CHECK POINT™

YOU DESERVE THE BEST SECURITY

PERFORMANCE OPTIMIZATION PART 3 - COREXL

We will start soon!

Valeri (VAL) Loukine

Cyber Security Evangelist | Community Lead

CheckMates Live Virtual Series 2022

Check Point CHECKMATES

CHECK POINT™

YOU DESERVE THE BEST SECURITY

PERFORMANCE OPTIMIZATION PART 3 - COREXL

PhoneBoy

Cyber Security Evangelist | Community Lead

CheckMates Live Virtual Series 3

Check Point CHECKMATES

Housekeeping Rules

- The session is being recorded, all participants will get a link
- Materials will be posted on CheckMates
- Use Q&A panel for questions, not Chat
- There is an option to raise a hand
- All questions are welcome, speak your mind

community.checkpoint.com

Full list of Performance Series

- Part 1 – Introduction
- Part 2 – SecureXL
- **Part 3 – CoreXL**
- Part 4 – Clustering and Hyperscale
- Special– Diagnostics How To

Agenda

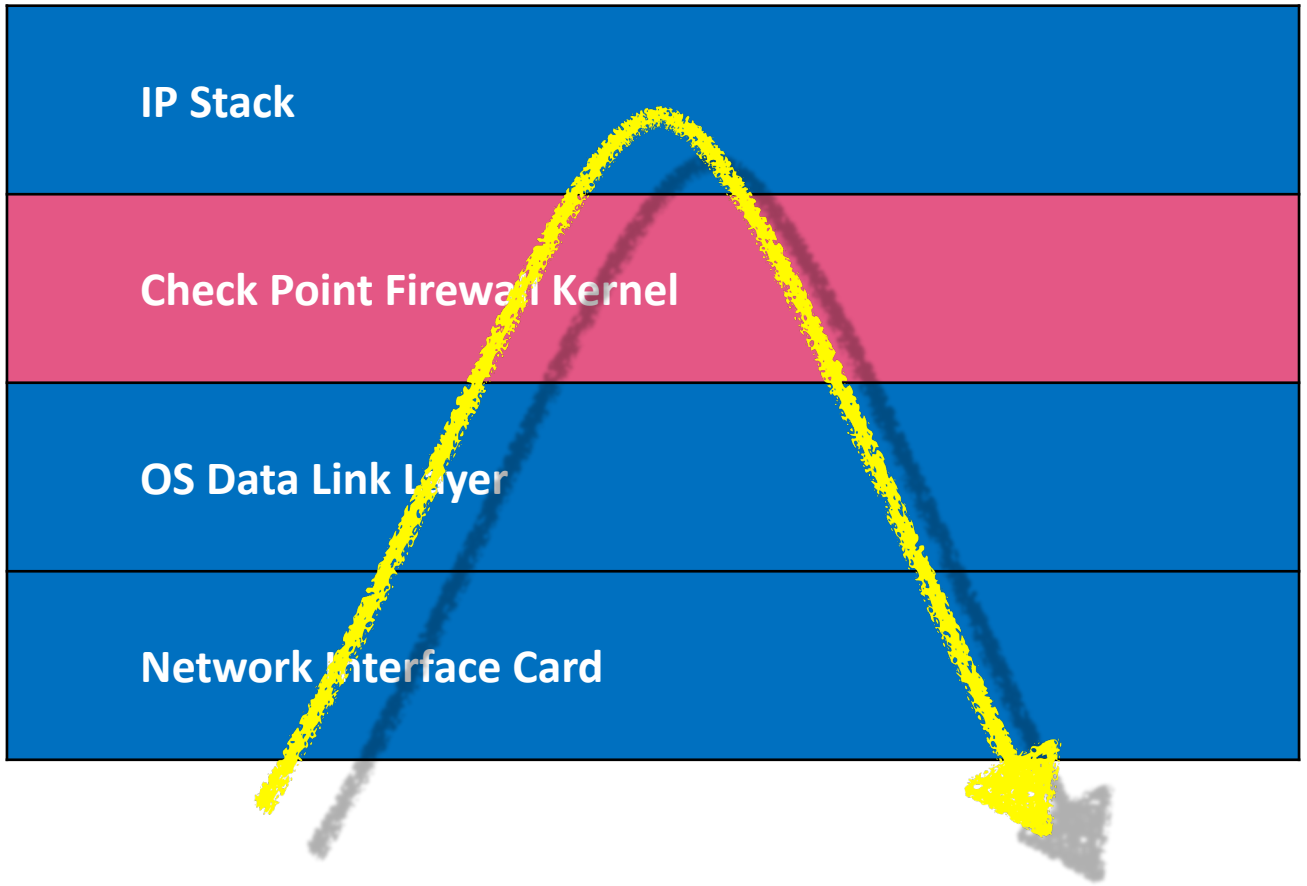
- Quick re-cap
- CoreXL
  - Terminology
  - Architecture
  - Optimization
  - Tools

Performance of Security Gateway depends on

- CPU - utilization / saturation / errors
- Memory - utilization / saturation / errors
- Network Interfaces - utilization / saturation / errors
- Storage device I/O, capacity, controller - utilization / saturation / errors
- Throughput (packet rate * packet size)

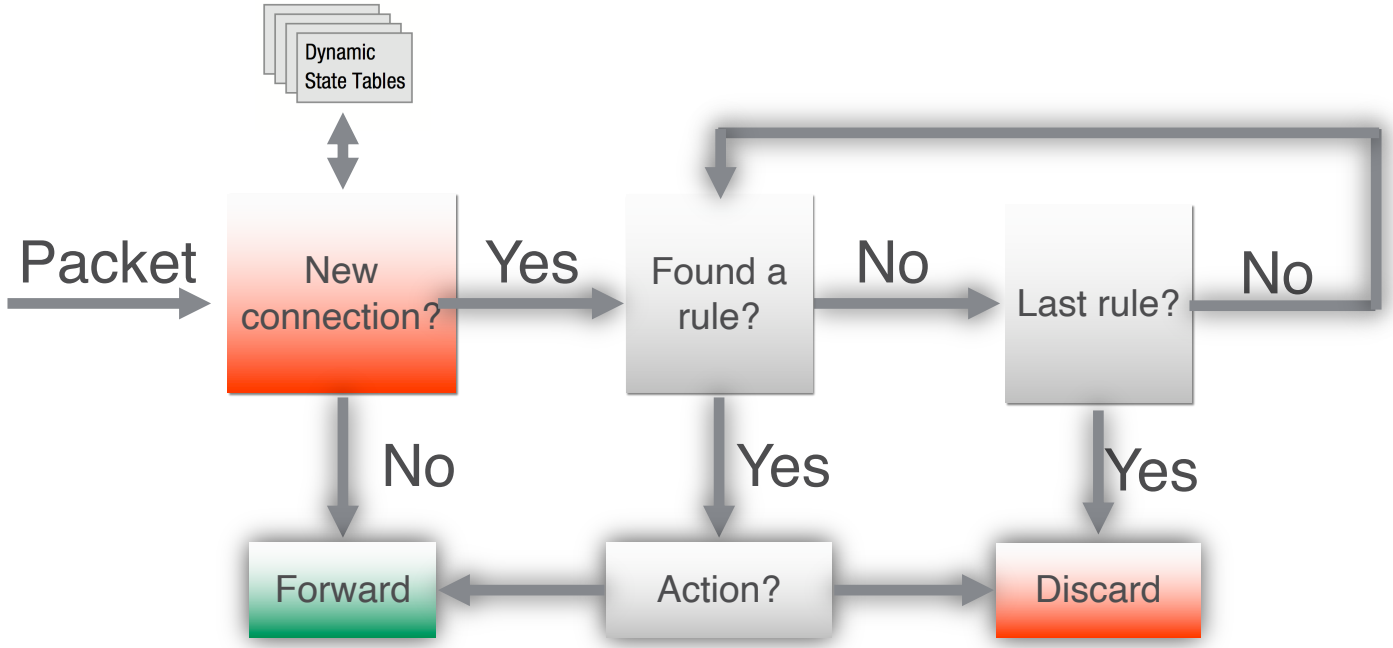
sk98348

Stateful Inspection



Stateful Inspection

- The best network security idea in the last 25 years
- Come with an eventual performance drag



Stateful Inspection as a performance bottleneck

- Confined in kernel space (originally)
- Cannot pass through before inspection
- Matching first packet takes lots of effort
- Dropping takes even more effort

Network Performance Terminology

- Throughput
- Packet per second rate
- Latency
- Number of concurrent connections
- New connection rate
- Jitter and Retransmissions

COVERED IN PART 2

SECUREXL

Why SecureXL?

Challenge:

- With growing packet rate and amount of connections, a single CPU FW instance is overwhelmed with security tasks

Approach:

- Offloading some of simple security decision to an additional computation unit (CPU on a card, another core, etc)

COREXL

Check Point
CHECKMATES

Why CoreXL

Challenge:

- With growing packet rate and amount of connections, both a single CPU FW instance and SecureXL are overwhelmed with security tasks

Approach:

- On top of SecureXL offloads, use more than one CPU cores for FW operations and deeper inspection of the traffic

CoreXL – the concept

sk98737

- The firewall kernel is replicated on multiple cores

- Each instance is a complete and independent FW kernel
- Instances run concurrently

