ZESZYTY NAUKOWE POLITECHNIKI RZESZOWSKIEJ 294, Elektrotechnika 35
RUTJEE, z. 35 (2/2016), kwiecień-czerwiec 2016, s. 67-77

Bartosz KOWAL[1]
Paweł DYMORA[2]
Mirosław MAZUREK[3]

WYBRANE ATAKI NA SYSTEMY BAZODANOWE

Systemy bazodanowe oraz zawarte w nich dane są jednym z najważniejszych elementów współczesnego świata informatyki. W obecnych czasach rośnie zagrożenie związane z bezpiecznym przechowywaniem danych. Celem tego artykułu jest dokonanie klasyfikacji oraz analizy wybranych typów ataków na system zarządzania bazami danych. W artykule dokonano klasyfikacji ataków, scharakteryzowano i przeprowadzono wybrane ataki na RDBMS w szczególności ataki DoS, wnioskowanie, ataki socjotechniczne, testy penetracyjne, podsłuchiwanie pakietów oraz ataki z wykorzystaniem luk w programach - SQL Injection.

Słowa kluczowe: systemy bazodanowe, ataki na DBMS, SQL Injection, sniffing

1. Klasyfikacja ataków na systemy bazodanowe

Atakiem na systemy bazodanowe nazywamy szkodliwe działanie, którego celem jest zamiana, usunięcie, dodanie, zniszczenie, zbieranie danych lub kradzież nośników fizycznych na których znajduje się system bazodanowy. Podstawowym wymogiem bezpieczeństwa systemów bazodanowych jest zapewnienie ochrony danych i ciągłości pracy serwera, zgodnie z obowiązującymi przepisami prawnymi, ustawami dotyczącymi przechowywania i przetwarzania danych, normami, jak i z regulaminem przedsiębiorstwa/firmy. Kategorie ataków na systemy bazodanowe są dosyć podobne, do ataków na systemy i sieci komputerowe. Do zdefiniowania kategorii ataków na systemy bazodanowe należy wziąć pod uwagę takie czynniki jak [1,2]:

- Aktywność;
- Skutek;
- Zamiar;
- Miejsce.

[1] Autor do korespondencji: Bartosz Kowal, Politechnika Rzeszowska, b.kowal.1991@gmail.com
[2] Paweł Dymora, Politechnika Rzeszowska, Katedra Energoelektroniki, Elektroenergetyki i Systemów Złożonych, pawel.dymora@prz.edu.pl
[3] Mirosław Mazurek, Politechnika Rzeszowska, Katedra Energoelektroniki, Elektroenergetyki i Systemów Złożonych, miroslaw.mazurek@prz.edu.pl

1.1. Podział ataku ze względu na ich aktywność

Ze względu na aktywność można wyróżnić ataki [1,2]:

- Aktywne;
- Pasywne.

Atak aktywny na system bazodanowy ma za zadanie przeprowadzenie takiego działania, w którym użytkownik traci kontrolę nad dostępem do zasobów systemu bazodanowego. Skutkiem takiego działania, może być modyfikacja danych przesyłanych między klientem-serwerem, całkowite lub częściowe zablokowanie połączenia z serwerem. Jest to kategoria ataków, które można wykryć i częściowo im przeciwdziałać, lecz pełne zabezpieczenie byłoby dosyć trudne do zrealizowania, gdyż wymagałoby to ochrony wszystkich urządzeń i mediów transmisyjnych między klientem, a serwerem [1,2].

Atakiem pasywnym na systemy bazodanowe nazywamy zaś próby odczytania informacji zawartych danych, bez ich wewnętrznej modyfikacji. Do ataków pasywnych można zaliczyć kopiowanie danych na nośniki danych, podsłuchiwanie pakietów, zbieranie i analiza danych. Ataki te są trudne do wykrycia, ponieważ nie są powiązane z innymi danymi, a większość danych jest dostępna publicznie [1,2].

1.2. Podział ataku ze względu na skutki ataków

Ze względu na skutki ataków można wyróżnić ataki:

- Udane;
- Nieudane.

Z atakiem udanym mamy do czynienia, gdy atak aktywny bądź pasywny zakończy się sukcesem. Z serwera bazodanowego zostaną pobrane, zmodyfikowane, bądź usunięte dane, zostanie przeprowadzony atak odmowy dostępu usług (ang. Denial of Service, DoS), czy wykorzystanie luk programowych SQL Injection, bądź nastąpi kradzież, zniszczenie nośników danych przez np. pożar, powódź itp. [1,2].

