

## 1. Introduction

In this chapter, we present and implement our proposed technique to preventing SQL injection attack. Then, we use a digital signature to support the authentication process. We also test the performance of our application and measure its efficiency in order to prevent SQL injection. The principal goal of this chapter is to show how to use a digital signature to build secure and strong framework against hackers' attacks specially 'Tautologies' theft.

## 2. Description of our proposed technique

Tautologies is a type of SQLI attack, injects SQL codes to the conditional query statement to be always evaluated true, This type of attack used to avoid authentication control and access to data by exploiting vulnerable input field which uses WHERE clause.

In our technique, we talk about a novel idea to prevent SQL injection attacks exactly tautologies attacks, which are based on cryptography systems. Our idea that lies on random digital signature that offers more security to protect web applications because, to break and avoid random algorithms requires a long time. The digital signature has several advantage as:

- ✓ Confidentiality: Users will be able to securely login to the web application using a digital signature.
- ✓ Digital signatures Security: The use of digital signatures reduces risks of data being intercepted, Read, destroyed, or altered while in transit.
- ✓ Imposter prevention: No one else can forge your digital signature or submit data falsely claiming it was signed by you. So, the hackers cannot trick or imitate your signature.

Our idea consists of four phases, which are selected algorithms, registration phase, login phase and key generation phase.

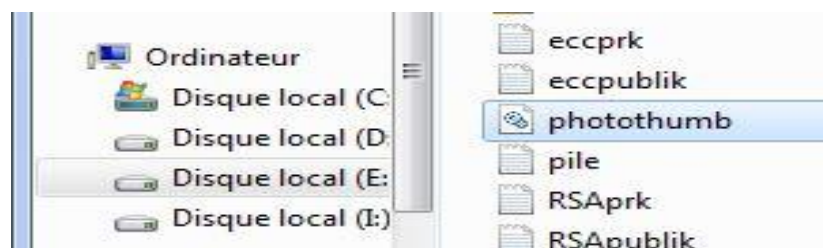
### 2.1. Selected algorithms

The choice of cryptography algorithms is a successful step to ensure perfect work in our strategy. There are other cryptography algorithms that we can use in our strategy, but we think it is less secure and performant to build a framework against a SQL injection. Hence the reason that makes us rely on RSA and ECC as cryptography algorithms in our idea is explained in points below:

- a. The advantage of public-key cryptography is increased security and convenience: private keys never need to be transmitted or show to anyone. In a traditional cryptography, by contrast, the secret keys must be transmitted (either manually or through a communication channel) since the same key is used for signing and verification. A serious concern is that there may be a chance that an enemy can discover the secret key during transmission. In addition, another major advantage of public-key systems is that they can provide digital signatures that cannot be repudiated.[36]
- b. ECC is the best algorithm of asymmetric cryptography system. The ECC is an emerging alternative for traditional Public-Key Cryptosystem like RSA, DSA, ECC that provides the highest strength-per- bit of any cryptosystem known today with smaller key sizes, resulting in faster computations, lower power consumption and memory. The ECC concept used with DH to solve the problem of key Exchange is the algorithm called ECDHA. ECC is also used for the digital signature and called ECDSA algorithm. [30]
- c. The generation key in ECC cryptosystem is faster than the other cryptosystems and this is due to the small key length; on the other hand, the signature generation and signature verification in RSA are faster than the other cryptosystems but the key length of RSA is very large. [31].

