LZ4 Compression and Improving Boot Time

Kyungsik Lee
Software Platform Laboratory,
LG Electronics

Contents

- Introduction
- Kernel Compression
- LZ4 in Kernel
- Improving Performance
- Measurement
- Case Study
- Potential users of LZ4
- Challenge & Next Step

Introduction

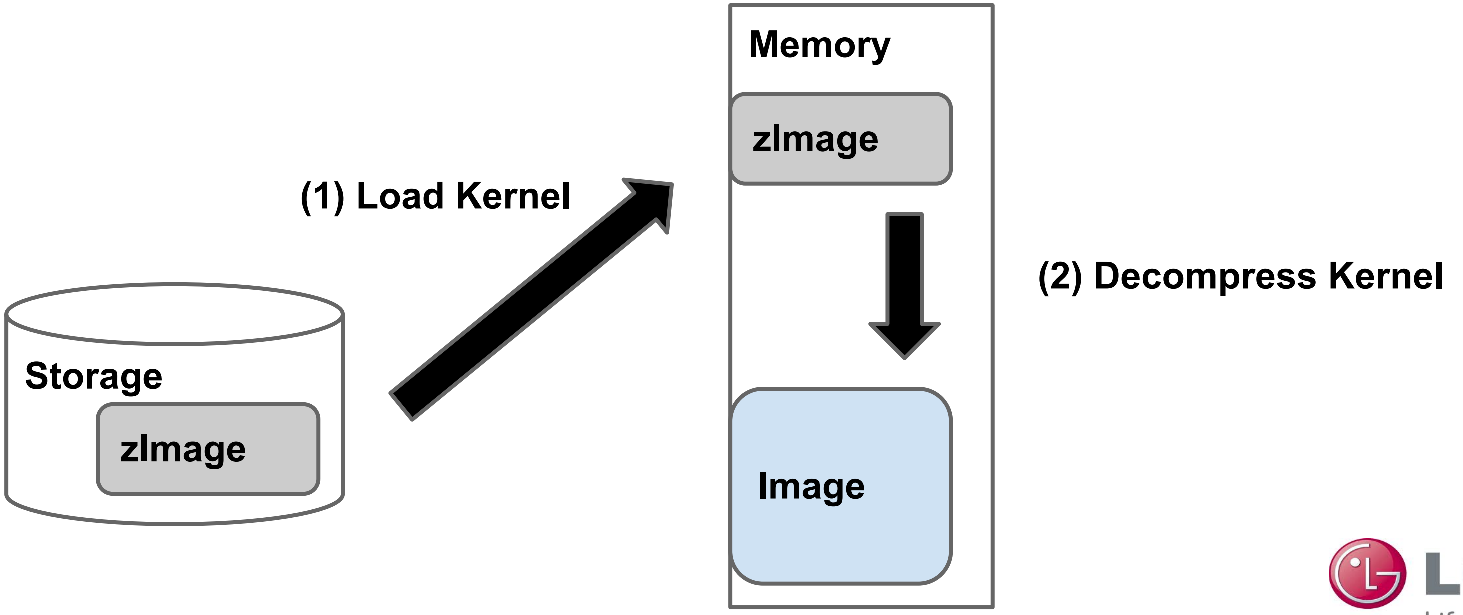
Why does Boot Time matter?

- Boot Time is very important in many consumer electronics products.
- Regardless of how well designed a consumer electronic device is, the time required to move the device from off to usable state is critical to obtaining a positive user experience.

Kernel Compression (1/2)

Can booting from uncompressed kernel can improve Boot Time?

