Sécurité matérielle des systèmes et des données
Modèle de sécurité des systèmes d'exploitation

Martin Peres

Doctorant de Francine Krief au LaBRI

October 11, 2013

Sommaire

Le CPU

- Fournit un système MLS à 4 niveaux appelés rings (0 = plus privilégié, 3 = moins privilégié)
- Le niveau (appelé CPL) ne peut que descendre ...
- à moins qu'une interruption soit levée

La mémoire

- Segmentation
  - Distinction entre code et données
  - Contrôle d'accès basé sur le CPL
  - code: rx; data: rw
- Pagination :
  - Espace d'adressage par processus
  - Séparation kernel/userspace, writable? NX?

Rôle d'un système d'exploitation

- Abstraire (dans une certaine limite) le hardware
- Proposer une interface unique permettant à des applications de s'exécuter (exemple: POSIX)
- Permet à l'utilisateur d'exploiter sa machine

Un OS peut être

- multi-utilisateur
- multi-tâche : Voir l'ordonnanceur, thread et processus
- temps réel

Thread vs Process

- Process :
  - Espace d'adressage virtuel et un tas
  - un thread associé
- Thread : Fil d'exécution
  - Utilise l'espace d'adressage virtuel du processus père
  - Partage le même tas que le processus père
  - A sa propre pile

Ordonnanceur

- Partage le temps processeur entre plusieurs applications
- Programme un timer hw pour récupérer des IRQs périodiquement
- Effectue un context-switch lors d'un réveil
- Plusieurs politiques d'ordonnancement (Round-robin, Deadline, CFS)

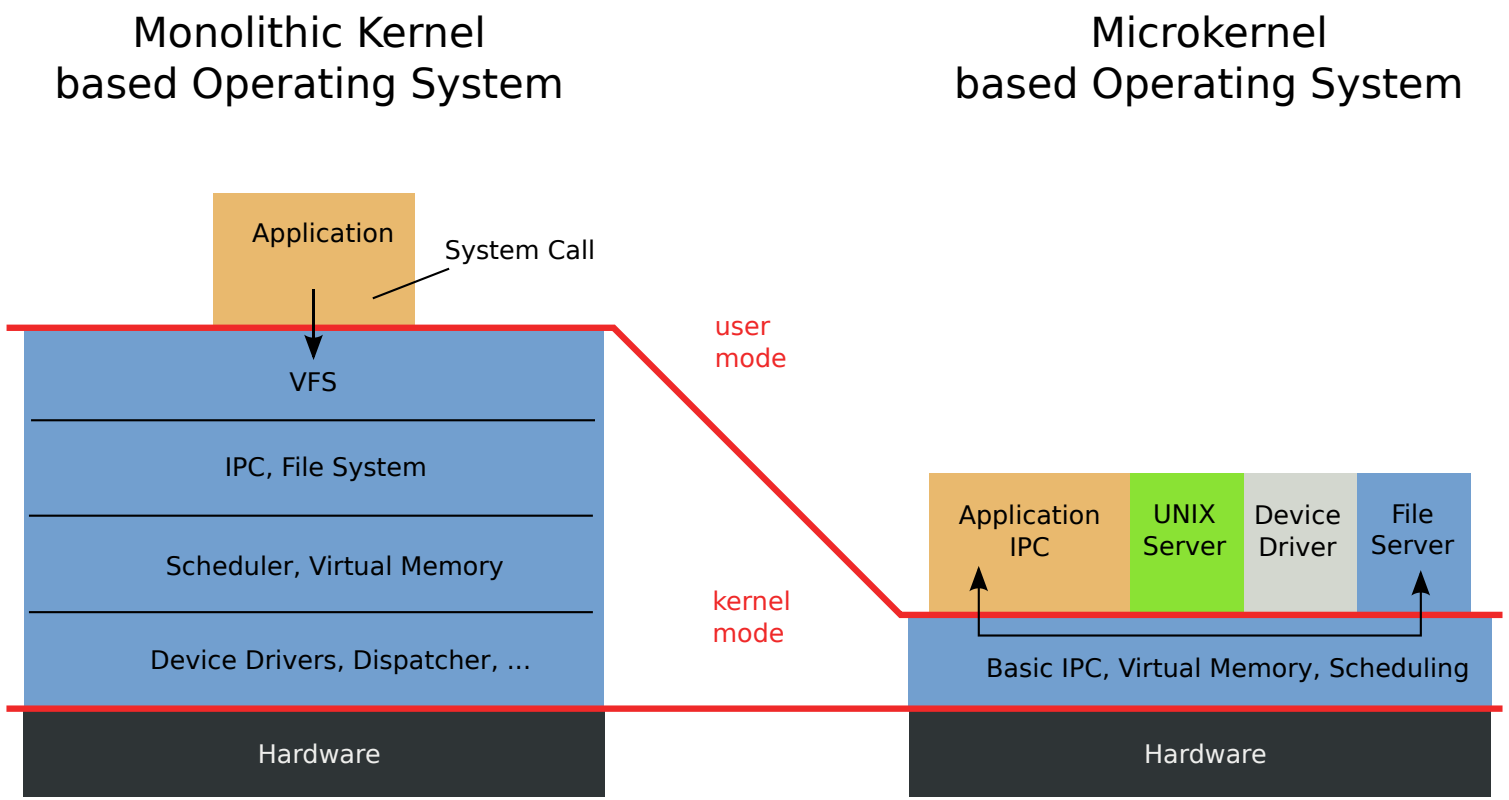
Communication inter-processus (IPC)