## 2.2. Key generation

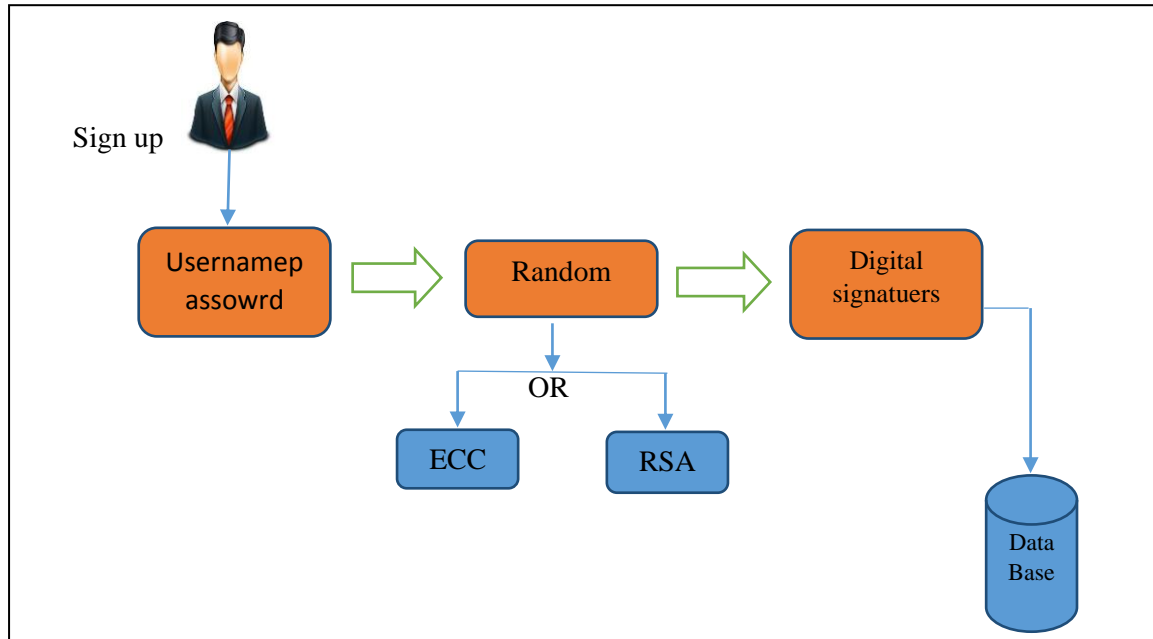
In the **Fig.4.1**, we showed the place of keys and explained the main reason behind putting the key in (E:) instead of in the database: we think it is very important to safeguard keys, because a successful key protection is considered an initial step to ensure perfect and secure work to cryptography algorithm. So, if hackers bypass the authentication process and steals the data, they cannot read information because they do not have the correct key. But, if hackers get the correct key, they can read and modify data. It is possible to store the key in a database, but in my opinion, it is not very secure.



**Fig.4.1.** RSA and ECC keys place.

### 2.3. Registration phase

In the first phase of the model, a new user needs to register by providing a unique combination of a username and a password through a web portal user interface. Upon submitting the user's credentials, data are sent to the server. In our application, when the user submits username and password in first time, the server executes the random algorithm.



**Fig.4.2.** Registration phase.

The random algorithm is relies on the randomization signature and is used to sign and convert the input into a ciphertext and either use RSA or ECC to sign the input of the user. In the figure below, we show the role of our strategy that relies on ~~in~~ randomization signing in the registration phase:

passwordc	dc
LtLlSKRpNT3oCpQ34tbRB%2FxACz%2FDLRQkdMmpoelhjoEhp...	2
MC0CFQDP4TtcV9Zcp%2BVX7BW1em1c2%2BfK0QIUb1HgQIOyMw...	1
MC0CFCqTw4uE10%2FQZ%2BR8YrgNjUgBg4WZAhUAouk46fYe4...	1
MCwCFFDrCYbFBXFULNezxZJU3MF9u%2BMTAhQNbhhTFEWHOIng...	1
MCwCFFkQIVtCJFYsJlVSIxIq7WkxP7AhRXIQ0zY0emvSu4wO...	1
MC0CFQCcmdDK46wW2vaxWTHY2TGCav3wpFAIUdrd%2BKSXw57EB...	1
BYA%2Fgb9BAAK%2BwH%2F%2FP4FA%2FkEafF%2F%2F39AP8H...	2
MC0CFEsledLXGhCBY1iXu1ovZ0AuFdULAhUAiXX1YnAvCbhW4i...	1
MCwCFHd7%2Bqzfz1eaJNw%2FAsiVHC2IG26kAhQPTXO9BbzxmC...	1
elbd7gmf51z7HBZ2p%2BNv3c%2BfBLmgNQifESWPrIMbGLTgV4...	2

**Fig.4.3.** The user's database.

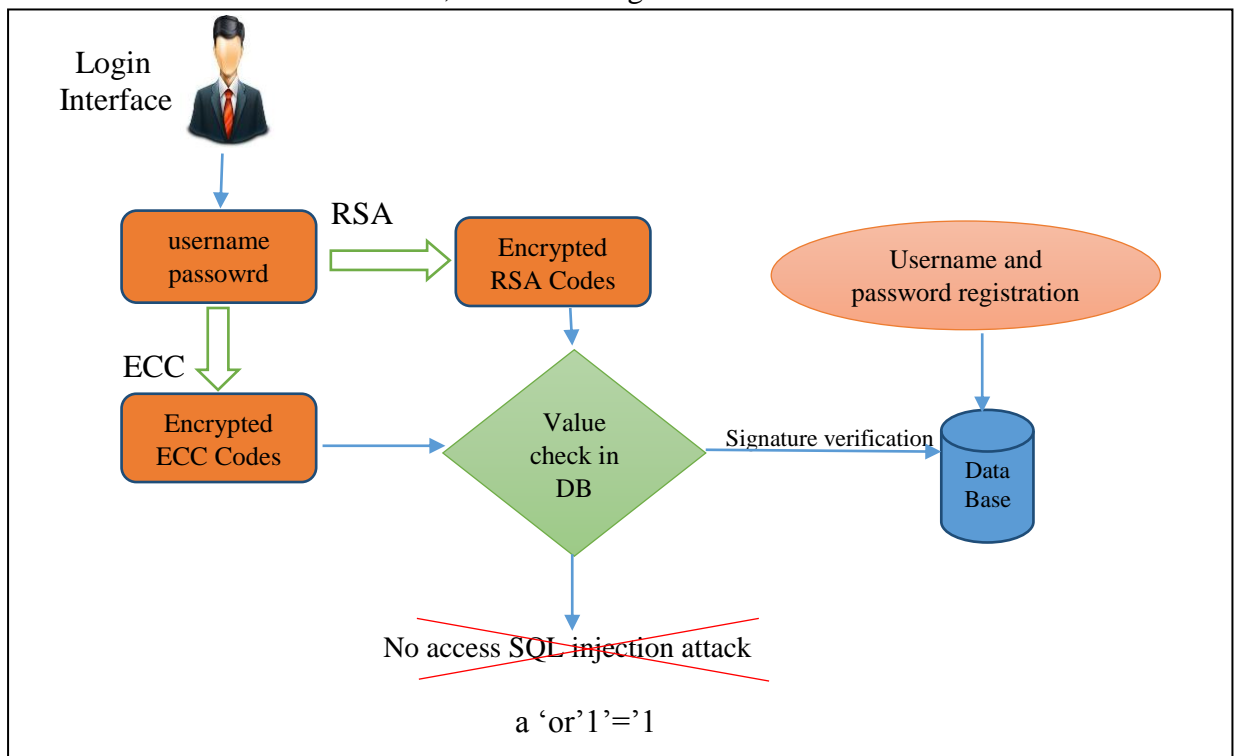
The “**idc**” is the type of algorithm which is used to sign the input of the user, collecting the username and password in the same column “**password**”, to avoid more Memory usage and less time to signature and signature verification.

As we have shown in figure 4.3, all user’s information in the table is not in natural form. We think it is very secure compared with another technique that saves the information in actual form. Hence, in our application we rely on encrypted data, till the hackers who can avoid the security levels cannot read or modify the information.

## 2.4. Login phase

An important security mechanism needed by most web applications is secure login where the user proves his identity. Traditionally, most applications have implemented login with a static user-ID and password. Digital signature offers more secure login which achieves true two-factor authentication. In the steps below, we explain how the login process occurs:

- ✓ User enters username and password.
- ✓ The server side verifies the signature (Password and username) by using the ECC. If (verification) returns true, this means login is successful.
- ✓ The signature is verified by using the RSA. If (verification) returns true, this means login is successful. If both RSA and ECC verify that the signature does not exist, and the verification returns false, this means login failed.



**Fig.4.4.** Login phase.

### **3. Implementation**

#### **3.1.The developement environnement**

##### **3.1.1.NetBeans IDE**

NetBeans is an open-source integrated development environment (IDE) developed with Java, PHP, C++, and other programming languages. NetBeans is also referred to as a platform of modular components used for developing Java desktop applications. [34]

##### **3.1.2. Servlet**

A servlet is a Java programming language class that is used to extend the capabilities of servers that host applications accessed by means of a request-response programming model. Although servlets can respond to any type of request, they are commonly used to extend the applications hosted by web servers. For such applications, Java Servlet technology defines HTTP-specific servlet classes. The `javax.servlet` and `javax.servlet.http` packages provide interfaces and classes for writing servlets. All servlets must implement the `Servlet` interface, which defines life-cycle methods. When implementing a generic service, you can use or extend the `GenericServlet` class provided with the Java Servlet API. The `HttpServlet` class provides methods, such as `doGet` and `doPost`, for handling HTTP-specific services. [35]

##### **3.1.3. JSP definition**

What exactly is JSP? Let us consider the answer to that question from two perspectives: that of an HTML designer and that of a Java programmer. If you are an HTML designer, you can look at JSP as an extension of HTML that gives you the ability to seamlessly embed snippets of Java code within your HTML pages. These bits of Java code generate dynamic content, which is embedded within the other HTML/XML content. Even better, JSP provides the means by which programmers can create new HTML/XML tags and JavaBeans components that offer new features for HTML designers without requiring you to learn how to program [32].

### **3.1.4. PHPmyadmin**

PHPMyAdmin is an open source free software, designed to deal with the administration and management of MySQL databases through a graphic user interface. Written in PHP, PHPMyAdmin has become one of the most popular web-based MySQL management tools. PHPMyAdmin comes with a detailed documentation and is being supported by a large multi-language community. PHPMyAdmin's ever growing list of features supports all commonly used operations such as browsing, dropping, creating, altering MySQL databases, tables, fields and indexes. In addition, PHPMyAdmin enables you to manage MySQL users and user privileges [33].

## **3.2. The cryptographic library in JAVA**

In our application, we used FlexiProvider to implement the different public-key cryptosystems schemes like NTRU, ECC, RSA..., We chose FlexiProvider because it offers the supplies fast and secure implementations of cryptographic algorithms which are easy to use even for developers who are not well-footed in the field of cryptography.

### **3.2.1.FlexiProvider [11]**

The FlexiProvider is a powerful toolkit for the Java Cryptography Architecture (JCA/JCE). It provides cryptographic modules that can be plugged into every application that is built on top of the JCA. The FlexiProvider has been developed by the Theoretical Computer Science Research Group of Prof. Dr. Johannes Buchmann at the Department of Computer Science at technische Universität Darmstadt, Germany. The FlexiProvider contains two sub libraries which I use in our application: CoreProvider, ECProvider...

#### **3.2.1.1. CoreProvider**

The CoreProvider contains well-known public key algorithms such as the RSA cipher and signature in different flavors according to PKCS #1, a multitude of symmetric block ciphers, most prominent among them the AES cipher Rijndael and 3-DES, password-based encryption (PBE) according to PKCS #5, hash functions such as MD5, SHA1, and RIPEMD, MACs such as CBC-MAC, CMAC, HMAC, plus its own pseudo-random number generator (BBS).

### 3.2.1.2. *ECProvider*

The ECProvider is a provider for cryptographic algorithms which are based on elliptic curves. The provider includes the digital signature schemes ECDSA and ECNR, the Elliptic Curve Diffie-Hellman (ECDH) key agreement scheme, and the Elliptic Curve Integrated Encryption Scheme (ECIES).

### 3.2.2.mysql-connector-java-5.1.37-bin

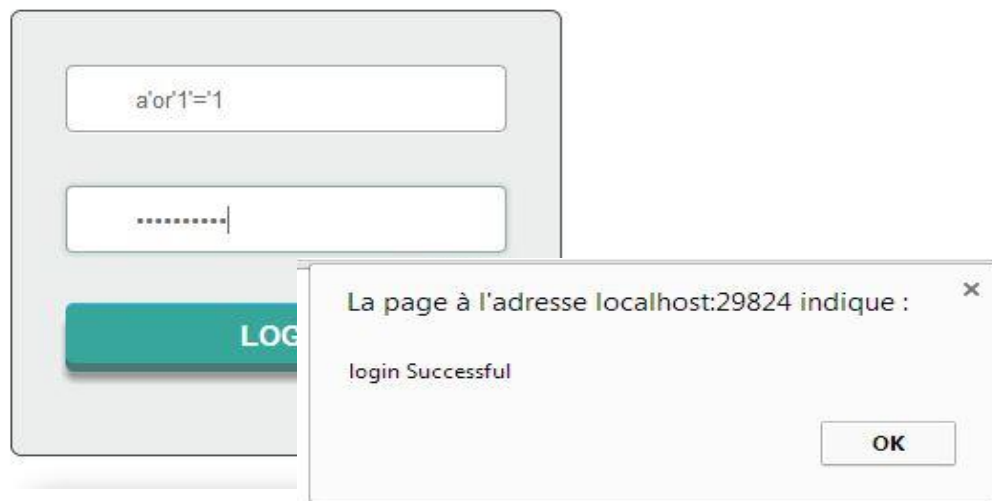
The mysql-connector-java-5.1.37-bin library that offers a way in order to connecting the database with NetBeans IDE.

### 3.2.3.Bouncycastle provider

It offers all cryptography algorithm (key generation, digital signature and verification, encryption and decryption)

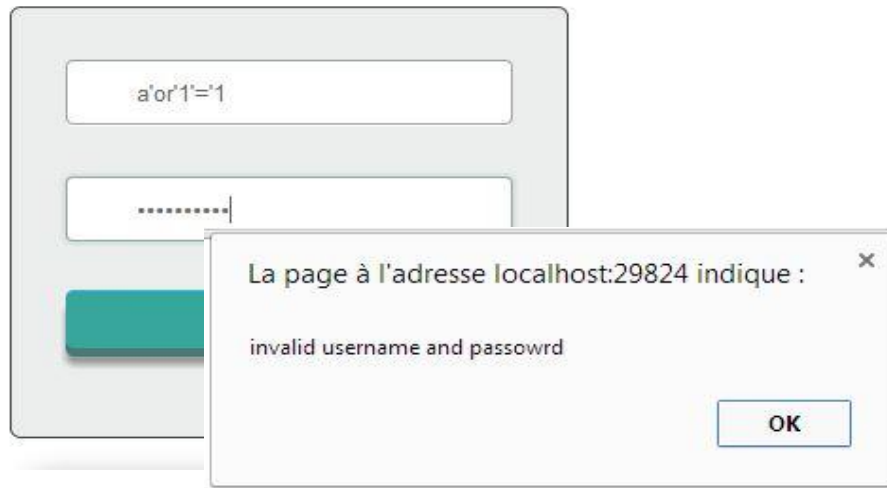
## 3.3.Experimental results

In our application, we try to build an example to apply simulation of SQL injection attacks, and test our application before and after applying our strategy which is based on a cryptography system, as shown in the figures below:



**Fig.4.5.** Before applying our strategy.

Before applying the encryption algorithm the attack using tautology technique which was discussed in chapter1, was successful. An illegal access was allowed. A sample screenshot shows the effect in [Fig.4.5]. After applying the encryption algorithm, it was found that the previously successful attack using tautology became unsuccessful. A sample screenshot to show this effect is in [Fig.4.6.].



**Fig.4.6.** after applying our strategy.

So, we think that the performance comparison of cipher text over normal text shows that, cipher text is very difficult and time consuming to crack. Because the time to crack a cipher text generated at random takes a long time. Hence, our application contains high levels of security to reduce the problem of a SQL injection.

