INNOVATIVE TECHNIQUES FOR DATA SECURITY: APPLIED CRYPTOGRAPHY IN INFORMATION SECURITY

Simona Bibic, Carmina Georgescu, Emil Simion, Antonela Toma

UNIVERSITY POLITEHNICA of BUCHAREST
FACULTY OF APPLIED SCIENCES
Department of Methods and Mathematical Models
Center for Research and Training in Innovative Techniques of Applied Mathematics in Engineering (CITI)

Emails: simona.bibic@upb.ro; emil.simion@upb.ro; carmina.georgescu@upb.ro; antonela.toma@upb.ro

Contents

MODULE 1. APPLIED CRYPTOGRAPHY IN INFORMATION SECURITY 4
INTRODUCTION 4
INFOSEC 4
INFOSEC Standards 6
CRYPTOGRAPHIC STANDARDS 7
FIPS 140-2 7
Cryptographic Module Validation Program (CMVP) 9
IT&C ASSURANCE STANDARDS (Common CRITERIA) 10
Target of Evaluation 11
Evaluation process 12
LESSON LEARNED 13
REFERENCES 13
MODULE 2. CASE STUDY VULNERABILITIES IN RSA ENCRYPTION ALGORITHM 14
INTRODUCTION 14
RSA ALGORITHM 14
ATTACKS ON RSA ALGORITHM 15
Chosen ciphertext attack Some attacks work against the implementation of RSA 15
Common Modulus Attack on RSA 15
Low encryption exponent attack against RSA 15
Low decryption exponent attack against RSA 16
Attack on encryption and signing with RSA 16
Attack in case of small difference between prime numbers p and q 16
Hardware attack 16
INCLUDING TRAP INFORMATION INTO RSA EXPONENTS 17
REFERENCES 17
MODULE 3. SOME EXAMPLES OF ADVANCED CRYPTOGRAPHIC ALGORITHMS & TECHNIQUES 19
CRYPTOGRAPHIC PRINCIPLES 19

PUBLIC CONTESTS 22
SYNTHESIS PROBLEMS 30
Questions 30
Answers 39

MODULE 1. APPLIED CRYPTOGRAPHY IN INFORMATION SECURITY

INTRODUCTION

This chapter focuses on the link between information security and cryptography. The link is represented by National Institute of Standards and Technology (NIST) cryptographic standards, Federeal Information Processing Standard FIPS 140-2 (Security requirements for cryptographic modules) standard and Common Criteria for Information Technologies Security Evaluation (ISO 15408) standard.

Information security is the science of protecting information and information systems from unauthorized access, use, disclosure, disruption, modification or destruction. *Cryptography* deals with design, implemention and evaluating cryptographic algorithms (e.g. NIST AES selection process, SHA-3 competion etc.) in order to be used by products (software and/or hardware) which are intended to protect information or information systems. Before using in information systems those cryptographic products need to be tested and evaluated also. One evaluation standard is FIPS 140-2. After this evaluation is obtained, from an accredited Laboratory, the system itself needs to be evaluated in order to have a image of the assurance level obtainded. Usually these evaluation is made using ISO 15408 (Common Criteria for Information Technology systems) standard.

INFOSEC

INFOSEC domain covers the following areas:

Physical security describes both measures that prevent or deter attackers from accessing a facility, resource, or information stored on a physical media and guidance on how to design structures to resist various hostile acts.

Personel security describes the restriction of data which is considered very sensitive. Under need-to-know restrictions, even if one has all the necessary official approvals (such as a security clearance) to access certain information, one would not be given access to

such information unless one has a specific *need to know*; that is, access to the information must be necessary for the conduct of one's official duties. As with most security mechanisms, the aim is to make it difficult for unauthorized access to occur, without inconveniencing legitimate access. Need-to-know also aims to discourage "browsing" of sensitive material by limiting access to the smallest possible number of people.

Procedural security deals with the establishment and enforcement of security procedures. Some of these procedures may be independent of the type or types of computers involved. Others may not. For example, perimeter security controls are usually similar for all type of systems. But desktop computers may require forms of antitheft protection not required by mainframes. Procedural security regulates the performance of duties associated with system operation and use, and with the physical storage of system information. Common security practices include partitioning computer operating duties, using several operators, and storing backup tapes at bonded, offsite depositories. Procedural security also encompasses and may regulate company policies that deal with information security, such as policies that regulate the way individuals manage their own passwords.

Communications security (**COMSEC**) describes the measures and controls taken to deny unauthorized persons information derived from telecommunications and ensure the authenticity of such telecommunications. Communications security includes crypto security, transmission security, emission security, traffic-flow security and physical security of COMSEC equipment.

Computer security is a branch of technology known as information security applied to computers. The objective of computer security includes protection of information and property from theft, corruption, or natural disaster, while allowing the information and property to remain accessible and productive to its intended users.

TEMPEST is a codename referring to investigations and studies of compromising emanations (CE). Compromising emanations are defined as unintentional intelligence-bearing signals which, if intercepted and analyzed, may disclose the information transmitted, received, handled, or otherwise processed by any information-processing equipment. Compromising emanations consist of electrical, mechanical, or acoustical energy intentionally or by mishap emitted by any number of sources within equipment/systems which process national security information. This energy may relate to the original encrypted

message, or information being processed, in such a way that it can lead to recovery of the plaintext. Laboratory and field tests have established that such CE can be propagated through space and along nearby conductors. The interception/propagation ranges and analysis of such emanations are affected by a variety of factors, e.g., the functional design of the information processing equipment, system/equipment installation, and, environmental conditions related to physical security and ambient noise. The term "compromising emanations" rather than "radiation" is used because the compromising signals can, and do, exist in several forms such as magnetic-and/or electric-field radiation, line conduction, or acoustic emissions.

Information assurance (IA) is the practice of managing information-related risks. More specifically, IA practitioners seek to protect and defend information and information systems by ensuring confidentiality, integrity, authentication, availability, and non-repudiation. These goals are relevant whether the information are in storage, processing, or transit, and whether threatened by malice or accident. In other words, IA is the process of ensuring that authorized users have access to authorized information at the authorized time.

INFOSEC Standards

INFOSEC standards can be stratified like in Figure 1: standards for cryptographic algorithms, cryptographic modules and for IT&C security. In this chapter we focus on standards for cryptograhic algorithms, cryptomodules (FIPS 140-2) and IT&C standards (e.g. ISO 15408).

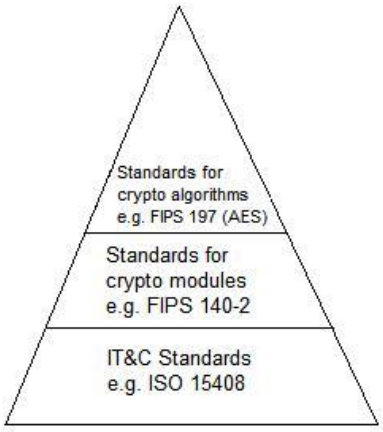


Figure 1 INFOSEC standards stratification

CRYPTOGRAPHIC STANDARDS

Our discussion is based on National Institute of Standards and Technologies (NIST) cryptographic standards. These standards can be divided in four classes: symmetric key, public key, secure hash and random number generation.

In symmetric key we can found for example AES (FIPS 197), DES (FIPS 46-3) for block ciphers standards or HMAC (FIPS 198) for hashing and message authentication code. We remained that simple DES was replaced by AES, 3-DES being in use.

In public key standards we can found Digital Signature Standard (FIPS 186-3), Key Establishing Schemes (DH&MQV, FFC&ECC SP 800-56A) and Key Management Guideline.

Secure hash is referring to SHA-1, SHA-224, SHA-384, SHA-512 (FIPS 180-2). At this time there exists a draft for SHA-3 which will replace SHA-2.

One standard for random number generation standards is SP 800-90.

The following table gives the theoretical comparable strengths of symmetric and asymmetric cryptographic algorithms.

Sym Key	80	112	128	192	256
Hash functions (for signatures)	160	224	256	384	512
FFC and IFC	1K	2K	3K	7.5K	15K
ECC	160	224	256	384	512

NIST approved standards are referred by NIST Cryptographic Toolkit.

Some of these standards are allowed to process classified information. For example, AES with 128 bit key can be used to protect SECRET classified information and AES with 192 or 256 bit key can be used to protect TOP SECRET classified information.

FIPS 140-2

Cryptographic controls are provided using cryptographic modules, which may include capabilities such as signature generation and verification, encryption and decryption, key generation, and key establishment.

An undetected error in a cryptographic module design could affect every user in the system for which it is supposed to provide protection. For example, the verification of a chain of public key certificates might not function correctly.

Verifying a chain of public key certificates helps a signature verifier determine if a signature was generated with a particular key. If the function is implemented incorrectly in a cryptographic module, the potential for the dissemination of weak cryptography could be introduced into the system, possibly allowing for signature forgery or the verification of invalid signatures. Therefore, it is important to have cryptographic modules tested before distributing them throughout a system.

The security requirements in FIPS 140-2 cover 11 areas related to the design and implementation of a cryptographic module:

- Cryptographic module specification includes definition of cryptographic boundary, approved algorithms and approved modes of operations;
- Cryptographic module ports and interfaces are referred to the specification of all interfaces and all input data paths. For security level 3 and 4 data ports for unprotected critical security parameters logically or physically separated from others data ports;
- Roles, services and authentication requires, for all security levels, logical separation of required and optional roles and services. For level 2 operators authentication must be role–based or identity–based. To achieve security level 3 and 4 operator authentication must be identity-based;
- Finite state model requires the specification of finite state model, required and optional states, state transition and specification of these transitions;
- Physical security is focusing to tamper evidence, detection and response (e.g. erasing critical security parameters);
- Operational environment is referring to evaluation, for example, of Protection Profile (PP) at (Evaluation Assurance Level) EAL 4;
- Cryptographic Key Management is referring to the key (secret, private and public) manipulation during its life time: generation, pre -activation, activation, usage, storage and deletion;
- EMI/EMC – electromagnetic compliance with Federal standards;
- Self – Tests includes power-up tests and conditional tests;
- Design assurance is referring to configuration management, secure installation, design policy and guidance documents;

- Mitigations of others attacks are referred to specification of mitigation of attacks for which no testable requirements are currently available.

