Błyskawiczna kryptoanaliza z samym szyfrogramem komunikacji szyfrowanej w systemie GSM

Elad Barkan Eli Biham Nathan Keller

Streszczenie. Zaprezentowano bardzo praktyczny atak z samym szyfrogramem na komunikację szyfrowaną w systemie GSM i różne aktywne ataki na protokoły GSM. Atak te mogą nawet włamywać się do sieci GSM, która używa szyfrów „nie do złamania". Opisano atak z samym szyfrogramem na A5/2, który wymaga tylko kilkunastu milisekund szyfrowanej off-the-air rozmowy i znajduje poprawny klucz w mniej niż sekundę na PC. Rozszerzono ten atak (do bardziej złożonego) na A5/1. Opisano nowe ataki na protokoły sieciowe, które wykorzystują A5/1, A5/3, czy nawet GPRS. Ataki te opierają się na błędach w bezpieczeństwie protokołów GSM i działają zawsze, gdy tylko telefon komórkowy obsługuje A5/2. Podkreślono, że są to ataki na protokoły i mogą być stosowane zawsze gdy telefon komórkowy używa słabego szyfru, zadziałają także z zastosowaniem kryptoanalizy A5/1. W przeciwieństwie do wcześniejszych ataków na GSM, które wymagały nierealnych ilości informacji, takich jak długi znany strumień tekstu jawnego, te ataki są bardzo praktyczne i nie wymagają żadnej wiedzy o zawartości rozmowy. Ataki te pozwalają atakującym na przechwycenie rozmowy i rozszyfrowanie jej w czasie rzeczywistym lub kiedykolwiek później. Pokazano także ataki aktywne takie jak: przechwytywanie rozmowy, zmianę danych wiadomości i kradzież wywołania (rozmowa na koszt innego użytkownika).

1. Wstęp

GSM jest najpopularniejszą używaną technologią komórkową. W grudniu 2002 ponad 787,5 milionów użytkowników GSM w ponad 191 krajach stanowiło około 71% całego cyfrowego bezprzewodowego rynku. GPRS (General Packet Radio Service) jest nową usługą w sieciach GSM, która oferuje usługi stale dostępne ('always-on'), o dużych możliwościach i przepustowościach, oparte o zawartość Internetu i oparte o pakiety danych. Umożliwia takie usługi jak przeglądanie Internetu w kolorze, pocztę elektroniczną w ruchu (on the move), rozbudowaną komunikację wizualną, wiadomości multimedialne i usługi oparte na lokalizacji.

GSM dostarcza mechanizmów bezpieczeństwa. Operatorzy sieciowi i ich klienci polegają na tych mechanizmach w kwestii prywatności swoich rozmów i integralności sieci komórkowej. Mechanizmy bezpieczeństwa chronią sieć poprzez uwierzytelnianie klientów w sieci, zapewniają poufność klientów poprzez szyfrowanie rozmów transmitowanych w eterze.

W GSM używane są trzy główne typy algorytmów kryptograficznych: A5 jest szyfrem strumieniowym stosowanym do szyfrowania, A3 jest algorytmem uwierzytelniania i A8 jest algorytmem uzgadniania klucza. Projekt A3 i A8 nie jest wyspecyfikowany w specyfikacji GSM, tylko zewnętrzny interfejs tych algorytmów jest podany. Konkretny projekt algorytmu może być wybierany przez operatorów niezależnie. Jednakże, wielu operatorów używało przykładu zwanego COMP128 podanego w Memorandum of Understending (MoU) GSMu. Choć nigdy nie został on oficjalnie opublikowany, jego opis znaleźli Briceno, Goldberg i Wagner [6]. Przeprowadzili oni kryptoanalizę COMP128 [7] pozwalającą znaleźć współdzielony klucz główny telefonu komórkowego i sieci, co umożliwia klonowanie. Opis A5 jest częścią specyfikacji GSM, ale nigdy nie był opublikowany. Obecnie stosowane są dwie wersje A5: A5/1 jest ograniczoną eksportowo „silną" wersją, a A5/2 jest „słabą" wersją bez ograniczeń eksportowych. Dokładny opis zarówno A5/1 jak i A5/2 odtworzył Briceno za pomocą inżynierii wstecznej z rzeczywistego telefonu w 1999 i skonfrontował ze znanymi wektorami testowymi. Nową dodatkową wersją, która podlega standaryzacji a nie jest jeszcze używana w sieciach GSM jest A5/3. Wybrano ją nie dawno i oparto o szyfr blokowy KASUMI. Zauważmy, że podobna konstrukcja oparta na KASUMI jest używana także w sieciach trzeciej generacji (3GPP) [1], do których nie nawiązano w tym artykule.

A5/1 został wstępnie poddany kryptoanalizie przez Golicia [14], później przez Biryukova, Shamira i Wagnera [4], Bihama i Dunkelmana [2] i ostatnio przez Ekdahla i Johanssona [11].

Natychmiast po odtworzeniu za pomocą wstecznej inżynierii A5/2 został on poddany kryptoanalizie przez Goldberga, Wagnera i Greena [13]. Ich atak jest atakiem ze znanym tekstem jawnym i wymaga znajomości różnicy między dwoma ramkami GSMu, odległymi od siebie o dokładnie 2^{11} ramek (około 6 sekund). Średnia złożoność tego ataku wynosi 2^{16} iloczynów skalarnych 114-bitowych wektorów. Późniejsza praca Petrovica i Fustera-Sabatera [17] sugerowała potraktowanie początkowego stanu wewnętrznego szyfru jako zmiennych, zapisanie każdego wyjściowego bitu z A5/2 jako kwadratowej funkcji tych zmiennych i linearyzację wyrażeń kwadratowych. Pokazali oni, że wyjście z A5/2 można przewidzieć z bardzo dużym prawdopodobieństwem po kilkuset znanych bitach wyjściowych. Jednak atak ten nie odkrywa klucza początkowego A5/2, więc nie może go stosować jako elementu składowego bardziej zaawansowanych ataków, jak te pokazane w rozdziale 5. Złożoność czasowa tego ataku jest proporcjonalna do 2^{17} eliminacji Gaussa macierzy o rozmiarach około 400×719. Ten późniejszy atak używa właściwie nadopisanego układu równań kwadratowych

w czasie tych obliczeń, ale nie ma potrzeby rozwiązywania tego układu do przeprowadzenia kryptoanalizy. Układ ten jest stosowany tylko do przewidzenia wyjść w przyszłych ramkach.

Rozwiązywanie nadopisanego układu równań kwadratowych, jako metoda kryptoanalizy przyciągało znaczącą uwagę w literaturze. Metoda ta zostało początkowo zastosowana przez Kipnisa i Shamira do kryptosystemu klucza publicznego HFE [15], później udoskonalona przez Courtois, Klimova, Patarina i Shamira [9]. Kolejna praca zawierała kryptoanalizę szyfrów blokowych Courtios i Pieprzyka [10]. Metoda ta zastosowana została także do szyfrów strumieniowych, patrz praca Courtois o Toyocrypt'cie [8]. Złożoność większości tych metod jest trudna do oszacowania.

