Security – Authorization and Access Control

Authentication

Kerberos

Kerberos – Mediated Authentication

- Widely adopted and implemented in popular operating systems
  - See http://www.kerberos.org
  - See http://www.kerberos.org/software/tutorial.html
- Kerberos implements mediated authentication with tickets
- Kerberos uses time stamps as “nonces” in the mutual authentication phase of the protocol
- It aims to provide a universal “single sign-on” to services within a network

Kerberos

- Originally developed at M.I.T
- Widely adopted and implemented in popular operating systems
  - See http://www.kerberos.org
- The core Kerberos protocol is a service based on secret keys for providing authentication in open networks
- Authentication is mediated by a trusted third party on a network
  - "Authentication Server" : acts as the Key distribution Center (KDC)
- It aims to provide a universal "single sign-on" to services within a network

Kerberos Protocol

- Realm
  - Describes a Kerberos authentication domain
  - Users, hosts and services are registered within a realm
- Principal
  - A Kerberos Principal is the unique identity to which Kerberos can assign tickets
    - is an identifier that refers to entries in the Kerberos authentication database (registered users, hosts and services)
    - A principal is a text string with a specific format:

      E.g.: “martin@SOME_REALM”, “xxx/admin@ABERDEEN.AC.UK”

  - Each user, network host and service are represented by a Principal
  - The secret key stored with the principal is regarded the “Principal’s long-term secret key”
    - This long-term secret key is derived from the user’s password with the ‘string2key’ function, when the user registers with Kerberos
      - Is a hash function (e.g. MD5, SHA1 etc.) applied to the password
    - Each key change (e.g. through change of password) is recorded by incrementing the ‘Key Version Number’ (kvno), which is stored with principal in the Kerberos database

Kerberos Protocol

- Secret keys
  - The secret key stored with the principal is regarded the "Principal's long-term secret key"
  - This long-term secret key is generated for each principal
    - For users: derived from the user's password with the 'string2key' function, when the user registers with Kerberos
      - Is a hash function (e.g. MD5, SHA1 etc.) applied to the password
    - For applications / services, an administrator has to generate this secret key
  - Each key change (e.g. through change of password) is recorded by incrementing the 'Key Version Number' (kvno), which is stored with principal in the Kerberos database

