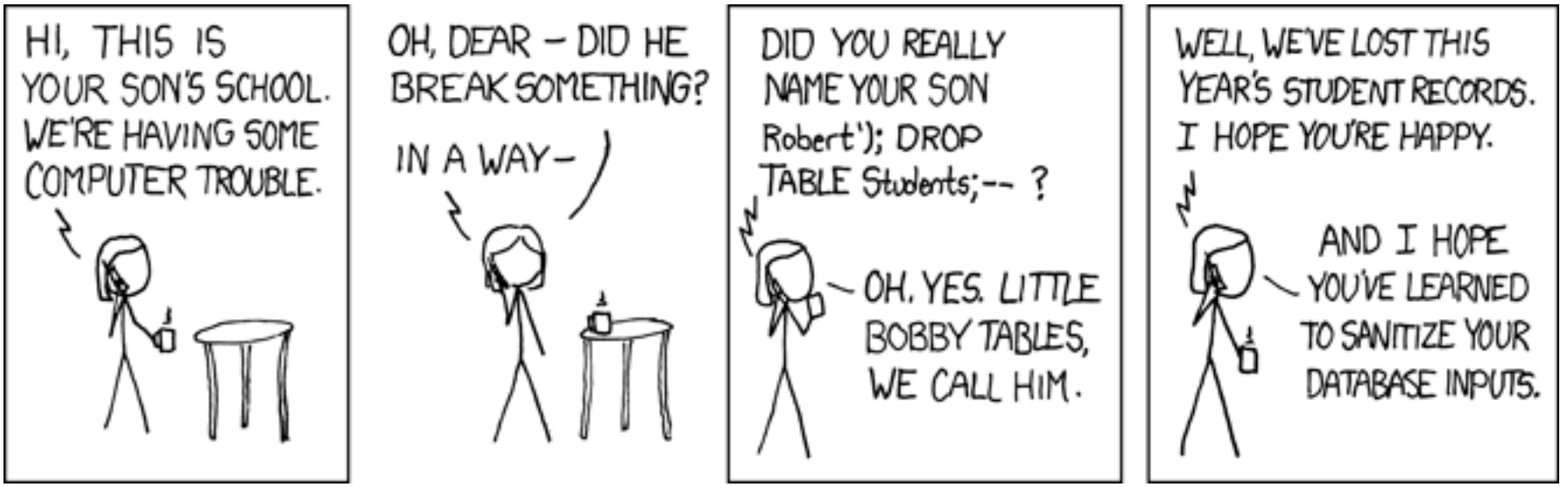


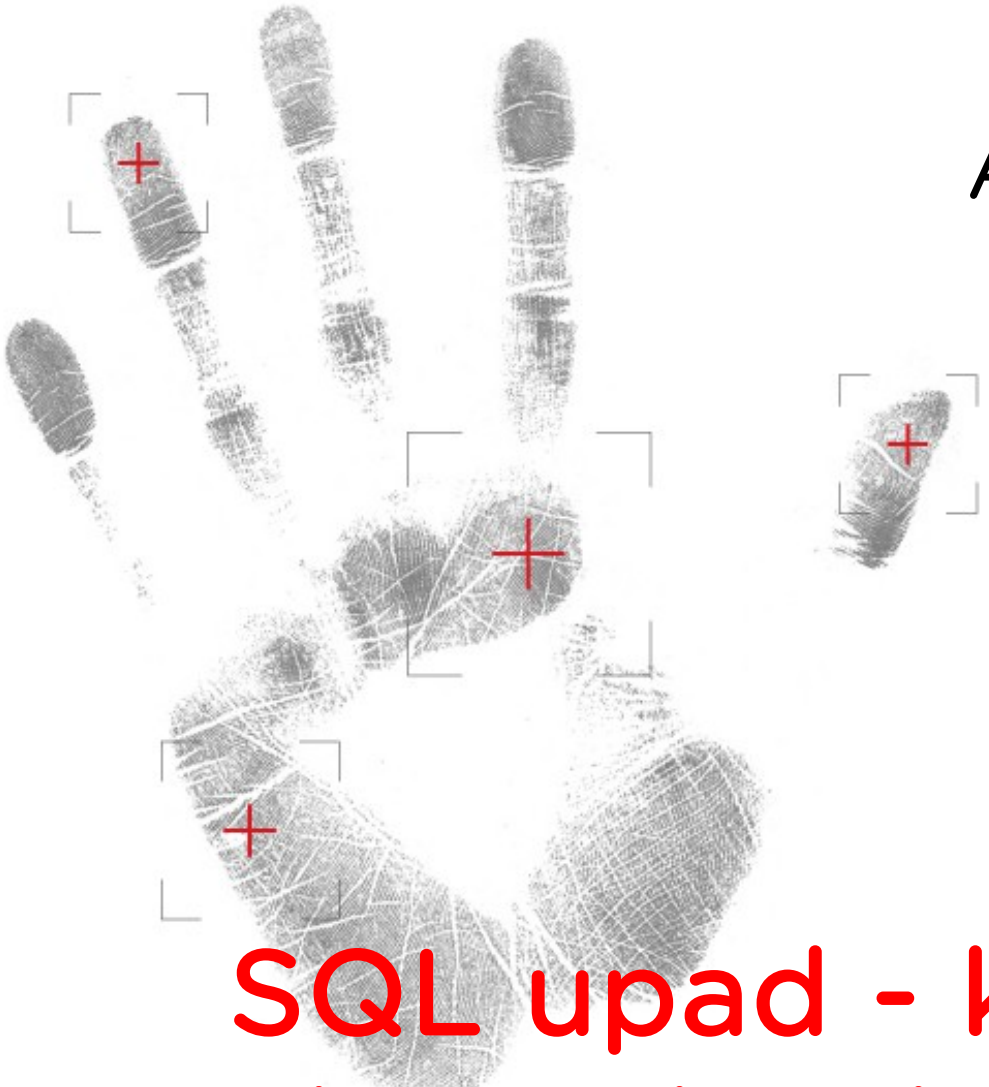
Tom Kyte (asktom) u svom članku, koji govori o SQL upadu, napravio je grešku, koja omogučava SQL upad.

http://tkyte.blogspot.com/2012/02/all-about-security-sql-injection.html


ZU 0666',0,0); DROP DATABASE TABLICE

Abakus ARBITER

ORACLE Gold Partner



Boris Oblak
Abakus plus d.o.o.

