STUDIA INFORMATICA 2016
Volume 37 Number 1 (123)

Sabina SZYMONIAK
Politechnika Częstochowska, Instytut Informatyki Teoretycznej i Stosowanej

WERYFIKACJA PROTOKOŁÓW ZABEZPIECZAJĄCYCH Z UWZGLĘDNIENIEM OPÓŹNIEŃ W SIECI

Streszczenie. W pracy zawarto opis problemu modelowania i weryfikacji czasowych protokołów zabezpieczających z uwzględnieniem opóźnień w sieci. Specyfikacje protokołów są zapisane w formacie ProToc, który umożliwia pełną specyfikację czasowego protokołu. Integralną częścią tej pracy jest zaprojektowane i zaimplementowane narzędzie, służące weryfikacji czasowych protokołów zabezpieczających. Narzędzie warunkuje określenie podatności danego protokołu zabezpieczającego na ataki, przy uwzględnieniu opóźnień w sieci.

Słowa kluczowe: weryfikacja, protokoły zabezpieczające, bezpieczeństwo

VERIFICATION OF SECURITY PROTOCOLS INCLUDING DELAYS IN THE NETWORK

Summary. This paper contains a description of the problem for modeling and verification of timed security protocols including delays in the network. Protocol specifications are written in the format ProToc, which allows full specification of the timed protocol. An integral part of this work is to designed and implemented tool for verifying time network security protocols. This tool allows specifying a particular protocol security vulnerability to attacks, taking into account the delays in the network.

Keywords: verification, security protocols, security

1. Wprowadzenie

W obecnych czasach z komunikacją przede wszystkim kojarzony jest Internet, czyli ogromna sieć komputerowa. Internet pozwala na wykonywanie wielu czynności, które ułatwiają ludzkie życie. Wśród nich warto wymienić rozmowy pomiędzy ludźmi mieszkającymi na odległych krańcach świata czy płacenie rachunków. Oczywiście każda z wykonywanych

czynności przy użyciu Internetu potrzebuje odpowiedniego poziomu zabezpieczeń, który wiąże się z doborem protokołu zabezpieczającego. Protokół ten stanowi sekwencję wymiany komunikatów, zapewniającą bezpieczeństwo komunikacji za pomocą technik kryptograficznych.

Niestety, przesyłane informacje są narażone na działanie Intruza, który może je wykraść i wykorzystać. W związku z tym bardzo ważne jest sprawdzanie, czy protokół użyty do zabezpieczania komunikacji rzeczywiście zapewnia odpowiedni poziom bezpieczeństwa. W sytuacjach, gdy kontynuujemy protokół po pewnej przerwie czasowej, możemy zakładać, że klucz kryptograficzny, którym została zaszyfrowana wiadomość, został już złamany, a zatem jego użycie nie gwarantuje żadnego bezpieczeństwa.

W celu uniknięcia sytuacji, w których przebieg protokołu z powodzeniem kontynuowany jest po przerwie czasowej, stosuje się tzw. znaczniki czasowe. Znacznik czasowy jest unikalnym identyfikatorem, którego wartość jest dostarczana przez lokalny zegar jednostki wysyłającej wiadomość. Taki znacznik określa, kiedy dana wiadomość została wygenerowana.

Niezależnie od tego, jakie zabezpieczenia są stosowane, w celu uniknięcia ataków ze strony Intruza, konieczne jest modelowanie i weryfikacja działania protokołów zabezpieczających. Wśród metod weryfikacji należy wyróżnić symulacje oraz formalne modelowanie i weryfikację. Symulacje polegają na testowaniu rzeczywistych systemów bądź symulowaniu ich działania za pomocą maszyn wirtualnych. Z kolei wśród metod formalnego modelowania i weryfikacji można wyróżnić metody indukcyjną, dedukcyjną oraz weryfikację modelową. Metoda indukcyjna polega na udowodnieniu spełniania określonych własności przez system, metoda dedukcyjna wiąże się ze skonstruowaniem specjalnego systemu dedukcyjnego (logiki), natomiast weryfikacja modelowa wiąże się z utworzeniem modelu w postaci odpowiedniego systemu tranzycyjnego [9].

