An Architecture for Automated Verification of Academic Testimonials in E-Learning

Kh. Amirul Islam, Akash Nag, Sunil Karforma, Sripati Mukhopadhyay

Abstract: Universities offering e-learning courses often provide their students with a hard copy of the marksheet. When that same student wants to apply for a job through the online application portal of a company, he/she must scan the marksheet and upload the scanned copy. This is a nuisance because there can be many such marksheets and not everyone has access to a scanner at home. The candidate is also required to provide the name of the University which issued the degree as well as the marks obtained, because these information cannot be extracted from the scanned marksheet image using OCR with 100% success rate due to many factors including: varying marksheet formats, presence of background watermarks, differing fonts, loss in quality during scanning, etc. The company must now manually verify each such application by matching the entered marks against the marks printed in the marksheet, which is a tedious process. In this paper, we propose an alternative approach where the data printed on the marksheet is also embedded in a digital copy of the marksheet. This digital copy, in the form of an image, can then be downloaded by the students from the University portal thereby eliminating the need for scanning. Furthermore, when this image is uploaded, the company, i.e. job provider, can easily verify the information by invoking a standard API exposed by the University (or some nodal agency), which will then extract the embedded information. This eliminates the need for any manual verification and the entire process is automated, simple, fast and hassle-free. Security features are also inherent in our approach thereby reducing any chances of fraud.

Keywords : steganography, e-learning, LSB steganography, API

I. INTRODUCTION

Steganography may be used as efficient data security mechanism for hiding intellectual and valuable information of e-learning resources such as marksheets, admit cards, certificates, lecture materials, etc. by embedding data within another message, called the cover.

Depending on the type of cover, we have image steganography, text steganography or even audio/video steganography. When the embedded data is used to identify the source of the message or in some way ensure authentication, the technique is called watermarking rather than steganography. In image steganography, one of the most popular and simple techniques is to use the least significant bit (LSB) of the color value of each pixel to embed information. Depending on how many bits are replaced, we have LSB-1 or LSB-2 steganographic approaches. Since the more significant bits are left unchanged, the change in the image is not noticeable by the naked eye leaving the causal error no room for suspicion. In color images, there are 3 channels, namely red, green and blue. Each of these channels usually contains 8 bits of information, and therefore, using LSB-2, we have the storage capacity of storing 6 bits per pixel. Alternatively, we may employ masking and filtering approaches, or other variations of selection mechanisms on the basic LSB scheme.
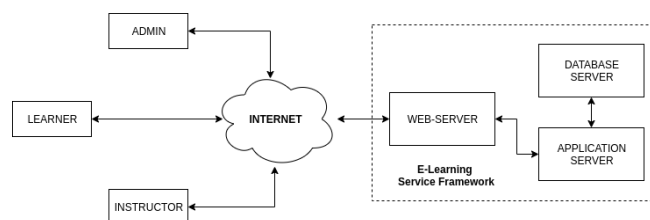

Fig 1. Typical E-learning framework

In Fig. 1, we present a typical architecture of an e-learning system. It is based on the client-server model, where the client can either be a learner/student, an instructor, or an administrator. The administrator is responsible for creating courses, generating and distributing marksheets, admit cards, etc. The instructors are responsible for creating and delivering lectures & lecture-materials, designing question papers for examinations, etc. The learners are the primary users of the system, who enroll in courses, watch lecture videos, sit for examinations and receive certificates on qualifying those examinations. In this paper, we focus more on the problems faced by a learner in an e-learning environment when they apply for jobs after completing their courses. The problems of the students revolve around uploading marksheets and certificates to job portals, while the more important problem of verification of these testimonials is faced by the job providers. In Section 1.A we discuss the first problem, while we discuss the latter in Section 1.B.

A. Problems faced in testimonial uploading

Students often get a hard copy of their marksheets after completing their online courses. When they subsequently apply for a job online, they are required to upload a scanned copy of the marksheet. Each candidate typically has more than one marksheet (e.g. starting from their secondary to post-graduation/Ph.D.) to scan, and not everyone has access to a scanner at home.

