T4 und The Red Crypto Stack

Stefan Hinker & Heinz-Wilhelm Fabry
Oracle Deutschland B.V. & Co. KG
München

Schlüsselwörter
Sicherheit, Verschlüsselung, Solaris, Oracle Datenbank, SPARC T4

Einleitung
Verschlüsselung ist ein wichtiger Bestandteil jeder Sicherheitsarchitektur. Gleichzeitig sind sowohl der häufig hohe Resourcenverbrauch als auch die oft komplexe Konfiguration Hemmnisse, die den Einsatz von Verschlüsselung oft verhindern oder zumindest erschweren. In diesem Vortrag wird an einem Beispiel gezeigt, wie beide Bereiche durch innovative Technologien und die hohe Integration durch gesamtheitliche Entwicklung adressiert werden. Dabei wird sowohl auf die Features von Hardware in der SPARC T4 CPU als auch von Software wie dem Solaris Cryptographic Framework oder der Oracle Datenbank eingegangen.

Am Beispiel der Oracle Datenbank wird in diesem Vortrag gezeigt, wie Verschlüsselung in allen Schichten der Implementierung umgesetzt wird. In Anlehnung an das bekannte „Red Stack“ Diagram werden dabei folgende Komponenten betrachtet:

- Die SPARC T4 CPU mit der Hardwareunterstützung für Verschlüsselung
- Solaris und das Cryptographic Framework
- IPsec
- ZFS und verschlüsselte Dateisysteme
- Immutable Zones als sichere Ablaufumgebung
- Oracle TDE für verschlüsselte Datenbanken
- Oracle ASO mit SSL-Verschlüsselung für sicheren Client-Zugang

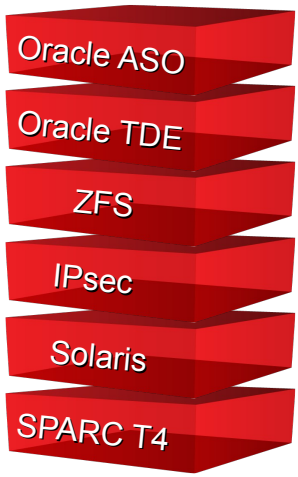

Abbildung 1: The Red Crypto Stack

SPARC T4 – Die Vierte Generation CMT mit Hardware Crypto

Mit der Einführung der SPARC T4 CPU hat Oracle die CMT Architektur grundlegend überholt. Insbesondere die Single Thread Leistung dieser CPU profitiert durch das vollständig neue Kern-Design und ist gleichauf mit aktuellen CPUs mit x86 oder Power Befehlssatz. Auch die Crypto Units wurden erneut überarbeitet und unterstützen nun insgesamt 16 verschiedene Algorithmen. Dabei wurde auch der Zugang zu diesen Algorithmen durch eine Erweiterung des SPARC Befehlssatzes vereinfacht, was zu einer weiteren Leistungssteigerung führt.

Das Solaris Cryptographic Framework

Als Bindeglied zwischen den Crypto-Funktionen der Hardware und deren Nutzern dient das Solaris Cryptographic Framework.

