The Best Security Innovation of 2009

Ksplice

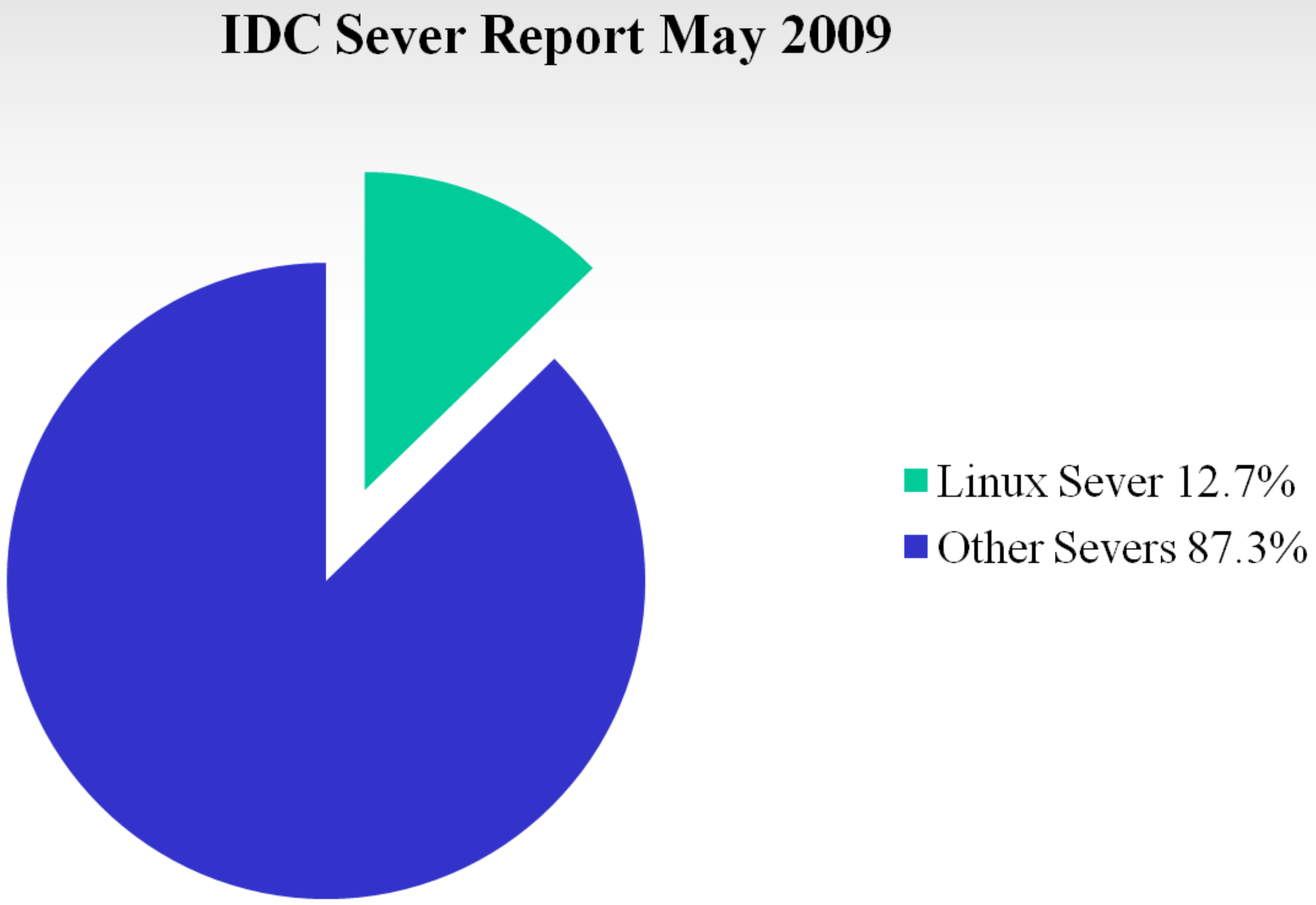
By

Yang Huang (Chloe), Minh Tuan Pascal Nguyen and Yifan Mei (Sam)