Within most areas, a cryptographic module receives a security level rating of 1 to 4, from lowest to highest, depending on what requirements are met. For other areas that do not provide for different levels of security, a cryptographic module receives a rating that reflects the fulfillment of all of the requirements for that area.

An overall rating is issued for the cryptographic module, that indicates the:

1. Minimum of the independent ratings received in the areas with levels, and
2. Fulfilment of all the requirements in the other areas.

On a vendor's validation certificate, individual ratings are listed as well as the overall rating. It is important for vendors and users of cryptographic modules to realize that the overall rating of a cryptographic module is not necessarily the most important rating. The rating of an individual area may be more important than the overall rating, depending on the environment in which the cryptographic module will be used (this includes understanding what risks the cryptographic module is intended to address). Modules may meet different levels in different security requirement areas; for example, a module may implement identity-based authentication (level 3 or 4) and display tamper evidence (level 2).

At this time the draft for FIPS 140-3 where NIST has updated the standard to reflect changes in technology has a fifth security level. In this draft there is a special section dedicated to software security and specifying requirements to protect against non-invasive attacks. Also the reference to Common Criteria (ISO 15408) and requirements for the use of Common Criteria certified operating systems has been dropped. In this draft NIST improves the requirements for authentication for level 4 at two-factor authentication (at least two of three: something known, something possessed and some physical property). Also a greater importance is given to physical security requirements to defeat non - invasive attacks/side channel attacks (protection to timing attacks (TA), differential power analysis (DFA) etc.)

Cryptographic Module Validation Program (CMVP)

NIST and the Communications Security Esablishment (CSE) of the government of Canada established the CMVP. The goal of the CMVP is to provide Federal agencies with a security metric to use in procuring equipment containing cryptographic modules. The results of the

independent testing by accredited laboratories provide this metric. Cryptographic module validation testing is performed using the Derived Test Requirements (DTRs) for FIPS 140-2. The DTRs list of all the vendor and tester requirements for validating a cryptographic module are the basis of testing done by the Cryptographic Module Testing (CMT) accredited laboratories. Figure 2 illustrates the CMV process.

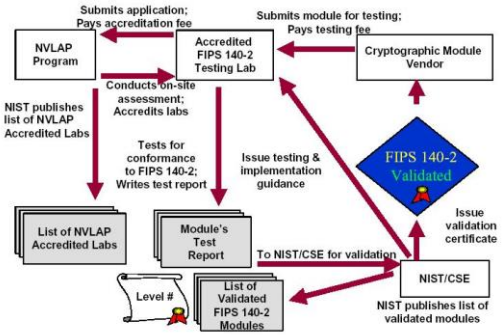


Figure 2 CVMP Process

IT&C ASSURANCE STANDARDS (Common CRITERIA)

Information Technology Security Evaluation Criteria (**ITSEC**), precedesor of **Common Criteria for Information Technology Security Evaluation** (abbreviated as Common Criteria or CC), is a structured set of criteria for evaluating computer security within products and systems. The ITSEC was first published in May 1990 in France, Germany, the Netherlands, and the United Kingdom based on existing work in their respective countries. Following extensive international review, Version 1.2 was subsequently published in June 1991 by the Commission of the European Communities for operational use within evaluation and certification schemes. Since the launch of the ITSEC in 1990, a number of other European countries have agreed to recognise the validity of ITSEC evaluations.

Thus Common Criteria is an international standard (ISO/IEC 15408) for computer security certification. It is currently in version 3.1. Common Criteria is a framework in which computer system users can specify their security requirements, vendors can then implement and/or make claims about the security attributes of their products and testing laboratories can evaluate the products to determine if they actually meet the claims. In other words,

Common Criteria provides assurance that the process of specification, implementation and evaluation of a computer security product has been conducted in a rigorous and standard manner. Common Criteria is performed on computer security products and systems and provides similarly-defined evaluation levels, implements the target of evaluation concept and the Security Target document.

Target of Evaluation

Target of Evaluation (TOE) - the product or system that is the subject of the evaluation.
The evaluation serves to validate claims made about the target. To be of practical use, the evaluation must verify the target's security features. This is done through the following:
Protection Profile (PP) - a document, typically created by a user or user community, which identifies security requirements for a class of security devices (for example, smart cards used to provide digital signatures, or network firewalls) relevant to that user for a particular purpose. Product vendors can choose to implement products that comply with one or more PPs, and have their products evaluated against those PPs. In such a case, a PP may serve as a template for the product's ST (Security Target, as defined below), or the authors of the ST will at least ensure that all requirements in relevant PPs also appear in the target's ST document. Customers looking for particular types of products can focus on those certified against the PP that meets their requirements.
Security Target (ST) - the document that identifies the security properties of the target of evaluation. It may refer to one or more PPs. The TOE is evaluated against the SFRs (see below) established in its ST, no more and no less. This allows vendors to tailor the evaluation to accurately match the intended capabilities of their product. This means that a network firewall does not have to meet the same functional requirements as a database management system, and that different firewalls may in fact be evaluated against completely different lists of requirements. The ST is usually published so that potential customers may determine the specific security features that have been certified by the evaluation.
Security Functional Requirements (SFRs) - specify individual security functions which may be provided by a product. The Common Criteria presents a standard catalogue of such functions. For example, an SFR may state how a user acting a particular role might be authenticated. The list of SFRs can vary from one evaluation to the next, even if two targets

are the same type of product. Although Common Criteria does not prescribe any SFRs to be included in an ST, it identifies dependencies where the correct operation of one function (such as the ability to limit access according to roles) is dependent on another (such as the ability to identify individual roles).

Evaluation process

The evaluation process also tries to establish the level of confidence that may be placed in the product's security features through quality assurance processes:

Security Assurance Requirements (SARs) - descriptions of the measures taken during development and evaluation of the product to assure compliance with the claimed security functionality. For example, an evaluation may require that all source code is kept in a change management system, or that full functional testing is performed. The Common Criteria provides a catalogue of these, and the requirements may vary from one evaluation to the next. The requirements for particular targets or types of products are documented in the ST and PP, respectively.

Evaluation Assurance Level (EAL) - the numerical rating describing the depth and rigor of an evaluation. Each EAL corresponds to a package of security assurance requirements (SARs, see above) which covers the complete development of a product, with a given level of strictness. Common Criteria lists seven levels, with EAL 1 being the most basic (and therefore cheapest to implement and evaluate) and EAL 7 being the most stringent (and most expensive). Normally, an ST or PP author will not select assurance requirements individually but choose one of these packages, possibly 'augmenting' requirements in a few areas with requirements from a higher level. Higher EALs do not necessarily imply "better security", they only mean that the claimed security assurance of the TOE has been more extensively validated.

So far, most PPs and most evaluated STs/certified products have been for IT components (e.g., firewalls, operating systems, smart cards). Common Criteria certification is sometimes specified for IT procurement. Other standards containing, e.g, interoperation, system management, user training, supplement CC and other product standards. Examples include the ISO 17799 (or more properly BS 7799-2, which is now ISO/IEC 27002) or the German IT-Grundschutzhandbuch.

Details of cryptographic implementation within the TOE are outside the scope of the CC. Instead, national standards, like FIPS 140-2, give the specifications for cryptographic modules, and various standards specify the cryptographic algorithms in use.

LESSON LEARNED

This lecture presented the connections between ISO 15408 (Common Criteria for information Technologies Security Evaluation), FIPS 140-2 (Security requirements for cryptographic modules) and cryptographic algorithms.

REFERENCES

[1] Alexander W. D., Chris J. M. (2006). *User's guide to Cryptography and Standards*, Artech House.

[2] Barker, E.B., Barker, W.C., & Lee, A.(2005). *Guide line for implementing cryptography in the federal systems*- Second Edition (SP 800-21), Gaithersburg, USA: National Institute of Standards and Technology (NIST).

[3] Common Criteria for Information Technology Security Evaluation, ISO *15408*.

[4] Federal Information Processing Standards Publication (FIPS) 140-2 (2002). *Security requirements for cryptographic requirements*, Gaithersburg, USA: National Institute of Standards and Technology.

[5] ISO standards: http://www.iso.ch/

[7] NIST standards: http://www.nist.gov/ , http://www.csrc.nist.gov/

[6] Security requirements for cryptographic requirements, *FIPS 140-2*.

MODULE 2. CASE STUDY VULNERABILITIES IN RSA ENCRYPTION ALGORITHM

INTRODUCTION

The concept of public-key cryptography was invented by Whitfield Diffie and Martin Hellman, and independently by Raplhy Merkle. In public key cryptography there are two keys for each user: a public key, which is used to encrypt the message and a private key, which is used to decrypt the message. The security of a public key encryption scheme is based on the fact that it is computationally difficult to derive the private key form the public key. Usually there is a connection between the public and the private key. This paper will present in section 2 the RSA public key encryption algorithm and in section 3 same "bad implementations" of the encryption method, which can conduct to derive the private key from the public key. We also present, in section 4, some tricks, used in generation of a public/private key, to enable a smart brute force attack.