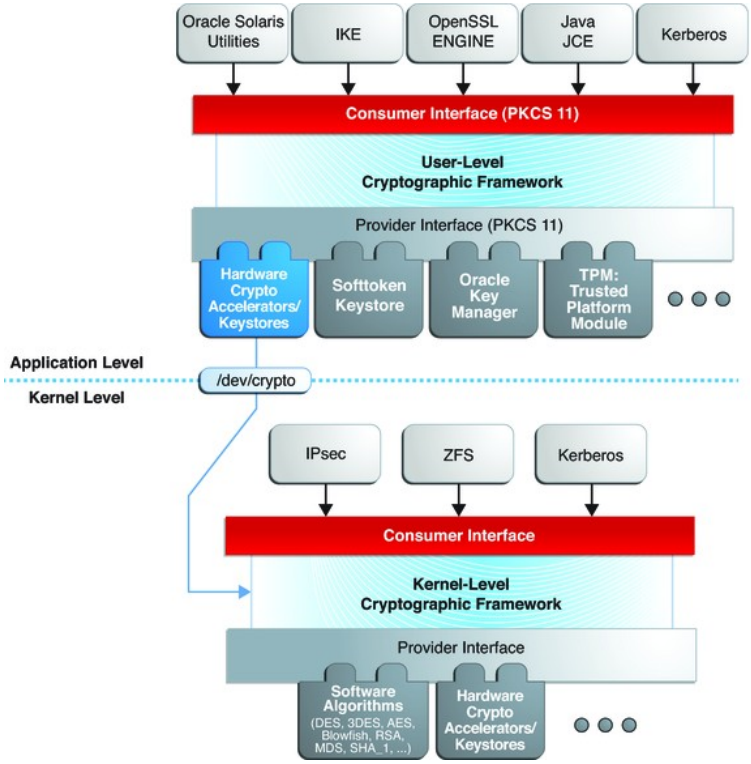

Abbildung 2: Das Solaris Cryptographic Framework

Wie in Abbildung 2 dargestellt, verbindet es Anbieter cryptografischer Dienste, bspw. der T4 CPU, mit Abnehmern üblicherweise über die standardisierte Schnittstelle PKCS#11. Kernel-Dienste des Betriebsystems, wie bspw. IPsec oder die Verschlüsselung von ZFS nutzen dabei das Kernel Programmer Interface – einen direkten Hardware-Zugang im Kontext des Kernels, um die Hardware-Beschleunigung zu verwenden. Für nicht-priviligierte Anwendungen steht PKCS#11 zur Verfügung. Auf diese Weise nutzt bspw. der Apache Webserver über die PKCS#11-Engine von OpenSSL die Hardware-Beschleunigung der T-Serie CPUs. Java Anwendungen nutzen mittels der Java Cryptographic Extensions (JCE) ebenfalls PKCS#11 und damit die Hardware-Beschleunigung.

Neu bei T4 ist, dass die Crypto-Funktionen über eine Erweiterung des SPARC Befehlssatzes direkt im User-Kontext einer Anwendung zur Verfügung stehen und nicht mehr nur, wie bei den Vorgängern, über ein Kernel-Modul. Hiermit wurde u.A. eine neue Crypto-Engine „T4“ für OpenSSL entwickelt, die in Solaris 11 zur Verfügung steht. Auch die Verschlüsselung von Daten mit Oracle TDE macht von diesem Weg gebrauch. Der Vorteil hierbei ist ein wesentlich kürzerer Code-Pfad mit entsprechend weniger Overhead, was insb. bei kleinen Datenpaketen vorteilhaft ist.

Entsprechend beeindruckend sind die Leistungen der Verschlüsselung mit SPARC T4. In synthetischen Benchmarks[1] mit verschiedenen Varianten von AES war die T4 CPU mit 2.85 GHz zwischen 9% und 3.6x schneller als eine Intel Xeon X5690 CPU mit 3.47GHz. In etwas wirklichkeits-näheren Tests mit ZFS Verschlüsselung[2] beträgt der Unterschied zwischen 3.5x und 5.2x. Dabei ist der

1 https://blogs.oracle.com/BestPerf/entry/20110930_sparc_t4_crypto_umbk
2 https://blogs.oracle.com/BestPerf/entry/20110930_t4_zfs_encryption

Vergleich mit Intel und dem Algorithmus AES noch besonders herausfordernd, weil auch Intel in den aktuellen Xeon CPUs Hardware-Unterstützung für diesen Algorithmus integriert hat. Bei anderen Algorithmen wie bspw. RSA oder SHA sind die Unterschiede noch wesentlich gravierender – hier ist die T4 CPU die derzeit einzige CPU mit entsprechender Hardware-Beschleunigung.

