KRIPTOGRAFIJA U ORACLE BAZI

Zlatko Sirotić, univ.spec.inf.
Istra informatički inženjering d.o.o.
Pula

Sigurnosne osobine (poruka) informacija

- **povjerljivost** (tajnost, engl. confidentiality): poruku koju osoba A šalje osobi B ne može pročitati nitko treći;
- **integritet** (besprijekornost, engl. integrity): osoba B je sigurna da poruka koju je poslala osoba A nije promijenjena (od neke treće osobe);
- **raspoloživost** (engl. availability): poruke moraju uvijek biti na raspolaganju ovlaštenim osobama, unatoč mogućim nepogodama / nesrećama ili zlonamjernim napadima;
- **autentičnost** (vjerodostojnost, engl. authenticity): osoba B zna da je samo osoba A mogla poslati poruku koju je primila;
- **neporecivost** (nepobitnost, engl. non-repudiation): osoba A ne može zanijekati da je poslala poruku koju je osoba B primila (ako ju zaista poslala).

Standardni kriptosustavi

- **Osnovna zamisao: napadač može znati kako radi algoritam, ali ako ne zna ključ(eve), ne može ništa**.
- 1976. je prihvaćen kao standard kriptosustav koji je dobio ime **Data Encryption Standard** (DES), a koristi 56-bitne ključeve.
- 1998. se prvi put javno pokazalo da je DES nesiguran.
- 2000. je izabran nasljednik DES-a - **Advanced Encryption Standard** (AES), a koristi 128, 192, ili 256-bitne ključeve.
- Navedeni kriptosustavi su **simetrični kriptosustavi**, tj. koriste isti ključ i u procesu kriptiranja i u procesu dekriptiranja, što znači da je ključ poznat i pošiljatelju i primatelju.
- **Simetrični ključ** ima manu da se mora (na siguran način) razmijeniti među sudionicima u komunikaciji.

Simetrični i asimetrični kriptosustavi

- Zato su razvijeni **asimetrični kriptosustavi** (ili kriptosustavi s javnim ključem). Glavna ideja je da se konstruiraju kriptosustavi kod kojih je na temelju poznavanja funkcije kriptiranja praktički nemoguće izračunati funkciju dekriptiranja.
- Svaki sudionik u komuniciranju ima svoj **javni ključ**, koji može biti poznat svima, i svoj **tajni ključ**, koji je poznat samo njemu.
- Kriptiranje se radi javnim ključem primatelja, a dekriptiranje tajnim ključem primatelja (kod digitalnog potpisa je drugačije!), tako da samo primatelj može pročitati poruku.
- Asimetrični kriptosustavi nisu zamjena za simetrične, već jedni druge nadopunjuju, čime se dobivaju **hibridni kriptosustavi**.

Usporedba između simetričnih (SK) i asimetričnih kriptosustava (AK)

- **sigurnost**: kod AK, samo privatni ključ mora biti tajan, a javni ključ se može slobodno distribuirati; kod SK, zajednički (simetrični) ključ mora biti poznat dvjema (ili više) sudionicima u komunikaciji;
- **dugotrajnost ključa**: kod AK, isti par ključeva se može koristiti dugo vremena, a kod SK je poželjno mijenjati ključ kod svake sesije;
- **upravljanje ključevima**: ako je n broj sudionika u međusobnoj komunikaciji, SK traži ukupno n * (n – 1) / 2 ključeva, pri čemu svaki sudionik mora kod sebe pamtiti (n – 1) ključ; kod AK, treba ukupno n javnih ključeva, koji se mogu slobodno pohraniti u nekoj bazi javnih ključeva, a svaki sudionik mora pamtiti samo svoj tajni ključ;

