Darwin, il cuore Open Source di Mac OS X

Viaggio nel kernel made in Apple

Francesco Del Prato

Linux Day - 25.X.2008

Indice dei contenuti

Breve panoramica del Kernel

Microkernel vs Kernel monolitico

Address space: 3/1 e 4/4 (ma non solo)

I punti di forza

Un occhio alla licenza

Indice

Breve panoramica del Kernel

Microkernel vs Kernel monolitico

Address space: 3/1 e 4/4 (ma non solo)

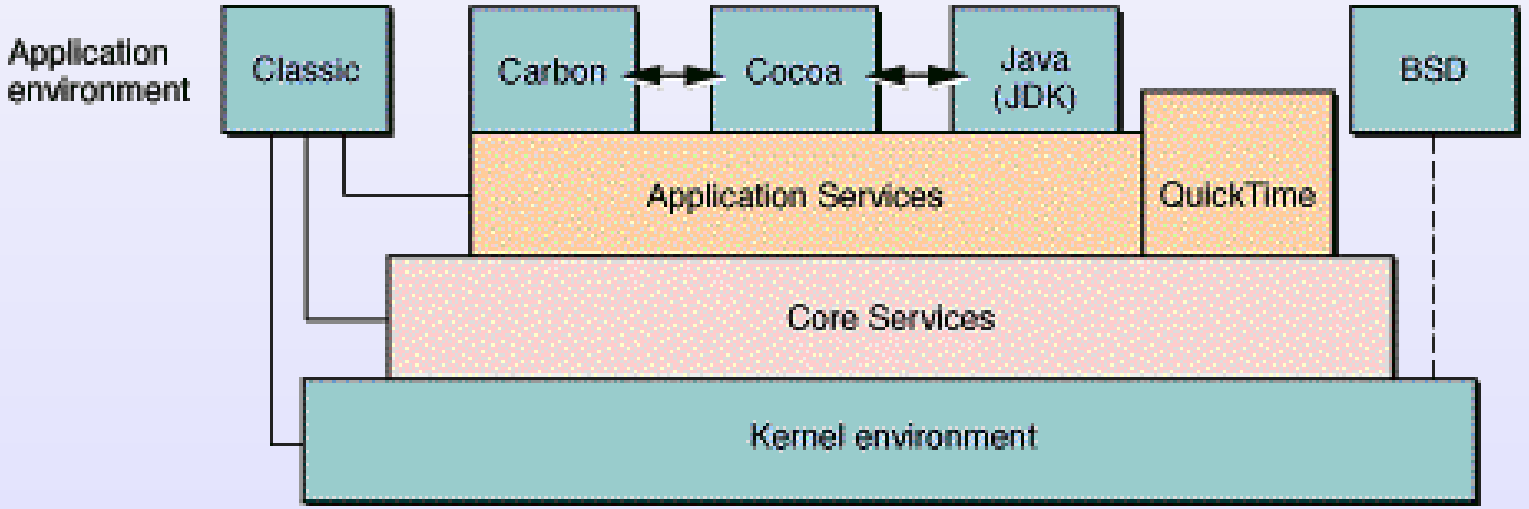
I punti di forza

Un occhio alla licenza

Dove?

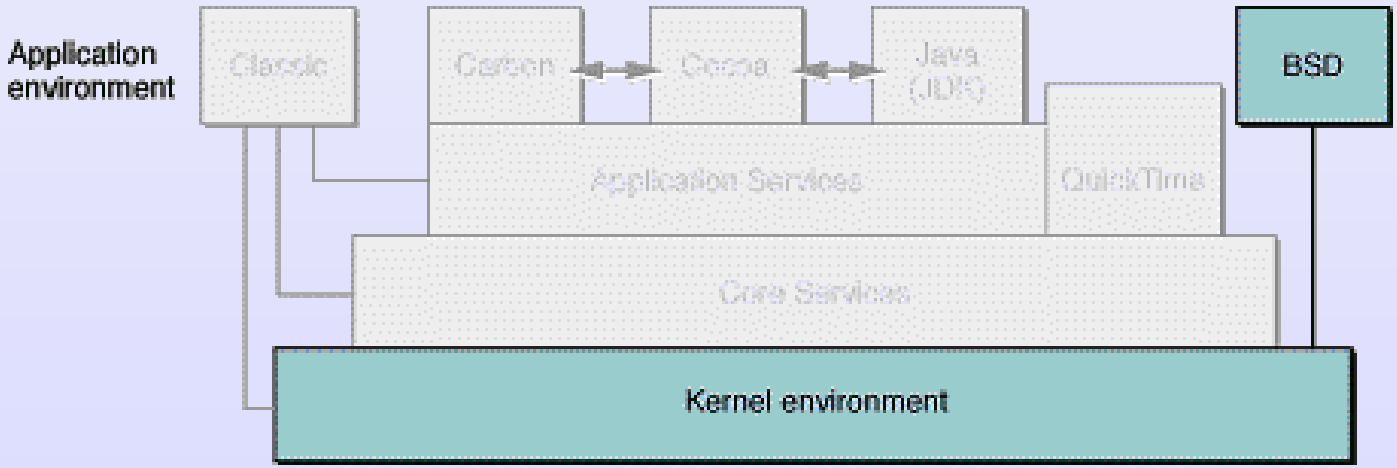
Mac OS X e iPhone OS utilizzano lo stesso kernel

Architettura di OS X



Darwin?

Darwin è il nome con cui si fa riferimento all'ambiente del kernel di Mac OS X. Esso include il kernel vero e proprio (XNU) ed altre parti di software di base.
Darwin è un progetto Open Source ed è considerato un sistema operativo completo, basato su molte delle stesse tecnologie che stanno dietro Mac OS X.



Darwin? (2)

Darwin:

Darwin? (2)

Darwin:

- È basato su:

Darwin? (2)

Darwin:

- È basato su:
  - XNU (Il Kernel "vero e proprio")

Darwin? (2)

Darwin:

- È basato su:
  - XNU (Il Kernel "vero e proprio")
  - Apple Technologies

Darwin? (2)

Darwin:

- È basato su:
  - XNU (Il Kernel "vero e proprio")
  - Apple Technologies
- E' un progetto Open Source, rilasciato sotto i termini della APSL

Darwin? (2)

Darwin:

- È basato su:
  - XNU (Il Kernel "vero e proprio")
  - Apple Technologies
- E' un progetto Open Source, rilasciato sotto i termini della APSL
- Viene portato avanti come progetto separato da Mac OS X

