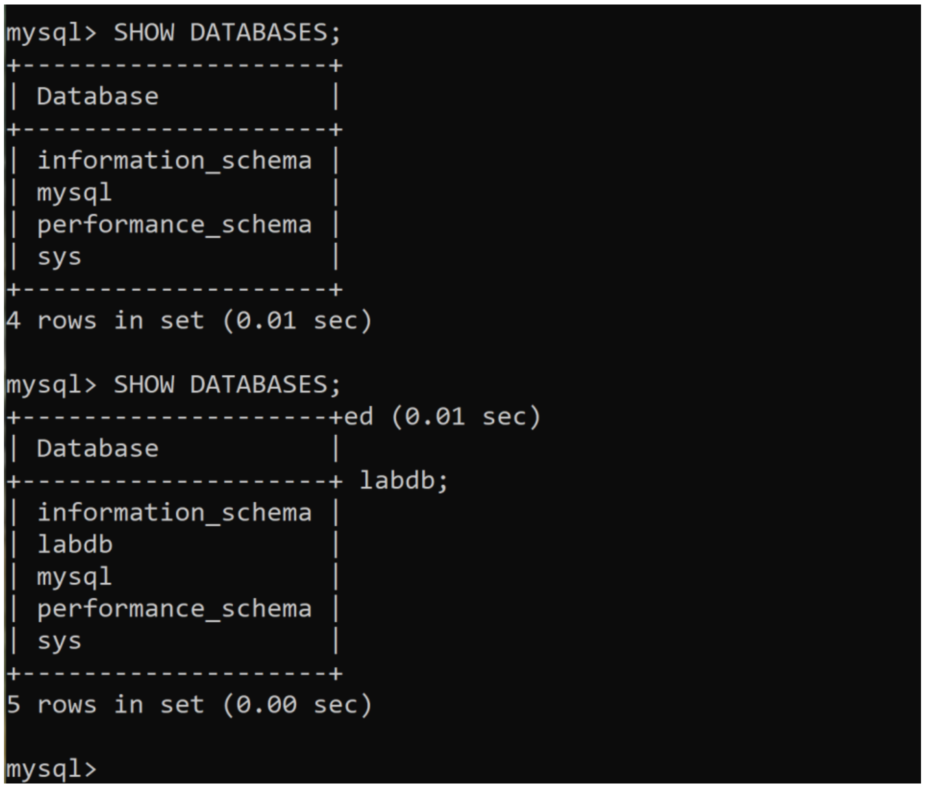
Lab 7: Database Security

PART 1: BASICS OF MYSQL

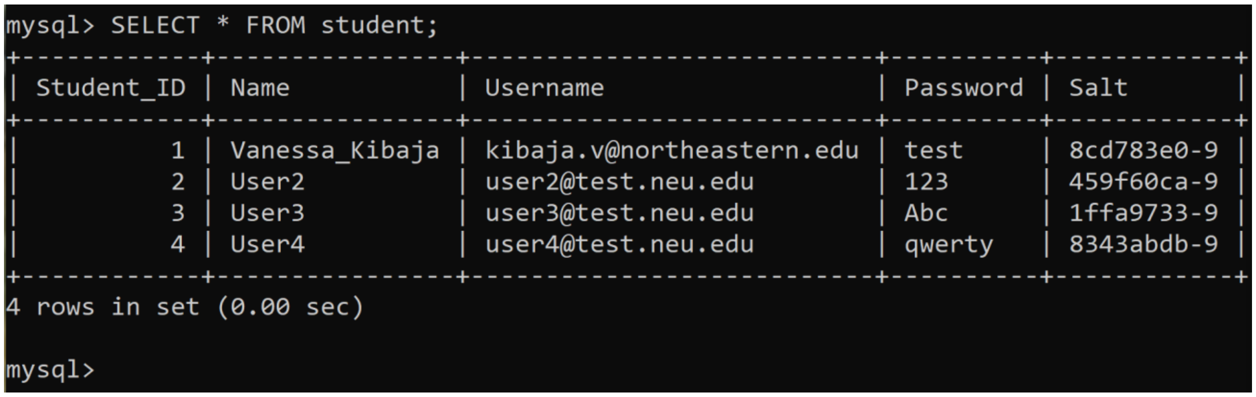
Pulled MySQL docker container and created a new database named labdb as seen in screenshot 1.



```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.01 sec)

mysql> SHOW DATABASES;
+--------------------+ed (0.01 sec)
| Database           |
+--------------------+ labdb;
| information_schema |
| labdb              |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql>
```

Screen Capture 1: labdb database

Created a table named student and inserted some values as seen in screenshot 2.

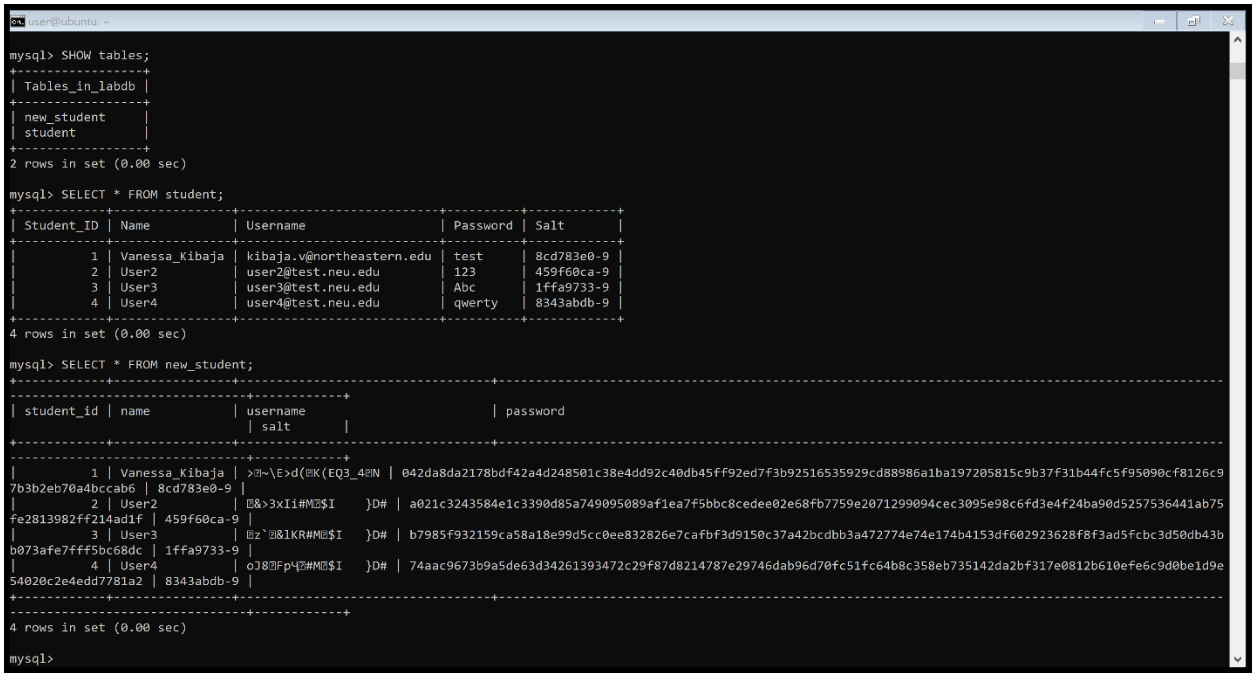


```
mysql> SELECT * FROM student;
+------------+----------------+---------------------------+----------+------------+
| Student_ID | Name           | Username                  | Password | Salt       |
+------------+----------------+---------------------------+----------+------------+
|          1 | Vanessa_Kibaja | kibaja.v@northeastern.edu | test     | 8cd783e0-9 |
|          2 | User2          | user2@test.neu.edu        | 123      | 459f60ca-9 |
|          3 | User3          | user3@test.neu.edu        | Abc      | 1ffa9733-9 |
|          4 | User4          | user4@test.neu.edu        | qwerty   | 8343abdb-9 |
+------------+----------------+---------------------------+----------+------------+
4 rows in set (0.00 sec)

mysql>
```

