Auditing – revisted

Dr. Günter Unbescheid, Database Consult GmbH

In jüngster Zeit erleben die Themen „Security" im Allgemeinen und „Datenbank-Security" im Besonderen zu Recht eine Renaissance. Neben den klassischen Bereichen wie „Authentifizierung", „Autorisierung" und „Verschlüsselung" ist „Auditing" zur Gewährleistung der Nachvollziehbarkeit von Aktionen nach wie vor ein essenziell wichtiger Bestandteil jedes ernstzunehmenden Sicherheitskonzepts und wird in dieser Bedeutung oftmals unterschätzt. Die vielfältigen technischen Möglichkeiten sind unter der aktuellen Datenbank-Version 12*c* nochmals gewachsen. Hinzu kommen Oracle-interne, aber auch Zusatzprodukte von Drittanbietern. Der Artikel gibt einen strategischen und technischen Überblick.

Zur effizienten Umsetzung eines wirksamen Auditing-Konzepts für Oracle-Umgebungen genügt es in den meisten Fällen nicht, sich nur auf die Datenbank selbst und ihre Möglichkeiten zu konzentrieren. Vielmehr muss die Implementierung vor dem Hintergrund des tatsächlichen Aktivitäten-Profils der betroffenen Benutzer erfolgen. Für die Nachvollziehbarkeit von Datenbank-Administrator-Aktionen bedeutet dies, auch die Audit-Möglichkeiten des betreffenden Betriebssystems mit einzubeziehen, um die Aktionen im Kontext von Skripten zu integrieren. Die Planung und Umsetzung eines effizienten Auditing-Konzepts erfordert neben der technischen Expertise aber auch ein hohes Maß an konzernpolitischer Feinfühligkeit und Teamfähigkeit.

Die Grundlagen

Auditing ist Bestandteil eines jeden Sicherheitskonzepts. Folglich gelten auch für dieses Thema die nachstehenden Prämissen, die sich alle Beteiligten stets vor Augen halten sollten:

- Sicherheit ist ein permanenter und iterativer Prozess, der nicht einmalig geplant und durchgeführt wird, sondern der periodisch überprüft und an die aktuellen Anforderungen angepasst werden muss.
- Sicherheit stellt Anforderungen an die Beschaffung bestehender und das Entstehen neuer Applikationen und ist kein Nachbrenner, der am Ende von Projekten durchgeführt wird.
- Sicherheit baut auf Prozesse. Technologien bilden die Basis hierzu – nicht mehr und nicht weniger.
- Auch wenn Sicherheit in der Regel schrittweise umgesetzt wird, kommt es auf ein schlüssiges und abgestimmtes Gesamtkonzept an, das neben der Absicherung der Systeme auch die Effizienz bestehender Prozesse im Auge hat.
- Sicherheit ist immer Team-Arbeit. Die Bereichs- und Abteilungs-übergreifende Abstimmung und Umsetzung ist häufig der kritischste Faktor in den anhängigen Projekten.

Generell lässt sich der Begriff „Auditing" nur schwer vom Thema „Logging" trennen. Eine mögliche, wenn auch in vielen Fällen unscharfe Differenzierung ergibt sich durch die mögliche Aktivierung von Audit-Optionen, die den Kontext und Umfang der protokollierten Aktionen sehr differenziert festlegen können. In jedem Fall lässt sich eindeutig der Zweck von Auditing bestimmen, nämlich die persönliche Nachvollziehbarkeit sicherheitsrelevanter Aktionen zu gewährleisten und Bedrohungen zu erkennen, die – trotz Aufgaben-gemäßer Vergabe von Privilegien – entstehen können. Darüber hinaus existiert in vielen Bereichen die gesetzliche Notwendigkeit derartiger Nachweise. Um diese Ziele wirksam zu erreichen, ist Folgendes zu berücksichtigen:

- Nachvollziehbarkeit bedeutet eindeutige Zuordnung von Aktionen und diese ist nur gegeben, wenn persönliche Benutzer-Accounts existieren.
- Die Protokollierung der Aktionen muss alle relevanten Schichten des genutzten Software-Stacks berücksichtigen. In der Regel müssen hierzu diverse Audit-Trails genutzt und bei Bedarf zusammengeführt werden, im Falle von Datenbank-Administratoren sind dies zumindest die Trails des Betriebssystems und der Datenbank.
- Auditing verhindert keine (destruktiven) Aktionen, sondern hilft nur dabei, diese zu entdecken. Auditing ohne nachgelagerte forensische Analyse der Daten ist in der Regel wirkungslos. Um Bedrohungen zu erkennen, ist eine regelmäßige und festgelegte Auswertung der Daten nötig, die bei konkreten Verdachtsfällen vertieft werden kann.
- Auditing zeichnet personenbezogene Daten auf und unterliegt daher dem Datenschutz. Je umfangreicher Audit-Optionen aktiviert werden, desto lückenloser lässt sich das Aktivitätsprofil eines Benutzers, also einer natürlichen Person, nachvollziehen. Anders als in den USA gehören personenbezogene Daten in der EU nicht dem Sammler, sondern der beschriebenen Person. Damit sind sie im Fokus des Bundes- und der Landesdatenschutzgesetze sowie diverser Rechtsnormen, wie beispielsweise des Telekommunikationsgesetzes (TKG). Mit anderen Worten: Die detaillierte Festlegung von Audit-Optionen und der damit verbundenen Aufbewahrungsfristen kann nur in Abstimmung mit dem jeweiligen Betriebsrat erfolgen. Das Gleiche gilt für die geplanten Auswertungen.

- Die Konfiguration und Überwachung von Audit-Optionen kann IT-technisch sehr umfangreich und aufwändig sein. In diesem Zusammenhang hat es sich bewährt, Systeme zu klassifizieren, beispielweise in die Klassen „Öffentlich", „Vertraulich" und „Streng vertraulich". Die Einteilung in Schutzklassen kann vor dem Hintergrund möglicher Schadens-Szenarien und -Umfänge erfolgen, die im Falle einer Kompromittierung der Daten auftreten könnten. Audit-Optionen können dann nur für die höchsten Schutzklassen in adäquaten Ausprägungen eingerichtet werden.
- Der Audit-Trail muss – aus Sicht der Betroffenen – revisionssicher gespeichert sein. Dies kann durch Zusammenführung der Daten auf einem Remote Server und/oder durch lokalen Schutz mit Mitteln des Betriebssystems und der Datenbank erfolgen.

Die im Audit-Trail aufgezeichneten Daten können sehr umfangreich werden. Aus diesem Grunde gehören zu jedem Audit-Konzept auch Regeln für das Housekeeping. Im Einzelnen bedeutet dies die Regelung von Aufbewahrungsfristen – am Ende der Frist werden die Daten dann entweder gelöscht oder für eine vorgegebene Dauer anonymisiert aufbewahrt. Generell gilt: Je aufgabengerechter die Privilegien erteilt und je überlegter die Audit-Optionen gesetzt sind, desto schlanker ist der Audit-Trail und desto effizienter kann die Auswertung der Daten erfolgen, ohne die Risiken für unentdeckte Kompromittierungen zu erhöhen.

Auditing für Administratoren

Die Planung von Audit-Trails muss vor dem Hintergrund der Aktivitäten-Profile und Zugriffswege der zu überwachenden Benutzer erfolgen. Für Datenbank-Administratoren, die neben den Datenbank-internen Aktionen bekanntlich auch umfangreiche Aufgaben im Kontext des Betriebssystems erledigen müssen, wie beispielsweise das Editieren von Skripten und Parameterdateien sowie das Starten und Stoppen von Komponenten, erfordert dies die Überwachung folgender Kontexte:

- Aufzeichnung des Verbindungsaufbaus zu den Target-Servern, also zu den Servern, auf denen Datenbanken betrieben werden. In vielen Umgebungen ist es üblich, sich von Windows-Clients ausgehend per „PuTTY/ssh" auf Jump Server und von dort aus oder aber direkt auf die Target-Systeme zu verbinden. SSH schreibt standardmäßig Verbindungsprotokolle.
- Nach einem erfolgreichen Connect kann die Ausführung von OS-Kommandos auf den Target-Systemen auf unterschiedliche Weise aufgezeichnet werden. Im Kontext von Linux bieten sich beispielsweise folgende Alternativen an:
  - TTY-Logging (etwa über „pam_tty_audit.so"-Module) zeichnet den Kommando-Input – leider auch die eingegebenen Passwörter – auf.
  - Generell bietet der Audit-Daemon von Linux-Systemen vielfältige Möglichkeiten, Audit-Regeln zu definieren und gesammelte Daten auszuwerten (Kommandos „ausearch" und „aureport").
  - Keylogger oder Shell-Erweiterungen wie „rootsh" oder „sudosh" zeichnen nicht nur den Input, sondern auch den Output auf – erfreulicherweise jedoch keine verdeckten Eingaben wie Passwörter.
  - Aufrufe, die über „sudo" privilegiert gestartet werden, werden ebenfalls im SYSLOG erfasst.
  - Process Accounting in Form des Package „psacct" (oder „acct") zeichnet Daten zu Benutzer-Sessions und deren Kommandoaufrufe auf.
- Aktionen, die innerhalb der Datenbank erfolgen, wie beispielsweise das Anlegen und Löschen von Benutzern, werden – falls konfiguriert – über den Audit-Trail der Datenbank aufgezeichnet. Wenn Keylogger zum Einsatz kommen, ergeben sich hier Redundanzen, denn auch das unten besprochene SYS-Auditing zeichnet SQL-Befehle auf.

Für Administratoren, die von „remote" kommend über Passwörter oder lokal über die zugeordnete Unix-Gruppe als „SYSDBA" in der Datenbank arbeiten, gilt:

- Grundsätzlich werden alle Verbindungen von „SYSDBA" und „SYSOPER" sowie die Start- und Stopp-Aktionen der Datenbank im sogenannten „Mandatory Auditing" erfasst, das im Verzeichnis „AUDIT_FILE_DEST" („init"-Parameter) für jede Instanz und jede Prozess-ID einzelne „.aud"-Dateien schreibt.
- Darüber hinaus lässt sich ein generelles SYS-Auditing über den „init"-Parameter „AUDIT_SYS_OPERATIONS" einschalten, das sämtliche unter „SYSOPER" und „SYSDBA" ausgeführten Kommandoeingaben entweder in Dateien unter „AUDIT_FILE_DEST" oder – besser, weil revisionssicherer – über den „SYSLOG"-Daemon (Parameter „AUDIT_SYSLOG_LEVEL") schreibt. Im Fall von „SYSLOG" ist unbedingt auf einen ausreichenden Durchsatz beim Schreiben des Datenbank-Audit-Trail zu achten. Insbesondere RMAN-Aktionen erzeugen umfangreiche Audit Records, die schnell zu einem Performance- oder Kapazitäts-Engpass führen können.

Bekanntlich sind Benutzer, die unter „SYSDBA" in der Datenbank arbeiten, intern als „SYS" registriert. Dies stellt für die Nachvollziehbarkeit von Aktionen immer dann kein Problem dar, wenn die zugehörigen natürlichen Personen unter einem persönlichen Account auf dem Betriebssystem angemeldet wurden. Der Datenbank-Audit-Trail schreibt nicht nur diese Informationen mit, sondern ebenso andere externe Kennungen, wie beispielsweise „distinguished names" oder „principal names" für Benutzer, die über Directory Services authentifiziert wurden, etwa im Falle von Enterprise-Usern aus dem Kontext des Oracle-Internet-Directory.

Datenbank-Auditing (klassisch)

Klassisches Datenbank-Auditing bezeichnet hier die Methoden und Möglichkeiten des Auditing vor der Einführung des sogenannten „Unified Auditing" (siehe nachfolgenden Abschnitt) unter Version 12*c*. Für den klassischen Audit-Trail stehen über den init-Parameter „AUDIT_TRAIL" unterschiedliche Optionen zur Verfügung. Dieser Parameter beeinflusst nicht das vorstehend beschriebene „SYS"-Auditing und ist daher für alle Sessions wichtig, die als Funktions- oder Endbenutzer außerhalb von „SYSDBA" in der Datenbank arbeiten und deren Aktionen aufgrund der Schutz-

anforderungen protokolliert werden müssen.

Zur Aktivierung bzw. Deaktivierung des Audit-Trails ist ein Neustart der Datenbank erforderlich. Gleichwohl existiert eine Schwachstelle, die die Manipulation dieses, aber auch des „AUDIT_SYS_OPERATIONS"-Parameters ohne Neustart über „oradebug" ermöglicht. Diese wurde jedoch mittlerweile über die Patches „15805002", „15808245" und „16177780" behoben. Der Parameter „AUDIT_TRAIL" kann folgendermaßen gesetzt sein:

- *NONE*
  Das klassische Datenbank-Auditing ist ausgeschaltet.
- *DB [,EXTENDED]*
  Schreibt Audit Records in die Tabelle „SYS.AUD$". Diese Einstellung ist ab der Version 11*g* Standard. Generell wird die Revisionssicherheit durch die Manipulationsmöglichkeiten von „SYS" in der Tabelle „AUD$" erschwert. Auf der anderen Seite bieten sich jedoch gute Auswertungsmöglichkeiten der Daten über SQL-Konstrukte.
- *OS*
  Schreibt OS-Files in das Audit- oder Syslog-Verzeichnis. Diese Option erschwert Auswertungen („grep", „awk", „sed" etc.), verbessert jedoch die Revisionssicherheit, wenn über „SYSLOG" geschrieben wird (in Kombination mit „AUDIT_SYSLOG_LEVEL").
- *XML [,EXTENDED]*
  Schreibt XML-Dateien. Diese Option bietet verbesserte Auswertungsmöglichkeiten durch XML-fähige Programme.

Die „EXTENDED"-Zusätze schreiben zusätzlich SQL-Text und Bindevariablen in den Audit-Trail. Aus Gründen der Revisionssicherheit ist es empfehlenswert, den „SYSLOG"-Daemon zur verwenden, womit auch das Schreiben auf Remote Server konfiguriert werden kann („rsyslog"-Daemon); das erhöht die Revisionssicherheit nochmals.

Die Planung und Konfiguration von Audit-Optionen, die sich der Aktivierung des Audit-Trail anschließen muss, erfordert große Sorgfalt und sollte nicht nach dem Gießkannenprinzip erfolgen. Dies setzt – auch hier – eine Analyse der betroffenen Applikationen und deren Nutzung voraus, um relevante Operationen – aber nur diese – zu erfassen. Bedingt dadurch sind Audit-Optionen dieser Art nicht nur von den Schutzklassen, sondern auch von den Applikationen abhängig. Sie lassen sich grundsätzlich in folgenden Kontexten über das „audit"-Kommando aktivieren:

- *Objekt-bezogen*
  Nach Statement-Typ. So kann die Protokollierung von „update"-Kommandos auf Tabelle „T1" von Benutzer „X" wie folgt aktiviert werden: „audit update on x.t1;". Dies kann beispielsweise empfehlenswert für sensible Tabellen mit hohem Schutzbedarf sein.
- *Privilegien-Auditing*
  Nutzung von erteilten Systemprivilegien, etwa die Nutzung des „select any table"-Privilegs durch „audit select any table;". Dies ist empfehlenswert, wenn aus guten Gründen Benutzern weitreichende Systemprivilegien erteilt werden müssen, deren Nutzung jedoch

```
SQL> conn audsys/welcome1
ERROR:
ORA-46370: cannot connect as AUDSYS user
```

Listing 1

```
create audit policy <polname> actions delete on scott.emp;
audit policy <polname>;
```

Listing 2

```
cd $ORACLE_HOME/rdbms/lib
make -f ins_rdbms.mk uniaud_on ioracle
```

Listing 3

```
Oracle Database 12c Enterprise Edition Release 12.1.0.1.0 - 64bit Pro-
duction
With the Partitioning, OLAP, Advanced Analytics, Real Application Test-
ing and Unified Auditing options
```

Listing 4

```
EXECUTE DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_PROPERTY( -DBMS_AUDIT_MGMT.AU-
DIT_TRAIL_UNIFIED, - DBMS_AUDIT_MGMT.AUDIT_TRAIL_WRITE_MODE, - DBMS_AU-
DIT_MGMT.AUDIT_TRAIL_IMMEDIATE_WRITE);
```

Listing 5

sicherheitsrelevant ist und einer Kontrolle bedarf.

- *Statement-Auditing*
  Objektunabhängige Nutzung von Statements wie dem „update"-Befehl.
- *Network*
  Netzwerk-Fehler, beispielsweise im Kontext der Netzverschlüsselung.

