УДК 004

М. А. Поляничко
Петербургский государственный университет путей сообщения

АРХИТЕКТУРА СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ОБНАРУЖЕНИЯ И РАЗРЕШЕНИЯ КОНФЛИКТОВ ПРОГРАММНЫХ СРЕДСТВ ЗАЩИТЫ ИНФОРМАЦИИ

Описывается принцип функционирования автоматизированной системы обнаружения и разрешения конфликтов программных средств защиты информации и предлагается структура, позволяющая реализовать основную функциональность и обеспечить хранение информации о конфликтах, правилах их обнаружения и разрешения. Приведена структура базы данных и состав программных средств.

программные средства защиты информации, конфликтное взаимодействие, обнаружение и разрешение конфликтов.

Введение

Большинство современных компьютерных приложений обладают высокой ресурсоемкостью. Особенно это относится к программным средствам защиты информации (СЗИ), поскольку они интенсивно взаимодействуют со многими аппаратными и программными компонентами компьютерных систем и используют их во время комплексных системных проверок и других операций по обеспечению информационной безопасности. Несмотря на то, что разработчики постоянно улучшают работу СЗИ и процесс их взаимодействия с компьютерными системами, потребность таких программ в системных ресурсах продолжает неуклонно расти вследствие увеличения сложности и растущего количества вредоносного программного обеспечения, такого как вирусы и прочие типы.

Программные СЗИ могут вмешиваться в работу других программ. Например, антивирус может принять действие другой программной СЗИ за вредоносное действие и впоследствии ограничить доступ конфликтующего приложения к системным ресурсам, таким как память, системный реестр и прочие. Из-за этого вмешательства конфликтующие программы не могут выполняться должным образом и повторяют свои запросы к системным ресурсам, что еще сильнее снижает производительность системы и ослабляет её защищенность.

Обеспечение неконфликтного взаимодействия СЗИ является важной задачей для обеспечения комплексной безопасности системы и повышения производительности. Предлагаемая архитектура реализует метод обнаружения и разрешения конфликтов, в основе которого лежит применение правил реагирования на изменение показателей работы системы.

1 Архитектура системы обнаружения конфликтов

Для обнаружения и разрешения возникающих конфликтов взаимодействия используется информация, поступающая непосредственно от операционной системы.

В состав системы обнаружения и разрешения конфликтов входят:

– локальный программный агент – сборщик информации о системной конфигурации, наблюдает за показателями использования ресурсов и вносит изменения в конфигурацию приложений, запущенных на ПК;

– механизм принятия решений – средство определения критических уровней использования системных ресурсов и конфликтного взаимодействия приложений;

– база данных – содержит правила определения критических уровней и правила разрешения конфликтов, основанные на нечеткой логике.

На рисунке 1 показана схема предлагаемой системы. Возможно несколько вариантов реализации, как с использованием удаленного доступа к механизму разрешения конфликтов через сеть, так и установка всей системы на локальном компьютере.

2 Локальный программный агент

Локальный программный агент взаимодействует с операционной системой (в приведенной реализации – Windows) и обеспечивает сбор необходимых данных для последующего анализа. В качестве источников данных используются встроенные службы «Монитор производительности (Performance Monitor)» и «Журнал событий (Event Viewer)». Использование этих служб позволяет получить наиболее достоверные данные и исключить дополнительную нагрузку на операционную систему [1]. В текущей реализации программный агент следит за использованием ресурсов оперативной памяти, процессора и жесткого диска. В качестве базиса выбрано 47 показателей [2]. Каждый показатель принадлежит к одному из пяти типов [3]. Описание и формула, используемая для вычисления, приведены в таблицах 1–5.

Агент также следит за составом установленных СЗИ, связанными с ними библиотеками, ключами реестра и конфигурационными файлами. Информацию об установленных программах агент получает через объекты WMI.

