1.

***Kriptografija***

Šiuolaikinė kriptografija, tai mokslo šaka, sprendžianti elektroninės informacijos saugos problemas. Kadangi kasmet vis daugiau informacijos siunčiama elektroninėmis ryšio priemonėmis, labai svarbu užtikrinti jos saugumą, nes elektroninė informacija (toliau - informacija) dažniausiai perduodama nesaugiais kanalais, pavyzdžiui, interneto ryšiu, ir gali būti pasiekiama beveik visiems.

Terminą „kriptografija“ (angl. cryptography) sukūrė anglų fizikas ir rašytojas Tomas Braunas (Thomas Browne) 1658 m. Visuotinai ši sąvoka buvo priimta 1967 m., kai buvo atspausdinta Kahan‘o enciklopedinio pobūdžio knyga „The codebreakers“. Žodis „kriptografija“ susideda iš dviejų graikiškų žodžių: „kryptos“ reiškiančio „paslėptas“ ir „graphein“ reiškiančio „rašyti“.

***Pagrindinės sąvokos***

**Kriptografija**: mokslas, susijęs su principais ir metodais, skirtais įprastinio teksto transformavimui į neperskaitomą tekstą, o po to pakartotinai transformuodamas atgal užšifruotą tekstą į pradinį įskaitomą pavidalą.

**Tekstograma:** pradinis normalus įskaitomas tekstas.

**Šifrograma:** transformuotas (užkoduotas) tekstas.

**Šifras:** algoritmas tekstogramos transformavimui į šifrogramą.

**Raktas:** tam tikra svarbi informaciją, kurią naudoja šifras ir kurią žino tik siuntėjas ir gavėjas.

**Užšifravimas (užkodavimas):** tekstogramos transformavimo į šifrogramą procesas, naudojant šifrą ir raktą.

**Dešifravimas (dekodavimas):** šifrogramos transformavimo į tekstogramą procesas, naudojant šifrą ir raktą.

**Kriptanalizė: į**prastinio teksto (tekstogramos) transformavimo į užkoduotą tekstą (šifrogramą), nežinant rakto, principų ir metodų tyrimas. Kartais vadinama **kodo nulaužimu**.

**Kriptologija:** kriptografija ir kriptanalizė kartu (kartais suprantama tik kaip kriptografija).

**Steganografija:** teksto nuslėpimo metodai: nematomas rašalas, miniatiūrinės skylutės popieriuje, matomos tik prieš šviesą ir pan. (senovės pvz. – vergui nuskutama galva, ištatuiruojama informacija ir užauginami plaukai).

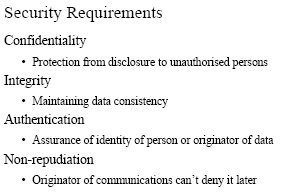
2.

***Kriptografinių sistemų saugos analizė***

( Literatūra. Eligijus Sakalauskas, Narimantas Listopadskis, Gediminas Dosinas, Kriptografijos teorija, Mokomoji knyga, Vitae Litera, 2008)

Tarkime, kad subjektas ***A*** perduoda informaciją subjektui ***B***. Vaizdumo dėlei pavadinkime juos *Aldona* ir *Broniumi*. Tarkime, *Aldona* nori perduoti *Broniui* informaciją elektroninėmis priemonėmis. Čia kyla keli svarbūs uždaviniai:

1. ***Informacijos konfidencialumas (****angl.* *data confidentiality*). Reikia užtikrinti, kad *Aldonai* perduodant slaptą informaciją *Broniui*, jos negalėtų perskaityti kitas subjektas, kuriam prieinamas jų naudojamas ryšio kanalas. Tai pasiekiama informaciją šifruojant. *Aldona*, naudodama kokį nors šifravimo algoritmą, užšifruoja informaciją taip, kad tik *Bronius* galėtų ją perskaityti, o kitiems ji būtų tik beprasmis simbolių rinkinys.
2. ***Informacijos vientisumas*** (*angl.* *data integrity*). Reikia užtikrinti, kad duomenų, kuriuos *Aldona* siunčia *Broniui*, būtų neįmanoma suklastoti ar pakeisti. Dažniausiai tam pasitelkiamos specialios santraukos funkcijos (*angl. hash function*), kurios iš turimų duomenų sugeneruoja duomenis atitinkančią santraukos reikšmę (*angl. hash code*). *Aldona*, siųsdama pranešimą *Broniui*, gali prie jo prisegti santraukos reikšmę. Jei Bronius, gavęs pranešimą ir jam pritaikęs tą pačią santraukos funkciją, gauna skirtingą reikšmę, galima užtikrintai tvirtinti, kad duomenys pakeliui buvo pakeisti.
3. ***Subjekto autentifikavimas*** (*angl. subject authentification*). *Broniui*, gavusiam informaciją, turi būti sudaryta galimybė įsitikinti, kad duomenų siuntėjas tikrai yra *Aldona*, o ne koks nors apsimetėlis. Vienas autentifikavimo metodų – elektroninis parašas.
4. ***Non-repudiation.***