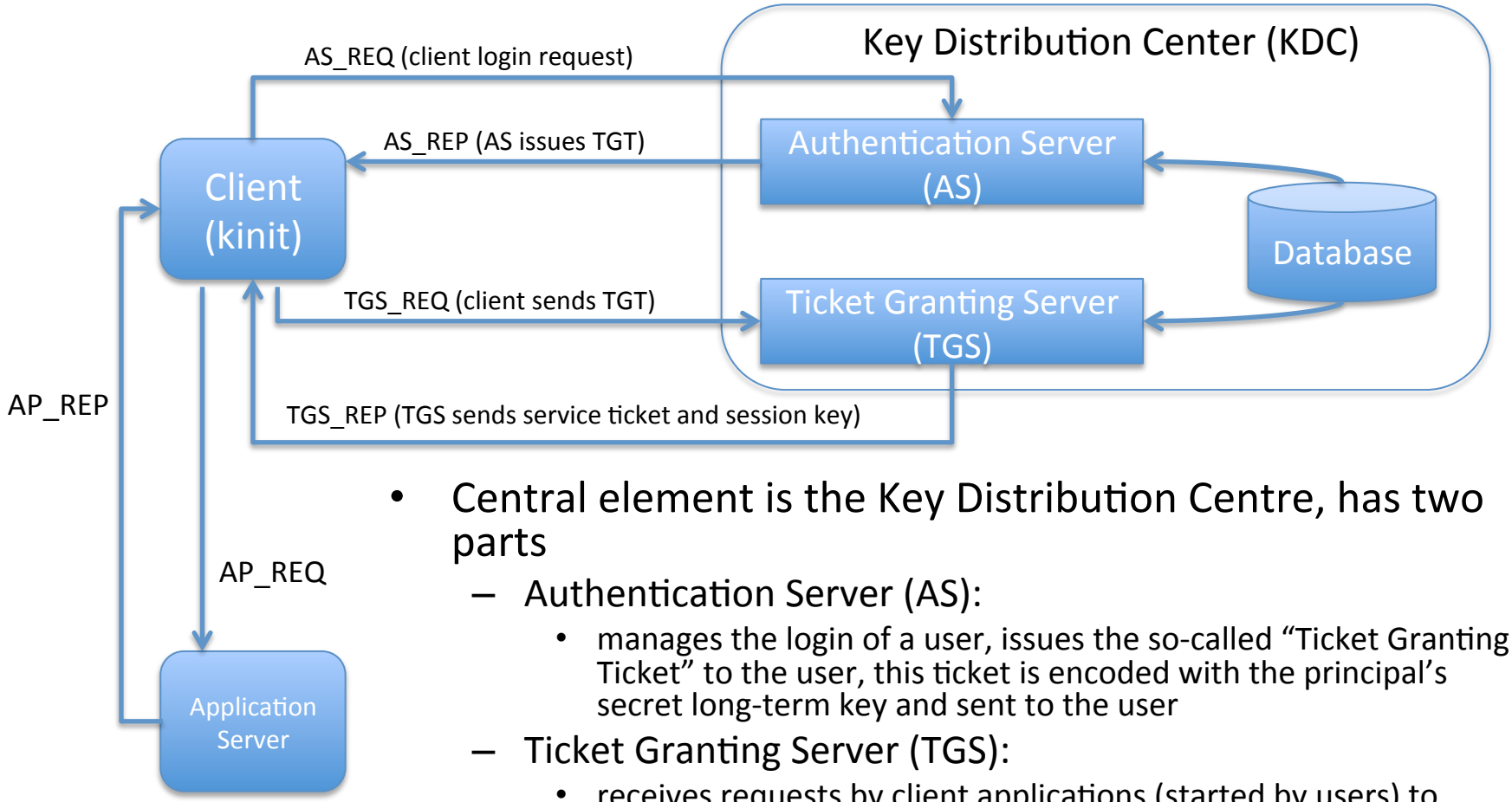
Kerberos Authentication

- Authentication method:
  - Users enter password once on local client machine only (kinit)
  - Authenticated via a central KDC
  - No passwords travel over the network
- Two types of tickets:
  - "Ticket-granting Ticket" (TGT):
    - Issued during login by the KDC (AS component), usually valid for 10 hours
    - TGT is used for authentication and to receive specific session tickets
    - A TGT has a timestamp (nonce) and expires after a set time
  - Session ticket:
    - Issued for each service or application server the client wants to contact
    - Client has to authenticate itself with the TGT to receive such a session ticket, no password required
    - KDC issues a session ticket (TGS component), together with a generated secret session key
    - A session ticket has a timestamp (nonce) and expires after a set time

Kerberos Authentication

- Kerberos uses time stamps as “nonces” in the mutual authentication phase of the protocol
  - This helps to identify whether an interceptor is replaying a message – messages with identical time stamps are rejected
- This means that Kerberos requires reasonably synchronised clocks among the users of the system

Kerberos Infrastructure



- Central element is the Key Distribution Centre, has two parts
  - Authentication Server (AS):
    - manages the login of a user, issues the so-called "Ticket Granting Ticket" to the user, this ticket is encoded with the principal's secret long-term key and sent to the user
  - Ticket Granting Server (TGS):
    - receives requests by client applications (started by users) to access servers, client sends the TGT to the KDC for such a request
    - TGS calculates a secret session key for communication between client and server, is sent to client with a new service ticket (service ticket is encoded with the server's long-term secret key)

