Fun with LD_PRELOAD

Kevin Pulo
kev@pulo.com.au

Australian National University Supercomputing Facility,
National Computational Infrastructure National Facility,
Canberra, Australia

2009-01-23

Overview

1. Intro: Dynamic linking and LD_PRELOAD
2. Review: Existing LD_PRELOAD hacks
3. Code: Writing LD_PRELOAD hacks
4. Advanced: xlibtrace and xmultiwin

Part 1

1. Intro: Dynamic linking and LD_PRELOAD

2. Review: Existing LD_PRELOAD hacks

3. Code: Writing LD_PRELOAD hacks

4. Advanced: xlibtrace and xmultiwin

$LD_PRELOAD?

- Ordinarily the dynamic linker loads shared libs in whatever order it needs them

$LD_PRELOAD?

- Ordinarily the dynamic linker loads shared libs in whatever order it needs them
- **`$LD_PRELOAD` is an environment variable containing a colon (or space) separated list of libraries that the dynamic linker loads before any others**

$LD_PRELOAD?

- Ordinarily the dynamic linker loads shared libs in whatever order it needs them
- **`$LD_PRELOAD` is an environment variable containing a colon (or space) separated list of libraries that the dynamic linker loads before any others**
- Entries containing '/' are treated as filenames
- Entries not containing '/' are searched for as usual

$LD_PRELOAD?

- Ordinarily the dynamic linker loads shared libs in whatever order it needs them
- **`$LD_PRELOAD` is an environment variable containing a colon (or space) separated list of libraries that the dynamic linker loads before any others**
- Entries containing '`/`' are treated as filenames
- Entries not containing '`/`' are searched for as usual
- eg. `LD_PRELOAD="libc.so.6"`
  `LD_PRELOAD="/test/lib/libc.so.6"`

$LD_PRELOAD?

- Ordinarily the dynamic linker loads shared libs in whatever order it needs them
- **`$LD_PRELOAD` is an environment variable containing a colon (or space) separated list of libraries that the dynamic linker loads before any others**
- Entries containing `'/'` are treated as filenames
- Entries not containing `'/'` are searched for as usual
- eg. `LD_PRELOAD="libc.so.6"`
  `LD_PRELOAD="/test/lib/libc.so.6"`
- `man ld.so` for full dynamic linker info

Okay — so what?

- Preloading a library means that its functions will be used before others of the same name in later libraries

Okay — so what?

- Preloading a library means that its functions will be used before others of the same name in later libraries
- Allows functions to be overridden/replaced/ intercepted

Okay — so what?

- Preloading a library means that its functions will be used before others of the same name in later libraries
- Allows functions to be overridden/replaced/intercepted
- Program behaviour can be modified “non-invasively”

Okay — so what?

- Preloading a library means that its functions will be used before others of the same name in later libraries
- Allows functions to be overridden/replaced/intercepted
- Program behaviour can be modified “non-invasively”
  - ie. no recompile/relink necessary

Okay — so what?

- Preloading a library means that its functions will be used before others of the same name in later libraries
- Allows functions to be overridden/replaced/intercepted
- Program behaviour can be modified “non-invasively”
  - ie. no recompile/relink necessary
  - Especially useful for closed-source programs

Okay — so what?

- Preloading a library means that its functions will be used before others of the same name in later libraries
- Allows functions to be overridden/replaced/intercepted
- Program behaviour can be modified “non-invasively”
  - ie. no recompile/relink necessary
  - Especially useful for closed-source programs
  - And when the modifications don’t belong in the program or the library

System-wide

- `export LD_PRELOAD="foobar.so"` in /etc/profile
  (`unset LD_PRELOAD`)
- `/etc/ld.so.preload`
  (loaded before `$LD_PRELOAD`, non-overridable)

System-wide

- `export LD_PRELOAD="foobar.so"` in /etc/profile
  (`unset LD_PRELOAD`)
- `/etc/ld.so.preload`
  (loaded before `$LD_PRELOAD`, non-overridable)
- Affects *all* dynamically linked programs: be careful
- Consider selective use:

```
LD_PRELOAD="foobar.so" someprogram        (sh/bash)
env LD_PRELOAD="foobar.so" someprogram    (csh/any)
```

Issues

- Affects dynamically linked programs only - not static

Issues

- Affects dynamically linked programs only - not static
- Also affects child processes

Issues

- Affects dynamically linked programs only - not static
- Also affects child processes
- setuid/setgid programs: only libs in standard paths that are also setuid/setgid will be preloaded

Issues

- Affects dynamically linked programs only - not static
- Also affects child processes
- setuid/setgid programs: only libs in standard paths that are also setuid/setgid will be preloaded
- Can only override library functions, not system calls
  (glibc wrappers are fair game)

Issues

- Affects dynamically linked programs only - not static
- Also affects child processes
- setuid/setgid programs: only libs in standard paths that are also setuid/setgid will be preloaded
- Can only override library functions, not system calls
  (glibc wrappers are fair game)
- Options must be passed via environment
  (could still use getopt() etc to parse an env var)

Part 2

1. Intro: Dynamic linking and LD_PRELOAD

2. Review: Existing LD_PRELOAD hacks

3. Code: Writing LD_PRELOAD hacks

4. Advanced: xlibtrace and xmultiwin

Existing LD_PRELOAD hacks

Vague broad categories:

- Filesystem shenanigans
- Network shenanigans
- Debugging and testing
- Annoyance reduction
- Library tracing and logging
- Graphical augmentation
- Finessing/augmenting program behaviours

Filesystem shenanigans

By intercepting libc functions that deal with the filesystem, the preload library can modify how the filesystem looks and behaves for programs.

Filesystem shenanigans

By intercepting libc functions that deal with the filesystem, the preload library can modify how the filesystem looks and behaves for programs.