Il kernel visto da Apple

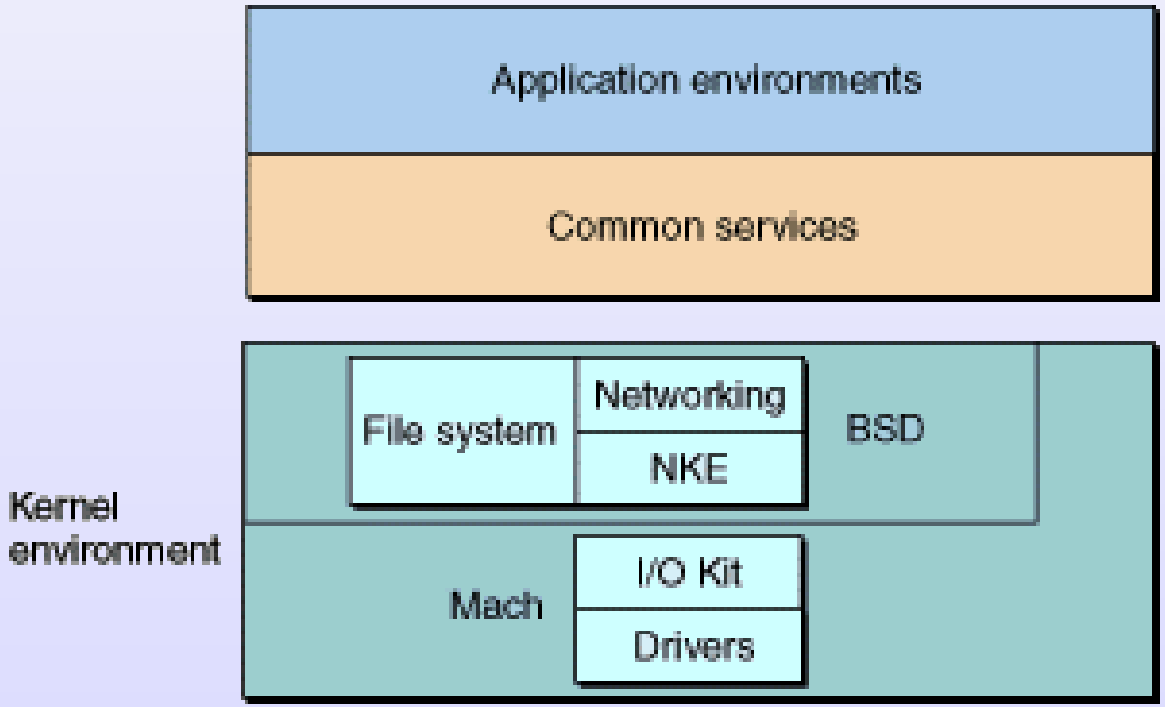
"A kernel, in traditional operating-system terminology, is a small nucleus of software that provides only the minimal facilities necessary for implementing additional operating-system services"

Design and Implementation of the 4.4 BSD Operating System, McKusic, Bostic, Karels, and Quarterman, 1996.

In Mac OS X, l'ambiente del kernel è molto di più del semplice microkernel.
Include infatti Mach Kernel, BSD, I/O Kit, file system, e networking.

Architettura del kernel

Il Kernel di Mac OS X viene chiamato XNU (XNU is not UNIX. Familare..?)



XNU: XNU is not UNIX

Le 3 grandi componenti di XNU

XNU: XNU is not UNIX

Le 3 grandi componenti di XNU

XNU: XNU is not UNIX

Le 3 grandi componenti di XNU

XNU: XNU is not UNIX

Le 3 grandi componenti di XNU

Mach: piccola overview

Mach è un microkernel, su cui si basano i servizi e le primitive fondamentali del kernel di Mac OS X.
Mach gestisce risorse come l'uso della CPU e della memoria, si occupa dello scheduling, garantisce la protezione della memoria. Garantisce inoltre:

Mach: piccola overview

Mach è un microkernel, su cui si basano i servizi e le primitive fondamentali del kernel di Mac OS X.
Mach gestisce risorse come l'uso della CPU e della memoria, si occupa dello scheduling, garantisce la protezione della memoria. Garantisce inoltre:

- Untyped Interprocess Communication (IPC)

Mach: piccola overview

Mach è un microkernel, su cui si basano i servizi e le primitive fondamentali del kernel di Mac OS X.
Mach gestisce risorse come l'uso della CPU e della memoria, si occupa dello scheduling, garantisce la protezione della memoria. Garantisce inoltre:

- Untyped Interprocess Communication (IPC)
- Remote Procedure Calls (RPC)

Mach: piccola overview

Mach è un microkernel, su cui si basano i servizi e le primitive fondamentali del kernel di Mac OS X.
Mach gestisce risorse come l'uso della CPU e della memoria, si occupa dello scheduling, garantisce la protezione della memoria. Garantisce inoltre:

- Untyped Interprocess Communication (IPC)
- Remote Procedure Calls (RPC)
- Scheduler Support for Symmetric Multiprocessing (SMP)

Mach: piccola overview

Mach è un microkernel, su cui si basano i servizi e le primitive fondamentali del kernel di Mac OS X.
Mach gestisce risorse come l'uso della CPU e della memoria, si occupa dello scheduling, garantisce la protezione della memoria. Garantisce inoltre:

- Untyped Interprocess Communication (IPC)
- Remote Procedure Calls (RPC)
- Scheduler Support for Symmetric Multiprocessing (SMP)
- Supporto per la memoria virtuale

Mach: piccola overview

Mach è un microkernel, su cui si basano i servizi e le primitive fondamentali del kernel di Mac OS X.
Mach gestisce risorse come l'uso della CPU e della memoria, si occupa dello scheduling, garantisce la protezione della memoria. Garantisce inoltre:

- Untyped Interprocess Communication (IPC)
- Remote Procedure Calls (RPC)
- Scheduler Support for Symmetric Multiprocessing (SMP)
- Supporto per la memoria virtuale
- Architettura modulare

BSD: piccola overview

Oltre il livello del Mach, troviamo il livello BSD, che garantisce le APIs e i servizi. È basato sul kernel BSD, e fornisce:

BSD: piccola overview

Oltre il livello del Mach, troviamo il livello BSD, che garantisce le APIs e i servizi. È basato sul kernel BSD, e fornisce:

- File system

BSD: piccola overview

Oltre il livello del Mach, troviamo il livello BSD, che garantisce le APIs e i servizi. È basato sul kernel BSD, e fornisce:

- File system
- Networking

BSD: piccola overview

Oltre il livello del Mach, troviamo il livello BSD, che garantisce le APIs e i servizi. È basato sul kernel BSD, e fornisce:

- File system
- Networking
- Supporto syscall

BSD: piccola overview

Oltre il livello del Mach, troviamo il livello BSD, che garantisce le APIs e i servizi. È basato sul kernel BSD, e fornisce:

- File system
- Networking
- Supporto syscall
- Modello di processi BSD, includendo ID e segnali dei processi

BSD: piccola overview

Oltre il livello del Mach, troviamo il livello BSD, che garantisce le APIs e i servizi. È basato sul kernel BSD, e fornisce:

- File system
- Networking
- Supporto syscall
- Modello di processi BSD, includendo ID e segnali dei processi
- Le API del kernel FreeBSD

BSD: piccola overview

Oltre il livello del Mach, troviamo il livello BSD, che garantisce le APIs e i servizi. È basato sul kernel BSD, e fornisce:

- File system
- Networking
- Supporto syscall
- Modello di processi BSD, includendo ID e segnali dei processi
- Le API del kernel FreeBSD
- Molte delle API POSIX

I/O Kit: piccola overview

L'I/O Kit fornisce un framework per lo sviluppo semplificato di driver, oltre che:

I/O Kit: piccola overview

L'I/O Kit fornisce un framework per lo sviluppo semplificato di driver, oltre che:

- Plug and Play

I/O Kit: piccola overview

L'I/O Kit fornisce un framework per lo sviluppo semplificato di driver, oltre che:

- Plug and Play
- Gestione dinamica dei dispositivi

I/O Kit: piccola overview

L'I/O Kit fornisce un framework per lo sviluppo semplificato di driver, oltre che:

- Plug and Play
- Gestione dinamica dei dispositivi
- Caricamento dinamico dei driver