Podczas modelowania i weryfikacji protokołów zabezpieczających bardzo ważnym etapem jest modelowanie czasu zarówno dla generowania czy wysyłania wiadomości, jak i czasu opóźnień w sieci, które są związane z transmisją wiadomości przez sieć pomiędzy nadawcą, a odbiorcą.

W latach 70. ubiegłego stulecia miał miejsce początek rozwoju weryfikacji protokołów zabezpieczających. Kluczową rolę odegrała tutaj praca Needhama i Schroedera [1], w której autorzy zaproponowali protokół uwierzytelniający, który przez wiele lat był stosowany w praktyce i uznawany za bezpieczny. Kilka lat później Dolev i Yao zaproponowali model intruza, dzięki któremu możliwa stała się dokładniejsza analiza protokołu, ze względu na uwzględnienie intruza w rozważaniach [2]. Niestety, szybko odkryto, że protokoły te można oszukać i nie zapewniają one pożądanego poziomu zabezpieczeń. Wraz z upływem czasu pojawiały się kolejne metody ich weryfikowania, najpierw metoda dedukcyjna [3], następnie indukcyjna [7] oraz weryfikacja modelowa [6]. Natomiast w tzw. międzyczasie Gavin Lowe

wykazał, że protokół zaproponowany przez Needhama i Schredera w bardzo prosty sposób można oszukać oraz zaproponował sposób jego naprawy [4, 5].

Wskazane prace z tej dziedziny odegrały bardzo ważną rolę w weryfikacji protokołów zabezpieczających. Prac dotyczących badania własności protokołów jest bardzo dużo. Jednakże badane są wyłącznie własności związane z bezpieczeństwem [6, 8]. Z kolei prac związanych z badaniem protokołów zależnych od czasu jest bardzo mało. W tych pracach, gdzie badane są protokoły czasowe, nie wskazywano wpływu czasu na bezpieczeństwo.

W pracy habilitacyjnej Kurkowskiego [9] został zaproponowany bardzo ciekawy model formalny wykonań protokołów, który posłużył do badania poprawności protokołów zabezpieczających również czasowych oraz do szukania ataków na nie. W pracy tej badane były własności związane z uwierzytelnianiem i poufnością, ale nie własności czasowe. Tylko w pracach Penczka i Jakubowskiej [10, 11] uwzględniano opóźnienia w sieci. W pracach tych wprowadzona została metoda, polegająca na obliczaniu czasu poprawnego wykonania sesji. W modelu wykonania protokołu znajdowały się zarówno limity czasu, jak i opóźnienia w sieci, a ustawione ograniczenia czasowe dla wykonywanych akcji umożliwiały wskazanie wpływu czasu na bezpieczeństwo protokołu. Niestety, w tym podejściu badania dotyczyły wyłącznie pojedynczej sesji, nie rozważano przeplotów wykonań protokołów, a prace te nie były kontynuowane. Wielokrotnie na konferencjach praktycy pytają o modele opóźnień w sieciach.

2. Protokół KaoChow v. 1

Jednym z protokołów zabezpieczających jest protokół KaoChow w wersji pierwszej zaproponowany przez I Long Kao i Randy'ego Chowa w 1995 roku. Protokół ten składa się z czterech kroków, a jego zadaniem jest dystrybucja nowego klucza symetrycznego współdzielonego pomiędzy użytkownikami A i B oraz wzajemne uwierzytelnienie tych użytkowników. Składnia czasowej wersji tego protokołu, utworzonej na podstawie [12], jest następująca:

$$\alpha_1\ A \to S : A, B, Ta;$$
$$\alpha_2\ S \to B :< A, B, Ta, Kab >_{Kas}, < Ta >_{Kbs};$$
$$\alpha_3\ B \to A :< A, B, Ta, Kab >_{Kas}, < TA >_{Kab} Tb;$$
$$\alpha_4\ A \to B :< Tb >_{Kab};$$

gdzie: **A**, **B** – są oznaczeniami uczciwych uczestników komunikacji, **S** – jest oznaczeniem zaufanego serwera, **Ta**, **Tb** – oznaczają znaczniki czasowe użytkowników A oraz B, **Kas**, **Kbs** – oznaczają klucze symetryczne współdzielone pomiędzy uczciwymi użytkownikami

i serwerem, **Kab** – oznacza nowy symetryczny klucz współdzielony pomiędzy uczciwymi użytkownikami.

