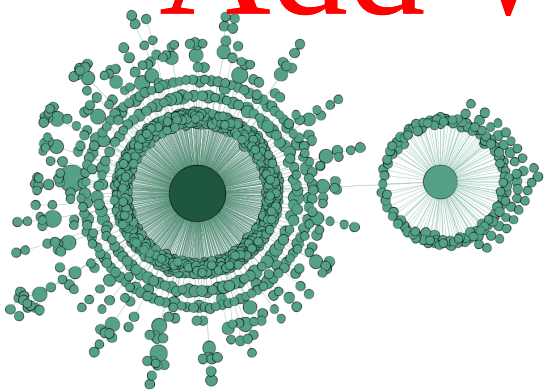
Week 9 Workshop - Database Security

Qing Wang

Website: http://users.cecs.anu.edu.au/~u6170295/
https://graphlaoana.github.io/website/

Deep learning on graphs
Graph algorithms.



House Keeping

- L
si
- Assignment 2 (Database Theory) is due at 23:59.

Week 9 Workshop - Database Security

“Hard

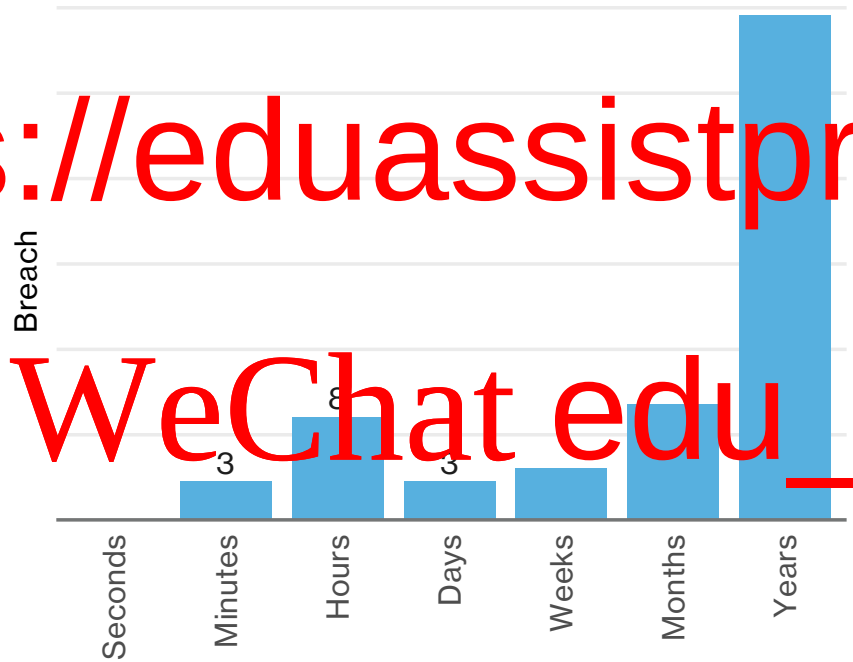
buy a

more t

seconds; and be stolen without you

Data Breaches

- In 80% of cases, attackers are able to compromise an organization within minutes. However, in almost 60% of cases, it takes years to learn that they have been breached.[1]



Time-to-discovery within Public breaches (n=66)

[1] Verizon 2016&2017 Data Breach Investigation Reports

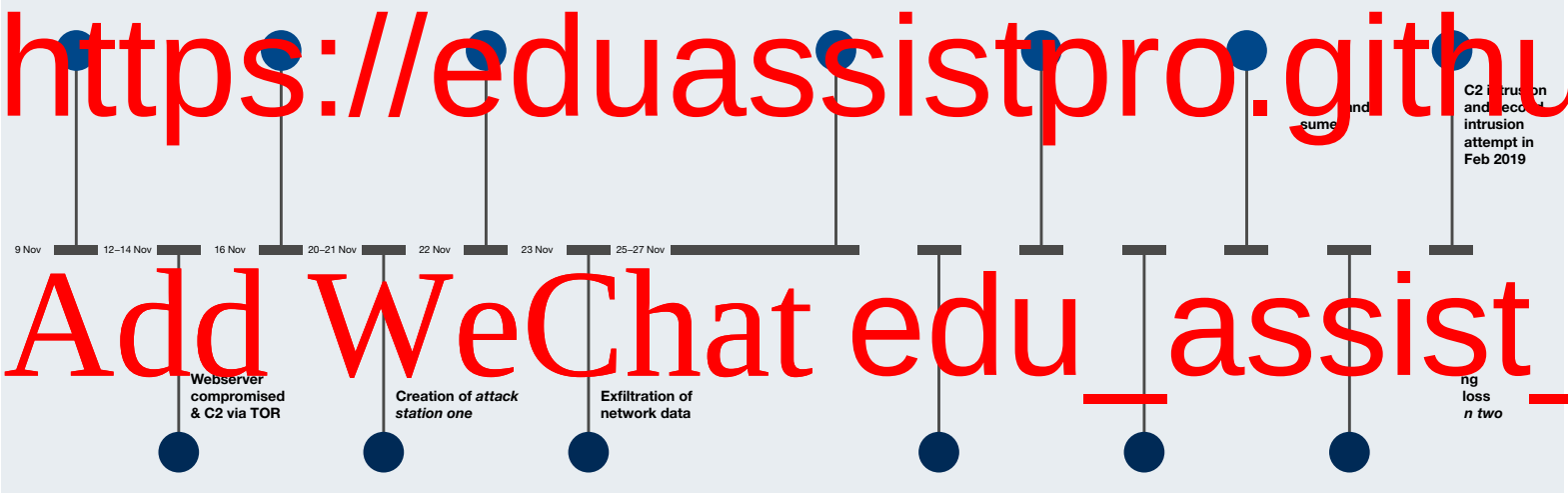
Data Breaches

- June 2019, ANU community was notified of a data breach.

Data Breaches

- June 2019, ANU community was notified of a data breach.