- Anything that's passed or returns filenames/paths
- eg. open(), mkdir(), opendir(), stat(), etc

Filesystem shenanigans

By intercepting libc functions that deal with the filesystem, the preload library can modify how the filesystem looks and behaves for programs.

- Anything that's passed or returns filenames/paths
- eg. open(), mkdir(), opendir(), stat(), etc
- **flcow (File Link Copy-On-Write)**: http://www.xmailserver.org/flcow.html
  Intercepts open() and friends, breaks hard-links when changing files.

flcow demo

$

flcow demo

```
$ ls -ld linux*
```

flcow demo

```
$ ls -ld linux*
drwxr-xr-x 21 4096 2009-01-17 15:37 linux-2.6.26.5
$
```

flcow demo

```
$ ls -ld linux*
drwxr-xr-x 21 4096 2009-01-17 15:37 linux-2.6.26.5
$ du -sk linux*
```

flcow demo

```
$ ls -ld linux*
drwxr-xr-x 21 4096 2009-01-17 15:37 linux-2.6.26.5
$ du -sk linux*
330592  linux-2.6.26.5
$
```

flcow demo

```
$ ls -ld linux*
drwxr-xr-x 21 4096 2009-01-17 15:37 linux-2.6.26.5
$ du -sk linux*
330592  linux-2.6.26.5
$ cp -al linux-2.6.26.5 linux-2.6.26.5-test
$
```

flcow demo

```
$ ls -ld linux*
drwxr-xr-x 21 4096 2009-01-17 15:37 linux-2.6.26.5
$ du -sk linux*
330592  linux-2.6.26.5
$ cp -al linux-2.6.26.5 linux-2.6.26.5-test
$ ls -ld linux*
```

flcow demo

```
$ ls -ld linux*
drwxr-xr-x 21 4096 2009-01-17 15:37 linux-2.6.26.5
$ du -sk linux*
330592  linux-2.6.26.5
$ cp -al linux-2.6.26.5 linux-2.6.26.5-test
$ ls -ld linux*
drwxr-xr-x 21 4096 2009-01-17 15:37 linux-2.6.26.5
drwxr-xr-x 21 4096 2009-01-17 15:38 linux-2.6.26.5-test
$
```

flcow demo

```
$ ls -ld linux*
drwxr-xr-x 21 4096 2009-01-17 15:37 linux-2.6.26.5
$ du -sk linux*
330592  linux-2.6.26.5
$ cp -al linux-2.6.26.5 linux-2.6.26.5-test
$ ls -ld linux*
drwxr-xr-x 21 4096 2009-01-17 15:37 linux-2.6.26.5
drwxr-xr-x 21 4096 2009-01-17 15:38 linux-2.6.26.5-test
$ du -sk linux*
```

flcow demo

```
$ ls -ld linux*
drwxr-xr-x 21 4096 2009-01-17 15:37 linux-2.6.26.5
$ du -sk linux*
330592  linux-2.6.26.5
$ cp -al linux-2.6.26.5 linux-2.6.26.5-test
$ ls -ld linux*
drwxr-xr-x 21 4096 2009-01-17 15:37 linux-2.6.26.5
drwxr-xr-x 21 4096 2009-01-17 15:38 linux-2.6.26.5-test
$ du -sk linux*
330592  linux-2.6.26.5
5816    linux-2.6.26.5-test
$
```

flcow demo

$

flcow demo

```
$ grep LOCALVERSION= linux*/.config
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ vim linux-2.6.26.5-test/.config
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ vim linux-2.6.26.5-test/.config
$
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ vim linux-2.6.26.5-test/.config
$ grep LOCALVERSION= linux*/.config
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ vim linux-2.6.26.5-test/.config
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION="-test"
linux-2.6.26.5/.config:CONFIG_LOCALVERSION="-test"
$
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ export LD_PRELOAD="libflcow.so:$LD_PRELOAD"
$
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ export LD_PRELOAD="libflcow.so:$LD_PRELOAD"
$ export FLCOW_PATH="$PWD"
$
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ export LD_PRELOAD="libflcow.so:$LD_PRELOAD"
$ export FLCOW_PATH="$PWD"
$ vim linux-2.6.26.5-test/.config
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ export LD_PRELOAD="libflcow.so:$LD_PRELOAD"
$ export FLCOW_PATH="$PWD"
$ vim linux-2.6.26.5-test/.config
$
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ export LD_PRELOAD="libflcow.so:$LD_PRELOAD"
$ export FLCOW_PATH="$PWD"
$ vim linux-2.6.26.5-test/.config
$ grep LOCALVERSION= linux*/.config
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ export LD_PRELOAD="libflcow.so:$LD_PRELOAD"
$ export FLCOW_PATH="$PWD"
$ vim linux-2.6.26.5-test/.config
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION="-test"
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ export LD_PRELOAD="libflcow.so:$LD_PRELOAD"
$ export FLCOW_PATH="$PWD"
$ vim linux-2.6.26.5-test/.config
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION="-test"
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ du -sk linux*
```

flcow demo

```
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION=""
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ export LD_PRELOAD="libflcow.so:$LD_PRELOAD"
$ export FLCOW_PATH="$PWD"
$ vim linux-2.6.26.5-test/.config
$ grep LOCALVERSION= linux*/.config
linux-2.6.26.5-test/.config:CONFIG_LOCALVERSION="-test"
linux-2.6.26.5/.config:CONFIG_LOCALVERSION=""
$ du -sk linux*
330592  linux-2.6.26.5
5856    linux-2.6.26.5-test          (was 5816)
$
```

Other fs-related stuff

- **plasticfs**: http://plasticfs.sourceforge.net/
  Intercepts everything in glibc that accepts/returns filenames, provides several wide-ranging “filters” that can be combined.

