Network Security
Firewall

総合情報学科

セキュリティ情報学コース

Firewallの種類

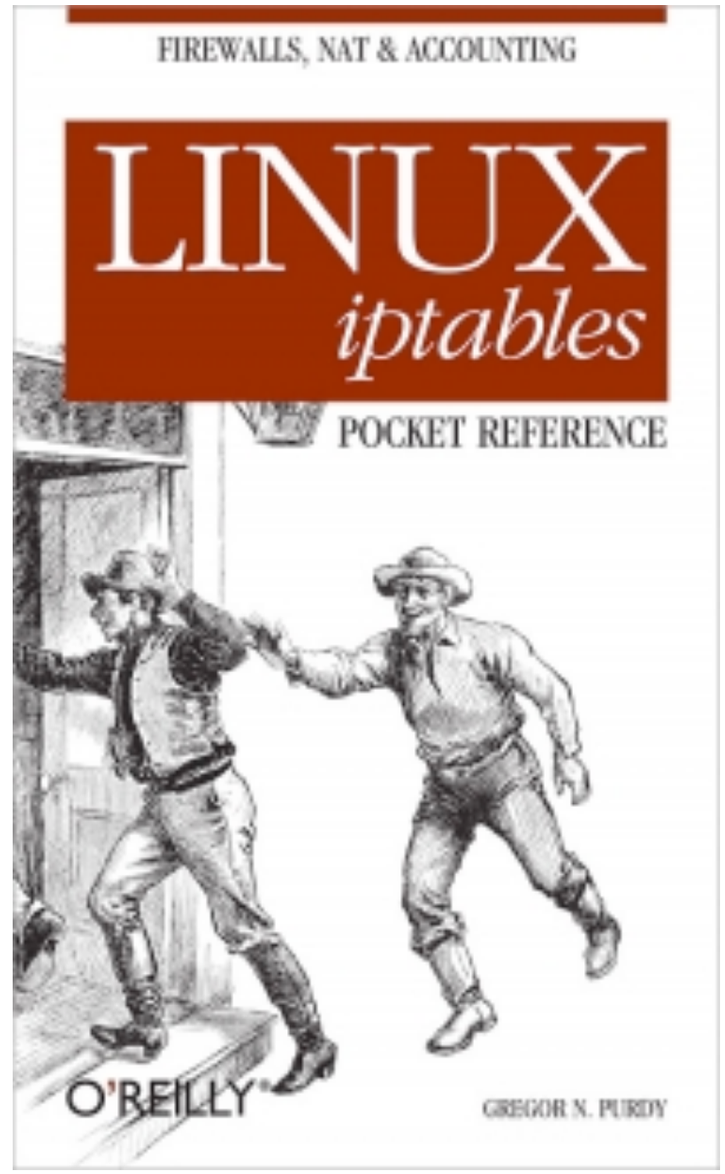
- **Packet filtering** (1st generation fw.)
- **Application level firewall** (2nd generation fw.)
  - Proxy server (URL/domain filtering)
  - Web application firewall (WAF)
- **Dynamic packet filtering** (3rd generation fw.)
  - Stateful packet inspection (SPI)

Note) fw.=firewallの略

Packet filtering

- OSI参照モデルのnetwork(第3層), transport(第4層)の情報を利用して通信制御
  - 第3層：Source & Destination *IP address*
  - 第4層：Source & Destination *Port number*
- Open sourceから商用製品まで多数の実装
  - 著名な実装: iptables(linux), ipfw(freebsd), pf(openbsd) ...
  - 一般的に packet filtering と呼ばれる

iptables



- Packet filtering の一実装
- Linuxで広く使われている
  - Open source
- 多機能 (NAT機能、簡易SPI 他)
  - Linux で Router & Firewall の実装を可能に

