Building Reliable Cross-Cloud Kubernetes Clusters on Spot Instances with Drafter and PVM

Felicitas Pojtinger
@pojntfx

Loophole Labs

Chapters

Commoditization

Silo

Architect

PVM

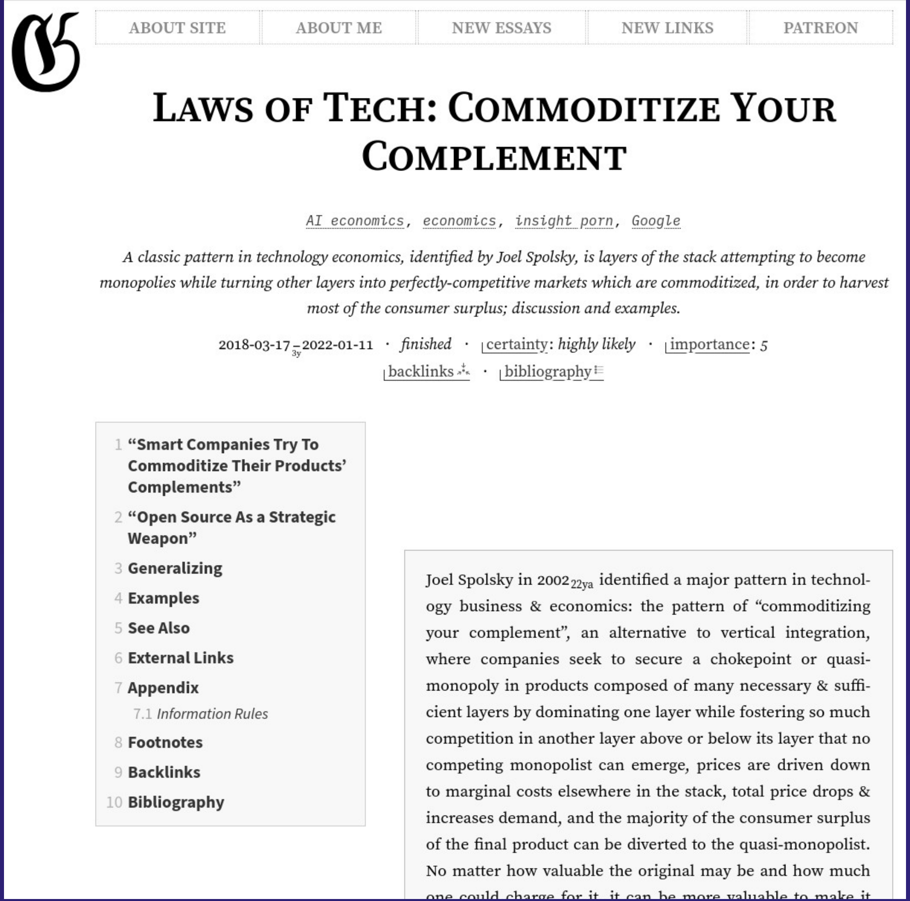
Conduit

Drafter

Felicitas Pojtinger

Fediverse:	@pojntfx@mastodon.social
Bluesky:	@pojntfx.mastodon.social.ap.brid.gy
Github:	@pojntfx
LinkedIn:	in/pojntfx
Web:	felicitas.pojtinger.com

Loophole Labs



ABOUT SITE | ABOUT ME | NEW ESSAYS | NEW LINKS | PATREON

Laws of Tech: Commoditize Your Complement

AI economics, economics, insight porn, Google

A classic pattern in technology economics, identified by Joel Spolsky, is layers of the stack attempting to become monopolies while turning other layers into perfectly-competitive markets which are commoditized, in order to harvest most of the consumer surplus; discussion and examples.

2018-03-17–3y–2022-01-11 · *finished* · certainty: *highly likely* · importance: *5*

backlinks · bibliography

Joel Spolsky in 2002 22ya identified a major pattern in technology business & economics: the pattern of "commoditizing your complement", an alternative to vertical integration, where companies seek to secure a chokepoint or quasi-monopoly in products composed of many necessary & sufficient layers by dominating one layer while fostering so much competition in another layer above or below its layer that no competing monopolist can emerge, prices are driven down to marginal costs elsewhere in the stack, total price drops & increases demand, and the majority of the consumer surplus of the final product can be diverted to the quasi-monopolist. No matter how valuable the original may be and how much one could charge for it, it can be more valuable to make it

https://gwern.net/complement

“Smart Companies Try To Commoditize Their Products’ Complements”

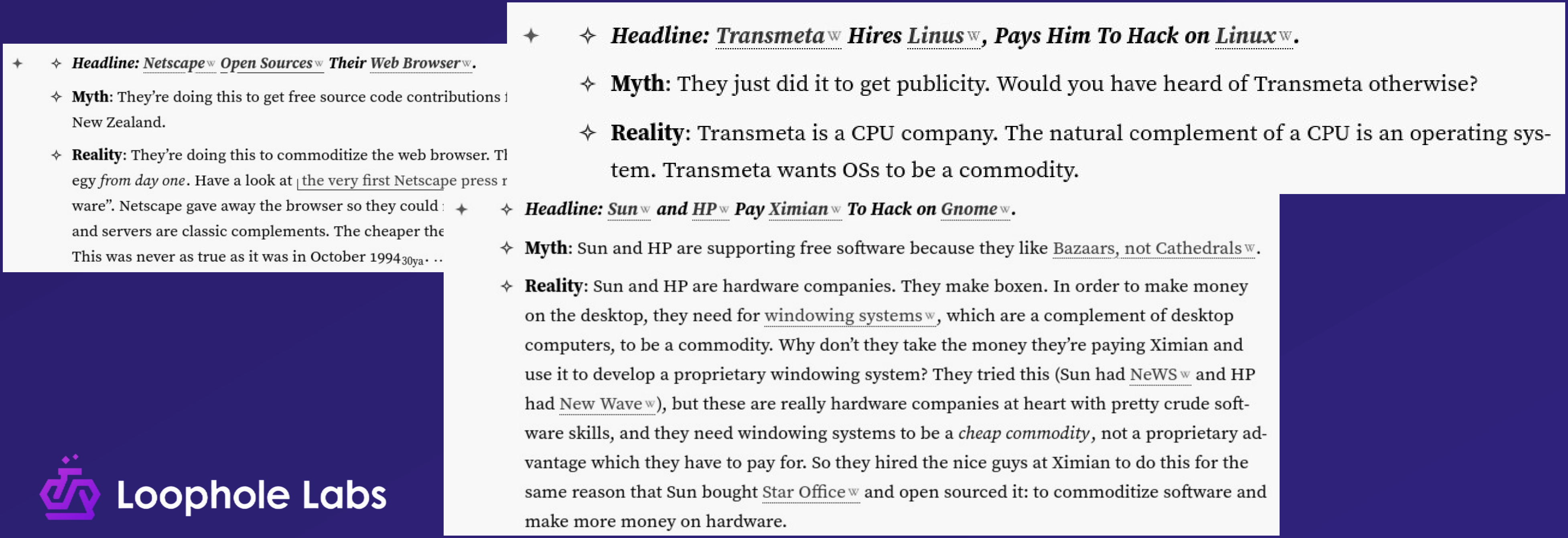
Loophole Labs

https://gwern.net/complement

Cars → Electricity/Gas

Computers → Software

Shipping company → Rail/Road



- ***Headline: Netscape Open Sources Their Web Browser.***
  - **Myth**: They're doing this to get free source code contributions f
    New Zealand.
  - **Reality**: They're doing this to commoditize the web browser. Th
    egy *from day one*. Have a look at the very first Netscape press r
    ware". Netscape gave away the browser so they could
    and servers are classic complements. The cheaper the
    This was never as true as it was in October 1994$_{\text{30ya}}$. ..

- ***Headline: Transmeta Hires Linus, Pays Him To Hack on Linux.***
  - **Myth**: They just did it to get publicity. Would you have heard of Transmeta otherwise?
  - **Reality**: Transmeta is a CPU company. The natural complement of a CPU is an operating system. Transmeta wants OSs to be a commodity.

- ***Headline: Sun and HP Pay Ximian To Hack on Gnome.***
  - **Myth**: Sun and HP are supporting free software because they like Bazaars, not Cathedrals.
  - **Reality**: Sun and HP are hardware companies. They make boxen. In order to make money on the desktop, they need for windowing systems, which are a complement of desktop computers, to be a commodity. Why don't they take the money they're paying Ximian and use it to develop a proprietary windowing system? They tried this (Sun had NeWS and HP had New Wave), but these are really hardware companies at heart with pretty crude software skills, and they need windowing systems to be a *cheap commodity*, not a proprietary advantage which they have to pay for. So they hired the nice guys at Ximian to do this for the same reason that Sun bought Star Office and open sourced it: to commoditize software and make more money on hardware.

Loophole Labs

https://gwern.net/complement

If you produce software, what is your complement?

Loophole Labs

aws
Loophole Labs
HETZNER

How would commoditized compute look like?

Loophole Labs

Can run **everything**

Can run **everywhere**

Can it run **everything**?

VMs

Loophole Labs

QEMU

Cloud
Hypervisor

KVM

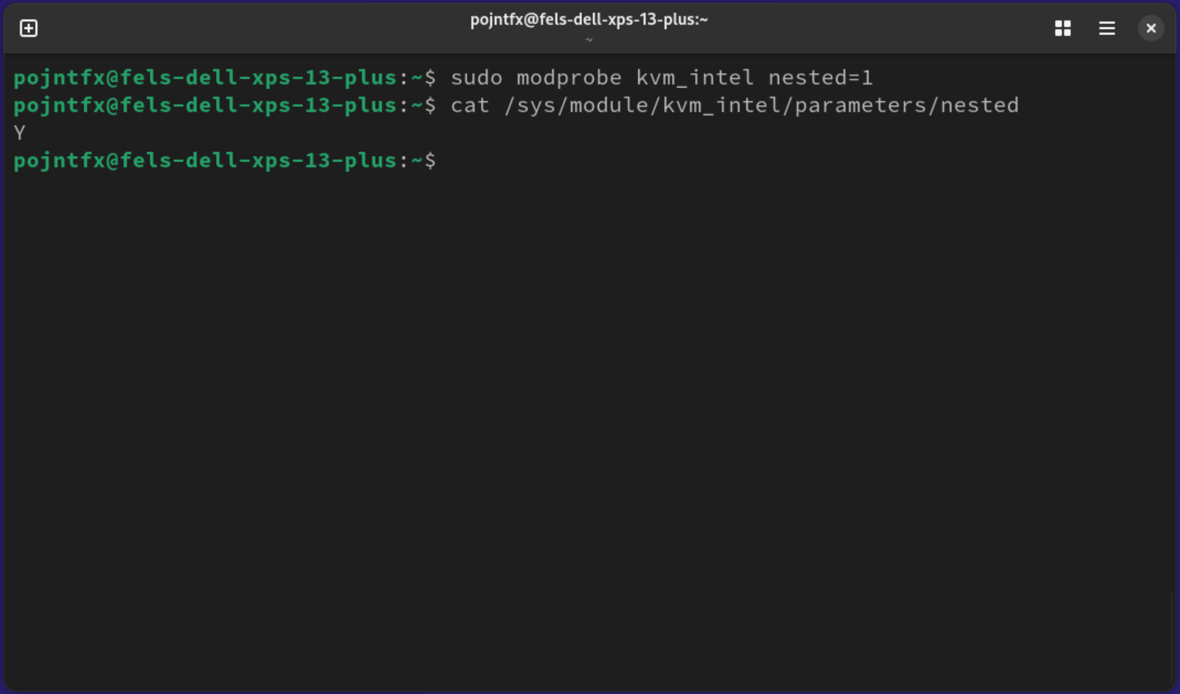
Loophole Labs

Firecracker

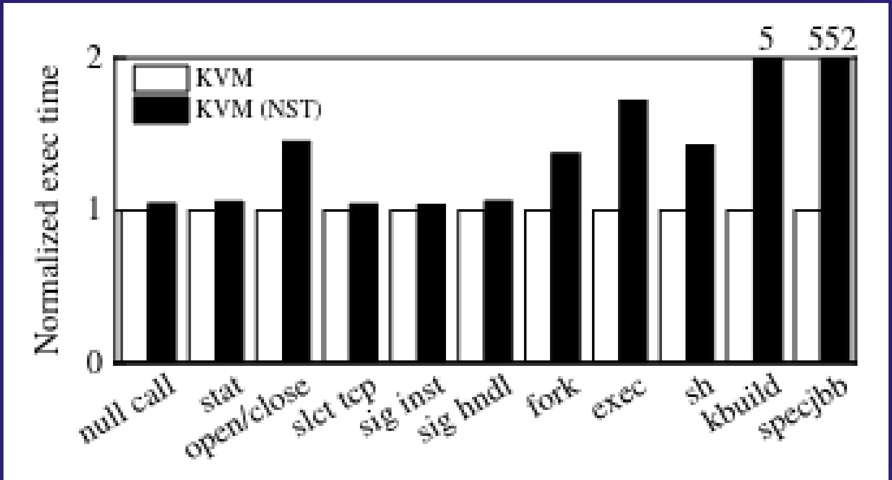
VT-d (kvm_intel)

AMD-V (kvm_amd)

aws
Loophole Labs
HETZNER



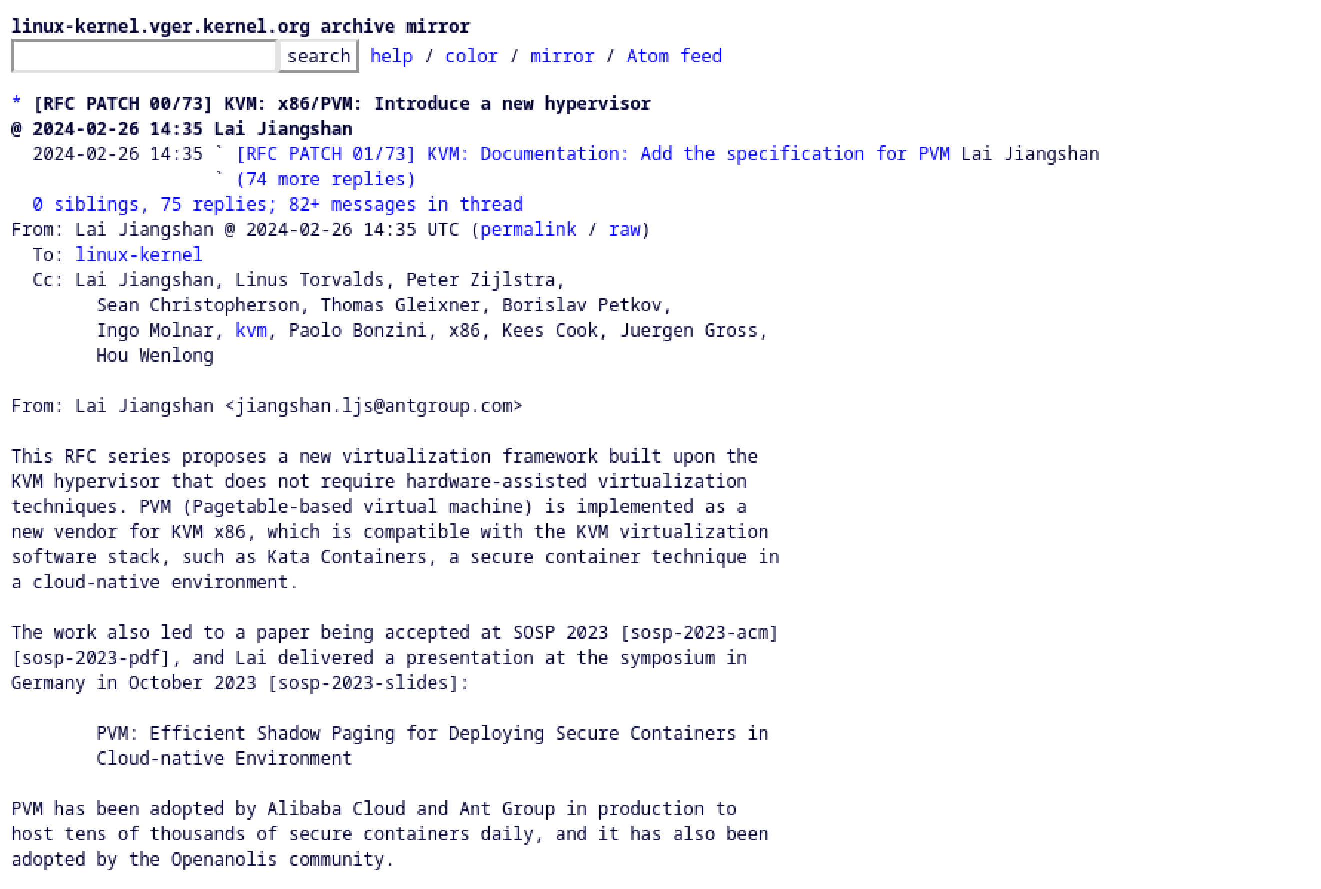
```
pojntfx@fels-dell-xps-13-plus:~$ sudo modprobe kvm_intel nested=1
pojntfx@fels-dell-xps-13-plus:~$ cat /sys/module/kvm_intel/parameters/nested
Y
pojntfx@fels-dell-xps-13-plus:~$
```



https://huilucs.github.io/pubs/pvm.pdf

Component 1: **PVM**

Loophole Labs

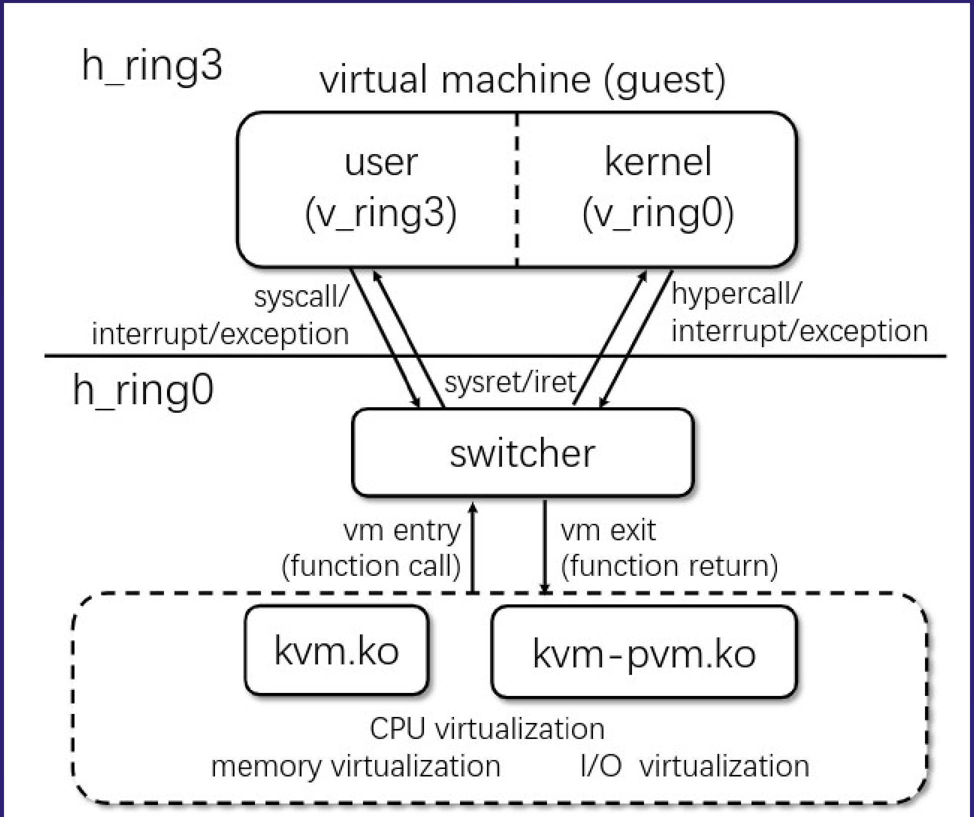


linux-kernel.vger.kernel.org archive mirror

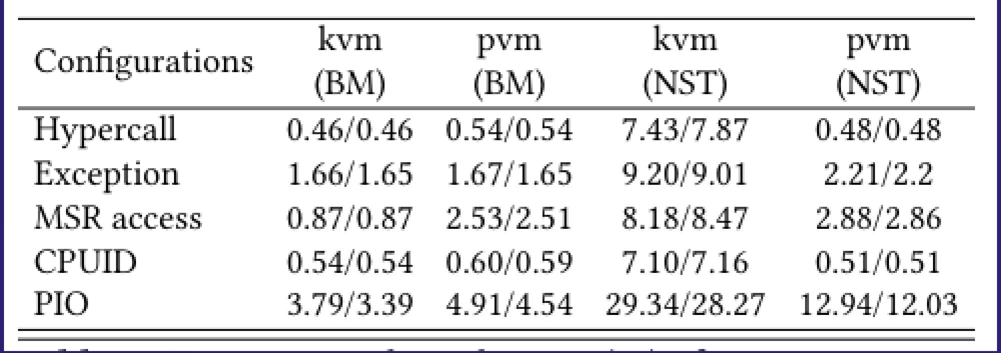
search help / color / mirror / Atom feed

```
* [RFC PATCH 00/73] KVM: x86/PVM: Introduce a new hypervisor
@ 2024-02-26 14:35 Lai Jiangshan
  2024-02-26 14:35 ` [RFC PATCH 01/73] KVM: Documentation: Add the specification for PVM Lai Jiangshan
                   ` (74 more replies)
  0 siblings, 75 replies; 82+ messages in thread
From: Lai Jiangshan @ 2024-02-26 14:35 UTC (permalink / raw)
  To: linux-kernel
  Cc: Lai Jiangshan, Linus Torvalds, Peter Zijlstra,
        Sean Christopherson, Thomas Gleixner, Borislav Petkov,
        Ingo Molnar, kvm, Paolo Bonzini, x86, Kees Cook, Juergen Gross,
        Hou Wenlong

From: Lai Jiangshan <jiangshan.ljs@antgroup.com>

This RFC series proposes a new virtualization framework built upon the
KVM hypervisor that does not require hardware-assisted virtualization
techniques. PVM (Pagetable-based virtual machine) is implemented as a
new vendor for KVM x86, which is compatible with the KVM virtualization
software stack, such as Kata Containers, a secure container technique in
a cloud-native environment.

The work also led to a paper being accepted at SOSP 2023 [sosp-2023-acm]
[sosp-2023-pdf], and Lai delivered a presentation at the symposium in
Germany in October 2023 [sosp-2023-slides]:

        PVM: Efficient Shadow Paging for Deploying Secure Containers in
        Cloud-native Environment

PVM has been adopted by Alibaba Cloud and Ant Group in production to
host tens of thousands of secure containers daily, and it has also been
adopted by the Openanolis community.
```

https://lore.kernel.org/lkml/CABgObfaSGOt4AKRF5WEJt2fGMj_hLXd7J2×2etce2ymvT4HkpA@mail.gmail.com/T/