****

3.

Įsibrovėliai gali būti dviejų tipų:

1. ***Pasyvieji***. Tinkle tokie įsibrovėliai nėra aktyvūs – jie tik stebi informacijos mainus tarp *Aldonos* ir *Broniaus*. Svarbiausias jų tikslas – pažeisti informacijos konfidencialumą.
2. ***Aktyvieji***. Tokie įsibrovėliai ne tik stebi informacijos srautą, bet ir patys gali įsiterpti į jį, suklastoti, sugadinti ar perimti duomenų paketus, apsimesti legaliais kriptografinės sistemos vartotojais. Bendrąja prasme tokie įsibrovėlių veiksmai vadinami apsimetimo ataka (*angl. impersonation attack*).

4.

**Pakeitimas**

***Pakeitimas***, tai toks šifravimo būdas, kai tekstogramos alfabeto simboliai pakeičiami kitais to paties arba kito alfabeto simboliais.

Vienas iš seniausių tokio šifravimo pavyzdžių yra Julijaus Cezario šifras. Tai toks šifras, kai lotynų abėcėlės raidė, kurios numeris alfabete yra *m*, keičiama to paties alfabeto raide, kurios numeris *c* ir kuris apskaičiuojamas pagal formule

,

t.y., raidė, pavyzdžiui ***a****,* keičiama kita raide, kuri nutolusi nuo jos per tris pozicijas, tai yra raide ***D***.

***Pastaba.*** Paprastai tekstogramos alfabetas žymimas mažosiomis raidėmis, o šifrogramos alfabetas – didžiosiomis raidėmis.

Aišku, kad Cezario šifro raktas yra trys. Vadinasi, norint iššifruoti šifrogramą, yra taikoma tokia formulė: .



******

******



5.

******

6.

***Vernam cipher***over the binary alphabet is defined by

ci = mi\_ki for i = 1; 2; 3 : : : ; where m1;m2;m3; : : : are the plaintext digits, k1; k2; k3; : : : (the *keystream*) are the key digits, c1; c2; c3; : : : are the ciphertext digits, and \_ is the XOR function (bitwise addition modulo 2). Decryption is defined by mi = ci\_ki. If the keystream digits are generated independently and randomly, the Vernam cipher is called a *one-time pad*, and is unconditionally secure (x1.13.3(i)) against a ciphertext-only attack. More precisely, if M, C, and K are random variables respectively denoting the plaintext, ciphertext, and secret key, and if H() denotes the entropy function (Definition 2.39), then H(MjC) = H(M). Equivalently, I(M;C) = 0 (see Definition 2.45): the ciphertext contributes no information about the plaintext.

**(*Vigen`ere cipher*)** Let A = fA;B;C; : : : ;X;Y; Zg and t = 3. Choose

e =(p1; p2; p3), where p1 maps each letter to the letter three positions to its right in the alphabet, p2 to the one seven positions to its right, and p3 ten positions to its right. If

**m = THI SCI PHE RIS CER TAI NLY NOT SEC URE**

**then c = Ee(m) = WOS VJS SOO UPC FLBWHS QSI QVD VLM XYO**: \_

Polyalphabetic ciphers have the advantage over simple substitution ciphers that symbol

frequencies are not preserved. In the example above, the letter E is encrypted to both O and L. However, polyalphabetic ciphers are not significantly more difficult to cryptanalyze, the approach being similar to the simple substitution cipher. In fact, once the block length t is determined, the ciphertext letters can be divided into t groups (where group i, 1 \_ i \_ t, consists of those ciphertext letters derived using permutation pi), and a frequency analysis can be done on each group.