I/O Kit: piccola overview

L'I/O Kit fornisce un framework per lo sviluppo semplificato di driver, oltre che:

- Plug and Play
- Gestione dinamica dei dispositivi
- Caricamento dinamico dei driver
- Possibilità di multiprocessore

Indice

Breve panoramica del Kernel

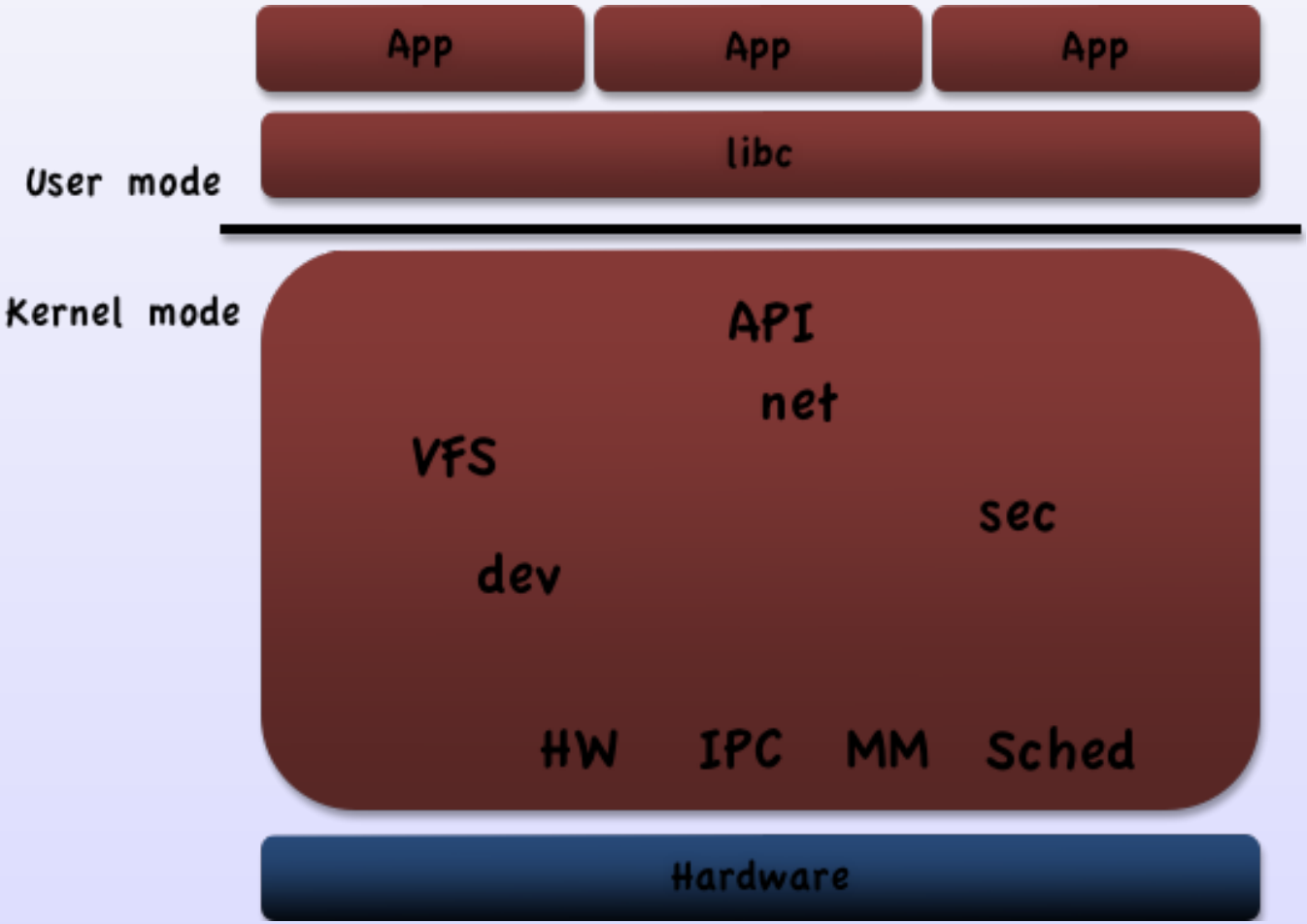
Microkernel vs Kernel monolitico

Address space: 3/1 e 4/4 (ma non solo)

