Enabling “Protocol Routing” in Internet Communications via DMM

www.huawei.com

Waterman Cao（Cloud Networking Lab, CSI）
曹水（中央软件院罗素部）

HUAWEI TECHNOLOGIES CO., LTD.

Agenda

- What we face
- Challenges in Transportation Layer Protocol
- Proposed Solutions
- Use Cases

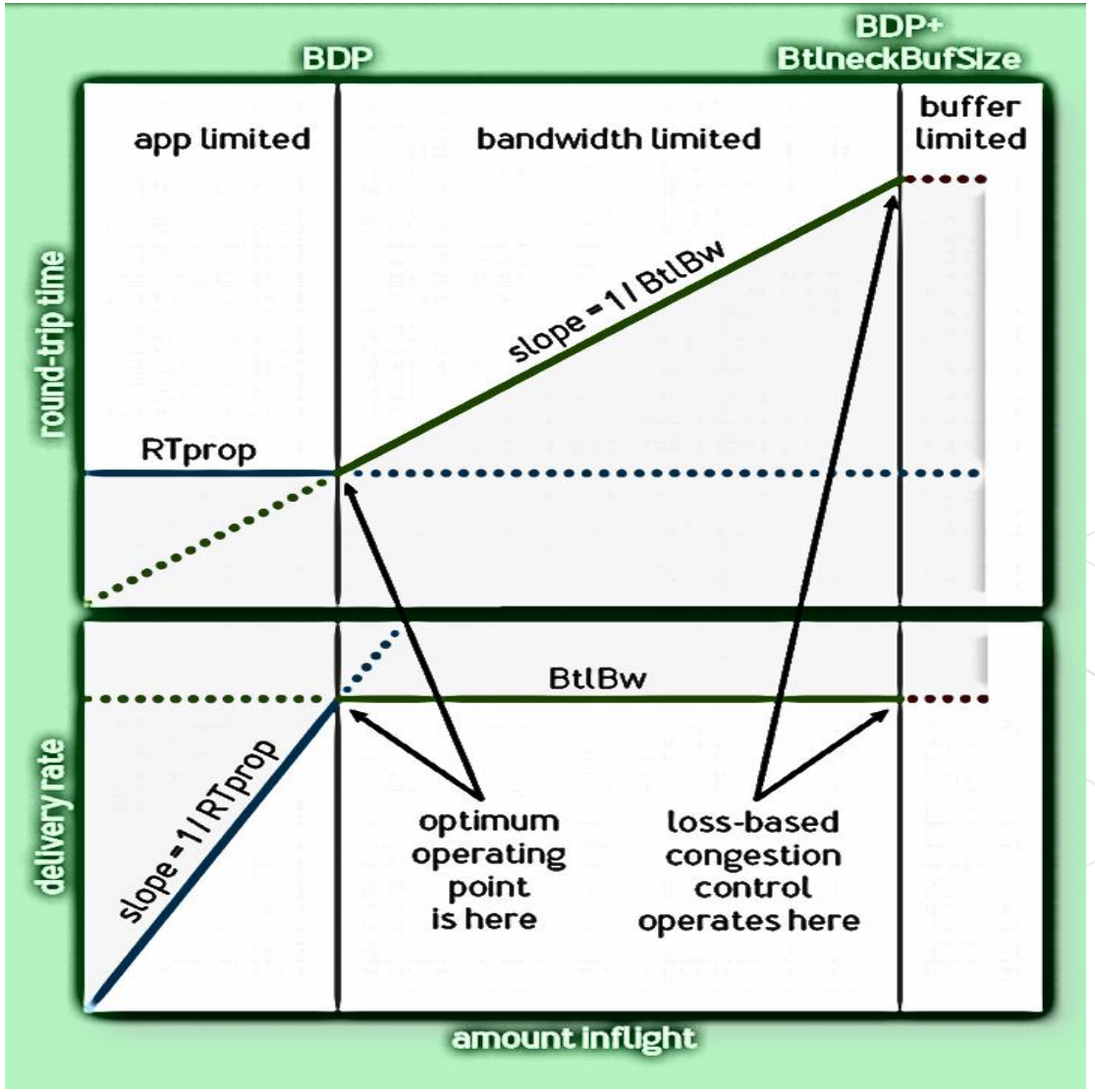
What we face today

Why is TCP Protocol usually found as the performance killer ?

- ✓ Loss Sensitive
- ✓ Poor in Larger BDP① network
- ✓ No delivery latency SLA guarantee
- ✓ Difficult to tune performance

① :Bandwidth-Delay Product

Delivery Rate and Round-[illegible]



Challenges in the future Internet Transport Layer

Ultimate performance requirement

Emerging applications is bringing extremely high performance requirements to the network system. For example: VR/AR required dozens of HD Video Transportation.

Diversified transport QoS/SLA requirements

The choice for congestion control algorithm is well known as a tradeoff between throughput and latency. An one-fit-all protocol or algorithm becomes less and less feasible.

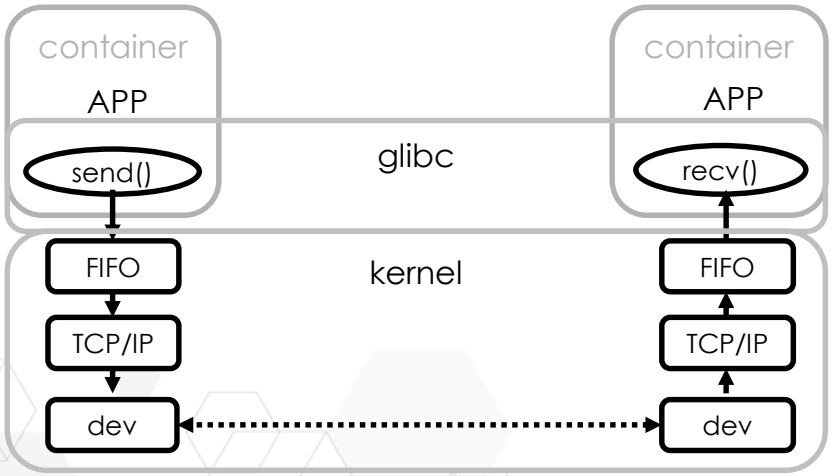
Complicated network environments

The coexistence of virtual network, wireless network and other types of network leads to both great challenges and opportunities on transport protocol design.

Trend: Kernel space or User space ?

Oops..

