Intrusions

SSTIC05 – 01/06/2005

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

Oracle et la sécurité

- **Oracle:**
  - Un des systèmes de gestion de bases de données les plus répandus.
  - Au cœur de nos réseaux, contient souvent les informations les plus sensibles de l'entreprise.
  - Vendu comme « Unbreakable » depuis 2001.
- **Constat: les serveurs Oracle sont souvent très mal sécurisés (euphémisme).**
  - Très nombreuses **vulnérabilités** connues pour une installation par défaut.
    - (et simples à exploiter, en plus…)
  - Peu de **correctifs** appliqués en général.
    - Les « anciennes » failles sont toujours d'actualité.
  - Risque de compromission pour les données, et pour l'OS du serveur.

Oracle et la sécurité

- **Nombreuses (bonnes) raisons à cela:**
  - La priorité pour les bases de données sensibles est généralement la **disponibilité**.
    - Installer un correctif est toujours risqué.
    - « Si ça marche, n'y touche surtout pas ! »
  - L'administration Oracle est **très complexe**, un DBA peut difficilement tout maîtriser.
    - Très nombreux paramètres et concepts à connaître.
  - Oracle est souvent livré comme un **simple moyen de stockage** « boîte noire » dans un système.
  - Les **formations** Oracle abordent très peu les aspects sécurité. (surtout pas sous l'angle de l'attaquant)
  - **La sécurité est coûteuse** en temps.
    - Installation et test de correctifs.
    - Précautions nécessaires pour serveurs de production.
  - Cf. http://www.hsc.fr/ressources/presentations/sec-sgbd-ossir/index.html.fr

1) Quelques bases utiles sur le fonctionnement d'Oracle

Architectures Oracle

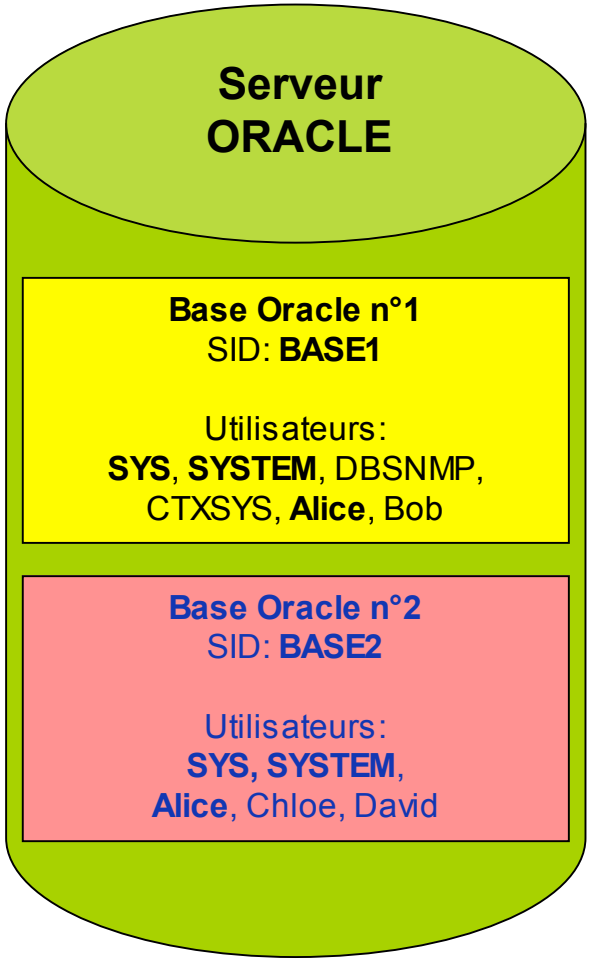
- **1) Connexion client/serveur native Oracle**
  - Protocole OracleNet sur TCP
- **2) Connexion HTTP(S) sur une application web basée sur Oracle**
  - Architecture 3-Tiers

Serveurs, bases et comptes

- **1 serveur Oracle peut contenir plusieurs bases de données**
- Chaque base est identifiée par un **SID**.
  - Exemples : « orcl », « annu », ...
- **1 base Oracle contient entre autres:**
  - des données dans des **tables**,
  - des procédures stockées,
  - des paramètres,
  - et des **comptes utilisateurs**.