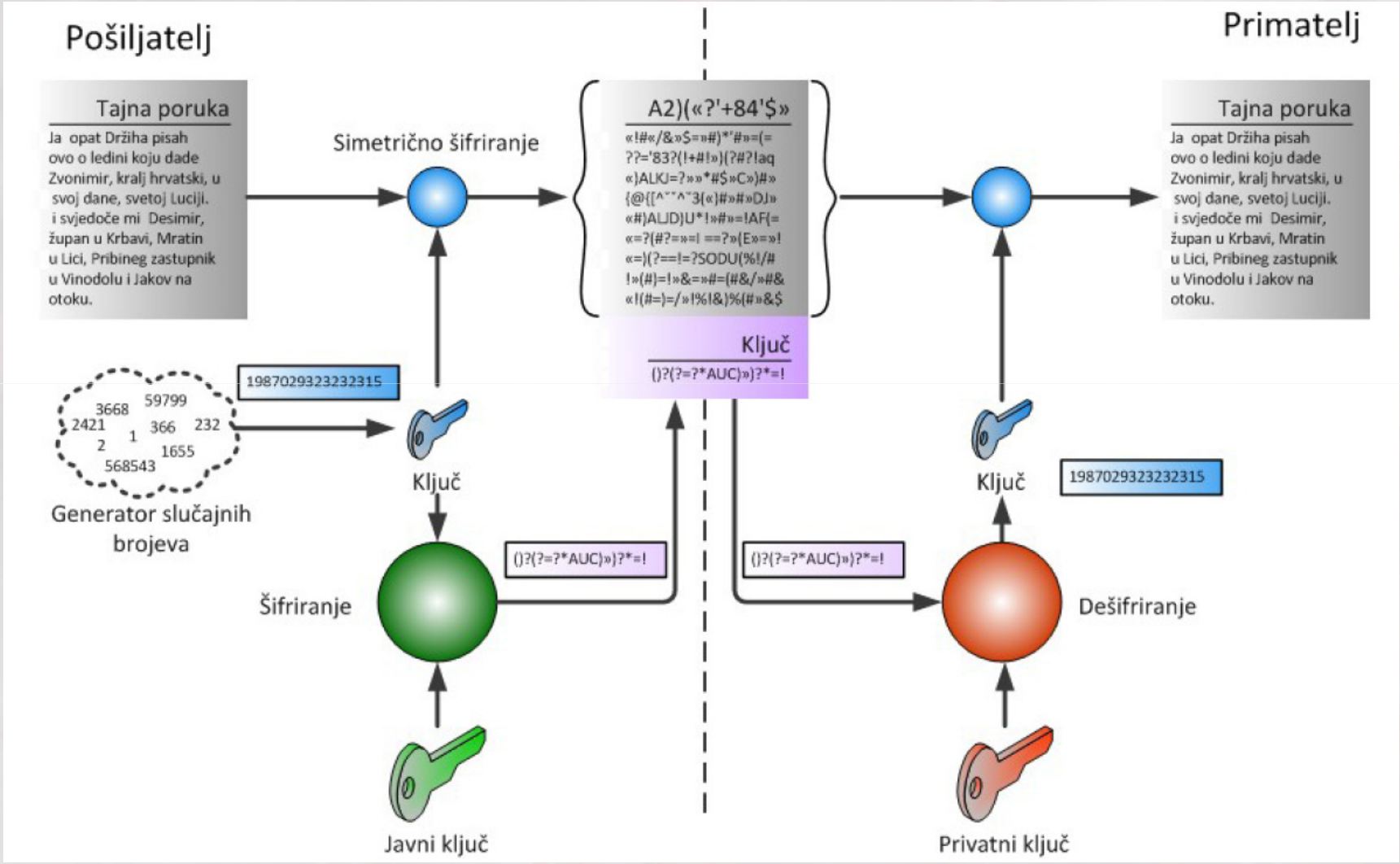
Usporedba između simetričnih (SK) i asimetričnih kriptosustava (AK)

- **razmjena ključeva**: kod AK se ne moraju razmjenjivati ključevi između sudionika; razmjena ključeva kod SK je obavezna; budući da je to opasno, upravo se AK koristi za sigurnu razmjenu simetričnog ključa, čime se onda omogućava SK (zajedno je to hibridni kriptosustav);
- **efikasnost**: AK su značajno sporiji od SK; npr. RSA kriptosustav je oko 1000 puta sporiji od DES kriptosustava;
- **veličina ključeva**: ključevi za AK su značajno veći od ključeva za SK (za istu razinu sigurnosti); npr. privatni ključ u RSA mora imati barem 1024 bita, dok je kod većine SK 128 bitova dovoljno.