Other fs-related stuff

- **plasticfs**: http://plasticfs.sourceforge.net/
  Intercepts everything in glibc that accepts/returns filenames, provides several wide-ranging “filters” that can be combined.
- **installwatch**: http://www.asic-linux.com.mx/~izto/checkinstall/installwatch.html
  Intercepts many functions to keep track of files created/modified by package installations

Network shenanigans

By intercepting libc functions that deal with the network, the preload library can modify how the network looks and behaves for programs.

- eg. socket(), connect(), bind(), listen(), gethostbyname(), etc

Network shenanigans

By intercepting libc functions that deal with the network, the preload library can modify how the network looks and behaves for programs.

- eg. socket(), connect(), bind(), listen(), gethostbyname(), etc
- **libshape**: http://freshmeat.net/projects/libshape/
  Limits the download rate of programs (ie. user-space network traffic rate-shaping).
- **netjail**: http://netjail.sourceforge.net/
  Controls where and how programs may connect to the network. Useful for dealing with spyware, etc.

Debugging and testing

By deliberately causing functions to fail, it's possible to test if programs are able to deal with such situations.

Debugging and testing

By deliberately causing functions to fail, it's possible to test if programs are able to deal with such situations.

- **failmalloc**: http://www.nongnu.org/failmalloc/
  Intercepts malloc() and friends, induces them to fail (return NULL) more often.
- **libeatmydata**:
  http://www.flamingspork.com/projects/libeatmydata/
  Intercepts fsync() and friends, disables them.

Debugging and testing

Critical functions can be intercepted to watch for potential bugs, security holes, etc.

- **electricfence**: http://perens.com/works/software/ElectricFence/
- **segv_handler**: http://junkcode.samba.org/#segv_handler

electricfence

Intercepts malloc() and friends, surrounds allocated memory with protected memory to catch buffer over/under-runs.

electricfence

Intercepts malloc() and friends, surrounds allocated memory with protected memory to catch buffer over/under-runs.

segv_handler

Signal handler for segfaults that produces a backtrace.

- Installs a signal handler for SIGSEGV and SIGBUS
- Signal handler basically just runs gdb on current process to dump backtrace into a logfile
- (program is still free to use its own handler)
  (lib could also intercept signal() to always force the backtrace)
- No overhead if system-wide in /etc/ld.so.preload
- Tiny and simple: 34 lines

Graphical augmentation

Since most graphical display is also done via libraries, it is also possible (though usually more complex) to intercept these graphical libraries to augment the display in various ways.

Graphical augmentation

Since most graphical display is also done via libraries, it is also possible (though usually more complex) to intercept these graphical libraries to augment the display in various ways.

- **libglfps**: http://users.dakotacom.net/~donut/programs/libglfps.html Adds a framerate display to OpenGL programs.

Part 3

1. Intro: Dynamic linking and LD_PRELOAD

2. Review: Existing LD_PRELOAD hacks

3. Code: Writing LD_PRELOAD hacks

4. Advanced: xlibtrace and xmultiwin

dlfcn

- `#include <dlfcn.h>` (**D**ynamic **L**inker **F**un**C**tio**N**s)
- Two main functions:

```
void* dlopen(const char* filename, int flag)
void* dlsym(void* handle, const char* symbol)
```

- Link against `libdl.so`, ie. `-ldl`

Motivating example

- Suppose we have some closed-source multi-threaded application that we are stuck with

Motivating example

- Suppose we have some closed-source multi-threaded application that we are stuck with
- This app is spawning too many threads (one for each CPU in the system), AND has no way to config it

Motivating example

- Suppose we have some closed-source multi-threaded application that we are stuck with
- This app is spawning too many threads (one for each CPU in the system), AND has no way to config it
- Digging around with strace, ltrace, gdb, /usr/include/bits, etc we find that it's calling `sysconf(_SC_NPROCESSORS_CONF)`

Motivating example

- Suppose we have some closed-source multi-threaded application that we are stuck with
- This app is spawning too many threads (one for each CPU in the system), AND has no way to config it
- Digging around with strace, ltrace, gdb, /usr/include/bits, etc we find that it's calling `sysconf(_SC_NPROCESSORS_CONF)`
- So we want a preload lib which overrides the return value of sysconf() for this case — "behaviour finessing"

libsysconfcpus

http://www.kev.pulo.com.au/libsysconfcpus/

We will construct this library from the ground up, as a simple example of how to create a preload library

libsysconfcpus

http://www.kev.pulo.com.au/libsysconfcpus/

We will construct this library from the ground up, as a simple example of how to create a preload library

- Version 0.1: Empty base project

libsysconfcpus

http://www.kev.pulo.com.au/libsysconfcpus/

We will construct this library from the ground up, as a simple example of how to create a preload library

- Version 0.1: Empty base project
- Version 0.2: Library debug and init

libsysconfcpus

http://www.kev.pulo.com.au/libsysconfcpus/

We will construct this library from the ground up, as a simple example of how to create a preload library

- Version 0.1: Empty base project
- Version 0.2: Library debug and init
- Version 0.3: Intercept sysconf()

libsysconfcpus

http://www.kev.pulo.com.au/libsysconfcpus/

We will construct this library from the ground up, as a simple example of how to create a preload library

- Version 0.1: Empty base project
- Version 0.2: Library debug and init
- Version 0.3: Intercept sysconf()
- Version 0.4: Adjust sysconf() behaviour

libsysconfcpus

http://www.kev.pulo.com.au/libsysconfcpus/

We will construct this library from the ground up, as a simple example of how to create a preload library

