Härten – effektiver Grundschutz für Datenbanken

Autor: Heinz-Wilhelm Fabry, ORACLE Deutschland GmbH

IT-Organisationen benötigen Konzepte, um einzelne Systembestandteile wie Netzwerk, Betriebssystem, Datenbank und Applikationen abzusichern, damit der Missbrauch des Gesamtsystems nicht oder nur mit unverhältnismäßig hohem Aufwand möglich ist.

Die Bedeutung der Absicherung hängt im Einzelfall von unterschiedlichen Faktoren ab, beispielsweise:

- Zwang zur Einhaltung von Gesetzen und Vorschriften (Compliance)
- Kosten, die durch kompromittierte Daten entstehen können
- Individuelle Risikoeinschätzung durch das Unternehmen

In der Regel ist die Datenbank nicht das Einfallstor für Hacker – fast 90 Prozent der Angriffe auf Datenbanken erfolgen über ein geknacktes Betriebssystem –, aber das entbindet den Datenbank-Administrator natürlich nicht davon, auch die Datenbank so gut wie möglich gegen Missbrauch zu schützen.

Ein wesentlicher Schritt in diese Richtung ist das so genannte Härten. Im Leitfaden IT-Sicherheit (Ausgabe 2006, Seite 24) definiert das Bundesamt für Sicherheit in der Informationstechnik (BSI): "Härten (engl. Hardening) bedeutet in der IT-Sicherheit die Entfernung aller Software-Bestandteile und Funktionen, die zur Erfüllung der vorgesehenen Aufgabe durch das Programm nicht zwingend notwendig sind." Im Zusammenhang mit einer Datenbank gibt es allerdings neben dieser Reduktion auf das Notwendige noch weitere, zum Teil relativ einfach zu implementierende Praktiken, die den Schutz vor Missbrauch erhöhen. Auch diese werden nachfolgend vorgestellt.

Optionen, Features und Beispiel-Schemas

Selbst wenn beispielsweise im Rahmen von konzernweiten Lizenzvereinbarungen die Installation des gesamten Software-Stacks der Datenbank möglich ist, sollte man bei jeder Installation nur die Komponenten installieren, die auch tatsächlich genutzt werden. Bei der nachträglichen Suche nach installierten Komponenten helfen Views wie V$OPTION und DBA_REGISTRY. Hinweise auf tatsächlich verwendete Komponenten liefert DBA_FEATURE_USAGE_STATISTICS.

Der Verzicht auf nicht benötigte Komponenten hat im Alltag den Vorteil, dass weder bei Upgrades noch bei der Installation von Sicherheitspatches eine Auszeit erforderlich wird oder sich der Zeitbedarf dafür reduziert. Im Rahmen von Sicherheitserwägungen ist dieser Verzicht sogar noch bedeutsamer, weil Komponenten, die nicht installiert sind, auch keine Angriffsfläche für Hacker bieten.

Sind bei bestehenden Datenbanken nicht verwendete Komponenten installiert, gibt es unterschiedliche Möglichkeiten, diese zu entfernen. Man kann zum Beispiel eine in den Programmcode der Datenbank gelinkte Komponente wie die Option OLAP einfach durch erneutes Linken deinstallieren.

Für das nachträgliche Löschen eines Features muss die Oracle-Software in der Regel nicht neu gelinkt werden. Der Lieferumfang der Datenbank enthält häufig Skripte zum Deinstallieren, beispielsweise entfernt man $ORACLE_HOME/rdbms/admin/catnoexf.sql mit dem Feature Expression Filter.

Auch für das Entfernen von Demo-Schemata stehen solche Skripte zur Verfügung. Das Skript hr_drop.sql in $ORACLE_HOME/demo/schema/human_resources löscht beispielsweise den Benutzer HR und seine Objekte. Bei Fragen zu den von Oracle installierten Benutzern liefert übrigens die Metalink-Note 160861.1 Informationen zur Verwendung der Benutzer und ihrer Passwörter.

Benutzer-Management und Passwörter

Was für die mitgelieferten Beispiel-Benutzer zutrifft, gilt natürlich auch für selbst eingerichtete Benutzer, die nicht mehr benötigt werden, etwa wenn der entsprechende Mitarbeiter die Firma verlassen hat. Lässt sich ein solcher Account nicht ohne weiteres löschen, sollte man mindestens den Status auf EXPIRED & LOCKED setzen.

