IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON DATA-ENABLED INTELLIGENCE FOR DIGITAL HEALTH

Received October 30, 2019, accepted November 15, 2019, date of publication November 25, 2019,
date of current version December 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2955466

A Complex Event Processing-Based Online Shopping User Risk Identification System

ZHUOJING MA[1,2], WANGYANG YU[1,2], XIAOJUN ZHAI[3], (Member, IEEE), AND MENGHAN JIA[1,2]

[1]Key Laboratory for Modern Teaching Technology, Ministry of Education, Shaanxi Normal University, Xi'an 710119, China
[2]School of Computer Science, Shaanxi Normal University, Xi'an 710062, China
[3]School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, U.K.

Corresponding authors: Wangyang Yu (ywy191@snnu.edu.cn) and Xiaojun Zhai (xzhai@essex.ac.uk)

This work was supported by the National Natural Science Foundation of China (No.: 61602289), the Key Laboratory of Embedded System and Service Computing (Tongji University), Ministry of Education, ESSCKF2018-01, the Fundamental Research Funds for the Central Universities of China (No.: GK201803081, GK201801004, and GK201806012), the National Natural Science Foundation of China No.: 61872228), the Primary Research and Development Plan of Shaanxi Province (NO: 2018SF-361), the National Key RD Program of China under Grant No. 2017YFB1402102, the Secretary and Professor Studio Program of SNNU (Grant No.: SDTR201809), the Golden Course Project (Computational Thinking) of Shaanxi Normal University, and the Public Course Reform Project of Shaanxi Normal University under Grant No. 19GGK-JG02.

ABSTRACT Online shopping is an important part of the development of the Internet and plays a critical role in the current and future economy. However, there are many risks in the trading process. In order to reduce the hidden risks, it is necessary to study the method of risk identification. This paper proposes user risk identification method of online shopping system based on Complex Event Process (CEP). In this paper, we use the Esper as the CEP engine and the risk behavior patterns are defined as the event pattern language. Firstly, the CEP system captures event streams by analyzing data streams in real-time. Secondly, the captured event streams are sent to the CEP's engine. Finally, the Esper intelligently analyzes user's online shopping risk behaviors in real-time according to the event pattern languages. User risk identification effectively guarantees the fund and account security of the shopping users.

INDEX TERMS Complex event process, behavior patterns, risk identification, esper, intelligent analysis.

I. INTRODUCTION

In recent years, the Internet industry has shown a booming development tread, the increasing number of users have provided a powerful impetus for the rapid growth of online shopping. More and more people use online payment for shopping systems, which facilitates people's lives and has become a core part. However, there are many loopholes in the processes of payment. The criminals use the loopholes to carry out the illegal and criminal activities, which pose a great threat to the online shopping users' funds and account security. Therefore, to propose a system that could perform real-time identification of online shopping behavior is extremely urgent.

In order to ensure the security of online shopping, traditional research methods mainly focused on identity authentication such as the digital certificate [1], [2], encryption technology [3], and dynamic verification code [4], [5], etc. However, the identity authentication cannot guarantee the legitimacy of users' behaviors. For example, while the user identity is legal, he/she can achieve malicious purposes by using API calls, multiple accounts information to achieving alternative attacks, and spoofing identity. Currently, there are many such cases [6]–[10]. Another part of researches focused on secure protocols [11]. However, these studies cannot guarantee the security of users' behavior.

Complex event processing (CEP) is an analysis technology based on event flow in a dynamic environment and a relatively complete event processing framework, which can achieve efficient access to real-time data streams and provide advanced query functions [12]–[16], which is an ideal framework [17] in the process of online shopping. Real-time monitoring of the user's shopping data streams and intelligent identification of shopping behavior risks, which can effectively improve the security of online shopping. Events usually refer to meaningful state changes. The complex event processing platform converts the data streams into event streams by using techniques such as filtering, byte filling and aggregation, then sends the obtained event streams to the engine, and formulates event pattern languages (EPL)

The associate editor coordinating the review of this manuscript and approving it for publication was Yongtao Hao.

according to the timing relationship and logical relationship between the events, and finally make continuous queries on the sequences of the events that meet the requirements of the event streams [18], [19]. The CEP engine is a major component of the CEP system that handles events. In this paper, the CEP engine we use is Esper, an open source, JAVA-based, lightweight, and extensible event stream processing engine [20]–[22]. The research on complex event processing is mainly applied to financial detection [23], [24], intrusion detection [25], RFID reading [26]–[28], and wireless sensor network [29]. However, in the literature, as we know, it is rarely used in online shopping.

This paper proposes a CEP based risk identification method for online shopping systems, which mainly includes the following aspects: Firstly, the users' shopping data streams are preprocessed and event patterns are formulated according to the risk behaviors of online shopping users. Secondly, Esper identifies the users' shopping behaviors through the event pattern languages. Finally, the risk events are sent to the response processing template for the early warning operation.

Therefore, the main contributions described in this paper mainly include:

- We divide the behaviors according to the role for identifying user risk in online shopping systems.
- We extract behaviors from the original shopping data streams, and then design the corresponding user risk behavior patterns as the basis for logic reasoning.
- We translate user risk behavior patterns into EPL and then identify user risk behaviors in Esper.

The paper is organized as follows: Section II introduces the methods used in the paper. Section III introduces the specific implementation of CEP for online shopping risk identification. Section IV concludes this paper.

II. METHODS

This section introduces the related knowledge of CEP, and then, describes the event patterns of online shopping risk identification.