******

7.

***Simetrinė kriptosistema.***

Naudojantis bendruoju slaptuoju raktu *k*, galimame tik šifruoti duomenis, bet ir atlikti kitas kriptografines funkcijas, pvz., skaičiuoti santraukos funkcijas, kurti elektroninio parašo schemas, generuoti pseudoatsitiktinius skaičius ir t.t. Tuo remdamiesi galime apibrėžti simetrinę kriptosistemą.

***Apibrėžimas.*** Simetrine kriptosistema vadinama tokia sistema, kuri visoms kriptografinėms funkcijoms atlikti naudoja vieną slaptąjį raktą *k*, žinoma abiem ( arba keliems) protokolo dalyviams.

Nors simetrinės kriptosistemos turi nemažai privalumų, tačiau turi ir viena esminį trūkumą. Tarkime, kad kriptosistemoje yra *N* kriptosubjektų. Norint išsaugoti kriptosubjektų tarpusavio konfidencialumą, jiems reikia paskirstyti raktų porų. Kiekvienas jų turi akylai saugoti  raktą ryšiams su kitais kriptosubjektais. Nesunku įsitikinti, kad, didėjant kriptosubjektų skaičiui, reikalingų raktų porų skaičius didėja kvadratine priklausomybe. Išvengti šio trūkumo padeda viešojo rakto kriptosistema.

Simetrinė kriptosistema buvo naudojama nuo seniausių laikų, todėl galima pateikti labai daug tokios kriptosistemos pavyzdžių.

Šioje kriptosistemoje dažniausiai naudojami du principai: ***pakeitimas*** (*angl. substitution*) ir ***perstatymas*** (*angl. trasposition).*

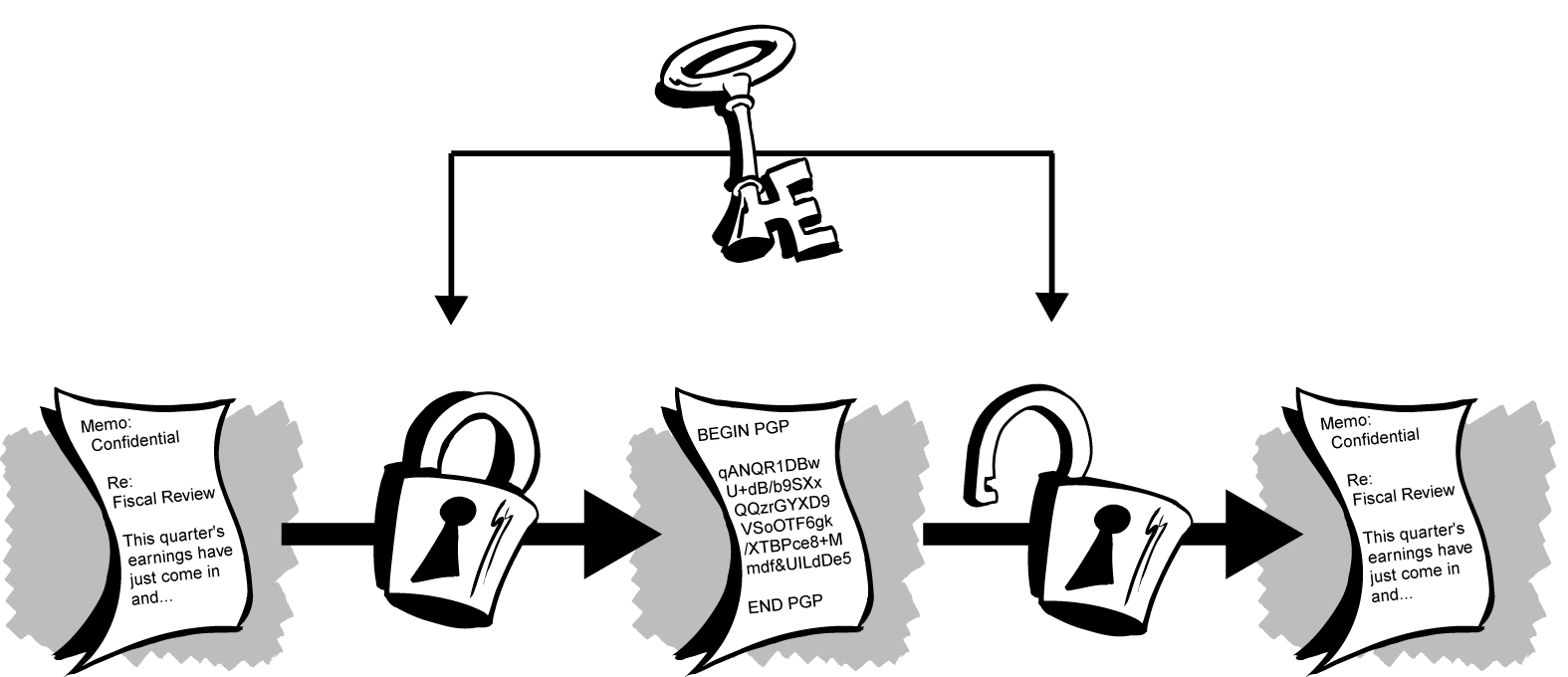
***Viešojo rakto (asimetrinė) kriptosistema (VRK).***

