Chinese-American Networking Symposium 2013

An IPv6 Transition Scenario

BUPT

zhaoqin

2013.09.10

What we have learned from IPv4

- 1995.5 China Telecom begin to establish CHINANET backbone
- 1996.1 Internet is provided by CHINANET throughout the country
- The first Internet content providers emerge
  - 1997.06 163.com is online
  - 1998.02 sohu.com is online
  - 1998.12 sina.com.cn is online
- The Internet users emerge
  - 1998 The word 网民 (netizen) is firstly created in China
  - Now The number of netizen grows rapidly to 591 million

IPv4 backbone is chicken, ICPs and Users are eggs;

IPv4 backbone is evoled from telecommunication,

IPv4 backbone isn't hatched from ICPs and Users;

Today

- IPv4 ecosystem is crashing
  - IPv4 address space is already depleted
  - NAT is used to keep IPv4 continuing
- IPv6 ecosystem is building
  - Many IPv6 trial networks have been verified successfully
  - Transition solutions are proposed to keep the switch between two ecosystems smoothy
- Transition period may last for decades

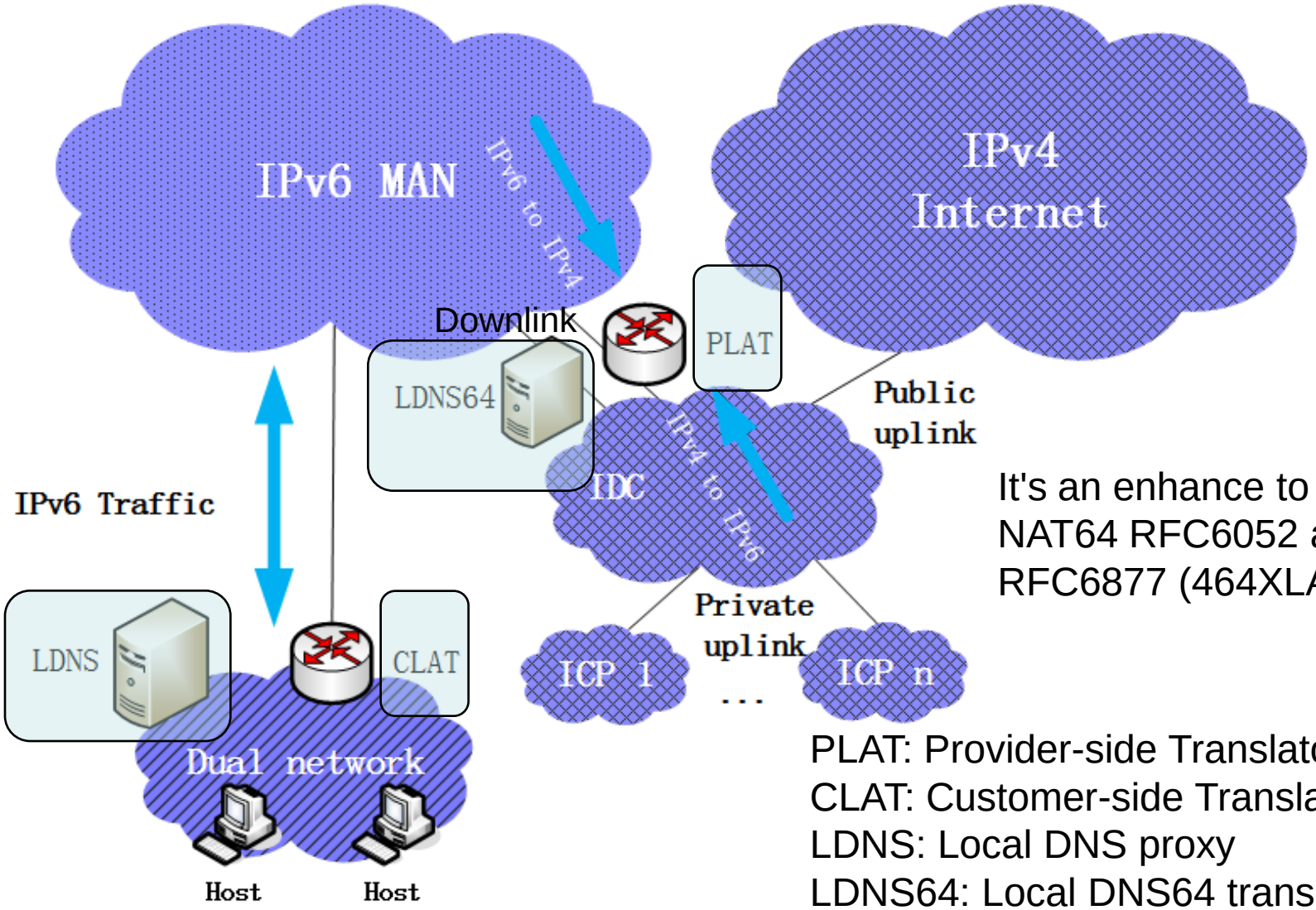
What we have seen about IPv6

- IPv6-only MAN (Metropolitan Area Network) may appear soon
- Users will be provided with dual stack access network for compatibility
- IPv4 connectivity may be isolated as islands in the oceans of IPv6 MAN
- CDN (Content Delivery Network) operator requires relation between IPv4 and IPv6 address to offer nearest web cache to IPv6 users

A model of IDC

- Downlinks are connected to the MAN (Metropolitan Area Network) of ISP
- Private uplinks are connected to ICP (Internet Content Provider), CDN (Content Delivery Network) operator or ICP’s own CDN group
- Public uplinks are connected to superior network operator

