Kryptografia Stosowana

Trivium - prosty szyfr strumieniowy

1 Wstęp

W ostatnich latach można zaobserwować tendencję odchodzenia od szyfrów strumieniowych. Ich godnym następstwem są coraz szerzej stosowane szyfry blokowe. Dzieje się tak, mimo iż te pierwsze posiadają pewne zalety, których nie należy ignorować. Symetryczny szyfr strumieniowy Trivium może być tego przykładem. Pomysł na Trivium był taki, aby stworzyć możliwie prosty szyfr strumieniowy. Miał być on zarówno bardzo prosty, bezpieczny oraz elastyczny. Miał także być stworzony na podstawie metodologii tworzenia i działania szyfrów blokowych, zamieniając jednak elementy budujące szyfr blokowy odpowiadającymi im komponentami szyfrów strumieniowych.

Niniejsza praca ma na celu przegląd stanu sztuki na temat szyfru Trivium, jego krótki opis, a także analizę funkcjonalną oraz analizę bezpieczeństwa.

W rozdziale drugim przedstawiono krótkie porównanie szyfrów blokowych oraz strumieniowych. Następnie w rozdziale trzecim przedstawiono wprowadzanie oraz rys historyczny szyfru Trivium. W kolejnym rozdziale zostały omówione aspekty techniczne szyfru, natomiast w rozdziale piątym przedstawiono analizę bezpieczeństwa i efektywności.

2 Szyfry blokowe vs szyfry strumieniowe

Zarówno szyfry blokowe jak i strumieniowe zaliczają się do kategorii szyfrów symetrycznych. Oznacza to, że używają jednego klucza do przeprowadzenia operacji szyfrowania tekstu jawnego w tekst zaszyfrowany oraz deszyfrowania tekstu zaszyfrowanego, aby z powrotem otrzymać tekst jawny [2]. Najważniejszą różnicą między szyframi blokowymi, a szyframi strumieniowymi jest to, że blokowe dzielą tekst jawny na wiele mniejszych wiadomości o określonej długości - bloki i dopiero wtedy te mniejsze wiadomości są szyfrowane. W przypadku szyfrowania strumieniowego tekst do zaszyfrowania dzielony jest na pojedyncze bity i szyfrowany bit po bicie [3].

Szyfry blokowe są obecnie dużo szerzej stosowane niż szyfry strumieniowe. Ma to głównie związek z faktem, że te pierwsze są znacznie lepiej rozpoznane i zrozumiane przez osoby bezpośrednio zainteresowane tym tematem. W szczególności istotny jest tu poziom zrozumienia struktury szyfrów blokowych w odniesieniu do poziomu ich bezpieczeństwa. Szyfry strumieniowe również znajdują swoje nisze jak połączenia SSL/TLS, czy komunikacja Bluetooth, jednak ich zastosowanie jest obecnie w znacznym stopniu ograniczone i są coraz częściej zastępowane szyframi blokowymi [3].

Powodem dla którego szyfry strumieniowe nie zostały całkowicie wyparte przez szyfry blokowe jest to, że są one z reguły szybsze, przy jednoczesnym wykorzystaniu mniejszej

ilości zasobów. Fakt ten nie został zignorowany przez środowisko kryptograficzne i dzięki temu powstał m.in. szyfr Trivium, który ma być szyfrem strumieniowym, czerpiącym z zalet szyfrów blokowych.

3 Trivium - wprowadzenie

Trivium to bardzo prosty szyfr strumieniowy, który został zaprojektowany przez Christoph'a De Canniere'a i Bart'a Preneel'a. Został on po raz pierwszy zaprezentowany pod szyldem projektu eSTREAM. Projekt ten miał na celu poznawanie szyfrów strumieniowych, które potencjalnie mogą okazać się odpowiednie do szerokich zastosowań i został wybrany do prezentacji w portfolio projektu eSTREAM [11].