Kerberos Authentication

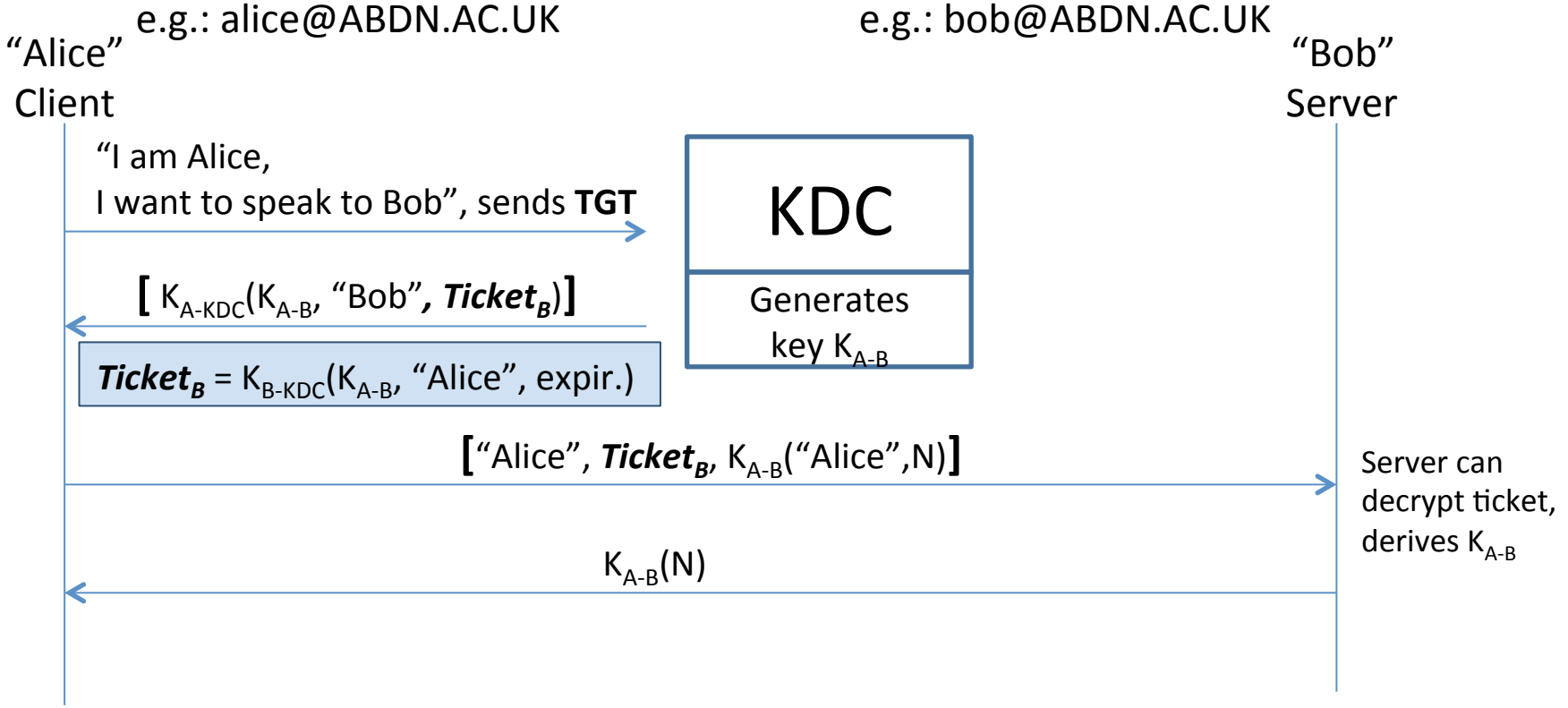
- AS_REQ:
  - Initial user authentication request, made with the Kerberos client application “kinit”
  - Sent to the AS component of the KDC
- AS_REP:
  - Reply message of the AS, contains the TGT encrypted with the TGS secret key (the Principal’s long-term

