空間的解釈に基づくファイアウォール解析システムの研究（継続）

高　橋　直　久　　名古屋工業大学大学院工学研究科教授

1 はじめに

安全で安定したネットワークを実現するためには，ファイアウォールや侵入検知システム（IDS）などのネットワークアクセス検査装置が不可欠である.ネットワーク管理者は,セキュリティの運用指針に従って，これらを正しく設定し，意図した通りに動作させるように維持管理しなければならない．しかし，企業等で実運用されているファイアウォールの設定を調べた結果として，ファイアウォールの設定誤りにより，通過すべき通信を通過させていないなどの状況が数多く発生していることが報告されている[1].ファイアウォールの設定を正しく維持管理する上で，以下のような問題がある.

（1）ネットワークアクセス検査装置は，一般に，管理者が設定した一連のルール（フィルタと呼ぶ）を手続き型プログラムのように解釈実行することにより，パケットの通過と棄却を制御し，許されていない外部からの侵入を防ぐ．このため，経験豊富な管理者であっても，設定内容から動作を把握することは難しい作業となり，設定の不足や冗長を見落としてしまうことがある.

（2）メールを正しく転送できない，あるいは，意図したホームページにアクセスできないなどのネットワークに係わる異常事象が生じた場合には，その原因として多くの可能性が考えられる．このため，ファイアウォールの設定が起因していたとしても，異常の原因を究明し設定を修復するためには，数多くの試行錯誤を伴い多大な時間を要する場合がある.

（3）ネットワークの規模が大きくなると，これらの装置をネットワーク内に多数分散して設置し，それぞれ異なる管理者により設定されるようになる．このような場合には，相互の影響も考慮して注意深く設定する必要がある.

本研究では，ファイアウォール設定の基本となるIPパケットフィルタをとりあげ，その設定を解析するシステムについて探求する．本研究の目的は，上のような問題に対処するため，フィルタ系列を手続き型プログラムのように扱って各フィルタの動作を順に追いかける作業からネットワーク管理者を解放し，自動的に設定異常を検出して設定を正しく維持管理するための方法論を探求することにある.

我々は，昨年度までに空間的関係に基づいてパケットフィルタの解析と診断を行う方法論を提案し，その方法論に基づくファイアウォール設定診断システムの基本機能を実現した[2-5]．今年度は，この研究を継続発展させて，パケットフィルタの空間的関係について詳細に考察し，ファイアウォールの設定誤りを効率的に扱う手法を開発した．具体的には，トポロジーに基づくコンフリクト解析手法を開発し，評価実験により有効性を確認した．また，時間変動を伴うファイアウォールに対しても開発手法を適用可能にする方式について検討を進めた.

2 パケットフィルタリング

2-1 パケットフィルタ

IPパケットフィルタでは，IPパケットの通過と遮断を制御する．このため，ネットワーク管理者は，あらかじめ，IPパケットヘッダの特定フィールド（キーフィールドと呼ぶ）の値を用いて通過すべきか遮断すべきか定めたルール（フィルタと呼ぶ）の系列（フィルタ系列）を記述する．たとえば，図1は，キーフィールドとして送信元IPアドレスと送信先ポート番号の二つを用いて,フィルタf1からf3の三つのルールを定めた例である．この例では，ネットワーク管理者は，フィルタf1により送信元のIPアドレスが123.4.56.70以上123.4.56.90未満のパケットを通過させ，フィルタf2により送信先ポート番号が10～25のパケットを通過させ，フィルタf3により送信元のIPアドレスが123.4.56.50以上123.4.56.60未満で送信先ポート番号が10～25のパケットを通過させ，その他のパケットを遮断するように指定している.

f1	123.4.56.70≦SrcIP＜123.4.56.90		Accept
f2		10≦DstPort≦25	Accept
f3	123.4.56.50≦SrcIP＜123.4.56.60	15≦DstPort≦20	Accept
default	*	*	Deny

