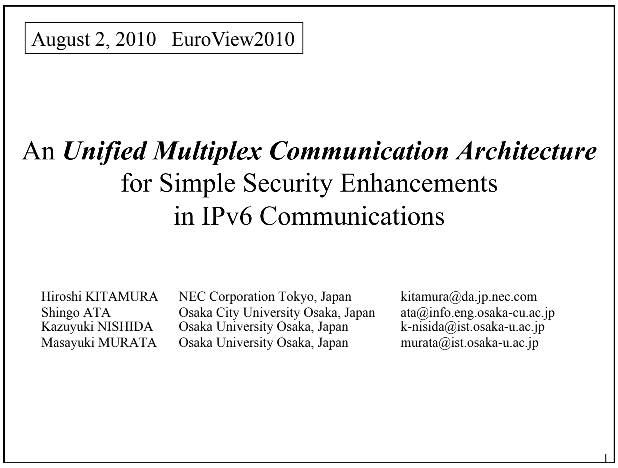
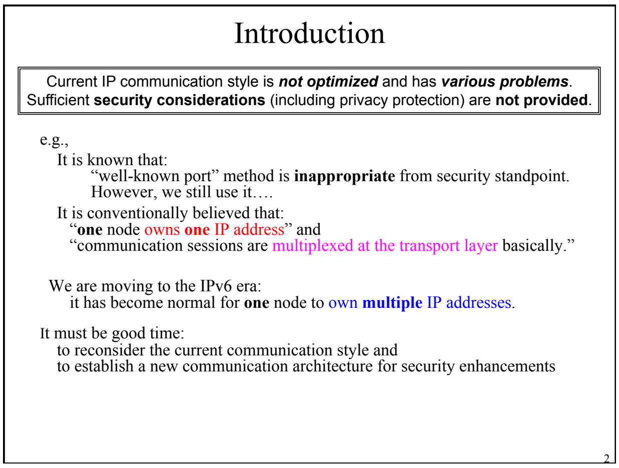


August 2, 2010 EuroView2010

An *Unified Multiplex Communication Architecture* for Simple Security Enhancements in IPv6 Communications

Hiroshi KITAMURA NEC Corporation Tokyo, Japan kitamura@da.jp.nec.com
Shingo ATA Osaka City University Osaka, Japan ata@info.eng.osaka-cu.ac.jp
Kazuyuki NISHIDA Osaka University Osaka, Japan k-nisida@ist.osaka-u.ac.jp
Masayuki MURATA Osaka University Osaka, Japan murata@ist.osaka-u.ac.jp



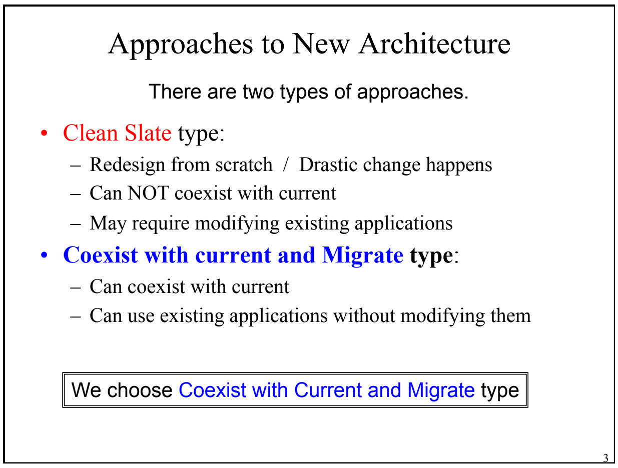
Introduction

Current IP communication style is ***not optimized*** and has ***various problems***.
Sufficient **security considerations** (including privacy protection) are **not provided**.

e.g.,
It is known that:
"well-known port" method is **inappropriate** from security standpoint.
However, we still use it….
It is conventionally believed that:
"**one** node owns **one** IP address" and
"communication sessions are multiplexed at the transport layer basically."

We are moving to the IPv6 era:
it has become normal for **one** node to own **multiple** IP addresses.

It must be good time:
to reconsider the current communication style and
to establish a new communication architecture for security enhancements

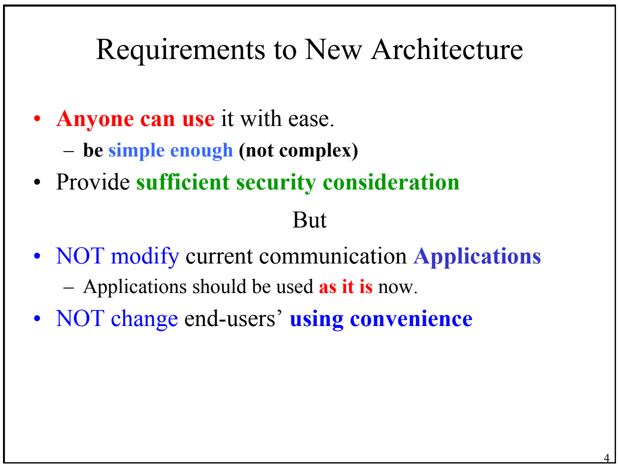


Approaches to New Architecture

There are two types of approaches.

