A Novel SQL Injection Prevention Technique Using Data Hashing

MSc Cyber Security
Internship Paper

Slim Khili
Student ID:x18123155

School of Computing
National College of Ireland

Supervisor: Mr.Vikas Sahni

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Slim Khili
Student ID:	x18123155
Programme:	Cyber Security
Year:	2019
Module:	MSc Internship
Supervisor:	Mr Vikas Sahni
Submission Due Date:	12/08/2019
Project Title:	A Novel SQL Injection Prevention Technique Using Data Hashing
Word Count:	4441
Page Count:	17

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	12/08/2019

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

A Novel SQL Injection Prevention Technique Using Data Hashing

Slim Khili
x18123155
MSc Cyber Security

Abstract

SQL Injection (SQLI) is one of the most discussed topics when it comes to web application security. Attackers use this technique to break into the database by gaining unauthorized access. This work presents a novel technique to prevent SQLI attacks based on cryptography by applying a more secure encryption mechanism.The previously commonly deployed encryption algorithms are md5, SHA family and RIPEMD; however they are considered to be weak and vulnerable and they should not be used anymore. This new method aims to prevent SQLI by securing the database with the more secure encryption algorithm, bcrypt. The key motivation for this choice is that bcrypt is generally resistant to attacks and offers the best security. This paper demonstrates the strength of bcrypt in defending against SQLI.

1 Introduction

Data security is an important aspect in the world of the cybersecurity, especially for organization, business also personal, hence securing data from attacker is a challenge especially with the variety of threats affecting web applications such as Denial of Service (DoS), Structured Query Language (SQL) injection, Cross-site Scripting (XSS) ,Remote File Inclusion attacks etc. SQLI is one of the severe threats to web application security; since long time and according to OWASP [1] SQLI ranked in the top 10 most critical web application security risks[1]. Furthermore, vulnerabilities continuously get reported and the problem persists today. SQLI is a common technique used by attackers. The scenario of this attack is an attacker sends injection code to the database to be executed. This can allow the attacker to easily gain access to all data stored in the database. If the web application fails to prevent SQLI this can lead to a security breach and loss of sensitive data. Previously many methods were proposed to address the SQL injection threat such as defensive programming practice stored procedure and white input validation prepared statement and character escaping[2]. However all of them have some limitations and challenges. Some other proposed solution based on the science of cryptography, encryption could prevent the SQLI attack only if the encryption is strong enough, otherwise there is

[1]https://www.owasp.org/index.php/Category:OWASP$_{T}op_{T}en_{P}roject$

[2]https://www.owasp.org/index.php/SQL$_{I}njection_{P}revention_{C}heat_{S}heetDefense_{O}ption_{4}$ $:_{E}$ $scaping_{A}ll_{U}ser_{S}upplied_{I}nput$

a high probability data can be decrypted and stolen (for instance SHA and md5 are considered broken and should never be used,Dawson (2006). This work seeks solution based on cryptography but proposes a more secure encryption mechanism which is bcrypt which is based on the Blowfish symmetric block cipher cryptographic algorithm and introduces a novel security factor[3], which allows to determine how expensive the hash function will be. Moreover,unlike other algorithms bcrypt has the strength to combine salt and the hash and use a technique called Key Stretching to overcome and impede various attacks such as the dictionary attack and rainbow tables.

2 Related Work

Over the last few years SQLI has been showing a steadily increasing trend in the scale of the attacks. As of today, OWASP [4] (Open Web Application Security Project) ranks SQL injection as number one on its list of top 10 most critical web application security risks.

SQL injections can occur when the web application does not have a proper mitigation technique. The injection code sent from the client will be executed on the database, the attacker's commands and can allow attackers to get access to data and read, update, alter, or delete sensitive information stored in an applications database, which is very harmful for users and especially for organizations.

To prevent SQL injection attacks, it is very important to know the different types of SQLI and how they occur.

2.1 The different types of SQL attacks

Below is the different types of SQL attacks.

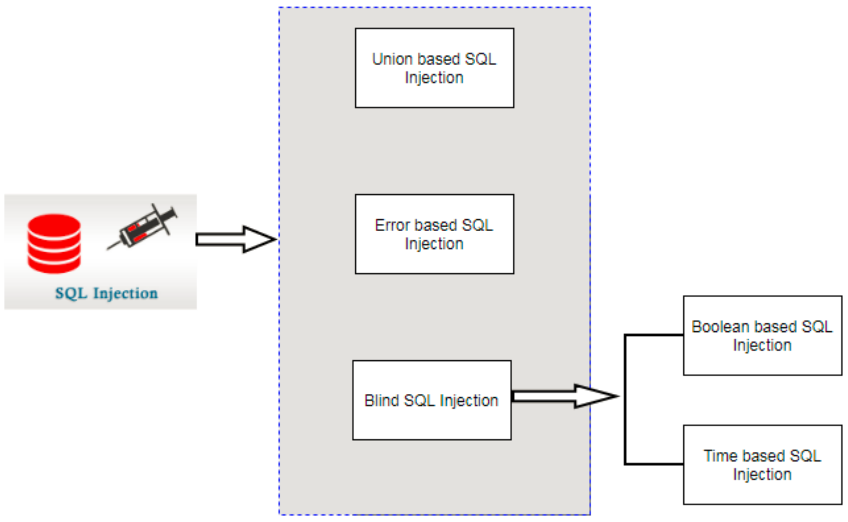

SQL Injection
Union based SQL Injection
Error based SQL Injection
Blind SQL Injection
Boolean based SQL Injection
Time based SQL Injection

Figure 1: Types of SQL attacks

[3]https://thehackernews.com/2014/04/securing-passwords-with-bcrypt-hashing.html
[4]https://www.owasp.org/index.php/Category:OWASP$_{Top}Top_{T}en_{P}roject$

2.1.1 Union Based SQL Injection

The union operator is used in SQL to join two statement. The union-based SQL injection exploits this technique by using the union operator; the concept is to enter invalid data in the correct field and add the union operator at the beginning of the query. By using the union keyword the attacker can run various actions such as altering, viewing or even deleting the entire data in the database. Mishra (2019).

2.1.2 Error Based SQL Injection

The concept of this attack is to deliver invalid input to the database via the input field in order to make an error occur in database. This is done by manipulating the database to execute malicious commands which will result in an error. From the error generated from the database the attacker can gather some information and use this information to exploit the database.

2.1.3 Blind SQL Injection

In this type of attack, the attacker tries to communicate with the database to get some useful information and based on what he manages to get, he decides what further action to carry out, Compared to the previous mentioned methods (2.1.1,2.1.2), this type of attack is considered the most difficult type as there is no available information about the database. Mishra (2019) Within this category of attack,there are two approaches,time-based SQL injection attacks and boolean based SQL injection.Below more details about them:

- **Time based SQL injection attacks**
  This consists of sending an SQL query to the database which will force the database to be on hold for specified amount of time, before it replies back. Based on response time attacker can know if the result of the query is true or false. Usually the http response will be returned immediately or with delay.

- **Boolean based SQL injection**
  This consists of sending an SQL query to the database in order to force the application to return different result which can be either true or false. Depending on the result, http response will be same or will be modified. On this attack, the attacker needs to enumerate the database character by character which is time consuming and this attack is slow and laborious.

