ネットワークを支える技術と使う技術

東京大学 大学院理学系研究科
情報システムチーム 本城剛毅

2023/05/30 木曽シュミットシンポジウム2023

略歴

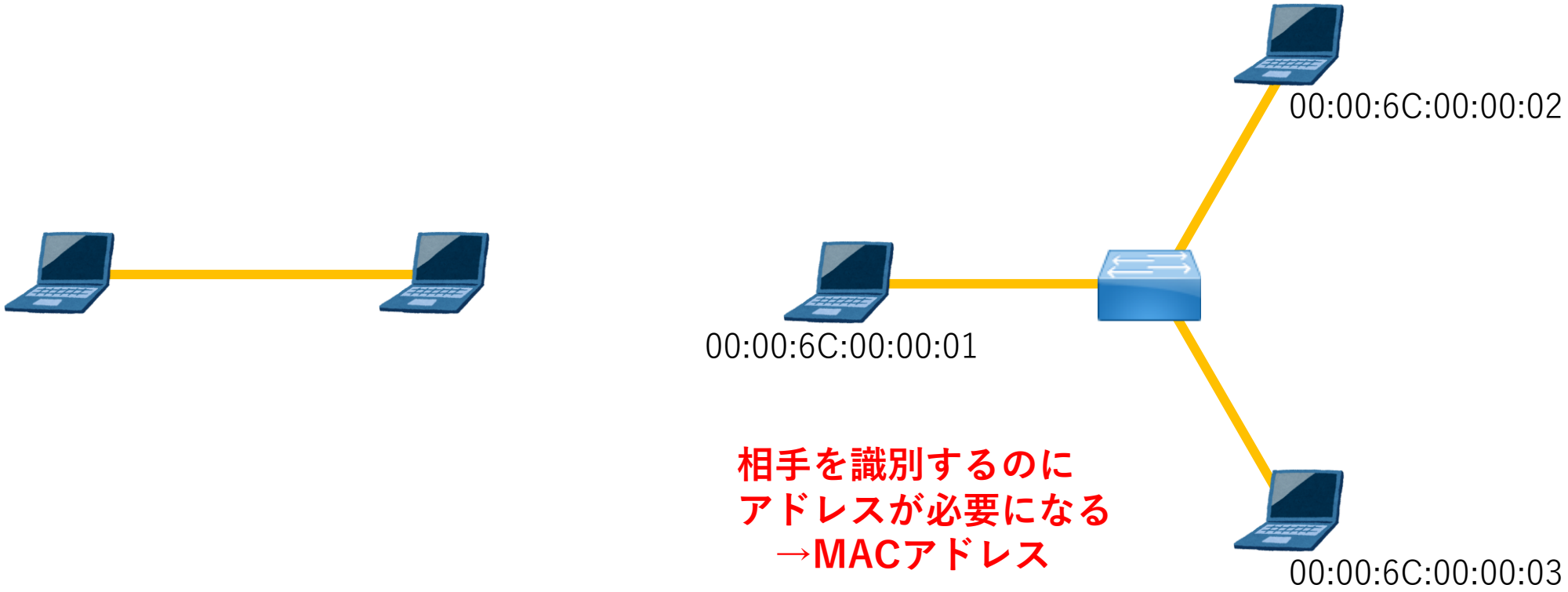
本城 剛毅

- 東大 理学部 情報科学科 卒
- 東大 情報理工学系研究科 コンピュータ科学専攻 修士課程修了
- 東大 情報理工学系研究科 コンピュータ科学専攻 博士課程中途退学
- 東大 理学系研究科 情報システムチーム 助教
  - 情報システムチーム：理学系研究科の情報基盤に関すること、e-learningに関すること、広報支援、講堂等のAV設備管理、情報システム支援、その他

