Waldemar NOWAKOWSKI, Zbigniew ŁUKASIK, Tomasz CISZEWSKI

BEZPIECZEŃSTWO TRANSMISJI DANYCH W SYSTEMACH STEROWANIA RUCHEM KOLEJOWYM

Streszczenie

Systemy sterowania ruchem kolejowym są systemami związanymi z bezpieczeństwem, a tym samym muszą odpowiadać wysokim wymaganiom jakościowym i niezawodnościowym. Wymagania te zawarte są w dokumentach normatywnych, w tym głównie w normach CENELEC (European Committee for Electrotechnical Standardization). W artykule przedstawiono zagrożenia występujące w bezprzewodowych systemach transmisji danych oraz opisano metody przeciwdziałania tym zagrożeniom, w kontekście wykorzystania ich w rozproszonych systemach sterowania ruchem kolejowym.

WSTĘP

Współczesne systemy sterowania ruchem kolejowym (srk) powszechnie wykorzystują nowoczesne technologie, w tym m.in. teleinformatykę do przetwarzania i przesyłania danych [3, 5, 7, 8]. Jednym z aktualnych obszarów badawczych jest zapewnienie bezpieczeństwa transmisji danych w rozproszonych systemach sterowania ruchem kolejowym wykorzystujących łączność radiową [1, 9, 10]. Powstające wówczas zagrożenia bezpieczeństwa są wynikiem m.in. nieznanej liczby użytkowników, którzy mogą chcieć uzyskać dostęp do sieci oraz nieznanej liczby oraz rodzaju sprzętu, który może zostać włączony do sieci. Stwarza to potencjalne zagrożenie dla bezpieczeństwa systemów srk, głównie z faktu możliwości pojawienia się danych o nieznanych formatach, jak również nieznanych ilościach, a także możliwości wystąpienia ataków sieciowych ze strony nieautoryzowanych użytkowników. Dlatego też niezbędne jest spełnienie wymagań zawartych w dokumentach normatywnych, w tym głównie w normach CENELEC (*European Committee for Electrotechnical Standardization*) [2, 11].

1. ANALIZA ZAGROŻEŃ TRANSMISJI DANYCH ORAZ METOD PRZECIWDZIAŁANIA ZAGROŻENIOM

Bezprzewodowa transmisja danych w rozproszonych systemach srk wykorzystuje otwarty układ transmisyjny, a tym samym bezpieczeństwo wymiany informacji w tych systemach należy oprzeć na następujących działaniach:

- podejściu do systemu transmisji jako systemu niezaufanego, niezależnie od tego jakie stosuje on wewnętrzne zabezpieczenia,
- bezpiecznych funkcjach transmisyjnych,
- bezpiecznych funkcjach dostępu.

Głównym zagrożeniem bezpieczeństwa tych systemów, wynikającym z niezaufanego systemu transmisyjnego, jest niepowodzenie w uzyskaniu przez odbiorcę ważnego i autentycznego telegramu. Stan taki może być spowodowany przez: powtórzenie telegramu (*repetition*), skasowanie telegramu (*deletion*), utworzenie telegramu przez nieautoryzowanego nadawcę (*insertion*), zmianę kolejności telegramów (*resequence*), uszkodzenie telegramu (*corruption*), opóźnienie w odebraniu telegramu (*delay*), maskaradę (*masquerade*).

System transmisji radiowej wykorzystywany w systemie srk, pod względem oceny bezpieczeństwa należy traktować jako niezaufany. W celu ograniczenia zagrożeń ze strony otwartego układu transmisyjnego można wziąć pod uwagę następujące warunki: autentyczność telegramów, integralność telegramów, określony czas przesyłania telegramów, kolejność telegramów. Przyjmując jako podstawę w/w warunki norma PN-EN 50159 podaje szereg metod zapewnienia bezpieczeństwa danych w systemach z otwartym układem transmisji, które określono jako funkcje bezpieczeństwa [11]:

A. Numerowanie telegramów (*sequence number*).
B. Stosowanie w telegramach znaczników czasu (*time stamp*).
C. Zdefiniowanie maksymalnego czasu oczekiwania na odpowiedź (*time-out*).
D. Dodawanie do telegramów identyfikatora nadawcy i odbiorcy.
E. Stosowanie komunikatów zwrotnych (*freedback message*).
F. Wykorzystywanie procedur autoryzacji (*identification*).
G. Stosowanie kodów bezpieczeństwa (*safety code*).
H. Szyfrowanie danych (*cryptographics*).

