На правах рукописи

Косых Петр Александрович

РАЗРАБОТКА И ИССЛЕДОВАНИЕ ФАЙЛОВОЙ СИСТЕМЫ СО СЛЕЖЕНИЕМ ЗА ЦЕЛОСТНОСТЬЮ

05.13.11 – математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

05.13.19 – методы и системы защиты информации, информационная безопасность

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата технических наук

Автор: 

Москва – 2006

Работа выполнена в Московском инженерно-физическом институте (государственном университете)

Официальные оппоненты: доктор технических наук
Клыков Александр Викторович,
кандидат технических наук,
доцент Зубарев Игорь Витальевич

Ведущая организация. Московский Научно-исследовательский институт приборной автоматики

Защита диссертации состоится "12" апреля 2006г. в 14 часов 00 минут на заседании диссертационного совета Д 212.130 03 в конференц-зале главного корпуса Московского инженерно-физического института (государственного университета) по адресу: 115409, г Москва, Каширское шоссе, 31.

С диссертацией можно ознакомиться в библиотеке института

Автореферат разослан "3" марта 2006 г.

Ученый секретарь
диссертационного совета
д т н , профессор Шумилов Юрий Юрьевич

2006А
4565

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы. Жизнь современного общества немыслима без повсеместного использования программных систем, связанных с вводом, хранением, обработкой и выводом информации. Всеобщая компьютеризация, помимо очевидных выгод несет с собой и многочисленные проблемы, наиболее сложной из которых является проблема информационной безопасности. Можно выделить следующие причины трудоемкости решения задачи защиты программных систем: все большее отстранение пользователя от процессов обработки данных и передача его полномочий ПО, обладающему некоторой свободой в своих действиях и поэтому не всегда функционирующему так, как предполагает пользователь; появление новых технологий программирования, затрудняющих оценку качества программных продуктов.

Архитектуры современных многозадачных, многопользовательских операционных систем (ОС) удовлетворяют следующим основным требованиям: наличие механизма дискреционного разграничения доступа; изоляция адресных пространств процессов; изоляция кода ядра от злонамеренного воздействия кода пользовательских процессов; невозможность получения пользовательским процессом неочищенной памяти при ее выделении; аудит событий, критических с точки зрения безопасности. Однако выполнение этих и некоторых других требований само по себе не делает программную систему защищенной. Функционирование вредоносных программ (ВП) может свести «на нет» все преимущества ядра и ОС в целом. Полностью возможность попадания ВП в программную систему исключать нельзя, поэтому проблема защиты от ВП актуальна для компьютерных систем (КС) любого уровня защищенности. Необходима следующая поддержка со стороны программной среды: программная система должна быть максимально стойкой к проникновению и действиям ВП; нужно быть готовым к возможным проникновениям ВП и уметь обнаруживать это проникновение; необходимо ПО для борьбы с обнаруженными ВП и восстановления объектов КС от результатов деятельности КВ.

Серьезную угрозу безопасности для программной системы представляют умышленные и случайные деструктивные воздействия. При этом наиболее трудоемкой являтся защита от разрушающих программных воздействий (РПВ). Можно выделить следующие основные причины трудоемкости решения задачи защиты от РПВ: постоянное

РОС. НАЦИОНАЛЬНАЯ
БИБЛИОТЕКА
С.Петербург
09 [illegible] 183

совершенствование компьютерных технологий, появление новых математических методов, расширяющих возможности создателей РПВ; необходимость учета особенностей ОС при разработке средств защиты от РПВ; ограниченные возможности применяемых методов защиты от РПВ; недопустимость наличия дефектов в программном коде средств защиты типа "переполнение буфера", которые дают возможность атакующему подменять алгоритмы функционирования программных средств защиты, получать права администратора, вызывать отказы в обслуживании со стороны средств защиты; необходимость при разработке программных средств защиты обеспечения самозащиты от РПВ, наличия средств самоконтроля целостности, обеспечения гарантированности свойств; недопустимость использование только пассивных методов защиты, при реализации которых нападающая сторона всегда находится в более выигрышном положении, чем защищающаяся сторона; необходимость оперативной реакции со стороны разработчиков средств защиты от РПВ на появление принципиально новых технологий создания ВП.

