VGTU Elektronikos fakulteto

Kompiuterijos ir ryšių technologijų katedros

Kompiuterių inžinerijos programos

Įterptinių kompiuterių specializacijos

_____2024-12-08_____

(data)

IV kurso Kikf-21 gr. studento Airono Petraičio

Duomenų šifravimas ir autentifikacija elektroninėse prekybinėse svarstyklėse referatas

Turinys

1.Šifravimo metodai 3

AES (Advanced Encryption Standard) šifravimo metodas 3

RSA (Rivest-Shamir-Adleman) šifravimo metodas 3

TLS (Transport Layer Security) 4

SSL (Secure Sockets Layer) 5

2. Autentifikacijos metodai 6

OAuth autentifikacija 6

API raktų autentifikacija 7

Nuorodos 9

1.Šifravimo metodai

AES (Advanced Encryption Standard) šifravimo metodas

Išplėstinis šifravimo standartas (AES) nurodo FIPS patvirtintą kriptografinį algoritmą, kuris gali būti naudojamas elektroniniams duomenims apsaugoti. AES algoritmas yra simetriškas blokinis šifras, galintis užšifruoti (šifruoti) ir iššifruoti (iššifruoti) informaciją. Šifravimas konvertuoja duomenis į nesuprantamą formą, vadinamą šifruotu tekstu; iššifravus šifruotą tekstą, duomenys vėl konvertuojami į pradinę formą, vadinamą paprastu tekstu. AES algoritmas gali naudoti 128, 192 ir 256 bitų kriptografinius raktus, kad užšifruotų ir iššifruotų duomenis 128 bitų blokais.

AES yra naudojamas duomenų šifravimui realiuoju laiku. Elektroninėse svarstyklėse tai gali būti taikoma apsaugoti svėrimo duomenis, kainų skaičiavimus ir kitą jautrią informaciją, kuri siunčiama į POS (Point of Sale) sistemas arba saugoma vietinėje atmintyje.

- **Privalumai**: AES yra greitas ir efektyvus, todėl tinkamas mažos galios įrenginiams, tokiems kaip prekybinės svarstyklės. Dėl savo efektyvumo AES naudojamas IoT (daiktų interneto) įrenginiuose.
- **Praktinis pavyzdys**: Jei svarstyklės turi galimybę jungtis prie interneto arba dirbti su išorinėmis POS sistemomis, AES gali būti naudojamas šifruoti šiuos ryšius.

RSA (Rivest-Shamir-Adleman) šifravimo metodas

RSA yra asimetrinis viešojo rakto kriptografijos algoritmas, plačiai naudojamas elektroninėje prekyboje. Algoritmą 1977 m. aprašė Ron Rivest, Adi Shamir ir Len Adleman; raidės RSA yra jų pavardžių inicialai.

RSA sistemos saugumas priklauso nuo labai didelių sveikųjų skaičių faktoringo sudėtingumo. Nauji greiti algoritmai šioje srityje gali padaryti RSA nesaugų, tačiau paprastai manoma, kad tai mažai tikėtina.

Tarkime, Bobas nori išsiųsti m žinutę Alisa. Jis žino N ir e, apie kuriuos Alisa paskelbė. Jis paverčia m skaičiumi n < N, naudodamas kokį nors anksčiau sutartą grįžtamąjį protokolą. Pavyzdžiui, kiekvienas paprasto teksto pranešimo simbolis gali būti konvertuojamas į jo ASCII kodą, o kodus galima sujungti į vieną skaičių. Jei reikia, jis gali suskirstyti m į dalis ir užšifruoti kiekvieną gabalėlį atskirai. Tada jis apskaičiuoja šifruotą tekstą C:

$$c \equiv n^e \pmod{N}$$

RSA yra daug lėtesnė nei DES ir kitos simetriškos kriptosistemos. Praktikoje Bobas paprastai užšifruoja slaptą pranešimą simetriniu algoritmu, užšifruoja (palyginti trumpą) simetrinį raktą su RSA ir perduoda tiek RSA užšifruotą simetrinį raktą, tiek simetriškai užšifruotą pranešimą Alice.

Ši procedūra kelia papildomų saugumo problemų. Pavyzdžiui, labai svarbu simetriniam raktui naudoti stiprų atsitiktinių skaičių generatorių, nes priešingu atveju Ieva galėtų apeiti RSA atspėjusi simetrinį raktą.

- **Kur naudojamas?**: RSA paprastai naudojamas ne duomenims šifruoti, o raktams keistis. Pvz., RSA gali būti naudojamas autentifikuoti svarstykles, kad tik autorizuoti naudotojai galėtų atlikti operacijas ar prisijungti prie centrinės sistemos.
- **Privalumai**: RSA naudojamas sukurti saugią pradinę sesiją, po kurios jau gali būti naudojamas AES greitesniam duomenų šifravimui.
- **Praktinis pavyzdys**: Kai svarstyklės pirmą kartą prisijungia prie centrinės POS sistemos, RSA gali būti naudojamas užtikrinti, kad abi pusės yra teisėtos ir gali pasitikėti viena kita. Tada sistema naudoja AES tolesnei komunikacijai.