Serveurs, bases et comptes

- **Important** : Chaque base Oracle contient **ses propres comptes utilisateurs**.
  - Les comptes peuvent être différents dans chaque base.
  - Un compte de même nom peut avoir des mots de passe différents dans chaque base, et correspondre à des personnes différentes.



Authentification des utilisateurs

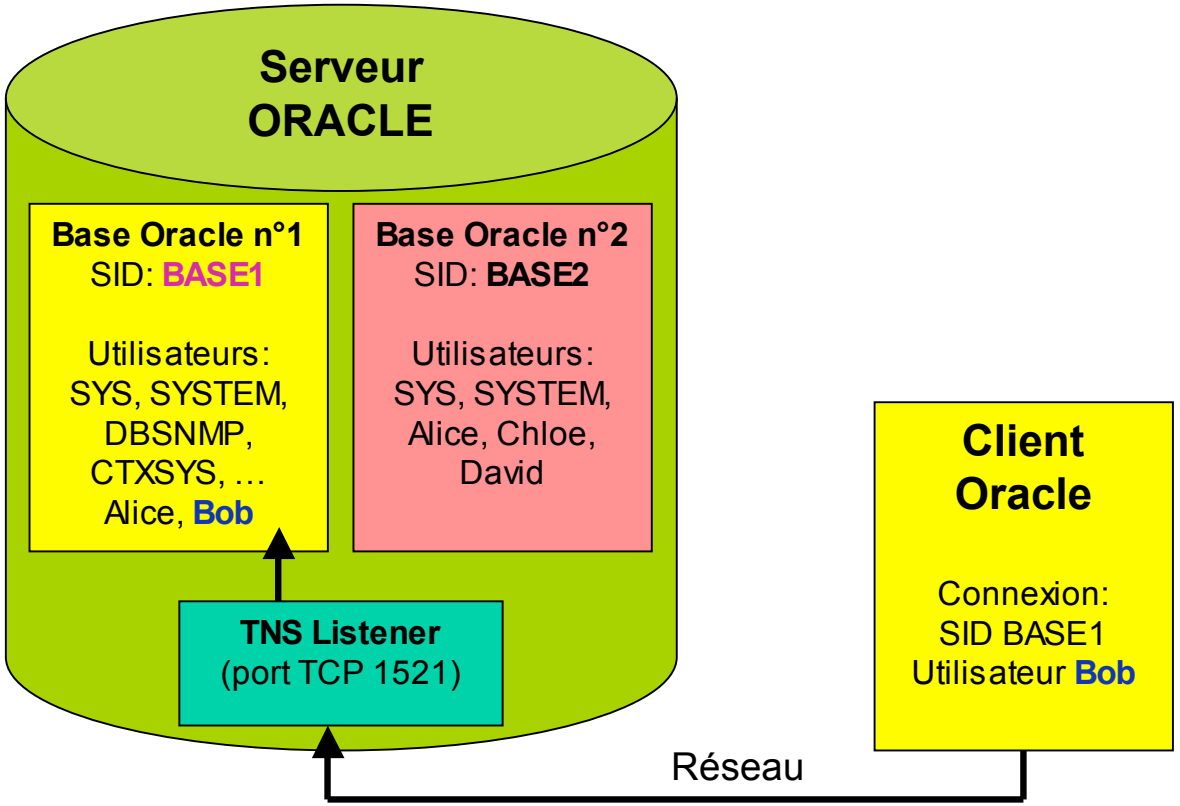
- **Authentification Oracle classique :**
  - **login / mot de passe**
- Authentification en mode « NTS » :
  - Oracle sur un serveur Windows intégré dans un domaine délègue l'authentification à Windows.
  - => pas de mot de passe Oracle à saisir.
- Autres authentifications possibles :
  - délégation au système Unix
  - Kerberos
  - Certificats avec OAS, …

