Contents

Fourth International Derive TI-89/92 Conference
Liverpool John Moores University, July 12 – 15, 2000

Cryptology with DERIVE in the classroom

Dirk Warthmann
St. Ursula-Gymnasium, Düsseldorf, Germany
e-mail: d.warthmann@gmx.de

The encryption and decryption of messages is a subject of great interest for students. (Every teacher knows the problem that pupils send each other little messages during the lessons). Using CAS (for example Derive) the process of encryption and decryption can be simulated by every student even if an asymmetrical process like RSA is used.

The subject is of great importance when using the internet and whenever the transfer of sensitive data (credit card numbers, pin code) is necessary. Programmes like PGP are using RSA for the encryption of keys.

On the one hand, it is possible to only present the calculation process to the students in order to simulate encryption and decryption with RSA. Using "Derive" the students can test how the encryption and decryption process works. On the other hand, this subject and the use of Derive offer the opportunity to teach parts of number theory like congruences, calculation in finite fields and Fermat's Little Theorem, if the students see a practical use and can test their results with a programme like Derive.

I tested the following sequence of lessons with students at the age of 14 or 15. These students had chosen a combination of mathematics and computer sciences in addition to the normal subject of mathematics. Every student had the opportunity to use a PC and "Derive" during the lessons of this course.

The sequence had the following contents:

1. Caesar-Code as an example of a very simple cryptosystem
2. Further examples of symmetrical cryptosystems.
3. RSA
4. Installation and use of PGP

During the lessons and exercises the following subjects of number theory were treated:

- Calculation with congruences,
  especially the problem to find an inverse number in finite fields (modulo a prime p or modulo p*q, p, q prime numbers)
- The use of Euclidean Algorithm
- Factorisation of large numbers
- Fermat's Little Theorem and Euler's Corollary
- Principles of RSA:
  Choosing the encryption exponent (public key) and calculating the decryption exponent (secret key). Due to having only a limited amount of time it was not possible to treat the special cases

and traps, which J. Wiesenbauer presented in his workshop "Factoring and RSA-Codes using Derive".

In the following text I will concentrate on topics 1 and 3.

Caesar's cryptosystem

Caesar's cryptosystem is quite suitable to introduce students to cryptology. Normally some students know or use the possibility to encrypt little messages by moving the letters for a fixed distance, probably by one letter.

As we know from history Caesar encrypted his messages as follows:

A → D   B → E   C → F   ...   X → A   Y → B   Z → C

So the main idea is easy to understand and is realised by a linear substitution of letters. Therefore, the main topics like the functions of encryption and decryption can be explained by using this example.

In order to use Derive we have to transform the letters into numbers:

A	B	C	D	E	F	G	H	......	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	......	19	20	21	22	23	24	25

The message "THIS IS SECRET"
is represented by [19, 07, 08, 18, 08, 18, 18, 04, 02, 17, 04, 19]. By using Derive the whole message can be treated with a vector pt (plaintext).

pt := [19, 7, 8, 18, 8, 18, 18, 4, 2, 17, 4, 19]

The students may find the correct functions for encryption and decryption as follows:

$f(x) = x + 3 \bmod 26$ and $f^{-1}(x) = x - 3 \bmod 26$.

In the correct syntax of Derive the students have to define the function in the following way: F(x) := MOD(x + 3, 26) Finvers(x) := MOD(x - 3, 26)

By using the possibilities of manipulation vectors the students get the encrypted vector as follows:

ct := VECTOR(F(ELEMENT(pt, n)), n, 1, 12) and the result: ct = [22, 10, 11, 21, 11, 21, 21, 7, 5, 20, 7, 22]

With this knowledge the decryption process is no problem:

pt := VECTOR(FINVERS(ELEMENT(ct, n)), n, 1, 12)

and the result: pt := [19, 7, 8, 18, 8, 18, 18, 4, 2, 17, 4, 19].
This is the plaintext in numerical form.