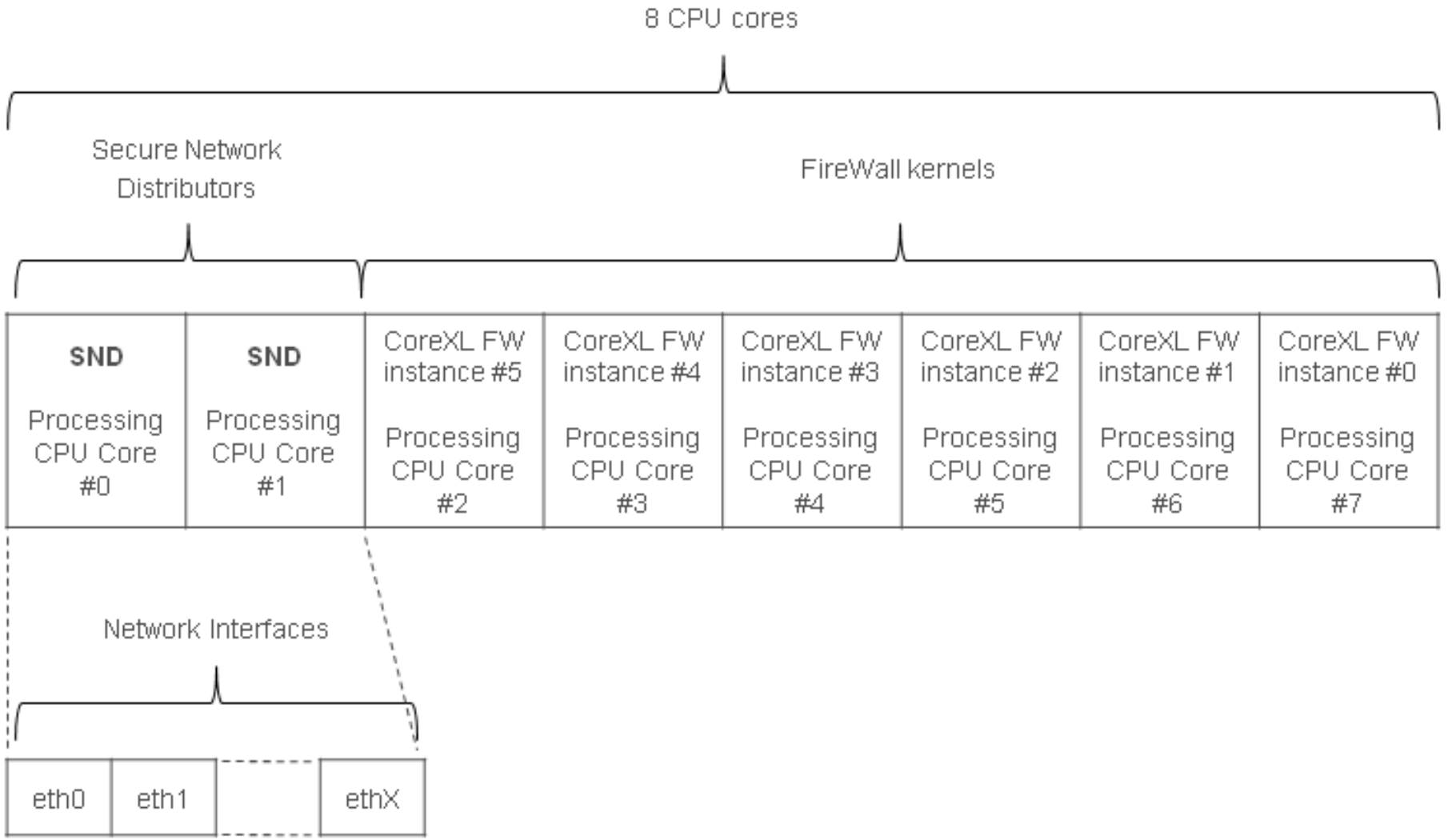
CoreXL - Terminology

- Secure Network Distributor (SND)
- FW Instance
- Affinity
- Paths
  - Accelerated
  - Medium (PXL)
  - Firewall (F2F)
- Passive Streaming Library (PSL)

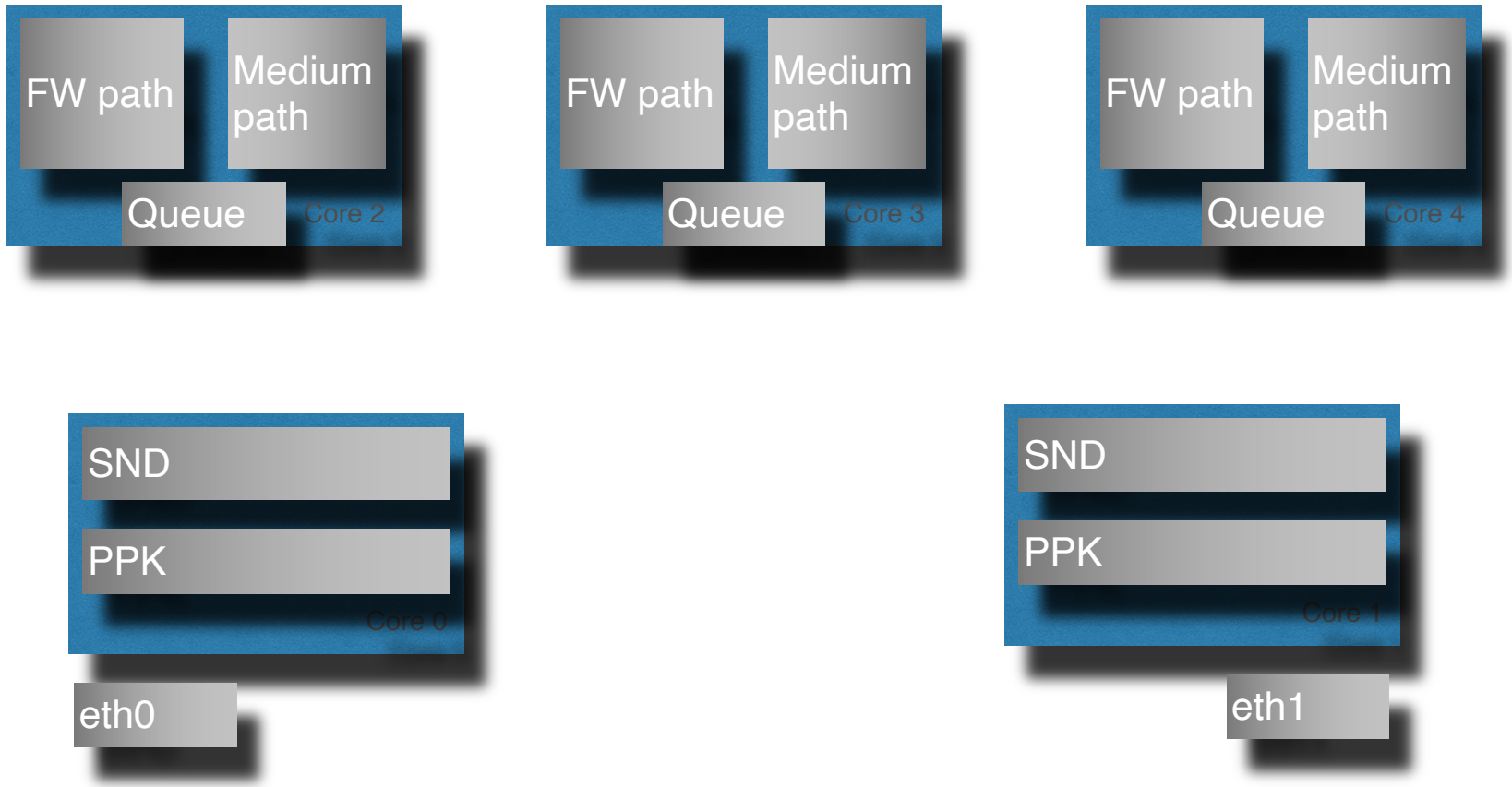
Distributor

- Secure Network Distributor (SND)
- It component distributes the packets between the instances
- Maintains perfect stickiness of a connection
- Also used for acceleration

