INSTYTUT ŁĄCZNOŚCI
PAŃSTWOWY INSTYTUT BADAWCZY

Zakład Radiokomunikacji Morskiej w Gdańsku (Z-8)

Koncepcja systemu bezpiecznej transmisji w paśmie krótkofalowym – Etap I

Praca nr 08300018

Gdańsk, listopad 2008

Koncepcja systemu bezpiecznej transmisji w paśmie krótkofalowym – Etap I

Praca nr 08300018

Słowa kluczowe: Kryptografia, szyfracja, poufność, uwierzytelnianie, autoryzacja, integralność, dostępność, transmisja danych, kanał krótkofalowy, modem HF

Kierownik pracy: mgr inż. Krzysztof Bronk

Kierownik zakładu: dr inż. Rafał Niski

Spis treści

Wprowadzenie

W sieciach łączności bezprzewodowej, zwłaszcza o charakterze globalnym, istotnym problemem jest zapewnienie bezpieczeństwa transmisji danych. Medium bezprzewodowe jest dużo bardziej podatne na podsłuch niż medium przewodowe, przez co jest wrażliwe na niepożądane działania osób do tego nieupoważnionych, których celem jest nieautoryzowane zdobycie informacji, a często również wprowadzenie szkodliwych danych do sieci. W dobie terroryzmu, gdy bardzo często poufne przesyłanie wiadomości ma znaczenie krytyczne, zagadnienie to zasługuje na tym większą uwagę.

W sytuacji globalnego zagrożenia, szczególnie ważne pod względem użytecznym są satelitarne systemy bezprzewodowe z natury rzeczy zapewniające łączność globalną. Niestety, z oczywistych względów, systemy te są mało odporne na działania destrukcyjne. W takiej sytuacji należy szukać innego sposobu bezprzewodowej, globalnej transmisji danych. W ostatnim czasie rozwiązania tego problemu upatruje się w wykorzystaniu łączności krótkofalowej (HF), która między innymi dzięki odbiciom fal radiowych od warstw jonosfery umożliwia transmisję dalekosiężną. Obecnie systemy radiowe działające na falach krótkich zyskują ponownie na znaczeniu, zwłaszcza do zastosowań specjalnych i w niedalekiej przyszłości mogą stać się bezpiecznym rozwiązaniem, służącym między innymi do przesyłania ważnych, a często także tajnych informacji, w sytuacjach zagrożenia bezpieczeństwa i nie tylko.

Celem pracy jest zatem dobór i modyfikacja efektywnych algorytmów szyfrujących, zapewniających integralność i autentyczność danych, jak również mechanizmów uwierzytelniania i zarządzania kluczami szyfrującymi.

Niniejsza praca jest pierwszym etapem na drodze do stworzenia w pełni funkcjonalnego modułu kryptograficznego możliwego do wykorzystania w dzisiejszych modemach krótkofalowych zrealizowanych w technologii radia programowalnego. Praca ta stanowi również autorski wkłada do otwartego 21 września 2007 roku przez Radę Naukową Instytutu Łączności przewodu doktorskiego na temat „Badania i analiza problemu bezpiecznej transmisji danych w paśmie krótkofalowym”.

1. Podstawowe informacje związane z kryptografią

Ochroną informacji zajmuje się kryptologia, czyli nauka o szyfrowaniu – tzn. o bezpiecznych sposobach przekazywania informacji. Bezpiecznych, to znaczy uniemożliwiających ich odczytanie przez osoby niepowołane, które nie posiadają tzw. klucza do informacji, za pomocą którego można ją odszyfrować. Kryptologia dzieli się na kryptografię i kryptoanalizę – dwie przeciwstawne dziedziny. Kryptografia zajmuje się zabezpieczaniem informacji a kryptoanaliza próbami obejścia tych zabezpieczeń lub ich złamania. Kryptologia zawdzięcza swoje powstanie i rozwój walce tych przeciwstawnych dziedzin.

Szyfrowanie jest rozumiane jako zamiana postaci informacji z jawnej na tajną. Dokonuje się tego poprzez zastosowanie określonego algorytmu kryptograficznego i dodatkowej informacji – tzw. klucza, bez którego (nawet znając algorytm szyfrowania) nie da się zdeszyfrować danej wiadomości, czyli zamienić jej postaci z powrotem z tajnej na jawną.

Bezpieczeństwo systemów kryptograficznych w dużej mierze opiera się na tzw. mechanizmach kryptografii. Zanim jednak zostaną one przedstawione, należy wyjaśnić kilka podstawowych pojęć [1]:

Informacja jawna (tekst jawny) – Niezaszyfrowane dane, które mogą być odczytane przez każdego.

Informacja tajna (szyfrogram, tekst zaszyfrowany) – Dane zaszyfrowane, odczyt niemożliwy bez znajomości algorytmu szyfrującego i klucza.

Szyfrowanie – Proces zamiany informacji jawnej na tajną.

Deszyfrowanie – Proces zamiany informacji tajnej na jawną.

Klucz – tajna wartość używana w algorytmie kryptograficznym do szyfrowania i deszyfrowania, znana zwykle tylko nadawcy i uprawnionemu odbiorcy danej szyfrowanej wiadomości. Wyróżniamy klucze tajne, prywatne i publiczne.

Algorytm z kluczem symetrycznym (szyfrowanie symetryczne) – Szyfrowanie, w którym zarówno dla szyfrowania jak i deszyfrowania wiadomości konieczna jest znajomość tego samego klucza.

Algorytm z kluczem asymetrycznym (szyfrowanie asymetryczne) – Szyfrowanie, w którym do szyfrowania używany jest jeden klucz, a do deszyfrowania inny klucz (klucz prywatny i publiczny).

Kryptosystem (system kryptograficzny) – Zespół mechanizmów, w ramach których odbywa się szyfrowanie, transmisja i deszyfrowanie, wraz z zapewnieniem podstawowych mechanizmów kryptografii.

Funkcja skrótu (funkcja haszująca) – funkcja dostarczająca skróconego sprawdzenia wiadomości jawnej, zwykle używana do sprawdzania integralności wiadomości, dostarczająca coś na kształt odcisku palca danej wiadomości, charakteryzująca się tym, że na podstawie skrótu nie można wyznaczyć oryginału wiadomości, z którego został sporządzony skrót, ale dana wiadomość może zostać powiązana z konkretnym skrótem (tej samej wiadomości odpowiada zawsze ten sam skrót).

1.1. Mechanizmy kryptograficzne

Aby system ochrony przesyłanych informacji był w pełni bezpieczny należy zrealizować główne mechanizmy kryptografii [1]:

- uwierzytelnianie,
- poufność,
- integralność,
- dostępność,
- autoryzacja.

1.1.1. Uwierzytelnianie

Informacja jest wierzytelna, jeżeli pochodzi z wiarygodnego źródła. Prostym przykładem weryfikacji jest rozmowa telefoniczna, w której obie strony naturalnie identyfikują się wzajemnie za pomocą głosu (jeżeli rozmawiające osoby się znają). W sytuacji, gdy osoby się nie znają lub jakość głosu przesyłanego przez kanał jest zła i uniemożliwia identyfikację, jak również w przypadku przesyłania danych należy zastosować specjalne mechanizmy umożliwiające uwierzytelnianie. Jednym ze sposobów weryfikacji jest zastosowanie mechanizmu MAC (ang. *Message Authentication Code*) [1]. W tej metodzie do wiadomości jawnej, dołącza się niewielką ilość danych zaszyfrowanych za pomocą kodu MAC i klucza tajnego. Po stronie odbiorczej następuje odszyfrowywanie kodu MAC, ponieważ strona odbiorcza posiada ten sam klucz tajny, co nadawca (jest to przykład algorytmu symetrycznego). Dzięki zgodności kodów po obu stronach oraz znajomości algorytmu szyfrującego, jest możliwa weryfikacja nadawcy. Rozwiązanie to ma jednak pewne wady, ponieważ każdy nadawca i odbiorca musi przechowywać w swoim urządzeniu nadawczo-odbiorczym klucze wszystkich osób, z którymi chciałby się komunikować lub wymieniać dane, co może prowadzić do nadużyć – wszyscy posiadają klucze wszystkich i jedna osoba może się podszyć pod inną. Rozwiązanie to sprawdzi się tylko w niewielkiej lub zaufanej sieci. Warto tu zauważyć, że jeden klucz przydzielić można całej grupie osób. Wtedy taki klucz identyfikuje nie jednostkę, a całą grupę osób.

Alternatywą dla tej sytuacji jest użycie podpisów cyfrowych przy wykorzystaniu kluczy prywatnych i publicznych – jest to przykład szyfrowania asymetrycznego – oraz funkcji skrótu (funkcji haszujących). Podpis cyfrowy jest dołączany do oryginalnej wiadomości. Dzięki niemu, odbiorca wiadomości jest w stanie zweryfikować nadawcę. Nie posiada jednak jego klucza prywatnego, tylko klucz publiczny. Za pomocą klucza publicznego można odszyfrować wiadomość zaszyfrowaną kluczem prywatnym i odwrotnie.

Innym sposobem jest uwierzytelnianie czasowe. Wymagane jest wtedy, by wszystkie komunikujące się ze sobą urządzenia miały jednakowe czasy systemowe (najczęściej dopuszczalny jest jednak pewien margines błędu).

1.1.2. Poufność

Informacja jest poufna, jeżeli bez znajomości klucza (tajnej informacji) i algorytmu szyfrującego jest teoretycznie niemożliwe jej odczytanie (odszyfrowanie). Zwykle poufność uzyskuje się za pomocą algorytmu szyfrującego z użyciem klucza tajnego. Do zapewnienia poufności można wykorzystać zarówno symetryczne jak i asymetryczne szyfrowanie. Korzyścią symetrycznego jest to, iż zwykle jest ono wydajniejsze. Z tego powodu duże ilości danych powinno się szyfrować algorytmami z kluczem symetrycznym.

1.1.3. Integralność

Integralność informacji jest zapewniona, gdy jest pewność, że wiadomość nie została zmieniona od momentu wysłania jej przez wiarygodne źródło do dotarcia do odbiorcy. Integralność może być zapewniona przez wykorzystanie podpisu cyfrowego. Wtedy źródłem dla generowania podpisu przez algorytm jest cała wiadomość. Zmiana jakiegokolwiek elementu wiadomości spowoduje, że wygenerowany podpis na podstawie zmienionej wiadomości będzie się różnił od podpisu dołączonego do zmienionej wiadomości. Po stronie odbiorczej można będzie przeprowadzić weryfikację – dzięki użyciu klucza publicznego. Jeżeli otrzymany podpis różni się od dołączonego do wiadomości oznacza to, że wiadomość została zmieniona w kanale, w którym była transmitowana.

1.1.4. Dostępność

Dostępność do informacji (a właściwie ograniczanie tej dostępności) polega na tym, że niepowołane osoby nie mogą korzystać z danej informacji lub z urządzenia przechowującego ją. Najprostszym przykładem zapewnienia dostępności jest zwykły zamek w drzwiach chroniący mieszkanie – mający klucz dostaną się do mieszkania. Ci nieposiadający klucza teoretycznie nie mają do niego. W systemach bezpieczeństwa dostępność do usługi, urządzenia, czy danych jest zwykle realizowana za pomocą osobistego hasła, numeru PIN, osobistego numeru identyfikacyjnego. Tutaj ważna jest sprawa związana z tzw. siłą hasła. Inną metodą ograniczania dostępu mogą być właściwości biometryczne osoby – cechy niepowtarzalne dla danych osób, takie jak: odcisk palca, skan tęczówki oka, czy identyfikacja za pomocą głosu lub nawet DNA.

1.1.5. Autoryzacja

Autoryzacja jest procesem, w którym ustala się, czy dana osoba ma uprawnienia do dostępu do danych lub usługi, której żąda. Proces ten jest często mulony z uwierzytelnianiem. Jednak po potwierdzeniu tożsamości użytkownika, czyli uwierzytelnieniu, należy jeszcze sprawdzić, czy faktycznie ma on prawo do otrzymania danych, do których próbuje się dostać. Najprościej można to wyjaśnić następująco: proces uwierzytelniania sprawdza, kim jest osoba próbująca uzyskać dostęp do systemu, a proces autoryzacji sprawdza, czy dana osoba ma prawo korzystać z określonego zasobu. Mechanizm ten jest często wykorzystywany do hierarchizacji systemów bezpieczeństwa. Jako przykład można tu podać hierarchię kont użytkowników w komputerach PC – rozdział na administratora i użytkownika z ograniczonymi uprawnieniami. Tylko administrator może zakładać nowe konta użytkowników i zmieniać ich uprawnienia, podczas gdy użytkownik z ograniczeniami może tylko np. korzystać z programów zainstalowanych na komputerze.

2. Algorytmy kryptograficzne

W poniższym rozdziale zostaną przedstawione przykładowe algorytmy kryptograficzne z uwzględnieniem jednak tylko tych, które wykorzystywane będą w zaproponowanym systemie bezpiecznej transmisji danych..

2.1. Podstawowe algorytmy szyfrujące

Do najprostszych algorytmów kryptograficznych należą szyfry podstawieniowe. Jednym z szyfrów podstawieniowych jest algorytm Cezara. W takim szyfrze każda litera lub grupa liter tekstu jawnego jest zastąpiona inną literą lub inną grupą liter. W klasycznej kryptografii wyróżnia się cztery grupy szyfrów podstawieniowych [2]:

- monoalfabetyczne;
- homofoniczne;
- wieloalfabetowe;
- poligramowe.

Szyfry monoalfabetowe zawierają takie same znaki zarówno w tekście jawnym, jak i w szyfrogramie. W szyfrach homofonicznych każdemu znakowi tekstu jawnego przyporządkowuje się po kilka znaków kryptogramu. Szyfry wieloalfabetowe stanowią kombinacje wielu prostych szyfrów podstawieniowych. W szyfrach poligramowych szyfruje się grupy znaków. Nie zostaną one tu szczegółowo opisane, ze względu na charakter niniejszej pracy, która ma na celu prezentację rozwiązań, które zostaną wykorzystane w projektowanym systemie.

Kolejną grupą szyfrów są szyfry przedstawieniowe (permutacyjne). Zmieniają on kolejność znaków w tekście zaszyfrowanym. Natomiast szyfrem kaskadowym jest szyfr, w którym połączono podstawowe metody szyfrowania, jakimi są pojedyncze podstawienia i przestawienia. Szyfry kaskadowe są połączeniem szyfrów podstawieniowych i przestawieniowych. Mają one lepsze właściwości kryptograficzne niż samo podstawienie i przestawienie. Szyfry te są realizowane w maszynach rotorowych (np. Enigma) i algorytmach komputerowych (DES [3], AES [4]).

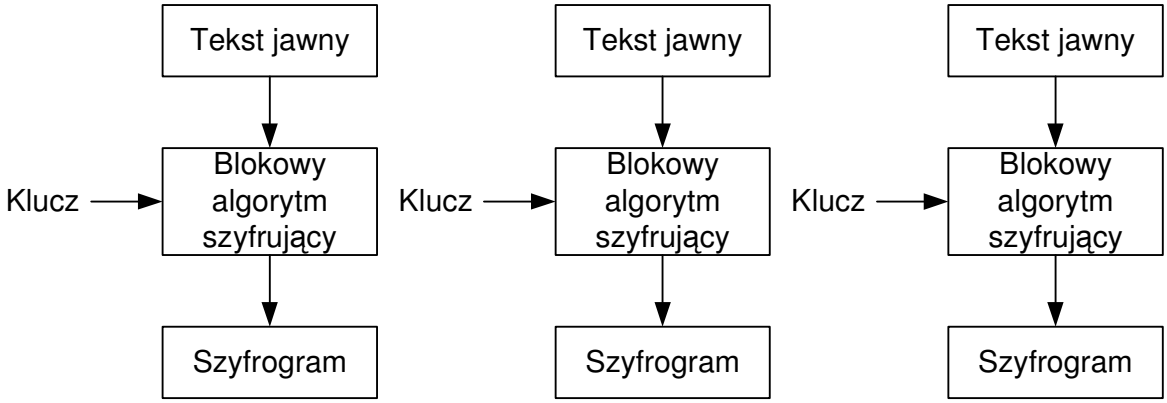
2.2. Tryby pracy blokowych algorytmów kryptograficznych

Istnieją dwa typy algorytmów symetrycznych: szyfry blokowe i szyfry strumieniowe [1]. Szyfry blokowe wykonują operacje na blokach tekstu jawnego lub szyfrogramu. Bloki te mają zwykle 64 lub więcej bitów długości (zwykle wielokrotność tej wartości). Natomiast szyfry strumieniowe działają na ciągach tekstu jawnego i szyfrogramu za każdym razem po jednym bicie, bajcie lub słowie. W przypadku szyfru blokowego jeden blok tekstu jawnego jest zamieniany w blok szyfrogramu o tej samej długości. W przypadku szyfru strumieniowego, jeden bit, bajt czy słowo tekstu jawnego jest zamieniane w odpowiednio bit, bajt lub słowo szyfrogramu. Tryby pracy blokowych algorytmów kryptograficznych stosuje się czasami, aby uzyskać właściwości szyfrowania zbliżone do właściwości szyfrów strumieniowych [1].

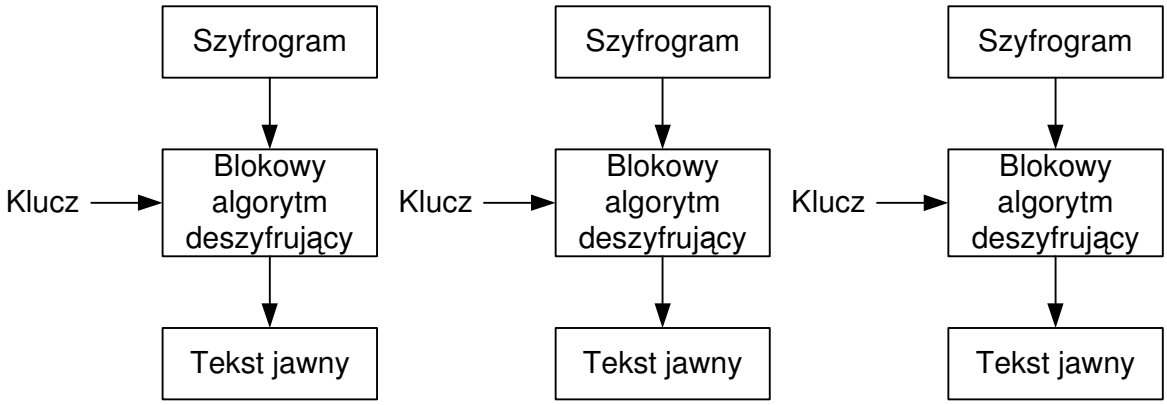
Poniżej przedstawiono tryby pracy szyfrów blokowych [5].

Jednym z trybów pracy blokowych algorytmów kryptograficznych jest tryb elektronicznej książki kodowej (ECB – *Electronic Code Book*). Ponieważ jeden blok tekstu jawnego jest szyfrowany w blok szyfrogramu. Możliwe jest zatem stworzenie tzw. książki kodowej, czyli

tablicy ze wszystkimi możliwymi przekształceniami bloku tekstu jawnego w blok szyfrogramu. Dla każdego klucza tablica kodowa ma inną postać. Przykładowo, jeśli długość bloku ma 64 bity, to książka kodowa ma 2^{64} możliwych wartości wejściowych dla jednego klucza. Zaletą takiego rozwiązania jest możliwość niezależnego szyfrowania każdego bloku tekstu jawnego. Przykładowo można zaszyfrować tylko część bloków, bez konieczności szyfrowania reszty. Bardzo istotne jest to dla danych, do których możliwy jest dostęp losowy do dowolnego miejsca. Na przykład w bazach danych każdy rekord może być szyfrowany niezależnie. Podczas modyfikowania, wstawiania i kasowania rekordu nie trzeba deszyfrować poprzednich rekordów. Podejście to ma jednak sporą wadę. Identyczne bloki tekstu jawnego posiadać będą zawsze identyczną postać szyfrogramu, co ułatwiać może deszyfrację informacji. Tryb ten jest jednak wydajny: szybkość szyfracji jest taka jak samego szyfru blokowego, szyfrogram jest dłuższy od tekstu jawnego co najwyżej o długość dopełnienia i przetwarzanie może być zrównoleglone. Rozwiązanie to zabezpiecza również przed dodatkowymi błędami: błąd w szyfrogramie wpływa tylko na jeden blok tekstu jawnego a błąd synchronizacji nie jest odtwarzalny. Schematy blokowe prezentujące sposób szyfracji i deszyfracji z wykorzystaniem tej metody przedstawiono na rysunkach odpowiednio 2.1 oraz 2.2.



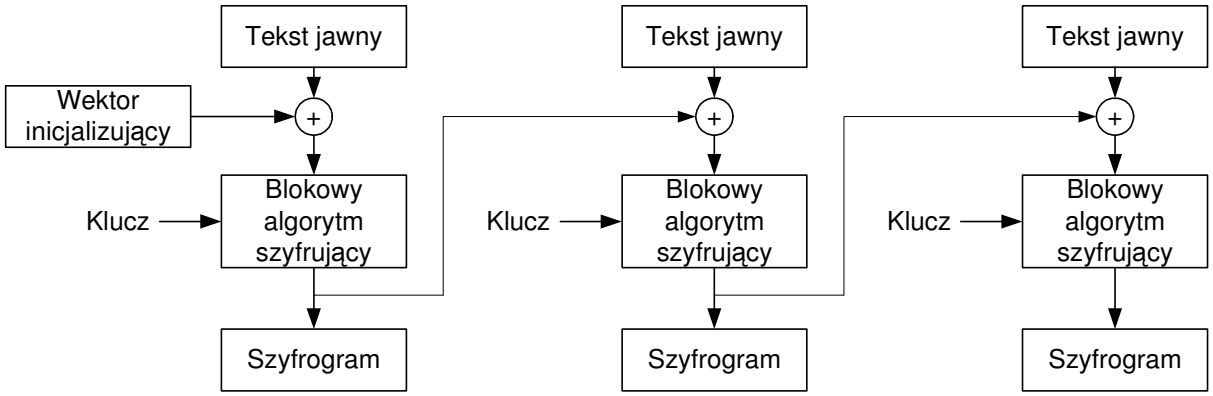
Rys. 2.1. Szyfracja z wykorzystaniem szyfrów blokowych w trybie ECB.