図１　IP パケットフィルタの例

2-2 パケットフィルタの操作的解釈

多くのパケットフィルタリングシステムでは，パケットが到着すると，フィルタ系列を上から順番に読み出して，そのパケットがフィルタの条件を満たすか調べる．条件を満たす場合には，そのフィルタが指定したアクション（通過又は遮断）に従って，そのパケットを通過又は廃棄する．条件を満たさない場合には，次のフィルタを読み出して調べるという動作を繰り返す．このような動作を if 文により C プログラム風に記述してフィルタ系列の意味を表すと，図１のフィルタ系列は図２のようになる．図２で，P はパケットを，P.X はパケット P のキーフィールド X の値を表す．

```
if (123.4.56.70≦P.SrcIP and P.SrcIP＜123.4.56.90) then Accept P
else if (10≦P.DstPort and P.DstPort≦25) then Accept P
     else if ((123.4.56.50 ≦P.SrcIP and P.SrcIP＜123.4.56.90)
          and (15≦P.DstPort and P.DstPort≦20) ) then Accept P
else Deny P
```

図２　パケットフィルタの操作的解釈の例（図１の IP パケットフィルタの場合）

2-3 空間的解釈に基づくパケット分類器

フィルタは，キーフィールドの値に従ってアクションを施すべきパケットを指定する．今，キーフィールド数を M としたとき，各キーフィールドの値を座標とする M 次元空間上の点としてパケットを表すような空間（パケット空間と呼ぶ）を考える．このとき，フィルタの条件を，その条件を満たすパケットに対応するパケット空間上の総ての点の集合からなる部分空間（フィルタ空間と呼ぶ）として表すことができる．たとえば，図１のフィルタ系列は，図３のような２次元空間上の領域として表される．

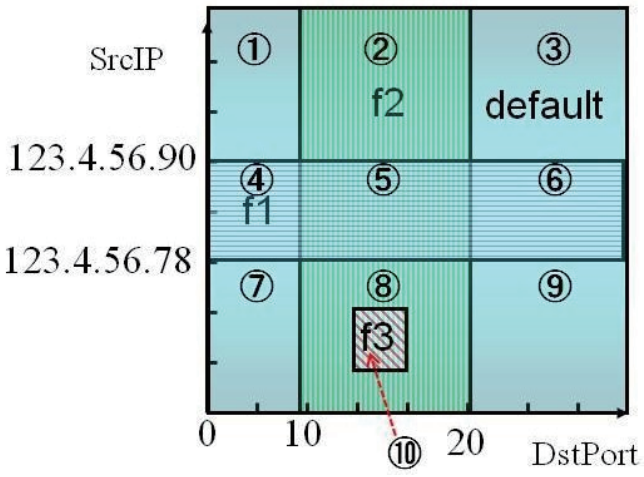

図３　パケットフィルタの空間的表現の例（図１の IP パケットフィルタの場合）

このようにフィルタの意味を空間的に表すと，パケットフィルタリングにおいて，パケットが到着したときに，適用すべきフィルタを選択する問題は，パケット空間上で，パケットがどのフィルタ空間内に位置するかを求める問題，すなわちパケット分類器は計算幾何学の位置決定問題となる．たとえば，図３のようにパケット空間を①から⑩の 10 個の空間に分割したとき，パケットが④の空間に属することが分かった場合には，そのパケットに適用すべきフィルタは f1 と default であることを意味する．通常，フィルタの記述順