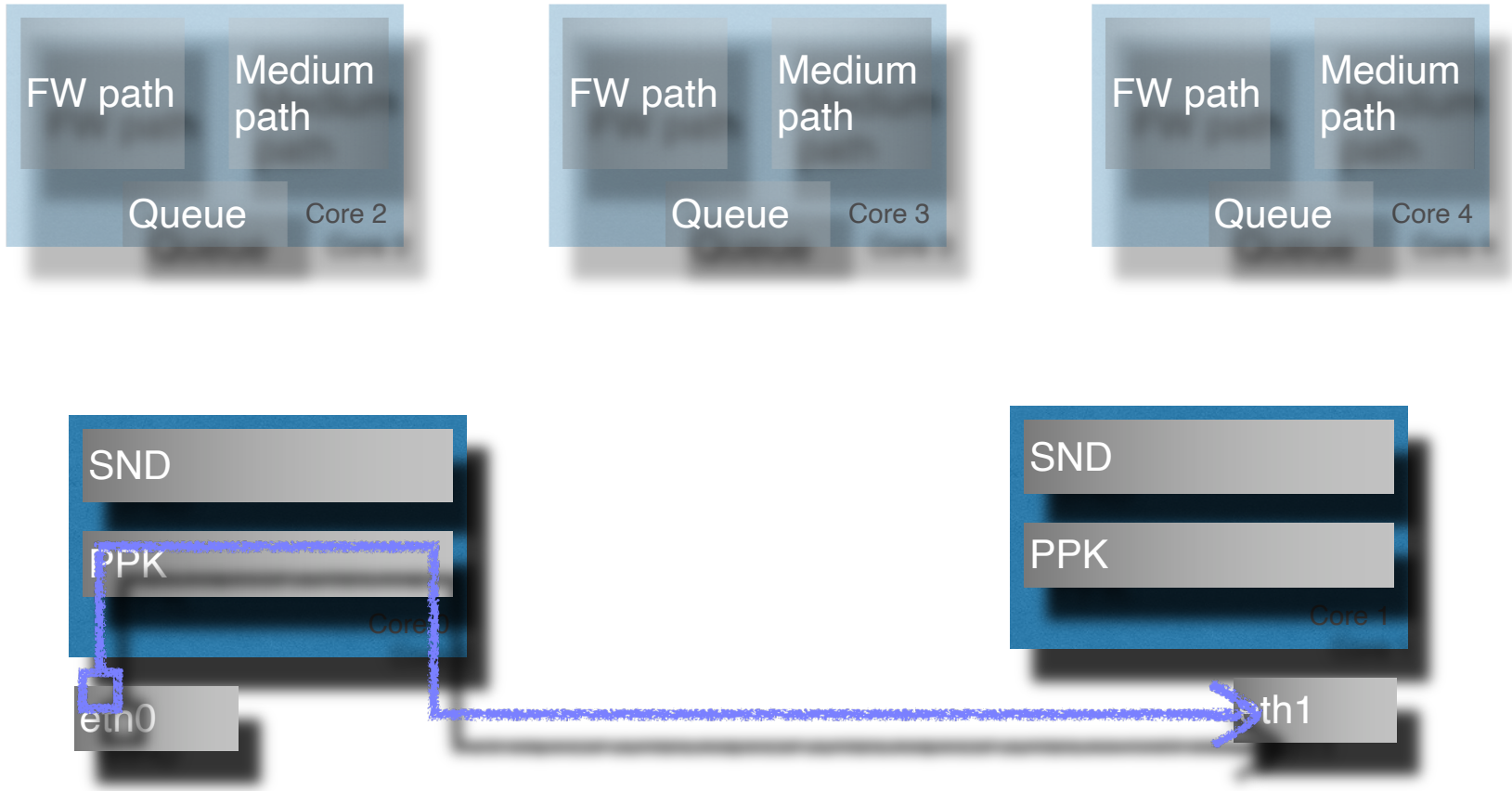
Multiple Cores



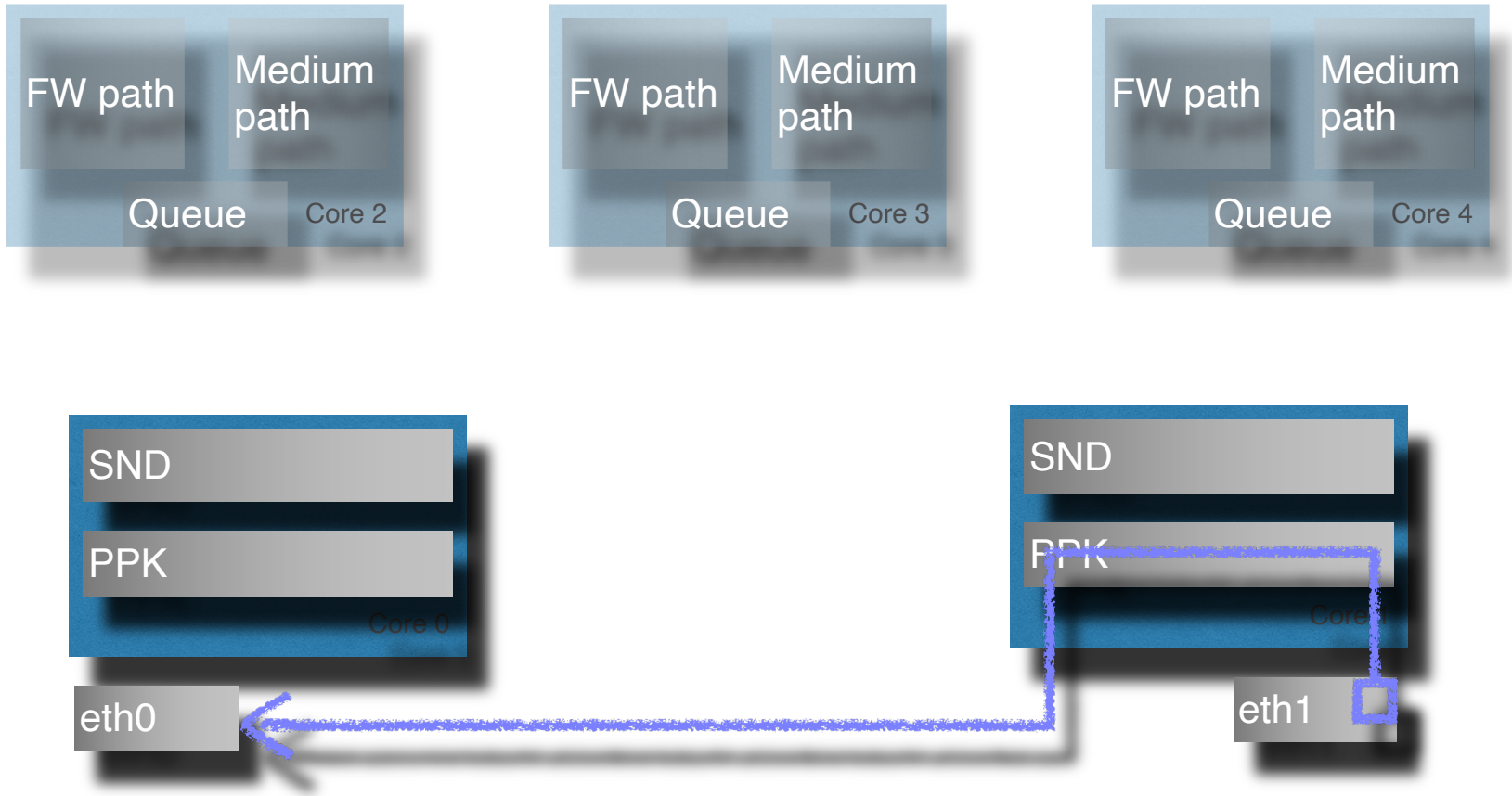
How does it work



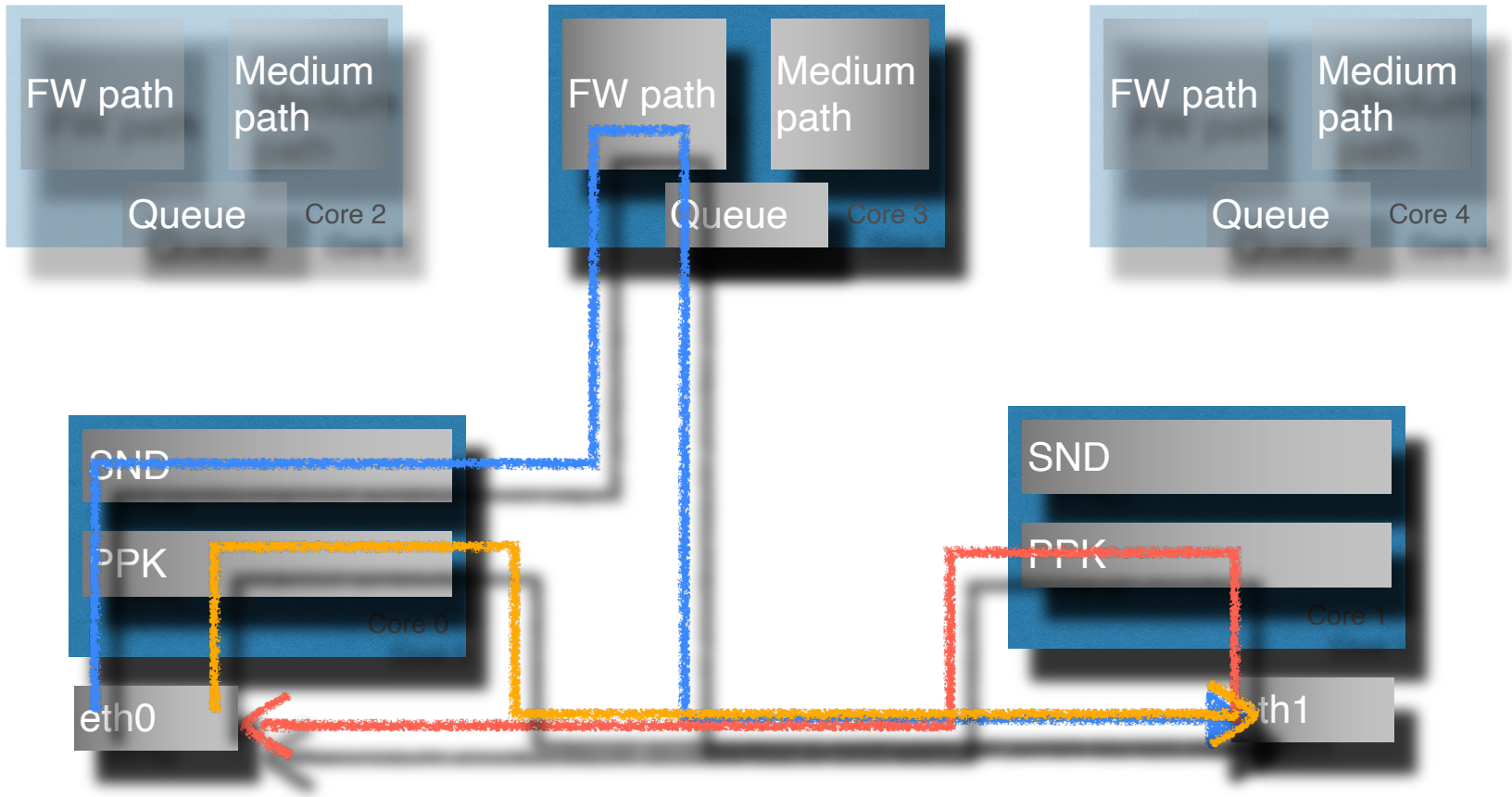
Accelerated - Syn



Accelerated - SynAck



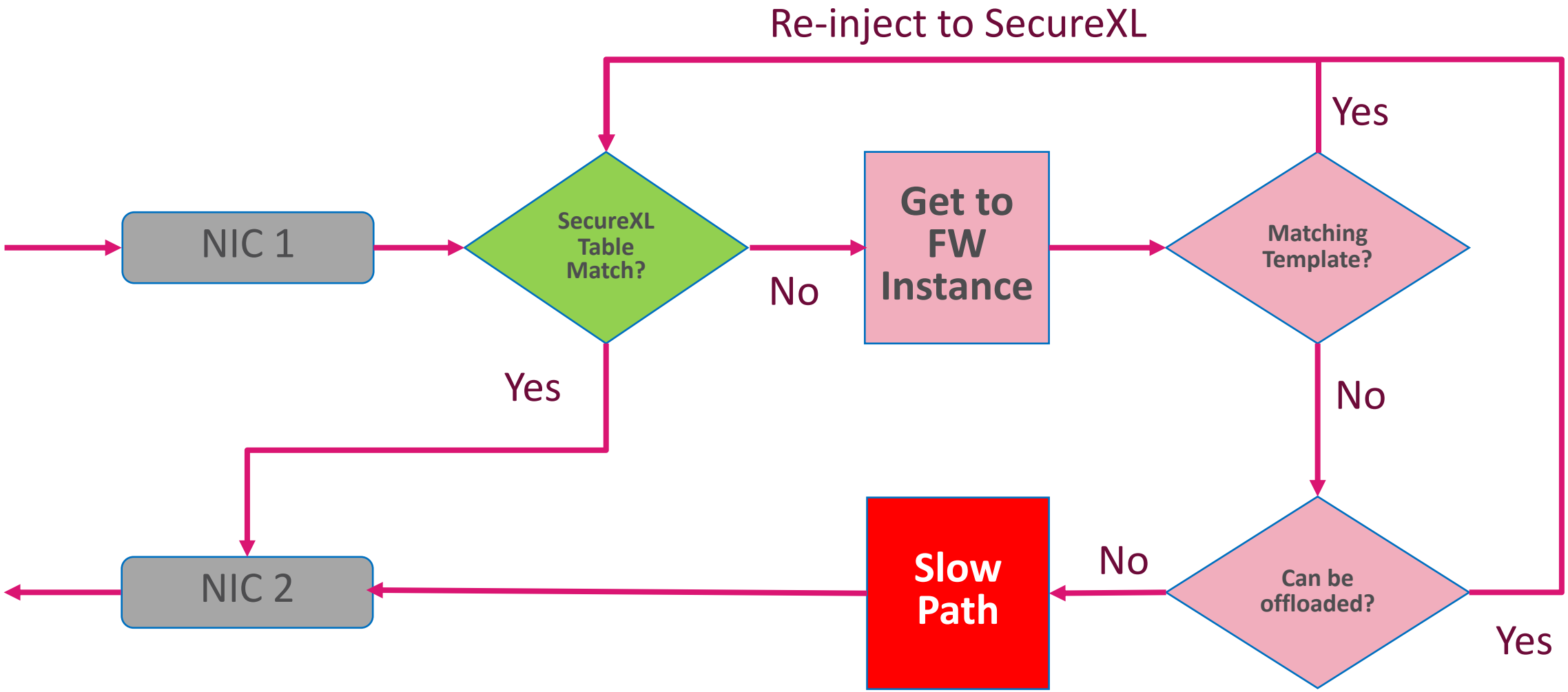
No template



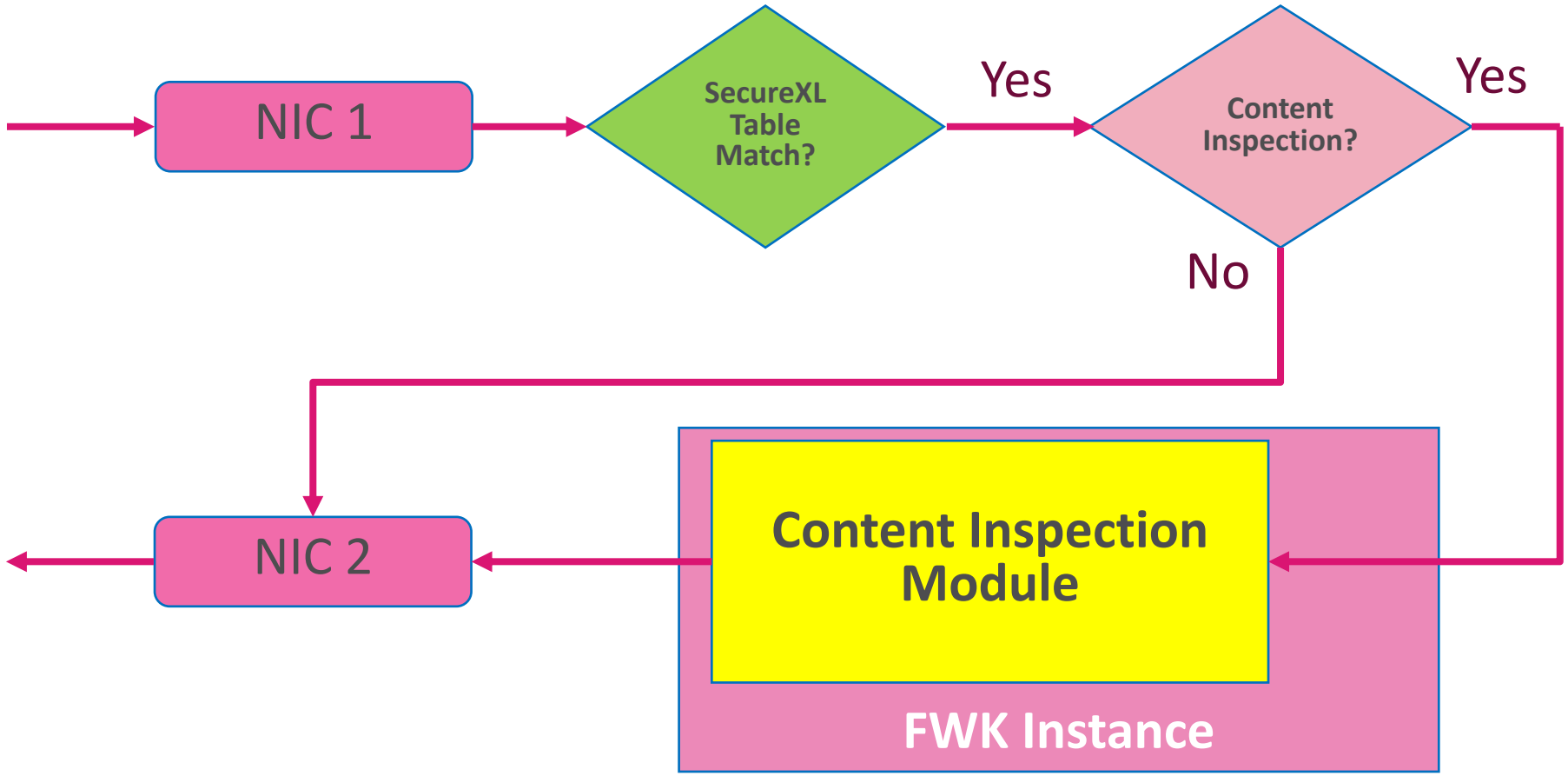
SecureXL & CoreXL – Paths

- Firewall Path
  - Connection cannot be accelerated
  - All packets within the connections are handled by Firewall Instance
- Accelerated Path
  - All packets are handled by SecureXL
- Medium Path
  - Packet flow is partially handled by SecureXL
  - Data flow is run through Firewall Instances Instances for content inspection
  - Only available with CoreXL

SXL - Simplified Flow R80.20 and up sk153832



Medium Path R80.20 and up (Simplified)



CORE ASSIGNMENT

Check Point
CHECKMATES

Default number of FW IPv4 Instances

- Subject of actual licensed cores

Number of CPU cores	Default number of CoreXL IPv4 FW instances	Default number SNDs
4	3	1
6 to 20	[Number of CPU cores] - 2	2
More than 20	[Number of CPU cores] - 4 Note: However, no more than 40 in R80.10 and above.	4

Maximum number of FW IPv4 Instances

- Subject of actual licensed cores
- Requires at least 6 GB of RAM

Version	Number of Cores
R80.20 and above	40
R80.10 32 bit	16
R80.10 64 bit	40
R77.30 32 bit	16
R77.30 64 bit	32

Controlling CoreXL

- cpconfig CoreXL menu
- `# fw ctl multik start/stop`

- `# cat /etc/fw.boot/boot.conf`

```
CTL_IPFORWARDING              1
DEFAULT_FILTER_PATH   /etc/fw.boot/default.bin
KERN_INSTANCE_NUM             3
COREXL_INSTALLED              1
KERN6_INSTANCE_NUM            2
IPV6_INSTALLED          0
```

FW Kernel Tables Notes

- Kernel tables are required for Stateful Inspection
- Each FW Instance has its own kernel tables
- Syntax:

```
# fw [-i FW_INSTANCE_ID] tab [-t TABLE] [-s] [-f [-r]] [-u | -m
MaxValues]
```

- For example,

```
Expert@HostName]# fw -i 3 tab -t connections -s
```

LIMITATIONS

Check Point
CHECKMATES

CoreXL Limitations - Historical overview

- Check Point QoS (Quality of Service)
- 'Traffic View' in SmartView Monitor
- Route-based VPN
- IP Pool NAT
- IPv6
- Firewall-1 GX
- Overlapping NAT
- SMTP Resource
- VPN Traditional Mode
- Virtual Tunnel Interface (VTI)
- 6in4 traffic

