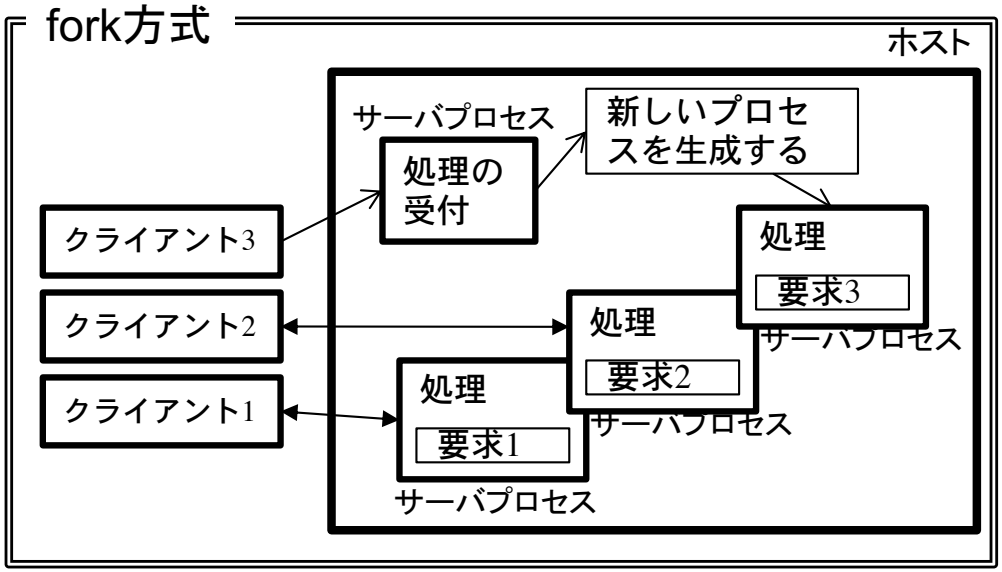
ネットワークコンピューティング
第7回

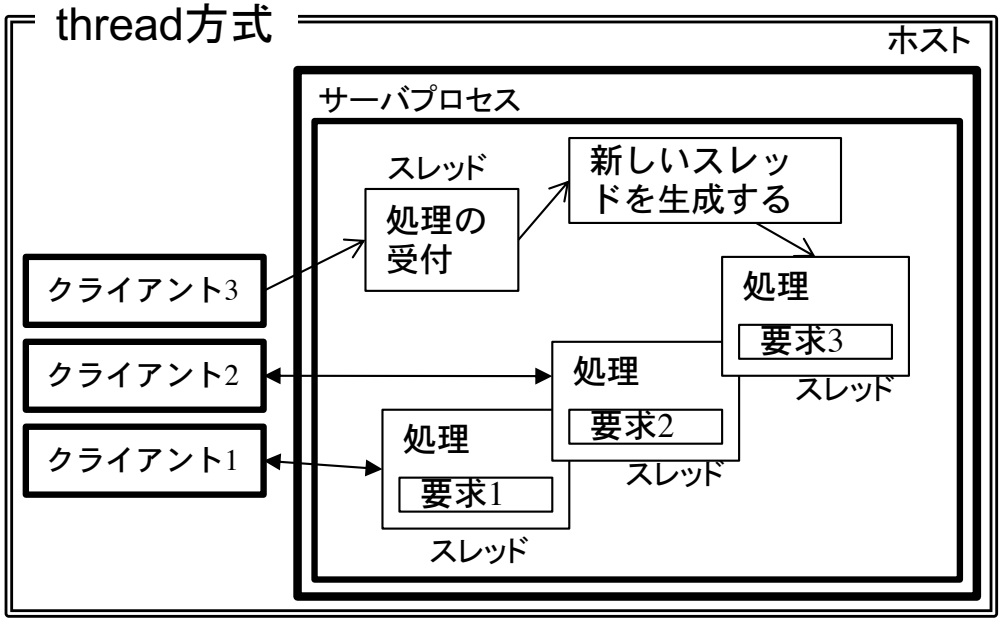
中山　雅哉 (m.nakayama@m.cnl.t.u-tokyo.ac.jp)
関谷　勇司 (sekiya@nc.u-tokyo.ac.jp)

