Bezpečnost v databázích

bezpečnostní problémy, se kterými se potýkají databázové systémy jsou obdobné, jako problémy operačních systémů

- fyzická integrita databáze - odolnost proti výpadkům napájení, zotavení z poškození
- logická integrita databáze - musí být zachována struktura dat a vzájemné vazby
- elementární integrita - data obsažená v jednotlivých položkách jsou korektní
- 1auditabilita - kdo a jak přistupoval k položkám v databázi
- kontrola přístupu - kdo co může s čím dělat
- autentizace uživatelů - každý, komu je povolen přístup musí být pozitivně identifikován
- dostupnost - uživatelé mohou přistupovat k datům tak, jak jsou oprávněni

Integrita báze dat

správce báze musí zajistit, že změny dat mohou provádět pouze oprávnění uživatelé
systém musí obsahovat prostředky překonávající nedostupnost položky nebo dokonce celé báze, z hlediska OS a správce systému jde o ochranu relevantních souborů a programů, zálohy, kontroly zařízení atp.
z pohledu SŘBD přistupuje systém transakcí a logů, umožňující rekonstruovat stav databáze

Elementární integrita

autorizovaní uživatelé mohou vkládat data, ale činí chyby - ty musí SŘBD zachycovat a vyžádat si opravu
metody:

- field checks - test vhodnosti vkládaných dat: zda je to číslo, zda jde o jméno, …
- kontrola přístupu - mechanismus řešící kdo co může měnit, jak naložit s kolizními případy, následnost úprav
- log změn - záznam o všech provedených změnách, obsahuje původní a novou hodnotu
- kontrola čtyř očí
- vícenásobné pořízení dat

Auditabilita

je třeba vést záznamy o tom, kdo co dělal s kterými položkami - nejen pro to, abychom byli schopni sledovat přístupy a změny, ale i pro dlouhodobé sledování uživatelů a následné rozhodování, zda vyhovět žádosti
je nutné zvolit vhodnou granularitu - bloky, záznamy, položky
zde přistupuje tzv. *pass through problem* - uživatel smí přistupovat k objektu ale tento mu nesmí být předán (např. při vyhledávání)
uživatel může zjistit hodnotu položky i bez přímého dotazu - nestačí log žádostí o přístup k odhadu toho, co ví

Autentizace uživatelů

SŘBD potřebuje přesně vědět, komu odpovídá
protože však zpravidla běží jako uživatelský proces, nemá spolehlivé spojení s jádrem OS a tedy musí provádět vlastní autentizaci

Dostupnost

SŘBD má vlastnosti systému a aplikačního programu zároveň - běží jako program a používá služeb systému, pro mnoho uživatelů je však jediným pracovním prostředím
musí rozhodovat současné žádosti více uživatelů
musí být schopen odepřít poskytnutí i nechráněných dat aby nedošlo ke kompromitaci utajovaných informací

Spolehlivost a integrita

v prostředí databází jsou tyto pojmy ještě důležitější, než obvykle
jde nám o zachování těchto tří vlastností:

1. integrita databáze - celková správnost, ochrana před technickými závadami a poničením globálních struktur
2. elementární integrita - že změny a záznamy mohou provádět pouze autorizované entity
3. elementární správnost - že jsou přijata jen korektní data odpovídající typem, hodnotou, …

Dále se budeme zabývat mechanismy pro udržování těchto vlastností.
- tyto nejsou absolutní, např. nemohou zabránit oprávněným entitám vkládat nesprávná, leč typem a ostatními atributy akceptovatelná data

Ochrana poskytovaná OS

DBS jsou soubory a programy - tedy spadají pod standardní obranné mechanismy OS:
ochrana souborů, kontrola přístupu, zálohy, testy integrity na úrovni systému, …

Dvoufázový update

problémem je selhání výpočetního systému během provádění modifikace dat
proces modifikace rozdělíme na dvě fáze:

1. v průběhu první fáze - *záměr* (intent) se provede načtení relevantních dat, uzamčení záznamů, vytvoření pomocných záznamů a kalkulace výsledků
   poslední událostí první fáze je operace *commit*
2. v rámci druhé fáze jsou prováděny trvalé změny dat s ohledem na data získaná v první fázi (= zapsání výsledků)

Pokud některá z fází nedoběhne, může být snadno opakována, po ukončení fáze je systém konzistentní.

