ОСОБЕННОСТИ ИСПОЛЬЗОВАНИЯ СКРЫТЫХ КРИПТОКОНТЕЙНЕРОВ ПРИ СОКРЫТИИ ДАННЫХ

А. М. Кадан, И. А. Сазановец
Кафедра системного программирования и компьютерной безопасности, Гродненский государственный университет им.Янки Купалы
Гродно, Республика Беларусь
E-mail: kadan@mf.grsu.by, sazanovec_ia_13@mf.grsu.by

В работе ставилась задача исследования особенностей использования скрытых криптоконтейнеров при сокрытии данных. Рассмотрены основные способы прозрачного шифрования разделов жёсткого диска или директории файловой системы, а также способы реализации идеи шифрования с отрицанием в таких криптосистемах. Отмечено, что при скрытии информации в криптоконтейнерах важно не допускать некоторые типичные ошибки, позволяющие обнаружить факт существования скрытых данных.

Введение

Известно немало способов защиты цифровых данных с помощью шифрования. Один из них – использовать зашифрованные виртуальные жёсткие диски, так называемые криптоконтейнеры. При их монтировании пользователю требуется ввести пароль, а дальше шифрование происходит прозрачно.

Прозрачное шифрование (transparent encryption, on-the-fly encryption, real-time encryption) – вид шифрования, при котором пользователь работает с данными как с незашифрованными, а шифрование/дешифрование выполняется на низком уровне (драйвером программы). При этом незашифрованные файлы как таковые не существуют на файловой системе, а расшифрованные части зашифрованных данных хранятся временно в оперативной памяти или в кэше [1].

I. Скрытые криптоконтейнеры

Традиционно, пользователь открывая файл криптоконтейнера, авторизуется для работы с ним, и криптоконтейнер монтируется как локальный диск. Однако некоторые программы (VeraCrypt [2], Jetico BestCrypt Container Encryption [3]) позволяют также создавать скрытые криптоконтейнеры в уже готовых внешних контейнерах. Скрытый образ лежит на незанятом пространстве внешнего контейнера. Такая схема представлена на рис. 1.

Например, внешний криптоконтейнер может быть объемом в 1 Гбайт, занято 300 Мбайт, значит, на оставшихся 700 Мбайтах можно создать скрытый криптоконтейнер, например, на 500 Мбайт.

Если пользователь вводит пароль от внешнего контейнера, монтируется внешний (если из примера выше, то на 1 Гбайт), если же от скрытого, то монтируется скрытый (соответственно, на 500 Мбайт). Т.к. свободное пространство внешней части тоже шифруется, то выявить наличие скрытой части достаточно затруднительно.

Если же владелец контейнера будет подвергаться неприемлемому воздействию, то он может, пусть и не сразу, чтобы не вызвать подозрения, всё же ввести пароль – но от внешнего контейнера, отвлекая тем самым внимание эксперта от наличия секретной информации. Соответственно, лучше секретную информацию хранить в скрытом криптоконтейнере, а во внешней части хранить информацию, которую, с одной стороны, имеет смысл шифровать, но с другой стороны, раскрытие которой не повлекло бы за собой недопустимых последствий.

При этим необходимо соблюдать ряд рекомендаций. Например, после создания скрытой части, на внешний криптоконтейнер ничего нельзя записывать (иначе можно повредить скрытую часть), а также на внешней части лучше использовать более простые файловые системы, такие как FAT32, а вот NTFS использовать нельзя, т. к. она хранит свою MFT не только в начале раздела, и в таком случае, работая со скрытой частью, можно повредить MFT внешней части, а значит, и её файловую систему, что, несомненно, вызовет подозрения.

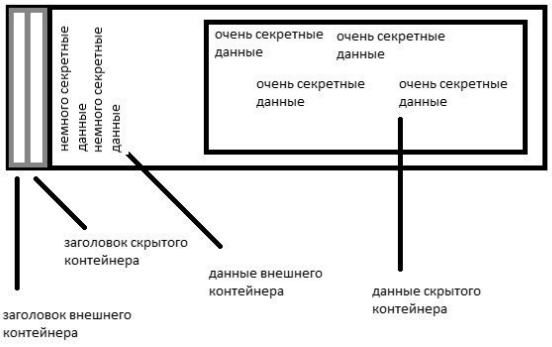

Рис. 1 – Пример схемы расположения скрытого образа на незанятом пространстве внешнего контейнера

II. Детектирование скрытой части

К настоящему времени нет надежных средств, обеспечивающих возможность детектирования наличия скрытой части техническими

средствами, имея доступ к внешней части. В ряде случаев, можно анализировать файловую систему различных версий одного криптоконтейнера, взятых в разное время, и на основе полученных изменений делать выводы. Ингда используется тот факт, что на внешнюю часть, после создания скрытой, прктически ничего нельзя записывать. А это значит, что метки времени файлов, так называемые timestamp'ы, изменяться не будут.