Kerberos Initial Authentication

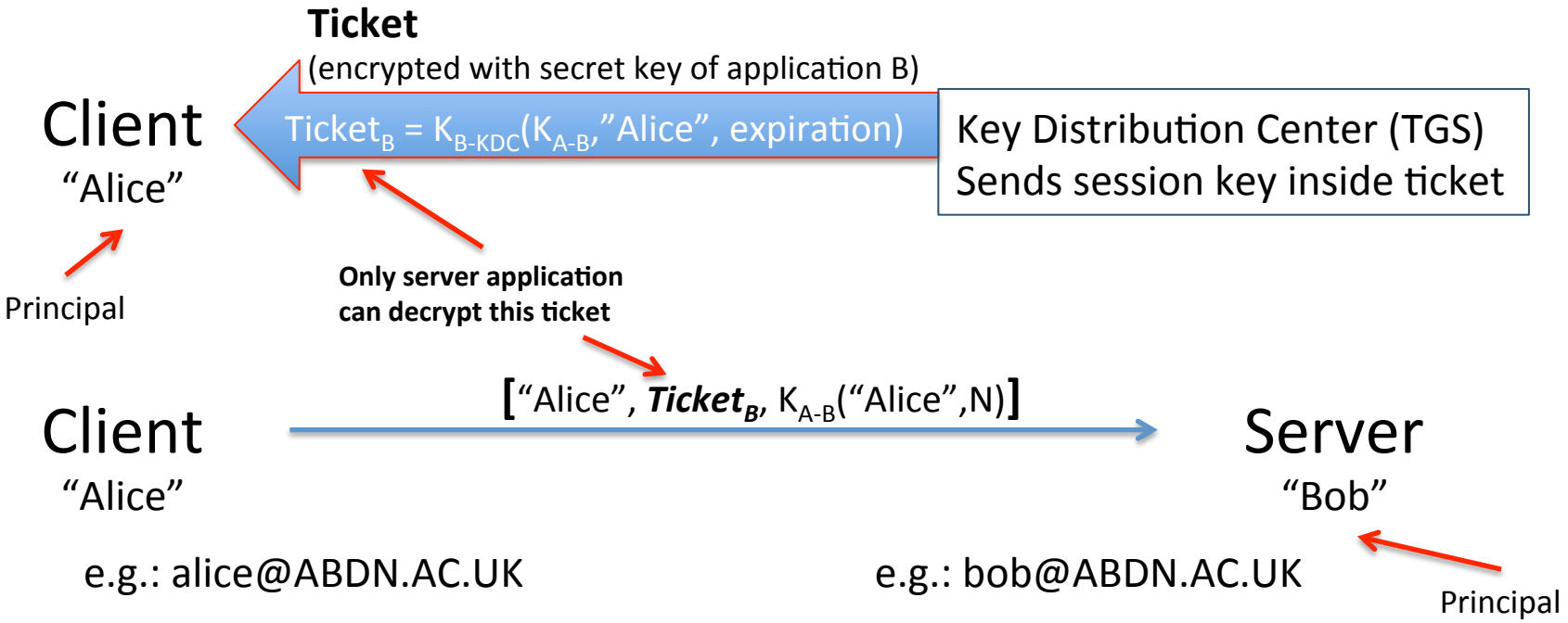
- Initial Authentication of a User within a Kerberos realm:
  - At local workstation via a special Kerberos client application (kinit)
    - This client application contacts the Kerberos KDC
- KDC returns the Ticket Granting Ticket, encrypted with the user's long-term secret key ($K_{A\text{-}KDC}$)
  - At the users workstation, this TGT is decrypted (user's long-term key derived from password)
- TGT is stored locally at the client for subsequent authorisation requests of this user

Kerberos Authentication and Access to Applications

- Client sends TGT for authentication to KDC (TGS)
- KDC (TGS) responds with a session ticket encoded with the server's long-term key ($K_{B\text{-}KDC}$)
- A timestamp N is used as a nonce



Ticket Transfer



- Key Distribution Center transmits a ticket that is encrypted with the server's long-term secret key ($K_{B\text{-}KDC}$)
- Only server can decrypt this ticket, therefore authenticating the client

Kerberos Access to Applications

- A client application (used by an authenticated user) sends user's TGT to the KDC, indicating that it wants to use a particular service
- The KDC authenticates the client
  - checks access privileges to service,
  - generates a random symmetric (short-term) session key K_{A-B} for communication between client and server
- The KDC sends a message back to the client, encoded with the shared key K_{A-KDC}
  - the value of K_{A-B}, and a **ticket** for accessing the service
    - $K_{A-KDC}(K_{A-B}, Ticket_B)$, where TicketB = K_{B-KDC} ("client", K_{A-B}, expir.)
- The client sends the ticket to the service, together with an **authenticator** for message to the service
  - the authenticator consists of the client name and a timestamp (nonce) *N* encrypted with K_{A-B}, that is K_{A-B}("client", *N*)
- The service decrypts the ticket, using the secret key K_{B-KDC} , with that it will learn about the session key K_{A-B}
- The service sends back the nonce to the client, encoded with K_{A-B} to show that it received the secret session key and is "alive". This is the **mutual authenticator**.

