STUDIA INFORMATICA 2001
Volume 22 Number 3 (45)

Marcin KUCZORA
Politechnika Śląska, Instytut Informatyki

SSL JAKO ELEMENT BEZPIECZEŃSTWA E-SERWERÓW

Streszczenie. Artykuł prezentuje rezultaty testu ponad 8000 losowo wybranych serwerów SSL (*Secure Socket Layer*), serwerów używanych do prowadzenia działalności typu e-commerce. Przegląd poziomu zabezpieczeń przez nie oferowanego jest poprzedzony krótką charakterystyką tego standardu. Artykuł próbuje wskazać powody istniejącego stanu rzeczy, ostrzegając o potencjalnych zagrożeniach związanych z niewłaściwą implementacją SSL-a na serwerach internetowych oraz niebezpieczeństwie związanym z pracą z domyślnymi parametrami przeglądarek WWW.

SSL AS A PART OF E-SERVER SECURITY LEVEL

Summary. Article presents results of the survey carried out in order to check out over 8000 random chosen e-commerce SSL servers. After the short description of SSL standard and presentation of test results, the conclusion section includes the reasons and consequences of present state, as well as potential danger connected with improper implementation of SSL on Web servers and the danger of working with default parameters of some Web browsers.

1. Wprowadzenie do zagadnień związanych z protokołem SSL

1.1. Protokół SSL

Protokół TCP/IP (*Transmision Control Protocol/Internet Protocol*) zarządza transportem i sposobem przepływu danych przez Internet. Inne protokoły, takie jak HTTP (*HyperText Transport Protocol*), LDAP (*Lightweight Directory Access Protocol*) lub IMAP (*Internet Messaging Access Protocol*) działają "na wierzchu" TCP/IP w tym sensie, że wykorzystują TCP/IP do wykonywania typowych zadań aplikacyjnych takich, jak wyświetlanie stron WWW lub prowadzenie serwerów z kontami pocztowymi.

Protokół SSL działa jako warstwa ponad protokołem TCP/IP, ale poniżej protokołów takich, jak HTTP lub IMAP, używając TCP/IP na rzecz protokołów wyższego poziomu. W trakcie rozpoczynania tzw. *bezpiecznego połączenia* umożliwia serwerowi z uaktywnionym mechanizmem SSL . samoautentyfikację względem klienta SSL, autentyfikację odwrotną klienta oraz pozwala tym samym obydwu stronom na nawiązanie bezpiecznego połączenia [10].

Już powyższy krótki opis funkcji protokołu SSL wskazuje, że zaspokaja on fundamentalne potrzeby związane z bezpieczną komunikacją poprzez Internet i inne sieci oparte na TCP/IP, tzn.:

- **autentyfikację serwera SSL** (potwierdzenie tożsamość serwera); klienci mogą wykorzystywać standardowe techniki kryptograficzne zakładające istnienie klucza publiczego (*public-key*) w celu kontroli, czy certyfikat serwera jest ważny i został wydany przez zaufanego wystawcę certyfikatu (z ang. *CA – certificate authority*). Potwierdzenie takie ma kluczowe znaczenie dla klienta np. w przypadku wysyłania przez sieć numeru swojej karty kredytowej,
- **autentyfikację klienta SSL**; pozwala serwerowi na potwierdzenie tożsamości klienta. Technika jest analogiczna do stosowanej przy autentyfikacji serwera SSL. Autentyfikcja taka może mieć znaczenie dla serwera np. gdy bank wysyła poufne informacje finansowe do klienta i chce uzyskać potwierdzenie tożsamości odbiorcy danych,
- **bezpieczne połączenie SSL**; kodowanie obejmuje wszystkie informacje wysyłane podczas dialogu, zapewniając poziom bezpieczeństwa, którego jakość jest głównym tematem niniejszego artykułu. Oprócz poufności mającej znaczenie dla obydwu stron w trakcie każdej prywatnej transakcji, jest chroniona integralność danych przy użyciu mechanizmu wykrywającego modyfikację informacji podczas transportu [6].

1.2. Szyfry używane z SSL

Protokół SSL zapewnia współpracę z wieloma różnymi algorytmami kryptograficznymi lub **szyframi**, by używać ich w operacjach takich, jak wzajemna autentyfikacja serwera i klieta, transmisja certyfikatów, ustalanie kluczy sesyjnych. W związku z tym, że klienci i serwery mogą obsługiwać różne **zestawy szyfrów**, w zależności od czynników takich,jak np. wersja SSL, istnieje specjalny schemat **wymiany potwierdzeń** (z ang. *handshake*) określający, w jaki sposób serwer i klient negocjują zestawy szyfrów, których użyją do wzajemnej autentyfikacji, transmisji certyfikatów i określenia kluczy sesyjnych, które z kolei zostaną wykorzystane podczas właściwej sesji.

Wykorzystywane zestawy szyfrów zawierają następujące algorytmy: DES (*Data Encryption Standard*), DSA (*Digital Signature Algorithm*), KEA (*Key Exchange Algorithm*),

MD5 (*Message Digest*), RC2, RC4, RSA, RSA z wymianą kluczy, SHA (*Secure Hash Algorithm*), SKIPJACK, Triple-DES (szyfr,w którym DES jest stosowany trzykrotnie) [8].

1.3. Wybór zestawu szyfrów i wystawcy certyfikatów (CA)

Decyzja odnośnie do wyboru zestawu wykorzystywanych szyfrów jest zwykle pozostawiona administratorowi serwera, który dobiera zestaw biorąc pod uwagę konieczny poziom bezpieczeństwa związany z poufnością danych, szybkością szyfrów i ich dostępnością ze względu na amerykańskie ograniczenia eksportowe [1].

Jak już stwierdzono, protokół SSL opiera certyfikaty wydawane przez „wiarygodne" instytucje certyfikujące (tzw. wydawcy certyfikatów - CA), które za opłatą podpisują cyfrowo certyfikaty poszczególnych serwerów. Niestety, wprawdzie każdy elektroniczny certyfikat musi zostać podpisany przez CA, jednak nic nie stoi na przeszkodzie, aby zostać CA dla samego siebie.

