仮想型耐侵入システムの確率モデルによる性能評価

鄭　俊俊，岡村　寛之，土肥　正

本稿では，仮想型耐侵入システムに対する確率モデルの構築を行う．耐侵入システムは不正アクセスが発生したとしてもサービスを継続し，侵入検知システムの機能を補完するものである．本稿では耐侵入システムの概要を示し，ハイパーバイザ型の仮想化プラットフォーム上で実装される耐侵入システムに対して確率モデルによる信頼性評価を行う．特に，システムへのリクエストがマルコフ型到着過程に従って発生する一般的な確率モデルを考え，セキュリティ障害が発生するまでの時間分布を待ち行列理論を応用して解析的に導出する．

キーワード：耐侵入システム，確率モデル，信頼性評価，待ち行列解析，マルコフ型到着過程

1. はじめに

多くの情報システムにおいて，システム外部からの悪意ある攻撃が発生した場合の対策はシステムセキュリティの観点から重要である．悪意ある攻撃からシステムを防御する手段として侵入検知技術 (Intrusion Detection Technology) がよく知られている．侵入検知技術はシステムのリソースをモニタリングし，既知の不正アクセスパターンとのマッチングを行ったり，統計的に異常な状態を検出することで，システムの異変を特定することができる．侵入検知技術は悪意ある攻撃からシステムを防御するのに有効な手段であるが，未知の不正アクセスパターンや誤検出の問題があるため完璧な防御手段とはなりえない．換言すると，どれだけ悪意ある攻撃への対策を講じたとしても不正アクセスが行われるリスクが存在する．そのため，耐侵入技術 (Intrusion Tolerant Technology) が侵入検知技術などの不正アクセスを防御する手段を補うものとして提案されている．

耐侵入技術とは，不正アクセスが発生したとしてもシステムが提供するサービスを正しく継続させるための技術であり，具体的な技術としてはサーバやデータなどの冗長化や回復動作がある．冗長化は複数のサーバやデータを用いることで，そのうちのいくつかが不正アクセスをされたとしても，不正アクセスをマスクして正しい結果を導くことができる．一方，回復動作は不正アクセス後のシステム復旧を迅速に行うことでシステムの可用性を高める手法である．本稿では冗長化による耐侵入技術に着目する．

冗長化による耐侵入技術は多数決による検証が基本となる．つまり，複数のサーバに同じ処理をさせ多数決をとることで，サーバが不正アクセスされたとしても正しい結果を得ることができる．具体的に，耐侵入技術での多数決にはビザンチン障害に対する耐障害アルゴリズムが利用される．ビザンチン障害とは「うそ」の出力を含めて最悪の振る舞いをする障害であり，不正アクセスが行われたサーバがこれに該当する．ビザンチン障害が発生する環境のもとで正常に動作しているすべてのサーバが正しい結果を得る問題（これを「合意」を形成すると言う）はビザンチン将軍問題と呼ばれ，分散アルゴリズムの分野では古くからこの問題に対するアルゴリズムが議論されてきた．一般的には，全サーバ数 n のうち f 個のサーバにビザンチン障害があるとき，$n \geq 3f+1$ が満たされるのであれば合意形成が可能である．実際，Wang ら [1]，Merideth ら [2]，Zhao [3] は，このアルゴリズムを利用した耐侵入システムを議論している．しかしながら，たとえば 1 台のサーバに対する不正アクセスによるセキュリティ障害をマスクするために 4 台以上のサーバを準備することは非常にコストがかかる．一方，確実に信頼できるサーバが 1 台存在した場合，ビザンチン将軍問題で合意形成されるための条件は $n \geq 2f+1$ となる．このことを利用して，Junior ら [4] は仮想化技術で信頼できる経路を確保することで，より低コストの耐侵入システムを実現している．さらに，Lau ら [5] は仮

てい　しゅんしゅん
立命館大学情報理工学部
〒 525–8577 滋賀県草津市野路東 1–1–1
jzheng@asl.cs.ritsumei.ac.jp
おかむら　ひろゆき，どひ　ただし
広島大学大学院工学研究科
〒 739–8527 広島県東広島市鏡山 1–4–1
okamu@hiroshima-u.ac.jp
dohi@hiroshima-u.ac.jp

想化サーバを段階的に増やすことで，よりオーバーヘッドの少ない耐侵入システムを提案している．

より低コストでオーバーヘッドの少ない耐侵入システムが提案されるなかで，定量的に耐侵入システムの信頼性や性能を評価する試みが行われている．Madan ら [6] や Uemura and Dohi [7] は SITAR (Scalable Intrusion Tolerance Architecture) と呼ばれる耐侵入システムに対してマルコフ・セミマルコフモデルを構築し，信頼性を定量的に評価している．また，Zheng ら [8, 9] は Lau ら [5] による仮想型耐侵入システムに対する性能評価を行っている．特に文献 [9] では，システムへのリクエストがポアソン過程に従って到着する状況において耐侵入システムの信頼性を待ち行列理論を用いた確率モデルで解析している．

本稿では，文献 [9] におけるモデルの一般化ならびに一般化されたモデルで耐侵入システムの信頼性評価を行う．具体的には，システムへのリクエスト到着がマルコフ型到着過程 (Markovian Arrival Process) に従う場合を考え，待ち行列解析における行列解析法 (Matrix Geometric Analysis) を適用し，システム障害時間分布の導出を行う．

