Computers, Materials & Continua
DOI:10.32604/cmc.2021.012220
Article

Tech Science Press

An Automated Penetration Semantic Knowledge Mining Algorithm Based on Bayesian Inference

Yichao Zang[1,*], Tairan Hu[2], Tianyang Zhou[2] and Wanjiang Deng[3]

[1]State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou, 450000, China
[2]National Engineering Technology Research Center of the Digital Switching System, Zhengzhou, 450000, China
[3]NUS Business School, National University of Singapore, Singapore, 119077, Singapore
[*]Corresponding Author: Yichao Zang. Email: zangyeechao@sina.com
Received: 20 June 2021; Accepted: 19 July 2020

Abstract: Mining penetration testing semantic knowledge hidden in vast amounts of raw penetration testing data is of vital importance for automated penetration testing. Associative rule mining, a data mining technique, has been studied and explored for a long time. However, few studies have focused on knowledge discovery in the penetration testing area. The experimental result reveals that the long-tail distribution of penetration testing data nullifies the effectiveness of associative rule mining algorithms that are based on frequent pattern. To address this problem, a Bayesian inference based penetration semantic knowledge mining algorithm is proposed. First, a directed bipartite graph model, a kind of Bayesian network, is constructed to formalize penetration testing data. Then, we adopt the maximum likelihood estimate method to optimize the model parameters and decompose a large Bayesian network into smaller networks based on conditional independence of variables for improved solution efficiency. Finally, irrelevant variable elimination is adopted to extract penetration semantic knowledge from the conditional probability distribution of the model. The experimental results show that the proposed method can discover penetration semantic knowledge from raw penetration testing data effectively and efficiently.

Keywords: Penetration semantic knowledge; automated penetration testing; Bayesian inference; cyber security

1 Introduction

With the increasing size of computer networks and complexity of information systems, security problems faced by corporations have become much more prominent than ever [1]. Penetration testing, as a common security check approach, is widely used in discovering attack paths existing in computer networks to improve the security level of systems. Penetration testing aims to obtain control over specific computers in a target network. Usually, it starts with a controlled computer, based on which hackers try to gather computer and network information through scanners and vulnerability exploitation. After gathering information, hackers choose the best vulnerability exploitation program to penetrate the target host. Even though penetration testing could discover attack paths hidden in network, the quality of

penetration testing depends heavily on the security expert's experience, and it has become a popular topic to automate penetration testing in both academic and engineering areas [2]. Mining semantic knowledge hidden in raw penetration testing data is an essential part of the solution for automated penetration testing. Penetration semantic knowledge is a kind of mapping relation {*software*: *version*→*vulnerability*} which means that the specific *version* of the *software* may cause the *vulnerability*, based on which we could choose the corresponding vulnerability exploitation program to control the host when faced with the specific version of the software. Existing studies have extracted penetration semantic knowledge by transforming specific vulnerability database such as metasploit and nessus. There are two disadvantages to this kind of approach: one is that the semantic knowledge extracted from a specific vulnerability database does not match the information gathered through scanners, for example, the operating system information gathered by nmap scanner is *windows_7_sp1* for vulnerability numbered cve–2019–0708 (common vulnerability and exposure, cve), but the operating system description of vulnerability numbered cve–2019–0708 in metasploit is *windows_7 sp1*. Even though they look similar (*windows_7_sp1* and *windows_7 sp1*), they do not match each other, missing the knowledge rule {*windows_7 sp1*→cve–2019–0708} during penetration testing. The other is that the penetration semantic knowledge becomes tedious without considering information gathered through multiple scanners. During penetration testing, multiple tools are used to gather information to form intact knowledge, meaning that the penetration semantic knowledge is composed of multiple sources, and transforming a specific vulnerability database can not meet the demand. Tab. 1 shows an example of host configuration and vulnerability information gathered by nmap and shodan scanners [3]. It indicates that the software *Apache*:*2.4*, *OpenSSH*:*7.4* and *CCMPlayer*:*1.5* are installed on the host, the operating system is either *windows xp sp3* or *windows 7*, and there is also a vulnerability numbered cve–2011–5170 on the host. The penetration knowledge hidden in the raw data is {*CCMPlayer*:*1.5*, *windows xp sp3*→cve–2011–5170} which means that when faced with the software *CCMPlayer*:*1.5* and the operating system *windows xp sp3*, the vulnerability exploitation program corresponding to cve–2011–5170 could be used to control the target host. In summary, the aim of penetration semantic knowledge mining is to discover all of these reliable mapping relations existing in raw penetration testing data gathered by multiple scanners.

Table 1: Example of scanned configuration and vulnerability information for a single host through nmap and shodan scanners

Software: Version	OS	CVE
Apache:*2.4*	*windows xp sp3*	cve–2011–5170
OpenSSH:*7.4*	*windows 7*	/
CCMPlayer:*1.5*	/	/

Considering the problems with current penetration semantic knowledge mining methods, this paper applies Bayesian inference techniques to mine semantic knowledge from raw penetration testing data. The remainder of the paper is organized as follows. Section 2 presents related works involving associative rule mining algorithms that could be used to mine penetration semantic knowledge. Section 3 introduces some definitions, and the problem statement is given. Section 4 presents the proposed Bayesian inference based penetration semantic knowledge mining algorithm. In Section 5, the details of penetration testing data are analysed, and the performance of associative rule mining algorithms is compared to the performance of the proposed method on the same data. Section 6 summarizes the study and notes some future research objectives.

