JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 11 (2019) 101-105
Wysłane: 2018-12-01
Przyjęte: 2018-12-06

Porównanie wydajności algorytmów szyfrowania na platformie iOS

Jakub Tudruj*, Piotr Kopniak
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Niniejszy artykuł dotyczy porównania wydajności algorytmów szyfrowania używanych na mobilnej platformie iOS. Skupiono się na analizie algorytmów kryptografii symetrycznej i asymetrycznej oraz przetestowano, ile czasu zajmują im operacje szyfrowania i generowania klucza. Badania przeprowadzono na różnych urządzeniach działających na najwyższej wspieranej wersji systemu operacyjnego.

Słowa kluczowe: algorytmy szyfrujące; system iOS; wydajność szyfrowania

*Autor do korespondencji.

Adres e-mail: jakub.tudruj@pollub.edu.pl

Comparison of encryption algorithms performance on iOS platform

Jakub Tudruj*, Piotr Kopniak
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B,20-618 Lublin, Poland

Abstract. The subject of this article is to compare iOS mobile operation system encryption algorithms performance. The main target is to analyze symmetrical and asymmetrical algorithms, test them how much time encryptionand key generation operations take. Tests were carried out on various devices with the same version of operating system.

Keywords: encryption algorithms; iOS system; encryption performance

*Corresponding author.

E-mail address: jakub.tudruj@pollub.edu.pl

1. Wstęp

Coraz większa popularność smartfonów i tabletów oraz ich wykorzystanie w codziennym życiu sprawia, że producenci oprogramowania coraz częściej wprowadzają na rynek mobilne wersje swoich aplikacji. Przekłada się to na zwiększającą sięliczba programów dostępnych w sklepach Google Play oraz App Store. Rzutuje to bezpośrednio na coraz większą liczbę danych przetwarzanych, przesyłanych oraz przetrzymywanych na urządzeniach mobilnych.

Na co dzień wielu użytkowników smartfonów przetwarza wrażliwe dane, takie jak wiadomości, zdjęcia, dane lokalizacyjne, nagrania rozmów, notatki, informacje o swoim stanie zdrowia czy dostępy do kont społecznościowych lub bankowości internetowej. Może to rokować na duże zainteresowanie takimi informacjami przez potencjalnych hakerów lub przypadkowe osoby mogące uzyskać dostęp do skradzionego lub zagubionego urządzenia mobilnego. Wyciek tych danych może być niezwykle bolesny ze względu na możliwość wykorzystania ich do oczernienia ich właściciela, pozyskania informacji o jego rodzinie, kradzieży własności intelektualnej czy utraty środków na koncie.

Fakty te sprawiają, że zarówno użytkownicy smartfonów jak i programiści powinni zwracać uwagę na bezpieczeństwo informacji użytkowników. Jednym ze sposobów na zabezpieczenie wrażliwych treści jest użycie szyfrowania.

Niniejszy artykuł ma za zadanie porównać wydajności algorytmów szyfrowania symetrycznego i asymetrycznego oraz czasu na generowanie klucza kryptograficznego. Dane te mogą być niezwykle przydatne dla twórców oprogramowania. Dzięki nim możliwe będzie wprowadzenie odpowiedniego kompromisu pomiędzy siłą zabezpieczeń a wydajnością aplikacji mobilnej.

2. Cel i przedmiot badań

Przedmiotem badań byłaanaliza porównawcza szybkości generowania kluczy kryptograficznych oraz szyfrowaniadanych.

Celem badań była próba potwierdzenia tezy: Czas szyfrowania tych samych danych za pomocą tych samych algorytmów szyfrujących opartych o te same klucze kryptograficzne oraz generowanie tych samych kluczy kryptograficznych zajmuje mniej czasu na nowszym urządzeniu.

Kolejnym celem badań była próba potwierdzenia hipotez:

1) Zastosowanie klucza kryptograficznego o większej długości zwiększa czas szyfrowania tych samych danych,
2) Czas wygenerowania klucza kryptograficznego lub pary kluczy kryptograficznych wzrasta proporcjonalnie do zwiększenia się długości generowanego klucza lub pary kluczy.