- 業務：WEBシステム・映像配信 etc.
- 研究：コンピュータアーキテクチャ・高速ネットワーク転送

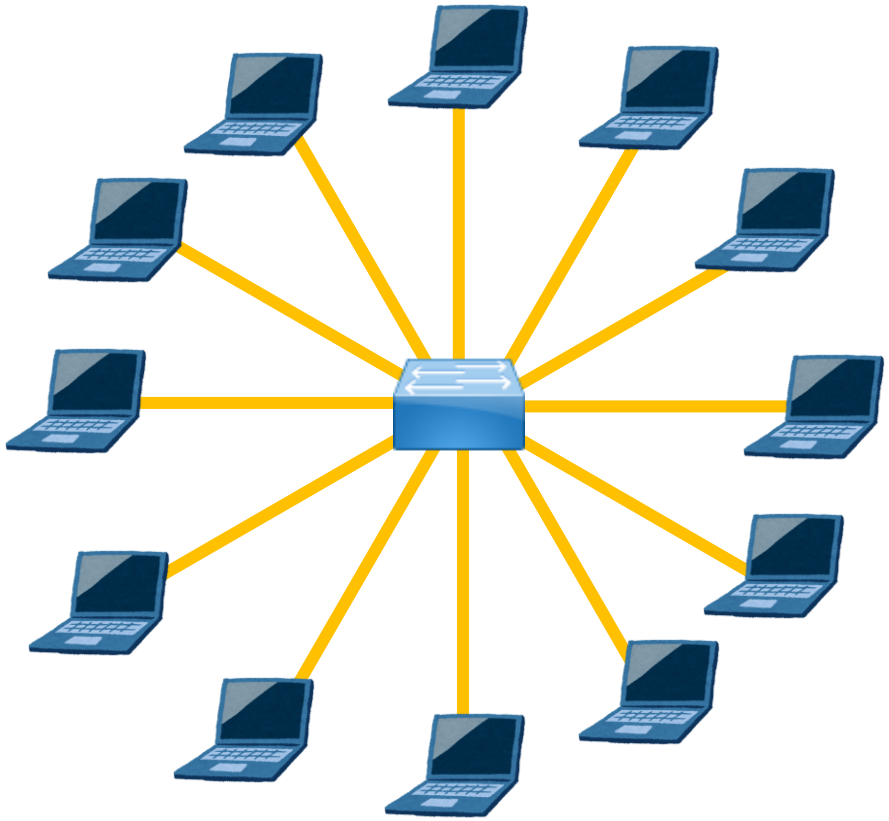
ネットワーク

- コンピュータ同士が結合したもの



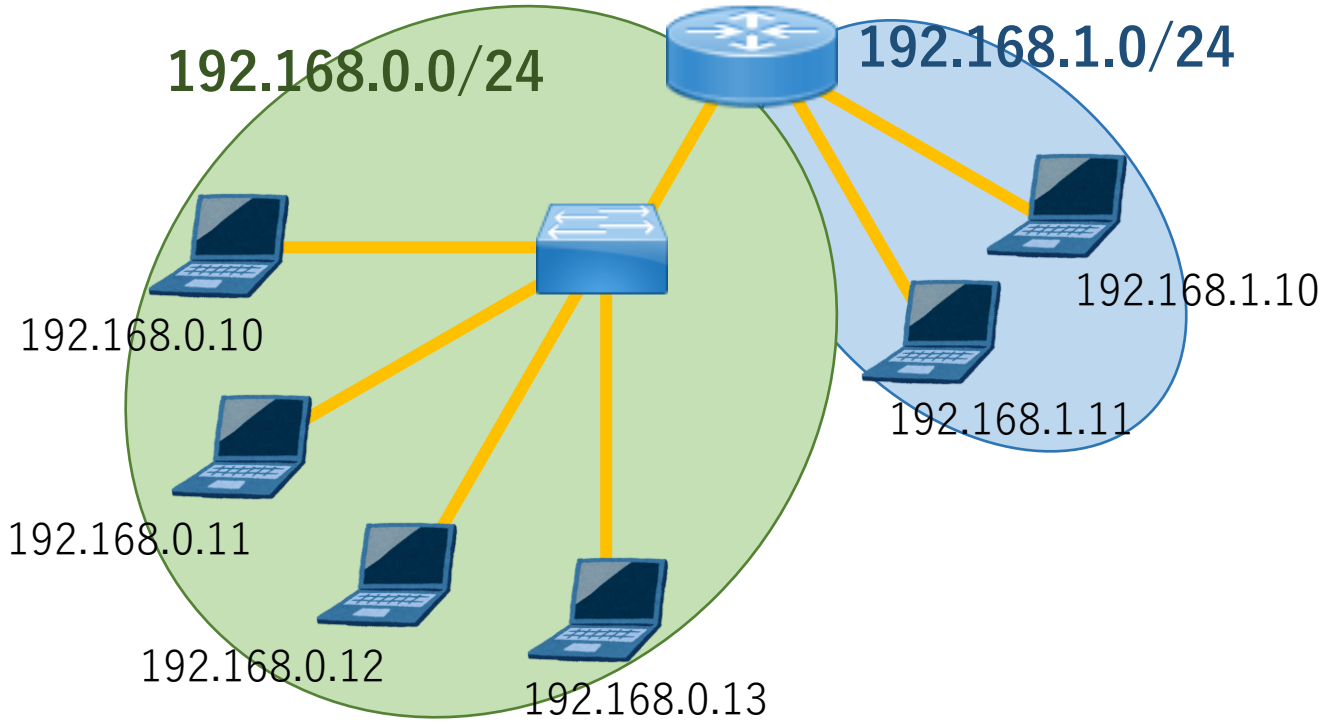
ネットワーク

- コンピュータ同士が結合したもの



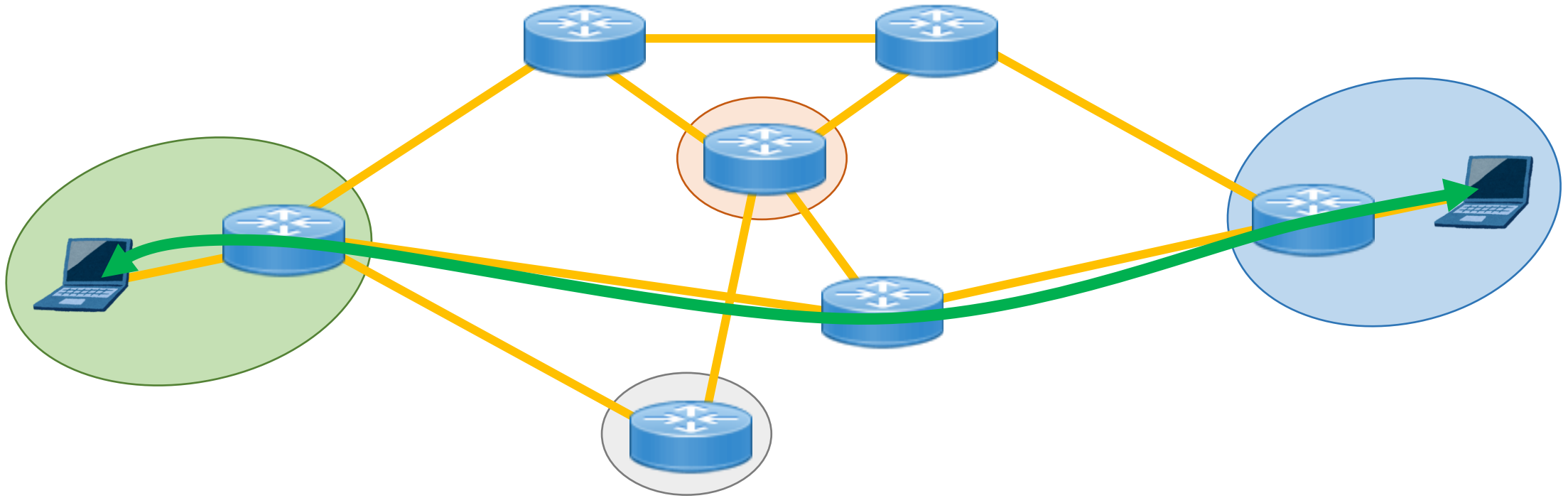
お互いMACアドレスがあれば通信できる

MACアドレスと別の体系として**IPアドレスを使用する**

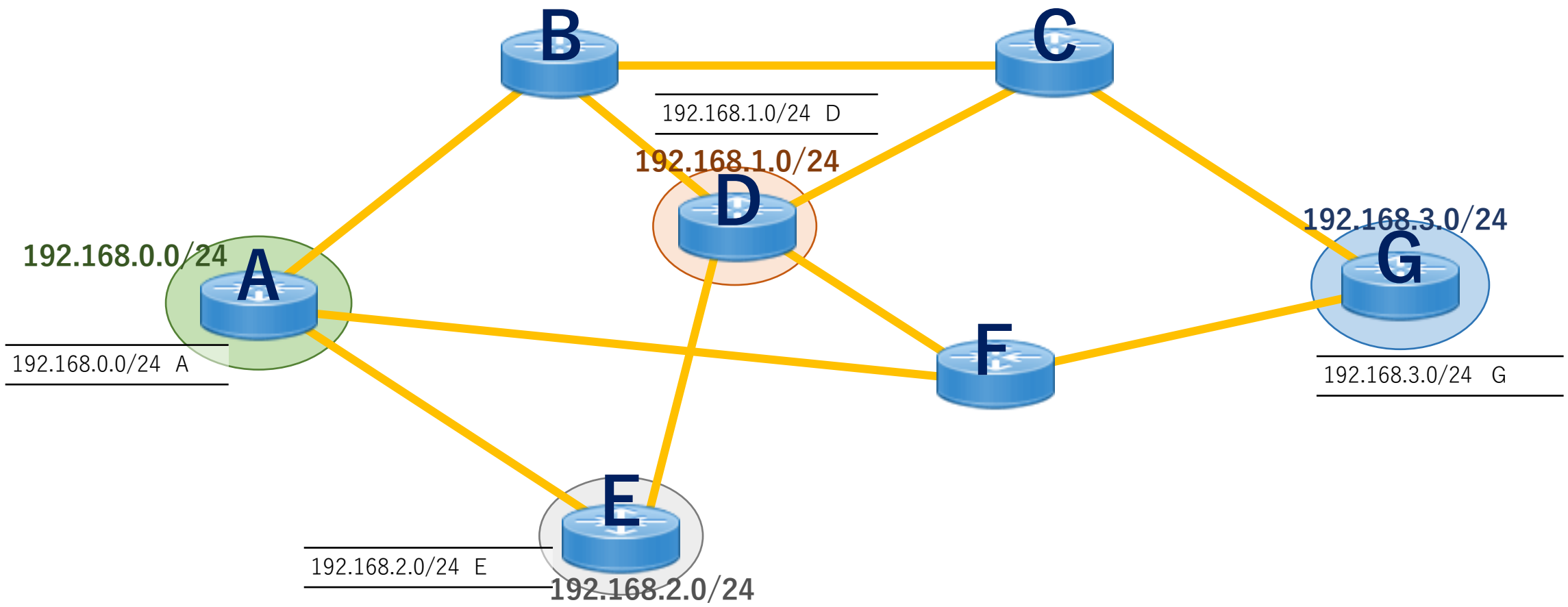


インターネット

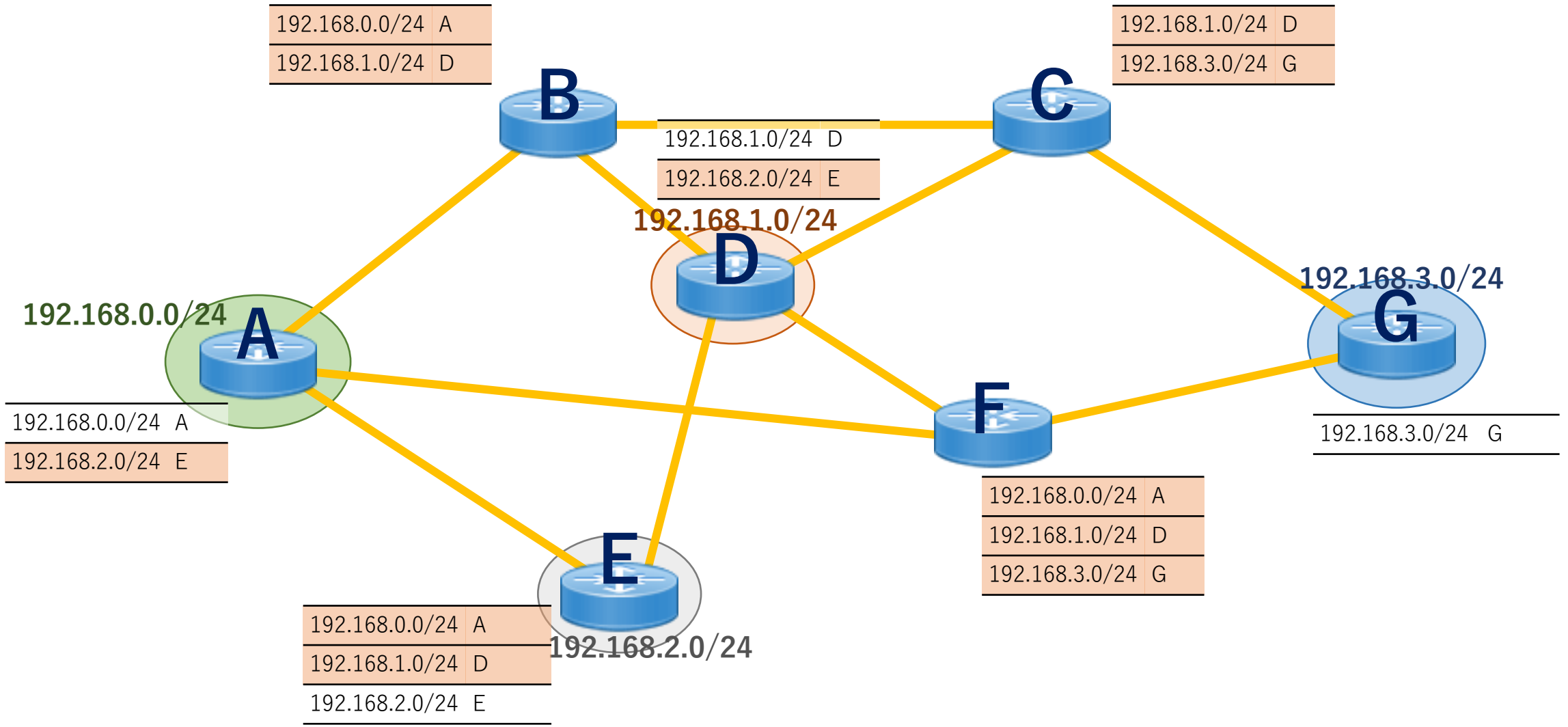
- **ネットワーク同士が結合したもの**
  - この場合、「東京大学」「SINET」「国立天文台」「OCN」「Google」「aws」「Microsoft」…
    - 「天文センター」「東大理学系研究科」とかではない



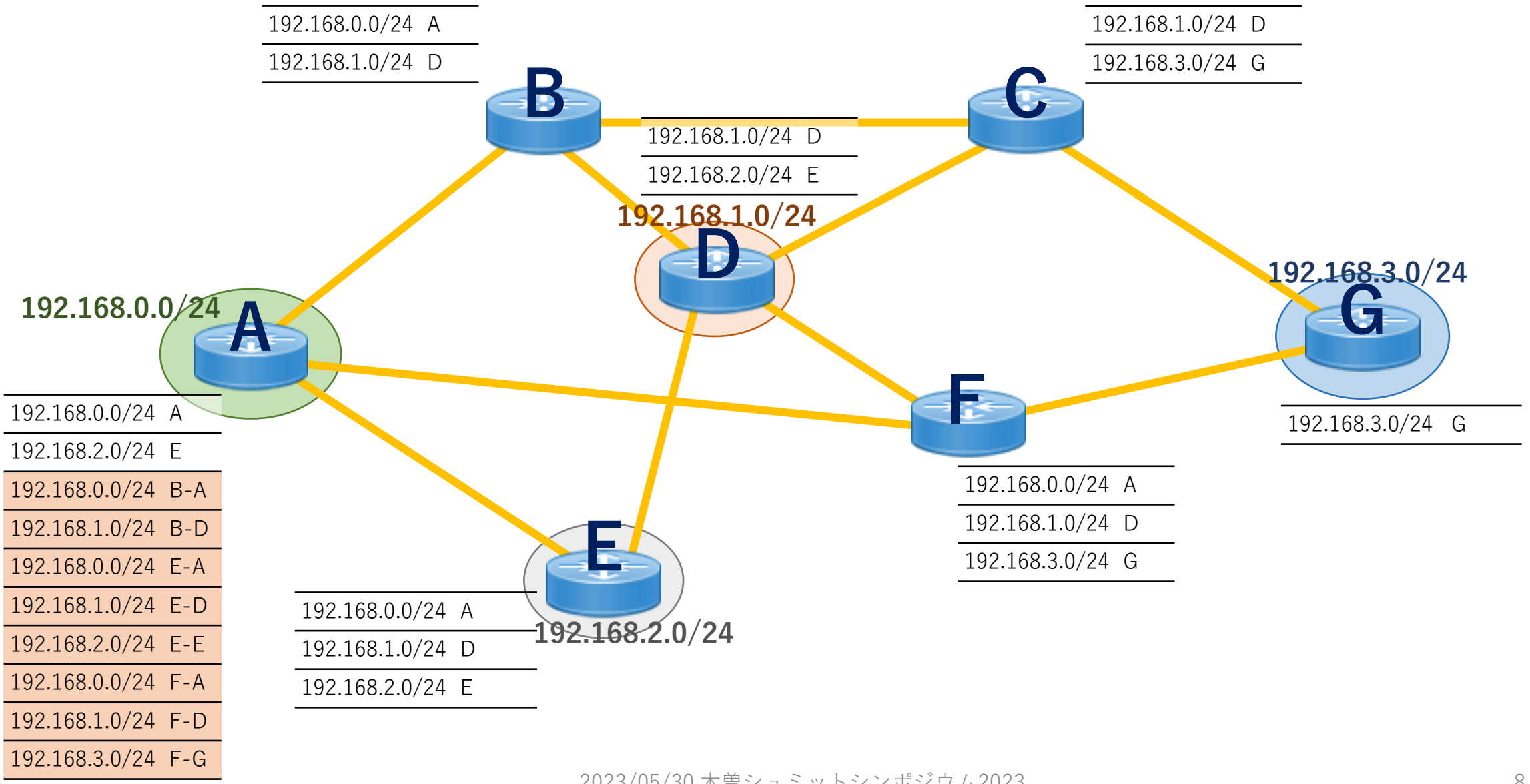
ルーティング


B
C
192.168.1.0/24 D
192.168.1.0/24
D
192.168.3.0/24
G
192.168.0.0/24
A
F
192.168.0.0/24 A
192.168.3.0/24 G
E
192.168.2.0/24 E
192.168.2.0/24

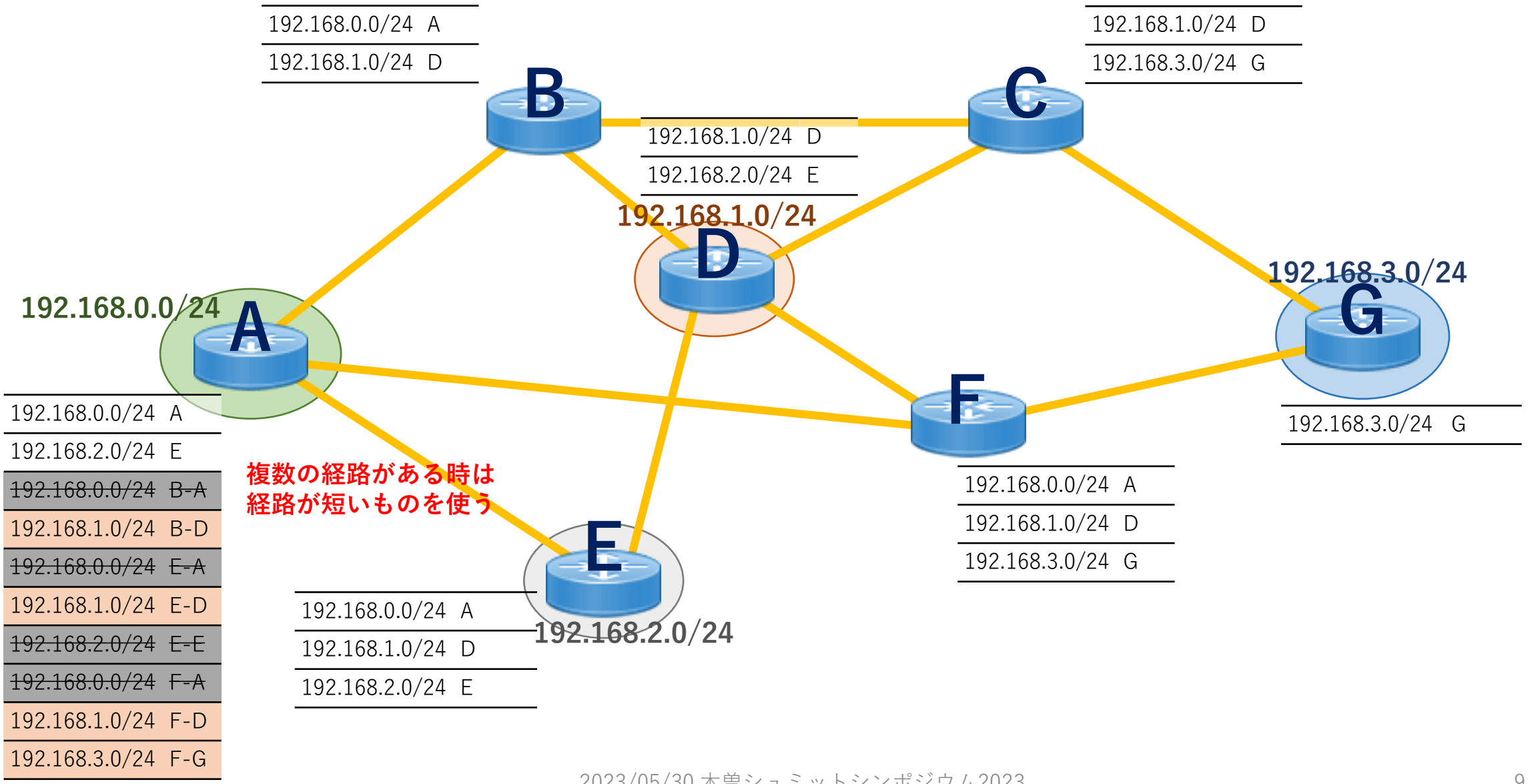
ルーティング


192.168.0.0/24 A
192.168.1.0/24 D
B
C
192.168.1.0/24 D
192.168.2.0/24 E
192.168.1.0/24
D
192.168.1.0/24 D
192.168.3.0/24 G
192.168.3.0/24
G
192.168.0.0/24
A
F
192.168.0.0/24 A
192.168.2.0/24 E
192.168.3.0/24 G
192.168.0.0/24 A
192.168.1.0/24 D
192.168.3.0/24 G
E
192.168.0.0/24 A
192.168.1.0/24 D
192.168.2.0/24 E
192.168.2.0/24