2. 仮想型耐侵入システム

本稿では Lau ら [5] による仮想型耐侵入システムを考える．図 1 は仮想型耐侵入システムのアーキテクチャを示している．ハイパーバイザ型の仮想化プラットフォームにおいて複数の仮想マシン (VM) がサーバとして外部のクライアントへサービスを提供している．各仮想マシンではハイパーバイザ VMM (Virtual Machine Monitor) が提供する共有メモリを通じて値の読み／書きが行われる．クライアントからアクティブ (Active) となっているすべての仮想マシン（n 台）にリクエストが送信される．Agreement Service (AS) はすべての仮想マシンからの返答を共有メモリから集計し，値の検証を行い，クライアントへ返答する仮想マシンを決定した後に共有メモリを通じて通知する．値の検証は多数決による合意で行われる．合意されなかった場合は，新たに r 台の仮想マシンをアクティベートし，再度リクエスト処理を行う．検証は合意が得られるまで r 台の仮想マシンを増やしながら複数回行われる．m 回目の検証においても合意が得られない場合は，セキュリティ障害が発生したものとしてシステムを停止する．各リクエストに仮想マシンは初期化され，アクティブな仮想マシンは n 台に戻る．

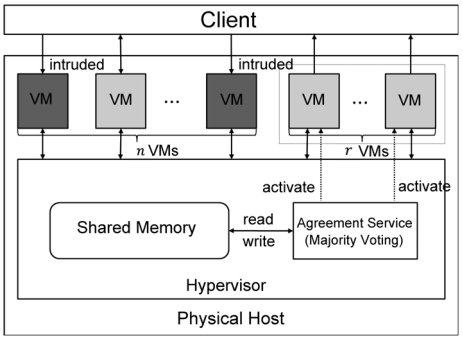

図 1　仮想型耐侵入システムのアーキテクチャ

3. 確率モデルを用いた性能評価

3.1 モデルの概要

前節で説明した仮想型耐侵入システムに対する確率モデルを考える．システムへのリクエストはパラメータ $\boldsymbol{D}_0$, $\boldsymbol{D}_1$ のマルコフ型到着過程（付録参照）に従う．到着したリクエストは FCFS (First Come First Serve) で一つずつシステムによって処理されるものとする．リクエストは複数の仮想マシンで同時に処理される．単一の仮想マシンが単一のリクエストを処理するために要する時間を非負の確率変数 S で表し，その累積分布関数を $G(t) = P(S \leq t)$ とする．悪意あるユーザから仮想マシンへの攻撃は各々独立に行われ，発生率 η のポアソン過程に従う．簡単のため，仮想マシンへの攻撃は必ず成功し，個々の仮想マシンでセキュリティ障害（情報の改ざんなど）が発生するものとする．セキュリティ障害が発生している仮想マシンは任意の振る舞いをする．セキュリティ障害が発生している仮想マシンからの結果を排除するため，耐侵入システムでは各仮想マシンで処理された結果の検証を行う．結果の検証は多数決で行われ，同じ結果（正しく動作している仮想マシンからの結果）が過半数を超えた場合は合意が得られたものとしてシステムは正しい結果を返す．一方で，過半数を超える結果がない場合，システムは r 台の仮想マシンをアクティベートし同様の処理を行う．最終的に m 回の検証を行っても合意が得られない場合はシステム全体のセキュリティ障害としてシステムを停止させる．また，初期においてアクティブな仮想マシン台数を n とする．

ここでは，仮想型耐侵入システムに対して解析的に以下の二つの性能指標を導出する．

- 指標 1：セキュリティ障害が発生するまでの時間分布の LS (Laplace-Stieltjes) 変換
- 指標 2：システムのトラフィック密度

指標 1 は信頼性評価において標準的に使われる「平均障害発生時間 (Mean Time to Failure; MTTF)」に対応する平均セキュリティ障害時間 (Mean Time to Security Failure; MTTSF) の算出を可能とする．また，指標 2 は耐侵入スキームを導入しない場合のトラフィック密度と比較することで，耐侵入スキームのオーバーヘッドを評価することができる．

主な解析の流れとして，まず，単一のリクエストに対する処理成功確率ならびに時間分布の LS 変換を導出し，次に，導出したサービス分布のもとでの $MAP/G/1$ 待ち行列解析を行う．特に，システム全体のセキュリティ障害が発生した場合にはシステムが停止する点が通常の $MAP/G/1$ 待ち行列と比較して異なる点である．

3.2 単一リクエストに対するセキュリティ障害確率の導出

単一の仮想マシンにおいてセキュリティ障害が発生することなく単一リクエストに対するサービスが完了する確率 p_S は以下のようになる．

$$p_S = \int_0^\infty e^{-\eta t} \mathrm{d}G(t). \tag{1}$$

確率変数 $\{N_i, i = 1, 2, \ldots, m\}$ を $i-1$ 回目の検証で合意が得られず i 回目の検証を行った際にセキュリティ障害が発生していない仮想マシンの台数とする．このとき，N_i は次の条件付き確率で表される．

$$P(N_i = y | N_{i-1} = x) = \binom{x+r}{y} {p_S}^y (1-p_S)^{x+r-y}$$
$$y = 0, \ldots, x+r. \tag{2}$$

