2010/06/11 戦略的情報通信研究開発推進制度（SCOPE）第6回成果発表会

セッション多重化技術の刷新により安心・安全な通信を実現するIPv6 Unified Multiplex通信アーキテクチャの研究開発 (08063984)

研究代表者
北村 浩　　日本電気株式会社
研究分担者
阿多 信吾　大阪市立大学 大学院工学研究科
村田 正幸　大阪大学 大学院情報科学研究科

はじめに

既存の通信アーキテクチャはセキュリティなどの視点を
中心に綻びや歪みが見えている
今後のインターネットの持続的な発展を鑑みると
見直す時期にきている

- 誰もが簡単に使え
- セキュリティに十分に配慮できること

でも、

- 今使っている通信アプリケーションはそのまま使え
- エンドユーザの使い勝手を変えないこと

Unified Multiplex 通信アーキテクチャ

では、何をするか

- 現状のIP通信のセッション/エンドポイントの**多重化・識別**に**「address情報+port情報」**を用いるという
  通信アーキテクチャを根本的に見直す
- **port情報**の扱いを大きく変更　⇒　**不要へ**
- IPv6 の広いアドレス空間を活用することにより実現される
  “**Unified Multiplex 通信アーキテクチャ**”
  新しい通信スタイルの提案
  - 実現モデル
  - 既存通信環境との共存と新環境への移行

- 提案機構で問題を総合的に解決
  - シンプルで効率の高い機能実装
  - プライバシーを中心とした高いセキュリティ機能を提供

IP通信のセッションの多重化・識別と サービス情報の提供方法 に関する問題

- 既存の多重化・識別方式　(**Legacy Multiplex方式**)
  - **2つ** の層、**4つ** の情報を用いる
    (情報のひとつでも異なれば、
    異なるセッション)

	宛先	送信元
Transport層	Dst. Port	Src. Port
Network層	Dst. Addr	Src. Addr

  - 1ノードが、**1つ** のIP addressしか持たない時代の発想
    トランスポート層でport概念を導入して多重化
  - 機能実装の効率の面、セキュリティの観点などで問題

- 提案する多重化・識別方式 (**Unified Multiplex方式**)
  - **1つ** の層、**2つ** の情報をだけで実現

	宛先	送信元
Network層	Dst. Addr	Src. Addr

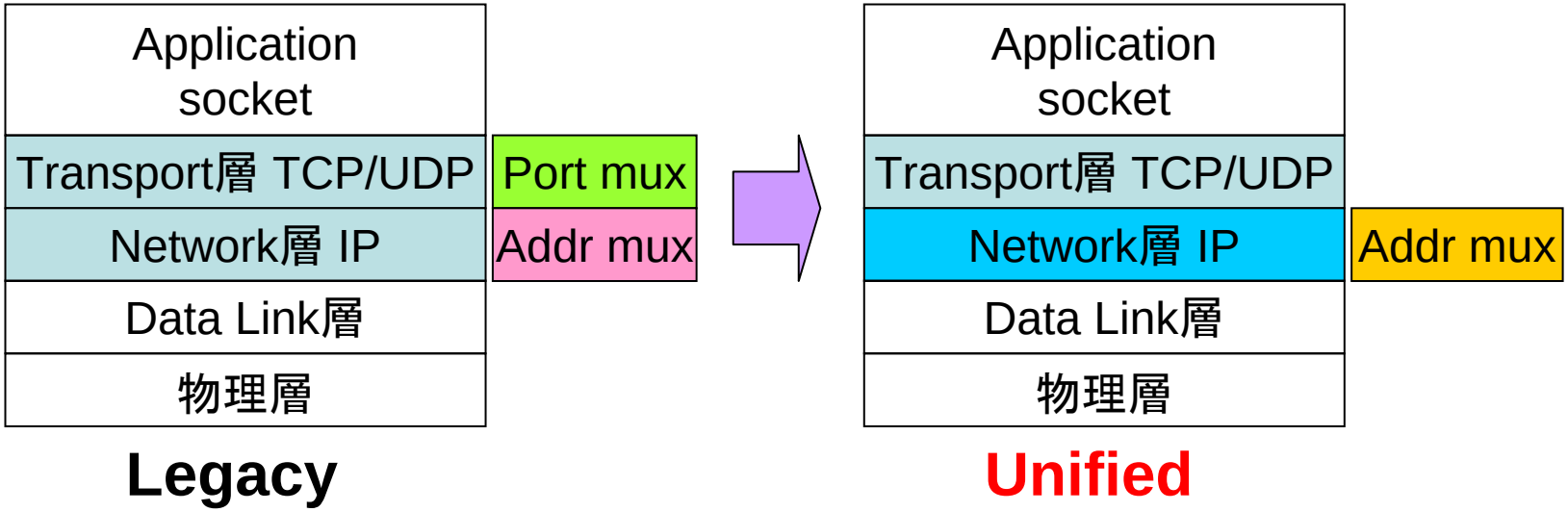
  - 1ノードが、**複数** のIP address を持つ**IPv6時代の発想**
  - 機能実装の効率の面、セキュリティの観点などの問題を解決

セッションの多重化・識別方式を根本的に見直す時が来ている

既存の多重化・識別方式 (Legacy) の問題

1. 複数層（L3,L4）にわたる情報を使った識別
   識別のために4つの情報を用いることは必然的ではなく
   機能実装の効率などに関わる問題がある
2. well-known port によるサービス情報提供
   port 情報はサービスの本質を代表していない
   性善説に基づき、プライバシーへの配慮などが不十分
3. Anycast / Multicast 通信でのサービスの本質
   サービスの本質は IP address 情報を通して示されている
   重要なのは IP address 情報のみで、port番号は考慮の対象外
4. ICMP通信の多重化とサービス指向の通信

Unified Multiplex 通信アーキテクチャの目指すセッションの多重化・識別の変化



- 多重化・識別処理は
  Network層だけで行う処理だけで 十分にできるはず
- それに必要な情報は
  送信元と宛先の IP アドレスだけ のシンプルにできる

