IMPLEMENTATION OF DIGITAL SIGNATURE IN WEB TO IMPROVE THE QUALITY OF MANAGEMENT INFORMATION SYSTEMS IN EDUCATION SECTOR

Arbnor Halili
Blerim Rexha
Albert Aliu
Department of Business Informatics', Globus University College, Prishtina, Kosovo

Abstract

Information technology is becoming the world's most influential factors in everyday life. It has even large economic impact in family and state budget. Web software systems, due to their friendly network configuration and hardware independence properties ate gaining more and more user acceptance. But, in web systems the authenticity of transmitted and stored data is still an open challenge. This paper aims to give the answer to these questions using digital certificates, as defined by ISO X.509 standard, for encrypting communication channel and digitally signing student data for archiving.
The implementation of digital signatures on the web has special challenges. Digital signature uses digital certificate of the current user, which certificate is stored in user's local store or file and it has restricted access. This paper presents an implementation web digital signature in two different platforms: (i) using Microsoft browser, and (ii) non-Microsoft browser and comparing their properties, functionality and efficiency. This paper, furthermore, describes in detail the implementation of the digital signature on the web starting from the definition of the problem up to the technical implementation, including as a case study the Student Management System installed at the Faculty of Electrical and Computer Engineering, University of Pristina

Keywords: Digital signature, information technology, management information systems, ActiveX, Applet

Introduction

The development of technology in general has increased the demand of individuals and institutions for various information services. This trend has affected the South East Europe countries in general and Kosovo in particular. The same goes for digital signatures. Compared to the world countries this trend is slightly delayed for this region. This is based on very concrete cases of implementation of digital signing and drafting of appropriate legislation that will precede technological development. As one of the cases to prove this it suffices to note that the former U.S. President Clinton signed in 2000 for the first time a bill in electronic form [2]. In addition to the U.S. in the period up to 2002 there were many other countries that have worked in drafting the legislation regulating digital signature [8], [9].

However, the academic institutions in Kosovo have expressed the need for the use of new technologies as a facilitator element in learning process and progress of activities. Therefore, different systems for managing the progress of student data in academic institutions or the so-called management systems for university were developed. In line with the new trends, in 2006 with an initiative of computer engineering department of Faculty of

Electrical and Computer Engineering, University of Pristina, a System of University Management as a win application was developed and deployed [3].

It was an innovation for the period of time when it was implemented, but the main problem was encountered in later stages when it was tried to transform it in a web platform even though referring to the [19] users are as twice slower when using web applications against desktop or win applications but the world trend every day is going toward this type of technology. The main issue raised with the generation of digital signature in web and the compatibility of browsers in terms of the usage of digital signature to sign the student grades, which are also the focus and the main problem of this paper. Furthermore, for each message m (seen only as ranges of bits) signatory may apply a specific algorithm using its private key sk; consequently, this results in a signature that can be verified by anyone who has the corresponding public key pk using the specific algorithm for inspection outcome [1].

Digital signature of student grades in the system is necessary because the digital signature schemes contain the property of non-repudation, thus signature S cannot be denied the signing which was made long time ago by himself (to a third party) [20].

The researchers in this paper have used different tools and operating systems in order to find out the solution of implementation of digital signatures in different browsers and platforms.

Definition of the problem

In order to define a solution regarding a particular problem, it needs to base on fundamental issues on which the problem arises.

The key issues in this problem are:

- Current systems for implementation of digital signature,
- Analysis and study of the current system and its implementation in the Faculty of Electrical Engineering and Computer
- Analysis and definition of opportunities for the realization of the digital signature on the web
- Analysis and determination of other opportunities for improving the quality of technical solution and grading signing
- Testing and adjusting the system to the new changes**.**

There are various methods for implementation which are used to achieve the expected results or project work. Selected methods are oriented in respect of the rules in developing secure applications as well as de facto standards for ensuring the security condition. Methods are based on several pillars which are presented below:

- The proper use of cryptography
- Simplification of algorithms application
- Tracking algorithms schemes
- Compatibility with browsers
- Distribution Strategy

The proper use of cryptography is intended to be used in realizing the advantages offered and used in different systems, advantages which are expressed in form of various algorithms or schemes. As a part of solving the problem the researchers used the classes offered by. NET Framework and more specifically from the System.Security.Cryptography namespace [10].

