ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 2000
Seria: AUTOMATYKA z. 130 Nr kol. 1475

Gizela JAKUBOWSKA, Jerzy PEJAŚ
Politechnika Szczecińska

FORMALNE METODY ANALIZY KRYPTOGRAFICZNYCH PROTOKOŁÓW UWIERZYTELNIANIA PRZY ZASTOSOWANIU KOLOROWANYCH SIECI PETRIEGO

Streszczenie. W artykule została przedstawiona propozycja zastosowania kolorowanych sieci Petri do opisu modeli protokołów uwierzytelniania, oraz modeli intruzów, atakujących analizowany protokół. Narzędzia będące rezultatem ich zastosowania wymagają jednak aktywnej współpracy weryfikatora, który na ich podstawie jest w stanie ocenić stopień bezpieczeństwa protokołu oraz budować bazę wiedzy intruza.

FORMAL METHODS OF ANALYZING CRYPTOGRAPHIC PROTOCOLS FOR AUTHENTICATION USING COLOURED PETRI NETS

Summary. In this paper we present an approach to model authenticating protocols and intruders inside analyzing system using Coloured Petri Nets. However, tools based on the CP-Nets need active human-verifier assistance, who can estimate a degree of security of analyzing protocol and build intruder's database of knowledge.

1. Wprowadzenie

Zanim omówione zostaną techniki stosowane przez analityków i projektantów protokołów do wykrywania luk w protokołach, pokażemy, jakie zagrożenie stwarza źle zaprojektowany protokół, oraz zaprezentujemy dwie klasy ataków na tego typu protokoły.

Rozpatrzmy protokół Needhama-Schroedera [3, 14] (NST), oparty na kluczach tajnych, znanych tylko upoważnionym stronom. Uczestniczą w nim trzy strony: A, B i S. S jest serwerem uwierzytelniającym, zaufaną trzecią stroną, A - podmiotem, który chce zainicjować i następnie prowadzić bezpieczną (poufną) wymianę informacji ze stroną B. Jeśli przyjmiemy, że każdy z kroków protokołu ma postać: A→B: wiadomość, zaś przez $\{M\}_K$ oznaczymy wiadomość M zaszyfrowaną kluczem symetrycznym K, wówczas protokół NST można przedstawić następująco:

1. $A \rightarrow S : A, B, N_a$
2. $S \rightarrow A : \{N_a, B, K_{ab}, \{K_{ab}, A\}_{K_{bs}}\}_{K_{as}}$
3. $A \rightarrow B : \{K_{ab}, A\}_{K_{bs}}$

4. $B \rightarrow A : \{N_b\}_{K_{ab}}$
5. $A \rightarrow B : \{N_b - 1\}_{K_{ab}}$

W protokole tym jedynie strona A nawiązuje połączenie z serwerem S, który dostarcza jej klucz sesyjny K_{ab}, certyfikat zawierający klucz sesyjny i identyfikator strony A, zaszyfrowany kluczem K_{bs} współdzielonym przez stronę B z S. Strona B odszyfrowuje certyfikat uzyskując klucz sesyjny. Klucz K_{as} jest znany jedynie stronom A i S, z kolei N_a oraz N_b są losowo dobieranymi ciągami bitów, które z założenia mają umożliwić stronom rozpoznawanie wiadomości należących do aktualnie prowadzonej sesji protokołu.

W rezultacie wykonania protokołu strony A i B są w posiadaniu klucza sesji K_{ab}, który może im następnie posłużyć do szyfrowania wymienianej informacji, dotyczącej np. poleceń przelewów pomiędzy kontami bankowymi.

W systemach sieciowych możliwe jest, że sesja (sesje) pomiędzy stronami A, B i S prowadzona według dowolnego protokołu będzie podsłuchiwana i rejestrowana przez intruza I. Załóżmy, że zarejestrował on jedną z takich sesji w całości, włącznie z wymianą poufnej informacji. Wykorzystując posiadaną informację, szyfrowaną kluczem sesji K_{ab}, intruz I może podjąć próbę złamania klucza K_{ab}. W przypadku sukcesu może wykorzystać ten fakt do zaburzenia kolejnej wymiany komunikatów realizowanych wg protokołu NST.

