Vol.2020-CSEC-90 No.16
Vol.2020-SPT-38 No.16
2020/7/20

脆弱性データベースを使用した脅威分析 ートピックモデル分析による攻撃事例と大規模脆弱性DBの突合手法の複数事例への適用ー

小柳 洋貴[1]　寶木 和夫[2]　三科 雄介[2]　梅澤 克之[1,2]

概要：製品開発の初期段階からセキュリティを考慮した設計を行うことは重要である．これを支援する脅威分析支援ツールは多々開発研究されている．しかし開発初期段階の自然言語で書かれる各種設計書などに内在する脅威を判定することは難しい．また扱えるデータ数に制限があるものも少なくない．そして過去の攻撃事例に酷似した攻撃も珍しくない．そこで我々はトピックモデルの手法の 1 つである LatentDirichlet Allocation(LDA) を用いて大規模な脆弱性データベース (DB) と既存の攻撃事例を突合し，その攻撃で使われた脆弱性に類似する既存の脆弱性を抽出する手法を提案した．本論文ではその提案手法を複数事例に適用し，その有効性を検証する．

Threat analysis using the vulnerability database ー Application of Attack Cases and Large-Scale Vulnerability DB Collation Methodology to Multiple Cases Using Topic Model Analysis ー

1. はじめに

1990 年代以降から，サイバー脅威に関する情報共有の必要性は高まっている．近年では，2015 年，2016 年のウクライナ送電網に対するサイバー攻撃，2016 年の Mirai ボットネット攻撃のような新たな攻撃手法が発見されたことは記憶に新しい．国境を超えてサイバー攻撃やマルウェア感染等が頻繁に生じるセキュリティの世界では，グローバルに情報共有するため一定の仕組みが作られつつある．本稿では，その情報共有の仕組みが与えられているとの前提をおく．そのうえで，外部のシステムでサイバー攻撃やマルウェア感染等が発生してその情報が共有されたときに，自組織のシステムでは，そのサイバー攻撃なりマルウェア感染を防げるようになっているのか，あるいは対策が不十分で損害を被ってしまうようになっているのかを分析，評価する方法について述べる．既に，分野別にサイバー情報を共有する仕組みとして ISAC があり，ICT，エネルギー，金融のそれぞれについては長年の ISAC 運用の歴史がある．また，より汎用的に情報処理面に着目してサイバー攻撃の発生時に情報共有を行う仕組みとして，CSIRT がある．例えば，EU の NIS 指令では，エネルギー，輸送，水道供給，銀行，金融市場インフラ，医療，デジタルインフラ及びデジタルサービスプロバイダ（検索エンジン，クラウドコンピューティングサービス及びオンライン市場）のような，経済及び社会にとって重要な部門を横断するセキュリティ措置及びインシデントに関して ISAC なり CSIRT による情報共有を義務化している．また，米国でも，既に，VEP や E-ISAC，ICT-CERT のような一定の情報共有の仕組みが制度化されている．日本の各産業界も脆弱性データベース共有の必要性を認識しており，相当の努力をしている．現在，脅威分析を行う際の基となる外部の脆弱性データベースは，CWE，CVE 等を利用することを前提としている．いずれも，米国のサイバー・セキュリティ機関が世界的に広く提供しているものである．日本の IPA と JPCERT/CC が共同運営する脆弱性情報サイト JVN が，CVE と互換認

[1] 湘南工科大学, 神奈川県藤沢市辻堂西海岸 1-1-25
Shonan Institute of Technology, 1-1-25 Tsujido-Nishikaigan, Fujisawa, Kanagawa 251-8511, Japan
[2] 国立研究開発法人産業技術総合研究所, 東京都江東区青海 2-4-7
National Institute of Advanced Industrial Science and Technology (AIST), 2-4-7 Aomi, Koto-ku, Tokyo 135-0064, Japan