Instytucje podpisujące certyfikaty serwerów oraz wydające certyfikaty osobiste różnią się przede wszystkim regulaminem (rodzaj serwera, wymagane dokumenty) oraz ceną. Najpopularniejszym i jednym z najdroższych CA jest VeriSign. Jego certyfikaty są akceptowane przez wszystkie wersje Netscape i MSIE. Innym popularnym CA jest firma Thawte Consulting, której certyfikaty są rozpoznawane przez wszystkie przeglądarki.

Ponadto certyfikaty wystawiane są m.in. przez następujące firmy: EuroSign, Computer Security Technologies, CompuSource, Nortel Entrust [16].

1.4. Podsumowanie

Podstawowym celem protokołu SSL jest zapewnienie prywatności i wiarygodności pomiędzy dwoma aplikacjami komunikującymi się między sobą. Na najniższym poziomie protokołu SSL, w górnej warstwie protokołu komunikacyjnego (np.: TCP), jest *SSL Record Protocol*, służący do upakowania (enkapsulacji) protokołów występujących na wyższym poziomie. Jak już stwierdzono, *SSL Handshake Protocol*, pozwala serwerowi i klientowi na wzajemną autentyfikację i negocjację algorytmu kodującego i kluczy kryptograficznych przed rozpoczęciem właściwej wymiany danych. Ważne cechy protokołu SSL to także:

- niezależność od protokołu aplikacji,
- **przeźroczystość** dla protokołów z wyższej warstwy.

Protokół SSL zapewnia bezpieczne połączenie, które ma trzy podstawowe właściwości: prywatność połączenia, szyfrowanie symetryczne (np.: DES, RC4, itd.) lub asymetryczne, wykorzystujące klucz publiczny (np.: RSA, DSS, itd.), oraz kontrolę wiarygodności połączenia, tzn. komunikaty sprawdzające integralność przekazu przy użyciu kodu

uwierzytelnienia wiadomości MAC (z ang. *Message Authentification Code*), do którego obliczeń wykorzystywane są bezpieczne funkcje haszujące (np.: SHA, MD5, itd.) [17].

2. Poziom bezpieczeństwa oferowany przez serwery SSL

2.1. Kryptografia to nie magiczna kula

W swojej książce „Secrets and Lies", Bruce Schneier pisze, że kiedyś sądził, że kryptografia jest rodzajem lekarstwa, które jest w stanie pomóc rozwiązać wszystkie problemy bezpieczeństwa. Teraz stwierdza, mając więcej doświadczenia, że kryptografia nie jest produktem, ale procesem, a w związku z tym rozwiązania takie, jak SSL, nie istnieją w próżni, dlatego warto spróbować pokrótce przeanalizować jak działa SSL.

HTTP (RFC-2616) nie zaspokaja, niestety, odpowiedniego poziomu bezpieczeństwa. Aby zapewnić bezpieczeństwo „nieuzbrojone" serwery WWW mogą zaoferować dwa podstawowe mechanizmy ochronne:

- odmowa połączenia użytkownikowi,
- umożliwienie autentyfikacji za pomocą nazwy użytkownika i hasła.

Jednak autentyfikacja w ten sposób przeprowadzona w żaden sposób nie gwarantuje ochrony hasła przesyłanego przez sieć lub nawet nie daje pewności, że użytkownik połączył się z właściwym serwerem, także dlatego, że każde dane chronione przy transmisji jedynie przez mechanizmy protokołu HTTP mogą zostać podsłuchane [1].

Od czasu, kiedy protokół HTTP stał się powszechnie wykorzystywanym protokołem w Internecie, opublikowanych zostało wiele publikacji dotyczących zapewnienia bezpieczeństwa interakcyjnej wymiany danych. Jednym z oficjalnych projektów sponsorowanych przez IETF (*Internet Engineering Task Force*) był tzw. *bezpieczny HTTP* (*Secure HTTP, SHTTP, RFC-2660*) [11]. Powstało także wiele nieoficjalnych projektów zabezpieczeń, a jednym z rozwiązań stworzonych początkowo jedynie na potrzeby wewnętrzne jest **SSL – rozwiązanie kryptograficzne firmy Netscape** [10], które **nie stało się nigdy tak naprawdę żadnym oficjalnym standardem bezpieczeństwa**, będąc jedynie **projektem** RFC opublikowanym przez firmę Netscape [6].

Zarówno SHHTP, jak i SSL jako protokoły mają do wykonania cztery zadania:

- autentyfikację serwerów;
- ochronę poufności wymienianych informacji;
- ochronę integralności przesyłanych danych.
- autentyfikację klientów względem serwerów [1].

2.2. (Nie)świadome korzystanie z SSL

Wykorzystywanie protokołu SSL rozpoczynamy już w momencie kliknięcia na adres rozpoczynający się *https* zamiast *http*. Przeglądarka rozpoczyna komunikację przez port 443, tzn. domyślny port SSL (zamiast portu do połączeń HTTP - nr 80), automatycznie przechodząc do procesu negocjacji połączenia SSL z serwerem za pomocą tzw. komunikatu powitalnego (z ang,. *Hallo message*). W komunikacie tym klient zawiadamia serwer, które z rodzajów szyfrowania i kompresji obsługuje. Jeśli serwer obsługuje SSL, odpowiada swoim własnym komunikatem powitalnym, który zawiera wybrane przez niego do dalszej komunikacji metody szyfrowania i kompresji, certyfikat serwera oraz zestaw losowo wybranych liczb, używanych później do kodowania transmisji (przeznaczonych do inicjacji generatorów).

Klient weryfikuje certyfikat otrzymany od serwera sprawdzając, czy *nazwa własna certyfikatu* przechowywana wewnątrz certyfikatu (ang. *common name*) odpowiada serwerowi, z którym chciał się połączyć i czy certyfikat jest jeszcze ważny (tzn. nie nastąpiło jego przedawnienie). Proces sprawdzania ważności certyfikatu obejmuje kontrolę tzw. podpisu cyfrowego (z ang. *digital signature*) przy użyciu odpowiedniego rekordu odnalezionego na liście kluczy publicznych CA, która jest dostarczana razem z przeglądarką. Lista taka może być także zmodyfikowana później samodzielnie przez użytkownika lub zaimportowana nowa - jeśli zachodzi taka potrzeba.

W przypadku gdy:

- certyfikat jest nieważny,
- nie ma jego nazwy na liście CA (oczywiście, użytkownik może ją zawsze „uaktualnić"),
- nie odpowiada nazwie serwera, z którym chcieliśmy się połączyć,

przeglądarka klienta **może (!)** zakończyć połączenie.

Jeżeli wynik sprawdzenia jest pomyślny, to przeglądarka klienta wysyła swoją własną liczbę losową, i w zależności od wybranego klucza szyfrującego wymianę informacji, wysyła także opcjonalnie inne „tajne wartości", określające np. wersję używanego szyfru itd. Przesłane dane zostaną teraz użyte do generacji kluczy głównych i innych pomocniczych wartości wykorzystywanych podczas wymiany danych. Od tego momentu wszystkie przesyłane dane będą szyfrowane i „podpisane elektronicznie" (w celu zapewnienie ich poufności i integralności).

Serwer może także zażądać certyfikatu od klienta, w celu autentyfikacji klienta względem serwera, ale jest to jedynie **opcja** (rzadko stosowana). Jeśli faza negocjacji zakończy się sukcesem, przeglądarka wysyła serwerowi tzw. komunikat gotowości do transmisji (z ang., *Finished Message*) i teraz obydwie strony dialogu są w stanie rozpocząć wymianę zaszyfrowanych i elektronicznie podpisanych danych [1,6].

Niestety, przygotowanie i wymiana kluczy w celu rozpoczęcia właściwej „bezpiecznej" sesji są dość czasochłonne w porównaniu z innymi operacjami wykonywanymi przez serwer. Przykładowo w warunkach laboratoryjnych serwerowi wyposażonemu w procesor Pentium III 550 Mhz jedna wymiana kluczy w przypadku użycia algorytmu RSA z kluczem 1024 (w środowisku Windows NT) zabiera ok. 24 ms, (czyli średnio 40 razy na sekundę). Wprawdzie wartość ta jest niewielka, jednak w realnych warunkach może być znacznie niższa, tzn. ok. 10-20 wymian na sekundę. W przypadku wykorzystywania SSL czas na określenie kluczy do wymiany jest ograniczony od dołu także liczbą nowych połączeń do zajętego serwera internetowego. Protokół SSL pozwala jednak w pewnych przypadkach na redukcję niezbędnego czasu wymiany kluczy. Jeżeli przeglądarka WWW nawiązuje kolejne połączenie do tego samego serwera SSL, to może przesłać tzw. identyfikator sesji (z ang., *session identifier*), który użyje uprzednio wynegocjowanych warunków wymiany danych [1].

2.3. Problem bezpieczeństwa serwerów SSL

W opracowaniu mówiąc o SSL będę miał na myśli wszystkie jego wersje, tzn. SSLv2, SSLv3 i TLSv1. Warto dodać, że pomimo szerokiego rozpowszechnienia tylko TLSv1 doczekał się oficjalnego projektu określającego jego specyfikację (RFC-2246).

Jak już wspomniano, „bezpieczne" serwery wykorzystują *Secure Sockets Layer* (SSL) w celu utworzenia bezpiecznego kanału transmisyjnego pomiędzy klientem i serwerem. Można sobie więc wyobrazić, że SSL tworzy „wirtualny kanał komunikacyjny" pomiędzy klientem i serwerem, który ochrania dane.

Często w literaturze pisząc o SSL, jako osobny wyodrębnia się protokół negocjacyjny SSL (*SSL handshake*), który w rzeczywistości nie jest niczym innym, jak schematem wczesnej fazy połączenia dokonującym uwierzytelnienia stron (patrz 2.2) i uzgadniającym zestawy szyfrujące przez nie wykorzystywane w trakcie dalszej transmisji.

Dla przypomnienia, podstawowe kroki podczas procesu negocjacji to:

- określenie zestawu szyfrów, które będą używane podczas nawiązywanego połączenia;
- autentyfikacja serwera względem klienta, wymiana danych koniecznych do dalszej komunikacji (do generacji kluczy symetrycznych) przy użyciu asymetrycznego szyfru „wynegocjowanego" w poprzednim kroku;
- **opcjonalna** autentyfikacja klienta względem serwera;
- rozpoczęcie przesyłania danych opartego na symetrycznym szyfrowaniu [8].

Warto zwrócić uwagę na fakt sygnalizowany ostatnio ponownie przez Rika Farowa [1], autora artykułów dotyczących bezpieczeństwa transakcji w Internecie oraz Erica Murraya [9], amerykańskiego specjalistę ds. bezpieczeństwa. Podobnie jak inne problemy bezpieczeństwa, jak np. usterki w pakietach szyfrujących, **problem nie leży w samym**

protokole SSL, ale w oprogramowaniu służącym jego implementacji, tzn. tzw. „oprogramowaniu wspomagającym". Wprawdzie konstrukcja protokołu SSL została przemyślana bardzo dokładnie, jednakże nie można stwierdzić, że nie ma błędów. Istotnym problemem może być długość kluczy, które mogą być złamane za pomocą tzw. brutalnej siły (z ang., *brute force*), tzn. odgadnięte poprzez próbę wszystkich możliwych kombinacji [1]. Niestety, międzynarodowe wersje przeglądarek Netscape 4.0 i Microsoft Internet Explorer 4.0 używają jedynie kluczy 40-bitowych ze względu na amerykańskie ograniczenia eksportowe [2]. Podobnie jest także z wersjami Linux-a dostarczanymi z USA, takimi jak np. RedHat, które także mają długość klucza ograniczoną do 40 bitów.

3. Test serwerów e-commerce

Eric Murray, konsultant ds. bezpieczeństwa, przeprowadził badanie ponad 8000 serwerów za pomocą swojego programu, który łączy się z serwerem SSL, otrzymuje jego certyfikat i określa jego poziom bezpieczeństwa. W teście wykonanym w lipcu ubiegłego roku okazało się, że 31,53% miało niski poziom bezpieczeństwa, 9,84% średni poziom, a tylko 58,63% było naprawdę bezpieczne [9]. Poniżej pokrótce opiszę jego wyniki badań.

Poniżej przed prezentacją rezultatów omówione zostaną główne przyczyny słabego poziomu zabezpieczenia serwerów, które jak wykaże rozdział, wynikają także z nieprzemyślanej implementacji samego protokołu SSL.

3.1. Problem zbyt krótkich kluczy

Z uwagi na amerykańskie ograniczenia eksportowe wszelkie oprogramowanie użytkowe wyprodukowane przed 1998 rokiem nie mogło wykorzystywać silniejszego kodowania niż:

- 40-bitowych kluczy w przypadku szyfrowania symetrycznego;
- 512-bitowych kluczy w przypadku szyfrowania asymetrycznego [2].

Powszechnie wiadomo, że klucz 40-bitowy nie jest zabezpieczeniem zapewniającym wysoką odporność na ataki. Jako dowód, jak słabe jest to zabezpieczenie, przytoczę kilka przykładów. W 1996 roku, student francuski Damien Doligez, przy użyciu sieci bezczynnych stacji roboczych zdołał za pomocą tzw. „brutalnego łamania kluczy" odgadnąć klucz SSL, wykorzystywany przez jeden z banków francuskich. Wyszukanie zajęło jego sieci 8 dni [3].

W tym samym roku za pomocą tego samego sposobu złamano tzw. „podwójnie zmienny szyfr Hala", co zabrało około 32 godzin [1].

W 1998 roku grupa wspomagana przez EFF zbudowała tzw. „Deep Crack", maszynę do łamania szyfru DES. Komputer ten potrafił złamać 56-bitowy klucz DES w niecałe 3 dni [4].

Jeżeli weźmiemy pod uwagę ilość możliwych kombinacji w przypadku klucza 56-bitowego 2^{56} (tj. około 72.057.590.000.000.000) i porównamy tą liczbę z przestrzenią 40-bitowego klucza, to dojdziemy do wniosku, że „Deep Crack" złamie klucz 40-bitowy w mniej niż 12 godzin(!) [9].

Niestety, także kryptografia asymetryczna nie oferuje 100% bezpieczeństwa, ponieważ także nie jest odporna na „brutalny atak". Najbardziej znanym przypadkiem było złamanie szyfru *RSA Crypto Challenge*. Najnowszym było odkrycie wpływu dwóch liczb pierwszych na konstrukcje 512-bitowego klucza RSA. Pomimo dużej odporności na złamanie – „odgadnięcie" klucza zabrało 5,2 miesiąca przy użyciu 292 komputerów rozrzuconych w sieci Internet – kod ten nie można uznać za 100% bezpieczny. Wynik tego testu spowodował, że w chwili obecnej firma *RSA Labs* rekomenduje używanie „bezpiecznych" kluczy o długości co najmniej 768 bitów [7].

Część niezależnych ekspertów uważa, że maszynę podobną do „Deep Crack", tyle, że o wiele bardziej technologicznie zaawansowaną, posiada i użytkuje rząd USA, jednak w chwili obecnej żadna z agend rządowych nie potwierdza tej informacji, ani jej nie zaprzecza.

3.2. Stosowanie przestarzałej wersji protokołu SSL 2.0

Z uwagi na to, że test wykrył całkiem sporą liczbę serwerów wykorzystujących SSLv2, wydaje się pożądane wymienienie trzech możliwych sposobów wykorzystania nieprawidłowości występujących w tym protokole:

- atak aktywny: zmiana (przekłamanie) bitów podczas fazy negocjacji pomiędzy klientem a serwerem powoduje użycie tzw. słabszych kluczy, co ułatwia późniejszy, właściwy atak;
- atak polegający na skracaniu przesyłanych komunikatów, a przez to przekłamanie ich treści (np. „zakup 10.000 akcji PBG" zamiast „zakup 10.000 akcji PBGX");
- możliwość przerwania sesji pomiędzy klientem a serwerem i „podszycia się pod jedną ze stron" (trudne, aczkolwiek nie niemożliwe) [9].

3.3. Uwierzytelnianie własnych certyfikatów

SSL wykorzystuje model certyfikacji zgodny z normą X.509. Model prezentowany w tej normie ma budowę hierarchiczną, który na samym szczycie zawiera tzw. Wystawcę Certyfikatu (CA – *Certificate Authority*), a każdy inny CA lub serwer jest uwierzytelniany przez swoich poprzedników (znajdujących się powyżej w łańcuchu). Inaczej: każda jednostka podpisuje certyfikat jednostkom podrzędnym (znajdującym się „poniżej" w łańcuchu. Element na szczycie struktury hierarchicznej ma własność polegającą na tym, że sam podpisuje sobie certyfikat, a jak już zostało to dokładnie wyjaśnione, jednostki podrzędne

(w tym przypadku klienci SSL) mają listę wystawców certyfikatów (węzły), których uważają za „godnych zaufania" [5].

Większość przeglądarek jest dostarczanych z długą listą CA umieszczoną w specjalnej bazie „zaufanych wystawców certyfikatów". Problem polega na tym, że większość, ale nie wszystkie serwery SSL wykorzystują certyfikaty wydane przez CA. Z uwagi na to, że sam mechanizm kontroli certyfikatów pomiędzy klientem/serwerem został już w tym artykule dość dokładnie omówiony (patrz rozdz. 2.2), przejdziemy od razu do kwestii związanych z uzyskaniem certyfikatu (patrz także rozdz. 1.3).

W przypadku gdy operator serwera chciałby uruchomić sklep internetowy, powinien wysłać prośbę o uzyskanie certyfikatu do wybranego CA, który podpisuje klucz publiczny serwera i odsyła go z powrotem do wykorzystania na serwerze. Biorąc pod uwagę, że operacje wykonywane przez CA mają pewien związek z ryzykiem prawnym, większość CA pobiera opłatę za swoje usługi.

Jeżeli operator serwera SSL chce uniknąć opłaty za uzyskanie certyfikatu wydanego przez CA, może skonfigurować serwer tak, że będzie on sam podpisywał (uwierzytelniał) swoje certyfikaty. Wszystko, co do tej operacji potrzebuje, to oprogramowanie typu shareware, freeware lub public domain. W tym przypadku żaden CA nie „ręczy" za bezpieczeństwo serwera. Oczywiście każdy może przygotować certyfikat podpisany przez samego siebie (z ang., *self-signed certificate*). Jednakże **tego rodzaju certyfikat nie ma żadnej wartości** (zobacz np. [12]). Serwer jest wtedy sam dla siebie CA i wygląda to tak, jak gdyby operator serwera mówił do siebie: „Tak, ufam sobie" [9].

Przeglądarki akceptują certyfikaty serwerów podpisane przez operatorów serwera, uprzedzając o tym użytkownika jedynie za pomocą okienka dialogowego w rodzaju „Czy jesteś pewny, że chcesz kontynuować?", czyli użytkownik nie jest w sposób jednoznaczny informowany o tym, że dzieje się coś niezwykłego (!). Warto dodać, że o ile w okienku dialogowym przeglądarek Netscape i Internet Explorer jest informacja mówiąca, że jest to certyfikat serwera podpisany przez operatora serwera, ale nie ma żadnego wyjaśnienia, że istnieje dość znaczne ryzyko w akceptowaniu tego rodzaju certyfikatów (a który z normalnych użytkowników wybiera przycisk „Więcej informacji"?). Ale jest napisane, że należy zaakceptować certyfikat, aby móc kontynuować [13]. Większość użytkowników skonfrontowanych z okienkiem dialogowym, którego prawdziwego znaczenia nie rozumieją po prostu wybiera OK, ale chyba nie o to chodzi...

3.4. Wykorzystywanie „przedawnionych" certyfikatów

Konstrukcja kluczy osadzonych w certyfikatach jest taka, że mają one ograniczony czas ważności – nawet jeśli są one (lub powinny być) odpowiednio długie i w związku z tym sprawdzanie wszystkich możliwych kombinacji nie jest opłacalne ze względów czasowych, to im dłużej są one wykorzystywane, tym większa jest szansa, że zostaną złamane lub ktoś przypadkowo lub celowo odkryje ich konstrukcję.

W związku z powyższym, w imię zapewnienia maksymalnego bezpieczeństwa, powinno się stosować zasadę okresowej zmiany klucza. Warto tu dodać, że zgodnie z modelem X.509 (patrz rozdz.3.3 oraz [5]), każdy certyfikat wygasa samoczynnie po pewnym ustalonym czasie. Operator serwera jest zobowiązany do uzyskania nowego certyfikatu zanim „stary" utraci swoją ważność oraz do rekonfiguracji serwera w celu wykorzystywania nowego certyfikatu i klucza. Niestety, nie każdy operator o tym pamięta.

W przypadku gdy przeglądarka napotka certyfikat serwera, którego „okres ważności" już się skończył, jest pokazywane specjalne okienko dialogowe z informacją, że wykorzystywany certyfikat wygasł lub data systemowa jest nieprawidłowa, ale nie ma żadnego wyjaśnienia, że istnieje dość znaczne ryzyko w akceptowaniu tego rodzaju certyfikatów.

4. Wyniki i podejrzenia

Zapytanie testowe zostało wysłane do 9754 serwerów, uważanych za bezpieczne, z czego 8189 odpowiedziało, a 8081 potrafiło przekazać swoje certyfikaty i tylko wyniki uzyskane od tych ostatnich są brane pod uwagę [9].

Poniżej zostanie krótko przedstawiona metodologia testu oraz przyjęte kryteria oceny jakości serwerów w rozumieniu poziomu prezentowanego przez nie bezpieczeństwa.

4.1. Klasyfikacja wyników testu

4.1.1. Serwery o niskim poziomie bezpieczeństwa

W teście przyjęto, że „słabe" klucze serwerów to takie, które są oparte o RSA o długości 512 bitów lub mniejszej. Wszystkie serwery poddane testowi wykorzystywały kod RSA.

Za „słaby" serwer uważać się będzie w teście taki, który posiada słabe klucze, obsługuje jedynie dawno złamany protokół SSLv2, posiada certyfikaty podpisane przez samego siebie. lub certyfikaty, których ważność wygasła, oraz taki, który jest w stanie obsługiwać tylko „słabe" zestawy szyfrów.

Prawie każdy z przetestowanych serwerów (99% dla SSLv2, 79% dla SSLv3 i 75% dla TLS1) obsługiwał co najmniej jeden „słaby" klucz. Przeglądarki, które zostały skonfigurowane do wykorzystywania tylko słabych zestawów szyfrów, będą szczególnie podatne na atak [9]. Artykuł ten nie zajmuje się liczbą wykorzystywanych tzw. słabych przeglądarek, jednak należy oczekiwać, że w Polsce oraz na całym świecie (z wyjątkiem kontynentu amerykańskiego), z uwagi na wykorzystywanie tzw. międzynarodowych wersji przeglądarek (z ang. *international versions*) o obniżonym poziomie bezpieczeństwa z uwagi na ograniczenia eksportowe, ich liczba jest duża i prawdopodobnie w znaczny sposób wpłynęłaby na liczbę „słabych" sesji SSL, poza 32% wspomnianymi w niniejszym opracowaniu. W dużych firmach do dziś na dziesiątkach komputerów wykorzystuje się IE 4.0 (z systemu WIN NT). Dlaczego? Często na instalację nowej wersji na wielu stacjach roboczych po prostu brak czasu lub ktoś nie uznał tego za konieczne i niezbędne.

4.1.2. Serwery o średnim poziomie bezpieczeństwa

W teście za serwery o „średnim" poziomie bezpieczeństwa uważa się serwery wykorzystujące klucz publiczny dłuższy, bądź równy 512 bitów, oraz obsługujące „silne" zestawy szyfrów (te z kluczem symetrycznym mają rozmiar większy niż 40 bity).

4.1.3. Serwery o wysokim poziomie bezpieczeństwa

Za „silne" serwery, na potrzeby artykułu, będziemy uważali takie serwery, które są skonfigurowane do pracy z kluczem publicznym większym bądź równym 1000 bitów, oraz obsługują zestawy szyfrów z kluczem symetrycznym o rozmiarze większym niż 64 bity.

4.2. Szczegółowe omówienie wyników testu oraz próba wskazania powodów zaistniałej sytuacji

Na 8081 serwerów, które zwróciło certyfikaty, 2709 (tj. 31,53%) oferowało relatywnie niski poziom bezpieczeństwa, 161 (tj. 9,84%) średni, a tylko 4271 (tj. 58,63%) wysoki poziom.

W świetle przeprowadzonego testu serwery uważane w artykule za „słabe" to takie, które:

- wykorzystywały tylko 40-bitowe zestawy szyfrów (27,7% słabych serwerów),
- wykorzystywały przestarzałą wersję protokołu SSLv2 (1,2%),
- miało nieważny (wygasły) certyfikat (9,9%),
- używało certyfikatów podpisanych przez samych siebie (2,8%).

Oczywiście, część z serwerów oferujących niski poziom bezpieczeństwa miało więcej niż jeden słaby punkt (!) [9].

4.2.1. Zestawy szyfrów 40-bitowych

Znacząca część, tj. 27,7%, słabych serwerów odkrytych w teście używało tylko słabych zestawów szyfrów, co oznacza, że nie nawiązywały one połączeń SSLv3 lub TLS1 przy użyciu innych niż zestawy szyfrów oferujące niski poziom bezpieczeństwa [9].

Warto przypomnieć, że krótkie klucze symetryczne są relatywnie proste do złamania przy użyciu brutalnej siły. Rozmiar klucza nie jest już ograniczony przez amerykańskie ograniczenia eksportowe, które uległy zmianie w grudniu 1998 i od tego czasu produkty zawierające 56-bitowe klucze symetryczne i oferujące standard TLS mogą być legalnie wykorzystywane na całym świecie. Ponadto wiele darmowych wersji SSL-a także oferuje 56-bitowe i lepsze zestawy szyfrów. Przeglądarki, które są w stanie używać tylko 40-bitowych kluczy, będą jednak negocjować połączenie SSL nawet jeśli serwer będzie obsługiwał chociaż jeden 40-bitowy zestaw szyfrów [8] (!).

Niestety, z uwagi na jakiekolwiek prawne ani techniczne przyczyny, wykorzystywanie przez serwery tylko 40-bitowych kluczy nie może być usprawiedliwione.

4.2.2. Wykorzystywanie tylko SSLv2

Modele ataków, na które jest podatny SSLv2, są opisane w rozdziale 3.2.

4.2.3. Słabe klucze serwera

Większość serwerów ze słabymi kluczami (max 512 bitowymi) wykorzystywało także klucze 40-bitowe, tzn. ok. 83% serwerów wykorzystuje ten sam poziom bezpieczeństwa jeśli chodzi o klucze serwera i zestawy szyfrów. Nawet, jeśli złamanie słabego klucza RSA zabiera więcej czasu, to jeżeli napastnikowi raz się udało, to wszystkie przesyłane dotąd informacje do i z serwera mogą zostać zdekodowane, a jeżeli napastnik prowadził „nasłuch" przez wystarczająco długi czas, skutki mogą być nie do przewidzenia.

Nie będzie tak w przypadku sesji, które zapewniają *Perfect Forward Secrecy*. Niestety, PFS nie jest wykorzystywana przez większość serwerów (najbardziej popularnym zestawem szyfrów jest obsługiwany przez 26% serwerów SSLv3 SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA) [9].

4.2.4. Samodzielnie podpisane certyfikaty

Serwery o samodzielnie podpisanych certyfikatach to najczęściej serwery testowe. Część z liczby przetestowanych serwerów jest zatem prawdopodobnie serwerami testowymi lub serwerami skonfigurowanymi do pracy stosunkowo niedawno, które nie były jeszcze w prawdziwym użyciu. Niestety, nie jest możliwe programowe określenie, czy serwer jest serwerem testowym czy „prawdziwym". Warto dodać, że większość przetestowanych serwerów były „online" wystarczająco długo (kilka dni), aby znalazła je wyszukiwarka internetowa [9].

4.2.5. Przedawnione certyfikaty

Nawet jeśli nie ma bezpośredniego, prostego ataku na serwery z wygaśniętymi (lub samodzielnie podpisanymi) certyfikatami, to „przyzwyczajają" one użytkowników do akceptowania stanu, który nie powinien występować. Możliwe jest także, że znalezione serwery to nie utrzymywane „martwe" serwery lub tzw. kopie lustrzane (z ang., *mirrors*) już nie utrzymywanych serwerów, ale kiedyś „uwierzytelnione" certyfikatami serwery.

4.3. Interesująca statystyka

Przetestowano 8189 serwerów, certyfikaty zwróciło 8081.

Klasyfikacja wg poziom bezpieczeństwa serwerów: niski poziom 2548 (31,53%); średni poziom 795 (9,84%); wysoki 4738 (58,63%).