With deadlines approaching, the candidates are hard-pressed to get their scans ready on time for upload. Furthermore, there is no standard among companies for accepting these scanned copies, and the required file format (jpg/pdf) as well as the file-size & resolution limits vary from one portal to another. The candidate then has to use image processing software to edit the scanned images to fit within the prescribed guidelines.

B. Problems faced in testimonial verification

On the other end of the chain, the company (i.e. job provider) is also looking forward to a tedious job of manually verifying the marks obtained by matching it against the uploaded scanned copy of the marksheet. Usually this job is outsourced to a third company, which then typically contacts the concerned department of the University to verify the marksheet. The departmental staff must then manually and tediously verify the signature, date, and marks against its own record. This process cannot be automated through the use of OCR because of the following factors:

1. OCR software do not have a 100% recognition rate
2. Various marksheets have various formats
3. The marksheet contains various graphics, watermarks, signatures, and tables. This makes OCR software incapable of extracting information successfully.
4. Even if data is extracted, the semantics cannot be determined automatically. e.g. an extracted number can either be a full-marks, or obtained marks, or even the the paper code. Knowing which data is which is important if we want to automatically verify these.

To alleviate all of these problems, we propose an architecture of testimonial verification that is simple, fast, efficient, secure and hassle-free. The architecture involves a common open standard that all job portals & Universities should adhere to, thereby freeing the candidates from having to scan & then edit their scanned copies, and also freeing the companies from having to manually verify each application.

This paper is organized as follows: in Section 2, we describe existing work in this domain and we describe our proposed method in Section 3. In Section 4, we discuss the results obtained from our method, and the advantages of using this against existing procedures. Finally we conclude with our closing remarks and avenues for future enhancements in Section 5.

II. PRIOR WORK

Extensive research has already gone into image steganographic approaches. In the most basic LSB scheme, a pseudorandom sequence generator is employed to determine whether to increment or decrement the LSB of the next pixel [1]. This basic scheme has been augmented with masking and filtering [2]. In Bailey et al. [3], we find a comparison between seven different LSB techniques both for grayscale and color images. Soft computing tools like Genetic algorithms [4] have been used [5] as well to create a substitution table for LSB embedding purposes. This table is used for transforming the pixel value so that the transformed value mostly closely resembles the original while still embedding data. It has also been observed that using simple bit alteration, the MSE (mean square error between the original and the modified pixel values) can be reduced [6]. Another very successful algorithm in this domain is the Optimal Pixel Adjustment Process (OPAP) [7] and using randomization by Dey et al. [8]. Security and Authentication aspects using steganography in the domain of e-learning had been studied by Banerjee et al [9], while data storage security using steganography was studied by Garg & Kaur [10]. However, the authors are unaware of any complete framework driven approach that have been proposed which facilitate automatic marksheet verification.

III. METHODS

The proposed architecture will now be discussed in the context of the following components:

1. An open format for textual representation of testimonials
2. Digital copy generation
3. The online portal for students
4. The public API for automated verification
5. The data extractor
6. The job portal

To understand the role of these components better, it is important to understand the key stakeholders in the system:

1. **The University/E-Learning Provider:** This is any Institute that offers courses to students.

2. **The Higher Education Agency:** This is any nodal (usually Government) agency that will grant affiliation to all Universities and is the key player in our system to enforce a common standard in API design and digital marksheet preparation. It also assigns a unique University code to each affiliated University, and unique codes for each separate exam/degree name. e.g. in India, this can either be UGC or AICTE.

3. **The Students/Candidates:** These are the students enrolled in the courses provided by the University. On completion of their course, they shall receive both a hard copy and a digital copy of their marksheet, which they are free to upload to any job portal whenever required.

4. **The Job Provider/Company:** The company is any job provider having an online portal where candidates can register themselves, upload their details and marksheets, and apply for any vacant positions within that company.