Załóżmy, że doszło do skompromitowania klucza K_{ab} i oznaczmy go jako K'_{ab}. Jeśli strony A i B będą realizowały nowy protokół NST, wówczas intruz może przechwycić wiadomość wysłaną w trzecim kroku i podstawić wiadomość, którą kiedyś przechwycił, o złamanym przez intruza kluczu K'_{ab}:

3. $A \rightarrow B : \{K_{ab}, A\}_{K_{bs}}$

Strona B odpowiada zgodnie z czwartym krokiem protokołu, intruz przechwytuje wiadomość, deszyfruje za pomocą klucza K'_{ab} i kończy protokół z B. Strona B jest przekonana, że dialog prowadziła i zakończyła ze stroną A. Od tego momentu intruz I może w sposób poufny wymieniać informację ze stroną B, która jest przekonana, że rozmawia z A.

Innym przykładem protokołu uwierzytelniania jest również protokół Needhama-Schroedera [11], wykorzystujący klucze publiczne stron uczestniczących w protokole (oznaczmy go przez NSP). Ma on następującą postać:

1. $A \rightarrow B : \{N_a, A\}_{K_b}$
2. $B \rightarrow A : \{N_a, N_b\}_{K_a}$
3. $A \rightarrow B : \{N_b\}_{K_b}$

W wyniku realizacji protokołu obie strony powinny uwierzytelnić się wzajemnie oraz ustalić znane tylko im wartości N_a i N_b. Protokół ten jest podatny na tzw. atak z osobą pośrodku (ang. *a middleperson attack*).

Załóżmy, że intruz (Charlie) siedzi pomiędzy A (Alicją) i B (Bobem). Podaje się za Charliego wobec Alicji, ale udaje Alicję wobec Boba. Protokół może przebiegać następująco:

1. $A \rightarrow C : \{N_a, A\}_{K_c}$
1'. $C \rightarrow B : \{N_a, A\}_{K_b}$
2'. $B \rightarrow C : \{N_a, N_b\}_{K_a}$
2. $C \rightarrow A : \{N_a, N_b\}_{K_a}$
3. $A \rightarrow C : \{N_b\}_{K_c}$
3'. $C \rightarrow B : \{N_b\}_{K_b}$

Łatwo zauważyć, że Charlie uczestniczy w realizacji jednocześnie dwóch protokołów z dwoma różnymi osobami: z Bobem i Alicją. Ponadto wszyscy troje znają wartości N_a i N_b.

2. Metody analizy oparte na konstruowaniu ataku

Niepoprawnie zaprojektowany protokół może zostać przełamany przez intruza, przez co nie będzie realizował zadań, jakie ma spełniać bezpieczny protokół. Z uwagi na trudności w zlokalizowaniu błędów protokołu, co wynika ze złożoności procesu analizy, dąży się do automatyzacji zadania analizy. Metody automatycznej weryfikacji protokołów powinny dostarczać w pierwszym rzędzie narzędzi do opisu protokołu oraz specyfikacji jego działania. Następnie narzędzia te, działając na bazie określonych procedur automatycznej weryfikacji, określają, czy protokół jest bezpieczny, czy też nie. Aby wykazać, że protokół spełnia stawiane mu wymogi, metoda automatycznej weryfikacji musi dokonać przeglądu wszystkich możliwych przejść protokołu i wykazać, że żadna z tych sekwencji nie stanowi zagrożenia dla bezpieczeństwa analizowanego protokołu. W przypadku złożonego protokołu, z bardzo dużą liczbą akceptowalnych sekwencji kroków, procedury automatycznej weryfikacji mogą być niezwykle czasochłonne. Wynika to z konieczności przeanalizowania dużego zbioru przejść, nim wykryta zostanie jedna lub wszystkie sekwencje kroków, ilustrujące możliwe scenariusze ataku na bezpieczeństwo protokołu.

Formalne metody analizy bezpieczeństwa protokołów kryptograficznych pozwalają zarówno na gruntowną analizę środowiska wymiany wiadomości, jak i różnych możliwych scenariuszy postępowania intruza, które mogą być zaakceptowane jako poprawne przez legalnych uczestników [12]. Przy ich użyciu, w oparciu o algebraiczne właściwości

algorytmów działania protokołów konstruuje się prawdopodobny zbiór ataków. Analiza protokołów z wykorzystaniem metod tej klasy bazuje na specyfikacji protokołu jako automatu ze skończoną liczbą stanów, rachunku predykatów lub w oparciu o algebrę procesu [1][6].

Większość późniejszych prac opartych na formalnej analizie protokołów kryptograficznych bazuje na modelu Doleva i Yao [4] lub jego odmianach. Przy ich konstruowaniu łączone są zwykle różne typy technik eksploracji przestrzeni stanów. Ich celem jest zweryfikowanie, czy istnieje jakaś ścieżka przechodząca przez daną przestrzeń, odpowiadająca udanemu atakowi intruza. Techniki stanowe wspierane są często przez indukcyjne metody dowodzenia twierdzeń (np. NRL Protocol Analyzer [13]), by wykazać, iż przeszukiwana przestrzeń stanów jest duża i gwarantuje bezpieczeństwo protokołu.

