И.Б.Бурдонов, А.С.Косачев, П.Н. Яковенко
Защита целостности файлов на внешних носителях в условиях недоверенной операционной системы. Методы и технические средства обеспечения безопасности информации: респ. научно-технич. конф.: тез. докл. – СПб. Изд-во СПбГТУ, 2010.
2 стр.

Защита целостности файлов на внешних носителях в условиях недоверенной операционной системы

Защита информации, а именно обеспечение конфиденциальности, целостности и доступности данных, представляющих ценность для пользователя, является одной из важнейших задач современного информационного сообщества. Важность каждого из указанных аспектов защищенности информации определяется спецификой предметной области. Во многих случаях целостность информации выходит на передний план. Характерным примером такой ситуации является обеспечение целостности базы данных госпиталя, хранящей в том числе список лекарств, назначенных врачом для лечения пациента. Утечка этой информации – истории болезни пациента – может привести к нежелательным, но не жизненно опасным, последствиям для него. В то время как нарушение целостности учетной записи пациента, в частности замена одного лекарства другим, является смертельно опасной для него.

Нарушение целостности информации может быть результатом случайного (ошибки в программе) или преднамеренного (действиями злоумышленника) события в работе вычислительной системы. Мы рассматриваем задачу обеспечения целостности информации от преднамеренных действий злоумышленника.

Целостность информации обеспечивается выполнением двух свойств: отсутствием в ней повреждений и актуальностью информации. В отсутствие контроля актуальности данных злоумышленник может подменить файл одной из его более ранних версий. Контроль актуальности информации тем более важен, что несанкционированная замена одной версии файла другой не требует от злоумышленника знаний структуры (формата) файла.

Существующие решения защиты целостности данных от несанкционированной модификации основаны на динамическом контроле запросов программы на доступ к данным [Overshadow, VPFS]. Данные, получаемые из хранилище в ответ на запросы программы, проверяются специальным монитором, который в случае обнаружения расхождений между фактическим и ожидаемым состоянием данных возвращает программе ошибку. Монитор обнаруживает нарушение целостности посредством поддержания в актуальном состоянии контрольных сумм (например, криптографических хэш-кодов) для хранимых данных. Контрольные суммы обновляются при выполнении программой операции записи в хранилище и проверяются при каждой операции чтения. Контрольные суммы хранятся в защищенной области памяти, доступ к которой возможен только монитору.

Система Overshadow обеспечивает обнаружение целостности данных в условиях, когда программа выполняется под управлением той же недоверенной операционной

системы, которая управляет хранилищем. Монитор при этом располагается в теле гипервизора, а контрольные суммы хранятся в зашифрованном состоянии в файловой системе [Overshadow]. В системе VPFS программа и монитор выполняются в отдельной доверенной виртуальной машине [VPFS]. Контрольные суммы также хранятся внутри этой виртуальной машины. Недостатком этих подходов является наличие возможности у вредоносного программного обеспечения повредить данные, т.к. в обоих случаях хранилище управляется недоверенной операционной системой, что обуславливает необходимость наличия средств восстановления данных к корректному состоянию. Для этого могут использоваться системы, сохраняющие всю историю изменений информации в защищенной от несанкционированной модификации области памяти [Secure File System Versioning at the Block Level]. В результате для обеспечения целостности данных необходимости поддерживать не только контрольные суммы критичных данных, но и историю их изменений.

В предыдущей нашей работе мы предложили систему защиты конфиденциальности данных от утечки через сетевое соединение, основанную на разделении полномочий по доступу к данным и сетевому интерфейсу между двумя изолированными друг от друга виртуальными машинами [МИТСОБИ’2009]. Виртуальная машина, в которой расположены критичные данные (вычислительная виртуальная машина), выполняется под управлением недоверенной операционной системы. В контексте этой машины также функционируют авторизованные пользователем доверенные приложения, которым гипервизор предоставляет доступ к сети посредством удаленного обслуживания ряда системных вызовов, выполняемых ими, в другой (коммуникационной) виртуальной машине. Гипервизор также обеспечивает защиту контекста доверенных процессов от несанкционированной модификации, в том числе со стороны ядра операционной системы.