Kerberos Credentials

- Ticket
  - Allows user to authenticate to a service
  - Used to securely pass the identity of the user to which the ticket is issued between the KDC and the server
- Authenticator (Timestamp)
  - Proves that the user presenting the ticket is the user to which the ticket was issued
  - Proof that user knows the session key
  - Prevents ticket theft from being useful
  - Prevents replay attacks with an authenticator (timestamp encrypted with the session key):
    - K_{A-B}(timestamp)
    - Server maintains a cache of recent authenticators

Kerberos: Advantages

- Passwords aren't exposed to eavesdropping
- Password is only typed at the local workstation
  - Never travels over the network
  - Never transmitted to a remote server
- Password guessing more difficult
- Single Sign-in
  - More convenient: only one password, entered once
  - Users may be less likely to store passwords

Kerberos: Advantages

- Stolen Tickets hard to reuse
  - Need an authenticator (nonce) as well which can't be reused
- Much easier to effectively secure a small set of limited access machines (the KDC's)
- Easier to recover from host compromises
- Centralised user account administration

Kerberos: Limitations

- Kerberos server can impersonate anyone
- KDC is a single point of failure
  - But can have replicated KDC's
- KDC could be a performance bottleneck
  - Everyone needs to communicate with it frequently
  - Not a practical concern these days
  - Having multiple KDC's alleviates the problem

Kerberos: Limitations

- If local workstation is compromised, user's password could be stolen by a Trojan horse
  - Only use a desktop machine or laptop that you trust
  - Use hardware token pre-authentication
    - Require use of some hardware token to prove your identity before the TGT is issued.
- Kerberos vulnerable to password guessing attacks
  - Choose good passwords
  - Use hardware pre-authentication

Authorization

Access Control

Authorisation

- *Authorisation* - verifying what you are allowed to do (what services to use, what actions to perform), after a server has authenticated you
- Access Control is a mechanism to control (allow / deny) access to resources
  - Examples include files, services, machines on a network etc.
- Authorisation needs Authentication
  - User id has to be verified to determine authorised actions
  - Authorisations are linked to to a user ID

Access Control

Access Control

- Assign access rights to resources
  - Specify access policies
    - What access rights to resources should be granted to a user?
    - Which entity can do what ?
      - What actions are allowed / forbidden, what access rights to services
  - Examples
    - File access rights, use of network services, databases etc.
- Enforce access rights
  - Verify access rights of individuals, based on authentication
- Counteract security threats
  - Stop insiders from abusing access rights
  - Limit damage from external hackers

Access Control: Examples

- Possible restrictions:
  - Customer:
    - Can only see own data, has limited modification rights
  - Staff:
    - Restrictions on actions, e.g. monetary transactions: staff can only perform transactions below 1000 pounds, otherwise extra permission is required
  - Web applications:
    - Limited rights to access and manipulate a database

Deciding on Access Rights

- Classify users, resources, rights
- Specify a policy
  - Decide what rights each type of user has for each type of resource
  - Should these rights be permanent or temporary?
  - Should access be audited (logged in log files)?
- Enforce the policy
  - Implement access rights using system features (a security architecture)
  - User / group rights in UNIX
  - Access rights in SQL Databases
  - Active Directory

Classification

- Classify users into groups
  - Lectures, students, patients, doctors, administrator
- Classify resources into groups
- Classify access rights
  - Read, write, delete, modify, append

Access Control Lists (ACL)

- For each resource (or resource type), specify the access permission of each group
  - Each resource has a list of allowed groups
- Examples
  - Doctors have read-access to the blood test database
  - Lab personnel have read / write access to the blood test database
- Program-specific permissions
  - Introduce application-specific restrictions

Classic UNIX Access Control

- Unix users are organised in groups
  - A user is identified by its user ID (uid)
  - A user can be in multiple groups
    - A group is identified by a group ID (gid)
- Each UNIX resource / file has a particular owner (a UNIX user)
- Each process (program) started by the user inherits the user ID (uid) and group ID (gid) of the user
- These two ID's are used to identify what files / resources these programs may access
  - Each file has access rights defined for the file owner and for groups

The UNIX File System

- Consists of a hierarchical directory structure
- Each directory is a list of pairs:
  - (filename, inode number)
- Information contained in an inode:
  - Where the file is stored
  - File length
  - Last time file was read / written
  - Owner (the uid of the process that created the file)
  - Group (the gid of the process that created the file)
  - **Protection privileges: 12 bits**

UNIX Access Privileges

- Nine of the 12 bits encode access rights
  - User: read, write, execute
  - Group: read, write, execute
  - Other: read, write, execute

```
drwxrwsr-x   4 mkolling lecturer    4096 Jul 28  2011 public_html
```

