Autentizácia používateľov a autorizácia dát

- základný princíp a pojmy
- autentizačný kód správy (MAC)
- optimálne poradie šifrovania a MAC
- hašovaný autentizačný kód správy HMAC
- príklad využitia autorizácie dát v GCM móde pri zabezpečení TCP/IP komunikácie
- princíp autentizácie užívateľov s využitím hašovania v operačných systémoch

Primárny zdroj informácii k dnešnej prednáške:

[1] prof. Ing. Dušan Levický, CSc.
APLIKOVANÁ KRYPTOGRAFIA
od utajenia správ ku kybernetickej bezpečnosti
Elfa, Košice, 2018 (str.176-186)

[2] Paar, Ch., Pelzl, J.: Understanding Cryptography. Springer 2010, (http://www.crypto-textbook.com/)

[3] Drutarovský, M.: Kryptografia pre vstavané procesorové systémy. Technická univerzita v Košiciach, 2017.
Digitalizovaná verzia voľne dostupná v TUKE knižnici - http://ebooks.lib.tuke.sk/login
Doplňujúce materiály a zdrojové kódy k učebnici - http://aplikovanakryptografia.fei.tuke.sk/

+ ďalšie linky uvedené v texte

Základný princíp a pojmy

Kryptografia s verejným kľúčom umožnila vyriešiť problém bezpečnej komunikácie a bezpečnej distribúcie kľúčov. Zároveň však priniesla problém autorizácie používateľov a autorizácie dát, ktorý patrí k najzávažnejším problémom sieťovej bezpečnosti.

Autentizáciu používateľa možno charakterizovať ako proces jeho identifikácie a verifikácie. Autentizácia používateľa má v kryptografii zásadný význam najmä s ohľadom na potenciálne útoky ako sú predstieranie identity, resp. podvrhnutie verejného kľúča.

V informačných systémoch možno autentizáciu používateľov realizovať v zásade troma prístupmi, a to na základe:

- **znalosti** dohodnutej tajnej informácie (heslo)
- **vlastníctva** určitého typu identifikačného prostriedku (čipová karta, súkromný kľúč)
- **biometrických** parametrov (otlačok dlane, prsta, očná dúhovka, reč,...).

V kryptografii s verejným kľúčom je autentizácia používateľa založená na vlastníctve súkromného kľúča.

Autorizácia dát je proces identifikácie zdroja, resp. pôvodu dát, ktorý má zároveň zabezpečiť **integritu dát**, *t.* j. ochranu pred ich modifikáciou. Modifikácia dát môže zahŕňať modifikáciu ich obsahu, resp. usporiadania a modifikáciu časových relácií (oneskorenie, spätné presmerovanie dát). Autorizácia dát v kryptografii s verejným kľúčom sa realizuje **digitálnym podpisom** (digital signature).

Autentizácia používateľa a autorizácia dát sa niekedy označuje ako **autentizácia správy** (message authentication) a táto autentizácia môže prebiehať na dvoch úrovniach.

Nižšia úroveň autentizácie správy zahŕňa aplikáciu **autentizačnej funkcie** (authentication function), ktorá na základe danej správy produkuje určitú hodnotu, označovanú ako **autentifikátor** (authentificator).

Vyššia úroveň zahŕňa generovanie autentizačného protokolu, v ktorom sa využíva autentifikátor a ktorý umožňuje realizovať samotnú autentizáciu správy.

Autentizačné funkcie možno rozdeliť do troch skupín. Sú to:

- šifrovanie správy (message encryption)
- autentizačný kód správy (MAC – Message Authentication Code)
- hašovacie funkcie (hash functions).

Šifrovanie správy možno chápať ako funkciu, ktorá produkuje zašifrovaný text a tento slúži ako autentifikátor. Argument tejto autentizačnej funkcie je celá správa, resp. celý otvorený text.

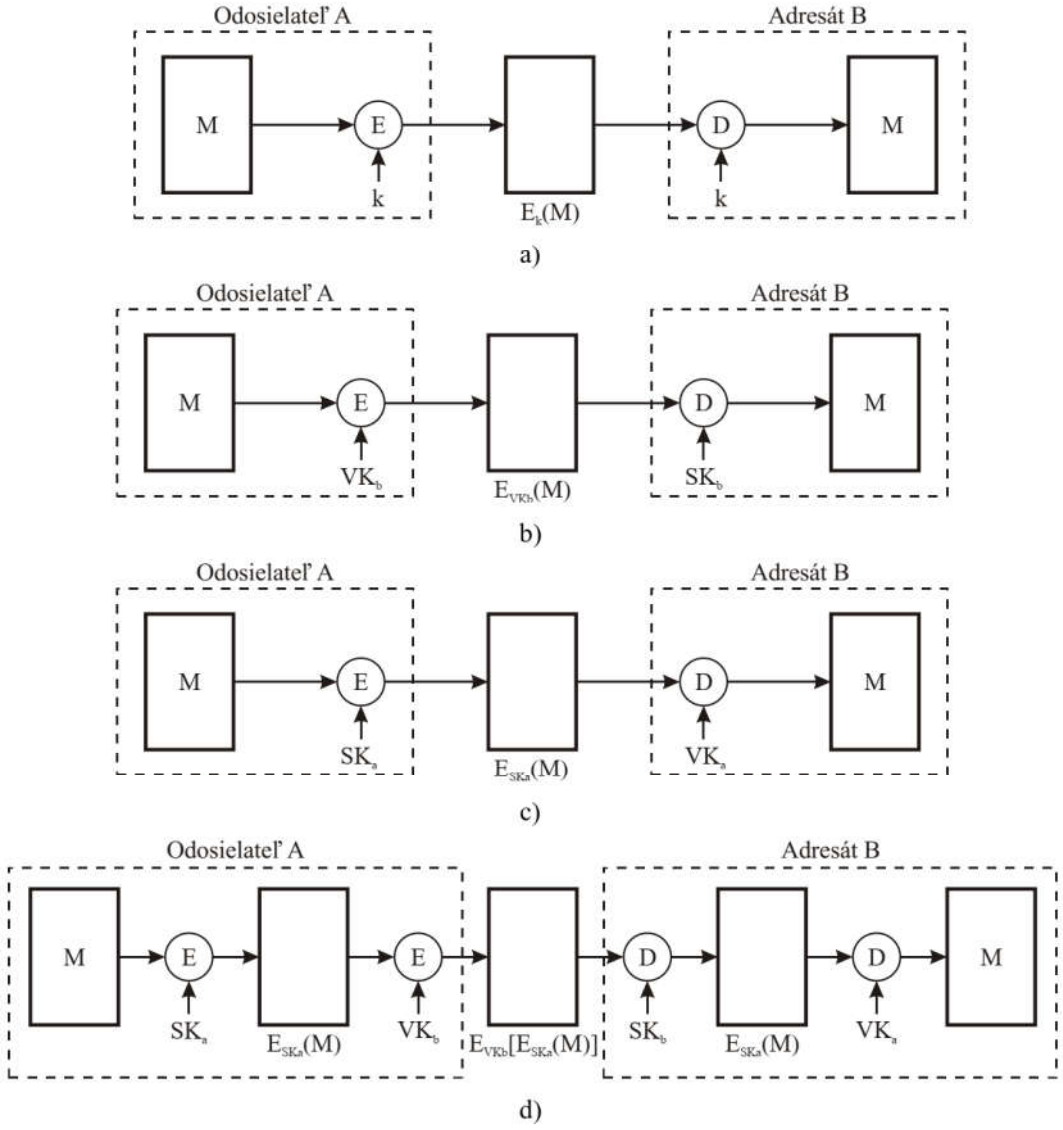
Autentizačný kód správy je funkcia, ktorá pomocou tajného kľúča produkuje výstupnú hodnotu vyjadrenú fixným počtom bitov, ktorý slúži ako autentifikátor. Uvedená funkcia závisí od správy a tajného kľúča.

