# Speaking UNIX: Get to know Ksplice

## Give reboots the boot

Skill Level: Intermediate

Martin Streicher
Software Developer
Pixel, Byte, and Comma

27 Jul 2010

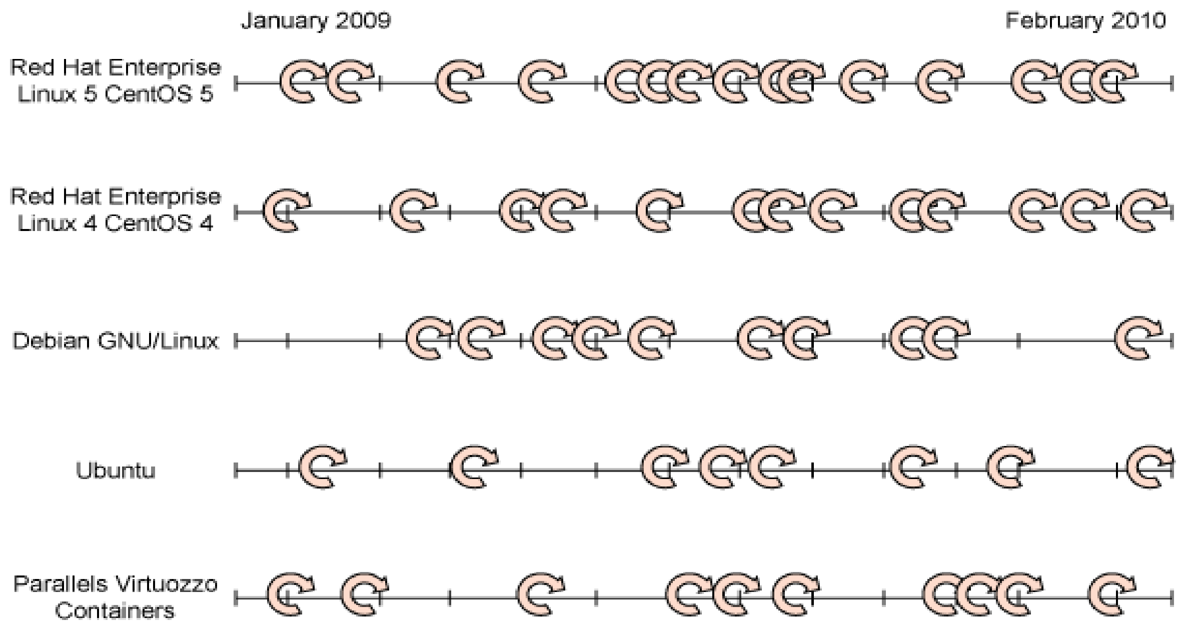
Ksplice applies kernel patches on-the-fly—no reboot required—in a fraction of a second. Here's a hands-on guide to performing painless system updates.

UNIX® machines run and run (and run). A desktop or portable system can be left on for weeks, even months, and server uptime can stretch to a year or more. Indeed, if you could preclude hardware failures and Mother Nature, a UNIX system might run perennially. Alas, hardware is imperfect, Mother Nature has a mind of her own, and software suffers from bugs. The gear requires replacement. Thunderstorms can and do happen. And systems require patches and restarts. Much like death and taxes, downtime is a certainty.

However, unlike death and taxes, you can *minimize* downtime. Proactive and regular system maintenance and hot spares boost availability, as does a robust data center replete with backup power and redundant connections to the Internet. Thankfully, too, most applications and libraries can be updated seamlessly and on demand using tools such as rpm, Aptitude, and yum.

Unfortunately, kernel updates—modifications to address vulnerabilities and flaws in the core system software—aren't so painless. A kernel update is very disruptive, requiring scheduled downtime to temporarily halt all services on each and every machine affected. Although such upkeep is necessary and vital, keeping pace with kernel updates can nonetheless make operations something akin to a yo-yo. To wit, the time lines in Figure 1 show the frequency of critical kernel updates for a number of popular operating systems between January 2009 and February 2010. Each cycle icon represents a mandatory restart (image courtesy of Ksplice, Inc.).

**Figure 1. Frequency of kernel updates by operating system**



But now, you can patch your kernel as it runs, eliminating the otherwise wholesale interruption caused by a restart. In other words, scheduled upgrades no longer require a schedule.