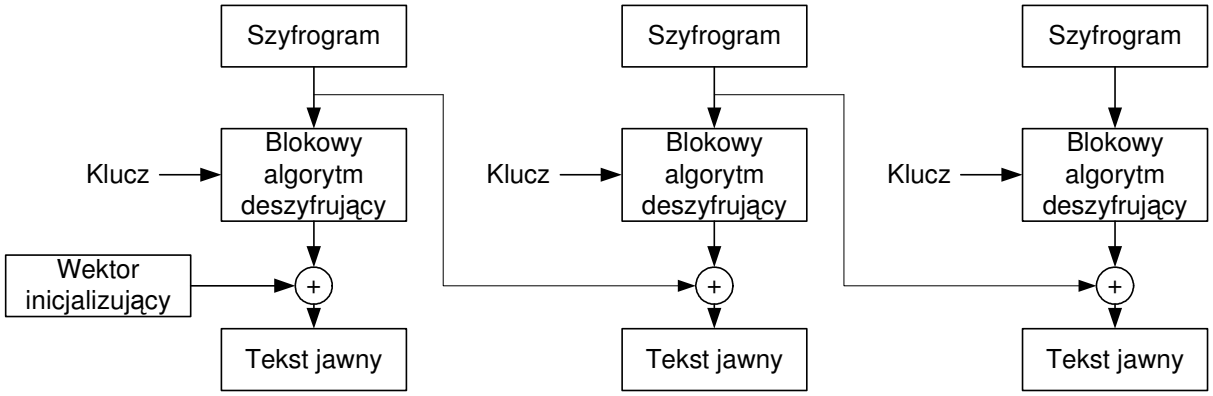
Rys. 2.2. Deszyfracja z wykorzystaniem szyfrów blokowych w trybie ECB.

Innym trybem jest tryb wiązania bloków zaszyfrowanych (CBC – *Cipher Block Chaining).* Wiązanie to polega na wykorzystaniu mechanizmu sprzężenia zwrotnego, czyli wyniki szyfrowania poprzedniego bloku są danymi wejściowymi dla operacji szyfrowania kolejnego bloku. Każdy kolejny zaszyfrowany blok jest wykorzystywany do modyfikacji szyfrowania bloku następnego, czyli każdy blok szyfrogramu zależy od bieżącego tekstu jawnego i poprzedniego bloku zaszyfrowanego. Wykorzystywany jest również wektor inicjalizujący, który zapewnia pseudolosową postać pierwszego bloku wejściowego. Blok tekstu jawnego przed zaszyfrowaniem jest sumowany modulo 2 z poprzednim blokiem szyfrogramu. Ze względów

bezpieczeństwa ten tryb jest lepszy od trybu ECB, bo charakterystyczne fragmenty tekstu są ukrywane poprzez wykonywanie operacji modulo 2 z poprzednimi blokami a ciąg wejściowy szyfru blokowego ma charakter losowy. Więcej niż jedna wiadomość może być zaszyfrowana tym samym kluczem. Szyfr ten jest nieco mniej wydajny od ECB. Jego szybkość jest taka sama jak szybkość szyfru blokowego i szyfrogram jest co najwyżej dłuższy o długość dopełnienia w stosunku do tekstu jawnego, lecz szyfrowanie nie może być zrównoleglone, ze względu na to, że każdy kolejny szyfrowany blok zależy od poprzednich. Błąd w szyfrogramie wpływa na jeden cały blok tekstu jawnego i odpowiadający mu bit w bloku następnym. Błąd synchronizacji nie jest odtwarzalny. Deszyfrowanie może być zrównoleglone, ponieważ kolejne szyfrogramy potrzebne do odszyfrowania danych są dostępne od razu. Schematy blokowe prezentujące sposób szyfracji i deszyfracji z wykorzystaniem tej metody przedstawiono na rysunkach odpowiednio 2.3 oraz 2.4.



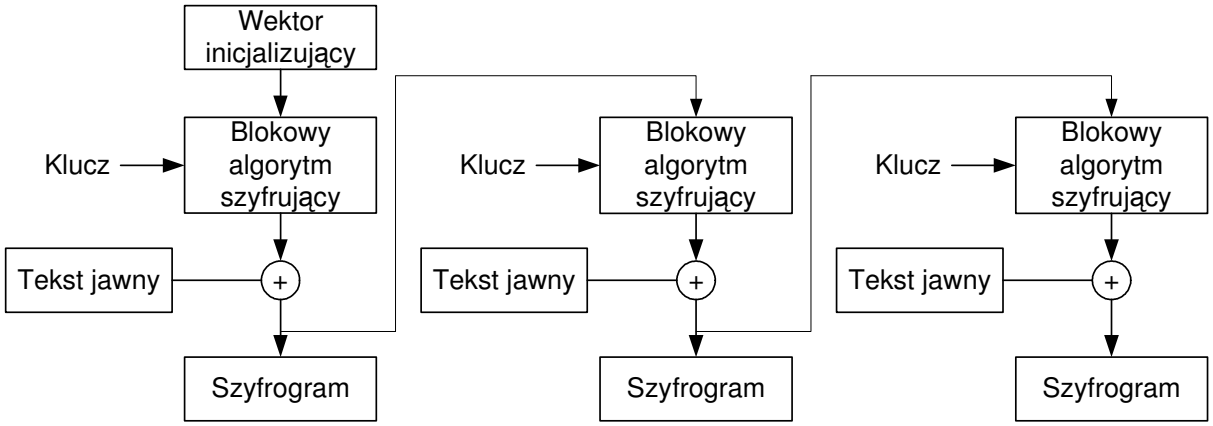
Rys. 2.3. Szyfracja z wykorzystaniem szyfrów blokowych w trybie CBC.



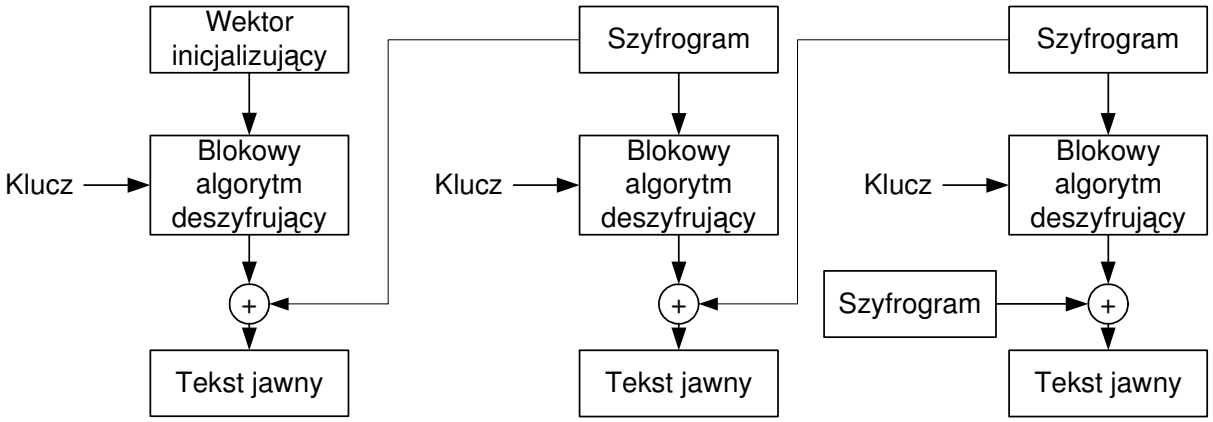
Rys. 2.4. Deszyfracja z wykorzystaniem szyfrów blokowych w trybie CBC.

Kolejnym trybem szyfrowania jest tryb sprzężenia zwrotnego szyfrogramu (CFB – *Cipher Feedback*). Podobnie jak algorytm CBC, ten algorytm łączy znaki tekstu jawnego w ten sposób, że szyfrogram zależy od całego poprzedzającego tekstu jawnego. Podobnie jak w trybie CBC, używany jest wektor inicjalizujący oraz dodatkowo parametr będący liczbą naturalną, który określa wielkość bloku szyfrogramu i tekstu jawnego. Bezpieczeństwo: charakterystyczne fragmenty tekstu jawnego są ukrywane, ciąg wejściowy szyfru blokowego ma charakter losowy i więcej niż jedna wiadomość może być zaszyfrowana tym samym kluczem, przyjmując, że będzie używany inny ciąg inicjujący. Szybkość szyfru jest taka sama, jak szybkość szyfru blokowego jedynie wtedy, gdy sprzężenie zwrotne jest takiego samego rozmiaru co dla bazowego szyfru blokowego. Podobnie jak poprzednio, szyfrogram jest tej samej długości, co tekst jawny plus dopełnienie do pełnego bloku. Szyfrowanie nie może być zrównoleglone, ale deszyfrowanie tak. Błąd w szyfrogramie wpływa na jeden bit odpowiadającego mu tekstu jawnego i cały następny blok tekstu jawnego. Błędy synchronizacji dla całych

bloków są odtwarzalne. Szyfrowanie nie może być zrównoleglone, natomiast deszyfrowanie w tym trybie może być zrównoleglone, jeżeli uprzednio bloki wejściowe zostaną skonstruowane z wektora inicjalizującego i szyfrogramów. Schematy blokowe prezentujące sposób szyfracji i deszyfracji z wykorzystaniem tej metody przedstawiono na rysunkach odpowiednio 2.5 oraz 2.6.

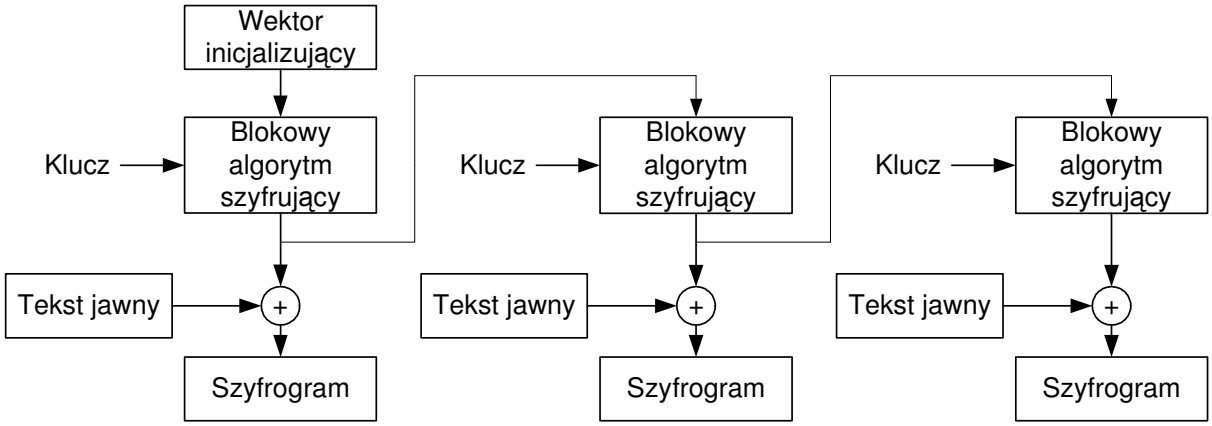


Rys. 2.5. Szyfracja z wykorzystaniem szyfrów blokowych w trybie CFB.

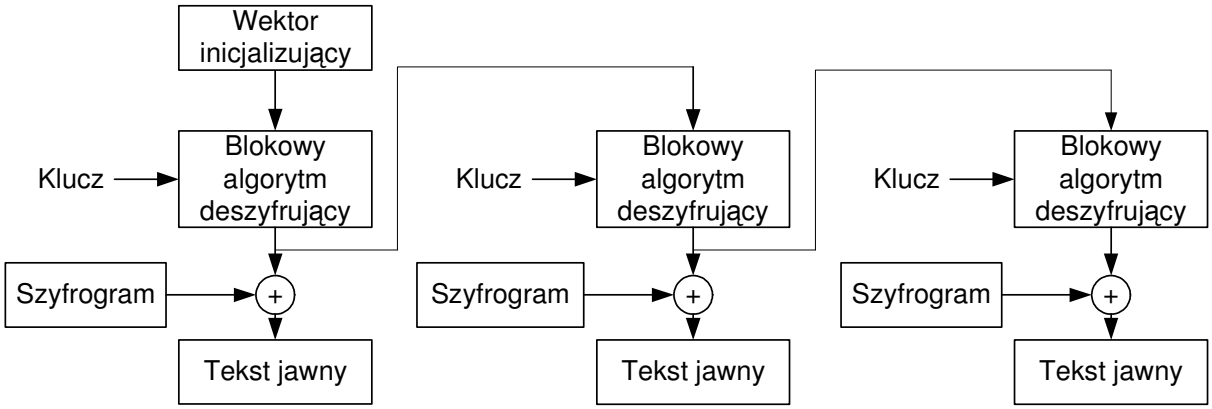


Rys. 2.6. Deszyfracja z wykorzystaniem szyfrów blokowych w trybie CFB.

Czwartym z trybów szyfrowania jest tryb sprzężenia zwrotnego wyjściowego (OFB – *Output Feedback*). Przetwarzanie tutaj nie może być zrównoleglone, błąd w szyfrogramie wpływa na odpowiadający mu bit tekstu jawnego i błędy synchronizacji nie są odtwarzane. Podobnie jak w poprzednich trybach stosuje się wektor inicjalizujący, który musi być unikalny dla każdego użycia szyfrowania z tym samym tajnym kluczem. Schematy blokowe prezentujące sposób szyfracji i deszyfracji z wykorzystaniem tej metody przedstawiono na rysunkach odpowiednio 2.7 oraz 2.8.

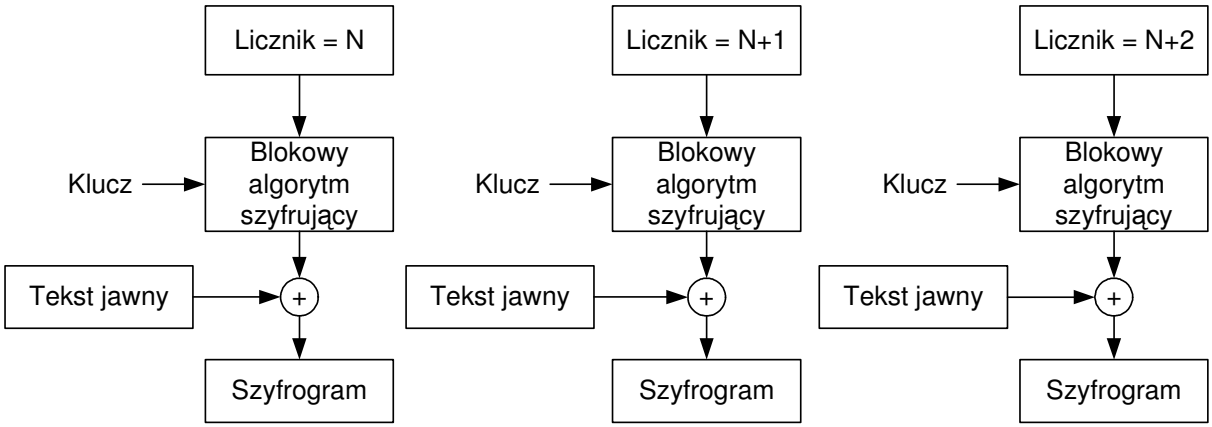


Rys. 2.7. Szyfracja z wykorzystaniem szyfrów blokowych w trybie OFB.

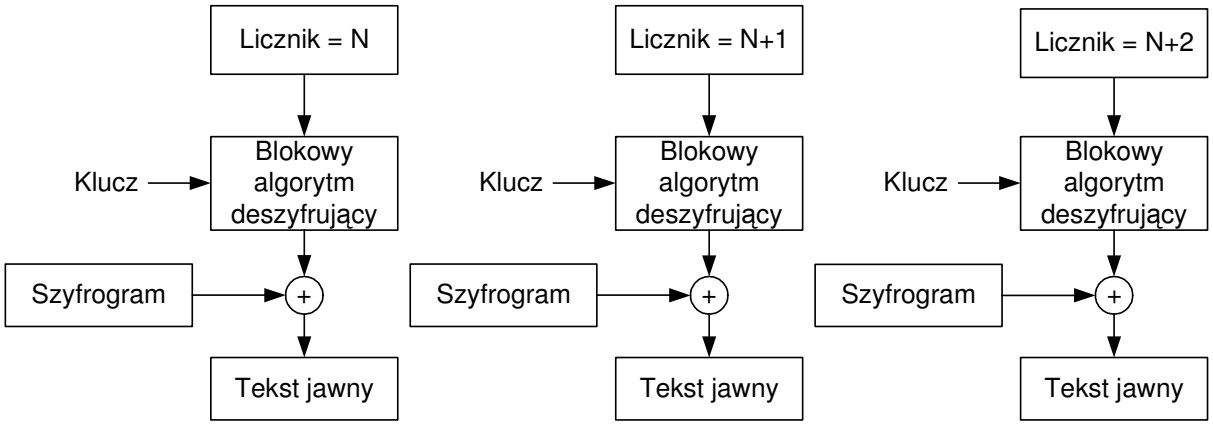


Rys. 2.8. Deszyfracja z wykorzystaniem szyfrów blokowych w trybie OFB.

Ostatnim trybem szyfrowania jest tryb licznikowy (*Counter Mode* – CTR). Tryb ten jest bardzo podobny do szyfrowania w trybie ECB, jednak wejściem do funkcji szyfrującej jest tu tzw. licznik, którego wartość przy szyfrowaniu każdego kolejnego bloku jest zwiększana. Tekst jawny jest dodawany modulo 2 na wyjściu funkcji szyfrującej – powstaje szyfrogram. Tryb ten jest bardzo korzystny ze względu na to, że postać każdego bloku na wejściu jest pseudolosowa, podczas szyfrowania tego samego tekstu tym samym kluczem, dzięki temu, że wartości liczników są inne, otrzymujemy inne wartości szyfrogramów. Zaletą jest również fakt, że zarówno operację szyfrowania jak i deszyfrowania można zrównoleglić, a błędy nie propagują się. Schematy blokowe prezentujące sposób szyfracji i deszyfracji z wykorzystaniem tej metody przedstawiono na rysunkach odpowiednio 2.9 oraz 2.10.



Rys. 2.9. Szyfracja z wykorzystaniem szyfrów blokowych w trybie CTR.



Rys. 2.10. Szyfracja z wykorzystaniem szyfrów blokowych w trybie CTR.

W kolejnym paragrafie niniejszego rozdziału przedstawiony zostanie jeden z najbezpieczniejszych algorytmów blokowych i symetrycznych stosowanych obecnie. Może on być wykorzystany w każdym z omówionych tu trybów.

2.3. Advanced Encryption Standard (AES)

26 maja 2002 roku, w odpowiedzi na niedostateczne bezpieczeństwo standardu DES [3], wprowadzono w życie nowy standard: AES [4]. Standard ten specyfikuje algorytm Rijndael [4], szyfr symetryczny blokowy o blokach danych 128 bitowych i długościach klucza 128, 192 lub 256 bitów. W algorytmie Rijndael można wykorzystywać również klucze o większej długości, lecz nie są one ujęte w standardzie AES. Wejściem i wyjściem dla algorytmu są sekwencje bitów zgrupowane w bloki 128 bitowej długości.

Operacje algorytmu AES odbywają się na dwuwymiarowych tablicach bajtów. Tablice te nazywane są stanami algorytmu. Tablica stanu algorytmu składa się z czterech wierszy, z których każdy zawiera *Nb* bajtów, gdzie *Nb* jest to długość bloku wejściowego podzielona przez 32. Ze względu na to, że wejście algorytmu jest 128 bitowe, to *Nb = 4*. Cztery bajty w każdej kolumnie tablicy stanu tworzą tablicę stanów z 32 bitowymi słowami. Może być ona interpretowana jako jednowymiarowa tablica 32 bitowych słów (kolumn). Długość klucza reprezentowana jest liczbą *Nk*, która może równać się 4, 6 albo 8, w zależności od wyboru długości klucza. Liczba ta określa liczbę 32 bitowych słów (liczbę kolumn) w kluczu szyfrującym.

Dla algorytmu AES, liczba rund wykonywanych cyklicznie, koniecznych do zaszyfrowania wiadomości zależy od długości wybranego klucza. Liczba rund może być przedstawiona za pomocą liczby *Nr*, która jest równa 10, gdy *Nk* = 4, 12 gdy *Nk* = 6 i 14 gdy *Nk* = 8. Zależności pomiędzy długością klucza, rozmiarem bloku i liczbą rund przedstawia tabela 2.1.

Tab. 2.1. Zależności pomiędzy długością klucza, rozmiarem bloku i liczbą rund w algorytmie AES.

	Długość klucza (*Nk* słów)	Długość bloku (*Nb* słów)	Liczba rund (*Nr* rund)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

W każdej rundzie algorytmu przeprowadzane są cztery transformacje, które zostały przedstawione w dalszej części tego paragrafu. Są to:

1. Podstawianie bajtów macierzy stanów;
2. Przesuwanie cykliczne wierszy macierzy stanu;
3. Przestawianie kolumn macierzy stanu według przekształceń na ciałach Galois;
4. Dodawanie (XOR) klucza rundy do stanu.

W celu opisania algorytmu posłużono się pseudokodem zaczerpniętym ze specyfikacji algorytmu AES [4]. Na potrzeby pseudokodu należy zdefiniować pseudofunkcje odpowiadające poszczególnym krokom każdej rundy. I tak, zamienianiu pojedynczych bajtów macierzy stanów odpowiada funkcja *SubBytes()*, przesuwaniu cyklicznemu – *ShiftRows()*, przestawianiu kolumn – *MixColumns()* i dodawaniu klucza rundy: *AddRoundKey()*.