Atakiem nieudanym nazywa się każdą czynność, która zakończy się porażką. Przykładem takiego nieudanego ataku może być wykorzystanie ataku SQL Injection na zabezpieczonej stronie, czy też nieudana próba skanowania portów serwera bazodanowego [1,2].

1.3. Podział ataku ze względu na ich zamiar

Ze względu na zamiar można wyróżnić ataki:

- Zamierzone;
- Niezamierzone.

W atakach zamierzonych, atakujący zdaje sobie sprawę z tego co robi i jakie może ponieść konsekwencje prawne i finansowe. Celowym atakiem może

być np. atak odmowy dostępu do usług, mający na celu całkowite lub częściowe zablokowanie dostępu do serwera [1,2].

W atakach niezamierzonych, atakujący nieświadomie lub przypadkowo dokonuje ataku na system bazodanowy. Najlepszym przykładem może być grupa osób przypadkowo wpisująca losowy ciąg znaków w aplikacji, co może doprowadzić do wystąpienia błędów i sprawić, że serwer trzeba będzie uruchomić ponownie [1,2].

1.4. Podział ataku ze względu na miejsce ich przeprowadzenia

Ze względu na miejsce przeprowadzenia ataku można wyróżnić ataki:

- Lokalne;
- Zdalne.

Do ataków lokalnych (wewnętrznych) zaliczymy wszystkie zagrożenia związane ze sprzętem oraz infrastrukturą sieciową. Do ataków lokalnych należy przede wszystkim nieuprawniony dostęp do urządzeń, kradzież nośników z kopiami zapasowymi, niszczenie sprzętu, awarie prądu, pożary, powodzie, oraz także ataki z wewnętrznej infrastruktury sieciowej np. komputera znajdującego się w tej samej sieci komputerowej co serwer bazodanowy. Jest to jedno z najpoważniejszych i najkosztowniejszych zagrożeń na systemy bazodanowe, ponieważ w porównaniu do ataków zdalnych nie da się przewidzieć jaki typ zagrożenia lokalnego może nastąpić, podpięcie się do sieci lokalnej może spowodować wyciek danych [1,2].

Ataki zewnętrzne to ataki przeprowadzane z sieci atakującego na serwer bądź sieć komputerową, gdzie znajduje się serwer bazodanowy. Najlepszym przykładem takiego ataku jest atak DoS lub DDoS na adres IP serwera. Atakujący w tej kategorii nie ma fizycznej możliwości dostępu do danych czy też urządzeń znajdujących się na serwerze [1,2].

