ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

NOME CORSO

Fintech Software Developer

Unità Formativa (UF): Basi di dati SQL

Docente: Durando Giulio

Titolo argomento: Sicurezza

Sicurezza database SQL Server

I moderni RDBMS (Relational DataBase Management System, ossia le banche dati o database) costituiscono il fondamento di ogni sistema di business e di altre attività di natura differente.

Due sono le ragioni fondamentali che rendono tali sistemi uno dei target più appetibili per gli hackers: la prima è sicuramente riconducibile alla presenza di dati che molto spesso assumono il carattere di estrema confidenzialità (numeri di carte di credito, dati finanziari, strategici, ecc...) mentre la seconda è data dal fatto che la compromissione di un server di database determina, non di rado, a fronte di una difficoltà di penetrazione talvolta irrisoria, la possibilità da parte dell'attaccante di acquisire il completo controllo della macchina se non addirittura della intera infrastruttura di rete.

Tutti i più moderni RDBMS adottano una architettura di sicurezza basata su tre differenti livelli di protezione:

- **autenticazione**: questa è la fase più delicata che contempla la verifica dell'identità dell'utente attraverso una password generalmente nota soltanto allo stesso;
- **autorizzazione**: è la fase immediatamente seguente l'autenticazione nella quale il sistema deve determinare a quali risorse l'utente può avere accesso e con quali modalità operative. Di regola le differenti attività che coinvolgono tale fase vengono comunemente raggruppate sotto la terminologia di "amministrazione della sicurezza degli utenti";
- **auditing**: questa fase si contraddistingue per l'adozione di mezzi idonei ad identificare e riconoscere possibili abusi oltre che ad assicurare l'integrità delle informazioni;

La gestione dei rischi

Quando si parla di sicurezza dei database si intende principalmente fare riferimento alla necessità di garantire la completa accessibilità dei dati, a prescindere dal loro valore intrinseco. Qualunque attività diretta ad implementare e/o rafforzare la sicurezza di questi sistemi presuppone il rispetto dei seguenti principi:

1. ***la definizione di pratiche e procedure di sicurezza chiare e di facile attuazione;***
2. ***la predisposizione di livelli di sicurezza multipli;***
3. ***l'applicazione costante del principio "dei privilegi minori";***

È importante non sottovalutare l'efficacia di questi criteri ed, in particolare, di quelli indicati nei punti 2 e 3. In effetti senza la presenza di "strati" di sicurezza multipli diventa estremamente facile connettersi ad un database sfruttando la presenza di account noti e con password di default oppure bypassando la sicurezza del sistema operativo e dei vari dispositivi che compongono l'infrastruttura di rete (firewall, router, IDS, ecc…).

Analogamente il principio dei privilegi minori, in virtù del quale ogni utente deve avere assegnati soltanto i privilegi strettamente indispensabili al compimento delle sue attività, evita i danni che possono derivare da un uso improprio (leggi abuso) oppure da azioni compiute in modo accidentale.

Applicazione pratica degli standard di sicurezza

L'attuazione dei principi appena enunciati non può prescindere dalla considerazione di alcuni aspetti fondamentali tra i quali meritano di essere ricordati i seguenti:

- **segretezza e confidenzialità**: i dati devono poter essere consultati e/o modificati soltanto da parte di chi sia debitamente autorizzato;
- **integrità ed autenticità**: i dati non devono poter essere manipolati dolosamente od accidentalmente e la loro provenienza deve essere verificabile;
- **accessibilità**: i dati devono essere sempre disponibili eventualmente anche attraverso il loro immediato ripristino;

Gli aspetti maggiormente coinvolti nel processo di protezione di un RDBMS sono i seguenti:

- ***configurazione iniziale;***
- ***autenticazione degli utenti;***
- ***autorizzazione all'uso degli oggetti del database;***
- ***amministrazione e l'aggiornamento delle policies;***
- ***auditing;***
- ***strategie di backup e di ripristino dei dati;***

Autenticazione degli utenti

L'utilizzo delle risorse di un database deve essere sempre subordinato alla preventiva autenticazione degli utenti che di regola avviene mediante la fornitura di una password. Data la delicatezza dell'intero processo è opportuno predisporre delle policies che stabiliscano:

- ***la durata minima delle password;***
- ***la lunghezza minima ed il rispetto di determinati criteri quali ad esempio***
- ***l'uso combinato di lettere, numeri e simboli;***
- ***la predisposizione di meccanismi di lock degli account dopo un certo numero di tentativi falliti di login;***
- ***la determinazione del ciclo di vita degli account;***
- ***l'opportunità di verificare la facilità con la quale le password possono essere indovinate per mezzo di attacchi basati su un dizionario dati oppure condotti con il metodo della forza bruta;***
- ***la necessità di verificare il fatto che le password siano memorizzate nel database in forma crittografata;***

Autorizzazione all'uso degli oggetti del database

Generalmente i permessi assegnati agli utenti del database racchiudono in sé stessi il diritto di accedere ai vari oggetti (tabelle, viste, stored procedure, ecc...) con modalità operative differenti.

Relativamente a questo aspetto della sicurezza è consigliabile individuare preventivamente, in fase progettuale, i vari utenti del database ed identificare per essi i ruoli (intesi in termini di raccolte di privilegi) più consoni al rispetto del principio dei privilegi minimi.

Amministrazione e aggiornamento delle policies

Nessun processo di protezione può dirsi veramente efficace senza che siano state sviluppate delle policies adeguate. Queste infatti rappresentano il framework utilizzato in tutte le attività ed i processi diretti a rafforzare la sicurezza ed a gestire i rischi ed il loro compito fondamentale è quello di garantire agli amministratori del database che le attività compiute siano effettivamente appropriate per l'organizzazione. Vista la loro importanza è opportuno quindi che le policies regolamentino aspetti fondamentali quali:

- ***la creazione di standard relativamente ad account utente, password, oggetti e ruoli;***
- ***i requisiti minimi in termini di esecuzione di attività di auditing e logging;***
- ***le procedure per la gestione delle patch rilasciate dal produttore del software;***
- ***il controllo degli accessi agli oggetti del database;***
- ***l'utilizzo dei ruoli predefiniti e la predisposizione di nuovi ruoli;***
- ***ogni altro aspetto che assuma rilevanza per l'organizzazione ai fini della sicurezza;***

Auditing

La maggior parte degli odierni database mette a disposizione funzionalità di auditing più o meno estese che tuttavia non sono attive di default a causa del loro impatto negativo sulle performance dell'intero sistema.

Nonostante l'inevitabile carico di lavoro aggiuntivo tali funzionalità hanno una importanza fondamentale ai fini della individuazione di tutte le attività non autorizzate o dolosamente poste in essere.

Pertanto, data la loro importanza, è sempre consigliabile ricorrere alle stesse cercando di adottare un giusto compromesso tra prestazioni complessive e quantità di dati raccolti per effetto del monitoraggio.

Backup e ripristino dei dati
La cancellazione o l'alterazione delle informazioni a causa di malfunzionamenti di carattere generale o per l'esecuzione di operazioni dannose può comportare delle ingenti perdite, non soltanto di carattere economico, qualora non vengano predisposte delle adeguate strategie di backup periodico dei dati.

Inoltre queste procedure non soltanto devono essere previste ma devono altresì essere sottoposte a continui test in modo da assicurare una elevata probabilità di ripristino dei dati in qualsiasi momento.

Scenari di protezione delle applicazioni in SQL Server

Non esiste un unico modo corretto per creare un'applicazione client SQL Server sicura. Ogni applicazione è unica per requisiti, ambiente di distribuzione e popolazione di utenti. Un'applicazione ragionevolmente sicura quando viene inizialmente distribuita può diventare meno sicura nel tempo. È impossibile prevedere con precisione quali minacce potrebbero emergere in futuro.

SQL Server, come prodotto, si è evoluto in molte versioni per incorporare le ultime funzionalità di sicurezza che consentono agli sviluppatori di creare applicazioni di database sicure. Tuttavia, la sicurezza non è inclusa nella scatola; richiede un monitoraggio e un aggiornamento continui.

Minacce comuni

Gli sviluppatori devono comprendere le minacce alla sicurezza, gli strumenti forniti per contrastarle e come evitare falle di sicurezza autoinflitte. La sicurezza può essere considerata al meglio come una catena, in cui un'interruzione in un anello compromette la forza dell'insieme. L'elenco seguente include alcune minacce alla sicurezza

Attacco SQL injection

SQL injection è il processo mediante il quale un utente malintenzionato immette istruzioni Transact-SQL anziché un input valido. Se l'input viene passato direttamente al server senza essere convalidato e se l'applicazione esegue inavvertitamente il codice inserito, l'attacco potrebbe danneggiare o distruggere i dati. È possibile contrastare gli attacchi di SQL Server injection utilizzando stored procedure e comandi parametrizzati, evitando SQL dinamiche e limitando le autorizzazioni a tutti gli utenti.

