Fast-Reboot

What Fast-Reboot is?

- Fast-Reboot – updating the control plane with short (<=25 secs) disruption of the data plane.
- Requirements:
  - data plane disruption not more than 25 seconds
  - control plane disruption not more then 90 seconds:
    - data plane will use stale RIB/FIB information while control plane reboots
  - 2000 hosts connected to vlan interfaces
  - 6000 ipv4 + 3000 ipv6 /64 bgp routes

How it works. Required technologies

1. LACP in slow mode:
   - updates every 30 seconds
   - timeout in 90 seconds
2. quagga BGP supports BGP graceful restart: rfc4724
3. quagga BGP supports announcing of preserved Forwarding state
4. Linux kernel has enabled kexec feature, which enables load another linux kernel without cold reboot of the switch
5. Testbed supports:
   1. link-state propagation from fanout ports to veos VM virtual ports
   2. arp responder to emulate hosts in vlans responding on arp requests

How it works. Before control plane reboot

Fast-reboot is initiated by running /usr/bin/fast-reboot executable (bash script). This script must be run when a switch is stable. The fast-reboot script does following:

1. dump FDB and ARP entries from the ASIC db tables into the swss container
2. stop (-9) bgpd process to force BGP graceful restart
3. stop teamd process allowing teamd to send last update to its peers
4. stop docker service otherwise the filesystem of the docker containers will be corrupted
5. stop Broadcom drivers (for Broadcom platform) to avoid kernel panics
6. load a new kernel from the disk, set fast-reboot argument for the kernel and reboot into the new kernel

The data plane is still working

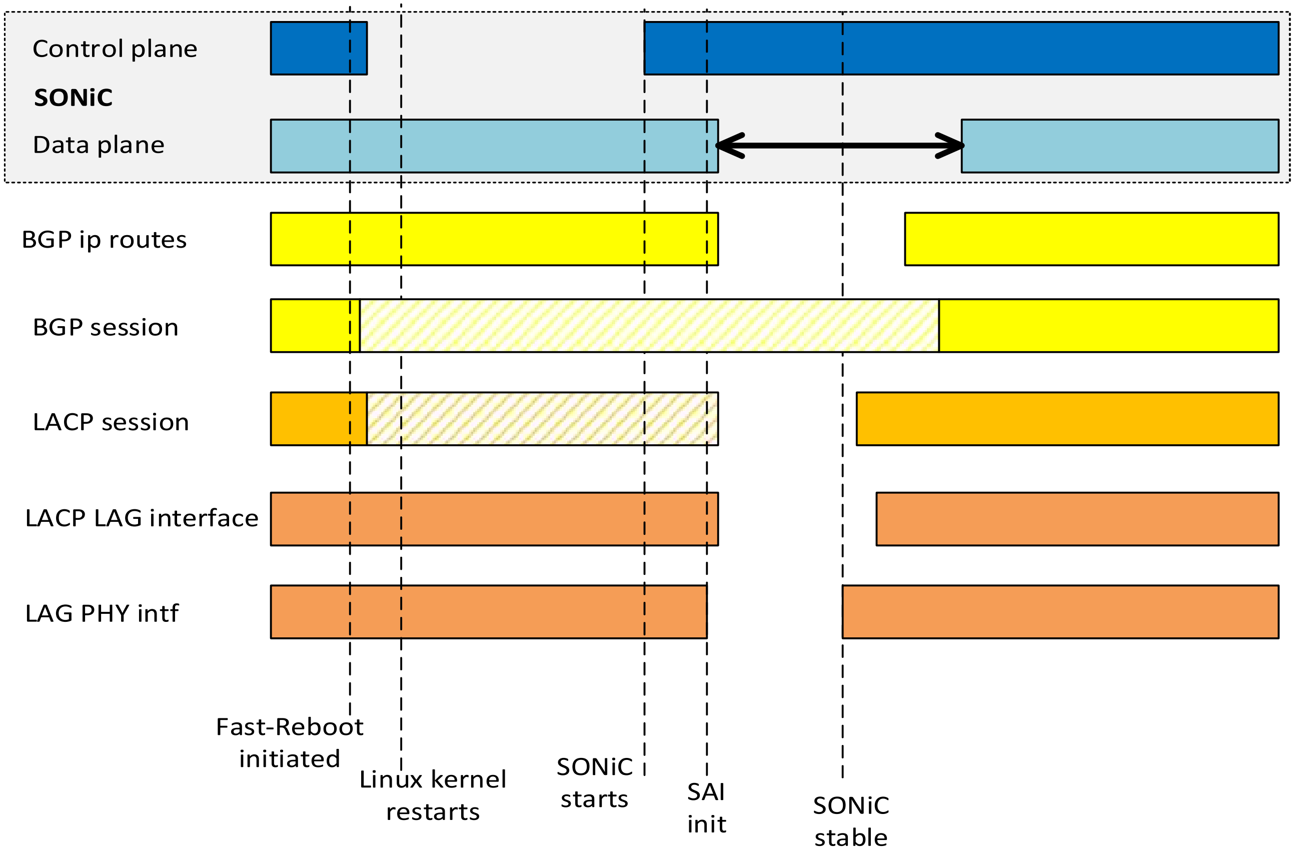
How it works. After control plane reboot