3. Opis metodologii badań i narzędzi

W celu przeprowadzenia badania i uzyskania wyników czasowych działania poszczególnych algorytmów szyfrujących wykorzystano zestaw narzędzi deweloperskich dostarczanych przez firmę Apple. Kod aplikacji służącej dokonaniu pomiarów został napisany w języku Swift przy pomocy środowiska programistycznego Xcode. Badanie zostało przeprowadzone czterokrotnie na każdym z urządzeń w celu zminimalizowania błędu pomiarowego spowodowanego procesami uruchomionymi w tle. Następnie wyniki każdego z badań zostały uśrednione.

3.1. Urządzenia testowe

Badanie przeprowadzono z użyciem smartfonów:

- iPhone 5,
- iPhone 6 Plus,
- iPhone 6S,
- iPhone X.

Wszystkie urządzenia działają w oparciu o systemem najnowszą wspieraną wersję systemu iOS. W przypadku iPhone 5 jest to wersja 10.3, natomiast reszta smartfonów posiada zainstalowaną wersję oznaczoną numerem 12. Ich specyfikacja została wyszczególniona w Tabeli 1.

Tabela 1. Specyfikacja urządzeń testowych [5]

Model urządzenia	Procesor	Pamięć RAM
iPhone 5	Apple A6 1.3 GH, 2 rdzenie	1 GB
iPhone 6 Plus	Apple A8 1.4 GHz, 2 rdzenie	1 GB
iPhone 6S	Apple A9 1.85 GHz, 2 rdzenie	2 GB
iPhone X	Apple A11 Bionic 2.39 GHz, 6 rdzeni	3 GB

3.2. Testowane algorytmy

Do przeprowadzenia badania wybrano dwa najpopularniejsze algorytmy kryptografii asymetrycznej: RSA oraz ECC. Poniżej przedstawiono badane długości kluczy.

1) RSA:
   - 1024 bity
   - 2048 bitów
   - 4096 bitów
   - 8192 bity,
   - 15130 bitów,
2) ECC:
   - 160 bitów,
   - 224 bity,
   - 256 bitów,
   - 384 bity,
   - 512 bitów.

Zbadano również symetryczny algorytm AES używając następujących długości klucza:

- 128 bitów,
- 192 bity,
- 256 bitów.

Z powodu ograniczenia wielkości danych możliwych do zaszyfrowania za pomocą algorytmów asymetrycznych, w pierwszej części badań skupiono się na sprawdzeniu wydajności szyfrowania klucza AES o długości 128 bitów. Druga część badań polegała na zaszyfrowaniu dużej ilości danych za pomocą klucza AES. Przetestowano szyfrowanie krótkiego tekstu o długości 32 znaków, długiego tekstu składającego się z 10093 znaków oraz dwóch plików graficznych. Mniejsza z grafik miała rozmiar 22 KB, natomiast większa 1,8 MB.

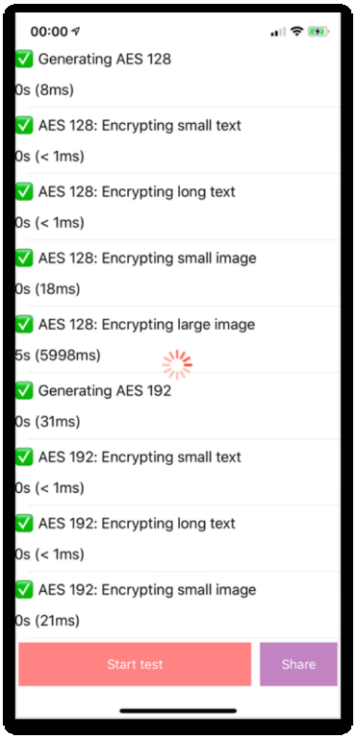
Poniżej (Tabela 2) przedstawiono porównanie bezpieczeństwa w zależności od długości klucza algorytmów ECC oraz RSA [3] [4].