Die genannten Optionen können pauschal für einzelne Benutzer oder abhängig vom Erfolg oder Nicht-Erfolg ihrer Ausführung gesetzt sein. Darüber hinaus lässt sich festlegen, ob die Protokolle pro Zugriff und Ausführung oder einmal pro Session geschrieben werden.

Es hat sich bewährt – für jede Schutzklasse und unabhängig von den Anwendungen –, Audit-Optionen auf der Basis von Privilegien und Statements festzulegen. Ergänzend dazu ist es sinnvoll, Objekt-bezogene Optionen für jede Applikation, die mit schützenswerten Daten arbeitet, in Zusammenarbeit mit den Applikations-Verantwortlichen zu definieren.

Das sogenannte „Fine Grained Auditing" (FGA) schreibt Protokolle in Abhängigkeit von Inhalten, also beispielweise immer dann, wenn für die Tabelle „EMPLOYEES" des Schemas „HR" die Spalte „SALARY" im Rahmen von „INSERT"- oder „UPDATE"-Statements verändert wird. FGA wird unabhängig von dem oben beschriebenen Audit-Trail über das Paket „DBMS_FGA" und dort im Rahmen von Policies administriert.

Event-Handler gestatten zusätzliche Aktionen, wie beispielsweise die Versendung von Mail-Nachrichten. FGA-Daten werden in einer eigenen Tabelle („SYS.FGA_LOGS$") oder im XML-Format in das über „AUDIT_FILE_DEST" spezifizierte Verzeichnis geschrieben. Die Nutzung des „SYSLOG"-Daemons ist in diesem Falle nicht möglich.

Unified Auditing

In der aktuellen Datenbank-Version 12*c* ist die Audit-Funktionalität maßgeblich erweitert und nachhaltig verbessert worden.

Unter dem Terminus „Unified Auditing" sind unterschiedliche, bisher getrennt verarbeitete Bereiche zusammengeführt sowie zusätzliche Möglichkeiten geschaffen, um Audit-Optionen zu verwalten. Die Neuerungen:

- Die Aktivierung erfolgt nicht über einen „init"-Parameter, sondern über das Linken des Datenbank-Kernel.
- Audit-Optionen lassen sich über Policies bündeln und als Ganzes aktivieren.
- Neben den aus dem klassischen Auditing bekannten Optionen aus dem vorangehenden Abschnitt werden nun auch die Protokolle von SYS-Auditing, Data Pump, RMAN, Database Vault, Direct Load, Label Security und Real Application Security im Unified-Audit-Trail zusammengeführt.
- Die Daten werden in einer eigenen „read only"-Tabelle im Schema „AUDSYS" in der „SYSAUX"-Tablespace in Form von BLOBs gespeichert. Das Schema „AUDSYS" ist darüber hinaus resistent gegenüber jeglichen Connect-Versuchen, auch nachdem das Passwort angepasst und der Account geöffnet wurde *(siehe Listing 1)*.
- Wenn die Datenbank heruntergefahren ist, werden die Daten zunächst in das Filesystem geschrieben und können von dort aus nachträglich in den Audit-Trail geladen werden.

Die anfallenden Daten werden standardmäßig nicht direkt, sondern über eine eigene Queue periodisch in die Tabelle geschrieben. „Direct Writes" können jedoch konfiguriert werden, ebenso explizite „Flushes", die zur Kontrolle bei Tests hilfreich sind.

Neu angelegte oder migrierte Datenbanken finden sich – was das Auditing betrifft – standardmäßig zunächst im sogenannten „Mixed Mode". Hier ist sowohl das klassische Auditing aktiv als auch das neue Unified Auditing in Form einer vordefinierten Policy namens „ORA_SECURECONFIG". Beide Welten existieren völlig unabhängig voneinander, werden auch getrennt verwaltet und schreiben ihre eigenen Daten.