Connexions réseau natives

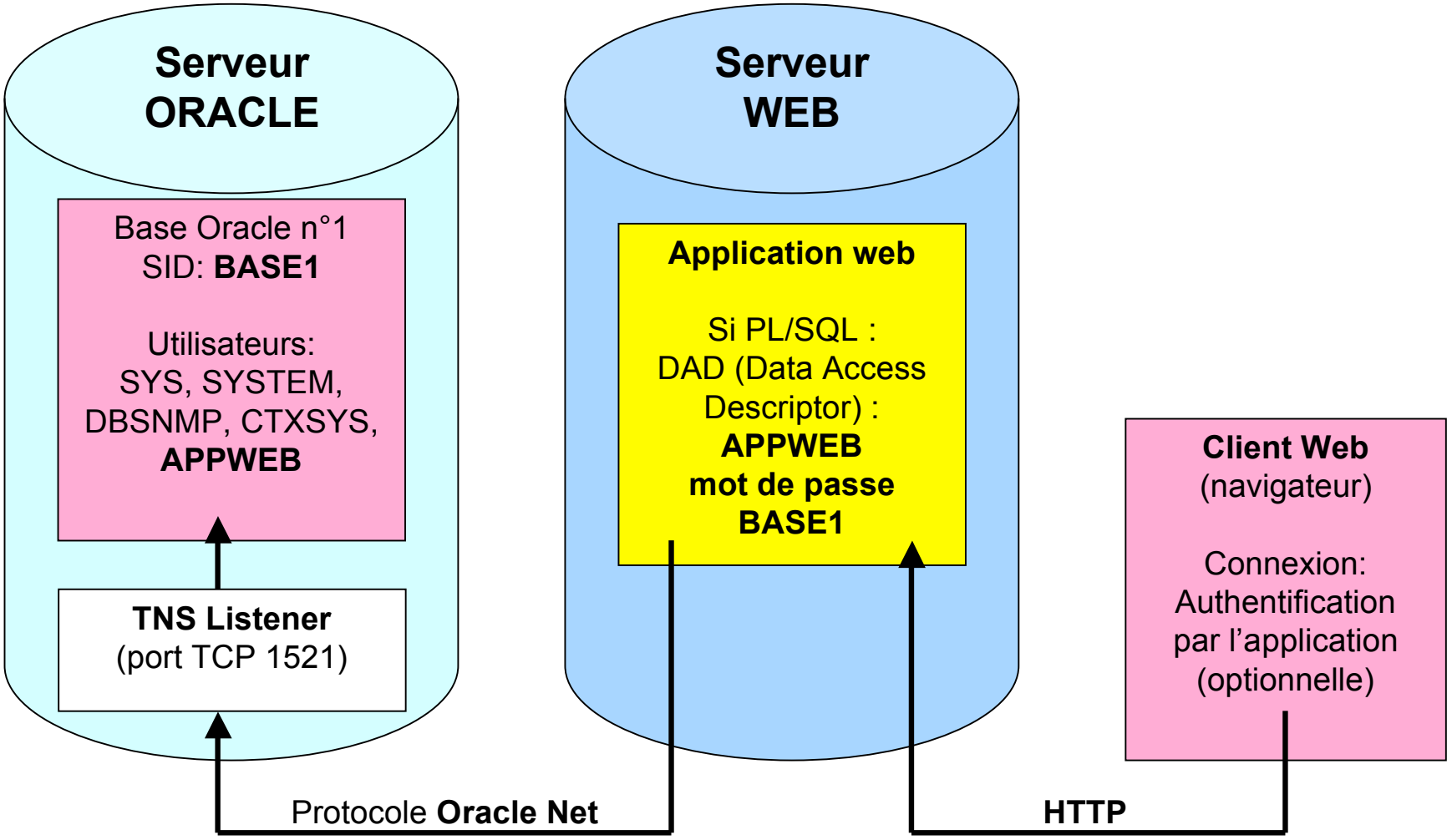
- **TNS Listener** : service qui accepte les connexions réseau des utilisateurs vers les bases Oracle du serveur.
- **Protocole** : OracleNet sur TCP/IP (ou IPX, …)
  - Le même protocole s'appelait SQL*Net pour Oracle 7
  - Net8 pour Oracle 8
  - OracleNet pour Oracle 9 ou 10
- **Port TCP par défaut** : **1521**
- **Autres ports possibles** : 1522 à 1540, 1630, 2483, 2484, … ou autre choisi par l'administrateur.
- Après connexion, négociation **dynamique** de port TCP => filtrage difficile si passage de pare-feu.
  - Solutions: restriction de la plage dynamique, ou installation d'Oracle Connection Manager.

