ВЕСТНИК ТОМСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА
2014 Управление, вычислительная техника и информатика № 1 (26)

УДК 004.056.53

П.И. Банокин, В.Н. Вичугов

РЕАЛИЗАЦИЯ ПРОГРАММНОЙ СИСТЕМЫ ДЛЯ ПРЕДОТВРАЩЕНИЯ ВНУТРЕННИХ УТЕЧЕК КОРПОРАТИВНЫХ ДАННЫХ

Рассматриваются подходы к созданию программных систем для предотвращения внутренних утечек данных. Автор представляет две архитектуры программных систем. Первая архитектура применима для клиент-серверных программных приложений и использует драйвер СУБД промежуточного уровня. Вторая архитектура служит для защиты многоуровневых программных систем и использует агенты по сбору статистики. В качестве метода предотвращения утечек данных оба решения применяют анализ поведения пользователей программных приложений. В статье описаны форматы и способы хранения статистических данных о действиях пользователей и их профилей.

Ключевые слова: информационная безопасность; утечки данных; внутренние утечки данных; поведение пользователей; СУБД; программная система для предотвращения утечек данных.

Доступные статистические данные свидетельствуют о том, что количество случаев утечек данных ежегодно увеличивается [1]. В качестве средств предотвращения внутренних утечек данных используются аппаратные и программные решения. Аппаратные средства часто реализуются как вычислительное устройство, подключенное к корпоративной сети для исследования трафика с помощью технологии глубокого исследования сетевых пакетов (*deep packet inspection*) [2]. Распространенными методами в системах предотвращения утечек данных являются статистический анализ и лингвистический анализ.

Использование сетевых экранов, своевременное обновление программного обеспечения, установка прав доступа – эффективные меры для снижения рисков внешних утечек данных. Значительно труднее выявить и предотвратить хищения корпоративной информации, совершенные в результате умышленных действий сотрудников предприятия [3]. Информация о случаях внутренней утечки данных несет большие репутационные риски, поэтому значительная ее часть не присутствует в официальной статистике [4].

Сотрудников предприятия можно условно разделить на две категории: универсальные сотрудники (директоры, руководители и другие лица с широкими полномочиями) и сотрудники с фиксированным перечнем обязанностей (специалисты). Именно для второй группы сотрудников могут быть созданы поведенческие профили. Ежедневно в течение рабочего дня такие сотрудники пользуются разными источниками данных, формируют отчеты, обращаются к справочникам, создают новые записи. Поведение пользователя меняется в случае целенаправленной выгрузки данных. Частота обращения к определенным источникам данных, количество просмотров записей определенной категории и другие параметры могут измениться. Архитектуры программной системы, представленные далее, могут уменьшить риски внутренних утечек данных с помощью анализа поведения пользователей программного обеспечения.

Решение проблемы

Представленные далее архитектуры имеют общий компонент «Обработчик статистики» и отличаются способом сбора информации о действиях пользователей. «Обработчик статистики» хранит статистические данные, получаемые при выполнении каждого значимого действия пользователя или его запроса к источнику данных; на основании собранной ранее статистики определяет безопасность текущего действия пользователя.

Первая архитектура ориентирована на защиту клиент-серверных (клиент-СУБД) приложений (рис. 1). Для сбора статистики о поведении пользователей в данной архитектуре используется ODBC-драйвер промежуточного уровня. Данный компонент зависит от платформы и используемой СУБД, является поставщиком статистических данных и ответствен за запрет выполнения подозрительных запросов (рис. 2).