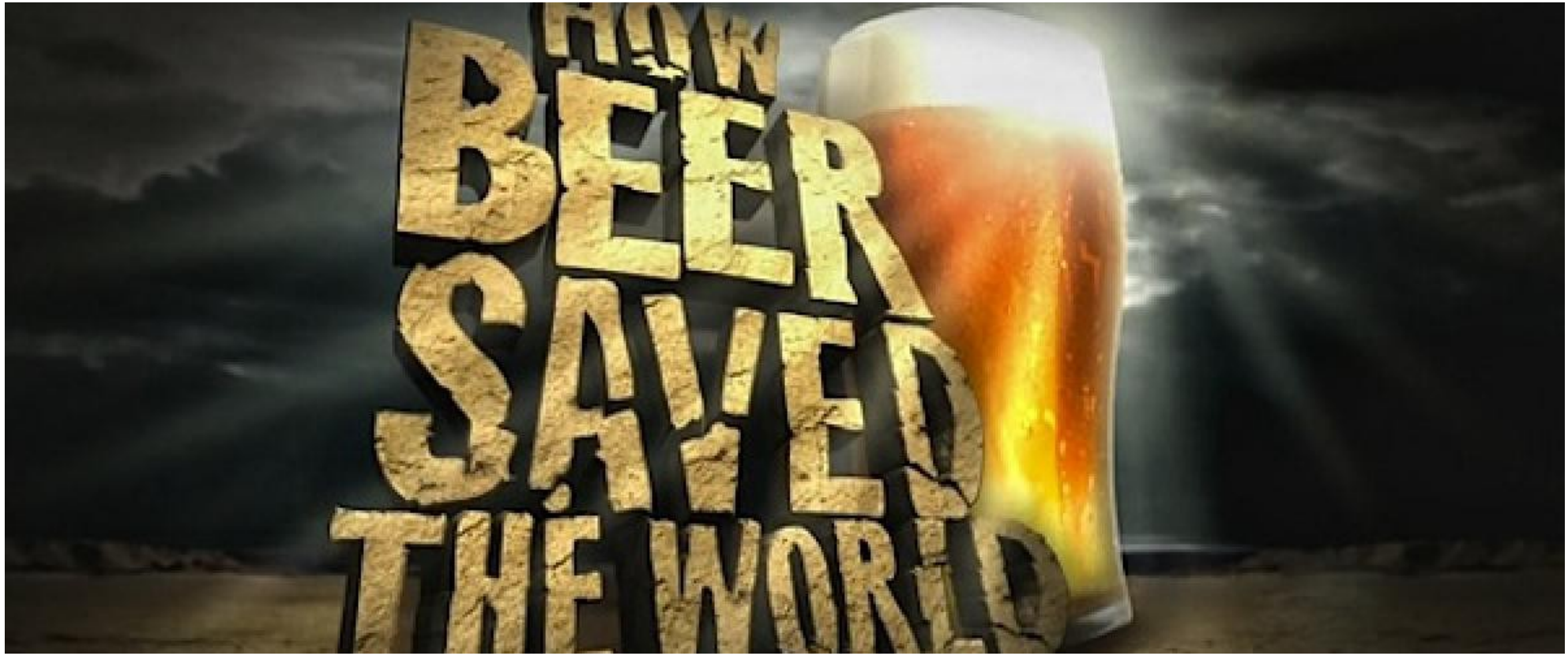
ORACLE | CERTIFIED PROFESSIONAL

SQL upad - krađa 130 milijuna kreditnih kartica

RBC

Real Beer Clusters

20 godina Abakus plus d.o.o.



Abakus
As na disku.

Najveći primjeri krađe u povijesti

- Albert Gonzalez – osuđen na 20 godina zatvora
- sa sučesnicima su upotrijebili SQL injection za upad u sustav

- http://en.wikipedia.org/wiki/Albert_Gonzalez

Abakus
As na disku.

Svijest o opasnosti

- SQL upad je najveća opasnost
- premalo ljudi je svijesno te opasnosti
- aplikacija prima SQL naredbu od neprovjerenih izvora (korisnički unos) i izvodi je

Bind variables

- bind variables
- bez upotrebe je kod manje siguran
- primjer: unos korisničkog imena i zaporke

```
select count(*)
  from user_table
 where username = <USER_NAME>
   and password = <PASSWORD>;
```

Bind variables

```
create table user_table
   ( username varchar2(30),
    password varchar2(30) );
insert into user_table
  values ( 'boris',
  'never_guess' );
commit;

SQL> accept Uname prompt "Enter username: "
Enter username: boris

SQL > accept Pword prompt "Enter pass: "
Enter pass: nemam_pojma' or 'x' = 'x
```

Bind variables

- bez »bind variables«

```
select count(*)
     from user_table
    where username = '&Uname'
      and password = '&Pword'
/

old   3:     where username = '&Uname'
new   3:     where username = 'boris'
old   4:       and password = '&Pword'
new   4:       and password = 'nemam_pojma' or 'x'
= 'x

  COUNT(*)
----------
         1
```

Bind variables

- sa »bind variables«

```
variable uname varchar2 (30);
variable pword varchar2 (30);
exec :uname := 'boris';
exec :pword := 'nemam_pojma'' or ''x'' = ''x';

select count(*)
  from user_table
 where username = :uname
   and password = :pword
/
```

```
  COUNT(*)
----------
         0
```

Bind variables

- neželjene nuspojave, ako ne koristimo povezane varijable

```
accept pword prompt "Pass: "
Geslo: hr.fire_employee (1234)

!?
```

SQL upad – najveći problem

- možda sve izgleda prenapuhano ?
- www.google.com - „SQL injection“
  - 5.650.000 hit-ova (rujan 2012)
- ne samo VB (Active Server Pages), ne samo JavaServer Pages, php, ...
- svi jezici koji izvode SQL naredbe, koje su unesene izvana

