

SELinuxの開発と、 コミュニティへの飛び込み方

(Oct 28/Open Source Conference 2006 Tokyo/Fall)

NEC OSS推進センタ

海外 浩平

<kaigai@ak.jp.nec.com>

目 次

- ・ **イントロダクション**
- ・ XATTR on JFFS2
- ・ コミュニティからのフィードバック
- ・ コミュニティに飛び込んでみよう
- ・ 付録

そもそも何をやっている人ですか？

- SELinux周りの開発をしています。
 - SELinuxのスケラビリティ改善 (2004)
 - ✓ 昔のSELinuxはパフォーマンスが出なかった
 - ✓ RCUを使ってロックレス参照を可能にした
 - **JFFS2のXATTR対応** (2005-2006)
 - ✓ JFFS2 = Flash-ROM用のファイルシステム
 - ✓ SELinuxを使うにはXATTRの対応が必須
 - SELinux対応PostgreSQLの開発 (進行中)
 - ✓ OSと一体化した情報フロー制御

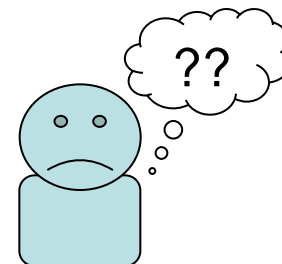
JFFS2のXATTR対応 ～開発のスタート(1)～

・ 組み込み分野とSELinuxの親和性

- セキュリティパッチのリリースがあっても
 - ⇒ 即座にパッチ適用ができないケースもある
- 同一構成のマシンが大量に出回る
 - ⇒ 0-dayアタックの脅威が予測不能
- 生命・財産に対する脅威に直結する

・ では、何が必要になるのか？

- busyboxのSELinux対応コマンド
- セキュリティポリシーのサイズ削減
- ディスクレスファイルシステムでのXATTR対応



JFFS2のXATTR対応 ～開発のスタート(2)～

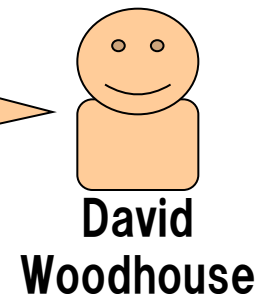
- **本家コミュニティの開発動向は？**
 - 実際に聞いてみた。

Jul, 2005@Ottawa Linux Symposiumにて

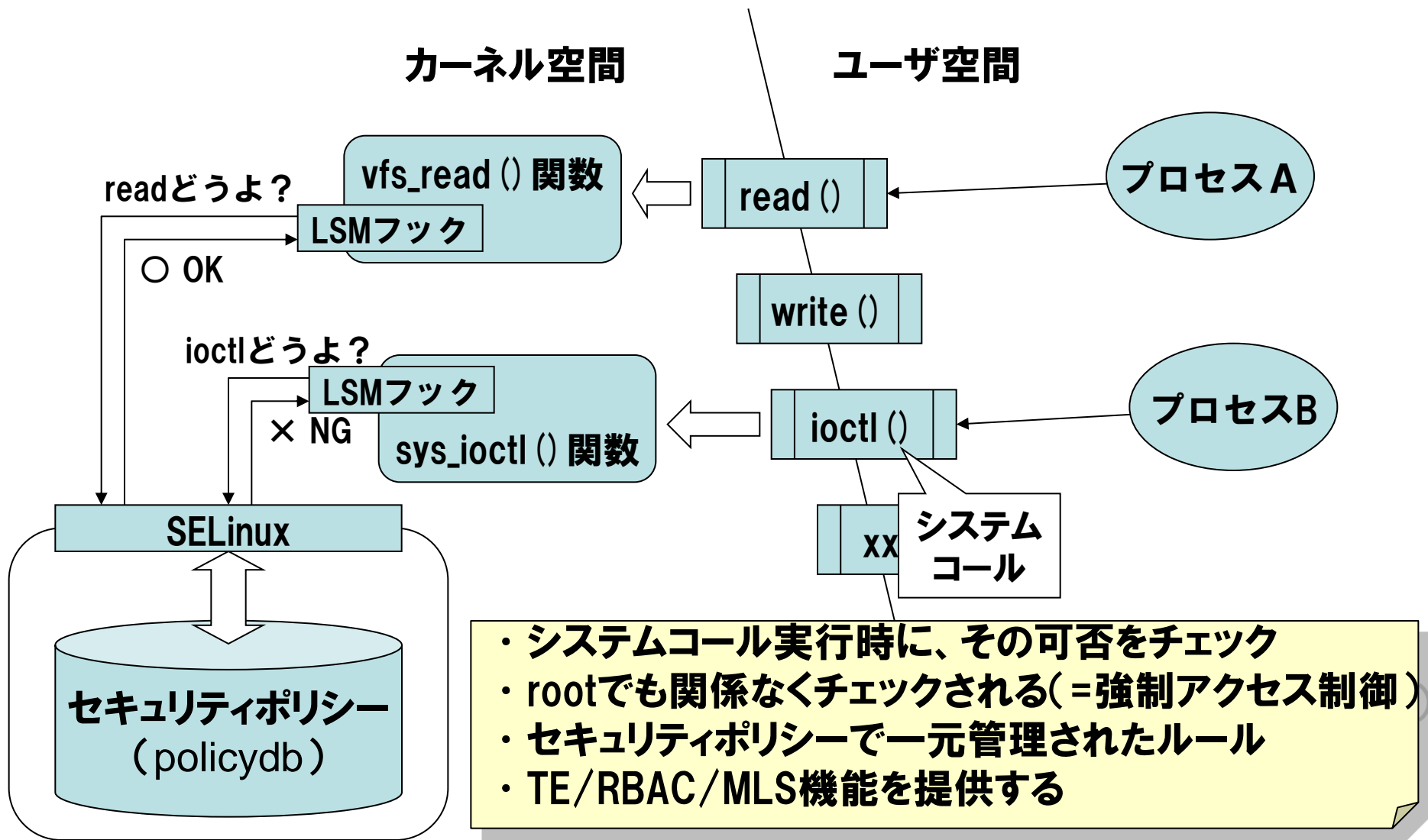


1. busyboxはRussell Cockerのパッチがあったはず
2. ポリシーのサイズ削減は、丁度開発してるところ
3. JFFS2のXATTRは、隠しファイルを作って実装した人がいたみたいだけでも...

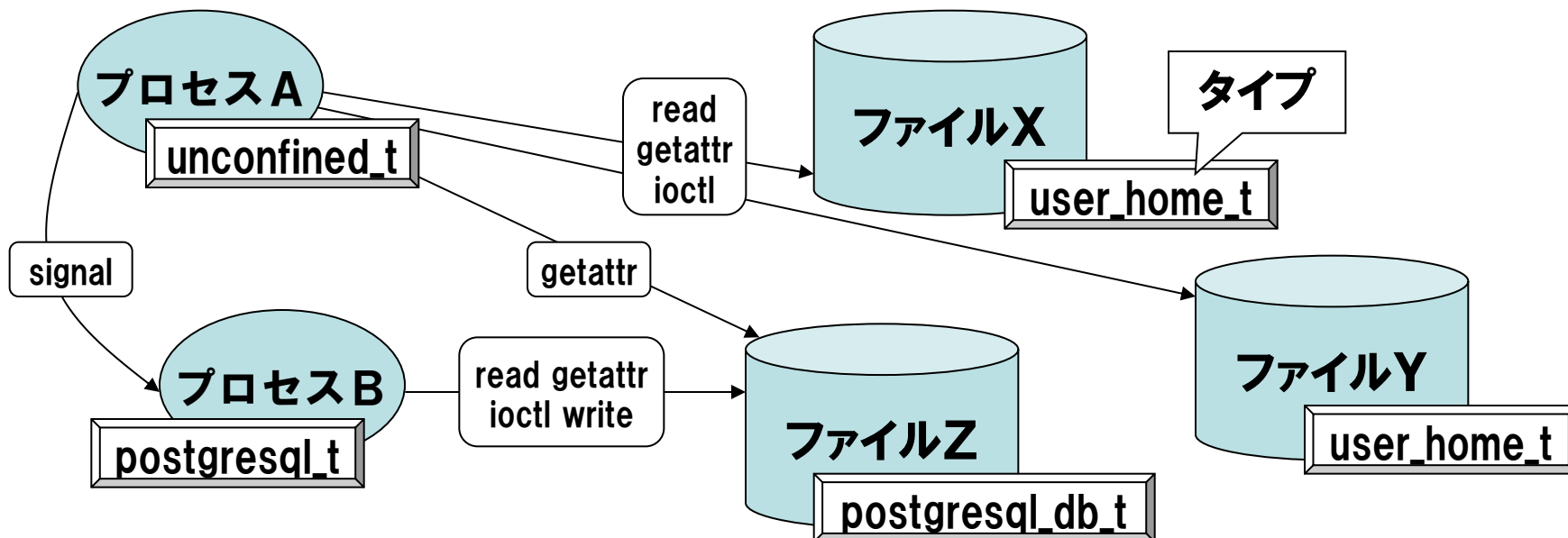
JFFS2のXATTR実装は今のところ計画していないけれども、コントリビューションは大歓迎。



その前に、、SELinuxって何ですか？（１）



その前に、、SELinuxって何ですか？（２）



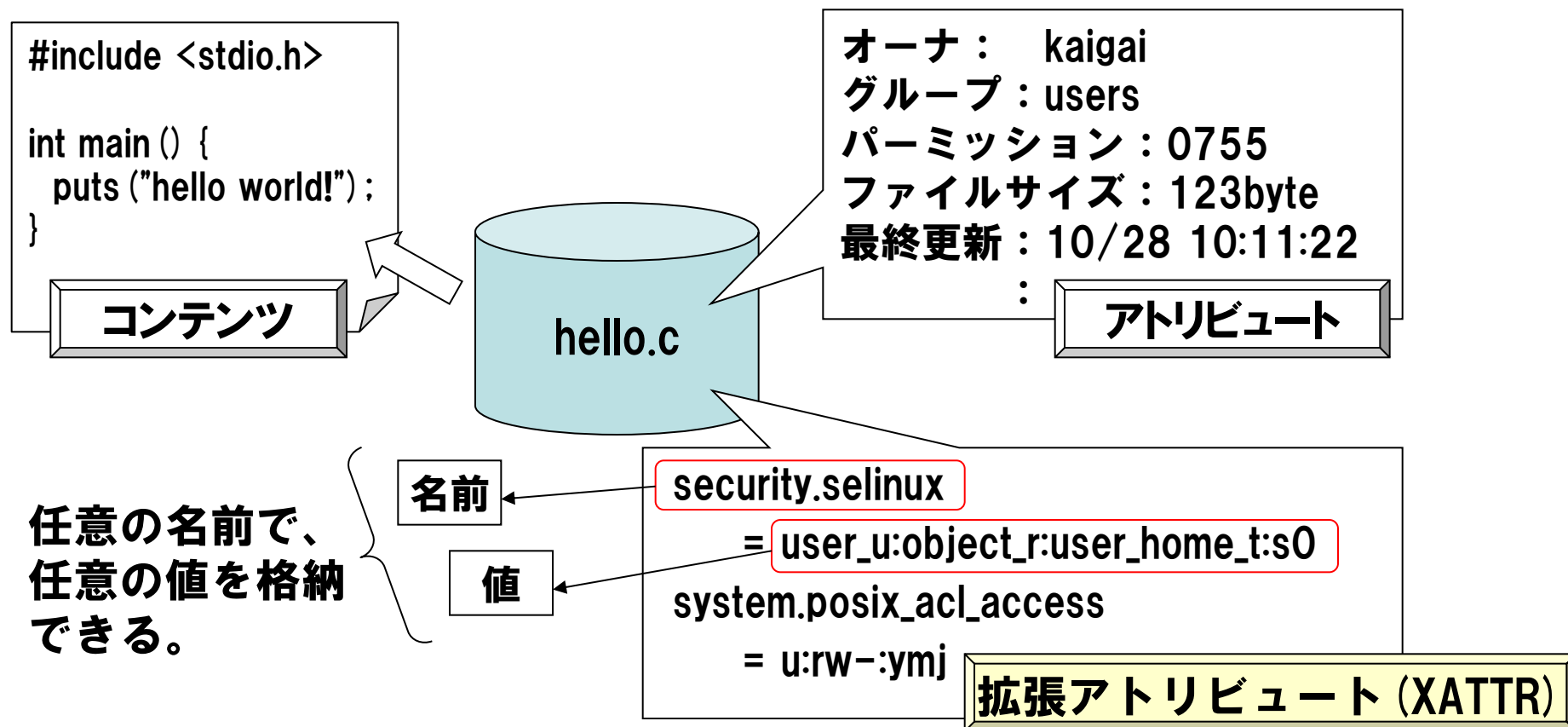
■ セキュリティポリシー ■

```
allow unconfined_t user_home_t:file {read getattr ioctl};  
allow unconfined_t postgresql_db_t:file {getattr};  
allow postgresql_t postgresql_db_t:  
    file {read getattr ioctl write};  
allow unconfined_t postgresql_t:process {signal};
```