Group 15

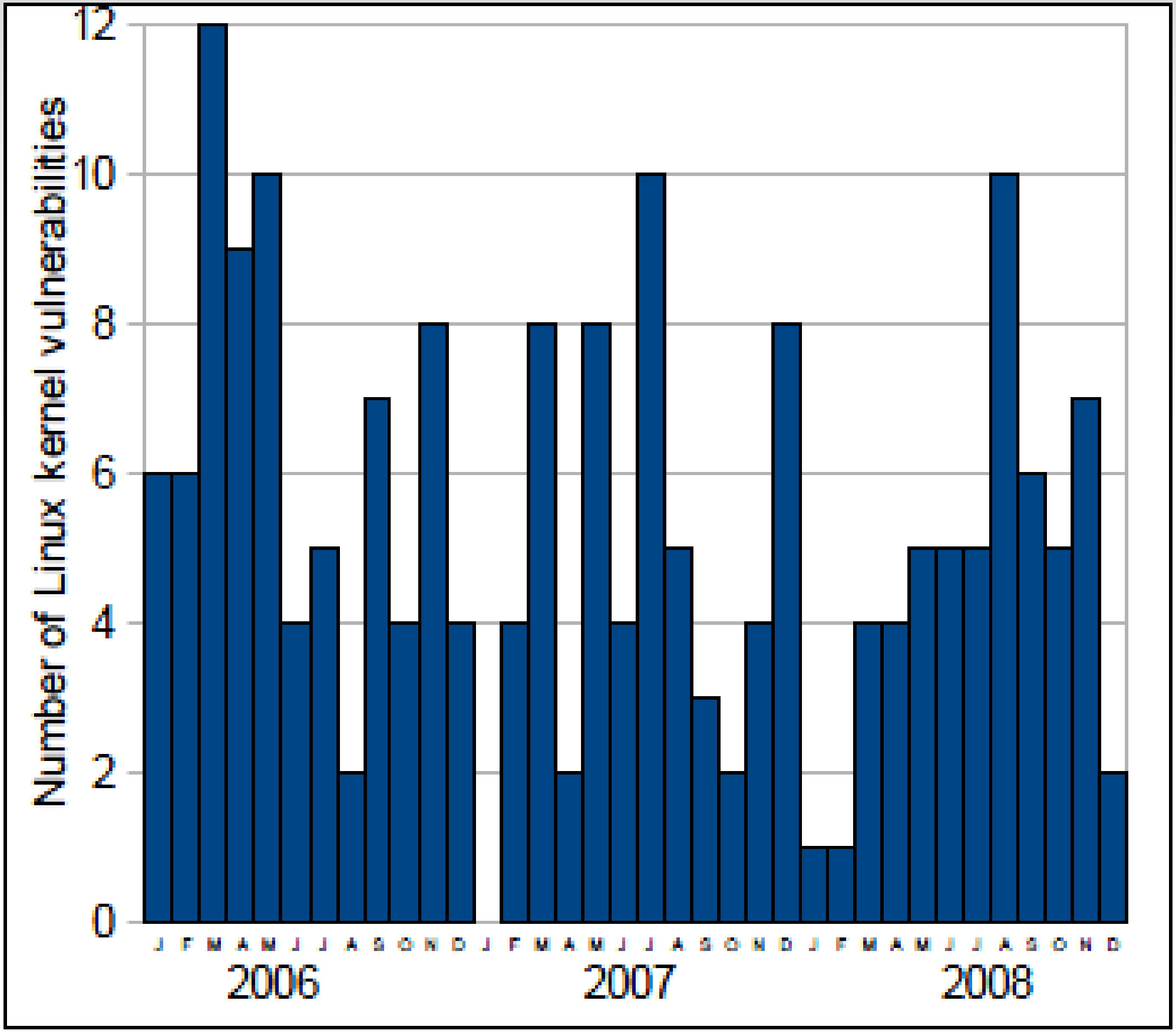
29/10/2009

Proportion of Linux Sever in the market



