Wie mache ich meine Oracle Datenbank fit fürs PCI Audit?

Angela Espinosa
Lufthansa Systems AG
Kelsterbach

Schlüsselworte

PCI DSS und CIS Standard, Härtungsregeln, Auditing Tipps und Kniffe, SIEM Smartconnector für Oracle, FIM

Einleitung

Die neue Security Awareness macht es uns schwer durch ein PCI Audit zu kommen ohne sich auf dem Gebiet der Datenbank Security inklusive angeschlossenen Systemen (SIEM oder Platformen) auszukennen. Dieser Vortrag soll eine kleine Einführung geben, wie man sich besser auf ein PCI Audit im Datenbankbereich (Oracle) vorbereitet und dieses auch erfolgreich bestehen kann (auch mit den Stolpersteinen, die uns z.B. Oracle seitig gestellt werden).

Definition PCI DSS und warum braucht man überhaupt PCI Compliance?

PCI DSS = Payment Card Industry Data Security Standard, ist ein Regelwerk im Zahlungsverkehr, das sich auf die Abwicklung von Kreditkartentransaktionen bezieht und von allen wichtigen Kreditkartenorganisationen unterstützt wird. Handelsunternehmen und Dienstleister, die Kreditkarten-Transaktionen speichern, übermitteln, oder abwickeln, müssen die Regelungen erfüllen. Halten sie sich nicht daran, können Strafgebühren verhängt, Einschränkungen ausgesprochen, oder ihnen letztlich die Akzeptanz von Kreditkarten untersagt werden. Die Regelungen bestehen aus einer Liste von zwölf Anforderungen an die Rechnernetze der Unternehmen.

Aller Anfang ist schwer

Man ist nicht auf einen Schlag „PCI compliant“. Es ist ein Lernprozess, von Mal zu Mal verbessert man Prozesse und die Sicherheit in der Datenbank. Zuerst schafft man eine Basis auf der man die PCI Anforderungen aufsetzen kann. Dies umfasst Härtungsregeln und Prozesse, die erweitert werden müssen. Folgende Dokumente von Organisationen können bei der Definition von Härtungsregeln unterstützen. Das Center for Internet Security bietet Security Benchmarks für Datenbanken auch für Oracle. Erhebliche Verbesserungen in der Qualität des Dokuments wurden durch das Mitwirken von u.a. Alexander Kornbrust erzielt.

Die PCI DSS Regeln als solches geben viele Punkte vor. Ab November 2013 wird es eine neue Version 3.0 geben, die zahlreiche neue Unteranforderungen enthält. Die aktuelle Version 2.0 wird bis 31. Dezember 2014 aktiv bleiben und darauf werde ich mich beziehen.

Zusätzlich stellt sich die Frage, wie die Kreditkartendaten in der Datenbank abgelegt sind. Erfolgt die Verschlüsselung schon durch die Applikation oder muss dies durch die Datenbank selbst gewährleistet werden. Wenn verschlüsselte Daten in der Datenbank liegen, verringert sich der Aufwand der Härtung. Sofern jedoch die Daten in Klartext abgelegt werden, muss man zusätzliche Sicherheitsmaßnahmen ergreifen, wie zum Beispiel die Installation und Konfiguration von Oracle Database Vault und der Oracle TDE (Transparent Data Encryption), sowie die Berücksichtigung deren

Besonderheiten im Betrieb. Diese Themen würden einen weiteren Vortrag mühelos füllen und können hier leider nicht berücksichtigt werden.

Härtung der Datenbank - Anforderungen aus PCI DSS 2.0 für die Oracle Datenbank

Im weiteren Verlauf werden auszugsweise aus den PCI DSS 2.0 Anforderungen Härtungsregeln und Umsetzungen abgeleitet.

Anforderung 2: Keine vom Anbieter gelieferten Standardeinstellungen für Systemkennwörter und andere Sicherheitsparameter verwenden

Dieser Punkt ist wichtig, da es sonst ein Angreifer sehr einfach hat, in die Datenbank einzubrechen. Deshalb ist es unumgänglich, Standardpasswörter aller Oracle internen Benutzer zu ändern, Standardeinstellungen wie Listenerport 1521 und Datenbankname zu ändern, Demo- oder Testschemas zu löschen und nicht benötigte Features (Minimalinstallation) erst gar nicht zu installieren.