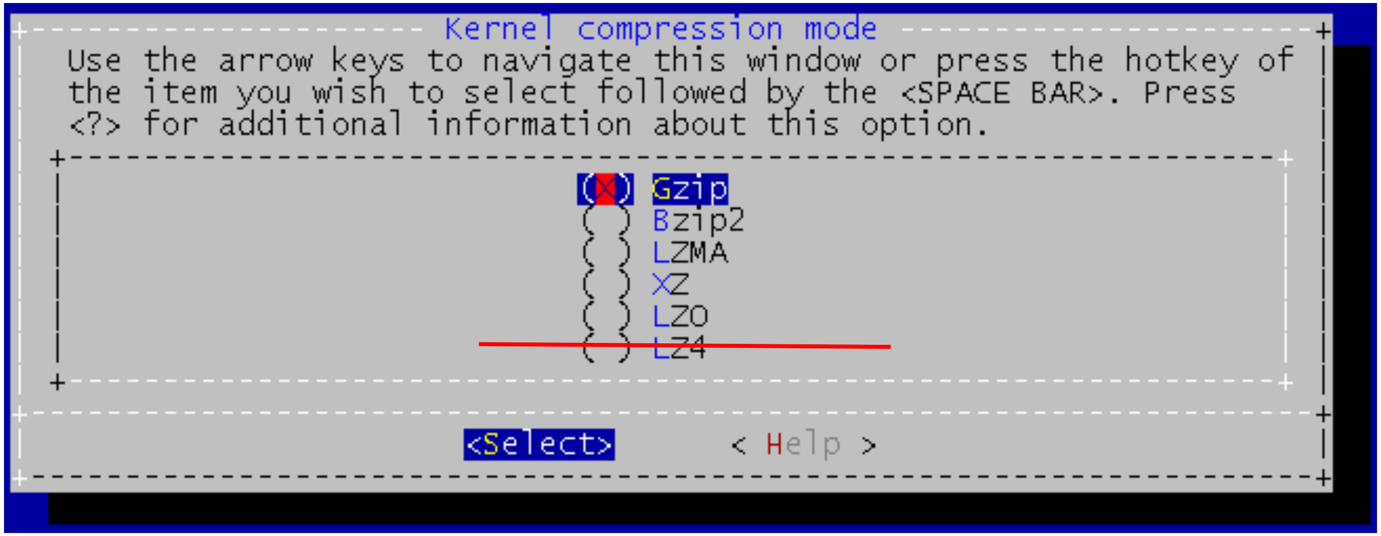
- In storage, the Kernel image is usually compressed to save space. (Reduced by 58% with gzip)
- Generally it takes longer to load Kernel than decompression of Kernel (Improved by 2% with gzip)
- **Boot Time can be reduced by using Compressed Kernel**



Kernel Compression (2/2)

Which one of these is the best for Boot Time?

- Compression algorithm's ratio; Kernel size
- Decompression speed
- ~~Compression speed~~