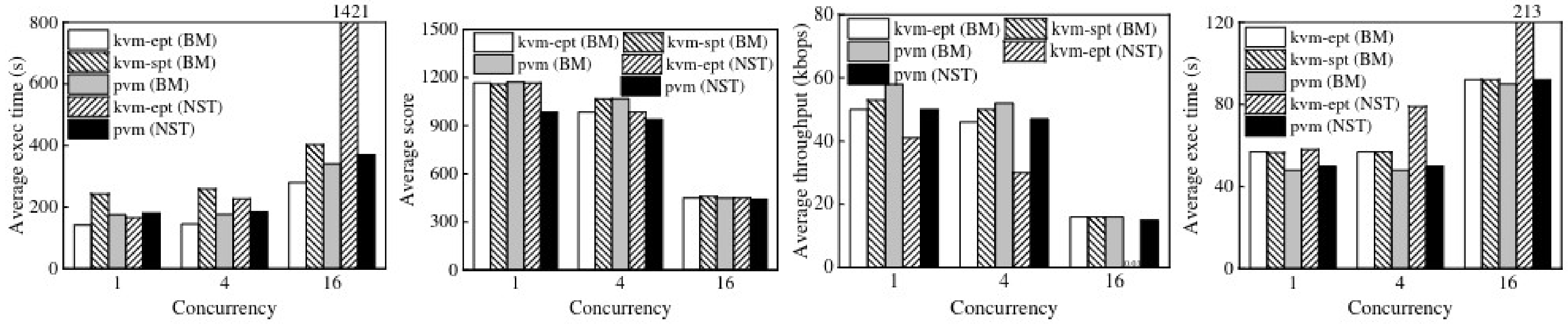


https://huilucs.github.io/pubs/pvm.pdf



Configurations	kvm (BM)	pvm (BM)	kvm (NST)	pvm (NST)
Hypercall	0.46/0.46	0.54/0.54	7.43/7.87	0.48/0.48
Exception	1.66/1.65	1.67/1.65	9.20/9.01	2.21/2.2
MSR access	0.87/0.87	2.53/2.51	8.18/8.47	2.88/2.86
CPUID	0.54/0.54	0.60/0.59	7.10/7.16	0.51/0.51
PIO	3.79/3.39	4.91/4.54	29.34/28.27	12.94/12.03

https://huilucs.github.io/pubs/pvm.pdf



(a) Kbuild (lower is better)

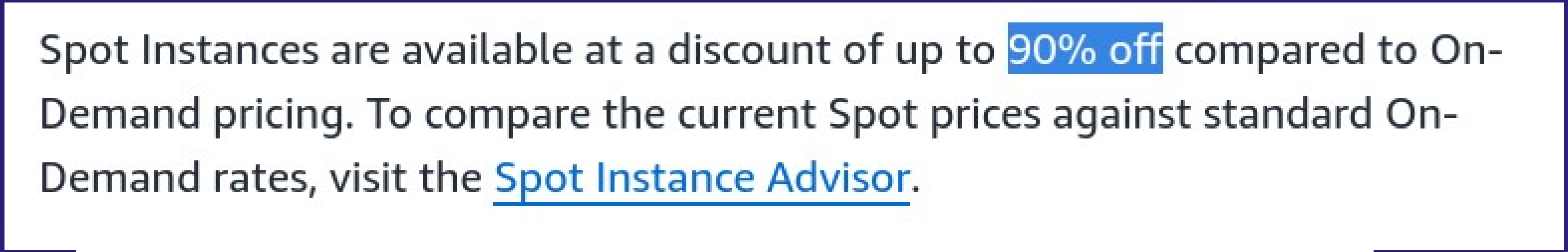
(b) Blogbench (higher is better)

(c) Specjbb2005 (higher is better)

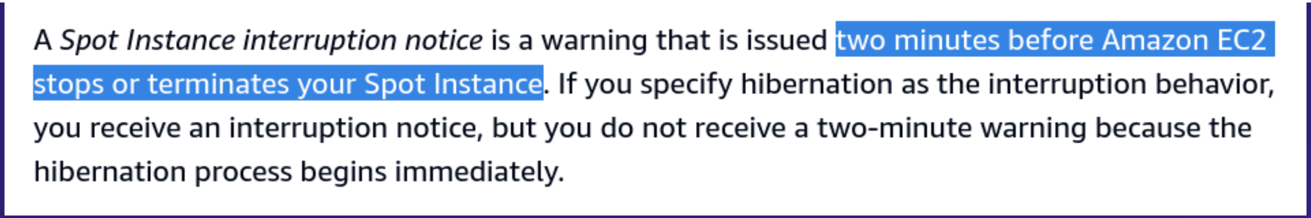
(d) Fluidanimate (lower is better)

https://huilucs.github.io/pubs/pvm.pdf

Can it run **everywhere**?



Spot Instances are available at a discount of up to 90% off compared to On-Demand pricing. To compare the current Spot prices against standard On-Demand rates, visit the Spot Instance Advisor.



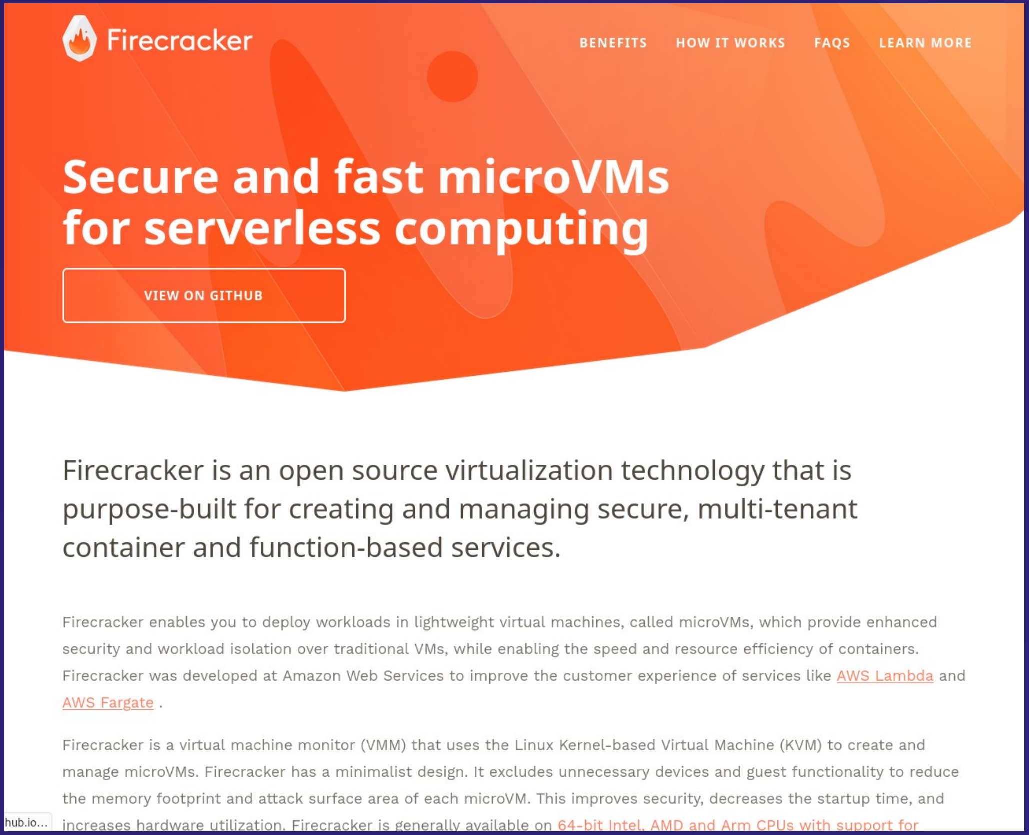
A *Spot Instance interruption notice* is a warning that is issued two minutes before Amazon EC2 stops or terminates your Spot Instance. If you specify hibernation as the interruption behavior, you receive an interruption notice, but you do not receive a two-minute warning because the hibernation process begins immediately.

Live Migration

Loophole Labs

Component 2: **Firecracker**

Loophole Labs



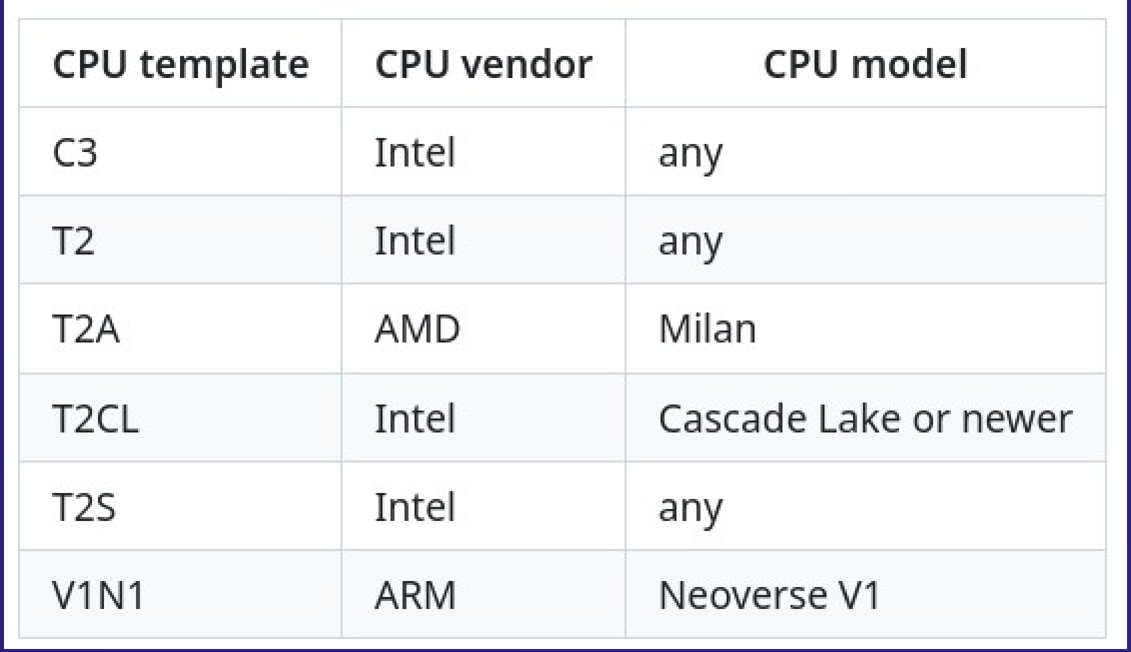
Secure and fast microVMs for serverless computing

Firecracker is an open source virtualization technology that is purpose-built for creating and managing secure, multi-tenant container and function-based services.

Firecracker enables you to deploy workloads in lightweight virtual machines, called microVMs, which provide enhanced security and workload isolation over traditional VMs, while enabling the speed and resource efficiency of containers. Firecracker was developed at Amazon Web Services to improve the customer experience of services like AWS Lambda and AWS Fargate .

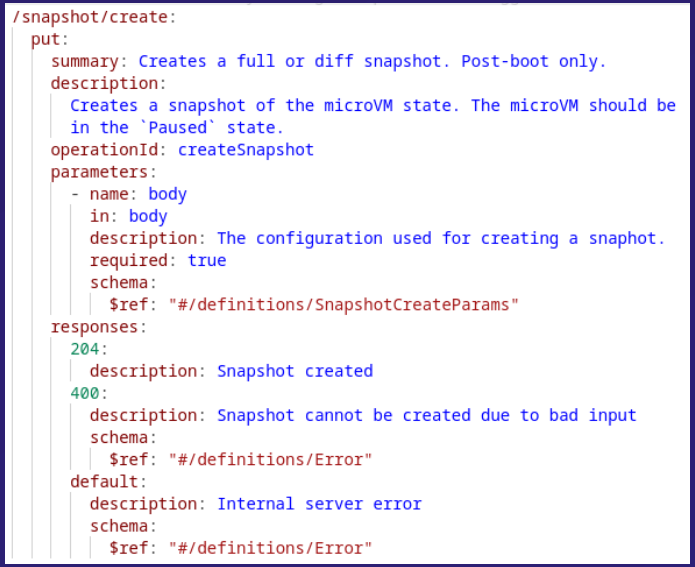
Firecracker is a virtual machine monitor (VMM) that uses the Linux Kernel-based Virtual Machine (KVM) to create and manage microVMs. Firecracker has a minimalist design. It excludes unnecessary devices and guest functionality to reduce the memory footprint and attack surface area of each microVM. This improves security, decreases the startup time, and increases hardware utilization. Firecracker is generally available on 64-bit Intel, AMD and Arm CPUs with support for

https://firecracker-microvm.github.io/

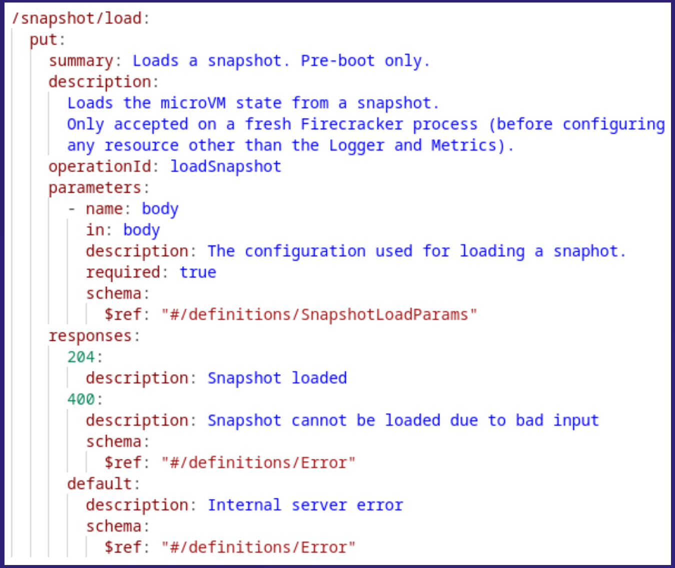


CPU template	CPU vendor	CPU model
C3	Intel	any
T2	Intel	any
T2A	AMD	Milan
T2CL	Intel	Cascade Lake or newer
T2S	Intel	any
V1N1	ARM	Neoverse V1

https://github.com/firecracker-microvm/firecracker/blob/main/docs/cpu_templates/cpu-templates.md

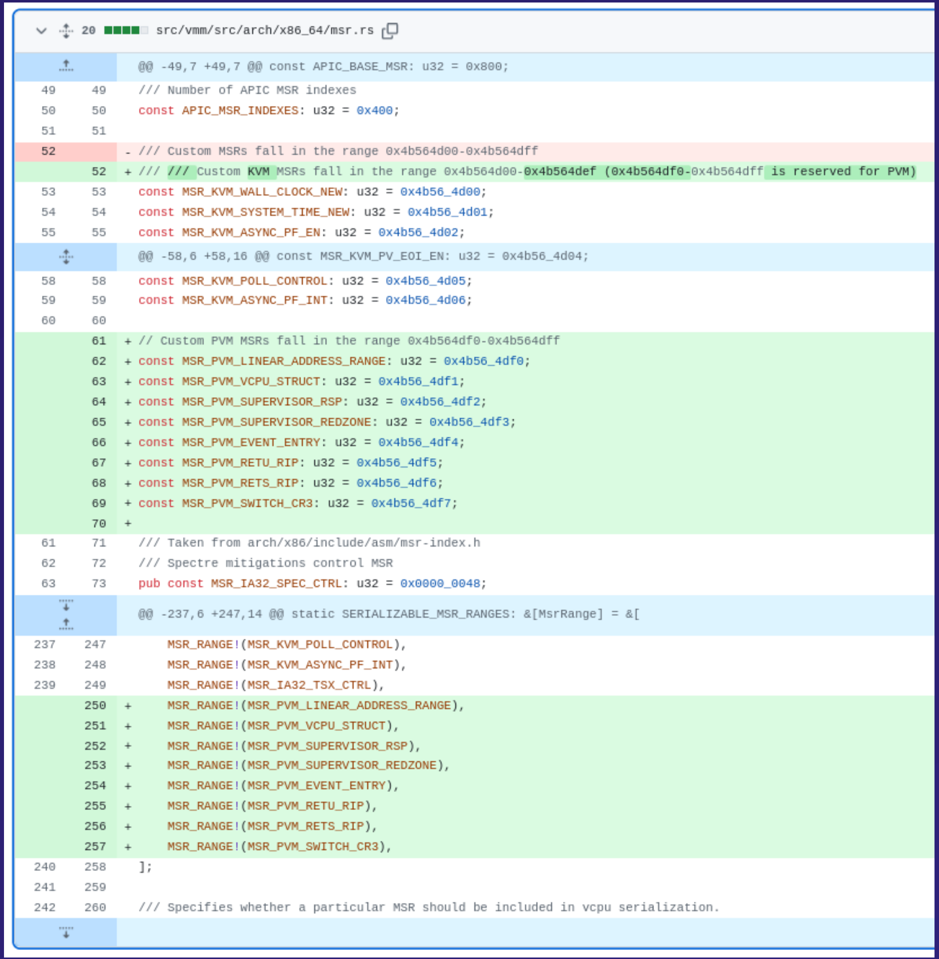


```
/snapshot/create:
  put:
    summary: Creates a full or diff snapshot. Post-boot only.
    description:
      Creates a snapshot of the microVM state. The microVM should be
      in the `Paused` state.
    operationId: createSnapshot
    parameters:
      - name: body
        in: body
        description: The configuration used for creating a snaphot.
        required: true
        schema:
          $ref: "#/definitions/SnapshotCreateParams"
    responses:
      204:
        description: Snapshot created
      400:
        description: Snapshot cannot be created due to bad input
        schema:
          $ref: "#/definitions/Error"
      default:
        description: Internal server error
        schema:
          $ref: "#/definitions/Error"
```