Użytkownik A pragnie skomunikować się z użytkownikiem B. W związku z tym, w pierwszym kroku, przesyła do zaufanego serwera wiadomość, w której umieszcza identyfikatory stron komunikujących się (swój i B) oraz swój wygenerowany znacznik czasowy. Serwer komponuje dwa podobne szyfrogramy i przesyła je jako jedną wiadomość do użytkownika B. Pierwszy cząstkowy szyfrogram został zaadresowany do A, zatem do szyfrowania wykorzystano klucz Kas. W wiadomości tej znalazły się identyfikatory obu użytkowników, znacznik czasowy A oraz nowowygenerowany przez S klucz symetryczny. Podobną budowę ma drugi cząstkowy szyfrogram. Różnicą jest wyłącznie klucz szyfrujący. W tym przypadku wiadomość została zaszyfrowana kluczem Kbs. W trzecim kroku B przesyła do A tę część poprzedniej wiadomości, która została zaadresowana właśnie do A, dołącza do niej znacznik czasowy A zaszyfrowany kluczem Kab oraz swój wygenerowany znacznik czasowy. W celu potwierdzenia tożsamości A szyfruje znacznik czasowy B kluczem Kab i przesyła go do B.

3. Model formalny i struktura obliczeniowa

Weryfikacja protokołów zabezpieczających z uwzględnieniem opóźnień w sieci wymaga przygotowania odpowiedniego języka formalnego, dzięki któremu zostanie zdefiniowany protokół traktowany jako algorytm. Język formalny, zastosowany to tego rozwiązania, będzie stanowił naturalne rozszerzenie języka zaproponowanego przez Kurkowskiego w [9]. Następnie za pomocą struktury obliczeniowej będzie możliwe odwzorowanie tego protokołu w różne jego wykonania. W kolejnym kroku otrzymane różne wykonania tego samego protokołu (różniące się w czasie) zostaną zakodowane w przebiegi w sieci czasowych automatów.

Bardzo ważnymi składowymi języka formalnego są warunki czasowe, które umożliwiają uwzględnienie opóźnień w sieci podczas wykonywania protokołu oraz krok protokołu. Warunki czasowe definiuje następująca gramatyka:

$$tc ::= true \mid \tau_i + \tau_d - \tau_j \le L_F \mid tc \wedge tc, \text{ dla } \tau_i \in T_R, \tau_j \in T_T, \tau_d \in T_T \text{ oraz } L_F \in T_T$$

gdzie: τ_i – oznacza czas wysyłania wiadomości, τ_d – oznacza czas opóźnienia w sieci, τ_j – oznacza czas wygenerowania znacznika czasowego, L_F – oznacza okres ważności znacznika czasowego, T_R – oznacza zbiór zmiennych rzeczywistych określających chwile wysłania wiadomości, T_T – oznacza zbiór symboli reprezentujących znaczniki czasowe, T_D – oznacza zbiór symboli reprezentujących opóźnienia w sieci, T_L – oznacza zbiór symboli reprezentujących okresy ważności znaczników czasowych.

Gramatyka ta definiuje trzy rodzaje warunków czasowych. Pierwszy z nich (*true*) wskazuje na to, że dany krok protokołu nie jest zależny od czasu, zatem wszystkie warunki czasowe są spełnione. Drugi rodzaj warunku definiuje zależność pomiędzy różnicą sumy $\tau_i + \tau_d$, czyli czasu wysyłania wiadomości i czasu opóźnienia w sieci, a τ_j, czyli czasu wygenerowania znacznika czasowego, od L_F, czyli okresu ważności znacznika czasowego. Trzeci rodzaj warunku czasowego wskazuje na to, że wszystkie warunki czasowe mogą być łączone w bardziej złożone warunki za pomocą koniunkcji.

Z kolei krok protokołu zabezpieczającego, który uwzględnia opóźnienia w sieci, będzie stanowić para uporządkowana (α^1, α^2), gdzie pierwszy element tej pary informuje o zewnętrznych parametrach komunikacji, czyli kto i do kogo wysyła wiadomość. Zatem α^1 jest trójką $(P,Q,L) \in T_P \times T_P \times T$. P jest oczywiście nadawcą wiadomości, Q jej odbiorcą, a L – przesyłaną wiadomością.