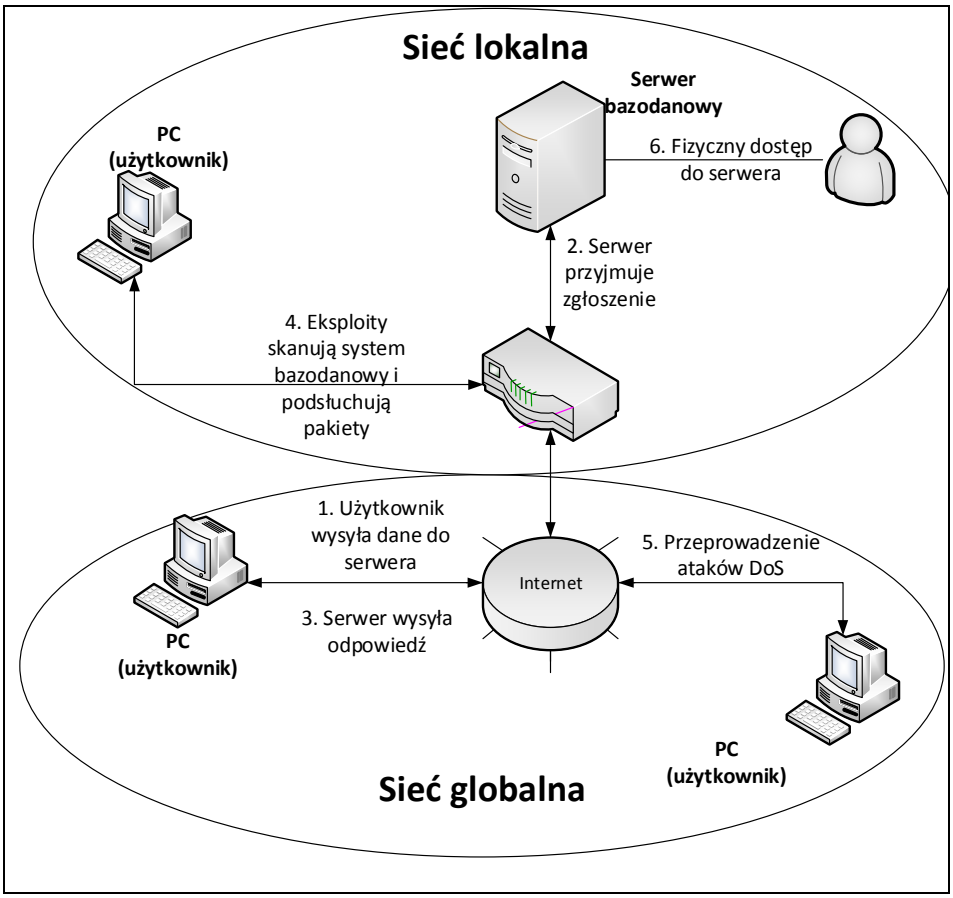
2. Analiza ataków na systemy bazodanowe

2.1. Topologia serwera bazy danych oraz sieci

Na Rys. 1. została przedstawiona topologia obsługująca bazę danych poprzez lokalny serwer z dostępem przez sieć globalną. Zasadę działania można przedstawić w ten sposób:

1. *Użytkownik wysyła dane do serwera* – użytkownik wprowadza dane, żeby serwer mógł wysłać zapytanie SQL do bazy.
2. *Serwer przyjmuje zgłoszenie* – serwer otrzymuje zapytanie SQL, wykonuje odpowiednie polecenie.
3. *Serwer wysyła odpowiedź* – serwer wysyła dane do użytkownika w formie odpowiedniego komunikatu.

4. *Eksploity skanują system bazodanowy i podsłuchują pakiety* – programy skanują porty i usługi używane przez system bazodanowy, oraz przechwytuje wszystkie pakiety wysyłane między komputerem PC, a serwerem bazodanowym.
5. *Przeprowadzenie ataków DoS* – przeprowadzenie wybranych ataków DoS na serwer bazodanowy.
6. *Fizyczny dostęp do serwera* – użytkownicy mają fizyczny dostęp do serwera bazodanowego.



Rys. 1. Topologia serwera bazodanowego i sieci komputerowej

Fig. 1. Database server and Network topology

2.2. Atak blokady usług, DoS

Atak blokady usług (ang. Denial of Service, DoS), to atak mający na celu obciążenia zasobów serwera, oprogramowania, a także łączy sieciowych poza granicę ich wydajności. Zazwyczaj celem ataku DoS na systemy bazodanowe jest zablokowanie dostępu do serwera bazodanowego [3].

Atak DoS może zostać wymierzony w serwer HTTP, bądź w konkretny port serwera bazodanowego. Dla przykładu w systemie bazodanowym Oracle, atakuje się najczęściej port usługi Oracle TNS Listener (1521), która odpowiada za komunikację użytkownik – serwer [3].

Jednym z przykładów ataku DoS, może być złe napisanie funkcji czy procedury w PL/SQL i wywołanie jej w aplikacji. Najpoważniejszym błędem jest stworzenie procedury zawierającej w sobie pętle bez warunku jej zakończenia np. LOOP co przedstawiono na Rys. 2. Czynność ta będzie się wykonywać w nieskończoność, co w konsekwencji doprowadzi do 100% wykorzystania zasobów serwera i spowoduje częściowe lub całkowite zablokowanie możliwości obliczeniowych dla pozostałych operacji.

```
SQL*Plus: Release 11.2.0.1.0 Production on N Mar 6 21:07:07 2016

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

Proszę podać nazwę użytkownika: sys as sysdba
Proszę podać hasło:

Połączono z:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> declare
  2  begin
  3  loop
  4  dbms_output.put_line('Przykladowy atak DoS');
  5  end loop;
  6  end;
  7  /
```

Rys. 2. Atak DoS – przykład złej procedury PL/SQL

Fig. 2. DoS attack - bad PL/SQL procedure example

2.3. Wnioskowanie

Wnioskowanie to nic innego jak zbieranie danych, które są powszechnie dostępne np. ze stron zawierających informacje statystyczne. Ten typ ataku polega przede wszystkim na badaniu związków między podanymi danymi. Często dane są upubliczniane, część z tych danych może zawierać informacje poufne. Jeśli kilka takich zestawów danych zostanie upublicznionych, atakujący może połączyć ze sobą dane i w ten sposób zyskać informacje tj. adres zamieszkania, usługi z jakich dana osoba korzysta, wiek, nawyki [2].