Szyfr Trivium powstał, aby pokazać że szyfry strumieniowe nie są jedynie wspomnieniem w środowisku kryptograficznym, a wciąż mogą być użyteczne. Celem projektu miało być przywrócenie zaufania do szyfrów strumieniowych, poprzez zaprojektowanie bardzo prostego szyfru, który będzie opierał się na jasnych i godnych zaufania kryteriach. Stąd pomysł, aby stworzyć szyfr strumieniowy, który czerpałby z dobrze poznanych kryteriów tworzenia szyfrów blokowych - a więc hybrydy, która jest bardzo dobrze określona, prosta w zrozumieniu i zarazem bezpieczna, a także bardzo wydajna i nie będzie wymagała dużej ilości zasobów do implementacji [5].

Głównym zastosowaniem tego szyfru miała być implementacja sprzętowa i tam należy doszukiwać się najlepszych osiągów, natomiast Trivium całkiem dobrze radzi sobie jako implementacja programowa i również w tym przypadku osiąga rozsądną wydajność. Szyfr ten mógłby potencjalnie znaleźć zastosowanie w sytuacjach, gdy wymagana jest duża szybkość, jednak zasoby fizyczne są bardzo ograniczone. Według literatury, szyfr Trivium jest szyfrem bardzo wydajnym, bezpiecznym oraz szybkim, przynajmniej w czasach, gdy był on tworzony [5], natomiast jego parametry wciąż są bardzo przyzwoite.

Jak podają oficjalne źródła projektu eSTREAM [5], Trivium nie powinien być szeroko stosowany (stan na 2006 r.), gdyż miał na celu sprawdzenie do jakiego stopnia można uprościć szyfr strumieniowy, bez zauważalnego wpływu na jego bezpieczeństwo, szybkość czy też elastyczność. W momencie publikowania tych zaleceń, najskuteczniejszym atakiem przeciwko Trivium był atak Brute Force, natomiast trzeba wziąć pod uwagę, że przeprowadzenie odpowiednich testów bezpieczeństwa każdego szyfru powinno trwać wiele lat. Brak efektywnych metod łamania szyfru może wiązać się z tym, że jest on bardzo bezpieczny lub z faktem, że jego wady nie zostały jeszcze odkryte. Aktualnie ciężko szukać jakiegokolwiek komercyjnego zastosowania szyfru Trivium, mimo iż powstał on ponad 15 lat temu. Sytuacja może się jednak zmienić wraz z rozwojem IoT (Internet of Things), gdyż Trivium (bądź pewne wariacje tego szyfru) może okazać się idealnym rozwiązaniem w świeci komunikujących się ze sobą urządzeń (M2M), w związku z jego niskimi wymaganiami na energię elektryczną,

elastyczność, prostotę i potencjalnie dużą odporność na wszelkie ataki [1][6].

Trivium oczywiście nie jest jedynym szyfrem, z kategorii lekkich szyfrów strumieniowych, które mogą być wykorzystane dla komunikacji urządzeń IoT. Jest to jednak bardzo ciekawy kandydat.

4 Trivium - aspekty techniczne

Jak już wspomniano, Trivium jest szyfrem strumieniowym. Jest on w stanie wygenerować 2^{64} bitów na wyjściu, na podstawie 80 bitowego klucza i 80 bitowego wektora inicjalizacji.

Z założenia jest on bardzo prosty i przeznaczony dla implementacji hardware'owych. Posiada on 288 bitów stanu w trzech rejestrach przesuwnych, których długość nie jest jednakowa. W najprostszej implementacji wymaga on 3488 bramek logicznych, natomiast możliwe są różne modyfikacje potencjalnie zwiększające szybkość kosztem większej ilości zasobów [11].

Do generowania strumienia klucza Trivium używa trzech połączonych ze sobą rejestrów przesuwnych jak na Rysunku 1. Na wyjściu otrzymujemy strumień s_i, który jest wynikiem sumy działania XOR wyjściach z tych rejestrów. Główną cechą tego szyfru jest wzajemne połączenie wyjść z wejściami rejestrów przez, co często przedstawiany jest na planie koła - z jednym kołowym rejestrem.

Na wejściu każdego z rejestrów liczona jest suma modulo 2 z wybranego bitu tego rejestru oraz z wyjścia na poprzedzającym go rejestrze, natomiast na wyjściu każdego rejestru liczona jest suma modulo 2 z wyjścia tego rejestru, wybranego bitu tego rejestru oraz wyniku operacji AND z dwóch bitów tego rejestru.