2 Related Works

Associative rule mining [4], which was first proposed by Argawal et al., aims to discover itemsets that appeared frequently, and this approach is similar to penetration semantic knowledge mining. The Apriori algorithm [5] was the first algorithm used to mine associative rules by generating candidate frequent itemsets. However, this method has high computational complexity because of the large number of database scan operations required. Han et al proposed the FP-Growth [6] algorithm to address this problem, in FP-Growth, an FP-tree structure is created to avoid database scan operations. However, the FP-Growth algorithm is limited to small-scale databases because of the extremely high memory consumption. Qin et al. [7] proposed compact FP-tree to mine associative rules to avoid conditional FP-tree generation to decrease memory consumption. Zaki proposed the Eclat [8] algorithm, which can mine associative rules by applying set union operations based on a vertical database structure. Pei proposed another memory-based associate rule mining algorithm, H-mine [9], based on an H-struct structure. H-mine achieves frequent pattern mining by partitioning data so that it can mine scalable databases. Deng proposed a data structure called N-list for a cache database, and the PrePost [10] algorithm was proposed based on N-list. Deng also adopted children–parent equivalence to prune search branches in PrePost, which effectively reduced the number of candidate frequent itemsets [11]. Because the N-list structure requires high memory consumption for both pre-traversal and post-traversal information, Deng proposed the NodeSets structure [12] to store pre-traversal information to mine associative rules; this approach reduced memory consumption by 50%. Aryabarzan et al. proposed another prefix tree structure, NegNodeSet [13], and adopted bitmap technology to mine associative rules. In addition, some researchers adopted high utility itemset mining techniques to achieve associative rule mining. The two-phase algorithm [14] is a famous and classical candidate based high utility itemset mining algorithm that is composed of two phase: the first phase prunes the search space and generates candidates by the proposed transaction weight downward closure property, and the second phase scans the database to filter high utility itemset from high transaction weight utility itemsets identified in phase I. Vincent et al. proposed an algorithm named UP-Growth [15] to mine high utility itemset, this algorithm can construct a utility pattern tree from databases based on DGU(discarding global unpromising items), DGN(discarding global node utility), DLU(discarding local unpromising items) and DLN(discarding local node utility) strategies to prune the search spaces. Even though many tricks have been proposed to prune search spaces, there are still many candidate itemsets waiting to be tested in phase II, which will consume a large amount of memory and time. To overcome these problems, Liu et al. proposed an algorithm called HUI–Miner [16] to mine high utility itemsets without generating candidates. HUI–Miner uses a novel structure called utility-list to store both utility information and heuristic information of the itemset for pruning the search space. Based on the utility-list, the high utility itemsets can be mined by joining utility lists instead of by scanning the database, which decreases the mining time. Further, Krishnamoorthy et al. [17] proposed an algorithm called HUP-Miner that employs two novel pruning strategies to prune search spaces. Peng et al. utilized IHUP tree structure [18] to guide the itemset expansion process to avoid considering itemsets that are nonexistent in the database. Although associative rule mining algorithms can be used to mine penetration semantic knowledge from penetration testing data, the effect is not ideal. Notably, some host configuration and vulnerability information does not meet the requirements for frequent patterns.

3 Definitions and Problem Statements

3.1 Definitions

Definition 1. Given itemset $I = \{i_1, i_2, \cdots, i_n\}$ and database $DB = \{T_1, T_2, \cdots, T_m\}$ where $T_i, 1 \leq i \leq m$ is a transaction that satisfies $T_i \subseteq I$, for any itemset $A \subseteq I$, A is contained by transaction T_i when $A \subseteq T_i$ holds. The support of A is the number of transactions containing A, denoted as $\sup(A)$. A is called a frequent itemset if $\sup(A) \geq \xi$ for any specified minimal support ξ.

Definition 2. A Bayesian network is a kind of probabilistic graphical model, denoted as $G = \{V, E\}$, where V is a set of random variables and E is a set of dependence relationships among variables, denoted

as $p(v|\pi(v))$; $\pi(v)$ is set of parent nodes of v. A Bayesian network is used to represent cause-effect relationships, denoted as $<\pi(v),\ v> \in E$; the specific value for $\pi(v)$ is the cause, and v is the effect. Thus, $<\pi(v)=a, v=b>$ has a strong causal effect if $p(<\pi(v)=a, v=b>) \geq \alpha$ holds.

Definition 3. d-separation can be defined as follows: Given a Bayesian network $G=\{V,E\}$ and a set $X,Y,Z \subseteq V$, where each pair is disjoint, for each path between X and Y, Z d-separates X and Y under the following three conditions: (1) There is one node a that belongs to Z in the path, and the two neighbours b and c, which are in sets X and Y, respectively, follow the same edge direction $<b,a><a,c>$ or $<a,b><c,a>$. (2) There is one node a that belongs to Z in the path, and the two neighbours b and c are in sets X and Y, respectively, with the inverse edge direction $<a,b><a,c>$. (3) There is one node a in the path, the node and its descendants do not belong to Z, the two neighbors b and c are in sets X and Y, respectively, and the edge direction satisfies $<b,a><c,a>$.

