Segurança

Capítulo 21

Agenda:

1. Introdução
2. Controle de Acesso
   - Discretionary Access Control – DAC
   - Mandatory Access Control – MAC
3. Segurança para Aplicações de Internet
   - Criptografia
   - Certificando Servidores: O Protocolo SSL
   - Autenticação de Usuários
   - Assinaturas Digitais
4. Segurança em Bancos de Dados Estatísticos

Nos exemplos foram usados os seguintes schemas:

- Sailors (sid:integer, sname:string, rating:integer, age:real)
- Boats (bid:integer, bname:string, color:string)
- Reserves (sid:integer, bid:integer, day:dates)

Introdução a segurança em BD

- Sigilo: Usuários não devem acessar dados aos quais não têm permissão.
  - Ex: Um estudante não pode ver notas de outros estudantes.
- Integridade: Usuários não devem modificar dados sem permissão.
  - Ex: Somente professores podem dar notas.
- Disponibilidade: Usuários devem poder modificar e acessar dados aos quais tenham permissão. Um professor pode alterar uma nota, caso seja necessário.

Controle de Acessos

- Uma política de segurança especifica quem tem autorização e para que.
- Um mecanismo de segurança nos permite forçar o uso de uma política de segurança.
- Dois mecanismos no nível do DBMS são:
  - Discretionary Access Control (DAC)
  - Mandatory Access Control (MAC)

Discretionary Access Control (DAC)

- Baseado no conceito de direito de acesso ou privilégio a objetos (tabelas e visões), e mecanismos para conceder e revogar privilégios de usuários.
- O criador de uma tabela ou uma visão automaticamente tem todos privilégios sobre o que criou.
  - DBMS mantém dados de quem recebeu e perdeu privilégios e garante que somente requisições feitas apenas por usuários que tenham os privilégios necessários (no momento em que a requisição foi feita) sejam permitidas.
- SQL suporta DAC através dos comandos GRANT e REVOKE.

Comando GRANT

GRANT privilégios ON objeto TO usuários [WITH GRANT OPTION]

- O seguintes privilégios podem ser especificados:
  - SELECT: Pode ler todas as colunas (incluindo aquelas adicionadas posteriormente via comando ALTER TABLE).
  - INSERT(nome-col): Pode inserir tuplas com valores não nulos ou não default na coluna especificada.
  - UPDATE(nome-col): Pode atualizar coluna especificada nas tuplas.
  - DELETE: Pode remover tuplas.
  - REFERENCES (nome-col): Pode definir chaves estrangeiras (em outras tabelas) que se referem à coluna especificada.

  Obs: Se não informada coluna, refere-se a todas.
- Se um usuário tem privilégios com GRANT OPTION, então ele pode passar seus privilégios para outros usuários (com ou sem GRANT OPTION).
- Somente o dono do schema pode executar CREATE, ALTER e DROP. Esse privilégio não pode ser repassado ou revogado.

Exemplos de GRANT

Considerando que Joe criou as tabelas Boats, Reserves e Sailors.

GRANT INSERT, DELETE ON Reserves TO Yuppy WITH GRANT OPTION → Yuppy pode inserir e remover dados da tabela Reserves e repassar o privilégio a outros usuários

GRANT SELECT ON Reserves TO Michael

GRANT SELECT ON Sailors TO Michael WITH GRANT OPTION → Michael pode selecionar dados em Reserves e Sailors e repassar o privilégio em Sailors, não em Reserves. Ele também poderia criar uma visão que referenciasse as duas tabelas, mas não poderia repassar o privilégio de seleção sobre a visão a outros usuários.

Michael também poderia criar uma restrição baseada nas informações de Sailors ou Reserves.

CREATE TABLE Sneaky (maxrating INTEGER CHECK (maxrating >=

(SELECT MAX(S.rating) FROM Sailors)))

A cada inserção de novos registros em Sneaky, o maxrating é comparado com o maior valor existente em Sailors. Por esse motivo, Michael precisa ter privilégio de seleção em Sailors.