A. CEP FRAMEWORK FOR ONLINE SHOPPING RISK IDENTIFICATION

The CEP is real-time processing technology that stores data first and processes data later. It has changed the conventional data mining approach, to reduce the processing time of refinement of the data streams. At the same time, the technology provides the specified time-based or length-based query methods. For example, using a time-based window, CEP can create an average shopping amount for a user over a period of time.

The Esper is an open source implementation framework of CEP, which monitors users' shopping information and triggers monitoring when risk event occurs. And the Esper is a rule-based complex event processing engine, which can store the monitored risk information in the database or display it on the user interface.

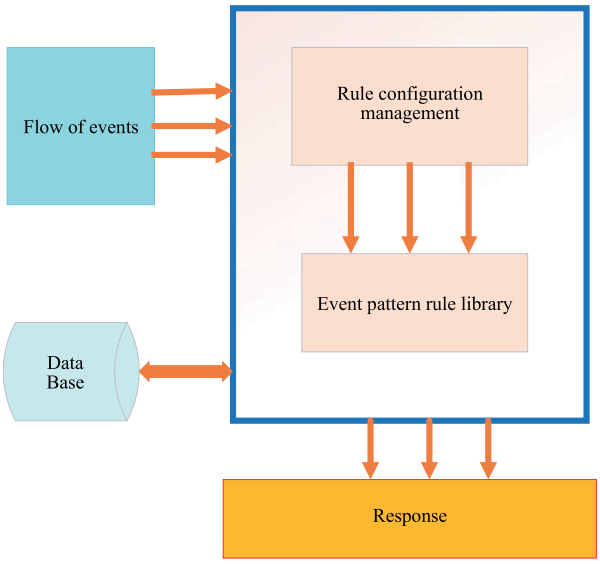

FIGURE 1. CEP framework.

As shown in Figure 1, the event flows of the online shopping process are sent to the Esper engine for processing. The main purpose of this processing is to transform the unstructured data into the structured data so that the system can incorporate and process them.

According to the risk category of shopping information, the system administrators will define corresponding processing rules in advance. There is an event flow that goes into the esper and matches it to the event rules base, and the rule match is successful, then the engine triggers the listeners and finally catches the risk event. At the same time, the risk events are sent into the database or displayed them on the user interface.

B. DEFINITIONS OF COMPLEX EVENT PATTERNS

We firstly obtain data streams of the online shopping process and convert a unstructured data into a structured data by byte filling, filtering and aggregation, etc. ; then use them as the input event streams of the CEP engine; finally, the online shopping risks are identified by the complex event processing platform. Here, according to the typical user malicious behavior patterns [6]–[10], we define the relevant variables, atomic events, and the relevant event patterns in this section.

1) VARIABLE DEFINITION

The variables involved in the paper are as follows:

userID: ID number of a user
money: shopping amount of a user
orderID: order ID of a user
tradeway: payment way of a user
tradeplace: trading address of a user(1-paid, 0-unpaid)
tradetime: shopping time of a trading
tradestate: shopping status of a user

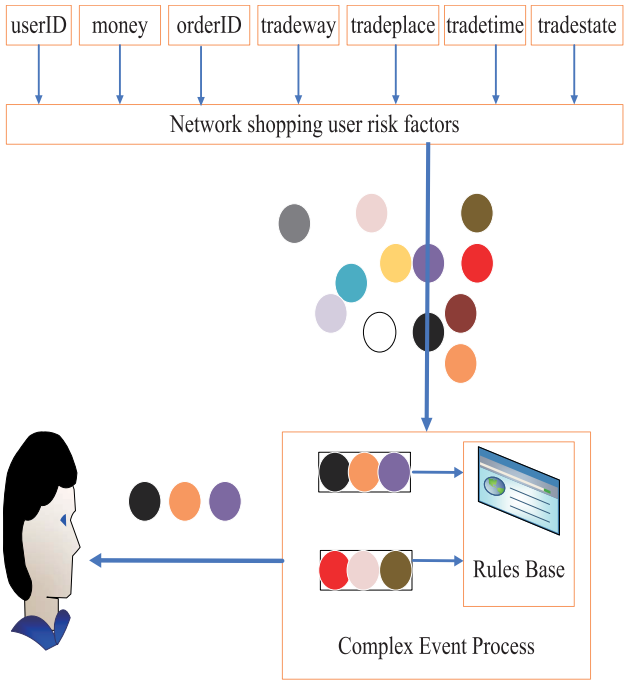

FIGURE 2. Event process.

2) ATOMIC EVENTS DEFINITION

E is the set of events in risk detection of the online shopping process

$E = \{e_n | n = 1, 2, 3...N\}, e_n = \{a, t_1, t_2\}$

TE is the set of events that capture user shopping behavior in real-time.

$TE = \{e_n | n = 1, 2, 3...N\}, e_n = \{a, t_1, t_2\}$

$a = \{userID, money, orderID, tradeway, tradeplace, tradetime, tradestate\}$

3) EVENT PATTERN DEFINITION

This section summarizes that the event pattern languages are defined by considering single account and multiple accounts.

The event patterns of the single account mainly analyze the shopping data streams of the same accounts, and capture the user risk behavior by analyzing the shopping amount, transaction address, and payment way. Mainly includes event pattern 1, 2, 3 and 4.

The event patterns of the multi-user account mainly analyze that a person has different accounts for shopping. As shown in Figure 3. Order replacement means that a user uses different accounts to achieve the purchase of high-priced item for a low price. Mainly includes event pattern 5 and event pattern 6.