To simplify the application and usage of such algorithms they should be better defined through various documents. If a certain programming language provides possibilities for certain desired properties, then there is no need to select a different way, which in the best case would complicate very little the entire implementation.

Tracking algorithms schemes constitutes a link between rational uses of cryptography algorithms. Linking in a simple way during the implementation makes an algorithm more reliable and more easily manageable. The more complex is the scheme, the more difficult it becomes the compilation of arguments on the security of the scheme [7].

Compatibility with browsers also defined as the application request is a way to access the finalized product. To have a module which is suitable for different browsers, the choice lies between the main module implemented as ActiveX and the module implemented as Java Applet. Java Applets (except browsers) are also independent of the hardware and the card manufacturer [4]. On the other hand, it is noteworthy that the ActiveX is an OLE object created by Microsoft that provides support for IUnknown interfaces [11]. In addition, the distribution strategy has to do with the installation of the module in the devices of the users. As a matter of fact, this strategy was developed in such a way that the distribution was easier through automatic installation or through links to be downloaded.

The issues of client-server architecture:

The main issue starts exactly at the basic element of software systems on the web, especially in their architecture. It is preferable that the web software systems to be developed in client – server architecture where the client and the server usually are executed on different machines connected by computer network [6], especially in the three tier architecture since it has the possibility of expanding and scaling, thus promoting various updates from time to time in this architecture. Systems developed in this architecture are typically thin client systems. In this case all calculations and business logic on the server becomes and remains the only part of the presentation to the client.

On the other side, digital certificate is the main element of digital signature in web through which the digital signature is done. It is known that the certificate contains the elements needed to create a valid digital signature, which is the case of X509 certificates [17]. Digital certificates are located in the warehouse or store of the client, therefore, in order to achieve unsurpassed signature the main requirement is to have access to the client store. This gives us the first problem, how to get access to local warehouse client certificates?