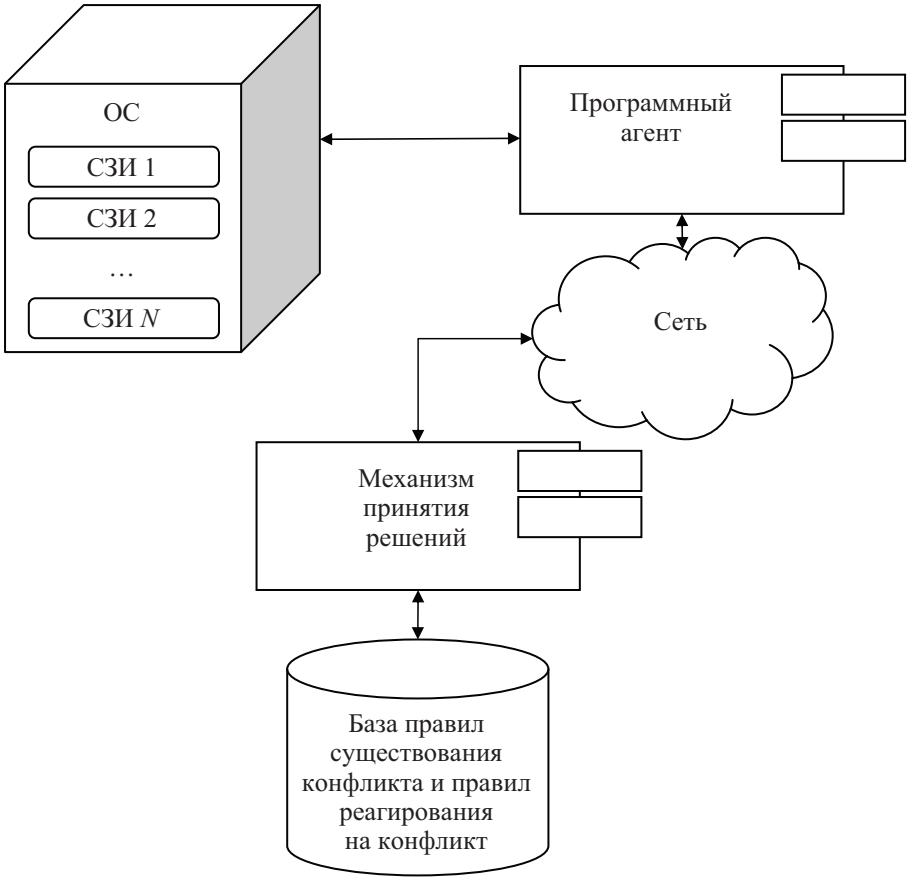

Рис. 1. Схема системы разрешения конфликтов

ТАБЛИЦА 1. Тип показателя PERF_100NSEC_TIMER

Описание	Показывает время активности компонента как процент от всего прошедшего времени базового интервала. Он измеряет время интервалами по 1 мс. Счетчики этого типа предназначены для измерения утилизации одного компонента в данный момент времени
Тип	Значение в процентах
Формула	$\frac{(N_1 - N_0)}{(D_1 - D_0)} \cdot 100,$ где знаменатель (D) представляет прошедшее полное время базового интервала, а нумератор (N) – части базового интервала, в течение которого исследуемые компоненты были активны
Среднее значение	$\frac{(N_x - N_0)}{(D_x - D_0)} \cdot 100$

ТАБЛИЦА 2. Тип показателя PERF_RAW_FRACTION

Описание	Показывает отношение подмножества к его полному множеству в процентах. Например, для сравнения числа байт, используемых на диске, и его общего размера в байтах. Счетчики этого типа показывают текущее процентное отношение, а не среднее значение за какое-то время
Тип	Текущее значение, значение в процентах
Формула	$\frac{N_0}{D_0},$ где D – измеряемый атрибут, N – один из компонентов этого атрибута
Среднее значение	$\sum \frac{N}{D} / x$

ТАБЛИЦА 3. Тип показателя PERF_COUNTER_COUNTER

Описание	Показывает среднее число операций, законченных в течение одной секунды – базового интервала. Счетчики этого типа измеряют время импульсами сигналов системного таймера. F – переменная, которая представляет число импульсов сигнала таймера в секунду. Значение F разлагается в уравнении так, чтобы результат мог быть отображен в секундах
Тип	Различие
Формула	$\frac{(N_1 - N_0) \cdot F}{(D_1 - D_0)},$ где нумератор (N) представляет число операций, выполненных в течение последнего базового интервала; знаменатель (D) – число импульсов таймера за последний базовый интервала; F – частота импульсов таймера
Среднее значение	$\frac{(N_x - N_0) \cdot F}{(D_x - D_0)}$

ТАБЛИЦА 4. Тип показателя PERF_COUNTER_RAWCOUNT

Описание	Показывает только последнее полученное значение, не показывает среднее значение
Тип	Текущее значение
Формула	Не используется. Исходные данные показываются так, как они были собраны
Среднее значение	$\frac{\sum N}{x}$

ТАБЛИЦА 5. Тип показателя PERF_SAMPLE_FRACTION

Описание	Показывает среднее отношение попаданий ко всем операциям в течение последних двух базовых интервалов
Тип	Процент; процент попаданий
Формула	$\frac{(N_1 - N_0)}{(D_1 - D_0)}$, где нумератор ($N$) представляет число успешных операций в течение последнего базового интервала, знаменатель (D) – изменение в числе всех операций (измеряемого типа), законченных в течение этого интервала. Счетчики Hit% лучшие наблюдать через System Monitor в виде диаграмм. Попадания часто появляются в виде коротких пиков, которые можно не заметить в текстовом представлении. Также среднее число, отображенное для Hit% в строке состояния в виде диаграммы, не соответствует среднему числу, отображенному в текстовом представлении, потому что они рассчитываются по-другому. В виде диаграммы Hit% является средним числом всех изменений в счетчике в течение исследуемого интервала; в текстовом представлении это будет среднее число различий между первым и последним измерениями в течение базового интервала
Среднее значение	$\frac{(N_x - N_0)}{(D_x - D_0)}$