序に従い優先度が定められているため，この場合にはフィルタ f1 を適用して，f1 のアクション ACCEPT により，そのパケットを通過させる．

3 先行フィルタにより生じるコンフリクト

図３をみると，f1，f2 のフィルタ空間の一部に重なりがあることがわかる．また，f3 のフィルタ空間が f2 のフィルタ空間に完全に包含されていることがわかる．このようにフィルタ空間に重なりがある場合には，これらのフィルタにより共通して指定されるパケットが存在する．このようなパケットに対しては，先に評価されるフィルタ（先行フィルタと呼ぶ）のアクションが施され，後から評価されるフィルタ（後続フィルタと呼ぶ）のアクションは施されることはない．このような場合に，後続フィルタに対して先行フィルタによるコンフリクトが生じたという．f3 に対して f2 により生じるコンフリクトでは，どのようなパケットに対しても f3 のアクションは決して施されることはない．このようなコンフリクトは設定誤りの結果生じたものである．一方，f2 に対して f1 により生じるコンフリクトでは，f2 が指定するパケットの一部は f2 のアクションが施される．f2 をこのようなパケットだけを指定するように変更すれば，コンフリクトは生じない．しかし，f2 の記述が複雑になる，あるいは，複数のフィルタに分割しなければならないという事態が生じる．ネットワーク管理者が，このような問題を回避するために敢えてコンフリクトを生じるように f2 を記述することもあるので，このようなコンフリクトは設定誤りとはいえない場合もある．しかし，この場合でも，意図的にコンフリクトを生じさせたのか，設定誤りか確認するため，ネットワーク管理者にコンフリクトが生じていることを通知する必要がある．

図２のような分岐文を含むプログラムにおいて，分岐文の条件によりプログラムの実行経路が変化する．条件部の値がどのように変化しても決して実行されない経路（実行不能経路(infeasible path)と呼ぶ）がある場合には，バグの可能性が高いため，実行不能経路の検出法が研究されている．例えば，図２の例の場合には，３つ目の if 文の条件は決して真にならないので，この if 文の then パートに至る実行経路は，実行不能経路とみなされる．このように，フィルタの意味を操作的に解釈する場合には，コンフリクトの検出は，実行不能経路検出問題となる．一方，フィルタの意味を空間的に解釈する場合には，この問題は，前述のようにフィルタ空間の重なりを検出する問題となる．以下では，フィルタ空間の重なり状態を詳細に分類し，この結果を用いてコンフリクトを詳細に分類して，各コンフリクトの検出法を明らかにする．

パケット空間上に２つのフィルタ空間を表すと，フィルタ間の空間的関係を得ることができる．フィルタ f，g のフィルタ空間を *S(f)*，*S(g)*とすると，f，g の空間的関係 *R(f, g)*は，以下のように分類できる．

$$R(f,g)=\begin{cases} Disjoint:(S(f)\cap S(g))=\phi \\ Equivalent:S(f)=S(g) \\ Inclusion1:S(f)\supset S(g) \\ Inclusion2:S(f)\subset S(g) \\ Correlation:otherwize \end{cases} \qquad (1)$$

図１の例のように，フィルタのキーフィールドが送信元 IP アドレス *SrcIP*，送信先ポート番号 *DstPort* の二つのとき，２つのフィルタの空間的関係は図４に例示するように２次元平面上の２つの矩形領域の包含関係として表される．図４の(b)から(e)のように，２つのフィルタ空間がオーバーラップしている場合，コンフリクトがあるといい，フィルタを誤って設定した可能性がある．たとえば，図４の(b)の場合には，フィルタ *f* と *g* のフィルタ空間は同じであり，フィルタ系列で下方に記述されているフィルタのアクションは決して実行されない．

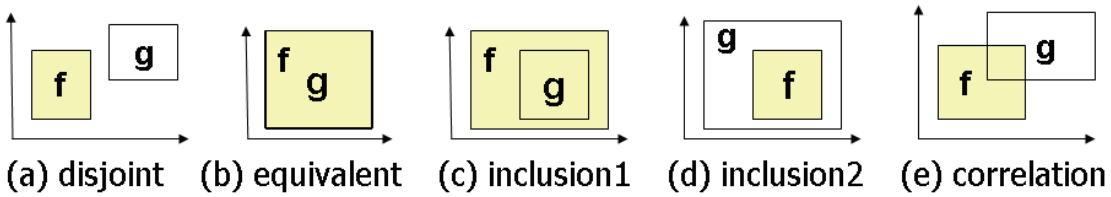

図４　２つのフィルタの空間的関係

（1） 単一ファイアウォールにおけるコンフリクト

(a) リダンダンシーエラー(redundancy error)