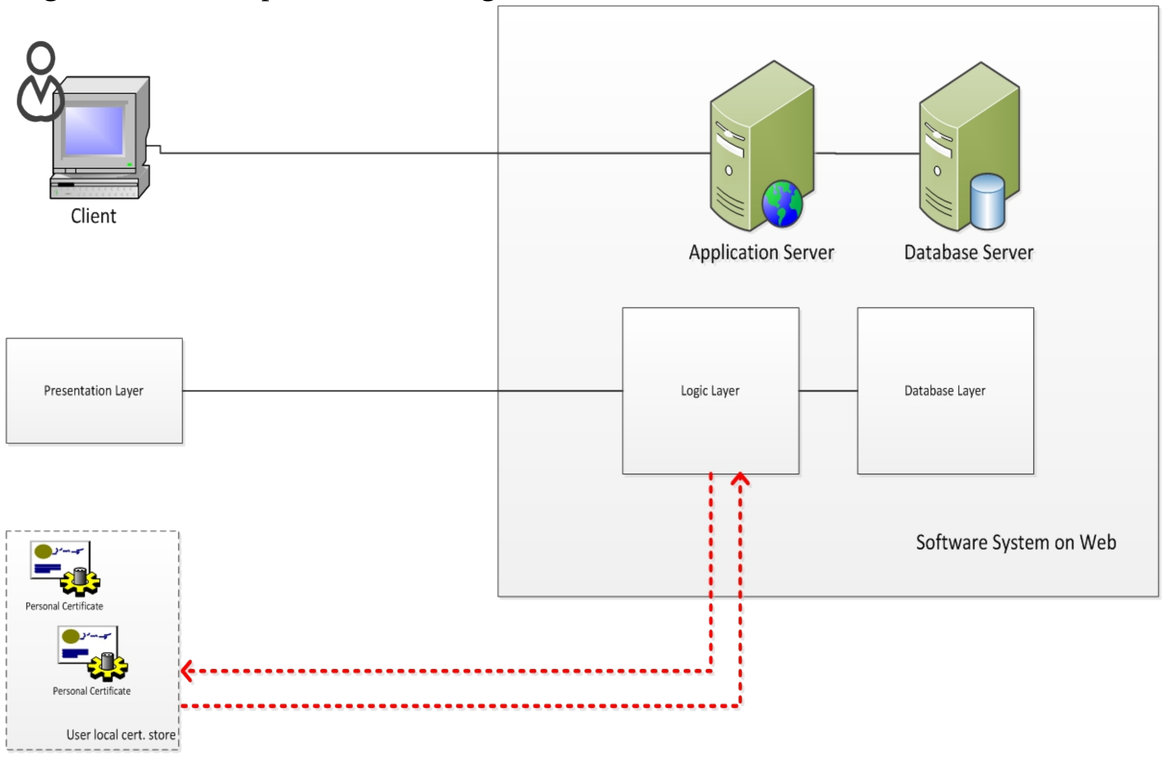

Figure 1. The communication infrastructure for access to warehouse certificates

From Figure 1 it can be seen that the direct communication from the logic layer to local storage is impossible. It should be noted that client to server communication is done via the HTTP protocol. This protocol works in logic of request and response (request - response). Given that the digital certificate is inevitable there should be a solution to achieve access to digital certificates. To achieve the solution the researchers needs to find a way that is part of the code implemented on the client side.

The next problem that meets the general problem during the implementation of digital signature on the web is the problem of compatibility. This is a problem that often occurs in software systems in general and in particular in those in web. Therefore a question arises, on how we realize a system which does not depend on the platform, which in this case the system does not depend on web browser either?

The compatibility in dualistic terms, being independent from the platform of operating system and independent of the browser in which is running poses a significant danger element in software systems on the web. Various technologies exist that claim to be independent from the platform and also from the browser, nonetheless is hardly acceptable and difficult to be proved. There are technologies that run on certain systems or certain browsers (such as Flash) but this number is limited and eliminates compatibility of any sort of absolute form.

Implementation of the solution through the ActiveX control:

ActiveX controls is an "OLE Object ", more specifically, a “Component Object Model (COM) object." In other words, it can be said that is a COM object that supports the IUnknown interface and also self - registration [15].

It was used Visual Studio 2010 IDE to build the ActiveX control for digital signature. ActiveX controls consist of two interfaces. In one of the interfaces are defined functions which are owned by ActiveX controls. This is similar to the header files of C ++ programming language in which initially the used functions which are part of the program are declared, and later become their defining class.

To realize the digital signature, there are defined two functions, which will be called depending on the particular case. Functions will be discussed in detail in the following paper. ActiveX controls have an advantage since they enable the implementation of graphical user interface within them. This facilitates the users and makes control more usable. In our case, the control has in itself an implemented graphical user interface for the user. Through this interface the user is given the signal that the signature is being generated until the entire process is carried out.

