インターネットアーキテクチャ概論

村井純(慶應義塾大学)

１９９８年１２月１５日

InternetWeek 98　**国立京都国際会館**

(社)日本ネットワークインフォメーションセンター編

目的

このチュートリアルで、インターネットアーキテクチャそのものが持つ特徴と、そのコンポーネントの技術的概要を理解してもらいたいと思っています。そして、インターネットアーキテクチャの中での技術の使命がどういうふうに変化しているのかを理解して戴きたいのです。情報の基盤は社会の基盤だというコンセンサスが得られてからまだそんなに日が経っていませんが、更に情報の基盤がインターネットなのではないかというコンセンサスがやはり形成されつつあります。今回のチュートリアルの目的を突き詰めて言えば、皆さんに次の世代のインターネットデザインを一緒に考えて戴きたいという事です。

目次

1.はじめに

私は昨日夜、アメリカのフロリダ州で開催されていた、インターネットソサエティの会議、インターネット技術の会議（ＩＴＦ）から帰ってまいりました。インターネット個々の技術を作っているグループがITFで、ITFのチェアマンの集まりがIESG、そしてITFのアンブレラになっているのがインターネットソサエティとなっています。新しい技術がインターネットから出てきたときにアーキテクチャ全体に対して親和性が有るか否かを考えるインターネットアーキテクチャボードというグループに、私は1993年から２年間務めていました。その時はちょうど次世代インターネットプロトコルの設計をステアリングしていた時期でもありました。インターネットアーキテクチャ概論というチュートリアルが私に任されたという背景もそこにあるのではないかと思います。

キーワード

インターネット・アーキテクチャは、以下のような幾つかのキーワードで特徴付けられます。

- **Best Effort**
- **Scalability**
- **Operation**
- **Autonomy**
- **Distributed**
- **Exponential Growth**
- **End System**

インターネットだからこうなのだと考える時、あるいはインターネットに新技術を使おうと思った際に、この技術はインターネットに対してどういう関係があるのだろうかと考える時、このキーワードが重要になります。ですから一番最初にキーワードを紹介しますが、この後の説明で何度も使いますので、その時にこの意味を理解して戴ければ良いと思います。

Best Effort。最近はBEと略されますが、最善努力という事です。インターネットがデータを端から端まで運ぶときの根本的な考え方です。つまり一生懸命データを送ろうとして、反対側まで送ろうとするのですが、もし遅れなかったらごめんねというのがインターネットプロトコルです。なぜインターネットは着かなかったらごめんねと言えるのかというと、インフラストラクチャをすごく軽く作れるからです。保証をしなくて良い、責任を取らなくて良い、といって網が開放された瞬間に網はものすごい勢いで広がってきたのです。ところが、ここ数か月から１年以内の話ですが、Best Effortのトラフィックとそうでないトラフィックとがインターネットに共存しなければならなくなってききています。Best Effortというキーワードがあえて話題になってきたのは、そうではないものを今インター

ネットは受け入れようとしているからです。

Scalability。新しい技術を誰かが作ったら、その技術のScalabilityはどうなっているのだというように必ず問われる言葉です。これにきちんと応えていなければインターネットで新しい技術を作ってはいけないとまで言われているキーワードです。Scalability、すなわち規模の問題です。インターネットに何人のユーザが繋がると思いますか？ これから、インターネットには何台のコンピュータが繋がると思いますか？ これの答えは誰もわかりません。すごい勢いで伸びてきたインターネットの歴史を私たちは持っており、今後もすごい勢いで伸びていくでしょう。そしてインターネットの全ての機能はそれに耐えられるように作らなければならないのです。アーキテクチャのデザインを考える時も、これ以上大きくなったときはこういう技術で対応しようという事が考えられているのがスケーラブルの発想です。インターネットの仕組みを考えていく上で大変重要なキーワードだということが理解して戴けると思います。

Security。インターネットには皆が繋がっていますから、Securityのことを考えていない機能が一つでもあると、そこから悪いことができてしまいます。従って、要素技術に関してはSecurityのことを考えなければならないのです。Securityは暗号化の技術をディジタル技術で使うような事が主になりますが、プライバシー、それから認証等とも関連します。特に今は重要です。現在インターネットのアプリケーション分野でもっとも重いデューティは電子商取引だと思います。電子商取引が世界の経済を変えると思っている人は多いし、その根本的な技術の鍵がSecurityです。

Operation。これを私たちは忘れがちです。格好良いものや斬新なものを作ることは、勿論すごく大切です。ところがそれを365日24時間動かさなければなりません。インターネットの技術は動かさなければならないのです。 Operationがきちんとできるのかという軸と良いものかという軸とは直交しているような概念かもしれませんが、これをきちんと考えるのはインターネットアーキテクチャの中ではとても大切な事です。

Autonomy。自律性です。これは次のDistributed(分散)と大変深い関係があります。自律分散システムの典型的なインプリメンテーションがインターネットです。ネットワークを運用している単位はそれ自身で自律していますので、基本的には何をしても良いし、どんな技術を使っても良く、次のネットワークと繋がる時に約束を守って繋がれば良いのです。全体を動かすという努力をしていないところがインターネットのスケーラブルな理由の一つで、Scalabilityの根拠の一つは自律性です。それぞれが勝手に分散して動いているのです。

Exponential Growth。 べき乗のカーブで伸びていくという事です。インターネットの事象は全て対数グラフに乗ります。今後これが落ちることはないでしょう。インターネッ

トアーキテクチャデザインの知恵というのは、本当に対数グラフのような勢いで伸びていく需要に対して、用意するリソースが何とかもつように、そしてリソースがきちんと生き延びられるように考え出す事です。リソースをどういうふうにアサインしていくかというのは重要な、深刻な経済問題です。少なくともグラフが直線グラフで上に昇っていくグラフにスケールするぐらいの知恵を絞りだして、このアーキテクチャを設計しなければならないのです。

End System。私達のコンピュータの事です。 最近はエッジシステムという言葉も出てきますので覚えておいてください。 インターネットの網はBest Effortの網なのです。そして最後に私たちは保証をされた信頼できるサービスを受けなければいけません。網はBest Effortで、信頼性を作ってくれないので、End Systemつまり 私達のコンピュータが信頼性を作り出すのです。その信頼性を作るところは当然、最初申し上げたように高価なOperationです。もし届かなかったら保証しなければいけないし、お金もかかるし、よく見ていなければならないから信頼性を作るのは大変です。その大変な仕事を、End Systemに振り回すというのは、ボランティアベースで動いている社会のようなものです。

このキーワードの本質を皆さんに理解をして戴ければ今日のチュートリアルの目的を達成するのではないかと思うぐらいです。これからお話しする内容がこれらのキーワードとどう絡んでくるかということをガイドラインにして戴くと、インターネットのアーキテクチャの特徴がわかりやすくなると思います。
私のチュートリアルの守備範囲は、現在までの事と、今日のスナップショット、明日の方向性までです。 明後日のアーキテクチャは皆さんが作らなければならないし、皆さんにお任せしたいというのが今日の内容です。

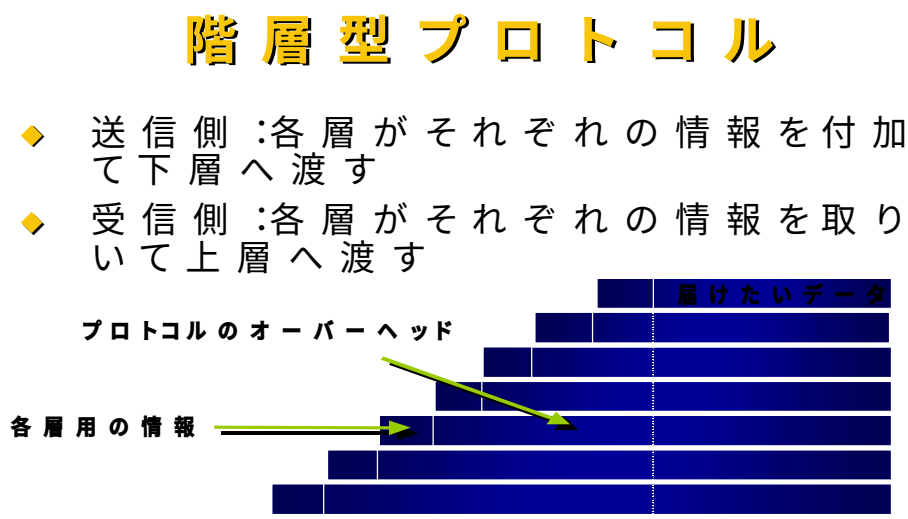
OSIの７階層モデル

ＯＳＩの７階層モデル

- プロトコル設計のものさし
- わかりやすい機能の分離と独立性
- エンドシステムの n層同士が通信を行っているつもり
- 中間システムは１～３層でリレーするだけ