Definition 4. A directed bipartite graph $G=\{V,E\}$ is a kind of special Bayesian network for which V can be divided into two disjoint sets A and B. For each edge $<v_m,v_n> \in E$ in the graph, v_m belongs to A, and v_n belongs to B.

3.2 Problem Statement

Penetration semantic knowledge mining aims to discover associative rules from vast amounts of penetration testing data, including rules for open ports, services, operating systems, etc. There are two problems with penetration semantic knowledge mining: One is that there is much more irrelevant penetration semantic knowledge than critical knowledge data, and the other is that there are many invalid rules that can not be used to improve the efficiency of automated penetration testing. Therefore, penetration semantic knowledge mining aims to discover rare penetration knowledge while avoiding redundant and low-value penetration testing data.

4 Penetration Semantic Knowledge Mining

4.1 Model

Considering the above problem, we adopted a Bayesian network to formalize penetration testing data, and the maximum likelihood estimation method was adopted to retrieve parameters. Finally, we adopted a variable elimination method to retrieve the probability of each mined rule and discover all penetration testing knowledge of interest. The steps are described as follows.

Step 1: Given host configuration information $A=\cup_{i=1}^{k}A_i$ and vulnerability information $B=\cup_{i=1}^{k}B_i$ where $i=1,\cdots,k$ is the host identifier, the Bayesian network is composed of union set $A \cup B$, where A is set of conditional nodes with host configuration information for the operating system, application, etc., and B is a set of deductive nodes of host vulnerabilities. The Bayesian network is constructed by adding all directed edges $<v_m,v_n>$, where $v_m \in A_i$ and $v_n \in B_i$. Because there are only edges from the conditional node to the deductive node, the constructed Bayesian network is actually a directed bipartite graph, as shown in left panel of Fig. 1.

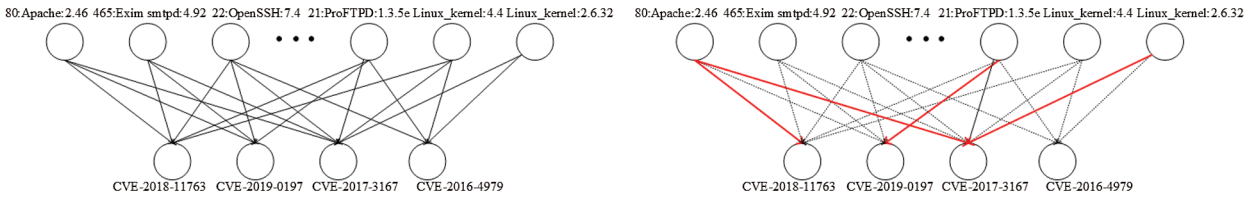

Figure 1: Bayesian inference based penetration knowledge discovery. The left panel shows a Bayesian network model based on original penetration data and the right panel shows penetration knowledge inferred based on a Bayesian network model constructed with the method below

Step 2: The parameters of the directed bipartite graph can be described with a conditional probability distribution $p(v|\pi(v))$, where $\pi(v)$, the set of conditional nodes, is the parent set of deductive node v. If $\pi(v)$ is empty, then the conditional probability distribution reduces to the initial probability distribution. Finally, the parameters of the directed bipartite graph can be described as set θ, where each element in θ is represented as

$$\theta_{ab} = \{p(v = b|\pi(v) = a)|\pi(v) \subseteq A, v \in B\} \tag{1}$$

which is the probability of vulnerability b conditioned on a combination of host configurations a. Concretely, this value is the probability of vulnerability cve–2017–0472 occurring when *windows 10* and the *IE 10* browser are installed on a host. Given the constructed Bayesian network, the parameter θ_{ab} is expressed as follows by optimizing the maximum likelihood function.

$$L(\theta|D) = \log \prod_{l=1}^{d} p(D_l|\theta) = \sum_{l=1}^{d} \log p(D_l|\theta) \tag{2}$$

where $p(D_l|\theta)$ is the probability of record D_l based on θ.

Step 3: Given the maximum likelihood function $L(\theta|D)$, we can optimize the objective function to retrieve parameter θ_{ab}, based on which we can obtain the probability for each expression $<\pi'(v) = a, v = b>$ by simplifying irrelevant variables, where $\pi'(v) \subseteq \pi(v)$. If the probability of $<\pi'(v) = a, v = b>$ is larger than the user–specified threshold α, the penetration testing rule of interest is established; otherwise, it is not. Specifically, as shown in right panel of Fig. 1, the host is influenced by vulnerability b with high probability when the combined host configuration is a. The algorithm for constructing the directed bipartite graph is shown in Alg. 1.

Algorithm 1. Directed bipartite graph construction algorithm

Input: penetration testing database $D = \{D_l|l = 1 \cdots N\}$;

Output: directed bipartite graph $G(V, E)$;

```
Initialize A,B,V, E =ϕ
for l in [1:N]
    for item in D_l  /*iterating whole penetration testing database*/
        tmp_A, tmp_B =ϕ
        if is_vulner(item):/*identify node type*/
            tmp_B.add(item)
        else tmp_A.add(item)
                V.union(tmp_A); V.union(tmp_B); /*add vertex to graph*/
                for j in tmp_B:
                    for i in tmp_A:
                E.add(<i, j>) /*add edge to graph*/
                end for
    end for
end for
return (V, E)
```

4.2 Penetration Semantic Knowledge Mining Algorithm

In this section, we introduce the use of penetration testing data to calculate the parameters of the Bayesian network and improve the parameter solution efficiency through *d*–separation. However, we first introduce the following theorem.

