# Intrusions Oracle

Philippe Lagadec

DGA/CELAR
philippe.lagadec(at)dga.defense.gouv.fr

**Résumé** Les informations les plus sensibles d'une entreprise sont généralement stockées dans des serveurs de bases de données comme Oracle. Or de tels serveurs sont souvent peu sécurisés, car le « monde Oracle » est extrêmement riche et complexe : un administrateur Oracle a tellement de paramètres à gérer que les aspects sécurité passent souvent au second plan. De plus, l'installation par défaut d'un serveur Oracle comporte de nombreuses vulnérabilités importantes dont un attaquant averti peut facilement tirer profit, parfois même indirectement à travers un pare-feu ou un serveur web intermédiaire. Cet article et la présentation associée ont pour but de montrer concrètement les principales intrusions permettant d'accéder aux données du serveur, ou même de prendre le contrôle du système d'exploitation hôte, en se basant sur les vulnérabilités d'Oracle. Il s'agit bien sûr de sensibiliser les administrateurs et les responsables sécurité aux problèmes spécifiques d'Oracle, et d'apporter des recommandations pour se protéger contre ces intrusions.

## 1 Introduction

Les informations les plus sensibles d'une entreprise sont généralement stockées dans des serveurs de bases de données comme Oracle. Pour de nombreuses bonnes raisons, la sécurisation de ces serveurs est souvent paradoxalement sous-estimée.

En effet, la complexité et la richesse des produits Oracle rend difficile le travail des « DBAs » (administrateurs Oracle) et des développeurs. Il y a déjà tellement de concepts à connaître et d'actions à mener au quotidien pour qu'un système Oracle fonctionne, et la sécurité n'est pas une priorité. En parallèle, les responsables sécurité et les auditeurs techniques ont aussi fort à faire, et ils n'ont pas souvent le temps de se former à Oracle pour comprendre ses problèmes de sécurité spécifiques.

Au final les risques associés à un serveur Oracle sont généralement sous-estimés voire ignorés. En effet, la configuration par défaut d'un serveur Oracle comporte de nombreuses vulnérabilités importantes dont un attaquant averti peut facilement tirer profit, parfois même indirectement à travers un pare-feu ou un serveur web intermédiaire.

Cet article et la présentation associée ont pour but de montrer concrètement les principales intrusions permettant d'accéder aux données du serveur, ou même de prendre le contrôle du système d'exploitation hôte, en se basant sur les vulnérabilités d'Oracle. Il s'agit bien sûr de sensibiliser les administrateurs et les responsables sécurité aux problèmes spécifiques d'Oracle, et d'apporter des recommandations pour se protéger contre ces intrusions.

# 2 Oracle : quelques rappels

Ce chapitre rappelle très rapidement quelques notions de base qui concernent la sécurité d'Oracle.

## 2.1 Bases et comptes utilisateurs

Un serveur Oracle peut contenir plusieurs bases de données. Chaque base contient ses propres comptes utilisateurs. Dans le mode d'authentification par défaut, chaque compte possède normalement un mot de passe protégeant l'accès à la base, ainsi que des privilèges qui restreignent plus ou moins l'accès aux données et aux paramètres de la base.

Toute nouvelle base est dotée d'un certain nombre de comptes par défaut, parmi lesquels SYS et SYSTEM sont les principaux comptes d'administration, avec des privilèges étendus.

Du point de vue de la gestion des utilisateurs, une base Oracle peut donc être considérée comme un système d'exploitation indépendant.

Il existe également d'autres modes d'authentification où Oracle fait confiance au système d'exploitation hôte plutôt que de demander un mot de passe.

## 2.2 Accès réseau : TNS Listener et OracleNet

L'accès aux bases par le réseau passe généralement par un protocole propriétaire sur TCP/IP, nommé SQL*Net, Net8 ou OracleNet suivant les versions d'Oracle. Sur le serveur Oracle, c'est le module TNS Listener (Transparent Network Substrate) qui est chargé de gérer les connexions réseau vers les différentes bases à travers ce protocole. Le port TCP par défaut est 1521, mais Oracle peut très bien employer un autre port. Sur le poste client, l'utilisateur emploie généralement le logiciel client Oracle, soit directement soit par l'intermédiaire d'une application spécifique (interfaces Java, ODBC, ...).

Chaque base Oracle sur le serveur est accessible grâce à un identificateur appelé « SID ». Lorsqu'un client se connecte à une base par le réseau, il doit connaître le serveur, le port TCP du TNS Listener, le SID de la base, ainsi que le login et le mot de passe du compte Oracle utilisé.

Dans certains cas particuliers, il est possible d'employer des alternatives au mot de passe Oracle, comme par exemple l'authentification par le compte utilisateur Unix, ou bien l'authentification native d'un domaine Windows.

Le schéma suivant représente la connexion réseau d'un utilisateur Bob sur une base dont le SID est « BASE1 », après avoir saisi son mot de passe Oracle sur son poste client :

