パケットフィルタにおけるルールの最適配置について

井上 裕司　趙 亮　山本 英雄
宇都宮大学工学部情報工学科
栃木県宇都宮市陽東 7-1-2
e-mail:raezu@degas.is.utunomiya-u.ac.jp

概要

ネットワーク利用の際のコンピュータシステムのセキュリティ対策としてパケットフィルタがよく用いられるが、パケットフィルタを行うことで遅延時間が発生する。本論文ではパケットフィルタのルールの配置を最適化することで遅延時間の短縮を図る。

1. はじめに

近年セキュリティ対策としてパケットフィルタリングがよく用いられるようになっている。ルータの機能を強化して、個々のパケットの単位で通過させたり、禁止したりできるようにしたもののことをパケットフィルタといい、パケットフィルタリングとは、パケットフィルタを用いてネットワークパケットを選択的にフォワードする処理のことである。本研究ではその中の IP パケットフィルタについて考える。

パケットフィルタに関する研究は様々あるが、ほとんどの研究は利便性を向上させるための研究と外部からの攻撃への対応のようなセキュリティ面での研究の 2 種類に分類される。本研究ではパケットフィルタリングによる遅延時間に着目する。パケットフィルタリングでは個々のパケットについて通過、禁止などの判断を行うためにその処理時間が発生する。この処理時間は一つのパケットあたりでは微細なものであるが、非常に多くのパケットを処理するために全体の処理時間は無視できないほど大きくなる場合が考えられる。本研究ではパケットフィルタにおけるルールの配置を最適化することにより、そのような遅延時間の短縮を図ることを目的とする。本研究では Linux の iptables[1]を用いて実験を行うが、他のパケットフィルタでも原理は同じである。

iptables では様々な項目について記述することで、パケット毎の処理を細かく分けることができる。このパケット毎の処理を記述したものをルールと呼ぶ。図 1 にルールの例を示す。

```
rule1; -i eth1 -d 192.168.253.10 -p tcp --dport 1234 -j ACCEPT
rule2; -p tcp --sport 1024:65535 -j DROP
```

図 1 ルール記述例

rule1 は eth1 からきた 192.168.253.10 のポート 1234 宛の tcp パケットを許可するというルールで、rule2 はポート 1024 から 65535 までのポートから送信されたパケットを破棄するというルールである。

またこれらのルールの集合をテーブルと呼ぶ。パケットとルールの照合はテーブルの上のルールから順に適合するルールが見つかるまで 1 つずつ順番に比較していく。これは線形探索という。

このため、適したルールがテーブルの下位にあるほどそのパケットに対する処理時間は増大する。したがって出現頻度が高いパケットに適したルールがテーブルの下位に位置するような場合は多数の入力パケットに対する平均的な遅延時間が大きくなることが明らかである。

そこである期間のパケットフィルタのログからルールの適用頻度を調べ、適用頻度の高いルール程テーブルの上位に位置するように再配置を行う。またその際にルールの位置や順番に制限があるものについての考慮を行う。これはルールの並び順によってパケットフィルタの動作が最初に意図したものと異なるものになることを防ぐために必須である。さらに既存のルールの並び替えだけではなく、より遅延時間を短縮するために新しいルールを作成して追加することを検討する。

本論文の構成を以下に示す。第 2 章ではルールの再配置について、配置基準の考察を行い、またデフォルトのルールから新たなルール作成して追加する方法についての考察を行う。第 3 章ではルールの配置を変更した際の有効性について検証を行い、考察を行う。第 4 章では本論文のまとめと今後の課題について述べる。

2. ルール再配置における

パフォーマンスの改善

2.1 ルール再配置

ルール再配置とはテーブル内のルールをパケットフィルタによる遅延時間がより短くなるように並び替えることである。1 章で説明したように使用頻度の高いルールをよりテーブルの上位に配置することでパケットフィルタによる遅延時間を短縮できると考えられる。そこで予備実験によりテーブル内のルールの配置によって遅延時間が実際にどの程度短縮できるかを調べた。