前述の f と g が同じファイアウォールに設定されたフィルタであり，*f* が *g* よりも上に記述されていたとする． *R(f,g)*が *Equal* または *Inside* のとき，*g* のアクションは決して実行されない．このとき，*f* と *g* のアクションが同じ場合は，*g* で意図した操作は *g* がなくても *f* で実行されるので，*g* はリダンダンシーエラーであるという．

(b) シャドーイングエラー(shadowing error)

f と *g* のアクションが異なる場合には，*g* で意図した操作が *f* の操作に隠れて決して実行されない．このようなコンフリクトをシャドーイングエラーと呼ぶ．

(c) ジェネラリゼーションウォーニング(generalization warning)

*R(f,g)*が *Inclusion2* で，*f* と *g* のアクションが異なる場合には，フィルタの条件を簡明にするためなど，意図的に *g* のフィルタ空間を大きくしてコンフリクトを発生させた可能性がある．このため，このようなコンフリクトはジェネラリゼーションウォーニングと呼び，エラーとは区別して注意を促すために管理者に提示する．

(d) コリレーションウォーニング(correlation warning)

*R(f,g)*が *correlation* で，*f* と *g* のアクションが異なるとき，上記の場合と同様に注意を促すためにコリレーションウォーニングとして提示する．

（2）複数ファイアウォールにまたがるコンフリクト

図５のように上流と下流にファイアウォールが設置されていて，下流のファイアウォールの設定 (F2) で，あるノードに対するパケットの通過を許可しているときに，上流のファイアウォールの設定 (F1) では当該パケットの通過を禁止している場合に，F1 と F2 に矛盾がある． *f* を F1 に設定されたフィルタ，*g* を F2 に設定されたフィルタとすると，上記（1）と同様に *f* と *g* の空間的関係を調べると，エラー及びウォーニングとなるフィルタを求めることができる．

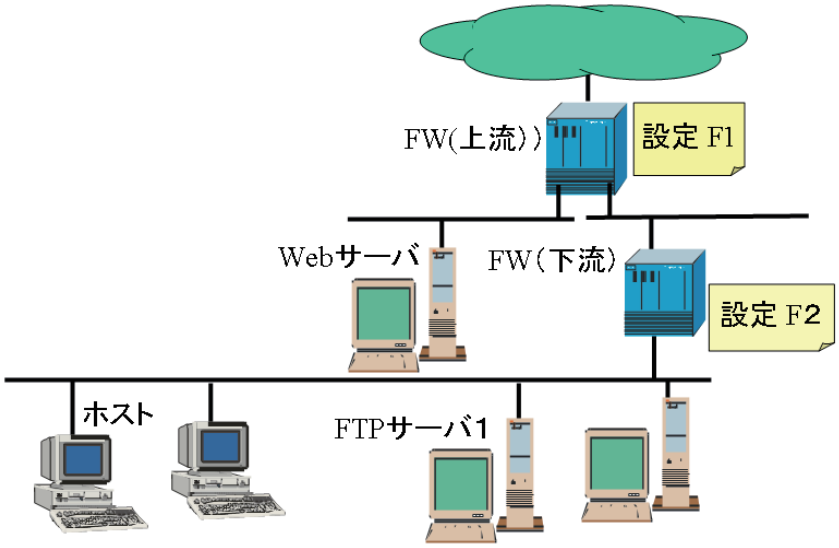

図５ ファイアウォール（FW）が多段に設置されたネットワークの例

4 トポロジーに基づくコンフリクトの検出

4-1 ジオメトリーとトポロジー

パケット分類器では，パケット空間上でのフィルタの空間的関係のうちジオメトリーが重要になる．すなわち，前述のようにパケットが位置する部分空間を覆うフィルタを求めるため，各部分空間の位置が重要である．従来の空間的関係に基づくコンフリクト検出法でも，パケット分類器と同様に，パケット空間上でのフィルタのジオメトリーを求め，その結果に基づいてコンフリクトを検出し分類している[2]．しかし，前章で述べたように，フィルタ間のコンフリクトはフィルタ間の空間的関係のうちトポロジーにより定まる．このため，従来手法では，パケット空間上のジオメトリーからトポロジーを求めている．

