

# Виртуализация

проф. д-р инж. Христо Вълчанов

<http://cs.tu-varna.bg>

# Началото

---

- 1967г.
- IBM създават System 370 mainframe
- Операционна система CP/CMS (VM/CMS)



# Причини за развитие на виртуализацията

---

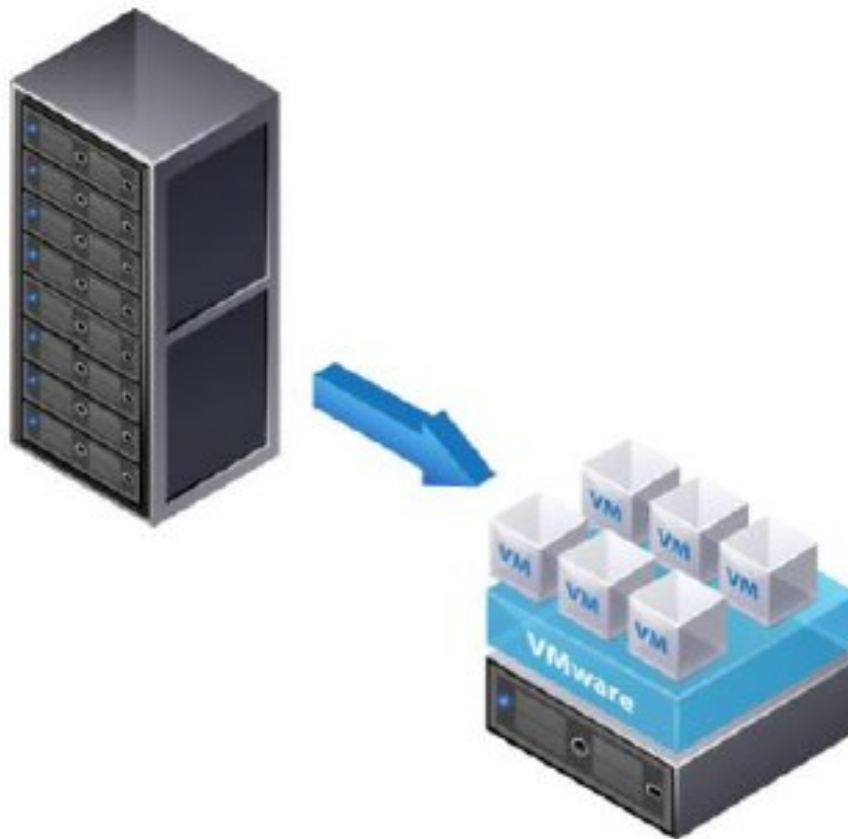
Предизвикателства:

- Ниска степен на натоварване на инфраструктурата
- Повишаване на разходите на физическата инфраструктура
- Повишаване на IT разходите за управление
- Недостатъчно защита от грешки и бедствия
- Висока поддръжка за крайните потребителски настолни компютри

# Сървърна консолидация

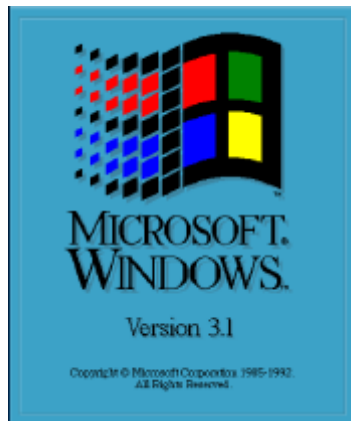
---

- По-добро използване на хардуера – от 5% до 80%



# Други приложения

---



Microsoft  
**SQL Server**



# Среда за научни изследвания, разработка и тестване

---

- Възможност за записване, възстановяване и презареждане на състоянието
- Възможност за миграции и репликации

# Как работи виртуализацията?

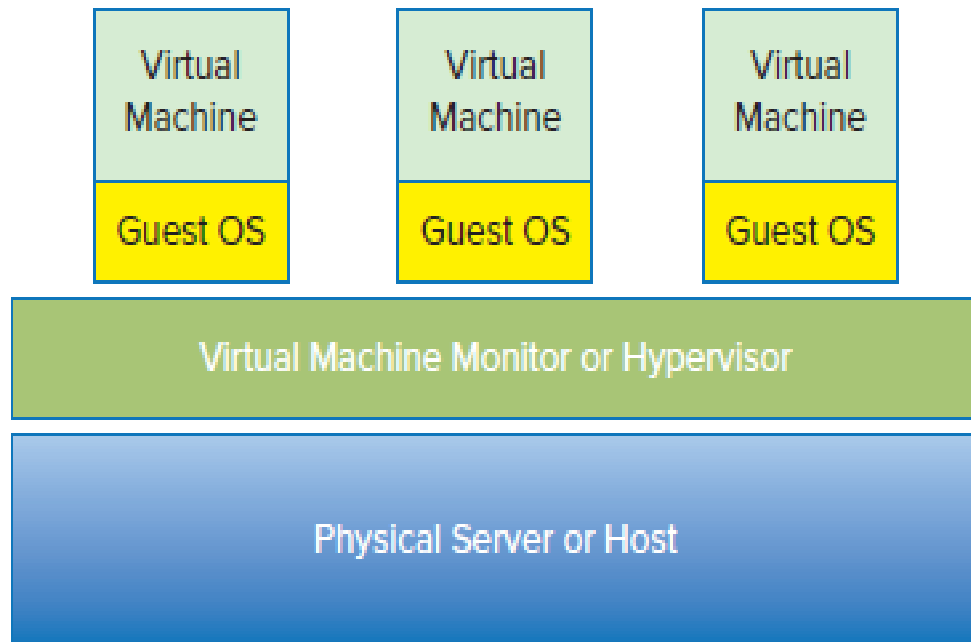
---

- Трансформира или виртуализира хардуерните ресурси на x86-базирани компютри, в напълно функционална виртуална машина
- Виртуалната машина е цялостна система от ОС и приложен софтуер
- Хардуерните ресурси се разпределят динамично и прозрачно
- Няколко ОС могат да се работят едновременно

# Термини

---

- **HOST** – физическата платформа
- **GUEST** – ОС, изпълняваща се на виртуалната машина
- **Virtual Machine Monitor / Hypervisor** – софтуер, поддържащ виртуализацията





# Виртуална машина

---

- Изолиран софтуерен контейнер, който може да стартира собствена ОС и приложения
- VM се държи като истински компютър и има собствени виртуални (софтуерно базирани) процесор, памет, диск и мрежови адаптери
- Гост ОС не може да направи разлика между физическа и виртуална машина



# Предимства на ВМ

---

- **Съвместимост** - виртуалните машини са съвместими с всички стандартни x86 компютри
- **Изоляция** - виртуалните машини са изолирани една от друга, както физическите отделни компютри
- **Капсулация** - виртуалните машини капсулират пълна компютърна среда
- **Хардуерна независимост** - виртуалните машини са независими от прилежащия хардуер

# ВМ и хипервайзор

---

- От гледна точка на хипервайзора, ВМ е набор от файлове



# Видове VM

---

- **Process Virtual Machine** – предназначена е за стартиране на един процес
- **System Virtual Machine** – готова системна платформа, която поддържа изпълнението на цяла ОС

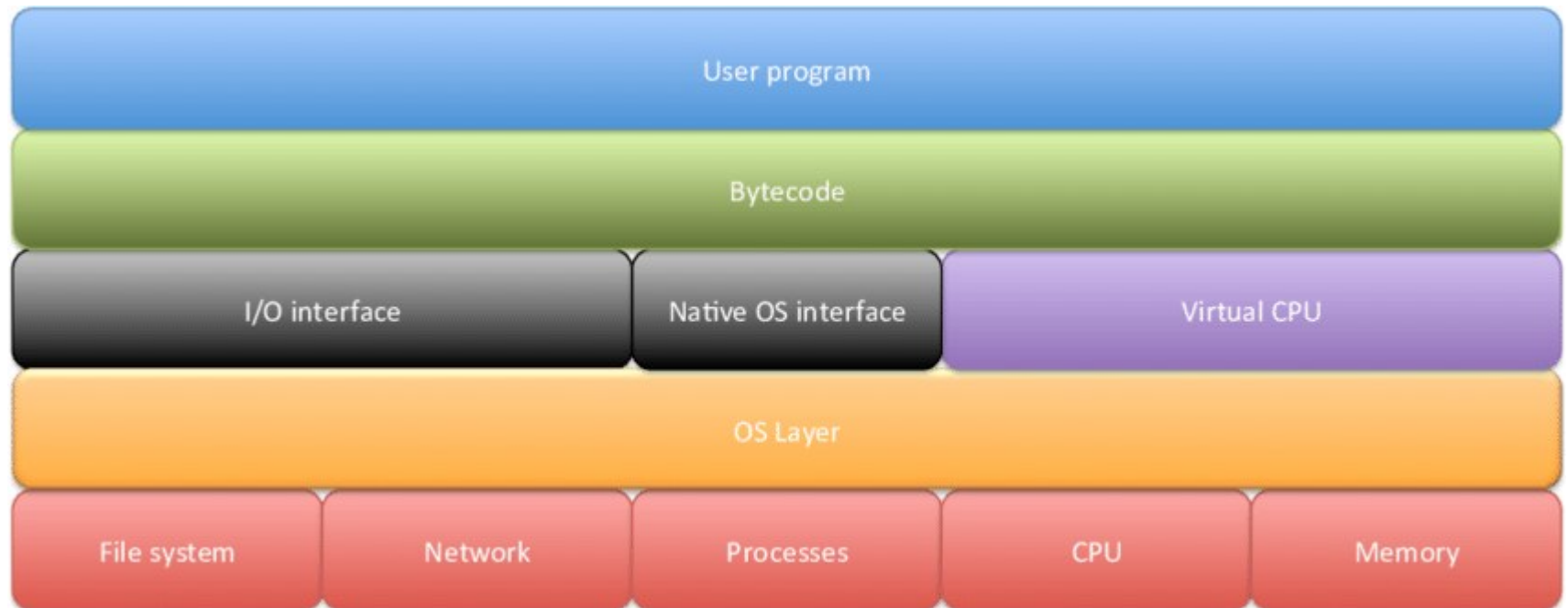
# Процесни ВМ

---

- Поддържа само един процес
- Осигурява платформена независимост
- Създава се при стартиране на процес
- Унищожава се при завършване на процеса
- Ограничение от ресурсите на самата виртуална машина