Hašovacia funkcia je funkcia, ktorá transformuje správu s ľubovoľnou dĺžkou na výstupnú hodnotu vyjadrenú fixným počtom bitov bez použitia kľúča. Výstupná hodnota označovaná ako hašovací kód slúži ako autentifikátor.

9.1 Šifrovanie správy

Šifrovanie možno realizovať v zásade dvoma spôsobmi a to:

- šifrovanie s tajným kľúčom (symetrické šifrovanie)
- šifrovanie s verejným kľúčom (asymetrické šifrovanie).



Obr. 9.1 Šifrovanie a dešifrovanie správy
a) Symetrické šifrovanie: utajenie a autentizácia
b) Asymetrické šifrovanie: utajenie
c) Asymetrické šifrovanie: autentizácia a podpis
d) Asymetrické šifrovanie: utajenie, autentizácia a podpis

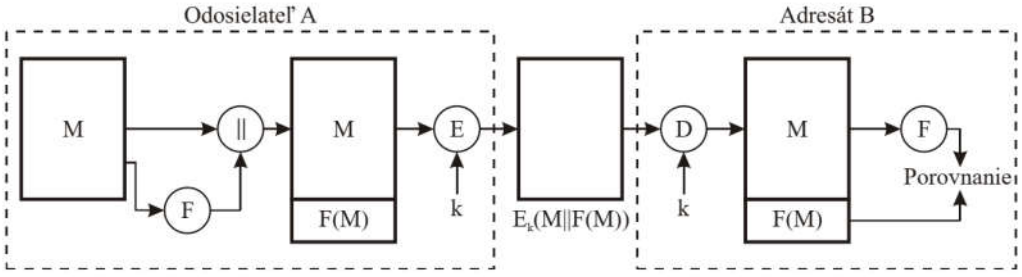
Šifrovanie s tajným kľúčom, resp. symetrické šifrovanie predpokladá použitie rovnakého tajného kľúča na šifrovanie aj dešifrovanie správy *M (Obr. 9.1a)*. Odosielateľ A zašifruje správu *M* kľúčom *k* a adresát B ju dešifruje rovnakým kľúčom *k*. Ak kľúč *k* nie je známy ďalšiemu subjektu, potom je zabezpečené utajenie obsahu správy *M*. Naviac možno zároveň predpokladať, že ak adresát B dešifroval správu *M* kľúčom *k*, potom správa *M* pochádza od odosielateľa A. Šifrovanie s tajným kľúčom teda zabezpečuje utajenie obsahu správy A a autentizáciu odosielateľa A.

Uvedený predpoklad je potrebné však upresniť najmä z pohľadu dešifrovania šifrovanej správy adresátom B. Dešifrovanie používa dešifrovaciu funkciu D_k závislú od kľúča *k*, teda dešifrovanie je funkcia $Y = D_k(X)$, kde vstup *X* je argumentom funkcie D_k a *Y* je hodnota funkcie. Ak *X* je zašifrovaný text zodpovedajúci skutočnej (originálnej) správe *M*, získaný šifrovacou funkciou E_k, potom *Y* je otvorený text zodpovedajúci správe *M*. Ak však predpokladáme, že funkcia D_k akceptuje akýkoľvek vstup *X*, potom vzniká problém, aké kritérium možno použiť na to, aby sa výstup *Y* považoval za otvorený text, ktorý zodpovedá originálnej správe *M*. Je zrejmé, že ak správa *M* je zmysluplná, potom aj dešifrovaný otvorený text, teda výstup *Y* by mal byť tiež zmysluplný, resp. dešifrovaný otvorený text by mal vykazovať znaky jazyka otvoreného textu.

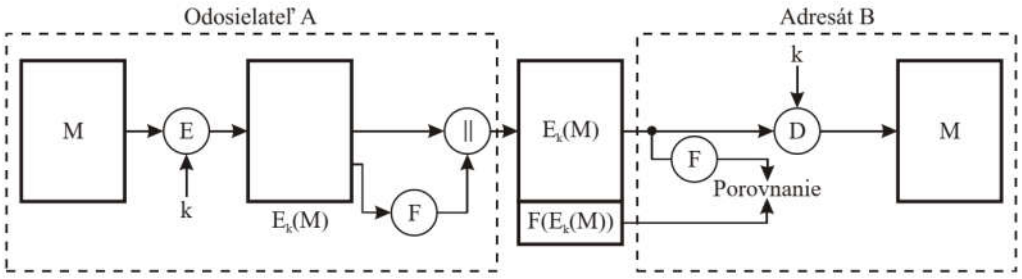
Originálnu správu *M* však môže tvoriť ľubovoľná postupnosť symbolov, resp. bitov (dátové súbory), teda *X* môže byť tiež ľubovoľný argument a potom výstup dešifrovacej funkcie $Y = D_k(X)$ musí byť akceptovaný ako autentický otvorený text. Kritérium zmysluplnosti dešifrovaného otvoreného textu nemožno použiť vždy, resp. má značné obmedzenia. V uvedenom prípade vzniká nebezpečie útoku, ktorý spočíva v možnosti podvrhu falošných správ. Okrem toho nepovolaná osoba môže narušiť integritu prenášanej správy. Integritu správy môžu narušiť aj šumy a náhodné chyby.

Uvedené problémy možno riešiť metódou vytvárania **kryptografického kontrolného súčtu** (cryptographic checksum), ktorá je analogická metóde korekčných kódov. Rozdiel medzi oboma metódami však spočíva v tom, že spôsob vytvorenia kryptografického súčtu nie je verejne známy.

Princíp uvedenej metódy je uvedený na *Obr. 9.2a*.



a)



b)

Obr. 9.2 Kryptografický kontrolný súčet: a) interný, b) externý

Z originálnej správy M sa funkciou F vytvorí kryptografický kontrolný súčet $F(M)$, ktorý sa pridá k správe M, čo možno vyjadriť symbolickým zápisom $M||F(M)$. Tento súbor sa zašifruje kľúčom k, čím sa získa $E_k\left(M \,||\, F\left(M\right)\right)$. Na prijímacej strane sa dešifrovaním získa súbor $M||F(M)$. Z prijatej správy M sa opäť vypočíta $F(M)$ a porovná s prijatým $F(M)$. V prípade zhody možno predpokladať, že správa M nebola počas prenosu modifikovaná, resp. nebola narušená jej integrita.

Opísaný spôsob testovania integrity správy sa označuje ako **interný**.

Externý spôsob testovania je uvedený na *Obr. 9.2b*, v ktorom sa najprv správa M zašifruje kľúčom k a kryptografický kontrolný súčet sa vytvára zo zašifrovanej správy $E_k(M)$. Pri autentizácii sa využíva interný spôsob vytvárania a testovania kryptografického kontrolného súčtu, pretože jeho spôsob generovania nie je verejne známy a teda nepovolaná osoba nie je schopná generovať falošnú správu so správnou kontrolou kryptografického kontrolného súčtu.

Šifrovanie s verejným kľúčom poskytuje možnosť realizácie viacerých funkcií.

Na *Obr. 9.1b* je znázornený spôsob šifrovania, ktorý na šifrovanie využíva verejný kľúč VK_b a na dešifrovanie sa využíva súkromný kľúč SK_b. Uvedený spôsob realizácie asymetrického šifrovania zabezpečuje utajenie obsahu správy M, pretože správu môže dešifrovať iba adresát B, ktorý vlastní SK_b. Uvedený spôsob však nezabezpečuje autentizáciu A, pretože verejný kľúč VK_b môže využiť akýkoľvek subjekt, resp. nepovolaná osoba.