В этой работе мы предлагаем подход защиты целостности файловых ресурсов, с которыми работают доверенные приложения. Для этого используется третья (файловая) доверенная виртуальная машина, в которой располагаются все файловые ресурсы доверенных приложений, целостность которых критична для пользователя. Файловая виртуальная машина предоставляет доступ к своей файловой системе для программного обеспечения в вычислительной виртуальной машине посредством стандартных механизмов доступа к сетевым файловым системам операционной системы Linux. В файловой виртуальной машине работает сервер сетевой файловой системы – специальная программа, – которая обслуживает все запросы по доступу к файлам. При получении запроса сервер извещает о нем гипервизор. Если запрос является операцией модификации (например, запись в файл), то сервер также запрашивает подтверждение у гипервизора о допустимости выполнения такой операции.

Гипервизор санкционирует операцию модификации файла, только если она была сформирована, как реакция на системный вызов доверенного процесса. Заметим, что

вычислительная виртуальная машина (в отличие от файловой) выполняется под управлением недоверенной операционной системы, и запросы посылаемые ею, могут быть сфальсифицированы. Гипервизор перехватывает все системные вызовы доверенных процессов и может сопоставить системный вызов с запросом, полученным сервером сетевой файловой системы. Операции чтения данных проверяются в точке возврата доверенного процесса из системного вызова. Здесь гипервизор сопоставляет ответ сервера сетевой файловой системы с фактическими данными, возвращаемыми доверенному процессу. Обнаружение гипервизором изменений в данных в процессе их передачи между точкой выполнения системного вызова и сервером файловой системы говорит о нарушении целостности данных читаемых или записываемых доверенным процессом в файл.

При такой архитектуре важно, чтобы протокол доступа к сетевой файловой системе был синхронным и не кэшировал операции доступа к удаленной файловой системе. Для такого протокола можно утверждать, что при выполнении доверенным процессом системного вызова по доступу к файлам, все соответствующие запросы к файловой виртуальной машине и ответы на них будут переданы до того, как системный вызов завершится. Кроме того, протокол доступа к сетевой файловой системе должен быть достаточно высокоуровневым, чтобы гипервизор мог сопоставить параметры системного вызова и содержимое пакетов данного протокола. Для обеспечения доступа к ресурсам файловой виртуальной машины мы использовали сетевую файловую систему V9FS [V9FS], использующую протокол 9P, изначально разработанный для распределенной операционной системы Plan9 [Plan9]. Протокол 9P удовлетворяет перечисленным нами требованиям. Клиентская часть для данной сетевой файловой системы входит в стандартную поставку ядра операционной системы Linux.

Файловая виртуальная машина работает под управлением доверенной операционной системы. Сервер сетевой файловой системы также является доверенной программой. Поэтому все операции модификации файлов, выполняемые в этой виртуальной машине являются санкционированными, коль скоро корректными являются полученные запросы, а их корректность в свою очередь подтверждается гипервизором посредством их сопоставления с системным вызовом, выполненным доверенным процессом. Для повышения доверия к файловой виртуальной машине мы выполняли ее под управлением микроядерной операционной системы Minix3. Мы также перенесли сервер файловой системы V9FS с Linux на Minix3. Суммарный размер кода ядра Minix3 и сервера сетевой файловой системы составляет менее десяти тысяч строк, что на несколько порядков меньше размера ядра операционной системы Linux, под управлением которой работает вычислительная виртуальная машина.

Преимущество данного подход состоит в том, что он не требует использования каких-либо контрольных сумм для проверки целостности информации, т.к. проверка

корректности операции доступа к файлам осуществляется путем сравнения блоков памяти в вычислительной и файловой виртуальной машине непосредственно во время обслуживания каждого системного вызова по доступу к файловой системе. Память обеих виртуальных машин полностью доступна гипервизору. Операция модификации данных в файловой виртуальной машине санкционируется гипервизором только в том случае, если она была сформирована как реакция на системный вызов доверенного процесса. Все остальные операции являются несанкционированными, и сервер сетевой файловой системы возвращает ошибку при их получении, не выполняя никаких модификаций файлов.