# Процесни ВМ

---



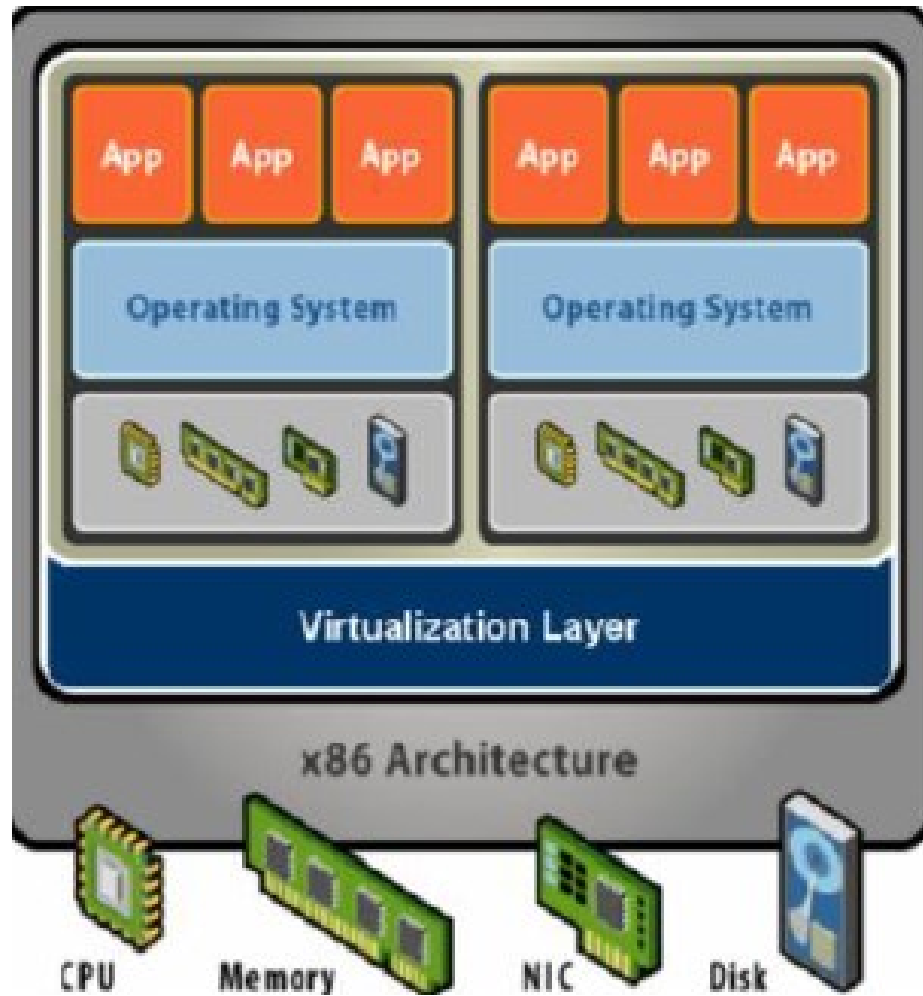
# Системни ВМ

---

- Виртуализираният софтуер е разположен между хардуера и гост ОС
- Осигурява абстракция на хардуера от гледна точка на софтуера във ВМ
- Позволява множество ОС да работят едновременно върху един физически компютър

# Системни ВМ

---





# Изисквания към хипервайзорите

---

➤ **1974г. Попек и Голдберг дефинират изисквания към компютърна система за поддръжка на виртуализация:**

1. Точност – създаваната среда за VM е по същество идентична на оригиналната физическа машина
2. Изолация или защитеност – хипервайзорът трябва да има пълен контрол върху системните ресурси
3. Производителност – трябва да има много малка или да няма разлика в производителността на VM и физическата машина

**Изпълнението на критерий 3 -> ефикасен хипервайзор**

# Предимства на ВМ

---

## Физически машини

- Неудобни за преместване
- По-трудни за поддръжка
- Хардуерни ограничения

## Виртуални машини

- Лесни за преместване
- Лесни за управление
- Осигуряват възможност за стартиране на стари приложения
- Позволяват сървърна консолидация

# Видове хипервайзори

## Тип 1 – Bare-metal

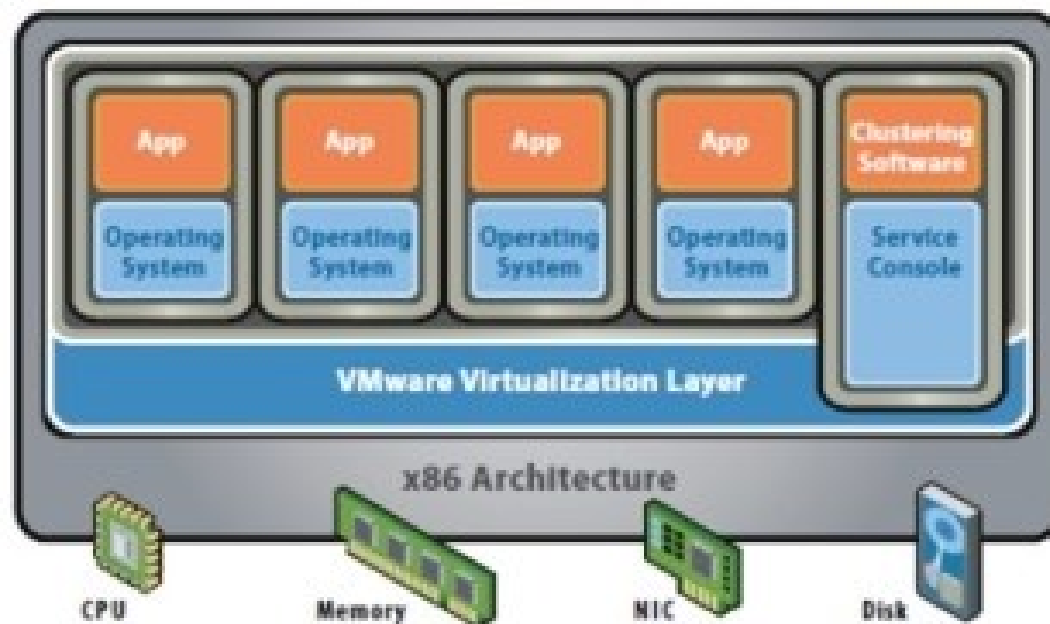
- Работят директно върху хардуера
- Не изисква отделна ОС
- Висока ефективност и производителност