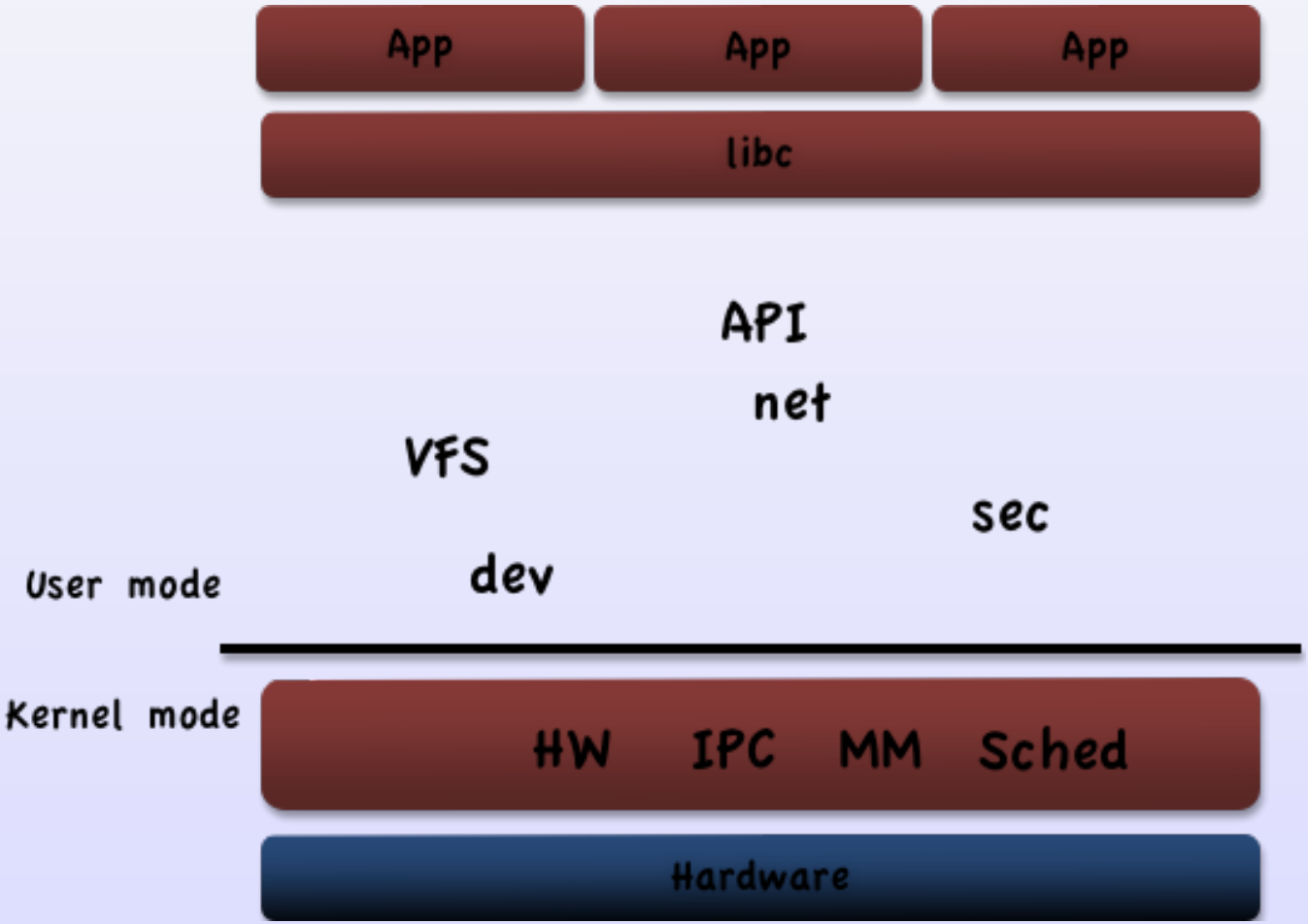
I punti di forza

Un occhio alla licenza

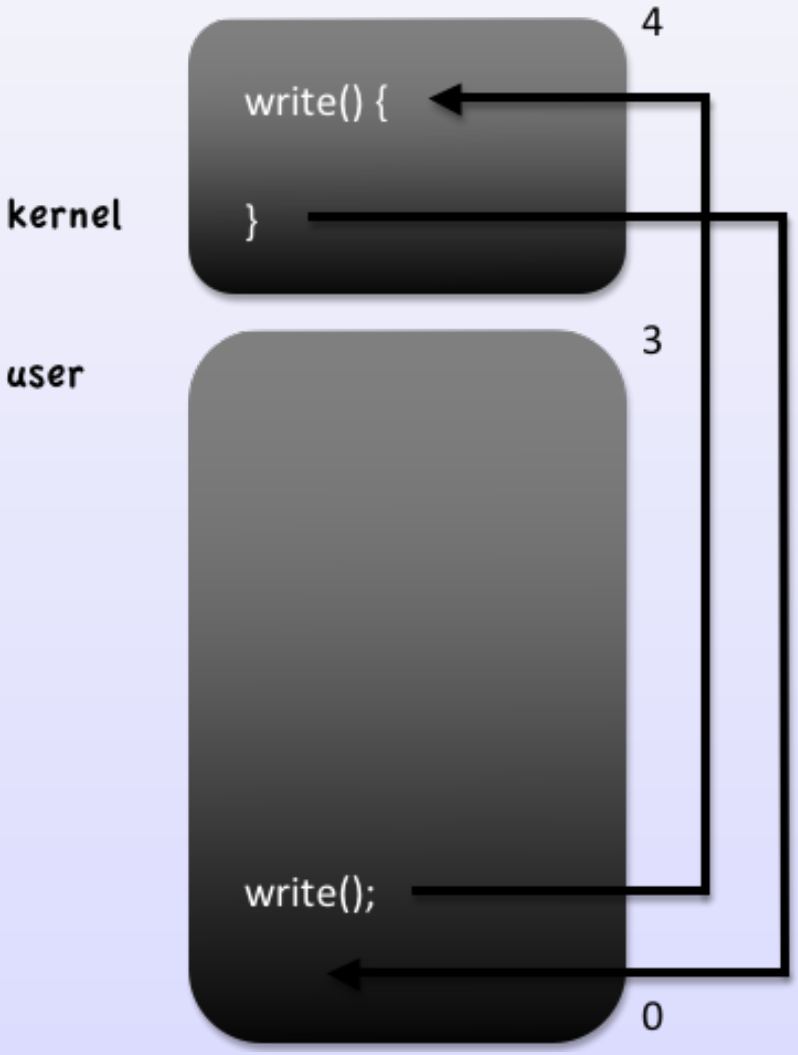
Kernel Monolitico



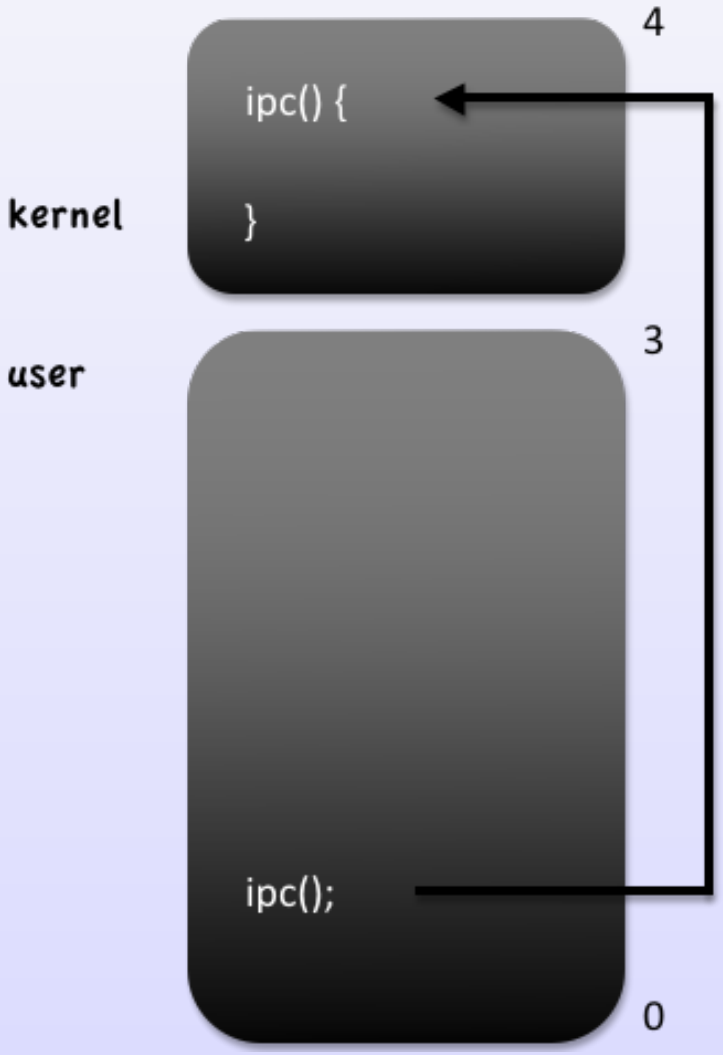
Microkernel



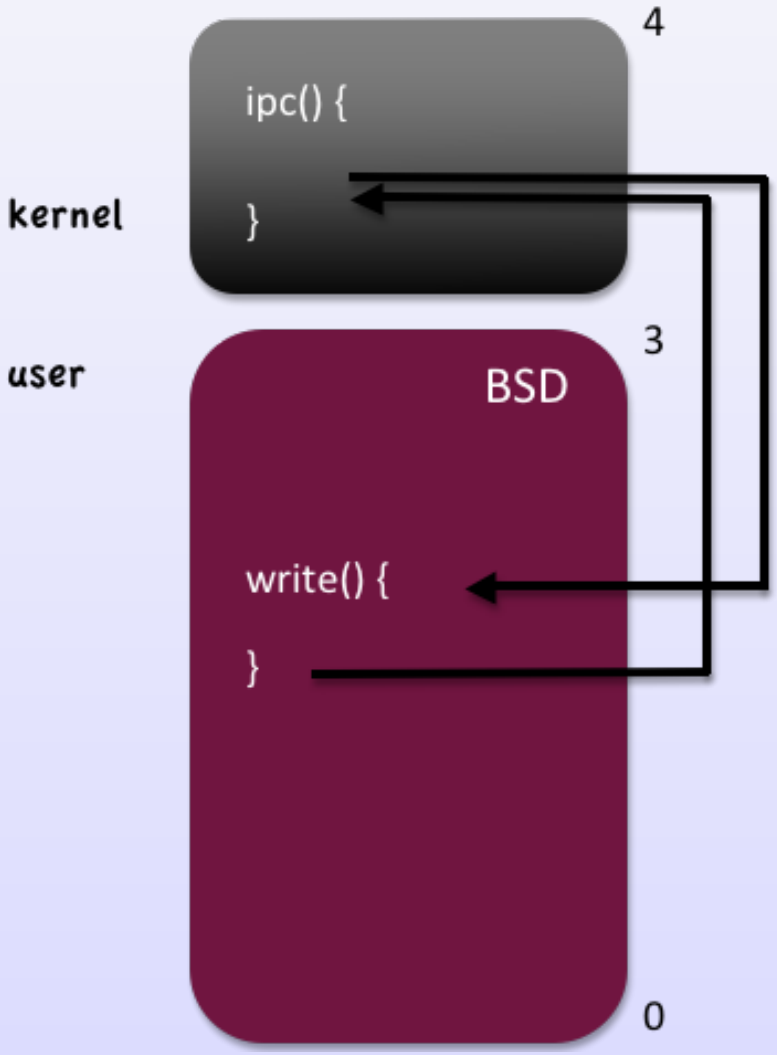
Kernel monolitico: comunicazione



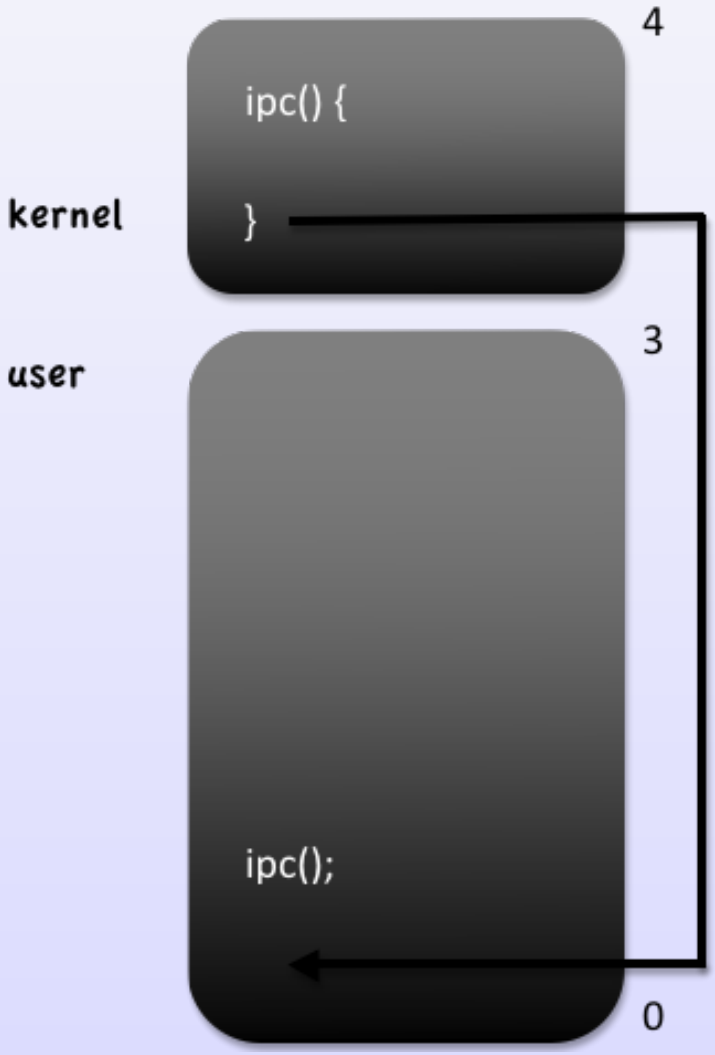
Microkernel: comunicazione



Microkernel: comunicazione



Microkernel: comunicazione



Problema!

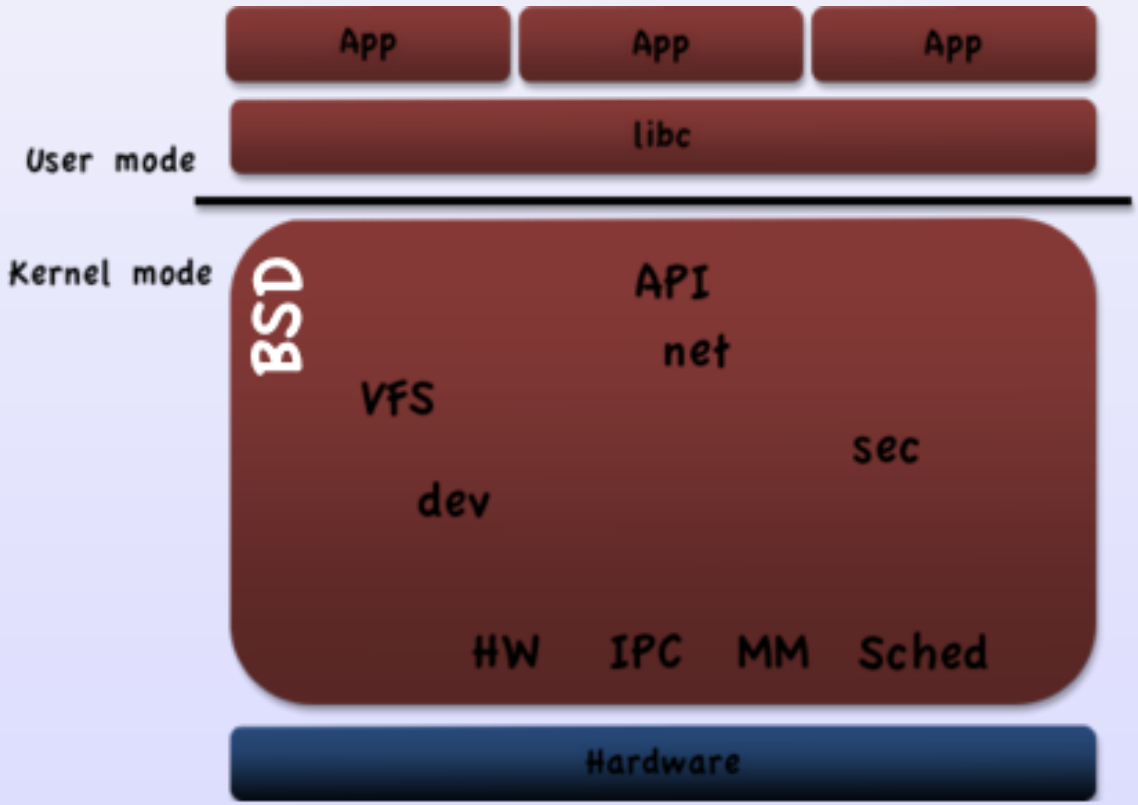
Sorge qui un problema: questo cambiare address space da parte del kernel comporta un problema per cui non funziona bene il **TLB** (Translation Lookaside Buffer).

Il TLB è una cache nella CPU che contiene porzioni della *page table*, una struttura dati che traduce gli indirizzi di memoria virtuali in indirizzi reali.

Il sistema si rallenta ad ogni cambiamento, perché si attende che il TLB venga ripopolato.

Soluzione!

BSD e Mach vengono eseguiti entrambi in kernel mode! (Co-location)



Ma...

