1.

A

i.

DAC , depending on your wishes, accessing one place after another or granting others relevant access.

ii.

MAC, often used for sensitive or confidential data. It allows one entity (or user) to have access to a piece of data while another entity does not.

iii.

RBAC is a relatively new and widely used access control system. It is a system that pays privileges to roles (or positions) rather than to specific entities. Any entity that assumes the role can have the corresponding privileges.

iv.

ABAC, the system designer can decide which attributes are important for the authority of things such as subjects and environments, and then come up with policies.

B

i.

In UNIX systems, Discretionary access control is generally used to establish access policies.

First unix has read, write, and execute attributes to match the access policy and provide different levels of access to the user.

Then, there are group class that specify the permissions of the group of file owners.

And if you want to extend access list, then the system administrator can put users into or put out of different groups for control.

ii.

Mark and Helen first need to select the most ACL entry that best matches the process.

If the matching group entry for the file appears in these matching group entries, then select the matching group entry with the requested permissions. Conversely, if there is not a corresponding group of matching entries, access will be denied.

C

First, you can make your own password checker, which will filter as many passwords as possible that can be easily guessed by a dictionary attack. Then more algorithmically generate a more robust password that is difficult to guess.

The next step is to give the password file to a specific user, i.e., the only user in the system who has access to view it. This means that if a hacker needs to view the password file, he must first crack the user account that has access to the password file.

The two are used in combination to achieve the desired effect.

2.

A

OS Commanding:

Hacker inject the operating system command through an HTTP request to the web application by uploading malwares.

SQL Injection:

Hackers injecting commands with SQL syntax, using methods such as the comment character or "1==1", can read or modify the database, or corrupt the original SQL statement.

SSI Injection
Hackers are sending offensive code to the web server, which will be executed locally if the web server fails to filter it in time.

XPath Injection Attack
Hackers exploit the nature of XPath query elements to inject offensive data into a website or application to perform XPath queries on a website. This can often
bypass authentication or access information without the need for proper authorization.

B
The reason for sql injection is that, on the surface, it is due to splicing strings to form sql statements, not using sql pre-compilation, and binding variables. But the deeper reason is that the user input string is treated as a "sql statement" to execute.

For example, for union injection attack, if the program does not put the user input parameters into the SQL statement WITHOUT filtering, then it is likely to contain some illegal strings in the parameters.

We not only need to filter the parameters, but also need to pre-compile the function. Pre-compilation can check whether the statement is legitimate or offensive without running the statement.

In addition, we can do things like
String sql = "select * from test where id =" +id;
in such a way that the id can only be of type int

C
i.
The given code contians two statement
Statement 1: it is trying to obtain the information by using SELECT from the row “id” = 5, to get the username and email specifically.
Statement 2: If the username selected is ‘sa’, which stands for system administrator in this point, then this statement will execute 1/0 (1 divided by 0), which is a nonsensical arithmetic, that will forces the system to throw an error.

If this code gives an error in the end, attacker will know that the system administrator is running the database.

ii.
Pre-compilation of SQL statements. Pre-compilation checks statements for illegal characters without executing the code.

In particular, it filters for custom special symbols such as "/", ";", "<", etc., and terminates immediately with an error if found.

There is another way to check using 2 variables.

First parameterize the value, e.g. store all characters after WHERE in a variable a1

Then filter, extract, etc. on a1. Store the result in a2.

Compare a1 and a2, if the variables are not the same, then there is SQL injection. If the variables are different, then there is SQL injection.