Ksplice is a set of tools to patch the kernel *in situ,* as it's running—no reboot required. Given an existing kernel, its source code, and one or more unified diff files (a unified diff is the canonical form for kernel patches), Ksplice replaces existing, errant object code in the resident kernel with new object code. Ksplice can replace both program code and data structures. Better yet, a kernel *splice* interrupts normal system operation for a mere fraction of a millisecond, leaving daemons, processes, and connections intact.

Let's look briefly at how Ksplice works and learn how to use its tools to keep a kernel up to date. There are three ways to use Ksplice:

1. Graphical user interface (GUI)
2. Equivalent, high-level command-line utilities
3. Raw Ksplice tools (if you have the source to your kernel)

An Ubuntu version 9.04 or version 9.10 user, for example, can download and install a point-and-click application to choose and apply kernel modifications. All three variants of Ksplice are introduced here.

## Patching a live kernel

Putting it succinctly, Ksplice patches a running kernel by replacing one or more vulnerable or faulty functions with newer, correct implementations. To effect the substitution, the tool amends a running kernel image with the new functions' object code and injects a jump at the head of each existing function to call its new counterpart. Virtually the entire kernel remains unchanged, save for the new object code and a few instructions to redirect the function calls.

Additionally, Ksplice can affect data structures, albeit with a little extra programming. Ksplice can run code during an update to facilitate such a change. Ksplice also provides a pair of hooks to run setup and tear-down code immediately before and immediately after the kernel is updated. You can also add new functions to augment the kernel; new functions simply have no counterparts in the original kernel.

You can apply Ksplice to virtually any kernel, even those that shipped before Ksplice was developed. To splice a kernel, you must have its source code, the set of patches you want to apply, and a compiler capable of isolating each function and data structure in the kernel in its own section in the object code. For example, the GNU Compiler Collection (GCC) provides the flags `-ffunction-sections` and `-fdata-sections`, respectively, for those exact purposes (Other compilers, such as the Intel® `C` Compiler, have similar capabilities). Such "modular" object code is more easily manipulated by Ksplice. The running kernel to be patched need not have been compiled with the same special options.

Ksplice refers to the kernel compiled from the original source as the *pre* kernel. The new kernel built from the patched source is the *post* kernel. And the running kernel is nicknamed the *run* kernel. Ksplice fails if *pre* is not the same as run—a clear indication that the source code does not match the running kernel. Apart from that special condition, Ksplice compares *pre* to *post* and each difference becomes a splice. All splices are bundled into a single object file ready to be injected into the kernel.

Ksplice uses the `stop_machine` feature (or its equivalent, depending on your flavor of UNIX) to prepare to splice new code. Think of `stop_machine` as a traffic signal; it allows a sole CPU to proceed while idling all others. After executing `stop_machine`, Ksplice analyzes each function marked for replacement to determine whether it is active. If even a single thread's instruction pointer refers to a function's code in memory or a single thread's kernel stack refers to a return address within the function, the function is considered active and cannot be replaced. Otherwise, the function is deemed inactive and is replaced. Ksplice tries repeatedly to make a replacement but may ultimately abandon the effort. If so, it reports an error and stops.

## Kernel patching was never this easy

If you run one of the more popular UNIX distributions, Ksplice couldn't be easier to use. The authors of Ksplice provide a client application customized for your distro. You can keep your kernel up to date for a nominal fee of less than USD$5 per system per month. You can find a list of supported operating systems on the Ksplice website (see Resources). The Ksplice software is available free of charge for Ubuntu 9.04 (Jaunty Jackalope) and Ubuntu 9.10.

For demonstration purposes, this article uses Ubuntu 9.10 and splices its kernel. To use Ksplice on Ubuntu, you must download and install the Ksplice Uptrack client (see Resources for a link). Uptrack manages the rebootless kernel updates available for your system. You can view and install Ksplice updates, and you can review alerts when new kernel updates are available. The software is provided as a Debian package. You can also download the software from the command line using a tool such as `wget` or cURL.