定されている．CVE 等はソフトウェアの具体的な脆弱性のデータベースで，ソフトウェアの脆弱性が発見されたら登録される．これには，机上で見つけた脆弱性であって，それによるインシデントが実システムでは顕在化していないものも含まれる．各脆弱性情報は概要を自然言語で記述した文章とソフトウェアコードである．本提案では，この脆弱性情報が与えられたときに，自組織のシステムのなかに脆弱性を生むソフトウェアがある場合，それを発見し，修正するための基本方法を開発することを目的とする．この目的に対し，国立研究開発法人産業技術総合研究所が開発したセキュリティ要件分析ツール (TACT)[1] を用いてトピックモデルによる文書間の類似度を算出することで自然言語で書かれた文書を突合分析し，類似度の高い脆弱性を発見する研究 [2][3] が行われている．そこでは，一つの自動車の設計情報や脆弱性特性に関して自然言語で記述した文書と，自動車に限らず広く脆弱性について自然言語で記述した文書との間でトピックモデルによる 2 文書間突合分析が行われた．その結果，その自動車にマッチする新たな脆弱性を導出し得るという有意な効果を得ている．ただし，後者の脆弱性のデータベース空間を増減する効果については，十分な分析が行われていなかった．そこで我々は Python のトピックモデル用のライブラリを使用し，自然言語記述の類似度に基づく脆弱性分析のツール（以下提案ツールとする）を作成した．提案ツールを用いて，大規模脆弱性データベース（以下大規模脆弱性 DB とする）と論文 [3] で使われているテスラ社への自動車への攻撃の論文 [4] の BROWSER HACKING に関する節をマッチング処理し，類似度を算出する．その後，大規模脆弱性 DB から一部を抜粋したデータと全データでの該当した脆弱性情報から差を比較することで提案ツールの有効性を示す研究 [5] を行っている．しかしながら単一の比較対象で検証しただけでは有効性に欠ける．そのため本提案では新たに比較用の文書として，論文 [4] の LOCAL PRIVILEGE ESCALATION の節で既存研究 [5] と同様の手順で結果を検証する．本提案では，2 章は本提案に関連する技術や使用するツールなど事前の説明を記述する．3 章では本提案の手法の紹介，4 章では提案手法に則って文書を分類した際の結果を記述する．5 章では結果からの考察を行い，6 章にてまとめを示す．7 章で本提案により生じた課題を記述する．

2. 関連研究

2.1 大規模脆弱性 DB

米国 MITRE 社は CVE(Common Vulnerability and Exposure：共通脆弱性識別子)[6] を公開している．従来から脆弱性情報に関するデータベースはあったが，MITRE 社により 1999 年に脆弱性情報を一意に特定できるよう CVE が提案された．現在 CERT/CC や IBM など多くの主要な脆弱性情報サイトと連携し，多くの脆弱性情報を統合して提供している．公式サイトからは csv，html，text，xml の 4 種のフォーマットで，Unix compressed，Gzipped，Raw のデータ圧縮フォーマットを入手することができる．本提案では csv 形式のデータを用いる．データ構成は Name，Status，Description，Reference，Phase，Votes，Comments からなる．このうち Name と Description を抽出し用いる．Name には一意な ID が含まれ，Description には脆弱性に関する自然言語 (英語) で記述された説明文が入っている．

2.2 LDA(Latent Dirichlet Allocation)

文書には潜在的なトピックがあると仮定し，文書内の単語はそのトピックから生起されその単語集合が文書であるという考え方がある．この文書の集合から潜在的なトピックを推定しようというものがトピックモデルと呼ばれる．トピックモデルの中に潜在的意味解析と呼ばれるものがある．これを発展させたものに LDA がある．潜在的意味解析から LDA の手法が提案されるまでには以下の流れがある．まず基礎として LSA(Latent Semantic Analysis) というものがある．これが日本語で潜在的意味解析と呼ばれるものである．これは文書を単語の出現回数でベクトル化したものを，一般的には TF-IDF により重み付けする（TF-IDF の詳細は 2.3 に記す）．重みづけしてあるベクトル群は膨大なため，特異値分解により近い意味を持つ者同士を近似することで，ベクトル空間を削減する．つまり単語から潜在的トピックに変換していることに相当する．このできたベクトル群を用いて，文書のクラスタリングや文書分類をしていくものが LSA である．しかし LSA はトピックに意味づけが数学的にはされるが，文の意味や単語の意味に関連した意味づけはされないという課題がある．LSA に確率の概念を追加したものに pLSA(Probabilistic LSA) というものがある．これは文書はある確率モデルに基づいて生成されるという考え方であり，pLSA は各文書における各トピックの重要度の割合と各トピックにおける各単語の重要度の割合を仮定する．この仮定されたモデルのパラメータを一般的に EM アルゴリズムと呼ばれる手法を用いて更新していき，最も尤度の高い確率モデルを生成する．pLSA の利点として値がすべて確率なので 0 から 1 で出力され，各単語が複数のトピックに属することができる．pLSA の理解を簡単にするためグラフィカルモデルを図 1 に示す．使われる記号の説明を表 1 に記す．pLSA では M 個の文書の中から各文書 d について考え，文書中の潜在的なトピック分布 c を推定する．推定される c からトピックに含まれる単語 w を割り当てる．この単語へ割り当てを N 単語分，M 個の文書分繰り返すことでモデルを作成している． LDA は David M. Blei 氏の論文 [7] によって提案された．LDA ではパラメータの値を 1 つに決定するのではなく，事前分布を用意し，パラメータの事後分布