Tabela 2. Porównanie bezpieczeństwa RSA oraz ECC w zależności od długości klucza[3] [4]

Algorytm		
Symetryczny	**RSA**	**ECC**
SKIPJACK 80 bitów	1024 bity	160 bitów
3DES 112 bitów	2048 bitów	224 bity
AES 128 bitów	3072 bity	256 bitów
AES 192 bity	8192 bity	384 bity
AES 256 bitów	1530 bitów	512 bitów

3.3. Opis aplikacji testowej

Aplikacja testowa została napisana w języku Swift w wersji 4.2 przy użyciu środowiska programistycznego Xcode 10. Dostęp do funkcji umożliwiających generowanie kluczy oraz szyfrowanie uzyskano dzięki wykorzystaniu biblioteki CommonCrypto dostarczanej przez firmę Apple. Poniżej (Rysunek 1) przedstawiono zrzut ekranu aplikacji uruchomionej na urządzeniu iPhone X.



Rys. 1. Urządzenia testowe podczas przeprowadzania badania

Aplikacja umożliwia podgląd na żywo wyników poszczególnych etapów badania oraz eksport wszystkich pomiarów używając formatu danych pliku csv.

Kod aplikacji testowej został umieszczony na publicznym repozytorium w serwisie GitHub pod adresem: https://github.com/JakubTudruj/encryptionTimeTest.

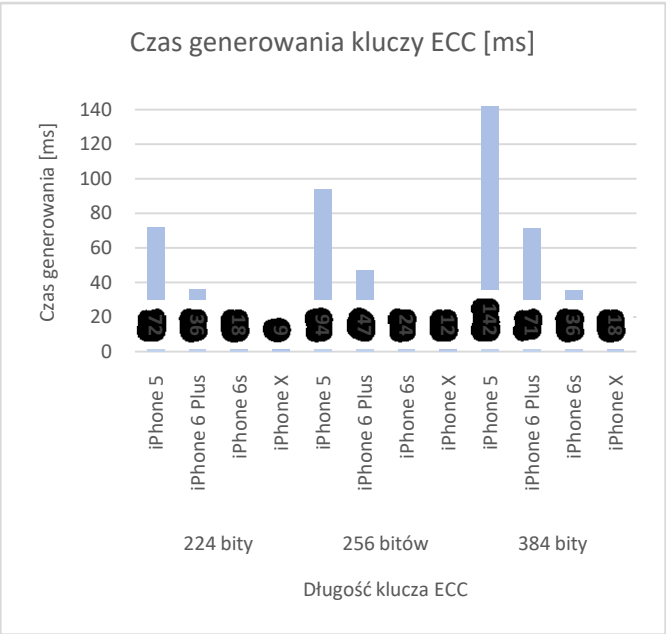
4. Wyniki badań

Podczas próby przeprowadzania testów okazało się, że biblioteka CommonCrypto nie wspiera tworzenia kluczy ECC o długości 160 bitów oraz 512 bitów i więcej. W przypadku kluczy RSA maksymalną długością okazała się para kluczyo długości 8192 bitów. Wynikiem operacji był błąd „OSStatus -50”, który przedstawiono na Przykładzie 1.

Przykład 1. Błąd biblioteki CommonCrypto przy generowaniu pary kluczy o niedozwolonej długości

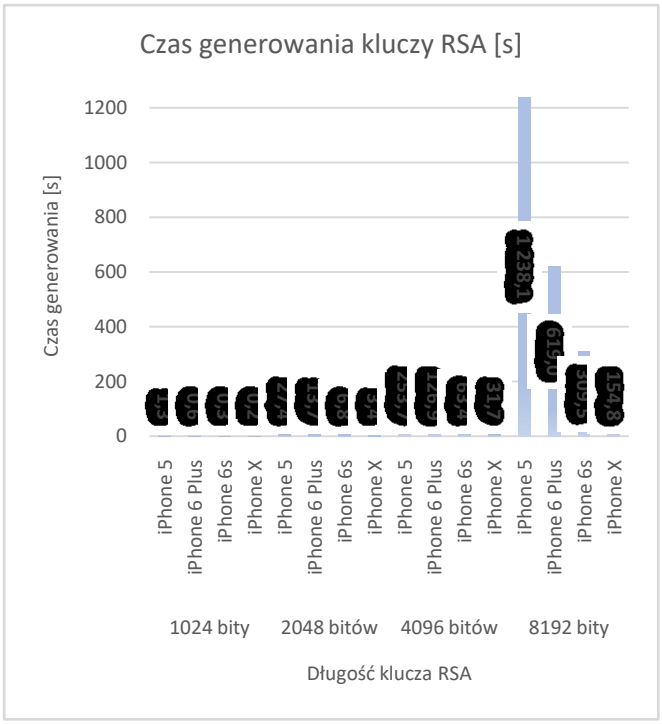
```
The operation couldn't be completed. (OSStatus error -50 -
 Key generation failed, error -50)
```