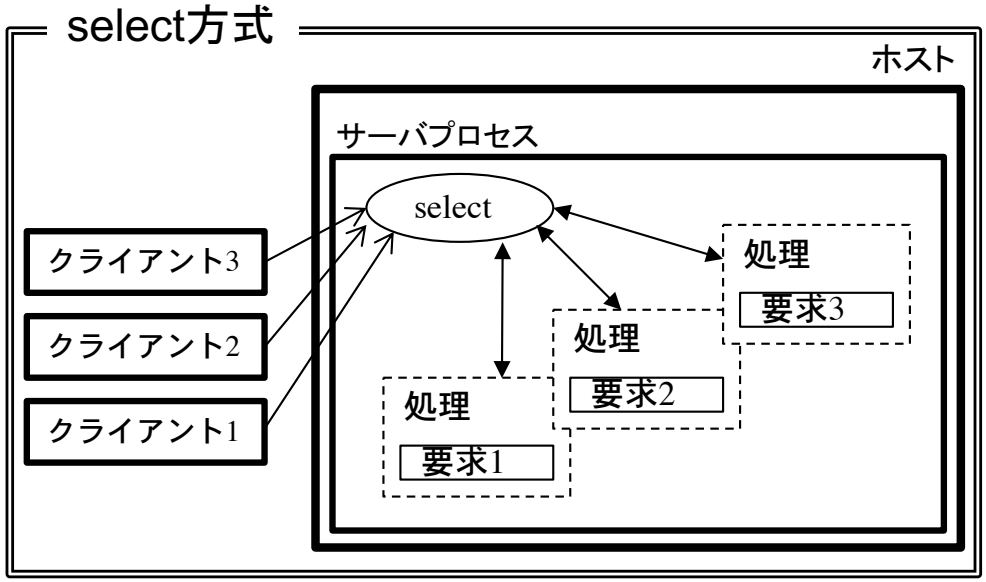
授業に関する情報

- 授業スライド、連絡事項、課題等に関する連絡
  - Web
    - https://lecture.sekiya-lab.info/
  - Mail
    - lecture@sekiya-lab.info
- 実験用ホスト
  - Resources
    - login1.sekiya-lab.info
    - login2.sekiya-lab.info

並行サーバの構成方法

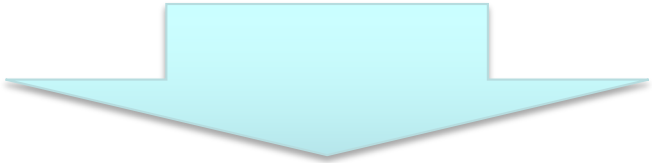

fork方式
ホスト
サーバプロセス
処理の受付
新しいプロセスを生成する
処理
要求3
サーバプロセス
処理
要求2
サーバプロセス
処理
要求1
サーバプロセス
クライアント3
クライアント2
クライアント1


thread方式
ホスト
サーバプロセス
スレッド
処理の受付
新しいスレッドを生成する
処理
要求3
スレッド
処理
要求2
スレッド
処理
要求1
スレッド
クライアント3
クライアント2
クライアント1