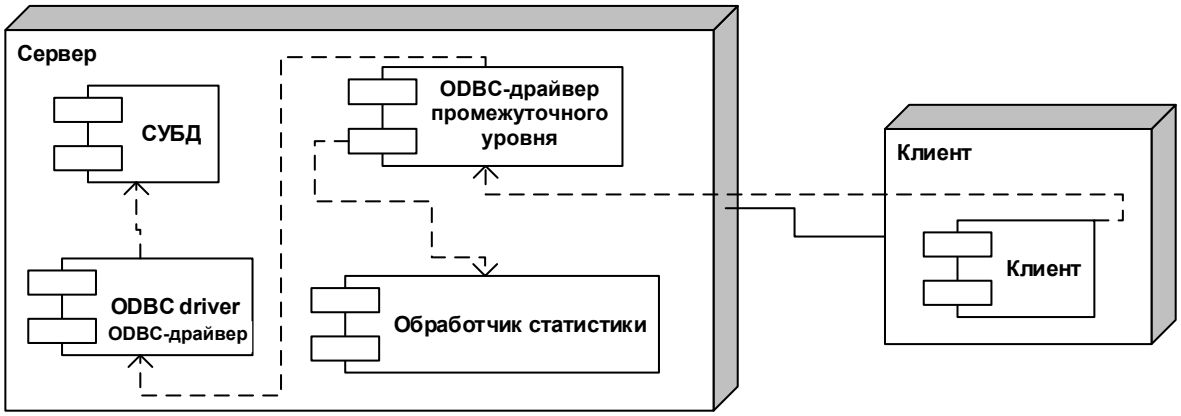

Рис. 1. Базовая архитектура №1

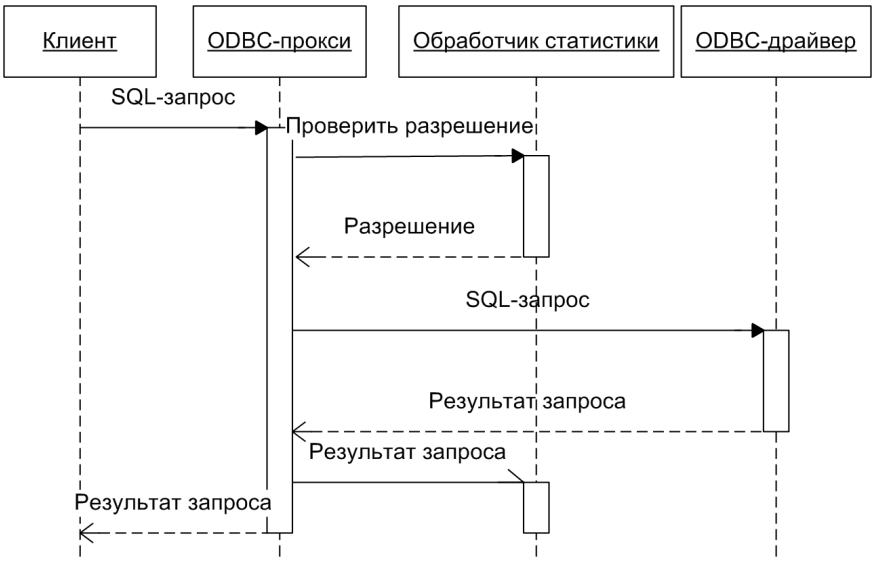

Рис. 2. Схема выполнения запроса через промежуточный ODBC-драйвер

Промежуточный ODBC-драйвер позволяет полностью контролировать запросы к источнику данных и прерывать их в случае необходимости. В многоуровневой программной системе авторизацию пользователей проводит сервер приложений. В этом случае с помощью драйвера промежуточного уровня невозможно идентифицировать пользователя ввиду того, что обращение к базе данных происходит от учетной записи сервера приложений. Также современные системы программных приложений обращаются к нескольким источникам данных, в том числе к веб-сервисам. Второе архитектурное решение ориентировано на защиту многоуровневых систем программных приложений (см. рис. 3). В данной архитектуре отслеживаются значимые действия пользователей, а не отдельные запросы к базе данных. Необходимость разработки дополнений к используемым информационным системам на предприятии является недостатком этой архитектуры. На диаграмме компонентов (рис. 3) также присутствует компонент «Инструмент администрирования», являющийся веб-приложением. Данное веб-приложение предоставляет графический интерфейс для управления настройками обработчика статистики, просмотра уведомлений и профилей пользователей.