Stored procedures

- pohranjeni postupak za brisanje zaposlenika

```
create or replace procedure remove_emp (p_schema in
varchar2, p_ename in varchar2)
is
  l_str clob;
begin
  l_str := '
  begin
    delete from ' || p_schema ||
     '.emp where ename = ''' || p_ename || ''';
    delete from ' || p_schema ||
     '.bonus where ename = ''' || p_ename || ''';
  end;';
  execute immediate l_str;
end;
/
```

Stored procedures

```
create table t (id int);

--Preverimo, koliko zapisov imamo:
SQL> select count (*) from emp where ename =
'KING';

  COUNT(*)
----------
         1

SQL> select count (*) from bonus where ename =
'KING';

  COUNT(*)
----------
         1
```

Stored procedures

```
begin
  remove_emp
  ( 'scott',
    'KING''; execute immediate ''drop table t''';
--' );
end;
/
begin
*
ERROR at line 1:
ORA-00942: table or view does not exist
ORA-06512: at line 4
ORA-06512: at "SCOTT.REMOVE_EMP", line 13
ORA-06512: at line 2

SQL> rollback;
```

Stored procedures

```
SQL> select count (*) from emp where ename =
'KING';

  COUNT(*)
----------
         0

SQL> select count (*) from bonus where ename =
'KING';

  COUNT(*)
----------
         1
```