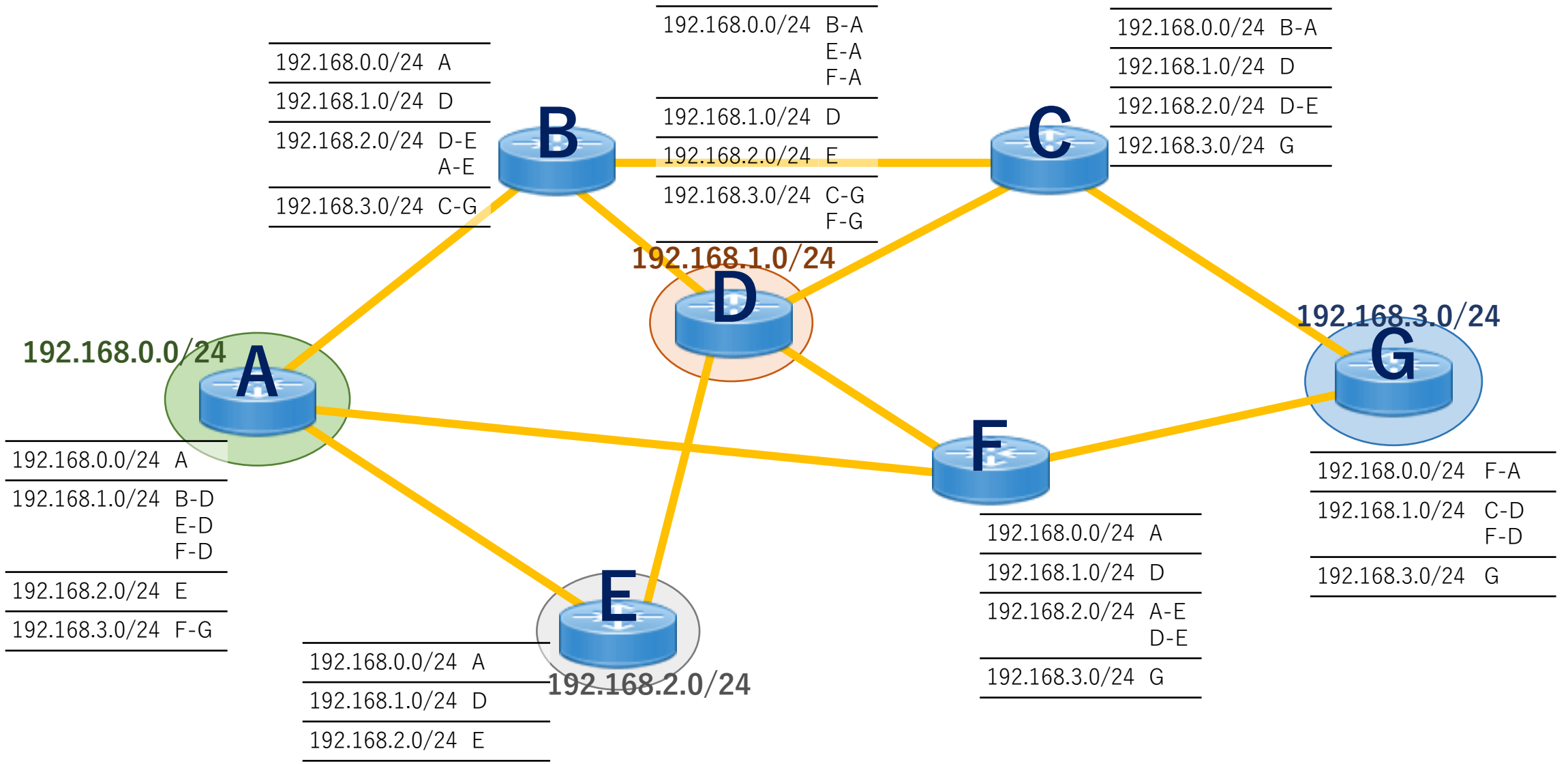
ルーティング


192.168.0.0/24 A
192.168.1.0/24 D
B
C
192.168.1.0/24 D
192.168.2.0/24 E
192.168.1.0/24
192.168.1.0/24 D
192.168.3.0/24 G
192.168.3.0/24
192.168.0.0/24
A
D
G
F
192.168.0.0/24 A
192.168.2.0/24 E
192.168.0.0/24 B-A
192.168.1.0/24 B-D
192.168.0.0/24 E-A
192.168.1.0/24 E-D
192.168.2.0/24 E-E
192.168.0.0/24 F-A
192.168.1.0/24 F-D
192.168.3.0/24 F-G
192.168.3.0/24 G
192.168.0.0/24 A
192.168.1.0/24 D
192.168.3.0/24 G
E
192.168.0.0/24 A
192.168.1.0/24 D
192.168.2.0/24 E
192.168.2.0/24

ルーティング

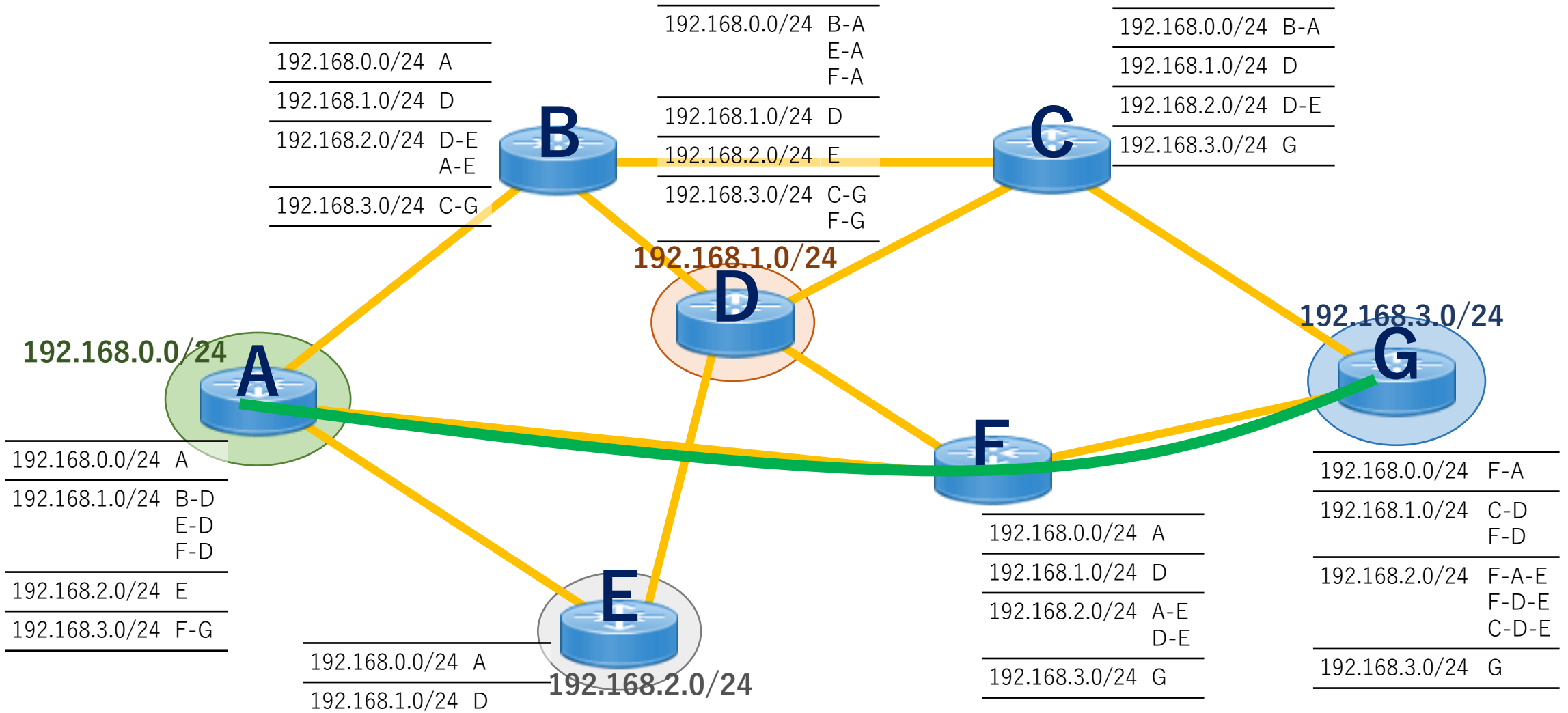

192.168.0.0/24 A
192.168.1.0/24 D
B
C
192.168.1.0/24 D
192.168.2.0/24 E
192.168.1.0/24
192.168.1.0/24 D
192.168.3.0/24 G
D
192.168.3.0/24
192.168.0.0/24
A
G
F
192.168.3.0/24 G
192.168.0.0/24 A
192.168.2.0/24 E
~~192.168.0.0/24 B-A~~
192.168.1.0/24 B-D
~~192.168.0.0/24 E-A~~
192.168.1.0/24 E-D
~~192.168.2.0/24 E-E~~
~~192.168.0.0/24 F-A~~
192.168.1.0/24 F-D
192.168.3.0/24 F-G
複数の経路がある時は
経路が短いものを使う
192.168.0.0/24 A
192.168.1.0/24 D
192.168.2.0/24 E
E
192.168.2.0/24
192.168.0.0/24 A
192.168.1.0/24 D
192.168.3.0/24 G

ルーティング


192.168.0.0/24 A
192.168.1.0/24 D
192.168.2.0/24 D-E A-E
192.168.3.0/24 C-G
192.168.0.0/24 B-A E-A F-A
192.168.1.0/24 D
192.168.2.0/24 E
192.168.3.0/24 C-G F-G
192.168.0.0/24 B-A
192.168.1.0/24 D
192.168.2.0/24 D-E
192.168.3.0/24 G
192.168.1.0/24
192.168.3.0/24
192.168.0.0/24
A
B
C
D
E
F
G
192.168.0.0/24 A
192.168.1.0/24 B-D E-D F-D
192.168.2.0/24 E
192.168.3.0/24 F-G
192.168.0.0/24 A
192.168.1.0/24 D
192.168.2.0/24 E
192.168.2.0/24
192.168.0.0/24 A
192.168.1.0/24 D
192.168.2.0/24 A-E D-E
192.168.3.0/24 G
192.168.0.0/24 F-A
192.168.1.0/24 C-D F-D
192.168.3.0/24 G

ルーティング

実際はA・B…ではなく、IPアドレスやAS番号を用いますが、説明を簡単にするために



192.168.0.0/24	A
192.168.1.0/24	D
192.168.2.0/24	D-E A-E
192.168.3.0/24	C-G

192.168.0.0/24	B-A E-A F-A
192.168.1.0/24	D
192.168.2.0/24	E
192.168.3.0/24	C-G F-G

192.168.0.0/24	B-A
192.168.1.0/24	D
192.168.2.0/24	D-E
192.168.3.0/24	G

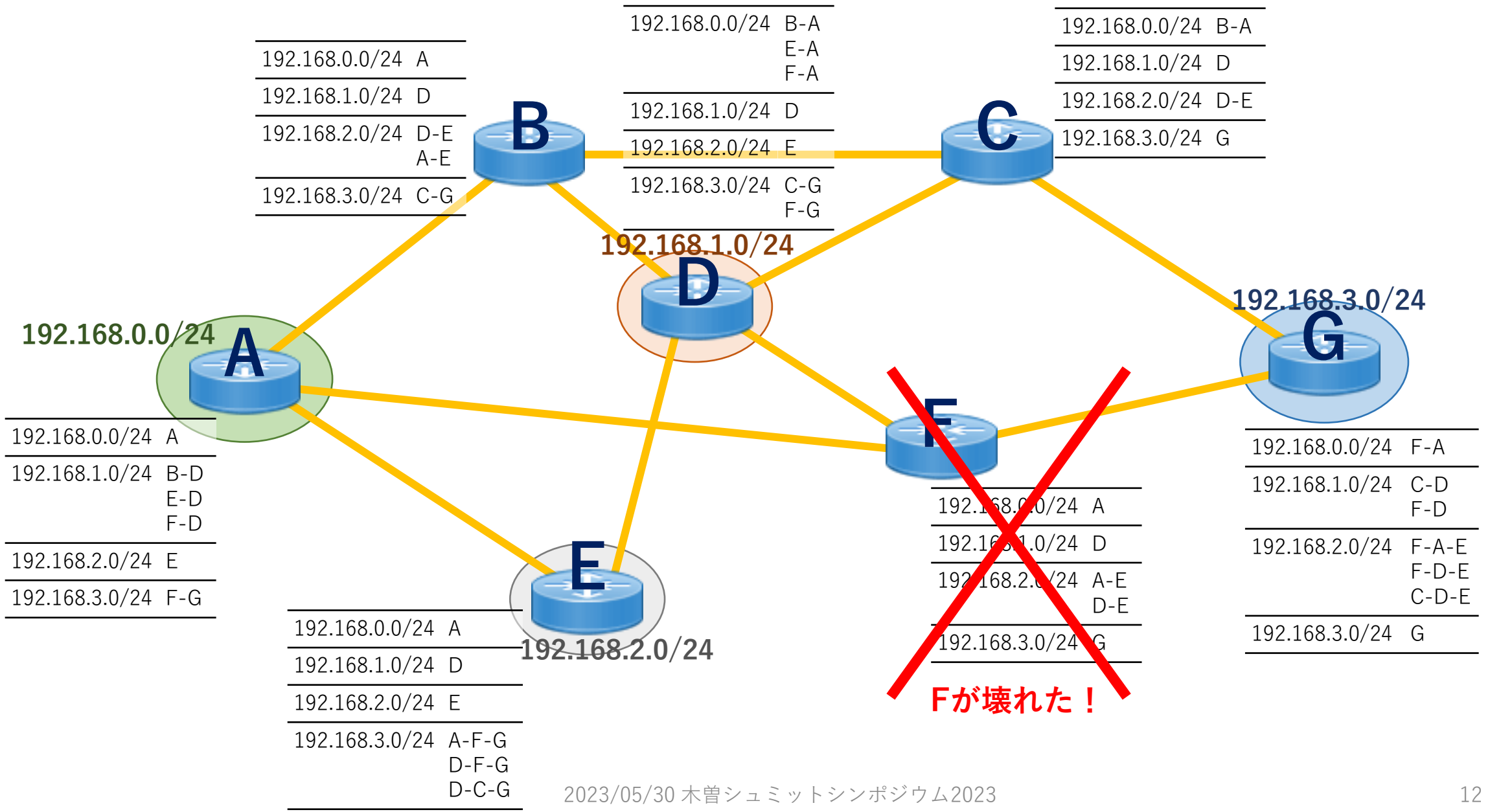
192.168.0.0/24	A
192.168.1.0/24	B-D E-D F-D
192.168.2.0/24	E
192.168.3.0/24	F-G