Neki važni pojmovi kod primjene asimetričnih kriptosustava

- **Sažetak** (hash, digest): za razliku od funkcije kriptiranja, koja dva različita X-a preslikava u dva različita Y-a, funkcija sažetka može preslikavati više X-eva u isti Y, što znači da nema inverznu funkciju. Koristi se kao pomoćno sredstvo, npr. kod spremanja hash-a lozinki (umjesto samih lozinki) u bazu podataka, kod digitalnog potpisa i dr.
- **Digitalna omotnica**: njome se postiže **povjerljivost** poruke. Za to bi se mogli koristiti i samo simetrični ključevi. No, budući da je razmjena nekriptiranih simetričnih ključeva opasna, pošiljatelj kreira novi simetrični ključ (samo za tu sesiju), pomoću tog ključa kriptira izvornu poruku, a onda simetrični ključ kriptira pomoću javnog ključa primatelja, te onda šalje oboje - poruku kriptiranu simetričnim ključem i kriptirani simetrični ključ.

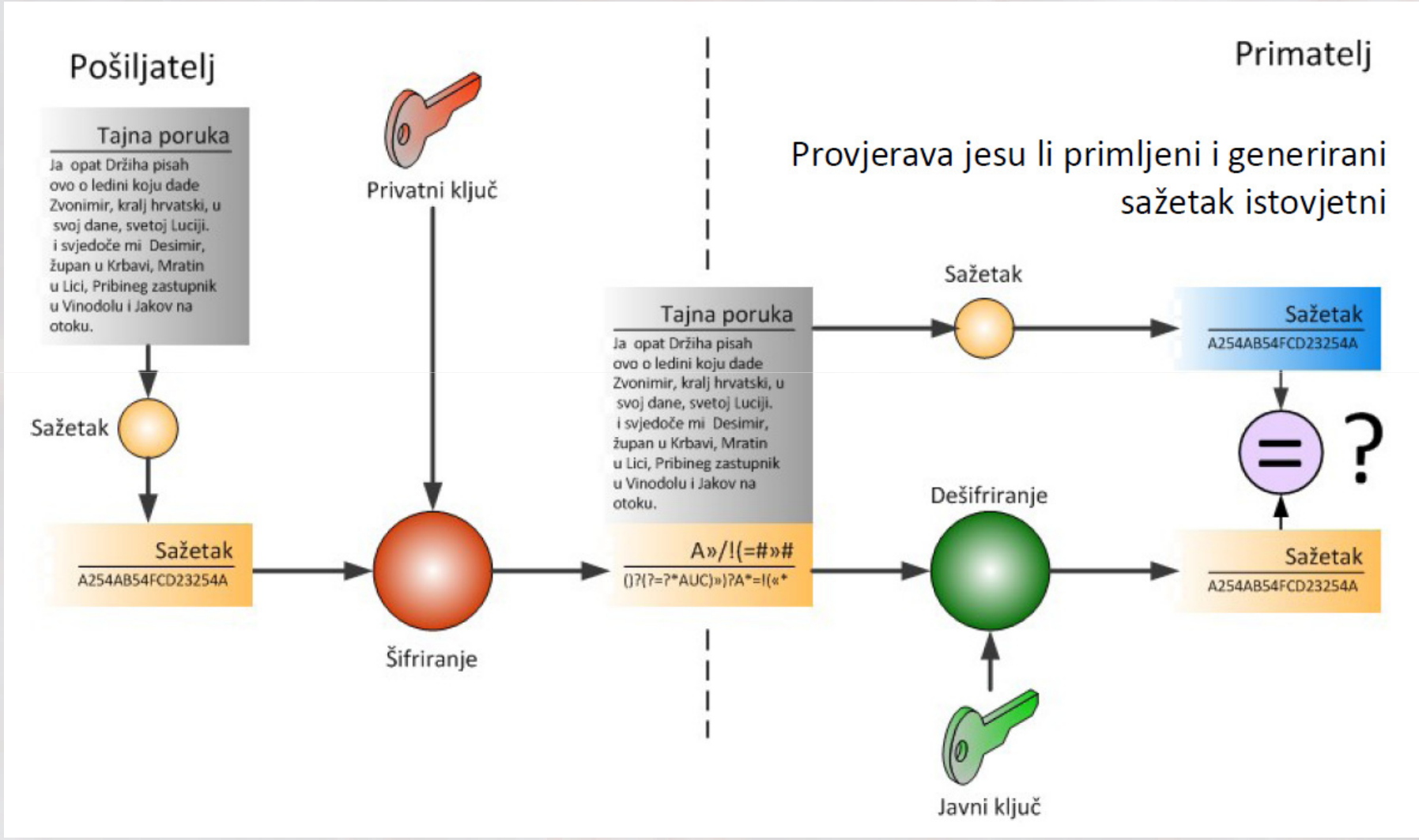
Digitalna omotnica



Digitalni potpis

- **Digitalni potpis**: njime se postiže **integritet i neporecivost** poruke. Pošiljatelj uz izvornu (nekriptiranu) poruku šalje i izvornu poruku kriptiranu pomoću svog tajnog ključa (**suprotno od uobičajene sheme asimetričnog kriptiranja**, npr. one kod digitalne omotnice!).
- Primatelj pomoću javnog ključa pošiljatelja dekriptira poruku i uspoređuje ju s izvornom. Budući da samo pošiljatelj zna svoj privatni ključ, osiguran je integritet i neporecivost poruke.
- U praksi se najčešće ne kriptira izvorna poruka, već sažetak (hash) izvorne poruke, a razlozi su (barem) sljedeći: sažetak je kraći od izvorne poruke, pa se brže kriptira/dekriptira, ukupna poruka je kraća, izvorna poruka se može isto kriptirati (što onda čini digitalni pečat).