Unified Multiplex 通信アーキテクチャ の 基本的考え方

- 既存の通信環境との 共存は極めて重要
  - 共存できないような不連続なジャンプは現実的でない
  - エンドユーザの使い勝手は変えない

- 実現する上での基本的な考え方
  - Legacy な環境と共存できる
  - Unified を用いれば、プラスアルファとしてご利益が得れる

  - ユーザランド の通信アプリケーションは変更しない
  - **カーネル (OS)** を入れ替えるだけで対応

1ノード-1固定アドレス方式
というIPアドレスの利用形態に対する先入観

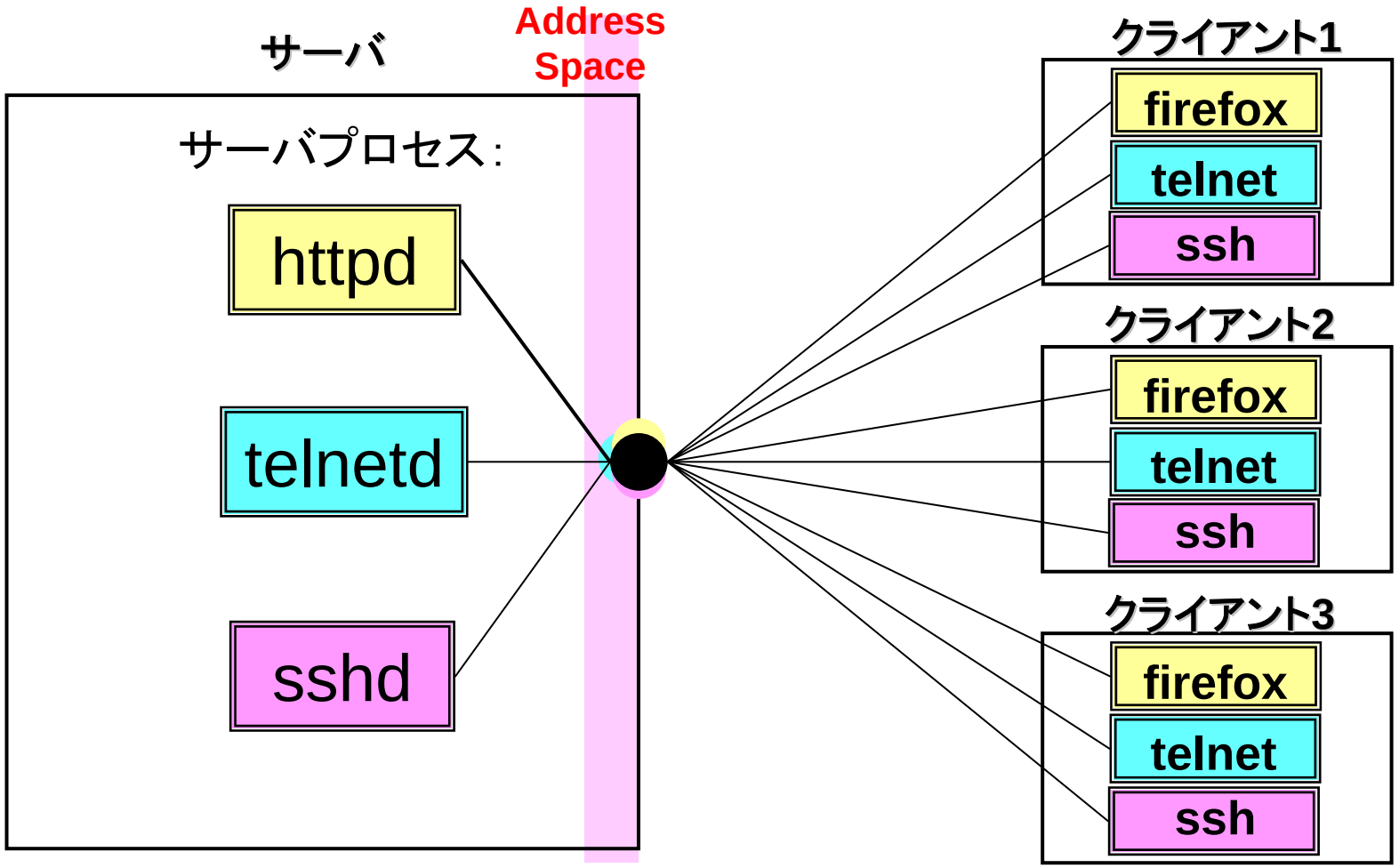
既存の Legacy 通信アーキテクチャでは

- 1つのIPアドレスを設定
  - 1つの通信ノード（一つのインターフェース）に基本的に1つのIPアドレスを割り振る
- 固定IPアドレス
  - 一旦設定されたIPアドレスはその後変化することがない固定的なもの

従来の視点からすると極めて自然
最小のIPアドレス消費で、強固な固定概念

1ノード-1固定アドレス方式というやり方を
疑うところから この研究は始まる

Legacy 環境（1ノード-1固定アドレス方式）での IPアドレス利用形態



セキュリティの視点からして、この形態のままでいいのか？

1ノード-1固定アドレス方式 の問題

高いセキュリティ保護が求められる、現状からすると
1ノード-1固定アドレス方式には多くの問題がある

1. 役割の共有/兼務に関する問題
   IPアドレスの果す役割や目的は多種多様
   でも、一つしかIPアドレスがないので、全ての処理を兼務・共有
   当然 無理が発生している状態
   潜在的に多くの問題が隠れている
2. 固定であることの問題
   一旦受けたアタックを根本的に回避するには、IPアドレス変更が不可避
   固定のままでは 回避できない
3. IPアドレスの状態と有効期間
   IPv4 には アドレス状態やそれが遷移するとか有効期限の概念がない
   1ノード-1固定アドレス方式ではこれで良かったのかもしれないが
   基本的な機能が欠如している状態

提唱するUnified Multiplex通信アーキテクチャの基本理念と1ノード-複数変動アドレス方式

1ノード-複数変動アドレス方式を基本的な原理として採用

Legacy ⇒ **Unified** が意味するのは

IPアドレスの利用形態の変化

★ “固定 ⇒ 変動へ”
★ “兼用 ⇒ 専用へ” or “共有 ⇒ 専有へ”

Unified Multiplex通信アーキテクチャが目指し実現する変化

1ノード-1固定アドレス方式　⇒　1ノード-複数変動アドレス方式

	Legacy方式	Unified方式
利用・設定 アドレス数	1つしかない	多数を設定利用
共有/専有	兼用・共有利用 皆同じものを使う	専用・専有利用 役割・目的毎に変える
サービス 提供方法	常時待機	必要時のみ待機
情報 固定/変動	固定で変化しない	更新され変動する

Brute Forceの効かない広大なアドレス空間と“教えない情報”を利用するセキュリティ

- Unified Multiplexで使っている主なセキュリティ技術は
  いわば“**教えない情報**”を利用したセキュリティ
- 携帯電話の迷惑電話対策で用いている:
  “*信用できる人にしか電話番号を教えない*” という方法と同じ
- 番号情報さえ教えなければ、
  迷惑電話はかかってこない確実な方法
- 誰もがその原理を理解しており、
  容易かつ実感として分かる状態で利用できる方法
- Unified Multiplexでもこの特長を引き継いでいる

IPv6 の 2^{128}というBrute Forceの効かない
広大なアドレス空間が成せる 使えるセキュリティ機能

アドレス空間の広さの定量的分析
同一アドレスの再選択確率

- 広大な空間でも、定量的な分析は必要
  - 一度選んだアドレスはどれくらいの時間をかけると、再度選択するかを分析
- ランダム選択なら、Birthday Paradox という逆説が知られている
- 確率 算出の式:

  64bit（ユーザ裁量アドレス）空間を対象、**ランダム**選択する場合
  一度選択したアドレスと同じアドレスを***n*回目に選択する確率**:

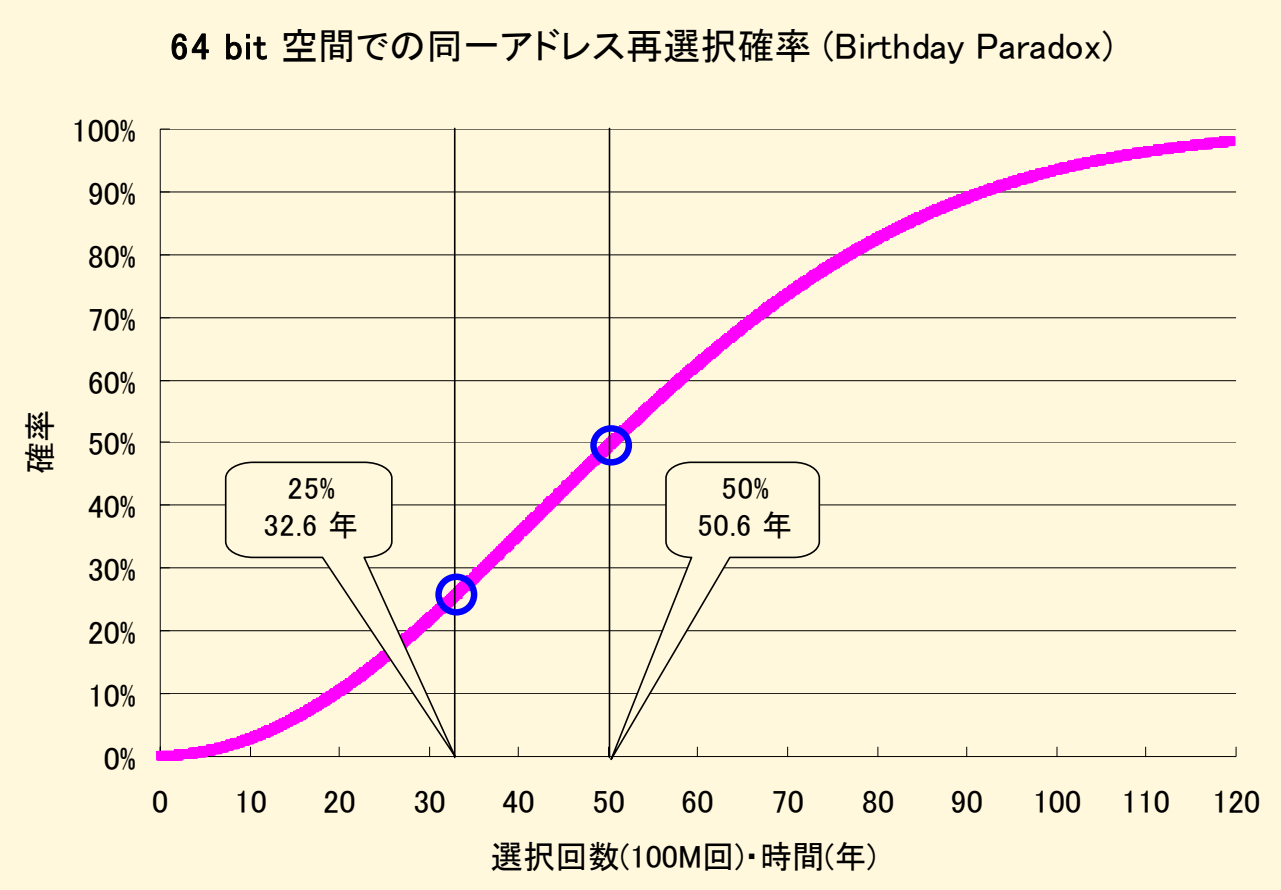
  $$= 1 - (2^{64}-1)/2^{64} \times (2^{64}-2)/2^{64} \times \ldots \times (2^{64}-n)/2^{64}$$

- 回数情報を 時間情報として把握するための変換:
  単位時間あたりの、アドレス消費数の見積り

	/ 年	/ 日	/ 時	/ 分	/ 秒
Case A	31,536,000	86,400	3,600	60	1.0
Case B	100,000,000	273,973	11,416	190	3.2