Next, use `gdebi` to install the package. If you do not yet have `gdebi`, you can install it with the command:

```
sudo apt-get install gdebi
```

If you prefer a graphical client, you can install Ksplice with the command:

```
sudo gdebi-gtk ksplice-uptrack.deb
```

`gdebi` installs Uptrack and its dependencies, which include `kerneloops`, the YAML library, Python, and the cURL library (see Listing 1).

### Listing 1. Installing Uptrack and its dependencies

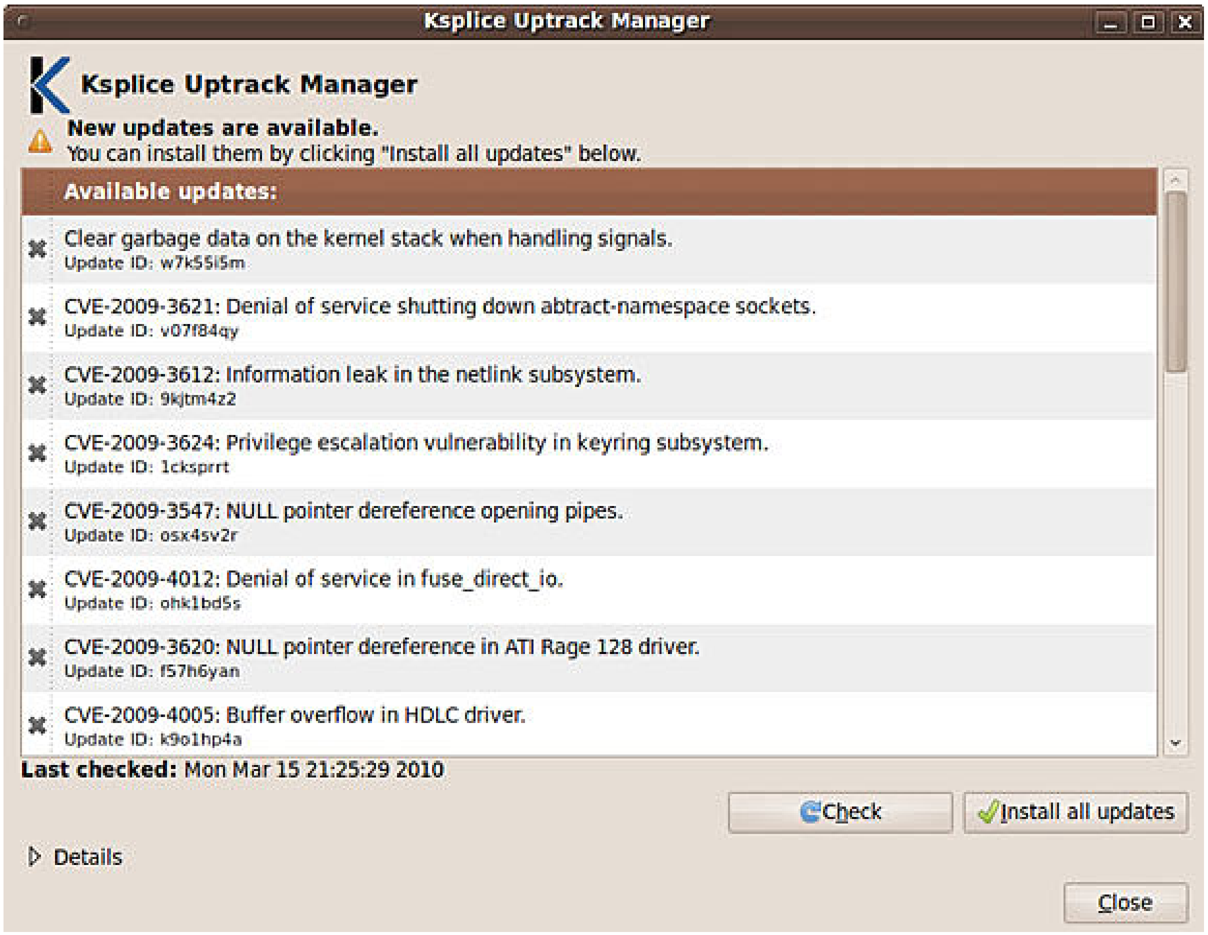
```
$ wget
http://www.ksplice.com/uptrack/dist/karmic/ksplice-uptrack.deb
$ sudo gdebi ksplice-uptrack.deb
Reading package lists: Done
Reading state information: Done
Reading state information: Done
Reading state information: Done
Reading state information: Done

Requires the installation of the following packages:
kerneloops  curl  libcurl3  python-yaml  libyaml-0-1
Client for the Ksplice Uptrack service
The Ksplice Uptrack service enables you to keep your
system
up to date and secure without rebooting it.
This package contains the command-line and graphical
Uptrack clients.
```

```
Do you want to install the software package? [y/N]:y
...
```

