Teaching Cryptology At All Levels Using CrypTool

Rong Yang[1], Layne Wallace[2] and Ian Burchett[1]

Abstract - It is of increasing importance that we incorporate security and cryptology in both the undergraduate and graduate curriculums. This paper introduces cryptology in the framework of general cybersecurity and advocates that it is an appropriate mechanism for introducing security issues into the classroom at all level of the curriculum. A practical free software package called CrypTool which can be a major asset in any attempt to teach cryptology to a range of student audiences is presented. Applications and classroom experiences using CrypTool are discussed along with some student feedback.

Index terms – Cryptology, Education, Security

I. Introduction

One of the major challenges facing computer science today is the need for improvement in cybersecurity education for computer science majors, minors, and the overall student body. The signs of the need and measures to meet it are widespread [1, 2]. It is thus of increasing importance that we incorporate security in both the undergraduate and graduate curriculums - even non-technical users need to understand the importance of these concepts and how to use them to protect themselves and their employers. As one would expect, there is currently a good deal of discussion in the computer science community concerning how best to integrate these security concerns into the curriculum [3]. Cryptology is a fundamental aspect of cybersecurity which has long fascinated students and so is a natural candidate for providing exposure to these important issues. The difficulty of programming many cryptographic methods has hampered attempts to effectively introduce cryptology, especially at the lower levels. A comprehensive, animated, open-source piece of free software, CrypTool, alleviates many of these problems and is introduced in this paper.

II. Background

A. What is cryptology?

Cryptology derives from two Greek words, kryptós (meaning hidden) and lógos (meaning word). It is the science of secure (and often secret) communications, generally by means of a key, which is kept sufficiently secret for the data at hand. There are two major branches:

- Cryptography (from the Greek *kryptós* and *gráphein*, "to write") is the study of the principles and techniques by which information can be concealed in ciphers and later revealed by legitimate users employing a key, but for which it is either impossible or computationally infeasible for an unauthorized person to do so.

- Cryptanalysis (from the Greek *kryptós* and *analýein*, "to loosen" or "to untie") is the science (and art) of recovering information from ciphers without knowledge of the key.

B. A Little History

Cryptology originally began with providing security for written messages. Historically, a number of ingenious methods were used, including tattoos on the heads of slaves covered by re-grown hair and the clever Spartan Scytale device [4] shown in Figure 1 where a long strip of paper is wrapped around a rod with known diameter to both encode and decode the message.

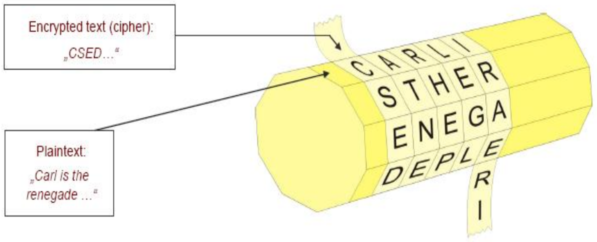


Figure 1: Spartan Scytale device

[1]: *Western Kentucky University, Bowling Green, KY42101*
[2]: *University of North Florida, Jacksonville, FL 32224*

C. Cryptology Today

Today cryptology has expanded and finds its primary applications in the electronic arena as a mainstay of cybersecurity in many areas:

- Phone cards, cell phones, remote controls
- Cash machines, money transfer between banks
- Electronic cash, online banking, secure email
- Satellite TV, Pay TV
- Immobilizer systems in cars
- Digital Rights Management (DRM) – CDs, DVDs, MP3s, etc.

III. CrypTool Software Package

CrypTool is free, open-source Windows GUI software designed for cryptography. It has been under development since 1998 and currently many people are working worldwide on its improvement and maintenance. Its catalog of state-of-the-art cryptographic functions is nearly comprehensive. It provides resources for both encryption and decryption and coverage from ancient times to the present. Features for cryptography include Caesar, Vigenère, Hill, Playfair, transposition, permutation, DES, AES and RSA. Cryptanalysis features include Entropy, Histograms, Brute force, floating frequency, n-gram analysis, auto-correlation, periodicity, factorization of RSA module, lattice-based and side-channel attacks.