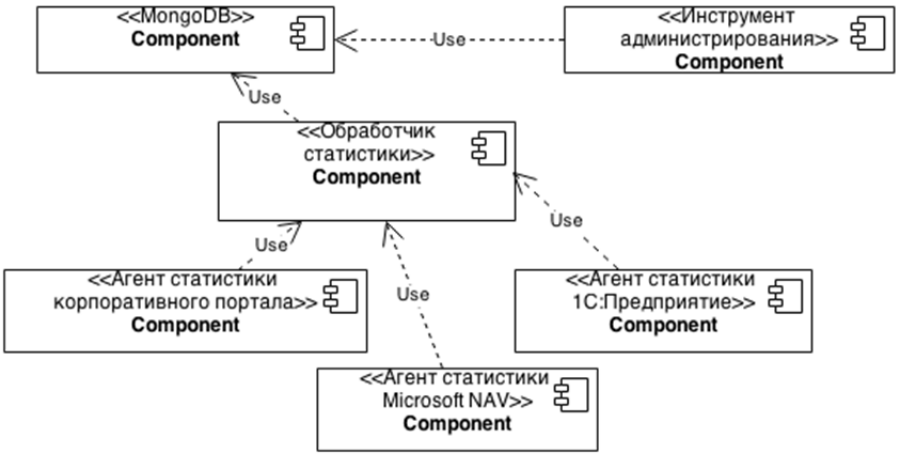

Рис. 3. Базовая архитектура № 2

Обе архитектуры используют два хранилища данных: хранилище статистики действий пользователей и хранилище профилей пользователей. Диаграмма потоков данных приведена на рис. 4.

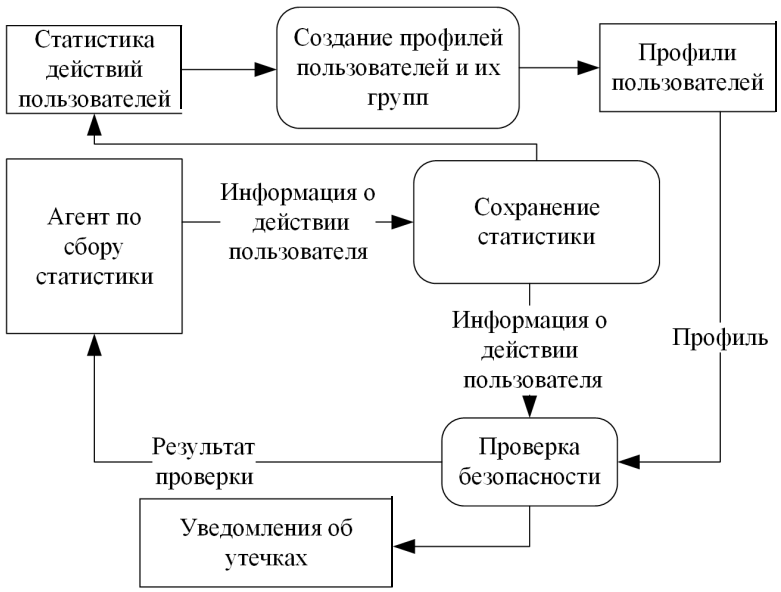

Рис. 4. Диаграмма потоков данных

История действий пользователей и сформированные профили пользовалей хранятся в документно-ориентированной БД *MongoDB*. Также в этом хранилище хранятся настройки обработчика статистики и уведомления. Формат хранения данных – *JSON*. Данные о действиях пользователей достаточно объемны, но имеют простую схему. Их хранение в документно-ориентированном хранилище может обеспечить бóльшую масштабируемость и производительность в сравнении с реляционной базой данных. Информация о действии пользователя отправляется обработчику статистики асинхронным способом и включает следующие поля:

1. Имя пользователя (логин), роль пользователя.