*VMware ESXi*

*Microsoft Hyper-V*

*Citrix Xen server*

*KVM*



# Пример



# Видове хипервайзори

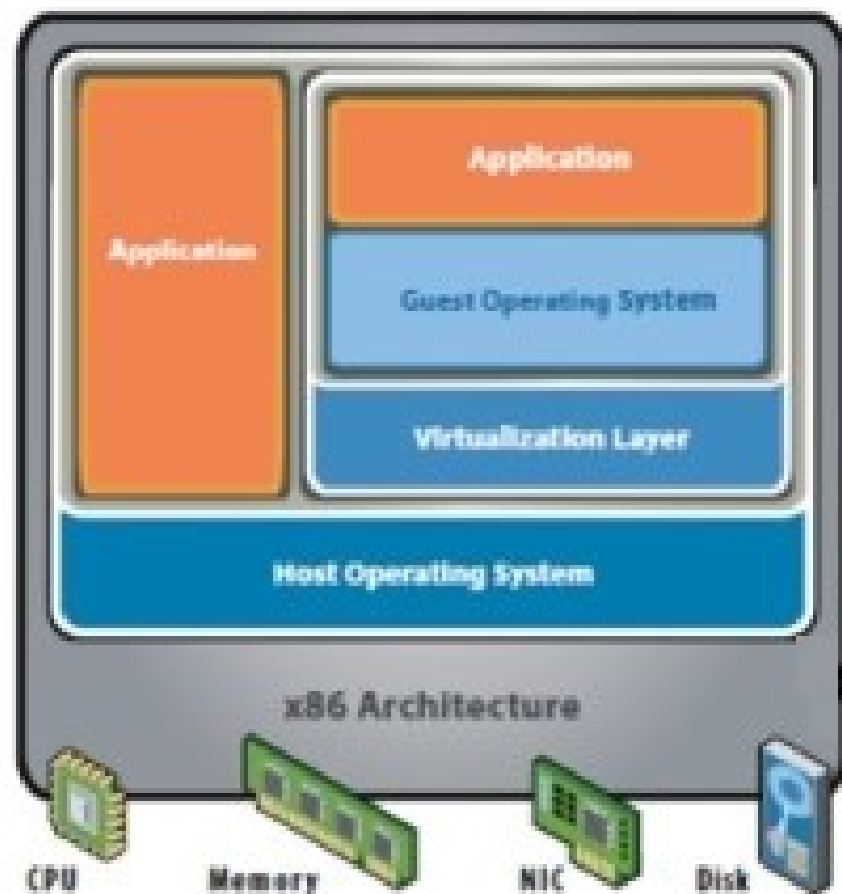
## Тип 2 – Hosted

- Изисква инсталирана ОС
- Работи като приложение в средата на ОС

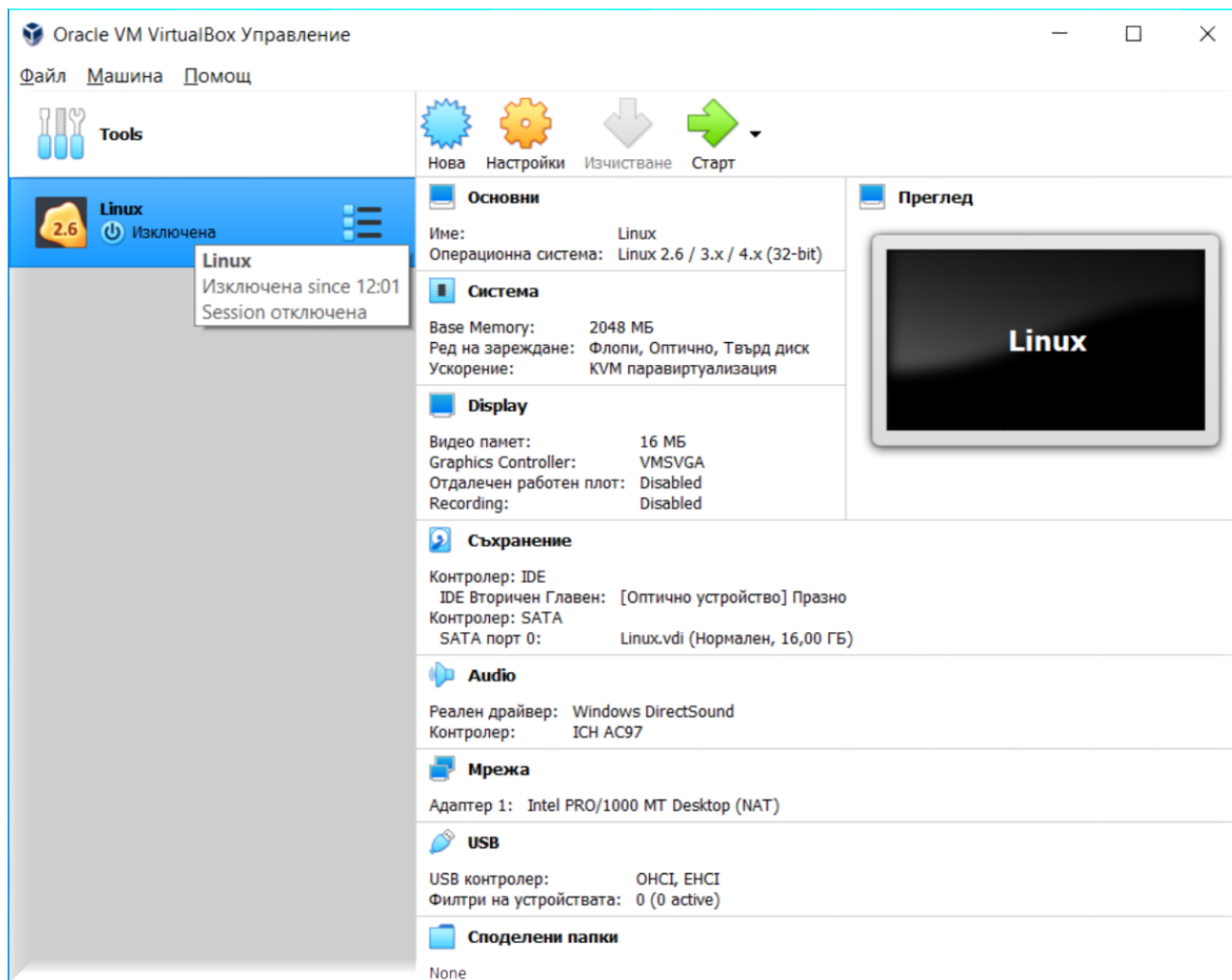
*VMware workstation*

*Oracle VirtualBox*

*Microsoft Virtual PC*



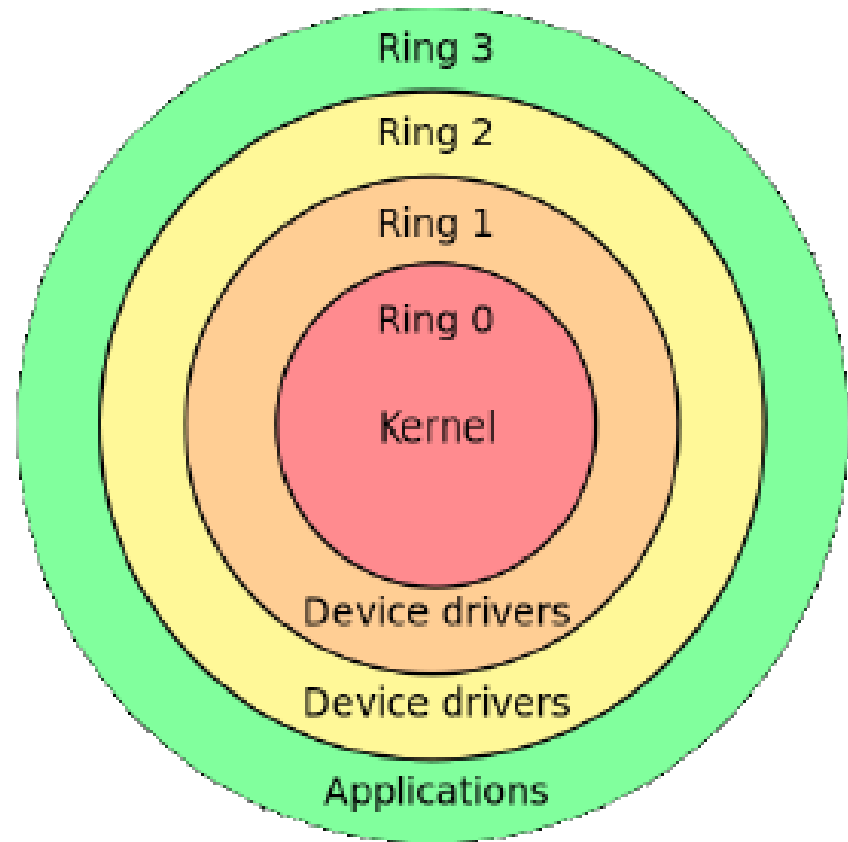
# Пример



# Нива на привилегии на процесора

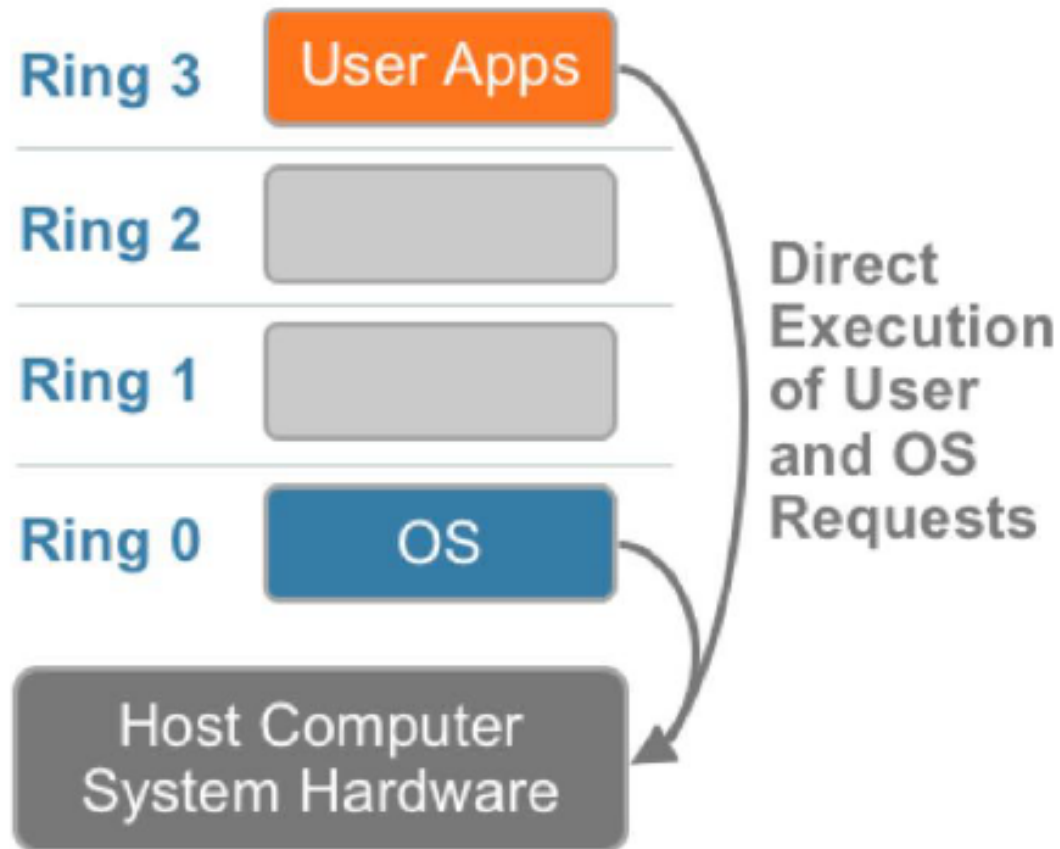
---

- X86 архитектура на привилегии: 4 нива
- В повечето съвременни системи нива 1 и 2 не се използват