図３のパケットフィルタの空間的表現の例を再度考える．この例では，パケット空間を①から⑩の１０個

の部分空間に分割されている．このとき，部分空間④と⑥はジオメトリーとしては異なるが，これらの部分空間のフィルタ間のトポロジー関係は同じである．すなわち，これらの部分空間の内部だけをみると，いずれもフィルタ f1 と default フィルタだけが部分空間全体を覆っている.パケット空間全体におけるフィルタ間のトポロジー関係を求める場合には，トポロジー関係の異なる部分空間を調べれば良い．今，部分空間を覆っているフィルタの集合により，その部分空間のトポロジーを表すと，④と⑥は{default,f1}と表すことができる．すべての部分空間を表すと，{default}（①，③，⑦，⑨），{default,f1}（④，⑥），{defult,f2}（②，⑧），{default,f1,f2}（⑤），({default,f2,f3}（⑩）となる．パケット空間全体におけるフィルタ間のトポロジー関係は，これら部分空間のトポロジー関係から求めることができる．たとえば，f2 と f3 のトポロジー関係は，{defult,f2}，{default,f2,f3}から，inclusion であると判定できる．また，f1 と f2 のトポロジー関係は，{default,f1}，{defult,f2}，{default,f1,f2}から，correlation であると判定できる．

4-2 コンフリクトの検出と分類

フィルタ間のコンフリクトの検出では，フィルタの位置は本質的ではないということに着目して，フィルタ間の空間的な関係のうち，トポロジーのみを用いて設定誤りを求める方式を開発した[6]．この方式では，解析の初期の段階で位置情報を排除して，従来方式では異なるデータとして扱っていた解析結果をまとめることにより，解析に必要なメモリ量と計算時間を軽減することが可能になる．コンフリクトの検出と分類のための手続きを以下に示す．

［コンフリクトの検出と分類の手順］
ステップ１：　フィルタを各次元に分解
ステップ２：　各次元毎に空間分割
ステップ３：　各次元毎にフィルタ間のトポロジーの計算に必要なフィルタ集合を計算
ステップ４：　N 次元空間でフィルタ間のトポロジーを計算
ステップ５：　エラー又はウォーニングのフィルタ対を計算

ステップ１～３では，図６に例示するように，default フィルタを除く全フィルタに対して，各次元毎にフィルタの条件に従って空間を分割し，各部分空間を覆うフィルタ集合を求める．提案方式では，フィルタ集合をビットベクタで表して，ビットベクタを用いた空間演算系を定義し，この演算系により各ステップの計算を効率的に実現している[7]．