Kako otkriti upad

- jako teško
- forenzička istraživanja nakon akcije - uvijet: uključen AUDIT

Abakus ARBITER

ARBITER

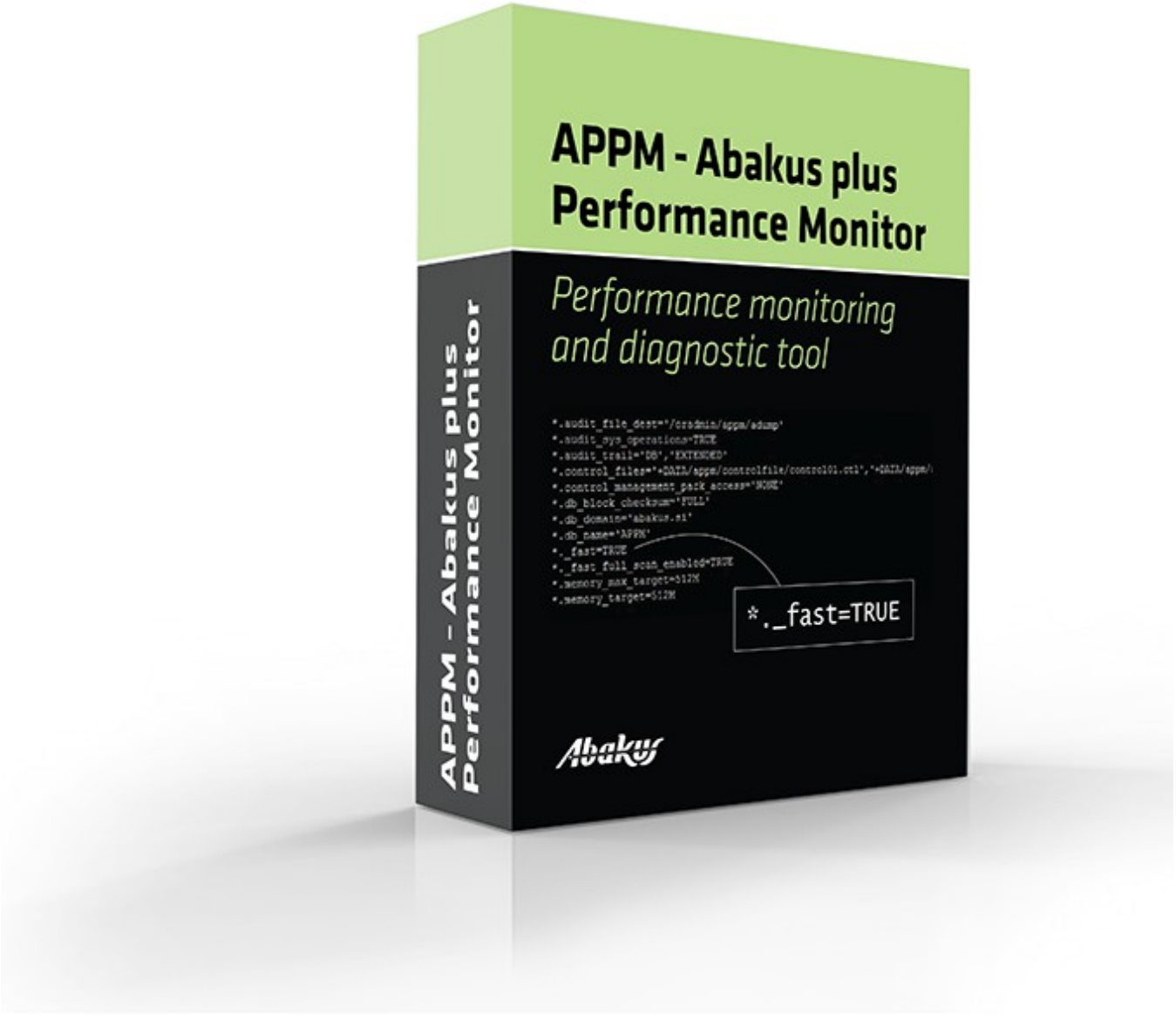
The Trustworthy trail

Kako otkriti upad

- jako teško
- forenzička istraživanja nakon akcije - uvijet: uključen AUDIT
- pretraživanje v$sql - ne radi ako je CURSOR_SHARING = FORCE/SIMILAR

Abakus
As na disku.

Kako otkriti upad



Kako otkriti upad

- jako teško
- forenzička istraživanja nakon akcije - uvijet: uključen AUDIT
- pretraživanje v$sql - ne radi ako je CURSOR_SHARING = FORCE/SIMILAR
- otkriti odakle dolaze literali
- ako se radi o korisničkom unosu, onda imamo ozbiljne probleme

Abakus
As na disku.

Teško odkrivanje

```
CREATE OR REPLACE PROCEDURE inj(p_date IN DATE)
IS
   u_rec all_users%ROWTYPE;
   c     SYS_REFCURSOR;
   l_sql CLOB;
BEGIN
   l_sql := 'select * from all_users
      where created = ''' || p_date || '''';
   dbms_output.put_line(l_sql);
   OPEN c FOR l_sql;
   FOR i IN 1 .. 5
   LOOP
      FETCH c INTO u_rec;
      EXIT WHEN c%NOTFOUND;
      dbms_output.put_line(u_rec.username);
   END LOOP;
   CLOSE c;
END;
```

Teško odkrivanje

- parametar je datum i nije string (otpada or 1=1)
- `where created = ''' || p_date || ''''';`