Digitalni potpis



Digitalni pečat i digitalni certifikat

- **Digitalni pečat**: njime se postiže **povjerljivost, integritet i neporecivost poruke**. Digitalni pečat je kombinacija digitalne omotnice i digitalnog potpisa, tj. digitalni pečat je digitalno potpisana digitalna omotnica.
- **Digitalni certifikat**: njime se postiže **autentičnost** poruke. Digitalni certifikat je (specijalna) poruka koja je (najčešće) digitalno potpisana od **certifikacijskog centra** (engl. Certification Authority, CA), kome se po definiciji vjeruje.
- Sadrži barem par podataka - identifikator i javni ključ pošiljatelja, ali obično sadrži i rok valjanosti, podatak o algoritmima za sažimanje i kriptiranje kod potpisivanja, i dr.
- Pomoću digitalnog certifikata primatelj može provjeriti da li je dobiveni javni ključ jednak onome koji piše u digitalnom certifikatu, tj. da li je pošiljatelj autentičan.

RSA (Rivest, Shamir, Adleman) asimetrični kriptosustav

- **RSA** je prvi kriptosustav s javnim ključem (objavljen 1977.)
- Postupak izgradnje RSA kriptosustava:

1. Odaberu se **dva tajna velika prosta broja p i q**, svaki veličine barem 512 bitova (a preporučljivo je i 1024).
2. Izračunava se umnožak $n = p\ q$.
3. Izračunava se umnožak $\varphi(n) = (p - 1)\ (q - 1)$.
4. Odabere se broj $e < \varphi(n)$ i relativno prost u odnosu na $\varphi(n)$.
5. Izračunava se broj $d < \varphi(n)$ tako da bude:
   $e\ d \equiv 1 \pmod{\varphi(n)}$, što je isto kao: $e\ d \equiv k\ \varphi(n) + 1$.
6. Brojevi *e* i *n* se obznanjuju i čine javni ključ.
7. Brojevi *p, q, d* su tajni, pri čemu je *d* tajni ključ (zapravo, dio ključa – ključ čine *d* i *n*, koji je javno poznat).

