾ device_model/
  ▸ src/
    Cargo.toml
▸ docs/
▸ license/
▾ machine_manager/
  ▸ src/
    Cargo.toml
▾ src/
    main.rs
▸ target/
▾ util/
  ▸ src/
    Cargo.toml
  Cargo.lock
  Cargo.toml
  README.ch.md
  README.md
```

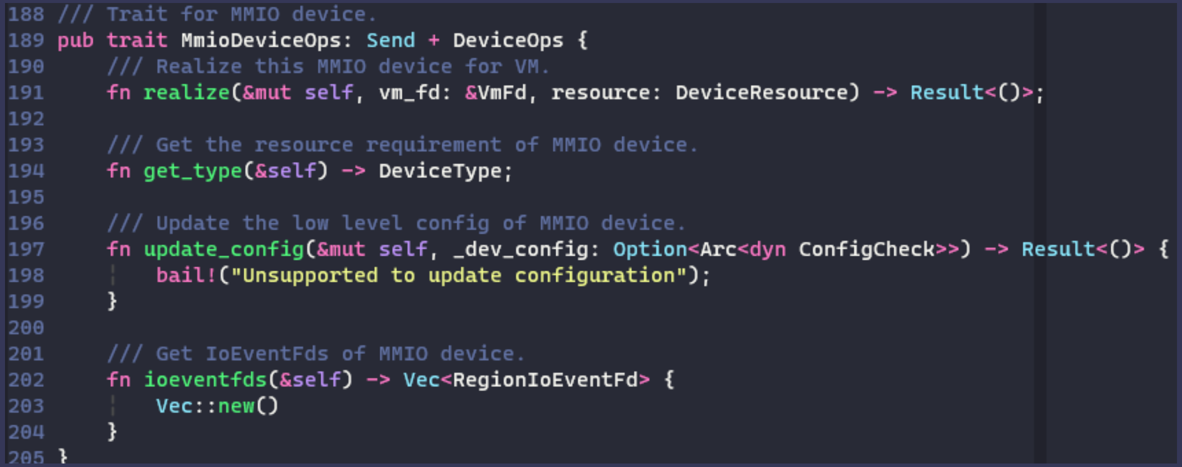
StratoVirt – Virtio-mmio



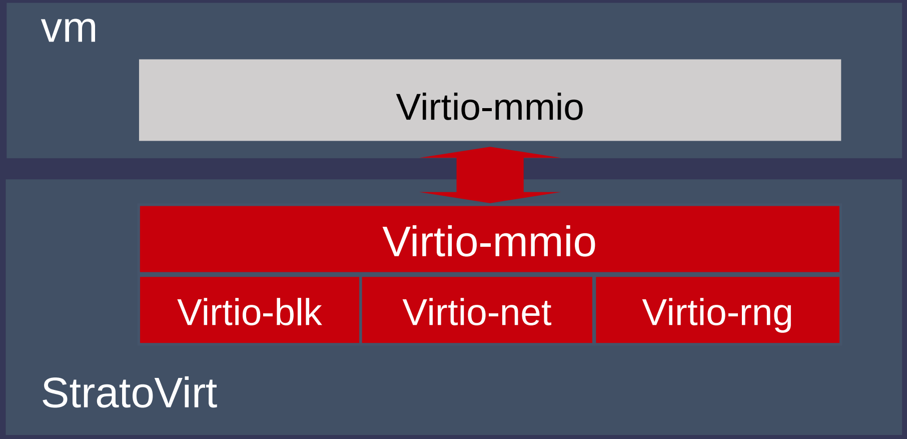
```rust
/// MMIO Bus.
pub struct Bus {
    /// The devices inserted in bus.
    devices: Vec<MmioDevice>,
    /// All replaceable device information.
    replaceable_info: MmioReplaceableInfo,
}
```

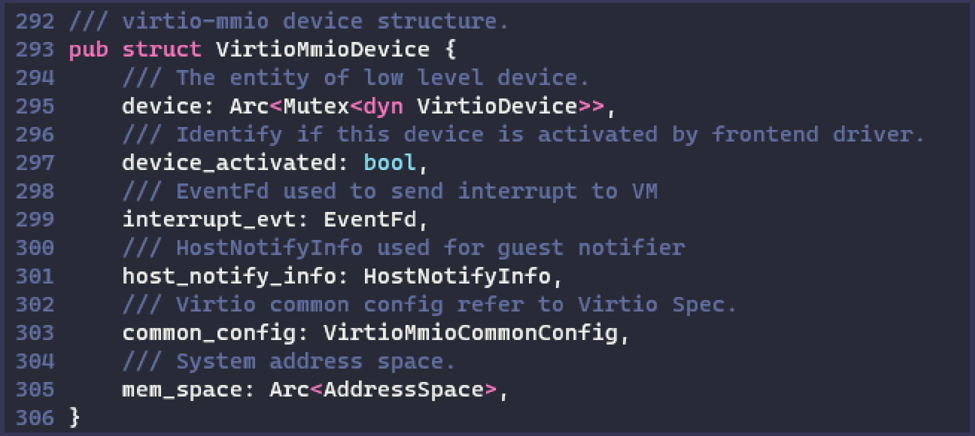