W pseudokodzie algorytm AES przedstawić można w następujący sposób:

```
Cipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
      byte state[4,Nb]
      state = in
      AddRoundKey(state, w[0, Nb-1])
      for round = 1 step 1 to Nr-1
            SubBytes(state)
            ShiftRows(state)
            MixColumns(state)
            AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
      end for
      SubBytes(state)
      ShiftRows(state)
      AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])
      out = state
end
```

Transformacja *SubBytes()* zależy od bieżącego stanu algorytmu. Dla każdego wejściowego słowa obliczana jest jego wartość według poniższego równania:

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \tag{2.1}$$

b_0 - b_7 – bity bajtu wejściowego stanu

b'_0 - b'_7 – bity bajtu wyjściowego funkcji *SubBytes()*

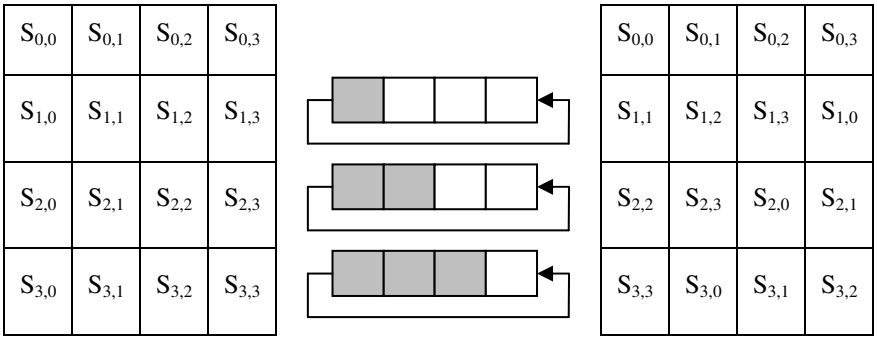
Prościej można przedstawić tą transformację, jako tzw. tablicę S-box, dla której bajtowi wejściowemu przyporządkowywany jest bajt według tabeli 2.2.

Tab. 2.2. Tablica S-box dla algorytmu AES

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Np. jeśli wejście ma postać {af}, to na wyjściu otrzyma się {79}.

Transformacja *ShiftRows()* polega na cyklicznej zamianie bajtów trzech ostatnich wierszy stanu algorytmu. Obrazuje to rysunek 2.11.



Rys. 2.11. Transformacja *Shift Rows* w AES

Pierwszy bajt drugiego wiersza tablicy stanu jest przesuwany na koniec drugiego wiersza, dwa pierwsze bajty trzeciego wiersza są przesuwane na jego koniec i analogicznie trzy pierwsze bajty czwartego wiersza są przesuwane na jego koniec. Pozostałe bajty „cofają się", tak jak na rysunku.

Transformacja *MixColumns()* wykonuje następującą operację mnożenia na tablicy stanu:

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \tag{2.2}$$

Przykładowo, dla pierwszego wiersza tablicy stanów, przeprowadzona jest następująca operacja:

$$S'_{0,c} = (\{02\} \bullet S_{0,c}) \oplus (\{03\} \bullet S_{1,c}) \oplus S_{2,c} \oplus S_{3,c} \tag{2.3}$$

Transformacja *AddRoundKey()* – podczas tej transformacji klucz rundy jest dodawany do stanu poprzez proste dodawanie XOR. każdy klucz rundy składa się z *Nb* słów. Dodawany klucz zmienia się z każdą rundą w sposób opisany poniżej (funkcja *KeyExpansion()*).

Funkcja *KeyExpansion()* – Algorytm AES za pomocą klucza szyfrującego K oblicza z każdą rundą rozszerzenie klucza. Funkcja *KeyExpansion()* generuje całkowitą liczbę *Nb(Nr+1)* słów. Algorytm wykorzystuje początkowy zestaw *Nb* słów i w każdej rundzie wykorzystuje się nowy zbiór *Nb* słów danych kluczy. Rozszerzenie wejściowego klucza przebiega w następujący sposób [4]:

```
KeyExpansion(byte key[4*Nk], word w[Nb*(Nr+1)], Nk)
begin
      word temp
      i = 0
      while (i < Nk)
            w[i] = word(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
            i = i+1
      end while
      i = Nk
      while (i < Nb * (Nr+1)]
            temp = w[i-1]
            if (i mod Nk = 0)
                  temp = SubWord(RotWord(temp)) xor Rcon[i/Nk]
            else if (Nk > 6 and i mod Nk = 4)
                  temp = SubWord(temp)
            end if
            w[i] = w[i-Nk] xor temp
            i = i + 1
      end while
end
```

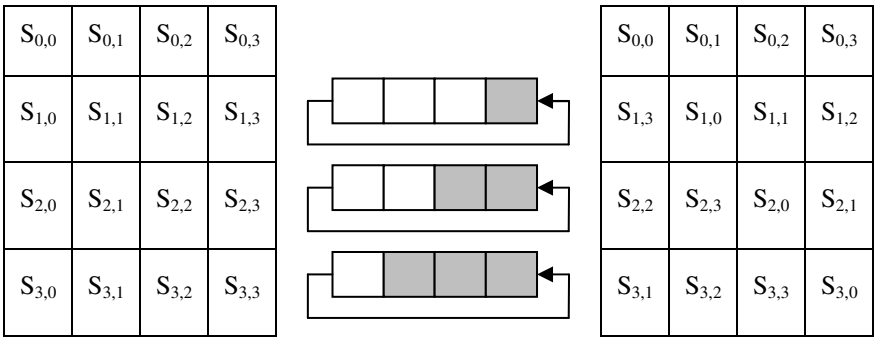
Funkcja *SubWord()* pobiera czterobajtowe słowo wejściowe i dokonuje na nim operacji jak funkcja *SubBytes()*. *RotWord()* dokonuje cyklicznego przesunięcia bitowego słów o jeden bajt w lewo. *Rcon[i/Nk]* to stała czterobajtowa tablica postaci $(\{02^{(i/Nk)-1}\},\{00\},\{00\},\{00\})$.

Deszyfrowanie AES polega na wykonaniu operacji szyfrowania w odwrotnej kolejności. Indywidualne transformacje używane przez deszyfrator AES to *InvShiftRows()*, *InvSubBytes()*, *InvMixColumns()* i *AddRoundKey()*. Deszyfrowanie można przedstawić za pomocą następującego pseudokodu [4]:

```
InvCipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
byte state[4,Nb]
      state = in
```

```
        AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])
        for round = Nr-1 step -1 downto 1
              InvShiftRows(state)
              InvSubBytes(state)
              AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
              InvMixColumns(state)
        end for
        InvShiftRows(state)
        InvSubBytes(state)
        AddRoundKey(state, w[0, Nb-1])
        out = state
end
```

Transformacja *InvShiftRows()* działa podobnie jak *ShiftRows()*, co zobrazowano na rysunku 2.12.



Rys. 2.12. Transformacja *Inverted Shift Rows* w AES

Transformacja *InvSubBytes()* wykonuje zadania odwrotne do *SubBytes()*. Tablica S-box ma tu postać przedstawioną w tabeli 2.3.

Tab. 2.3. Tablica S-box dla deszyfrowania AES

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Transformacja *InvMixColumns()* wykonuje zadania odwrotne do *MixColumns()*:

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \tag{2.4}$$

Przykładowo, dla pierwszego wiersza tablicy stanów, przeprowadzona jest następująca operacja:

$$S'_{0,c} = (\{0e\} \bullet S_{0,c}) \oplus (\{0b\} \bullet S_{1,c}) \oplus (\{0d\} \bullet S_{2,c}) \oplus (\{09\} \bullet S_{3,c}) \tag{2.5}$$

Zaletą algorytmu AES jest to, że nie posiada on słabych kluczy, nie ma żadnych ograniczeń co do wyboru klucza, byleby pozostał on tajny. Algorytm AES pozostał jak dotąd niezłamany. Istnieje możliwość przedstawienia szyfru w bardzo wygodny algebraicznie sposób. Stawia to jednak bezpieczeństwo algorytmu pod znakiem zapytania, ponieważ istnieje podejrzenie, że w przyszłości znajdzie się równie łatwa reprezentacja algebraiczna służącą złamaniu algorytmu. Jednak jest to niewystarczający argument i nie może on zachwiać bezpieczeństwem AES. Można bowiem stwierdzić, że w przyszłości każdy szyfr może zostać złamany. Na dzień dzisiejszy nie są znane skuteczne i szybkie mechanizmy pozwalające na wykorzystanie wygodnej algebraicznie formuły AES do złamania tego szyfru.

2.4. Szyfracja z kluczem publicznym – Algorytm RSA

W algorytmie RSA [6] stosowane są dwa typy kluczy – klucz publiczny RSA oraz klucz prywatny RSA. Razem oba te klucze tworzą tzw. parę kluczy RSA. Obecna specyfikacja RSA wspiera tzw. *multi-prime* RSA, który polega na tym, że dzielnik klucza może mieć ponad dwa dzielniki w postaci liczb pierwszych. Zaletą *multi-prime* RSA jest mniejsza złożoność obliczeniowa algorytmu – lepsza wydajność może być uzyskana na jednoprocesorowych platformach. Jednak zastosowanie tego rozwiązania wpływa na bezpieczeństwo RSA.

W podstawowej realizacji RSA klucz publiczny składa się z dwóch komponentów:

- *n* dzielnika RSA (liczba naturalna);
- *e* publicznego eksponentu RSA (liczba naturalna).

W prawidłowym kluczu RSA, dzielnik RSA *n* jest produktem mnożenia dwóch (lub większej liczby – *multi-prime*) liczb pierwszych *p* oraz *q*:

$$n = p \cdot q \tag{2.6}$$

Eksponent *e* jest liczbą naturalną z przedziału od 1 do *n-1*. Jest to względnie pierwsza liczba z funkcją Eulera $\varphi(n)$:

$$\varphi(n) = (p-1) \cdot (q-1) \tag{2.7}$$

Klucz prywatny RSA składa się z pary liczb *(n, d),* gdzie jej składniki mają następujące znaczenie:

- *n* dzielnik RSA (liczba naturalna);
- *d* prywatny eksponent RSA (liczba naturalna).

W prawidłowej reprezentacji klucza RSA wartość *n* jest taka sama jak przy kluczu publicznym RSA. Prywatny eksponent RSA jest liczbą naturalną mniejszą od *n* i spełniającą równanie:

$$e \cdot d \equiv 1 (\mathrm{mod}\, \varphi(n)) \tag{2.8}$$

Szyfracja w algorytmie RSA odbywa się z wykorzystaniem jawnej wiadomości *m* oraz klucza publicznego RSA *(n, e)*. Wiadomość, która podlega zaszyfrowaniu musi być z przedziału od 0 do *n-1*. Proces szyfracji przedstawia poniższa zależność:

$$c = m^e \bmod n \tag{2.9}$$

gdzie *c* oznacza tekst zaszyfrowany.

Deszyfracja z kolei odbywa się z wykorzystaniem zaszyfrowane wiadomości *c* oraz klucza tajnego RSA *(n, d)*. Wiadomość zaszyfrowana musi być z przedziału od 0 do *n-1*. Proces deszyfracji przedstawia poniższa zależność:

$$m = c^d \bmod n \tag{2.10}$$

Warto w tym momencie dodać, że algorytm RSA może być również stosowany dla realizacji uwierzytelniania. W takim przypadku pozwala on na wygenerowanie podpisu *s* wiadomości *m* (*m* z przedziału od 0 do *n-1*) lub jej skrótu (np. z wykorzystaniem algorytmu SHA-256) przy wykorzystaniu klucza prywatnego *(n, d)*:

$$s = m^d \bmod n \tag{2.11}$$

Weryfikacji tego podpisu (*s* z przedziału od 0 do *n-1*) dokonuje się z wykorzystaniem klucza publicznego *(n, e)*:

$$m = s^e \bmod n \tag{2.12}$$

W przypadku szyfracji nadawca jest pewny, że jego wiadomość odebrać może tylko osoba, która posiada klucz prywatny tworzący parę z kluczem publicznym użytym do szyfracji.

Dla uwierzytelniania z kolei odbiorca jest pewny, że odebrana przez niego wiadomość pochodzić może tylko od osoby, która posiada klucz prywatny związany z kluczem odbiorczym użytym do weryfikacji.

W idealnej zatem sytuacji, gdy komunikują się ze sobą dwie osoby, każda z nich posiada swój klucz prywatny, a klucze publiczne tych osób są ogólnie dostępne. W tym wypadku możliwa jest jednoczesna realizacja mechanizmów poufności oraz uwierzytelniania.

2.5. Funkcja skrótu – Algorytm SHA-256

Funkcja skrótu, jak już wcześniej wspomniano, jest wykorzystywana do podpisywania dokumentów oraz w algorytmach z kluczem niesymetrycznym. Ogólnie – funkcja, za pomocą konkretnego algorytmu (SHA [7] lub MD5 [8]), dokonuje skrótu wiadomości do ciągu znaków z góry ustalonej długości. Algorytm czyta dane ze źródła, którego ma być utworzony skrót, w specyficzny sposób przetwarza te dane, aby na wyjściu otrzymać pseudolosowy, z góry ustalonej długości kod. Ten kod to jakby odcisk palca danego dokumentu (danych).

Jednym z charakterystycznych właściwości algorytmu jest to, iż niewielka zmiana w danych wejściowych całkowicie zmienia postać danych wyjściowych (odcisku palca) algorytmu. Dlatego też funkcje te są tak przydatne do sprawdzania integralności. Algorytm SHA-1 daje zawsze na wyjściu 160 bitów, SHA-256 daje 256 bitów, a algorytm MD5 – 128 bitów.

Maksymalnym ciągiem wejściowym dla algorytmów SHA-1 oraz SHA-256 są wiadomości o długości 2^{64} bitów. Dla SHA-384 oraz SHA-512 ograniczenie to wynosi 2^{128} bitów. W dalszej części tego paragrafu przedstawiono zagadnienia związane z algorytmem SHA-256, ponieważ tylko ten algorytm wykorzystywany będzie w proponowanym systemie bezpieczeństwa. Dodatkowe informacje odnośnie pozostałych odmian algorytmu SHA można znaleźć w [7].

Poniżej został przedstawiony algorytm funkcji SHA-256 [7]:

a. Podstawowe funkcje i stałe wykorzystywane przez SHA-256

Algorytm ten używa sześciu funkcji logicznych, z której każda operuje na 32 bitowych słowach (x, y, z). Na wyjściu każdej funkcji otrzymuje się słowo 32 bitowe.

$$Ch(x, y, z) = (x \wedge y) \oplus (-x \wedge z) \tag{2.13}$$

$$Maj(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z) \tag{2.14}$$

$$\sum_0^{\{256\}}(x) = ROTR^2(x) \oplus ROTR^{13}(x) \oplus ROTR^{22}(x) \tag{2.15}$$

$$\sum_1^{\{256\}}(x) = ROTR^6(x) \oplus ROTR^{11}(x) \oplus ROTR^{25}(x) \tag{2.16}$$

$$\sigma_0^{\{256\}}(x) = ROTR^7(x) \oplus ROTR^{18}(x) \oplus SHR^3(x) \tag{2.17}$$

$$\sigma_1^{\{256\}}(x) = ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus SHR^{10}(x) \tag{2.18}$$

SHA-256 używa również sekwencji 64 stałych 32 bitowych słów $K_0^{\{256\}}$, $K_1^{\{256\}}$, … $K_{63}^{\{256\}}$. Zapisane od lewej do prawej przedstawiają się następująco:

```
428a2f98 71374491 b5c0fbcf e9b5dba5 3956c25b 59f111f1 923f82a4 ab1c5ed5
d807aa98 12835b01 243185be 550c7dc3 72be5d74 80deb1fe 9bdc06a7 c19bf174
e49b69c1 efbe4786 0fc19dc6 240ca1cc 2de92c6f 4a7484aa 5cb0a9dc 76f988da
983e5152 a831c66d b00327c8 bf597fc7 c6e00bf3 d5a79147 06ca6351 14292967
27b70a85 2e1b2138 4d2c6dfc 53380d13 650a7354 766a0abb 81c2c92e 92722c85
a2bfe8a1 a81a664b c24b8b70 c76c51a3 d192e819 d6990624 f40e3585 106aa070
19a4c116 1e376c08 2748774c 34b0bcb5 391c0cb3 4ed8aa4a 5b9cca4f 682e6ff3
748f82ee 78a5636f 84c87814 8cc70208 90befffa a4506ceb bef9a3f7 c67178f2
```

Prezentowany algorytm wykorzystuje funkcję $ROTL^n(x)$ zdefiniowaną jako przesunięcie bitowe cykliczne słowa x o n bitów w lewo oraz $ROTR^n(x)$ – cykliczne przesunięcie bitowe w prawo o n bitów.

Dodatkowo, używana jest też funkcja $SHR^n(x)$ zdefiniowana jako przesunięcie bitowe słowa x o n bitów w prawo.

b. Przetwarzanie wstępne

Zanim algorytm rozpocznie obliczanie funkcji skrótu danej wiadomości, należy tą wiadomość odpowiednio przygotować. Po pierwsze, dla potrzeb algorytmu SHA-256 długość wiadomości musi być wielokrotnością 512 bitów.

W celu przygotowania odpowiedniej długości wiadomości, przeprowadzane są przedstawione poniżej operacje.

Należy założyć, że długość wiadomości *M* jest równa *l* bitów. Do końca wiadomości należy dopisać bit 1 a następnie *k* zerowych bitów, tak aby *k* było najmniejszą liczbą naturalną spełniającą równanie:

$$(l + 1 + k) \bmod 512 = 448 \tag{2.19}$$

Na końcu tak powstałej wiadomości, należy dopisać długość wiadomości w bitach, wyrażoną poprzez 64 bitową sekwencję.

Dzięki tej operacji uzyskuje się wiadomości o odpowiedniej długości.

Kolejnym krokiem przetwarzania wstępnego jest dzielenie wiadomości do bloków określonej długości. Dla algorytmu SHA-256 poprzednio sformatowana wiadomość jest dzielona na *N* 512 bitowych bloków $M^{(1)}, M^{(2)}, \ldots M^{(N)}$. Dzięki temu, że 512 bitów wejściowego bloku może być wyrażone za pomocą szesnastu 32 bitowych słów, pierwsze 32 bity *i*-tego bloku wiadomości jest oznaczone jako $M_0^{(i)}$, następne 32 bity jako $M_1^{(i)}$, aż do ostatnich 32 bitów: $M_{15}^{(i)}$.

Ostatnim krokiem przetwarzania wstępnego jest ustalenie początkowych wartości funkcji haszującej $H^{(0)}$. Dla SHA-256 $H^{(0)}$ składa się z następujących ośmiu 32 bitowych słów:

$$H_0^{(0)} = \texttt{6a09e667}$$
$$H_1^{(0)} = \texttt{bb67ae85}$$
$$H_2^{(0)} = \texttt{3c6ef372}$$
$$H_3^{(0)} = \texttt{a54ff53a}$$
$$H_4^{(0)} = \texttt{510e527f}$$
$$H_5^{(0)} = \texttt{9b05688c}$$
$$H_6^{(0)} = \texttt{1f83d9ab}$$
$$H_7^{(0)} = \texttt{5be0cd19}$$

Na tym kończy się proces przetwarzania wstępnego.

c. obliczanie SHA-256 (przedstawione za pomocą pseudokodu):

Pętla od $i = 1$ do N:

{

1. Przygotowanie harmonogramu wiadomości (*message schedule*) $\{W_t\}$

$$W_t = \begin{cases} M_t^{(i)} & 0 \le t \le 15 \\ \sigma_1^{\{256\}}(W_{t-2}) + W_{t-7} + \sigma_0^{\{256\}}(W_{t-15}) + W_{t-16} & 16 \le t \le 63 \end{cases}$$

2. Zainicjowanie ośmiu roboczych wartości a, b, c, d, e, f, g ,h, wartościami *i-1* funkcji haszującej:

$a = H_0^{(i-1)}$

$b = H_1^{(i-1)}$

$c = H_2^{(i-1)}$

$d = H_3^{(i-1)}$

$e = H_4^{(i-1)}$

$f = H_5^{(i-1)}$

$g = H_6^{(i-1)}$

$h = H_7^{(i-1)}$

3. Pętla od $t = 0$ do 63

{

$$T_1 = h + \sum\nolimits_1^{\{256\}}(e) + Ch(e, f, g) + K_t^{\{256\}} + W_t$$

$$T_2 = \sum\nolimits_0^{\{256\}}(a) + Maj(a, b, c)$$

$h = g$

$g = f$

$f = e$

$e = d + T_1$

$d = c$

$c = b$

$b = a$

$a = T_1 + T_2$

}

4. Obliczenie i-tej wartości funkcji haszującej $H^{(i)}$

$H_0^{(i)} = a + H_0^{(i-1)}$

$H_1^{(i)} = b + H_1^{(i-1)}$

$H_2^{(i)} = c + H_2^{(i-1)}$

$H_3^{(i)} = d + H_3^{(i-1)}$

$H_4^{(i)} = e + H_4^{(i-1)}$

$H_5^{(i)} = f + H_5^{(i-1)}$

$H_6^{(i)} = g + H_6^{(i-1)}$

$H_7^{(i)} = h + H_7^{(i-1)}$

}

Po wykonaniu *N* powyższych operacji (do zakończenia wiadomości), wyjściową 256 bitową wartością funkcji haszującej SHA-256 jest:

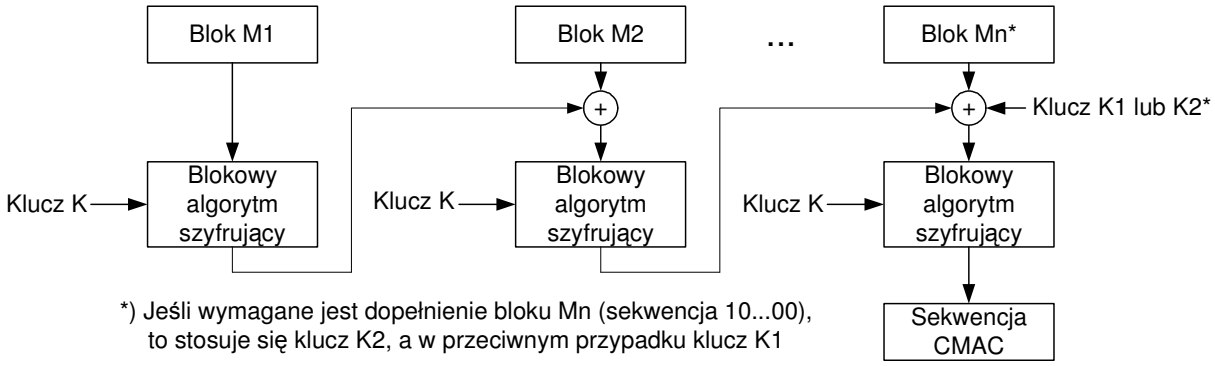
$$H_0^{(N)} || H_1^{(N)} || H_2^{(N)} || H_3^{(N)} || H_4^{(N)} || H_5^{(N)} || H_6^{(N)} || H_7^{(N)} \quad (2.20)$$

2.6. Algorytm CMAC

Algorytm CMAC (*Cypher-based MAC*) pozwala na realizację mechanizmu integralności oraz uwierzytelniania z wykorzystaniem kodów uwierzytelniających MAC. Zgodnie z jego specyfikacją [9] bazuje on na blokowych szyfrach symetrycznych (np. AES). Może być on rozumiany jako pewien dodatkowy tryb pracy algorytmów blokowych (patrz punkt 2.2).