We will now discuss each component of our system. The overall architecture of our system is illustrated in Fig. 2.
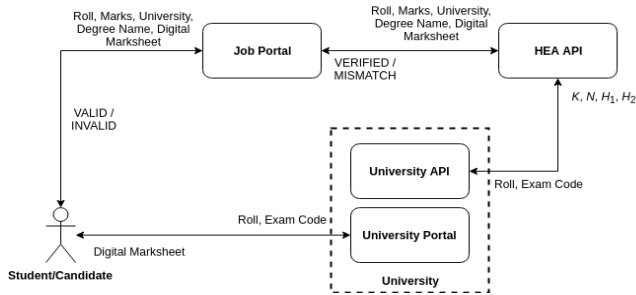
Fig. 2. The system architecture

A. An open format for textual representation of testimonials

As part of our system, the HEA shall enforce a common and open text format for storing data of any academic testimonial. This format can be either XML based [11] or JSON based [12]. There shall be separate formats for admit-cards, marksheets, certificates, etc. A typical marksheet format in XML can be as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<marksheet issuer="The University of Technology"
degree="B.Sc. (Honours)" month="June"
year="2019" type="semester">
<student>
<name>John Doe</name>
<roll>16SH/116/144</roll>
<registration>20164411668415</registration>
</student>
<marks>
<semester>
<paper theory-marks="40" practical-marks="20"
internal="15" theory-credits="4" practical-
credits="2">
<code>CC-1</code>
<name>Programming in C/C++</name>
<theory-marks-obtained>35</theory-marks-
obtained>
<practical-marks-obtained>18</practical-marks-
obtained>
<theory-credits-obtained>5.2</theory-credits-
obtained>
<practical-credits-obtained>1.8</practical-
credits-obtained>
<internal-marks-obtained>10</internal-marks-
obtained>
<total-marks-obtained>63</total-marks-obtained>
<total-credits-obtained>5.1</total-credits-
obtained>
</paper>

...