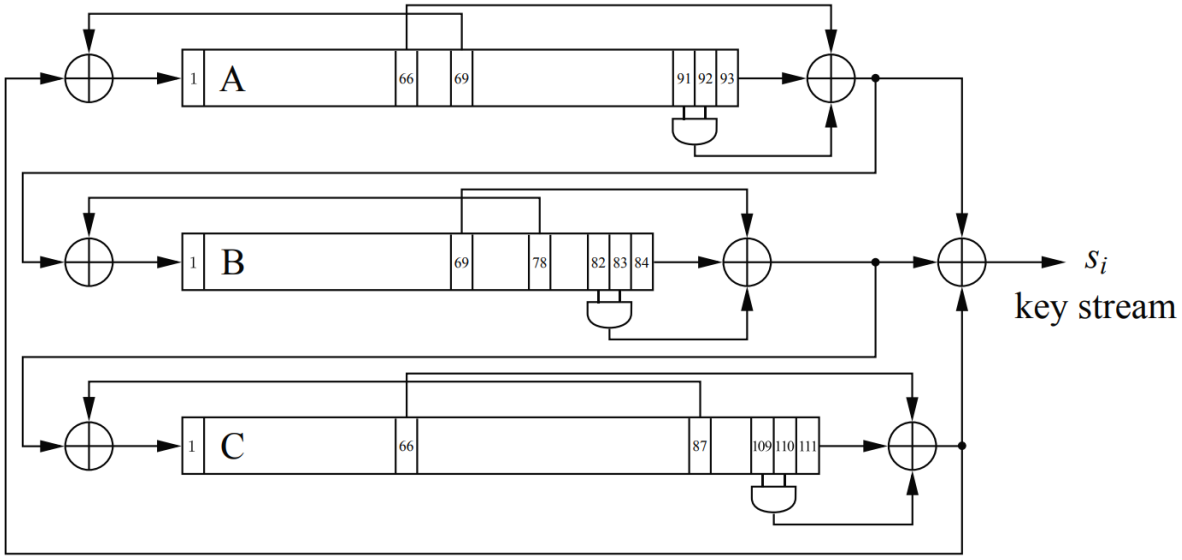
Jak zostało zauważone w [10], dzięki użyciu operacji AND wprowadzona zostaje nieliniowość co jest kluczowym czynnikiem dla bezpieczeństwa Trivium, jak i wielu innych szyfrów kryptograficznych.

Możemy wyróżnić 3 etapy działania szyfru:

1. **Etap inicjalizacji** — 80 bitowy wektor inicjalizacyjny umieszczany jest od lewej strony rejestru A, analogicznie umieszczany jest 80 bitowy klucz w rejestrze B. Wszystkie inne bity są zerowane za wyjątkiem trzech ostatnich bitów rejestru C — one ustawiane są na 1.

2. **Etap mieszania** — stan rejestrów jest wstępnie przygotowywany poprzez taktowanie zegara $4 * 288 = 1152$ razy. Dzięki wstępnemu taktowaniu wiemy, że stan wszystkich rejestrów zależny jest zarówno od klucza jak i wektora inicjalizacji oraz jest wystarczająco losowy do wykonywania następnego etapu.

3. **Etap szyfrowania** — od 1153 cyklu następuje szyfrowanie wiadomości ze strumieniem klucza.



Rysunek 1: Wewnętrzna struktura szyfru Trivium [10]

5 Analiza - bezpieczeństwo i efektywność szyfru

5.1 Bezpieczeństwo

Algorytm Trivium wykorzystuje 80-bitowy klucz oraz 80-bitowy wektor inicjalizacji do szyfrowania 64 bitów danych. Są to wartości, które w teorii mogą zapewnić akceptowalną odporność na ataki. Niestety w przypadku algorytmu Trivium zweryfikowanie tego nie jest łatwe i najlepszą ścieżką analizy jest omówienie znanych metod ataków.

