

Ministry of Education, Culture and Reasearch of Republic of Moldova  
 Tehnical University of Moldova

FCIM, Software Engineering

Report  
for laboratory work No. 6

# For the course „Cryptographic Method of Data Protection”

Done by: Alexandra Konjevic, gr. FAF-213

Marked by: Cătălin MÎȚU

Chișinău – 2023

**Subject: Cryptography using public keys**

**Objectives:**

Task 2: Using the Wolfram Alpha platform or the Wolfram Mathematica application, generate the keys, perform the signing and validation of the digital signature for the message "m" that you obtained in laboratory work number 2. The signing will be done using the RSA signature. The value of "n" must be at least 3072 bits. The hash algorithm will be selected from the list below, according to the formula i = (k mod 24) + 1, where k is the student's order number in the group list, and i is the index of the hash function in the list:

1. MD4
2. MD5
3. MD2
4. MD6-128
5. MD6-256  
   ........

23. Haval224,4 🡨 my variant

Task 3: Using the Wolfram Alpha platform or the Wolfram Mathematica application, perform the signing and validation of the digital signature for the message "m" that you obtained in laboratory work number 2. The signing will be done using the ElGamal signature (the values of "p" and the generator are provided below). The hash algorithm will be selected from the list below, according to the formula i = (k mod 24) + 1, where k is the student's order number in the group list, and i is the index of the hash function in the list:

1. NTLM
2. MD4
3. MD5
4. MD2
5. MD6-128  
   ........

23. Haval224,4 🡨 my variant

**Theory:**

**Hashing:**

1. **Definition:** Hashing is a process of taking an input (or 'message') and producing a fixed-size string of characters, which is typically a hexadecimal number. The output, called the hash value or hash code, is unique for every unique input, and even a small change in the input will produce a significantly different hash value.
2. **Purpose:** Hashing is primarily used for data integrity and security. It ensures that data hasn't been tampered with during transmission or storage by generating a checksum, often referred to as a digital fingerprint, for the data.
3. **Properties of Hash Functions:** A good hash function should have the following properties:
   * Deterministic: For the same input, it will always produce the same hash value.
   * Fast to compute the hash value.
   * Pre-image resistance: Given the hash value, it should be computationally infeasible to determine the original input.
   * Collision resistance: It should be extremely unlikely for two different inputs to produce the same hash value.
   * Avalanche effect: A small change in the input should result in a drastically different hash value.
4. **Common Hash Functions:** Some commonly used hash functions include MD5, SHA-1, SHA-256, SHA-512, and more. However, MD5 and SHA-1 are no longer considered secure for cryptographic purposes due to vulnerabilities.

**Haval224,4:**

HAVAL (HAsh of VAriable Length) is a family of cryptographic hash functions. HAVAL224,4 specifically refers to a variant of HAVAL that produces a 224-bit hash output and operates on 4 passes.

The HAVAL algorithm operates in several steps:

1. **Padding:** Similar to many cryptographic hash functions, HAVAL pads the input message to a multiple of a fixed block size before processing it.
2. **Breaking the Message into Blocks:** The padded message is divided into blocks for processing. HAVAL224,4 processes the message in blocks, and each block goes through four passes.
3. **Compression:** Each block undergoes a compression function. In HAVAL, these compression functions include bitwise logical operations (such as XOR, AND, OR) and modular addition. These operations are applied to subsets of the input block in a series of rounds.
4. **Transformation Rounds:** Each pass involves a certain number of transformation rounds. For HAVAL224,4, there are four passes, and within each pass, there are several rounds where the compression function is applied to the block multiple times, with the results from each round affecting the subsequent rounds.
5. **Finalization:** After processing all blocks of the input message, the final hash value is obtained. For HAVAL224,4, it produces a 224-bit hash value as the output.