На данный момент идеология современной многозадачной операционной системы являтся завершенной концепцией. Анализируя ядра самых распространенных ОС, таких как ядра Windows NT / 2000 / XP, Linux и UNIX систем, можно утверждать, что, несмотря на некоторые внешние отличия, все они используют одинаковые парадигмы. В мире ОС имеется две категории сущностей: субъекты - изолированные процессы, запущенные от имени пользователя, и объекты – файлы, принадлежащие структурированной файловой системе и обладающие рядом атрибутов. Эта схема не меняется уже десятки лет, что, учитывая изменчивый мир информационных технологий (ИТ), позволяет сделать вывод о сформированной концепции ОС.

Файловая система это, вообще говоря, база данных особого рода Исторически, в идеологии файловой системы заложен минимум требований к ресурсам при приемлемом удобстве доступа. В то же время с ростом мощности компьютерных систем, становятся очевидными недостатки такой модели: и в смысле удобства работы пользователя и в смысле вопросов защиты информации. В качестве доказательства можно отметить тот факт, что, несмотря на незыблемость технических идей при реализации файловой системы, имеется тенденция сближения концепций БД и файловой системы.

Файловая система является хранилищем информации, поэтому одним из базовых и актуальных вопросов является задача поддержания целостности информации. Действительно, нарушение целостности данных это фактически несанкционированная модификация данных, причиной которой может быть целый класс угроз: умышленные и случайные деструктивные воздействия, ошибки администрирования и т.д.

В настоящее время существует множество решений задачи поддержания целостности файлов в файловой системе: от расширенного протоколирования до ведения баз данных контрольных кодов, однако все они обладают рядом существенных недостатков, основными из которых являются то, что эти решения не интегрированы в файловую систему и обычно не прозрачны для пользователя. Поэтому в последнее время появляется тенденция встраивания функций контроля целостности непосредственно в файловую систему. Примером таких решений могут служить антивирусная файловая система (AVFS) и модуль безопасности ENFORCER. Тем не менее, у подобных решений также имеются недостатки: узкая специализация или необходимость использования специальной аппаратуры, невозможность слежения за целостностью копируемых (распространяемых) файлов, нетривиальность администрирования и др.

Поэтому **актуальными** научными задачами являются:

- разработка нового метода слежения за целостностью и подлинностью файлов, встроенного в файловую систему ОС и лишенного недостатков существующих методов;
- разработка архитектуры файловой системы, использующей созданный метод слежения за целостностью и подлинностью данных.

Целью диссертационной работы является разработка и исследование методов контроля целостности данных, обеспечивающих повышение устойчивости файловой системы к воздействию вредоносных программ. Для достижения поставленной цели необходимо решение следующих задач:

- анализ уязвимостей программного кода, наличие которых позволяет подменять алгоритмы функционирования программных средств защиты, получать права администратора, вызывать отказы в обслуживании со стороны средств защиты;
- исследование особенностей существующих методов поддержания целостности файлов;

- анализ существующих файловых систем с функцией поддержки целостности;
- разработка нового метода слежения за целостностью и подлинностью информации файловой системы, лишенного недостатков существующих средств;
- создание (проектирование и реализация) файловой системы со слежением за целостностью данных;
- исследование файловой системы и разработка методов повышения ее быстродействия.

Методы исследований. При проведении исследований и разработок в диссертационной работе были использованы методы теории алгоритмов, основанных на свойствах эллиптических кривых, теории конечных множеств, математической логики, системного программирования, технология безопасного программирования.

Научная новизна работы состоит в том, что:

- разработана новая концепция *контейнера* и его *представлений*, при этом механизм контейнеров позволяет связывать с файлами произвольную информацию, которая не теряется при распространении контейнеров, а структура контейнера скрыта от пользователя за механизмом представлений;
- разработан новый метод слежения за целостностью и подлинностью файлов файловой системы во время их открытия, с использованием электронной цифровой подписи (ЭЦП); при этом связь ЭЦП с файлом осуществляется с помощью механизма контейнеров;
- обоснована и разработана архитектура файловой системы со слежением за целостностью;
- разработан и исследован вариант построения системы противодействия РПВ, использующий новый метод слежения за целостностью;
- произведен анализ существующих файловых систем с функцией контроля целостности;
- доказана принципиальная возможность внедрения эллиптических алгоритмов в ядро ОС Linux (в виде файловой системы).