GRANT UPDATE(rating) ON Sailors TO Leah → Leah pode atualizar a coluna rating em Sailors. Contudo, ela não poderia executar o comando:

UPDATE Sailors S SET S.rating = S.rating -1

Pois ela não tem privilégio de seleção na coluna rating.

GRANT REFERENCES(bid) ON Boats TO Bill → Bill pode referenciar bid como chave estrangeira em outra tabela.

Comando REVOKE

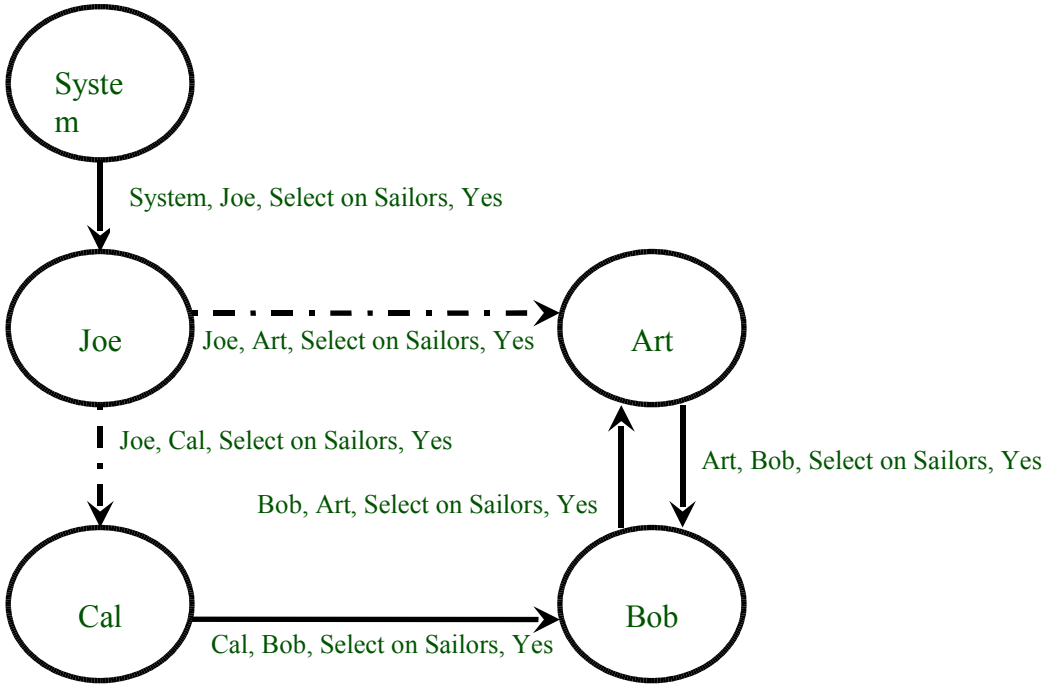
- REVOKE é o comando complementar ao GRANT. Com ele pode tanto ser retirado um privilégio ou o GRANT OPTION de um privilégio de um usuário. Sintaxe:

  REVOKE [GRANT OPTION FOR] privilégios ON objeto FROM usuários {RESTRICT | CASCADE}

- Nem sempre que um usuário revogar o privilégio de outro fará com que este perca realmente a permissão sobre o objeto, pois ele pode ter recebido o mesmo privilégio múltiplas vezes de outros usuários.
- A cláusula CASCADE faz com que um privilégio ou GRANT OPTION concedido a um usuário seja revogado dele e de todos os usuários que receberam o privilégio exclusivamente dele. Se esses usuários que estão perdendo o privilégio tiverem concedido o privilégio a outros, todos perdem também, desde que não tenham recebido GRANT de outro usuário. O privilégio é considerado abandonado se foi revogado como conseqüência do usuário que concedeu o privilégio tê-lo perdido.
- A cláusula RESTRICT faz com que o comando seja recusado se ao retirar o privilégio dos usuários escolhidos em usuários outros privilégios tornem-se abandonados.
- Se um usuário receber privilégio múltiplas vezes de um mesmo usuário, basta um único REVOKE para que ele perca o privilégio concedido.

