STUDIA INFORMATICA 2003
Volume 24 Number 2B (54)

Paweł KASPROWSKI, Stanisław KOZIELSKI, Piotr KUŹNIACKI, Tadeusz PIETRASZEK
Politechnika Śląska, Instytut Informatyki

ZAGROŻENIA APLIKACJI INTERNETOWYCH UDOSTĘPNIAJĄCYCH BAZY DANYCH

Streszczenie. W pracy przedstawiono analizę zagrożeń dla aplikacji internetowych umożliwiających dostęp do baz danych. Szczególną uwagę zwrócono na ataki włamywaczy do systemów internetowych (hakerów). Wyróżniono szereg kategorii ataków, wykorzystujących błędy projektantów, administratorów i użytkowników. Przedstawiono metody przeciwdziałania omówionym zagrożeniom.

Słowa kluczowe: aplikacje internetowe, bazy danych, bezpieczeństwo.

THREATS FOR DATABASE WEB-BASED APPLICATIONS

Summary. This paper presents analysis of different types of threats for Web-based applications which are using databases. On-purpose hackers' attacks are in special consideration. Several attack types, which take advantage of errors made by application developers, administrators and common users, are described. The paper presents also methods used to avoid this kind of attacks.

Keywords: Web-based applications, databases, security.

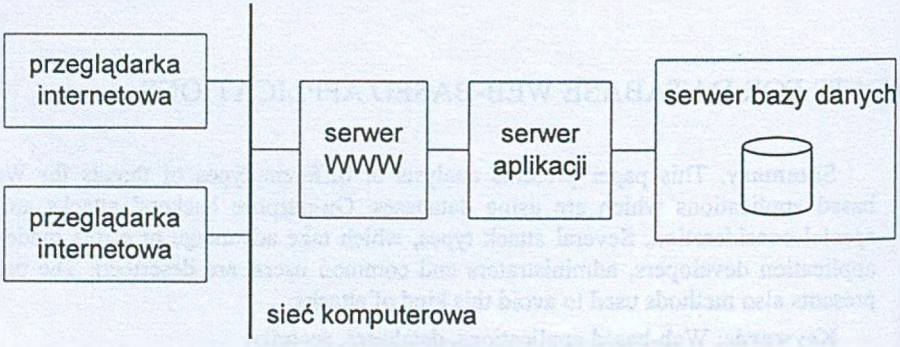
1. Wstęp

Ogromne ilości informacji dostępnych poprzez Internet pochodzą zazwyczaj z baz danych. Aplikacje internetowe oraz wykorzystywane przez nie bazy danych są szczególnie zagrożone różnego rodzaju atakami. Celem pracy jest przedstawienie analizy zagrożeń dla aplikacji internetowych korzystających z baz danych oraz metod przeciwdziałania tym zagrożeniom. W pierwszej części scharakteryzowano architekturę wielowarstwowych systemów udostępniających bazy danych poprzez Internet. W drugiej części wymieniono źródła zagrożeń dla tego rodzaju systemów. Za najgroźniejsze uznano ataki hakerów i im

poświęcono resztę pracy. Kolejną główną część pracy zajmuje szczegółowa analiza ataków na internetowe systemy udostępniające bazy danych. Wyróżniono kilka rodzajów takich ataków, wykorzystujących: błędy w aplikacjach, błędy bądź brak zabezpieczeń protokołów komunikacyjnych, błędy systemu, błędy konfiguracji i błędy użytkowników. Z analizą poszczególnych rodzajów ataków połączono dyskusję metod przeciwdziałania tym atakom. Pracę kończy krótkie podsumowanie.

2. Architektura systemów udostępniających bazy danych poprzez Internet

Klasyczne systemy korzystające z baz danych rozwijały się od systemów scentralizowanych, poprzez systemy klient-serwer, aż do systemów o organizacji wielowarstwowej. Architektura wielowarstwowego systemu informatycznego udostępniającego bazę danych poprzez Internet stanowi najbardziej rozwiniętą formę architektury klient-serwer. Organizację takich systemów przedstawia rys. 1.



Rys. 1. Architektura wielowarstwowego bazodanowego systemu internetowego
Fig. 1. Architecture of multi-layered database web-based system

Zadania realizowane w poszczególnych warstwach są następujące:

Przeglądarka: prezentuje strony internetowe (zapisane w języku HTML), umożliwia też przesłanie do serwera informacji wprowadzonych przez użytkownika.

Serwer WWW: odbiera zgłoszenia od przeglądarki i w najprostszym przypadku odsyła do przeglądarki żądaną stronę (zapisaną w języku HTML). Często spotykanym rozwiązaniem jest wykorzystanie tzw. aktywnych stron zawierających szablon w kodzie HTML oraz wstawki w dedykowanym, skryptowym języku programowania. Serwer WWW przetwarza takie skrypty i generuje odpowiednią postać strony internetowej w języku HTML. Skrypty zawierają też zazwyczaj polecenia dotyczące operacji na bazie danych, przekazywane do

serwera aplikacji. (W niektórych rozwiązaniach przetwarzanie skryptów odbywa się w serwerze aplikacji.)

Serwer aplikacji: w nim implementowana jest tzw. logika biznesowa aplikacji i jest on odpowiedzialny za współpracę z bazą danych. Serwer aplikacji kieruje do serwera bazy danych zapytania wyrażone w języku SQL, odbiera dane wynikowe, przetwarza je i przekazuje końcowe rezultaty serwerowi WWW.