Drugi element pary zawiera wewnętrzne parametry komunikacji, takie jak parametry czasowe czy zbiory obiektów kryptograficznych, z których tworzona jest wiadomość przesyłana w danym kroku. Zatem α^2 jest następującą piątką:

$$(\tau, \tau_d, X, G, tc) \in T_R \times 2^T_{fin} \times 2^{T_K \cup T_N \cup T_T}_{fin} \times C,$$

gdzie: τ – oznacza czas wysyłania wiadomości, τ_d – oznacza czas opóźnienia w sieci, X – oznacza zbiór listów potrzebnych do skomponowania wiadomości L, G – oznacza zbiór listów generowanych przez wysyłającego, w celu przygotowania listu L, tc – oznacza zbiór warunków czasowych.

Składnia kroków czasowej wersji protokołu KaoChow jest następująca:

$$\alpha_1 = (\alpha_1^1, \alpha_1^2):$$
$$\alpha_1^1 = (A; S; I_A \cdot I_B \cdot \tau_A),$$
$$\alpha_1^2 = (\tau_1; D_1 \{I_A, \tau_A, I_B\}; \{\tau_A\}; \tau_1 + D_1 - \tau_A \le L_F),$$
$$\alpha_2 = (\alpha_2^1, \alpha_2^2):$$
$$\alpha_2^1 = (S; A; \langle I_A \cdot I_B \cdot \tau_A \rangle_{K_{AS}} \cdot \langle I_A \cdot I_B \cdot \tau_A \rangle_{K_{BS}}),$$
$$\alpha_2^2 = (\tau_2; D_2; \{I_A, \tau_A, I_B, K_{AB}, K_{AS}, K_{BS}\}; \{\emptyset\}; \tau_2 + D_2 - \tau_A \le L_F),$$
$$\alpha_3 = (\alpha_3^1, \alpha_3^2):$$
$$\alpha_3^1 = (S; B; \langle I_A \cdot I_B \cdot \tau_A \rangle_{K_{AS}} \cdot \langle \tau_A \rangle_{K_{AB}} \cdot \tau_B),$$
$$\alpha_3^2 = (\tau_3; D_3; \{I_A, I_B, \tau_A, \tau_B, K_{AB}, K_{AS}\}; \{\tau_B\}; \tau_3 + D_3 - \tau_A \le L_F \wedge \tau_3 + D_3 - \tau_B \le L_F),$$
$$\alpha_4 = (\alpha_4^1, \alpha_4^2):$$
$$\alpha_4^1 = (A; B; \langle \tau_B \rangle_{K_{AB}}),$$

$$\alpha_4^2 = (\tau_4; D_4; \{\tau_B, K_{AB}\}; \{\emptyset\}; \tau_4 + D_4 - \tau_A \le L_F \wedge \tau_4 + D_4 - \tau_B \le L_F).$$

Warunek czasowy dla pierwszego kroku protokołu KaoChow mówi, że zaufany serwer może odebrać i przetwarzać wiadomość wtedy, gdy różnica czasowa między czasem odebrania wiadomości i opóźnienia w sieci oraz czasem wygenerowania znacznika czasowego jest mniejsza od okresu ważności. Z kolei warunek czasowy z kroku drugiego wskazuje, że użytkownik B może odebrać i przetwarzać wiadomość wtedy, gdy są spełnione następujące zależności: różnica czasowa między czasem odebrania wiadomości i opóźnienia w sieci oraz czasem wygenerowania znacznika czasowego A jest mniejsza od okresu ważności. W przypadku trzeciego i czwartego kroku pojawiają się złożone warunki czasowe, które uwzględniają oba wygenerowane znaczniki czasowe.