IDC Sever Report May 2009

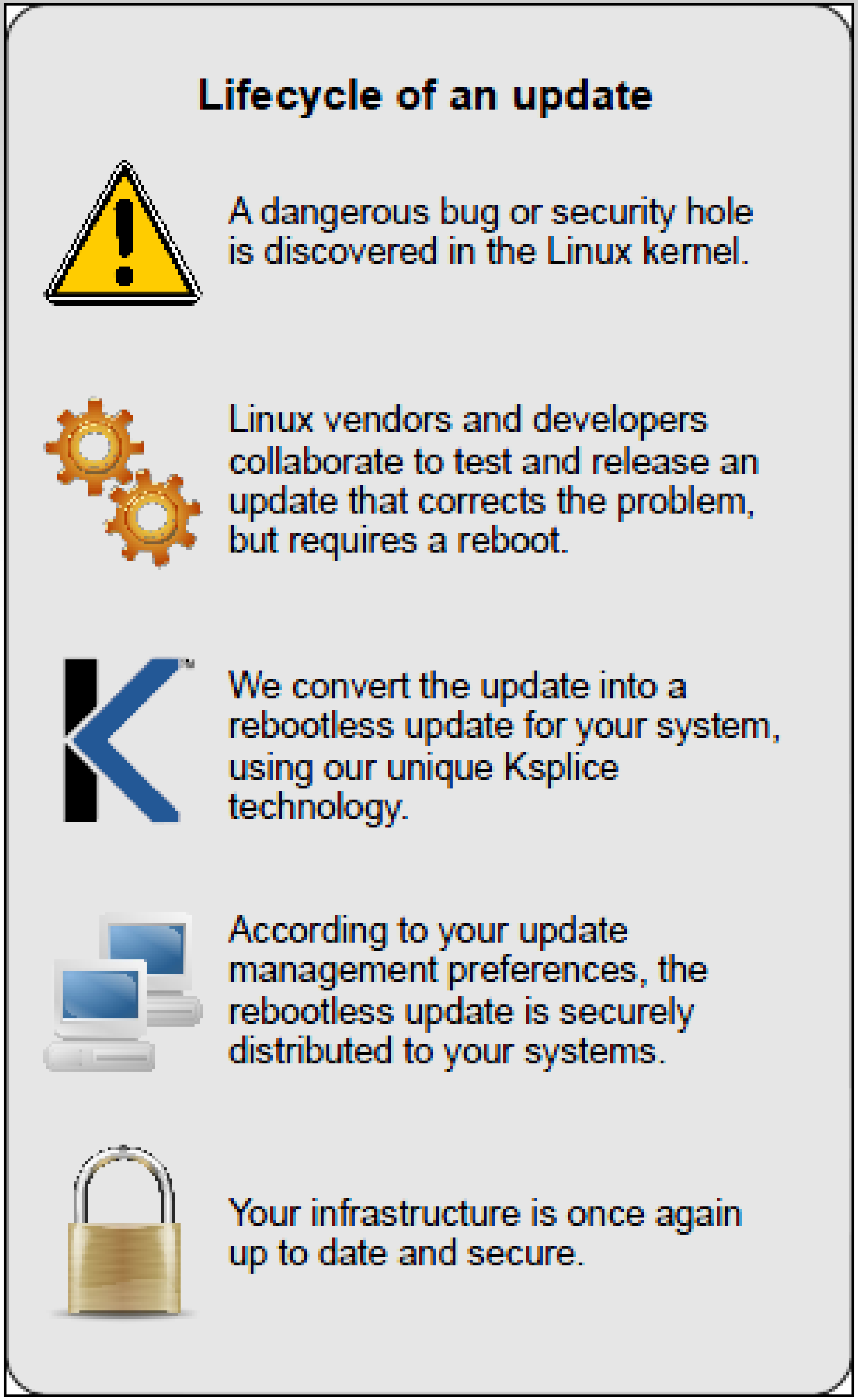
Vulnerabilities/Time Graph



What is Ksplice?