192.168.0.0/24	A
192.168.1.0/24	D
192.168.2.0/24	E
192.168.3.0/24	A-F-G D-F-G D-C-G

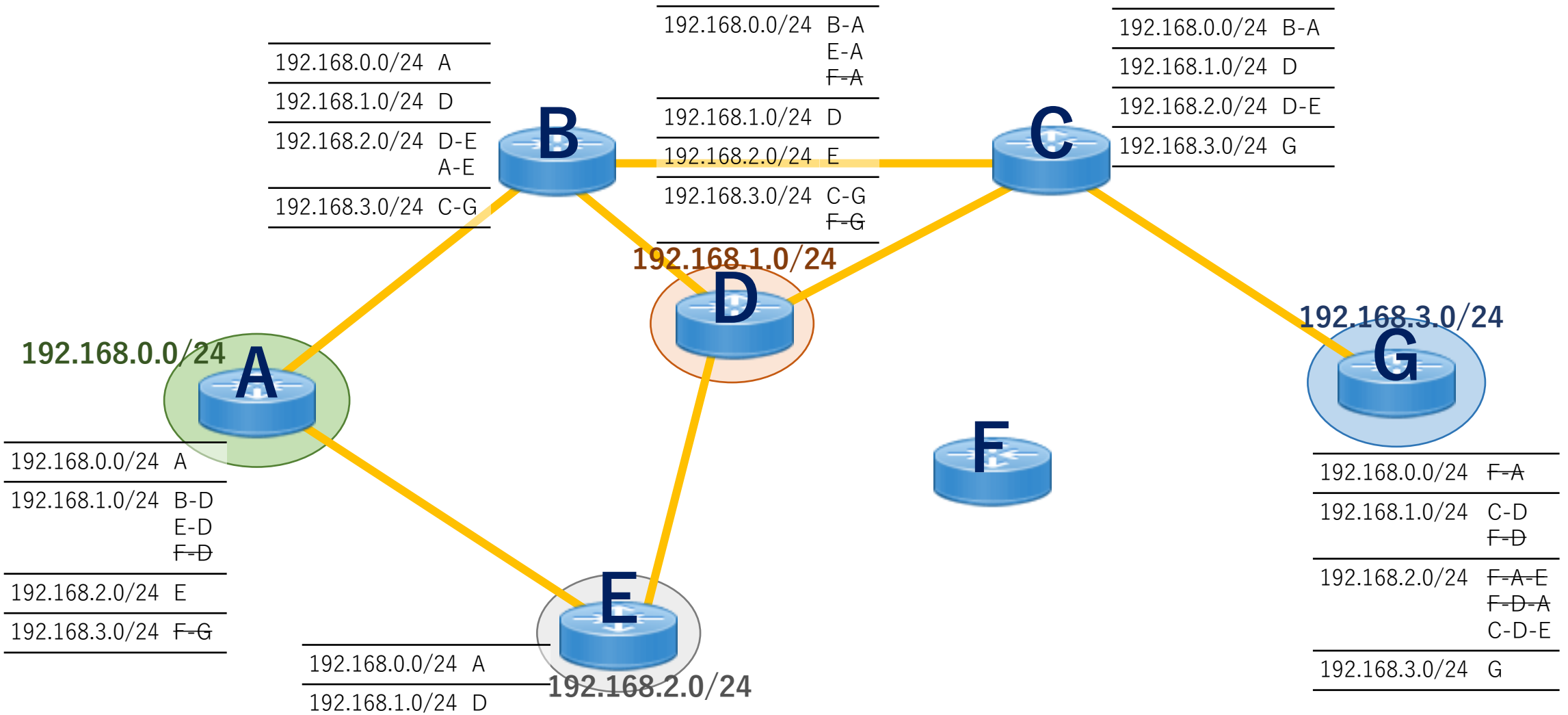
192.168.0.0/24	A
192.168.1.0/24	D
192.168.2.0/24	A-E D-E
192.168.3.0/24	G

192.168.0.0/24	F-A
192.168.1.0/24	C-D F-D
192.168.2.0/24	F-A-E F-D-E C-D-E
192.168.3.0/24	G

ルーティング


192.168.0.0/24 A
192.168.1.0/24 D
192.168.2.0/24 D-E A-E
192.168.3.0/24 C-G
192.168.0.0/24 B-A E-A F-A
192.168.1.0/24 D
192.168.2.0/24 E
192.168.3.0/24 C-G F-G
192.168.0.0/24 B-A
192.168.1.0/24 D
192.168.2.0/24 D-E
192.168.3.0/24 G
192.168.1.0/24
192.168.3.0/24
192.168.0.0/24
A
B
C
D
E
F
G
192.168.0.0/24 A
192.168.1.0/24 B-D E-D F-D
192.168.2.0/24 E
192.168.3.0/24 F-G
192.168.0.0/24 A
192.168.1.0/24 D
192.168.2.0/24 E
192.168.3.0/24 A-F-G D-F-G D-C-G
192.168.2.0/24
192.168.0.0/24 A
192.168.1.0/24 D
192.168.2.0/24 A-E D-E
192.168.3.0/24 G
Fが壊れた！
192.168.0.0/24 F-A
192.168.1.0/24 C-D F-D
192.168.2.0/24 F-A-E F-D-E C-D-E
192.168.3.0/24 G

ルーティング



192.168.0.0/24	A
192.168.1.0/24	D
192.168.2.0/24	D-E A-E
192.168.3.0/24	C-G

192.168.0.0/24	B-A E-A ~~F-A~~
192.168.1.0/24	D
192.168.2.0/24	E
192.168.3.0/24	C-G ~~F-G~~

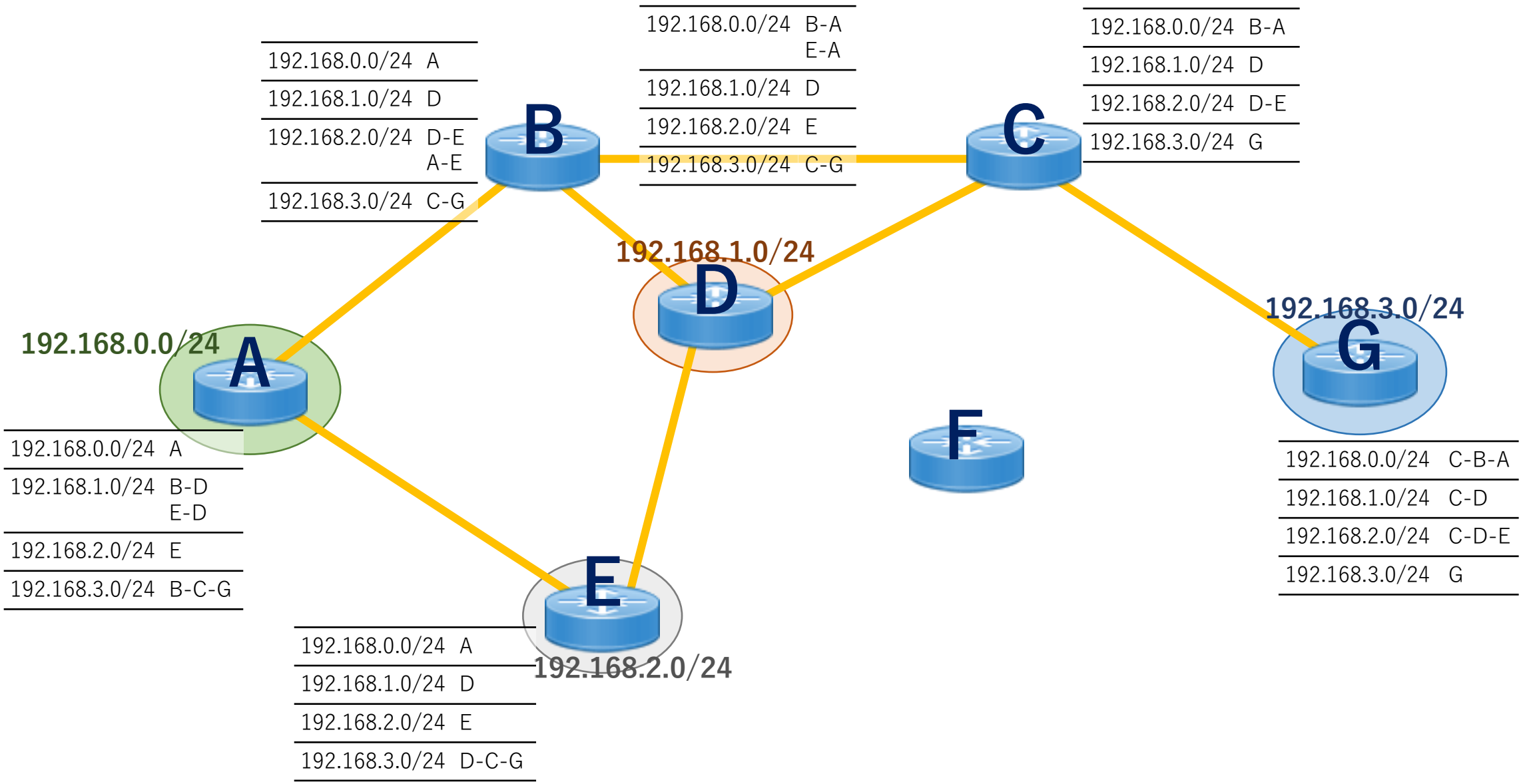
192.168.0.0/24	B-A
192.168.1.0/24	D
192.168.2.0/24	D-E
192.168.3.0/24	G

192.168.0.0/24	A
192.168.1.0/24	B-D E-D ~~F-D~~
192.168.2.0/24	E
192.168.3.0/24	~~F-G~~

192.168.0.0/24	A
192.168.1.0/24	D
192.168.2.0/24	E
192.168.3.0/24	~~A-F-G~~ ~~D-F-G~~ D-C-G

192.168.0.0/24	~~F-A~~
192.168.1.0/24	C-D ~~F-D~~
192.168.2.0/24	~~F-A-E~~ ~~F-D-A~~ C-D-E
192.168.3.0/24	G

ルーティング


192.168.0.0/24 A
192.168.1.0/24 D
192.168.2.0/24 D-E A-E
192.168.3.0/24 C-G
192.168.0.0/24 B-A E-A
192.168.1.0/24 D
192.168.2.0/24 E
192.168.3.0/24 C-G
192.168.0.0/24 B-A
192.168.1.0/24 D
192.168.2.0/24 D-E
192.168.3.0/24 G
192.168.1.0/24
192.168.0.0/24
192.168.3.0/24
192.168.2.0/24
A
B
C
D
E
F
G
192.168.0.0/24 A
192.168.1.0/24 B-D E-D
192.168.2.0/24 E
192.168.3.0/24 B-C-G
192.168.0.0/24 A
192.168.1.0/24 D
192.168.2.0/24 E
192.168.3.0/24 D-C-G
192.168.0.0/24 C-B-A
192.168.1.0/24 C-D
192.168.2.0/24 C-D-E
192.168.3.0/24 G

