FBOSS experience of migrating massive scale networking systems to SAI
FBOSS experience of migrating massive scale networking systems to SAI

Shrikrishna Khare, Software Engineer, Facebook
Rajan Kumar, Software Engineer, Facebook
FBOSS

- Facebook Open Switching System (FBOSS)
- Facebook’s software stack for controlling/managing network switches deployed in Facebook’s Datacenters
FBOSS Architecture

- **SwSwitch**
- **HwSwitch**
  - BcmSwitch
  - SaiSwitch

**ASICs**:
- ASIC 1
- ASIC 2
- ASIC 3
- ...
FBOSS + SAI

• SAI
  ○ Switch Abstraction Interface
  ○ Project under Open Compute Project (OCP)
  ○ Open source API to control forwarding elements
  ○ Vendor independent

• FBOSS SAI based implementation:
  ○ HwSwitch: multiple ASICs, ASIC vendors
  ○ Easy to onboard newer ASICs
  ○ Open source contributions
    ■ FBOSS is open source
    ■ Facebook contributes to SAI spec
Development Strategy

- Big Matrix: [ASICs] X [Roles] X [Features]
  - [TD2, TH ] x [RSW, FSW… ] x [ACLs, QoS...]
- Not every combination is used in the production
  - ✅ [TD2][RSW][ACLs], [TH3][FSW][Mirroring] …
  - ❌ [TD2][FSW][*], [TH3][RSW][LAG] …
- Develop [Features] for a subset of [ASICs][Roles]
- Deploy while developing for other [ASICs][Roles] in parallel
- First phase: RSW: fewer features, but large deployments
- Later phases: other switch roles, require more feature support

TD2: Trident2, TH: Tomahawk, TH3: Tomahawk3
RSW: Rack Switch, FSW: Fabric Switch
Development Model

- SwSwitch remains same, but new HwSwitch: SaiSwitch
  - SwitchState delta applied to ASIC using SAI APIs
- Validation:
  - HwTest: verifies an aspect of functionality used in prod
  - Extensive coverage: 500+ tests
  - Criteria: if it passes on non-SAI, must pass on SAI
Development Model (contd.)

- Development in close collaboration with Broadcom
  - Broadcom provides SAI implementation
  - At times, parallel feature development: FBOSS & BRCM-SAI
  - Periodic EA drops from BRCM, and GA on Feature complete
  - Facebook contributed several patches to BRCM-SAI
  - Debugging:
    - Joint debug calls
    - SAI Replayer: auto-generated C code with SAI API calls from FBOSS
Brownfield Deployment

- Push: continuous process that updates FBOSS software
- Uses qualified bundle
- Schedules new SAI migrations (disruptive, cold boot)
- Updates devices already running SAI (non-disruptive: warm boot)
Challenges

- Vendor SAI implementation
- FBOSS SAI implementation
- Push Tooling
- Ability to drain devices

Subtle bugs that can only be found in production despite a large test suite
Vendor SAI implementation

- ACL drops srcMAC == routerMac ingress on non-CPU port
  - Security ACL: default created by the SAI implementation
  - One prod service sent such traffic
- Incorrect ECMP hash configuration
- Route points to Drop instead of pointing to CPU
- Rare race during callback processing and warmboot shutdown
FBOSS SAI implementation

• ACL counters programmed but not exported
• Queue watermark stats not created for all queues
• Link flap on few ports of few rack types
• Route incorrectly programmed to CPU instead of port
Mitigation

- Pause migrations, and resume with fix
- Pause only for affected ASIC/Role/Deployment type
- Challenge on resumption
  - new migrations
  - ‘fixing’ affected devices without traffic disruption
- Fixing as part of the regular Push vs. one-off
- Warmboot one-off vs. disruptive one-off
- Feedback loop: prevent recurrence
  - Introduce HwTests to capture scenario in the bug
  - HwTests run on-diff, continuous runs
Mitigation: Rare bug

- Rare bug, could not reproduce:
  - HwTest
  - Series of retries of production workflow
- Resume SAI rollout with:
  - Extensive targeted logging
  - Pre-undrain detection: don’t return to prod if bug found
  - Continuous monitoring and remediation for ALL devices
- Longer Term
  - Detect discrepancy between SwSwitch, SaiSwitch, ASIC
  - Replayer for SwSwitch, SaiSwitch
FBOSS PHY Management

- FBOSS networking switches use internal and external PHY devices
- PHY devices are managed by vendor SDK and the homegrown SDK
- FBOSS support multiple switches at various networking layers having different PHY devices
- Needed a standard interface to manage all PHY devices
Migration to SAI

- Consolidate PHY management function calls from application to channel through SAI layer
  - PHY initialization, firmware download, port/lane settings, counters, flags, MACSEC
- PHY management functionality moved to a different process to work with SAI based driver
- PHY management goes through SAI switch, an adaptation layer through which NPU and PHY are managed using respective SAI based drivers
Advantage

- Common abstract interface for all kind of PHY devices
- Ease of migration from one vendor to another. Ability to share PHY management code across vendors
- Integration with home grown SAI switch adaptation layer to support features like warm-boot, API logging/replayer
- Alignment of FBOSS switches to leverage SAI for all ASIC programming in the switch
Advantage for testing

- Able to leverage the common HW Test infrastructure for testing the Phy functionality like MACSEC
- Able to leverage common unit test infrastructure build on top of SAI Switch which uses the Fake SAI (software emulation of SAI API)
Challenges

- Not many vendors in the PHY space providing SAI based SDK as of now
- Maintain dual PHY management support in code - for SAI based SDK and non-SAI SDK
- SAI is still evolving for PHY functionality. Gap in the feature/functionality exists
- Common SAI adaption layer mandates common SAI API between various devices to have same attribute support
Road Ahead

• Move all existing PHY SDK to SAI based SDK
• Strengthen SAI API support for PHY functionality
• More counters and debug flags in SAI to aid PHY debugging (Work with vendor to get the port level counters, lane status flags etc implemented)
• Device software emulation model support addition for FBOSS SAI test infrastructure
Call to Action

- SAI Spec revisions should not break warm-boot
  - e.g. enum re numbering has broken warm-boot in the past.
- SAI Spec enhancements
  - Faster turnaround
- SAI Spec needs to add more counters, debug ability for the PHY
  - e.g. Link level parameters like SNR, BER, Eye diagram
Thank you!