Vol.2020-CSEC-90 No.16
Vol.2020-SPT-38 No.16
2020/7/20

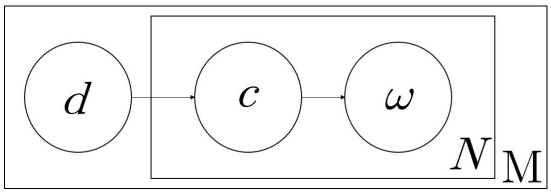

図 1 pLSA のグラフィカルモデル

表 1 pLSA のグラフィカルモデルにおける記号表

記号	説明
M	文書数
N	各文書の単語数
d	各文書
c	トピック分布
w	トピックごとの単語分布

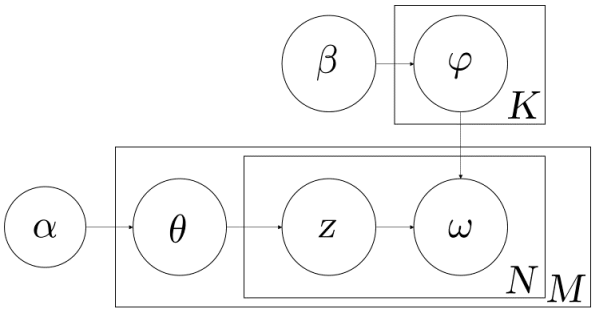

図 2 LDA のグラフィカルモデル

表 2 LDA のグラフィカルモデルにおける記号表

記号	説明
M	文書数
N	各文書の単語数
α	各文書ごとのトピック分布の事前分布のハイパーパラメータ
β	トピックごとの単語分布の事前分布のハイパーパラメータ
θ	各文書ごとのトピック分布
z	単語ごとのトピック分布
w	トピックごとの単語分布
φ	トピックごとの単語分布
K	トピック数

を推定する．このときに事前分布には主にディリクレ分布を，事後分布には多項分布を一般的に仮定する．LDA のグラフィカルモデルも pLSA と同様にイメージを容易にするため，図 2 に示す．LDA のグラフィカルモデルに使われる記号説明は表 2 に示す．LDA のモデルはトピック分布 θ がハイパーパラメータである事前分布 α から生成され，生成された θ から文書内の各単語のトピック分布 z が生成される．もう 1 つのハイパーパラメータである β からはあるトピックの語彙の出現確率が生成されてる．ここまでの z と φ により w が生起される．pLSA と LDA の違いは，LDA では各文書とトピックごとの単語分布に対してそれぞれディリクレ分布をハイパーパラメータとし与えていることにある．結果的に LDA は pLSA に比べ汎化性能が高くなることが多く，様々なものに活用することが可能である．LDA ではパラメータを解析的に求められない際のために，サンプリング手法の 1 つであるギブスサンプリングが採用されている．

2.3 TF-IDF

TF-IDF は TF と IDF という二つの概念を組み合わせたものである．TF は Term Frequency の略であり，単語の出現頻度を表すものである．以下この節で登場する記号は表 3 に示す．TF は式 (1) によって導出できる．

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{1}$$

IDF は Inverse Document Frequency の略であり，こちらはある単語の含まれる文書の割合の逆数を表す．式 (2) のように導出する．

$$idf_i = \log \frac{|D|}{|\{d : d \ni t_i\}|} \tag{2}$$

導出された TF と IDF を乗算することで TF-IDF を得ることができる．以下に簡単な例を示す．次の表 4 に新すような，単語と出現数からなる各文章があるとする．各文書における単語の出現数を P_1，P_2・・・P_n と表現する．また各文書における各単語の TF の値を T_1，T_2・・・T_n と表現し，TF-IDF の値を TI_1，TI_2・・・TI_n と表現するものとする． 表 4 のような文書群に対し TF の値は表 5 のようになる．例えば文書 1 の車の TF 値は tf(車，文書 1) $= \frac{10}{34} = 0.294... \fallingdotseq 0.29$ と計算される．次に IDF の値を計算すると表 6 となる．例えば車の IDF 値は idf(車) $= \log_2 \frac{5}{4} = 0.321... \fallingdotseq 0.32$ と計算される．導出した TF と IDF を乗算することで表 7 を得ることができる．例えば文書 1 の車の TF-IDF 値は $tfidf$(車，文書 1) $= 0.29 \times 0.32 = 0.0928 \fallingdotseq 0.09$ と計算される．

表 3 式 (1),(2) の記号

記号	説明
i, j, k	添え字
$n_{i,j}$	文書 d_i における t_i の出現回数
$n_{k,j}$	文書 d_i における単語の出現回数
$\|D\|$	総文書数
t_i	i 番目の単語
$\|\{d : d \ni t_i\}\|$	単語 t_i を含む文書数

表 4　各文書における各単語の出現頻度

単語	P_1	P_2	P_3	P_4	P_5
車	10	14	6	0	20
バイク	8	5	11	2	14
電車	16	0	0	12	9
船	0	0	12	10	0
計	34	19	29	24	43

表 5　各文書における各単語の TF の値

単語	T_1	T_2	T_3	T_4	T_5
車	0.29	0.74	0.21	0.00	0.47
バイク	0.24	0.26	0.38	0.08	0.33
電車	0.47	0.00	0.00	0.50	0.21
船	0.00	0.00	0.41	0.42	0.00

表 6　各単語の IDF の値

単語	IDF 値
車	0.32
バイク	0.00
電車	0.74
船	1.32

表 7　各文書における各単語の TF-IDF の値

単語	TI_1	TI_2	TI_3	TI_4	TI_5
車	0.09	0.24	0.07	0.00	0.15
バイク	0.00	0.00	0.00	0.00	0.00
電車	0.35	0.00	0.00	0.37	0.16
船	0.00	0.00	0.54	0.55	0.00

2.4　評価指標

トピックモデルの評価指標は数多く提案されている．その中に Perplexity と Coherence がある．Perplexity とは分岐数または選択肢の数を表し，確率の逆数で表される．つまりどれほどの選択肢の中から単語が選択されているかを表しており，予測性能と考えることができる．一般的に値は低いほど良いとされる．Coherence はトピックの品質を表す指標として使われ，人間にとって解釈がしやすいかどうかを表す．しかし Coherence に対する定義は明確にされていないのであるのが現状である．Coherence について最初に人間による評価方法 [8] が提案され，その後自動評価方法 [9] が発表されている．一般的に Coherence は高いほど良いとされる．

3. LDA モデルの作成と文書間類似度の算出方法

3.1　提案の概要

本提案では Windows 10 上に Oracle VM VirtualBox を用いた Ubuntu18.04LTS の仮想 OS で Pyrhon 3.6.9 の環境で提案の検証を行った．データは CVE の 1999 年から 2019 年 7 月 25 日までのデータから不確実な情報や重複，統合などの情報を除去した 119479 件のデータを使用した．(以下 CVE データとする) 本提案で用いる LOCAL PRIVILEGE ESCALATION に関する節で主として使われた脆弱性は CVE-2013-6282 である．BROWSER HACKING に関する節で主として使われた脆弱性は CVE-2011-3928 である．CVE データを用から LDA モデルの作成し，作成したモデルを使い 2 つの文書にもトピック分布を付与する．その後文書間の類似度を算出し，文書を分類する．分類後抽出されたデータ件数を検証し，比較したい文書ごとの精度を考察する．

3.2　比較用データの作成

本提案では論文 [4] の LOCAL PRIVILEGE ESCALATION に関する節と BROWSER HACKING に関する節を比較用文書として用いる．LOCAL PRIVILEGE ESCALATION に関する節から直接的な表現である”CVE-2013-6282”と”Tesla”は文章から除去した．また説明のための余分なガイド文も除去を行った．図 3 が本提案で用いる加工後の LOCAL PRIVILEGE ESCALATION の節の内容である．以後本提案中では加工後の LOCAL PRIVILEGE ESCALATION の節を，頭文字をとり LPE 文書と呼ぶ．図 4 は BROWSER HACKING に関する節の加工後の文書である．元の文章から直接の表現となる CVE-2011-3928 のワードを消去してある．さらに BROWSER HACKING の節内ではブラウザバージョンが CVE と表記が異なるた

It seems that the Linux kernel version of CID is very old, there is nearly no exploiting mitigations on Linux kernel 2.6.36. we can get the arbitrary read/write in kernel context, it is pretty easy to write an exploit. In our exploit, firstly we patched setresuid() syscall to get the root privilege, and then we invoked reset_security_ops() to disable AppArmor. It’s obviously that we’re now in god mode.

図 3　LPE 文書の内容

Since the User Agent of Tesla web browser is ”Mozilla/5.0 (X11; Linux) AppleWebKit/534.34 Google Chrome (KHTML, like Gecko) QtCarBrowser Safari/534.34”, it can be deduced that the version of QtWebkit is around 2.2.x. In such old version, there are many vulnerabilities in QtWebkit.

[70 行省略]

7. Get the address of the JIT memory from JSCell address and JSC::ExecutableBase structure. 8. Write shellcode to JIT memory and execute this JavaScript function. We must say it is difficult to develop a feasible and stable exploit without any debugging method and without QtCarBrowser binary from Tesla CID. However, it was deserved as the final exploit gave us the first shell from Tesla CID and the shell is very stable.

図 4　BH 文書の内容

め，”Google Chrome”を単語として差し替えてある．こちらの文書は以後本提案中では BROWSER HACKING に関する節を，頭文字をとり BH 文書と呼ぶ．

3.3 Gensim による LDA モデルの作成

本提案では Python のライブラリである Gensim を用いて LDA が実装された提案ツールを使用する．Gensim は LGPL のライセンスで提供され，教師なしトピックモデリングと自然言語のためのオープンソースライブラリである．提案ツールは既存研究 [10] で公開されているソースコードを利用し，データ読み込み部やストップワードと呼ばれる文書の助詞や副詞などを除く処理等を適宜追加している．LDA 生成プロセスの詳細は以下のとおりである．

(1) CVE の Description の行を配列データとして読み込む
(2) 全ての単語に対して Python のライブラリである nltk.corpus の wordnet クラスの morphy メソッドを用いて単語を原形に戻す
(3) 読み込んだ文書群からストップワードを nltk.corpus の stopwords クラスの words メソッドで english を指定し，文書から自動で取り除く
(4) 残った文書データから任意の出現回数以下（本研究では 2 回以下）の単語を除去する
(5) (3) までの処理を終えた文書群から corpora クラスの Dictionary メソッドにより単語辞書を作成する
(6) 作成した辞書と文書群から各文書を dictionary クラスの doc2bow メソッドにより単語のベクトルに変換する（この処理をコーパスの作成と呼ぶ）
(7) Gensim の ldamodel クラスの LdaModel メソッドに作成した辞書とコーパス，パラメータを指定して LDA モデルを作成する

パラメータの 1 つに，1 文書当たりの持ち得るトピック数の項目がある．このトピック数の決定は 3.3 に記述する．上記の手順で作成するコーパスは，文書を単語の出現回数で表した BoW(Bag of Words) と呼ばれる形式のベクトル表現である．本提案では BoW ではなく 2 章で述べた，TF-IDF で重みづけを行ったコーパスだけを用いる．TF-IDF を適用したコーパス（以下 TF-IDF コーパスとする）の作成手順は以下の通りである．