SQL injections will be prevented because only the encrypted values are checked with the database and not the actual input.

#### 4. Performance Evaluation

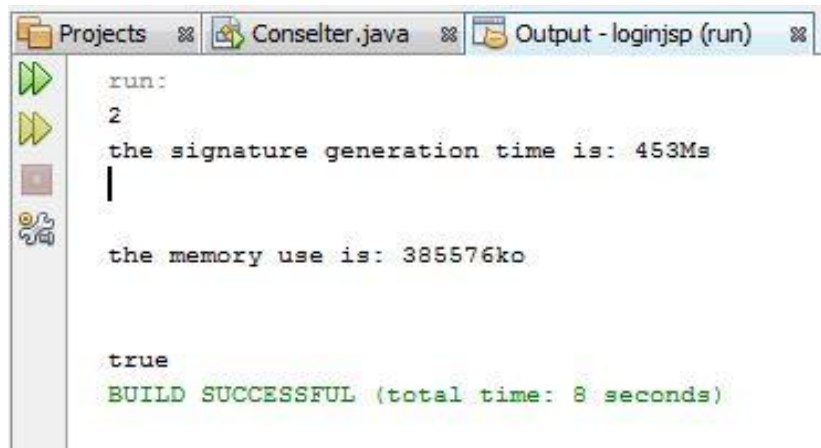
The performance of any system needs to be evaluated on certain criteria. These criteria then, decide of the basis of performance of any system. Such parameters are known as performance metrics. The 3 types of performance metrics are used to evaluate the performance of encryption and decryption algorithm. In addition, digital signature and signature verification in this part of chapter, are described below: [28]

- ✓ **Key length:** RSA (1024), ECC (106)
- ✓ **The word that use to evaluation:** marwan

##### 4.1. Digital signature time

Digital signature time is yet another important issue because it is basically used to calculate the throughput of a digital signature scheme as well as it indicates its speed. The digital signature time can be defined as the time that an cryptography algorithm takes to produce a digital generation from a plaintext.

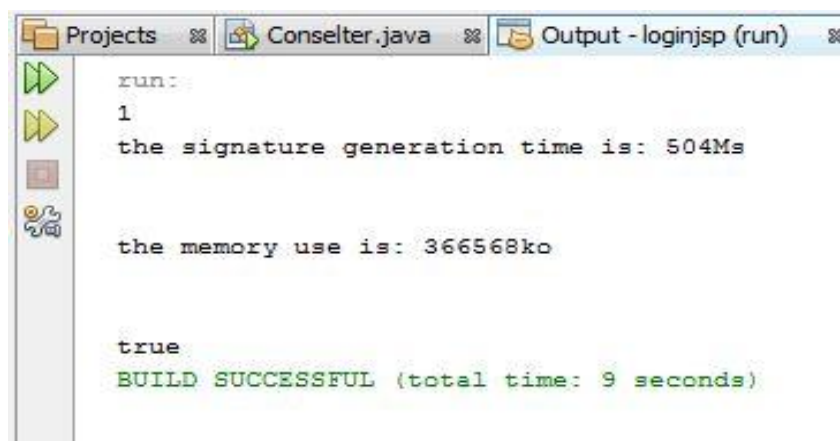




```
run:
2
the signature generation time is: 453Ms
|

the memory use is: 385576ko

true
BUILD SUCCESSFUL (total time: 8 seconds)
```

