ИНФОРМАТИКА

Вестник Сыктывкарского университета.
Серия 1: Математика. Механика. Информатика.
Выпуск 1 (38). 2021

УДК 004.056.55, 004.3 DOI: 10.34130/1992-2752_2021_1_43

ПРОГРАММНО-АППАРАТНЫЙ КОМПЛЕКС ДЛЯ ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ПРИ ИСПОЛЬЗОВАНИИ ЭЛЕКТРОННЫХ ПОДПИСЕЙ

Д. А. Гертнер, Д. О. Леонтьев, Л. С. Носов, Д. С. Шучалин

В данной работе представлен прототип программно-аппаратного комплекса для безопасного подписания электронных документов на базе специализированного для данных операций доверенного устройства.

Ключевые слова: программно-аппаратный комплекс, электронная подпись, Python.

1. Введение

Электронная подпись (ЭП) используется для подтверждения авторства составителя электронного документа, а также позволяет удостовериться в том, что данный документ не изменялся с момента его подписания [1]. При этом закрытый ключ пользователя является наиболее уязвимым компонентом всей цифровой подписи, так как его компрометация может привести к тому, что злоумышленник сможет выдавать себя за владельца украденного ключа. Поэтому необходимо обеспечить надёжную защиту конфиденциальности секретного ключа пользователя. Зачастую пользователь хранит секретный ключ и подписывает документы на своём персональном компьютере, что содержит в себе такие угрозы, как кража закрытого ключа вредоносным программным обеспечением, угроза осуществления несанкционированного доступа к компьютеру с возможностью последующего хищения ключа или подписания документа, а также угроза осуществления удалённого управления средствами подписания.

Для противодействия этим угрозам необходимо использовать более совершенный механизм хранения приватных ключей, реализованный вне персонального компьютера, и механизм подписания, исключающий возможность его удаленного запуска. В данной работе представлен прототип программно-аппаратного комплекса для безопасного подписания электронных документов на базе специализированного для данных операций доверенного устройства.

2. Требования к разрабатываемому прототипу программно-аппаратного комплекса

Для обеспечения взаимодействия между доверенным устройством и пользовательским компьютером было принято использовать клиент-серверную архитектуру. Данное решение позволяет хранить закрытый ключ и подписывать документы на отдельном устройстве, которое специализировано исключительно на операциях, касающихся ЭП, защиты закрытого ключа, а также безопасного обмена информацией между компьютером пользователя и доверенным устройством.

Данный программно-аппаратный комплекс должен обеспечивать защиту целостности страниц электронного документа методом наложения на них специального защитного рисунка. На стороне клиента должны осуществляться следующие операции:

1. Загрузка документа на доверенное устройство.
2. Получение документа, защищённого рисунком, с возможностью последующего удаления этого рисунка.
3. Сохранение оригинального документа, защищённого ЭП.
4. Возможность проверки ЭП документа.

На стороне сервера должны осуществляться следующие операции:

1. Получение оригинального документа.
2. Перевод документа в изображение с последующей генерацией на нём защитного рисунка.
3. Подтверждение подлинности полученного документа.
4. Подписание ЭП оригинального документа.