W tym artykule pokazano jak zastosować atak z samym szyfrogramem do A5/2. Atak ten wymaga kilkunastu milisekund szyfrowanych danych, a jego złożoność czasowa wynosi około 2^{16} iloczynów skalarnych. W przeprowadzonych symulacjach, atak znalazł klucz w mniej niż sekundę na PC. Pokazano, że ten atak na A5/2 może być wykorzystany do przeprowadzenia ataku aktywnego na sieci GSM stosujące A5/1, A5/3 lub sieci GPRS i w ten sposób umożliwia on przeprowadzenie w czasie rzeczywistym aktywnego ataku na sieci GSM bez żadnej wcześniejszej wymaganej wiedzy. Cały atak złożony jest z trzech głównych kroków:

1. Pierwszy krok to bardzo efektywny atak ze znanym tekstem jawnym na A5/2, który znajduje klucz początkowy. Pierwszy atak ma charakter algebraiczny. Wykorzystuje niski stopień algebraiczny wyjściowej funkcji A5/2. Reprezentujemy wyjście z A5/2 jako funkcję kwadratową wielu zmiennych stanu początkowego rejestrów. Następnie konstruujemy nadopisany układ równań kwadratowych, który wyraża proces generowania strumienia klucza i rozwiązujemy ten układ.
2. Drugi krok to udoskonalenie ataku ze znanym tekstem jawnym do ataku z samym szyfrogramem na A5/2. Obserwujemy, że GSM wykorzystuje kody korekcji błędów przed szyfrowaniem. Pokazujemy jak wykorzystać tę obserwację do przekształcenia pierwszego ataku na A5/2 na atak z samym szyfrogramem.
3. Trzeci krok to przeniesienie ataku z A5/2 na aktywny atak na A5/1, A5/3 lub sieci GSM z GPRS-em. Zgodnie z projektem interfejsu modułów bezpieczeństwa GSM, klucz, który używany jest w A5/2 jest taki sam jak w A5/1, A5/3 i GPRS-ie. Pokazano jak przeprowadzić aktywny atak na dowolną sieć GSM.

Następnie pokażemy jak przeprowadzić bierny atak z samym szyfrogramem na sieci używające A5/1. Jest to po prostu atak wykorzystujący zależność czas/pamięć/dane.

Parametry tego ataku można wybrać na wiele sposobów, cztery z nich podano w tabeli 1. Ten atak na A5/1 może być prosto przeniesiony na aktywne ataki na protokoły GSM, ale złożoność jest większa niż w przypadku A5/2.

Artykuł jest zorganizowany następująco: w rozdziale 2 opisujemy A5/2 i sposób w jaki jest on stosowany, podajemy także pewne aspekty bezpieczeństwa GSM. Nowy atak prezentujemy w rozdziale 3. W rozdziale 4 udoskonalamy ten atak do ataku z samym szyfrogramem. W rozdziale 5 pokazujemy jak atak ten przenieść na atak aktywny na dowolną sieć GSM. Następnie w rozdziale 6 opisujmy bierny atak z samym szyfrogramem na A5/1. O wpływie tych ataków na wiele scenariuszów ataku dyskutujemy w rozdziale 7. W rozdziale 8 dokonujemy podsumowania całego artykułu.

2. Opis A5/2 i bezpieczeństwo GSM

W tej części opiszemy strukturę wewnętrzną A5/2 i sposób jego stosowania. A5/2 składa się z 4 LFSR maksymalnej długości: R1, R2, R3 i R4. Rejestry te mają długości odpowiednio: 19, 22, 23 i 17 bitów. Każdy rejestr posiada odczepy i sprzężenie zwrotne. Ich wielomiany nieredukowalne to: $x^{19} \oplus x^5 \oplus x^2 \oplus x \oplus 1$, $x^{22} \oplus x \oplus 1$, $x^{23} \oplus x^{15} \oplus x^2 \oplus x \oplus 1$, $x^{17} \oplus x^5 \oplus 1$. Przyjmiemy reprezentację w której bity w rejestrze dane są w kolejności odwrotnej, tj. x^i odpowiada odczepowi o indeksie *len-i-1*, gdzie *len* oznacza długość rejestru. Na przykład, gdy R4 jest taktowany, to obliczany jest XOR R4[17-0-1=16] i R4[17-5-1=11]. Następnie rejestr jest przesuwany o jeden w prawo, a wynik XOR umieszczany jest w R4[0].

W każdym kroku A5/2 R1, R2 i R3 taktowane są zgodnie z mechanizmem taktowania opisanym dalej. Następnie taktowany jest R4. Po tym taktowaniu jeden bit wyjściowy jest gotowy. Bit ten jest wyznaczany z nieliniowej funkcji stanów wewnętrznych rejestrów R1, R2 i R3.

Po inicjalizacji 99 bitów wyjścia jest odrzucanych, a następne 228 bitów stanowi wyjście używane jako strumień klucza.

Oznaczmy: i-ty bit 64-bitowego klucza sesji K_c przez $K_c[i]$, i-ty bit rejestru j przez Rj[i], a i-ty bit 22-bitowego publicznie znanego numeru ramki przez f[i].

Inicjalizacja stanu wewnętrznego kluczem K_c i numerem ramki wykonywana jest następująco:

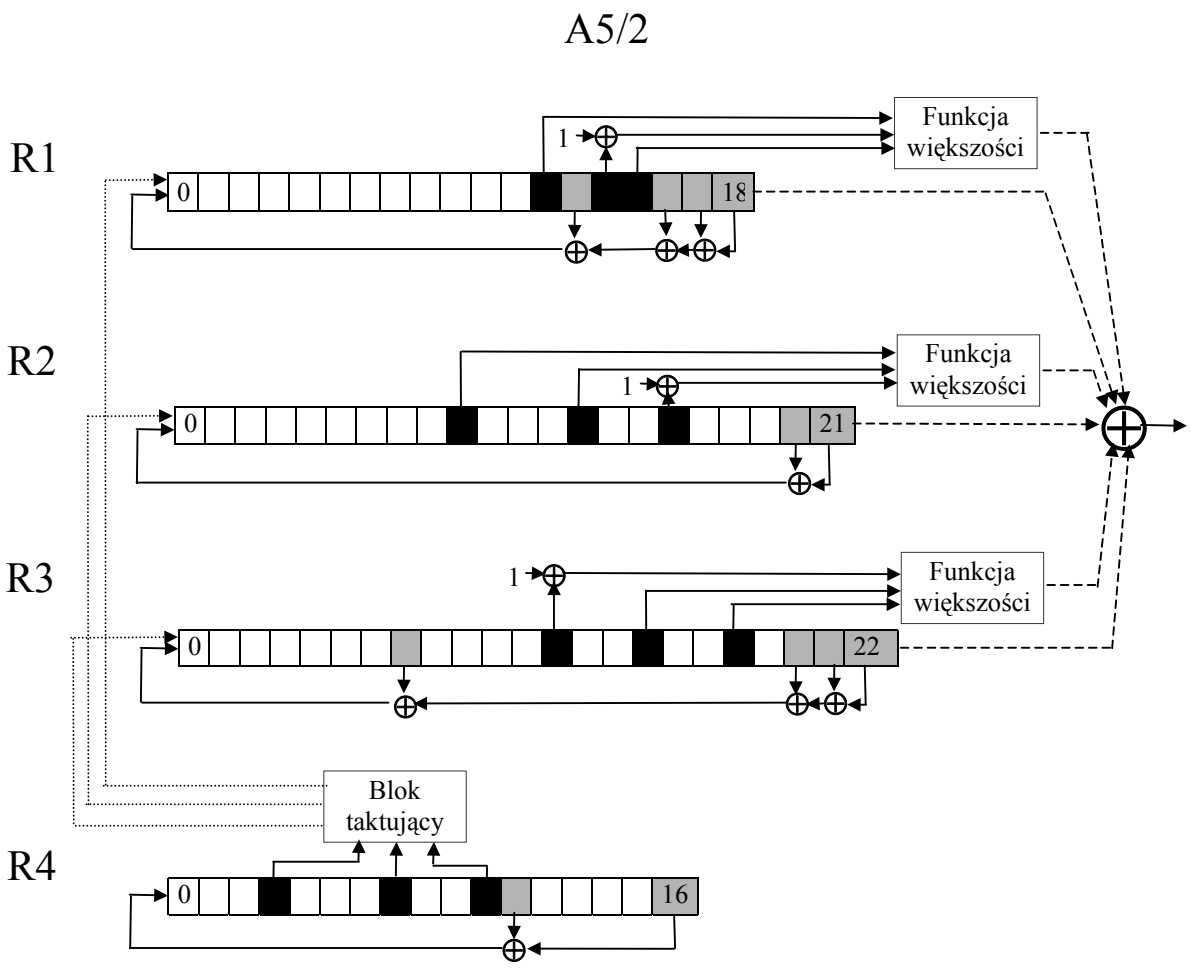
- Wszystkie rejestry zerujemy (R1 = R2 = R3 = R4 = 0).
- Dla i = 0 do 63 wykonujemy:
    1. Taktujemy wszystkie 4 LFSRy.
    2. $R1[0] \leftarrow R1[0] \oplus K_c[i]$.