***Apibrėžimas.*** Viešojo rakto (asimetrine) kriptosisema vadinama tokis sistema, kurioje kriptografinės funkcijos realizuojamos naudojant matematiškai susijusių raktų porą: privatųjį raktą – *PR* (*angl. private key*), žinoma tik jo savininkui, ir su juo susijusį viešąjį raktą – *VR* (*angl. public key*), žinoma visiems vartotojams tinkle. Ši raktų pora vadinama asimetriniais raktais.

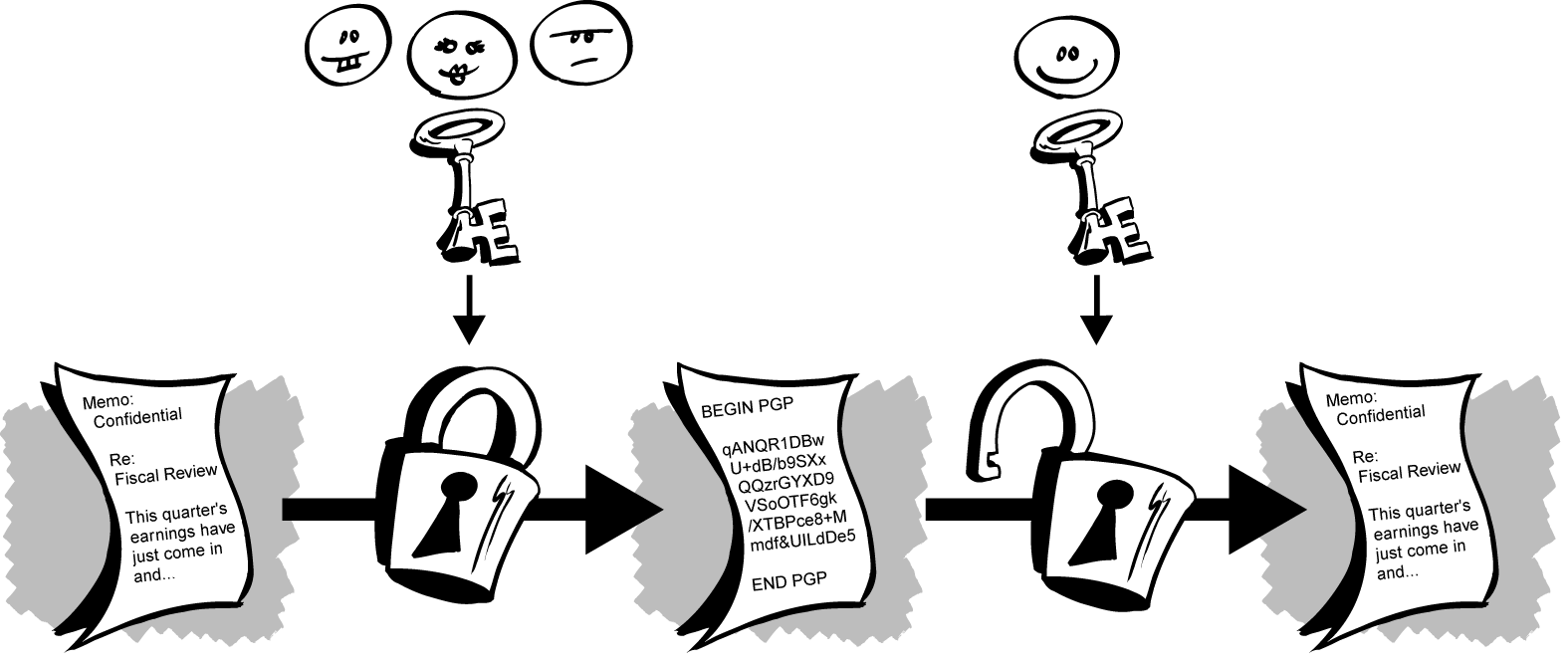
*VRK* leidžia spręsti asimetrinio šifravimo, elektroninio parašo, raktų keitimosi ir kitus svarbius šiuolaikinei informacinei visuomenei uždavinius. Jau šiuo metu kai kuriose valstybėse (tarp jų ir Estijoje) naudojamos ir veikia elektroninio balsavimo sistemos. 2007 metais *EuroCrypt* konferencijoje buvo paskelbtas straipsnis, kuriame siūloma praktiškai įgyvendinama ir pakankamai patogi elektroninių pinigų sistema.