OSI７階層モデルの重要なことは階層モデルだということです。まず図の横に引いている線を考えていただきたいのです。横の線（水平の線）がディジタルコミュニケーションを作る時に、とても大切になります。
今私が皆さんにお話をしている、そして皆さんは私の話を聞いているという事を考えてみて下さい。私は頭の中で思っている事を皆さんに伝えたいと思っています。私の頭の中には皆さんに伝えたい意味があります。それを私は言葉にするということをやるわけです。私は意味を日本語で皆さんに伝えようと思うので言葉にします。そしてその言葉を私は声帯を使って空気を震わせます。この音が皆さんのところに行って、皆さんの鼓膜を震わせます。その音で皆さんは言葉を作って、そしてその言葉を意味にするのです。私が意味から日本語にしようと思ったとき、日本語から意味を感じ取ろうという逆のことを皆さんはやっています。そして日本語を音にしようと私がやっているときに、皆さんは音を日本語にしようと逆のことをやっています。そして私は声帯を震わせそれが物理的に皆さんのところへ行って鼓膜を震わせます。
この構造は、コミュニケーションには対応する役割分担があって、お互いに、それを伝える物理的なメディアがあるのだ、ということがわかります。ディジタルコミュニケーションがやることは、そういったモデルをまず作って、役割を決めて、それで物理的な媒体で最後に数値を送る事です。これが、基本的なアーキテクチャのモデルです。
オープンシステムインターコネクション(開放型システム間接続)のモデルを考えだしたのが70年代の後半で、上図の通りです。
覚えて戴きたいのが、ネットワーク層とトランスポート層です。
ネットワーク層はコンピュータからコンピュータまで繋ぐということで、トランスポート層はコンピュータに着いたらそれぞれのプログラムにばらまくということを司っているのです。インターネットはネットワーク層がBest Effortで、トランスポート層で信頼性を作っています。End Systemにしかトランスポート層はないのですから、End Systemだけで信頼性を作っているというのが良くわかります。それでもう一つこの図からここで理解しておいていただきたい事があります。
中継ノードはいちばん上に上がっても、原則として(例外はありますが)ネットワーク層までしか上がらないで折り返すということです。中継ノードの役割が定義されることで、数珠つなぎになったネットワークを想定することができます。物理的には様々なものがあっても、その上に乗っているネットワークがあり、それが相互に繋がって、エンドツーエンドでコミュニケーションできるのがこのアーキテクチャの特徴です。。

階層型プロトコル

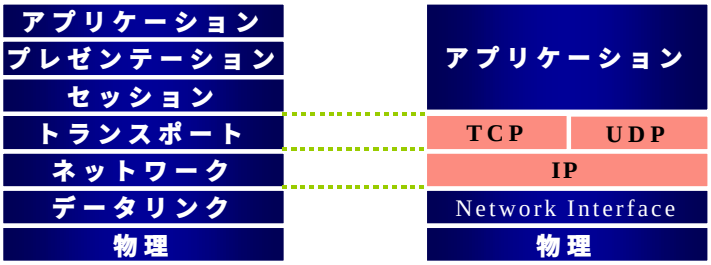


この階層型プロトコルをビルディングにたとえます、そして皆さんは６階の住人だと考えて下さい。皆さんの仕事は相手のビルディングの６階の方と話をする事です。それが先ほどのプロトコルの役割です。(７階層モデルの横の線です。)
ビルディングの７階から１階までの住人はそれぞれ役割分担が決まっています。
つまり、６階の住人は、上（７階）のフロアからもらってくるものを受け取ります。そして下（５階）の階の人に、これを相手ビルディングの同じ階（６階）に届けてくれといって渡すのです。そういうのがこの構造ですから、相手のビルに届いて、相手の同じ階（６階）の人とやり取りをする制御情報を付けます。それをヘッダといいます。
頭に付けるからヘッダです。つまり皆さんは７階からもらったデータに相手の６階とうまく話ができるようにヘッダを付けて５階の人に渡すわけです。そうすると５階の人はそれ全体がデータになって、更に５階のヘッダ情報を付けて下に渡すのです。こういうことを繰り返していきます。
そうするとこのヘッダがそれぞれ付いてきます。そして相手に渡って、今度は逆に相手のビルディング(上の絵全体を相手のビルディングだと思ってください)の人は川から流れてきたデータを受け取ると、１階から２階、そして３階、４階と、各階のヘッダを読んで、その上に渡します。
このようにデータが構成されていますから、実際にネットワークのトラフィックとして流れるのは中身だけではなくヘッダを含めた大きさになります。ヘッダの分はオーバーヘッドになります。
例えばインターネットフォンでダイヤルアップをして、音声を流したとします。その時には音声データにヘッダが全部付いているわけですから、一個一個のデータにはオーバーヘッドになっているのです。このように全体のトラフィックをオーバーヘッドで割ったものが、回線実効データの転送効率になります。今のインターネットではIPだけで20バイトが

付いています。
これが階層型のアーキテクチャだということを理解して戴きたいと思います。

OSIモデルとインターネットアーキテクチャ

ＯＳＩモデルとインターネットアーキテクチャ



インターネットは実際に動かすための技術ですから、OSI７階層モデルの内のいくつかを簡素化して作られています（上図)。一番重要なのは、**IPとTCP**です。IPの上にTCP、UDPが乗っています。基本的にはIPの上に何が乗っていても良いのです。重要なのはコンピュータからコンピュータにデータを流すのはIP、インターネットプロトコルだというです。したがってデータリンク層より下は何でも良いという事です。IPさえ通れば何でも良いのです。これはとても大きな意味を持っています。

テレビの場合、飛ばす電波(UHFやVHF)の周波数割当てが決まっており、どの放送局がどこのテリトリーでフランチャイズできるかということもきっちり決まっています。電波が混信しては困るからです。電波の方式が決まっていますから使い方が決まっており、使い方が決まっていると（放送局から電波を流すと皆さんの受信機に受け取るという）コミュニケーションモデルが決まります。その上にビジネスができ、ビジネスの上に文化ができて、私たちはテレビと付き合っています。物理的なメディアをどう使うか決めた瞬間からもうテレビの文化のすべては決まっているのです。テレビは電波の割り当て、使い方、レギュレーション、そしてビジネス、その上の文化、全部縦に積んでいるテクノロジーです。インターネットはIPが通れば下は何でも良いいうテクノロジーです。したがって横形といっても良いでしょう。IPが本当にやってしまったことは世界の人類が持っている通信技術に風呂敷をかけて、「これで誰とでもどういう具合にも数値データをやり取りする事できるので、何か好きなことをやって下さい」とIP Layerが語っているのです。

