Introduction to Linux Kernel Development

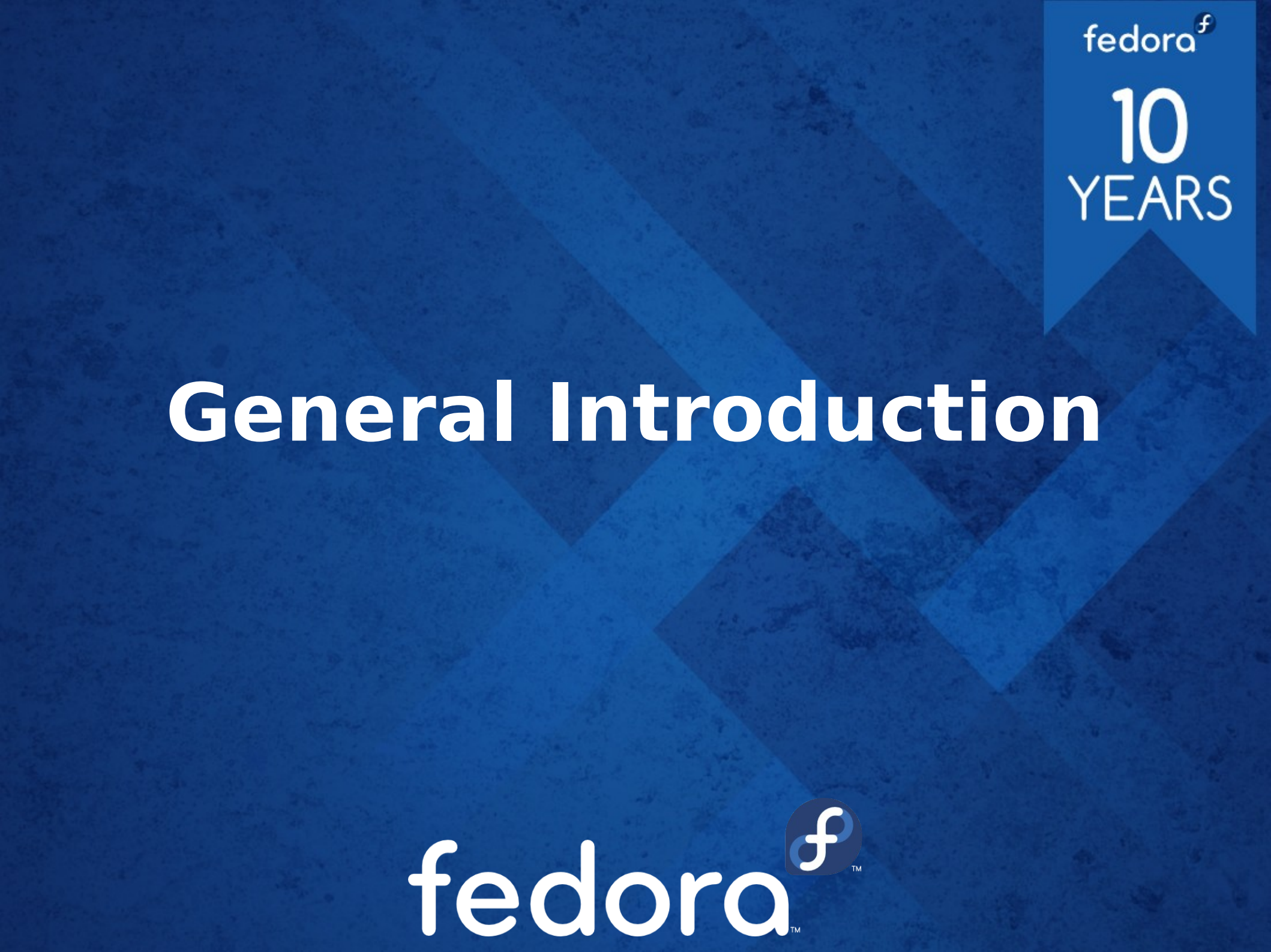
Your first patch

Levente Kurusa
Linux kernel hacker & enthusiast
Fedora Project Hungary
<levex@linux.com>

Developer Conference 2014, Brno, CZ.

