FURUKAWA ELECTRIC

Bound to Innovate

HAに挑む

平成22年11月1日

MPLS JAPAN 2010

福永 智之
古河ネットワークソリューション株式会社

MPLS JAPAN

はじめに

FURUKAWA ELECTRIC

- 最近はHAといえば「クラスタ」
- ネットワークのHAを考えると、アイテムがとても多い
  - レイヤ(L1, L2, L3)
  - 機器(PE, switch, CORE, RR, end-Node)
  - 対象(H/W, 回線)
  - 方式(NSR, ISSU, FRR, BFD..)
  - 設定(timer, action..)
- これらを、どこで障害が発生しても矛盾・対立なくサービスを続けられるかの考慮が必要
- とりあえず本日は、箱のメーカとして、箱(L3)の立場からのHAを語ります

これまでのHAへの取り組み

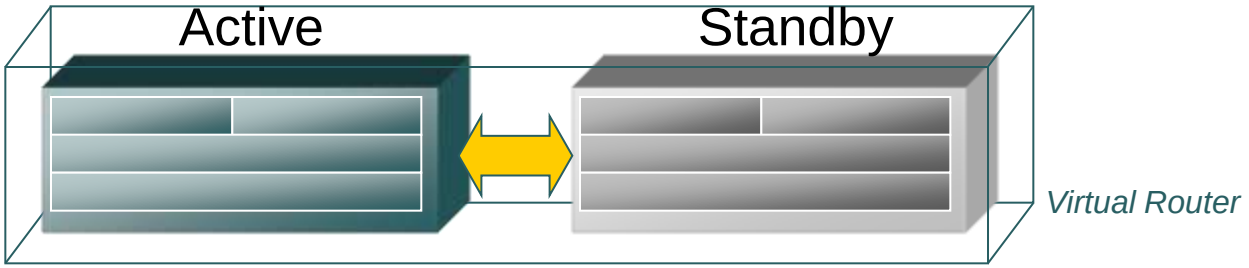
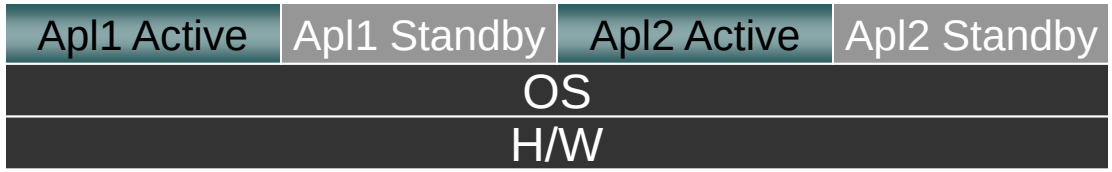
FURUKAWA ELECTRIC

	HA機能への取り組み	世間の動き
2002	TCPのhandoverを内々に成功させる	2002 W杯(日・韓)
2003		はやぶさ打ち上げ
2004	Smooth Update(無停止モジュールアップデート)	地デジ開始 紙幣デザイン変更
2005		アテネオリンピック blog, SNS 京都議定書
2006	Smooth Update 拡張	郵政民営化 WBC クラウド
2007	IPsec-HA ISSU対応	2006 W杯(独) 無料動画
2008		JANOG10周年 スマートフォン 北京オリンピック
2009	NSR対応	リーマンショック オバマ政権
2010		政権交代 上海万博 2010 W杯(南ア)

HA.1 smooth update (1)

FURUKAWA ELECTRIC

- ソフトウエア・モジュール単位の無停止アップデート
  - 詳細は2004年MPLS-Japan発表資料を参照
- ISSUの対象を細分化

装置	
ハードウエア モジュール	
ソフトウエア モジュール	

HA.1 smooth update (2)