重要な事は通信技術に縛られないで何でもコミュニケーションできるようにするという部分です。

ＩＰの部分さえあれば下はどうでも良いといった事は、私たちがこれからインターネットを使っていく上では本当に重要な側面になってきます。

通信形態（ＶＣ型vsDG型）

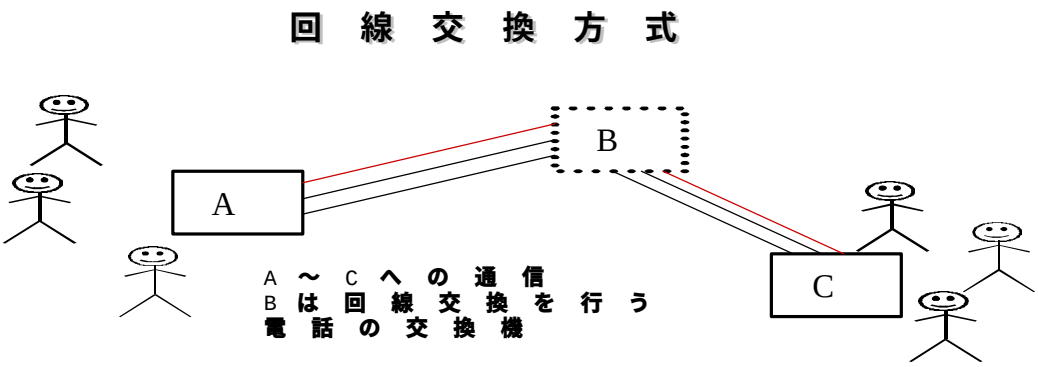
バーチャルサーキット(VC)、データグラム(DG)という二つの言葉を覚えて下さい。
OSIの言葉では、コネクションオリエンテド(CO)、コネクションレス（CL）といいます。
バーチャルサーキットは、基本的に電話と同じです。最初にダイアルして繋がる(コネクションを張る)と、相手と会話をすれば良く、会話が終われば電話を置く（コネクションが切れる）という事です。
データグラムは、葉書と同じです。葉書に住所とメッセージを書いて送る。もう一枚メッセージを書きたければ、２枚目にも住所とメッセージを書いて送る。ということをやって、必要なら何枚ものはがきに住所とメッセージを書いて送れば良い。その代わり着く時に順番が狂うかもしれません。
２つの通信形態の比較を次に示します。

バーチャルサーキット(VC)型	**データグラム(DG)型**
信頼性	信頼性保証せず
順序保証	順序保証せず
フロー制御	フロー制御なし
再送	再送なし
従量課金	固定課金
信頼性があるが、オーバーヘッドが大きく、処理能力が必要	信頼性はないが、オーバーヘッドが少なく、処理能力を要求しない。

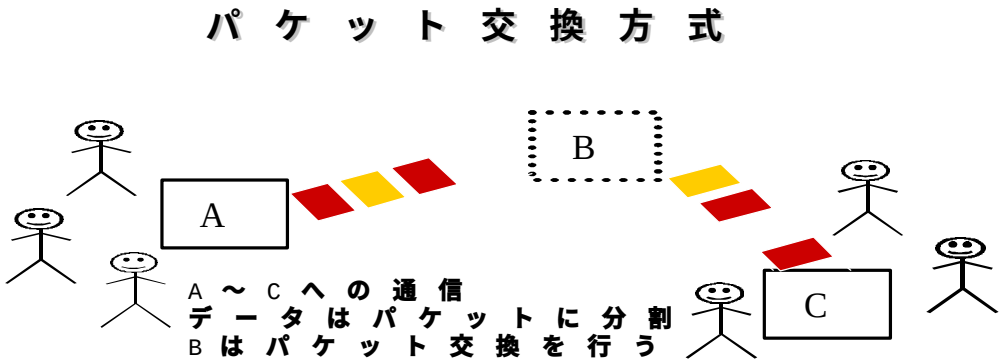
インターネットの場合、IPはデータグラムですが、その上の、TCPはバーチャルサーキット、UDPはデータグラムを作ります。現在のインターネットでは、80から90パーセントがTCPです。
Web、電子メール、ファイル転送、Telnet、全てがTCPです。そのどれを取っても１ビットでも落ちたら困るもので、信頼性が必要です。だからTCPを使っています。それでは信頼性のいらない通信はあるのかと問われそうですが、これがあります。例えばテレビです。テレビは１秒間に30枚の画像が送られていますが、30枚から１フレーム落ちても構わないのです。こういう通信はUDP向きなのです。音声も同じです。インターネットでは、これから先、画像や音声のトラフィックがものすごく増えてくることを予想しています。 今までのインターネットはTCP(バーチャルサーキット)が殆どで他のことを無視してよかった。これからのインターネットはUDP(データグラム型)を含めて新しいコントロールと応用を受け入れなければなりません。従って、これからアーキテクチャ上の非常に面白い挑戦が出てくるのです。

回線交換方式vsパケット交換方式

サーキットスイッチング(回線交換)、パケットスイッチング（パケット交換）も同じような概念としてあります。これもアーキテクチャを考える上で重要になってきます。



回線交換は回線を張って相手と繋ぎます。ここでのポイントは、Bの交換機はAの３人のために、それぞれ帯域を確保する事です。誰がいつ話しても、きちんと繋がってデータが伝わる保証があります。これが回線交換の考え方です。重要なことは 、リソースが確保してあり、それが小さくなることを心配しなくて良いことです。これと今のインターネットの考え方は全然違います。インターネットの考え方はどちらかというと、パケット交換の考え方です。

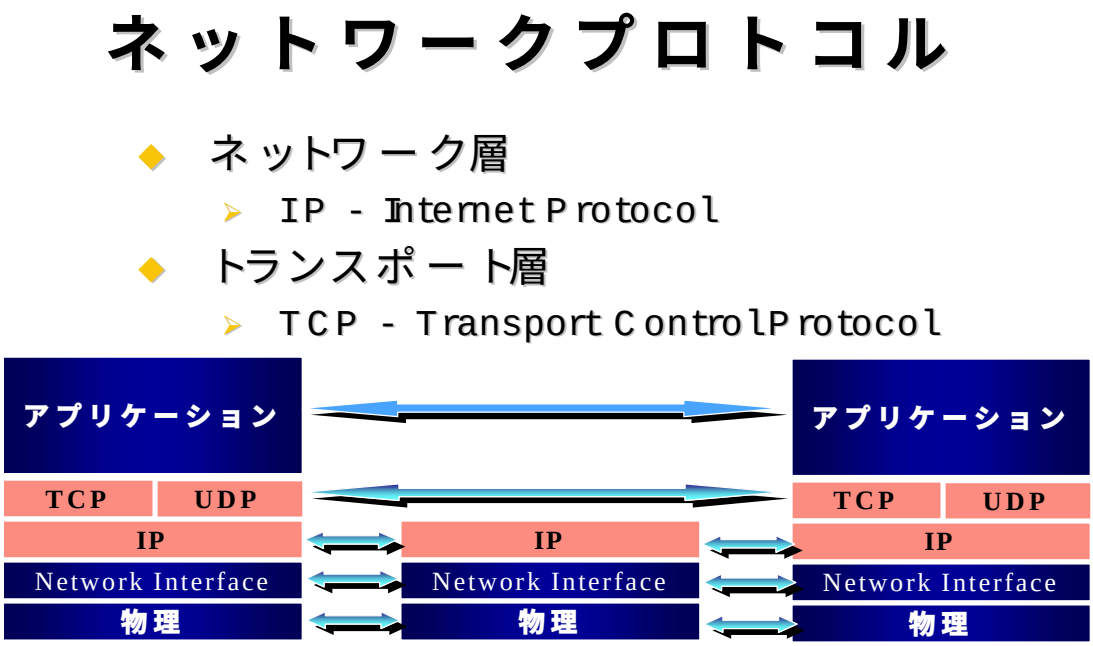


パケット交換は３人のパケットが適当に混ざって送られていくのですが、それをコントロールして割り当てないのです。したがって全体のトラフィックが多くなるとBの交換機は溢れるということが起こりますが、Bが回線交換の交換機と同じキャパシティであれば全く同じ通信ができるのです。

ディジタル衛星放送は典型的な回線交換型の考え方を持っています。パーフェクTVでもダイレクTVでも100チャンネルなら100チャンネル分の容量を持っています。１チャンネルに30Mbps相当のデータの送信が必要ですから、衛星は30Mbps掛ける100チャンネルのキャパシティーを確保しておく必要があるわけです。最初のスケールの問題を考えてみまし

ょう。回線交換方式は上限がある程度決まっているときに可能です。衛星放送が最大何チャンネルになるかわからないようでは、衛星を何発打ち上げても間に合わなくなるでしょう。インターネットは、Exponential Growthで上限が設定できないのでパケット交換方式なのです。しかし帯域を確保してないという事実があるので、溢れた時にはパケットドロップが起こることになるのです。