Topics

1 General introduction
2 Building
3 Kernel API
4 Patching
5 Sending it off


fedora
10
YEARS
General Introduction
fedora

What is a kernel?

The operating system's core

Responsibilities:

1 Initialise the system for the user

2 Protect the data

3 Introduce abstractions to help the development of new software

Kernel types

1 Microkernel
2 Monolithic kernel
3 Hybrid kernel

Some exotics:

1 Exokernel
2 Cache kernel

Linux specifics

1 Monolithic kernel

2 Modular

3 Error-happens-so-panic type

The source of panic()

" I remarked to Dennis that easily half the code I was writing in Multics was error recovery code. He said, "We left all that stuff out. If there's an error, we have this routine called panic, and when it is called, the machine crashes, and you holler down the hall, 'Hey, reboot it.'" "

-- Tom van Vleck to Dennis Ritchie

fedora 10 YEARS

Building

fedora

Kconfig

Allows for specific code to be excluded via optional preprocessor macros.

Why do we need this?

1 To allow different configurations!

2 Headers are lengthy!

(and incomprehensible anyways)

Kconfig symbols

Specific symbols that will be converted to CONFIG_* preprocessor defines.

Types:
1 bool (true or false) [y, n]
2 tristate (built-in, module, false) [y, m ,n]

They have default values

Kconfig example


```
config ATA_ACPI
    bool "ATA ACPI Support"
    depends on ACPI && PCI
    default y
    help
      This option adds support for ATA-related ACPI objects.
      These ACPI objects add the ability to retrieve taskfiles
      from the ACPI BIOS and write them to the disk
    controller.
      These objects may be related to performance, security,
      power management, or other areas.
```

Makefiles are still in use!

We still use makefiles to define the relations between the source files.

GNU Make gives us a pretty cool interface!

We don't reinvent the wheel!

How to configure?

Configuration is easy thanks to GNU make!

1 'make defconfig' -> Create a default configuration for $ARCH

2 'make menuconfig' -> Shows a cool ncurses menu, which allows pretty configuration

3 'make randconfig' -> Randomly configure the kernel

4 'make xconfig' -> For all the GUI fans out there!

Building

Configuration completed, let's build.

'make -j`nproc`'