Neka pitanja i odgovori vezani za implementaciju RSA

- Važna praktična **pitanja** su:

1. Kako dobiti dva velika prosta broja *p* i *q*?
2. Kako odabrati javni ključ *e*?
3. Kako izračunati tajni ključ *d*?

- **Odgovori** mogu biti sljedeći:

1. Teškoća faktorizacije velikih prirodnih brojeva osnova je za sigurnost RSA kriptosustava, pa ne možemo faktorizacijom naći dokazivo proste brojeve p i q. Zato se nalaze **vjerojatno prosti brojevi**, npr. pomoću heurističkog ispitivanja po Miller-Rabinu (prikazano u prilozima A i B).
2. Za *e* se danas preporučuje staviti 65537 (= 2**16 - 1).
3. Tajni ključ *d* može se izračunati na temelju proširenog Euklidovog algoritma (prilog A i B).

Kriptiranje podataka u tranzitu

- Kriptiranje podataka koji su u tranzitu nije vezano samo za baze podataka i takvo se kriptiranje uglavnom koristilo i prije kriptiranja podataka koji stoje u bazi.
- Kada je riječ o Oracle bazi, postoje rješenja kriptiranja podataka u tranzitu koja su vezana uz Oracle bazu, ali postoje i rješenja nezavisna od Oracle baze.
- Obje varijante kriptiranja podataka u tranzitu imaju isti cilj, zaštiti tri vrste paketa koji se prenose mrežom:

1. **paketi koji služe za inicijaciju sesije** između klijenta i servera baze podataka;
2. **paketi koje klijent šalje bazi**, uključujući SQL naredbe;
3. **paketi koje baza šalje klijentu** (kao odgovore).

Kriptiranje podataka u tranzitu

- Zašto je važno zaštiti i 1.vrstu paketa, iako Oracle baza tokom logon procesa **uvijek prenosi lozinku u kriptiranom obliku** (bez obzira da li je promet mrežom kriptiran ili nije)?
- Ako promet mrežom nije kriptiran, tokom postupka identifikacije/autentikacije na Oracle bazu može se pojaviti jedna **potencijalna ranjivost – ako napadač zna hash lozinke i prisluškuje mrežu**.
- Proces kriptiranja podataka u tranzitu može se raditi pomoću dva Oracle rješenja (uz Advanced Security Options): **Network Data Encryption** (NDE) i sa **Secure Socket Layer** (SSL).
- Moguće je koristiti i nezavisna softverska ili hardverska rješenja, npr. **Secure Shell** (SSH), **Internet Security Protocol** (IPSec), **hardverske akceleratore**.

Kriptiranje podataka na diskovima

- Bez obzira na sve druge načine osiguravanja podataka u bazi, vidjelo se da su **jako opasni oni slučajevi u kojima se ukradu podaci na diskovima**, pa onda napadači imaju dovoljno vremena i mogućnosti da pristupe osjetljivim podacima i kada nemaju direktan pristup kroz bazu podataka.
- Poduzeća (i druge organizacije) danas pristupaju kriptiranju podataka na diskovima najčešće iz tri razloga:

1. da bi **spriječili lošu reputaciju** koju dobiju nakon što se otkrije da su napadači otuđili osjetljive podatke;
2. da izbjegnu ili **smanje financijske troškove** koje imaju zbog odšteta koje su dužne nadoknaditi oštećenim stranama;
3. da budu u skladu s različitim **regulatornim propisima** o zaštiti podataka, naročito zaštiti privatnih podataka.