CrypTool is billed as an “Educational Tool”. In support of that goal, it includes a number of unique features:

- A range of animations and demos (Caesar, Vigenère, Enigma, AES, RSA, digital signatures, side-channel attacks, secret sharing, number theory, and more).

- Extensive documentation. There is a comprehensive online help system as well as PDF documents which provide history, motivation, and additional details about cryptography.

CrypTool and its relatives are available for download from the website [5]. The CrypTool organization also runs the site CrypTool-Online [6] which allows for cryptographic exploration without having to install any programs locally, and features a password generator, password check, a number theory based game (TaxMan), and many useful links.

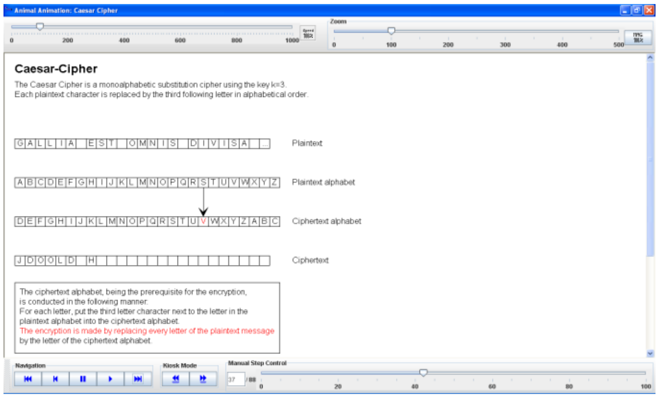
There have been some interactive tools [7, 8, 9] introduced for teaching cryptology, but in many cases they were designed for a specific encryption or cryptanalysis algorithm or, over time, have become unavailable. None provides comprehensive and extensive support materials and programs for teaching cryptology as does CrypTool.

IV. Lower Division Applications

The introductory computer security course is a challenge simply because of the vast amount and diverse nature of the material. To cover the breadth of computer security in one course, the topics must be covered quickly. This rapid coverage often leaves students with a limited understanding of specific aspects of computer security. There seem to be two options to provide a quality educational experience: reduce the number of topics covered in the introductory course or use tools (either online or installed on personal machines) to help present the material at the students’ convenience. Obviously, reducing the number of topics results in an incomplete view of computer security. However, the tools used need to be more than just an online ‘film strip’ video about the topic. In other words, to be effective in teaching computer security, the tools must have a hands-on, interactive flavor that encourages students to take an active part in their education. For the introductory computer security class, CrypTool is a good example of an application that provides both a quality educational experience and a depth of coverage that many tools lack.

With the step-by-step dialog control available in CrypTool students are able to see the impact and implications of different cryptographic algorithms. A typical programming assignment for cryptography is to code several of the simple encryption algorithms. One of the authors has used CrypTool as a design mechanism for the students. By watching CrypTool do the encryption students are able to get a more visual sense of the functionality than by simply reading through a textbook. Instead of taking three class periods to explain the simple encryption techniques, only one class period is needed. This has proven to be even more effective for students in computing fields such as Information Systems and Information Technology that do not have the mathematical background that Computer Science students usually have. Even these students are well prepared to understand many of the classical approaches described in general textbooks [10, 11, 12, 13, 14] and to experiment with and program them.

A typical example is the Caesar cipher, attributed to Julius Caesar in the 1st century BCE. It is a shift cipher, where each letter of the plain-text is replaced by the letter k positions after it (wrapping around back to the beginning after Z) to produce the cipher-text. Caesar actually used k = 3. For example, with the key k = 3, “Cryptology Rocks” becomes “Fubswrorjb Urfnv”. Animations illustrating the algorithm can help students understand the cipher methods easily. An animated screen from CrypTool is shown in Figure 2.



Caesar-Cipher

The Caesar Cipher is a monoalphabetic substitution cipher using the key k=3.
Each plaintext character is replaced by the third following letter in alphabetical order.

G A L L I A E S T O M N I S D I V I S A ...