Functions of the ActiveX control

Two functions were carried out in order to realize the digital signature in ActiveX control. Both functions have input parameters but differ in their number. Functions are named as Nënshkrimi and NënshkrimiSN. To realize the signature is used PKCS#7 standard (details of the standard [5]). Through this standard can be achieved compatibility with different systems in the future or in terms of advancing the current solution.

```
public string NenshkrimiSN(string SN,string txtTekstiPerNenshkrim)
{
        X509Store store = new X509Store(StoreName.My);
        store.Open(OpenFlags.ReadOnly);
        if (store.Certificates.Count > 0)
        {
                X509Certificate2 certSign=null;
                foreach (X509Certificate2 certSign1 in    store.Certificates)
                {
                        if (certSign1. Thumbprint == SN)
                        {
                                certSign = certSign1;
                                break;
                        }
                }
                string strPlain = txtTekstiPerNenshkrim;
                byte[] bytPlain = Encoding.UTF8.GetBytes(strPlain);
                byte[] rezultati;
                try
                {
                        rezultati = SignMsg(bytPlain, certSign);
                        store.Close();
                        return Convert.ToBase64String(rezultati);
                }
                catch (Exception ex)
                {
                        return ex.ToString();
                }
        }
        else
                return "0";
}
```

Listing 1: Code of the function NënshkrimiSN.

NënshkrimiSN is a function which returns a string in Base64 value of the digital signature. This function takes two string parameters. The first parameter named as SN means thumbprint certificate for signature. The second parameter txtTekstiPerNenshkrim means for signature or plain text Text for signature. The procedure here is thinking of signing a little different in order to meet the need for a valid signature on the web. The exchange of certificates is done in the beginning, then, the client and server handshake which has SSL protocol implemented. Precisely this is the starting point and basis of the use of certificates and their validation. The first step is to check if the user is using an encrypted secure channel to communicate with the server. This is done by checking whether the URL contains https requesting a reserved word that means this protocol, which is expressed in code below.

```
if (Request.ServerVariables["HTTPS"] == "off" || Request.ServerVariables["HTTPS"] == "")
{
        Response.Redirect(Request.Url.AbsoluteUri.Replace("http://", "https://"));
}
```

Listing 2: Security checks if the user is using secure HTTPS channel

In this way, if the https protocol is not used, the server then automatically redirects to this protocol. The example given earlier shows how this can be done in C# programming language.

Once the client has chosen the certificate to present to the client, the process goes on. It should be noted that the IIS server has the possibility of such configuration to require the certificates and to be mandatory for the client or can be left in the mode as an optional feature. As a result, it is necessary to check if the client certificate is present. Such control is easy and accomplished as in Listing 3 below.

```
HttpClientCertificate cert = Request.ClientCertificate;
if (cert.IsPresent)
{
}
```

Listing 3: Checking through the code if the user has provided a certificate

The communication of browser with ActiveX control is done through JavaScript as a lightweight, interpreted, object-oriented language with first-class functions, most known as the scripting language for Web pages, but used in many non-browser environmentsas well such as node.js orApache CouchDB [12]. This solution is implemented on student management system in the Faculty of Electrical and Computer Engineering where student marks are digitally signed. In Figure 2 is depicted appearance of control during signing of a certain mark and complete page interface.

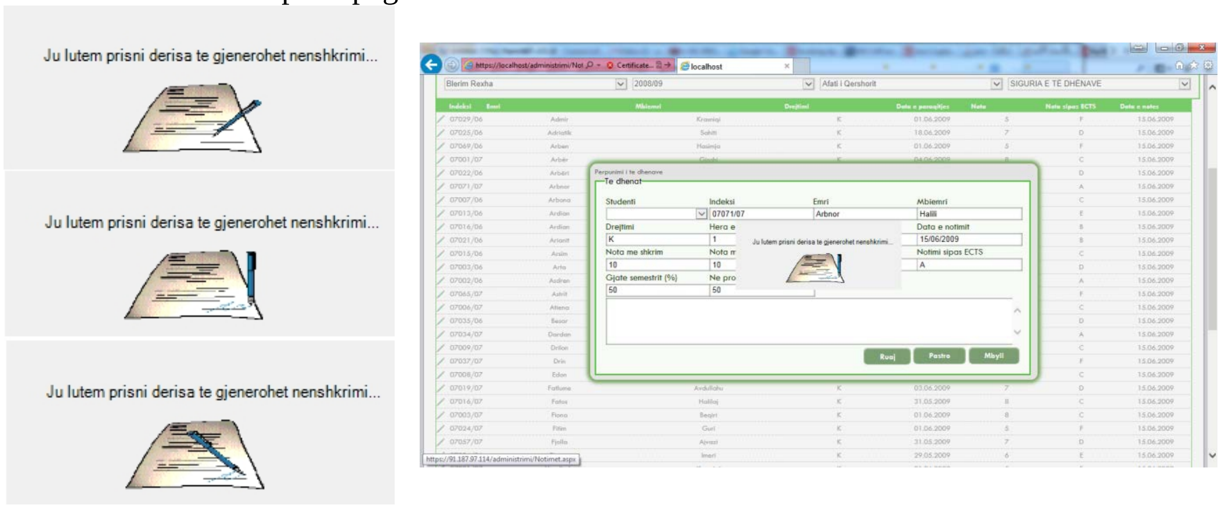
Ju lutem prisni derisa te gjenerohet nenshkrimi...
Ju lutem prisni derisa te gjenerohet nenshkrimi...
Ju lutem prisni derisa te gjenerohet nenshkrimi...
Te dhenat
Studenti
Indeksi
Emri
Mbiemri
Ruaj
Pastro
Mbyll