**Digital Signatures:**

1. **Definition:** A digital signature is a cryptographic technique that verifies the authenticity and integrity of a digital document or message. It is the digital equivalent of a handwritten signature or a seal on a physical document.
2. **Components of a Digital Signature:**
   * **Private Key:** The private key is known only to the owner and is used to create the digital signature.
   * **Public Key:** The public key is shared with others and is used to verify the digital signature.
   * **Hash Value:** A hash of the message is created using a secure hash function.
   * **Signature:** The private key is applied to the hash value, creating a unique signature for that message.
3. **Process of Digital Signing:**
   * The sender generates a hash value of the message using a secure hash function.
   * The sender signs the hash value with their private key, creating a digital signature.
   * The signature and the message are sent to the recipient.
4. **Verification of Digital Signature:**
   * The recipient receives the message and signature.
   * They calculate the hash value of the received message.
   * They use the sender's public key to decrypt and obtain the hash value from the digital signature.
   * If the calculated hash value matches the decrypted hash value, the signature is considered valid, and the message is deemed authentic and unaltered.
5. **Benefits:** Digital signatures provide data integrity, authentication of the sender, and non-repudiation, which means the sender cannot deny sending the message.
6. **Common Digital Signature Algorithms:** Common digital signature algorithms include RSA, DSA, and ECDSA.

Hashing and digital signatures are fundamental tools in information security and are widely used in various applications, such as secure communication, data integrity verification, and electronic document signing. They play a crucial role in ensuring the trustworthiness of digital information and transactions.

## Task Completion:

## The tasks will be solved using the website <https://www.wolframalpha.com>, as well as other tools. The text that I decrypted in the 2nd lab, 22nd variant, is the following:

## while herbert yardley may be the best known cryptologist,uncontestably the greatest is william frederick friedman. unlike hiscontemporary, his eminence is due most emphatically to what he did.indeed, two more dissimilar men in a single field can scarcely beimagined. Where yardley was rabelaisian, outgoing, superficial, free andeasy with the details of a good story, and ever ready for the main chance,friedman tended toward introversion, depth of study, personal security,timidity, dedication, and accuracy, nicety, and validity of work. Despitethe relative drabness of these personal traits—or perhaps because ofthem, friedman's theoretical contributions and his practical attainmentsexceed those of any other cryptologist. yardley's career was like anamazing skyrocket that explodes in fantastic patterns against theheavens. friedman's was like the sun.he was born wolfe friedman on september 24, 1891, in kishinev,russia, the oldest son and second child of frederick and rosa friedman.his father, a rumanian who spoke eight languages and worked as aninterpreter for the russian post office, emigrated to america in 1892, at which time his son's name was changed to william. the family settledin pittsburgh, where his father managed a sewing machine agency.william graduated in 1909 as one of the ten honor students in a class of300 at pittsburgh central high school; he then went to work as chiefclerk in the erie city iron works, a firm that sold steam engines. aboutthat time the back-to-the-farm movement called to city boys, and in thefall of 1910, friedman and three friends enrolled in michiganagricultural college, whose chief attraction was that it was tuition-free.but friedman soon discovered that farming held little interest for him.he was an inventive young fellow who liked to fix things and had writtensome science fiction for his high-school paper; he was rapidly coming tothe conclusion that he liked science. at the end of the term he learnedthat tuition was also free in a scientific field allied to agriculture—genetics—at one of the ivy league universities, cornell. He borrowedtrain fare and arrived in ithaca, new york, in february, 1911, where hegot a job waiting on tables. after commencement in february of 1914, heattended graduate school, managing to fall in love twice, once with abrunette, once with the blonde daughter of a movie-house owner. whilehe was there, a wealthy textile merchant, george fabyan, whomaintained laboratories in acoustics, chemistry, genetics, and cryp-tology (to try to prove that bacon wrote shakespeare's plays) on his 500-acre estate, riverbank, at geneva, illinois, decided that he needed ageneticist to improve the grains and livestock on his farm. he applied tocornell for a "would-be-er," not an "as-is-er," and hired friedman, tobegin june 1, 1915.fabyan was a man of no formal education but of intelligence andenergy. he had a great desire to be "somebody," and that desiremotivated his subsidizing the baconian studies: proof of thisrevolutionary thesis would cover its patron as well as its actualdiscoverers with glory. he himself read little, but he absorbed enoughfrom those around him to make his talk on almost any subject soundimpressive—at least superficially. he was autocratic, never allowing hisstaff to disagree with him, but otherwise not unpleasant so long asemployees recognized that he was boss. a cardinal article of faith withhim was that a well-executed sales campaign could put across almostanything.