Theorem 1: Given a directed bipartite graph $G(V, E)$, where $V = A \cup B$, A is set of conditional nodes, and B is a set of deductive nodes. $P(a, b|M) = P(a|M)P(b|M)$ holds for every two nodes $a, b \in B$ and set of conditional nodes $M \subseteq A$.

Proof. First, we prove that $P(a, b|m) = P(a|m)P(b|m)$ holds for each conditional node $m \in M$. Because the Bayesian network G is a directed bipartite graph, there are three kinds of relations between m and a, b. (1) Both $< m, a >$ and $< m, b >$ belong to E. (2) One and only one of $< m, a >$ or $< m, b >$ belongs to E. (3) Neither $< m, a >$ or $< m, b >$ belongs to E. For (1), m is the branch node of a and b, and $P(a, b|m) = P(a|m)P(b|m)$ holds. For (2), assuming that $< m, a > \in E$, there is no connection between b and a, m; therefore, b is independent from a and m, so $P(a, b|m) = P(a|m)P(b) = P(a|m)P(b|m)$ holds. For (3), there are no connections among a, b and m, so these nodes are independent from each other, and we can conclude that $P(a, b|m) = P(a, b) = P(a)P(b) = P(a|m)P(b|m)$. For an arbitrary node m, $P(a, b|M) = P(a|M)P(b|M)$ holds.

Theorem 1 indicates that the parameter solution process for a large Bayesian network can be divided into many smaller processes to improve efficiency. By assuming that the Bayesian network constructed from penetration testing data $D = \{D_l | 1 \le l \le n\}$ is denoted as G (V, E), the maximum likelihood function $L(\theta|D)$ can be expressed as follows:

$$\begin{aligned} L(\theta|D) &= \sum_{l=1}^{d} \log p(D_l|\theta) \\ &= \sum_{l=1}^{d} \log \prod_{i=1}^{n} \prod_{j=1}^{J} \prod_{k=1}^{K} p(v_i = k|\pi(v_i) = j)^{\phi(i,j,k;D_l)} \\ &= \sum_{l=1}^{d} \sum_{i=1}^{n} \sum_{j=1}^{J} \sum_{k=1}^{K} \phi(i,j,k;D_l) \log p(v_i = k|\pi(v_i) = j) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{J} \sum_{k=1}^{K} \phi(i,j,k) \log p(v_i = k|\pi(v_i) = j) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{J} \sum_{k=1}^{K} \phi(i,j,k) \log \theta_{ijk} \end{aligned} \tag{3}$$

where $\phi(i,j,k)$ is the number of records that satisfy $\pi(v_i) = j$ and $v_i = k$. We can optimize the objective function and obtain the parameter as follows:

$$\theta_{ijk} = \frac{\phi(i,j,k)}{\sum_{k=1}^{K} \phi(i,j,k)} \tag{4}$$

representing the ratio of records for which the value of v_i is k, the value of the corresponding parent $\pi(v_i)$ is j, and all possible values of v_i are considered. Assuming that the obtained parameter set is denoted as $\theta = \{\theta_{1::}, \theta_{2::}, \cdots \theta_{n::}\}$ where $\theta_{i::} = p(v_i|\pi(v_i))$ is the conditional probability distribution of node v_i, we need to implement a variable elimination process [19] to retrieve key factors. Assuming that the sequence of variable elimination is denoted as $\rho, \rho \subseteq \pi(v_i)/\pi'(v_i)$, for each variable $z \in \rho$, the probability distribution of node v_i can be represented as $\theta_{i::} = p(v_i|\pi(v_i)) = \{p_1, p_2, \cdots, p_m\}$ according to the definition of $\theta_{i::}$, where m is the number of free variables. Let $g = \prod_{i=1}^{d} p_i$ be the function of variable z in the Bayesian network, d be the number of free variables containing z in the probability distribution set of

v_i, and $h = \sum_z g(z)$ be the probability distribution function after eliminating variable z and adding it back to $\theta_{i::}$. Then, ρ is iterated until it is an empty set to obtain all possible penetration semantic rules with probabilities larger than a user-specified support value α. The final penetration semantic knowledge mining algorithm is shown as follows.

Algorithm 2. Penetration semantic knowledge mining algorithm

Input: penetration testing database $D = \{D_l | l = 1 \cdots N\}$, support degree threshold α,variable elimination queue ρ;

Output: penetration semantic knowledge set

initialize rules = ϕ

G,A,B = CreateGraph(D);/*create directed bipartite graph */

for i in B:

  for j in $val(\pi(v_i))$:/*iterating parent set of vertex v_i*/

    for k in $val(B)$: /*iterating deductive node*/

      $\phi(i,j,k) \leftarrow count(D)$ /*count */

      $\phi(i,j) \leftarrow \sum_{k=1}^{K} \phi(i,j,k)$

    $\theta_{ijk} \leftarrow \phi(i,j,k) \Big/ \sum_{k=1}^{K} \phi(i,j,k)$/*parameter calculation*/

let $\theta_{i::} = p(v_i|\pi(v_i)) = \{p_1, p_2, \cdots, p_m\}$

for $z \in \rho$:/*variable elimination*/

  $g = \prod_{i=1}^{m} p_i$; $h = \sum_z g(z)$;

  $\theta_{i::}$.add(h)

for θ_{ijk} in $\theta_{i::}$:

  if $\theta_{ijk} \geq \alpha$/*identify knowledge rule*/

    rules.add ()

return rules

5 Experiments

In this section, we compare our proposed algorithm with other associative rule mining algorithms based on four real penetration testing datasets.