CoreXL Limitations - sk61701

R80.10 and above, remaining limitations are:

- Overlapping NAT
- 6in4 traffic - always processed on *fw_worker_0*

ClusterXL & CoreXL

- Amount of FW instances should be identical on both cluster members
- If not, one with less FW instances will be Active
- The second member will be in Ready state
- Same if one of them has CoreXL disabled

DID WE FIX IT YET?

Check Point
CHECKMATES

More CoreXL challenges

- Static Distribution if traffic between instances
  - **Dynamic Dispatcher**
- Some connections are heavier than others
  - **Priority Queues**
- Not enough CPUs in the kernel mode
  - **User Mode FW**
- Static split of SNDs and FWKs may not be ideal
  - **Dynamic Balancing**

DYNAMIC DISPATCHER

Dynamic Dispatcher - sk105261

- FWK static distribution based on:
  - Source IP addresses
  - Destination IP addresses
  - IP 'Protocol' type
- This is problematic with certain types of traffic
- Some cores could be busier than others
- Need something smarter

Dynamic Dispatcher - sk105261

- R77.30 and up – Dynamic Dispatcher
- Automatically enabled in R80.10 and up
- Allows changing connection distribution based on CPU utilization
- Supports VSX from R80.20 and up

Dynamic Dispatcher - controls

- Check the mode:

```
# fw ctl multik dynamic_dispatching get_mode
```

```
[Expert@R80.10:0]# fw ctl multik dynamic_dispatching get_mode
Current mode is On
[Expert@R80.10:0]#
```

- Set ON/OFF

```
fw ctl multik dynamic_dispatching on/off
```

- Requires reboot

HEAVY CONNECTIONS AND PRIORITY QUEUES

Check Point
CHECKMATES

Heavy Connection

- A single connection
  - consumes very large amounts of bandwidth
  - takes lots of CPU time on the assigned FWK core
  - Goes either Firewall Path or Medium Path
  - Usual suspects: backups, updates, virtualization sync, etc,
- A.K.A. “elephant flow”

- Other connection on the same core might have degraded performance

Identifying Heavy Connections

- `top`
- `cpview`
- `fw ctl multik print_heavy_conn` (Kernel Mode only, R80.20 JHF 47 and up):
  - Specific instance CPU is over 60%
  - Suspected connection lasts more than 10 seconds
  - Suspected connection utilizes more than 50% of the total work the instance does
- I.e., for CPU running 60%, connection CPU utilization must be > 30%