Ruleと適用方法

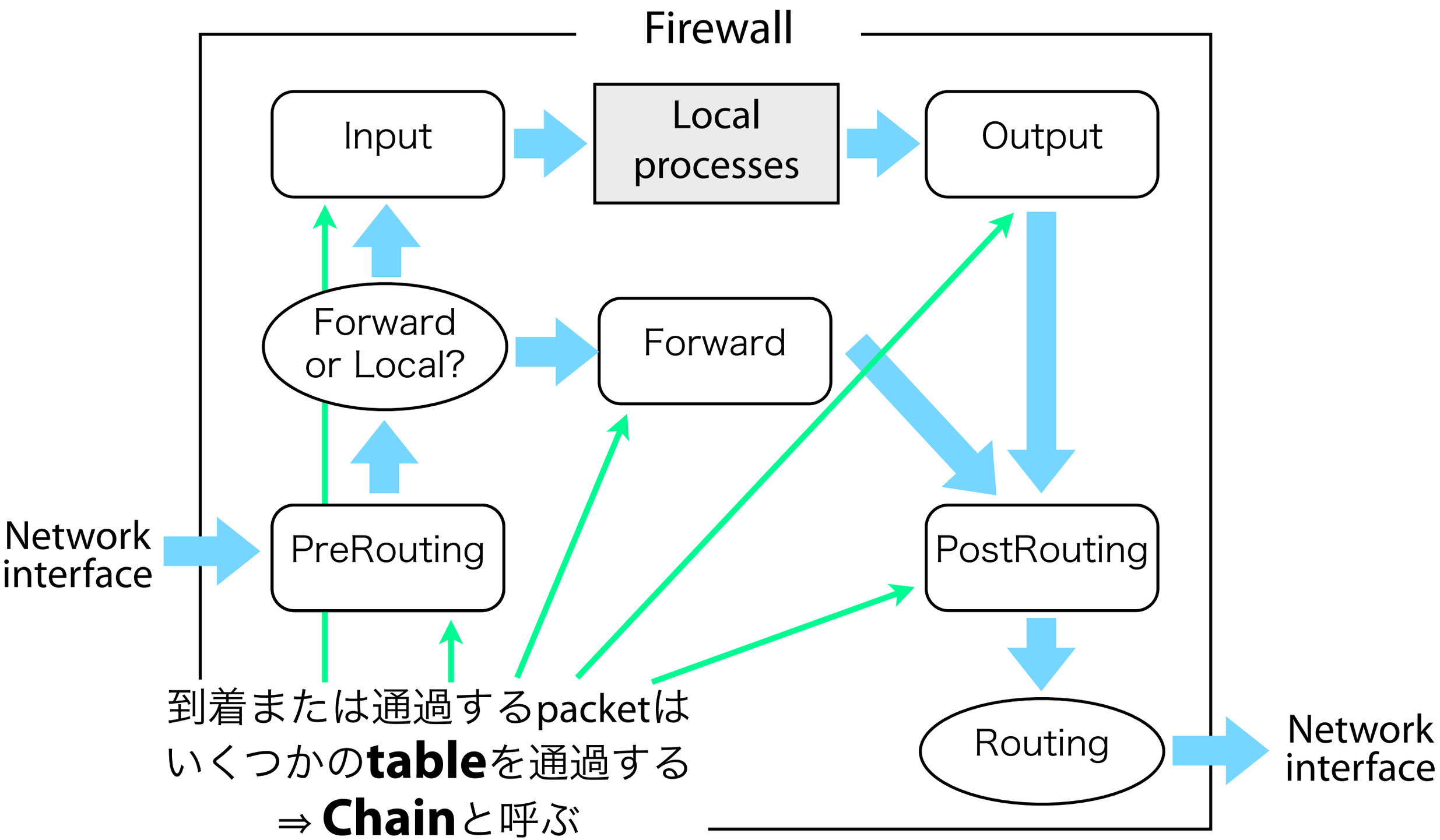

Firewall
Input
Local processes
Output
Forward or Local?
Forward
Network interface
PreRouting
PostRouting
Routing
Network interface
到着または通過するpacketは
いくつかのtableを通過する
⇒ Chainと呼ぶ