# Изпълнение при не-виртуализирана система

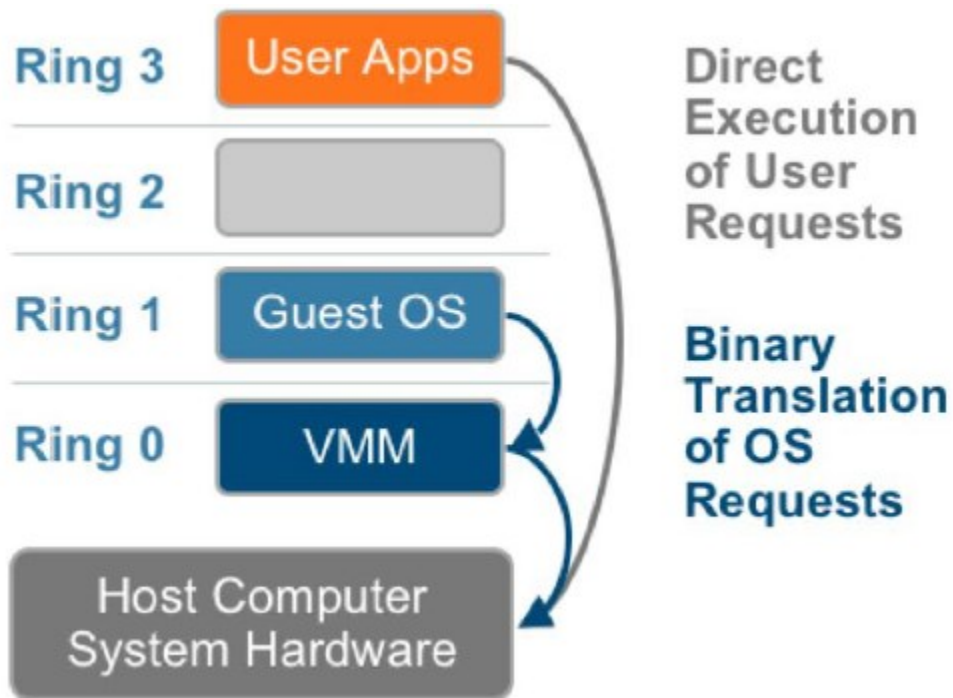
---





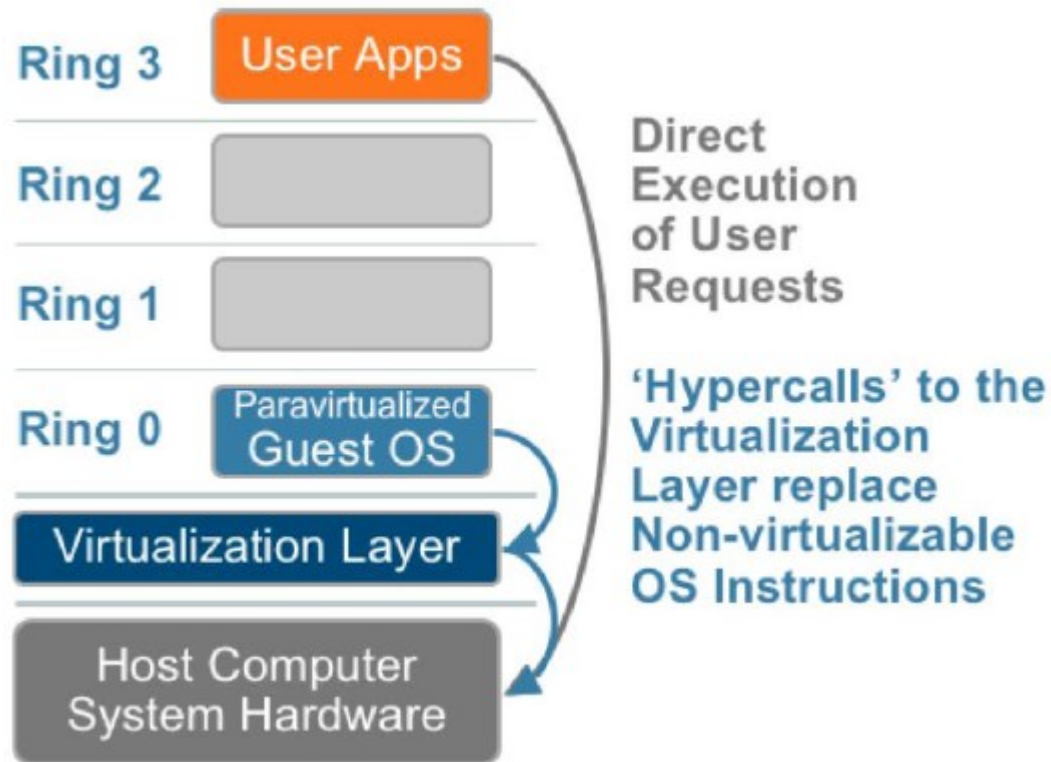
# Бинарно транслиране

---

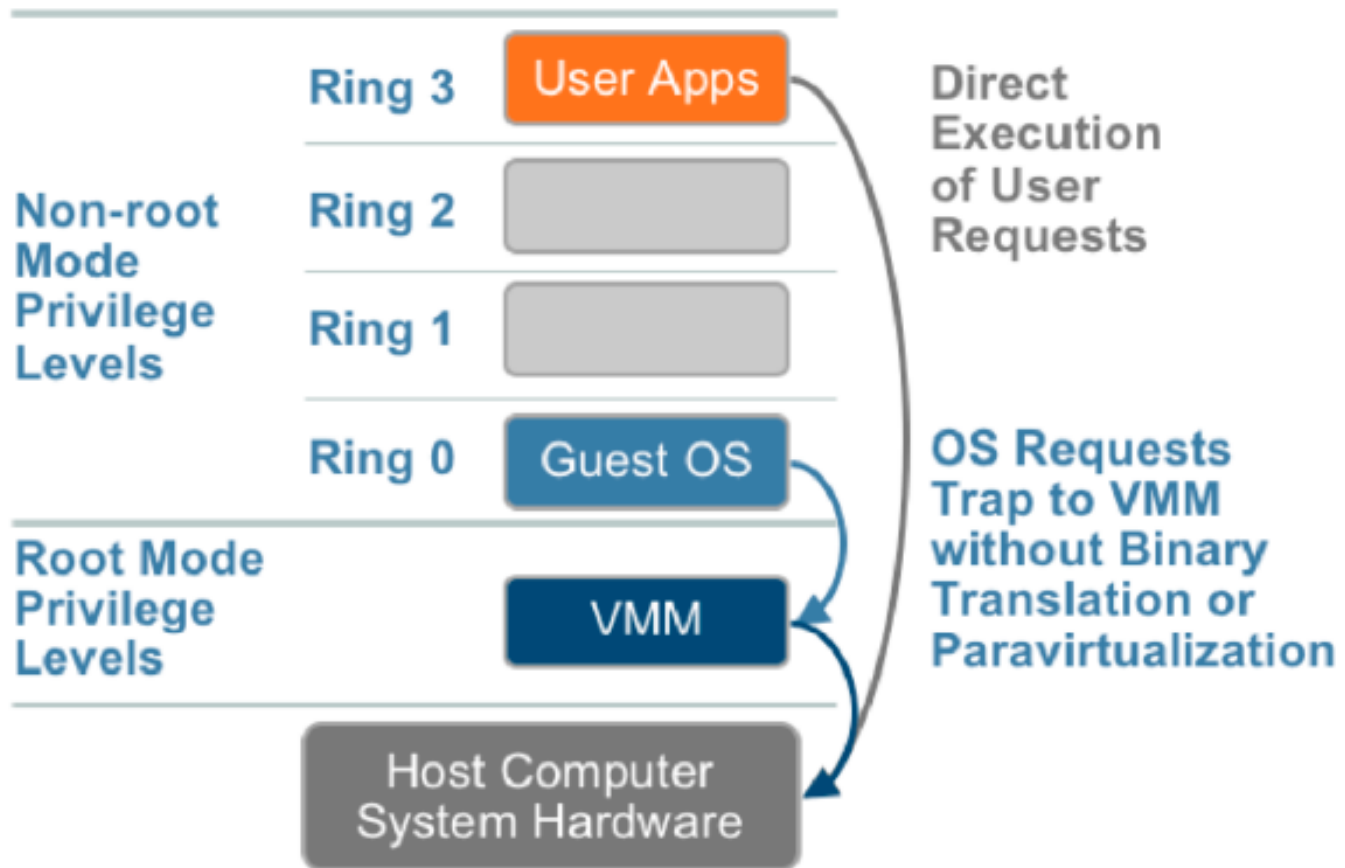


# Паравиртуализация

---

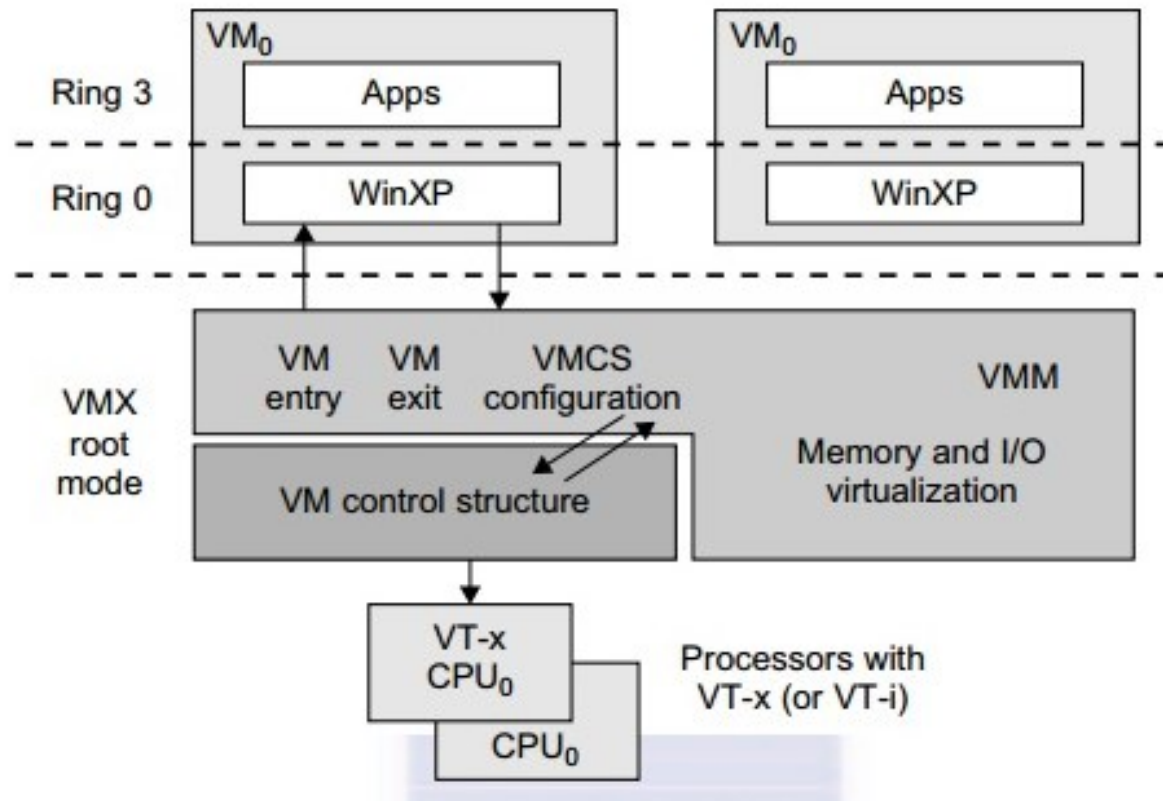


# Хардуерно подпомогната виртуализация



# Хардуерно подпомогната виртуализация

- Intel: VT-x (x86), VT-i (Itanium)
- AMD: AMD-V (x86)



# Хардуерно подпомогната виртуализация

---

- Нов режим на работа на CPU:
  - **VT Root операции** (за хипервайзора)
  - **Non root операции** (за гост ОС)
- Елиминира се компресията на ринга
- Нови инструкции на процесора – Virtual Machine Extensions (VMX)

*VMPTRLD, VMPTRST, VMCLEAR, VMREAD, VMWRITE,  
VMCALL, VMLAUNCH, VMRESUME, VMXOFF, VMXON*

# VMCS

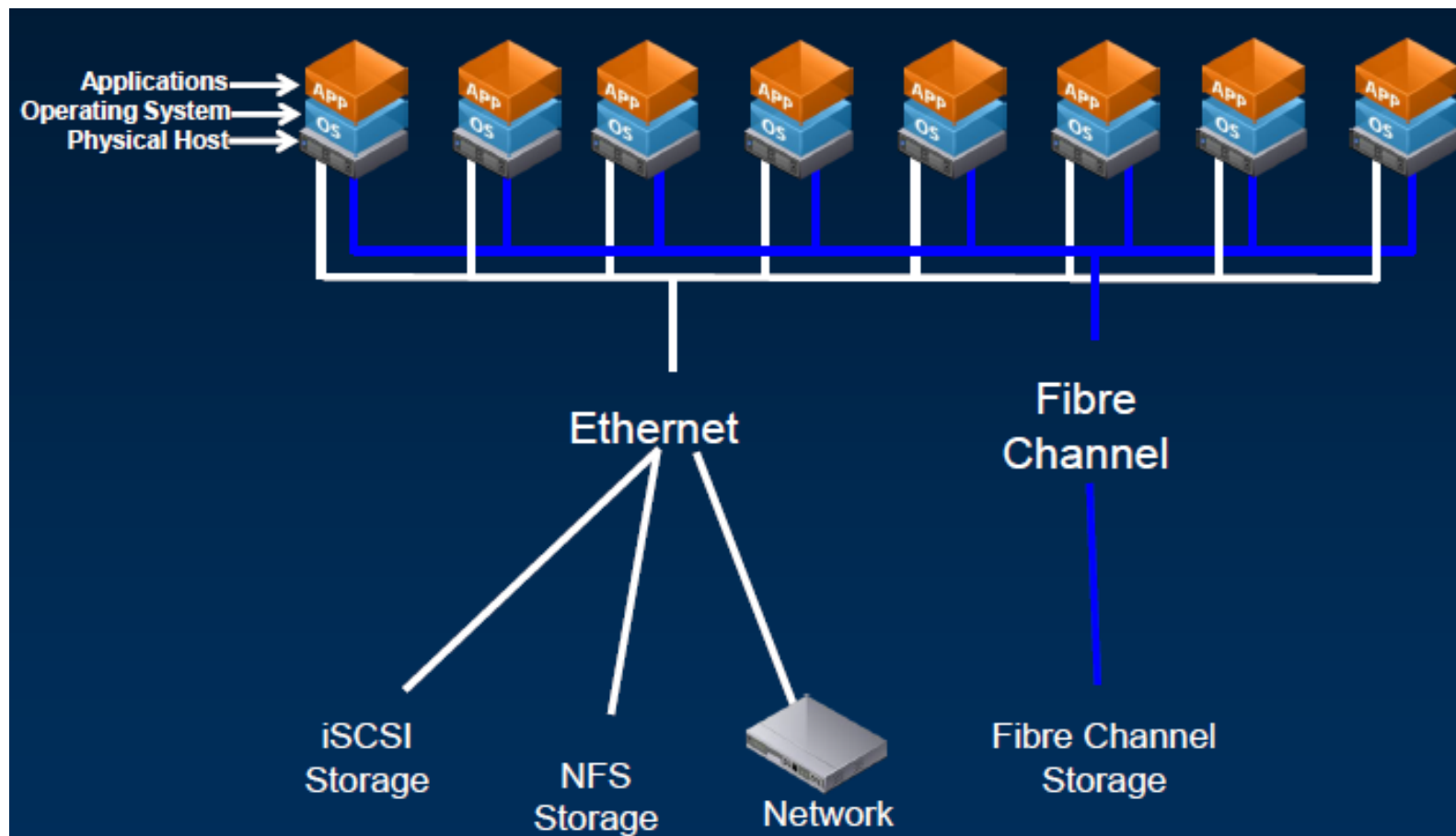
GUEST STATE AREA				
CR0	CR3		CR4	
DR7				
RSP	RIP		RFLAGS	
CS	Selector	Base Address	Segment Limit	Access Right
SS	Selector	Base Address	Segment Limit	Access Right
DS	Selector	Base Address	Segment Limit	Access Right
ES	Selector	Base Address	Segment Limit	Access Right
FS	Selector	Base Address	Segment Limit	Access Right
GS	Selector	Base Address	Segment Limit	Access Right
LDTR	Selector	Base Address	Segment Limit	Access Right
TR	Selector	Base Address	Segment Limit	Access Right
GDTR	Selector	Base Address	Segment Limit	Access Right
IDTR	Selector	Base Address	Segment Limit	Access Right
IA32_DEBUGCTL	IA32_SYSENTER_CS	IA32_SYSENTER_ESP	IA32_SYSENTER_EIP	
IA32_PERF_GLOBAL_CTRL	IA32_PAT	IA32_EFER	IA32_BNDCFGS	
SMBASE				
Activity state	Interruptibility state			
Pending debug exceptions				
VMCS link pointer				
VMX-preemption timer value				
Page-directory-pointer-table entries	PDPTE0	PDPTE1	PDPTE2	PDPTE3
Guest interrupt status				
PML index				
HOST STATE AREA				
CR0	CR3		CR4	
RSP	RIP			
CS	Selector			
SS	Selector			
DS	Selector			
ES	Selector			
FS	Selector	Base Address		
GS	Selector	Base Address		
TR	Selector	Base Address		
GDTR	Base Address			
IDTR	Base Address			
IA32_SYSENTER_CS	IA32_SYSENTER_ESP		IA32_SYSENTER_EIP	
IA32_PERF_GLOBAL_CTRL	IA32_PAT		IA32_EFER	