RSA ALGORITHM

RSA gets its security from the difficulty of factoring large numbers. The public and private keys are functions of a pair of large (100 or 200 digits or larger) prime numbers. Recovering the plain text from the public key and the ciphertext is conjectured to be equivalent to factoring the product of two primes. To generate the two keys, choose two random large numbers, p and q. For maximum security, choose p and q of equal length. Compute the product $n=pq$. Then randomly choose the encryption key, e, such that e and $(p-1)(q-1)$ are relatively prime. Finally, use the extended Euclidian algorithm to compute the decryption key, d, such that $ed=1 \bmod (p-1)(q-1)$. In other words $d=e^{-1} \bmod ((p-1)(q-1))$. Note that d and n are also relative prime. The numbers e and n are the public key; the number d is the private key. The two primes, p and q, are no longer needed. They should be discarded, but never revealed. To encrypt a message m, first divide it into numerical blocks smaller than n (with binary data, choose the largest power of 2 less than n). That is, if both p and q are 100 – digit primes, then n will have just fewer than 200 digits and each message block, m_i, should be just under 200 digits long. The encrypted message, c, will be made up of similarly sized message blocks, c_i, of about the same length. The encryption formula is simply: $c_i=m_i^{\,e} \bmod n$. To decrypt a message, take each encrypted block c_i and compute: $m_i=c_i^{\,e} \bmod n$. Since $(c_i)^{\,d} = (m_i^{\,e})^d = m_i^{\,e\,d} = m_i^{\,k(p-1)(q-1)+1} = m_i m_i^{\,k(p-1)(q-1)} = m_i \bmod n$. In hardware, RSA is about

1000 times slower then DES. In software, DES is about 100 faster than RSA. RSA encryption goes much faster if we choose smart values of the encryption exponent e. The three most common choices are 3 (recommended by standards PEM and PKCS#1), 17 and 65537 (recommended by standards X.509 and PKCS#1). There are no security problems with using any of these values for e (assuming you pad messages with random values), even if a whole group of users uses the same value for *e*. Private key operations can be speeded up with the Chinese remainder theorem if you save the values of *p* and *q*, and additional values such as *d* mod *(p-1)*, *d* mod *(q-1)*, and *q -1* mod *p*. The additional numbers can easily be calculated from the private and public keys.

ATTACKS ON RSA ALGORITHM

Chosen ciphertext attack Some attacks work against the implementation of RSA

These are not attacks against the basic algorithm, but against the protocol. It's important to realize that it's not enough to use RSA. This attack is presented in Schneier [8] and can be avoided if we use a one-way hash function before signing a document.

Common Modulus Attack on RSA

A possible RSA implementation gives everyone the same n, but different values for the exponents e and d. Unfortunately, this doesn't work. The most obvious problem is that if the same message is never encrypted with two different exponents (both having the same modulus), and those two exponents are relative prime (which they generally would be), then the plain text can be recovered without either of the decryption exponents. This attack is presented in Schneier [8] and is feasible if we use a common n among a group of users.

Low encryption exponent attack against RSA

RSA encryption and signature verification are faster if we use a low value for e, but that can also be insecure. If you encrypt e(e+1)/2 linearly dependent messages with different public keys having the same value of e, there is an attack against the system. If there are fewer than that many messages, or if the messages are unrelated, there is no problem. If the messages are identical, then e messages are enough. The easiest solution is to pad messages with independent random values. Most real-world RSA implementations –PEM and PGP for example –do this. To avoid this kind of attack the messages must be padded with random values before encrypting them; make sure that m is about the same size as n.

Low decryption exponent attack against RSA

Another attack, this one by Michael Wiener, will recover d, when d is up to one quarter the size of n and e is less than n. This rarely occurs if *e* and *d* are chosen at random, and cannot occur if *e* has small value. This attack can be avoided if we chose a large value for *d*.

Attack on encryption and signing with RSA

It makes sense to sign a message before encryption it, but not everyone follows this practice. With RSA, there is an attack against protocols that encrypt before signing. This attack is presented in Schneier [8].

Attack in case of small difference between prime numbers p and q

In the situation that the prime numbers *p* and *q* are close one to each other the following remark allows us to develop an efficient searching algorithm:

$$\left(\frac{p+q}{2}\right)^2 - n = \left(\frac{p-q}{2}\right)^2.$$

For the factorization of n we must test all integers $x > \sqrt{n}$ for which $x^2 - n$ is perfect square; let be this integer denoted by *y*. We can write

$$\begin{cases} p = x + y \\ q = x - y. \end{cases}$$

Thus the numbers *p* and *q* must be of large enough.

Hardware attack

Hardware attacks exploits some hardware parameters such running time, simple power analysis (SPA), differential power analysis (DFA) and fault analysis (FA). Examples of hardware attacks are:

-Timings attacks: depending the running time we can predict which of the bits from the key are zero and which of the bits of the key are one;

-SPA attack: depending the consume power of the cryptographic engine and using adequate math we can derive the key bits. This attack is suitable for crypto devices with external power supply such as smart cards, using the consumed power we can recover the code source from the inside of the smart card;

-DPA attack: in the most cases of microprocessors the consumed power depends on the value of the operand (for example erasing a bit requires a less power then setting a bit), measuring different inputs we can deduce the value of the operand;

-Fault analysis: we can induce same faults in the processor computations and using some math we can derive the key bits.

INCLUDING TRAP INFORMATION INTO RSA EXPONENTS

The developer of an RSA cryptosystem may include traps at the key generation level. Obviously the reveal of this trap information may compromise the image of the cryptographic provider. Will present some methods of including such trap information:

- hiding the low private exponent d using a permutation function of the odd numbers smaller then n (the encryption modulus);

- hiding the low public exponent e and some information about the private exponent d using a permutation function of the odd numbers smaller then *n* (the encryption modulus);

- hiding the prime number p in the product $n=pq$;

For each of the hiding methods mentioned above there are detection test.

REFERENCES

[1] IEEE P1363/D13, Standard Specification for Public Key Cryptography, Draft Version 13, November 1999.

[2] Koblitz N., Elliptic Curve Cryptosystems, Mathematics of Computations, vol. 48, nr.177, 1987, pag. 203-209.

[3] Koblitz N., Introduction to Elliptic Curves and Modular Forms, Springer-Verlag, 1984.

[4] Koblitz N., A Course in Number Theory and Cryptography, Springer-Verlag, 1988.

[5] Koblitz N., Algebric aspects of Cryptography, Springer-Verlag, 1999.

[6] Salomaa A., Public Key Cryptography, Springer-Verlag, 1989.

[7] Schroeder M.R., Number Theory in Science and Communications, Third Edition, Springer Verlag, 1997.

[8] Schneier B., Applied Cryptography, Protocols, Algorithms, and Source Code in C, 20th Anniversary Hardcover: ISBN 978-1-119-09672-6, John Wiley&Sons, 2006.
[9] Simion E., Teoria deciziilor, Cercetări Operaţinale şi Criptologie, Ed. Politehnica Press, 2001.
[10] Simion E., Criptanaliza. Rezultate şi Tehnici Matematice, Ed. Univ. Bucureşti, 2004.
[11] Welschenbach M., Cryptography in C and C++, APress, 2001.

MODULE 3. SOME EXAMPLES OF ADVANCED CRYPTOGRAPHIC ALGORITHMS & TECHNIQUES

CRYPTOGRAPHIC PRINCIPLES

Exercise 1. The method one-time pad (OTP) figures a message *m* by applying the operation *XOR* with a secret key *k*. Considering that a good key has, statistically, half the bits zero and that the *XOR* operation with zero does not modify anything, it follows that the *OTP* method leaves half of message in clear. In other words, by simply observing a text ciphered with this method, a attacker knows half of the bits of clear text. This means, in fact, that the method *OTP* is a very weak one? How can a block cipher be considered "perfect" that figures only half of the clear text?

Exercise 2. Checking the El Gamal signature involves performing the operation $a^x b^y \bmod p$ where *a, b* are fixed and *x, y* are variables. Show that the number of multiplications required for calculation is less than the number of operations required to calculate the $a^x b^y \bmod p$ by two successive exponents.

Exercise 3. We consider two prime numbers *p* and *q*. Either $i_p = p^{-1} \bmod q$ and $i_q = q^{-1} \bmod p$ and $n = pq$. What is the value resulting from the operation $q \cdot i_q + p \cdot i_p$? Can you explain how this value can be used to reduce the storage of the secret key when implementing *RSA CRT*?

Exercise 4. You want to sign two messages with the El Gamal signature algorithm. How can we calculate the values g^{k1} and g^{k2} to produce the signatures in a shorter time than the one needed to calculate two sequential signatures?

Exercise 5. We consider the Fiat-Shamir protocol where the secret *s* is chosen so that $vs^2 = 1 \bmod n$, *v* being the public key. The protocol is as follows:

- Alice chooses a random *r* and sends Bob $x = r^2 \bmod n$;
- Bob responds with a random bit *e*;
- Alice responds with $y = s^e r \bmod n$;

- Bob verifies if $y^2 = v^e \; x \; mod \; n$.

Show that the values resulting from the protocol, i.e. {*x; r; y*} define a distribution which can be simulated without using *s*. Explain why this provides the protocol with a very good security.

Exercise 6. Given a black box that runs the *AES* algorithm (12 rounds for a key 192 bits); the box contains an unknown key *k* and accepts as a parameter an entire *r* whose value can be set to 12, 11 or 10 by the user. You are allowed to enter in box clear texts as you wish. How would you attack this deployment?

Exercise 7. A system administrator has a 100-bit key that he wants to share with the two users he trusts equally. He wants access to information to be possible only when the two cooperate. How many bits of the key should give each of the two users?

Exercise 8. To speed up the verification of s_i *RSA* signatures of messages m_i, use the following idea: check if $(\prod s_i)^e = \prod \text{hash}(m_i) \mod n$ where "hash" represents full domain hash - a signature scheme based on *RSA* that first applies a hash function and then the *RSA* signature. Show that this idea is not safe for a small exponent *e* and propose a countermeasure.

