F-023

SQL インジェクション攻撃検知の為のアクセスログマイニング
Mining Access Logs for SQL injection Attack Detection

広瀬 俊亮† 山形 昌也† 山西 健司† 岩井 博樹‡
Shunsuke Hirose Masaya Yamagata Kenji Yamanishi Hiroki Iwai

1 はじめに

近年、SQL インジェクション攻撃の検出問題がインターネットセキュリティの分野で大きく注目されている。この攻撃は Web アプリケーションへの攻撃の一つであり個人情報の流出等の深刻な被害をもたらす。しかし、この攻撃は通常シグニチャを構成できるほど定型的なパターンを持たない為、シグニチャベースの検出は難しい。従って、Web アクセスログから SQL インジェクション攻撃をより早くより少ない誤報の下で検出するという問題は非常に重要である。本稿では、Web アクセスログを入力として SQL インジェクション攻撃をより早くより少ない誤報で検出することを目的とする。

データの通常のパターンから外れたものを異常として検出することでシグニチャでの表現が難しい攻撃を検知するマイニングベースの手法が、Web アプリケーションへの攻撃検出に関して数多く提案されている [1, 2]。[1] では珍しいアクセスを検出して、[2] では Web ページの閲覧時間とそのページのデータサイズとの相関関係が通常と異なるアクセスを検出して攻撃を検出している。

Web アクセスログはアクセスの種類 (カテゴリカル変数) と各種類のアクセスの回数 (数値) の二種類の情報を含むヘテロなデータである。しかし、既存手法の多くは両者の内片方のみに注目した異常検出である。

本稿では、この点に注目した SQL インジェクション攻撃検出の手法を提案する。検出手法に含まれる主要なアイデアは、(1) アクセス回数の変化点検出によって異常なトラフィックを伴うアクセスを検出する、(2) アクセスの種類を入力として異常な種類のアクセスを検出する、(3) 1 と 2 を組み合わせて SQL インジェクション攻撃に対応する異常なアクセスを検出する、の三点である。実際の SQL インジェクション攻撃を含むデータを用いた検出実験を行い、回数と種類の両方を考慮することで片方のみを考慮した場合と比較して攻撃を早く正確に検出可能であることを示す。

2 問題設定

本稿では、SQL インジェクション攻撃をより早くより少ない誤報で検出することを目的とする。

Web アクセスログを入力とする。一回のアクセスがアクセスログの一行として記録される。各行はタイムスタンプ t、IP アドレスのようなカテゴリカル変数及びファイルサイズのような連続変数を含む。本稿ではカテゴリカル変数とその出現回数とを攻撃検出に用いる。本稿では、アクセスの異常度を表すスコアを算出して高スコアのアクセスを攻撃の開始に対応するアクセスと見做すという攻撃検出の問題を扱う。

† NEC 共通基盤ソフトウェア研究所
Common Platforms Software Research Labs., NEC
‡ LAC コンピューターセキュリティ研究所
Computer Security Laboratory, LAC

3 提案手法

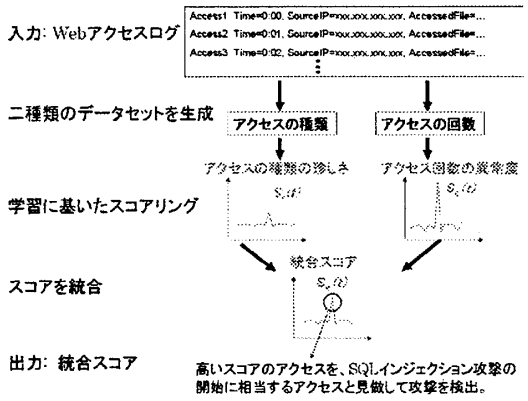

図 1: 提案手法の概要。2 種類のスコアを統合して SQL インジェクションの開始を表すスコアを算出する。

以下では提案手法について述べる。提案手法の概要を図 1 にまとめた。提案手法は以下の 3 ステップから成る。

Step 1 カウントベクトルの生成

アクセスログから、アクセスの種類と出現回数とを成分に持つ量であるカウントベクトルの時系列を作る。カウントベクトルの定義は以下で述べる。

アクセスログに含まれる 2 つのカテゴリカル変数に注目する。タイムスタンプ t のアクセスの持つそれらの変数の値を $\alpha(t) = (a_1(t), a_2(t))$ とする。各変数はそれぞれ離散シンボル集合 $\mathcal{A}_i (i = 1, 2)$ に属する $(a_i(t) \in \mathcal{A}_i)$。$(a_1(t), a_2(t))$ が取り得る値を $\{\alpha_1, \alpha_2, \cdots, \alpha_M\} = \mathcal{A}$ $(\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2)$ と表す。例えば a_1 と a_2 はアクセスされたアプリケーションの名前とその引数等のように採る。この場合、$\mathcal{A}_1$ と $\mathcal{A}_2$ はそれぞれの取りうる値の集合となり、これらの組合せが取り得る値の集合が $\mathcal{A}$ となる。

アクセスログを時間幅 Δt のスロットに区切る (本稿では $\Delta t = 5$ 分とした)。l 番目のスロット中の $\alpha(t) = \alpha_i$ であるアクセスの出現回数を $n_i(l)$、$n_i(l) (i = 1, \cdots, M)$ の最大値を $\hat{n}(l)$、それに対応する離散変数の組を $\hat{\alpha}(l) = (\hat{a}_1(l), \hat{a}_2(l))$ とする。このときカウントベクトルを $(\hat{\alpha}(l), \hat{n}(l))$ として定義する。

カウントベクトルは特徴的なアクセスの種類と頻度とを表す。$\hat{n}(l)$ は同一種類のアクセス数を表し、これの急激な変化は異常なトラフィックの出現に対応する。$\hat{\alpha}(l) = (\hat{a}_1(l), \hat{a}_2(l))$ はアクセスの種類を表し、稀な $\hat{\alpha}(l)$ の出現は珍しい種類のアクセスの出現に対応する。

Step 2-1 トラフィックの異常さのスコアリング

アクセス回数 $\{\hat{n}(l)\}$ の系列からタイムスロット毎にトラフィックの異常さを表すスコア S_a を算出する。トラフィックの急激な変化を異常だと見做し、系列 $\{\hat{n}(l)\}$ の変化度を S_a とする。

変化度の算出には [4] で提案されている以下の変化点検出手法を用いる。$\hat{n}_l$ を AR モデルでオンライン忘却型学習し、確率密度関数 $p_{AR}(\hat{n}(l)|\hat{n}(l-1))$ $((l=1,2,\cdots))$ を得る。ウィンドウ幅を w とし、このモデルに於ける $\hat{n}(l)$ の外れ値スコア $-\log p_{AR}(\hat{n}(l)|\hat{n}(l-1))$ の移動平均 $v_l = -\frac{1}{w}\sum_{i=l-w+1}^{l}\log P_{AR}(\hat{n}(l)|\hat{n}(l-1))$ を新たな時系列として AR モデルで学習し、その確率密度関数を $\tilde{p}_{AR}(v_l|v^{l-1})$ とする。各タイムスロット l で、$-\log\tilde{p}_{AR}(v_l|v^{l-1})$ を時系列の変化度を表すスコアとする。これはバースト的な急激な変化が起こっている点で高いスコアを与える。

Step 2-2 アクセスの珍しさのスコアリング

アクセスの種類 $\{\hat{\alpha}(l)\}$ の系列からタイムスロット毎にアクセスの種類の珍しさを表すスコア S_r を算出する。

S_r の算出には [3] で提案されている以下の手法を用いる。$\hat{\alpha}_l = (\hat{a}_1(l), \hat{a}_2(l))$ を混合隠れマルコフモデルでオンライン忘却型学習し、得られた確率密度関数を $p(\hat{\alpha}(l))$ $((l=1,2,\cdots))$ とする。各タイムスロット l で、$-\log\tilde{p}(\hat{\alpha}(l))$ を $\hat{\alpha}(l)$ の異常度を表すスコア S_r とする。通常は現れない種類のアクセスの S_r は高くなる。

Step 3 スコアの統合

$S_a(l)$ と $S_r(l)$ とを用いて、統合スコア $S_w(l)$ を求める。統合スコアを以下のように定義する。

$$S_w(l) = \big(S_a(l) - T_{aq}(l)\big)\big(S_r(l) - T_{rq}(l)\big) \times\theta\big(S_a(l) - T_{aq}(l)\big)\theta\big(S_r(l) - T_{rq}(l)\big). \quad (1)$$

$T_{rq}(l)$ と $T_{aq}(l)$ は S_r と S_a の q パーセンタイル点を表す (本稿の実験では $q = 5\%$ とした)。$\theta(x)$ は階段関数 $(\theta(x\leq 0) = 0,\ \theta(x > 1) = 1)$。

$S_w(l)$ の値が大きいほど攻撃の開始点らしいと見做して、攻撃の開始点を検出する。$S_w(l)$ の定義より、高いスコアは珍しい種類のアクセスの急激な増加に対応する。

4 実験

提案手法を用いて SQL インジェクション検出の実験を行った。実験に用いたログは 3 日分の Web アクセスログで、実際の SQL インジェクション攻撃を 2 箇所含む。2 回の攻撃の内、最初の攻撃は失敗、2 回目の攻撃は成功していた。2 回の攻撃の開始時点の時間差は 22 時間だった。従って、1 回目の攻撃は情報漏洩 (攻撃の成功) の予兆が漏洩の 22 時間前に現れたものと見做せる。

3 種類のスコア S_a, S_r, S_w を用いて攻撃の開始点を検出した。スコアに閾値を設けてスコアが閾値を超えた場合に攻撃があったと判断した。3 つのスコアを比較することで、異常なトラフィック若しくは稀なアクセスのみを見る場合と両者を見る場合との検出精度を比較した。

攻撃検出の早さを検出性能の指標とした。検出の早さを表す量として平均 benefit を用いた。スロット l^* から攻撃が開始されてその後 l_b スロット以内に攻撃を検出する必要があるとする (本稿では $l_b = 6(= 30$ 分) とした)。攻撃開始後 l スロットで攻撃が検出された場合に benefit は $1 - \frac{l-l^*}{l_b}$ $(0\leq l - l^* < l_b)$ と表される。benefit は $0\leq$benefit≤ 1 の範囲の値をとり、攻撃を時間差 0 で検出できた場合が 1 で検出が遅いほど小さくなる。攻撃について benefit の平均をとったものを平均 benfit とする。

表 1: SQL インジェクション攻撃検出の結果。

スコア	$\mathcal{R}$	γ
S_r(稀なアクセス)	0.10	1.8×10^{-2}(5.2 alarms/day)
S_a(異常なトラフィック)	0.75	3.4×10^{-3}(0.97 alarms/day)
S_w(統合スコア)	0.88	2.3×10^{-3}(0.66 alarms/day)

閾値を変化させて行き、横軸にアラーム率を縦軸に平均 benefit を採った曲線を描いた。この曲線から以下の二つの量を算出し、検出の早さの指標とした： (1) $0 <$ アラーム率 < 0.005 の範囲での曲線の下側の面積を 1 に規格化したもの $\mathcal{R}$ $(0\leq\mathcal{R}\leq 1$。アラーム率の上限を 0.5% とした際の平均 benefit のアラーム平均) 、 (2) 平均 benefit$= 1$ となるアラーム率の最小値 γ (時間差 0 で全ての攻撃を検出するのに必要なアラームの発生頻度)。

3 種類のスコアを用いた検出の結果を表 1 にまとめた。この結果から、異常なトラフィックを検出することで時間差 0 という早さ、1 日当り 1 件程度のアラームという正確さで SQL インジェクション攻撃の開始を検出できたと言える。加えて、トラフィックのみでなくアクセスの種類も考慮することで、更に早く攻撃の開始を検出できたと言える。アクセスの種類も考慮した場合、トラフィックのみの場合と比較して $\mathcal{R}$ の値で比較して 17%、γ の値で比較して 32%、攻撃の開始を早く検出できていた。

提案手法により 2 件の攻撃両方の開始点を検出できた。これは情報漏洩の予兆を漏洩の 22 時間前に検出できたことを意味する。攻撃の失敗と成功との時間差は常に 22 時間もあるわけではなく、場合によって異なる。しかし、SQL インジェクション攻撃は開始後即座に成功するとは限らないことから、少なくともこの結果から提案手法によって情報漏洩の予兆を捉えられると期待される。

5 結論

本稿では Web アクセスログからの SQL インジェクション攻撃検出手法を提案した。提案手法は稀な種類のアクセスと異常なトラフィックの検出及びこれら二つの統合とからなる。実際の攻撃を含むログを用いた実験では種類とトラフィックの両方を考慮することで片方のみを考慮した場合と比較して攻撃を早く正確に検出できた。SQL インジェクション攻撃は必ずしも即座には成功しないので、提案手法を用いて攻撃の開始点を早く検出することで情報漏洩の予兆を捉えられると期待される。

参考文献

[1] C. Kruegel and G. Vigna Anomaly Detection of Web-based Attacks. In *Proc. of ACM Conference on Computer and Communication Security(CCS2003)*, 2003.

[2] T. Yatagai,T. Isohara and I.Sasase Detection of HTTP-GET flood Attack Based on Analysis of Page Access Behavior. In *Proc. of IEEE Pacific Rim Conference on Communiocations, Computers and Signal Processing(PACRIM2007)*, 2007.

[3] K. Yamanishi and Y. Maruyama. Dynamic syslog mining for network failure monitoring. In *Proc. of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD2005)*, 2005.

[4] K. Yamanishi and J. Takeuchi. Unifying framework for detecting outliers and change points from non-stationary time series data. In *Proc. of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD2002)*, 2002.