Serwer bazy danych: odbiera polecenia (w języku SQL) i wykonuje je (w dużym uproszczeniu) w następujących etapach:

- kontrola uprawnień użytkownika przekazującego polecenia,
- optymalizacja poleceń,
- wykonanie operacji na bazie danych,
- kontrola integralności bazy w trakcie wykonywania operacji,
- wysłanie odpowiedzi do serwera aplikacji.

3. Źródła zagrożeń dla aplikacji internetowych

Zagrożenia dla aplikacji udostępniających bazy danych poprzez Internet związane są z wszystkimi elementami składowymi takich systemów i wszystkimi formami działań zarówno użytkowników, jak i administratorów tych systemów. Typowy podział [2, 4] rozgranicza zagrożenia na wynikające z celowego działania nieuprawnionych użytkowników oraz na te, które nie są skutkiem celowego działania. Celowe działania osób nieuprawnionych mogą obejmować kradzieże informacji, ataki czy akty sabotażu. Do drugiej grupy można zaliczyć: awarie sprzętu, błędy i niedopatrzenia użytkowników, różne zdarzenia losowe.

Kolejno krótko scharakteryzujemy poszczególne rodzaje zagrożeń, wskazując na najważniejsze sposoby przeciwdziałania tym zagrożeniom.

3.1. Awarie sprzętowe

Ciągłe ulepszanie sprzętu komputerowego obniża jego awaryjność, jednak wzrastający stopień jego skomplikowania nadal jest źródłem wielu problemów. Awarie sprzętowe można podzielić na:

- Awarie komputerów – drobne i pojedyncze awarie podzespołów komputerowych, takich jak płyty główne, procesory, pamięci operacyjne czy monitory, to najczęściej spotykane uszkodzenia.

- Awarie nośników – uszkodzenia dotyczące nośników danych, czyli dysków twardych, dysków optycznych, taśmowych są szczególnie niebezpieczne, gdyż mogą wiązać się z trwałą utratą danych.
- Awarie sieci komputerowych – awarie sieci komputerowych mogą obejmować fizyczne zniszczenie okablowania sieciowego, skutkują one brakiem dostępu do systemów.
- Zdarzenia w centrum danych – to zdarzenia obejmujące największe zagrożenia, takie jak pożar, trzęsienia ziemi czy kradzież.

Do głównych sposobów przeciwdziałania tym zagrożeniom oraz ich skutkom zaliczyć można:

- regularne tworzenie kopii bezpieczeństwa bazy danych,
- stosowanie macierzy dysków RAID (zwiększenie bezpieczeństwa przy awariach pojedynczych dysków),
- stosowanie zasilaczy awaryjnych (niwelowanie zakłóceń sieci energetycznej, podtrzymanie prac systemu w przypadku zaniku zasilania),
- instalowanie zapasowych serwerów.

3.2. Błędy w administrowaniu systemem

Administrator systemu odgrywa kluczową rolę w planowaniu i realizacji odpowiedniej polityki bezpieczeństwa. Dlatego w wielu punktach niniejszego opracowania pojawiają się uwagi odnoszące się do wymaganych działań administratora. W niniejszym podrozdziale zwracamy uwagę na problem autoryzacji użytkowników, tzn. nadawania użytkownikom określonych uprawnień. Taka autoryzacja odbywa się zazwyczaj na dwóch poziomach: w serwerze WWW (i/lub serwerze aplikacji) oraz w serwerze bazy danych. Ponieważ dostęp do serwera WWW na ogół może mieć praktycznie każdy użytkownik sieci, więc administrator systemu powinien maksymalnie ograniczać uprawnienia takich użytkowników. Główny ciężar kontroli dostępu będzie wtedy spoczywał na serwerze WWW. Obsługując zgłoszenia użytkowników, serwer ten będzie kontrolował ich uprawnienia, a następnie, występując w roli tzw. zaufanego podsystemu [1], kierował zapytania do bazy danych. Zaletą omawianego rozwiązania jest duża skalowalność systemu (nawet przy ograniczonej zwykle puli połączeń do serwera bazy danych), natomiast pewnymi wadami są: zmniejszenie możliwości szczegółowego monitorowania operacji przez serwer bazy danych oraz, w przypadku ewentualnego przejęcia przez włamywacza zaufanego podsystemu, wzrost ryzyka narażenia na szwank serwera bazy danych.

Błędy w procesie nadawania uprawnień przez administratora mogą mieć katastrofalne skutki dla bezpieczeństwa systemu. Formą śledzenia i kontroli działań użytkowników może

być monitorowanie wykonywanych przez nich operacji zarówno na serwerze WWW, jak i operacji na bazie danych, w tym szczególnie monitorowanie wykorzystania przez użytkowników posiadanych uprawnień [1, 2].

3.3. Wirusy komputerowe

Wirusy komputerowe są to programy komputerowe, które zakłócają prawidłową pracę systemów i aplikacji. Programy te mogą niszczyć dane, blokować lub obciążać system poprzez wykonywanie dodatkowej pracy. Próbując sklasyfikować wirusy komputerowe, można je podzielić na:

- Wirusy – programy, które potrafią się rozmnażać wewnątrz plików wykonywalnych lub sektorach dysków.
- Makrowirusy – programy tworzone za pomocą makropoleceń, rozprzestrzeniające się wraz z dokumentami, np. edytorami tekstu lub arkuszami kalkulacyjnymi.
- Bakterie – to grupa programów samopowielających się, które zużywają zasoby komputera, jak procesor czy pamięć dyskowa.
- Robaki – grupa programów podobnych do bakterii, jednak polem ich działania jest sieć komputerowa, gdzie zmniejszają przepustowość, obciążają czy blokują serwery.
- Bomby logiczne – to szczególna grupa wirusów, które aktywują się w określonym momencie zdeterminowanym wystąpieniem konkretnych okoliczności.
- Konie trojańskie – to wirusy, które udają pożyteczne programy. Pozyskując zaufanie użytkowników, zdobywają cenne dane lub wykonują czynności zagrażające systemowi.