Kol kas apžvelkime *VRK* taikymo būdus duomenims šifruoti ir elektroniniam parašui realizuoti. Tarkime, *Aldona* turi asimetrinių raktų porą  , o *Bronius* – atitinkamai . Iš *VRK* apibrėžimo seka, yra privatieji raktai, žinomi tik jų savininkams. Tuo tarpu  yra viešieji raktai, kuriuos žino visi dalyviai.

8.



9.



Asimetrinė raktų sistema

Asymmetric algorithms use pairs of keys. One is used for encryption and the other one for decryption. The decryption key is typically kept secretly, therefore called ``**private key**'' or ``secret key'', while the encryption key is spread to all who might want to send encrypted messages, therefore called ``**public key**''. The idea of asymmetric algorithms was first published 1976 by Diffie and Hellmann.

10.

11.

Asymmetric ciphers vs symmetric ciphers (pros & cons)

* GREITAVEIKA. A disadvantage of asymmetric ciphers over symmetric ciphers is that they tend to be about **1000 times slower**.
* KANALO SAUGUMAS. An important advantage of asymmetric ciphers over symmetric ciphers is that no secret channel is necessary for the exchange of the public key.
* NULAUŽIMAS. Another disadvantage is that symmetric ciphers can be cracked through a "**brute-force**" attack.

12.

**Maišos funkcija** ([angl.](http://lt.wikipedia.org/wiki/Angl%C5%B3_kalba) *hash function*) – tai [funkcija](http://lt.wikipedia.org/w/index.php?title=Funkcija_%28programavimas%29&action=edit&redlink=1" \o "Funkcija (programavimas) (puslapis neegzistuoja)) *h = h(m)*, priskirianti argumentui *m* pseudoatsitiktinį skaičių, vadinamą *maišos kodu*. Tam pačiam argumentui funkcija visada turi duoti tokį patį rezultatą, taigi ji nėra atsitiktinė. Dažniausiai funkcijos [reikšmių sritis](http://lt.wikipedia.org/w/index.php?title=Reik%C5%A1mi%C5%B3_sritis&action=edit&redlink=1" \o "Reikšmių sritis (puslapis neegzistuoja)) yra, palyginti su [apibrėžimo sritimi](http://lt.wikipedia.org/w/index.php?title=Apibr%C4%97%C5%BEimo_sritis&action=edit&redlink=1" \o "Apibrėžimo sritis (puslapis neegzistuoja)), nedidelė.

Pagrindinės maišos funkcijos savybės:

* apibrėžimo sritis didelė palyginti su fiksuoto dydžio reikšmių sritimi;
* funkciją nesunku apskaičiuoti bet kokiam argumentui;
* daug kartų kviečiant funkciją su įvairiais argumentais, jos rezultatai pasiskirsto tolygiai;
* pagal rezultatą negalima vienareikšmiškai nustatyti argumento;
* labai sunku surasti du argumentus m1 ir m2 tokius, kad *h(m1) = h(m2)* (svarbu kriptografijoje).