Heavy Connection - cpview

```
CPVIEW.CPU.Top-Connections.Instances0-2.Instances1
------------------------------------------------------------------------------
Overview SysInfo Network CPU I/O Software-blades Hardware-Health Advanced
------------------------------------------------------------------------------
Overview Top-Protocols Top-Connections
------------------------------------------------------------------------------
Instances0-2
------------------------------------------------------------------------------
Instances0 Instances1 Instances2
------------------------------------------------------------------------------
CPU Utilization

Average connection CPU utilization          0.00%

                          Connections     CPU utilization
Top connections                     1              92.00%
Other connections               49927               8.00%
Total connections               49928             100.00%
------------------------------------------------------------------------------
Top Connections

Connection                                        Protocol    % out of CPU
192.168.102.181:49927 -> 209.167.231.17:443       TCP:https         92.00%
```

Mitigation options

- Priority Queues - sk105762

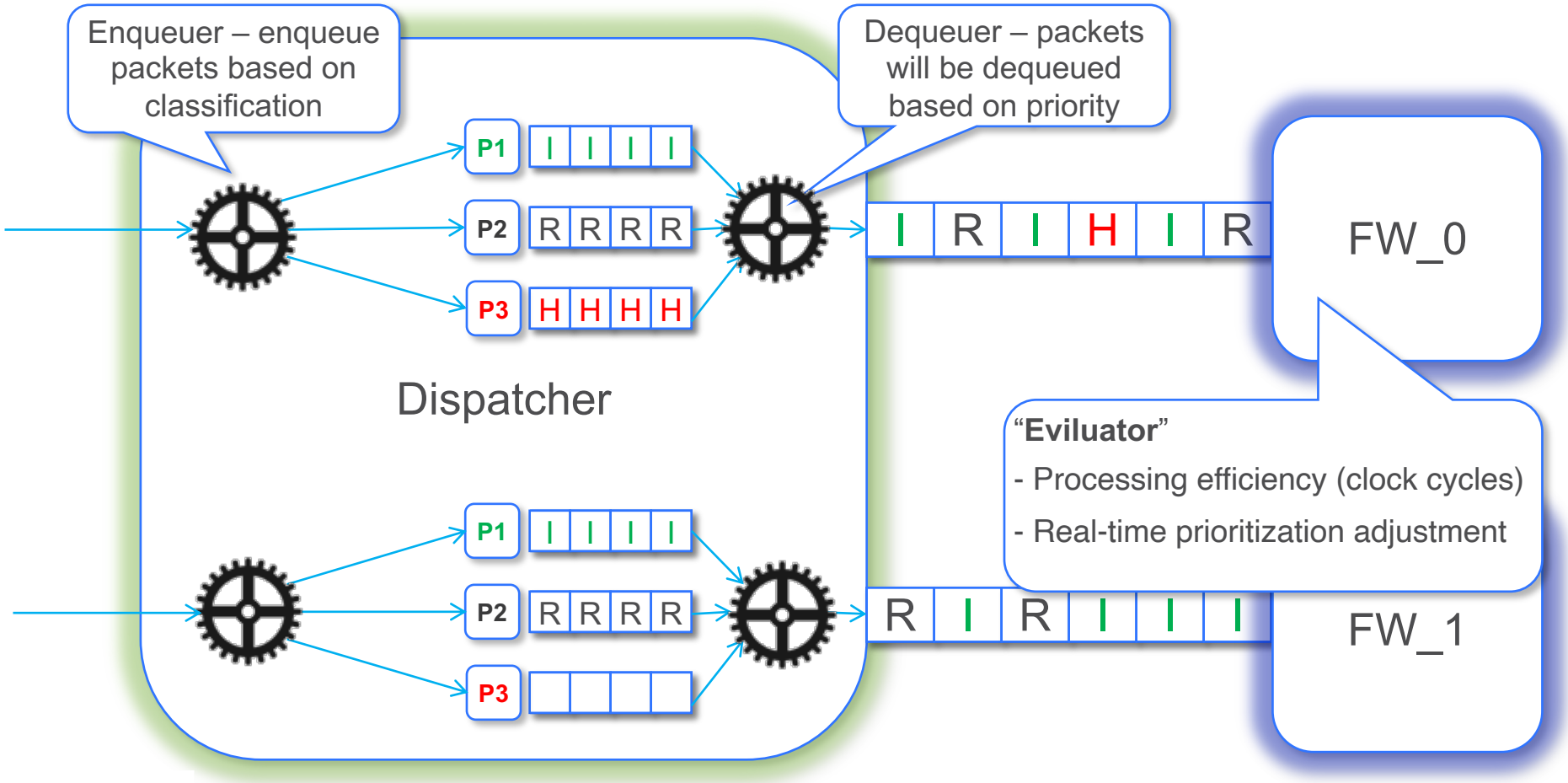
- Avoiding Medium Path (fw fast_accel) sk156672
- QoS Policy
  - R80.20 and above is SXL friendly

PRIORITY QUEUES

Priority Queues – Queue Types

Priority	Name	Type	Can Migrate?
0	Routing	DHCP, VRRP, OSPF, BGP, IGMP, PIM, ...	No
1	Control	GUI / SSH / ports 18xxx / Mgmt service	Yes
2	Cluster sync	Local / Full Sync	No
3	High Priority	User defined	Yes
4	Light Data Queue	Light connections	Yes
5	Default Data Queue	Medium weight connection, or New connection	Yes
6	Log Notification	Log and Drop notifications	No
7	Heavy Data Queue	Heavy connections	Yes

Firewall Priority Queues (sk105762)



H Heavy

R Regular

I Important

Priority Queues – Modes

- `# fw ctl multik prioq 1`
- `# reboot`

Mode	Name	Notes
0	Off	Firewall Priority Queues are completely disabled
1	Evaluator-only	Allows monitoring of Heavy Connections (that consume the most CPU resources) in CPView
2	On	Firewall Priority Queues are fully enabled

Priority Queues – Details

- For Kernel Mode only!
  - When in mode 2, kicks in with CPU 100%,
  - disengages when below 100%
- R80.40 – 2
- R80.30 – 1
- Below R80.30 – 0 (inactive)

- In User Mode, only static config from $FWDIR/conf/prioq.conf
  - See sk105762

Priority Queues

- If Firewall Worker CPU utilization hits 100% and Priority Queues is set to mode 2 (On), the default FIFO-based handling of that instance's packet queue is suspended.

- The priority of packets belonging to identified heavy connections will be lowered to the absolute minimum, allowing packets for all other connections (including delay-sensitive "mice") to "jump the queue" in front of the elephant connection(s) and be processed first.

Priority Queues (cont.)

- Once that FW Worker's utilization is <100%, FIFO-based handling resumes.
- In R80.40 Priority Queues are fully enabled
- USFW Enabled: Only static Priority Queue preferences for elephant flows with known attributes can be defined in the prioq.conf file, see sk105762.

COVERED IN THE PART 2

FAST_ACCEL

Check Point
CHECKMATES

Special Case - SecureXL Fast Accelerator

- SecureXL Fast Accelerator (fw fast_accel) sk156672
  - Deep packet inspection bypass for trusted assets
  - R80.40
  - R80.30 Take 107
  - R80.20 Take 103
- Bypass deep packet inspection for trusted
- Fast Pass instead of Medium Pass

SecureXL Fast Accelerator - controls

```
fw ctl fast_accel <option>
```

- add — Add a connection
- delete — Delete a connection
- enable — Set feature state to on
- disable — Set feature state to off
- show_table — Display the rules configured by the user
- show_state — Display the current feature state
- reset_stats — Reset the statistics collected by the feature
- --help/-h — Display help message

- `fw ctl fast_accel add 1.1.1.1 2.2.2.0/24 80 6`
- `fw ctl fast_accel delete 1.1.1.1 2.2.2.0/24 80 6`

FW INSTANCES IN USER MODE

Check Point
CHECKMATES

Kernel-mode Limitation

- Kernel-mode maximum number of running FW instances is limited to 40 due to Linux kernel memory limitation.
- CoreXL architecture needs to load a large driver (~40MB) dozens of times (according to the CPU number and up to 40 times).
- New platforms that contains more than 40 cores (and with HT even more) are not fully utilized.

New USFW Mode

- Using USFW is the only way to fully utilize the amount of cores of our newest appliances.
- USFW Code is based on VSX infrastructure – User-space process (fwk) that runs the FW code in user-space instead of kernel-mode.
- USFW can utilize up to 64 FW instances (compared to 40 in kernel-mode).
- ‘Look and feel’ is the same as kernel-mode FW.

Supported Releases & Platforms

- USFW was already part of R80.20 and improved performance by ~20% compared to kernel-mode on high-end appliances.
- It is enabled by default in R80.30 on all appliances with 40+ CPUs (23900 and future).
- It is the only available mode in the new Quantum release for Quantum appliances (and 23900).

COVERED IN THE PART 2

MULTI-QUEUE

Check Point
CHECKMATES

Static Core assignment challenge

- Changing queues / SNDs requires reboot
- With changing traffic partner SND/FWK balance may need to be changed too

R80.40 comes with Dynamic Workflows feature

- Before we discuss it, let recall Multi-Queue

Why Multi-Queue

Challenge:

- With growing frames rate on a network interface, a single CPU core is overwhelmed with interrupts.

Approach:

- Use more than one CPU core for receiving and processing frames.

What is Multi-Queue

- Also know as Receive-Side Scaling (RSS)
- We can configure more than one traffic queue for each network interface
- For each interface, more than one CPU core is used for acceleration
- Multi-Queue is applicable only if SecureXL is enabled

- Reference: Performance Tuning Guide

Maximum number of RX Queues per driver

Driver	Max Speed (Gbps)	Maximal Number of RX Queues
igb	1	2-16 (depends on the interface)
ixgbe	10	16
i40e	40	64
i40evf	40	4
mlx5_core	40	60
ena	20	Configured automatically
virtio_net	10	Configured automatically
vmxnet3	10	Configured automatically

COREXL
DYNAMIC BALANCING

Check Point
CHECKMATES

What is CoreXL Dynamic Balancing?