Screen Capture 2: student table

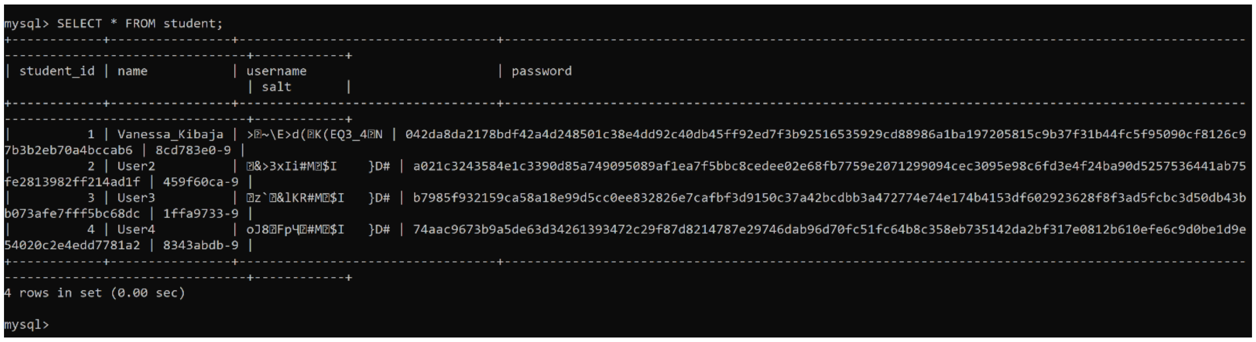
PART 2: ATTRIBUTE ENCRYPTION

Created new_student table with encryptions on username and password.



user@ubuntu: ~

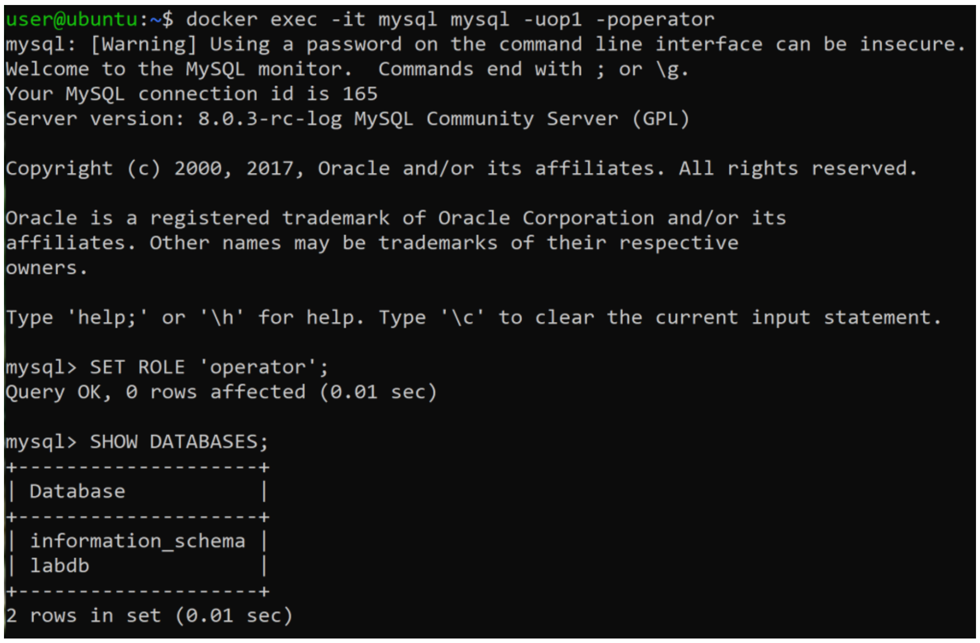
```
mysql> SHOW tables;
+----------------+
| Tables_in_labdb |
+----------------+
| new_student     |
| student         |
+----------------+
2 rows in set (0.00 sec)

mysql> SELECT * FROM student;
+------------+----------------+----------------------------+----------+------------+
| Student_ID | Name           | Username                   | Password | Salt       |
+------------+----------------+----------------------------+----------+------------+
|          1 | Vanessa_Kibaja | kibaja.v@northeastern.edu  | test     | 8cd783e0-9 |
|          2 | User2          | user2@test.neu.edu         | 123      | 459f60ca-9 |
|          3 | User3          | user3@test.neu.edu         | Abc      | 1ffa9733-9 |
|          4 | User4          | user4@test.neu.edu         | qwerty   | 8343abdb-9 |
+------------+----------------+----------------------------+----------+------------+
4 rows in set (0.00 sec)

mysql> SELECT * FROM new_student;
+------------+----------------+----------------------------------+-----------------------------------------------------------------------------------------------------------------------------------
--------------------------------+------------+
| student_id | name           | username                         | password
                                | salt       |
+------------+----------------+----------------------------------+-----------------------------------------------------------------------------------------------------------------------------------
--------------------------------+------------+
|          1 | Vanessa_Kibaja | >�~\E>d(�K(EQ3_4�N | 042da8da2178bdf42a4d248501c38e4dd92c40db45ff92ed7f3b92516535929cd88986a1ba197205815c9b37f31b44fc5f95090cf8126c9
7b3b2eb70a4bccab6 | 8cd783e0-9 |
|          2 | User2          | �&>3xIi#M�$I    }D# | a021c3243584e1c3390d85a749095089af1ea7f5bbc8cedee02e68fb7759e2071299094cec3095e98c6fd3e4f24ba90d5257536441ab75
fe2813982ff214ad1f | 459f60ca-9 |
|          3 | User3          | �z`�&lKR#M�$I   }D# | b7985f932159ca58a18e99d5cc0ee832826e7cafbf3d9150c37a42bcdbb3a472774e74e174b4153df602923628f8f3ad5fcbc3d50db43b
b073afe7fff5bc68dc | 1ffa9733-9 |
|          4 | User4          | oJ8�FpЧ�#M�$I   }D# | 74aac9673b9a5de63d34261393472c29f87d8214787e29746dab96d70fc51fc64b8c358eb735142da2bf317e0812b610efe6c9d0be1d9e
54020c2e4edd7781a2 | 8343abdb-9 |
+------------+----------------+----------------------------------+-----------------------------------------------------------------------------------------------------------------------------------
--------------------------------+------------+
4 rows in set (0.00 sec)

mysql>
```



```
mysql> SELECT * FROM student;
+------------+----------------+----------------------------------+-----------------------------------------------------------------------------------------------------------------------------------
--------------------------------+------------+
| student_id | name           | username                         | password
                                | salt       |
+------------+----------------+----------------------------------+-----------------------------------------------------------------------------------------------------------------------------------
--------------------------------+------------+
|          1 | Vanessa_Kibaja | >�~\E>d(�K(EQ3_4�N | 042da8da2178bdf42a4d248501c38e4dd92c40db45ff92ed7f3b92516535929cd88986a1ba197205815c9b37f31b44fc5f95090cf8126c9
7b3b2eb70a4bccab6 | 8cd783e0-9 |
|          2 | User2          | �&>3xIi#M�$I    }D# | a021c3243584e1c3390d85a749095089af1ea7f5bbc8cedee02e68fb7759e2071299094cec3095e98c6fd3e4f24ba90d5257536441ab75
fe2813982ff214ad1f | 459f60ca-9 |
|          3 | User3          | �z`�&lKR#M�$I   }D# | b7985f932159ca58a18e99d5cc0ee832826e7cafbf3d9150c37a42bcdbb3a472774e74e174b4153df602923628f8f3ad5fcbc3d50db43b
b073afe7fff5bc68dc | 1ffa9733-9 |
|          4 | User4          | oJ8�FpЧ�#M�$I   }D# | 74aac9673b9a5de63d34261393472c29f87d8214787e29746dab96d70fc51fc64b8c358eb735142da2bf317e0812b610efe6c9d0be1d9e
54020c2e4edd7781a2 | 8343abdb-9 |
+------------+----------------+----------------------------------+-----------------------------------------------------------------------------------------------------------------------------------
--------------------------------+------------+
4 rows in set (0.00 sec)

mysql>
```