```rust
/// MmioDevice structure which used to register into system address space.
#[derive(Clone)]
pub struct MmioDevice {
    /// MmioDeviceOps used to be invoked in function realize().
    device: Arc<Mutex<dyn MmioDeviceOps>>,
    /// RegionOps used to be registered into system address space.
    region_ops: RegionOps,
    /// The DeviceResource required by this MMIO device.
    resource: Arc<DeviceResource>,
}
```



```rust
/// Trait for MMIO device.
pub trait MmioDeviceOps: Send + DeviceOps {
    /// Realize this MMIO device for VM.
    fn realize(&mut self, vm_fd: &VmFd, resource: DeviceResource) -> Result<()>;

    /// Get the resource requirement of MMIO device.
    fn get_type(&self) -> DeviceType;

    /// Update the low level config of MMIO device.
    fn update_config(&mut self, _dev_config: Option<Arc<dyn ConfigCheck>>) -> Result<()> {
        bail!("Unsupported to update configuration");
    }

    /// Get IoEventFds of MMIO device.
    fn ioeventfds(&self) -> Vec<RegionIoEventFd> {
        Vec::new()
    }
}
```

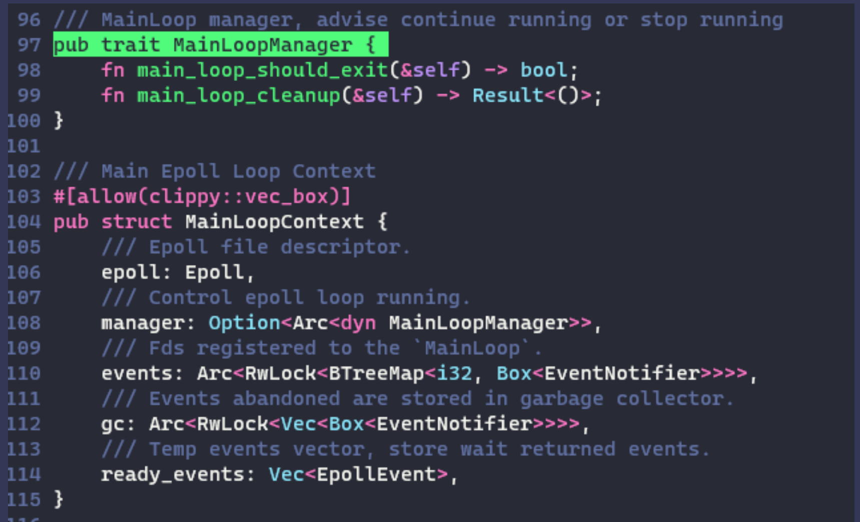




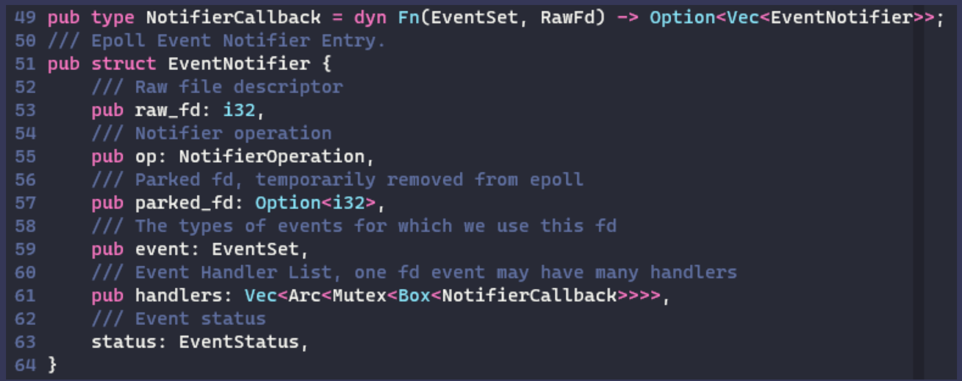
```rust
/// virtio-mmio device structure.
pub struct VirtioMmioDevice {
    /// The entity of low level device.
    device: Arc<Mutex<dyn VirtioDevice>>,
    /// Identify if this device is activated by frontend driver.
    device_activated: bool,
    /// EventFd used to send interrupt to VM
    interrupt_evt: EventFd,
    /// HostNotifyInfo used for guest notifier
    host_notify_info: HostNotifyInfo,
    /// Virtio common config refer to Virtio Spec.
    common_config: VirtioMmioCommonConfig,
    /// System address space.
    mem_space: Arc<AddressSpace>,
}
```

StratoVirt – IO handlering

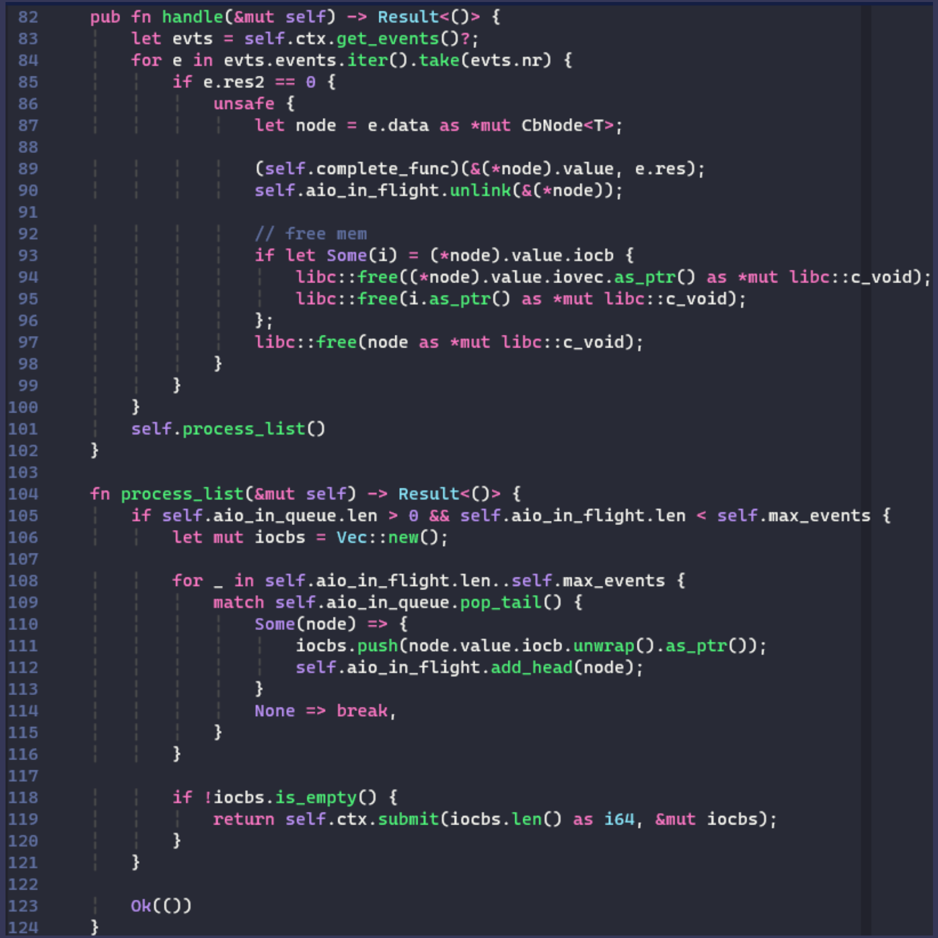
- 单线程 epoll： 满足基本的 IO 性能需求，降低内存开销
- 存储设备后端支持异步 IO



```rust
/// MainLoop manager, advise continue running or stop running
pub trait MainLoopManager {
    fn main_loop_should_exit(&self) -> bool;
    fn main_loop_cleanup(&self) -> Result<()>;
}