Stando così le cose però, Mac OS X **non** ha un microkernel design... In quanto combinazione di Mach (che è *propriamente* un microkernel) e BSD.

Ma...

Stando così le cose però, Mac OS X **non** ha un microkernel design... In quanto combinazione di Mach (che è *propriamente* un microkernel) e BSD.

Ma questa situazione non è proprio quella di un kernel monolitico?

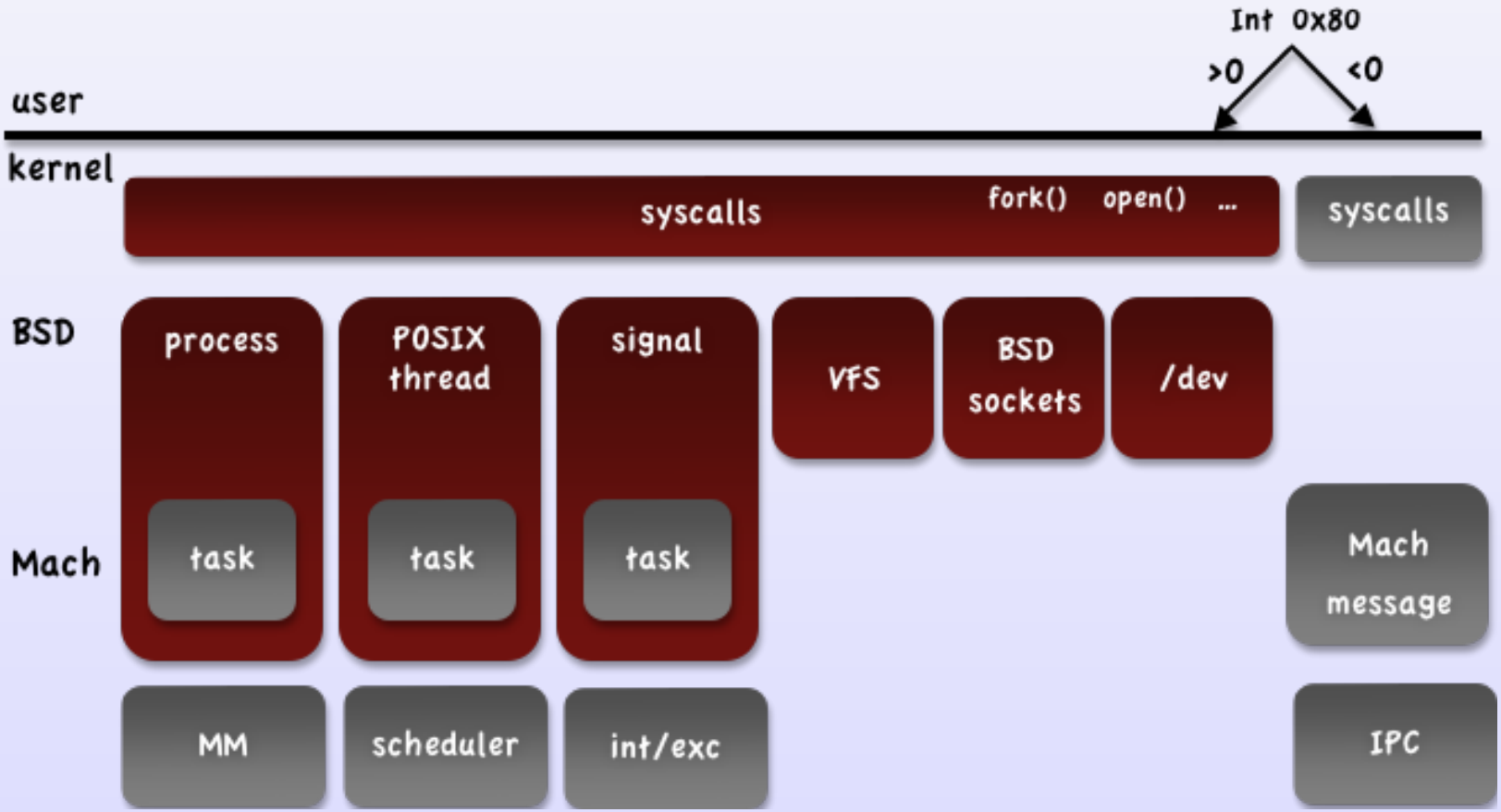
Ma...

Stando così le cose però, Mac OS X **non** ha un microkernel design... In quanto combinazione di Mach (che è *propriamente* un microkernel) e BSD.

Ma questa situazione non è proprio quella di un kernel monolitico?

La risposta è **si**, ma con i dovuti vantaggi.

Panoramica


Int 0x80
>0
<0
user
kernel
syscalls
fork()
open()
...
syscalls
BSD
process
POSIX thread
signal
VFS
BSD sockets
/dev
Mach
task
task
task
Mach message
MM
scheduler
int/exc
IPC

Indice

Breve panoramica del Kernel

Microkernel vs Kernel monolitico

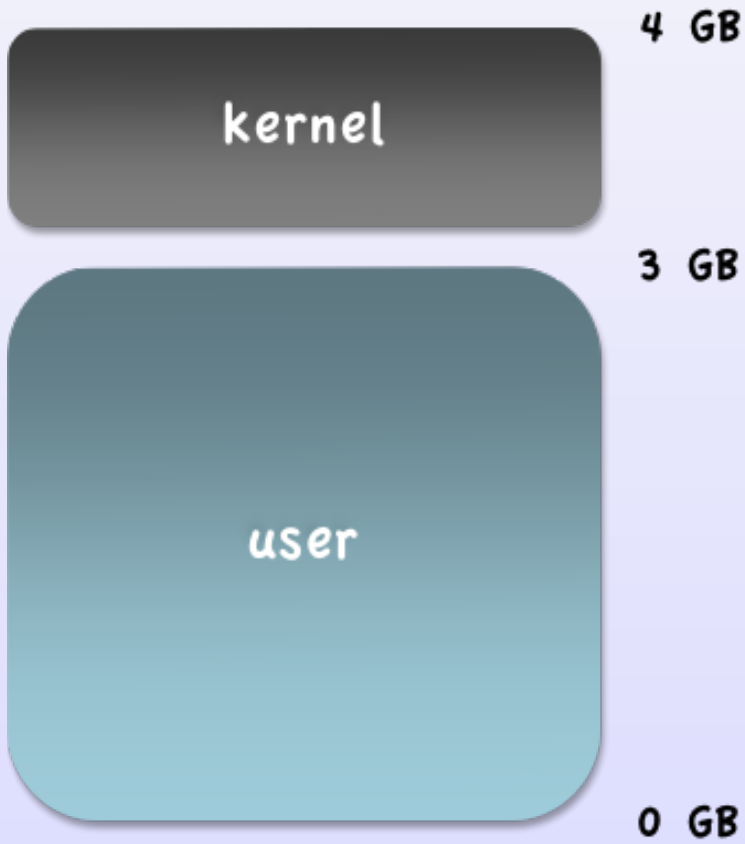
Address space: 3/1 e 4/4 (ma non solo)