3. R2[0] ← R2[0] ⊕K_c[i].
4. R3[0] ← R4[0] ⊕K_c[i].
5. R4[0] ← R4[0] ⊕K_c[i].

- Dla i = 0 do 21 wykonujemy:
  1. Taktujemy wszystkie 4 LFSRy.
  2. R1[0] ← R1[0] ⊕f[i].
  3. R2[0] ← R2[0] ⊕f[i].
  4. R3[0] ← R4[0] ⊕f[i].
  5. R4[0] ← R4[0] ⊕f[i].

Generowanie strumienia klucza przebiega następująco:

1. Inicjujemy stan wewnętrzny kluczem K_c i numerem ramki.
2. Wymuszamy jedynkę na pozycjach R1[15], R2[16], R3[8] i R4[10].
3. Uruchamiamy A5/2 na 99 taktów i odrzucamy wyjście.
4. Uruchamiamy A5/2 na 228 taktów generując strumień klucza.



Po wykonaniu pierwszego taktowania pierwszy wyjściowy bit jest gotowy na wyjściu z A5/2. Na rysunku pokazano wewnętrzną strukturę A5/2. Mechanizm taktowania działa następująco: R4 kontroluje taktowanie rejestrów R1, R2 i R3. Bity R4[3], R4[7] i R4[10] podawane są na blok taktujący. Blok ten wylicza wartość funkcji większości tych bitów. R1 jest taktowany wtedy i tylko wtedy, gdy bit R4[10] zgadza się z wyjściem z funkcji większości. R2 jest taktowany wtedy i tylko wtedy, gdy bit R4[3] zgadza się z tą większością. R3 jest taktowany wtedy i tylko wtedy, gdy bit R4[7] zgadza się z tą większością. Po tych taktowaniach, taktowany jest R4.

Po taktowaniu bit wyjściowy jest gotowy. Jest on wyznaczany następująco: w każdym rejestrze obliczana jest większość z dwóch bitów i negacji trzeciego, wynik tej większości i prawy skrajny bit każdego rejestru xorowane są tworząc wyjście (patrz rysunek). Zauważmy, że funkcja większości jest kwadratowa: $maj(a,b,c) = a \cdot b \oplus b \cdot c \oplus c \cdot a$.

A5/2 zbudowany jest na nieco podobnej zasadzie co A5/1. Funkcje sprzężenia zwrotnego R1, R2 i R3 są takie same jak w A5/1. Proces inicjalizacji też jest podobny. Różnicą jest inicjalizowanie także rejestru R4 i to, że jeden bit każdego rejestru jest zmieniany na jedynkę po inicjalizacji. A5/2 odrzuca 99 bitów, a A5/1 100 bitów wyjścia. Mechanizm taktowania jest ten sam, z tym, że bity do tego mechanizmu pochodzą z rejestru R4 w A5/2, a w A5/1 z rejestrów R1, R2 i R3. Projektanci chcieli wykorzystać podobne bloki, aby zaoszczędzić sprzęt w telefonach komórkowych [16].

Algorytm ten daje 228 bitów strumienia klucza. Pierwszy blok 114 bitów używany jest jako strumień klucza do zaszyfrowania połączenia z sieci do użytkownika, a drugi blok do zaszyfrowania połączenia od użytkownika do sieci. Szyfrowanie wykonywane jest poprzez proste xorowanie strumienia klucza z wiadomością.

Chociaż A5 jest szyfrem strumieniowym to jest on używany do szyfrowania 114-bitowych bloków zwanych *ramkami*. Ramki numerowane są kolejno (modulo 2^{22}) *numerem ramki TDMA*. Numer ramki f, który używany jest w czasie inicjalizacji każdej ramki A5 jest właściwie ustaloną permutacją bitową numeru ramki TDMA. W dalszej części tego artykułu istnienie tej permutacji zostanie pominięte, gdyż nie ma ona wpływu na naszą analizę.

2.1 Bezpieczeństwo GSM: A3/A8 i GPRS

W tym rozdziale podano dokładniejszy opis stosowania i specyfikacji A3 i A8. Można go także znaleźć w [12].

A3 umożliwia uwierzytelnienie telefonu komórkowego wobec sieci, a A8 stosowany jest do uzgodnienia klucza sesji. Bezpieczeństwo tych algorytmów opiera się na tajnym

kluczu K_i właściwym dla każdego użytkownika (user-specific) i jest on wspólny dla telefonu i sieci. Specyfikacja GSM nie określa długości klucza K_i, więc zostaje to w gestii operatora, ale jest to zwykle klucz 128 bitowy. Uwierzytelnienie użytkowników w sieci odbywa się poprzez algorytm uwierzytelnienia A3 następująco: sieć generuje wyzwanie (challenge) dla użytkownika, którym jest 128 bitowa losowa wartość *RAND*. Użytkownik wylicza 32-bitową odpowiedź $SRES = A3(K_i, RAND)$ i wysyła ją do sieci, która weryfikuje jej poprawność.

Klucz sesji K_c otrzymywany jest za pomocą A8 następująco: $K_c = A8(K_i, RAND)$. Zauważmy, że A8 i A3 zawsze wywoływane są razem z tymi samymi parametrami. W większości implementacji są one realizowane jako jeden algorytm z dwoma wyjściami SRES i K_c. Stąd zwykle oznaczane są jako jeden algorytm A3/A8.

Bezpieczeństwo w GPRS oparte jest na tych samych mechanizmach co GSM. Jednakże GPRS używa innego klucza szyfrującego do szyfrowania swojej komunikacji. Do uwierzytelnienia użytkownika używany jest ten sam algorytm A3/A8 z tym samym kluczem K_i, ale z innym RAND. Uzyskany K_c jest używany do szyfrowania ruchu w GPRS. Oznaczymy go przez GPRS-K_c, dla odróżnienia od K_c stosowanego w szyfrowaniu głosu w GSM. Podobnie oznaczymy SRES i RAND przez GPRS-SRES i GPRS-RAND dla odróżnienia ich odpowiedników z szyfrowania głosu w GSM. Szyfr z GPRS oznaczymy jako GPRS-A5 lub Algorytm szyfrujący GPRS (GPRS Encryption Algorithm - GEA). Obecnie mamy trzy wersje tego algorytmu: GEA1, GEA2 i GEA3 (właściwie A5/3).

3. Atak ze znanym tekstem jawnym na A5/2

W tej części zaprezentujemy nowy atak ze znanym tekstem jawnym (ze znanym strumieniem klucza) na A5/2. Dokładniej, na podstawie znanego strumienia klucza, podzielonego na ramki, z odpowiednimi numerami ramek, atak ten znajduje klucz sesji.