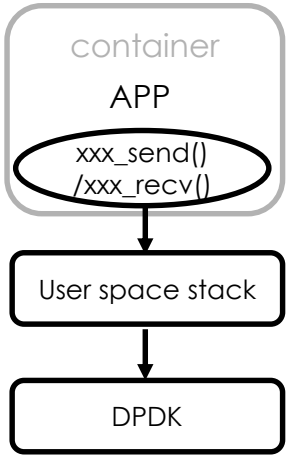
Go through kernel space and user space many times when communication between containers



How about

User space stack instead of kernel

TCP/IP stack



Challenge：the use space stack need to provide backward compatibility to the kernel networking ecosystem (netfilters/iptables, qdisc/tc, etc.) which some App heavily relies on

Trend: Apps have different flavors

Oops…

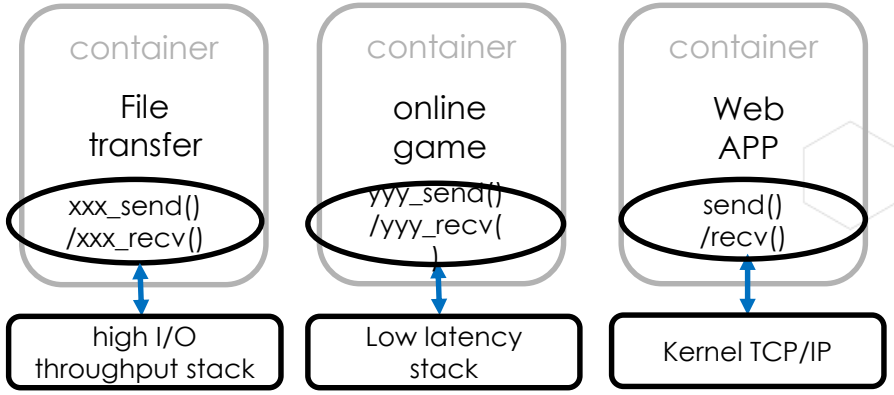
different APP may need different user space stack

--Some requires high I/O throughput, Some requires low latency，and some may require more protocol compliances/security

--Some may even need to choose on-demand (according to peers and network environment)

How about…

Develop different stack to match different requirement



Challenge：Application can not choose stack on demand on the deploy-time

Trend: Hardware is not all the same

Oops…

Network Protocol Stack developers may not be aware of the underlying hardware platform (cloud scenario), and thus could not may fully use of the hardware all by its own…

How about…

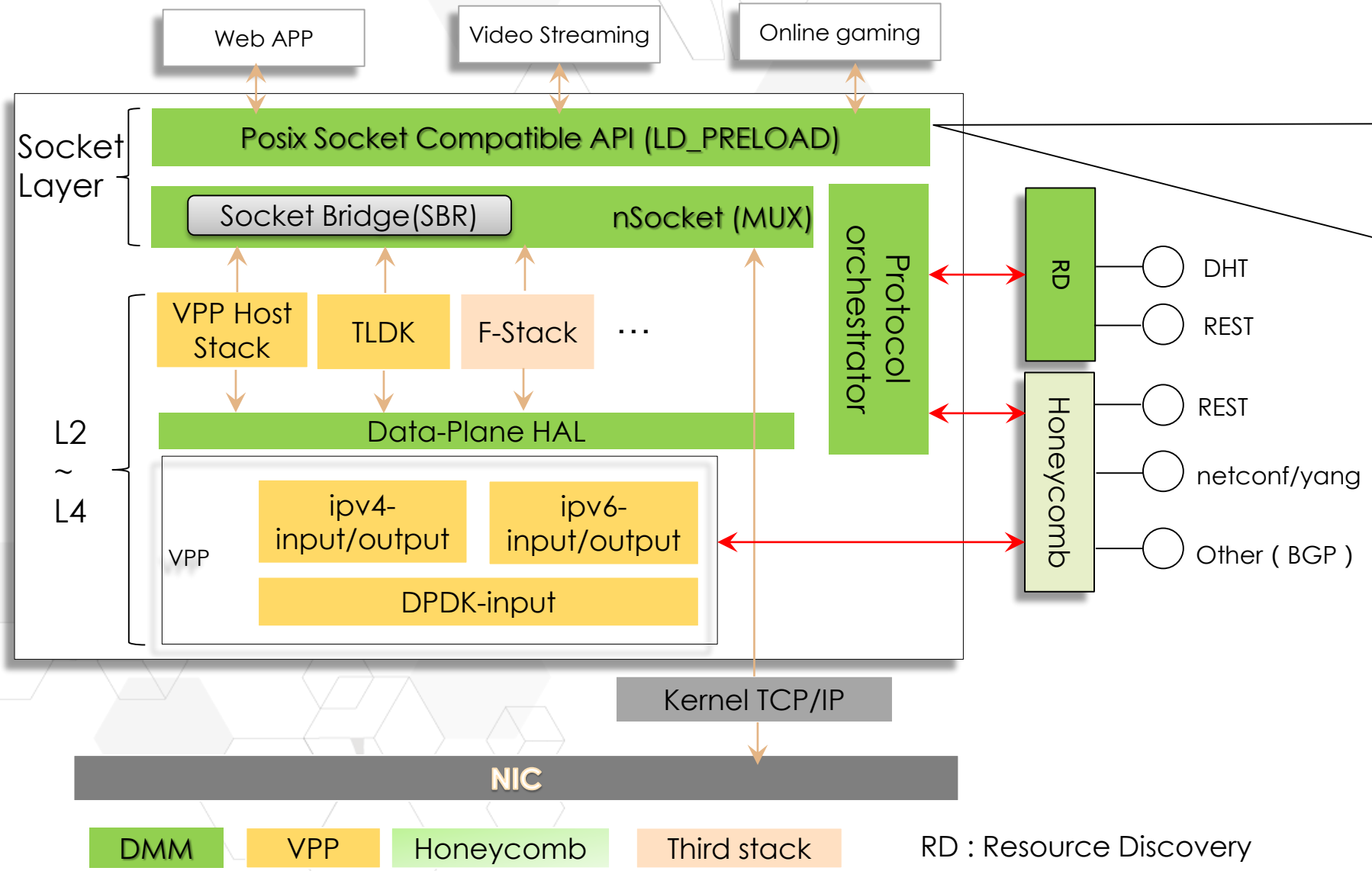
Providing more choices at the same time, each handling one possible choice, and put them together…

Challenge：How to identify the difference of hardware e.g. how to adapt RDMA, and how to host them simultaneously?