After you accept the terms of service and the installation finishes, Uptrack launches automatically. The installer also places a Ksplice icon—a large letter K in the task bar—for easy access. Figure 2 shows the Uptrack Manager launching for the first time on a clean installation of Ubuntu 9.10. To patch the kernel, click **Install all updates**.

**Figure 2. Launching the Uptrack Manager**



Ksplice Uptrack Manager

**New updates are available.**
You can install them by clicking "Install all updates" below.

**Available updates:**

Clear garbage data on the kernel stack when handling signals.
Update ID: w7k55i5m

CVE-2009-3621: Denial of service shutting down abtract-namespace sockets.
Update ID: v07f84qy

CVE-2009-3612: Information leak in the netlink subsystem.
Update ID: 9kjtm4z2

CVE-2009-3624: Privilege escalation vulnerability in keyring subsystem.
Update ID: 1cksprrt

CVE-2009-3547: NULL pointer dereference opening pipes.
Update ID: osx4sv2r

CVE-2009-4012: Denial of service in fuse_direct_io.
Update ID: ohk1bd5s

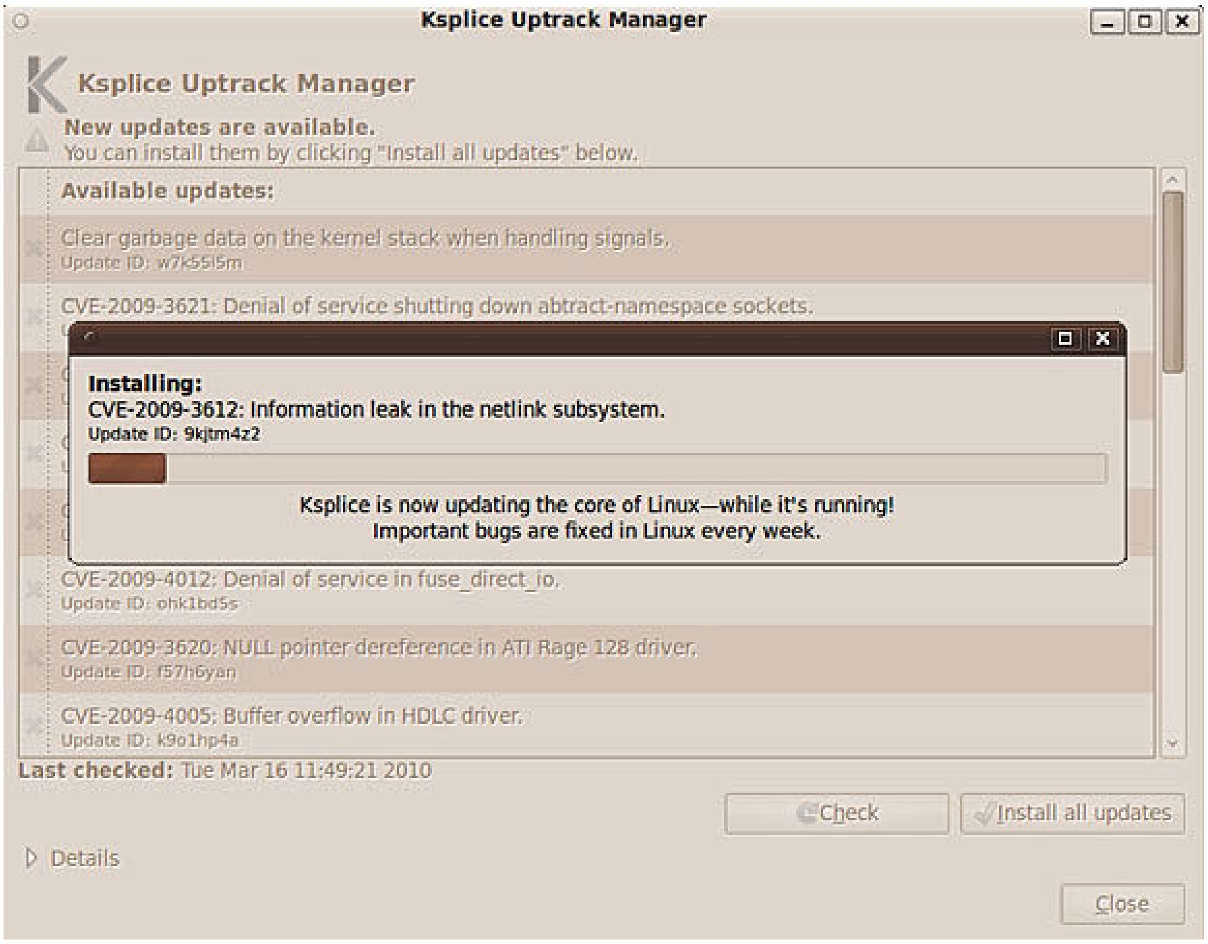
CVE-2009-3620: NULL pointer dereference in ATI Rage 128 driver.
Update ID: f57h6yan