/// Main Epoll Loop Context
#[allow(clippy::vec_box)]
pub struct MainLoopContext {
    /// Epoll file descriptor.
    epoll: Epoll,
    /// Control epoll loop running.
    manager: Option<Arc<dyn MainLoopManager>>,
    /// Fds registered to the `MainLoop`.
    events: Arc<RwLock<BTreeMap<i32, Box<EventNotifier>>>>,
    /// Events abandoned are stored in garbage collector.
    gc: Arc<RwLock<Vec<Box<EventNotifier>>>>,
    /// Temp events vector, store wait returned events.
    ready_events: Vec<EpollEvent>,
}
```



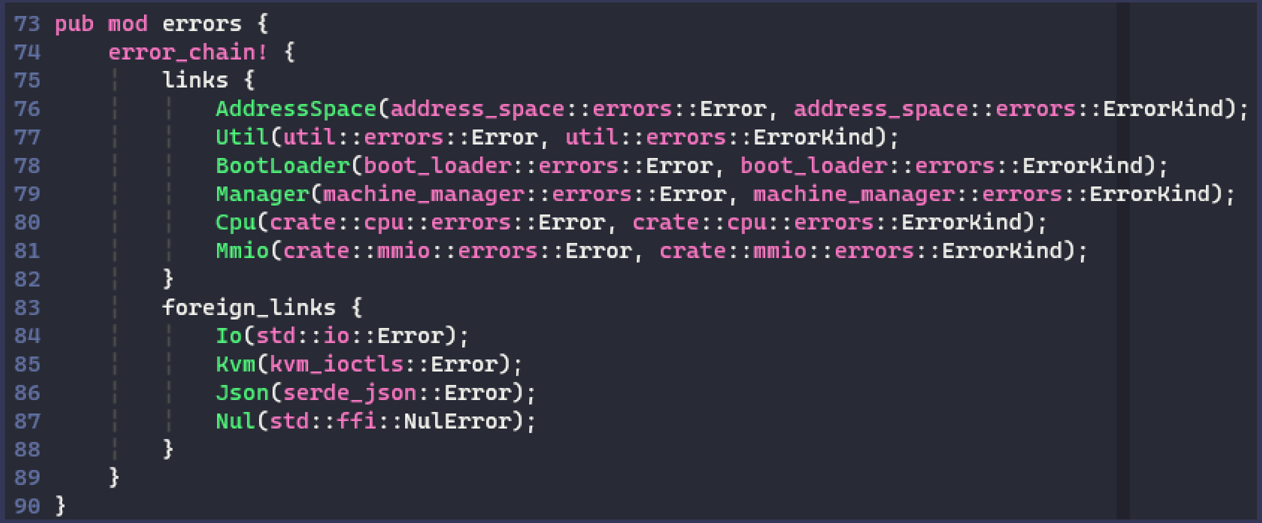
```rust
pub type NotifierCallback = dyn Fn(EventSet, RawFd) -> Option<Vec<EventNotifier>>;
/// Epoll Event Notifier Entry.
pub struct EventNotifier {
    /// Raw file descriptor
    pub raw_fd: i32,
    /// Notifier operation
    pub op: NotifierOperation,
    /// Parked fd, temporarily removed from epoll
    pub parked_fd: Option<i32>,
    /// The types of events for which we use this fd
    pub event: EventSet,
    /// Event Handler List, one fd event may have many handlers
    pub handlers: Vec<Arc<Mutex<Box<NotifierCallback>>>>,
    /// Event status
    status: EventStatus,
}
```



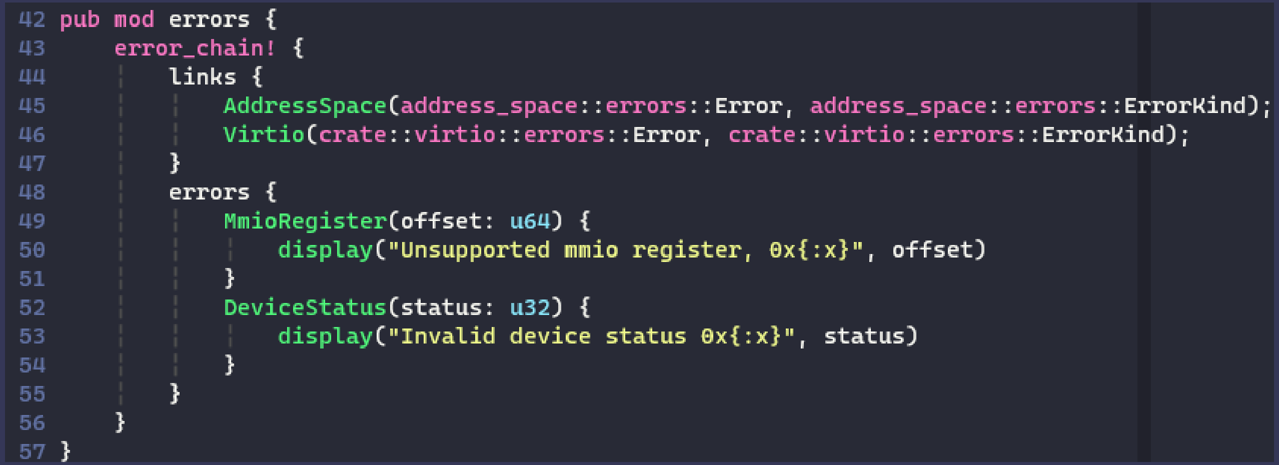
```rust
    pub fn handle(&mut self) -> Result<()> {
        let evts = self.ctx.get_events()?;
        for e in evts.events.iter().take(evts.nr) {
            if e.res2 == 0 {
                unsafe {
                    let node = e.data as *mut CbNode<T>;