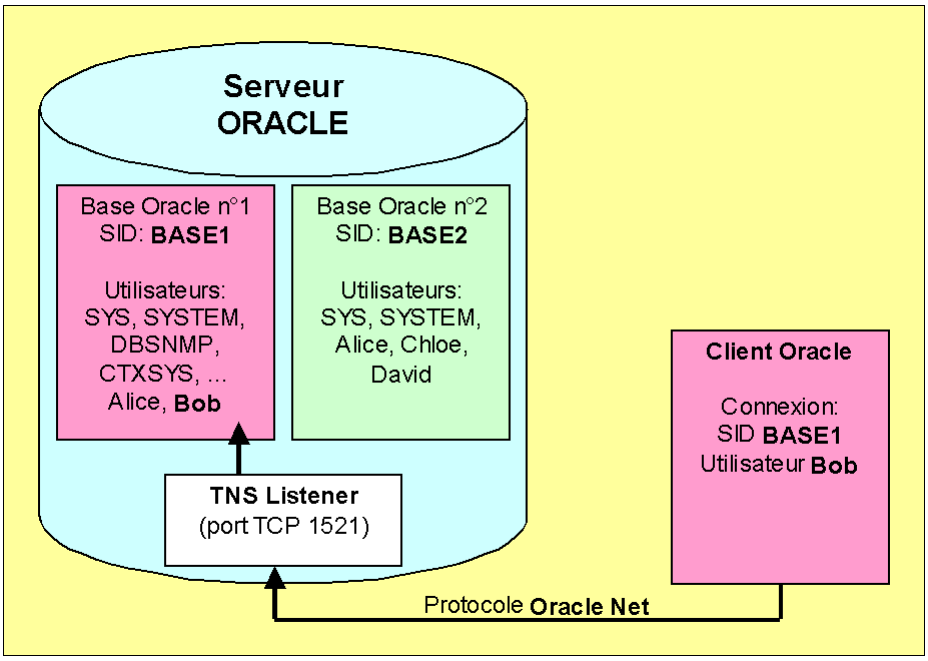


**Fig. 1.** Exemple de connexion réseau Oracle

### 2.3 Applications web 3-tiers

De plus en plus d'applications basées sur Oracle emploient une interface web avec un simple navigateur, ce qui évite de déployer un client lourd sur chaque poste utilisateur. Dans ce cas l'architecture est différente :

Depuis son navigateur, l'utilisateur se connecte sur un serveur web pour accéder à l'application. Suivant la nature de cette application, l'utilisateur n'a pas forcément besoin de s'authentifier. Par exemple un annuaire interne d'entreprise peut être accessible publiquement en lecture sans authentification.

L'application web a besoin d'accéder aux données d'un serveur Oracle, qui peut être sur une machine séparée ou non suivant les cas. Comme dans le cas précédent, l'accès à la base de données nécessite un compte utilisateur Oracle et un mot de passe. C'est donc l'application web qui doit s'authentifier par rapport à la base, et non l'utilisateur. Là aussi plusieurs cas sont possibles : soit l'application web emploie un compte et un mot de passe Oracle qui lui sont propres, soit elle transmet le login et le mot de passe saisis par l'utilisateur dans son navigateur.

Les distributions standard des serveurs Oracle fournissent un serveur web Apache adapté pour accéder directement aux bases de données (accès Java ou module -« mod_plsql"fg, XSQL, ...), mais il existe de nombreux autres serveurs d'application web tiers permettant de bâtir ce type d'architecture.

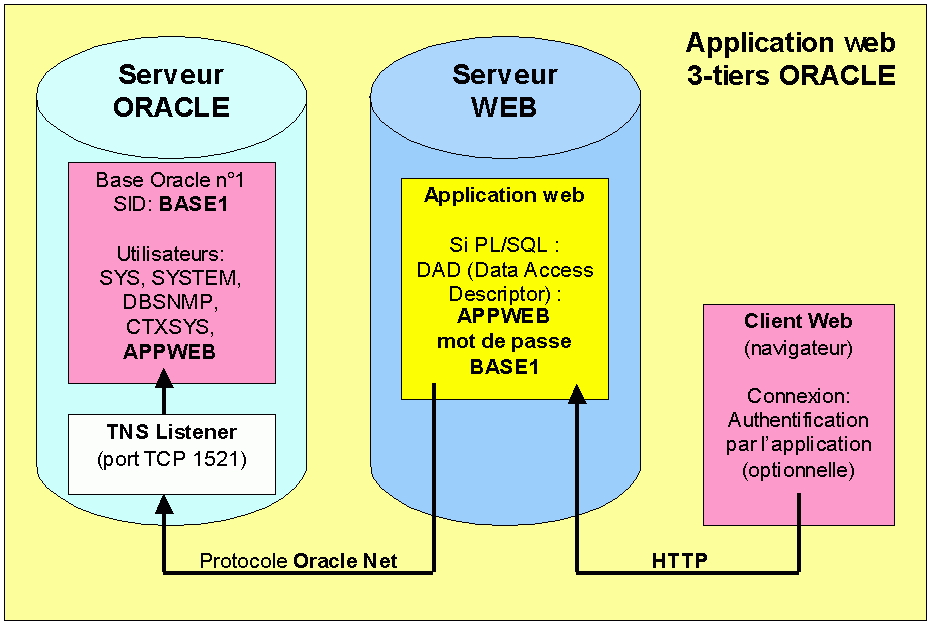


**Fig. 2.** Exemple d'application web 3-tiers Oracle

## 3 Scénarios pour une intrusion Oracle par le réseau (OracleNet)

Nous allons maintenant étudier la démarche d'une personne malveillante qui cherche à mettre à profit les vulnérabilités d'un système Oracle pour accéder aux données qu'il contient, ou pour accéder au système d'exploitation des machines qui hébergent Oracle. Cela nous permettra d'en déduire l'ensemble des protections à mettre en place.