全てのプロセス・リソースに
タイプを付与する。
→ "型"の強制、即ち
TE (Type Enforcement)

その前に、XATTRって何ですか？

- XATTR=eXtended ATTRibute (拡張アトリビュート)



XATTRの対応状況 (2.6.18カーネル)

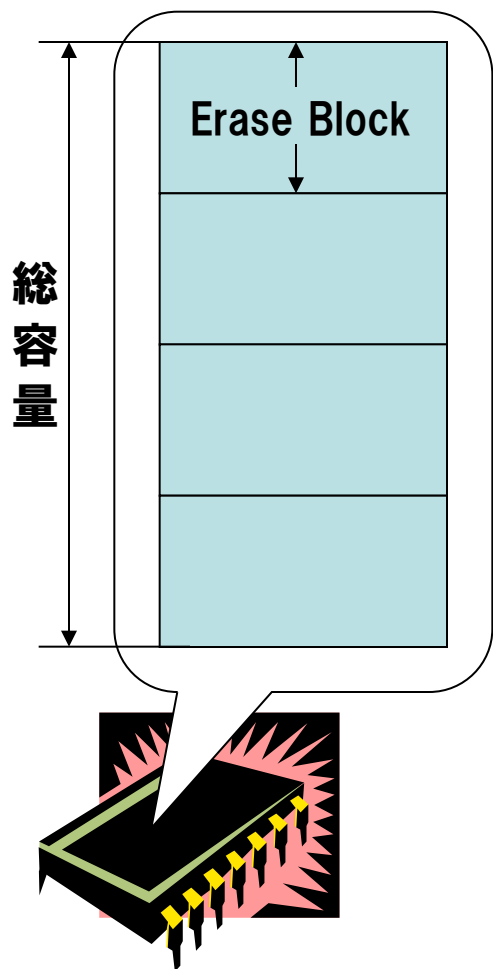
- ・ **サーバ/WS向け**
 - Ext2, Ext3, XFS, ReiserFS, JFS
- ・ **組み込み用途向け**
 - JFFS2 

2.6.18カーネルで追加されました。
 - ・ Flash-ROM等を有効利用するために開発されたFS
 - ・ ロバストネスとメディアの寿命を重視した設計
- ・ **ネットワークファイルシステム**
 - NFSv3, NFSv4, CIFS (但し、POSIX-ACLのみの対応)
- ・ **擬似ファイルシステム**
 - FUSE

目 次

- ・ イントロダクション
- ・ **XATTR on JFFS2**
- ・ コミュニティからのフィードバック
- ・ コミュニティに飛び込んでみよう
- ・ 付録

JFFS2ってどんなファイルシステム？（１）



・ Flash-ROMの特性

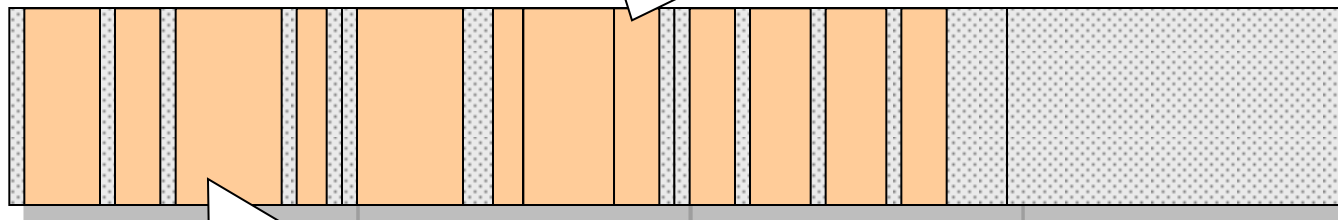
- ✓ MTDデバイスとして認識される
- ✓ 書き込み回数に制限。およそ10万回程度
- ✓ 一度書き込まれたデータは、Erase Block単位でしか消去できない。

・ JFFS2の特徴

- ✓ MTDデバイスの上位レイヤとして実装
- ✓ メディア上の特定箇所を酷使しない工夫（追記型アーキテクチャ）
- ✓ ガベージコレクタがメディア上のデータを再配置し、Erase Block単位の消去を可能に。

JFFS2ってどんなファイルシステム？（2）

追記型アーキテクチャ

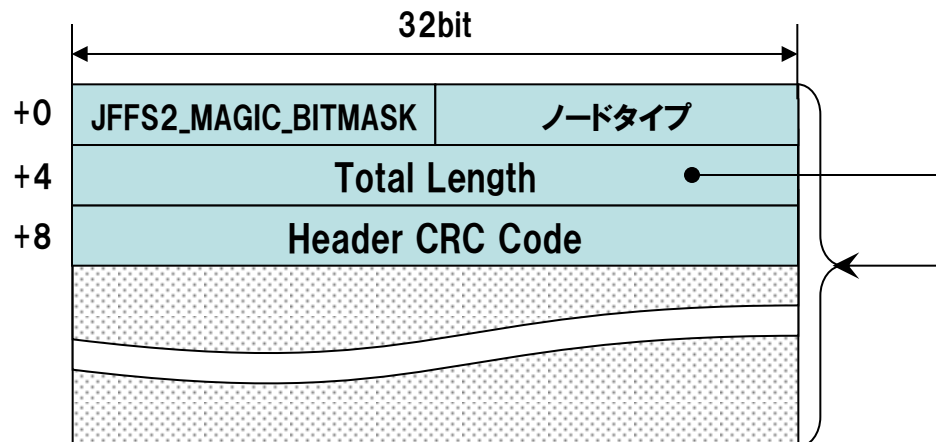


書き込み単位：ノード

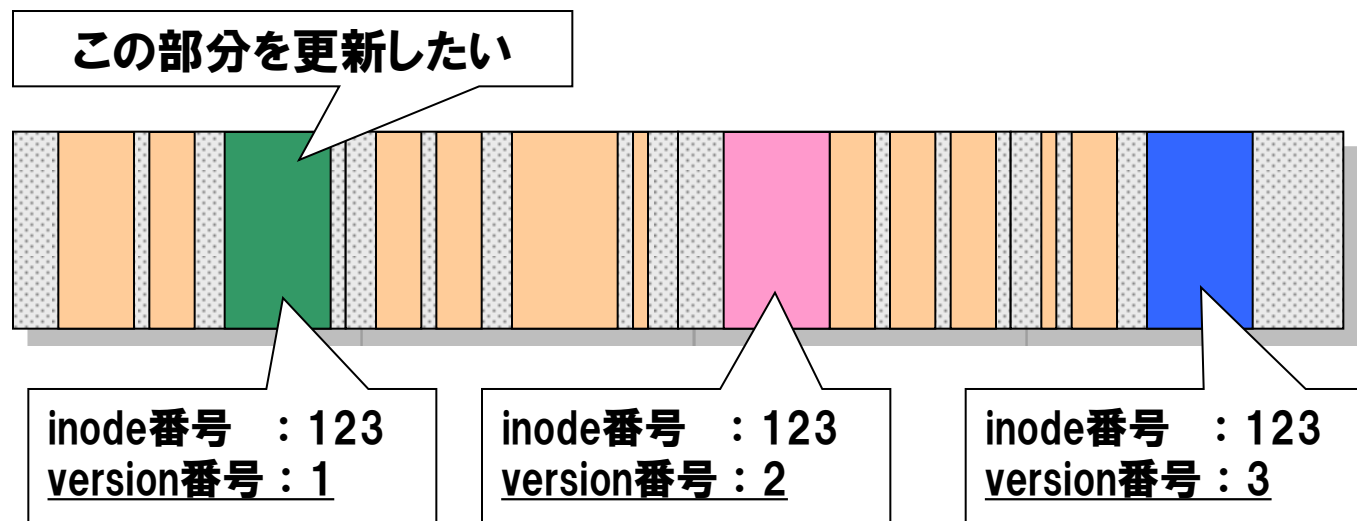
- ・ 任意の長さで記録できる
- ・ CRCコードで整合性を確認
- ・ inode型など7種類

JFFS2_NODETYPE_INODE
JFFS2_NODETYPE_DIRENT
JFFS2_NODETYPE_CLEANMARKER
JFFS2_NODETYPE_PADDING
JFFS2_NODETYPE_SUMMARY
JFFS2_NODETYPE_XATTR
JFFS2_NODETYPE_XREF

