Security

12

What the reader will learn:

- The importance of database security
- Physical security considerations
- Database security risks and how to mitigate them
- Security issues in the cloud
- How to develop a Security Policy

12.1 Introduction

As mentioned in Chap. 2, data is one of organisations most important assets, therefore steps need to be taken to protect it. Security generally has three aspects to it: physical security, software security and procedures. In each case precautions need to be in place along with a risk assessment and a recovery plan. Security is also closely linked with availability which was discussed in Chap. 11. The simple difference is that availability refers to making sure the systems don't shut down, whereas security is keeping the 'bad guys' out. However if you have a security breach it may mean your database is not available, or your data has been corrupted.

In the case of physical security, the main question to ask is how to protect the servers and infrastructure from damage or loss. This may range from a catastrophic event in the data centre through to someone stealing the actual server. Physical security is not only an issue of restricting access to a computer centre, but how to recover operations if there is a major incident. In other words it is closely linked to the question of availability.

A more recent physical security threat has emerged from the increase in use of mobile devices. These include smartphones, laptop and tablet computers as well as removal storage devices such as pen drives. Loss and theft of these devices which may contain sensitive information or have the ability to automatically connect to sensitive systems has been a growing problem for some time.

Although physical security is important, software security is the more important security consideration on a day to day basis. Many databases have internet access to them, although this is usually through another server. It is however where most

P. Lake, P. Crowther, *Concise Guide to Databases*,
Undergraduate Topics in Computer Science, DOI 10.1007/978-1-4471-5601-7_12,

threats come from. The nature of these threats and what can be done to mitigate them will be discussed in this chapter.

The final aspect is the procedures which are in place. These will also have an impact on how physical and software security is implemented and maintained. Questions to ask are what procedures are in place, are they adequate and how are these audited. A major component of this is a risk register which needs to be regularly updated and reviewed.

12.2 Physical Security

Physical security is probably the easiest aspect of security to deal with, but it is potentially the most catastrophic when things go wrong.

A review by Scalet (2009) gives a comprehensive 19 point guide to physical security including building a secure centre capable of surviving explosions. A lot of this is in response to the aftermath of the terrorist attacks of 9/11 and much of it is beyond the capability of an average company. However, as the following Buncefield example shows, there needs to be a contingency plan as to what to do if the processing centre is destroyed. On 11th December 2005 as a result of an industrial accident there was an explosion at the Buncefield oil storage terminal at Hemel Hempstead just north of London in the UK. The resultant fire was still burning on the 14th December. There were a number of businesses in the adjacent business park which were also damaged or destroyed. One of these was Northgate Information Solutions whose headquarters were badly damaged by a wall being blown out, it was rendered completely unusable.

Northfield hosted systems for other organisations including the payroll system for the author's university. As a result of the explosion it was not possible to process the payroll for the next payday (18 December and just before Christmas). Although the company had other processing sites it was impossible to completely set the system up in time for the next payday. As a result a stop gap measure where staff were paid the same as they were paid the previous month was instituted. The system was backed up at another site in time for the January 2006 payday.

This illustrates the need of having a contingency plan for when something does go wrong. It is extremely rare that a data centre gets completely destroyed, but floods and fire do happen as the example shows. One question that needs to be asked is how much an organisation is willing to spend on disaster proofing physical infrastructure—it is probably impossible to make a building totally indestructible. A better question is how will operations continue if it does happen and how will the database be recovered.

The other component of physical security is access control where physical access to sensitive hardware and parts of a building are restricted. A 3 line security system consisting of something you are, something you know and something you have gives the best level of security for access to both building space and access to systems:

Something you are: iris recognition, fingerprint recognition
Something you know: password or code

Something you have: passkey or other device
(see Federal Financial Institutions Examination Council *Authentication in an Internet Banking Environment* http://www.ffiec.gov/pdf/authentication_guidance.pdf accessed 28/07/2013).

Oracle's Security Guide for Oracle 10g (Oracle Corp 2012a, 2012b) gives a physical and personnel control checklist. At its most basic, the policy should be that no one should be able to walk into a data centre without proper authorisation and identification (which is checked). It should be noted that although this list is not included in the Oracle 11g security guide it is still valid. Scalet (2009) suggests that there should be a minimum of 3 separate checks of personnel—on entering the building, on entering the employee area and on entering the data centre 'core'. She further points out that access to the core should be strictly controlled and on a needs only basis.

Oracle adds that the elaborate measures documented by Scalet are probably not needed by the majority of organisations. Factors such as organisation size, risk of loss, access controls and frequency of external visitors will determine the level of measures needed. They also add that the visibility of security measures such as video surveillance will act as a deterrent. Oracle say: 'Make it difficult to get in, difficult to remain or leave unobserved or unidentified, difficult to get at sensitive or secure areas inside, and difficult not to leave a trace.' (Oracle® Database Security Guide 10g Release 2 (10.2) B14266-09 p 2-2). Even a small company should restrict access to their server, even if that means keeping it locked in a cupboard.