Princíp asymetrického šifrovania, ktoré zabezpečuje autentizáciu je uvedený na *Obr. 9.1c*. Na šifrovanie správy M sa používa súkromný kľúč A, teda SK_a na dešifrovanie verejný kľúč A, t. j. VK_a.

Keďže dešifrovanie možno realizovať iba VK_a, správa M bola zašifrovaná súkromným kľúčom SK_a. Je potrebné však opäť zabezpečiť, aby správa M bola upravená tak, aby bolo možné znemožniť podvrhnutie falošnej správy, resp. doplniť správu kryptografickým kontrolným súčtom.

Z faktu, že správa bola zašifrovaná subjektom A, ktorý vlastní SK_a vyplýva, že uvedený spôsob šifrovania plní aj funkciu digitálneho podpisu, aj keď je potrebné poznamenať, že ide iba o princíp, lebo digitálne podpisy sa vytvárajú odlišným spôsobom.

Uvedený spôsob šifrovania však nezabezpečuje utajenie obsahu správy M, lebo verejný kľúč VK_a je verejne dostupný a dešifrovanie môže realizovať akýkoľvek subjekt.

Zabezpečenie utajenia správy a autentizácie zdroja správy M zabezpečuje asymetrické šifrovanie, ktorého princíp je uvedený na *Obr. 9.1d*.

Správa M sa najprv zašifruje súkromným kľúčom SK_s, potom sa takto získaný produkt zašifruje verejným kľúčom adresáta B, t. j. VK_b. Šifrovanie kľúčom SK_b zabezpečuje autentizáciu A, šifrovanie kľúčom VK_b realizuje utajenie obsahu správy M. Dešifrovanie sa realizuje v dvoch krokoch. Najprv sa prijatý zašifrovaný text dešifruje súkromným kľúčom SK_b a potom verejným kľúčom VK_a.

Nevýhodou uvedeného princípu sú vyššie časové nároky na komunikáciu než pri symetrickom šifrovaní.

9.2 Autentizačný kód správy MAC

Alternatívnou technikou autentizácie je použitie tajného kľúča na generovanie kryptografického kontrolného súčtu, ktorý sa označuje ako MAC (Message Authentication Code).

Kryptografický kontrolný súčet MAC je blok bitov s konštantnou dĺžkou, ktorý sa pripája k originálnej správe, pričom dĺžka bloku je pomerne malá.

V uvedenej metóde sa predpokladá, že dve komunikujúce strany A a B vlastnia rovnaký tajný kľúč *k*. Ak strana A chce poslať správu strane B vypočíta kryptografický kontrolný súčet MAC ako funkciu správy *M* v tvare

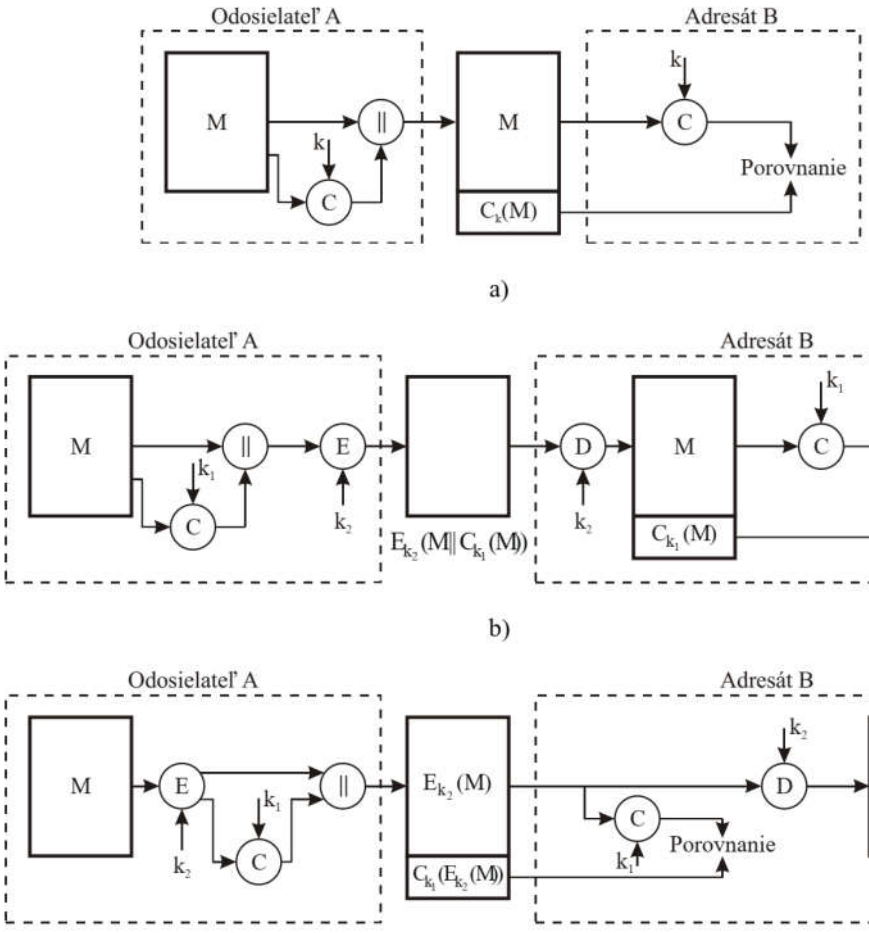
$$\mathrm{MAC}=\mathrm{C}_{\mathrm{k}}\left(M\right)$$

kde M – vysielaná správa

C – funkcia MAC

k – tajný kľúč

Princíp autentizácie správy s využitím MAC je uvedený na *Obr.* 9.3*a*. Z vysielanej správy M sa vypočíta pomocou kľúča k MAC, ktorý sa pripojí k správe M. Na prijímacej strane sa z prijatej správy vypočíta pomocou rovnakého kľúča k MAC, ktorý sa porovná s prijatým MAC.



Obr. 9.3 Použitie funkcie MAC
a) Autentizácia správy
b) Utajenie a autentizácia správy, MAC je pripojený k otvorenému textu
c) Utajenie a autentizácia správy, MAC je pripojený k zašifrovanému textu

Ak predpokladáme, že iba strana A a B vlastnia tajný kľúč k a ak sa vypočítaná hodnota MAC rovná prijatej hodnote MAC potom možno vyvodiť tieto závery:

1. Adresát B má istotu, že správa M nebola modifikovaná. Ak útočník zmenil správu a nezmenil hodnotu MAC, potom porovnaním prijatej hodnoty MAC a vypočítanej hodnoty MAC na strane adresáta B je zistená nezhoda. Pretože útočník nepozná tajný kľúč k, nie je schopný určiť správnu hodnotu MAC k zmenenej správe.
2. Adresát B má istotu, že správa M pochádza od deklarovaného subjektu A, pretože iba subjekt A vlastní tajný kľúč k, ktorý umožňuje generovať správnu hodnotu MAC.

Funkcia MAC je zdanlivo podobná šifrovaniu, avšak rozdiel spočíva v tom, že algoritmus, ktorý generuje MAC nie je reverzibilný, teda dešifrovanie nemožno realizovať. Funkcia MAC je zobrazenie typu „mnoho na jeden“. Argument funkcie MAC predstavuje súbor možných správ M a hodnota funkcie predstavuje autentizačný kód o určitej dĺžke. Ak dĺžka kódu (hodnoty) MAC je n bitov, potom počet všetkých možných hodnôt MAC je 2^n, pričom počet možných správ je N a platí $N>>2^n$. Navyše pri dĺžke kľúča k bitov je možno použiť 2^k rôznych kľúčov. Napr. ak správa M má 100 bitov a MAC je 10 bitov, potom pre úplný súbor 2^{100} rôznych správ možno využiť 2^{10} rôznych hodnôt MAC. Na každú hodnotu MAC pripadá $2^{100}/2^{10}=2^{90}$ rôznych správ.