Wyniki badania czasu generowania par kluczy ECC o długościach 224 bity, 256 bitów oraz 384 bity wykazały, że bez względu na długość klucza czas potrzebny na jego wygenerowanie jest na tyle krótki (poniżej 142 ms), że z powodzeniem można stosować nawet najdłuższy możliwy klucz na starszych urządzeniach. Użyty do testów iPhone 5 mający jako jedyny procesor 32 bitowy, który jednocześnie charakteryzował się najwolniejszym zegarem, generował parę kluczy ECC w czasie kolejno: 72, 94 oraz 142 milisekund. Dwukrotnie szybszy w każdym przypadku okazał się jego następca: iPhone 6 Plus. Każde kolejne urządzenie było również dwukrotnie szybsze od swojego poprzednika. Wyniki tej części badania zostały przedstawione na Rysunku 2.



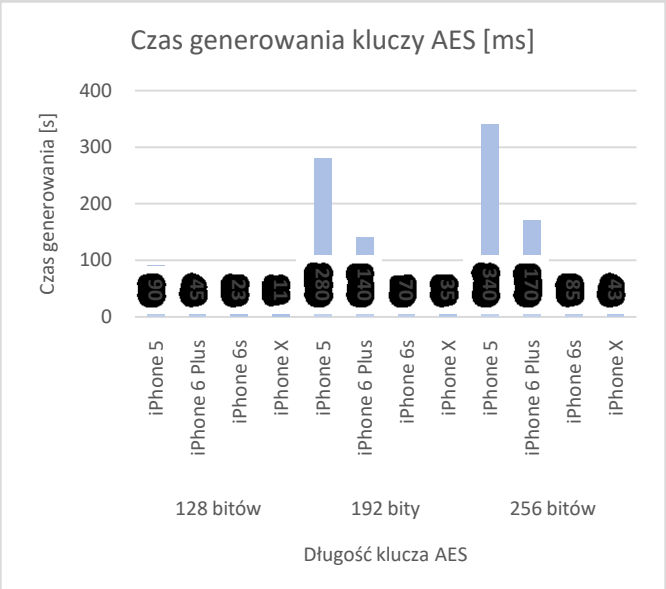
Rys. 2. Wykres przedstawiający czas generowania pary kluczy ECC o różnych długościach na urządzeniach testowych

W kolejnej części badania sprawdzono czas generowania par kluczy RSA o długościach od 1024 bitów do 8192 bitów. Podobnie jak podczas poprzedniego badania okazało się, że nowsze urządzenia są szybsze od starszych oraz że długość pary kluczy przekłada się na czas potrzebny do ich wygenerowania. Operacja generowania pary kluczy RSA charakteryzowała się niską wydajnością dla urządzeń mobilnych. Najkrótsza z par kluczy o długości 1024 bitów na każdym urządzeniu, oprócz iPhone 5, generowała się w czasie krótszym niż sekunda. Każdorazowe zwiększenie długości kluczy skutkowało znacznym wzrostem czasu potrzebnego do ich wygenerowania. Przekroczenie czasu 3 sekund na najwydajniejszym urządzeniu testowym (iPhone X) dla pary kluczy RSA 2048 jest ostatnią długością klucza, która powinna być akceptowalna w aplikacji mobilnej, aby ta zachowała płynność działania [2].Najbardziej czasochłonną operacją było generowanie najdłuższej pary kluczy RSA o długości 8192 bitów. iPhone 5 potrzebował na to aż 1238 sekund (ponad 20 minut). Nawet iPhone X wyróżniający się najszybszym procesorem uzyskał wynik 154 sekund (2,5 minuty). Powyższe dane wskazują, że algorytm RSA jest mało praktyczny w zastosowaniach mobilnych. Wyniki pomiarów zostały przedstawione poniżej (Rysunek 3).