DMM Project

- **DMM Protocol Framework: Dual mode, Multiple protocols & Multiple instances, aim to provide a new solution of diverse protocol stacks for developers.**
  - Support Kernel Space and User Space
  - Simplify new protocol adoptions and Integrations
  - Enable "protocol routing" in Cloud Networking
- **DMM will be open source software, licensed as Apache.**

DMM: Protocol Stack Common Framework



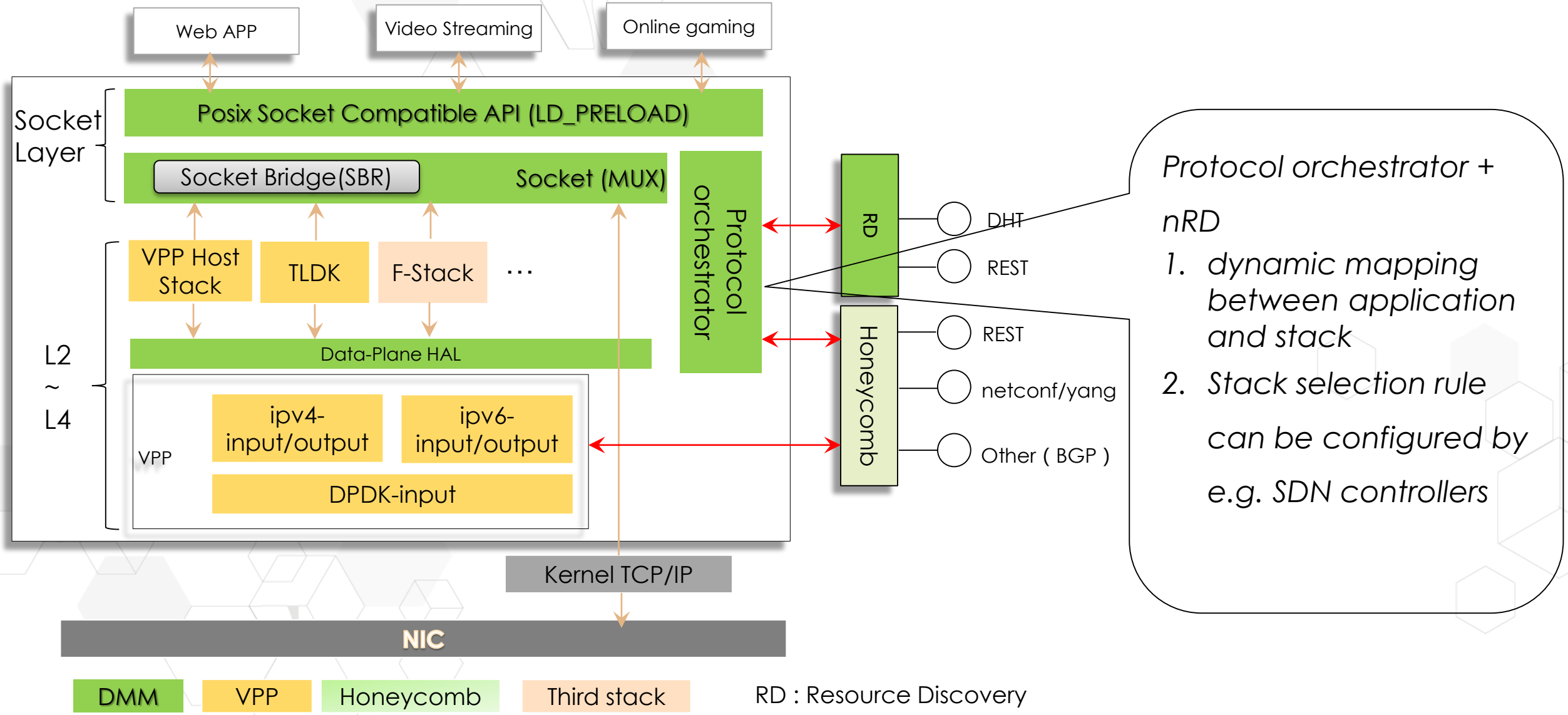
Posix Socket Compatible API + nSocket

1. Posix compatible and uniform socket API to APP
2. *Support both kernel TCP/IP stack and user space stack*

DMM | VPP | Honeycomb | Third stack

RD : Resource Discovery

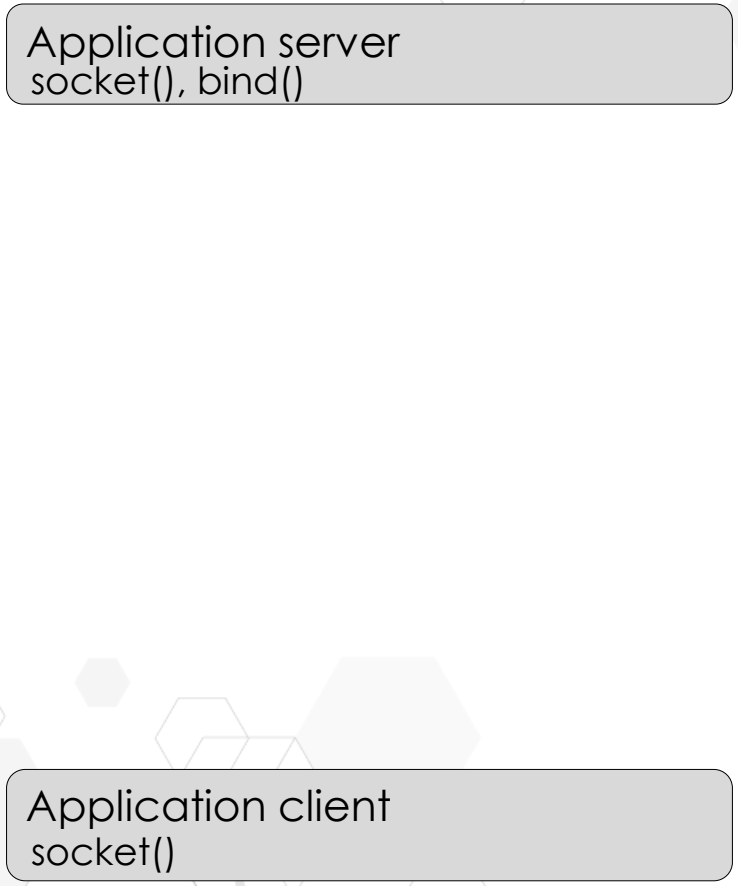
DMM: Protocol Stack Common Framework



Protocol orchestrator + nRD

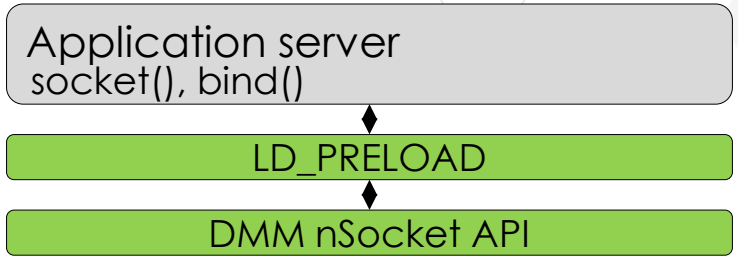
1. *dynamic mapping between application and stack*
2. *Stack selection rule can be configured by e.g. SDN controllers*