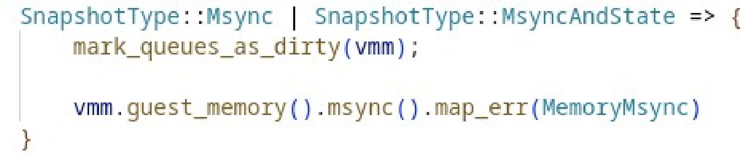
```
/snapshot/load:
  put:
    summary: Loads a snapshot. Pre-boot only.
    description:
      Loads the microVM state from a snapshot.
      Only accepted on a fresh Firecracker process (before configuring
      any resource other than the Logger and Metrics).
    operationId: loadSnapshot
    parameters:
      - name: body
        in: body
        description: The configuration used for loading a snaphot.
        required: true
        schema:
          $ref: "#/definitions/SnapshotLoadParams"
    responses:
      204:
        description: Snapshot loaded
      400:
        description: Snapshot cannot be loaded due to bad input
        schema:
          $ref: "#/definitions/Error"
      default:
        description: Internal server error
        schema:
          $ref: "#/definitions/Error"
```

src/firecracker/swagger/firecracker.yaml

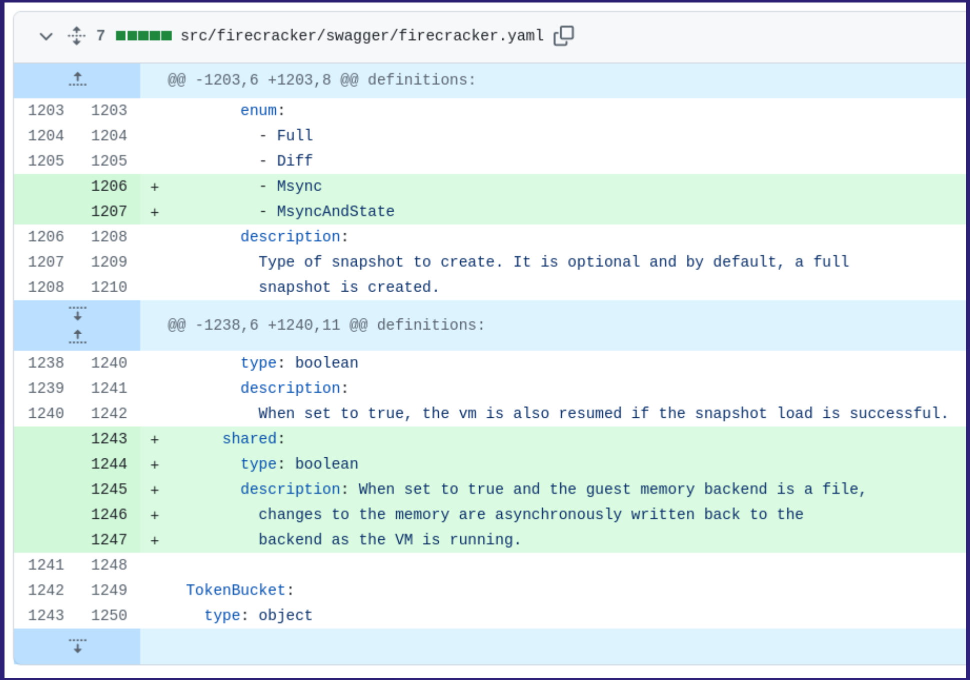


```
20 src/vmm/src/arch/x86_64/msr.rs
@@ -49,7 +49,7 @@ const APIC_BASE_MSR: u32 = 0x800;
  /// Number of APIC MSR indexes
  const APIC_MSR_INDEXES: u32 = 0x400;

- /// Custom MSRs fall in the range 0x4b564d00-0x4b564dff
+ /// /// Custom KVM MSRs fall in the range 0x4b564d00-0x4b564def (0x4b564df0-0x4b564dff is reserved for PVM)
  const MSR_KVM_WALL_CLOCK_NEW: u32 = 0x4b56_4d00;
  const MSR_KVM_SYSTEM_TIME_NEW: u32 = 0x4b56_4d01;
  const MSR_KVM_ASYNC_PF_EN: u32 = 0x4b56_4d02;
@@ -58,6 +58,16 @@ const MSR_KVM_PV_EOI_EN: u32 = 0x4b56_4d04;
  const MSR_KVM_POLL_CONTROL: u32 = 0x4b56_4d05;
  const MSR_KVM_ASYNC_PF_INT: u32 = 0x4b56_4d06;

+ // Custom PVM MSRs fall in the range 0x4b564df0-0x4b564dff
+ const MSR_PVM_LINEAR_ADDRESS_RANGE: u32 = 0x4b56_4df0;
+ const MSR_PVM_VCPU_STRUCT: u32 = 0x4b56_4df1;
+ const MSR_PVM_SUPERVISOR_RSP: u32 = 0x4b56_4df2;
+ const MSR_PVM_SUPERVISOR_REDZONE: u32 = 0x4b56_4df3;
+ const MSR_PVM_EVENT_ENTRY: u32 = 0x4b56_4df4;
+ const MSR_PVM_RETU_RIP: u32 = 0x4b56_4df5;
+ const MSR_PVM_RETS_RIP: u32 = 0x4b56_4df6;
+ const MSR_PVM_SWITCH_CR3: u32 = 0x4b56_4df7;
+
  /// Taken from arch/x86/include/asm/msr-index.h
  /// Spectre mitigations control MSR
  pub const MSR_IA32_SPEC_CTRL: u32 = 0x0000_0048;
@@ -237,6 +247,14 @@ static SERIALIZABLE_MSR_RANGES: &[MsrRange] = &[
      MSR_RANGE!(MSR_KVM_POLL_CONTROL),
      MSR_RANGE!(MSR_KVM_ASYNC_PF_INT),
      MSR_RANGE!(MSR_IA32_TSX_CTRL),
+     MSR_RANGE!(MSR_PVM_LINEAR_ADDRESS_RANGE),
+     MSR_RANGE!(MSR_PVM_VCPU_STRUCT),
+     MSR_RANGE!(MSR_PVM_SUPERVISOR_RSP),
+     MSR_RANGE!(MSR_PVM_SUPERVISOR_REDZONE),
+     MSR_RANGE!(MSR_PVM_EVENT_ENTRY),
+     MSR_RANGE!(MSR_PVM_RETU_RIP),
+     MSR_RANGE!(MSR_PVM_RETS_RIP),
+     MSR_RANGE!(MSR_PVM_SWITCH_CR3),
  ];

  /// Specifies whether a particular MSR should be included in vcpu serialization.
```

https://github.com/loopholelabs/firecracker/pull/15/files#diff-646931758f35a261e2f848a0970552a6da048816eae0393b597d5830d857fba1



```
SnapshotType::Msync | SnapshotType::MsyncAndState => {
    mark_queues_as_dirty(vmm);

    vmm.guest_memory().msync().map_err(MemoryMsync)
}
```



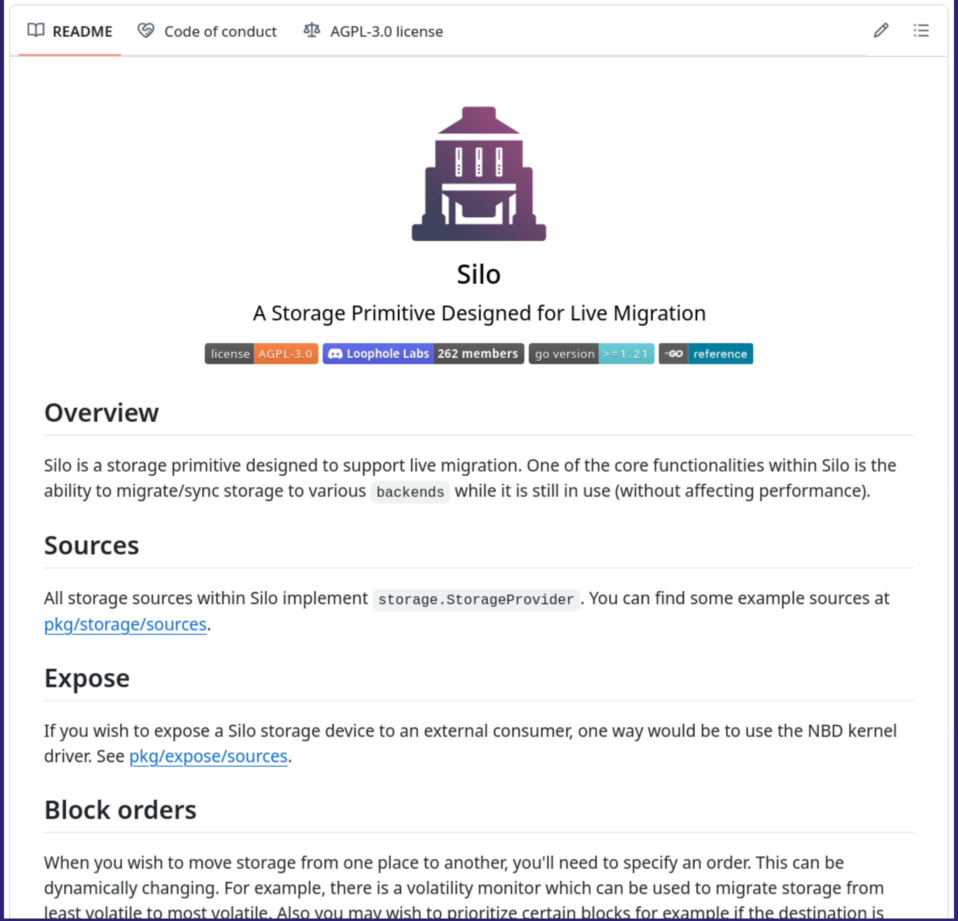
7 src/firecracker/swagger/firecracker.yaml

```
@@ -1203,6 +1203,8 @@ definitions:
          enum:
            - Full
            - Diff
+           - Msync
+           - MsyncAndState
          description:
            Type of snapshot to create. It is optional and by default, a full
            snapshot is created.
@@ -1238,6 +1240,11 @@ definitions:
          type: boolean
          description:
            When set to true, the vm is also resumed if the snapshot load is successful.
+       shared:
+         type: boolean
+         description: When set to true and the guest memory backend is a file,
+           changes to the memory are asynchronously written back to the
+           backend as the VM is running.

    TokenBucket:
      type: object
```

https://github.com/loopholelabs/firecracker/pull/15/
files#diff-646931758f35a261e2f848a0970552a6da048816eae0393b597d5830d857fba1

Component 3: **Silo**

Loophole Labs



Silo

A Storage Primitive Designed for Live Migration

Overview

Silo is a storage primitive designed to support live migration. One of the core functionalities within Silo is the ability to migrate/sync storage to various `backends` while it is still in use (without affecting performance).

Sources

All storage sources within Silo implement `storage.StorageProvider`. You can find some example sources at pkg/storage/sources.

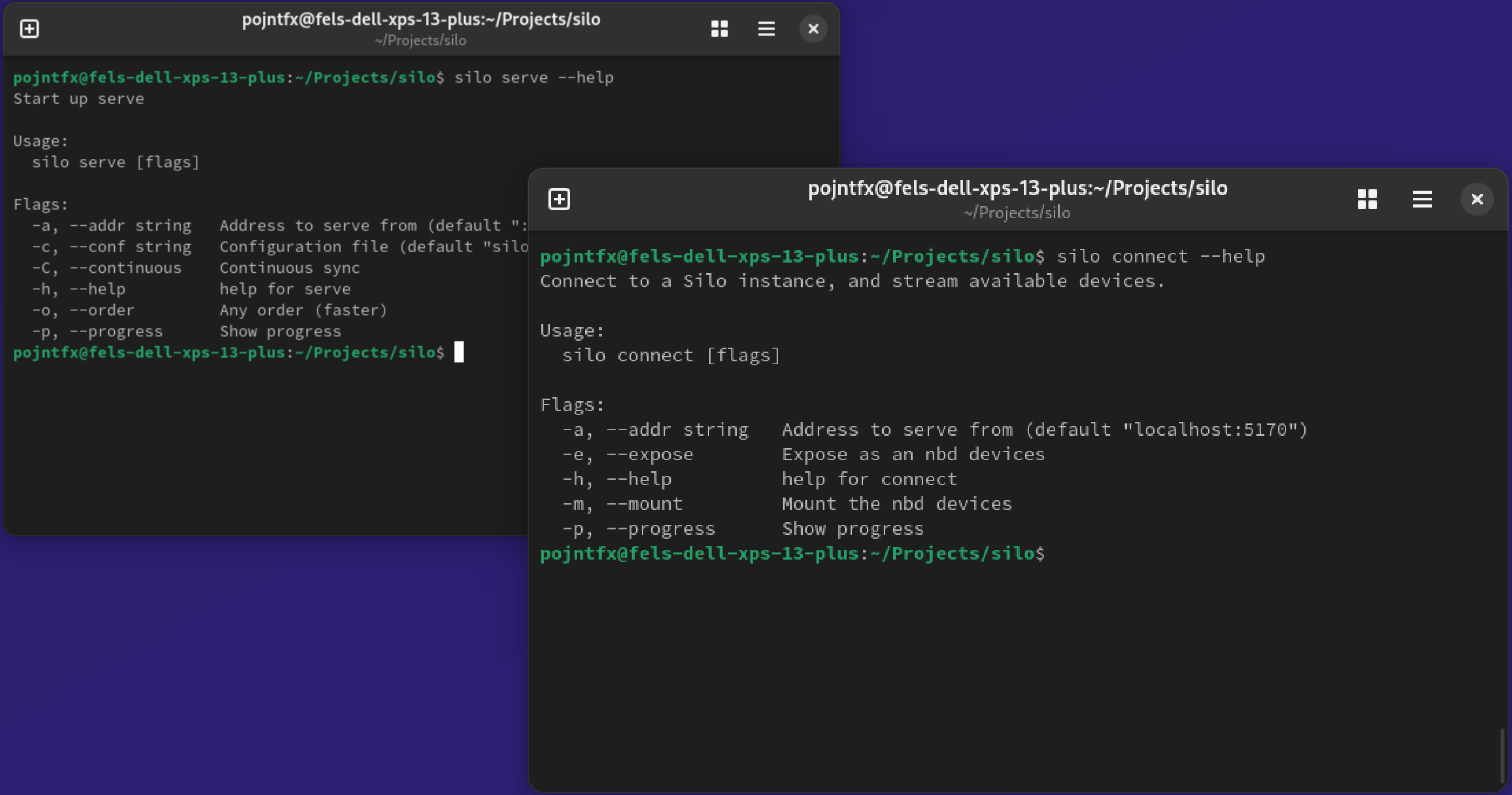
Expose

If you wish to expose a Silo storage device to an external consumer, one way would be to use the NBD kernel driver. See pkg/expose/sources.

Block orders

When you wish to move storage from one place to another, you'll need to specify an order. This can be dynamically changing. For example, there is a volatility monitor which can be used to migrate storage from least volatile to most volatile. Also you may wish to prioritize certain blocks for example if the destination is

https://github.com/loopholelabs/silo

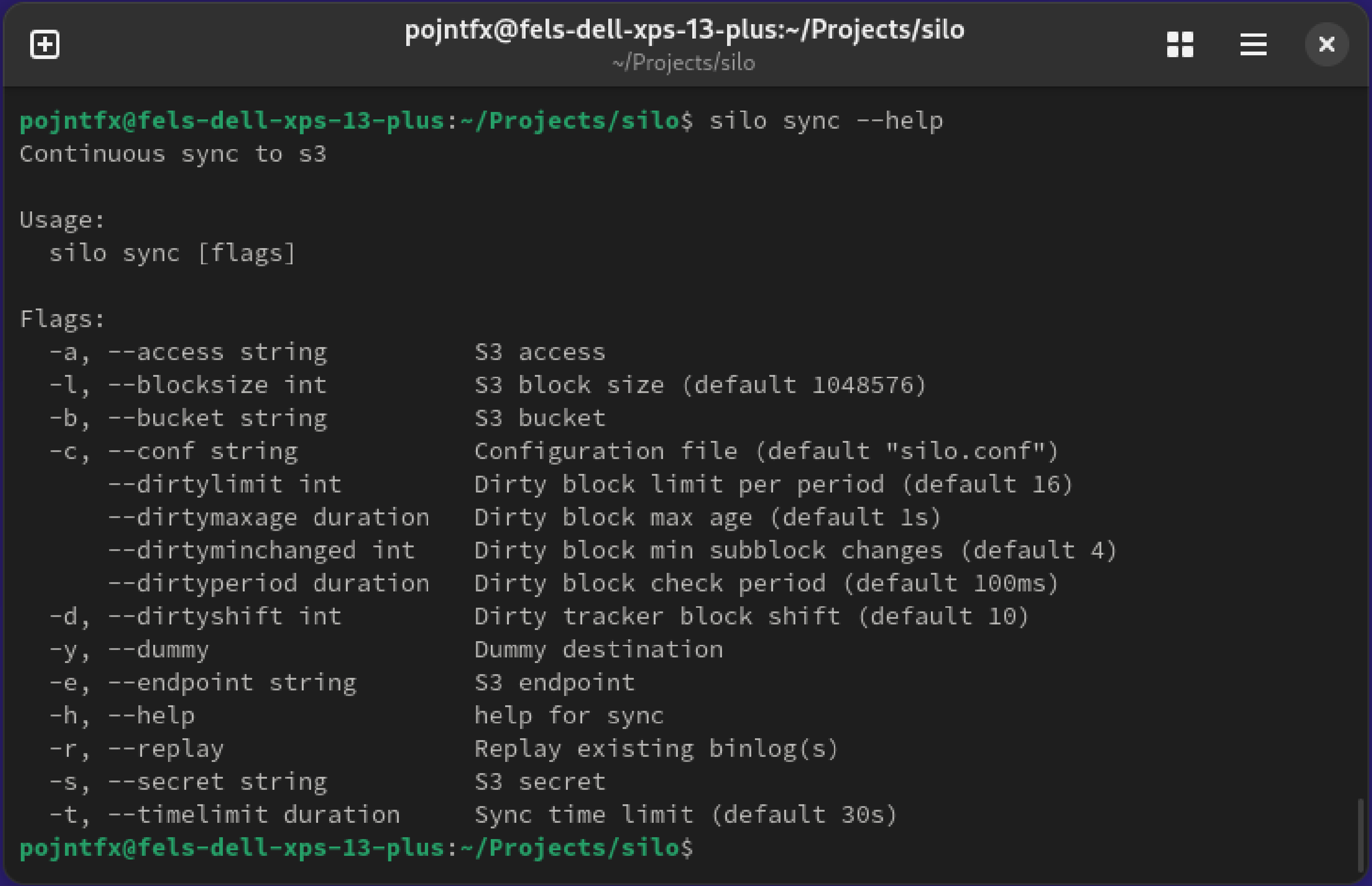


pojntfx@fels-dell-xps-13-plus:~/Projects/silo
~/Projects/silo

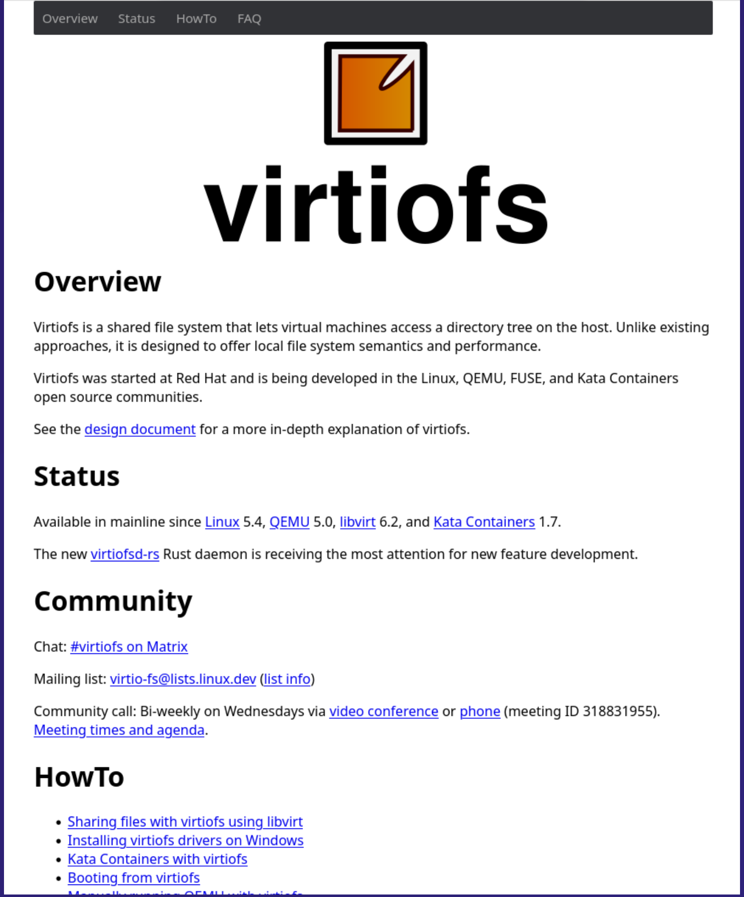
```
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$ silo serve --help
Start up serve

Usage:
  silo serve [flags]

Flags:
  -a, --addr string   Address to serve from (default ":
  -c, --conf string   Configuration file (default "silo
  -C, --continuous    Continuous sync
  -h, --help          help for serve
  -o, --order         Any order (faster)
  -p, --progress      Show progress
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$
```

pojntfx@fels-dell-xps-13-plus:~/Projects/silo
~/Projects/silo

```
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$ silo connect --help
Connect to a Silo instance, and stream available devices.

Usage:
  silo connect [flags]

Flags:
  -a, --addr string   Address to serve from (default "localhost:5170")
  -e, --expose        Expose as an nbd devices
  -h, --help          help for connect
  -m, --mount         Mount the nbd devices
  -p, --progress      Show progress
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$
```



```
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$ silo sync --help
Continuous sync to s3

Usage:
  silo sync [flags]

Flags:
  -a, --access string          S3 access
  -l, --blocksize int          S3 block size (default 1048576)
  -b, --bucket string          S3 bucket
  -c, --conf string            Configuration file (default "silo.conf")
      --dirtylimit int         Dirty block limit per period (default 16)
      --dirtymaxage duration   Dirty block max age (default 1s)
      --dirtyminchanged int    Dirty block min subblock changes (default 4)
      --dirtyperiod duration   Dirty block check period (default 100ms)
  -d, --dirtyshift int         Dirty tracker block shift (default 10)
  -y, --dummy                  Dummy destination
  -e, --endpoint string        S3 endpoint
  -h, --help                   help for sync
  -r, --replay                 Replay existing binlog(s)
  -s, --secret string          S3 secret
  -t, --timelimit duration     Sync time limit (default 30s)
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$
```



Overview

Virtiofs is a shared file system that lets virtual machines access a directory tree on the host. Unlike existing approaches, it is designed to offer local file system semantics and performance.

