DPDK | IT大咖说 知识分享平台

DPDK DATA PLANE DEVELOPMENT KIT

THE LINUX FOUNDATION

DPDK SUMMIT CHINA 2017

主办方：intel

参与方：腾讯云 ZTE 美团云 MEITUAN OPEN SERVICES Panabit® 太一星晨 Balance Your Networks UnitedStack有云 云杉网络 Yunshan Networks

协办方：SDNLAB 专注网络创新技术　　视频支持方：IT大咖说 知识分享平台

Tencent Data Center Security Use case

DPDK based security service layer on datacenter

haohao zhang

主办方： intel

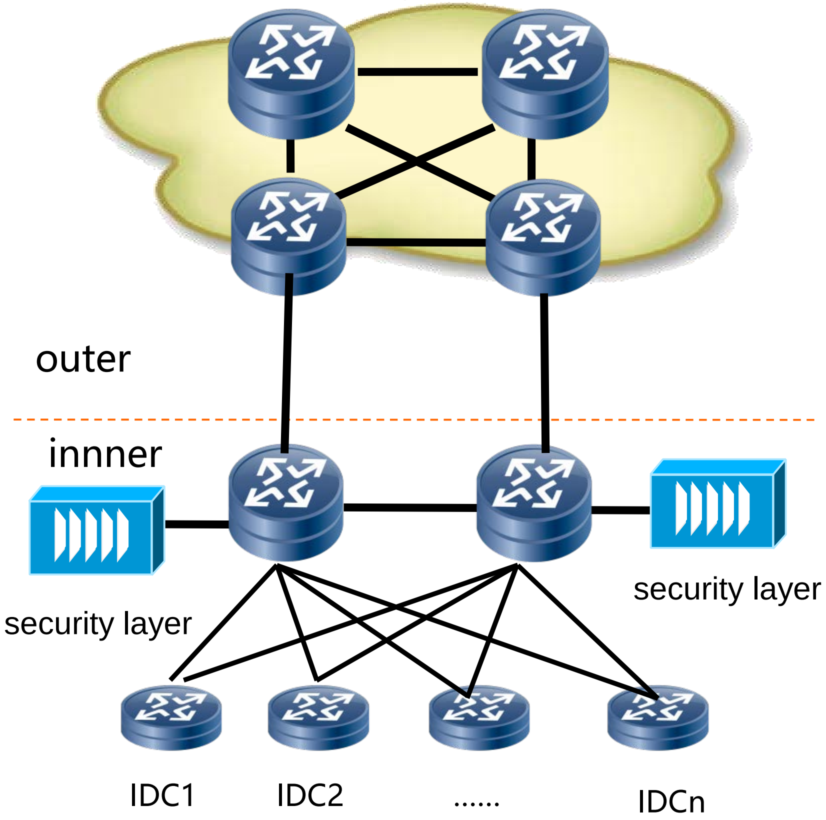
参与方： 腾讯云 ZTE 美团云 Panabit 太一星晨 云杉网络

协办方： SDNLAB　　视频支持方： IT大咖说

Agenda

- what and why of the security service layer
- how to design
- performance optimize
- evolution for large volume

where is the layer



security on host：

different OS
CPU used must be controled
complex policy

challenges of deploy on network perimeter：

traffic control
foward information--act like a router
performace --latency,Throughput
debug

support security function

Anti-DDoS
WAF
IDS
Forensics
....