"It

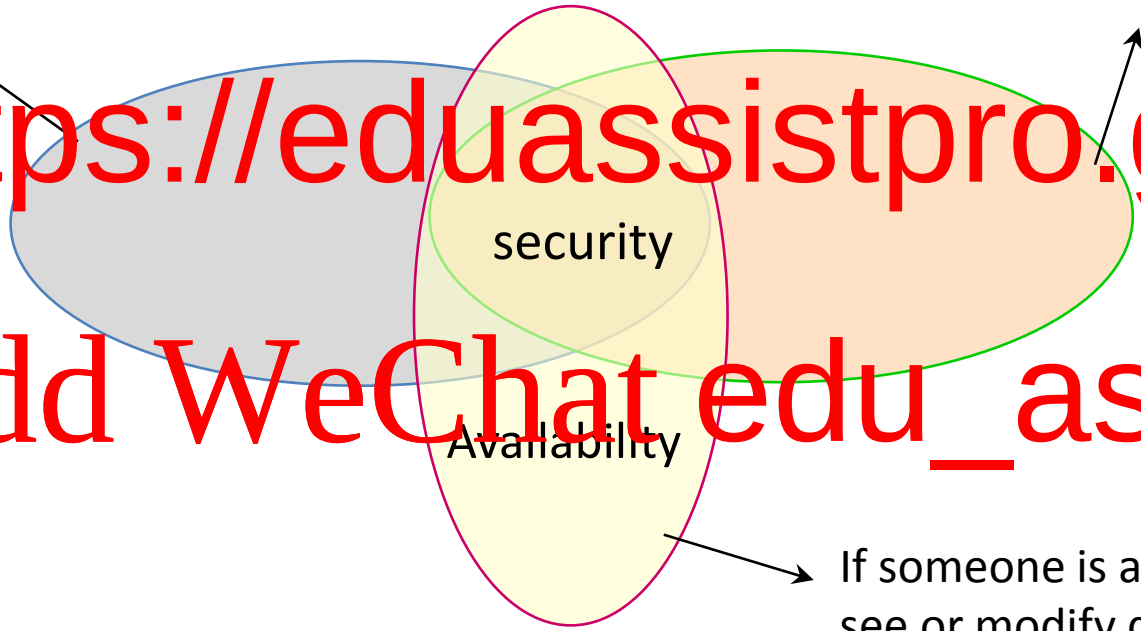


"While we cannot confirm exactly what data was taken, we know it was much less than the 19 years' worth we originally reported"

Objectives of Database Security

Data should only be shown to people who are all

Data should only be modified by people who modify it.



If someone is allowed to see or modify data, they should be able to do so.

Database Security - Examples

1. *A health-care information system*

Database Security - Examples

1. *A health-care information system*

Database Security - Examples

1. *A health-care information system*

Database Security - Examples

1. *A health-care information system*

Database Security - Examples

1. *A health-care information system*
2. *A military system*

Database Security - Examples

1. *A health-care information system*
2. *A military system*
   - The target of a missile cannot be given to an unaut

Database Security - Examples

1. *A health-care information system*

2. *A military system*
   - The target of a missile cannot be given to an unaut
   - The target of a missile cannot be arbitrarily modified.

Database Security - Examples

1. *A health-care information system*
2. *A military system*
   - The target of a missile cannot be given to an unaut
   - The target of a missile cannot be arbitrarily modified.
   - The target of a missile can be accessed when needed.

Database Security - Core Services

- Confidentiality

Database Security - Core Services

- **Confidentiality**
  - E.g. enforced by access control mechanisms

Database Security - Core Services

- **Confidentiality**
  - E.g. enforced by access control mechanisms

Database Security - Core Services

- Confidentiality
  - E.g. enforced by access control mechanisms
-
-
  - E.g. enforced by recovery and concurr

Database Security - Core Services

- **Confidentiality**
  - E.g. enforced by access control mechanisms
-

-
  - E.g. enforced by recovery and concurr

Some further services

- **Encryption**: to protect data when being tra
  and when being stored on secondary storage
- **Query authentication**: to ensure a query result is correct by using signature mechanisms and data structures
- ...

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Access Control Mechanisms

- T

1. Discretionary access control (DAC)
2. Mandatory access control (MAC)
3. Role-based access control (RBAC)

Database Security - DAC

Your objects at your own discretion!

Grant privileges

Revoke privileges

Bob

Alice

Granting/Revoking/Delegating Privileges


```
GRANT privileges ON object TO users [WITH GRANT OPTION]
```

Granting/Revoking/Delegating Privileges


```
GRANT privileges ON object TO users [WITH GRANT OPTION]
```

R

Granting/Revoking/Delegating Privileges


```
GRANT privileges ON object TO users [WITH GRANT OPTION]
```

```
R
```


- Possible privileges:
  - `SELECT`
  - `INSERT` and `INSERT`(column)
  - `UPDATE` and `UPDATE`(column)
  - `DELETE`
  - `REFERENCES`(column)
  - ...

Granting/Revoking/Delegating Privileges

- The privileges of an object can be given to a user with or without the `GRANT OP`

Granting/Revoking/Delegating Privileges

- The privileges of an object can be given to a user with or without the `GRANT OP`

- The privileges of an object can be taken away from a
to only revoke the `GRANT OPTION` on a privil


```
REVOKE SELECT ON Supplier FROM Bob;
REVOKE GRANT OPTION FOR SELECT ON Supplier FROM Bob;
```

Question

- In

(3) The user is a superuser of the database.

(4) The user has received the privilege with the owner of the object.

Example - Granting Privileges

- Alice owns table EMPLOYEE

```
(Alice):  GRANT SELECT, INSERT ON Employee TO Bob WITH GRANT OPTION;
(Alice):  GRANT SELECT ON Employee TO Jane WITH GRANT OPTION;
```

Example - Granting Privileges

- Alice owns table EMPLOYEE

```
(Alice): GRANT SELECT, INSERT ON Employee TO Bob WITH GRANT OPTION;
(Alice): GRANT SELECT ON Employee TO Jane WITH GRANT OPTION;
```

- Questions:
  1. What privilege(s) does Jane receive?