Virtiofs was started at Red Hat and is being developed in the Linux, QEMU, FUSE, and Kata Containers open source communities.

See the design document for a more in-depth explanation of virtiofs.

Status

Available in mainline since Linux 5.4, QEMU 5.0, libvirt 6.2, and Kata Containers 1.7.

The new virtiofsd-rs Rust daemon is receiving the most attention for new feature development.

Community

Chat: #virtiofs on Matrix

Mailing list: virtio-fs@lists.linux.dev (list info)

Community call: Bi-weekly on Wednesdays via video conference or phone (meeting ID 318831955). Meeting times and agenda.

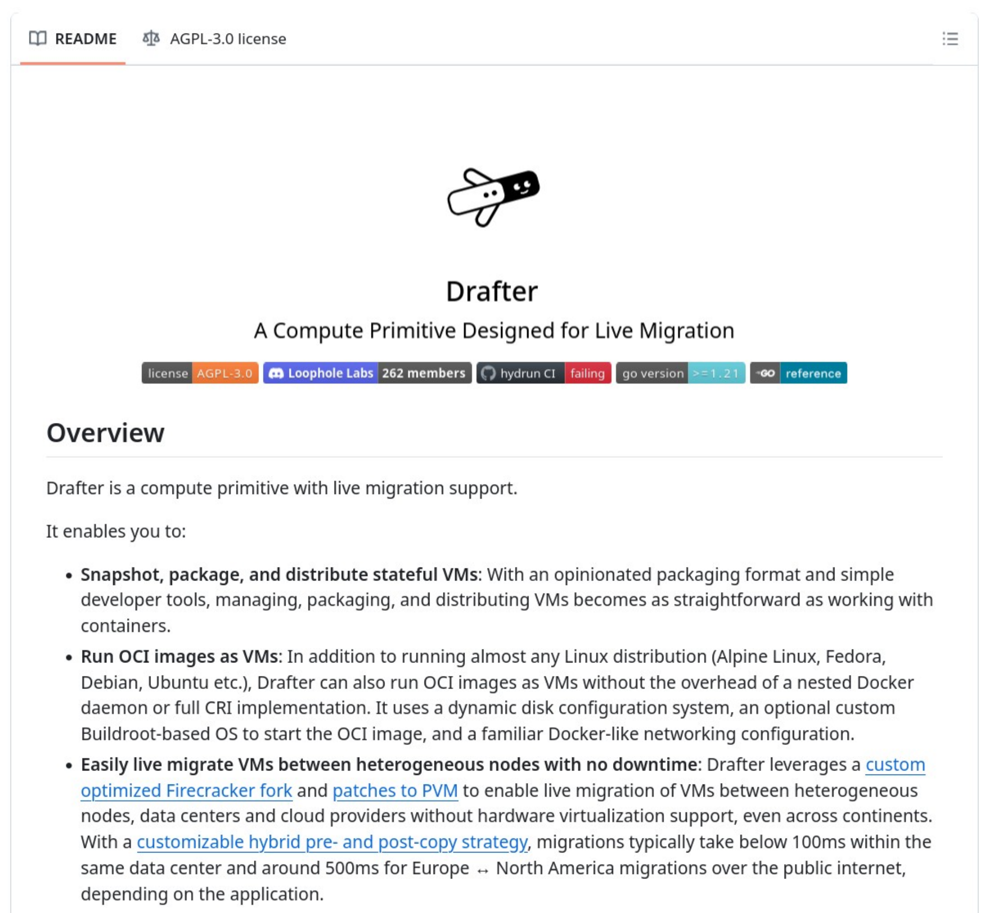
HowTo

- Sharing files with virtiofs using libvirt
- Installing virtiofs drivers on Windows
- Kata Containers with virtiofs
- Booting from virtiofs
- Manually running QEMU with virtiofs

https://virtio-fs.gitlab.io/

Component 4: **Drafter**

Loophole Labs



Drafter

A Compute Primitive Designed for Live Migration

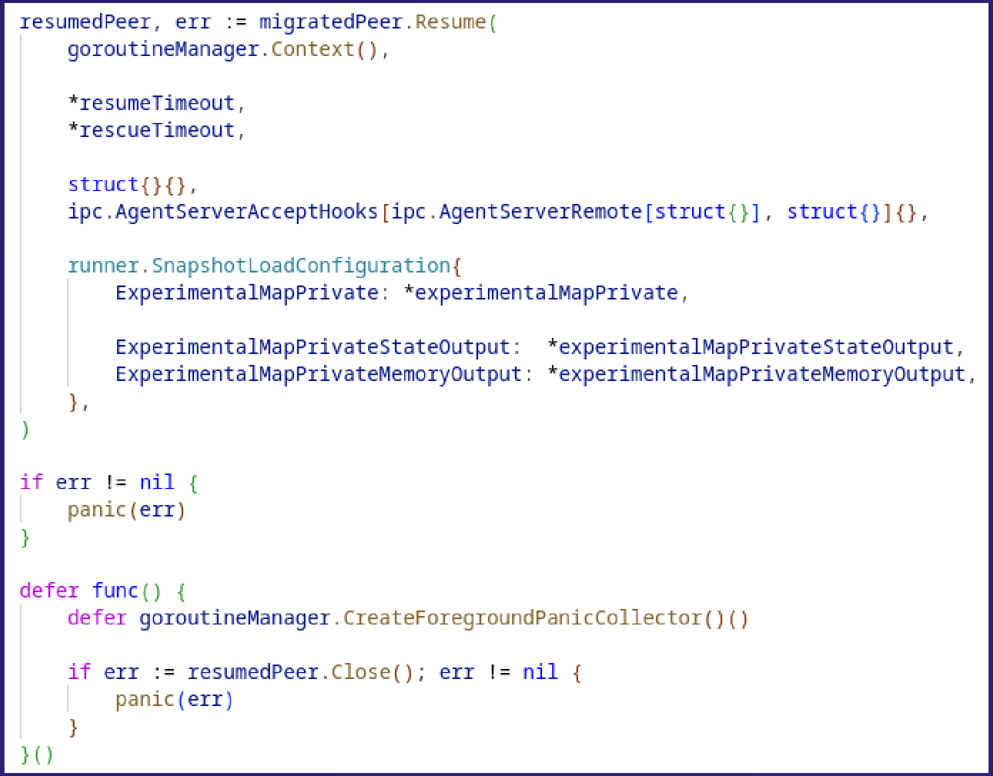
Overview

Drafter is a compute primitive with live migration support.

It enables you to:

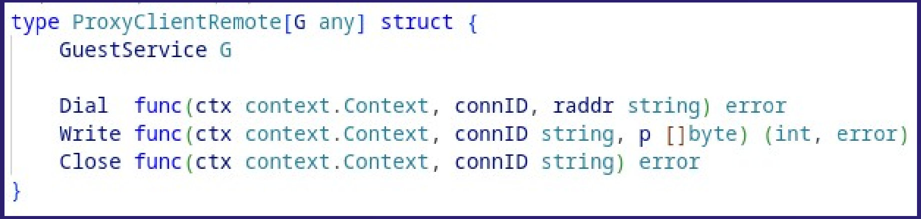
- **Snapshot, package, and distribute stateful VMs**: With an opinionated packaging format and simple developer tools, managing, packaging, and distributing VMs becomes as straightforward as working with containers.
- **Run OCI images as VMs**: In addition to running almost any Linux distribution (Alpine Linux, Fedora, Debian, Ubuntu etc.), Drafter can also run OCI images as VMs without the overhead of a nested Docker daemon or full CRI implementation. It uses a dynamic disk configuration system, an optional custom Buildroot-based OS to start the OCI image, and a familiar Docker-like networking configuration.
- **Easily live migrate VMs between heterogeneous nodes with no downtime**: Drafter leverages a custom optimized Firecracker fork and patches to PVM to enable live migration of VMs between heterogeneous nodes, data centers and cloud providers without hardware virtualization support, even across continents. With a customizable hybrid pre- and post-copy strategy, migrations typically take below 100ms within the same data center and around 500ms for Europe ↔ North America migrations over the public internet, depending on the application.

https://github.com/loopholelabs/drafter

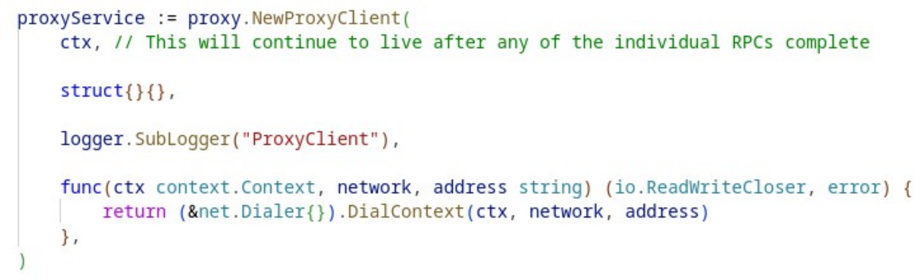


```go
resumedPeer, err := migratedPeer.Resume(
    goroutineManager.Context(),

    *resumeTimeout,
    *rescueTimeout,

    struct{}{},
    ipc.AgentServerAcceptHooks[ipc.AgentServerRemote[struct{}], struct{}]{},

    runner.SnapshotLoadConfiguration{
        ExperimentalMapPrivate: *experimentalMapPrivate,

        ExperimentalMapPrivateStateOutput:  *experimentalMapPrivateStateOutput,
        ExperimentalMapPrivateMemoryOutput: *experimentalMapPrivateMemoryOutput,
    },
)

if err != nil {
    panic(err)
}

defer func() {
    defer goroutineManager.CreateForegroundPanicCollector()()

    if err := resumedPeer.Close(); err != nil {
        panic(err)
    }
}()
```

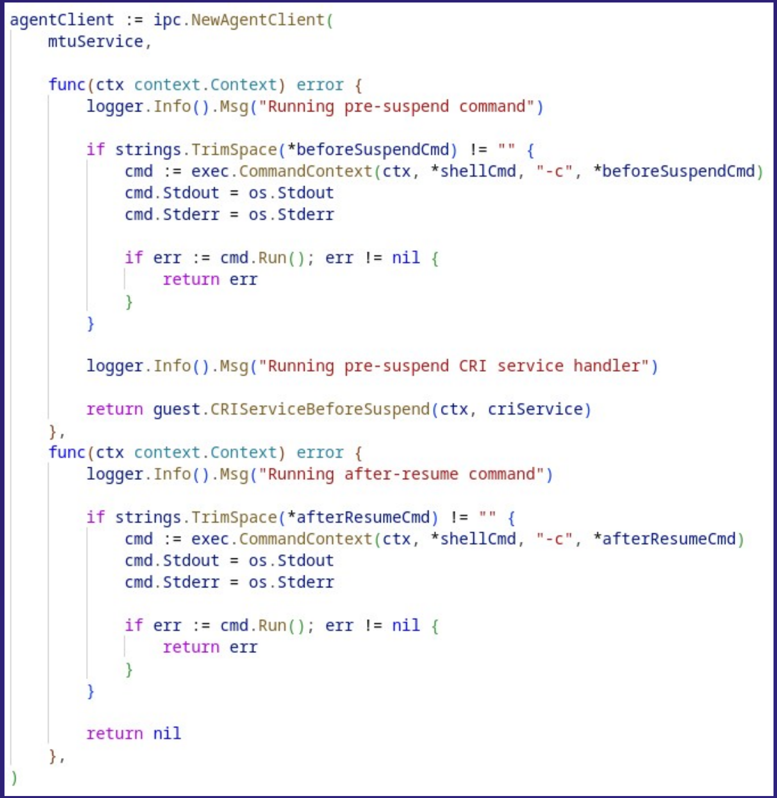
cmd/drafter-peer/main.go



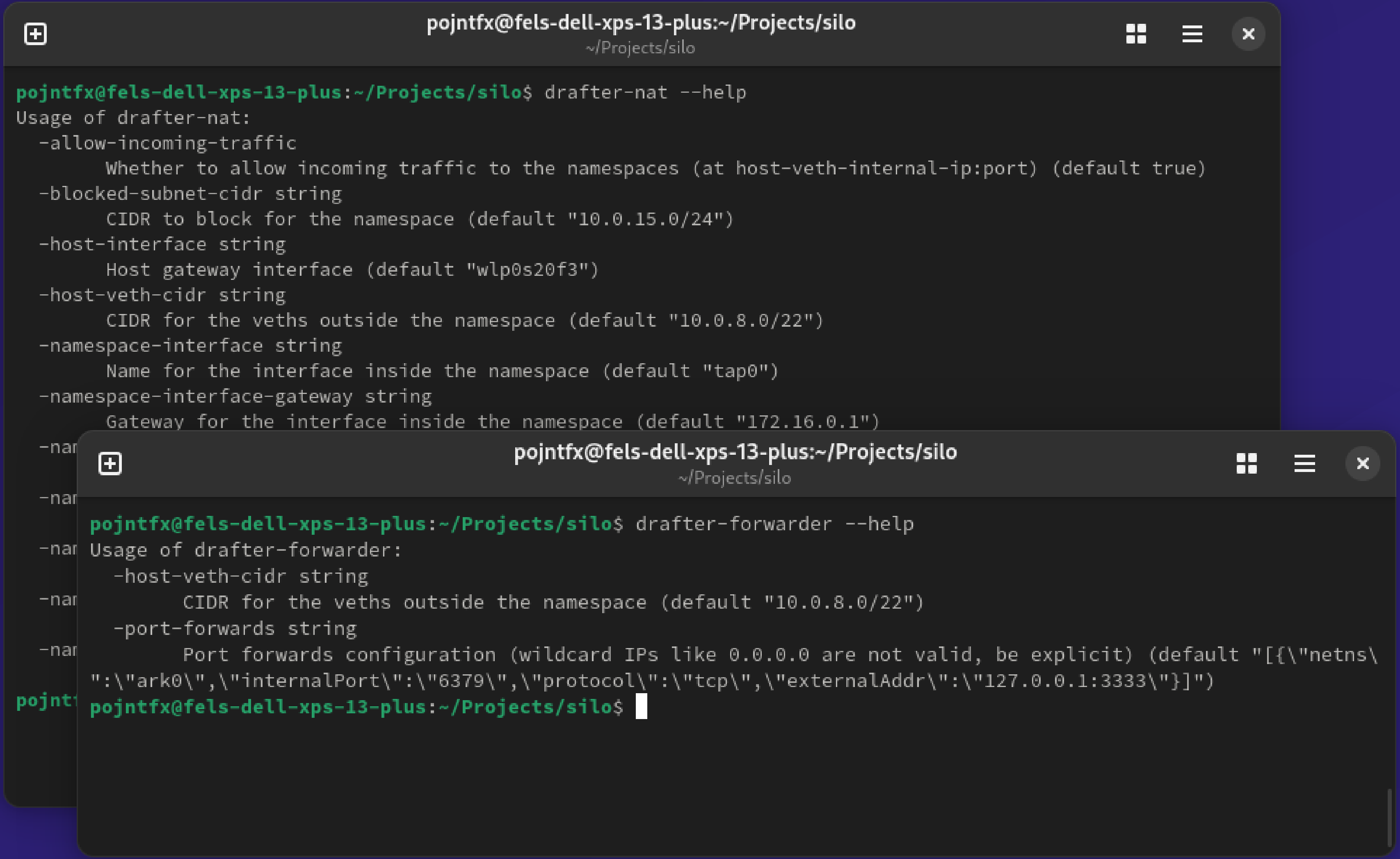
```go
type ProxyClientRemote[G any] struct {
    GuestService G

    Dial  func(ctx context.Context, connID, raddr string) error
    Write func(ctx context.Context, connID string, p []byte) (int, error)
    Close func(ctx context.Context, connID string) error
}
```



```go
proxyService := proxy.NewProxyClient(
    ctx, // This will continue to live after any of the individual RPCs complete

    struct{}{},

    logger.SubLogger("ProxyClient"),

    func(ctx context.Context, network, address string) (io.ReadWriteCloser, error) {
        return (&net.Dialer{}).DialContext(ctx, network, address)
    },
)
```