また，i 回目の検証時におけるアクティブな仮想マシン台数 $n_i = n + r*(i-1)$ に対して，セキュリティ障害が発生している仮想マシンが $f_i = \lfloor (n_i - 1)/2 \rfloor$ 以下であれば合意が得られ，それ以降の検証作業を行わないことになる．そのため，N_i を離散時間マルコフ連鎖とみなしたとき，i 回目の検証で合意を得られないときの推移確率行列は

$$\left[\overline{\boldsymbol{P}}_i\right]_{x,y} = P(N_i = y | N_{i-1} = x), \tag{3}$$
$$x = 0, \ldots, n_{i-1} - f_{i-1} - 1,$$
$$y = 0, \ldots, n_i - f_i - 1$$

となり，i 回目の検証で合意が得られる場合の推移確率行列は

$$[\boldsymbol{P}_i]_{x,y} = P(N_i = y | N_{i-1} = x), \tag{4}$$
$$x = 0, \ldots, n_{i-1} - f_{i-1} - 1,$$
$$y = n_i - f_i, \ldots, x+r.$$

となる．ここで，$\boldsymbol{P}$, $\overline{\boldsymbol{P}}$ はともに正方行列でないことに注意する．また，N_1 に対する確率ベクトル $\overline{\boldsymbol{p}}_1$ を

$$\left[\overline{\boldsymbol{p}}_1\right]_y = \binom{n_1}{y} {p_S}^y (1-p_S)^{n_1-y}, \tag{5}$$
$$y = 0, \ldots, n_1 - f_1 - 1$$

$$\left[\boldsymbol{p}_1\right]_y = \binom{n_1}{y} {p_S}^y (1-p_S)^{n_1-y}, \tag{6}$$
$$y = n_1 - f_1, \ldots, n_1$$

と定義する．このとき，単一のリクエスト処理に対するセキュリティ障害発生確率 $P_{failure}$ ならびに処理成功確率 $P_{success}$ は

$$P_{failure} = \overline{\boldsymbol{p}}_1 \overline{\boldsymbol{P}}_2 \cdots \overline{\boldsymbol{P}}_m \mathbf{1}, \tag{7}$$
$$P_{success} = \boldsymbol{p}_1 \mathbf{1} + \overline{\boldsymbol{p}}_1 \boldsymbol{P}_2 \mathbf{1} + \cdots$$
$$\cdots + \overline{\boldsymbol{p}}_1 \overline{\boldsymbol{P}}_2 \cdots \overline{\boldsymbol{P}}_{m-1} \boldsymbol{P}_m \mathbf{1} \tag{8}$$

となる．

3.3 単一リクエストに対するサービス分布

次に単一リクエストに対するサービス分布を考える．いま τ_i を i 回目の検証にかかる時間を表す確率変数とする．τ_i は n_i 台の仮想マシンの処理のうち最も遅い時間になるため

$$F_{xy}^{(i)}(t) = P(\tau_i \le t, N_i = y | N_{i-1} = x)$$
$$= \binom{x+r}{y} G(t)^{n_{i-1}-x} G_S(t)^y G_F(t)^{x+r-y} \tag{9}$$

を得る．すでに不正アクセスされている仮想マシンが $n_{i-1} - x$ 台あるが，システムはどの仮想マシンが不正アクセスされているか，この時点ではわからないため $G(t)^{n_{i-1}-x}$ の項も含むことに注意する．さらに，$G_S(t)$, $G_F(t)$ は単一の正常な仮想マシンがセキュリティ障害を起こさない／起こすときに処理が時刻 t 以内に終了する確率を表し，以下で与えられる．

$$G_S(t) = \int_0^t e^{-\eta u} \mathrm{d}G(u), \tag{10}$$
$$G_F(t) = \int_0^t (1 - e^{-\eta u}) \mathrm{d}G(u). \tag{11}$$

最終的に τ_i の LS 変換は次のように得られる．

$$\mathrm{E}[e^{-s\tau_i} I(N_i = y) | N_{i-1} = x] = \int_0^\infty e^{-st} \mathrm{d}F_{xy}^{(i)}(t). \tag{12}$$

ここで $I(A)$ は事象 A に対する指標関数を表す．

式 (12) の LS 変換を用いて，3.2 節と同様に以下のような行列を定義する．

- i 回目の検証で合意が得られないとき

$$\left[\overline{\boldsymbol{L}}_i(s)\right]_{x,y} = \mathrm{E}[e^{-s\tau_i} I(N_i = y) | N_{i-1} = x], \quad (13)$$
$$x = 0, \ldots, n_{i-1} - f_{i-1} - 1,$$
$$y = 0, \ldots, n_i - f_i - 1.$$

- i 回目の検証で合意が得られるとき

$$[\boldsymbol{L}_i(s)]_{x,y} = \mathrm{E}[e^{-s\tau_i} I(N_i = y) | N_{i-1} = x], \quad (14)$$
$$x = 0, \ldots, n_{i-1} - f_{i-1} - 1,$$
$$y = n_i - f_i, \ldots, x + r.$$

