CRIU - Checkpoint/Restore in User-space

Parallels™

Profit from the Cloud

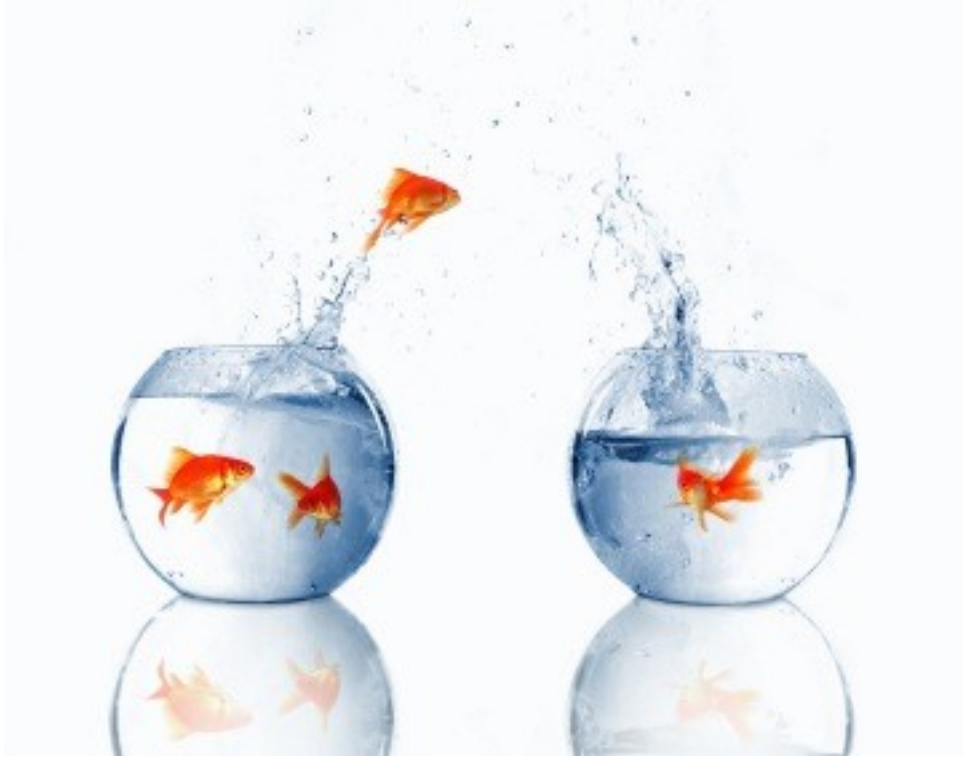
Andrey Vagin <avagin@openvz.org>

What is C/R and how can it be used?

C/R is the ability to save states of processes and to restore them later.

Usage scenarios:

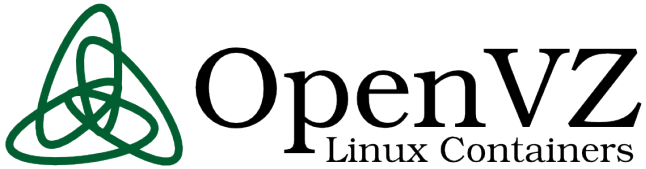
- Failure recovery
- Live migration
- Reboot-less upgrade
- Speed up of slow-boot services
- HPC issues