Example - Granting Privileges

- Alice owns table EMPLOYEE

```
(Alice): GRANT SELECT, INSERT ON Employee TO Bob WITH GRANT OPTION;
(Alice): GRANT SELECT ON Employee TO Jane WITH GRANT OPTION;
```

- Questions:
  1. What privilege(s) does Jane receive?
  2. What privilege(s) does Tom receive?

Example - Granting Privileges

- Alice owns table EMPLOYEE

```
(Alice): GRANT SELECT, INSERT ON Employee TO Bob WITH GRANT OPTION;
(Alice): GRANT SELECT ON Employee TO Jane WITH GRANT OPTION;
```

- Can these commands be executed?

Example - Granting Privileges

- Alice owns table EMPLOYEE

```
(Alice):  GRANT SELECT, INSERT ON Employee TO Bob WITH GRANT OPTION;
(Alice):  GRANT SELECT ON Employee TO Jane WITH GRANT OPTION;
```


- Can these commands be executed?

Example - Granting Privileges

- Alice owns table EMPLOYEE

```
(Alice):  GRANT SELECT, INSERT ON Employee TO Bob WITH GRANT OPTION;
(Alice):  GRANT SELECT ON Employee TO Jane WITH GRANT OPTION;
```


- Can these commands be executed?
  - The first three are fully executed.

Example - Granting Privileges

- Alice owns table EMPLOYEE

```
(Alice):  GRANT SELECT, INSERT ON Employee TO Bob WITH GRANT OPTION;
(Alice):  GRANT SELECT ON Employee TO Jane WITH GRANT OPTION;
```


- Can these commands be executed?
  - The first three are fully executed.
  - The fourth one is not executed, because Bob do `UPDATE` privilege on the table.

Example - Granting Privileges

- Alice owns table EMPLOYEE

```
(Alice):  GRANT SELECT, INSERT ON Employee TO Bob WITH GRANT OPTION;
(Alice):  GRANT SELECT ON Employee TO Jane WITH GRANT OPTION;
```


- Can these commands be executed?
  - The first three are fully executed.
  - The fourth one is not executed, because Bob do `UPDATE` privilege on the table.
  - The fifth one is partially executed because Jane has the `SELECT` and `INSERT` privileges but no `GRANT OPTION` for `INSERT`. Therefore, Tom only receives the `SELECT` privilege.

Granting/Revoking/Delegating Privileges

- A user can only revoke privileges that the user has granted earlier, with two optional keywords in the `REVOKE` command:

  Possible implementations:

  (1) Causing an error message in some DB privilege is still delegated

  (2) Revoking the privilege from the specifi

- If a user receives a certain privilege from multiple sources, and the user would lose the privilege only after all sources revoke this privilege.

Example - Revoking Privileges

- Again, Alice owns table EMPLOYEE:


```
(Bob):  REVOKE SELECT ON Employee FROM Tom;
```


- Will Tom lose the SELECT privilege on EM

Example - Revoking Privileges

- Again, Alice owns table EMPLOYEE:


```
(Bob):  REVOKE SELECT ON Employee FROM Tom;
```


- Will Tom lose the SELECT privilege on EM

  - Tom will still hold the SELECT privilege on EMPLOYEE, since he has independently obtained such privilege from Jane.

Example - Revoking Privileges

- Again, Alice owns table EMPLOYEE:


```
(Jane): GRANT SELECT ON Employee T
(Alice): REVOKE SELECT ON Employee FROM Bob CA
```


- Will Tom lose the SELECT privilege on EMPLOYEE?

Example - Revoking Privileges

- Again, Alice owns table EMPLOYEE:


```
(Jane): GRANT SELECT ON Employee TO Tom;
(Alice): REVOKE SELECT ON Employee FROM Bob CA
```


- Will Tom lose the SELECT privilege on EM

Example - Revoking Privileges

- Again, Alice owns table EMPLOYEE:


```
(Jane): GRANT SELECT ON Employee TO Tom;
(Alice): REVOKE SELECT ON Employee FROM Bob CA
```


- Will Tom lose the SELECT privilege on EM
  - Tom will lose the SELECT privilege on EMPLOYEE.

Delegating Privileges - Propagation

- There are techniques to limit the propagation of privileges.

  - Limiting **vertical propagation**: limits t privileges.

- How can we keep track of privilege propagation?

Privilege Propagation

- tuna owns:

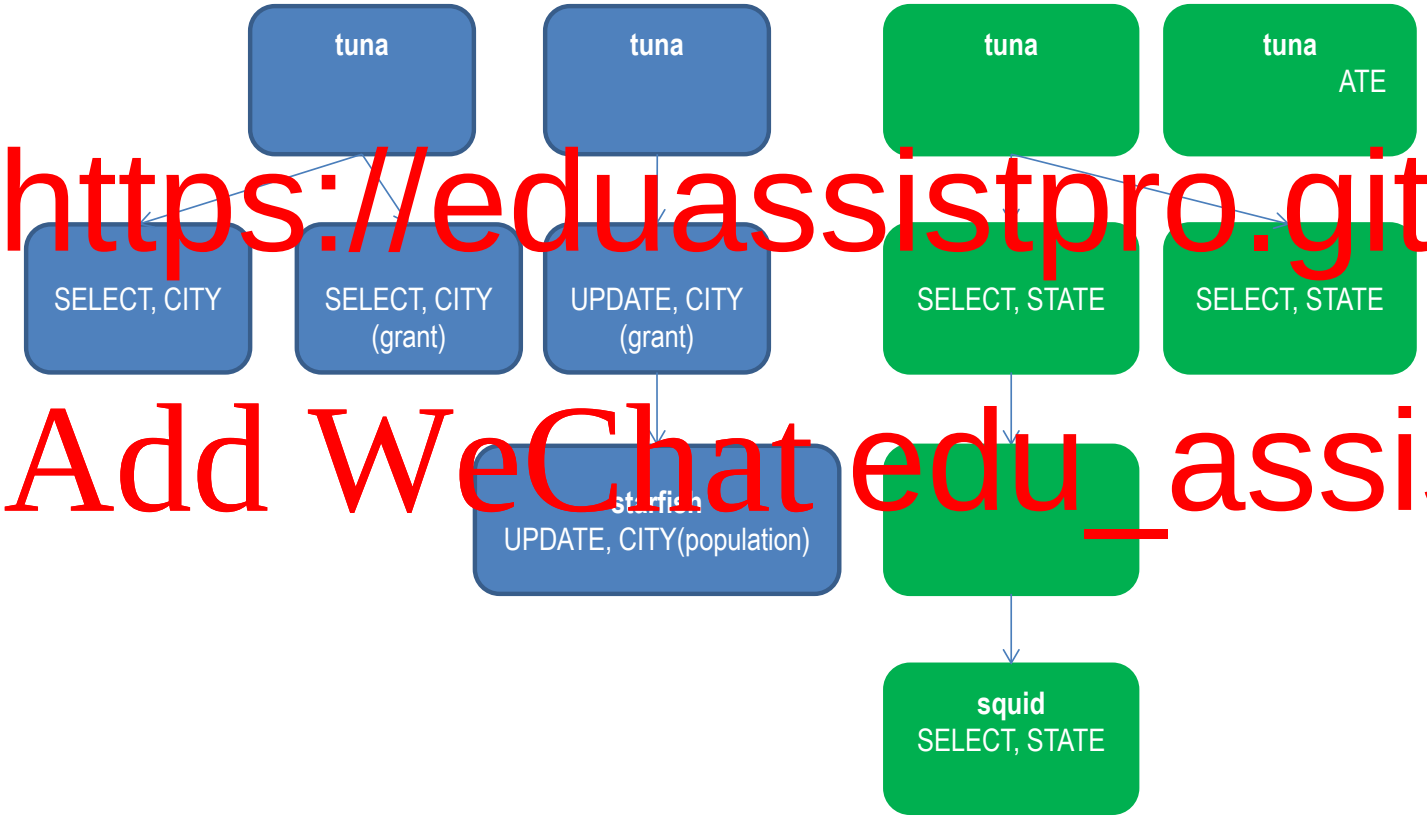
CITY(name, state, population)

- T

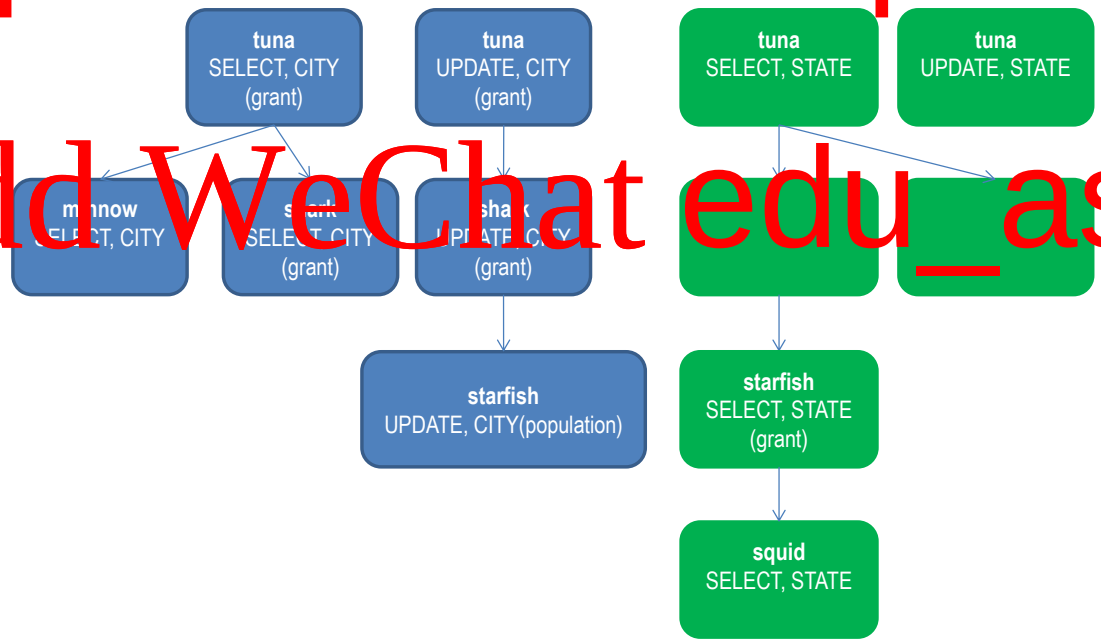
```
    (tuna):  GRANT SELECT, UPDATE ON CITY
    (tuna):  GRANT SELECT ON CITY TO minnow;
    (tuna):  GRANT SELECT ON STATE TO shark, m
   (shark):  GRANT SELECT ON STATE TO starfi
   (shark):  GRANT UPDATE (population) ON CITY TO starfish;
(starfish):  GRANT SELECT ON STATE TO squid;
   (shark):  ...
```

Privilege Propagation

- A grant graph can be used to keep track of privilege propagation.



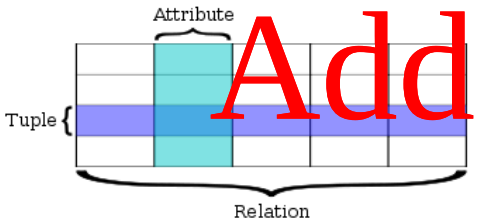
```
(tuna):  GRANT SELECT, UPDATE ON City TO shark WITH GRANT OPTION;
(tuna):  GRANT SELECT ON City TO minnow;
(tuna):  GRANT SELECT ON State TO shark, minnow WITH GRANT OPTION;
(shark):  GRANT SELECT ON State TO starfish WITH GRANT OPTION;
(shark):  GRANT UPDATE (population) ON City TO starfish;
(star
```

Using Views


```
CREATE VIEW view_name AS
SELECT attribute_list
FROM table_list
```


- Vi
  a



Some examples:

1. The owner A of a relatio d access to some columns of that includes only those colum
2. The owner A of a relation R wants to give a user B read access to some rows of R. A can create view V_2 that selects only those rows from R.

Using Views

- E

```
(To                                                    AMS
        GROUP BY CourseID
        Having AVG(Grade)<50;
(Tom): CREATE VIEW AllCourses AS
        SELECT CourseID, Grade FROM EXAMS;
```

Database Security - DAC

Your objects at your own discretion!

Grant privileges
Revoke privileges

Bob

Alice


Database Security - DAC

Your objects at your own discretion!

Grant privileges
Revoke privileges

Bob

Alice


– `GRANT SELECT ON` tableB `TO` Alice;

Database Security - DAC

Your objects at your own discretion!

Grant privileges
Revoke privileges

Bob

Alice


- `GRANT SELECT ON` tableB `TO` Alice;
- `REVOKE SELECT ON` tableB `FROM` Alice;

Database Security - DAC

Your objects at your own discretion!

Grant privileges
Revoke privileges

Bob

Alice


- `GRANT SELECT ON` tableB `TO` Alice;
- `REVOKE SELECT ON` tableB `FROM` Alice;
- `GARNT SELECT ON` tableB `TO` Alice
  `WITH GRANT OPTION`;

Database Security - DAC

Your objects at your own discretion!

Grant privileges
Revoke privileges

Bob

Alice


- `GRANT SELECT ON tableB TO Alice;`
- `REVOKE SELECT ON tableB FROM Alice;`
- `GARNT SELECT ON tableB TO Alice`
  `WITH GRANT OPTION;`
- `REVOKE GRANT OPTION FOR SELECT`
  `ON tableB FROM Alice;`

Database Security - DAC

Your objects at your own discretion!

Grant privileges
Revoke privileges

Bob

Alice


- `GRANT SELECT ON tableB TO Alice;` – G ;
- `REVOKE SELECT ON tableB FROM Alice;`
- `GARNT SELECT ON tableB TO Alice`
  `WITH GRANT OPTION;`
- `REVOKE GRANT OPTION FOR SELECT`
  `ON tableB FROM Alice;`

Database Security - DAC

Your objects at your own discretion!

Grant privileges
Revoke privileges

Bob

Alice


- `GRANT SELECT ON` tableB `TO` Alice;
- `REVOKE SELECT ON` tableB `FROM` Alice;
- `GARNT SELECT ON` tableB `TO` Alice
  `WITH GRANT OPTION`;
- `REVOKE GRANT OPTION FOR SELECT`
  `ON` tableB `FROM` Alice;

- `G` ;
- `REVOKE SELECT ON` tableA `FROM` Bob;

Database Security - DAC

Your objects at your own discretion!

Grant privileges
Revoke privileges

Bob

Alice


- `GRANT SELECT ON` tableB `TO` Alice;
- `REVOKE SELECT ON` tableB `FROM` Alice;
- `GARNT SELECT ON` tableB `TO` Alice `WITH GRANT OPTION`;
- `REVOKE GRANT OPTION FOR SELECT ON` tableB `FROM` Alice;
- `G` ;
- `REVOKE SELECT ON` tableA `FROM` Bob;
- `GRANT SELECT ON` tableB `TO` Tom;

Database Security - DAC

Your objects at your own discretion!