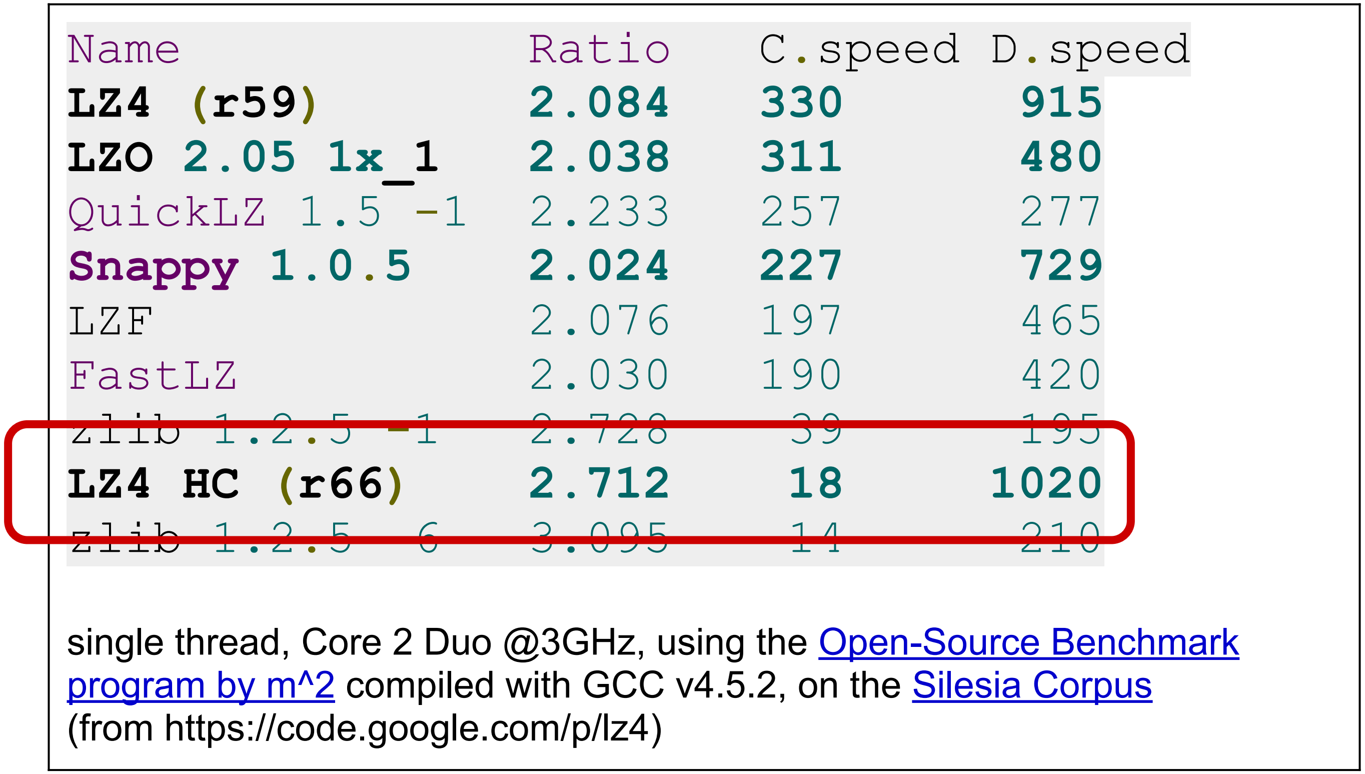
Kernel compression mode

Use the arrow keys to navigate this window or press the hotkey of the item you wish to select followed by the <SPACE BAR>. Press <?> for additional information about this option.

(X) Gzip
( ) Bzip2
( ) LZMA
( ) XZ
( ) LZO
~~( ) LZ4~~

Quick Comparison

Can LZ4 reduce the Kernel decompression time?



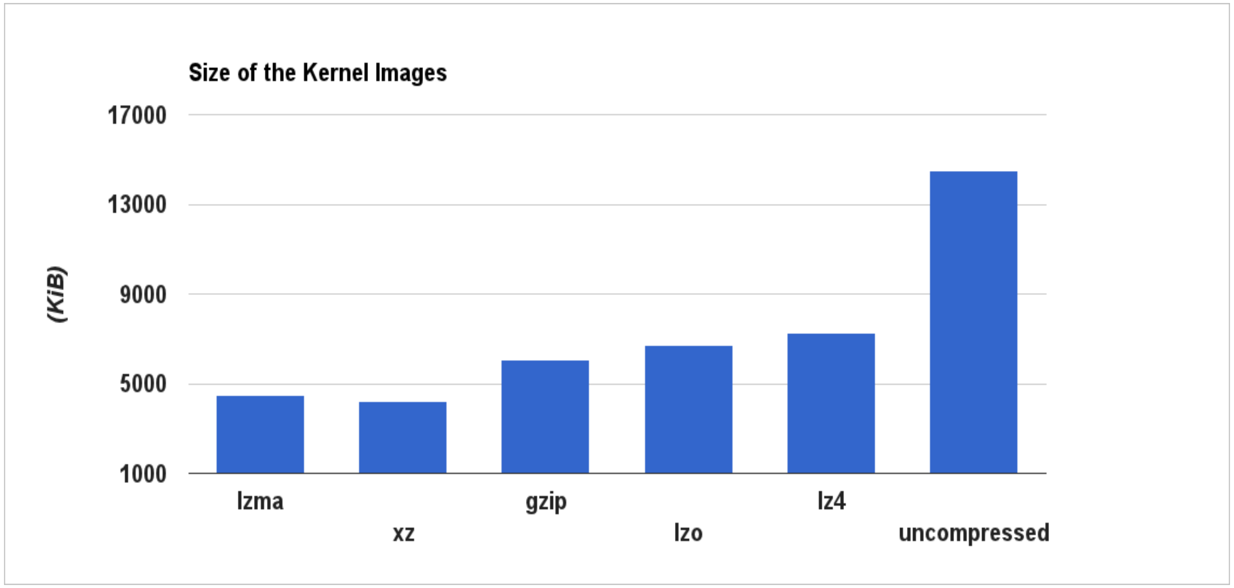
Name	Ratio	C.speed	D.speed
LZ4 (r59)	**2.084**	**330**	**915**
LZO 2.05 1x_1	**2.038**	**311**	**480**
QuickLZ 1.5 -1	2.233	257	277
Snappy 1.0.5	**2.024**	**227**	**729**
LZF	2.076	197	465
FastLZ	2.030	190	420
zlib 1.2.5 -1	2.728	39	195
LZ4 HC (r66)	**2.712**	**18**	**1020**
zlib 1.2.5 -6	3.095	14	210