Goldberg, Wagner i Green pokazali pierwszy atak [13] na A5/2. Złożoność czasowa tego ataku jest bardzo mała. Jednak wymaga on znajomości xora tekstów jawnych w dwóch ramkach odległych od siebie o 2^{11} ramek. Ich atak pokazuje, że szyfr ten jest bardzo słaby, jednakże może się on okazać trudny do realizacji w praktyce. Problemem jest znajomość dokładnego xora tekstów jawnych w dwóch ramkach odległych od siebie o 6 sekund. Inną kwestią jest czas, jaki upłynie od początku ataku do jego zakończenia. Ich atak trwa co najmniej 6 sekund, ponieważ tyle trwa gromadzenia danych. Może się wydawać, że nasz atak wymaga więcej danych, jednak działa już z kilkoma milisekundami danych. Ulepszymy ten atak w rozdziale 4 do ataku z samym szyfrogramem, który wymagał będzie tylko kilkudziesięciu milisekund zaszyfrowanych nieznanych danych. Dlatego też nasz atak jest

bardzo łatwy do realizacji w praktyce. Przeprowadzone symulacje tego ataku na PC potwierdzają to. Symulacja odzyskuje klucz w mniej niż sekundę. Złożoność czasowa i pamięciowa tego ataku są podobne do ataku Goldberga, Wagnera i Greena. Atak ze znanym tekstem jawnym Petrovica i Fustera-Sabatera [17] ma podobne wymagania na dane, jednak nie znajduje klucza inicjującego.

Znając stany początkowe R1, R2, R3 i R4 i początkowy numer ramki można odzyskać klucz sesji za pomocą prostych operacji algebraicznych. Wynika to z tego, że proces inicjalizacji jest liniowy w stosunku do klucza sesji i początkowego numeru ramki. Dlatego też atak skupia się na ujawnieniu początkowych stanów rejestrów.

Niech k_f, k_{f+1}, k_{f+2}, ... będą wyjściem z A5/2 podzielonym na ramki. Zauważmy, że każde k_j jest wyjściowym strumieniem klucza dla całej ramki, tzn. każde k_j ma długość 114 bitów. Niech f, f + 1, f + 2, ... będą numerami ramek odpowiadającymi tym ramkom. Oznaczymy i-ty bit strumienia klucza w ramce j-ej przez $k_j[i]$. Wewnętrzny stan początkowy rejestru Ri w ramce j-ej (po inicjalizacji, ale przed 99 taktowaniami) oznaczymy przez Ri_j.

Załóżmy, że znamy stan początkowy $R4_f$ rejestru R4 w pierwszej ramce. Ważną obserwacją jest to, że R4 kontroluje taktowaniem pozostałych rejestrów. Jeśli znamy $R4_f$ to dla każdego wyjściowego bitu znamy dokładną liczbę taktowań rejestrów do wygenerowania tego bitu. Każdy rejestr ma liniowe sprzężenie zwrotne, stąd, wiedząc ile razy rejestr był taktowany możemy każdy bit jego stanu wewnętrznego wyrazić jako liniową kombinację bitów oryginalnego (początkowego) stanu wewnętrznego.

Wyjście z A5/2 jest xorem ostatnich bitów R1, R2, R3 i trzech funkcji większości bitów z R1, R2 i R3 (patrz rysunek). Stąd funkcja wynikowa jest kwadratowa w odniesieniu do zmiennych, którymi są bity stanów początkowych rejestrów. Wykorzystamy tę własność niskiego stopnia algebraicznego wyjścia. Celem następnego paragrafu jest wyrażenie każdego bitu całego wyjścia szyfru (złożonego z wielu ramek) jako kwadratowej funkcji wielu zmiennych stanu początkowego. Następnie zbudujemy nadopisany układ równań kwadratowych, który wyraża proces generowania strumienia klucza i na koniec rozwiążemy go.

Mając dany numer ramki f, istnieje algebraiczny opis (wzór) na każdy bit wyjścia. Przeprowadzamy linearyzację wyrażeń kwadratowych w tych algebraicznych wzorach. Zauważamy, że każda funkcja większości operuje na bitach pojedynczego rejestru. Dlatego też, mamy kwadratowe wyrażenia zawierające pary zmiennych z tego samego rejestru. Biorąc pod uwagę, że jeden bit w każdym rejestrze jest ustawiany na 1, R1 daje 18 liniowych zmiennych i wszystkie swoje (17*18)/2=153 iloczyny. W ten sam sposób R2 daje

21+(21*20)/2=21+210 zmiennych, a R3 22+(22*21)/2=22+231 zmiennych. Stąd mamy 18+153+21+210+22+231 = 655 zmiennych po linearyzacji. Razem za stałą 1 mamy zbiór 656 zmiennych. Oznaczymy ten zbiór 656 zmiennych przez V_f. Z tych zmiennych 18+21+22 = 61 zmiennych tworzy cały stan początkowy rejestrów R1, R2 i R3.

Każdy bit wyjścia, który znamy wprowadza liniowe równanie ze zmiennymi z V_f. Ramka składa się ze 114 bitów. Dlatego też, dostajemy 114 równań dla każdej ramki. Rozwiązanie tego układu równań ujawnia wartości zmiennych z V_f, a między nimi te, które bezpośrednio wyznaczają stan początkowy rejestrów. Jednakże, nie mamy wystarczająco dużo równań na tym etapie, aby efektywnie rozwiązać ten układ.

Główną obserwacją jest, to że mając zmienne z V_f dla ramki f możemy przedstawić bity każdej innej ramki za pomocą wyrażeń liniowych w zmiennych zbioru V_f. Gdy przechodzimy do kolejnej ramki, numer ramki jest zwiększany o 1 i stan początkowy jest reinicjalizowany. Załóżmy, że znamy wartość $R4_f$. Zgodnie z metodą inicjalizacji numer ramki jest wprowadzany poprzez xor bit po bicie do rejestrów (patrz rozdział 2), znamy więc wartość $R4_{f+1}$. Nie znamy więc wartości $R1_f$, $R2_f$ i $R3_f$ ani wartości $R1_{f+1}$, $R2_{f+1}$ i $R3_{f+1}$, ale znamy różnice między $R1_f$, $R2_f$ i $R3_f$, a odpowiednio $R1_{f+1}$, $R2_{f+1}$ i $R3_{f+1}$. Zdefiniujemy zbiór zmiennych, które opisują ich stan i linearyzację tych zmiennych jako V_{f+1}, w taki sam sposób jak to zrobiliśmy tworząc zbiór V_f. Zgodnie z metodą inicjalizacji, dla każdego rejestru Ri znamy różnicę między Ri_f, a Ri_{f+1}. Więc możemy zapisać zmienne ze zbioru V_{f+1} jako liniowe kombinacje zmiennych z V_f. Podkreślmy, że nawet kwadratowe wyrażenia mogą być reprezentowane w ten sposób. Aby to zobaczyć załóżmy, że $a_{f+1} \cdot b_{f+1}$ jest wyrażeniem kwadratowym w V_{f+1}, a $a_f \cdot b_f$ jest wyrażeniem kwadratowym w V_f. Znamy różnicę $d_a = a_f \oplus a_{f+1}$ i $d_b = b_f \oplus b_{f+1}$. Stąd $a_{f+1} \cdot b_{f+1} = (a_f \oplus d_a) \cdot (b_f \oplus d_b) = a_f \cdot b_f \oplus a_f \cdot d_b \oplus b_f \cdot d_a \oplus d_a \cdot d_b$. Ponieważ d_b i d_a są znanymi stałymi równanie to jest liniowe dla zmiennych w V_f. Fakt ten pozwala nam na wykorzystanie bitów wyjściowych drugiej ramki do otrzymania dodatkowych równań liniowych dla zmiennych z V_f. W podobny sposób zapiszemy zmienne z każdego zbioru V_i jako liniowe kombinacje zmiennych z V_f. W sumie dostaniemy układ równań postaci: $S \cdot V_f = k$, gdzie S jest macierzą układu a k jest konkatenacją k_f, k_{f+1}, itd.