Практическая ценность работы заключается в следующем:

- проведен анализ существующих уязвимостей программных систем, которые не выявляются на стадии отладки и тестирования и наличие которых сводит на нет все усилия по обеспечению безопасности информации в КС;

- разработан и реализован новый механизм представлений, позволяющий связывать с файлом произвольную информацию;
- разработана и реализована файловая система с функцией слежения за целостностью и подлинностью файлов, основанная на механизме представлений;
- разработан и реализован быстрый алгоритм выработки и проверки ЭЦП;
- разработаны и реализованы базовые утилиты администрирования файловой системы со слежением за целостностью;
- экспериментальное исследование разработанной файловой системы в режимах персонального использования и защиты от РПВ показало, что ее использование позволяет прозрачно для пользователя контролировать подлинность и целостность распространяемых файлов и обеспечивать эффективную защиту от многих видов РПВ соответственно.

Реализация результатов. Результаты работы внедрены в ряд НИОКР, проводимых ВНИИНС, в рамках Межведомственной координационной программы «Информационные технологии специального назначения», подготовленной в соответствии с поручением Президента Российской Федерации от 24 мая 2002 г. № К863с.

Апробация работы. Результаты работы докладывались и обсуждались на научных сессиях МИФИ-2003, 2004, 2005, XII Общероссийской научно-технической конференции (Санкт-Петербург, 2004), 60-й научной сессии, посвященной дню Радио (Москва, 2005).

Публикации. По теме работы опубликованы 9 печатных работ.

Структура работы. Работа состоит из введения, пяти глав, заключения и приложения. Основной материал изложен на 140 страницах и содержит 23 рисунка. Список литературы содержит 59 наименований. В приложении приведены результаты анализа механизмов функционирования РПВ в среде защищенной ОС, а также встроенных в ОС средств защиты информации.

На защиту выносятся:

- результаты анализа уязвимостей программных систем;
- разработанная концепция представлений;
- разработанный механизм контроля целостности данных;
- архитектура разработанной файловой системы со слежением за целостностью;

- основные принципы построения файловой системы со слежением за целостностью;
- результаты исследования файловой системы со слежением за целостностью.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во введении обоснована актуальность темы, определены цели и задачи исследований, представлены основные положения диссертационной работы, выносимые на защиту.

В первой главе проводится исчерпывающий анализ уязвимостей программных систем, которые не выявляются на стадии отладки и тестирования. Наличие подобных уязвимостей означает существование угрозы проникновения и функционирования РПВ в любой системе. На основании этого утверждения делается вывод о необходимости в составе КС функции контроля целостности, как эффективного средства в борьбе против РПВ. Анализируются существующие подходы по поддержанию целостности файловой системы (ФС): расширенный аудит событий ядра; поддержание базы данных контрольных кодов для отслеживания операций создания, удаления и *изменения* файлов; внедрение механизма ЭЦП на уровне конкретных приложений и/или форматов файлов; антивирусное ПО, предотвращающее операции, потенциально несущие в себе угрозу целостности; аппаратно-программные комплексы, направленные на обеспечение доверенной загрузки системы; криптографические средства.

Все существующие подходы обладают как достоинствами, так и недостатками. Основными недостатками являются: запаздывание во времени, непрозрачность работы, отсутствие контроля подлинности информации. Недостатки позволяют сформулировать требования к системе контроля целостности ФС: защита произвольных файлов от несанкционированной модификации (контроль целостности); контроль не только целостности, но и подлинности файла; выполнение операции "выполнение файла", "загрузка драйвера" только для доверенных файлов; сохранность данных, связанных с файлом и необходимых для проверки его целостности, при любом копировании и/или переносе файла; запрет доступа к файлу с нарушенной целостностью; фиксация факта нарушения целостности файла для последующего анализа администратором; прозрачность реализации для пользователя.

Поддержка целостности файлов означает защиту от несанкционированной модификации данных и позволяет защититься от целого класса угроз, самой опасной из которых является угроза проникновения РПВ. Действительно, РПВ часто сопровождаются несанкционированной модификацией данных, а сама возможность проникновения РПВ не исключена, что и означает необходимость наличия в КС программных средств контроля целостности как средства борьбы с РПВ.

