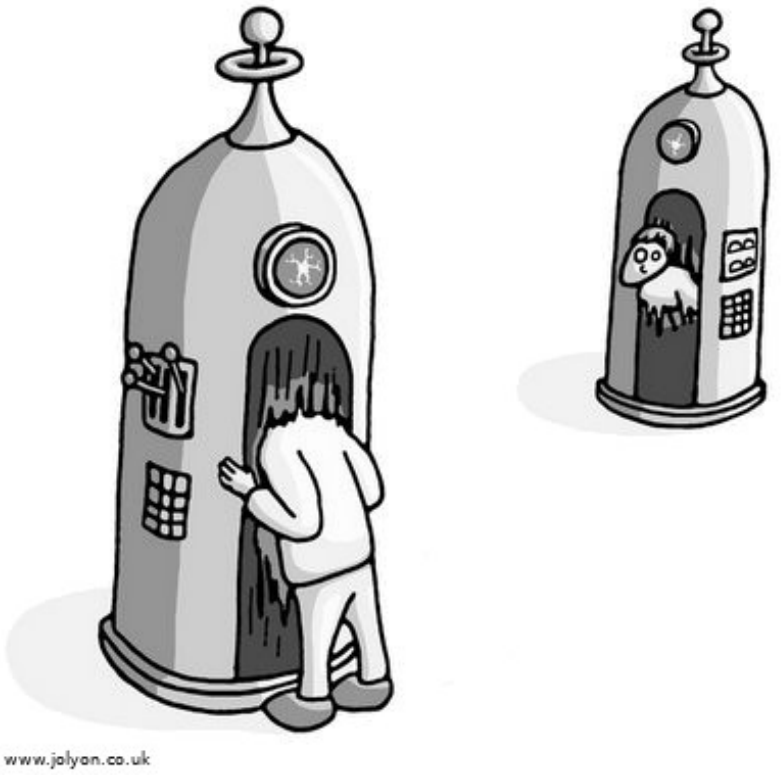


Tracing in virtualized environments

Lucian Carata, Domagoj Stolfa

Which tracing?

- Dynamic tracing tools such as DTrace
- Tools that continuously provide information about the underlying system with low overhead

- What doesn’t apply
  - Debuggers
  - Intercepting and high overhead tools (truss, strace)
  - Post-mortem tools (ktrace)

Difficulties

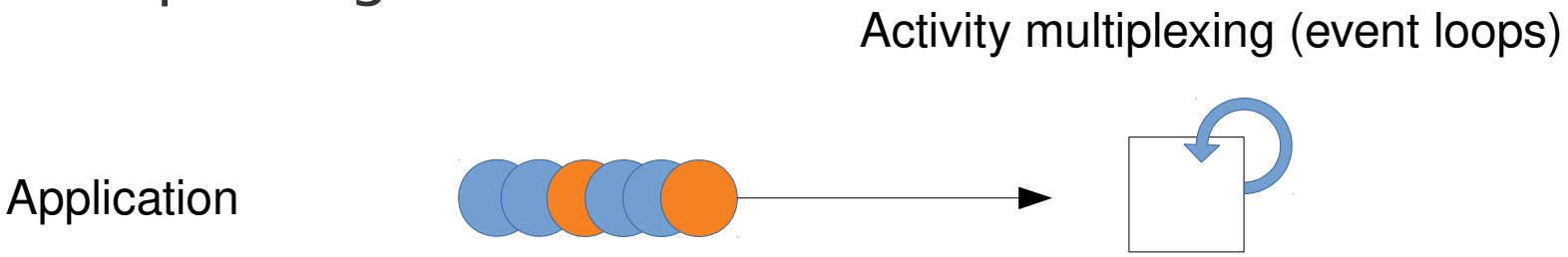
- The hypervisor - yet another layer of resource multiplexing
  - vCPUs, physical to machine memory mapping
- Problematic reading of hardware MSRs, PMU data
  - In the presence of save/restore and migration
- Complexity of overlayed stacks (2 schedulers, 2 TCP stacks, 2 I/O stacks...)