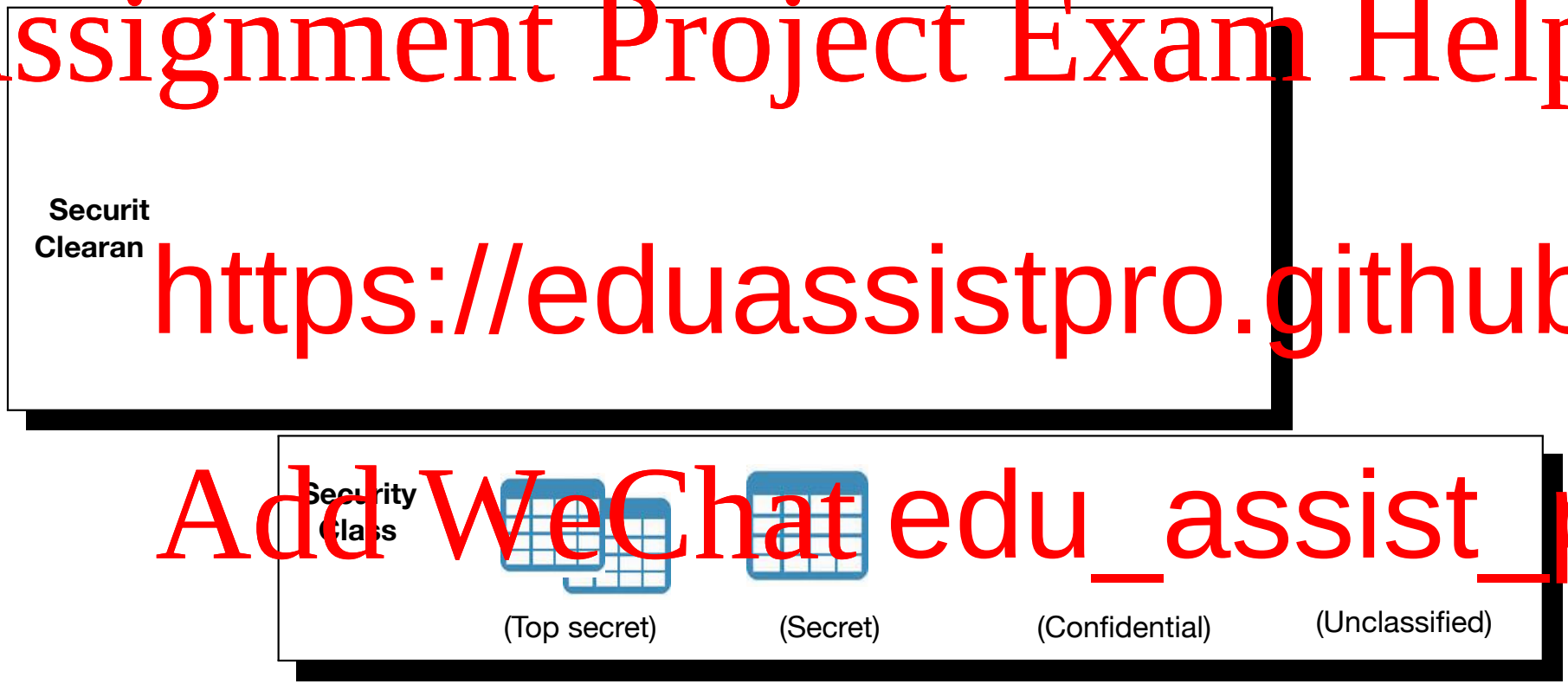
Grant privileges
Revoke privileges

Bob

Alice


- `GRANT SELECT ON tableB TO Alice;`
- `REVOKE SELECT ON tableB FROM Alice;`
- `GARNT SELECT ON tableB TO Alice WITH GRANT OPTION;`
- `REVOKE GRANT OPTION FOR SELECT ON tableB FROM Alice;`
- `G ;`
- `REVOKE SELECT ON tableA FROM Bob;`
- `GRANT SELECT ON tableB TO Tom;`
- `REVOKE SELECT ON tableB FROM Tom;`

Database Security - MAC



System-wide policies govern controlled access to classified information.

Mandatory Access Control

- It is based the **Bell-LaPadula** model (originally developed for U.S. Department of Defense multilevel security policy).
  - Subjects (e.g. users) are assigned *security clearances*;
  - *ses*.

Mandatory Access Control

- It is based the **Bell-LaPadula** model (originally developed for U.S. Department of Defense multilevel security policy).
  - Subjects (e.g. users) are assigned *security clearances*;
  - *ses*.

- Two rules are enforced by the model:
  1. Subject X can read object Y only if
     $\hookrightarrow$ **Read down**
  2. Subject X can write object Y only if $clearance(X) \leq class(Y)$.
     $\hookrightarrow$ **Write up**

Mandatory Access Control

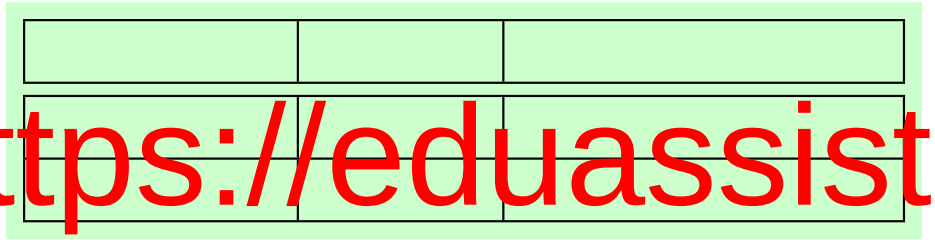
- It is based the **Bell-LaPadula** model (originally developed for U.S. Department of Defense multilevel security policy).
  - Subjects (e.g. users) are assigned *security clearances*;
  - *ses*.

- Two rules are enforced by the model:
  1. Subject X can read object Y only if $clearance(X) \geq class(Y)$.
     ↪ **Read down**
  2. Subject X can write object Y only if $clearance(X) \leq class(Y)$.
     ↪ **Write up**

- The key idea is "**preventing information in high level objects from flowing to low level subjects**".

Mandatory Access Control

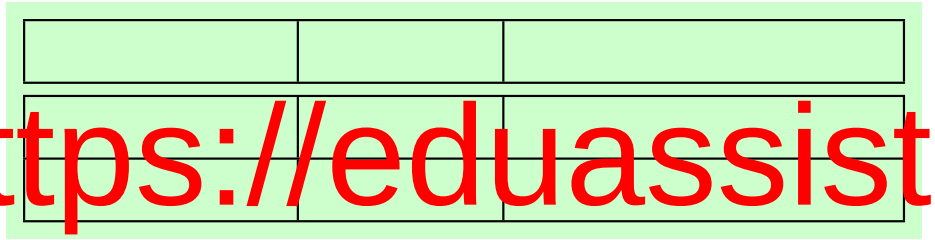
- **Multilevel relations:** Assume that each row is assigned a security class. Then users with different security clearances see a different collection of rows when they access the same table.



- Bob with C clearance can only access the se
- Peter with S clearance can access both tupl

Mandatory Access Control

- **Multilevel relations:** Assume that each row is assigned a security class. Then users with different security clearances see a different collection of rows when they access the same table.

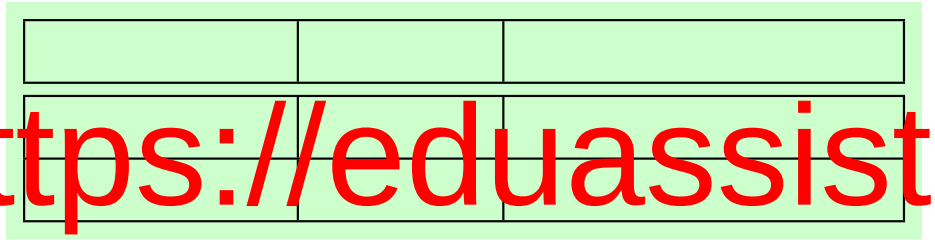


  - Bob with C clearance can only access the se
  - Peter with S clearance can access both tupl

- Suppose that city is the primary key, and Bob with C cl add a row (*Paris*, 4, *confidential*(*C*)).
  1. What would happen?

Mandatory Access Control

- **Multilevel relations:** Assume that each row is assigned a security class. Then users with different security clearances see a different collection of rows when they access the same table.