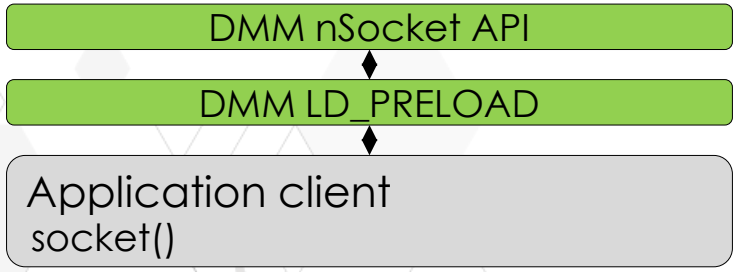
Protocol Routing Workflow



1 Application server and client calls socket interface.

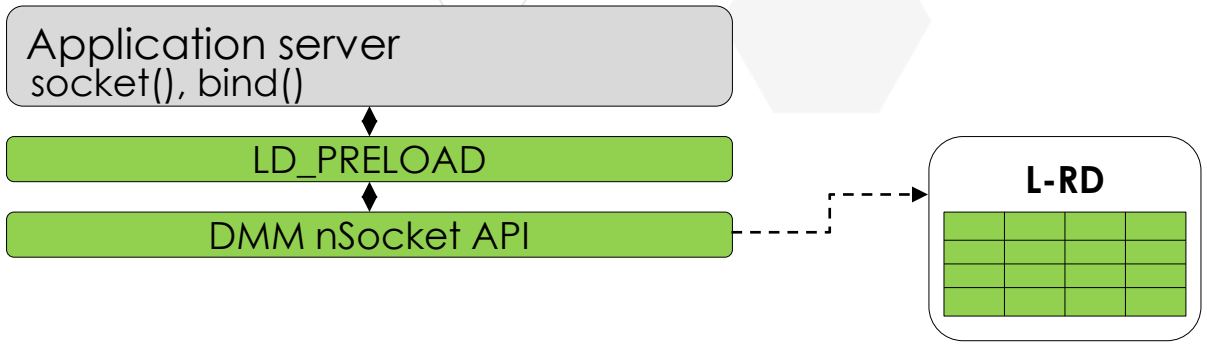
Protocol Routing Workflow

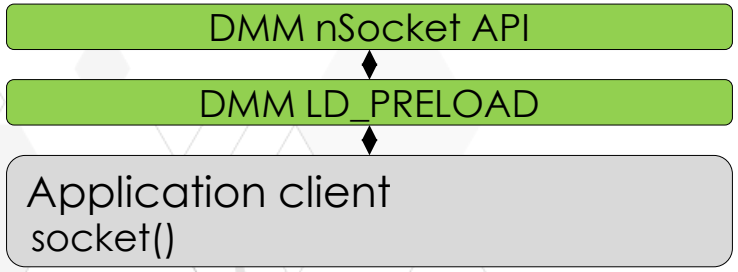




1. Application server and client calls socket interface.
2. Socket APIs are hijacked to DMM nSocket APIs.

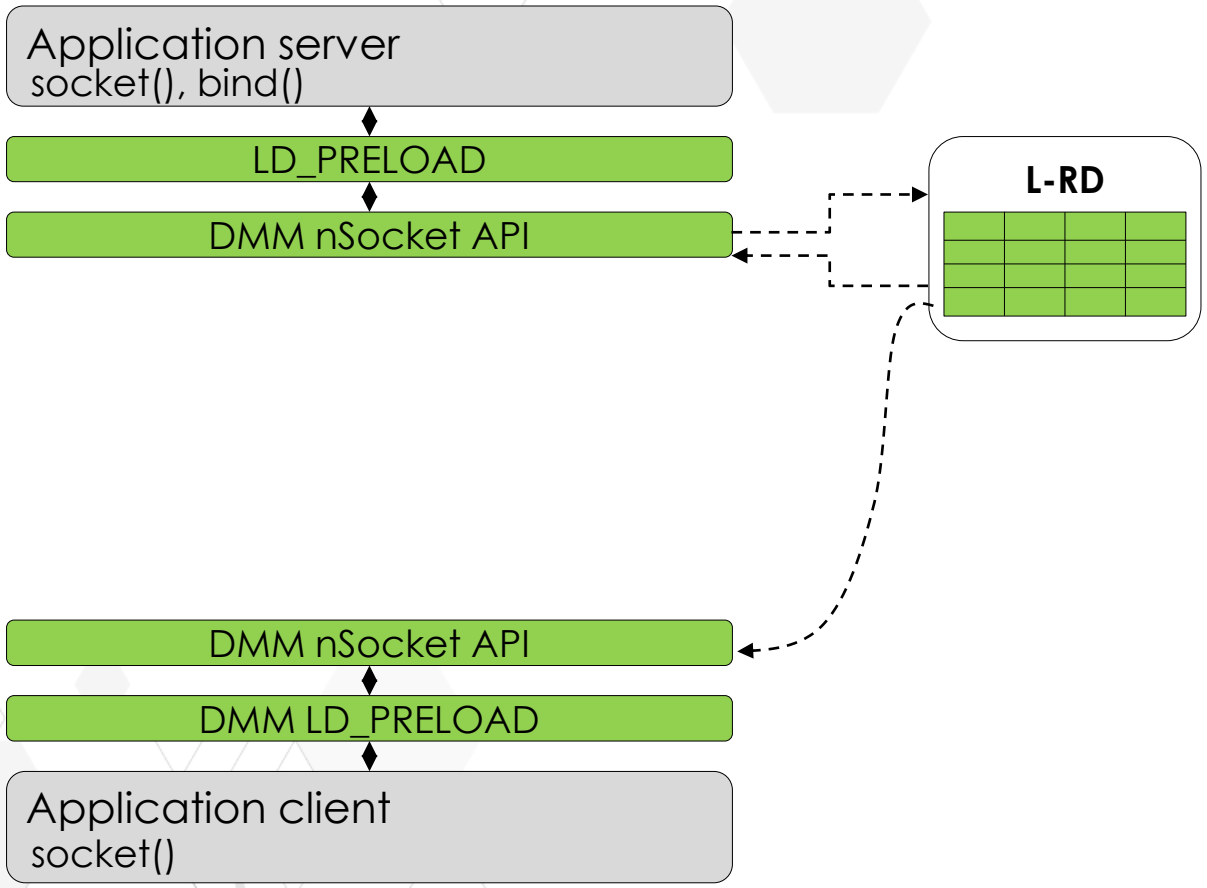
Protocol Routing Workflow





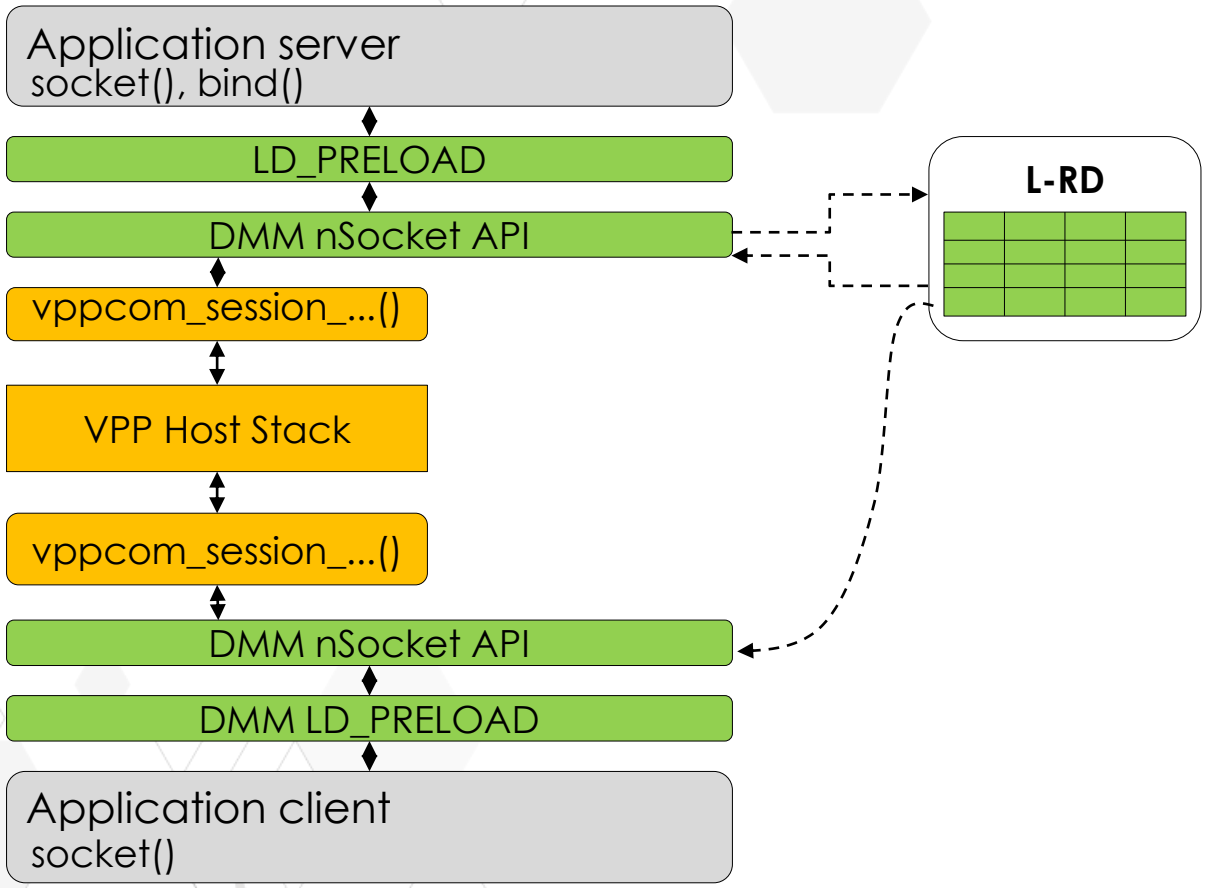
1. Application server and client calls socket interface.
2. Socket APIs are hijacked to DMM nSocket APIs.