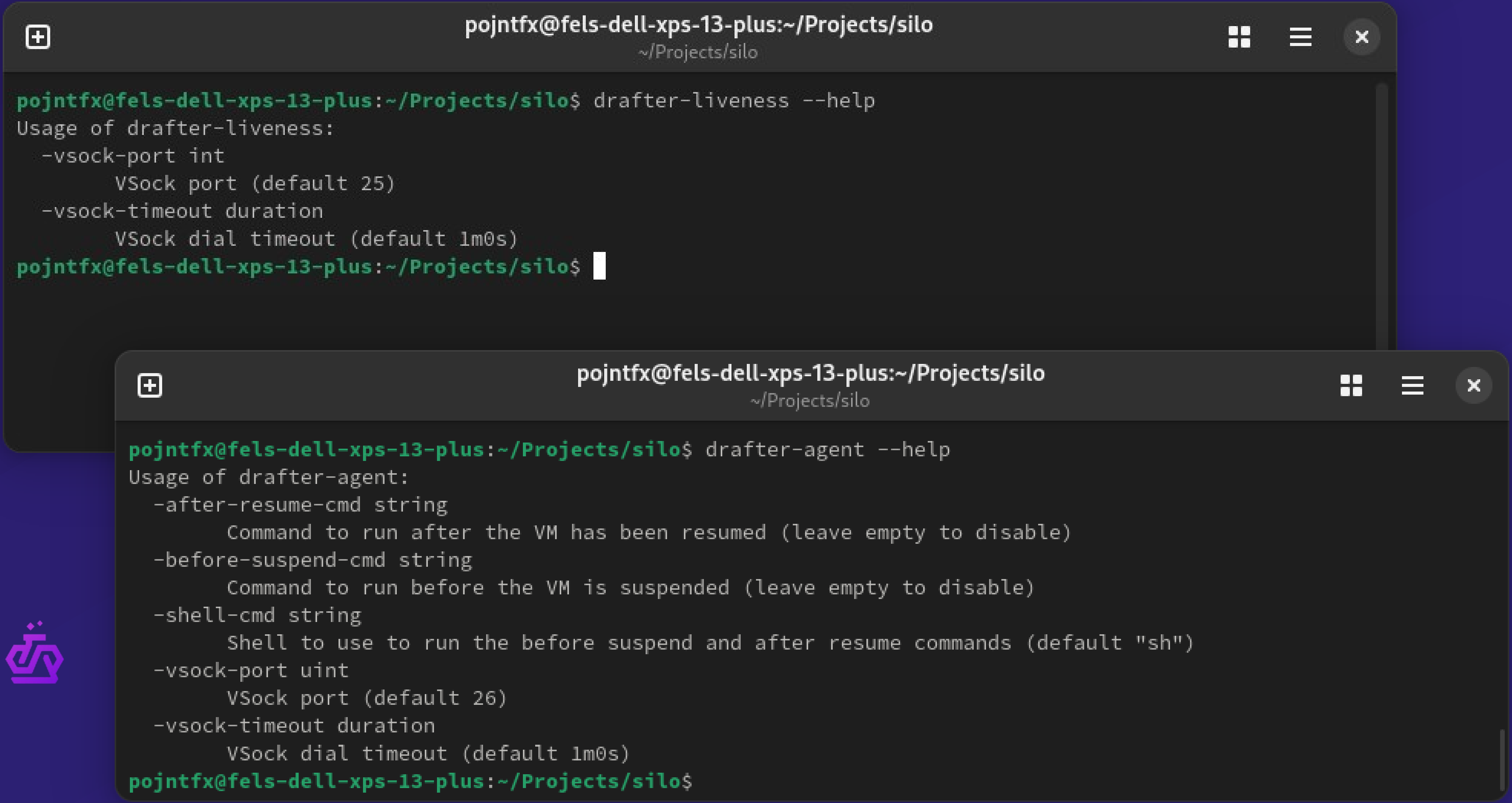


```go
agentClient := ipc.NewAgentClient(
    mtuService,

    func(ctx context.Context) error {
        logger.Info().Msg("Running pre-suspend command")

        if strings.TrimSpace(*beforeSuspendCmd) != "" {
            cmd := exec.CommandContext(ctx, *shellCmd, "-c", *beforeSuspendCmd)
            cmd.Stdout = os.Stdout
            cmd.Stderr = os.Stderr

            if err := cmd.Run(); err != nil {
                return err
            }
        }

        logger.Info().Msg("Running pre-suspend CRI service handler")

        return guest.CRIServiceBeforeSuspend(ctx, criService)
    },
    func(ctx context.Context) error {
        logger.Info().Msg("Running after-resume command")

        if strings.TrimSpace(*afterResumeCmd) != "" {
            cmd := exec.CommandContext(ctx, *shellCmd, "-c", *afterResumeCmd)
            cmd.Stdout = os.Stdout
            cmd.Stderr = os.Stderr

            if err := cmd.Run(); err != nil {
                return err
            }
        }

        return nil
    },
)
```



```
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$ drafter-nat --help
Usage of drafter-nat:
  -allow-incoming-traffic
        Whether to allow incoming traffic to the namespaces (at host-veth-internal-ip:port) (default true)
  -blocked-subnet-cidr string
        CIDR to block for the namespace (default "10.0.15.0/24")
  -host-interface string
        Host gateway interface (default "wlp0s20f3")
  -host-veth-cidr string
        CIDR for the veths outside the namespace (default "10.0.8.0/22")
  -namespace-interface string
        Name for the interface inside the namespace (default "tap0")
  -namespace-interface-gateway string
        Gateway for the interface inside the namespace (default "172.16.0.1")
  -nar

  -nar

  -nar

  -nar

  -nar

pojnt
```

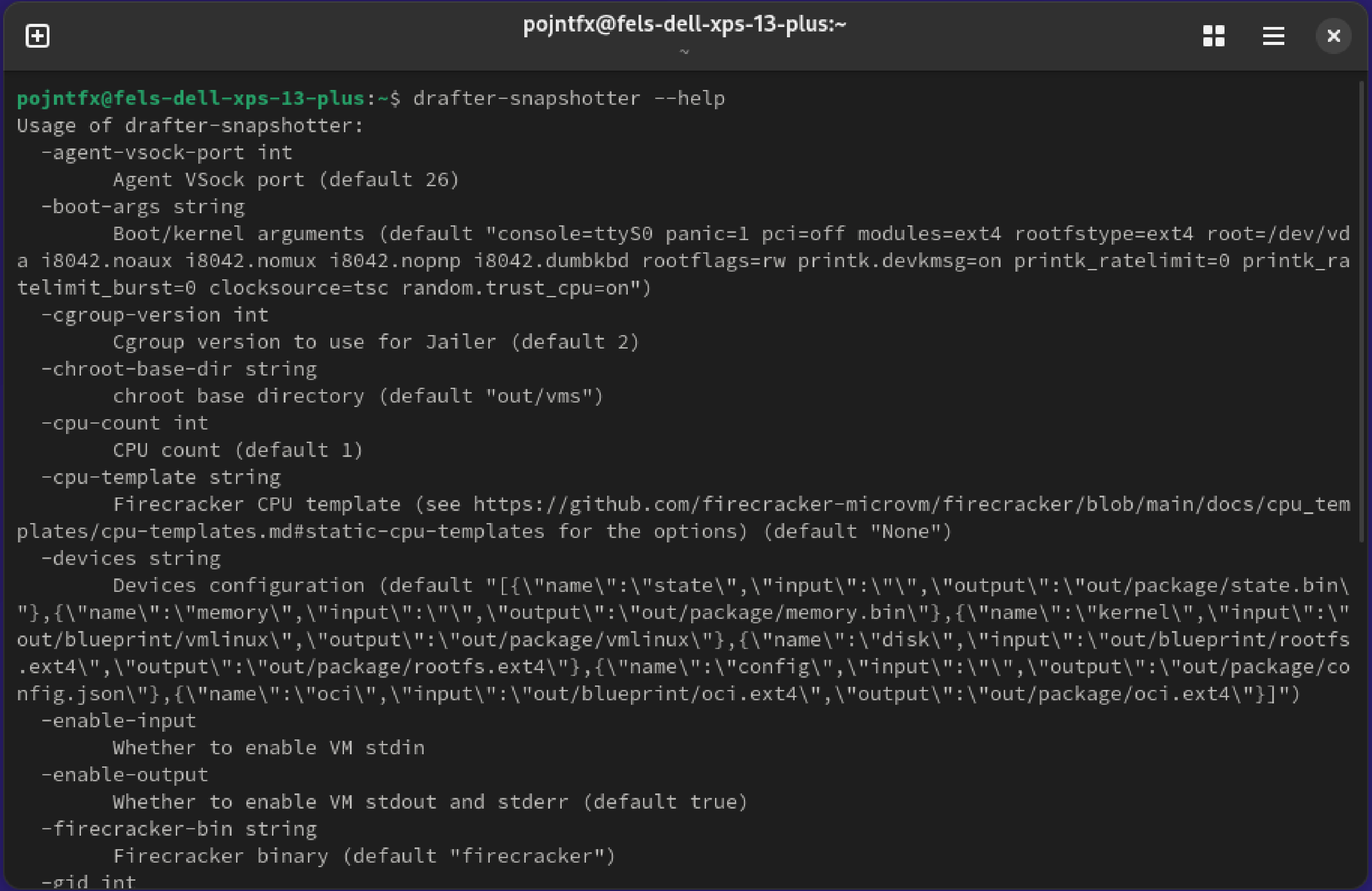
```
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$ drafter-forwarder --help
Usage of drafter-forwarder:
  -host-veth-cidr string
        CIDR for the veths outside the namespace (default "10.0.8.0/22")
  -port-forwards string
        Port forwards configuration (wildcard IPs like 0.0.0.0 are not valid, be explicit) (default "[{\"netns\
":\"ark0\",\"internalPort\":\"6379\",\"protocol\":\"tcp\",\"externalAddr\":\"127.0.0.1:3333\"}]")
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$
```


pojntfx@fels-dell-xps-13-plus:~/Projects/silo
~/Projects/silo

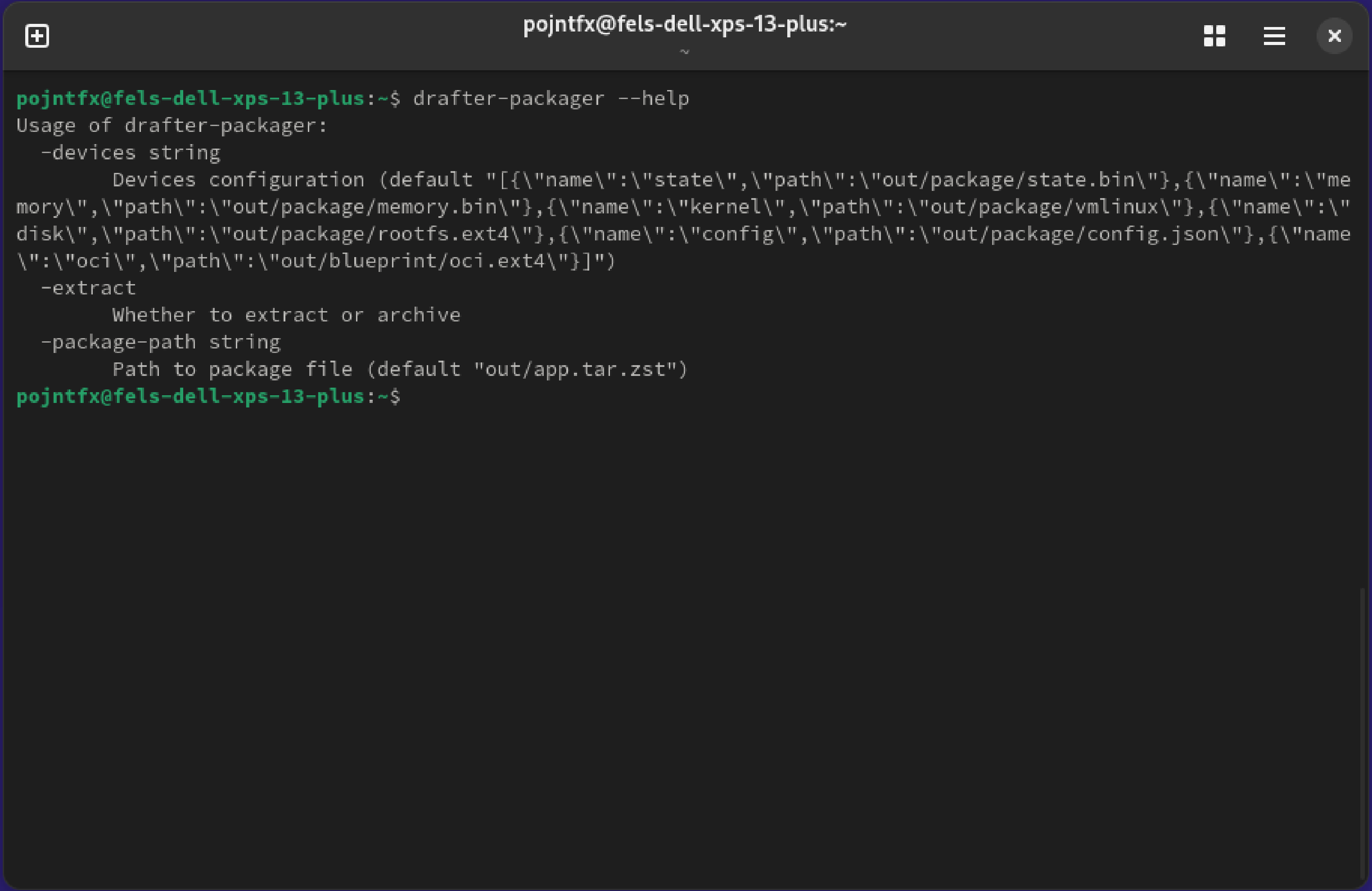
```
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$ drafter-liveness --help
Usage of drafter-liveness:
  -vsock-port int
        VSock port (default 25)
  -vsock-timeout duration
        VSock dial timeout (default 1m0s)
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$ 
```

pojntfx@fels-dell-xps-13-plus:~/Projects/silo
~/Projects/silo

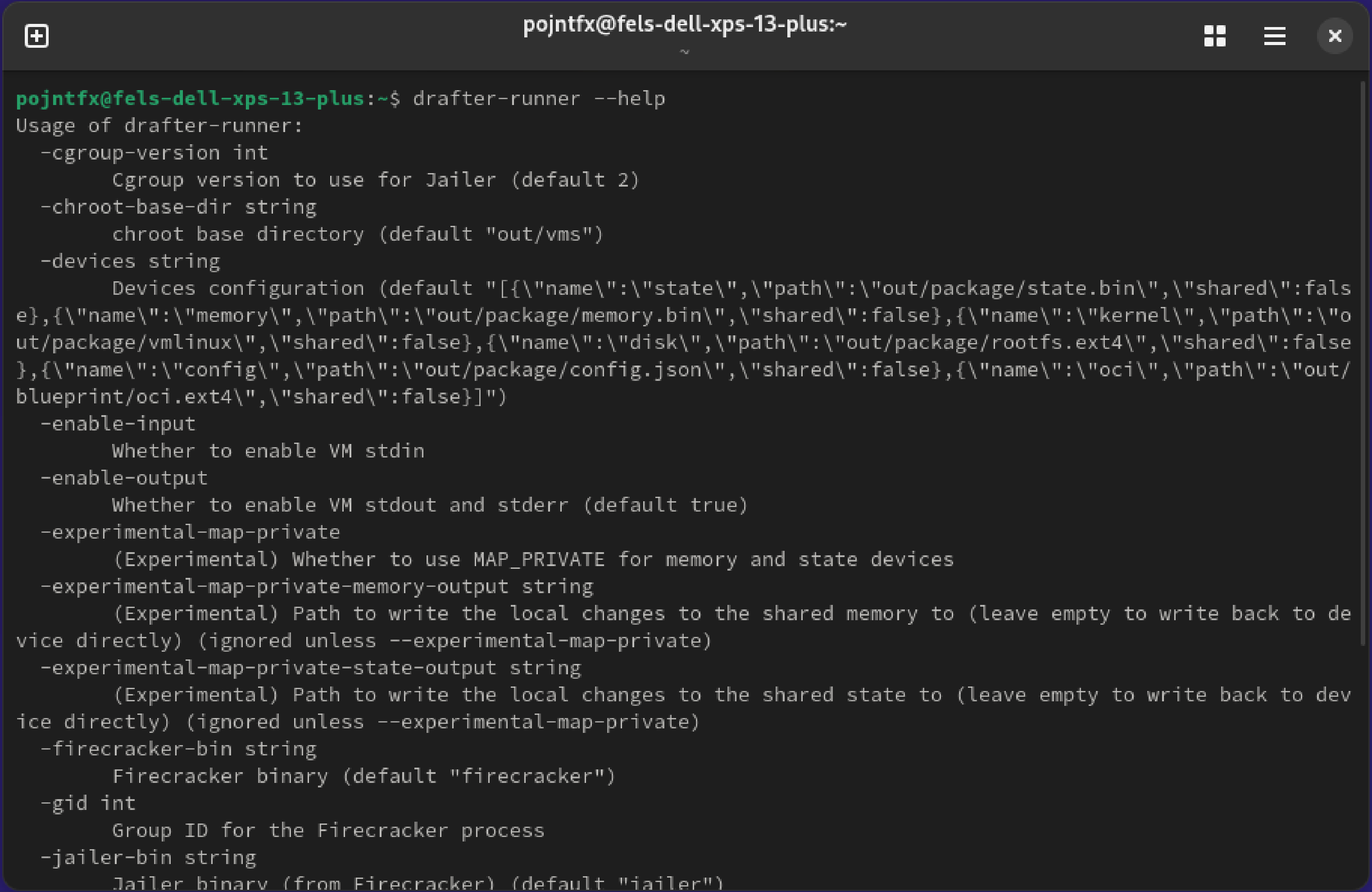
```
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$ drafter-agent --help
Usage of drafter-agent:
  -after-resume-cmd string
        Command to run after the VM has been resumed (leave empty to disable)
  -before-suspend-cmd string
        Command to run before the VM is suspended (leave empty to disable)
  -shell-cmd string
        Shell to use to run the before suspend and after resume commands (default "sh")
  -vsock-port uint
        VSock port (default 26)
  -vsock-timeout duration
        VSock dial timeout (default 1m0s)
pojntfx@fels-dell-xps-13-plus:~/Projects/silo$
```



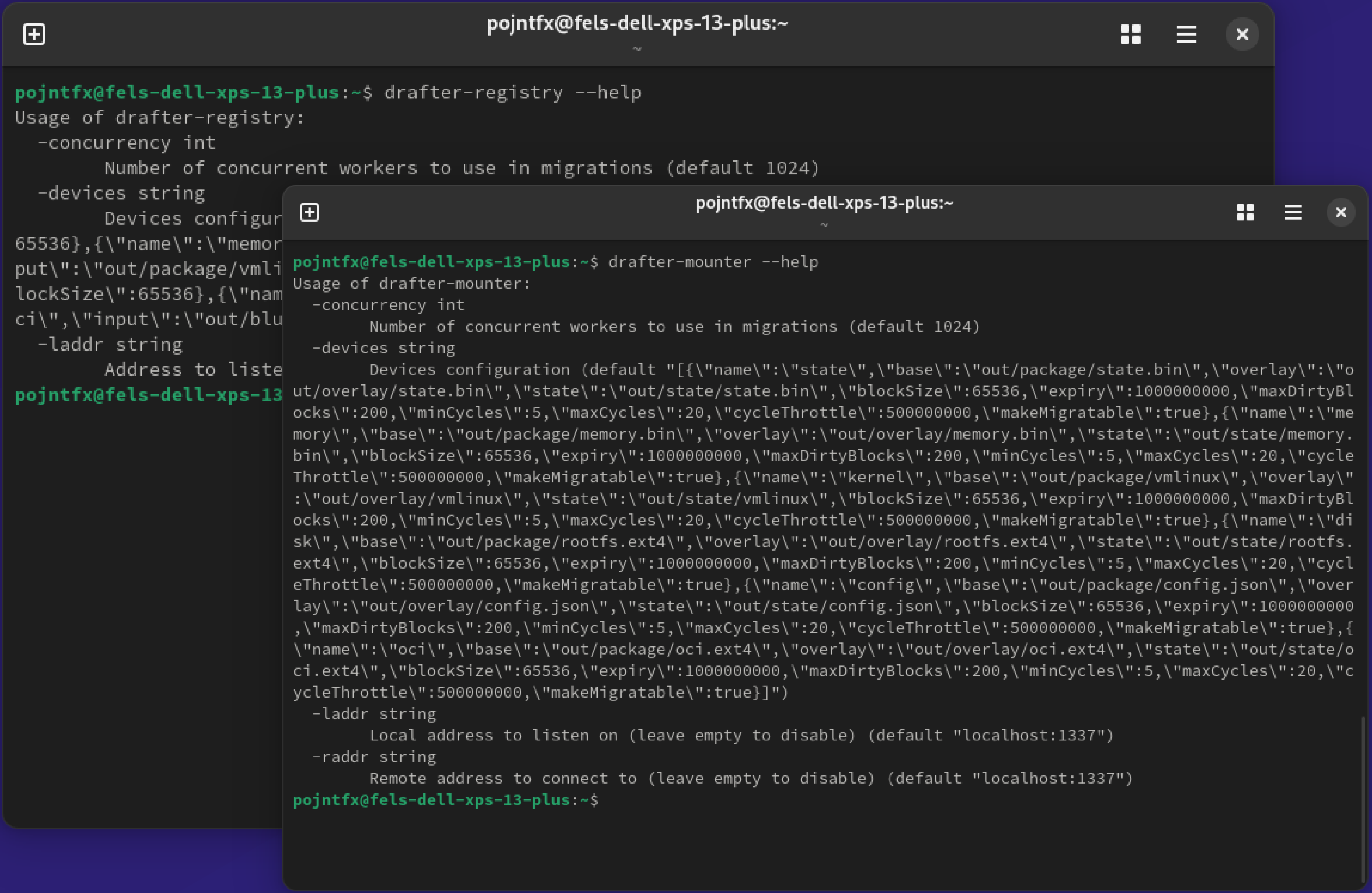
```
pojntfx@fels-dell-xps-13-plus:~$ drafter-snapshotter --help
Usage of drafter-snapshotter:
  -agent-vsock-port int
        Agent VSock port (default 26)
  -boot-args string
        Boot/kernel arguments (default "console=ttyS0 panic=1 pci=off modules=ext4 rootfstype=ext4 root=/dev/vd
a i8042.noaux i8042.nomux i8042.nopnp i8042.dumbkbd rootflags=rw printk.devkmsg=on printk_ratelimit=0 printk_ra
telimit_burst=0 clocksource=tsc random.trust_cpu=on")
  -cgroup-version int
        Cgroup version to use for Jailer (default 2)
  -chroot-base-dir string
        chroot base directory (default "out/vms")
  -cpu-count int
        CPU count (default 1)
  -cpu-template string
        Firecracker CPU template (see https://github.com/firecracker-microvm/firecracker/blob/main/docs/cpu_tem
plates/cpu-templates.md#static-cpu-templates for the options) (default "None")
  -devices string
        Devices configuration (default "[{\"name\":\"state\",\"input\":\"\",\"output\":\"out/package/state.bin\
"},{\"name\":\"memory\",\"input\":\"\",\"output\":\"out/package/memory.bin\"},{\"name\":\"kernel\",\"input\":\"
out/blueprint/vmlinux\",\"output\":\"out/package/vmlinux\"},{\"name\":\"disk\",\"input\":\"out/blueprint/rootfs
.ext4\",\"output\":\"out/package/rootfs.ext4\"},{\"name\":\"config\",\"input\":\"\",\"output\":\"out/package/co
nfig.json\"},{\"name\":\"oci\",\"input\":\"out/blueprint/oci.ext4\",\"output\":\"out/package/oci.ext4\"}]")
  -enable-input
        Whether to enable VM stdin
  -enable-output
        Whether to enable VM stdout and stderr (default true)
  -firecracker-bin string
        Firecracker binary (default "firecracker")
  -gid int
```



```
pojntfx@fels-dell-xps-13-plus:~$ drafter-packager --help
Usage of drafter-packager:
  -devices string
        Devices configuration (default "[{\"name\":\"state\",\"path\":\"out/package/state.bin\"},{\"name\":\"me
mory\",\"path\":\"out/package/memory.bin\"},{\"name\":\"kernel\",\"path\":\"out/package/vmlinux\"},{\"name\":\"
disk\",\"path\":\"out/package/rootfs.ext4\"},{\"name\":\"config\",\"path\":\"out/package/config.json\"},{\"name
\":\"oci\",\"path\":\"out/blueprint/oci.ext4\"}]")
  -extract
        Whether to extract or archive
  -package-path string
        Path to package file (default "out/app.tar.zst")
pojntfx@fels-dell-xps-13-plus:~$
```



```
pojntfx@fels-dell-xps-13-plus:~$ drafter-runner --help
Usage of drafter-runner:
  -cgroup-version int
        Cgroup version to use for Jailer (default 2)
  -chroot-base-dir string
        chroot base directory (default "out/vms")
  -devices string
        Devices configuration (default "[{\"name\":\"state\",\"path\":\"out/package/state.bin\",\"shared\":fals
e},{\"name\":\"memory\",\"path\":\"out/package/memory.bin\",\"shared\":false},{\"name\":\"kernel\",\"path\":\"o
ut/package/vmlinux\",\"shared\":false},{\"name\":\"disk\",\"path\":\"out/package/rootfs.ext4\",\"shared\":false
},{\"name\":\"config\",\"path\":\"out/package/config.json\",\"shared\":false},{\"name\":\"oci\",\"path\":\"out/
blueprint/oci.ext4\",\"shared\":false}]")
  -enable-input
        Whether to enable VM stdin
  -enable-output
        Whether to enable VM stdout and stderr (default true)
  -experimental-map-private
        (Experimental) Whether to use MAP_PRIVATE for memory and state devices
  -experimental-map-private-memory-output string
        (Experimental) Path to write the local changes to the shared memory to (leave empty to write back to de
vice directly) (ignored unless --experimental-map-private)
  -experimental-map-private-state-output string
        (Experimental) Path to write the local changes to the shared state to (leave empty to write back to dev
ice directly) (ignored unless --experimental-map-private)
  -firecracker-bin string
        Firecracker binary (default "firecracker")
  -gid int
        Group ID for the Firecracker process
  -jailer-bin string
        Jailer binary (from Firecracker) (default "jailer")
```