2.2 Previous mitigation techniques

This thesis focuses on prevention of SQLI by using a solution based on cryptography mechanisms. This section reviews some of those works and conclude look on their pros and cons.

Johari and Sharma (2012) in their paper A Survey On Web Application Vulnerabilities (SQLIA,XSS) Exploitation and Security Engine for SQL Injection, they present the various SQL attack methodolgies and highlight the proposed mitigation techniques and the weakness, they found in all previous solutions which have have some vulnerability and weakness. They conclude by stating that it is very important that the developers use good coding techniques for websites as they serve as the first line of defense.

Temeiza et al. (2017) proposed a new solution to prevent SQLI based on cryptography and syntax awareness. The methodology proposed in this work was composed of 2 stages. The first stage consists on extracting the values of the query attributes then hashing the query. The second stage consist on comparing the hash values with the normal query hash value which was extracted. Based on the result of the comparison if the two hash values are different so the SQL query will be rejected. Otherwise if they have the same hash value the SQL query will be executed. The authors of this paper proved that the performance of SHA-1 is the best when compared to MD5 and SHA-512. However in other research Dawson (2006) and Savage (n.d.) prove that even SHA-1 is vulnerable to collusion attack.

Mehar Sood (2017) proposed another new technique to prevent SQLI by using two different encryption types single (AES,MD5) and double which combine the two of them in order to ensure a high level of security. The technique consists of encrypting the data by using algorithms then comparing the time taken for each one to find out the faster ,then they test the efficacity of these algorithms to protect from the SQLI attacks such as Bypass authentication, Unauthorized knowledge of database, Injected Additional query and second order SQL Injection
The idea of combining MD5 and AES was helpful and it proves more powerful then using the single encryption.However the double encryption is based on two weak cryptography algorithms still vulnerable to SQLI.Khili (2019)
K. Rajeswari (2016) proposed a technique to prevent various SQLI attacks using RC4 algorithm and Blowfish. This technique relies on query tokenization; the principle of this technique consists of taking the query as input then converting it into separate tokens. By detecting some special character as space, single quotes and double dashes, the token is generated.Once one of this character appears. The execution of the tokenization process comprises from 5 stages:

- Stage 1: Analyse the input in order to find and replace the character whish may lead to an attack and present a risk for the injection.

- Stage 2: Search for query contain one of the below characters:
  −Space.
  −Single quote (').
  −Double quote (").

- Stage 3: Divide the input query to different tokens.

- Stage 4: Save the generated token of the previous stage into dynamic table.

- Stage 5: Once all this stage done successfully, the dynamic token table and encrypted input are forwarded to the server.

Over the last 20 years plenty of hash functions have been deployed. MD5 and SHA-1 are the most popular used hash functions with many applications, nevertheless neither of them are good enough and they are not secure to be implemented in applications especially as they are susceptible to attacks.

The MD5 hash function was created in 1991 by Rivets and it is based on Merkle/Damgard. The main concept of MD5 consists of taking an arbitrary message length less than 264 bits, then starting the compression in order to make the size into 128-bit hash value, then the message M will be divided into a block of 512-bits size.If the message length is not multiple of 512-bits, so a single 1 bit followed by 0s will be added in order to make the length of the message to multiple of 512-bits. Savage (n.d.)

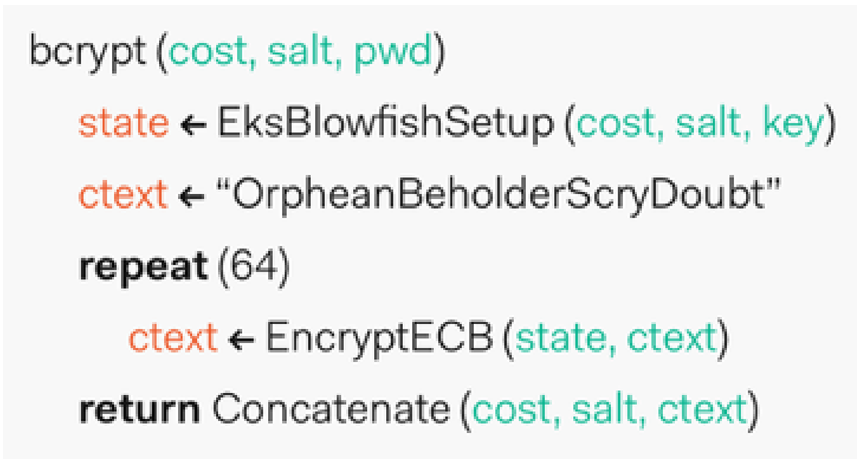
John Black (2016) provided a detailed explanation on how the MD5 attack is performed.Also Wang Xiaoyun and Yu (2004)found the first practical collision on MD5. Here is an overview of how the attack is carried out. The attack used on MD5 was based an approach of cryptoanalysis and 2 blocks of collision used with a complexity of 2.39 hashing operation for the algorithm; the technique used was based on finding a value approximately close to the first block message.Once this value is found then it will be converted to collison.Then the same process is applied to the second block message.
After Wang Xiaoyun and Yu (2004) published their result on the finding on MD5 many proposed solutions to enhance finding the collision on the algorithm Klima (2005)and Satoh. (2005)

Based on the latest collision on md5 should not be used anymore in any application Klima (2005) , Wang and Yu (2005).

2.3 Bcrypt

Ertaul and Kaur (n.d.) Bcrypt is one of the most strongest hashing algorithm is based on Blowfish cipher. Bcrypt is able to prevent many threats and password hacking and is relies on procedure called EKSBlowfish which has powerful effect to make the password encryption more strong by using salt to create non-recurrent hash[5].

2.3.1 Bcrypt Algorithm



```
bcrypt (cost, salt, pwd)
    state ← EksBlowfishSetup (cost, salt, key)
    ctext ← "OrpheanBeholderScryDoubt"
    repeat (64)
        ctext ← EncryptECB (state, ctext)
    return Concatenate (cost, salt, ctext)
```

Figure 2: Bcrypt Algorithm

[5]https://asecuritysite.com/encryption/bcrypt

Ertaul and Kaur (n.d.) The execution of bcrypt represented in 2 stages:

- Stage1:

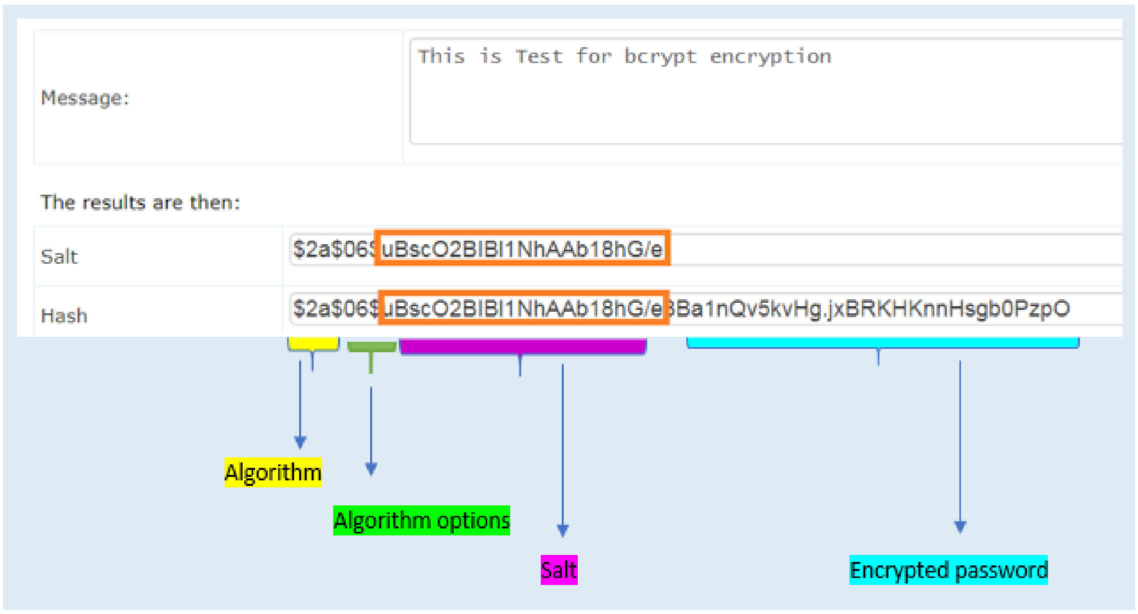
  –The function EksBlowfishSetup called which take as parameter (cost, the salt, and the password).
  –The password used as primary key.
  –Bcrypt run an expensive key to perform a key derivation in order to derive set of subkeys from primary key.
  –In case the password to short then key stretching process started in order to make the password large.

- Stage2:

  –OrpheanBeholderScryDoubt is encrypted at 64 times from the previous phase to the particular state using Eksblowfish in ECB.
  –Finally, the output generated is composed from the generated combined with the encrypted message.

2.3.2 Bcrypt encryption

Below is the structure of the bcrypt encryption.



Message:	This is Test for bcrypt encryption

The results are then:

Salt	$2a$06$uBscO2BIBl1NhAAb18hG/e
Hash	$2a$06$uBscO2BIBl1NhAAb18hG/e3Ba1nQv5kvHg.jxBRKHKnnHsgb0PzpO

Figure 3: Bcrypt encryption

3 Research Methodology

In this section we introduce novel solutions which are originally inspired from the previous researches after studying the previous researches and analyzing the different proposed solution, we can conclude that none of them manage to prevent the SQL attack and they have some vulnerability and weaknesses. The proposed model is based on implementing the current powerful encryption algorithm. The design of bcrypt is the reason behind it success to prevent various attacks such dictionary attack and brute force attack. The logarithmic bar chart[6] below, illustrates the required time to perform brute force attack, the same test carried out on different hashing algorithm, MD5,SHA1,RipeMD-160,SHA-256,SHA-512,Wirlpool,scrypt and bcrypt.
The test result shows that MD5,SHA-1,SHA-256,SHA-512,SHA-3 just from [2 to 49] seconds can be attacked however to brute force bcrypt the time required is more than 3 years. This test proves how secure and resistant to attack bcrypt is.Bcrypt has two processes, salting and hashing below are more details about them:

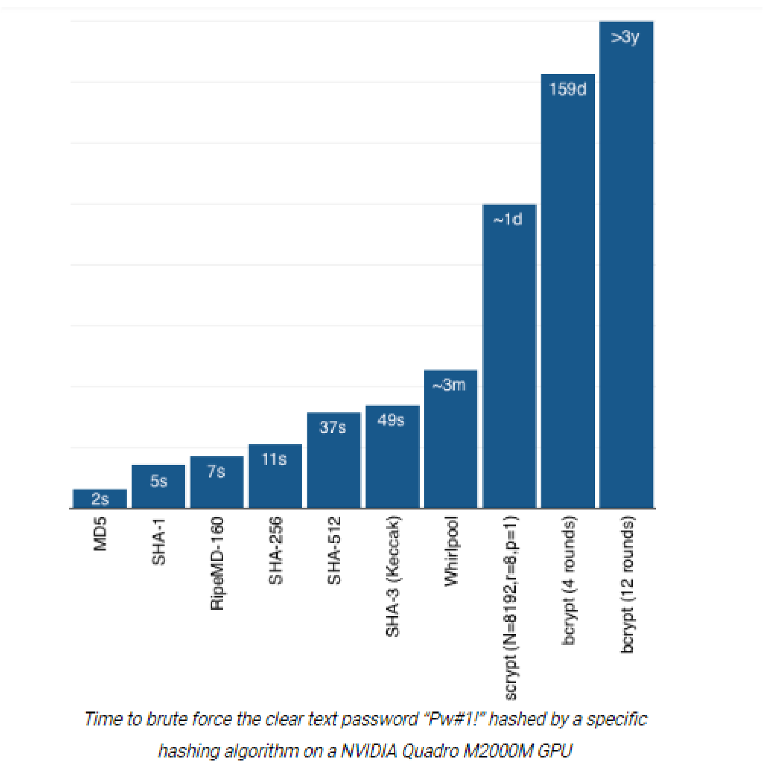


Figure 4: Time to brute force the clear text password

3.1 Salted password hashing

The number of operating websites on the world wide web is increasing every second.There are over 1 billion websites and this number keep growing. They had different design and developed in different programming languages and many of them they store sensitive data of users. How data is stored in the database and protected against data breach is the

[6]https://www.novatec-gmbh.de/en/blog/choosing-right-hashing-algorithm-slowness/

most important thing we need to start thinking about. So salted password hashing could be considered best practice to secure passwords and sensitive data.

3.2 Password hashing

Hashing is a one-way function that means the original text cannot be retrieved from hashed string.That is why it is called one way function and is used to create digital signature. The property of one way hashing function if there is for example two inputs slightly different, output will completely different for both of them. This feature it is very important as it can guarantee a high level of security.Levent Ertaul (n.d.).

In order to demonstrate the security and efficacy of the proposed model the web application has been developed linked to database for simulation and testing,

The web application developed it is an employee management system developed for demonstration purpose, the web application has login page which requires login name and password, after user manage to login successfully there is different pages as add (department, designation, leave, event, etc.) and view (department, designation, leave, event, etc)

The focus will be on the registration page, where there is an option to ask user which encryption to be used (md5 or bcrypt) based on the selected choice of encryption mechanism the password will be encrypted then save it in the database. The purpose behind that is to compare both encryption mechanism and highlight the weakness of the previous used encryption mechanism and prove the strength of the proposed new encryption mechanism,

The proposed methodology consists of checking the query before proceeding with it; there are some criteria that need to be met which aim to ensure that the query is valid, and it does not contain any SQL injection, for the encryption otherwise the encryption will not happen.

The general workflow for employee registration and authentication in a hash-based account system is as follows:

1. The user add a new employee.
2. Their password is hashed and stored in the database.
3. When the employee attempts to login, the hash of the password they entered is checked against the hash of their real password.
4. If the hashes match, the user is granted access. If not, the user is told they entered invalid login credentials.