Открытым остаётся вопрос о закладках (backdoor'ax) в подобных программах. Хотя VeraCrypt имеет открытый исходный код, но это ещё не гарантирует отсутствие закладок, не говоря уже о коммерческом продукте от Jetico.

Несмотря на эти недостатки, многие используют скрытые криптоконтейнеры для хранения важных данных, наивно веря в их высокую надёжность и в то, что их невозможно обнаружить. Следовательно, работа в этом направлении имеет крайне важное значение: она позволяет определить, является ли техника сокрытия одного контейнера в другой надёжной с точки зрения стеганографии, и если нет, то стоит разрабатывать подходы, как сделать скрытую часть ещё более трудно обнаруживаемой.

Так в работе [4] проводилось исследование, как, имея несколько версий контейнера, полученных в разное время, и имея доступ к внешнему контейнеру, установить наличие скрытой части и, более того, определить её размер и положение. Для этого производилось сравнение незанятого пространства внешних частей разных версий одного и того же контейнера. После сравнения определялось, какие участки криптоконтейнера изменялись и каким образом, и на этом основании делались выводы.

Наиболее важные правила работы со скрытыми криптоконтейнерами:

- на внешней части следует использовать нежурналируемые файловые системы (многие программы под Windows предлагают выбрать либо FAT32, либо NTFS);
- NTFS вообще выбирать нельзя: во первых – слишком сложная, во вторых – её MFT-таблицы располагаются не только вначале, но и в середине, а значит, риск её повреждения во время работы со скрытой частью крайне высок;
- выбор надо делать в пользу FAT32 – её File Allocation Table находится в начале и соответствующий риск минимизируется;
- после создания внешней части и записи в неё информации следует выполнить дефрагментацию (чтобы сдвинуть все данные к началу раздела) и больше ничего сюда не записывать);
- еще одна важная проблема – если мы не будем работать с внешней частью, а только со скрытой, это будет подозрительно (установить, что мы не работаем с внешней частью, можно, например, проанализировав timestamp'ы её файлов).

III. Программа для имитации работы на внешнем контенере

Одно из решений очевидно: надо создать видимость работы на файловой системе без изменения самих файлов. Будем считать, что пользователь выбрал для внешней части FAT32. Тогда, для того, чтобы создать такую иллюзию, мы можем менять лишь атрибуты файлов. К примеру, удобно менять следующие атрибуты: либо только дату и время последнего доступа к файлу, либо дату и время последнего изменения и доступа.

Для этой цели была разработана программа Fat32 User Work Simulator, которая позволяет пакетно изменять атрибуты у определенного процента файлов выбранного типа, расположенных по определенному пути в файловой системе.

Данный продукт является приложением WPF, требует для работы .NET Framework и написан на языке C. .

IV. Заключение

С технической точки зрения, грамотно реализованная технология скрытых криптоконтейнеров позволяет надежно спрятать файлы. Но при этом следует учитывать риски повреждения внешнего контейнера. Достаточно «безопасной» файловой системой для внешней части может служить FAT32.

Главная задача использования внешней части – ввести в заблуждение эксперта, отвести внимание от реально секретной информации. Для этого можно использовать различные подходы: давать ложную информацию, использовать психологические приёмы, не выдавать себя тем, что с внешней частью давно не велась работа.

Можно выбрать противоположную тактику – дать определённые намёки на то, что скрытая часть имеется, а в самой скрытой части тоже ничего ценного не хранить и использовать её как ещё один способ отвести внимание. А саму информацию хранить в другом месте, например, на другом физическом устройстве.

Список литературы

1. Отрицаемое шифрование [Электронный ресурс] / Википедия. – Режим доступа: https://ru.wikipedia.org/wiki/Отрицаемое_шифрование. – Дата доступа: 10.08.2016.
2. BestCrypt [Электронный ресурс] / Wikipedia. – Режим доступа: https://en.wikipedia.org/wiki/BestCrypt. – Дата доступа: 10.08.2016.
3. VeraCrypt – Documentation [Электронный ресурс] / CodePlex. – Режим доступа: https://veracrypt.codeplex.com/documentation. – Дата доступа: 10.08.2016.
4. Detecting Hidden Encrypted Volumes [Электронный ресурс] / HAL-Inria. – Режим доступа: https://hal.inria.fr/hal-01056376/document. – Дата доступа: 10.08.2016.