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

J D O O L D H

The ciphertext alphabet, being the prerequisite for the encryption, is conducted in the following manner.
For each letter, put the third letter character next to the letter in the plaintext alphabet into the ciphertext alphabet.
The encryption is made by replacing every letter of the plaintext message by the letter of the ciphertext alphabet.

Figure 2: Caesar Cipher Animation

Tools are also provided to analyze various encryptions. A typical example is the use of a frequency histogram as shown in Figure 3 to figure out the key for a Caesar cipher. Since E is the most frequently occurring letter, spotting the most frequent letter in the histogram for the ciphertext will often reveal the key (here H is the most common letter, so E maps to H and the key is 3).

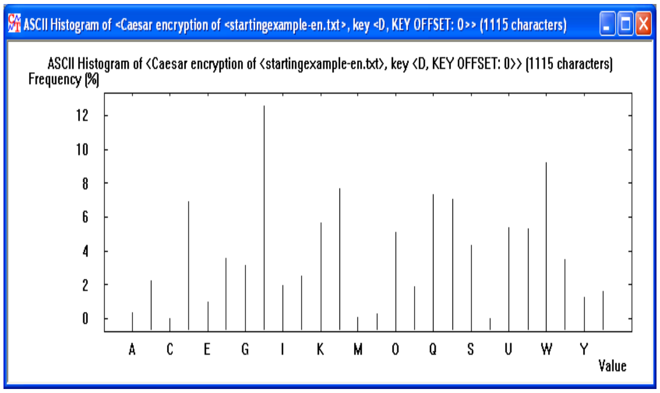


Figure 3: Frequency Histogram

CrypTool has also been used by one of the authors to demonstrate just a few of the myriad ways that a Caesar cypher can be adapted. Instead of having the students develop software for a set of Caesar variants, the students use CrypTool for development and testing. The cryptography concepts are the primary focus of the exercise instead of the coding. Analysis of the encryption method typically takes at least two classes to explain. With an animated tool such as CrypTool, the class discussion can focus on the meaningful analysis instead of the underlying code to do the analysis. After developing a Caesar variant, the students are asked to exchange encrypted output files so they are testing data from another student.

Similar experiments may be carried out for many of the classical ciphers. The Vigenère cipher, the Playfair cipher, the Hill cipher, and permutation encryption are all examples of techniques that may be approached with a minimum of mathematical background.

In addition to simple experimentation and associated decryption contests and the like, beginning computer science students are certainly capable of programming many of the classical algorithms which are at their heart rather simple.

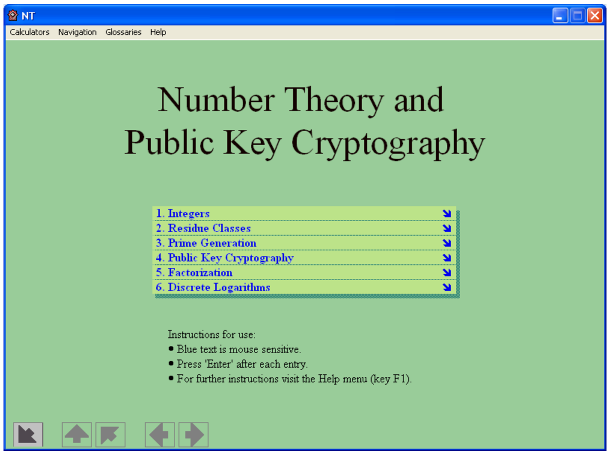
Another interesting class exercise is to have students devise their own encryption scheme. Then either they or others in the class can analyze its effectiveness.

V. Upper Division Applications:

At the upper division (junior or senior) level, students are ready to tackle more up to date encryption techniques. Most of these hinge on properties of very large integers and depend on the intractability of certain mathematical problems. Thus more mathematical sophistication is necessary. We outline a few typical examples.

A. Numerical Theory

In addition to providing implementations of the modern algorithms, CrypTool has many features to help the student fill in the necessary background. Figure 4 shows a feature provided by CrypTool for Number Theory and Public Key Cryptography.



Number Theory and
Public Key Cryptography

Instructions for use:
- Blue text is mouse sensitive.
- Press 'Enter' after each entry.
- For further instructions visit the Help menu (key F1).