select方式
ホスト
サーバプロセス
select
処理
要求3
処理
要求2
処理
要求1
クライアント3
クライアント2
クライアント1

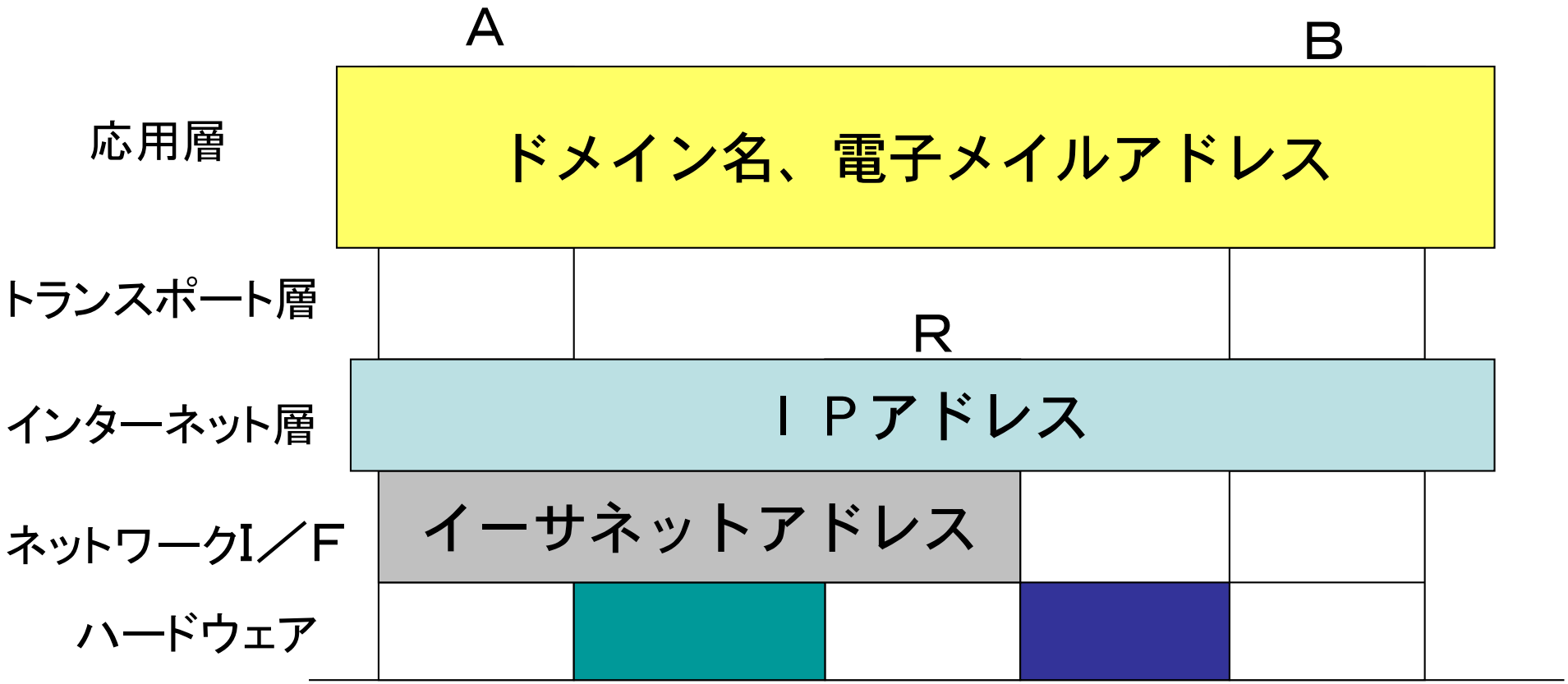
名前解決

- 接続するサーバを IP アドレスで指定する
  - 多くのサーバの IP アドレスを覚えておくのは難しい



- インターネット上のコンピュータ識別方法
  - 「IPアドレス」による識別方法
    - IPv4: 4オクテット(32bit) で構成。オクテット単位の10進数を “.” で区切って表記
    - IPv6: 16オクテット(128bit)で構成。2オクテット単位の16進数を “:” で区切って表記
  - 「ドメイン名」による識別方法
    - 英数字と「-」(マイナス) で構成。論理的階層で割り当て可能
  - 両者の対応は、DNS (Domain Name System) が行う