また，初回の検証については以下の確率および LS 変換を考える．

$$F_y^{(1)}(t) = \binom{n_1}{y} G_S(t)^y G_F(t)^{n_1 - y}, \quad (15)$$

$$\mathrm{E}[e^{-s\tau_1} I(N_1 = y)] = \int_0^\infty e^{-st} \mathrm{d}F_y^{(1)}(t). \quad (16)$$

このとき，次のベクトルを定義する．

- 初回の検証で合意が得られないとき

$$\left[\overline{\boldsymbol{l}}_1(s)\right]_y = \mathrm{E}[e^{-s\tau_1} I(N_1 = y)], \quad (17)$$
$$y = 0, \ldots, n_1 - f_1 - 1.$$

- 初回の検証で合意が得られるとき

$$[\boldsymbol{l}_1(s)]_y = \mathrm{E}[e^{-s\tau_1} I(N_1 = y)], \quad (18)$$
$$y = n_1 - f_1, \ldots, n_1.$$

以上の準備のもと，確率変数 T を単一リクエストに対する処理時間とすると，その LS 変換は以下のように得られる．

$$\mathrm{E}[e^{-sT} \mathcal{I}_S]$$
$$= \boldsymbol{l}_1(s)\mathbf{1} + \overline{\boldsymbol{l}}_1(s)\boldsymbol{L}_2(s)\mathbf{1} + \cdots$$
$$\cdots + \overline{\boldsymbol{l}}_1(s)\overline{\boldsymbol{L}}_2(s)\cdots\overline{\boldsymbol{L}}_{m-1}(s)\boldsymbol{L}_m(s)\mathbf{1}, \quad (19)$$
$$\mathrm{E}[e^{-sT} \mathcal{I}_F]$$
$$= \overline{\boldsymbol{l}}_1(s)\overline{\boldsymbol{L}}_2(s)\cdots\overline{\boldsymbol{L}}_m(s)\mathbf{1}. \quad (20)$$

ここで $\mathcal{I}_S, \mathcal{I}_F$ はそれぞれ単一リクエスト処理中にセキュリティ障害が起きない／起きる事象を表す指標確率変数である．

3.4 セキュリティ障害時間分布の導出

$MAP/G/1$ 待ち行列の解析手法を応用して耐侵入システムでセキュリティ障害が発生するまでの時間分布を導出する．いま，B を $MAP/G/1$ 待ち行列における単一のビジー期間の時間の長さを表す確率変数とする．特に，システム障害が起こった場合は障害発生

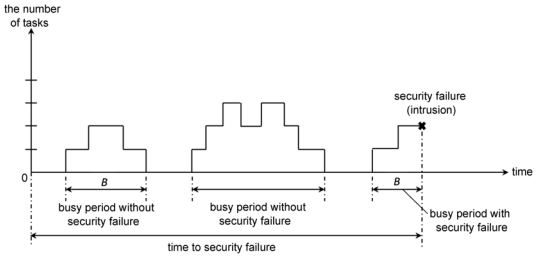

図 2　システム内のリクエスト数の振る舞い

までの期間をビジー期間と定める（図 2）．このとき，確率変数 B に対して次の LS 変換を考える．

$$B_{i,j}(s) = \mathrm{E}[e^{-sB} I(J(B) = j)\mathcal{B}_S | J(0) = i], \quad (21)$$
$$Z_{i,j}(s) = \mathrm{E}[e^{-sB} I(J(B) = j)\mathcal{B}_F | J(0) = i]. \quad (22)$$

ここで $\mathcal{B}_S, \mathcal{B}_F$ はビジー期間中にシステムのセキュリティ障害が起こらない／起こる事象を表す指標確率変数である．また，$J(t)$ は時刻 t での到着過程の位相を表している．つまり $B_{i,j}(s)$ はビジー期間の最初の位相が i という条件のもと，ビジー期間中にセキュリティ障害が発生せず，かつ，ビジー期間終わりでの位相が j であるときのビジー期間の長さの LS 変換を表している．さらに，$B_{i,j}(s)$ と $Z_{i,j}(s)$ を (i,j) 要素とする行列 $\boldsymbol{B}(s), \boldsymbol{Z}(s)$ を定義して，ビジー期間で最初に処理するリクエストを考えると，次の式を得る．

$$\boldsymbol{B}(s) = \int_0^\infty \sum_{n=0}^\infty \boldsymbol{P}(n,t) e^{-st} \boldsymbol{B}(s)^n \mathrm{d}H_S(t). \quad (23)$$

ここで $\boldsymbol{P}(n,t)$ は付録の式 (43) で定義される MAP の推移確率行列，$H_S(t)$ は式 (19) のセキュリティ障害が発生しないときの単一リクエストに対するサービス時間に対する累積分布関数を表している．$MAP/G/1$ 待ち行列解析と同様に，次の行列 $\boldsymbol{A}_n(s)$ を定義する．

$$\boldsymbol{A}_n(s) = \int_0^\infty e^{-st} \boldsymbol{P}(n,t) \mathrm{d}H_S(t). \quad (24)$$

このとき，式 (23) は

$$\boldsymbol{B}(s) = \sum_{n=0}^\infty \boldsymbol{A}_n(s) \boldsymbol{B}(s)^n \quad (25)$$