3. Server call listen() triggers L-RD to negotiate protocol policies.
   L-RD: manage local DMM Policies and Protocol Configure.

Protocol Routing Workflow



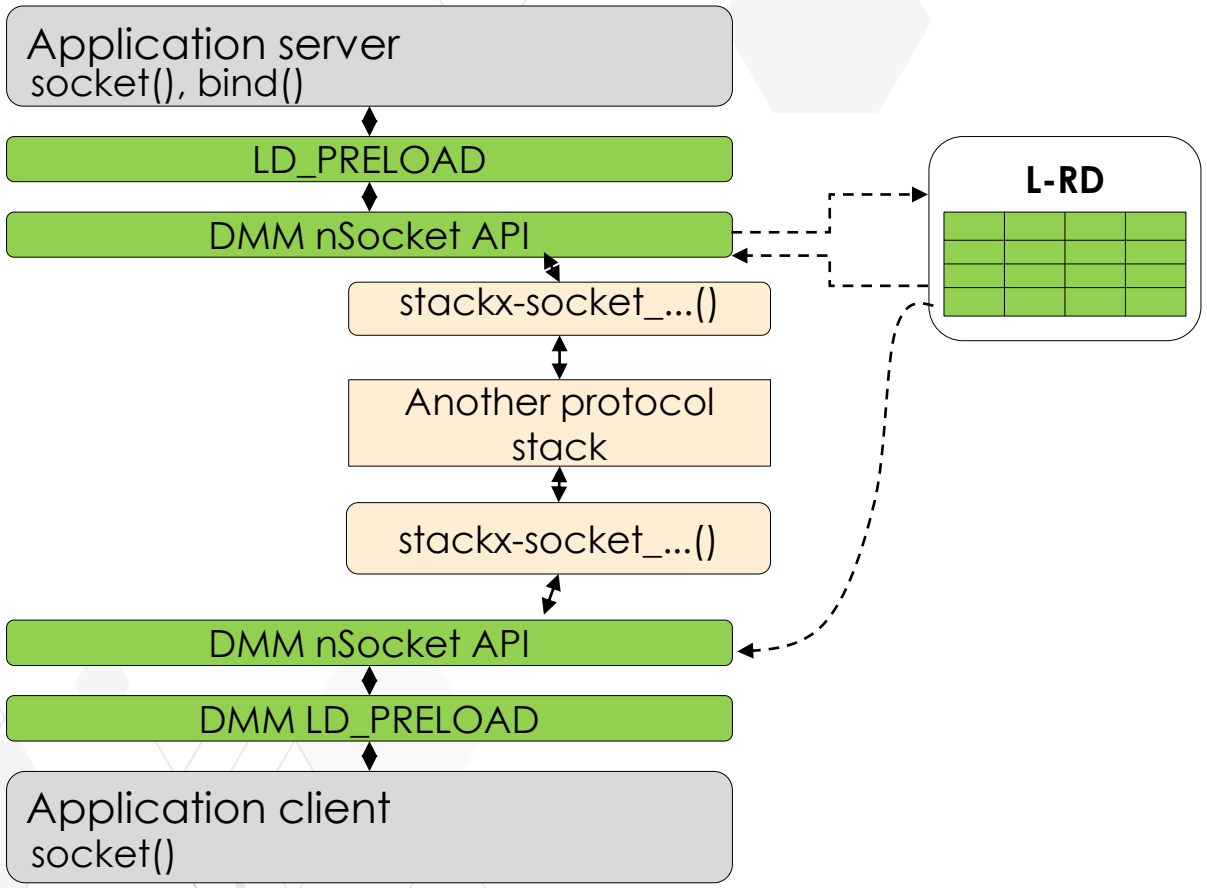
1. Application server and client calls socket interface.
2. Socket APIs are hijacked to DMM nSocket APIs.
3. Server call listen() triggers L-RD to negotiate protocol policies.
   L-RD: manage local DMM Policies and Protocol Configure.
4. Server call accept() and client call connect() trigger L-RD to retrieve and resolve protocol stack mapping.