IDC IPv6 Transition Scenario

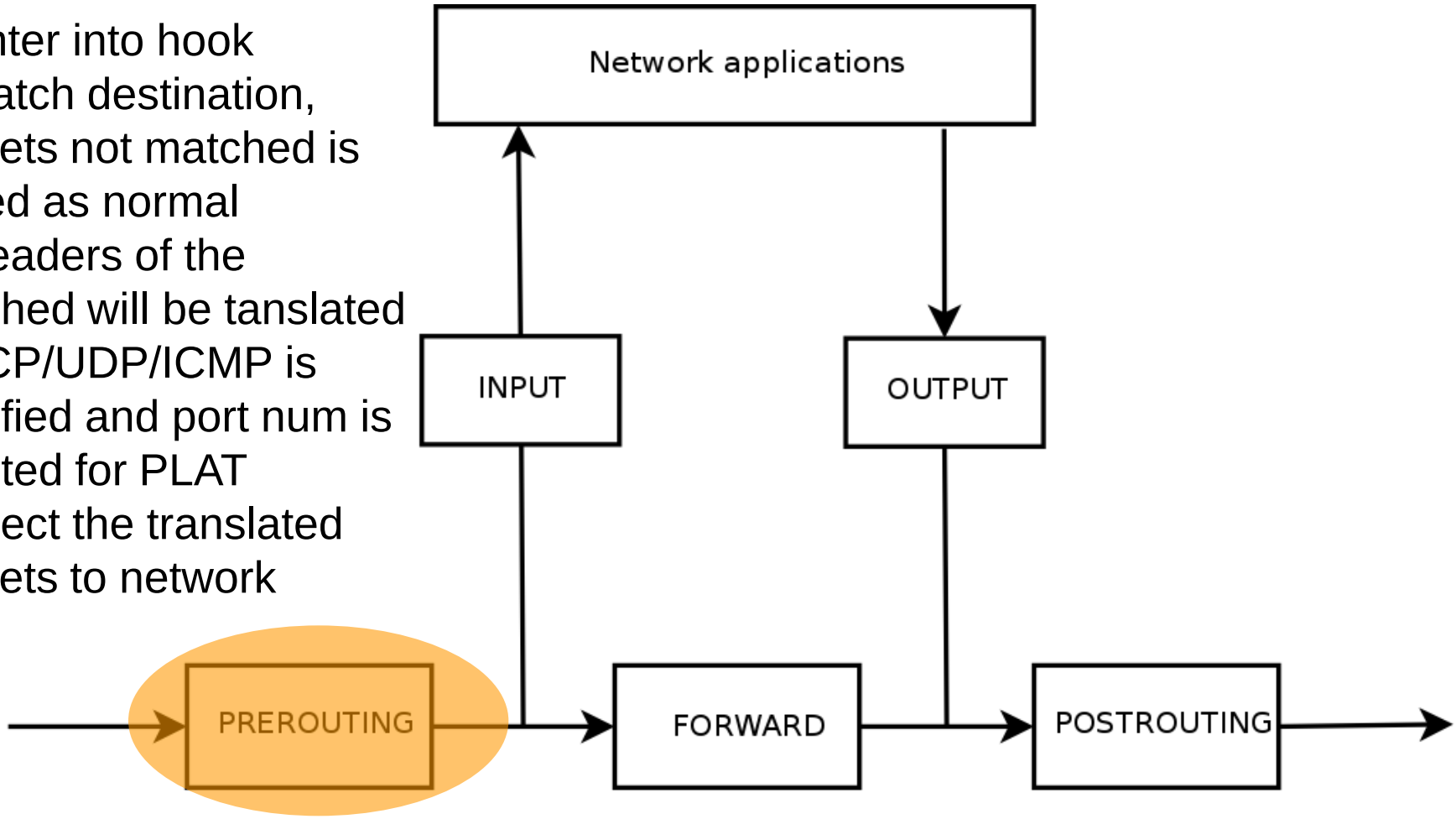


It's an enhance to NAT64 RFC6052 and RFC6877 (464XLAT)

PLAT: Provider-side Translator
CLAT: Customer-side Translator
LDNS: Local DNS proxy
LDNS64: Local DNS64 translator

XLAT translator: Impl. in Linux

1. Enter into hook
2. Match destination, packets not matched is routed as normal
3. Headers of the matched will be tanslated
4. TCP/UDP/ICMP is modified and port num is rewrited for PLAT
5. Inject the translated packets to network