TLS (Transport Layer Security)

TLS yra tinklo apsaugos protokolas, naudojamas apsaugoti duomenų perdavimą tarp įrenginių. Svarstyklės, turinčios Wi-Fi ar Ethernet ryšį su POS, galėtų naudoti TLS, kad apsaugotų duomenų srautą.

Bet kokio ryšio, naudojant Transport Layer Security (TLS), saugumas labai priklauso nuo pasirinktų šifrų rinkinių ir saugos parametrų. Šio straipsnio tikslas yra padėti jums priimti šiuos sprendimus, kad būtų užtikrintas kliento ir serverio ryšio konfidencialumas ir vientisumas. „Mozilla Operations Security“ (OpSec) komanda palaiko wiki įrašą su nuorodų serverių konfigūracijomis.

Transporto lygmens saugos (TLS) protokolas yra standartas, leidžiantis dviem tinklo programoms ar įrenginiams privačiai ir patikimai keistis informacija. Programos, kurios naudoja TLS, gali pasirinkti savo saugos parametrus, kurie gali turėti didelės įtakos duomenų saugumui ir patikimumui. Šiame straipsnyje apžvelgiama TLS ir kokie sprendimai, kuriuos reikia priimti, kai saugosite turinį.

- **Privalumai**: TLS užtikrina, kad visi duomenys, siunčiami per tinklą, yra šifruojami ir negali būti perimti ar pakeisti trečiosiomis šalimis.

- **Praktinis pavyzdys**: Jei svarstyklės siunčia svėrimo rezultatus per tinklą į centralizuotą serverį, TLS šifravimas užtikrina, kad jokie išoriniai asmenys negalės perimti ar pakeisti šių duomenų. Tai ypač svarbu, kai svarstyklės veikia kaip dalis IoT (daiktų interneto) infrastruktūros.

TLS teikia tris pagrindines paslaugas, kurios padeda užtikrinti su juo keičiamų duomenų saugumą:

Autentifikavimas
Autentifikavimas leidžia kiekvienai bendravimo šaliai patikrinti, ar kita šalis yra tokia, kokia ji teigia esanti.

Šifravimas
Duomenys yra užšifruojami perduodant tarp vartotojo agento ir serverio, kad būtų išvengta jų skaitymo ir interpretavimo neįgaliotiems asmenims.

Sąžiningumas
TLS užtikrina, kad tarp duomenų šifravimo, perdavimo ir iššifravimo jokia informacija nebūtų prarasta, sugadinta, sugadinta ar suklastota.

TLS ryšys prasideda rankos paspaudimo faze, kai klientas ir serveris susitaria dėl bendros paslapties ir deramasi dėl svarbių parametrų, pvz., šifravimo rinkinių. Kai parametrai ir duomenų mainų režimas, kai keičiamasi programos duomenimis, pvz., HTTP.

SSL (Secure Sockets Layer)

- SSL yra TLS pirmtakas, bet jis nebeturėtų būti naudojamas dėl žinomų saugumo spragų.
  SSLv2 ir SSLv3 yra dvi šio protokolo versijos (SSLv1 niekada nebuvo viešai išleistas). Po SSLv3 SSL buvo pervadintas į TLS.

- Rekomenduojama naudoti TLS vietoj SSL, nes TLS yra modernesnis ir saugesnis protokolas.

SSL sertifikatai (Secure Sockets Layer) tapo pagrindine svetainės saugumo dalimi. Jie užtikrina, kad svetainės galėtų saugiai ir privačiai perduoti duomenis į pradinį serverį, įjungdamos SSL/TLS šifravimą. Šie sertifikatai užšifruoja visas sąveikas naudodami viešąjį svetainės raktą ir jas iškoduoja naudodami sertifikato privatų raktą, kuris saugomas sertifikatą išdavusiame serveryje. Be šifravimo, šie sertifikatai taip pat padeda patikrinti svetainės tapatybę ir užtikrinti, kad vartotojas bendrautų su tinkamu serveriu. Šiame dokumente trumpai minima kliento ir serverio sąveika, pagrįsta HTTP protokolu. Jame pateikiamas išsamus Secure Socket Layer protokolo projektavimo ir įgyvendinimo paaiškinimas, po kurio aptariami su SSL susiję kriptografiniai pažeidžiamumai ir galimi jų pašalinimo būdai.