Connexions réseau natives

- Exemple de connexion avec SQLplus:
  - sqlplus bob/azerty@base1



Application web sur Oracle (3-Tiers)



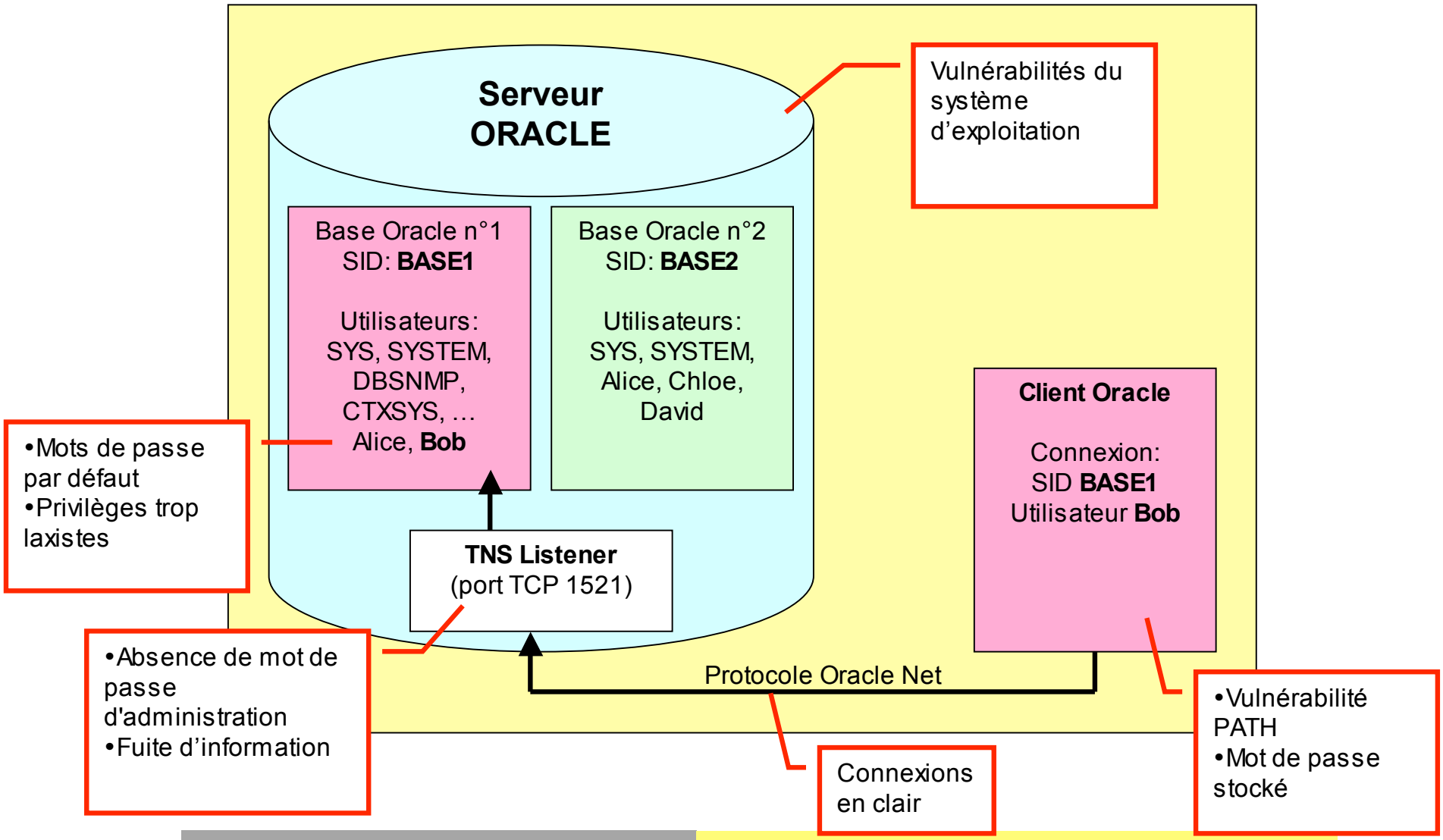
2) Vulnérabilités d'Oracle

Vulnérabilités Oracle

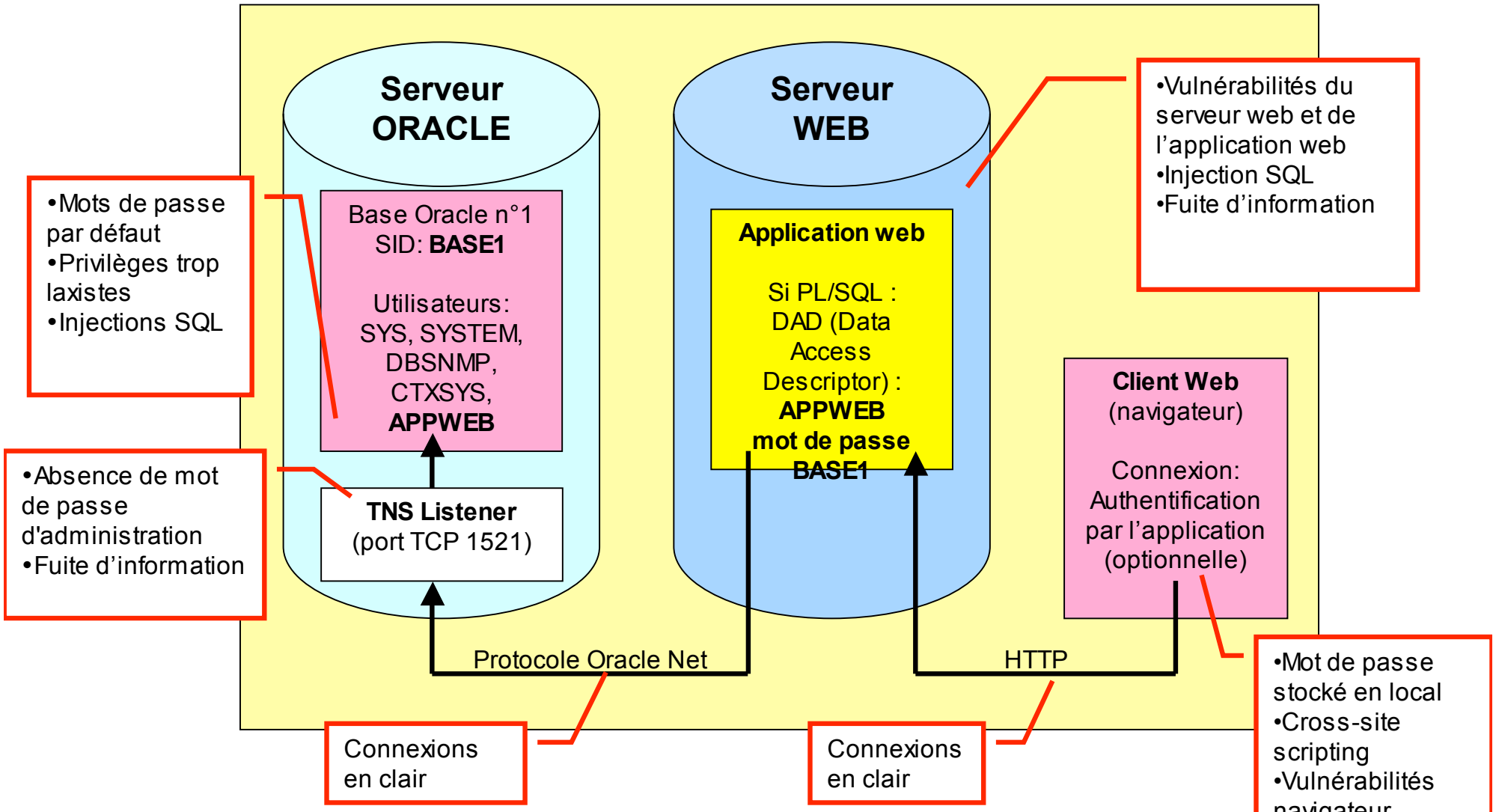
- Peuvent être classées en 2 catégories:
- Vulnérabilités de **conception**, **d'implémentation**
  - Peuvent être éliminées par des **correctifs** diffusés par l'éditeur.
- Vulnérabilités de **configuration**
  - Nécessitent une **action** de paramétrage volontaire et réfléchie de l'administrateur.