Der Befehl „audit delete on scott.emp" aktiviert beispielsweise das klassische Auditing für „delete"-Operationen auf der Tabelle „EMP" von „SCOTT", hat aber keinen

Einfluss auf den Unified-Audit-Trail. Um dort die gleiche Operation zu protokollieren, sind die folgende Schritte notwendig *(siehe Listing 2)*. Man beachte, dass jede Audit Policy über ein eigenes „audit"-Kommando aktiviert werden muss. Im Einzelnen wird der Mixed Mode folgendermaßen erkannt:

- Die View „v$option" zeigt für Unified Auditing „FALSE" an, die Option wurde also nicht gelinkt.
- Die Policy „ORA_SECURECONFIG" existiert und ist aktiviert (View „AUDIT_UNIFIED_ENABLED_POLICIES").
- Die Audit-Parameter in der „init.ora" stehen auf 11*g*-Standard („AUDIT_TRAIL=DB" etc.).
- Der init-Parameter „unified_audit_sga_queue_size" ist gesetzt und legt die Größe der Audit-Queue fest.

Um Unified Auditing vollumfänglich zu konfigurieren, muss die Datenbank heruntergefahren, der Kernel gelinkt und danach die Datenbank neu gestartet werden *(siehe Listing 3)*. *Listing 4* zeigt, wie sich der Banner durch diese Aktion geändert hat.

Im Zuge der oben beschriebenen Aktion wird die Rolle „AUDIT_ADMIN" angelegt, die „select"-Operationen auf den Audit Views, „execute"-Rechte für die Pakete „DBMS_FGA" und „DBMS_AUDMGMT" sowie die Systemprivilegien „AUDIT SYSTEM" und „AUDIT ANY" enthält. Die Rolle wird standardmäßig an „SYS" vergeben. Darüber hinaus wird die Rolle „AUDIT_VIEWER" mit Select-Rechten für die Audit-Views erzeugt.

Mit der Umstellung verlieren sämtliche klassischen Audit-Parameter ihre Wirksamkeit. Die Tabellen und Views existieren zwar weiter, ebenso wie ihre Daten. Klassische Audit-Optionen sind nach wie vor gesetzt, aber wirkungslos. Besonders gewöhnungsbedürftig ist, dass neue Optionen in klassischer Syntax gesetzt werden können, auch fehlerlos akzeptiert werden, jedoch – wie bereits erwähnt – keine Auswirkungen haben, also keine Audit Records generieren, obwohl die Optionen in den klassischen Views, wie „DBA_STMT_AUDIT_OPTS", angezeigt sind. Für die Administration des Unified Audit-Trail steht das Paket „DBMS_AUDIT_MGMT" zur Verfügung, beispielsweise um den direkten Modus zu aktivieren *(siehe Listing 5)*.

Weitere Möglichkeiten

Die vorstehend beschriebenen Möglichkeiten des Auditing können durch weitere Techniken und Werkzeuge erweitert werden. Diese sind aus Platzgründen nur stichwortartig und beispielhaft erwähnt und verdienen eine eigene Betrachtung:

- Im Rahmen des sogenannten „Applikation Auditing" lassen sich beispielsweise Werte-Entwicklungen ganzer Datensätze oder einzelner Spalten nachvollziehen. Hier bieten sich Trigger-Techniken, Journaling-Tabellen oder diverse Flashback-Techniken inklusive Flashback Data Archive an. Es versteht sich, dass diese Möglichkeiten eng mit dem Design und der Entwicklung der jeweiligen Applikationen verbunden sind und nicht nachträglich durch Administratoren hinzukonfiguriert werden können.
- Die zentrale Speicherung und Auswertung von Datenbank Audit Records kann auch über Audit Vault realisiert werden, das neuerdings zusammen mit Database Firewall vermarktet und lizenziert wird.
- Das Produkt Database Activity Monitoring (DAM) von McAffee bietet vergleichbare Möglichkeiten. Zusätzlich stehen hier jedoch auch virtuelle Patch-Techniken zu Verfügung.

Fazit

Für die Konzeption und Konfiguration von Auditing im Umfeld von Oracle-Datenbanken stehen vielfältige Techniken zur Verfügung, die sorgfältig aufeinander abgestimmt werden müssen, um ein schlüssiges Gesamtkonzept zur Nachvollziehbarkeit sicherheitsrelevanter Aktionen zu ergeben. Der Blick muss dabei neben der Datenbank ebenso auf das Betriebssystem und das Netzwerk gerichtet sein. Die Klassifizierung von Systemen hilft dabei, die Aufwände zu reduzieren.

Dr. Günter Unbescheid
g.unbescheid@database-consult.de