table について

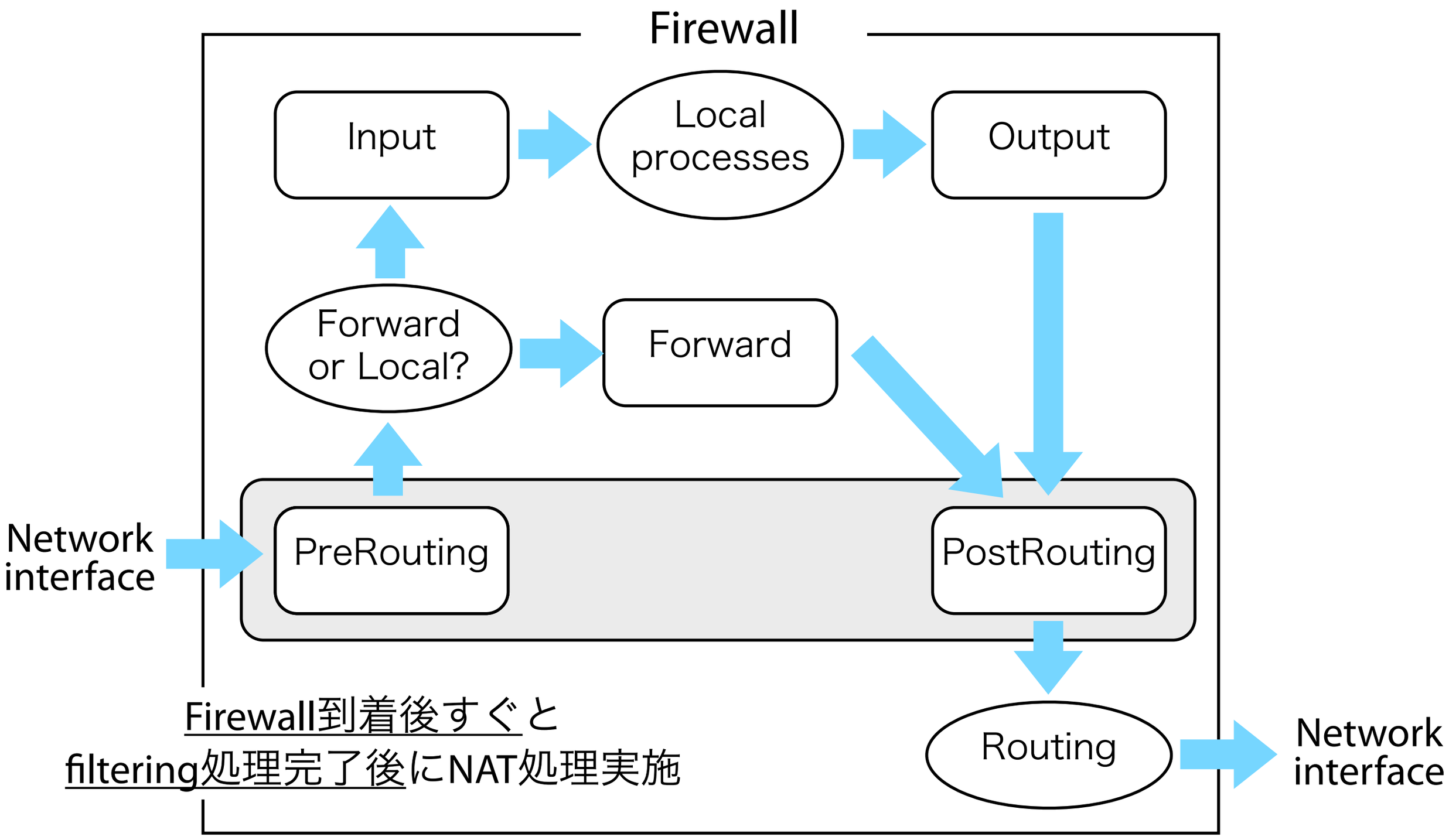
主に2つの役割

- Packet Filtering
  - Network packetの入出力可否を制御
- Network Address translation (NAT)
  - Packetのアドレスを変換 (書き換え)