В качестве основного объекта защиты были выбраны программные системы, работающие под управлением ОС Linux. Такой выбор обусловлен двумя факторами: для эффективного решения задачи поддержания целостности необходима тесная интеграция с ядром, что возможно только в случае наличия исходных текстов; результаты работы использованы в НИОКР, связанных с созданием систем на основе ОС Linux.

Во второй главе анализируются существующие ФС, а также выделяются наиболее общие решения, применяемые при их разработке:

- обычно ФС реализована как файловая система-фильтр, это позволяет расширять функциональность любой существующей ФС;
- для передачи управления ФС могут использоваться *представления*, ФС может обеспечивать трансляцию содержимого файла, однако эта идея не стала популярной из-за трудности ее применения;
- обычно, часть кода ФС размещена в пользовательском режиме, чтобы не загромождать ядро и облегчить перенос ФС на другие версии ядер;
- для криптографических ФС обычно используется локальная парольная аутентификация.

Полученные данные необходимы для реализации ФС со слежением за целостностью и помогают построить архитектуру ФС. В результате анализа не было найдено ни одной ФС, прямо направленной на поддержание целостности.

В третьей главе описывается процесс разработки архитектуры ФС со слежением за целостностью. Анализируется способ поддержания целостности файлов, обосновывается выбор механизма слежения за целостностью. Вводится понятие *представления*, влияющее на все слои архитектуры ФС (как центральная *идея* и как способ реализации). Из сформулированных требований к функции контроля целостности следует, что контроль целостности файла необходимо осуществлять криптографически, так как целостность защищаемого файла должна

быть гарантирована даже при компрометации системы (при этом, конечно, необходимо вмешательство доверенного лица). Более слабый контроль целостности на уровне реализации политики безопасности ядра уже присутствует во всех современных многозадачных ОС в виде прав доступа к файлу (дискреционного разграничения доступа) и его дублирование не имеет смысла. Поэтому в качестве алгоритма контроля целостности файлов была выбрана схема ЭЦП. Внедрение ЭЦП имеет множество достоинств, однако существует недостаток, в значительной степени сдерживающий популярность внедрения ЭЦП. ЭЦП является атрибутом файла, логически неразрывно связанным с ним. Однако интерфейс ФС является устоявшейся концепцией и технически невозможно отличить копирование файла от его создания, что означает потерю атрибута при естественном копировании файла при его распространении или даже в пределах локальной ФС. Попытки внедрить ЭЦП на уровне конкретных приложений также не очень популярны, так как это не спасает от *умышленных* воздействий.

В результате рассмотрения требований к ФС со слежением за целостностью и на основе анализа ФС было предложено следующее решение. Единственным способом сделать ЭЦП неразрывным атрибутом файла является размещение ЭЦП внутри файла. Тогда при *любом* распространении файла, ЭЦП не будет утеряна, так как файл копируется целиком - это единая сущность в понятии ФС. В тоже время, размещение ЭЦП внутри файла нарушает его структуру с точки зрения ПО, работающего с этим файлом и рассчитывающего на определенный формат файла. Этот конфликт разрешается с введением механизма *представлений*. Пусть имеется файл определенного формата (далее контейнер), содержимое которого выглядит следующим образом: в заголовке расположена служебная информация (далее атрибуты), а за заголовком следуют данные произвольного формата - вложенный файл (см. рис. 1).

Файл-контейнер доступен в ФС по *двум* именам: реальному имени - имени контейнера, и виртуальному имени - имени представления. При доступе к файлу по имени контейнера - ФС обеспечивает представление "как есть". Контейнер используется для распространения файла. При доступе к файлу по виртуальному имени - ФС скрывает атрибуты во всех файловых операциях, обеспечивая приложению представление вложенного в контейнер файла. При реализации такой идеи для ПО будет создана полная видимость доступа к вложенному файлу и при распространении файла его атрибуты не будут теряться Следующим ша-

гом рассуждений является переход к использованию информации, необходимой для проверки ЭЦП, в качестве атрибутов контейнера и построение архитектуры ФС на основе механизма представлений. Пусть в атрибутах контейнера содержится служебная информация для проверки ЭЦП, которой подписан вложенный в контейнер файл.

