Modeling and Vulnerable Points Analysis for E-commerce Transaction System with a Known Attack

Chapter · November 2016

DOI: 10.1007/978-3-319-49148-6_35

4 authors, including:

Guanjun Liu
Tongji University
39 PUBLICATIONS **234** CITATIONS

SEE PROFILE

Modeling and Vulnerable Points Analysis for E-commerce Transaction System with a Known Attack

Mimi Wang[1,2], Guanjun Liu[1,2(✉)], Chungang Yan[1,2], and Changjun Jiang[1,2]

[1] Department of Computer Science, Tongji University, Shanghai 201804, China
wangmimi2013@hotmail.com,
{liuguanjun,yanchungang,cjjiang}@tongji.edu.cn
[2] Key Laboratory of Ministry of Education for Embedded System and Service Computing, Tongji University, Shanghai 201804, China

Abstract. With the rapid development of network and mobile terminal, online trading has become more and more widespread. However, E-commerce transaction systems aren't completely strong due to the openness of network. Some points of a system is vulnerable in the real world and thus they can be utilized by attackers and cheaters. We focus on E-commerce transaction systems with attacks, and propose a kind of Petri nets called VET-net (Vulnerable E-commerce Transaction nets) to model them. A VET-net considers both normal actions belonging to the related system and malicious actions ones such as tampering with a data. Based on VET-net, this paper proposes the concepts of vulnerable points and vulnerable levels in order to describe the cause and levels of vulnerability. And then it uses the dynamic sling method to locate the vulnerable points. A real example is used to illustrate the effectiveness and rationality of our concepts and method.

Keywords: E-commerce transaction · Vulnerable piont · Vulnerable level · Dynamic slicing · Petri nets · Known attack

1 Introduction

With E-commerce booming in China, online trading has become more and more widespread. Electronic payment (e-payment) has become an important trading activity in people's daily life. iResearch Consulting statistics show that the size of China's online-shopping market amounted to 16.4 trillion Yuan in 2014, and they have become an important force to promote the economy. The volume of online shopping with a third-party payment platform (TPP) reached 11.8 trillion Yuan in 2015, representing an increase of 46.9 % over 2014 [1].

However, e-payment is a new way of trade in recent years and still in a development process. There are some cases are shown that the system is under attack. We analyze the case and show the E-commerce system vulnerable to *combined attack*. The inherent vulnerability of the system to combined attack

G. Wang et al. (Eds.): SpaCCS 2016, LNCS 10066, pp. 422–436, 2016.
DOI: 10.1007/978-3-319-49148-6_35

lies in the usage of the Trojan and fishing website in the pay process, which imperil the security and dependability of E-commerce system. In a E-commerce system, where processes are attacked, the challenge of finding vulnerabilities becomes more complicated and more pressing at the same time. For example, a company offering a vulnerable point to others will not only suffer locally from an internal defect, but might face penalties arising from, e.g., attack. [2] divides system vulnerability into two types: one is deadlock or internal defect of system itself, another is the vulnerability due to influence of external actions such as external attack. Vulnerability types have to be manually derived before tools can employ the corresponding vulnerability patterns for automated analyses.

Figure 1 shows a real case occurred in Mongolia. Criminal B buys a Trojan program through the QQ, and then obtains the buyers' IDs and the sellers' contact phone numbers from the TaoBao transaction records by using this Trojan program. B calls and tells seller C that he is a friend of buyer A and helps him to change the address. B calls to A and says that he is seller C and AliPay is upgrading now, and thus A should pay one yuan to activate this upgrade. B deliberately complicates activation process and then A cannot operate it. A has to ask for B to provide a remote assistance through QQ. B implants a Trojan program into the computer of A through QQ remote assistance. Then B steals the money from the net card of A and transfers them to account $a1$ of B. B uses this account to buy Unicode prepaid phone $CARDS$ and builds his own online shop on TaoBao. B asks his partner D to virtually buy card on online store, and then B transfers money from account $a1$ to account $a2$ of AliPay of B. In fact, Fig. 1 gives an attack process. And Fig. 3 illustrates this cheat process. In this case, we observe that a combination of both Trojan and fishing website would prevent the system from being vulnerable to the combined attack.

Technology today has a big impact on system, both in the manner of daily usage in our private lives and even more when it comes to business. For the vulnerability problem, it is different to define vulnerability and vulnerable points due to different concern objects and application scope. Krsul [12] first defined software vulnerabilities in his doctoral dissertation. Some weak points might only be some component defects, but most of the vulnerability stemmed from the operating system kernel, privilege processes, file systems, service process and the network system components. [13] gave the definition of security vulnerability of E-commerce: a security vulnerability is a weakness in a product that can allow an attacker to compromise the integrity, availability, or confidentiality of that product. However, in the real word, due to more and more attack forms such as Trojan, Fishing site attack, only to study the system itself' correctness or holes of designing system already cannot satisfy the needs of reality. We think that it is significant to consider the vulnerability under an attack. So the definition of vulnerability is based on an attack. We think that for the real E-commerce transaction system, the *vulnerability* is a weakness that induced the actions of some attack. In fact, for a unknown attack form, there isn't a good method to locate the *vulnerable region* (the vulnerable region can be considered to be some vulnerable points joined up to implement an attack)of system. As shown in Fig. 1,

what we consider is the vulnerability due to external attacks. Fault/attack trees [7–9] and Failure Mode and Effects Analysis (FMEA) [10,11] are two prominent representatives of manual analysis methods. The strength of these methods is that they leave much room for the security expert to apply subjective skills and personal experience, enabling the discovery even of completely new types of vulnerabilities. However, given the vulnerabilities observed in real-world systems, the importance of searching for completely new types of vulnerabilities should not be overestimated. And these methods require much experience and provide little guidance during the analysis. So it has be a challenge foe us to analyze vulnerability in the absence of experience.

For a known attack form, we may try our best to find the *vulnerable region*. If we know the attack, it is more effective and credible to calculate the vulnerable parts.

According to the condition, we know that:

(1) the vulnerability type is the second one, i.e. the vulnerability of E-commence transaction system is caused by external attacks. At the same tine, attack process is known. In fact, Fig. 1 describes the attack process.
(2) the E-commerce transaction system itself is not strong. The attack process shows that for a E-commerce transaction system, attackers are likely to attack it. It shows that there are some safety loopholes in the system. For example, why the transaction record can be divulged. Whether the case will not or easily occur if the system has a verification process when the user uses activation process. Why there is no verification process when the user transfers money to account $a1$.
(3) there are some vulnerable regions. From the case, there are more than one points touched to the attacker' ultimate goal. Attacker uses more than one points to attack the system.

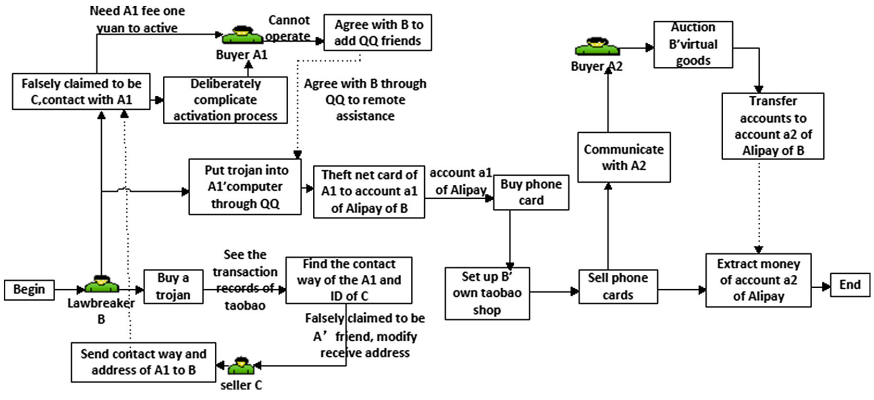

Fig. 1. A real case of E-commerce transaction system under an attack occurred in Mongolia

So based on the above, the questions are: how to use a formal method to locate the vulnerable points of a system with some given attacks and how to formally declare its vulnerable levels. This paper focuses on these questions and obtains the following results:

(1) We start from a real case of Alipay, and define *Vulnerable E-commerce Transaction net (VET-net)*. The VET-net is a labeled Petri net which can describe a real system considering malicious actions. The malicious actions are caused by some attacks.
(2) We propose the concept of *vulnerable points* that result in unpredictable states. And then we give three *vulnerable levels* to describe three different *vulnerable point sets* which express different safety levels.
(3) Based on the dynamic slicing algorithm, we further present a new method to locate the vulnerable points.

The structure of this paper is as follows: Sect. 2 reviews some basic knowledges. Section 3 defines VET-net. Section 4 defines vulnerable points and three vulnerable levels. Section 5 proposes a method to detect the vulnerable points, and conclusion is given in Sect. 6.

2 Background

This section recalls the basic concepts and definitions used in this paper. For more details, one can refer to [14,18].

Computing a net slice can be seen as a graph reachability problem. Program slicing is first defined by Weiser [17] in the context of program debugging. In particular, Weiser uses program slicing to isolate a program that contains a bug and thus it becomes easier for programmers to find this bug. In general, slicing extracts these statements that can affect some point of interest as the slicing criterion. The concepts of slicing criterion and slice can be found in [4,22,23].

Definition 1 (Slicing Criterion). *Let $N = (P, T, F)$ be a net. A slicing criterion of N is a pair $\langle M_0, Q\rangle$ where M_0 is an initial marking of N and $Q \subseteq P$ is a set of places.*

Definition 2 (Slice). *Let $N = (P, T, F)$ be a net and $\langle M_0, Q\rangle$ be a slicing criterion of N. $N' = (P', T', F')$ is a slice of N w.r.t. $\langle M_0, Q\rangle$ if the following conditions hold:*

(1) N' is a subnet of N; and
(2) for each firing sequence $\sigma = t_1 t_2 \cdots t_n$ of (N, M_0) such that $M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \cdots \xrightarrow{t_n} M_n$ and $M_{n-1}(p) < M_n(p)$ for some $p \in Q$, there exists a firing sequence σ' for (N', M'_0) with $M'_0 = M_0 \mid_{P'}$, such that
(i) σ' is the projection of σ over T',
(ii) $M'_0 \xrightarrow{\sigma'} M'_m$, $m \leq n$, and
(iii) M'_m covers $M_n \mid_{P'}$ (i.e. $M'_m \geq M_n \mid_{P'}$)

For example in [23], Fig. 2(b) shows the slice N_1 considering the slicing criterion $\langle M_0, \{p_5, p_7, p_8\}\rangle$.