Case B:（1年に100M個のアドレス消費）は 事実上消費不可能なくらい大量

同一アドレスの再選択確率の計算結果



1年に100M個（1秒に3.2個）のペースで アドレスを消費するとして、

再選択の確率は　10年で2.8%、20年で10.3%　しかない

25%の確率に達するには32.6年、50%に達するには 50.6 年かかる

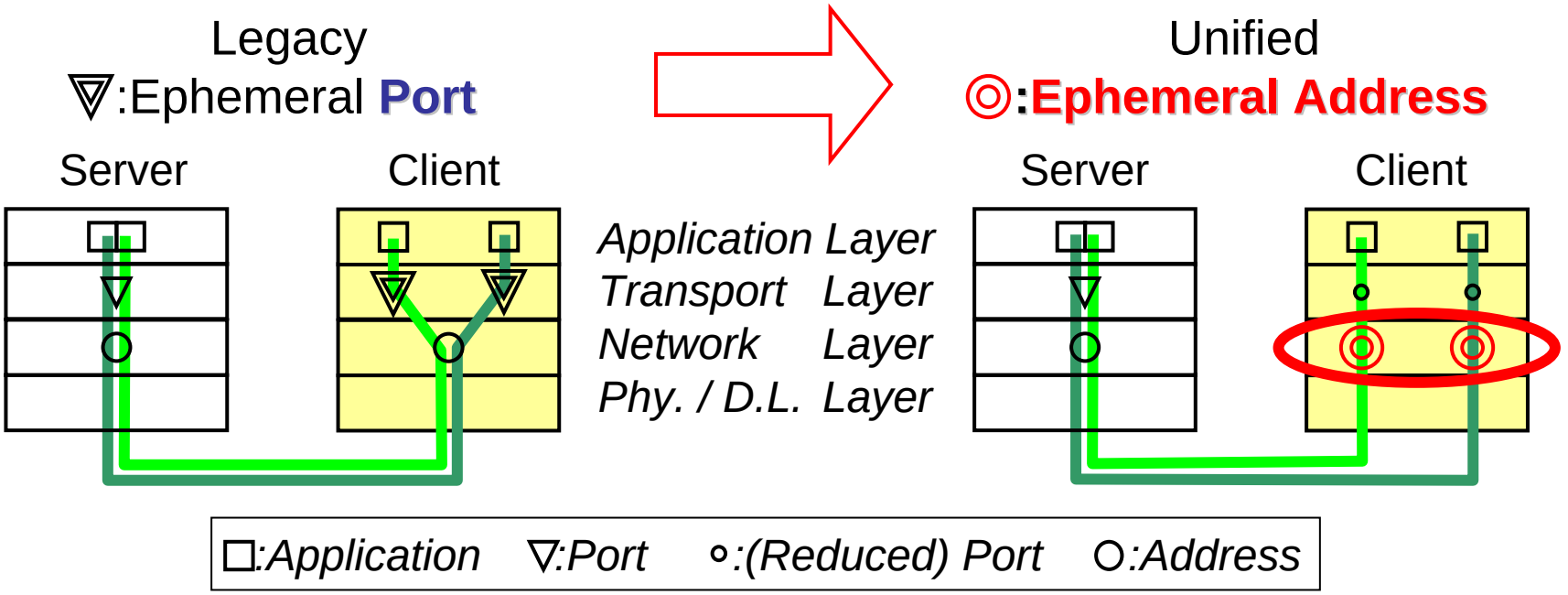
全く問題にならないくらい、64bit の空間は十分広い

Unified Multiplex 通信アーキテクチャで導入するセッション毎の専用IPアドレス

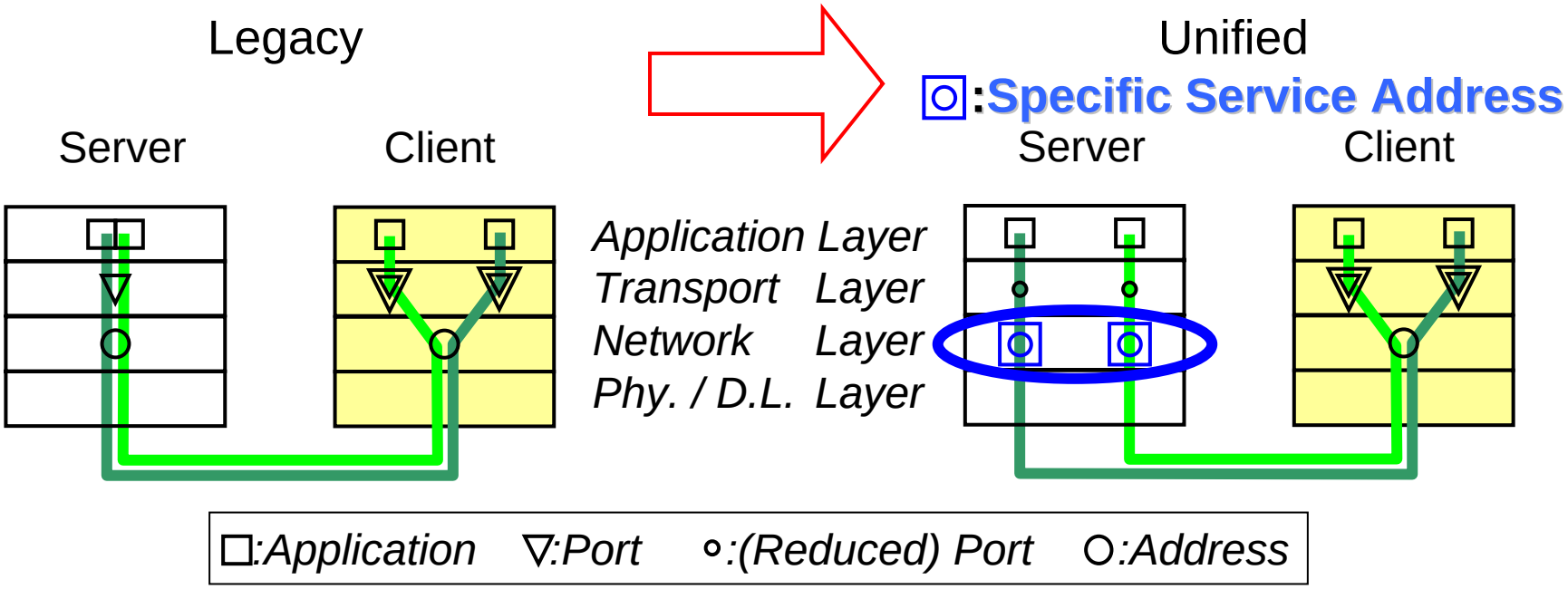
それぞれのセッション専用となる
2種類の新しいタイプのアドレスを導入

- クライアント側:
  - **EA (Ephemeral Address)**
- サーバ側 :
  - **SSA (Specific Service Address)**

レイヤ構造視点からみる
クライアント側での EA (Ephemeral Address)の導入



レイヤ構造視点からみる
サーバ側での SSA (Specific Service Address)の導入



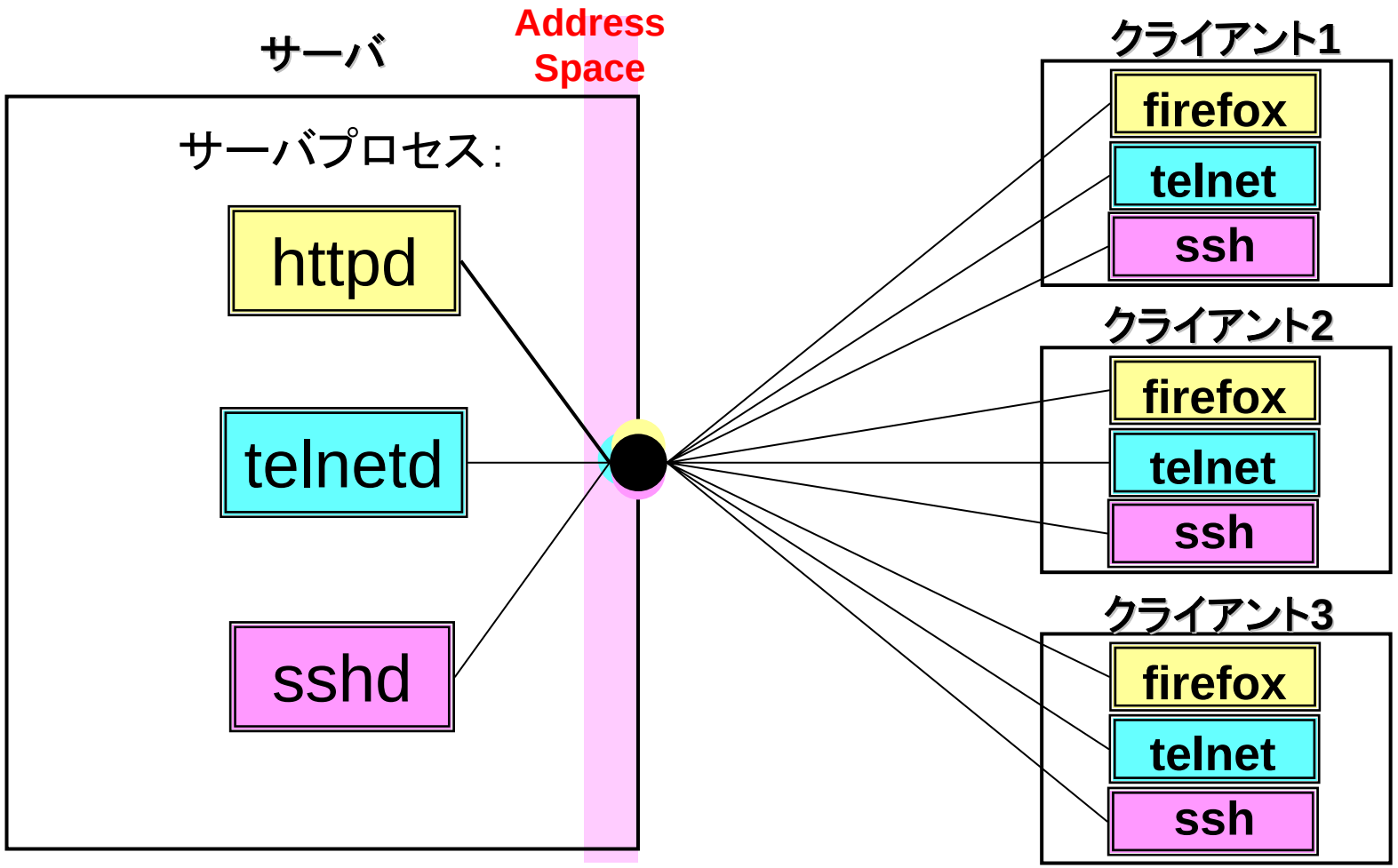
Unified Multiplex 通信アーキテクチャで導入するセッション毎の専用IPアドレスの特性