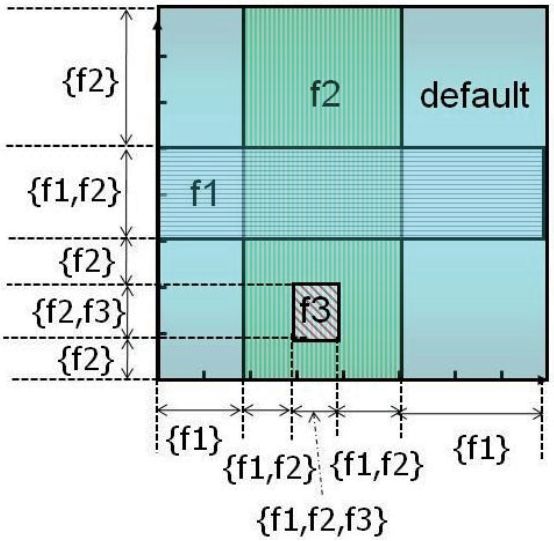

図６　各次元毎の空間分割（図１のフィルタに対する例）

4-3 評価実験

（１） 実験方法

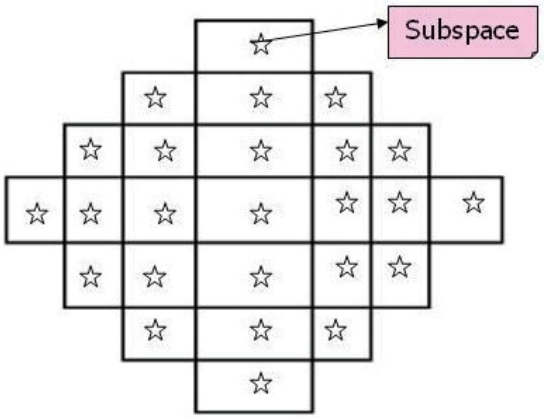

図７　フィルタセットＡでできる部分空間

空間的関係を用いるパケット分類及びフィルタ解析では，部分空間の数が多くなると計算時間とメモリ量が膨大になるという問題がある．このため，図７に示すようにフィルタの数に対して部分空間が大量に発生するフィルタセットＡを用いて以下の２つの実験を行い，提案方式（トポロジーを用いる方式）と従来方式（ジオメトリーを用いる方式）を比較した．

