Информационные процессы, Том 5, № 5, 2005, стр. 443–447.

Об особенностях хранения электронной подписи в программных продуктах[1]

Ю.С.Булыгин[*,**], **А.А.Ефремов**[**]

[*] *Кафедра радиотехники, Московский физико-технический институт (ГУ)*
141700, Россия, Долгопрудный МО, Институтский пер. 9
[**] *Отдел технологических исследований, ЗАО "Лаборатория Касперского"*
125363, Россия, Москва, ул. Героев Панфиловцев 10
E-mail: {yury.bulygin, andrey.efremov}@kaspersky.com

Поступила в редколлегию 01.11.2005

Аннотация—В работе предложен способ хранения электронной цифровой подписи в программных продуктах, имеющих модульную архитектуру и применяющих криптографическую защиту своего содержимого от несанкционированной модификации, в частности лицензионных данных и исполнимого кода.

1. ВВЕДЕНИЕ В ПРОБЛЕМУ

Электронная цифровая подпись (ЭЦП) применяется в программных продуктах для защиты их содержимого от подмены или несанкционированной модификации. Например, изменение функциональности программного продукта компьютерными вирусами или встраивание шпионских и других вредоносных программ в программное обеспечение. И наконец, серьезной угрозой является выпуск и распространение нелицензионных программ. Это достигается модификацией содержимого программы или лицензионных данных, используемых программой для контроля функциональности или срока ее работы в рамках лицензионного соглашения.

Специфика законодательства и правового регулирования различных государств в области защиты информации, требований к уровню защищенности различных секторов рынка, на которых представлены программные продукты, а также требований различных государственных или крупных организаций к применяемым в программных продуктах средствам защиты влечет необходимость проектировать системы криптографической защиты с учетом возможности смены и одновременной поддержки различных криптографических средств защиты информации, в том числе сертифицированных различными регулирующими органами. Для того, чтобы удовлетворять возросшим потребностям бизнеса, системы криптографической защиты программных продуктов должны проектироваться с гибкими механизмами, позволяющими обновлять криптографические примитивы, включая ЭЦП, открытые ключи или сертификаты открытых ключей, криптографических алгоритмов и их параметров, а также модулей, реализующих криптографические алгоритмы. В то же время, к системам криптографической защиты содержимого программных продуктов зачастую не предъявляются следующие необходимые требования:

1. *Смена и одновременная поддержка различных стандартов криптографической защиты.* В приложениях, защищаемых с помощью ЭЦП, сложно перейти на использование средств шифрования, сертифицированных различными организациями как России, так и других стран, поскольку нет единого и гибкого механизма одновременной поддержки и замены

[1] Работа выполнена при поддержке ЗАО "Лаборатории Касперского"

различных средств шифрования.

2. *Превентивная и по факту компрометации смена ключевых пар ЭЦП.* В случае таких внутренних угроз, как разглашение секретных ключей производителя программного продукта или потенциальной угрозы разглашения (например, после ухода сотрудника, имевшего доступ к ключам), в приложениях отсутствует возможность быстрой смены ключевых пар. Как следствие, нет возможности запретить использование файлов с подделанной ЭЦП.

3. *Защита открытых ключей от подмены в программном продукте.* Даже в случае надежного хранения секретных ключей, в продуктах зачастую реализована достаточно слабая защита ЭЦП файлов, позволяющая даже слабо подготовленным злоумышленникам подменять подписанные файлы как для нелегального использования и распространения лицензий, так и для несанционированного изменения функциональности приложения. Например, компьютерный вирус может изменить функциональность программы так, что она будет наносить вред пользователям, а следовательно и репутации ее производителя.

Системы защиты программных продуктов, проектируемых на основе компонентной (модульной) архитектуры, компоненты которых могут обновляться независимо от остальных, должны также обладать следующим свойством:

4. *Обеспечение целостности независимых наборов файлов программного продукта.* Проверяется ЭЦП отдельного файла программного продукта, но отсутствует проверка, что содержимое этого файла соответствует содержимому связанных с ним файлов. Возможно заменить отдельный файл программного продукта на, например, его более старую версию с неустраненной ошибкой/уязвимостью. Т.о. отсутствует *целостность набора* файлов, формирующих отдельный компонент программного продукта.