Dans un premier temps, nous allons nous pencher sur le cas d'un attaquant qui a un accès direct par le réseau aux serveurs Oracle, via le protocole natif OracleNet. Le schéma suivant synthétise la plupart des vulnérabilités d'un système Oracle dans ce cas de figure, qui sont décrites par la suite :

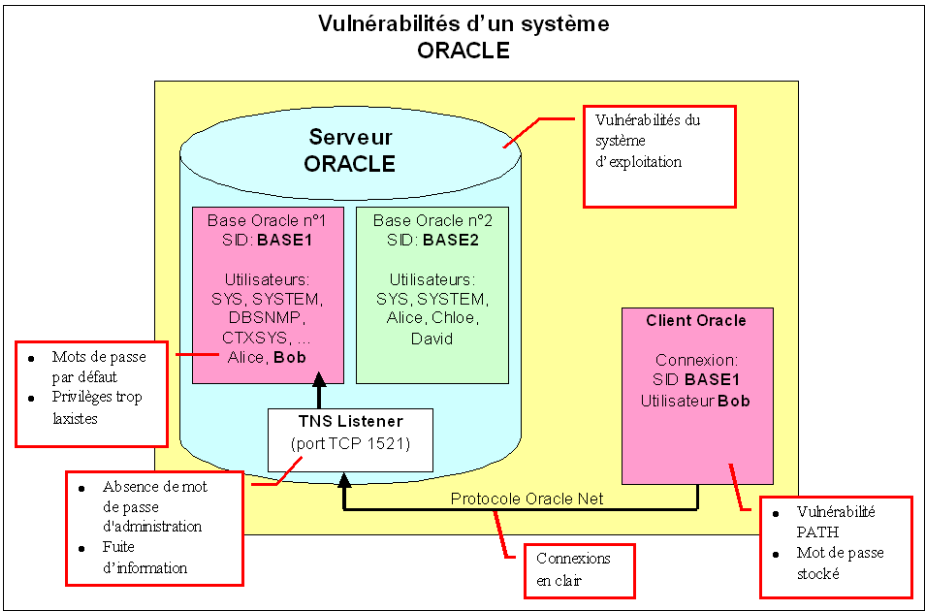


**Fig. 3.** Vulnérabilités d'un système Oracle

Voici les étapes possibles pour une intrusion sur un système Oracle via le réseau :

### 3.1 Prise d'empreinte réseau

Avant toute chose, un attaquant doit détecter et identifier correctement sa cible pour pouvoir adapter ses actions. Cela passe généralement par le réseau.

**Écoute réseau** Si l'attaquant a la possibilité d'écouter le trafic réseau, il peut identifier les connexions entre clients et serveurs, en détectant les flux OracleNet. Comme ces flux sont en clair par défaut, de nombreuses informations sont accessibles : adresses des clients et des serveurs, identifiants des bases de données. Certains outils comme dsniff permettent même de capturer les séquences d'authentification des utilisateurs.

**Scan de ports et de services** En utilisant des outils de scan de ports comme nmap (option -sV) ou amap, un attaquant peut détecter précisément la présence d'un serveur Oracle, et identifier le numéro de port TCP utilisé par le service TNS Listener (1521 par défaut).

Les tests de détection " OS fingerprinting " (par exemple nmap -O) permettent d'identifier le système d'exploitation qui héberge le serveur Oracle et sa version, informations très utiles pour rechercher les vulnérabilités exploitables.

Voici un exemple de résultat :

```
#nmap -sV -O -p 1-65535 192.168.0.1
Interesting ports on 192.168.0.1 :
PORT STATE SERVICE VERSION
22/tcp open ssh OpenSSH 3.4p1 (protocol 1.99)
80/tcp open http IBM HTTP Server 1.3.19.4 (Derived
from Apache 1.3.20 ; Unix)
94/tcp open objcall ?
111/tcp open rpcbind 2-4 (rpc #100000)
443/tcp open https ?
657/tcp open unknown
1521/tcp open oracle-tns Oracle Listener
...
Running : IBM AIX 5.X
OS details : IBM AIX 5.1-5.2
```

**Requêtes du TNS Listener** En ouvrant une connexion simple sur le service TNS Listener identifié grâce à un outil comme lsnrctl (fourni avec tout serveur Oracle) ou tnscmd.pl (cf. [TNS] et [TNSCMD]), l'attaquant peut obtenir de nombreuses informations supplémentaires grâce aux commandes « status » ou « services », sans aucun besoin d'authentification.

Les informations intéressantes fournies par le TNS Listener sont notamment :

- Le type et la version du système d'exploitation hôte
- La version d'Oracle
- Les SIDs des bases de données accessibles
- Les paramètres de sécurité du TNS Listener
- Parfois même les variables d'environnement du compte sous lequel tourne le TNS Listener, ce qui révèle le compte utilisateur sous lequel fonctionne Oracle, son répertoire home, etc...

Exemple :

```
#tnscmd.pl STATUS -h 192.168.0.1 -p 1521 --indent
sending (CONNECT_DATA=(COMMAND=status)) to 192.168.0.1 :1521
connect
writing 89 bytes
reading
DESCRIPTION=
VSNNUM=153093120
ALIAS=LISTENER
SECURITY=OFF
VERSION=TNSLSNR for Linux : Version 9.2.0.4.0 - Production
START_DATE=22-OCT-2004 16 :44 :17
SIDNUM=1
LOGFILE=/opt/oracle/9.2/network/log/listener.log
PRMFILE=/opt/oracle/9.2/network/admin/listener.ora
TRACING=off
```

```
UPTIME=188512
SNMP=OFF
PID=2120
SERVICE=
SERVICE_NAME=ORATEST
INSTANCE=
INSTANCE_NAME=ORATEST
NUM=1
INSTANCE_STATUS=UNKNOWN
NUMREL=1
SERVICE=
SERVICE_NAME=PLSExtProc
INSTANCE=
INSTANCE_NAME=PLSExtProc
NUM=1
INSTANCE_STATUS=UNKNOWN
NUMREL=1
#tnscmd.pl SERVICES -h 192.168.0.1 -p 1521 --indent
sending (CONNECT_DATA=(COMMAND=services)) to 192.168.0.1 :1521
[...]
INSTANCE=
INSTANCE_NAME=ORATEST
NUM=2
HANDLER=
[...]
```

```
ENVS='HOSTNAME=localhost, TERM=xterm, SHELL=/bin/bash,
TMPDIR=/persist/opt/9.2/tmp, USER=ora920,
ORACLE\_SID=ORATEST, ORACLE\_BASE=/persist/opt/oracle,
MAIL=/var/spool/mail/ora920, PATH=/persist/opt/oracle/9.2/bin:
/opt/bin: /bin: /usr/bin: /usr/local/bin: /usr/sbin: /usr/X11R6/bin,
SECURE\_LEVEL=2, PWD=/persist/opt/9.2, JAVA\_HOME=/usr/local/java,
HOME=/persist/opt/9.2, TMP=/persist/opt/9.2/tmp,
LOGNAME=ora920, DISPLAY=:0.0, ORACLE\_HOME=/persist/opt/oracle/9.2,
...'
```