実験１：　フィルタ数を変化させて，サブスペース数と計算時間を測定する．

実験２：　キーフィールドの数（次元）を変化させて，サブスペース数と計算時間を測定する．

（２）　実験結果

図８，９に実験１，２の結果を示す．図８，９より，いずれの実験においても，提案方式が従来方式に比べて，サブスペースの数及び計算時間が大幅に減少することがわかる．

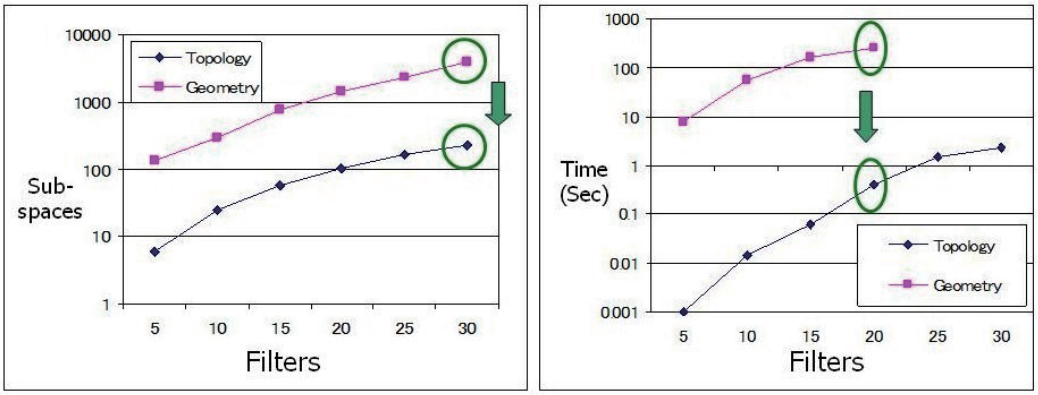

図８　実験１の結果

（フィルタ数を変化させた場合のサブスペース数と計算時間）

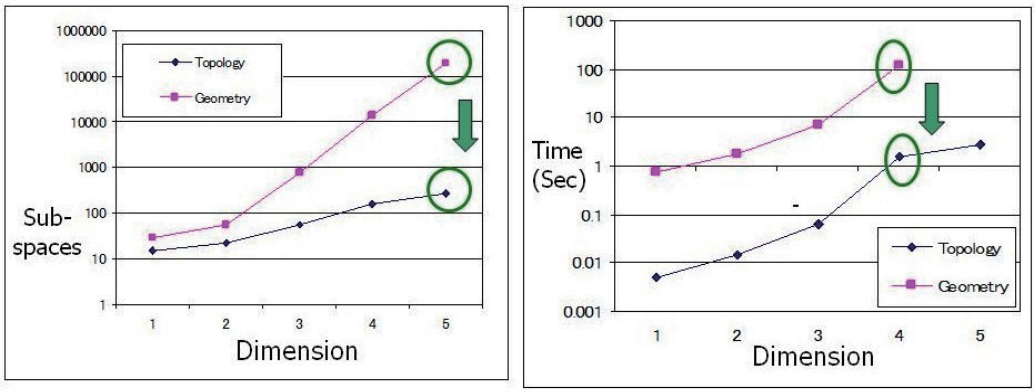

図９　実験２の結果

（キーフィールド数を変化させた場合のサブスペース数と計算時間）

5 時間依存型フィルタの解析

近年，ネットワークに配置されるホストやサーバの増加，ネットワークセキュリティに対する意識の高まりによって，ファイアウォールに設定するルール(以下フィルタ)数は増加している．また，サーバの追加や削除などのネットワーク構成を変更することにより，ファイアウォールの設定を実際のネットワークに合わせて変更する必要がある．このような状況においてファイアウォールにフィルタの追加や削除などを行うには設定を詳細に把握した上で行う必要がある.例えば，「来週の月曜日の 10 時から 12 時まで公開実験をするから，その時間帯だけポートを空けておいて欲しい」といった要求があった場合，管理者は来週の月曜日の 10 時と 12 時にファイアウォールの設定を書き換えなければならない．また，このような要求が頻繁に起こった場合，書き換える回数が膨大になり設定の間違いが起こりやすくなる．

このような問題を解決するために，時間依存型フィルを許すファイアウォールが開発されている．これは，図１０に示すように，各フィルタに当該フィルタが有効となる時間帯（開始時刻と終了時刻）を指定しておき，パケット到着時刻に有効なフィルタのみがパケットフィルタリングに用いられる．このようなフィルタセットを用いる場合には，実際に用いられるフィルタが時刻によって異なるため，設定異常を見つけるのが従来以上に難しくなる．

本研究では，図６の空間分割の方法と同様に，各フィルタの指定時間帯を用いて時間分割を行って，各時間帯（インターバル）に対して有効なフィルタセットを求める方式を開発した[８]．また，各インターバルのフィルタセットに対して，前述の空間的解析によりフィルタ間のコンフリクトを検出・分類するシステムを実現した．図１０のフィルタセットの場合には，図１１のように４つのインターバルに分割し，フィルタセット{f3}，{f0,f2,f3}，{f1,f2,f3} についてコンフリクトの検出と分類を行う．

	SrcIP	DesIP	StartTime	StopTime	Action
f0:	129.6.48.*	123.4.5.*	2008/12/25/10/00	2008/12/28/17/00	Accept
f1:	129.6.48.5	123.4.5.9	2008/12/28/17/00	2008/12/31/17/00	Deny
f2:	129.6.48.9	123.4.5.*	2008/12/25/10/00	2008/12/31/17/00	Accept
f3:	*	*	0000/00/00/00/00	9999/99/99/99/99	Deny

図１０　時間依存型フィルタセットの例

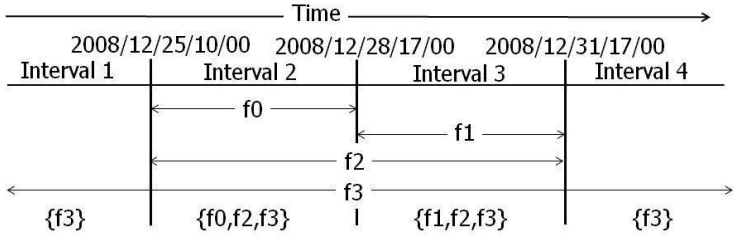

図１１　時間分割の例（図１１の場合）

6 おわりに

本稿では，空間的解釈及び時間的解析により IP パケットフィルタを解析して，設定誤りを検出するシステムに関する研究について述べた．今後，以下の３つの方向に研究を展開する予定である．

（１）設定異常検出機能の適用領域を拡大する

通信状況より動作が変化するステートフルファイアウォールなど，設定内容が時間とともに複雑に変化するような場合に対しても適用できるように空間的解釈と時間的解析に基づく解析方式を拡張する．

（２） システムとしての完成度を高める

GUI の整備などを図り，ツールとしての実用性を高める．さらに，学内ファイアウォール等への適用実験を通して，設定異常の分類を詳細化し，システムへ反映させる．

（３）自己管理型ファイアウォールへ発展させる

検出した設定異常を回復するために自動的に設定を変更する機能について検討を進める．

【参考文献】

[1] A. Wool, “A Quantitative Study of Firewall Configuration Errors. Computer, vol. 37, no. 6, pp. 62-67, Jun., 2004.

