Sécurité Système

– Examen (1) –

1 Audit de code noyau (5 points)

Le code suivant fait partie d'un module noyau qui introduit un nouveau fichier `/dev/tostring`. On peut lire le code d'une structure interne ainsi que les fonctions de lecture et d'écriture dans `/dev/tostring`. Trouvez au moins deux bugs potentiels dans ce code, expliquez comment exploiter ce module (si c'est possible) et comment le corriger.

```
1  typedef struct
2  {
3    unsigned long long int stack[64];
4    unsigned long long int pointer;
5  } tostring_t;
6
7  static ssize_t
8  tostring_read(struct file *f, char __user *buf, size_t len, loff_t *off)
9  {
10   if (!tostring.pointer)
11     return 0;
12
13   return snprintf(buf,len,"%lld\n", tostring.stack[--tostring.pointer]);
14 }
15
16 static ssize_t
17 tostring_write(struct file *f, char __user *buf, size_t len,loff_t *off)
18 {
19   char *bufk = kmalloc(len, GFP_DMA);
20   if (copy_from_user(bufk, buf, len))
21     return -EFAULT;
22
23   tostring.stack[tostring.pointer++] = *((long long int *) bufk);
24   kfree(bufk);
25
26   return len;
27 }
```

2 Developing MacOS X Kernel Rootkits (15 points)

Lisez l'article "*Developing Mac OS X Kernel Rootkits*" par wowie (Phrack #66 2009). Puis rédigez des réponses aux questions suivantes.

Questions

1. Rappelez rapidement les buts et les principales fonctionnalités d'un rootkit (au moins les plus couramment observées). Et détaillez les différences entre un rootkit logiciel et un rootkit noyau.
2. Expliquez ce que fait l'appel système `SYS_getdirentries` sous MacOS X et donnez le nom de l'appel système équivalent sous Linux.
3. Les appels systèmes du noyau MacOS X sont-ils au niveau du micro-noyau XNU ou bien de la couche BSD ? Expliquez votre réponse.
4. Dans la section 4.1, l'auteur propose une fonction `*_start()` qui modifie l'appel système `getuid()`. Mais, il ne donne pas la fonction `*_stop()`. Écrivez les deux fonctions pour que l'on puisse mettre en place le hook puis le désinstaller à la sortie du plugin noyau.
5. Expliquer comment filtrer la liste des processus sous MacOS X et les principales difficultés qui s'y rapportent. Rappeler comment cela est fait sous Linux. Et, finalement, dire, à votre avis, lequel des deux mécanismes est le plus simple à détourner (donnez quelques arguments).
6. Expliquez comment retirer une extension noyau de la liste des extensions sous MacOS X. Proposez une manière qui permettrais de retirer et de réinsérer l'extension noyau dans la liste à volonté (par exemple en le reliant à un appel système autre). Inutile de proposer un code complet, détaillez seulement les grands principes.
7. Suggérez au moins trois manières de contrôler les fonctionalités d'un rootkit depuis le user-space.
8. Expliquez la technique du "*runtime kernel patching*" qui sert à insérer un rootkit dans le système.
9. Expliquer les deux techniques décrites à la section 5 utilisées pour localiser et patcher la table des appels systèmes ou des fonctions particulières de l'API noyau.
10. Détaillez les techniques de détection de rootkits décrites dans l'article et ajoutez-en certaines, si vous pensez qu'il en manque.

=> phrack.org/issues/66/16.html#article