Alle Nichtkonsolen-Verwaltungszugriffe müssen mittels einer starken Verschlüsselung (SSH, VPN, SSL/TLS) erfolgen. Also darf keine Remote SQLNET Verbindung stattfinden, sofern nicht Advanced Security aktiviert ist. (ab 11.2.0.4 inklusive).

Die Datenbank muss durch Setzen von Sicherheitsparametern entsprechend gehärtet werden. Beispielsweise erlaubt oder verbietet der Parameter `07_DICTIONARY_ACCESSIBILITY` den Zugriff auf Objekte im SYS Schema durch die EXECUTE ANY PROCEDURE und SELECT ANY DICTIONARY Berechtigung und sollte aus Sicherheitsgründen auf FALSE stehen. `REMOTE_LOGIN_PASSWORDFILE` beschreibt, ob beim Login ein Passwortfile benutzt werden soll und wie viele Datenbanken dieses File benutzen können und sollte auf NONE stehen. `SEC_CASE_SENSITIVE_LOGON` gibt Aufschluss darüber, ob die Passwörter „case sensitive“ gespeichert werden oder nicht. Durch eine Schwachstelle (CVE-2012-3137), die im CPU/PSU Oktober 2012 behoben wurde, konnten Passwörter geknackt werden, wenn dieser Parameter auf TRUE stand. `SEC_MAX_FAILED_LOGIN_ATTEMPTS` bestimmt, wie oft man sich mit falschem Passwort einloggen darf, bevor Oracle die Verbindung kappt. Diese Einstellung kann durch Verwendung von Profilen, die man jedem Benutzer zuweist, „überschrieben“ und sollte auf 6 gesetzt werden. `SEC_RETURN_SERVER_RELEASE_BANNER` liefert die Information der Release- und Patchnummer. Damit kann ein Angreifer besser vorbereitet vorgehen und bekannte Schwachstellen in einem Release ausnutzen. Deshalb sollte dieser auf FALSE gesetzt werden. `SQL92_SECURITY` steht standardmäßig auf FALSE und sollte auch nicht geändert werden, da sonst Benutzer, die nur UPDATE oder DELETE Privilegien brauchen, auch SELECT direkt auf Daten ausführen können.

Unbedingt wichtig ist es, Ausführungsrechte von PUBLIC auf Pakete und Objekte in Absprache mit der Applikation zu entfernen. PUBLIC an sich hat sehr viele mächtige Rechte, die unbedingt eingeschränkt werden müssen. Zusätzlich sollten die Benutzer (auch Oracle interne Benutzer) selbst in ihren Möglichkeiten begrenzt werden und so wenige Privilegien (ANY, DBA-Rechte) wie möglich erhalten.

Weiterhin muss der Listener sicher konfiguriert werden. Beispielsweise sollte der `ADMIN_RESTRICTIONS_<listener_name>` Parameter auf ON gesetzt sein, damit der aktive Listener nicht geändert werden kann in seinen Einstellungen außer über das Ändern in der listener.ora und dem manuellen Restart. `SECURE_REGISTER_listener_name` gibt an, welche Protokolle sich auf den Listener verbinden dürfen. IPC oder TCPS.

Anforderung 6: Entwicklung und Wartung sicherer Systeme und Anwendungen

Die Datenbank muss mittels Patchen/Releasewechsel immer auf den neuesten Stand gehalten werden inklusive der Sicherheitspatches (spätestens Einspielen innerhalb eines Monats). Hierzu sollte auch ein Prozess etabliert und dokumentiert sein.

Mit diesem Statement kann der Patchstand abgefragt werden:

```
SELECT * FROM DBA_REGISTRY_HISTORY;
```

Anforderung 8: Zuweisung einer eindeutigen ID für jede Person mit Computerzugrif

Alle Nichtverbraucherbenutzer und Administratoren müssen personalisiert in der Datenbank vorhanden sein und sich authentifizieren durch Passwort (kein Sammelaccount). Außerdem sollten diese Benutzer nicht den „/ as sysdba“ auf Betriebssystemebene ausführen können.

Zusätzlich muss durch Profile (und der Zuweisung derer zu Benutzern) und Einrichten einer Passwort Verify Funktion sichergestellt werden, dass sichere Passwörter verwendet werden, die nach 90 Tagen ablaufen, mindestens 7 Zeichen lang sind, sowie alphabetische und numerische Zeichen enthalten. Zusätzlich müssen sich die letzten vier Kennwörter unterscheiden. Nach 6 Mal falscher Eingabe des Passwortes, muss der Benutzeraccount gesperrt werden. Sofern ein Benutzer angemeldet ist, aber 15 Minuten inaktiv, muss eine Neuanmeldung geschehen.