Najskuteczniejszą metodą walki z wirusami komputerowymi jest niedopuszczanie ich do systemów, co można osiągnąć przez zdefiniowanie odpowiedniej polityki bezpieczeństwa dla pracowników systemu. Jednak niezbędnym elementem bezpiecznego systemu jest uaktualniane i zaawansowane oprogramowanie antywirusowe, opierające się na poszukiwaniu sygnatur wirusów, analizie heurystycznej i sprawdzaniu integralności plików.

3.4. Ataki hakerów

Ataki hakerów stanowią największe zagrożenie dla systemów internetowych, w tym szczególnie systemów wykorzystujących bazy danych. Dlatego bardziej szczegółowej analizie tych problemów poświęcimy kolejny rozdział.

4. Rodzaje ataków

4.1. Ataki wykorzystujące błędy aplikacji

Aplikacje internetowe komunikują się z użytkownikami za pomocą protokołu HTTP. Protokół ten został początkowo stworzony na potrzeby udostępniania statycznych stron WWW, stąd jego nieduże możliwości i duża podatność na ataki.

Komunikacja za pomocą protokołu HTTP sprowadza się do wysłania przez użytkownika żądania strony i zwrotnego odesłania strony przez serwer. Do adresu strony użytkownik może dodać dowolne parametry w postaci `'nazwa_parametru=wartość_parametru'`.

Oczywiście w przypadku aplikacji internetowych strona WWW odsyłana do użytkownika jest dynamicznie generowana zgodnie z podanymi przez niego parametrami i jego historią pracy z serwerem. Stwarza to pole do różnorodnych ataków.

4.1.1. Celowa zmiana parametrów wywołania

W klasycznej aplikacji internetowej użytkownik rozpoczyna pracę wpisując adres strony startowej. Dalsza jego praca z systemem polega na poruszaniu się pomiędzy kolejnymi oknami poprzez wybór (kliknięcie) pojawiających się w oknie programu obiektów (najczęściej przycisków). W rzeczywistości każde naciśnięcie przycisku powoduje wysłanie do serwera żądania strony i listy parametrów. Parametry te mogą być widoczne dla użytkownika w oknie przeglądarki (metoda GET) lub nie (metoda POST). W związku z tym użytkownik może w prosty sposób wysłać specjalnie spreparowane żądanie, podmieniając na przykład wartość niektórych parametrów lub dodając nowe. Zapytanie takie w przypadku nieprawidłowo zaprojektowanej aplikacji mogłoby dać mu dostęp do danych, do których nie powinien mieć uprawnień.

4.1.2. Odkrycie furtek dla projektantów

Zdarza się, że projektanci aplikacji zostawiają w niej 'tylne drzwi', które pozwalają w prosty sposób przejść do modułów administracyjnych. Takie rozwiązanie ułatwia testowanie i analizę działania aplikacji w fazie projektowania. Oczywiście po przejściu do fazy wdrożenia wszystkie 'tylne drzwi' powinny zostać usunięte. Zostawienie takich furtek jest klasycznym błędem projektowym dla każdego rodzaju aplikacji. Problem polega na tym, że odkrycie tego typu furtek w aplikacji internetowej jest znacznie łatwiejsze niż w zwykłej aplikacji klient-serwer. W takiej aplikacji kod po stronie klienta przechowywany jest w postaci binarnej i wszelka jego analiza wymaga wcześniejszej dekompilacji. W przypadku aplikacji internetowej kod po stronie klienta jest zwykle przesyłany w postaci źródłowej

i interpretowany ad hoc przez przeglądarkę internetową. Z tego powodu 'podpatrzenie' i analiza kodu są wręcz dziecinnie proste.

Częstym błędem aplikacji internetowych jest nieusuwanie niepotrzebnego już kodu, a jedynie 'komentowanie' go w źródłach. O ile w przypadku aplikacji kompilowanych praktyka ta nie jest groźna (zakomentowany kod po prostu się nie kompiluje), o tyle w przypadku przesyłania do klienta źródeł programu, może to być niezwykle niebezpieczne. Na przykład, zostawienie w źródłach przesyłanej strony algorytmu dostępu serwisowego z wszelkimi niezbędnymi hasłami i adresami otwiera hakerowi, który zajrzy do tego kodu za pomocą przeglądarki, pełne uprawnienia.

4.1.3. Zmiana zapytania SQL

W przypadku aplikacji internetowych obsługujących bazy danych bardzo często wartości wpisanych przez użytkownika parametrów są używane do tworzenia zapytań SQL. Poważnym i często spotykanym błędem jest wklejenie parametru, bez sprawdzania jego przesłanej wartości, bezpośrednio w tekst, który jest z kolei wysyłany do bazy jako treść zapytania. Przy odpowiednio przygotowanym ciągu znaków użytkownik może zupełnie zmienić treść zapytania i spowodować na przykład uszkodzenie bazy danych.

4.1.4. Przepełnienie bufora parametrów

Atak przez przepełnienie bufora jest w chwili obecnej jednym z najpopularniejszych ataków dotyczących systemów internetowych. W przypadku parametrów jego mechanizm polega na przesłaniu do aplikacji żądania zawierającego parametr, którego wartość jest 'nienormalnie' długa. Na przykład, wartość parametru zawierająca 10.000 znaków może, przy błędnym zaprojektowaniu aplikacji, spowodować jej błędne działanie. W rzeczywistości jednak ataki tego typu zależą mniej od aplikacji, a więcej od konfiguracji serwera WWW i systemu operacyjnego, dlatego szerzej zostaną omówione w rozdziale 4.2.2.

4.1.5. Ogólne zasady projektowania bezpiecznych aplikacji internetowych

Podstawową zasadą projektowania aplikacji internetowej powinien być kompletny brak zaufania do przesyłanych przez użytkownika parametrów. Wszystkie przesyłane dane powinny być przed ich użyciem dokładnie sprawdzane pod względem poprawności.

Dodatkowym zabezpieczeniem bazy danych może być nadanie użytkownikowi internetowemu jak najniższych uprawnień, dzięki czemu nawet w przypadku złamania zabezpieczeń aplikacji włamywacz nie będzie mógł dokonać zniszczeń w bazie danych.

Innym – rzadziej spotykanym – sposobem zabezpieczeń jest stworzenie systemu delegowania użytkowników do bazy danych. W takim systemie aplikacja łączy się z bazą danych jako konkretny, zalogowany użytkownik. Dzięki temu proces sprawdzania uprawnień

do poszczególnych opcji systemu jest przesunięty na poziom bazy danych i trudniejszy do zmylenia.

4.2. Ataki wykorzystujące błędy konfiguracji

Jak zwykle najsłabszym ogniwem w łańcuchu bezpieczeństwa jest człowiek. Postęp w dziedzinie bezpieczeństwa, nowoczesne zabezpieczenia opierające się na nowych produktach sprawiają, że w wielu przypadkach użytkownicy i administratorzy systemów czują się bezpieczni i niezagrożeni. Dzięki licznym zapewnieniom producentów oprogramowania o bezpieczeństwie ich produktów wielu naiwnych administratorów pozostawia zainstalowane produkty bez żadnej konfiguracji, przyjmując, że domyślne ustawienia producenta muszą być najbardziej restrykcyjne i bezpieczne. Nic bardziej mylnego. Polityka marketingowa wielu firm zakłada, że ich produkty muszą być przyjazne i wygodne w użyciu, stawiając zadowolenie klienta na pierwszym miejscu. Niestety, bardzo często zdarza się, że jest to okupione obniżeniem poziomu bezpieczeństwa. Przykładowo, produkty firmy Microsoft są powszechnie uważane za mało bezpieczne, co w większości przypadków wynika z pozostawienia przez niedoświadczonych administratorów praktycznie w ogóle nie skonfigurowanych systemów. Przyjazny interfejs, ładna grafika zachęcają użytkownika i sprawiają, że wszystko wydaje się łatwe i oczywiste. Niestety, świadomość faktu, że produkt wyjęty z pudełka nie jest bezpieczny i należy go poprawnie skonfigurować, jest bardzo niska wśród wielu użytkowników.

4.2.1. Ataki przez niepotrzebnie uruchomione usługi

Błędnie skonfigurowany system operacyjny, serwer WWW lub innego rodzaju usługi dodatkowe mogą stać się przyczyną ataku na system. Każdy haker przed przystąpieniem do ataku na system przeprowadzi skanowanie (*scanning*), mające na celu zgromadzenie wielu interesujących informacji o jego słabych punktach, które można wykorzystać do ataku. Za pomocą specjalistycznego oprogramowania można zdobyć informacje o otwartych portach, czyli uruchomionych usługach. Dodatkowo na tej podstawie oraz poprzez analizę stosu IP systemu, czyli przesyłanie szczególnych lub błędnych pakietów do systemu, można z dużym prawdopodobieństwem określić typ oraz wersję systemu operacyjnego. Na podstawie tych informacji haker może odnaleźć znaną mu lukę bezpieczeństwa, którą jest w stanie wykorzystać. Proces, zwany enumeracją, jest wyszukiwaniem poprawnych kont użytkowników lub źle zabezpieczonych zasobów współdzielonych. Źle zabezpieczony system udostępnia wiele interesujących informacji, takich jak nazwy kont użytkowników, zasobów współdzielonych, domeny w sieci, nazwy komputerów i ich funkcji w sieci itd. Przykładowe furtki, które mogą zostać wykorzystane w produktach firmy Microsoft, to pusta sesja

NetBIOS, enumeracja SNMP, transfer stref DNS, enumeracja Active Directory. W systemach UNIX niebezpieczne są: NFS, narzędzia typu Finger, RFC. Tylko kwestią czasu pozostaje moment, w którym intruz, mający dostęp do tych danych, zdobędzie lub złamie odpowiednie hasła lub wykorzysta luki protokołów udostępniania zasobów.

Istnieje wiele innych programów i usług sieciowych, które są potencjalnymi furtkami dla atakujących, przykładowo FTP, Telnet czy usługi do zdalnej administracji systemami. Z tego powodu zadaniem administratora jest minimalizacja liczby dostępnych usług. Uruchomione powinny być wyłącznie usługi, które są niezbędne do normalnej pracy systemów, wszystkie dodatki, mogące zostać wykorzystane do włamania, powinny zostać wyłączone. W przypadku konieczności zachowania usług, które nie są bezpieczne, należy je udostępniać wewnątrz sieci zaufanej. Na granicy sieci zewnętrznej i wewnętrznej powinny znajdować się urządzenia lub oprogramowanie typu firewall, blokujące dostęp do tych usług. Pomocne jest także oprogramowanie typu IDS, pozwalające na wykrywanie ewentualnych ataków hakerów.

4.2.2. Ataki wykorzystujące błędy w starych i nie aktualizowanych usługach o powszechnie znanych błędach

Jedną z ważniejszych czynności każdego administratora systemu jest regularne sprawdzanie, czy w oprogramowaniu przez niego używanym nie zostały znalezione jakieś luki, a co za tym idzie, czy system nie jest otwarty na ataki. Oprogramowanie powinno być regularnie aktualizowane, gdy tylko producent udostępni nowszą, bezpieczniejszą wersję produktu. Obowiązkiem każdego administratora jest regularna analiza stanu systemów w celu znajdowania słabych punktów i ich usuwanie, ze szczególnym naciskiem na kluczowe pod względem bezpieczeństwa usługi. Oprogramowanie serwera WWW jest częstym celem ataków hakerów i wiele luk zostało w tym oprogramowaniu znalezionych. Ma to związek z faktem, że ataki są przeprowadzane przy wykorzystaniu portów powszechnie używanych przez sieć Web, poprzez które komunikacja jest w większości przypadków akceptowana przez urządzenia graniczne.

Bardzo popularne jest wykorzystywanie luk w obsłudze CGI (*Common Gateway Interface*), technologii coraz rzadziej używanej. Źle napisane skrypty CGI należą do najgroźniejszych luk w zabezpieczeniach w Internecie. Pomijając podstawowe błędy programistyczne, bardzo niebezpieczne jest przepełnienie bufora (*Buffer Overflow*) danymi o zbyt dużym rozmiarze, co może doprowadzić do nadpisania np. adresu powrotu funkcji, oraz fakt nienależytego sprawdzania poprawności danych wejściowych w skryptach CGI. Podobne problemy występują także w skryptach PHP (*PHP Hypertext Preprocessor*) czy ASP (*Active Server Pages*), nawet w tych, które są domyślnie instalowane z serwerami WWW i mają za zadanie wspomagać zdalne administrowanie serwera lub też być przykładowymi skryptami do nauki. Zdecydowana większość luk serwerów WWW wiąże się

właśnie z niedoskonałościami towarzyszących skryptów, bibliotek czy filtrów, a nie z samym działaniem serwera.

Istnieją specjalistyczne programy i skrypty wspomagające zarządzanie, potrafiące odszukać uchybienia w konfiguracji serwerów WWW, związane ze znanymi metodami ataku.

4.3. Ataki wykorzystujące błędy systemu

Często administratorzy po prawidłowym skonfigurowaniu systemu przestają się martwić o bezpieczeństwo, uważając system za w pełni zabezpieczony. Jednak niejednokrotnie słyszymy o nowo znalezionych furtkach w oprogramowaniu uznawanym dotychczas za bezpieczne i nie do złamania. Niestety, wraz z ciągłym udoskonalaniem zabezpieczeń oprogramowania, także coraz bardziej wyrafinowane stają się narzędzia do ich łamania. Administrator nigdy nie może być pewny, że system, który jest bezpieczny dziś, nie będzie otwarty na ataki w niedalekiej przyszłości. Nierzadko może to nastąpić z powodu błędów samego systemu, za które administrator nie jest w żaden sposób odpowiedzialny.

4.3.1. Ataki na znane słabe punkty systemów za pomocą narzędzi hakerskich

Wiele spośród skutecznych ataków hakerów ma za cel złamanie samego systemu operacyjnego, a nie wykorzystanie luk dodatkowego oprogramowania. Oprócz prymitywnych metod ataku polegających na odgadywaniu haseł, siłowym łamaniu haseł lub wykorzystaniu niezabezpieczonych usług, istnieje wiele zaawansowanych metod ataku. Podstawowym słabym punktem systemów wykorzystywanym przez hakerów jest przepełnianie bufora (*Buffer Overflow*) danymi o zbyt dużym rozmiarze, co może spowodować nadpisanie stosu i uruchomienie dowolnego kodu. Specjalne programy lub wywołania wykorzystujące luki takiego typu w konkretnych usługach systemu mogą przejąć kontrolę nad systemem.

Za pomocą specjalistycznego oprogramowania wywołanego lokalnie, które potrafi wstawiać spreparowany kod do niskopoziomowych procesów, można wykonywać dowolne zadania, np. dodawać użytkowników do grupy administratorów.

Jeżeli haker zdobędzie uprawnienia administratora, oprócz dostępu do danego systemu, może także zdobyć hasła do innych systemów, takich jak kontroler domeny. Bardzo często na skutek błędnej polityki haseł, hasło lokalne administratora jest identyczne lub niewiele zmodyfikowane w stosunku do hasła kontrolera domeny. Haker może zdobyć także inne, lokalne hasła systemu, które mogą być pomocne w znalezieniu hasła kontrolera domeny, przy użyciu oprogramowania pozwalającego na złamanie haseł metodą siłową. Uzyskanie haseł jest tylko kwestią czasu i zależy jedynie od ich stopnia skomplikowania. Inną metodą zdobycia potrzebnych haseł może być użycie oprogramowania służącego do rejestracji wciskanych klawiszy.

Ostatecznie haker może zostawić w opanowanym systemie tylne wejścia, pozwalające na powrót do systemu pomimo zmian haseł i zabezpieczeniu luk, za pomocą których włamał się do systemu. Oprócz wielu prostych programów pozwalających to uzyskać, istnieją zaawansowane programy umożliwiające ukrywanie swojego istnienia w systemie.

Ostatnim krokiem zdeterminowanego hakera, który nie jest w stanie złamać zabezpieczeń systemu, może być atak typu DoS (*Denial Of Service*) polegający na próbie ograniczenia dostępu do usług. Ataki tego typu mogą także służyć do wymuszenia ponownego uruchomienia systemu w celu aktywowania się pewnych narzędzi, ułatwiających pracę hakerowi. To ataki bardzo niebezpieczne i trudno się przed nimi obronić. Ataki te można podzielić na pochłaniające przepustowość sieci oraz pochłaniające zasoby komputera. Istnieje bardzo wiele odmian tego typu ataków wykorzystujących niedociągnięcia protokołów komunikacyjnych.

Dla zapewnienia bezpieczeństwa administrator musi regularnie przeszukiwać system w celu znajdowania potencjalnych luk oraz instalować uaktualnienia systemu dostarczane przez producenta. Bardzo wartościowe jest także analizowanie pojawiających się wpisów na grupach dyskusyjnych dotyczących zarówno bezpieczeństwa, jak i metod włamywania się do systemów. Można się tam doszukać cennych informacji o nowo znalezionych lukach bezpieczeństwa i metodach ich zapobiegającym. Jest to szczególnie istotne, gdy czas od pojawienia się luki do uaktualnień producenta może okazać się znaczny i kluczowy dla bezpieczeństwa systemów.

4.4. Ataki wykorzystujące błędy i brak zabezpieczeń protokołów komunikacyjnych

Częstym sposobem ataku jest wykorzystanie słabości protokołów komunikacyjnych. Ze względu na charakter komunikacji w ogólnodostępnej sieci Internet opartej na protokole IP, nie można zakładać bezpieczeństwa przesyłanych danych. Cała komunikacja może zostać podsłuchana, przechwycona lub sfałszowana.

Sniffing określa atak polegający na podsłuchiwaniu informacji przesyłanych siecią. Pozwala on na dostęp atakującego do cennych informacji, takich jak numery kart kredytowych, hasła dostępu itp. Jest to szczególnie ważne w przypadku sklepów internetowych oraz aplikacji bankowych.

Spoofing to jedna z technik ataków aktywnych, polegającą na podszywaniu się pod inny komputer. Do wielu odmian tej techniki należą podszywanie się pod IP oraz DNS.

Hijacking to metoda ataku polegająca na przechwytywaniu sesji w protokole TCP i ściśle wiąże się z poprzednią.

Wspomniane metody ataku mogą być wspomagane przez specjalistyczne oprogramowanie, pozwalające na automatyczne zbieranie podsłuchiwanych danych czy wyodrębnianie wybranych danych.

Metodami zapobiegającymi tego typu atakom są: szyfrowanie transmisji na poziomie aplikacji (SSL), wykorzystanie kryptograficznej odmiany protokołu IP (IPSec) lub stosowanie analizatorów mogących wykrywać anomalie związane z atakiem. Ze względu na duże zagrożenie wynikające z tych ataków należy przedsięwziąć środki, mające na celu odpowiednie zabezpieczenie komunikacji. W Internecie stosowane są 2 standardowe protokoły zabezpieczeń: SSL i IPSec.

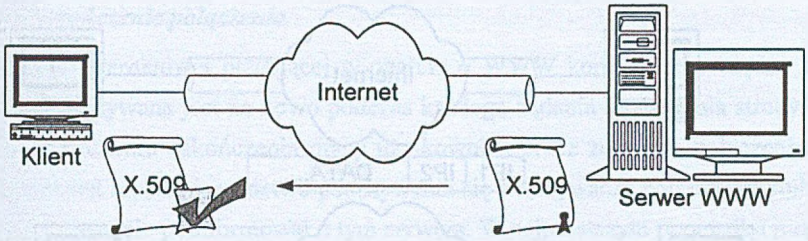
4.4.1. Protokół SSL

Protokół SSL (*Secure Socket Layer*) został opracowany przez firmę Netscape ([6]). Protokół ten działa w warstwie aplikacji TCP/IP i realizuje następujące 4 zadania:

- Identyfikacja serwera – pozwala na potwierdzenie tożsamości serwera. Oprogramowanie klienta, przy wykorzystaniu mechanizmów kryptografii publicznej, może jednoznacznie potwierdzić autentyczność serwera, który posiada certyfikat i klucz publiczny wydany przez zaufanego wystawcę certyfikatów (*CA - Cerficate Authority*).
- Identyfikacja klienta – umożliwia weryfikację certyfikatu klienta przez serwer, czyli identyfikację klienta, na podobnej zasadzie, na jakiej następuje identyfikacja serwera. Metoda ta może zastępować lub uzupełniać hasło użytkownika.
- Ochrona transmisji przed podsłuchem – cała komunikacja klient – serwer jest szyfrowana, co uniemożliwia przechwycenie przekazywanych danych.
- Ochrona transmisji przed modyfikacją – całość przesyłanych danych jest zabezpieczona przed modyfikacją poprzez funkcję skrótu, która umożliwia weryfikację ich poprawności.

