CHINA LINUX KERNEL
中国Linux内核开发者大会

華中科技大學
网络空间安全学院
School of Cyber Science and Engineering, HUST

# 第19届中国 Linux内核开发者大会

**赞助单位**

| | | | | |
|---|---|---|---|---|
| 華中科技大學 | HUAWEI | 阿里云 | FUJITSU | 腾讯云 |
| oppo | 火山引擎 | vivo | 龙芯中科 LOONGSON TECHNOLOGY | 蚂蚁集团 ANT GROUP |

**支持单位**

| | | | | |
|---|---|---|---|---|
| 清華大學 Tsinghua University | 迪捷软件 DIGIPROTO | OpenEuler | OpenAnolis 龙蜥社区 | OpenCloudOS |

**支持社区&媒体**

| | | | | |
|---|---|---|---|---|
| CSDN | 云巅论剑 | sf 思否 | 阅码场 yomocode | InfoQ 极客传媒 |
| 51CTO | 开源江湖 | | | |

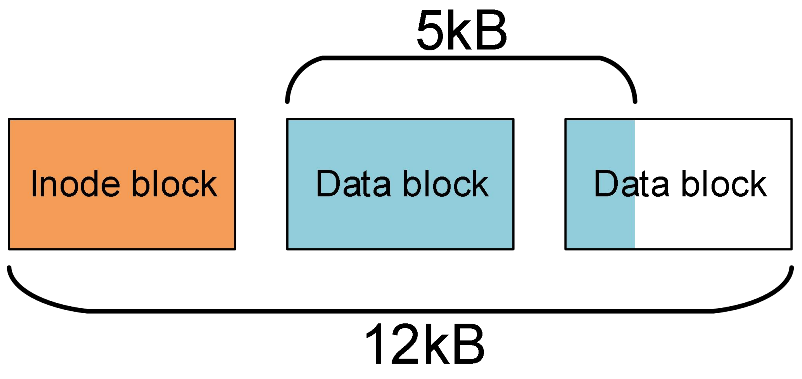