- `where created = to_date (to_char (p_date));`
- 2x implicitna pretvorba
- često viđen kod

Implicitna pretvorba je zlo

- neželjene nuspojave
  - trunc (datum)
- logične greške
  - 14.10.2012 in 14.10.1912?
  - NLS_DATE_FORMAT = 'dd.mm.rr' (Slovenian)

Implicitna pretvorba je zlo

```
SQL> set serveroutput on;
SQL> exec inj (sysdate);

       select *
        from all_users
       where created = '25.09.12'

PL/SQL procedure successfully completed.
```

Teško spriječiti

```
SQL> alter session set nls_date_format =
'dd.mm.yyyy''' or ''a'' = ''a''';

SQL> exec inj (sysdate);

       select *
        from all_users
       where created = '25.09.2012' or 'a' = 'a'
APPM
ABAKUS
U1
BORIS
REV_SRC_USER

PL/SQL procedure successfully completed.
```

Party počinje

```
SQL> nls_date_format = ''''union select
tname,0,null from tab--'';

SQL> exec inj (sysdate);

       select *
        from all_users
       where created = ''union select
tname,0,null from tab--'
BIN$yn+TlSkTCtDgQIrBPS9M3w==$0
BONUS
DEPT
EMP
SALGRADE

PL/SQL procedure successfully completed.
```

Number

```
CREATE OR REPLACE PROCEDURE inj (p_num IN NUMBER)
IS
   l_sql CLOB;
BEGIN
   l_sql := 'select object_name from all_objects
where object_id = ' || p_num;
   EXECUTE IMMEDIATE l_sql;
END;
```