single thread, Core 2 Duo @3GHz, using the Open-Source Benchmark program by m^2 compiled with GCC v4.5.2, on the Silesia Corpus
(from https://code.google.com/p/lz4)

LZ4: Extremely Fast Decoder

Description

- LZ4 is a very fast compressor, based on well-known LZ77 (Lempel-Ziv) algorithm. (Author: Yann Collet)
- It features an extremely fast decoder.
- This algorithm favors speed over compression ratio in Kernel.



Support for LZ4-compressed Kernel (1/3)

- ARM and x86 are supported for LZ4 compressed-Kernel and initramfs
- 20 files changed, 663 insertions(+), 3 deletions(-)
- Current Status: linux-next

```
arch/arm/Kconfig                      |   1 +
arch/arm/boot/compressed/.gitignore   |   1 +
arch/arm/boot/compressed/Makefile     |   3 +-
arch/arm/boot/compressed/decompress.c |   4 +
arch/arm/boot/compressed/piggy.lz4.S  |   6 +
arch/x86/Kconfig                      |   1 +
arch/x86/boot/compressed/Makefile     |   5 +-
arch/x86/boot/compressed/misc.c       |   4 +
include/linux/decompress/unlz4.h      |  10 ++
include/linux/lz4.h                   |  62 +++++++++++
init/Kconfig                          |  13 ++-
lib/Kconfig                           |   7 ++
lib/Makefile                          |   2 +
lib/decompress.c                      |   5 +
lib/decompress_unlz4.c                | 199 ++++++++++++++++++++++++++++++++++
lib/lz4/Makefile                      |   1 +
lib/lz4/lz4_decompress.c              | 199 ++++++++++++++++++++++++++++++++++
lib/lz4/lz4defs.h                     | 129 ++++++++++++++++++++++
scripts/Makefile.lib                  |   5 +
usr/Kconfig                           |   9 ++
```

Support for LZ4-compressed Kernel (2/3)