5.1 Metric

The experimental metric adopted is the ROC curve, which is used to evaluate the performance of algorithms. This curve is composed of two parts, the true positive ratio (TPR) and false positive ratio (FPR), and corresponding formulas are shown as follows:

$$TPR = \frac{|D \cap P|}{|P|} \times 100\%, \quad FPR = \frac{|D \cap N|}{|N|} \times 100\% \tag{5}$$

where P is the set of all true knowledge rules, N is the set of all false knowledge rules and D is the set of all mined knowledge rules.

5.2 Dataset

The experimental datasets encompass four services, namely, Apache, IIS, MySQL and nginx, containing host configuration information and vulnerability information. The process used to collect this information is described as follows. First, we collect host IPs containing these services through a zoomeye scanner [20]. Then, we collect host configuration information for each IP with an nmap scanner. Last, we use a shodan scanner to collect vulnerability information for each IP and merge this information with the host information collected above to form an intact penetration testing record for each IP. To describe the distribution characteristics, we plot the information in Fig. 2, from which we can see that the plot of the number of vulnerabilities has a long-tail. There are many vulnerabilities that have a few records. The number of records for most vulnerabilities is limited within 10, and the records for the vulnerability type account for approximately 17.5% of those of all vulnerabilities. The comparative algorithms are implemented based on SPMF [21] and include the Apriori, FP-Growth, LCMFreq and PrePost + algorithms. The support of the algorithm ranges from 0.2 to 0.8 and the host configuration is a Core–i7–8750H12 with 64GB of memory.

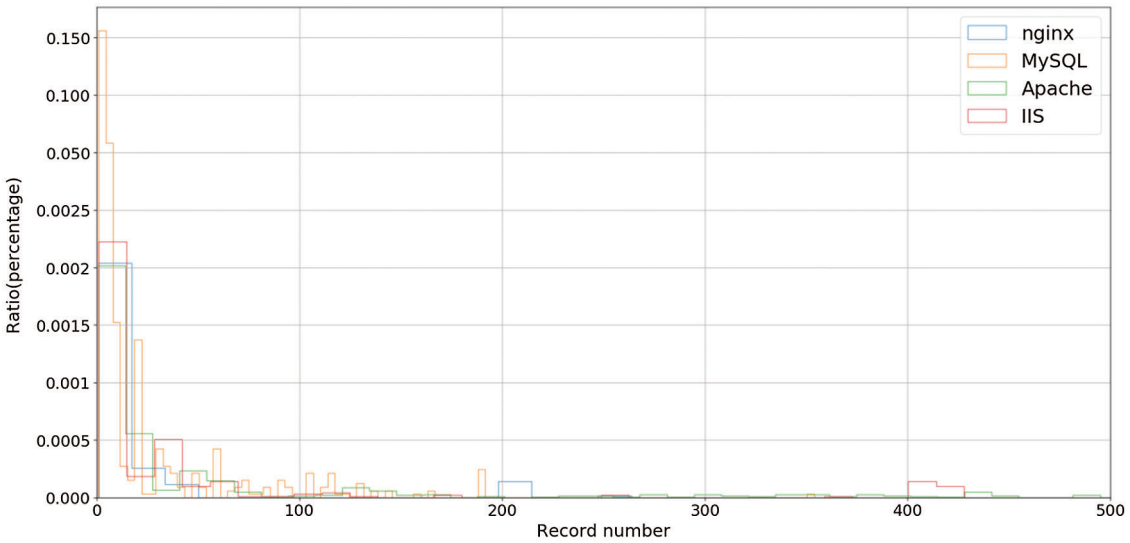

Figure 2: Quantity distribution of host vulnerability regarding to record number

5.3 Results and Analysis

This section compares the proposed algorithm with traditional associative rules mining algorithms based on accuracy and efficiency. The accuracy of algorithms versus the support based on the Apache dataset is shown in Tab. 2, from which we can see that the accuracy of most associative rule mining algorithms is low, at less than 2%. Notably, the data have a long-tailed distribution, and most of the cases have low support, causing the associative rule mining algorithms to fail. However, the proposed algorithm displays better performance than the other algorithms, with the highest accuracy reaching 98.22%. The accuracy decreases with increasing support because the higher the support level is, the fewer the number of knowledge rules provided for the support.