comparion between DPDK and aother option

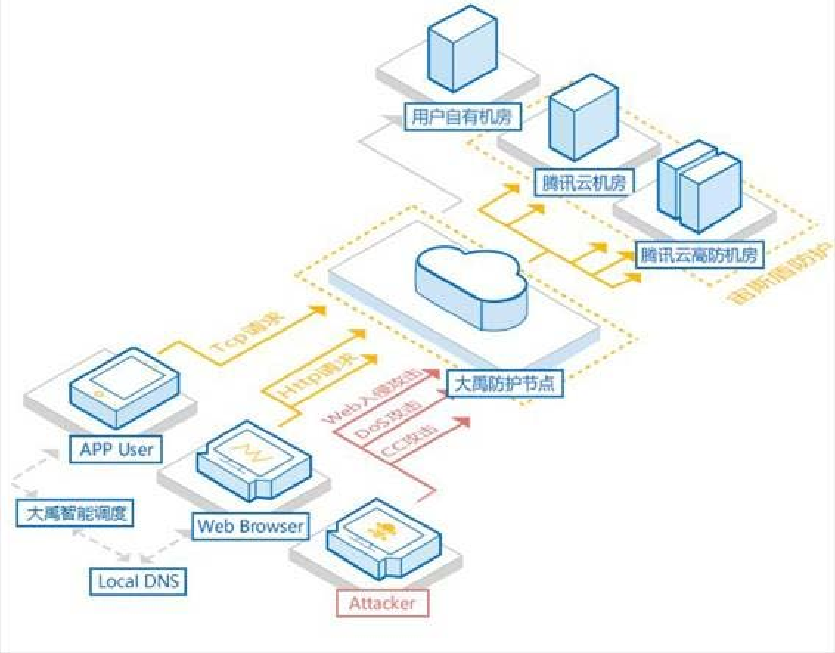
	multi core platform	DPDK on x86
Performance	high	middle
Reliability on massive deployment	high malfunction ratio	good
Devepment cost	high	middle
Debug	hard	midlle

1 Reliability is very important when you need manage thousands of devcies.

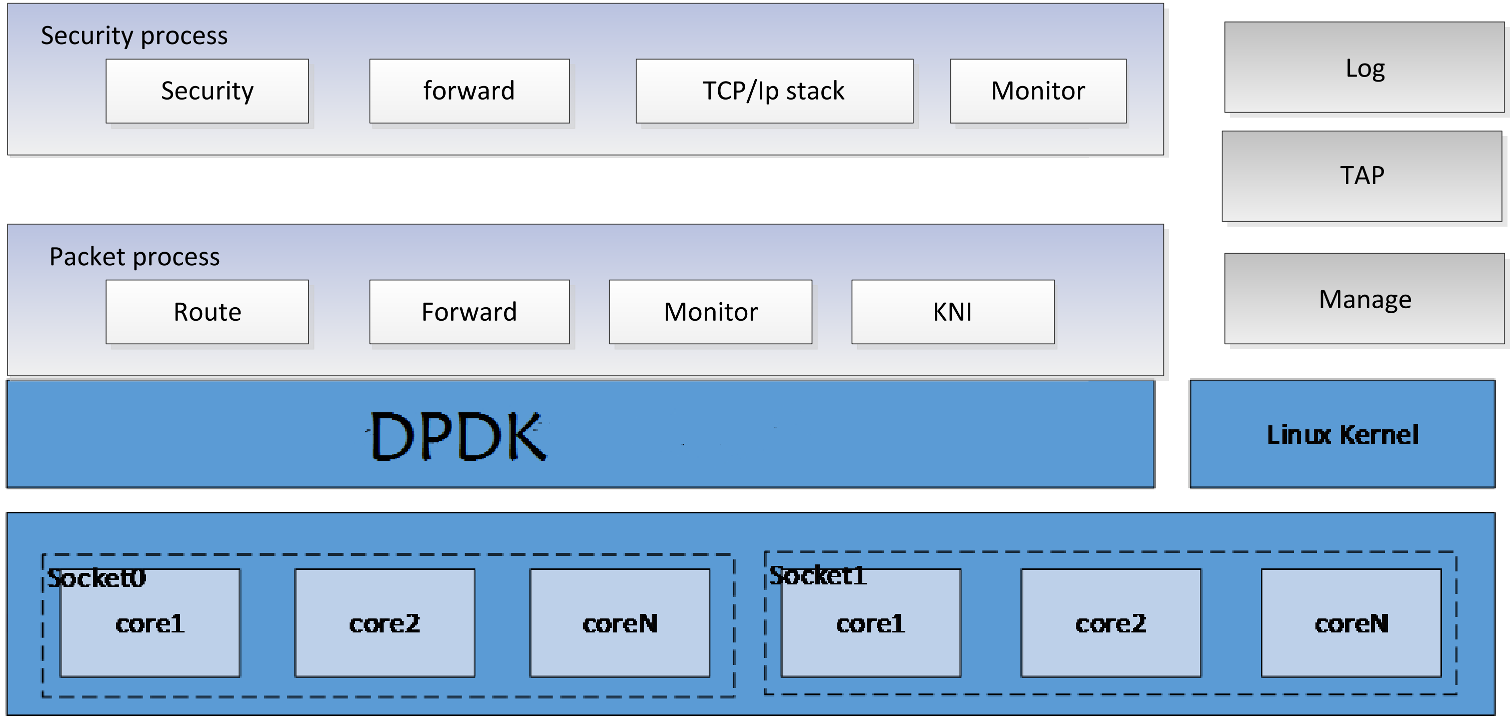
2 Some weird problem leads to hareware like fiber