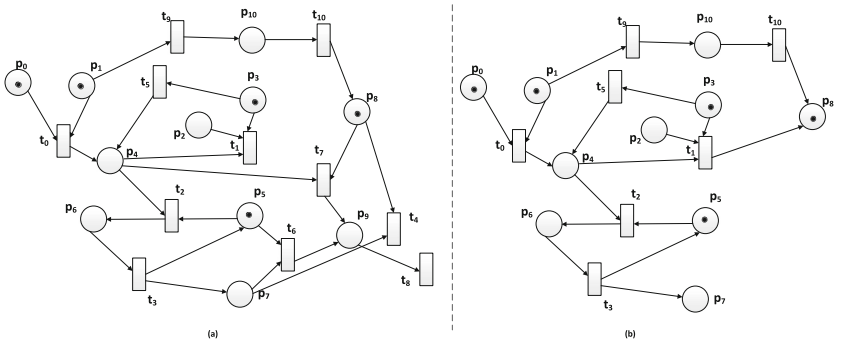

Fig. 2. An example introducing the slice: (a) an example from [23], (b) its slice.

3 Vulnerable E-commerce Transaction Net Under Attacks

In this section, we define Vulnerable E-commerce Transaction net (VET-net). We first recall *Attack*.

Definition 3 (Attack [20]). *An attack is a sequence of actions* $a_1.a_2 \cdots a_n \in T$ *such that there exists an elementary path from some initial state induced by* $a_1 \cdots a_n$ *and which reaches the set* G*. Let Attack(T) be the set of attacks in* T*; it is finite.* G*- the final states are the states where attackers hope to achieve.*

In our case of Fig. 3, the set G consists of state p_{b20} and p_{b23}.

Definition 4 (LPN [19]). *An LPN is a 6-tuple* $\Sigma = (P, T, F, M_0, E, \lambda)$*, where*

(1) (P, T, F, M_0) *is a Petri net;*
(2) E *is a set of actions including unobservable action* ε*; and*
(3) $\lambda : T \rightarrow E$ *is a label function.*

In order to describe and understand VET-net better, we first give the following two concepts.

Definition 5 (Malicious transition). *Given a net* $N = (P, T, F)$*, a malicious transition* $t_Y \in T$ *is a transition that is not a normal action of a transaction system (such as replaced or tampering with data).*

Definition 6 (Malicious place). *Given a net* $N = (P, T, F)$*, a maliciousplace* $p_Y \in P$ *is a place if it is associated with* t_Y*.*

In Fig. 3, the place p_{b20} and p_{b23} are two malicious ones since transition b_{19} is malicious. Our idea is to look for those source points leading to these malicious states.

Definition 7 (VET-net). *An VET-net is an 11-tuple* $(P, T, F, M_0, p_I, p_F, A, I, O, \lambda)$*, where:*

(1) (P, T, F, M_0) *is a Petri net;*P *is a finite set of places,* $p_I \in P$ *is the source place satisfying* ${}^\bullet p_I = \emptyset$ *and* $p_F \in P$ *is the sink place satisfying* ${}^\bullet p_F = \emptyset$*;*
(2) A *is a finite set of action;*
(3) $I = \{yo, no, \epsilon\}$ *is the set of input symbols;*
(4) $O = \{yf, nf, \epsilon\}$ *is the set of output symbols;*
(5) $\lambda : T \to (A \times I \times O)$ *is the label function.*

We know that $P = P_Y \bigcup P_N$ and $T = T_Y \bigcup T_N$, which P_Y is the *malicious place set* and T_Y is the *malicious transition set.*

We characterize the real case using VET-net such as Fig. 3, obtaining an initial and fuzzy model. Table 1 introduces the minings of transitions in Fig. 3. In Fig. 3, the transitions labeled by yo, yf mean that the actions aren't themselves' or tampered, the transitions labeled by no, yf mean that the actions aren't themselves' or tampered. In order to better describe the abnormal trading with tampered or replace behaviors in the electronic transaction process, we put forward a vulnerable points based on VET-net.

In real trading process, there may be some unmoral trading state, such as amount stolen, illegal trade. There are many reasons caused by malicious state, such as unsubstantial safety awareness of user itself, malicious attacks from the outside, the design problem of the system itself. We in addition to call for the user cautious in online transactions, mainly to find the system itself' vulnerable points. According to the real case in this section, we want to find the points which are not strong of system, in order to ensure that the similar situations don't occur.

4 Vulnerable Points and Vulnerable Levels

The Vulnerable points we considered are under attack. Firstly in this section, we recall the concept of attack. For more details one can refer to [20,21]

Definition 8 (Vulnerable Point). *An vulnerable point is a place that induced the actions of some attack.*

(Malicious state in Fig. 3 is the state p_{b20} that isn't the desired state of normal trading process).

For a reachability graph, maybe not only one path to the malicious state, and the same points that are in different path may not have the same vulnerability.

Flowing, from a computation view, we give some related concepts.