Verschlüsselte Dateisysteme – Schutz für Metadaten, Audit Trails und Konfigurationsdateien

Verschlüsselung ist immer nur ein Bestandteil von vielen in einem Sicherheitskonzept. Das Verschlüsseln eines Dateisystems ist kein Ersatz für eine sorgfältige Rechteverwaltung für die Dateien dieses Dateisystems. Einmal „geöffnet" (mounted), funktioniert dieses Dateisystem genau wie eines ohne Verschlüsselung. Die Verschlüsselung schützt jedoch sehr effektiv gegen Angriffe von außerhalb des Systems: Daten auf einem SAN-Kabel können nun nicht mehr abgehört werden, Daten auf einer defekten und ausgetauschen Festplatte sind und bleiben geschützt.

ZFS[3] bietet eine transparente und flexible Verschlüsselung einzelner Dateisysteme. Dabei nutzt es, soweit verfügbar, die Hardware-Beschleunigung der T4 CPU. Bereits bestehende Dateisysteme können nicht nachträglich verschlüsselt werden. Bestehende Daten müssen daher ggf. in ein neues, verschlüsseltes Dateisystem übertragen werden. Zur Verschlüsselung werden derzeit zwei verschiedene Modi von AES angeboten: AES CCM, welches zusätzlich die Deduplikation verschlüsselter Dateisysteme erlaubt sowie AES GCM, das im Katalog „Suite B" der NSA[4] veröffentlicht wurde. Die verwendeten Schlüssel können aus unterschiedlichen Quellen stammen. Derzeit unterstützt wird neben der interaktiven Passwort-Abfrage ein Zugriff mittels PKCS#11, https sowie aus Dateien und der Oracle Key Management Station. Ein Auswechseln des Schlüssels ist jederzeit möglich und kann ohne Neuverschlüsselung der Daten erfolgen. Auch weitere Features wie Deduplikation oder Kompression werden gemeinsam mit Verschlüsselung unterstützt.

Solaris Immutable Zones als sichere Ablaufumgebung für Anwendungen

Solaris Zonen waren bereits in Solaris 10 die empfohlene Umgebung zur Isolierung von Anwendungen. Zonen bieten ein Höchstmaß an Isolierung und Schutz für die Globale Zone, ohne dabei Kompromisse bei Skalierbarkeit oder IO-Leistung einzugehen. Das Sicherheitskonzept der Zonen wurde in Solaris 11 mit der Einführung der „Immutable Zones" weiter verstärkt. Damit ist es nun in verschiedenen Abstufungen möglich, Teile des der Zone zugänglichen Dateisystems, insbesondere Betriebssystemdateien und deren Konfiguration von außen mit einem Schreibschutz zu versehen und so gegen versehentliche oder bösartige Manipulationen zu schützen. Dieses Prinzip läßt sich auch auf die Programm- und Konfigurationsdateien einer Anwendung wie bspw. der Oracle Datenbank anwenden.

Sicherheit für Daten in der Datenbank: TDE – mit T4 Hardware Support

TDE steht für Transparent Data Encryption und ist Bestandteil der Advanced Security Option (ASO) der Datenbank. Alle Bestandteile der ASO können ausschließlich mit einer Enterprise Edition der Datenbank (EE) eingesetzt werden.

TDE wurde in Oracle Database 10g zunächst als Möglichkeit eingeführt, einzelne Tabellenspalten zu verschlüsseln. Allerdings gibt es bei dieser Variante eine Reihe von Einschränkungen, zum Beispiel bei den Datentypen, die verschlüsselt werden können. Im Kontext dieses Beitrags ist aber entscheidender, dass bei der Verschlüsselung von Spalten nicht auf das Ver- und Entschlüsselungsangebot der CPUs zurückgegriffen wird.