- Version 0.1: Empty base project
- Version 0.2: Library debug and init
- Version 0.3: Intercept sysconf()
- Version 0.4: Adjust sysconf() behaviour
- Version 0.5: Extra features

v0.1: configure.ac

$ cat configure.ac

```
AC_PREREQ(2.5)
AC_INIT(libsysconfcpus, [0.1], [kev@pulo.com.au])
AC_CONFIG_SRCDIR([src/libsysconfcpus.c])
AM_INIT_AUTOMAKE
AM_CONFIG_HEADER([config.h])

AC_PROG_CC
AC_PROG_INSTALL
AC_PROG_LN_S

...
```

v0.1: configure.ac

$ cat configure.ac

```
AC_PREREQ(2.5)
AC_INIT(libsysconfcpus, [0.1], [kev@pulo.com.au])
AC_CONFIG_SRCDIR([src/libsysconfcpus.c])
AM_INIT_AUTOMAKE
AM_CONFIG_HEADER([config.h])
```

```
AC_PROG_CC
AC_PROG_INSTALL
AC_PROG_LN_S

...
```

Generic AC/AM setup

v0.1: configure.ac

$ cat configure.ac

```
AC_PREREQ(2.5)
AC_INIT(libsysconfcpus, [0.1], [kev@pulo.com.au])
AC_CONFIG_SRCDIR([src/libsysconfcpus.c])
AM_INIT_AUTOMAKE
AM_CONFIG_HEADER([config.h])
```

```
AC_PROG_CC
AC_PROG_INSTALL
AC_PROG_LN_S
```

Check for programs

...

v0.1: configure.ac

```
...

AC_ENABLE_SHARED
AC_DISABLE_STATIC
AM_PROG_LIBTOOL

AC_HEADER_STDC
AC_C_CONST

AC_CONFIG_FILES([Makefile src/Makefile])
AC_OUTPUT
$
```

v0.1: configure.ac

```
...
AC_ENABLE_SHARED
AC_DISABLE_STATIC
AM_PROG_LIBTOOL
```

libtool shared libs

```
AC_HEADER_STDC
AC_C_CONST

AC_CONFIG_FILES([Makefile src/Makefile])
AC_OUTPUT
$
```

v0.1: configure.ac

```
...

AC_ENABLE_SHARED
AC_DISABLE_STATIC
AM_PROG_LIBTOOL

AC_HEADER_STDC
AC_C_CONST

AC_CONFIG_FILES([Makefile src/Makefile])
AC_OUTPUT

$
```

Check for standard stuff

v0.1: configure.ac

```
...

AC_ENABLE_SHARED
AC_DISABLE_STATIC
AM_PROG_LIBTOOL

AC_HEADER_STDC
AC_C_CONST

AC_CONFIG_FILES([Makefile src/Makefile])
AC_OUTPUT

$
```

Generic AC/AM conclusion

v0.1: Makefile.am

```
$ cat Makefile.am
SUBDIRS = src

$ cat src/Makefile.am
AUTOMAKE_OPTIONS = 1.4 foreign
CFLAGS += -Wall

lib_LTLIBRARIES = libsysconfcpus.la
libsysconfcpus_la_SOURCES = libsysconfcpus.c
libsysconfcpus_la_CFLAGS = -O1
libsysconfcpus_la_LIBADD = -ldl

#dist_man_MANS = libsysconfcpus.1

$ touch src/libsysconfcpus.c
```

v0.1: Makefile.am

```
$ cat Makefile.am
SUBDIRS = src
```

All source in `src/`

```
$ cat src/Makefile.am
AUTOMAKE_OPTIONS = 1.4 foreign
CFLAGS += -Wall

lib_LTLIBRARIES = libsysconfcpus.la
libsysconfcpus_la_SOURCES = libsysconfcpus.c
libsysconfcpus_la_CFLAGS = -O1
libsysconfcpus_la_LIBADD = -ldl

#dist_man_MANS = libsysconfcpus.1

$ touch src/libsysconfcpus.c
```

v0.1: Makefile.am

```
$ cat Makefile.am
SUBDIRS = src
```

```
$ cat src/Makefile.am
AUTOMAKE_OPTIONS = 1.4 foreign
CFLAGS += -Wall
```

Generic AM setup

```
lib_LTLIBRARIES = libsysconfcpus.la
libsysconfcpus_la_SOURCES = libsysconfcpus.c
libsysconfcpus_la_CFLAGS = -O1
libsysconfcpus_la_LIBADD = -ldl

#dist_man_MANS = libsysconfcpus.1
```

```
$ touch src/libsysconfcpus.c
```

v0.1: Makefile.am

```
$ cat Makefile.am
SUBDIRS = src

$ cat src/Makefile.am
AUTOMAKE_OPTIONS = 1.4 foreign
CFLAGS += -Wall

lib_LTLIBRARIES = libsysconfcpus.la
libsysconfcpus_la_SOURCES = libsysconfcpus.c
libsysconfcpus_la_CFLAGS = -O1
libsysconfcpus_la_LIBADD = -ldl

#dist_man_MANS = libsysconfcpus.1

$ touch src/libsysconfcpus.c
```

Setup for `libsysconf-cpus.so`

v0.1: Makefile.am

```
$ cat Makefile.am
SUBDIRS = src

$ cat src/Makefile.am
AUTOMAKE_OPTIONS = 1.4 foreign
CFLAGS += -Wall

lib_LTLIBRARIES = libsysconfcpus.la
libsysconfcpus_la_SOURCES = libsysconfcpus.c
libsysconfcpus_la_CFLAGS = -O1
libsysconfcpus_la_LIBADD = -ldl
```

```
#dist_man_MANS = libsysconfcpus.1
```

If you can be bothered

```
$ touch src/libsysconfcpus.c
```

v0.1: Makefile.am

```
$ cat Makefile.am
SUBDIRS = src

$ cat src/Makefile.am
AUTOMAKE_OPTIONS = 1.4 foreign
CFLAGS += -Wall

lib_LTLIBRARIES = libsysconfcpus.la
libsysconfcpus_la_SOURCES = libsysconfcpus.c
libsysconfcpus_la_CFLAGS = -O1
libsysconfcpus_la_LIBADD = -ldl

#dist_man_MANS = libsysconfcpus.1
```

```
$ touch src/libsysconfcpus.c
```