ノード共通ヘッダ

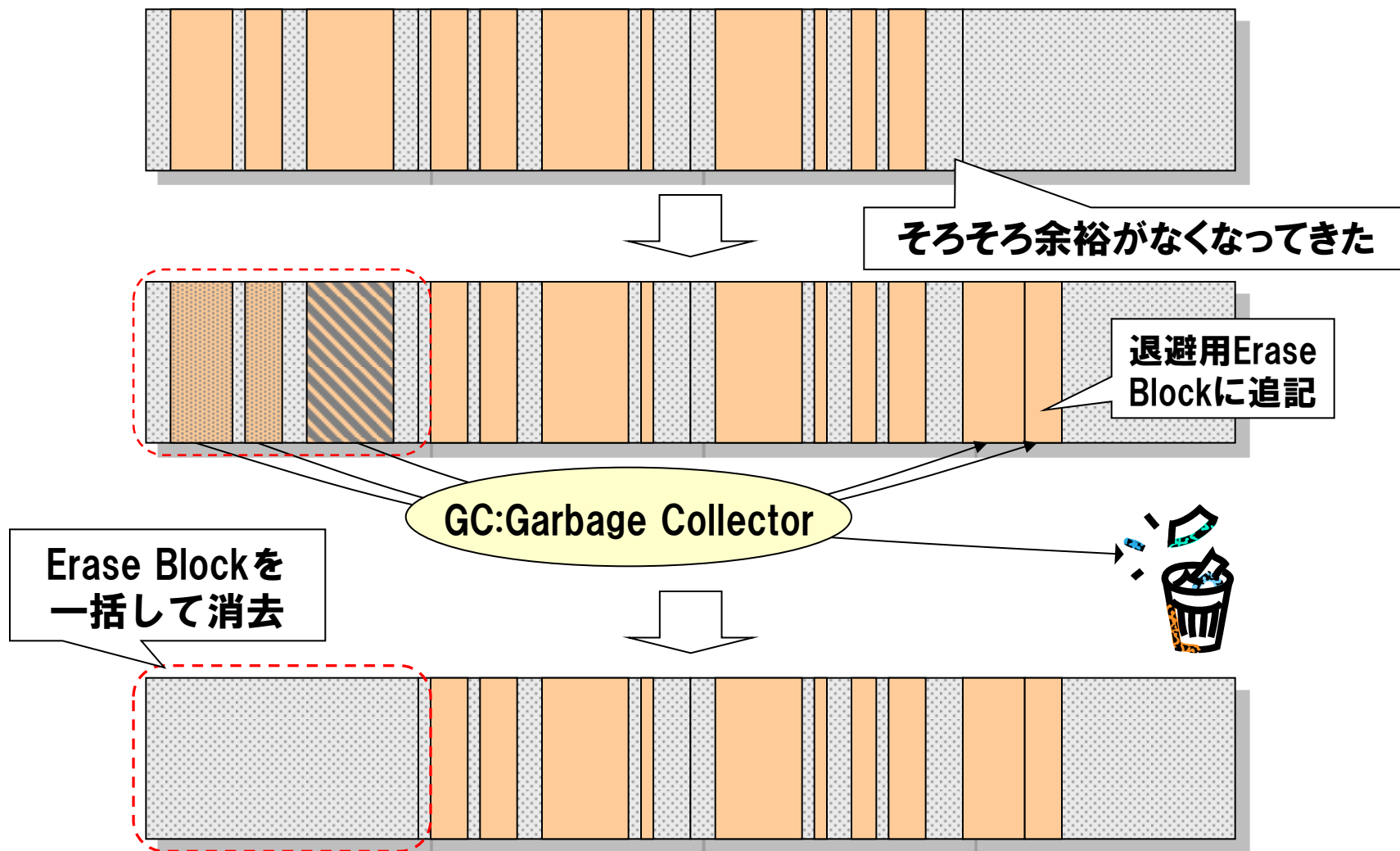


JFFS2ってどんなファイルシステム？（3）

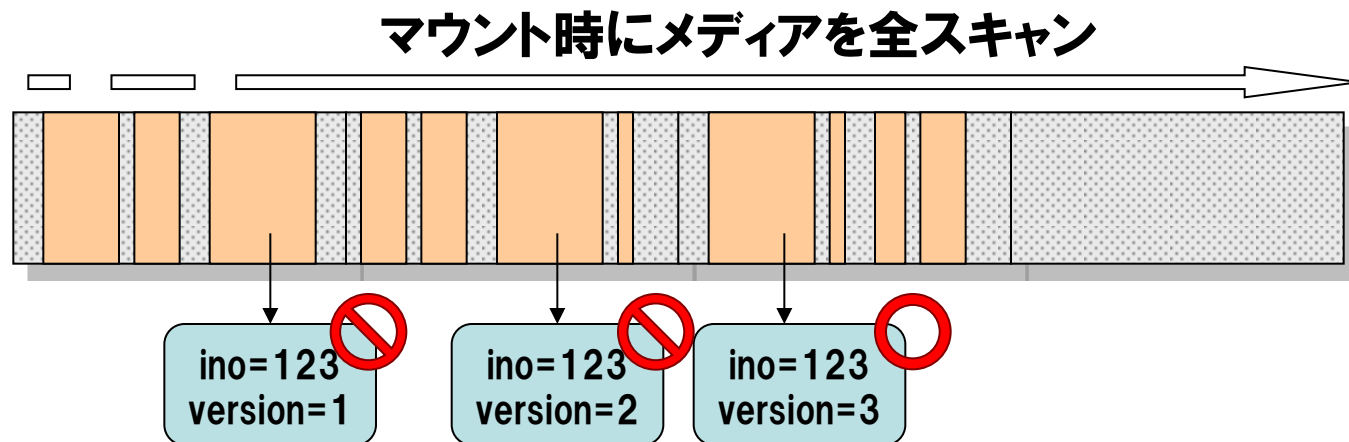


- ・ 更新 = version番号のインクリメント + 追記
 - ・ inode番号が同じで、version番号の異なる複数のノードが存在
 - ・ 古いversion番号のノードは無視される
- やがてGCによって回収され、Erase Blockの消去後に領域は利用可能となる。

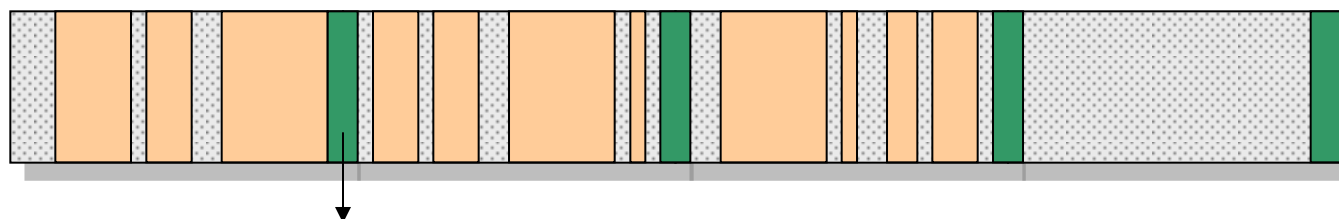
JFFS2ってどんなファイルシステム？（４）



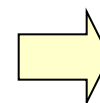
JFFS2ってどんなファイルシステム？ (5)



EBS (Erase Block Summary) 機能



Erase Block末尾のSummaryだけ読めば、
どの種類のノードが、どこに書かれているか
情報を得ることができる。



マウント時間短縮

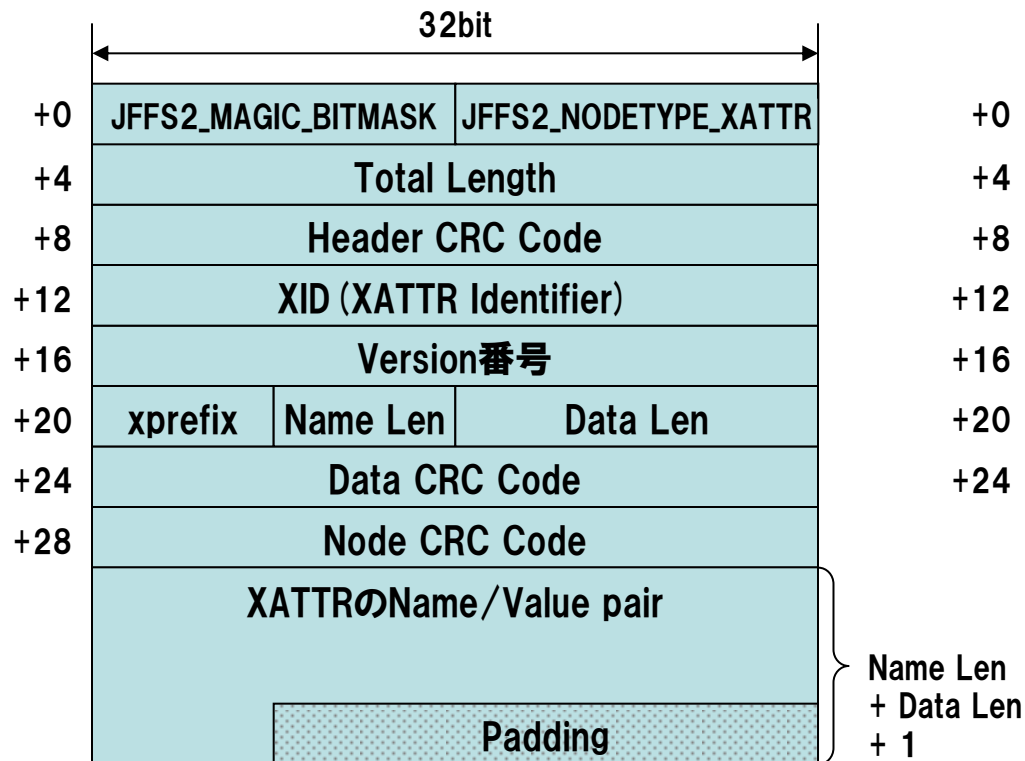
JFFS2ってどんなファイルシステム？（6）

- ・ **追記型アーキテクチャ**
 - － メディア上の特定箇所を酷使しない。→ 長保ち！
 - － Flash-ROMの特性に合わせ、GCがノードを再配置
- ・ **ロバスト性**
 - － 特定位置にSuper Blockを持たない
 - － データの読み込み時にCRCチェックを実行する
- ・ **欠点も、、、**
 - － FSのマウント時には、メディアの全スキャンが必要
 - ・ EBS (Eraseblock Summary Support) でマシになった

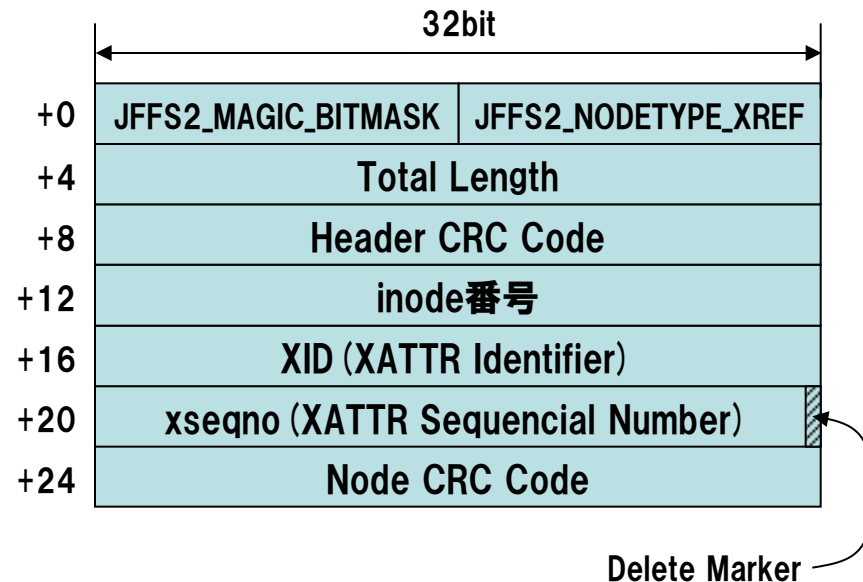
XATTR on JFFS2 の基本構造 (1)