Shinder (2007) in her article lists 10 physical security measures every organization should take which add a few more concerns to the list we have already seen. Workstations, particularly any which are left logged on in an unattended are a potential security risk. The machine at the front receptionist's desk can be a particular risk. Related to this is the physical removal of workstation components such as the hard disk drive. A case lock is a low cost solution to this problem.

A classic television and cinema scenario is the good (or bad) guy breaks into an office and downloads the information on a workstations hard drive to a memory sick or pen drive. In practice it is more likely an employee who is going to do this—often not for malicious intent (it may be they want to work on data at home). It is however a security risk and many organisations take measures to prevent unauthorised downloading by disabling USB ports and any device capable or writing to removal media.

A final, often forgotten risk in physical security is disposal of equipment. It is essential hard drives are properly erased. This is more than just deleting files because in most cases only directories which point to data are deleted. The actual data often remains and there have been cases of machines being bought at auction and their hard drives being reconstructed. In some cases even a simple delete had not been carried out. Free utilities such as Active@Kill Disk—Hard Drive Eraser from LSoft Technologies (see http://www.killdisk.com accessed 30/07/2013) exist for overwriting disks. Disks can also be cleared magnetically—a process known as degaussing.

12.3 Software Security—Threats

On line access security is the area which is arguably of greater concern to businesses and potentially a much more of a serious risk than the physical securing of a database. Once a building is physically secure, assuming there are no lapses in protocol, the issue is solved. Problems associated with software tend to be more on going and continually evolving, particularly unauthorised access and changes to data.

Therefore this type of security includes access to software systems (rather than physical access to hardware discussed earlier) and data integrity measures. Data integrity is ensuring the accuracy, reliability and consistency of data in a database. There also needs to be a decision as to where security protocols are enforced. It is not normal for an internet user to directly interact with a database, there is usually some middleware handling security and checking transactions before a database is updated. For example, some banks require up to three security questions to be answered before a user is allowed access to on-line banking. The Yorkshire Bank requires a user ID, then a password followed by a security question. The password itself is not entered but randomly selected characters from it are requested. This is also done when purchases are made on-line. Only when these security measures have been passed is the user (in this case a customer) able to carry out on-line financial transactions.

There are number of types of threats to databases: privilege abuse, platform weaknesses, SQL injection, weak audit, protocol vulnerabilities, authentication vulnerabilities, backup data exposure and mobile device based threats, each of which we will review below.

12.4 Privilege Abuse

Users of a database will require different levels of access known as privileges. For example, a database administrator will need to be able to modify the database definition using data definition language and perform database management tasks such as brining the server down or starting it. At the other end of the spectrum there will be users who should only be able to view a restricted subset of the data in the database. Between these extremes there will be users who will need various levels of access to manipulate data to fulfil their roles.

Database Privileges can be arranged in the following way with progressive capabilities to make changes to the database. They are arranged by user: those who are concerned with data and administrator: those who are concerned with management of the databases structure:

User

Read—the basic level most users require to access data. Modification is not allowed

Insert—add new data, but no modification of existing data. This may not automatically give read privilege

Update—modify, but not delete data
Delete—ability to delete a complete record

Administrator

References—an extension to update where foreign keys can be created
Index—create or drop indexes
Create—ability to create new tables
Create a new database
Modify—add or delete attributes
Drop—delete a table
Grant—ability to grant privileges including create new users Start and stop a database

The way these privileges are maintained depends on the system. For example, on IBM systems these privileges are held in an access control list (ACL). Privilege abuse occurs when users or administrators are given privileges which exceed their requirements and then proceed to exploit these 'excess' privileges.

To mitigate against this type of problem the ACL or its equivalent needs to be developed and maintained. A precursor to this is that the role of the user must be considered and only the minimum rights should be granted to enable the user to carry out their roll. This is not a trivial task in terms of its creation and maintenance. IBM divides the ACL into three parts:

- Access
- User
- Privilege

Basically the first two determine the third. In terms of access, IBM visualises the hierarchy as an inverted pyramid with Manager at the top and 'No Access' at the bottom.

MANAGER	Can access everything
DESIGNER	Can modify all design elements
EDITOR	Can create documents and edit all documents
AUTHOR	Can create documents and edit those documents they have created
READER	Can only read documents, but not create or modify documents
DEPOSITOR	Can only create documents, but not read those documents
NO ACCESS	No access to anything

Adapted form 'The ABC's of using the ACL' at http://www.ibm.com/developerworks/lotus/library/ls-Using_the_ACL/#N10085 accessed 04/06/2013.

No Access, Reader and Depositor access is fairly straight forward, but when you get to Author there are a number of options available. For example, you might be designated an author of a document, but you might not have a 'delete' right for the document (even though you created it). The same is true for Editor access.