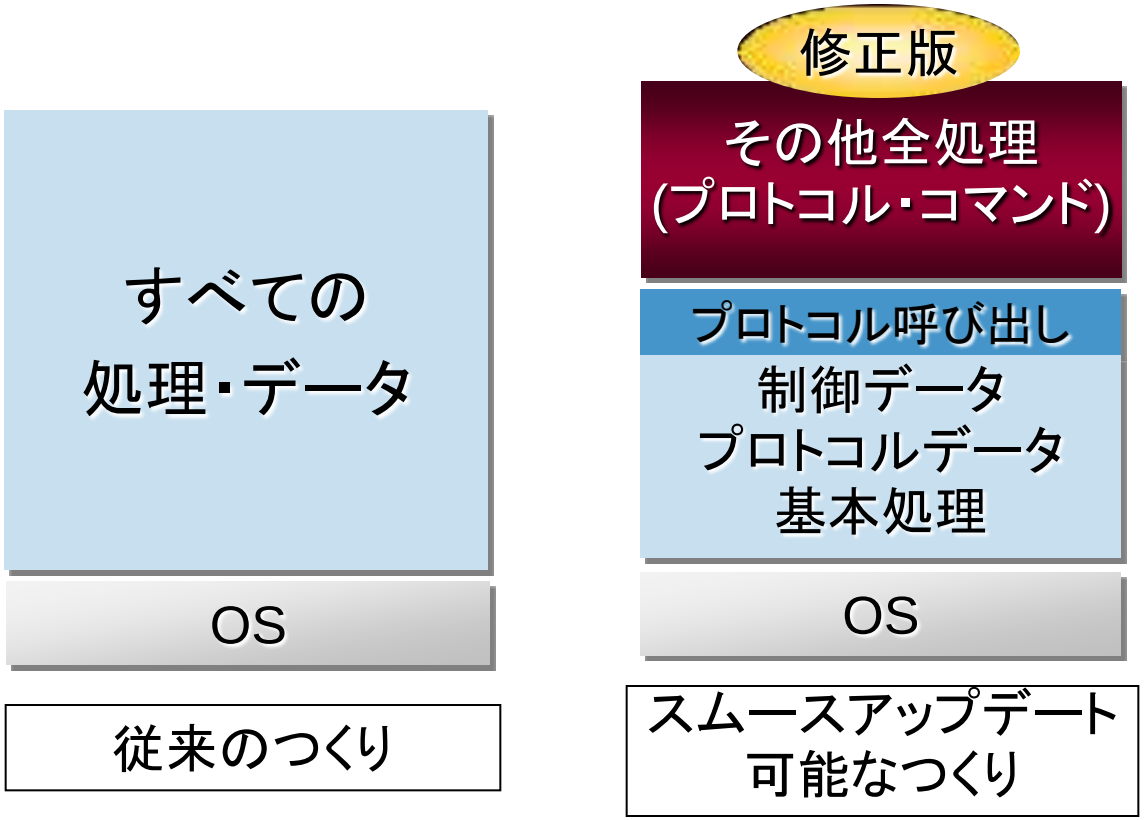
FURUKAWA ELECTRIC

- 以下の種類のモジュールに対応
  - CLIのなかま
    - config, command
  - 止めても影響が比較的少ないモジュール
    - SNMP, NTP等ステートレスなモジュール
  - 止めてはいけないモジュール
    - BGP, OSPF, LDPなどのステートを持つプロトコルたち
  - カーネル等低レイヤのモジュール
    - 2004年当時は出来なかったけど最近はやってる
- フィールドでの運用実績あり
  - アップデート中はtelnet排他等の安全機能
  - 但しロールバックできない→前の版に更新し直すことで何とか
  - データベースが変更になった場合はコストが発生（DB再構築）
  - アップデートしたモジュールはshowで確認できる

HA.1 smooth update (3) 止められないモジュール

FURUKAWA ELECTRIC

■ 予めスムースアップデート可能なつくりにしておく必要あり



プロトコルデータ・制御データを維持したままプロトコル処理等を入れ替える
→プロトコルセッションを維持したままアップデートが出来る。つまりISSU。

MPLS JAPAN

HA.1 smooth update (4) 低レイヤモジュール(1)