Kernel Compression

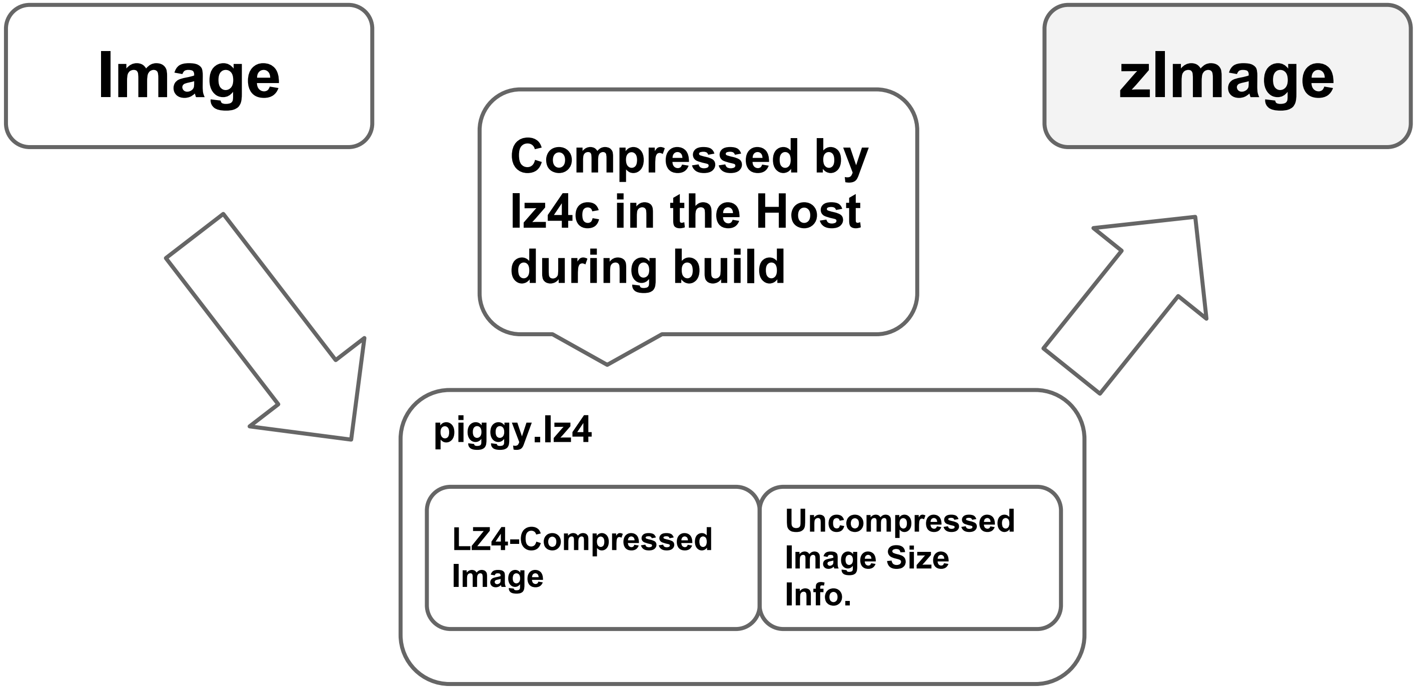
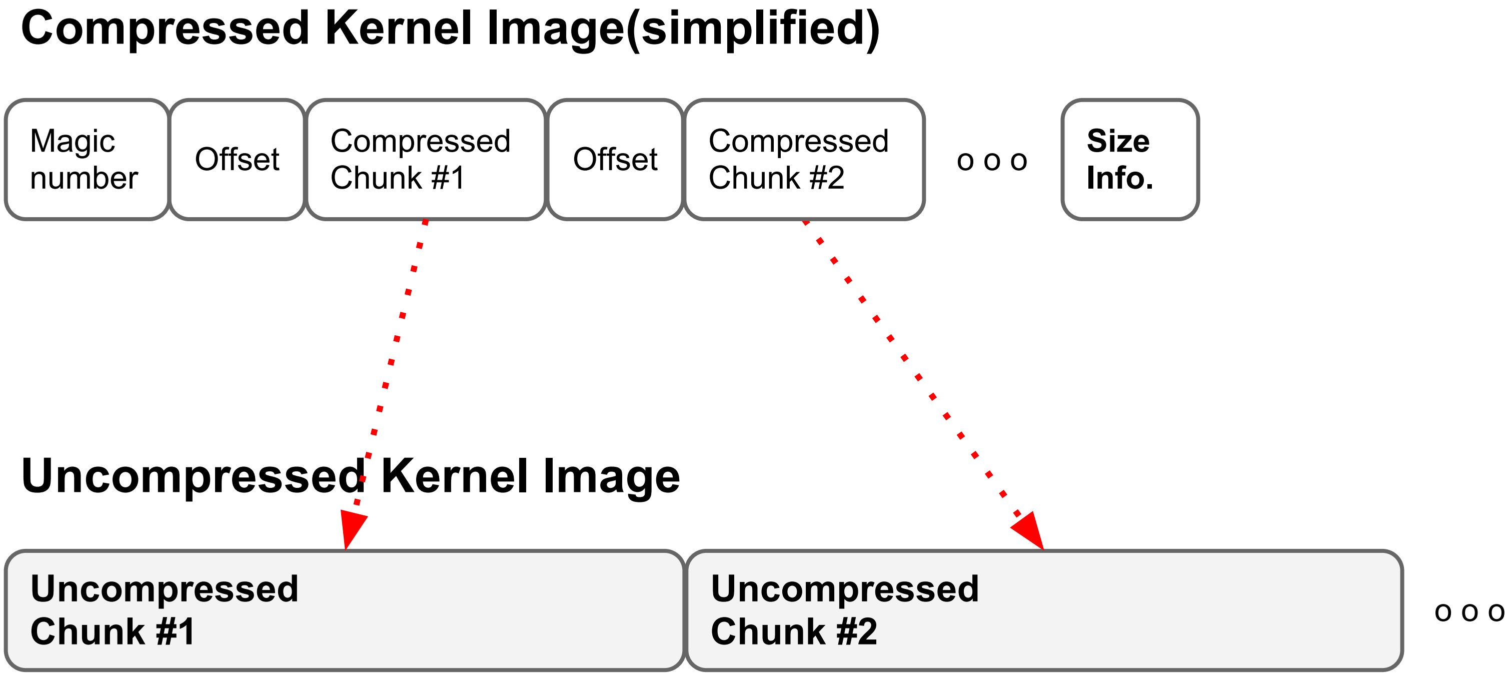