- Ksplice Uptrack is an update management solution that lets you effortlessly keep your Linux infrastructure up to date with the latest important security and reliability updates.

- Ksplice streamlines software updates by eliminating the need for disruptive reboots, reducing maintenance costs and increasing system dependability.



Lifecycle of an update

A dangerous bug or security hole is discovered in the Linux kernel.

Linux vendors and developers collaborate to test and release an update that corrects the problem, but requires a reboot.

We convert the update into a rebootless update for your system, using our unique Ksplice technology.

According to your update management preferences, the rebootless update is securely distributed to your systems.

Your infrastructure is once again up to date and secure.

Quotes

- “Ksplice is truly first class. … One day in the future when Ksplice becomes the norm, we’ll look back and wonder what we ever used to do.”

  —Christopher Smart, Linux Magazine

- I run busy servers and Ksplice has increased uptime and decreased my ‘worry time’. I don’t have to get up at 3 a.m. to make major changes. How great is that!

  —Michael Alft, Alft&Wilson Publishing

How Hot is Ksplice

- Ksplice construct hot updates at object code level

- Check the update patch change the Data structure or not /A programmer write new codes

- Ksplice does not require any preparation before update

- Ksplice performs replacements on entire functions

How does Ksplice work?

- Ksplice uses pre-post differencing to generate object code for the update

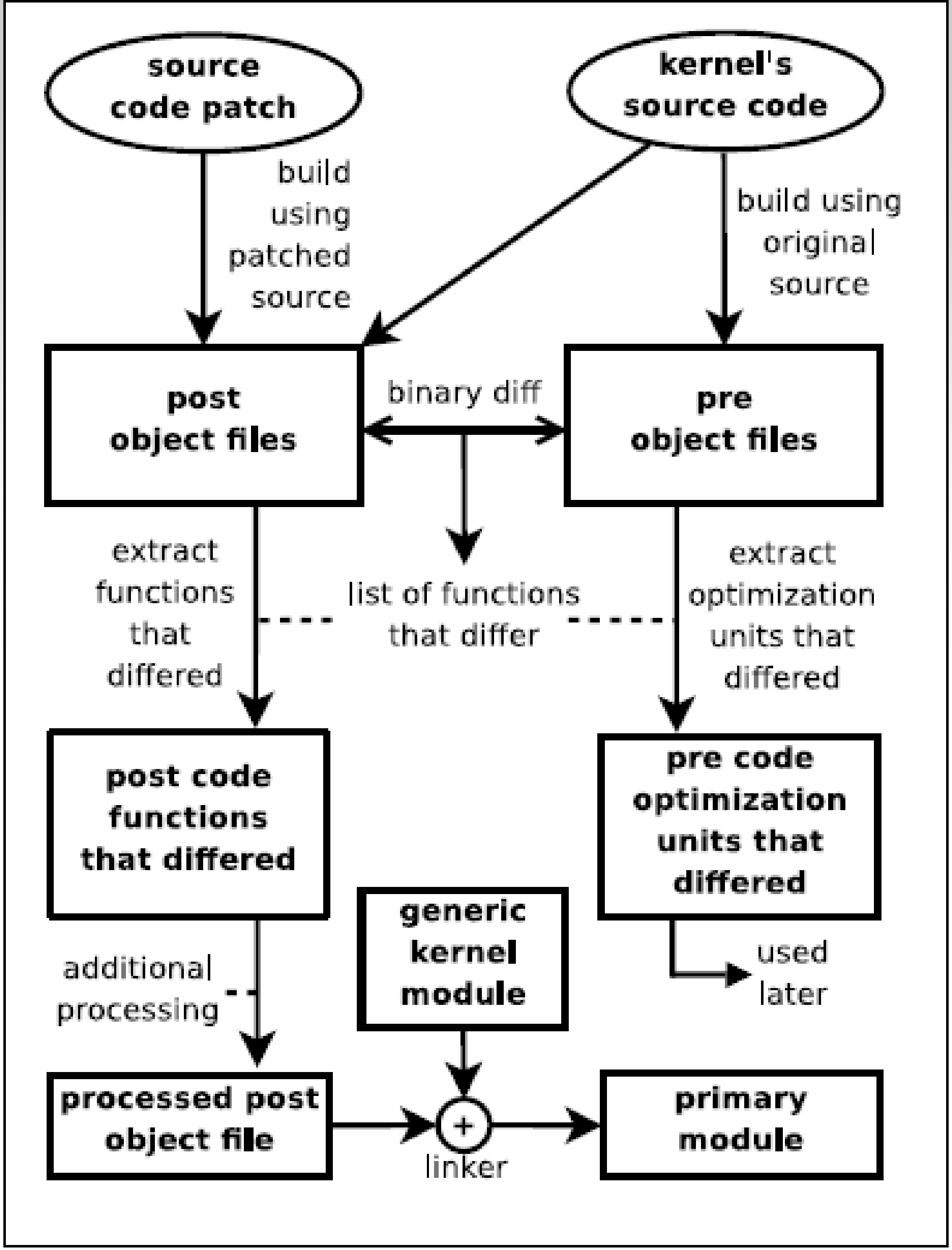
- Ksplice resolves symbols correctly and provides safety using run-pre matching

