ОБЕСПЕЧЕНИЕ ЗАЩИТЫ ДАННЫХ В ОПЕРАЦИОННОЙ СИСТЕМЕ ПУТЕМ ШИФРОВАНИЯ И АРХИВИРОВАНИЯ

Очилов Низомиддин Нажмиддин угли

главный специалист, программист, Государственный центр тестирование при Кабинете Министров, Республика Узбекистан, г. Ташкент

ENSURING DATA PROTECTION IN THE OPERATING SYSTEM BY ENCRYPTION AND ARCHIVING

Nizomiddin Ochilov

Chief specialist, programmer, State testing center under the Cabinet of Ministers, Republic of Uzbekistan, Tashkent

Аннотация. Статья посвящена организации защиты данных в операционной системе посредством шифрования и архивирования в операционных системах с открытым кодом. А также она посвящена организации шифрования и дешифрования системных файлов в процессе инициализации, разработке графических программных обеспечений шифрования для файловой системы и оценке эффективности функционирования графического программного комплекса. Для отдела инициализации целостность данных важнее, чем конфиденциальность ваших данных. Метод шифрования LUKS (Linux Unified Key Setup) служит для устранения таких недостатков (рис.1). Одним из основных преимуществ является то, что зашифрованный раздел трудно подделать [1].

Abstract. The article is devoted to the organization of data protection in the operating system through encryption and archiving in open source operating systems. And also it is devoted to the organization of encryption and decryption of system files during the initialization process, the development of graphical encryption software for the file system and the evaluation of the effectiveness of the graphical software package. Data integrity is more important to the provisioning department than the privacy of your data. The LUKS (Linux Unified Key Setup) encryption method serves to eliminate such shortcomings (Fig. 1). One of the main advantages is that the encrypted partition is difficult to forge.

Ключевые слова: хеширования; шифрования; дешифрования; файловая система; криптографический модуль.

Keywords: hashing; encryption; decryption; file system; cryptographic module.

Введение. Рассмотрим возможность использования TPM (Trusted Platform Module) для хранения ключа шифрования и проверки безопасной среды загрузки. TPM на самом деле является криптопроцессором в системе. Эта технология позволяет выполнять безопасное шифрование в системе без необходимости ввода ключа (например, используя вход по

отпечатку пальца или метод аутентификации, не зависящий от метода шифрования). В идеале он должен работать с UEFI Secure Boot, который, в свою очередь, не позволяет выполнять расшифровку при повреждении системных настроек.

Рисунок 1. Изображение. Метод шифрования LUKS (Linux Unified Key Setup)

Однако поддержка TPM в Linux все еще находится в зачаточном состоянии. Мы используем UEFI Secure Boot, чтобы полностью покрыть цепочку инициализации электронной подписью.

Поскольку локальный стандарт шифрования O'zDSt 1105:2009 имеет тот же размер ключа, что и AES-256, было принято решение не менять порядок генерации ключей из последовательности паролей. Для этого 7zip использует алгоритм хеширования SHA-2. Причина, по которой он имеет хорошую статистическую криптостойкость заключается в том, что он используется в качестве генератора псевдослучайных последовательностей.

Однако процедура расширения коммутатора кардинально отличается для AES и местного стандарта. Поэтому весь модуль, находящийся в файле AES.c, был модифицирован encrytp.c. 7zip осуществляет разбиение текста на 128-битные блоки и заполнение их до нужной длины в соответствии с правилами. encryptp.c модуль изменяет размер блока на 64 бита, поскольку число 128 является 64 кратным. А также это изменение не только осуществляет изменению константы, но и удваивает количество блоков [2-3].

7zip использует шифрование в режиме CBC (режим слияния зашифрованных текстовых блоков), но можно использовать и режим счетчика. Этот же метод был учтен при создании модуля encryptp.c. Поскольку функция расширения ключа изначально была заполнена раундовыми ключами с использованием встроенного уникального свойства пользовательского массива, созданного 7zip, была создана только одна раундовая функция шифрования (осуществляется во время шифрования и дешифрования).

Этот метод используется в разных режимах. В большинстве случаев отклонения во времени запуска в зависимости от выбранного режима незначительны. В режимах CBC и CFB (режим обратной связи шифротекста) время запуска увеличивается. При выборе шифрования в режиме CBC для данного метода обеспечивается криптостойкость [4].