Image
Compressed by lz4c in the Host during build
zImage
piggy.lz4
LZ4-Compressed Image
Uncompressed Image Size Info.

Support for LZ4-compressed Kernel (3/3)

Kernel Decompression



Compressed Kernel Image(simplified)

Magic number | Offset | Compressed Chunk #1 | Offset | Compressed Chunk #2 | o o o | **Size Info.**

Uncompressed Kernel Image

Uncompressed Chunk #1 | **Uncompressed Chunk #2** | o o o

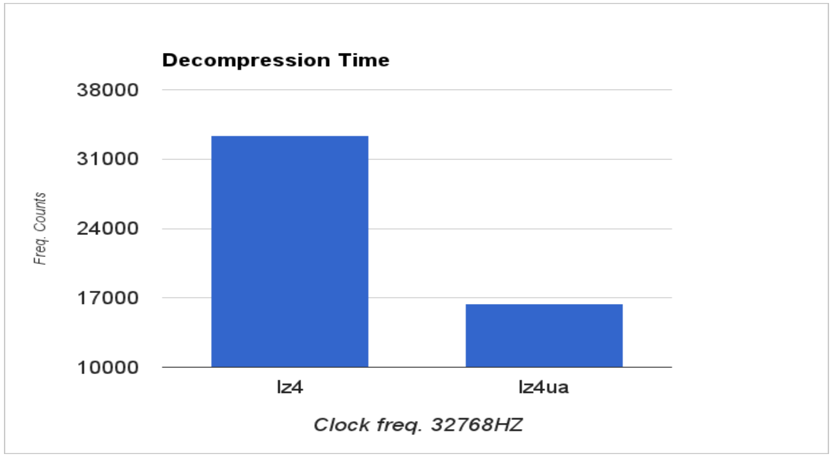
Improving Performance

Enable unaligned memory access:

- Supported by x86, powerpc and ARM(v6 and later, not default option in Kernel).

ARM: 7583/1: decompressor: Enable unaligned memory access for v6 and above by Dave Martin

- Decompression time(LZ4) is improved by 58% in our target board, ARMv7.



Measurement

Kernel Load and Decompression Time

- There is not a well-known method to measure Kernel decompression time in the early Kernel start-up

- Redundant Timer used, supported by HW platform and it is initialized in the bootloader and make it keep running