Figure 2. Sequence of ActiveX controls during signing

Compatibility with browsers

The problem with ActiveX control is the compatibility or suitability of platforms and browsers. Initially, the adaptability initially is treated with browsers from where the rest goes to suitability for different platforms or systems. The ActiveX control is not supported by different browsers which differ from Microsoft Internet Explorer. This makes it more difficult since there is an impose by Microsoft to use only their technologies.

In order to have the solution to this problem it is necessary to follow a different approach. ActiveX problem is a problem that has plagued earlier software engineers and has challenged them. This is because it is a barrier for other applications and compilation in other technologies is undoubtedly with higher costs. A more efficient solution is that other browsers needs to become like Internet Explorer in terms of structure. This is possible through the installation of additional modules also known as browser plug-ins.

One of the most used Plug-in in the market that simulates the behavior of Internet Explorer to other browsers is IETab plug-in. This plug-in is available for different browsers and is free of charge. IE Tab for Chrome is a browser add-in that allows users to use Internet Explorer to submit web pages tab. Some of the most common uses of IE Tab are [14]:

- Presentation of Web pages that require execution of ActiveX controls
- Testing of web pages with IE to render machine
- Use Windows Explorer to locate image files, with full support for Explorer icons, menus etc..
- Using Outlook Web Access

- Using Sharepoint properties intended only for Internet Explorer.

Same usage can be done for other browsers such as Firefox. Installing this add-on is done very easily and as said earlier is free.

Implementation of the solution through Java Applet

Another alternative for the implementation of digital signature is done through Java Web Applet.

Building Java Applets

Java applet as a Java code which is executed in the browser is certainly a strong element. Java Applet usually is embedded in a web page and is executed on the browser [13]. Java applet came out as a result of the total failure in need fulfillment by ActiveX control. The construction or development of Java Applet was done based on ActiveX control. These controls are developed taking into account the recognized standards for data security in order to be compatible with each other in order to generate data which are needed to generate digital signature.

To realize the applet the researchers used NetBeans IDE 7.0.1. NetBeans IDE is one of the most popular programming languages in Java and provides an easy interface to encrypt and many opportunities for different types of applications