インターネットモデル


A
B
応用層
ドメイン名、電子メイルアドレス
トランスポート層
R
インターネット層
ＩＰアドレス
ネットワークI／F
イーサネットアドレス
ハードウェア

ドメイン名と IP アドレス

- インターネットでのネットワークプログラムはIPアドレスを通信の識別子として使う
- しかし、ブラウザやメールなど（人が使う）プログラムではIPアドレスを入力することは滅多に無い
  - URL　　http://lecture.sekiya-lab.info/
  - メールアドレス　　lecture@sekiya-lab.info
- IPアドレスは数字の並びなので、人が使うには適さない
  - 210.152.135.178 と www.u-tokyo.ac.jp で覚え易いのは？
- ネットワークプログラムでIPアドレスのかわりにドメイン名が使えると便利
- DNS (Domain Name System) を使ってドメイン名とIPアドレスの相互交換をしている

ドメイン名について

- ラベルを “.” で区切り階層的に管理する
- 各ラベルは 0～63octets で構成される
- 各ラベルで利用できるのは、{a-z or A-Z, 0-9, “-”} のみ(注)
- ドメイン名の最大長は255octets まで

（注）IDN (Internationalized Domain Name) の制定により見かけ上、日本語などをラベルに使用できる様になった。

（RFC3490, RFC3491, RFC3492 を参照のこと）

ドメイン名の一意性

- ICANN (Internet Corporation for Assigned Names and Numbers)
  - TLD (Top Level Domain) の割当てを契約に基づき選定された組織(レジストリ)に委譲する
  - SLD (Second Level Domain) 以降は、選定された組織(レジストリ)が、割当て業務を行う組織(レジストラ)と契約して利用者への割当てを行う

例: “.JP” のレジストリは JPRS が担っている

資源割当てに関わる組織

- ICANN
  - http://www.icann.org/
    - ccTLD
    - gTLD
- JPRS
  - http://jprs.jp/
- IANA
  - http://www.iana.org/
- APNIC
  - http://www.apnic.net/
- JPNIC
  - http://www.nic.ad.jp/