For the actual application needs, we give following three kinds of vulnerable points set.

Definition 9 (First-level Vulnerable Points Set (FVP)**).** *Let VET-net=* $(P, T, F, M_0, p_I, p_F, A, I, O, \lambda)$ *a vulnerable E-commerce Transaction net and with* $i_i = t_{i1}t_{i2}\cdots t_{im}$*. And for a state* p_q*, the first-level vulnerable points set* FVP *satisfy the following conditions:*

(1) $FVP \subseteq P$*;*

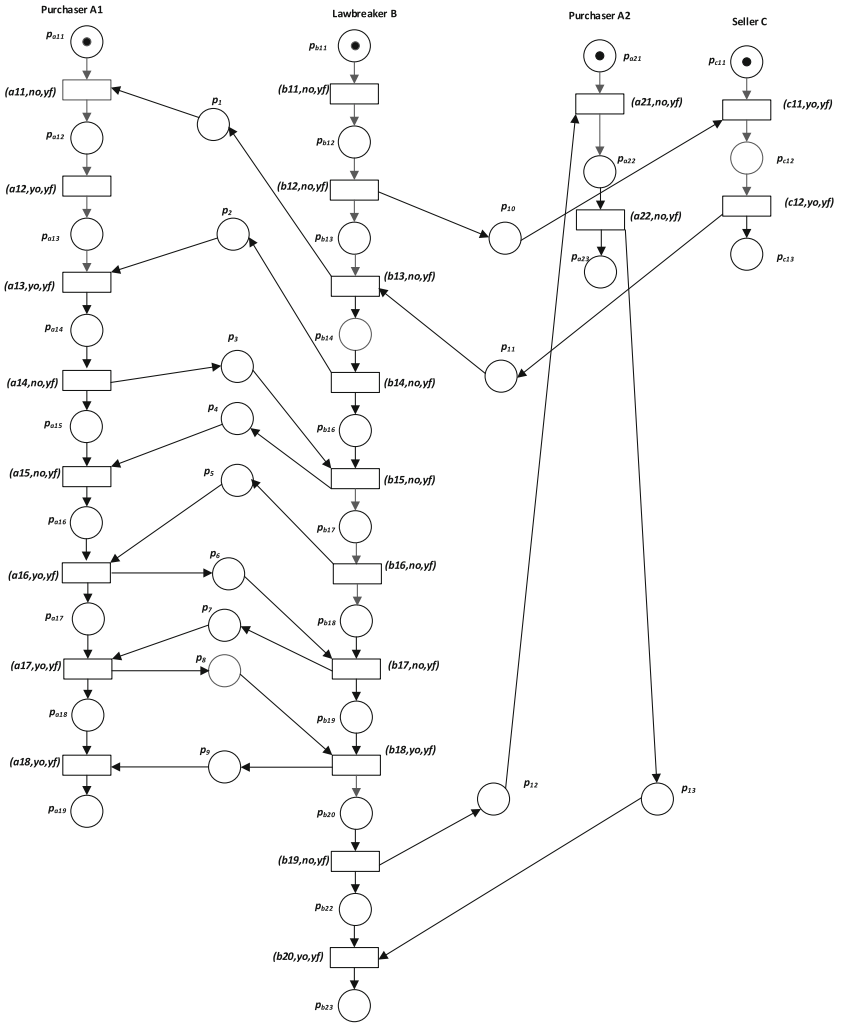

Fig. 3. *VET*-net model of Fig. 1

(2) $\sigma_1 \cap \sigma_2 \cap \cap \sigma_n \neq \emptyset$;
(3) $FVP = \{p_i \in^{\bullet} t_i \mid t_i \in (\sigma_1 \cap \sigma_2 \cap \cdots \sigma_n)\}$.

The path which leads to the first-level vulnerable points set is the public path. And these points in $FVP \backslash \{M_0\}$ is considered to be *Minimal Vulnerable Region*.

Definition 10 (Minimal Vulnerable Region(M_iVR)**).** *Let VET-net=*$(P, T, F, M_0, p_I, p_F, A, I, O, \lambda)$ *a vulnerable E-commerce Transaction net and with* $\sigma_i = t_{i1}t_{i2}\cdots t_{im}$. *And for a state* p_q, *the minimal vulnerable region* M_iVP *satisfy the following conditions:*

Table 1. Meaning of transitions in Fig. 3

Transitions in Fig. 3	Meaning
b_{11}	Query TaoBao transaction records, and get A1' and C' information
b_{12}	Falsely claimed to be A1' friend, contact with C
b_{13}	Contact with A1 and need A1 fee one yuan
b_{14}	Deliberately complicate activation process
b_{15}	Apply for A1 for QQ friends
b_{16}	Remote assistance through QQ
b_{17}	Put Trojan into A1' computer through QQ
b_{18}	Theft net card of A1 to account a1 of AliPay of B
b_{19}	Set up TaoBao shop to sell phone cards
b_{20}	Extract money of account a2 of AliPay
a_{11}	Receive B' call
a_{12}	Log on TaoBao
a_{13}	Fee the activation
a_{14}	Pay for failure
a_{15}	Agree with B to add as a QQ friend
a_{16}	Agree with B to remote assistance
a_{17}	Agree to pay 1 yuan for activation fee
a_{18}	Agree to transfer net silver to account a1 of AliPay
a_{21}	Auction B' virtual goods
a_{22}	Transfer accounts to account a2 of AliPay of B
c_{11}	Receive B' call
c_{12}	Send contact way and address of A1 to B

(1) $M_iVP \subseteq FVP$;
(2) $M_0 \notin M_iVP$.