В данной работе мы подробнее остановимся на первых двух и четвертой свойствах, как напрямую связанных с способом хранения ЭЦП файлов программных продуктов. Проблема защиты открытых ключей, применяемых для проверки подписи файлов программных продуктов от подмены существенно сложнее. В частности, эта проблема рассматривается в работе [1].

2. ХРАНЕНИЕ ЭЦП В РЕЕСТРЕ

Для решения перечисленных выше проблем система защиты содержимого программного продуктеа должна поддерживать следующие механизмы:

1. Механизм смены и одновременного использования различных алгоритмов шифрования и ЭЦП, принятых как стандарты в различных странах и сертифицированных различными правительственными организациями.

2. Механизм смены, как плановой так и по требованию, и вывода из использования старых секретных ключей, использующихся для подписи файлов уже функционирующих программных продуктах.
   Программные продукты, функционирующие в правительственных и госсударственных организациях, а также во многих крупных компаниях, перед установкой проверяются на наличие программных закладок, и как следствие в таких организациях политикой безопасности запрещено автоматическое обновление функционала программного продукта без участия администратора безапасности организации. В этом случае смена секретных ключей, используемых для подписи программного кода криптографически защищаемого программного продукта, должна осуществляться без обновления подписываемого ими программного

кода.

3. Механизм защиты ЭЦП файлов программных продуктов от подмены, использующий общепринятые криптографические алгоритмы и протоколы.
   Данный механизм сводится к применению криптографической защиты открытых ключей, используемых для проверки подписи, а также защиты самой функциональности проверки подписи.

4. Механизм обеспечения целостности набора, т.е. непротиворечивости файлов между собой в пределах набора, формирующего компонент программного приложения, обновляемый независимо от остальных.

Заметим, что механизмы, реализуемые криптографической системой защиты напрямую зависят от способов хранения ЭЦП файлов программных продуктов.

Механизмы обновления секретных ключей и смены криптографических алгоритмов или реализующих их модулей влекут переподписывание файлов программных продуктов, которые уже установлены и функционируют (или будут установлены) на компьютерах конечных пользователей. Для смены ЭЦП у файла программного продукта необходимо определить, как будет храниться ЭЦП - внутри самого файла, как наиболее очевидный способ, либо отдельно от файла. Хранение подписи вместе с файлом оправдано для файлов, подпись которых обновлять нецелесообразно, например, файлов, содержащих лицензионную информацию, выданную только конкретному пользователю [2]. Однако данный способ не позволяет обеспечить простой механизм смены подписи, т.к. требует либо модификации файла на компьютере пользователя, либо загрузку всего файла, подписанного новой ЭЦП, с сервера производителя программного продукта даже в том случае, если сам файл не изменился. Оба варианта имеют множество недостатков и неприемлимы в программных продуктах, автоматическое обновление которых запрещено политикой безопасности.

В работе рассматривается хранение ЭЦП отдельно от файла программного продукта. Одним из вариантов может быть хранение ЭЦП файла в отдельном файле. Основной недостаток этого способа заключается в том, что он не позволяет решить проблему целостности набора файлов программы, поскольку каждый файл может быть проверен только отдельно от остальных файлов набора.

В качестве решения в работе рассматривается способ раздельного хранения ЭЦП, позволяющий обеспечить целостность набора файлов программного продукта (свойство 4), который заключается в хранении ЭЦП файлов набора, целостность которого необходимо обеспечить, в одном списке, называемым *реестром набора*. Реестр набора в свою очередь подписан. ЭЦП реестра набора гарантирует, что файл f_1, входящий в состав набора и имеющий ЭЦП s_1, может использоваться только с файлом f_2, имеющем только ЭЦП s_2. Т.о. обеспечивается неизменность содержимое всего набора файлов. Подобный способ используется в ряде реализаций для обеспечения целостности целого набора во время обновления. В частности, схожий способ используется в Catalog-файлах [3] для обновления файлов в составе наборов.

3. СОСТАВ НАБОРОВ ПРОГРАММНОГО ПРОДУКТА

В модульном программном продукте состав наборов определяется следующим правилом: набор файлов, целостность которого необходимо обеспечивать, обновляется независимо от остальных наборов. Для данных наборов формируются реестры, содержащие ЭЦП всех составляющих наборы файлов. Рассмотрим следующий пример состава консистентных наборов программного продукта.