                    (self.complete_func)(&(*node).value, e.res);
                    self.aio_in_flight.unlink(&(*node));

                    // free mem
                    if let Some(i) = (*node).value.iocb {
                        libc::free((*node).value.iovec.as_ptr() as *mut libc::c_void);
                        libc::free(i.as_ptr() as *mut libc::c_void);
                    };
                    libc::free(node as *mut libc::c_void);
                }
            }
        }
        self.process_list()
    }

    fn process_list(&mut self) -> Result<()> {
        if self.aio_in_queue.len > 0 && self.aio_in_flight.len < self.max_events {
            let mut iocbs = Vec::new();

            for _ in self.aio_in_flight.len..self.max_events {
                match self.aio_in_queue.pop_tail() {
                    Some(node) => {
                        iocbs.push(node.value.iocb.unwrap().as_ptr());
                        self.aio_in_flight.add_head(node);
                    }
                    None => break,
                }
            }

            if !iocbs.is_empty() {
                return self.ctx.submit(iocbs.len() as i64, &mut iocbs);
            }
        }

        Ok(())
    }
```

StratoVirt – Error handlering

- 提高可维性
- 代码优雅、简洁



```rust
pub mod errors {
    error_chain! {
        links {
            AddressSpace(address_space::errors::Error, address_space::errors::ErrorKind);
            Util(util::errors::Error, util::errors::ErrorKind);
            BootLoader(boot_loader::errors::Error, boot_loader::errors::ErrorKind);
            Manager(machine_manager::errors::Error, machine_manager::errors::ErrorKind);
            Cpu(crate::cpu::errors::Error, crate::cpu::errors::ErrorKind);
            Mmio(crate::mmio::errors::Error, crate::mmio::errors::ErrorKind);
        }
        foreign_links {
            Io(std::io::Error);
            Kvm(kvm_ioctls::Error);
            Json(serde_json::Error);
            Nul(std::ffi::NulError);
        }
    }
}
```



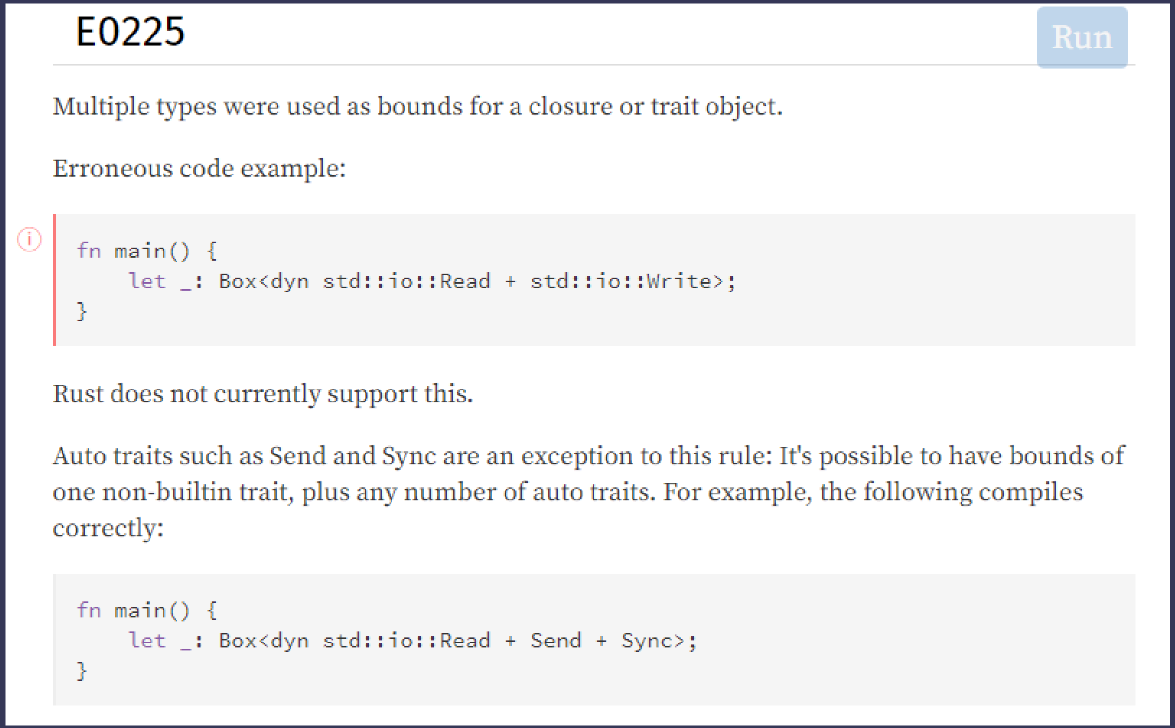
```rust
pub mod errors {
    error_chain! {
        links {
            AddressSpace(address_space::errors::Error, address_space::errors::ErrorKind);
            Virtio(crate::virtio::errors::Error, crate::virtio::errors::ErrorKind);
        }
        errors {
            MmioRegister(offset: u64) {
                display("Unsupported mmio register, 0x{:x}", offset)
            }
            DeviceStatus(status: u32) {
                display("Invalid device status 0x{:x}", status)
            }
        }
    }
}
```

Challenges

- Unsafe
  - 调用 C 接口
  - 性能敏感
  - 反序列化
  - 访问全局可变变量
  - 链表

Challenges

- Trait object



E0225

Run

Multiple types were used as bounds for a closure or trait object.

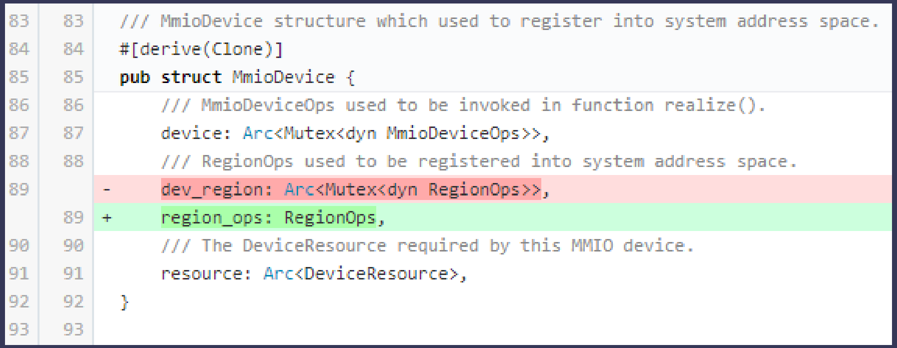
Erroneous code example:

```
fn main() {
    let _: Box<dyn std::io::Read + std::io::Write>;
}
```

Rust does not currently support this.

Auto traits such as Send and Sync are an exception to this rule: It's possible to have bounds of one non-builtin trait, plus any number of auto traits. For example, the following compiles correctly:

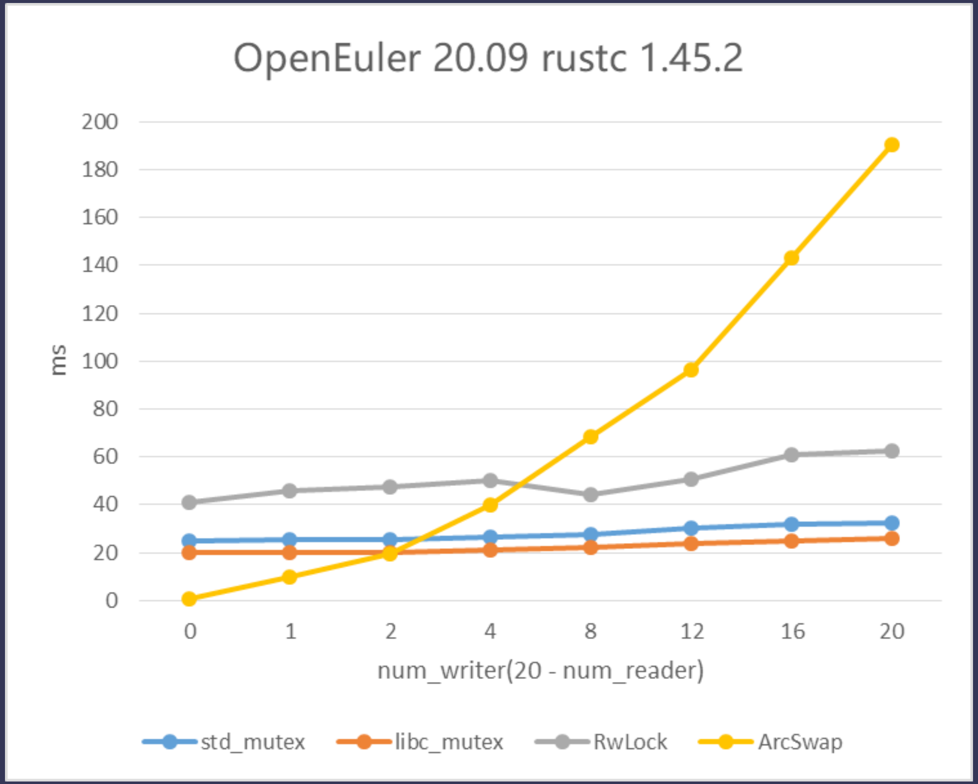
```
fn main() {
    let _: Box<dyn std::io::Read + Send + Sync>;
}
```