CMAC stanowi silne zabezpieczenie przed przypadkowym czy celowym naruszeniem integralności wiadomości. Może również stanowić mechanizm dla realizacji uwierzytelniania wiadomości.

Sposób generacji kodu CMAC przedstawiono na rysunku 2.13.



Rys. 2.13. Algorytm CMAC.

W pierwszym etapie wyznaczania sekwencji CMAC należy podzielić całą wiadomość *M* na *n* bloków tej samej długości. Jeśli jest to konieczne, to ostatnią sekwencję należy dopełnić ciągiem postaci *10…00*. W przypadku, gdy dopełnienie miało miejsce wykorzystuje się *Klucz K2*, a gdy długość wiadomość jest całkowitą wielokrotnością długości bloku *Mi* stosuje się *Klucz K1*.

Długość pojedynczego bloku *Mi* (*i = 1, 2, …, n*) zależy od zastosowanego szyfru blokowego. W przypadku algorytmu AES-128 sekwencje: *Mi, Klucz K, Klucz K1, Klucz K2 oraz sekwencja CMAC* mają po 128 bitów.

Procedura generacji kluczy *K1* oraz *K2* jest następująca [9]:

1. Niech *L* będzie wynikiem szyfracji (kluczem *K* i danym szyfrem blokowym) pojedynczego bloku składającego się z samych zer.
2. Jeśli najbardziej znaczącym bitem *L* będzie 0, to $K1 = L << 1$, a w przeciwnym wypadku $K1 = (L << 1) \oplus Rb$, gdzie *Rb* dla np. szyfrów 128-bitowych to sekwencja 128-bitowa postaci 00…0010000111.
3. Jeśli najbardziej znaczącym bitem *K1* będzie 0, to $K2 = K1 << 1$, a w przeciwnym wypadku $K2 = (K1 << 1) \oplus Rb$.

Sekwencja CMAC jest dołączana przez nadawcę do przesyłanej wiadomości.

Weryfikacja sekwencji CMAC polega na jej ponownym wyznaczeniu przez odbiorcę z wykorzystaniem tajnego klucza *K* i porównaniem wyniku z sekwencją zawartą w odebranej wiadomości. Pozytywna weryfikacja oznacza, że wiadomość jest integralna i autentyczna, a w ogólności oznacza, że sekwencja CMAC zawarta w odebranej wiadomości została wyznaczona na podstawie klucza *K* oraz tej właśnie wiadomości.

3. Specyfika transmisji w krótkofalowym kanale radiowym

Rozdział ten pełni rolę informacyjną i ma na celu prezentację danych na temat kanału krótkofalowego, a w szczególności zjawisk w nim występujących oraz trudności w jego wykorzystaniu w przypadku łączności radiowej. Ta część pracy opiera się głównie na opracowaniu [10].

Prezentowane tu rozważania dotyczą fal krótkich obejmujących zakres częstotliwości od 3 do 30 MHz (długość fali elektromagnetycznej λ = 100 – 10 m). Ponieważ na propagację fal w paśmie HF wpływa głównie warstwa atmosfery zwana jonosferą, na początku tego rozdziału zostanie omówiony jej wpływ na propagację fal.

3.1. Wpływ jonosfery na propagację fal elektromagnetycznych

Jonosfera to zjonizowana część atmosfery, znajdująca się na wysokości powyżej 60 km. Wyniki pomiarów wskazują no to, że do wysokości 90 km atmosfera posiada taki sam skład, jak w pobliżu powierzchni Ziemi. Na większych wysokościach różnice mas gazów w atmosferze powodują jej rozwarstwienie: cięższe z nich gromadzą się w warstwach położonych niżej. W rozrzedzonej atmosferze, pod wpływem promieniowania słonecznego, zachodzi dysocjacja tlenu i azotu. Dysocjacja tlenu rozpoczyna się na wysokości około 90 km, a dysocjacja azotu na wysokościach powyżej 220 km.

Temperatura, wraz z wzrostem wysokości, zmienia się następująco: w troposferze temperatura spada wraz z wysokością średnio o sześć stopni na kilometr. W pewnych przypadkach, na skutek działania lokalnych czynników, temperatura w tej strefie może wzrastać wraz z wysokością. Jest to tzw. inwersja temperatur. Inwersja może powstać na skutek poziomych ruchów mas powietrza (inwersja adwekcyjna), obserwowana wczesną wiosną, gdy nad powłoką śnieżną pokrywającą ziemię unoszą się nadchodzące z południa masy ciepłego powietrza. Innym typem inwersji jest przesuwanie się nagrzanego powietrza znad lądu nad chłodniejszą powierzchnię morza. Po przekroczeniu górnej granicy troposfery (10 km w strefach podbiegunowych, 12 km szerokościach umiarkowanych i 18 km w okolicach równikowych), spadek temperatury ustaje. Temperatura utrzymuje się na poziomie 219 K do około 20 km. Później temperatura powietrza zaczyna wzrastać i osiąga maksimum około 400 K na wysokości około 60 km. Kolejne minimum występuje na wysokości około 80 km i ma wartość 200-250 K. Następnie temperatura zaczyna ponownie wzrastać do wartości rzędu tysiąca kelwinów i więcej.

W jonosferze występuje jonizacja, czyli usuwanie jednego lub (rzadziej) kilku elektronów z atomów gazu wchodzącego w skład atmosfery. Jonizacja danego gazu zachodzi pod wpływem promieniowania o częstotliwości przewyższającej pewną wartość krytyczną, zwaną częstotliwością jonizacji. Podstawowym źródłem jonizacji atmosfery jest Słońce. Fotosfera o temperaturze około 6000 K promieniuje fale radiowe o bardzo szerokim widmie częstotliwości. Chromosfera i korona o temperaturze rzędu $2 \cdot 10^6$ K są źródłami promieniowania ultrafioletowego i miękkiego rentgenowskiego. Ponadto Słonce wyrzuca strumienie elektronów i innych cząstek tworzących promieniowanie korpuskularne. Innymi czynnikami jonizującymi są: promieniowanie gwiazd, promieniowanie kosmiczne, pył kosmiczny i meteory.

Wraz z procesem powstawania jonów i elektronów swobodnych zachodzi w atmosferze proces odwrotny, polegający na ponownym łączeniu się swobodnych elektronów z jonami, czyli tzw. proces rekombinacji. Przy rekombinacji wydziela się ilość energii równa uprzednio włożonej pracy jonizacji. Prawdopodobieństwo rekombinacji jest tym większe, im większa jest gęstość elektronowa (liczba elektronów w jednostce objętości gazu), czyli im bardziej

jonizacja jest intensywna. Przy określonych warunkach wytwarza się stan równowagi dynamicznej pomiędzy jonizacją i rekombinacją. W rzeczywistości warunki jonizacji ulegają ciągłym zmianom – gęstość elektronowa podlega ciągłej fluktuacji. W godzinach porannych i przedpołudniowych przeważa proces jonizacji i gęstość elektronowa wzrasta. Po południu zaczyna stopniowo dominować rekombinacja, która w nocy znacznie postępuje. Ostateczne górne warstwy atmosfery utrzymują się w ciągłym stanie jonizacji. Poza zmianami dobowymi występują zmiany sezonowe oraz długookresowe związane z cykliczną zmiennością aktywności słonecznej.

Wskutek tych czynników rozkład gęstości elektronowej jonosfery jest nierównomierny. W ciągu dnia wyróżnia się cztery obszary jonosfery a w ciągu nocy dwa. Przedstawia to tabela 3.1.

Tab. 3.1. Obszary jonosfery.

Parametr	**Obszar D**	**Obszar E**	**Obszar F_1**	**Obszar F_2**
wysokość	60-90 km w ciągu dnia w nocy obszar D znika	100-120 km	180-240 km w ciągu dnia w nocy obszar F_1 znika	300-400 km w lecie 230-350 km w zimie
mechanizm jonizacji	jonizacja NO promieniowaniem linii L_α; jonizacja wszystkich gazów miękkim promieniowaniem rentgenowskim	jonizacja wszystkich gazów miękkich promieniowaniem rentgenowskim	jonizacja O przy szybkim zmniejszaniu się współczynnika rekombinacji z wysokością	jonizacja O ultrafioletowym, rentgenowskim i prawdopodobnie korpuskularnym promieniowaniem

Obszar F_1 występuje tylko w porze letniej. W nocy gęstość elektronowa dwóch obszarów, które nie znikają wyraźnie maleje. Obszary D, E i F_1 charakteryzują się dużą stabilnością, tzn. dobowe zmiany gęstości elektronowej i wysokości maksimów powtarzają się z dnia na dzień. Obszar F_2 jest niestabilny. Gęstość elektronowa i wysokość maksimum w tym obszarze ulegają znacznym zmianom z dnia na dzień.

Od czasu do czasu na wysokości obszaru E pojawia się warstwa silnie zjonizowana, o nazwie warstwy sporadycznej E i oznaczeniu E_s. Warstwa ta może powstać w dowolnej porze doby i roku. W średnich szerokościach geograficznych najczęściej pojawia się w ciągu dnia w porze letniej.

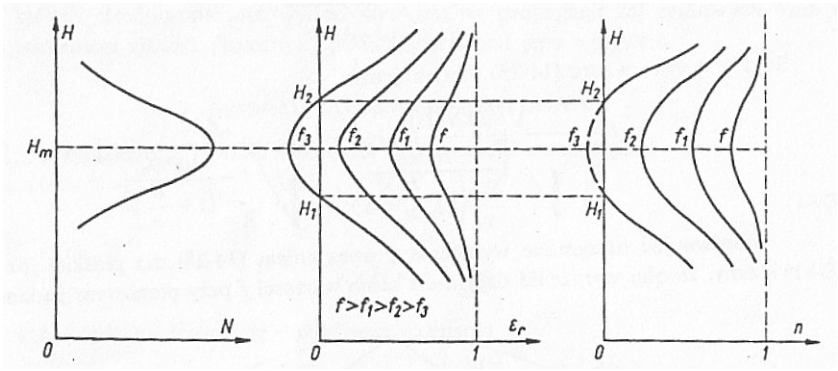
W jonosferze mają miejsce ruchy związane z przypływami i odpływami, wywoływanymi oddziaływaniem Księżyca i Słońca. Wpływ mają nie tylko siły grawitacyjne, ale również wpływy cieplne. Z przypływami i odpływami związane są wiatry jonosferyczne. Powstają również wiry (turbulencje) powodujące lokalne niejednorodności jonosfery. Są one powodem rozpraszania fal elektromagnetycznych rozchodzących się w jonosferze.

W pewnych okresach stan zjonizowania jonosfery ulega silnym zaburzeniom, związanym z występowaniem zorzy polarnej i zaburzeń w polu geomagnetycznym. Są to tzw. burze jonosferyczne. Podczas nich gęstość elektronowa warstwy F_2 maleje tak bardzo, że praktycznie warstwa ta przestaje istnieć.

Atmosfera ziemska jest nieustannie bombardowana przez liczne meteory. Ich tory w atmosferze są liniami prostymi o różnych kierunkach. Meteory wchodząc z ogromna prędkością w atmosferę rozgrzewają się i ulegają wyparowaniu, pozostawiając za sobą silnie zjonizowany ślad o średniej długości 25 km i średnicy w początkowej fazie rzędu kilku centymetrów.

Najbardziej zewnętrzną część jonosfery tworzą naturalne obszary promieniowania, czyli tzw. strefy Van Allena. Są to obszary otaczające kulę ziemską mające konfigurację linii sił pola geomagnetycznego, wypełnionymi cząstkami o dużej energii kinetycznej. Cząstki te wykonują skomplikowane ruchy wzdłuż i wokół geomagnetycznych linii sił.

Dopóki współczynnik załamania jest rzeczywisty, propagacja fali w jonosferze nie będzie się zasadniczo różniła od propagacji fali w troposferze. Przy zmniejszaniu częstotliwości promieniowania, przenikalność elektryczna, a więc i współczynnik załamania w obrębie warstwy jonosferyczne ulegają zmniejszeniu. Przy dalszym zmniejszaniu częstotliwości sygnału, na pewnej wysokości (H_1) przenikalność elektryczna stanie się równa zeru, a w przedziale wysokości H_1-H_2 będzie ujemna. Współczynnik załamania w tym obszarze wysokości będzie więc urojony. oznacza to, że w zakresie wysokości H_1-H_2 określone częstotliwości nie mogą się rozchodzić. Fale o tych częstotliwościach muszą ulec odbiciu na wysokości H_1. Ilustruje to następujący rysunek 3.1.



Rys. 3.1. Odbicie fali w propagacji jonosferycznej [10].

Dla wysokości większych od H_2 przenikalność elektryczna jonosfery przyjmuje znów wartości dodatnie i współczynnik staje się wartością rzeczywistą tak, że propagacja fali o częstotliwości f_3 znów jest możliwa. Jednak fala nie może przeniknąć do tego obszaru, ze względu na obecność bariery na w przedziale wysokości H_1-H_2. Fala radiowa o częstotliwości f_3 odbija się od obszaru H_1-H_2. Pionowo wypromieniowany sygnał radiowy o częstotliwości f ulega odbiciu na takiej wysokości, na której przenikalność elektryczna (oraz współczynnik załamania) jonosfery staje się równy zeru.

Jeżeli będzie zwiększana częstotliwość sygnału, to zjawisko odbicia będzie następowało na coraz większej wysokości. Maksymalną częstotliwość, przy której pionowo wypromieniowana fala radiowa ulega odbiciu od warstwy jonosferycznej, nazywamy częstotliwością krytyczną. Przy częstotliwościach większych od częstotliwości krytycznej zjawisko odbicia nie występuje – warstwa jonosfery staje się przezroczysta dla fal radiowych.

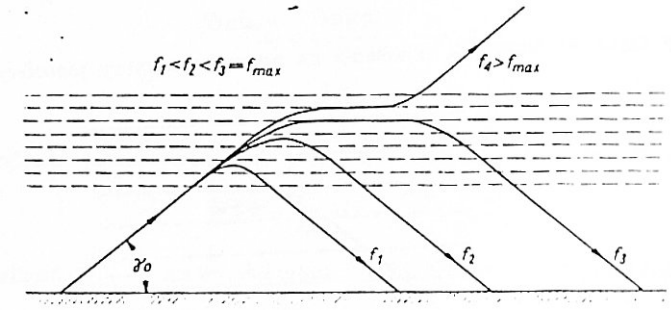
Rozważając zagadnienie załamania fal radiowych w płaskiej warstwowej jonosferze, można stwierdzić, że: fala o częstotliwości f padając na jonosferę pod kątem Θ_0 i fala o częstotliwości f_0 wypromieniowana pionowo odbija się na tej samej wysokości, oraz że przy ustalonym kącie padania Θ_0 odbicie fali następuje na tym większej wysokości, im większa jest jej częstotliwość.

$$f = f_0 \sec\Theta_0 \tag{3.1}$$

Zwiększając częstotliwość fali można dojść do obszaru, w którym $f_0 = f_{kr}$:

$$f_{\max}(\Theta_0) = f_{kr} \sec\Theta_0 \tag{3.2}$$

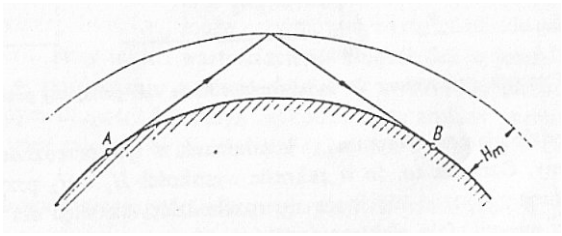
Częstotliwość ta jest nazwana częstotliwością maksymalną. Jest ona funkcją padania Θ_0 i częstotliwości mniejszej od częstotliwości maksymalnej.



Rys. 3.2. Załamanie fali radiowej w płaskiej, warstwowej jonosferze [10].

Przy ustalonym kącie padania Θ_0 i częstotliwości mniejszej od częstotliwości maksymalnej, odległość punktu padania promienia odbitego od punktu nadawania wzrasta wraz ze wzrostem częstotliwości sygnału. Gdy $f > f_{max}$ fala wchodzi w obszar ujemnego gradientu gęstości elektronowej i jej trajektoria zostaje odchylona ku górze (patrz rysunek 3.2).

Podobnie można określić padanie fali do określania częstotliwości maksymalnej w kulisto-warstwowej jonosferze (patrz rysunek 3.3).



Rys. 3.3. Załamanie fali w kulisto-warstwowej jonosferze [10].

Rozważając wpływ pola magnetycznego na propagację fal w jonosferze można stwierdzić, że pod wpływem pola magnetycznego Ziemi, jonosfera staje się ośrodkiem dwójłomnym, czyli takim, w którym fala radiowa przy przechodzeniu przez ten ośrodek ulega (w ogólnym przypadku) rozszczepieniu na dwie fale. Jonosfera, wskutek działania pola magnetycznego Ziemi, staje się ośrodkiem anizotropowym. Na skutek anizotropowości, występuje skręcenie płaszczyzny polaryzacji fal.

Rozchodzący się w przestrzeni sygnał radiowy składa się z wielu harmonicznych składowych – jest on więc nieskończonym zbiorem fal monochromatycznych. Jeśli ośrodek, w którym rozchodzą się fale elektromagnetyczne, jest ośrodkiem niedyspersyjnym, to wszystkie fale monochromatyczne rozchodzą się z jednakową prędkością fazową (cały zbiór fal będzie się rozchodził z jednakową prędkością – prędkość grupowa będzie równa prędkości fazowej). Inaczej jest w przypadku ośrodka dyspersyjnego, takiego jak jonosfera. Prędkość fazowa jest tutaj funkcją częstotliwości, więc każda fala monochromatyczna, z której składa się impuls radiowy, rozchodzi się z inną prędkością fazową. Prędkość grupowa nie jest równa prędkości fazowej. Prędkość grupowa jest zawsze mniejsza od prędkości światła dla dyspersji normalnej.

Fala radiowa, przechodząc przez jonosferę, ulega tłumieniu wskutek strat spowodowanych przez zderzenia elektronów z jonami i neutralnymi cząstkami gazu. Jest to tzw. absorpcja jonosferyczna. Istnieją dwa rodzaje absorpcji: niedewiacyjna (fala przechodzi przez warstwę jonosferyczną, nie ulegając w niej znaczniejszej refrakcji – na przykład w warstwie D, gdy fala odbija się od warstwy E) oraz dewiacyjna (gdy współczynnik refrakcji jest znacznie

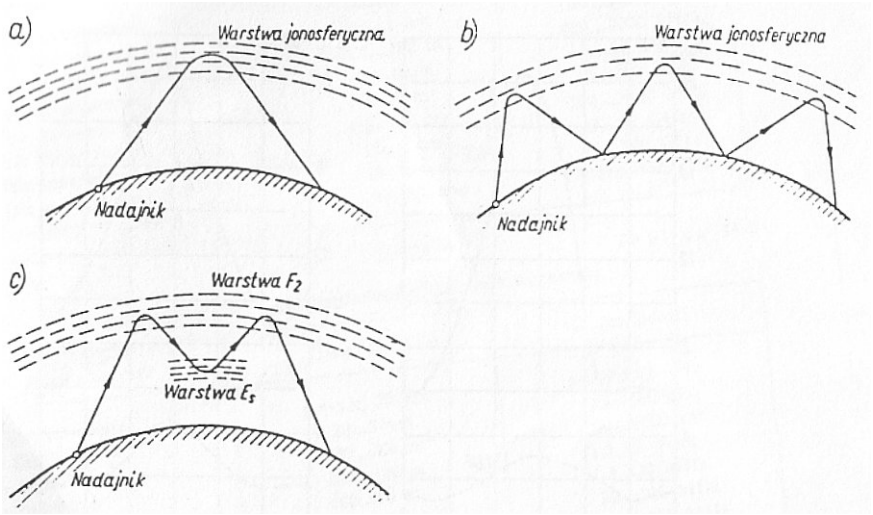
mniejszy od jedności i fala ulega silnemu załamaniu – mamy do czynienia wtedy, gdy fala ulega odbiciu od danej warstwy lub warunki są zbliżone do warunków, w których zachodzi pełna refrakcja). Fala radiowa doznaje największego tłumienia w dolnej warstwie jonosfery, zarówno przy wejściu do niej, jak i wyjściu. Absorpcja niedewiacyjna maleje w przybliżeniu proporcjonalnie do kwadratu częstotliwości.

Absorpcja dewiacyjna ma na ogół mniejsze znaczenie w propagacji jonosferycznej, ponieważ zachodzi na stosunkowo krótkiej drodze, w porównaniu z absorpcją niedewiacyjną. Oprócz normalnej absorpcji dewiacyjnej i niedewiacyjnej, występuje jeszcze czasem absorpcja anomalna związana z nadmierną aktywnością słoneczną.

3.2. Rozchodzenie się fal krótkich

Zasięg fali powierzchniowej (fala rozchodząca się przy powierzchni Ziemi) w zakresie fal krótkich na lądzie jest znikomo mały, ze względu na znaczne tłumienie wnoszone przez powierzchniowe warstwy gruntu oraz ze względu na krzywiznę Ziemi. Maksymalnymi odległościami, dla których możliwy jest odbiór fal krótkich za pośrednictwem fali powierzchniowej, to około kilkadziesiąt kilometrów dla fal 100 m oraz kilka kilometrów dla fal 10 m. Na fali jonosferycznej odbiór na falach krótkich jest możliwy na bardzo dużych odległościach. Z tego względu fale krótkie są głównie wykorzystywane dla celów dalekosiężnej radiokomunikacji o zasięgu ogólnoświatowym.