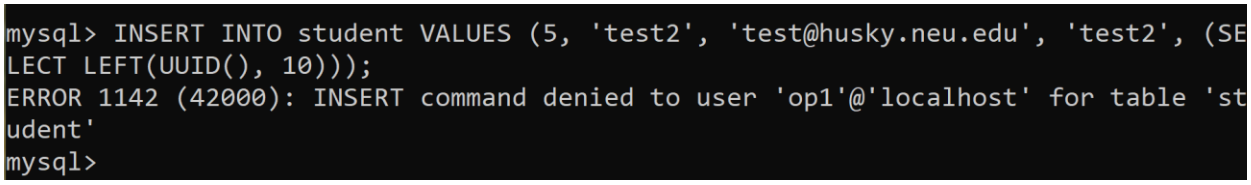
Screen Capture 3: student table

PART 3: DATABASE ACCESS CONTROL



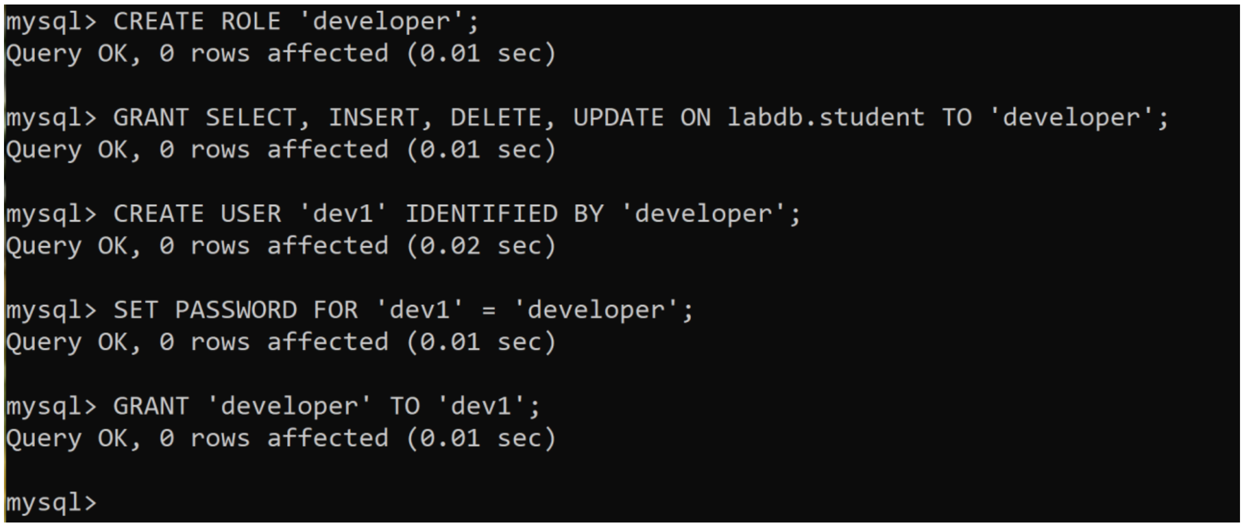
```
user@ubuntu:~$ docker exec -it mysql mysql -uop1 -poperator
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 165
Server version: 8.0.3-rc-log MySQL Community Server (GPL)

Copyright (c) 2000, 2017, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SET ROLE 'operator';
Query OK, 0 rows affected (0.01 sec)

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| labdb              |
+--------------------+
2 rows in set (0.01 sec)
```

