Progress Report Meeting

Hypertracing
Tracing Through Virtualization Layers

Abderrahmane Benbachir

May 10, 2018

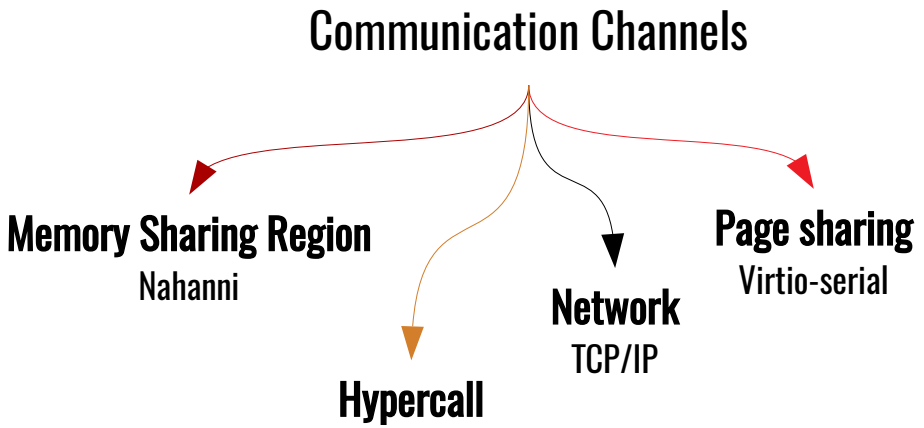
École Polytechnique de Montréal

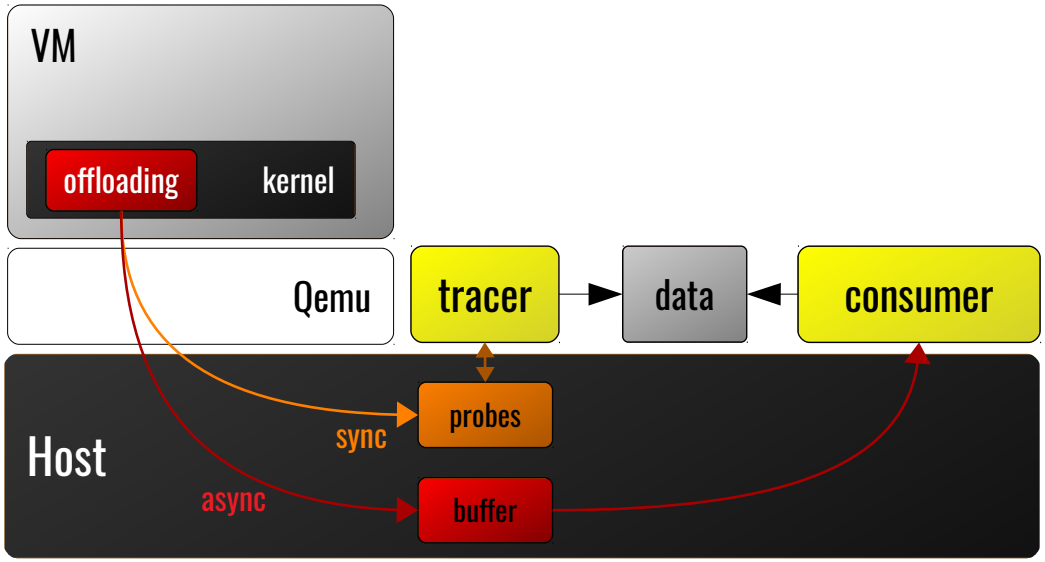
Laboratoire **DORSAL**

Agenda

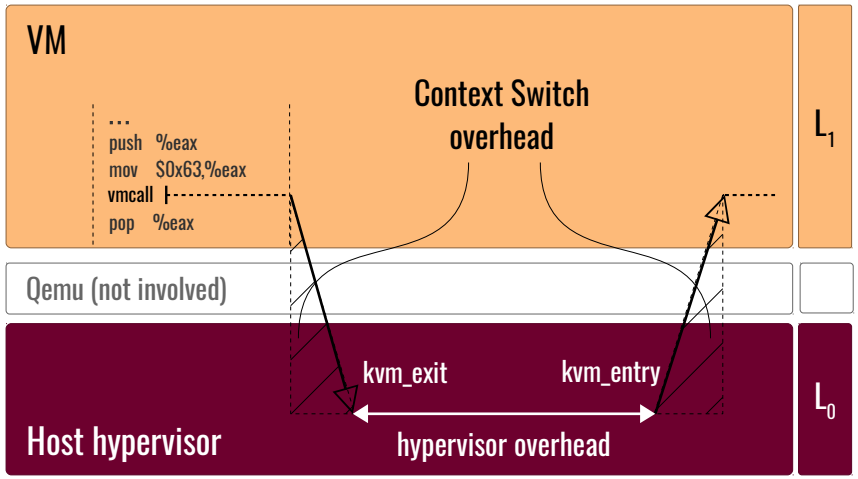
- Introduction
- Tracing With Paravirtualization
- Use Cases
- Questions