3 Структура базы данных

База данных используется для хранения информации о конфликтах, правил обнаружения конфликтов и правил реагирования на конфликт. В таблице 6 описаны сущности предлагаемой структуры реляционной базы данных.

4 Механизм разрешения конфликта

Механизм разрешения конфликта обрабатывает информацию из базы данных и управляет программным агентом. Результатом работы модуля является решение о внесении изменений в настройку СЗИ для обеспечения неконфликтного взаимодействия, основанного на правилах, которые хранятся в базе.

Правила реагирования представляют собой активационные функции нечетких множеств, граничными значениями для которых являются данные о ресурсах системы, а значения чувствительности задаются пользователем. При описании правил применяются пять уровней серьезности конфликта. Используется стандартная классификация событий для систем семейства Windows [1], которая приведена в таблице 7.

ТАБЛИЦА 6. Структура базы данных

Таблица	Описание
Application	Предназначена для назначения каждому приложению уникального номера – идентификатора
AppVersion	Потомок таблицы Application. В таблице перечислена информация для каждого сравниваемого приложения. Если таблица Application позволяет назначить ключевой номер каждому сравниваемому приложению, то AppVersion позволяет назначить ключ для каждой версии приложения. В приложении может не быть пакетов или может быть несколько пакетов
AddPath	Потомок таблицы AppVersion. В ней содержится подробная информация обо всех путях для всех сравниваемых приложений. Чтобы в данной таблице была запись о пути, она должна принадлежать пакету (AppVersion). Конфликтующие пути требуют указания конфликтного уровня в поле Conflict таблицы
Config	Также является потомком таблицы AppVersion. Она содержит подробную информацию обо всех изменениях файла Config.sys (файл конфигурирования операционных систем семейства Windows, текстовый файл, содержащий директивы настройки системы и команды загрузки драйверов)
IniChange	Потомок таблицы AppVersion, в ней хранятся все изменения INI-файлов. Каждая запись в данной таблице представляет собой одну запись в одной секции одного INI-файла. INI-файлы необходимы для функционирования программ, следовательно, любое их изменение, которое привело к конфликту, должно иметь указание в поле Conflict для данного файла
Service	Таблица также является потомком таблицы AppVersion. Она содержит подробную информацию обо всех службах, установленных как часть пакета
File	Является еще одним потомком таблицы AppVersion. В ней содержится подробная информация обо всех файлах, установленных как часть конкретной версии приложения, известной как пакет
FileString	Потомок таблицы File, является централизованной иерархической базой данных строк, которые являются путями назначения для файлов
FileAppMap	Является потомком таблицы File, позволяет соотнести файловые строки с приложениями
FileConflict	Является потомком таблицы File. Файл должен существовать в таблице File и иметь номер больше нуля в поле Conflict таблицы File перед попаданием в таблицу FileConflict.Таблица FileConflict содержит поле под названием CLevel (конфликтный уровень), которое указывает на серьезность конфликта
SrcAppMap	Потомок таблицы File, содержит подробную информацию, помогающую определить соответствие между исходными строками файла и источника
SrcString	Еще один потомок таблицы File, является централизованной иерархической базой данных строк. Каждой секции пути к файлу назначено числовое значение
System	Является потомком таблицы AppVersion. В ней находится подробная информация обо всех файлах драйвера устройства. Чтобы в данной таблице существовал файл, он сначала должен принадлежать пакету (расположенному в таблице AppVersion)