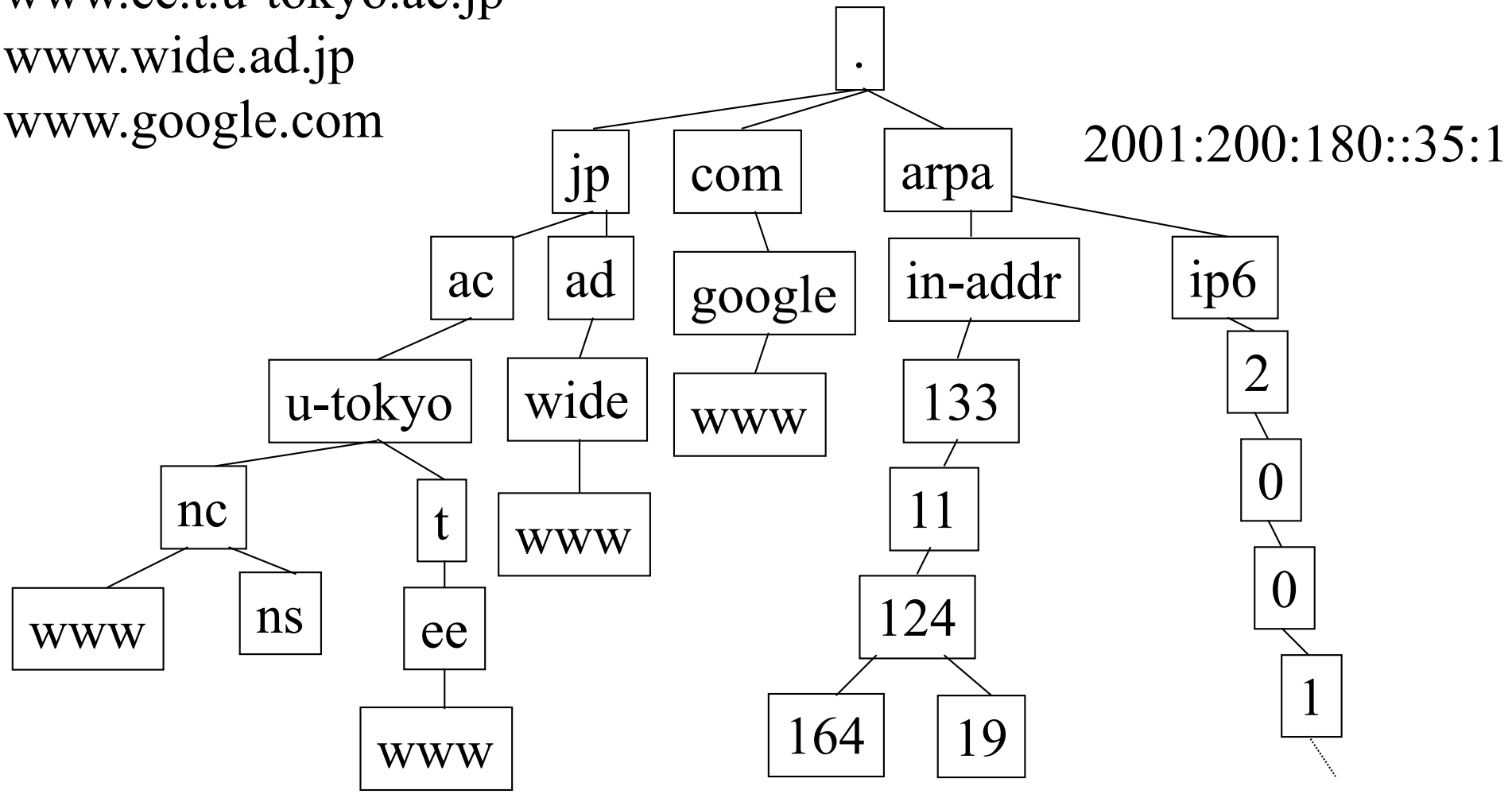
階層構造による情報管理

www.nc.u-tokyo.ac.jp
ns.nc.u-tokyo.ac.jp
www.ee.t.u-tokyo.ac.jp
www.wide.ad.jp
www.google.com

133.11.124.19

133.11.124.164

2001:200:180::35:1



DNS とは？

- Domain Name System
- 階層構造の名前管理システム
- Standards Track の RFC (STD13)
  - RFC1034
    - Domain Names – Concepts and facilities
  - RFC1035
    - Domain Names – Implementation and Specification

  がベース。非常に多くの拡張（Updates）がある。

DNS (Domain Name System)

- 名前(ドメイン名)ーIPアドレスの変換
  - FQDN (Fully Qualified Domain Name) と IP address を対応させる仕組み
  - リソースレコード (RR: Resource Record) の種類
    - A: ドメイン名を IPv4 アドレスにマップする
    - AAAA: ドメイン名を IPv6 アドレスにマップする
    - MX: メールを配信するドメイン名を指定する
    - CNAME: 別名とするドメイン名を指定する
    - PTR: IP アドレスをドメイン名にマップする
    - NS: Name Server のドメイン名を指定する

RR による関連付け

```
ns.nc.u-tokyo.ac.jp.          IN  A          133.11.124.164
                              IN  AAAA       2001:200:180::53:1
www.nc.u-tokyo.ac.jp.         IN  A          133.11.124.19
                              IN  AAAA       2001:200:180:23::8000
www.ee.t.u-tokyo.ac.jp.       IN  CNAME      bell.ee.t.u-tokyo.ac.jp.
bell.ee.t.u-tokyo.ac.jp.      IN  A          157.82.13.244
                              IN  MX  0      mx.ee.t.u-tokyo.ac.jp.
www.iij.ad.jp.                IN  A          202.232.2.164
                              IN  AAAA       2001:240:bb81::10:1

164.124.11.133.in-addr.arpa. IN  PTR         ns.nc.u-tokyo.ac.jp.
244.13.82.157.in-addr.arpa. IN  PTR          bell.ee.t.u-tokyo.ac.jp.
164.2.232.202.in-addr.arpa. IN  PTR          www.iij.ad.jp.

1.0.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.8.b.b.0.4.2.0.1.0.0.2.ip6.arpa.
                              IN PTR         www.iij.ad.jp.
```