Sieci Petriego, w tym także kolorowane sieci Petriego (*CP-nets*) należą do grupy metod, wykorzystujących techniki eksploracji przestrzeni stanów. Możliwe są dwa scenariusze postępowania podczas analizy z wykorzystaniem CP-sieci. W pierwszym badamy, czy z określonego znakowania początkowego istnieje sekwencja przejść wzbudzonych, umożliwiających osiągnięcie stanu końcowego, drugi natomiast polega na sprawdzeniu, czy z przyjętego znakowania końcowego istnieje możliwość dojścia do stanu początkowego. Analiza "wstecz" (ang. *backward state analysis*) jest szczególnie interesująca z punktu widzenia weryfikacji protokołów kryptograficznych [1, 15].

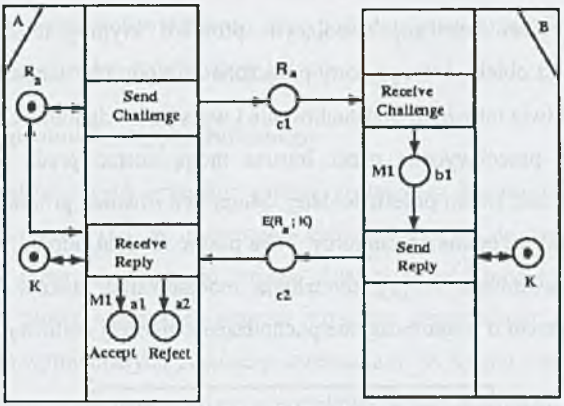
2.1. Modelowanie protokołów kryptograficznych z wykorzystaniem CP-sieci

Kolorowane sieci Petriego należą do grupy sieci Petriego wysokiego poziomu [7] i są rozwinięciem wcześniejszych wersji sieci typu miejsce/przejście. Kolorowana sieć Petriego jest graficznie zorientowanym językiem projektowania, specyfikacji, symulacji oraz weryfikacji systemów. Jej szczególnym obszarem zastosowania są problemy komunikacji, synchronizacji oraz współdzielenia zasobów, m.in. protokoły komunikacyjne, systemy zautomatyzowanej produkcji, systemy rozproszone oraz analiza przepływów.

Podstawową zaletą CP-sieci jest uproszczony formalizm opisu systemu, ułatwiający matematyczna analizę sieci. Wynika to głównie z faktu, iż w przeciwieństwie do znaczników sieci Petriego niskiego poziomu, w CP-sieci każdemu znacznikowi przypisuje się kolor, określający jego tożsamość. Przez pojęcie kolor rozumie się przy tym ściśle określoną wartość. Zbiór kolorów tworzy typ, który może odpowiadać złożonym strukturom danych (szczegóły patrz [8]).

[1] ang. *process algebra*, algebraiczna teoria stosowana do sformalizowania pojęcia obliczeń współbieżnych.

Przy modelowaniu protokołów kryptograficznych korzysta się z obiektów sieci Petriego, utożsamianych z hermetycznie zamkniętą siecią Petriego, z którą kontakt utrzymywany jest jedynie za pośrednictwem portów wejścia/wyjścia [1]. Hermetyzacja sieci umożliwia sterowanie poziomem szczegółowości opisu modelu sieci Petriego (patrz rys.1) i jednocześnie w przypadku modelowania protokołów kryptograficznych dobrze oddaje rzeczywistość: sekrety oraz działania każdej ze stron uczestniczących w protokole są zamykane wewnątrz obiektu, porty zaś odpowiadają liniom komunikacyjnym, poprzez które przesyłane są wiadomości pomiędzy stronami.



Rys.1. Funkcjonalny model sieci Petriego z dwoma obiektami (źródło: [1])
Fig.1. Functional CP-net model with two agents (the source: [1])

Modelowanie protokołów kryptograficznych za pomocą CP-sieci wydaje się być naturalne i łatwe do zrealizowania. Problem leży jednak w tym, że nas nie interesuje tylko normalna wymiana wiadomości pomiędzy uprawnionymi do tego stronami, ale przede wszystkim sytuacja, kiedy proces ten jest w dowolny sposób zaburzany przez intruza.