実験の方法は、パケットを送信する側の PC とパケットを受信する側の PC の 2 台を用意し、その 2 台で LAN を構築する。パケットを送信する側の PC ではパケットフィルタを行わないでパケットを受信する側の PC のみパケットフィルタを行う。この際のパケットフィルタ用のテーブルは図 2 のように設定する。図 2 のテーブルは第 n 番目のルールが宛先ポート番号 n の TCP パケットを許可するというルールになっている。そして宛先ポート番号が 1 のパケットから宛先ポート番号が 1000 のパケットまでをそれぞれ送信し、それぞれの場合について受信側 PC のスループットを計測した。

実験の環境は、パフォーマンスを測定するのには ttcp を用いた。LAN については、クロスケーブルを用いて 2 台の PC を直接つないで 100BASE-TX の LAN を構築した。これはルータやハブの性能によるスループットの変化が生じないようにするである。使用した PC は、まずパケットの送信側 PC には CPU は AthlonXP2200+(実クロックは 1800MHz)でありメモリは 256MB のものを用いた。受信側の PC には CPU は Celeron733MHz でメモリは 128MB のものを使用した。以下の図 3 に実験環境を示す。

```
rule1:-p tcp --dport 1 -j ACCEPT
rule2:-p tcp --dport 2 -j ACCEPT
rule3:-p tcp --dport 3 -j ACCEPT
'
'
'
ruleN:-p tcp --dport N -j ACCEPT
'
'
'
rule998:-p tcp --dport 998 -j ACCEPT
rule999:-p tcp --dport 999 -j ACCEPT
rule1000:-p tcp --dport 1000 -j ACCEPT
```

図 2 予備実験用テーブル

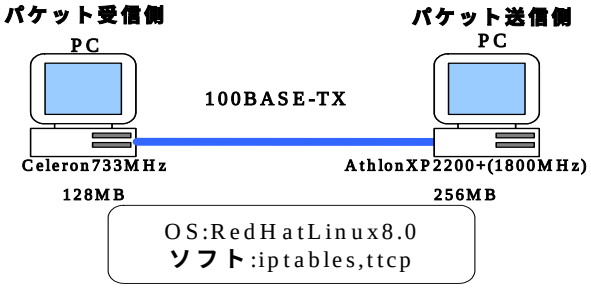

図 3 予備実験の環境

そして実験の結果は図 4 のようになった。大きいポート番号のパケットの場合ほどスループットが低下していることがわかる。

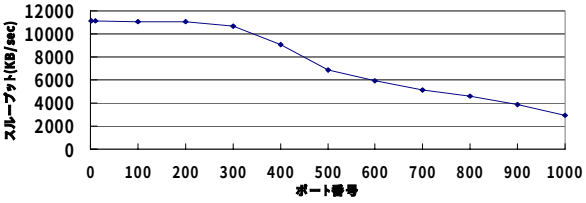

図 4 予備実験の結果

結果よりパケットがテーブルの下位にあるルールによって処理される場合ほど遅延時間の増大によりスループットが低下し、最も低下するポート番号が 1000 の場合にはスループットが 25%にまで低下することが検証できた。また今回パケットフィルタを行った PC の CPU の性能は通常ルータに使用される CPU よりも非常に高速であり、ルータに使用されるような低い性能の CPU においてはスループットの低下がより早い段階で起こることが考えられる。

2.2 ルールの配置基準

ルールの最適配置において最も重要なパラメータはルール毎の適用頻度である。このルール毎の適用頻度はパケットフィルタのログを採取することで調べることが可能である。従って最も単純な配置基準として、ログから得られるルールの適用頻度について降順にテーブルに並べるという配置基準を考えてみる。しかしテーブル内のルールの並び順はパケットフィルタの方針に従うため、単純に適用頻度順に並べ替えることはその方針に反することがある。そこで次のような配置基準でパケットフィルタの方針を保ったまま並び替えることを考えた。