個々のtableに対して規則を定義

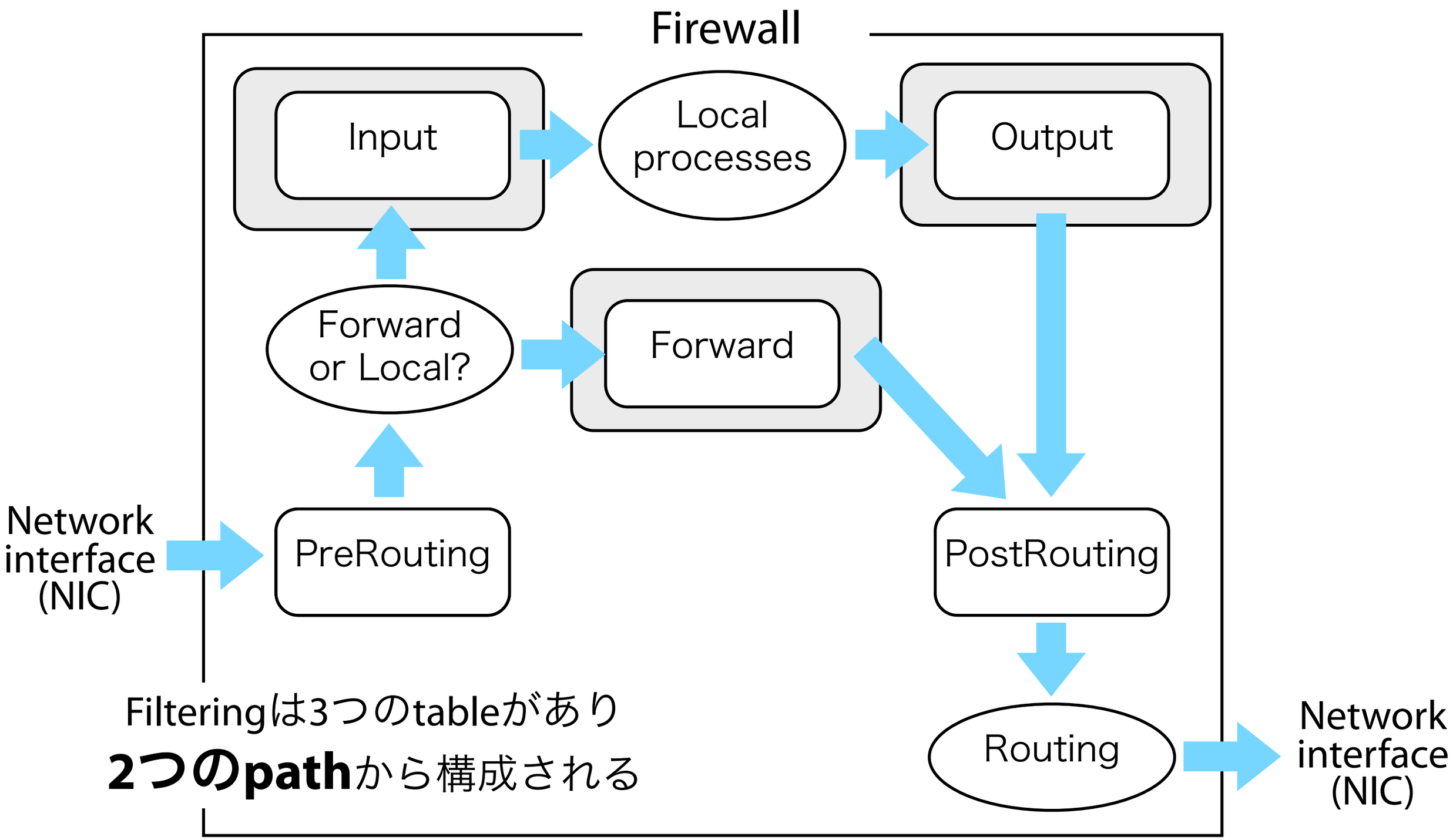
注意：他の役割(機能)にもある

NAT 処理



Firewall到着後すぐと
filtering処理完了後にNAT処理実施

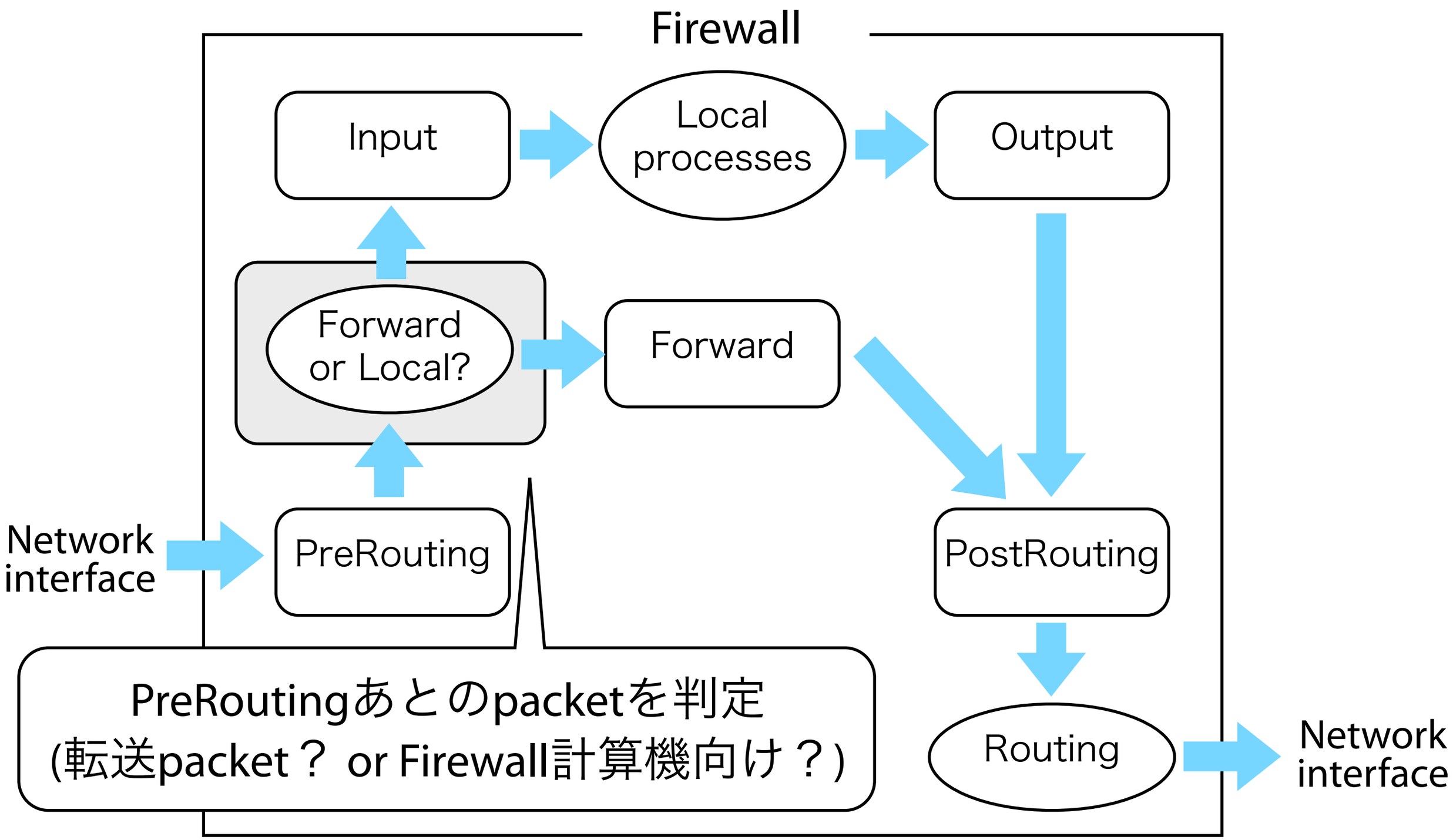
Filtering 処理



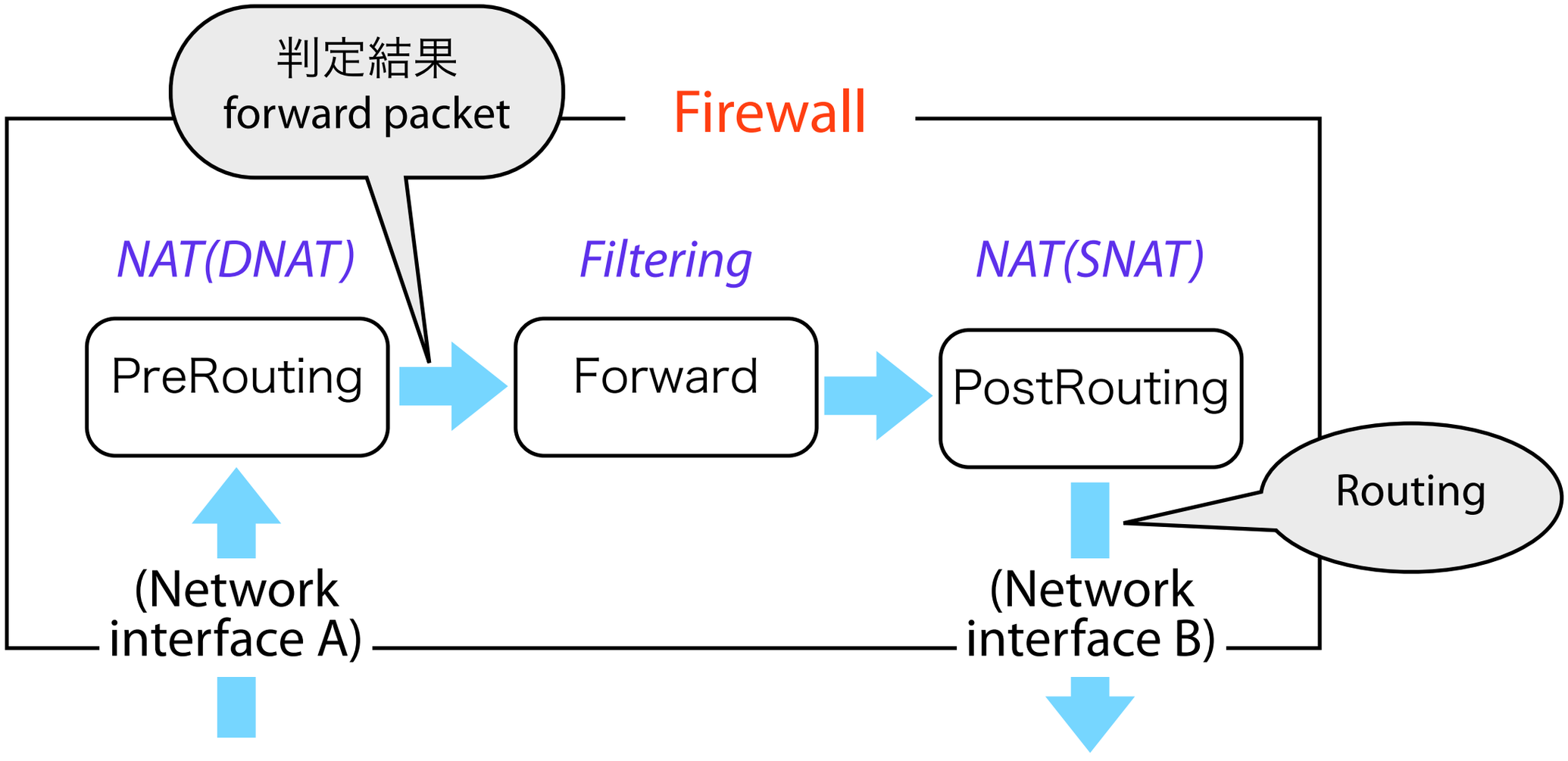
Filteringは3つのtableがあり

2つのpathから構成される

2つのfiltering path



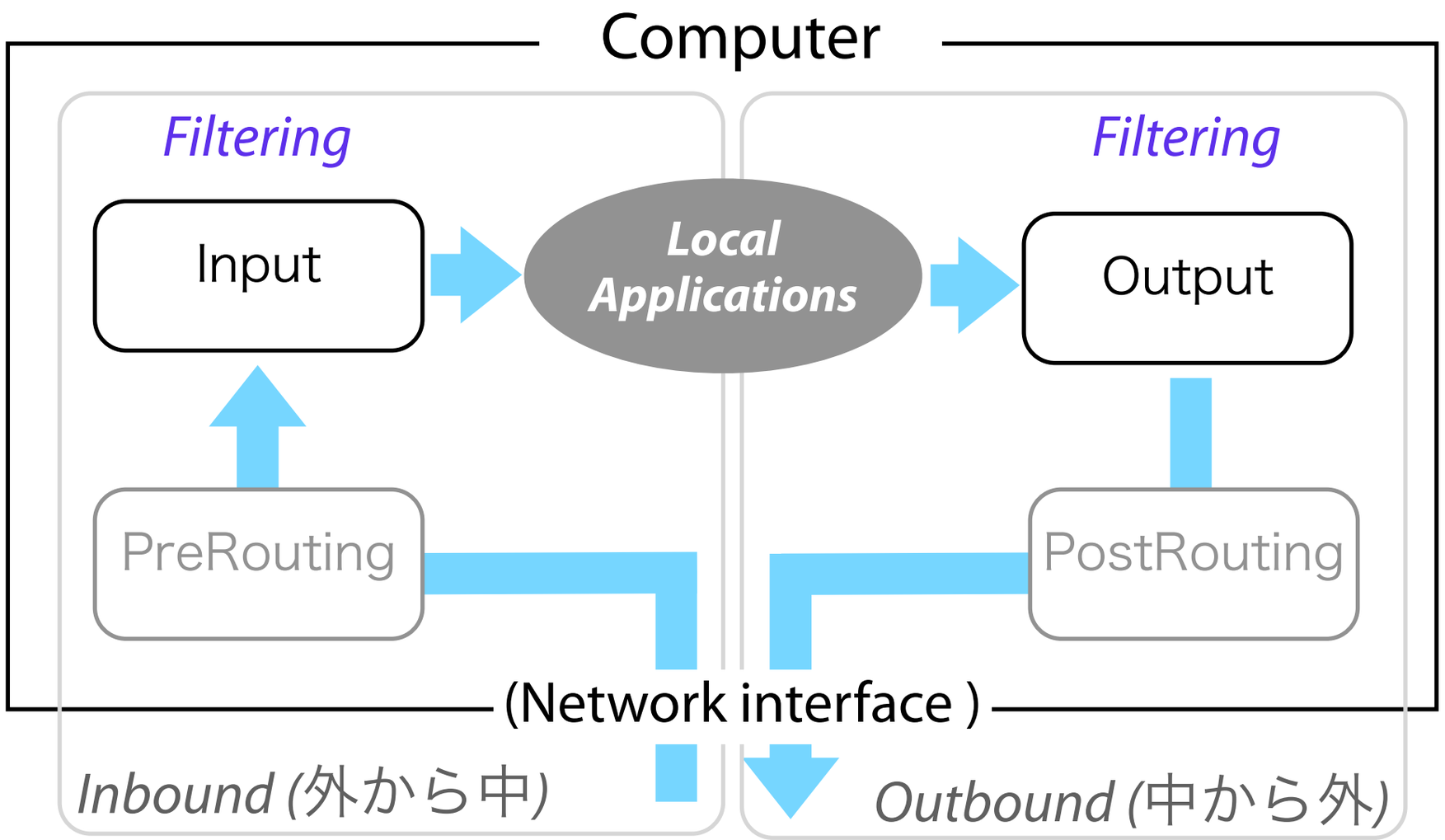
2つの filtering path (1/2)



Packet転送時の処理フロー

いわゆるgateway/routerでの通信制御処理

2つの filtering path (2/2)



Client 計算機におけるpacket filtering

Inbound (外部から中へ)とOutbound (内部から外へ)の双方で制御可能

制御規則の定義

3情報から構成

- どのタイミング (= どのtableへ定義する？)
  ⇒ input, output, forward, prerouting, postrouting
- どのPacketが処理対象か？
- どういう処理を適用？

(処理タイミング, 対象packet, 適用処理)

処理対象packetの定義

基本：Addr. 4 情報 (src IP:srcPort:dst IP:dstPort) で定義

- Source IP addr.：
  - 例) -s 192.168.200.200
- Destination IP addr.：
  - 例) -d 172.21.0.0/24 (network設定)
- Source Port number：
  - 例) --sport 80
- Destination Port number：
  - 例) --dport 20：443 (値域設定 20〜443)