Exemplos de REVOKE

GRANT SELECT ON Sailors TO Art WITH GRANT OPTION (executed by Joe)
GRANT SELECT ON Sailors TO Bob WITH GRANT OPTION (executed by Art)
GRANT SELECT ON Sailors TO Art WITH GRANT OPTION (executed by Bob)
GRANT SELECT ON Sailors TO Cal WITH GRANT OPTION (executed by Joe)
GRANT SELECT ON Sailors TO Bob WITH GRANT OPTION (executed by Cal)
REVOKE SELECT ON Sailors FROM Art CASCADE (executed by Joe)
REVOKE SELECT ON Sailors FROM Cal CASCADE (executed by Joe)



GRANT/REVOKE em Visões

- Se o criador de uma visão perde o privilégio de SELECT em uma tabela usada pela visão, então a visão é apagada!
- Se o criador de uma visão perde um privilégio dado com GRANT OPTION em uma tabela usada na visão, ele perde o privilégio na visão; e também usuários que haviam ganho o privilégio sobre a visão!
- Se o criador de uma visão receber privilégios adicionais nos objetos usados pela visão, automaticamente os receberá na visão.

Visões e Segurança

```
CREATE VIEW ActiveSailors (name,age,day)
AS SELECT S.sname, S.age,R.day
FROM Sailors S, Reserves R
WHERE S.sid = R.sid AND R.rating > 6
```

- Visões podem ser usadas para apresentar dados necessários, enquanto escondem detalhes em tabelas subjacentes.
  - Dado **ActiveSailors**, mas não Sailors ou Reserves, podemos achar navegadores que tenham uma reserva, mas não o *bid* do barco que foi reservado.
- O Criador de uma visão tem um determinado privilégio sobre a visão se tem este privilégio em todas tabelas usadas pela visão.
- Juntamente com comandos GRANT/REVOKE, visões são ferramentas poderosas para o controle de acesso.

Role Based Access Control (RBAC)

- No SQL-92, privilégios são dados para authorization ids, que podem denotar um usuário ou um grupo de usuários.
- No SQL:1999 (e em muitos sistemas atuais), privilégios são dados para papéis (roles).
  - Papéis podem então ser concedidos para outros usuários ou outros papéis.
  - Reflete como as organizações trabalham.
  - Ilustra como padrões freqüentemente evoluem para se adequar a padrões utilizados em sistemas reais.

Segurança a nível de campos

- Pode-se criar uma visão que retorna apenas um campo de uma tupla? (Como?)
- E então dar acesso a esta visão?
- Permite controle com granularidade *arbitrária*, entretanto:
  - É difícil de especificar, apesar disto poder ser minimizado com uma boa UI.
  - A performance é inaceitável se é preciso utilizar granularidade a nível de campos frequentemente (muitas criações de views e look-ups).

Desvantagens do DAC

- DAC tem alguns problemas, ex, Cavalo de Tróia (*Trojan horse)*:
  - Dick cria uma tabela "MineAllMine" e da privilégio de INSERT para Justin (que não sabe nada sobre isto).
  - Dick modifica o código de alguma aplicação usada por Justin para adicionalmente ler dados secretos que somente Justin tem permissão e escrever na tabela "MineAllMine".
  - Agora, Dick pode ver a informação secreta.
- A modificação do código está além do controle do SGBD, porém podemos prevenir o uso do banco de dados como um canal para informações secretas.