Je potrebné tiež poznamenať, že použitie MAC podľa *Obr. 9.3a* zabezpečuje iba autentizáciu správy a zdroja ale nezabezpečuje utajenie, pretože správa M sa prenáša vo forme otvoreného textu. Autentizáciu a utajenie správy umožňujú postupy, ktoré sú uvedené na *Obr. 9.3b* a *Obr. 9.3c*.

V oboch prípadoch sa využívajú dva tajné kľúče k_1 a k_2. V prípade, ktorý je uvedený na *Obr. 9.3b*, sa najprv zo správy M a pomocou k_1 vypočíta MAC a pripojí sa k správe M a potom sa tento blok $M \| C_{k_1}(M)$ zašifruje kľúčom k_2, čím sa získa $E_{k_2}\left(M \| C_{k_1}(M)\right)$. Kryptografický súčet MAC sa teda pripojí k otvorenému textu.

V druhom prípade sa správa M najprv zašifruje kľúčom k_2 a potom sa z tejto zašifrovanej správy $E_{k_2}(M)$ vypočíta MAC s využitím kľúča k_1, čím vznikne autentizačný kód v tvare $C_{k_1}\left(E_{k_2}(M)\right)$, ktorý sa pripojí k zašifrovanej správe $E_{k_2}(M)$.

V oboch prípadoch sa na prijímacej strane vypočíta MAC z prijatej správy a realizuje dešifrovanie, pomocou ktorého sa získa M. Z uvedených dvoch postupov sa viac využíva postup zobrazený na *Obr. 9.3b*, v ktorom sa MAC pripája k otvorenému textu. Zároveň je potrebné poznamenať, že MAC nemožno použiť na digitálne podpisy, pretože odosielateľ aj adresát vlastnia rovnaký tajný kľúč.

9.2.1 Vlastnosti a realizácia MAC

Ak predpokladáme, že nepovolaná osoba pozná hodnotu C funkcie MAC ale nepozná kľúč k, potom funkcia MAC by mala mať tieto vlastnosti:

1. ak nepovolaná osoba pozná M a $C_k(M)$, potom by malo byť obťažné pre nepovolanú osobu vytvoriť správu M' takú, aby $C_k(M')=C_k(M)$
2. $C_k(M)$ by mala byť rovnomerne rozloženou v tom zmysle, že pre náhodný výber správ M a M' pravdepodobnosť toho, že $C_k(M)=C_k(M')$ je 2^{-n}, kde n je počet bitov hodnoty MAC
3. ak M' je určitou transformáciou M, teda platí $M'=f(M)$ a funkcia f znamená inverziu jedného, resp. niekoľkých bitov, potom pravdepodobnosť toho, že $C_k(M)=C_k(M')$ je 2^{-n}.

Prvá požiadavka znamená, že pre nepovolanú osobu by malo byť obťažné vytvoriť rovnakú správu M' s rovnakou hodnotou MAC bez znalosti kľúča.

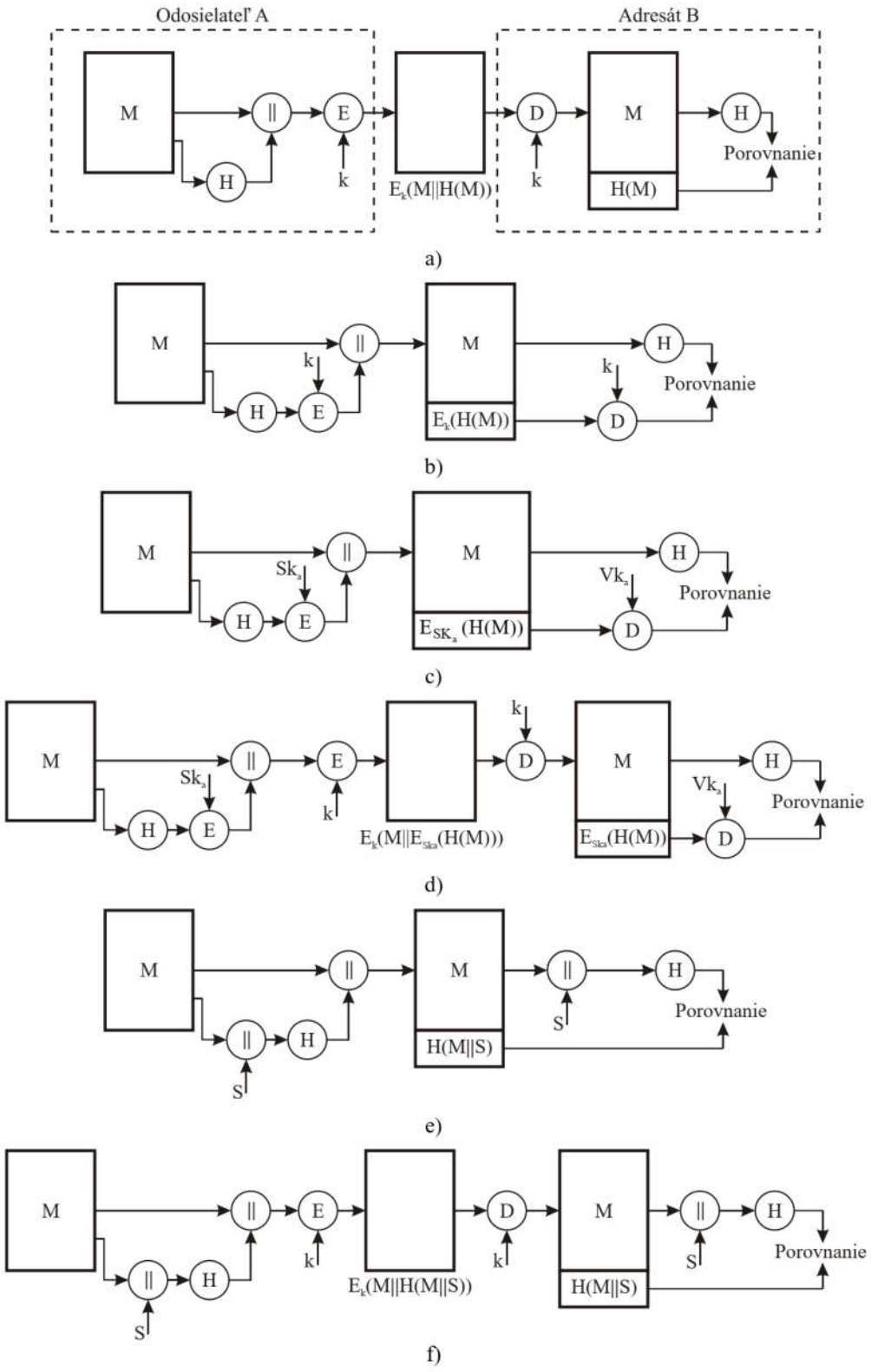
Druhá požiadavka je motivovaná bezpečnosťou voči útoku metódou totálnych skúšok založenou na znalosti vybraného otvoreného textu. Nepovolaná osoba nepozná kľúč k, ale má prístup k hodnote MAC. Útokom metódou totálnych skúšok generuje rôzne správy s cieľom nájsť správu s danou hodnotou MAC. Ak funkcia MAC sa vyznačuje rovnomerným rozdelením, potom útok vyžaduje v priemere 2^{n-1} pokusov.

Tretia požiadavka vyjadruje vlastnosť autentizačného algoritmu na báze MAC, ktorý by nemal byť slabší, resp. menej bezpečný s ohľadom na určitú časť správy, resp. určitú skupinu bitov, ako je jeho bezpečnosť s ohľadom na celú správu.