まず、ログファイルを元に適用回数を調べる。ルール(i)の適用回数つまり適用頻度を P_i とする。次にテーブルの上から順に 2 つのルールについて着目する。

まず i=1 とし、P_i と P_{i+1} を比較する。適用頻度の高いルールをテーブルの上位へ配置することが基本であるため $P_i \geq P_{i+1}$ となる場合、i←i+1 とする。もし $P_i < P_{i+1}$ となる場合は交換することが出来れば遅延時間を短縮することが出来るので、パラメータを比較して交換可能か否かを判断する。

まず比較するパラメータはパケットに対する処理(アクション)である。ルール(i)のパケットに関する処理を ACT_i とする。$ACT_i = ACT_{i+1}$ である場合、ルール(i)とルール(i+1)の入れ替えを行ったとしても問題は生じない。これはこの二つのルールが互いに影響を与えないためである。問題が生じる場合というのは、順序の入れ替えによりあるパケットが意図した処理とは別の処理を受けてしまう場合である。一例として次の図 5 のようなテーブルがあったとする。

```
rule1: -p tcp --dport 8080 -j ACCEPT
rule2: -p tcp --dport 1024:65535 -j DROP
rule3: -p udp --dport 1080 -j ACCEPT
rule4: -p udp --dport 6900 -j ACCEPT
rule5: -p udp -j DROP
```

図 5 テーブルの例

この場合 rule1 と rule2 を入れ替えてしまうと rule1 によって許可したかった宛先ポート番号が 8080 であるパケットも rule2 によって破棄されるようになり、最初に意図したパケットフィルタとは異なるものになってしまう。つまり $ACT_i \neq ACT_{i+1}$ の場合安易にルール(i)とルール(i+1)を交換しては危険であると言える。しかしこの場合でも交換可能な場合もあるので他のパラメータも比較する。

そこで我々は $ACT_i \neq ACT_{i+1}$ の場合プロトコルを比較することを考えた。ルール(i)のプロトコルを PRO_i とする。プロトコルが重複していなければ交換しても問題ないと言える。これは例として図 5 の rule2 と rule3 を見ると明らかである。プロトコルが重複している場合は次の比較パラメータとして宛先ポート番号を用いる。ルール(i)の宛先ポート番号を DPO_i とする。rule3 と rule4 を比べてみるとパケットの処理が異なり、プロトコルは同じであるがポート番号が重複していな

いため全く異なるパケットへのルールであることがわかる。つまりポート番号が重複しない場合は交換してもパケットフィルタの方針には支障がないと言える。またこれらプロトコルと宛先ポート番号の二つのパラメータはどちらから比較しても結果が一緒になるのは明らかである。以上のルール交換のアルゴリズムを図 6 に示す。

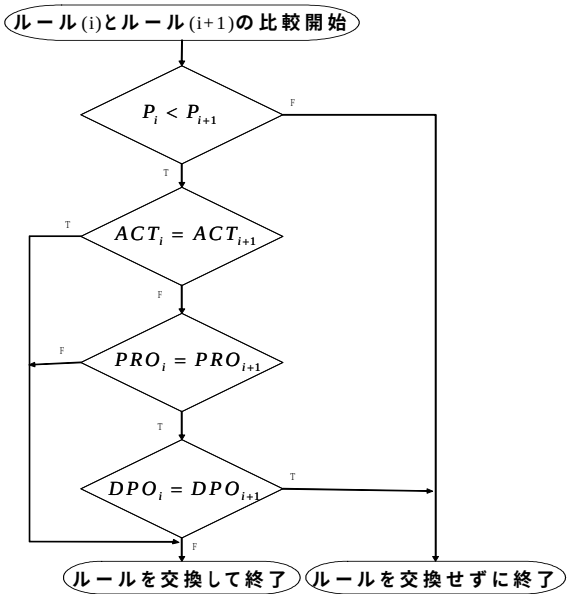

図 6 隣接ルールの交換可否の比較

これらのパラメータを用いた比較を行う際に注意しなければならないのは、これらのパラメータについての記述が無いルールを並び替えようとする時である。

例えば図 5 の rule4 と rule5 に着目してみるとパケットの処理が異なり、プロトコルは同じで、そこで宛先ポート番号を比較しようとすると rule5 にはポート番号に関する記述はない。このような場合は全てのポート番号に一致するとみなしてよい。