Wer die aus dem Versuch des Einloggens in einen solchen gesperrten, aber noch bestehenden Account resultierende Fehlermeldung "ORA-28000: account is locked" als Sicherheitsrisiko einstuft, kann diesen auch mit einem ungültigen Passwort schützen.

```
ALTER USER mustermensch
IDENTIFIED BY VALUES 'ungueltig'
```

Das ungültige Passwort führt dann beim Versuch, sich in den Account einzuloggen, zur Fehlermeldung "ORA-01017: invalid username/password; logon denied" und lässt damit die Existenz des Accounts und die Möglichkeit eines falschen Passworts offen.

Zum Thema Benutzer-Management im weiteren Sinne gehört auch der Hinweis, nur die Privilegien zu vergeben, die tatsächlich benötigt werden (least privilege). Das betrifft vor allem die Vergabe von so genannten ANY-Privilegien. Aber auch ein so mächtiges Privileg wie CREATE DATABASE LINK sollte nicht jedem offenstehen. (Anmerkung: Das Privileg war vor der Version 10.1 Bestandteil der Rolle CONNECT). Bei der Suche nach den erteilten

Privilegien helfen beispielsweise die entsprechenden Menüpunkte des Enterprise Managers (siehe Abb. 1) oder die Berichte des kostenlosen Werkzeugs SQL Developer.

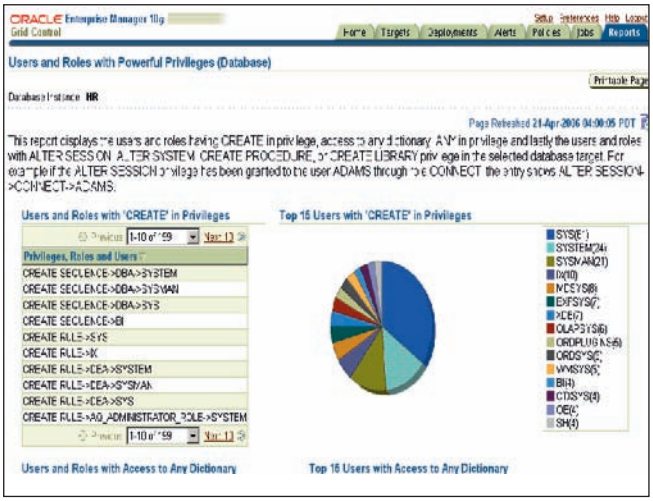

Abbildung 1: Anzeige von Privilegien im Enterprise Manager

Auch das Thema Passwörter gehört in den Bereich des Account-Managements. Auf der Betriebssystem-Ebene – jedenfalls unter Unix/Linux – sollte man beachten, dass Passwörter über die History-Dateien der UNIX-Shells kompromittiert werden können. Vergleichbare Gefahr geht auch von Log- und Trace-Dateien der Datenbank und der Werkzeuge aus, die auf die Datenbank zugreifen. Aber nicht nur im Klartext gespeicherte Passwörter stellen eine Gefahr dar, sondern auch Defaults und ungenügend komplexe Passwörter.

Wenn Datenbanken über Skripte aufgesetzt werden, haben eventuell eingerichtete Demo-Accounts nicht den Status EXPIRED & LOCKED. Auch Passwörter dieser Accounts sind weithin bekannte Defaults. Sie sind das ideale Einfallstor für jeden Hacker. Solche Accounts lassen sich sehr leicht mit dem im Rahmen des Patch 4926128 auf Metalink angebotenen ORACLE DEFAULT PASSWORD SCANNER identifizieren. Ohne den Zugriff auf Metalink sind Passwort-Scanner im Internet teilweise kostenlos bei unterschiedlichen Anbietern als Download verfügbar.

Default-Passwörter ändern

Die Gefahr, die von Default-Passwörtern ausgeht, ist vergleichbar mit jener von ungenügend komplexen Passwörtern. Wird ein solches Passwort – selbst wenn es verschlüsselt ist – mit einem Netzwerk-Sniffer ausgelesen, fällt es nicht mehr schwer, dieses beispielsweise mittels Rainbow Tables in Klartext umzuwandeln.

Oracle bietet zur Erhöhung des Passwort-Schutzes die Möglichkeit, über eine so genannte Password Verify Function die Komplexität der Passwörter im Rahmen von Profilen festzuschreiben. Ein Beispiel dazu liefert utlpwdmg.sql unter $ORACLE_HOME/rdbms/admin. Über Profile lassen sich zudem Account-Charakteristika festlegen, etwa der Zeitraum, nach dessen Ablauf ein Passwort verfällt, oder die Anzahl falscher Passwort-Eingaben vor Sperrung des Accounts.

Hierzu abschließend noch der Hinweis, dass die Änderung eines Passworts mit dem Befehl ALTER USER unver-

```
CREATE PROFILE pwsicherheit LIMIT
FAILED_LOGIN_ATTEMPTS 4 -- Anzahl
PASSWORD_LIFE_TIME 90   -- Tage
PASSWORD_REUSE_TIME 7   -- Tage
PASSWORD_REUSE_MAX 3    -- Anzahl
PASSWORD_LOCK_TIME 30   -- Tage
PASSWORD_GRACE_TIME 3   -- Tage
PASSWORD_VERIFY_FUNCTION funktionsname;

ALTER USER mustermensch PROFILE
pwsicherheit;
```

schlüsselt über das Netzwerk gesendet wird – sofern der Netzwerkverkehr nicht mit der Oracle Advanced-Security-Option oder anderen Mitteln verschlüsselt ist. Alternativ bietet sich der Befehl PASSWORD in SQL*Plus an, der das Passwort verschlüsselt an die Datenbank schickt.

PL/SQL-Packages

Im Lieferumfang der Datenbank sind zahlreiche PL/SQL-Packages enthalten, die jeder Datenbank-Benutzer ausführen kann. Darin steckt eine große Gefahr: Ein Datendieb kann beispielsweise Packages wie UTL_TCP oder UTL_SMTP verwenden, um interne Informationen an externe Adressaten zu schicken. UTL_FILE kann durch die Manipulation von Konfigurationsdateien des Listeners eine Denial-of-Service-(DoS)-Attacke einleiten. Und DBMS_LOB erlaubt das Lesen oder Manipulieren von Datenbank-Dateien, die schließlich nichts anderes sind als LOBs.

Dem Benutzer PUBLIC sollte deshalb der Zugriff auf diese und andere PL/SQL-Packages durch ein REVOKE EXECUTE genommen werden. Allerdings sollte man sich vor einem solchen Schritt vergewissern, dass keine Anwendungen diese Packages verwenden.

Oracle Netzwerk-Konfiguration

Der Zugriff über ein Netzwerk auf die Datenbank ist sehr viel sicherer, wenn die Rechner benannt werden, von denen aus ein Zugriff ausschließlich möglich oder auch ausdrücklich nicht möglich sein soll. Oracle bietet diese Option durch Einträge in der Datei SQLNET.ORA. Setzt man den Parameter TCP.VALIDNODE_CHECKING auf YES, erlaubt dies sowohl die Angabe einer Negativ-Liste (Parameter TCP.EXCLUDED_NODES) als auch einer Positiv-Liste (Parameter TCP.INVITED_NODES).

Der Default-Port 1521 für den Listener sollte auf jeden Fall geändert werden. Das schützt zwar nicht vor Angriffsversuchen, erhöht allerdings den Aufwand für einen Hacker, um das System zu knacken. Im Idealfall führt dieser zusätzliche Aufwand dazu, dass sich der Hacker ein einfacher zu manipulierendes Opfer sucht.

Das Risiko, das der Aufruf manipulierter externer Prozeduren darstellt, sollte nur eingegangen werden, wenn tatsächlich mit solchen Prozeduren gearbeitet wird. Deshalb sollte der als Standard für den Listener eingerichtete Service EXTPROC auch nur dann in der Konfigurationsdatei enthalten sein. Das gilt natürlich auch für alle anderen, nicht benötigten Services.

Auch der Listener kann durch ein Passwort geschützt werden. Für ältere Datenbank-Versionen (vor Oracle10*g*) empfiehlt sich dieser Schutz auf jeden Fall. Seit Oracle10*g* lässt sich der Listener allerdings nur lokal administrieren. Man muss deshalb abwägen, ob der Schutz durch die ausschließlich lokale Administration nicht dem Risiko vorzuziehen ist, den ein passwortgeschützter Listener darstellt. Denn dieser passwortgeschützte Listener ist nicht nur lokal zu administrieren, sondern auch von anderen Systemen aus. Damit ist er auch wieder offen für Brute-Force-Angriffe, bei denen unter Verwendung eines Wörterbuchs immer wieder neue Passwörter zur Identifikation an den Listener geschickt werden.