Property 1. For a VET-net, if it exits a M_iLR, then the M_iLR must be only one.

Definition 11 (Second-level Vulnerable Points Set(SVP)**).** *Let VET-net=*$(P, T, F, M_0, p_I, p_F, A, I, O, \lambda)$ *a vulnerable E-commerce Transaction net, with* $\sigma_i = t_{i1}t_{i2}\cdots t_{im}$. *And for a state* p_q, *the second-level vulnerable points set* SVP *satisfy the following conditions:*

(1) $SVP \subseteq P$;
(2) $\sigma_1 \cup \sigma_2 \cup \cdots \sigma_n \neq \emptyset$;
(3) $SVP = \{p_i \in^{\bullet} t_i \mid t_i \in (\sigma_1 \cup \sigma_2 \cup \cdots \sigma_n)\}$.

The path which leads to the second-level vulnerable points set is the path only leading to malicious state. And these points in $SVP \setminus \{M_0\}$ is considered to be *Dimity Vulnerable Region*.

Definition 12 (Dimity Vulnerable Region (DVR)**).** *Let VET-net=*$(P, T, F, M_0, p_I, p_F, A, I, O, \lambda)$ *a vulnerable E-commerce Transaction net and with* $\sigma_i = t_{i1}t_{i2}\cdots t_{im}$*. And for a state* p_q*, the dimity vulnerable region DVR satisfies the following conditions:*

(1) $M_iVP \subseteq SVP$*;*
(2) $M_0 \notin M_iVP$*.*

We can see that for a VET-net, if it exits a M_iLR, then the M_iLR may be more than one.

Definition 13 (Third-level Vulnerable Points Set (TVP)**).** *Let VET-net=* $(P, T, F, M_0, p_I, p_F, A, I, O, \lambda)$ *a vulnerable E-commerce Transaction net, with* $\sigma_i = t_{i1}t_{i2}\cdots t_{im}$*. And for a state* p_q*, the third-level vulnerable points set TVP satisfy the following conditions:*

(1) $TVP \subseteq P$*;*
(2) there isn't any state reached by firing sequence σ *expect for state* p_q*;*
(3) $TVP = \{p_i \in^{\bullet} t_i \mid t_i \in \sigma\}$*.*

The path which leads to the third-level vulnerable points set is all paths leading to malicious state. And these points in $TVP \setminus \{M_0\}$ is considered to be *Maximal Vulnerable Region*.

Definition 14 (Maximal Vulnerable Region (M_aVR)**).** *Let VET-net=*$(P, T, F, M_0, p_I, p_F, A, I, O, \lambda)$ *a vulnerable E-commerce Transaction net and with* $\sigma_i = t_{i1}t_{i2}\cdots t_{im}$*. And for a state* p_q*, the maximal vulnerable region* (M_aVP) *satisfies the following conditions:*

(1) $M_aVR \subseteq TVP$*;*
(2) $M_0 \notin M_aVR$*.*

According to three vulnerable points set, we can see that the vulnerable levels are similar to the dynamic slicing. So we can learn from the dynamic slicing computation.

The minimal vulnerable region, dimity vulnerable region, maximal vulnerable region are collectively called *Vulnerable Region*.

Property 2. For a VET-net, it must exit one and only one M_aLR.

Property 3. For a VET-net, if it exits M_iLR and DVR, then it must hold that $M_iLR \subseteq DVR \subseteq M_aLR$.

In this section, we firstly give a brief introduction of Llorens's dynamic slicing algorithm. And then we give our improved algorithm based on Llorens'.

In a seminal paper [23], Llorens has proposed a new algorithm to obtain dynamic slicing. The algorithm is based on the traces, generally produces smaller slices by considering a particular firing sequence.

Definition 15 (Dynamic Slice [23]). *Let $N = (P, T, F)$ be a Petri net and let $\langle M_0, Q\rangle$ be a slicing criterion for N, with $\sigma = t_1 t_2 \cdots t_n$. Then, we compute a dynamic slice N' of N w.r.t. $\langle M_0, \sigma, Q\rangle$ as follows:*

(1) We have $N' = (P', T', F')$, where $M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \cdots \xrightarrow{t_n} M_n$, $P' \bigcup T' = slice(M_n, \sigma, Q)$, $P' \subseteq P$, $T' \subseteq T$, and $F' = F \mid_{(P',T')}$. Auxiliary function slice is defined as follows:
(2) The initial marking M'_0 is the restriction of M_0 over P', i.e. $M'_0 = M_0 \mid_{P'}$.

Trivially, given a marked Petri net (N, M_0), the complete net N is always a correct slice w.r.t. any slicing criterion. The challenge then is to produce a slice as small as possible.

Algorithm 1. Improved slice algorithm

Input: A PN, its reachability graph RG, a goal state set p_q.