[2] Yi Yin, Yoshiaki Katayama and Naohisa Takahashi, ”Detection of Conflicts Caused by a Combination of Filters Based on Spatial Relationships," IPSJ Journal, Vol. 49, No.9, pp. 3121-3135 (2008).

[3] Yi Yin, Kazuaki Hida, Yoshiaki Katayama, Naohisa Takahashi, "Implementation of Filter Reverse Search System based on Spatial Relationships of Filters, " Journal of Convergence Information Technology, Vol. 3, No. 2, pp.6-12 (2008).

[4] Yi Yin, R. S. Bhuvaneswaran, Yoshiaki Katayama and Naohisa Takahashi, "Analysis Methods of Firewall Policies by using Spatial Relationships between Filters," Proc. of IEEE International Conference on Signal Processing Communications and Networking 2007 (ICSCN 2007), Chennai, India, 23-24 Feb. 2007, pp.348-354 (2007).

[5] Yi Yin, R. S. Bhuvaneswaran, Yoshiaki Katayama and Naohisa Takahashi, "Inferring the Impact of Firewall Policy Changes by Analyzing Spatial Relations between Packet Filters," Proc. of 2006 IEEE International Conference on Communication Technology, Guilin, China, 27-30 Nov 2006, IEEE Communications Society (2006).

[6] Subana Thanasegaran, Yi Yin, Yuichiro Tateiwa, Yoshiaki Katayama and Naohisa Takahashi, "Topological Approach to Detect Conflicts in Firewall Policies," International Workshop on Security in Systems and Networks, Proc. of 23rd IEEE International Parallel and Distributed Processing Symposium, SSN-1569173665-paper-3.pdf (2009).

[7] Subana Thanasegaran, Yi Yin, Yuichiro Tateiwa, Yoshiaki Katayama and Naohisa Takahashi, "BISCAL: Bit Vector Based Spatial Calculus for Analyzing the Mis-configurations in Firewall Policies," IEICE Technical Report, IA2008-65, pp.101-106 (2009).

[8] Subana Thanasegaran・立岩佑一郎・片山喜章・高橋直久, “Detection of Conflicts in Time-Dependent Firewall Policies,” 電子情報通信学会 2009 年総合大会講演論文集, A-7-13 (2009).

〈発 表 資 料〉

題 名	掲載誌・学会名等	発表年月
Implementation of Filter Reverse Search System based on Spatial Relationships of Filters	Journal of Convergence Information Technology	2008 年 6 月
Detection of Conflicts Caused by a Combination of Filters based on Spatial Relationships	情報処理学会論文誌	2008 年 9 月
フィルタの空間的関係解析のためのビットベクタ型空間計算法	平成 20 年度電気関係学会東海支部連合大会講演論文集	2008 年 9 月
BISCAL: Bit-Vector-Based Spatial Calculus for Analyzing the Mis-configurations in Firewall Policies	電子情報通信学会インターネットアーキテクチャ研究会技術報告	2009 年 1 月
セキュリティポリシーとファイアウォールポリシーの不整合検査手法について	電子情報通信学会 2009 年総合大会講演論文集	2009 年 3 月
Detection of Conflicts in Time-Dependent Firewall Policies	電子情報通信学会 2009 年総合大会講演論文集	2009 年 3 月
Topological Approach to Detect Conflicts in Firewall Policies	Proc. of 23rd IEEE International Parallel and Distributed Processing Symposium	2009 年 5 月