2.2. Model intruza

Zachowanie intruza należy zamodelować jako proces, który odpowiadać będzie dowolnemu typowi ataku (patrz [16]) możliwemu do przeprowadzenia przez intruza. Przy formułowaniu modelu intruza należy wziąć więc pod uwagę jego różne zachowania oraz możliwości, tj.:

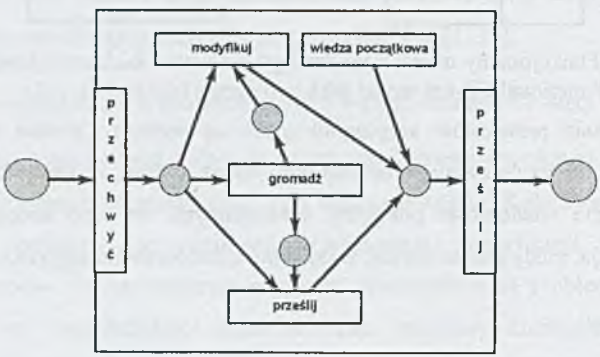
- podsłuch i/lub przejęcie dowolnej wiadomości przesyłanej pomiędzy stronami protokołu; podsłuchane wiadomości mogą być przez niego gromadzone,
- deszyfrowanie wiadomości za pomocą kluczy będących w posiadaniu intruza (w tym także kluczy publicznych należących do innych stron); odczytane wiadomości mogą być gromadzone przez intruza,
- wprowadzanie do systemu nowych wiadomości, bazując na wcześniej rozszyfrowanych,

- powtórzenie dowolnej wiadomości (atak powtórzeniowy) podsłuchanej (zmieniając np. tylko część przesyłaną otwartym tekstem), nawet jeśli intruz nie rozumie zawartości części zaszyfrowanej.

Zakładamy ponadto, że intruz nie posiada dostępu do kluczy kryptograficznych prawowitych uczestników protokołu, ani nie jest w stanie ich złamać w rozsądnym czasie, korzystając z dostępnych technik łamania kluczy. Założenie to nie będzie jednak dotyczyć tzw. kluczy sesji, które mogą być uzgodnione między stronami w wyniku realizacji protokołu, szczególnie w sytuacji, gdy związane z nimi algorytmy kryptograficzne są powszechnie znane jako łatwe do przełamania.

W modelu sieci Petriego, opisującym protokół kryptograficzny, intruz będzie reprezentowany przez obiekt, którego porty połączone są z portami uczestników protokołu w sposób, który umożliwia intruzowi podsłuchiwanie i wysyłanie wiadomości (patrz rys.2).

Wiadomości przechwycone przez intruza mogą zostać przez niego natychmiast zmodyfikowane lub bez zmian przesłane dalej. Mogą być również gromadzone i później po modyfikacji (lub bez) przesłane do odbiorcy. Taka postać modelu intruza, w przeciwieństwie do modelu przedstawionego w [1], umożliwia modelowanie ataków powtórzeniowych, realizowanych w oparciu o wiadomości nie pochodzące z bieżącej realizacji protokołu.



Rys.2. Ogólny model intruza
Fig.2. General model of an intruder

Ogólny model intruza podlega procesowi konkretyzacji, zależnemu od postaci analizowanego protokołu kryptograficznego. Konkretyzacja modelu intruza polega przede wszystkim na określeniu takich danych wyjściowych, tworzonych, powtarzanych lub przepuszczanych przez intruza, które traktowane są przez ich odbiorcę (prawowitego uczestnika protokołu kryptograficznego) jako poprawne, nawet jeśli w następnych krokach protokołu mogą być odrzucone. Zakładamy przy tym, że wiedza intruza wynika z założeń

początkowych protokołu oraz nagromadzonej przez niego informacji w wyniku podsłuchu poprzednich realizacji protokołu.

Przyjęcie założenia o wiedzy początkowej intruza uwzględnia istniejące w praktyce sytuacje (patrz [10]), w których intruz inicjuje lub kontynuuje wiele sesji protokołu z jednym lub kilkoma (w skrajnym przypadku wszystkimi) prawowitymi uczestnikami protokołu (może to robić z własnej inicjatywy lub zostać pobudzonym do tego typu działania przez innego uczestnika protokołu). Aby uwzględnić tego typu sytuacje, przyjmujemy, że z każdym uczestnikiem związany jest jeden intruz. Wszyscy intruzi kooperują ze sobą, prowadząc wspólną bazę przechwyconych wiadomości oraz posiadają taką samą wiedzę początkową (w praktyce może to być więc ten sam fizyczny intruz).