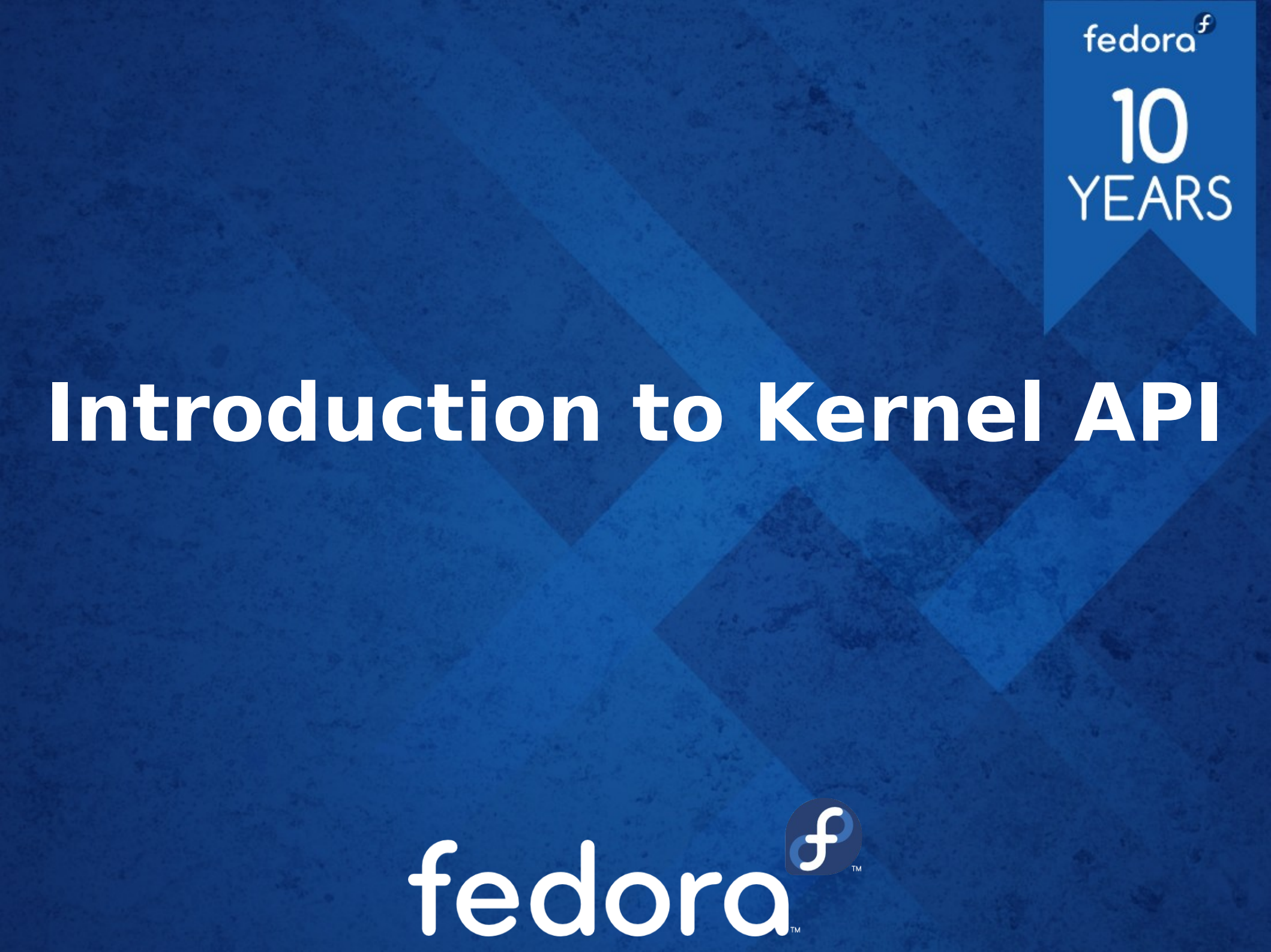
(Looks way cooler than 'make -j4')

Post-build

To actually boot, you will need an initrd or initramfs.

The kernel file that needs to be booted will be at:
./$ARCH/boot/bzImage

Or if you need the vmlinux file it will be in the root.


fedora
10
YEARS
Introduction to Kernel API
fedora

No C library, no developer

Most developers flee when they see they can't use printf()

But wait, there is a replacement for that! printk()

Actually, the relevant parts from the C library are there!

This means: sscanf(), [vscn]printf(), kstrtoX() etc...

Thread-safety concerns

Since the kernel supports Symmetric MultiProcessing, we need ways to achieve mutual exclusion.

This is what we have in store:

1 mutex (with global/local IRQ disable, full flags save)
2 semaphores!
3 spinlock_t

A cool module system

Your device will most likely not used by all users, so modules are a great way.

Modules are built-up like this:

1 module_init(init function ptr);
    Shows which function will be called when the module is passed to insmod

2 module_exit(exit function ptr);
    This function will be called when your module is rmmod'd.

PCI drivers

PCI drivers are built up in the exact same way, but they also have a table consisting of pci_device_id's.

To register your PCI driver, you would call:
pci_register_driver(ptr-to-pci_operations-struct);

This together with a probe() function will suffice.

kobjects

Kobjects are primitives that are parts of a garbage collection system.

Kobjects are usually 'inlined' in a container struct which handles them.

They have a few helper functions i.e.
kobj_{put,get}(struct kobject *kobj);

They have types! (struct kobj_type)


fedora
10
YEARS
Patching
fedora

Contibuting to the Kernel

1 Reviewing patches for common errors
2 Building randconfigs
3 Adding support for new device
4 Fixing a bug
5 Doing staging work

This is what we will do!

checkpatch.pl

Excellent utility to find code-style errors!

Sometimes it also finds bugs!

All patches have to be checkpatch-safe before being applied!

Code style?