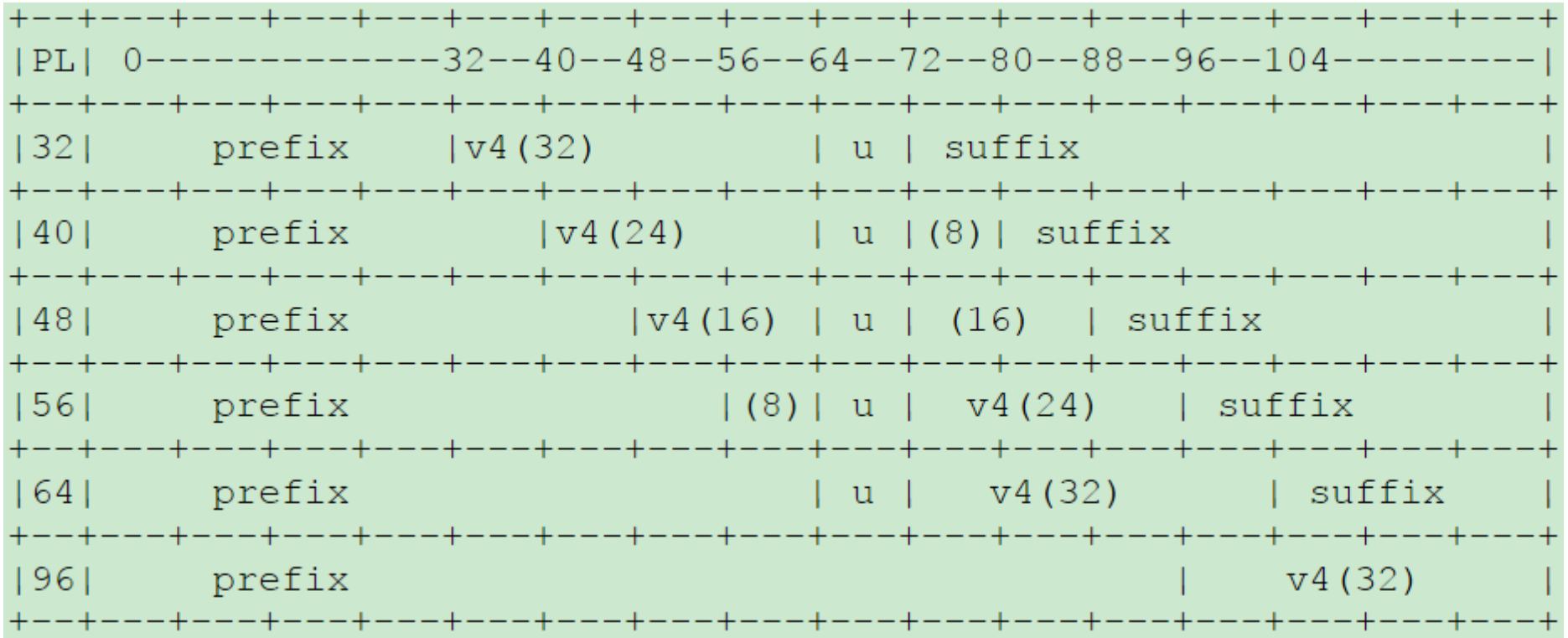
XLAT translator: mapping of header

IPv4 Header	Translated to IPv6
Version (0x4)	Version (0x6)
IHL	(discarded)
ToS	Traffic Class
Total Length	Payload Length (Total Length-IHL*4)
Identification	(discarded)
Flags	(discarded)
Offset	(discarded)
TTL	Hop Limit
Protocol	Next Header
Header Checksum	(discarded)
Source Address	stateful/stateless Mapping
Destination Address	stateful/stateless Mapping
Options	(discarded)

IPv6 Header	Translated to IPv4
Version (0x6)	Version (0x4)
Traffic Class	ToS
Flow Label	(discarded)
Payload Legnth	Total Length (Payload Length+20)
Next Header	Protocol
Hop Limit	TTL
Source Address	stateful/stateless Mapping
Destination Address	stateful/stateless Mapping
-	IHL (5)
-	Header Checksum (recalculate)