CONTROL FIELDS				
Pin-Based VM-Execution Controls	External-interrupt exiting		NMI exiting	Virtual NMIs
	Activate VMX-preemption timer		Process posted interrupts	
Primary processor-based VM-execution controls	Interrupt-window exiting		Use TSC offsetting	
	HLT exiting	INVLPG exiting	MWAIT exiting	RDPMSR exiting
	RDTSR exiting	CR3-load exiting	CR3-store exiting	CR8-load exiting
	CR8-store exiting	Use TPR shadow	NMI-window exiting	MOV-DR exiting
	Unconditional I/O exiting	Use I/O bitmaps	Monitor trap flag	Use MSR bitmaps
	MONITOR exiting	PAUSE exiting	Activate secondary controls	
Secondary processor-based VM-execution controls	Virtualize APIC accesses	Enable EPT	Descriptor-table exiting	Enable RDTSCP
	Virtualize x2APIC mode	Enable VPID	WBINVD exiting	Unrestricted guest
	APIC-register virtualization		Virtual-interrupt delivery	PAUSE-loop exiting
	RDRAND exiting	Enable INVPCID	Enable VM functions	VMCS shadowing
	Enable ENCLS exiting	RDSEED exiting	Enable PML	EPT-violation #VE
	Conceal VMX non-root operation from Intel PT		Enable XSAVES/XRSTORS	
	Mode-based execute control for EPT		Use TSC scaling	
Exception Bitmap		I/O-Bitmap Addresses		TSC-offset
Guest/Host Masks for CR0		Guest/Host Masks for CR4	Read Shadows for CR0	Read Shadows for CR4
CR3-target value 0	CR3-target value 1	CR3-target value 2	CR3-target value 3	CR3-target count
APIC Virtualization	APIC-access address		Virtual-APIC address	
	EOI-exit bitmap 0	EOI-exit bitmap 1	EOI-exit bitmap 2	EOI-exit bitmap 3
	Posted-interrupt notification vector		Posted-interrupt descriptor address	
Read bitmap for low MSRs	Read bitmap for high MSRs	Write bitmap for low MSRs	Write bitmap for low MSRs	
Executive-VMCS Pointer		Extended-Page-Table Pointer	Virtual-Processor Identifier	
PLE_Gap	PLE_Window	VM-function controls	VMREAD bitmap	VMWRITE bitmap
ENCLS-exiting bitmap		PML address		
Virtualization-exception information address		EPTP index		XSS-exiting bitmap
VM-EXIT CONTROL FIELDS				
VM-Exit Controls	Save debug controls		Host address space size	Load IA32_PERF_GLOBAL_CTRL
	Acknowledge interrupt on exit	Save IA32_PAT	Load IA32_PAT	Save IA32_EFER
	Save VMX-preemption timer value	Clear IA32_BNDCFGS		Conceal VM exits from Intel PT
VM-Exit Controls for MSRs	VM-exit MSR-store count		VM-exit MSR-store address	
	VM-exit MSR-load count		VM-exit MSR-load address	
VM-EXIT INFORMATION FIELDS				
Basic VM-Exit Information	Exit reason		Exit qualification	
	Guest-linear address		Guest-physical address	
VM Exits Due to Vectored Events		VM-exit interruption information		VM-exit interruption error code
VM Exits That Occur During Event Delivery		IDT-vectoring information		IDT-vectoring error code
VM Exits Due to Instruction Execution	VM-exit instruction length		VM-exit instruction information	
	I/O RCX	I/O RSI	I/O RDI	I/O RIP
	VM-instruction error field			

- Natural-Width fields.
- 16-bits fields.
- 32-bits fields.
- 64-bits fields.

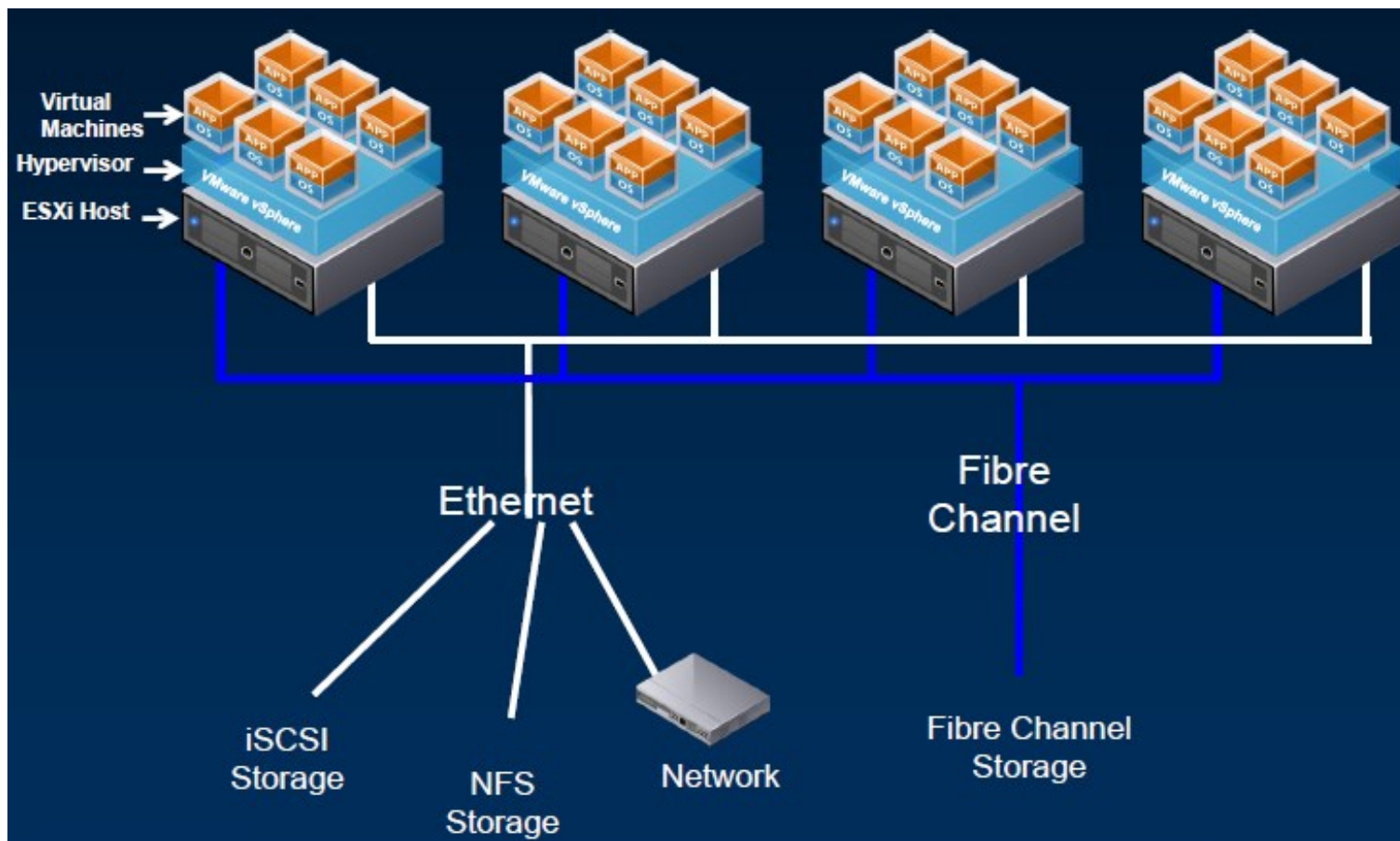
Copyright 2017, @Noteworthy (Intel Manual of July 2017)