### 3.2 Récupération de mots de passe Oracle

Pour accéder à une base de données Oracle via le protocole Oracle Net, il est nécessaire de s'authentifier en tant qu'utilisateur de cette base. En général cette authentification se fait par un login et un mot de passe d'un compte Oracle. Un attaquant dispose de plusieurs méthodes pour trouver un login / mot de passe valide :

**Mot de passe par défaut** Une des vulnérabilités principales d'une base de données Oracle est la présence de nombreux comptes utilisateurs par défaut avec des mots de passe « usine » bien connus ([ODPL] en liste plus de 600 !). En effet, toute création d'une base Oracle provoque l'ajout de comptes par défaut, par exemple SYS, SYSTEM, DBSNMP, OUTLN, ...

Chacun de ces comptes possède un mot de passe fixé par défaut, par exemple « manager » pour SYSTEM et « dbsnmp » pour DBSNMP. Cette liste de mots de passe par défaut est bien documentée et accessible publiquement sur Internet. La majorité des mots de passe est simplement identique au nom de compte.

Avec les versions récentes d'Oracle et suivant la méthode de création de la base (assistant DBCA ou création manuelle), l'administrateur peut être forcé par le système de choisir un mot de passe différent pour SYS et SYSTEM. Cependant cette fonctionnalité est en général limitée à ces 2 comptes, et elle n'était pas présente auparavant. Il n'est donc pas rare de retrouver ces comptes intacts sur d'anciennes bases de données toujours en production.

Il existe de très nombreux outils pour tester rapidement la liste des logins / mots de passe connus, comme par exemple OPWG (Oracle PassWord Guesser, cf. [OAT]) :

```
C :\OAT>opwg -s srv_ora1
Oracle Password Guesser v1.3.1 by patrik@cqure.net
-------------------------------------------------
INFO : Running pwcheck on SID BASE1
Successfully logged in with CTXSYS/CTXSYS
Successfully logged in with DBSNMP/DBSNMP
Successfully logged in with DEMO/DEMO
Successfully logged in with SYS/CHANGE_ON_INSTALL
Successfully logged in with SYSTEM/MANAGER
Skipping PLSExtProc ...
INFO : Running pwcheck on SID BASE2
Successfully logged in with CTXSYS/CTXSYS
Successfully logged in with DBSNMP/DBSNMP
Successfully logged in with OUTLN/OUTLN
```

**Écoute réseau** Une connexion OracleNet en clair peut révéler le login et le mot de passe Oracle de l'utilisateur, si l'attaquant emploie un outil comme dsniff.

**Mot de passe stocké sur un poste client** Il n'est pas rare de rencontrer des applications qui emploient Oracle comme un simple moyen de stockage de données, et qui sont conçues pour éviter à l'utilisateur la gestion d'un mot de passe supplémentaire. Dans ce cas, le mot de passe Oracle employé pour accéder aux données est souvent stocké " en dur " dans le code de l'application ou bien dans un fichier du disque, soit en clair soit sous forme obscurcie (chiffrement réversible). Un attaquant ayant accès au poste client peut alors trouver ce mot de passe, s'il dispose d'assez de temps et d'imagination.

**Mot de passe visible sur le serveur** Si l'attaquant dispose d'un accès au serveur en tant qu'utilisateur légitime du système d'exploitation, il pourrait obtenir l'accès à un mot de passe Oracle non protégé, par exemple :

- Mot de passe accessible dans un script ou un fichier de configuration du serveur mal protégé.
- Mot de passe visible dans une ligne de commande, grâce à la commande « ps ».

**Accès via une autre base Oracle** Si un attaquant a déjà accès à une base Oracle, il peut mettre à profit certaines vulnérabilités pour progresser dans le système en se connectant à d'autres bases liées :

- Intrusion via une base de données de développement mal protégée, contenant des liaisons vers d'autres bases (« database links » : mots de passe stockés en clair).
- Intrusion via la base « Repository » mal protégée d'un serveur Oracle Enterprise Manager, qui centralise les mots de passe d'administration de toutes les bases administrées.

### 3.3 Intrusion Oracle via le système d'exploitation

**Sur le serveur** Si l'attaquant est capable d'exploiter une vulnérabilité du système d'exploitation sur le serveur Oracle, il peut s'en servir pour obtenir l'accès aux bases de données :

- Modification de fichiers de configuration ou de scripts pour ajouter un compte d'administration Oracle.
- Ajout d'utilisateurs au groupe des administrateurs Oracle.

**Sur le poste client d'un administrateur Oracle** Comme le montre [ADMIN], le poste d'un administrateur Oracle peut être sujet à de nombreuses vulnérabilités, et un attaquant peut s'en servir pour obtenir l'accès aux bases de données. Il est par exemple possible de modifier le fichier « glogin.sql » pour injecter des commandes au lancement de SQL*Plus, ou bien « toad.ini » pour l'outil d'administration TOAD.