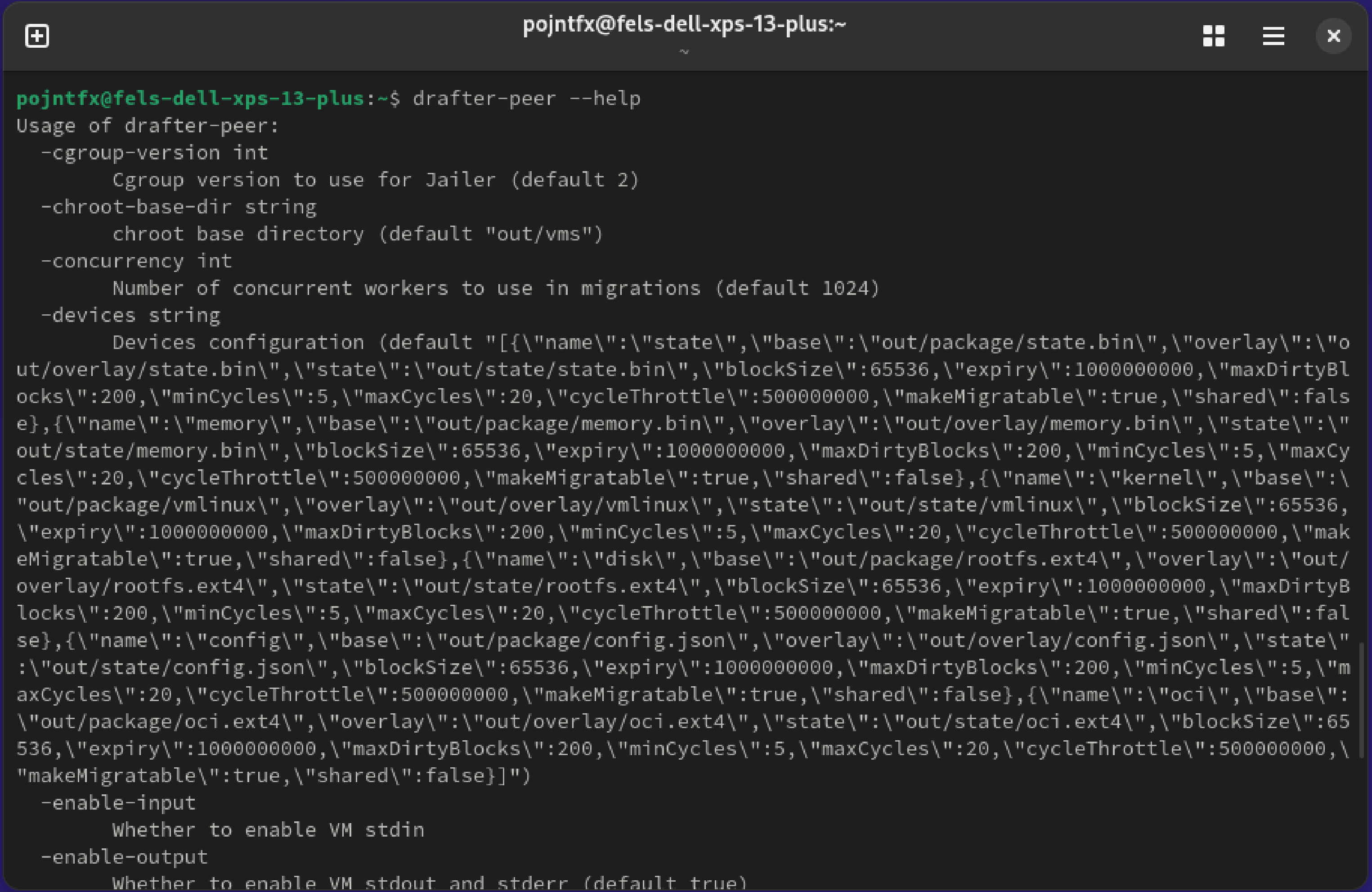


pojntfx@fels-dell-xps-13-plus:~
~

```
pojntfx@fels-dell-xps-13-plus:~$ drafter-registry --help
Usage of drafter-registry:
  -concurrency int
        Number of concurrent workers to use in migrations (default 1024)
  -devices string
        Devices configur
65536},{\"name\":\"memor
put\":\"out/package/vmli
lockSize\":65536},{\"nam
ci\",\"input\":\"out/blu
  -laddr string
        Address to liste
pojntfx@fels-dell-xps-13
```

pojntfx@fels-dell-xps-13-plus:~
~

```
pojntfx@fels-dell-xps-13-plus:~$ drafter-mounter --help
Usage of drafter-mounter:
  -concurrency int
        Number of concurrent workers to use in migrations (default 1024)
  -devices string
        Devices configuration (default "[{\"name\":\"state\",\"base\":\"out/package/state.bin\",\"overlay\":\"o
ut/overlay/state.bin\",\"state\":\"out/state/state.bin\",\"blockSize\":65536,\"expiry\":1000000000,\"maxDirtyBl
ocks\":200,\"minCycles\":5,\"maxCycles\":20,\"cycleThrottle\":500000000,\"makeMigratable\":true},{\"name\":\"me
mory\",\"base\":\"out/package/memory.bin\",\"overlay\":\"out/overlay/memory.bin\",\"state\":\"out/state/memory.
bin\",\"blockSize\":65536,\"expiry\":1000000000,\"maxDirtyBlocks\":200,\"minCycles\":5,\"maxCycles\":20,\"cycle
Throttle\":500000000,\"makeMigratable\":true},{\"name\":\"kernel\",\"base\":\"out/package/vmlinux\",\"overlay\"
:\"out/overlay/vmlinux\",\"state\":\"out/state/vmlinux\",\"blockSize\":65536,\"expiry\":1000000000,\"maxDirtyBl
ocks\":200,\"minCycles\":5,\"maxCycles\":20,\"cycleThrottle\":500000000,\"makeMigratable\":true},{\"name\":\"di
sk\",\"base\":\"out/package/rootfs.ext4\",\"overlay\":\"out/overlay/rootfs.ext4\",\"state\":\"out/state/rootfs.
ext4\",\"blockSize\":65536,\"expiry\":1000000000,\"maxDirtyBlocks\":200,\"minCycles\":5,\"maxCycles\":20,\"cycl
eThrottle\":500000000,\"makeMigratable\":true},{\"name\":\"config\",\"base\":\"out/package/config.json\",\"over
lay\":\"out/overlay/config.json\",\"state\":\"out/state/config.json\",\"blockSize\":65536,\"expiry\":1000000000
,\"maxDirtyBlocks\":200,\"minCycles\":5,\"maxCycles\":20,\"cycleThrottle\":500000000,\"makeMigratable\":true},{
\"name\":\"oci\",\"base\":\"out/package/oci.ext4\",\"overlay\":\"out/overlay/oci.ext4\",\"state\":\"out/state/o
ci.ext4\",\"blockSize\":65536,\"expiry\":1000000000,\"maxDirtyBlocks\":200,\"minCycles\":5,\"maxCycles\":20,\"c
ycleThrottle\":500000000,\"makeMigratable\":true}]")
  -laddr string
        Local address to listen on (leave empty to disable) (default "localhost:1337")
  -raddr string
        Remote address to connect to (leave empty to disable) (default "localhost:1337")
pojntfx@fels-dell-xps-13-plus:~$
```



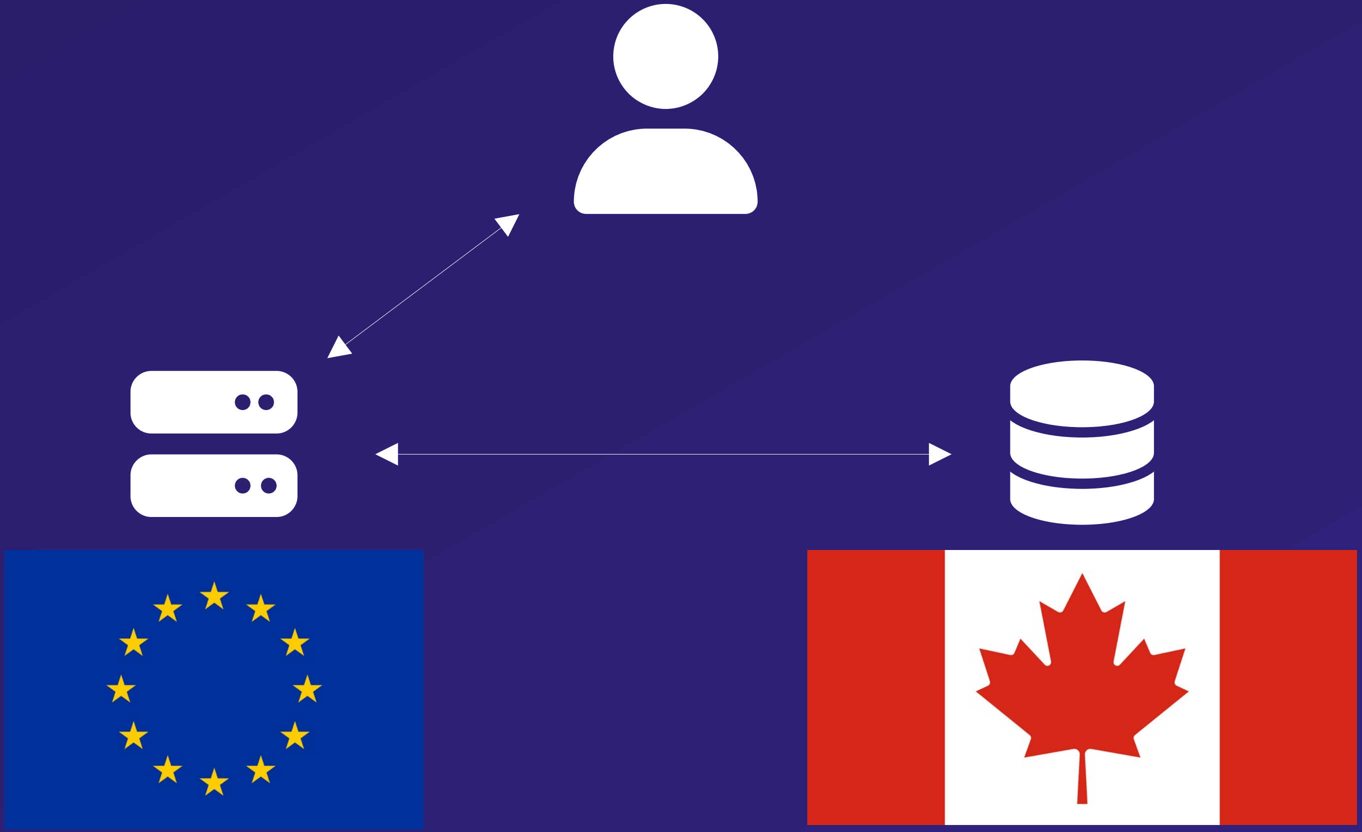
```
pojntfx@fels-dell-xps-13-plus:~$ drafter-peer --help
Usage of drafter-peer:
  -cgroup-version int
        Cgroup version to use for Jailer (default 2)
  -chroot-base-dir string
        chroot base directory (default "out/vms")
  -concurrency int
        Number of concurrent workers to use in migrations (default 1024)
  -devices string
        Devices configuration (default "[{\"name\":\"state\",\"base\":\"out/package/state.bin\",\"overlay\":\"o
ut/overlay/state.bin\",\"state\":\"out/state/state.bin\",\"blockSize\":65536,\"expiry\":1000000000,\"maxDirtyBl
ocks\":200,\"minCycles\":5,\"maxCycles\":20,\"cycleThrottle\":500000000,\"makeMigratable\":true,\"shared\":fals
e},{\"name\":\"memory\",\"base\":\"out/package/memory.bin\",\"overlay\":\"out/overlay/memory.bin\",\"state\":\"
out/state/memory.bin\",\"blockSize\":65536,\"expiry\":1000000000,\"maxDirtyBlocks\":200,\"minCycles\":5,\"maxCy
cles\":20,\"cycleThrottle\":500000000,\"makeMigratable\":true,\"shared\":false},{\"name\":\"kernel\",\"base\":\
"out/package/vmlinux\",\"overlay\":\"out/overlay/vmlinux\",\"state\":\"out/state/vmlinux\",\"blockSize\":65536,
\"expiry\":1000000000,\"maxDirtyBlocks\":200,\"minCycles\":5,\"maxCycles\":20,\"cycleThrottle\":500000000,\"mak
eMigratable\":true,\"shared\":false},{\"name\":\"disk\",\"base\":\"out/package/rootfs.ext4\",\"overlay\":\"out/
overlay/rootfs.ext4\",\"state\":\"out/state/rootfs.ext4\",\"blockSize\":65536,\"expiry\":1000000000,\"maxDirtyB
locks\":200,\"minCycles\":5,\"maxCycles\":20,\"cycleThrottle\":500000000,\"makeMigratable\":true,\"shared\":fal
se},{\"name\":\"config\",\"base\":\"out/package/config.json\",\"overlay\":\"out/overlay/config.json\",\"state\"
:\"out/state/config.json\",\"blockSize\":65536,\"expiry\":1000000000,\"maxDirtyBlocks\":200,\"minCycles\":5,\"m
axCycles\":20,\"cycleThrottle\":500000000,\"makeMigratable\":true,\"shared\":false},{\"name\":\"oci\",\"base\":
\"out/package/oci.ext4\",\"overlay\":\"out/overlay/oci.ext4\",\"state\":\"out/state/oci.ext4\",\"blockSize\":65
536,\"expiry\":1000000000,\"maxDirtyBlocks\":200,\"minCycles\":5,\"maxCycles\":20,\"cycleThrottle\":500000000,\
"makeMigratable\":true,\"shared\":false}]")
  -enable-input
        Whether to enable VM stdin
  -enable-output
        Whether to enable VM stdout and stderr (default true)
```

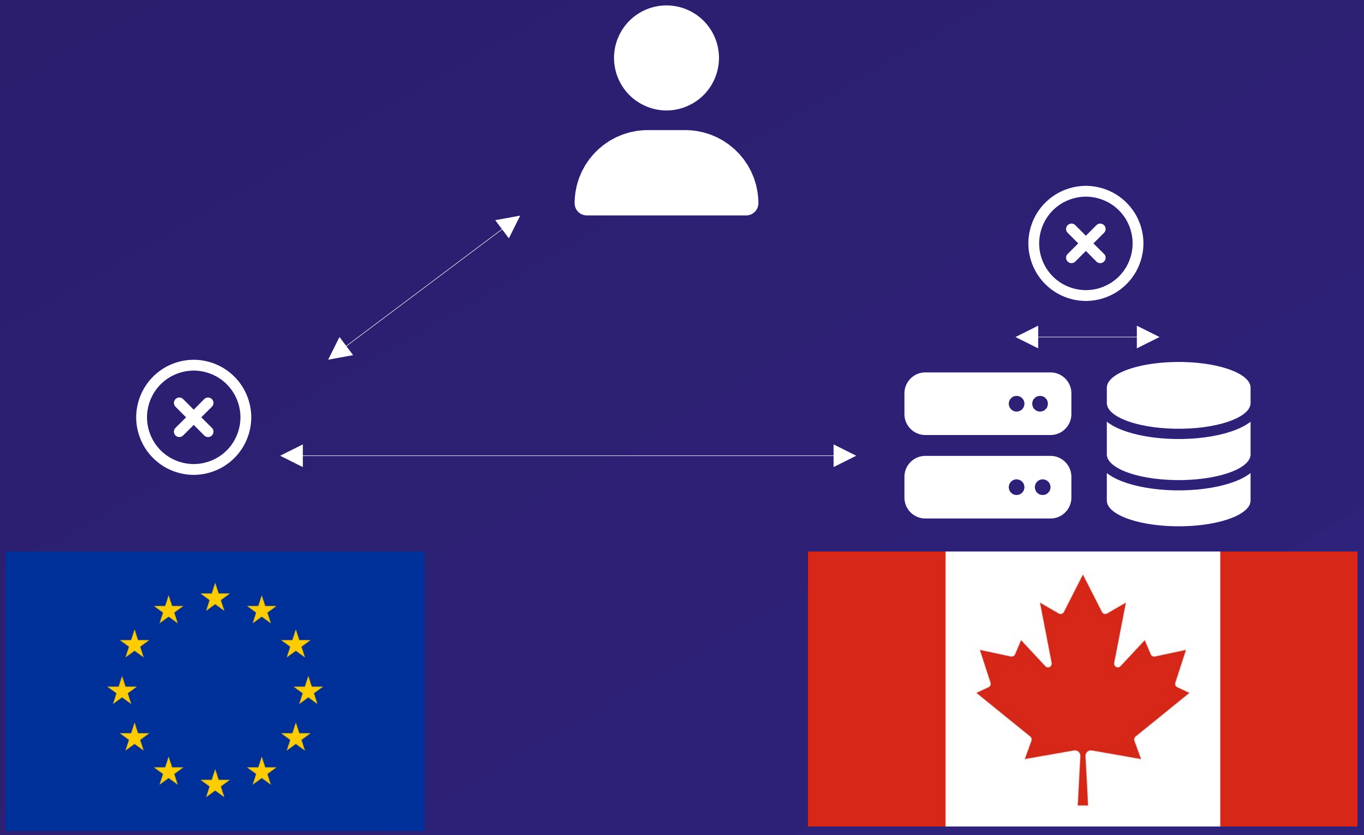
Demo 1:
Live Migrating Valkey/Redis between Cloud Providers with Drafter