I punti di forza

Un occhio alla licenza

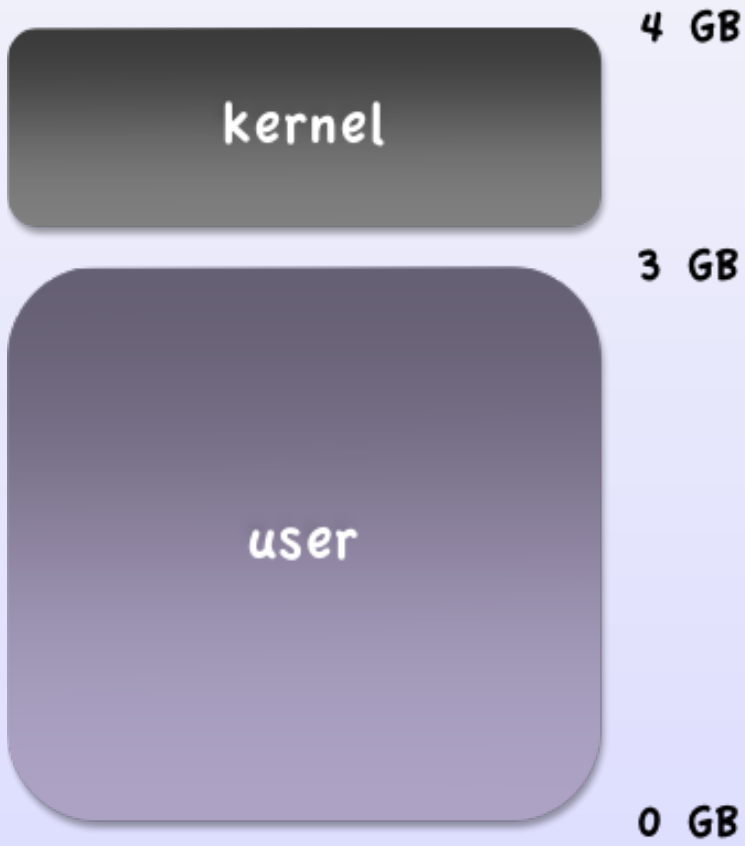
3/1

Utilizzato su Linux



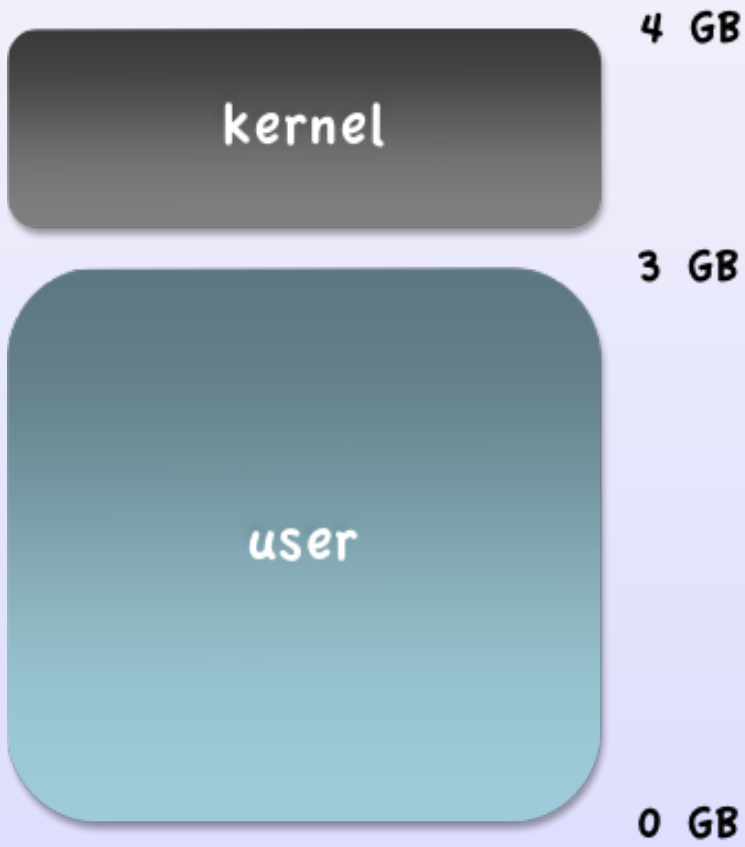
3/1

Lo user space viene cambiato, la *page table* rimane la stessa.



3/1

Lo user space viene cambiato, la *page table* rimane la stessa.



4/4

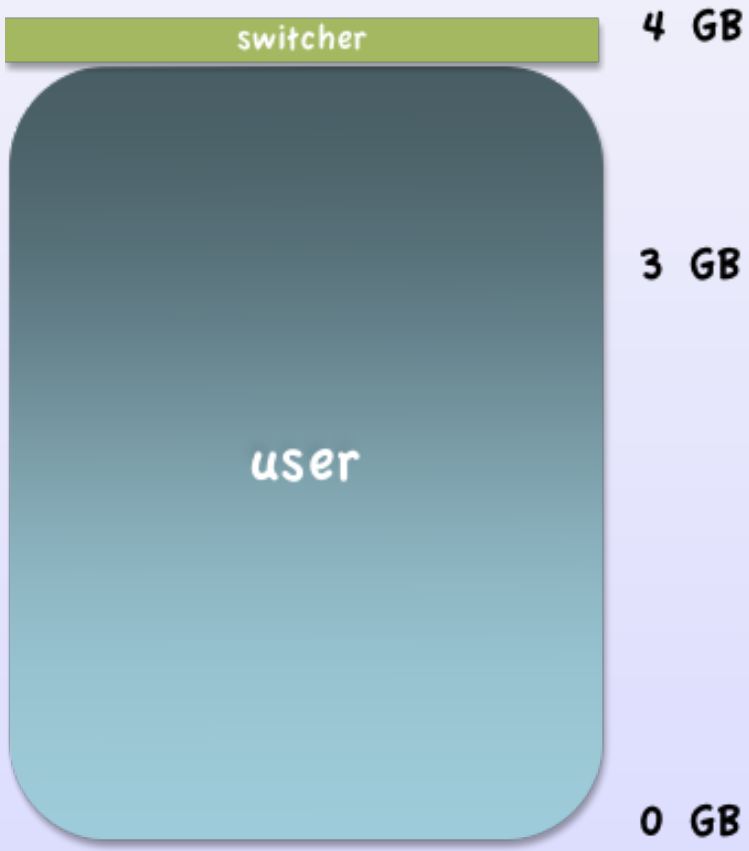
Usato su Mac OS X

4/4

Lo switcher cambia l'address space tra user space e kernel space, assegnando ad entrambi 4 GB.

4/4

Lo switcher cambia l'address space tra user space e kernel space, assegnando ad entrambi 4 GB.



4/4: i vantaggi

Il sistema "4/4" adottato da Mac OS X implementa, come detto, uno switcher che, sostituendo di volta in volta la *page table* permette il cambio di address space.

4/4: i vantaggi

Il sistema "4/4" adottato da Mac OS X implementa, come detto, uno switcher che, sostituendo di volta in volta la *page table* permette il cambio di address space.

Questo comporta due grandi vantaggi:

4/4: i vantaggi

Il sistema "4/4" adottato da Mac OS X implementa, come detto, uno switcher che, sostituendo di volta in volta la *page table* permette il cambio di address space.

Questo comporta due grandi vantaggi:

- I task utente usano 4 GB invece che 3.