Noticed that was only able to see these databases as an operator.



```
mysql> INSERT INTO student VALUES (5, 'test2', 'test@husky.neu.edu', 'test2', (SE
LECT LEFT(UUID(), 10)));
ERROR 1142 (42000): INSERT command denied to user 'op1'@'localhost' for table 'st
udent'
mysql>
```

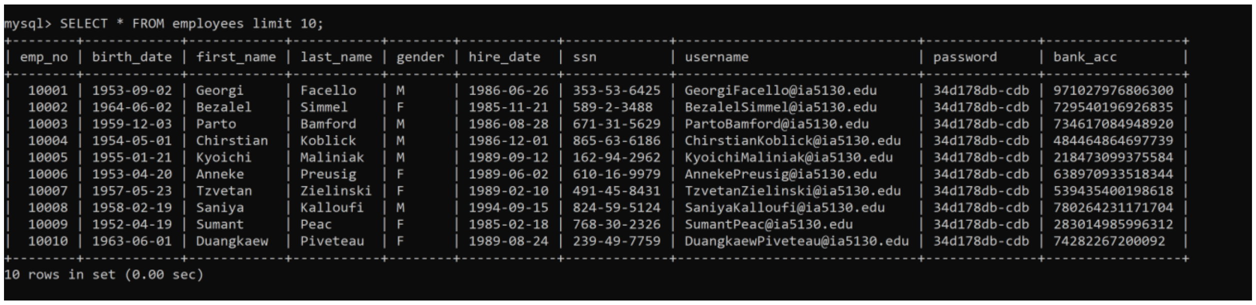
Screen Capture 4: Showing role op1 only has view access.



```
mysql> CREATE ROLE 'developer';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT SELECT, INSERT, DELETE, UPDATE ON labdb.student TO 'developer';
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE USER 'dev1' IDENTIFIED BY 'developer';
Query OK, 0 rows affected (0.02 sec)

mysql> SET PASSWORD FOR 'dev1' = 'developer';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT 'developer' TO 'dev1';
Query OK, 0 rows affected (0.01 sec)

mysql>
```

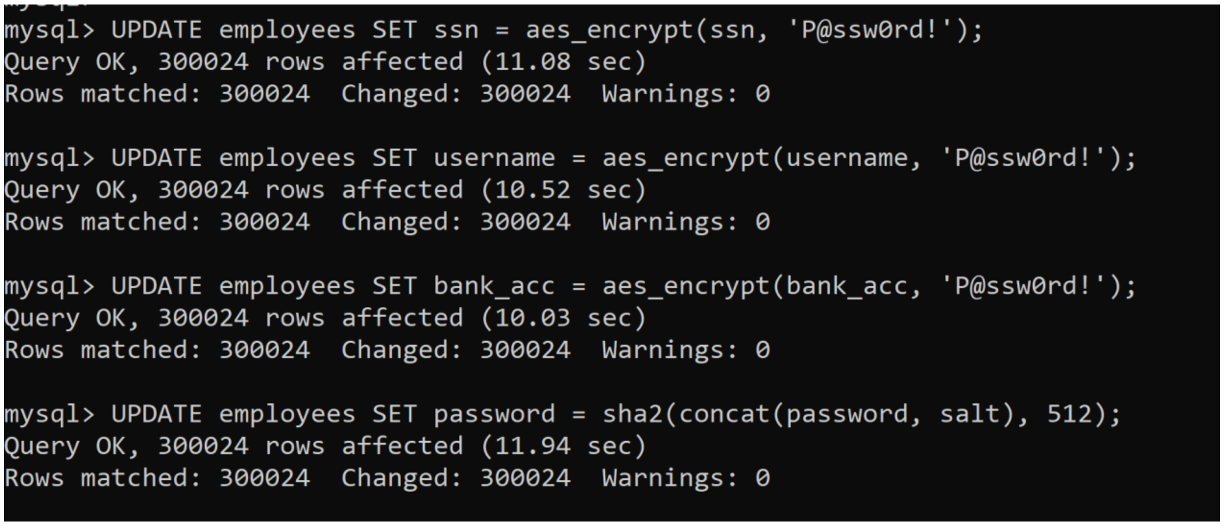
Role developer created.

PART 4: HARDENING MYSQL DATABASE



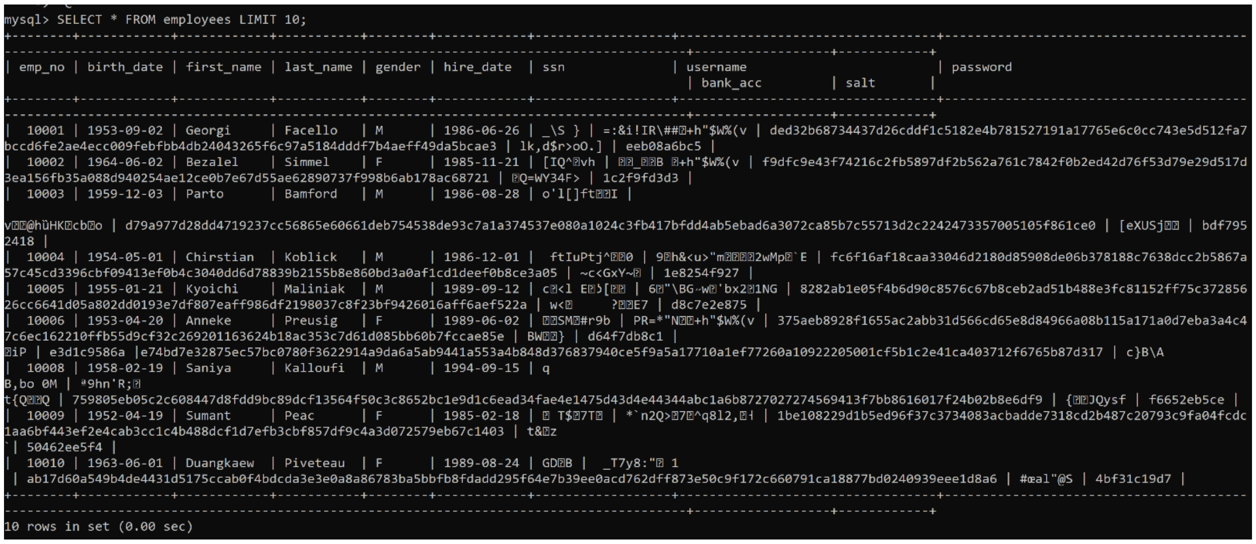
```
mysql> SELECT * FROM employees limit 10;
+--------+------------+------------+-----------+--------+------------+-------------+---------------------------------+--------------+-----------------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  | ssn         | username                        | password     | bank_acc        |
+--------+------------+------------+-----------+--------+------------+-------------+---------------------------------+--------------+-----------------+
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 | 353-53-6425 | GeorgiFacello@ia5130.edu        | 34d178db-cdb | 971027976806300 |
|  10002 | 1964-06-02 | Bezalel    | Simmel    | F      | 1985-11-21 | 589-2-3488  | BezalelSimmel@ia5130.edu        | 34d178db-cdb | 729540196926835 |
|  10003 | 1959-12-03 | Parto      | Bamford   | M      | 1986-08-28 | 671-31-5629 | PartoBamford@ia5130.edu         | 34d178db-cdb | 734617084948920 |
|  10004 | 1954-05-01 | Chirstian  | Koblick   | M      | 1986-12-01 | 865-63-6186 | ChirstianKoblick@ia5130.edu     | 34d178db-cdb | 484464864697739 |
|  10005 | 1955-01-21 | Kyoichi    | Maliniak  | M      | 1989-09-12 | 162-94-2962 | KyoichiMaliniak@ia5130.edu      | 34d178db-cdb | 218473099375584 |
|  10006 | 1953-04-20 | Anneke     | Preusig   | F      | 1989-06-02 | 610-16-9979 | AnnekePreusig@ia5130.edu        | 34d178db-cdb | 638970933518344 |
|  10007 | 1957-05-23 | Tzvetan    | Zielinski | F      | 1989-02-10 | 491-45-8431 | TzvetanZielinski@ia5130.edu     | 34d178db-cdb | 539435400198618 |
|  10008 | 1958-02-19 | Saniya     | Kalloufi  | M      | 1994-09-15 | 824-59-5124 | SaniyaKalloufi@ia5130.edu       | 34d178db-cdb | 780264231171704 |
|  10009 | 1952-04-19 | Sumant     | Peac      | F      | 1985-02-18 | 768-30-2326 | SumantPeac@ia5130.edu           | 34d178db-cdb | 283014985996312 |
|  10010 | 1963-06-01 | Duangkaew  | Piveteau  | F      | 1989-08-24 | 239-49-7759 | DuangkaewPiveteau@ia5130.edu    | 34d178db-cdb | 74282267200092  |
+--------+------------+------------+-----------+--------+------------+-------------+---------------------------------+--------------+-----------------+
10 rows in set (0.00 sec)
```