The class for generating digital signature

Class for digital signature generation is appointed as clsSignature. This class contains two functions for signature generation. However, with a great interest is to discuss the function which makes sealing and which is used in the applet. The function code for the signature looks like Listing 4.

```
public String Signature(String strPlainText, PrivateKey _privateKey,X509Certificate _publicCertificate) {
        String str = "";
        try {
                if (_privateKey == null) {
                        return "";          }
                byte[] data = strPlainText.getBytes();
                Signature signature = Signature.getInstance("SHA1WithRSA");
                signature.initSign(_privateKey);
                signature.update(data);
                byte[] signedData = signature.sign();
                X500Name                              xName                              =
X500Name.asX500Name(_publicCertificate.getSubjectX500Principal());
                BigInteger serial = _publicCertificate.getSerialNumber();
                AlgorithmId digestAlgorithmId = new AlgorithmId(AlgorithmId.SHA_oid);
                AlgorithmId signAlgorithmId = new AlgorithmId(AlgorithmId.RSAEncryption_oid);
                SignerInfo  sInfo  =  new  SignerInfo(xName,  serial,  digestAlgorithmId,  signAlgorithmId,
signedData);
                ContentInfo cInfo = new ContentInfo(data);
                PKCS7 p7 = new PKCS7(new AlgorithmId[]{digestAlgorithmId}, cInfo,
                                new java.security.cert.X509Certificate[]{_publicCertificate},
                                new SignerInfo[]{sInfo});
                ByteArrayOutputStream bOut = new DerOutputStream();
                p7.encodeSignedData(bOut);
                byte[] encoded = bOut.toByteArray();
                str = Base64.encode(encoded);
        } catch (Exception ex) {
                System.out.println(ex.getMessage());          }
        return str;
}
```

Listing 4: Code for signature function in Java

The function takes three input parameters of type String, PrivateKey and X509 Certificate. From the type it can be seen that is plain text, the private key for digital signature and public certificate for eventual verification of the signature and other data for the signatories. The function returns the signature as information of type string similar as in the case of ActiveX control.

In order for the digital signature to be compatible it is important to follow recognized standards mentioned earlier, mainly to PKCS # 7 messages for digital signatures. The procedure translated into code in order to maintain this standard is provided via the following code. In Java, signature and other security elements are enabled classes available through built-in, java.security and sun.security as they are explained in official Java site [18].
PKCS # 7 in Java is like a separate class. Its initiation is done through several parameters. In this case initiation is done in Listing 5.

```
PKCS7 p7 = new PKCS7(new AlgorithmId[]{digestAlgorithmId}, cInfo,
new java.security.cert.X509Certificate[]{_publicCertificate},
new SignerInfo[]{sInfo});
```

Listing 5: Initiation of class object that makes PKCS # 7 signed message structure according to this standard

The first attribute indicates what algorithm will be used to generate the hash function. The second attribute indicates the element mentioned earlier to the relevant standard PKCS # 7 which is ContentInfo [5]. The next attribute is public certificate and the last element is SignerInfo. Each of these attributes are initiated and completed with relevant data earlier in the code. Finally, similarly to ActiveX controls, but this time the syntax of Java converts bytes to Base64 string and returns that value as a result.

Compliance with platforms and browsers

Adaptability is the main element of their development of solution based on the applets. The adaptability will be discussed here in two aspects, in terms of platforms or operating systems and browsers terms.

In terms of platforms, a detailed study was conducted by the researchers to adapt to different platforms and work without having to make additional configurations. The basic advantage of Java is that it provides support for various platforms. It is only required to have installed Java Virtual Machine and the execution is provided. However the problem is not limited to the execution, it is continuous.

Fundamental challenge to the adequacy of the platform lies in obtaining digital certificates. For different platforms it is followed a different approach by local key store. The first step toward solving this problem is to identify the mode of access to appropriate storage platforms. Once the ways of access have been identified, through the code has been verified the operating system and the approach relevant for that operating system. This approach is shown in Listing 6.

```
KeyStore keystore;
if (System.getProperty("os.name").contains("Windows")) {
        keystore = KeyStore.getInstance("Windows-MY", "SunMSCAPI");
} else if (System.getProperty("os.name").contains("Linux")) {
        keystore = KeyStore.getInstance("pkcs12");
} else {
        keystore = KeyStore.getInstance("KeychainStore", "Apple");
}
```

Listing 61: Use of different platforms to access the KeyStore

Access to the store is enabled by determining the type of store. In the case of a Windows operating system, the type to be marked for access in the store is Windows-MY whereas as a service provider is SunMSCAPI. SunMSCAPI provider enables applications to use the JCA / JCE API to access cryptographic libraries, certificates stores and the holder of the keys to the Microsoft Windows platform. SunMSCAPI provider itself does not contain cryptographic functionality, but is simply a connecting channel between the Java environment and cryptographic services on Windows [16].