- Vulnérabilités différentes selon l'architecture:
  - Connexions natives Client/serveur
  - Application web
- Analyse plus complète dans les actes SSTIC.

Vulnérabilités (connexions natives)



Vulnérabilités (applications web)



3) Scénarios d'intrusion

(morceaux choisis)

Vulnérabilité n°1: Les mots de passe par défaut

- **Toute nouvelle base contient à sa création de nombreux comptes avec des mots de passe par défaut et connus. (listes sur Internet)**
  - **Comptes administrateurs:**
    - SYSTEM / MANAGER
    - SYS / CHANGE_ON_INSTALL
  - **Comptes privilégiés:**
    - CTXSYS / CTXSYS
    - MDSYS / MDSYS
    - TRACESVR / TRACE
  - **Et de très nombreux autres…**
- Amélioration dans Oracle 9i et surtout 10g.
- Dans la pratique, les DBAs oublient très souvent de modifier l'intégralité des mots de passe !

Scénario n°1:
Les mots de passe par défaut

- **Un attaquant trouve un login / mot de passe connu. (ou attaque par dictionnaire)**
- Si ce compte dispose de privilèges élevés:
  - L'attaquant peut **accéder à toutes les données** (privilège « select any table »), voire **les modifier**.
  - L'attaquant peut faire **exécuter du code** sur l'OS du serveur grâce au privilège « create library ».
- Si des procédures sont vulnérables à des injections SQL:
  - L'attaquant peut **élever ses privilèges**.
  - Il existe aujourd'hui ce type de failles non corrigées dans Oracle 10g.

Vulnérabilité n°2: TNS Listener

- **La connexion au TNS Listener ne nécessite pas d'authentification.**
  - L'authentification Oracle ne se fait qu'au niveau de la **base**.
- Le TNS Listener donne beaucoup d'infos.
  - Exemples: tnscmd.pl status/services.
- **Certaines commandes d'administration TNS permettent de modifier les paramètres du Listener via le réseau.**
  - Exemple: emplacement du fichier de log du TNS Listener.
- Par défaut, pas de mot de passe d'administration du Listener.
- **=> Possibilité d'écrire dans tout fichier texte du serveur en modifiant le fichier de log TNS.**

Scénario n°2a : intrusion Windows via un TNS Listener non sécurisé

- 1) Avec « tnscmd.pl status », l'attaquant identifie:
  - Que le Listener n'est pas protégé par mot de passe (security=off).
  - Que le serveur tourne sous Windows (version=…Windows…), donc il fonctionne sous le **compte système**.
- 2) Avec tnscmd.pl, il redirige le fichier log du TNS Listener vers un script qui sera exécuté à chaque ouverture de session.
  - Par exemple: C:\Documents and Settings\All Users\Menu Démarrer\Programmes\Démarrage\getadmin.bat
- 3) il ajoute des commandes à ce fichier, en générant des erreur TNS avec tnscmd.pl
  - net user sesame ouvre-3! /add
  - net localgroup administrateurs sesame /add
  - net start telnet
- 4) Dès qu'un administrateur se connecte, la porte s'ouvre.