As well as access, IBM considers five user types.

PERSON	You as an individual
PERSON GROUP	List of names belonging to a group
SERVER	Individual server
SERVER GROUP	Servers in a group
MIXED GROUP	Combination of server and person groups.

The user types are a means of assigning an ID for accessing the database and all vendors have an equivalent system although they vary widely in implementation. An individual ID will not give a user access to a database that requires a group ID. Likewise if a server or server group is specified, a user will not be able to access the database unless they use specified server. The final option only requires you to be a member of a group and you can be either a server or a client.

These settings allow a database administrator to have a structured approach to privileges. It should also be noted that a user may be another server rather than a human. Once created the ACL needs to be maintained. Users roles change and when this happens their privileges must be reviewed.

Even when you have established privileges and keep the ACL up to date, there can still be an issue with legitimate privilege abuse where legitimate access privileges are used for unauthorised purposes. Examples of this kind of abuse include:

- Accessing confidential data unnecessarily. There was a case where a tax official was viewing a celebrities records for no other reason than they were a fan.
- Retrieving large quantities of data for no legitimate reason causing performance degradation.
- Downloading data and storing it locally. This often happens when someone sets up their own bespoke system—typically on a spreadsheet. The issue is compounded if it is sensitive data and it gets lost. How many times have you read about laptops and pen drives with sensitive information on them getting lost or stolen?

This kind of abuse is more difficult to control but could be aided by more granular ACL policies and firm personnel management.

The final type of privilege abuse is privilege elevation where a user who has legitimate access exploits a vulnerability to gain administrative privileges. This will be discussed further in the next section on platform vulnerabilities.

As already discussed, the primary way of dealing with privilege abuse is through a ACL or an equivalent and we have seen the philosophy behind IBM's approach. However different vendors set this up in different ways. For example Oracle does not have an ACL but has privileges grouped into roles which are placed in a hierarchy. There are 52 predefined user roles in Oracle 11g and new roles can be created, assuming you have the privileges to do it. Some roles like **sys** also need to have root privileges for the operating system.

SQL Server on the other hand does not have a dedicated ACL but instead has a series of operations which can be applied to define privileges and give the same resource as an ACL. This is held as a hierarchy with the top level user being iden-

tified as the principle. Below that there are the operating system, SQL Server and database level permissions.

The other way to restrict access to data is via views. A view is in effect a virtual table which may allow only certain attributes to be viewed and manipulate. Views can include joins so the virtual table a user is working with is made up of columns from two or more tables. As well as restricting access to data it makes complex queries easy by hiding the original (complex) SQL query. It also allows the same data to be presented in different ways to meet user requirements.

In ORACLE and SQL Server a simple view can be created by:

```
CREATE VIEW phone_contact_vu
    AS SELECT customer_name, customer_phone, customer_mobile
    FROM customer;
```

This would create a view to retrieve telephone details ignoring other columns such as address details. A WHERE clause can be added to further restrict data:

```
CREATE VIEW phone_contact_vu
    AS SELECT customer_name, customer_phone, customer_mobile
    FROM customer
    WHERE customer_city = 'Sheffield';
```

Once the view is created it can be used to retrieve data as if it was a table, for example:

```
SELECT * FROM phone_contact_vu;
```

You can add restrictions, for example if this was to be used only for reference and the user did not have permission to change details:

```
CREATE VIEW phone_contact_vu
    AS SELECT customer_name, customer_phone, customer_mobile
    FROM customer
    WHERE customer_city = 'Sheffield'
    WITH READ ONLY;
```

Any attempt to alter the table with either a DELETE FROM, INSERT or UPDATE statement would result in an error. An alternative to the **READ ONLY** clause is **WITH CHECK OPTION** where any DELETE, INSERT or UPDATE must conform to the definition of the view. So if you created the view with:

```
CREATE VIEW phone_contact_vu
    AS SELECT customer_name, customer_phone, customer_mobile
    FROM customer
    WHERE customer_city = 'Sheffield'
    WITH CHECK OPTION;
```

and then tried an UPDATE of the form:

```
UPDATE phone_contact_vu SET customer_city = 'Leeds';
```

you would get an error because the view is defined such that the customer city can only be Sheffield.

A complex view has more than one table in its definition, for example:

```
CREATE VIEW outstading_invoice_vu
    AS SELECT customer_name, invoice_id, invoice_date
    FROM customer
    INNER JOIN invoice
        ON customer.customer_id = invoice.customer_id
    WHERE invoice_status <> 'paid'
    ORDER BY invoice_date;
```

which can be used to retrieve customers who have an outstanding invoice. One problem with complex views in ORACLE is that you cannot perform data manipulation with them. To achieve this requires a procedure or trigger needs to be written.