・ XATTRを格納するノード型の定義

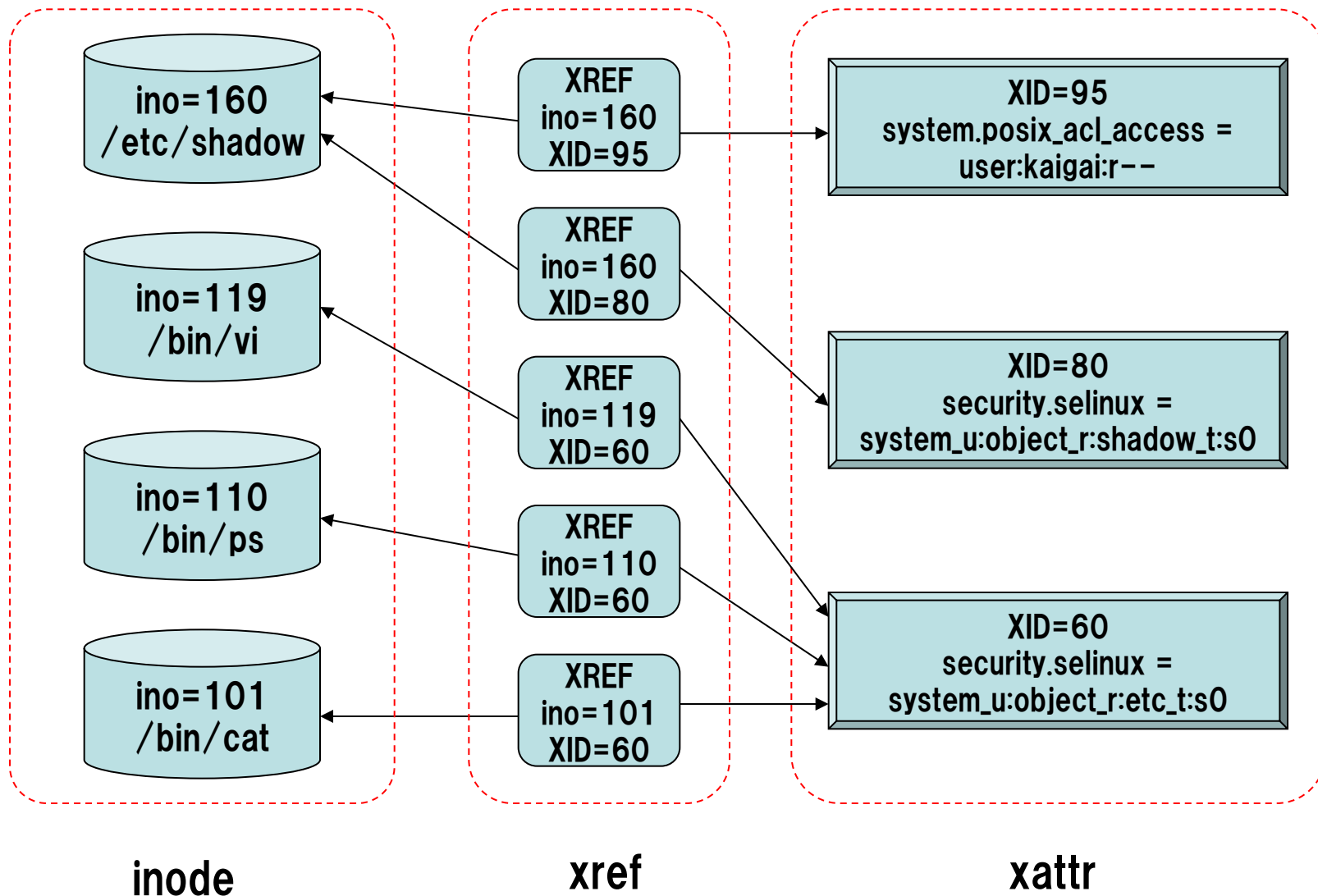
JFFS2_NODETYPE_XATTRノード



JFFS2_NODETYPE_XREFノード



XATTR on JFFS2 の基本構造 (2)



XATTR on JFFS2 の基本構造 (3)

- inodeとxattrがN:M関係を持つ
 - xattrのname/valueペアを複数のinodeが共有する
 - ストレージ/メモリ利用効率で優れる
- 既存のノード構造には手を加えない
 - 古いカーネルでは、単に「未知のノード」として無視される
- 格納することができるのは
 - SELinuxラベル (security.*)
 - POSIX ACL (system.posix_acl_ {access|default})
 - 任意のユーザ型データ (user.*, trusted.*)

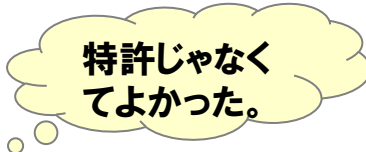
過去に提案された方法

・ 隠しファイルを利用した方法

- Lorenzo Hernández García-Hierro氏
- ReiserFSで利用されている方法と同じ
 - ・ JFFS3のロードマップに載っているから & バグが取れなくて困っているという理由で投げ出してしまった....。

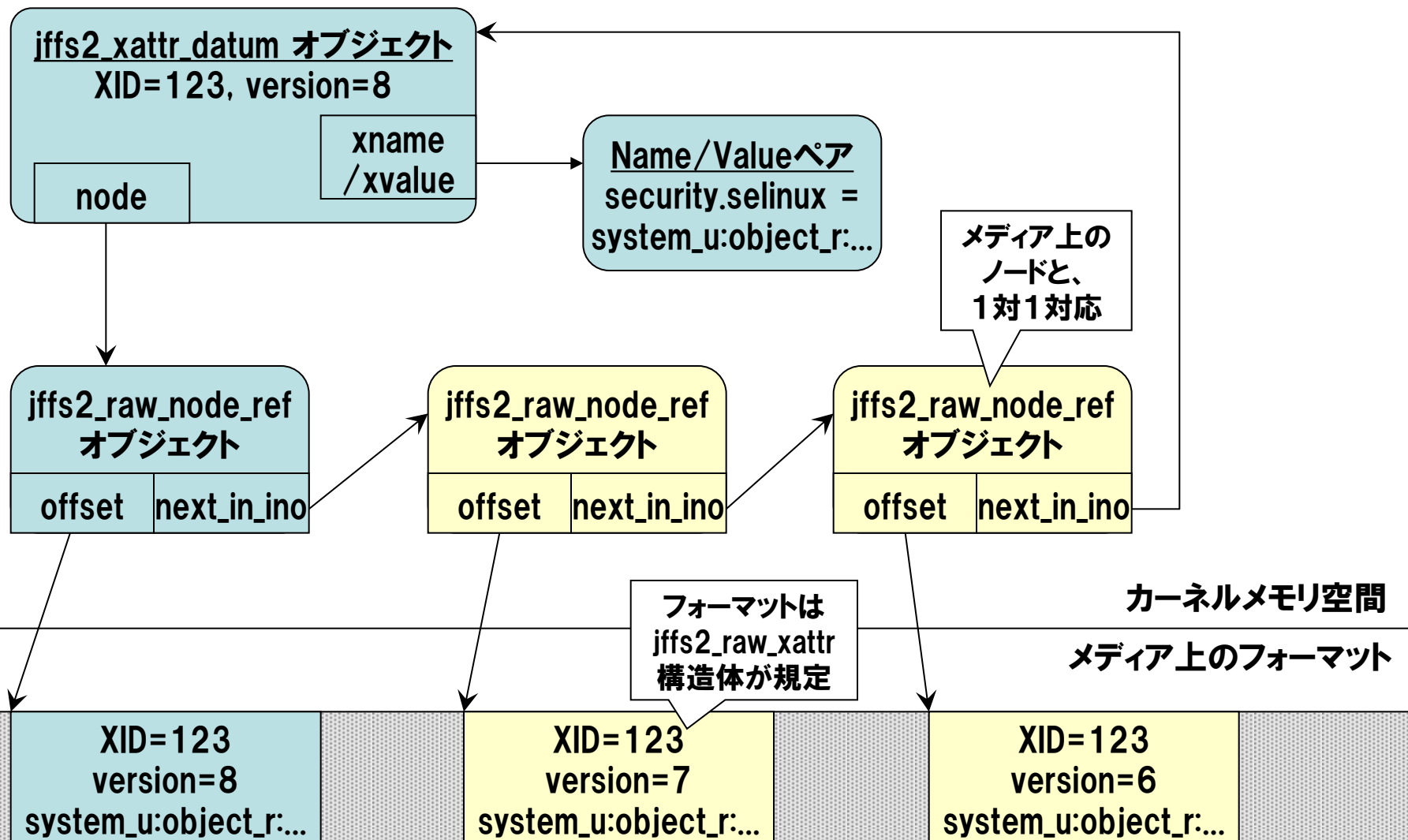
・ ディレクトリエントリの拡張

- Ma Yun氏
- 私がパッチを投稿する6時間前 (!) にポストされた。
- ディレクトリエントリの中にXATTRのName/Valueペアを入れる。
 - ・ 同一のName/Valueペアを共有できない。
 - ・ SELinuxでは、非常に多くの同一Name/Valueペアが使われる。
 - ・ JFFS2_NODETYPE_DIRENTノードの意味を変えるのは望ましくない。