Mandatory Access Control (MAC)

- Baseado em políticas globais do sistema que não podem ser alteradas por usuários específicos.
  - Cada objeto do BD é associado a uma classe de segurança.
  - Cada usuário ou programa é associado a um passe (clearance) para uma classe de segurança.
  - Regras baseadas em classes de segurança e direitos governam quem pode ler/escrever quais objetos.
- A maioria dos sistemas comerciais não suportam controle de acesso MAC. Versões de alguns SGBD suportam; usados para aplicações especializadas (ex, militar).

Modelo Bell-LaPadula

- Objetos (ex., tabelas, visões, tuplas)
- Sujeitos (ex, usuários, programas)
- Classes de Segurança e Passes:
  - Top secret (TS), secret (S), confidencial (C), não classificada (U): **TS > S> C > U**
- Cada objeto é ligado a uma classe de segurança e cada sujeito a um passe.
  - Sujeito S pode ler objeto O só se classe(S) >= classe (O) (Propriedade Simples de Segurança)
  - Sujeito S pode escrever no objeto O só se classe(S) <= classe(O) (Propriedade *)

Intuição

- A idéia é assegurar que a informação nunca trafegue de um nível alto para um nível mais baixo.
- Ex, Se Dick tem classe C, Justin tem classe S, e a tabela secreta de Justin é classificada como S:
  - A tabela criada por Dick (MineAllMine), tem no máximo a classificação C.
  - A aplicação de Justin tem passe S.
  - Então o programa não poderá escrever na tabela "MineAllMine". (Propriedade *)
- As regras impostas pelo MAC são usadas de forma complementar ao DAC.

Relações de Multiníveis

- Implementando-se MAC em um SGBD, cada objeto deve estar associado a uma Classe de Segurança.
- Os objetos podem estar na granularidade de tabelas, views, linhas ou colunas.
- Se, por exemplo, cada linha está associada a uma Classe de Segurança em uma tabela, tem-se então uma tabela Multinível onde cada usuário com diferentes passes vêem um conjunto de linhas diferentes quando acessam a tabela (diferentes instâncias).

Relações de Multiníveis (Exemplo)

bid	bname	color	class
101	Salsa	Red	S
102	Pinto	Brown	C

- Usuários com passe S e TS conseguem ver ambas linhas; um usuário com passe C vê somente a 2ª linha; um usuário com passe U não vê nenhuma.

Relações de Multiníveis (Cont.)

- Se um usuário C tenta inserir <101,Pasta,Blue,C>:
  - Permitir a inserção → violaria a restrição de chave
  - Não permitir a inserção → o usuário conclui que há outro objeto com chave 101 que tem classe de segurança > C.
- O problema é resolvido considerando a classe parte da chave.

bid	bname	color	class
101	Salsa	Red	S
101	Pasta	Blue	C
102	Pinto	Brown	C

Poli-Instanciação

- A presença de objetos que aparentam ter valores diferentes para usuários com diferentes passes é chamada de Poli-Instanciação.

Usuário com passe C

bid	bname	color	class
101	Pasta	Blue	C
102	Pinto	Brown	C

Usuário com passe S ou TS

bid	bname	color	class
101	Salsa	Red	S
101	Pasta	Blue	C
102	Pinto	Brown	C

Segurança e Internet

- Questões chave: Autenticação de Usuários e Confiança.
  - Quando bancos de dados são acessados de uma localização segura, autenticação baseada em mecanismo de senhas é normalmente adequada.
  - Para acessos de uma rede externa (a compra de um livro pela internet), o estabelecimento de confiança é difícil.
    - Se alguém com o cartão de crédito registrado para Sam quer comprar um livro pela internet, como a Amazon pode ter certeza que ninguém roubou o cartão dele?
    - Como Sam pode ter certeza que o formulário pedindo o número do cartão é realmente uma parte legítima do site da Amazon, e não uma aplicação fraudulenta com o objetivo de obter o número de seu cartão de crédito?
    - Criptografia é a técnica utilizada para tratar estes problemas.