- クライアント側: **EA (Ephemeral Address)**
  - Ephemeral Port と同じように、
    クライアントアプリケーションがサーバとのセッションを開設する際に、
    カーネルが動的に割当て使用　セッションの終了と共に開放
  - ユーザが意識せずとも使え、**導入障壁は極めて低い**
- サーバ側：**SSA (Specific Service Address)**
  - Legacy方式では類例がない**独創性に富む方法**
  - Unified 方式でのみ実現される
  - 提供するサービスのセッション毎に専用となるSSA を割当て使用

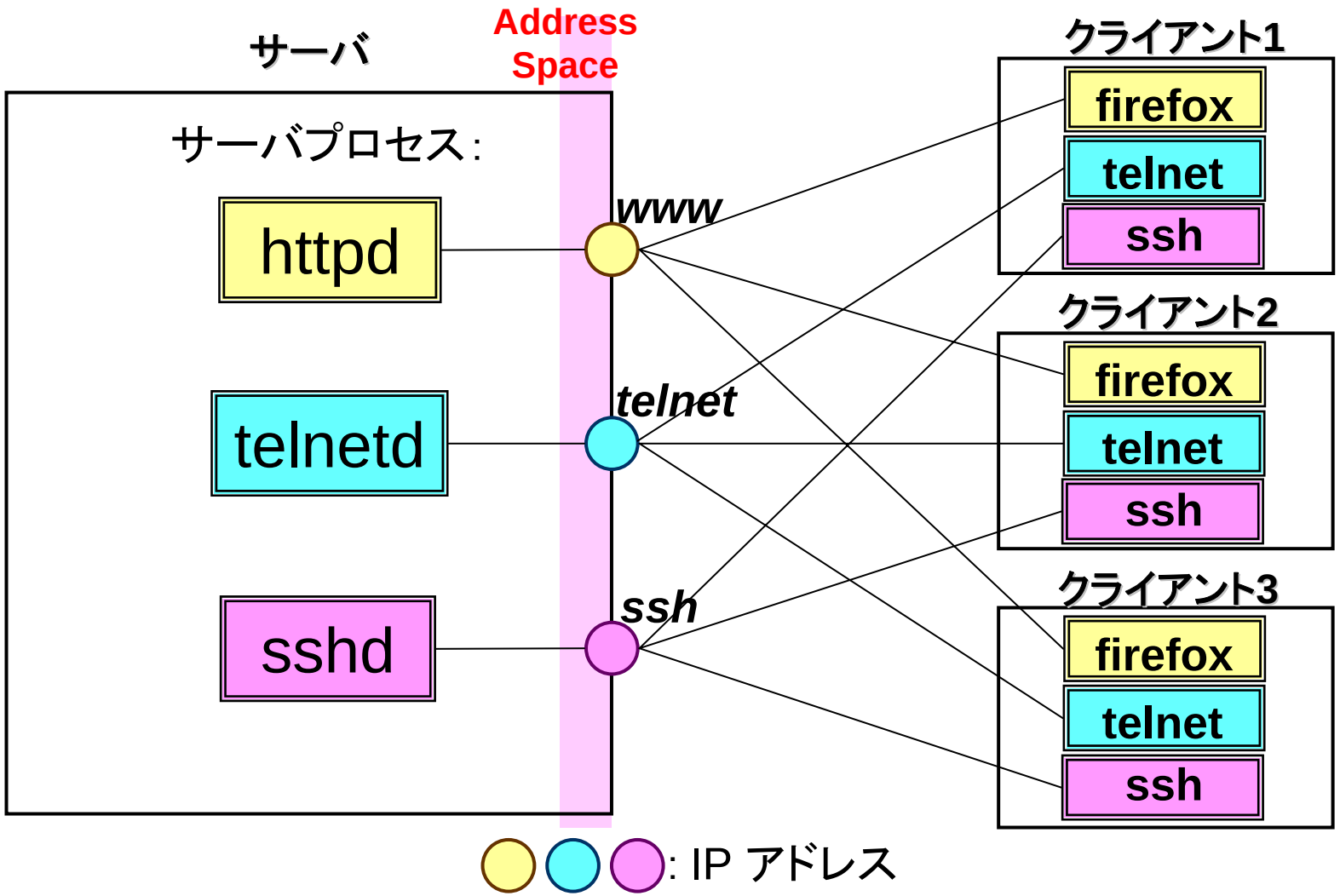
以降は SSAの利用を中心に説明

1. Legacyからの進化過程を対比的に
2. 専用のアドレスとはどの粒度に対して用いるか

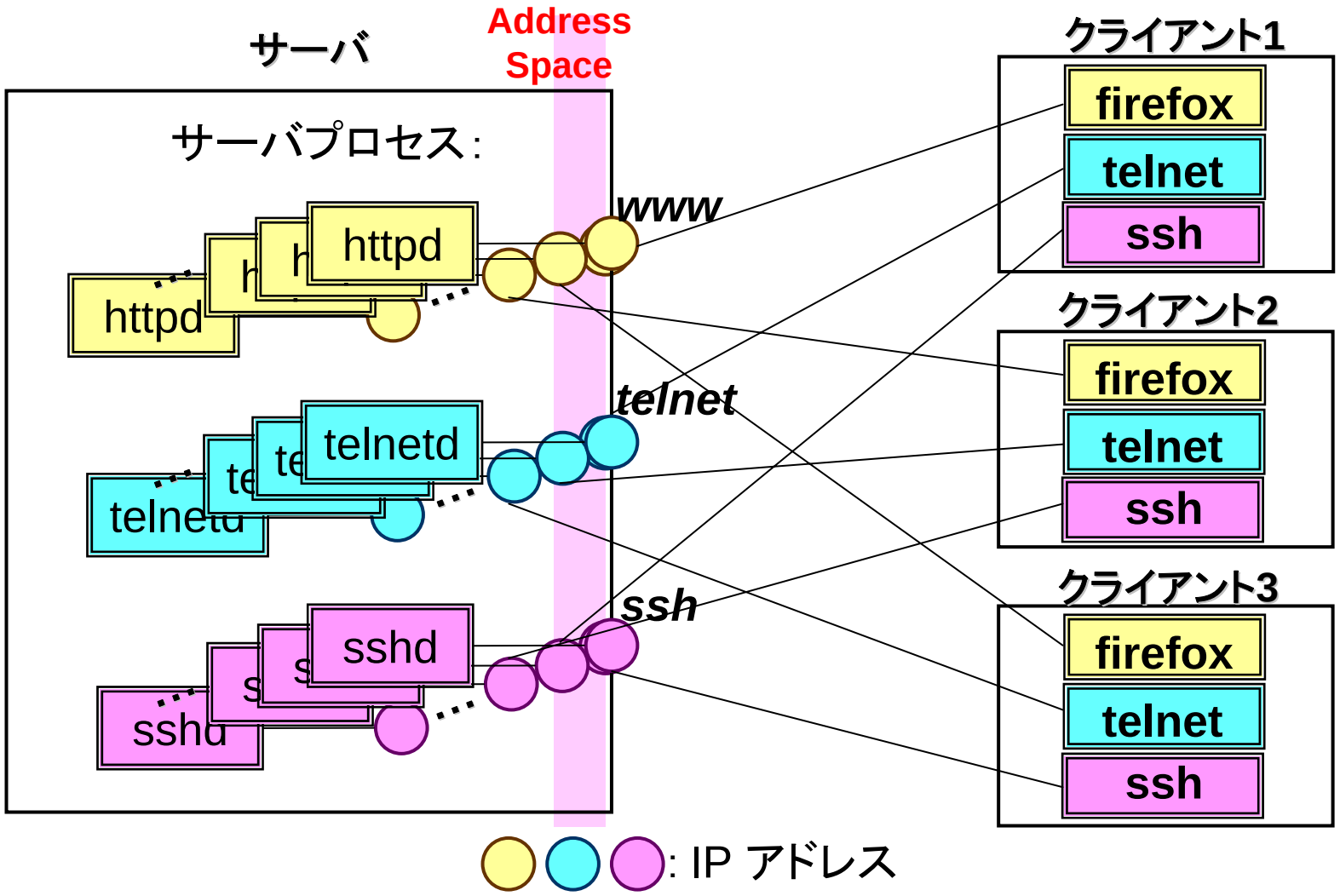
Legacy 環境でのIPアドレス利用形態



SSAに至る前の過渡的な利用形態



Unified MultiplexにおけるSSAの利用形態