Protocol Routing Workflow



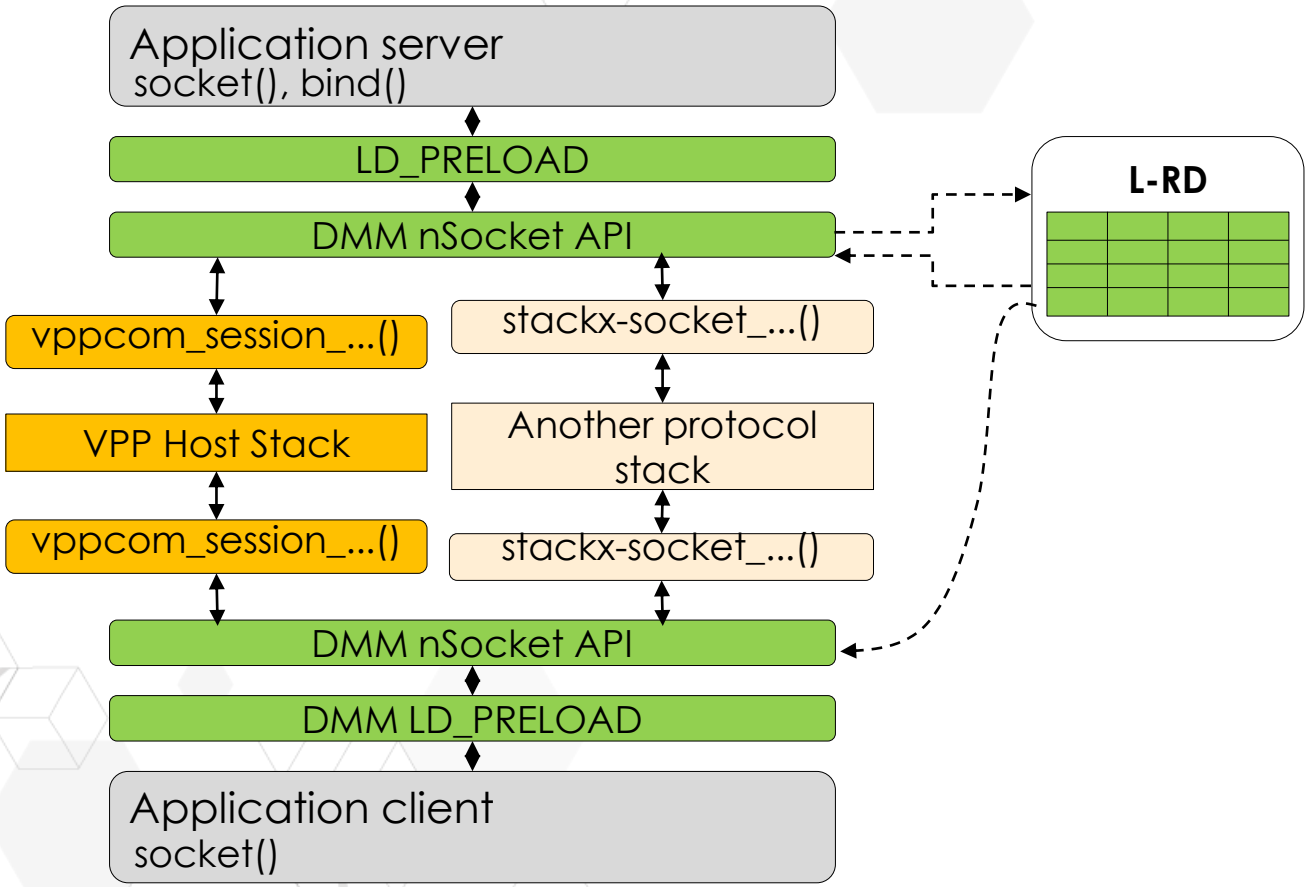
1. Application server and client calls socket interface.
2. Socket APIs are hijacked to DMM nSocket APIs.
3. Server call listen() triggers L-RD to negotiate protocol policies.
   L-RD: manage local DMM Policies and Protocol Configure.
4. Server call accept() and client call connect() trigger L-RD to retrieve and resolve protocol stack mapping.
5. According to the mapping, the socket is instantiated to one protocol stack

Protocol Routing Workflow



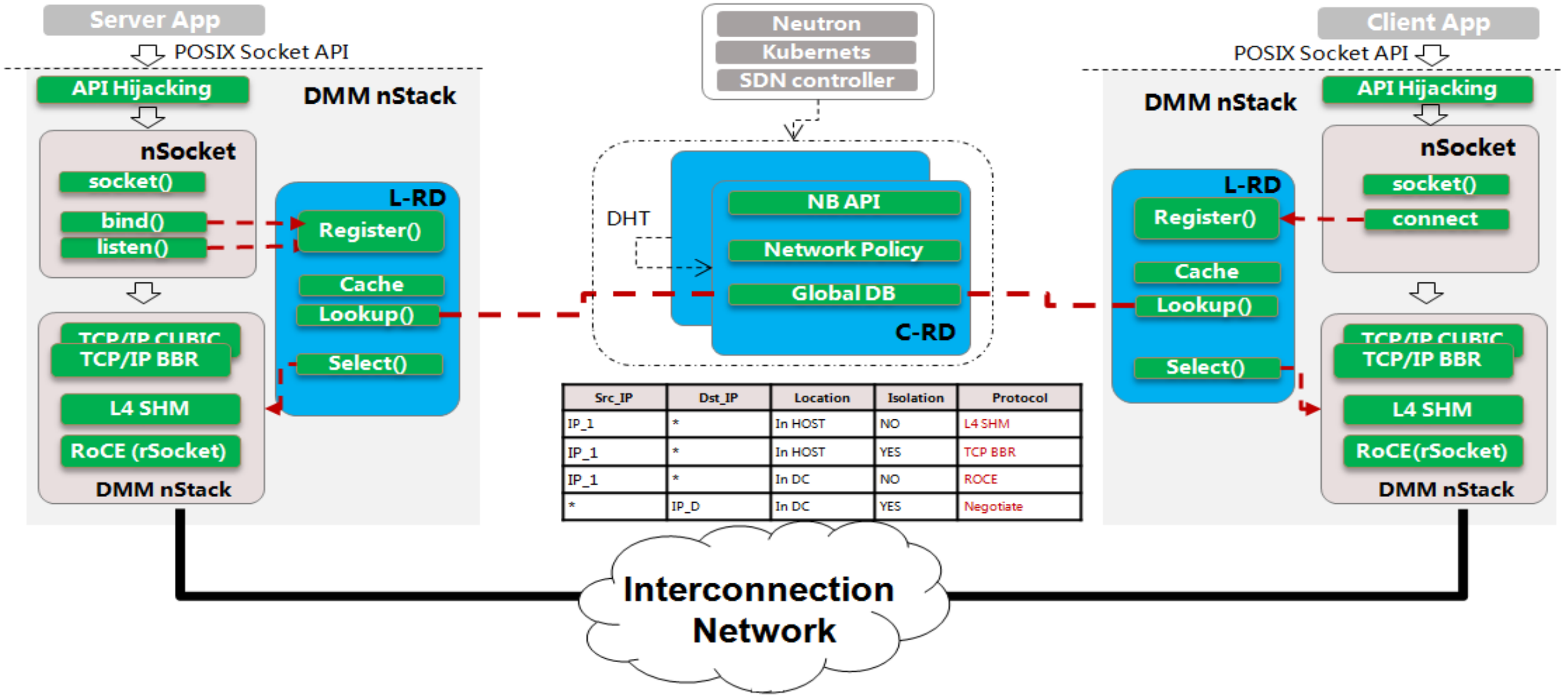
1. Application server and client calls socket interface.
2. Socket APIs are hijacked to DMM nSocket APIs.
3. Server call listen() triggers L-RD to negotiate protocol policies.
   L-RD: manage local DMM Policies and Protocol Configure.
4. Server call accept() and client call connect() trigger L-RD to retrieve and resolve protocol stack mapping.
5. According to the mapping, the socket is instantiated to one protocol stack or Another.