Exercise 9. Why is the next context uncertain? A reliable authority generates an *RSA n* module whose factorization remains secret. The authority provides each user with from the system a pair $(e_i; d_i)$ so that $e_i d_i = 1 \; mod \; \varphi(n)$ where $i \neq j \Rightarrow d_i \neq d_j$.

Exercise 10. Let's say someone sends encrypted messages using *DES* in *OFB* operating system with a secret (fixed) initial value IV.
1) Show how a clear text attack can be performed to decrypt the messages transmitted?
2) Is *CFB* the best mode of operation?
3) What about the *CBC* mode of operation?

Exercise 11. After studying the Diffie-Hellman protocol, a young cryptographer decides to implement it. To simplify implementation, he decides to use the additive group $(Z_p; +)$ instead of the multiplier group $(Z^*_p; \square)$. As an experienced cryptographer, what do you think of this protocol?

Exercise 12. Suppose Alice and Bob use *RSA* public keys with the same module n but with different public exponents e_1 and e_2.

1) Show that Alice can decrypt messages sent to Bob;

2) Show that Alice a passive interceptor can decrypt messages sent to Alice and Bob if $gcd\ (e_1; e_2) = 1$.

Exercise 13. We assume that $n = pq$, where p and q are distinct prime numbers.

1) Compute $S = n + 1 - \varphi(n)$.

2) What are the roots of the equation $x^2 - Sx + n$? Give the expressions of these roots and explain how to find p and q using a simple algorithm for calculating entire square roots?

3) Factor n in the following two cases:

a) $n = 667$; $\varphi\ (n) = 616$;

b) $n = 15049$; $\varphi\ (n) = 14800$.

Exercise 14. Let's build a *MAC* using *CFB* deployment mode, instead of *CBC* mode: given the clear text blocks $\alpha_1; \ldots; \alpha_n$, we define the initialization vector $\beta_0 = \alpha_1$. Then we encrypt the sequence of blocks $\alpha_2; \ldots; \alpha_n$ according to the formulas:

$$\beta_i = \alpha_{i+1} \oplus E\ (\beta_{i-1}; K).$$

Finally, $MAC\ (\alpha_1 \| \ldots \alpha_n) = E\ (\beta_{i-1}; K)$. Show that it is identical to *CBC MAC*.

Exercise 15. For the *S*-box S_5 of *DES* calculate the trend of the random variable:

$$X_2 \oplus Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4.$$

Exercise 16. In a symmetrical cipher system, a k key is weak if $e_k = d_k$. Determine all the weak keys of the affine systems over Z_{15}:

PUBLIC CONTESTS

In this chapter we propose to make a brief description of the 4 problems given at MITRE Cyber Challenge[1], January 9-12, 2012. For each problem we also present a suggestion of resolution.

The first three problems are linked together, in the sense that in order to solve the second problem, it requires the password obtained after solving the first problem, and solving the second problem leads us to a useful clue in solving the third problem, Issue 3. The last problem is independent of the first three, which is actually intended to evidence of a vulnerability of ECDSA (the same type of vulnerability that has been also used to find the signature key from PlayStation3).

Issue 1. The objective of the first problem is to recognize when classical cryptography (Caesar, Vigenère, Hill, etc.) was used in the digital environment.

The hypothetical scenario is this: we find a "weird" file, which we did not create, in personal computer. This file, neededinformation.txt, is made available within the problem.

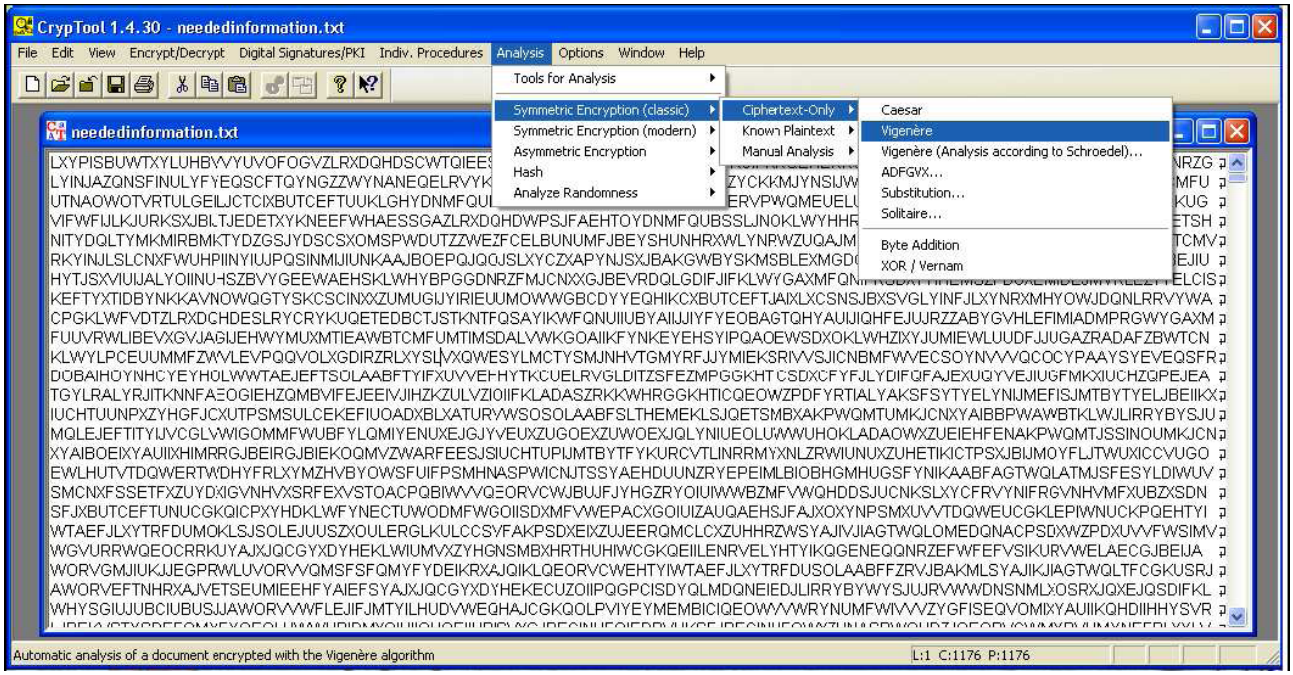
It requires decrypting the information contained in this file and finding the hidden password inside. We know that this password starts with "S", ends with "D" and consists only of capital letters.

The problem can be solved very easily using the **CrypTool** software package to make a cryptanalysis of nedeedinformation.txt: *Analysis* → *Symmetric encryption(classic)* → *Ciphertext-Only* → *Vigenère*.

From this cryptanalysis it follows for a start that the length of the key used is 6, and at the next step we get the key "SQUARE" with which we can decrypt the text contained in needinformation.txt. At the end of the decrypted text there is also the password we are looking for: "[...]THEPASSWORDFORTOMMOROWISSTRONGPASSWORDSAREGOOD".

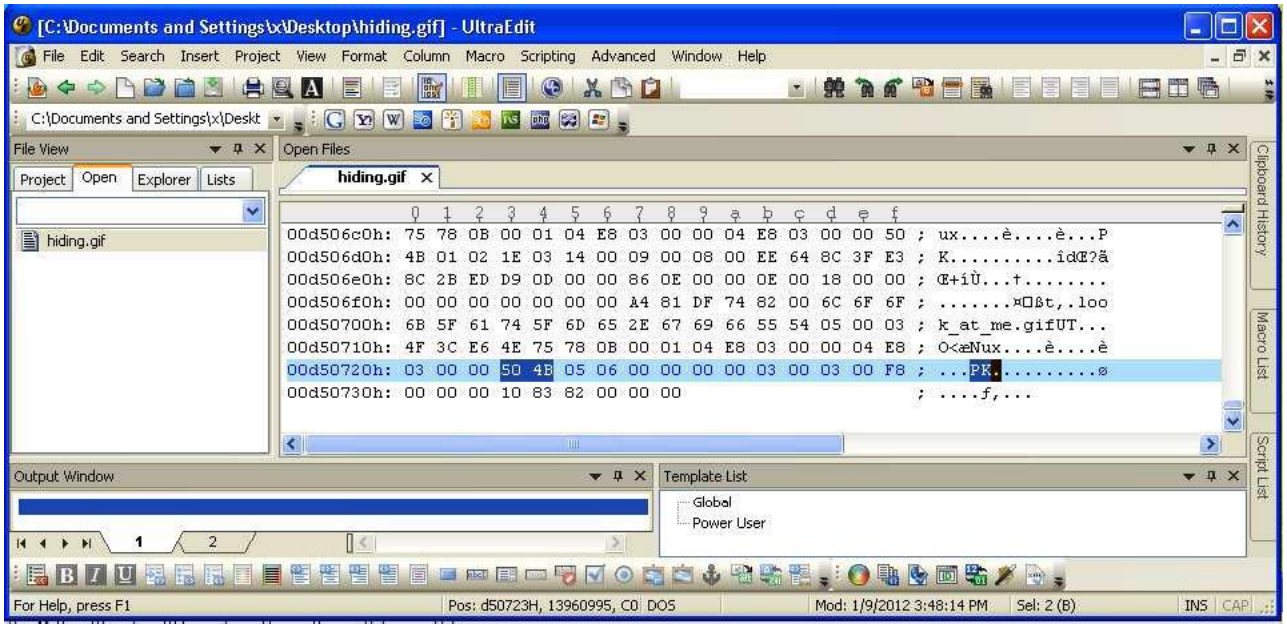
[1] http://www.iccs.fordham.edu/mitre/

Issue 2. This problem aims to show possible places where an opponent can hide information, as well as the ways in which this can be done. Specifically, the problem involves finding information hidden inside an image.