Maišos funkcijos naudojamos:

* kai kuriose [duomenų struktūrose](http://lt.wikipedia.org/wiki/Duomen%C5%B3_strukt%C5%ABra) rikiuojant duomenis.
* duomenų bazėse ([maišos jungtyse](http://lt.wikipedia.org/wiki/Mai%C5%A1os_jungtis" \o "Maišos jungtis)).
* įvairioje programinėje įrangoje – patikrinimui, ar tam tikri duomenys (pavyzdžiui, failas) nepakeisti sistemos išorėje.
* [kriptografijoje](http://lt.wikipedia.org/wiki/Kriptografija)
* rikiavimo algoritmuose (pavyzdžiui, sujungiant [skaitmeninį rikiavimą](http://lt.wikipedia.org/wiki/Skaitmeninis_rikiavimo_algoritmas) su kuriuo nors kitu)

Jei funkcijos apibrėžimo sritis yra [objektas](http://lt.wikipedia.org/wiki/Objektas_%28programavimas%29), kaip maišos funkcija neretai vartojamas fizinis to objekto adresas kompiuterio atmintyje. Tuomet tariama, jog kiekvienas objektas gali būti lygus tik sau pačiam. Jei pagal programos algoritmą du skirtingą padėtį atmintyje užimantys objektai gali būti lygūs tarpusavyje, būtina iš naujo apibrėžti ne tik palyginimo operaciją, bet tos klasės objektams taikomą maišos funkciją, pakeičiant du sutartus, kiekvieno objekto turimus [virtualius metodus](http://lt.wikipedia.org/wiki/Metodas_%28programavimas%29) (pavyzdžiui, *equals* ir *hashCode* Java kalboje).

13.

# Message authentication code

In [cryptography](http://dictionary.sensagent.com/Cryptography/en-en/), a **message authentication code** (often **MAC**) is a short piece of information used to [authenticate](http://dictionary.sensagent.com/Authenticate/en-en/) a [message](http://dictionary.sensagent.com/Message/en-en/).

A MAC [algorithm](http://dictionary.sensagent.com/Algorithm/en-en/), sometimes called a **keyed** (**cryptographic**) **hash function**, accepts as input a [secret key](http://dictionary.sensagent.com/Secret_key/en-en/) and an arbitrary-length message to be authenticated, and outputs a MAC (sometimes known as a *tag*). The MAC value protects both a message's [data integrity](http://dictionary.sensagent.com/Data_integrity/en-en/) as well as its [authenticity](http://dictionary.sensagent.com/Authentication/en-en/), by allowing verifiers (who also possess the secret key) to detect any changes to the message content.

The term **message integrity code (MIC)** is frequently substituted for the term MAC, especially in communications[[1]](http://dictionary.sensagent.com/message+authentication+codes/en-en/#cite_note-0), where the acronym MAC traditionally stands for [Media Access Control](http://dictionary.sensagent.com/Media_Access_Control/en-en/). However, some authors[[2]](http://dictionary.sensagent.com/message+authentication+codes/en-en/#cite_note-1) use MIC as a distinctly different term from a MAC; in their usage of the term the MIC operation does not use secret keys. This lack of security means that any MIC intended for use gauging message integrity should be encrypted or otherwise be protected against tampering. MIC algorithms are created such that a given message will always produce the same MIC assuming the same algorithm is used to generate both. Conversely, MAC algorithms are designed to produce matching MACs only if the same message, secret key and [initialization vector](http://dictionary.sensagent.com/Initialization_vector/en-en/) are input to the same algorithm. MICs do not use secret keys and, when taken on their own, are therefore a much less reliable gauge of message integrity than MACs. Because MACs use secret keys, they do not necessarily need to be encrypted to provide the same level of assurance.

While MAC functions are similar to [cryptographic hash functions](http://dictionary.sensagent.com/Cryptographic_hash_function/en-en/), they possess different security requirements. To be considered secure, a MAC function must resist [existential forgery](http://dictionary.sensagent.com/Existential_forgery/en-en/) under [chosen-plaintext attacks](http://dictionary.sensagent.com/Chosen-plaintext_attack/en-en/). This means that even if an attacker has access to an [oracle](http://dictionary.sensagent.com/Oracle_machine/en-en/) which possesses the secret key and generates MACs for messages of the attacker's choosing, the attacker cannot guess the MAC for other messages without performing unfeasible amounts of computation.