</semester>
</marks>
</marksheet>
```

B. Digital copy generation

Along with the usual hard copy of a marksheet, the University shall also generate digital copies of the same, seemingly identical in all respects to the issued hard copy. This generation procedure is now described in this section. The University shall maintain a database where the roll number, examination/degree code/name, and marks of the students are stored. Along with this, there is provision to store four more integers, named N, K, H1 and H2.

The HEA shall maintain a repository of 2048-bit DSA [13] public keys of every University affiliated to it,

against a unique code for each University. There shall also be unique codes for each degree in that University.

The University shall use an automated XML or JSON generator to generate the textual representation of the marksheet, as detailed in Section 3.1, using the data stored in the database. This generator is trivial to code in any standard programming language. The generated data is then compressed using the LZ-77 [14] algorithm, and the resulting compressed data is converted to binary. This binary stream is then signed by the University using its own DSA private key, and the resulting augmented data is further compressed using the Pack-Bits [15] algorithm. The length of this resulting binary stream, B, is the value of N. The SHA-256 hash of this data is the value of H1.

The University shall use a custom software that prints the data stored in its database onto a watermark/background image of the marksheet, complete with the table structures and signature. This software is also trivial to write as it requires only the knowledge of the specific marksheet format for that University. This image will now be used in the next step.

We have developed a data embedder that embeds the compressed binary data B into the image using a modified LSB steganography approach. The algorithm for this program is described in Algorithm 1. The algorithm takes as its input the compressed binary stream B, and the marksheet image. The algorithm works by generating the random seed K using any standard pseudorandom sequence generator [16]. If the length of B is not a multiple of 4 bits, the data stream is padded to make it such; however N will remain unaffected and will still store the length of the unpadded data. Using the seed K, a sequence of N/4 random integers in the range 0 to M-1 are generated, where M is the resolution (product of width and height) of the image. These random integers are mapped to (x,y) pixel coordinates on the image. Each pixel contains a 24-bit color information consisting of three 8-bit color channels, namely red, green and blue. Since the human eye is most sensitive to fluctuations in the green channel, we have decided to embed data into the red and blue channels only, taking 2 bits from each, thereby getting a storage capacity of 4 bits per pixel. The binary data B is then divided into groups of 4 bits and each group is embedded in a random pixel determined by the random sequence. The embedding process is illustrated in Fig. 3.
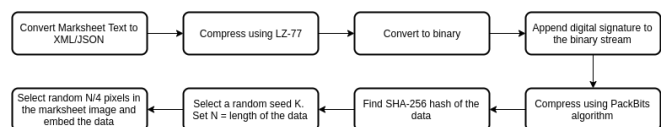

Fig. 3. The data embedding process

The resulting image, hereinafter referred to as the digital marksheet, after embedding is identical (to the human eye) to the original one. This image is then saved in the database, and the 256-bit SHA [17] hash of this image is the value of H2.

C. Online portal for students

The University shall provide an online portal where any student can provide his/her roll and registration numbers, date of birth, phone number and select the exam/degree code, and can then retrieve the digital marksheet. The students can then upload this to any job portal.

D. Public API for automated testimonial verification

The University shall expose a public API function that takes as input the following:

1. Exam code
2. Student roll
3. Student registration

This functionality will be protected by additional authentication, typically an OTP sent to the registered e-mail/phone.

This function will be invoked by the HEA server. To verify the authenticity of the HEA server, the API shall use a challenge-response system, where the API throws a challenge and the HEA server responds with a signed response. The response and the signature is verified by the API after which it takes the 3 inputs as listed above. The function searches the University database for a matching record and returns the K, N, H1 and H2 values if found. If not found, it reports an error message.

E. Data extractor and verifier

The HEA server shall maintain a data extractor and verifier function which can be accessed publicly by any private or public Institution through an API. This API takes as input the following parameters:

1. Roll number
2. Registration number
3. University code
4. University exam/degree code
5. Marks obtained
6. Full marks
7. Digital copy of the marksheet

The HEA API invokes the appropriate University API and negotiates with it using a proper challenge response. The University server is located from the University code supplied by the caller. The HEA maintains a database consisting of all affiliated Universities with their codes, server URLs, exam codes, etc. The HEA then sends the Roll number, Registration number, and University exam code to the University API, which then returns the K, N, H1 and H2 values through a secure HTTPS channel.

Using the K and N values, the data is extracted using the decoding algorithm, complementary to Algorithm 1. The extracted data and the digital marksheet itself are verified using the H1 and H2 hashes. The marks obtained and full marks are then matched against the extracted data. If they do match, a "VERIFIED" response is sent back, else a "MISMATCH" response is sent to the caller.

F. The Job Portal

Every company wishing to avail the facilities of the automated testimonial verification system shall have a job portal where candidates are required to register

themselves and enter their marks, roll numbers, University name, etc. The candidates are also required to upload the digital marksheet against each degree obtained. The portal then invokes the HEA API by sending all of these information. The HEA API responds by either a VERIFIED or MISMATCH message, which is then instantaneously reported back to the user.

Algorithm 1. Data embedding

```
procedure EMBED_DATA(image, width, height, data):
      K = RANDOM_INT( )
      INITIALIZE_RANDOM_GENERATOR(K)
      N = LENGTH(data)
      M = width * height
      for i=0 to (N/4)-1 do
            R = RANDOM_INT(0, M-1)
            x = R mod height
            y = R / height
            EMBED_IN_PIXEL(image, data, 4*i, x, y)
      return <image, K, N>
end procedure