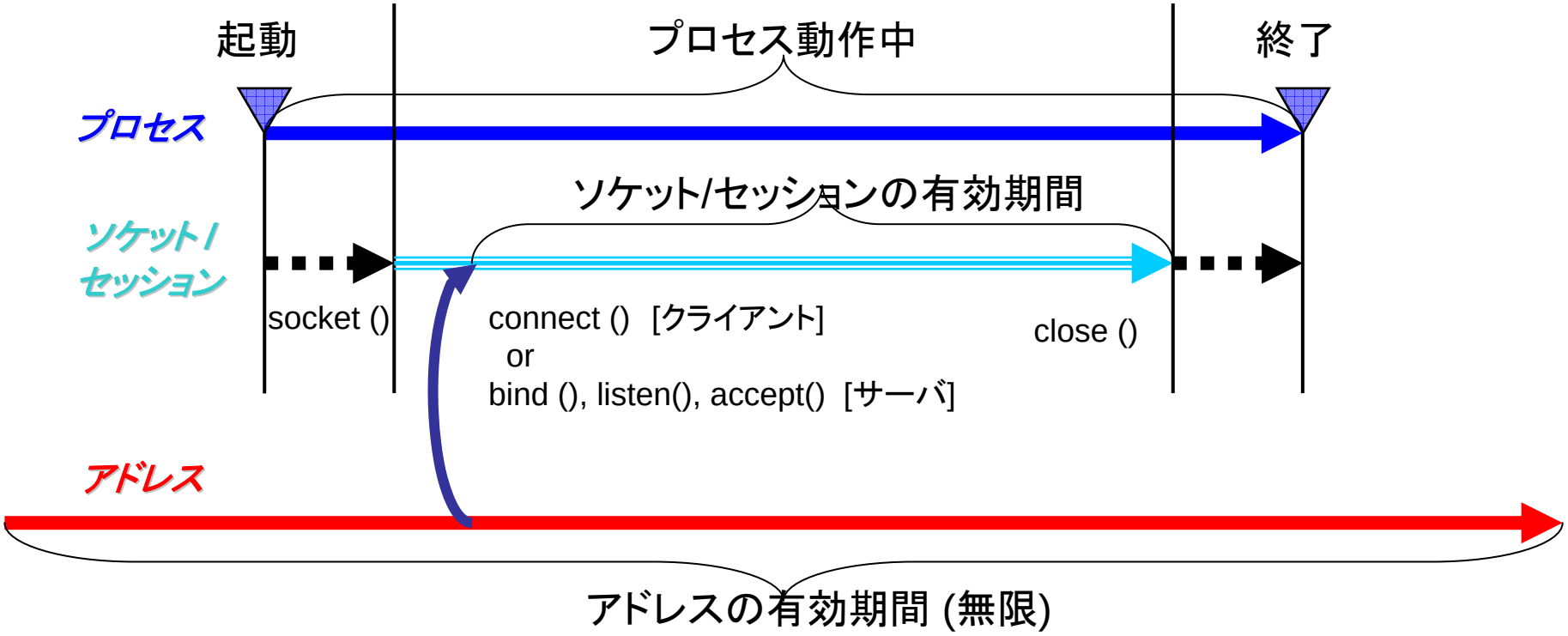


通信プロセスと専用IPアドレスの関係 1/4

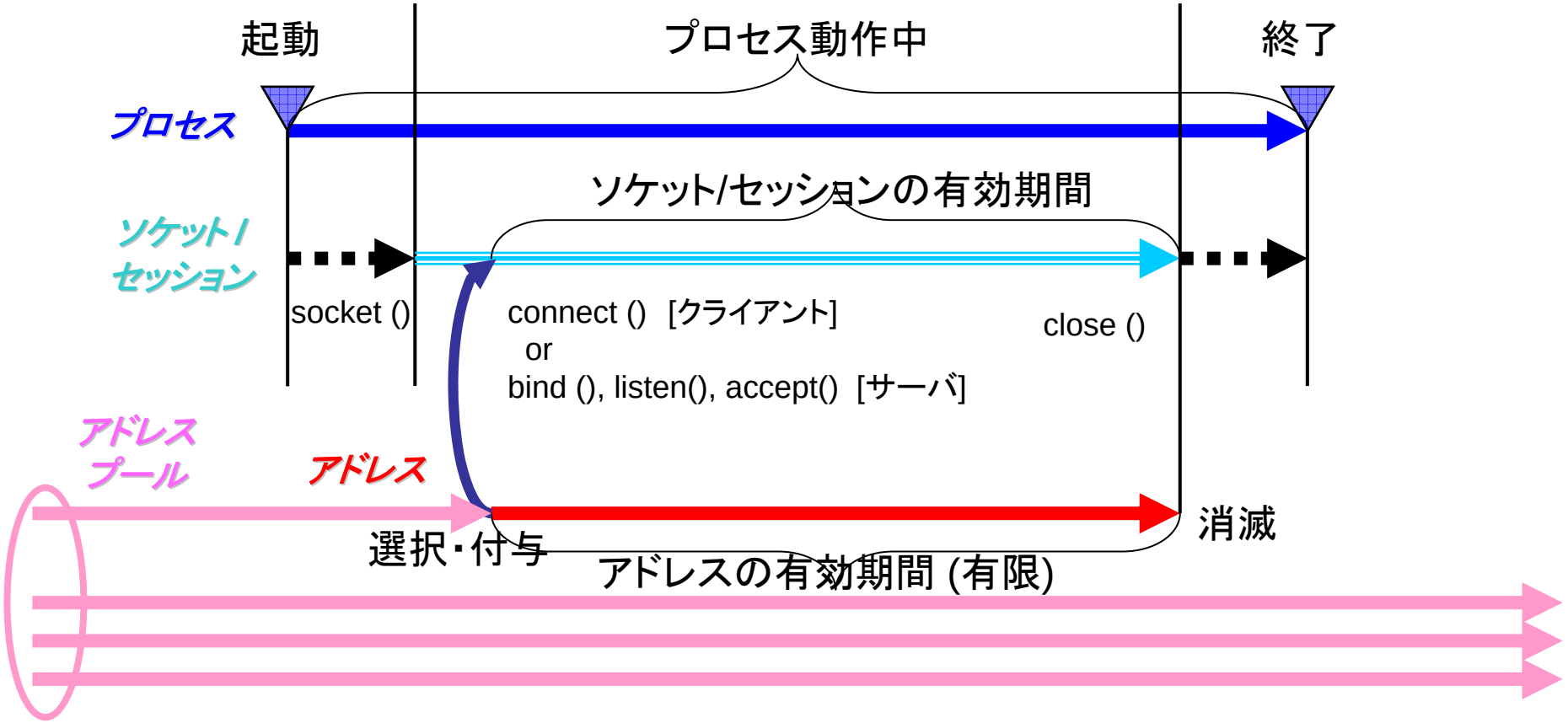
- 通信セッションの有効期間:
  - 通信プロセスにおける関数の呼び出し手順で制御
- TCPでの典型的手順と有効期間
  - 有効期間の開始
    - クライアント: connect()を呼んだ時
    - サーバ:　　bind()、listen()、accept()を呼んだ時
  - 有効期間の終了
    - クライアントでもサーバでもclose()を呼んだ時
- 通信セッションで用いる IPアドレスの選択
  - 通信プロセス指定と連係して動作するカーネルで選択
- Unified のアドレス（EA、SSA）の有効期間の制御
  - **セッションの有効期間とアドレスの有効時間を同じに**
  - セッション**開始時に有効化** / セッション**終了時に無効化**

既存のアプリケーションを変更せずに、
カーネルへの新たな機能の追加のみで実現

通信プロセスと専用IPアドレスの関係 2/4