2.4. Socjotechnika

Socjotechnika to nic innego jak wywieranie wpływu na ludzi poprzez stosowanie różnych metod działania mających na celu osiągnięcie określonego

celu, w tym przypadku kradzież informacji z systemu bazodanowego np. poprzez mocną siłę perswazji lub doświadczenie. W socjotechnice występuje wiele różnych metod ataków w zależności od potrzeby, mogą być one realizowane na bieżąco lub zostaną one połączone ze sobą tworząc bardziej zaawansowany atak. Obecnie ataki socjotechniczne są coraz częściej wykorzystywane. W rozmowie telefonicznej można podszyć się pod administratora serwera i wykraść dane np. logowania czy inne wrażliwe informacje. Zazwyczaj użytkownik jest proszony o ponowne wpisanie danych i wysłanie ich na odpowiedni adres e-mail lub przez podaną stronę internetową. Użytkownik wprowadzając takie informację przekazuje je agresorom, co w konsekwencji doprowadza do wycieku informacji. Innym częstym zagrożeniem socjotechnicznym jest kradzież urządzeń. W dzisiejszych czasach na przenośnych urządzeniach znajduje się co raz więcej wrażliwych danych tj. hasła do kont pocztowych, loginy i hasła do serwisów. Brak zaszyfrowanych urządzeń może spowodować wyciek informacji z danego urządzenia, atakujący może dostać się do naszego konta i zrobić z nim praktycznie co zechce [4].

2.5. Testy penetracyjne oraz podsłuchiwanie pakietów

Testy penetracyjne mają za zadanie przeprowadzenie kontrolowanego skanowania i ataku na sieć komputerową. Celem takiego testu jest ocena stanu zabezpieczeń danego systemu bądź usług. Takie skanowanie może dostarczyć osobie atakującej informacje dotyczące luk w oprogramowaniu, błędy w zabezpieczeniach, wersję systemu operacyjnego serwera oraz wiele innych cennych informacji. Większość wykorzystywanych luk w systemach jest zależna od zainstalowanej wersji oprogramowania [5].

W niniejszym przykładzie do skanowania sieci użyto narzędzia NMAP, opartego na licencji GNU GPL. Jak pokazano na Rys. 3. można dowiedzieć się o skanowanym systemie np. jakie są używane porty, jakie aplikacje sieciowe pracują na skanowanym komputerze itp. Najważniejszą odczytaną informacją jest: port Oracle TNS Listener (1521), który odpowiada za połączenia sieciowe między serwerami i hostami [4,5].

Podsłuchiwanie i przechwytywanie pakietów (ang. Sniffing) ma na celu gromadzenie i analizę informacji przesyłanych między stacją roboczą, a serwerem. Atakujący może z tych pakietów dowiedzieć się jaka wersja systemu bazodanowego jest zainstalowana na serwerze, jak nazywa się logowany użytkownik, sprawdzić jakie polecenia SQL zostały wysłane na serwer, czy też sprawdzić z jakim portem serwera łączy się stacja robocza. Sniffing w sieci ma działanie pasywne, administrator serwera nie wie, które pakiety są przechwytywane. Z działań przedstawionych na Rys. 4-5 można dowiedzieć się jaka jest zainstalowana wersja systemu bazodanowego na serwerze (odczytana wartość to: Oracle 11g) oraz na jakim numerze IP i porcie (IP: 102.168.0.111, port 1521) [5].

```
root@kali:~# nmap -sV 192.168.0.111

Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-03-06 23:39 CET
Nmap scan report for 192.168.0.111
Host is up (0.0014s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE      VERSION
135/tcp   open  msrpc        Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows 98 netbios-ssn
445/tcp   open  microsoft-ds (primary domain: WORKGROUP)
1521/tcp  open  oracle-tns   Oracle TNS Listener
5357/tcp  open  http         Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
49152/tcp open  msrpc        Microsoft Windows RPC
49153/tcp open  msrpc        Microsoft Windows RPC
49154/tcp open  msrpc        Microsoft Windows RPC
49155/tcp open  msrpc        Microsoft Windows RPC
```