deployment status

- covery all the perimeter globally
- Thousands of DPDK based security device
- support all type of business,incldue web,game,video...
- continue increasing



the software architecture



multi process model

- **primary process is stable**

  packet forwarding information

  router inforamtion

  memory management

  basic statistic

  TAP management

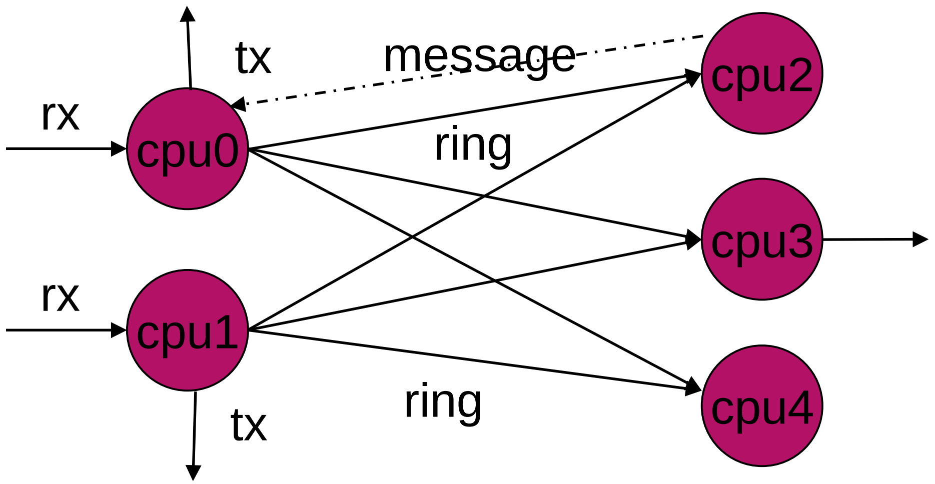
- **secondary process changes quickly**

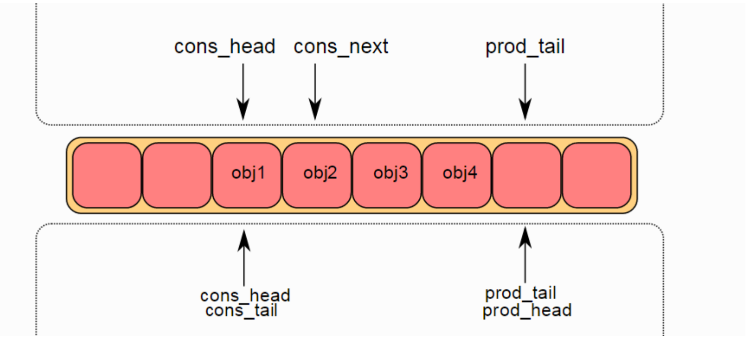
  doing the security logical

  update frequently

fake dequeue

1. avoid packet losing after secondary process crash
2. not in high performance mode