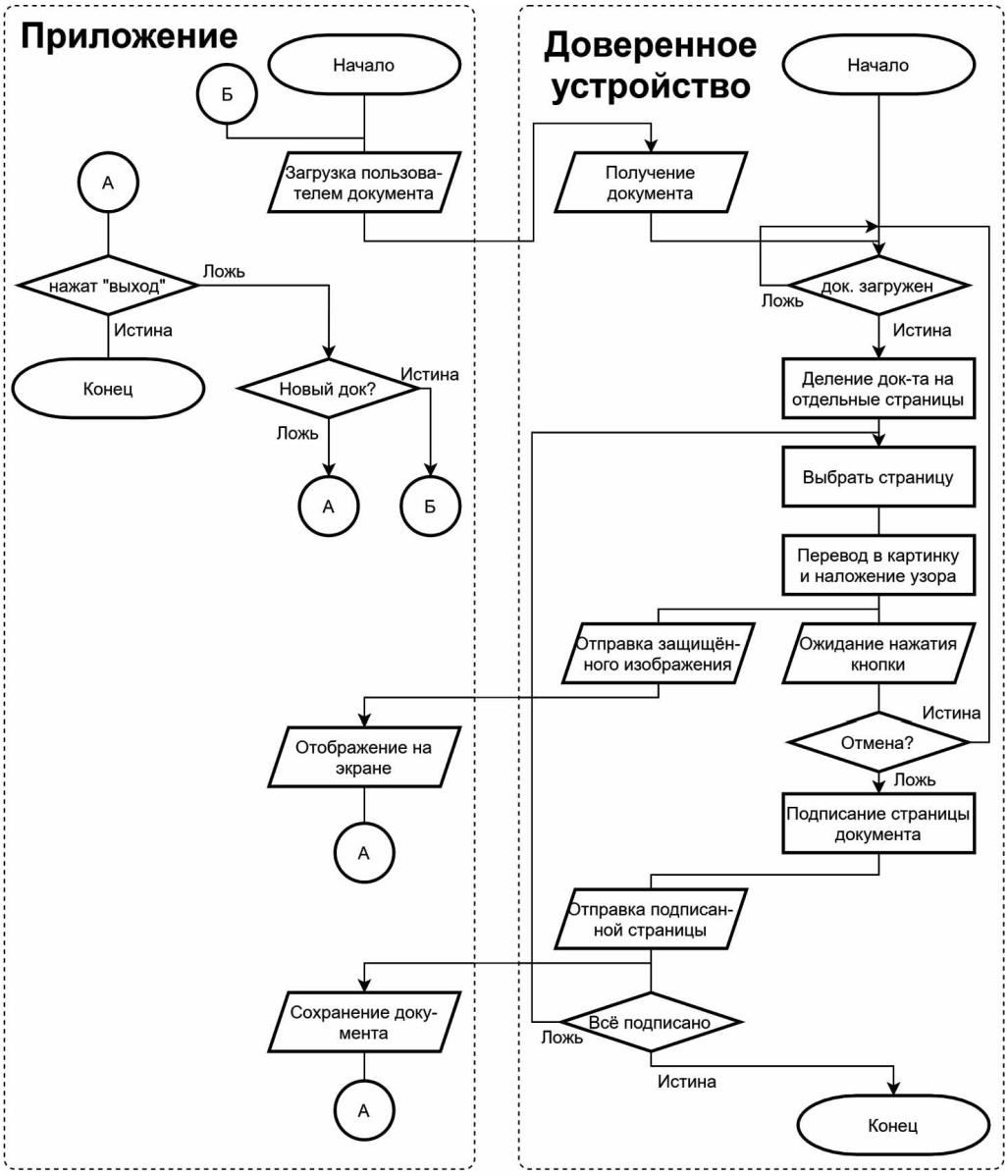

Рис. 1. Схема работы комплекса

На рис. 1 изображена схема работы программно-аппаратного комплекса с точки зрения взаимодействия клиентского приложения и доверенного устройства. Суть данного взаимодействия в том, чтобы свести к минимуму вероятность компрометации системы ЭП документов, поместив наиболее важные этапы обработки данных в доверенную среду.

В соответствии с требованиями к разрабатываемому программно-

аппаратному комплексу выбор был сделан в пользу одноплатного компьютера **Raspberry Pi 3 model A+** [2]. В качестве комплектации принято решение подключить экран, который может одновременно выводить две строки по 16 символов, а также три кнопки, служащие для подтверждения, отмены и получения статуса операции. Внешний вид устройства представлен на рис. 2.