Для оценки эффективности работы системы информационной безопасности при проведении эксперимента по выполнению файла с расширением «exe» 1, 2, 5 и 10 раз в секунду на процессоре с заданной частотой система информационной безопасности строилась в встроенном и неустановленном режимах. Для каждого значения частоты выполнения файла выполнялось по 10 экспериментов, после чего вычислялась ошибка результата измерения.

Основная часть. Для обработки результатов эксперимента были предприняты следующие шаги:

1. Среднее значение 10 тестов рассчитывается по следующей формуле:

$$x_o = \frac{\sum_{i=1}^{N} x_i}{N}$$

2) Ошибка рассчитывалась по следующей формуле:

$$\Delta x_i = |x_o - x_i|$$

3) Квадратичные ошибки вычислялись по следующей формуле.

4) Средние квадратичные ошибки среднего арифметического рассчитываются по следующей формуле:

$$S_{x_o} = \sqrt{\frac{\sum(\Delta x_i)^2}{n(n-1)}}$$

5) Значение надежности измерения было равно 0,95.

6) Для значения, полученного из достоверности измерения и количества проведенных экспериментов, был определен коэффициент Стьюдента $t = 2{,}262$.

7) Доверительный интервал (ошибка измерения) определялся по следующей формуле:

$$\Delta x = S_{x_{\tilde{y}}} \times t$$

Из таблицы известно, что дополнительная загрузка центрального процессора в приложениях не превышает 19%. В первом эксперименте это значение не превышало 23%. Из этого можно сделать вывод, что полученная модель и результаты верны, а использование центрального процессора можно предсказать на основании результатов, которые мы привели выше.

При этом время загрузки для защищенной файловой системы увеличилось с 5 до 8 секунд, для программы записи внешних устройств – с 4 до 8 секунд, а для графического программного обеспечения шифрования для файловой системы – с 5 до 7 секунд. Время запуска полной СИБ увеличилось до 4 секунд.

Определена эффективность средств защиты и изучено влияние операционной системы на загрузку вычислительных ресурсов, при этом дополнительная загрузка процессора в режиме запуска приложений не превышала 23%, а в обычном режиме работы программ не превышала 17%, при этом время запуска программы не превышало 4 секунд (Рис.2.).

Адекватность полученных результатов доказано статистической обработкой эксперимента.