となる．同様に

$$\boldsymbol{Z}(s) = \int_0^\infty \sum_{n=0}^\infty \boldsymbol{P}(n,t) e^{-st} \mathrm{d}H_F(t)$$
$$+ \int_0^\infty \sum_{n=1}^\infty \boldsymbol{P}(n,t) e^{-st} \sum_{l=1}^n \boldsymbol{B}(s)^{l-1} \boldsymbol{Z}(s) \mathrm{d}H_S(t). \quad (26)$$

ここで $H_F(t)$ は式 (20) のセキュリティ障害が発生するときの単一リクエストに対するサービス時間に対する累積分布関数を表している．さらに，以下の LS 変換を定義する．

$$\boldsymbol{G}_S^*(s) = \int_0^\infty e^{-st} \exp((\boldsymbol{D}_0 + \boldsymbol{D}_1)t) \mathrm{d}H_S(t), \quad (27)$$

$$\boldsymbol{G}_F^*(s) = \int_0^\infty e^{-st} \exp((\boldsymbol{D}_0 + \boldsymbol{D}_1)t) \mathrm{d}H_F(t). \quad (28)$$

これを用いて式 (26) を整理すると，次の式を得る．

$$\boldsymbol{Z}(s) = (\boldsymbol{I} - \boldsymbol{B}(s))(\boldsymbol{I} - \boldsymbol{G}_S^*(s))^{-1} \boldsymbol{G}_F^*(s). \quad (29)$$

ここで $\boldsymbol{I}$ は単位行列である．最終的に，アイドル期間に関する LS 変換

$$\boldsymbol{Y}(s) = \int_0^\infty e^{-st} \exp(\boldsymbol{D}_0 t) \boldsymbol{D}_1 \mathrm{d}t \quad (30)$$

を定義すると，システムのセキュリティ障害発生時間 X に対する LS 変換 $[\boldsymbol{X}(s)]_{i,j} = \mathrm{E}[e^{-sX} I(J(X) = j) | J(0) = i]$ は

$$\begin{aligned} \boldsymbol{X}(s) &= \boldsymbol{Y}(s) \sum_{n=0}^{\infty} (\boldsymbol{B}(s)\boldsymbol{Y}(s))^n \boldsymbol{Z}(s) \\ &= \boldsymbol{Y}(s)(\boldsymbol{I} - \boldsymbol{B}(s)\boldsymbol{Y}(s))^{-1} \\ &\quad \times (\boldsymbol{I} - \boldsymbol{B}(s))(\boldsymbol{I} - \boldsymbol{G}_S^*(s))^{-1} \boldsymbol{G}_F^*(s) \quad (31) \end{aligned}$$

となる．

LS 変換から得られる平均値 $\mathrm{E}[X]$ が平均セキュリティ障害時間 (MTTSF) となる．実際，時刻 $t = 0$ における到着過程の位相に対する確率ベクトルを $\boldsymbol{\pi}_0$ とすると，MTTSF は

$$\begin{aligned} \mathrm{MTTSF} = &\frac{\mathrm{E}[T]}{P_{failure}} \\ &+ \boldsymbol{\pi}_0 (\boldsymbol{I} - \boldsymbol{Y}(0)\boldsymbol{B}(0))^{-1} \left(-\boldsymbol{Y}'(0)\right) \boldsymbol{1} \end{aligned} \quad (32)$$

となる．ここで

$$\boldsymbol{Y}(0) = (-\boldsymbol{D}_0)^{-1} \boldsymbol{D}_1, \quad (33)$$

$$\boldsymbol{Y}'(0) = -(-\boldsymbol{D}_0)^{-2} \boldsymbol{D}_1 \quad (34)$$

であり，$\mathrm{E}[T]$, $P_{failure}$ は単一のリクエストに対する平均処理時間および単一のリクエスト処理中にセキュリティ障害が発生する確率を表している．また，式 (32) の第二項はセキュリティ障害が発生するまでの累積のアイドル時間の期待値を表している．そのため，リクエストの到着過程がポアソン過程の場合，MTTSF は単一ビジー期間でセキュリティ障害が発生しない確率 $B(0)$ を用いて，

$$\mathrm{MTTSF} = \frac{\mathrm{E}[T]}{P_{failure}} + \frac{1}{\lambda(1 - B(0))} \quad (35)$$

として表される．一方で，$B(0)$ は次の方程式の解であることに注意する．

$$B(0) = H_S^*(\lambda(1 - B(0))). \quad (36)$$

ここで $H_S^*(s)$ は式 (19) で表される単一のリクエストに対するサービス時間分布の LS 変換である．

最後にトラフィック密度について，耐侵入スキームを実装しない場合のシステムの基本的なトラフィック密度は $\rho_0 = \lambda \mathrm{E}[\mathrm{S}]$ であるのに対して，耐侵入スキームでは単一のリクエスト処理時間が長くなるため，実際のトラフィック密度は $\rho_e = \lambda \mathrm{E}[T]$ となる．

4. 数値例

ここでは，前述した結果に基づいて耐侵入システムの性能評価を行う．まず耐侵入システムの設計パラメータとして以下の場合を考える．

- 初期状態の仮想マシン台数：$n_1 = 2, 3, 4, 5$.
- 検証後にアクティベーションする仮想マシン台数：$r = 1, 2$.
- 最大検証回数：$m = 1, 2, 3, 4$.