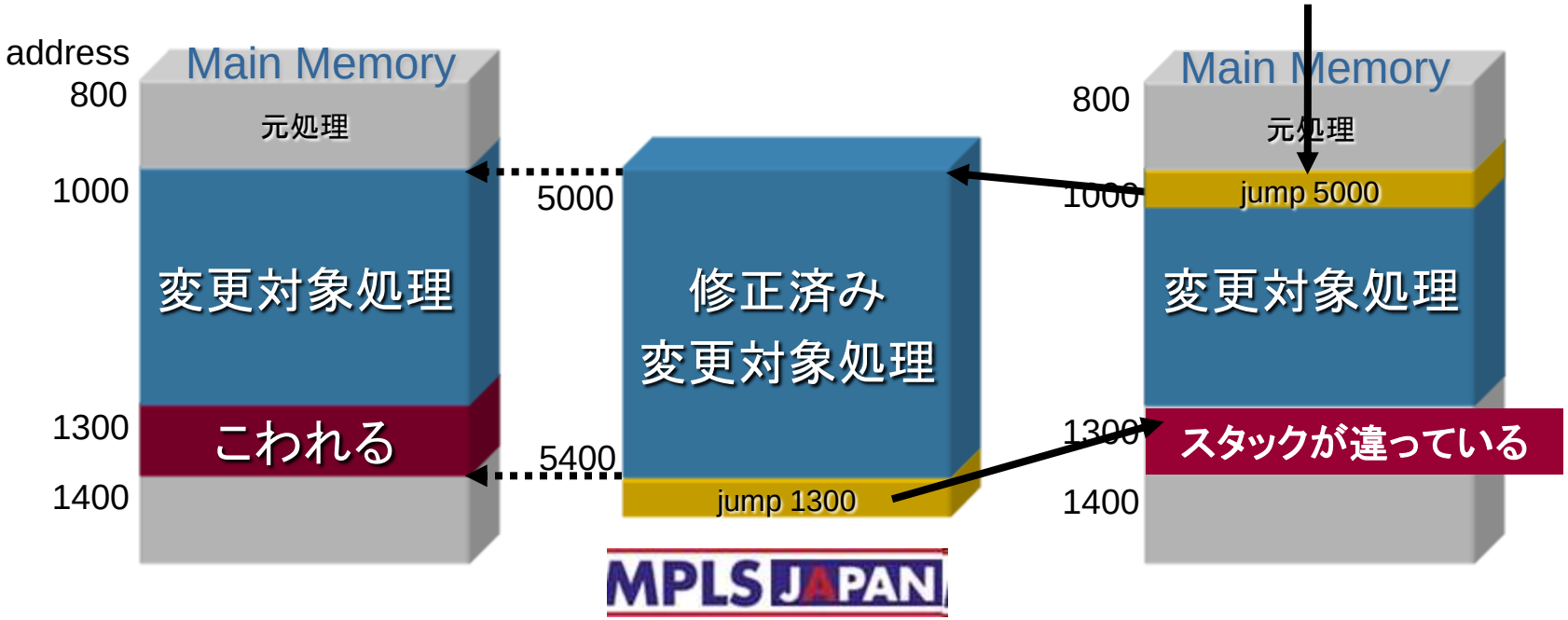
FURUKAWA ELECTRIC

- スムースアップデート可能なつくりになっていない && 止められないモジュールもISSUしたい
- スムースアップデート可能なつくりになっていないので同手法は使えない
- どうする？　→　バイナリパッチしかない
- コマンド一発でバージョンチェック・修正後バイナリ展開・パッチ当て作業・作業のチェックを行うツールを作成・ご提供
- フィールドで無事kernelなどのISSUに成功

HA.1 smooth update (5) 低レイヤモジュール (2)