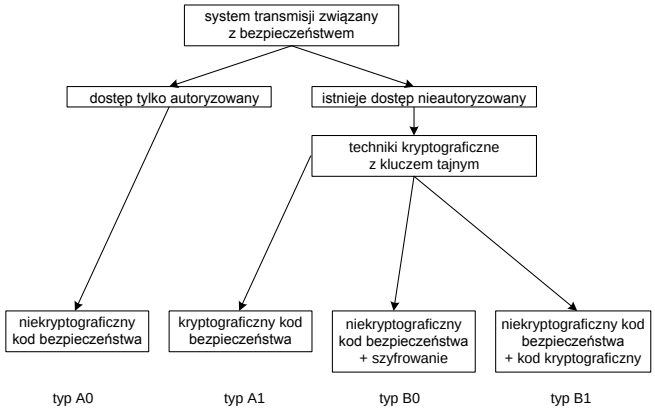
2. ZASADY WYBORU METOD OBRONY PRZEZ ZAGROŻENIAMI

Na podstawie normy PN-EN 50159 w tabeli 1 podano zestawienie zagrożeń i metod obrony (funkcji bezpieczeństwa).

Tab. 1. *Zestawienie zagrożeń i metod obrony [11]*

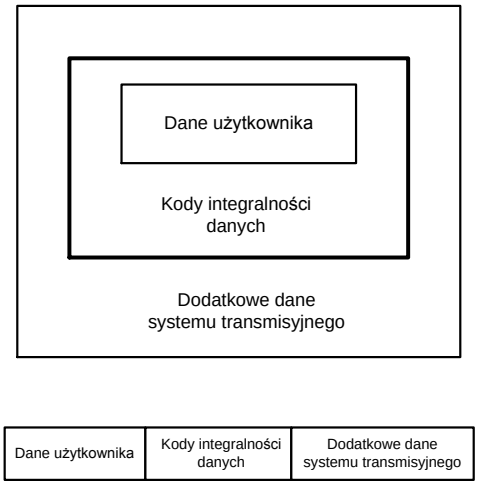
	A	B	C	D	E	F	G	H
Powtórzenie	X	X						
Skasowanie	X							
Brak autoryzacji	X			X	X	X		
Zmiana kolejności	X	X						
Uszkodzenie							X	X
Opóźnienie		X	X					
Maskarada					X	X		X

Analiza sposobu transmisji danych dla każdego z systemów bezpieczeństwa powinna odpowiedzieć na pytanie: czy możliwy jest nieautoryzowany dostęp? Jeśli w całym cyklu życia systemu wykluczymy nieautoryzowany dostęp wówczas nie musimy stosować technik kryptograficznych, a wyłącznie kody integralności danych, które zabezpieczą transmisję przed przypadkowymi błędami. Telegramy zabezpieczone w ten sposób oznaczane są w normie PN-EN 50159 jako typ A0 (rys. 1). Najczęściej takie rozwiązanie stosuje się w przypadku sieci lokalnych (LAN).



Rys. 1. *Metody zabezpieczenia transmisji dla systemów związanych z bezpieczeństwem [11]*

Model telegramu z kodem integralności danych przedstawiono na rys. 2.



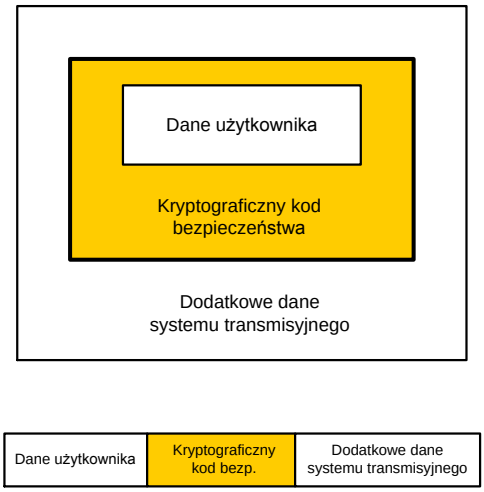
Rys. 2. *Model telegramu z kodem integralności danych (typ A0)*

Inne podejście należy zastosować, gdy przyjmiemy założenie braku pewności wykluczenia nieautoryzowanego dostępu. Wówczas zaleca się stosowanie technik kryptograficznych z użyciem tajnego klucza. Norma przewiduje w tym przypadku wykorzystanie technik kryptograficznych. Jednym z rozwiązań jest dodanie kryptograficznego kodu bezpieczeństwa np. w postaci zaszyfrowanego kodu integralności. Mówimy wówczas o telegramie typu A1. Model takiego telegramu dla tej metody przedstawiono na rys. 3.

Kolejny z możliwych sposobów ochrony danych polega na szyfrowaniu całej wiadomości, czyli danych i np. kodów integralności. Tego typu telegramy oznaczono symbolem B0. Model telegramu dla tej metody przedstawiono na rys. 4.