1. The SONiC loads in usual way after kexec (The data plane is still working)
2. syncd determines sonic is loaded after fast-reboot and initialize ASIC in fast-reboot mode (initialize ASIC only, not PHY part. The data plane is disrupted now
3. SONiC starts in an usual way, but swss loads FDB and ARP dumps which were saved before reboot. It allows us to save ~10 seconds for 500 hosts under vlan
4. After LAG member interfaces go up LACP restores LACP LAG interfaces in a second
5. After LACP LAG interfaces are restored Quagga BGP form new sessions and exchange BGP information in 1-6 seconds. After this our data plane is restored

How Fast-Reboot looks from an external host

1. BGP session is closed by SONiC device without the notification. BGP session is preserved in graceful restart mode, bgp routes are still active, because next-hop LAG interfaces are up
2. teamd sees that there’re no LACP updates from SONiC, but the member interfaces are still up, so teamd still waits for updates
3. LAG member interfaces go down, LAG interfaces goes down, BGP routes are removed from the routing table. Nothing is sent to SONiC device since then
4. after the LAG member interfaces go up, teamd starts forming LACP LAG very fast. When LACP LAG is formed, bgp makes active the previous bgp routes. From now SONiC devices restores its work
5. BGP sessions set up and bgp graceful restart mode ends
6. Everything works in a normal mode

Fast-Reboot timeline

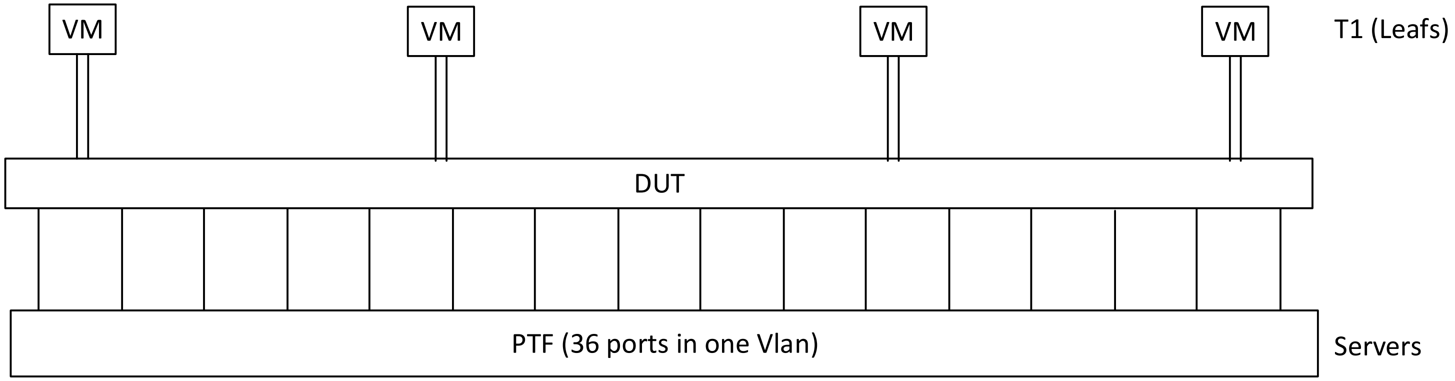


Fast-Reboot implementation

- /usr/bin/fast-reboot - fast-reboot
- /usr/bin/fast-reboot-dump.py - fast-reboot-dump.py
- orchagent/fdborch.cpp - sonic-swss/eed4a2
- quagga bgpd: enable graceful restart F-flag - sonic-quagga/940dc2
- syncd:/usr/bin/syncd_init_common.sh - syncd_init_common.sh
- swss:/usr/bin/swssconfig.sh - swssconfig.sh

Fast-Reboot test

- ptf test - fast-reboot.py
- ansible part of the ptf test - fast-reboot.yml
- linkstate propagator- linkstate



Fast-Reboot test. How to run?

- ./testbed-cli.sh start-vms server_3 ~/.password
- ./testbed-cli.sh add-topo vms3-t0-s6100 ~/.password
- ansible-playbook -i linkstate/testbed_inv.py -e target_host=vms3-t0-s6100 --vault-password-file=~/.password linkstate/up.yml
- ansible-playbook -i str -l str-s6100-acs-1 -t fast_reboot test_sonic.yml --vault-password-file=~/.password -e ptf_host=10.0.0.21 -e "vm_hosts=[10.0.0.200,10.0.0.201,10.0.0.202,10.0.0.203]"
- ansible-playbook -i linkstate/testbed_inv.py -e target_host=vms3-t0-s6100 --vault-password-file=~/.password linkstate/down.yml

Questions?