- TechTalk about this feature
- sk164155

- Changing CoreXL split between FW workers and SND on the fly based on CPU utilization
- Requires on OS 3.10 (USFW/Kernel);
- Check Point appliances with 8 cores or more

What's the motivation behind CoreXL Dynamic Balancing?

- The two main ideas behind Dynamic Balancing are
  - Usability
  - Simplicity

- Usability by dynamically changing CoreXL split (SNDs/FW_workers balance) without reboot
- Simplicity by improving Out Of The Box performance experience for specific customer scenarios

Additional SND is needed

- If the system detects that SecureXL cores are loaded and more connection handling capacity is needed, we add more SecureXL cores by reducing the number of CoreXL cores
  - We can't just disable CoreXL instances because that would cause dropped connections
  - Instead we instruct CoreXL instances to finish processing sessions already assigned to them and not to take on new connections
  - Once the CoreXL instance finishes processing all existing sessions we stop CoreXL processes on the core and start SecureXL on it
    - Remember, the queue is already assigned to the CPU just traffic was not sent there yet

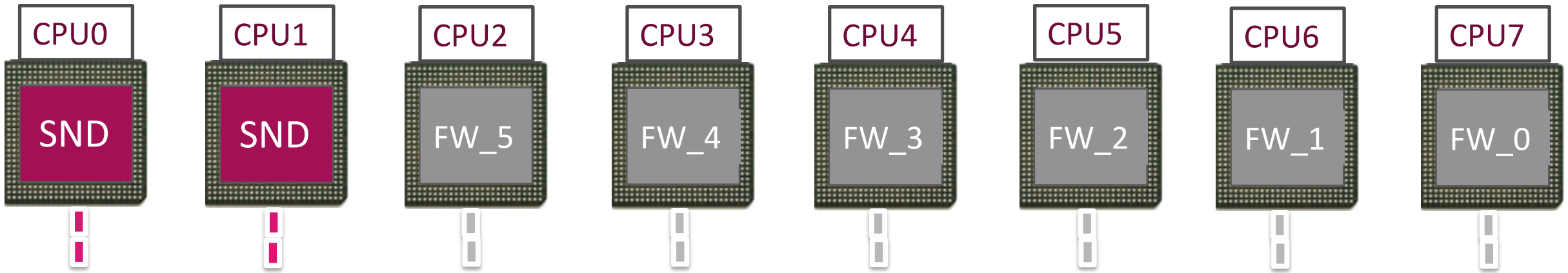
Additional Instance is needed

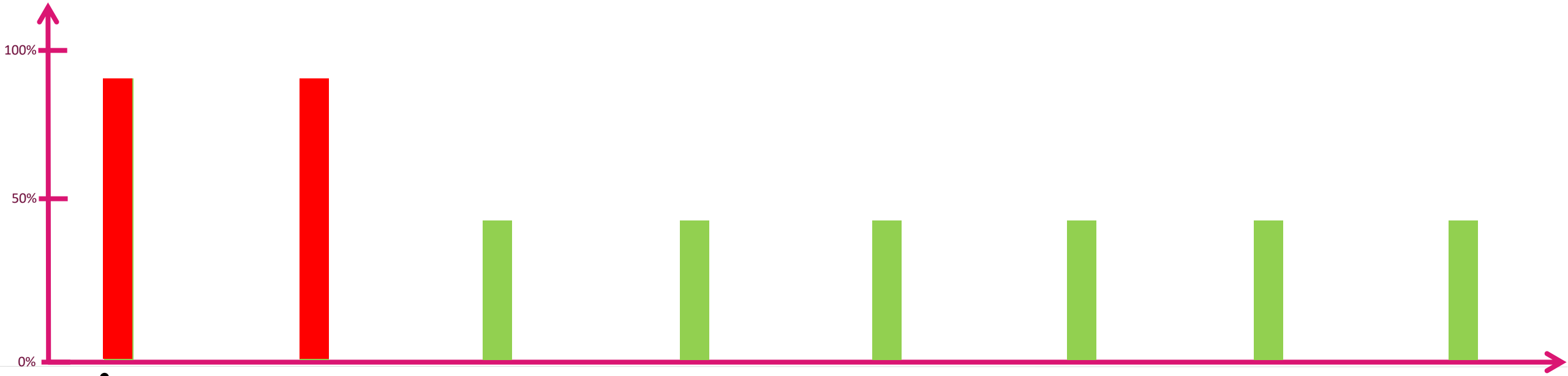
- If the system detects that CoreXL instances are loaded and more inspection capacity is needed, we add more CoreXL cores by reducing the number of SecureXL cores
  - Exactly the opposite is happening as on the previous slide
- Important!!!
  - Each Check Point Security Gateway Appliances has a default SecureXL/CoreXL split allocation
    - e.g. appliance with 8 cores has: 2x SNDs, 6x FW Instances
  - We can't start more CoreXL instances than this default number
  - Don't tinker with cpconfig and the number of CoreXL instances!
    - You change the original number of instances and that disables this feature permanently

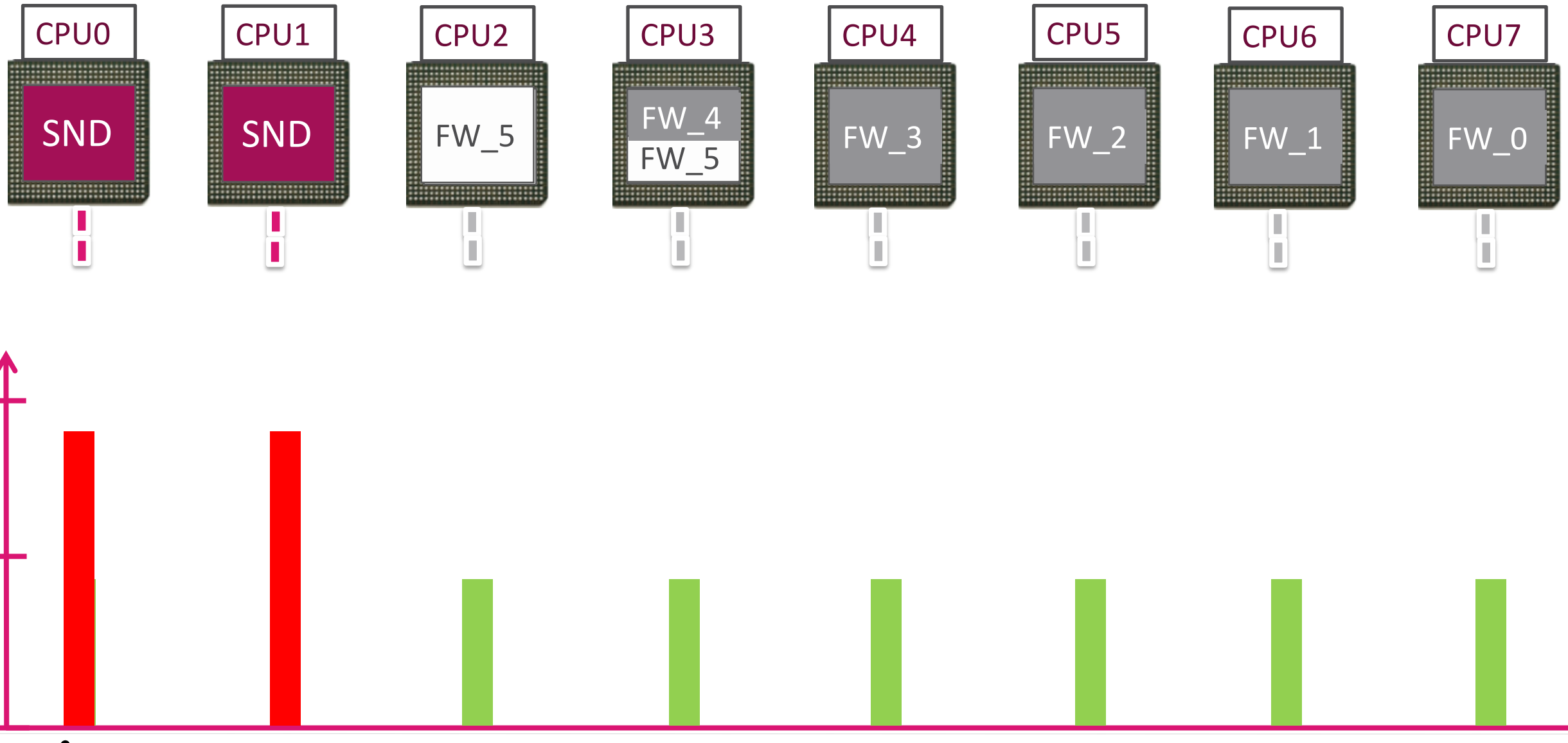
When the change is done?