Třífázový update

řešení zápisů a čtení z distribované DB
oproti dvoufázovému update definuje ještě další „nultou fázi“ modifikace
0. ustanovení kvóra – zjišťuje se, je-li k dispozici dostatečný počet instancí distribuované databáze pro provedení operace

kvórum pro čtení – počet instancí, které musí souhlasit s provedením čtecí operace
kvórum pro zápis – počet instancí databáze, které musí souhlasit s provedením zápisu
součet obou kvór musí být větší než celkový počet instancí , kvórum pro zápis > 1/2

Cílem je zabránit stavu, kdy by část distribuované databáze poskytovala zastaralá data, resp. došlo k „rozdvojení“ vývoje datového obsahu

Redundance / Vnitřní konzistence

databáze často obsahují různé redundantní informace za účelem odhalení chyb, zvýšení spolehlivosti, nebo docílení potřebné rychlosti

Detekční a samoopravné kódy

jde o vhodné způsoby zakódování informací přidáním vhodné redundance tak, aby bylo s co možná největší pravděpodobností detekovat náhodné změny
samoopravné kódy mají navíc schopnost lokalizovat a vyčíslit jisté množství chyb, takže mohou být opraveny
vždy při ukládání je záznam zakódován, při načítání je kontrolována jeho správnost
k dispozici je celá ředa kódů pro různé účely

Stínové záznamy (Shadow fields)

vybrané atributy nebo věty jsou uloženy v několika kopiích

v případě nedostupnosti nebo chyby v originálu je použita kopie
metoda účinná leč náročná na prostor

Zotavení

SŘBD musí vést log o všech akcích, zejména o změnách
v případě havárie potom na základě těchto záznamů a vhodné záložní kopie znovu vygeneruje aktuální stav

Paralelismus / Konzistence

SŘBD musí zajistit současný přístup více uživatelů, vyřešit konflikty plynoucí z situací, kdy dochází k současnému zápisu více hodnot do stejné položky, nebo zápisu závisejícím po předchozím čtení položky

Monitory

jednotky SŘBD zajišťující strukturální integritu databáze - testují, zda vkládaná data typově odpovídají, zda jsou konzistentní s ostatnímy daty v systému atd.

Porovnání mezí

vkládaná hodnota je testována na příslušnost do jistého intervalu, meze intervalu mohou být určeny i dosti komplikovanou funkcí závisící na jiných hodnotách v databázi
tento test může být rovněž použit pokud je podezření, že uložená data jsou poškozena

Stavová omezení

popisují podmínky, které musí celá databáze splňovat
nejsou-li stavová omezení splněna, jsou data v databázi vadná
např. ve skladu nelze mít -5,1 rohlíku

Tranzitivní omezení

jsou omezení, které musí splňovat obsah databáze před provedením určité operace
např. před prodejem 1 rohlíku nesmí být sklad rohlíků prázdný

Senzitivní data

… data, která by neměla být veřejně známa
data se stávají senzitivními z mnoha různých důvodů:

- *přirozeně senzitivní* (inherently s.) - informace sama o sobě je utajovaná (plat toho kterého ministra)
- *ze senzitivního zdroje* - např. pokud je z informace patrné, kdo ji poskytl

- *deklarované jako senzitivní* - správce, nebo majitel DB prohlásí informaci za utajovanou
- *senzitivní atribut nebo záznam* - v DB může být jistý sloupec nebo řádek prohlášen za tajný
- *senzitivní ve vztahu k dříve vyzrazeným skutečnostem* - např. prozradím-li zeměpisnou šířku tajné základny, neměl bych už povědět nic o zem. délce

největší problémy nastávají v případě, kdy pouze *některé* informace v databázi jsou senzitivní

Rozhodování o přístupu

zavedeme následující značení:
správce SŘBD je program zajišťující dodržování rozhodnutí učiněných administrátorem o tom, kdo z uživatelů má mít jaká přístupová práva
pro jednoduchost budeme uvádět též správce rozhodne …

Dostupnost dat

kromě obvyklých problémů s dostupností informací, zde přistupuje mechanismus zamykání záznamů, který brání načtení nekonzistentních dat
uvažme případ, kdy dojde k poruše stanice, která právě provedla uzamčení záznamů (t.j. provedené zámky neuvolní)

Akceptovatelnost přístupu

SŘBD musí mít stále na zřeteli, které záznamy jsou senzitivní a nepřístupné uživateli, musí zajistit, aby nedošlo k jejich vyzrazení
problém rozhodování o poskytnutí přístupu je složitý, systém nesmí vydat informace, ze kterých by uživatel mohl dovodit utajované skutečnosti
přesto v některých případech by měl vydat výsledek, závisící na senzitivních informacích - např. různé statistické hodnoty z údajů v databázi

Zajištění autenticity

kromě správné identifikace lze opět omezit přístup uživatele časem, místem apod.
navíc, rozhodnutí o poskytnutí přístupu by mělo záviset též na předchozích dotazech, které uživatel kladl

Vyzrazení dat

Přesné hodnoty

uživatel se záměrně nebo omylem dotáže na tajná data, vlivem špatného mechanismu rozhodování nebo nestability v systému je dotaz zodpovězen

Meze

vydání mezí intervalu, ve kterém se utajované hodnoty nacházejí může být rovněž vážným nedostatkem - zvláště pokud systém tuto informaci vydá o libovolné podmnožině záznamů
na druhou stranu někdy může být vhodné moci podat informace o spodním a horním odhadu pro specifickou množinu záznamů

Negativní výsledek

uživatel může pokládat dotazy směřující k oveření záporné skutečnosti - hodnota specifické položky není rovna hodnotě *x*, časté zejména při ověřování, žehodnota je nenulová
zatímco informace, že hodnota je různá od 42 je téměř bezcenná, informace, že hodnota je nenulová sama o sobě již může znamenat 7vážné porušení utajení

Existence

stejně jako v případě obecných dat vrámci OS, sama existence nějaké informace může být senzitivní informace ☺
uživatel např. vůbec nesmí zjistit existenci určitého atributu - ani v projekcích, ani dotazy na strukturu databáze

Pravděpodobné hodnoty

opsaný příklad:
dotaz - Kolik lidí bydlí na Pennsylvania Avenue 1600?
odpověď - 4
dotaz - Kolik obyvatel Pennsylvania Avenue 1600 je registrovanými komunisty?
odpověď - 1
-> S pravděpodobností 25% je prezident USA komunista.

Bezpečnost versus přesnost

Systém by se měl snažit udržet *bezpečnost* senzitivních dat, a to i proti nepřímým dotazům - což vede ke "konzervativní" strategii v poskytování dat, která způsobí odmítnutí mnoha neškodných dotazů.
Z pohledu uživatele je naopak vhodné poskytovat co nejúplnější a nejpřesnější odpovědi - tedy udržet bezpečnost, ale vydat co nejvíce nesenzitivních informací.
Ideální stav by byl umět vydat právě všechny ne-senzitivní informace.