Alphanumerical output with Derive

Having done some exercises students normally ask, if with the help of Derive there is any possibility to get an alphanumerical output, meaning an output with ascii-characters. Normally you

Contents

can get such an output only with systems, that allow the transformation of numbers into characters and vice versa. These conditions are not realised in this Computer Algebra System. But there is a trick to achieve this output:

Students know different bases such as decimal base, hexadecimal base and others. Derive even works with a base of 36. When we switch the output base from decimals to 36 we have the following digits:

0	1	2	3	4	5	6	7	8	9	A	B	...	O	P	Q	R	S	T	U	V	W	X	Y	Z

Students should define a function, transforming the numbers in the vectors pt and ct to the interval between 10 and 36. Then they have to change the output base to 36. At last the output of the vectors ct (alphact) and pt (alphapt) should lead to the desired result:

```
T(x) := x + 10
OutputBase := 10
alphact := VECTOR(T(ELEMENT(ct, n)), n, 1, 0C)
alphact = [0W, 0K, 0L, 0V, 0L, 0V, 0V, 0H, 0F, 0U, 0H, 0W]
alphapt := VECTOR(T(ELEMENT(pt, n)), n, 1, 0C)
alphapt = [0T, 0H, 0I, 0S, 0I, 0S, 0S, 0E, 0C, 0R, 0E, 0T]
```

The reader might be disturbed by the leading zero in front of the characters. Of course this is not a perfect solution. Nevertheless, students are pleased to have a readable translation of the numerical code. Unfortunately, all numbers of the output have a base of 36. More complicated is a solution having an input and an output base of 36. This should be an exercise for interested students. In any case the user should return to the decimal base directly after the output.

An attack on this Caesars cryptosystem was only a difficult matter for common Roman soldiers. Nevertheless, this system was used in the Russian army in 1915, because other systems were too complicated (Wobst 96, 30). In our days the Russian cryptology is up to date. Students notice quickly that there are only 25 possibilities to get a different encryption. With examples of encrypted messages a successful attack can be demonstrated. With the Derive-orders mentioned above all possibilities can be tried, to find out the correct decryption.

The Caesar cryptosystem is special case of the substitution of letters. Of course, there are a lot of interesting cryptosystems, belonging to the class of the substitution of letters. The principles of the Enigma, the German encryption machine during the second world, would be worth being presented, because it is possible to enlarge the view upon the subjects of mathematics and history. In the last few years some interesting books and essays concerning the history of the decryption of the enigma-code and its consequences have been published. I have experienced that students rather like to learn something about cryptology in our time, the encryption of e-mails or pin-code numbers.

First of all, the difference between symmetrical and asymmetrical cryptosystems should be taught to the students.

The cryptosystems, mentioned up to now, belong to the symmetrical cryptosystems. The function of the encryption or the encryption key is similar to the function of the decryption (decryption key). With the knowledge of the encryption key it is possible to conclude upon the decryption key. Therefore both keys have to be secret. If a group of people wants to communicate with encrypted messages, everyone has to depose two keys with every partner. One participant needs $n-1$ pairs of key; for the whole group there are $\frac{n(n-1)}{2}$ pairs of key necessary. For a large group, as it is normally the case in the web, this is not acceptable.

Using asymmetrical cryptosystems it is not possible to conclude from the encryption key (public key) to the decryption key (secret key). If in a communication circle an asymmetrical cryptosystem is used, one participant has to give his public key to all interested partners. He always uses the same key, his secret key to decrypt all the messages from different partners. No one from outside is able to decrypt a message without the knowledge of the secret key, even if he knows the public key.

In the following paragraphs I want to show, how students can convince themselves about the functioning of such a cryptosystems with the help of Derive.