## It will be used as the target of the digital signature for the tasks.

## Task 1:

## For calculating the hash, I used the following tool: https://md5hashing.net/hash/haval224,4.

## And got the result for the hash (in hex):

## 49937c129dd4738e8f31de7a5bf87e2827da7032b2bc79bf94168439

## Then, to convert to decimal, the following tool was used: <https://www.rapidtables.com/convert/number/hex-to-decimal.html>

## With the result:

## 7748468961461525902282151250356838907585142880510257296443803337785

## Next comes the key generation for RSA. The following commands were ran in Wolfram Alpha for generating the primes P and Q:

## P = RandomPrime[{2^1536, 2^1792}]

## Q = RandomPrime[{2^1792, 2^2048}]

## Results:

## P = 41796554082063829467849149489136611475498483139708172462041849035620540451635113582888000863705432131741100859333489650657791915104513129722585029226956714675319604874718617376059890819831091102577993992263713245780592949725342715285200793312571345676985706966300529291265126128408374850567996419642717405478526334508199695874740689241295470243373261451475456125647046188669510730778080572515741274740223297080718161422659901851406718808331870497595617270564190841742060924025469212570248621328513541404069538378234664658751288123696764831

## Q = 3337249033747613701782900610012396972607827375568542661788339917086290926322558514796452464872084788575244759626108125191671117831299915938879508367314791371666661988285130264240478354187235445697665416342180385374245966691349037092703485337776205480283282920507280670267734122260069521253711604797798817507290630977218184418711224054917748174747673455694991236471654303474140700408408184139566951446179749068770780641276255609523933124440930392439276798601274462535405136667531847298088853416670741194038717673084534851369602273340895101292513812882527597688947800660350841156804597650838874347226869340963923428771

## The exponent e is chosen as 65537.

## The public key is calculated as n = P\*Q, using the tool <https://www.dcode.fr/big-numbers-multiplication> and the result is:

## 139485509724347394054917423676449076696151692745415852714338389541751437622509613334790692787867224438116807343233214984787230462026426454511145709419385507243528375251987631349587111286317582198575695137338359539271021620444731829474689353014249301622755111806220079536318050707470354165321957922228807717991660303013908836339717469996486819585338329404264180906966491760987750192929205644334372957845019718736208145142245961302133570494761286276528210330002891871808852462265113841513513919887602795328712833359129804226365797122012199244800920615558196457869745579235076225879929622399774434418844184410465822508515024647957980012676367175739055793053048827585282683812756429270116718961779469141719954138653882561174788452937374654995672462938806176778143651181242008116152966565282096803278468608900025611069114407192865584550065968810081057060631177510790233120714800115249535622815222550111305784151862319042435213685967013816089029476207570924623733662945918560400360898182306384479077972152674500219309271786082048733076840247021209253358046522625778105342597409339239564663928738783928093416117618830827334515588570303532053850898271777566352701

## We now have the public key that consists of the pair {n, e}. Next, I calculated the LCM of (P-1)\*(Q-1), which equals to