2024年10月　湖北 • 武汉

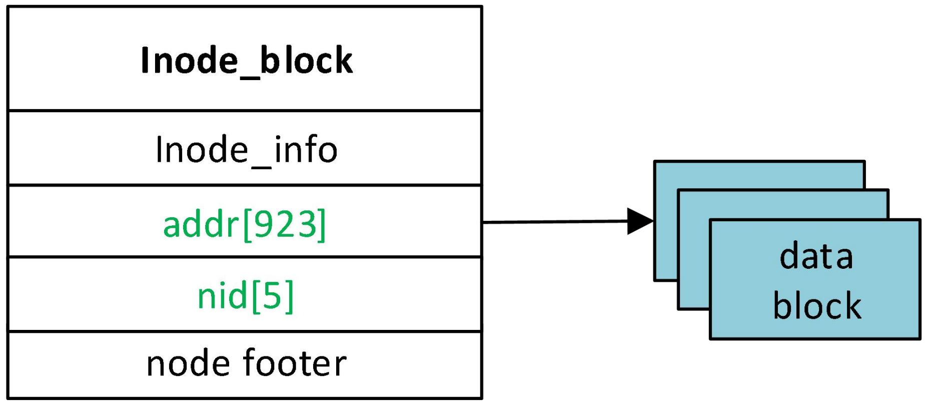
# Inline Tail：大幅减少 F2FS 小文件空间占用

伍勃（bo.wu@vivo.com）
vivo 存储系统工程师

## F2FS Inode is Too Large

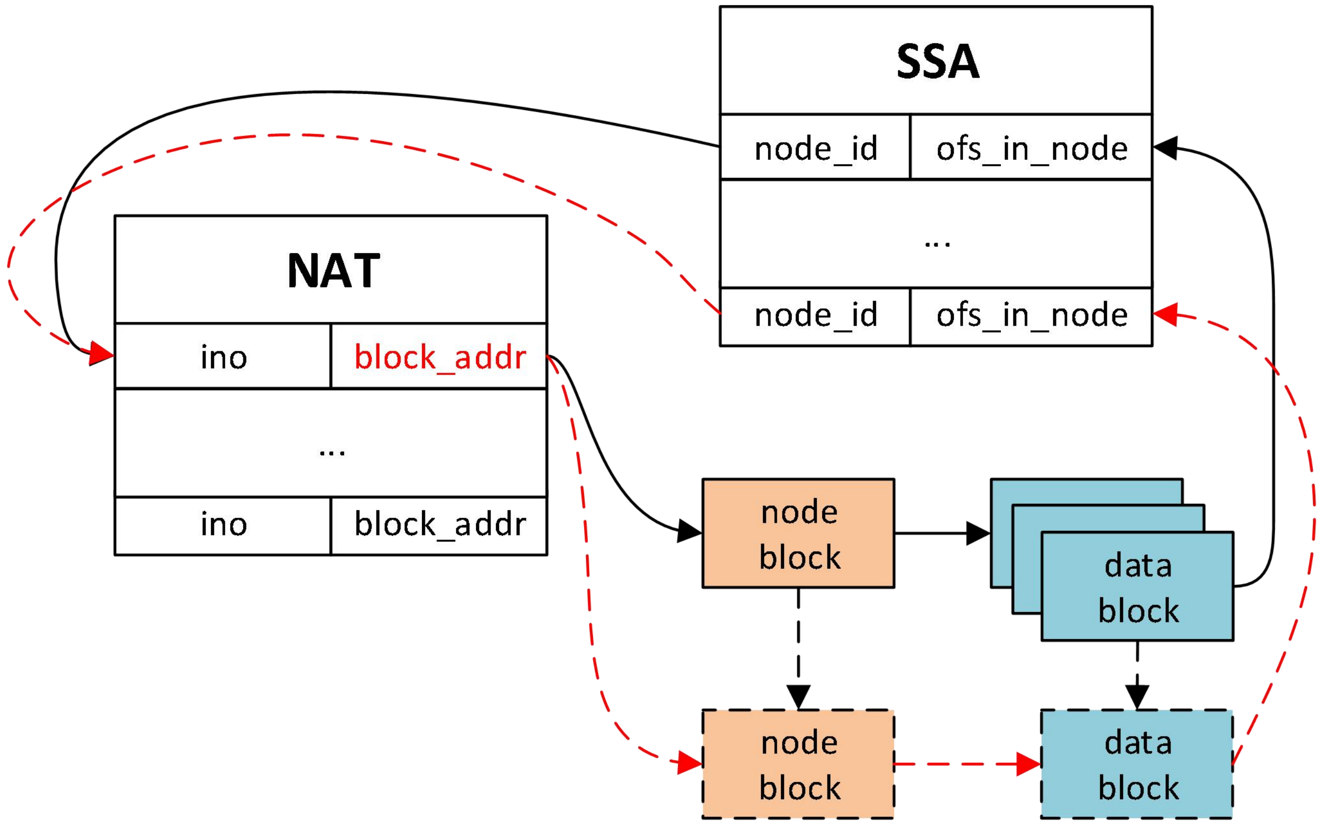
- F2FS Inode occupies an entire 4kB block
- EXT4 : 256B, XFS : 512B
- File storage space = inode block + data block
- **Small files waste too much space**





## Why is the F2FS Inode So Large

- F2FS defines two block types (node and data)
- Using NAT to avoid wandering tree problem
- **1 inode use 1 node block**
- Reducing inode size requires restructuring all metadata