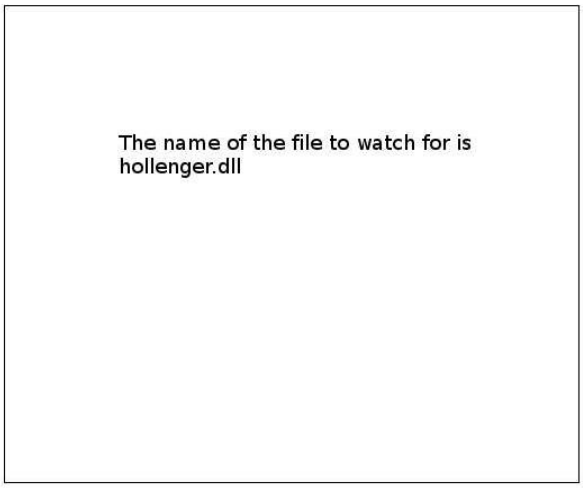
We assume we have a hiding.gif image. The requirement of the problem is to find information hidden in this image, knowing that it starts with "h", ends with "l", and the size of each letter matters. Also, as mentioned above, we will need the password obtained at the first problem.

Looking at the properties of the image hiding.gif, we notice that it has 13.3 MB, which which we find suspiciously high. To see more details, we open hiding.gif with **UltraEdit** and we notice that "PK" appears in hexa 50 4B format, which means that it is about an archive (PK represents the initials of Phil Katz).



Therefore, we change the extension and get hinding.zip. Opening this archive, we find other images, one of them (which attracts particular attention) being look_at_-me.gif. In order to see this image, however, we need the password obtained at Issue 1.

We finally find the information we are looking for, which is hollenger.dll.



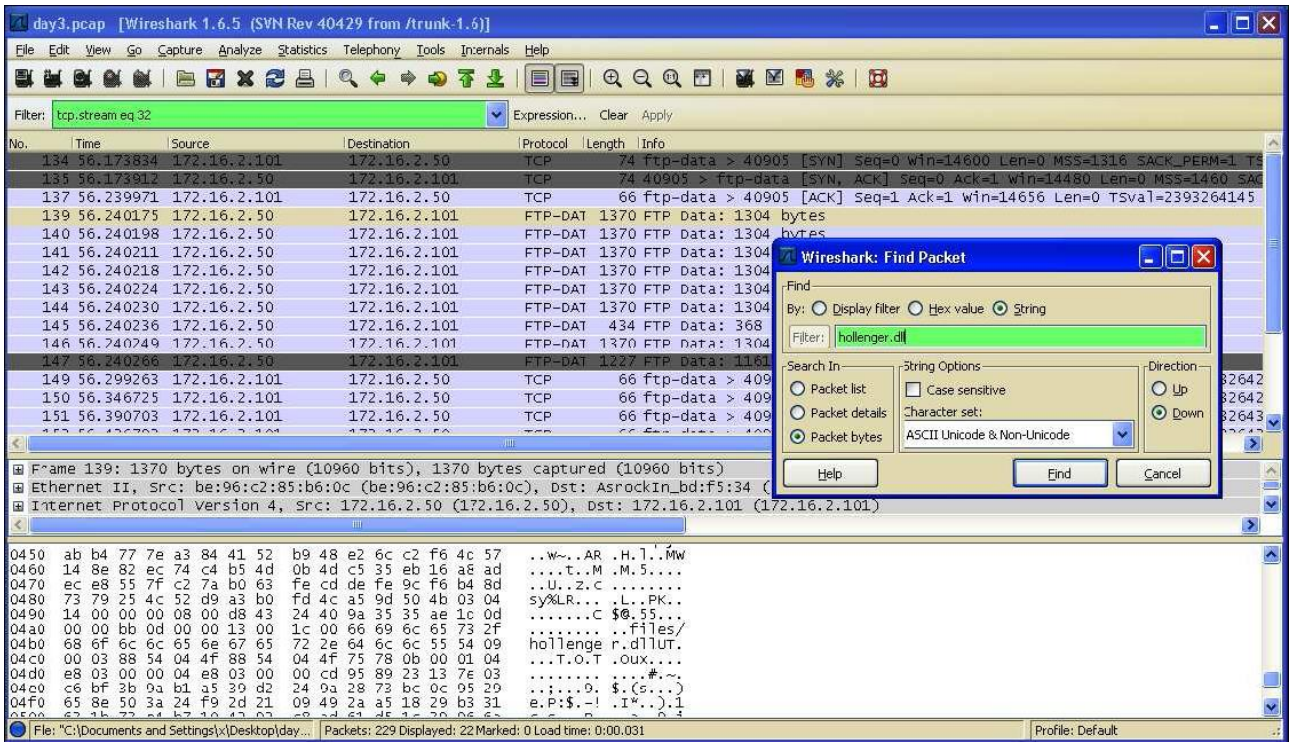
Issue 3. The third problem is related to data traffic analysis.

We assume that we have at our disposal a data traffic capture, day3.pcap.

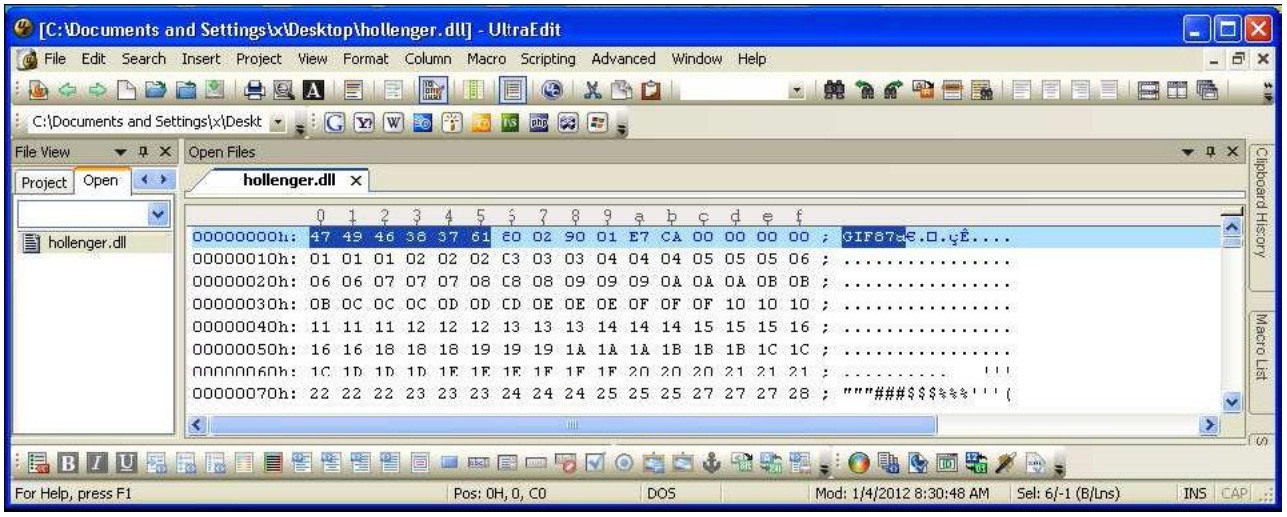
It is required to find, with the help of the answer from the previous problem, the transferred file from your personal computer to an unknown source. The answer to this problem will be the information hidden in that file. We know it starts with "P", ends with "k" and the size of each letter is important.

To be able to open day3.pcap we use **Wireshark**. We are still looking hollenger.dll as follows:

Edit → *Find Packet* → *Filter:* hollenger.dll (we select *Packet bytes* and *String*) → *Find*, and then *Follow* TCP *stream.*



We observe PK again and use the *Save as* option to get day3.zip. The archive contains several files, including hollenger.dll. We open hollenger.dll with **UltraEdit** and observe the "magic number" GIF87a (in hexa format 47 49 46 38 37 61), which means it is a picture.



So, by changing the extension we get hollenger.gif, this being an image that contains the following phrase: "The Root Password is Pengu1nsR0ck".

Issue 4. The objective of this problem is to recover an ECDLSA private key that was used to sign two different messages.

Before continuing to present this last problem, however, we recall the ECDSA signature algorithm:

The public parameters in this case are: a prime number p, an elliptical curve $E(F_p)$ and a point $G \in E(F_p)$ with ord $G = q$ and q a prime number.

The public (verification) key $V \in E(F_p)$ is built using the private (signing) key $1 \leq s \leq q-1$ so that: $V = sG$.

The signature of message m (mod q), calculated using an ephemeral key e (mod q), is defined as the pair $(s_1, s_2) = (x_{eG} \bmod q\ ,\ (m + ss_1)e^{-1} \bmod q)$, where by x_{eG} we mean the x coordinate of the point $eG \in E(F_p)$.

The signature (s_1, s_2) of the message m is checked if the following equality takes place (in which $v_1 = ds_2^{-1} \bmod q$ si $v_2 = s_1 s_2^{-1} \bmod q$): $x_{v1G+v2V} \bmod q = s_1$.

We are back to our problem now. The data provided to us is in three files: signatures.txt, parameters.der and public.oct.