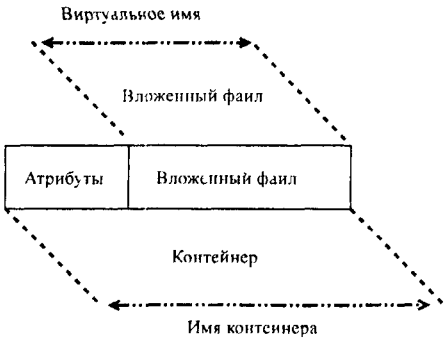

Рис. 1· Структура контейнера

В смысле реализации ФС со слежением за целостностью выполнена как фильтр ФС, так как рациональнее разрабатывать ФС в виде надстройки, что позволяет добавить необходимые свойства к любой ФС с нужными характеристиками. В качестве основной идеи реализации лежат понятия контейнера и представления. Таким образом, ФС поддерживает механизм представлений на основе имен файлов. В зависимости от имени файла, файл представлен пользователю либо "как есть", либо как файл, вложенный в контейнер (сокрытие атрибутов). В случае открытия файла по его виртуальному имени выполняется проверка ЭЦП и определяется степень доверия к пользователю, подписавшему файл. Работа с ЭЦП производится в режиме пользователя, а не ядра (выполняется пользовательским приложением, а не драйвером), так как проверка ЭЦП является сложной операцией и ее реализация в ядре не является желательной в смысле безопасности и «отзывчивости» системы. База данных открытых ключей доверенных абонентов является общей для пользователей системы и поддерживается администратором. Ведется протоколирование нарушений целостности файлов при их открытии и выполнении. Архитектура разработанной файловой системы со слежением за целостностью (SignFS) представлена на рис. 2.

Прозрачность проверки ЭЦП это некий компромисс между удобством и безопасностью. В связи с этим в работе использована схема из двух уровней доверия: уровень абонентов-разработчиков ПО и администраторов, которые могут подписывать выполняемые файлы, и уровень обычных пользователей. Очевидно, что первый уровень (далее привилегированный) не должен быть многочисленным, в то время как последний никак не ограничен. Также необходим компромисс между удобством применения ЭЦП для защиты целостности файлов и ее применением для проверки подлинности. В работе акцент сделан на поддержку целостности файлов и поэтому открытые ключи принадлежат не конкретному пользователю, а системе и едины для всех пользователей системы. Таким образом, введение ЭЦП в ФС дает много преимуществ с точки зрения защиты от РПВ, не говоря уже о том, что в режиме персонального применения пользователи получают возможность подписывать обычные файлы для прозрачной передачи их по сети и прозрачной же проверки подлинности, что не представлено ни в одном существующем решении. При этом, что не менее важно, концепция остается крайне простой, а реализация - прозрачной.

Четвертая глава содержит ответы на вопрос, как реализовать спроектированную в третьей главе ФС рациональным образом (с точки зрения безопасности, быстродействия и простоты архитектуры).

Повышение быстродействия достигалось, в основном за счет реализации процедуры проверки ЭЦП: для ускорения процедуры проверки ЭЦП на эллиптических кривых был использован метод ускорения операции редукции для чисел Мерсенна; для ускорения проверки ЭЦП в системе был введен кэш хеш-значений содержимого проверенных файлов, таким образом удалось избежать проверки ЭЦП для файлов, которые наиболее часто открываются в системе и в тоже время их содержимое не изменяется; в качестве хеш-функции есть возможность выбрать алгоритмы ГОСТ Р 34.11-94 и SHA, причем реализация алгоритма SHA выполняется в 2 раза быстрее для одинаковых объемов данных. Кроме того, были предприняты усилия для уменьшения накладных расходов при поддержке механизма представлений в ядре. ФС перехватывает интерфейсные функции модифицируемой ФС (которые находятся между уровнями: виртуальной ФС и реализацией конкретной ФС), что позволило добиться приемлемого быстродействия и избавиться от необходимости модификации исходного кода ядра ОС Linux. Кроме того, это позволило минимизировать часть ФС, расположенную в ядре ОС