Rys. 3. Skanowanie serwera bazodanowego narzędziem NMAP

Fig. 3. Scanning of the database server by the NMAP tool

```
0070  28 44 45 53 43 52 49 50  54 49 4f 4e 3d 28 43 4f   (DESCRIP TION=(CO
0080  4e 4e 45 43 54 5f 44 41  54 41 3d 28 53 45 52 56   NNECT_DA TA=(SERV
0090  49 43 45 5f 4e 41 4d 45  3d 58 45 29 28 43 49 44   ICE_NAME =XE)(CID
00a0  3d 28 50 52 4f 47 52 41  4d 3d 43 3a 5c 61 70 70   =(PROGRA M=C:\app
00b0  5c 41 64 6d 69 6e 69 73  74 72 61 74 6f 72 5c 70   \Adminis trator\p
00c0  72 6f 64 75 63 74 5c 31  31 2e 32 2e 30 5c 63 6c   roduct\1 1.2.0\cl
00d0  69 65 6e 74 5f 31 5c 42  49 4e 5c 73 71 6c 70 6c   ient_1\B IN\sqlpl
00e0  75 73 2e 65 78 65 29 28  48 4f 53 54 3d 43 4c 49   us.exe)( HOST=CLI
00f0  45 4e 54 29 28 55 53 45  52 3d 43 6c 69 65 6e 74   ENT)(USE R=Client
0100  29 29 29 28 41 44 44 52  45 53 53 3d 28 50 52 4f   )))(ADDR ESS=(PRO
0110  54 4f 43 4f 4c 3d 54 43  50 29 28 48 4f 53 54 3d   TOCOL=TC P)(HOST=
0120  31 39 32 2e 31 36 38 2e  30 2e 31 31 31 29 28 50   192.168. 0.111)(P
0130  4f 52 54 3d 31 35 32 31  29 29 29                  ORT=1521 )))
```

Rys. 4. Podsłuchany pakiet wysłany do serwera bazodanowego Oracle

Fig. 4. Sniffed packet sent to the Oracle database server

```
0000  08 00 27 9d 3c 6b 08 00  27 6a 67 28 08 00 45 00   ..'.<k.. 'jg(..E.
0010  00 fb 08 d7 40 00 80 06  00 00 c0 a8 00 72 c0 a8   ....@... .....r..
0020  00 6f c0 82 05 f1 34 80  fb 65 5c ec 84 c2 50 18   .o....4. .e\...P.
0030  00 fe 83 1f 00 00 00 d3  00 00 06 00 00 00 00 00   ........ ........
0040  03 5e 15 61 80 00 00 00  00 00 00 01 e1 00 00 00   .^.a.... ........
0050  01 0d 00 00 00 01 01 00  00 00 00 01 00 00 00 00   ........ ........
0060  00 00 00 00 00 00 00 00  00 01 00 01 01 01 00 00   ........ ........
0070  00 00 00 00 00 00 01 01  00 00 00 00 00 00 00 00   ........ ........
0080  00 00 00 00 00 00 fe 40  73 65 6c 65 63 74 20 2a   .......@ select *
0090  20 66 72 6f 6d 20 65 6d  70 6c 6f 79 65 65 73 20    from em ployees
00a0  77 68 65 72 65 20 66 69  72 73 74 5f 6e 61 6d 65   where fi rst_name
00b0  3d 27 53 74 65 76 65 6e  27 20 6f 72 20 66 69 72   ='Steven ' or fir
00c0  73 74 5f 6e 61 6d 65 3d  0b 27 41 6c 65 78 61 6e   st_name= .'Alexan
00d0  64 65 72 27 00 01 00 00  00 00 00 00 00 00 00 00   der'.... ........
00e0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00f0  00 01 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0100  00 00 00 00 00 00 00 00  00                       ........ .
```