5.1.1 Ataki korelacje

Pomimo, że bity wyjściowe algorytmu Trivium są liniowo zależne ze stanami wewnętrznymi, to niejasne jest, w jaki sposób możliwe jest odzyskanie wewnętrznego stanu automatu algorytmu na podstawie wyjścia, ponieważ ewolucja stanów algorytmu zachodzi w sposób nieliniowy. Ciężko przewidzieć co przyniesie przyszłość, natomiast na chwilę obecną odporność na ataki korelacyjne jest powyżej wymaganego minimum [4].

5.1.2 Okresowość

Rozpatrując ataki oparte na możliwości wystąpienia pewnych okresowych powtórzeń, ponownie należy odnieść się do nieliniowej ewolucji stanów szyfru. Według [4], ze względu

na tą nieliniowość, trudno określić co jaki okres szyfr się powtórzy, natomiast pewne uproszczenia - pomijanie bramek AND, które skutkują pojawieniem się liniowości w przejściu do kolejnych stanów, pokazują, że również tego typu ataki nie są dla Trivium zagrożeniem.

5.1.3 Guess and Determine Attacks

Teoretyczne rozważanie odnośnie tego ataku, które zostały przeprowadzone przez autorów szyfru w dokumencie [4], wskazują że aktualnie znane metody nie zagrażają bezpieczeństwu szyfru Trivium, jednak jest to kategoria ataków, która powinna budzić zainteresowanie. Według autorów cytowanego artykułu niewykluczone jest powstanie bardziej wyszukanych metod kryptoanalizy, które mogłyby potencjalnie wpłynąć na bezpieczeństwo szyfru Trivium.

5.1.4 Ataki algebraiczne

Pomimo, że algorytm Trivium daje się reprezentować przy pomocy układu wielomianów stosunkowo niskiego rzędu nie znaleziono skutecznego ataku algebraicznego, który umożliwiałby odzyskanie klucza. Dużą przeszkodą w analizie algorytmu jest nieliniowa ewolucja stanów. Z pracy Simona i Fischera Willi'ego Meiera wnika, że ciężko dla analizy algebraicznej algorytmu Trivium osiągnąć wystarczającą zależność równań, aby je rozwiązać [8].

5.1.5 Ataki resynchronizacji

Ataki resynchronizacji polegają na manipulacji wektorem inicjalizacji w celu zbadania zależności i odkrycia klucza prywatnego. Aby temu zapobiec algorytm Trivium inicjując się wykonuje 4 pełne rotacje maszyny stanów. Według M. Vielhabera powinno to być wystarczające, aby zabezpieczyć się przed tego typu atakami [12].

5.2 Efektywność

5.2.1 Implementacje sprzętowe

Algorytm Trivium został zaprojektowany z myślą o sprzętowych implementacjach. Ma on zapewniać wysokie bezpieczeństwo, szybkość i efektywność energetyczną dla urządzań, w których koszt implementacji oraz zużycie energii stanowią ograniczenia. Algorytm pozwala na wprowadzenie optymalizacji szybkości i efektywności energetycznej kosztem złożoności implementacyjnej. Możliwe jest zrównoleglenie do 64 iteracji algorytmu.

W realizacji sprzętowej zaproponowanej przez T. Good'a i M. Benaissa konieczne było wykorzystanie jedynie 4921 bramek NAND dla najbardziej rozbudowanej wersji algorytmu [9].

Na podstawie Tablicy 1 i 2, oraz Rysunku 2 można wywnioskować, że algorytm Trivium zapewnia wysoką efektywność energetyczną, jak również implementacyjną, wymagając

Komponenty	1-bit	8-bit	16-bit	32-bit	64-bit
Przerzutniki	288	288	288	288	288
Bramki AND	3	24	48	96	192
Bramki XOR	11	88	176	352	704
Bramki NAND	3488	3712	3968	4480	5504

Tablica 1: Porównanie liczby elementów w implementacjach o różnym poziomie zrównoleglenia [4].