Output: Slice S_W.

for *all firing sequences $\sigma_1, \sigma_2, \cdots, \sigma_n$ from M_0 to P_q* **do**
  $S_W = \emptyset$;
  if $\sigma_1 \cap \sigma_2 \cap \cdots \sigma_n = C \neq \emptyset$ **then**

$$S_W = \begin{cases} P_q, & if\ i = 0 \\ slice(M_{i-1}, C, P_q), & if \forall p \in P_q . M_{i-1}(p) \geq M_i(p) \\ \{t_i\} \cup slice(M_{i-1}, C, P_q \cup^{\bullet} t_i), & otherwise \end{cases}$$

  end
  if $\sigma_1 \cap \sigma_2 \cap \cdots \sigma_n = \emptyset$ *and there is a path σ_i such that there is not any other state expect for P_q from σ_i* **then**

$$S_W = \begin{cases} P_q, & if\ i = 0 \\ slice(M_{i-1}, \sigma_i, P_q), & if \forall p \in P_q . M_{i-1}(p) \geq M_i(p) \\ \{t_i\} \cup slice(M_{i-1}, \sigma_i, P_q \cup^{\bullet} t_i), & otherwise \end{cases}$$

  else

$$S_W = \begin{cases} W, & if\ i = 0 \\ slice(M_{i-1}, \bigcup_{i=1}^{n} \sigma_i, W), & if \forall p \in W . M_{i-1}(p) \geq M_i(p) \\ \{t_i\} \cup slice(M_{i-1}, \bigcup_{i=1}^{n} \sigma_i, W \cup^{\bullet} t_i), & otherwise \end{cases}$$

  end
end

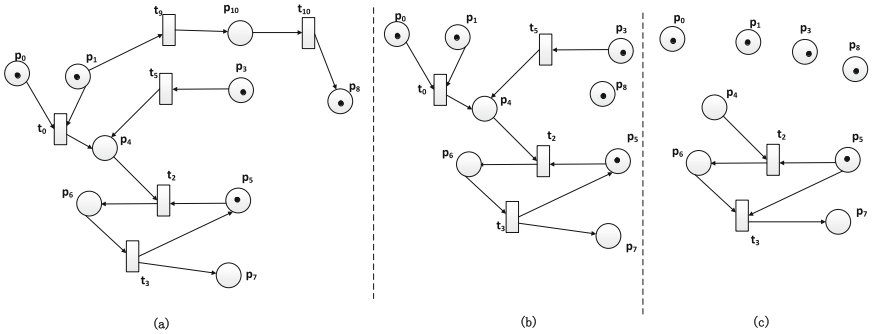

Fig. 4. The algorithm result for the example form [23]: (a) the results of Algorithm 1 in [23], (b) the results of Algorithm 2 in [22], (c) the results of our Algorithm.

Table 2. Algorithm comparison based on example in [23]

Algorithm	Required number of places	Required number of transitions
Algorithm from [22]	10	7
Algorithm 1 from [23]	9	6
Algorithm 2 from [23]	8	4
Our algorithm	8	2

We improve dynamic slicing method as shown in Algorithm 1.

For the example of Fig. 2(a) in [23]. It can get Fig. 4(a) by using the Llorens's algorithm 1, and Fig. 4(b) by using the Llorens's algorithm 2. Using our algorithm, it can get Fig. 4(c). We use the slice algorithm to get the vulnerable points, whose challenge is to produce a slice as small as possible. In fact, according to Table 2, we can see that our algorithm only uses the less points to reach the same state, which explains that our algorithm is better.

5 Vulnerability Analysis Method Based on Dynamic Slicing

To accurately find the vulnerable points of electronic trading system, we need to narrow vulnerable points down when ensuring that the system may result in malicious effect with less points, which need to study the following two questions: (1) in case of uncertain vulnerable points, how to find these vulnerable points; (2) when determining vulnerabilities, how to find lessees points to achieve the malicious state. So we rely on the Llorens's algorithm to lock preliminary points, and then use the improved algorithm to lock the less points.

In Sect. 4, we have introduced our computing method for dynamic slicing is more optimal than [22,23]. Then in this section, we firstly use the improved

dynamic slicing to lock vulnerable regions, and secondly we solve the problem proposed in Sect. 1.

According to Algorithm 1, we can obtain the S_W, then for all transitions of S_W, we can get at least one firing sequence to the goal state P_q which may be one or more than one states. Then there are three conditions as follows:

(1) for all firing sequences $\sigma_1, \sigma_2, \cdots \sigma_n$, $\sigma_1 \cap \sigma_2 \cap \cdots \cap \sigma_n \neq \emptyset$;
(2) for all firing sequences $\sigma_1, \sigma_2, \cdots \sigma_n$, and exit a firing sequences such that there is not any other state expect for Pq;
(3) for all firing sequences $\sigma_1, \sigma_2, \cdots \sigma_n$, and don't exit any firing sequences such that there is not any other state expect for Pq;

So based on the improved dynamic slicing method, we can get the vulnerable regions as shown in Algorithm 2. We can get the reachability graph as shown in Fig. 5 of Fig. 3 using the tool $PIPE$. For the known malicious state $S20$, we can get the firing sequence which can reach the $S20$.