2. インターネットのコア技術
ネットワーク層（ＩＰ、ＩCMP）



インターネットはパケット交換方式で、溢れた時にはパケットドロップが起こることになるという問題をどうするかですが、その中にインターネットのコア技術があります。
さて、インターネットは、エンドツーエンド、即ちあるコンピュータからあるコンピュータに行くことに責任を持っているのは下図の「IP(インターネットプロトコル)」です。途中で物理層を通って中継ノードに行き、この中継ノードで乗り換えて、次のネットワークに乗り最終的な目的地に行きます。
着いたら必要に応じてＩＰからTCP、あるいはUDPに行き、アプリケーションに行くという具合になります。

ＩＰ

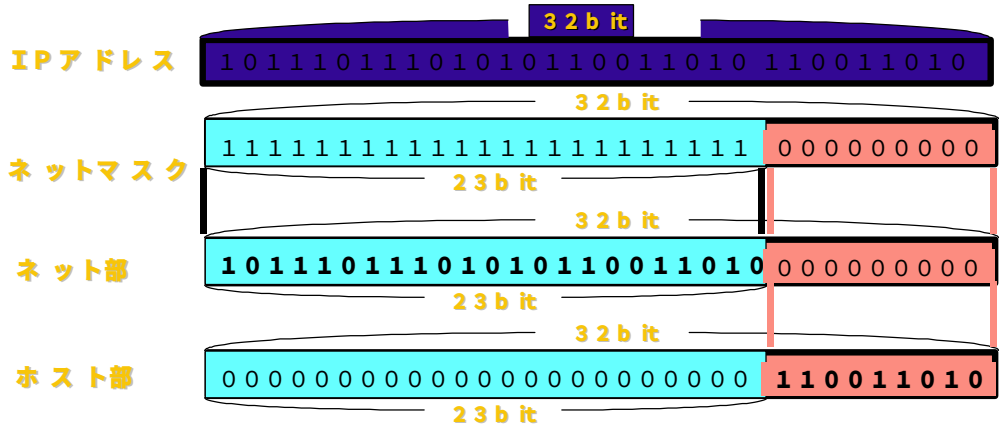
インターネットの個々の技術の話をします。

ＩＰの機能

・ホストの識別	32bitの識別子: ＩＰアドレス
・経路の決定	IPアドレスの一部から経路を決定
・データの中継	バケツリレー方式
・データの分割	一度に送れないデータを分割、再編成

その時に理解のし易い方法をお話します。 皆さんは人間であることを忘れて、IP Layerのソフトウエアになったと思い、自分が何をするべきかを考えると良く理解できます。
皆さんはIPです。 皆さんの使命は世界中のコンピュータに乗っているIP、皆さんの仲間と話ができるという事を保証してあげなければなりません。Best Effortですからうまく行かなかったらごめんねと言っても良いのですが、届けられる時は届けられるのだということをコミットしてくれなければ困ります。
まず最初に皆さんがやらなければならないことは相手の名前を知る事です、識別することです。ユニークに識別しなければなりません。これがエンドツーエンドのコミュニケーションの一番重要な事です。世界中のIPのユニークな識別が、IPアドレスです。現在(IPv4)のIPアドレスは32ビットですから、40億個の番号を付ける事ができます。すべてのノードにＩＰアドレスを付けるのをアドレスアロケーションと言い、JPNIC等で行っています。
皆さん京都にいます。近鉄線で来たお客さんがいて東京に行きたいとします。皆さんはお客さんの行き先が東京であることを見て、新幹線に乗せる事を案内する使命があります。IPがその使命を果たさないとデータは着きません。というようにデータ中継をしますが、もう一つ、使命があります。今はもう無いようですが、京都駅から出ている京都市電があるとしましょう。多くのお客さんが新幹線で京都に来て、京都駅で京都市電に乗り換えさせてあげますが、新幹線１車両分の長いデータを乗り換えるので、１両の京都市電には乗りません。それで何両かの市電に分乗させなければなりません。これをフラグメンテーションといいます。乗り換え元のパケットサイズよりも乗り換え先のパケットサイズが小さかった時は分割して下さいというのが、IPの皆さんへのお願いです。
まとめます。IPアドレスがユニークに振ってあるので、上から来たデータは適当な経路制御を考えて行先を案内してやり、必要なところに降ろしてやるのです。そして下から受け取ったデータは同じように降ろしてやってください。ただし到着アドレスが自分だったらどのプロトコルに渡せば良いかを考えて上に上げてください。基本的な仕事はこれだけです。つまり、これだけのソフトウエアなので、IPのプログラムはものすごく簡単です。簡単だと世界中のプログラマが書けます。プログラムを書ける人、理解できる人、支えられる人がいなければインターネットは動かないのだという哲学が、IPをものすごく簡単にするというアーキテクチャに繋がっています。

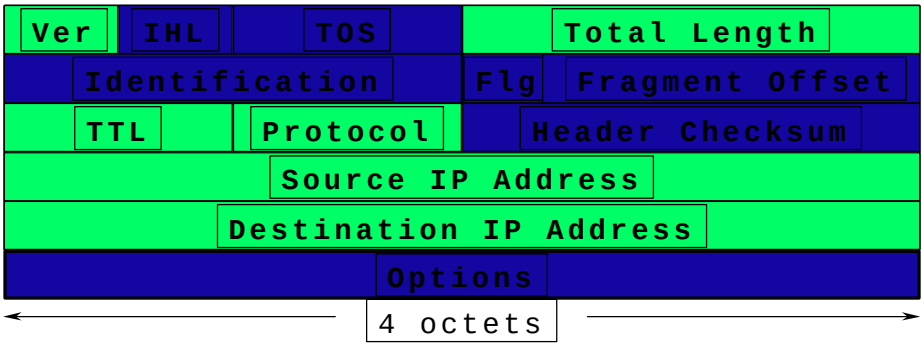
ＩＰアドレスとネットマスク



ところが、このままですと、地球上すべてのIPアドレス、経路制御表を持たなければならないという大きさ（Scalability）の問題になってきます。この事はアドレス枯渇問題と並んで、あるいはそれ以上に問題になっているのです。インターネットに繋がっているノード数が増えるに従い、調べなければならないテーブルはそれに比例して大きくなるのです。従って、Scalability作戦、知恵を働かせなければなりません。

皆さんが京都駅のIPで、山手線の原宿行、山手線の目黒行、山手線の目白行という具合にテーブルを持ったら、テーブルは駅の数だけ必要です。 ところがせっかく山の手線の原宿行、山の手線の目黒行と言われているのですから、全部新幹線に放り投げれば良いのです。山手線の駅は新幹線に投げるということだけ持っていれば、山手線の数だけテーブルが減ります。山手線にいくつ駅ができようと、山手線は新幹線に乗せるいう表を持っていれば充分だという事になります。こうすると、スケールが効いてきて、ものすごいカーブで駅数が伸びてもテーブルはまっすぐにしか伸びないという知恵が出てきました。つまり、アドレスを山手線（これをネット部分といいます）と何々駅（これをホスト部分といいます）の２つに分けます。そして、どこで区切ったのかを伝える為に、ネット部分に１を立て、ホスト部分に０を入れた32ビットのデータ（マスク）を用意し、アドレスといっしょに使います。マスクが23ビットある時、IPアドレスの後にスラッシュ23と書いたり、これをそのまま１バイトずつ255.255.255.のように書きます。

ＩＰヘッダ（ＩＰｖ４）



上図がヘッダの実態ですが、皆さんの仕事はここの情報だけを見れば良いようになっています。

・Ver ― バージョン番号.。現在４が入っていますが、将来６になります。

・IHK ― ヘッダレングス。ヘッダの長さです。オプションを含めると長くなります。

・TOS ― Type of Service。中継ノードでの希望サービスです。

・Total Length ― 全体のデータ長。

Identification ― ＩＤ番号。

・Fragment Offset― フラグメンテーションした時、データが先頭からどの位置なのかを示す値が入っています。

・Time To Live ― 寿命を示すカウンターです。 インターネットは自律的に動いているので経路制御テーブルに矛盾があったら、パケットがネット内をループする可能性があります。そこでパケットを乗り換えさせるときに、このカウンターから１引いてゼロになったら捨ててしまうのです。ループをしていてもう行き先がなく、死人同然なのに死なない生き残っているパケットをゾンビと呼びますが、このゾンビパケットを作らないためのメカニズムです。