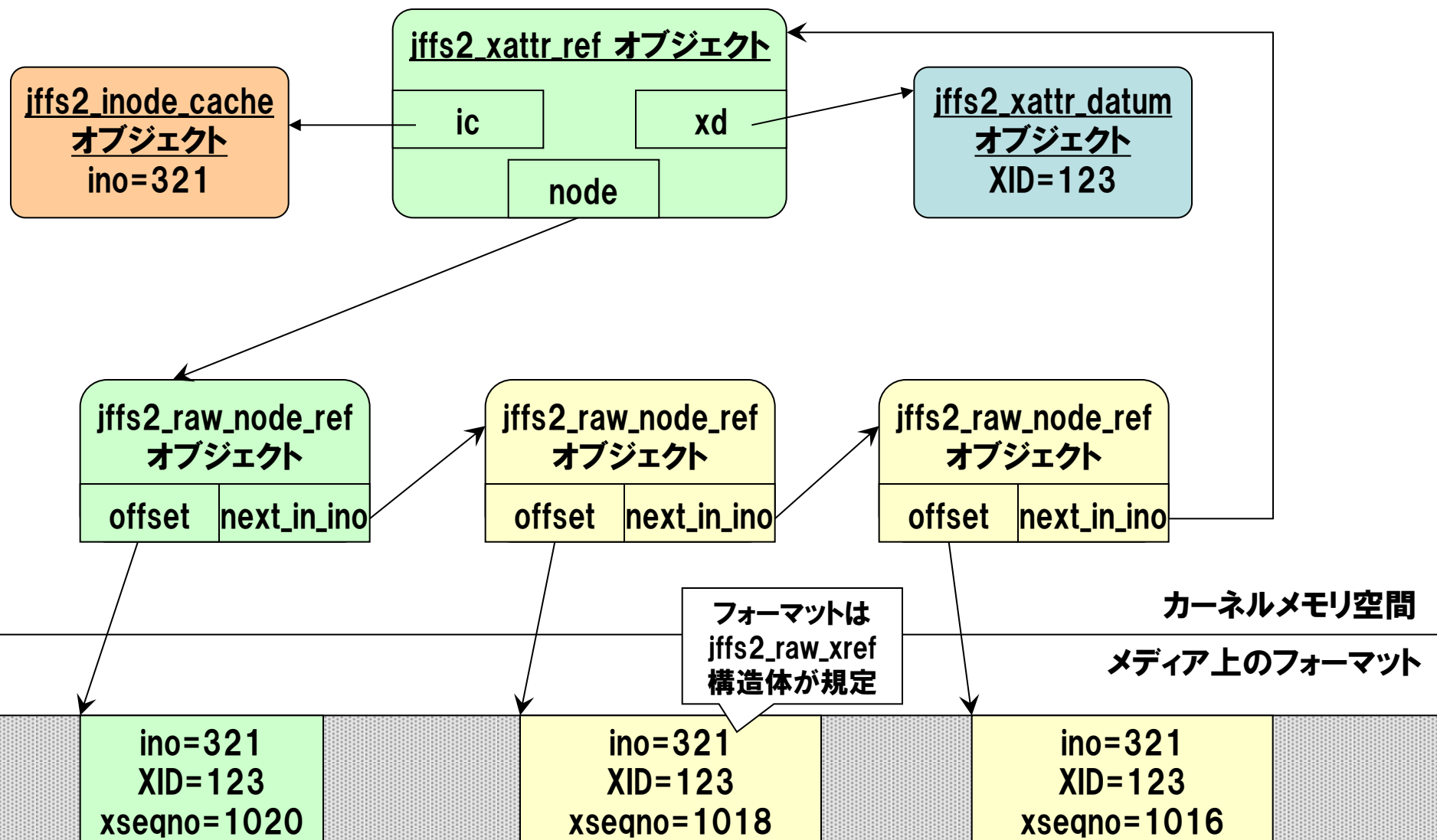


特許じゃなくてよかった。

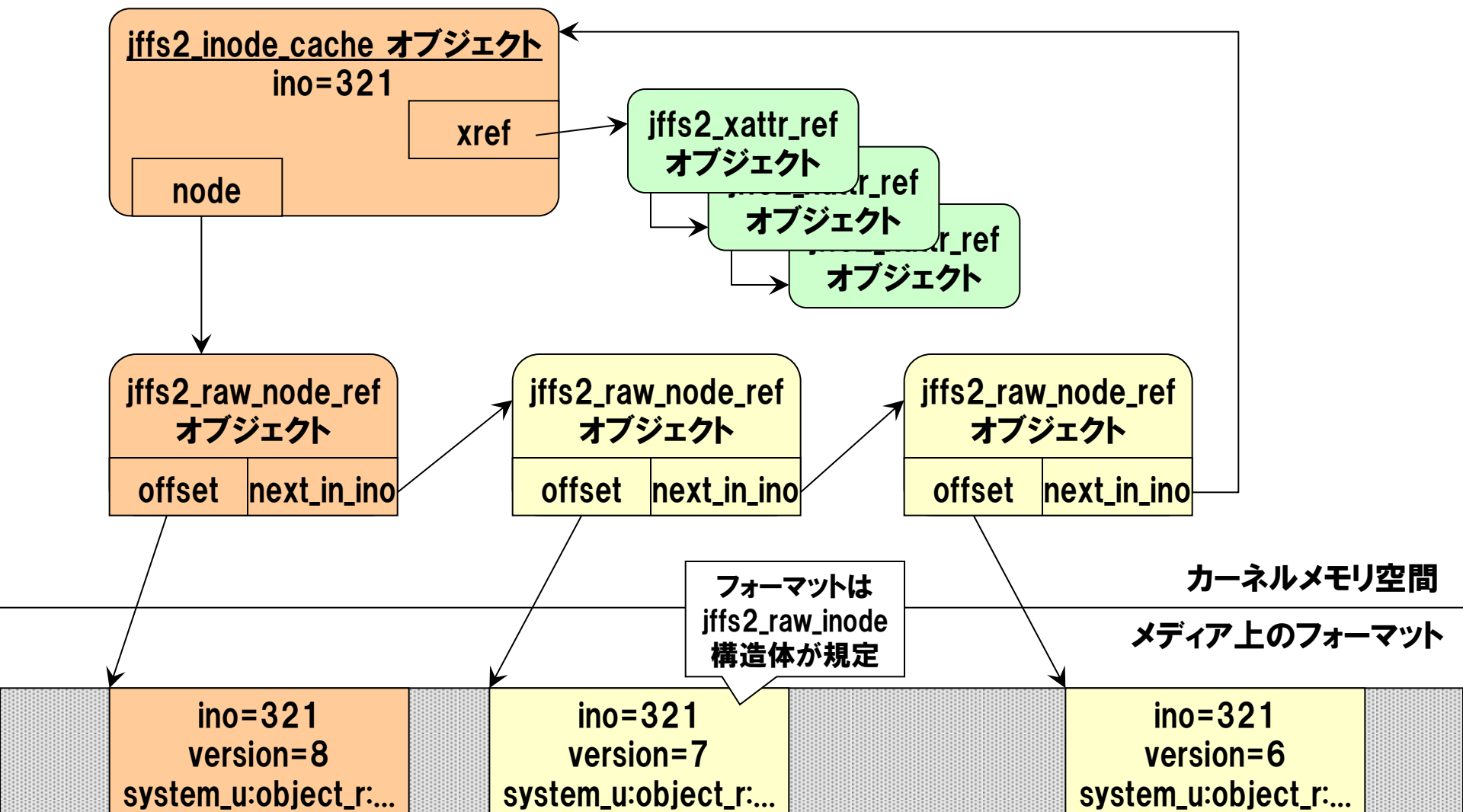
XATTR on JFFS2 のデータ構造 (1)



XATTR on JFFS2 のデータ構造 (2)



XATTR on JFFS2 のデータ構造 (3)



XATTR on JFFS2 のデータ構造 (4)

getxattr (file, xname, buffer, size)

ユーザ空間

カーネル空間

VFS

vfs_getxattr (dentry, xname, buffer, size)

inode_operation->getxattr

generic_getxattr (dentry, xname, buffer, size)

do_jffs2_getxattr (inode, xprefix, xname, buffer, size)

jffs2_inode_cache
オブジェクト

jffs2_xattr_ref

jffs2_xattr_datum
Name/Valueペア

jffs2_xattr_ref

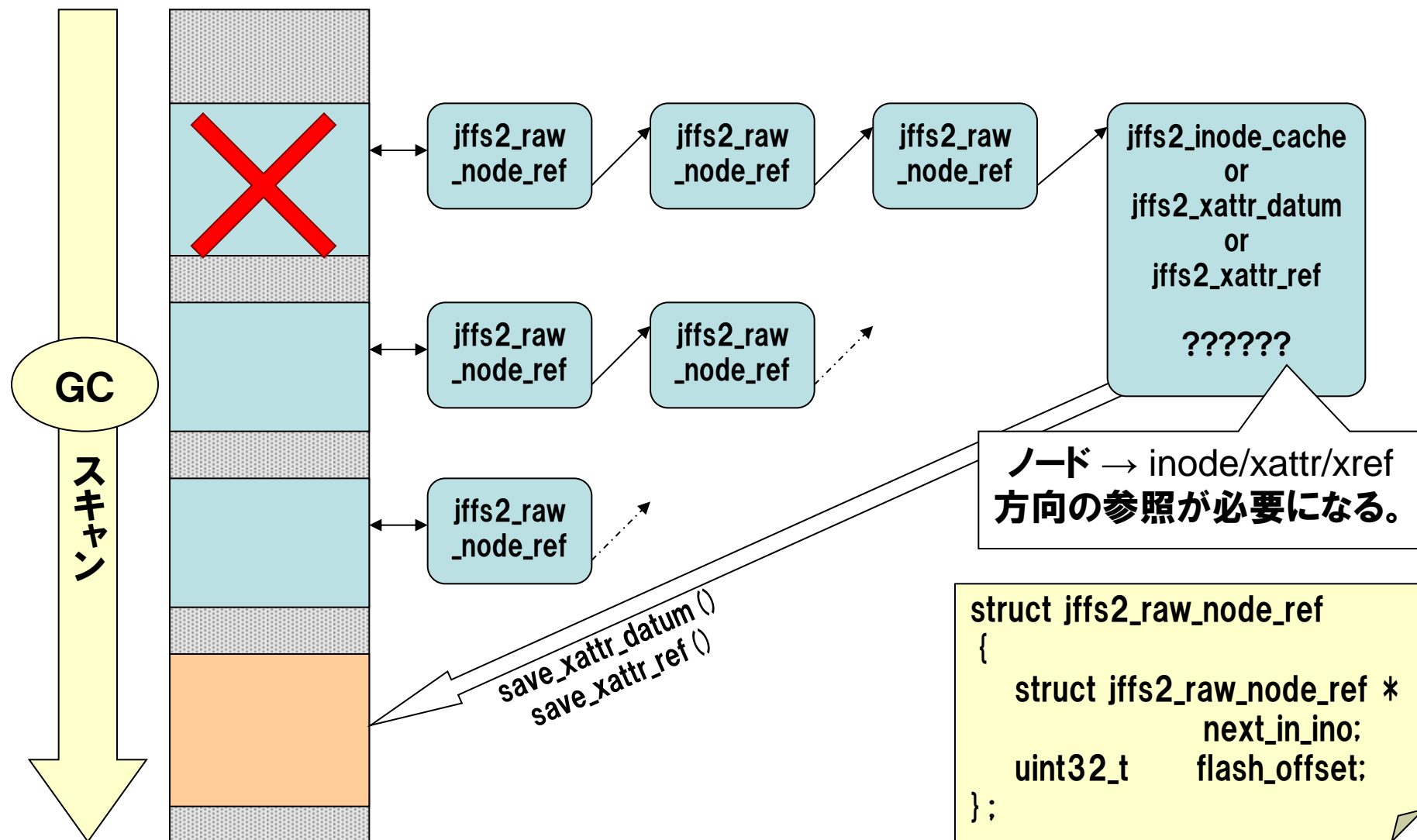
jffs2_xattr_datum
Name/Valueペア

xnameにマッチする
Name/Valueペアを探索

jffs2_xattr_ref

jffs2_xattr_datum
Name/Valueペア

XATTR on JFFS2 のデータ構造 (5)



XATTR on JFFS2 のデータ構造 (6)

jffs2_inode_cache構造体

void *always_null;		
struct jffs2_raw_node_ref *nodes;		
u8 class;	u8 flags;	u16 state;
u32 ino;		

RAWNODE_CLASS_INODE_CACHE

jffs2_xattr_datum構造体

void *always_null;		
struct jffs2_raw_node_ref *nodes;		
u8 class;	u8 flags;	u16 xprefix;
struct list_head xindex;		

RAWNODE_CLASS_XATTR_DATUM

jffs2_xattr_ref構造体

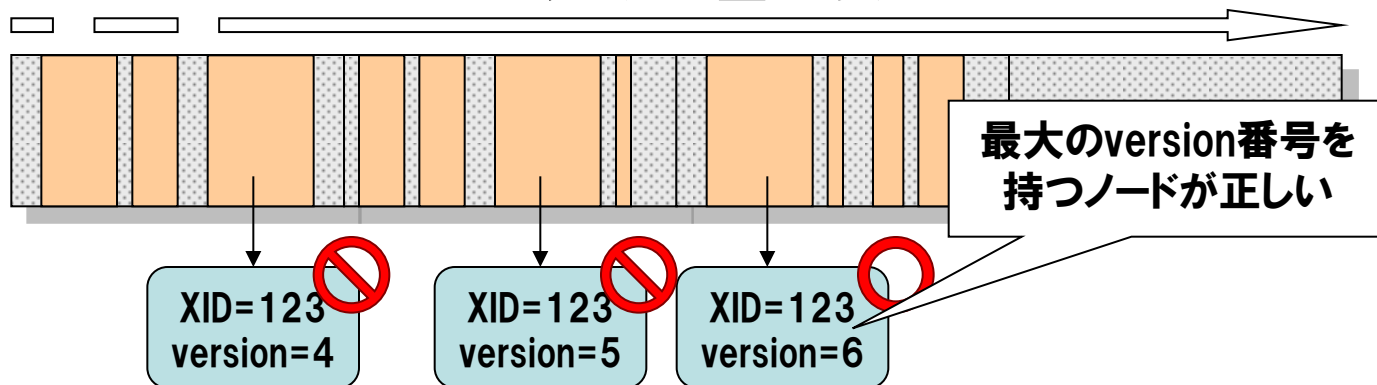
void *always_null;		
struct jffs2_raw_node_ref *nodes;		
u8 class;	u8 flags;	u16 unused;
u32 xseqno;		