Protocol Routing Workflow



1. Application server and client calls socket interface.
2. Socket APIs are hijacked to DMM nSocket APIs.
3. Server call listen() triggers L-RD to negotiate protocol policies.
   L-RD: manage local DMM Policies and Protocol Configure.
4. Server call accept() and client call connect() trigger L-RD to retrieve and resolve protocol stack mapping.
5. According to the mapping, the socket is instantiated to one protocol stack or Another.
6. Dual mode(kernel or user-space), Multiple protocols, Multiple instances can exist simultaneously.

Protocol Routing Workflow (with Centeral RD)

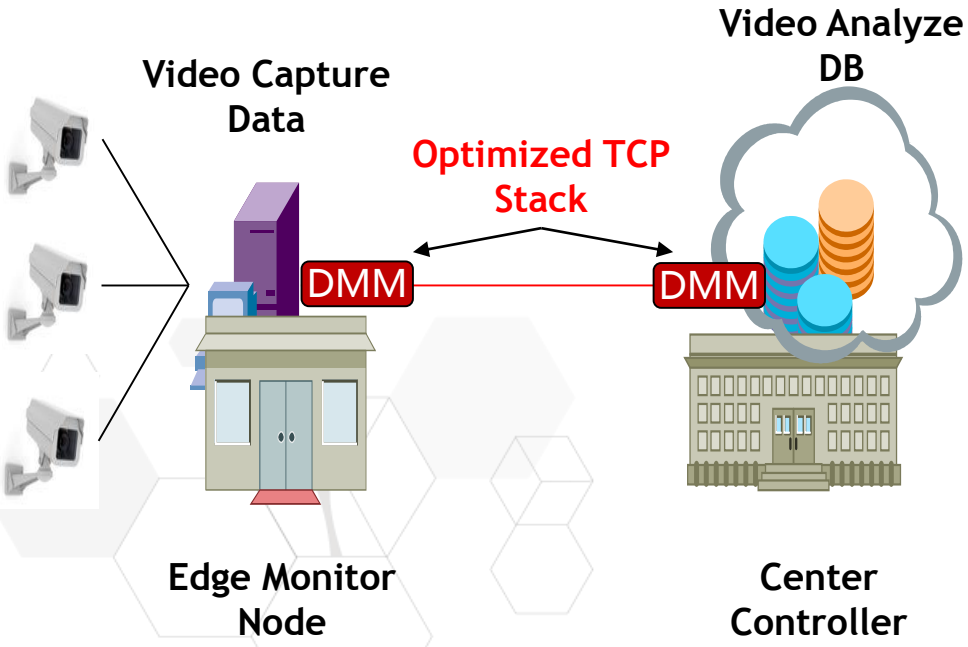


Src_IP	Dst_IP	Location	Isolation	Protocol
IP_1	*	In HOST	NO	L4 SHM
IP_1	*	In HOST	YES	TCP BBR
IP_1	*	In DC	NO	ROCE
*	IP_D	In DC	YES	Negotiate

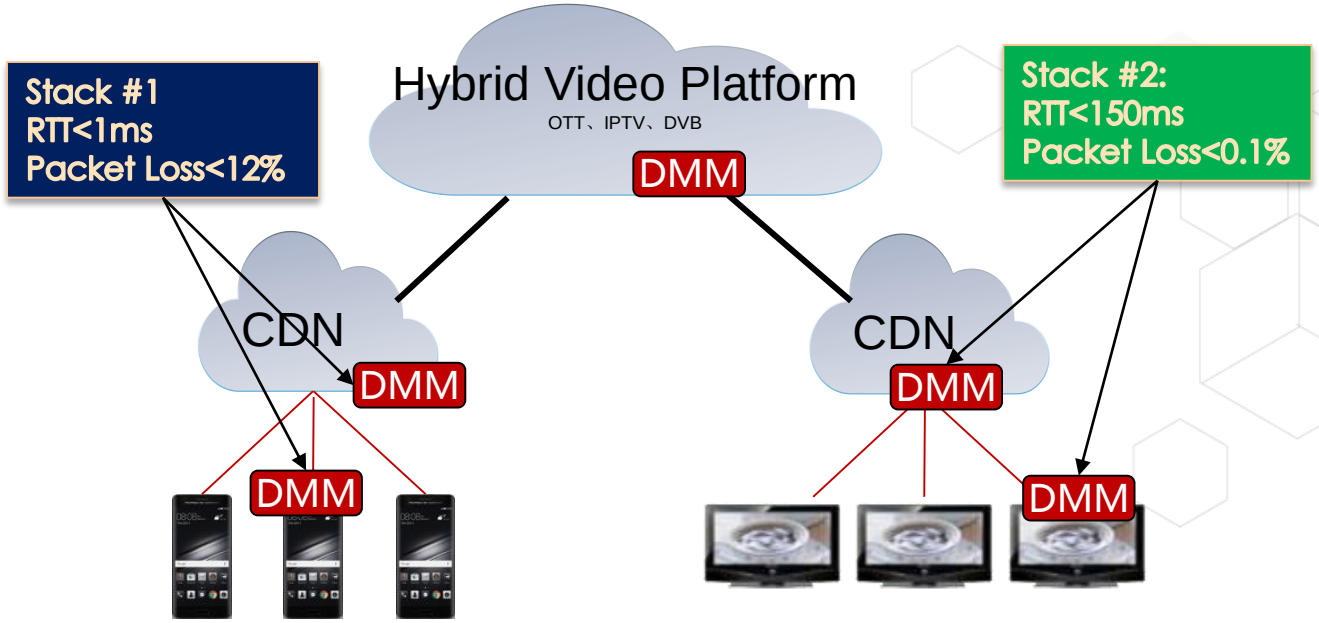
User Case for DMM

- **Traditional TCP has limited bandwidth usage in the WAN, because:**
- **1) TCP Congestion Control will limit delivery rate**
- **2) Packet Loss and Resend will decrease throughput.**

Optimized TCP stack in DMM can achieve 90% bandwidth usage and low latency, easy to deploy.