## 139485509724347394054917423676449076696151692745415852714338389541751437622509613334790692787867224438116807343233214984787230462026426454511145709419385507243528375251987631349587111286317582198575695137338359539271021620444731829474689353014249301622755111806220079536318050707470354165321957922228807717991660303013908836339717469996486819585338329404264180906966491760987750192929205644334372957845019718736208145142245961302133570494761286276528210330002891871808852462265113841513513919887602795328712833359129804226365797122012199241463671581810582756086844969222679253272102246831231772630504267324174896185914713297423451311123729451005159555469425152774456680050798641354987811195352983892169964989818186188955333406368439306672464205653907672809591930605248200737348023914357999143935657281788264240756998154859631085057868221285920834484799406980034468732505038231200831583876640726711694461929562425259017399799256539740937887421717812912547622143317047584991780331604867756532768640100131392298431329514011453563505262153620113363233654629759056238144706797879947430408054932230861191898068337151472636306921381086423041965277642689946159100

## With this information, the inverse of raising to the power of 65537 can be found using the extended euclid’s algorithm <https://www.dcode.fr/modular-inverse> and the result is calculated by the formula e^(-1) mod Ф.

## d= 78291262572441809822087331277571132127926820233457774113812619730737242060881885446698439571565617146896055024197534418960255619659766195114095776751943114331647638488859194335940947688591059419482230581610259176527526897905907507930275231558190954120467015392706495960197438016300669514493617684196510244691139726358646818511016786453770658688171116272271509142358704234675593799638384876128597727304103794096638793644163109182583630478200007868564173169189257633161246896019381611304676893099553974490237599754575123189448187239776290478618813176326384892238805441092150332355986406910399633126525466126306873311242088112756483459415084423382589899567310133890907258123939576456707304802188985038580834675839021910687427550136000120480745163876573443158839116336024765615199765929011993202460795475999531563792150649656095479864102453881053418626475825652083066547525792015727338599766578102935588763325738953768908479967280342621273481984343039015946172094266103047346119588905275804874466246945745659180276736136425398161855761910385215528122281849271203804265575774905989993859351216596918213283648073334405311908541785606055687606889819282995398973

## Now, we have the private key, which consists of the pair {n, d}. As the final step, the digital signature is calculated by raising the hashed value to the power of the private key mod n. (s = h^d mod n)

## The result being:

## Signature=

## 69123596540091814582088431424764209508257104762492308295860337274111117112150153853192279706113812024677020619986756304864078156221151995376320191420930105315497430343846551290819451718974520063869903176590292363914794705964047780121193745681127501831664165304448270845661460059218771203940537562836374445261558276251009827618868869386182379681485453244484685376214832083468429299516067870115023872909796158523101277337113468216311894918702478150792029824059767514361867528147643807634287765370820932562153148224201843609830533402138325378509233342893398282438550142169774675236235802105519692796684809496253905490715573417752722909998512107040028595412825783995602026047422634472722703824112319929438379465076075177301583562506347296397060089726154670718883661745583633261252464325934395277655034763055641398480854382306888249677121525153465673139144510144836478927772571780428177846102065295830116417016510206484032573795380327045373144011903293459036406610354092849188818406483983392053146586199305252663605036774254387466531018301449705464840354803269435485362360915089421326889701773159615968358044331087619373464547296812101240803449078625229572176

## To verify that the file is signed by me, the user would have to go through the following process:

## S^e mod n

## 7748468961461525902282151250356838907585142880510257296443803337785

## Since the result is the same as the checksum of the file, the digital signature is part of the metadata of the file or is published alongside it and the public key is associated with my identity – the user can conclude that I am the author of the file and I signed it with my private key.