2. Тип инструмента. Программные приложения предоставляют широкий набор инструментов. Справочники, средства составления отчетов, обработки, специальные графические интерфейсы (интерфейс наблюдения за статистикой продаж, интерфейс сотрудника *call*-центра, интерфейс "Рабочее место кассира"). Тип инструмента передается в виде строковой константы из заранее определенного перечня.

3. Название программного приложения.

4. Название инструмента в составе программного приложения. Например, если пользователь открывает справочник "Контрагенты", то в качестве названия передается значение "Контрагенты".

5. Список коллекций данных, к которым обратился пользователь. Коллекция – произвольный набор данных одного типа независимо от используемого хранилища. При использовании реляционного хранилища названием коллекции является название таблицы или представления.

6. Тип выполненной операции. Операции условно разделены на пять типов: просмотр коллекции (списка, таблицы), просмотр детальной информации об отдельной записи, создание, удаление, редактирование. Данный параметр также передается в виде строковой константы.

7. Данные или их часть, которые получил или сохранил пользователь в виде *JSON*-объекта.

8. Категория данных.

Следующая *JSON*-строка является примером статистики о действии пользователя: {"user": "Svetlana", "tool": "prebuild_report", tool_name:"daily sales report", "entities": ["sales", "employees", "shops"], "operation":"view_col", "app": "1C_Retail", "data":{"category": "null", "date": "06-03-2014", search_query: "Иван Федоров" }}.

Собранная информация о действиях пользователей является входными данными для процесса построения и обновления профилей пользователей, выполняемого обработчиком статистики по расписанию. Профиль пользователя представляет собой описание ежедневной работы с источниками. Для каждого пользователя создаются поведенческие профили за разные временные промежутки (профиль предыдущего дня, профиль последних 30 дней и др.). Выявление факта утечки данных происходит путем сравнения параметров текущей работы пользователя с его профилями (рис. 5). Несмотря на то, что ложные уведомления о возможных утечках данных неизбежны, использование нескольких профилей для каждого пользователя может уменьшить их количество. Итоговая оценка безопасности действия пользователя формируется в результате сложения независимых оценок с каждым из профилей.

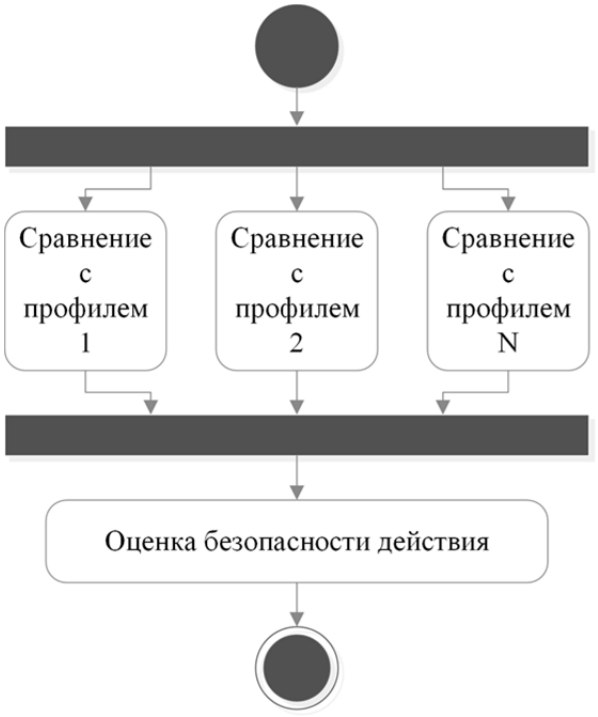

Рис. 5. Сравнение с профилями

Заключение

Областью использования программной системы, описанной в статье, являются предприятия, работающие с персональной информацией, и предприятия, работающие со сведениями, представля-

ющими коммерческую или государственную тайну. Предполагается, что использование системы предотвращения утечек данных целесообразно при количестве пользователей, превышающем 100 человек. Статистика о поведении пользователей, собираемая представленным программным решением, может быть использована для анализа производительности труда сотрудника предприятия или для анализа использования рабочего времени сотрудником. Благодаря наличию данных об используемых инструментах также можно выявить, насколько те или иные программные приложения или инструменты программных приложений востребованы и наиболее часто используются сотрудниками предприятия.