**Fig.4.7.**RSA Digital signature time.

```
run:
1
the signature generation time is: 504Ms

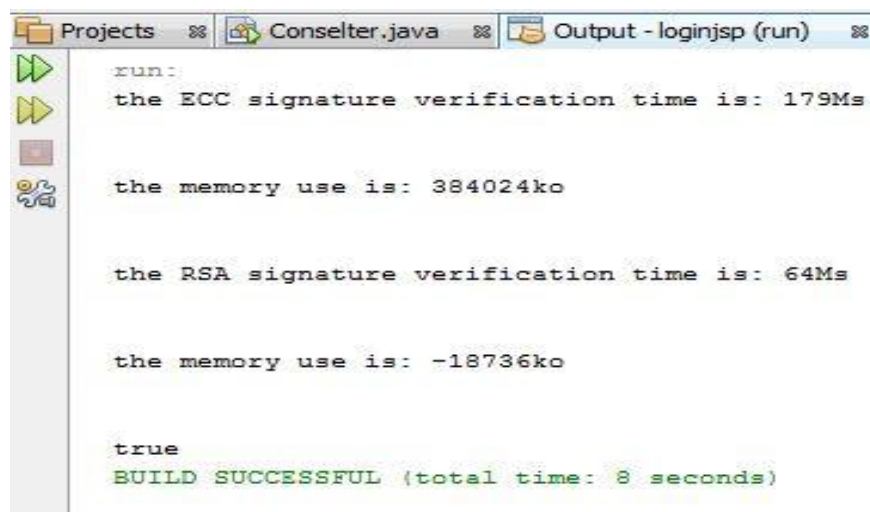
the memory use is: 366568ko

true
BUILD SUCCESSFUL (total time: 9 seconds)
```

**Fig.4.8.**ECC Digital signature time.

## 4.2. Digital signature verification time

The digital verification time is the inverses of digital signature time that can be defined as the time that a cryptography algorithm takes to verify the signature.



```
run:
the ECC signature verification time is: 179Ms

the memory use is: 384024ko

the RSA signature verification time is: 64Ms

the memory use is: -18736ko

true
BUILD SUCCESSFUL (total time: 8 seconds)
```

**Fig.4.9.**Digital signature verification time.

### 4.3. Memory usage

This is the amount of memory consumed the digital generation and verification signature process.

		ECC	RSA
Digital signature time		504Ms	453Ms
Signature verification time		179Ms	64Ms
Memory usage	Signature	366568 Ko	385576 Ko
	Verification	384024 Ko	18736 Ko

**Table.4.1.**performance evaluation.

In addition to **fig 4.7** and **fig 4.8,fig 4.9**, in the **table 4.1**clearly show the time and memory usage that consumed during authentication and registration phase.

## 5. Discussion

The results obtained from Fig.4.1 clearly show that the generation key of ECC and RSA in (E:) we realised is more secure when not stored in the database. This is mainly because there is a kinds of a SQL injection attack (Logically incorrect query), which we think represents a threat to the database. For example, if hackers get a message that came from the database including useful information, this may help attackers to steal a keys or other sensitive information.

In our project, we did not make a comparison between the algorithms, but we studied them. We found good cryptographic algorithms like ECC and RSA that we used in our application. So, from Fig.4.7 and Fig.4.8, Fig.4.9, we proved the reason behind choosing the ECC and RSA. The generation key in ECC cryptosystem is faster than the other cryptosystems and this is due to the small length of the key. On the other hand, the signature generation and signature verification in RSA is faster than the other cryptosystems but the key length of RSA is very large. It is very important to select a good algorithm (less time to signature generation and signature verification, less memory usage)on which we can rely to build a strong and secure strategy to prevent a SQL injection.

The previous results in Fig.4.5 and Fig.4.6, show the efficient role of our strategy after applying them to protect a web application against SQLI. In addition, “**a’or’1’=’1**” request became unsuccessful. Hence, the hacker cannot bypass the authentication process.

Finally, our idea was based on a random algorithm in the registration phase that given other fact of security level. We chose algorithm of cryptosystem that has good characteristics and that offers high levels of security to protect web application and prevent SQLI.

## **6. Conclusion**

In this chapter, we used some algorithms of Asymmetric cryptosystem in our application to try to prevent a SQL injection. We used RSA and ECC algorithms in order to build a strong strategy to protect web applications against SQLI. Also, we relied on digital signature to sport user authentication, as well as to apply a RANDOM algorithm in the registration phase that gives and offers more security and efficiency to our application, Finally, we discussed the obtained results.