Kriptiranju podataka na diskovima može se pristupiti na tri načina

- **"ručno" kriptiranje kroz aplikaciju, programiranjem**; dobra strana je da je to vrlo fleksibilno; no, ne može se izvesti na brzinu, jer su aplikacije kompleksne; kriptiranje uvijek pogoršava performanse, a kriptiranje vlastitim programiranjem najčešće je najsporije; najveća mana je da se ne može na adekvatan način zaštiti ključeve za kriptiranje;
- **"automatsko" kriptiranje kroz sustav za upravljanje bazom podataka**; bolje je od prethodne varijante; no, za razliku od programiranja kroz aplikaciju, "transparentno" kriptiranje ne pomaže kod sakrivanja podataka od onih koji imaju pristup bazi;
- **"automatsko" kriptiranje na razini diskovnog sustava**; isto ne može sakriti podatke od onih koji imaju pristup bazi.

Kako raditi sa ključevima

❖ **Varijante broja ključeva - po jedan ključ za**:

- kriptiranje podataka u cijeloj bazi podataka;
- svaku tablicu baze podataka (ili npr. za tablespace);
- svaki stupac koji se kriptira;
- svaki redak koji se kriptira.

❖ **Gdje se sprema ključ (ključevi)**:

- u bazu podataka;
- u datoteke izvan baze podataka (na serveru baze podataka, aplikacijskom serveru ili klijentu);
- u programski kod aplikacije; ovo je najlošije, jer postoje različiti načini dekodiranja (disasembliranja) programa i dolaska do ključa koji je sakriven u programu.

Kriptiranje podataka pomoću paketa DBMS_CRYPTO

- „Ručno" kriptiranje podataka u bazi vrlo je zahtjevno. Traži puno vremena za izvedbu, ima najlošije performanse, omogućuje ostavljanje "rupa" u kriptiranju, a najveći je problem kod njega - kako upravljati ključevima za kriptiranje.
- No, ponekad se mora koristiti "ručno" kriptiranje, npr.:

- **na raspolaganju je samo Standard edicija baze**, koja nema mogućnosti za "transparentno" kriptiranje (koje je dodatna opcija u Enterprise ediciji);
- **želi se kriptirati (neke) podatke i korisnicima baze** - "transparentno" kriptiranje ne može sakriti podatke korisnicima baze.

- Za "ručno" kriptiranje koristi se paket (na bazi) DBMS_CRYPTO.

Kriptiranje podataka pomoću paketa DBMS_CRYPTO

- podržava kriptografske algoritme: DES, 3DES, 3DES_2KEY, AES, RC4; preporučljivo je koristiti algoritam AES, koji omogućava kriptiranje sa 128, 192 ili 256-bitnim ključevima;
- kriptografski hash algoritmi: MD5, SHA-1, MD4;
- MAC algoritmi (algoritmi za autentikaciju poruka) HMAC_MD5, HMAC_SH1;
- kriptografski generatori pseudoslučajnih brojeva u formatu RAW, NUMBER, BINARY_INTEGER;
- kriptiranje blokova podataka: kriptiranje ulančavanjem blokova CBC (Cipher Block Chaining, CBC), CFB (Cipher Feedback Chaining) i OFB (Output Feedback Chaining)
- ispunjavanje (padding) podataka nulama ili pomoću metode PBCS5 (preporučljivo).

Problematika ključeva i Oracle Wallet

- Najbolje je da korisnik predaje ključeve pomoću hardverskog modula za sigurnost (**Hardware Security Module**, HSM).
- Ako toga nema, problem upravljanja ključevima za kriptiranje nije jednostavno riješiti. Oracle u Enterprise ediciji, sa ASO opcijom, nudi **Oracle Wallet**.
- **Wallet je datoteka (ili više njih) izvan baze**, koja sadrži ključeve za kriptiranje/dekriptiranje. Najčešće je wallet datoteka na razini OS-a, čiji je sadržaj kriptiran lozinkom (iako može biti spremljena npr. i u HSM modulu, što je najbolje).
- Prije nego se otvori baza podataka, administrator baze podataka (ili netko drugi tko zna lozinku) otvori wallet, a onda baza podataka može koristiti ključeve koji su pohranjeni u wallet-u za kriptiranje/dekriptiranje podataka u bazi.