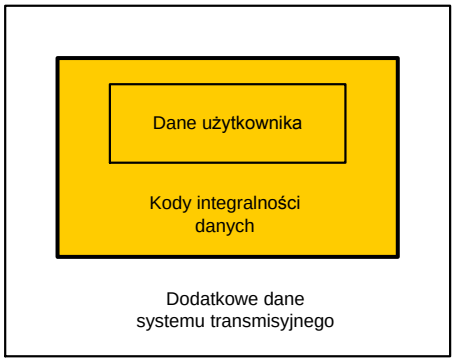
W metodzie z zaszyfrowaną wiadomością koszt obliczeniowy rośnie wraz ze wzrostem wielkości wiadomości. Bardzo ważnym zagadnieniem jest wówczas odpowiedni dobór algorytmu szyfrującego, który charakteryzuje się nie tylko dobrymi właściwościami zabezpieczającymi, ale również dużą wydajnością.

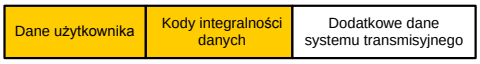
Ostatnim z typów telegramów przewidzianych w normie PN-EN 50159 jest typ B1, w którym oprócz danych występuje zarówno niezaszyfrowany kod integralności danych, jak również kryptograficzny kod bezpieczeństwa (rys. 5).



- Zaszyfrowane dane

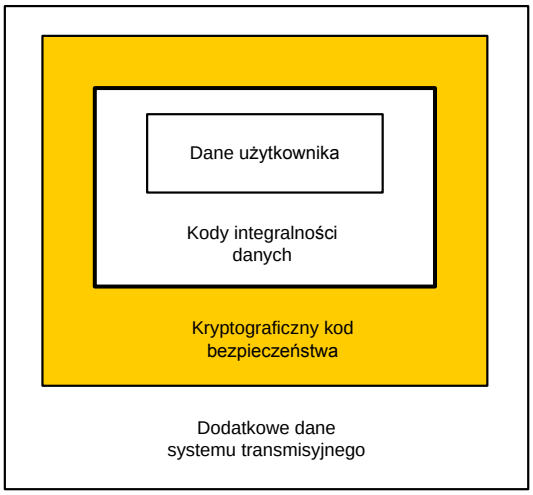
Rys. 3. *Model telegramu z kryptograficznym kodem bezpieczeństwa (typ A1)*





- Zaszyfrowane dane

Rys. 4. *Model telegramu z zaszyfrowaną wiadomością (typ B0)*



- Zaszyfrowane dane

Rys. 5. *Model telegramu z zaszyfrowaną wiadomością (typ B1)*

W przypadku telegramów typu A1 i B1 mamy do czynienia z algorytmem charakteryzującym się niskim kosztem obliczeniowym. Można powiedzieć, że koszt szyfrowania jest stały i nie rośnie wraz z wielkością wiadomości, gdyż zależy wyłącznie od wielkości kodu integralności (np. kodu CRC lub skrótu wiadomości). Jest to niewątpliwie zaleta tej metody. Należy jednak zwrócić uwagę, że nie zabezpieczamy w ten sposób poufności danych w otwartych układach transmisyjnych, a jedynie zabezpiecza je przed przypadkową lub celową modyfikacją.

Norma PN-EN 50159 podaje również zalecenia w zakresie doboru algorytmów wyznaczania kodów bezpieczeństwa w zależności od typów telegramów, które przedstawiono w tabeli 2.

Tab. 2. *Zestawienie kodów bezpieczeństwa dla różnych typów telegramów [11]*

Typ	Rodzaj modelu telegramu dla systemu bezpieczeństwa transmisji			
	A0	**A1**	**B0**	**B1**
CRC	R	US	-	R
MAC	R	HR	R	R
Hash code	R	US	HR	HR
Digital signature	R	R	R	R

Oznaczenia:
„HR" -oznacza, że ten typ kodu bezpieczeństwa jest bardzo polecany,
„R" -oznacza, że ten typ kodu bezpieczeństwa jest polecany,
„-" -oznacza, że ten typ kodu bezpieczeństwa nie ma żadnych rekomendacji, za czy przeciw użyciu,
„US "-oznacza, że ten typ kodu bezpieczeństwa nie jest zalecany.

W zakresie algorytmów szyfrowania w normie PN-EN 50159 zaleca się stosowanie sprawdzonych algorytmów szyfrowania takich jak DES. Oficjalnym następcą DES jest algorytm AES. Poziom bezpieczeństwa dla AES jest wyższy ze względu na dłuższy niż w DES klucz szyfrujący. W zakresie doboru algorytmów szyfrowania norma PN-EN 50159 nie zaleca stosowania trybu szyfrowania blokowego ECB w przypadku, gdy wielkości bloków danych wejściowych (przed zaszyfrowaniem) są większe niż po szyfrowaniu.