RAWNODE_CLASS_XATTR_REF

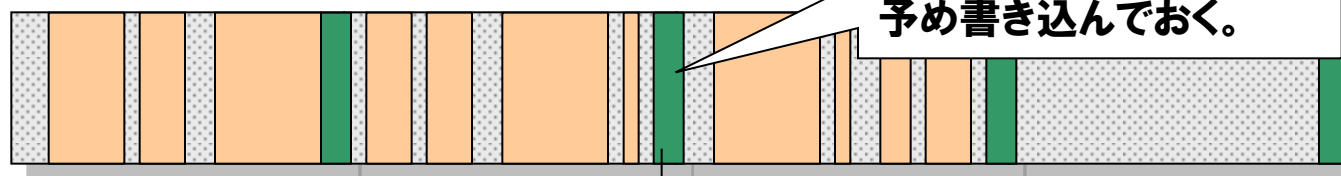
- jffs2_raw_node_refオブジェクトは非常に多く生成される
 - 構造体のサイズを増加させることは許されない。
- 構造体の頭の部分は共通のフォーマット
 - class変数を参照すれば、オブジェクトの型を特定できる。
 - 元々は32bit幅あったstate変数を、16bitに縮小して確保した領域
→ jffs2_inode_cache構造体のサイズは、jffs2_xattr_refへのポインタ分だけの増加で済んだ。(+4byte)
 - ガベージコレクタの処理を大部分共通化することを可能に。

マウント時の処理 (1)

メディアの全スキャン



EBS (Erase Block Summary) 機能

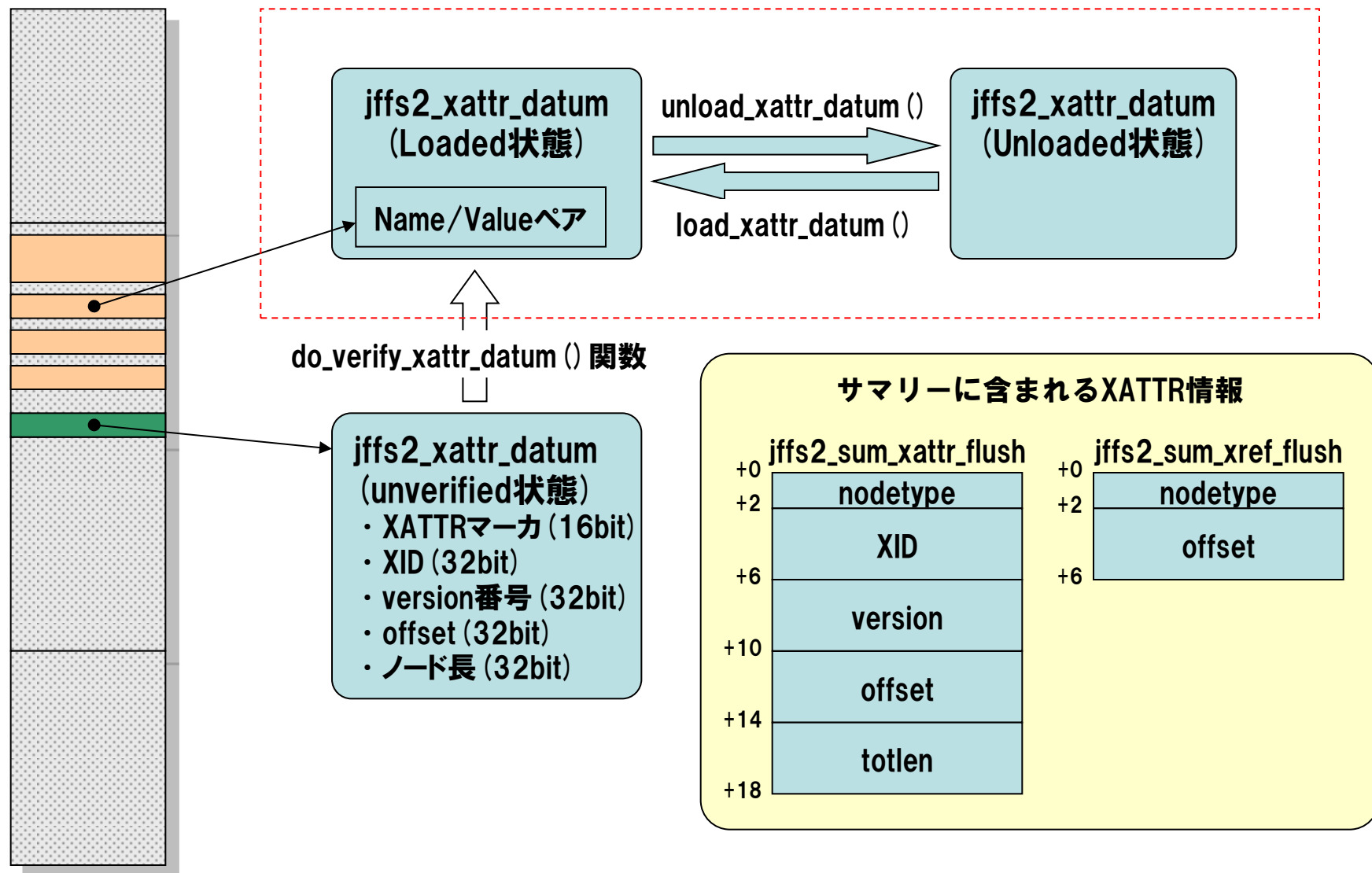


xattrの開発中、先に
本流にマージされた！

XATTRも対応が
必要になった

ココだけ読めばいい
⇒ マウント時間短縮

マウント時の処理 (2)



目 次

- ・ イントロダクション
- ・ XATTR on JFFS2
- ・ **コミュニティからのフィードバック**
- ・ コミュニティに飛び込んでみよう
- ・ 付録

ノードを削除する時の処理 (1)

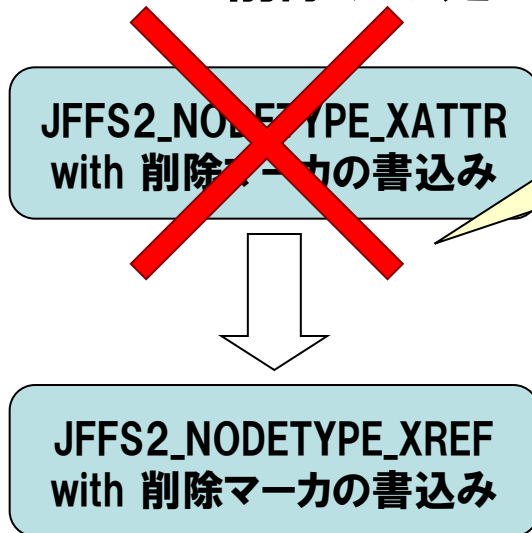
- **海外による当初の実装 (～revision.5) では**
 - xattr_datum, xattr_refは常に一個しかノードを持たない
 - ノードを削除するときには、ノードのヘッダを0xffで上書き
 - これはNAND型フラッシュメモリでは使えないと指摘
- **"delete marker"を使った実装 (revision.6) に変更**
 - 「このXATTRは削除されています」というマークを、最も大きなバージョン番号を持つノードに持たせる
 - 次回マウント時に、このXATTRはスキップされる
 - XATTR削除のアトミック性の問題
 - セットしたXATTRが"削除済み"になってしまう

ノードを削除する時の処理 (2)

・ XATTR削除のアトミック性問題

- どのinodeからも参照されないXATTRを「削除済み」として扱うよう修正

XATTRを削除する処理

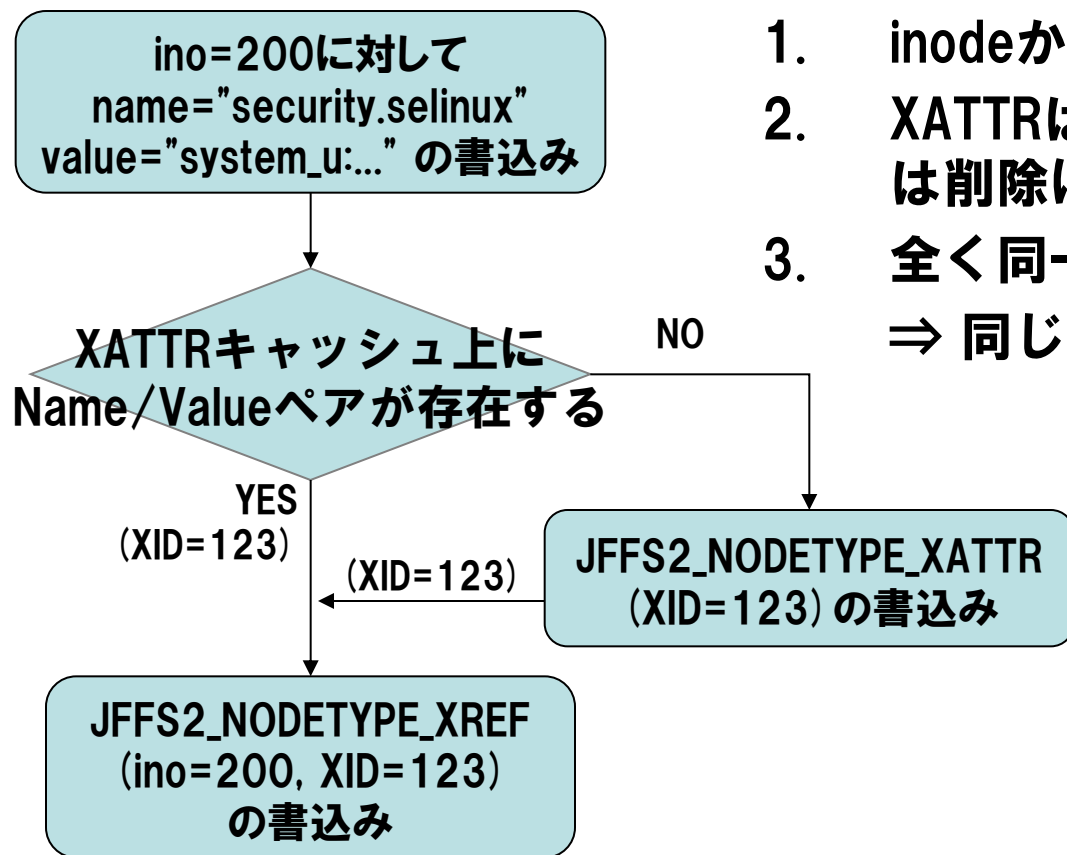


- ・ XREFを書き込む前に、、
メディアが一杯になったら？
OSがクラッシュしたら？

JFFS2は元々、マウント時にメディア上の全ノードをスキャンする
⇒「XATTRを参照するXREFの個数」を削除フラグとして利用すればよい

ノードを削除する時の処理 (3)

・書き込んだXATTRが“削除済み”になってしまう



1. inodeからXATTR (XID=123) を削除
2. XATTRは共有されるので、この時点では削除にならない
3. 全く同一のName/Valueペアをセット
⇒ 同じino/xidの組合せでXREFを生成