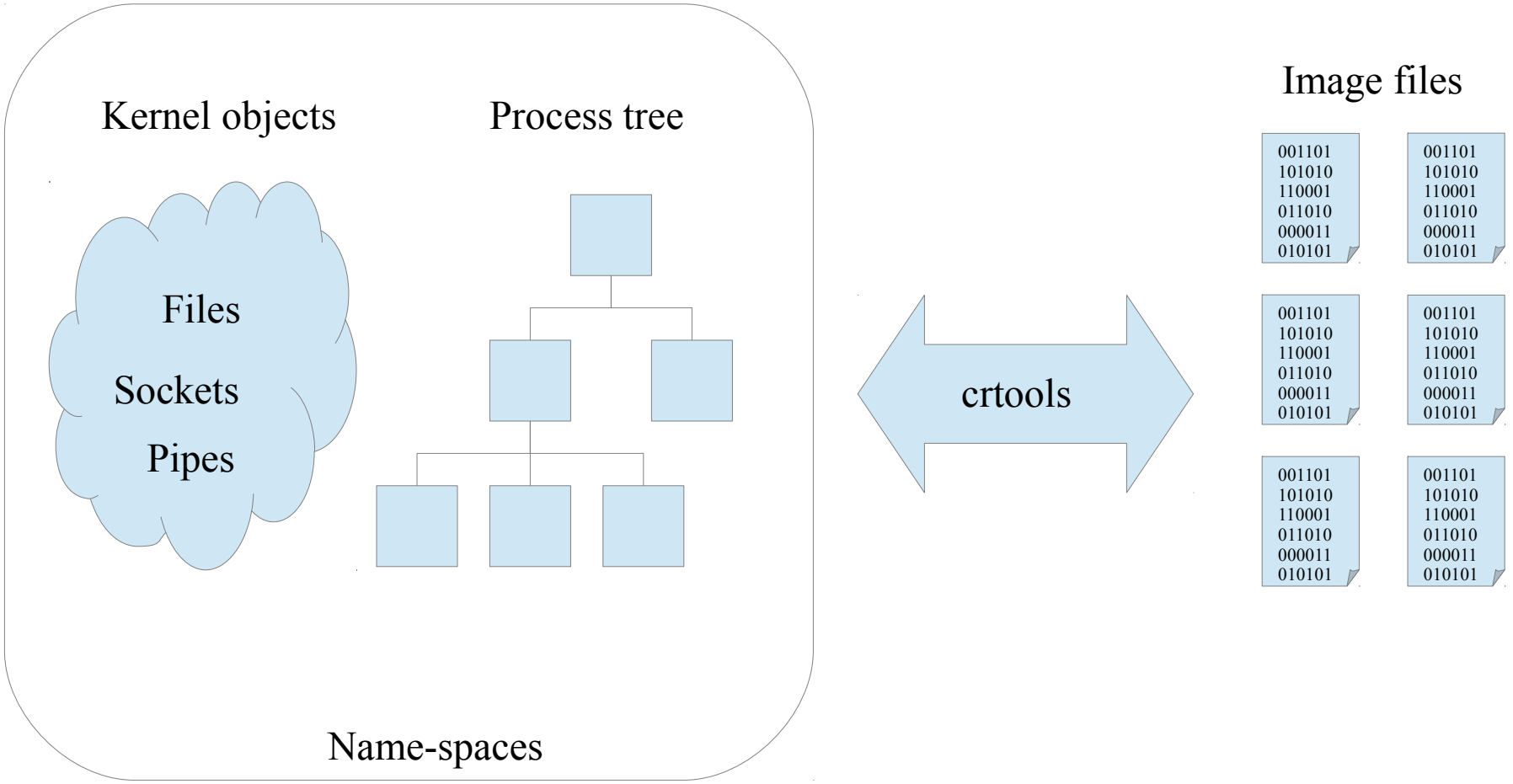
History

BLCR	DMTCP	OpenVZ	Linux C/R	CRIU
2003	2007	2005	2008	2011

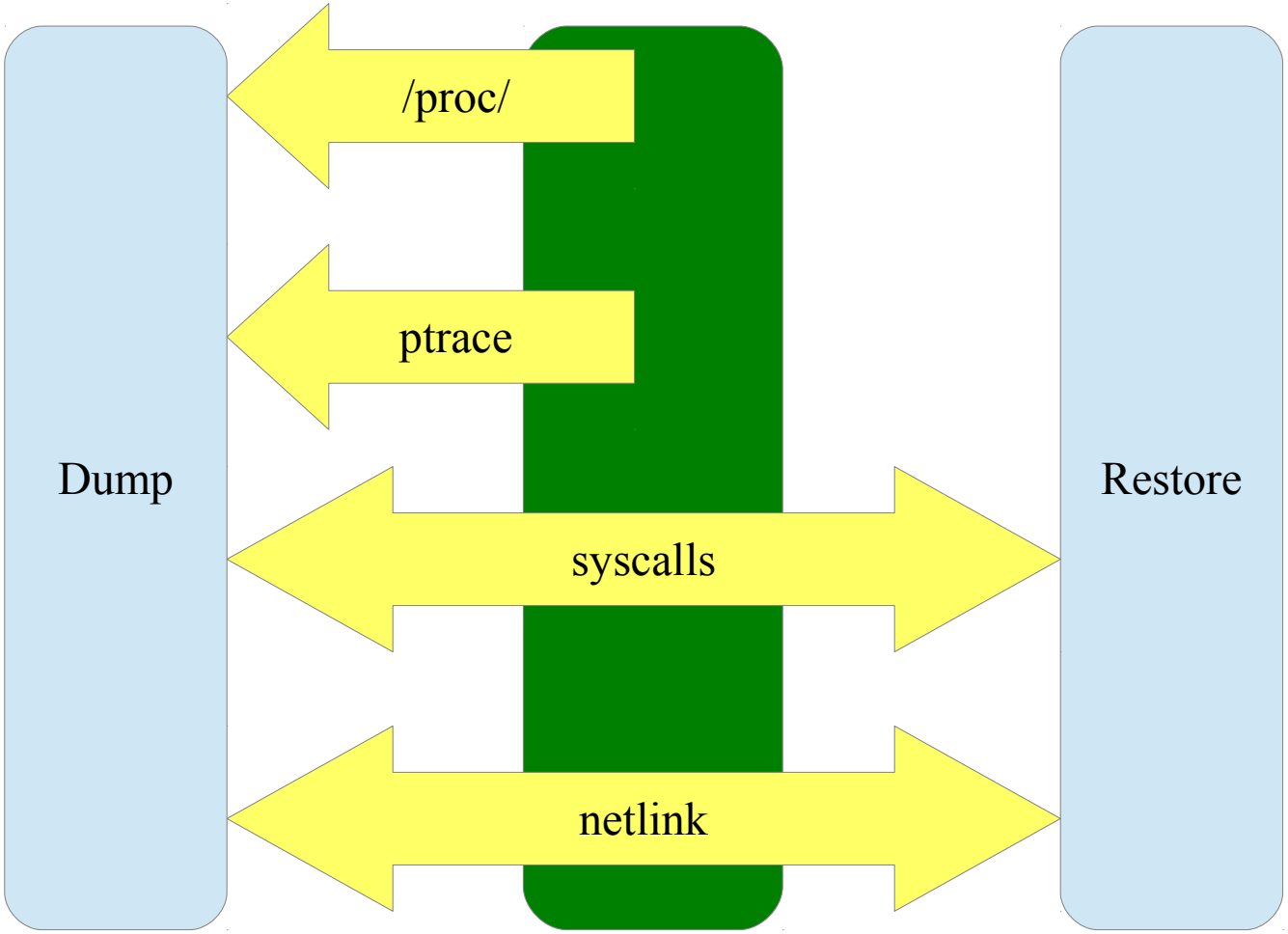
- Berkeley Lab Checkpoint/Restart (BLCR) (2003)
  - Load a kernel module and link with a library
- DMTCP: Distributed MultiThreaded CheckPointing (2004-2006)
  - Preload a library
- OpenVZ (2005)
  - OpenVZ kernel
- Linux Checkpoint/Restart by Oren Laadan (2008)
  - A non-mainline kernel
- CRIU (2011)



How does this work?