9.3 Hašovacie funkcie

Určitou variáciou autentizačného kódu správy MAC je jednocestná funkcia (one–way function), ktorá sa označuje ako ***hašovacia funkcia*** (hash function). Hašovacia funkcia, podobne ako funkcia MAC, používa ako argument správu M s premennou dĺžkou. Hodnota hašovacej funkcie $H(M)$ sa označuje ako ***hašovací kód h***, teda platí $h=H(M)$. Hašovací kód h na rozdiel od MAC nepoužíva tajný kľúč a je závislý len od správy M. Hašovací kód sa tiež označuje ako výťah zo správy (message digest), resp. ako hašovacia hodnota a je funkciou všetkých bitov správy. Z toho vyplýva, že zmena akéhokoľvek bitu, resp. bitov správy vyvolá zmenu hašovacieho kódu, ktorý má teda vlastnosti kryptografického kontrolného súčtu, resp. je vhodný na zabezpečenie integrity správy.

Rôzne spôsoby použitia hašovacích funkcií sú uvedené na *Obr. 9.4.*

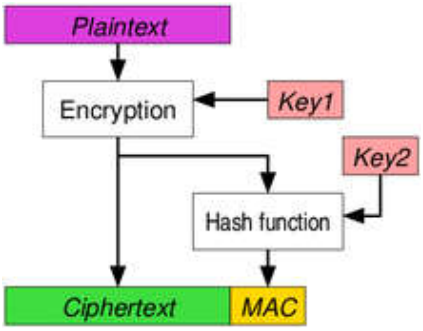


Obr. 9.4 Základné spôsoby použitia hašovacích funkcií

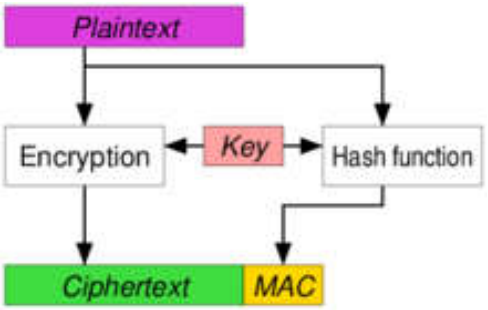
Bezpečné poradie šifrovania a MAC – diskusia

Možné riešenia autentizovaného šifrovania
(https://en.wikipedia.org/wiki/Authenticated_encryption)

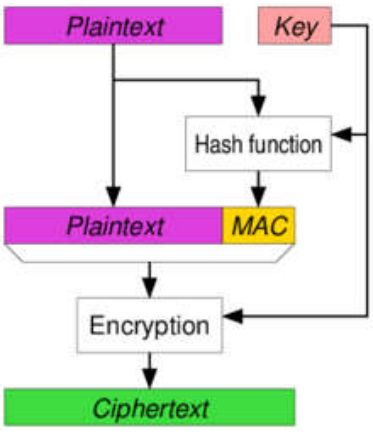
Encrypt-then-MAC



Encrypt-and-MAC



MAC-then-Encrypt



Z pohľadu bezpečnosti: Používajte Encrypt-then-MAC!!!

Dôvody pozri napr.:
https://crypto.stackexchange.com/questions/202/should-we-mac-then-encrypt-or-encrypt-then-mac

Vybrané odpovede z diskusie:

Hugo Krawczyk has a paper titled The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?). It identifies 3 types of combining authentication (MAC) with encryption:

1. Encrypt then Authenticate (EtA) used in IPsec;
2. Authenticate then Encrypt (AtE) used in SSL;
3. Encrypt and Authenticate (E&A) used in SSH.

It proves that EtA is *the* secure way to use, and both AtE and E&A are subject to attacks, unless the encryption method is either in CBC mode or it is a stream cipher.

In short, Encrypt-then-MAC is the most ideal scenario. **Any modifications to the ciphertext that do not also have a valid MAC can be filtered out before decryption, protecting against any attacks on the implementation.** The MAC cannot, also, be used to infer anything about the plaintext. MAC-then-Encrypt and Encrypt-and-MAC both provide different levels of security, but not the complete set provided by Encrypt-then-MAC.

Hašovaný autentizačný kód správy (HMAC)

(https://sk.wikipedia.org/wiki/HMAC
https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.198-1.pdf)

10.8 Algoritmus HMAC

Koncepcia autentizačného kódu správy MAC je založená na využití symetrických blokových šifier, ktoré používajú tajný kľúč. V poslednom období sa však zvýšená pozornosť venuje MAC, ktorý je odvodený od hašovacích funkcií a označuje sa skratkou HMAC.

Dôvody zvýšeného záujmu o túto modifikáciu MAC možno zhrnúť takto:

1. hašovacie funkcie MD5 a SHA–1 v softvérovej implementácii sú vo všeobecnosti rýchlejšie než blokové šifry, napr. DES
2. knižnica hašovacích funkcií je plne dostupná
3. na hašovacie funkcie nie sú aplikované obmedzenia, najmä zo strany USA, resp. iných krajín, na rozdiel od obmedzení týkajúcich sa symetrických šifier

Na zabudovanie tajného kľúča do existujúcich hašovacích funkcií možno použiť viaceré postupy, z ktorých uvedieme algoritmus HMAC definovaný v odporúčaní RFC2104. Algoritmus HMAC sa využíva v oblasti bezpečnosti IP sietí, resp. v iných internetových protokoloch, ako napr. v SSL.

Požiadavky na algoritmus HMAC podľa RFC2104 možno formulovať takto:

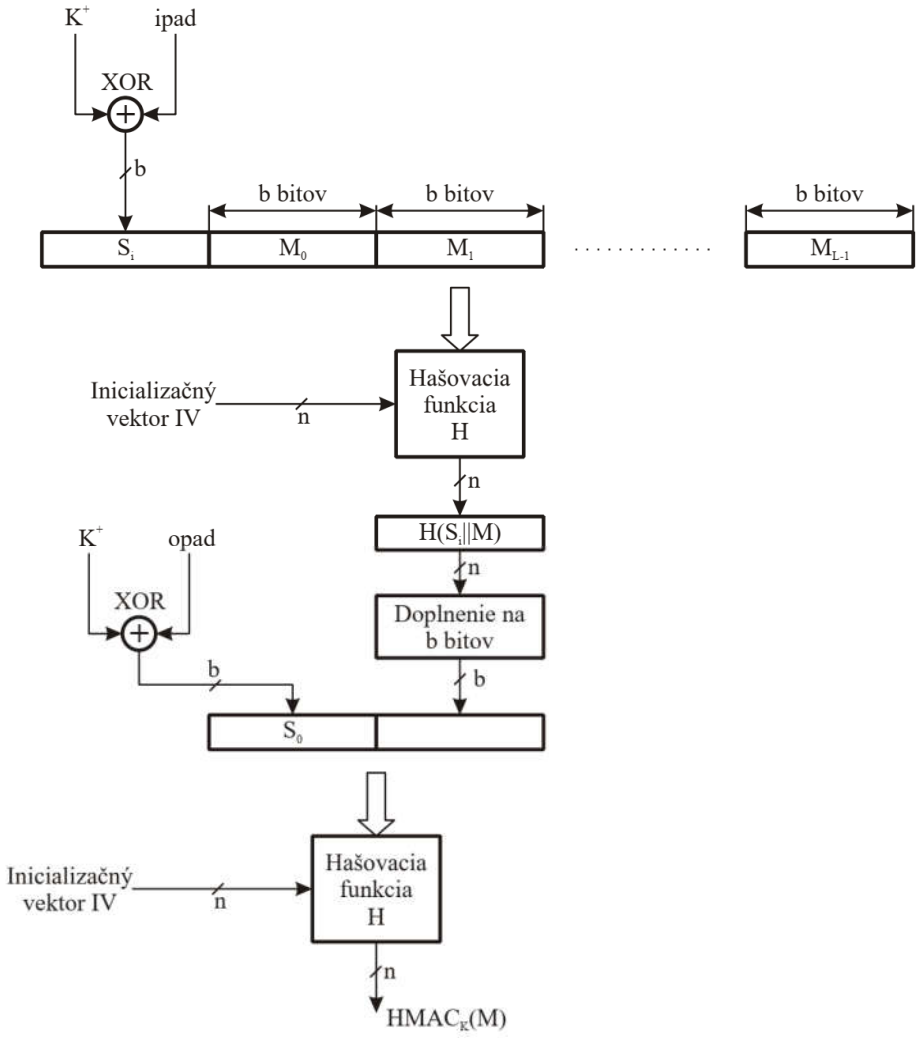
- použitie dostupných hašovacích funkcií bez ich modifikácie
- možnosť náhrady použitej hašovacej funkcie inou rýchlejšou, resp. bezpečnejšou hašovacou funkciou
- zachovanie pôvodnej výkonnosti, resp. bezpečnosti použitej hašovacej funkcie bez jej významnej degradácie
- jednoduché použitie kľúča
- ľahký odhad bezpečnosti a odolnosti voči kryptoanalýze.