MACs differ from [digital signatures](http://dictionary.sensagent.com/Digital_signature/en-en/) as MAC values are both generated and verified using the same secret key. This implies that the sender and receiver of a message must agree on the same key before initiating communications, as is the case with [symmetric encryption](http://dictionary.sensagent.com/Symmetric_encryption/en-en/). For the same reason, MACs do not provide the property of [non-repudiation](http://dictionary.sensagent.com/Non-repudiation/en-en/) offered by signatures: any user who can verify a MAC is also capable of generating MACs for other messages. In contrast, a digital signature is generated using the private key of a key pair, which is [asymmetric encryption](http://dictionary.sensagent.com/Asymmetric_encryption/en-en/). Since this private key is only accessible to its holder, a digital signature proves that a document was signed by none other than that holder. Thus, digital signatures do offer non-repudiation.

MAC algorithms can be constructed from other cryptographic primitives, such as [cryptographic hash functions](http://dictionary.sensagent.com/Cryptographic_hash_function/en-en/) (as in the case of [HMAC](http://dictionary.sensagent.com/HMAC/en-en/)) or from [block cipher](http://dictionary.sensagent.com/Block_cipher/en-en/) algorithms ([OMAC](http://dictionary.sensagent.com/OMAC_%28cryptography%29/en-en/), [CBC-MAC](http://dictionary.sensagent.com/CBC-MAC/en-en/) and [PMAC](http://dictionary.sensagent.com/PMAC_%28cryptography%29/en-en/)). However many of the fastest MAC algorithms are constructed based on [universal hashing](http://dictionary.sensagent.com/Universal_hashing/en-en/).

14.

***Apibrėžimas.*** Elektroniniu parašu (e. parašu) vadinamas asimetrinis šifravimas, kai duomenys užšifruojami privačiuoju subjekto raktu, o iššifruojami – viršuoju raktu.

15.

***Pirminiai skaičiai.***

***Apibrėžimas.*** *Natūralusis skaičius vadinamas pirminiu, jei jis dalosi tik iš vieneto ir pats iš savęs; priešingu atveju šis skaičius vadinamas sudėtiniu.*

Pavyzdžiui, skaičiai 2, 3, 5, 7, 11, 13, 17 yra pirminiai , o skaičiai 4, 6, 8,10,12 yra sudėtiniai.

***Euklido teorema.*** *Egzistuoja be galo daug pirminių skaičių*.

Ši teorema įrodoma prieštaravimo būdu. Tarkime, kad pirminių skaičių aibė *P* yra baigtinė: . Panagrinėkime skaičių: . Šis skaičius nesutampa nei su vienu iš skaičių . Be to, jis dalosi tik iš vieneto ir pats iš savęs, todėl yra pirminis. Vadinasi, prielaida, kad pirminių skaičių aibė yra baigtinė, - klaidinga.

Kaip rasti visus pirminius skaičius, nedidesnius už *n*. Ši uždavinį išsprendė senovės Aleksandrijos matematikas Eratostenas, gyvenęs 284 – 204 metais prieš Kristų.

***Eratosteno rėtis.***

Eratosteno pasiūlytas pirminių skaičių apskaičiavimo metodas yra toks.

1. Išrašykime visų skaičių nuo 1 iki *n* seką. Pradėkime nuo pirminio skaičiaus 2.
2. Sekoje pabraukime kas antrą skaičių nuo 2 (skaičių 2 nepabraukiame), t.y. pabraukiame lyginius skaičius 4, 6, 8, 10 ir t.t. Pirmasis nepabrauktas skaičius bus 3. Jis yra pirminis, nes nesidalo iš 2.
3. Sekoje pabraukime kas trečią skaičių, skaičiuodami nuo 3 (skaičiaus 3 nepabraukiame), t.y. pabraukiame skaičius 6, 9, 12, 15 ir t.t. Dabar pirmasis nepabrauktas skaičius yra 5. Jis yra pirminis, nes nesidalo nei iš 2, nei iš 3.
4. Skaičiaus 5 nepabraukime, o kas penktą skaičių, skaičiuojant nuo 5, - pabraukime, t.y. pabraukime skaičius 10, 15, 20, 25 ir t.t. Dabar mažiausias nepabrauktas skaičius bus 7. Jis yra pirminis, nes nesidalija nė iš vieno mažesnio už jį pirminio skaičiaus 2, 3, 5.