- Measure the function duration with the timer, eg. decompress() in decompress.c

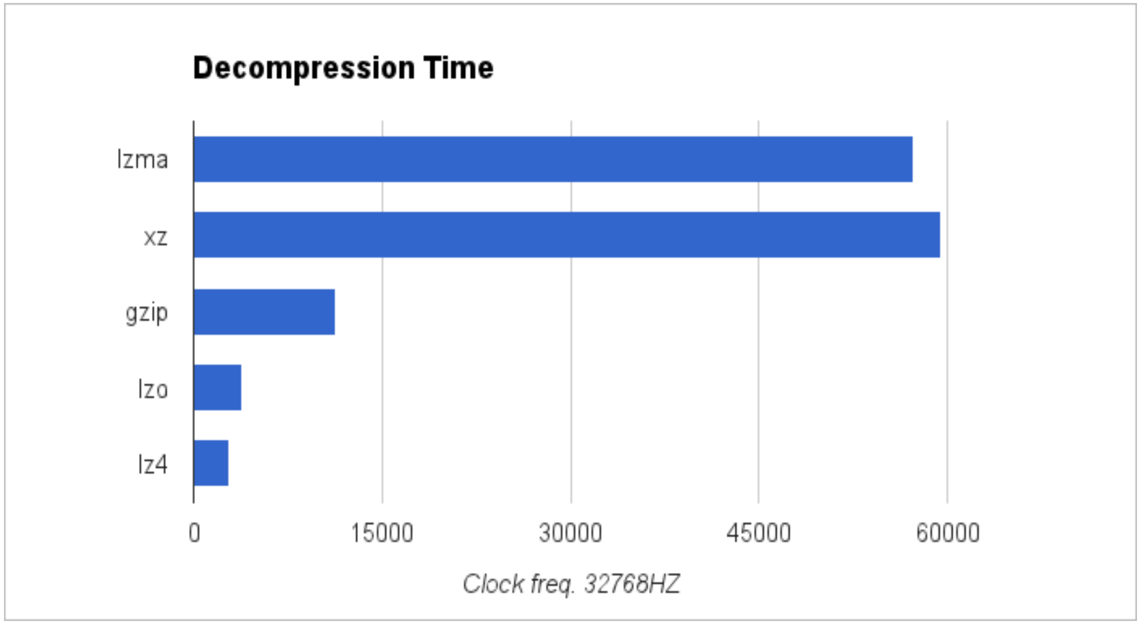
Case Study

- ARM v7 platform
- gcc 4.6.2
- Kernel 3.4
- Improved LZO patch applied
- Unaligned memory access enabled(LZ4 & LZO)

Decompression Time

Result

- Improved by 28% against lzo and 75% against gzip



Load vs. Decompression Time (1/2)

- Load Time (LT):
  Required Time to load compressed Kernel into memory

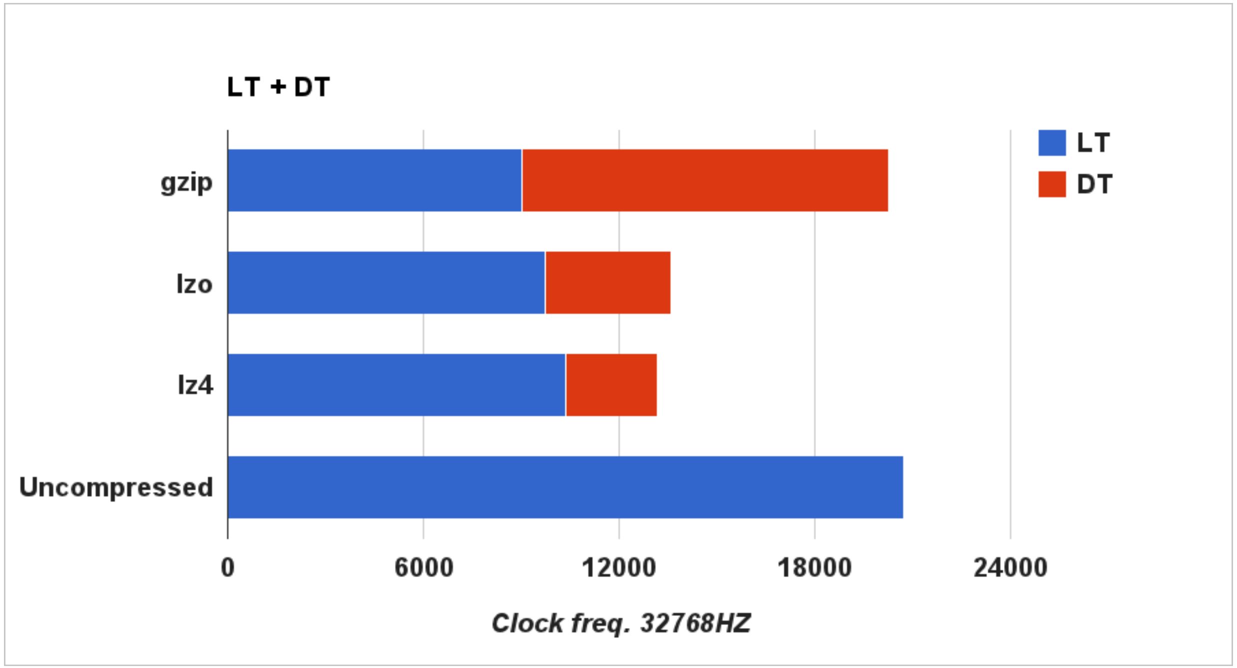
- Decompression Time (DT):
  Required Time to decompress compressed Kernel

- if Kernel is not compressed then DT is 0.