function EMBED_IN_PIXEL(image, data, start, x, y)
      <R, G, B> = EXTRACT_COLOR(image, x, y)
      high_order = (data[start] << 1) |
      data[start+1]
      low_order = (data[start+2] << 1) |
      data[start+3])
      R = (R & 0xFC) | high_order
      B = (B & 0xFC) | low_order
      INSERT_COLOR(image, x, y, <R, G, B>)
end function
```

IV. RESULTS AND DISCUSSION

A. Results

The encryption algorithm (Algorithm 1) was implemented in Java on an Ubuntu 18.10 system running on Intel Pentium Quad-Core 1.6GHz processor with 4 GB memory. A sample image before data embedding and after embedding is shown in Fig. 3. Running times were under 1s but it will depend on the image resolution. The algorithm supports lossless BMP format images so that the data is preserved. The reported PSNRs varied between 150 and 216, and hence it can be safely said that picture quality does not degrade enough to be noticeable. In fact, any PSNR above 50 is typically considered undetectable by the human eye.

B. Comparison against existing data embedding approaches

We will now compare our approach against existing watermarking and steganographic approaches. In Table I we present the results of embedding data using our proposed technique in two types of testimonials, namely: admit cards and marksheets. Our approach is different to watermarking because the objectives of the two systems are completely different. Watermarking is used to verify the authenticity of a document, and possibly in DRM and digital forensics. On the other hand, our approach is used to eliminate the need for manual testimonial verification and OCR because both methods are either time consuming or error prone.

Compared to other simple steganographic approaches, our approach is superior.

If we compare our method with LSB-1 we see that it can encode 3 bits per color pixel, whereas our method can encode 4 bits per color pixel. But, we know that LSB-2 encodes 6 bits per color pixel, but the pixels selected are completely sequential and therefore, do not blend with the statistical distribution of the colors in the rest of the pixels. In our case, the pixels selected for modification are random and therefore there is no region to be found in the image that is statistically contrasting with the rest of the image. Moreover, since only 4 bits are modified rather than 6, the green channel is unaffected and the statistical distribution is also less affected.

Lastly, our system is not just a data embedding approach. Rather it is an entire architecture that can simplify the work for both candidates applying for a job, and for companies who up till now had to manually verify all testimonials.

C. Advantages and limitations of our approach
There are numerous advantages of our proposed approach and we will discuss these below.

C.1 *Advantages to Students/Candidates*
1. The candidates applying for a job will not require to get their hard copy marksheets scanned. This will save both time and cost as most people do not have access to a scanner at their residence.
2. The candidates will also be saved from having to use image editing tools to resize images, format them, etc. according to the company's requirements, because all marksheets will now be in a standard resolution.

The candidates can get a color true-copy printout of their marksheets from their digital marksheets. These digital marksheets will be printed with a DC ("digital copy") label that can also be taken to mean a duplicate copy. The hard copy will not contain this marker. Therefore, the candidate need not apply for a duplicate copy to the University and go through numerous formalities. They can just download and print the digital marksheet from the University portal.

C.2 *Advantages to Job Providers*
The job providing organizations will not have to manually verify each marksheet as the entire verification procedure is now completely automated.

C.3 *Advantages to Universities*
1. The University no longer needs to issue duplicate copies for students who have lost the original.
2. The University can even completely do away with printing original marksheets and everything can be done digitally, thus going paperless.

C.4 *Other advantages*
Apart from the above mentioned advantages, there are several security features as well as other advantages in our system, such as:
1. The hash values prevent unauthorized tampering and changing of the data.
2. The signed response by the HEA during the API call to the University ensures that only the HEA has access to the K and N values and no one else.

Therefore, the algorithm and the seed K are kept secure. This ensures the privacy of each candidate as no third party can access the candidates' details.
3. The random pixel selection ensures that the data cannot be extracted without knowing the seed K. For standard LSB systems, the data is sequentially stored from the start of the image and hence, extracting the data (if it is known that it is a stego image) is trivial.