Figure 4: CrypTool Features for Number Theory and Public Key Cryptography

The number theory learning tool allows students to gain a foundation in number theory, which is critical when trying to understand and implement advanced cryptography. The relevant elementary number theory is covered, explaining divisibility, primes, Euclid’s Algorithm, and least common multiples.

Residue classes are covered, involving congruences, prime residue classes, subgroups, and primitive residue classes.

Prime generation is demonstrated, with the techniques for finding primes shown. Fermat’s test for primes is demonstrated, showing that the test is probability based, etc. The Miller-Rabin Test is shown, a test based on the Fermat test, but relying on the unproven generalized Riemann hypothesis. Detailed descriptions about those tests techniques can be found in [11].

Public key cryptography, including the basic concepts and procedure, RSA, Rabin cipher, Diffie and Hellman, and more are covered, with demonstrations of the processes of all of these.

Factorization is included, with Pollard Rho, Fermat, Pollard’s p-1, and Quadratic Sieve Factorization all demonstrated and explained.

Lastly, discrete logarithms are covered, showing how to find exponents, Shank’s Babystep-Giantstep method, Pollard’s Rho Algorithm, and other methods.

By teaching the fundamentals of number theory pertaining to cryptography, the students may review the lessons in class on these subjects, or refresh their prior knowledge on the theory, or even to learn these subjects for the first time in preparation for assignments on these subjects.

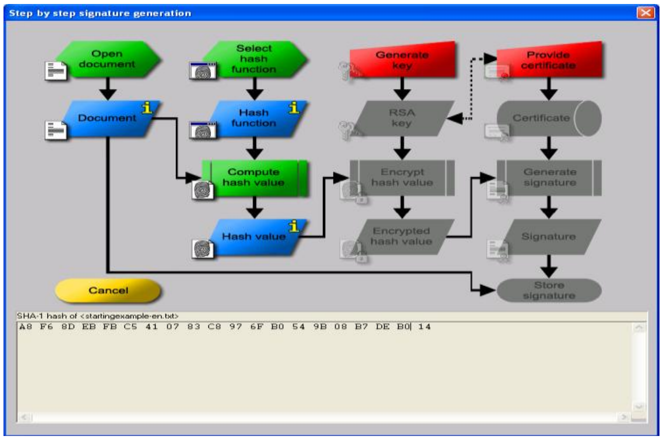
A good understanding of these subjects is critical to success in learning and implementing cryptographic methods, so CrypTool can be pivotal in students’ success in this regard.

B. Signature Generation

These students are also likely to be interested in some of the important applications of cryptological ideas besides creating secret messages. Among these are

- Password quality and entropy
- Secret sharing
- Digital signatures

Figure 5 shows the screen that demonstrates signature generation step by step.



SHA-1 hash of <startingexample-en.txt>

```
A8 F6 8D EB FB C5 41 07 83 C8 97 6F B0 54 9B 08 B7 DE B0 14
```

Figure 5: Step by Step Signature generation

The signature visualization allows the student to see the steps involved in signature generation, step by step. A student may select an input file to be signed.

Next the student selects the type of hash function to be used. Many hashes are available, from MD2 to MD5, to SHA, etc. Once the function is chosen, information is shown about the hash function, such as bit length, and the hash value is then computed.