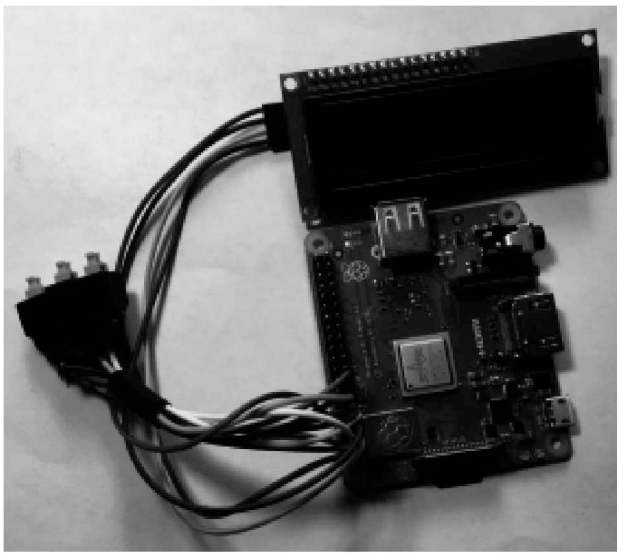

Рис. 2. Внешний вид устройства

Для реализации проекта был выбран высокоуровневый язык программирования **Python**. Определяющим фактором стала его кроссплатформенность и удобное управление модулями.

Для написания пользовательского интерфейса было принято решение использовать модуль **PyQt5** [3]. Он содержит все необходимые элементы **GUI**, имеет удобные методы взаимодействия между элементами интерфейса и исполняемыми частями кода а также графический интерфейс для построения **GUI**. Для работы с изображениями была выбрана библиотека **Pillow** [4], поддерживающая все базовые растровые графические форматы и различные варианты цветовых пространств, в том числе изображения с альфа- каналом. Для реализации клиент-серверной архитектуры была выбрана система удалённого вызова процедур **gRPC** [5].

3. Тестирование программно-аппаратного комплекса

3.1. Тестирование клиентского приложения и программной части в режиме эмуляции

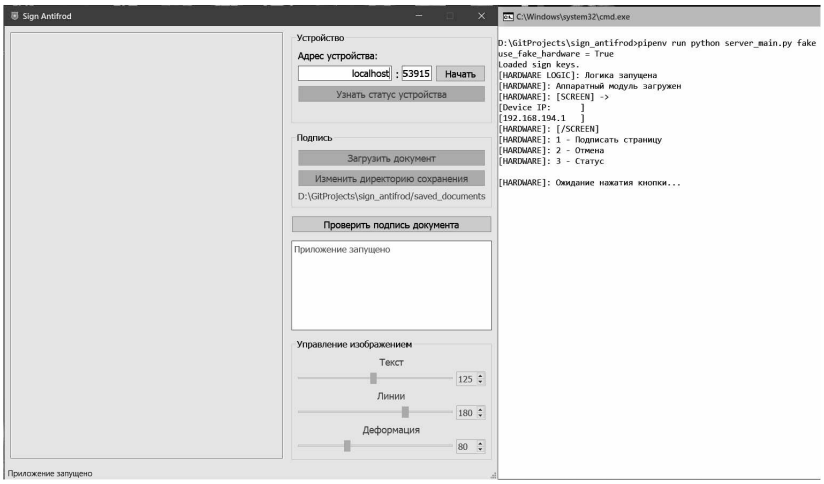

Рис. 3. Запуск клиента и сервера в режиме эмуляции

После запуска клиента и сервера (рис. 3) необходимо ввести адрес эмулятора в локальной сети и нажать «Начать», чтобы соединиться с устройством (рис. 4).

Директорию сохранения документов можно поменять с помощью стандартного файлового диалога, после нажатия соответствующей кнопки. Для выбора документа для загрузки используется диалог, позволяющий выбрать PDF-документ (рис. 5).

Слайдеры управления изображением работают в диапазоне [0, 255] и изменяют прозрачность защитных слоёв без подтормаживаний и ошибок (рис. 6).

После нажатия кнопки 1 происходит подписание документа и на устройстве выводится сообщение: «Successfully signed.» (рис. 7). После подписания текущей страницы подписанный документ и его подпись в формате «.sign» сохраняются в директории сохранения, а в окне клиентского приложения отобразится следующая страница.