DNSを用いたプログラミング

- ドメイン名⇔IPアドレス変換のライブラリを利用
  - gethostbyname(), gethostbyname2()
    - （基本的な）ドメイン名→IPアドレス変換関数
  - getaddrinfo()
    - getaddrbyname を汎用的に拡張した関数
    - マルチプロトコル対応

これらの利用方法については、第6回 (2018/05/24) の資料を参照のこと

DNS を用いたサーバ指定の利点

- 大量のクライアントから同時期にアクセスが集中するケース（特に並行サーバでは）
  - 1サーバで同時に処理できるクライアント数には
    - （OS で扱うことができる）プロセス数
    - （OS で扱うことができる）ソケット数
    - CPU やメモリの使用量
    - ネットワーク I/F の処理能力
  - などの制約で上限がある

⇒ DNS で複数の同じ機能を提供するサーバを指定することで負荷分散できる（例：www.yahoo.com）

ポート番号

ポート番号とサービス

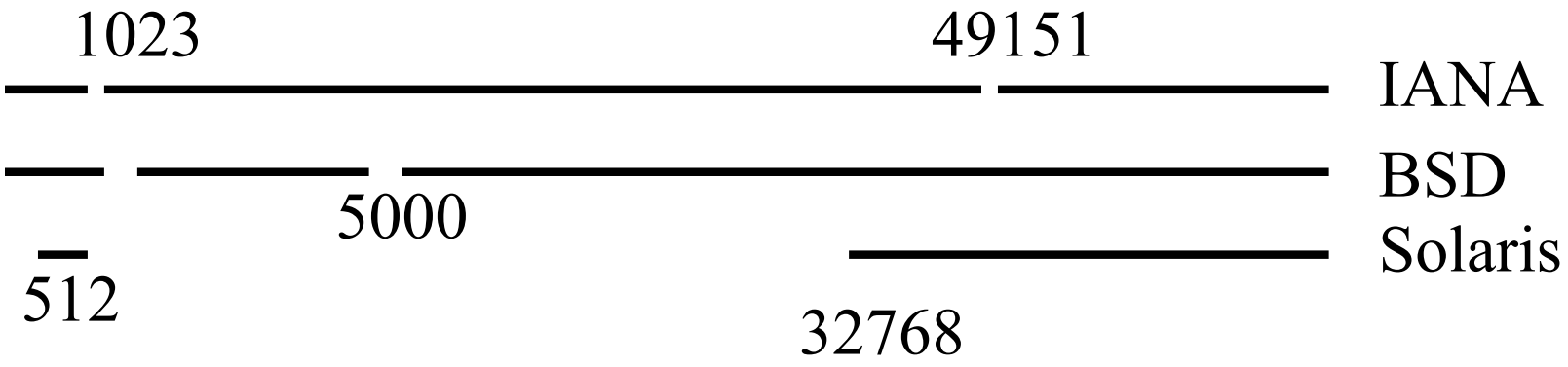
- UNIX 系 OS
  - /etc/services
- Windows
  - C:¥Windows¥system32¥drivers¥etc¥services

```
tcpmux          1/tcp           # TCP port service multiplexer
echo            7/tcp
echo            7/udp
discard         9/tcp           sink null
discard         9/udp           sink null
systat          11/tcp          users
daytime         13/tcp
daytime         13/udp
…
```