What is Hypertracing



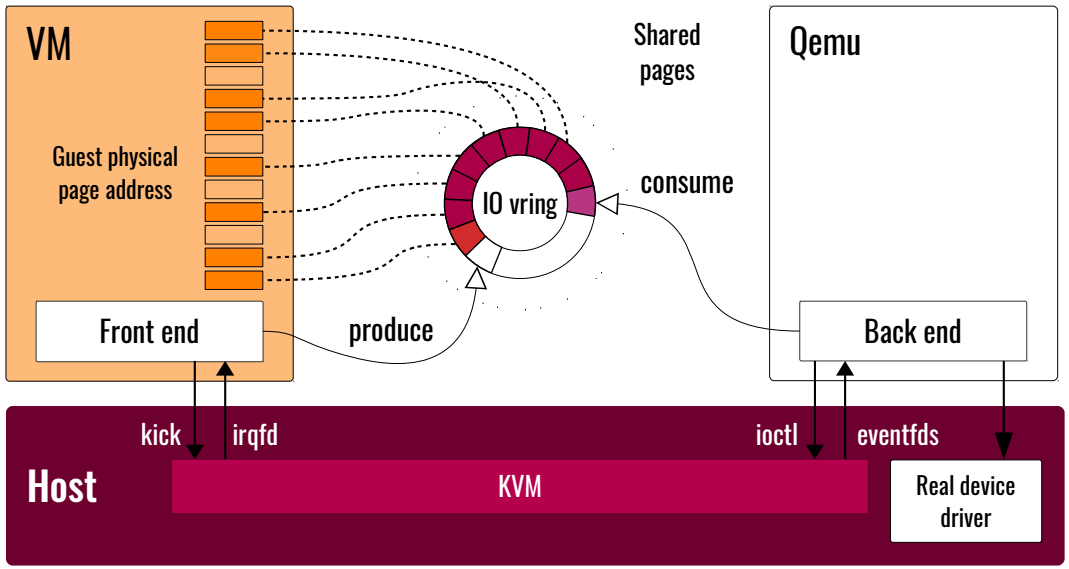


Hypercall



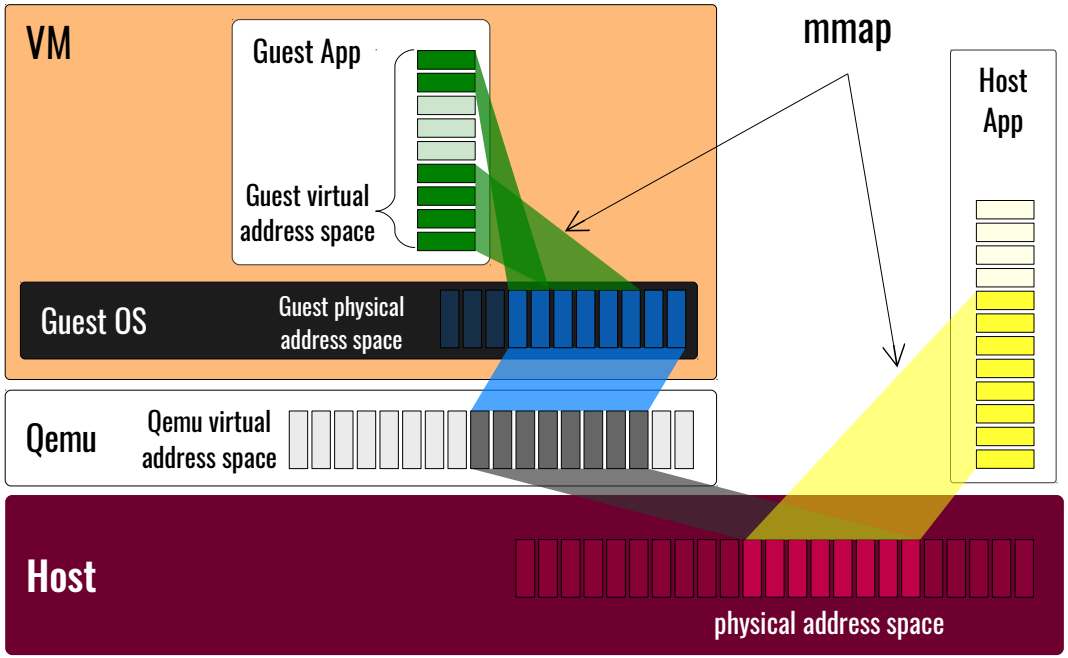

```
88f: 55                 push   %rbp
890: 48 89 e5           mov    %rsp,%rbp
893: 53                 push   %rbx
894: 89 7d f4           mov    %edi,-0xc(%rbp)
897: 48 89 75 e8        mov    %rsi,-0x18(%rbp)
...
89b: b8 63 00 00 00     mov    0x63,%eax
8a0: bf 64 00 00 00     mov    0x64,%edi
8a5: b9 65 00 00 00     mov    0x65,%ecx
8aa: ba 66 00 00 00     mov    0x66,%edx
8af: be 67 00 00 00     mov    0x67,%esi
8b4: 89 fb              mov    edi,%ebx

8b6: 0f 01 c1           vmcall

8b9: 5b                 pop    %rbx
8ba: 5d                 pop    %rbp
...
```