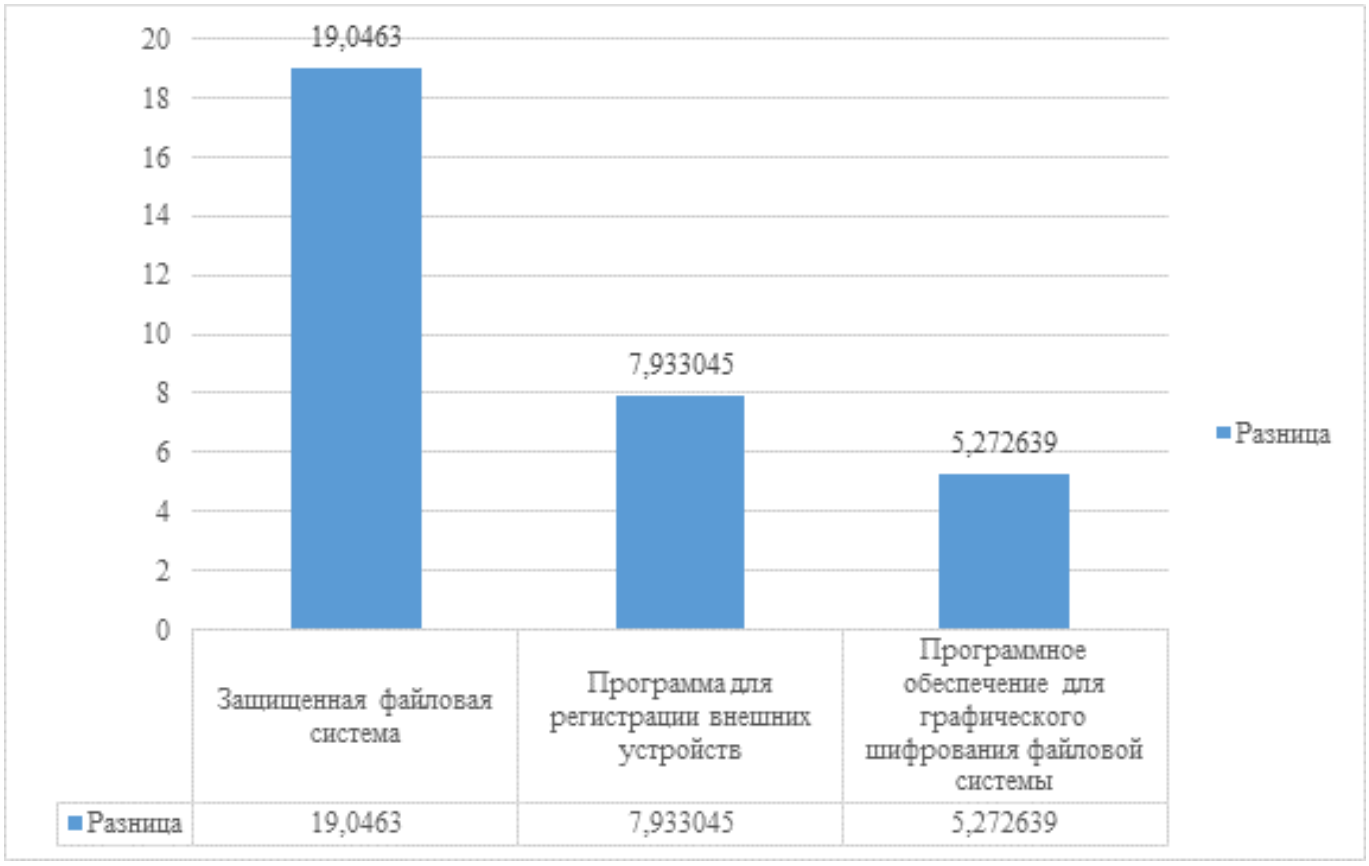

Рисунок 2. Средние квадратичные результаты производительности разработанных программ

Программное обеспечение для архивации 7zip выполняет шифрование алгоритмом AES и делает все необходимые приготовления данных перед шифрованием (создание ключа-пароля, добавление сообщения к длине блока путем проверки правильности расшифровки с помощью расширенной последовательности, создание вектора инициализации и т.д.). Также было изучено, что из-за того, что размер блока AES в 2 раза больше размера блока ГОСТ 28147-89, использование ГОСТ 28147-89 также несколько увеличивает скорость шифрования на единицу.

Причина того, что шифрования диска недостаточно для обеспечения конфиденциальности данных, заключается в том, что шифрование всей цепочки инициализации с помощью UEFI Secure Boot и GPG позволило добиться хорошего уровня защиты от замены всей системы и взлома системных программ.

Для доступа к криптографическому модулю был разработан очень удобный графический интерфейс, так что пользователь может получить доступ к командной строке и получить доступ к программе p7zip без необходимости запоминать последовательность команд.

В максимальном режиме скорость архивации можно увеличить до 2-3 раз за счет уменьшения параметров файла, и эти тесты доказали, что специально созданную архивацию, а также программу шифрования по местному стандарту можно использовать практически на любых компьютер.

Заключение. Созданный криптографический модуль протестирован и одобрен на соответствие местному стандарту. Определена эффективность работы средств защиты и изучено влияние операционной системы на загрузку вычислительных ресурсов, при этом дополнительная загрузка процессора в режиме запуска приложений не превышала 23%, а в обычном режиме работы программы не превышала превышать 17%. Адекватность полученных результатов подтверждалась статистической обработкой эксперимента.

Список литературы:

1. R. Nivedhaa, J. Jean Justus, A Secure Erasure Cloud Storage System Using Advanced Encryption Standard Algorithm and Proxy Re-Encryption - Proceedings of the 2018 IEEE International Conference on Communication and Signal Processing, ICCSP 2018.

2. Shivarajkumar Hiremath, Sanjeev R. Kunte, Ensuring Cloud Data Security using Public Auditing with Privacy Preserving - Proceedings of the 3rd International Conference on Communication and Electronics Systems, ICCES 2018.

3. Fitzroy D. Nembhard, Marco M. Carvalho, Thomas C. Eskridge, Towards the application of recommender systems to secure coding - Eurasip Journal on Information Security 2019.

4. Timo, Speedtest and Comparsion of Open-Source Cryptography Libraries and Compiler Flags, – 20.08.2022. Режим доступа:https://idlebox.net/2008/0714-cryptographyspeedtest-comparison