The risk behaviors of shopping are characterized by algebraic expressions, which are easy to be read and understood. At the same time, the correctness of semantics is guaranteed. Algebraic expressions construct atomic events for complex events through operators. The threshold t is seted according to the event pattern of the user's online shopping process.

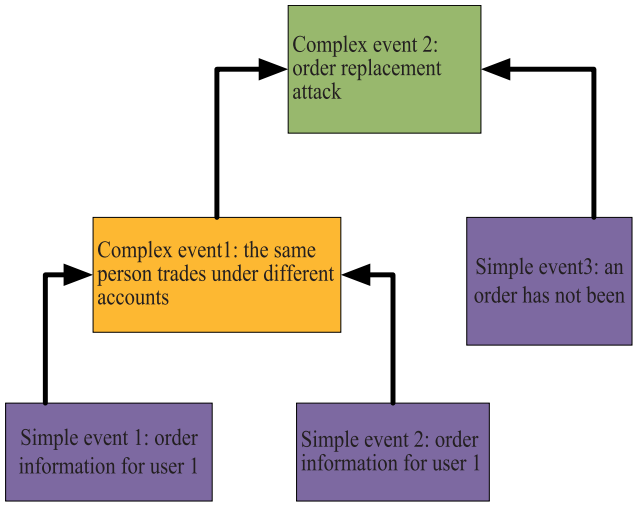

FIGURE 3. Multi-user complex events.

The symbols involved in the algebraic expressions of this paper mainly include: C_i (i>=1) that represents the general name of user shopping events; θ_i (i>=1) that represents specific shopping information of the user; σ is the event selection operator, for the pattern matching of the user event flow, so as to find qualified target events; "$\rightarrow$" is the event sequence operator, such as A $\rightarrow$ B, indicating that B occurs after A occurs.

Event pattern 1. In the time period T, the user m, whose shopping amount exceeds the average amount v, is algebraically expressed as follows:

$C_1 = \sigma\theta(TE)T;$

$\theta = (userID = m) \wedge (money > v) \wedge (tradestate = 1);$

Event pattern 2. For user m, the four consecutive shopping sessions are o_1, o_2, o_3, o_4, and the shopping times are t_1, t_2, t_3, and t_4 respectively. The shopping amount shows an increasing trend and the fourth shopping amount is more than four times of the first shopping amount.The algebraic expression is as follows:

$C_2 \rightarrow C_3;$

$C_3 \rightarrow C_4;$

$C_4 \rightarrow C_5;$

$C_2 = \sigma\theta_1(TE)t_1;$

$C_3 = \sigma\theta_2(TE)t_2;$

$C_4 = \sigma\theta_3(TE)t_3;$

$C_5 = \sigma\theta_4(TE)t_4;$

$\theta_1 = (userID = m) \wedge (tradestate = 1) \wedge (money = m_1) \wedge (tradedate = t_1) \wedge (orderID = o_1);$

$\theta_2 = (userID = m) \wedge (tradestate = 1) \wedge (money = m_2) \wedge (tradedate = t_2) \wedge (orderID = o_2);$

$\theta_3 = (userID = m) \wedge (tradestate = 1) \wedge (money = m_3) \wedge (tradedate = t_3) \wedge (orderID = o_3);$

$\theta_4 = (userID = m) \wedge (tradestate = 1) \wedge (money = m_4) \wedge (tradedate = t_4) \wedge (orderID = o_4);$

$m_1 < m_2 < m_3 < m_4;$

$m_4 >= 4 \times m_1;$

$o_1 < o_2 < o_3 < o_4;$

It is important to note that the operator "→" indicates that there is a sequence among different events and that one event can occur after another.

Event pattern 3. The payment way and shopping status (paid) of the user m at time t_1 are same as the time $t2$, respectively p and 1 (paid). However, the trading addresses of the two are different as p_1 and p_2, respectively. It should be noted that the shopping interval of the same user account is less the threshold t.

The algebraic expression is as follows:

$C_6 \rightarrow C_7$;
$C_6 = \sigma\theta_5(TE)t_1$;
$C_7 = \sigma\theta_6(TE)t_2$;
$\theta_5 = (userID = m) \wedge (tradeplace = p_1) \wedge (tradeway = w) \wedge (tradestate = 1)$;
$\theta_6 = (userID = m) \wedge (tradeplace = p_2) \wedge (tradeway = w) \wedge (tradestate = 1)$;
$t_2 - t_1 < t$;

Event pattern 4. The trading address and shopping status (paid) of the user m at time t_1 are same as the time t_2, where trading address is p and shopping status is paid. However, the payment way of the two are different as w_1 and w_2, respectively. Besides, the time interval between the two signals is less than the threshold value t.

The algebraic expression is as follows:

$C_8 \rightarrow C_9$;
$C_8 = \sigma\theta_7(TE)t_1$;
$C_9 = \sigma\theta_8(TE)t_2$;
$\theta_7 = (userID = m) \wedge (tradeplace = p) \wedge (tradeway = w_1) \wedge (tradestate = 1)$;
$\theta_8 = (userID = m) \wedge (tradeplace = p) \wedge (tradeway = w_2) \wedge (tradestate = 1)$;
$t_2 - t_1 < t$;

Event pattern 5. Obtaining the online shopping information with the user name m_1 at the time t_1, specifically: the order ID o_1, the trading address is p; the online shopping information with the user name m_2 is obtained at time t_2, the order ID o_2, the trading address is p. The time interval between the two signals is less than the threshold value t. Two user accounts (actually the same person) place two orders at the same time, and the algebraic expression is as follows:

$C_{10} \rightarrow C_{11}$;
$C_{10} = \sigma\theta_9(TE)t_1$;
$C_{11} = \sigma\theta_{10}(TE)t_2$;
$\theta_9 = (userID = m_1) \wedge (orderID = o_1) \wedge (tradeplace = p) \wedge (tradestate = 0)$;
$\theta_{10} = (userID = m_2) \wedge (orderID = o_2) \wedge (tradeplace = p) \wedge (tradestate = 0)$;
$t_2 - t_1 < t$;

Event pattern 6. On the basis of the event pattern 5, one of the two orders was not paid, whose algebraic expression is:

$C_{12} \rightarrow C_{13}$;
$C_{12} = \sigma\theta_{11}(TE)t_1$;
$C_{13} = \sigma\theta_{12}(TE)t_2$;
$\theta_{11} = (userID = m_1) \wedge (orderID = o_1) \wedge (tradeplace = p) \wedge (tradestate = 0)$;
$\theta_{12} = (userID = m_2) \wedge (orderID = o_2) \wedge (tradeplace = p) \wedge (tradestate = 1)$;
$t_2 - t_1 < t$;

III. IMPLEMENTATION

The simulation includes 10000 user shopping data for analyzing and validation. The system development environment includes eclipse, Java (JDK version 1.9), Navicat for MySQL, Esper engine 7.0.

The details of specific implementation process are presented, as follows:

A. EVENT PATTERN LANGUAGE

According to the relevant definitions and event descriptions of online shopping risk, the corresponding rules are constructed by Event Processing Language (EPL), a subset of SQL, extended with features for stream processing. Thus, rules can be established in a familiar, easy-to-learn, high-level language.

Corresponding to the event pattern 1 in Section II, the user will be warned if the shopping amount exceeds the average amount within a certain period of time, and the corresponding EPL statement is:

EPL rule1:

```
select avg(money) as a,userID as u, *
from TranEvent.win:time(25 sec)
group by userID
having money>avg(money)
and tradestate=1
```

According to the event pattern 2 in section II, during the monitoring period, the shopping amount of the user presents an increasing state, and the fourth shopping amount is four times more than the first one, so an early warning can be issued, and the corresponding EPL statement is:

EPL rule2:

```
select * from TranEvent
match_recognize (
      measures A as temp1,
                  B as temp2,
                  C as temp3,
                  D as temp4
                  pattern (A B C D)
      define A as A.money IS NOT NULL,
               B as (A.money<B.money),
               C as (B.money< C.money),
               D as (C.money<D.money)
               and D.money>4*A.money)
having temp1.userID=temp2.userID and
        temp2.userID=temp3.userID and
        temp3.userID=temp4.userID and
        temp1.tradestate=1 and
        temp2.tradestate=1 and
        temp3.tradestate=1 and
        temp4.tradestate=1
```

According to the event pattern 3 in section II, when the user changes the shopping address within a short period of time, an early warning will be issued. Our short time here is 10 seconds. The setting of 10s can meet the requirement of the identifying risk behaviors.

The corresponding EPL statement is:

EPL rule3:

```
select * from pattern
[every temp1=TranEvent->temp2=TranEvent
 (temp2.tradeplace! =temp1.tradeplace,
  temp1.userID=temp2.userID,
  temp1.tradeway=temp2.tradeway,
 (temp2.tradetime.getTime()-
     temp1.tradetime.getTime())/1000<10,
  temp1.tradestate=true,
  temp1.tradestate=temp2.tradestate)]
```

Corresponding to the event pattern 4 in Section II, the user's shopping mode changes within a short period of time and an alert will be issued. The corresponding EPL statement is:

EPL rule4:

```
select * from pattern
[every temp1=TranEvent->temp2=TranEvent
(temp2.tradeway!=temp1.tradeway,
 temp1.tradeplace=temp2.tradeplace,
 temp1.userID=temp2.userID,
 (temp2.tradetime.getTime()-
     temp1.tradetime.getTime())/1000<10,
 temp1.tradestate=true,
 temp1.tradestate=temp2.tradestate)]
```

For the event pattern 5 in section II, the same user places two orders at the same time with different accounts, an alert will be issued. The corresponding EPL statement is:

EPL rule5:

```
select * from TranEvent
match_recognize
   (measures A as temp1,
             B as temp2
    pattern (A B)
    define
    A as A.tradeplace IS NOT NULL,
    B as(B.tradeplace=A.tradeplace))
having temp1.userID!=temp2.userID
    and temp1.money=temp2.money
    and (temp2.tradetime.getTime()-
     temp1.tradetime.getTime())/1000<1
```

Corresponding to the event pattern 6 in section II, on the basis of event 5, there is an order that has not been paid; the prompt of order replacement attack will be issued. The corresponding EPL statement is:

EPL rule6:

```
select * from
pattern[every temp1=TranEvent->temp2=TranEvent
   (temp2.tradeway=temp1.tradeway
    temp1.tradeplace=temp2.tradeplace,
    temp1.userID!=temp2.userID,
    (tmep2.tradetime.getTime()
       -temp1.tradetime.getTime())/1000<5,
    temp1.tradestate!=temp2.tradestate)]
```

B. PROGRAM IMPLEMENTION

1) CREATE EVENT OBJECT

When building user online shopping structured data streams, we need to filter the invalid data streams, such as the types of products purchased by users, non-existent user names, and invalid shopping records, etc.