Page Sharing

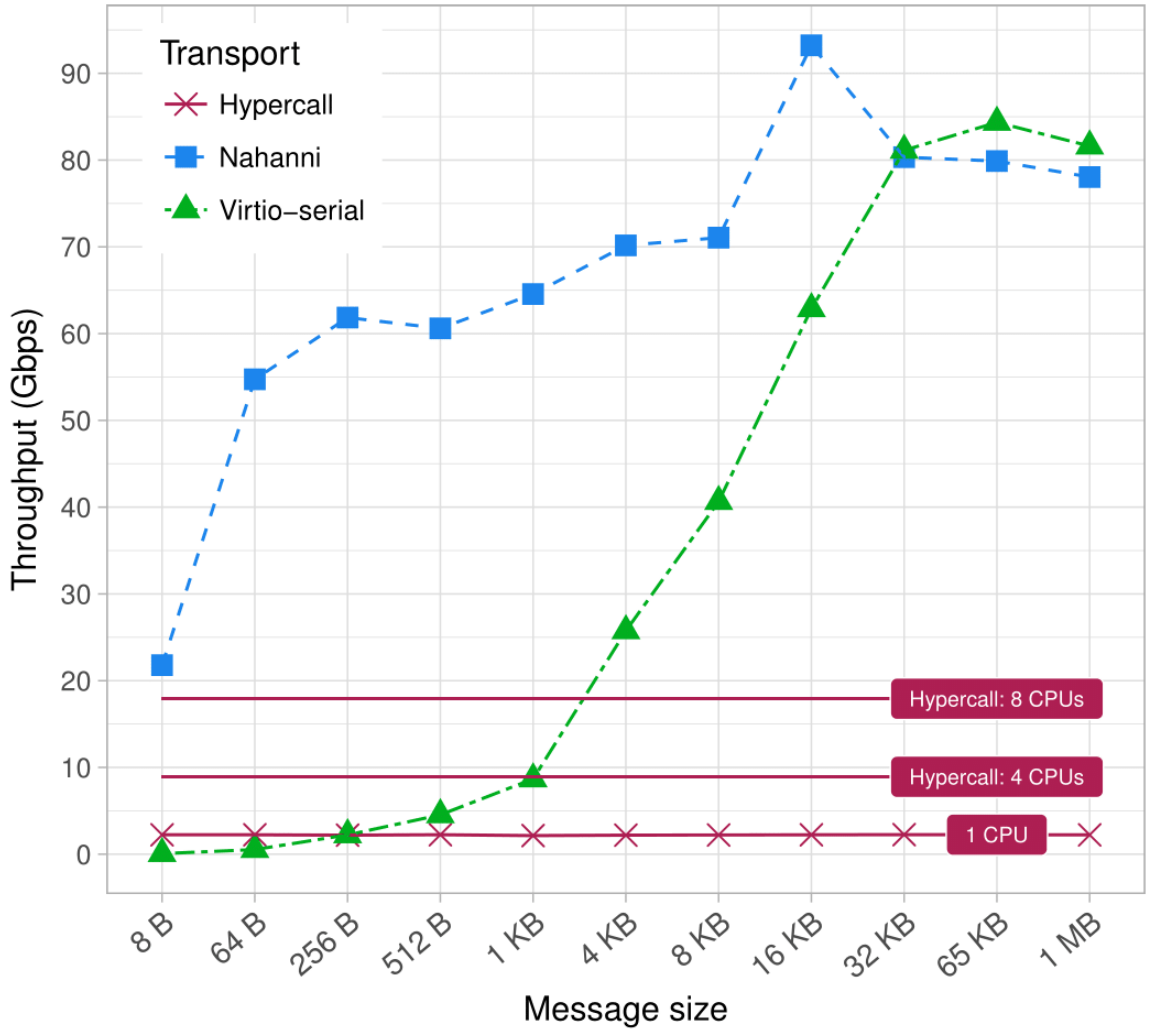


Memory Sharing Region

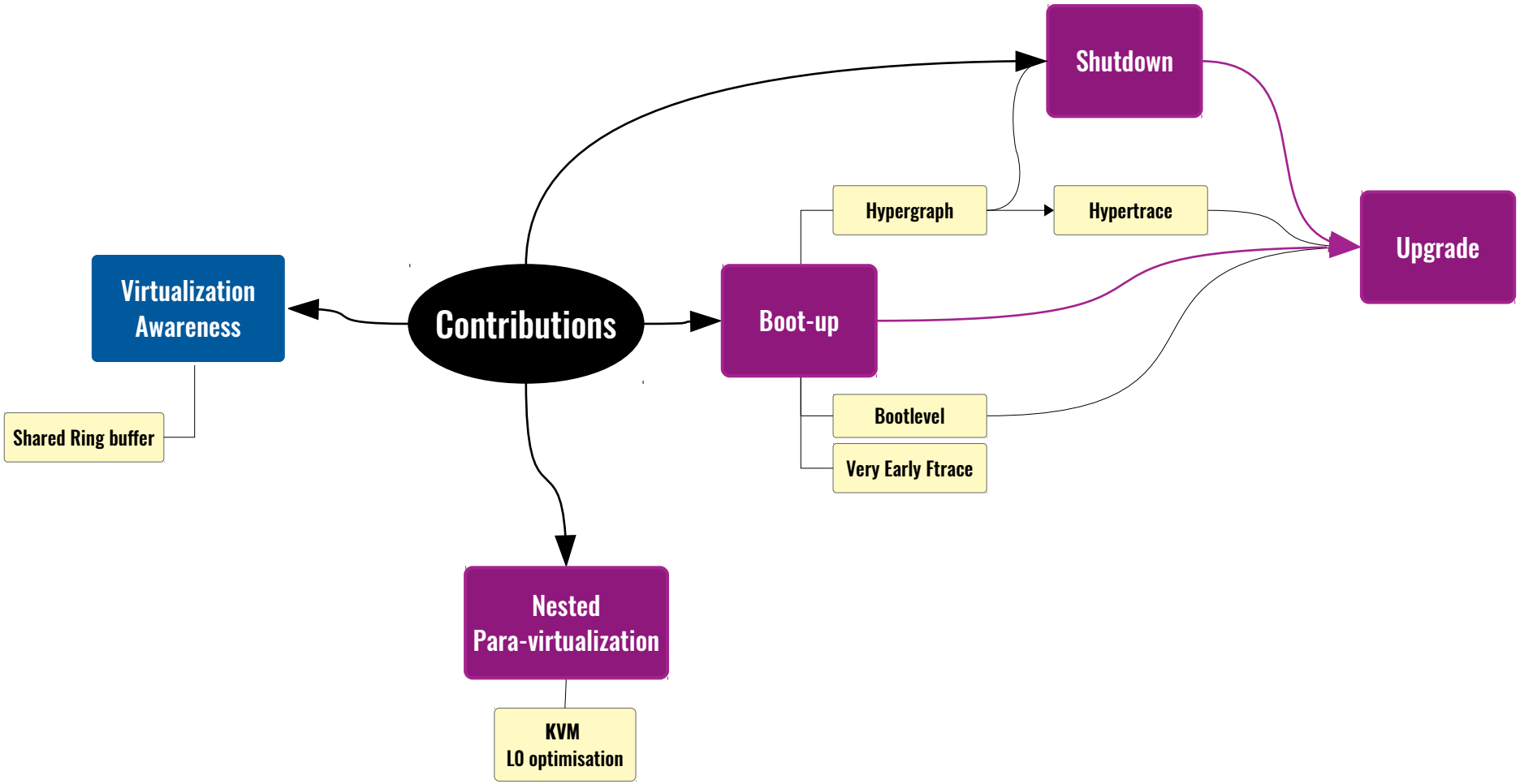
Nahanni



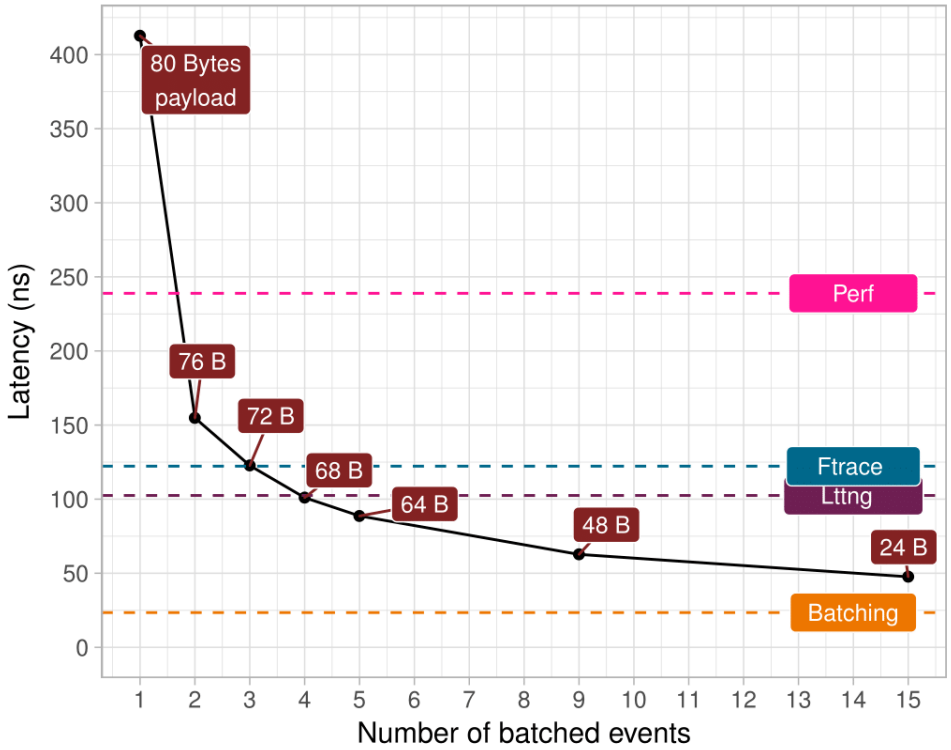
Performance Analysis