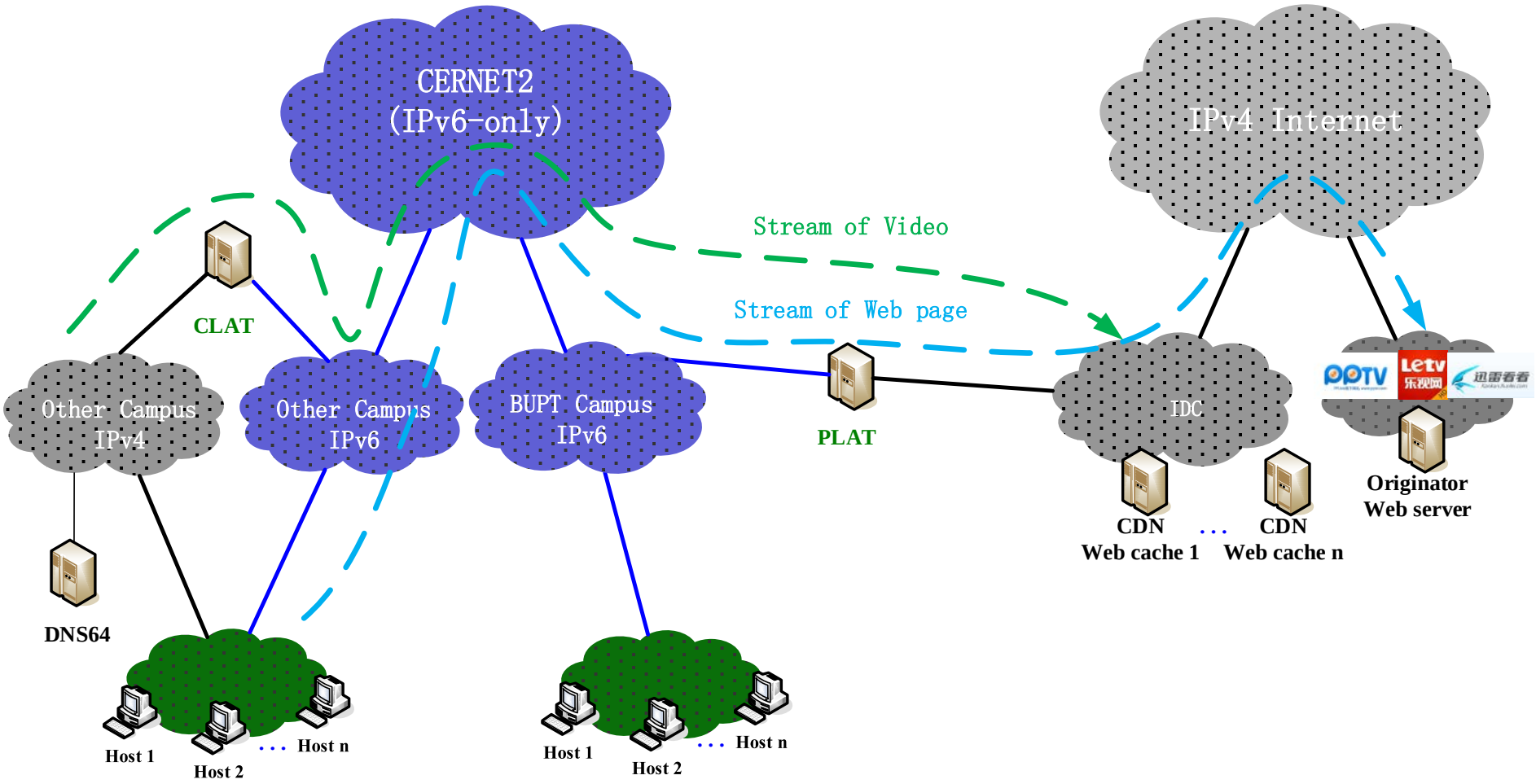
Conform to RFC6145
(IP/ICMP Translation Algorithm)

XLAT translator: mapping of address