以上のように今回は配置基準としてルール(i)の適用頻度 P_i、パケットに対する処理 ACT_i、プロトコル PRO_i、宛先ポート番号 DPO_i の4つのパラメータを用いた。なお、今回は使用しなかった発信 IP アドレス、受信 IP アドレス、インターフェースなど iptables の記述に用いる他のパラメータを判断基準として用いることも可能である。

実際の例として図 7 のようなテーブルのパケットフィルタを行った際に送られてきたパケットが表 1 のようになった場合を考える。

```
rule1: -p tcp --port 8080 -j ACCEPT
rule2: -p udp --port 53 -j ACCEPT
rule3: -p tcp --port 80 -j ACCEPT
rule4: -p tcp --port 3 -j ACCEPT
rule5: -j DROP (デフォルトルール)
```

図 7 元のテーブル

表 1 受信パケット一覧

プロトコル	宛先ポート番号	受信回数	適用されたルール
tcp	3	16	rule4
udp	53	300	rule2
tcp	80	6164	rule3
udp	520	5179	rule5
tcp	1080	1356	rule5
tcp	8080	2465	rule1

この場合のパケットの総処理回数は式(3.1)のようになる。

$$16 \times 4 + 300 \times 2 + 6164 \times 3 + 5179 \times 5 + 1356 \times 5 + 2465 \times 1 = 54296 \quad \textbf{(3.1)}$$

ここで、表 1 をもとに図 7 のテーブルに対しルールの再配置を行うと図 8 のようになる。

```
rule3: -p tcp --port 80 -j ACCEPT
rule1: -p tcp --port 8080 -j ACCEPT
rule2: -p udp --port 53 -j ACCEPT
rule4: -p tcp --port 3 -j ACCEPT
rule5: -j DROP (デフォルトルール)
```

図 8 並べ替えを行ったテーブル

このテーブルで表 1 のパケットを処理する場合の総処理回数は式(3.2)のようになる。

$$16 \times 4 + 300 \times 3 + 6164 \times 1 + 5179 \times 5 + 1356 \times 5 + 2465 \times 2 = 44733 \quad \textbf{(3.2)}$$

この場合処理回数は約 18%削減できている。

2.3 デフォルトルールから新ルール作成

デフォルトルールとはパケットフィルタを設定する際に特に処理を明記されなかったルールを処理するた

めのルールで、常にテーブルの一番下に置かれる。一般的には、セキュリティの面で明示的に許可されていないアクセスはすべて拒否するために、通過させるルールにあてはまらないパケットを全て破棄するというデフォルトルールが設定されることが多い。このデフォルトルールは常にテーブルの最も下に位置するため、デフォルトルールによって処理されるパケットは全てのルールと比較されることになる。このようなパケットが非常に多い場合、そのなかの特に出現頻度の高い条件についてのルールをテーブルに追加することで、デフォルトルールより前の段階での処理が可能になり、全体の処理時間を減らすことができる。

例として図 7 のテーブルと表 1 のパケットの場合を考える。表 1 の宛先ポート番号と受信回数に注目すると宛先ポート番号が 520 であり、デフォルトルールによって破棄されたパケットが非常に多いことがわかる。そこで仮にこのポート番号を破棄するというルール(これを rule6 とする)を追加する。この新しく作ったルールの初期位置はデフォルトルールのすぐ上とすることで、ルール追加を行ってもパケットフィルタの動作に変更は起こらない。そしてルール追加後に図 6 のアルゴリズムを用いたルールの再配置を行うことで最終的に新ルールも適切な位置に移動させることができる。