・Protocol ― プロトコル番号です。目的地が自分だった場合に、TCPに上げるのかUDPに上げるのか、それともRTPに渡すのかを示します。、

・Header Checksum ― ヘッダのチェックサムです。、

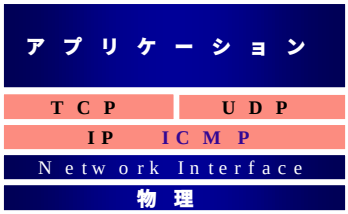
・SourceIP Address、DestinationIP Adress ― 送り元と、目的のアドレスです。Destination Addressは目的地に着くために必要ですが、Source Addressは送った人に返事を出す場合に使用します。

ICMP

IP Layerの仕事をする皆さんに、今度エラーに関してどうすれば良いかをお教えします。皆さんのリソースやメモリがなくなったり、それから経路制御テーブルを引いても行き先がわからなかった場合には、パケットを捨てて構いません。その代わりSource Addressに向かって捨てた理由を報告してください。それも着かないかもしれませんが、ないよりましなので、一応それだけをやってください。これがIP Layerに対するお願いです。

ＩＣＭＰ

- Internet Control Message Protocol
- 機能的には IP を補うもの
  - 送信先への到達不能，フロー制御など
  - IP にかわってエラー報告する．



この仕組みのことをICMP（Internet Control Message Protocol）といいます。
pingやトレースルートというコマンドはICMPを利用しているのです。トレースルートは、パケットがどういう経路を経由して宛て先まで届くかを知るコマンドです。トレースルートは、始めはTTLを１にして送ります。そうすると１個目の乗り換え地でゼロになるので、１個目の乗り換え地のＩＰはTTLがゼロになったのでパケットを捨て、Source Addressに向かって、寿命が尽きた事を報告します。１回目の中継ノードはその送ってきたICMPのSource Addressを見るとわかるのです。次に２にして送ります。そうすると２個目の中継点からレポートが戻ってきますから、更にに３にして、４にして、目的地に着くまで１個ずつ上げていくと、到着地までの道のりが全部わかるのです。また、報告までの時間を計

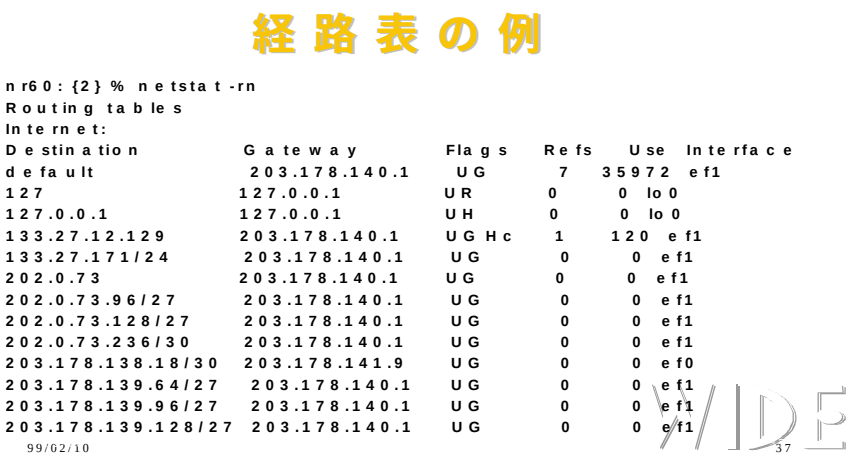
ＩＣＭＰメッセージタイプ

- エコー応答（Echo replay）
- 宛先到達不能（Destination Unreachable）
- 発信抑制（Source Quench）
- ルート変更（Redirect）
- エコー要求（Echo request）
- Datagramの時間超過（Time exceed）
- Datagramのパラメータ異常（Parameter Problem）

測していると、経路のどこで時間かかったかまでがわかってしまうのです。
又、ネットスタットマイナスというコマンドを使うと、自分のコンピュータが受け取っている中継ノードからのICMP全部履歴が統計情報としてみる事ができます。

経路制御

経路制御は基本的にはどの路線に乗せればいいかということだけを決めています。



上のリストが経路表（ルーティングテーブル）の例です。
目的地(Destination)、乗換駅(Gateway)、どのイーサネットのインターフェース(Interface)に投げれば良いかいいかということが書かれているのが理解できると思います。
新しい線ができたり、新しいコンピュータが繋がったら、経路制御テーブルを更新しなければなりません。経路制御のテーブルをいつも新鮮にする為のプロトコルを経路制御、経路情報交換のプロトコルと呼びます。
一番プリミティブなものに、RIP(Routing Information Protocol)あります。

- 送信元と宛先との間で最適なルートを探す
- 送信元と宛先間の通過しなければならないネットワークの数を単純にカウントする
- ホップ数の一番短いルートが最適ルート
- 最適ルートの計算には簡単なコスト計算（三角不等式）を用いる。来た情報をかき集めて、自分との差分を継ぎ足し、バケツリレー的に作っていくという方法です。

今主に使われているのは、OSPF(Open Shortest Path Find)というプロトコルです。

- サイト内のルータは同じデータベースを共有する
- この情報を元に最短パスツリー(Shortest Path Tree)を形成する
- ネットワークの変化に柔軟に対応できる

RIPは、定期的に自分が持っている情報を全部周りにばらまくので、情報量がものすごい量になってきます。私自身、日本のインターネットとアメリカのインターネットのIPをつ

ないだのが1989年の１月15日でした。まる10年になりますが当時はRIPです。そのとき9.6kの線で日本とアメリカをつないだのですが、その瞬間にアメリカ全部のRIPトラフィックが9.6kのパイプで日本に向かって流れだし、しばらく動かなくなるという経験をしました。急激にIPが増えてトラフィックを圧迫し、RIPは破綻し、現在はOSPFです。そして、もう一段階上のASという概念が出てきています。

トランスポート層(TCP、UDP)

TCPとUDP、つまりトランスポート層(Transport Layer)と呼ばれる、IPの上の層に移ります。Transport Layerの上にはソケットがあり、ソケットを介して、いろんなプログラム（Webのクライアント、Webのブラウザ、Webのサーバ、電子メールソフトウエア等）が動いています。これらの交通整理をするために、それぞれにポート番号を振ります。アイアナ(インターネットのコミュニケーション全体の番号を付ける方針を決めているところ)で主なポート番号は決まっています。例えばWebのサーバは80番のポート、電子メールは25番のポートという具合です。

UDP(User Datagram Protocol)

UDPはIPにポート番号という概念を付けただけで、データグラムがそのまま通ります。もし信頼性を作りたいのなら、アプリケーションの責任という事になります。例えば、NFS、ビデオコンファレンス、インターネットフォン等のアプリケーションは全部UDPで通信をしています。

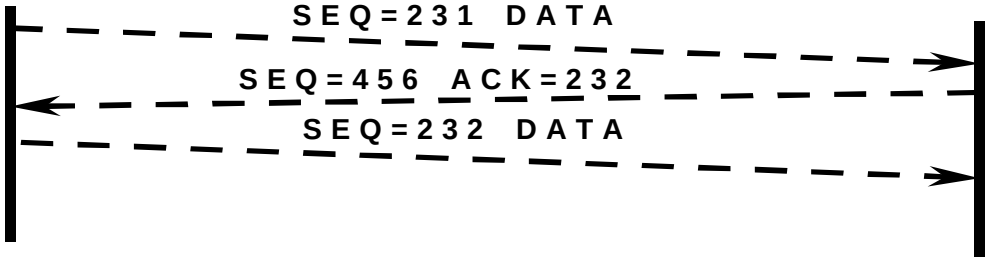
TCP(Transmission Control Protocol)

今のインターネットのトラフィックは90パーセント位がTCPで、大変重要です。
TCPはポートから来たデータに信頼性を加える事です。その為に次の事をやります。
・エラー検出とエラー訂正
・フロー制御
・順序の再構成