Чтобы увидеть код подлинности страницы, нужно нажать клавишу 3. Если сверить код подлинности на доверенном устройстве с кодом

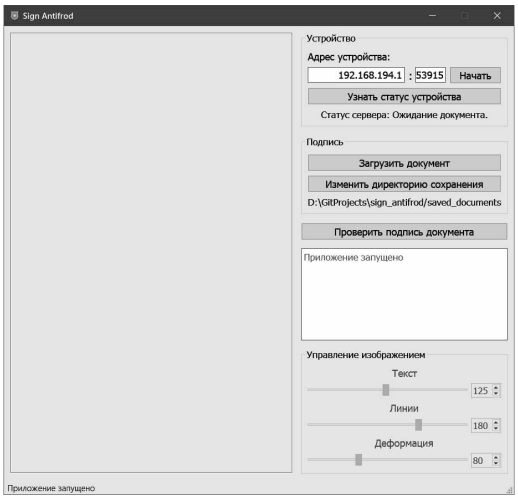

Рис. 4. Вид приложения после подключения к серверу

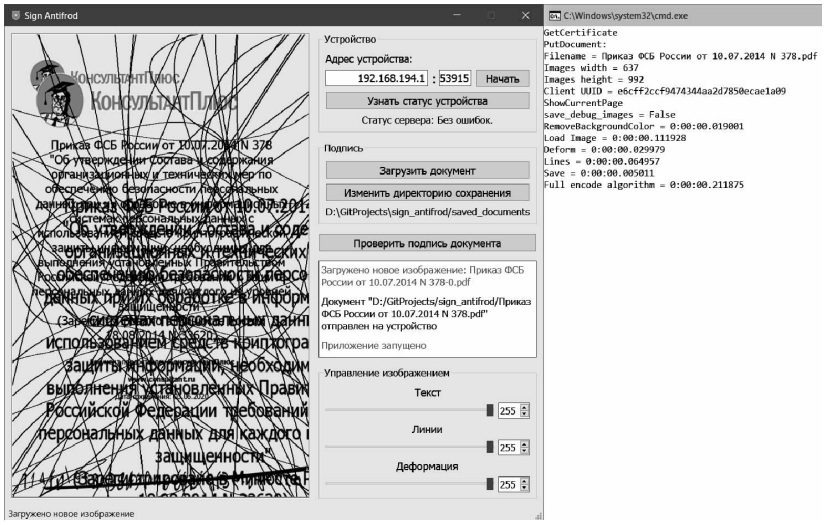

Рис. 5. Загрузка документа на сервер и отображение первой страницы

подлинности, показанным на странице, можно убедиться, что они совпадают (рис. 7).

Прервать обработку документа можно, нажав клавишу 2. После нажатия этой клавиши доверенное устройство очистит все данные о текущей сессии и выведет на дисплей «Successfully aborted», а на клиентском устройстве отобразится информация о том, что подписание было