3. WYBRANE METODY OBRONY PRZED ZAGROŻENIAMI

3.1. Funkcje skrótu

Pod pojęciem funkcji skrótu *h* (*hash function*) rozumie się, łatwe do wyznaczenia przekształcenie *d=h[M]* odwzorowujące wiadomość *M* o dowolnej, skończonej długości, w ciąg bitów *d* o określonej, stałej długości. Funkcje skrótu posiadają następujące własności [6, 12]:

- kompresja: oznaczająca, że rozmiar skrótu musi być mniejszy od rozmiaru samej wiadomości |d|<|M|
- łatwość obliczeń: czas wielomianowy wyznaczenia h[M] dla dowolnego M
- odporność na „podmianę" argumentu: dla danego h[M] obliczeniowo trudne znalezienie M' takiego, że h[M]=h[M']
- odporność na „kolizję": obliczeniowo trudne znalezienie dwóch dowolnych argumentów *M≠M'* takiego, że *h[M]=h[M']*

Funkcje skrótu są jawne – nie kryją w sobie żadnych tajemnic. Bezpieczeństwo funkcji skrótu polega na jej jednokierunkowości. W żaden dostrzegalny sposób ciąg wyjściowy nie zależy od ciągu wejściowego. Zmiana wartości w dowolnym bicie ciągu wejściowego powoduje zmianę przeciętnie połowy bitów ciągu wyjściowego. Przy danej wartości skrótu jest praktycznie niewykonalne znalezienie ciągu wejściowego, który w wyniku daje taką samą wartość skrótu.

Funkcje skrótu mają szerokie zastosowanie praktyczne. Stosuje się je w schematach podpisu cyfrowego, do przechowywania haseł systemów operacyjnych czy też baz danych, do kontroli integralności wiadomości MIC (*Message Integrity Check*) (bez szyfrowania) oraz jako kody uwierzytelniające MAC (*Message Authentication Code*) (z szyfrowaniem). Przykładem funkcji skrótu może być MD-5 (*Message Digest*) lub SHA-1 (*Secure Hash Algorithm*).

Funkcja MD5

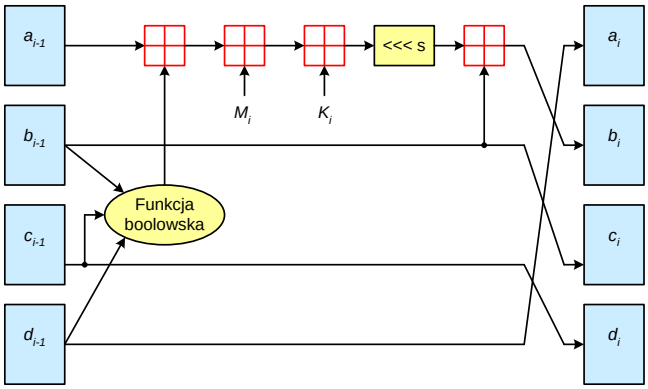
Algorytm funkcji skrótu MD5 (*Message Digest*) został opracowany przez Rona Rivesta jako unowocześniona wersja MD4 [12]. W wyniku działania funkcji MD5 otrzymujemy 128-bitowy skrót wiadomości. Proces wyznaczenia skrótu przebiega w trzech fazach:

- przekształcenia początkowe,
- pętla główna,
- obliczenia końcowe.

W pierwszej fazie, tekst jawny jest uzupełniany ciągiem bitów w taki sposób, aby był o 64 bity krótszy od wielokrotności 512. Ciąg uzupełniający składa się z zer i jedynki na końcu. Następnie dołącza się 64 bity określające pierwotną długość wiadomości. Po przygotowaniu ciągu wejściowego inicjowane są cztery zmienne łańcuchowe o wartościach:

- *A=0x01234567*
- *B=0x89abcdef*
- *C=0xfedcba98*
- *D=0x76543210*

i rozpoczyna się główna pętla algorytmu, która realizowana jest oddzielnie dla tylu 512-bitowych bloków, ile zawiera ich wiadomość. Cztery zmienne: *A, B, C, D* kopiowane są na cztery zmienne pomocnicze: *a, b, c, d*. Pętla główna składa się z czterech cykli, każdy po 16 operacji, czyli 64 kroków. W każdym kroku, do kolejnej zmiennej pomocniczej dodawana jest prosta funkcja boolowska trzech pozostałych zmiennych pomocniczych, a następnie dodawany jest jeden z 16-tu 32-bitowych fragmentów bloku wejściowego *Mi* oraz pewna stała *Ki* zależna od numeru kroku. Wynik jest przesuwany cyklicznie w prawo o liczbę bitów zależną od kroku, a potem sumowany z jedną ze zmiennych pomocniczych (rys. 6).