Scénario n°2b: intrusion Unix RSH via un TNS Listener non sécurisé

- 1) Par un scan de ports, l'attaquant détecte sur le serveur Unix:
  - Le TNS Listener
  - Les services RSH et RLOGIN
- 2) Avec « tnscmd.pl services », il identifie:
  - Le compte Unix sous lequel tourne le TNS Listener
  - Que le Listener est vulnérable (security=off)
- 3) Avec tnscmd.pl, il redirige le fichier log du TNS Listener vers le fichier .rhosts de ce compte.
- 4) il ajoute « + + » à ce fichier, en générant une erreur
  - Ce qui autorise tout le monde à se connecter
- 5) il peut alors se connecter à Unix via rsh ou rlogin sans avoir besoin d'un mot de passe.

Scénario n°2c: Intrusion SSH via un TNS Listener non sécurisé

- Hypothèse: Serveur Unix correctement sécurisé, accès uniquement par SSH.

- Par défaut SSH effectue une authentification par login / mot de passe Unix.
- On peut aussi employer une **authentification par clé publique RSA** ou DSA.
- Si **clé publique dans $HOME/.ssh/authorized_keys**, authentification sans demander de mot de passe.

Scénario n°2c: Intrusion SSH via un TNS Listener non sécurisé

- **Génération d'une clé d'authentification SSHv2** RSA 768 bits:
  - Ssh-keygen –b 768 –t rsa –f id_rsa
- Récupération de la **clé publique** dans id_rsa.pub (1 ligne de texte)
- Ajout de cette clé dans .ssh/authorized_keys, pour le compte Oracle.
- La connexion SSH peut alors se faire sans mot de passe !

Intrusion via un TNS Listener non sécurisé

- **Conclusion: un TNS Listener non sécurisé est une vulnérabilité importante pour le système d'exploitation, pas seulement pour Oracle.**

Vulnérabilité n°3: Oracle HTTP Server

- **Serveur Web Apache customisé par Oracle, intermédiaire entre client HTTP et serveur Oracle**
- L'application web utilise **un seul compte Oracle** pour les requêtes: mot de passe dans un fichier de configuration « wdbsvr.app ».
  - DAD (Data Access Descriptor): nom utilisé dans les URLs pour désigner l'application web
  - Exemple d'URL pour une requête PL/SQL:
    - http://serveur:port/**pls**/**monDAD**/package.methode?paramètres
- **Nombreuses vulnérabilités** (cf. doc NGSS)
  - Exemple (Oracle 9iR1): certains modules PL/SQL (owa_util, …) permettent de **lancer des requêtes SQL** sur la base sans aucune authentification préalable.
  - Page d'administration par défaut: http://…/admin_

Vulnérabilité n°4: Client Oracle pour Windows

- **L'installation ajoute souvent un répertoire Oracle « bin » au début du PATH système, accessible en écriture à tout le monde**
  - Piégeage du système possible, par exemple en y plaçant un fichier « cmd.exe » qui crée un compte administrateur ou installe un cheval de Troie…

4) Sécurisation Oracle

Sécurisation Oracle: l'essentiel

- **Appliquer régulièrement les correctifs publiés.**
  - Oracle et système d'exploitation hôte.
- **Changer TOUS les mots de passe par défaut.**
- **Sécuriser le TNS Listener.**
  - Mot de passe d'administration, admin_restrictions, …
- **Désinstaller / désactiver toute fonction non utilisée.**
  - Plus un système est riche et complexe, plus il coûte cher en temps de sécurisation / mise à jour.
- Filtrer l'accès réseau aux serveurs.
- Chiffrer les liaisons client/serveur si besoin.
- Analyser la sécurité des applications web.
- Auditer la sécurité avec des outils adaptés.
- Pour aller plus loin:
  - recommandations Oracle, CIS et autres (http://www.petefinnigan.com)