Po nawiązaniu połączenia w protokole SSL następuje negocjacja wersji protokołu oraz wybór algorytmów kryptograficznych. Następnie strony są uwierzytelniane, po czym w sposób bezpieczny przesyłane są wygenerowane klucze symetryczne służące do szyfrowania dalszej komunikacji.

Protokół SSL wprowadza dość duży narzut przy nawiązywaniu połączenia (negocjacje algorytmów, uwierzytelnianie i wymiana kluczy), natomiast sama transmisja danych nie jest kłopotliwa.



Rys. 2. Bezpieczna komunikacja za pomocą protokołu SSL
Fig. 2. Safe communication using SSL protocol

Protokół ten najczęściej stosowany jest do bezpiecznego dostępu do stron WWW, rzadziej do szyfrowania komunikacji z bazami danych. Protokół SSL umożliwia zabezpieczenie wyłącznie transmisji realizowanych za pomocą protokołu TCP/IP.

4.4.2. Protokół IPSec

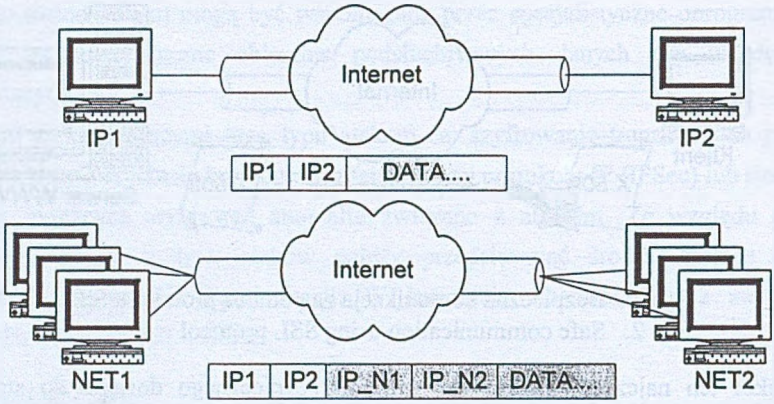
Protokół IPSec jest protokołem opracowanym przez IETF [3]. Działa on w warstwie sieciowej pomiędzy protokołami TCP i IP i umożliwia bezpieczne tunelowanie pakietów w sieci Internet. Zadania realizowane przez protokół są następujące:

- weryfikacja nadawcy,
- ochrona danych przed podsłuchem i modyfikacją,
- zarządzanie kluczami kryptograficznymi.

W skład IPSec wchodzą 3 klasy protokołów: ESP – protokół kapsułkowania, AH – protokół nagłówków uwierzytelniania, ISAKMP – protokół zarządzania kluczami. W protokole IPSec możliwa jest negocjacja funkcji skrótu i algorytmów kryptograficznych stosowanych do szyfrowania transmisji, a samo szyfrowanie jest dokonywane za pomocą szyfrów symetrycznych.

Protokół IPSec może działać w 2 trybach (jak przedstawiono na rysunku 3):

- tryb transportowy - w tym trybie szyfrowane są tylko dane z pakietu IP, gdyż komunikacja odbywa się pomiędzy dwoma komputerami,
- tryb tunelowy – w tym trybie szyfrowane są wszystkie dane oraz nagłówki oryginalnych pakietów IP, co dodatkowo ukrywa nadawcę i odbiorcę.



Rys. 3. Tryby pracy protokołu IPSec
Fig. 3. IPSec modes

Tunelowanie IPSec jest przezroczyste dla działających aplikacji i nie wymaga ich modyfikacji. Protokół ten stosuje się do komunikacji pomiędzy serwerami aplikacji i serwerami bazy danych, jeżeli nie znajdują się one w zaufanej i wydzielonej sieci.

4.5. Ataki wykorzystujące błędy użytkowników

Nawet doskonałe zabezpieczenia serwisu internetowego nie spełnią swoich zadań, jeśli jego słabym ogniwem okażą się jego użytkownicy. Projektując aplikację internetową, należy pogodzić sprzeczne interesy użytkowników i bezpieczeństwa. Użytkownicy chcieliby mieć system jak najbardziej przyjazny i łatwy w obsłudze. Wprowadzanie zabezpieczeń zwykle powoduje utrudnienia dla użytkowników – należy w specjalny sposób skonfigurować przeglądarkę, zapamiętać hasło dostępu, zainstalować dodatkowe oprogramowanie itp. Najczęściej podczas projektowania trzeba więc znaleźć złoty środek pomiędzy wygodą a bezpieczeństwem.

4.5.1. Słabe hasła

Najczęstszym błędem popełnianym przez użytkowników jest użycie zbyt łatwych do odgadnięcia haseł. Bezpieczny system powinien zabraniać użytkownikom tworzenia zbyt krótkich haseł lub wręcz takich samych jak nazwa użytkownika. Wprowadzenie zbyt dużych ograniczeń na tworzone hasła może jednak spowodować, że staną się one trudne do zapamiętania przez użytkowników, a co za tym idzie pojawią się w ich notatnikach, czy też w postaci kartek włożonych pod klawiaturę. Jest to dobry przykład na to, że wprowadzenie zbyt dużych ograniczeń może – paradoksalnie – powodować obniżenie bezpieczeństwa systemu.