Not all mistakes in data entry are deliberate, so integrity checks should be set up which are enforced by the database management system. These include check constraints to limit the chances of wrong data being entered. For example to add a constraint:

```
ALTER TABLE customer
ADD CONSTRAINT customer_city_ck
    CHECK (customer_city IN ('Sheffield', 'Rotherham', 'Doncaster'));
```

In the above example it should be noted that each of the city names has to be in the format described. 'SHEFFIELD', for example would generate an error. A better way to limit invalid data entry is to use lookup tables so the user has to choose from a list rather than entering raw data. This would also reduce the possibility of mistyping data on entry. To implement this would require a simple program known as a procedure or trigger to be written.

Where possible, rules to trap transcription and transposition errors should always be implemented. A typical example of a transcription error is entering a zero in place of the character 'O'. Transposition errors are harder to trap, for example entering 547619 instead of 546719.

Some database management systems such as those following CODASYL conventions use a concept of subschemas instead of views. A subschema is part of the database that a user can access and manipulate. This was quite common in early mainframe databases with hierarchical and network structures which were described in Chap. 2.

Ultimately the guiding principle for privileges is to only give a user the bare minimum they require to be able to complete their tasks. When a new user is added to the system unnecessary privileges (often added as defaults) should be removed. If a user's responsibilities change, their privileges should be reviewed.

12.5 Platform Weaknesses

Unfortunately most products and their underlying operating systems and services have weaknesses and unresolved issues. That is why regular patches are issued by vendors. These have two roles. The first and of most concern here are to fix reported security weaknesses and bugs in a product. The second is to improve usability and performance. It is always a good idea to install these as soon as possible otherwise your system could be vulnerable.

An example of what can happen was posted on 14th January 2013 in Yahoo news (http://news.yahoo.com/oracle-says-java-fixed-feds-maintain-warning-230702904–finance.html)

> "Oracle Corp. said Monday it has released a fix for the flaw in its Java software that raised an alarm from the U.S. Department of Homeland Security last week. Even after the patch was issued, the federal agency continued to recommend that users disable Java in their Web browsers.
> "This and previous Java vulnerabilities have been widely targeted by attackers, and new Java vulnerabilities are likely to be discovered," DHS said Monday in an updated alert published on the website of its Computer Emergency Readiness Team. "To defend against this and future Java vulnerabilities, consider disabling Java in Web browsers until adequate updates are available."
> The alert follows on the department's warning late Thursday. Java allows programs to run within websites and powers some advertising networks. Users who disable Java may not be able to see portions of websites that display real-time data such as stock prices, graphical menus, weather updates and ads. Vulnerability in the latest version, Java 7, was "being actively exploited," the department said. Java 7 was released in 2011. Oracle said installing its "Update 11" will fix the problem.
> Security experts said that special code to take advantage of the weakness is being sold on the black market through so-called "Web exploit packs" to Internet abusers who can use it to steal credit card data, personal information or cause other harm. The packs, sold for upwards of $1,500 apiece, make complex hacker codes available to relative amateurs. This particular flaw even enables hackers to compromise legitimate websites by taking over ad networks. The result: users are redirected to malicious sites where damaging software can be loaded onto their computers."

Oracle issue regular security alerts (http://www.oracle.com/technetwork/topics/security/) which includes critical patch updates. Likewise Microsoft issues SQL Server alerts (see http://support.microsoft.com/kb/959420).

It is beyond the scope of this book to discuss intrusion and virus threats in detail. New viruses are released constantly so updates to anti-virus software are released almost daily and sometimes more often. Likewise attempts to gain unauthorized access to systems are getting more sophisticated. It goes without saying you should have firewall and virus detection software in place that is regularly updated. There are both free (for example Avast, www.avast.com, last accessed 30/07/2013) and commercial (for example McAfee, www.mcafeeprotection.com, last accessed 30/07/2013) solutions readily available.

12.6 SQL Injection

In some ways this is an extension to the previous topic on system vulnerabilities, but it is a common way of compromising databases. It refers to the insertion of SQL statements in a data entry field. It is the most common type of attack on web connected databases. If the attack is successful it can result in access to unauthorized data and manipulation of that data or privilege elevation which we have already discussed under privilege abuse. SQL injection comes in four main forms: SQL manipulation, code injection, function call injection and buffer overflow.

SQL manipulation is the most common form of SQL injection and involves an attacker attempting to modify existing SQL by adding elements to the **WHERE** clause and seeing if any records were returned. If the original statement was:

```
SELECT * FROM users
WHERE username = 'Dorian' AND password = '&mypassword';
```

and the attacker added an **OR** clause so it became:

```
SELECT * FROM users
WHERE username = 'Dorian' AND password = '&mypassword' OR 'a' = 'a';
```

then the **WHERE** clause would always be true because of operator precedence and the attacker would have gained access to the application.