Sposób rozchodzenia się krótkofalowych fal radiowych za pośrednictwem jonosfery najlepiej przedstawia rysunek 3.4.



Rys. 3.4. Sposoby rozchodzenia się fali jonosferycznej w zakresie fal krótkich [10]

W przypadku a) jest przedstawione jednorazowe odbicie od jonosfery, czyli tzw. transmisji jednoskokowej. Fale mogą się również wielokrotnie odbijać od jonosfery i od ziemi, co przedstawia przypadek b) – jest to transmisja wieloskokowa. Dzięki niej fale krótkie mają bardzo duży zasięg (kilkanaście tysięcy kilometrów). Fale krótkie odbijają się głównie w warstwie F_2, a w pewnych okresach czasu również w warstwie E, E_s i F_1. W przypadku c) przedstawiona jest transmisja typu M, gdzie fala radiowa odbija się najpierw od warstwy F_2, a później od warstwy E_s.

Wszystkie częstotliwości krytyczne i maksymalne dla jonosfery leżą w zakresie fal krótkich. W związku z tym nie można stosować dowolnej częstotliwości dla zapewnienia łączności między dwoma punktami w określonym przedziale czasowym.

Największą częstotliwością, przy której można jeszcze nawiązać łączność między dwoma, punktami jest tzw. maksymalna częstotliwość użytkowa MUF (*Maximum Usable Frequency*). Jak już wcześniej wspomniano, tłumienie w jonosferze zachodzi głównie w dolnych jej warstwach (D i E). Z fal odbijanych przez warstwę F_2, najmniejszemu tłumieniu podczas przenikania przez warstwę E podlegają fale o częstotliwości bliskiej MUF. Ze wzrostem długości fali tłumienie zwiększa się. Zakres częstotliwości użytkowych jest wiec ograniczony od góry przez MUF, a od dołu przez duże tłumienie wnoszone przez niższe warstwy jonosfery. Najmniejszą częstotliwością, która, z uwagi na tłumienie może, być jeszcze użyta do zapewnienia transmisji na daną odległość jest tzw. najmniejsza częstotliwość użytkowa LUF (*Lowest Usable Frequency*).

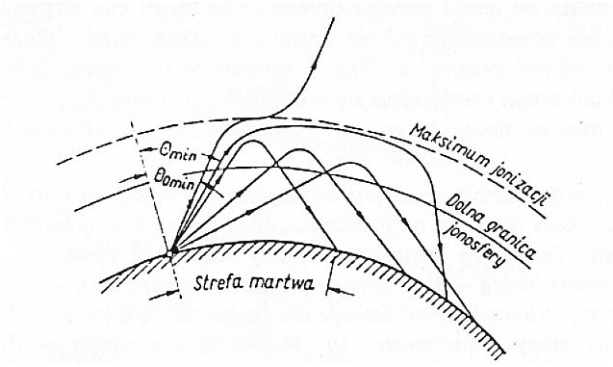
Gdy częstotliwość jest większa od MUF, fala przenika przez warstwę F_2 i nawet bardzo duże zwiększenie mocy nadajnika nie doprowadzi do nawiązania połączenia. Gdy częstotliwość jest bliska LUF, tłumienie można skompensować poprzez zwiększenie mocy nadajnika.

Najbardziej dogodną częstotliwością do transmisji, jest częstotliwość nieco mniejsza od MUF, ponieważ podlega ona małemu tłumieniu. Nie warto zbytnio zbliżać się do MUF, gdyż w razie chwilowego zmniejszenia MUF może nastąpić zanik całkowity (zanik jonizacji granicznej). Przyjęto, że optymalna częstotliwość robocza oznaczona międzynarodowo skrótem FOT (*Frequénce Optimum de Traffic*) wynosi:

$$\mathrm{FOT} = 0{,}85\ \mathrm{MUF} \tag{3.3}$$

Przyjęty piętnastoprocentowy margines w stosunku do MUF wystarcza do zapewnienia niewielkiego tłumienia w warstwie E i jednoczesnego niewielkiego prawdopodobieństwa zaników jonizacji granicznej.

Bardzo charakterystycznym zjawiskiem, który towarzyszy propagacji fal krótkich, jest występowanie tzw. stref martwych. Zilustrowane jest to na rysunku 3.5.



Rys. 3.5. Powstawanie strefy martwej przy rozchodzeniu się fal krótkich [10].

Załóżmy, że fala o częstotliwości f pada na jonosferę pod dostatecznie dużym kątem Θ, takim że zachodzi odbicie fali. Jeśli teraz, przy tej samej częstotliwości, będzie zmniejszać się kąt padania fali, to zgodnie z prawem secansa, wysokość H punktu odbicia będzie się zmieniała tak, aby było spełnione równanie:

$$f = f_0(H)\sec\Theta \tag{3.4}$$

Przy zmniejszaniu kąta padania, odbicie będzie się odbywać na coraz większej wysokości, a promień odbity początkowo będzie padać na ziemię w coraz mniejszej odległości od nadajni-

ka. Jednak zwiększanie wysokości punktu odbicia towarzyszy wzrost promienia krzywizny trajektorii. Począwszy od pewnej wartości kąta padania $\Theta = \Theta_{min}$, zmniejszanie się tego kąta spowoduje oddalanie się punktu padania promienia odbitego na powierzchnię Ziemi. Będzie się tak działo aż do wartości $\Theta = \Theta_{0min}$, przy której ustalona częstotliwość fali stanie się częstotliwością maksymalną jonosfery. Dalsze zmniejszanie kąta spowoduje przenikanie fali przez jonosferę.

Najmniejszą odległością od nadajnika, na którą zbliża się punkt padania promienia odbitego jest tzw. uskok. Zasięg fali powierzchniowej nie przekracza kilkudziesięciu kilometrów, a wartość uskoku wynosi średnio kilkaset kilometrów lub więcej, istnieje tzw. strefa martwa, która rozciąga się w odległości od kilkudziesięciu do kilkuset i więcej kilometrów od nadajnika, w której natężenie pola elektromagnetycznego pochodzącego od nadajnika jest praktycznie równe zeru.

Przy wypromieniowaniu poziomym, promień odbity od warstwy F_2 pada w odległości około 4000 km od nadajnika. Przy odbiciu od warstwy F_2 odległość ta wynosi około 2000 km. Odległości te odpowiadają transmisji jednoskokowej i są przyjmowane jako odległości odniesienia i nazwane odpowiednio trasą długą i trasą krótką. Mapy jonosferyczne są opracowywane dla wymienionych odległości.

Podczas dnia, jak już wcześniej wspomniano, zmieniają się właściwości jonosfery, przez co zmieniają się również warunki propagacyjne fal krótkich. O warunkach propagacyjnych decydują pora doby, pora roku oraz faza jedenastoletniego cyklu aktywności słonecznej. Warunki propagacyjne decydują o wartości LUF i MUF oraz o poziomie natężenia sygnału w miejscu odbioru. Poza anormalnymi warunkami propagacji:

- wartości MUF są większe w dzień niż w nocy
- wartości MUF w nocy są większe w lecie niż w zimie
- wartości MUF w dzień dla warstwy F_2 są większe w zimie niż w lecie; dla innych warstw MUF w dzień jest większa w lecie niż w zimie
- wartości MUF są większe w okresach silnej aktywności słonecznej

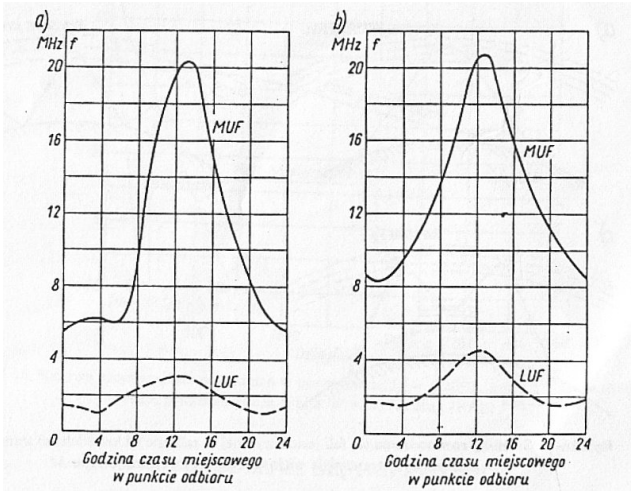
Na trasach krótkich MUF zwiększa się w godzinach przedpołudniowych czasu miejscowego. W zimie MUF dla warstwy F_2 charakteryzuje się wysokim maksimum i niskim minimum oraz znaczną stromością w okresach przejściowych. W lecie przebieg MUF jest bardziej wyrównany. Krzywe LUF na trasach krótkich wykazują maksimum w godzinach południowych i spadają poniżej zakresu fal krótkich w nocy. Na trasach długich krzywe LUF są nieregularne. Przykładowe przebiegi przedstawia rysunek 3.6.

Oprócz zwykłych warunków propagacji, występują również szczególne przypadki propagacji fal krótkich. Zaliczają się do nich:

- Odbicia od warstw sporadycznych (pokazane wcześniej przy omawianiu propagacji jonosferycznej) – odbicia zachodzą równocześnie od warstwy F_2 lub F_1 i od warstwy E_s. Prawdopodobieństwo pojawienia się warstwy E_s jest tak duże, że propagacja dalekosiężna dla częstotliwości poniżej 15 MHz odbywa się za pośrednictwem tej warstwy sporadycznej przez 25-50% czasu. Odbicia sporadyczne rozszerzają zakres częstotliwości możliwych do wykorzystania w propagacji fal krótkich.
- Propagacja pozaortodromowa – fale krótkie rozchodzą się w normalnych warunkach po ortodromach, czyli po trajektorii leżącej w płaszczyźnie wielkiego koła ziemskiego. W praktyce często obserwuje się odchylenia rzeczywistej drogi fali od ortodromy.

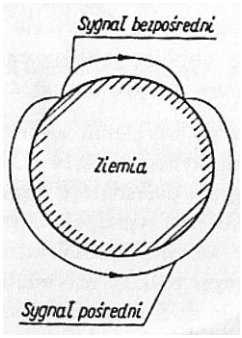
Dzieje się tak na skutek falistości jonosfery oraz wskutek odbić od nachylonych względem poziomu terenów w miejscu odbicia fali od ziemi.

- Zaburzenia jonosferyczne – wynikające z aktywności Słońca (promieniowanie korpuskularne najsilniej oddziałujące na warstwę F_2), burze jonosferyczne trwające od kilku godzin do paru dni. Nasilają się one podczas najwyższej aktywności Słońca w jego jedenastoletnim okresie aktywności. Występują również takie zjawiska, jak zanik powszechny, zwany zjawiskiem Mögel-Dellingera, które polega na nagłym wzroście jonizacji warstwy D, gwałtownym wzroście absorpcji i w konsekwencji zaniku odbioru fal krótkich na części kuli ziemskiej, która jest aktualnie oświetlana przez Słońce. Czas zaniku waha się od kilku minut do 2-3 godzin. Najdłużej trwa on na mniejszych częstotliwościach krótkofalowych a najkrócej na częstotliwościach największych. Przyczyną tych zaników są wybuchy gazów na Słońcu.



Rys. 3.6. Przykładowy przebieg dobowy wartości MUF i LUF na trasie krótkiej (1500 km): a)grudzień; b) czerwiec [10].

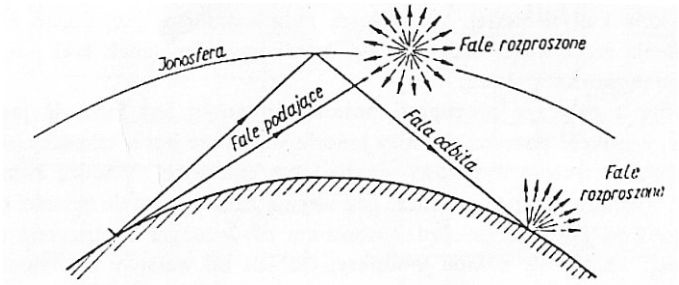
- Zjawisko Dopplera – występuje w okresach silnego przyrostu lub ubytku gęstości elektronowej w jonosferze, kiedy wysokość warstwy jonosferycznej może szybko zmieniać się w czasie. Występuje wtedy zjawisko zmiany częstotliwości fali. Zmiana częstotliwości jest dodatnia, gdy wysokość pozorna warstwy jonosferycznej maleje, zaś ujemna, gdy ta wysokość wzrasta. W zakresie HF występują zmiany dochodzące do kilku kiloherców.
- Zjawisko echa – sygnał może docierać do odbiornika od strony krótszego łuku w płaszczyźnie wielkiego koła przechodzącego przez punkt nadawania i odbioru (sygnał bezpośredni) oraz od strony dłuższego łuku (sygnał pośredni). Ilustruje to rysunek 3.7.



Rys. 3.7. Sygnał bezpośredni i sygnał pośredni przy odbiorze fal krótkich [10]

Sygnał pośredni dociera do odbiornika później niż sygnał bezpośredni. Nadejście sygnału pośredniego nazywane jest echem. Sygnały mogą być także odebrane przez odbiornik po wielokrotnym okrążeniu Ziemi. Czas okrążenia przez sygnał kuli ziemskiej po wielkim kole wynosi około 0,14 s – na każde 1000 km różnicy dróg sygnału bezpośredniego i sygnału pośredniego wypada odstęp czasowy równy około 3 ms. Prócz tego echa, występują również tzw. echa bliskie (rzędu 1 ms), które powstają podczas, gdy do odbiornika docierają dwa promienie wzdłuż różnych dróg propagacji.

Rozpraszanie fali jonosferycznej – fala w jonosferze ulega rozproszeniu na niejednorodnościach jonosfery oraz powierzchni ziemi. Obrazuje to rysunek 3.8.



Rys. 3.8. Rozproszenie fal krótkich w jonosferze oraz na powierzchni ziemi [10].

Dzięki temu rozpraszaniu można nawiązać łączność przy zastosowaniu fal o częstotliwości większej od MUF. Rozpraszanie na powierzchni ziemi powoduje pojawienie się zwrotnej fali rozproszonej.

Należy jeszcze omówić zjawiska związane z zanikami przy odbiorze fal krótkich. Są one bardzo dokuczliwe, w porównaniu np. do zaników dla fal średnich. Na skutek małej stabilności warstwy F_2 oraz wielodrogowości sygnału, przy odbiorze HF występują częste i głębokie zmiany natężenia pola. Zaniki można podzielić na kilka rodzajów:

- Zaniki interferencyjne – interferencja promieni, które dotarły do miejsca odbioru różnymi drogami.
- Zaniki polaryzacyjne – powstają przy interferencji w miejscu odbioru promienia zwyczajnego i nadzwyczajnego (promienie powstałe wskutek rozszczepienia promienia padającego na jonosferę pod wpływem ziemskiego pola magnetycznego, spolaryzowane kołowo z przeciwnymi zwrotami). Ponieważ amplitudy obu promieni są nierówne, to w wyniku interferencji wypadkowe pole ma polaryzację eliptyczną. Zmiana polaryzacji jest równoważna zmianie natężenia pola elektromagnetycznego, gdyż odbiornik zwykle jest w stanie odebrać jedynie składową o określonej polaryzacji (pionowa lub poziomą).
- Zaniki absorpcyjne – spowodowane wahaniami tłumienia fali przez warstwę E (występuje głównie około południa)
- Zaniki graniczne – (jak już wspomniano wcześniej) występuje, gdy częstotliwość sygnału jest bliska MUF, przy chwilowych wahaniach gęstości elektronowej. Charakterystyczną właściwością tego zaniku są znaczne wartości natężenia pola tuż przed wystąpieniem zaniku (tłumienie na częstotliwościach bliskich MUF jest małe)
- Zaniki uskokowe – zachodzą w miejscach znajdujących się w pobliżu uskoku danej fali, przy zmianie granicy między strefą martwą a strefą odbioru

Wiele z tych zaników występuje jednocześnie. Mogą mieć też charakter synchroniczny lub selektywny. Przy zaniku synchronicznym natężenie pola wszystkich składowych fali zmodu-

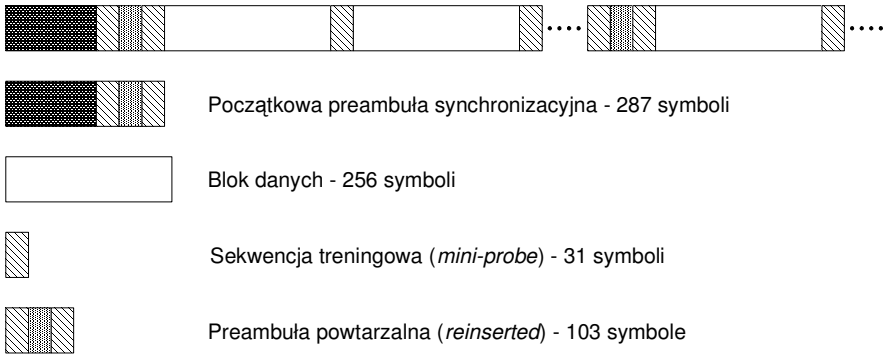
lowanej ulega jednakowym wahaniom. Przy zaniku selektywnym zmiany natężenia pola poszczególnych składowych widma są różne.

3.3. Modemy krótkofalowe

W niniejszej pracy nie przedstawia się specyfikacji typowych modemów stosowanych w łączności krótkofalowej, ponieważ szczegółowo omówiono je już w pracach statutowych [11 – 13]. Ich dokładna specyfikacja dostępna jest również w dokumentach [14, 15].

Wspomnieć należy tu tylko, że autor ogranicza się do specyfikacji modemów krótkofalowych, których prędkości transmisji są większe niż 2400 bitów na sekundę (3200, 4800, 6400, 8000, 9600 i 12800 bps).

Warto w tym miejscu jeszcze dodać, że format ramki wykorzystywanej przez modem został przedstawiony na rysunku 3.9. Ramka rozpoczyna się od preambuły synchronizacyjnej o długości 287 symboli, po której następują 72 bloki danych o długości 256 symboli, transmitowane pomiędzy dwoma sekwencjami treningowymi (*mini-probes*) o długości 31 symboli. Każda 72-ga sekwencja *mini-probe* zastępowana jest przez preambułę powtarzalną, określaną jako *reinserted*, której zadaniem jest ułatwienie właściwego odbioru danych (między innymi, umożliwienie kompensacji przesunięcia Dopplera oraz zapewnienie synchronizacji).



Rys. 3.9 Format ramki dla modemów HF o przepływnościach powyżej 2400 bps.

W niniejszej pracy przytoczono powyższy rysunek głównie w celu pokazania, że podstawową jednostką danych w tego typu urządzeniach jest blok danych zawierający 256 symboli modulacyjnych. Wartość ta jest istotna z punktu widzenia projektowanego kryptosystemu.

Zaproponowane i omawiane w tej pracy rozwiązanie kryptosystemu dla potrzeb transmisji danych w kanale krótkofalowym może zostać wykorzystane w modemach HF opartych o technologię SDR [16]. Przykładowym rozwiązaniem urządzenia zdefiniowanego programowo może być choćby [17, 18].

4. Projekt systemu

Koncepcja systemu zakłada przede wszystkim realizację wszystkich mechanizmów kryptograficznych niezbędnych do stworzenia w pełni bezpiecznego systemu transmisji danych w łączu krótkofalowym. Opracowany system opiera się w dużej mierze na istniejących algorytmach kryptograficznych takich jak: AES, RSA oraz SHA-256 oraz ich modyfikacjach. Bezpieczeństwo zaproponowanego rozwiązania zależy zatem w znacznym stopniu od skuteczności i odporności na ataki kryptoanalityczne wspomnianych algorytmów. Przy projektowaniu niniejszego systemu założono więc, że szyfracja przy pomocy AES jest w pełni poufna, funkcja RSA natomiast gwarantuje uwierzytelnienie oraz wraz z SHA-256 integralność.

Przy projektowaniu tego typu systemu należy wziąć pod uwagę pewne dodatkowe elementy, które w znaczny sposób wpłyną na postać końcową proponowanego rozwiązania. Te dodatkowe składniki to przede wszystkim:

- Specyfika kanału HF i jego wpływ na transmisję radiową (patrz. rozdział 3);
- Istniejąca standaryzacja modemów krótkofalowych (patrz punkt 3.3) oraz [14, 15];
- Minimalne ograniczenie pasma użytecznego spowodowane wprowadzeniem systemu.

W kolejnych paragrafach rozdziału czwartego przedstawiona zostanie podstawowy projekt zaproponowanego systemu z uwzględnieniem powyższych ograniczeń.

4.1. Informacje wstępne

W celu stworzenia autorskiego rozwiązania systemu bezpiecznej transmisji danych w paśmie HF zebrano i przeanalizowano dokumentację istniejących standardów związanych z kryptografią:

- Standardy szyfracji: DES [3], 3DES [3], AES [4], RSA [6];
- Funkcje skrótu: SHA [7], MD5 [8];
- Kody uwierzytelniające MAC [9, 19];
- Standard podpisu cyfrowego [20];
- I inne [21 – 23].

W dalszej kolejności przeanalizowano systemy bezpieczeństwa w wybranych znanych systemach łączności bezprzewodowej (IEEE 802.11 [24], Bluetooth [25], GSM/UMTS [26]). Szczególną uwagę zwrócono tu na odmiennie realizowane poszczególne mechanizmy kryptograficzne oraz sposoby zarządzania całym systemem bezpieczeństwa, a w szczególności zarządzania kluczami szyfrującymi.