Load vs. Decompression Time (2/2)

Result

- Improved by 4% against lzo and 35% against gzip



Potential users of LZ4

- **Btrfs**
  one of main features available is: Compression(zlib and lzo)
- **squashFS**
  is a compressed read-only file system, zlib, xz and lzo compression is supported
- **zram**
  Increase performance by avoiding paging on disk and instead use a compressed block device in RAM, lzo is used for compression
- There will be more...

Challenges

Kernel Contribution Issues

- No distros are supporting LZ4 compression tool, lz4c which is required to compress Kernel.

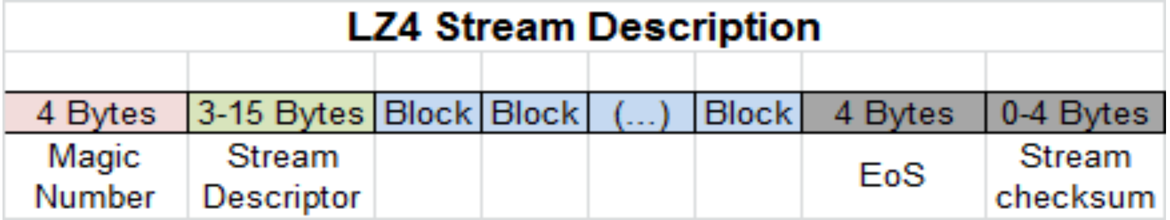
- There are many decompressors for Kernel: gzip(default), bzip2, lzma, xz, lzo and lz4(linux-next) more coming in the future

- Replace it if we have a replacement one for one of these below, suggested by Russell King
  one decompressor which is the fastest
  one decompressor for the highest compression ratio
  one popular decompressor(eg, gzip)
  (excerpts from http://thread.gmane.org/gmane.linux.kbuild.devel/9157)

Next Steps

- Supporting New LZ4 Streaming format specification

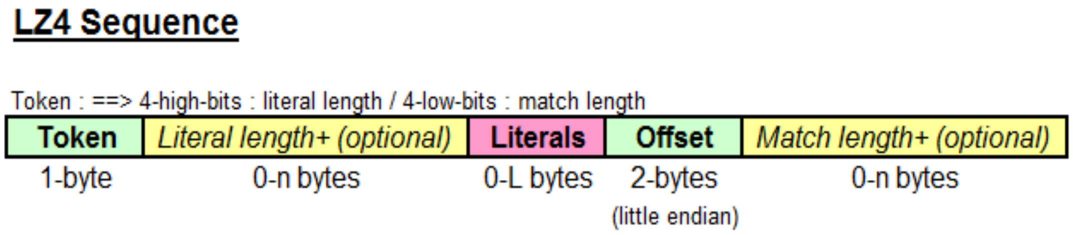
New



LZ4 Stream Description

4 Bytes	3-15 Bytes	Block	Block	(...)	Block	4 Bytes	0-4 Bytes
Magic Number	Stream Descriptor					EoS	Stream checksum

Legacy



LZ4 Sequence

Token : ==> 4-high-bits : literal length / 4-low-bits : match length

Token	*Literal length+ (optional)*	Literals	Offset	*Match length+ (optional)*
1-byte	0-n bytes	0-L bytes	2-bytes (little endian)	0-n bytes

References

1. https://code.google.com/p/lz4/
2. http://elinux.org/Boot_Time
3. http://fastcompression.blogspot.fr/2013/04/lz4-streaming-format-final.html
4. http://fastcompression.blogspot.kr/p/lz4.html
5. http://en.wikipedia.org/wiki/ZRam

Thank you!

LG

Life's Good