4. Due to added compression capabilities, the volume of data that can be embedded is larger.

C.5 *Issues and Limitations*
There are some minor issues that may restrict wider adoption of our system, such as:
1. All Universities must provide an online portal for letting users download digital marksheets
2. The Universities must follow the same standard for embedding data.
3. The HEA must provide an online portal to handle all verification requests.

The major limitation of our scheme is that the system can only work with BMP images. We hope to rectify this issue in the future.

D. Design Considerations
Although it seems that the University server will be exhausted in terms of storage because of having to store the digital copies of thousands of existing and past students, it is not so. This is because, the entire data embedding and digital marksheet generation procedure can be done on-the-fly whenever a user makes a request to download the marksheet.

One may wonder about the purpose of embedding the data when the data itself can be retrieved from the University database, via the API, to verify it. This is because, the HEA being a nodal agency is always expected to have a sophisticated server architecture with firewalls and network intrusion detection software. However, each University may not afford such an elaborate security scheme. Verification of the data by the University would require accessing the marks database by the server, thereby exposing it to grave security risks of the marks being altered maliciously. In our scheme, the University shall maintain two databases: the marks database, and the verification database. The marks database shall not be connected to the cloud thereby keeping it offline. The verification database will not contain any marks but only the roll, registration and associated K, N, H1 and H2 values. This database is accessed by the API, thus defeating any hacking attempts to change the marks. In the unlikely event that these K and N values are altered, the job portal would report to the user that verification has failed. The user can then try to download the marksheet again from the University portal. Since these marksheets are generated on-the-fly, new K, N, H1 and H2 values would be generated by the embedding procedure and will overwrite any previously saved values in the database, thus restoring database integrity.

E. Security Considerations

Steganalysis is the art of detecting the presence of secret messages in an image. One of the simplest forms of attack is the Chi-Square attack. However, our proposed approach is immune to this attack as the Chi-Square test can only reliably detect the presence of secret data only if the embedding positions are known, e.g. sequential, etc. In our case, the data is scattered randomly and hence, is secure against this type of attack.

Other forms of attacks also exist but are meaningless in the context of our proposed system. This is because, in our system, the objective of inserting the data is not to hide it from any party. In fact, everyone is aware of the fact that steganography is being used. Rather, our purpose is two-fold:

1. To make the data easily retrievable for verification purposes
2. To make the data impossible to retrieve without proper authentication

The achieve the second objective listed above, we must understand that various steganalysis techniques can only detect the presence or absence of secret data. However, most techniques are unable to actually retrieve the data itself. Therefore, our system is secure against most of these attacks. Properly retrieving the data requires knowledge of the values of K and N, none of which are revealed to any third party.

V. CONCLUSION

Steganography is a useful tool for embedding data into non-noticeable parts of an image, thus saving space with no appreciable loss in quality. As a result, it can be a useful tool in the domain of e-learning security [18]. In this paper we demonstrated how steganography can be used effectively to alleviate a common real-life problem of manual testimonial verification, rather than using steganography for the more typical purpose of exchanging secret messages. We have proposed an architecture that is robust, secure and efficient. If this architecture is implemented across the entire academic field, testimonial verification can become totally automatic and instantaneous, thereby saving time. Also, scanning marksheets will become a thing of the past, saving time and money for the candidate. The system has numerous security features which prevent tampering with the digital marksheet, as well as prevent unauthorized access to the data embedded in it. Incidentally, this also eliminates various problems like loss of quality that occurs during scanning. Loss of the original marksheet by the students is also less problematic due to the fact that