フロー制御と輻輳制御

データの転送

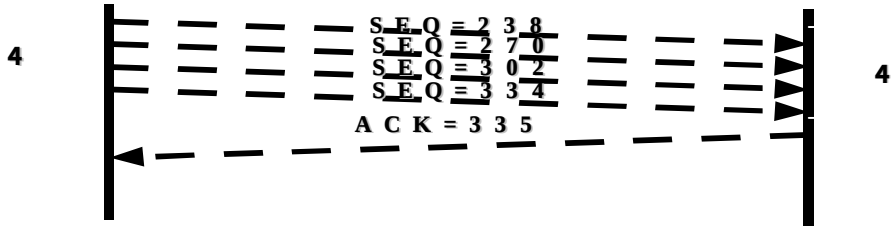
- SequenceNumberとAcknowledgeNumberによって、データの順番を保証
- ACKが帰ってきたら次のデータを送る
- ACKが帰ってこなかったら前のデータを再転送



トランスポート層はポートとポートを結びつける為にコネクションを張ります。コネクションを張ってデータ転送を行います。ところが転送効率が悪いという問題が出てきました。１回送る度にACKを待っていると待ち時間が発生し、データ転送時間が長くなるのです。先にデータを送っておいて、後でまとめてACKをもらえれば、転送効率は上がります。この際、どこまでの幅をまとめて送ってもいいかということは相手の能力に依存します。

ウィンドウコントロール

- 先送りする幅を決める仕組み
- 送るパケットを制限する



相手の能力を見て、こちらから送るデータを制御するのを「フローコントロール」と言います。相手がどれだけの幅をまとめて受け取れるのかというネゴシエーションをダイナミックにすることで、効率が良くなります。
例えば、四つ(上図)なら四つ送って返事を一個もらう。それで次の四つを送るというふうにやっていくのです。これを、ウィンドウベースのフローコントロールといいます。

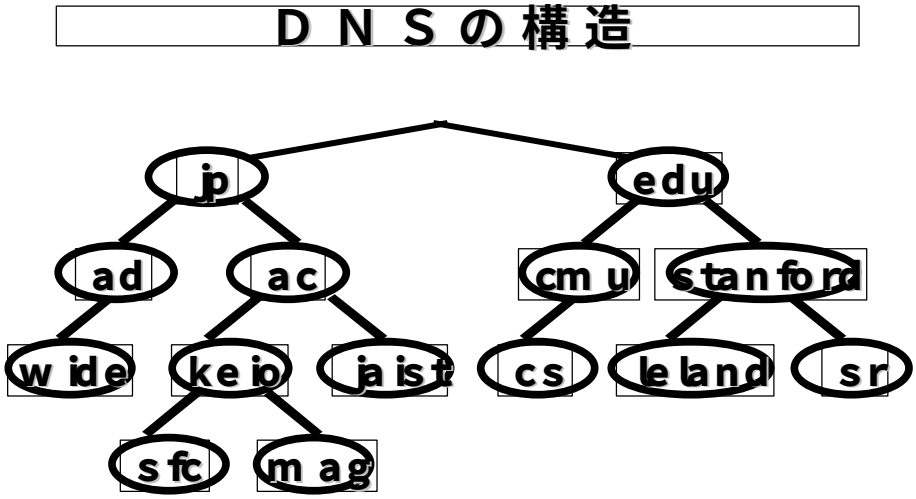
混雑をしたときに何をすべきかを考える事を、「輻輳制御(コンジェションコントロール)」と言いますが、先程のウインドウコントロールで、ネットワークを超えてデータがやり取りされる時は、これを輻輳制御だと思うことができます。通信相手の能力が一杯だったり、ネットワークの真ん中が混んでいれば、データ転送が遅くなり、問題がなければ早くなるので、遅くなったら優しく送ってやり、早くなったら太く投げる。このようにすると、フローコントロールのメカニズムなのですが、きちんと輻輳制御の役割もし始めるのです。
インターネットに参加する人が増えて混雑が起こると、返事が遅くなるので輻輳制御が働き、小さなデータを送るようになります。小さなデータばかりになると混雑は解消され、混雑が解消され返事が早く返ってくるので、又輻輳制御が働き一斉にデータを大きくします。そうすると又.混雑し、.....という具合に繰り返されます。
そこで新しいルールをくわえました。要するに空いても急ぐなということで、このアルゴリズムのことをスロースタートといいます。これはインターネットの最初の設計には入っていませんでしたが、あとから加わりました。これが今のインターネットを救っているのです。

3.インターネットの運用技術

名前とIPアドレス

IPアドレスは数字で人間には覚えにくいので、人間に覚えやすい名前をつけました。つまり、名前からIPアドレスが引けたり、IPアドレスから名前が引けるという仕組みさえ持っていれば、私たちは相手を識別するのに、IPアドレスを使わなくても名前を使うことができるようになります。

DNS



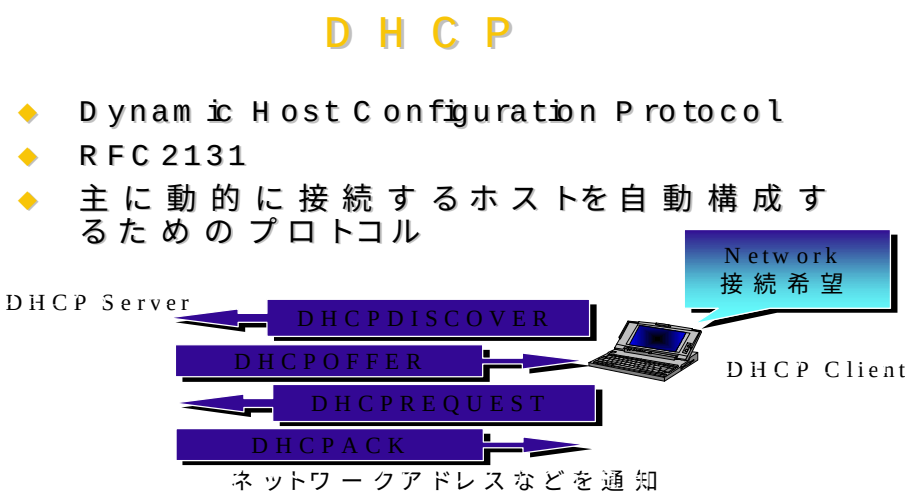
DNS(Domain Name System)という仕組みでわれわれは今作っています。これもスケールの問題です。どんな名前も誰でも自由に付けられるように、名前の空間がどんどん定義できるよにということで階層的な名前の構造を考えました。

その名前からIPアドレスを引いたり、その逆をするというのがDNSの仕組みです。

私たちは長い間これはIPアドレスを名前として覚えやすいようにするためのメカニズムだと思って、最初に来た人に名前を渡しています。ところが最近アメリカで商標知的所有権問題として裁判が起こりクローズアップされました。

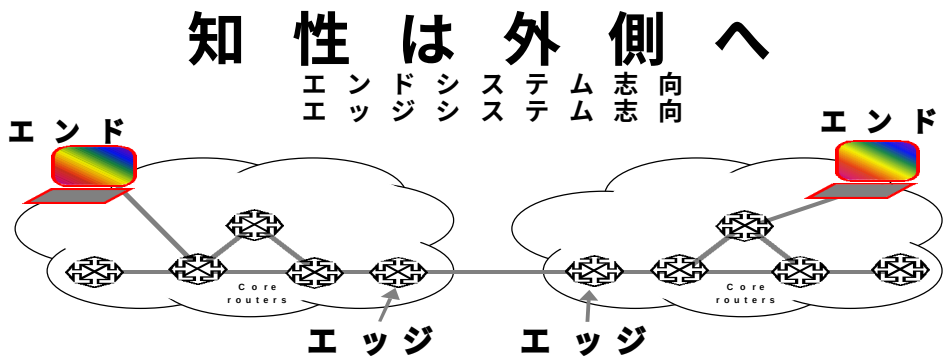
今日はアーキテクチャの話ですから、技術的な機能を話題にすれば良いのですが、社会に出てきた時には、知的所有権の問題と深く絡み合ってくるのです。それを解決する新しいアーキテクチャを考えていかなければならない時代なのだと思います。

DHCP



IPアドレスを動的に振りたいという要望があります。ラップトップは繋いだところでIPアドレスが振られて欲しいというのがその例です。
これに関連した技術にモバイルIPという技術がありますが、これもアーキテクチャ上大変難しい問題です。ダブルミーニングもあります。これからのインターネットをどうするかを考えたときに解決していかなければならない問題です。

QoS(Quality of Service)