以上の手順で図 7 のテーブルに新ルールを追加し並び替えを行ったものが図 9 のテーブルである。

rule3: -p tcp --port 80 -j ACCEPT
rule6: -p tcp --port 520 -j DROP(追加したルール)
rule1: -p tcp --port 8080 -j ACCEPT
rule2: -p udp --port 53 -j ACCEPT
rule4: -p tcp --port 3 -j ACCEPT
rule5: -j DROP (**デフォルトルール**)

図 9 新ルールを追加したテーブル

このテーブルを用いた場合の処理回数は式(3.3)のようになる。

$$16\times5+300\times4+6164\times1+5179\times2+1356\times6+2465\times3 \quad = \quad 33333 \qquad \textbf{(3.3)}$$

これは図 8 の場合と比べても約 25.5%の処理回数の削減になっており、図 7 の場合と比較すると約 39%も処理回数が削減できている。

以上のようにデフォルトルールから効果的なルールを作成し追加することで、パケットフィルタによる遅延時間をより短縮することができる。

3. 検証実験

3.1 実験の目的

実際に運用されているパケットフィルタについて、フィルタリングのログを採取し、そのログを元に、新ルールを追加して再配置を行ったテーブルを作成し、2 つのテーブルによるパケットフィルタに対してログファイルと同じパケット群を送信した場合の処理回数を計算し、比較を行うことでどのくらいの改善が見られるのかを検証する。

3.2 実験方法と実験環境

まず 2.3 で示した方法により新ルールをデフォルトルールのすぐ上に追加した。ただし今回は最も回数が多い宛先ポート番号について調べた。また追加するルールは一つとした。

次に 2.2 で示したように隣接するルール間での交換を上から順に行った。ただし追加したばかりの新ルールに関しては、新ルール作成で用いた出現回数を適用頻度とした。

上から順に図 6 のアルゴリズムによる比較を行うと、総ルール数を N とすると N-1 回の比較でテーブルの上から下まで全て比較することになるが、この比較一巡だけでは上方向には最大でも 1 つしかルールは移動できないため、N 巡この比較を行った。こうすることでどのルールも十分に移動させることができるからである。具体的な流れを図 10 に示す。

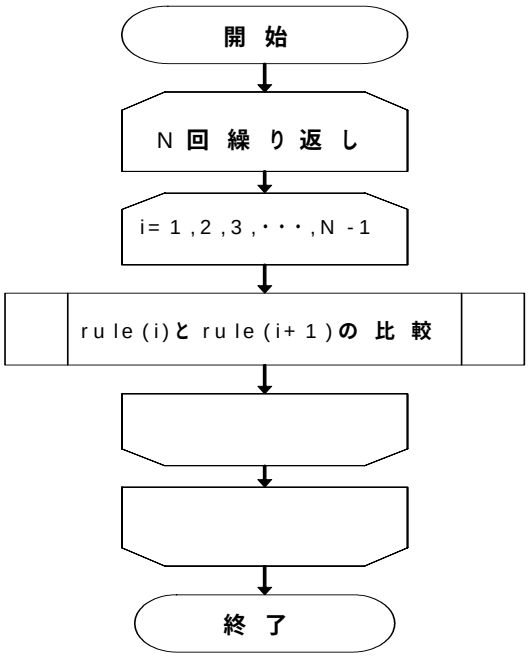

図 10 ルール再配置の流れ

実際に実験に使用したテーブルは図 11 のテーブルである。

Rule 1	Rule 27
Rule 2	Rule 28
Rule 3	Rule 29
Rule 4	Rule 30
Rule 5	Rule 31
Rule 6	Rule 32
Rule 7	Rule 33
Rule 8	Rule 34
Rule 9	Rule 35
Rule 10	Rule 36
Rule 11	Rule 37
Rule 12	Rule 38
Rule 13	Rule 39
Rule 14	Rule 40
Rule 15	Rule 41
Rule 16	Rule 42
Rule 17	Rule 43
Rule 18	Rule 44
Rule 19	Rule 45
Rule 20	Rule 46
Rule 21	Rule 47
Rule 22	Rule 48
Rule 23	Rule 49
Rule 24	Rule 50
Rule 25	Rule 51 (デフォルトルール)
Rule 26	Rule 52 (デフォルトルール)
	Rule 53 (デフォルトルール)