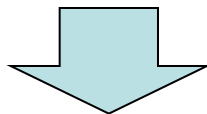
version番号を持つのは誰？

↓
XREF固有のversion番号を使うと、“削除済み”の方が新しいと判定される場合がある。

グローバルに一意的なシーケンス番号 (xseqno/31bit) を付与

David Woodhouse氏のリファクタリング

- ・ jffs2_raw_node_refのサイズ削減 (16byte→8byte)
 - 物理ノードを獲得するインターフェースが変更
- ・ jffs2_inode_cacheのフォーマット変更
 - GCの実装をinode/xattr/xrefで共通化



inodeの部分は修正したから、xattr, xrefの実装を新しいフォーマットに合わせてね。



David
Woodhouse

ファイルシステム全体を俯瞰した
より見通しのよいコーディングに

それは要らないんじゃないの？

・ XATTRのName/Valueペアの長さ制限

#define XATTR_NAME_MAX	255
#define XATTR_SIZE_MAX	65536

- XATTR/JFFS2では、EraseBlockの大きさに依存
(NOR-Flash : 64KB-128KB、NAND Flash : 8KB-16KB)
- 「複数ノードにまたがったXATTRを実装できない？」
- 「それって本当に必要なの！？」
 - ・ SELinuxやPOSIX-ACLでは、高々100byte程度で十分
 - ・ Ext2/3でも、XATTRの総サイズはBlock-Size (4Kbyte)
- 「Ext2/3でもその程度なら、まあ問題ないか。」

2.6.18カーネルでXATTR/JFFS2がマージ

[projects](#) / [linux/kernel/git/torvalds/linux-2.6.git](#) / [commit](#)

[summary](#) | [shortlog](#) | [log](#) | [commit](#) | [commitdiff](#) | [tree](#)

[JFFS2][XATTR] XATTR support on JFFS2 (version. 5)

author KaiGai Kohei <kaigai@ak.jp.nec.com>
Sat, 13 May 2006 06:09:47 +0000 (15:09 +0900)
committer KaiGai Kohei <kaigai@ak.jp.nec.com>
Sat, 13 May 2006 06:09:47 +0000 (15:09 +0900)
commit aa98d7cf59b5b0764d3502662053489585faf2fe
tree e98e83f3e69ebe3a1112394a19d440419e899749 [tree](#)
parent 4992a9e88886b0c5ebc3d27eb74d0344c873eeea [commit](#) | [commitdiff](#)

[JFFS2][XATTR] XATTR support on JFFS2 (version. 5)

This attached patches provide xattr support including POSIX-ACL and SELinux support on JFFS2 (version.5).

There are some significant differences from previous version posted at last December.

The biggest change is addition of EBS(Erase Block Summary) support. Currently, both kernel and usermode utility (sumtool) can recognize xattr nodes which have JFFS2_NODETYPE_XATTR/_XREF nodetype.

In addition, some bugs are fixed.

- A potential race condition was fixed.
- Unexpected fail when updating a xattr by same name/value pair was fixed.
- A bug when removing xattr name/value pair was fixed.

The fundamental structures (such as using two new nodetypes and exclusion mechanism by rwsem) are unchanged. But most of implementation were reviewed and updated if necessary.

Especially, we had to change several internal implementations related to load_xattr_datum() to avoid a potential race condition.

[1/2] xattr_on_jffs2.kernel.version-5.patch

[2/2] xattr_on_jffs2.utils.version-5.patch

Signed-off-by: KaiGai Kohei <kaigai@ak.jp.nec.com>

Signed-off-by: David Woodhouse <dwmw2@infradead.org>

その他、2.6.18カーネルにはJFFS2/XATTRの実装に合計20本のパッチがマージ

[JFFS2]	[XATTR]	Fix memory leak in POSIX-ACL support
[JFFS2]	[XATTR]	Fix xd->refcnt race condition
[JFFS2]	[XATTR]	coexistence between xattr and write ...
[JFFS2]	[XATTR]	Fix wrong copyright
[JFFS2]	[XATTR]	Re-define xd->refcnt as atomic_t
[JFFS2]	[XATTR]	Fix memory leak with jffs2_xattr_ref
[JFFS2]	[XATTR]	rid unnecessary writing of delete marker.
[JFFS2]	[XATTR]	Fix ACL bug when updating null xattr ...
[JFFS2]	[XATTR]	using 'delete marker' for xdatum/xref ...
[JFFS2]	[XATTR]	Fix obvious typo
[JFFS2]	[XATTR]	Handling the duplicate JFFS2_NODETYPE ...
[JFFS2]	[XATTR]	remove redundant pointer cast in acl.c
[JFFS2]	[XATTR]	remove '__KERNEL__' from acl.h
[JFFS2]	[XATTR]	remove senseless comment
[JFFS2]	[XATTR]	Unify each file header part with any ...
[JFFS2]	[XATTR]	'#include <linux/list.h>' was added ...
[JFFS2]	[XATTR]	Remove jffs2_garbage_collect_xattr (c, ic)
[JFFS2]	[XATTR]	Remove 'struct list_head ilist' from ...
[JFFS2]	[XATTR]	Add a description about c->xattr_sem
[JFFS2]	[XATTR]	remove typedef from posix_acl related ...
[JFFS2]	[XATTR]	XATTR support on JFFS2 (version. 5)

OpenZaurusにSELinuxを入れたみた

注意
表示部を無理に回転させないでください。
無理に回転させることがあります。

Connection: default

```
root@c7x0:/# id
uid=0(root) gid=0(root) context=system_u:system_r:initrc_t
root@c7x0:/# ls -lZ
drwxr-xr-x  2 system_u:object_r:bin_t
drwxr-xr-x  2 system_u:object_r:boot_t
drwxr-xr-x 11 system_u:object_r:bin_t
drwxr-xr-x 33 system_u:object_r:bin_t
drwxr-xr-x 13 system_u:object_r:bin_t
drwxr-xr-x  9 system_u:object_r:bin_t
drwxr-xr-x 10 system_u:object_r:bin_t
drwxr-xr-x  2 system_u:object_r:bin_t
drwxr-xr-x  2 system_u:object_r:bin_t
drwxr-xr-x 52 system_u:object_r:bin_t
drwxr-xr-x  2 system_u:object_r:bin_t
drwxr-xr-x  2 system_u:object_r:bin_t
drwxr-xr-x 11 system_u:object_r:bin_t
lrwxrwxrwx  1 system_u:object_r:bin_t
drwxr-xr-x 10 system_u:object_r:bin_t
drwxr-xr-x 11 system_u:object_r:bin_t
root@c7x0:/# ps -Z
LABEL                                PID TTY          TIME CMD
system_u:system_r:initrc_t          3531 pts/0    00:00:00 sh
system_u:system_r:initrc_t          4024 pts/0    00:00:00 ps
```

セキュリティコンテキストが表示されている

kernel: 2.6.16-oz + KaiGaiパッチ
Policy: SPDL (Simplified Policy)
libselinux, libsepol, /sbin/init, busybox
等を追加インストール

OLPC (One Laptop Per Child) プロジェクト

- ・ 発展途上国の子供たちにPCを！
 - Digital Divideの解消
 - Linuxの普及・啓発
- ・ \$100PC
 - 100万台のオーダー × 数カ国
 - 記憶デバイスは512MBのFlash-ROM + JFFS2
 - セキュリティ強化にSELinuxの利用
 - ・ 多数の同一構成マシンが出るので、0-dayアタックは超脅威
 - ・ パッチ適用が難しい環境での利用もあり得る
 - ・ SELinuxの利用には、JFFS2上でのXATTRのサポートが必須



今後の課題

- **JFFS2向けPOSIX-ACLのメモリ使用量削減**
 - Ext2/3のPOSIX-ACLを参考にして実装
 - JFFS2のXATTRはname/valueペアをキャッシュできる
- **busybox向けSELinuxコマンドの開発**
 - chcon, setsebool, setenforce 等々
- **組み込み分野に特化したポリシーの開発**
 - Reference Policyはサーバ利用を前提
 - ポリシーサイズの削減は attribute の利用が鍵
 - サーバ用でないディストロ向けのセキュリティポリシー

目 次

- ・ イントロダクション
- ・ XATTR on JFFS2
- ・ コミュニティからのフィードバック
- ・ **コミュニティに飛び込んでみよう**
- ・ 付録

コミュニティに飛び込んでみよう(1)

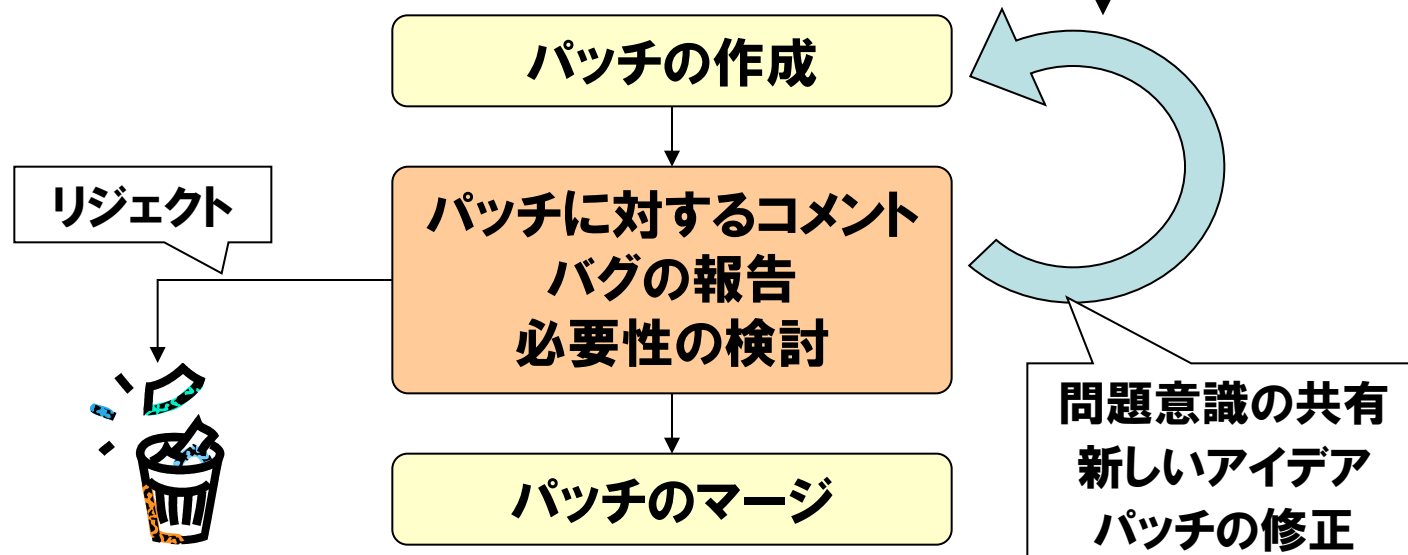
- **どこで開発やってるの？**
 - 基本はMLでの意見交換
 - ・ パッチ付きのメールで意見を求めることが多い
 - ML, SCM, Web-siteがワンセット
 - 過去のアーカイブで雰囲気をつかもう
 - ・ <http://marc.theaimsgroup.com/>
- **誰が開発しているの？**
 - 企業に属している人が多いが、誰でも参加可能
 - キーパーソンの発言に注目

コミュニティに飛び込んでみよう(2)