```
/// MmioDevice structure which used to register into system address space.
#[derive(Clone)]
pub struct MmioDevice {
    /// MmioDeviceOps used to be invoked in function realize().
    device: Arc<Mutex<dyn MmioDeviceOps>>,
    /// RegionOps used to be registered into system address space.
-   dev_region: Arc<Mutex<dyn RegionOps>>,
+   region_ops: RegionOps,
    /// The DeviceResource required by this MMIO device.
    resource: Arc<DeviceResource>,
}
```

Challenges

- 锁
  - 性能
  - 努力避免死锁

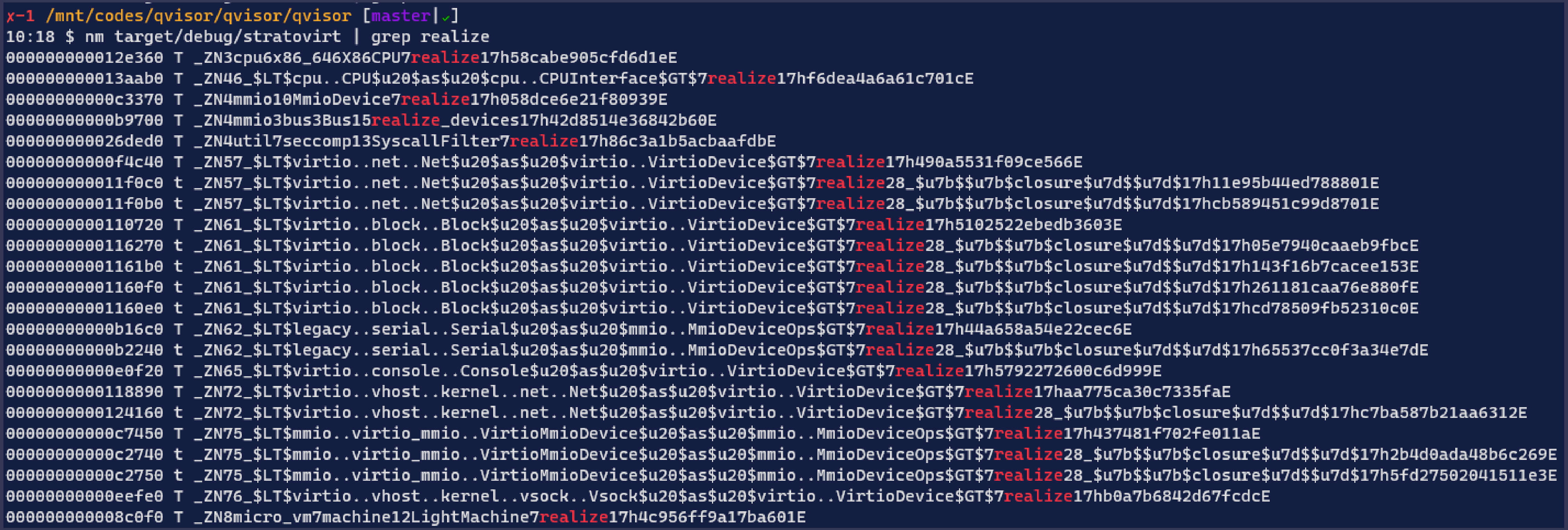


Challenges

- 第三方 crate
  - 可靠
  - 稳定
  - 活跃
  - 满足功能需求

Challenges

- 热补丁技术