Ryšys visada prasideda rankos paspaudimu tarp kliento ir serverio. Šis rankos paspaudimas skirtas suteikti slaptą raktą tiek klientui, tiek serveriui, kuris bus naudojamas srautui šifruoti.

Iš tikrųjų pagrindinė paslaptis gaunama iš rankos paspaudimo, iš kurio gaunamas slaptasis raktas. OpenSSL šis master_secret saugomas SSL sesijoje SSL_SESSION.

Pradinis rankos paspaudimas gali užtikrinti serverio autentifikavimą, kliento autentifikavimą arba iš viso ne autentifikuoti.

Apibendrinimas

Duomenų šifravimas – tai procesas, kurio metu duomenys yra paverčiami į neskaitytą formą, siekiant apsaugoti juos nuo neteisėtos prieigos. Šifravimas naudoja raktus, kad paverstų informaciją į koduotą tekstą, ir tik turint teisingą raktą galima ją atkurti. Pagrindiniai šifravimo metodai yra **simetrinis**, pvz., **AES**, ir **asimetrinis**, pvz., **RSA**. Simetrinis šifravimas naudojamas, kai tas pats raktas tiek šifruoja, tiek dešifruoja duomenis, o asimetrinis naudoja viešąjį ir privatųjį raktą, suteikiant didesnį saugumą. Taip pat, naudojant **TLS** ir **SSL** protokolus, užtikrinamas saugus duomenų perdavimas internete.
Šifravimas yra esminis elementas tiek duomenų saugumui užtikrinti, tiek autentiškumo ir identifikavimo metodams, tokiems kaip **OAuth** ar **API rakto autentifikavimas**.

2. Autentifikacijos metodai

(OAuth, API key).

Autentifikacija – tai procesas, kurio metu patvirtinama naudotojo, įrenginio ar sistemos tapatybė, kai jie bando pasiekti tam tikrus išteklius. Autentifikacija užtikrina, kad tik įgalioti vartotojai, įrenginiai ar sistemos galėtų gauti prieigą prie svarbios informacijos ar valdyti sistemos funkcijas. Tai gali būti duomenys apie prekių kainas, svorius ar operacijų žurnalus ir pan.
Autentifikacija yra labai svarbi elektroninėse prekybinėse svarstyklėse, nes šie įrenginiai dažnai yra prijungti prie didesnių tinklų, įskaitant debesų kompiuterijos sistemas, skirtas realaus laiko duomenų dalijimuisi, nuotolinei priežiūrai ir atnaujinimams. Jei nėra patikimos autentifikacijos, įsilaužėliai galėtų manipuliuoti svarstyklių duomenimis, dėl to būtų padaryta finansinė žala, kiltų sukčiavimo grėsmė ir pakenktų klientų pasitikėjimui. Todėl tinkamų autentifikacijos mechanizmų diegimas padeda užtikrinti duomenų vientisumą, apsaugoti nuo neteisėtos prieigos ir atitikti teisės aktų reikalavimus.

OAuth autentifikacija

OAuth – atviroji autorizacija yra atvirasis protokolas, kuris leidžia trečiosioms šalims saugiai suteikti prieigą prie išteklių nesidalijant vartotojo prisijungimo duomenimis. Vietoje prisijungimo vardų ir slaptažodžių naudojami prieigos raktai (angl. *access tokens*), kurie suteikia galimybę pasiekti išteklius saugiai ir ribotai. Tokia priemonė plačiai taikoma interneto programėlėse, mobiliosiose programėlėse, daiktų interneto (IoT) įrenginiuose ir kitose technologijose

OAuth naudoja **prieigos raktų** (tokens) sistemą, leidžiančią įrenginiams ir programėlėms autentifikuoti vartotojus. Procesas apima šiuos pagrindinius vaidmenis:

- **Klientas**: Programėlė arba įrenginys, prašantis prieigos (pvz., elektroninės svarstyklės).
- **Išteklių savininkas**: Naudotojas arba subjektas, kuris turi teisę į išteklius (pvz., parduotuvės savininkas).
- **Autorizacijos serveris**: Tikrina naudotojo tapatybę ir išduoda prieigos raktą.
- **Išteklių serveris**: Serveris, kuriame saugomi ištekliai (pvz., debesų kompiuterijos sistema).

OAuth gali būti naudojamas norint suteikti prieigą prie **debesų paslaugų** (pvz., inventoriaus valdymo sistemų), be poreikio rankiniu būdu įvesti prisijungimo duomenis kiekviename įrenginyje.

Saugumo funkcijos:

- **Prieigos raktų galiojimo laikas**: Prieigos raktai galioja tik tam tikrą laiką, o tai sumažina riziką, jei raktas būtų pavogtas.
- **Atnaujinimo raktai**: Leidžia išduoti naujus prieigos raktus be pakartotinės autentifikacijos.
- **Prieigos apimties apribojimai**: Raktai gali suteikti prieigą tik prie tam tikrų išteklių ar funkcijų.