Empty source file for now

v0.1: Generate configure

```
$
```

v0.1: Generate configure

```
$ touch NEWS README AUTHORS ChangeLog
```

v0.1: Generate configure

```
$ touch NEWS README AUTHORS ChangeLog
$
```

v0.1: Generate configure

```
$ touch NEWS README AUTHORS ChangeLog
$ autoreconf -i
```

v0.1: Generate configure

```
$ touch NEWS README AUTHORS ChangeLog
$ autoreconf -i
configure.ac: installing `./install-sh'
configure.ac: installing `./missing'
src/Makefile.am: installing `./compile'
src/Makefile.am: installing `./depcomp'
Makefile.am: installing `./INSTALL'
Makefile.am: installing `./COPYING'
$
```

v0.1: configure

```
$ ./configure --prefix=/tmp/sysconf
checking for a BSD-compatible install... /usr/bin/ginstall -c
checking whether build environment is sane... yes
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking for gcc... gcc
...
checking for an ANSI C-conforming const... yes
configure: creating ./config.status
config.status: creating Makefile
config.status: creating src/Makefile
config.status: creating config.h
config.status: executing depfiles commands
$
```

v0.1: make

```
$ make
...
/bin/sh ../libtool --tag=CC --mode=link gcc  -g -O2 -→
    →Wall   -o libsysconfcpus.la -rpath /home/kev/→
    →libsysconfcpus/install/lib  libsysconfcpus_la-→
    →libsysconfcpus.lo -ldl
gcc -shared  .libs/libsysconfcpus_la-libsysconfcpus.o  →
    →-ldl  -Wl,-soname -Wl,libsysconfcpus.so.0 -o .→
    →libs/libsysconfcpus.so.0.0.0
...
$
```

v0.1: make install

```
$
```

v0.1: make install

```
$ make install
```

v0.1: make install

```
$ make install
...
$
```

v0.1: make install

```
$ make install
...
$ ls -l /tmp/sysconf/
```

v0.1: make install

```
$ make install
...
$ ls -l /tmp/sysconf/
drwxr-xr-x 2 4096 2008-12-29 08:39 lib
$
```

v0.1: make install

```
$ make install
...
$ ls -l /tmp/sysconf/
drwxr-xr-x 2 4096 2008-12-29 08:39 lib
$ ls -l /tmp/sysconf/lib/
```

v0.1: make install

```
$ make install
...
$ ls -l /tmp/sysconf/
drwxr-xr-x 2 4096 2008-12-29 08:39 lib
$ ls -l /tmp/sysconf/lib/
-rwxr-xr-x 1  852 2008-12-29 08:39 libsysconfcpus.la
lrwxrwxrwx 1   23 2008-12-29 08:39 libsysconfcpus.so ->
    -> libsysconfcpus.so.0.0.0
lrwxrwxrwx 1   23 2008-12-29 08:39 libsysconfcpus.so.0 ->
    ->-> libsysconfcpus.so.0.0.0
-rwxr-xr-x 1 6770 2008-12-29 08:39 libsysconfcpus.so->
    ->.0.0.0
$
```

v0.1: make dist

```
$ make dist
...
$ ls -la *.tar.gz
-rw-r--r-- 1 313430 2008-12-29 09:26 libsysconfcpus→
    →-0.1.tar.gz
$
```

v0.2: debug

src/libsysconfcpus.c

```
#include <stdio.h>
#include <stdarg.h>

static int do_debug() {
    return getenv("LIBSYSCONFCPUS_DEBUG");
}
static void dprintf(char *fmt, ...) {
    va_list ap;
    if (do_debug()) {
        va_start(ap, fmt);
        vfprintf(stderr, fmt, ap);
        va_end(ap);
        fflush(stderr);
    }
}
```

v0.2: init/fini

src/libsysconfcpus.c

```
static void _libsysconfcpus_init(void)
    __attribute__ ((constructor));
static void _libsysconfcpus_fini(void)
    __attribute__ ((destructor));

static void _libsysconfcpus_init(void) {
    dprintf("libsysconfcpus: starting up\n");
}

static void _libsysconfcpus_fini(void) {
    dprintf("libsysconfcpus: shutting down\n");
}
```

v0.2: support shell scripts

src/sysconfcpus.in

```
#!/bin/sh
LD_PRELOAD="@libdir@/libsysconfcpus.so:$LD_PRELOAD" \
exec "$@"
```

v0.2: support shell scripts

src/sysconfcpus.in

```
#!/bin/sh
LD_PRELOAD="@libdir@/libsysconfcpus.so:$LD_PRELOAD" \
exec "$@"
```

src/sysconfcpus-debug.in

```
#!/bin/sh
LD_PRELOAD="@libdir@/libsysconfcpus.so:$LD_PRELOAD" \
LIBSYSCONFCPUS_DEBUG=y \
exec "$@"
```

v0.2: support shell scripts

src/Makefile.am

```
edit = $(SED) \
    -e 's,@bindir\@,$(bindir),g' \
    -e 's,@mandir\@,$(mandir),g' \
    -e 's,@libdir\@,$(libdir),g' \
    -e 's,@datadir\@,$(datadir),g' \
    -e 's,@localstatedir\@,$(localstatedir),g'

sysconfcpus:: Makefile $(srcdir)/sysconfcpus.in
    rm -f sysconfcpus sysconfcpus.tmp && \
    $(edit) $(srcdir)/sysconfcpus.in >sysconfcpus.tmp \
    && mv sysconfcpus.tmp sysconfcpus
```

v0.2: support shell scripts

src/Makefile.am

```
nodist_bin_SCRIPTS  = sysconfcpus
nodist_bin_SCRIPTS += sysconfcpus-debug

dist_noinst_DATA  = sysconfcpus.in
dist_noinst_DATA += sysconfcpus-debug.in

DISTCLEANFILES  = sysconfcpus
DISTCLEANFILES += sysconfcpus-debug
```