In un attacco **SQL injection** il malware viene inserito in stringhe successivamente passate al Motore di database per l'analisi e l'esecuzione. Per la prevenzione degli attacchi di questo tipo è necessario esaminare tutte le procedure che creano istruzioni SQL, poiché in SQL Server vengono eseguite tutte le query sintatticamente valide ricevute. Tutti i sistemi di database sono a rischio di attacchi SQL injection e molte delle vulnerabilità vengono introdotte nell'applicazione che sta eseguendo una query sul Motore di database.

Come funziona un attacco SQL injection

La forma principale di un attacco intrusivo nel codice SQL consiste nell'inserimento diretto di codice in variabili di input utente concatenate a comandi SQL ed eseguite. Una forma meno diretta di attacco consiste nell'inserimento di malware in stringhe destinate all'archiviazione in una tabella o come metadati. Quando le stringhe archiviate vengono successivamente concatenate in un comando SQL dinamico, il codice dannoso viene eseguito.

Il processo di intrusione termina prematuramente una stringa di testo e aggiunge un nuovo comando. Poiché prima che venga eseguito è possibile che al comando inserito vengano aggiunte ulteriori stringhe, l'utente malintenzionato termina la stringa inserita con un contrassegno di commento "--". Al momento del'esecuzione, il testo successivo al segno di commento viene ignorato.

Nello script seguente viene illustrata una semplice intrusione nel codice SQL. Nello script viene compilata una query SQL tramite concatenazione di stringhe codificate con una stringa immessa dall'utente:

```
var Shipcity;
ShipCity = Request.form ("ShipCity");
var sql = "select * from OrdersTable where ShipCity = '" + ShipCity + "'";
```

L'utente riceve la richiesta di immettere il nome di una città. Se l'utente immette **Redmond**, la query assemblata dallo script sarà simile alla seguente:

```
SELECT * FROM OrdersTable WHERE ShipCity = 'Redmond'
```

Si supponga, tuttavia, che l'utente immetta la stringa seguente:

```
Redmond'; drop table OrdersTable--
```

In questo caso, la query assemblata dallo script sarà la seguente:

```
SELECT * FROM OrdersTable WHERE ShipCity = 'Redmond';drop table OrdersTable--'
```

Il punto e virgola (;) indica la fine di una query e l'inizio di un'altra. Il doppio trattino (--) indica che il resto della riga corrente è un commento che deve essere ignorato. Se il codice modificato è sintatticamente corretto, verrà eseguito dal server. Quando SQL Server elabora questa istruzione, SQL Server seleziona prima tutti i record in **`OrdersTable`** in cui `ShipCity` è uguale a `Redmond` e poi SQL Server rimuove la tabella **`OrdersTable`**.

Se il codice SQL inserito è sintatticamente corretto, le manomissioni non possono essere rilevate a livello di programmazione. È pertanto necessario convalidare tutti gli input utente e rivedere attentamente il codice per l'esecuzione dei comandi SQL generati nel server in uso

Elevazione del privilegio

Gli attacchi di elevazione dei privilegi si verificano quando un utente è in grado di assumere i privilegi di un account attendibile, ad esempio un proprietario o un amministratore.

- Eseguire sempre con account utente con privilegi minimi e assegnare solo le autorizzazioni necessarie.
- Evitare di utilizzare account amministrativi o proprietari per l'esecuzione del codice. Questo limita la quantità di danni che possono verificarsi se un attacco riesce.
- Quando si eseguono attività che richiedono autorizzazioni aggiuntive, utilizzare la firma della procedura o la rappresentazione solo per la durata dell'attività. È possibile firmare stored procedure con certificati o utilizzare la rappresentazione per assegnare temporaneamente autorizzazioni.

Sondaggio e osservazione intelligente

Un attacco di sondaggio può utilizzare i messaggi di errore generati da un'applicazione per cercare le vulnerabilità della sicurezza. Implementare la gestione degli errori in tutto il codice procedurale per impedire che le informazioni sugli errori di SQL Server vengano restituite all'utente finale.