The first file contains the values of the hashs and signatures for the two messages (in hexa format):

m_1=DE37B3145DB7359A0ACC13F0A4AFBD67EB496903

s_{11}=ACB2C1F5898E7578A8A861BDF1CA39E7EF41EAC0B6AAA49468DD70E2

s_{12}=BE4FA99C9D261C5F387A3ACE025702F6FB7884DD07CE18CAD48654B8

m_2=28469B02BF0D2CFC86FF43CB612EE8FC05A5DBAA

s_{21}=ACB2C1F5898E7578A8A861BDF1CA39E7EF41EAC0B6AAA49468DD70E2

s_{22}=D3540E2B13E51605F5FEB8C87EE8E176E59213F31EA8B8FFDAD077E2

To be able to see the information in the second file, parameters.der, we will use **OpenSSL** as follows:

```
openssl ecparam -inform DER -in /cygdrive/e/parameters.der
                -outform PEM -out /cygdrive/e/parameters.pem
openssl ecparam -text -in /cygdrive/e/parameters.pem  -noout
Field Type: prime-field
Prime:
    00:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:
    ff:ff:ff:ff:ff:ff:ff:ff:ff:fe:ff:ff:e5:6d
A: 0
B: 5 (0x5)
Generator (uncompressed):
    04:a1:45:5b:33:4d:f0:99:df:30:fc:28:a1:69:a4:
    67:e9:e4:70:75:a9:0f:7e:65:0e:b6:b7:a4:5c:7e:
    08:9f:ed:7f:ba:34:42:82:ca:fb:d6:f7:e3:19:f7:
    c0:b0:bd:59:e2:ca:4b:db:55:6d:61:a5
Order:
    01:00:00:00:00:00:00:00:00:00:00:00:00:00:01:
    dc:e8:d2:ec:61:84:ca:f0:a9:71:76:9f:b1:f7
Cofactor: 1 (0x1)
```

Therefore, parameters.der actually contains public parameters:

- the prime number p:

p=FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEFFFFE56D.

- the elliptical curve E: $y^2 = x^3 + 5$ considered over F_p.
- the coordinates of point G (04 mean that no compression has been applied to the coordinates of the G-point, therefore half of the following bytes will constitute the x-coordinate of the G-point, and the other half will constitute the *y-coordinate* of the G-point):

  x_G=85CEE9C98EFDFDFCF64CB522A773F1435D568173677D1D28FC00643

  y_G=58A105CC1AB1A53D77B278850776E144197F3FA4E27AA676408DFE22
- the prime number q, this is the order of point G:

  q=0100000000000000000000000001DCE8D2EC6184CAF0A971769FB1F7.
- the cofactor, which in this case is 1, which means that the G point is the generator for the group of points of the elliptical curve considered.

For the last file, public.oct, we use **UltraEdit** and find the hexa representation of the information contained inside it:

04:85:CE:EE:9C:98:EF:DF:DF:CF:64:CB:52:2A:77:3F:14:35:D5:
68:17:36:77:D1:D2:8F:C0:06:43:58:A1:05:CC:1A:B1:A5:3D:77:
B2:78:85:07:76:E1:44:19:7F:3F:A4:E2:7A:A6:76:40:8D:FE:22

This is the public key, specifically, the point V of coordinates:

x_V =85CEEE9C98EFDFDFCF64CB522A773F1435D568173677D1D28FC00643

y_V =58A105CC1AB1A53D77B278850776E144197F3FA4E27AA676408DFE22

We now have all the data needed to find out the private key s.

The important remark on which the whole resolution is based is that the values s_{11} and s_{21} are equal. In this case, if we note with e_1, respectively e_2 the ephemeral keys used for signing messages m_1, respectively m_2, it results either $e_1 = e_2 = e$, or that $e_1 + e_2 = q$.

We will show how we can find out the private key s *if* we assume it's the first case, namely that for signing the two different messages m_1 and m_2 used the same ephemeral key e. Denoting by r the common value $s_{11} = s_{21}$, we have the following two relationships:

$$s_{21} = (m_1 + sr)e^{-1} \bmod q = r_1 \text{ si } s_{22} = (m_2 + sr)e^{-1} \bmod q = r_2$$

from where we can find the private key s as follows:

$$r_1 r_2^{-1} = (m_1 + sr)(m_2 + sr)^{-1} \bmod q \Rightarrow s = (m_2 r_1 - m_1 r_2)[r\,(r_2 - r_1)]^{-1} \bmod q$$

Next, we will work in PARI/GP, therefore first we transform all values we need from base 16 to base 10. One way to do this can be the following:

```
gp> n=length(w);
gp> for(i=1,n,if(w[i]==A,w[i]=10,if(w[i]==B,w[i]=11,if(w[i]==C,w[i]=12,
        if(w[i]==D,w[i]=13,if(w[i]==E,w[i]=14,if(w[i]==F,w[i]=15)))))));
gp> W=sum(i=1,n,16 (i-1)*w[n+1-i]);
```

We find out now, assuming that the same ephemeral key *e* was used, the private key *s:*

```
gp> q=26959946667150639794667015087019640346510327083120074548994958668279;
gp> m1=126863809213821016326075805582242953806661035033 9;
gp> m2=22993418633568584075671939532439464628845372 1002;
gp> r=181872508000979720105210800739375851001549018585711307784371661334 74;
gp> r1=200421066876435888723892421805065268328322513716312598231736221912 88;
gp> r2=22255471905305126694378074733040389009439136736542793238977855911906;
gp> s=(((m2*r1-m1*r2)%q))*(bezout((r*(r2-r1))%q,q)[1])%q
        150105758150298517726420852183293232330918155587226707130866411800 71
```

We check that this is correct, which means we want to see if it is really the equality $V = sG$. For this purpose, we initialize the elliptical curve *E* over which we want to work, and then calculate the point *sG*:

```
gp> p=26959946667150639794667015087019630673637144422540572481099315275 11;
gp> E=ellinit([0,0,0,0,5]*Mod(1,p));
gp> xG=16983810465656793445178183341822322175883642221536626637512293983324;
gp> yG=13272896753306862154536785447615077600479862871316829862783613755813;
gp> G=[xG,yG];
gp> ellpow(E,G,s);
```

We get that:

x_{sG} =140916617108525568708337286057514040338636759754648142546592973471 39

y_{eG} = 93337225411387194870329268062846037753744917245016116572944899763 54

These values are equal to x_V, respectively y_V, therefore the private key *s* that we found is good.

Because the problem required the private key *s* in hexa format, we finally do the transformation of the number *s* from base 10 to base 16:

```
gp> v=vector(60);
gp> v[1]=divrem(s,16)[1];
gp> for(i=2,60,v[i]=divrem(v[i-1],16)[1]);
gp> w=vector(60);
gp> w[1]=divrem(s,16)[2];
gp> for(i=2,60,w[i]=divrem(v[i-1],16)[2]);
gp> S=vector(60,i,w[61-i]);
gp> for(i=1,60,if(S[i]==10,S[i]=A,if(S[i]==11,S[i]=B,if(S[i]==12,S[i]=C,
        if(S[i]==13,S[i]=D, if(S[i]==14,S[i]=E, if(S[i]==15,S[i]=F)))))));
```

We get that S=8E88B0433C87D1269173487795C81553AD819A1123AE54854B3C0DA7.

SYNTHESIS PROBLEMS

Questions

1. Complete: The purpose of the encryption is to ensure . . . of a communication.

(a) the authenticity

(b) the confidentiality

(c) the integrity

(d) the non-repudiation

2. The following text was obtained using the Caesar cipher system (accents, spaces and punctuation marks were removed):

MHPEUDVVHPPRLYDODLPVFHVWSRXUOHWRXIIHU.

What is his decryption?

(a) Chacun semble des yeux approuver mon courroux.

(b) Ma bouche mille fois lui jura le contraire.

(c) J'embrasse mon rival mais c'est pour l'étouffer.

(d) De grâce, apprenez-moi, Seigneur, mes attentats.

3. Encrypt the text "Attaque à l'aube" using the substitution algorithm specified below:

A	B	C	D	E	F	G	H	I	J	K	L	M
J	G	F	K	P	R	M	T	S	V	Z	D	Q

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
I	Y	B	C	W	A	O	X	E	H	N	U	L

What is the ciphered text obtained?

(a) JOOJCXPJDJXGP

(b) SHHSMYVSWSYPV

(c) JOOJCXPJBJXGP

(d) SHHSMYVSZSYPV

4. The cipher Vigenère is an improved encryption mode for simple substitution encryption systems. What does this consist of?

(a) in the successive application of several alphabetical substitutions to the same text.

(b) in the application of alphabetical substitutions that never figure a letter in itself.

(c) in the encrypting of the letters that occur most frequently (such as e) in several different symbols.

(d) in choosing several independent substitute alphabets and changing the alphabet used, in each letter, cyclically.

5. The representation on base 2 of the number 1729 is:

(a) 10010110100

(b) 11011000001

(c) 11001100011

(d) 6C1

6. We propose the following cipher algorithm: Alice and Bob want to change a message m that represents an integer between 0 and N - 1. For this, they share a common secret key k randomly extracted between 0 and N - 1. The ciphered message is obtained as $c = m + k \mod N$. What do you think of system security?

(a) Bad: the system is a variant of Caesar's system.

(b) Good, if the opponent does not know the cipher algorithm.

(c) Very good, provided they only use the k key once.

(d) Excellent: the system is a variant of the RSA algorithm.

7. Alice sends Bob a ciphered message obtained using the previous algorithm. How does Bob determine the original message *m?*

(a) $m = c + k \bmod N$

(b) $m = c - k \bmod N$

(c) $m = c \times k \bmod N$

(d) $m = c^k \bmod N$

8. Which of the following acronyms designates a block cipher algorithm?

(a) AES

(b) HMAC

(c) SHA-1

(d) NIST

9. The reverse of 17 modulo 100:

(a) is 83.

(b) is 53.

(c) is 1/17.

(d) does not exist.

10. I have in my possession a message *m* that I do not want to disclose yet, but I want to be able to prove in a few years that I already knew it in 2010 (according to the time stamp). For this, it is enough to publish today:

(a) a ciphertext corresponding to *m* with a key known only to me.

(b) a ciphered text corresponding to *m* with a key known to everyone.

(c) the image of *m* through a dispersion function (hash function).

(d) the image of *m* through a MAC using a random key.