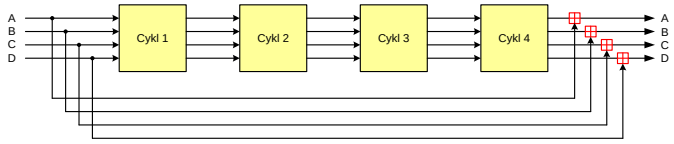
Rys. 6. *Schemat jednego kroku algorytmu MD5*

W każdym z cykli stosowana jest inna funkcja boolowska:

- w cyklu 1 (kroki 0 do 15)
  $F(X,Y,Z)=(X \wedge Y) \vee (\neg X) \wedge Z$
- w cyklu 2 (kroki 16 do 31)
  $G(X,Y,Z)=(X \wedge Z) \vee (Y \wedge (\neg Z))$
- w cyklu 3 (kroki 32 do 47)
  $H(X,Y,Z)=X \oplus Y \oplus Z$
- w cyklu 4 (kroki 48 do 63)
  $I(X,Y,Z)=Y \oplus (X \vee (\neg Z))$

Stała *Ki* dla kroku *i* jest częścią całkowitą liczby $2^{32}*abs(sin(i))$, przy czym *i* jest podane w radianach. W fazie trzeciej do zmiennych

pomocniczych: *a, b, c, d* dodawane są zmienne łańcuchowe: *A, B, C, D*. Następnie, z tak wyznaczonymi zmiennymi łańcuchowymi, rozpoczyna się przetwarzanie kolejnego bloku wiadomości (rys. 7). Wartość wyjściowa jest konkatenacją zmiennych: *A, B, C, D*.



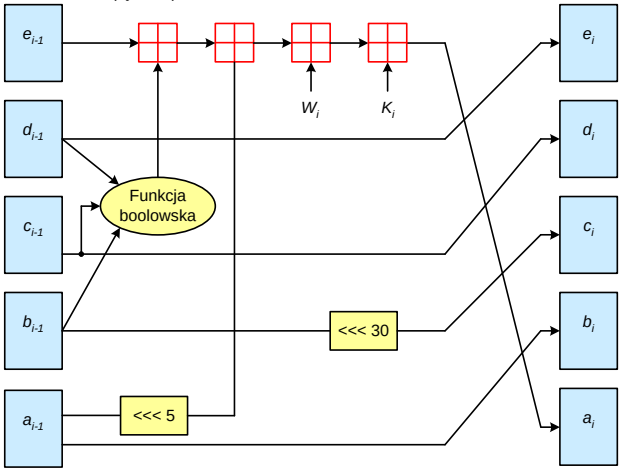
Rys. 7. *Schemat pętli głównej algorytmu MD5*

Funkcja SHA-1

Algorytm SHA-1 (*Secure Hash Algorithm*) opracowany został przez NIST, a jego pierwowzorem był algorytm MD5 [13]. SHA-1 ma wiele zastosowań m.in. jest stosowany w asymetrycznym algorytmie szyfrującym DSA (*Digital Signature Algorithm*) do generacji i weryfikacji cyfrowego podpisu. SHA-1 mimo, iż tworzy skrót 160-bitowy, jest bardzo podobny do algorytmu MD5. W pierwszej fazie, tekst jawny jest uzupełniany ciągiem bitów w taki sposób, aby był o 64 bity krótszy od wielokrotności 512. Ciąg uzupełniający składa się z zer i jedynki na końcu. Następnie dołącza się 64 bity określające pierwotną długość wiadomości, jak w algorytmie MD5. Po przygotowaniu ciągu wejściowego inicjowanych jest pięć zmiennych łańcuchowych o wartościach:

- *A=0x67452301*
- *B=0xefcdab89*
- *C=0x98badcfe*
- *D=0x10325476*
- *E=0xc3d2e1f0*

i rozpoczyna się główna pętla algorytmu, która realizowana jest oddzielnie dla tylu 512-bitowych bloków, ile zawiera ich wiadomość. Pięć zmiennych: *A, B, C, D, E* kopiowanych jest na pięć zmiennych pomocniczych: *a, b, c, d, e*. Pętla główna składa się z czterech cykli, każdy po 20 operacji, czyli 80 kroków. Pojedynczy krok ma następujący przebieg. Zmienne *b, c* oraz *d* są przekształcane przez funkcję boolowską, a następnie dodawane do zmiennej *e*. Wynik tej operacji jest sumowany ze zmienną *a* przesuniętą o pięć bitów w lewo. Do wyniku dodawany jest najpierw blok wiadomości *Wi*, a następnie stała *Ki* zależna od numeru kroku. Zmienna *b* jest przesuwana o 30 bitów w lewo (rys. 8).