図 11 実験に使用したテーブル

これは本研究室の gateway で実際に使用しているテーブルであるが、ルールの内容はセキュリティ上の都合のため省略してある。rule51、rule52、rule53 がデフォルトルールであり、順に INPUT、OUTPUT、FORWARD のパケットを DROP するものである。またこのテーブルを使った場合のパケットフィルタのログを用意した。その一部を図 12 に示す。またこのログもセキュリティ上の都合により一部を伏せてある。

Rule	From	To	SRC IP	DST IP	PROTO	SPT/TYPE	DPT/CODE	size
14		eth0	192.168.[redacted]	192.168.[redacted]	TCP	1037	22	60
23		eth1	160.12[redacted]	192.5[redacted]	UDP	62420	53	74
23		eth1	160.12[redacted]	192.33[redacted]	UDP	17432	53	62
23		eth1	160.12[redacted]	192.12[redacted]	UDP	8148	53	62
23		eth1	160.12[redacted]	192.35[redacted]	UDP	39131	53	60
4		eth0	192.168.[redacted]	192.168.[redacted]	TCP	1037	22	100
7		eth0	192.168.[redacted]	192.168.[redacted]	TCP	1037	22	100
3	eth1		202.208.[redacted]	160.12[redacted]	TCP	32773	22	100

図 12 ログファイルの一部

3.3 実験結果と考察

まずデフォルトルールによって処理されたパケットの宛先ポート番号別の回数は図 13 のようになった。

ただし 100 回以下のものは省略してある。

port135・・・242回
port137・・・2474回
port138・・・4014回
port161・・・156回
port520・・・23365回
port7001・・・259回
port10000・・・375回

図 13 デフォルトルールで処理されたパケットの回数

またログファイルから読み取った各ルールの適用回数は図 14 のようになった。

新ルールの追加と並び替えを行って最終的にルールが再配置されたテーブルを図 15 に示す。なお追加された新ルールは宛先ポート番号が 520 となっているパケットを破棄するというルールであった。

1番目のルールの出現回数は	46	2番目のルールの出現回数は	46
3番目のルールの出現回数は	30269	4番目のルールの出現回数は	16852
5番目のルールの出現回数は	25397	6番目のルールの出現回数は	1203
7番目のルールの出現回数は	11378	8番目のルールの出現回数は	18944
9番目のルールの出現回数は	0	10番目のルールの出現回数は	0
11番目のルールの出現回数は	8766	12番目のルールの出現回数は	321
13番目のルールの出現回数は	8	14番目のルールの出現回数は	5
15番目のルールの出現回数は	19	16番目のルールの出現回数は	1449
17番目のルールの出現回数は	0	18番目のルールの出現回数は	0
19番目のルールの出現回数は	60	20番目のルールの出現回数は	0
21番目のルールの出現回数は	42	22番目のルールの出現回数は	295
23番目のルールの出現回数は	2809	24番目のルールの出現回数は	69
25番目のルールの出現回数は	2	26番目のルールの出現回数は	0
27番目のルールの出現回数は	2408	28番目のルールの出現回数は	0
29番目のルールの出現回数は	54	30番目のルールの出現回数は	109
31番目のルールの出現回数は	42	32番目のルールの出現回数は	0
33番目のルールの出現回数は	3905	34番目のルールの出現回数は	0
35番目のルールの出現回数は	6	36番目のルールの出現回数は	129
37番目のルールの出現回数は	0	38番目のルールの出現回数は	5304
39番目のルールの出現回数は	3578	40番目のルールの出現回数は	0
41番目のルールの出現回数は	0	42番目のルールの出現回数は	0
43番目のルールの出現回数は	0	44番目のルールの出現回数は	0
45番目のルールの出現回数は	13	46番目のルールの出現回数は	0
47番目のルールの出現回数は	0	48番目のルールの出現回数は	92
49番目のルールの出現回数は	0	50番目のルールの出現回数は	321
51番目のルールの出現回数は	30269	52番目のルールの出現回数は	3
53番目のルールの出現回数は	1235		