Esper has special data structure conventions for events. The event structures that can be handled by Plain Ordinary Java Object (POJO), MAP, Object Array, Extensible Markup Language (XML). In this paper, we use POJO for structured representation of online shopping user data.

Esper's event object could be a container to send events to Esper's engine. The event class of online shopping risk identification is defined as *TranEvent* (event object), a standard Java class, which mainly include *userID, money, orderID, tradeway, tradeplace, tradetime, and tradestate*.

2) EVENT PROCESSING ENGINE

The Esper engine processor consists of the following parts:

- EPServiceProvier: creating data streams, threads and engines.
- EPStatement: By registering the statement in the engine, the predefined rules are added to the engine.
- UpdateListener: Event listener, an interface provided by Esper, will be triggered when an event comes in and matches a predefined rule.

The event streams of user online shopping are arranged based on the time stamp and sent to the engine for further processing. Then EPLs are constructed to implement flow quires and patterns matching based on the risk that may arise in the shopping process. Once the query is registered, the incoming shopping user data will be processed. Meanwhile it is worth noting that all EPLs must be registered at the beginning. Next, we need to add listeners to receive risk results. Whenever a risk event (complex event) occurs, as long as it matches the rules, a response will be sent to the appropriate listener.

```
Sending TranEvent:Tranevent-User3—Account: 68.0  Tradeway: alipay  Tradeplace: Chengdu   OrderID: 1  Time: 2019-04-11 16:56:32.0  Tradestate: true
Sending TranEvent:Tranevent-User1—Account: 475.0  Tradeway: wechat  Tradeplace: Beijng   OrderID: 1  Time: 2019-04-11 17:07:01.0  Tradestate: true
Sending TranEvent:Tranevent-User4—Account: 123.0  Tradeway: alipay  Tradeplace: Chengdu   OrderID: 1  Time: 2019-04-11 17:07:04.0  Tradestate: true
Sending TranEvent:Tranevent-User2—Account: 56.0  Tradeway: alipay  Tradeplace: Hangzhou   OrderID: 1  Time: 2019-04-11 17:07:03.0  Tradestate: true
Sending TranEvent:Tranevent-User1—Account: 100.0  Tradeway: wechat  Tradeplace: Taiyuan   OrderID: 3  Time: 2019-04-11 18:49:08.0  Tradestate: true
Sending TranEvent:Tranevent-User1—Account: 176.0  Tradeway: wechat  Tradeplace: Chengdu   OrderID: 2  Time: 2019-04-11 17:07:13.0  Tradestate: true
--------------------------------------------------
Change of address within a short period of time
Tranevent-User1—Account: 100.0  Tradeway: wechat  Tradeplace: Taiyuan   OrderID: 3  Time: 2019-04-11 18:49:08.0  Tradestate: true
Tranevent-User1—Account: 176.0  Tradeway: wechat  Tradeplace: Chengdu   OrderID: 2  Time: 2019-04-11 17:07:13.0  Tradestate: true
This indicates that the user is at risk of account theft
--------------------------------------------------
Sending TranEvent:Tranevent-User3—Account: 400.0  Tradeway: alipay  Tradeplace: Beijing   OrderID: 3  Time: 2019-04-11 19:07:08.0  Tradestate: true
```

FIGURE 4. Result of address change risk.

```
Sending TranEvent:Tranevent-User3—Account: 400.0  Tradeway: alipay  Tradeplace: Beijing   OrderID: 3  Time: 2019-04-11 19:07:08.0  Tradestate: true
Sending TranEvent:Tranevent-User2—Account: 100.0  Tradeway: wechat  Tradeplace: Shanghai   OrderID: 2  Time: 2019-04-11 19:17:08.0  Tradestate: true
Sending TranEvent:Tranevent-User3—Account: 400.0  Tradeway: alipay  Tradeplace: Shanghai   OrderID: 3  Time: 2019-04-11 19:58:08.0  Tradestate: true
Sending TranEvent:Tranevent-User1—Account: 536.0  Tradeway: alipay  Tradeplace: Shanghai   OrderID: 2  Time: 2019-04-11 19:59:08.0  Tradestate: true
Sending TranEvent:Tranevent-User8—Account: 123.0  Tradeway: alipay  Tradeplace: Hunan   OrderID: 2  Time: 2019-04-11 20:07:08.0  Tradestate: true
Sending TranEvent:Tranevent-User6—Account: 258.0  Tradeway: alipay  Tradeplace: Hunan   OrderID: 3  Time: 2019-04-11 20:17:21.0  Tradestate: false
Sending TranEvent:Tranevent-User5—Account: 124.0  Tradeway: alipay  Tradeplace: Beijing   OrderID: 1  Time: 2019-04-11 20:23:08.0  Tradestate: false
Sending TranEvent:Tranevent-User4—Account: 64.0  Tradeway: wechat  Tradeplace: Hangzhou   OrderID: 2  Time: 2019-04-11 20:35:24.0  Tradestate: true
Sending TranEvent:Tranevent-User2—Account: 23.0  Tradeway: wechat  Tradeplace: Beijing   OrderID: 3  Time: 2019-04-11 20:37:02.0  Tradestate: true
Sending TranEvent:Tranevent-User2—Account: 368.0  Tradeway: null  Tradeplace: null   OrderID: 0  Time: null  Tradestate: false
Sending TranEvent:Tranevent-User7—Account: 258.0  Tradeway: wechat  Tradeplace: Hunan   OrderID: 2  Time: 2019-04-11 20:44:08.0  Tradestate: false
Sending TranEvent:Tranevent-User9—Account: 258.0  Tradeway: wechat  Tradeplace: Hunan   OrderID: 3  Time: 2019-04-11 20:44:12.0  Tradestate: true
--------------------------------------------------
Two user accounts (the same person) place two orders in succession
--------------------------------------------------
--------------------------------------------------
Tranevent-User7—Account: 258.0  Tradeway: wechat  Tradeplace: Hunan   OrderID: 2  Time: 2019-04-11 20:44:08.0  Tradestate: false
Tranevent-User9—Account: 258.0  Tradeway: wechat  Tradeplace: Hunan   OrderID: 3  Time: 2019-04-11 20:44:12.0  Tradestate: true
Two user accounts (the same person) place two orders in succession
 One order has not been paid
Emergency: order replacement attack
--------------------------------------------------
```

FIGURE 5. Result of order replacement risk.

```
Sending TranEvent:Tranevent-User7—Account: 65.0  Tradeway: alipay  Tradeplace: Beijing   OrderID: 3  Time: 2019-04-11 21:56:32.0  Tradestate: true
Sending TranEvent:Tranevent-User7—Account: 69.0  Tradeway: alipay  Tradeplace: Beijng   OrderID: 4  Time: 2019-04-11 22:07:03.0  Tradestate: true
Sending TranEvent:Tranevent-User7—Account: 70.0  Tradeway: alipay  Tradeplace: Beijng   OrderID: 5  Time: 2019-04-11 22:17:04.0  Tradestate: true
Sending TranEvent:Tranevent-User7—Account: 300.0  Tradeway: alipay  Tradeplace: Beijng   OrderID: 6  Time: 2019-04-11 23:07:08.0  Tradestate: true
--------------------------------------------------
A user's continuous shopping gross continues to grow
Explain that the user has the risk of the account being stolen
--------------------------------------------------
Sending TranEvent:Tranevent-User2—Account: 100.0  Tradeway: wechat  Tradeplace: Taiyuan   OrderID: 3  Time: 2019-04-11 23:19:08.0  Tradestate: true
Sending TranEvent:Tranevent-User1—Account: 176.0  Tradeway: wechat  Tradeplace: Chengdu   OrderID: 2  Time: 2019-04-11 23:37:13.0  Tradestate: true
```

FIGURE 6. Result of the user's amount continuing to rise.

The code snippet to get the engine instance is:

```
Configuration cepConfig=new Configuration ();
cepConfig.addEventType("TranEvent",TranEvent.class.
    getName());
EPServiceProvider cep=EPServiceProviderManager.
    getProvider("myCEPEngine", cepConfig);
```

Esper handles the streaming data by using POJO. Because it is a good way to represent trading events to better predict the risk of the user's trading behaviors. The shopping information that enters the Esper engine mainly includes: user name, trade way, amount and other information. In this paper, The user name has the unique ID for the risk identification of online shopping. Using a unique ID, we can efficiently

```
Sending TranEvent:Tranevent-User3—Account: 144.0  Tradeway: alipay  Tradeplace: Hunan   OrderID: 2  Time: 2019-04-11 19:27:05.0  Tradestate: true
Sending TranEvent:Tranevent-User11—Account: 222.0  Tradeway: alipay  Tradeplace: Hunan   OrderID: 3  Time: 2019-04-11 19:35:08.0  Tradestate: true
Sending TranEvent:Tranevent-User9—Account: 42.0  Tradeway: alipay  Tradeplace: Beijing   OrderID: 1  Time: 2019-04-12 13:01:34.0  Tradestate: true
Sending TranEvent:Tranevent-User10—Account: 635.0  Tradeway: wechat  Tradeplace: Hangzhou   OrderID: 2  Time: 2019-04-12 13:01:36.0  Tradestate: true
Sending TranEvent:Tranevent-User9—Account: 578.0  Tradeway: wechat  Tradeplace: Beijing   OrderID: 3  Time: 2019-04-12 13:01:38.0  Tradestate: true
--------------------------------------------------

Change of tradeway within a short period of time。
Tranevent-User9—Account: 42.0  Tradeway: alipay  Tradeplace: Beijing   OrderID: 1  Time: 2019-04-12 13:01:34.0  Tradestate: true
Tranevent-User9—Account: 578.0  Tradeway: wechat  Tradeplace: Beijing   OrderID: 3  Time: 2019-04-12 13:01:38.0  Tradestate: true
 This indicates that the user is at risk of account theft
--------------------------------------------------
Sending TranEvent:Tranevent-User8—Account: 72.0  Tradeway: alipay  Tradeplace: Hunan   OrderID: 2  Time: 2019-04-12 14:07:05.0  Tradestate: true
Sending TranEvent:Tranevent-User7—Account: 38.0  Tradeway: alipay  Tradeplace: Hunan   OrderID: 3  Time: 2019-04-12 14:07:08.0  Tradestate: true
--------------------------------------------------
Two user accounts (the same person) place two orders in succession
--------------------------------------------------
Sending TranEvent:Tranevent-User9—Account: 566.0  Tradeway: wechat  Tradeplace: Beijing   OrderID: 2  Time: 2019-04-12 14:07:08.0  Tradestate: true
Sending TranEvent:Tranevent-User4—Account: 68.0  Tradeway: wechat  Tradeplace: Nanjing   OrderID: 1  Time: 2019-04-12 14:07:08.0  Tradestate: true
```

FIGURE 7. Result of the way of shopping change risk.

and quickly discover all the shopping orders of this user for analyzing.