Окончание табл. 6

Таблица	Описание
Shortcut	Потомок таблицы AppVersion, содержит все ярлыки и информацию о ярлыках для установленных приложений. В данной таблице не может существовать файл, пока он не принадлежит пакету
Registry	Потомок таблицы Package. В ней находится подробная информация о всех записях реестра в пакете
RegString	Предоставляет централизованное хранилище имен ключей реестра
RegAppMap	Содержит все строки реестра в обоих сравниваемых приложениях. Кодирование путей реестра обеспечивает их эффективное хранение
RegConflict	Является потомком таблицы Registry, поэтому имеет три указателя на предка: AppKey, PackKey и RegKey. Таблица RegConflict перечисляет все записи реестра, которые имеют конфликты
RegValue	Содержит блоки по 255 символов для значений реестра, длина которых больше 255 символов. Первые 255 символов хранятся в таблице Registry, а последующие символы – в данной таблице. Это позволяет хранить значения реестра неопределенного размера в поле для общих символов самого малого размера
RegKeyName	Содержит названия ключей реестра, которые задает пользователь. Данное значение позволяет быстро идентифицировать файлы реестра, тем самым повысить скорость разрешения конфликтов и снизить количество использованных строк
DriverServ	Потомок таблицы AppVersion, перечисляет все драйверы, источники данных и их управляющие приложения
DriverAtt	Является потомком таблицы DriverServ. В таблице DriverAtt показаны все атрибуты и их значения для драйверов, указанных в таблице DriverServ, также таблица записывает все конфликты между драйверами
DriverConflict	Является потомком таблицы DriverAtt. Таблица DriverConflict содержит все конфликтующие атрибуты драйверов. Атрибут должен существовать в таблице DriverAtt и иметь уникальный номер больше нуля в поле Conflict, чтобы попасть в таблицу DriverConflict
MSIComponent	Потомок таблицы AppVersion. Ее функцией является запись всех компонентов Microsoft Windows Installer в сравниваемых приложениях. Компонент Microsoft Windows Installer является наименьшим уровнем установки Microsoft Windows Installer. Все изменения, которые могут быть внесены в систему, привязаны к данному компоненту [4]
GenConflict	Таблица содержит все нефайловые и нереестровые конфликты. Блок Type позволяет указывать тип файла, вызывающий конфликт, например INI-файлы, Autoexec.bat и пр.
Rule	Таблица хранит логические правила, записанные в лингвистическом виде, пригодном для дальнейшей обработки механизмом разрешения конфликта
SysInfo	Таблица содержит данные о системе и используется для определения пороговых значений производительности
LogMsg	Таблица является служебной для отражения в журнале событий системы разрешения конфликтов

ТАБЛИЦА 7. Типы событий системы

Тип события	Пояснение
Ошибка (Error)	Определяется серьёзная ошибка приложения, например: исполнение приложения прервалось из-за нехватки ресурсов
Предупреждение (Warning)	Этим типом приложение обычно информирует о том, что скоро может возникнуть проблема, например: закончится дисковое пространство
Информация (Information)	Этим типом приложение обычно информирует об успехе какой-либо важной операции, например о старте сервиса
Успешный отчёт (Success Audit)	Сообщает об успехе какой-либо операции доступа, например: пользователь вошёл в систему
Неуспешный отчёт (Fault Audit)	Означает, что произошла какая-то ошибка при доступе к ресурсу, например: пользователь не смог обратиться к сетевому диску

В системе возможна реализация одного из существующих методов разрешения конфликтов, например описанных в [4], [5], [6], но предлагается проведение дополнительного анализа и создание комбинированного метода.

Заключение

Предложенная структура может служить основой для построения программного комплекса обнаружения конфликтов программных СЗИ и их разрешения. В отличие от системы, предложенной в работе [5], имеет более централизованную архитектуру и позволяет работать с показателями производительности.

Применение системы обнаружения конфликтов программных СЗИ позволит повысить эффективность работы системного администратора и уменьшить время, требуемое для выявления наличия конфликтного взаимодействия. Приблизительная оценка времени, затрачиваемого на поиск конфликта, показывает, что применение предлагаемой структуры позволит сэкономить до 30% времени за счет четкого указания области возникновения конфликта.

Библиографический список

1. **M79 Microsoft** Windows Server 2003 : полное руководство / Р. Моримото, К. Гардиньер, М. Ноэл, О. Драуби ; пер. с англ. – 2-е изд. – М. : Изд. дом «Вильямс», 2005. – 1312 с.

2. **MSDN** [Электронный ресурс]. – Режим доступа: http://msdn.microsoft.com/en-us/library/windows/desktop/aa373083(v=vs.85).aspx (Дата обращения 15.09.2012).

3. **MSDN** Counter Type [Электронный ресурс]. – Режим доступа: http://msdn.microsoft.com/ru-ru/library/vstudio/bb156214(v=vs.90).aspx (Дата обращения 15.09.2012).

4. **Method** and system of managing software conflicts in computer system that receive, processing change information to determine which files and chared resources conflict with one another / J.J. McMillan. – Wise Solutions, Inc., Canton, MI (US), Appl. No.: 09/189,559; Nov. 11, 1998.

5. **Adaptive** configuration of conflicting applications / O.V. Zaitsev. – Kaspersky Lab, ZAO, Appl. No.: 10193144.2; 30.11.2010.

6. **Method** and system for avoidance of software conflict / Y. Ding. – International Business Machines Corporation, Armonk, NY (US), Appl. No.: 11/615,898; Dec. 22, 2006.