Problém odvoditelnosti

jde o možnost odvodit senzitivní informace ze znalosti (velkého množství) ne-senzitivních
Mějme následující DB:

Jméno	Pohlaví	Hodnocení	Podpora	Pokuty	Drogy	Kolej
Adams	M	C	5000	45	1	Holmes
Bailey	M	B	0	0	0	Grey
Chin	Ž	A	3000	20	0	West
Dewitt	M	B	1000	35	3	Grey
Earthart	Ž	C	2000	95	1	Holmes
Fein	Ž	C	1000	15	0	West
Groff	M	C	4000	0	3	West
Hill	Ž	B	5000	10	2	Holmes
Koch	Ž	C	0	0	1	West
Liu	Ž	A	0	10	2	Grey
Majors	M	C	2000	0	2	Grey

Přímý útok

útočník se snaží získat informace přímými dotazy, jejichž odpověď závisí na velmi malém počtu vět, které vyhověly podmínkám dotazu

```
list Jméno where
    Pohlaví = M & Drogy = 1
```

takový dotaz může obsahovat velké množství uměle vložených nesplnitelných podmínek

```
list Jméno where
    (Pohlaví = M & Drogy = 1) or
    (Pohlaví ≠ M & Pohlaví ≠ Ž) or
    (Kolej = Grey)
```

neb každý ví, že na koleji Grey o drogy nikdo ani nezavadí

Nepřímý útok

často je z databáze obsahující např. senzitivní osobní údaje povoleno zveřejňovat statistické údaje, které není třeba považovat za tajné
z těchto údajů lze ovšem za vhodných okolností získat původní utajované informace

Součet

Na první pohled nevinný dotaz na součet finanční podpory studentů dle pohlaví a koleje na které bydlí vede k vyzrazení tajné informace o výši finanční podpory studentky Liu

Počet

dotaz na počet může být kombinován s dotazem na součet
kombinujeme-li předchozí dotaz s dotazem na počet studentů toho kterého pohlaví bydlících na jednotlivých kolejích, zjistíme, že na koleji Holmes bydlí kdosi, kdo dostává 5000 finanční podpory

pokud se ještě zeptáme na seznam obyvatel této koleje (což pravděpodobně není tajné) opět jsme získali senzitivní informace

Medián

utajované hodnoty lze získat z dotazu na medián:
sekvence dotazů
q = median(Podpora where Pohlaví = M)
p = median(Podpora where Drogy = 2)
vede k vyzrazení přesné hodnoty finanční podpory studenta Majorse

Tracker attack

účinnou obranou je, pokud správce databáze odepře odpověď na dotazy, jejichž výsledek závisí na malém množství záznamů
útočník však může získat informace porovnáním výsledků několika dotazů:
namísto dotazu

count ((pohlaví = Ž) & (Hodnocení = C) & (Kolej = Holmes))

útočník položí následující dotazy

count (pohlaví = Ž)
count ((pohlaví = Ž) & (Hodnocení ≠ C) & (Kolej ≠ Holmes))

Odečtením výsledků získáme výsledek prvního dotazu, který správce nevydal

Ochrana proti odvoditelnosti

- rozbor dotazů i s ohledem na minulost - velice komplikovaný, nákladný a málo spolehlivý, efektivní pouze proti přímým útokům.
- ochrana vlastních dat - pasivní ochrana, hlavními metodami jsou potlačení (suppression) a skrytí (concealing) výsledků
  - potlačení - systém nevydá odpověď na všechny dotazy, ale případné odpovědi jsou přesné
  - skrytí - systém odpovídá na všechny dotazy, ale odpovědi jsou záměrně nepřené

Potlačení malých výsledků (limited response suppression)

systém nevydá odpověď, pokud výsledná hodnota nepřekračuje stanovený limit, případně závisí na příliš malém, nebo příliš velkém oboru (tzn. $<k$ nebo $>n-k$ řádků z celkových n pro zvolené k)

Kombinování výsledků

systém nevydává výsledky týkající se jednotlivých hodnot z dané domény ale pouze souhrné výsledky pro intervaly těchto hodnot

Modifikace výsledků (response modification)

systém spočítá přesný výsledek, který následně mírně poškodí
možností je zaokrouhlování výsledků, navracení průměru výsledků pro interval hodnot z dané domény apod.

