Journal of Physics: Conference Series **396** (2012) 042028 doi:10.1088/1742-6596/396/4/042028

# Rebootless Linux Kernel Patching with Ksplice Uptrack at BNL

**Christopher Hollowell**[1], **James Pryor**[1], **Jason Smith**[1]

[1]Brookhaven National Laboratory, Upton, NY 11973, USA

E-mail: hollowec@bnl.gov, pryor@bnl.gov, smithj4@bnl.gov

**Abstract.** Ksplice/Oracle Uptrack is a software tool and update subscription service which allows system administrators to apply security and bug fix patches to the Linux kernel running on servers/workstations without rebooting them. The RHIC/ATLAS Computing Facility (RACF) at Brookhaven National Laboratory (BNL) has deployed Uptrack on nearly 2,000 hosts running Scientific Linux and Red Hat Enterprise Linux. The use of this software has minimized downtime, and increased our security posture. In this paper, we provide an overview of Ksplice's rebootless kernel patch creation/insertion mechanism, and our experiences with Uptrack.

## 1. Introduction

The RHIC/ATLAS Computing Facility at Brookhaven National Laboratory is the primary computing center for the PHENIX and STAR experiments at the Relativistic Heavy Ion Collider (RHIC), and the US Tier One computing facility for the ATLAS experiment at the Large Hadron Collider (LHC). At the time of writing (May 2012), our facility provides our users with a processor farm consisting of roughly 25,000 logical CPUs on approximately 2,000 servers. Most of the farm is reserved for batch system use, with a small subset available for interactive logins.

The users of our facility typically do not checkpoint the progress of their batch and interactive jobs. Because of this, and because jobs at our facility are frequently long running (24 hours or more) our farm nodes cannot be rebooted without advance scheduling of a day or more. The situation is made even more challenging by the fact that many of our processor farm nodes are also dCache, XROOTD, and HDFS data servers. Rebooting more than a small subset of these systems at a time can largely affect data availability.

In order to reduce instantaneous impact on processing and data service resources, we generally schedule such reboots in a "rolling" manner: a subset of the processor farm is closed to new batch jobs and interactive logins, and rebooted once running processes finish. This is an administratively difficult task, and can take many days or weeks to complete. Since we provide services to multiple experiments, reboot scheduling typically involves coordination and compromise with several computing coordinators, further increasing administrative complexity. Linux kernel updates do not take effect until a reboot is performed. Therefore, delays introduced by reboot scheduling can leave systems exposed to kernel security vulnerabilities for some time.

Our solution to reboot scheduling for kernel updates has been to adopt the use of Ksplice/Oracle's Uptrack [1] rebootless kernel patching software and subscription service at our facility. Through the use of this tool, we've been able to avoid rebooting our hosts to correct

many kernel security and bug issues, and have increased our security posture by applying updates as soon as they're available.

## 2. Experience with Uptrack

Uptrack has been in large-scale use at our facility since April 2011. It's primarily being utilized on our processor farm, but is also installed on several critical hosts where reboots are highly disruptive. At the time of writing, we've had a positive experience with the software/subscription service and have applied over 120 updates to our running kernels using it. None of the installed updates have caused any system stability issues (i.e. kernel panics).

Uptrack kernel updates generally become available on or close to the same day they are released by the Scientific Linux (SL) developers. In some cases, the Uptrack updates are released even before they are available for SL. While we have rebooted farm systems during facility downtimes, we've not had to schedule any reboots to patch kernel issues since we started using the product. Updating user space software, in contrast with the kernel, generally does not require reboots or otherwise interfere with running user processes. We've therefore continued to use yum with a local mirror of the SL repositories for such updates.

Oracle/Ksplice has been fairly flexible with our needs. For instance, when we requested that they modify the Uptrack software such that it could utilize a locally caching site proxy, they had no problem implementing this functionality for us. We now receive all our updates through this local server. Also, we were encountering deadlocked processes on a number of our systems due to a known *dentry_stat→nr_unused* corruption bug [2], and requested a rebootless kernel patch to correct the issue. While typically only kernel security updates are automatically provided through the Uptrack subscription service, Ksplice was able to provide us with the requested bug fix update.

## 3. Applying Updates

After installing the Uptrack software on a host, all available kernel updates can be applied by executing a single command: *uptrack-upgrade -y*. The *uptrack-show* command is available to view applied updates locally on a system. However, the Uptrack service also provides a web interface for status monitoring (see figure 1) of all of one's hosts. A RESTful web API is also provided for Uptrack, allowing users to write custom scripts to interface/display system status information. It is not possible to push updates to hosts via the web interface or API, but an "autoinstall" configuration option exists, which will cause an Uptrack cronjob to automatically install updates as they become available.