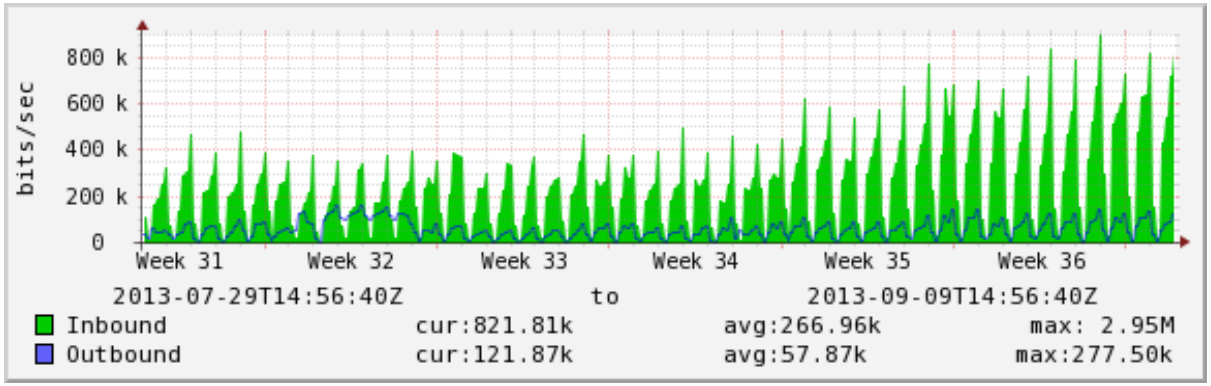
Conform to RFC6052
(IPv6 Addressing of IPv4/IPv6 Translators)