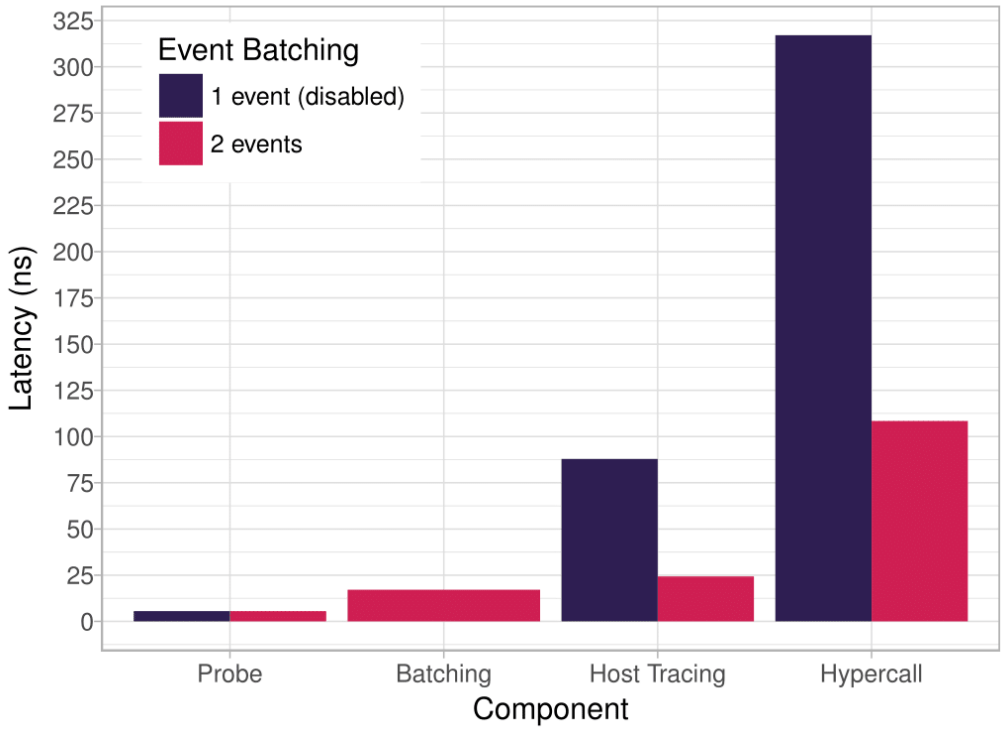


Contributions



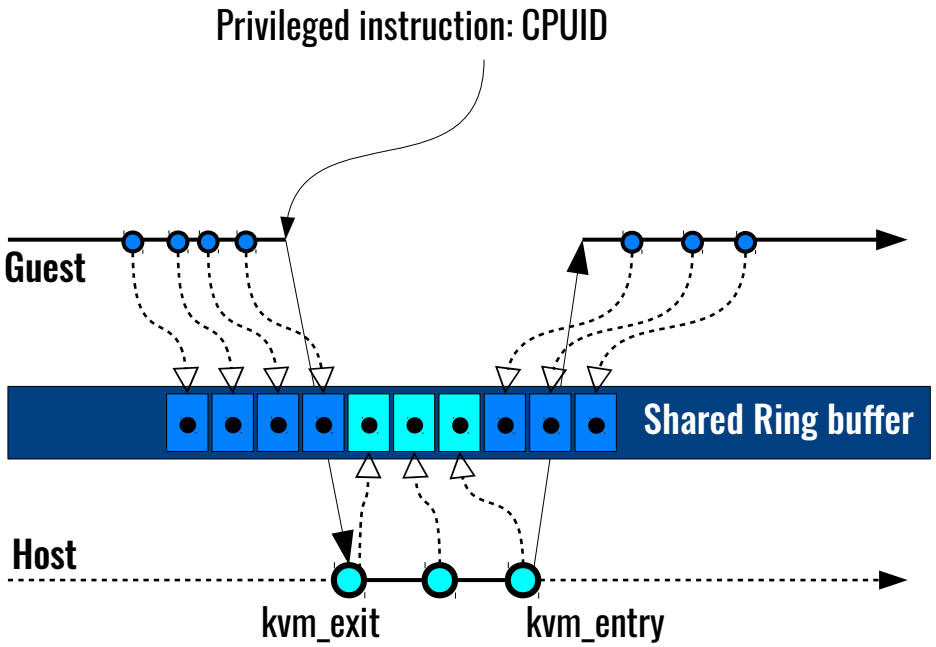
Tracing through Hypercall





Tracing through Shared Memory

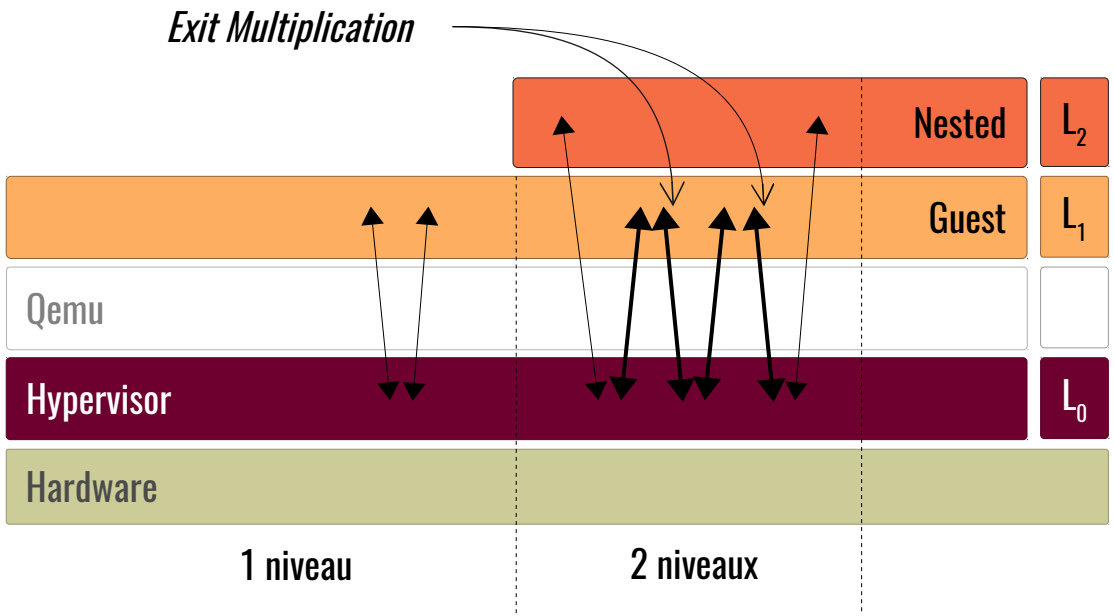