The code sends an event to the engine is as follow:

```
EPRuntime cepRT = cep.getEPRuntime();
EPAdministrator cepAm =
    cep.getEPAdministrator();
```

To construct EPLs, we need to register a query in the engine, and the system will analyze the incoming events. The code is as follows:

```
public class WarningEvent {
    public static String getEPL(){
    String ep1=''select avg(money)as a,userID as u,
          * from TranEvent.win:length(5)
          group by userID having money>avg(money)'';
    return ep1;
              }
}
String ep1=WarningEvent.getEPL();
EPStatement cepStetement2=
          cepAm.createEPL(ep1);
```

When the event flow conforms to the EPL rules, the engine constantly displays the results to the listener. The Esper engine provides an *UpdateListener* interface to identify whether the user's shopping behavior matches the predefined event pattern. in the engine, and notify the *UpdateListener* when an event comes in and produces a result.

The code added to the listener is as follow:

```
cepStetement.addListener(
        new CEPListener());
```

We use the *update*() to complete a listener. The engine publishes the risky events in the online shopping process to the listener by using the new event parameters. The code for the listener to obtain the query result is as follow:

```
public class CEPListener implements UpdateListener{
public void update(EventBean[] newEvents, EventBean[]
      oldEvents) {
if(newEvents!=null){
for(int i=0; i<newEvents.length;i++){
       EventBean event=newEvents[i];
       system.out.println(''A trend in which a user shopping
            amount continues to grow.'');
       system.out.println(event.get(''temp1'')+event.get(''
            temp2'')+event.get(''temp3'')+event.get(''temp4
            ''));
       system.out.println(''This indicates that the user is at
            risk of account theft'');
              }
          }
       }
}
```

C. RESULT

This section introduces the partial experiment results in online shopping risk identification. It can be clearly seen from Figure 4 that the shopping information of user 1 satisfies event pattern 3. The shopping address of user 1 changes in a short time, which can timely capture the shopping risk of user 1. At the same time, we can also capture the case of order substitution attack, as shown in Figure 5.

As can be seen from Figure 6, the shopping information of the user 7 presents an increasing trend in a short time, which is satisfied the event mode 2. As can be seen from Figure 7, the shopping way of the user 9 changes in a short time, and the event mode 4 is satisfied, thereby identifying and capturing the abnormal behavior of the user.

IV. CONCLUSION

Online shopping has brought great convenience for users. However, there are also common problems in online shopping: user data will be stolen, and criminals will use user information for consumption or transfer which poses a great threat to the property security of the users. Thus, the security problem has become the main factors affecting the development of online shopping.

In this paper, CEP can better fix the problem of risk identification in the real-time trading by designing relevant user trading scenarios and constructing complex event patterns (EPL rules). By constructing EPL rules and related key codes of Esper, the application technology and development environment of the complex event processing platform can be completely presented, which is conducive to the risk identification in the online shopping systems. The analysis of shopping data flows verifies the feasibility and practicability of the proposed method which potentially improves the security of online shopping environment. The proposed method can play an important role of early real-time warning when risks occur in the online shopping process.

REFERENCES

[1] E. F. Le Saint, "Trusted and unsupervised digital certificate generation using a security token," U.S. Patent 9 331 990, May 3, 2016.

[2] N. D. Byrd, J. Q. Zhang, and E. G. Alger, "Methods and systems for providing a signed digital certificate in real time," U.S. Patent 9 032 204, May 12, 2015.

[3] X. Hu and L. Ma, "A study on the hybrid encryption technology in the security transmission of electronic documents," in *Proc. Int. Conf. Inf. Sci. Manage. Eng.*, vol. 1, Aug. 2010, pp. 60–63.

[4] Y. Wang, C. Hahn, and K. Sutrave, "Mobile payment security, threats, and challenges," in *Proc. 2nd Int. Conf. Mobile Secure Services*, Feb. 2016, pp. 1–5.

[5] Y. Zhang and S. Zhou, "Research on computer network security based on data encryption technology," in *Proc. 2nd Int. Conf. Mater. Sci., Machinery Energy Eng. (MSMEE)*. Paris, France: Atlantis Press, 2017, pp. 1383–1387.

[6] W. Yang, Y. Zhang, J. Li, H. Liu, Q. Wang, Y. Zhang, and D. Gu, "Show me the money! finding flawed implementations of third-party in-app payment in Android apps," in *Proc. NDSS*, Nov. 2017, pp. 1–15.

[7] R. Wang, S. Chen, X. Wang, and S. Qadeer, "How to shop for free Online—Security analysis of cashier-as-a-service based Web stores," in *Proc. IEEE Symp. Secur. Privacy*, May 2011, pp. 465–480.

[8] E. Y. Chen, S. Chen, S. Qadeer, and R. Wang, "Securing multiparty Online services via certification of symbolic transactions," in *Proc. IEEE Symp. Secur. Privacy*, May 2015, pp. 833–849.

[9] F. Sun, L. Xu, and Z. Su, "Detecting logic vulnerabilities in e-commerce applications," in *Proc. NDSS*, Feb. 2014, pp. 23–26.

[10] S. Wen, Y. Xue, J. Xu, L.-Y. Yuan, W.-L. Song, H.-J. Yang, and G.-N. Si, "LOM: Discovering logic flaws within mongodb-based Web applications," *Int. J. Autom. Comput.*, vol. 14, no. 1, pp. 106–118, Feb. 2017.

[11] A. Beimel, Y. Ishai, and E. Kushilevitz, "Ad hoc PSM protocols: Secure computation without coordination," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Cham, Switzerland: Springer, 2017, pp. 580–608.