Algorithm 2. Locking vulnerable regions algorithm

Input: A VET-net, its reachability graph RG, a slice S_W derived from Algorithm 1.

Output: Minimum vulnerable region M_iVR, dimity vulnerable region DVR , and maximum vulnerable region M_aVR.

for *all transitions of* S_W*, find a firing sequence* σ_i *of* RG **do**
  $M_iVR = \emptyset$;
  $DVR = \emptyset$;
  $M_aVR = \emptyset$;
  if σ_i *covers all transitions of* S_W **then**
    Computer the triggered transitions t_i of σ_i;
    Computer the correspondence points ${}^\bullet t_i$ of PN, written as P_{mi};
    $M_iVR = P_{mi}$;
  end
  if *find anther a firing sequence* σ_j*, and* $\sigma = \sigma_i \cup \sigma_j$ **then**
    Computer the triggered transitions t_j of σ_j;
    Computer the correspondence points ${}^\bullet t_j$ of PN, written as P_{mj};
    $DVR = P_{mj}$;
  else
    find until the last firing sequence σ_n such that $\sigma = \sigma_1 \cup \sigma_2 \cup \cdots \cup \sigma_n$;
    Computer the triggered transitions t_m of σ;
    Computer the correspondence points ${}^\bullet t_m$ of PN, written as P_{mt};
    $M_aVR = P_{mt}$;
  end
end

According to Fig. 5, there are six firing sequences to reach the state $S20$ as follows:

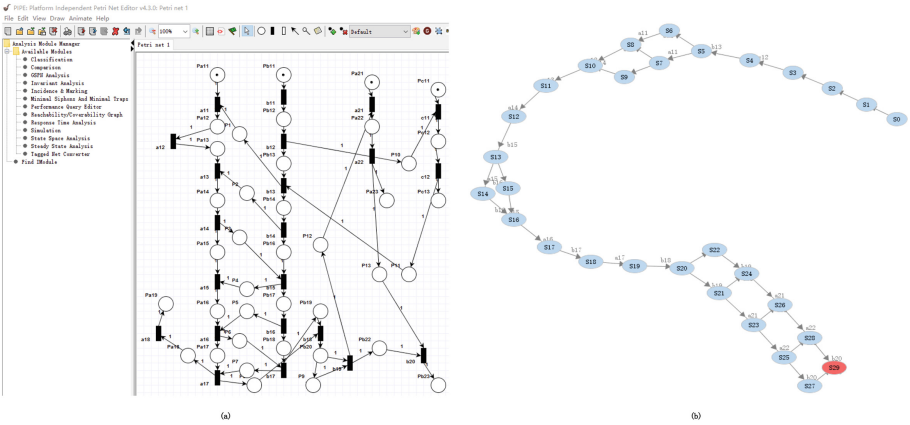

Fig. 5. PIPE implementation result of Reachability graph of Fig. 3: (a) Input Fig. 3, (b) output reachability graph of Fig. 3

(1) $\sigma_1 = b_{11}b_{12}c_{11}c_{12}b_{13}b_{14}a_{11}a_{12}a_{13}a_{14}b_{15}a_{15}b_{16}a_{16}b_{17}a_{17}b_{18}$;
(2) $\sigma_2 = b_{11}b_{12}c_{11}c_{12}b_{13}a_{11}a_{12}b_{14}a_{13}a_{14}b_{15}a_{15}b_{16}a_{16}b_{17}a_{17}b_{18}$;
(3) $\sigma_3 = b_{11}b_{12}c_{11}c_{12}b_{13}a_{11}b_{14}a_{12}a_{13}a_{14}b_{15}a_{15}b_{16}a_{16}b_{17}a_{17}b_{18}$;
(4) $\sigma_4 = b_{11}b_{12}c_{11}c_{12}b_{13}b_{14}a_{11}a_{12}a_{13}a_{14}b_{15}b_{16}a_{15}a_{16}b_{17}a_{17}b_{18}$;
(5) $\sigma_5 = b_{11}b_{12}c_{11}c_{12}b_{13}a_{11}a_{12}b_{14}a_{13}a_{14}b_{15}b_{16}a_{15}a_{16}b_{17}a_{17}b_{18}$;
(6) $\sigma_6 = b_{11}b_{12}c_{11}c_{12}b_{13}a_{11}b_{14}a_{12}a_{13}a_{14}b_{15}b_{16}a_{15}a_{16}b_{17}a_{17}b_{18}$.

Because $\sigma_1 \cap \sigma_2 \cap \cdots \sigma_6 \neq \emptyset$, according to Algorithm 2, it holds that $P_{mi} = \{p_{b1j}, p_{a1k}, p_s\}(j = 1, 2, \cdots, 18; k = 1, \cdots, 18; s = 1, 2, \cdots, 11)$, then $M_iVP = P_{mi} = \{p_{b1j}, p_{a1k}, p_s\}(j = 1, 2, \cdots, 18; k = 1, \cdots, 18; s = 1, 2, \cdots, 11)$. If any one of those points in M_iVP don't happen, the real case will not happen. Back to real Electronic trading system, we need to avoid those vulnerable points happen. In fact, we should consider the real system without attack in real case. These points not from attacker such as those interactive points p_i are vulnerable. In fact, in real E-commerce tradition system also exits those points. In the future, we will protective these points.