Implementacja	Przep. [Mbps]	Est. moc [µW]	Energia/bit [pJ/bit]	Opóźnienie [µs]
Trivium	0.100	5.6	56.1	13330
Trivium x4	0.400	5.9	14.6	3360
Trivium x8	0.800	6.4	8.0	1700
Trivium x16	1.600	8.1	5.1	870
Trivium x32	3.200	20.3	3.2	450
Trivium x64	6.400	14.3	2.2	240
Salsa20 1h	0.099	26.3	264.0	5330
Salsa20 32	1.896	68.5	36.1	300

Tablica 2: Porównanie wybranych parametrów realizacji sprzętowych algorytmów w procesie CMOS 0.13µm, zegar taktujący 100 kHz [9]

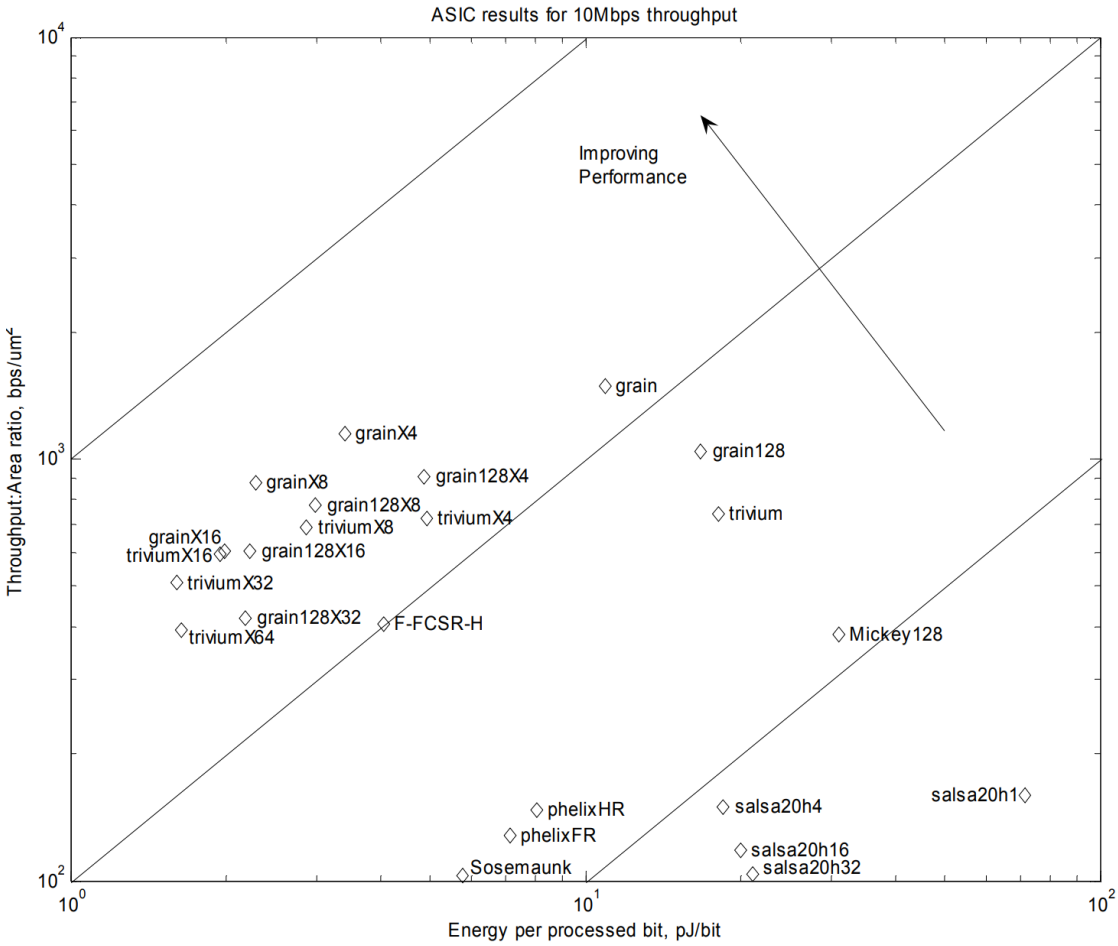
niewielkiej powierzchni krzemowej do realizacji. Zapewnia również możliwość optymalizacji na przestrzeni złożoności implementacji, przepustowości, efektywności energetycznej i opóźniania. W porównaniu do popularnego algorytmu szyfrowania strumieniowego Salsa20 zaobserwować można dużą nad nim przewagę.