図 14 ルール毎の適用回数

Rule	1	Rule	15
Rule	2	Rule	45
Rule	3	Rule	13
Rule	4	Rule	35
Rule	5	Rule	14
Rule	54 追加した新ルール	Rule	25
Rule	8	Rule	9
Rule	7	Rule	10
Rule	11	Rule	17
Rule	38	Rule	18
Rule	33	Rule	20
Rule	39	Rule	26
Rule	23	Rule	28
Rule	27	Rule	32
Rule	16	Rule	34
Rule	6	Rule	37
Rule	12	Rule	40
Rule	50	Rule	41
Rule	22	Rule	42
Rule	36	Rule	43
Rule	30	Rule	44
Rule	48	Rule	46
Rule	24	Rule	47
Rule	19	Rule	49
Rule	29	Rule	51 (デフォルトルール)
Rule	21	Rule	53 (デフォルトルール)
Rule	31	Rule	52 (デフォルトルール)

図 15 新ルール追加とルール再配置後のテーブル

次にパケットと1つのルールとの比較時間を*t*とした場合のテーブルごとの処理時間を計算すると表 2 のようになった。なお処理時間の算出は式(3.1)や式(3.2)や式(3.3)と同様の計算を行った。

表 2 総処理時間

	もとのテーブル	新しいテーブル
総処理時間 (遅延時間)	1,151,583t	528,709t

表 2 を見ると、新しいルールの追加と再配置を行った場合は約 54%処理時間が短縮できている。パケットフィルタの遅延時間を大幅に短縮することができることが分かる。

4. 結論及び今後の課題

本稿は、パケットフィルタにおいてルールの再配置を行い、テーブルを最適化することによって遅延時間の短縮を行うことについて検証し、その有効性を示した。その結果ルールの最適配置は遅延時間の短縮において大きな効果があることを示した。さらに既存のルールの配置を変更するだけでなくデフォルトルールからの新ルールの作成も行ったが、この際作成するルールは宛先ポート番号だけを参照するようなルールであったが、新ルールの初期位置をデフォルトルールの直前とし、追加後にパケットフィルタの動作に変更が加わらない基準でテーブルの再配置を行うことでこの新ルールの位置も最終的に適した位置に変更できた。

またルールの最適配置については、今回はルール毎の処理時間を全て一定の時間*t*であると見なしてルールの再配置を行ったが、実際は比較項目の数によってルール毎にかかる処理時間が違うことが予想される。例えば宛先ポート番号しか比較の項目がないルールと送信元 IP アドレス、宛先 IP アドレス、送信元ポート番号、宛先ポート番号といった比較項目が4つあるルールとではどちらも1つのルールではあるが処理に要する時間は異なるであろうことが容易に想像できる。そこでルール内の比較項目の数による処理時間の相違について検証を行い、結果処理時間が異なるようであれば処理時間も並び替えの際の

基準のひとつにすることでより遅延時間の短縮に効果的なルールの再配置が行えると考えられる[2]ので、今後の課題としたいと思う。

参考文献

[1]netfilter/iptables　URL:http://www.netfilter.org/

[2]Liang Zhao, Yuji Inoue, Hideo Yamamoto,
“Delay Reduction for Liner-Search Based Packet Filters”
ITC-CSCC 2004