- Clean Slate type:
  - Redesign from scratch / Drastic change happens
  - Can NOT coexist with current
  - May require modifying existing applications
- **Coexist with current and Migrate type**:
  - Can coexist with current
  - Can use existing applications without modifying them

We choose Coexist with Current and Migrate type

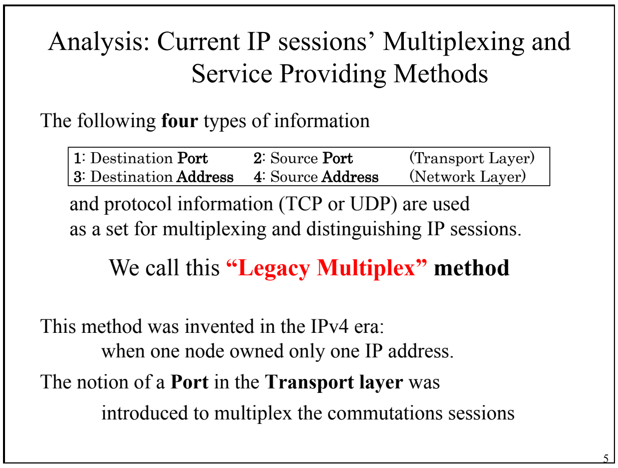


Requirements to New Architecture

- **Anyone can use** it with ease.
  - **be simple enough (not complex)**
- Provide **sufficient security consideration**

But

- NOT modify current communication **Applications**
  - Applications should be used **as it is** now.
- NOT change end-users' **using convenience**



Analysis: Current IP sessions' Multiplexing and Service Providing Methods

The following **four** types of information

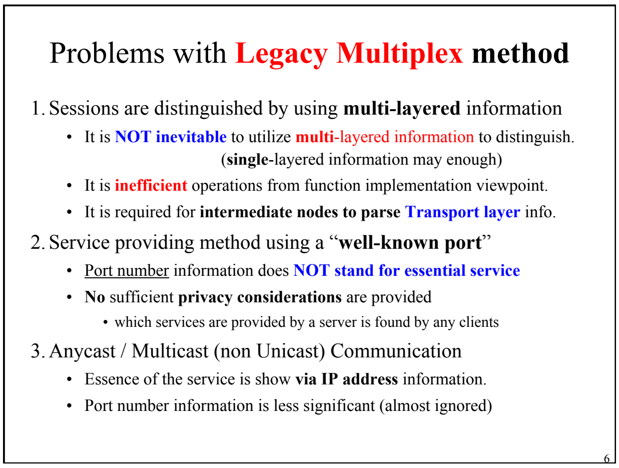
1: Destination **Port**	2: Source **Port**	(Transport Layer)
3: Destination **Address**	4: Source **Address**	(Network Layer)

and protocol information (TCP or UDP) are used
as a set for multiplexing and distinguishing IP sessions.

We call this **"Legacy Multiplex" method**

This method was invented in the IPv4 era:
when one node owned only one IP address.

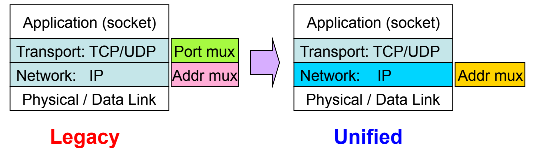
The notion of a **Port** in the **Transport layer** was
introduced to multiplex the commutations sessions



Problems with **Legacy Multiplex method**

1. Sessions are distinguished by using **multi-layered** information
   - It is **NOT inevitable** to utilize **multi**-layered information to distinguish.
     (**single**-layered information may enough)
   - It is **inefficient** operations from function implementation viewpoint.
   - It is required for **intermediate nodes to parse Transport layer** info.
2. Service providing method using a "**well-known port**"
   - Port number information does **NOT stand for essential service**
   - **No** sufficient **privacy considerations** are provided
     - which services are provided by a server is found by any clients
3. Anycast / Multicast (non Unicast) Communication
   - Essence of the service is show **via IP address** information.
   - Port number information is less significant (almost ignored)

Proposal: **Unified Multiplex** Communication Architecture as a Solution



Application (socket)
Transport: TCP/UDP | Port mux
Network: IP | Addr mux
Physical / Data Link