v0.2: rebuild

```
$
```

v0.2: rebuild

```
$ make clean
```

v0.2: rebuild

```
$ make clean
...
$
```

v0.2: rebuild

```
$ make clean
...
$ autoreconf -m
```

v0.2: rebuild

```
$ make clean
...
$ autoreconf -m
...
$
```

v0.2: rebuild

```
$ make clean
...
$ autoreconf -m
...
$ make install
```

v0.2: rebuild

```
$ make clean
...
$ autoreconf -m
...
$ make install
...
$
```

v0.2: rebuild

```
$ make clean
...
$ autoreconf -m
...
$ make install
...
$ ls -l /tmp/sysconf/
```

v0.2: rebuild

```
$ make clean
...
$ autoreconf -m
...
$ make install
...
$ ls -l /tmp/sysconf/
drwxr-xr-x 2 4096 2008-12-30 23:07 bin
drwxr-xr-x 2 4096 2008-12-30 23:07 lib
$
```

v0.2: rebuild

```
$ make clean
...
$ autoreconf -m
...
$ make install
...
$ ls -l /tmp/sysconf/
drwxr-xr-x 2 4096 2008-12-30 23:07 bin
drwxr-xr-x 2 4096 2008-12-30 23:07 lib
$ ls -l /tmp/sysconf/bin/
```

v0.2: rebuild

```
$ make clean
...
$ autoreconf -m
...
$ make install
...
$ ls -l /tmp/sysconf/
drwxr-xr-x 2 4096 2008-12-30 23:07 bin
drwxr-xr-x 2 4096 2008-12-30 23:07 lib
$ ls -l /tmp/sysconf/bin/
-rwxr-xr-x 1 102 2008-12-30 23:07 sysconfcpus
-rwxr-xr-x 1 125 2008-12-30 23:07 sysconfcpus-debug
$
```

v0.3: no error checking or debug

```
#include <dlfcn.h>

long sysconf(int name) {
    static void *libc_handle = NULL;
    static long (*underlying)(int name);
    long retval;

    if (!libc_handle) {
        libc_handle = dlopen("libc.so.6", RTLD_LAZY);
        underlying = dlsym(libc_handle, "sysconf");
    }
    retval = (*underlying)(name);
    return retval;
}
```

v0.3: full code - part 1

```
long sysconf(int name) {
   static void *libc_handle = NULL;
   static long (*underlying)(int name);
   long retval;
   const char *err;

   dprintf("libsysconfcpus: sysconf(%d): entered\n", name);

   if (!libc_handle) {
      libc_handle = dlopen("libc.so.6", RTLD_LAZY);
      dprintf("libsysconfcpus: sysconf: libc_handle = 0x%x\n",
              libc_handle);
      if (!libc_handle) {
         fprintf(stderr, "libsysconfcpus: Error: Unable to find "
                         "libc.so: %s\n", dlerror());
         exit(1);
      }
      ...
```

v0.3: full code - part 2

```
    dlerror();
    underlying = dlsym(libc_handle, "sysconf");
    dprintf("libsysconfcpus: sysconf: underlying = 0x%x\n",
            underlying);
    err = dlerror();
    if (err) {
        dprintf("libsysconfcpus: sysconf: err = \"%s\"\n", err);
    }
    if (!underlying || err) {
        fprintf(stderr, "libsysconfcpus: Error: Unable to find the "
                        "underlying sysconf(): %s\n", dlerror());
        exit(1);
    }
}
dprintf("libsysconfcpus: about to call underlying sysconf()\n");
retval = (*underlying)(name);
dprintf("libsysconfcpus: sysconf(%d) = %ld\n", name, retval);
return retval;
}
```

v0.3: test

```
$
```

v0.3: test

```
$ rudeapp
```

v0.3: test

```
$ rudeapp
Rude application v1.0
Spawning 8 threads
^C
$
```

v0.3: test

```
$ rudeapp
Rude application v1.0
Spawning 8 threads
^C
$ /tmp/sysconf/bin/sysconfcpus-debug rudeapp
```

v0.3: test

```
$ rudeapp
Rude application v1.0
Spawning 8 threads
^C
$ /tmp/sysconf/bin/sysconfcpus-debug rudeapp
libsysconfcpus: starting up
Rude application v1.0
libsysconfcpus: sysconf(83): entered
libsysconfcpus: sysconf: libc_handle = 0x1738eb78
libsysconfcpus: sysconf: underlying = 0x16cbc260
libsysconfcpus: about to call underlying sysconf()
libsysconfcpus: sysconf(83) = 8
Spawning 8 threads
^C
$
```

Version 0.4: Adjust sysconf()

```
long sysconf(int name) {
    static void *libc_handle = NULL;
    static long (*underlying)(int name);
    long retval;

    ...

    /* Opportunity to modify parameters */

    retval = (*underlying)(name);

    /* Opportunity to modify return value */

    return retval;
}
```

v0.4: override retval

```
long v;    char *env, *endptr;
...
if (name == _SC_NPROCESSORS_CONF ||
    name == _SC_NPROCESSORS_ONLN) {
   env = getenv("LIBSYSCONFCPUS");
   if (env) {
      v = strtol(env, &endptr, 10);
      if (endptr == env)
         dprintf("libsysconfcpus: Warning: $%s does not "
             "contain a number\n", libsysconfcpus_envvar);
      else
         retval = v;
   }
}
dprintf("libsysconfcpus: sysconf(%d)=%ld\n",name,retval);
```

v0.4: test

```
$
```

v0.4: test

```
$ LIBSYSCONFCPUS=2 /tmp/sysconf/bin/sysconfcpus-debug →
    →rudeapp
```

v0.4: test

```
$ LIBSYSCONFCPUS=2 /tmp/sysconf/bin/sysconfcpus-debug →
    →rudeapp
libsysconfcpus: starting up
Rude application v1.0
libsysconfcpus: sysconf(83): entered
libsysconfcpus: sysconf: libc_handle = 0x1738eb78
libsysconfcpus: sysconf: underlying = 0x16cbc260
libsysconfcpus: about to call underlying sysconf()
libsysconfcpus: underlying sysconf(83) = 8
libsysconfcpus: sysconf(83) = 2
Spawning 2 threads
^C
$
```