---

3 http://www.oracle.com/technetwork/articles/servers-storage-admin/manage-zfs-encryption-1715034.html
4 http://en.wikipedia.org/wiki/NSA_Suite_B_Cryptography

Seit der Datenbankversion 11.1 kann man ganze Tablespaces verschlüsseln. Tablespaces sind eine Art Container für beliebig viele, in der Regel logisch verbundene Tabellen. Bei der Verschlüsselung dieser Tablespaces gibt es keinerlei Einschränkungen bezüglich der Datentypen usw. Weil das Ver- und Entschlüsseln hier im Rahmen der Ein- / Ausgabeoperation erfolgt, ist es ohnehin performanter als das Verschlüsseln vieler einzelner Spalten und erklärt ausserdem, warum die CPU hier Unterstützung leisten kann. Es ist deshalb verständlich, dass der Tablespace-Verschlüsselung im Allgemeinen der Vorzug zu geben ist. Zu beachten ist lediglich, dass ein Tablespace nicht nachträglich verschlüsselt werden kann, sondern immer als verschlüsselt angelegt werden muß. Vorhandene Daten müßten also mit geeigneten Verfahren in ein verschlüsseltes Tablespace verschoben werden.

Seit der Datenbankversion 11.2.0.3 nutzt die Datenbank das Verschlüsselungsangebot der T4 CPU automatisch bei der Entschlüsselung der Tablespacedaten. Will man auch bei der Verschlüsselung die CPU Unterstützung nutzen, ist dazu das Einspielen eines Patch (10296641) erforderlich.

Zentrales Element jeder Verschlüsselung ist die Verwaltung und Erzeugung der Schlüssel. Dies erfolgt im Rahmen von TDE immer durch die Datenbank. Das Verfahren ist zweistufig: Jedes Tablespace verfügt über einen eigenen Schlüssel, mit dem die dazugehörigen Daten ver- und entschlüsselt werden. Dieser Schlüssel wird innerhalb des Tablespace gespeichert. Die zweite Stufe ist der sogenannte Master Key, ein Schlüssel, der ausschließlich dazu dient, die Schlüssel der Tablespaces zu ver- und zu entschlüsseln.

Der Master Key wird ausserhalb der Datenbank gespeichert. Entweder in einer verschlüsselten Datei, einem sogenannten Wallet mit dem Namen ewallet.p12, oder ab der Datenbankversion 11.1 in einer speziellen Hardware, einem sogenannten Hardware Security Modul (HSM). Oracle selbst bietet dazu die PCI Karte Sun SCA6000 an, aber es werden selbstverständlich auch HSMs anderer Hersteller unterstützt. Der DBA legt über einen Eintrag in der Datei sqlnet.ora fest, welche Variante genutzt wird. Eine Änderung des Master Key ist mit einem Befehl möglich.

Nach dem Anlegen des Wallet und des Master Key können Tablespaces verschlüsselt werden - ganz einfach indem beim Anlegen die Verschlüsselungsklausel genutzt wird, zum Beispiel so:

```
CREATE TABLESPACE doag2012
DATAFILE '/app/oracle/oradata/dateiname.dbf' SIZE 10G
ENCRYPTION USING 'AES256' DEFAULT STORAGE (ENCRYPT)
```

Alle Daten, die in verschlüsselten Tablespaces eingefügt bzw. dort manipuliert werden, werden unter Beibehaltung der bekannten Befehle ohne weitere Klauseln - in der Oracle-Terminologie 'transparent' - ver- bzw. entschlüsselt. Das bedeutet auch, dass bestehende Anwendungen NICHT geändert werden müssen, um mit verschlüsselten Daten arbeiten zu können.

Leistungsvergleich: TDE auf SPARC T4 und Intel AES-NI

Nicht nur die SPARC T4 CPU unterstützt die Ver- und Entschlüsselung von Daten mit TDE. Die „AES-NI“ genannten Erweiterungen des x86-Befehlssatzes verfolgen einen ganz ähnlichen Ansatz, allerdings ist die Crypto-Unterstützung weniger vollständig. Beide Technologien werden von Oracle TDE gleichermaßen unterstützt. Ein Leistungsvergleich bietet sich daher an. Ein einfacher Test[5] mit Abfragen aus einem 25GB großen Tablespace auf zwei vergleichbaren, kleinen Serversystemen zeigt einen klaren Vorteil von knapp über 40% für die SPARC T4, und zwar unabhängig von der Schlüsselgröße. Beachtenswert ist hier, daß die Intel-CPU einen um ca. 600MHz höheren CPU-Takt nicht zu ihren Gunsten nutzen konnte.