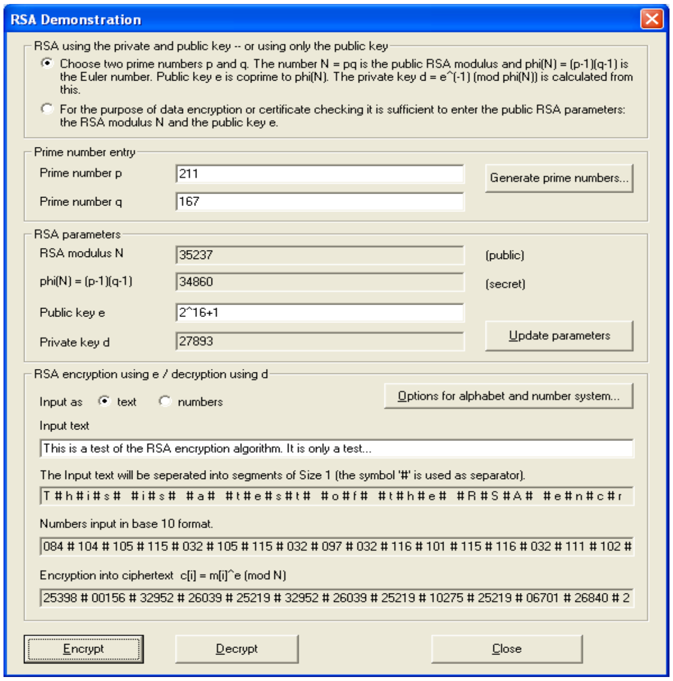
Once computed, the hash value may be shown. Next, key generation is done. The student is given the option of using their own two primes to generate the RSA key, or it will generate very large primes for you. Once the key is stored, the hash is encrypted, and shown to the student.

After the hash is encrypted, the certificate must be provided. The certificate is generated based on the credentials you provide, and via random number generation. The random numbers are generated in a novel way: based on the user’s mouse and key strokes over a certain period (once enough data is gathered, it stops storing them). The certificate is displayed once it is generated.

Finally, the signature is generated based on all the above steps, and is displayed for the student to see. By following along as these steps are executed, the student gets a very good demonstration of the process, and the visual demonstration helps to cement the knowledge from classroom instruction.

C. RSA Demonstration

CrypTool also provides step by step demonstrations for many of the more advanced algorithms, such as RSA, via dialogs like the one shown in Figure 6:



RSA Demonstration

RSA using the private and public key -- or using only the public key

● Choose two prime numbers p and q. The number N = pq is the public RSA modulus and phi(N) = (p-1)(q-1) is the Euler number. Public key e is coprime to phi(N). The private key d = e^(-1) (mod phi(N)) is calculated from this.

○ For the purpose of data encryption or certificate checking it is sufficient to enter the public RSA parameters: the RSA modulus N and the public key e.

Prime number entry

Prime number p: 211

Prime number q: 167

RSA parameters

RSA modulus N: 35237 (public)

phi(N) = (p-1)(q-1): 34860 (secret)

Public key e: 2^16+1

Private key d: 27893

RSA encryption using e / decryption using d

Input as ● text ○ numbers

Input text

This is a test of the RSA encryption algorithm. It is only a test...

The Input text will be seperated into segments of Size 1 (the symbol '#' is used as separator).

```
T # h # i # s #  # i # s #  # a #  # t # e # s # t #  # o # f #  # t # h # e #  # R # S # A #  # e # n # c # r
```

Numbers input in base 10 format.

```
084 # 104 # 105 # 115 # 032 # 105 # 115 # 032 # 097 # 032 # 116 # 101 # 115 # 116 # 032 # 111 # 102 #
```

Encryption into ciphertext c[i] = m[i]^e (mod N)

```
25398 # 00156 # 32952 # 26039 # 25219 # 32952 # 26039 # 25219 # 10275 # 25219 # 06701 # 26840 # 2
```

Figure 6: RSA dialogs

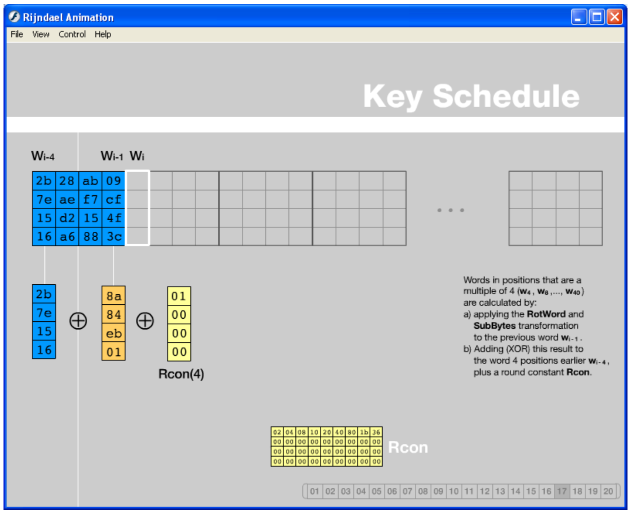
CrypTool's RSA demonstration shows in a stepwise fashion how RSA is used, either with only a public key, or with both the public and private key.