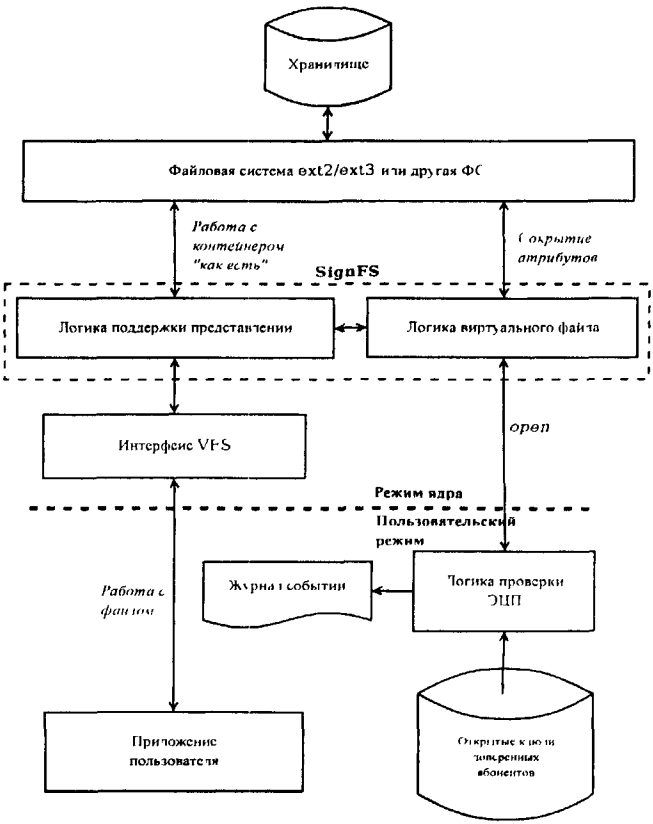

Рис. 2: Архитектура SignFS

Открытые ключи доверенных пользователей хранятся в системе в открытом виде в текстовой форме. Личные ключи хранятся в системе в зашифрованном на пароле пользователя виде. Для шифрования используется алгоритм ГОСТ 28147-89. Чтобы сделать ФС максимально гибкой, в качестве идентификатора пользователя (открытого ключа) используется открытый ключ абонента. Инфраструктура работы с ключами ЭЦП является открытой настолько, насколько это возможно без потери надежности и безопасности. Это делает использование ФС гибким, расширяемым и масштабируемым, так как легко вписывается в любую политику безопасности со своей инфраструктурой поддержки и обмена ключами. Проверка ЭЦП в режиме пользователя

выполнена в виде сервиса (далее демона) и работает в многопоточном режиме сервера: на каждый запрос запускается поток обработки запроса. Разделение реализации на компоненты "драйвер" и "демон", а также использованная в демоне серверная модель обработки запросов позволила сохранить "отзывчивость" системы при работе с множеством защищенных файлов на приемлемом уровне.

ФС должна различать обращение к реальному файлу и к файлу-представлению максимально быстро В работе используется однозначная схема соответствия имени представления и имени его контейнера. Имя контейнера для представления получается путем добавления к имени представления постфикса '.sgn'. Такая простая схема соответствия позволяет избежать накладных расходов для операции открытия файла, который не является представлением. Для удобства защиты системных файлов реализована также поддержка *внешних* контейнеров. Внешний контейнер содержит в себе только служебную информацию для проверки ЭЦП и не включает в себя вложенный файл. Внешний контейнер должен располагаться рядом с файлом (который логически, но не физически, является вложенным в контейнер файлом).

Вместе с ФС были разработаны утилиты администрирования, работающие в текстовом режиме и выполняющие следующие основные действия: подпись файла и создание контейнера, проверка контейнера, добавления личных и открытых ключей в систему и др.

В пятой главе приводятся результаты экспериментального исследования разработанной ФС, которое с одной стороны доказало правильность выбранной архитектуры и реализации, а с другой, позволило выявить узкие места реализации и повысить быстродействие ФС. Результаты исследования подкрепляются математической и логической интерпретацией. Также в главе демонстрируется прозрачность работы ФС со слежением за целостностью. Кроме того, очерчиваются границы областей применения данной ФС.

С точки зрения быстродействия узкими местами являются:

- просмотр (чтение) каталога, в котором присутствует множество контейнеров или файлов с постфиксном '.sgn';
- открытие выполняемого файла (библиотеки);
- попытка открытия файла, который не существует;
- открытие представления.