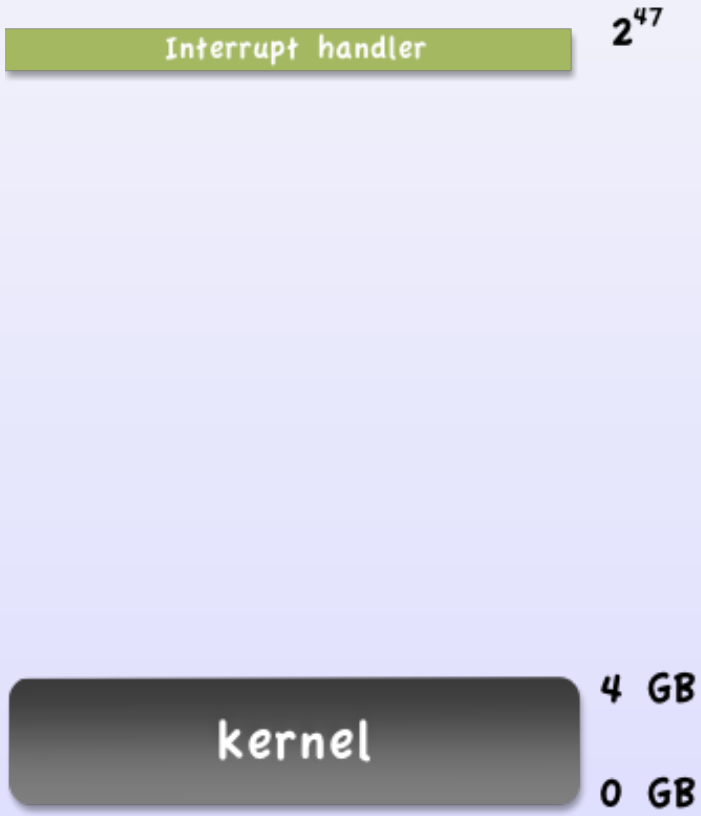
4/4: i vantaggi

Il sistema "4/4" adottato da Mac OS X implementa, come detto, uno switcher che, sostituendo di volta in volta la *page table* permette il cambio di address space.

Questo comporta due grandi vantaggi:

- I task utente usano 4 GB invece che 3.
- C'è la possibilità di mappare più device, come grosse schede grafiche, nell'address space del kernel.

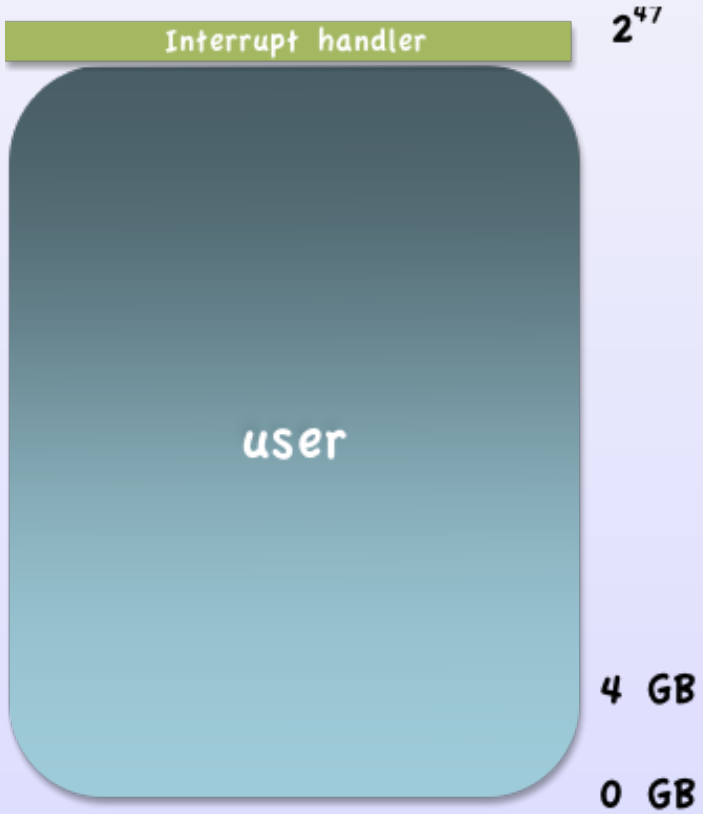
x86_64



x86_64

x86_64

Lo stesso kernel, a 32 bit, pu quindi essere eseguito su macchine a 32 o a 64 bit.



Indice

Breve panoramica del Kernel

Microkernel vs Kernel monolitico

Address space: 3/1 e 4/4 (ma non solo)

I punti di forza

Un occhio alla licenza

Numero 6

Perfetta conformità agli standard POSIX

Numero 5

La netta separazione tra il codice di gestione delle funzionalità di basso livello (Mach) e il BSD-server permettono, oltre che una maggiore pulizia del codice stesso, una migliore astrazione da parte dell'ambiente kernel

Numero 4

Il codice del kernel è a 32 bit, ma supporta applicazioni utente a 64 bit

Numero 3

L'I/O Kit permette riutilizzo del codice per i driver senza bisogno di duplicarlo

Numero 2

La struttura di XNU permette un livello elevatissimo di mantenibilità

Numero 1

Questa è facile...

Numero 1

Questa è facile...

È **Open Source!**

Indice

Breve panoramica del Kernel

Microkernel vs Kernel monolitico

Address space: 3/1 e 4/4 (ma non solo)

I punti di forza

Un occhio alla licenza

APSL

Darwin è rilasciato sotto licenza **APSL** (Apple Public Source License).
Vediamola un po' più in dettaglio:

APSL

Darwin è rilasciato sotto licenza **APSL** (Apple Public Source License). Vediamola un po' più in dettaglio:

- È stata scelta appositamente per invogliare la community di sviluppatori open source a migliorare il progetto Darwin.

APSL

Darwin è rilasciato sotto licenza **APSL** (Apple Public Source License). Vediamola un po' più in dettaglio:

- È stata scelta appositamente per invogliare la community di sviluppatori open source a migliorare il progetto Darwin.
- L'attuale versione (2.0) è conforme alle linee guida della Free Software Foundation

APSL

Darwin è rilasciato sotto licenza **APSL** (Apple Public Source License). Vediamola un po' più in dettaglio:

- È stata scelta appositamente per invogliare la community di sviluppatori open source a migliorare il progetto Darwin.
- L'attuale versione (2.0) è conforme alle linee guida della Free Software Foundation
- Tuttavia, la FSF ha consigliato agli sviluppatori di non rilasciare altri software con questa licenza, perché non è compatibile con la GPL

Tuttavia...

...ci sono alcuni limiti:

Tuttavia...

...ci sono alcuni limiti:

- Non esistono repository, nonostante il codice sia "free"

Tuttavia...

...ci sono alcuni limiti:

- Non esistono repository, nonostante il codice sia "free"
- Mancano alcune parti di codice

Tuttavia...

...ci sono alcuni limiti:

- Non esistono repository, nonostante il codice sia "free"
- Mancano alcune parti di codice

Tuttavia il kernel può comunque essere compilato in un sistema funzionante.

References

Qualche sito utile per chi ne volesse sapere di più:

- http://developer.apple.com/opensource/index.html - **Home Page dei progetti Open Source Apple**
- http://developer.apple.com/documentation/Darwin/index.html - **Darwin Guides**
- http://en.wikipedia.org/wiki/Darwin_(operating_system) - **Darwin OS su Wikipedia**
- http://www.kernelthread.com/mac/osx/arch_xnu.html - **XNU su kernelthread.com**

Grazie

Grazie per l'attenzione!

Domande?

(a cui spero di saper rispondere =))

La presentazione sarà disponibile su http://www.gulp.linux.it