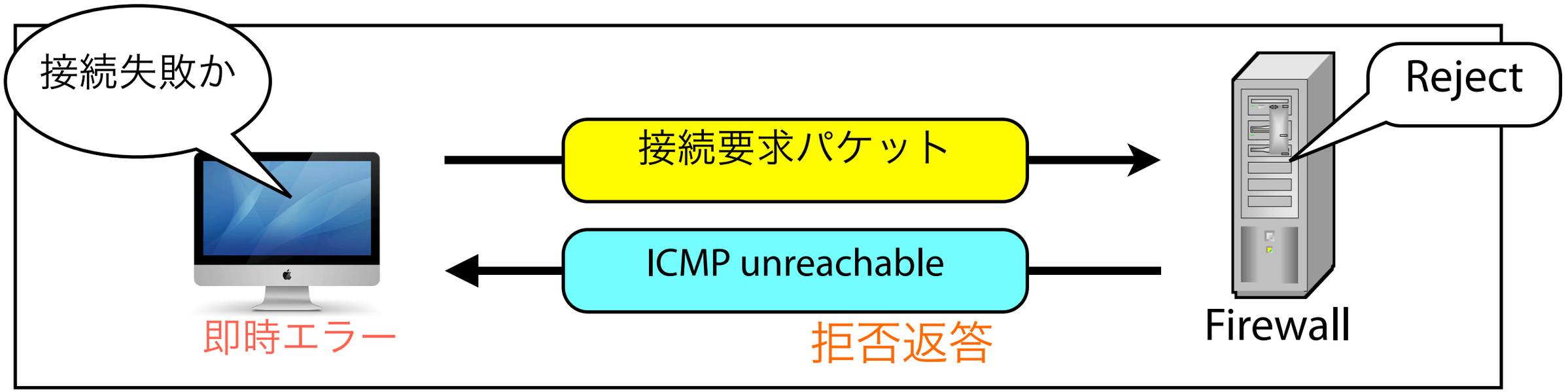
適用処理

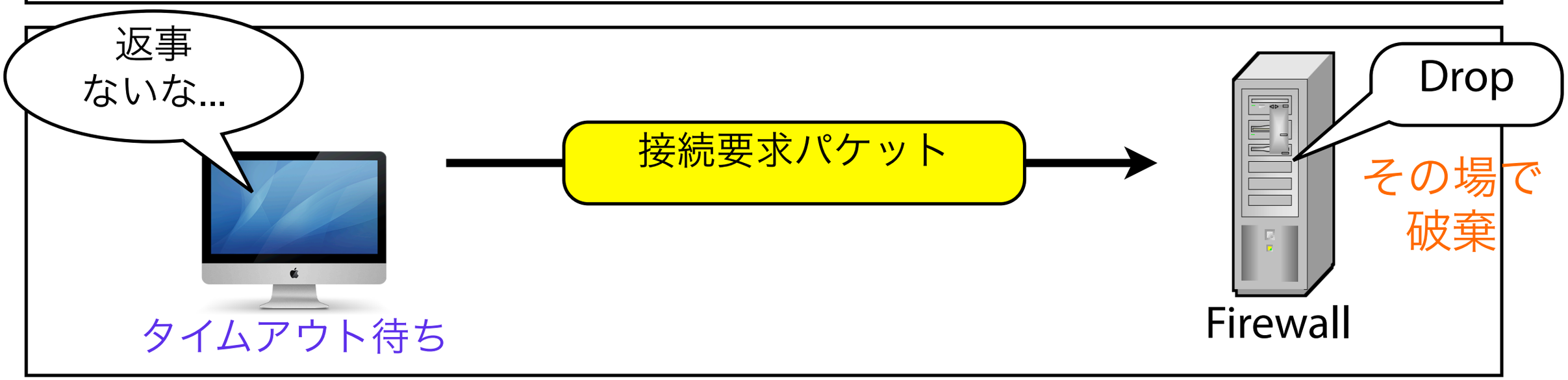
filtering table : 4種類の処理

- ACCEPT　：packetの送受信/転送を許可
- REJECT　：packetの送受信/転送を拒否
- DROP　：packetを破棄
- LOG　：packet情報を記録 (logging)
  対象になったpacketと適用処理を記録

REJECTとDROPの違い

エラー通知 の有無(ICMP unreachable)
(packet送受信/転送拒否は同じ)





REJECTとDROPの違い

- 接続側(Client)に対し、Firewall(サーバ)の存在を隠蔽したい場合にはDROPを選択
- しかし、効果は限定的
  - 同一 dst IP の 80番と81番ポートに接続試行
    - 80番：接続できる (⇒ Web serverが稼働中)
    - 81番：タイムアウト
  - 上記の状態 ⇒ 81番への接続をDropしてもサーバの存在は隠せていない (80番で接続できている ⇒ サーバはある)
  - 実際：Port Scan への遅延効果ぐらい？

Port Scan

- 攻撃予備行為(偵察)の一つ
  - Network接続を試み、以下の情報を収集
    通信機器の有無、稼働サービス、Operating Systemの特定
- 諸刃の剣なツール (使う人の意図次第)
  - System管理者：安全性検証ツール
  - 攻撃者：脆弱性探索ツール

