Linux Kernel Live Patching mit kpatch und kGraft

GUUG-Frühjahrsfachgespräch 2015

26. März 2015

Stefan „seife“ Seyfried
Linux Consultant & Developer
B1 Systems GmbH
seife@b1-systems.de

Christan Rost
Linux Consultant
B1 Systems GmbH
rost@b1-systems.de

Vorstellung B1 Systems

- gegründet 2004
- primär Linux/Open Source-Themen
- national & international tätig
- über 60 Mitarbeiter
- unabhängig von Soft- und Hardware-Herstellern
- Leistungsangebot:
  - Beratung & Consulting
  - Support
  - Entwicklung
  - Training
  - Betrieb
  - Lösungen
- dezentrale Strukturen

Schwerpunkte

- Virtualisierung (XEN, KVM & RHEV)
- Systemmanagement (Spacewalk, Red Hat Satellite, SUSE Manager)
- Konfigurationsmanagement (Puppet & Chef)
- Monitoring (Nagios & Icinga)
- IaaS Cloud (OpenStack & SUSE Cloud & RDO)
- Hochverfügbarkeit (Pacemaker)
- Shared Storage (GPFS, OCFS2, DRBD & CEPH)
- Dateiaustausch (ownCloud)
- Paketierung (Open Build Service)
- Administratoren oder Entwickler zur Unterstützung des Teams vor Ort

Was ist Live Patching?

Was ist Live Patching?

- Kernel Update ohne Reboot
- Warum nicht einfach „gefixtes Modul laden“?
  - Das alte Modul muss dazu entladen werden
  - Schwierig z. B. beim Treiber für das Root-Dateisystem
  - Unmöglich bei fest einkompilierten Kernelteilen

Warum Kernel Live Patching?

Warum Kernel Live Patching?

- Warum überhaupt Kernel Updates?
  - Security Fixes
  - Bugfixes
  - Neue Funktionen, Verbesserung vorhandener Funktionen
- Warum Live und nicht Rebooten?
  - „Business Critical“ Anwendungen
    $\Rightarrow$ keine Downtime möglich oder gewünscht
  - Keine HA Konfiguration/Applikation nicht HA-fähig
  - Lange laufende Berechnungen/Simulationen
  - Manager: „Müssen wir wirklich rebooten?“

Welche Möglichkeiten gibt es?

Welche Möglichkeiten gibt es?

(Chronologische Reihenfolge)

- Ksplice
- kGraft (in SLES 12)
- kpatch (in RHEL 7)
- Kernel Live Patching (ab Kernel 4.0)

Ksplice

- Erstes Release 2008
- Software war Open Source, Ksplice, Inc. bot Service an
- 2011 von Oracle gekauft
- Nur noch für Oracle Linux, RHEL, Fedora und Ubuntu verfügbar
- Letztes Release 2011

kGraft

- Erste Ankündigung Januar 2014
- Erstes Release März/April 2014
- In SUSE Linux Enterprise Server 12 enthalten (extra Subscription, nicht im Default-Supportumfang)
- Für den Mainline-Linux-Kernel eingereicht im April 2014

kpatch

- Erstes Release Februar 2014
- In Red Hat Enterprise Linux 7.0 als „technology preview“ enthalten
- Für den Mainline-Linux-Kernel eingereicht im Mai 2014

Kernel Live Patching

- Im Mainline Linux Kernel ab 4.0 (April 2015)
- „Schnittmenge“ aus kpatch und kGraft
- In Zusammenarbeit von SUSE, Red Hat und der Kernel Community

Technische Details

Ksplice

- Userspace
- kompiliert original und gepatchten Code
- Vergleich von Quellcode und erzeugtem Binary
- Erzeugt Patch-Modul
- „`stop_machine()` based patching“
- Funktionspointer werden direkt umgebogen (alle anderen Methoden verwenden die FTrace-Infrastruktur)

kGraft

- Der Ziel-Kernel muss mit `CONFIG_KGRAFT` kompiliert sein
- Der Patch-Code wird für den jeweiligen Fix geschrieben
- Erzeugt ein Patch-Modul
- Userspace-Tools verfügbar, um Patch-Code zu erstellen
- Deren Benutzung ist momentan eine „Herausforderung“
- kein „`stop_machine()`“, stattdessen fließender Übergang von „nicht gepatcht“ zu „gepatcht“

kpatch

- Der Ziel-Kernel muss nicht besonders vorbereitet sein
- Tools verfügbar, die aus dem Diff einen binären Patch erzeugen
- Erzeugt ein Patch-Modul und ein Modul „`kpatch.ko`“, das die Infrastruktur bereitstellt
- Der Patch-Code ist dabei nicht „sichtbar“, alles geht automatisch
- Die Userspace-Tools funktionieren, zumindest für relativ einfache Beispiele und auf den supporteten Distributionen
- Patches können rückgängig gemacht und das Patch-Modul entladen werden
- „`stop_machine()` based patching“

Kernel Live Patching

- Ab Kernel 4.0 Upstream
- Zu patchender Kernel muss mit `CONFIG_LIVEPATCH=y` konfiguriert sein
- Noch keine Userspace-Tools verfügbar, Patch muss manuell erzeugt werden
- Patches können an- und ausgeschaltet werden, aber das Patch-Modul kann (noch) nicht wieder entladen werden

(einige) Details zur Implementierung

Gemeinsamkeiten der Implementierungen

(KSplice nicht beachtet)

- FTrace-Infrastruktur wird benutzt, um gepatchte Funktionsaufrufe umzuleiten
- Problematisch: Patches, die die Datenstrukturen im Kernel ändern
- Problem: Konsistenz beim Umschalten zwischen „ungepatcht“ und „gepatcht“

Unterschiede der Implementierungen

Konsistenz beim Übergang

Zu lösendes Problem: die zu patchende Funktion darf in diesem Moment *nicht* genutzt werden

- Kernel Live Patching: noch nicht implementiert
- kpatch: alle Prozesse werden angehalten, geprüft ob alle Prozesse in einem „sicheren“ Zustand sind, wenn ja wird gepatcht und alles wieder gestartet
- kGraft: für jeden Prozess wird einzeln „umgeschaltet“, wenn er gerade patchbar ist. Prozesse müssen eventuell mit einem Signal geweckt werden, damit sie in den gepatchten Modus wechseln

Unterschiede der Implementierungen

Konsistenz beim Übergang

Zu lösendes Problem: die zu patchende Funktion darf in diesem Moment *nicht* genutzt werden

- kpatch: es ist theoretisch möglich, dass ein Patch nicht angewendet werden kann, weil nie alle Prozesse gleichzeitig die kritische Funktion verlassen
- kGraft: es ist theoretisch möglich, dass ein Patch niemals „fertig“ wird, weil ein Prozess nicht aufgeweckt werden kann. Alle anderen Prozesse benutzen dann aber schon den gepatchten Code.

Unterschiede der Implementierungen

Kernel Live Patching (in Kernel 4.0):

```
This first version does not implement any consistency mechanism
that ensures that old and new code do not run together. In practice,
~90% of CVEs are safe to apply in this way, since they simply add a
conditional check. However, any function change that can not execute
safely with the old version of the function can _not_ be safely
applied in this version.
```

Vielen Dank für Ihre Aufmerksamkeit!

Bei weiteren Fragen wenden Sie sich bitte an info@b1-systems.de
oder +49 (0)8457 - 931096