Rys. 5. Podsłuchany pakiet z zapytaniem SQL

Fig. 5. Sniffed packet with SQL query

2.6. Wykorzystanie luk w programach SQL Injection

Błędy w oprogramowaniu to jedna z najbardziej popularnych metod dotarcia do danych zawartych w systemach bazodanowych. SQL Injection jest to typ ataku wykorzystujący luki w zapytaniach SQL. Atakujący ma możliwość wpływu na to jakie zapytanie zostanie wysłane na serwer bazodanowy. Będąc w stanie wpływać na to co zostanie wysłane, można łatwo odczytać zawarte informacje w bazie danych. Sam atak SQL Injection nie wpływa bezpośrednio na sam kod aplikacji, jego zasada działania polega na wysłaniu odpowiednio zmodyfikowanego zapytania na serwer. Wiele aplikacji w dzisiejszych czasach oparta jest na technologiach takich jak JSP, ASP, PHP itp. Pobierają one dane od użytkownika np. imię i wykorzystują je do sformułowania zapytania do bazy danych. Najlepszym przykładem ataku SQL Injection jest odczytanie podstawowych informacji o użytkowniku po podaniu podstawowych danych takich jak: PESEL oraz nazwisko przez stronę internetową napisaną w PHP (Rys. 6), która wysyła zapytanie do serwera, a następnie je wyświetla za pomocą tabeli [6].

Ataki SQL Injection można podzielić na [6]:

- *Modyfikacje zapytań* - polegający na zmianie wartości zmiennych, które zostaną przesłane do serwera bazodanowego.
- *Blokowanie serwera (atak DoS)* – taka zamiana wartości zmiennych, aby serwer bardzo długo obliczał wynik np. przez użycie funkcji benchmark w systemie bazodanowym MYSQL/.
- „*Ślepy atak*" - wykonywanie ataku typu SQL Injection na stronie lub serwerze, która nie wyświetla komunikatów błędów.
- *Użycie dodatkowych zapytań* – są to ataki SQL Injection, które mogą zawierać kilka zapytań na raz, np.: `SELECT * FROM employees WHERE first_name = 'Valli '; DROP TABLE testowa; SELECT '1';.`

Gdy witryna nie jest zabezpieczona przed atakami SQL Injection np. przez wiązanie zmiennych, łatwo można wyciągnąć dane z serwera bazodanowego, ponieważ serwer bazodanowy sprawdza tylko czy składnia zapytania SQL się zgadza. Badając składnie zapytania: `SELECT * FROM users where PESEL="$pesel" AND nazwisko="$nazwisko";` można dostrzec, że w miejsce zmiennych „PESEL" i „nazwisko" można wpisać jeden z warunków logicznych, dzięki którym zapytanie zawsze będzie prawdziwe, w związku z czym zostanie wyświetlona cała tabela z której wczytywane są informacje. Użycie logicznego warunku w ten sposób sprawia, że komenda do bazy przyjmuje następującą postać: `SELECT * FROM users WHERE PESEL = '1' or '1'='1' AND nazwisko = '1';`. Wynik polecenia pokazano na Rys.7.

```
<?php
$conn = oci_connect('user', 'be4tds', '192.168.0.111/XE');
if ((isset($_POST['pesel']))&&(isset($_POST['nazwisko']))) {
if (!empty($imie)) {
$$stid = oci_parse($conn, 'SELECT * FROM users where PESEL="$pesel"
AND nazwisko="$nazwisko"'; ');
$imie ."'");
if (!$stid) {
echo "error1";
$e = oci_error($conn);
trigger_error(htmlentities($e['message'],ENT_QUOTES), E_USER_ERROR);}
$r = oci_execute($stid);
if (!$r) {$e = oci_error($stid);
trigger_error(htmlentities($e['message'],ENT_QUOTES), E_USER_ERROR);
} else {
print "<table border='1'>\n";
while ($row = oci_fetch_array($stid, OCI_ASSOC+OCI_RETURN_NULLS)) {
print "<tr>\n";
foreach ($row as $item) {
print"<td>".($item!==null?htmlentities($item,ENT_QUOTES):" ").
```

Rys. 6. Przykład witryny w PHP
Fig. 6. The example of PHP site

100	Steven	King	SKING	515.123.4567	03/06/17	AD_PRES	24000			90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	05/09/21	AD_VP	17000		100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	01/01/13	AD_VP	17000		100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	06/01/03	IT_PROG	9000		102	60
104	Bruce	Ernst	BERNST	590.423.4568	07/05/21	IT_PROG	6000		103	60

Rys. 7. Odczytane dane za pomocą ataku SQL Injection
Fig. 7. The read data using SQL Injection Attack