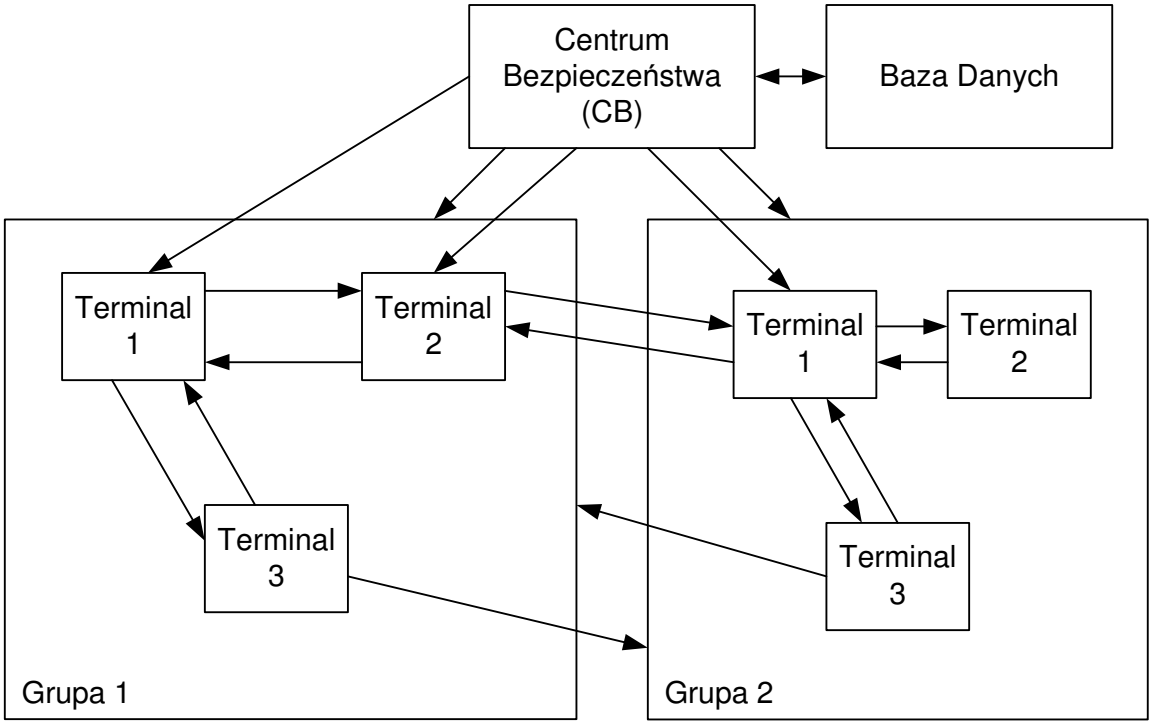
Na podstawie zdobytej wiedzy opracowano projekt systemu bezpiecznej transmisji danych w paśmie HF. Projekt ten uwzględnia między innymi:

- Wybór oraz modyfikację odpowiednich algorytmów do realizacji poszczególnych mechanizmów kryptograficznych;
    - Wykorzystanie zmodyfikowanego algorytmu szyfrującego AES (AES-CTR);
    - Modyfikacja algorytmu CMAC dla realizacji mechanizmu integralności i uwierzytelniania;

    - Wykorzystanie zmodyfikowanego standardu podpisu cyfrowego dla realizacji mechanizmów uwierzytelniania oraz integralności w komunikacji centrum bezpieczeństwa z terminalami;
- Stworzenie protokołu komunikacyjnego pomiędzy terminalami oraz pomiędzy centrum bezpieczeństwa a terminalem;
    - Określenie struktur podstawowych wiadomości;
    - Zdefiniowanie sposobu adresowania oraz „podpisywania" wiadomości;
    - Określenie sposobu tworzenia elementów składowych wiadomości sterujących oraz zawierających dane użyteczne z wykorzystaniem nanoinstrukcji (pikoinstrukcji);
    - Oraz inne szczegóły implementacyjne protokołu.
- Opracowanie sposobu dystrybucji i zmiany kluczy szyfrujących, a również mechanizmu ich generowania;
- Opracowanie schematu hierarchizacji kluczy szyfrujących wraz z określeniem warunków, kiedy tracą one ważność i podlegają aktualizacji;
- Określenie sposobu wykorzystania oraz generacji sekwencji specjalnych dla algorytmow AES-CTR oraz zmodyfikowanego CMAC, jak również do bezpieczniejszego przesyłania kluczy szyfrujących;
- Opracowanie autorskiego mechanizmu pozwalającego na łatwe określanie poziomu autoryzacji danego terminala oraz wykorzystywanie tylko niewielkiej liczby kluczy do realizacji transmisji typu terminal-terminal w całym systemie;
- Opracowanie struktury modułu kryptograficznego oraz centrum bezpieczeństwa (CB);
    - Zdefiniowanie parametrów (klucze, maski, ID) opisujących te dwa elementy;
    - Określenie sposobów definiowania tych parametrów możliwości ich zmian oraz znaczenia w całym systemie;
    - Zdefiniowanie zakresu komend sterujących wysyłanych przez CB oraz sposobu reagowania na nie przez terminal;
- Określenie sposobu współpracy terminala z modułem kryptograficznym;
- Dopasowanie systemu do dostępnych standardów modemów HF oraz zastosowania go w trudnym, jonosferycznym kanale radiowym;
- Opracowanie sposobu zarządzania całym systemem poprzez generowanie instrukcji sterujących przez centrum bezpieczeństwa;
- Określenie sposobu tworzenia wirtualnych kanałów transmisyjnych pozwalających na transmisję o określonym poziomie aoutoryzacji;
- Oraz wiele innych istotnych szczegółów pozwalających na implementację zaprojektowanego systemu.

4.2. Struktura zaproponowanego kryptosystemu

Przykładowa struktura zaproponowanego rozwiązania przedstawiona została na rysunku 4.1.



Rys. 4.1. Przykładowa struktura zaproponowanego systemu bezpieczeństwa.

Na podstawie powyższego rysunku widać, że prezentowany system bezpieczeństwa bazuje na łączach jednokierunkowych. Jest to jedna z najistotniejszych cech odróżniająca to rozwiązanie od wielu systemów tego typu, które bazują na łączach dwukierunkowych (GSM, UMTS, WLAN itp.).

Realizacja bezpiecznego systemu bazującego tylko na transmisji informacji i nie posiadającego możliwości uzyskania informacji zwrotnej o poprawności przeprowadzonej transmisji jest dużo trudniejsza niż w przypadku systemów typu pytanie-odpowiedź.

Należy w tym miejscu oczywiście wyraźnie zaznaczyć, że wspomniana tu jednokierunkowość tyczy się tylko wymiany informacji dotyczących opracowanego protokołu transmisji informacji związanych z systemem kryptograficznym. Najczęściej bowiem modemy HF posiadają wbudowane protokoły umożliwiające uzyskanie informacji zwrotnych. Nie jest to jednak część systemu bezpieczeństwa i nie ma nic wspólnego z rysunkiem 4.1.

W tym miejscu należy dopowiedzieć na pytanie: dlaczego wybrano taką strukturę systemu. Odpowiedź okazuje się być stosunkowo prosta. W przypadku systemu łączności pracującego w bardzo destruktywnym kanale krótkofalowym (patrz rozdział 3) trudno jest wymagać, aby jakości i dostępności systemu na poziomie spotykanym na przykład w sieciach komórkowych. Autor założył zatem, że zaproponowane rozwiązanie musi funkcjonować nawet w warunkach, w których łączność z CB jest ograniczona przez nawet bardzo długi czas, a komunikacja zwrotna może często nie być możliwa.

Każda odebrana wiadomość z centrum bezpieczeństwa (adresowana do danego terminala, grupy lub rozsiewcza) powinna zatem zawierać już informacje, które pozwalają na dokonanie aktualizacji danych w module kryptograficznym. A dodatkowo wiadomość taka musi być wiarygodna, poufna oraz integralna.

Komunikacja między terminalami podlega podobnym zasadom. Pojedyncza wiadomość przesłana od jednego terminala do drugiego pozwala na jej deszyfrację, sprawdzenie jej integralności i uwiarygodnienie nadawcy. Określony może w niej być również poziom autoryzacji wymagany do jej odczytania.

Przy takim podejściu pojawiają się jednak problemy. Po pierwsze, terminal nadawczy nie wie czy jego wiadomość dotarła do adresata. Rozwiązanie tej sprawy jest nieistotne z punktu widzenia bezpieczeństwa systemu. Nie jest zatem definiowane w zaproponowanej specyfikacji tego systemu. Warto tu jednak przypomnieć, że problem ten jest zazwyczaj rozwiązany przez inne protokoły komunikacyjne wykorzystywane w przypadku transmisji danych w łączu HF.

Kolejny problem jednak stanowi fakt, że również CB nie posiada informacji czy terminal (a właściwie moduł kryptograficzny terminala) zaktualizował swoje dane. Z tą sprawą wiąże się kilka aspektów:

- Centrum bezpieczeństwa powinno transmitować dane aktualizacyjne do terminali w sposób względnie ciągły. Może być do tego przystosowany np. pewien konkretny kanał HF.
- CB powinno wykorzystywać wiadomości typu multicast w celu uzyskania większej wydajności wysyłanych aktualizacji.
- Czas ważności kluczy w module zapewnia spory margines bezpieczeństwa. Jest bowiem sporo czasu, w którym terminal wystarczy, że odbierze właściwie tylko jedną informację aktualizacyjną od CB przeznaczoną np. dla jego grupy.
- Zmniejszenie bezpieczeństwa systemu jest ograniczone tylko do danych grup, z których np. skradziony terminal może odbierać informacje (nadawać może również poza, ale tego typu wiadomości są z góry określane jako mało bezpieczne!). Niebezpieczeństwo ograniczone jest również czasowo, ponieważ klucze skradzionego terminala z czasem same utracą ważność, a na pewno nie uzyska on aktualizacji.

Na podstawie rysunku 4.1 widać, że CB może wysyłać wiadomości nie tylko do konkretnego terminala, ale również do całej grupy terminali. Wiadomości wysyłane przez CB będą w ogólności nazywane SMM (*Security Management Messages*).

Terminale mogą przesyłać aż pięć typów wiadomości zwanych ogólnie UDM (*User Data Messages*):

- Wiadomość rozsiewcza (broadcast) – mało bezpieczna wiadomość możliwa do odebrania przez wszystkie terminale w zasięgu stacji nadawczej i posiadające odpowiednie klucze;
- Wiadomość multicast typ 1 – wiadomość do całej grupy nadawcy wiadomości;
- Wiadomość multicast typ 2 – wiadomość do tej części grupy nadawcy wiadomości, która posiada odpowiednie uprawnienia (odpowiedni poziom autoryzacji) do odbioru tej wiadomości;
- Wiadomość unicast typ 1 – wiadomość do jednego terminala będącego członkiem grupy nadawcy;
- Wiadomość unicast typ 2 – mało bezpieczna wiadomość do jednego terminala nienależącego do grupy nadawcy.

Szczegóły odnośnie podstawowych struktur wiadomości SMM oraz UDM przedstawione zostaną w dalszej części niniejszego rozdziału.

Protokół transmisji pomiędzy CB oraz bazą danych (patrz rysunek 4.1) nie podlega specyfikacji i zależy od konkretnej realizacji kryptosystemu.

Poszczególne realizacje systemu będą w pełni niezależne od siebie i nie będzie możliwości interakcji pomiędzy należącymi do nich CB. Występować będzie jednak pewien sposób komunikacji pomiędzy terminalami należącymi do różnych kryptosystemów (patrz punkt 4.3).

W celu zapewnienia możliwości współistnienia wielu systemów, a więc i takich nie określonych w tej pracy, wprowadzono niezależny identyfikator systemu. Jest to parametr jednobajtowy i w przypadku prezentowanego systemu ma on wartość *0x86*. W dalszej części pracy *System ID* będzie rozumiane zawsze jako właśnie ta wartość.

Innym parametrem stosowanym w dalszym opisie będzie *Provider ID* określającym poszczególne i niezależne realizacje systemu bezpieczeństwa (np. zaproponowanego systemu o identyfikatorze 0x86). Konkretna wartość *Provider ID* będzie odnosić się do danego centrum bezpieczeństwa.

4.3. Moduł kryptograficzny

Stanowi on niezależną część terminala użytkownika, a w szczególności modemu HF a jego projekt powstał w oparciu o standard [27]. Moduł ten pełnić będzie w przypadku nadawania pierwszy blok przetwarzania danych użytkownika, po którym występować będą kolejne zdefiniowane w [14, 15] bloki funkcjonalne nadajnika. W przypadku odbiornika blok ten stanowić będzie ostatni element toru odbiorczego.

Warto tu zwrócić uwagę na dwa aspekty. Po pierwsze: moduł kryptograficzny powinien umożliwiać działanie modemu HF z wyłączonym systemem bezpieczeństwa. Po drugie: szyfracji nie mogą podlegać:

I. Wszelkie preambuły i ciągi treningowe (a jedynie dane użytkownika);

II. Sekwencja EOM (*End Of Message*) [14, 15] – konieczna do wykrycia końca transmisji.

Oczywistym jest fakt, że powyższe dwa warunki będą zawsze spełnione, gdy moduł kryptograficzny będzie pierwszym blokiem funkcjonalnym w torze nadawczym. Zostały one jednak tu przedstawione w celu uwypuklenia tego faktu.

Moduł kryptograficzny to element, który posiadać musi pewien zasób pamięci, w którym przechowywane będą klucze kryptograficzne oraz wiele innych identyfikatorów czy sekwencji skramblujących. Wymogi co do pamięci nie są duże. Pamięć wymagana do przechowywania danych wraz ze sporym zapasem to 64 kB (kilobajtów) na każdą realizację systemu! Przy rozsądnym założeniu maksymalnie czterech systemów, z którymi współpracować będzie dany moduł, daje to łącznie 256 kB wymaganej pamięci. Warto tu jednak zauważyć, że część z tych 256 kB pamięci powinna być typu ROM, ponieważ niektóre parametry nie będą mogły być zmieniane. Sposób jednak organizacji tej pamięci nie podlega specyfikacji i jest dowolny w zależności od realizacji. Ważne jest jednak, aby uniemożliwić zewnętrzny dostęp do tych zasobów, ponieważ informacje tam zawarte mogą przyczynić się do złamania danej realizacji systemu.

Zasoby pamięci wymagane z kolei do realizacji obliczeń są również niewielkie. Potrzeba bowiem zaledwie 16 kB pamięci RAM.

Do realizacji obliczeń wymagany będzie układ o niezbyt dużych wymogach odnośnie szybkości przetwarzania. Jest to związane z niewielkimi przepływnościami osiąganymi w modemach

krótkofalowych [14, 15]. Autor zaleca stosowanie układów typu FPGA a nie procesorów DSP, ponieważ reprogramowalność i elastyczność nie jest tu cechą krytyczną. Raz zaprogramowany układ FPGA może pełnić swa rolę w każdej realizacji systemu *0x86*.

Do podstawowych zadań modułu kryptograficznego należą między innymi:

- Deszyfracja informacji przesyłanych przez CB oraz aktualizacja konfiguracji w zależności od otrzymanych instrukcji;
- Uwierzytelnianie centrum bezpieczeństwa oraz kontrola integralności danych otrzymywanych od CB;
- Szyfracja i deszyfracja danych w trakcie transmisji pomiędzy terminalami;
- Uwierzytelnianie terminala nadawczego oraz kontrola integralności danych otrzymanych od niego;
- Deszyfracja kluczy otrzymywanych od CB;
- Kontrola aktualności uprawnień, kluczy szyfrujących oraz czasu pozostałego do końca aktywacji modułu;
- Rozpoznawanie nadawcy, adresata oraz typu otrzymywanych wiadomości (a także ich długości i innych parametrów);
- Dekodowanie wiadomości SMM oraz UDM;
- Tworzenie wiadomości UDM;
- Odrzucanie wiadomości niespełniających odpowiednich kryteriów;
- Wyznaczanie właściwych wartości licznika CTR przy realizacji zmodyfikowanego algorytmu AES-CTR;
- Algorytmy wykorzystywane przez moduł kryptograficzny:
    - Szyfracja i deszyfracja zmodyfikowanym AES-CTR;
    - Wyznaczanie i weryfikacja zmodyfikowanej sekwencji CMAC;
    - Uwierzytelnianie CB z wykorzystaniem RSA oraz SHA-256;
- Inne.

Moduł kryptograficzny odrzuci wiadomość, która:

- Jest zakodowana z wykorzystaniem innego kryptosystemu niż *0x86*;
- Została zakodowana z wykorzystaniem nieobsługiwanej realizacji systemu (nieobsługiwany *provider*) *0x86*;
- Jest adresowana do kogoś innego;
- Wymaga wyższego poziomu autoryzacji;
- Nie można jej poprawnie zdeszyfrować;
- Posiada błędne instrukcje;
- Nie pozwala na uwierzytelnienie nadawcy;
- Nie jest integralna.

Każdy moduł kryptograficzny musi zawierać dane zdefiniowane w tabeli 4.1.

Tab. 4.1. Wartości przechowywane w module kryptograficznym.

Nazwa		Długość [bity]	Możliwość zmiany	Opis
System ID		8	Brak	Identyfikator prezentowanego kryptosystemu (wartość stała *0x86*)
Provider ID 1 – 4		8	Brak	Identyfikator jednej z maksymalnie czterech konkretnych realizacji kryptosystemu obsługiwanych jednocześnie przez pojedynczy moduł kryptograficzny
Module ID		64	Brak	Stały identyfikator modułu kryptograficznego
User ID 1 – 8	*Group ID 1 – 8*	24	Jest	Identyfikatory maksymalnie ośmiu grup, do których może należeć użytkownik
	InGroup ID 1 – 8	8	Jest	Identyfikatory użytkownika w maksymalnie ośmiu grupach, do których może on należeć
AuthorizationMask 1 – 8		128	Jest	Maska uprawnień maksymalnie ośmiu grup, do których może należeć użytkownik
RSAPublicKey		1024	Brak	Publiczny klucz RSA centrum bezpieczeństwa
MasterKey 0		128	Brak	Klucz główny modułu kryptograficznego
GroupKey 1 – 8		128	Jest	Klucze grup maksymalnie ośmiu grup, do których może należeć użytkownik
SessionKey A, B		128	Jest	Klucze sesji (A – aktualny, B – następny)
SessionBasicKey F		128	Brak	Klucz podstawowy sesji (wartość stała w danej realizacji systemu)
ActivationDate		16	Jest	Data wygaśnięcia aktywacji modułu kryptograficznego
GroupAuthorizationDate 1 – 8		16	Jest	Data wygaśnięcia uprawnień w maksymalnie ośmiu grupach, do których może należeć użytkownik
KeyValidityDate A, B		16	Jest	Daty ważności poszczególnych kluczy sesji
ScramblingTable A1 – A4		256 x 128	Brak	Cztery tablice wykorzystywane do generacji końcowych wartości licznika w algorytmie AES-CTR
ScramblingTable B1 – B4		256 x 128	Brak	Cztery tablice wykorzystywane przy uwierzytelnianiu z wykorzystaniem zmodyfikowanego algorytmu CMAC oraz do szyfracji kluczy AES otrzymanych od CB

W dalszej części niniejszego paragrafu powyższe parametry zostaną przedstawione bardziej szczegółowo.

W tym miejscu należy wyraźnie zaznaczyć, że niektóre parametry z powyższej tabeli wiążą się ze sobą jednoznacznie. Identyfikator modułu (*Module ID*) jest jednoznacznie powiązany z kluczem głównym (*MasterKey 0*). Dodatkowo oba te parametry są niepowtarzalne w całym kryptosystemie *0x86*. Nie może istnieć sytuacja, w której istnieć będą dwa moduły o takim samym *Module ID* (jest to jednoznaczny identyfikator). Nie powinno być też sytuacji, kiedy dwa lub więcej modułów posiada ten sam klucz główny. Jako jedyny jest on zawsze przechowywany w postaci jawnej (ponieważ nigdy nie jest przesyłany!)

Jednoznaczne powiązanie występuje również pomiędzy kluczami grup (*GroupKey*) oraz ich identyfikatorami (*Group ID*). Parametry te nie mogą się powtarzać w obrębie jednej realizacji systemu, ale nie tyczy się to przypadku różnych *Provider ID*.

Parametr *Module ID* rozumiany może być jako globalny adres użytkownika w systemie. Może on być wykorzystywany do adresowania połączeń w trybie punkt-punkt (*unicast*).

Group ID z kolei oznacza właściwie adres pewnej grupy użytkowników i wykorzystywany może być przy realizacji transmisji multicastowych.

Group ID wraz z *InGroup ID* tworzą łącznie pewien identyfikator użytkownika (*User ID*), jednoznaczny jednak tylko w obrębie danej realizacji systemu. Może on służyć do adresowania połączeń unicastowych. Sam parametr *InGroup ID* jest lokalnym adresem użytkownika w obrębie danej grupy. W danej grupie może znajdować się maksymalnie 256 użytkowników.

Maska uprawnień (*AuthorizationMask*) to 128-bitowy ciąg bitowy, w którym każda jedynka oznacza uprawnienie do odbioru/nadawania a zero jego brak. Numer pozycji, na której znajduje się dana jedynka lub zero nazywany będzie identyfikatorem tak zwanego wirtualnego kanału transmisyjnego w danej grupie. Kanały te wykorzystywane będą tylko w trakcie transmisji w obrębie danej grupy. Można więc stworzyć do 128 niezależnych kanałów w każdej z grup. Dany użytkownik może nadawać w danym wirtualnym kanale tylko, gdy posiada do tego uprawnienia (jedynka na odpowiedniej pozycji). Analogicznie terminal może odbierać informacje tylko, gdy posiada odpowiedni poziom autoryzacji.

Klucz publiczny RSA (*RSAPublicKey*) konieczny jest dla uwierzytelniania wiadomości pochodzących z CB oraz kontroli ich integralności. Parametr ten jest jawny w całej realizacji systemu.

Klucz główny oraz klucze grup wykorzystywany są przez CB do transmisji informacji do terminali. Klucze grup wykorzystywane są również przez moduły do realizacji zmodyfikowanego algorytmu CMAC w celach uwierzytelniania i kontroli integralności.

Klucze sesji (*SessionKey*) wykorzystywane są przez moduły kryptograficzne do transmisji informacji użytkowych. Z założenia do odbioru i nadawania powinien być wykorzystywany zawsze klucz A (aktualny), ale użyty może być również klucz B (następny), jeśli klucz A utracił swą ważność (*KeyValidityDate*). Jeśli oba klucze utraciły swa ważność, to nie mogą być wykorzystywane (opcjonalnie mogą zostać automatycznie skasowane).

W związku z kluczami sesji obowiązuje zasada, że podczas ich aktualizacji na pozycję aktualnego klucza trafia klucz następny, a ten z kolei jest aktualizowany na nowy. Moduł nie zaktualizuje kluczy na starsze (wraz z kluczem CB musi przesyłać ich datę ważności).