Šį procesą baigsime, kai nebus galima pabraukti naujų skaičių.

Aišku, kad pats didžiausias pirminis skaičius tarp skaičių nuo 1 iki *n*, yra nedidesnis už  (nes ).

***Merseno pirminiai skaičiai***

Keletą šimtmečių buvo vaikomasi pirminių skaičių. Daugelis matematikų siekė šlovės, stengdamiesi rasti didesnį už tuo metu žinomus pirminius skaičius. Dabar toji pirminių skaičių medžioklė aprimo, ji tęsiama viena kryptimi, kur ir sulaukta sėkmės.

***Merseno pirminiai skaičiai*** – tai savito pavidalo pirminiai skaičiai

**, čia p – pirminis skaičius.**

Nagrinėjami skaičiai taip pavadinti prancūzų vienuolio Mereno Merseno (1588 – 1648) garbei.

Jei pradėsime rašyti skaičius , atitinkančius skirtingus pirminius skaičius *p*, tai pamatysime, kad ne visi jie yra pirminiai.

Pavyzdžiui,

 - pirminis,

 - pirminis,

- pirminis,

 - pirminis,

.

Bendruoju atveju, norint rasti didelius pirminius Merseno skaičius, reikia tikrinti visus skaičius , atitinkančius skirtingus pirminius skaičius *p*. Tie skaičiai labai sparčiai didėja, todėl taip pat sparčiai didėja ir jiems rasti reikalingos darbo sanaudos.

Merseno skaičių tyrimo istorijoje galima išskirti ankstyvąjį laikotarpį, kurio kulminacija buvo 1750 m. Tada Leonardas Oileris nustatė, kad skaičius  yra pirminis. Iki tol jau buvo rasti septyni Merseno pirminiai skaičiai, atitinkantys

, , , , , , .

Daugiau kaip šimtą metų Oilerio skaičius  buvo didžiausias iš žinomų pirminių skaičių. 1876 m prancūzų matematikas Lukasas nustatė, kad milžiniškas skaičius  yra pirminis. Šį skaičių sudaro 39 skaitmenys. Už pastarąjį mažesni Perseno pirminiai skaičiai atitinka, be anksčiau nurodytų *p* reikšmių, dar šias:

, , .

Tie 12 Merseno pirminių skaičių buvo rasti naudojantis tik pieštuku ir popieriumi. Sukūrus kompiuterius, naujų Merseno pirminių skaičių buvo ieškoma su jais.

Didžiausias (žodis „didžiausias“ yra sąlyginis) Merseno pirminis skaičius, rastas Ilinojaus universitete, yra . Šie skaičiai yra milžiniški. Norėdami rasti skaičiaus  skaitmenų skaičių, vietoje jo imkime skaičių . Šie du skaičiai turi tiek pat skaitmenų: tarus, kad turi vienu skaitmeniu daugiau, tai reikštų, kad skaičius  paskutinis skaitmuo yra 9. O to negali būti. Tam, kad rastume skaičius  skaitmenų skaičių, logaritmuokime: . Vadinasi, skaičius  skaitmenų skaičius *n* yra lygus , čia simbolis““ reiškia sveikąją dalį link plius begalybės. Matlabas turi funkcija „***ceil***“.

Pavyzdžiui,

n=ceil(11213\*log10(2)),

n = 3376.

Tai reiškia, kad didžiausias žinomas Merseno pirminis skaičius turi 3376 skaitmenis.

16.

***Pagrindinė aritmetikos teorema (The fundamental theorem of arithmetic) .***

*Bet kuris sveikasis skaičius a, didesnis už vienetą, užrašomas pirminių skaičių, įskaitant jų laipsnį, sandauga:  ; čia ,  - pirminiai skaičiai,  - natūralieji skaičiai. Šis užrašas yra vienintelis, jeigu nekreiptume dėmesio į daugiklių išsidėstymo tvarką.*

Pavyzdžiui, , .