Legacy環境でのアドレスとセッションの関係

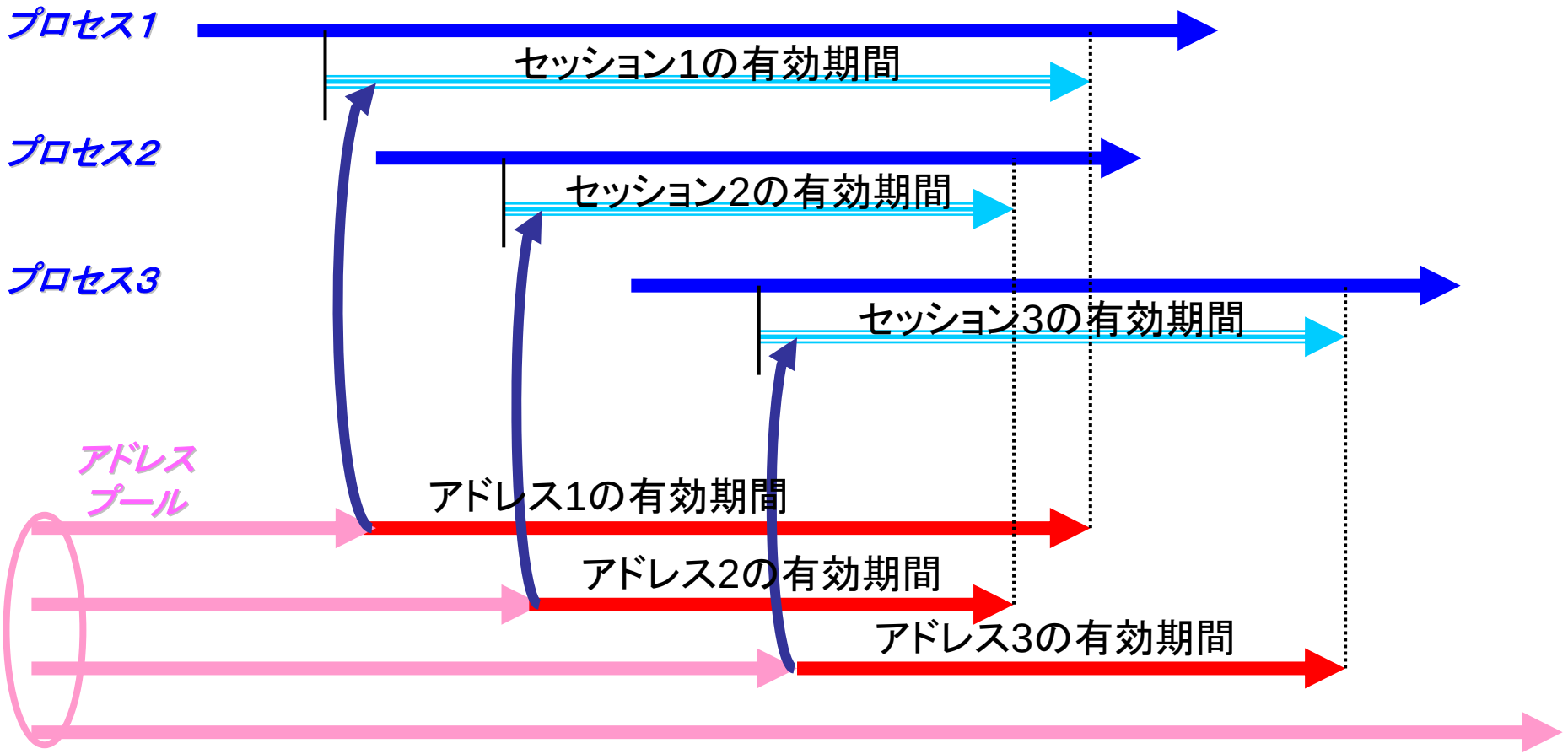


通信プロセスと専用IPアドレスの関係 3/4
Unified環境でのアドレスとセッションの関係



アドレスプール（すぐに有効にできるアドレスを貯める）を導入

通信プロセスと専用IPアドレスの関係 4/4
EA or SSA の有効期間とセッションの関係



サーバ側のSSA 実現のための新機能 1/2
解決すべき課題

1. 接続待ち受け方法
   - Legacy: サーバでは “多重同時接続待ち受け”
   - Unified: SSAの専用アドレスという概念にそぐわない
2. 接続待ち受けに用いるIPアドレスの決定タイミング
   - Legacy: サーバプロセスがlisten()を
     呼出前に、どのIPアドレス用いるかが決まっている
   - Unified: SSAの場合　提供サービスの詳細は、
     どのクライアントが接続するかが決まってから明確になる
     つまり、listen()を呼出す前に、
     用いるIPアドレスが確定していないのが一般的