Rys. 8. *Schemat jednego kroku algorytmu SHA-1*

W każdym z cykli stosowana jest inna funkcja boolowska:

- w cyklu 1 (kroki 0 do 19)
  $F(X,Y,Z)=(X \wedge Y) \vee ((\neg X) \wedge Z)$
- w cyklu 2 (kroki 20 do 39)
  $G(X,Y,Z)= X \oplus Y \oplus Z$
- w cyklu 3 (kroki 40 do 59)
  $H(X,Y,Z)=(X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$
- w cyklu 4 (kroki 60 do 79)
  $I(X,Y,Z)= X \oplus Y \oplus Z$

Jeden blok wiadomości składający się z 16-tu 32-bitowych fragmentów bloku wejściowego *Mi* jest przekształcany w 80 słów 32-bitowych *Wi* przy zastosowaniu poniższych zależności:

$W_i=M_i$ *dla* $i=0,\ldots,15$

$W_i=(W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16})<<<1$ *dla* $i=16,\ldots,79$

Ki są wielkościami stałymi:

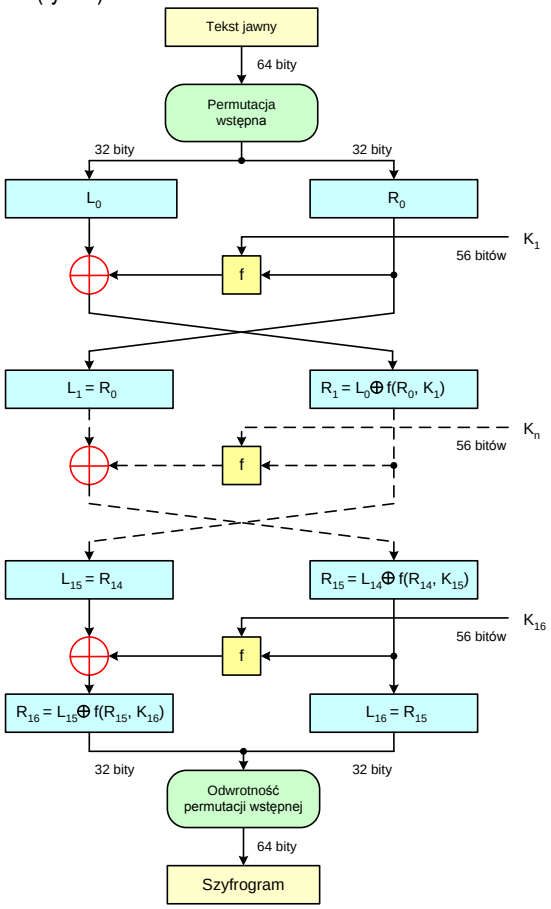
- $Ki = 0x5a827999$ dla $i = 0,\ldots,19$
- $Ki = 0x6ed9eba1$ dla $i = 20,\ldots,39$
- $Ki = 0x8f1bbcdc$ dla $i = 40,\ldots,59$
- $Ki = 0xca62c1d6$ dla $i = 60,\ldots,79$

Po zakończeniu ostatniego cyklu zmienne pomocnicze: *a, b, c, d, e* są dodawane do zmiennych: *A, B, C, D, E*, po czym rozpoczyna się szyfrowanie kolejnego bloku wiadomości. Wynikiem działania całego algorytmu jest konkatenacja zmiennych: *A, B, C, D, E*.

3.2. Techniki kryptograficzne

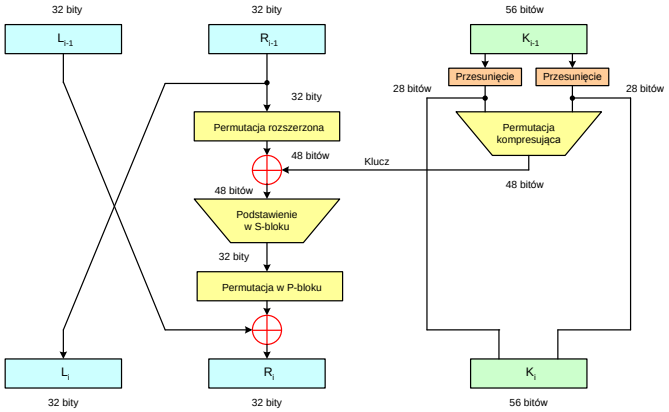
Algorytm DES

DES (*Data Encryption Standard*) jest algorytmem symetrycznym szyfrującym kolejne bloki o długości 64 bitów z wykorzystaniem 56-bitowego klucza [12, 13]. Algorytm ma postać sieci Feistela z 16 rundami (rys. 9).



Rys. 9. *Schemat algorytmu DES*

Po wstępnej permutacji blok wejściowy jest dzielony na dwa 32-bitowe bloki *L0, R0*. Następnie wykonywanych jest 16 rund, w których naprzemiennie szyfrowane są 32-bitowe bloki danych. Po szesnastej rundzie bloki *R16* i *L16* są łączone i wykonywana jest permutacja będąca odwrotnością permutacji wstępnej. Schemat pojedynczej rundy algorytmu DES przedstawiono na rys. 10.