Anforderung 10: Verfolgung und Überwachung des gesamten Zugriffs auf Netzwerkressourcen und Karteninhaberdaten

Nachvollziehbarkeit ist bei PCI DSS ein großes Thema. Automatisierte Audit-Trails müssen eingerichtet werden, um folgende Ereignisse für alle Systemkomponenten rekonstruieren zu können. Alle individuellen Zugriffe auf Karteninhaberdaten und alle von einer Einzelperson mit Root- oder Administratorrechten vorgenommene Aktionen, sowie der Zugriff auf die Audit-Trails muss dokumentiert werden (mittels bspw. Linux Auditing). Ungültige logische Zugriffsversuche, das Verwenden von Identifizierungs- und Authentifizierungsmechanismen und die Initialisierung der Audit-Protokolle sowie das Erstellen und Löschen von Objekten auf Systemebene muss protokolliert werden. Nachfolgende Einträge müssen die Audit-Trails enthalten: Benutzeridentifizierung, Ereignistyp, Datum und Uhrzeit, Angabe von Erfolgen oder Fehlern, Ereignisursprung, Identität oder Namen der betroffenen Daten, Systemkomponenten oder Ressourcen. Zusätzlich müssen die Audit-Dateien vor Veränderungen geschützt werden, der Zugriff auf diese muss dementsprechend eingeschränkt werden und darf nur von der Benutzergruppe eingesehen werden, die aus geschäftlichen Gründen darauf zugreifen muss. Außerdem müssen sie möglichst schnell auf einen zentralen Protokollserver gesichert werden.

Das sind viele Anforderungen, die dank Oracle (auch ohne AuditVault), wenn auch mit ein paar Hürden bewältigt werden können. Verschiedene Audit Einstellungen in ihrer Kombination führen zum Erfolg. Zuerst beginnt man mit den privilegierten Benutzern. Der `AUDIT_SYS_OPERATIONS` Parameter muss auf TRUE gesetzt sein, damit alle Operationen die von Accounts mit den Privilegien SYSDBA oder SYSOPER durchgeführt, geloggt werden. `AUDIT_TRAIL` ist wichtig, um einzustellen, wohin (DB Tabelle, Filesystem oder syslog) und wie die Auditdaten geloggt werden. `AUDIT_TRAIL=OS` lässt eine Weiterleitung an den syslog Daemon des Betriebssystems zu, wenn `AUDIT_SYSLOG_LEVEL` entsprechend gesetzt wird z.B. auf `LOCAL2.WARNING`.

Jedoch werden andere (nicht privilegierte) Benutzer nicht vollständig (mit SQL Statement, sondern nur mit SQL Bind und Text ID) geloggt. Für PCI ist das aber eine Voraussetzung. Um das zu

erreichen, muss man folgenden Trick anwenden. `AUDIT_TRAIL=XML, EXTENDED` liefert alle wichtigen Informationen zu Benutzern und deren Statements, jedoch nicht an den syslog Deamon. Die XML Dateien werden zusätzlich in der `AUDIT_DUMP_DEST` (sollte in einem eigenen Filesystem liegen, nicht im Oracle Home) erzeugt. Und schreiben fleißig das Filesystem voll. Außerdem dürfen diese XML Dateien nicht von dem Oracle Benutzer manipulierbar sein. Dafür ist es wichtig, dass dieser Standardparameter *`_TRACE_FILES_PUBLIC=FALSE`* nicht geändert wird. Dieser beschreibt, welche Berechtigung die von Oracle erzeugten Logdateien erhalten. Dann sind diese nur vom Oracle Software Owner beschreibbar, von keinem anderen Benutzer oder Gruppe. Was heißt das im Umkehrschluss? Erstens, man muss den Oracle Software Owner einschränken in seiner Benutzung. Und zweitens, wie schon vorher erwähnt, darf „/ as sysdba“ nicht „regelmäßig“ benutzt werden, da dann die Nachvollziehbarkeit „gestört“ ist. Dafür muss man einen Prozess etablieren, der beschreibt, wann dieses Szenario stattfindet und wie man das dokumentiert. Um normal (auf Betriebssystemebene) arbeiten zu können, muss man einen weiteren Betriebssystembenutzer einrichten, der eingeschränkter in seinen Rechten ist, also nicht der dba Gruppe angehören darf. Damit ist auch der Zugriff auf die Audit-Dateien eingeschränkt und diese sind nur noch für diesen Benutzer lesbar. Nur was tut man jetzt mit diesen XML Dateien, die das ganze Filesystem „vollmüllen“? Diese Dateien müssen an den zentralen Protokollserver geschickt werden. Dafür hat ArcSight beispielsweise einen Connector (SmartConnector for Oracle Audit XML Connector) zur Verfügung gestellt, der diese XML Dateien verarbeiten kann. Wie man die Dateien dann dorthin bekommt, steht auf einem anderen Blatt. Folgende Lösung hat sich als praktikabel erwiesen. Über einen Share können die Daten empfangen werden. Der Share wird an das Betriebssystem gemounted und darf nur von root zugreifbar sein. Die XML Dateien müssen regelmäßig in diesen Share übertragen werden (auch nur durch root) und sind somit auch nicht mehr manipulierbar durch den Oracle Software Owner und schreiben auch nicht das Filesystem voll, da sie verschoben werden.