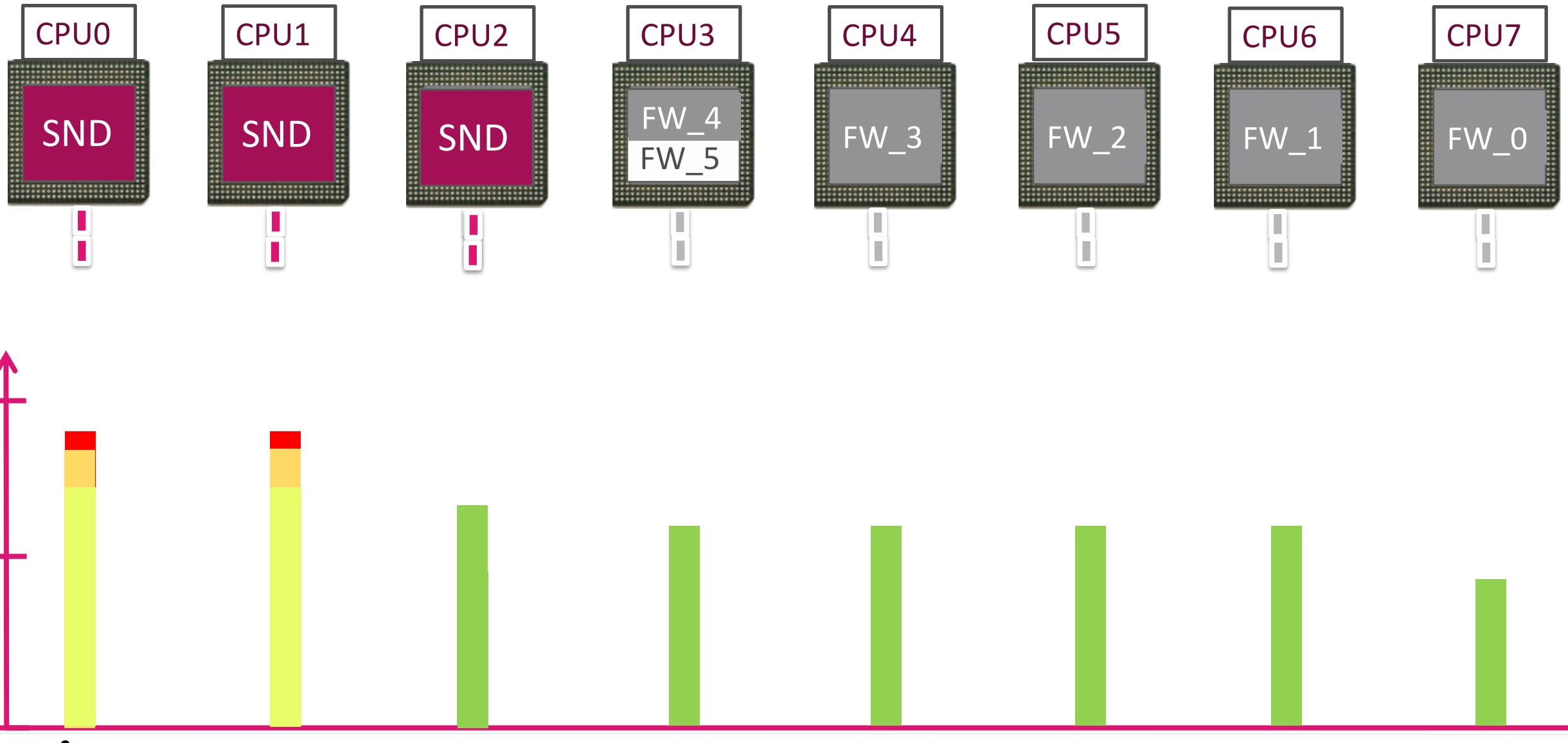
- The system continuously monitors the load of CPUs
- We then estimate what the utilization of the system would be after a change and compare that to the current utilization
- If the estimated difference would improve things by 10% or better, then change occurs
  - 10% is to avoid flapping
  - Special situations can override this, see next slide

Flow

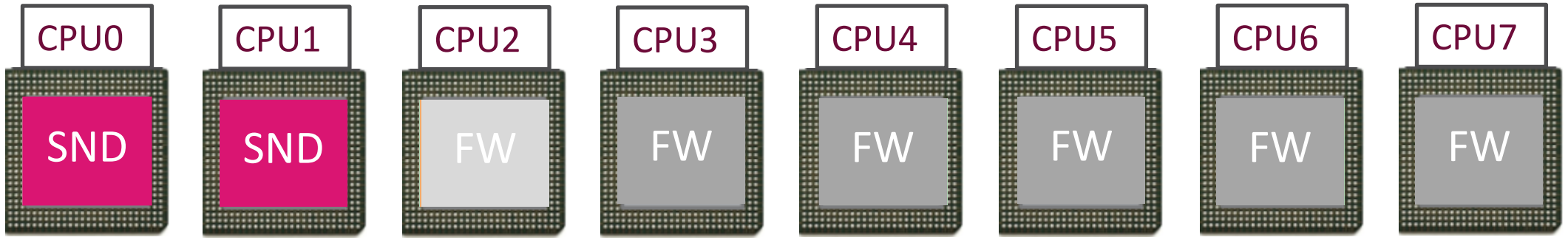


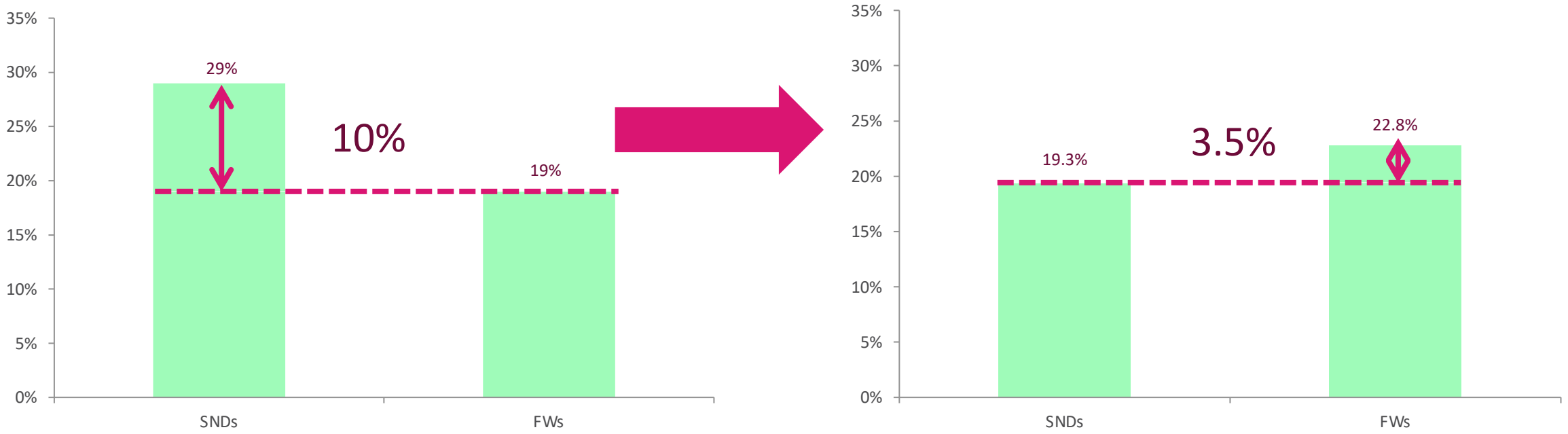






Dynamically controlling the split





Special situations

- If CoreXL load reaches 40%, but SecureXL load is already at 60% we don’t perform a change since SecureXL would be at 80% after the change
- If a Heavy Connection (an Elephant Flow) is the reason for a CoreXL core being loaded, we stop a different core with lower load.
- If a Heavy Connection is the reason for high SecureXL load we do not reallocate CPUs since it wouldn't help to add more SecureXL
- IPv6:
  - We prefer not to turn off IPv6 instances in case there are few IPv6 instances compared to IPv4 instances – each would be a significant reduction
  - In case they have similar numbers this is not an issue we can turn those off as well

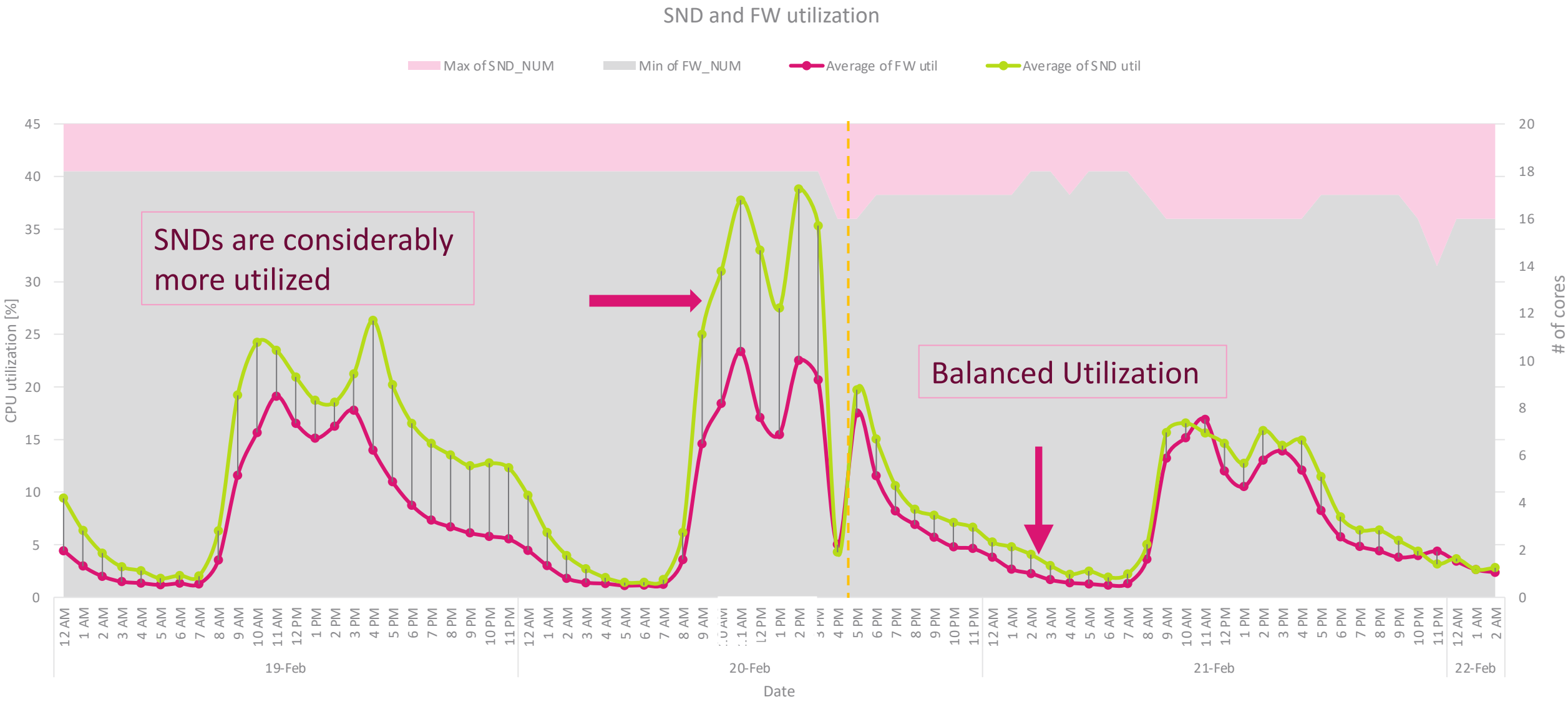
Additional things to consider

- Only Check Point Security Gateway appliances are supported
  - Only those appliances which have at least 8 CPUs (4 cores with HT)
- USFW is supported
- Do not touch cpconfig CoreXL settings!!!
- Limitations:
  - No VSX support, because fixed instance number is a business case
  - Load-Sharing cluster is not supported
  - If you specify a setting in cpconfig CoreXL changes, you will disable CoreXL Dynamic Balancing permanently