Per-vCPU shared buffers

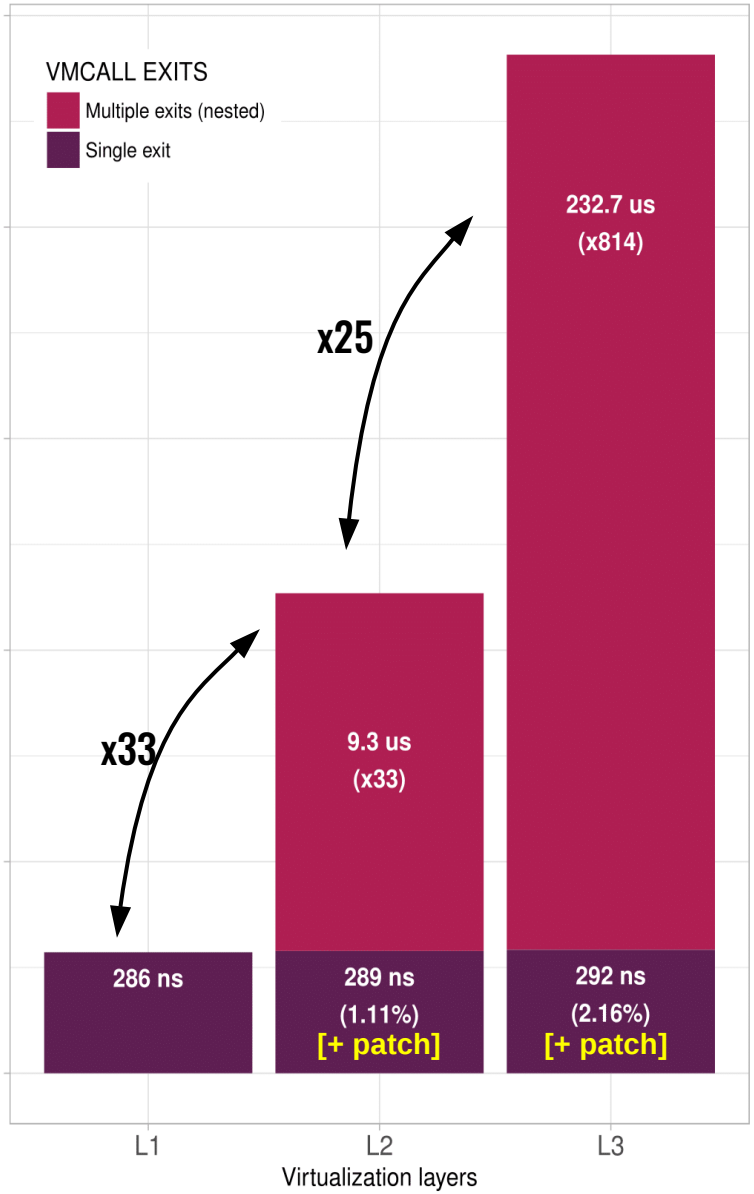


Nested Para-virtualization ?

Intel(R) Core(TM) i7-6700K CPU @ 4.00GHz (x86_64)

EPT-on-EPT-on-EPT