Application (socket)
Transport: TCP/UDP
Network: IP | Addr mux
Physical / Data Link

Legacy

Unified

- Sessions' multiplex / distinguish operations is simplified: can be done **only on the single Network Layer**
- Necessary information for the operations is simplified: **Destination and Source IP address information only**

1: Destination Address 2: Source Address (Network Layer)

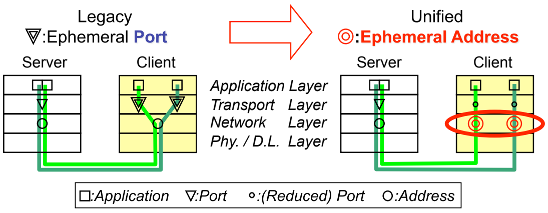
Two types of newly introduced Session Specific IP addresses

If sessions are different,
used addresses in the sessions are **different**.
Used Address becomes **specific for each session**.

- Client Side:
  - **EA (Ephemeral Address)**
- Server Side:
  - **SSA (Specific Service Address)**

Addresses are dynamically ***Generated*** and ***Released*** when their sessions are ***Started*** and ***Ended***.
Address valid time period is limited

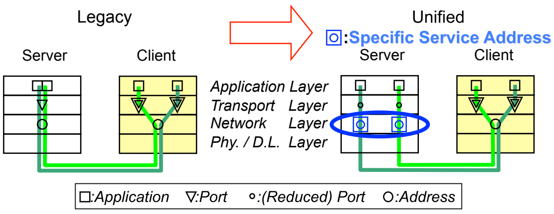
EA (Ephemeral Address) at Client Side from layer structure viewpoint



Legacy
▽:Ephemeral Port
Server
Client
Application Layer
Transport Layer
Network Layer
Phy. / D.L. Layer
Unified
◎:Ephemeral Address
Server
Client

□:Application ▽:Port ∘:(Reduced) Port O:Address

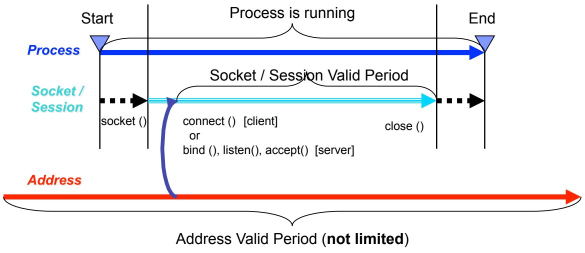
SSA (Specific Service Address) at Server Side from layer structure viewpoint



Legacy
Server
Client
Application Layer
Transport Layer
Network Layer
Phy. / D.L. Layer
Unified
⊡:Specific Service Address
Server
Client

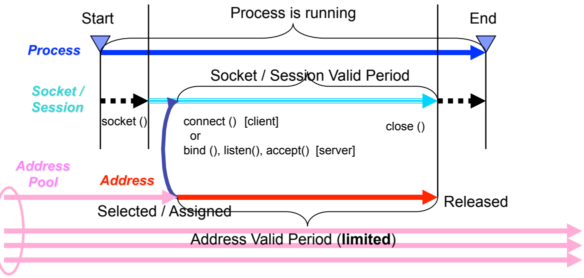
□:Application ▽:Port ∘:(Reduced) Port O:Address

Relationship between Process and Address Legacy Environment



Start
Process is running
End
Process
Socket / Session
Socket / Session Valid Period
socket ()
connect () [client]
or
bind (), listen(), accept() [server]
close ()
Address
Address Valid Period (**not limited**)

Relationship between Process and Specific Address Unified Multiplex Environment



Start
Process is running
End
Process
Socket / Session
Socket / Session Valid Period
socket ()
connect () [client]
or
bind (), listen(), accept() [server]
close ()
Address Pool
Address
Selected / Assigned
Released
Address Valid Period (**limited**)

Address Pool function is introduced

Improvements in Address Usages, Service Providing Methods etc.

1 Node-1 Fixed Address ⇒ **1 Node - Multi-Floating Address**

	Legacy	(Proposed) **Unified**
Number of Used Addresses	Use Only **One** Address (Basically)	Use **Multiple** Addresses
Information Dealing	General and **Share** Use **Same** Address	**Specific and Dedicated** Use **Different** Address
Service (on Servers)	Wait for **Anytime** (24hour / 365days)	Wait for **Only When** Access Expected to Come
Information Fluidity	**Fixed** (Not Changed)	**Floating (Changed** and **Updated)**

Quantitative Analysis: "Meet Again" Probability for the **same** Address

Condition:
Ephemeral Address Creation/Selection Rule is:
"***At Random***" from 64bit Interface ID space.