- Regroupe les outils d'échange de messages et de synchronisation
- Mémoire partagée (SHM)
- Tubes (pipe)

Synchronisation

- Verrou/mutex: Permet de protéger de la concurrence une zone critique
- Sémaphore: Permet de limiter l'accès à des ressources (ex: 10 lecteurs maximums)
- Les signaux

Sommaire



Architecture OS

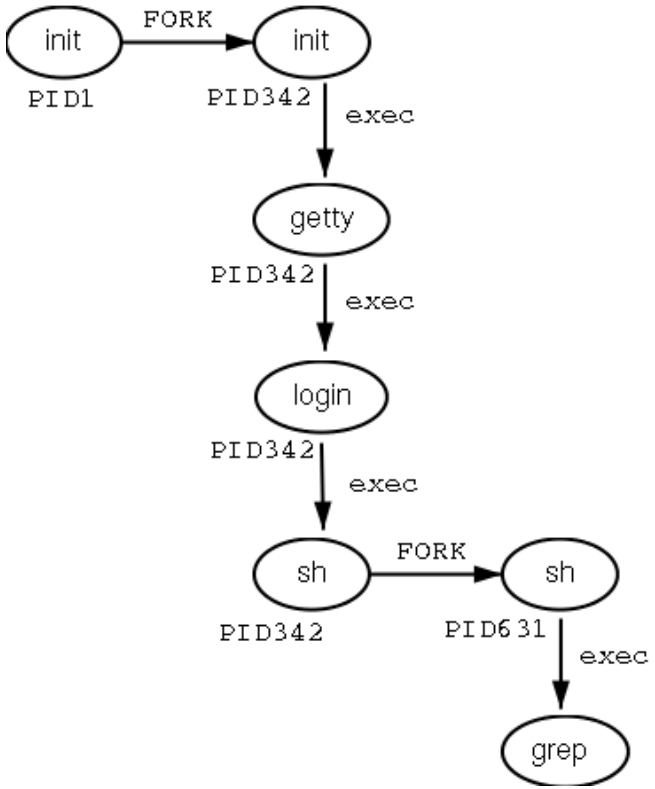
- Monolithique modulaire: Linux, Mac OS X, Windows
- Microkernel: GNU Hurd, QNX (RTOS)

Kernel Monolithique

- ring 0: FileSystem, IPC, Scheduler, drivers, IRQ dispatcher
- ring 3: Les applications
- Avantages: Développement plus rapide car couplage plus fort
- Inconvénients: Sécurité et sûreté de fonctionnement

Microkernel

- ring 0: MessagePassing, Scheduler, IRQ dispatcher
- ring 3: Les applications, les FileSystem et IPC étendues
- Avantages: Sécurité et sûreté de fonctionnement
- Inconvénients: Développement plus difficile car besoin de plus d'interfaces



Création d'un processus

- Initié par l'appel système fork()
- Généralement suivi d'un exec()

Création d'un processus

- Création d'une nouvelle VM
- Recopie de l'état du processus ayant forké
- reprise de l'exécution (avec un PID différent) en ring 3

Context-switching

- Géré par l'ordonnanceur
- Permet de donner l'illusion de simultanéité à l'utilisateur
- Se passe dans le kernel space

Sommaire

Implémentation sur Linux

Utilisation de 4 segments :

- user code: Base = 0x0, limit = 0xffffffff, DPL = 3
- user data: Base = 0x0, limit = 0xffffffff, DPL = 3
- kernel code: Base = 0x0, limit = 0xffffffff, DPL = 0
- kernel data: Base = 0x0, limit = 0xffffffff, DPL = 0

Raisons de la non-utilisation de la segmentation

- En partie redondant avec la pagination
- Gestion de la mémoire plus simple quand on partage les mêmes segments
- Tous les processeurs (notamment les RISC) ne gèrent pas bien la segmentation. Portabilité simplifiée.