RSA cryptosystem

There are a lot of publications about the development and use of RSA-cryptosystems (for example Koblitz, 94). I only want to treat the main theorems and corollaries, which are necessary to understand the RSA-cryptosystem. The teacher must decide, whether to make the students find important theorems or to present them beforehand. The students must know about divisibility and congruences:

Definition:

$a|b \Leftrightarrow \exists r \in N : b = r \cdot a$

1) $a|b \wedge c \in IN \Rightarrow a|b \cdot c$

2) $a|b \wedge b|c \Rightarrow a|c$

3) $a|b \wedge a|c \Rightarrow a|b \pm c$

4) $p \;\; prime \;\; \wedge p|a \cdot b \Rightarrow p|a \vee p|b$

Characteristics:

$a \equiv b \mod m \Leftrightarrow m|a-b$

We define:

1) $a \equiv a \mod m$

2) $a \equiv b \mod m \Leftrightarrow b \equiv a \mod m$

4) $a \equiv b \mod m \wedge c \equiv d \mod m \Rightarrow$

$a \pm c \equiv b \pm d \mod m \wedge a \cdot c \equiv b \cdot d \mod m$

3) $a \equiv b \mod m \wedge b \equiv c \mod m$

$\Rightarrow a \equiv c \mod m$

Characteristics:

The basis of RSA-cryptosystem is Fermat's Little Theorem:

$a^p \equiv a \mod p$

- a is representative of a numerical plaintext (for example a letter in ASCII); p is prime. We can take a^p mod p as a summarised encryption and decryption process, because we get the plaintext in numerical form after calculating a to the power of p in the finite field modulo p.

- We get a cryptosystem if we succeed in factorising the exponent p which would be impossible with p being prime.

Therefore a field modulo n is necessary with n being the product of two large prime numbers:

$$n = p \cdot q$$

We get an extension of Fermat's Little Theorem with the following corollary:

decryption exponent d

$$\left(a^e\right)^d \equiv a \mod p \cdot q$$

encryption exponent e

- e and n are public (public key)
- d, p and q are secret (private key)

Now students may test the functioning of the system with the following examples by using Derive:

n := 46927 d := 26767 e := 39423

Now we choose a plaintext, only one letter, represented by the digit 7. We calculate:

$ct := \text{mod}\left(7^e, n\right)$ ct = 42493 (ciphertext in numerical form)

decryption process: $pt := \text{mod}\left(42493^d, n\right)$ pt = 7 (plaintext)

On April 10, 1996 Prof. Arjen K. Lenstra found that n = 1807082088687404805951656164405905566278102516769401349170127021450056662540244048387341127590812303371781887966563182013214880557 has the following factorization: p := 39685999459597454290161126162883786067576449112810064832555157243

and q := 45534498646735972188403686897274408864356301263205069600999044599 (Beutelspacher 1997).

With Derive it is possible to use n for RSA encryption for example with the following encryption and decryption exponents: e = 455344986467359721884036868972744088643563012632050696009990446121

and d := 8642897079425470367314389825428392176199928285178813043330577831635931491171366127140704723161798585604013227868659119607891245531

Students can test the encryption and decryption process with the commands mentioned above.

Now some questions occur:

1. Which prime numbers p and q are suitable to get n?
2. In which way should the encryption exponent e be selected?

---

[1] e is a prime number, chosen at random; d is calculated with Inverse_mod (File: Number.mth).

3. Is there an algorithm to calculate the exponent d, when e, p and q are known?

At this point some simplifications are necessary due to didactic considerations. The students should know that p and q have to be prime numbers with a certain range between each other. The product of p and q should have about 150 digits in order to have a high level of security. For risks and traps in connection with the selection of p and q cf. the documentation of J. Wiesenbauer's workshop in Gettysburg. For the selection of "e" a randomly chosen prime number will be required. The most interesting question is according to my experience the calculation of the decryption key "d".