Jasnym jest, że jak tylko otrzymamy 656 liniowo niezależnych równań układ można będzie łatwo rozwiązać stosując eliminację Gaussa. Jednakże, w praktyce bardzo trudno jest zebrać 656 liniowo niezależnych równań. Wynika to z częstych reinicjalizacji i małego stopnia funkcji większości. Zauważmy, że właściwie nie potrzebujemy rozwiązań dla wszystkich zmiennych, tzn. wystarczy znaleźć wartości zmiennych liniowych tego układu,

ponieważ pozostałe zmienne zdefiniowane są jako ich iloczyny. Przeprowadziliśmy testy i stwierdziliśmy, że po zgromadzeniu kolejno 450 równań, można znaleźć oryginalne zmienne liniowe z V_f stosując eliminację Gaussa. W przypadku gdy ilość danych dostępnych dla atakującego jest mała można zastosować dodatkowe metody do zmniejszenia liczby wymaganych równań. Na przykład gdy tylko wyznaczymy wartość zmiennej x_i, wszystkie zmienne kwadratowe postaci $x_i \cdot x_j$ można uprościć do 0 lub x_j odpowiednio gdy $x_i=0$ lub $x_i=1$. Można zastosować także algorytm XL w przypadku szczupłości dostępnych danych.

Atak możemy podsumować następująco: próbujemy wszystkie 2^{16} możliwe wartości dla $R4_f$, dla każdej takiej wartości rozwiązujemy zlinearyzowany układ równań zbudowany na podstawie bitów (ramek) wyjściowych. Rozwiązanie każdego układu sugeruje nam stan początkowy rejestrów R1, R2 i R3. Razem z R4 mamy sugestię całego stanu początkowego. Większość z $2^{16} - 1$ złych stanów można zidentyfikować w związku z wykryciem sprzeczności w trakcie eliminacji Gaussa. Jeśli dostaniemy więcej niż jeden niesprzeczny układ równań to poprawny stan początkowy wykryjemy stosując do wszystkich znalezionych próbne szyfrowanie.

Złożoność tego ataku jest następująca: Mamy 2^{16} możliwych wartości $R4_f$. Powinniśmy pomnożyć tę wartość przez czas potrzebny do rozwiązania liniowego, binarnego układu równań z 656 zmiennymi, który wynosi około $656^3 \approx 2^{28}$ operacji xor. Więc całkowita złożoność wynosi około 2^{44} bitowych operacji xor. Na 32-bitowym komputerze złożoność ta to 2^{39} operacji xor na rejestrach.

Implementacja tego algorytmu na PC 800MHz PIII z Linuxem znajduje stan początkowy w ciągu około 40 minut i wymaga relatywnie małej pamięci, przechowanie zlinearyzowanego układu w pamięci wymaga 656^2 bitów ≈ 54 KB.

3.1 Optymalizacja ataku ze znanym tekstem jawnym na A5/2.

W zoptymalizowanej implementacji, którą przedstawimy szczegółowo w pełnej wersji tego artykułu średnia złożoność czasowa może być zredukowana do około 2^{28} operacji pobitowych XOR-ów (mniej niż 1 sekunda na naszym PC). Złożoność pamięciowa rośnie do około $2^{27,8}$ bajtów (mniej niż 250 MB). Optymalizacja ta wymaga kroku obliczeń wstępnych, którego złożoność czasowa wynosi około 2^{46} operacji XOR (około 160 minut na naszym PC). Złożoność danych jest nieco wyższa, ale wciąż w zakresie kilkunastu milisekund danych. Optymalizacja opiera się na spostrzeżeniu, że dla każdej kandydującej wartości $R4_f$ układ równań zawiera liniowo zależne wiersze. W etapie obliczeń wstępnych rozważamy wszystkie

możliwe wartości $R4_f$, dla każdej takiej wartości wyznaczamy układ równań. Sprawdzamy, które wiersze są liniowo zależne poprzez przeprowadzenie eliminacja Gaussa. W drugiej fazie ataku (w czasie rzeczywistym) filtrujemy złe wartości $R4_f$ poprzez sprawdzenie, czy liniowe zależności, które znaleźliśmy w fazie obliczeń wstępnych zachodzą dla bitów strumienia klucza. Ten rodzaj filtracji wymaga obliczenia średnio dwóch iloczynów skalarnych dla każdej złej wartości $R4_f$. Gdy tylko wyznaczymy możliwe poprawne wartości $R4_f$, to poprawny klucz znajdziemy stosując metodę opisaną w rozdziale 3.

Zauważmy, że gdy stosujemy ten zoptymalizowany atak wymagany jest pewien kompromis. Ponieważ wymagane są cztery znane ramki tekstu jawnego, musimy znać różnice XOR: $f \oplus (f+1)$, $f \oplus (f+2)$, $f \oplus (f+3)$, wcześniej, zanim poznamy wprost wartość f. Te różnice XOR wymagane są aby wyrazić bity strumienia klucza ramek jako wyrażeń liniowych nad zbiorem V_f i wyznaczenia układów równań. Elementem problematycznym jest operacja dodawania np. f+1 może dawać przeniesienie, które może propagować się poprzez f, więc nie pozwalające na wyznaczenie różnicy XOR wcześniej. Dlatego też wymagamy, aby w f specyficzny bit ustawić na 0. Wymaganie to chroni przed propagacją przeniesienia za ten specyficzny bit. Wymagamy, aby dwa ostatnie bity w f miały ustaloną wartość i wykonujemy obliczenia wstępne dla każdej z czterech kombinacji dwóch ostatnich bitów f. Wymaganie to jest powodem nieco zwiększonej złożoności danych i sprawia także, że złożoność pamięciowa i złożoność czasu obliczeń wstępnych należy pomnożyć przez 4.

4. Błyskawiczny atak z samym szyfrogramem na A5/2

W tym rozdziale zaadaptujemy atak z rozdziału 3 na A5/2 do ataku z samym szyfrogramem. Zauważmy, że przed szyfrowaniem w GSM stosowane są kody korekcji błędów. Stąd tekst jawny ma strukturę wysoce nadmiarową.

Istnieje wiele rodzajów metod korekcji błędów stosowanych w GSM i różnych schematów korekcji błędów używanych w różnych kanałach danych. Skupimy się na kanałach sterowania, a szczególnie na kodach korekcji błędów kanału SACCH (Slow Asocciated Control Channel). Zauważmy, że ten kod korekcji błędów jest tym stosowanym podczas inicjalizacji rozmowy. Wystarczy więc skupić się na tym kodzie. Wykorzystując ten kod przeprowadzimy atak, który znajduje klucz z samym szyfrogramem. Należy tutaj podkreślić, że idee tego ataku można z powodzeniem zastosować do innych kodów korekcji błędów.

W SACCH wiadomość kodowana za pomocą kodu korekcji błędów ma ustaloną długość 184 bitów. Wynikiem jest wiadomość o długości 456 bitów. Te 456 bitów

wiadomości są następnie przeplecione i podzielone na cztery ramki. Te ramki są z kolei szyfrowane i transmitowane.