Based on the provided primes, the public and secret keys are generated, based on the RSA modulus and the phi(n) functions.

Additionally, encryption and decryption of text is allowed in the demonstration, allowing the student to see the result of the operation, and the encrypted text, or to decrypt text that they have encrypted already. Through the explanation of the algorithm, the demonstration of what the ciphered plaintext is, and what the deciphered plaintext are of encrypted text, the student can gain a full understanding of the algorithm.

D. Rijndael Algorithm

Figure 7 is a glimpse of the Rijndael algorithm (winner of the AES competition to replace the DES standard) in action:



2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

W_{i-4}			Rcon(4)
2b	⊕	8a	01
7e		84	00
15		eb	00
16		01	00

02	04	08	10	20	40	80	1b	36
00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00

Words in positions that are a multiple of 4 (w_4, w_8,..., w_{40}) are calculated by:
a) applying the **RotWord** and **SubBytes** transformation to the previous word w_{i-1}.
b) Adding (XOR) this result to the word 4 positions earlier w_{i-4}, plus a round constant **Rcon**.

Figure 7: Rijndael algorithm

The Rijndael encryption algorithm is an extremely complex one. It involves ten iterations of processing for each block of plaintext to be encrypted, and each iteration contains four major steps. In a situation like this, the detailed step-by-step visualization provided by CrypTool is invaluable for understanding the algorithm.

E. Encryption Strength

Additionally, CrypTool allows students to test the strength of different encryption methods. For many years, one of the authors has used a batch system, John the Ripper, to test the strength of passwords and passphrases using different algorithms from a light-weight examination to a simple dictionary search to a full brute force method. CrypTool allows the students to 'see' how the tests are done. A simple assignment is to have students compare the batch aspects of John the Ripper to CrypTool and write a report about the strengths and weaknesses of each in the educational setting and in a production setting.

As an advanced forensics assignment using CrypTool, students can be given a data file encrypted by an widely available encryption system such as the Unix-like encrypt/decrypt command, encrypted cookies from an online merchandiser, a password file from a Unix-like system, or one of the sign-on password storage files for Firefox and asked to decrypt it. The object of the exercise is to produce a report outlining the techniques used, the final result of the decryption, and any interesting data points found during the analysis. The students are informed that the analysis may or may not yield a successful decryption but that the forensics activities are best demonstrated using an animation system like CrypTool.

VI. Graduate Level Applications

Students at the graduate level should be fully capable of appreciating and understanding even the most modern of algorithms. Thus they can benefit from any of the features mentioned above for use at the undergraduate level.

One of the advantages of CrypTool at this level is that it is open source, so that students can study, modify, expand, and improve upon the existing package. The code should be accessible to most students. It is currently written in C++, but beta versions are already available in C# and Java. CrypTool actively invites programmers and students to join in the further development of the project.

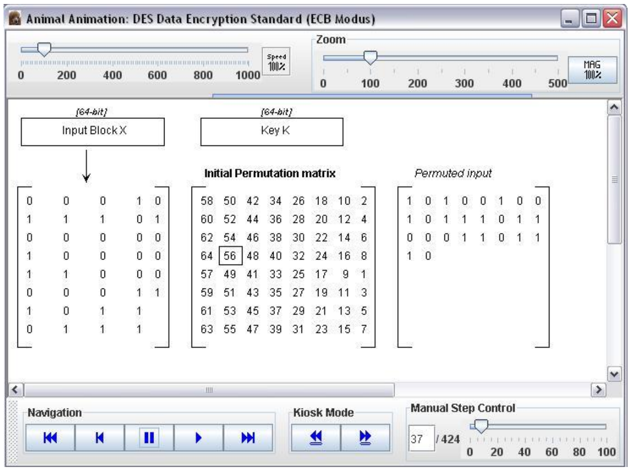
VII. Student Feedback

The following remarks are based on comments made by students in a class taught by one of the authors.