Table- I: Performance of the proposed technique against standard LSB-2 steganography

Image	Dimensions	File-Size	PSNR	
			Proposed method	Standard LSB-2
[image]	1024 x 768	2,359,342 bytes (2.4 MB)	170.238	145.61
[image]	534 x 696	1,116,440 bytes (1.1 MB)	168.571	141.209

REFERENCES

1. Sharp, Toby. "An implementation of key-based digital signal steganography." International Workshop on Information Hiding. Springer, Berlin, Heidelberg, 2001.
2. Johnson, Neil F., Zoran Duric, and Sushil Jajodia. Information Hiding: Steganography and Watermarking-Attacks and Countermeasures: Steganography and Watermarking: Attacks and Countermeasures. Vol. 1. Springer Science & Business Media, 2001.
3. Bailey, Karen, and Kevin Curran. "An evaluation of image based steganography methods." Multimedia Tools and Applications 30.1 (2006): 55-88.
4. Mitchell, Melanie. "An introduction to genetic algorithms." MIT press, 1998.
5. Wang, Ran-Zan, Chi-Fang Lin, and Ja-Chen Lin. "Hiding data in images by optimal moderately-significant-bit replacement." Electronics Letters 36.25 (2000): 2069-2070.
6. Chan, Chi-Kwong, and L. M. Cheng. "Improved hiding data in images by optimal moderately-significant-bit replacement." Electronics Letters 37.16 (2001): 1017-1018.
7. Chan, Chi-Kwong, and Lee-Ming Cheng. "Hiding data in images by simple LSB substitution." Pattern recognition 37.3 (2004): 469-474.
8. Dey, Somdip, Kalyan Mondal, Joyshree Nath, and Asoke Nath. "Advanced steganography algorithm using randomized intermediate QR host embedded with any encrypted secret message: ASA_QR algorithm." International Journal of Modern Education and Computer Science 4.6 (2012): 59-67.
9. Banerjee, Soumendu, Sunil Karforma, and Akash Nag. "Applying LSB steganography for disseminating academic testimonials in e-learning and its authentication aspects." International Journal of Computer Trends and Technology 47.3 (2017): 170-175.
10. Garg, Nancy, and Kamalinder Kaur. "Data storage security using steganography techniques." International Journal of Technical Research and Applications 4.6 (2016): 93-98.
11. "XML 1.0 Specification". World Wide Web Consortium. (URL: https://www.w3.org/TR/REC-xml/) Retrieved 29 September 2019.
12. "A Modern Reintroduction To AJAX". (URL: http://javascript-coder.com/tutorials/re-introduction-to-ajax.phtml) Retrieved 29 September 2019.
13. Gallagher, Patrick. "Digital signature standard (dss)." Federal Information Processing Standards Publications, volume FIPS (2013): 186-3.
14. Ziv, Jacob, and Abraham Lempel. "A universal algorithm for sequential data compression." IEEE Transactions on information theory 23.3 (1977): 337-343.
15. "Technical Note: TN1023, Understanding Packbits". (URL: https://web.archive.org/web/20080705155158/http://developer.apple.com/technotes/tn/tn1023.html) Retrieved 29 September 2019.
16. Schneier, Bruce. "Applied cryptography: protocols, algorithms, and source code in C". John Wiley & Sons, 2007.
17. Eastlake, Donald, and Paul Jones. "US secure hash algorithm 1 (SHA1)." (2001).
18. Weippl, Edgar R. "Security in E-learning". Vol. 16. Springer Science & Business Media, 2006.
19. "DSpace: An Open Source Dynamic Digital Repository", D-Lib Magazine, January 2003.

AUTHORS' PROFILE

Mr. Kh. Amirul Islam completed his Bachelor of Computer Application from Dumkal Institute of Engineering & Technology under West Bengal University of Technology in 2012, and his Master of Computer Application from The University of Burdwan in the year 2015. He is currently pursuing Ph.D. in Department of Computer Science, The University of Burdwan in the field of information security. His research interests include e-learning, network security, cryptography, and steganography.

Akash Nag completed his M.Sc. in Computer Science from St. Xavier's College, Kolkata in 2013, and his Ph.D. from The University of Burdwan in 2018. His research interests include bioinformatics, algorithms, network security, programming languages and operating systems. He is currently a faculty member at M.U.C. Women's College, where he teaches undergraduates of B.Sc. Computer Science.

Prof. Sunil Karforma has completed B.E and M.E from Jadavpur University. He has completed Ph. D. in the field of Cryptography . He is presently holding the post of professor and head of the Department in the Department of Computer Science, The University of Burdwan. Network Security , e-Commerce , e- Learning and cryptography are his fields of interest in research area.

Prof Sripati Mukhopadhyay, M. Tech., Ph.D.. is a former professor and head of Department of Computer Science, University of Burdwan. He has served North Bengal University , Vidyasagar University , Visva-Bharti etc. He has 33 years of teaching and research experience. His research interests include Computational Intelligence, Software Engineering and Database Systems.