De plus, la plupart des outils d'administration Oracle ont la fâcheuse habitude de stocker certains mots de passe dans des fichiers sur le poste client. Même s'ils sont stockés sous forme chiffrée, il est parfois possible de recopier les fichiers de configuration sur une machine possédant le même outil pour se connecter directement à la base Oracle.

### 3.4 Intrusion système via Oracle

A l'inverse, certaines vulnérabilités d'Oracle permettent de compromettre le système d'exploitation, ou d'attaquer d'autres machines depuis Oracle.

**Exécution de code** Comme beaucoup de logiciels certaines versions d'Oracle sont vulnérables à des débordements de tampons qui permettent de déclencher du code exécutable à distance dans le contexte du TNS Listener.

**Vulnérabilité « Create Library »** L'outil OSE du package [OAT] montre qu'il est relativement simple d'exploiter une vulnérabilité d'Oracle pour obtenir un shell sur un serveur Oracle (sous Windows ou Solaris pour la version actuelle). Cet outil nécessite un accès Oracle avec un compte disposant du privilège « create library ».

Exemple :

```
c :\>ose -s 192.168.0.1 -t Windows -u system -p manager
OracleSysExec v1.3.1 by patrik@cqure.net
----------------------------------------
INFO : Local IP seems to be 192.168.0.10
SERVER :[2] Tftp Server thread started.
INFO : Uploading netcat to Oracle Server
INFO : Recieved read request from /192.168.0.1
INFO : Completed read request from /192.168.0.1
INFO : Sleeping for 2 seconds
INFO : Creating shell on port 31337 Oracle Server
INFO : Cleaning up !
INFO : The nc.exe will be left in %temp%
INFO : Stopping TFTP Server
c :\>nc 192.168.0.1 31337
Microsoft Windows [version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.
C :\WINDOWS\system32>
```

**Sécurité du TNS Listener** Le problème le plus répandu est l'absence de sécurisation par défaut du TNS Listener, même si tous les correctifs de sécurité ont été appliqués. Il est possible d'ouvrir une connexion et de transmettre des commandes d'administration au service TNS Listener sans authentification particulière, et sans accéder à une base. Voici les commandes d'administration du TNS Listener les plus importantes concernant la sécurité (une liste plus complète est fournie dans le document [TNS]) :

**status :** Fournit des informations, notamment la liste des SIDs des bases accessibles, la présence d'un mot de passe de protection, l'activation de SNMP, et la version d'Oracle.

**services :** Fournit des informations plus détaillées que status, en particulier l'environnement de l'utilisateur sous lequel tourne le Listener : compte utilisateur du TNS Listener, répertoire « home », etc.

**stop :** stoppe le TNS Listener. Cela peut aboutir à un déni de service, en bloquant l'accès réseau aux bases.

**log_file :** Modifie l'emplacement du fichier de log du TNS Listener.

A part « status », l'accès à ces commandes peut être protégé par un mot de passe, cependant par défaut ce mot de passe n'est pas activé.

La vulnérabilité la plus grave est liée à la commande « log_file », qui permet à un attaquant d'ajouter des lignes de texte quelconques dans tout fichier du système hôte, pourvu qu'il soit accessible au compte utilisateur sous lequel tourne le TNS Listener.

Si le système d'exploitation du serveur est un Unix, il est par exemple possible de modifier le fichier « .rhosts » de ce compte pour accéder au système sans authentification, si les services rsh ou rlogin sont activés. Ce type d'attaque est décrit en détail dans la documentation de l'outil tnscmd.pl (cf. [TNSCMD]).

Sur le même principe, cette vulnérabilité permet entre autres d'obtenir un accès SSH par injection de clé publique dans le fichier « .ssh/authorized_keys » (si le sous-répertoire .ssh existe déjà), d'injecter des commandes SQL dans le script « glogin.sql » de SQL*Plus, ou de piéger un serveur Windows grâce à un fichier BAT judicieusement placé.

Cette vulnérabilité du TNS Listener permet donc une intrusion au niveau du système d'exploitation hôte, sans passer par une authentification Oracle.

**Modules Oracle** Une fois obtenue une connexion à une base de données, certains modules sont accessibles par programmation et permettent de scanner ou d'attaquer d'autres machines du réseau, qui ne seraient pas forcément visibles autrement.

Par exemple le module « utl_tcp » rend possible la création d'un scanner de ports TCP depuis une base de données Oracle. Si le serveur Oracle est accédé par un attaquant à travers un pare-feu, ce module permet de scanner les autres machines normalement inaccessibles, derrière le pare-feu.

**Piégeage d'un poste client Oracle sous Windows** Le client Oracle pour Windows peut aussi poser des problèmes de sécurité pour le système d'information. Par exemple, il n'est pas rare de rencontrer des postes où le client Oracle a modifié la variable PATH du système lors de son installation pour placer le répertoire "bin" d'Oracle en tête, devant %systemroot%\system32, comme ceci :

```
PATH=C :\oracle\ora90\bin;C :\WINNT\system32;C :\WINNT;...
```

On peut généralement constater que ce répertoire « bin » est accessible en écriture à tout le monde. Il en résulte que n'importe quel utilisateur peut facilement piéger la machine en introduisant des exécutables possédant les mêmes noms que ceux situés dans « system32 » et utilisés par l'administrateur : cmd.exe, net.exe, regedit.exe,

### 3.5 Récupération de données

Si une base de données contient des informations sensibles, il est nécessaire de prendre des précautions pour les protéger, à la fois sur le réseau et dans la base elle-même. En effet ces informations pourraient être récupérées par un attaquant par divers moyens :

- Ecoute des connexions en clair, si le protocole Oracle Net n'est pas chiffré.
- Récupération d'une sauvegarde de base de données stockée dans un lieu mal protégé.
- Exploitation de privilèges ou rôles trop étendus donnant l'accès aux données : privilèges « any », rôle DBA, privilèges du groupe PUBLIC, ...