- Other bits
  - Set user ID: change user ID on execution
    - if this bit is set, then the current user executing this file (program) has the same rights as the owner of the file being executed
  - Set group ID: change group ID on execution
    - If this bit is set, then the group rights of the owner of the file are inherited by the currently executing user
    - For directories: a new file created in a directory will inherit the group of the directory and not the group of the user / process that created the file

Windows Access Control

- Access Control Lists
  - Is a list of “access control entries” (ACE)
- Each ACE is defined for a resource
  - “open a file”
  - “run a program”
  - Etc.
- Each ACE specifies for a resource
  - A trustee (the individual, group or session)
  - The access rights (allowed, denied, audited)
- There are two types of ACLs
  - A DACL (discretionary access control list) is used to check whether a trustee can access the resource
  - A SACL (system access control list) is used to specify what access actions (or attempts) are to be audited

Access Control In SQL

- SQL
  - Access rights are granted to individual users
  - Access rights for resources such as tables, columns
  - Privileges
    - Select, insert, update, delete, grant, revoke, etc.
- MySQL additions
  - More privileges
    - Drop, alter, create, index
    - Also admin privileges (e.g. Shutdown, file)
    - Can also limit number of queries per hour
  - Groups
    - Specify rights by host as well as by user
    - Specify rights by connection type (e.g. SSL)
  - Can also audit access through general logs

Inference Attacks

Inference Control

- Can people with limited access rights use these to learn things we don't want them to know?
- Real danger with aggregate queries
  - E.g.: Exam:
    - Why is it wrong to make public the average marks grouped by UK, EU and International students, when there is only one UK student on the course?
- Inference attack
  - Using information from one security level from a database to infer a fact that should be protected at a higher security level

Tracker Queries

- Deduce information about individuals by querying census databases
  - Census contains anonymised information about a person: address, income, occupation, marital status, children etc.
  - If I know where a person lives and know other details except income, I can formulate a query that may give me a result set with one entry – I may be able to deduce the income of a person from anonymised data in a census database
- This is called a “tracker”
- Exploits extreme values (e.g. a very special occupation) and other background knowledge

Tracker Defence

- Possible defence
  - Database doesn't respond, if the number of entries in the result set is below a minimum value
- Does not work
  - Use multiple separate queries and intersect the result sets manually
- Defence is hard (see Anderson, chapter 8.3)

Blurred lines

- In practice, the boundary between authentication and authorization (the enforcement/verification process part) is not clear.
- For example, in many systems when a user logs in, the system generates an **access token**. The token is then used by processes operating on behalf of the user when they need to access resources.

Example Online Bookstore

Example: Online Bookstore

- What are the groups?
- What are the resources?
- What are the access rights?
- What are the threats?

Example: Online Bookstore

- Designing the access groups:
  - Customers, suppliers, affiliates, staff, ...
- Resources
  - Orders, customer information, payments, ...
- Access rights
  - Read, write
- Threats
  - Theft of money (payments)
  - Theft of customer info (credit card number)
  - Theft of info about what books a customer is reading

Example: Online Bookstore

- Access: Customer
  - Should have full access to their own orders and details
  - Summary access to other orders
    - E.g.: recommendations such as “N people have bought book X, and M of them also bought book Y”
    - How much detail is revealed via this information?
  - No access to other sensitive information

Example: Online Bookstore

- Access: Affiliates
  - Full access to payments owed to them
  - Limited access to orders placed through them
    - What books are ordered through them?
    - When are books ordered
    - No names of individuals who ordered the books
  - Maybe some demographic information and statistics?
    - Too much demographic information may allow a customer to be identified

Example: Online Bookstore

- Access: Staff
  - Staff have full access to everything
    - How to stop someone entering a payment to themselves?
    - Or selling customer details to an investigator?
  - Possible defences
    - Audit of access to data
    - Different rights assigned to different staff
    - Use of dual authorisation

Key Points

- Access control:
  - Authentication, Authorisation, Accountability
  - Reference Monitor
  - Policy: define the rights different users have to data
  - Use OS/DB features to
  - It is essential to limit the damage that can be done by insiders as well as outsiders
  - Design of the right and effective policies may be hard
- Inference control
  - Attacker can learn a lot from what is regarded as limited or non-critical data