Another variation is to use the **UNION** operator to try and return rows from another table, so if the original query was:

```
SELECT item_description FROM stock_item
    WHERE price <10;
```

the attacker might attempt:

```
SELECT item_description FROM stock_item
    WHERE price <10
    UNION
    SELECT username FROM admin_user
        WHERE username like '%';
```

which returns not only a list of items but also all database administration users.

Code injection tends to be more of a problem with Microsoft SQL Server and PostgreSQL (Kost 2007) because it involves adding an extra SQL statement to an existing statement. Oracle does not support multiple SQL statements per database request so it is afforded some protection. However, underlying platform weaknesses as already discussed may allow a java program in a web application to execute multiple requests.

Function call injection is a variation on SQL manipulation where a database function (either a built in one or a custom written one) is added to vulnerable SQL code. Oracle supplies some functions which perform network communication which could be exploited as well as over a thousand other functions which would have no use in an attack.

Buffer overflow occurs when a program writes data to an area of memory and overwrites adjacent memory. A buffer is a temporary area in memory storage space where data is kept prior to writing to another device. In the case of databases, that device is disk storage. Some Oracle functions are susceptible to buffer overflows and this can be exploited through a SQL injection attack.

It is easy to protect against SQL injection attacks so it ultimately comes down to enforcing a series of methods to every web accessible procedure and function as even one unprotected SQL statement could cause problems. The following are a minimum:

- Always use bind variables (substitution variables that are used in place of literals). In any SQL statement where a variable is required, for example in a select statement, the variable should have been declared as a bind variable. It also improves system performance because statements are reused rather than reparsed at every execution.
- Always validate every passed string parameter.
- Restrict all functions that are not absolutely necessary

12.7 Weak Audit

Database auditing is at the core of any database security policy. Database systems generate an audit log that captures a record of all data access and alternations. This log is managed internally by the database management system and is the most accurate source of monitoring activity. However, there is a possibility of issues arising if an organisation bases all of its audit policies on the built in database mechanisms. The main problem here is that the system will generate a large amount of data as each transaction is logged. The sheer volume of data requires knowledge of what to look for to make sure there have been no security violations, for example privilege elevation. There are five things that should be monitored for suspicious activity:

- Who accessed which systems when and how. If there is evidence of failed logins there is a high probability someone is trying to break into your system. This is particularly important when you consider most users do not directly log in to a database but go through other applications which automatically connect.
- User and administrator activity should be examined. Failed queries should be taken seriously as they may be an indication of an attacker probing known vulnerabilities in a system including assuming specific features or structures will be present. The error details which are returned when a query fails should be logged rather than displayed to the user as they may use the details to refine their attack. Even if the failed query is not a result of an attack it should still be checked because it may be an indication of a flaw in a script which in turn could lead to application failure.
- Abnormal activity such as unusual access to sensitive data should be logged and investigated. This may require identifying what is sensitive data then applying filters to audit logs to isolate activity on such data.

- All databases should be audited for vulnerabilities and threats. This relates back to platform weaknesses. If you know about weaknesses, you can monitor activity attempting to exploit them.
- Changes to metadata should be monitored. Changes to the databases structure can lead to new vulnerabilities. It is essential to establish a baseline policy for monitoring changes to database configuration. If there are any deviations from that baseline, they should be tracked back to establish their source.

Recording everything in an audit log provides lots of useful information, but there is an impact on performance. Therefore it is necessary to understand the audit settings on your system and implement them in a way that will capture critical information without overly degrading performance. Oracle allows generation of '. . . audit records that include information about the operation that was audited, the user performing the operation, and the date and time of the operation. Audit records can be stored in the database audit trail or in files on the operating system. Standard auditing includes operations on privileges, schemas, objects, and statements.' (Oracle, '12c Traditional Database Auditing', http://www.oracle.com/technetwork/database/security/index-085803.html, accessed 28/07/2013). Oracle suggests that the audit records should be stored on the operating system as this has the least overhead compared to storing it on the host database system. It should also be noted that every write to the audit files is a system overhead that will reduced system performance. This has to be balance with the need to recover a system or look for evidence of security breaches.

12.8 Protocol Vulnerabilities

A communication protocol defines the methods of exchanging messages between computer systems and there can be weaknesses. As an example, in 2003 the SQL Slammer Worm exploited a weakness in Microsoft SQL Server and Desktop Server protocol. It was small and resident in memory. Its effect was to generate random IP addresses and send itself to those addresses. It should be noted the worm did not use SQL code, but exploited a buffer overflow problem. It was used to execute denial of service attacks severely compromising performance. Interestingly a patch was available to guard against the problem six months before the worms launch, but many systems did not have it installed.

Although this example features SQL Server, communication protocol vulnerabilities have been identified for every vendor's products. Basically a communication protocol is how components of a system communicate and if a component can be impersonated, then there is a security issue.

It is therefore essential that protocol validation is enabled. This technology breaks down database traffic, a process known as parsing, and looks for anomalies. Only normal client generated messages should be allowed through with requests using hidden features blocked. Obviously know attacks (like the SQL Slammer Worm) should be checked for which means implementing security patches as they become available.