Kernel interfaces



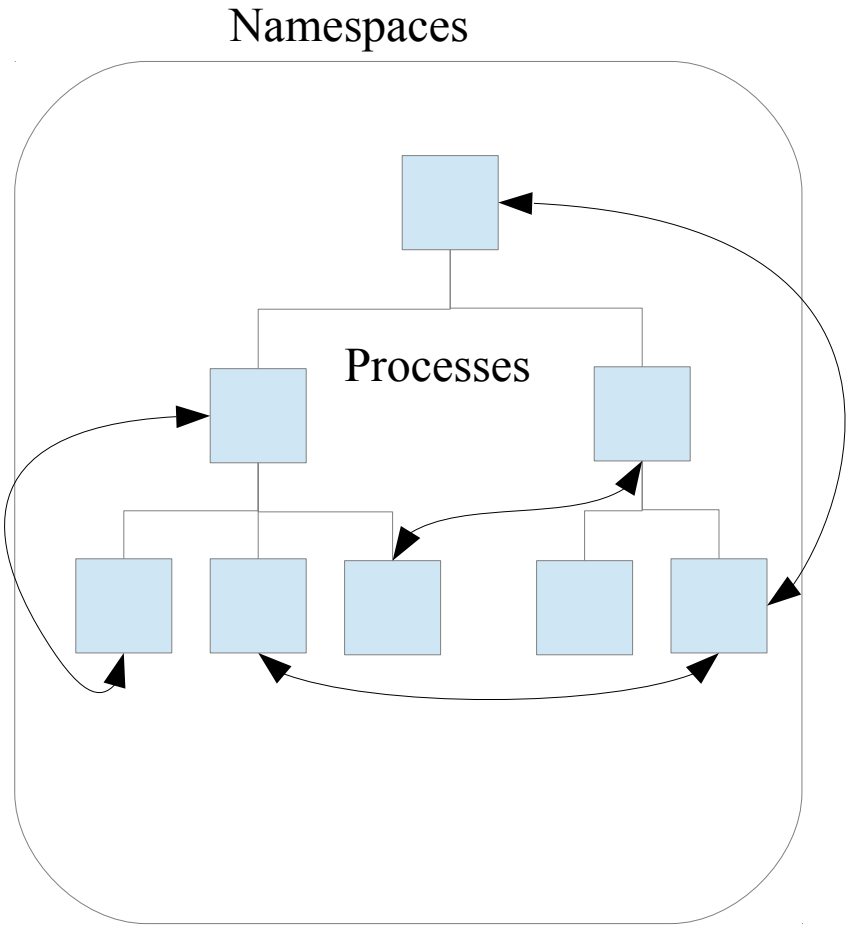
Dump

- **Parasite code**
  - Receive file descriptors
  - Dump memory content
  - Prctl(), sigaction, pending signals, timers, etc.
- Ptrace
  - freeze processes
  - Inject a parasite code
- Netlink
  - Get information about sockets, netns
- Procfs

  /proc/PID/maps, /proc/PID/map_files/, /proc/PID/status, /proc/PID/mountinfo

Restore

- Collect shared objects
- Restore name-spaces
- Create a process tree
  - Restore SID, PGID
  - Restore objects, which should be inherited
- Files, sockets, pipes, ...
- Restore per-task properties.
- Restore memory
- Call sigreturn
- Awesome

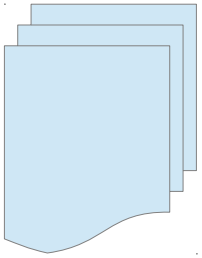


Interesting moments

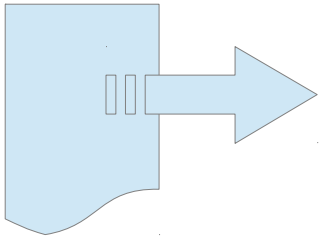
- How to restore shared objects?
  - Send file descriptors via unix sockets
  - Map files from /proc/self/map_files/ for restoring anon shared mappings
- How to restore memory mappings on the correct places?
  - Map a new code block and a stack
  - Unmap crtools' mappings
  - Remap task's mappings on the correct places
- How to resume a process?
  - Create a signal frame
  - Call sigreturn()

Kernel impact

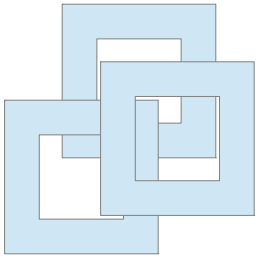
~140 patches merged



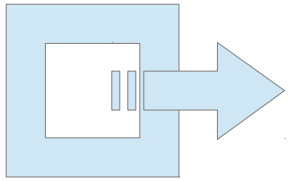
~10 patches in flight



~11 new features appeared



~2 new features to come



New features in a kernel

- Parasite code injection (by Tejun Heo)
  - Read task states, that are currently retrieved by a task only about itself
- The kcmp() system call
  - *Helps checking which kernel objects are shared between processes*
- Proc map_files directory
  - *Find out what exact file is mapped*
  - *Mappings sharing info*
- A bunch of prctl extensions
  - *Set various private stuff on task/mm objects (c/r-only feature)*
- Last-pid sysctl
  - *Restore task with desired PID value*

New features in a kernel

- TCP repair mode
  - Read intimate state of a TCP connection
    and reconstructs it from scratch on a freshly created socket
- Sockets information dumping via netlink (sock_diag)
  - Extendable sockets state retrieving engine
- Virtual net devices indexes
  - Allows to restore network devices in a namespace
- Socket peeking offset
  - Allows peeking sockets queues (reading without removing data from queue)
- Task memory tracking
  - incremental snapshots, online migration

What are already supported?

- X86_64 architecture
- Process tree linkage
- Multi-threaded apps
- All kinds of memory mappings
- Terminals, groups, sessions
- Open files (shared and unlinked)
- Established TCP connections
- Unix sockets, Packet sockets
- Name-spaces (net, mount, ipc)
- Non-posix files (epoll, inotify)
- Pipes, Fifo-s, IPC, ...

- ARM architecture
- Pending signals
- Iterative snapshots
- VDSO
- LXC and OpenVZ containers

In flight

- Posix timers
- Convert OpenVZ images

How is CRIU tested?