Test in BUPT

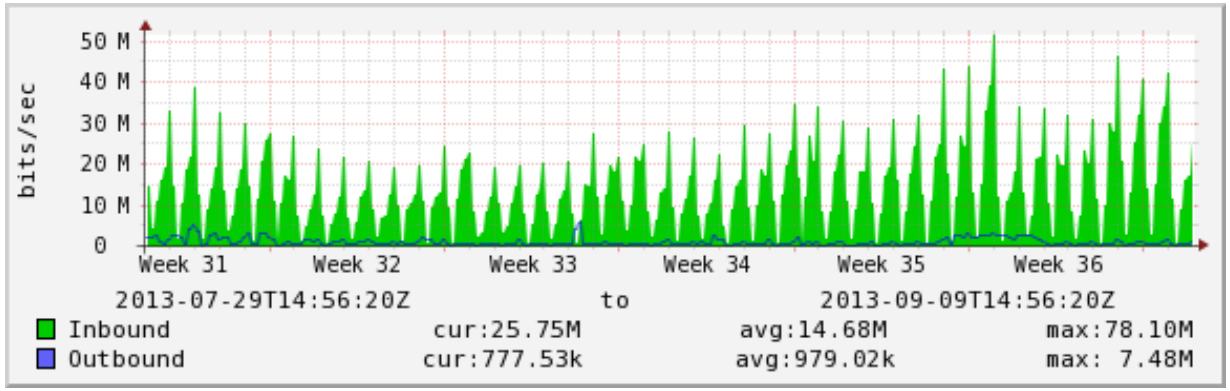

CERNET2
(IPv6-only)
IPv4 Internet
Stream of Video
Stream of Web page
CLAT
Other Campus
IPv4
Other Campus
IPv6
BUPT Campus
IPv6
PLAT
IDC
DNS64
Host 1
Host 2
Host n
Host 1
Host 2
Host n
CDN
Web cache 1
CDN
Web cache n
Originator
Web server

Traffic statistics

- Traffic of web page



- Traffic of video



Operating UI

```
[root@Centos nat64]# ./nat64-stop.sh
Unload module nf_nat64 successfully!
[root@Centos nat64]# ./nat64-start.sh
**************
nf_nat64 setup
**************
Virtual Netdevice Name is nat64

Info: Using 113.31.40.236/30 as the NAT64 IPv4 address.

modprobe -r nf_nat64
modprobe nf_nat64 nat64_ipv4_addr=113.31.40.236 nat64_i
len=64 ethname=nat64
ifconfig nat64 mtu 1500 up
ip route add 113.31.40.236/30 dev nat64
sysctl -w net.ipv4.conf.all.forwarding=1
net.ipv4.conf.all.forwarding = 1
sysctl -w net.ipv6.conf.all.forwarding=1
net.ipv6.conf.all.forwarding = 1
[root@Centos nat64]#
```