In this part it is provided full support for the Windows platform and certificates installed on it. In the following branching are treated the systems based on Linux such as Ubuntu. Keystore in this case based on the type PKCS #12. This standard defines the structure of the digital certificate. When it comes to Linux there are various possibilities of creating personal KeyStore , in which was left in a field that allows you to write a personal path to the keystore.

The last part of the branch has to do with the Mac operating system. To have access to this system is necessary to approach KeychainStore type as permanent store on the Mac and Apple's provider.

This allocation enables the most popular systems (platforms) in the world market to be fully supported. The next step before using KeyStore is uploading them in the application/applet. In order for the load to be successful it is important to check whether we are dealing with Linux systems and if the path to the keystore is listed on the applet interface areas. Otherwise the applet allows marking the password for the specified keystore.
Consequently, the execution and the results will be the same in three most popular platforms worldwide. The following figure show how such a feature is visually seen[166].

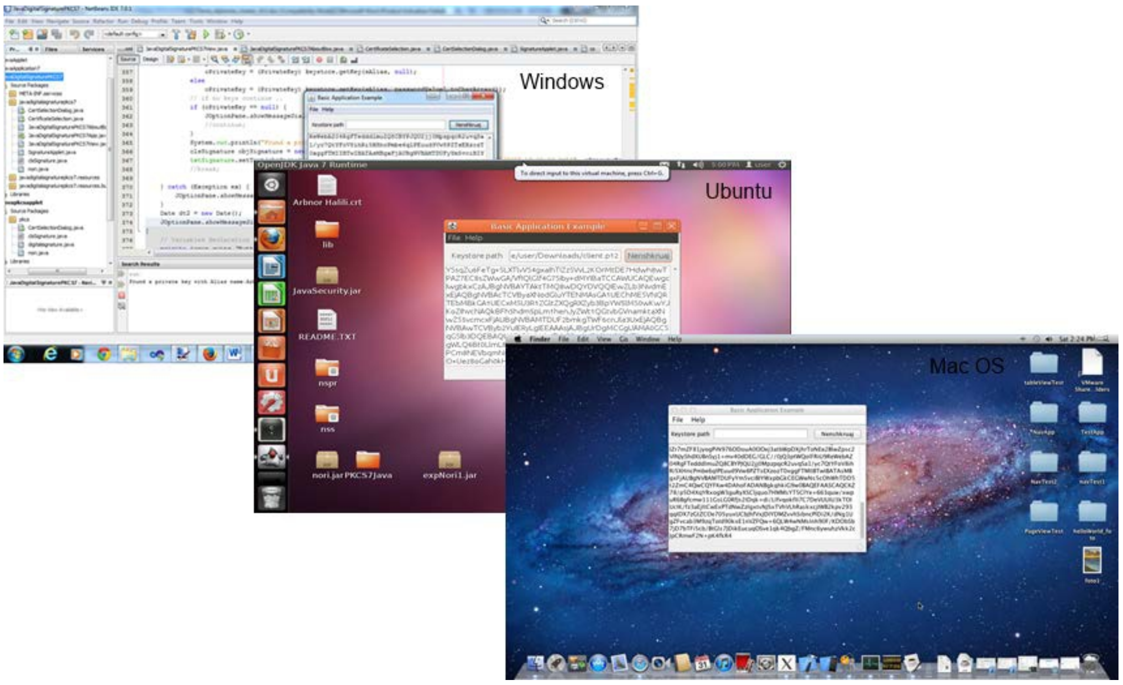

Figure 3. Screenshots from different Operating Systems (Windows, Ubuntu, Mac)

Another concept that should be discussed here is the suitability for different browsers. This is determined precisely by using Java technology which is known as cross platform and Java applets known as supplements that are supported by all popular browsers.
Also here the communication with browsers is carried out through JavaScript.