12.9 Authentication Vulnerabilities

What you are trying to defend against here is someone guessing account names and passwords. In a large English language western organisation the chances of there being a 'John Smith' with legitimate access rights is quite large. Some organisations have a standard which is used for both e-mail addresses and userid's for example, J.Smith@somecompany.org and then J.Smith as a userid. Combine this with a predictable choice of password ('password' is the most common password) and you have access. Unfortunately many people use the same password for multiple systems, so once you have access to one system, you may have access to multiple systems. This type of knowledge is used in so called 'brute force' attacks were attacking systems use common names then a list of common passwords in an attempt to break in.

Measures to mitigate these problems include using a different standard for e-mail addresses and userids with the userids avoiding real names. There should be a strong password policy. The authors' institution requires passwords to be changed on a regular basis, the password to be at least eight characters long, the password to contain a mixture of alphabetic and other characters and the password not to have been used recently. The authors' bank doesn't enforce a password change but requires random characters from it to be entered along with a 'secret' word randomly selected from a list provided by the user. Most systems now disable an account if three unsuccessful attempts are made to log in.

Even with these authentication measures in place, unsuccessful access attempts should be logged and investigated, particularly if there is a pattern of related attacks.

12.10 Backup Data Exposure

Backup and recovery are topics dealt with in the chapter on availability. However it is worth emphasising that databases and in fact any data should be regularly backed up. Most database management systems supply utilities to do this. However there have been many incidents where backup media has been lost or stolen. This has become an escalating problem with the proliferation of mobile devices and memory sticks. Often the data on these is not encrypted meaning there is a potential for large amounts of sensitive data to be compromised.

Encryption of data won't stop devices being lost, but it will keep the data secure. There are however costs, the most noteworthy being performance degradation. Encryption is also often application dependent sometimes with complex key management.

To mitigate backup data exposure, some organisations have banned the use of none encrypted devices to transport data. The authors organisation requires staff transporting sensitive data on memory sticks to use IronKey devices. IronKey is a proprietary device with built in encryption. It can be configured to delete all data stored on it if the maximum number of failed password entry attempts is exceeded.

12.11 Mobile Device Based Threats

Most of the threats mentioned above also apply to mobile devices, especially backup data exposure. Loss or theft of an unsecured mobile device is a major concern. For example, loss and theft of mobile devices cost the BBC over £750,000 between 2010 and 2012 (ComputerWorld UK 2013). This was just the value of the hardware and did not include the cost of data loss and security exposure related to stored passwords. Many mobile devices require a user to activate password security, the default being it is off meaning if this was the case with the BBC devices there was a major security breach. As already mentioned many systems including Microsoft Windows will offer to remember passwords. That should always be declined and password security to the device should be enabled.

There is another aspect which relates to distributed databases. Most mobile databases are distributed among wired components which are accessed by mobile devices rather than being held on the devices themselves. This is the thin client approach where the mobile device is the thin client and all the processing is done on the host system. Following the rules of never let your device store the passwords and always activate the password security on the device, this should be secure. But in a second scenario data is distributed among both wired and wireless components with active data being distributed to the mobile device. Data management involves data fragmented between mobile devices and fixed servers or base stations. In this scenario base stations and mobile devices must all be secure.

Something that can be regarded either as a threat or a security tool is a wireless sniffer. This is a form of packet analyser. Data is transmitted in packets and this tool decodes the packet's raw data, showing the values of various fields in the packet, and analyses its content. On the positive side it can be used to analyse network problems and detect network intrusion attempts and network misuse. However wireless sniffers can also be used to gain information for executing a network intrusion and to spy on network users and gather sensitive information. To guard against these threats router security should be activated and data should be encrypted.

12.12 Security Issues in Cloud Based Databases

Cloud computing is a technology that gained prominence in 2006 when Amazon introduced the Elastic Compute Cloud (EC2) and was at the peak of Gartner's hype cycle for emerging technologies from 2009 onwards (see Chap. 2 for a description of the Gartner Hype Cycle). Since 2012, Gartner published an annual snap shot of the hype cycle specifically for cloud computing. One aspect of this is using the cloud as a database service platform.

The idea of cloud computing is that an organisation does not have to store its own data, or even applications. Instead they are stored on a provider's servers and accessed via the internet. This has many advantages for a user but introduces some new security issues and threats.

Because a user is no longer directly responsible for database security, there is the possibility of failure to provide adequate security by the provider. The provider controls the servers, network and security regime therefore the user must trust the provider's security. It follows that when choosing a cloud provider an organisation should verify the providers security measures before signing a contract with them and to monitor the provider's security performance. It should be noted however that a cloud provider often has security measures that far exceed those of most small to medium sized enterprises.