1 Code style contributes to the consistency of the kernel

2 Helps to spot bugs

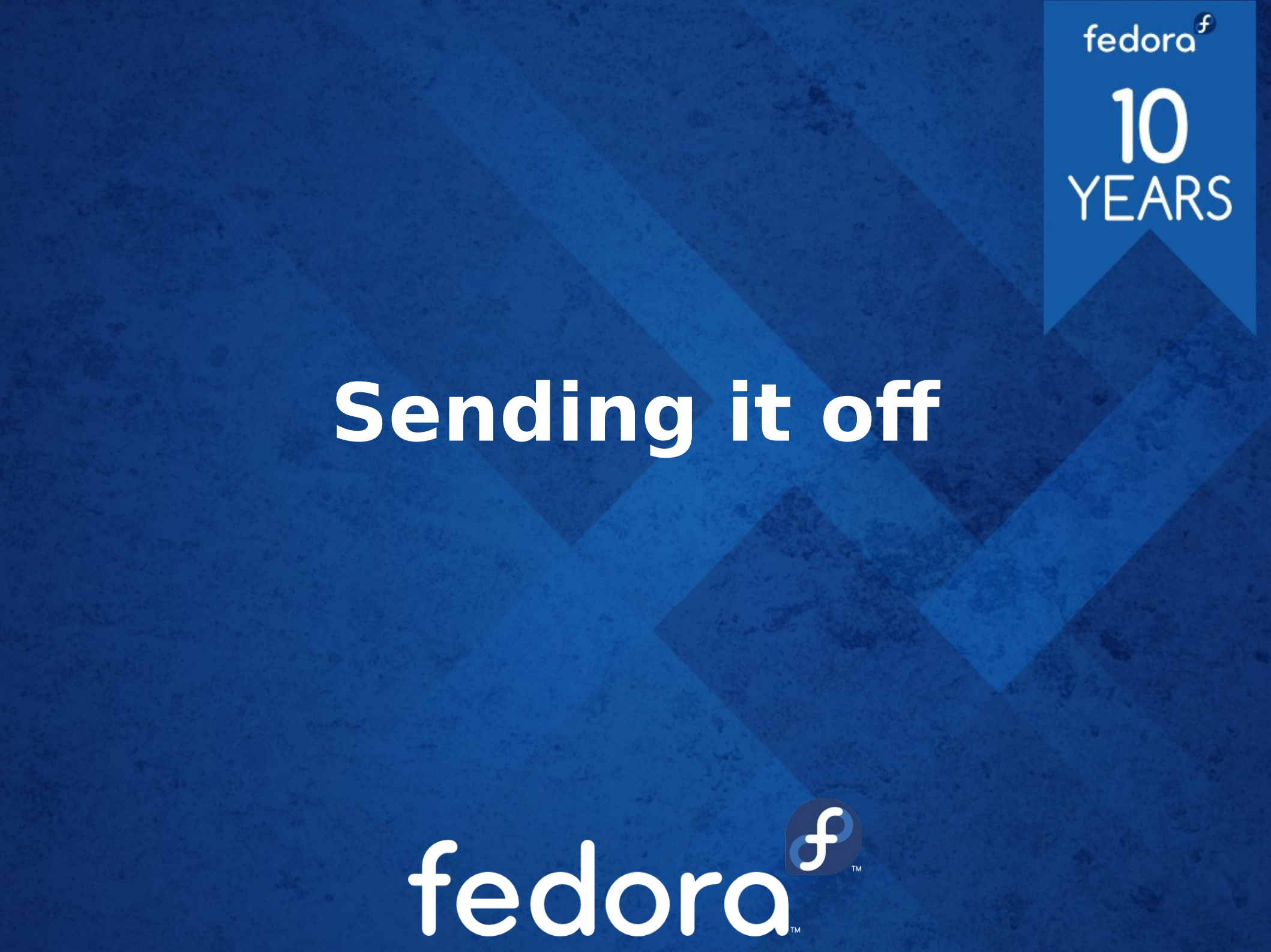
3 Don't be afraid, this isn't that terribly bad code style!

Developer Certificate of Origin

1 I created this change and I have the right to post it; or

2 The change is based on a previous change under the same license.

3 The change was sent to me by someone who has certified any of these options.


fedora
10
YEARS
Sending it off
fedora

Thanks for your attention!

Levente Kurusa
<levex@linux.com>

Developer Conference 2014, Brno, CZ.

http://devconf.cz/f/105