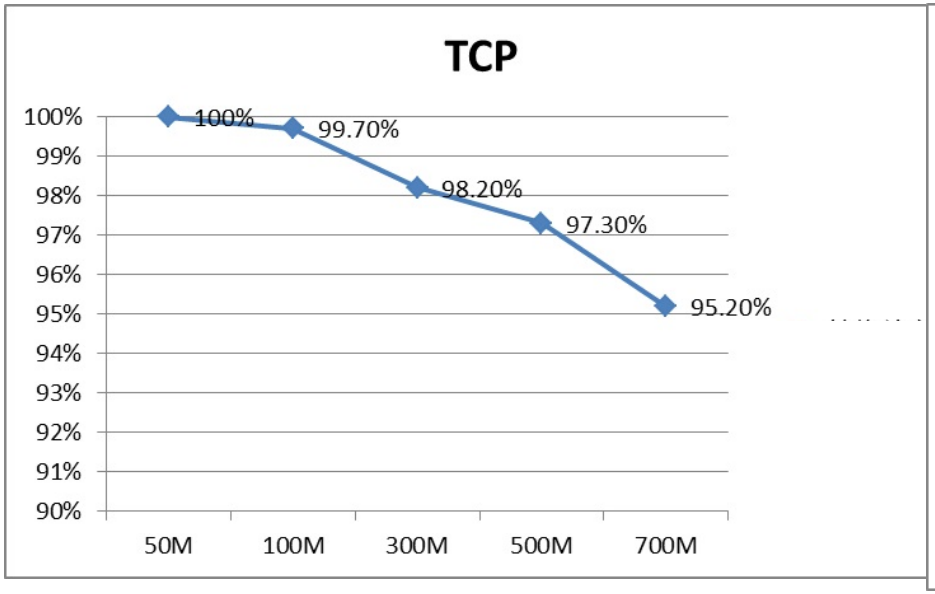
```
[root@nat64 ~]# nativi-stop
Unload module nf_nativi sucessfully!
[root@nat64 ~]# nativi-start
**************
nativi setup
**************
Translation1:
ipv4_addr:0.0.0.0/0
src_prefix:2401:aa00:2:2:2:2::/96
dst_prefix:2001:da8:202:a07:0:100::/96

WARN:Second translation will not be up because invalid variable!

modprobe -r nf_nativi
modprobe nf_nativi nativi_ipv4_addr=0.0.0.0 nativi_ipv4_mask=0 na
_addr=2401:aa00:2:2:2:2:: nativi_src_prefix_len=96 nativi_dst_pre
a8:202:a07:0:100:: nativi_dst_prefix_len=96
ifconfig nativi mtu 1500 up
ip -6 route add 2401:aa00:2:2:2:2::/96 dev nativi
sysctl -w net.ipv4.conf.all.forwarding=1
net.ipv4.conf.all.forwarding = 1
sysctl -w net.ipv6.conf.all.forwarding=1
net.ipv6.conf.all.forwarding = 1
[root@nat64 ~]#
```

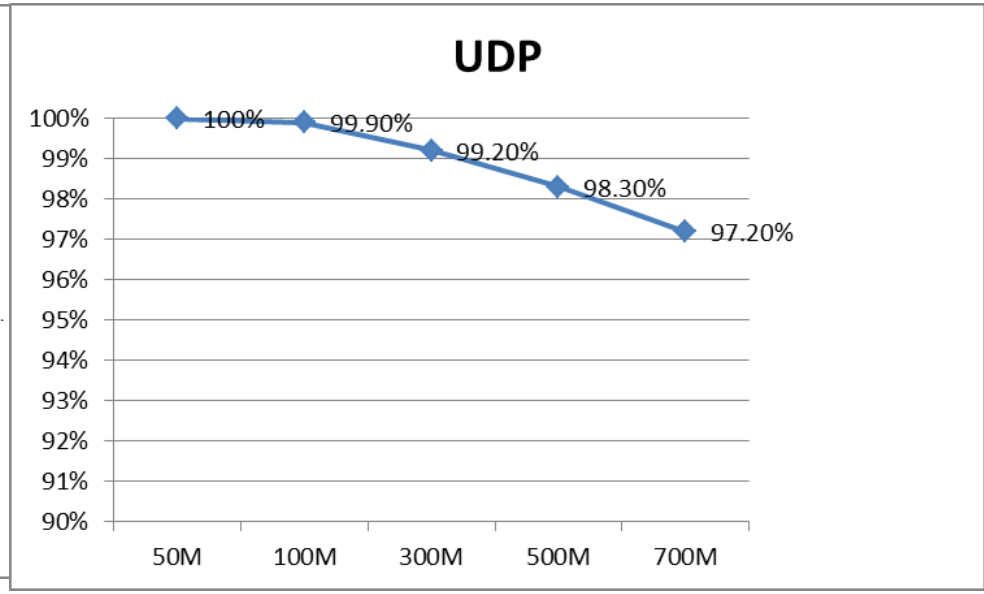
Online video stream

Translation Efficiency

- Packets are injected from CLAT to PLATusing iperf tool
- Test is on two virtual machines with 1GB memory
- Efficiency decrease when throughput reaching 1Gbps
- New translator based on DPDK is in progress

(Data Plane Development Kit)





* UDP is specified as1400 bytes length

Thanks!