Štruktúra algoritmu HMAC je uvedená na *Obr. 10.9*. V štruktúre sú použité tieto symboly:

- H – vložená hašovacia funkcia (napr. MD5, SHA–1, RIPEMD–160)
- *IV* – inicializačný vektor
- *M* – originálna správa
- M_i – *i*–tý blok *M*, pre $0 \le i \le L-1$
- *L* – počet blokov v správe *M*
- *b* – počet bitov v bloku M_i
- *n* – dĺžka hašovacieho kódu vloženej hašovacej funkcie, pričom obvykle $n<b$
- *K* – tajný kľúč; dĺžka kľúča by mala byť väčšia, resp. rovná *n*
- K^+ – doplnený kľúč *K*, pričom doplnenie na *b*–bitov sa realizuje 0 vľavo od posledného MSB bitu kľúča *K*
- *ipad* – konštanta s veľkosťou 00110110 (36_H), ktorá sa opakuje *b*/8 krát
- *opad* – konštanta s veľkosťou 01011100 ($5C_H$), ktorá sa opakuje *b*/8 krát

Generovanie výsledného kódu HMAC možno vyjadriť v tvare

$$HMAC_K(M) = H\left[\left(K^+ \oplus opad\right) \| H\left[\left(K^+ \oplus ipad\right) \| M\right]\right]$$



Obr. 10.9 Štruktúra algoritmu HMAC

Slovne možno štruktúru HMAC opísať takto:

1. pridanie nulových bitov zľava ku kľúču K pred bit MSB tak, aby sa vytvoril reťazec K^+ s dĺžkou b–bitov (ak napr. kľúč K má 160 bitov a b=512, potom je potrebné pridať zľava 352 nulových bitov, t. j. 44 bajtov)
2. vykonanie operácie XOR reťazcov K^+ a *ipad*, čím sa vytvorí blok S_i s dĺžkou b–bitov
3. pridanie správy M k bloku S_i
4. aplikovanie hašovacej funkcie H na reťazec vytvorený v kroku 3, pričom sa v prvom kroku použije inicializačný vektor IV s dĺžkou n–bitov. Výsledný hašovací kód $H(S_i \| M)$ sa doplní na b–bitov
5. vykonanie operácie XOR reťazcov K^+ a *opad*, čím sa vytvorí blok S_0 s dĺžkou b–bitov
6. pridanie bloku S_0 k výsledku v kroku 4
7. aplikovanie hašovacej funkcie H na reťazec vytvorený v kroku 6, čím vznikne výsledný reťazec $HMAC_K(M)$ s dĺžkou n–bitov.

Referenčné testovacie vektory pre rôzne hašovacie funkcie z rodín SHA:
https://csrc.nist.gov/projects/cryptographic-standards-and-guidelines/example-values

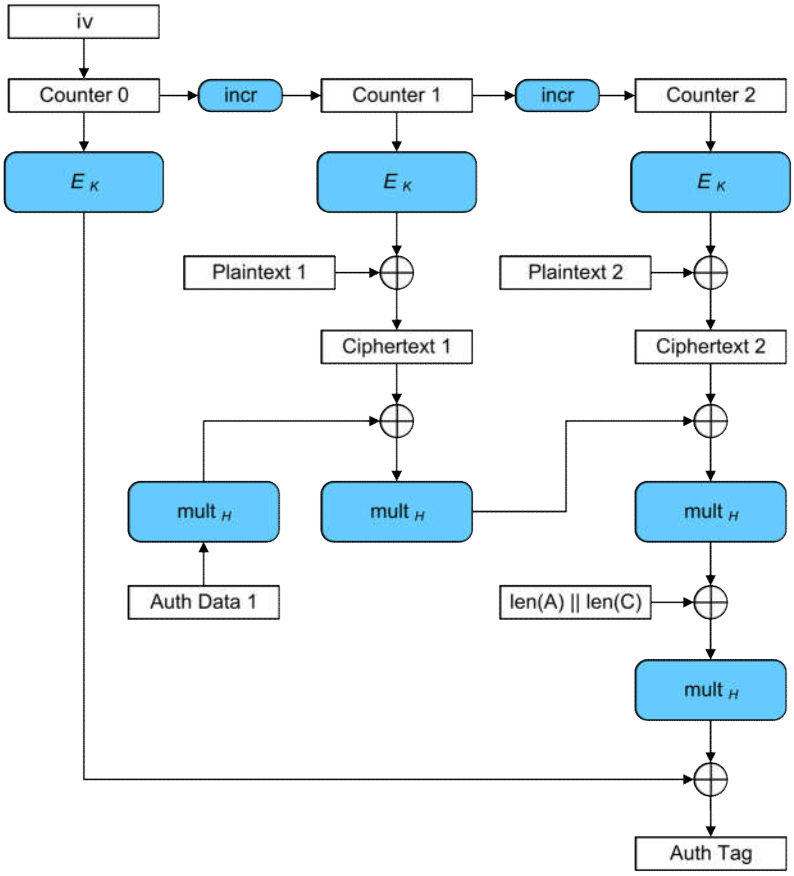
Príklad implementácie v jazyku C pre SHA256 a testovacie vektory z RFC 4231 – viď. napr. **test_hmac.c** v balíku https://github.com/01org/tinycrypt/ z cvičenia 3.

Galois Counter Mode (GCM)
([3, str.144-149],[2, str.134-136])

V praxi potrebujeme pri prenose okrem utajenia (dôvernosti) a integrity aj **autentizáciu dát** => využitie
Autentizované šifrovanie s priduženými dátami (Authenticated encryption with associated data (AEAD)), ktoré využíva kombináciu šifrovania a MAC kódu.

V praxi sa často používa ďalší mód blokovej šifry AES128, **Galoisov/ čítačový režim (GCM - Galois/Counter Mode)** ktorý realizuje okrem **šifrovania dát** (správy) aj výpočet **autentizačného kódu správy** (**Message Authentication Code**) pomocou **Encrypt-then-MAC** postupu.

Základný princíp GCM je zobrazený na nasledujúcom obrázku (mult $_H$ reprezentuje operáciu násobenia v príslušnom GF(Galois Field)- konečnom poli **GF(2^128)** a využíva **AES128**.