Utilisation de la pagination

Linux utilise la pagination pour garantir :

- la protection contre les erreurs d'adressage
- dissocier les adresses virtuelles de la mémoire physique (facilite le swap)

Context-switch

Lors d'un context switch, Linux :

- vide le TLB
- change le pointeur CR3 pour pointer vers le bon "page directory"

Utilisation du bit NX

Rend la pile et le tas non exécutable à moins que :

- le binaire soit marqué dans les en-têtes ELF comme nécessitant que le tas ou la pile soit exécutable (PT_GNU_STACK, PT_GNU_HEAP)
- le paramètre kernel noexec=off (ou ne soit pas égal à on)

Utilisation de l'ASLR

Address-Space Layout Randomization (ASLR) rend aléatoire les adresses virtuelle de base :

- de la pile et le tas
- des bibliothèques chargées

Conclusion sur la gestion de la mémoire dans Linux

- Segmentation utilisée au minimum
- toute la sécurité repose principalement sur la pagination
- l'ASLR et le bit NX viennent rendre plus difficile l'exploitation de buffers overflow

Sommaire

Conseils généraux

- Empêcher les zones mémoires d'être accessibles en écriture/exécution
- Empêcher qu'un utilisateur puisse interférer avec le déroulement voulu par le programmeur

mprotect

- Contrôle les autorisations d'accès à une partie de la mémoire.
- Appel fait par le processus lui même. Réversibilité possible (sic!)

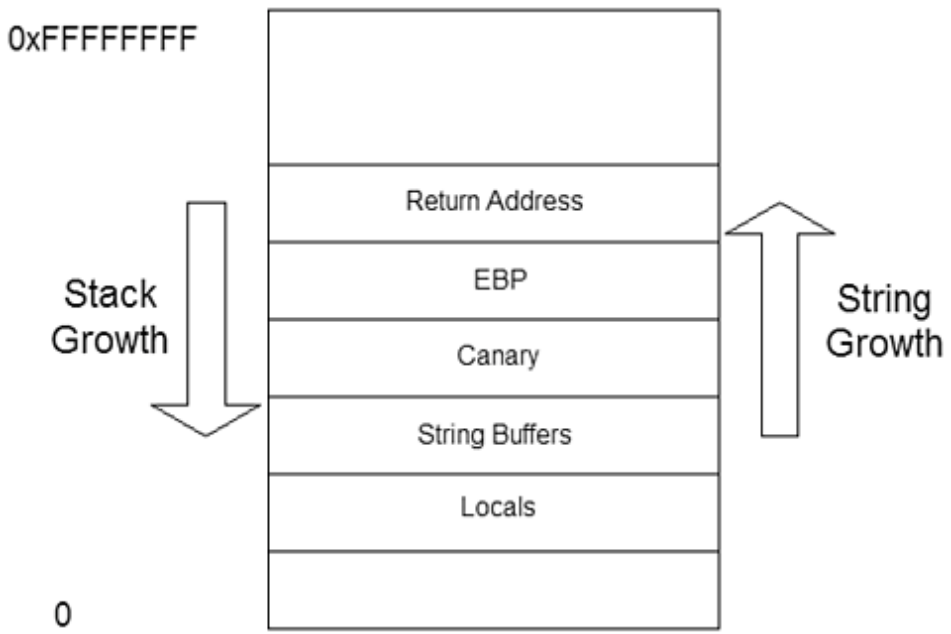
Utilisation

- int mprotect(const void *addr, size_t *len, int prot);
- prot est un bitfield:
  - PROT_NONE: Pas d'accès à la mémoire
  - PROT_READ: Ajoute l'accès en lecture
  - PROT_WRITE: Ajoute l'accès en écriture
  - PROT_EXEC: la zone contient du code exécutable

La sécurité à la compilation : Les Canaries

Part d'un constat simple, la plupart des attaques sont de type buffer overflow. Solution :

- Utilisation des prologues/epilogues d'appel de GCC pour les appels de fonction
- prologue: génère un nombre aléatoire (canary), stocké dans la pile avant les valeurs à protéger
- epilogue: vérifie que le canary n'a pas été modifié avant d'éxecuter la suite du code


0xFFFFFFFF
Return Address
EBP
Canary
String Buffers
Locals
0
Stack Growth
String Growth

Pax

- Suite de patch pour Linux (en cours de merge, voir la mailing list kernel-hardening)
- se concentre sur l'augmentation de la sécurité mémoire
- a proposé la première et toujours la meilleure implémentation de l'ASLR
- ne résout pas le problème des applications mal conçues
- Maintenu par Brad Spencer, développeur de GrSecurity

Pax - quelques options

- PAGEEXEC: Implémentation logicielle du bit NX
- SEGMEXEC: Autre implémentation logicielle du bit NX utilisant la segmentation (mais diminue l'espace d'adressage virtuel à 1.5Gio)
- mprotect: Empêche d'avoir l'écriture et l'exécution lors d'un appel à mprotect
- et bien plus pour 0€!

Sommaire

Conclusion

- Les systèmes d'exploitation utilisent certaines fonctions du hardware concernant la sécurité
- certaines améliorations/patchs existent pour tirer encore plus parti du hw
- Le hardware ne résout pas tout et l'OS non plus. Au contraire, il rajoute des failles.
- Conclusion habituelle: La sécurité se fait à tous les niveaux!