Difficulties

- The hypervisor - yet another layer of resource multiplexing

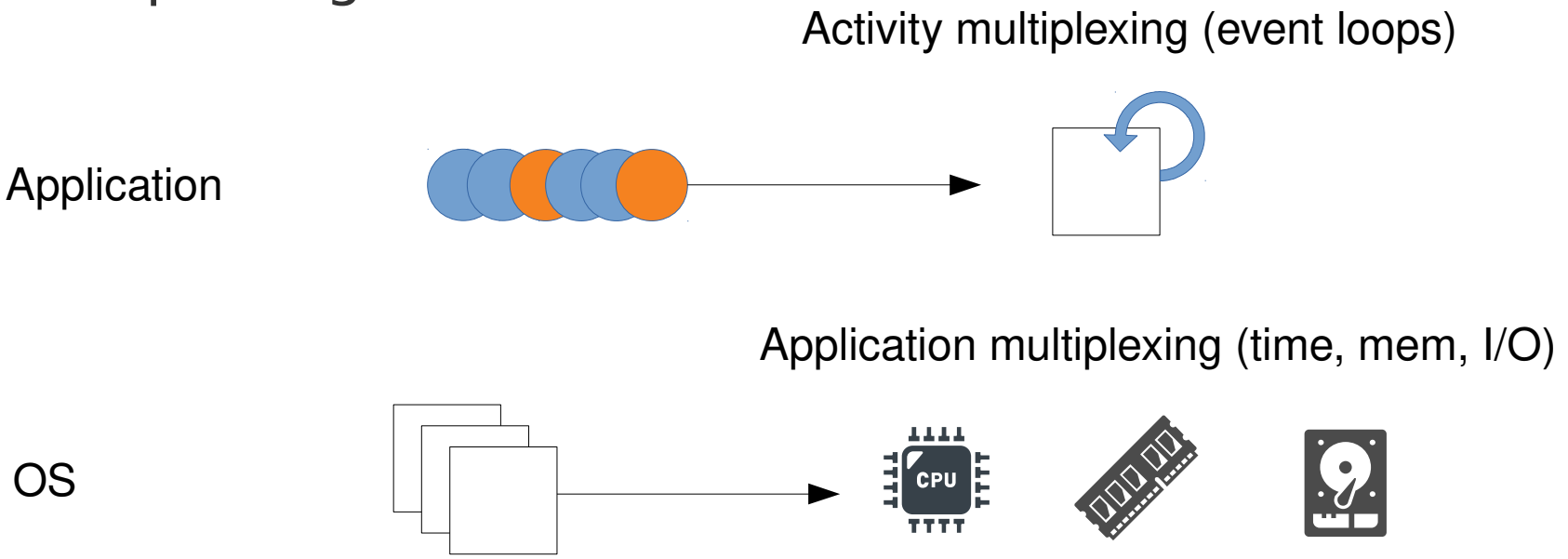
Difficulties

- The hypervisor - yet another layer of resource multiplexing



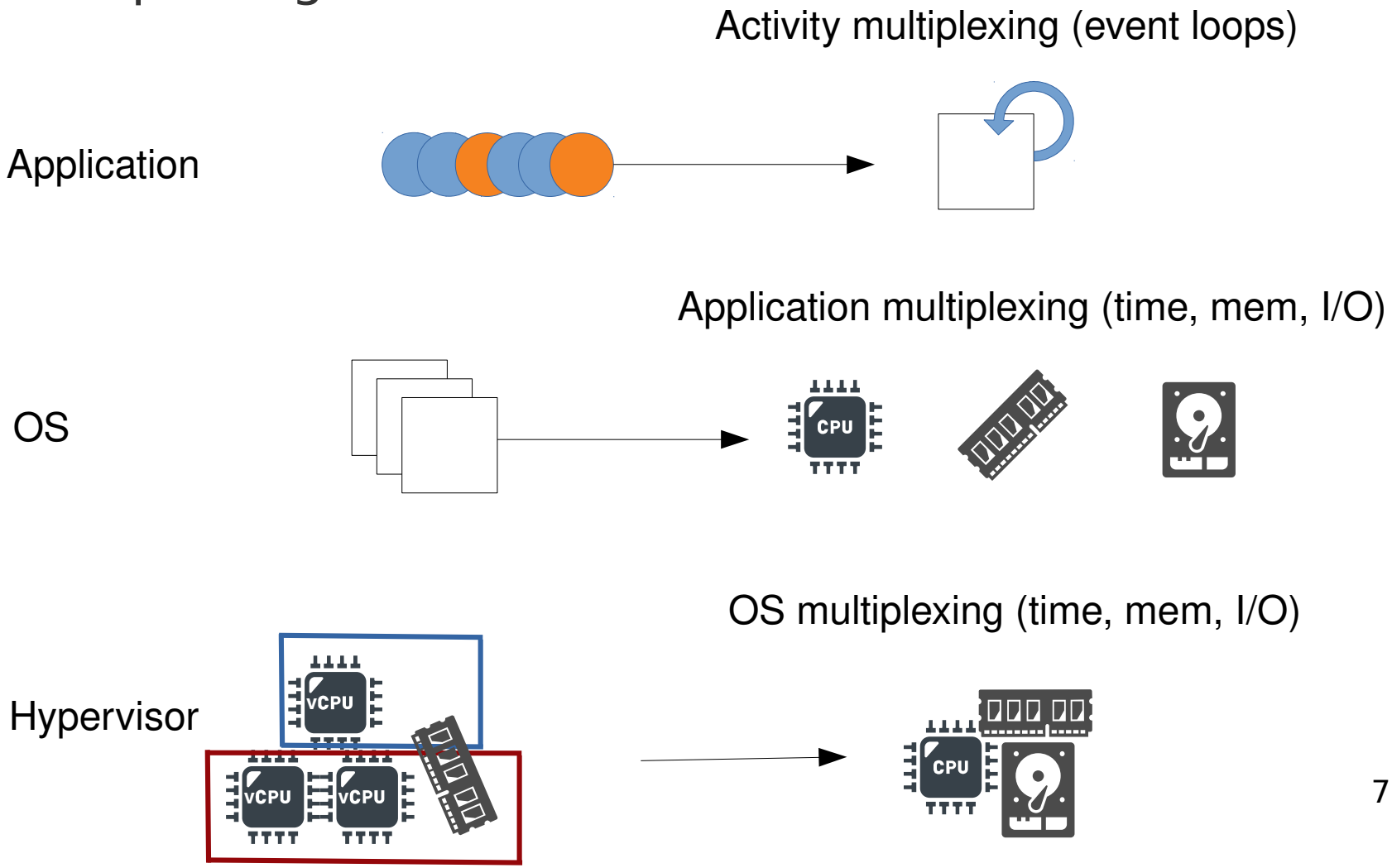
Difficulties

- The hypervisor - yet another layer of resource multiplexing



Difficulties

- The hypervisor - yet another layer of resource multiplexing



Difficulties

- Problematic reading of hardware MSRs, PMU data
  - In the presence of save/restore and migration

Difficulties

- Problematic reading of hardware MSRs, PMU data
  - In the presence of save/restore and migration

- Time (tsc)

```
constant_tsc + nonstop_tsc = invariant TSC
```

  - Defaults in hypervisors:
    - slow but always correct
    - fast but sometimes wrong