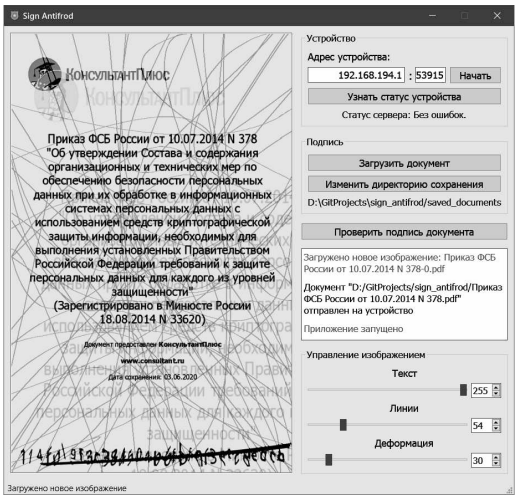

Рис. 6. Управление изображением с помощью слайдеров

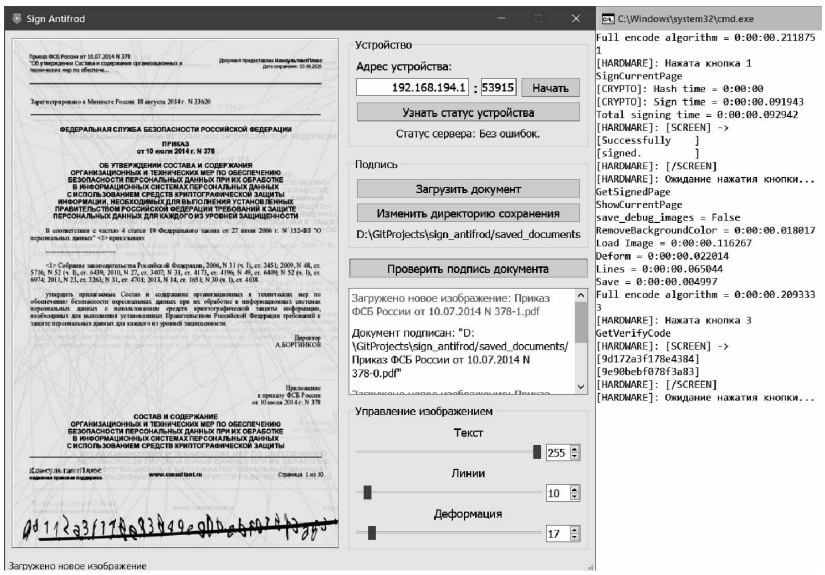

Рис. 7. Подписание документа и проверка подписи

прервано (рис. 8).

Проверка подписи документа идёт на клиенте после нажатия кнопки «Проверить подпись документа» и выбора PDF-файла для верификации. Пример результата выполненной верификации подписанной страницы отображён на рис. 9.

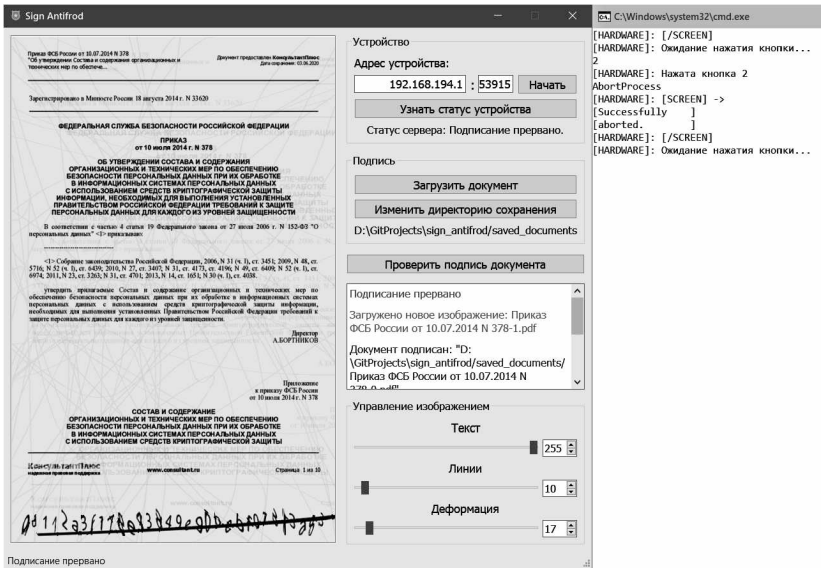

Рис. 8. Отмена обработки документа нажатием на кнопку 2

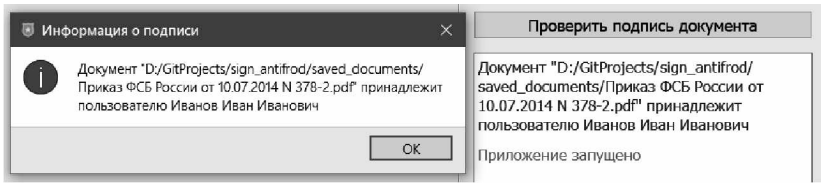

Рис. 9. Результат проверки ЭП ранее подписанной страницы

Все элементы программной части, в том числе логика взаимодействия аппаратной части комплекса с программной, функционируют корректно и без ошибок.