Table 2: Comparison of the accuracy ratio for penetration testing knowledge discovery based on the Apache dataset with different support levels

Algorithm	20%	30%	40%	50%	60%	70%	80%
Apriori	0.01268	0.01057	0.00861	0	0.0053	0.0047	0.00453
FP-Growth	0.01892	0.01516	0.01326	0	0.00971	0.00851	0.00782
LCMFreq	0.01189	0.00983	0.00792	0	0.00508	0.00468	0.00467
PrePost+	0.01506	0.0113	0.0094	0	0.00585	0.00465	0.00396
Bayesian	**0.98221**	**0.98221**	**0.97966**	0.59085	0.02541	0.00127	0

As accuracy cannot reflect all aspects of performance, we adopt receiver operating characteristic curves (ROC) analysis to describe the performances of the algorithms, as shown in Fig. 3. The performance of the algorithms is assessed based on four experimental datasets, where (a), (c), (e) and (g) display the performance for support values ranging from 0.2 to 0.8 and (b), (d), (f) and (h) are locally enlarged regions of (a), (c), (e) and (g) respectively. Because of the memory corruption problem, there is no line representing FP-Growth on IIS and nginx datasets. Furthermore, from (a), (c), (e) and (g), we can see that the ROC curve is sensitive to the support parameter. Although the ROC curve is a similar curve to the standard line, this similarity does not mean that the proposed algorithm is a random algorithm because mining penetration semantic knowledge rules from large quantities of penetration testing data is not a simple classification task. Moreover, the ROC curves of the comparative associative rule mining algorithms show a "cluster" phenomenon, and the accuracy is low regardless of the value of the support parameter. This phenomenon illustrates that associative rule mining algorithms are not suitable for penetration semantic knowledge mining problems, and the performance of these methods is far less than the performance of the proposed method. Additionally, the locally enlarged plots show that the Apriori algorithm performs better than other associative rule mining algorithms, but the corresponding accuracy is still less than 2%, which is far from meeting the requirements for penetration semantic knowledge mining. To obtain the best support for the proposed algorithm, we adopt the Youden index [22] to evaluate the performance of the proposed algorithm based on two different support levels. The Youden index formula is shown as follows:

$$Youden = TPR - FPR \tag{6}$$

The performance of the Youden index based on the Apache dataset is shown in Tab. 3, from which we can see that the critical value is 0.0559, meaning that the best support parameter is 50%. When the best support parameter is used, the TPR improves, and the FPR decreases.

Furthermore, we compared the memory and CPU consumption of algorithms on the Apache dataset. The performance is shown in Fig. 4, where each bottom point is used to differentiate support. Fig. 4 shows that associative rule mining algorithms have similar memory consumption, and the PrePost + algorithm consumes the most memory (approximately 1080 MB). The proposed algorithm has a similar memory consumption of approximately 1250 MB. Regarding CPU consumption, the proposed algorithm consumes less memory than the compared associative rule mining algorithms, with a reduction of approximately 1.8%. What causes this difference is that the proposed algorithm transforms the parameter solution problem into a statistical problem so that the computational complexity is considerably reduced, thus decreasing CPU consumption.