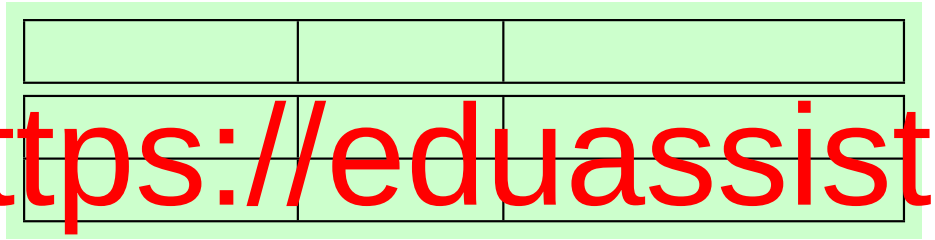


  - Bob with C clearance can only access the se
  - Peter with S clearance can access both tupl

- Suppose that city is the primary key, and Bob with C cl add a row (*Paris*, 4, *confidential*(*C*)).
  1. What would happen? The first record may be (partial) inferred.

Mandatory Access Control

- **Multilevel relations:** Assume that each row is assigned a security class. Then users with different security clearances see a different collection of rows when they access the same table.

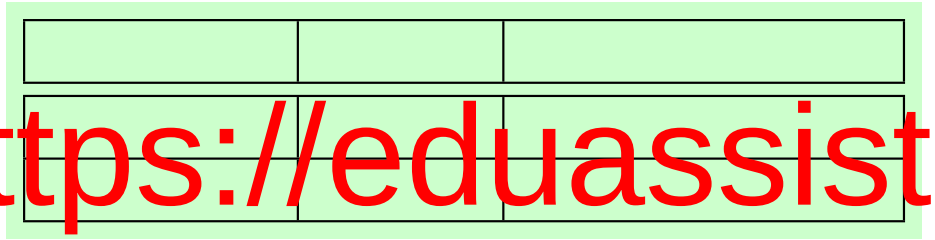


  - Bob with C clearance can only access the se
  - Peter with S clearance can access both tupl

- Suppose that city is the primary key, and Bob with C cl add a row (*Paris*, 4, *confidential*(*C*)).
  1. What would happen? The first record may be (partial) inferred.
  2. How to solve the potential security issues?

Mandatory Access Control

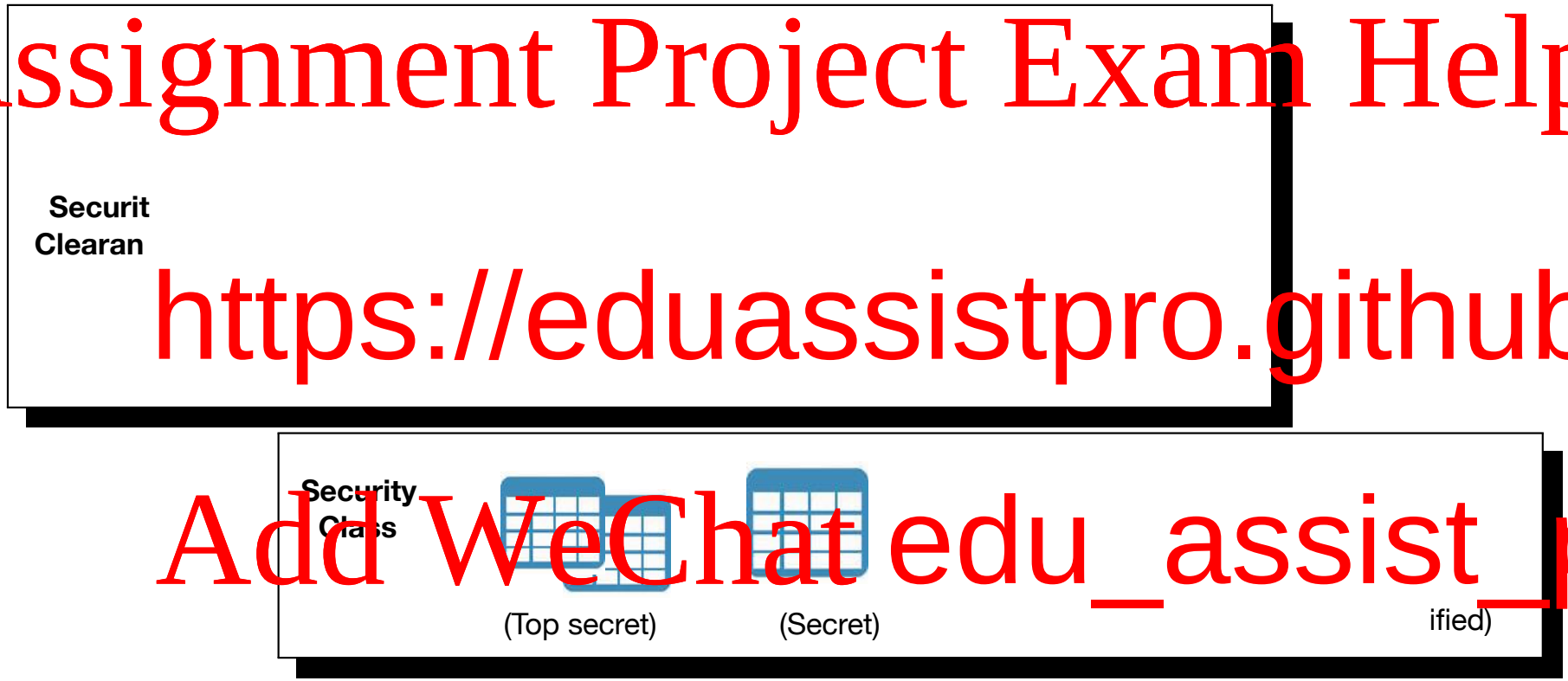
- **Multilevel relations:** Assume that each row is assigned a security class. Then users with different security clearances see a different collection of rows when they access the same table.