Version 0.5: Extra features

- Use `$LIBSYSCONFCPUS_CONF` and `$LIBSYSCONFCPUS_ONLN` with `$LIBSYSCONFCPUS` as a fallback
- Make the shell scripts take command line arguments

```
$ ../install/bin/sysconfcpus
sysconfcpus version 0.5
Usage:
    sysconfcpus [options] <command> <args> ...
Valid options are:
    -v, --version   Display the sysconfcpus version
    -h, --help      Display this help
    -d, --debug     Display debug info
    -n, --num <n>   Number of processors (both conf and onln)
    -c, --conf <n>  Number of processors (conf(igured))
    -o, --onln <n>  Number of processors (online)
$
```

RTLD_NEXT

- GNU extension: special library pseudo-handle
- Find symbol in ‘next’ library after current

```
#define __USE_GNU     // or #define _GNU_SOURCE
#include <dlfcn.h>
...

#if defined(RTLD_NEXT)
      libc_handle = RTLD_NEXT;
#else
      libc_handle = dlopen("libc.so.6", RTLD_LAZY);
#endif
      underlying = dlsym(libc_handle, "sysconf");
```

Part 4

1. Intro: Dynamic linking and LD_PRELOAD

2. Review: Existing LD_PRELOAD hacks

3. Code: Writing LD_PRELOAD hacks

4. Advanced: xlibtrace and xmultiwin

xlibtrace

http://www.kev.pulo.com.au/xlibtrace/

The idea:

- Intercept all functions in xlib
- Make each output tracing info like strace/ltrace

xlibtrace

http://www.kev.pulo.com.au/xlibtrace/

The idea:

- Intercept all functions in xlib
- Make each output tracing info like strace/ltrace

Why not just use ltrace?

- Can verbosely interpret and output library-specific data structures
- Basis for wide-intercepting preload libs
- Portable - doesn't require per-architecture changes

xlibtrace output

```
[32095] (0): int XNextEvent(Display * display = (0x60a3b0) (⟶
    ⟶opaque), XEvent * event_return = (0x7fffbd645870) { int⟶
    ⟶ type = ReparentNotify, unsigned long serial = 35, Bool⟶
    ⟶ send_event = False, Display * display = (0x60a3b0) (⟶
    ⟶opaque), Window event = 0x8400003, Window window = 0⟶
    ⟶x8400003, Window parent = 0x3203016, int x = 0, int y =⟶
    ⟶ 0, Bool override_redirect = False }) = <unfinished...>
[32095] (0): int XNextEvent(Display * display = (0x60a3b0) (⟶
    ⟶opaque), XEvent * event_return = (0x7fffbd645870) { int⟶
    ⟶ type = ConfigureNotify, unsigned long serial = 35, ⟶
    ⟶Bool send_event = True, Display * display = (0x60a3b0) ⟶
    ⟶(opaque), Window event = 0x8400003, Window window = 0⟶
    ⟶x8400003, int x = 2853, int y = 24, int width = 100, ⟶
    ⟶int height = 100, int border_width = 0, Window above = ⟶
    ⟶0x3203015, Bool override_redirect = False }) = 0
```

xlibtrace skeleton

```
int XClearWindow(Display* display, Window w) {
    static int (*underlying)(Display* display, Window w);
    int retval;

    if (!underlying) {
        underlying = dlsym(RLTD_NEXT, "XClearWindow");
    }
    /* Print function name and parameters */
    retval = (*underlying)(display, w);
    /* Print return value */
    return retval;
}
```

xlibtrace intercept function

Heavy use of C preprocessor and awk to generate code

```
/* int XClearWindow(Display * display, Window w) */
#define __TRACE_SAFERETTYPE_XClearWindow__ int
#define __TRACE_PROTOARGLIST_XClearWindow__ (Display* display,Window w)
#define __TRACE_ARGLIST_XClearWindow__ (display, w)
#define __TRACE_ADDITIONAL_LOCAL_VARS_XClearWindow__
#define __TRACE_ADDITIONAL_PRE_RUN_UNDERLYING_XClearWindow__
#define __TRACE_ADDITIONAL_POST_RUN_UNDERLYING_XClearWindow__
#define __TRACE_PRINTF_BODY_XClearWindow__ \
    always_reprint = 1; \
    __TRACE_PRINTF_ARG__(XClearWindow, Display_p, display) \
    __PRINT_COMMA__(out) \
    __TRACE_PRINTF_ARG__(XClearWindow, Window, w)
#define __TRACE_RUN_UNDERLYING_EPILOGUE_XClearWindow__ \
    __TRACE_RUN_XFLUSH_XSYNC__(XClearWindow, Display_p, display)
__TRACE__(TYPED, FIXED, XClearWindow)
```

xlibtrace print function

```
#define __REALTYPE__(t) __REALTYPE_##t##__

#define __REALTYPE_XPoint__    XPoint
#define __REALTYPE_XPoint_p__  XPoint *
#define __REALTYPE_XPoint_pp__ XPoint **

#define __TRACE_PRINT_TYPE_STRUCT_BODY_XPoint__(safetype) \
  __TRACE_PRINT_STRUCT_MEMBER__(f,safetype, *value, short, x)\
  __PRINT_COMMA__(f) \
  __TRACE_PRINT_STRUCT_MEMBER__(f,safetype, *value, short, y)

__TRACE_PRINT_TYPE_STRUCT__(XPoint)
```