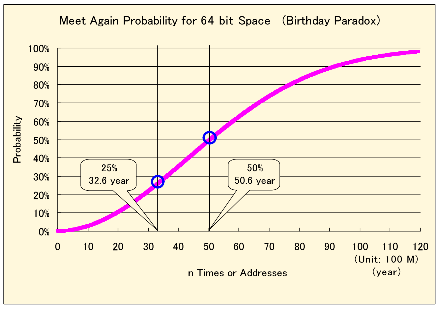
Probability Formula (Birthday Paradox):
"***n***" times probability:
$= 1 - (2^{64}-1)/2^{64} * (2^{64}-2)/2^{64} * \ldots * (2^{64}-n)/2^{64}$

Estimation: Number of ***consumed addresses*** per (year, day, hour, min, sec)

/ year	/ day	/ hour	/ min	/ sec
31,536,000	86,400	3,600	60	1.0
100,000,000	273,973	11,416	190	3.2

"100M addr. / year" is much enough (***sufficient estimation***)

"Meet Again" Probability Results for the **same** IP Address



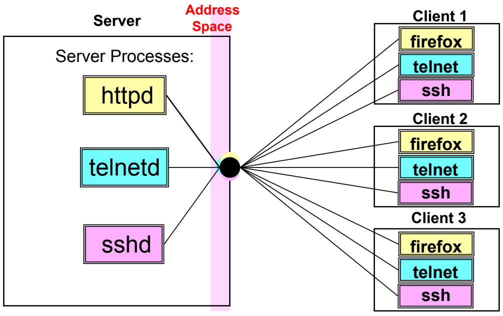
Consume 100M addr. / year (274k addr./day : 3.2 addr./sec)
10years: 2.8% 20years: 10.3%
25%: 32.6 years **50%: 50.6 years** 75%: 71.6 years

Characteristics of Specific Addresses (EA and SSA) introduced in the Unified Multiplex

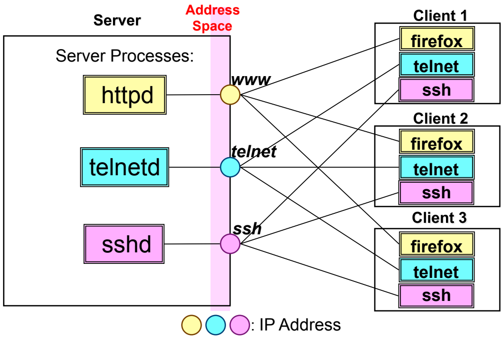
- Client side: **EA (Ephemeral Address)**
  - Any users can use **Ephemeral Address** easily
  - It is not necessary for users to be conscious the existence of **Ephemeral Address** function (like Ephemeral Port).
    Very low threshold to deploy and use this
- Server side: **SSA (Specific Service Address)**
  - Completely newly invented functions and **very unique**
  - No analogical functions can be found in the Legacy method
    Further researches are required to fully utilize this

Hereafter, SSA issues are disccused

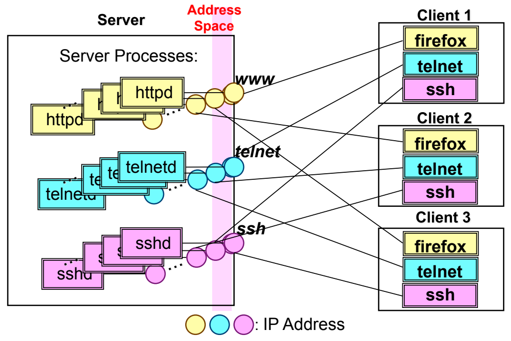
IP Address Using Style: **Legacy** Multiplex Environment



IP Address Using Style: **Transient** Environment until SSA



IP Address Using Style:
Unified Multiplex Environment with SSA



Implementation and Verification Status

Unified Multiplex Communication Architecture
functions have been implemented on the
followings.

- FreeBSD 6.2R FreeBSD 8.0R
- Linux kernel 2.6.24 (implemented functions are limited)

➢Without modifications of communication Applications:
➢Only **with** the **Kernel replacement**:

It has verified that
basic functions work correctly as they are designed

Conclusion

We have proposed:
an new communication architecture “**Unified Multiplex**”
and new address types (**EA** and **SSA**).
– This can coexist with current communication style.
– Anyone can use this with ease.

It have been proved:
this is an ***advanced communication architecture***
that can provide ***sufficient security consideration***.
– No fatal problems have not observed until now.
– Veiled problems may be remained

We will continue refining the design and implementation
and evaluating the architecture by utilizing
its functions on various communication applications.