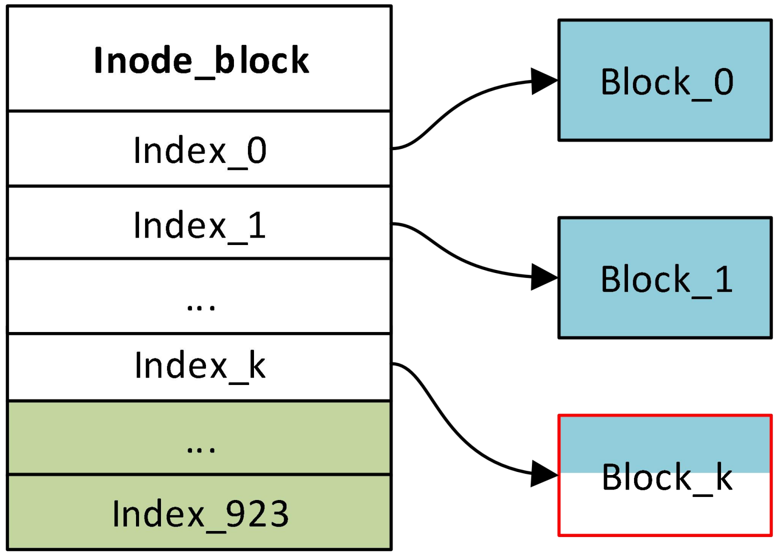
# F2FS Inode Space Utilization Optimization

- Xattr inline
    - Save xattr in inode
    - Default 200B
- Data inline
    - Save small file in inode
    - Size < 3.4kB
- **Small file size > 4kB?**

| **inode block** | **4096** | **xattr inline** | | **data inline** | |
|---|---|---|---|---|---|
| inode info | 360 | | | | |
| addr table[923] | 3692 | extra info | 0~36 (i_extra_isize) | | |
| | | **addr[n]** | **3456~3492** | reserved addr | 4 |
| | | | | data inline | 3452~3488 |
| | | inline xattr[50] | 200 | | |
| nid table[5] | 20 | | | | |
| node footer | 24 | | | | |

## Principle of F2FS Inline Tail

- The tail block of a file is often not fully filled
- Address table of small files is essentially empty
- **Inline tail: store tail block into inode block**
- Can save one block(4kB)



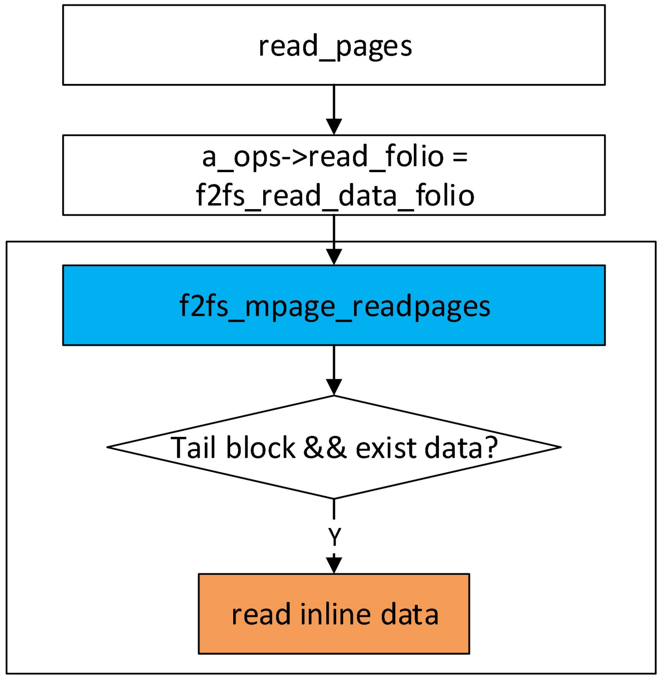
## Design of F2FS Inline Tail

- **Size limitation: <68kB**
- Small files use a compact block address array with 16 entries
- Mixed blocks: tail block stored inline, others use traditional block
- Why 16 entries?: The marginal benefit decreases as the size increases