Schließlich sei noch darauf hingewiesen, dass es in sensiblen Umgebungen sicherlich angebracht ist, regelmäßig die Log-Datei des Listeners auf die Anwendungsprogramme zu durchsuchen, die sich bei der Datenbank angemeldet haben. Findet man hier Programme, die normalerweise nicht im Unternehmen verwendet werden, ist eine sofortige Klärung nötig.

Dateien und Verzeichnisse

Auf allen Plattformen legt Oracle im Rahmen von Upgrades Sicherungskopien alter Programm-Dateien ab. Sie heißen oracle0, oracle.bak oder ähnlich. Diese Dateien können gelöscht oder unter Unix/Linux in ihrem Schutz so verändert werden, dass keinerlei Aktion mehr mit ihnen möglich ist (chmod 000).

Die beiden folgenden Hinweise gelten ausschließlich für Unix-/Linux-Systeme: Die Programme der Datenbank sind teilweise über das SUID-bit beziehungsweise für jeden (world) ausführbar. Das ist in der Regel nicht sinnvoll, sondern stellt ein Sicherheitsrisiko dar. Deshalb sollte der Dateischutz von Dateien wie oracle oder auch sqlplus auf 0700 geändert werden. Das führt zwar dazu, dass auch lokale Verbindungen zur Datenbank immer über den Listener unter Angabe z.B. des Service-Namens hergestellt werden müssen, aber auch ein Einloggen mittels AS SYSDBA ist nur nach Eingabe eines Passworts möglich. Selbst bei einem erschlichenen Zugriff auf den Account des DBAs ist eventuell ein weiteres, dann hoffentlich anderes Passwort für das Einloggen in die Datenbank erforderlich.

Bei der Installation wird die umask der Oracle-SW-Verzeichnisse auf 022 gesetzt. Das führt dazu, dass jeder Benutzer (world) etwa aus den Dump- und Log-Verzeichnissen Dateien kopieren kann. Liegen hier Backup-Sets oder Dump-File-Sets von Data Pump, sind sie quasi öffentlich verfügbar. Deshalb sollte die Einstellung in jedem Fall geändert werden, etwa durch Setzen von umask auf 177.

Sicherheitspatches

Das Einspielen von Sicherheitspatches (CPUs) konkurriert selbstverständlich mit den Ansprüchen an die nahezu unterbrechungsfreie Verfügbarkeit der Datenbank. Allerdings ist unbestritten, dass eine nicht gepatchte Datenbank für den Hacker ein leichteres Opfer darstellt. Selbst wenn dieser nicht über genügend Wissen zum Ausnutzen einer nicht behobenen Sicherheitslücke verfügt, sind Exploits für bereits behobene Sicherheitslücken gegen geringes Entgelt oder sogar gratis im Internet zu finden.

Probleme mit fehlenden Sicherheitspatches können auch durch alte Datenbank-Versionen (kleiner 9.2.0.x) entstehen, für die Oracle keine Sicherheitspatches mehr erstellt. Sie bieten dem Hacker einen leichten Zugang, von dem aus die kriminellen Aktivitäten auf andere Systeme ausgeweitet werden können. Das gilt auch für Datenbanken, die eigentlich nur zum Testen, etwa für Praktikanten oder externen Consultants, aufgesetzt wurden und dann in Vergessenheit geraten sind. Auch hier gibt es Werkzeuge, mit denen der Hacker sehr leicht herausfinden kann, welche Datenbank-Version betrieben wird oder welche Datenbanken leicht anzugreifen sind, weil sie zum Beispiel unter dem Namen test.firmenname.com im Internet sichtbar sind.

Abschließende Hinweise

Natürlich kann man die Sicherheit einer Datenbank noch weiter steigern, etwa durch die Verschlüsselung kritischer Informationen mit dem PL/SQL-Package DBMS_CRYPTO oder dem Feature Transparent Data Encryption (TDE) aus der Advanced-Security-Option. Und auch ohne Auditing ist jedes Sicherheitskonzept unvollständig. Das geht allerdings über den Grundschutz hinaus. Wer sich allerdings weiter mit dem Thema beschäftigen möchte, sei der Artikel Project Lockdown von Arup Nanda empfohlen, zu finden unter http://www.oracle.com/technology/pub/articles/project_lockdown/project-lockdown.pdf.

Kontakt:
Heinz-Wilhelm Fabry
heinz-wilhelm.fabry@oracle.com