4 Design Specification

4.1 Existing System

Before discussing the proposed model, we need to start from the previous research related to the same topic to prevent the SQLI, previously numerous encryption algorithm have been used to secure data, Using weak cryptographics it seems to be useless and cannot guarantee any level of security that means the risk of stealing data by decryption is very high.The existing systems are based on broken and insecure encryption mechanisms and have many vulnerabilities such as md5 and SHA family and RIPEMD, so that is why it is necessary to develop new approach to prevent SQLI.

4.2 Proposed System

The propsed system is based on bcrypt,as it is able to defend against various attacks such as:

– Brute-force attack.
–Rainbow table.
–Dictionary attack.

4.3 System overview

The general work of the system is as follows:

1 The user connects to the web application.
2 The second step is the authentication of the user.
3The user can add new items (department, designation, role, employee).
a The query will be processed to check if it is valid or not.
b If the query is valid the user can select the encryption type.
4The data will be encrypted then saved in the database.

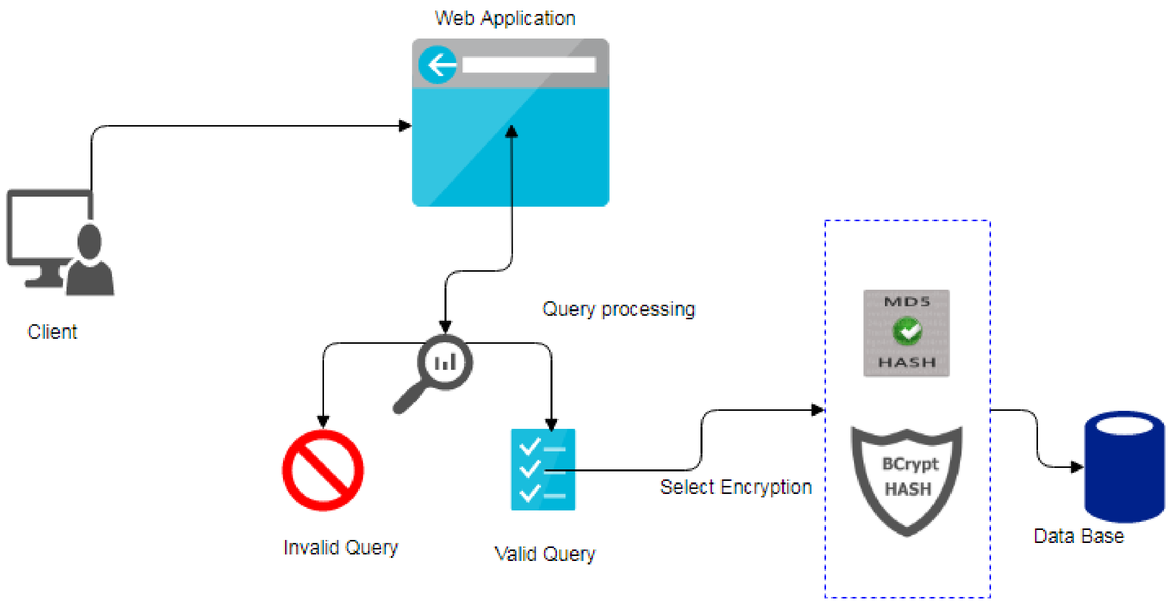
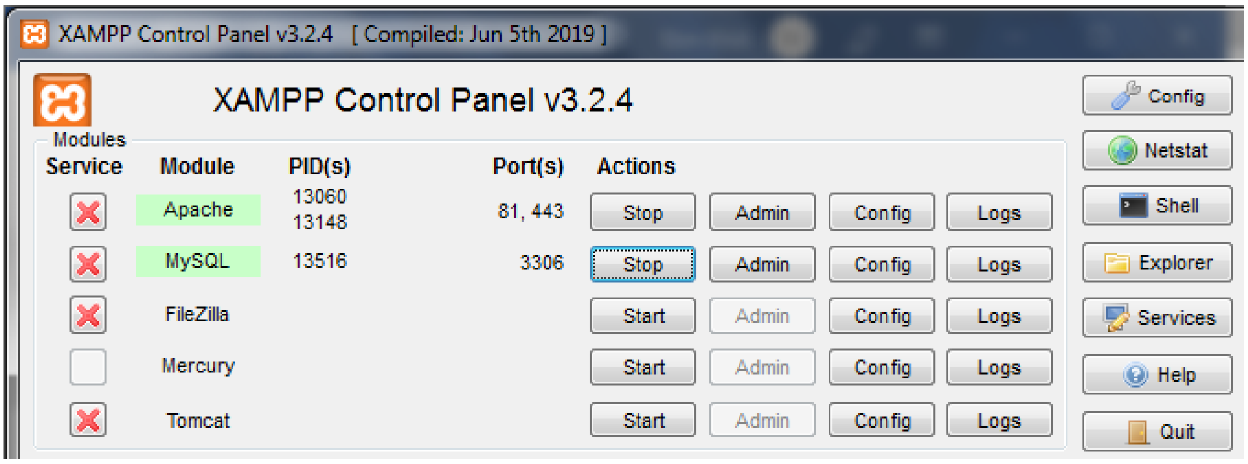


Figure 5: Architecture overview

5 Implementation

In order to run the web application a local web server is required and should be started and kept running, XAMPP is used as web server; it is an open-source cross-platform web server solution created by Apache friends[7].The advantage of using XAMPP is that allows the easy creation of web server for deployment and testing purposes. To run the web application and get the access to the database both MYSQL and Apache of XAMPP are required and need to be started otherwise the web application will not be connected to the database.


XAMPP Control Panel v3.2.4 [Compiled: Jun 5th 2019]

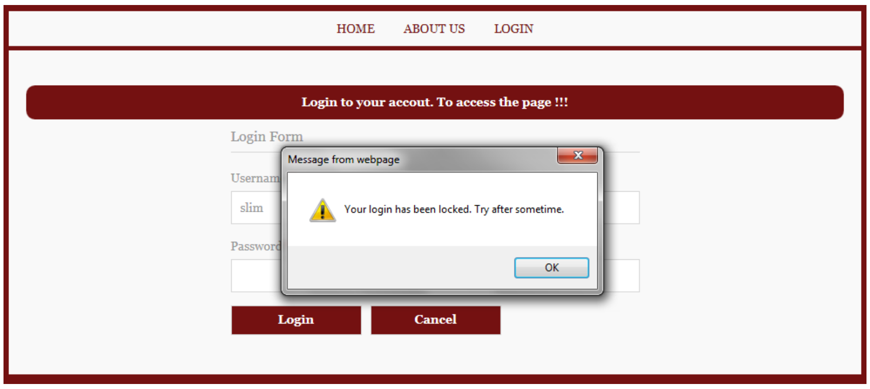
XAMPP Control Panel v3.2.4

Modules

Service	Module	PID(s)	Port(s)	Actions			
✗	Apache	13060 13148	81, 443	Stop	Admin	Config	Logs
✗	MySQL	13516	3306	Stop	Admin	Config	Logs
✗	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
✗	Tomcat			Start	Admin	Config	Logs

Figure 6: XAMP Control Panel

Once the web application is started the login page will appear. The user is required to enter a valid username and password.If the user enters wrong credentials error message will appear and if user 3 times successively enters a wrong username and password his account will be locked. Otherwise if the credentials are correctly entered so he will be redirect it to the login page.