Wenn alle Events nach SIEM geloggt werden, kann man hier auch definieren, wann etwas auffällig oder verdächtig ist und entsprechend konfigurieren, wann ein Alarm ausgelöst werden soll. Beispielsweise sollte man alarmiert sein, wenn ein Benutzer einen Datenbankparameter ändert oder Benutzer angelegt werden, nicht nur durch das konventionelle *`create user`*, etc. Hier gibt es genügend Anregungen durch u.a. Herrn Kornbrust (*`grant user identified by`*, *`become user`*,…), die diese Konfiguration beliebig erweitern können.

Um Veränderungen an statischen Dateien erkennen zu können, muss eine Software zur Dateiintegritätsüberwachung eingesetzt werden. FIM (File Integrity Monitoring) kann auf Linux Ebene via samhain eingerichtet werden. Hier ist es wichtig, als Datenbankverantwortlicher Input zu liefern, damit auch diese Anforderung erfüllt und Sicherheit gewährleistet werden kann. Programm-, Bibliotheks-, Skript- und Konfigurationsdateien müssen von der Software überwacht werden, sich ständig ändernde Dateien wie Protokolldateien und Datenbank Datendateien dürfen herausgenommen werden. Die Überprüfung erfolgt gegen eine Datenbank von Prüfsummen, die bei der initialen Installation erstellt und nach jedem eingespielten Softwareupdate aktualisiert wird. Die FIM-Prüfsummendatenbank muss selbst gegen unberechtigte Veränderungen geschützt sein. Werden Abweichungen von den gespeicherten Prüfsummen festgestellt, müssen diese Ereignisse an den zentralen Protokollierungsserver gemeldet werden.

Zusätzlich zu den technischen Details ist es wichtig, Prozesse zu definieren und zu dokumentieren, wie man zum Beispiel mit Neuzugängen und Abgängen im Team umgeht, wer was nachweislich genehmigt und in welcher Form. Gibt es regelmäßige Überprüfungen des Benutzerbestandes in der Datenbank in Abgleich mit den zulässigen Datenbankadministratoren, werden die Privilegien und Härtungsregeln geprüft, was passiert, wenn ein Alarm ausgelöst wird und so weiter. Es stellen sich viele Fragen, je nachdem welcher Auditor vor einem sitzt und wie tief dieser in der Materie steckt.

Das Thema PCI stellt interessante Herausforderungen und regt an, über das Thema Sicherheit in der Datenbank mehr nachzudenken als normal üblich. Und darüber hinaus kann man auch die Systeme verbessern, die nicht ganz so schützenswert sind.

Quellen:

http://de.wikipedia.org/wiki/PCI_DSS

http://www.cisecurity.org/

http://benchmarks.cisecurity.org/downloads/show-single/?file=oracle11gr2.100

https://www.pcisecuritystandards.org/documents/DSS_and_PA-DSS_Change_Highlights.pdf

https://www.pcisecuritystandards.org/documents/pci_dss_de-de_v2.pdf

Kontaktadresse:
Angela Espinosa
Lufthansa Systems AG
Am Weiher 24
D-65451 Kelsterbach

Telefon: +49 (0) 69-696 91904
E-Mail angela.espinosa@lhsystems.com
Internet: www.lufthansa-systems.com