Criptografia

- Mascara dados para transmissão ou armazenamento seguros
  - Encrypt(dado, chave de criptografia) = dado criptografado
  - Decrypt(dado criptografado, chave de descriptografia) = dado original
  - Sem a chave de descriptografia, o dado criptografado não pode ser decifrado.
- Criptografia Simétrica:
  - Chave de criptografia = chave de descriptografia; todos usuários autorizados sabem a chave (problema).
  - DES, usado desde 1977, tem chave de 56 bits; AES tem 128 bits (opcionalmente, 192 ou 256).
- Criptografia baseada em Chave Pública: Cada usuário tem duas chaves:
  - Uma chave pública para criptografia: conhecida por todos
  - Uma chave privada para descriptografia: conhecida somente pelo usuário

Criptografia

- Uma propriedade muito importante dos algoritmos de criptografia e descriptografia é que a função das chaves de criptografia de descriptografia pode ser trocada, ou seja:

decrypt(d,(encrypt(e,I))) = I = decrypt(e,(encrypt(d,I)))

onde:

e = chave de criptografia

d = chave de descriptografia

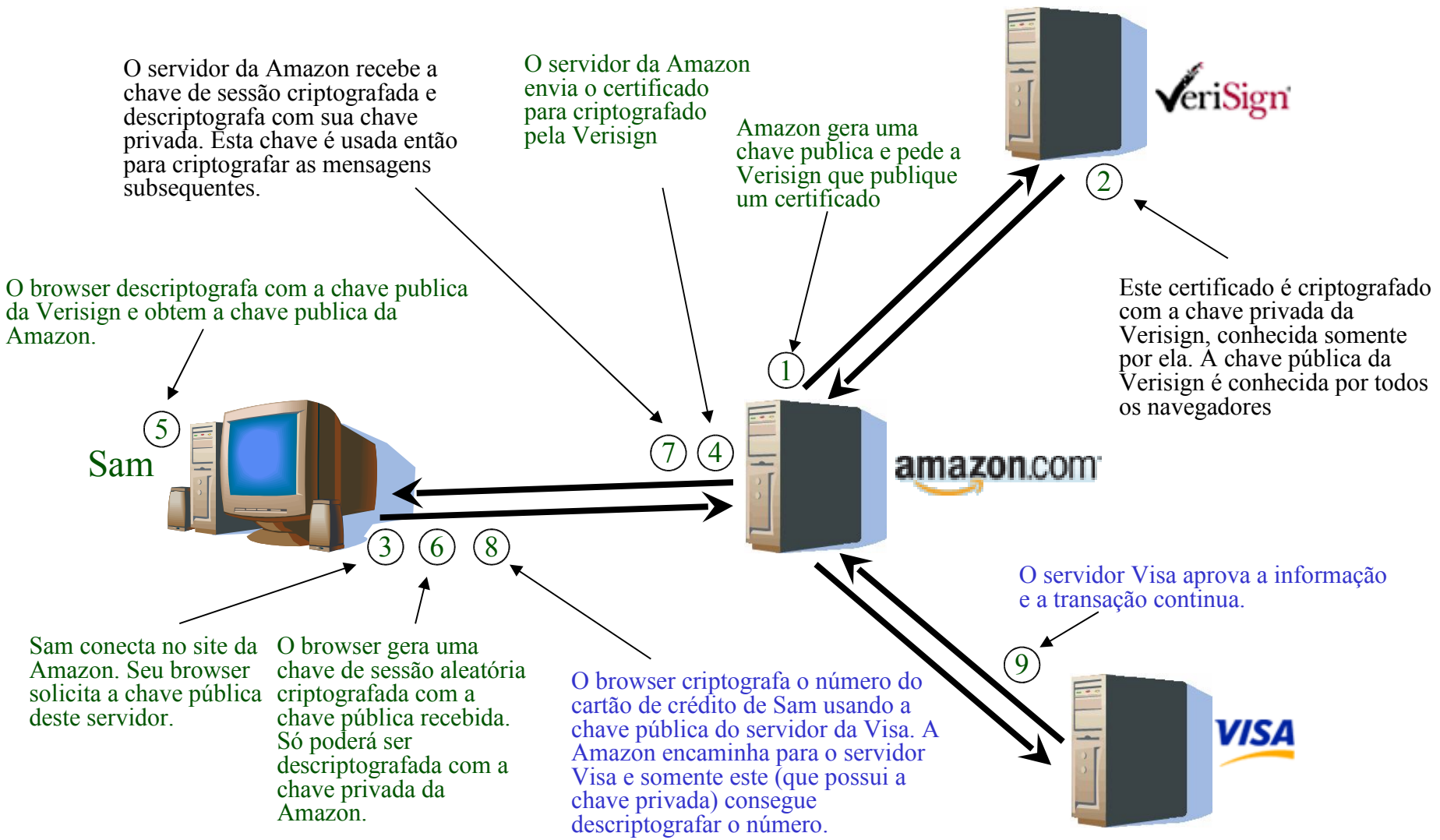
I = dado

- Uma vez que a função das chaves de criptografia e descriptografia podem ser trocadas, elas são simplesmente chamadas de chave pública e chave privada.

Certificando Servidores: SSL

- Se a Amazon distribui a sua chave pública, o navegador usado por Sam poderá usa-lá para criptografar o seu pedido.
  - Somente a Amazon poderá decifrar o pedido, pois ninguém mais possui a chave privada correspondente.
- Mas como Sam sabe que a chave pública é realmente da Amazon? O protocolo SSL garante isto.
  - Amazon pede a Verisign que publique um certificado
  - Este certificado é criptografado com a chave privada da Verisign, conhecida somente por ela.
  - A chave pública da Verisign é conhecida por todos os navegadores, que podem portanto decifrar o certificado e obter a chave pública da Amazon, tendo certeza que ela é genuína.
  - O navegador então gera uma chave de sessão temporária, codifica-a usando a chave pública da Amazon, e a envia para a Amazon.