A second issue is attacks by other users. This process was documented by Ristenpart et al. (2009) where a method of stealing data from Amazons cloud was demonstrated. It was based on the fact that if you hired multiple virtual machines from Amazons cloud, they had similar IP addresses. If you knew an application was cloud based, an attacker could bombard a victim site with requests forcing the target organisation to hire more virtual machines. At the same time the attacker would hire more virtual machines themselves. It was found that 40 % of the time the attackers and victims ended up on the same server where the attacker could monitor the victim's virtual machine. Although they didn't actually steal any data, the authors said outright data theft was possible. It comes about because provider resources are shared with other parties, therefore the provider must take measures to separate users data and applications. Amazon has since closed this security loophole.

A related issue is data sanitization. Cloud computing is dynamic with data constantly being moved as demand on servers fluctuates. When data is moved, the physical location where the data is moved from needs to be overwritten rather than just being flagged as 'free'. This differs from an in-house application where pointers to data are removed and the disk space is flagged as being available rather than data being physically removed. In a cloud environment this is not sufficient as other clients of the cloud provider may (and probably will) use the space and could potentially see residual data. On in-house systems not physically deleting data has the advantage that if the 'accidental' delete is recognised soon enough the data can be 'un-deleted'. In a cloud environment with multiple organisational users, it is a security risk.

An area of vulnerability for data in the cloud is during data transfer. Standard communication protocols and procedures such as Hypertext Transfer Protocol Secure (HTTPS) and Secure Shell (SSH) should be implemented. The data should be encrypted to guard against data sanitisation failure.

Although not directly a security threat, there is an issue with regulatory compliance. For example if an organisation in the United Kingdom is using cloud resources and the servers supplying that resource are in the United States, then the resource is subject to US law and regulation—in other words location matters. This has resulted in many organisations adopting a policy of only non-critical systems with data that is not sensitive being deployed in the cloud.

A lot of the security issues associated with databases in the cloud come down to the service provider. Before embarking on a project to move data to the cloud the provider should be thoroughly vetted. Questions to ask related to the issues raised in this section to provide an idea of risk should include:

- What is the providers security policy including segregation of users?

- What is the providers' backup and recovery policy?
- What are the providers' encryption procedures?
- Where is the physical location of the data (although this could be several locations)?
- Conduct a risk assessment of the providers viability (how likely are they to go bust or get taken over?)

12.13 Policies and Procedures

Unfortunately the greatest online security threat comes from within an organisation from personnel who have legitimate access to a database. We have already looked at some of the issues here in the section on privilege abuse. To reduce the internal threat to an organisations database system there needs to be policies and procedures in place which are enforced and monitored:

Personnel This comes down to good personnel management and monitoring. All personnel should have a profile detailing what access level they need to carry out their duties. This should be set at a minimum and relates to both physical access to parts of the building and system access. Any attempts, whether deliberate or accidental to circumvent this must be followed up. In the event of an employee leaving, their privileges and access must be immediately revoked. This often means paying out an employee's notice period rather than letting them remain on the organisations premises. If an employee is fired or disgruntled, this becomes even more critical.

Software Upgrades These should be carried out as soon as practical, particularly security upgrades relating to firewalls, viruses and system vulnerabilities. This needs to be monitored to make sure it is happening. If you have deployed a database in the cloud you also need to check the provider has those policies in place and is adhering to them.

Building Maintenance There is little point having CCTV surveillance if the cameras or recording devices are not working. The same goes for access controls, for example secure doors should never be wedged open, even during cleaning operations. If any part of the physical security of the organisations infrastructure develops a fault it should be fixed immediately.

12.14 A Security Checklist

The following is based on Oracle's checklist, but is generalised to include other platforms. It details software and configuration security rather than physical security of a system.

- Disable default user accounts. Most database management systems come with preconfigured user accounts—Oracle has some 20 of these. They are a common first place an attempt to break into a system will occur. Other proprietary systems may have a 'guest' user or other default accounts.

- Change default passwords. This is a trivial issue, but one that can leave a system vulnerable if not implemented. It is crucial for administrative accounts. This is important for any system—not just databases (how many of you still have a default password set on your broadband hub?). Passwords should also be changed regularly. If your system does not have a feature to enforce this, develop and enforce a procedure for your users.
- Protect the important resource of your database data dictionary. This stores the name, meaning, relationships to other data, usage, and format of all components of your database. Protection must therefore be enabled and access restricted to those with DBA roles.
- Give users the lowest level of privilege possible. This means granting only necessary privileges, but also revoking unnecessary privileges if a user's role changes. Monitor employees roles and review and update privileges if their roles change.
- Create and maintain a Access Control List or its equivalent for your system.
- Authenticate client systems properly—in other words there needs to be stringent authentication of any client system trying to access the database.
- Protect any database gateway to the network. Particularly do not let any gateway software read or write files in the database or sever address space. This is another favoured way of attacking systems via sql injection.
- Monitor who accesses your system. This should be done at the user rather than server level as servers may be compromised.
- Apply all security patches as soon as they are available. In most cases there is an automatic update, but it is worth regularly checking your vendors web site to check for security alerts and patches. Related to this is to inform the vendor if you become aware of a vulnerability so a patch can be developed.
- If you are using a web interface to your database consider including a CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) test where a string of characters are presented in a deformed way and the user is asked to enter them. This stops attacks where a computer program attempts to access a website. Recent reports suggest that ways are being found to negate this test.
- Enforce a minimum access security regime of a userid and a password. Make sure there is a password policy which vets passwords to make sure they are not easily guessable. In a database environment there needs to be several levels of security so you may wish to include further security checks.
- If you are using a cloud service provider for your database verify that they have a security checklist and enforce it.