[12] K. Jaein, K. Nacwoo, Y. Simkwon, and L. Byungtak, "A study on CEP performance in mobile embedded system," in *Proc. Int. Conf. ICT Converg. (ICTC)*, Oct. 2012, pp. 49–50.

[13] M. Ai, N. Chen, X. Ge, Z. Li, T. Pu, Y. Li, Z. Chen, P. Lin, and W. Wu, "A CEP based ETL method of active distribution network operation monitoring and controlling signal data," in *Proc. CIRED Workshop*, 2016, pp. 1–5.

[14] B. Chinnaiah, "CEP support for detection of application layer attacks," in *Proc. 4th Int. Conf. Comput. Intell. Commun. Technol. (CICT)*, Feb. 2018, pp. 1–5.

[15] S. Nielsen, C. Chambers, and J. Farr, "Systems and methods for complex event processing of vehicle information and image information relating to a vehicle," U.S. Patent 8 560 164, Oct. 15, 2013.

[16] K. Taylor and L. Leidinger, "Ontology-driven complex event processing in heterogeneous sensor networks," in *Proc. Extended Semantic Web Conf.* Berlin, Germany: Springer, 2011, pp. 285–299.

[17] D. Luckham, *The Power of Events*, vol. 204. Reading, MA, USA: Addison-Wesley, 2002.

[18] H. Macià, V. Valero, G. Díaz, J. Boubeta-Puig, and G. Ortiz, "Complex event processing modeling by prioritized colored Petri nets," *IEEE Access*, vol. 4, pp. 7425–7439, 2016.

[19] V. Govindasamy, R. Ganesh, G. Nivash, and S. Shivaraman, "Prediction of events based on Complex Event Processing and Probabilistic Fuzzy Logic," in *Proc. Int. Conf. Comput. Power, Energy, Inf. Commun. (ICCPEIC)*, Apr. 2014, pp. 494–499.

[20] S. Stoa, M. Lindeberg, and V. Goebel, "Online analysis of myocardial ischemia from medical sensor data streams with Esper," in *Proc. 1st Int. Symp. Appl. Sci. Biomed. Commun. Technol.*, Oct. 2008, pp. 1–5.

[21] R. Bruns, J. Dunkel, H. Masbruch, and S. Stipkovic, "Intelligent M2M: Complex event processing for machine-to-machine communication," *Expert Syst. Appl.*, vol. 42, no. 3, pp. 1235–1246, Feb. 2015.

[22] N. Zacheilas, N. Zygouras, N. Panagiotou, V. Kalogeraki, and D. Gunopulos, "Dynamic load balancing techniques for distributed complex event processing systems," in *Proc. IFIP Int. Conf. Distrib. Appl. Interoperable Syst.* Cham, Switzerland: Springer, 2016, pp. 174–188.

[23] A. Adi, D. Botzer, G. Nechushtai, and G. Sharon, "Complex event processing for financial services," in *Proc. IEEE Services Comput. Workshops*, Sep. 2006, pp. 7–12.

[24] R. S. Hoefelmeyer, C. N. Dang, and A. Arch-Espigares, "Method and system for detecting fraud based on financial records," U.S. Patent Appl. 11 872 490, Apr. 16, 2009.

[25] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 DARPA off-line intrusion detection evaluation," *Comput. Netw.*, vol. 34, no. 4, pp. 579–595, 2000.

[26] G. Marrocco, E. Di Giampaolo, and R. Aliberti, "Estimation of UHF RFID reading regions in real environments," *IEEE Antennas Propag. Mag.*, vol. 51, no. 6, pp. 44–57, Dec. 2009.

[27] W. Yao, C.-H. Chu, and Z. Li, "Leveraging complex event processing for smart hospitals using RFID," *J. Netw. Comput. Appl.*, vol. 34, no. 3, pp. 799–810, May 2011.

[28] N. Mehdiyev, J. Krumeich, D. Enke, D. Werth, and P. Loos, "Determination of rule patterns in complex event processing using machine learning techniques," *Procedia Comput. Sci.*, vol. 61, pp. 395–401, Jan. 2015.

[29] R. Bhargavi, V. Vaidehi, P. T. V. Bhuvaneswari, P. Balamurali, and G. Chandra, "Complex event processing for object tracking in wireless sensor networks," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. Intell. Agent Technol.*, vol. 3, Aug./Sep. 2010, pp. 211–214.

ZHUOJING MA is currently pursuing the master's degree with the School of Computer Science, Shaanxi Normal University, Xi'an, China. Her research interests include Petri nets theory, formal modeling of online transactions, and complex event processing.

WANGYANG YU received the Ph.D. degree from Tongji University, Shanghai, China, in 2014. He is currently an Associate Professor with the School of Computer Science, Shaanxi Normal University, Xi'an, China. His research interests include the theory of Petri nets, formal methods in software engineering, and trustworthy software.

XIAOJUN ZHAI (M'19) received the Ph.D. degree from the University of Hertfordshire, U.K., in 2013. He is currently a Lecturer with the Embedded Intelligent Systems Laboratory, University of Essex. He has authored/coauthored over 60 scientific articles in international journals and conference proceedings. His research interests mainly include the design and implementation of the digital image and signal processing algorithms, custom computing using FPGAs, embedded systems, and hardware/software co-design. He is a member of BCS and a Fellow of HEA.

MENGHAN JIA is currently pursuing the master's degree with the School of Computer Science, Shangxi Normal University. Her research interests include Petri net theory, social networks, and task preemption scheduling.