ルーティング

- **誰か偉い人が一元管理しているわけではない**
- **自動的・自律的に**新しい最適状態に移行できる
- **不適切な情報**が出てきても止められない

- 実際はピアリングなど、別の理由で経路が決定されることも

- **インターネットは、隣の人の情報を信じることで動いている**

TCPとUDP

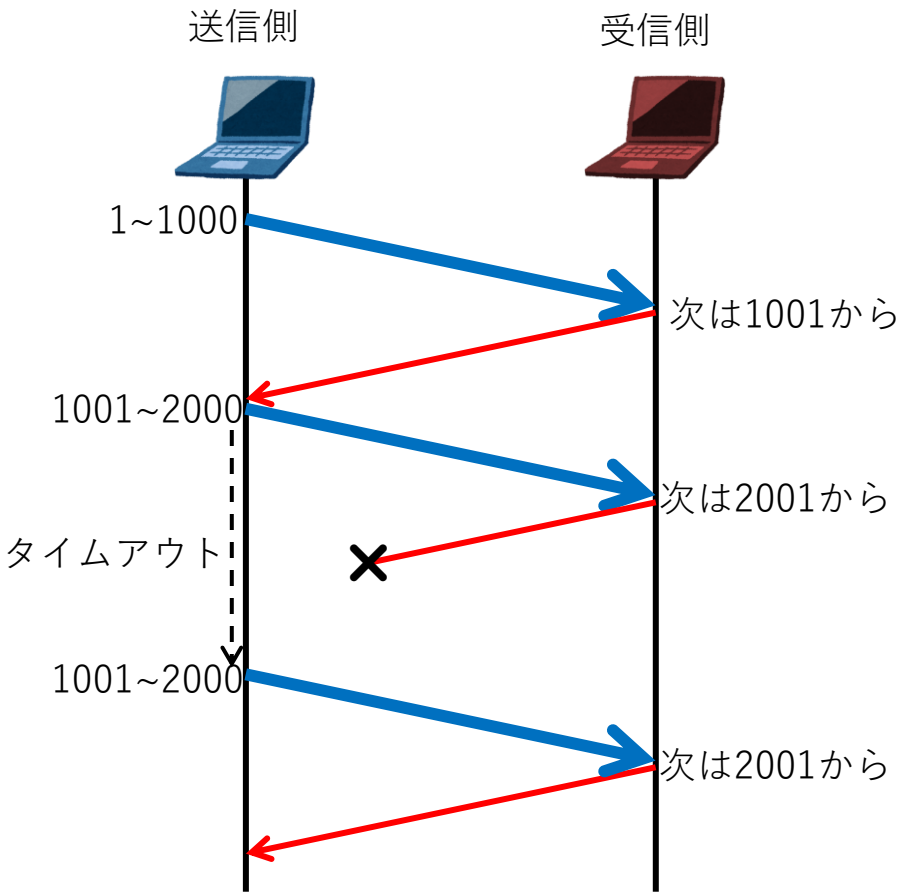
- TCPの特長
  - ストリーム通信（大きなデータ）
    - 到達を保証する
    - 順序も保証する
  - 流量制御
  - 輻輳制御

- UDPの特長
  - 低レイテンシ
  - システム上の負荷が小さい

- 両方が持つ特長
  - アプリケーション間通信
    （ポート番号の概念）
  - データの誤り検出（check sum）

到達保証

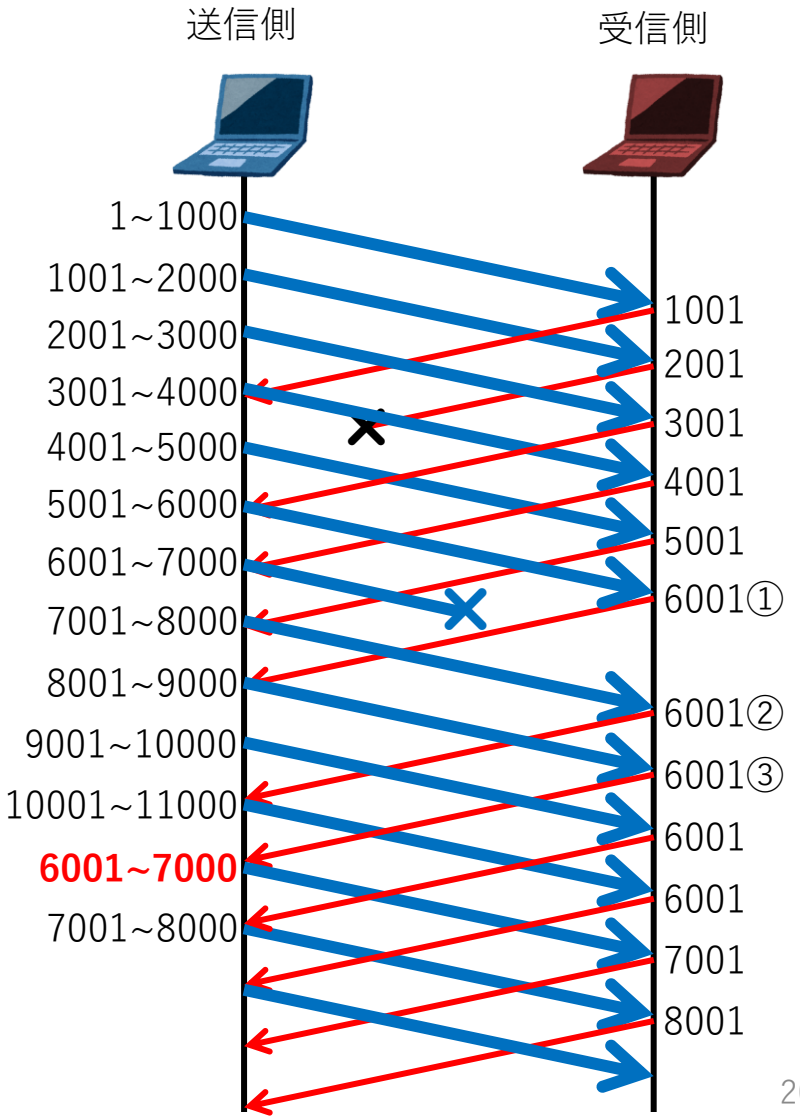
- **受け取ったら受け取りましたと答える**



- 「受け取りました」＝**ACK**と呼ぶ
- ACKが**タイムアウト＝パケットロス**
  - **パケットの再送**が行われる
- 送信側はACKが帰るまではデータを捨てずにバッファリング
- 木曽～本郷間の往復は約4.4 ms
- ＝1000 bytesずつ送ると約1.82 Mbps

到達保証

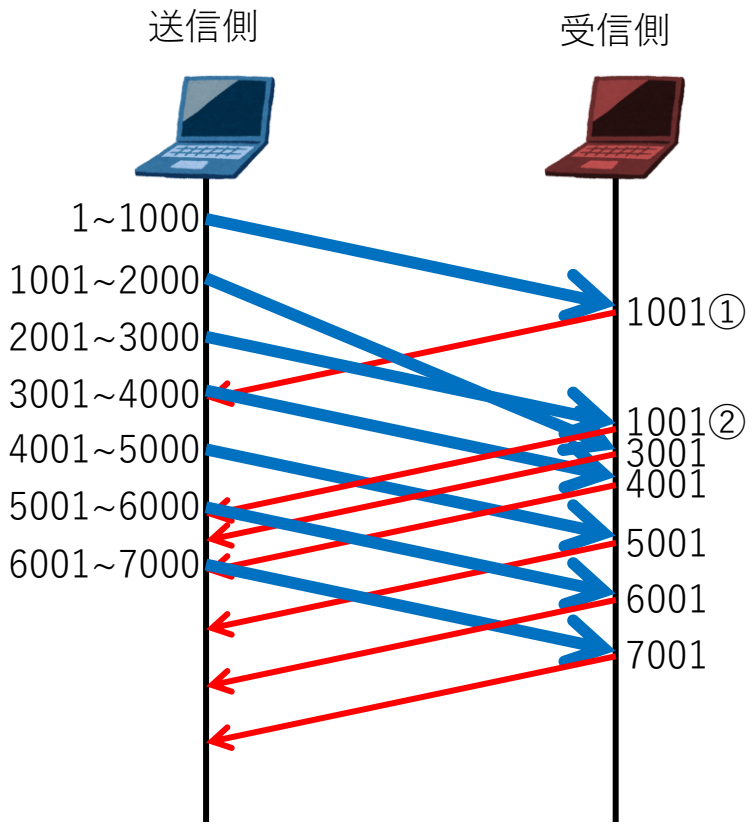
- いちいちACKを待たずにどんどん送る



- ACKがロスしても、後ろのACKがあれば正常に送れているとみなす
- データがパケロスしたら、後ろのデータがいくら来てもACKは進まない
- **3回同じACKが来たらパケットロス！**
  - パケット再送