3. Sposób analizy protokołu kryptograficznego

Jedna z podstawowych procedur analizy protokołów kryptograficznych polega na realizacji dwóch kroków [1]. W pierwszym kroku modeluje się zachowanie protokołu kryptograficznego, ignorując obecność intruza. Analizując taki model próbuje się znaleźć potencjalnie słabe punkty protokołu poprzez sztuczne generowanie, modyfikowanie lub blokowanie danych wymienianych pomiędzy uczestnikami protokołu. Jeśli uda się przekazać stronie przeciwnej tego typu dane i strona ta potraktuje je jako poprawne, zapamiętuje się je, dopisując do zbioru zagrożeń protokołu i kontynuuje się eksperymenty do momentu, w którym eksperymentator uzna, że przebadał wszystkie możliwe przypadki. W drugim kroku dla każdej danej ze zbioru zagrożeń próbuje się budować model intruza w taki sposób, by zdefiniować stan końcowy naruszający bezpieczeństwo protokołu. Następnie wykazuje się, czy jest on możliwy do wyprowadzenia z początkowego stanu sieci (znakowania początkowego).

Metoda ta posiada naszym zdaniem dwa ewidentne ograniczenia. Po pierwsze, szukanie słabości protokołu poprzez analizowanie tylko zamodelowanego protokołu dla prawowitych uczestników uniemożliwia pełne przeszukanie wszystkich możliwych ataków. Dotyczy to w szczególności zagrożeń wynikających z odbicia wiadomości przez intruza lub przechowywania jej i późniejszego powtórzenia. Po drugie, budowanie modelu intruza, opierające się na zbiorze zagrożeń, sprowadza się de facto do opracowania scenariusza ataku na protokół. Zamodelowanie takiego scenariusza i wykazanie, że jest on osiągalny ze stanu początkowego, jest zazwyczaj oczywiste.

W związku z powyższym proponuje się łączyć dwa omawiane kroki w jeden, tzn. zbudować ogólny model zachowania intruza i opierając się na jego wiedzy początkowej

oraz wiedzy zdobywanej w trakcie biegu protokołu wskazać niebezpieczne stany końcowe, wyprowadzone z różnych prawdopodobnych stanów początkowych. Model intruza może być w kolejnych krokach poszerzany na podstawie wykonanych eksperymentów. Najbliższy realistycznym zachowaniom byłby oczywiście samouczący się model intruza.

Proponowana metoda wymaga większych nakładów obliczeniowych, ale z drugiej strony daje możliwość znacznie dokładniejszego przeszukania przestrzeni stanów niebezpiecznych oraz automatycznego zbudowania takiego stanu.