In the following sequence the corollaries and theorems, which are necessary for the calculation process are mentioned:

1) $\varphi\ (p) = p-1$

2) $\varphi\ (p^{\alpha}) = p^{\alpha} - p^{\alpha-1}$

3) $\gcd(m,n) = 1 \Rightarrow \varphi\ (m \cdot n) = \varphi\ (m) \cdot \varphi\ (n)$

The Euler phi-function $\varphi(n)$ is defined to be the number of non-negative integers b less than n:

$$\varphi(n) = \left\{0 \le b < n \middle| \gcd(b,n) = 1\right\}\ b\varepsilon\ \ IN$$

characteristics:

1) $a \ne 0 \bmod\ p \Rightarrow a^{p-1} \equiv 1 \bmod\ p\ \ (Fermat)$

2) $\gcd(a,n) = 1 \Rightarrow a^{\varphi\ (n)} \equiv 1 \bmod n\ \ (Euler)$

3) $\gcd(a,n) = 1 \wedge m \equiv m_a \bmod \varphi\ \ (n)\ \ (Corollar)$

$\Rightarrow a^{m} \equiv a^{m_a} \bmod\ \ n$

4) $special\ \ case: \ m = e \cdot d \wedge m_a = 1 \wedge$

$e \cdot d \equiv 1 \bmod\ \ \varphi\ \ (n) \Rightarrow a^{e \cdot d} \equiv a^{1} \bmod\ \ n$

Basic Propositions and Corollaries concerning RSA:

(Koblitz 94, 22)

First step: $\varphi(n) = \varphi(p \cdot q) = \varphi(p) \cdot \varphi(q) = (p-1) \cdot (q-1)$

Using the special case 4) we can calculate d, if e, p and q are known.

It is sufficient to find d with the quality of being the inverse number of $e\ \ \bmod \varphi(n)$: $e \cdot d \equiv 1 \bmod \varphi(n)$

Then we have the RSA system $\left(a^{e}\right)^{d} \equiv 1 \bmod\ \ n$

So the problem is reduced to finding the inverse number of $e\ \ \bmod \varphi(n)$

To solve this problem we can use the capabilities of Derive. There exists a file named "number.mth" with a lot of useful functions of number theory. The function Inverse_Mod calculates the inverse

Contents

element in finite fields. The algorithm of this function is developed by J. Wiesenbauer and works very quickly even with large numbers about 100 digits. Here is one example:

$$p = 167 \quad q = 283 \quad n = p \cdot q = 47261$$
$$\varphi \quad (n) = (p-1)(q-1)$$
$$\varphi \quad (n) = 166 \cdot 282 = 46812$$
$$e = 293$$
$$d := Inverse\_Mod(293, 46812) \quad d = 37865$$

At last here is the calculation to get "d" without using a Derive function:

$$e \cdot d \equiv 1 \bmod (p-1) \cdot (q-1)$$
$$\Rightarrow (a^e)^d \equiv a \bmod p \cdot q$$
$$293 \cdot d \equiv 1 \bmod 46812$$
$$\Rightarrow (a^{293})^d \equiv a \bmod 47261$$
$$293 \cdot d - 1 = r \cdot 46812$$
$$293 \cdot d + \left(-r\right) \cdot 46812 = 1$$

Now we get "d" with the help of Euclidean Algorithm:

46812=159*293+225	225=46812-159*293
293=1*225+68	68=293-1*225
225=3*68+21	21=225-3*68
68=3*21+5	5=68-3*21
21=4*5+1	1=21-4*5

$$8947 \cdot 293 \equiv -1 \quad \bmod 46812$$
$$-8947 \cdot 293 \equiv 1 \bmod 46812$$
$$37865 \cdot 293 \equiv 1 \bmod 46812$$

1=21-4*(68-3*21)=13*21-4*68
1=13(225-3*68)-4*68=13*225-43*68
1=13*225-43(293-1*225)=56*225-43*293
1=56(46812-159*293)-43*293=56*46812-8947*293
8947*293 + 1 = 56*46812
8947*293 - (-1) = 56*46812

This calculation is only possible, if the factorisation of n in p and q is known. The function Inverse_Mod can only be used if p and q are known. The security of RSA-Cryptosystem is dependent on the difficulty to factorise large numbers.