Login to your aceout. To access the page !!!

Login Form

Usernam

slim

Password

Message from webpage

Your login has been locked. Try after sometime.

Figure 7: Login Page

The user will be able to see this page only after the authentication step with administrator role. The admin can perform different actions such as add/delete/edit or view the:

[7]https://en.wikipedia.org/wiki/XAMPP

- Employee
- Department
- Leave/Leave Type
- Salary
- Roster

There is also an option to allow the administrator to change the password, the normal user (employee) has just some restricted privileges such as to apply for leave, view payslip, view leave.

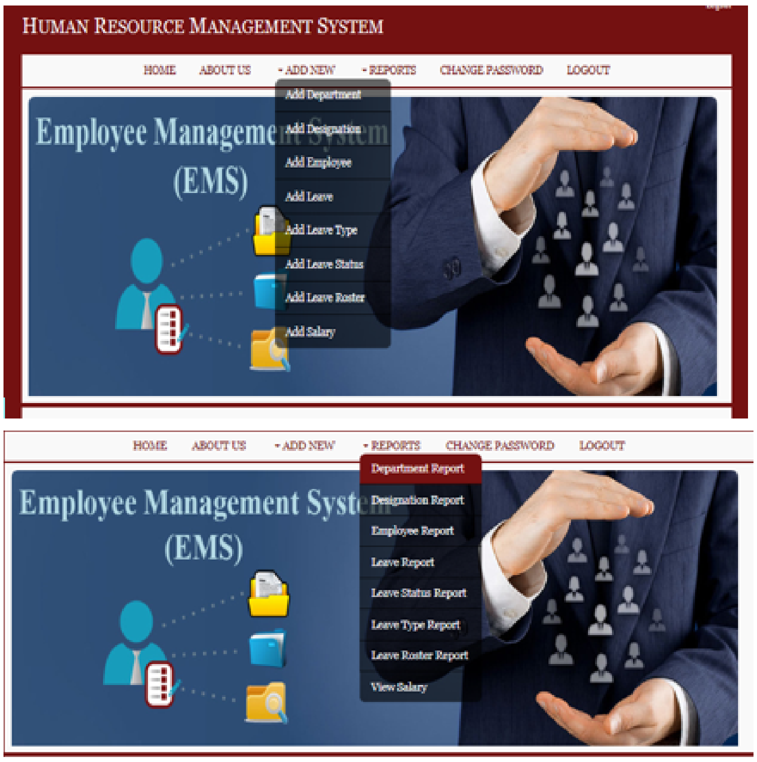
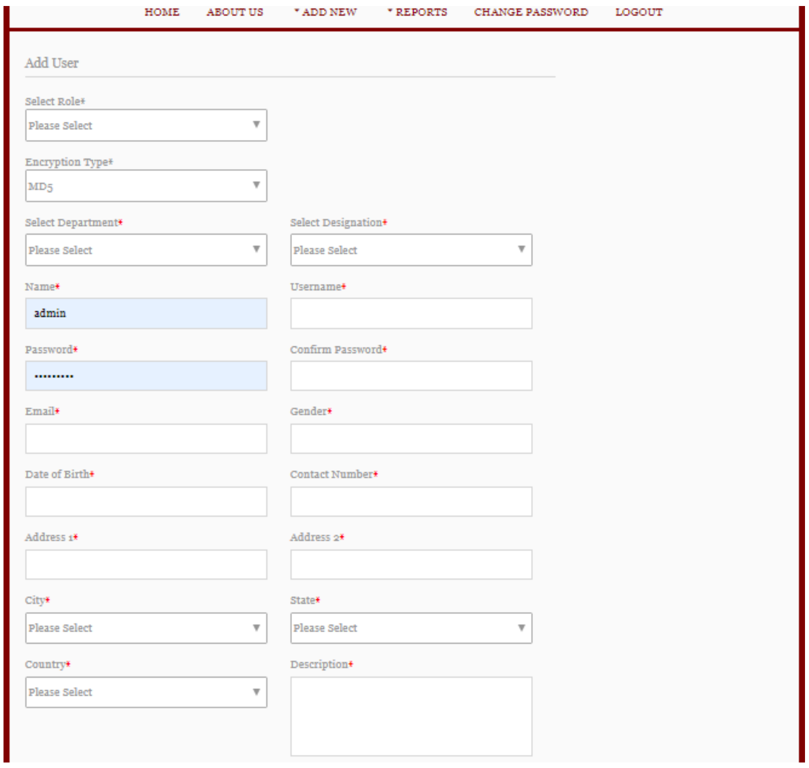


Figure 8: Main Page

The registration page allows admin to add a new employee it requires to entry of the details of the new employee (department, username, password, etc.) there are also features to select the type of encryption of the data in the database. This page uses the Whitebox technique.In order that the encryption happens successfully the whiteboxing technique is used to ensure the data to be encrypted does not contain any SQL injection, in other words the special characters which are mainly used to perform the SQL injection are not authorized. Once the data has entered by the user the encryption can be done successfully. As we mentioned earlier this feature is only for demonstration purposes to compare both encryption mechanisms.

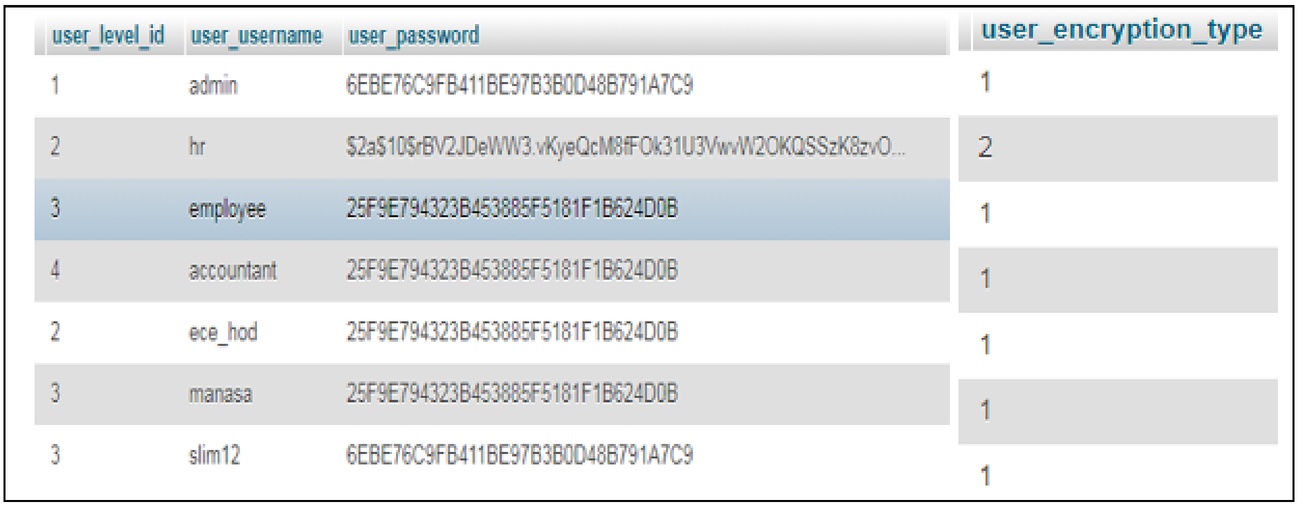
Two different users are added from the registration page. For the first the encryption used was md5 and for the second the encryption type used is bcrypt. The screenshot below illustrates the difference and shows the encryption in database, the encryption type 1 represents the md5 and the encryption type 2 represents bcrypt.