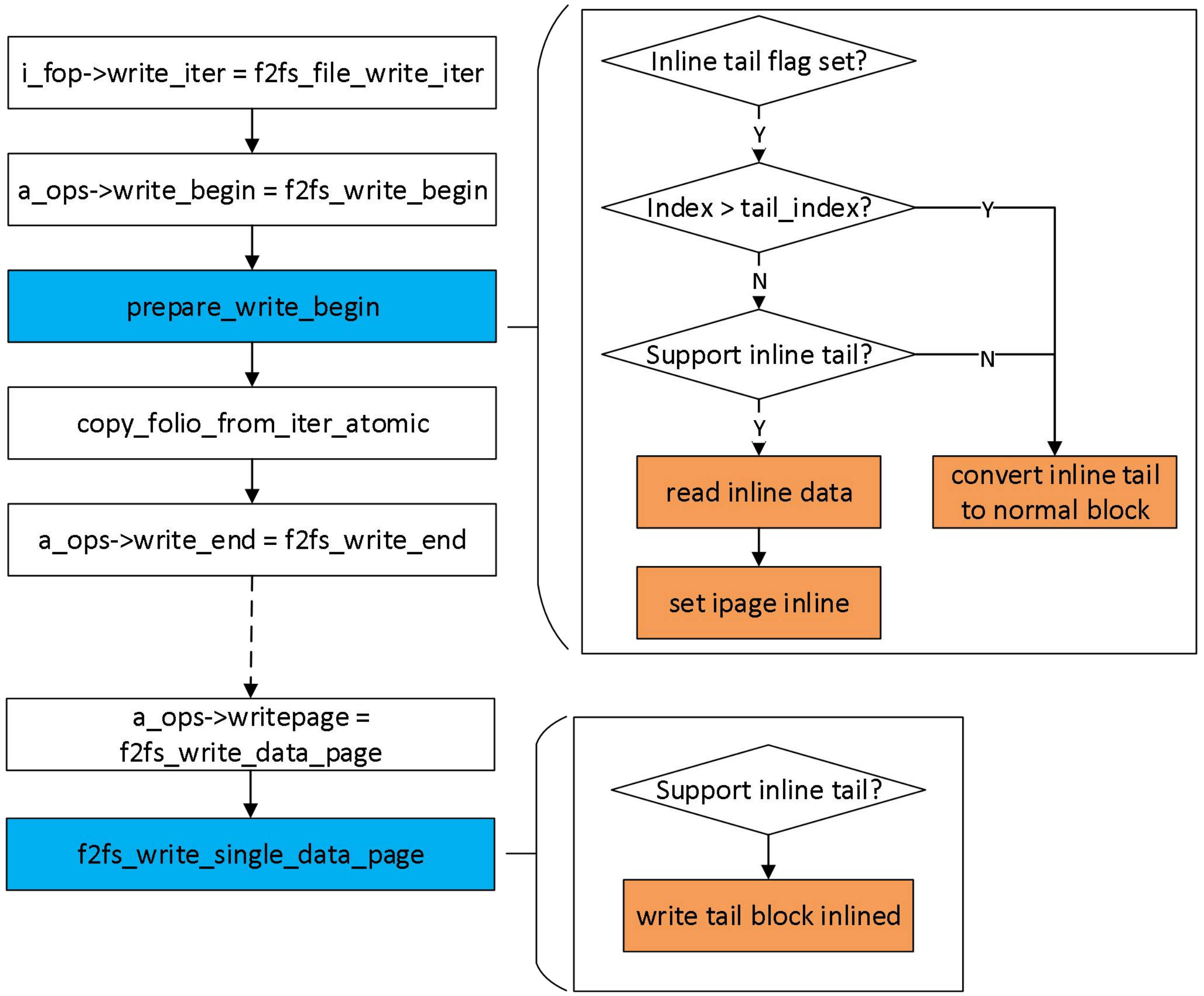
| **inode block** | **4096** | **tail inline** | |
|---|---|---|---|
| inode info | 360 | | |
| addr table[923] | 3692 | extra info | 0~36 (i_extra_isize) |
| | | **addr[16]** | **64** |
| | | reserved addr | 4 |
| | | **inline tail** | **3392~3428** |
| | | inline xattr[50] | 200 |
| nid table[5] | 20 | | |
| node footer | 24 | | |