Moduł aktywny do transmisji wykorzystać może również klucz F (*SessionBasicKey*), ale jest to rozwiązanie mało bezpieczne, ponieważ klucz ten jest stały i znany w danej realizacji systemu. Podejście takie wykorzystać może moduł nie mający ważnych kluczy sesji.

Data do której ważna jest aktywacja modułu (*ActivationDate*) wykorzystywana jest w celu automatycznej dezaktywacji tegoż modułu po jej upłynięciu. W tym miejscu warto zdefiniować pojęcie modułu aktywnego. Jest to taki moduł, który posiada jakikolwiek klucz grupy wraz z identyfikatorem *User ID*. Po upłynięciu daty aktywacji w module pozostają tylko parametry niezmienialne (patrz tabela 4.1) oraz wyzerowane zostają wszystkie daty. Moduł nieaktywny ponadto nie może aktualizować kluczy sesji, ponieważ nie posiada kluczy grup.

Data autoryzacji w grupie (*GroupAuthorizationDate*) to data, po której nie możliwy stanie się odbiór i nadawanie wiadomości w danej grupie. Po upływie tej daty bowiem wyzerowana zostanie maska uprawnień danej grupy, skasowany będzie odpowiedni klucz grupy oraz identyfikator *User ID*.

Znaczenie tablic skramblujących (*ScramblingTable A* oraz *B*) zostanie szerzej przedstawione w paragrafie 4.5 niniejszej pracy.

Podsumowując należy stwierdzić, że pojedynczy moduł obsługiwać może tylko jeden typ systemu (w tym przypadku *0x86*), ale kilka (maksymalnie cztery) jego realizacji.

Nie ma możliwości transmisji pomiędzy dwoma różnymi realizacjami (różne parametry *Provider ID*) systemu (nawet w trybie punkt-punkt). Jeśli moduł jednak obsługuje więcej niż jedną realizację, to transmisja może odbywać się niezależnie w obrębie danej realizacji.

Moduł po odebraniu i zdekodowaniu wiadomości zwraca na wyjściu bajt statusu i ewentualnie zdeszyfrowane dane użytkownika (tylko dla poprawnie zdekodowanej wiadomości UDM – patrz punkt 4.6).

W tabeli 4.2 przestawiono możliwe postacie bajtu statusu.

Tab. 4.2. Status modułu kryptograficznego.

Status	Opis
00	Błędne funkcjonowanie modułu
10	Wiadomość SMM poprawnie zdekodowana (Wszelkie instrukcje od CB zostały wykonane)
11	Wiadomość UDM poprawnie zdekodowana
20	System inny niż *0x86*
30	Nieobsługiwana realizacja systemu
40	Wiadomość adresowana do kogoś innego
41	Brak uprawnień do odbioru wiadomości
50	Błąd deszyfracji (Błędne Nano 2F – patrz punkt 4.6)
60	Błędna(e) nanoinstrukcja(e)
61	Błędna treść nanoistrukcji
70	Błąd uwierzytelniania/kontroli integralności
80	Moduł kryptograficzny jest nieaktywny
90	Brak aktualnego klucza sesji

W powyższej tabeli kolor zielony oznacza poprawne zdekodowanie wiadomości, a kolor czerwony błędne.

4.4. Centrum Bezpieczeństwa (CB)

Centrum bezpieczeństwa jest jednym z najistotniejszych elementów większości kryptosystemów. Pełni ono zasadniczą rolę w procesie zarządzania systemem bezpieczeństwa. W zaproponowanym rozwiązaniu do zadań CB należą między innymi:

- Generowanie kluczy;

Problem generacji kluczy jest wydawałoby się sprawą oczywistą. W rzeczywistości jednak element ten stanowi słabe ogniwo projektowanych systemów. Ręczne generowanie jest obarczone schematycznym myśleniem człowieka. Tak zwany czynnik ludzki jest tu krytyczny. Do generowania stosuje się zatem najczęściej generatory pseudolosowe. Dla potrzeb niniejszego projektu wykorzystano generator deterministyczny BBS (*Bluma-Bluma-Shuba*) [28] uważany za jeden z najbezpieczniejszych algorytmów tego typu obecnie stosowanych.

W przypadku generowania kluczy istotna jest również ich długość. Dzisiaj uważa się, że klucze o długości rzędu 64 bitów nie zapewniają wystarczającego poziomu bezpieczeństwa i metodą całkowitego przeglądu można złamać je w stosunkowo niedługim okresie czasu [28]. Na potrzeby proponowanego rozwiązania wybrano zatem klucze długości 128 bitów (dla algorytmu AES oraz zmodyfikowanego CMAC). Długość ta uważana jest dzisiaj za bezpieczną wartość. W przypadku algorytmu RSA zdecydowano się na bardzo bezpieczną długość kluczy – 1024 bity.

- Przechowywanie kluczy i danych użytkowników;

Przechowywanie jest istotną sprawą z punktu widzenia systemu bezpieczeństwa, ponieważ nieautoryzowany dostęp do bazy zawierającej klucze np. wszystkich użytkowników systemu oznaczałby naruszenie bezpieczeństwa systemu. W proponowanym rozwiązaniu przyjęto ogólna zasadę, która stanowi, że nie należy przechowywać kluczy w postaci jawnej, a raczej zaszyfrowanej z wykorzystaniem innego klucza.

Warto tu również dodać, że w przypadku przechowywania kluczy i danych użytkowników istotne jest ograniczenie dostępu do tych informacji tylko dla osób do tego upoważnionych. Ważna jest tu zatem ścisła realizacja mechanizmu dostępności.

- Dystrybucja i zmiany kluczy;

Jest to jedno z ważniejszych zadań CB i kluczowe zagadnienie projektowanego kryptosystemu. Mechanizm dystrybucji kluczy jest bowiem krytyczny jeśli chodzi o nieautoryzowany dostęp. W dalszej części niniejszej pracy przedstawiony zostanie szczegółowo zaproponowany bezpieczny protokół komunikacyjny pomiędzy CB a użytkownikami systemu, który umożliwiać będzie między innymi bezpieczną dystrybucję kluczy w trybie OTAR (*Over The Air Re-keying*). Mechanizm dystrybucji kluczy opracowano opierając się o standard [29].

Określony musi zostać również schemat zmian (aktualizacji) kluczy. Trzeba w tym miejscu rozgraniczyć dwie odmienne sytuacje. Pierwsza to taka, w której klucze tracą swą ważność i muszą zostać zmienione, aby utrzymać bezpieczeństwo całego systemu. Częstość tych aktualizacji zależy od konkretnego systemu. Dla potrzeb niniejszego rozwiązania przyjęto, że klucze sesji powinny być ważne około miesiąca (parametr ten jest modyfikowalny). Druga sytuacja to taka, w której naruszone zostało bezpieczeństwo systemu i należy zaktualizować klucze, aby krypto system mógł dalej pełnić swą rolę. Szczegóły dystrybucji kluczy będą przedstawione w dalszej części pracy.

- Aktywacja i dezaktywacja modułów kryptograficznych;

Centrum bezpieczeństwa musi posiadać możliwość zdalnego zarządzania użytkownikami systemu. Jednym z działań, które może podjąć CB jest właśnie aktywacja nowych modułów kryptograficznych oraz dezaktywacja tych które nie są dalej wykorzystywane lub zostały np. skradzione. Szczegóły tego procesu omówiono w dalszej części pracy.

- Przydzielanie do grup, zmiana poziomu autoryzacji i inne.

CB musi posiadać inne narzędzia pozwalające na bardziej szczegółową kontrolę nad całością systemu. W niniejszym rozwiązaniu zaproponowano, że możliwe będzie między innymi przydzielanie użytkowników do pewnych grup posiadających pewne wspólne cechy. Wprowadzona zostanie również tak zwana maska uprawnień dla danego użytkownika, która określać będzie poziom jego autoryzacji w danej grupie. Możliwym będzie również dowolne dodawanie i usuwanie użytkowników z jednej lub wielu grup. Szczegóły tych działań przedstawione zostaną w dalszej części niniejszej pracy.

- Tworzenie poprawnych składniowo wiadomości SMM zawierających instrukcje dla terminali.
- Wyznaczanie właściwych wartości licznika CTR przy realizacji zmodyfikowanego algorytmu AES-CTR;
- Szyfracja kluczy AES przysyłanych modułom kryptograficznym;
- Algorytmy wykorzystywane przez CB:
    - Szyfracja zmodyfikowanym AES-CTR;
    - Generacja uproszczonego podpisu z wykorzystaniem RSA oraz SHA-256;
- Inne.

Każde centrum bezpieczeństwa posiada swój własny identyfikator wspomniany już wcześniej *Provider ID*. Jest to wielkość jednobajtowa i rozgranicza poszczególne realizacje systemu bezpieczeństwa tu prezentowanego. Należy tu jednak zauważyć, że konkretna wartość *Provider ID* nie musi odnosić się wcale do jednego tylko CB. Często występować będzie bowiem sytuacja, w której dla jednej realizacji systemu dostępnych będzie kilka CB w celu np. zapewnienia dostępu do informacji aktualizacyjnych na większym obszarze. Istotny jest tu tylko fakt, że niezależnie od tego ile CB w danej realizacji będzie przewidzianych, to muszą się one komunikować z jedną wspólną bazą danych. Sposób tej komunikacji nie jest określany. Ważne tylko, aby w bazie znajdowały się zawsze aktualne informacje dotyczące terminali.

Same centra bezpieczeństwa są zatem jednostkami zarządzającymi systemem, a wszelkie dane powinny być przechowywane w dedykowanej dla danej realizacji systemu bezpiecznej bazie danych, do której dostęp będą mieć tylko autoryzowane osoby.

W bazie danych muszą znajdować się zatem dane poszczególnych użytkowników danej realizacji systemu.

Dodatkowo centrum bezpieczeństwa musi posiadać prywatny klucz RSA (*RSAPrivateKey* - ten sam w danej realizacji systemu) oraz znać tablice skramblujące A oraz B (*ScramblingTable A, B*).

4.5. Stosowane modyfikacje algorytmów kryptograficznych

Niniejszy projekt systemu opiera się na kilku podstawowych algorytmach kryptograficznych (AES-CTR, RSA, SHA-256 oraz CMAC). Dla potrzeb zaproponowanego rozwiązania część z nich została zmodyfikowana w celu dopasowania ich właściwości do wymogów stawianych systemowi oraz dla zwiększenia całościowego bezpieczeństwa systemu jak również umożliwieniu sprawnego nim zarządzania.

4.5.1. Zmodyfikowany AES-CTR

Klasyczne rozwiązanie AES przedstawiono w punkcie 2.3 niniejszej pracy, a tryb pracy CTR w punkcie 2.2. Dla potrzeb projektowanego kryptosystemu wybrano 128-bitowy algorytm AES, ponieważ zapewnia on bardzo wysoki poziom poufności przesyłanych wiadomości i na dzień dzisiejszy nie istnieją szybkie sposoby ataków na niego (poza atakiem pełnego przeglądu – wymagany bardzo długi czas).

Tryb CTR wybrano z kolei ze względu na:

- Brak powtarzalności szyfrogramów (ten sam tekst jawny daje różne teksty zaszyfrowane);
- Możliwość zrównoleglenia przetwarzania;
- Błędy nie propagują się;
- Możliwość modyfikacji licznika – dodatkowe zabezpieczenie.

Modyfikacja klasycznego rozwiązania AES-CTR wiąże się tylko i wyłącznie ze zmodyfikowanym sposobem wyznaczania kolejnych wartości licznika. Do tego celu korzysta się z tablicy skramblującej A.

Wartość licznika ustalana jest zgodnie ze wzorem:

$$CTR_i^{(128)} = SS_{j,Ak}^{(128)} \oplus (adresat, \{rok\}, \{miesiąi\}, \{dzień\}, \{godzina\}, \{i\}, ..., \{i\}) \qquad (4.1)$$

gdzie:

$CTR_i^{(128)}$ - oznacza 128-bitową sekwencję licznika dla *i*-tego 128-bitowego bloku danych;

$SS_{j,Ak}^{(128)}$ - oznacza *j*-ty wektor z *k*-tej tablicy skramblującej A;

adresat – adresat wiadomości (od 0 do 4 bajtów) – patrz punkt 4.6;

rok – rok wysłania wiadomości (dwie ostatnie cyfry dziesiętne zapisana binarnie, jeden bajt);

miesiąc – miesiąc wysłania wiadomości (od 1 do 12 binarnie, jeden bajt);

dzień – dzień wysłania wiadomości (od 1 do 31 binarnie, jeden bajt);

godzina – godzina wysłania wiadomości (od 0 do 23 binarnie, jeden bajt);

i – numer bloku wiadomości (jeden bajt) powtarzany tyle razy, aby długość bloku sumowanego modulo 2 z sekwencja skramblująca miała dokładnie 128 bitów.

Wartość *i* zmienia się w zakresie od *0x01* do *0xFF* (lub mniej jeśli wiadomość jest krótsza niż maksymalna długość – patrz punkt 4.6). Dla każdej nowej wiadomości *i* jest ustawiane na wartość początkową *0x01*.

Wartość *j* powinna być wybierana losowo (dla każdej wysyłanej wiadomości) z zakresu od 0 do 255 (256 wektorów w tablicy *ScramblingTable A*). Numer tablicy A powinien być zmieniany tylko w przypadku naruszenia bezpieczeństwa systemu.

Na podstawie wzoru 4.1 widać, że każda wiadomość wysyłana z wykorzystaniem zmodyfikowanego algorytmu AES-CTR jest ważna tylko przez godzinę. Należy zatem zapewnić poziom synchronizacji pomiędzy terminalami i pomiędzy terminalami a CB na poziomie

(przynajmniej z dokładnością do kilkunastu minut) pozwalającym na realizację transmisji. Czas powinien być przedstawiany zawsze w formie GMT (Greenwich Mean Time).

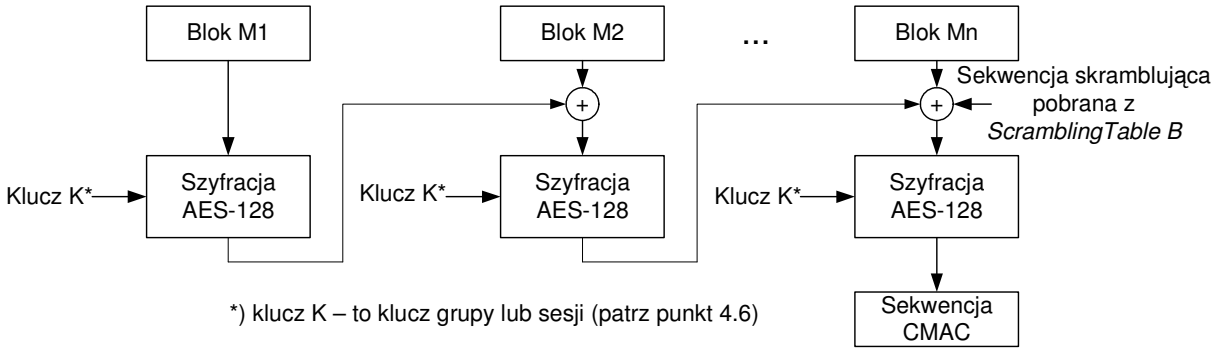
4.5.2. Zmodyfikowany CMAC

Podstawową wersję algorytmu CMAC przedstawiono w punkcie 2.6 niniejszej pracy. Dla celów zaproponowanego rozwiązania zmodyfikowano ten algorytm. Działania te podjęto ze względu na:

- W klasycznym rozwiązaniu CMAC czasami stosuje się dopełnienie ostatniego bloku danych w celu umożliwienia wyznaczenia sekwencji CMAC. W prezentowanym systemie taka sytuacja nie będzie miała miejsca, ponieważ długość wiadomości jest zawsze wielokrotnością 128 bitów (patrz punkt 4.6);
- Dla poprawy bezpieczeństwa systemu. Zamiast stosowania klucza *K1* (patrz punkt 2.6), który byłby rzadko zmieniany (podobnie zresztą jak klucz *K* – patrz punkt 4.6), postanowiono skorzystać ze 128-bitowej sekwencji skramblującej pobranej z tablicy skramblującej B (*ScramblingTable B*).

Numer wektora skramblującego wybierany jest losowo dla każdej wiadomości. Zmiana numeru tablicy B powinna być dokonywana tylko po naruszeniu bezpieczeństwa systemu.

Na rysunku 4.2 przedstawiono schemat realizacji zmodyfikowanego algorytmu CMAC.



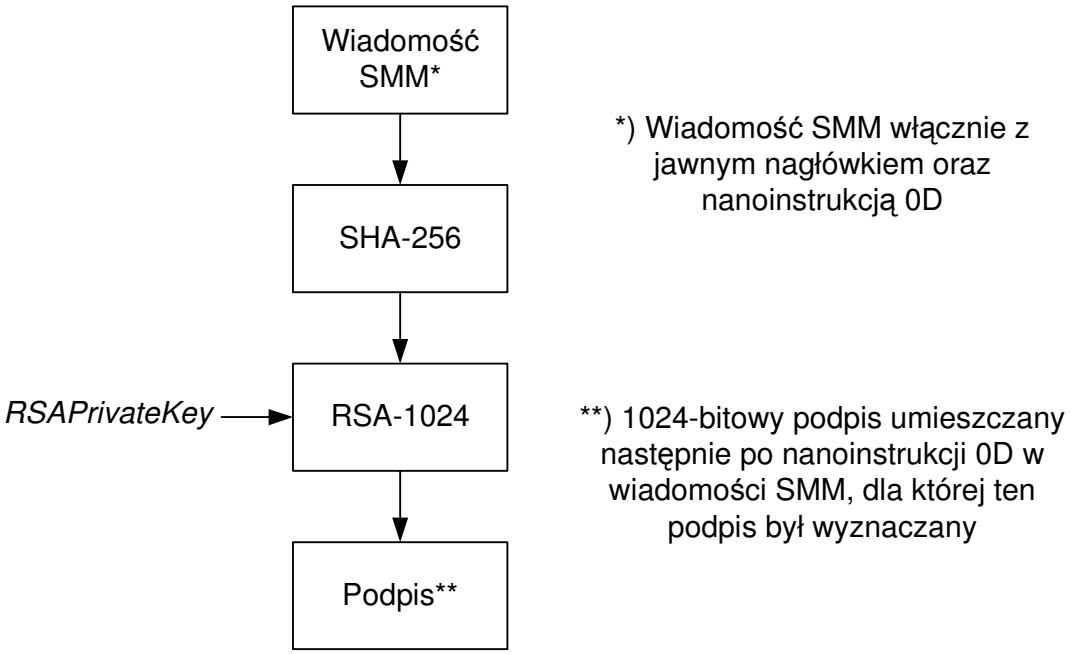
Rys. 4.2. Zmodyfikowany algorytm CMAC.

4.5.3. Uproszczony podpis RSA

Podpis RSA stosuje centrum bezpieczeństwa wysyłając wiadomości do terminali. Do tego celu wykorzystywany jest 1024-bitowy algorytm RSA (patrz punkt 2.4) oraz funkcja skrótu SHA-256 (patrz punkt 2.5).

W pierwszym etapie wyznaczania podpisu obliczany jest skrót wiadomości wraz z nagłówkiem oraz nanoinstrukcją 0D (patrz punkt 4.6). W efekcie uzyskuje się 256-bitową sekwencję, która w dalszej kolejności szyfrowana jest algorytmem RSA z wykorzystaniem prywatnego klucza RSA (*RSAPrivateKey*) centrum bezpieczeństwa. W rezultacie otrzymuje się 1024-bitową sekwencję, która umieszczana jest na końcu wiadomości SMM (po nanoinstrukcji 0D).

Cała procedura wyznaczania uproszczonego podpisu RSA przedstawiona została na rysunku 4.3.



Rys. 4.3. Procedura wyznaczania uproszczonego podpisu RSA.

W tym miejscu warto tylko dodać, że w przypadku wiadomości UDM kontrolę integralności oraz uwierzytelnianie realizuje się z wykorzystaniem zmodyfikowanego algorytmu CMAC (patrz punkt 4.5.2). Sekwencję CMAC wyznacza się również dla całej wiadomości włącznie z nagłówkiem oraz nanem 0D. Nie jest tu już wymagana funkcja skrótu. Wyznaczoną sekwencję dopisuje się również na końcu wiadomości (po nano 0D).

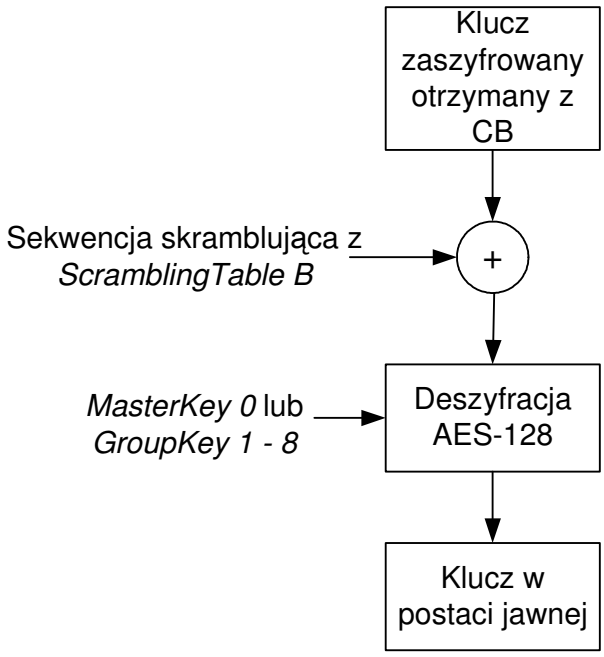
4.5.4. Szyfracja i deszyfracja kluczy AES

W zaproponowanym rozwiązaniu nigdy nie może wystąpić sytuacja, w której jakikolwiek klucz przesyłany przez CB w wiadomości SMM jest w niej zawarty w postaci jawnej. Poza faktem szyfracji wiadomości jako całości, dla kluczy stosuje się dodatkowe zabezpieczenie.