Oracle Wallet – definiranje (u SQLNET.ORA) i korištenje

- ENCRYPTION_WALLET_LOCATION=
  (SOURCE=(METHOD=FILE)
  (METHOD_DATA=(DIRECTORY=…/network/admin/))
)
- Kreiranje walleta:
  ALTER SYSTEM SET ENCRYPTION KEY
  IDENTIFIED BY lozinka_za_wallet;
- Nakon otvaranja baze mora se pokrenuti otvaranje walleta:
  ALTER SYSTEM SET ENCRYPTION WALLET **OPEN**
  IDENTIFIED BY lozinka_za_wallet;
- Privremeno zatvaranje walleta:
  ALTER SYSTEM SET ENCRYPTION WALLET **CLOSE**
  IDENTIFIED BY lozinka_za_wallet;

Kriptiranje podataka pomoću TDE Column Encryption

- Kao i Oracle Wallet, i TDE CS se može koristiti samo u EE ediciji baze, pri čemu je (isto) potrebna i ASO opcija.
- TDE CS služi **za zaštitu određenih stupaca tablica podataka na disku**, u slučaju da netko neovlašteno dođe do tih podataka izvan baze.
- No, ti stupci **za korisnike baze uvijek transparentno dekriptirani**. Treba napomenuti da stupci nisu kriptirani samo u tablicama, već i REDO, UNDO i TEMP strukturama.
- Može se kreirati novi kriptirani stupac postojeće tablice, ili kriptirati postojeći stupac, ili kreirati potpuno nova tablica:

```
CREATE TABLE t (
        c1 varchar2(30),
        c2 varchar2(30) ENCRYPT);
```

Kriptiranje podataka pomoću TDE Column Encryption

- **Čini se kao da podaci nisu kriptirani**
  - zato jer ih baza sama transparentno dekriptira:

```
INSERT INTO t VALUES
      ( '***ovo NIJE kriptirano***', '***ovo JE kriptirano***');
SELECT * from t;
      C1 C2
      ---------------------------- ----------------------------
      ***ovo NIJE kriptirano*** ***ovo JE kriptirano***
```

- **Ako se privremeno zatvori wallet**:

```
INSERT INTO t VALUES
      ( '***ovo NIJE kriptirano***', '***ovo JE kriptirano***');
ili SELECT c2 FROM t; -- pokušaj čitanja kriptiranog stupca
ERROR at line 1: ORA-28365: wallet is not open
```

TDE Column Encryption - mane i ograničenja

- **Smanjene perfomanse** u odnosu na bazu bez kriptiranja. Jedan od razloga je i taj što se podaci čuvaju u kriptiranom obliku i u memoriji (tj. u SGA strukturi baze).
- Zbog potrebe da se kriptirani podaci zaokruže na višekratnik od 16 bajtova, a i zbog dodavanja SALT podataka, TDE CS **troši više prostora na disku** (i u SGA dijelu memorije).
- Ako kriptirani stupac ima indeks, tada i indeks treba biti kriptiran, pa onda kriptiranje **ne može koristiti SALT opciju**.
- Kriptirani indeksi se mogu koristiti samo za pretragu određene vrijednosti, ali **ne i za pretragu raspona vrijednosti**.
- Kriptirani stupci se **ne mogu koristiti za funkcijske indekse**.
- Nad kriptiranim stupcima **ne mogu se raditi vanjski ključevi**.