FURUKAWA ELECTRIC

- 稼働中のバイナリ入れ替えは至極簡単なものから頭を使うものまで多種多様。
- バイナリサイズが変わらない変更(論理変更など)
  - そのまま上書きすればよい（例:”jne” -> “je” 1bit変更）
- バイナリサイズが変わる変更
  - いろいろと工夫が必要



MPLS JAPAN

HA.1 smooth update (6) 運用実績

FURUKAWA ELECTRIC

- フィールドでの適用実績
  - 適用延べ回数：約2000回
  - 適用失敗回数：3回（詳細は後述）
  - 無停止アップデート率：99.8%

- *なぜ失敗したのか....（今後の反省のために）*
  - 新モジュール展開処理と、プロトコルの高負荷処理が重なった場合にプロトコル側のタイムアウトが発生
  - 障害対応でアップデートした際、同障害により破壊されたDBを修復するプログラムも同時に実行したところ、同プログラムとアップデートモジュールの初期化処理が重なり一時高負荷状態に。
  - 新モジュール展開場所の誤りがあった（展開処理でエラーしたために影響はなし）。

HA.2 planned NSR

FURUKAWA ELECTRIC

- Planned NSR: コマンドによるCPU切替
- 時代はISSU/NSR。(smoothだけでは付いていけない)
- 以前から持っていたTCPハンドオーバ技術とプロトコル側の移転技術・コンフィグ移転技術を組み合わせてplanned NSR機能を開発
- 基本動作は割にすんなり動いた。そして熟成、、な時期に、
- **「やはりUnplanned NSRが必要ではないか」**(上から)

MPLS JAPAN

HA.3 unplanned NSR

FURUKAWA ELECTRIC

- TCP/UDP常時同期技術は新開発。しかし、「Standbyと同期してからピアにACK返せばOK! 」なんて簡単なものではない。
  - Nagleアルゴリズムや遅延ACK
  - 各種タイマ
  - 未来ACK
  - 3way handshakeのタイミングも微妙に異なる。connect, accept...
- さらにコンフィグやプロトコルも常時同期へ
  - コンフィグ同期中にマスタが落ちたら・・？
  - ピア+Active+Standby+フォワーディングプレーン全員が同じ情報を持つ必要あり
- 冗長はCPUだけではなく、他のデバイス（回線コントロールカードなど）も冗長。冗長2系統が同時に落ちた場合の考慮が大変。さらにそのときにI/Fダウン、回線カード抜き、切り戻しがあっても大丈夫か・・・。
- とか色々ありましたが、何とかunplanned NSRを完成
  - CPUをいきなり抜いても転送・シグナリングのロスは「0」で切り替わる。（ただし抜いた直後のみ、フォワーディングプレーンへの経路反映に2秒ほど要する）

ISSU 3段階を経た今...

FURUKAWA ELECTRIC

- やはり圧倒的に難しいのはUnplanned NSRだった。
  - 故障が発生して切り替える処理よりは、いつ切り替わってもよいための「常時同期」が曲者。
  - 別CPUである以上、ある程度の同期は出来ても「絶対的完全同期」は不可能。その差分をギリギリまで詰めるのか、逆に差分をどうリカバリするのかで実装方針が変わる。
  - その中でも一番辛いのが、「二重故障」
    - 抜き＋抜き
    - planned switchover + 抜き
  - いつ発生するかわからない故障のために常に同期して動き続けるコスト
    - 今後、全開発機能が常時同期上で動くことが前提
    - レビュー観点、試験項目全てに常時同期の観点が組み込まれる

NSRとHA

- 二重故障をはじめ、NSRが巧く動作しない状況は正直まだあります
- メーカとして当然ながら、上記の問題や、まだ見えない問題を解決して行きます
- ただ、、どこまで追求していくのか、ある程度で見切ってより単純且つ効果的なHAを考えるのか、色々な立場の方々のご意見を聞いて考えてみたいと思います。