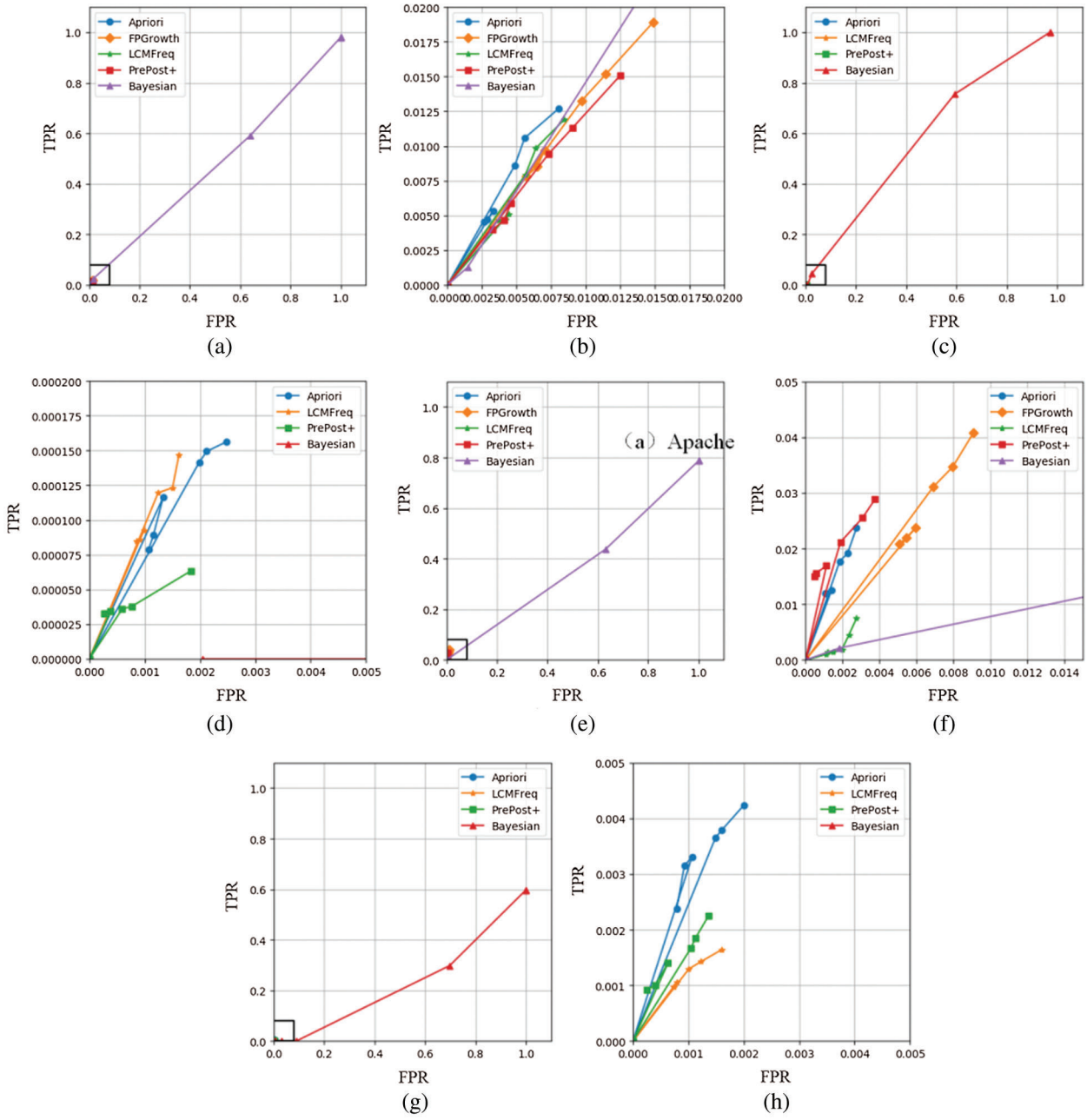

Figure 3: Receiver operating characteristic curves and locally enlarged receiver operating characteristic curves of different algorithms for each dataset. (a) Apache. (b) Apache detail. (c) IIS. (d) IIS detail. (e) MySQL. (f) MySQL detail. (g) Nginx. (h) Nginx detail

Table 3: Comparison of Youden index values for the proposed algorithm based on the Apache dataset

No.	Support	TPR	FPR	Youden
1	20%	1.0000	0.9822	0.0178
2	30%	1.0000	0.9822	0.0178
3	40%	1.0000	0.9822	0.0178
4	50%	0.6340	0.5781	**0.0559**
5	60%	0.0313	0.0165	0.0148
6	70%	0.0029	0.0	0.0029
7	80%	0.0	0.0	0.0

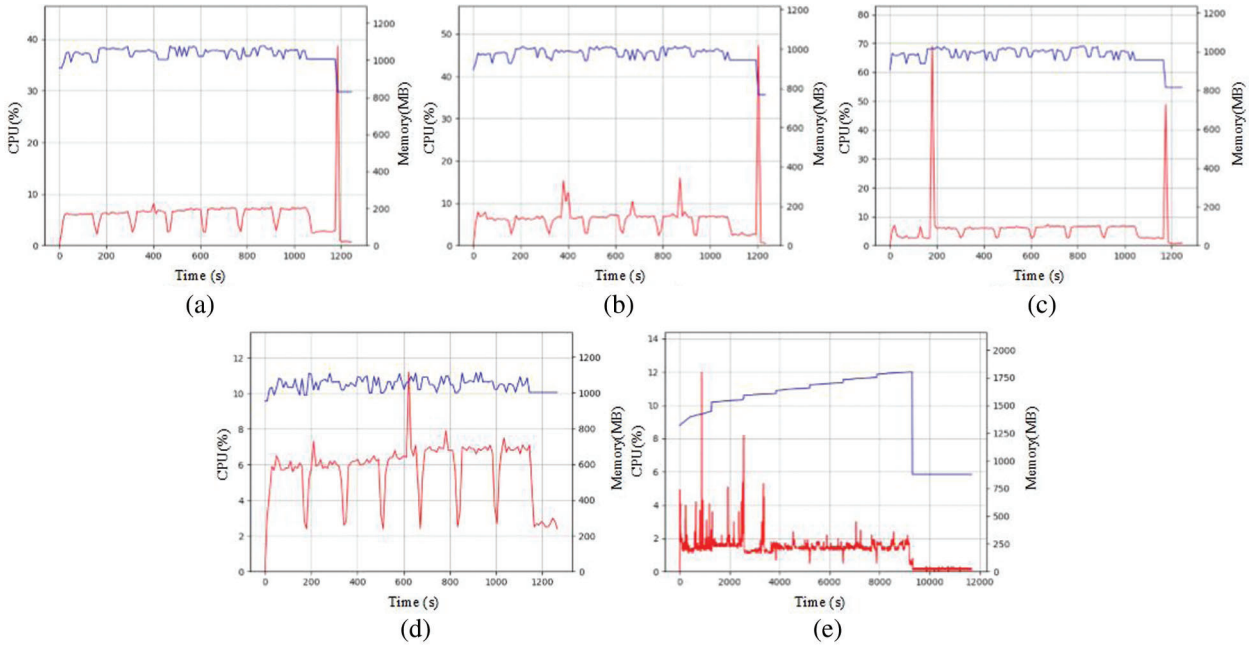

Figure 4: Comparison of memory and CPU consumption for each data mining algorithm based on the Apache dataset. (a) Apriori. (b) FP-Growth. (c) LCMFreq. (d) PrePost+. (e) Bayesian inference

6 Conclusion

Considering the problem of mining penetration semantic knowledge to automate penetration testing, this paper proposed a Bayesian inference based penetration semantic knowledge mining algorithm. First, a Bayesian network model is constructed according to penetration testing data, and the parameter solution process divides the whole network into smaller subnetworks according to independence analysis, after which the variable elimination method is adopted to retrieve penetration semantic knowledge. The experimental results show that the proposed method is superior to other associative rule mining algorithms. Moreover, there are still some tasks that could contribute to this work such as adopting domain knowledge in penetration semantic knowledge mining to improve the accuracy and decreasing computational complexity through parallel techniques.

Funding Statement: This research was supported by the National Natural Science Foundation of China No. 61502528.

Conflicts of Interest: We declare that there are no conflicts of interest to report regarding the present study.