  - Todas mensagens subseqüentes entre o navegador e a Amazon são codificadas usando a chave de sessão com criptografia simétrica (ex. DES), que é mais eficiente que criptografia por chave pública.
- E se Sam não confia na Amazon com relação ao cartão de crédito?

Certificando Servidores: SSL (Cont.)



Autenticando Usuários

- A Amazon pode utilizar autenticação baseada em senha, isto é, pedir que Sam entre na sua conta.
  - É feita depois que SSL foi utilizado para estabelecer um id de sessão, portanto a transmissão da senha é segura.
  - Mas a Amazon ainda corre o risco de que o cartão de crédito seja roubado e a senha seja "hackeada".
- Como evitar isso?
  - Sam criptografa o pedido usando a sua chave privada, e então criptografa o resultado usando a chave pública da Amazon.
  - A Amazon descriptografa a mensagem com a sua chave privada, e então descriptografa o resultado com a chave pública de Sam.
  - Utiliza a capacidade de troca entre chaves públicas e privadas para criptografia.
  - Agora ninguém pode se passar por Sam e, por outro lado, Sam também não pode alegar que alguém se fez passar por ele.

Assinaturas Digitais

- Se a autenticação do remetente é o objetivo mais importante, pode-se reduzir o custo da criptografia usando uma assinatura da mensagem.
- Verifica-se a integridade da mensagem e também a identidade do remetente usando uma **assinatura digital.**
- Para criar uma assinatura digital, o assinante aplica um esquema hashing à mensagem original e obtem uma versão resumida da mensagem (hash). Então, utiliza sua chave privada para criptografar o hash. O hash criptografado é a assinatura digital. E este é incluido na mensagem original.
- O receptor recria o hash através da mensagem recebida e então utiliza a chave pública do assinante para descriptografar o hash criptografado incluído na mensagem recebida. Se os dois resultados forem iguais, pode-se concluir que:
  - A mensagem realmente veio do assinante uma vez que foi assinada digitalmente por ele, garantindo a autenticidade da origem da mensagem.
  - A mensagem não foi modificada, garantindo a integridade da mesma.