4.5.2. Niekończenie połączenia

W aplikacji internetowej pracującej w oparciu o WWW komunikacja między klientem a serwerem nawiązywana jest na nowo podczas każdego żądania i przesłania strony. Nie ma tu wyraźnego momentu zakończenia pracy użytkownika przez zerwanie połączenia. Z tego powodu realne jest niebezpieczeństwo podszywania się włamywacza pod użytkownika, który skończył pracę, lecz nie poinformował o tym serwera. W najprostszym przypadku mogłoby to być po prostu wykorzystanie tego samego komputera. W bardziej zaawansowanym przypadku można próbować podłączać się do serwera jako użytkownik, który skończył pracę używając technik podszywania się (*spoofing*).

Jedynym pewnym sposobem usunięcia tego rodzaju zagrożeń jest pozwolenie użytkownikowi na wylogowanie się – a więc poinformowanie go, że skończył już pracę. Oczywiście rozwiązanie to jest bezpieczne tylko w przypadku, jeśli użytkownicy będą je stosować. Jeśli będą oni kończyć pracę z systemem przez zwykłe zamknięcie przeglądarki, sytuacja nie ulegnie poprawie.

4.5.3. Łatwy fizyczny dostęp do serwera

Oprócz zabezpieczenia od strony sieci, serwer, na którym znajdują się dane, musi być także zabezpieczony przed fizycznym dostępem. Nie chodzi tu tylko o zabezpieczenie komputera przed kradzieżą czy uszkodzeniem fizycznym. Często potencjalnemu włamywaczowi wystarczy kilka sekund przy konsoli serwera, żeby zdobyć krytyczne dane (na przykład plik z hasłami), które posłużyć mogą do późniejszego włamania z sieci. Konieczne jest stworzenie w ramach przedsiębiorstwa czy organizacji procedur limitujących dostęp do pomieszczeń, w których znajdują się serwery.

4.5.4. Zdobywanie informacji na temat serwera

Wszelkie informacje dotyczące organizacji przedsiębiorstwa są niezwykle cenne dla potencjalnych włamywaczy. W szczególności utrudniony powinien być dostęp do informacji, takich jak: lista użytkowników, nazwa administratora, rodzaj i wersje używanego oprogramowania systemowego i użytkowego. Oczywiście utajnienie tego typu informacji nie zawsze jest możliwe (szczególnie w przypadku ataku dokonywanego przez pracownika firmy), jednak dostęp do nich nie powinien być z definicji powszechny.

5. Podsumowanie

W pracy przeprowadzono analizę zagrożeń dla aplikacji internetowych, szczególnie takich, które udostępniają użytkownikom zawartość baz danych. Po wstępnej, ogólnej

charakterystyce źródeł zagrożeń przedstawiono szczegółową analizę rodzajów ataków hakerów na systemy internetowe. W każdym przypadku przedyskutowano metody zabezpieczeń przed takimi atakami.

LITERATURA

1. Building Secure ASP.NET Applications: Authentication, Authorization, and Secure Communication,http://msdn.microsoft.com/library/default.asp?url=/library/enus/dnnetsec/html/secnetlpMSDN.asp.
2. Castano S., Fugini M. G., Martella G., Samarati P.: Database Security. Addison-Wesley Publishing Company, 1995.
3. IP Security Protocol (ipsec) Charter, http://www.ietf.org/html.charters/ipsec-charter.html.
4. Maiwald E.: Bezpieczeństwo w Sieci. Wydawnictwo Edition, Warszawa 2000.
5. Stokłosa J., Bilski T., Pankowski T.: Bezpieczeństwo danych w systemach informatycznych. Wydawnictwo Naukowe PWN, Warszawa 2001.
6. SSL 3.0 Specification, http://wp.netscape.com/eng/ssl3/.

Recenzent: Dr inż. Andrzej Białas

Wpłynęło do Redakcji 8 kwietnia 2003 r.

Abstract

Security is a base topic in web applications. The main aim is to understand sources of threats in order to avoid them. This paper provides an overview of the different types of threats for web-based applications which are using databases.

The first chapter contains an introduction. The second chapter describes an architecture of database web-based systems. The multi-tier architecture consists of the following components: browser, web server, application server and database server. A commonly used web application deployment model is shown in Figure 1. Every part of this architecture, especially sensitive data transfer between application tiers, gives opportunities to be a source of danger.

Different types of threats are presented in the third chapter. Generally threats can be divided into two groups: the first – on-purpose activity of non-authorized users and the latter

- threats which are the result of accidental events such as hardware or equipment failure. On-purpose activity of non-authorized users is information stealing, attacks or sabotage.

Hackers attacks are specially dangerous, for this reason special attention is paid to these aspects in the paper. Chapter four describes main types of hackers attacks. Many attacks take advantage of errors made by application developers. Early design of authentication and authorization eliminates a high percentage of application vulnerabilities. Hackers can also avail oneself of administrators oversights and break the system by unsafe service. Sometimes sniffing interesting information in unprotected communication protocols can affect loosing important data or even breaking the system with sniffed password. Protocols SSL (Fig. 2) and IPSec (Fig. 3) offer secure communication and solve many problems connected with hackers . In many cases administrators can't avoid a disaster. It is because of breaking defective services or operating systems known as safe, which is independent of administrators. But research has shown that the most deceptive part of security system is still human, common users.

Adresy

Paweł KASPROWSKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, kasprowski@polsl.pl .

Stanisław KOZIELSKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, kozielski@zti.iinf.polsl.gliwice.pl .

Piotr KUŹNIACKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, p.kuzniacki@zti.iinf.polsl.gliwice.pl .

Tadeusz PIETRASZEK: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, tadekp@polsl.gliwice.pl .