11. The dispersion function (hash) SHA-512 returns values between 0 and 2^{512} - 1. One calculates images by this function randomly. What is the order of magnitude of the numbers for which the values must be calculated by this function in order to find 2 values that have the first 20 bits equal?

(a) 20

(b) 1000

(c) 1000000

(d) 2^{512}

12. We build a pseudo-random number generator that initializes with $x0$ with a value between 0 and 999 and determines $x_{n+1} = 500x_n+789 \bmod 1000$. Under what conditions would you use this generator?

(a) To produce random numbers between 0 and 999, if there is no interest in the security level.

(b) To generate a *one-time pad* key.

(c) For the construction of a dispersion function.

(d) Never.

13. How is the secret key required to encrypt communication when connecting to a secure website is obtained?

(a) It is obtained from the password entered for login, through a key derivation algorithm such as PBKDF (Password Based Key Derivation Function).

(b) It comes from the public key of the server, contained in a certificate.

(c) It comes from the private server key, disclosed to the client after the connection is established.

(d) It is obtained by an exchange of keys between the client and the server, such as the exchange of Keys Diffie-Hellman.

14. What is the difficulty of factoring a prime number on 1024 bits today?

(a) It's simple!

(b) The number can be factored with the help of several thousand current computers running between 1 and 2 years.

(c) No one can do that at the moment, but maybe it will be done by agents like the NSA.

(d) This will not be possible for several millennia.

15. The RSA algorithm (without padding) is a cipher algorithm:

(a) symmetrical, block type.

(b) symmetrical, fluid type (flow).

(c) partly homomorphic.

(d) based on identity.

16. Let the Gee generator be described by three **LFSR**i displacement registers (whose feedback polynomials are primitive grade 19, 21 and 24 respectively) and the output of the formula: $y(t) = a_1(t) \cdot a_3(t) \oplus \bar{a}_1(t) \cdot a_2(t)$. What is the complexity of the **LC** and the **P** period of this generator?

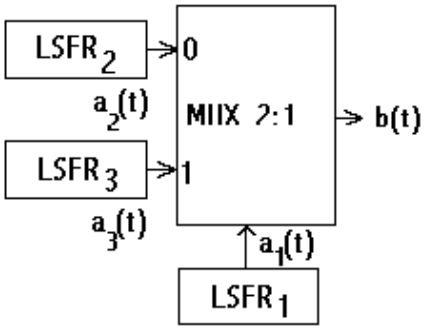


Figure: Geffe Generator.

(a) LC= 640, P= 2^{64}*:*

(b) LC=64 , P=$(2^{19} - 1)(2^{21} - 1)(2^{24} - 1)$*:*

(c) LC=876 , P=$(2^{19} - 1)(2^{21} - 1)(2^{24} - 1)$*:*

(d) None of the answers are correct.

17. Consider the sequence given by the binary representation (written on 8 bits) of the number *i, i* = 0, ..., 255:

00000000 | 00000001 | 00000010 | 00000011 | 00000100 | ... 11111111

What is the frequency test statistic applied to this binary sequence? Is the sequence random, relative to the frequency test, at the risk of order 1 of 5%?

(a) $f_{tf} = 256$, the string is not random.

(b) $f_{tf} = 1$, the string is random.

(c) $f_{tf} = 0$, the string is random.

(d) none of the answers is correct.

18. Which of the following statements are true:

(a) Successful attack on two preimages of a hash function involves successful collision-generating attack.

(b) Successful attack by generating collisions on a hash function involves successful attack on two preimages of the same hash function.

19. Which of the following statements are true:

(a) A moving register of length *n* shall have a period of $2^n - 1$*:*

(b) A moving register of length *n* shall have a maximum period of $2^n - 1$*:*

(c) A moving register of length *n,* with the characteristic primitive polynomial, has a period of $2^n - 1$.

20. The probability of collision of two messages of length *n* bits processed by the same ideal hash function, which has the output on *m* bits, is:

(a) 2^{-m}.

(b) 2^{-n}.

(c) 2^{-mn}.

(d) 2^{m-n}.

(e) 2^{n-m}.

(f) None of the above values.

21. Let the Galois GF(3^2) extension be generated by the polynomial root $X^2 - X - 1$. In this extension, the value of $log_{2\alpha+1}(1 + \alpha)$ is:

(a) 8.

(b) 4.

(c) 2.

(d) 5.

(e) 6.

(f) None of the above values.

22. Jacobi's symbol $\left(\frac{6278}{9975}\right)$ este:

(a) -1.

(b) 0.

(c) 1.

(d) None of the above values.

23. In the context of a judicial action, one of the two judges on duty is to be appointed. Since neither of the two wishes to do so voluntarily, it is proposed the decision method based on the result obtained from the throwing of a coin. Thus, Judge *A* chooses "head" or "tail" and Judge *B* throws away the coin, the decision being taken following the result obtained. Considering that *A* and *B* in different physical locations is proposed by the cryptographer, the following protocol.

STEP 1. Participant *A* chooses $x = 0$ ("head") or $x = 1$ ("tail") and a random key *k*. The x value is ciphered using the DES algorithm: $y = \text{DES}(x; k)$.

STEP 2. Participant A transmits the value of *y to B*.

STEP 3. *B* throws a coin and communicates *to A* the result obtained.

STEP 4. A communicates the key *k to B*.

STEP 5. *B* deciphers *y,* using the DES algorithm and gets what *A* chose.

The cryptographer states that "participant *A* cannot change its option" due to the *y* value transmitted. Show the following:

a) Using "birthday attack" user *A* can cheat;

b) What is the complexity of the attack in point (a)?

c) What is the requirement of the cryptographic primitive that ensures the validity of the statement "Participant *A* cannot change its option";

d) Correct the protocol so that the attack in point a) is no longer possible.

24. Let p be a prime number and let G be the set of all elements $x \in \mathbb{Z}_{p^2}$ that satisfy the relation $x \equiv 1 \bmod p$. Show that:

a) G is multiplier group;

b) $|G| = p$;

c) $L : G \to \mathbb{Z}_p$ defined by $L(x) = (x-1)\, p^{-1} \bmod p$ is an isomorphism of groups;

d) $p + 1$ is a generator of G *and that* isomorphism is the logarithm in the base $p + 1$ of G. In other words, we have: $(p + 1)^{L(x)} \bmod p^2 \equiv x$ for any x.

25. Consider the DSS signing algorithm with parameters $p, q, g,$ a hash H function and a secret key x. Within the implementation, it is precalculated the pair (k, r) that satisfies the relationship $r = (g^k \bmod p) \bmod q$, this being used for generating signatures. Recover the private signature key.

26. The Wired Equivalent Privacy (WEP) protocol used in the IEEE 802.11 standard is used to protect data in wireless transmissions. The WEP protocol has a 40-bit K key, shared between the entities that communicate and is used to protect each "frame"[2] transmitted. In this exercise we will assume that the K key is fixed and does not change its value. In order for user A to transmit a "frame" to B we proceed as follows:

STEP 1. CRC encoding: Giving a message M of n-bits (n is constant), A calculates a control amount of 32 bits, L(MM), where L is a linear function[3] that does not depend on K. The clear text, of length $(n + 32)$ bits, is $P = M \,||\, L(M)$.

STEP 2. A encrypts P with the RC4 algorithm, the K key and the 24-bit IV vector specific to each "frame" transmitted. The ciphered text will be $C = \mathrm{P} \oplus RC4(IV; K)$.

STEP 3. A transmits to B on the radio channel $(IV; C)$.

Questions:

[2] data package.

[3] $L(X \oplus Y) = L(X) \oplus L(Y)$.

a) Certain manufacturers specify that the WEP protocol has a security of 40+24=64 bits of key. What do you think of that fact? Justify the answer.

b) How does *B* extract the original Message *M?*

c) In implementations, the 24-bit IV vector is chosen randomly at each "frame" transmitted. You show that this leads to security issues when data traffic is high. Propose a way to fix the problem.

d) Let's examine another WEP protocol security issue. We will assume that the attacker intercepts the data (*IV, C*) transmitted by *A*. Show that the opponent, even if he does not know the *K* key, he can easily calculate a ciphered text *C** (*C** ≠ *C*) and retransmit (*IV, C**) without *B* being able to detect this. How many choices do we have for *C**? What security property is being violated?

27. Decipher, using the RSA-CRT algorithm, indicating the meanings of the algorithm elements, the message:

C = 9686 9613 7546 2206 1477 1409 2225 4355 8829 0575 9991 1245 7431 9874 6951 2093 0816 2982 2514 5708 3569 3147 6622 8839 8962 8013 3919 9055 1829 9451 5781 5154.

The clear text is in English.

The parameters of the algorithm are as follows:

a) the cipher exponent is *e* = 9007,

b) *p* = 3490 5295 1084 7650 9491 4784 9619 9038 9813 3417 7646 3849 3387 8439 9082 0577,

c) *q* = 0003 2769 1329 9326 6709 5499 6198 8190 8344 6141 3177 6429 6799 2942 5397 9828 8533.

28. Consider the prime numbers *q* = 7541 and *p* = 2*q* + 1. Let *α* = 604 and *β* = 3791.

a) Show that *ord* (*α*) = *ord* (*β)* = *q* in $\square_q$. Moreover, show that *α* and *β* generate the same subgroup *G in* $\square_p^*$.

b) Define the hash *function h*: $\square_q \times \square_q \rightarrow G$, *h* ($x_1$, x_2) = $x_1^{\alpha}\, x_2^{\beta}$. Compute *h* (7431, 5564) and *h* (1459, 954).

c) At the previous point you obtained a collision for *h*. Use it to calculate the discrete logarithm $dlog_{\alpha}\ \beta$

d) Using the computed discrete logarithm, determine other collisions for *h*.