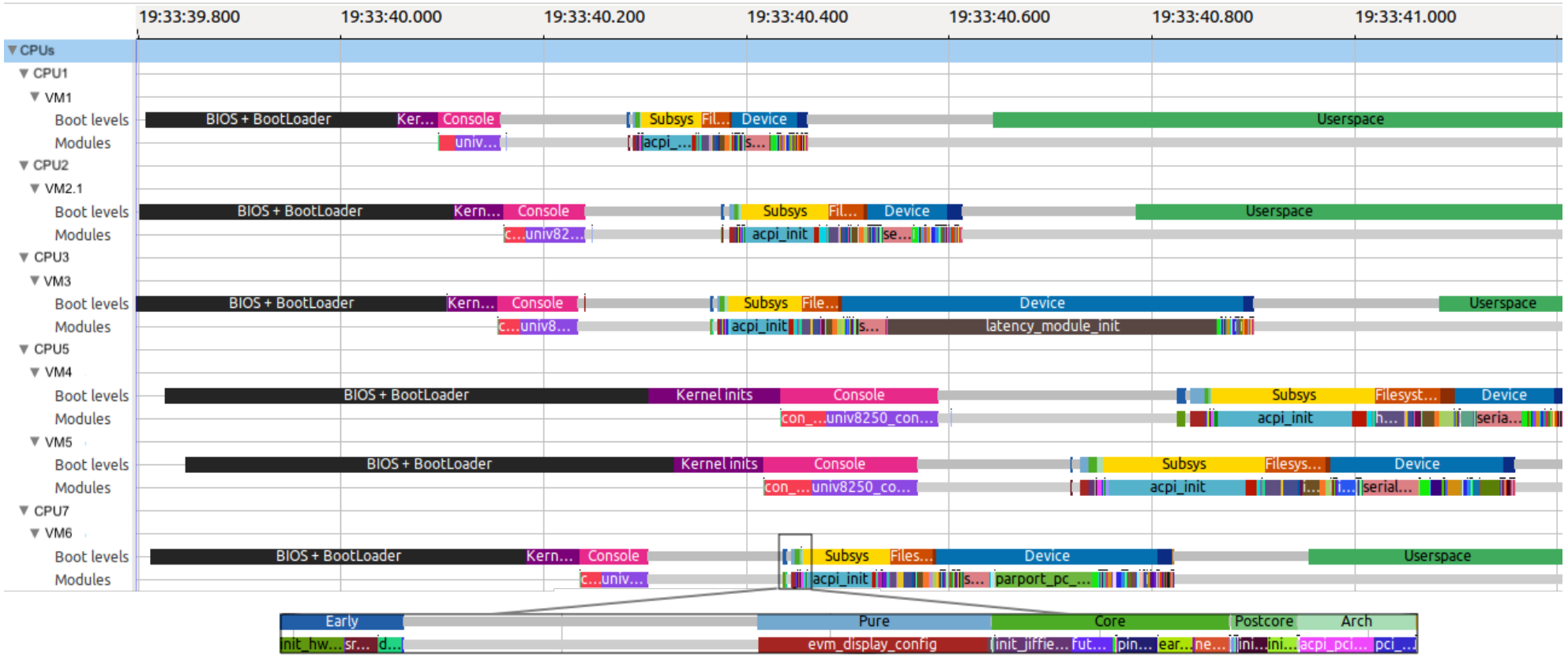
Use Cases

Boot-up

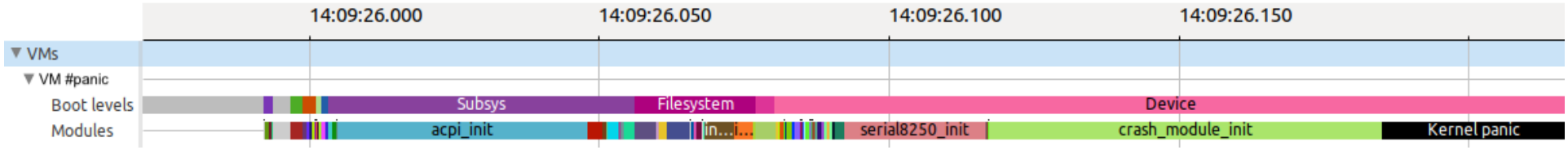
Shutdown

Virtualization Awareness

Boot-up



Tracing VMs boot-up levels and built-in modules.