Privalumai	Trūkumai
Saugumas (nesikeičia slaptažodžiai)	Sudėtinga diegti ir konfigūruoti
Raktų galiojimo laikas sumažina pavojų	Rizika, kad raktas gali būti pavogtas
Leidžia riboti prieigą pagal apimtį	Reikalinga papildoma infrastruktūra

API raktų autentifikacija

API raktų autentifikavimas – tai paprastas būdas kontroliuoti prieigą prie API (programavimo sąsajų). API raktas yra unikalus identifikatorius, kurį klientas naudoja norėdamas prisijungti prie serverio ir atlikti užklausas. Kiekvieną kartą, kai programa kreipiasi į API, raktas siunčiamas kartu su užklausa, o serveris jį patikrina prieš suteikdamas prieigą.

Kaip veikia API: Įrenginys siunčia API raktą su kiekviena užklausa į serverį. Serveris patikrina raktą ir nustato, ar įrenginys turi teisę naudotis reikiamomis paslaugomis.

API raktai gali būti naudojami valdyti prieigą prie debesijos paslaugų, pavyzdžiui:

- **Programinės įrangos atnaujinimai**: Tik įgalioti įrenginiai gali atsisiųsti atnaujinimus.
- **Duomenų analizė**: Svarstyklės gali siųsti duomenis apie pardavimus į apskaitos platformas.

- **Įrenginio diagnostika**: Tik įgalioti technikai gali gauti diagnostinius duomenis.

Saugumo funkcijos:

- **Užklausų apribojimai**: Ribojamas užklausų skaičius per tam tikrą laiką.
- **IP adresų sąrašai**: Tik tam tikrų IP adresų užklausos yra leidžiamos.
- **Parašu pasirašytos užklausos**: Užklausos pasirašomos naudojant kriptografinius raktus.

Autentifikacija yra esminė **elektroninių prekybinių svarstyklių** apsaugos dalis. OAuth suteikia **didžiausią saugumą** dėl riboto prieigos raktų galiojimo ir prieigos ribojimo. API raktai yra **paprastesnis sprendimas**, bet sukelia daugiau saugumo rizikos, jei raktas patenka į netinkamas rankas. Tinkamai įdiegus OAuth ar API raktus, svarstyklės gali užtikrinti **saugų ryšį su debesų paslaugomis**, taip apsaugant verslo ir klientų duomenis.

Privalumai ir trūkumai

Privalumai	Trūkumai
Paprasta naudoti ir įdiegti	Jei raktas pavogiamas, juo galima piktnaudžiauti
Greitas autentifikavimo procesas	Reikalingas raktų tvarkymas ir keitimas
Naudojamas mašinos-mašinai (M2M) autentifikacijai	API raktai nepasibaigia galiojimo laiko atžvilgiu

Apibendrinimas:

Nuo API raktų prie OAuth 2.0, siekiant geresnio saugumo

Kaip ir daugelyje technologijų, naudojimo paprastumas susijęs su didesne rizika. Skirtingai nei trumpalaikiai OAuth 2.0 prieigos žetonai, API raktai paprastai galioja ilgą laiką. Ypač pavojinga, kai API raktai įtraukiami į užklausų parametrus. Naršyklės išsaugo URL istoriją, o žurnalų (log) sistemos gali įrašyti visą URL adresą, dėl to raktas tampa labiau pažeidžiamas nei naudojant HTTP antraštes (*headers*).

Saugumo problemos, susijusios su API raktais
API raktai gali patekti į netinkamas rankas, todėl galima neteisėta prieiga. Pavyzdžiui, API raktai gali netyčia būti įtraukti į versijų valdymo sistemas (pvz., „GitHub“), todėl prie rakto gali prisijungti kiti naudotojai, kurie peržiūri kodą arba jo istoriją. Kadangi API raktai galioja ilgai, rakto nutekėjimo rizika tampa didelė. Jei raktas yra pavogtas, jis lieka aktyvus tol, kol jis rankiniu būdu atšaukiamas arba pasibaigia jo galiojimo laikas. Dėl šios priežasties pavogtas API raktas gali būti naudojamas neteisėtai prieigai prie sistemų ir paslaugų.

Nuorodos

https://www.nist.gov/publications/advanced-encryption-standard-aes

https://en.wikibooks.org/wiki/Cryptography/RSA

https://developer.mozilla.org/en-US/docs/Web/Security/Transport_Layer_Security

https://wiki.openssl.org/index.php/SSL_and_TLS_Protocols

https://www.rfc-editor.org/rfc/rfc6749

https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

https://auth0.com/blog/why-migrate-from-api-keys-to-oauth2-access-tokens/

https://auth0.com/blog/why-migrate-from-api-keys-to-oauth2-access-tokens/