Pre-Post Difference

Ksplice compares the kernel’s binary code:

- Before applying the patch (pre)
- After applying the patch (post)

Generating replacement code based on a patch


source code patch
kernel's source code
build using patched source
build using original source
post object files
binary diff
pre object files
extract functions that differed
list of functions that differ
extract optimization units that differed
post code functions that differed
pre code optimization units that differed
generic kernel module
used later
additional processing
processed post object file
+
linker
primary module

Avoid big differences

- Avoid extraneous differences for practicality.
- Be able to generate object code that makes no assumptions about where functions are stored.
- Important for creating the replacement code, this then references existing functions.
- Certain compiler options are enabled to reduce location assumptions.

What happens after?

- Ksplice extracts the changed functions from the post object.
- Puts them into their own object file
- Ksplice creates a kernel module called *primary module* that will load processed post code into the kernel.
- Last step before the file can be used…

Safety verification

- when a hot update system makes unchecked assumptions about the run code

- Ksplice detects all differences between the pre code and the post code

- adding a step to the hot update process to check the run code against the pre code

Run-Pre Matching

Run-Pre Matching technique is used to address the following challenges:

- How to resolve symbols in the replacement code

- How to verify the safety of an update

Hot update

- run-pre check will be safe if:
  - using exactly the same compiler version
  - assembler version
  - compiler options