① secondary read head and tail of ring

② secondary copy mbuf without dequeue

③ secondary process packet

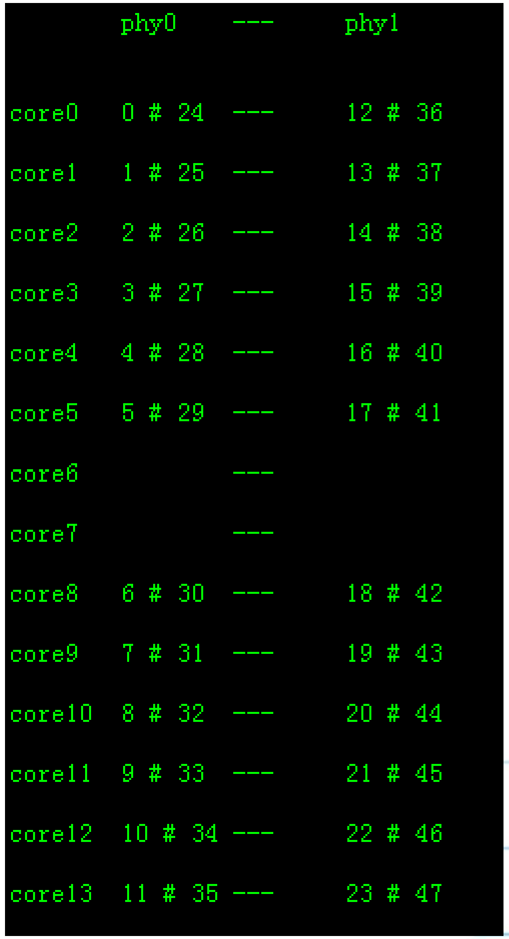
④ use volitile varible notify the primary process to dequeue

performance optimization

- Local varible VS global varible
- Hyper Thread for different scenarios:

  whether threads number need larger than physical thread

  pipiline mode
- sometimes tx queue number can affect
- assemble language can be used for critical function
- memcpy cost

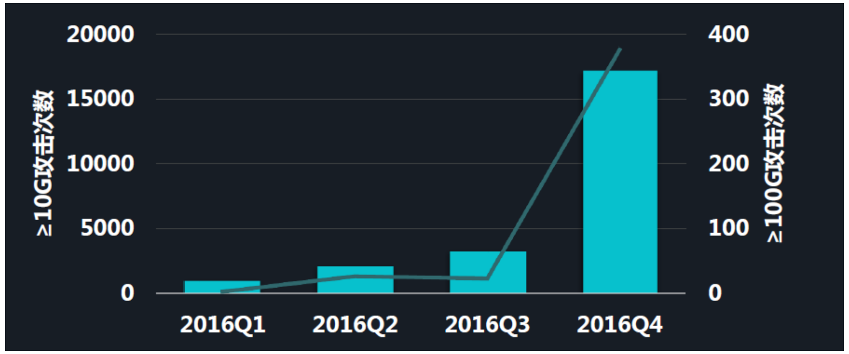


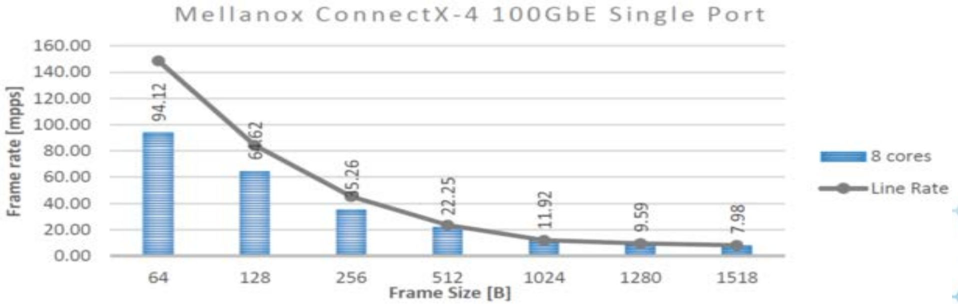
```
        phy0    ---    phy1

core0   0 # 24  ---    12 # 36
core1   1 # 25  ---    13 # 37
core2   2 # 26  ---    14 # 38
core3   3 # 27  ---    15 # 39
core4   4 # 28  ---    16 # 40
core5   5 # 29  ---    17 # 41
core6           ---
core7           ---
core8   6 # 30  ---    18 # 42
core9   7 # 31  ---    19 # 43
core10  8 # 32  ---    20 # 44
core11  9 # 33  ---    21 # 45
core12  10 # 34 ---    22 # 46
core13  11 # 35 ---    23 # 47
```

evolution for large volume

- 100G NIC
- FPGA
- CPU on other NUMA Node





Thanks!!