Náhodný šum

před vyhodnocením výsledku je ke každé použité položce připočtena malá náhodna chyba, chybové hodnoty volíme jako náhodnou veličinu se střední hodnotou 0

Náhodný výběr (random sample)

systém neodpovídá na základě všech hodnot v databázi ale pouze na základě provedeného náhodného výběru ze všech relevantních položek
aby nebylo možné počítáním průměrných hodnot výsledků opakovaných dotazů získat přesné hodnoty, měl by se pro ekvivalentní dotazy používat stále stejný výběr

Náhodné zmatení (random data perturbation)

ke každé položce v databázi přičteme náhodnou chybu *e,* přičemž na rozdíl od náhodného šumu pro opakované dotazy systém zajišťuje, že použitá chyba je stále stejná
pokud je chyba z okolí nuly, je vliv této úpravy na statistické výsledky typu součet, průměr, … malý
metoda je vhodnější než předchozí, neboť lze snáze zajistit stejné výsledky ekvivalentních dotazů.

Víceúrovňové databáze (multilevel db)

opět se budeme zabývat případy, kdy je nutno uvažovat několik úrovní důvěrnosti či utajení zpracovávaných informací
je velmi důležitá granularita, s jakou je ochrana prováděna - je jasné, že hodnoty jednotlivých atributů mohou mít rozdílnou citlivost, rovněž citlivost tak řádků se může lišit
v případě databází je dále třeba brát v úvahu

- senzitivita dané položky může být jiná než senzitivita všech ostatních položek v daném řádku nebo sloupci - je tedy třeba implementovat bezpečnost na úrovni položek
- obecně bude nutno počítat s několika stupni citlivosti a též s rozdělením dle "tematických okruhů" obdobně jako v případě mřížkového modelu
- citlivost agregovaných hodnot může být odlišná od citlivosti individuálních hodnot

- nestačí sledovat pouze jednotlivé hodnoty, ale je nutné chránit i různé kombinace uchovávaných hodnot, které mohou mít opět různou citlivost

SŘBD musí zachovávat integritu a utajení dat, ale sám se nemůže řídit např. *-vlastností popsanou v Bell-LaPadulově modelu (musí být schopen číst a zapisovat všechny položky)
utajování dat má i druhou stránku - pokud pracovník zjistí, že údaje poskytované databází nejsou z jeho pohledu úplné (byly vypuštěny utajené záznamy) může provést doplnění těchto "chybějících" záznamů

Metody ochrany ve víceúrovňových databázích

Parcelizace (partitioning)

databáze je rozdělena dle stupně citlivosti informací na několik subdatabází

- vede k zvýšení redundance s následnou ztíženou aktualizací
- neřeší problém nutnosti současného přístupu k objektům s různým stupněm utajení

Šifrování

senzitivní data jsou chráněna šifrováním před náhodným vyzrazením
zná-li útočník doménu daného atributu, může snadno provést chosen plaintext attack (zašifrováním všech hodnot z domény)
řešením je používat jiný klíč pro každý záznam, což je však poměrně náročné
v každém případě nutnost neustálého dešifrování snižuje výkon systému

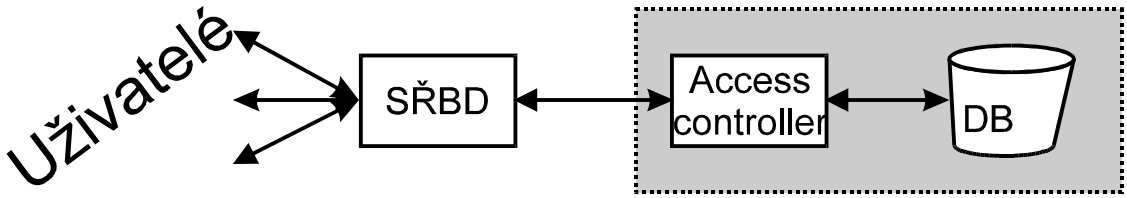
Integrity lock

každá položka v databázi se skládá ze tří částí:

<vlastní data : klasifikace : checksum>

- vlastní data jsou uložena v otevřené formě
- klasifikace musí být nepadělatelná, nepřenositelná a skrytá, tak aby útočník nemohl vytvořit, okopírovat ani zjistit klasifikaci daného objektu
- checksum zajišťuje svázání klasifikace s daty a integritu vlatních dat

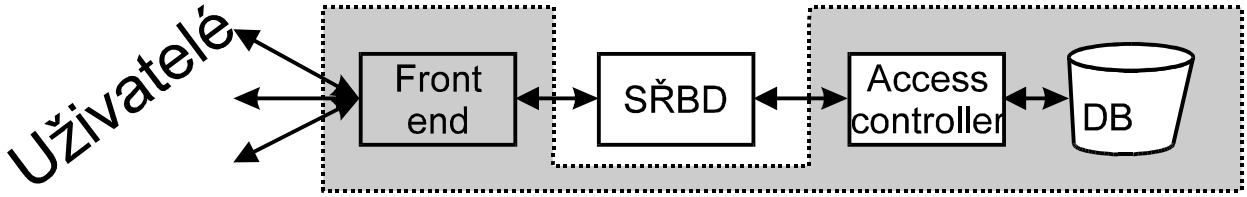
model byl navržen jako doplněk (access controller) komerčního SŘBD, který měl zajistit bezpečnost celého systému



šedá oblast vyznačuje bezpečnostní perimetr systému

Spolehlivý front-end (guard)

systém je opět zamýšlen jako doplněk komerčních SŘBD, které nemají implementován u bezpečnost



uživatel se autentizuje spolehlivému front-endu, který od něho přebírá dotazy, provádí kontrolu autorizace uživatela pro požadovaná data, předává dotazy k vyřízení SŘBD a na závěr provádí testy integrity a klasifikace výsledků před předáním uživateli
SŘBD přistupuje k datům prostřednictvím spolehlivého access kontroleru

Komutativní filtr (Commutative Filter)

jde o proces, který přebírá úlohu rozhraní mezi uživatelem a SŘBD

- filter přijímá uživatelovy dotazy, provádí jejich přeformulování a upravené dotazy posílá SŘBD k vyřízení
- z výsledků, které SŘBD vrátí, odstraní data, ke kterým uživatel nemá přístupová práva a takto upravené výsledky předává uživateli

filter je možno použít k ochraně na úrovni záznamů, atributů a jednotlivých ;položek
v rámci přeformulování dotazu může např. vkládat další podmínky do dotazu, které zajistí, že výsledek dotazu závisí jen na informacích. ke kterým má uživatel přístup

Pohled (View)

pohled je část databáze, obsahující pouze data, ke kterým má daný uživatel přístup
pohled může obsahovat i záznamy nebo atributy, které se v původní databázi nevyskytují a vznikly nějakou funkcí z informací původní databáze
pohled je generován dynamicky, promítají se tedy do něho změny původní DB
uživatel klade dotazy pouze proti svému pohledu - nemůže dojít ke kompromitaci informací, ke kterým nemá přístup

záznam / atribut původní databáze je součástí pohledu pokud alespoň jedna položka z tohoto záznamu / atributu je pro uživatele viditelná, ostatní položky v tomto jsou označeny za nedefinované
uživatel při formulování dotazu může používat pouze omezenou sadu povolených funkcí
tato metoda je již návrhem směřujícím k vytvoření bezpečného SŘBD

Bezpečnost v aplikačních serverech

aplikační server využívá OS a databázi jako persistentní repository vlastních dat včetně nastavení bezpečnostního mechanismu
nezbytný vlastní bezpečnostní mechanismus, zahrnující:

- autentizaci
- autorizaci
- auditní záznamy
- správu prostředků
- zajištění dostupnosti
- ochranu komunikace
- konzistenci dat
- řízení změn
- testování
- ...,

Standardem oddělení vývoje od testování (školení, sandbox, ...) a produktivního prostředí
Nezbytností řešit globálně personální politiku a separaci rolí
Vhodné dedikovat kapacity OS(DB) výhradně pro alikační server