Operację kodowania i operację przeplotu można zamodelować razem jako jedną macierz o wymiarze 456×184 nad ciałem GF(2), którą oznaczymy przez G. Wiadomość do zakodowania będzie 184 bitowym wektorem binarnym P. Wynikiem operacji kodowania i przeplotu jest M = G·P. Wektor wynikowy M zawiera 4 ramki. W procesie szyfrowania każda ramka jest xorowana z wyjściowym strumieniem klucza z A5/2 dla odpowiedniej ramki.

Ponieważ G jest binarną macierzą o wymiarze 456×184, to mamy 456 – 184 = 272 równania opisujące jądro przekształcenia odwrotnego (rozmiar jądra jest nie większy niż 272 w związku z własnościami macierzy G). Innymi słowy, dla każdego wektora M, M = G·P, mamy 272 liniowo niezależne równania na jego elementy. Niech K_G będzie macierzą opisującą te 272 liniowe równania, tzn. $K_G \cdot M = 0$ dla każdego takiego M.

Wyjściowy ciąg bitów z A5/2 dla 4 ramek oznaczymy przez $k = k_j \,||\, k_{j+1} \,||\, k_{j+2} \,||\, k_{j+3}$, gdzie || oznacza konkatenację. Szyfrogram C jest wyznaczany jako C = M⊕k. Stosujemy te same 272 równania na C, czyli:

$$K_G \cdot C = K_G \cdot (M \oplus k) = K_G \cdot M \oplus K_G \cdot k = 0 \oplus K_G \cdot k = K_G \cdot k.$$

Ponieważ szyfrogram C jest znany, to faktycznie dostaniemy liniowe równania nad elementami k. Zauważmy, że te otrzymane równania nie zależą od P, zależą tylko od k. Za każdy bit k podstawimy jego wyrażanie liniowe nad V_f (patrz rozdział 3) i stąd otrzymamy równania ze zmiennymi z V_f. Każdy 456-bitowy blok kodowy dostarcza 272 równań. Reszta szczegółów tego ataku i jego złożoność czasowa jest podobna do przypadku z poprzedniego rozdziału, z tym, że w tym ataku znamy liniowe kombinacje strumienia klucza i stąd odpowiadające im równania są odpowiednimi kombinacjami liniowymi tych równań: niech $S \cdot V_f = k$ będzie układem równań z rozdziału 3, gdzie S jest macierzą układu. W ataku z samym szyfrogramem mnożymy ten układ przez K_G następująco: $(K_G \cdot S) \cdot V_f = (K_G \cdot k)$. K_G jest ustaloną znaną macierzą, która zależy tylko od (kodowo-przeplotowej) macierzy G, S jest macierzą układu, różną dla każdej wartości $R4_f$ (i dla różnych różnic XOR f⊕(f+1), f⊕(f+2), f⊕(f+3), itd.). Stąd w ataku zoptymalizowanym wszystkie możliwe macierze $K_G \cdot S$ wyliczane są w czasie obliczeń wstępnych, a dla każdej takiej macierzy znajdujemy liniowe zależności miedzy wierszami za pomocą eliminacji Gaussa. W fazie ataku w czasie rzeczywistym filtrujemy złe wartości $R4_f$ poprzez sprawdzenie czy liniowe zależności, które znaleźliśmy w fazie obliczeń wstępnych zachodzą dla bitów $K_G \cdot k$.

Zauważmy, że podczas gdy cztery ramki danych wystarczają do rozpoczęcia ataku z rozdziału 3, to w ataku z samym szyfrogramem potrzebujemy 8 ramek, ponieważ dla każdej szyfrowanej ramki w porównaniu do ataku ze znanym tekstem jawnym dostajemy tylko około połowę informacji. Złożoność czasowa tego ataku jest taka sama jak dla ataku z rozdziału 3.1.

Przeanalizujemy teraz złożoność czasową i pamięciową ataku z samym szyfrogramem z zastosowaniem optymalizacji jak w rozdziale 3.1. W rozdziale 3.1 ograniczyliśmy wartości dwóch najmniej znaczących bitów numeru ramki f, stąd potrzebowaliśmy czterech ramek danych. Atak w tym rozdziale wymaga 8 ramek danych, dlatego ograniczymy trzy najmniej znaczące bity numeru ramki f. Ograniczenie to podwaja złożoność pamięciową w porównaniu do ataku z rozdziału 3.1 i podwaja także złożoność czasu obliczeń wstępnych.

Podsumowując, złożoność ataku z samym szyfrogramem (używając zoptymalizowanej implementacji) jest następująca: średnia złożoność czasowa ataku z samym szyfrogramem wynosi około 2^{16} iloczynów skalarnych, złożoność pamięciowa to około $2^{28,8}$ bajtów (mniej niż 500 MB), a złożoność czasowa obliczeń wstępnych to około 2^{47} pobitowych xorów. Nasza implementacja na PC (korzystająca z zalety 32-bitowego xora) znajduje klucz K_c w mniej niż sekundę i wymaga około 320 minut (mniej niż 5,5 godziny) na jednokrotne wykonanie obliczeń wstępnych.

Skutecznie poprawiliśmy także atak Goldberga, Wagnera i Greena oraz atak Petrovica i Fustera-Sabatera do ataku z samym szyfrogramem stosując nasze metody. Szczegóły tych ataków pojawią się w pełnej wersji tego artykułu.

5. Przełożenie ataków na wszystkie sieci GSM

Atak pokazany w rozdziale 4 zakłada, że algorytmem szyfrującym jest A5/2. Stosując ten atak łatwo jest znaleźć klucz K_c w czasie rzeczywistym na podstawie kilkunastu milisekund szyfrogramu. Powstaje pytanie co się stanie jeśli algorytmem szyfrującym będzie nie A5/2, ale A5/1 lub też nowo wybrany A5/3. Odpowiedź jest zaskakująca: korzystając z słabości protokołów prawie taki sam atak ma tutaj zastosowanie. Co więcej, warianty tego ataku działają na sieci GPRS. Wszystko czego potrzeba do powodzenia takiego nowego ataku to telefon komórkowy obsługujący A5/2 do rozmów głosowych. Zdecydowana większość telefonów obsługuje szyfrowanie A5/2 (aby umożliwić szyfrowanie podczas roamingu w sieciach używających tylko A5/2).

Następujące trzy ataki odzyskują klucz szyfrujący, którego używa sieć podczas zastosowania A5/1 lub A5/3. W pierwszym ataku klucz jest odkrywany poprzez atak typu man-in-the-middle na użytkownika-ofiarę. W ataku tym atakujący gra dwie role. Odgrywa

sieć dla użytkownika i użytkownika dla sieci. W drugim ataku, atakujący potrzebuje zmiany bitów (negacji bitów) w konwersacji telefonu z siecią (można go przeprowadzić także jako atak typu man-in-the-middle). W trzecim atakujący odgrywa rolę sieci w krótkiej sesji radiowej z telefonem. Zauważmy, że te rodzaje ataków są stosunkowo bardzo łatwe do przeprowadzenia w systemie telefonii komórkowej.