employees table before.



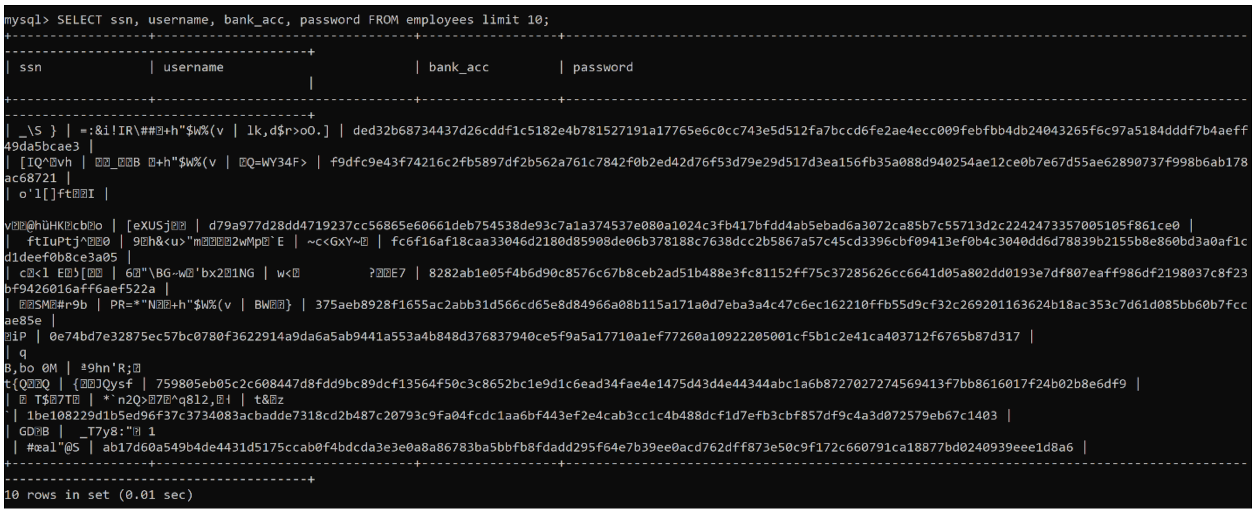
```
mysql> UPDATE employees SET ssn = aes_encrypt(ssn, 'P@ssw0rd!');
Query OK, 300024 rows affected (11.08 sec)
Rows matched: 300024  Changed: 300024  Warnings: 0

mysql> UPDATE employees SET username = aes_encrypt(username, 'P@ssw0rd!');
Query OK, 300024 rows affected (10.52 sec)
Rows matched: 300024  Changed: 300024  Warnings: 0

mysql> UPDATE employees SET bank_acc = aes_encrypt(bank_acc, 'P@ssw0rd!');
Query OK, 300024 rows affected (10.03 sec)
Rows matched: 300024  Changed: 300024  Warnings: 0

mysql> UPDATE employees SET password = sha2(concat(password, salt), 512);
Query OK, 300024 rows affected (11.94 sec)
Rows matched: 300024  Changed: 300024  Warnings: 0
```