## Task 2:

## The haval224,4 hash of the text will be computed the same as for the previous task.

## 7748468961461525902282151250356838907585142880510257296443803337785

## The given n is

## p = 32317006071311007300153513477825163362488057133489075174588434139269806834136210002792056362640164685458556357935330816928829023080573472625273554742461245741026202527916572972862706300325263428213145766931414223654220941111348629991657478268034230553086349050635557712219187890332729569696129743856241741236237225197346402691855797767976823014625397933058015226858730761197532436467475855460715043896844940366130497697812854295958659597567051283852132784468522925504568272879113720098931873959143374175837826000278034973198552060607533234122603254684088120031105907484281003994966956119696956248629032338072839127039

## And the generator (g) is 2.

## The private key, k pr A, will be chosen via Wolfram Alpha with the following command:

## RandomInteger[2^1024]

## (given that n is 600+ digits, this number is within the limits of the constrains) which evaluates to:

## k pr A =

## 154690150718128703853980663948434841827825094646381886797656144907576954872603462042327658458152819101513789929304325496100121535541344283622424970211098613418290035877758019425937121746549699038344630564997792237195680071468092031886922658575786973578717009000969491107027120762436054920684429092093189829188

## The y, is computed using the formula:

## y = (g^(k pr A) mod p)

## And again, using the tool <https://www.dcode.fr/modular-exponentiation> that evaluates to:

## y =

## 15926911628786630121430108546080751092773942023369000664848347726021527540182710410596354037391179159111450830562289988977695627882198642662213511680396167605416620092716858778443404631927059132918814428560523688504304327970679300469924764815609026939902727385040107325024092918057393335360612006936706706885522151583609249501670907555131741023623071862862910335820676894982426898650090449766391283693965069284214829011906953264971530423460862584235445347511405081909588852808774327625778942595445178588865256581887283499652011200283014712514619294163942102014550593328833058439059728425944153875707535381143054991312

## So, we have the public key {p, g, y}.

## Now choose a random k such that 0 < k < p-1 and gcd(k, p-1) = 1

## k = 75849134584418449423219197436745188066402093667118815462772583230972532564529008739137973386026474474756529741454492508432584404529144493496969275126193889341656631159712994007459306208388159083335939277951167736810182370120385095092747801365643462981461815421920733837583425371004527132218222593037653631721

## Compute r = g^k mod p

## r = 11616752278473456278794816378047952765787583019768654833809546074044374436539369029619432826920974277888997732703206176599711801918171700351365184724439475511460952679056442530136754245489989589209769201391105092848168707723141548844351930552889324161927987995211147275424624073017977198728678914284916609107118049878786049647262900496478199756797877877833504593652641654319180853668586189721357556068366328237036510043429971434991870654767901926621933237593473362268865902492228586632275892680069613933215703421167902942096244555877423743920028249162850469900372306277799901235063610097620546120764348225814427036942

## The signature will be calculated using the formula

## S = (H(m) – xr)k^(-1) (mod p-1)

## S =

## 14334196653335106888575192002674381430894282287285510645063426989808478367681209751811265984906624973897117581617683808802557344812204906707651654645407833235456696144480254033957798773162889154859682336516457459495763315988051273485684518976722510141866398176129971723619126121048508703856520483829312890916386425492128958245042457731060449358667186894213212098560548847284901345306051404323477836656273435065733024088400093617902147411069797427676400110128540648903959692317934144644707478283869704021001327293600174983516688023596905030523619251756072420047561214506173566593872942094785104052615484564470125314725

## The pair (r, s) is the digital signature of the message m.

## Signature verification:

## The signature for the message m is (r,s) and is verified as follows:

## 0 < r < p

## 0 < s < p-1

## g^(H(m)) ≅ (y^r)(r^s) mod p

## 

## We calculate the v1 value by the formula: (y^r)(r^s) mod p and v2 by the formula: g^(H(m)) mod p.

**Conclusion:** For this laboratory work, I used hashing functions and two digital signature algorithms to sign a message. The hashing functions were used as a irreversible operation to produce a checksum which is used as the actual message in the algorithms and the 2 signing algorithms used were RSA and ElGamal. In the process, the messages were sucessfully digitally signed and also sucessfully verified, using the same hashing functions and the public keys provided. Both algorithms showed good results, but in my opinion RSA is the more practical one, as the same single set of private and public keys can be used for both encryption and digital signatures and the algorithm can be used in the same manner for other cyclic groups that are hard to reverse, example being elliptic curve encryption.