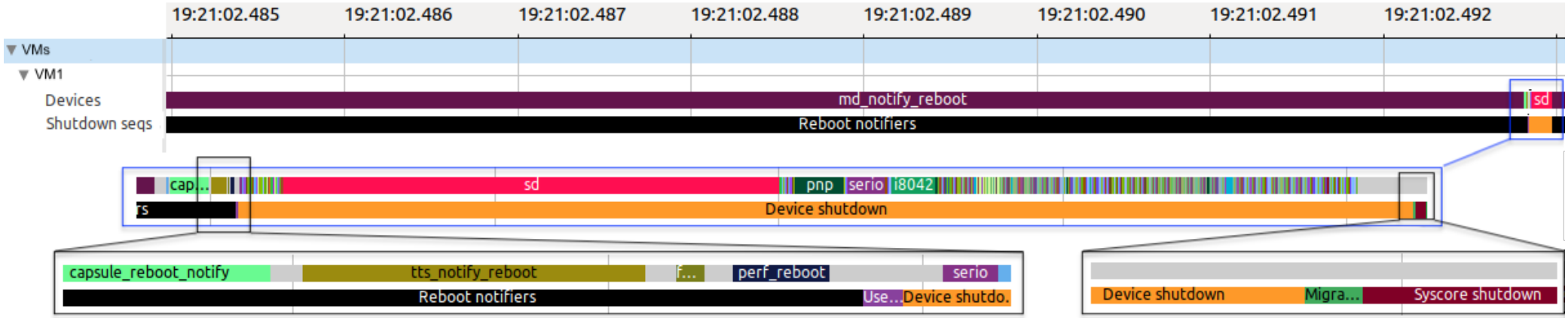


Detecting kernel panics happening during a VM boot-up.

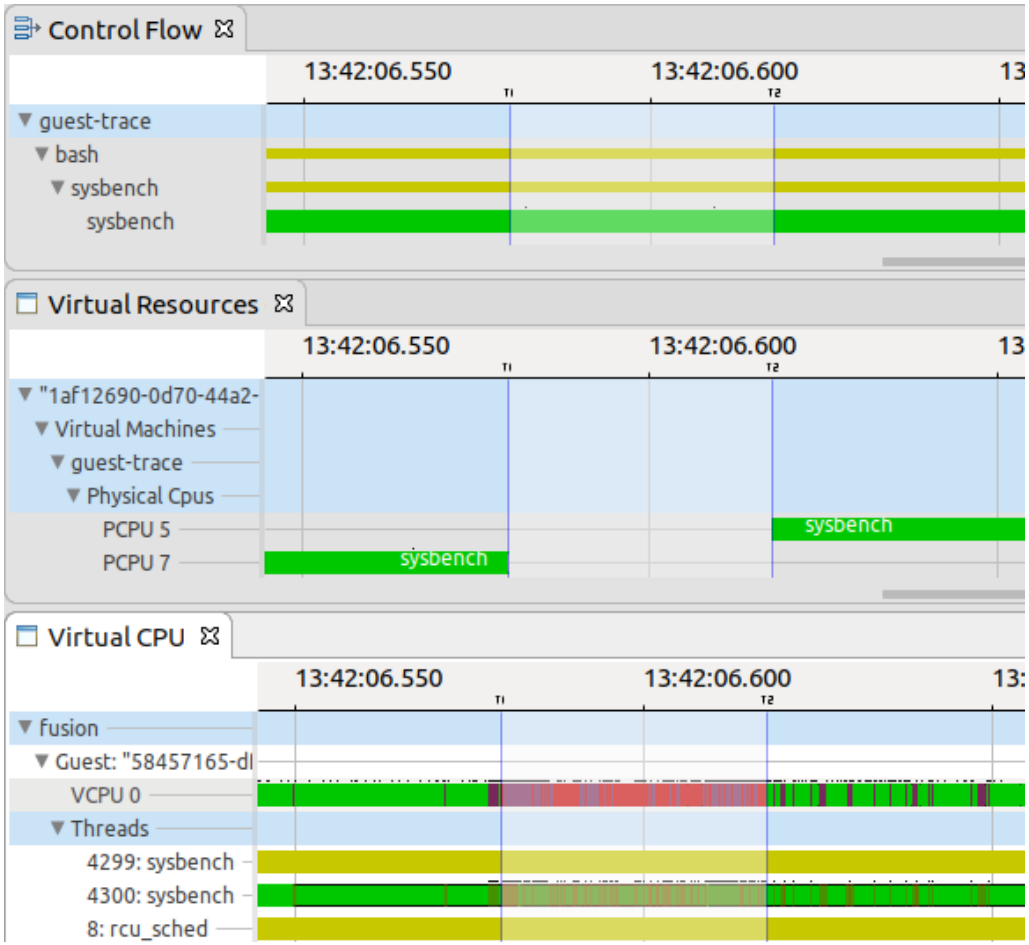
Shutdown

```
static int md_notify_reboot(struct notifier_block *this,
                            unsigned long code, void *x)
{
        struct list_head *tmp;
        struct mddev *mddev;
        int need_delay = 0;

        for_each_mddev(mddev, tmp) {
                if (mddev_trylock(mddev)) {
                        if (mddev->pers)
                                __md_stop_writes(mddev);
                        if (mddev->persistent)
                                mddev->safemode = 2;
                        mddev_unlock(mddev);
                }
                need_delay = 1;
        }
        /*
         * certain more exotic SCSI devices are known to be
         * volatile wrt too early system reboots. While the
         * right place to handle this issue is the given
         * driver, we do want to have a safe RAID driver ...
         */
        if (need_delay)
                mdelay(1000*1);

        return NOTIFY_DONE;
}
```