・ 必要なのは？

- 70%の コミュニケーション能力
- 20%の エンジニアリングスキル
- 10%の 語学力

コミュニティからの
フィードバックの活用が
OSS開発の鍵！！



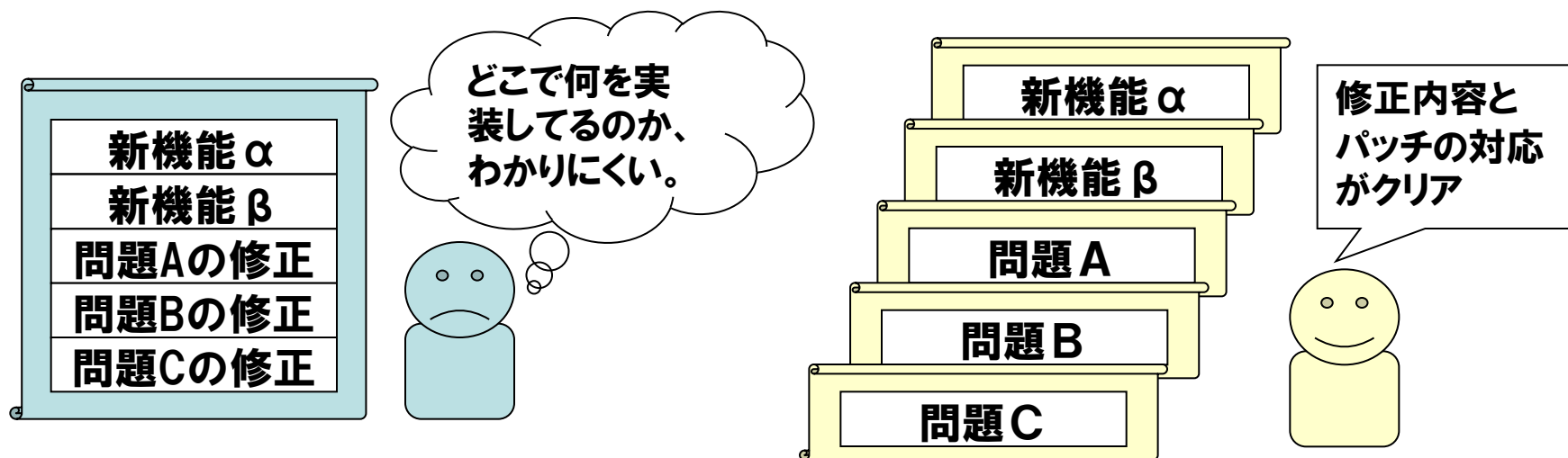
語学力が不安だ

カタコトの日本語を話す外国人のコードが、
バグだらけで価値の無いものだと思いますか？

- ・ 大切なのは、あなたのコードに価値があるのか？技術的に優れているのか？
- ・ 流暢である必要はない、伝われば十分
- ・ 自信が無ければ、機械を頼ってもOK
 - － スペルチェッカー
 - － 自動翻訳も参考になる

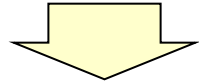
より良いフィードバックを得るために(1)

- パッチは細かく、分割しよう
 - 数千行のパッチをどかっと送られてきても、見るほうも大変
 - 一個のパッチで一個の修正というのがベスト
 - Linuxなら Documentation/SubmittingPatches で明文化



より良いフィードバックを得るために(2)

- ・ **ドキュメント/コメントも重要**
 - 添付のパッチが何をするもので、どんな変更を加えたのか？
 - どのようなアルゴリズムを用いているのか？
 - どんな関数/構造体を追加したのか？

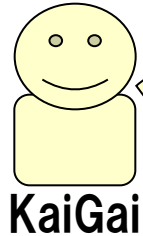


- ・ **ドキュメントが無いと、、、**
 - レビューする人が、リバースエンジニアリングと同じことをしなければならない
 - パッチに対するフィードバックを得られないことも

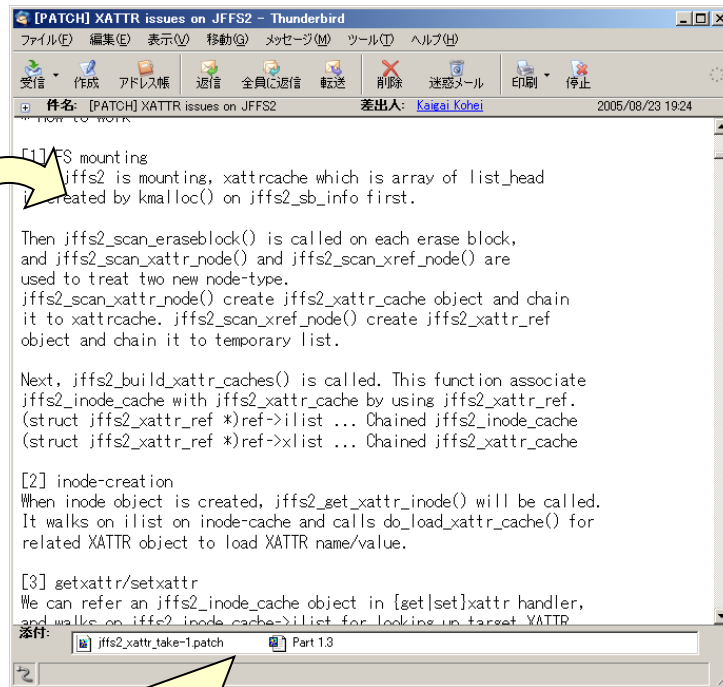
より良いフィードバックを得るために (3)

- ・ コーディングスタイルは揃えよう
 - ソースコードは人類の共有財産
 - "オレ流"は御法度
 - 各OSSのガイドラインに従おう
 - ・ Linuxなら Documentation/CodingStyle に明文化
 - ・ タブを残す？残さない？
 - ・ インデントの幅は8？それとも4？
 - ・ '{'の位置はどうする？
 - ・ 関数/変数の命名規則
 - ・ emacsのモードが用意されていたり

失敗例(1) ～XATTR/JFFS2パッチ～

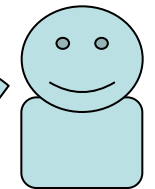
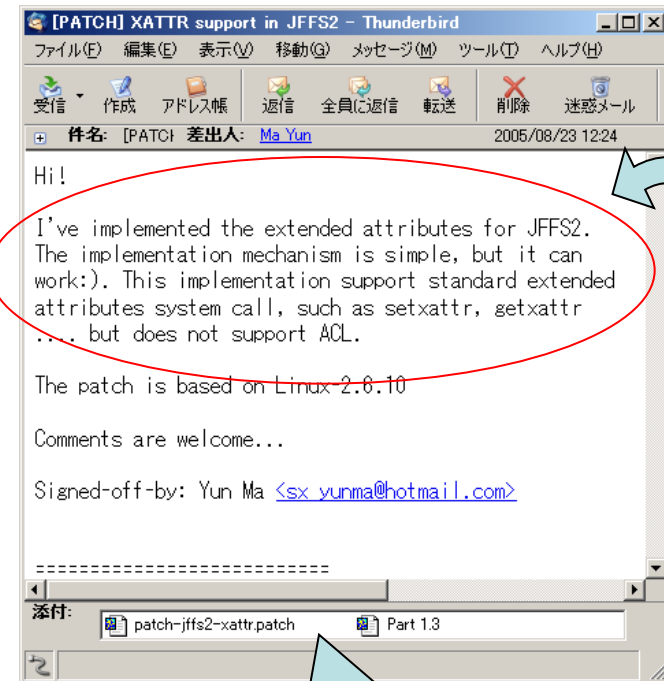


KaiGai



1500行のパッチと、130行の説明

- ・新たに追加したデータ構造
- ・処理の流れ
- ・実行例
- ・今後のロードマップ



Ma Yun

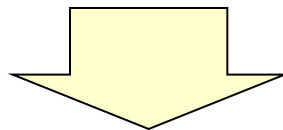
数行のコメントと、
650行のパッチ

パッチのマージ作戦(1)

- ・ **"ニッチな領域"から攻めていく**
 - 人が多い分野 = 競争も激しい
 - 他のプロジェクトと競合すると泥沼
 - 説明責任
 - ・ なぜ、あなたの提案は他の人よりも優れているのか
 - ・ パフォーマンスなど、数字で比較できれば楽だけど、、、
 - 過去に同じ提案がリジェクトされていないか？
- ・ **他の人が目を付けていない分野にフォーカス**
 - 利用頻度が少ない = テストが十分でない
 - "ここなら自分がNo.1"という領域で戦う

パッチのマージ作戦(2)

- ・ 分野の垣根を越えた提案をする
 - SELinuxスケーラビリティ向上 (SELinux+HPC)
 - JFFS2のXATTR対応 (SELinux+組込み)
 - PostgreSQLのSELinux対応 (SELinux+RDBMS)



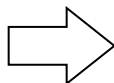
- ・ コミュニティ間の交流が無いことで、他分野のノウハウが活用されないケースがある。
- ・ 一つ一つはメジャーな分野でも、ANDを取ると非常にニッチな分野になる場合がある。

パッチのマージ作戦 (3)

- ・ 「俺がやるよ！」宣言をする
 - 競合するプロジェクトがあれば教えてもらえる。
 - ・ 誰にコンタクトを取るべきか
 - ・ どんな作業が進んでいるのか
 - 興味を持った人が協力してくれる。
 - 自分のところに情報が集まってくる。
 - 言った以上は後には引けなくなる。



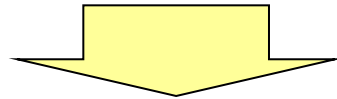
「SELinuxをRCUで高速化します！」宣言
「SELinux対応PostgreSQLを作ります！」宣言



Paul E. McKennyやRussell Cocker, Joshua Brindleら、
一流のエンジニアを開発に巻き込むことができた。

失敗例 (2) ～タイプ継承・ポリシーコンパイラ～

- ・ **タイプ継承可能なSELinuxポリシーコンパイラ**
 - 複数タイプに共通するパーミッション定義を、一回の記述で行うことを可能にする。



- ・ **Tresysの開発していたReference Policyと競合**
 - 類似のコンセプトを持っていた
 - ・ ポリシーのモジュール化/インターフェースの定義
 - コミュニティからのフィードバックを反映させたが、優位性を説明することができなかった
 - 最終的にはReference Policyの開発に協力することに

最後に

- ・ **コミュニティ = Human Network**
 - エンジニア同士のコミュニケーションの場所
 - パッチは何度でも修正できる。まずは、彼らと問題意識を共有することが大切。
 - フィードバックを大切にしよう
- ・ **郷に入れば、郷に従え。**
 - コミュニティには（暗黙の）ルールがある
 - コードの書き方、パッチの投稿の仕方、etc

Any Question?



目 次

- ・ イントロダクション
- ・ XATTR on JFFS2
- ・ コミュニティからのフィードバック
- ・ コミュニティに飛び込んでみよう
- ・ **付録**

付録 ～XATTR/JFFS2 開発の経過～

- Revision.1 ('05/08/23)
 - XATTR/JFFS2のプロトタイプ。GCは未実装。
- Revision.2 ('05/09/07)
 - XATTRノードに対するGCのサポートを追加。
- Revision.3 ('05/09/28)
 - Revision.2でのデッドロックのバグを修正したバージョン
- Revision.4 ('05/12/28)
 - Posix-ACLのサポートと、2.6.14カーネルでのXATTR初期化に対応
- Revision.5 ('06/05/09)
 - EBS (Erase Block Summary) サポートを追加
 - gitツリーのベースとなる
- Revision.6 ('05/06/24)
 - "Delete Marker"によるノードの削除に変更