### 3.6 Maintien d'un accès dans un système Oracle (rootkit)

Une fois qu'un attaquant est parvenu à obtenir un accès dans une base de données Oracle, il peut avoir besoin de s'y maintenir, et de camoufler sa présence. Il peut alors se créer une porte dérobée par divers moyens (procédure stockée, trigger, ), ou même utiliser des techniques de camouflage évoluées qui s'apparentent aux rootkits qui existent pour Unix ou Windows. Le document [ROOTKIT] montre qu'il est possible de maintenir une session utilisateur dans une base Oracle et de lancer des procédures de façon très furtive, tout en fournissant une méthode pour détecter ce type d'activité.

### 3.7 Nouvelles vulnérabilités en 2004 et 2005

En 2004, David Litchfield a révélé l'existence d'une cinquantaine de nouvelles vulnérabilités découvertes dans diverses versions des produits Oracle, jusqu'à la toute dernière 10g, sans préciser la nature exacte de ces failles. Depuis, Oracle a commencé à fournir des correctifs pour certaines des ces vulnérabilités, et sous la pression des utilisateurs et des médias certains détails ont été révélés fin décembre 2004 puis courant avril 2005 (cf. [VULN1,VULN2]).

Etant donné le nombre et la criticité des vulnérabilités découvertes, il est plus que jamais important de suivre et d'appliquer les correctifs au fur et à mesure de leur parution (cf. [CPUSA,OSA]).

## 4 Scénarios pour une intrusion via une application web

Le cas d'une intrusion par une application web est différent, car les vulnérabilités mises en oeuvre ne sont pas les mêmes que pour un accès direct par le réseau via le protocole OracleNet.

Comme pour les intrusions par le réseau, il est possible de dresser un schéma des intrusions possibles pour une application web 3-tiers, par exemple à base d'Oracle HTTP Server ou Oracle Application Server.

Oracle HTTP Server est un serveur web Apache adapté pour créer des applications web liées à un serveur Oracle (via des interfaces PL/SQL, XSQL, JSP ou SOAP). Il fait notamment partie de la distribution standard du serveur Oracle 8i et de Oracle 9iAS.

Oracle Application Server est une distribution spécifique contenant des fonctionnalités supplémentaires adaptées aux applications web, incluant Oracle HTTP Server.

Alors qu'un serveur Oracle est souvent protégé des accès extérieurs, il est plus fréquent de rencontrer un serveur Oracle HTTP en première ligne (par exemple dans une DMZ), servant d'interface entre les utilisateurs et les bases de données.

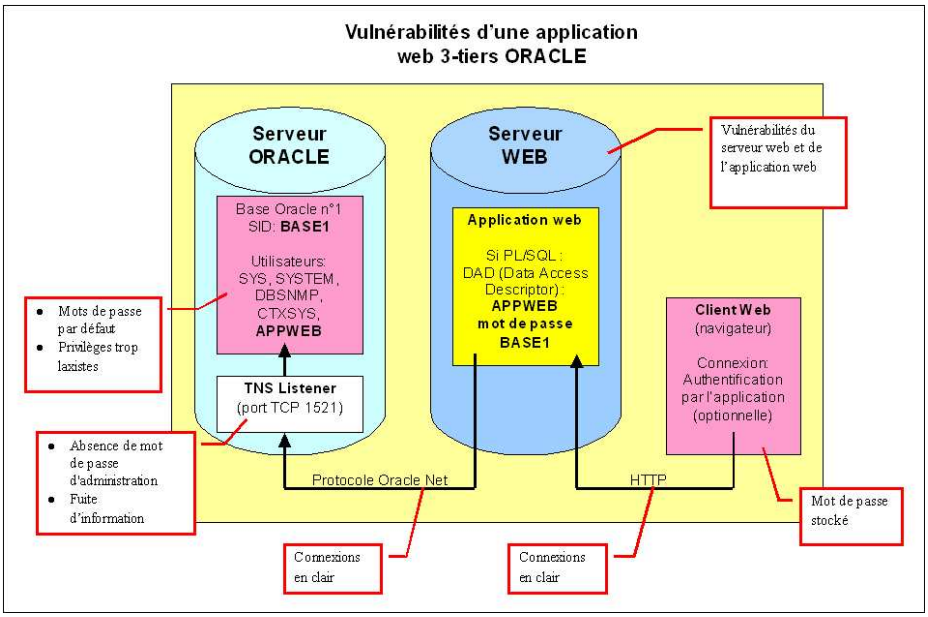


**Fig. 4.** Vulnérabilités d'une application web 3-tiers Oracle

### 4.1 Identification d'une application web basée sur Oracle

Si l'attaquant a accès à une application web, il peut être capable d'identifier qu'elle s'appuie sur un serveur Oracle, par exemple en étudiant les URLs employées ou les réponses à certaines requêtes. Cela est possible à la fois pour les applications publiques, où l'attaquant peut naviguer librement, et pour les applications nécessitant une authentification, où il est limité à la page d'authentification. En effet cette page d'authentification suffit souvent pour trahir la présence d'Oracle.