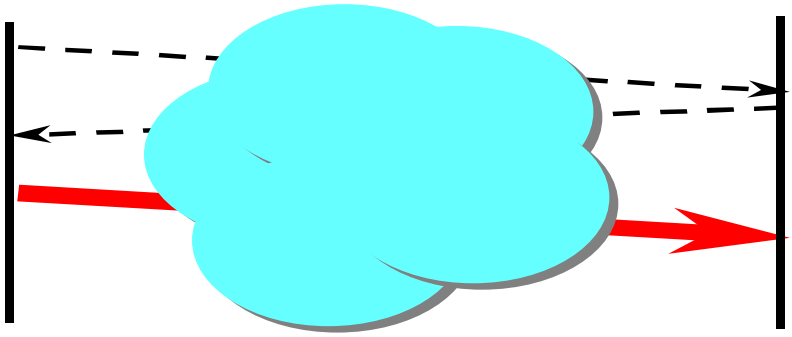


フローコントロール、コンジェションコントロールに説明を戻します。今日一番重要な話だと思います。アーキテクチャの心として私が最初に言ったキーワードの中に、エンドシステムとエッジシステムという言葉があります。エンドシステムは皆さんのコンピュータあるいは皆さんの相手をするサーバコンピュータです。エッジシステムという言葉は耳新しいと思いますが、ある自律的なネットワーク(サービスプロバイダのネットワークを考えるのが一番わかりやすい)と繋がっている端のコンピュータの事をいいます。

インターネットはこれから音声、オーディオ、ビデオ、そのほかの新しいメディア、放送型のメディアを乗せていかなければなりません。そのためには、これまでのBest Effortアーキテクチャだけではだめでしょう。でもスケールするためには真ん中のシステムに負担をかけるわけにはいかないので、何とかエッジシステムの必要最小限のコントロールで、今まで回線交換でしかなしえなかったことをインターネット上で実現したい。又、どうできるかという挑戦をしていかないと新しいインターネットはできないでしょう。

そのためにこのエッジシステムというものの役割を考えましょうということが、今から申し上げたいことです。

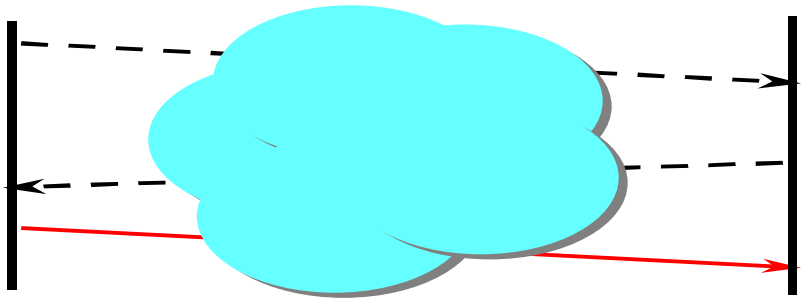
輻輳制御ルール1:返事で判断



返事が早い→空いてる→沢山送る

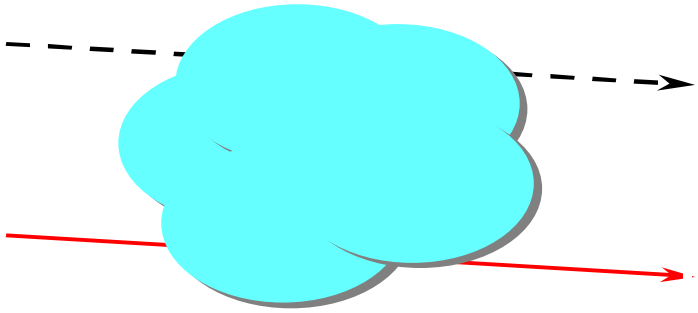
インターネットは混みます。混むから返事が早く返ってきたら基本的にはたくさんの情報量を
トラフィックに流していいと。こういうふうに思っていきましょう。それでやがて混んできて、返事が遅くなったら混んだと思って、それでデータ量、流すデータ量を制限しましょう。皆さん、TCPはとてもいい子なので網が、世の中が混んだと思ったら利己的なことは言わずに、主張せずに自分の出すデータを、遅くなるけれどしようがない、小さいデータを投げましょうと、いい子で言ってくれます。そして先ほど言ったように空いても急ぐのは、急いで大きくするなと。これも守ってくれます。これを守ってくれるのがインターネットでは、先ほども言っているようにトラフィックの90パーセントです。だからどんなに混んでもほとんどつながります。

輻輳制御ルール2:混んだら小さく



返事が遅い→混んでる→少し送る

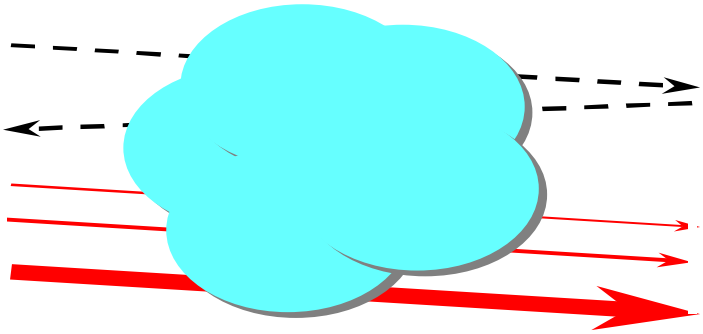
輻輳制御ルール2:混んだら小さく



返事が無い→混んでる→少し送る

返事がなかったらやはり遅いのといっしょだから少し控えましょうと。

輻輳制御ルール3:空いても急ぐな



返事が早い→空いてる→ゆっくり送る

皆さんがやるのはこれだけです。

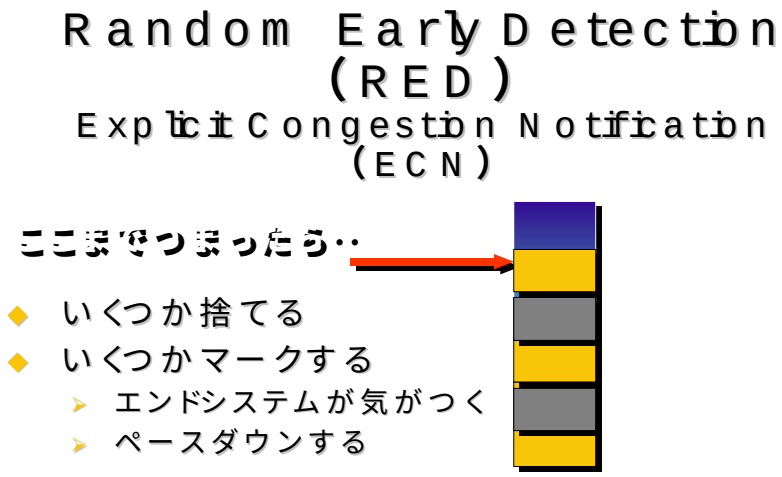
「インターネットが遅い」、何が遅いかというと「相手がのろい」か「途中がのろい」かどちらかです。相手がのろい場合というのは相手が過負荷になっていて、要は大きすぎるのだと。これを何とかしなければいけません。それから途中がのろいというのは何かというと、実はどこかの途中のルータの待ち行列があふれているということです。どういうことかというと、ルータが持っているキューの長さというのは一定ですから、それにあふれた途端にそれをこぼします。けれども途中の中継ノードのIPは、IPのところで言ったようにこぼしていいんです。リソースがなくなったら捨てていい。Best Effortですから捨てていいです。捨てると、先ほど言ったように全体としてはのろくなります。問題なのは捨てられたことでしょう。捨てられたものを流したTCPは、何をしなければいけないかといいますと、着かなかったのだから再転送をしなければいけません。再転送をすると、もしこれが混んだままならもう一度同じことが起こります。そうするとネットワーク全体の有効データ量を考えてください。何度も送り直して何度もこぼれているのだから、全体の有効データ量はネットワークが一生懸命キャリーしている全体のトラフィックに対するパーセンテージ、本当に必要だったデータのパーセンテージはものすごく小さくなってしまいます。
というわけでこれを起こしていけません。ここがいちばんのポイントで、ではどうすればいいのかというと、Differentiated Serviceという考え方です。