4. Przykład analizy

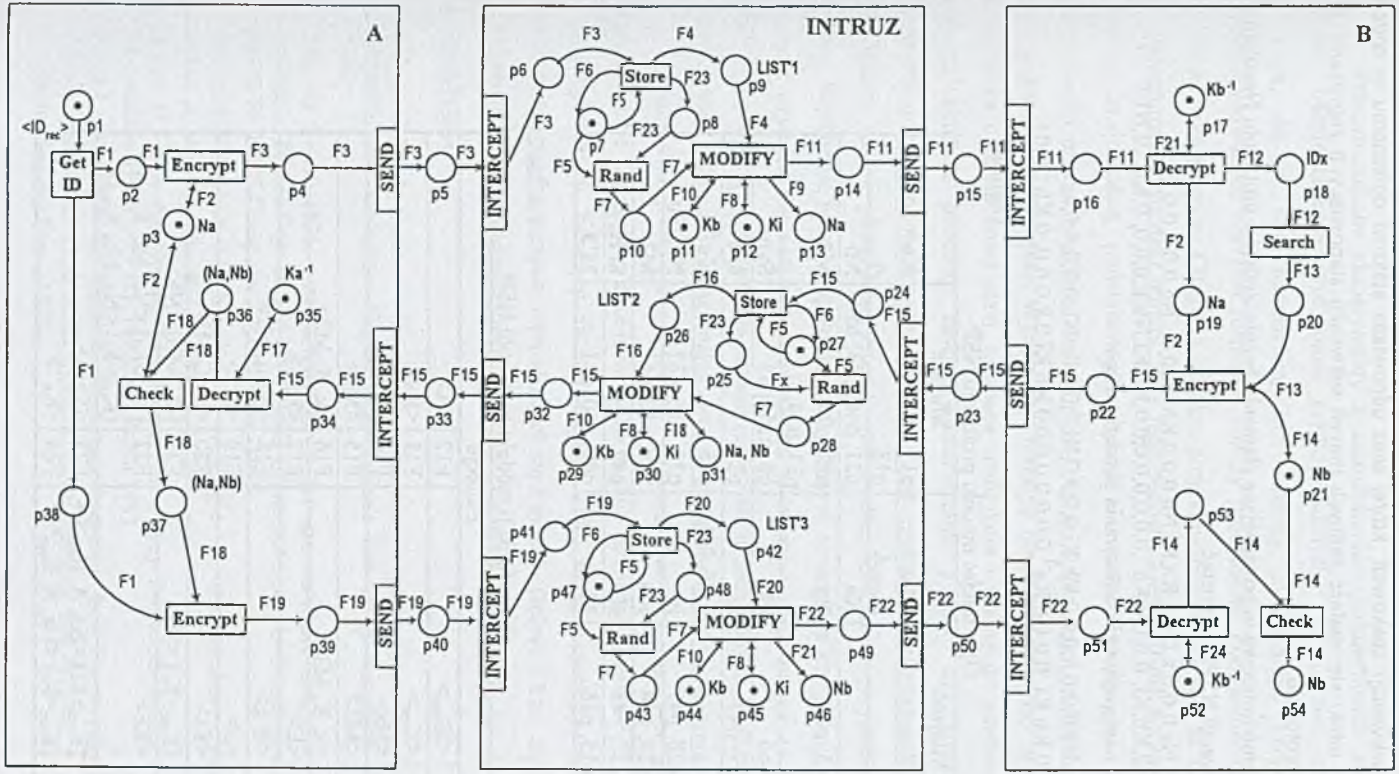
Ilustracją proponowanej metody analizy jest model CP sieci, przedstawiony na rys.3, prezentujący protokół Needhama-Schroedera w wersji NSP wraz z modelem intruza oraz zmodyfikowanym opisem samego protokołu (patrz model podmiotu A), po uwzględnieniu faktu, że intruz może być też prawnym uczestnikiem protokołu. W modelu nie uwzględniono możliwości kooperacji intruzów. W modelu z rys.3 zastosowano następujące oznaczenia:

ID_{send} - Identyfikator rozpoczynającego protokół. ID_{send} = A,
K_{send} - klucz publiczny rozpoczynającego protokół,
K_{rec} - klucz publiczny odbiorcy wskazanego przez rozpoczynającego protokół,
$K_{rec'}$ - klucz publiczny odbiorcy wskazanego przez intruza.

$$M1 = K_{rec}\{N_a, A\} \quad K_{rec} = \{K_i, K_b\}$$
$$M'1 = K_{rec}\{N_a, A\} \Leftrightarrow \quad K_{rec} = K_b$$
$$M'1 = K_{rec'}\{N_a, A\} \wedge K_{rec'} = K_b \Leftrightarrow \quad K_{rec} = K_i$$
$$M2 = K_{send}\{N_a, N_b\}$$
$$M'3 = K_{rec}\{N_b\} \Leftrightarrow \quad K_{rec} = K_b$$
$$M'3 = K_{rec'}\{N_b\} \wedge K_{rec'} = K_b \Leftrightarrow \quad K_{rec} = K_i$$

Przyjmijmy, że *list'1* ma strukturę postaci (k, K, $M1$), gdzie k – pozycja na liście, K – klucz publiczny, szyfrujący wiadomość M1. Przejścia *store*, *rand* i *modify* są modułami, których szczegóły pominięto. Moduł *store* umieszcza przechwyconą wiadomość w liście na pozycji k i zwiększa wartość k aż do maksymalnego rozmiaru listy. Z kolei moduł *rand* losuje numer p z listy i przekazuje go modułowi *modify*, który pobiera element umieszczony w liście na wylosowanej pozycji, a następnie próbuje go zmodyfikować.

Celem analizy jest pokazanie, że istnieje taka sekwencja znakowań, która prowadzi do zakończenia protokołu (przejście *check* w modelu podmiotu B zostanie uaktywnione i miejsce *Nb* zostanie oznakowane) i jednocześnie w modelu intruza oznakowane zostaną miejsca, które mogą być potencjalnymi zagrożeniami bezpieczeństwa protokołu: *Na*, *Nb*.



Rys.3. Model CP sieci protokołu Needhama-Schroedera z modelem intruza
Fig.3. The CP-net model for the Needham-Schroeder protocol with intruder

Liczne eksperymenty symulacyjne realizowane w oparciu o model z rys.3 pozwoliły na znalezienie sekwencji znakowań, której sens odpowiada atakowi opisanemu w rodz.1. Mimo prób nie udało się znaleźć żadnych innych sekwencji stanowiących zagrożenie dla analizowanego protokołu.