5.2.2 Implementacje programowe

Pomimo, że algorytm Trivium nie był tworzony z myślą o implementacjach programowych, pozwala on na szyfrowanie z zadowalającą szybkością. Na procesorze Intel Pentium M CPU 1700 MHz referencyjna implementacja algorytmu w języku C umożliwia szyfrowanie jednego bitu w 5,3 cyklu zegarowego procesora [7]. Przekłada się to na ok. 320 Mbps na dość już leciwym procesorze bez instrukcji dla akceleracji tego algorytmu.

6 Wnioski

Trivium to synchroniczny szyfr strumieniowy, który z założenia jest bardzo prosty, szybki, efektywny i bezpieczny. Jest on nastawiony na implementację sprzętową i to właśnie tam



Rysunek 2: Porównanie wydajności realizacji sprzętowych algorytmów dla przepustowości 10 Mbps [9]

osiąga najlepsze rezultaty. Szyfr ten należy do rodziny lekkich szyfrów strumieniowych, których przyszłość jest jeszcze nieznana, natomiast można przewidywać, że w przyszłości mogą być z powodzeniem stosowane w szyfrowaniu komunikacji między urządzeniami IoT. Skutkiem braku szerokiego zastosowania szyfru, jest niepewność odnośnie jego bezpieczeństwa. W rozważaniach teoretycznych odporność Trivium na ataki jest bardzo duża, natomiast do zbudowania zaufania brakuje bardziej złożonych prób złamania jego zabezpieczeń.

Bibliografia

[1] *A survey on lightweight ciphers for IoT devices*. URL: `https://www.researchgate.net/publication/326072910_A_survey_on_lightweight_ciphers_for_IoT_devices`. (Data dostępu: 17.11.2021).

[2] *Asymetric vs Symetric encryption*. URL: `https://www.thesslstore.com/blog/asymmetric-vs-symmetric-encryption/`. (Data dostępu: 15.11.2021).

[3] *Block Cipher vs Stream Cipher*. URL: https://www.thesslstore.com/blog/block-cipher-vs-stream-cipher/. (Data dostępu: 15.11.2021).

[4] Christophe De Canniere i Bart Preneel. "TRIVIUM Specifications". W: *eSTREAM, ECRYPT Stream Cipher Project* 2006 ().

[5] Bart Preneel Cristophe De Canniere. "A Stream Construction Inspired by Block Cipher Design Principles". W: (2006).

[6] *Cryptology ePrint Archive: Report 2021/1523*. URL: https://eprint.iacr.org/2021/1523. (Data dostępu: 17.11.2021).

[7] Christophe De Cannière i Bart Preneel. "Trivium". W: *New Stream Cipher Designs: The eSTREAM Finalists*. Red. Matthew Robshaw i Olivier Billet. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, s. 244–266. ISBN: 978-3-540-68351-3. DOI: 10.1007/978-3-540-68351-3_18. URL: https://doi.org/10.1007/978-3-540-68351-3_18.

[8] Simon Fischer i Willi Meier. "Algebraic Immunity of S-Boxes and Augmented Functions". W: *Fast Software Encryption*. Red. Alex Biryukov. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, s. 366–381. ISBN: 978-3-540-74619-5.

[9] T. Good i M. Benaissa. "Hardware Results for Selected Stream Cipher Candidates". W: *of Stream Ciphers 2007 (SASC 2007), Workshop Record*. 2007, s. 191–204.

[10] Christof Paar i Jan Pelzl. *Understanding Cryptography: A Textbook for Students and Practitioners*. eng. Berlin, Heidelberg: Springer Berlin Heidelberg.

[11] *Trivium cipher - Wikipedia*. URL: https://en.wikipedia.org/wiki/Trivium_(cipher). (Data dostępu: 15.11.2021).

[12] Michael Vielhaber. "Breaking ONE. FIVIUM by AIDA an Algebraic IV Differential Attack." W: *IACR Cryptol. ePrint Arch.* 2007 (2007), s. 413.