ルール定義のセオリー

1）まずDefault Policyを決定

⇒ “Drop ALL” or “Reject ALL” が多い

2) 次に外部からのInbound接続を定義

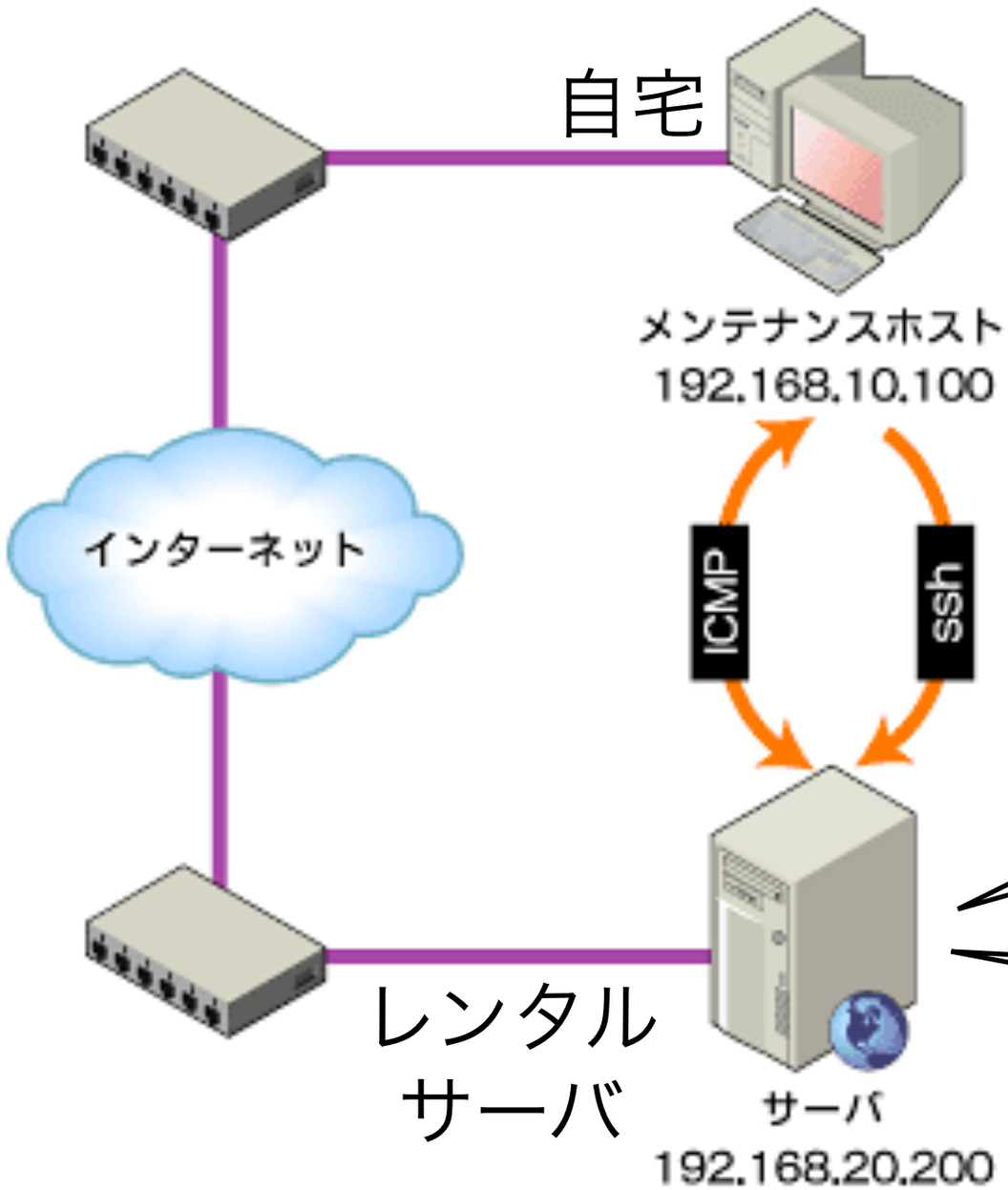
⇒ 必要な通信だけ、Accept

3) 必要ならOutbound接続も定義

4) 動作検証

まずふさぐ、そして穴をあける

iptables設定例



Rental serverにFirewall設定
自宅の計算機からサーバに
接続して維持管理

- 外部からの接続は基本的に拒否
- 自宅⇔サーバ間のpingを許可
- 自宅⇒サーバへのSSHを許可
- サーバ⇒自宅へのSSHは拒否

転送処理はない
⇒ Input, Output table
への規則定義

図引用: http://www.atmarkit.co.jp/ait/articles/0503/18/news122_2.html

iptablesの設定例

```
###
# マクロ定義
###
trustPC＝‘192.168.10.100’# 自宅計算機
myServer＝‘192.168.20.200’# Rental server
any＝‘0.0.0.0/0‘                    # すべてのpacket


###
# 初期化
###
iptables -F
iptables -X
```

iptablesの設定例

```
###
# Default Rules
###
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

###
# loopback (server内での通信用network interface)
###
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -i lo -j ACCEPT
```

この3設定で
Serverに届いた
全パケットは
DROPされる

iptablesの設定例

```
###
# PING trustHost ⇔ myServer
###
iptables -A INPUT -p icmp -s $trustPC -d $myServer -j
ACCEPT
iptables -A OUTPUT -p icmp -s $myServer -d $trustPC -j
ACCEPT
```

PING(活線監視)のための通信許可規則

PING(=ICMP)はLayer 3 protocolのためport numberなし

iptablesの設定例

```
###
# SSH trustHost ⇒ myServer
###
iptables -A INPUT -p tcp -s $trustPC -d $myServer --dport 22
-j ACCEPT
iptables -A OUTPUT -p tcp -s $myServer -d $trustPC --sport 22 -j
ACCEPT
```

dst. Port

src. Port