Students found CrypTool invaluable in illustrating and demonstrating complicated encryption techniques. This visual demonstration method enhances the students' understanding and lets them see the algorithm in action, filling any gaps in understanding the steps involved in the technique.

In addition to visualization, the students found practicing the techniques enhanced with the software, in that they could verify answers they had generated by hand in practice questions, via CrypTool. This reinforces the practice, since the students could review their own work, find trouble spots, and enhance their ability to perform the encryption, and their understanding of the techniques.

A specific example of students enhancing their understanding is via the DES visualization included with the software.



Animal Animation: DES Data Encryption Standard (ECB Modus)

Speed 100%

Zoom MAG 100%

[64-bit] Input Block X

[64-bit] Key K

```
                          Initial Permutation matrix      Permuted input
0   0   0   1  0          58 50 42 34 26 18 10  2         1 0 1 0 0 1 0 0
1   1   1   0  1          60 52 44 36 28 20 12  4         1 0 1 1 1 0 1 1
0   0   0   0  0          62 54 46 38 30 22 14  6         0 0 0 1 1 0 1 1
1   0   0   0  0          64 56 48 40 32 24 16  8         1 0
1   1   0   0  0          57 49 41 33 25 17  9  1
0   0   0   1  1          59 51 43 35 27 19 11  3
1   0   1   1             61 53 45 37 29 21 13  5
0   1   1   1             63 55 47 39 31 23 15  7
```

Manual Step Control 37 / 424

Figure 9: DES algorithm

The very lengthy and complicated steps are shown in a step-by-step fashion. Controls for stepping forwards and backwards are provided, allowing reviewing earlier steps. This very explicit demonstration allows a better understanding of the complicated steps involved.

VIII. Conclusion

CrypTool is a powerful and comprehensive free software package with very approachable interface. Along with its accompanying extensive support materials and programs, it provides substantial assistance to individuals interested in security, student wanting to explore the field, and as a teaching tool for instructors.

The variety of different algorithms implemented in CrypTool, from the earliest known to the most modern, both sets the background for current cryptography, and also enables an understanding of the most modern cryptographic principles and algorithms.

While tools like CrypTool do take class time to teach, the learning curve is not steep and the time that would otherwise be used to discuss coding the different cryptography and cryptanalysis tools can be used to cover additional cryptography concepts.

IX. References

[1]http://www.govinfosecurity.com/articles.php?art_id=2166

[2]http://www.npr.org/templates/story/story.php?storyId=128574055

[3] Computer Science 2008, An Interim Revision of CS 2001(http://www.acm.org//education/curricula/ComputerScience2008.pdf)

[4] Cryptology with CrypTool (http://www.cryptool.org/download/CrypToolPresentation-en.pdf)

[5] www.CrypTool.org

[6] www.CrypTool-online.org

[7] Richard Spillman "A software tool for teaching classical & contemporary cryptology", Journal of Computing Sciences in Colleges, Volume 20 Issue 2, 2004.

[8] Jesús Adolfo García-Pasquel, José Galaviz Ganzúa: A cryptanalysis tool for monoalphabetic and polyalphabetic ciphers, Journal on Educational Resources in Computing (JERIC), ACM, Volume 6 Issue 3, 2006.

[9] Mohamed S. Asseisah, Hatem M. Bahig, Sameh S. Daoud, “Interactive visualization system for DES”, AMT'10: Proceedings of the 6th international conference on Active media technology”, Springer-Verlag, 2010.

[10] David Bishop Introduction to Cryptography with Java Applets, 1st edition, Jones & Bartlett Learning, 2002.

[11] Schneier Bruce, Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd edition, Wiley, 1996.

[12] Stinson, Douglas, Cryptography: Theory and Practice, 2nd edition, Chapman and Hall, 2002.

[13] Stallings, William, Cryptography and Network Security, 4th edition, Prentice-Hall, 2005.

[14] Pfleeger and Pfleeger, Security in Computing, 4th edition, Prentice-Hall, 2007.