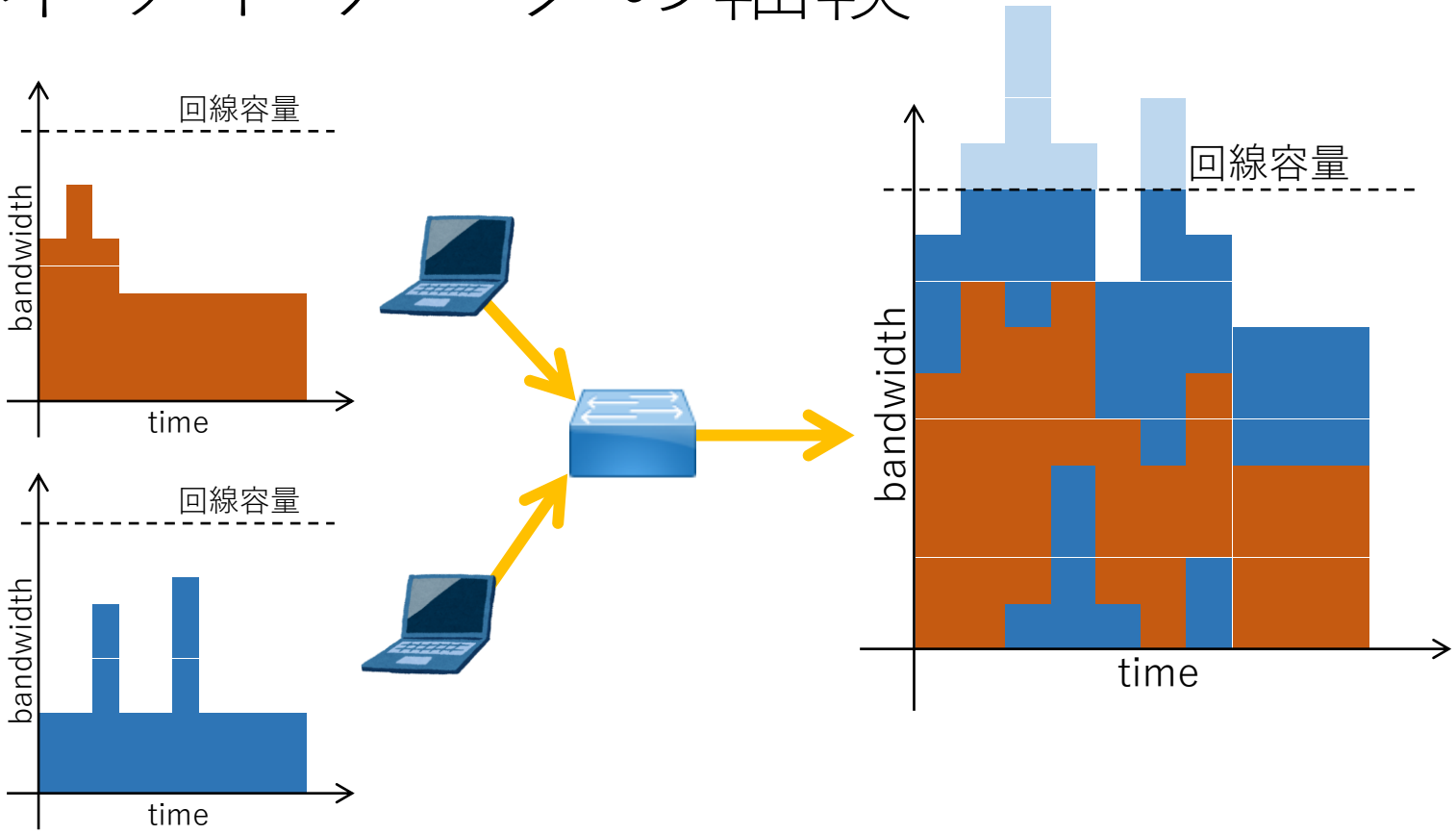
順序保証

- 送るときに番号が付いているので、到着順が違っても並び替えられる



- 受信側にもバッファがある
- 2回同じACKが帰ってくる
  - 3回続くまではロスとは思わない

ネットワークの輻輳



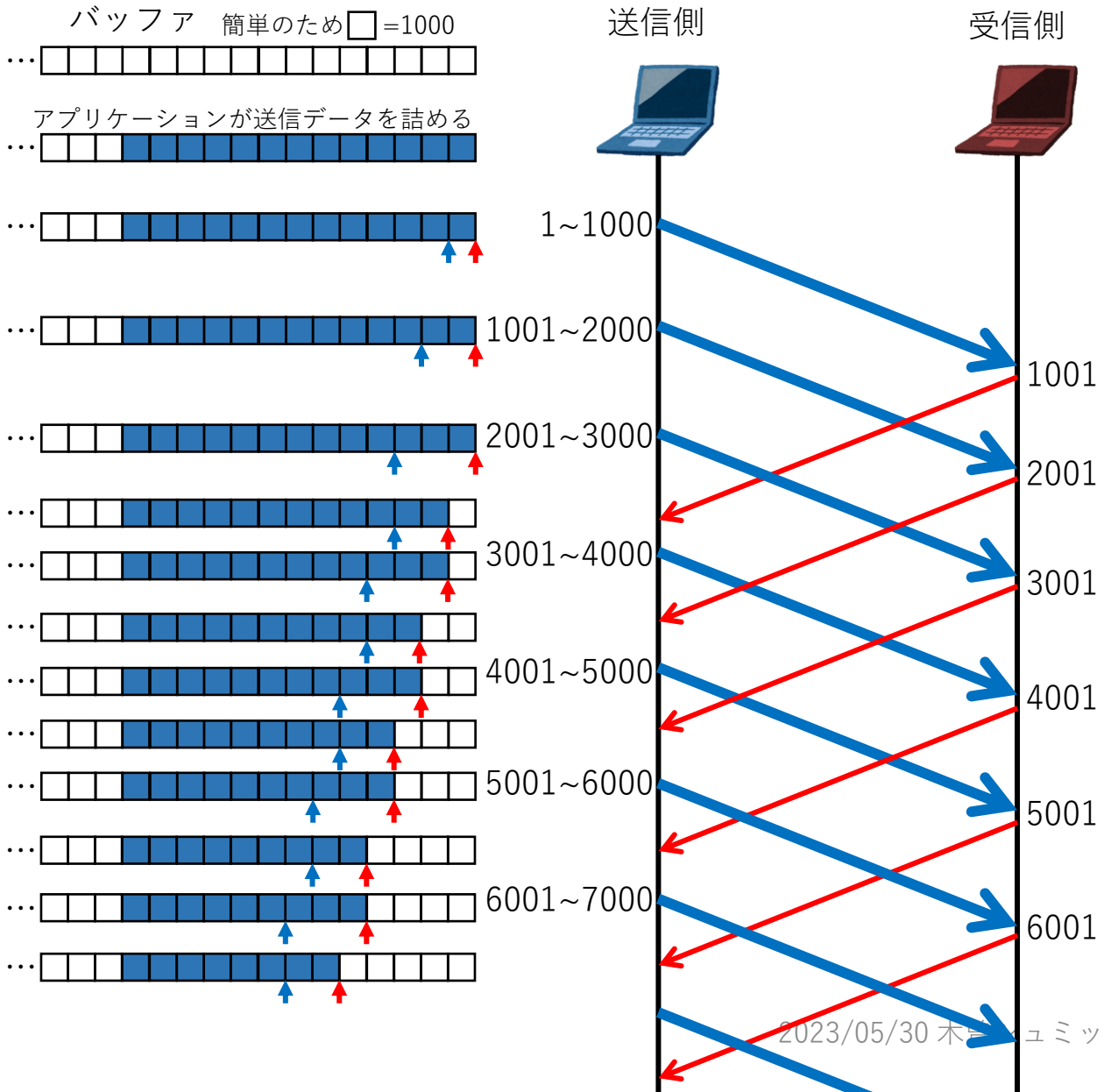
- 一瞬の超過であれば
  スイッチのバッファで
  やりくり
- **バッファ容量を超えると**
  やりくりしきれずに
  パケットを破棄
- 「パケットロス」

Buffalo LSW6-GT-8NS/BK
バッファー容量：192KB
https://www.buffalo.jp/product/detail/lsw6-gt-8ns_bk.html

Netgear XS716T
Packet Buffer：2MB
https://www.downloads.netgear.com/files/GDC/XS708T/XS708T_XS712Tv2_XS716T_XS728T_XS748T_DS.pdf

Dell PowerConnect 8024F
Packet Buffer Memory：16MB
https://www1.la.dell.com/content/products/productdetails.aspx/switch-powerconnect-8024f

送信側のバッファ



- 送信済み（のはずの）ポイントとACKの帰ってきたポイントがスライドしていく
- **「スライディングウィンドウ」**
- **窓の幅＝インフライトデータ量**
- **窓の幅に上限を設定すると、データ転送の帯域を制御できる**
  - 「流量制御」「輻輳制御」

流量制御・輻輳制御

- 流量制御
  - 受信側のバッファの空き容量を通知することで、先方が受け取れないほどのデータは送信しない

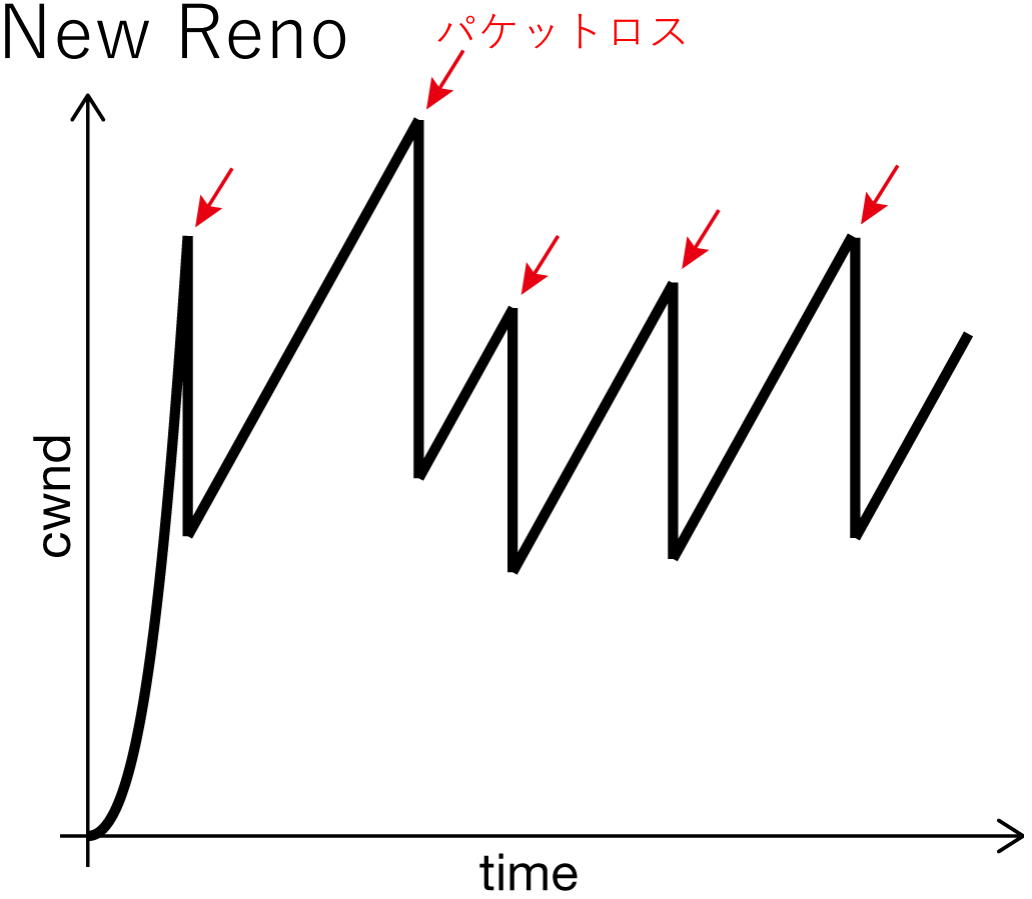
- 輻輳制御
  - 窓の幅を狭めて、流れるデータの量を抑える
  - 様子を見ながら窓の幅を広げて行く
  - コネクションごとの公平性を目指す
    - 最大最小公平

	A	B	C	D
要求量	**2**	**2.5**	**3**	**3.5**
↓	2	2	2	2
	2	2.5	2.5	2.5
最適値	**2**	**2.5**	**2.75**	**2.75**

リソース総量が**10**の時にどう配分するか？

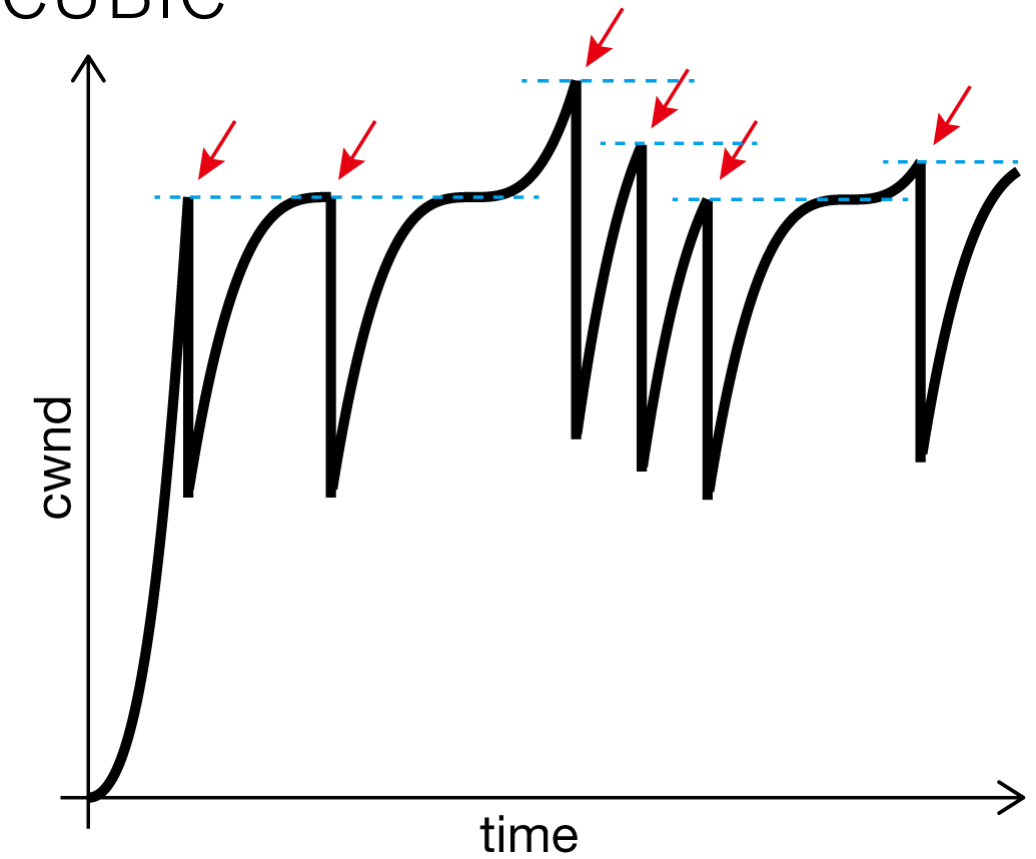
輻輳制御アルゴリズム

- New Reno



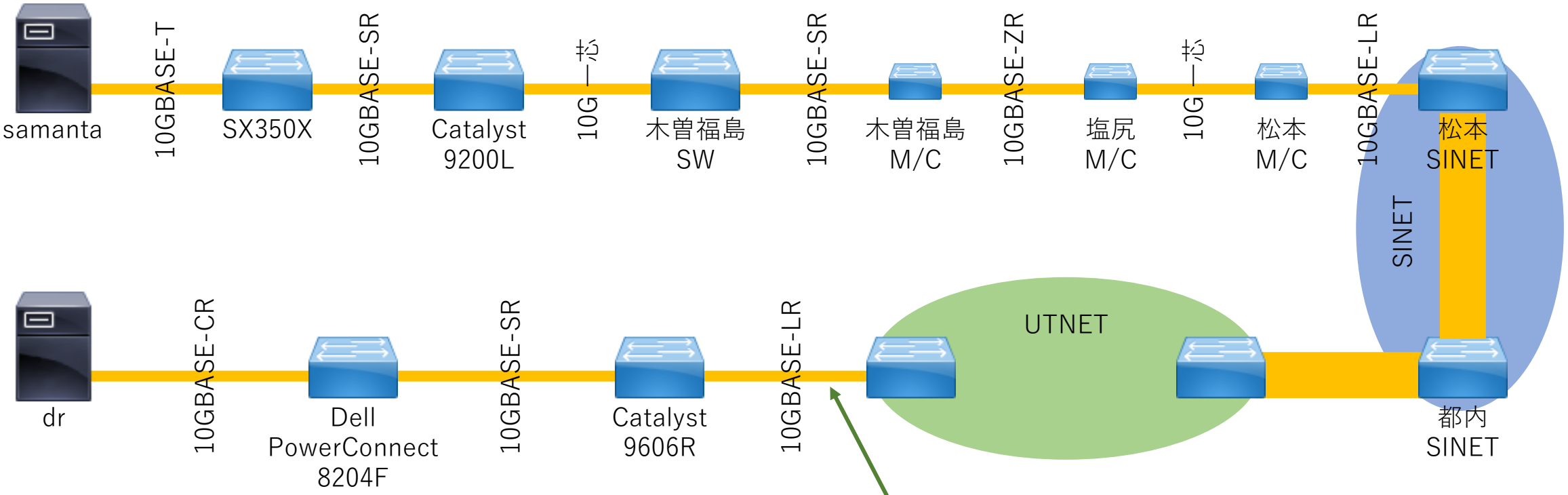
  - ロスしたら輻輳ウィンドウサイズを減らす
  - そこから一定の割合で増やす