W strukturze obliczeniowej zdefiniowane warunki czasowe oraz krok protokołu zostaną oczywiście odpowiednio odwzorowane za pomocą funkcji odwzorowujących. Zatem *f*-interpretację kroku α stanowić będą krotki:

$$((h(P), h(Q), h(L)), (h(\tau_\alpha), h(\delta_\alpha), h(X), h(G), h(tc))), \text{ jeżeli } h(P) \in P,$$
$$((h(P), h(Q), h(L)), (h(\tau_\alpha), h(\delta_\alpha), \{X \mid X \mid -f(L)\}, \emptyset, h(tc))), \text{ jeżeli } h(P) \in P_L.$$

Pierwsza krotka odnosi się do użytkownika uczciwego, natomiast druga do Intruza. Różnica pomiędzy tymi dwoma krotkami polega na tym, że Intruz może utworzyć wiadomość $h(L)$ z każdego zbioru, który generuje tę wiadomość. Ponadto przyjmuje się, że Intruz ma zbiór wcześniej wygenerowanych obiektów kryptograficznych, z których korzysta w różnych sesjach. Wykonanie protokołu będzie oczywiście sekwencją interpretacji kroków protokołu, przy założeniu że wszystkie warunki czasowe, nałożone na poszczególne kroki, zostały spełnione.

Możliwości tworzenia wiadomości przez użytkowników zarówno uczciwych, jak i Intruza zależą od aktualnego stanu ich wiedzy. Wiedza ta zmienia się w trakcie wykonywania protokołu. W początkowym zbiorze wiedzy użytkownika uczciwego znajdują się jego klucze prywatne oraz informacje publicznie dostępne, czyli identyfikatory innych użytkowników oraz ich klucze publiczne. Jeżeli dany użytkownik wysyła wiadomość w danym kroku, to zbiór jego wiedzy powiększa się o wszystkie elementy, które musiał wygenerować, aby utworzyć wiadomość. Jeżeli zaś użytkownik jest adresatem wiadomości w danym kroku, jego wiedza powiększa się o wszystkie elementy, które może wydobyć z otrzymanego szyfrogramu.

Z kolei w zbiorze początkowym wiedzy Intruza znajdują się informacje publicznie dostępne (takie same jak w przypadku uczciwego użytkownika) oraz klucz prywatny i zbiory wygenerowanych wcześniej znaczników czasowych i nonces. Dla rozważanego modelu Intruza Dolev-Yao wiedza Intruza zmienia się wtedy, gdy Intruz jest stroną odbierającą

w danym kroku. Wiedza Intruza powiększa się o otrzymany szyfrogram oraz o wszystkie elementy, które może pobrać z tego szyfrogramu przy bieżącym stanie jego wiedzy.

W wiedzy początkowej użytkownika A znajdują się identyfikatory użytkowników oraz symetryczny klucz współdzielony z zaufanym serwerem. Podczas wykonywania pierwszego kroku zbiór jego wiedzy powiększa się o element, który użytkownik musiał wygenerować, aby móc utworzyć wiadomość, czyli znacznik czasowy. Z kolei w drugim kroku protokołu, gdzie użytkownik A nie bierze udziału, zbiór jego wiedzy pozostaje niezmieniony. W trzecim kroku wiedza użytkownika A powiększa się o elementy pozyskane z szyfrogramu, czyli klucz symetryczny współdzielony z B oraz jego znacznik czasowy. Wykonanie czwartego kroku również nie zmienia stanu wiedzy A.

W przypadku użytkownika B, w zbiorze jego wiedzy początkowej znajdują się identyfikatory użytkowników oraz symetryczny klucz współdzielony z zaufanym serwerem. Użytkownik B nie bierze udziału podczas wykonywania pierwszego kroku, zatem jego wiedza nie zmienia się. W drugim kroku protokołu użytkownik B otrzymuje wiadomość od serwera, zatem jego wiedza powiększa się o wszystkie elementy, jakie może pobrać z otrzymanego szyfrogramu, czyli znacznik czasowy A oraz klucz symetryczny wygenerowany przez S. W trzecim kroku B generuje swój znacznik czasowy, a w czwartym kroku wiedza tego użytkownika nie zmienia się.

Ponadto w strukturze obliczeniowej uwzględniony został bardzo ważny parametr czasowy, jakim jest czas oczekiwania na odpowiedź (ang. *timeout*). Parametr ten stanowi oczywiście czas, jaki użytkownik oczekuje na zwrotną odpowiedź po wysłaniu swojej wiadomości. Czas ten został oznaczony jako t_{out} i obliczany zgodnie z następującym wzorem:

$$t_{out_i} = t_{gen_j} + d_j + t_{gen_{j+1}} + d_{j+1} + ... + t_{gen_k} + d_k, \tag{1}$$

gdzie: $i = 1 \ldots k\text{-}1$ – jest licznikiem timeoutów w danym wykonaniu, $j = 1 \ldots k$ – jest licznikiem kroków w danym wykonaniu, k – jest liczbą kroków w danym protokole, t_{outi} – jest i-tym timeoutem w danym wykonaniu, d_i – jest czasem opóźnienia w sieci.