Operating of run-pre matching

- matching system must be able to recognize no-op instruction sequences

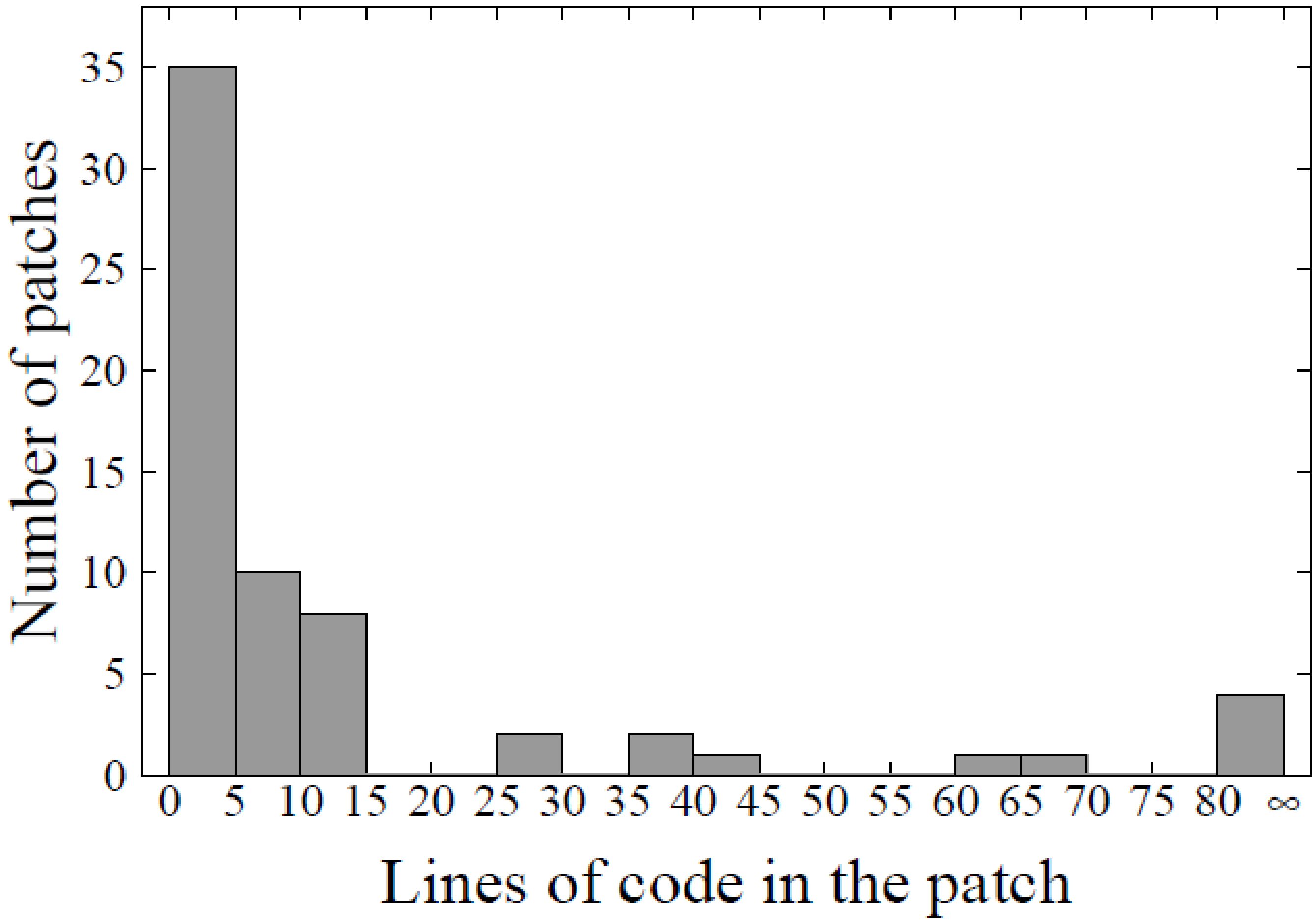
- must know basic information about the instruction set

Stackholders

- Common Linux users:
  - Less distruption while working.
- System Adminstrators:
  - No more downtime.
  - No more loss of state (eg network connections).
  - No more unexpected complications.
- Cybercriminals:
  - Exploit known vulnerablities if system not updated.

Evaluation

Figure 3: Number of patches by patch length



Evaluation(continue)

Most Linux kernel security vulnerabilities can be corrected by modifying relatively few lines of source code. Of the 64 vulnerabilities from this time interval, 53 vulnerabilities were corrected in 15 or fewer lines of source code changes, and 35 vulnerabilities were corrected in 5 or fewer lines of changes.

Conclusion

- No advance preparation required
- Virtualization-ready
- No negative performance impact
- Rollback capability

Top awards for Ksplice

- The Wall Street Journal named Ksplice the gest security innovation of 2009
- The Global Security Challenge declared Ksplice the winner of its 2009 Global Cybersecurity Challenge
- The Trophees du Libre international software competition named Ksplice the best security software of 2009 and awarded Ksplice a special prize.
- In 2009, Ksplice became the first software product to win the Massachusetts Institute of Technology $100K competition in more than 10 years.

Reference

1. http://www.ksplice.com/
2. http://www.ksplice.com/uptrack/
3. http://www.ksplice.com/uptrack/howitworks
4. http://www.ksplice.com/uptrack/faq
5. http://www.ksplice.com/doc/ksplice-brochure.pdf
6. http://www.ksplice.com/doc/ksplice.pdf

Thanks for Listening!