3.2. Тестирование программно-аппаратного комплекса в сборе

После успешного развертывания исходного кода, установки всех необходимых python-модулей и запуска аппаратной части с использованием модуля **hardware.py**, на экране **Raspberry Pi** отобразился адрес доверенного устройства (рис. 10a).

Статус устройства можно узнать, нажав на кнопку 3 (рис. 10b). С помощью клиентского приложения загрузим многостраничный PDF-документ. Спустя какое-то время на экране клиентского приложения

отобразилась первая защищённая страница. Проверим её код подлинности, нажав на кнопку 3 (рис. 10c).

Код подлинности выводится на аппаратное устройство корректно. В кодах подлинности на защищённой картинке и на дисплее устройства расхождений не обнаружено. Подпишем текущую страницу, нажав на кнопку 1. Спустя некоторое время статус устройства обновился (рис. 10d) и на клиентской стороне записались PDF-файл страницы документа и «.sign» файл ЭП.

После подписания первой страницы на экране клиента появилась вторая защищённая страница. Прервём обработку документа нажатием на кнопку 2 (рис. 10e). Обработка документа была прервана, и доверенное устройство ждёт дальнейших указаний.

Рис. 10. Работа аппаратной части устройства

Было проведено тестирование времени выполнения вложенных алгоритмов и сравнение показателей работы серверного кода **Raspberry Pi** с показателями, полученными в результате эмуляции данного устройства. Введем обозначения: T_{RPi} – время выполнения операции на **Raspberry Pi**; T_{em} – время выполнения операции на эмуляторе. Результаты тестирования представлены в таблице. Тестирование было проведено на примере десяти PDF-страниц одного документа.

Исходя из проведённого тестирования операции на **Raspberry Pi** функционируют исправно, но в среднем занимают в 7.044 раз больше времени, чем аналогичные операции, произведённые на полноценном персональном компьютере. При этом наибольшее время занимает работа с PFD-файлами, а именно: разбиение документов на отдельные

Таблица

Среднее время исполнения вложенных серверных алгоритмов

Осуществляемая операция	T_{RPi}	T_{em}	T_{RPi}/T_{em}
Разбиение документа на страницы (документы объёмом до 3 МБ)	5.595 с	0.651 с	8.59
Генерация PNG из PDF	2.246 с	0.431 с	5.21
Генерация защитного рисунка (637x992)	1.419 с	0.216 с	6.57
Время между клиентским запросом и получением защищённой картинки	4.994 с	0.836 с	5.97
Подписание страницы документа ЭП (страницы до 400 кб)	0.835 с	0.094 с	8.88

страницы, конвертация страниц в растровые изображения.

Для частичной оптимизации данной проблемы можно перевести разделение PDF-документов на сторону клиента, при этом оставив и оптимизировав механизм конвертации страниц в изображения на стороне доверенного устройства. Из таблицы видно, что на эмуляторе эта операция выполнялась в 8.59 раз быстрее, следовательно, и на стороне клиента она будет выполняться почти во столько же раз быстрее. В этой ситуации общее время всех операций уменьшится почти в 1.5 раза.

4. Заключение

В данной работе представлен прототип программно-аппаратного комплекса для обеспечения информационной безопасности при использовании электронных подписей.

Однако, в процессе тестирования было выяснено, что серверный код на **Raspberry Pi 3 model A+** работает недостаточно быстро. В связи с этим для комфортного использования данного программно-аппаратного комплекса необходимо задействовать более производительное аппаратное устройство (например, **Raspberry Pi** 4-го поколения, так как производительность его процессора в три раза быстрее [6]) и более эффективный серверный код, в частности модуль для разделения PDF-документов на отдельные страницы и конвертации этих страниц в растровые изображения, так как эти операции вместе занимают наибольшее время.