(I) 前述の (1) から (4) までの処理を行う
(II) Gensim の models クラスにある TfidfModel メソッドにより TfidfModel を作成する
(III) (6) で作成したコーパス（BoW コーパス）を TfidfModel に渡すことで TF-IDF コーパスを得る

以上により生成されるコーパスを用いて実験を行う．

3.4 トピック数の決定

前節までの手順により LDA モデルは作成可能であるが，

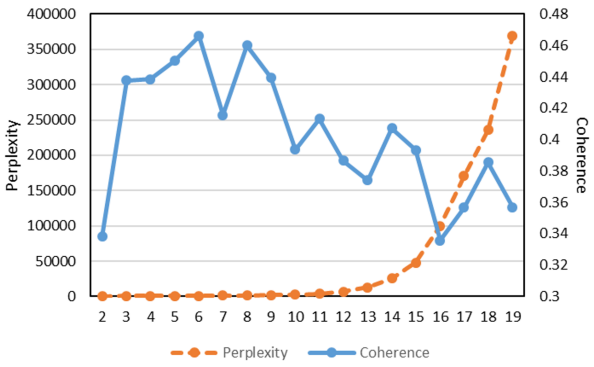

図 5 Perplexity と Coherence のグラフ (TF-IDF コーパス)

文書に存在するトピック数を決定する必要がある．今回は実験的にトピック数を求めた．図 5 は全ての CVE データを用いて，出現頻度 2 以下の単語を取り除く設定にしたときの LDA モデルの評価を行ったグラフである．LDA モデルはは CVE のデータを元に生成されるため, LPE 文書と BH 文書において共通である．図 5 より，トピック数 6 の時点で Perplexity が低く Coherence が高い状態で差が最大である．よって数値的に見た場合，グラフ内のトピック数毎の LDA モデルとしてはトピック数 6 が良好であると言える．提案 [5] ではこれらの評価結果より，良好だと思われるトピック数を LDA モデルの作成の際（生成プロセスの (7) のパラメータに該当）に用いた．本提案では Perplexity と Coherence の差が大きいトピック数についていくつか結果をとったところ，トピック数 11 が良好だった．そのため LPE 文書ではトピック数 11 の結果を採用した．BH 文書は論文 [5] を踏まえ最良と思われる，トピック数 6 ではなくトピック数 7 を採用した．

3.5 攻撃事例へのトピック分布の付与

比較用文書にもトピック分布を割り当てる必要がある，LDA モデルは 3.2 に示した CVE によるモデルを利用できる．LDA モデルを作成後の流れは以下のとおりである．

(i) 比較したい文書を読み込む（本提案では LPE 文書と BH 文書）
(ii) (1)〜(4) までの処理を行う
(iii) Gensim の dictionary クラスの doc2bow メソッドを使い比較したい文書を単語のベクトルにする
(iv) LdaModel クラスの get_document_topics メソッドにベクトル化した文書を引数として与え確率分布を割り当てる

3.6 文書間の類似度の算出

LDA モデルの作成と比較したい文書へのトピック分布の割り当て終了後，CVE の脆弱性情報と比較したい文書の文書間でコサイン類似度を求める．コサイン類似度は各文書のベクトルの内積によって求められる．

表 8　各データの該当件数

ツール名	使用データ	トピック数	総件数（件）	該当件数（件）	該当割合（ %）
従来ツール	小規模データ	不明	200	29	14.5
提案ツール（BH 文書）	大規模データ	7	119479	1514	1.3
提案ツール（LPE 文書）	大規模データ	11	119479	5180	4.3

3.7　脆弱性情報の抽出

CVE-2013-6282 は今回用いている LPE 文書における主として用いられた脆弱性であり，本提案では必ず検出されなければならない脆弱性である．よって CVE-2013-6282 と LPE 文書とのコサイン類似度を閾値として設定する．設定した閾値以上である脆弱性数をカウントする．BH 文書では主として用いられた脆弱性は CVE-2011-3928 であるため，CVE-2011-3928 と LPE 文書とのコサイン類似度を閾値として設定する．