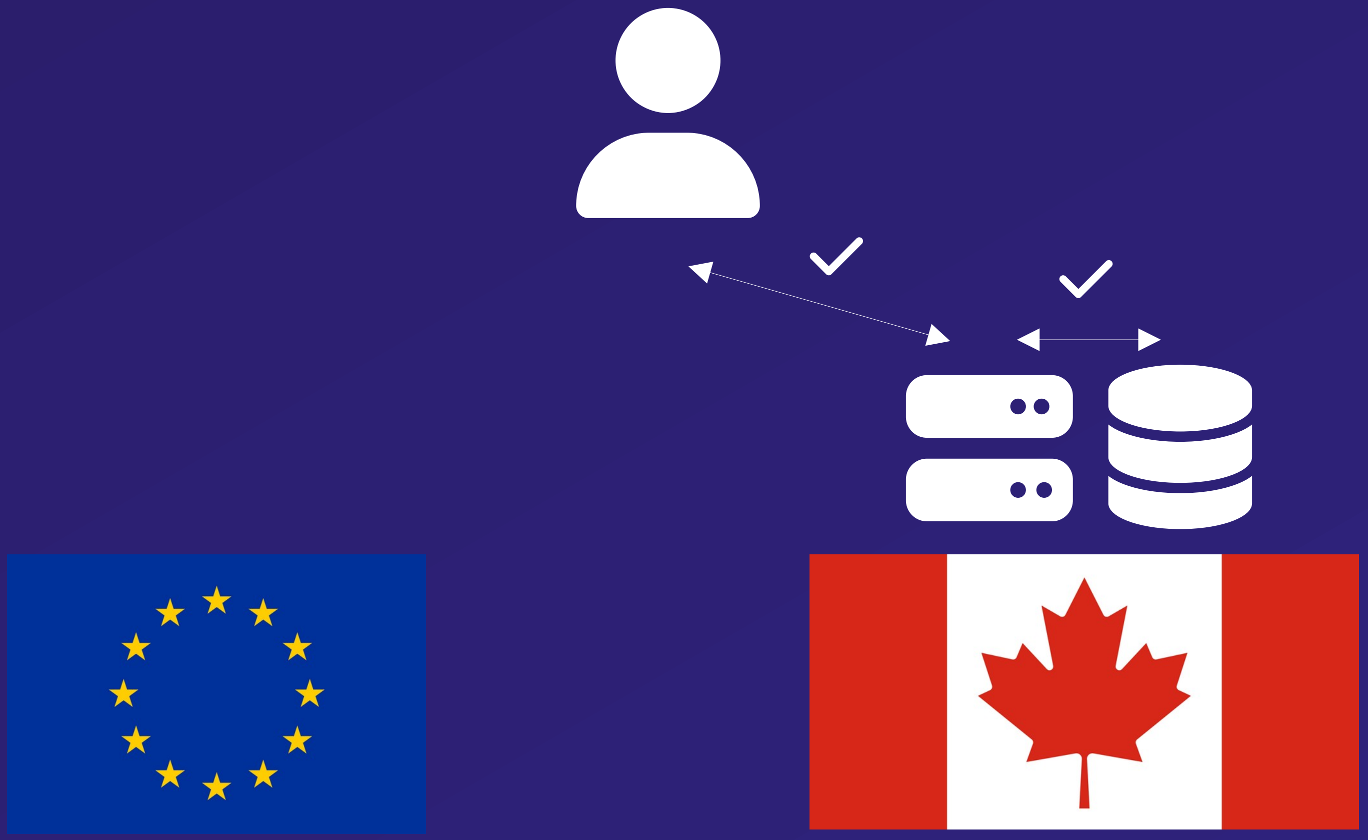
Loophole Labs

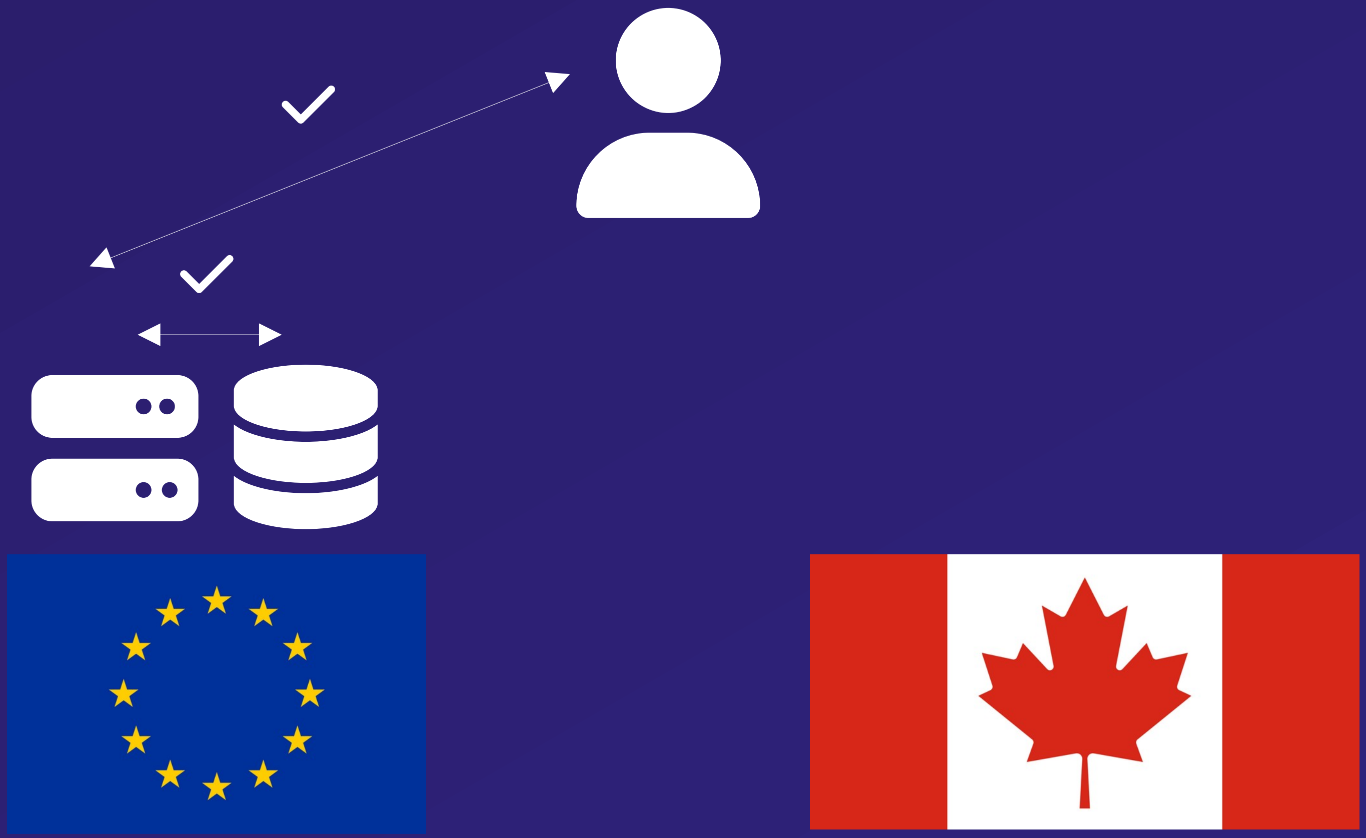
Component 5: **Conduit**

Loophole Labs









Bringing it all together: Architect

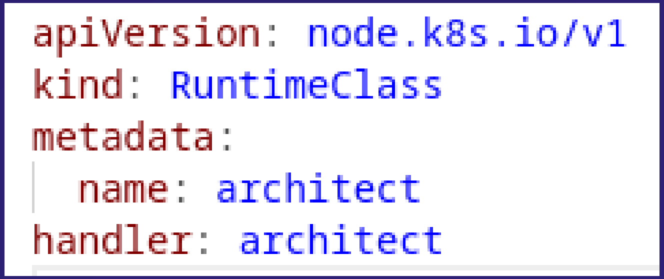
Demo 2:

Live Migrating a Kubernetes Cluster between Cloud Providers with Architect

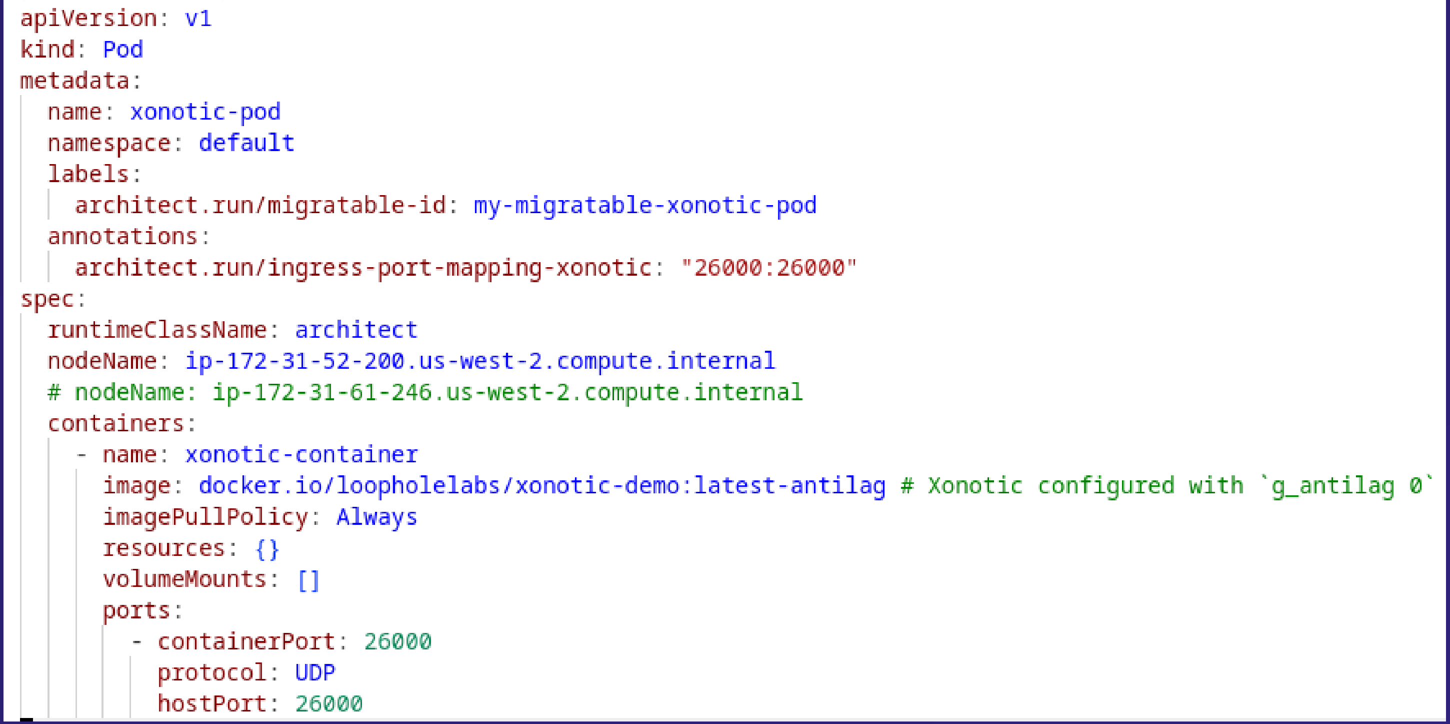
Loophole Labs

Can run **everything**

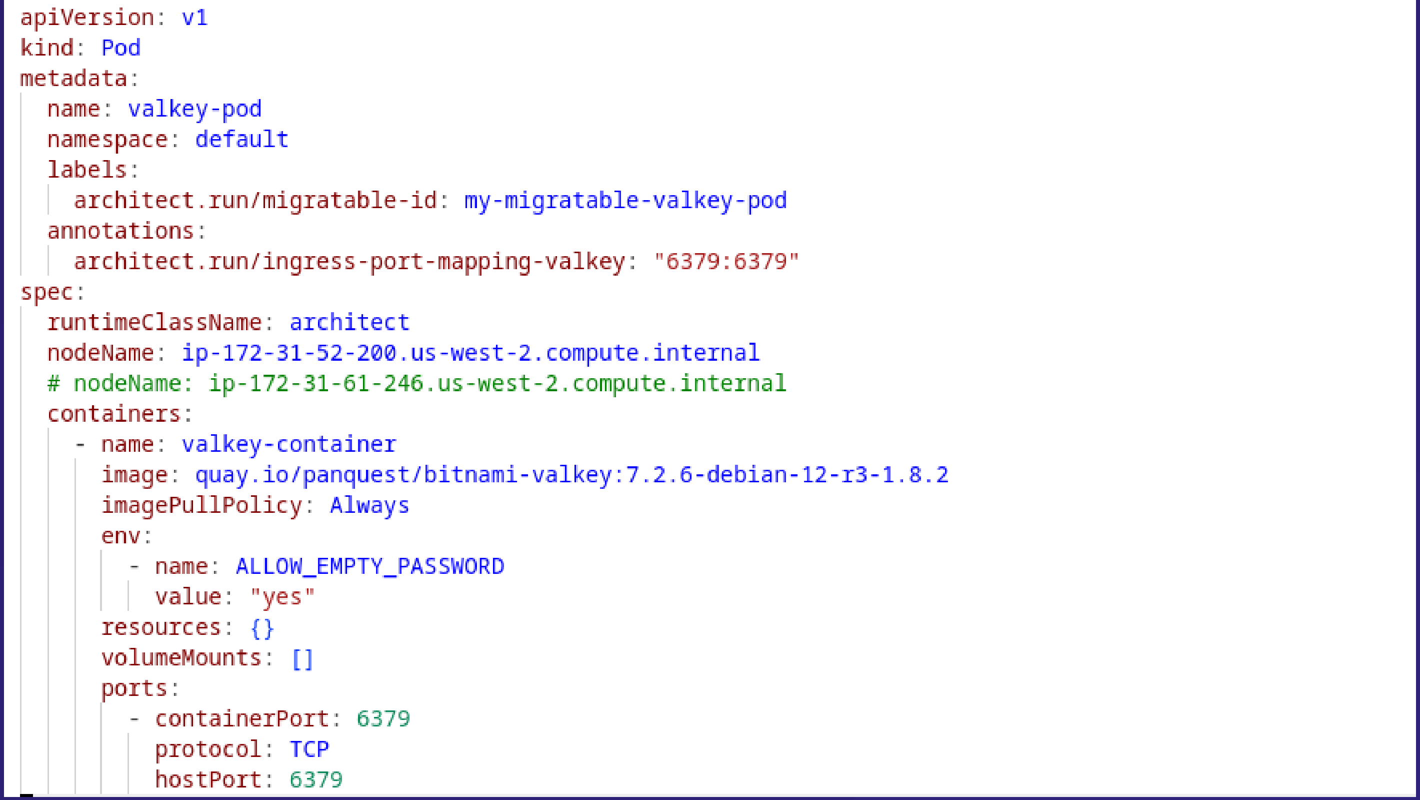
Can run **everywhere**



```yaml
apiVersion: node.k8s.io/v1
kind: RuntimeClass
metadata:
  name: architect
handler: architect
```



```yaml
apiVersion: v1
kind: Pod
metadata:
  name: xonotic-pod
  namespace: default
  labels:
    architect.run/migratable-id: my-migratable-xonotic-pod
  annotations:
    architect.run/ingress-port-mapping-xonotic: "26000:26000"
spec:
  runtimeClassName: architect
  nodeName: ip-172-31-52-200.us-west-2.compute.internal
  # nodeName: ip-172-31-61-246.us-west-2.compute.internal
  containers:
    - name: xonotic-container
      image: docker.io/loopholelabs/xonotic-demo:latest-antilag # Xonotic configured with `g_antilag 0`
      imagePullPolicy: Always
      resources: {}
      volumeMounts: []
      ports:
        - containerPort: 26000
          protocol: UDP
          hostPort: 26000
```