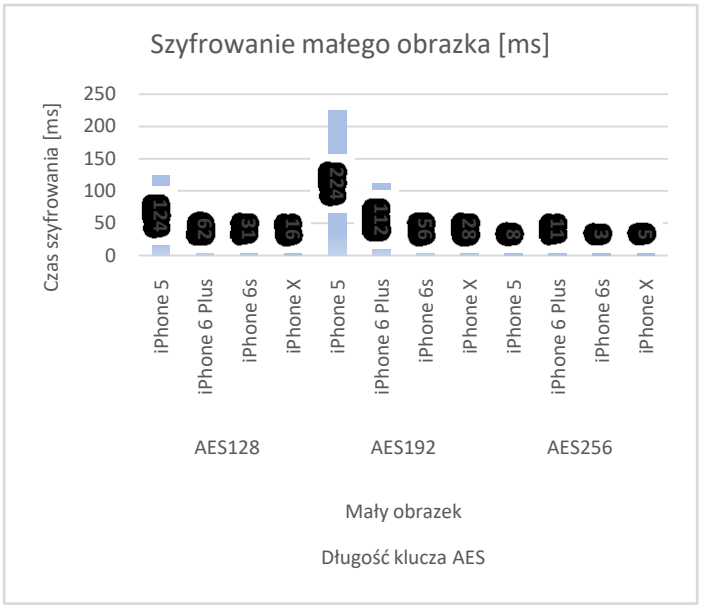


Rys. 3. Wykres przedstawiający czas generowania pary kluczy RSA o różnych długościach na urządzeniach testowych

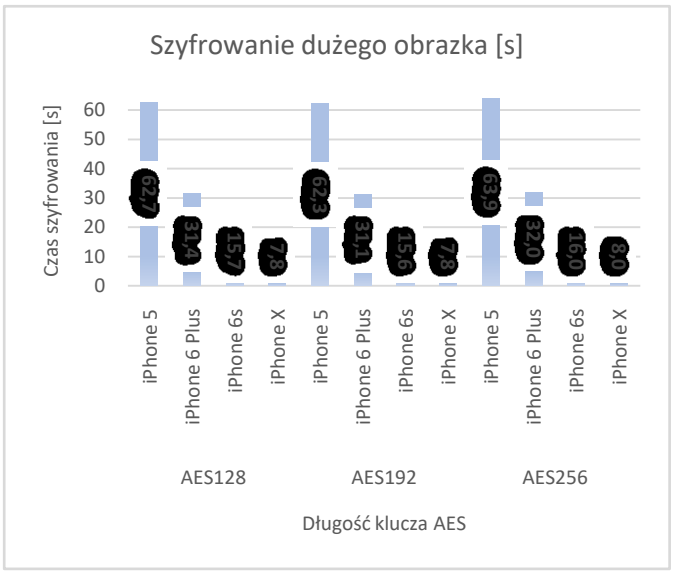
Ostatnia część badania w której skupiono się na generowaniu symetrycznego klucza AES wykazała, że podobnie jak w poprzednich przypadkach, czas generowania kluczy jest tym dłuższy, im dłuższy jest klucz szyfrujący. Większa moc obliczeniowa urządzenia skutkowała skróceniem czasu potrzebnego na wygenerowanie klucza. Nie przekraczał on jednak 340 milisekund. Wyniki tej części badania zostały przedstawione na Rysunku 4.



Rys. 4. Wykres przedstawiający czas generowania symetrycznego klucza AES o różnych długościach na urządzeniach testowych



Rys. 5. Wykres przedstawiający czas szyfrowania małego obrazka za pomocą kluczy AES o różnych długościach na urządzeniach testowych



Rys. 6. Wykres przedstawiający czas szyfrowania dużego obrazka za pomocą kluczy AES o różnych długościach na urządzeniach testowych

Po przeprowadzeniu operacji generowania kluczy kryptograficznych w następnej części zadania sprawdzono w jakim czasie wykonana zostanie operacja szyfrowania danych. W przypadku zaszyfrowania klucza AES 128 bitów za pomocą symetrycznego klucza publicznego okazało się, że bez względu na zastosowany algorytm oraz jego długość czas nie przekraczał 1 milisekundy.

