Master 2 SeCReTS 2020-2021
Module Sécurité Système
Examen

Sécurité des systèmes Unix

Merci de bien lire les consignes :

— une seule feuille A4 manuscrite autorisée ;
— aucune communication ;
— aucun accès à un ordinateur, une station de travail, un téléphone portable, une calculatrice, un PDA ou tout autre dispositif électronique, connectable ou non ;
— sujet à remettre en fin d'examen ;
— n'oubliez pas d'**indiquer nom et prénom sur la copie**.

La précision des réponses est importante. Si vous donnez des réponses complètes en un minimum de lignes, vous pouvez obtenir ***un point supplémentaire*** *sur l'examen. En revanche, si vous donnez des éléments hors sujet dans vos réponses,* ***vous serez pénalisés****.*

Première partie

Sécurisation intrinsèque du système (20 pts)

Questions de cours (10pts)

1. *(1pt)* Quels sont les trois aspects majeurs sur lesquels intervenir lorsque l'on planifie la sécurisation d'un système d'information ? Donnez un exemple rapide pour chaque.
2. *(1pt)* Citez deux exemples de manières d'empêcher un attaquant qui aurait obtenu un accès physique au système d'information d'en prendre le contrôle total ? Expliquez en quoi chaque contremesure l'empêche / le ralentit dans sa démarche.
3. *(1pt)* Définissez « identification », « authentification » et « autorisation ». Donnez deux exemples pour chaque, que vous inscrirez éventuellement dans un scénario.
4. *(2pts)* Expliquez rigoureusement chaque ligne de la configuration PAM suivante et les conditions exactes d'acceptation de l'accès à l'utilisateur. Concluez.

```
account     sufficient    pam_succeed_if.so   user = root
account     requisite     pam_ethylo.so       alcohol_level < 0.5
account     required      pam_stack.so        service = system-account
    account     optional      pam_time.so         hour >= 8
    account     required      pam_unix.so
```

5. *(1pt)* Expliquez l'importance de générer des fichiers de journaux sur un système d'information. Donnez en exemple trois types d'opérations qu'un administrateur de sécurité peut avoir besoin d'effectuer sur ces fichiers / données.
6. *(1pt)* Citez les trois types de possibilités d'exploitation qu'un attaquant peut mettre en oeuvre à l'encontre d'un système d'information. Expliquez brièvement en quoi elles consistent.
7. *(1pt)* Expliquez ce qu'est un compte captif et donnez un exemple avec `/bin/halt`, un binaire qui arrête le système. Vous choisirez arbitrairement les autres champs.
8. *(2pts)* Pourquoi doit-on limiter les paquets installés, les services en fonctionnement, les ports en écoute sur un système bastion ? Dans la liste suivante, quels sont les paquets qui vous semblent inopportuns ? Pour chaque, expliquez brièvement pourquoi.

   — `acpid` : daemon de gestion des évènements de l'alimentation

— `base-files` : divers fichiers de base du système Debian
— `bash` : shell utilisateur
— `bluez` : drivers et outils pour la gestion du bluetooth
— `dpkg` : système de gestion des packages pour Debian
— `iptables` : outils d'administration du pare-feu netfilter
— `lvm2` : gestionnaire de volumes logiques Linux
— `mount` : outils pour monter les systèmes de fichiers
— `nmap` : simple scanner réseau
— `ntpd` : daemon de synchronisation de l'horloge via le réseau
— `sox` : outils divers de manipulation et tranformation du son
— `ssh` : metapackage client et server pour le shell sécurisé
— `sudo` : outil de délégation de privilèges aux utilisateurs
— `tzdata` : données des fuseaux horaires
— `udev` : daemon de management des périphériques
— `xorg` : serveur graphique sous linux

Applications directes (10pts)

1. *(5pts)* Voici le code d'une application C dont l'attaquant exploite une vulnérabilité.

```
int main(int argc, char** argv) {
  char buffer[32];
  strcpy(buffer, argv[1]);
  printf("buffer = %s\n", buffer);
  return 0;
}
```

Dessinez la pile précisément telle qu'elle est en fin de programme, en faisant apparaître :

— les arguments des fonctions appelées
— les sauvegardes de registres nécessaires
— les zones de variables locales (position et contenu)
— des flèches pour le sens des adresses et de croissance de la pile

De quel type de vulnérabilité s'agit-il ?
Mettez-la en évidence en expliquant son fonctionnement. Soyez exhaustifs.

2. *(5pts)* Voici le code d'une application C dont l'attaquant exploite une vulnérabilité.

```
int main(int argc, char** argv) {
  char path[128], data[128]; int fd;

  //   "./" est utilisé comme préfixe à l'argument argv[1]
  //   pour forcer la lecture dans le répertoire courant.
  snprintf(path, 128, "./%s", argv[1]);

  fd = open(path, 0 /* O_RDONLY */);
  read(fd, data, sizeof(data));
  close(fd);

  fd = open("/var/log/reader.log", 1024 /* O_APPEND */);
  write(fd, path, sizeof(path));
  close(fd);

  return 0;
}
```

De quel type de vulnérabilité s'agit-il ?
Mettez-la en évidence en expliquant son fonctionnement. Soyez exhaustifs.

```
$ ldd ./reader
        linux-vdso.so.1 (0x00007fff0a169000)
        libc.so.6 => /usr/lib/libc.so.6 (0x00007ff21bc09000)
        /lib64/ld-linux.so.2 => /usr/lib64/ld-linux.so.2 (0x00007ff21be08000)
```

Dessinez l'arborescence du système de fichiers que vous mettriez en place pour protéger le système de ce binaire. Veillez à bien mettre en évidence la technique utilisée.