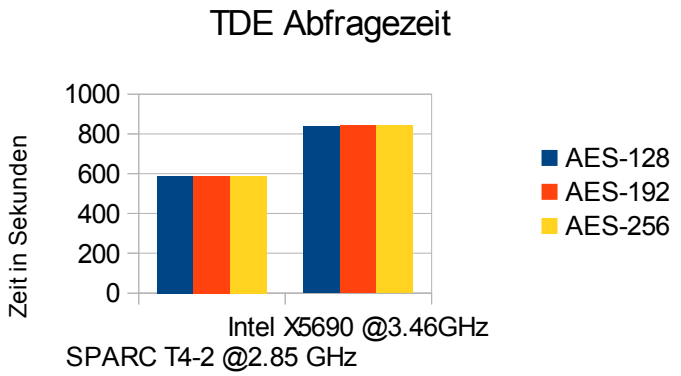

Abbildung 3: TDE Leistungsvergleich SPARC T4 / Intel X5690

Zugangsschicht: ASO und SSL-Verschlüsselter Client-Connect

Ein weiteres Feature von ASO ist die Verschlüsselung des Netzwerkverkehrs von der und zur Datenbank. Dabei ist es unerheblich, ob die Verschlüsselung über das Oracle eigene SQL*Net erfolgt oder über SSL. Ist die Netzwerkverschlüsselung nicht im Einsatz erfolgt der Einsatz von Verschlüsselungstechniken lediglich beim Aufbau einer Verbindung vom Client zur Datenbank. Da die Oracle Datenbank bei der Netzwerkverschlüsselung nicht auf die Ver- und Entschlüsselungsmöglichkeiten der CPU zurückgreift, sollen hier nur der Vollständigkeit halber die beiden verfügbaren Varianten der Netzwerkverschlüsselung skizziert werden.

Die einfachste Variante der Netzwerkverschlüsselung wird über SQL*Net implementiert. Da die Verschlüsselungskomponenten bei jeder Installation einer EE automatisch installiert werden, müssen lediglich Lizenzbestimmungen beachtet werden. Das Einschalten der Verschlüsselung ist im einfachsten Fall für jeden Verbindungsaufbau durch das Einfügen einer einzigen Zeile einer Konfigurationsdatei (SQLNET.ORA) auf der Serverseite möglich. Es ist jedoch genauso einfach, die Verschlüsselung für jeden einzelnen Client unterschiedlich aufzusetzen: Neben einem Eintrag in der Konfigurationsdatei auf der Serverseite ist dazu ein entsprechender Eintrag in den Konfigurationsdateien (ebenfalls SQLNET.ORA) der Clients nötig. Weil die SQL*Net Variante so einfach zu implementieren ist, ist sie die bevorzugte Variante der meisten Oracle Kunden.

Die SSL Verschlüsselung kommt vor allem bei solchen Kunden zum Einsatz, die nicht nur den Netzwerkverkehr zur Datenbank verschlüsseln, sondern die die Verschlüsselung des Netzwerks auch in anderen Bereichen des Unternehmens einsetzen. In der Regel haben solche Kunden dann auch Erfahrungen im Erzeugen, Verteilen und Verwalten von Zertifikaten.

Um die SSL Verschlüsselung für die Datenbank zu nutzern, wird nämlich ein solches Zertifikat auf der Serverseite in einer Datei ausserhalb der Datenbank, einem Wallet, gespeichert. Dazu bietet Oracle zwei Werkzeuge an, den Oracle Wallet Manager (owm) und das kommandozeilenorientierte Werkzeug ORAPKI. In der Datei SQLNET.ORA wird sodann vermerkt, wo dieses Wallet gespeichert ist, und schliesslich wird der Prozess (LISTENER), über den alle Verbindungen zur Datenbank