これまでに使用したサービス名とポート番号

Port 番号の割り当てについて

- Port 番号の一意性はIANAで保証される
  - 1～1023: IANA well known port
  - 1024～49151: IANA 登録済み port
  - 49152～65535: IANA 動的 or Private



- Port 番号の登録
  - Service Name and Transport Protocol Port Number Registry [RFC6355]
  - http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

新しいサービス(ポート)の利用

- クライアントから文字列を入力し、その文字列を反転して出力するサービス(サーバ)を設計してみよう
  - クライアントから入力される文字列の最大長は 1024文字
  - クライアントから文字列が入力されなくなるまでサービスを継続する
  - 複数のクライアントから(平行して)受け付ける様にする
  - サービスポートは、既に用いられているポート番号は用いない

22556-22762　Unassigned
の領域を使うため、id (1002) + 21600 = 22602 を用いることにする

HTTP サーバ

HTTP (Hyper Text Transfer Protocol)

- Web は　HTML (Hyper Text Markup Language) という言語によって記述されている
- この言語で記述されたデータ (ファイル) を転送するためのプロトコルが HTTP

- RFC1945 : HTTP/1.0
- RFC2616 : HTTP/1.1
- RFC7540 : HTTP/2

telnet を用いた HTTP の確認

```
% telnet www.u-tokyo.ac.jp 80
Trying 210.152.135.178...
Connected to www.u-tokyo.ac.jp.
Escape character is '^]'.
GET /index_j.html HTTP/1.0
```

```
HTTP/1.1 302 Found
Date: Tue, 29 May 2018 16:33:46 GMT
Server: Apache
Location: https://www.u-tokyo.ac.jp/index_j.html
Content-Length: 222
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>302 Found</title>
</head><body>
<h1>Found</h1>
<p>The document has moved <a href="https://www.u-tokyo.ac.jp/index_j.html">here</a>.</p>
</body></html>
```

簡易的な HTTP サーバ

```
while (1) {
    sock = accept(sock0, (struct sockaddr *)&client, &len);
    memset(&inbuf, 0, sizeof(inbuf));
    recv(sock, inbuf, sizeof(inbuf), 0);
    printf("%s", inbuf);

    memset(&obuf, 0, sizeof(obuf));
    snprintf(obuf, sizeof(obuf),
        "HTTP/1.0 200 OK¥r¥n"
        "Content-Type: text/html¥r¥n"
        "¥r¥n"
        "<font color=red><h1>HELLO</h1></font>¥r¥n");
    send(sock, obuf, (int)strlen(obuf), 0);
    close(sock);
}
```