Add User

Select Role*: Please Select	
Encryption Type*: MD5	
Select Department*: Please Select	Select Designation*: Please Select
Name*: admin	Username*:
Password*: •••••••••	Confirm Password*:
Email*:	Gender*:
Date of Birth*:	Contact Number*:
Address 1*:	Address 2*:
City*: Please Select	State*: Please Select
Country*: Please Select	Description*:

Figure 9: Registration Page



user_level_id	user_username	user_password	user_encryption_type
1	admin	6EBE76C9FB411BE97B3B0D48B791A7C9	1
2	hr	$2a$10$rBV2JDeWW3.vKyeQcM8fFOk31U3VwvW2OKQSSzK8zvO...	2
3	employee	25F9E794323B453885F5181F1B624D0B	1
4	accountant	25F9E794323B453885F5181F1B624D0B	1
2	ece_hod	25F9E794323B453885F5181F1B624D0B	1
3	manasa	25F9E794323B453885F5181F1B624D0B	1
3	slim12	6EBE76C9FB411BE97B3B0D48B791A7C9	1

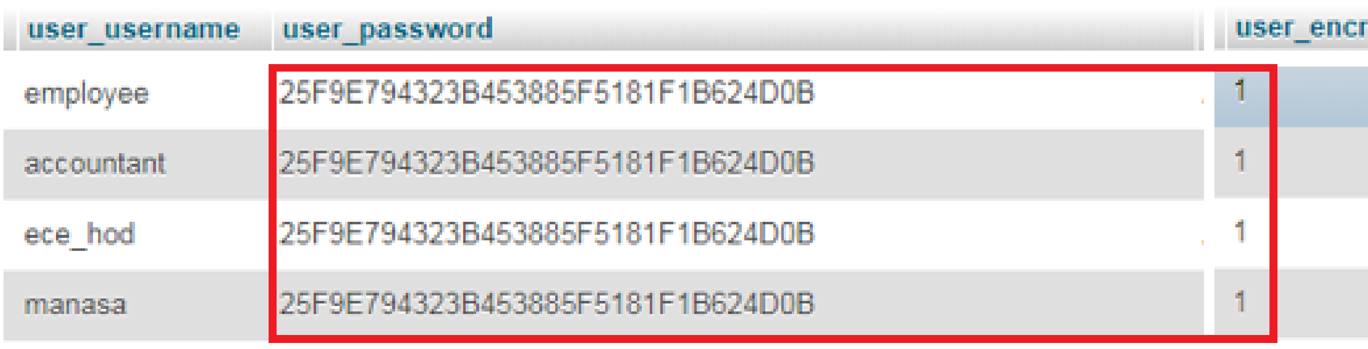
Figure 10: Database Encryption

6 Evaluation

6.1 Experiment 1

- Same password in MD5

A different user was added from the registration page and was assigned the same password so we can see there is same encryption for the different user as they have same password

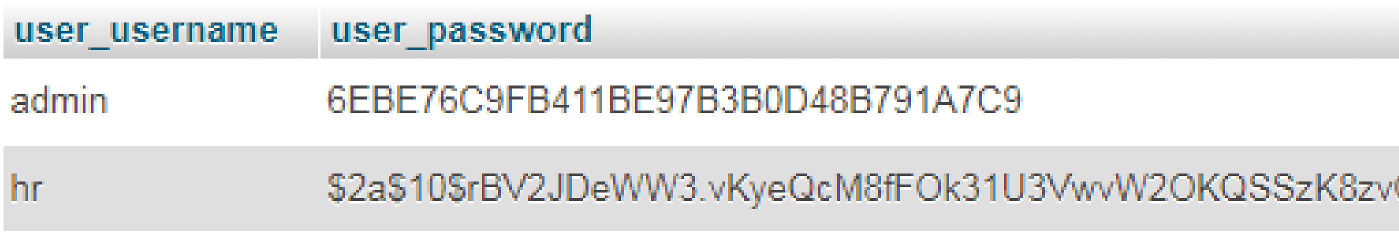


user_username	user_password	user_encr
employee	25F9E794323B453885F5181F1B624D0B	1
accountant	25F9E794323B453885F5181F1B624D0B	1
ece_hod	25F9E794323B453885F5181F1B624D0B	1
manasa	25F9E794323B453885F5181F1B624D0B	1

Figure 11: Same password in MD5

- Same password in Bcrypt

However this is not the case for bcrypt as every time a new hash is generated,it will never happen to have the same encryption even for the same password; moreover bcrypt is one-way encryption which means that even in the worst scenario if the attacker manages to get access to bcrypt encryption it will be useless and cannot be decrypted as in md5.



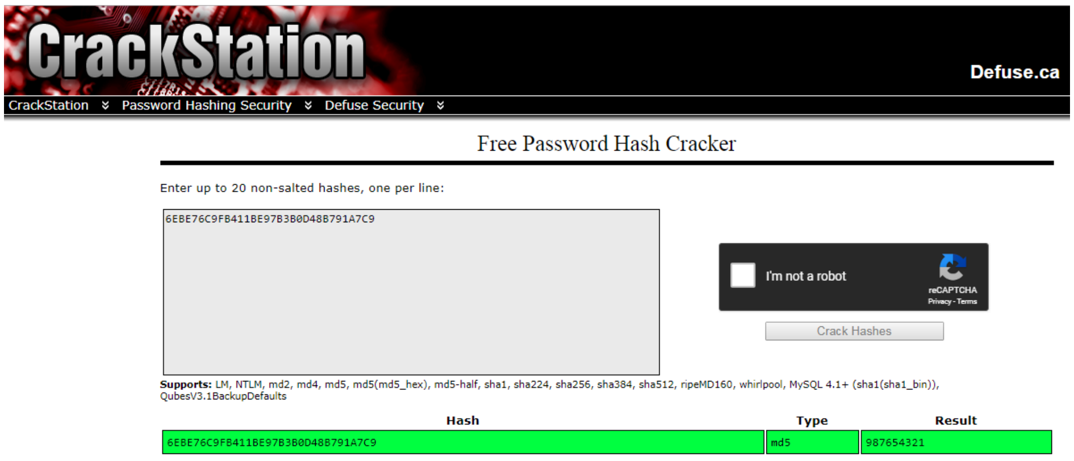
user_username	user_password
admin	6EBE76C9FB411BE97B3B0D48B791A7C9
hr	$2a$10$rBV2JDeWW3.vKyeQcM8fFOk31U3VwvW2OKQSSzK8zv

Figure 12: Same password in Bcrypt

6.2 Experiment 2

CrackStation[8] is the tool used in this experiment.It is designed by Defuse security; it is oriented to security researchers for demonstration; it is aimed to highlight the risk of storage of insecure passwords in web applications.

This experiment consists of trying to decrypt both passwords which are encrypted with different encryption mechanism



Free Password Hash Cracker

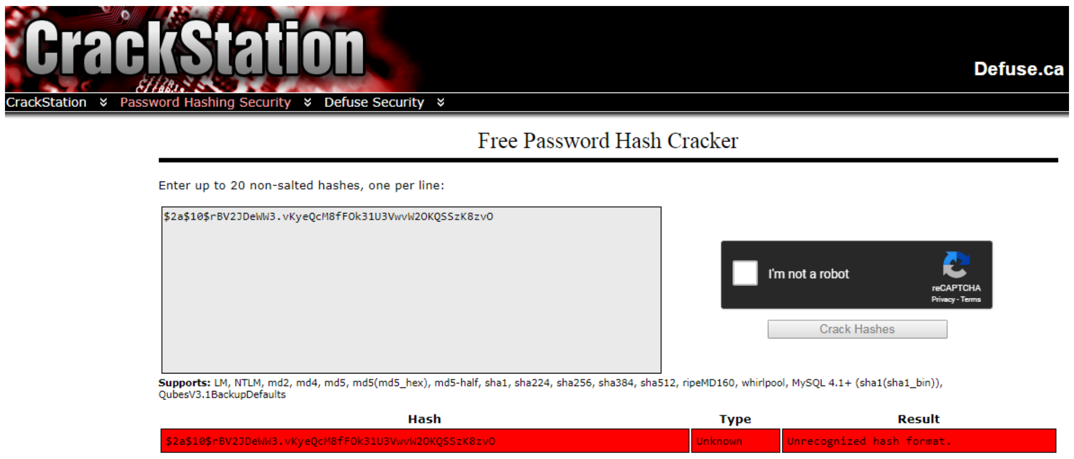
Enter up to 20 non-salted hashes, one per line:

```
6EBE76C9FB411BE97B3B0D48B791A7C9
```

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
6EBE76C9FB411BE97B3B0D48B791A7C9	md5	987654321

Figure 13: MD5 decryption



Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

```
$2a$10$rBV2JDeWW3.vKyeQcM8fFOk31U3VwvW2OKQSSzK8zvO
```

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
$2a$10$rBV2JDeWW3.vKyeQcM8fFOk31U3VwvW2OKQSSzK8zvO	Unknown	Unrecognized hash format.

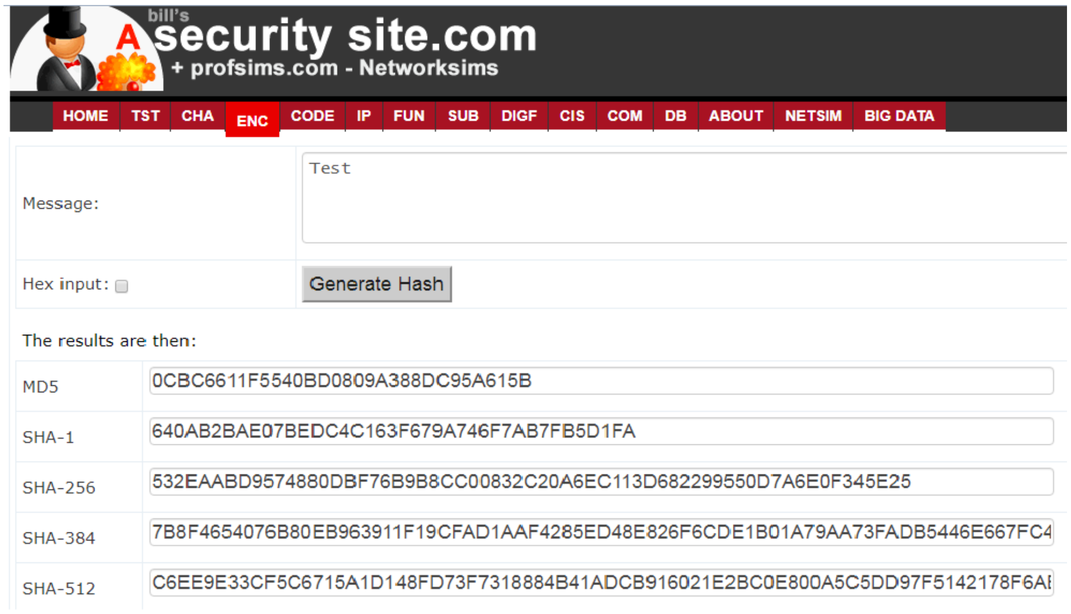
Figure 14: Bcrypt decryption

This experiment prove how weak is md5, while trying to decrypt the password encrypted with bcrypt was not successful

6.3 Experiment 3

MD5 and SHA-1,SHA-256,SHA-384 ANDSHA-512 produce a hash signature[9], but this can be attacked by rainbow tables.This algorithm always generates the same hash for the same message.

[8]https://crackstation.net/
[9]https://asecuritysite.com/encryption/bcrypt


bill's
A security site.com
+ profsims.com - Networksims
HOME TST CHA ENC CODE IP FUN SUB DIGF CIS COM DB ABOUT NETSIM BIG DATA
Generate Hash

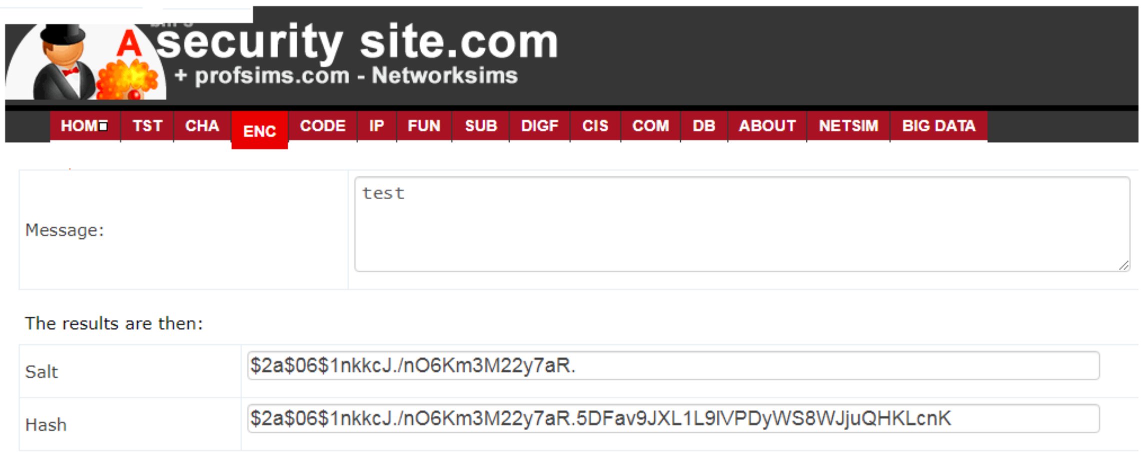
Message:	Test
Hex input: ☐	

The results are then:

MD5	0CBC6611F5540BD0809A388DC95A615B
SHA-1	640AB2BAE07BEDC4C163F679A746F7AB7FB5D1FA
SHA-256	532EAABD9574880DBF76B9B8CC00832C20A6EC113D682299550D7A6E0F345E25
SHA-384	7B8F4654076B80EB963911F19CFAD1AAF4285ED48E826F6CDE1B01A79AA73FADB5446E667FC4
SHA-512	C6EE9E33CF5C6715A1D148FD73F7318884B41ADCB916021E2BC0E800A5C5DD97F5142178F6A

Figure 15: Same Encryption Produced

Bcrypt is a more powerful and secure use hash generator for passwords and uses salt to create a non-recurrent hash.The experiment below shows how the same message has different hash; the hash will never be the same.