References

[1] W. H. Han, Z. H. Tian, Z. Z. Huang, L. Zhong and Y. Jia, "System architecture and key technologies of network security situation awareness system yhsas," *Computers, Materials & Continua*, vol. 59, no. 1, pp. 167–180, 2019.

[2] X. Su, X. C. Liu, J. C. Lin, S. M. He, Z. J. Fu *et al.,* "De-cloaking malicious activities in smartphones using http flow mining," *KSII Transactions on Internet and Information Systems*, vol. 11, no. 6, pp. 3230–3253, 2017.

[3] B. Genge and C. Enăchescu, "ShoVAT: Shodan-based vulnerability assessment tool for internet-facing services," *Security and Communication Networks*, vol. 9, no. 15, pp. 2696–2714, 2016.

[4] C. Zhang, G. Almpanidis, W. W. Wang and C. C. Liu, "An empirical evaluation of high utility itemset mining algorithms," *Expert Systems with Applications*, vol. 101, no. 1, pp. 91–115, 2018.

[5] H. Y. Wang and X. W. Liu, "The research of improved association rules mining apriori algorithm," in *Proc. 8th Int. Conf. on Fuzzy Systems and Knowledge Discovery*, Shanghai, China, pp. 961–964, 2011.

[6] A. Singh, J. Agarwal and A. Rana, "Performance measure of similis and fp-growth algorithm," *International Journal of Computer Applications*, vol. 62, no. 6, pp. 25–31, 2013.

[7] L. X. Qin, Y. X. Su, Y. B. Liu and B. Z. Liang, "A compact fp-tree and array-technique based algorithm for frequent patterns mining," *Journal of Computer Research and Development*, vol. 45, no. 1, pp. 244–249, 2008.

[8] M. Deypir and M. H. Sadreddini, "Eclat: An efficient sliding window based frequent pattern mining method for data streams," *Intelligent Data Analysis*, vol. 15, no. 4, pp. 571–587, 2011.

[9] X. J. Feng, J. Zhao and Z. Y. Zhang, "Mapreduce–based h–mine algorithm," *Application Research of Computers*, vol. 33, no. 3, pp. 754–758, 2016.

[10] Z. H. Deng, Z. H. Wang and J. J. Jiang, "A new algorithm for fast mining frequent itemsets using n-lists," *Science China Information Sciences*, vol. 55, no. 9, pp. 2008–2030, 2012.

[11] Z. H. Deng and S. L. Lv, "PrePost+: An efficient n-lists–based algorithm for mining frequent itemsets via children-parent equivalence pruning," *Expert Systems with Applications*, vol. 42, no. 13, pp. 5424–5432, 2015.

[12] Z. H. Deng and S. L. Lv, "Fast mining frequent itemsets using nodesets," *Expert Systems with Applications*, vol. 41, no. 10, pp. 4505–4512, 2014.

[13] N. Aryabarzan, B. Minaei-Bidgoli and M. Teshnehlab, "Negfin: An efficient algorithm for fast mining frequent itemsets," *Expert Systems with Applications*, vol. 105, no. 1, pp. 129–143, 2018.

[14] B. Luca and X. Li, "A two-phase algorithm for mining sequential patterns with differential privacy," in *Proc. 22nd Int. Conf. on Information and Knowledge Management*, San Francisco, USA, pp. 269–278, 2013.

[15] V. S. Tseng, C. W. Wu, B. Shie and P. S. Yu, "UP-Growth: An efficient algorithm for high utility itemset mining," in *Proc. of 16th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, Washington, DC, USA, pp. 253–262, 2010.

[16] M. Liu and J. F. Qu, "Mining high utility itemsets without candidate generation," in *Proc. 21st ACM Int. Conf. on Information and Knowledge Management*, Maui, HI, USA, pp. 55–64, 2012.

[17] Krishnamoorthy and Srikumar, "Pruning strategies for mining high utility itemsets," *Expert Systems with Applications*, vol. 42, no. 5, pp. 2371–2381, 2015.

[18] A. Y. Peng, Y. S. Koh and P. Riddle, "Mhuiminer: A fast high utility itemset mining algorithm for sparse datasets," in *Proc. 21st Pacific–Asia Conf. on Knowledge Discovery and Data Mining*, Jeju, South Korea, pp. 196–207, 2017.

[19] M. Babaeian, P. A. Blanche, R. A. Norwood, T. Kaplas, P. Keiffer *et al.,* "Nonlinear optical components for all-optical probabilistic graphical model," *Nature communications*, vol. 9, no. 1, pp. 1–8, 2018.

[20] A. Tundis, W. Mazurczyk and M. Mühlhäuser, "A review of network vulnerabilities scanning tools: Types, capabilities and functioning," in *Proc. of 13th Int. Conf. on Availability, Reliability and Security*, Hamburg, Germany, pp. 1–10, 2018.

[21] P. Fournier-Viger, J. C. Lin, A. Gomariz, T. Gueniche and A. Soltani, "The SPMF open-source data mining library version 2," *Proc. of Joint European Conf. on Machine Learning and Knowledge Discovery in Databases*, Riva del Garda, Italy, pp. 36–40, 2016.

[22] J. Luo and C. Xiong, "Youden index and associated cut-points for three ordinal diagnostic groups," *Communications in Statistics*, vol. 42, no. 6, pp. 1213–1234, 2013.