# Физическа инфраструктура



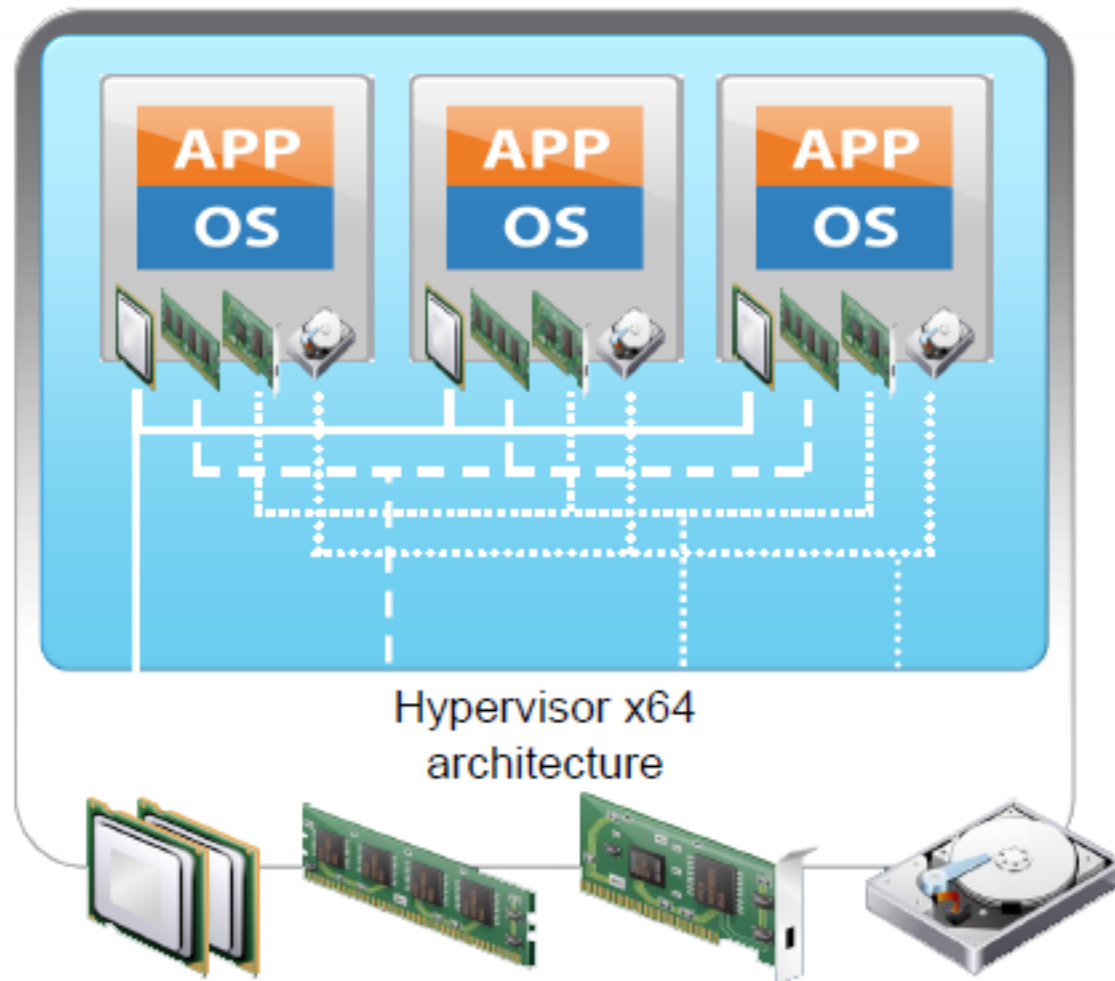


# Виртуална инфраструктура



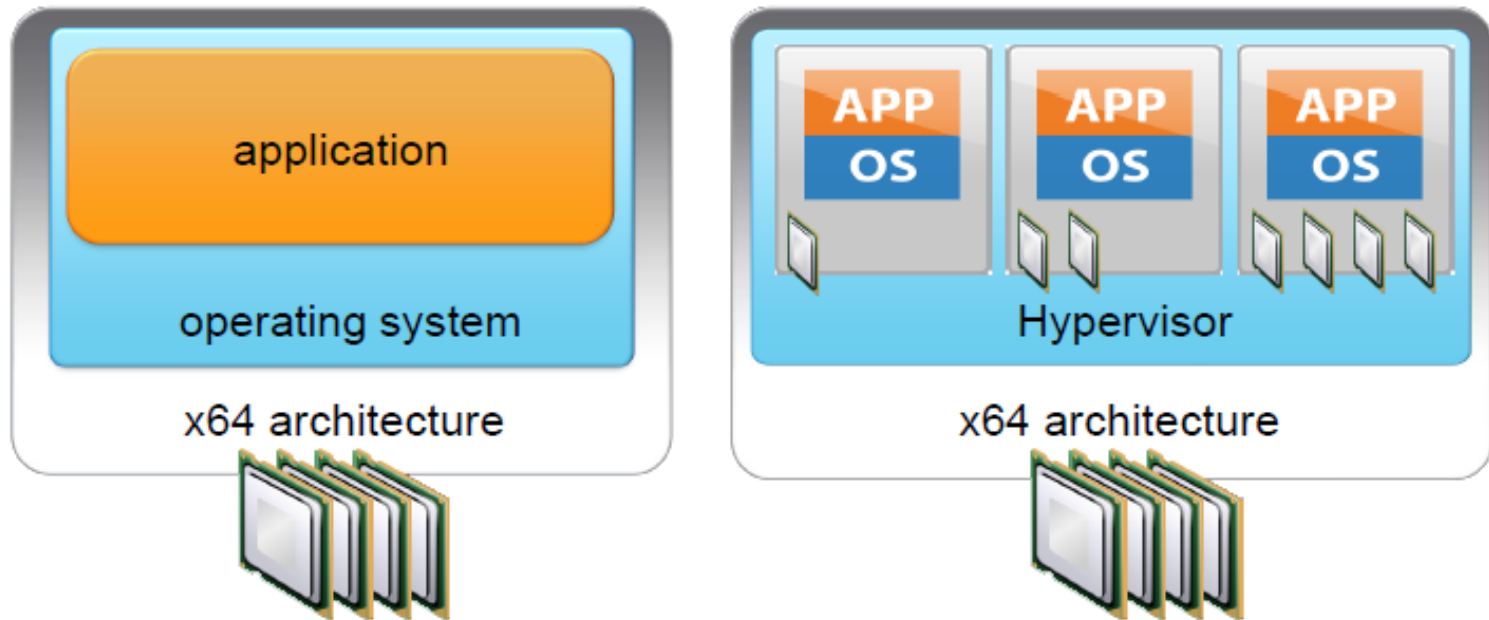


# Споделяне на ресурси

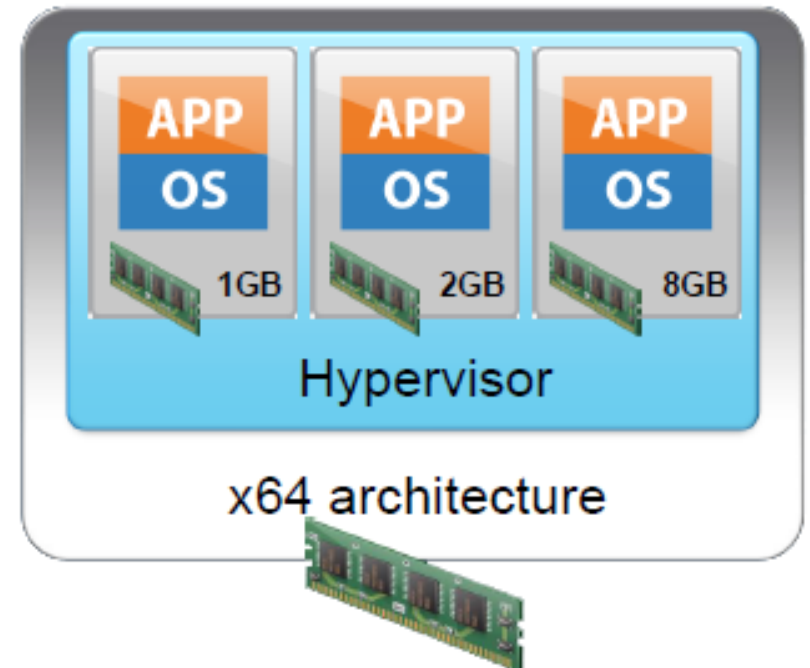
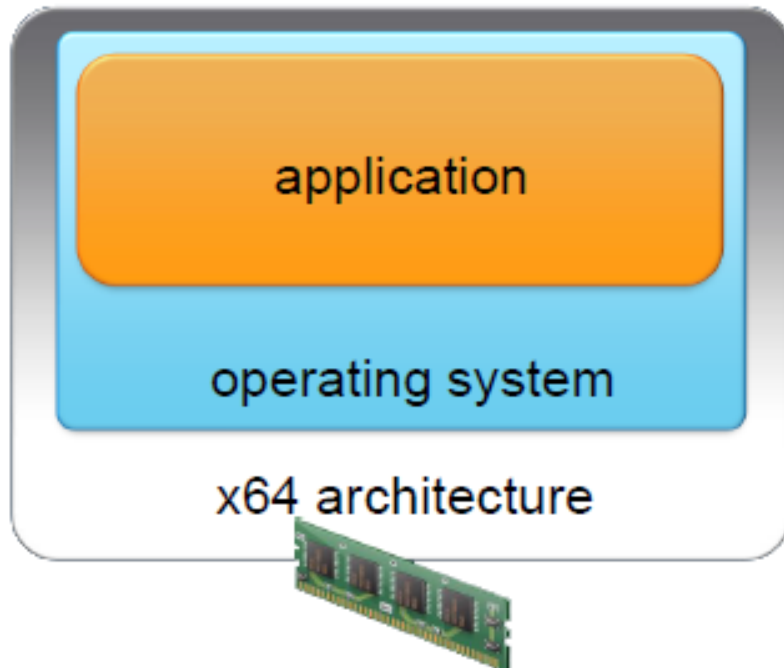