(zdroj: https://en.wikipedia.org/wiki/Galois/Counter_Mode)

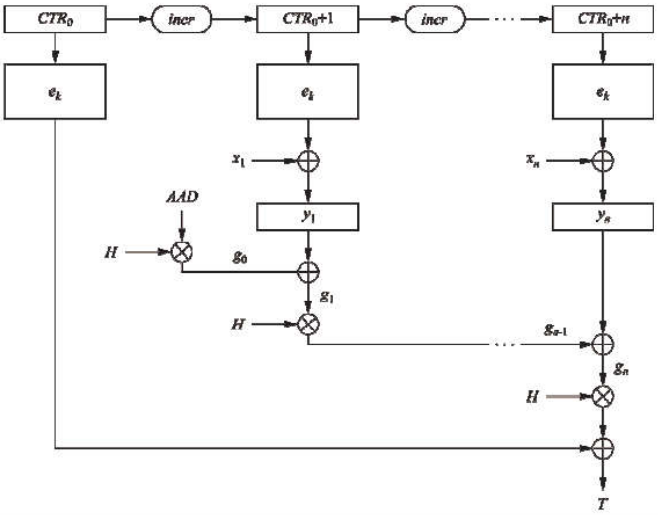
Galois Counter Mode (GCM)

- It also computes a *message authentication code* (MAC), i.e., a cryptographic checksum is computed for a message (for more information see Chapter 12 in *Understanding Cryptography*)
- By making use of GCM, two additional services are provided:
  - Message Authentication
    - the receiver can make sure that the message was really created by the original sender
  - Message Integrity
    - the receiver can make sure that nobody tampered with the ciphertext during transmission

Galois Counter Mode (GCM)

- For encryption
  - An initial counter is derived from an IV and a serial number
  - The initial counter value is incremented then encrypted and XORed with the first plaintext block
  - For subsequent plaintexts, the counter is incremented and then encrypted
- For authentication
  - A chained Galois field multiplication is performed (for more information Galois field see Chapter 4.3 in *Understanding Cryptography*)
  - For every plaintext an intermediate authentication parameter g_i is derived
    - g_i is computed as the XOR of the current ciphertext and the last g_{i-1}, and multiplied by the constant H
      - H is generated by encryption of the zero input with the block cipher
  - All multiplications are in the 128-bit Galois field $\mathrm{GF}(2^{128})$

Galois Counter Mode (GCM)



Encryption:

a. Derive a counter value CTR_0 from the IV and compute $\mathrm{CTR}_1 = \mathrm{CTR}_0 + 1$

b. Compute ciphertext: $y_i = e_k(\mathrm{CTR}_i) \oplus x_i, \quad i \geq 1$

Authentication:

a. Generate authentication subkey $H = e_k(0)$

b. Compute $g_0 = AAD \times H$ (Galois field multiplication)

c. Compute $g_i = (g_{i-1} \oplus y_i) \times H, \; 1 \leq i \leq n$ (Galois field multiplication)

d. Final authentication tag: $T = (g_n \times H) \oplus e_k(\mathrm{CTR}_0)$

Násobenie v GF(2^128) – Pseudokód

Each element is a vector of 128 bits. The i^{th} bit of an element X is denoted as X_i. The leftmost bit is X_0, and the rightmost bit is X_{127}. The multiplication operation uses the special element $R = 11100001\|0^{120}$, and is defined in Algorithm 1. The function rightshift() moves the bits of its argument one bit to the right. More formally, whenever $W = \text{rightshift}(V)$, then $W_i = V_{i-1}$ for $1 \leq i \leq 127$ and $W_0 = 0$.

Algorithm 1 Multiplication in $GF(2^{128})$. Computes the value of $Z = X \cdot Y$, where X, Y and $Z \in GF(2^{128})$.

```
Z ← 0, V ← X
for i = 0 to 127 do
  if Y_i = 1 then
    Z ← Z ⊕ V
  end if
  if V_127 = 0 then
    V ← rightshift(V)
  else
    V ← rightshift(V) ⊕ R
  end if
end for
return Z
```

Použité $\mathbb{GF}(2^{128})$ je definované ireducibilným polynómom

$$m(x) = x^{128} + x^7 + x^2 + x + 1. \tag{10.1}$$

V AES_GF, ktoré má len 2^8=256 prvkov sme násobenie dvoch prvkov mohli realizovať napr. aj s využitím logaritmickej a inverznej logaritmickej funkcie, ktorých tabuľky mali spolu len 512 bajtov.

Pre GF(2^128) by boli potrebné tabuľky obrovské a preto musíme využiť iné spôspby násobenia.

Vyššie uvedený pseudokód využíva rovnaký princíp výpočtu ako funkcia xtime() preberaná v BPS pri realizácii AES (pozri napr. implementáciu násobenia v GF(2^128) na stránke http://aplikovanakryptografia.fei.tuke.sk/).

Príklad využitia GCM módu (AEAD) na ochranu TCP/IP paketov

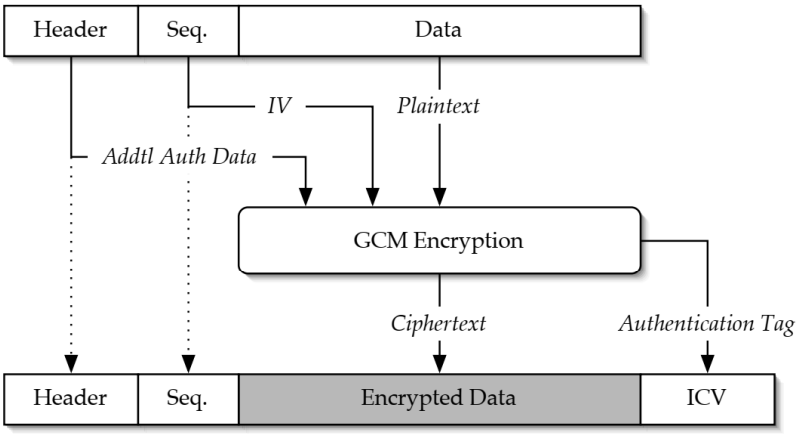


Figure 4: Using GCM to encrypt and authenticate a packet, showing how the fields of the security encapsulation map onto the inputs and outputs of the authenticated encryption mode.

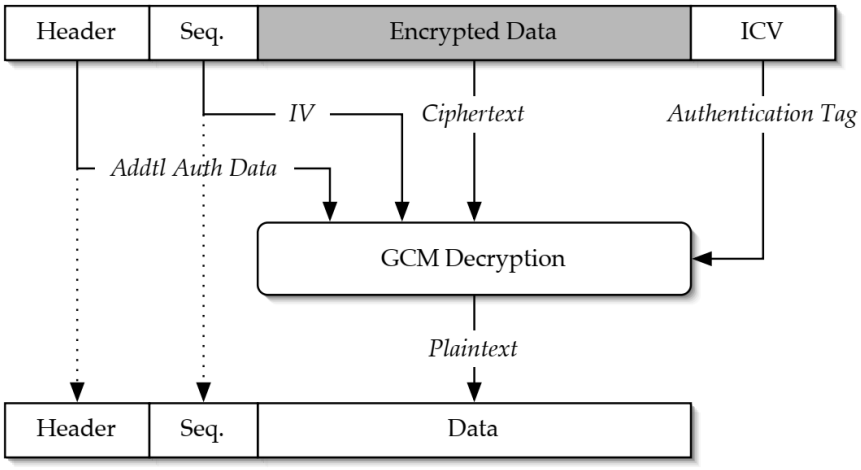


Figure 5: Using GCM to decrypt and verify the authenticity of a packet.

Princíp uloženia hesiel v operačných systémoch

Prihlasovacie heslá **nie sú ukladané** v otvorenom tvare.
V OS sú uložené **len hašované hodnoty** hesiel.