Voici quelques exemples d'URLs typiques d'une application basée sur Oracle HTTP Server ou Oracle Application Server :

- PL/SQL : http ://serveur/**pls**/xxx/
- XSQL : http ://serveur/**xsql**/xxx/
- SOAP : http ://serveur/**soap**/xxx/

### 4.2 Récupération de mots de passe web

Une application web qui emploie HTTP en clair et demande une authentification de l'utilisateur est vulnérable à une écoute réseau : le mot de passe de chaque utilisateur est facilement capturé.

Si l'utilisateur choisit de stocker son mot de passe sur son poste client grâce aux fonctionnalités des navigateurs récents, l'attaquant peut y accéder grâce aux éventuelles vulnérabilités du poste client, et s'en servir pour se connecter à l'application web.

### 4.3 Récupération de données (écoute réseau)

Si toute l'application web est basée sur HTTP en clair, un attaquant capable d'écouter le réseau entre un client et le serveur web peut récupérer toutes les données qui transitent.

### 4.4 Intrusion Oracle via une application web

Certaines vulnérabilités des applications web permettent à un attaquant de s'introduire dans une base de données Oracle associée, ou de lui faire exécuter des requêtes SQL depuis son navigateur. Si ces attaques ne sont pas spécifiquement filtrées par une passerelle adaptée, elles peuvent fonctionner à travers un pare-feu, puisque tout se passe au niveau applicatif sur HTTP.

**Injection SQL** Le type d'attaque le plus connu est l'injection SQL : l'attaquant saisit des données dans un formulaire de l'application web, en utilisant une syntaxe particulière. Si l'application exploite directement ces données sans filtrage dans un code PL/SQL, il se peut que l'attaquant parvienne à faire exécuter un code SQL de son choix, directement dans le contexte de la base de données (cf. [APPSEC,ADVSQL]). Dans certains cas particuliers, cela peut même aboutir à l'exécution de commandes au niveau du système d'exploitation.

Remarque : l'injection SQL n'est pas forcément liée à une application web. Toute autre application qui génère dynamiquement une requête SQL à base de données saisies par un utilisateur est potentiellement vulnérable à ce type d'attaque.

Exemple simplifié : considérons une application web qui demande à l'utilisateur de saisir une donnée « CODE_POSTAL » dans un formulaire, pour afficher une liste de personnes avec ce code postal. La requête SQL serait alors :

```
Select * from personnes where code = 'CODE_POSTAL'
```

Si l'attaquant saisit la chaîne suivante à la place d'un code postal valide :

```
75000' UNION select password from DBA_USERS
where 'q'='q
```

Alors la requête SQL réellement exécutée sera la suivante :

```
Select * from personnes where code = '75000'
UNION select password from DBA_USERS where 'q'='q'
```

ce qui risque de lui fournir la liste des mots de passe (hachés) de la base Oracle !

**Vulnérabilités des serveurs web Oracle HTTP Server et Oracle Application Server** L'installation par défaut de d'Oracle HTTP Server souffre généralement de nombreuses vulnérabilités spécifiques (cf. [HPOAS]).

Pour certaines versions comme Oracle 9iR1 on peut citer entre autres la possibilité de lancer des requêtes SQL sur le serveur sans avoir besoin d'authentification, à cause de modules PL/SQL vulnérables. Ces requêtes s'exécutent alors dans le contexte du compte utilisateur de l'application web. En effet une application web PL/SQL est configurée grâce à un DAD (Data Access Descriptor), qui correspond à un compte et un mot de passe de la base de données Oracle associée. Ainsi toutes les requêtes de l'application web sont effectuées sur la base à l'aide de cet unique compte.

Par exemple, l'URL suivante provoque le renvoi de la liste des comptes Oracle de la base associée à l'application web (cette liste est visible en affichant le code source de la page HTML retournée) :

`http://serveur/pls/dad/owa_util.listprint?p_theQuery=select%20*%20from%20all_users&p_cname=&p_nsize=`

Un serveur Oracle HTTP vulnérable peut donc servir d'intermédiaire à un attaquant pour accéder à des données et à des procédures normalement interdites, même si le serveur Oracle associé est correctement protégé.

**Google hacking** Un jeu à la mode consiste à exploiter la puissance du moteur de recherche Google pour dénicher de nombreuses vulnérabilités accessibles sur des serveurs web connectés à Internet.

Les serveurs web basés sur Oracle n'échappent pas à la règle, comme le montrent [GOOGLE1,GOOGLE2,GOOGLE3].

## 5 Recommandations pour sécuriser Oracle

Etant données toutes les vulnérabilités présentées au cours de cet article, la sécurisation parfaite d'un système Oracle n'est pas une tâche simple... Il est cependant possible d'améliorer très nettement la sécurité d'Oracle en appliquant quelques recommandations. Certaines de ces recommandations relèvent du bon sens, mais la réalité montre qu'il n'est jamais inutile de les rappeler, surtout dans un domaine aussi complexe que les systèmes Oracle :

1. Installer et activer uniquement les fonctionnalités indispensables. Même si l'installation est plus complexe et plus longue, c'est un gain de temps et d'argent à moyen et long terme : tout module non installé n'aura pas besoin d'être mis à jour, n'aura pas besoin d'être sécurisé, et surtout ne pourra jamais être attaqué !
2. Pour un serveur web Oracle HTTP Server ou Oracle Application Server, supprimer toutes les pages d'exemple, et désactiver les modules inutiles.
3. Sécuriser le système d'exploitation des serveurs Oracle, ainsi que tous les postes clients employés par les administrateurs.