- ZDTM – a set of unit-tests
- Real-life applications
  - Apache, Nginx
  - MySQL, MongoDB, Oracle
  - Make && gcc
  - Tar & gzip
  - Screen
  - Java
  - LXC
  - VNC server + GUI applications

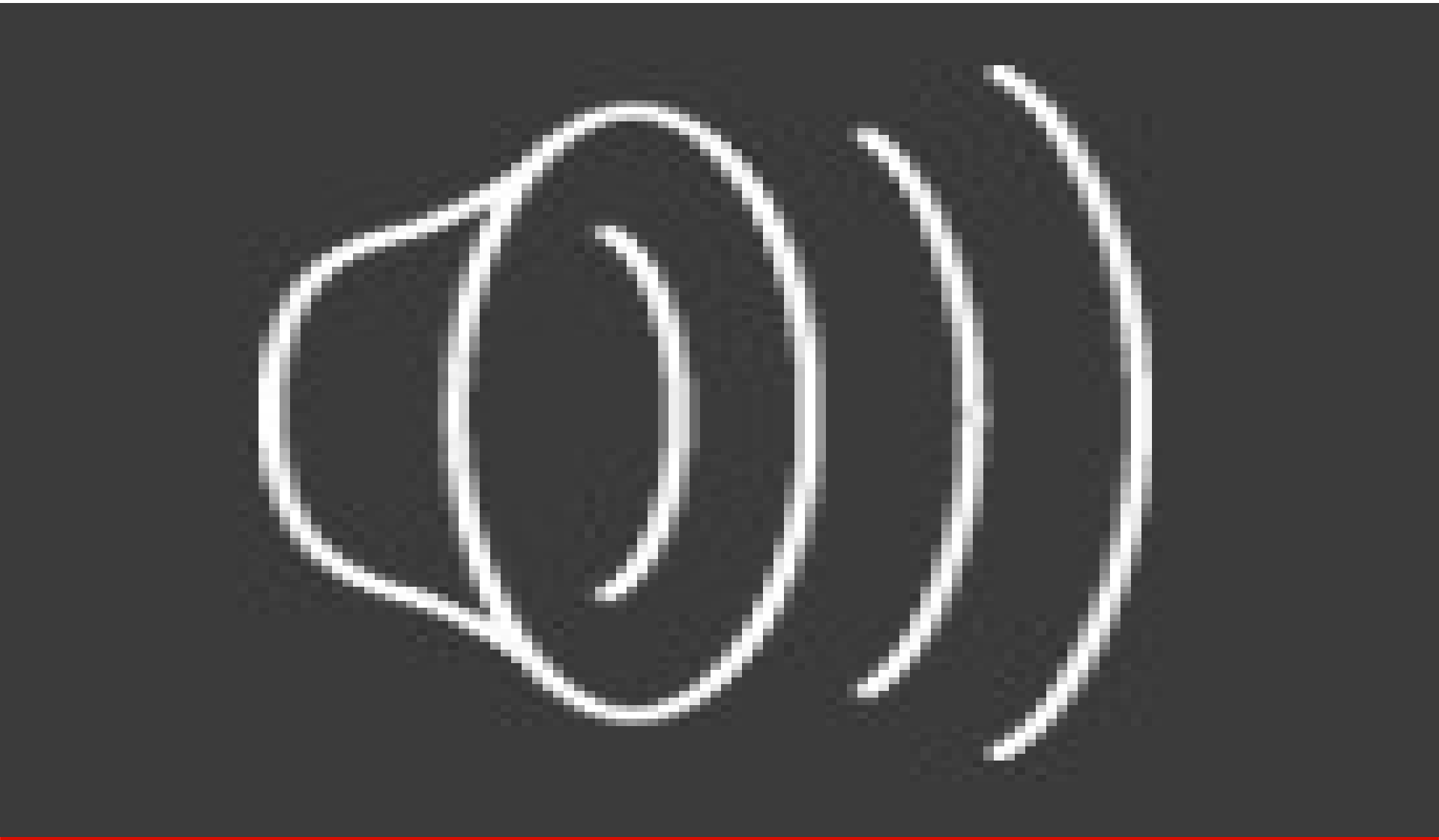
Future plans (Feb, 2013)

- Support all kinds of kernel objects
- Merge all in-flight patches in the mainstream kernel
- Integrate CRIU with OpenVZ and LXC utilities
- Iterative migration
  - Migrate memory content before freezing applications
- Integration in distributions
  - CRIU was accepted to Fedora 19

How to use

- ./crtools dump -t pid [<options>]
  - checkpoint a process/tree identified by pid
- ./crtools restore -t pid [<options>]
  - restore - restore a process/tree identified by pid
- ./crtools show (-D dir)|(-f file) [<options>]
  - show dump file(s) contents
- ./crtools check
  - checks whether the kernel support is up-to-date
- ./crtools exec -t pid <syscall-string>
  - exec - execute a system call by other task

Checkpoint/restore of a VNC server.



Questions?

http://criu.org