CVE-2009-4005: Buffer overflow in HDLC driver.
Update ID: k9o1hp4a

**Last checked:** Mon Mar 15 21:25:29 2010

A progress bar, shown in Figure 3, echoes each patch as it is installed. Patching the kernel takes just a few moments, even when a good number of patches are outstanding. And, of course, no reboot is required.

**Figure 3. The Ksplice progress bar**


Ksplice Uptrack Manager

Ksplice Uptrack Manager

New updates are available.
You can install them by clicking "Install all updates" below.

Available updates:

Clear garbage data on the kernel stack when handling signals.
Update ID: w7k55i5m

CVE-2009-3621: Denial of service shutting down abtract-namespace sockets.

Installing:
CVE-2009-3612: Information leak in the netlink subsystem.
Update ID: 9kjtm4z2

Ksplice is now updating the core of Linux—while it's running!
Important bugs are fixed in Linux every week.

CVE-2009-4012: Denial of service in fuse_direct_io.
Update ID: ohk1bd5s

CVE-2009-3620: NULL pointer dereference in ATI Rage 128 driver.
Update ID: f57h6yan

CVE-2009-4005: Buffer overflow in HDLC driver.
Update ID: k9o1hp4a

Last checked: Tue Mar 16 11:49:21 2010

When finished, the Uptrack Manager refreshes to show you the new state of your system, as shown in Figure 4. A green check mark indicates that the named patch was applied successfully and is now live in your running kernel. Click **Check** to look for additional kernel patches available for your system. Click **Close** to dismiss the window. It is not necessary to run `uptrack-manager` as root; the software prompts for an administrator's password before any modifications are applied.

MISSING FIGURE 4

You can open Uptrack Manager any time later with the command `uptrack-manager`, or click on the **K** icon on the taskbar. Ksplice runs continuously once launched. If you prefer, you can configure Ksplice to apply patches to the running kernel automatically or just alert you when updates become available. For example, to choose the former, edit the file /etc/uptrack/uptrack.conf and change the `autoinstall` line to read:

```
autoinstall = yes
```

In addition to `update-manager`, Uptrack Manager includes three other command-line utilities for convenience. (These utilities are different than the tools used to work with kernel source directly, which are demonstrated in the next section.) `uptrack-upgrade` installs the latest available kernel updates, while `uptrack-remove` *`id`* removes the kernel update tagged with ID *`id`*. If you type `uptrack-remove --all`, `uptrack-remove` purges all installed updates. `uptrack-show` shows a manifest of updates that are currently installed.

It is essential to remember that Uptrack Manager does not usurp the role of your distribution's standard package manager. All changes that Ksplice makes are applied to the running kernel and exist in memory only. Therefore, all Ksplice patches are lost upon shutdown. Continue to update your kernel on disk with a traditional software maintenance procedure, such as running:

```
apt-get update; apt-get upgrade
```

Run this as root on a regular basis or when your distribution informs you of new system updates. Keeping the kernel on disk up to date ensures that your system boots the best kernel available when your system eventually requires a restart.