Badanie szyfrowania kluczami AES o różnych długościach wykazało, że bez względu na zastosowaną długość klucza operacja zajmowała na tych samych urządzeniach podobną ilość czasu. Znaczenie miała tylko wydajność samego urządzenia testowego. Szyfrowanie długiego oraz krótkiego tekstu wykazało, że potrzebny czas nie przekraczał 1 milisekundy na każdym z urządzeń. Różnice wykazał test szyfrowania większej ilości danych. Szyfrowanie małego obrazka (22KB) nie przekraczało 224 milisekund. Co ciekawe, badanie wykazało, że najszybciej operacja przebiegła dla najdłuższego klucza AES 256 bitów, najwolniejszy z kolei okazała się operacja szyfrowania kluczem o długości 192 bitów. Czas tych operacji jest na tyle mały, że wyniki te mogły wynikać z błędu pomiarowego.

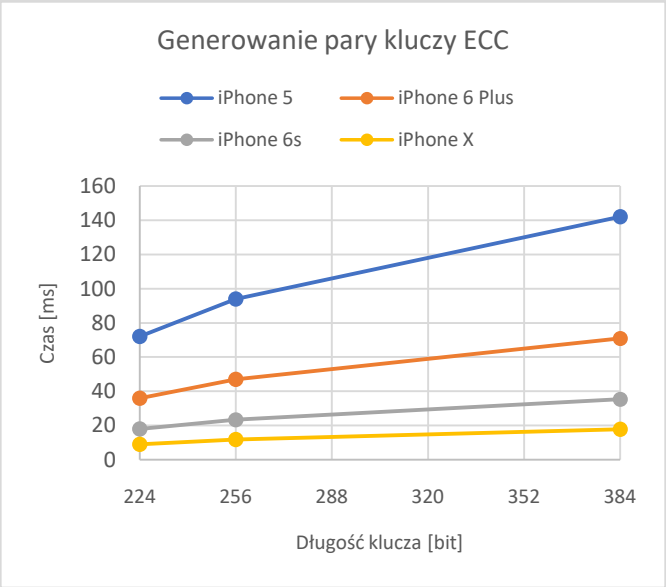
Każda kolejna generacja smartfonu z systemem iOS wykazywała się o połowę lepszym czasem szyfrowania dużego obrazka (1,8 MB) od swojego poprzednika. Podczas, gdy iPhone 5 potrzebował ok 1 minuty, iPhone X wykonywał tę samą operację w czasie nie przekraczającym 8 sekund.

Wyniki powyższego badania przedstawiono poniżej (Rysunku 5, Rysunek 6).

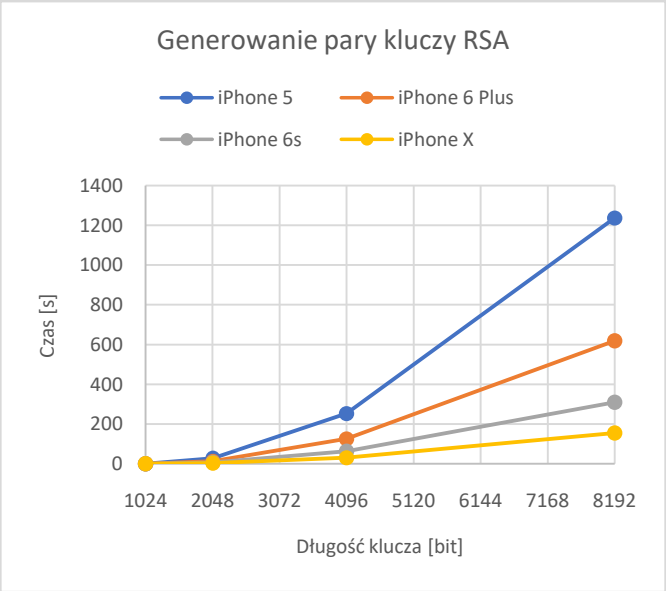
Powyższa część badania zaprzeczyła hipotezie 1 mówiącej, że zastosowanie klucza kryptograficznego o większej długości zwiększa czas szyfrowania tych samych danych. Jednocześnie też potwierdzona została hipoteza 2 mówiąca, że czas wygenerowania klucza kryptograficznego lub pary kluczy kryptograficznych wzrasta proporcjonalnie do zwiększenia się długości generowanego klucza lub pary kluczy.