Если в программный продукт входят два компонента, один из которых обновляется автоматически, а другой по желанию пользователя продуктом, то на каждый компонент необходимо формировать отдельный реестр для хранения ЭЦП файлов этих компонентов. Расположение ЭЦП всех файлов программного продукта в одном реестре приведет к следующей проблеме. Вместе с обновлением файлов автоматически обновляемого компонента необходимо обновить реестр, содержащий ЭЦП файлов этого компонента и файлов второго (обновляемого по желанию пользователя) компонента. Поскольку заранее не известно, какая версия второго компонента на момент обновления первого функционирует в программном продукте, то количество реестров должно быть равно прямому произведению всех выпущенных версий первого компонента и всех выпущенных версий второго компонента, что значительно усложняет логику функционирования и обновления программного продукта. С ростом числа версий или компонентов в программном продукте формирование и дальнейшая поддержка такого количества реестров становится практически невыполнимой задачей. Очевидно также, что часть реестра, относящаяся ко второму компоненту, избыточна в процессе обновления первого компонента.

Искусственное связывание независимых компонентов, чем является общий реестр на файлы этих компонентов, делает сложным, неочевидным и хрупким функционирование и обновление модульного программного продукта. Следует отметить, что в программном продукте нельзя хранить секретные ключи, использующиеся для создания ЭЦП файлов, а следовательно формировать реестры на компьютере конечного пользователя невозможно.

4. ОПТИМИЗАЦИЯ ПРОВЕРКИ ЭЦП

Поскольку целостность каждого отдельного файла, входящего в состав одного из наборов программного продукта, обеспечивается целостностью всего набора с помощью общей ЭЦП реестра, то избыточно хранить в реестре ЭЦП отдельных файлов набора, необходима информация, уникально описывающая содержимое отдельного файла. Исходя из этого, необходимым и достаточным условием обеспечения целостности отдельного файла в наборе, описываемым реестром, является хранение в реестре значений функции хеширования (далее хеш-значений) [4] содержимого файлов набора. Хранение хеш-значений в реестрах вместо ЭЦП также позволяет:

— сократить время формирования реестров, т.к. вычисление хеш-значения существенно быстрее вычисления ЭЦП;
— сократить время проверки набора файлов, поскольку вместо проверки ЭЦП каждого файла в отдельности необходимо проверить хеш-значения каждого файла и общую ЭЦП реестра всего набора файлов;
— уменьшить размер реестра, т.к. размер хеш-значения меньше размера ЭЦП (например, в 2 раза для стандарта функции хеширования ГОСТ 34.11 [5] и стандартов ЭЦП ГОСТ 34.10-94/2001 [6] и в 16 раз для алгоритмов MD5/RSA-2048 [4]).

5. ОБНОВЛЕНИЕ НАБОРОВ ПРОГРАММНОГО ПРОДУКТА

В процессе обновления файлов компонента программного продукта необходимо определить, какие имеенно файлы компонента обновились и их необходимо загрузить с сервера производителя программного продукта и заменить в продукте, поскольку неэффективно загружать все файлы программного продукта, в том числе необновленные. Надежным механизмом является сравнение бинарного содержимого файла в программном продукте с бинарным содержимым соответствующего файла на сервере, но данное сравнение требует безусловной загрузки всех файлов компонента с сервера, что недопустимо на практике. Однако для сравнения бинарного содержимого файлов достаточно сравнить их хеш-значения, т.о. перед обновлением необходимо получить список хеш-значений файлов компонента программного продукта с сервера производителя продукта.

СПИСОК ЛИТЕРАТУРЫ

1. Bulygin Y., Protecting public keys from forgery via "unkeyed" commutative encryption, in *Proc. of 8th International Symposium on Communication Theory and Applications*, Ambleside, UK, 2005, pp. 83-86

2. Булыгин Ю.С., Аспекты лицензирования: структура и защита лицензионных данных, *Проблемы информационной безопасности. Компьютерные системы*, 2005, № 1, pp. 23-28

3. Using Catalog files, Microsoft Corporation, `http://msdn.microsoft.com/library/default.asp?url=/workshop/delivery/download/overview/catalog.asp`

4. Menezes A., van Oorschot V. and Vanstone S., *Handbook of Applied Cryptography*, CRC Press, Inc., 1997

5. *ГОСТ Р 34.11-94.* Информационная технология. Криптографическая защита информации. Функция хеширования.

6. *ГОСТ 34.10-94/2001.* Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи.