WEB saugumas

Marius Gžegoževskis

Turinys

- Kriptografijos uždaviniai ir priemonės.
- Maišos funkcijos.
- Asimetrinio šifravimo algoritmai.
- HTTP ir HTTPS.
- SSL ir TLS.

Kriptografijos uždaviniai ir priemonės

- Apžvelkime gerokai supaprastintą padėtį, į kurią patenka šiuolaikinės informacinės visuomenės žmonės:
  - » A (Algis) siunčia informaciją B (Birutei), perdavimo kanalą kontroliuoja Z (Zigmas) ir jaučiasi padėties šeimininkas. Jis gali pasyviai stebėti perdavimo kanalą, skaityti siunčiamus pranešimus ir kaupti dosjė; jis gali veikti aktyviai - pakeisti dalį siunčiamos informacijos, o kartais - apsimesti A ir siųsti jo vardu pranešimus B arba apsimesti B. Taigi dėl Zigmo veiksmų gali būti pažeidžiamos šios siunčiamų duomenų savybės:

Kriptografijos uždaviniai ir priemonės

- **Slaptumas** (konfidencialumas, *confidentiality*)
  - » Šifravimas
- **Vientisumas** (integralumas, *integrity*)
  - » Kriptografinės maišos funkcijos
  - » Skaitmeninis parašas
- **Autentiškumas** (tapatumo nustatymas, *authenticity*)
  - » MAC
  - » Skaitmeninis parašas

Informacijos vientisumas

- **Vientisumas** *(integrity)* – tai garantija, kad bus išsaugotos teisingos duomenų reikšmės. Tai užtikrinama draudžiant neautorizuotiems vartotojams kokiu nors būdu pakeisti, modifikuoti, sunaikinti, arba kurti duomenis.

Maišos funkcija ir santrauka

- Pranešimo vientisumui užtikrinti naudojama **pranešimo santrauka** (*message digest*, kartais dar vadinama *Modification Detection Code* (*MDC*)).
- Pranešimo santrauka skaičiuojama, naudojant maišos funkciją. Pranešimo santrauka vadiname maišos funkcijos reikšmę.
- **Maišos funkcija** (angl. *hash function*) vadiname funkciją, kuri bet kokio baigtinio ilgio ženklų eilutei priskiria *fiksuoto ilgio* eilutę.
- Maišos funkcijos naudojamos ne tik informacijos vientisumo patikrinimui, bet ir dokumento santraukai gauti skaitmeninio parašo schemose, slaptažodžių saugojimui, paieškos raktų formavimui duomenų bazėse ir panašiai.

Maišos funkcijų rezultatų pavyzdžiai

- Pranešimo *Informacijos saugumas* įvairių maišos funkcijų reikšmės:
  - **CRC32** (32 bitai): 9468ffc5
  - **MD5** (128 bitai): 5ccb2201a7ee633d2b2dc1ff527d4c79
  - **SHA-1** (160 bitų): 09035a1b2703a81b7a86e1abe9965b5756666591
  - **SHA-256** (256 bitai): 7765253a2cf3996d41c2a54af094a80bfcb8f9c4b4b8d4352c3e1ab8e3c7bbfb

Santraukos ilgis yra fiksuotas

Pavyzdys. Skaičiuosime duoto pranešimo MD5 maišos funkcijos reikšmę:

- Tuščias pranešimas:
  - » d41d8cd98f00b204e9800998ecf8427e
- *A*:
  - » 7fc56270e7a70fa81a5935b72eacbe29
- *Ilgesnis pranešimas iš kelių žodžių*:
  - » cd894604746deba896568cb8f2f6f197
- 3,5 MB dydžio failas:
  - » 9659f0218ab08598f7f53edec512479e

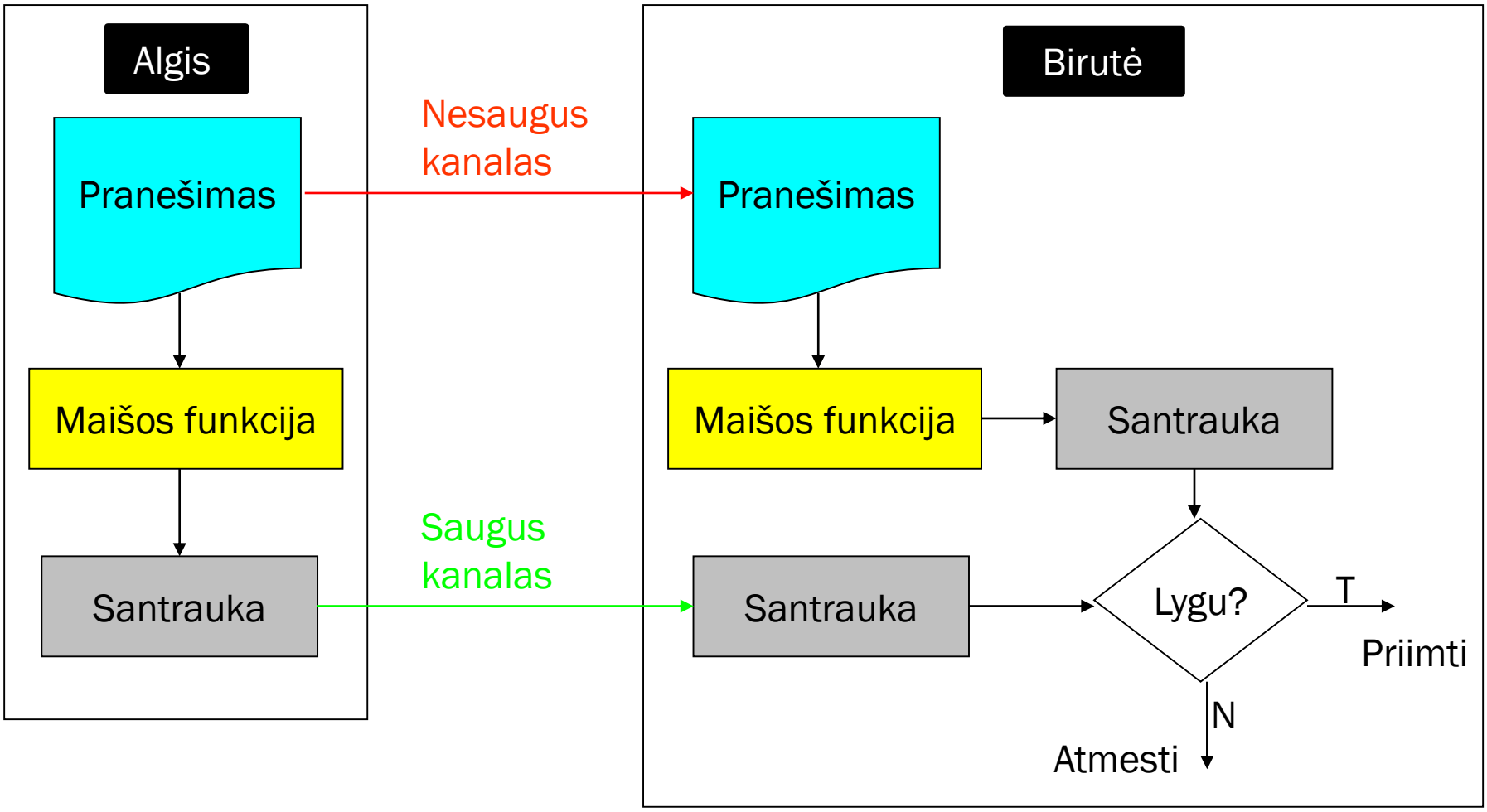
Maišos funkcijų savybės

- Maišos funkcijos yra *determinuotosios*, t. y. ne atsitiktinės:
  - » Skaičiuojant maišos reikšmę tai pačiai įvesčiai kelis kartus, visada bus gaunamas tas pats rezultatas.
- Kadangi galimų įvesčių yra daugiau, negu galimų išvesčių, tai maišos funkcijos nėra *injektyvios*:
  - » Gali būti, kad skirtingoms įvestims maišos reikšmės bus vienodos. Tai vadinama **sutapimu** (kolizija, angl. *collision*).

Pavyzdys

- Algis rašo testamentą, norėdamas paskirstyti savo turtą po mirties.
  - » Testamentas *neturi būti užšifruotas*, kad bet kas galėtų su juo susipažinti.
  - » Tačiau jo *vientisumas turi būti išsaugotas* (Algis nenori, kad jo testamentą pakeistų kas nors kitas).

Santraukos naudojimas vientisumui užtikrinti: Schema



Santraukos naudojimas vientisumui užtikrinti

- Kadangi maišos funkcijos yra žinomos viešai, tai bet kas gali apskaičiuoti jų reikšmes. Todėl santrauka turi būti perduodama *saugiu kanalu*, nes kitaip atakuojantysis galės pakeisti ir pranešimą, ir santrauką, apskaičiavęs pakeisto pranešimo maišos reikšmę.

- Be to, vientisumui užtikrinti bet kokios maišos funkcijos netinka. Pavyzdžiui, tarkime, kad maišos funkcija yra tokia, kad turint pranešimą ir jo santrauką, nesunkiai galima rasti kitą pranešimą su tokia pačia santrauka. Tada atakuojantysis galės pakeisti pranešimą į kitą pranešimą, ir gavėjas to nepastebės. Vientisumui užtikrinti reikia naudoti *kriptografines maišos funkcijas*.

Kriptografinės maišos funkcijos apibrėžimas

Kriptografine maišos funkcija vadiname maišos funkciją, kuri yra:

- **vienakryptė** (*preimage resistant*, *one-way*):

  pranešimo santrauką apskaičiuoti yra lengva, o turint santrauką rasti atitinkamą pranešimą yra skaičiavimų prasme neįmanoma (*computationally infeasible*),

- **atspari sutapimams** (*2-nd preimage resistant*, *weakly collision resistant*):

  turint pranešimą, skaičiavimų prasme neįmanoma rasti dar vieną pranešimą su ta pačia santrauka,

- **labai atspari sutapimams** (*collision resistant*, *strongly collision resistant*):

  skaičiavimų prasme neįmanoma rasti sutapimą, t. y. du pranešimus su ta pačia santrauka.

Lavinos efektas

- Pageidautina, kad kriptografinė maišos funkcija tenkintų savybę, vadinama **lavinos efektu** (angl. *avalanche effect*). Taip kriptografijoje vadinama kriptografinių algoritmų (paprastai blokinių kriptosistemų ir kriptografinių maišos funkcijų) savybė, kai nežymiai pakeitus įvestį (pavyzdžiui, pakeitus vos vieną bitą), išvestis pasikeičia žymiai (pavyzdžiui, pusė visų išvesties bitų pasikeičia).

- **Pavyzdys**. Žodžiai *Taisyklė* ir *taisyklė* skiriasi tik vienu bitu, o jų MD5 reikšmės labai skiriasi:
  - `T` – 0x54 – `0101 0100`
  - `t` – 0x74 – `0111 0100`
  - *Taisyklė*:
    - MD5: `f26ad7627a11f62e559a9a6516dd2392`
  - *taisyklė*:
    - MD5: `f0517727eaba12da46385b6b9418ff30`

Maišos funkcijos

- Pagrindinės plačiausiai naudojamos maišos funkcijos:

1. **MD5** (Message-Digest algorithm 5). Naudojama tokiuose plačiai naudojamuose protokoluose ir programose, kaip TLS ir SSL, SSH, PGP, S/MIME, IPsec.
2. **SHA** (Secure Hash Algorithm).

- Maišos (angl. **hash**) formavimo funkcijos yra vienakryptės funkcijos, kurios bet kokio ilgio pranešimui suformuoja fiksuoto ilgio maišos reikšmę (santrauką).
- Pavyzdžiui pritaikius MD5 maišos funkciją:
  - **Pradinis tekstas :** Automobilių spūstys vargina viso pasaulio vairuotojus.
  - **Santrauka (MD5):** E537F1B48B26FB88C2586E5F5F7E32BF

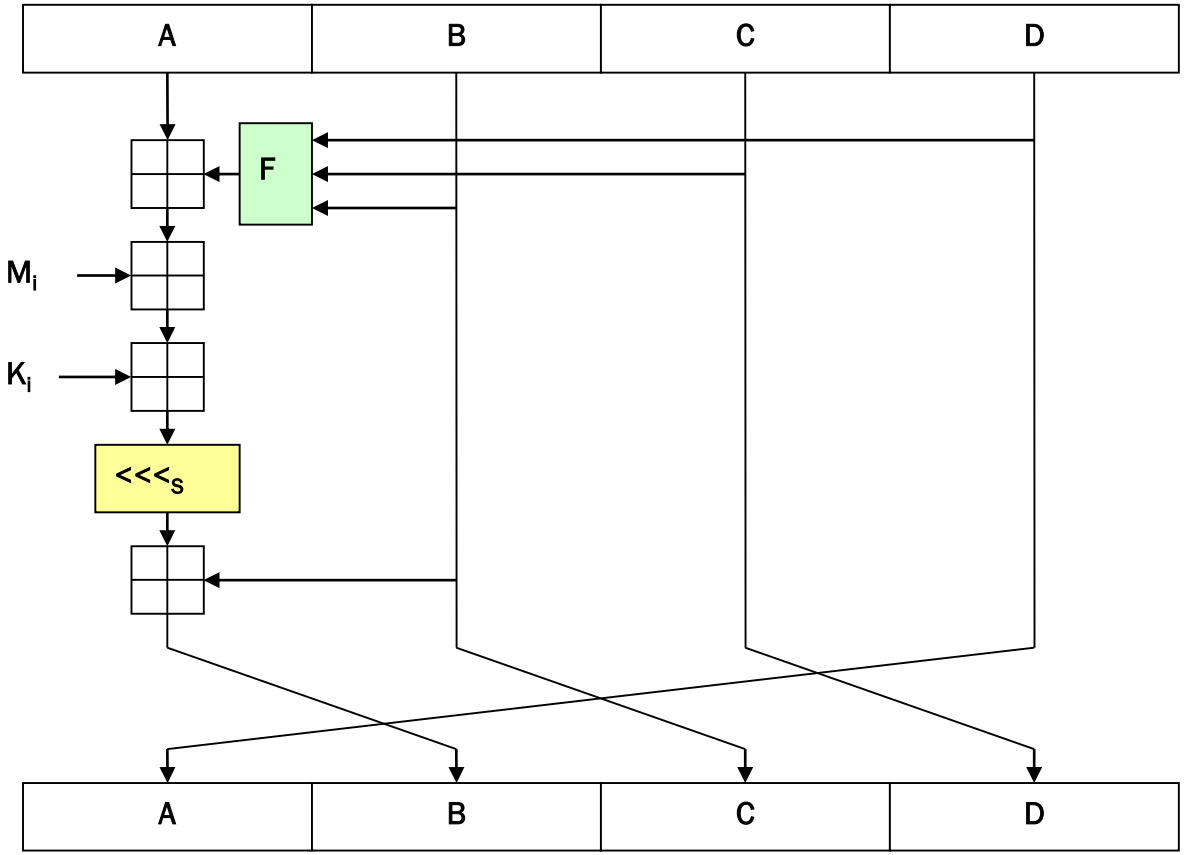
Maišos funkcijos

- Maišos funkcijos naudojamos informacijos vientisumo patikrinimui, slaptažodžių saugojimui, paieškos raktų formavimui duomenų bazėse ir panašiai.
- Prieš siunčiant pranešimą, jam suformuojama maišos reikšmė, kuri perduodama kartu su pranešimu.
- Vartotojas, gavęs pranešimą, vėl suformuoja jo maišos reikšmę ir sulygina ją su gautąja.

MD5 santraukos formavimo algoritmo esmė:

- Pradinis tekstas suskirstomas į N blokų po 512 bitų (64 baitus). Jei paskutiniame M_N bloke trūksta duomenų iki 512 bitų, bloko gale pridedamas 1 ir tiek nulių, kad būtų užpildyta likusi bloko dalis.
- Pagrindinis MD5 algoritmas dirba su 128 bitų rezultato eilute, padalinta į keturis 32 bitų žodžius, pažymėtus A, B, C ir D. Šie žodžiai užpildomi pradinėmis reikšmėmis.
- Po to algoritmas ima pradinio teksto 512 bitų ilgio blokus ir modifikuoja rezultato eilutę.
- Pranešimo bloko apdorojimas susideda iš 4 panašių ciklų, kurių kiekvienas susideda iš 16 operacijų pagrįstų netiesine funkcija F, sudėties moduliu ir postūmio į kairę operacijomis.

MD5 vieno ciklo veikimo schema:



$<<<_s$ – ciklinis postūmis kairėn per *s* bitų, *s* kinta kiekvienai operacijai; ⊞ - suma moduliu 2^{32}. K_i – konstanta, skirtinga kiekvienai iteracijai.

MD5 santraukos formavimo algoritmas

- Pradžioje būsenos eilutė užpildoma pradinėmis reikšmėmis:
  A: 01 23 45 67
  B: 89 ab cd ef
  C: fe dc ba 98
  D: 76 54 32 10
  Naudojamos keturios funkcijos F, kurių argumentai yra trys 32-bitų žodžiai, o rezultatas vienas 32-bitų žodis:

$$F(X,Y,Z) = (X \wedge Y) \vee (\neg X \wedge Z)$$

$$G(X,Y,Z) = (X \wedge Z) \vee (Y \wedge \neg Z)$$

$$H(X,Y,Z) = X \oplus Y \oplus Z$$

$$I(X,Y,Z) = Y \oplus (X \vee \neg Z)$$

$\oplus$ - suma moduliu 2, $\wedge$ - griežta disjunkcija (xor), $\vee$ - arba, $\neg$ - ne.

MD5 santraukos formavimo algoritmas

- Kiekviename etape žodžiai M_i perbėgami vis kita eilės tvarka.
- Konstantos K_i kiekvienoje iteracijoje yra skirtingos.
- Atlikus šiuos veiksmus visiems blokams gautos A, B, C, D reikšmės sujungiamos. Tai ir yra maišos funkcijos reikšmė.

Asimetrinio šifravimo algoritmai

- Diffte-Hellman.
- Elipsinės kreivės.
- **RSA** (Rivest-Shamir-Adleman).

RSA sistema

- **RSA** (Rivest-Shamir-Adleman, kintamo kodo ilgio) populiariausias viešojo rakto algoritmas.
- Šio metodo patikimumas pagrįstas didelių skaičių faktorizacijos sudėtingumu.
- RSA šifravimo raktai veikia abejomis kryptimis, t.y. galima užšifruoti žinutę privačiuoju raktu, o iššifruoti viešuoju siuntėjo raktu, arba galima užšifruoti žinutę viešuoju raktu, o iššifruoti privačiuoju raktu.
- Tai naudojama skaitmeniniam parašui, kadangi tik parašo autorius yra vienintelis asmeninio rakto savininkas. Tai garantuoja dokumento autoriaus ar siuntėjo autorystę ir duomenų pirmapradį originalumą.

RSA algoritmo esmė

1. Imami 2 pirminiai skaičiai *p* ir *q* (paprastai didesni už 10^{100}).
2. Apskaičiuojamos sandaugos $s = p * q$ ir $t = (p - 1) * (q - 1)$.
3. Randamas skaičius *a*, neturintis bendro vardiklio su *t*.
4. Randamas skaičius *b* toks, kad $b * a = 1$ pagal modulį *t*.
5. Išeities tekstas *T* suskaidomas blokais taip, kad $0 < T < s$. Tai galima padaryti grupuojant tekstą blokais po *k* bitų, kur *k* — didžiausias sveikas skaičius, kuriam $2^k<s$.
6. Informacijos užkodavimui reikia apskaičiuoti $C = T * b$ pagal modulį s, o atkodavimui - $T = C * a$ pagal modulį *s*. Tokiu būdu informacijos užšifravimui reikia žinoti skaičių porą *(b, s)*, o iššifravimui — skaičių porą *(a, s)*. Pirmoji pora yra viešasis raktas, antroji pora – asmeninis raktas.

RSA algoritmo pavyzdys (raktų formavimas) 1 iš 3

- Realiose realizacijose naudojami labai dideli skaičiai, iki 100+ skaitmenų. Pavyzdyje paprastumo dėlei paimsime mažus skaičius.

1. Tegul $p = 3$, $q = 11$. Dalybos moduliu operaciją žymėsime ***mod*** ženklu. Raktų porą gaunama taip:
   - $s = 3 * 11 = 33;$
   - $t = (3 – 1) * (11 – 1) = 20.$
2. Randame skaičių a, neturintį bendro vardiklio su 20. Tai bus $a = 7$.
3. Apskaičiuojame skaičių b tokį, kad $(b * 7)$ ***mod*** $t = 1$. Tai bus $b = 3$.
4. Taigi raktų pora: $(3, 33)$ – viešasis raktas; $(7, 33)$ – asmeninis raktas.

RSA algoritmo pavyzdys (raktų formavimas) 2 iš 3

- Tarkime, mums reikia užšifruoti keturioliktą abėcėlės raidę, kurios kodas yra 14.

1. Pakeliame laipsniu 3 skaičių 14. $14^3 = 2744$;
2. Daliname rezultatą moduliu 33: $2744 \bmod 33 = 5$;

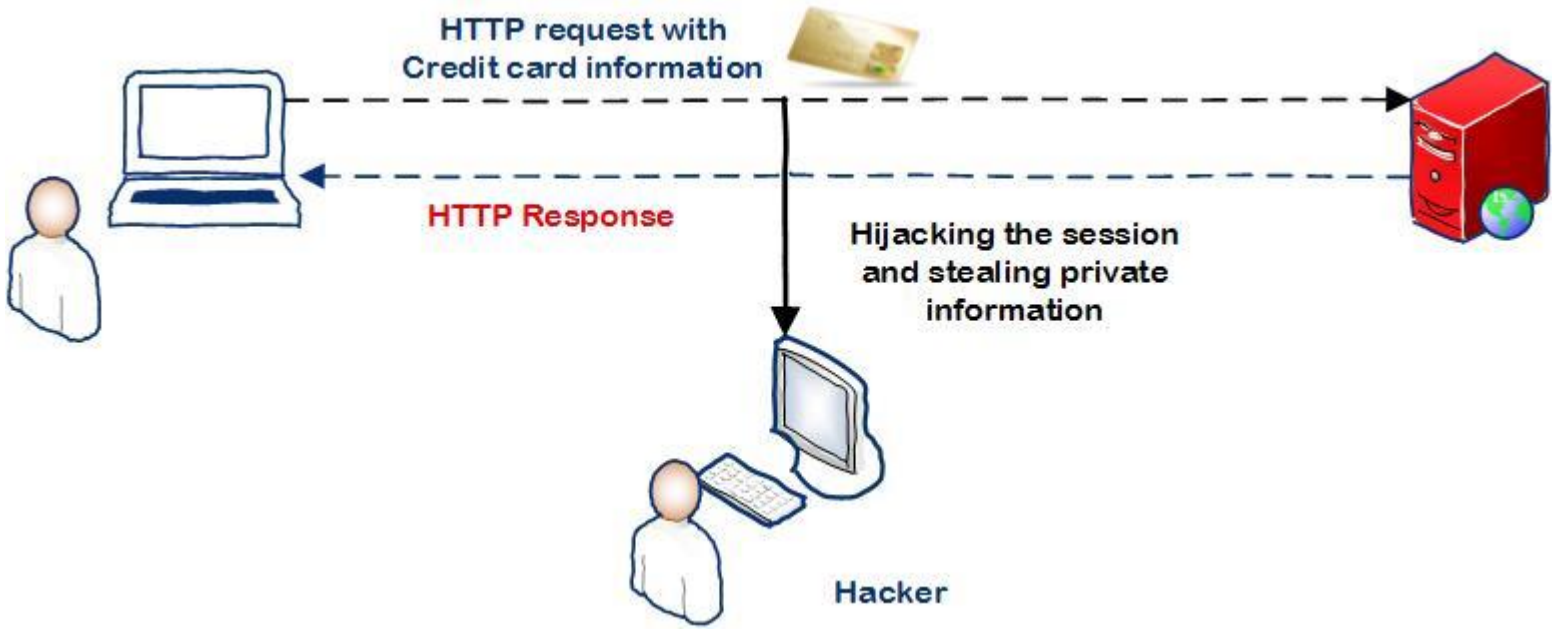
- Taip mūsų *keturioliktoji* abėcėlės raidė tampa *penktąja*.

RSA algoritmo pavyzdys (raktų formavimas) 3 iš 3

1. Raidei atstovaujantį skaičių 5 pakeliame laipsniu 7: $5^7 = 78125$;
2. Daliname rezultatą moduliu 33: 78125 **mod** 33 = 14;

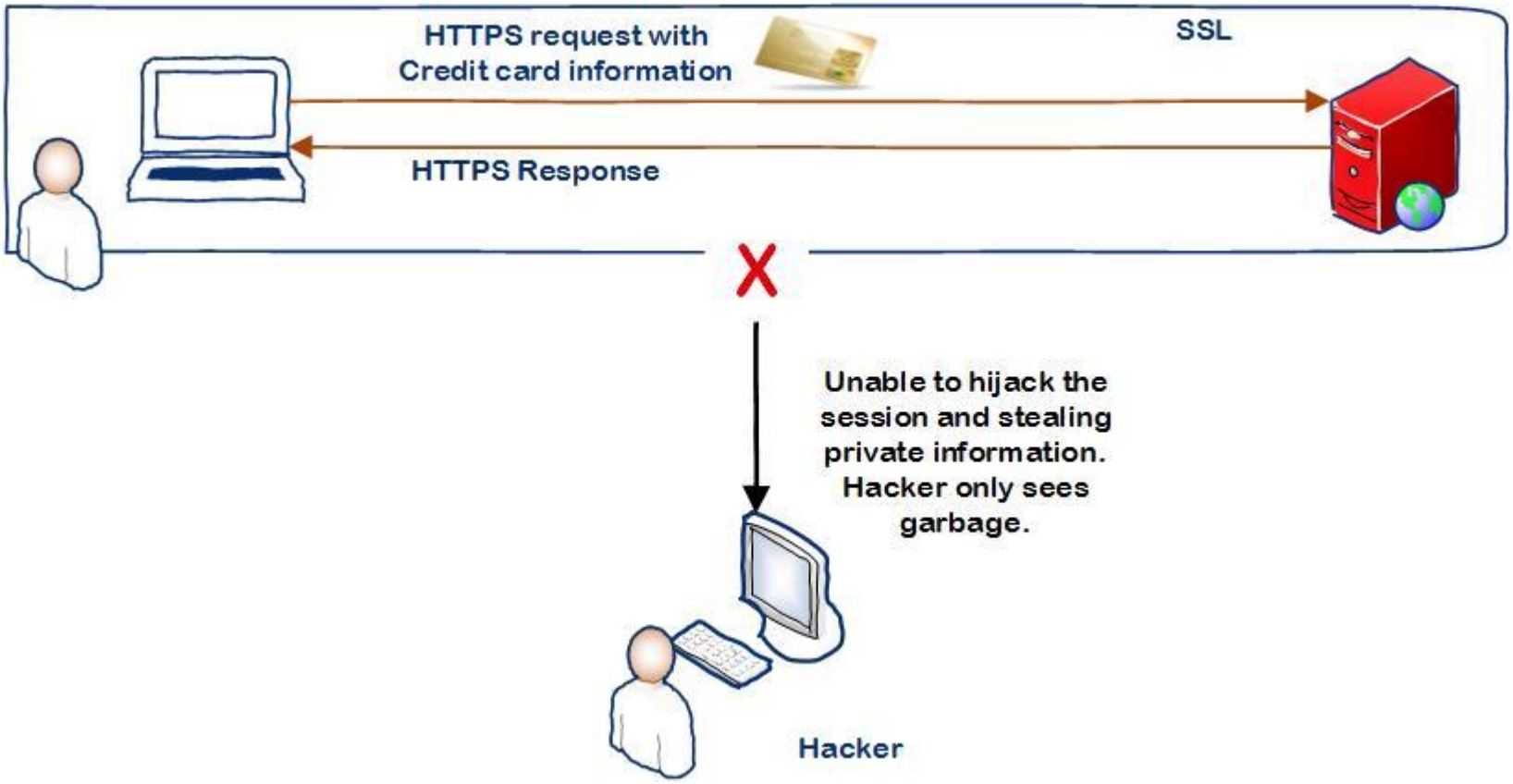
- Taip *penktoji* abėcėlės raidė tampa *keturioliktąja*.

HTTP protokolas



Duomenys nešifruojami

HTTPS protokolas



Duomenys šifruojami

SSL

- Šiuo metu Web paslaugos netelpa tik į HTTP protokolo rėmus, todėl sukurta daug patobulintų protokolų, skirtų įvairioms kliento reikmėms tenkinti. Vienas iš populiariausių papildomų servisų yra HTTPS (Secure HTTP) – saugus HTTP protokolas, kurį UNIX sistemoje valdo SSL (Secure Socket Layer).

- SSL (Secure Socket Layer) - tai rinkos standartu tapusi technologija, kuri garantuoja saugų duomenų perdavimą Internetu. Tai kriptografinis protokolas skirtas informacijos, sklindančios Internete apsaugojimui šifruojant. HTTPS technologijos šiuo metu, yra labai plačiai naudojamos elektroninėje komercijoje ir bankinėse sistemose.

SSL ir TLS

- SSL (Secure Sockets Layer) ir TLS (Transport Layer Security) - tai protokolai kurie atsako už perduodamų duomenų šifravimą ir autentifikavimą tarp vartotojo ir serverio.

HTTPS

- HTTPS – HTTP protokolo praplėtimas, palaikantis šifravimą. Duomenys perduodami HTTP protokolui „įpakuojami“ į SSL arba TLS protokolą, tuo pačiu užtikrinamas duomenų saugumas. Skirtingai nuo HTTP, HTTPS naudoją 433 TCP kanalą.
- HTTPS arba S-HTTP - tai Saugus HTTP protokolas (Secure Hypertext Transfer Protocol). Jis orientuotas siųsti saugioms žinutėms ir naudoti kartu su HTTP protokolu. Jis yra sukurtas egzituoti kartu su HTTP žinučių modeliu ir būti lengvai integruojamu HTTP programose.

SSL IR TLS

- Pats pagrindinis skirtumas tarp SSL ir TLS (Transport Layer Security), yra tame, kad SSL protokolo veikimas prasideda nuo apsaugos užtikrinimo, t.y prisijungimo prie apsaugoto tinklo ir susisiekimas su SSC (Secure Socket Certificate) ir t.t. Tuo tarpu TLS protokolas iš pradžių atlieka prisijungimą prie serverio su neapsaugotu „**Hello**“ pranešimu, ir tik tuomet atlieka apsaugos užtikrinimą kai buvo sėkmingai atliktas „pasisveikinimas“ tarp vartotojo ir serverio, kuris užtikrina „**TLS Handshake Protocol**“. Jeigu „pasisveikinimas“ neįvyko, tuomet sujungimas blokuojamas.

- Abu protokolai užtikrina, kad persiuntimo metu duomenys būtų šifruojami ir užtikrina sėkmingą jų kelionę iki tikslo. Tuo tarpu serveris turi palaikyti SSL ir TLS protokolus ir būti autentifikuotas SSC, tokiuose kaip Verisign, Thawte ar Cybertrust. Jeigu serveris nebus autentifikuotas SSC, tuomet personalinis kompiuteris perspės vartotoją apie tai.

SSL (Secure Socket Layer) — veikimo principas („kriptografiškai“)

1. Klientas prisijungia prie serverio ir siunčia jam palaikomų šifravimo algoritmų sąrašą.
2. Serveris atsako su algoritmo pavadinimu, savo viešuoju raktu, bendru raktu bei maišos algoritmo pavadinimu.
3. Klientas gali patikrinti, ar viešas raktas priklauso tam serveriui.
4. Tada klientas generuoja atsitiktinį seanso raktą ir siunčia jį serveriui, užšifravęs su serverio viešuoju raktu.
5. Serveris iššifruoja seanso raktą su savo slaptuoju raktu ir naudoja jį duomenų šifravimui seanso metu.
6. Klientas patikrina serverį, siųsdamas jam atsitiktinę eilutę, šifruotą seanso raktu.
7. Serveris patvirtina gavimą.

Šis metodas naudojamas kliento-serverio autentifikavimui bei elektroninėje komercijoje.

SSL (Secure Socket Layer) — supaprastintas veikimo principas

1. Vartotojas ateina į apsaugotą svetainę.

2. Serveris patvirtina svetainės tapatybę pasirašydamas sertifikatą ir nusiųsdamas jį klientui. Naršyklė panaudoja sertifikato viešąjį raktą (viešasis raktas gaunamas kartu su sertifikatu) serverio sertifikato savininko tapatybei patikrinti.

3. Naršyklė patikrina, ar sertifikatas buvo išduotas žinomo sertifikavimo paslaugų teikėjo. Jeigu sertifikatas yra išduotas nežinomo paslaugų teikėjo, naršyklė apie tai informuoja vartotoją.

4. Vartotojas pats gali patikrinti ar sertifikavimo paslaugų teikėjas išdavė sertifikatą tikrai tai svetainei, į kurią atėjo vartotojas.

5. Serveris reikalauja vartotojo skaitmeninio sertifikato, kad patikrinti jo tapatybę.

6. Vartotojas pasirenka, kurį iš jo turimų sertifikatų (žinoma, jeigu jis turi daugiau nei vieną sertifikatą) parodys serveriui.

7. Serveris sukuria ir užkoduoja seanso raktą bei saugiai nusiunčia jį internetu, ir tokiu būdu sukuriamas saugus virtualus kanalas tarp vartotojo naršyklės ir Web serverio.

SSL seanso metu galima būtu išskirti 3 pagrindines fazes

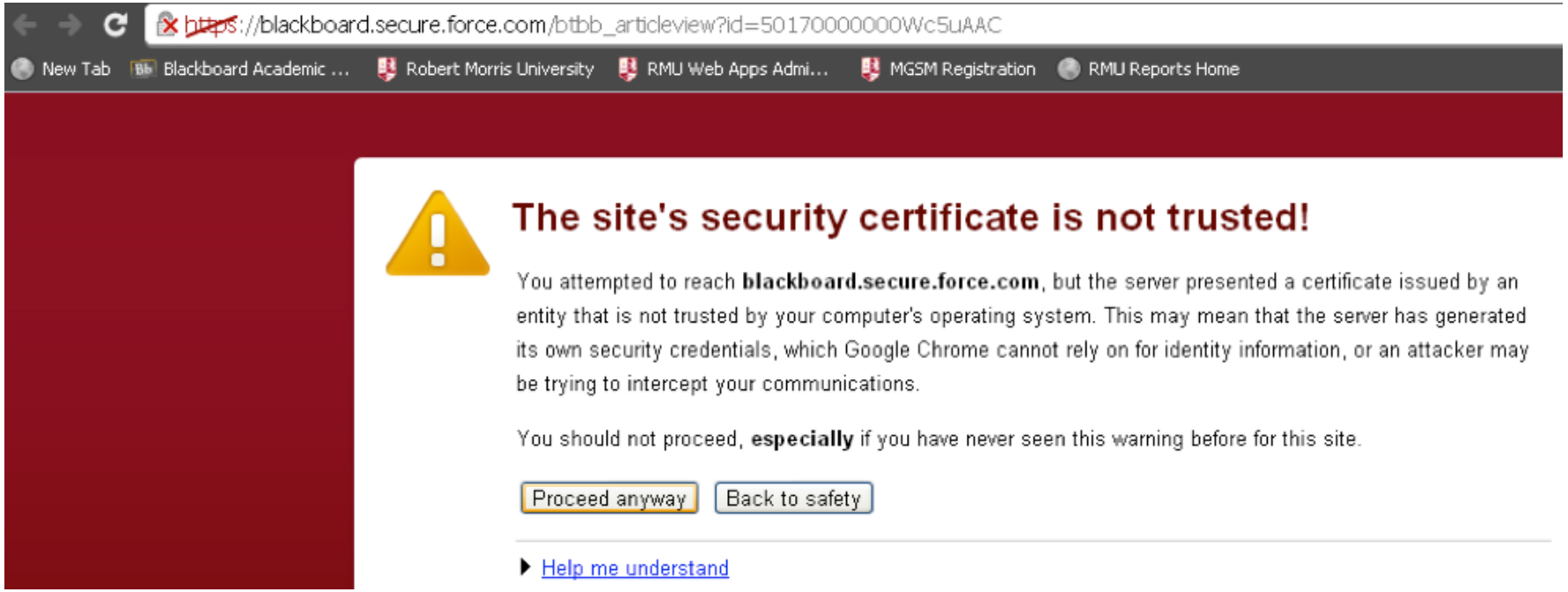
- SSL seanso metu galima būtu išskirti 3 pagrindines fazes:
  - » Dialogas tarp kliento ir tarnybinės stoties kurio metu pasirenkamas šifravimo algoritmas.
  - » Raktų pasikeitimas pagrįstas atvirojo rakto kriptografija ir identifikavimas pagrįstas sertifikatu.
  - » Simetriniais šifravimo algoritmais užšifruotu duomenų perdavimas.

  Taigi primoje fazėje klientas ir tarnybinė stotis aptaria kriptografinio algoritmo pasirinkimą tolesniam seansui. SSL 3.0 protokolo versijoje galimi šie algoritmai:

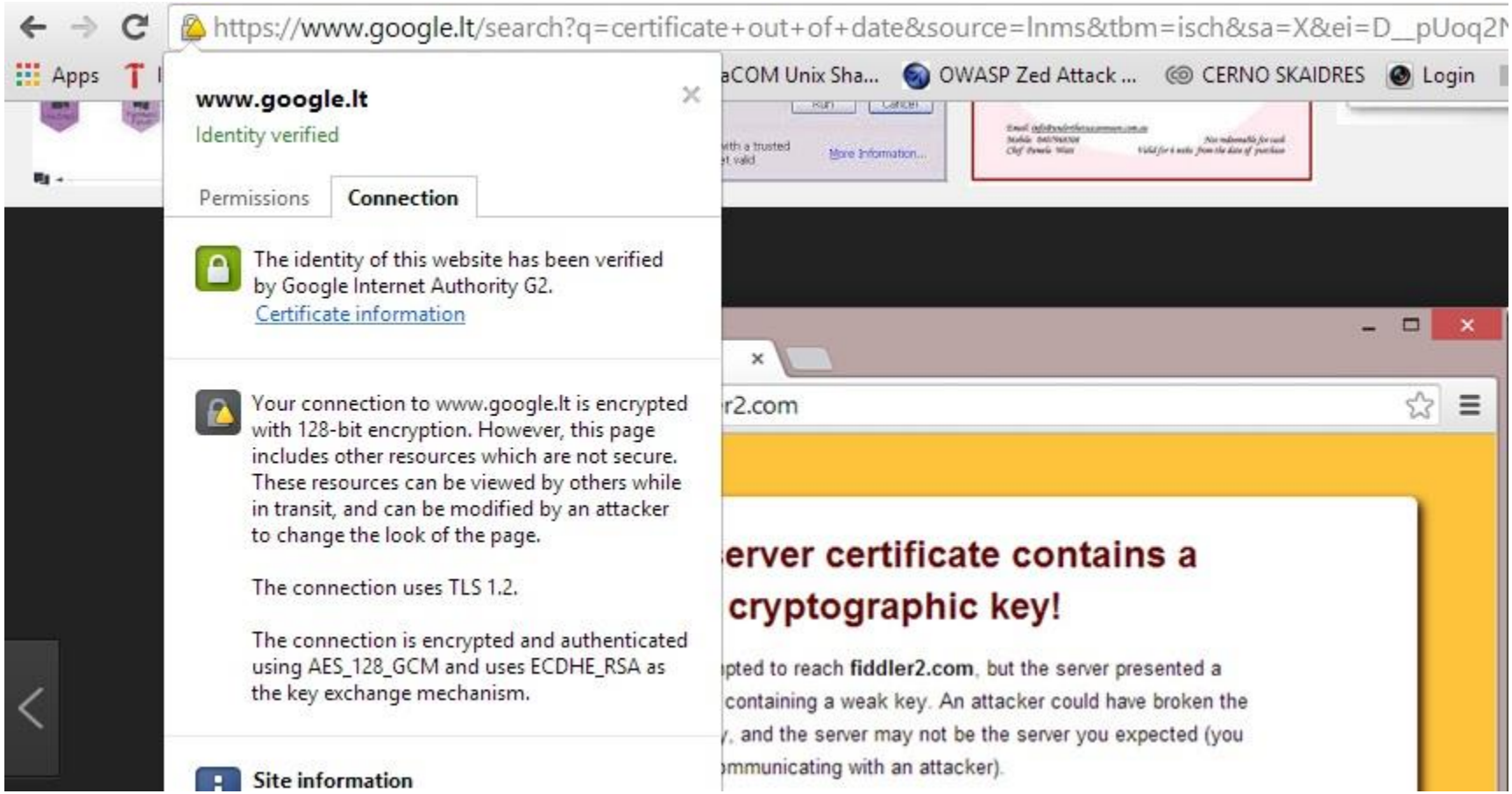
  - » Asimetriniam šifravimui: *RSA, Diffie-Hellman, DSA arba Fortezza*.
  - » Simetriniam duomenų šifravimui: *RC2, RC4, IDEA, DES, Triple DES arba AES*.
  - » „Sausainiukų“ funkcijom: *MD5 arba SHA*.

HTTPS galiojančių sertifikatų pavyzdžiai skirtingose naršyklėse

HTTPS nepatikimo sertifikato pavyzdys



HTTPS negaliojančio sertifikato pavyzdys



IŠVADOS

- HTTPS protokolas paremta SSL technologija puiki priemonė saugiai keistis duomenimis Internete. Simetrinis šifro raktas persiunčiamas pasinaudojus asimetrine kriptografija. Asimetrinės kriptografijos raktai kuriami taip, kad žinant viešąjį raktą būtų neįmanoma sužinoti privataus rakto.
- HTTPS protokolas tinka naudoti:
  - E-versle;
  - Elektroninėje bankininkystei;
  - Visur kur reikalingas saugus duomenų perdavimas.