5 https://blogs.oracle.com/BestPerf/entry/20110927_sparc_t4_2_oracle

aufgebaut werden, so konfiguriert, dass dieser mit der verschlüsselten Verbindung umgehen kann. Auf der Client Seite passiert nun das gleiche - nur dass statt der Listener Konfigurationsdatei die Konfigurationsdatei TNSNAMES.ORA bearbeitet wird. Diese Datei weist dem Client sozusagen den Weg zur Datenbank. Schliesslich wird der Datenbankbenutzer, der die verschlüsselte Verbindung nutzen soll, entweder geändert oder neu angelegt. Der dazu nötige Befehl folgt diesem Muster

```
CREATE USER doag2012 identified externally as 'CN=doag2012, OU=Tech,
O=Oracle, L=Frankfurt, ST=TN, C=IN';
```

Alternative: IPsec zur Absicherung der gesamten Netzwerk-Infrastruktur

IPsec ist eine einfache Möglichkeit, allen oder zumindest ausgewählten Netzwerk-Verkehr zwischen verschiedenen Systemen zu sichern. Systeme, die eine so gesicherte Kommunikation aufnehmen wollen, müssen Zugang zu entsprechenden Sicherheits-Zertifikaten haben, die sie üblicher Weise über den Internet Key Exchange (IKE) erhalten. Bei der Konfiguration von IPsec wird ggf. festgelegt, welche IP-Adressen oder Port-Ranges zwingend verschlüsselt werden müssen bzw. welche unverschlüsselt übertragen werden. Für die Anwendungen, die die Netzwerk-Kommunikation nutzen, ist IPsec vollständig transparent. Auf diese Weise kann bspw. ein Datenbank-Service auf Port 1521 unverschlüsselt arbeiten, der Datenverkehr wird jedoch dennoch mittels IPsec verschlüsselt übertragen. Gleichzeitig kann über das Regelwerk von IPsec und IKE eine genaue Zugangskontrolle bereits auf Netzwerk-Ebene umgesetzt werden, die die Sicherheits-Mechanismen der Anwendung ergänzen. IPsec läßt sich auch in Solaris Zonen benutzen und verwendet jede im System vorhandene Hardware-Beschleunigung vollständig automatisch.

Zusammenfassung: „Security Engineered to Work Together“

„Engineered to Work Together“ ist mehr als nur eine Marketing-Aussage. In diesem Beitrag wurde gezeigt, wie die Sicherheits- und Verschlüsselungs-Mechanismen unterschiedlicher Ebenen des Protokol-Stacks ineinander greifen und einander ergänzen, um eine Gesamtlösung zu realisieren. Dabei ist jede der vorgestellten Komponenten einerseits für sich alleine nutzbar. Andererseits ergeben sich durch ihre Kombination Synergien, die den Gesamtwert der Lösung erhöhen. Verschlüsselung und die daraus zu gewinnende Sicherheit wird so ohne großen Mehraufwand und ohne Einbußen in der Leistung möglich.

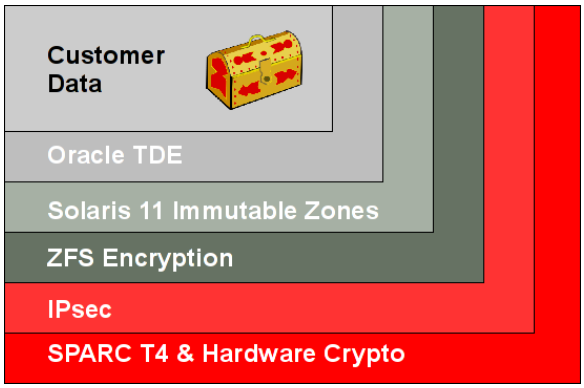

Abbildung 4: The Red Crypto Stack - Layered Protection

Kontaktadresse:
Stefan Hinker & Heinz-Wilhelm Fabry
Oracle Deutschland B.V. & Co. KG
Riesstr. 25
D-80992 München

	Stefan Hinker	Heinz-Wilhelm Fabry
Telefon:	+49 211 7483-9848	+49 89 1430-1534
E-Mail	stefan.hinker@oracle.com	heinz-wilhelm.fabry@oracle.com
Internet:	https://blogs.oracle.com/cmt	http://oracle.com/