4.　評価

4.1　評価結果

評価結果を表 8 に示す．従来ツールとは，従来研究 [3] で使用されたツールである．大規模データとは CVE の全データ 119479 件全てを学習させた LDA モデルを指す物である．小規模データとは従来ツール用に大規模データから CVE-2011-3928 を中心として，前方から 49 件，後方から 50 件を抽出したデータである．従来ツールを用いた場合は小規模データ 200 件に対して，BH 文書と類似していると判定された脆弱性は 29 件（全体割合 14.9 %）であった．提案ツールでは TF-IDF コーパスを用いて，BH 文書と大規模データを用いた場合は 119479 件中 1514 件（全体割合 1.3%）であった．次に提案ツールを用いて LPE 文書と大規模データを突合処理した場合，119479 件中 5180 件（全体割合 4.3%）であった．パーセンテージとしては近いと考えられなくもないが件数的には 3666 件増加しており無視できるかは難しい件数であった

5.　考察

既存研究 [5] では CVE の Description が短文なのが精度に影響すると考えていたが，提案中の検証実験から比較したい文書が短文であればこれも影響することがわかる．そのため両データは 1 つ 1 つがある程度長いものであることが望ましいと考えられる．

6.　まとめと今後の課題

BH 文書と突合した結果と比較し，LPE 文書と突合した結果の精度は悪くなった．しかしこれは BH 文書と比べて LPE 文書の文章が短いことが要因の 1 つであると考えられる．2 回の提案を通しこれ以上，精度を向上させるにはデータの増量や新たな重みづけの手法の導入などが必要だということが考えられる．今後は精度向上のため，Okapi BM25 のような TF-IDF とは異なる手法の重みづけや文書の正規化といった前処理の追加を行っていきたい．現状の結果からクラスター分析などにより今得られている結果から攻撃の傾向を抽出といった別方向の結果を得る手法を取り入れていきたい．具体的には脆弱性の系統ごとに分類されている米 MITRE 社の Common Weakness Enumeration（CWE）を活用したり，突合処理後のデータをクラスター分析することで脆弱性の傾向の提示などに広げていくことができる．

参考文献

[1] 半田剣一, 大崎人士, 竹内泉, (2017) ”セキュリティ要件分析支援ツール TACT.: 情報処理学会ソフトウェア工学研究会ウィンターワークショップ 2017・イン・飛騨高山予稿集 (2017)
[2] 梅澤克之，三科雄介，田口研治，寶木和夫.: 脆弱性データベースを使用した脅威分析方法の提案, 暗号と情報セキュリティシンポジウム (SCIS2018) 予稿集, 1C2-6, Jan. 2018.
[3] 梅澤克之, 三科雄介, Sven Wohlgemuth, 寶木和夫.: 脆弱性データベースを使用した脅威分析方法, 暗号と情報セキュリティシンポジウム (SCIS2019) 2F3-3, Jan. 2019.
[4] S. Nie et al.: FREE-FALL: HACKING TESLA FROM WIRELESS TO CAN BUS, Briefing,Black Hat USA 2017, July 2017.
[5] 小柳洋貴, 梅澤克之, 三科雄介, 寶木和夫.: 脆弱性データベースを使用した脅威分析. 研究報告コンピュータセキュリティ (CSEC). 2020, 2020-CSEC-88(38), p. 1-6.
[6] MITRE Corporation: CVE - Common Vulnerability and Exposure, 入手先 ⟨https://cve.mitre.org/⟩ (2020.01.15).
[7] D. Blei, A. Ng, and M. Jordan.: Latent Dirichlet Allocation, in Journal of Machine Learning Research(2003), pp. 1107-1135.
[8] Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L Boyd-Graber,and David M Blei.: Reading tea leaves: How humans interpret topicmodels, In Advances in NIPS, pp. 288-296, 2009.
[9] Newman, D., Lau, J. H., Grieser, K. and Baldwin, T.: Automatic Evaluation of Topic Coherence, pp. 100-108 (2010).
[10] Latent Dirichlet Allocation (LDA) ゆるふわ入門 - あらびき日記, 入手先 ⟨https://abicky.net/2013/03/12/230747⟩ (2020-01-15).
[11] Teh, Y. W.; Jordan, M. I.; Beal, M. J.; Blei, D. M. (2006).: Hierarchical Dirichlet Processes, Journal of the American Statistical Association. 101 (476), pp. 1566-1581.