また，単一の仮想マシンに対する悪意ある攻撃の発生率を $\eta = 1/360000[1/\mathrm{ms}]$ とする．これは一般的な環境と比較して非常に高い発生率を設定していることになる．さらに，単一の仮想マシンが単一のリクエストを処理するサービス時間分布が平均 $\mathrm{E}[S] = 100[\mathrm{ms}]$ の 2 次アーラン分布に従うものとする．リクエスト到着は，マルコフ型到着過程の特殊な場合としてポアソン過程を仮定し，システムの基本的なトラフィック密度が $\rho_0 = 0.1, 0.5$ となるような二つの到着率の場合を考える．

表 1 は基本トラフィック密度が $\rho_0 = 0.1$ であるときの MTTSF を示している．表では，攻撃発生や処理時間に設定した ms の単位から年の単位に変更している．この結果において，n および m が増加すると MTTSF も大きくなることがわかる．つまり，仮想マシンの台数が多くなると MTTSF が長くなる傾向がある．しかしながら，$r = 1$ の場合において，m に対して MTTSF が単調に増加しているわけではない．たとえば，$r = 1$ のとき，$n = 1$ と $m = 1$ の場合の MTTSF は 0.137 年であり，これは $n = 3$ かつ $m = 2$ の場合の MTTSF と同じ値となる．また，n が偶数のとき，$r = 2$ としたときの MTTSF が $r = 1$ とした場合の MTTSF よりも悪くなる場合がある．これらの結果か

表 1　$\rho_0 = 0.1$ のときの平均セキュリティ障害時間（MTTSF）

(a) $r = 1$

	$m=1$	$m=2$	$m=3$	$m=4$
$n=2$	5.71e−5	8.23e−2	8.23e−2	5.29e+1
$n=3$	1.37e−1	1.37e−1	9.25e+1	9.25e+1
$n=4$	6.86e−2	6.73e+1	6.73e+1	2.57e+4
$n=5$	1.48e+2	1.48e+2	5.88e+4	5.88e+4

(b) $r = 2$

	$m=1$	$m=2$	$m=3$	$m=4$
$n=2$	5.71e−5	5.88e−2	4.12e+1	1.97e+4
$n=3$	1.37e−1	1.48e+2	6.41e+4	8.88e+6
$n=4$	6.86e−2	5.29e+1	2.48e+4	2.20e+6
$n=5$	1.48e+2	8.86e+4	1.24e+7	5.04e+9

表 2　$\rho_0 = 0.5$ のときの平均セキュリティ障害時間（MTTSF）

(a) $r = 1$

	$m=1$	$m=2$	$m=3$	$m=4$
$n_1=2$	1.14e−5	1.65e−2	1.65e−2	1.06e+1
$n_1=3$	2.74e−2	2.74e−2	1.84e+1	1.84e+1
$n_1=4$	1.34e−2	1.20e+1	1.20e+1	5.23e+3
$n_1=5$	2.82e+1	2.82e+1	1.36e+4	1.36e+4

(b) $r = 2$

	$m=1$	$m=2$	$m=3$	$m=4$
$n_1=2$	1.14e−5	1.18e−2	8.23e+0	3.90e+3
$n_1=3$	2.74e−2	2.92e+1	1.69e+4	8.78e+6
$n_1=4$	1.34e−2	9.39e+0	5.01e+3	2.07e+6
$n_1=5$	2.82e+1	2.24e+4	1.22e+7	5.04e+9

表 3　$\rho_0 = 0.1$ のときの耐侵入システムのトラフィック密度

(a) $r = 1$

	$m=1$	$m=2$	$m=3$	$m=4$
$n_1=2$	0.13750	0.13759	0.13759	0.13759
$n_1=3$	0.16065	0.16065	0.16065	0.16065
$n_1=4$	0.17736	0.17736	0.17736	0.17736
$n_1=5$	0.19041	0.19041	0.19041	0.19041

(b) $r = 2$

	$m=1$	$m=2$	$m=3$	$m=4$
$n_1=2$	0.13750	0.13760	0.13760	0.13760
$n_1=3$	0.16065	0.16065	0.16065	0.16065
$n_1=4$	0.17736	0.17736	0.17736	0.17736
$n_1=5$	0.19041	0.19041	0.19041	0.19041

表 4　$\rho_0 = 0.5$ のときの耐侵入システムのトラフィック密度

(a) $r = 1$

	$m=1$	$m=2$	$m=3$	$m=4$
$n_1=2$	0.68750	0.68795	0.68795	0.68795
$n_1=3$	0.80324	0.80324	0.80324	0.80324
$n_1=4$	0.88679	0.88679	0.88679	0.88679
$n_1=5$	0.95207	0.95207	0.95207	0.95207

(b) $r = 2$

	$m=1$	$m=2$	$m=3$	$m=4$
$n_1=2$	0.68750	0.68799	0.68799	0.68799
$n_1=3$	0.80324	0.80324	0.80324	0.80324
$n_1=4$	0.88679	0.88679	0.88679	0.88679
$n_1=5$	0.95207	0.95207	0.95207	0.95207

ら，初期状態の仮想マシン台数は奇数であり，かつ r が 1 よりも大きい必要があることがわかる．表 2 は $\rho_0 = 0.5$ の場合の MTTSF を示している．表 1 と比較すると MTTSF は若干減少していることがわかる．また，その他の傾向については $\rho_0 = 0.1$ の場合と同様である．

表 3 は $\rho_0 = 0.1$ の場合における耐侵入スキームによるオーバーヘッドを考慮した実際のトラフィック密度を示している．これらの表から，基本のトラフィック密度 ρ_0 と比べると実際のトラフィック密度が増加していることがわかる．特に，実際のトラフィック密度は初期状態における仮想マシンの台数に強く依存している．表 4 は $\rho_0 = 0.5$ の場合における実際のトラフィック密度を示している．表 3 と比べると，実際のトラフィック密度は非常に大きく増加しており，もともとのトラフィックが高いシステムに耐侵入スキームを導入する際には十分な考慮が必要であることがわかる．

これまでの結果を総合すると，仮想型耐侵入システムにおいては $n = 3, r = 2$ が信頼度と処理オーバーヘッドの観点から最もバランスがとれた設計パラメータであり，セキュリティ障害となるまでの検証回数 m はシステムの最大で利用できるリソース（仮想マシン台数）を考慮して設計するのがよいものと考えられる．

5. まとめ

本稿ではマルコフ型到着過程に従ったリクエストが発生するもとで，仮想型耐侵入システムに対する確率モデルを構築し，待ち行列理論で用いられる行列解析法によりシステムの信頼性に関する尺度の定式化を行った．導出されたセキュリティ障害時間分布の LS 変換

から平均セキュリティ障害時間 MTTSF の導出が行えるだけでなく，Durbin 法や Gaver 法などの逆ラプラス変換 [10] を適用することにより具体的な障害時間分布の評価が行える．一方，確率モデルを用いた性能評価では実際のデータとの整合性が問題となることが多い．本モデルであれば，リクエスト到着過程としてのマルコフ型到着過程，セキュリティ障害の発生事象に対するポアソン過程，単一リクエスト単一仮想マシンのサービス時間分布などをデータから同定する必要がある．ポアソン過程や一般分布については，最尤法や各種の適合度検定が有用である．マルコフ型到着過程については，ポアソン過程や一般分布と比較すると難しい手続きを必要とするため，推定パッケージ [11] などの利用が有効である．

参考文献

[1] F. Wang, F. Gong, C. Sargor, K. Goseva, K. S. Trivedi and F. Jou, "Scalable intrusion tolerance architecture for distributed server," In *Proceedings of the 2nd IEEE SMC Information Assurance Workshop*, pp. 38–45, 2001.

[2] M. G. Merideth, A. Iyengar, T. Mikalsen, S. Tai, I. Rouvellou and P. Narasimhan, "Thema: Byzantine-fault-tolerant middleware for web-service applications," In *Proceedings of the 24th IEEE Symposium on Reliable Distributed Systems (SRDS 2005)*, pp. 131–140, 2005.

[3] W. Zhao, "BFT-WS: A Byzantine fault tolerance framework for web services," In *Proceedings of the 11th International IEEE Conference Enterprise Distributed Object Computing Conference Workshop (EDOCW 2007)*, pp. 89–96, 2007.

[4] V. S. Junior, L. C. Lung, M. Correia, J. S. Fraga and J. Lau, "Intrusion tolerant services through virtualization: A shared memory approach," In *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA 2010)*, pp. 768–774, 2010.

[5] J. Lau, L. Barreto and J. S. Fraga, "An infrastructure based in virtualization for intrusion tolerant services," In *Proceedings of the 19th IEEE International Conference on Web Services (ICWS 2012)*, pp. 170–177, 2012.

[6] B. B. Madan, K. Goseva-Popstojanova, K. Vaidyanathan and K. S. Trivedi, "A method for modeling and quantifying the security attributes of intrusion tolerant systems," *Performance Evaluation*, **56**, pp. 167–186, 2004.

[7] T. Uemura and T. Dohi, "Quantitative evaluation of intrusion tolerant systems subject to DoS attacks via semi-Markov cost models," In *Proceeding of International Conference on Embedded and Ubiquitous Computing (EUC 2007)*, *LNCS*, **4809**, pp. 31–42, 2007.

[8] J. Zheng, H. Okamura and T. Dohi, "Survivability analysis of VM-based intrusion tolerant systems," *IEICE Transactions on Information & Systems (D)*, **E98-D**, pp. 2082–2090, 2015.

[9] J. Zheng, H. Okamura and T. Dohi, "Performance evaluation of VM-based intrusion tolerant systems with Poisson arrivals," In *Proceedings of the 4th International Symposium on Computing and Networking (CANDAR 2016)*, pp. 181–187, 2016.

[10] M. R. Naeeni, R. Campagna, M. Eskandari-Ghadi and A. A. Ardalan, "Performance comparison of numerical inversion methods for Laplace and Hankel integral transforms in engineering problems," *Applied Mathematics and Computation*, **250**, pp. 759–775, 2015.