# Implementation of F2FS Inline Tail

- Only support buffer read & write
- Adapt inline tail for other FS interfaces



read_pages
a_ops->read_folio = f2fs_read_data_folio
f2fs_mpage_readpages
Tail block && exist data?
Y
read inline data



i_fop->write_iter = f2fs_file_write_iter
a_ops->write_begin = f2fs_write_begin
prepare_write_begin
copy_folio_from_iter_atomic
a_ops->write_end = f2fs_write_end
a_ops->writepage = f2fs_write_data_page
f2fs_write_single_data_page
Inline tail flag set?
Y
Index > tail_index?
Y
N
Support inline tail?
N
Y
read inline data
set ipage inline
convert inline tail to normal block
Support inline tail?
write tail block inlined

## Development of F2FS Inline Tail

- Data race issue:
  - b3d208f96d6b ("f2fs: revisit inline_data to avoid data races and potential bugs")
  - Only support normal buffer write
  - **Inline conversion is permanently, not bidirectional**
  - Take lock when changing inline flag
  - Take lock when read file size

- Regression test:
  - Tool: xfstest-bld ([https://github.com/tytso/xfstests-bld](https://github.com/tytso/xfstests-bld))
  - The test results are not very stable (random operations)

## Benefits of F2FS Inline Tail

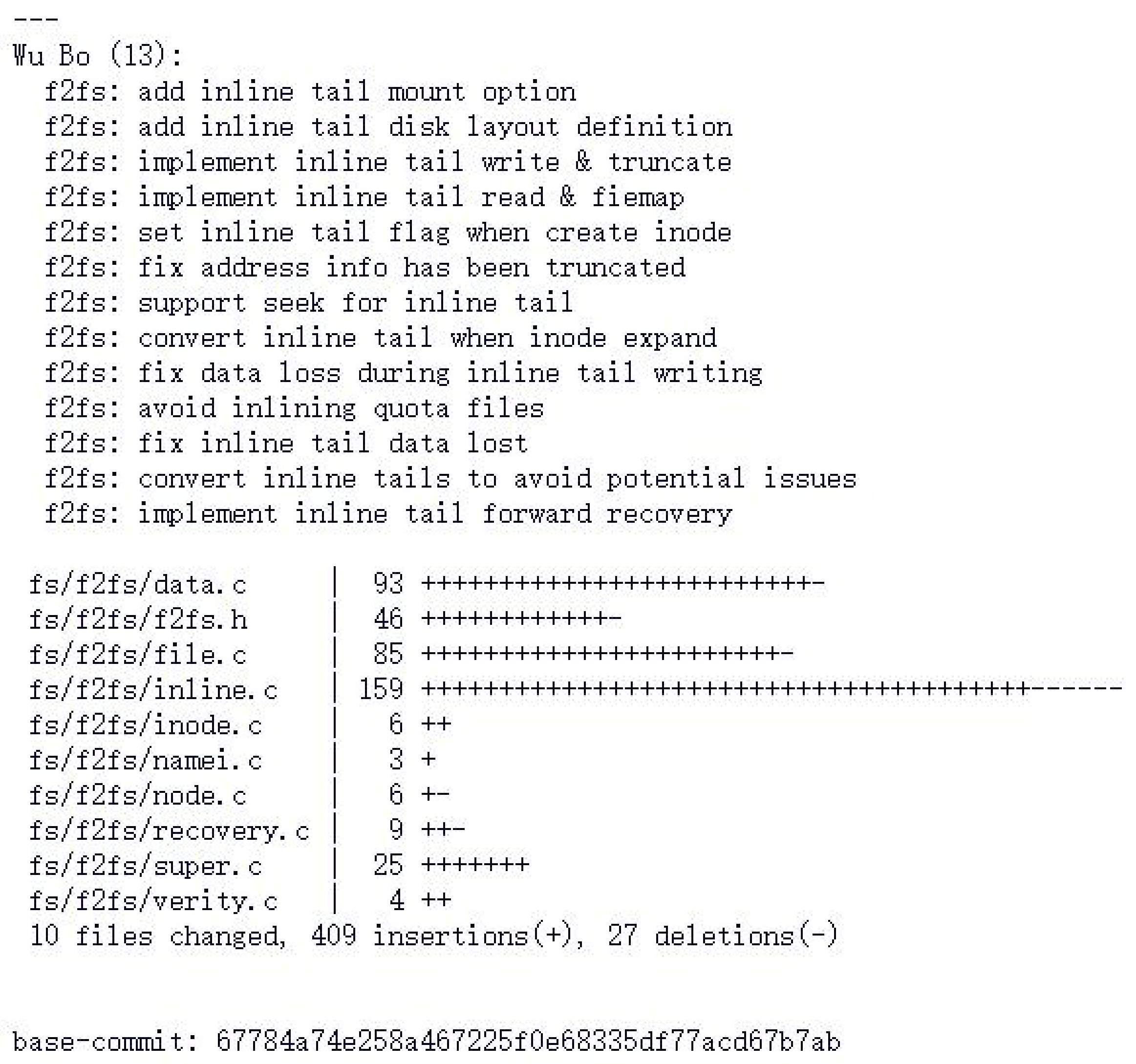
- Little files (4kB~68kB) can save one block(4kB), reduce most 1/3 space & IO request.
- Linux source code storage space -8%, copy time -10%
- Double the benefit from inline data