```
x-1 /mnt/codes/qvisor/qvisor/qvisor [master|✓]
10:18 $ nm target/debug/stratovirt | grep realize
000000000012e360 T _ZN3cpu6x86_646X86CPU7realize17h58cabe905cfd6d1eE
000000000013aab0 T _ZN46_$LT$cpu..CPU$u20$as$u20$cpu..CPUInterface$GT$7realize17hf6dea4a6a61c701cE
00000000000c3370 T _ZN4mmio10MmioDevice7realize17h058dce6e21f80939E
00000000000b9700 T _ZN4mmio3bus3Bus15realize_devices17h42d8514e36842b60E
0000000000026ded0 T _ZN4util7seccomp13SyscallFilter7realize17h86c3a1b5acbaafdbE
00000000000f4c40 T _ZN57_$LT$virtio..net..Net$u20$as$u20$virtio..VirtioDevice$GT$7realize17h490a5531f09ce566E
000000000011f0c0 t _ZN57_$LT$virtio..net..Net$u20$as$u20$virtio..VirtioDevice$GT$7realize28_$u7b$$u7b$closure$u7d$$u7d$17h11e95b44ed788801E
000000000011f0b0 t _ZN57_$LT$virtio..net..Net$u20$as$u20$virtio..VirtioDevice$GT$7realize28_$u7b$$u7b$closure$u7d$$u7d$17hcb589451c99d8701E
0000000000110720 T _ZN61_$LT$virtio..block..Block$u20$as$u20$virtio..VirtioDevice$GT$7realize17h5102522ebedb3603E
0000000000116270 t _ZN61_$LT$virtio..block..Block$u20$as$u20$virtio..VirtioDevice$GT$7realize28_$u7b$$u7b$closure$u7d$$u7d$17h05e7940caaeb9fbcE
00000000001161b0 t _ZN61_$LT$virtio..block..Block$u20$as$u20$virtio..VirtioDevice$GT$7realize28_$u7b$$u7b$closure$u7d$$u7d$17h143f16b7cacee153E
00000000001160f0 t _ZN61_$LT$virtio..block..Block$u20$as$u20$virtio..VirtioDevice$GT$7realize28_$u7b$$u7b$closure$u7d$$u7d$17h261181caa76e880fE
00000000001160e0 t _ZN61_$LT$virtio..block..Block$u20$as$u20$virtio..VirtioDevice$GT$7realize28_$u7b$$u7b$closure$u7d$$u7d$17hcd78509fb52310c0E
00000000000b16c0 T _ZN62_$LT$legacy..serial..Serial$u20$as$u20$mmio..MmioDeviceOps$GT$7realize17h44a658a54e22cec6E
00000000000b2240 t _ZN62_$LT$legacy..serial..Serial$u20$as$u20$mmio..MmioDeviceOps$GT$7realize28_$u7b$$u7b$closure$u7d$$u7d$17h65537cc0f3a34e7dE
00000000000e0f20 T _ZN65_$LT$virtio..console..Console$u20$as$u20$virtio..VirtioDevice$GT$7realize17h5792272600c6d999E
0000000000118890 T _ZN72_$LT$virtio..vhost..kernel..net..Net$u20$as$u20$virtio..VirtioDevice$GT$7realize17haa775ca30c7335faE
0000000000124160 t _ZN72_$LT$virtio..vhost..kernel..net..Net$u20$as$u20$virtio..VirtioDevice$GT$7realize28_$u7b$$u7b$closure$u7d$$u7d$17hc7ba587b21aa6312E
00000000000c7450 T _ZN75_$LT$mmio..virtio_mmio..VirtioMmioDevice$u20$as$u20$mmio..MmioDeviceOps$GT$7realize17h437481f702fe011aE
00000000000c2740 t _ZN75_$LT$mmio..virtio_mmio..VirtioMmioDevice$u20$as$u20$mmio..MmioDeviceOps$GT$7realize28_$u7b$$u7b$closure$u7d$$u7d$17h2b4d0ada48b6c269E
00000000000c2750 t _ZN75_$LT$mmio..virtio_mmio..VirtioMmioDevice$u20$as$u20$mmio..MmioDeviceOps$GT$7realize28_$u7b$$u7b$closure$u7d$$u7d$17h5fd2750204151le3E
00000000000eefe0 T _ZN76_$LT$virtio..vhost..kernel..vsock..Vsock$u20$as$u20$virtio..VirtioDevice$GT$7realize17hb0a7b6842d67fcdcE
000000000008c0f0 T _ZN8micro_vm7machine12LightMachine7realize17h4c956ff9a17ba601E
```

Rust is still young, but promising