Porównanie szybkości generowania symetrycznych kluczy kryptograficznych na różnych urządzeniach wykazało, że nowsze urządzenia są w stanie wykonać tę operację szybciej, niż ich starsze odpowiedniki. Na Rysunku 7 przedstawiono wykres zależności długości klucza ECC od czasu jego generowania. Zauważyć na nim można, że wykres wzrasta liniowo, natomiast w przypadku pary

kluczy RSA (Rysunek 8) wykres posiada charakterystykę rosnącą wykładniczo.



Rys. 7. Wykres przedstawiający porównanie czasu generowania pary kluczy ECC



Rys. 8. Wykres przedstawiający porównanie czasu generowania pary kluczy RSA

Przedstawione wyniki potwierdzają postawioną tezę: czas szyfrowania tych samych danych za pomocą tych samych algorytmów szyfrujących opartych o te same klucze kryptograficzne oraz generowanie tych samych kluczy kryptograficznych zajmuje mniej czasu na nowszym urządzeniu.

5. Wnioski

W niniejszym artykule skupiono się na porównaniu czasu generowania pary kluczy RSA i ECC oraz symetrycznego klucza AES o różnych długościach. Następnie zmierzono czas szyfrowania danych różnej wielkości na różnych urządzeniach testowych opartych o najnowszą wspieraną wersję systemu iOS. Nowsze urządzenia wyposażone w szybszy procesor oraz większą ilość pamięci operacyjnej wykazały się większą wydajnością od swoich poprzedników. Przewaga czasowa wynosiła często połowę czasu. Potwierdziło to postawioną tezę.

Wyniki badań wykazały, że operacja generowania pary kluczy ECC zajmuje o wiele mniej czasu niż generowanie pary RSA. Potwierdza to słuszność i popularność wykorzystania pierwszego z algorytmów w wielu aplikacjach i rozwiązaniach w urządzeniach mobilnych. Przykładem może być wymiana kluczy między urządzeniem mobilnym a serwerem w celu nawiązania bezpiecznego połączenia SSL (handshake) [1]. Czas generowania kluczy kryptografii symetrycznej nie ma jednak odzwierciedlenia w czasie szyfrowania klucza symetrycznego. W każdym przypadku jest on znikomy i nie przekracza 1 milisekundy. Stanowi to zaprzeczenie postawionej hipotezy mówiącej, że zastosowanie klucza kryptograficznego o większej długości zwiększa czas szyfrowania tych samych danych.

Wykazano również, że zwiększanie bezpieczeństwa danych poprzez zastosowanie dłuższego klucza AES nie wpływa negatywnie na czas wykonywania operacji szyfrowania na urządzeniu mobilnym, bez względu na ich wielkość, co jest potwierdzeniem hipotezy 1: „czas wygenerowania klucza kryptograficznego lub pary kluczy kryptograficznych wzrasta proporcjonalnie do zwiększenia się długości generowanego klucza lub pary kluczy. Szyfrowanie niewielkiej ilości danych obarczone zostało dużym błędem pomiarowym. Operacja zajmowała niewiele czasu, a sam błąd mógł być spowodowany procesami działającymi w tle na urządzeniach testowych.

Literatura

[1] Binnie C.: Linux Server Security: Hack and Defend, Wiley, 2016.

[2] Butkiewicz M., Madhyastha H. V., Sekar V., Wang D, Wu Z.: Reprioritizing Web Content to ImproveUser Experience on Mobile Devices, University of Michigan, https://www.usenix.org/system/files/conference/nsdi15/nsdi15-paper-butkiewicz.pdf, [10.11.2018]

[3] Hechenblaikner C., StrombergerC., Teufl P., Zefferer T.: iOS Encryption Systems, Deploying iOS Devices in Security-critical Environments, Institute for Applied 120 Information Processing and Communications, Graz University of Technology, Austria,2013.

[4] PERFORMANCE ANALYSIS OF ELLIPTIC CURVE MULTIPLICATION ALGORITHMS FOR ELLIPTIC CURVE CRYPTOGRAPHY, https://etd.lib.metu.edu.tr/upload/12607698/index.pdf, [10.11.2018]

[5] Support Apple: Identify your iPhoone model, https://support.apple.com/en-us/HT201296, [20.11.2018].