- implicitna pretvorba number -> char:
  to_char (p_num)

Number

```
SQL> select to_number ('1.01', '9d99') from dual;

TO_NUMBER('1.01','9D99')
------------------------
                    1.01


SQL> alter session set nls_numeric_characters='P ';
SQL> select to_number ('1P01', '9d99') from dual;

TO_NUMBER('1P01','9D99')
------------------------
                    1P01

SQL> select to_number ('0P01', '9d99') from dual;

TO_NUMBER('0P01','9D99')
------------------------
                     P01
```

Number

```
CREATE OR REPLACE FUNCTION p01 RETURN NUMBER
AUTHID CURRENT_USER IS
BEGIN
   FOR x_rec IN (SELECT tname
                   FROM tab)
   LOOP
      dbms_output.put_line(x_rec.tname);
   END LOOP;
   RETURN (1);
END;
/

grant execute on p01 to public;
create public synonym p01 for scott.p01;
```

Number

```
SQL> exec inj (.01);

BIN$yn+TlSkTCtDgQIrBPS9M3w==$0
BONUS
DEPT
EMP
SALGRADE
TTT
USER_TABLE
X

'select object_name from all_objects where
object_id = ' || p_num;

select object_name from all_objects where
object_id = P01;
```

Kako se zaštititi?

- postoji teži i lakši put :-)
- provjeriti sav kod
- testirati različite mogućnosti unosa
- dobri standardi kodiranja
  - nikada ne koristiti implicitne konverzije
  - uvijek koristiti eksplicitne formate datuma

Kako se zaštititi?

- lakši put
  - koristiti vezane varijable
- vezane variable nisu podložne SQL upadu!

```
l_sql := '
  select *
    from all_users
   where created = :d';
open c for l_sql USING p_date;
```

There ain't no such thing as a free lunch!

http://en.wikipedia.org/wiki/There_ain't_no_such_thing_as_a_free_lunch

AS Abakus

As na disku.

Bind variables

```
CREATE OR REPLACE FUNCTION check_user (
   p_user  IN VARCHAR2,
   p_table IN VARCHAR2)
RETURN BOOLEAN IS
   l_ret NUMBER;
   l_sql VARCHAR2 (4000);
BEGIN
   -- we cannot use bind variable for table name!
   l_sql := 'SELECT COUNT (*) FROM '
      || p_table
      || ' WHERE USERNAME = :user'
   dbms_output.put_line (l_sql);
   EXECUTE IMMEDIATE l_sql
      INTO l_ret
      USING p_user;
   RETURN (l_ret != 0);
END;
```

Vezane spremenljivke

```
BEGIN
   IF NOT check_user (
      'STIPE', 'MY_USERS WHERE evil_funct() = :a1 --') THEN
      dbms_output.put_line ('Korisnik ne postoji!');
   END IF;
END;
/
```

kako spriječiti

- dostup do baza omogućiti samo putem PL/SQL API-ja
- ako je moguće izbjegavati korištenje dinamičnog SQL-a
- koristiti povezane varijable
- koristiti siguran SQL tekst

Abakus
As na disku.

PL/SQL API

- korisnik nema dostupa do tablica i/ili view-a
- upotreba privatnih sinonima (ako su potrebni - bolje »set current schema«)
- sinonimi smiju pokazivati samo na PL/SQL kod
- vlasnik PL/SQL koda je ujedno i vlasnik tablica/view-a, zato dodatne privilegije nisu potrebne