[166] In different platforms it was tested as Java Application, not as Java Applet embedded in specific page. Functionality and other properties are the same as it was Java Applet. This was done in order to have easier testing and deploying.

Conclusion

In this paper the main work consisted in studying and solving the problem of digital signature on the web. The paper summarizes the issues regarding the use of digital signature on the web starting from the stating the problems and possibilities for implementation, and next part presents a concrete implementation which gives two choices .

As a conclusion, it is worth noting that the digital signature format presented in the paper is unique realization and it was implemented in Kosovo for the first time. The contribution of this paper can be summarized in the following several points:

- Implementation of unique digital signature on the web through controls developed by the researchers, meanwhile respecting the internationally recognized standards for data security.
- Implementation in open source technology and closed source (paid) including Java applet and ActiveX control respectively.
- Increased level of system security in Student Management System at the Faculty of Electrical and Computer Engineering, University of Pristina.

Despite the work and research done in this regard there are still possibilities to expand the solution. One of the aspects for further research and work is related to the optimization of automation for Linux-based systems, mainly because of the diversity of these systems. Another important work that can be done is the expansion of the solution in order to support the most popular mobile platforms such as Android, iOS and Windows Phone .

References:

Jonathan Katz, Digital Signature, Springer, 2010

Samli, R., Digital Signature in the way of law, Iinternation Journal of Electronics; Mechanical and Mechatronics Engineering Vol.2 Num.2 pp.(172-179)

Rexha, B., Lajqi, H., Limani, M., Implementing Data Security in Student Lifecycle Management System at the University of Prishtina, 2010

Alain Hiltgen, Thorsten Kramp, Thomas Weigold, Secure Internet Banking Authentication, 2005

RSA Laboratories, PKCS #7: Cryptographic Message Syntax Standard, 1993

Maffeis, S., Client/Server Term Definition, Encyclopedia of Computer Science, 1998

Kevin Fu, Emil Sit, Kendra Smith, Nick Feamster, Dos and Don’ts of Client Authentication on the Web, MIT Laboratory for Computer Science

Heyning Cheng, Ron Avnur, Traffic Analysis of SSL Encrypted Web Browsing

The Electronic Signatures Regulations 2002, http://www.legislation.gov.uk/uksi/2002/318/pdfs/uksi_20020318_en.pdf

MSDN Library, an essential source of information for developers using Microsoft® tools, products, technologies and services - http://msdn.microsoft.com/en-us/library/system.security.aspx (visited August, 2013)

MSDN Library, an essential source of information for developers using Microsoft® tools, products, technologies and services - http://msdn.microsoft.com/en-us/library/aa751972(VS.85).aspx (visited August, 2013)

Mozilla Developer Network - Javascript, https://developer.mozilla.org/en-US/docs/Web/JavaScript (visited August, 2013)

Java Applets, http://docs.oracle.com/javase/tutorial/deployment/applet (vizituar në gusht, 2013)

IETab.net, http://www.ietab.net (visited August, 2013)

Introduction to ActiveX Controls, http://msdn.microsoft.com/en-us/library/aa751972%28v=vs.85%29.aspx (visited August, 2013)

The SunMSCAPI Provider, http://docs.oracle.com/javase/6/docs/technotes/guides/security/SunProviders.html#SunMSCAPI (visited August, 2013)
X.509 Certificate and Certificate Revocation List (CRL) Extensions Profile for Personal Identity Verification Interoperable (PIV-I) Cards, Federal PKI Policy Authority, 2010
Java Security, http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136007.html (visited September, 2013)
Pop, P., Comparing Web Applications with Desktop Applications: An Empirical Study, Linköping University