Kolejnym przykładem ataku SQL Injection (Blind SQL Injection) jest wykorzystanie wyświetlanych błędów i ostrzeżeń przez serwer http do sprawdzenia poprawności wpisywanych zapytań. Jest to jeden z najczęściej wykorzystywanych sposobów na sprawdzenie czy istnieją wpisywane tabele, dane itp. Użytkownik wpisuje losowe dane i sprawdza, jakie informacje zwraca serwer. Inną metodą ataku jest wykorzystanie komendy `union select` …, która tworzy zapytania do wielu tabel i zwraca jeden wynik. Dzięki temu można łączyć tabele. Żeby się to udało liczba kolumn tabel źródłowych i dodawanej przez komendę `union` musi się zgadzać, z tego wniosek, że przydają się informacje, które dzięki Blind SQL Injection można zdobyć. Niekoniecznie trzeba znać typ danych, a tam gdzie się go nie zna, można spróbować wpisać wartości `null` [6].

Sposobami obrony przed atakami SQL injection są [6]:

- Sprawdzanie do jakiego typu należą dane wejściowe (podane dane powinny być odpowiedniej długości i typu np. nazwisko nie może posiadać cyfr);
- „Oczyszczanie" danych wejściowych;
- Wiązanie zmiennych (w Oracle: `oci_bind_by_name`), redukuje ataki SQL Injection, ponieważ przechowywane dane nie są traktowane jako część instrukcji SQL. Nie trzeba wpisywać apostrofów lub innych znaków specjalnych.

3. Podsumowanie

Każdy z systemów zarządzania bazami danych daje duże, a czasami ogromne możliwości w zakresie zabezpieczania zarówno danych zawartych w plikach serwera bazy danych, jak i zapewnieniu ciągłości działania serwerów bazodanowych. Zarządzanie zaawansowanym systemem bazodanowym oraz jego poprawne konfigurowanie jest nadrzędnym i niejednokrotnie ciężkim zadaniem dla administratora, od którego wymaga się dużej wiedzy w tej dziedzinie. W większości przypadków administrator zarządza już stworzonymi aplikacjami, musi on polegać na odpowiedzialności i dokumentacji programistów tworzących aplikacje. Jedynym dostępnym dla niego narzędziem jest test penetracyjny aplikacji oraz audyt bezpieczeństwa w miejscu gdzie pracuje. Nie można nigdy wykluczyć błędów ludzkich podczas programowania, czy używania programów. Powinno się też wprowadzić odpowiednią politykę bezpieczeństwa, aby nikt niepowołany nie miał dostępu do serwerów, czy sieci komputerowej.

Z punktu widzenia umiejscowienia ataków, można stwierdzić, że ataki lokalne tj. ataki znajdujące się w wewnętrznej sieci komputerowej lub przeprowadzone w budynku firmy są najniebezpieczniejsze i trudno się przed nimi obronić.

Literatura

[1] Graves K.: CEH: Official Certified Ethical Hacker Review Guide: Exam 312-50, Sybex, pp. 1-16, 2007.

[2] Kulkarni S., Urolagin S.: Review of Attacks on Databases and Database SecurityTechniques. International Journal of Emerging Technology and Advanced Engineering, vol. 2, Issue 11, pp. 253-263, 2012.

[3] Ben-Natan R.: HOWTO Secure and Audit Oracle 10g and 11g, Auerbach Publications, s. 29-189, 2009.

[4] Mitnick K., Simon W.: Sztuka podstępu. Łamałem ludzi, nie hasła, Helion, 2003.

[5] Muniz J., Lakhani A.: Kali Linux Testy penetracyjne, Helion, 2014.

[6] Clarke J.: SQL Injection Attacks and Defense, Syngress Publishing, 2009.

ATTACKS ON DATABASES SYSTEMS

S u m m a r y

Database systems and the data they contain, are one of the most important elements of the modern world of computer science. Nowadays, threat greatly increases for the safe storage of data. The purpose of this article is to classify and analyze the selected types of attacks on the database management system. At the beginning selected attacks on the DBMS are classified and described. Afterwards, exemplary attacks were carried out in the test environment. The attacks on the DBMS includes: DDoS attacks, inference, social engineering, penetration test, packet sniffing and attacks using vulnerabilities in programs like SQL Injection.

Keywords: database systems, attacks on DBMS, SQL Injection, sniffing

DOI: 10.7862/re.2016.11

Tekst złożono w redakcji: maj 2016
Przyjęto do druku: czerwiec 2016