Для оценки времени чтения каталога был написан тест, суть которого состоит в следующем: создание 10000 файлов с расширением

.txt и замер времени выполнения команды просмотра содержимого каталога; создание 10000 файлов с расширением .sgn и замер времени выполнения команды просмотра содержимого каталога При этом тест выполнялся при загруженном и выгруженном модуле ФС. Выполнение теста показало, что время просмотра для обычных файлов *не отличается* в обоих случаях. В тоже время чтение каталога, в котором присутствуют потенциальные контейнеры, увеличивается более чем в 100 раз. Однако в абсолютном измерении сама по себе операция не является критической - действительно, чтение каталога с 10000 файлами-контейнерами занимает на современной системе меньше секунды. Для оценки открытия файлов-контейнеров также был написан тест, логика которого заключена в следующей последовательности действий; создание файлов размером 32Кб, 100Кб, 1Мб, 10Мб, 50Мб, 100Мб; чтение файлов с замером времени; создание файлов размером 32Кб, 100Кб, 1Мб, 10Мб, 50Мб, 100Мб и подписывание их; чтение файлов с замером времени. Кроме выполнения тестирования зависимость накладных расходов от размера файла была описана математически следующим образом. Время открытия и чтения (из кэша) файла t_{or} в случае постоянных условий эксперимента (один и тот же файл) складывается из фиксированного времени открытия t_o и времени чтения t_r Время чтения файла зависит линейно от размера файла $t_r = k_r L$. Тогда $t_{or} = t_o + k_r L$. Время открытия и чтения файла-контейнера t'_{or} складывается из времени чтения, времени открытия, а также времени проверки ЭЦП t_e, которое в свою очередь можно записать как $t_{ecds} = t_e + k_h L$, где t_e - фиксированное время вычисления ЭЦП по подсчитанному хеш-значению, а $k_h L$ - время подсчета хеш-значения (линейно зависит от размера файла). Тогда накладные расходы (или замедление) принимает вид: $K = \frac{t'_{or}}{t_{or}} = \frac{(k_h + k_r)L + t_e + t_o}{k_r L + t_o}$

При $L \to 0$, $K \to \frac{t_e + t_o}{t_o}$, при $L \to \infty$, $K \to \frac{k_h + k_r}{k_r}$. При небольших размерах файлов (десятки килобайт) ощутимым становится время подсчета ЭЦП по хеш-значению, а время подсчета самого хеш-значения

пренебрежимо мало. При больших размерах файлов, напротив, становится ощутимым время подсчета хеш-значения. Несмотря на значительные относительные накладные расходы (полученные в ходе тестирования), открытие 10M файла на современной системе осуществляется менее, чем за 1 секунду. Для пользовательских файлов и файлов конфигурации это значение задержки не может являться критическим. А если принять во внимание тот факт, что большинство подобных файлов в типичной Linux-системе не превышает размера 32K, и для таких файлов время задержки исчисляется десятой долей секунды, а для ручной проверки файла требуются секунды, то незначительность таких задержек становится очевидной.

ОС Linux является сложной и взаимосвязанной системой компонентов. С точки зрения ФС, в ней содержится тысячи выполняемых файлов и библиотек, зависимости между которыми часто не очевидны. Поэтому для оценки накладных расходов операции выполнения защищаемого файла было проведено отдельное экспериментальное исследование быстродействия, оценка которого может быть только визуальной. В качестве модели была выбрана система (Pentium 4, 2.8Ghz, 512 RAM, OS Debian/Sarge), в которой была альтернативная корневая ФС с подписанными выполняемыми файлами и разделяемыми библиотеками. Активизировалась ФС со слежением за целостностью и запускалась оконная среда. Результат оказался удовлетворительным - система позволяла комфортно работать пользователю без ощутимых задержек.

ФС со слежением за целостностью может использоваться в нескольких режимах, основными из которых являются: персональное использование для распространения подписанных документов и автоматической проверки подлинности документов; использование в защищенных программных системах для защиты от РПВ.

Противодействие РПВ означает следующую политику безопасности:

- подписаны все выполняемые файлы;
- подписаны все разделяемые библиотеки;
- выполнение не подписанных файлов запрещено;
- использование не подписанных файлов запрещено;
- персональное использование ЭЦП может быть разрешено для выделенной группы пользователей.