Znakowanie końcowe odpowiadające złamaniu bezpieczeństwa protokołu Needhama - Schroedera (w wersji NSP) ma postać:

M_n^T={ID_{rec},0,Na,0,0,0,k,0,0,0,Kb,Ki,**Na**,0,0,0,Kb^{-1},0,0,0, Nb,0,0,0,0,0,k,
0,Kb,Ki, 0,0,0,0,Ka^{-1},0,0,0,0,0,0,0,0,Kb,Ki,**Nb**,k,0,0,0, 0,Kb^{-1},0,**Nb**}

Uzyskano je dla następującego znakowania początkowego:

M_0^T={ID_{rec},0,Na,0,0,0,k,0,0,0,Kb,Ki,0,0,0,0,Kb^{-1},0,0,0, Nb,0,0,0,0,0,k,
0,Kb,Ki, 0,0,0,0,Ka^{-1},0,0,0,0,0,0,0,0,Kb,Ki,0,k,0,0,0, 0,Kb^{-1},0,0}

Tablica 1

Opis miejsc modelu protokołu NSP

Miejsce		Miejsce	
p1, p38	**ID_{rec}**	p17, p52	**Kb^{-1}**
p2	**K_{rec},A**	p18	**A**
p20	**Ka**	p21, p46, p53, p54	**Nb**
p3, p13, p19	**Na**	p22, p23, p24, p32, p33, p34	**M2**
p4,p5,p6	**M1**	p8, p25, p48	**Sync**
p7, p27, p47	**k**	p26	**k, K, M2**
p9	**k, K, M1**	p35	**Ka^{-1}**
p10, p28, p43	**p**	p31, p36, p37	**Na, Nb**
p11, p29, p44	**Kb**	p39, p40, p41	**M3**
p12, p30, p45	**Ki**	p42	**k, K, M3**
p14, p15, p16	**M'1**	p49, p50, p51	**M'3**

Tablica 2

Opis funkcji modelu protokołu NSP

Funkcje			
F0	<ID_{rec}>	F12	<A>
F1	<K_{rec}>	F13	<Ka>
F2	<Na>	F14	<Nb>
F3	<M1>	F15	<M2>
F4	<k, K, M1>	F16	<k, K, M2>
F5	<k>	F17	<Ka^{-1}>
F6	<$\oplus$ k>	F18	<N_{send}, Nb>
F7	<p>	F19	<M3>
F8	<Ki>	F20	<k, K, M3>
F9	[K_{rec}=Ki] <Na>	F21	[K_{rec}=Ki] <Nb>
F10	<Kb>	F22	[$K_{rec}\neq$Ki] <Na, Nb, K_{rec}> + [K_{rec}=K_i] <Na, Nb, K_{rec}'>
F11	[$K_{rec}\neq$Ki] <Na, A, K_{rec}> + [K_{rec}=K_i] <Na, A, K_{rec}'>	F23 F24	<Sync> <K_b^{-1}'>

5. Podsumowanie

Przeprowadzone eksperymenty z zastosowaniem kolorowanych sieci Petriego do analizy bezpieczeństwa protokołów kryptograficznych, jak również analiza literatury pokazują, że są one narzędziem, które należy uwzględniać przy projektowaniu zautomatyzowanych narzędzi formalnej analizy protokołów. Podstawowym mankamentem podejścia bazującego na CP-sieciach (także proponowanego przez nas) są znaczne trudności w skonstruowaniu modelu intruza. Przy jego budowie kierowaliśmy się raczej intuicją, jak również znajomością podstawowych ataków na protokoły istniejące. W przypadku konieczności przeprowadzenia analizy protokołu na etapie jego projektowania zadanie to byłoby znacznie trudniejsze. Wynika to z faktu, iż nowo projektowany protokół może być podatny na całkiem nowe ataki, nieznane analitykom i trudne do zdefiniowania. Jednym słowem, aktualne pozostaje zawsze fundamentalne pytanie: czy problem weryfikacji (konkretnego) protokołu kryptograficznego jest problemem rozstrzygalnym? Uzyskanie odpowiedzi na tak postawione pytanie jest trudne, również w przypadku innych metod analizy. Wydaje się jednak, że problem ten w sposób szczególny komplikuje się w przypadku analiz z użyciem CP-sieci. Rozstrzygalność problemu analizy protokołu możliwa byłaby jedynie w przypadku zbudowania samouczącego się modelu intruza. Jednak czy sieci CP są odpowiednią i jednocześnie optymalną metodą, którą można w tej dziedzinie wykorzystać?