Każdy klucz odebrany od CB należy zsumować modulo 2 z odpowiednim wektorem z tablicy skramblującej B a nstępnie dokonać deszyfracji AES z odpowiednim kluczem (0 lub 1 – 8). Wybór klucza jest ściśle ustalony (nano 12 – patrz punkt 4.6). Generalnie jednak obowiązują dwie zasady:

- Klucz grupy powinien być zawsze przesyłany w wiadomościach zaszyfrowanych kluczem *MasterKey 0*;
- Klucz sesji powinien być zawsze przesyłany w wiadomości zaszyfrowanej kluczem grupy (*GroupKey 1 – 8*).

Na rysunku 4.4 przedstawiono schematycznie sposób deszyfracji kluczy otrzymanych od CB.



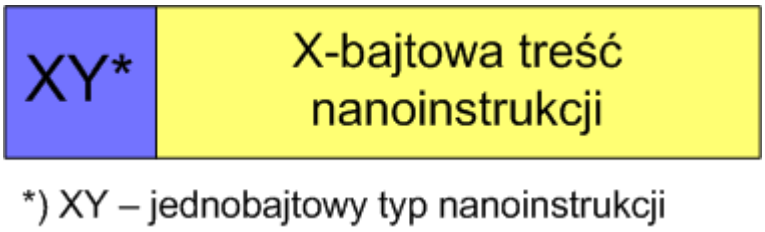
Rys. 4.4. Wyznaczanie jawnej postaci kluczy kryptograficznych otrzymywanych z centrum bezpieczeństwa.

4.6. Typy wiadomości oraz ich struktura

Jak już wcześniej wspomniano w zaproponowanym kryptosystemie stosuje się dwa typy wiadomości:

- SMM (*Security Management Message*) – wiadomość wykorzystywana przy komunikacji centrum bezpieczeństwa z terminalami;
- UDM (*User Data Message*) – wiadomość wykorzystywana przy komunikacji pomiędzy terminalami.

W celu omówienia tych dwóch typów wiadomości należy w pierwszej kolejności przedstawić podstawową jednostkę, która służy do ich budowy. Jest nią tak zwana nanoinstrukcja. Na rysunku 4.5 przedstawiono schematycznie jej ogólną postać.



Rys. 4.5. Ogólna postać nanoinstrukcji.

Jak widać każda nanoinstrukacja (w skrócie nano) składa się z dwóch części. Pierwszy bajt to typ nano, a dalej wystąpić może od zera do piętnastu bajtów treści nanoinstrukcji. Długość tego drugiego pola zależy od pierwszych czterech bitów typu nano. Wiadomości UDM a przede wszystkim SMM w dużej mierze składają się właśnie z nanoinstrukcji. Jest to bowiem bardzo wygodna forma pozwalająca na budowanie właściwie dowolnych instrukcji czy rozkazów oraz przesyłanie niewielkich ilości danych.

W tabeli 4.3 przedstawiono nanoinstrukcje wykorzystywane przez zaproponowany system.

Tab. 4.3. Nanoinstrukcje wykorzystywane w zaproponowanym kryptosystemie.

Typ nanoinstrukcji	**Opis**	**Sposób wykorzystania i Uwagi**	**Stosowane w UDM**
01 – 08	Kasowanie klucza grupy 1 – 8	Nie można usunąć klucza 0 ani F. Nana 00 oraz 0F są błędne.	Nie
0A, 0B	Kasowanie klucza sesji A lub B		Nie
0C	Nanoinstrukcja wypełniająca	Pomiędzy nanem 0D oraz 0E. Minimalne dopełnienie do całkowitej długości wiadomości (patrz nano 11).	Tak
0D	Znacznik początku podpisu lub sekwencji CMAC	Nana te mogą przylegać do siebie. Nie zawsze konieczne jest wypełnienie nanem 0C.	Tak
0E	Znacznik końca nanoinstrukcji lub danych użytkownika		Tak
10	Typ wiadomości	1001 – unicast SMM 1002 – multicast SMM 1011 – broadcast UDM 1012 – multicast UDM 1013 – multicast UDM w kanale wirtualnym 1014 – unicast UDM w grupie 1015 – unicast UDM poza grupę	Tak
11	Długość wiadomości	Wielokrotność 128 bitów (max. 4 kB). Dobierana tak aby zminimalizować liczbę nan 0C.	Tak
12	Typ klucza użytego do szyfracji wiadomości (i kluczy AES – tylko dla SMM)	1200 – szyfracja kluczem 0 1201 – szyfracja kluczem grupy 1202 – szyfracja kluczem sesji 120F – szyfracja kluczem F	Tak
13	Typ klucza użytego do kontroli integralności wiadomości oraz do uwierzytelniania nadawcy	1300 – uwierzytelnianie RSA 1301 – uwierzytelnianie CMAC kluczem grupy 1302 – uwierzytelnianie CMAC kluczem sesji 130F – uwierzytelnianie CMAC kluczem F	Tak
14	Numery tablic skramblujących	14 XY: X – numer tablicy A Y – numer tablicy B Numery otrzymane w ostatnim odebranym SMM od CB muszą być stosowane przez moduł do tworzenia UDM.	Tak
15	Numer wykorzystanego wektora z tablicy A	Od 0 (nano 1500 lub 1600) do 255 (nano 15FF lub 16FF), bo po 256 wektorów w każdej z tablic. Numery wybierane losowo dla każdej nowej wiadomości UDM lub SMM.	Tak
16	Numer wykorzystanego wektora z tablicy B		Tak
1A	Identyfikator adresata w grupie	1A lub 1B + Identyfikator *InGroup*	Tak (Tylko w UDM)
1B	Identyfikator nadawcy w grupie		Tak (Tylko w UDM)
1F	Numer wirtualnego kanału transmisji	Numer bitu maski uprawnień, na którym znajdować musi się jedynka, aby możliwy był odbiór tej wiadomości (od 1 do 128	Tak (Tylko w UDM)

		zapisane binarnie)	
20	Zmiana daty wygaśnięcia aktywacji	2X VV ZZ: VV – rok ZZ – miesiąc Format zapisu jak we wzorze 4.1. Wygaśnięcie/Koniec ważności z ostatnim dniem miesiąca ZZ.	Nie
21 – 28	Zmiana daty wygaśnięcia uprawnień w grupie 1 – 8		Nie
2A, 2B	Zmiana daty ważności klucza sesji A lub B		Nie
2F	Pierwsze tajne Nano występujące po jawnym nagłówku wiadomości	2F XX YY: XX – *System ID* YY – *Provider ID* Kontrola poprawności deszyfracji.	Tak
3A	Identyfikator grupy adresata wiadomości	3A + Identyfikator *Group ID*	Tak
41 – 48	Zmiana identyfikatora użytkownika dla grupy 1 – 8	4X(1 – 8) + Identyfikator *User ID*	Nie
4A	Identyfikator modułu kryptograficznego adresata	4A lub 4B + Identyfikator *Module ID*	Tak
4B	Identyfikator modułu kryptograficznego nadawcy		Tak (tylko w UDM)
90	Aktualizacja maski uprawnień	90 XY + (1/2 maski uprawnień): X – numer grupy Y (0 lub 1) – pierwsza (0) lub druga (1) część maski uprawnień	Nie
91	Dodanie/Aktualizacja klucza	91 XY + (1/2 klucza): X – typ klucza (1 – 8, A lub B) Y (0 lub 1) – pierwsza (0) lub druga (1) część zaszyfrowanego klucza 91F…, 910… - postacie błędne	Nie
F0 – FF	Anulowanie uprawnień użytkowników w grupie*	FX + (nieaktywne identyfikatory *InGroup* w grupie określonej przez nano 3A): X – od 0 do maksymalnie F (dowolna liczba z tego przedziału) Ostatni identyfikator *InGroup* ewentualnie powtarzany w celu uzyskania F (15) bajtów treści nanoinstrukcji.	Nie (Tylko w multicast SMM)
*) Użytkownik, który posiada uprawnienia w danej grupie i odbierze nano FX wysłane w wiadomości skierowanej do tej właśnie grupy, w którym znajdzie się jego identyfikator *InGroup*, utraci uprawnienia do nadawania i odbioru w obrębie tejże grupy (automatyczne wyzerowanie maski uprawnień oraz skasowanie klucza grupy). Podobna sytuacja będzie mieć miejsce, gdy przekroczona zostanie data ważności uprawnień w danej grupie *GroupAuthorizationDate*.			

Każda wiadomość (SMM oraz UDM) rozpoczyna się jawnym nagłówkiem. Jest on zawsze długości 128 bitów. W jego skład wchodzą następujące pola (w kolejności od pierwszego):

- *System ID* (zawsze *0x86*) – jeden bajt;
- *Provider ID* – jeden bajt;
- Typ wiadomości (nano 10) – dwa bajty;
- Długość wiadomości (nano 11) – dwa bajty;
- Rodzaj klucza użytego do szyfracji (nano 12) – dwa bajty;
- Rodzaj klucza użytego do kontroli integralności wiadomości oraz do uwierzytelniania nadawcy (nano 13) – dwa bajty;

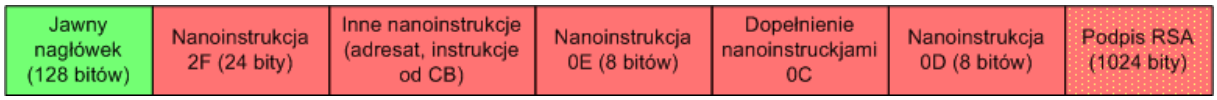
- Numery tablic skramblujących (nano 14) – dwa bajty;
- Numer wykorzystanego wektora z tablicy A (nano 15) – dwa bajty;
- Numer wykorzystanego wektora z tablicy B (nano 16) – dwa bajty.

Niezależnie również od typu wiadomości szyfracja zrealizowana jest w sposób właściwie identyczny. Wykorzystuje się do tego celu wspomniany wcześniej 128 bitowy zmodyfikowany algorytm AES-CTR (patrz punkt 4.5.1). Szyfracji podlega wiadomość bez nagłówka, ale wraz z wyznaczonym wcześniej i dodanym do niej podpisem. Różnice pomiędzy szyfracją wiadomości SMM oraz UDM przedstawione zostaną w punktach (4.6.1 oraz 4.6.2).

Poniżej przedstawiono sposób tworzenia oraz strukturę obu typów wykorzystywanych wiadomości.

4.6.1. Wiadomości SMM

Podstawowa struktura wiadomości SMM została przedstawiona na rysunku 4.6.



Rys. 4.6. Struktura wiadomości SMM.

Wiadomości SMM szyfrowane są z wykorzystaniem kluczy grup (*GroupKey 1 – 8*) lub klucza głównego (*MasterKey 0*) – patrz nano 12 w jawnym nagłówku. Teoretycznie istnieje możliwość wykorzystania również na przykład kluczy sesji, ale w przypadku wiadomości SMM należ unikać tego typu rozwiązań.

Uwierzytelnianie oraz kontrola integralności w wiadomościach SMM muszą być zawsze realizowane z wykorzystaniem uproszczonego podpisu RSA (patrz punkt 4.5.3 oraz nano 13).

Adresowanie wiadomości SMM zależne jest od jej typu. Dla wiadomości typu 1001 (unicast SMM) stosuje się nano 4A, które musi wystąpić zaraz po nano 2F.

W przypadku wiadomości typu 1002 (multicast SMM) stosuje się nano 3A, występujące również zaraz po nano 2F.

W wiadomościach SMM nie podaje się nadawcy (oczywisty – CB realizacji systemu o danym *Provider ID*).

Po nano 3A lub 4A wystąpić mogą już bardzo różne typy nanoinstrukacji (patrz tabela 4.3). Sposób formułowania instrukcji i rozkazów jest w dużej mierze dowolny i zależny od danej realizacji.

Przy tworzeniu wiadomości SMM należy pamiętać o kilku zasadach:

- Całkowita aktywacja modułu nie powinna odbywać się z wykorzystaniem jednej wiadomości SMM;
- Z wykorzystaniem klucza *MasterKey 0* przesyłane powinny być tylko informacje, których nie można wysłać z wykorzystaniem kluczy grup (np. same właśnie klucze grup);
- Klucze sesji powinny mieć dość długą ważność (około miesiąca);
- Należy minimalizować liczbę wystąpień nano 0C – trzeba dobierać odpowiednią długość wiadomości;
- Należy unikać zmian numerów tablic skramblujących (nano 14);

- Każdy moduł, który odbierze nano 14 musi potem tworzyć wiadomości UDM z wykorzystaniem zdefiniowanych w tym nano numerów tablic.

4.6.2. Wiadomości UDM

Podstawowa struktura wiadomości UDM została przedstawiona na rysunku 4.7.

Rys. 4.7. Struktura wiadomości UDM.

Wiadomości UDM szyfrowane są z wykorzystaniem kluczy sesji (SessionKey *A, B*) lub klucza F (*SessionBasicKey F*) – patrz nano 12 w jawnym nagłówku.

Uwierzytelnianie oraz kontrola integralności w wiadomościach UDM realizowane są z wykorzystaniem zmodyfikowanego algorytmu CMAC (patrz punkt 4.5.2). Stosuje się do tego celu klucze grup, sesji lub klucz F (patrz nano 13).

Sposób adresowania oraz przedstawiania nadawcy w wiadomościach UDM zależny jest od ich typu. Adresat występuje zaraz po nano 2F, a nadawca po adresacie. Możliwe schematy adresowania przedstawiono w tabeli 4.4.

Tab. 4.4. Schematy adresowania wiadomości UDM.

Typ wiadomości UDM	**Adresat**	**Nadawca**
1011 – broadcast UDM	Brak	4B + *Module ID*
1012 – multicast UDM	3A + *Group ID*	1B + *InGroup ID*
1013 – multicast UDM w kanale wirtualnym	3A + *Group ID*	1B + *InGroup ID* 1F + numer wirtualnego kanału
1014 – unicast UDM w grupie	3A + *Group ID* 1A + *InGroup ID*	1B + *InGroup ID*
1015 – unicast UDM poza grupę	4A + *Module ID*	4B + *Module ID*

Po nadawcy występuje nano 0E a następnie dane użytkownika. Po nich kolejne nano 0E oraz wypełnienie nanami 0C. Na końcu wiadomości występuje nano 0D oraz 128-bitowa sekwencja CMAC.

Przy tworzeniu wiadomości UDM należy:

- Korzystać przy szyfracji z kluczy A oraz B (unikać klucza F);
- Do tworzenia wiadomości UDM nie można stosować kluczy sesji, które utraciły swą ważność;
- Kiedy tylko to możliwe korzystać z kluczy grup do wyznaczania sekwencji CMAC (nie możliwe w wiadomościach typu 1011 oraz 1015);
- Minimalizować liczbę wystąpień nano 0C;
- Nie można tworzyć wiadomości UDM i transmitować w kanale wirtualnym, do którego nie ma się uprawnień (moduł kryptograficzny powinien automatycznie blokować taką możliwość);
- Moduł nieaktywny nie może nadawać ani odbierać żadnych wiadomości UDM.

Zarówno dekodowanie wiadomości SMM jak i tworzenie oraz dekodowanie wiadomości UDM powinno być automatycznie realizowane przez moduł kryptograficzny. Użytkownik powinien mieć tylko możliwość wyboru typu wysyłanych wiadomości UDM, adresata tych wiadomości oraz ewentualnie numeru kanału wirtualnego. Użytkownik powinien być również informowany (w postaci np. pewnego dziennika zdarzeń) o statusie modułu po otrzymaniu każdej wiadomości UDM czy SMM (patrz tabela 4.2).

Podsumowanie

Niniejsza praca stanowi pierwszy etap na drodze do stworzenia w pełni funkcjonalnego bloku kryptograficznego oraz modelu centrum bezpieczeństwa, które będą mogły być wykorzystywane do bezpiecznej transmisji danych w kanale krótkofalowym w oparciu o modemy HF zrealizowane w technologii radia programowalnego.

W opracowaniu tym zawarto podstawowe informacje związane z ogólnie rozumianą kryptografią i takie, które pozwolą na późniejszą realizację nowego kryptosystemu umożliwiającego bezpieczną transmisję danych w paśmie HF. Zawarto tu między innymi opisy wykorzystywanych algorytmów kryptograficznych związanych zarówno z szyfracją jak i uwierzytelnianiem nadawcy oraz zapewnianiem integralności przesyłanych wiadomości.

W pracy tej scharakteryzowano również kanał krótkofalowy, który stanowić będzie swoiste medium transmisyjne, w którym funkcjonować będzie zaproponowany kryptosystem. Przedstawiono tu zatem aspekty propagacyjne oraz wskazano trudności realizacyjne jakie nakłada kanał HF w związku z proponowanym rozwiązaniem.

Kluczowa jednak część pracy, to dokumentacja techniczna zaproponowanego kryptosystemu. Zawarto w niej opis założeń samego systemu oraz przedstawiono jego strukturę. Dokonano szczegółowej analizy dwóch podstawowych jednostek funkcjonalnych systemu (centrum bezpieczeństwa oraz modułu kryptograficznego) oraz zaproponowano swoisty protokół transmisyjny umożliwiający komunikację między tymi elementami. Uwidoczniono typy i struktury przesyłanych wiadomości oraz wymagania odnośnie ich kodowania. Stworzono również zasady zarządzania całym systemem bezpieczeństwa, wprowadzając hierarchizację kluczy oraz identyfikatorów jak również określono zasady ich dystrybucji, aktualizacji oraz wykorzystania w systemie.

W pracy zaproponowano dodatkowo autorskie modyfikacje powszechnie stosowanych algorytmów kryptograficznych, które dostosowano do potrzeb nowego kryptosystemu. Dzięki takiemu podejściu nowy system bezpieczeństwa realizować będzie właściwie wszystkie mechanizmy kryptograficzne i pozwali w przyszłości na w pełni bezpieczną transmisję.

W dalszych etapach zaprezentowany system zostanie zaimplementowany (najpierw w postaci procedur) i przetestowany pod względem spełnienia założeń początkowych (etap II), a następnie stworzone zostaną w pełni funkcjonalne: blok modułu kryptograficznego oraz model centrum bezpieczeństwa (etap III). Elementy te zrealizowane zostaną w technologii SDR i dzięki temu możliwe będzie ich wykorzystanie we współpracy z istniejącymi modemami HF wykorzystującymi radio programowalne.

Całość niniejszej pracy (jak i kolejne jej etapy) stanowi dodatkowo wkład do realizowanego przewodu doktorskiego na temat „Badania i analiza problemu bezpiecznej transmisji danych w paśmie krótkofalowym”.

Bibliografia

[1] Sutton R. J., *Bezpieczeństwo telekomunikacji*, WKŁ, Warszawa 2004.

[2] Pieprzyk J., Hardjono T., Seberry J., Teoria bezpieczeństwa systemów komputerowych, HELION.

[3] National Institute of Standards and Technology (NIST), *Data Encryption Standard (DES),* 1999.

[4] Federal Information Processing Standards (FIPS), PUB 197, *Advanced Encryption Standard,* 2001.

[5] National Institute of Standards and Technology (NIST), *Recommendation for Block Cipher Modes of Operation*, 2001.

[6] PKCS - RSA Laboratories, *RSA Cryptography Standard,* 2002.

[7] Federal Information Processing Standards (FIPS), PUB 180-2, *Secure Hash Standard,* 2002.

[8] R. Rivest, Network Working Group, *The MD5 Message-Digest Algorhitm,* 1992.

[9] National Institute of Standards and Technology (NIST), *Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*, 2005.

[10] Katulski R., *Materiały pomocnicze do przedmiotu Systemy Radiokomunikacyjne*.

[11] Stefański J., Bronk K., Lipka A., Radziwanowski M., *Szybka transmisja danych w paśmie krótkofalowym; Etap 3: Opracowanie platformy programowej modemu,* Instytut Łączności, Warszawa 2007.

[12] Stefański J., Bronk K. Niski R., Radziwanowski M., *Szybka transmisja danych w paśmie krótkofalowym; Etap 2: Opracowanie uniwersalnej platformy sprzętowej modemu*, Instytut Łączności, Warszawa 2006.

[13] Stefański J., Gencza S., Niski R., Radziwanowski M., *Szybka transmisja danych w paśmie krótkofalowym; Etap 1: Opracowanie pakietu symulującego pracę toru nadawczo-odbiorczego modemu w krótkofalowym kanale radiowym*, Instytut Łączności, Warszawa 2005.

[14] STANAG 4539 (Edition 1), *Technical Standards for Non-hopping HF Communications Waveforms*, NATO 2000.

[15] MIL-STD-188-110B, *Interoperability and Performance Standards for Data Modems*, Departament of Defense Interface Standard, July 2004.

[16] Bard J., Kovarik Jr. V. J., *Software Defined Radio, The Software Communications Architecture*, John Wiley & Sons, 2007.

[17] Bronk K., Stefański J., *Software Defined HF Data Modem*, Zeszyty Naukowe Akademii Marynarki Wojennej w Gdynia, 2007, str. 53-59.

[18] Bronk K., Stefański J., *Zdefiniowany programowo modem krótkofalowy dla potrzeb radiokomunikacji morskiej*, Zeszyty Naukowe Wydziału Elektroniki, Telekomunikacji i Informatyki Politechniki Gdańskiej; Radiokomunikacja, Radiofonia i Telewizja, nr 1, 2007, str. 113-116.

[19] Federal Information Processing Standards (FIPS), PUB 198, *The Keyed-Hash Message Authentication Code (HMAC)*, 2002.

[20] Federal Information Processing Standards (FIPS), PUB 186-2, *Digital Signature Standard (DSS),* 2000.

[21] MediaCrypt, *International Data Encryption Algorithm (IDEA).*

[22] Kaukonen K., Thayer R., *A Stream Cipher Encryption Algorithm "Arcfour" (RC4),* Internet-Draft, 1999.

[23] Standards for Efficient Cryptography, *Elliptic Curve Cryptography*, 2000.

[24] IEEE Standard for Information technology, Telecommunications and information exchange between systems, Local and metropolitan area networks, Specific requirements, *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 2007.

[25] Specification of the Bluetooth System, *Bluetooth specification version 2.1 + edr*, 2007.

[26] 3GPP Standards, 35 Series, *Security algorithms*.

[27] Federal Information Processing Standards (FIPS), PUB 140-2, *Security Requirements for Cryptographic Modules*, 2002.

[28] Stinson D. R., *Kryptografia w teorii i praktyce*, WNT, Warszawa 2005.

[29] National Institute of Standards and Technology (NIST), *Recommendation for Key Management – Part 1: General (Revised)*, 2007.