Autenticazione

Un attacco di iniezione della stringa di connessione può verificarsi quando si usano gli account di accesso di SQL Server se in fase di esecuzione viene creata una stringa di connessione basata sull'input dell'utente. Se la stringa di connessione non viene verificata per le coppie di parole chiave valide, un utente malintenzionato può inserire caratteri aggiuntivi, accedendo potenzialmente a dati sensibili o altre risorse sul server.

Utilizzare l'autenticazione di Windows ove possibile. Se è necessario utilizzare gli account di accesso di SQL Server, utilizzare **SqlConnectionStringBuilder** per creare e convalidare le stringhe di connessione in fase di esecuzione.

Le password

Molti attacchi riescono perché un intruso è stato in grado di ottenere o indovinare una password per un utente privilegiato. Le password sono la prima linea di difesa contro gli intrusi, quindi l'impostazione di password complesse è essenziale per la sicurezza del tuo sistema. Creare e applicare criteri di password per l'autenticazione in modalità mista.

Assegnare sempre una password complessa **sa** all'account, anche quando si utilizza l'autenticazione di Windows.

Ruoli a livello di server

SQL Server fornisce ruoli a livello di server per semplificare la gestione delle autorizzazioni in un server. Questi ruoli sono entità di sicurezza che raggruppano altre entità. L'ambito delle autorizzazioni dei ruoli a livello di server è l'intero server. I *ruoli* equivalgono ai *gruppi* nel sistema operativo Windows.

I ruoli predefiniti del server vengono forniti per motivi di praticità e compatibilità con le versioni precedenti. Laddove possibile, assegnare autorizzazioni più specifiche.

SQL Server fornisce nove ruoli predefiniti del server. Le autorizzazioni concesse ai ruoli predefiniti del server (ad eccezione di **public**) non possono essere modificate. A partire da SQL Server 2012 (11.x), è possibile creare ruoli del server definiti dall'utente e aggiungere autorizzazioni a livello di server a tali ruoli.

È possibile aggiungere entità a livello di server, ad esempio account di accesso di SQL Server, account di Windows e gruppi di Windows, nei ruoli a livello di server. Tutti i membri di un ruolo predefinito del server possono aggiungere altri account di accesso allo stesso ruolo. I membri dei ruoli del server definiti dall'utente non possono aggiungere altre entità del server al ruolo.

I ruoli predefiniti a livello di server e le relative funzionalità sono i seguenti

- **sysadmin:** i membri del ruolo predefinito del server **sysadmin** possono eseguire qualsiasi attività nel server.
- **serveradmin:** i membri del ruolo predefinito del server **serveradmin** sono autorizzati a modificare le opzioni di configurazione a livello di server e ad arrestare il server.
- **securityadmin:** i membri del ruolo predefinito del server **securityadmin** gestiscono gli account di accesso e le relative proprietà. Possono GRANT, DENY, e REVOKE le autorizzazioni a livello di server. Inoltre, possono GRANT, DENY, e REVOKE le autorizzazioni a livello di database se hanno accesso a un database. Questi membri sono inoltre autorizzati a reimpostare le password per gli account di accesso di SQL Server. **N.B**. La possibilità di concedere l'accesso al Motore di database e di configurare autorizzazioni utente consente all'amministratore responsabile della sicurezza di assegnare la maggior parte delle autorizzazioni server. Il ruolo **securityadmin** deve essere considerato equivalente al ruolo **sysadmin** .
- **processadmin:** i membri del ruolo predefinito del server **processadmin** sono autorizzati a terminare processi in esecuzione in un'istanza di SQL Server.
- **setupadmin:** i membri del ruolo predefinito del server **setupadmin** sono autorizzati ad aggiungere e rimuovere server collegati usando istruzioni Transact-SQL. L'appartenenza a **sysadmin** è necessaria per l'uso di Management Studio.
- **bulkadmin:** i membri del ruolo predefinito del server **bulkadmin** sono autorizzati a eseguire l'istruzione BULK INSERT. Il **ruolo bulkadmin** o le autorizzazioni ADMINISTER BULK OPERATIONS non sono supportati per SQL Server in Linux. Solo **sysadmin** può eseguire inserimenti bulk per SQL Server in Linux.
- **diskadmin:** il ruolo predefinito del server **diskadmin** consente di gestire i file su disco.
- **dbcreator:** i membri del ruolo predefinito del server **dbcreator** sono autorizzati a creare, modificare, eliminare e ripristinare qualsiasi database.
- **public:** ogni accesso SQL Server appartiene al ruolo del server **public**. Quando a un'entità del server non sono state concesse o sono state negate autorizzazioni specifiche per un

oggetto a protezione diretta, l'utente eredita le autorizzazioni concesse a public su tale oggetto. Assegnare le autorizzazioni public per un oggetto solo quando si desidera che l'oggetto sia disponibile a tutti gli utenti. Non è possibile modificare l'appartenenza a public. Il ruolo **public** viene implementato in modo diverso rispetto agli altri ruoli e le autorizzazioni possono essere concesse, negate o revocate dai ruoli predefiniti del server pubblico.

Ruoli a livello di database

Per gestire facilmente le autorizzazioni nei database, SQL Server fornisce diversi *ruoli che sono entità di sicurezza che raggruppano altre entità. I ruoli sono analoghi ai ***gruppi*** nel sistema operativo Microsoft Windows. L'ambito delle autorizzazioni dei ruoli a livello di database è l'intero database.

Per aggiungere e rimuovere utenti a un ruolo del database, usare le opzioni ADD MEMBER e DROP MEMBER dell'istruzione **ALTER ROLE** .

Esistono due tipi di ruoli a livello di database:

- *ruoli predefiniti del database* , che sono predefiniti nel database,
- *ruoli del database definiti dall'utente* , che possono essere creati.

I ruoli predefiniti del database vengono definiti a livello di database e sono presenti in ogni database. I membri del ruolo del database **db_owner** possono gestire l'appartenenza ai ruoli predefiniti del database. Nel database msdb sono presenti anche alcuni ruoli predefiniti del database per scopi specifici.

È possibile aggiungere qualsiasi account del database e altri ruoli SQL Server ai ruoli a livello di database. È meglio evitare di aggiungere ruoli del database definiti dall'utente come membri dei ruoli predefiniti, poiché in tal modo si potrebbe provocare un'imprevista intensificazione dei privilegi.

Le autorizzazioni dei ruoli del database definiti dall'utente possono essere personalizzate usando le istruzioni **GRANT**, **DENY** e **REVOKE**.

Non è possibile concedere autorizzazioni a livello di server ai ruoli del database. Gli account di accesso e altre entità a livello di server (come i ruoli del server) non possono essere aggiunti ai ruoli del database. Per sicurezza a livello di server in SQL Server, usare invece i ruoli del server

Ruoli predefiniti del database

- **db_owner:** i membri del ruolo predefinito del database **db_owner** possono eseguire tutte le attività di configurazione e manutenzione nel database `drop` e anche il database in SQL Server. In Database SQL e Azure Synapse alcune attività di manutenzione richiedono autorizzazioni a livello di server e non possono essere eseguite da ruoli **db_owners**.
- **db_securityadmin**: i membri del ruolo predefinito del database **db_securityadmin** possono modificare le appartenenze al ruolo solo per i ruoli personalizzati e gestire le autorizzazioni. I membri di questo ruolo possono potenzialmente elevare i propri privilegi ed è consigliabile monitorarne le azioni.
- **db_accessadmin**: i membri del ruolo predefinito del database **db_accessadmin** possono aggiungere o rimuovere le autorizzazioni di accesso al database per gli account di accesso di Windows, i gruppi di Windows e gli account di accesso di SQL Server.
- **db_backupoperator**: i membri del ruolo predefinito del database **db_backupoperator** possono eseguire il backup del database.

- **db_ddladmin**; i membri del ruolo predefinito del database **db_ddladmin** possono eseguire qualsiasi comando DDL (Data Definition Language) in un database.
- **db_datawriter**: i membri del ruolo predefinito del database **db_datawriter** possono aggiungere, eliminare o modificare i dati di tutte le tabelle utente.
- **db_datareader**: i membri del ruolo predefinito del database **db_datareader** possono leggere tutti i dati da tutte le tabelle e viste utente. Gli oggetti utente possono esistere in qualsiasi schema, ad eccezione *di sys e INFORMATION_SCHEMA*.
- **db_denydatawriter**: i membri del ruolo predefinito del database **db_denydatawriter** non possono aggiungere, modificare o eliminare dati delle tabelle utente contenute in un database.
- **db_denydatareader:** i **membri del** ruolo predefinito del database **db_denydatareader** non possono leggere dati dalle tabelle utente e dalle viste all'interno di un database.