Rys. 10. *Schemat pojedynczej rundy algorytmu DES*

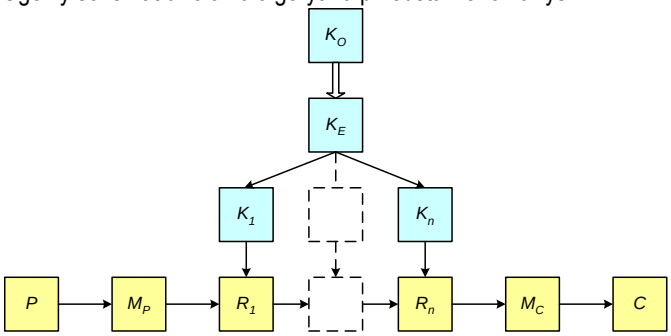
Szyfrowanie polega na wykonaniu, w każdej rundzie, kilku operacji oznaczonych na rys. 9 jako funkcja *f*. Najpierw 32-bitowa prawa połowa bloku danych jest poddawana rozszerzeniu do 48-bitów za pomocą permutacji z rozszerzeniem i sumowana „modulo 2" z 48-bitowym kluczem wyliczanym indywidualnie dla każdej rundy. Następnie w wyniku podstawienia do 8 S-bloków otrzymujemy blok 32-bitowy. W tym celu 48 bitów na wejściu jest dzielone na 8 wielkości 6-bitowych, które są indeksami każdego z S-bloków. Ponieważ S-bloki przechowują wielkości 4-bitowe, otrzymujemy na wyjściu blok 32-bitowy. Kolejną czynnością realizowaną przez funkcję *f* jest permutacja w P-bloku, w wyniku której każdy bit wejściowy odwzorowywany jest na bit wyjściowy, a tym samym blok wyjściowy pozostaje 32-bitowy. Blok ten jest następnie sumowany „modulo 2" z lewą połową bloku danych. Klucze szyfrujące kolejnych rund są inne, przy czym klucz dla i-tej rundy wykorzystuje klucz i-1 rundy. Przygotowanie klucza polega na rozdzieleniu 56-bitowego klucza na dwie 28-bitowe wartości, a następnie ich przesunięciu w lewo o 1 lub 2 bity, zależnie od numeru rundy. Po tej czynności wybieranych jest 48 z 56-bitów w wyniku działania permutacji z kompresją. Utworzona w ten sposób wartość stanowi klucz danej rundy, a zarazem stanowi podstawę do wyznaczenia klucza kolejnej rundy.

Algorytm AES

Algorytm AES (*Advanced Encryption Standard*), znany również jako Rijndael, zastąpił w 2000r. DES'a [12, 13]. AES może pracować ze stałą długością bloku danych (128-bitów) i zmienną długością klucza (128-, 192- i 256-bitów). Liczba wykonywanych rund jest zmienna i zależy od długości klucza:

128-bitów -10 rund,
192-bitów -12 rund,
256-bitów -14 rund.

Ogólny schemat działania algorytmu przedstawiono na rys. 11.



Rys. 11. *Schemat algorytmu AES*

W pierwszym etapie 128-bitowy blok tekstu jawnego wpisywany jest w macierz stanu *MP*. Ponieważ elementy macierzy są 8-bitowe ma ona wymiar 4x4. Na tym etapie klucz szyfrujący *KO* przekształcany jest w klucz rozszerzony *KE*, o długości równej iloczynowi liczby rund powiększonej o jeden. W każdej rundzie użytych jest cztery 32-bitowe słowa z klucza rozszerzonego. Pojedyncza runda algorytmu AES (oprócz ostatniej, w której brak *MixColumns*) składa się czterech operacji: *SubBytes*, *ShiftRows*, *MixColumns* oraz *AddRoudKey*. Operacja *SubBytes* to przekształcenie bloku wejściowego przy użyciu S-bloku będącego macierzą o wymiarach 16x16. Bajty wynikowe odnajdowane są w *S*-bloku pod adresem wiersza i kolumny utworzonym z 4 młodszych i 4 starszych bitów kolejnych bajtów macierzy stanu. Transformacja *ShiftRows* przesuwa w lewo drugi, trzeci i czwarty wiersz macierzy stanu odpowiednio o 1, 2 i 3 bajty. W kolejnej fazie, jaką jest *MixColumns* każda kolumna macierzy stanu mnożona jest przez specjalną macierz *C(x)*:

$$C(x) = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

Dopiero operacja *AddRoundKey* przeprowadza właściwe szyfrowanie. Macierz otrzymana w wyniku operacji *MixColumns* jest XOR-owana z kluczem rundy. Macierz wynikowa stanowi dane wejściowe dla kolejnej rundy, a gdy jest to runda ostatnia, jest macierzą szyfrogramu *MC*.