## Updating custom kernels

Uptrack Manager is a convenient option for desktop and server computers based on kernels built and distributed by a vendor or a community project. It won't work, though, if your kernel is customized and built locally, because Uptrack Manager cannot compare your kernel to a known, published operating system to generate splices. However, you can use Ksplice's "primitive" utilities to analyze public patches and create your own splices.

Describing how to build a kernel is beyond the scope of this article. If you're unfamiliar with the process, there are excellent instructions and a good number of satisfactory tutorials scattered across the Internet. Instead, let's focus on the tools Ksplice provides and how to proceed once you are able to build any kernel.

Ksplice provides a utility named `ksplice-create` to create kernel splices. The command requires two inputs: a patch file and the entire source tree for the running kernel, including any previous patches that have already been applied. Given the current source code and the patches, Ksplice generates two kernels, as mentioned above, and compares the revision to its original. The output of `ksplice-create` is a splice that can be applied with `ksplice-apply`. You can redact a splice from the kernel with `ksplice-remove`.

Here is an example (taken from the authors' academic paper and used with express permission) that creates an update for the "prctl vulnerability," CVE-2006-2451, using a patch file called *prctl* and a kernel source directory ~/src:

```
$ ksplice-create --patch=prctl ~/src
Ksplice update tarball written to ksplice-8c4o6u.tar.gz
$ sudo ksplice-apply ./ksplice-8c4o6u.tar.gz
Done!
```

The former command generates a tarball of object code, ksplice-8c4o6u.tar.gz, to be patched into the kernel. The second command reads the tarball, calculates the renovations required, and applies the splice.

## Twenty-four, Seven, Three Hundred Sixty-five

Ksplice is a proven technology. According to the authors' original paper and technical study, Ksplice was able to dynamically patch the Linux® kernel with all security and functional patches issued between 2006 and 2008. Of those, more than two-thirds required no special code to update the kernel. Of the remaining patches, special programming amounted to an average of 17 lines of code per patch.

Linux was chosen to prove the Ksplice approach. The Linux source is widely and frequently vetted for flaws and is patched often to address quickly the shortcomings. However, Ksplice can readily be adapted to any kernel for which source code is available, including BSD and Sun Solaris. Ksplice supports x86, x86_64, and ARM processors, too.

## Resources

### Learn

- Speaking UNIX: Check out other parts in this series.
- Ksplice: Learn more about Ksplice from the tool's site.
- Ksplice technical paper: Read the technical paper to learn more about how Ksplice works.
- kernel.org: Find information about the Linux kernel and the kernel source code archive.
- Common Vulnerabilities and Exposures: An exhaustive list of computer vulnerabilities is available from the CVE site. Each entry in the database includes a brief description of the security vulnerability or exposure and any pertinent references.
- AIX and UNIX developerWorks zone: The AIX and UNIX zone provides a wealth of information relating to all aspects of AIX systems administration and expanding your UNIX skills.
- New to AIX and UNIX? Visit the New to AIX and UNIX page to learn more.
- Technology bookstore: Browse the technology bookstore for books on this and other technical topics.

### Get products and technologies

- Ksplice Uptrack: Download the Uptrack software suitable for your system.
- Github: Github houses thousands of Git repositories, including the personal dot files of many expert users. Search for "dot files" to find examples. (You can learn more about Git online and in the Github Guides.)

### Discuss

- developerWorks blogs: Check out our blogs and get involved in the developerWorks community.
- Follow developerWorks on Twitter.
- Get involved in the My developerWorks community.
- Participate in the AIX and UNIX® forums:
    - AIX Forum
    - AIX Forum for developers
    - Cluster Systems Management

- IBM Support Assistant Forum
- Performance Tools Forum
- Virtualization Forum
- More AIX and UNIX Forums

## About the author

Martin Streicher

Martin Streicher is a freelance Ruby on Rails developer and the former Editor-in-Chief of *Linux Magazine*. Martin holds a Masters of Science degree in computer science from Purdue University and has programmed UNIX-like systems since 1986. He collects art and toys. You can reach Martin at martin.streicher@gmail.com.