Deuxième partie

Prévention et confinement des attaques (10 pts)

1. *(1pt)* Sur la prévention des attaques :
   — Expliquez brièvement quel principe est utilisé pour protéger un programme contre les *buffer overflow* dans la pile au moment de la compilation.
2. *(2pts)* Un attaquant possède les informations suivantes sur le système qu'il cible : la pile des programmes est non-exécutable (NX), mais la bibliothèque C standard (`libc`) est toujours chargée à la même adresse.
   — Comment l'attaquant peut exploiter un *buffer overflow* dans la pile d'un programme vulnérable avec ces informations ?
   — Quelle fonctionnalité du noyau l'administrateur du système ciblé devrait activer pour éviter qu'une attaque réussisse ?
3. *(3.5pts)* Conteneurs et virtualisation
   — *(1pt)* Expliquez la différence fondamentale entre les mécanismes de virtualisation (comme VirtualBox) et les mécanismes de conteneurs (comme LXC ou Docker).
   — *(0.5pt)* Quels sont les noms des 2 mécanismes du noyau qui permettent de créer des conteneurs sous Linux ?
   — *(1pt)* Donnez 2 cas d'usage de la virtualisation sous Linux.
   — *(1pt)* Décrivez la démarche permettant de restreindre le nombre de CPU utilisables par un groupe de processus.
4. *(3.5pts)* Mandatory Access Control
   — *(1pt)* On s'intéresse à l'idée de coupler le mécanisme de `chroot` à un mécanisme de *Mandatory Access Control* (MAC) comme SELinux.
      — Expliquez en quoi SELinux est complémentaire du mécanisme `chroot`.
      — Dans ce contexte, décrivez brièvement un schéma d'attaque qui serait bloqué par SELinux.
   — Observez ce listing qui représente un profil AppArmor :

```
# Declare an apparmor variable to help with overrides
@{MOZ_LIBDIR}=/usr/lib/firefox

#include <tunables/global>

/usr/lib/firefox/firefox {
  #include <abstractions/audio>
  #include <abstractions/cups-client>
  #include <abstractions/dbus-strict>
  #include <abstractions/dbus-session-strict>
  #include <abstractions/dconf>
  #include <abstractions/gnome>
  #include <abstractions/ibus>
  #include <abstractions/nameservice>
  #include <abstractions/openssl>
  #include <abstractions/p11-kit>

  # for networking
  network inet stream,
  network inet6 stream,

  /etc/ r,
  /etc/mime.types r,
  /etc/mailcap r,
  /etc/firefox*/ r,
  /etc/firefox*/** r,

  deny @{MOZ_LIBDIR}/** w,
  deny /usr/lib/firefox-addons/** w,

  owner @{HOME}/ r,
  owner @{HOME}/Public/ r,
  owner @{HOME}/Public/* r,
  owner @{HOME}/Downloads/ r,
  owner @{HOME}/Downloads/* rw,
  owner @{HOME}/.{firefox,mozilla}/ rw,
  owner @{HOME}/.{firefox,mozilla}/** rw,
  owner @{HOME}/.{firefox,mozilla}/**/*.{db,parentlock,sqlite}* k,
  owner @{HOME}/.cache/mozilla/{,firefox/} rw,
  owner @{HOME}/.cache/mozilla/firefox/** rw,
```

```
41    owner @{HOME}/.cache/mozilla/firefox/**/*.sqlite k,
42
43    # Addons
44    #include <abstractions/ubuntu-browsers.d/firefox>
45
46    # Site-specific additions and overrides. See local/README for details.
47    #include <local/usr.bin.firefox>
48  }
```

— *(0.5pt)* Quelle est l'application concernée par ce profil AppArmor ?
— *(2pts)* Pour chaque bloc de règles non commentées, décrivez la fonctionnalité concernée de l'application (on ne demande pas de détailler les règles une par une).

Troisième partie

Reverse engineering (6pts)

1. *(1pt)* Qu'est-ce que le flot d'exécution d'un programme ? (2 lignes)
2. *(2pts)* Décrivez les démarches d'analyse statique et d'analyse dynamique d'un programme. (5 lignes)
3. *(1pt)* Citez deux méthodes pour transmettre des arguments à une fonction en assembleur. (2 lignes)
4. *(2pts)* Qu'est-ce qu'une *race condition* ? (3 lignes)

Culture générale (*bonus*)

Citez un évènement public en rapport avec la sécurité informatique paru ces derniers jours (conférence, publication, parution d'outil, d'exploit, de vulnérabilité, une attaque ciblée, etc.). Donnez quelques détails sur ce sujet en particulier. Si votre réponse est très satisfaisante, elle peut donner lieu à des points hors barème.

Fin de l'examen.