Sécuriser le TNS Listener

- **Activer un mot de passe d'administration pour le TNS Listener**
  - Empêche la modification des paramètres TNS et l'arrêt du Listener, sauf si le mot de passe est fourni.
  - Restriction partielle des infos accessibles : requête « services » bloquée, mais « status » toujours possible.
  - Bémol: une attaque par force brute de ce mot de passe est possible, car pas de verrouillage après N échecs.
- **Ajouter le paramètre « ADMIN_RESTRICTIONS » dans listener.ora**
  - Les modifications de paramètres TNS « en ligne » deviennent impossibles, seul listener.ora est pris en compte.
- **Appliquer tous les correctifs…**
  - Autres failles connues du Listener: buffer overflows, create library, …
- Activer la journalisation, et surveiller le fichier de log.
  - Analyse des logs : « TNS-01169 » signale un échec de mot de passe.

Sécuriser le TNS Listener (suite)

- **Le TNS Listener ne doit surtout pas être accessible à tout le monde sur un réseau en environnement hostile, a fortiori sur Internet.**
  - => filtrage pour restreindre les accès au strict nécessaire.
  - Appliquer un filtrage sur l'adresse IP du client par le TNS Listener: tcp.validnode_checking
    - Applicable si quelques clients uniquement, car adresses IP individuelles, pas de plages.
  - Ou bien filtrer:
    - Par Oracle Connection Manager
    - Sur le serveur: iptables, filtrage IP Windows, …
    - Avec un pare-feu ou un routeur filtrant
- Voir le document d'Integrigy pour les détails sur toutes ces recommandations (et quelques autres).

Recommandations: Oracle HTTP Server

- **Nettoyer le serveur de tous les exemples, démos, …**
- **Appliquer des restrictions pour interdire les modules vulnérables:**
  - owa_util, utl_file, utl_tcp, …
- **Désactiver les interfaces inutilisées**
  - Suivant les besoins: PL/SQL, XSQL, JSP, SOAP, …
- Voir documents Oracle et NGSS « Hackproofing Oracle Application Server »

Méthode d'audit

- 1) Identifier les serveurs et bases
  - Voir doc du système et administrateurs
  - Scan de ports et de services (nmap –sV)
- 2) Sécurité du TNS Listener
  - Et connexions en clair ?
- 3) Recherche de mots de passe connus
- 4) Comptes, rôles et privilèges
- 5) Oracle HTTP Server (failles+HTTPS)
- 6) Sécurité des clients Windows

Références utiles

- La plupart des documents et outils concernant la sécurité Oracle sont référencés sur ce site:
  - http://www.petefinnigan.com
- Documents à lire en priorité:
  - Oracle: Check-list sécurité Oracle 9iR2
  - Integrigy: TNS Listener
  - NGSS: Oracle Application Server
  - CIS Oracle Benchmark (recommandations très complètes, avec outil de vérification)
- Et nombreuses références dans les actes

Boîte à outils pour l'audit Oracle

- Client Oracle complet (sqlplus, tnsping)
- Tnscmd.pl (avec Perl)
- Lsnrcheck.exe
- OAT: Oracle Auditing Tools (avec Java)
- Scripts d'audit SQL de Pete Finnigan
- MetaCoreTex (framework de test SGBD en Java)
- Nessus (quelques plugins Oracle)
- Oscanner
- Outils commerciaux

Conclusion

- La sécurité d'Oracle ne doit pas être sous-estimée parce qu'elle paraît compliquée.
  - Surtout pour des bases de données sensibles.
  - Ou si le serveur Oracle possède d'autres fonctions.
- Quelques actions de sécurisation simples permettent déjà de contrer 90% des attaques les plus courantes.
- Comme toutes les versions d'Oracle sont téléchargeables gratuitement, il est aujourd'hui facile de se former, de monter des architectures de test.
- Il ne faut surtout pas mettre en production un serveur Oracle non sécurisé.
- Mieux vaut Oracle 10g que 9i, et 9i que 8i…

select * from DBA_Questions;