12.15 Review Questions

The answers to these questions can be found in the text of this chapter.

- What are 8 types of threats to database security?
- What are the three basic things you should have for physical access to a secure part of a building?

- What is privilege abuse and what measures can be taken to mitigate it?
- Why are mobile devices a particular security threat and what measures can be taken to reduce that threat?
- What is an SQL injection attack?

12.16 Group Work Research Activities

These activities require you to research beyond the contents of the book and can be tackled individually or as a discussion group.

Activity 1 Create a table of four columns. In the first list the devices you own or have administrative rights to. Don't forget mobile devices such as phones and tablets. In the second column place a tick if you have changed the default password (and remember, the default may be no password). In the third column place an X for devices you have the same password for. If you have several passwords you use regularly, use other letters of the alphabet. Leave a blank if the password is unique. In the fourth column write the date of the last time you changed the password. If you have never changed it, put an X there.

You have a security issue if there is no tick in column two, anything in column three and an X or a date older than 30 days in column four.

If you have no security risks identified congratulations!

If you have any security issues identified take steps to rectify them including a schedule to change passwords. You should also ask yourself if the password is easily guessable (like 'password')

You might consider repeating this exercise for online services (including bank accounts) and fix them as well.

Did you remember to include your wireless hub?

Activity 2 This exercise deals with security software.

What virus checking and firewall software do you run?

Does it automatically download and install updates?

When did it last install an update?

When was the last time you did a full system check? It is possible that you had a virus last time you did a full scan that wasn't detected by what was then the most up-to-date version.

Is your product licence current?

Do you subscribe to a service that regularly gives news of security threats?

If any of your answers are 'none', 'don't know' or 'no' you need to take urgent action as your system may be exposed.

References

ComputerWorld UK (2013) Loss and theft of mobile devices costs the BBC over £750,000 in three years. http://www.computerworlduk.com/news/mobile-wireless/3441452/loss-and-theft-of-mobile-devices-costs-bbc-over-750000-in-three-years/. Accessed 28/07/2013

Kost S (2007) An introduction to SQL injection attacks for Oracle developers. Integrity white paper. http://www.integrigy.com/files/Integrigy_Oracle_SQL_Injection_Attacks.pdf. Accessed 30/07/2013

Oracle Corp (2012a) Oracle® database security guide 10g Release 2 (10.2) B14266-09. Available on line at docs.oracle.com/cd/B19306_01/network.102/b14266.pdf. Accessed 02/05/2013

Oracle Corp (2012b) Oracle® database security guide 11g Release 1 (11.1) B28531-19. Available on line at http://docs.oracle.com/cd/B28359_01/network.111/b28531.pdf. Accessed 28/07/2013

Ristenpart T, Tromer E, Shacham H, Savage S (2009) Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In: Proceedings of the 16th ACM conference on computer and communications security, pp 199–212. Available at http://www.cs.cornell.edu/courses/cs6460/2011sp/papers/cloudsec-ccs09.pdf. Accessed 15/05/2013

Scalet SD (2009) 19 ways to build physical security into a data center. http://www.csoonline.com/article/220665/19-ways-to-build-physical-security-into-a-data-center. Accessed 09/04/2013

Shinder D (2007) 10 physical security measures every organization should take. http://www.techrepublic.com/blog/10-things/10-physical-security-measures-every-organization-should-take/. Accessed 28/07/2013

Further Reading

IBM (2013) Database ACL settings. Available online at http://publib.boulder.ibm.com/infocenter/sametime/v7r5m1/topic/com.ibm.help.sametime.imlu.doc/st_adm_security_usertypeacl_c.html. Accessed 02/05/2013.

LSoft Technologies (2013) Active@ Kill Disk—Hard Drive Eraser. http://download.cnet.com/Active-Kill-Disk-Hard-Drive-Eraser/3000-2092_4-10073508.html?tag=mncol;2. Accessed 28/07/2013

SQL Server (2012) Security and protection (database engine). Available on-line at http://msdn.microsoft.com/en-us/library/bb510589.aspx. Accessed 28/07/2013