Obr. Základný princíp uloženia hesla užívateľa v OS

Heslo	Hash
123456	e10adc3949ba59abbe56e057f20f883e
password	5f4dcc3b5aa765d61d8327deb882cf99
12345	827ccb0eea8a706c4c34a16891f84e7b
12345678	25d55ad283aa400af464c76d713c07ad
football	37b4e2d82900d5e94b8da524fbeb33c0
qwerty	d8578edf8458ce06fbc5bb76a58c5ca4
1234567890	e807f1fcf82d132f9bb018ca6738a19f
1234567	fcea920f7412b5da7be0cf42b8c93759
princess	8afa847f50a716e64932d995c8e7435a
1234	81dc9bdb52d04dc20036dbd8313ed055
login	d56b699830e77ba53855679cb1d252da
welcome	40be4e59b9a2a2b5dffb918c0e86b3d7
solo	5653c6b1f51852a6351ec69c8452abc6
abc123	e99a18c428cb38d5f260853678922e03
admin	21232f297a57a5a743894a0e4a801fc3

Obr. Príklad tabuľky hašovaných hesiel pre najpoužívanejšie heslá

Uloženie hesiel v OS Windows (bez využitia tzv. Salt)

Používateľské záznamy sú uložené v SAM - databáze zabezpečenia účtov alebo v databáze služby Active Directory. Windows obvykle ukladá hesla v SAM (Security Account Manager) súbore. SAM súbor sa nachádza v D:\Windows\System32\config a využívajú sa vo Windows XP, Vista, 7, 8 a 10 na ukladanie hesiel. Môže byť použitý na lokálnu alebo vzdialenú autentizáciu používateľa. Využíva kryptografické prostriedky na zabránenie neautorizovaného prístupu k systému. Samotné hašované hodnoty sú uložené vo forme LM a NTLM hodnôt v SAM databáze, ktorá je súčasťou Windows registra. Každému používateľskému účtu môžu byť priradené dve heslá: LAN manažér kompatibilné heslo **(LAN Manager-compatible password)** a heslo systému Windows. LAN manažér kompatibilné heslo je založené na OEM – čísle originálneho výrobcu a môže obsahovať až 14 znakov, pričom nerozlišuje veľké a malé písmená. OWF verzia tohto hesla môže obsahovať až 16 znakov a je vypočítaná pomocou šifrovania DES. Systémové Windows heslo je založené na Unicode sade, rozlišuje veľké a malé písmená a môže mať až 128 znakov. Heslo je vypočítané pomocou hašovacej funkcie MD-4 firmy RSA. Používateľský účet môže obsahovať jednu verziu hesla, ale snaha je zachovať obe verzie hesiel. Od Windows 2000 SP2 je možné zabrániť ukladaniu LAN manažér hesla.

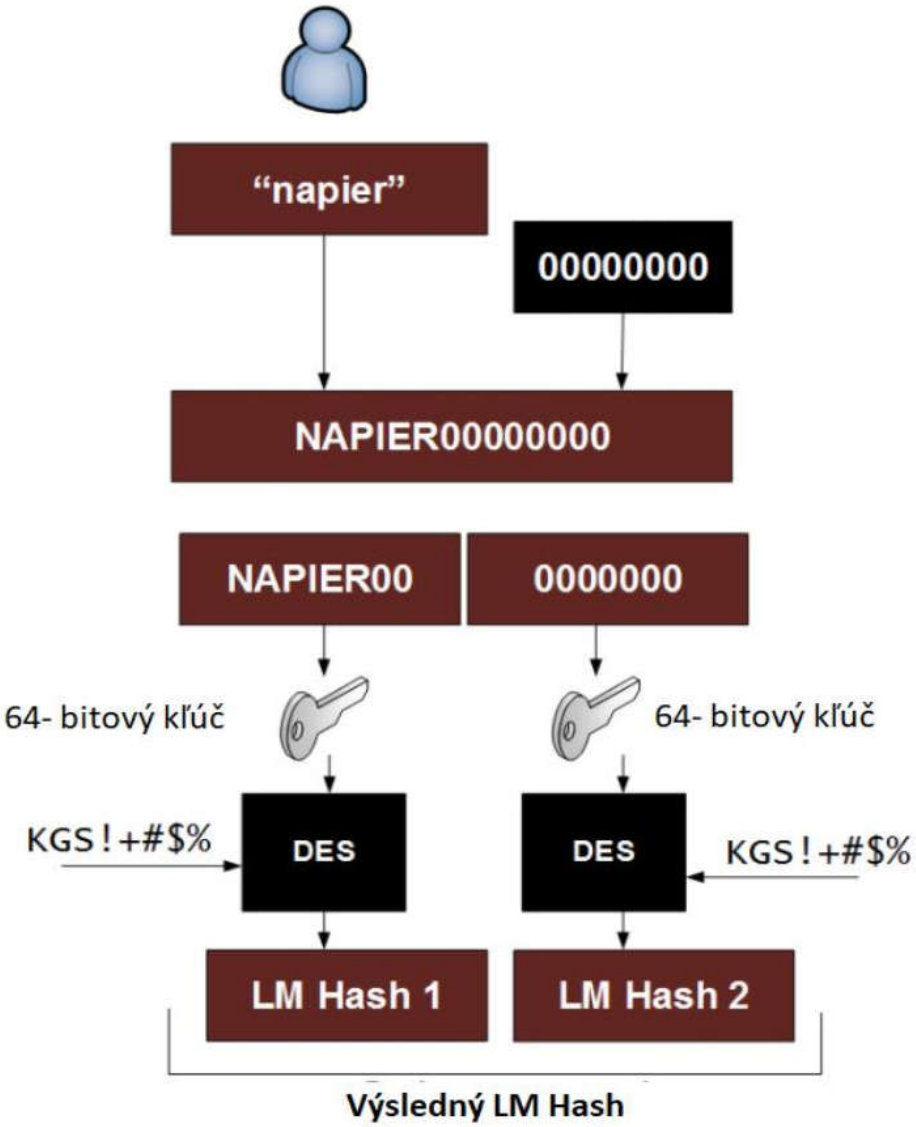
Samotný Windows nikdy nepoužil autentizáciu pomocou čistého textu. Windows vždy ukladal hesla transformované v podobe hašovacieho kódu a používa dva základné typy formátov: LM a NTLM.

LM algoritmus hašovania hesla

LM ukladá hesla založené na štandardu DES (Data Encryption Standard). Počas procesu sa všetky heslá konvertujú na veľké písmená, doplnia na štrnásť ASCII znakov a potom rozdelia na dve polovice. Maximálna povolená dĺžka hesla je 14 znakov. Celý tento postup robí LM formát veľmi ľahko prelomiteľný. Hlavná slabina LM autentizačného protokolu je, že nedokáže rozlišovať veľké a malé písmena. Veľkou nevýhodou je aj limitovaná dĺžka hesla na 14 znakov a pri procese jeho rozdelenia sa značne znižuje jeho kryptografická sila.

Samotný algoritmus pozostáva z týchto krokov:

1. Používateľské heslo je obmedzené na 14 znakov.
2. Heslo je vždy prevedené na veľké písmena a doplnené na 14 bajtov pomocou núl.
3. Takto doplnené heslo je rozdelené na dve 7 bajtové polovice.
4. Dve polovice sú potom použité na vytvorenie DES kľúčov. To sa vykoná prevedením oboch polovíc do bitového toku a vložením nulového bitu po každom siedmom bite (postupnosť **1010100** sa stane **10101000**). Takto sa vygeneruje 64 bitový kľúč, ale použitých je reálne len 56 bitov. Nulové bity pridané v tomto kroku sa neskôr zahodia.
5. Oba tieto kľúče sa šifrujú pomocou konštantného reťazca **KGS!@#$%**, ktorého výsledkom sú dve 8 bajtové zašifrované hodnoty. DES algoritmus by preto mal byť nastavený na ECB a režim vypĺňania textu by mal byť nastavený na **NONE**.
6. Tieto dve šifry sú potom spojené do 16 bajtového výsledku s názvom LM hašovací kód.



Obr. Proces vytváranie hašovanej hodnoty vo formáte LM