Atak typu man-in-the-middle przeprowadzamy następująco: gdy przeprowadzane jest uwierzytelnienie (podczas inicjalizacji rozmowy) sieć wysyła żądanie uwierzytelnienia do atakującego, a atakujący przesyła je do ofiary. Ofiara oblicza SRES i zwraca ją do atakującego, który przesyła ją z powrotem do sieci. Teraz atakujący jest „uwierzytelniony" w sieci. Następnie sieć prosi użytkownika o uruchomienie szyfrowania A5/1 lub A5/3. W naszym ataku ponieważ atakujący odgrywa rolę użytkownika, to faktycznie sieć prosi atakującego o rozpoczęcie szyfrowanie za pomocą A5/1 lub A5/3. Atakujący nie ma klucza, na razie, dlatego też nie jest w stanie rozpocząć szyfrowania. Atakujący potrzebuje klucza zanim zostanie poproszony o jego użycie. Aby to osiągnąć atakujący prosi ofiarę o szyfrowanie za pomocą A5/2 zaraz po tym jak ofiara zwróciła SRES i przed zwrotem tej informacji uwierzytelniającej do sieci. To wezwanie brzmi dla ofiary jak uprawnione żądanie, ponieważ ofiara uważa atakującego za sieć. Teraz atakujący wykorzystuje kryptoanalizę A5/2 do odtworzenia klucza szyfrującego algorytmu A5/2 stosowanego przez ofiarę. Wtedy też atakujący wysyła informację uwierzytelniającą do sieci. Klucz zależy tylko od RAND co oznacza, że klucz uzyskany w ataku na A5/2 jest tym samym kluczem co klucz stosowany gdy używany jest A5/1, a nawet gdy używany jest 64-bitowy A5/3! Teraz atakujący może szyfrować i deszyfrować zarówno algorytmem A5/1 jak i A5/3 stosując ten klucz.

Niektórzy czytelnicy mogą podejrzewać, że sieć może rozpoznać ten atak poprzez zauważenie małego opóźnienia w czasie potrzebnym do ukończenia procedury uwierzytelnienia. Jednakże, standard GSM czeka 12 sekund, aż telefon komórkowy zakończy obliczenia swojego uwierzytelnienia i zwróci odpowiedź, podczas gdy opóźnienie spowodowane atakiem jest mniejsze niż sekunda.

Drugi możliwy atak, który może być stosunkowo łatwo zauważany (i udaremniony) przez sieć jest atakiem typu „class-mark". Podczas inicjalizacji rozmowy telefon komórkowy wysyła do sieci dane o swoich możliwościach szyfrowania (informacja ta nosi nazwę „class-mark"). Większość telefonów komórkowych obsługuje obecnie A5/1, A5/2 i A5/0 (brak szyfrowania), ale może się to zmieniać dla różnych telefonów i w przyszłości. Atakujący zmienia (stosując np. atak typu man-in-the-middle) informację „class-mark", którą wysłał

telefon komórkowy i w ten sposób sieć myśli, że ten telefon może obsługiwać tylko A5/2 i A5/0. Teraz sieć standardowo wybierze A5/2 co umożliwi atakującemu słuchanie rozmowy. Atak ten wykorzystuje następujący słaby punkt protokołu: informacja „class-mark" nie jest chroniona.

Wiele sieci rzadko inicjuje procedurę uwierzytelnienia, a stosuje klucz wytworzony podczas ostatniego uwierzytelnienia. Atakujący może odkryć ten klucz poprzez podanie się za sieć wobec telefonu komórkowego ofiary. Teraz atakujący rozpoczyna sesję radiową z ofiarą i prosi jej telefon komórkowy o rozpoczęcie szyfrowania za pomocą A5/2. Atakujący przeprowadza atak, uzyskuje klucz i kończy sesję radiową. Właściciel tego telefonu komórkowego i sieć nie znajdą żadnych oznak tego ataku.

Przełożenie ataków w pierwszym i ostatnim przypadku opiera się na fakcie, że ten sam klucz jest ładowany do A5/2 i A5/1, a nawet do 64-bitowego A5/3 (w przypadku gdy A5/3 jest stosowany w GSM zgodnie ze standardami GSM). Stąd odkrycie klucza dla A5/2 ujawnia także klucz dla A5/1 i 64-bitowego A5/3. Zauważmy, że mimo, że A5/3 może być używany z kluczami długości od 64 do 128 bitów, to standard GSM pozwala na stosowanie tylko 64-bitowego A5/3.

Podobny atak można przeprowadzić na GPRS. Atakujący słucha GPRS-RAND wysłanego przez sieć do użytkownika. Atakujący może podać się za sieć głosową, zainicjować sesję radiową z użytkownikiem i rozpocząć procedurę uwierzytelnienia wykorzystując przechwyconą wartość GPRS-RAND, jako GSM-voice RAND. W rezultacie K_c będzie równe GPRS-K_c. Atakujący prosi użytkownika o szyfrowania za pomocą A5/2, znajduje klucz K_c i kończy sesję radiową. Atakujący może teraz szyfrować i deszyfrować komunikację GPRS użytkownika korzystają z uzyskanego K_c. Alternatywnie, atakujący może zapisać komunikację użytkownika i przeprowadzić podanie się za sieć w dowolnym późniejszym czasie w celu uzyskania GPRS-K_c, którym zapisane dane można odszyfrować. Jeśli telefon nie obsługuje A5/2, ale obsługuje A5/1, to powyższe ataki na A5/3 i GPRS można przeprowadzić stosując (bardziej złożoną) kryptoanalizę A5/1 zamiast A5/2 jako elementu składowego.

6. Bierna kryptoanaliza z samym szyfrogramem komunikacji szyfrowanej GSM-A5/1

W tym rozdziale przedyskutujemy możliwość ataków z samym szyfrogramem na komunikację GSM, szyfrowaną za pomocą A5/1. W przeciwieństwie do rozdziału 5, rozdział ten omawia ataki bierne, tzn. ataki, które nie transmitują żadnych sygnałów radiowych.

Wyjściowym punktem tych biernych ataków z samym szyfrogramem jest to, że kody korekcji błędów stosowane są przed szyfrowaniem, jak to omówiono w rozdziale 4.

Zastosujemy zależność czasu, pamięci i danych, podobnie do tej jaką zaprezentowali Biryukov, Shamir i Wagner [4], a potem omówili i uogólnili Biryukov i Shamir[3]. Ich oryginalny atak wymagał dwóch sekund znanego tekstu jawnego i wykonywał się w ciągu kilku minut na PC. Wymagał także czasu na obliczenia wstępne około 2^{48} i pamięci 4 dysków 73GB.

W naszym ataku znamy blok czterech zaszyfrowanych ramek. Stosujemy metodę z rozdziału 4 do obliczenia $K_G \cdot k$ o długości 272 bitów. Te 272 bity zależą tylko od wyjścia z A5/1 dla czterech kolejnych ramek. Nazywamy te bity wyjścia „strumieniem kodowanym" (coded-stream). Załóżmy, że wiemy, że numer pierwszej z czterech ramek dzieli się przez 4 bez reszty, założenie to ogranicza nas do wykorzystania średnio tylko ćwierci strumienia danych GSM. Cały proces możemy rozpatrywać jako funkcję stanu wewnętrznego A5/1 na strumień kodowany. Niech $h:\{0,1\}^{64} \rightarrow \{0,1\}^{272}$ będzie funkcją, która pobiera stan wewnętrzny A5/1 po inicjalizacji i daje strumień kodowany. Odwrotność h wyznacza stan wewnętrzny i łamie szyfr. Zauważmy, że wyjście funkcji h obliczymy na podstawie strumienia klucza z czterech ramek, podczas gdy wejście jest stanem wewnętrznym po inicjalizacji A5/1 w pierwszej ramce. Założenie, które przyjęliśmy na numery ramek ułatwia obliczenie początkowych stanów wewnętrznych dla pozostałych trzech ramek na podstawie początkowego stanu wewnętrznego w pierwszej ramce. Wystarczy spojrzeć tylko na 64 bity wyjścia h, stąd możemy założyć, że $h:\{0,1\}^{64} \rightarrow \{0,1\}^{64}$.