ブラウザからの
リクエスト受信

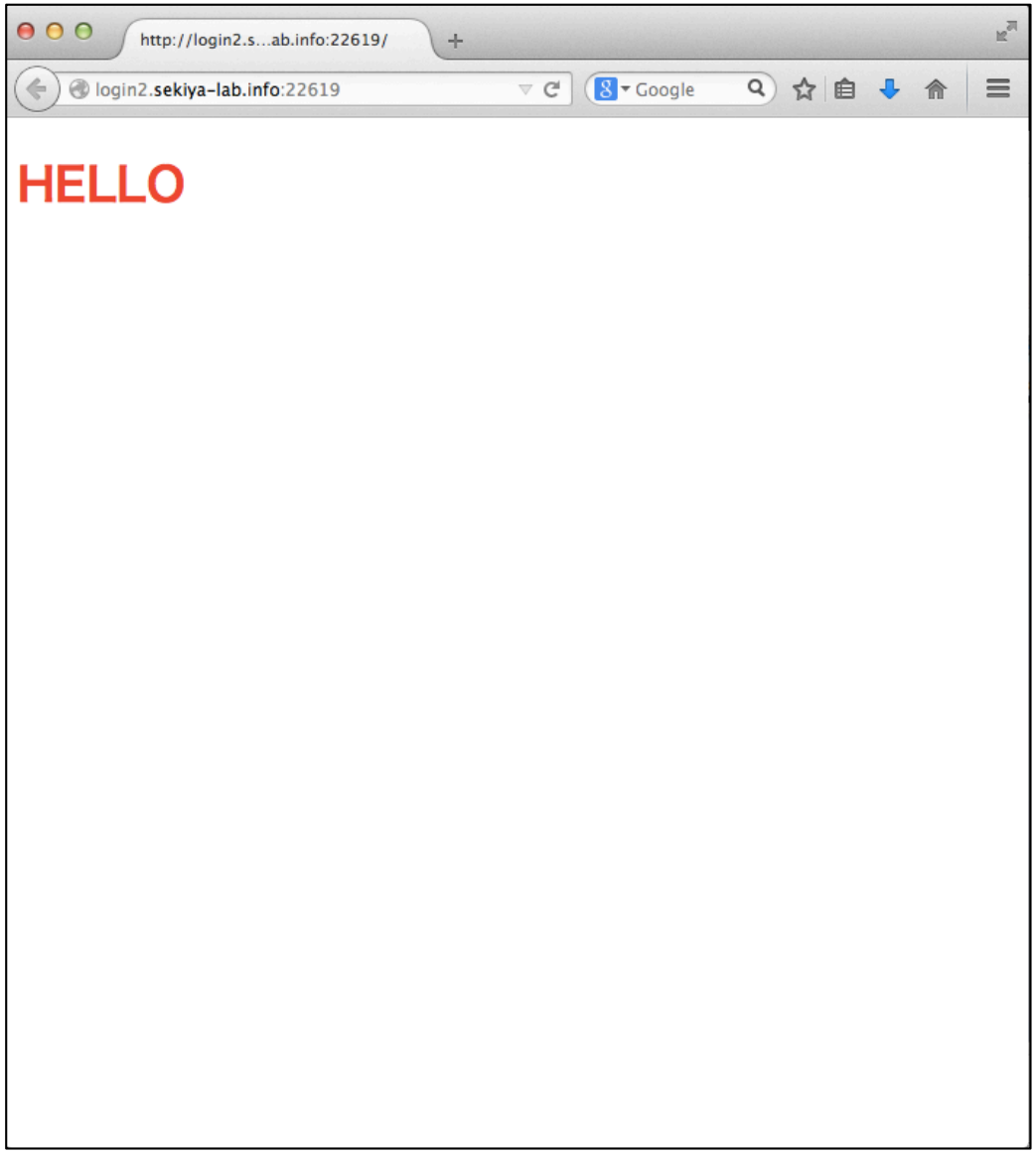
HTTPヘッダ
HTMLの返信

実行結果 (HTTPサーバ)

```
% ./web


GET / HTTP/1.1
Host: login2.sekiya-lab.info:22602
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.12; rv:59.0)
Gecko/20100101 Firefox/59.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

実行結果（HTTP クライアント）



第二回課題

- HTTP サーバを作ってみよう
  - fork(), thread(), select() を使って、複数のクライアントからの接続を受け付ける
  - 可能であれば、それぞれの方法を使って作ってみる
  - 難しければ、どれか1つ or 2つの方式で作ってみる
- サーバを立ち上げるホストとポート番号
  - サーバ用ホスト: login2.sekiya-lab.info
  - 利用するポート番号
    - UID (user ID) + 21600
    - 自分の user ID を確認するコマンドは、以下の通り

```
$ id
uid=1002(nakayama) gid=1002(naayama) groups=1002(nakayama)
```

第二回課題

- 締め切り : 2018/06/13 23:59 JST
- 提出方法 : 以下の情報をメールにて lecture@sekiya-lab.info に送信
  - 学籍番号
  - 氏名
  - ログイン名
  - HTTP サーバのプログラムリスト
  - 実行結果 (Web ブラウザでの表示結果等)