Made these changes.



```
mysql> SELECT * FROM employees LIMIT 10;
+--------+------------+------------+-----------+--------+------------+-------------------+------------------------------------+---------------------------------
------------------------------------------------------------------------------------------+------------------+-----------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  | ssn               | username                           | password
                                                                                           | bank_acc         | salt      |
+--------+------------+------------+-----------+--------+------------+-------------------+------------------------------------+---------------------------------
------------------------------------------------------------------------------------------+------------------+-----------+
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 | _\S } | =:&i!IR\##⍰+h"$W%(v | ded32b68734437d26cddf1c5182e4b781527191a17765e6c0cc743e5d512fa7
bccd6fe2ae4ecc009febfbb4db24043265f6c97a5184dddf7b4aeff49da5bcae3 | lk,d$r>oO.] | eeb08a6bc5 |
|  10002 | 1964-06-02 | Bezalel    | Simmel    | F      | 1985-11-21 | [IQ^⍰vh | ⍰⍰_⍰⍰B ⍰+h"$W%(v | f9dfc9e43f74216c2fb5897df2b562a761c7842f0b2ed42d76f53d79e29d517d
3ea156fb35a088d940254ae12ce0b7e67d55ae62890737f998b6ab178ac68721 | ⍰Q=WY34F> | 1c2f9fd3d3 |
|  10003 | 1959-12-03 | Parto      | Bamford   | M      | 1986-08-28 | o'l[]ft⍰⍰I |

v⍰⍰@hüHK⍰cb⍰o | d79a977d28dd4719237cc56865e60661deb754538de93c7a1a374537e080a1024c3fb417bfdd4ab5ebad6a3072ca85b7c55713d2c2242473357005105f861ce0 | [eXUSj⍰⍰ | bdf795
2418 |
|  10004 | 1954-05-01 | Chirstian  | Koblick   | M      | 1986-12-01 |  ftIuPtj^⍰⍰0 | 9⍰h&<u>"m⍰⍰⍰⍰2wMp⍰`E | fc6f16af18caa33046d2180d85908de06b378188c7638dcc2b5867a
57c45cd3396cbf09413ef0b4c3040dd6d78839b2155b8e860bd3a0af1cd1deef0b8ce3a05 | ~c<GxY~⍰ | 1e8254f927 |
|  10005 | 1955-01-21 | Kyoichi    | Maliniak  | M      | 1989-09-12 | c⍰<l E⍰ↄ[⍰⍰ | 6⍰"\BG~w⍰'bx2⍰1NG | 8282ab1e05f4b6d90c8576c67b8ceb2ad51b488e3fc81152ff75c372856
26cc6641d05a802dd0193e7df807eaff986df2198037c8f23bf9426016aff6aef522a | w<⍰     ?⍰⍰E7 | d8c7e2e875 |
|  10006 | 1953-04-20 | Anneke     | Preusig   | F      | 1989-06-02 | ⍰⍰SM⍰#r9b | PR=*"N⍰⍰+h"$W%(v | 375aeb8928f1655ac2abb31d566cd65e8d84966a08b115a171a0d7eba3a4c4
7c6ec162210ffb55d9cf32c269201163624b18ac353c7d61d085bb60b7fccae85e | BW⍰⍰} | d64f7db8c1 |
⍰iP | e3d1c9586a |e74bd7e32875ec57bc0780f3622914a9da6a5ab9441a553a4b848d376837940ce5f9a5a17710a1ef77260a10922205001cf5b1c2e41ca403712f6765b87d317 | c}B\A
|  10008 | 1958-02-19 | Saniya     | Kalloufi  | M      | 1994-09-15 | q
B,bo 0M | ª9hn'R;⍰
t{Q⍰⍰Q | 759805eb05c2c608447d8fdd9bc89dcf13564f50c3c8652bc1e9d1c6ead34fae4e1475d43d4e44344abc1a6b87270272745694137bb8616017f24b02b8e6df9 | {⍰⍰JQysf | f6652eb5ce |
|  10009 | 1952-04-19 | Sumant     | Peac      | F      | 1985-02-18 | ⍰ T$⍰7T⍰ | *`n2Q>⍰7⍰^q8l2,⍰┤ | 1be108229d1b5ed96f37c3734083acbadde7318cd2b487c20793c9fa04fcdc
1aa6bf443ef2e4cab3cc1c4b488dcf1d7efb3cbf857df9c4a3d072579eb67c1403 | t&⍰z
`| 50462ee5f4 |
|  10010 | 1963-06-01 | Duangkaew  | Piveteau  | F      | 1989-08-24 | GD⍰B |  _T7y8:"⍰ 1
 | ab17d60a549b4de4431d5175ccab0f4bdcda3e3e0a8a86783ba5bbfb8fdadd295f64e7b39ee0acd762dff873e50c9f172c660791ca18877bd0240939eee1d8a6 | #œal"@S | 4bf31c19d7 |
+--------+------------+------------+-----------+--------+------------+-------------------+------------------------------------+---------------------------------
------------------------------------------------------------------------------------------+------------------+-----------+
10 rows in set (0.00 sec)
```