| Inline Tail Saved Storage Space | | | |
|---|---|---|---|
| **file size** | **w/o** | **/w** | **saved** |
| <3.4k | 4 | 4 | 0% |
| 4k~7.4k | 12 | 8 | **33%** |
| 8k~11.4k | 16 | 12 | 25% |
| 12k~15.4k | 20 | 16 | 20% |
| 16k~19.4k | 24 | 20 | 17% |
| 28k~31.4k | 36 | 32 | 11% |
| 44k~47.4k | 52 | 48 | 8% |
| 64k~67.4k | 72 | 68 | 6% |

## Future Development & Outlook

- Replace inline data
  - **Inline data is a special case of inline tail**
- Measure the benefit of reducing IO request
  - 85% of files on user phones are smaller than 64kB
  - Reducing I/O should improve system performance
- Using inline tail as a file write buffer
  - Cache small synchronous data in the inline tail
- Support encryption on inline tail

# Patches for F2FS Inline Tail



```
---
Wu Bo (13):
  f2fs: add inline tail mount option
  f2fs: add inline tail disk layout definition
  f2fs: implement inline tail write & truncate
  f2fs: implement inline tail read & fiemap
  f2fs: set inline tail flag when create inode
  f2fs: fix address info has been truncated
  f2fs: support seek for inline tail
  f2fs: convert inline tail when inode expand
  f2fs: fix data loss during inline tail writing
  f2fs: avoid inlining quota files
  f2fs: fix inline tail data lost
  f2fs: convert inline tails to avoid potential issues
  f2fs: implement inline tail forward recovery

 fs/f2fs/data.c     |  93 +++++++++++++++++++++++++-
 fs/f2fs/f2fs.h     |  46 ++++++++++++-
 fs/f2fs/file.c     |  85 +++++++++++++++++++++++-
 fs/f2fs/inline.c   | 159 +++++++++++++++++++++++++++++++++++++++------
 fs/f2fs/inode.c    |   6 ++
 fs/f2fs/namei.c    |   3 +
 fs/f2fs/node.c     |   6 +-
 fs/f2fs/recovery.c |   9 ++-
 fs/f2fs/super.c    |  25 +++++++
 fs/f2fs/verity.c   |   4 ++
 10 files changed, 409 insertions(+), 27 deletions(-)


base-commit: 67784a74e258a467225f0e68335df77acd67b7ab
```

https://lore.kernel.org/linux-f2fs-devel/cover.1726024116.git.bo.wu@vivo.com/

# THANK YOU