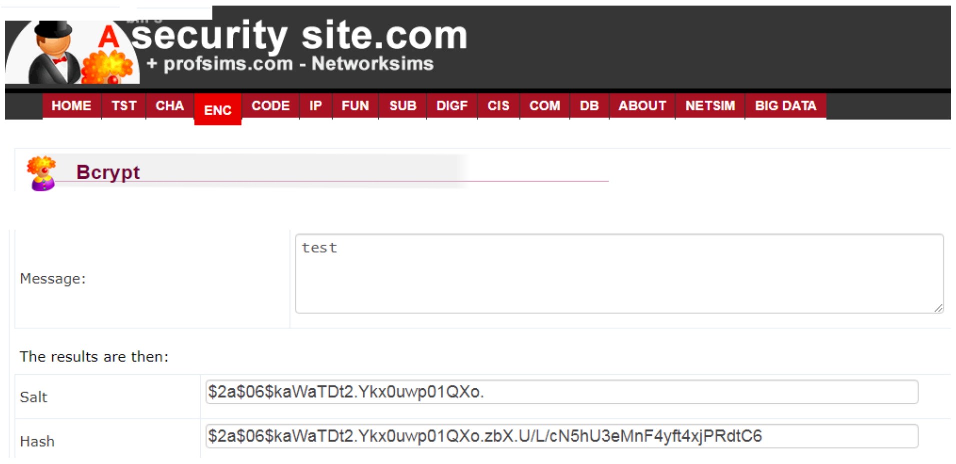

A security site.com
+ profsims.com - Networksims
HOME TST CHA ENC CODE IP FUN SUB DIGF CIS COM DB ABOUT NETSIM BIG DATA

Message:	test

The results are then:

Salt	$2a$06$1nkkcJ./nO6Km3M22y7aR.
Hash	$2a$06$1nkkcJ./nO6Km3M22y7aR.5DFav9JXL1L9lVPDyWS8WJjuQHKLcnK

Figure 16: Bcrypt Encryption Experiment 1


A security site.com
+ profsims.com - Networksims
HOME TST CHA ENC CODE IP FUN SUB DIGF CIS COM DB ABOUT NETSIM BIG DATA
Bcrypt

Message: test

The results are then:

Salt	$2a$06$kaWaTDt2.Ykx0uwp01QXo.
Hash	$2a$06$kaWaTDt2.Ykx0uwp01QXo.zbX.U/L/cN5hU3eMnF4yft4xjPRdtC6

Figure 17: Bcrypt Encryption Experiment 2

6.4 Discussion

The proposed web application aims to solve the SQLI attack based on the science of cryptography.The registration page for the new employee offer the user to choose the encryption type,in order to compare and contrast both output for the encryption and perform some test and use different tools in order to decrypt the encrypted password .

The outputs of the md5 encryption prove that the encrypted password is very weak and can be decrypted so easily by attackers. An interesting discovery of this work is that bcrypt is so complex algorithm; it takes an input,for example a password and after significant calculation generates the hashed password by using a one-way hashing function, which is irreversible, this process makes the storage of passwords.

We evaluated the performance of both encryption algorithms by analyzing the produced hash value of the password with md5 we found out that the hash can be decrypted in few a seconds while the bcrypt hash was not possible to decrypt.Currently it is appears to be the most secure encryption algorithm.

7 Conclusion and Future Work

In this study we have attempted to show the correlation between the web application security and the encryption algorithm.We engaged in simulating a web application with a database to demonstrate and show the impact of the encryption algorithm with the SQL injection. A traditional encryption algorithm was used in the application to compare it with the secure encryption algorithm. Hence, its understood the wrong and weak hashed it can be considered as useless and has same impact as saving the data as plaintext. The proposed hashing algorithm is the most secure one comparing to what is currently available,hence bcrypt could be a solution for the SQLI.

In future the concept of the application can be applied in social media websites as well as to web application where securing the sensitive data is very critical task such as online banking. We believe that bcrypt hashing is more secure and applicable and it could be a step towards improving the security of web application and mitigating the SQl injection

References

Dawson, Edward, G. P. M. A. (2006). *Collision Attacks on MD5 and SHA-1: Is this the 'Sword of Damocles' for Electronic Commerce?*, Proceedings of the AusCERT Asia Pacific Information Technology Security Conference.

Ertaul, L. and Kaur, M. A. (n.d.). Implementation and performance analysis of pbkdf 2 , bcrypt , scrypt algorithms varunkrg.

Johari, R. and Sharma, P. (2012). A survey on web application vulnerabilities (sqlia, xss) exploitation and security engine for sql injection, *2012 International Conference on Communication Systems and Network Technologies*, pp. 453–458.

John Black, Martin Cochran, T. H. (2016). *A Study of the MD5 Attacks: Insights and Improvements.*

K. Rajeswari, C. (2016). *SQL Injection Attack Prevention Using 448 Blowfish Encryption Standard*, SQL Injection Attack Prevention Using 448 Blowfish Encryption Standard.

Khili, S. (2019). *SQL Injection prevention technique using data hashing.*

Klima, V. (2005). *inding MD5 collisions A toy for a notebook*, ePrint Archive, Report.

Levent Ertaul, Manpreet Kaur, V. A. K. R. G. (n.d.). *Implementation and Performance Analysis of PBKDF2, Bcrypt, Scrypt Algorithms*, Int'l Conf. Wireless Networks.

Mehar Sood, S. S. (2017). *SQL injection prevention technique using Encryption*, International Journal of Advanced Computational Engineering and Networking, ISSN: 2320-2106.

Mishra, S. (2019). *SQL Injection Detection Using Machine Learning.*

Satoh., A. (2005). *Hardware Architecture and Cost Estimates for Breaking SHA-1s*, ISC, volume 3650 of Lecture Notes in Computer Science, pages 259273. Springer.

Savage, A. (n.d.). *MD5 and SHA-1 Collision Attacks: A Tutorial.*

Temeiza, Q., Temeiza, M. and Itmazi, J. (2017). A novel method for preventing sql injection using sha-1 algorithm and syntax-awareness, *2017 Joint International Conference on Information and Communication Technologies for Education and Training and International Conference on Computing in Arabic (ICCA-TICET)*, pp. 1–4.

Wang, X. and Yu, H. (2005). *How to Break MD5 and Other Hash Functions*, Advances in Cryptology - EUROCRYPT 2005, volume 3494 of Lecture Notes in Computer Science, pages 1935. Springer.

Wang Xiaoyun, Dengguo Feng, X. L. and Yu, H. (2004). *Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD. Cryptology*, ePrint Archive, Report.