Разработанный прототип программно-аппаратного комплекса выполняет безопасное подписание страниц документов с помощью электронной цифровой подписи, обеспечивая при этом защиту данного механизма от компрометации. Использование данного комплекса будет удачным решением для повседневной работы за счёт эргономичного форм-фактора.

Следует отметить, что данный прототип в случае его внедрения в электронный документооборот организаций потребует дальнейших исследований в части поиска более производительного аппаратного устройства и оптимизации кода программного обеспечения. Также потребуется рассмотрение вопроса сертификации разработанного устройства как средства защиты информации.

Список литературы

1. **ГОСТ Р 34.10-2012** Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи. URL: http://docs.cntd.ru/document/gost-r-34-10-2012 (дата обращения: 09.01.2021).

2. Raspberry Pi 3 Model A+. URL: https://www.raspberrypi.org/products/raspberry-pi-3-model-a-plus/ (дата обращения: 09.01.2021).

3. Python bindings for the Qt cross platform application toolkit. URL: https://pypi.org/project/PyQt5/ (дата обращения: 09.01.2021).

4. Документация библиотеки Pillow. URL: https://pillow.readthedocs.io/en/stable/# (дата обращения: 09.01.2021).

5. A high-performance open source universal RPC framework. // URL: https://grpc.io (дата обращения: 09.01.2021).

6. Raspberry Pi 4 vs Pi 3 – В чем различия? URL: https://cnx-software.ru/2019/06/25/raspberry-pi-4-vs-pi-3-v-chem-razlichiya/ (дата обращения: 09.01.2021).

Summary

Gertner D. A., Leontiev D. O., Nosov L. S., Shuchalin D. S. Hardware and software complex for ensuring information security when using electronic signatures

This paper presents prototype of a software and hardware complex for secure signing of electronic documents on the basis of a trusted device specialized for these operations.

Keywords: hardware and software system, electronic digital signature, Raspberry Pi, Python.

References

1. **GOST R 34.10-2012** Information technology. Cryptographic information protection. Processes for generating and verifying electronic digital signature, Available at: http://docs.cntd.ru/document/gost-r-34-10-2012 (accessed: 09.01.2021).

2. Raspberry Pi 3 Model A+. Available at: https://www.raspberrypi.org/products/raspberry-pi-3-model-a-plus/ (accessed: 09.01.2021).

3. Python bindings for the Qt cross platform application toolkit, Available at: https://pypi.org/project/PyQt5/ (accessed: 09.01.2021).

4. Pillow Library Documentation, Available at: https://pillow.readthedocs. io/en/stable/# (accessed: 09.01.2021).

5. A high-performance open source universal RPC framework, Available at: https://grpc.io (accessed: 09.01.2021).

6. Raspberry Pi 4 vs Pi 3 – What are the differences? Available at: https://cnx-software.ru/2019/06/25/raspberry-pi-4-vs-pi-3-v-chem-razlichiya/ (accessed: 09.01.2021).

Для цитирования: Гертнер Д. А., Леонтьев Д. О., Носов Л. С., Шучалин Д. С. Программно-аппаратный комплекс для обеспечения информационной безопасности при использовании электронных подписей // *Вестник Сыктывкарского университета. Сер. 1: Математика. Механика. Информатика. 2021. Вып. 1 (38). С. 43–55. DOI: 10.34130/1992-2752_2021_1_43.*

For citation: Gertner D. A., Leontiev D. O., Nosov L. S., Shuchalin D. S. Hardware and software complex for ensuring information security when using electronic signatures, *Bulletin of Syktyvkar University. Series 1: Mathematics. Mechanics. Informatics*, 2021, 1 (38), pp. 43–55. DOI: 10.34130/1992-2752_2021_1_43.

ООО «Крейф»
СГУ им. Питирима Сорокина

Поступила 09.02.2021