The students can try to get the factorisation of $n = p \cdot q$ only with little numbers. Using large numbers paralyses the computer.

In exercises the students may use the encryption and decryption process for every letter in numerical form. The output is numerical and even alphanumerical, if the trick with the transformation of bases, mentioned above, is used. Normally the sender tries to transform a group of letters to a number. An example is given with the syntax of Derive:

As a demonstration the encryption of the word ”INFORMATIK” is shown. Therefore I separate the word into two blocks with five letters ”INFOR” and ”MATIK”. Both strings are transformed into an alphabet with 27 characters (A = 0, B = 1, .. , Z = 26, ‘ ‘ = 27). Now we have: I = 8, N = 13, F = 5, O = 14, R = 17 und M = 12, A = 0, T = 19, I = 8, K = 10.
These codes are summarised as follows:
$a := 8\cdot 27^4 + 13\cdot 27^3 + 5\cdot 27^2 + 14\cdot 27 + 17$
$b := 12\cdot 27^4 + 0\cdot 27^3 + 19\cdot 27^2 + 8\cdot 27 + 10$
a = 4511447
b = 6391369
Now I’m looking for n = p*q with the following quality:
$27^5 < n < 27^6$
$27^5 = 14348907$
$27^6 = 387420489$
The interval, from which n is chosen, is fixed by the number of letters in the alphabet, we are working with (in our case 27) and the length of the block (in our case : 5).
p := NEXT_PRIME(1234) (chosen at random with the quality that $27^5 < n < 27^6$)
q := NEXT_PRIME(98765)
n := p·q
p = 1237
q = 98773
n = 122182201
phi := (p - 1)·(q - 1)
phi = 122082192
Choose the encryption exponent e:
e := NEXT_PRIME(654321)
e = 654323
Calculate the inverse exponent of e:
d := INVERSE_MOD(e, phi) (inverse_mod gives back the inverse of e: e d = 1 mod phi)
d = 61264523
Test:
e·d = 40086786482929
MOD(40086786482929, phi) = 1
Now I encrypt a:
MOD(a^e , n) = 36971144
Deciphering of a:
MOD(36971144^d , n) = 4511447
Encryption of b:
MOD(b^e , n) = 103809371
Deciphering of b:
MOD(103809371^d , n) = 6391369
By using large messages a length of blocks will be chosen so that larger prime numbers are possible for encryption process.

Contents

Summary

The topic of cryptology offers the opportunity, to treat aspects of number theory in the classroom, because there is a connection to real life.

- In spite of the subject matter being quite demanding all the students can take part in practical activities.
- The teacher can decide whether to make the students find important theorems or to present them beforehand.
- The subject matter cannot only be taught with the help of carefully chosen proofs, the level can also be lowered.
- Throughout the lessons the students can always see a connection to real life.
- The limits of CAS regarding factorisation become visible.

References

Beutelspacher, Albrecht: Geheimsprachen : Geschichte und Techniken. München: Beck, 1997.

Koblitz, Neal: A course in number theory and cryptography. New York: Springer, 1994.

Warthmann, Dirk: Informatik. In: Norbert Münnix/ Dirk Warthmann (Hrsg.): Fächer und Fächerübergreifender Unterricht des Gymnasiums in der Sekundarstufe I, Band 1: Naturwissenschaften. Heinsberg: Dieck, 1999.

Wiesenbauer, Johann: Factoring and RSA Codes using DERIVE. In: Presentations from the 1998 3rd International DERIVE and TI 92 Conference. Gettysburg, 1998

Wobst, Reinhard: Abenteuer Kryptologie: Methoden, Risiken und Nutzen der Datenverschlüsselung. Bonn [u.a.] : Addison-Wesley, 1998