# Виртуализация на процесор

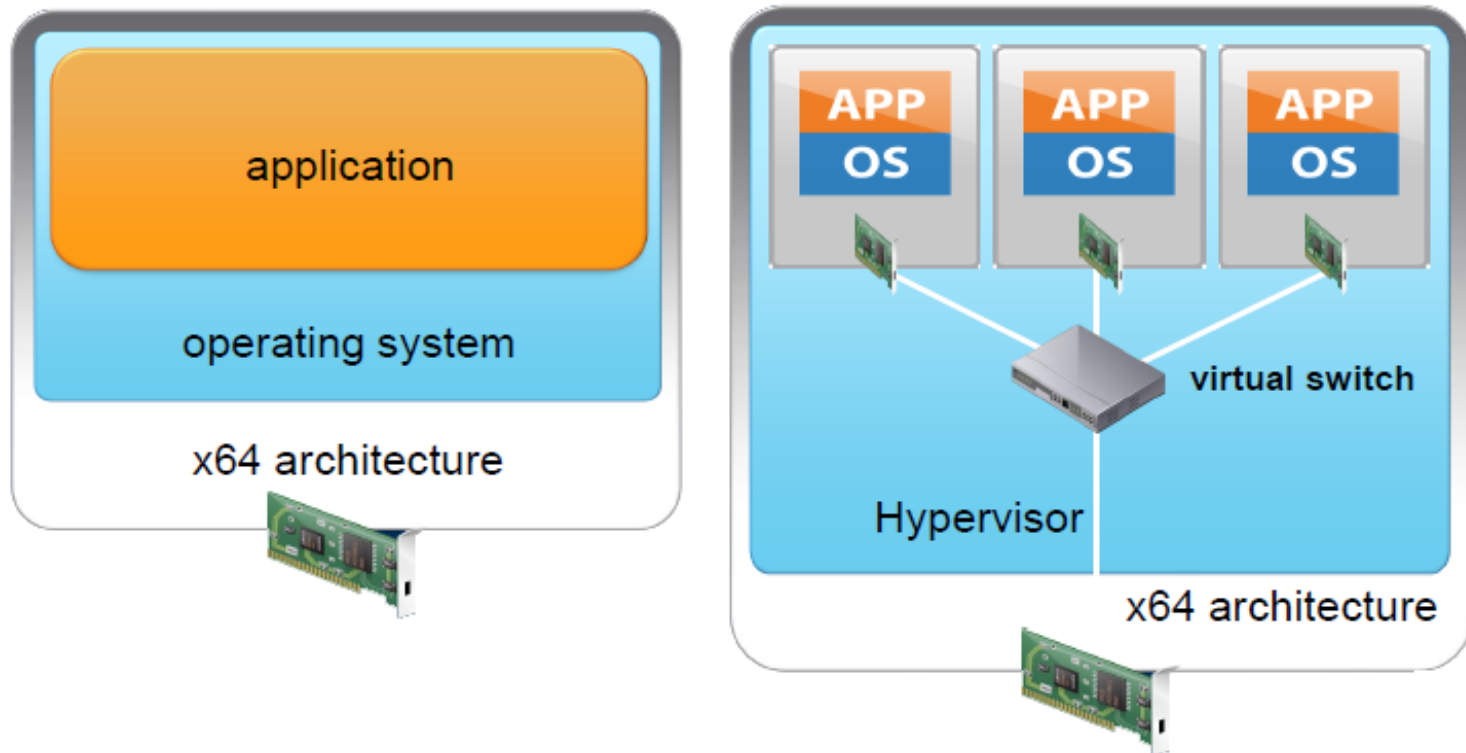
---



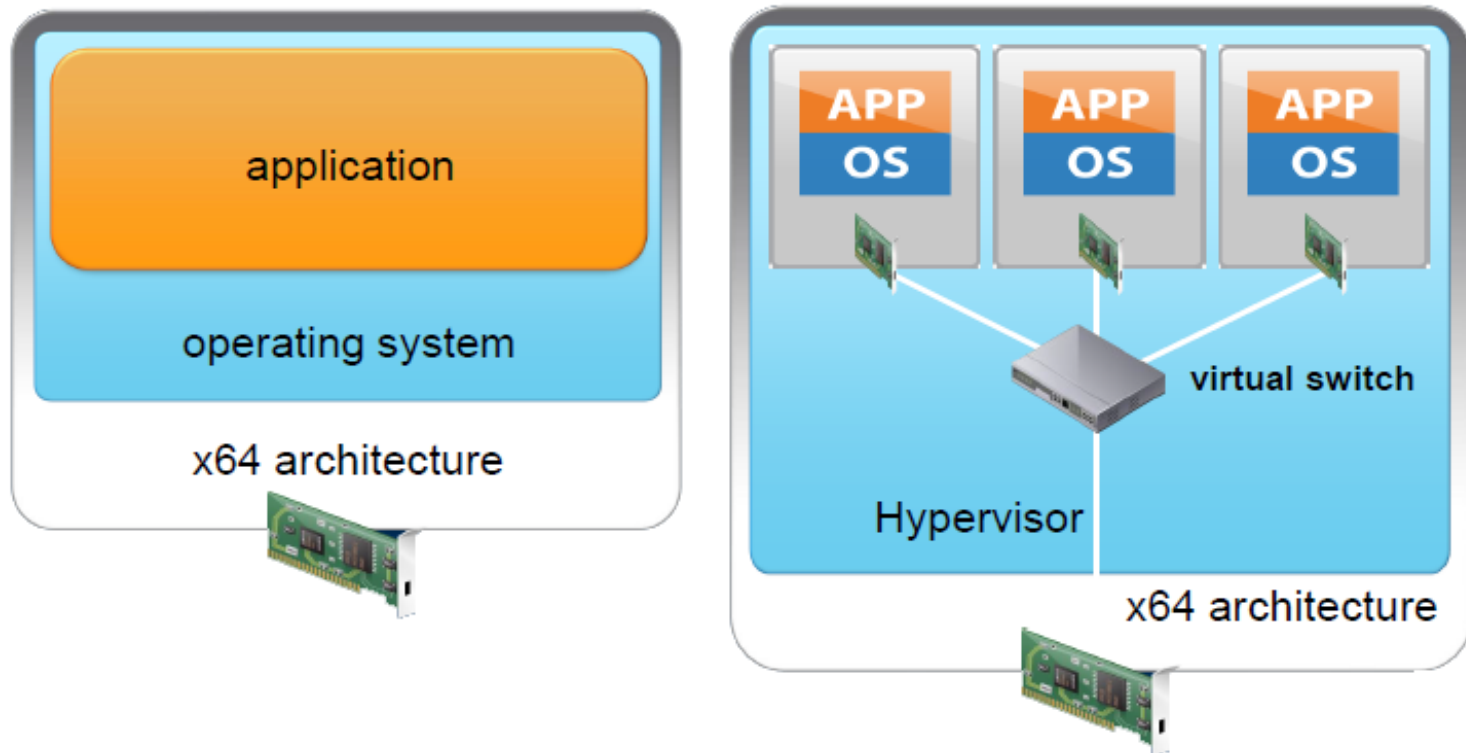
# Използване на паметта



# Физически и виртуални мрежи



# Физически и виртуални мрежи

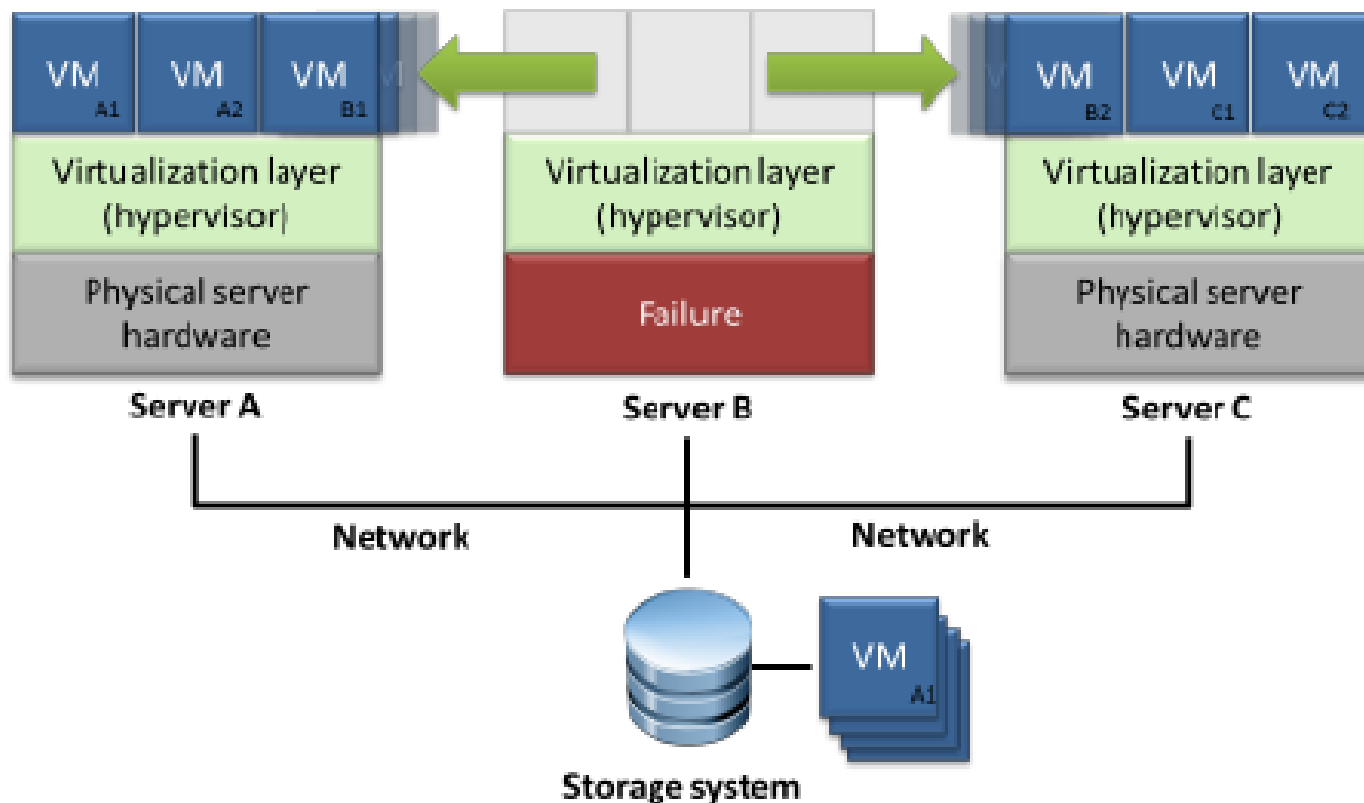


# Възможности при сървърна виртуализация

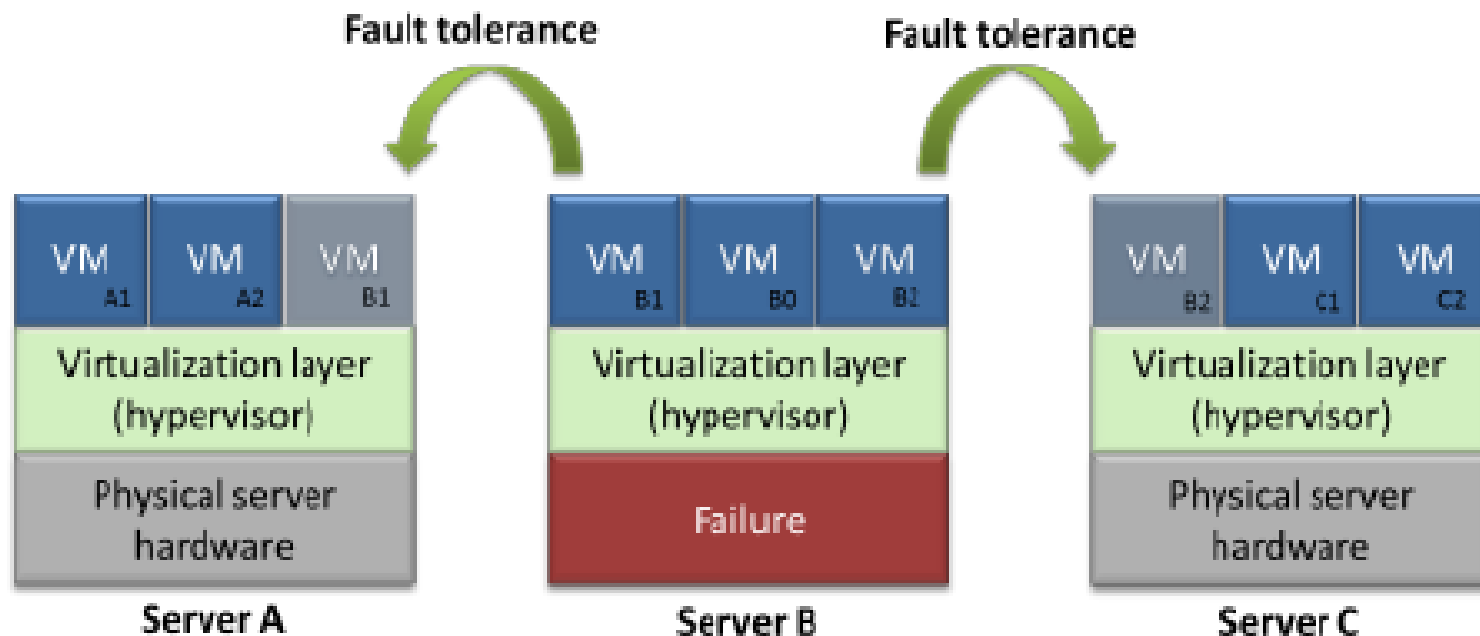
---

- High Availability (HA)
- Fault Tolerance (FT)
- Live Migration (LM)

# High Availability (HA)



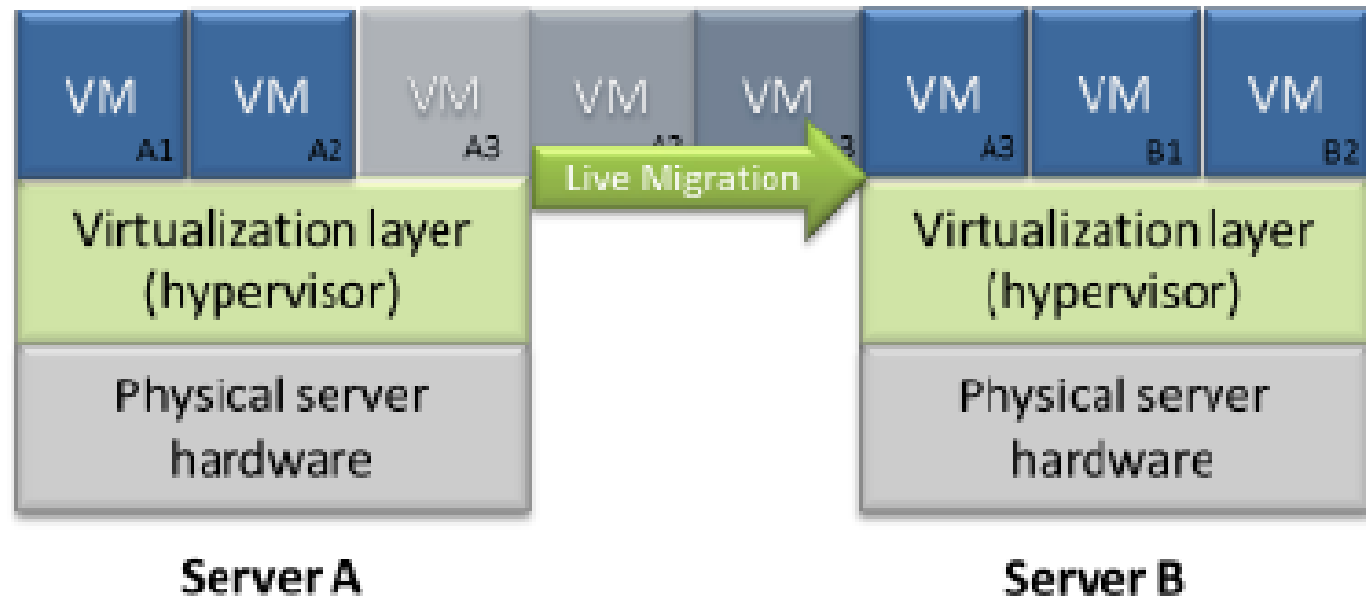
# Fault Tolerance (FT)





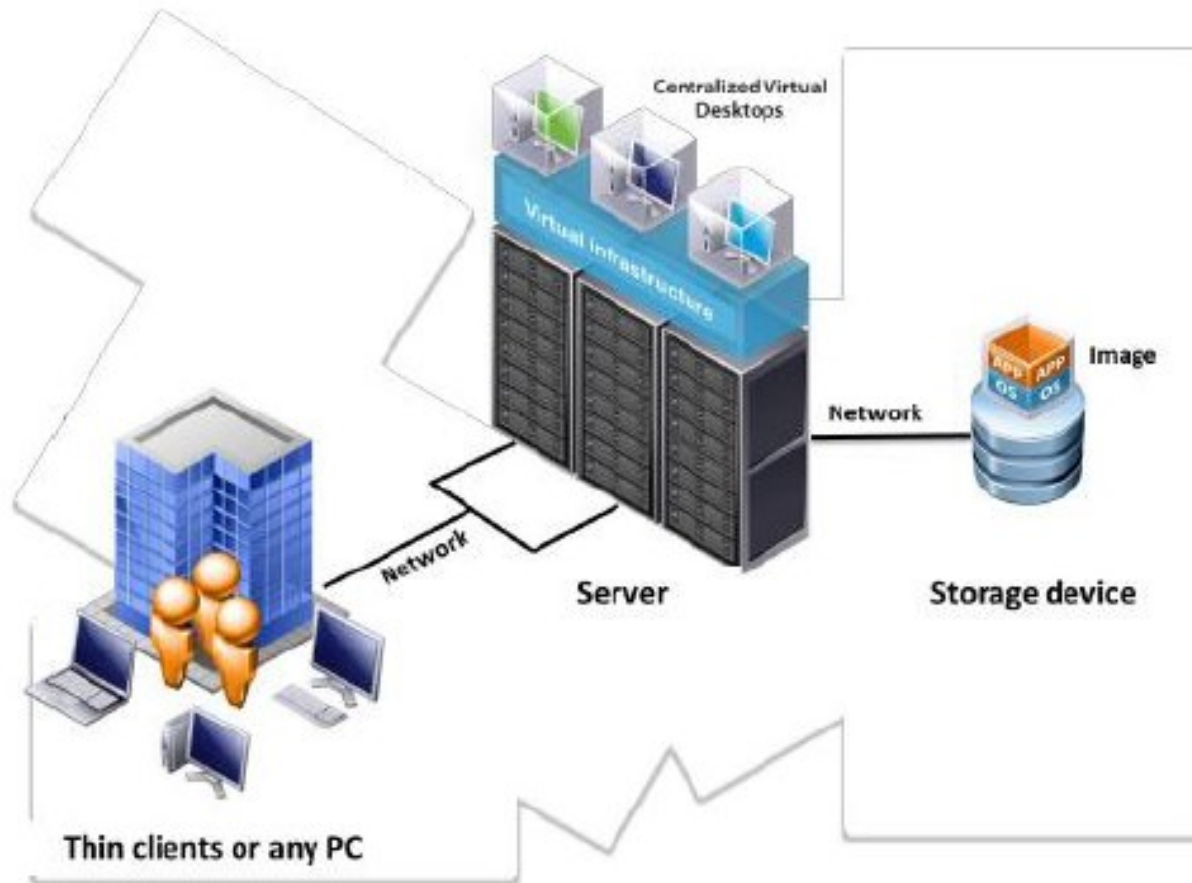
# Live Migration (LM)