Warto pamiętać, że nie zawsze w protokole występuje wyłącznie dwóch użytkowników, którzy wymieniają między sobą wiadomości. Istnieją sytuacje, w których w protokole występuje większa liczba użytkowników, a wiadomości nie są wymieniane między nimi sekwencyjnie. Występują również takie protokoły, które nie mają wiadomości zwrotnych (np. protokół Wide-Mouth Frog).

4. Wyniki eksperymentalne

Na potrzeby przeprowadzenia badań zostało zaimplementowane narzędzie (w języku C++), które ma służyć modelowaniu i generowaniu wykonań protokołów zabezpieczających z uwzględnieniem opóźnień w sieci. Specyfikacja protokołu została przygotowana w języku ProToc. Język ten umożliwia pełną specyfikację protokołu, uwzględniając wszystkie akcje (zewnętrzne i wewnętrzne), jakie są wykonywane w trakcie działania protokołu. ProToc został omówiony w [14].

Przeprowadzone badania eksperymentalne miały na celu wykazanie wpływu wartości parametrów czasowych na bezpieczeństwo protokołu. Na potrzeby badań zostały przyjęte założenia, uwzględnione w implementacji, mające na celu ograniczenie wybranych wartości czasowych oraz ograniczenie działalności Intruza. Wartość generowanego opóźnienia w sieci oraz czasu generowania i wysyłania wiadomości przez uczciwego użytkownika zostały ograniczone do liczby z zakresu od 0 do 1 jednostki czasowej. Wartość okresu ważności znacznika czasowego stanowi liczba z zakresu od 2 jednostek czasowych do liczby jednostek czasowych zadeklarowanej przez użytkownika (oznaczonej jako l_{max}). Z kolei czas generowania wiadomości przez Intruza został ograniczony od przedziału od 1 jednostki czasowej do liczby stanowiącej l_{max}. Wyniki przeprowadzonych badań zostały przedstawione poniżej.

Jako pierwszy został przebadany protokół KaoChow v1, opisany w sekcji 2. Wartości parametrów czasowych, uzyskanych dla tego protokołu, zostały zebrane i podsumowane w tabeli 1. Wszystkie wartości oczywiście zostały przedstawione w tzw. jednostkach czasowych [jc]. Uzyskane wartości opóźnień w sieci oscylowały pomiędzy wartościami 0,00271144 [jc] a 0,99033959 [jc]. Najmniejszy osiągnięty czas sesji wynosi 0,90722908 [jc], natomiast największy to 6,01417096 [jc]. Czasy generowania i wysyłania wiadomości zostały wygenerowane zgodnie z przyjętymi założeniami. Dla uczciwego użytkownika średni czas wyniósł 0,52559903 [jc]. W przypadku Intruza średni czas generowania i wysyłania wiadomości wyniósł 4,254905878 [jc].