Kriptiranje podataka pomoću TDE Tablespace Encryption

- TDE Tablespace Security (kao i TDE CS) može se koristiti samo u EE ediciji baze, pri čemu je potrebna i ASO opcija.
- Kako samo ime kaže, ovdje se **ne kriptira samo određeni stupac, već cijeli tablespace**.
- Za razliku od TDE CS, kod TDE TS nije moguće kriptirati postojeću tablicu ili postojeći tablespace, već je **potrebno kreirati novi kriptirani tablespace** i onda u njega kopirati postojeće tablice koje želimo kriptirati.
- Kreiranje kriptiranog tablespace-a radi se sa:

```
CREATE TABLESPACE kriptirani_tablespace
      DATAFILE ...
      ENCRYPTION DEFAULT STORAGE (ENCRYPT);
```

Kriptiranje podataka pomoću TDE Tablespace Encryption

- I ovdje je moguće odabrati algoritam za enkripciju. **Default je AES128**, a ne AES192 kao kod TDE CS, jer je ovdje količina podataka koja se kriptira/dekriptira puno veća - cijeli tablespace, a ne pojedini stupac/stupci.
- Svaka tablica koja se kreira u kriptiranom tablespace-u bit će u cijelosti kriptirana. Kriptiranje se radi na razini bloka tablice, a ne retka ili stupca, **pa nema potrebe za zaokruživanjem podataka na višekratnik od 16 bajtova**, jer je blok podataka sam po sebi višekratnik od 16 bajtova.
- Također, budući da je svaki blok podataka jedinstven, **nije potreban SALT podatak**.
- Rezultat je taj da TDE Tablespace Security **ne zauzima dodatni prostor na disku**.

Kriptiranje podataka pomoću TDE Tablespace Encryption

- **TDE TS kriptira podatke prije slanja podataka iz SGA na disk**, odnosno dekriptira ih u obrnutom smjeru.
- To znači da su **podaci u SGA u nekriptiranom obliku**, što predstavlja malu potencijalnu ranjivost u slučaju da netko neovlašten može
- Prednost je u tome da se **čitanje/upis u SGA radi brzo**.
- Podaci u REDO, UNDO i TEMP pomoćnim strukturama na disku su kriptirani, isto kao i kod TDE CS.
- Nakon kopiranja nekriptirane tablice iz nekriptiranog tablespace-a u kriptirani tablespace, te brisanja nekriptirane tablice, nije sigurno da više nema nekriptiranih podataka. Naime, **ti podaci ostaju određeno vrijeme** u REDO, UNDO i TEMP pomoćnim strukturama.

Zaključak – zašto se kriptiraju podaci

- Kriptiranje podataka važno je **zbog očuvanja privatnosti, integriteta i autentičnosti poruka (informacija)**, te zbog osiguranja **neporecivosti** slanja poruke.
- Za kriptiranje se koriste **simetrični i asimetrični kriptosustavi**. Simetrični kriptosustavi su puno brži (oko 1000 puta), ali asimetrični kriptosustavi omogućavaju razmjenu ključeva među velikim brojem sudionika.
- Važno je **kriptirati podatke u tranzitu, ali i podatke koji se nalaze na medijima** (najčešće diskovima). Podatke na medijima može se kriptirati nezavisno od baze podataka, ili ih se može kriptirati unutar baze podataka.
- **Oracle baza** direktno omogućava **simetrično kriptiranje**.

Zaključak - kriptografske mogućnosti različitih edicija Oracle baze 11g

- **Standard Edition** ima kriptiranje pomoću paketa DBMS_CRYPTO (programiranjem); njime se može postići sve, pa i sakrivanje podataka od (nekih) korisnika u bazi, ali uz lošije performanse i uz otvoreni problem spremanja ključeva za kriptiranje;
- **Enterprise Edition sa dodatnom opcijom Advanced Security (ASO)** ima transparentno kriptiranje podataka (TDE); TDE ne služi za sakrivanje podataka od legalnih korisnika baze, već samo za zaštitu podataka u slučaju krađe cijelog računala ili sustava za pohranu podataka;
- ASO opcija omogućava i dvije varijante kriptiranje podataka u tranzitu - jedno je rješenje Network Data Encryption (NDE), a drugo se zasniva na Secure Socket Layer (SSL).