PODSUMOWANIE

Wymogi bezpieczeństwa, jakie muszą być spełnione w procesie wymiany danych przez systemy sterowania ruchem kolejowym, w celu osiągnięcia przez cały system zakładanego poziomu nienaruszalności bezpieczeństwa SIL (*Safety Integrity Level*), określa norma PN-EN 50159. Norma ta definiuje podstawowe zagrożenia oraz metody przeciwdziałania zagrożeniom zarówno dla zamkniętych, jak i otwartych układów transmisji danych. W artykule zostały przedstawione metody realizacji bezpieczeństwa dla bezprzewodowej transmisji danych w rozproszonych systemach sterowania ruchem kolejowym. Przedstawiono zagrożenia oraz opisano metody przeciwdziałania tym zagrożeniom, w tym głownie metody kryptograficzne: funkcje skrótu oraz algorytmy szyfrowania blokowego.

BIBLIOGRAFIA

1. Aguado M., Jacob E., Saiz P. et al., *Railway signaling systems and new trends in wireless data communication*, 62nd IEEE Vehicular Technology Conference, Dallas, USA, IEEE VTS Vehicular Technology Conference Proceedings, pp. 1333-1336, 2005.
2. Briso C., Alonso J.I., Requirements of wireless communications for control and operation of railway systems, 46th Annual Congress of the Federation-of-Telecommunications-Engineers-of-the-European-Union (FITCE), Warsaw, Poland, 2007, Journal of the Institute of Telecommunications Professionals, vol. 1, pp. 13-18, part 1.
3. Liem M., Mendiratta V.B., Mission critical communication networks for railways, Bell Labs Technical Journal, vol. 16, issue 3, pp. 29-46, 2011.
4. Bester L., Analysis of Wireless Transmission in Additional Warning System for Drivers at the Railway Level Crossings, 19th International Scientific Conference on Transport Means, Kaunas, Lithuania, 2015, Transport Means - Proceedings of the International Conference, pp. 532-536.

5. Łukasik Z., Nowakowski W., Bezprzewodowe systemy sterowania ruchem kolejowym, Infrastruktura Transportu, nr 4/2013, str. 22-25, 2013.
6. Nowakowski W., Information security and privacy protection in emergency management software systems, Logistyka 4/2015, str. 8072-8077, 2015.
7. Nowakowski W., Warchoł A., Nowoczesne systemy sterowania i diagnostyki na przykładzie LCS Drzewica, Zeszyty Naukowo-Techniczne Stowarzyszenia Inżynierów i Techników Komunikacji w Krakowie, Seria: Materiały Konferencyjne, Wydanie 95 z. 154, str. 453-465, 2010.
8. Nowakowski W., Szczygielski M., Analiza bezpieczeństwa transmisji w systemie zabezpieczenia przejazdów SZP-1, XVI Międzynarodowa Konferencja „TransComp", Zakopane 2012r.
9. Łukasik Z., Nowakowski W., Wymiana informacji w systemach związanych z bezpieczeństwem, Logistyka 6/2008, 2008.
10. Łukasik Z., Nowakowski W., Kuśmińska-Fijałkowska A., Zarządzanie bezpieczeństwem infrastruktury krytycznej, Logistyka 4/2014, str. 758-763, 2014.
11. PN-EN 50159:2011, Zastosowania kolejowe - Systemy łączności, sterowania ruchem i przetwarzania danych - Łączność bezpieczna w systemach transmisyjnych, PKN, 2011.
12. Schneier B., Kryptografia dla praktyków. Protokoły, algorytmy i programy źródłowe z języku C, PWN, Warszawa, 2002.
13. Stallings W., Kryptografia i bezpieczeństwo sieci komputerowych. Matematyka szyfrów i techniki kryptologii, Helion, Gliwice, 2011.

SECURITY OF DATA TRANSMISSION IN RAILWAY TRAFFIC CONTROL SYSTEMS

Abstract

Railway traffic control systems are safety related systems, and thus must meet high quality and reliability requirements. These requirements are defined in normative documents, mainly in CENELEC (European Committee for Electrotechnical Standardization) standards. The article presents hazards that may appear in the wireless systems and describes the methods to prevent these threats, in the context of their use in distributed railway traffic control systems.

Autorzy:
dr inż. **Waldemar Nowakowski** – Uniwersytet Technologiczno-Humanistyczny im. Kazimierza Pułaskiego w Radomiu, Wydział Transportu i Elektrotechniki, 26-600 Radom, ul. Malczewskiego 29, e-mail: w.nowakowski@uthrad.pl
prof. dr hab. inż. **Zbigniew Łukasik** – Uniwersytet Technologiczno-Humanistyczny im. Kazimierza Pułaskiego w Radomiu, Wydział Transportu i Elektrotechniki, 26-600 Radom, ul. Malczewskiego 29, e-mail: z.lukasik@uthrad.pl
dr inż. **Tomasz Ciszewski** – Uniwersytet Technologiczno-Humanistyczny im. Kazimierza Pułaskiego w Radomiu, Wydział Transportu i Elektrotechniki, 26-600 Radom, ul. Malczewskiego 29, e-mail: t.ciszewski@uthrad.pl