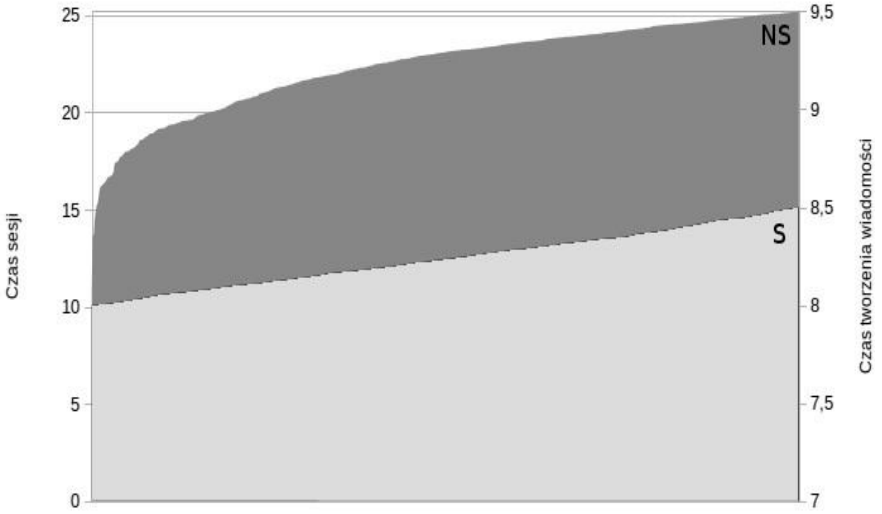
Tabela 1

Wyniki eksperymentalne dla protokołu KaoChow v.1

	Opóźnienie w sieci	Czas sesji	Czas generowania i wysyłania wiadomości	
			Przez Intruza	Przez uczciwego użytkownika
Wartość minimalna	0,00271144 [jc]	0,90722908 [jc]	1,060900811 [jc]	0,004705063 [jc]
Wartość średnia	0,51726392 [jc]	3,5677 [jc]	4,254905878 [jc]	0,52559903 [jc]
Wartość maksymalna	0,99033959 [jc]	6,01417096 [jc]	9,952968297 [jc]	0,983651944 [jc]

Badania zostały przeprowadzone dla okresów ważności znaczników czasowych z przedziału od 2 sekund do 12 sekund. Średni okres ważności wyniósł 5,51 [jc].

Na podstawie otrzymanych wyników można było odczytać zależności, jakie łączyły parametry czasowe wykonań. Jedną z odczytanych zależności był wpływ czasu generowania i wysyłania wiadomości przez Intruza na czas sesji. Zależność ta została pokazana na rys. 1.



Rys. 1. Wpływ czasu generowania i wysyłania wiadomości przez Intruza na czas sesji
Fig. 1. Influence of generating and sending time by Intruder for session time

Na wykresie zostały ujęte wartości czasu generowania i wysyłania wiadomości przez Intruza oraz wartości czasu sesji, czyli czasu jednego wykonania protokołu. Wzrost wartości czasu tworzenia wiadomości miał ogromy wpływ na czas sesji, który również zwiększał się. Jednakże w sytuacji, gdy czas ten przyjmował wartości powyżej 8,0000233468 [jc] (linia łącząca ciemniejszą i jaśniejszą część wykresu), warunki czasowe nałożone na poszczególne kroki wykonania protokołu nie zostały spełnione (NS na wykresie). W związku z tym wykonanie protokołu zakończyło się błędem, a Intruz nie mógł dokonać ataku.

Gdy czas tworzenia wiadomości przyjmował wartości poniżej wartości 8,0000233468 [jc], wykonanie protokołu zakończyło się poprawnie (S na wykresie). Intruz mógł zatem dokonać atak na ten protokół.

W przypadku wykonań, w których uczestniczyli wyłącznie uczciwi użytkownicy, wszystkie warunki czasowe zostały spełnione. Wykonania te przebiegły poprawnie. Wartości opóźnień w sieci oczywiście miały zasadniczy wpływ na czasy sesji.

5. Podsumowanie

Modelowanie i weryfikacja protokołów zabezpieczających z uwzględnieniem opóźnień w sieci jest bardzo ważnym etapem zabezpieczania komunikacji internetowej. Wykorzysty-

wany protokół powinien zapewniać odpowiedni poziom bezpieczeństwa komunikacji, aby uniknąć ataków ze strony Intruza.

W artykule został zaprezentowany formalny model wykonań protokołów zabezpieczających z uwzględnieniem opóźnień w sieci. Model ten odpowiednio odwzorowuje rzeczywiste wykonania protokołów w sieci. Dzięki temu możliwe jest weryfikowanie przebiegu modelowanego protokołu. Dodatkowo zostało zaimplementowane narzędzie służące do automatycznej weryfikacji protokołów zabezpieczających.

Wartość opóźnień w sieci ma ogromny wpływ na wykonanie protokołu. Podczas tworzenie infrastruktury sieciowej warto weryfikować jej podatność na ataki Intruza, mając na uwadze opóźnienia.

W kolejnych badaniach zostanie dokonany przegląd własności czasowych kolejnych protokołów zależnych od czasu protokołów z uwzględnieniem opóźnień w sieci.