ЛИТЕРАТУРА

1. *Data* Loss Statistics // Data Loss DB. URL: http://datalossdb.org/statistics
2. *Brandel M.* Data Loss Prevention Dos and Don'ts. URL: http://www.csoonline.com/article/221272/data-loss-prevention-dos-and-don-ts
3. *Data* Leakage Worldwide: The High Cost of Insider Threats // Cisco. URL: http://www.cisco.com/en/US/solutions/ collateral/ns170/ns896/ns895/white_paper_c11-506224.pdf
4. *Data* breach investigations report 2012 // Verizon Enterprise Solutions Worldwide Site. URL: http://www.verizonbusiness.com/resources/reports/rp_data-breach-investigations-report-2012_en_xg.pdf

Банокин Павел Иванович. E-mail: pavel805@gmail.com
Вичугов Владимир Николаевич. E-mail: vlad@aics.ru
Томский политехнический университет.

Поступила в редакцию 7 ноября 2013 г.

Banokin Pavel I., Vichugov Vladimir N. (Tomsk Polytechnic University, Russian Federation).
Implementation of software system for prevention of internal data leaks.
Keywords: information security; data leaks; DBMS; software users' behavior; software system.

In this article approaches on setting up software systems for prevention of internal data leaks is considered. It is stated that dishonest employee's behavior is different from behavior of other employees. One can detect the different parameters of user's behavior, the list of used often data sources, software instruments, performed often operations with data.

Two approaches to development of the software system are proposed: the software system based on intermediate ODBC-driver, and the software system based on the user behavior of statistics agents. Generally, statistics agents are software components or plug-ins which send information about every performed action of an user. Both these architectures share the statistics processor component. The statistics processor is responsible for three main tasks: statistics storage, statistics transformation into users' profiles, and data leak detection. The first architecture is applied to the client-server (client-DBMS) software while the second architecture supports a wide range of multi-layer corporate software. Moreover, many of the client-servers software treat data not only from relational databases, but also from document-oriented databases, web-services, files and other sources. Second architecture gives an opportunity to collect statistics about the wider range of users' actions while the first architecture monitors data problems only. The main drawback of the second architecture is the necessity of the plug-in development for every application which requires protection.

The formats and storages of users' behavior and users' profiles are described. The statistics entries, users' profiles, statistics processor settings, and notifications are stored as the JSON-documents for the simplicity and faster processing. The documents are stored in the MongoDB database. The statistics entries contain information about the user and the role name, software application name, software tool name, operation type. Periodically users' profiles are built or updated from statistical data. A profile contains the description of user's behavior. This description includes information how frequently one or another software application or tool is used, whether a user usually adds new data entries or read them, what data categories are usual for daily activity. Users' profiles are built not only for each of the users, but also for the different time periods. Thus, a user could have several profiles for the different time periods like previous month, last 30 days, last week, etc. Each of the activities is assessed as relevant or irrelevant to the profile. Nevertheless, false alarms are inevitable. We propose to use several profiles for every user as an approach for decrease of false alarms. The final evaluation of user's safety is formed as the composition result of independent assessments with each of the profiles.

REFERENCES

1. Data Loss Statistics. Data Loss DB. URL: http://datalossdb.org/statistics
2. Brandel M. Data Loss Prevention Dos and Don'ts. URL: www.csoonline.com/article/221272/data-loss-prevention-dos-and-don-ts
3. Data Leakage Worldwide: The High Cost of Insider Threats. Cisco. URL: http://www.cisco.com/en/US/solutions/ collateral/ns170/ns896/ns895/white_paper_c11-506224.pdf
4. Data breach investigations report. Verizon Enterprise Solutions Worldwide Site. (2012). URL: http://www.verizonbusiness.com/resources/reports/rp_data-breach-investigations-report-2012_en_xg.pdf