Differentiated Service

- **今までのインターネット**
  - **ベストエフォート**
    - **届くように努力する**
    - **コアシステムに負荷が無い**
- **サービス保証システム**
  - **契約したサービスを保証する**
  - **コアシステムの負荷が大きい**
- **これからのインターネット**
  - **ベストエフォート**＋DiffServe
  - **コアシステムに負荷をかけずに優先度のあるサービスを提供**

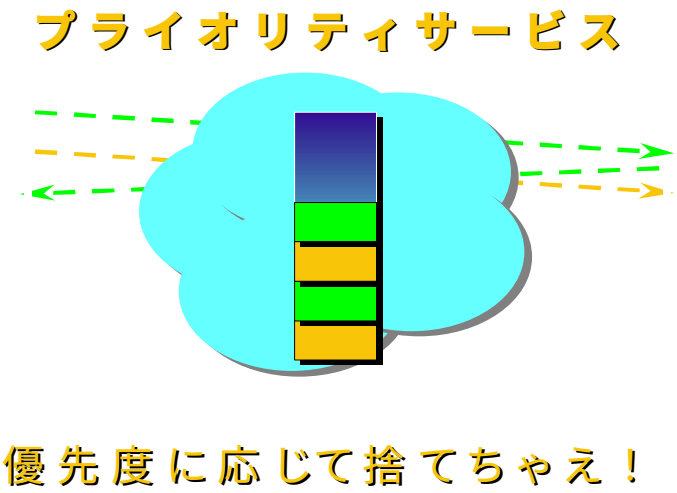
次に1999年からできる、Differentiated Serviceという考え方を紹介します。
今までのインターネットのアーキテクチャは、今まで説明したようにBest Effortです。届くように努力すること。コアシステムに負担がないことにする。これがこのアーキテクチャの心でした。それでサービスを保証するシステムというのは、契約をして契約者サービスを保証する代わりに、コアシステムの負担は大きく、課金をきちんとしなければいけないということです。

これからのインターネット・アーキテクチャの大きなチャレンジは、Best Effortと Differentiated Serviceという二つのことを組み合わせたトラフィックを流すことによって、放送だとかビデオ、オーディオを乗せていこうということです。これはコアシステムには負担をかけずに、エッジシステムにはちょっと負担をかけるけれども、優先度の制御ができるサービスをするという事です。
原理は簡単です。



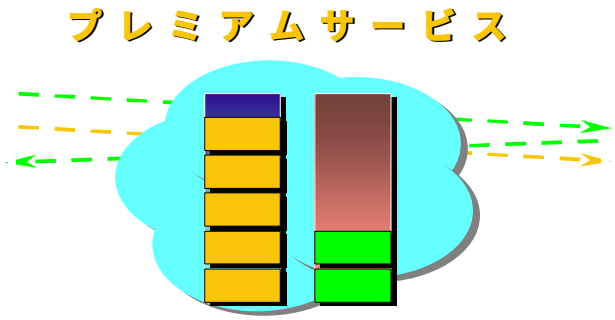
真ん中のキューがあふれそうになったとします。あふれさせたくないためにはどうするかというと、二通りのことを中継ノードはします。一つの方法は捨てることです。Best Effortだから捨ててもいいのです。さてもう一つ方法を、RED(Random Early Detection)といいます。あふれそうになったら、パケットにマークをつけて送ります。そうすると反対側のエンドシステムは途中がもうすぐあふれそうだということがわかるのです。そこでエンドシステムのアプリケーションプロトコルがトラフィックを下げてくれればインターネットは生き残るわけです。
今までのアーキテクチャは、エンドツーエンドで同じ階の人と話します。ところがこれはどうでしょう。真ん中のキューをコントロールしているものがエンドシステムのアプリケーションに対して、真ん中があふれそうだから何とかしてということを言っています。アーキテクチャとしては新しいです。
これは、真ん中のノードがキューのコントロールで明示的(Explicit)にコンジェションが起こっていることを伝えています。これをECN(Explicit Congestion Notification)といいます。
それでは、これで何ができるようになるかです。

プライオリティーサービス



まず、「プライオリティサービス」ができます。例えば、ファーストクラスとビジネスクラスとエコノミークラスのようなもので、Best Effortのものとプライオリティのあるものとのサービスが分けられるのです。 ISPはものすごいエコノミーサービスを作ったときには、Best Effortのキューに入れます。そしてそうでないときには、少し混んでいるときでもスムーズに動くというサービスクラスを作ることができるようになるということですが、これはビデオだとかオーディオだとかに応用できてきます。

プレミアムサービス



そしてもっと明快なこともできます。今のルータは実はキューを複数持つということができるようになっていて、一つのイーサネットならイーサネットの出口に待ち行列を複数用意しておきます。そこである印が付いているものは特別なキューに入れ、他のはBest Effortキューに入れます。

そしてスケジュール、すなわちこのキューをさばいていくのが、順番として特別なキュー

がなくなってからBestEffortキューを送る。これが先ほど言った、Best EffortとDiffServeの共存の一番わかりやすい例です。
こうすると帯域保証と同じことが起こります。これを「プレミアムサービス」といいます。これは帯域保証ではありません。なぜかというと特別キューのトラフィックがなくなったら必ずBestEffortキューのトラフィックが行くのです。ただ、Best Effortの方は細くなったと思うことになります。回線交換のサービスと同じようなクオリティの保証やサービスの種類を何とかインターネットで作り出すことができればインターネットのアーキテクチャで、電話、ファックス、テレビの電波、衛星、等の帯域を確保しなければできなかったことが、できるようになるだろうということが一つの挑戦なわけです。

4．通信媒体とインターネットアーキテクャ

1999年からインターネットで新しいことができるようになります。実は、EDとECNの技術は既に主なルータには実装されています。そして、ISP間でスペシャルサービスを伝える取り決めがITFで固まってきました。今からのアーキテクチャは、REDとECNを使ったDiffServeを含めたアーキテクチャで考えていくことになるのです。

これからいろいろな通信技術が出てきます。 光ファイバ、ＡＴＭ等の新しい技術が出てきます。しかし、それで私たちのトラフィックは本当にどうなるでしょう。3.4kHzの電話の音声のトラフィックは、上限があります。ファックスのトラフィックも上限があります。それに対してコンピュータとインターネットが生成するトラフィックには上限がありません。これは何を意味しているかというと、トラフィックに上限のない 汎用のディジタルコミュニケーションインフラストラクチャが出現するという事です。それはインターネットと呼ぼうと呼ぶまいとディジタル情報である限り、オーディオ、電話、ディジタルテレビ放送等が混在してインフラストラクチャを共有できるはずです。

そのインフラストラクチャーを共有するメディアは何が出てくるかというと何でもいいのです。これがアーキテクチャで一番説明したかったことの一つです。そしてその特徴をうまく使えば何でも良いのです。サテライトが同報性に強ければ、同報性の良いトラフィックはサテライトにルーティングをすれば良い。光ファイバの方が遅延が少ないのならば、光ファイバを使った経路に回せばいい。このような事が、先ほど説明した、経路制御、アーキテクチャで可能な事はおわかりだと思います。

インターネットでは、通信媒体は技術がどんどん進めば良いのです。テレビと異なります。通信媒体レベルの技術が決まっているからサービスが決まり、それで文化が決まり、われわれのコミュニケーションが決まるのではなくて、われわれのコミュニケーションは自由に決めて、コミュニケーションに対して使える通信技術を自由に選ぼうという発展可能な枠を作るために、通信媒体レベルの技術にふろしきをかけた。これがインターネットプロトコルだということです。そうなってくると、先ほどお話しした新しいトラフィックの制

御の挑戦を1999年からインターネットは始めます。新しいアーキテクチャです。それが何を作ってどんな社会を作っていくのかというのは、これは本当にあらゆる分野の人と力を合わせなければできないことだと思います。

最初にお願いしたように、このアーキテクチャを理解して戴いて、新しい時代のインターネットアーキテクチャを作っていくことは、今日ここに参加していただいた皆さん一人一人の課題です。あるいは我々の次の世代がそれを作っていくことになるかと思います。そして、ここまで来られたインターネットアーキテクチャの幾つかは当分の間重要であり続けると思います。
時間がまいりましたのでこれで私の話を終わりにします。

以上