Serwery uznane za oferujące niski poziom bezpieczeństwa - przyczyny: wygaśnięte certyfikaty 254 (9,97%); samodzielnie podpisane certyfikaty 71 (2,79%); wykorzystywanie tylko SSLv2 30 (1,18%); klucz serwera krótszy lub równy 512 bitów 2060 (80,85%); słabe zestawy szyfrów SSLv3/TLS 707 (27,75%).

Długości kluczy serwerów: 2048 bitów-> 15 (0,19%); 1024 bity -> 5653 (69,95%), 1023 bity-> 142 (1,76%); 950 bitów -> 2 (0,02%); 768 bitów -> 194 (2,40%); 700 bitów -> 2 (0,02%); 512 bitów ->1870 (23,14%); 511 bitów -> 190 (2,35%); pozostałe inne ok. 0,01% [9].

5. Kłopoty producentów przeglądarek naszymi kłopotami

Mówiąc o producentach przeglądarek mamy na myśli zwykle dwie, duże, konkurencyjne wobec siebie firmy: Microsoft i Netscape. Wprawdzie istnieją także inne przeglądarki (np. Opera, Amaya, Neoplanet, Genius, Ariadne Browser itd.), ale większość użytkowników na całym świecie korzysta albo z Internet Explorera (najnowsza wersja 5.5) albo z Netscape Navigatora (najnowsza wersja 6.0). Jednak nawet tacy giganci, jak Microsoft i Netscape, nie potrafili ustrzec się od błędów.

Wprawdzie Microsoft zastrzega się, że w najnowszych wersjach oprogramowania błąd ten został usunięty, ale... Jak się okazało IIS (*Internet Information Server*), zarówno wersje 4 jak i 5 wysyłają te same wartości sesyjne zawarte w cookies, dla połączeń SSL oraz zwykłego HTTP, co jest tylko jednym z problemów wykrytych przez Mitje Kolseka (ACROS, Slovenia), bowiem odkrył on także metodę, w jaki sposób można „ukraść hasła internetowe" z IE (zobacz [18]).

Przesłane nazwa użytkownika i hasło są częścią *HTTP Basic Authentification Scheme*, tzn. oczywiście, są one wymieniane za pośrednictwem protokołu SSL. Jednak w przypadku gdy ponownie odwiedzimy tę samą stronę bez SSL-a, nasza nazwa użytkownika i hasło zostaną przesłane nie zaszyfrowane, potencjalnie dostępne dla zainteresowanych. Microsoft zamieścił na swojej stronie rady i poprawki, jako elementy pomocne w rozwiązaniu tej „dziury" w bezpieczeństwie [18].

W maju ubiegłego roku Kevin FU, student MIT, odkrył, że Navigator może zostać oszukany poprzez akceptowanie nieprawdziwych certyfikatów. Jak – warto odwiedzić stronę (zobacz [12]). Ale to nie wszystko: w listopadzie ubiegłego roku Fu odkrył interesujące zachowanie oprogramowania dla serwerów proxy AKAMAI. Oprócz podwyższania szybkości dostępu do poszczególnych stron zachowuje się w nieco dziwny sposób – żądanie dostępu do strony FU przesłane za pośrednictwem serwera AKAMAI wraz z żądaniem certyfikatu spowodowało, że certyfikat rzeczywiście nadszedł, **ale z serwera PROXY, a nie właściwej strony**. Zatem oszukanie użytkownika przestało być problemem. Oczywiście, firma AKAMAI usunęła ten problem, ale często zdarza się tak, że duże serwery e-commerce używają dodatkowych serwerów SSL proxy, które w najlepszym przypadku znajdują się tylko kilka metrów od właściwych serwerów, w najgorszym nieco dalej [1,12]. Warto też pamiętać, że nawet w przypadku jednokrotnego przesłania danych (np. nr karty kredytowej), możemy być narażeni na ich kradzież - jeśli nie w trakcie przesyłu pomiędzy naszą przeglądarką i serwerem SSL, to z jednej ze źle chronionych stron.

6. Domyślne ustawienia przeglądarek WWW

SSL nie jest niczym niezwykłym, ale jednak trzeba wiedzieć, jak należy zamknąć drzwi, by nie wpuścić nieproszonych gości. Jak się jednak okazuje, producenci przeglądarek nie postanowili nam ułatwiać zadania.

Jak już powiedziano, Internet Explorer (w wersji 4.0 oraz IE 5.00.2614.3500 z 1999r.) dysponuje 40-bitową siłą szyfrowania, ale po sprawdzeniu sekcji *Zabezpieczenia* w wersji 5.5 (*Narzędzia->Opcje internetowe-> Zaawansowane->* sekcja *Zabezpieczenia*) okazuje się, że domyślnie:

- zezwala się na wykorzystywanie przestarzałej wersji SSL-a (SSL v2),
- opcje: „Sprawdź, czy certyfikat serwera nie został cofnięty" oraz „Sprawdź, czy certyfikat wydawcy nie został cofnięty" są nieaktywne.

Podobnie jest w przeglądarce Netscape Communicator w wersji 4.74 (*Communicator->Narzędzia->Informacje o ochronie*), która domyślnie zezwala na użycie oprócz SSLv2 oraz szyfrów RC4 i RC2 128-bitowych także na DES 56b oraz RC2 i RC4 40-bitowy (!).

W ten sam sposób zachowują się mniej popularne, aczkolwiek obecne na rynku przeglądarki, tzn.:

- Neoplanet v 5.00,
- Opera 5.0.

Zatem oprócz posiadania najnowszej wersji przeglądarki ważne jest, aby we właściwy sposób skonfigurować jej parametry.