Answers

1. *Answer:* (b). For authenticity, USE MAC or electronic signatures. For integrity, depending on the level of demand, you can use control amounts, hash functions, MAC, etc.

2. *Answer:* (c). You can help yourself by the position of the duplicate letters. Additional question: where do these lyrics come from?

3. *Answer:* (a). The letters on the second line are the images of the front line, not the other way around.

4. *Answer:* (d). Method (a) is only a normal substitution (composition of 2 permutations is still a permutation). Method (b) is weaker than the first one because it exposes more information about the clear text. Method (c) is called polyalphabetic substitution.

5. *Answer:* (b). It is enough to calculate the rest of the division of 1729 to 4 to remove (a) and (c). (d) is 1729 in hexadecimal (i.e. on base 16).

6. Answer: (c). The algorithm is a variant of the one-time pad. It provides perfect security if the encryption key is only used once. It can also be considered a variant of Caesar's cipher, but applied to a single letter and with a randomly chosen gap. Used in this way, Caesar's cipher would be safe. The system has nothing to do with RSA.
Answer (b) would not satisfy Kercko's principle: an encryption system must remain secure when the opponent knows everything about it, except the key used.

7. Answer: (b). The inverse operation of the addition with $k \bmod N$ is the decrease by $k \bmod N$.

8. *Answer:* (a). HMAC is MAC, SHA-1 is a dispersion function and NIST is an American standardization agency.

9. *Answer:*(b). $53 \times 17 = 1 \bmod 100$.

10. *Answer*: (c). At the time of the message disclosure, everyone will be able to verify that the hash is correct and that the message *m* was known at the time of calculating this hash. The method does not allow the message *m* to be revealed.
A cipher of *m* with a key known only by the one who makes the encryption does not guarantee anything: it can also publish a random word so that later choose the key that corresponds to a correct encryption. The same problem arises in the case of MAC. A key known to everyone would lead to the determination of the clear text *m*, which would be equivalent to the disclosure of the message *m*.

11. *Answer:* (b). According to the birth paradox, in order to obtain a collision on the first 20 bits of the dispersion function, it is necessary to calculate the value of the hash for $\sqrt{2^{20}}$, that is about 1000 numbers.

12. *Answer:* (d). The value of x_n is constant, equal to 289, starting with the third term. So it is not about random appearances.

13. *Answer*: (d). The session key is determined by an exchange of keys.

14. *Answer*: (a). Factoring of a prime number is immediate.

15. *Answer:* (c). The property of homomorfism is that the RSA cipher of the product of two messages (modulo *N*) is the product of the ciphers corresponding to the two numbers.
The rest of the variants are wrong, because the RSA is a cipher with a public key, so asymmetric.

16. *Answer*: (c). Apply the properties of the Geffe generator.

17. *Answer:* (c). In this situation the sequence subject to testing is ideal, the number of bits of 0 is equal to the number of bits of 1, that is 1024.

18. *Answer:* (a).

19. *Answer:* (b), (c). A displacement register of length n has $2^n - 1$ possible states (null state is excluded). If the characteristic polynomial is primitive, then it generates all possible states.

20. *Answer:* (a). The number of possible outputs, of an ideal hash function with m bit output, is $2m$.

21. *Answer:* (e).

22. *Answer:* (a).

23. *Answer:* a) *A* will determine two keys *k* and *k** so that: *DES* ("tail", *k*) = *DES* ("head", *k**). To do this, proceed as follows:
i) A will build two lists (*DES* ("tail", *k*); *k*) and (*DES* ("head", *k**); *k**), for all keys *k* and *k**. The lists are sorted in relation to the first field of each entry (i.e. *DES* ("tail", *k*) and *DES* ("head", *k**), respectively).
ii) A will look for collisions within these lists and will obtain *k*, *k** so that: *DES* ("tail", *k*) = *DES* ("head", *k**).
iii) After throwing the coin, A communicates to B the key *k* or *k**, depending on the case.
b) The complexity of the previous attack is the search for collisions within the two lists, 64-bit, *DES* ("tail", *k*) and *DES* ("head", *k**). According to the "birthday attack" it only takes 232 evaluations of the DES algorithm to cause a collision.
c) The requirement of the cryptographic primitive is that the functions: $k \rightarrow$ *DES* ("tail", *k*) and $k \rightarrow$ *DES* ("head", *k*) be collision resistant.
d) You can use a 128-bit block cipher algorithm, for example AES (in this case "birthday attack" needs 264 Evaluations of AES). As an alternative one can use a hash *h* collision

resistance function. Participant *A* chooses $x \in$ {"head", "tail"}, a random value *r* and calculates $y = h(x \| r)$. After *B* makes the choice, *A* can reveal *x* and *r.*

24. *Answer:* a) We will show that $G = \{x \in \mathbb{Z}_{p^2} | \ x \equiv 1 \bmod p\}$ with respect to the multiplication mode p^2, is a group. For this will be checked the following: the stability of the operation, the associativity, the neutral element and the symmetry element.

b) Any element *a of* $\mathbb{Z}_{p^2}$ can be written uniquely $a = a_1 + a_2 p$, where a_1 and a_2 are integers that satisfy the relation $0 \leq a_1; a_2 \leq p\text{-}1$. Any element *a of* $\mathbb{Z}_{p^2}$ is in *G* if and only if the corresponding element a_1 is equal to 1, hence the fact that $|G| = p$.

c) Consider $a = 1 + kp$, $0 \leq k < p$ and $b = 1 + lp$, $0 \leq l < p$ are elements of *G*. Check that *L* is homomorphism: $L(a \square b) = k + l \bmod p$ and $L(a) + L(b) = k + l \bmod p$, so $L(a \square b) = L(a) + L(b)$. Directly check the injectivity and surjectivity of *L,* so *L* is an isomorphism of groups.

d) We have to show that any element *of* $a \in G$ can be written as a power of $p + 1$. From Newton's binomial it results that:

$$(p + 1)^2 \bmod p^2 = \sum_{i=0}^{n} \binom{n}{i} p^i \bmod p^2 = 1 + np$$

So, $p + 1$ generates *G.* For any $y \in G$ we have: $y = log_{p+1}(x)$ if and only if $x = (p + 1)^y \bmod p^2$.

Because $(p + 1)^y \bmod p^2 = 1 + py$, we get:

$$y = \frac{x - 1}{p} \ mod \ p = \mathrm{L}(x)$$

This logarithm function is the basis of the Okamoto-Uchiyama cryptographic algorithm.

25. *Answer:* Let us consider the signatures for the messages *m* and *m**. The signatures are (*r, s*) and (*r, s**)*:* We have:

$$s = \frac{H(m) + xr}{k} \ mod \ q$$

$$s* = \frac{H(m*) + xr}{k} \ mod\ q.$$

We deduce that

$$k = \frac{H(m) - H(m*)}{s - s*} \ mod\ q.$$

We will then calculate $r = (g^k \bmod p) \bmod q$ and finally recover x from the following formula:

$$x = \frac{ks - H(m)}{r} \ mod\ q.$$

26. Answer: a) It is not correct to calculate the size of the key by summarizing the size of the two inputs in the algorithm because only one is secret. So the key size is 40 bits, not 64-bit.

b) First *B* rebuilds the clear text $P^* = C \oplus RC4(IV, K)$. Subsequently P^* is divided into two parts $P^* = M^* \,||\, Q^*$, where M^* is n bit and Q^* is 32 bits. *B* calculates $L(M^*)$ and compares it with Q^*. *B* accepts the message M^* if and only if $L(M^*) = Q$, otherwise it will reject the message M^*.

c) According to the "birthday paradox" choosing *IV* randomly at each "frame" it follows that every $2^{24/2} \approx 5000$ frames there is a collision for two IV *of* the 5000 transmitted from/ to the same user. In this situation we have a collision in the key strings, which can lead to information about the clear text. An alternative is to increment *IV.*

d) Let $M^* = M \oplus \Delta$ be a new message, where Δ is a string of n bits. We will calculate the difference between the new encryption text C^* and C:

$$\begin{aligned} C^* \oplus C &= (P^* \oplus RC4(IV, K)) \oplus (P \oplus RC4(IV, K)) \\ &= P^* \oplus P \\ &= (M \oplus M^*) \,||\, (L(M) \oplus L(M^*)) \\ &= \Delta \oplus L(\Delta): \end{aligned}$$

So, for any non-zero Δ, the opponent knows that $C^* = C \oplus (\Delta \,||\, L(\Delta))$ who check the CRC. In conclusion it has $(2^n - 1)$ possibilities for choosing Δ (and C^*).

The property violated is that of *the integrity of the message.* One conclusion that emerges from this exercise is that CRCs (with or without a key) provide us with protection against transmission errors and not against a malicious opponent.

27. *Answer:* By direct calculations we will obtain: $d = e^{-1}$ = 0001 0669 8614 3685 7802 4442 8687 7132 8920 1547 8070 9906 6339 3786 2801 2262 2449 6631 0631 2591 1774 4708 7334 0168 5974 6230 6553 9685 4451 3277 1090 5360 6095 mod$(p - 1)(q - 1)$.

Then, by direct calculation or using CRT, it follows:

$M = C^d$ = 20 0805 0013 0107 0903 0023 1518 0419 0001 1805 0019 1721 0501 1309 1908 0015 1919 0906 1801 0705 mod N, $N = p\,q$.

Using the encoding space = 00, A = 01, B = 02, ..., Z = 26, we get the text clear: "THE MAGIC WORDS ARE SQUEAMISH OSSIFRAGE".

Co-funded by the Erasmus+ Programme of the European Union

"The European Commission support for the production of this publication does not constitute an endorsement of the contents which reflects the views only of the authors, and the National Agency and Commission cannot be held responsible for any use which may be made of the information contained therein".