BIBLIOGRAFIA

1. Needham R., Schroeder M.: Using encryption for authentication in large networks of computers. Commun. ACM, Vol. 21(12), 1978, s. 993÷999.
2. Dolev D., Yao A.: On the security of public key protocols. IEEE Transactions on Information Theory, Vol. 29(2), 1983, s. 198÷208.
3. Burrows M., Abadi M., Needham R.: A logic of authentication. Proceedings of the Royal Society of London A, Vol. 426, DEC Systems Research Center, 1989, s. 233÷271.
4. Lowe G.: An Attack on The Needham-Schroeder Public-Key Authentication Protocol. Information Processing Letters, Vol. 56, No 3, 1995, s. 131÷133.
5. Lowe G.: Breaking and Fixing the Needham-Schroeder Public-Key Protocol Using FDR. Proceedings of TACAS, Springer-Verlag, 1996, s. 147÷166.
6. Kurkowski M., Penczek W.: Applying timed automata to model checking of security protocols. [in:] Wang J. (ed.): Handbook of Finite State Based Models and Applications, Chapman and Hall/CRC Press, 2013, s. 223÷254.
7. Bella G., Paulson L.C.: Using Isabelle to prove properties of the Kerberos authentication system. [in:] Orman H., Meadows C. (eds.): Proceedings of the DIMACS Workshop on Design and Formal Verification of Security Protocols (CD-ROM) (DIMACS'97), 1997.
8. Benerecetti M., Cuomo N., Peron A.: TPMC: A Model Checker For Time-Sensitive Security Protocols. Journal of Computers, North America, 2009.

9. Kurkowski M.: Formalne metody weryfikacji własności protokołów zabezpieczających w sieciach komputerowych. Wyd. Exit, Warszawa 2013.
10. Jakubowska G., Penczek W.: Modeling and Checking Timed Authentication Security Protocols. Proceedings of the International Workshop on Concurrency, Specification and Programming (CS&P'06), Informatik-Berichte, Vol. 206(2), Humboldt University 2006, s. 280÷291.
11. Jakubowska G., Penczek W.: Is Your Security Protocol on Time? Proceedings of the IPM International Symposium on Fundamentals of Software Engineering (FSEN'07), LNCS, Vol. 4767, Springer-Verlag, 2007, s. 65÷80.
12. Repozytorium protokołów zabezpieczających SPORE: http://www.lsv.ens-cachan.fr/Software/spore/table.html, dostęp listopad 2015.
13. Penczek W., Półrola A.: Advances in Verification of Time Petri Nets and Timed Automata: A Temporal Logic Approach. Springer-Verlag, 2006.
14. Kurkowski M., Grosser A., Piątkowski J., Szymoniak S.: ProToc – an universal language for security protocols specification. Advances in Intelligent Systems and Computing, Vol. 342, Springer-Verlag, 2015, s. 237÷248.

Abstract

This article presents the formal model of security protocols executions including delays in the network and implementation of tool for automatic verification of security protocols. Protocol specifications are written in the format ProToc, which allows allows full specification of the timed protocol. This tool allows specifying a particular protocol security vulnerability to attacks, taking into account the delays in the network.

The first section provides an overview of the literature and the current state of knowledge in the field of modeling and verification security protocols. The second section contains an example of security protocol which is KaoChow protocol. This protocol consists of four steps, and his task is to distribute a new symmetric key shared between users A and B, and mutual authentication of users. In the third section is described formal model and computational structure for modeling real security protocol executions. The fourth section describes a tool for automatic verification of security protocols.

In fifth section are shown experimental results. Some summary results are shown in Tab. 1. Based on these results could be read that spanned depending on the timing of executions. One of the influences was depending of time of generating and sending the message by Intruder during the session. This relationship is shown in Fig. 1.

Modeling and verification of security protocols including delays in the network is a very important step in securing Internet communications. The protocol used should provide adequate security of communications in order to avoid attacks from the Intruder.

Adres

Sabina SZYMONIAK: Politechnika Częstochowska, Instytut Informatyki Teoretycznej i Stosowanej, ul. Dąbrowskiego 69, 42-200 Częstochowa, Polska, sabina.szymoniak@icis.pcz.pl.