4. Appliquer régulièrement tous les correctifs de sécurité publiés par Oracle (cf. [CPUSA]). Bien sûr, la pertinence de chaque correctif doit être analysée, le correctif doit d'abord être installé et vérifié dans un environnement de test avant d'être appliqué en production.
5. A la création de toute nouvelle base Oracle, désactiver tous les comptes utilisateurs non indispensables.
6. Modifier les mots de passe de tous les comptes actifs.
7. Vérifier régulièrement ces mots de passe à l'aide d'outils de test comme [OAT,ODPAT].
8. Appliquer des contraintes sur les mots de passe : durée de vie, verrouillage, complexité, ...
9. Sécuriser le TNS Listener, en lui appliquant un mot de passe d'administration et en ajoutant le paramètre ADMIN_RESTRICTIONS (cf. [TNS]).
10. Vérifier régulièrement la sécurisation du TNS Listener à l'aide d'outils comme [LCHK,NESSUS].
11. Eviter de laisser un serveur Oracle accessible à tout le monde sur un réseau ouvert, surtout s'il s'agit d'Internet. Éventuellement protéger son accès par un pare-feu, ou configurer le TNS Listener pour n'autoriser qu'une liste d'adresses IP pour les clients (cf. [TNS]).
12. Surveiller régulièrement la sécurité et l'intégrité des bases Oracle sensibles, ainsi que tous les journaux d'évènements pouvant mettre en évidence des actions malveillantes.
13. Envisager le chiffrement des connexions réseau, qu'il s'agisse d'OracleNet avec SSL ou SSH, ou de HTTPS pour les applications web.
14. Pour une application web, vérifier que toutes les données saisies par les utilisateurs sont bien filtrées pour éviter les risques d'injection SQL.
15. Sur les postes clients Windows, vérifier les ACLs sur les répertoires du PATH.
16. Protéger correctement les sauvegardes des bases de données.

Cette liste est loin d'être exhaustive.

Pour aller plus loin, les documents [CHK9,TNS,CIS,SECOAS] ainsi que le site [FINN] fournissent de nombreuses informations supplémentaires pour la sécurisation des produits Oracle.

## 6 Conclusion

Cet article a montré un aperçu des vulnérabilités qu'un attaquant averti peut mettre en oeuvre pour compromettre une base de données Oracle ou bien le système d'exploitation du serveur qui l'héberge. Ces vulnérabilités ne doivent pas être prises à la légère, puisque de très nombreux systèmes utilisent Oracle pour y stocker des données sensibles.

La complexité du monde Oracle et le nombre impressionnant de fichiers et de paramètres disponibles ne doit pas empêcher un DBA ou un responsable sécurité

de se pencher sur la sécurisation des serveurs Oracle. L'application de quelques recommandations permet de se protéger contre la plupart des vulnérabilités principales.

## Références


[TNSCMD] Tnscmd.pl, http://www.jammed.com/%7Ejwa/hacks/security/tnscmd

[TNS] Oracle Database Listener Security Guide, Integrigy, http://www.integrigy.com/info/Integrigy_OracleDB_Listener_Security.pdf

[ODPL] Oracle default password list, Pete Finnigan, http://www.petefinnigan.com/default/default_password_list.htm

[OAT] Oracle Auditing Tools, http://www.cqure.net

[ODPAT] Oracle Default Password Auditing Tool, Marcel-Jan Krijgsman et Pete Finnigan, http://www.petefinnigan.com/default/default_password_checker.htm

[ADMIN] Hardening Oracle Admin PC, Alexander Kornbrust, http://www.red-database-security.com/wp/hardening_admin_pc_us.pdf

[ROOTKIT] Database Rootkits, Alexander Kornbrust, http://www.red-database-security.com/wp/db_rootkits_us.pdf

[VULN1] -Referenceshttp://www.petefinnigan.com/weblog/archives/00000183.htm

[VULN2] -Referenceshttp://www.ngssoftware.com/advisory.htm

[CPUSA] Oracle Critical Patch Updates and Security Alerts, Oracle Corporation, http://otn.oracle.com/deploy/security/alerts.htm

[OSA] Oracle security alerts, Pete Finnigan, http://otn.oracle.com/deploy/security/alerts.htm

[HPOAS] Hackproofing Oracle Application Server, David Litchfield, NGS Software, http://www.nextgenss.com/papers/hpoas.pdf

[APPSEC] Protecting Oracle Database White Paper, Aaron Newman, Application Security Inc., http://www.appsecinc.com/presentations/Protecting_Oracle_Databases_White_Paper.pdf

[ADVSQL] Advanced SQL Injection in Oracle databases, Esteban Martínez Fayó, http://security-papers.globint.com.ar/oracle_security/sql_injection_in_oracle.php

[GOOGLE1] Google Hacking, Johnny Long, http://www.blackhat.com/presentations/bh-europe-05/BH_EU_05-Long.pdf

[GOOGLE2] Search Engines Used to Attack Databases, Aaron Newman, http://www.appsecinc.com/presentations/Search_Engine_Attack_Database.pdf

[GOOGLE3] Google Hacking of Oracle Technologies, Alexander Kornbrust, http://www.red-database-security.com/wp/google_oracle_hacking_us.pdf

[LCHK] Lsnrcheck, Integrigy, http://www.integrigy.com

[NESSUS] Nessus, http://www.nessus.org

[CHK9] Oracle 9iR2 Security check-list, Oracle Corporation, http://otn.oracle.com/deploy/security/oracle9i/pdf/9ir2_checklist.pdf

[CIS] Oracle Database Security Benchmark v1.1, Center for Internet Security, `http://www.cisecurity.org/bench_oracle.html`

[SECOAS] Securing Oracle9iAS 1.0.2.x, Stephen Comstock, Oracle Corporation, `http://www.oracle.com/technology/deploy/security/oracle9iAS/pdf/securingias.pdf`

[FINN] Oracle Security, Pete Finnigan, `http://www.petefinnigan.com/orasec.htm`