PL/SQL API - 2

- dodatna prednost: otpada upotreba triggera (sve se odvija preko PL/SQL paketa)
- prošireno pravilo
  - osigurati dostup do view-a
  - na view-ima koristiti INSTEAD OF triggere (dodatna prednost je što nema mutating tablice)

bind variables

- UVIJEK, osim:
  - dinamičkog određivanja imena Oracle objekta
  - prije 11g uglavnom u skladištima podataka
    - 11g: variable peeking
  - postavljanje parametara u session-u (»`alter session set optimizer_mode=all_rows`«)
    - u tim slučajevima se preporučuju konstante

```
c_all_rows constant varchar2 (60) :=
   'alter session set optimizer_rule=all_rows';
…
execute immediate c_all_rows;
```

bind variables - 2

- kada SQL naredbu dajemo kao parametar u neku od Oracle funkcija
  - dbms_utility.exec_ddl_statement()
  - dbms_ddl.create_wrapped()
  - dbms_hs_passthrough (SQL upiti u ne-Oracle bazama)
  - owa_util (generiranje HTML strani)

siguran SQL tekst

- kada se generira SQL naredba, upotrijebiti
  - dbms_assert.simple_sql_name()
  - dbms_assert.enquote_literal()
  - to_char (x f, 'NLS_NUMERIC_CHARACTERS='',.''')
    - x: variable of a numeric datatype
    - f: format model 'TM' (text minimum number format model)

DBMS_ASSERT

- dodana v 10.2, backport na 8.1.7 -->
- enquote_literal
- enquote_name
- SIMPLE_SQL_NAME
- QUALIFIED_SQL_NAME
- SCHEMA_NAME
- SQL_OBJECT_NAME

enquote_literal

- predani parametar postavlja jednostruke navodnike, ako već nisu postavljeni
- omogućuje da se jednostruki navodnici gnijezde
- ako nađe jedan jednostruki navodnik vraća grešku:
  ORA-06502: PL/SQL numeric or value error

enquote_literal

```
SQL> SELECT DBMS_ASSERT.enquote_literal('literal without quotes') FROM dual;

DBMS_ASSERT.ENQUOTE_LITERAL('LITERALWITHOUTQUOTES')
--------------------------------------------------
'literal without quotes'

1 row selected.

SQL> SELECT DBMS_ASSERT.enquote_literal('literal without ''''quotes') FROM dual;

DBMS_ASSERT.ENQUOTE_LITERAL('LITERALWITHOUT''''QUOTES')
------------------------------------------------------
'literal without ''quotes'

1 row selected.

SQL> SELECT DBMS_ASSERT.enquote_literal('literal without ''quotes') FROM dual;
SELECT DBMS_ASSERT.enquote_literal('literal without ''quotes') FROM dual
       *
ERROR at line 1:
ORA-06502: PL/SQL: numeric or value error
ORA-06512: at "SYS.DBMS_ASSERT", line 308
ORA-06512: at "SYS.DBMS_ASSERT", line 358

SQL>
```

enquote_name

- predani parametar postavlja dvoostruke navodnike, ako već nisu postavljeni
- omogućuje da se dvonostruki navodnici gnijezde
- zadano mijenja parametar u velika slova (to se može promijeniti s parametrom 'capitalize')
- ako nađe dvostruki navodnik vraća grešku: ORA-06502: PL/SQL numeric or value error

enquote_name

```
SQL> SELECT DBMS_ASSERT.enquote_name('quoted and uppercase') FROM dual;

DBMS_ASSERT.ENQUOTE_NAME('QUOTEDANDUPPERCASE')
----------------------------------------------
"QUOTED AND UPPERCASE"

SQL> SELECT DBMS_ASSERT.enquote_name('"remains quoted and lowercase"') FROM dual;

DBMS_ASSERT.ENQUOTE_NAME('"REMAINSQUOTEDANDLOWERCASE"')
------------------------------------------------------
"remains quoted and lowercase"

SQL> SELECT DBMS_ASSERT.enquote_name('pairs of ""quotes"" are allowed') FROM dual;

DBMS_ASSERT.ENQUOTE_NAME('PAIRSOF""QUOTES""AREALLOWED')
-------------------------------------------------------
"PAIRS OF ""QUOTES"" ARE ALLOWED"
```

enquote_name

```
SQL> SELECT DBMS_ASSERT.enquote_name('individual "quotes" are not allowed') FROM
dual;
SELECT DBMS_ASSERT.enquote_name('individual "quotes" are not allowed') FROM dual
       *
ERROR at line 1:
ORA-06502: PL/SQL: numeric or value error
ORA-06512: at "SYS.DBMS_ASSERT", line 308
ORA-06512: at "SYS.DBMS_ASSERT", line 343
ORA-06512: at line 1

SQL> SET SERVEROUTPUT ON
SQL> EXEC DBMS_OUTPUT.put_line(DBMS_ASSERT.enquote_name('quoted and remains
lowercase', FALSE));

"quoted and remains lowercase"

PL/SQL procedure successfully completed.

SQL>
```

simple_sql_name

- checks the input string conforms to the basic characteristics of a simple SQL name:
  - The first character of the name is alphabetic.
  - The name only contains alphanumeric characters or the "_", "$", "#"
  - Quoted names must be enclosed by double quotes and may contain any characters, including quotes provided they are represented by two quotes in a row ("").