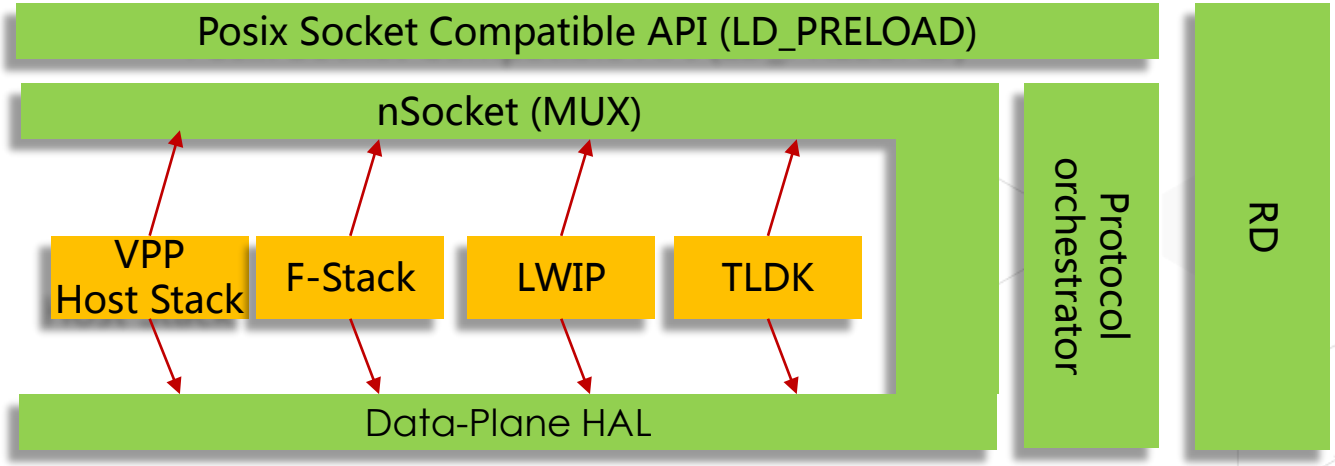
Optimized TCP stack in DMM can support mass concurrent connections and achieve smooth user experience even 12% packet loss rate.



Key Takeaways

- **Stack developers can concentrate on** user space protocol innovations;
- **Apps** can **dynamically choose different protocols.**
- **Support both kernel TCP/IP stack and user space stack ;**
- **Container network will easily build E2E communication capacities.**

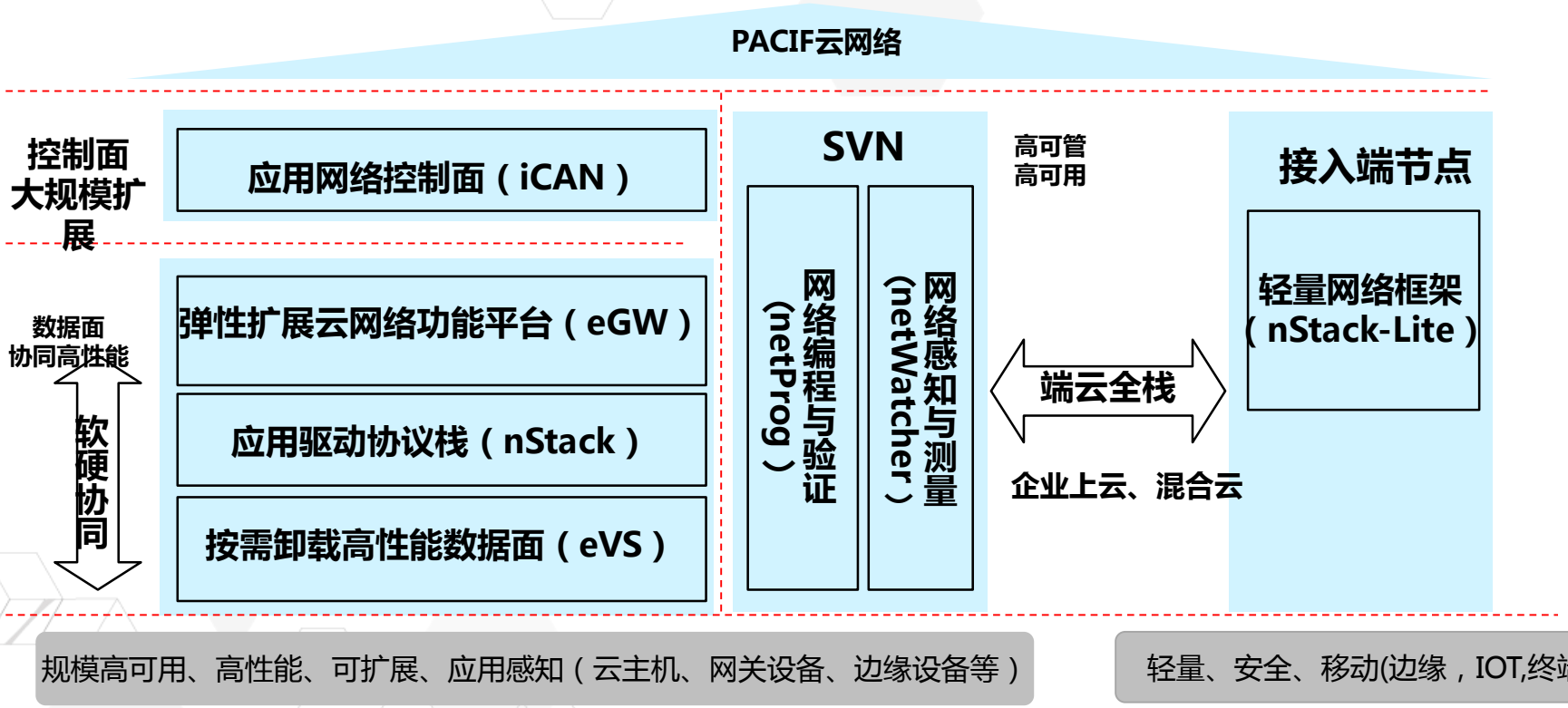
Common library for develop user space stack



HAL：Hardware Abstraction Layer，a IO adaptor

关于华为罗素：软件定义未来网络

支撑管道纵深（使能运营商，不可替代），云和IT，消费者终端和行业数字化“五大战略”



PACIF Cloud Network：

1、**P**redictable
2、**A**pp-centric
3、s**C**alable
4、**I**ntelligent
5、**F**lexible&veri**F**iable

我们的使命：构筑业界领先的，PACIF云网络和端云通信软件全栈，最佳通信体验支撑公司端管云战略。

Question ?

欢迎加入DMM讨论组

Thank you

www.huawei.com

The information in this document may contain predictive statements including, without limitation, statements regarding the future financial and operating results, future product portfolio, new technology, etc. There are a number of factors that could cause actual results and developments to differ materially from those expressed or implied in the predictive statements. Therefore, such information is provided for reference purpose only and constitutes neither an offer nor an acceptance. Huawei may change the information at any time without notice.