## 4. Oracle Acquisition

Ksplice was acquired by Oracle on July 21, 2011 [3]. At the time of writing, Oracle continues to support existing Ksplice customers' Red Hat Enterprise Linux (RHEL) and Scientific Linux installations. However, new customers may only obtain licenses for Uptrack use on Oracle Linux (except for a 30-day RHEL trial). Uptrack service also remains available free of charge for the Fedora and Ubuntu Linux distributions, but these distributions are typically used in desktop environments. Since we were an existing customer, the acquisition has thus far had no impact on our operation.

The "raw" Ksplice utility, which allows one to create and insert their own rebootless patches from source, remains available as it was released under the GNU GPL [4]. Assuming there are no changes affecting the layout or semantics of data structures, source kernel patches from Red Hat Enterprise Linux (made available via updated kernel source RPM files on their FTP server) can be used with the utility without modification. According to Ksplice's engineers, this is usually the case for security updates [5]. An example of using the "raw" Ksplice utility is presented in figure 2.

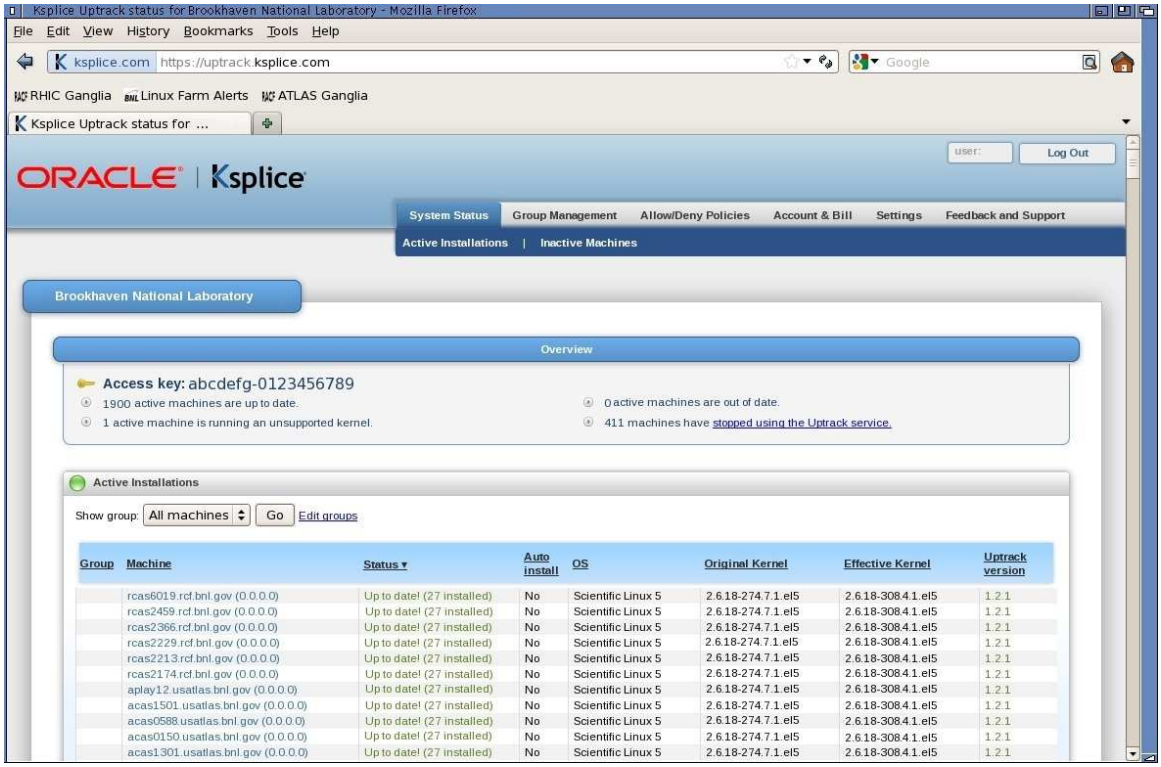


**Figure 1.** Ksplice Uptrack Web Status GUI

## 5. Overview of Ksplice Patch Creation and Insertion

The Ksplice utility creates rebootless kernel patches by compiling the original source for the target kernel with and without the specified source patches, and comparing the binary object files for differences. During compilation, the GCC *–ffunction-sections* and *–fdata-sections* options are passed to the compiler such that each kernel function and data structure is placed in its own ELF section. This simplifies the object code analysis, as the resultant machine code makes no assumptions about the location of data structures and functions in memory. The object code for each modified function is then inserted into a loadable kernel module. When *ksplice-apply* or *uptrack-upgrade* are executed, ksplice kernel modules are loaded which contain the modified functions, as well as code which replaces the first instruction of each original function in memory with a *jmp* instruction to the patched version.