Zdefiniowany powyżej problem wytycza kierunek naszych dalszych badań. Doświadczenie już zdobyte każe nam jednak sądzić, że znacznie lepsze cechy może posiadać system hybrydowy, bazujący na zastosowaniu kilku różnych podejść, np. CP-sieci oraz innych metod algebraicznych, wywodzących się z modelu Doleva i Yao, a przede wszystkim metod opartych na wnioskowaniu (m.in. logiki uwierzytelniania, HOL, itp., [2, 3, 5]).

LITERATURA

1. Basyouni A.: Analysis of wireless cryptographic protocols, Master thesis, Department of Electrical and Computer Engineering, Queen's University Kingston, Canada, August 1997.
2. Brackin S.: A HOL extension of GNY for automatically analyzing cryptograph8ic protocols, Proceedings of the 1996 IEEE Computer Security Foundations Workshop IX, pp.62-76, IEEE Computer Society Press, 1996.
3. Burrows M., Abadi M., Needham R.: A Logic of Authentication, Proc.Royal Society, Series A, Vol.426, No.1871, 1989, pp.233-271.
4. Dolev D., Yao A.: On the Security of Public Key Protocols. IEEE Transactions and Information Theory, 29 (2): 198-208, March 1983.
5. Gong L., Needham R., Yahalom R.: A reasoning about belief in cryptographic protocols, Proceedings of the 1990 IEEE Symposium on Security and Privacy, pp.234-248, IEEE Computer Society Press, 1990.

6. Gritzalis S., Spinellis D., Georgiadis P.: Security Protocols over open networks and distributed systems: Formal methods for their Analysis, Desin, and Verification, Computer Communications, 22(8): 695-707, May 1999.
7. Jensen K.: Coloured Petri Nets and the Invariant-Method. Theoretical Computer Science, 14:317-336, 1981.
8. Jensen K.: Coloured Petri nets, Advances in Petri Nets 1986,Part I, LNCS 254, Springer-Verlag, pp. 248-299, 1986.
9. Jensen K.: Coloured Petri Nets: a High Level Language for System Design and Analysis, Advances in Petri Nets, 1990, LNCS 483, Springer-Verlag, pp. 342-416, 1990.
10. Lowe G.: Some new attacks upon security protocols, Oxford University Computing Laboratory, United Kingdom, October 1996.
11. Lowe G.: Breaking and fixing the Neddham-Schroeder Public Key protocol using FDR, Oxford University Computing Laboratory, United Kingdom, October 1998.
12. Meadows C.: Open Issues in Formal Methods for Cryptographic Protocol Analysis, Proc. of DISCEX 2000, IEEE Computer Society Press, pp. 237-250, January, 2000.
13. Meadow C.: The NRL Protocol Analyzer: An Overview, Journal of Logic Programming, Vol. 26, No. 2, (1996) 113-131.
14. Needham R., Schroeder M.: Using encryption for authentication in large networks of computers, Communication of the ACM, 21(12): 993-999, December 1978.
15. Stal D.M., Tavares S. E., Meijer H.: Backward state analysis of cryptographic protocols using coloured Petri nets, Workshop on Selected Areas in Cryptography, SAC '94 Workshop Record, pp. 107-118, Kingston, Ontario, May 1994.
16. Syverson P.: A Taxonomy of replay attacks, Proc. 7th IEEE Comp. Security Foundations Workshop, pp.131-136, 1994.

Recenzent: Prof.dr hab.inż. M.Kubale

Abstract

In this paper we analyse the Needham-Schroeder Public-Key Protocol using Coloured Petri nets. There are introduced some important issues useful to develop models based on that technique for example: methods used to analyse cryptographic protocol, especially authentication one. Afterwards we review an attack construction method and its usefulness to take them as part of an automatic verification tool.

A class of methods based on state exploration techniques is Coloured Petri nets. The main idea of Coloured Petri nets is that CP-net assigns a set of colours to tokens and transitions to indicate their identity. There is functional dependency between the colour of the firing transition and the colour of the involved tokens.

The paper describes how CP-net can express the behaviour of large systems with few agents. Applying this method and Petri Net Objects (PNO), the model of agents taking part in the protocol is presented. There is also formulated the general model of an intruder who can interact with the protocol, observe, intercept messages and thus store the information and next use it later.

After that, an attack upon the protocol that allows the intruder to impersonate another agent is demonstrated. Finally, we give a brief conclusion and discussion about efficiency of CP-Nets as a method capable of discovering an attack upon authentication protocol.