[11] H. Okamura and T. Dohi, "Mapfit: An R-based tool for PH/MAP parameter estimation," In *Proceedings of the 14th International Conference on Quantitative Evaluation of Systems (QEST 2015)*, *LNCS*, **9295**, pp. 105–112, 2015.

付録．マルコフ型到着過程

マルコフ型到着過程 (Markovian Arrival Process; MAP) は客の到着（イベントの発生）時間間隔が同じ推移率行列（無限小生成行列）をもつ相型分布に従い，かつ，相型分布の初期状態が離散時間マルコフ連鎖によって支配される確率過程である．特別な例として，ポアソン過程，マルコフ変調ポアソン過程，相型再生過程を含む．一般の到着過程に対して稠密であることが知られており，情報通信ネットワークにおける到着過程をモデル化するのに利用されている．

いま，MAP において $N(t)$ を時刻 t までに到着した客の累積数，$J(t)$ を時刻 t における到着時間間隔分布の位相を表す確率過程とする．このとき，$N(t), J(t)$ は次の無限小生成行列をもつ連続時間マルコフ連鎖 (Continuous-Time Markov Chain; CTMC) で定義される．

$$\boldsymbol{Q}_{\mathrm{MAP}} = \begin{pmatrix} \boldsymbol{D}_0 & \boldsymbol{D}_1 & & \\ & \boldsymbol{D}_0 & \boldsymbol{D}_1 & \\ & & \ddots & \ddots \end{pmatrix}. \tag{37}$$

ここで，$\boldsymbol{D}_0$ は客の到着を伴わない位相変化を表す無限小生成行列，$\boldsymbol{D}_1$ は客の到着を伴った位相変化を表す無限小生成行列である．また，$J(T)$ は $\boldsymbol{D}_0 + \boldsymbol{D}_1$ を無限小生成行列とした CTMC であり，その定常分布（行ベクトル）を $\boldsymbol{\pi}_s$ とすると，MAP の到着率は

$$\lambda = \boldsymbol{\pi}_s \boldsymbol{D}_1 \mathbf{1} \tag{38}$$

として表される．ここで $\mathbf{1}$ はすべての要素が 1 の列ベクトルである．さらに行ベクトル

$$\boldsymbol{\pi}_d = \frac{\boldsymbol{\pi}_s \boldsymbol{D}_1}{\boldsymbol{\pi}_s \boldsymbol{D}_1 \mathbf{1}} \tag{39}$$

を定義すると，定常状態での到着時間間隔 X_0 は次の密度関数をもつ相型分布で表される．

$$f_{X_0}(t) = \boldsymbol{\pi}_d \exp(\boldsymbol{D}_0 t)\boldsymbol{D}_1\mathbf{1}. \tag{40}$$

よって，到着時間間隔の変動係数の二乗 (squared coefficient of variation; c_v^2) および k 番目の到着時間間隔 X_k との相関 (lag-k autocorrelation; τ_k) は次のようになる．

$$c_v^2 = \frac{\mathrm{E}[X_0^2] - \mathrm{E}[X_0]^2}{\mathrm{E}[X_0]^2} = \frac{2\boldsymbol{\pi}_d(-\boldsymbol{D}_0)^{-2}\mathbf{1}}{\left(\boldsymbol{\pi}_d(-\boldsymbol{D}_0)^{-1}\mathbf{1}\right)^2} - 1, \tag{41}$$

$$\begin{aligned} r_k &= \frac{\mathrm{E}[X_0X_k] - \mathrm{E}[X_0]^2}{\mathrm{E}[X_0^2] - \mathrm{E}[X_0]^2} \\ &= \frac{1}{c_v^2}\left(\frac{\boldsymbol{\pi}_d(-\boldsymbol{D}_0)^{-1}\boldsymbol{P}^k(-\boldsymbol{D}_0)^{-1}\mathbf{1}}{\left(\boldsymbol{\pi}_d(-\boldsymbol{D}_0)^{-1}\mathbf{1}\right)^2} - 1\right). \end{aligned} \tag{42}$$

ここで $\boldsymbol{P} = (-\boldsymbol{D}_0)^{-1}\boldsymbol{D}_1$ である．

行列 $\boldsymbol{P}(n,t)$ の (i,j) 要素を

$$\begin{aligned} &[\boldsymbol{P}(n,t)]_{i,j} \\ &= P(N(t) = n, J(t) = j | N(0) = 0, J(0) = i), \end{aligned} \tag{43}$$

と定義すると，次の微分差分方程式を満たす．

$$\frac{d}{dt}\boldsymbol{P}(0,t) = \boldsymbol{P}(0,t)\boldsymbol{D}_0, \tag{44}$$

$$\frac{d}{dt}\boldsymbol{P}(n,t) = \boldsymbol{P}(n,t)\boldsymbol{D}_0 + \boldsymbol{P}(n-1,t)\boldsymbol{D}_1, \quad n = 1,2,3,\ldots. \tag{45}$$

これより以下の結果を得る．

$$\begin{aligned} \boldsymbol{P}^*(z,t) &= \sum_{n=0}^{\infty} z^n \boldsymbol{P}(n,t) \\ &= \exp((\boldsymbol{D}_0 + z\boldsymbol{D}_1)t). \end{aligned} \tag{46}$$