7. Podsumowanie

Należy zdać sobie sprawę z faktu, że wprawdzie SSL podwyższa bezpieczeństwo transakcji zawieranych w Internecie, to nie jest to rozwiązanie idealne. Słabe szyfrowanie informacji, przypadkowe ujawnianie zastrzeżonych informacji takich, jak hasła lub sesyjne cookies, przypadki błędnej identyfikacji powodują osłabienie wrażenia, że SSL jest w 100% bezpiecznym rozwiązaniem. Możliwe jest także, że SSL, mimo stosunkowo długiej kariery, ma także inne, wciąż nierozpoznane, słabości. Jak wiadomo, polega on na przeglądarkach WWW, posiadających wstępnie zainstalowaną listę głównych Wystawców Certyfikatów. Ale to, czego używane przez nas przeglądarki domyślnie nie wykonują, to sprawdzenie, czy któryś z preinstalowanych certyfikatów nie został wycofany. W przypadku gdy użytkownik certyfikatu utraci kontrolę nad kluczem prywatnym związanym z tym certyfikatem, wtedy certyfikat nie może być już dłużej wykorzystywany i musi zostać wycofany. Ale istniejące dziś przeglądarki nie posiadają wbudowanego mechanizmu do uchylania certyfikatów. Wprawdzie jest to rzadkie zjawisko, ale występuje (zobacz CA-2000.19) [1,14].

Jednak rozwiązaniem problemu nie powinno być całkowite odrzucenie protokołu SSL, ale raczej unikanie słabości na miarę dzisiejszych możliwości. I właśnie dlatego, jako użytkownik Internetu, pamiętam, że widok zamkniętej kłódki wcale nie oznacza, że przesyłana informacja jest naprawdę bezpieczna, ale to, że istnieje duża szansa, że dane te trafią bezpiecznie poprzez Internet do właściwego serwera. Ponadto brak innych gwarancji....

LITERATURA

1. Farrow R.: SSL is not a magic bullet. Network Defense, styczeń 2001, www.spirit.com/Network/net1100.txt, www.linuxsecurity.com/articles/cryptography_article-2386.html
2. Bureau of Export Administration docket number 9809-11233-8318-02, http://jya.com.bxa123198.txt.

3. Doligez D.: I Broke Hal's SSL Chalenge, sierpień 1996, http://paullac.inria.fr/~doligez/ssl.
4. Cracking DES. EFF, 1998, http://www.eff.org/descracker.
5. Housley R., Polk W., Solo D.: RFC 2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile, styczeń 1999, IETF draft.
6. Freier A., Karlton P., Kocher P.: The SSL protocol Version 3.0, marzec 1996, IETF draft, http://home.netscape.com/eng/ssl3.
7. RSA Labs: RSA Crypto Challenge Sets New Security Benchmark, http://www.rsasecurity.com/news/pr/990826.html
8. Schneier B., Wagner D.: „Analysis of the SSL 3.0 Protocol", kwiecień 1997, http://www.counterpane.com
9. Murray E.: SSL Server Security Survey, lipiec 2000, http://www.lne.com/eric/papers
10. The draft of SSL v.3.02, http://home.netscape.com/eng/ssl3/draft302.txt, http://www.netscape.com/eng/ssl3/
11. The Secure HyperText Transfer Protocol: http://www.faqs.org/rfcs/rfc2660.html
12. An example of a self-signed certificate: https://snafu.fooworld.org/
13. Inconsistent Warning Messages in Netscape Navigator: http://www.cert.org/advisories/CA-2000-08.html
14. CERT advisory about the revocation of some of Sun Microsystem's certificates: http://www.cert.org/advisories/CA-2000-19.html
15. Netscape Navigator Improperly Validates SSL Sessions: CERT Advisory CA-2000-05, maj 2000, http://www.cert.org/advisories/CA-2000-05.html
16. Krawczyk P.: SSL. Rozdz. 13.4 Bezpieczeństwo WWW, pl.comp.www FAQ http://www.qdnet.pl/~Ziembor/bezpieczenstwo04.html
17. Ośrodek Informatyczny PG: Zasady bezpieczeństwa... Protokół SSL (Secure Socket Layer), http://www.pg.gda.pl/OI/security/ssl/
18. Kolsek M. (MIT): Report of problems with session cookies sharing: http://www.microsoft.com/technet/security/bulletin/MS00-076.asp

Recenzent: Dr inż. Krzysztof Nałęcki

Wpłynęło do Redakcji 26 lutego 2001 r.

Abstract

The article describes the SSL protocol. While SSL does improve the security of electronic transactions over the Web, it is not a perfect solution. Weak encryption, accidental divulging

of secrets like passwords or session cookies, cases of mistaken identity, combine to weaken the impression that SSL is a secure solution. But SSL as it is used today has probably an even greater, and still unaddressed, weakness. SSL relies on Web browsers have a pre-installed list of top level Certificate Authorities. But what your browser does not do is check to see if any of these certificates has been revoked. If the owner of a certificate loses control of the private key associated with that certificate, the certificate can no longer be used, and must be revoked.

To check the practical level of security there was carried on test of 8081 randomly-selected secure (SSL) servers, 2548 (31.53%) were found to have weak security. Of those servers, the weakness was because of supporting only weak (40-bit) ciphersuites (27.75%% of weak servers), small (512 bits or less) RSA server keys (80.85%), use of only the flawed SSLv2 protocol (1.18%), expired server certificates (9.97%), self-signed server certificates (2.79%), or a combination of more than one of these weaknesses. These weaknesses make the transactions that are protected by these servers easy to attack with modern key-cracking techniques and/or hacking techniques. There is no technical or legal reason to limit secure servers to using only SSLv2 or to use self-signed or expired certificates. The US export laws that restricted exportable US software to 40-bit symmetric encryption and 512-bit public-key encryption were changed in December 1998, so there is no legal reason to support only 40-bit cipher suites or 512-bit RSA keys. The reasons why so many sites have unnecessarily weak security can only be guessed at.

Big producers of browsers do not help in this matter. The default settings in IE and Netscape Navigator allows to use SSLv2 as well as weak 40-bit ciphersuites.

Anyway the solution is not to toss out SSL, but rather to address the weaknesses in the ways it is currently used. And, as a Web user, to remember that seeing the closed padlock icon does not mean that your information is really secure, but that hopefully your data has been securely transported across the Internet to the correct server. Beyond that, there are no guarantees.