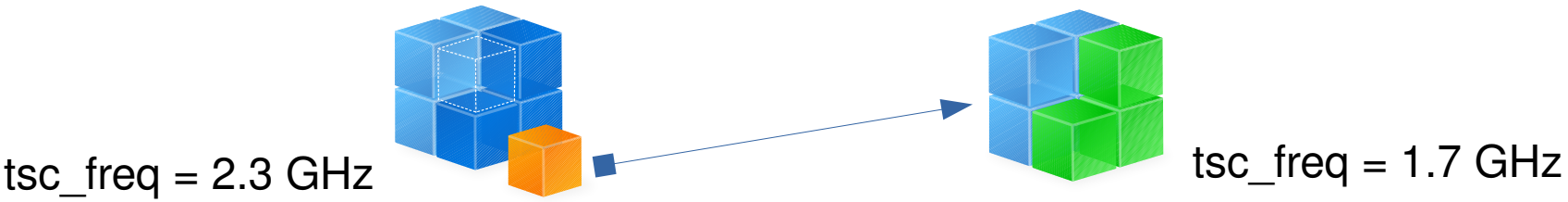
Difficulties

- Problematic reading of hardware MSRs, PMU data
  - In the presence of save/restore and migration

- Time (tsc)

```
constant_tsc + nonstop_tsc = invariant TSC
```

  - Defaults in hypervisors:
    - slow but always correct
    - fast but sometimes wrong



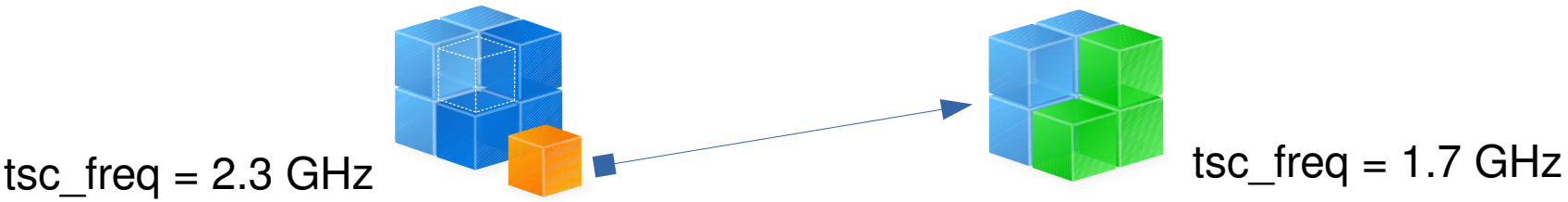
Difficulties

- Problematic reading of hardware MSRs, PMU data
  - In the presence of save/restore and migration

- Time (tsc)

```
constant_tsc + nonstop_tsc = invariant TSC
```

  - Defaults in hypervisors:
    - slow but always correct
    - fast but sometimes wrong



  - Option: paravirtualized `rdtscp`

Difficulties

- Complexity of overlayed stacks (2 schedulers, 2 TCP stacks, 2 I/O stacks...)

Difficulties

- Complexity of overlayed stacks (2 schedulers, 2 TCP stacks, 2 I/O stacks…)
  - If you measure anything that is faster in the VM than on bare metal….

Difficulties

- Complexity of overlayed stacks (2 schedulers, 2 TCP stacks, 2 I/O stacks…)
  - If you measure anything that is faster in the VM than on bare metal….
  - You’re likely not measuring all of it

DTrace-virt

- What it is
  - Currently utilizes bhyve exclusively
  - Currently DTrace only.
  - Provides global execution state across guests.
  - Synchronous.

- What it is not.
  - A replacement for debugging tools.
  - A way to inspect compromised guests.

DTrace-virt

- Possible applications
  - Monitoring tools.
  - Debugging virtualized systems.
  - Malware analysis.
  - Understanding how virtualized applications work and what the implications of virtualization are.

DTrace-virt

- Benefits
  - Given the execution state, conditionally do various things through DTrace.
  - Implied time ordering of events due to synchronous behaviour.
  - Inspection of state across guest machines.

- Pitfalls.
  - Requires large changes to DTrace.
  - Requires a synchronous trapping mechanism.
  - Potentially high overhead.

Discussion

- Harder problems:
  - Security model for tracing multiple guests
  - Malicious guests
  - What measurements need guest/hypervisor coordination?
  - Generalization of the approach to multiple types of systems (hypervisors, processes, kernels, …)
    - Solutions that are applicable to distributed systems?