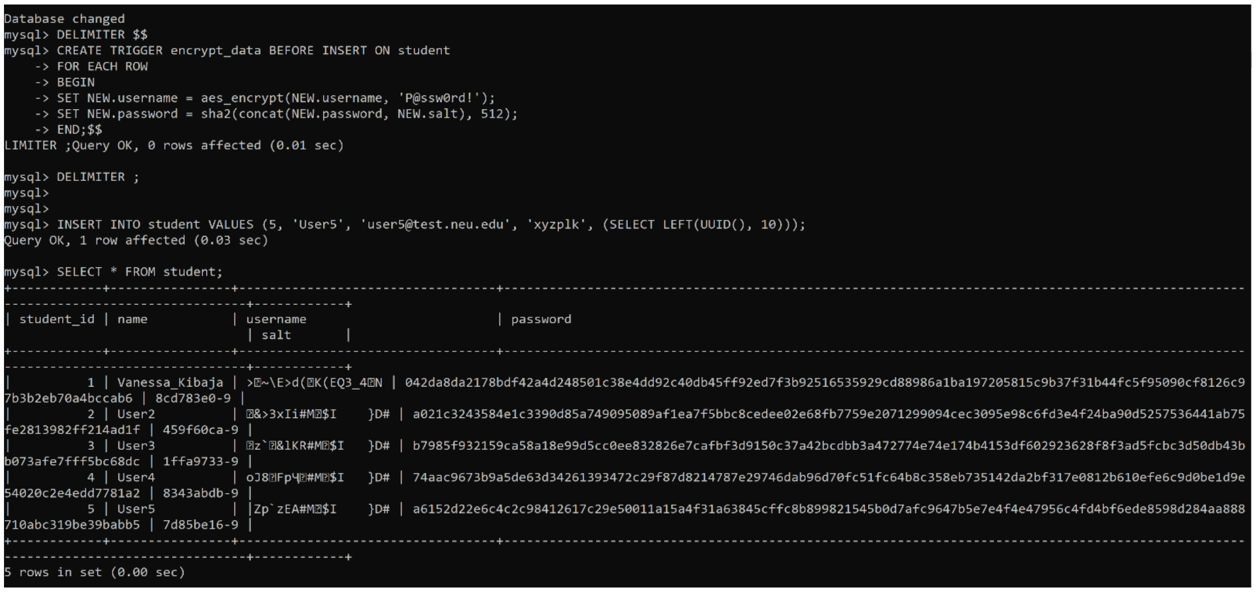
employees table after encrypting, hashing and adding salt column.



```
mysql> SELECT ssn, username, bank_acc, password FROM employees limit 10;
+------------------+------------------------------------+------------------+--------------------------------------------------------------------------------------------------------------------
----------------------------------------+
| ssn              | username                           | bank_acc         | password
                                        |
+------------------+------------------------------------+------------------+--------------------------------------------------------------------------------------------------------------------
----------------------------------------+
| _\S } | =:&i!IR\##�+h"$W%(v | lk,d$r>oO.] | ded32b68734437d26cddf1c5182e4b781527191a17765e6c0cc743e5d512fa7bccd6fe2ae4ecc009febfbb4db24043265f6c97a5184dddf7b4aeff
49da5bcae3 |
| [IQ^�vh | ��_��B �+h"$W%(v | �Q=WY34F> | f9dfc9e43f74216c2fb5897df2b562a761c7842f0b2ed42d76f53d79e29d517d3ea156fb35a088d940254ae12ce0b7e67d55ae62890737f998b6ab178
ac68721 |
| o'l[]ft��I |

v��@hüHK�cb�o | [eXUSj�� | d79a977d28dd4719237cc56865e60661deb754538de93c7a1a374537e080a1024c3fb417bfdd4ab5ebad6a3072ca85b7c55713d2c2242473357005105f861ce0 |
|  ftIuPtj^��0 | 9�h&<u>"m����2wMp�`E | ~c<GxY~� | fc6f16af18caa33046d2180d85908de06b378188c7638dcc2b5867a57c45cd3396cbf09413ef0b4c3040dd6d78839b2155b8e860bd3a0af1c
d1deef0b8ce3a05 |
| c�<l E�ʖ[�� | 6�"\BG~w�'bx2�1NG | w<�         ?��E7 | 8282ab1e05f4b6d90c8576c67b8ceb2ad51b488e3fc81152ff75c37285626cc6641d05a802dd0193e7df807eaff986df2198037c8f23
bf9426016aff6aef522a |
| ��SM�#r9b | PR=*"N��+h"$W%(v | BW��} | 375aeb8928f1655ac2abb31d566cd65e8d84966a08b115a171a0d7eba3a4c47c6ec162210ffb55d9cf32c269201163624b18ac353c7d61d085bb60b7fcc
ae85e |
�iP | 0e74bd7e32875ec57bc0780f3622914a9da6a5ab9441a553a4b848d376837940ce5f9a5a17710a1ef77260a10922205001cf5b1c2e41ca403712f6765b87d317 |
| q
B,bo 0M | ª9hn'R;�
t{Q��Q | {��JQysf | 759805eb05c2c608447d8fdd9bc89dcf13564f50c3c8652bc1e9d1c6ead34fae4e1475d43d4e44344abc1a6b8727027274569413f7bb8616017f24b02b8e6df9 |
| � T$�7T� | *`n2Q>�7�^q8l2,�꜡ | t&�z
`| 1be108229d1b5ed96f37c3734083acbadde7318cd2b487c20793c9fa04fcdc1aa6bf443ef2e4cab3cc1c4b488dcf1d7efb3cbf857df9c4a3d072579eb67c1403 |
| GD�B |  _T7y8:"� 1
 | #œal"@S | ab17d60a549b4de4431d5175ccab0f4bdcda3e3e0a8a86783ba5bbfb8fdadd295f64e7b39ee0acd762dff873e50c9f172c660791ca18877bd0240939eee1d8a6 |
+------------------+------------------------------------+------------------+--------------------------------------------------------------------------------------------------------------------
----------------------------------------+
10 rows in set (0.01 sec)
```

Screen Capture 5: employees table showing encrypted values of ssn, username, bank account, and password.

PART 5: AUTOMATIC ENCRYPTION



```
Database changed
mysql> DELIMITER $$
mysql> CREATE TRIGGER encrypt_data BEFORE INSERT ON student
    -> FOR EACH ROW
    -> BEGIN
    -> SET NEW.username = aes_encrypt(NEW.username, 'P@ssw0rd!');
    -> SET NEW.password = sha2(concat(NEW.password, NEW.salt), 512);
    -> END;$$
LIMITER ;Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
mysql>
mysql> INSERT INTO student VALUES (5, 'User5', 'user5@test.neu.edu', 'xyzplk', (SELECT LEFT(UUID(), 10)));
Query OK, 1 row affected (0.03 sec)

mysql> SELECT * FROM student;
+------------+----------------+----------------------------------+------------------------------------------------------------------------------------------------------------------------------------
------------------------------+------------+
| student_id | name           | username                         | password
                              | salt       |
+------------+----------------+----------------------------------+------------------------------------------------------------------------------------------------------------------------------------
------------------------------+------------+
|          1 | Vanessa_Kibaja | >�~\E>d(�K(EQ3_4�N | 042da8da2178bdf42a4d248501c38e4dd92c40db45ff92ed7f3b92516535929cd88986a1ba197205815c9b37f31b44fc5f95090cf8126c9
7b3b2eb70a4bccab6 | 8cd783e0-9 |
|          2 | User2          | �&>3xIi#M�$I    }D# | a021c3243584e1c3390d85a749095089af1ea7f5bbc8cedee02e68fb7759e2071299094cec3095e98c6fd3e4f24ba90d5257536441ab75
fe2813982ff214ad1f | 459f60ca-9 |
|          3 | User3          | �z`�&lKR#M�$I   }D# | b7985f932159ca58a18e99d5cc0ee832826e7cafbf3d9150c37a42bcdbb3a472774e74e174b4153df602923628f8f3ad5fcbc3d50db43b
b073afe7fff5bc68dc | 1ffa9733-9 |
|          4 | User4          | oJ8�FpЧ�#M�$I   }D# | 74aac9673b9a5de63d34261393472c29f87d8214787e29746dab96d70fc51fc64b8c358eb735142da2bf317e0812b610efe6c9d0be1d9e
54020c2e4edd7781a2 | 8343abdb-9 |
|          5 | User5          | |Zp`zEA#M�$I    }D# | a6152d22e6c4c2c98412617c29e50011a15a4f31a63845cffc8b899821545b0d7afc9647b5e7e4f4e47956c4fd4bf6ede8598d284aa888
710abc319be39babb5 | 7d85be16-9 |
+------------+----------------+----------------------------------+------------------------------------------------------------------------------------------------------------------------------------
------------------------------+------------+
5 rows in set (0.00 sec)
```

Screen Capture 6: data was encrypted automatically.

Link to Docker Hub

docker pull vanessakibaja/lab7_database_security:latest

QUESTIONS

1. **What is Role Based Access Control? What are the benefits of RBAC? List other authorization models.**
   Role Based Access Control is a restriction of access based solely on a person's role at an organization. Employees can only have access to things that will enable them to perform their duties effectively (Zhang). This has become very common among many organizations as it helps secure sensitive data in a company.