Before the running kernel is modified, a check is performed in kernel space to verify that the original source and object files match up with the running kernel. If they do, the kernel is quiesced via *stop_machine()*. All kernel threads are then checked to verify that their stack traces do not contain references to the soon to be replaced functions. If references to these functions are not found, the *jmp* instructions are inserted, and the updated code takes effect.

## 6. Conclusions

Ksplice/Oracle Uptrack is a useful tool which has helped us minimize the administrative effort and downtime associated with applying Linux Kernel updates. The updates provided have been of a high quality, and caused no instabilities/panics on our hosts. For unsupported distributions, the “raw” Ksplice utility remains available, and is fairly usable with some user effort and skill.

```
% cat /home/build/dmesg_open.patch
--- linux/fs/open.c.orig      2012-04-27 17:02:06.000000000 -0400
+++ linux/fs/open.c   2012-04-27 17:01:51.000000000 -0400
@@ -1180,6 +1180,8 @@
 {
        long ret;

+       printk("Open file[%s,%u]: %s\n", current_thread_info()->task->comm, current_thread_info()->task->pid, filename);
+
        if (force_o_largefile())
                flags |= O_LARGEFILE;
% cd rpmbuild/SPECS
% rpmbuild -bp kernel.spec
...
% cd ../BUILD/kernel-2.6.18
% ksplice-create --id=dmesgopen --patch=/home/build/dmesg_open.patch linux-2.6.18.x86_64
Starting kernel builds (this process might take a long time)...
rm ksplice-revert-stamp
  CHK     include/linux/version.h
  CHK     include/linux/utsrelease.h
...
  LD [M]  /tmp/ksplice-tmp-DsJ57E/kmodsrc/ksplice-dmesgopen_vmlinux-old.ko
make: Leaving directory '/home/build/rpmbuild/BUILD/kernel-2.6.18/linux-2.6.18.x86_64'
make: Entering directory '/home/build/rpmbuild/BUILD/kernel-2.6.18/linux-2.6.18.x86_64'
  INSTALL /tmp/ksplice-tmp-DsJ57E/kmodsrc/ksplice-dmesgopen_vmlinux-new.ko
  INSTALL /tmp/ksplice-tmp-DsJ57E/kmodsrc/ksplice-dmesgopen_vmlinux-old.ko
make: Leaving directory '/home/build/rpmbuild/BUILD/kernel-2.6.18/linux-2.6.18.x86_64'
Ksplice update tarball written to ksplice-dmesgopen.tar.gz
% su -
# ksplice-apply ksplice-dmesgopen.tar.gz
Done!
# cat /etc/redhat-release
Scientific Linux SL release 5.3 (Boron)
# dmesg
...
Open file[cat,22518]: /etc/ld.so.cache
Open file[cat,22518]: /lib64/libc.so.6
Open file[cat,22518]: /usr/lib/locale/locale-archive
Open file[cat,22518]: /etc/redhat-release
Open file[dmesg,22519]: /etc/ld.so.cache
Open file[dmesg,22519]: /lib64/libc.so.6
Open file[dmesg,22519]: /usr/lib/locale/locale-archive
# ksplice-undo dmesgopen
# dmesg
...
Open file[ksplice-undo,22520]: /sys/module/ksplice_dmesgopen/ksplice/stage
Open file[ksplice-undo,22520]: /sys/module/ksplice_dmesgopen/ksplice/stage
 <6>ksplice: Update dmesgopen reversed successfully
```

**Figure 2.** Using the raw Ksplice utility to create and install a custom rebootless patch to *sys_open()* on a machine at BNL. When loaded, the change causes the kernel to log calling process name, pid, and filename each time the *open()* system call is executed.

## References

[1] Ksplice: http://www.ksplice.com
[2] Red Hat Bugzilla Ticket #596548: https://bugzilla.redhat.com/show_bug.cgi?id=596548
[3] Oracle and Ksplice: http://www.oracle.com/us/corporate/Acquisitions/ksplice/index.html
[4] Ksplice Source Distribution: http://oss.oracle.com/ksplice/software/
[5] Arnold, J and Kaashoek, M F 2009 Ksplice: Automatic Rebootless Kernel Updates *Proceedings of the ACM EuroSys Conference (EuroSys 2009)*