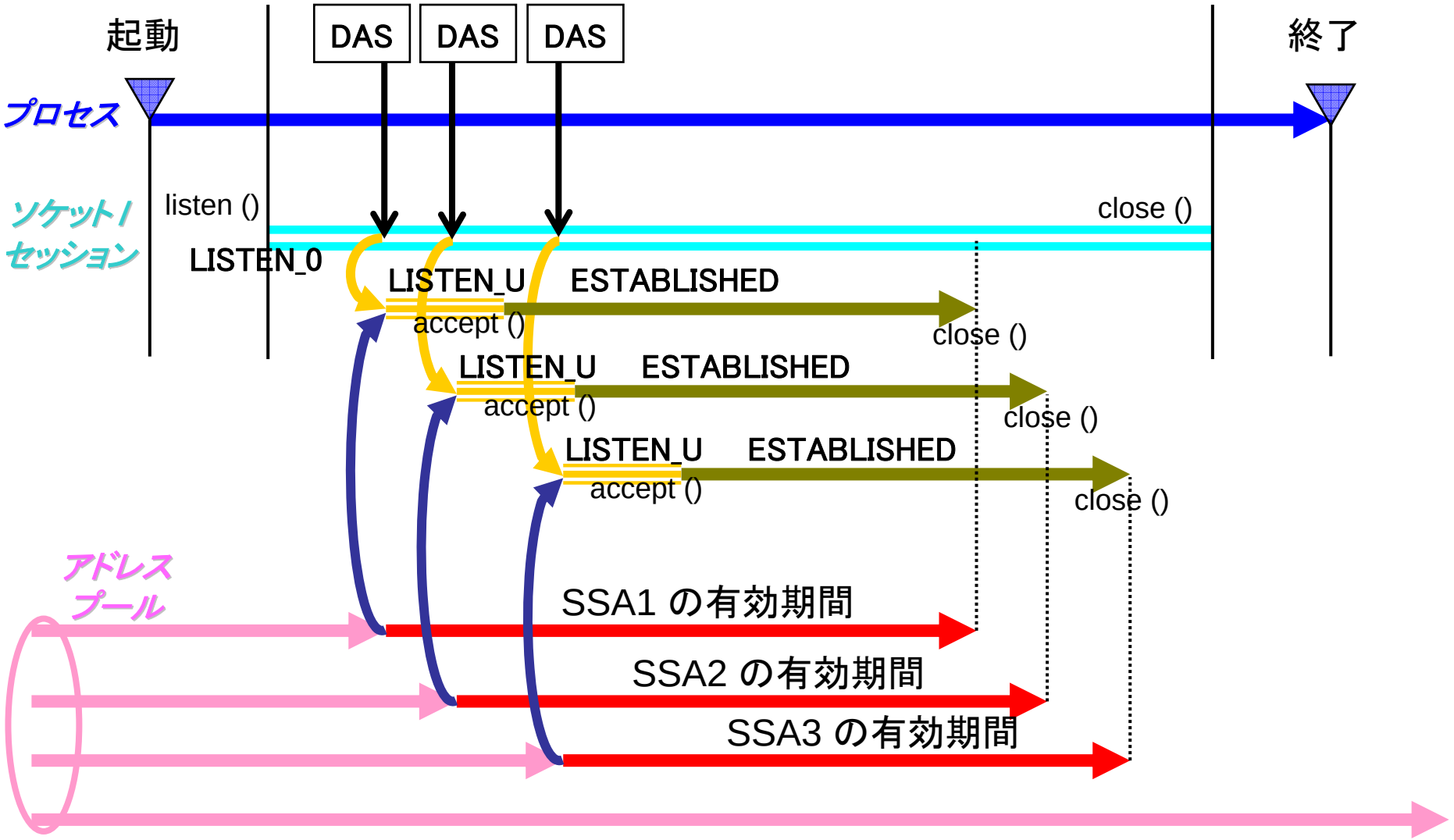
Legacyと同じ方法では対処できそうにない

サーバ側のSSA 実現のための新機能 2/2
課題の解決方法

- 課題を同時に解決するために導入する新機能
  - **DAS** (Delayed Address Setting)処理
    - サーバアプリケーションで listen() を開始したソケットに<u>ユーザが後から</u>接続待ち受けに用いる<u>IPアドレスを付ける</u>
  - **LISTEN_0**、**LISTEN_U** と呼ぶソケット状態
    - **LISTEN_0**
      - 見かけは普通の LISTEN ソケット状態と同じ
      - でも、接続要求は受け付けない
    - **LISTEN_U**
      - **DAS処理に行うことで LISTEN_0 の子供として生まれる**
      - 対象の SSA 用いて 一度の接続を受け付ける

DAS処理と連動して生成するLISTEN_U で
“一重専用接続待ち受け”を実現

サーバでの DAS処理 と SSAでの接続待ち受け



サーバでの一重専用接続待ち受けを実現する典型的手順DAS処理

1. サーバアプリケーションがlisten()を呼ぶ
   対象ソケットは**LISTEN_0** 状態になる
   実際には どこからの接続要求も受け付けない
2. サーバに接続するクライアントが決まる
   提供するサービスの詳細と用いる **SSA** が決まる
3. 決まったサービスの詳細情報から
   アプリケーションの 外から SSAを引数にDAS処理を実行
   LISTEN_U が(LISTEN_0 の子供として) 生成される
   LISTEN_U は SSA でのみ接続待ち受け状態となる

“一重専用接続待ち受け”を実現
接続待ち受けIPアドレス決定のタイミングが遅くても大丈夫

サーバアプリケーションがlisten()を呼んだ後は、クライアントからの接続要求があるまでソケットは長くその状態に留まるので、時間的にもDAS処理を実行することが可能

機能の実装及び検証状況

Unified Multiplex 通信アーキテクチャは以下のOS環境において既に実装済

- FreeBSD 6.2R、FreeBSD 8.0R
- Linux kernel 2.6.24 (実装対象機能は限定)

- 通信アプリケーションの変更なしで、
- カーネルの置換えだけで、
  基本機能が正しく動作することの検証済

Unified Multiplex のIPv4 への応用

- IPv4では　**1ノード-1アドレス**という
  利用形態が一般的　だが、
  - **1ノード-複数アドレス** という形態を排除していない
  - OS実装においても利用を許している
- Unified Multiplex通信アーキテクチャの
  “**固定から変動へ**” という基本概念がもたらす
  セキュリティ的効果は高い
- IPv4環境でもその利用が期待される

まとめ

Unified Multiplex 通信アーキテクチャ:

- 誰もが簡単に利用でき
- セキュリティに配慮できる
  進化した通信アーキテクチャであることを実証済
- 致命的な問題などは見つかっていない
- 未知の問題などがまだ残されているはず

今後とも機能強化や実際のアプリケーションを
利用した検証などの研究活動を継続していく予定