- CUBIC



  - ロスしたら輻輳ウィンドウサイズを減らす
  - そこから三次関数で増やす

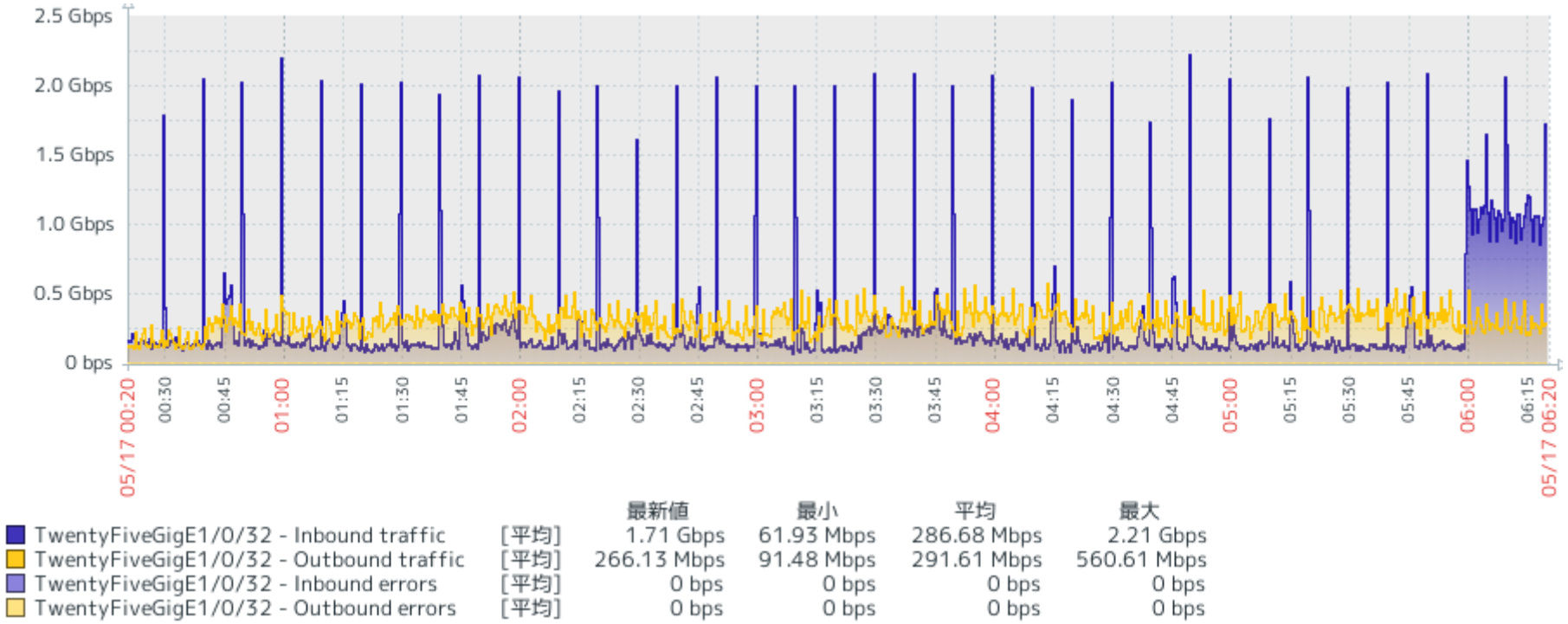
木曽観測所ー東大本郷間のネットワーク



- **木曽～本郷間のRTT: 4.4 ms**
- 工事直後は「10G 一芯」区間の端面に汚れがあり、パケットの5%がロス
  - 現在は解消

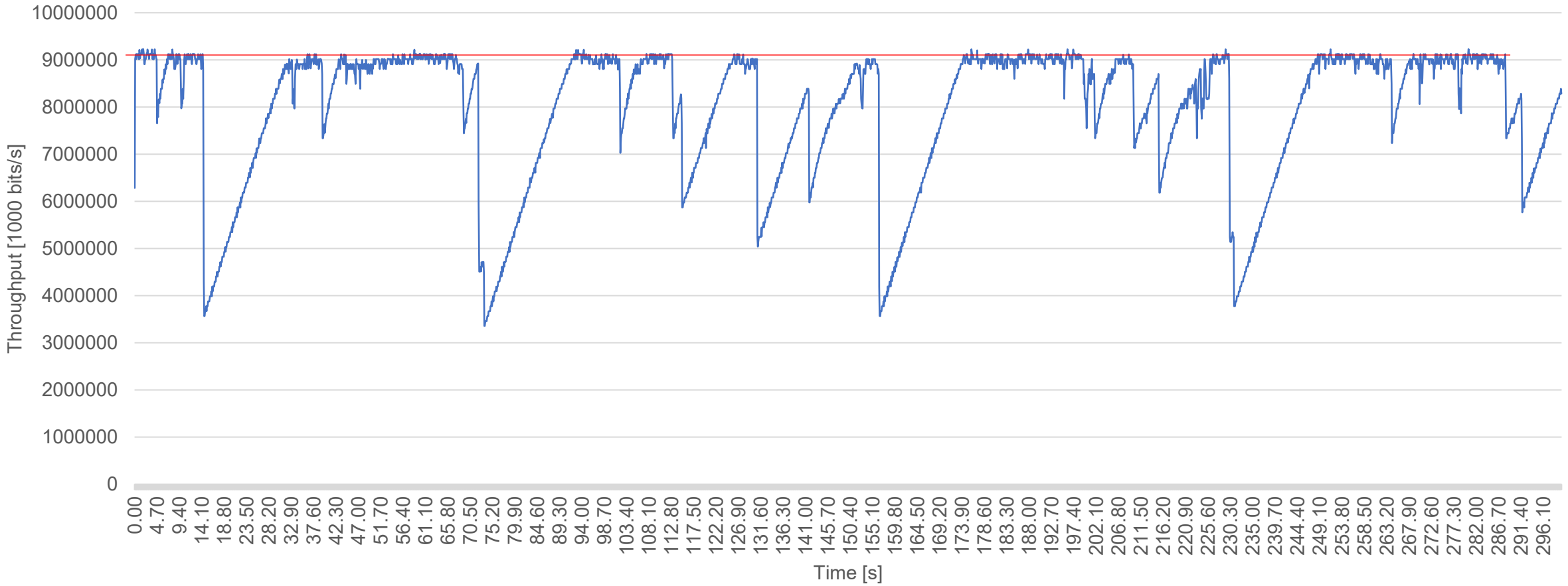
- 一番混むのはUTNET~C9606Rと予想される
- tomoe.mtkはC9606Rから分かれる

深夜の背景トラフィック



- 測定日（5/16）の夜間のトラフィック
  - C9606R~UTNET間の所
  - 青色がUTNET→C9606Rの方向
- 10分に一度、森さんがテストしていた山が見える
- 06:00~ 瀧田さんが転送していたらしい