   Examples:
   a. Only administrators will have access to things that requires administrative privileges.
   b. entry-level employees cannot get access to confidential information since they do not need them to perform their daily duties.
   c. Contactors hires for different roles are assigned access based on their roles.

   Benefits:
   a. <u>It maximizes operational efficiency</u>. Since this authorization model is applicable to all levels within the organization, it becomes very efficient to the organization. It ensures all users have access to the things needed for them to do their jobs as this is setup automatically when a user is assigned to a particular role.
   b. <u>Reduces IT and administrative work</u>. When an employee is hired or re-hired or promoted, administrators only have to switch the employee's role and the employees will have access to things needed based on their role. Administrators don't have to manually assign access and do all the paper work necessary for them.
   c. <u>Decreases the risk of breaches and data leakage (Petters)</u>. With limited access, a hacker can only do limited stuff to that of the associated hacked account role.
   d. <u>Improving compliance</u>. Enables organizations to meet regular requirements for privacy and confidentiality (Zhang). Since it is clear to see how data is being accessed and utilized via this authorization model.

   Other authorization models include:
   DAC – Discretionary Access Control, MAC – Mandatory Access Control, and ABAC Attribute-Access Control.

2. **What are some attacks on passwords? How to defend against such attacks?**
   "Passwords are the main cause of most data breaches" as emphasized by the article The Cost of Passwords (The Cost of Passwords). Common password attacks include:
   a. <u>Phishing:</u> common social engineering attack where an attacker imposes as a legitimate party in hopes that the victim will click on a link sent via email and sign-in using their credentials.

To avoid such attacks, users have to double check who sent the emails and the source of the email sent. Be alert when they ask you to provide sensitive information such as sign-in credentials, ID numbers, SSN etc.

b. man-in-the-middle: This is when an attacker interrupts, or intercepts two people or systems that are communicating back and forth. The attacker can then view various information shared between the two parties including passwords.

   To avoid such attacks, communication channels should use encryption, as well as two-factor authentication to make access to password less powerful.

c. brute force: This is when an attacker tries to login to a user account with all possible password combinations. All the attacker does is to guess multiple passwords.

   To avoid such attack, users should use complex passwords, require multi-factor authentication, limit failed login attempts, and limit rot access.

d. Dictionary: An attacker uses a list of common passwords collectively called dictionary. This can also include a combination of personal information such as birth dates and puppies' names.

   To avoid such attacks, users should use unique and strong passwords, administrators should set rules to lock account after too many password failures.

3. **Write a short description on Bcrypt, Scrypt and Argon2 and PBKDF2 algorithms.**

   These are slow-hashing functions used when hashing passwords.

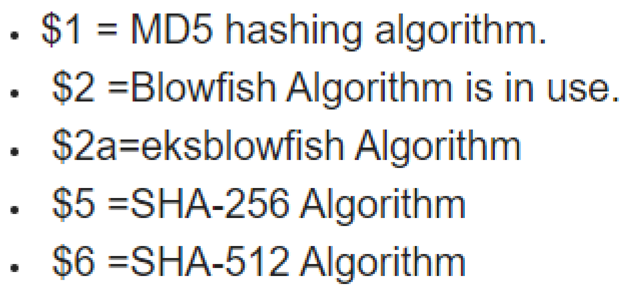
   a. Bcrypt and Scrypt (an updated version of Bcrypt): are hash algorithms used to store passwords but with the goal of making it harder to run the algorithm. Thus, presumably should take longer for an attacker to perform brute force attack. However, this was proven insignificant with specialized chips such as GPUs that runs faster than a regular computer. Makin these two algorithms insecure. Additionally, since the algorithm takes longer to compute, it wastes servers' resources.
   b. Argon2: is the hashing algorithm that overcame the fear of an attacker using larger GPU systems. It is stated to better than Bcrypt, Scrypt, and pbkdf2 with the right configurations. One can configure it using user's password, salt, iterations, memory size, paralleslism – number of threads, and output key length (Nakov argon2).
   c. PBKDF2: is a hashing algorithm that is resistant to dictionary attacks and rainbow table attacks as it uses other has-functions such as SHA-256 to calculate HMAC (Nakov pbkdf2).

4. **Explain how passwords are stored in Linux. (location, encryption algorithm, salt and file permissions)**

   Passwords in linux are stored in "/etc/shadow" file that is only accessible by root. This file has a very interesting permissions. Only the root can read it. Even root users can not make any changes to it since they have no permissions (Pillai).

The file saves passwords in a one-way hashed process. Such that a password ends up with 3 main fields: hashing algorithm used, the salt value, and hash value (salt + user password).

a. Algorithms: In Linux, if the first parameter in a shadow file is $6, we users understand that SHA-512 Algorithm was used as shown on the image below.



- $1 = MD5 hashing algorithm.
- $2 =Blowfish Algorithm is in use.
- $2a=eksblowfish Algorithm
- $5 =SHA-256 Algorithm
- $6 =SHA-512 Algorithm

Hash Algorithms

b. Salt: this is a random generated data used to combine with user password to make the auto-generated password hash strong. Thus, if an attacker gets a hold of the hashed value, it would be difficult to restore the password as there is a random salt combined.

c. Hash value: Uses a specific algorithm to generate a fixed value after combining user’s password along with the random generated salt.

References

Nakov, Svetlin. "MAC and Key Derivation." Nakov, Svetlin. *Practical Cryptography for Developers*. Sofia, 2018. Web. <https://cryptobook.nakov.com/mac-and-key-derivation/argon2>.

Petters, Jeff. "What is Role-Based Access Control (RBAC)?" 11 Nov 2020. *Varonis.com.* Web. 06 April 2021. <https://www.varonis.com/blog/role-based-access-control/>.

Pillai, Srath. "How are passwords stored in Linux (Understanding hashing with shadow utils)." 24 April 2013. *slashroot.in.* Web. 06 April 2021. <https://www.slashroot.in/how-are-passwords-stored-linux-understanding-hashing-shadow-utils>.

The Cost of Passwords. 23 Sept 2020. Web. 06 Apr 2021. <https://blog.acceptto.com/the-cost-of-passwords>.

Zhang, Ellen. "What is Role-Based Access Control (RBAC)? Examples, Benefits, and More." 01 Dec 2020. *Digital Guardian.* Web. 06 04 2021. <https://digitalguardian.com/blog/what-role-based-access-control-rbac-examples-benefits-and-more>.