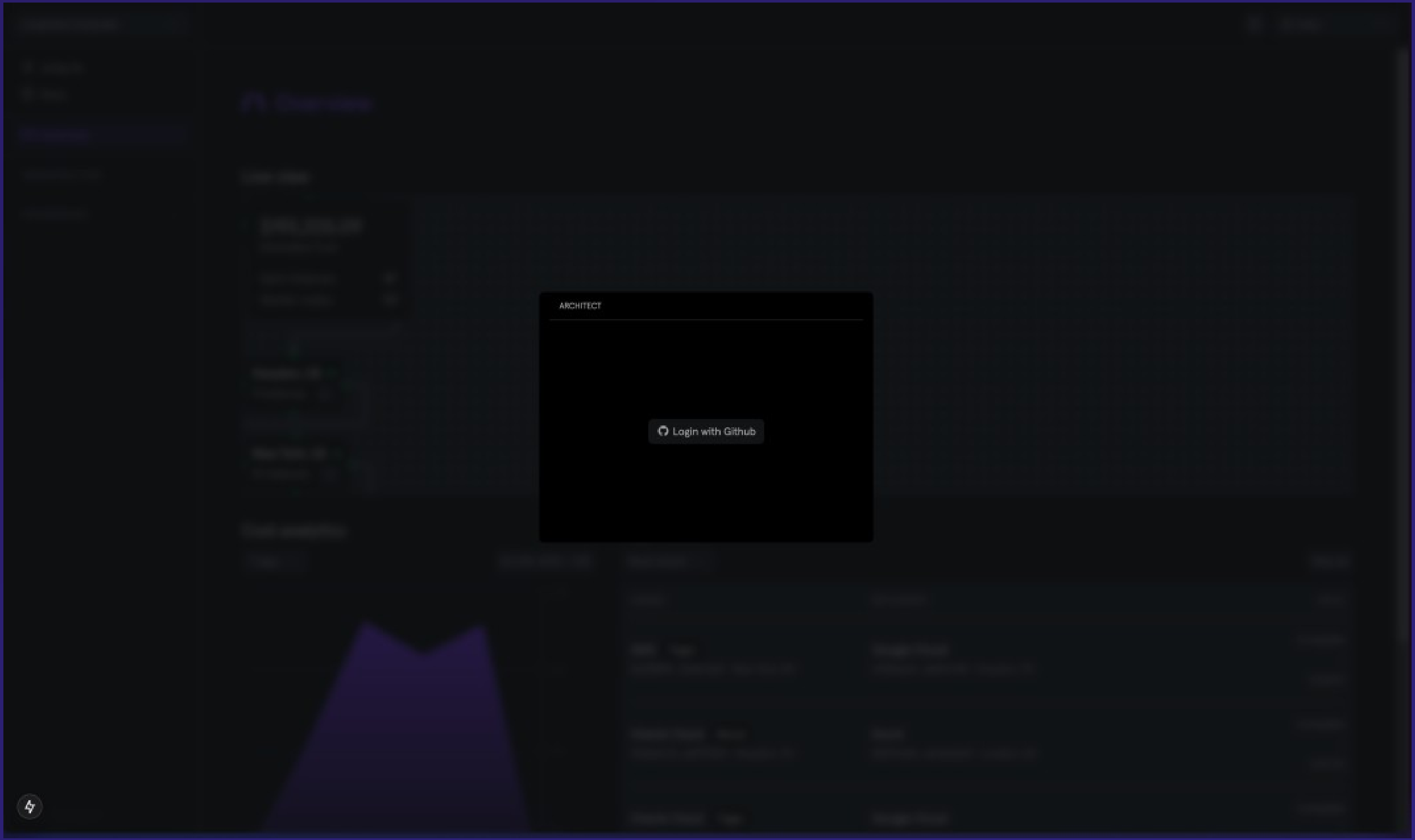


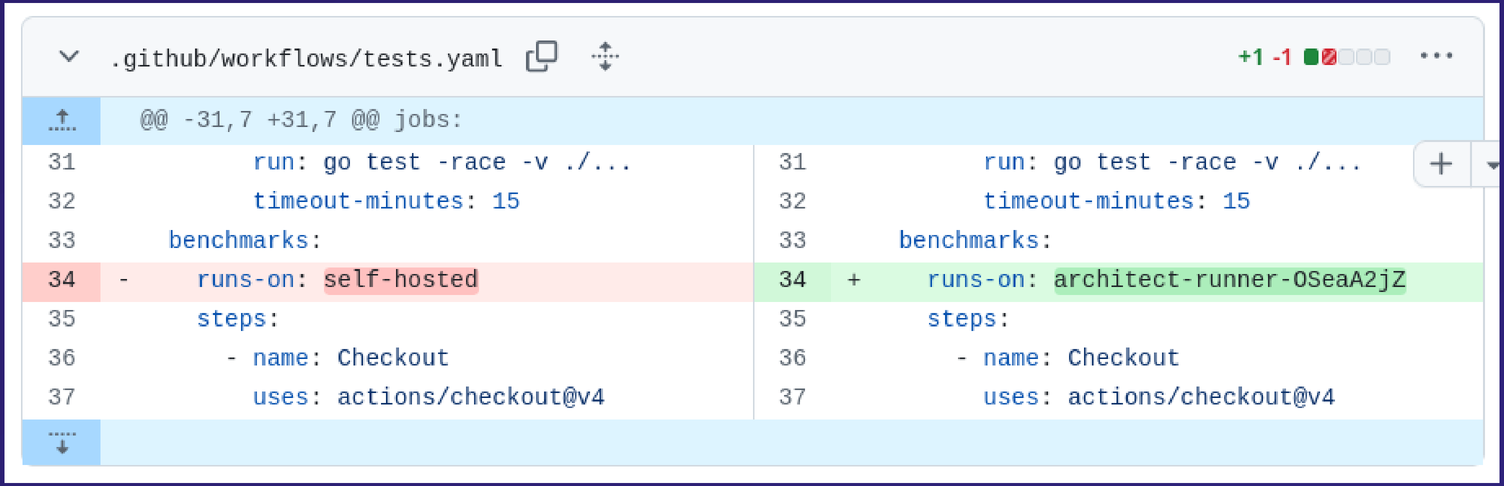
```yaml
apiVersion: v1
kind: Pod
metadata:
  name: valkey-pod
  namespace: default
  labels:
    architect.run/migratable-id: my-migratable-valkey-pod
  annotations:
    architect.run/ingress-port-mapping-valkey: "6379:6379"
spec:
  runtimeClassName: architect
  nodeName: ip-172-31-52-200.us-west-2.compute.internal
  # nodeName: ip-172-31-61-246.us-west-2.compute.internal
  containers:
    - name: valkey-container
      image: quay.io/panquest/bitnami-valkey:7.2.6-debian-12-r3-1.8.2
      imagePullPolicy: Always
      env:
        - name: ALLOW_EMPTY_PASSWORD
          value: "yes"
      resources: {}
      volumeMounts: []
      ports:
        - containerPort: 6379
          protocol: TCP
          hostPort: 6379
```

Demo 3:

GitHub Actions CI/CD on Spot Instances with Architect

Loophole Labs

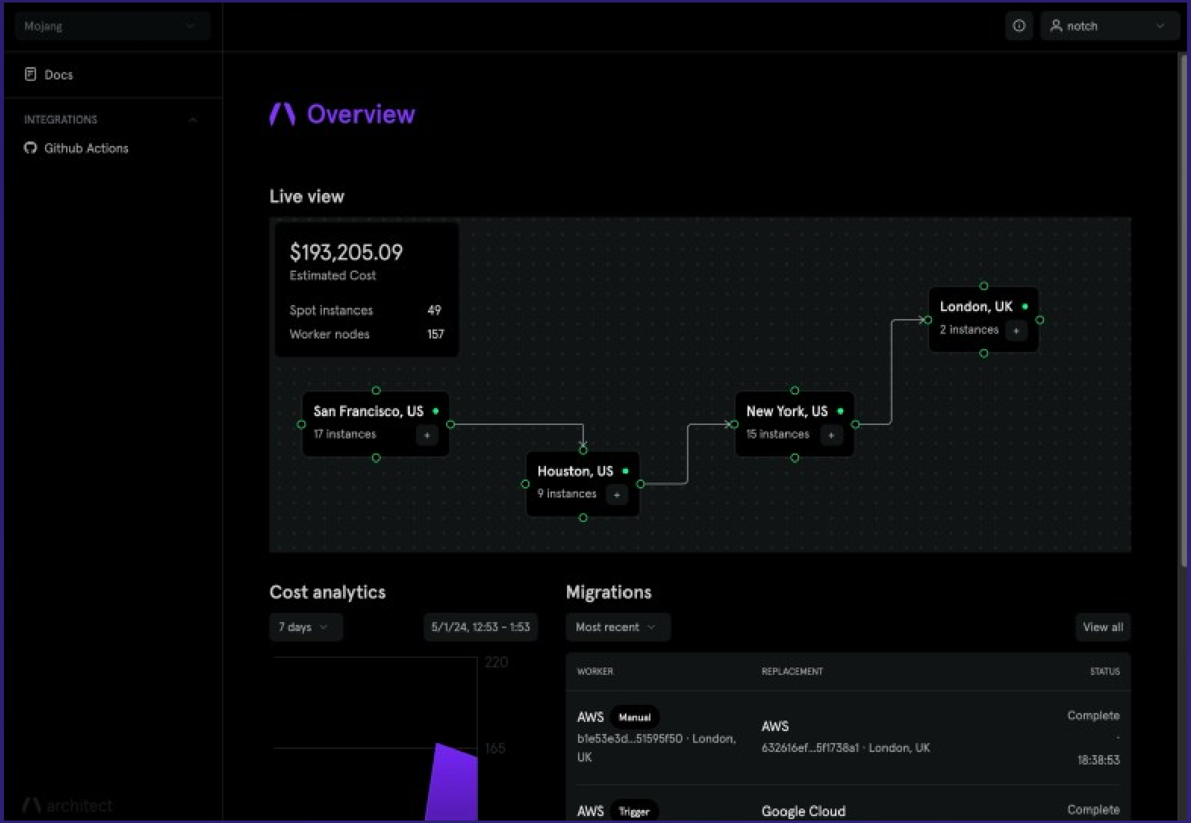

ARCHITECT
Login with Github



.github/workflows/tests.yaml

+1 -1

```diff
@@ -31,7 +31,7 @@ jobs:
         run: go test -race -v ./...
         timeout-minutes: 15
   benchmarks:
-    runs-on: self-hosted
+    runs-on: architect-runner-OSeaA2jZ
     steps:
       - name: Checkout
         uses: actions/checkout@v4
```



Overview

Live view

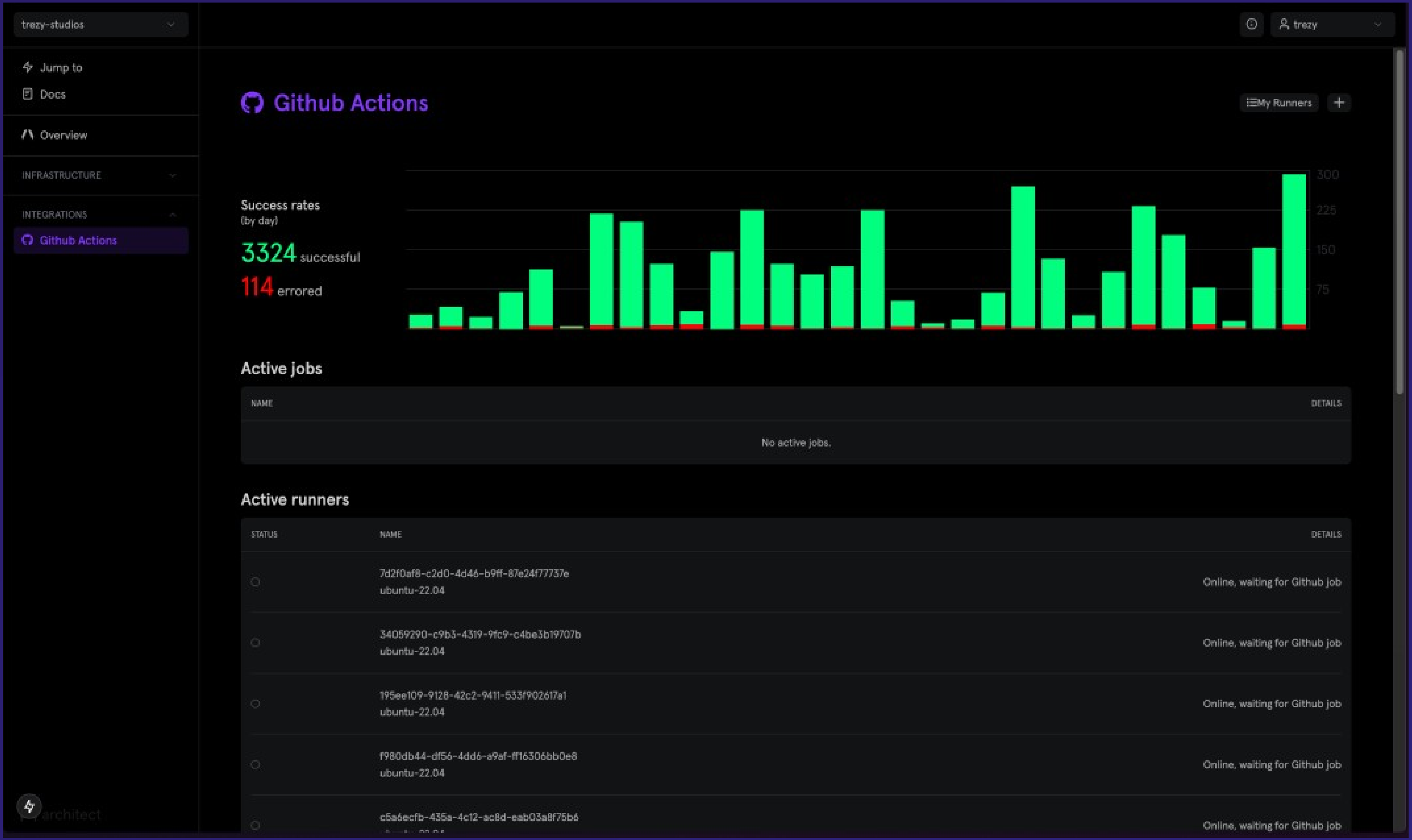
$193,205.09
Estimated Cost

Spot instances	49
Worker nodes	157

Cost analytics

Migrations

WORKER	REPLACEMENT	STATUS
AWS Manual b1e53e3d...51595f50 · London, UK	AWS 632616ef...5f1738a1 · London, UK	Complete - 18:38:53
AWS Trigger	Google Cloud	Complete



trezy-studios

trezy

Jump to

Docs

Overview

INFRASTRUCTURE

INTEGRATIONS

Github Actions

architect

Github Actions

My Runners +

Success rates
(by day)

3324 successful
114 errored

Active jobs

NAME	DETAILS
No active jobs.	

Active runners

STATUS	NAME	DETAILS
○	7d2f0af8-c2d0-4d46-b9ff-87e24f77737e ubuntu-22.04	Online, waiting for Github job
○	34059290-c9b3-4319-9fc9-c4be3b19707b ubuntu-22.04	Online, waiting for Github job
○	195ee109-9128-42c2-9411-533f902617a1 ubuntu-22.04	Online, waiting for Github job
○	f980db44-df56-4dd6-a9af-ff16306bb0e8 ubuntu-22.04	Online, waiting for Github job
○	c5a6ecfb-435a-4c12-ac8d-eab03a8f75b6	Online, waiting for Github job

Recap

Loophole Labs

Recap

Commoditized Compute

Silo

Architect

PVM

Conduit

Drafter

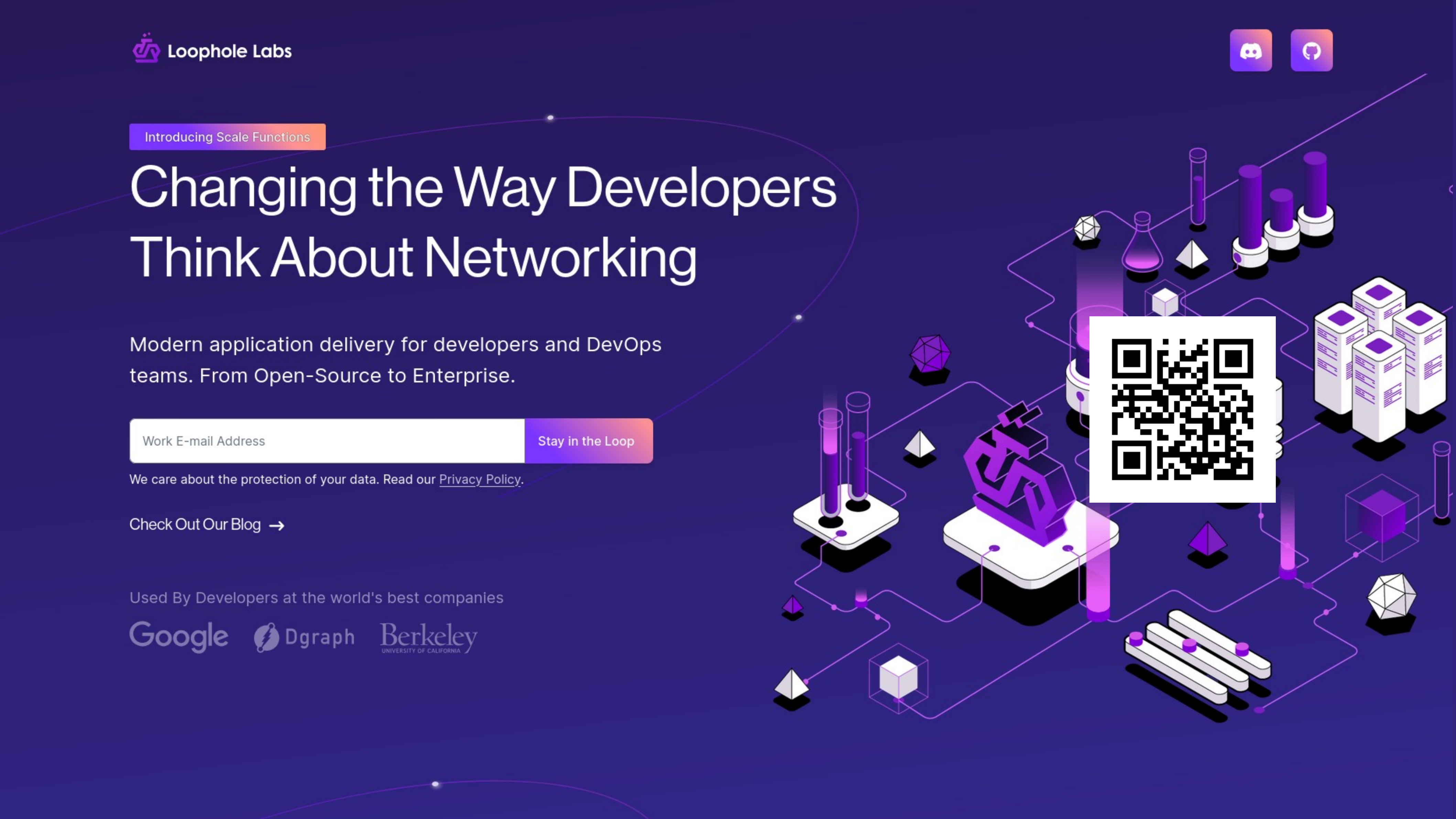
Links and Resources

https://loophole.sh/kubecon2024

Felicitas Pojtinger

Fediverse:	@pojntfx@mastodon.social
Bluesky:	@pojntfx.mastodon.social.ap.brid.gy
Github:	@pojntfx
LinkedIn:	in/pojntfx
Web:	felicitas.pojtinger.com

Loophole Labs



Loophole Labs

Introducing Scale Functions

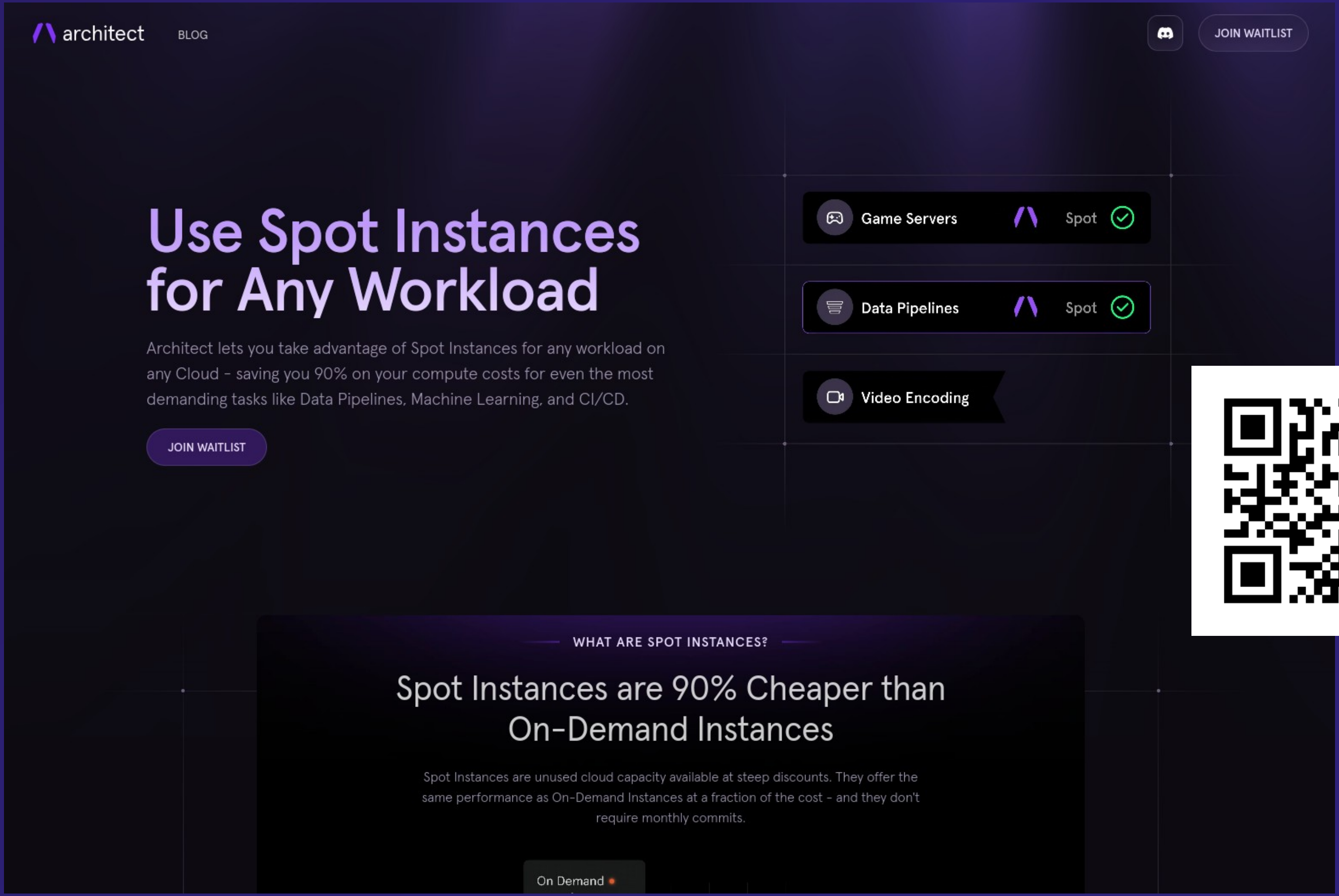
Changing the Way Developers Think About Networking

Modern application delivery for developers and DevOps teams. From Open-Source to Enterprise.

We care about the protection of your data. Read our Privacy Policy.

Check Out Our Blog →

Used By Developers at the world's best companies



Use Spot Instances for Any Workload

Architect lets you take advantage of Spot Instances for any workload on any Cloud - saving you 90% on your compute costs for even the most demanding tasks like Data Pipelines, Machine Learning, and CI/CD.

WHAT ARE SPOT INSTANCES?

Spot Instances are 90% Cheaper than On-Demand Instances

Spot Instances are unused cloud capacity available at steep discounts. They offer the same performance as On-Demand Instances at a fraction of the cost - and they don't require monthly commits.

https://architect.run/