xlibtrace print function

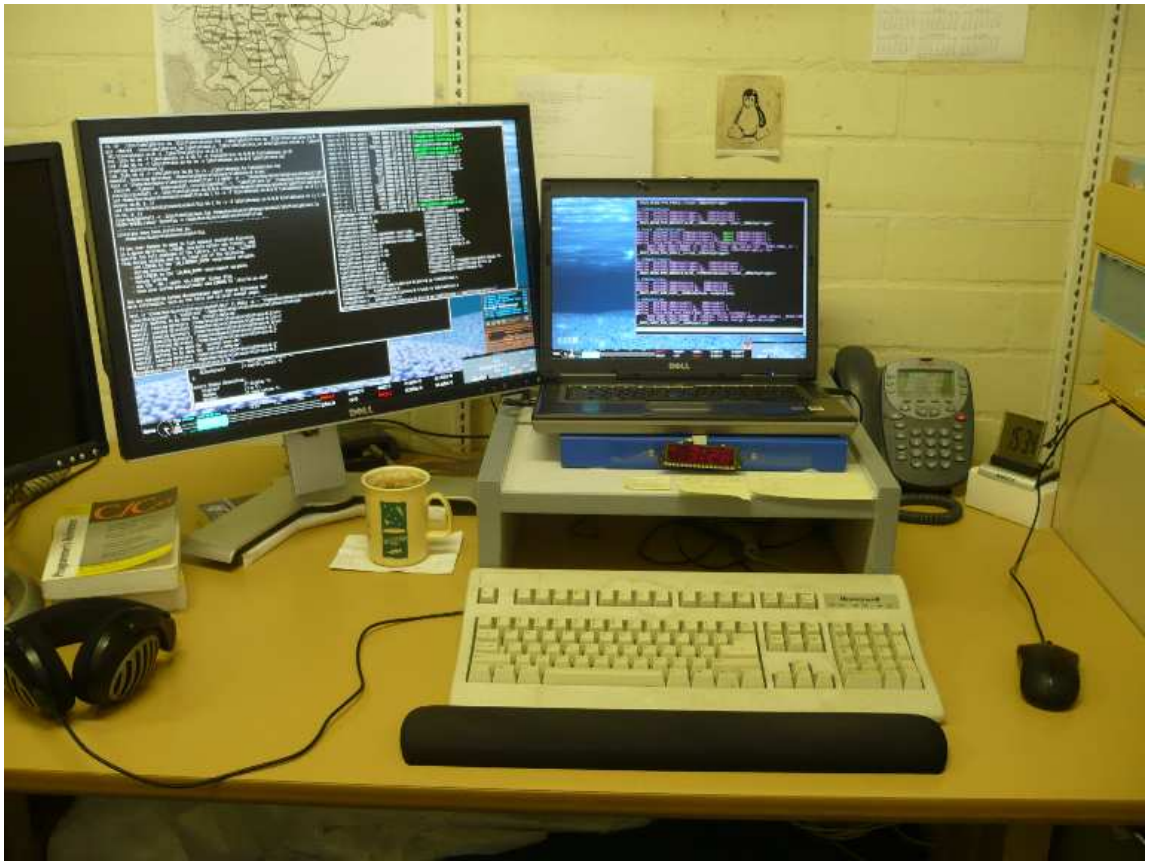
```
void print_type_XPoint_(FILE *f, const char *funcname, const
    →char *argname, const char *type, XPoint const * value)
{
    fprintf(f, "{ ");
    print_arg_name(f, "XPoint", "x", "short");
    print_type_short_(f, "XPoint", "x", "short", (*value).x);
    fprintf(f, ", ");
    print_arg_name(f, "XPoint", "y", "short");
    print_type_short_(f, "XPoint", "y", "short", (*value).y);
    fprintf(f, " }");
}
```

xlibtrace print function

```
void print_type_XPoint_p_(FILE *f, const char *funcname, const
    → char *argname, const char *type, XPoint const * value)
{
    if (value) {
        fprintf(f, "(%p) ", value);
        print_type_XPoint_(f, funcname, argname, type, value);
    } else {
        fprintf(f, "%p", value);
    }
}
```

Same for `XPoint**` and `XPoint***`

xmultiwin: multi-head display



xmultiwin: multi-head display

The problem

I want some windows shown on both monitors at the same time.

The problem

I want some windows shown on both monitors at the same time.

Some simple ones I just run 2 instances of:

- xload, pload, xapmload, xcal, xclock, xbiff, etc

The problem

I want some windows shown on both monitors at the same time.

Some simple ones I just run 2 instances of:

- xload, pload, xapmload, xcal, xclock, xbiff, etc

But some I can't:

- x11-ssh-askpass
- xmms
- xscreensaver dialogs
- wmwork

xmultiwin

http://www.kev.pulo.com.au/xmultiwin/

The idea:

- Intercept all window-related functions in xlib
- In XCreateWindow() automatically “clone” the window
- Then duplicate all xlib calls to the cloned windows

xmultiwin — “Fun”

- Parallel window trees
- Event translation from cloned windows
- Excessive redraw heuristics
- X extensions
- Support for higher level libs, eg OpenGL, cairo, etc
- Many monitors (> 1 clone)
- Clone on different $DISPLAY
- Clone to Xvfb: screen(1) for X?
- varargs with LD_PRELOAD?
  (current hack: count args and have generated cases)
- Silent 32/64 bit LD_PRELOAD?
  (without hacking glibc directly?)

Thank you

Questions?

+ LaTeX
+ powerdot: http://www.ctan.org/tex-archive/help/Catalogue/entries/powerdot.html
+ impressive: http://impressive.sourceforge.net/

THE AUSTRALIAN NATIONAL UNIVERSITY