Let’s see how it looks

```
CPVIEW.SysInfo
Overview SysInfo Network CPU I/O Software-blades Hardware-Health Advanced
Configuration Information:

Platform            Gaia 64Bit
Configuration       Check Point Security Gateway
CoreXL Status       on
CoreXL instances    28
SMT Status          Unknown
DS Status           On
General information:

System uptime                0 days, 04:19:13
Last policy install time     04Aug2019 13:33:34
Last policy name             Standard
Version Information:

fw1_wrapper package version   R80.30

              Branch Name                    Build Number
FW User Mode  gogo_heat_188_DS_Perf1_main    993000013
FW Kernel     gogo_heat_188_DS_Perf1_main    993000006
Accel Module  gogo_heat_188_main             993000034
Adpdrv Module gogo_heat_188_main             993000005
SIM Module    gogo_heat_188_main             993000033
Hardware Information:

HW Model            Check Point 15600
SAM CPLD Version    N/A
SAM Total Memory    N/A

Devices             Type      Slot      Status
Expansion-3         4x10GbE   3         Connected
```

Customer: In Practice



HYPERFLOW

USING DYNAMIC LOAD BALANCING TO SECURE ELEPHANT FLOWS

Check Point
CHECKMATES

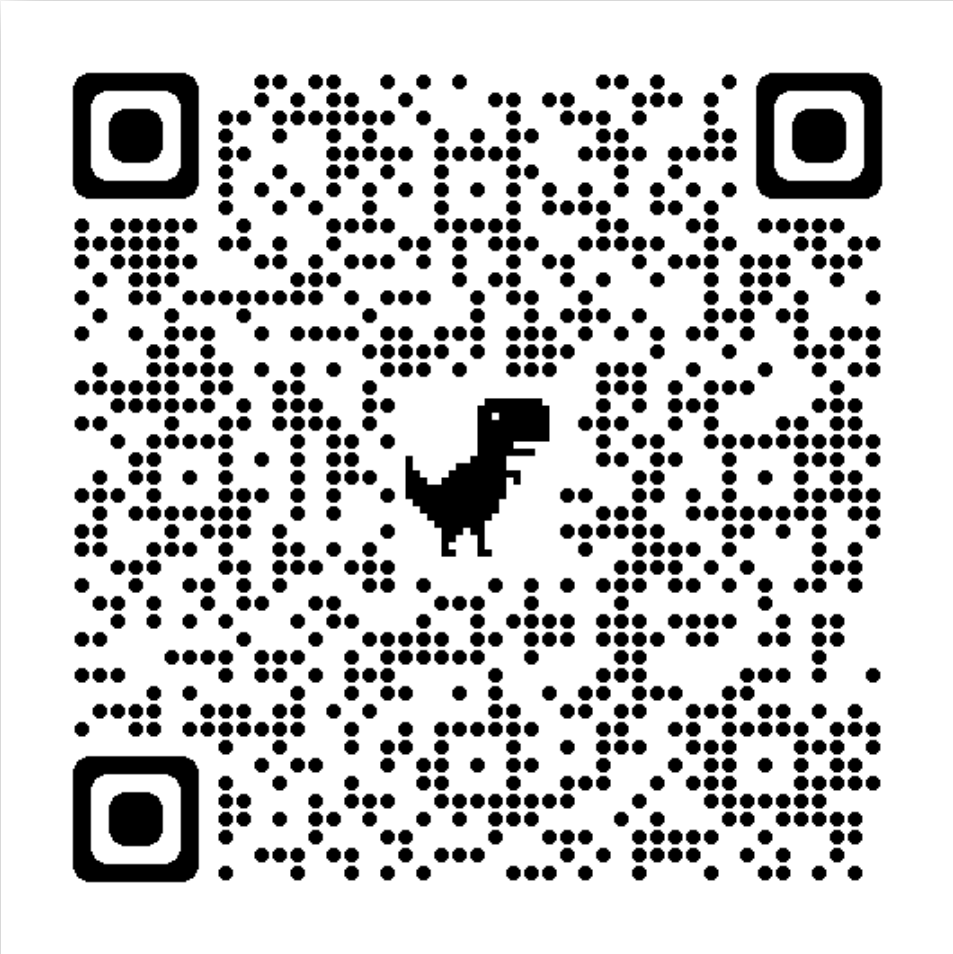
HYPERFLOW

In Quantum Security Gateways

We all know that network throughput

HyperFlow - More Information on CheckMates

- Community Discussion
- EA Info & Contact Details



ANALYSIS – TOOLS AND TIPS

Check Point
CHECKMATES

Potential bottlenecks

- CPUs
  - SNDs
  - FWKs
  - Interrupts
- Memory
  - Availability and leaks
- Network Cards
  - Settings
  - Buffers
  - Interrupts & Queues

Potential optimization points

- NIC settings and bandwidth
- Affinity
- CPU split between CoreXL and SecureXL
- SecureXL limitations and config

CPU UTILIZATION

CPU Utilization

Suboptimal split between SNDs and FWKs

Heavy connections / Elephant flows

Non-Optimal FW Policy (much less critical with R80.x)

Overwhelming NIC interrupts

CPU Utilization - diagnostics

- **top**
  - look into core utilization
- **cpview**
  - sk101878 – CPView Utility
- **cpstat -f multi_cpu os**
  - displays internal statistics for OS about all CPU cores as collected by Check Point
- **ps auxwwwf**
  - displays information about the current processes (daemons)

CPU Utilization – diagnostics, cont.

- **cat /proc/interrupts**
  - displays the number of interrupts on each CPU core from each IRQ
- **cat /proc/cpuinfo**
  - Displays a collection of CPU and system architecture dependent items about CPU (on multi-CPU (SMP) machines will show information for each CPU)
  - Collect this output to see the information about CPU (architecture, vendor, number)
- **dmesg**
  - Displays the Linux kernel ring buffer - bootup messages from various kernel modules and hardware components

COREXL DIAGNOSTICS

Check Point
CHECKMATES

Understanding Connections

- Display overall connection table (current & peak)
  `fw tab -t connections -s`
- Display connection table per FW instance (current & peak)
  `fw ctl multik stat` or
  `fw -i <instance_ID> tab -t connections -s`
- Display connection table limit
  `fw tab -t connections | grep limit`
- Display traffic type (TCP, UDP, ICMP, Other) in connection table
  `fw ctl pstat`

Understanding Connections

- List Top-10 Source IP in connection table:

```
fw tab -u -t connections -f | awk -F\; '{ print
$3 }' | sort -n | uniq -c | sort -nr | head -10
```

- List Top-10 Destination IP in connection table:

```
fw tab -u -t connections -f | awk -F\; '{ print
$5 }' | sort -n | uniq -c | sort -nr | head -10
```

Rulebase vs Connections

- Display rulebase statistics and hitcounts on gateway (sk106410)
  `cpstat blades`

```
[Expert@ClusterMember1:0]# cpstat blades

Packets accepted :          107465
Packets dropped :           0
Peak number of connections: 169
Number of connections:      27


Top Rule Hits
----------------------
|rule index|rule count|
----------------------
|Rule 0    |      1943|
|Rule 5    |       837|
|Rule 4    |        38|
----------------------
```

CoreXL diagnostics

- **cpview**
  - Displays the CoreXL status and statistics
- **fw ctl multik stat**
  - Displays status of CoreXL instances and summary for traffic that passes through each CoreXL FW instance

```
ID | Active   | CPU     | Connections | Peak
----------------------------------------------
 0 | Yes      | 3       |          26 |       66
 1 | Yes      | 2       |          36 |       64
 2 | Yes      | 1       |          39 |       53
```

CoreXL diagnostics, cons.

- **fw ctl affinity -l -r -v -a**
  - Displays affinity of interfaces, processes and CoreXL FW instances to CPU cores

```
CPU 0:   eth0 (irq 67) eth3 (irq 59)
CPU 1:   eth1 (irq 75)
         fw_2
CPU 2:   eth2 (irq 83)
         fw_1
CPU 3:   fw_0
All:     mpdaemon fwucd cpca vpnd fwm ... cpd cprid
```

CoreXL diagnostics, cons.

- **cat /proc/interrupts**
  - Displays the number of interrupts on each CPU core from each IRQ

```
           CPU0       CPU1       CPU2       CPU3
  0:   51254849          0          0          0   IO-APIC-edge  timer
  1:          4          0          0          5   IO-APIC-edge  i8042
  6:          2          3          0          0   IO-APIC-edge  floppy
  7:          0          0          0          0   IO-APIC-edge  parport0
  8:          3          0          0          0   IO-APIC-edge  rtc
  9:          0          0          0          0  IO-APIC-level  acpi
 12:          0          0          0        115   IO-APIC-edge  i8042
 15:        302       1043        168       1127   IO-APIC-edge  ide1
 59:     320145       6732       5944       6189  IO-APIC-level  ioc0, eth3
 67:   15765753          0          0          0  IO-APIC-level  eth0
 75:      24271       1285         36          0  IO-APIC-level  eth1
 83:      24272          0       1322          0  IO-APIC-level  eth2
NMI:          0          0          0          0
LOC:   50950084   50945459   50949418   50945128
ERR:          0
MIS:          0
```

Further reading

- Best Practices - Security Gateway Performance - sk98348
- SecureXL - sk98722 - ATRG: SecureXL
- CoreXL - sk98737 - ATRG: CoreXL
- SMT (HyperThreading) - sk93000 - SMT (HyperThreading) Feature Guide
- Multi-Queue - sk80940
- ClusterXL - sk93306 - ATRG: ClusterXL
- VPN - sk105119 - Best Practices - VPN Performance and to sk104760 - ATRG: VPN Core

QUESTIONS?

Check Point
CHECKMATES

Full list of Performance Series

- Part 1 – Introduction
- Part 2 – SecureXL
- **Part 3 – CoreXL**
- Part 4 – Clustering and Hyperscale
- Special– Diagnostics How To

THANK YOU

Valeri (VAL) Loukine

Cyber Security Evangelist | Community Lead

CheckMates Live Virtual Series 2022

Check Point CHECKMATES