Segurança em BD Estatísticos

- BD estatístico: Contém informações individuais mas permite apenas consultas agregadas (ex, média das idades).
- Novo problema: Pode ser possível inferir informações secretas!
  - Ex, Se Joe é o navegador mais velho, Posso perguntar "Quantos navegadores tem idade maior que X?" para valores diferentes de X até obtermos 1 como resposta; permitindo inferir a idade de Joe.
- Idéia: Só permitir consultas que envolvam pelo menos N linhas. Funciona? (Não!)

Porque "N" não é suficiente...

- Perguntando "Quantos navegadores tem pelo menos idade X?" até o sistema rejeitar a consulta, pode identificar N navegadores, incluindo Joe, que são mais velhos que X; Seja X=55 neste ponto.
- Depois perguntamos "Qual é a soma das idades maiores que X?" Seja S1 este resultado.
- Depois "Qual a soma de idade de navegadores exceto Joe que tem mais que X, mais minha idade?" Seja S2 este resultado.
- S1- S2 + minha idade é a idade de Joe!

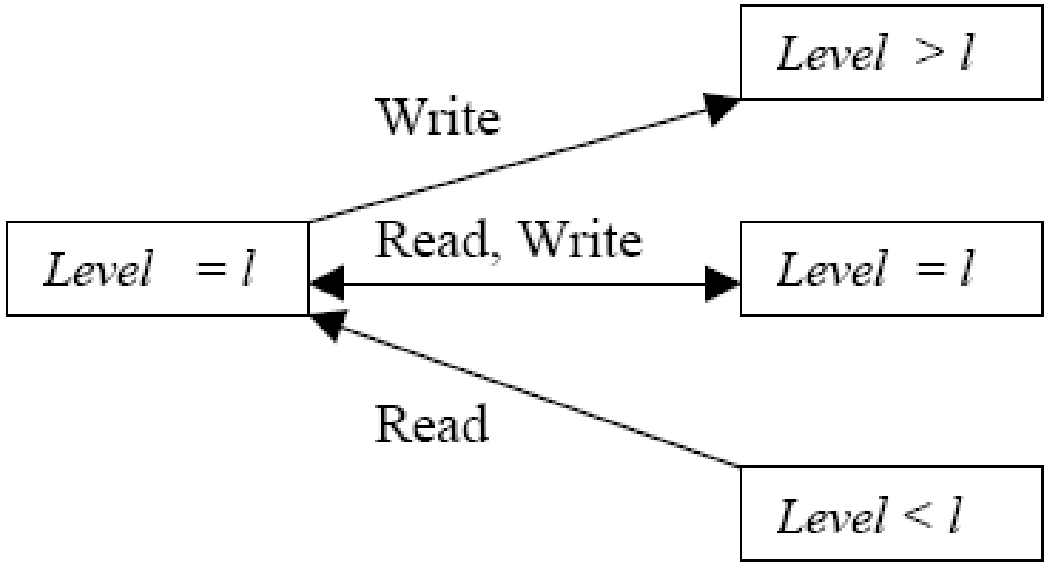
Sumário

- Três objetivos: sigilo, integridade, disponibilidade.
- O administrador é responsável pela segurança.
  - Estabelece políticas de segurança, mantém uma trilha de auditoria, ou histórico de acesso de usuários no BD.
- Dois modelos de segurança no SGBD: DAC e MAC.
  - DAC é baseado em privilégios.
  - MAC é baseado em classes de segurança.
- BDs estatísticos tentam proteger dados individuais permitindo apenas consultas agregadas, mas informações individuais ainda podem ser inferidas.

Backup

Backup

- Direitos de acesso (Modelo Bell-Lapadula)



Backup

- Esquema de criptografia