Elivepatch
Flexible distributed Linux Kernel live patching

Alice Ferrazzi
kernel :~ $ whoami

Alice Ferrazzi

- Gentoo
  - Gentoo Kernel Project Leader
  - Gentoo Kernel Security
  - Gentoo Foundation board member
  - Gentoo Google Summer of Code administrator and mentor for rust Gentoo project
- Cybertrust Japan
  - OSS Embedded Software Engineer
Summary

- Live patch explanation
- Current live patch services
  - Motivation for elivepatch
- Elivepatch solution
  - Implementation
  - Challenge
  - Status
  - Future
- Conclusion
At first this project was part of Google Summer of Code 2017 for the Gentoo organization.
Live patch explanation
Live patch

Modify the kernel without the need to reboot.
Why

- Downtime is expensive (containers, supercomputers)
- Security (vulnerability time shorter)
Where

- Embedded
- Desktops
- HPC (complex scientific computations)
- Cloud
- Any computer under heavy load
What

Kpatch

Livepatch

Kgraft
Kgraft

Suse Open Source live patching system that is routing the old function gradually.
Kpatch

Red Hat Open Source live patching system and use ftrace and stop_machine() for route functions toward the new function version.
Livepatch

Livepatch is a hybrid of kpatch and kgraft. Livepatch has been merged into the kernel upstream.

Kpatch-build can work with both kpatch and livepatch for creating the live patch.
Livepatch is just a module
Livepatch module problem

A module that takes just about 1+ hour to compile in a modern server
At Gentoo, we know what means to compile something for more than 1 hour…
Gentoo solution to compile for 1+ hour compilation problem

- Gentoo “binary host”
- Pre-compiled binary
What options do we have for compiling livepatch modules?
Current existing livepatch services
Current vendor solutions

- Oracle, Ksplice (support only Oracle Linux kernels)
- Suse Linux Enterprise Live Patching (support only Suse Kernels for one year)
- Canonical Live Patch (support only Ubuntu 16.04 LTS and Ubuntu 14.04 LTS)
- Red Hat live patch (Support only Red Hat kernel)
Motivation for elivepatch
Problems of vendor solutions

- trusting on third-party vendors
- Lacking support for custom kernel configurations
- Lacking support for request-driven customization
- Lacking long term support
- Closed source
Vendor solutions representation

Client

Loadable Patch Module

Third Party Live Patch Server
elivepatch solution
elivepatch

A web service framework to deliver Linux kernel live patches

● Supports custom kernel configurations
● User participation via request-driven customization
● Open source
Elivepatch solution

```
{
  "Request Payload": {
    "Kernel Version": "4.14.16",
    "Kernel Configuration": "/tmp/kernel.conf",
    "Patch": {
      "Main patch": "/tmp/main.patch",
      "Incremental patch": [
        "/tmp/elivepatch/0001.patch",
        "/tmp/elivepatch/0002.patch"
      ]
    }
  }
}
```

Live Patch Request

Elivepatch Client

Loadable Patch Module

Elivepatch Server

RESTful API

Kpatch-build
Implementation
Elivepatch-server (Main language: Python)
Flask + Flask-Restful + Werkzeug

Elivepatch-client (Main language: Python)
Requests + GitPython
Challenges
Challenges with elivepatch

- Some patches require manual modification to be converted to live patches
- Reproducing the build environment can be difficult:
  - Differences in compiler versions
  - Variations in the compiler and optimization flags
- Incompatible machine architectures (solaris, hpc)
Incompatibility with GCC

CCFLAGS and non vanilla gcc, can sometime break elivepatch.
Current status
Elivepatch status

- First open source release 0.1 on 2017/9/06
- Packaged for Gentoo
- **Kpatch version 0.6.2 in Gentoo**
- Presented as poster at SOSP 2017
- Close collaboration with kpatch maintainers
<table>
<thead>
<tr>
<th>Package</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CentOS 7</td>
<td>0.4.0</td>
</tr>
<tr>
<td>Debian Stable</td>
<td>0.3.2</td>
</tr>
<tr>
<td>Debian Testing</td>
<td>0.6.0</td>
</tr>
<tr>
<td>Debian Unstable</td>
<td>0.6.0</td>
</tr>
<tr>
<td>Deepin</td>
<td>0.3.2</td>
</tr>
<tr>
<td>Devuan Stable</td>
<td>0.3.2</td>
</tr>
<tr>
<td>Devuan Testing</td>
<td>0.6.0</td>
</tr>
<tr>
<td>Devuan Unstable</td>
<td>0.6.0</td>
</tr>
<tr>
<td>Funtoo</td>
<td>0.6.2</td>
</tr>
<tr>
<td>Gentoo</td>
<td>0.6.2</td>
</tr>
<tr>
<td>Kali Linux Rolling</td>
<td>0.6.0</td>
</tr>
<tr>
<td>Pardus</td>
<td>0.3.2</td>
</tr>
<tr>
<td>Parrot</td>
<td>0.6.0</td>
</tr>
<tr>
<td>PureOS green</td>
<td>0.6.0</td>
</tr>
<tr>
<td>PureOS landing</td>
<td>0.6.0</td>
</tr>
<tr>
<td>Raspbian Stable</td>
<td>0.3.2</td>
</tr>
<tr>
<td>Raspbian Testing</td>
<td>0.6.0</td>
</tr>
<tr>
<td>Rosa Server 7.3</td>
<td>0.3.2</td>
</tr>
<tr>
<td>Scientific Linux 7.x</td>
<td>0.4.0</td>
</tr>
<tr>
<td>Trisquel 8.0</td>
<td>0.3.2</td>
</tr>
<tr>
<td>Ubuntu 16.04</td>
<td>0.3.2</td>
</tr>
<tr>
<td>Ubuntu 17.10</td>
<td>0.3.2</td>
</tr>
<tr>
<td>Ubuntu 18.04</td>
<td>0.5.0</td>
</tr>
<tr>
<td>Ubuntu 18.10</td>
<td>0.5.0</td>
</tr>
<tr>
<td>Ubuntu 19.04</td>
<td>0.5.0</td>
</tr>
</tbody>
</table>
Future

What elivepatch needs
Future

- livepatch automatization
- Multi distribution
- Livepatch signing
- Kernel CI\CD check
- Elivepatch overlay
livepatch automatization

- Automatize the livepatch creation when there are no semantic changes.
- Tool for creating the extra relocations entries.
Multi distribution

Solve distributions compatibility issues

Current target:

- Debian
- Fedora
- Gentoo
- Android
Elivepatch client on Debian

Work in progress...

https://asciinema.org/a/187738

p.s. Gentoo kernel is still needed
Livepatch signing

- Implementing livepatch module signing in the server
- Implementing signing verification for the client
Kernel CI/CD checking

- Implement a buildbot plugin for testing elivepatch
- Implementing elivepatch-server on docker, for a ready to use livepatch building instance

[You can test your livepatch with the same settings and hardware as where you want to deploy it]
elivepatch overlay

Collaborative livepatch creation

Similar to Gentoo overlay for livepatch
example:

https://github.com/aliceinwire/elivepatch-overlay
Conclusion
Epilogue

- Livepatch is a module that takes time compiling
- Livepatch vendor service solutions solve the compilation problem in a proprietary way
- Elivepatch offers a wider solution
With the diffusion of embedded systems and robotics, Livepatch services will become always more important.
https://github.com/gentoo/elivepatch-client

Please send every issues you found
We opened the first elivepatch server node:

http://elivepatch.amd64.dev.gentoo.org:5000
If you are interested in contributing, Elivepatch is welcoming every form of contribution.