

Anonimowosc i bezpieczenstwo w sieci

Maciej Piasecki
Politechnika Bialostocka
ul. Wiejska 45a
macpias@poczta.fm

STRESZCZENIE

Artykul porusza bardzo wazny w dzisiejszych czasach temat bezpieczenstwa. Zostaly w nim przedstawione elementy, które sa niebezpieczne. Do nich naleza pliki Cookie, które pozwalaja sledzic uzytkownika i podobnie dzialajacy mechaizm Spyware. Druga grupa elementów sa elementy zabezpieczajace takie, jak Tunelowanie, protokól SSL. Artykul opisuje takze elementy zwiazane z anonimowoscia w sieci.

WPROWADZENIE

W dzisiejszych czasach coraz wiecej informacji przeplywa przez Internet. Korzystamy ze sklepów i ksiegarni internetowych. Wypelniamy swoimi danymi formularze. Podajemy swoje dane osobowe i numer karty kredytowej. Bedac podlaczonym do sieci jestesmy potencjalnymi ofiarami ze strony hakerów. Co prawda w domu rzadko sie zdarza, ze mamy jakies poufne dane, ale w firmach, gdzie sie tworzy projekty różnych aplikacji moze sie wlamac haker w celu skopiowania i późniejszej sprzedazy aplikacji uzyskujac korzysci majatkowe. Poza tym mozemy byc szpiegowani przez programy, które informuja producenta, co robilismy w sieci. Dlatego coraz bardziej zalezy nam na tym by byc skutecznie chronionym. Dlatego mój wyklad jest poswiecony bezpieczenstwu w sieci i tematowi z nim zwiazanym - anonimowosc.

REMAILERY

Zaczne od Remailerów, które sa przekaznikami pocztowymi. Sluza do wycinania z nagłówka listu informacji o koncie, z którego zostala wyslana dana wiadomosc. Stawianie Remailerów nie jest do konca legalne, gdyz uzywajac ich mozemy wyslac wiadomosci o różnej tresci pozostajac anonimowym. Z naszego Remailera moga skorzystac na przyklad przestepcy, albo ktos, kto chce zrobic mailbombing. A wtedy moze sie nami zainteresowac Interpol. Obecnie mamy doczynienia z dwoma typami Remailerów: Cyberpunk – jest typu 1, który wykorzystuje do szyfrowania mechanizm PGP, zostanie on omówiony w dalszej czesci wykladu i Mixmaster – remailer typu 2, który ma swój własny mechanizm kluczy oraz przesyłania wiadomosci. Poczta moze przechodzic przez jeden albo wiecej serwerów. W pierwszym przypadku to czy autor wiadomosci zostaje anonimowa zalezy od uczciwosci administratora. Natomiast w drugim przypadku wiadomosc jest przekazywana przez kilka serwerów. Wtedy kolejny serwer zna tylko adres poprzedniego serwera i adresata. Taki sposób przekazywania wiadomosci nazywamy lancuchowaniem. W celu wiekszego zabezpieczenia tresci wiadomosci mozemy zastosowac dodatkowo Ruting cebulowy, który polega na przyjeciu, ze kazdy remailer ma własny klucz szyfrujacy. Dzieki temu mozemy wypracowac nastepujacy schemat:

- 1) Wybieramy remailery

- 2) Tworzymy wiadomosc w postaci takiej, jaka ma odebrac odbiorca
- 3) Szyfrujemy ja kluczem ostatniego remailera w wybranym przez nas lancuchu
- 4) Orzymany szyfrogram przygotowujemy do przeslania jako anonimowa wiadomosc z adresatem – ostatnim remailerem i nadawca – przedostatnim remailerem lancuchu
- 5) Algorytm powtarzamy tyle razy, przez ile remailerów ma przejsc wiadomosc.

W ten sposób pierwszy remailer może poznać nadawcę wiadomości a ostatni adresata. Przy tym remailery w środku poznają tylko adres poprzedniego i następnego etapu. Nie poznają też treści wiadomości. Poza tym można wprowadzić opóźnienie w celu utrudnienia analizy ruchu. Obserwując ruch wiadomości można zaobserwować pewne prawidłowości. W celu wprowadzenia opóźnienia można wprowadzić – w przypadku remailerów typu pierwszego czas, po którym należy przekazać wiadomość. Podaje się go w nagłówku Latent-Time. Czas dla remailera typu drugiego czas jest przekazywany przez użytkownika i jest zaszyty w treści wiadomości. Wiadomości są wtedy kolejgowane i wysyłane po upływie zadanego czasu. Remailery mogą zmieniać ten czas. Określa się ile procent kolejki ma być wysyłane w ciągu godziny oraz ile powinno oczekiwać w kolejce. Zapobiega to atakowi polegającemu na splukiwaniu, czyli ktoś, kto zna ilość wiadomości, która ma zostać w kolejce, może wysłać do remailera ich tyle żeby wypchnąć je z kolejki wcześniej.

Jak już wspominałem wcześniej remailery dzielą się na dwa typy. Zaczne od omówienia remailera typu pierwszego.

Wiadomość do remailera pierwszego typu składa się z następujących części:

- Linii zawierającej dwa dwukropki, oznaczającej początek wiadomości;
- Nagłówka sterującego, podającego, dla kogo wiadomość jest przeznaczona;
- Pustej linii oznaczającej koniec nagłówków remailera i początek treści wiadomości;
- Samej wiadomości.

Taka wiadomość jest podatna na przechwycenie przez podsłuchującego przy wejściu do remailera. Z tego powodu remailery pierwszego typu obsługują szyfrowanie PGP. Żeby otrzymać te klucze należy do remailera wysłać komendę remailer-keys. Po zaszyfrowaniu wiadomości należy opatrzyć ją nagłówkiem informującym remailer o tym. Aby zabezpieczyć się przed programami śledzącymi ruch wchodzący do i wychodzący z remailera, stosuje się już wcześniej wspomniany ruting cebulowy.

Pierwszy remailer odbierze wiadomość, następnie odszyfruje ją, przetworzy i odrzuci pierwszy nagłówek, a w końcu wiadomość przesła pod adres wymieniony w nagłówku, czyli adres remailera lsd. Ten odszyfruje wiadomość i treść wiadomości przekaze pod wymieniony adres. W ten sposób można przygotować wiadomość, żeby przeszła przez dowolną ilość remailerów.

Każdy remailer po wysłaniu do niego wiadomości z tematem remailer-help odpowie listem zawierającym skróconą instrukcję obsługi, klucze PGP i tak zwany capstring (capabilities string), czyli opis remailera zrozumiały dla programu premail. Format ten to definicja elementu tablicy w PERLu, w której indeksem jest skrócona nazwa remailera, a wartość elementu to słowa kluczowe opisujące możliwości remailera.

Remailerem typu drugiego jest Mixmaster.

Pomimo, że wiadomości są zaszyfrowane, można je śledzić obserwując wiadomości odbierane, wysyłane przez reailer oraz ich rozmiar. Podsluchiawcz może porównać wielkość wiadomości odebranej przez reailer z rozmiarem wiadomości wysyłanej. Wiadomość wysyłana będzie mniejsza o nagłówki pocztowe oraz kilkadziesiąt bajtów nagłówka reailera. Poza tym treść wiadomości może zostać poznana przez podsluchiawcz lub administratora ostatniego reailera w łańcuchu, gdy nadawcy wiadomości nie szyfrują jej.

Reailery typu drugiego zostały opracowane w celu zabezpieczenia wiadomości przed analizą ruchu. Wiadomości w formacie Mixmaster są wysyłane i odbierane przez ten sam program, który przygotowuje je do wysyłki oraz odbieraniem i deszyfrowaniem. Ten sam program jest używany do prowadzenia reailera. Ze względu na funkcjonowanie Mixmastera należy często odświeżać jego pliki konfiguracyjne, zawierające listy reailerów i ich klucze.

Komunikacja z reailerami

Reailery obsługują następujące polecenia:

- Reailer-help – odesła list z pomocą dotyczącą używania reailera;
- Reailer-config – odesła list z opisem konfiguracji reailera, jego kluczami PGP i Mixmaster;
- Reailer-stat – odesła statystykę obciążenia reailera;
- Żeby zablokować wysyłanie wiadomości z reailera na nasz adres należy wysłać do reailera list o treści destination-block
- Nadużycia skierowane na nasz adres można zgłaszać na Reailer Abuse Blocklist (<http://www.paracrypt.com/reailerabuse/>). Jednak nie wszystkie reailery korzystają z tej listy.

Narzędzia:

- Preamail – jest to pierwszy program do kompleksowej obsługi komunikacji za pomocą reailerów
- aircrypt – dodatek do Emacs'a obsługujący szyfrowanie poczty za pomocą PGP i S/MIME oraz -reailery;
- anubis – jest to polski program służący do anonimowego wysyłania poczty;
- mixmaster – program ten służy do obsługi wiadomości typu 2.

Dosć łatwo jest zainstalować reailer na koncie pocztowym. Wystarczy dostęp do shella. Same prowadzenie jednak nie jest łatwe. Trzeba uzyskać zgodę dostawcy łącza, mieć komputer o dość sporej mocy i sporo czasu na administrację. Poza tym trzeba brać pod uwagę, że jesteśmy częściowo odpowiedzialni za ludzi korzystających z sieci reailerów. Wyczerpujące informacje o prowadzeniu reailera znaleźć można pod adresem <http://lexx.sbinn.net/faq/>.

TUNELOWANIE

Coraz częściej stykamy się z problemem zapewnienia poufności i integralności danych przesyłanych w sieci. Niestety powszechna praktyka staje się przeglądanie pakietów w węzłach sieci. Opowiem to na przykładzie poczty, która odbieramy ze zdalnego serwera. Chcielibyśmy, żeby ani treść, ani nadawca i odbiorca, a co najważniejsze nasza nazwa użytkownika i hasło na tym serwerze nie zostały przechwycone przez niepowołane osoby. W tym celu możemy zastosować następujące rozwiązania:

- VPN – działa poniżej warstwy aplikacyjnej, wymaga wsparcia ze strony jądra systemu

operacyjnego. W celu działania VPN buduje relacje wzajemnego zaufania między maszynami lub sieciami. Niestety czasami zaufanie do jednej maszyny, na którą się mogą logować różni użytkownicy jest sporo luka w bezpieczeństwie;

- Różne mechanizmy bezpiecznego uwierzytelniania hasła. Niestety wtedy tylko jest chronione nasze hasło, a nie treść korespondencji. Poza tym mnogość protokołów może spowodować, że nasz MUA i serwer nie będą miały wspólnego protokołu uwierzytelniania hasła;

- Szyfrowanie i podpisywanie listów za pomocą S/MIME lub PGP jest całkiem dobrym rozwiązaniem, gdyż zabezpiecza treść na całej trasie przesyłki. Wymaganiem jest, żeby odbiorca znalazł ten protokół. Poza tym trzeba wymienić klucze i certyfikaty. Niestety nadal nazwa konta i hasło pozostają niezabezpieczone;

- SSL/TLS jest najlepszym z tych wszystkich rozwiązań, gdyż zarówno zabezpiecza nazwę konta, hasło i wiadomość. Klucze ustalane są dynamicznie na poziomie sesji konkretnego użytkownika. Poza tym nie jest wymagane wsparcie ze strony jądra systemu. SSL jest wspierany przez wiele klientów poczty, ale niestety wybór serwerów ze wsparciem dla SSL.

Można oczywiście przerobić serwery usług tak, żeby się zamiast odwoływać do standardowego API, odwoływały się do biblioteki OpenSSL. Jednak jest ona dość źle udokumentowana, a poza tym przerabianie cudzego kodu jest dość trudnym zajęciem. Poza tym nie zawsze dysponujemy kodem źródłowym.

Wtedy może się przydać program Stunnel, który jest serwerem proxy pomiędzy klientem, znającym protokół SSL, a serwerem, który go nie zna. MUA łączy się z proxy, który zdejmuje szyfrowanie i łączy się z właściwym serwerem.

Usługi szyfrowania mają najczęściej własne porty, inne porty od tych, które nie wykorzystują szyfrowania. Program Stunnel ma wsparcie dla protokołów, które negocjują użycie SSL w zakresie, w jakim nie narusza to zasady KISS.

Zwykle Stunnel pracuje jako serwer SSL. Oznacza to, że będzie on negocjował szyfrowanie na przyjmowanych połączeniach, po czym łączy się dalej otwartym tekstem. Do połączenia ze zdalnym serwerem służy tryb kliencki. Stunnel oczekuje na połączenia nieszyfrowane, po czym łączy się ze zdalnym serwerem przy pomocy SSL. Przydaje się to wtedy, gdy używany przez nas klient nie potrafi korzystać z SSL (np. chcemy pobrać przez wget strony z serwera https).

Stunnel ma dwie możliwości przyjmowania połączeń. W trybie inetd może się komunikować przy pomocy standardowego wejścia/wyjścia. Program można wywołać jako samodzielny demon co jest bardziej wydajne.

Istnieją również dwa sposoby nawiązywania połączeń. Najprościej można się połączyć ze zdalną usługą przez sieć. Drugim sposobem jest wywołanie programu napisanego dla inetd.

Przykłady

Udostępnianie serwera IMAP klientowi wyposażonemu w SSL

```
stunnel -d imaps -r imap
```

dla trybu lokalnego

```
stunnel -d imaps -l /usr/bin/imapd imapd
```

Przy pomocy „-“ można poinformować Stunnel, że kończą się jego parametry, a zaczynają parametry uruchamianego programu.

```
stunnel -d 902 -l /usr/sbin/swat -- swat -s /etc/smb.conf
```

Przyjmijmy, że mamy klienta pop3 lokalnej stacji, który nie potrafi szyfrować, na serwerze skrzynka.pl działająca usługa pop3

```
stunnel -c -d localhost:pop3 -r skrzynka.pl:pop3s
```

Dobrym przykładem tunelowania usługi, w której ani serwer, ani klient nie znają SSL, jest rsync. Najpierw skonfigurujemy serwer rsync-SSL na porcie 2222 maszyny 192.168.1.20:

```
stunnel -d 2222 -l /usr/bin/rsync -- rsync --daemon
```

potem klienta na stacji, z której będziemy się łączyć:

```
stunnel -c -d localhost:837 -r 192.168.1.20:2222
```

Teraz pozostaje już tylko połączyć się przez przygotowany kanał:

```
rsync rsync://localhost:837/
```

A teraz przykładowy VPN przy użyciu Stunnela oraz pppd. Po stronie serwera uruchamiamy:

```
stunnel -d 5555 -L /usr/sbin/pppd -- pppd local
```

po stronie klienta analogicznie:

```
stunnel -c -r remote:5555 -L /usr/sbin/pppd -- pppd local
```

Opcja `-L` różni się od `-l` tym, że przydziela gniazdu pseudoterminale, którego wymaga pppd. Postępując zgodnie z powyższymi przykładami można, co prawda, zaszyfrować połączenie, ale nie można go ochronić przed aktywnymi atakami.

Aby uchronić się przed atakiem man-in-the-middle protokół SSL podczas negocjacji sesji wymaga, aby serwer przedstawił swój certyfikat. Jeżeli istnieje potrzeba uwierzytelnienia klienta to konieczne jest jawne wskazanie lokalizacji certyfikatu. Do tego służy opcja `-p`. W trybie serwera jest używany plik `stunnel.pem`. Do certyfikatu powinien mieć dostęp tylko właściciel.

Do włączenia weryfikacji certyfikatu, którym przedstawia się druga strona, służy opcja `-v poziom`. Poziom może być:

- 0 – Stunnel prosi o certyfikat i go zapisuje, a potem go ignoruje;
- 1 – certyfikat jest sprawdzany, jeżeli druga strona go przedstawi
- 2 – weryfikacja przebiega poprawnie, jeżeli w składnicy certyfikatów znajduje się właściwy certyfikat;
- 3 – wymagane jest dodatkowo, aby w składnicy znajdowały się wszystkie certyfikaty, które mają być pozytywnie zweryfikowane.

Program Stunnel pozwala nam zachować poufność naszych danych.

PGP

PGP (wymawia się: *pi-dzi-pi*) jest skrótem od nazwy Pretty Good Privacy, co w wolnym tłumaczeniu może oznaczać *Całkiem Niezła Prywatność*.

PGP to program służący do szyfrowania różnego rodzaju danych i informacji przechowywanych w formie elektronicznej. Wykorzystuje on metodę kodowania/rozkodowania, która jak do tej pory nie została przez nikogo na świecie „złamana”, tzn. nie został znaleziony sposób na rozszyfrowanie informacji bez posiadania znajomości hasła szyfrującego. Program jest rozpowszechniany na całym świecie bezpłatnie bez żadnych ograniczeń (dokładnie: to

rozpowszechniana jest jego tzw. międzynarodowa wersja - przeczytaj poniżej dlaczego). Każdy może wykorzystywać PGP do ochrony własnych danych, informacji, korespondencji i być przy tym pewnym, że nawet największe superkomputery, którymi dysponują tylko rządy albo agencje wywiadowcze, nie będą w stanie rozszyfrować tych danych.

Typowe szyfrowanie informacji polega na zastąpieniu normalnego, czytelnego tekstu (napisanego w dowolnym języku), odpowiadającym mu szeregiem znaków, który nie jest w ogóle czytelny ponieważ znaki alfabetu "pozamieniano" według pewnej umownej metody; w najprostszym przypadku taka metoda opisuje, że zamiast litery a trzeba napisać x, zamiast b trzeba użyć e, i tak dalej; w rzeczywistości stosuje się dużo bardziej skomplikowane metody zamiany, które dla uproszczenia używania przez ludzi opisywane są jednym zestawem umownego hasła-kłucza: wpisanie przez użytkownika do programu szyfrującego znanego mu hasła informuje komputer jaką metodą ma zamieniać znaki; przykładowo, jeśli napiszemy w wiadomości tekst *"spotkamy się jutro przed wejściem do Klubu Andromeda..."* to po zaszyfrowaniu będzie on wyglądał tak: *"poou23*(#asddl#asdffds}\32fnsuytrnfl;<@dsewqiy"*

Używanie jako klucza całego zestawu formulek typu "zastąp a przez w..." byłoby oczywiście bez sensu, ponieważ zajmowałoby mnóstwo czasu, tym właśnie zajmuje się komputer wykonując te wszystkie formułki, natomiast cały ich zestaw zapisany jest w jego pamięci i wywoływany wpisaniem przez człowieka odpowiedniego hasła; to samo hasło musi wpisać osoba szyfrująca i ktoś kto chce potem odszyfrować wiadomość; dawniej jako hasła można było stosować tylko pojedynczy zestaw znaków o określonej ilości tych znaków, obecnie jako hasło można używać dowolne zdanie o nieograniczonej długości, zawierające dowolne znaki, symbole, duże i małe litery, itd.; z jednej strony zwiększa to bezpieczeństwo, bo nie można już go odgadnąć metodą sprawdzenia np. wszystkich wyrazów ze słownika, z drugiej strony ułatwia zapamiętanie hasła, bo można używać jakiegos łatwego do zapamiętania dla jego właściciela powiedzenia, fragmentu piosenki, itp.; jak jednak można zauważyć obie strony używające szyfrowanej wiadomości muszą znać hasło, trzeba więc je jakos przekazać tak żeby nikt inny go nie zdobył - na przykład osobiście; nie zawsze jest to możliwe, ale jeśli będziemy mieli taki sposób to po co szyfrować wiadomość jak można ją przekazać tą samą drogą?

Jest jednak sposób aby bezpiecznie przekazać klucz szyfrujący bez potrzeby osobistego kontaktu z innym jego użytkownikiem; wykorzystuje się do tego celu technikę tzw. pary kluczy; w tym przypadku istnieją dwa pasujące do siebie klucze: publiczny i prywatny; klucz publiczny służy do szyfrowania informacji, które można potem rozszyfrować ale tylko pasującym do niego kluczem prywatnym; znajomość klucza publicznego nic nie da w celu rozszyfrowania ponieważ można nim tylko zaszyfrować informacje; dlatego klucze publiczne można swobodnie przekazywać dowolną metodą innym użytkownikom, którzy będą chcieli zaszyfrować informacje przeznaczone dla mnie, natomiast odpowiednio zabezpieczając własny klucz prywatny mogę być pewny, że tylko ja będę w stanie je rozszyfrować;

Klucz prywatny i publiczny jest zestawem znaków całkowicie niezrozumiałych dla człowieka.

Jak wyżej jest wyjaśnione, **klucz publiczny** można rozpowszechnić i przekazywać dowolną metodą, gdyż jego posiadanie umożliwia wyłącznie zaszyfrowanie wiadomości przeznaczonych dla jego właściciela (w żaden sposób posiadanie klucza publicznego nie pozwala rozszyfrować informacji albo nawet próbować stworzyć pasujący do niego klucz deszyfrujący). Natomiast **klucz prywatny** należy przechowywać wyłącznie do własnego użytku, najlepiej tylko na własnym

komputerze i to takim z którego nie korzystają inne osoby. Jednak klucz prywatny jest jeszcze dodatkowo zabezpieczony tzw. osobistym hasłem, każda próba użycia klucza prywatnego wywołuje zadanie podania hasła osobistego, bez którego klucz nie zadziała - i to jest ta jedyna rzecz, której nigdzie nie wolno zapisywać, osobiste hasło powinno być wyłącznie "w Twojej głowie" gdyż to jest ostateczna bariera chroniąca Twoją prywatność.

Program PGP wykorzystuje obie powyższe metody, tzn. właściwe informacje są szyfrowane i przesyłane/przechowywane z wykorzystaniem klucza-hasła, który jest losowo generowany przez program w każdym przypadku szyfrowania; natomiast informacje o tym jaki to jest klucz-hasło są szyfrowane i przekazywane metodą pary kluczy: prywatny/publiczny

Oprócz wykorzystania PGP do przesłania informacji w sposób bezpieczny i gwarantujący, że przeczyta je tylko wskazany odbiorca, PGP można wykorzystać jeszcze inaczej:

- ?? zabezpieczenie danych w komputerze: zaszyfrowanie plików na dysku albo całych folderów z informacjami, bez względu na to jaki format ma plik, np. teksty napisane w MSWord, dane finansowe w MSExcel, bazy danych, itp.
- ?? potwierdzenie autentyczności informacji: czasami bardziej potrzebne jest zapewnienie pewności odbiorcy wiadomości, że to co czyta rzeczywiście zostało przez nas napisane niż zaszyfrowanie samej treści informacji, dzięki PGP mamy możliwość dopisania na końcu wiadomości specjalnej formułki "elektronicznego podpisu" (wygląda podobnie jak klucz) którą odbiorca informacji, używając naszego klucza publicznego może sprawdzić czy treść wiadomości została napisana właśnie przez nas i czy nic w tej treści nie zostało zmienione; dokładniejszy opis jest na stronie "Jak używać PGP"

W czasie tworzenia własnej pary kluczy możemy wybierać, jakiej "wielkości" klucze chcemy używać, tzn. z jakiej ilości znaków mają one składać się. Im dłuższe klucze tym bezpieczniejsze, bo tym większa ilość możliwych kombinacji tworzą. Na przykład, większość zwykłych programów do przeglądania informacji w Internecie używa 40bitowych kluczy w czasie przesyłania niektórych informacji, np. wypełnionych formularzy na stronach WWW. PGP w wersji międzynarodowej umożliwia zrobienie kluczy o długości od 768 do 4096 bitów. Oczywiście ta ilość kombinacji zawsze można odnosić do urządzeń, które ktoś może zastosować do rozszyfrowania hasła (tzn. rozszyfrowania metoda sprawdzenia wszystkich możliwych kombinacji). Skala możliwości obecnie istniejących urządzeń jest bardzo duża (przeważnie przekłada się na koszt takiego urządzenia) i to co jest "szczytem" możliwości nawet najnowocześniejszego PCeta, który będzie potrzebował miesięcy albo i lat na przetestowanie wszystkich kombinacji, może zająć superkomputerowi wartemu miliony dolarów tylko kilka godzin. Zawsze pozostaje pytanie i odpowiedź: jak cenne są nasze informacje dla kogoś innego, czyli ile może on wydać na ich rozszyfrowanie. Poniżej pokazana jest tabela, w której porównano czasy potrzebne na rozszyfrowanie różnych kluczy różnymi urządzeniami.

Tab. 1 Przykładowe czasy łamania PGP

| rodzaj i koszt komputera zastosowanego do "złamania" klucza | czas potrzebny do rozszyfrowania klucza o długości: | | |
|--|--|---------------------|----------------------|
| | 100 bitów | 500 bitów | 1.000 bitów |
| przeciętny komputer PC z procesorem Pentium P90 i 16MB pamięci operacyjnej; wartość około 1.000\$ | 60dni | 10 ⁵ lat | ??? |
| najnowocześniejszy komputer wieloprocessorowy dostępny w handlu; wartość około 500.000\$ | 3dni | 6.000lat | 10 ¹¹ lat |
| specjalny superkomputer wykonywany na indywidualne zamówienie, np. agencji; wartość około 5mln.\$ | 10godzin | 500lat | 10 ⁸ lat |
| superkomputer z rozbudowanym systemem procesorowym (jak na przykład <i>BigBlue</i>); wartość około 500mln.\$ | 60min. | 6lat | 6.000.000lat |

Należy pamiętać, że w czasie rozszyfrowywania klucza liczy się nie tylko cena samego urządzenia, ale także czas jego pracy. Zastosowanie nawet najmniejszego klucza dostępnego w PGP daje wystarczającą pewność, że w przeważającej większości przypadków raczej nikt nie będzie chciał wydać miliona dolarów na "podejrzenie" naszych informacji. Z powyższych przykładów widać, że użyta w tłumaczeniu tytułu polska nazwa programu ***Całkiem Niezła Prywatność*** jest trochę ironiczna. Bowiem PGP to wcale nie taka "całkiem niezła", ale bardzo pewna, potężna, praktycznie "nie do złamania" ochrona informacji dostępna dla każdego. Na całym świecie, praktycznie za darmo (no... może z wyjątkiem tych państw, które "próbują" prawnie zakazać stosowania szyfrowania do prywatnego użytku).

SSL

SSL to skrót od ang. Secure Socket Layers. Opracował go Netscape'a. Został powszechnie zaakceptowany w Internecie do autoryzacji i szyfrowania połączeń pomiędzy klientem a serwerem. Nowy standard Transport Layer Security opracowany przez Internet Engineering Task Force jest oparty na SSL.

SSL działa ponad protokołem TCP/IP i poniżej protokołów aplikacyjnych takich jak http, imap itp. Używając TCP/IP zachowuje się tak jakby był protokołem wyższego poziomu. W ten sposób umożliwia serwerowi obsługującemu SSL autoryzację siebie samego klientowi. Działa to także w odwrotną stronę. Pozwala to obu maszynom na negocjacje szyfrowanego połączenia.

Wymienione tu mozliwosci sa podstawowymi dotyczacymi komunikacji przez Internet i w sieciach TCP/IP:

- ?? SSL server authentication pozwala uzytkownikowi potwierdzic tozsamosc serwera. Oprogramowanie klienta moze uzyc standardowych technik szyfrowania opartych na kluczu publicznym, zeby sprawdzic czy certyfikat serwera i publiczny klucz sa poprawne i potwierdzone certyfikatem autentycznosci, który znajduje w zbiorze zaufanych certyfikatów klienta. Te potwierdzenie moze byc bardzo wzne, jezeli uzytkownik na przyklad wysyla numer karty kredytowej przez siec i chce sprawdzic autentycznosc serwera.
- ?? SSL client authentication pozwala serwerowi sprawdzic tozsamosc klienta. Uzywajac tych samych technik co dla potwierdzenia autentycznosci serwera, serwer moze sprawdzic czy certyfikat i publiczny klucz klienta sa poprawne i czy certyfikat autentycznosci znajduje sie na liscie zaufanych certyfikatów serwera. Potwierdzenie moze byc przydatne na przyklad w przypadku gdy bank przesyła poufne dane finansowe do klienta i chce potwierdzic tozsamosc odbiorcy.
- ?? Szyfrowane polaczenie SSL wymaga, aby wszystkie informacje przesyłane pomiedzy klientem a serwerem byly szyfrowane przez wysylajace oprogramowanie i deszyfrowane przez odbierajace. W ten sposob zapewnia sie wysoki stopien poufnosci. Poufnosc jest wzna dla obu stron przy kazdej prywatnej transakcji. W dodatku wszystkie dane przesyłane przez zaszyfrowane polaczenie SSL sa chronione przez mechanizm wykrywajacy tampering – polega to na automatycznym sprawdzaniu czy jakies dane nie zostaly dolaczone podczas transmisji.

Protokól SSL zawiera dwa podprotokoly: the SSL record protocol i the SSL handshake protocol. The SSL record protocol definiuje format transmisji danych. The SSL handshake protocol wymusza uzycie the SSL record protocol do wymiany grupy wiadomosci miedzy serwerem a klientem, kiedy polaczenie SSL miedzy nimi jest negocjowane pierwszy raz. Ta wymiana wiadomosci zostala zaprojektowana, aby zapewnic zgodnosc z nastepujacymi krokami:

- Autoryzacja serwera przez klienta
- Wybranie przez klienta i serwer algorytmu szyfrujacego lub szyfru, który oboje znaja
- Opcjonalna autoryzacja klienta przez serwer
- Uzycie techniki szyfrowania za pomoca publicznego klucza, zeby zaszyfrowac poufne dane
- Ustanowienie szyfrowanego polaczenia

SSL uzywa wiele różnych algorytmów szyfrujacych i szyfrów podczas autoryzacji serwera przez klienta i odwrotnie, podczas transmisji certyfikatów, wymiany kluczy potrzebnych do nawiązania szyfrowanego polaczenia. Klient i serwer moga wspierac różne algorytmy szyfrujace, w zaleznosci od tego, jaka wersja SSL dysponuja, czy prawa firmy pozwalaja na silne szyfrowanie, czy tez rzad pozwolil na eksport algorytmu szyfrujacego. Protokól nawiazywania polaczenia okresla, jaki algorytm szyfrujacy klient i serwer uzyja do autoryzacji, do przesyłania certyfikatów i uzyskania kluczy sesji. Obecnie wspierane przez SSL algorytmy szyfrujace to: 3DES,DSE, KEA,MD5, RC2,RC4,RSA,RSA key exchange,SHA-1,SKIPJACK,Trippl-DES.

Tab. 2 Algorytmy szyfrujące

| Sila algorytmu i zastosowanie | Algorytmy szyfrujące |
|--|--|
| <p>Najsilniejsza. Można go używać tylko na terenie USA. Stosowany jest w bankach i innych instytucjach, które przechowują bardzo ważne dane</p> | <p>Triple DES, z silą szyfrowania 168-bit z autoryzacją za pomocą SHA-1. Triple DES jest najsilniejszym algorytmem wspieranym przez SSL, ale nie jest tak szybki, jak RC4. Triple DES używa kluczy trzy razy dłuższych niż zwykły DES. Ponieważ klucz jest tak długi, dlatego możliwe jest dużo więcej kluczy - maksymalnie $3.7 * 10^{50}$.</p> |
| <p>Silne szyfrowanie. Dozwolona tylko na terenie USA. Sila tego szyfrowania jest wystarczająca dla większości zastosowań biznesowych i rządowych.</p> | <p>RC4 z silą szyfrowania 128-bit i autoryzacją za pomocą MD5. Ponieważ sila szyfrowania RC4 i RC2 jest 128-bit sa na drugim miejscu po Triple DES (Standard szyfrowania danych), z silą szyfrowania 168-bit RC4 i RC2 128-bit pozwala na wygenerowanie $3.4 * 10^{38}$ możliwych kluczy, dzięki czemu sa ciężkie do złamania. Algorytm RC4 jest szybki.</p> |
| | <p>RC2 z silą szyfrowania 128-bit i autoryzacją za pomocą MD5. RC2 jest podobny do RC4 tyle, że jest wolniejszy.</p> |
| | <p>DES, z silą szyfrowania 56-bit z autoryzacją za pomocą SHA-1. DES jest silniejszym algorytmem niż algorytm z silą szyfrowania 40-bit, ale nie jest tak silny jak 128-bit. DES z silą szyfrowania 56-bit pozwala na wygenerowanie $7.2 * 10^{16}$ możliwych kluczy.</p> |
| <p>Algorytmy eksportowane. Nie sa one tak silne jak powyższe, ale mogą być eksportowane do innych państw. Sa używane w eksportowanych produktach</p> | <p>RC4 z silą szyfrowania 40-bit i autoryzacją za pomocą MD5. RC4 40-bit pozwala na wygenerowanie $1.1 * 10^{12}$ (trylion) możliwych kluczy. Algorytm RC4 jest szybki.</p> |
| | <p>RC2 z silą szyfrowania 40-bit i autoryzacją za pomocą MD5. RC2 40-bit pozwala na wygenerowanie $1.1 * 10^{12}$ (trylion) możliwych kluczy. RC2 jest wolniejszy od RC4.</p> |
| <p>Najsłabsze szyfrowanie. Jest używane do autoryzacji i detekcji przekłaman podczas transmisji. Nie dostarcza szyfrowania.</p> | <p>Brak szyfrowania, autoryzacja za pomocą MD5. Jest używany do detekcji przekłaman podczas transmisji. Jest używany jeżeli klient i serwer nie mają żadnych algorytmów wspólnych</p> |

Protokół używany do nawiązywania połączeń SSL używa kombinacji klucza publicznego i szyfrowanego klucza symetrycznego. Drugi sposób jest szybszy od kluczy publicznych, ale klucze publiczne pozwalają na lepszą autoryzację. Sesja SSL zawsze się zaczyna od wymiany informacji nazywanych SSL handshake. Pozwalają one na autoryzację serwera przez klienta używając kluczy publicznych, potem pozwalają na tworzenie przez serwer i klienta kluczy

symetrycznych używanych do szyfrowania, rozszyfrowywania, detekcji przekłaman podczas transmisji Opcjonalnie pozwala także na autoryzację klienta przez serwer.

Podczas nawiązywania połączenia serwer i klient wymieniają następujące informacje:

- 1) Klient wysyła numer wersji SSL jaka dysponuje i inne informacje potrzebne serwerowi.
- 2) Serwer przesyła klientowi informacje o SSL jakie dysponuje i swój certyfikat.
- 3) Autoryzacja serwera przez klienta
- 4) Klient przesyła wstępne dane zaszyfrowane za pomocą kluczy publicznych serwera.
- 5) Opcjonalna autoryzacja serwera klienta przez serwer.
- 6) Potwierdzenie przez serwer autoryzacji klienta.
- 7) Generacja kluczy i sprawdzenie ich.
- 8) Klient informuje serwer, że kolejne informacje będą przesyłane w postaci zaszyfrowanej.
- 9) Serwer informuje klienta, że kolejne dane będą przesyłane w postaci zaszyfrowanej.
- 10) Połączenie jest nawiązane.

W czasie nawiązywania połączenia serwer jest autoryzowany przez klienta. Klient wtedy musi wykonać następujące kroki:

- 1) Sprawdzenie ważności certyfikatu serwera.
- 2) Sprawdzenie czy certyfikat należy do zbioru certyfikatów zaufanych.
- 3) Porównanie za pomocą publicznego klucza z certyfikatu znajdującego się u klienta w zbiorze certyfikatów czy podpis elektroniczny przedstawionego certyfikatu się zgadza.
- 4) Sprawdzenie czy nazwa domeny z certyfikatu zgadza się z nazwą domeny, z której otrzymano certyfikat.
- 5) Zakonczenie autoryzacji.

Niektóre serwery mogą wymagać autoryzacji klienta. Wtedy serwer przeprowadza następujące kroki:

- 1) Sprawdzenie podpisu elektronicznego klienta za pomocą kluczy publicznych.
- 2) Sprawdzenie ważności certyfikatu.
- 3) Sprawdzenie czy certyfikat klienta znajduje się w zbiorze certyfikatów serwera.
- 4) Serwer używając kluczy publicznych z certyfikatu, który jest zapisany na serwerze, sprawdza podpis elektroniczny z certyfikatu przedstawionego przez klienta.
- 5) Opcjonalnie serwer może sprawdzić czy certyfikat użytkownika znajduje się w katalogu LDAP.
- 6) Sprawdzenie czy zautoryzowany klient ma prawo dostępu do zasobu.

Podsumowując SSL pozwala na bardzo dobrą ochronę danych przesyłanych przez sieć. System certyfikatów zaś zapobiega atakowi man-in-the-middle, gdyż nie można przesłać danych z domeny innej niż domena, z którą chcemy uzyskać połączenie. Niestety sporo także zależy od siły szyfrowania. Dlatego należy używać jeżeli to tylko możliwe jak najsilniejszych algorytmów.

SPYWARE

Spyware znany też jako „adware”, jest to ukryte oprogramowanie, które przesyła dane użytkownika przez Internet do reklamodawców w zamian za darmowe ściąganie oprogramowania.

Co jeśli będzie można ściągnąć za darmo oprogramowanie, które robi prawie to samo, co drogi i markowy produkt? W zamian za to, że można mieć oprogramowanie za darmo trzeba podać swoje

imie, nazwisko, adres, telefon i email oraz inne ogólne informacje. Można pomyśleć sobie, że to dobrze, ale co w przypadku, gdy dane personalne zostały zachowane gdzieś jeszcze na dysku, w tym przypadku są one przesyłane, co jakiś czas do reklamodawców z powrotem przez Internet w celu zwiększania ilości reklam. Wtedy mamy do czynienia z oprogramowaniem spyware.

Powołując się na jedną z firm reklamujących, Radiate „Konsument może kliknąć na bannery [znajdujące się w oprogramowaniu] i otrzymać strony sieci Web, żeby obejrzeć lub zamówić reklamowane produkty. Kiedy użytkownik z powrotem włącza się do sieci, odpowiednie informacje są przysyłane do reklamodawcy i śledzone przez Radiate”. Radiate używa programu Aureate. Jest on instalowany razem z darmową aplikacją. Są też inne firmy reklamowe, które oferują darmowe oprogramowanie bazujące na tej samej idei – rejestrowanie przyzwyczajenia użytkownika w trakcie kiedy szurkuje w Internecie i przesłanie tych informacji do reklamodawcy przez sieć w celu zwiększenia liczby reklam. W wielu przypadkach oprócz aplikacji sponsorowanej istnieje także wersja płatna. Niestety nie jest to wiadome podczas ściągania programu. Dlatego należy zwracać uwagę na warunki licencji i informacje dotyczące rejestracji programu w trakcie instalacji.

Wiele bardzo dobrych i darmowych aplikacji wydaje się mieć standardowy formularz rejestracji – ale niektóre mogą stać się w przyszłości źródłem informacji dla programów spyware. Dlatego też dostawca oprogramowania powinien poinformować użytkownika o tym, że te informacje mogą zostać wykorzystane przez reklamodawców. W praktyce takie informacje ciężko znaleźć.

Trzeba uważać, gdyż dostawcy oprogramowania nie muszą umieszczać tych informacji. Dlatego warto stosować zasadę: Każda aplikacja nie wymagająca dostępu do sieci zawierająca bannery wyświetlane w interfejsie użytkownika może używać programu spyware. W przypadku połączenia się z Internetem informacje o tym, na które bannery kliknięto się, i czasie spędzonym na stronie reklamodawcy zostaną przesłane do poszczególnych reklamodawców. Nowe bannery – zwykle bazujące na kliknięciach lub informacjach z czasu instalacji – zostaną wysłane do programu, żeby zmienić dotychczasowe.

Żeby sprawdzić czy oprogramowanie spyware jest zainstalowane w naszym komputerze należy użyć dodatkowych aplikacji takich jak:

- Programy wyszukujące takiego oprogramowania na dysku i usuwające je
- Zapory ogniowej żeby zablokować nieautoryzowane wysyłanie danych z komputera użytkownika.

Pozostaje, więc pytanie: czy należy używać darmowego oprogramowania? Z jednej strony użytkownik staje się celem wielu reklam. Z drugiej strony ktoś pracował ciężko, żeby napisać dobrą aplikację, którą używasz za darmo, dostaje za to pieniądze z reklam.

COOKIES

Cookies są to informacje, które zostawia strona na naszym dysku, żeby pamiętać niektóre informacje o nas, kiedy następnym razem będziemy odwiedzać tą stronę. Bardziej formalnie są to informacje służące do późniejszego ponownego ich wykorzystania pozostawiane przez serwer na dysku klienta. Są one dołączone do HTMLa i przysyłane do i od użytkownika. W cookies zapamiętywane są preferencje użytkownika odnośnie danej strony. Używając protokołu http, każde zadanie strony jest niezależne od innych zadań. Z tego powodu serwer nie pamięta, które strony wysłał ostatnim razem do klienta lub o ostatniej wizycie użytkownika. Mechanizm cookies

pozwala serwerowi zapisywać informacje o użytkowniku na jego własnym komputerze. Cookies robią użytek ze specyficznych dla danego użytkownika informacji przesyłanych przez serwer do użytkownika, więc mogą być później wykorzystane przez ten sam serwer lub inne. W wielu przypadkach nie tylko informacje o użytkowniku są zapisywane niezauważone, dlatego też serwer może mieć dostęp do nich. Serwery automatycznie mają dostęp do odpowiednich plików cookies, kiedy tylko użytkownik ustanowi połączenie, zwykle przez zadanie strony. Można obejrzeć zawartość plików cookies. Jednak w niektórych przypadkach użytkownik nie znajdzie żadnego sensu w tym zapisie. Położenie plików cookies zależy od przeglądarki.

Mechanizm cookies bazuje na dwufazowym procesie. Po pierwsze plik cookie jest zapisywany na komputerze klienta bez jego wiedzy. Na przykład w dającej się skonfigurować wyszukiwarce takiej jak My Yahoo! użytkownik wybiera kategorię, której się interesuje ze strony. Po tym serwer tworzy plik cookies zawierający tekst opisujący preferencje użytkownika i przesyła go do komputera użytkownika. Przeglądarka, jeżeli plik cookies jest bezpieczny odbiera go i zapisuje w specjalnym pliku. To się dzieje bez jakiegokolwiek powiadomienia użytkownika. W rezultacie dane użytkownika (w tym przypadku wybrane przez użytkownika kategorie) są formatowane przez serwer, przesyłane i zapisane przez komputer użytkownika.

Podczas drugiej fazy pliki cookies są niezauważenie i automatycznie przesyłane z komputera użytkownika do serwera. Kiedy użytkownik otwiera odpowiednią stronę w przeglądarce, przeglądarka bez wiedzy użytkownika przesyła plik cookies zawierające dane użytkownika związane z daną stroną do serwera.

Niestety, mimo, że na początku mechanizm cookies został zaprojektowany, żeby ułatwić surfowanie internetom ostatnio staje się coraz bardziej wykorzystywany przez firmy reklamowe. Bedac na stronie, na której jest włączony mechanizm doubleclick strona ta zada pliku cookies i sprawdza, kim jest użytkownik oraz inne informacje. Wtedy wysyła informacje do „doubleclick” z ID użytkownika zadając wszystkich możliwych informacji reklamowych. Nie do końca wiadomo skąd te informacje pochodzą, ale prawdopodobnie część z nich pochodzi ze śledzenia stron „doubleclick”, na których przebywał użytkownik. Użytkownik otrzymuje tylko przeznaczone dla niego bannery reklamowe. To znaczy, że jeżeli dwie osoby zalogują się na tej samej stronie w tym samym czasie to jedna będzie oglądać reklamy na przykład komputerów a druga ubrań.

Wiele firm używa mechanizmu cookies, żeby śledzić ruchy użytkownika na ich stronie. Plik cookie zawiera unikalny numer, który pozwala na otrzymanie informacji z bazy danych firmy. Firmy sprzedające powierzchnie reklamowe mając identyfikator użytkownika mogą wymieniać dane z firmami reklamowymi. Firmy mogą sprzedawać informacje o tym, co robił użytkownik bedac w Internecie. Cookie pozwalają stronom przez zapamiętanie pewnych informacji na sprawdzenie, kim jest użytkownik logujący się na stronę. Cookies mogą zapamiętywać to, co użytkownik szukał w wyszukiwarce. W oparciu o to mogą być wyświetlane reklamy związane z zapytaniem. Cookies mogą być także użyte do przechowywania listy zakupów, a także do przechowywania danych osobistych podanych w zamówieniu. Dzięki temu użytkownik nie musi ich ponownie podawać. Jednak wiąże się to z naruszeniem bezpieczeństwa, gdyż te dane są przechowywane na dysku. Za pomocą Cookies można także sprawdzać popularność reklam.

Niepokój budzi jednak to, że to wszystko dzieje się bez niczyjej wiedzy. Niektóre osoby mogą pomyśleć, że zbieranie danych o nich godzi w ich prywatność, ale na średnim poziomie prywatności, używanie tych informacji jest nieszkodliwe dopóki znamy ograniczenia tych sieci,

kto zbiera informacje, jakie informacje i w jakim celu. Z drugiej strony nie ma takiego prawa, które by pozwalało zbierać informacje na temat użytkownika bez jego wiedzy.

Cookies mogą stać się groźne, gdyż mogą przenosić wirusy. Jednak to nie jest do końca jak, ponieważ sam plik cookies to tekst. Niestety w niektórych systemach na przykład Unixie pliki mają atrybut pozwalający na wykonywanie pliku. Jeżeli pliki te będą przechowywane w plikach niewykonywalnych nic nam nie grozi. Nawet, jeżeli plik jest wykonywalny to i tak wirus się nie uaktywni dopóki nie uruchomi się pliku. Niestety czasami w przeglądarkach zdarzają się błędy pozwalające uruchomić plik cookies, w którym jest wirus. Maksymalna wielkość pliku cookies to 4kB. Linia usuwająca zawartość dysku twardego ma wielkość 18 bajtów, więc tego typu wirus mógłby wyrządzić szkody nawet gdyby to był niekompletny kon trojański. Jest to na szczęście tylko teoria, bo w rzeczywistości napisanie takiego wirusa wymagałoby bardzo dużo pracy. Teoria można łatwo porównać z innymi dziurawymi oprogramowaniami w sieci. Błąd w ActiveX pozwalał na dostęp do systemu plików. Także były pewne błędy w Javie związane z bezpieczeństwem.

Podsumowując cookies nie mogą uszkodzić komputera. Kontrowersyjne nie jest to, co cookies mogą zrobić w komputerze, ale jakie informacje przechowują i co mogą przesłać do serwera. Jest obecnie nowa propozycja, żeby ograniczyć możliwości protokołu cookies, co da użytkownikom większą kontrolę nad tym jakie pliki cookies mogą akceptować i skąd.

PODSUMOWUJAC

Podsumowując bezpieczeństwo jest bardzo ważne. Dzisiejsze technologie takie jak SSL pozwalają się dostatecznie zabezpieczyć przed atakami. Pozwalają także na ochronę swojej prywatności. Wystarczy odpowiednio skonfigurować przeglądarkę. Należy zablokować elementy potencjalnie niebezpieczne. Można także użyć programów takich jak zapory ogniowe. Z drugiej strony w programach bywają błędy, które pozwalają na obejście zabezpieczeń, dlatego warto jest śledzić informacje na temat odnalezionych błędów w zabezpieczeniach i starać się je w miarę możliwości usuwać. Zanim producent dostarczy odpowiednią łatkę można wyłączyć zagrożone oprogramowanie, albo spróbować je zabezpieczyć. Bardzo ważne jest to dla różnego rodzaju serwerów szczególnie tych, co przechowują różnego typu informacje. Nie należy także instalować podejrzanego oprogramowania, które może zawierać konia trojańskiego. Należy także używać oprogramowania z zaufanych i znanych witryn. Konkludując należy uważać, co się wpisuje w formularzach i na jakich stronach oraz używać podstawowych mechanizmów zabezpieczeń, chyba, że mamy do czynienia z serwerem, gdzie bezpieczeństwo jest bardzo ważne.

LITERATURA

Linux+ 2/2002

<http://anon.efga.org/Remailers/>

<http://www.sourceforge.net/projects/mixmaster/>

<http://premail.sourceforge.net/>

<http://anubis.sourceforge.net/>

<http://mailcrypt.sourceforge.net/>

<http://stunnel.nirt.net/>

<http://www.stunnel.org/>

<http://www.infonet.com.pl/pgp/>

<http://developer.netscape.com/docs/manuals/security/ssl/contents.htm>

<http://home.netscape.com/eng/ssl3/>

<http://www.webopedia.com/TERM/S/SSL.html>

<http://developer.netscape.com/docs/manuals/security/ssl/contents.htm>

<http://grc.com/optout.htm>

<http://cexx.org/problem.htm>

<http://www.accs-net.com/smallfish/advw.htm>

<http://www.zdnet.com/zdhelp/stories/main/0,5594,2612053,00.html>

http://www.deja.com/%5bST_rn=fs%5d/dnquery.xp?ST=MS&svclass=dnr&defaultOp=&&DBS=1&LNG=english&subjects=&authors=&fromdate=&todate=&showsort=score&maxhits=25&groups=alt.comp.freeware&QRY=spyware&x=8&y=6

<http://www.spychecker.com/spyware.html>

http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci211838,00.html

<http://www.cookiecentral.com/>

http://www.netscape.com/legal_notices/cookies.html

<http://yes.iq.pl/serwis/informacje/cookies.php>

<http://help.netscape.com/kb/consumer/19970226-2.html>

<http://www.junkbusters.com/cookies.html>

<http://www.republika.pl/dszczepanik/javascript/cookie.html>

<http://javascript.reporter.pl/js/cookie.html>

<http://www.modernmarketing.pl/print.php?pg=arta&magnr=199905&artnr=03>

<http://tichy.ch.uj.edu.pl/usefull/jsguide/cookies.htm>

<http://www.epic.org/privacy/internet/cookies/>