simple_sql_name

- The function ignores leading and trailing white spaces are ignored
- The length of the input string is not validated.

- when input string does not conform, ORA-44003 is raised:

ORA-44003: Invalid SQL name

qualified_sql_name

- <local qualified name> ::= <simple name> {'.' <simple name>}
- <database link name> ::= <local qualified name> ['@' <connection string>]
- <connection string> ::= <simple name>
- <qualified name> ::= <local qualified name> ['@' <database link name>]
- [SCHEMA-NAME.]OBJECT-NAME[@DBLINK-NAME]

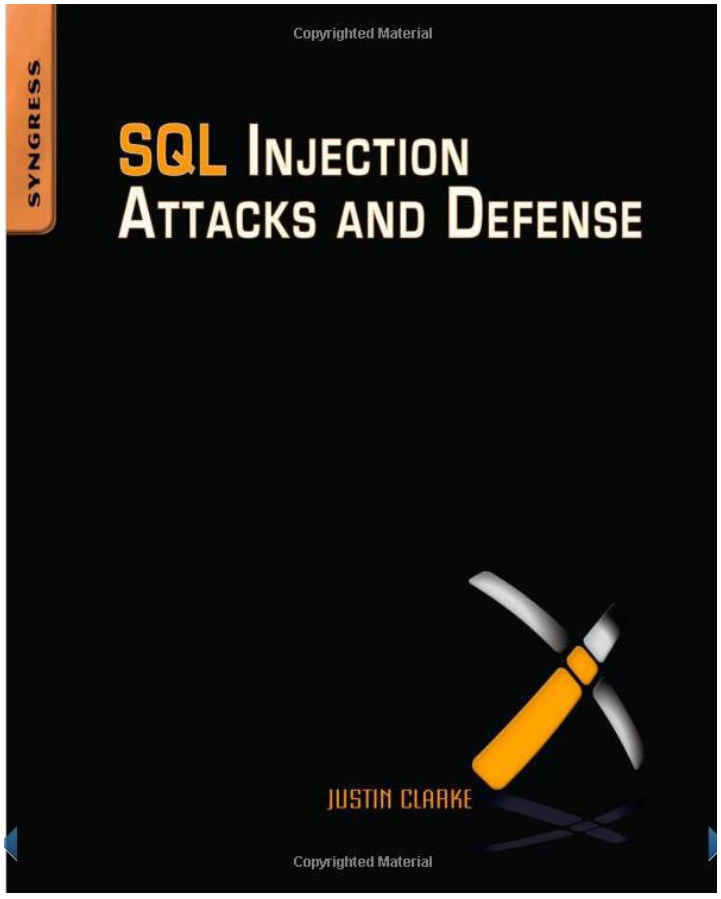
schema_name

- existing schema name
- case sensitive!

sql_object_name

- existing object, not case sensitive
  - 'dbms_assert'
  - 'sys.dbms_assert'
  - 'sys.dbms_assert@db_link'
- with db link only syntax is checked!

- How to write SQL injection proof PL/SQL
  (An Oracle White Paper)
  http://www.oracle.com/technetwork/database/features/plsql/overview/how-to-write-injection-proof-plsql-1-129572.pdf



SQL upadi predstavljaju najveću opasnost za SQL baze podataka jer ih je jako teško otkriti i spriječiti!

ORA-03113: end-of-file on communication channel

Boris Oblak
Abakus plus d.o.o.

ORACLE CERTIFIED PROFESSIONAL

ORACLE Gold Partner

SQL upad - krađa 130 milijuna kreditnih kartica