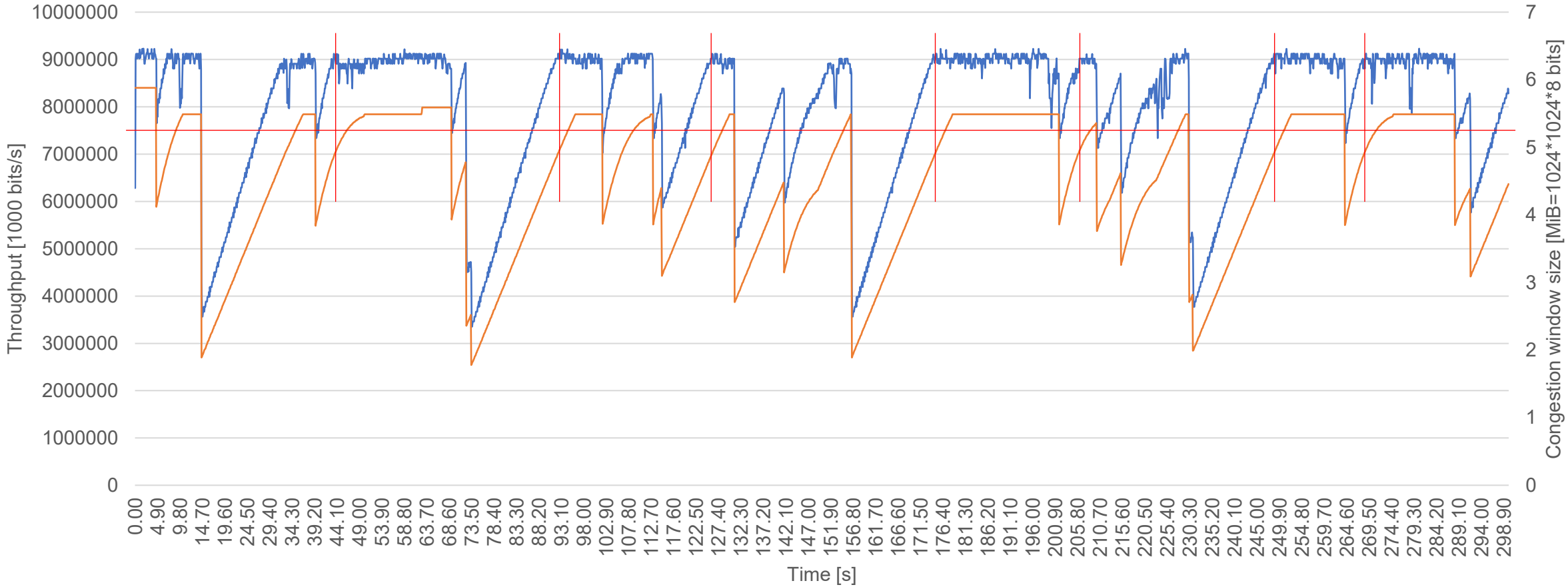
iperfのテスト結果



- 5分間のスループットの推移
- 平均スループット：7.96 Gbps

- 線の下を積分＝転送データ量
  ＝278 GiB
- **一度ロスが起こると痛い**

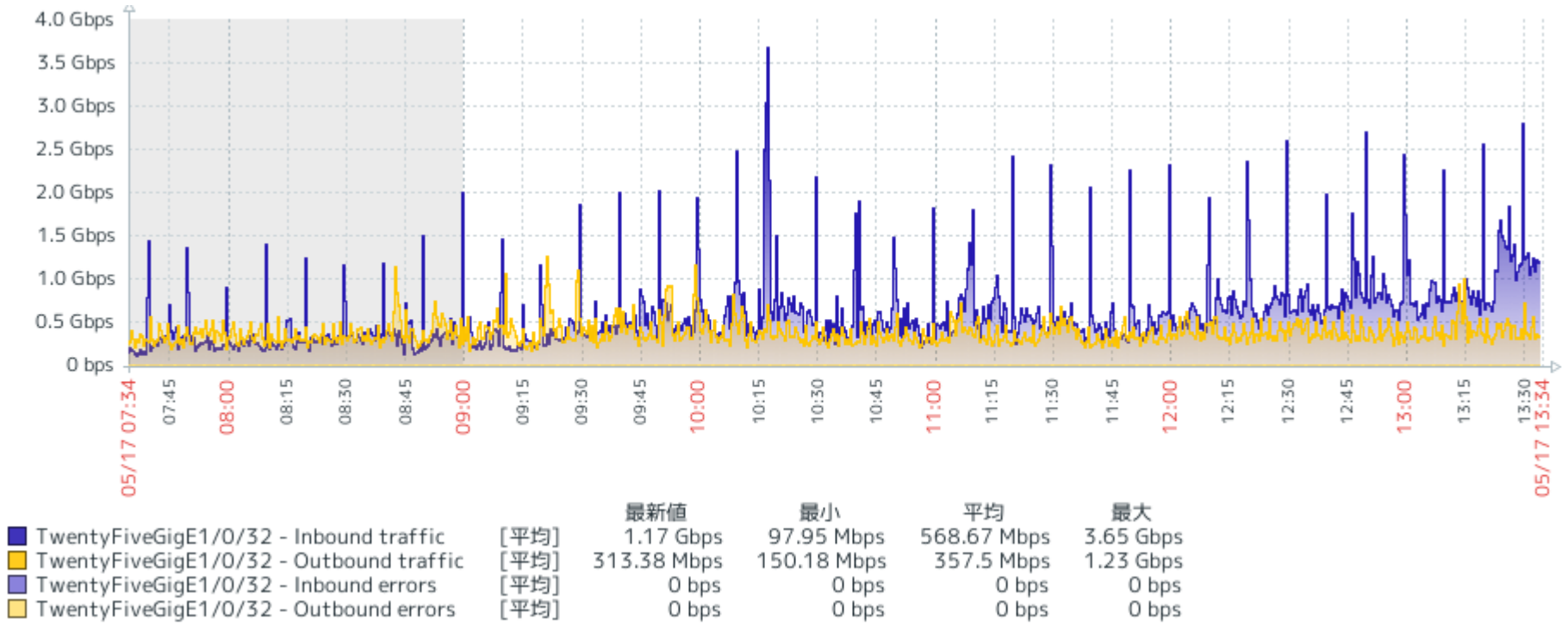
iperfのテスト結果



- 橙：輻輳ウィンドウサイズ
- 動きが概ね一致

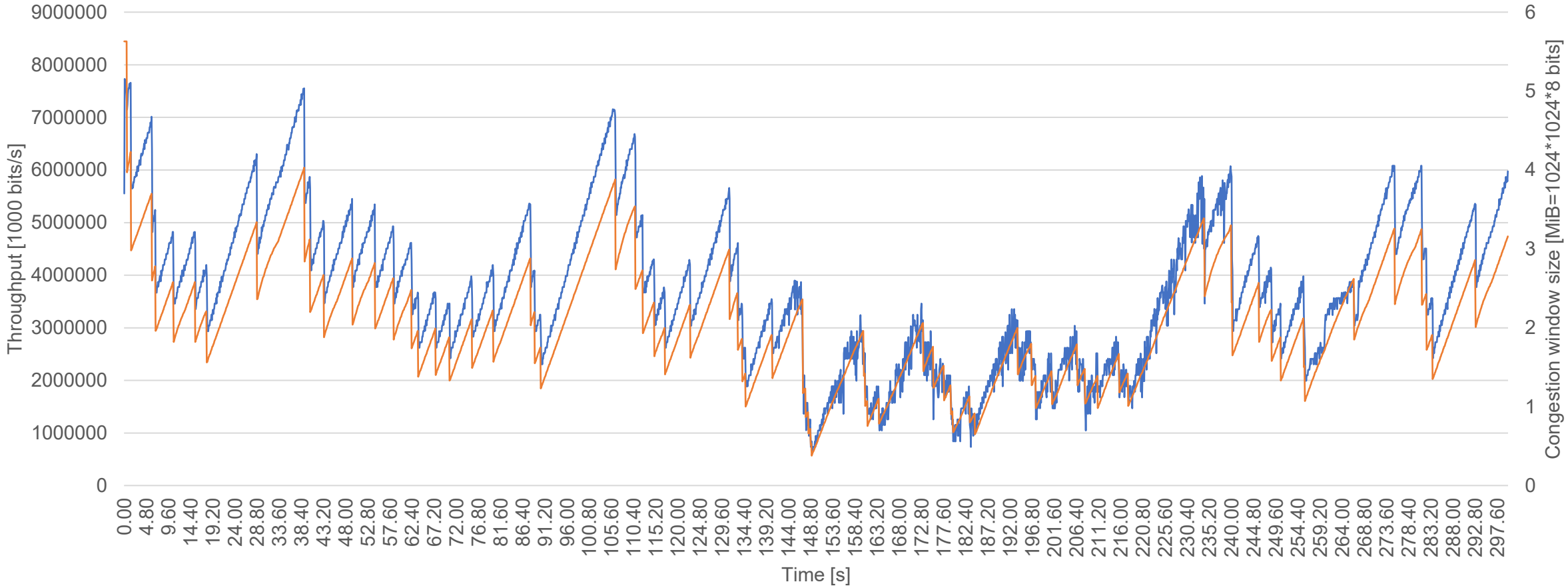
- RTT 4.4 [ms]の時、cwndはおよそ5.25MBあれば10Gを埋められる

[参考] 昼の背景トラフィック



- 全体的に高いし、上下動も多い

[参考] 昼のiperfのテスト結果



- 背景のバーストトラフィックがたくさんあるらしい…

- 平均 3.71 Gbps
- 転送データ量 130 GiB

10Gbpsでのデータ転送

10Gbpsネットワークの理論限界性能

- 10Gbps, 1500B frame
  - Ethernet: IPG 12B, Preamble+SFD 8B, Header 14B (+VLAN 4B), Data 1500B, FCS 4B
  - IP: Header 20B, Data 1480B
  - TCP: Header 20B+Timestamp 12B, Data 1448B

- パケットの先頭から次のパケットの先頭まで：1542B
- その中の正味のデータ量：1448B
- 1448B / 1542B = **93.90%**
- 10Gbps(=10,000,000,000 bits/s)ネットワークでは、**1.10 GiB/s**

伝送効率向上 1: Jumbo frameを使う

- Ethernet frameを大きくする
- 相対的にヘッダ等の占める部分が小さくなる
- 一般的に9000Bに設定することが多い
  - 9000Bまでは対応しているスイッチは多い
- 8948B / 9042B = **98.96%** （←1500Bでは93.90%）
- 10Gbpsネットワークでは **1.15 GiB/s**

伝送効率向上 2: ファイバーはきれいに

- ファイバー端面が汚れていると、確率的にパケットを破壊
- Ethernet frameのFCS部のCRC値が合わなくなる
- 接続されているスイッチのポートのCRC errorカウンタが増える

- 掃除道具の一例

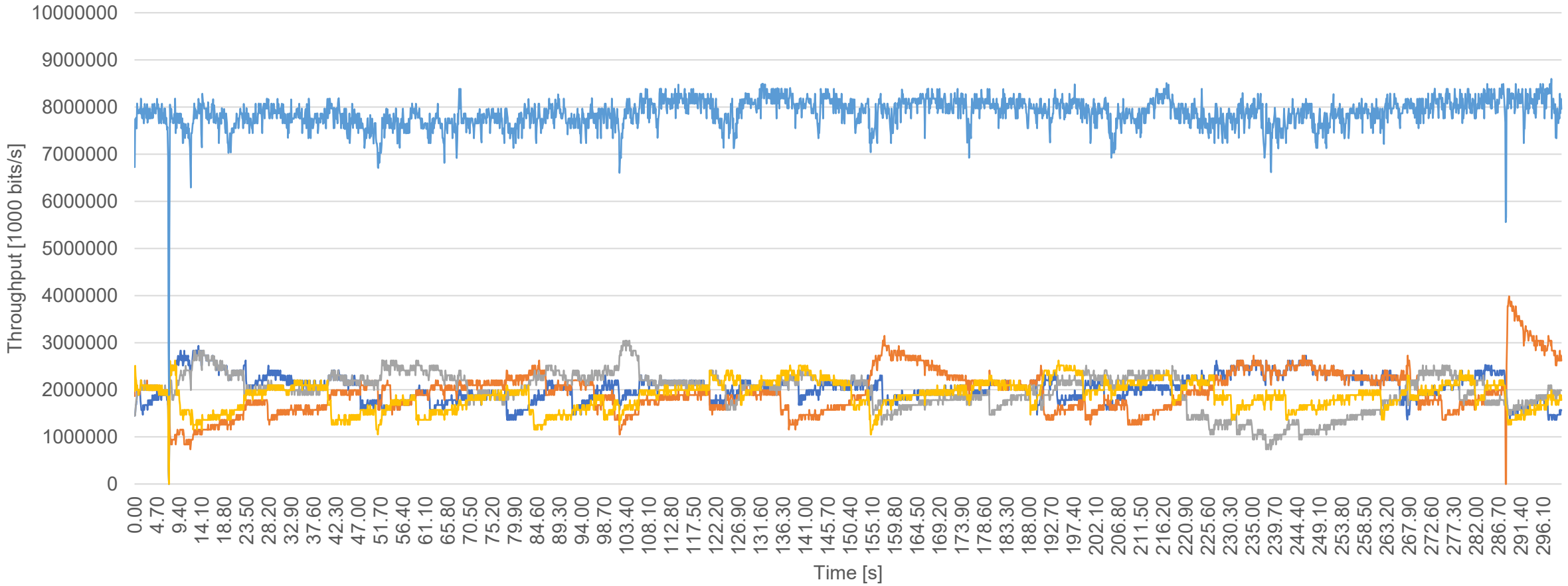
NTT-AT OPTIPOP®シリーズ

精工技研 SFM2.0-250/125

伝送効率向上 3: TCPコネクションを切らない

- 輻輳ウィンドウサイズは、開くのに少し時間がかかる
- 小さなファイル一つ送る度にTCPコネクションを切っていたらもったいない
- **TCPコネクションを切らない転送方法を使用する**
  - tarでまとめる
  - rsyncを使う

伝送効率向上 4: 複数のTCPコネクションを使う



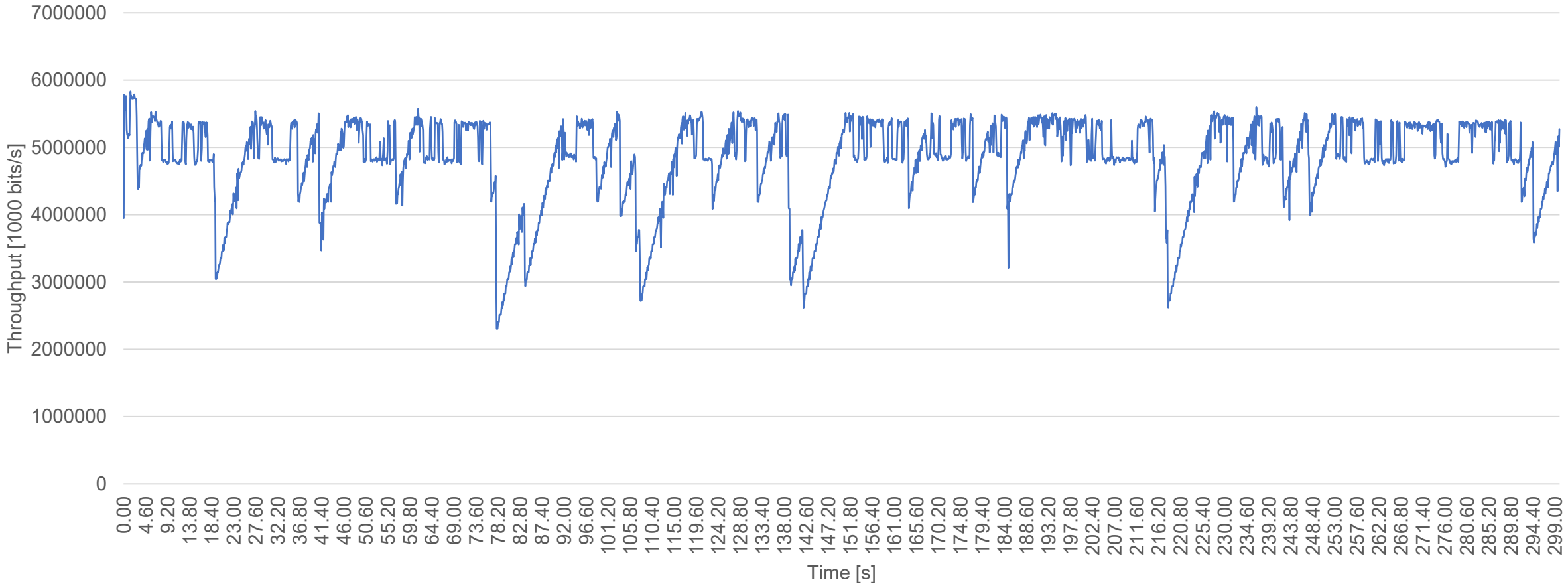
- 昼間です
- 4本のコネクションを使用
  お互い潰し合ったりもするが…

- 平均 7.87 Gbps
- 転送データ量 275 GB

伝送効率向上 5: NICにオフロードする

- TCPパケットの生成等の重い仕事はNICにオフロードさせる
  - オフロードさせることができるNICに限る
- 普通はオフロードするように設定されていると思う
- # ethtool -k enp23s0f1
  ... snip ...
  tcp-segmentation-offload: on
  　　tx-tcp-segmentation: on
  　　tx-tcp-ecn-segmentation: on
  　　tx-tcp-mangleid-segmentation: off
  　　tx-tcp6-segmentation: on
  udp-fragmentation-offload: off [fixed]
  generic-segmentation-offload: on
  generic-receive-offload: on
  ... snip ...

伝送効率向上 5: NICにオフロードする



- 夜間にoffloadingをoffにしてみた
- 遅い理由は追及していません
- 平均 4.84 Gbps
- 転送データ量 169 GB

伝送効率向上 6: 輻輳ウィンドウを大きくする

- OSをインストールしただけでは、ネットワークを埋められるほどの輻輳ウィンドウサイズにならないこともある
- カーネルパラメータを調整
    - 詳細は検索するといっぱい出てくると思います
  - net.core.rmem_default
  - net.core.wmem_default
  - net.core.rmem_max
  - net.core.wmem_max
  - net.ipv4.tcp_rmem
  - net.ipv4.tcp_wmem

伝送効率向上 6: 輻輳ウィンドウを大きくする

- 逆に、輻輳ウィンドウサイズを頭打ちにするには
  - int setsockopt(int sockfd, int level, int optname,
    const void *optval, socklen_t optlen);
    - level = SOL_SOCKET
    - optname = SO_SNDBUF と SO_RCVBUF

- 今のサイズを確認するには
  - # ss -ti state connected
  - State Recv-Q Send-Q Local Address:Port Peer Address:Port
  - ESTAB 0 0 133.11.36.**:34776 133.11.36.**:targus-getdata1
  - cubic wscale:12,12 rto:205 rtt:4.42/1.292 ato:40 mss:1448 rcvmss:536 advmss:1448 cwnd:10 bytes_acked:184 bytes_received:4 segs_out:8 segs_in:7 send 26.2Mbps lastsnd:79825 lastrcv:79810 lastack:79821 pacing_rate 52.4Mbps rcv_space:42340
  - ESTAB 0 8221000 133.11.36.**:34778 133.11.36.**:targus-getdata1
  - cubic wscale:7,7 rto:205 rtt:4.454/0.062 mss:1448 rcvmss:536 advmss:1448 cwnd:1114 ssthresh:1074 bytes_acked:32630017758 segs_out:22535661 segs_in:1018062 send 2897.3Mbps lastrcv:1143504580 pacing_rate 5793.8Mbps unacked:1084 retrans:0/28 rcv_space:29200

伝送効率向上 7: ネットワーク以外も速くする

- ネットワーク以外にもデータ転送のボトルネックとなる要素はある
  - ディスク・ファイルシステム
  - sftp / scp 等使っている場合は暗号化・復号処理

まとめ

- ネットワークを支える技術としてのルーティング、
  使う技術としてTCPの輻輳制御アルゴリズム

- 木曽観測所－東大本郷間でネットワークパフォーマンスを
  計った結果をご紹介

- ネットワークの性能を使い切れていないと思った時に
  使えるかも知れないチェックポイントをいくつかご紹介

ご清聴ありがとうございました