Dostępne dane D	M	Liczba 200GB dysków	P=N/D	Liczba PC-tów do wykonania obl. wstęp. w rok	T	Liczba PC-tów do wykonania ataku w czasie rzeczywistym
2^{12} (≈ 5 min)	2^{38}	≈ 22	2^{52}	140	2^{28}	1
$2^{6,7}$ (≈ 8 sec)	2^{41}	≈ 176	$2^{57,3}$	5000	$2^{32,6}$	1000
$2^{6,7}$ (≈ 8 sec)	2^{42}	≈ 350	$2^{57,3}$	5000	$2^{30,6}$	200
2^{14} (≈ 20 min)	2^{35}	≈ 3	2^{50}	35	2^{30}	1

Tabela 1. Cztery punkty na krzywej zależności czas/pamięć/dane.

Stosujemy atak zależności czasu, pamięci i danych Biryukova i Shamira z próbkowaniem [3] i stosujemy ich notację: liczbą możliwych stanów jest $N=2^{64}$, liczbę cztero-ramkowych bloków, które mamy oznaczymy przez D, T jest liczbą zastosowań h podczas fazy czasu

rzeczywistego ataku, M jest liczbą wierszy pamięci (w naszym przypadku każdy wiersz ma długość około 16 bajtów), a krzywa zależności ma postać: $TM^2D^2 = N^2$, $D^2 \leq T \leq N$. Złożoność obliczeń wstępnych P=N/D jest liczbą zastosowań h podczas obliczeń wstępnych. Atak dokonuje około $\sqrt{T}$ odwołań do pamięci. Zakładamy, że h można wykonać 2^{20} razy na sekundę na PC. Listę czterech punktów na krzywej zależności pokazuje tabela 1.

Dokładniejszy opis ataku biernego ukaże się w pełnej wersji tego artykułu.

7. Możliwe scenariusze ataków

Ataki zaprezentowane w tym artykule można zastosować w wielu scenariuszach. W tym rozdziale zaprezentowano cztery z nich: podsłuch rozmowy, przechwycenie rozmowy, zamiana danych wiadomości (sms) i kradzież rozmowy – dynamiczne klonowanie.

7.1 Podsłuch rozmowy (call wire-tapping)

Najbardziej naiwnym scenariuszem, który możemy przewidzieć jest podsłuchiwanie rozmów. Komunikacja zaszyfrowana za pomocą GSM może być odszyfrowana i podsłuchana przez atakującego, jeśli tylko atakujący zna klucz szyfrujący. Zarówno rozmowę głosową jak i dane, np. wiadomość SMS można podsłuchać. Wiadomości wideo i obrazkowe przesyłane za pomocą GPRS także można przechwycić. Ataki takie opisano w rozdziale 5.

W innym możliwym ataku podsłuchiwania atakujący może nagrywać zaszyfrowane rozmowy. Atakujący musi mieć pewność, że zna wartość RAND, która tworzy używany klucz (RAND jest przesyłana w formie niezaszyfrowanej). W późniejszym czasie, gdy mu jest wygodnie atakujący podszywa się za sieć wobec ofiary. Następnie atakujący inicjuje sesję radiową GSM, prosząc ofiarę o przeprowadzenie uwierzytelnienia z powyższym RAND i uzyskuje klucz sesji stosowany w nagranych rozmowach. Gdy tylko atakujący ma klucz łatwo odszyfruje rozmowę i wysłucha jej zawartości. Zauważmy, że atakujący może nagrywać wiele rozmów i w późniejszych atakach po kolei uzyskać wszystkie klucze. Atak ten ma tę zaletę, że może być przeprowadzony w czasie wygodnym dla atakującego, nawet po wielu latach nagrywania rozmów, lub gdy ofiara jest innym krajem, lub w miejscu wygodnym dla atakującego.

7.2 Przechwycenie rozmowy (call hijacking)

Gdy sieć GSM może przeprowadzić uwierzytelnienie na początku rozmowy, szyfrowanie jest sposobem GSM-u na przeciwdziałanie podszywaniu się w późniejszych fazach rozmowy.

Podstawowym założeniem jest, że oszust nie posiada K_c i dlatego nie może przeprowadzać szyfrowanych komunikacji. Pokażemy, jak zdobyć klucze szyfrujące. Jeśli atakujący zna klucze szyfrujące może wyłączyć ofiarę z rozmowy i podszyć się za nią wobec kogoś innego. Dlatego też przechwycenie rozmowy po uwierzytelnieniu jest możliwe. Przechwycenie może nastąpić w trakcie wczesnego nawiązywania rozmowy (call-setup), nawet przed tym jak telefon ofiary zacznie dzwonić. Operator właściwie nie może podejrzewać, że jest to atak. Jedynym możliwym punktem wykrycia tego ataku jest moment pewnej zwiększonej interferencji elektromagnetycznej.

Innym sposobem przechwycenia rozmów przychodzących jest przeprowadzenie ataku typu man-in-the-middle, ale zamiast przekazywania rozmowy do ofiary atakujący przechwytuje ją.

7.3 Zamiana danych wiadomości (sms)

Jeśli tylko rozmowa została przechwycona atakujący decyduje o jej zawartości. Może słuchać zawartości wiadomości przesyłanej przez ofiarę i wysyłać swoją wersję. Atakujący może przerwać wiadomość lub wysłać swoją własną wiadomość SMS. Kompromituje to integralność komunikacji GSM.

7.4 Kradzież rozmowy – dynamiczne klonowanie (call theft – dynamic cloning)

GSM jest uważana za bezpieczną przeciwko kradzieży rozmów, ze względu na stosowane procedur uwierzytelniania A3/A8 (przynajmniej dla operatorów, którzy stosują silne algorytmy A3/A8 zamiast COMP128).

Jednakże, w związku ze wspomnianą słabością, atakujący może dokonywać połączeń wychodzących na koszt ofiary. Gdy sieć prosi o uwierzytelnienie atak typu man-in-the-middle podobny do opisanego w rozdziale 5 może być zastosowany: atakujący inicjuje rozmowę wychodzącą do sieci komórkowej równolegle z sesją radiową z ofiarą. Gdy sieć poprosi atakującego o uwierzytelnienie, atakujący poprosi o uwierzytelnienie ofiarę i przekazuje uzyskane uwierzytelnienie z powrotem do sieci. Atakujące może także odkryć klucz K_c jak opisano w rozdziale 5. Teraz atakujący może zakończyć sesję z ofiarą i kontynuować rozmowę wychodzącą z siecią. Atak ten jest trudno wykrywalny przez sieć, gdyż sieć uważa to za normalny dostęp. Telefon ofiary nie dzwoni, a ofiara nie ma żadnych oznak, że jest ofiarą ataku, aż do nadejścia rachunku miesięcznego.

8. Podsumowanie

W tym artykule zaprezentowaliśmy nowe metody ataków na szyfrowanie w GSM i GPRS i na protokoły bezpieczeństwa. Opisane ataki są łatwe do przeprowadzenia i nie wymagają znajomości rozmowy. Podkreślamy, że operatorzy GSM powinni zastąpić algorytmy kryptograficzne i protokoły jak najszybciej lub przejść na bardziej bezpieczny system telefonii komórkowej trzeciej generacji.

W GSM, nawet sieci GSM używające nowego algorytmu A5/3 poddają się naszemu atakowi. Sugerujemy zmianę sposobu włączania A5/3 w ochronę sieci przed takimi atakami. Możliwą poprawą jest sprawienie, aby klucze stosowane w A5/1 i A5/2 nie zależały od kluczy stosowanych w A5/3. Włączanie GPRS posiada podobne wady, co powinno być wzięte pod rozwagę.

Pragniemy podkreślić, że nasz atak z samym szyfrogramem jest możliwy poprzez fakt, że kody korekcji błędów są stosowane przed szyfrowaniem. W przypadku GSM dodanie takiej nadmiarowości strukturalnej przed szyfrowaniem jest powodem istotnego obniżenia bezpieczeństwa systemu.