Для тестирования ФС в режиме противодействия РПВ использовались реальные файловые вирусы для ОС Linux, а также анализирова-

лось поведение ФС при атаке на систему сетевыми червями Тестирование показало эффективность использования ФС в борьбе против РПВ

В заключении отражены **основные результаты**, полученные в данной диссертационной работе.

1. Обоснована необходимость наличия у программной системы функции контроля целостности, как эффективного средства в борьбе против разрушающих программных воздействий;
2. Проведен детальный анализ уязвимостей программных систем, которые не выявляются на стадии отладки и тестирования и наличие которых сводит на нет все усилия по обеспечению безопасности информации, так как позволяет атакующему получать права администратора, подменять алгоритмы функционирования или вызывать отказы в обслуживании со стороны программных средств защиты информации.
3. Проанализированы существующие подходы к решению задачи поддержания целостности файловой системы, в том числе выделены наиболее общие решения, применяемые в файловых системах;
4. Сформулированы требования по поддержанию целостности файловой системы;
5. Введено новое понятие *контейнера* и его *представлений*, которое решает проблему неразрывной связи атрибутов с файлом и определяет способ реализации файловой системы;
6. Разработан новый механизм слежения за целостностью файлов;
7. Разработана архитектура файловой системы со слежением за целостностью;
8. Спроектированная файловая система реализована и получен рабочий прототип, исследование функционирования которого показало выполнение поставленных задач проектирования,
9. Проведено экспериментальное исследование разработанной файловой системы и разработаны методы повышения ее быстродействия;
10. Проведено экспериментальное исследование функционирования файловой системы, доказавшее ее эффективность как средства защиты от разрушающих программных воздействий.

ПУБЛИКАЦИИ ПО ТЕМЕ ДИССЕРТАЦИИ

1. Уязвимости существующих средств защиты от разрушающих программных воздействий / И.Ю.Жуков, М.А.Иванов, П.А.Косых и др. Научная сессия МИФИ-2005. Сборник научных трудов. Том 12. М.: МИФИ, 2005, с. 146-148.

2. Комплекс программных средств антивирусной защиты в среде ОС МСВС / И.Ю.Жуков, М.А.Иванов, П.А.Косых и др. Научная сессия МИФИ-2005. Сборник научных трудов. Том 12. М.: МИФИ, 2005, с. 149-150.

3. Файловая система со слежением за целостностью / П.А. Косых. Научная сессия МИФИ-2004. Сборник научных трудов. Том 12. М.: МИФИ, 2004, с. 170-171.

4. Антивирусная защита в АСОД / П.А.Косых, Д.В.Байбаков, Егоров Е.В., Тетерин И.И. Научная сессия МИФИ-2003. Сборник научных трудов. Том 12. М.: МИФИ, 2003, с. 168-169.

5 Операционные системы для защищенных АСОД / П.А.Косых, Д.В.Байбаков, Егоров Е.В., Тетерин И.И. Научная сессия МИФИ-2003 Сборник научных трудов. Том 12. М.: МИФИ, 2003, с. 170.

6. Защита от разрушающих программных воздействий. А.П Ананьев, Р.Р.Арабгаджиев, П.А.Косых и др. 60-я НАУЧНАЯ СЕССИЯ, ПОСВЯЩЕННАЯ ДНЮ РАДИО. Сборник научных трудов. Москва, 2005, с. 167

7. Косых П.А. Файловая система со слежением за целостностью 60-я НАУЧНАЯ СЕССИЯ, ПОСВЯЩЕННАЯ ДНЮ РАДИО. Сборник научных трудов. Москва, 2005, с. 167

8 Косых П.А Файловая система со слежением за целостностью Тезисы докладов 3-ей ежегодной международной научно-практической конференции "Инфокоммуникационные технологии глобального информационного общества".- Казанский государственный университет им. В.И. Ульянова-Ленина, 2005, с 101-102

9. Ефанов Д.В., Косых П.А., Жуков И.Ю. Подходы к построению файловой системы специализированной защищенной ОС // Материалы XII Общероссийской научно-технической конференции «Методы и технические средства обеспечения безопасности информации» - СПб.: Издательство Политехнического университета. 2004, с. 16.

2006A
4565

№ - 4565