6 Conclusion

In this paper, on the basis of previous study, we define VET-net for vulnerable E-commerce transaction system, and propose vulnerable points based on Petri net, and propose a method to lock vulnerable regions and so as to locate vulnerable points of E-commerce tradition system, last give the analysis for actual case of AliPay.

In future work, there are a lot of problems need to study. Such as preventing being attacked, and dealing with fault, adding the data flow information to lock the vulnerable points, and so on, are to be considered in future research.

Acknowledgments. This paper is partially supported by the National Natural Science Foundation of China under grant Nos. 91218301 and 61572360.

References

1. http://www.iresearch.com.cn/report
2. Lowis, L., Accorsi, R.: Vulnerability analysis in SOA-based business processes. J. IEEE Trans. Serv. Comput. **4**, 230–242 (2011)
3. Georgiadis, C.K., Pimenidis, E.: Web services enabling virtual enterprise transactions. In: Proceedings IADIS International Conference on e-Commerce, Barcelona, Spain, pp. 297–302 (2006)
4. Wang, R., Chen, S., Wang, X. F., et al.: How to shop for free online-security analysis of cashier-as-a-service based web stores. In: 2011 IEEE Symposium on Security and Privacy, pp. 465–480. IEEE Press, New York (2011)
5. Du, Y.Y., Jiang, C.J., Zhou, M.C.: A Petri net-based model for verification of obligations and accountability in cooperative systems. J. IEEE Trans. Syst. Man Cybern. A Syst. Hum. **39**, 299–308 (2009)
6. Yu, W.Y., Yan, C.G., Ding, Z.J., Jiang, C.J., Zhou, M.C.: Modeling and validating e-commerce business process based on Petri nets. J. IEEE Trans. Syst. Man Cybern. A Syst. Hum. **44**, 327–341 (2014)
7. Vesely, W.E., Goldberg, F.F., Roberts, N.H., Roberts, N.H.: Fault Tree Handbook. Nuclear Regulatory Commission, Washington DC (1981)
8. Schneier, B.: Attack trees. Dr. Dobb's J. **24**, 21–29 (1999)
9. Saini, V., Duan, Q., Paruchuri, V.: Threat modeling using attack trees. J. Comput. Sci. Coll. **23**, 124–131 (2008)
10. Ravi, S.N., Prabhu, B.S.: Modified approach for prioritization of failures in a system failure mode and effects analysis. J. Int. J. Qual. Reliab. Manage. **18**, 324–336 (2001)
11. Roseti, L., Serra, M., Bassi, A., et al.: Failure mode and effects analysis to reduce risks of errors in the good manufacturing practice production of engineered cartilage for autologous chondrocyte implantation. J. Curr. Pharm. Anal. **12**, 43–54 (2016)
12. Krsul, I.V.: Software vulnerability analysis. Ph.D. thesis, West Lafayette, IN, USA, Major Professor-Eugene H. Spafford (1998)
13. Dianxiang, X., Kendall, E.N.: Threat-driven modeling and verification of secure software using aspect-oriented Petri nets. J. IEEE Trans. Softw. Eng. **32**, 265–278 (2006)
14. Murata, T.: Petri nets: properties, analysis and applications. J. Proc. IEEE **1989**(77), 541–580 (1989)
15. Sun, H., Fu, X., Xie, S., Jiang, Y., Guan, G., Wang, B.: A novel slicing method for thin supercapacitor. J. Adv. Mater., Online press (2016)
16. Peterson, J.L.: Petri nets. J. ACM Comput. Surv. (CSUR) **9**, 223–252 (1997)
17. Weiser, M.: Program slicing. J. IEEE Trans. Soft. Eng. **10**, 352–357 (1984)
18. Yu, W., Ding, Z., Fang, X.: Dynamic slicing of Petri nets based on structural dependency graph and its application in system analysis. Asian J. Control **17**, 1403–1414 (2015)
19. Peterson, J.: Petri Net Theory and the Modeling of Systems. Prentice Hall, Upper Saddle River (1981)

20. Pinchinat, S., Acher, M., Vojtisek, D.: Towards synthesis of attack trees for supporting computer-aided risk analysis. In: Canal, C., Idani, A. (eds.) SEFM 2014. LNCS, vol. 8938, pp. 363–375. Springer, Heidelberg (2015). doi:10.1007/978-3-319-15201-1_24
21. Mauw, S., Oostdijk, M.: Foundations of attack trees. In: Won, D.H., Kim, S. (eds.) ICISC 2005. LNCS, vol. 3935, pp. 186–198. Springer, Heidelberg (2006). doi:10.1007/11734727_17
22. Rakow, A.: Slicing Petri nets with an application to workflow verification. In: Geffert, V., Karhumäki, J., Bertoni, A., Preneel, B., Návrat, P., Bieliková, M. (eds.) SOFSEM 2008. LNCS, vol. 4910, pp. 436–447. Springer, Heidelberg (2008). doi:10.1007/978-3-540-77566-9_38
23. Llorens, M., Oliver, J., Silva, J., Tamarit, S., Vidal, G.: Dynamic slicing techniques for Petri nets. J. Elec. Notes Theor. Comput. Sci. **223**, 153–165 (2008)