  - Bob with C clearance can only access the se
  - Peter with S clearance can access both tupl

- Suppose that city is the primary key, and Bob with C cl add a row (*Paris*, 4, *confidential*(*C*)).
  1. What would happen? The first record may be (partial) inferred.
  2. How to solve the potential security issues? whiteTreating security class as part of the primary key.

Database Security - MAC



- **Read down**: Subject X can read object Y only if clearance(X) $\geq$ class(Y).
- **Write up**: Subject X can write object Y only if clearance(X) $\leq$ class(Y).

DAC vs MAC

- How do DAC and MAC differ from each other?

DAC vs MAC

- How do DAC and MAC differ from each other?

DAC vs MAC

- How do DAC and MAC differ from each other?
  -
  -
  -
  -
  - ...
- **MAC is comparatively rigid.**
  - The system decides how data is shar
  - Each object is given a security class, a security clearance.
  - An object can then be accessed by users with the appropriate clearance.
  - ...

DAC - Limitations

- S

ver,
  - Steve may steal the information in *R* from Bob.

  - How?

DAC - Limitations

- S

ver,
  - Steve may steal the information in *R* from Bob.

  - How? **Trojan Horse attacks**.

DAC - Limitations

- **Trojan Horse attacks** If Steve tricks Bob into copying data from table R into table R', then the access control on table R doesn't apply to the copy of the data in table R'.

- Can this problem occur in MAC?

DAC - Limitations

- D
  - o

- MAC prevents illegitimate flow of information by a
  to objects and security clearances to subjects

SQL Injection Attacks

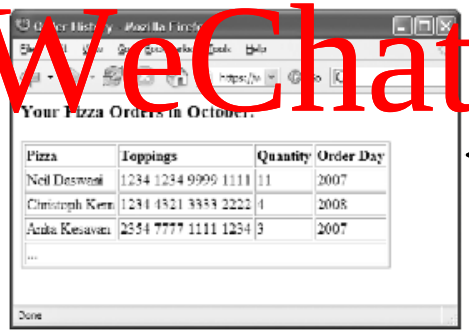
- SQL injection is one of the most basic and oldest tricks hackers use to get int
- w

1. Connect to the database;
2. Send SQL statements to the database;
3. Fetch the result and display data from the database;
4. Close the connection.

SQL Injection Attacks

- Many web applications take user input from a form.
- A user input is used in constructing a SQL query submitted to a database.
- A S

Hacker



Web forms

Fetch SQL results

Close the connection

Databases

SQL Injection - Example

- Consider a pizza-ordering application that allows users to review the orders they have made in a given month.

The pizza order review form

SQL Injection - Example

- When the form is submitted, it results in an HTTP request to the application:

  _ h=10

- W

```
s  _                                    _
        + "FROM orders "
        + "WHERE userid=" + session.getC
        + "AND order_month=" + reque
```


- Assuming that the current user's userid is 1234, we have:

```
SELECT pizza, toppings, quantity, order_day
  FROM orders
WHERE userid=1234 AND order_month=10
```

SQL Injection - Example

- The application then executes the query and retrieves the result set.

Pizza order his

- How can this application be attacked?

SQL Injection - Example

- What would happen if an attacker replaces '10' with '`0 OR 1=1`'?

- Alternatively, the attacker may modify the HTTP r

https://www.deliver-me-pizza.com/show_orders?month=0%20OR%201%3D1

Then request.getParameter("month") extracts '0%20*OR*%201%3*D*1' and returns the string '`0 OR 1=1`'.

SQL Injection - Example

- The SQL query that the application constructs and sends to the database now becomes:

- Since the operator precedence of the `AN` `OR`, the `WHERE` condition is equivalent to


```
WHERE (userid=4123 AND order_mo
```

- What happened?

  The (malicious) user supplied a parameter that, once inserted into the SQL query string, actually altered the meaning of the query!

SQL Injection - Example

- However, the attacker might be able to do even more damage, e.g., making a request such that the request parameter month evaluates to:

```
0 AND 1=0
```


- Then, the SQL query that the application construc database becomes:


```
SELECT pizza, toppings, quantity, order_
FROM orders
WHERE userid=4123 AND order_month=0 AND 1=0
UNION
SELECT cardholder, number, exp_month, exp_year
FROM creditcards
```

SQL Injection - Example

- As a result, the attacker receives an HTML page that contains the entire content of the ored towards able.

SQL Injection Attacks

- How can we prevent SQL injection attacks?
- Can SQL injection attacks be prevented by any of the following security so

1. **Firewall**

   i.e., monitors and controls the incoming and outgoing network traffic based on predetermined security rules

2. **Intrusion detection system** (IDS)

   i.e., monitors a network or systems for malic violations

3. **Authentication**

   i.e., the process by which a system can identify users

SQL Injection Attacks - Protection Techniques

- Several techniques of input validation:
  - **Blacklisting**?

  - **Whitelisting**?

    i.e., explicitly test whether a given input is wit
    values that are known to be safe, e.g., the par
    that represents a non-negative integer.

  - **Escaping**?

    i.e., transform dangerous input characters to turn a potentially dangerous input string into a sanitized one, e.g., escape(o‘connor)= o“connor (the double quote is the escaped version of the single quote).

SQL Injection Attacks - Protection Techniques

- The recommended solution: **Parameterized Queries**

  Two steps:

  1.
  2.


```
PreparedStatement stmt=conn.prepareStatement(
    "SELECT pizza, toppings, quant          _
    + "FROM orders WHERE userid=? AND or    _
stmt.setInt(1, session.getCurrent
stmt.setInt(2, Integer.parseInt(request.getParamenter("month")));
ResultSet res = ps.executeQuery();
```


- The key idea is "**separation between control and data**"!

Research Topics

- Differential privacy

- What Apple's differential privacy means for your data and the future machine learning: *https://techcrunch.com/2016/06/14/differential-privacy/*
- Learning with privacy at scale: *https://machinelearning.apple.com/research/learning-with-privacy-at-scale*

Research Topics

- Data anonymization

- k-anonymity: *https://en.wikipedia.org/wiki/K-anonymity*