1s delay



vCPU Migration Detection

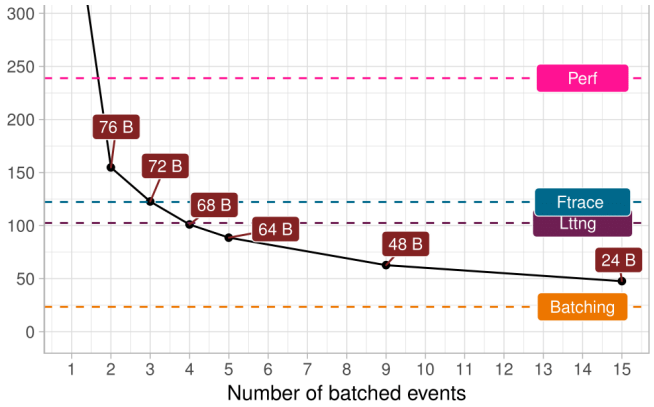


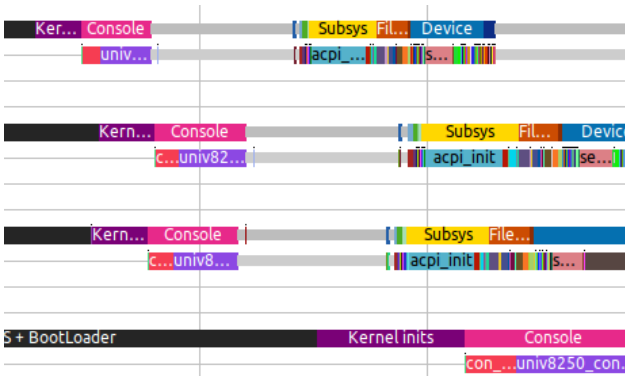
Using Shared Memory btw guest and host

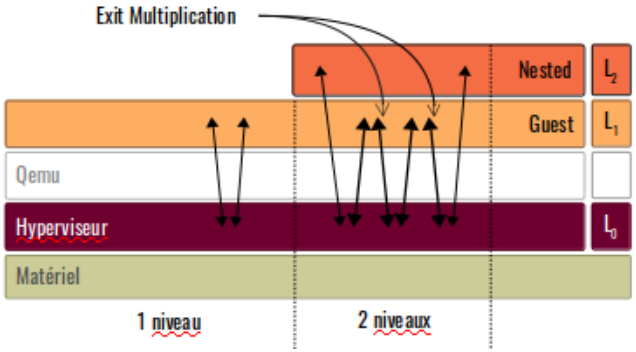
Overhead Analysis

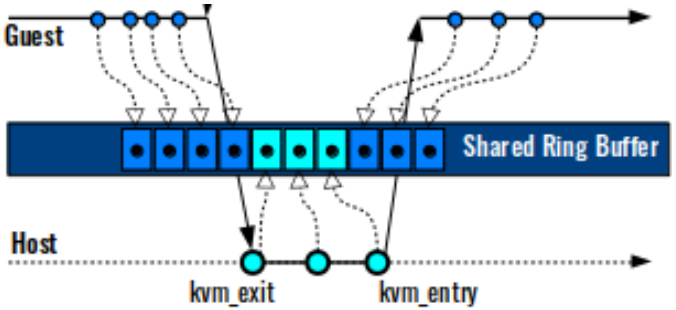
Boot-up tracing	Time (ms)	Overhead (%)
Baseline	734.43	-
Bootlevel	737.67	0.44
Hypergraph (initcalls)	738.28	0.52
Bootgraph (initcalls)	740.45	0.81
Ftrace function_graph (initcalls)	743.06	1.17

Benchmark	Baseline	Multi-level tracing	Hypertracing			Overhead (%)			
			Shared memory	Hypercall	Batching	Multi-level	Shared memory	Hypercall	Batching
File I/O (ms)	52.380	58.220	55.960	72.714	65.580	11.15	6.83	38.82	25.2
Memory (ms)	525.788	538.544	537.530	540.368	537.118	2.42	2.23	2.77	2.15
CPU (ms)	1380.34	1428.076	1421.426	1430.085	1426.404	3.45	2.97	3.6	3.33









Questions

abderrahmane.benbachir@polymtl.ca

https://github.com/abenbachir

References

Hypercall Implementation : https://gist.github.com/abenbachir/344822b5ba9fc5ac384cdec3f087e018

QEMU Hypertrace Patches: http://patchwork.ozlabs.org/project/qemu-devel/list/?state=&q=Hypertrace&archive

Trampoline: https://www.kernel.org/doc/ols/2009/ols2009-pages-47-54.pdf

Callstack xml analysis: https://gist.github.com/abenbachir/e813790f183945b6f74dc74ecee57c75