---



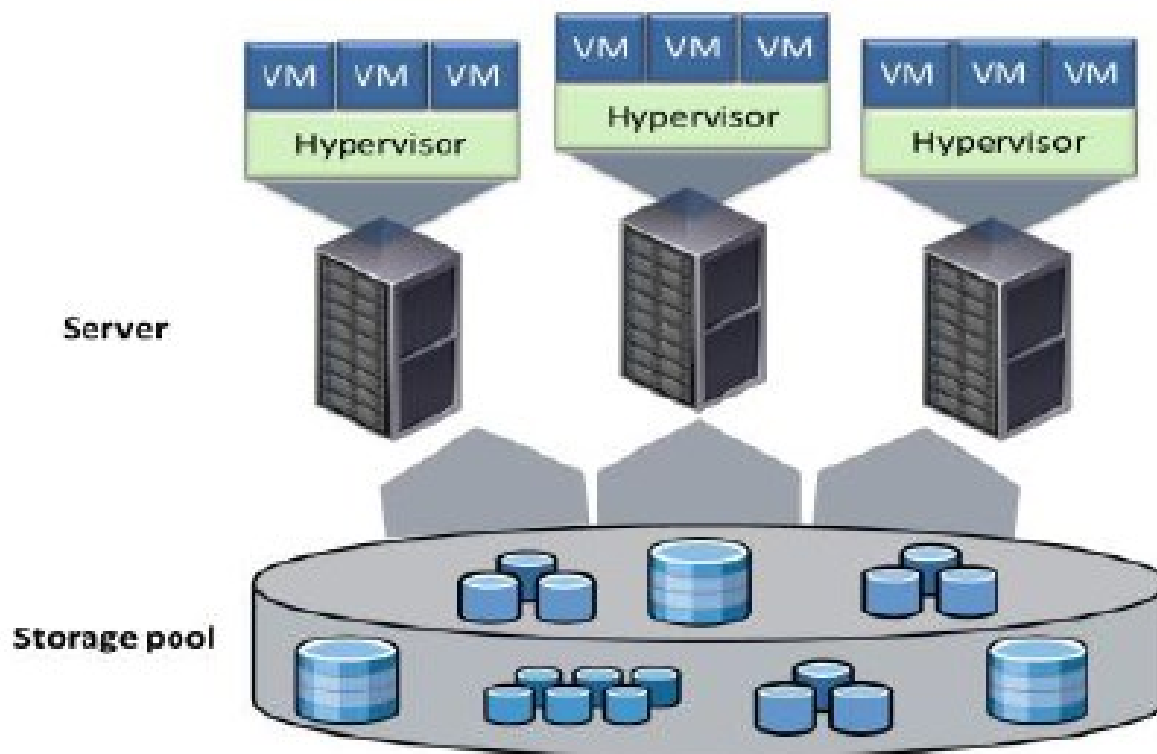
# Десктоп виртуализация

## ➤ Virtual Desktop Infrastructure (VDI)



# Сторидж виртуализация

---



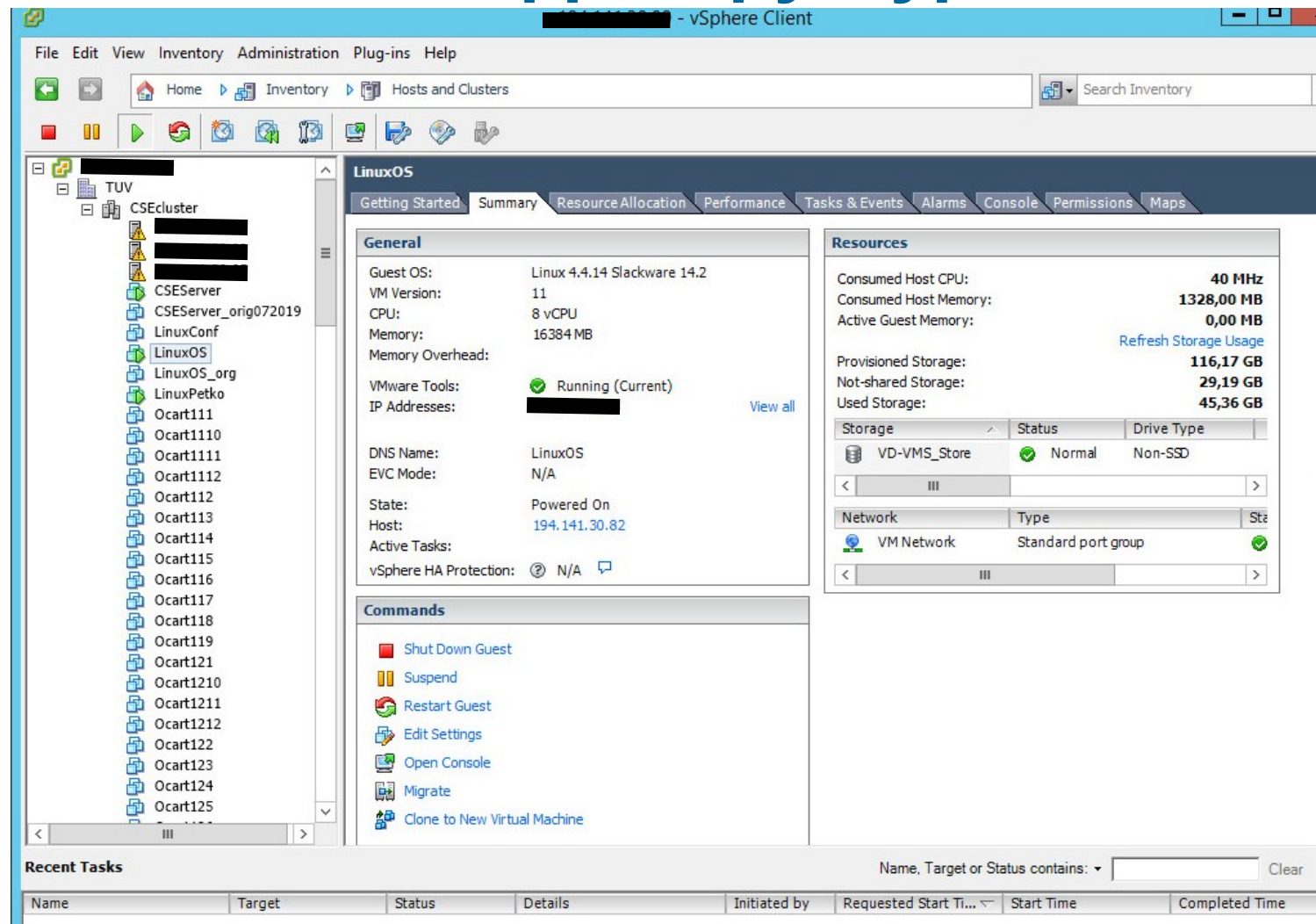
# КНТ клъстер

---

HP Proliant Gen8  
сървъри и  
виртуална VMware  
инфраструктура.



# Управление на виртуална инфраструктура

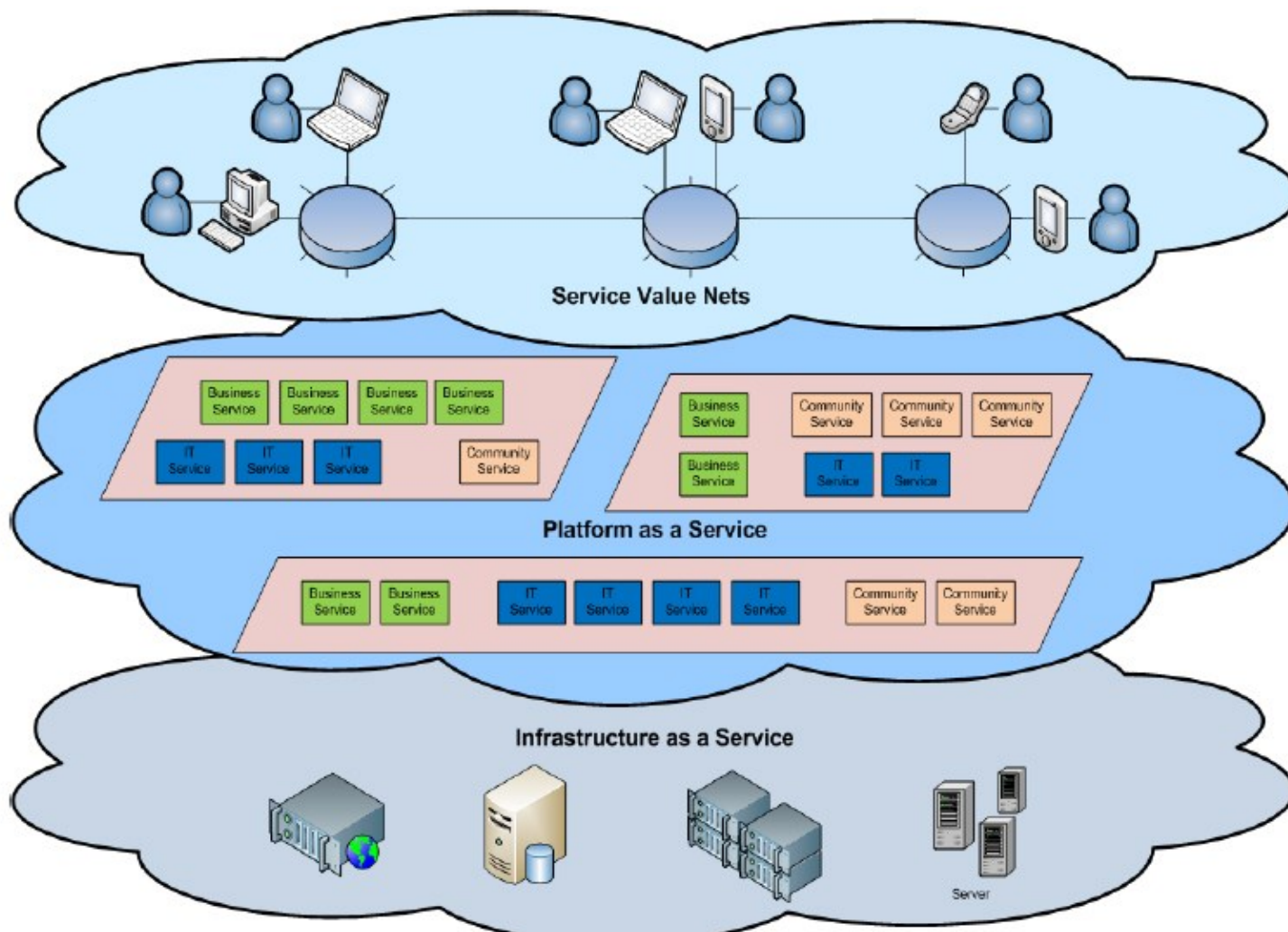


# Cloud computing

---

Модел за предоставяне на удобен мрежови достъп до споделена група от изчислителни ресурси (мрежи, сървъри, приложения), които бързо да могат да се заемат и освобождават (дефиниция на NIST).

# Архитектура на cloud computing



# Общи характеристики

---

- Възможност за масивно разширяване;
- Хомогенност;
- Географско разпределение;
- Виртуализация;
- Ниска цена на софтуера;
- Ориентация на услуги;
- Висока сигурност



# Service Level Agreement

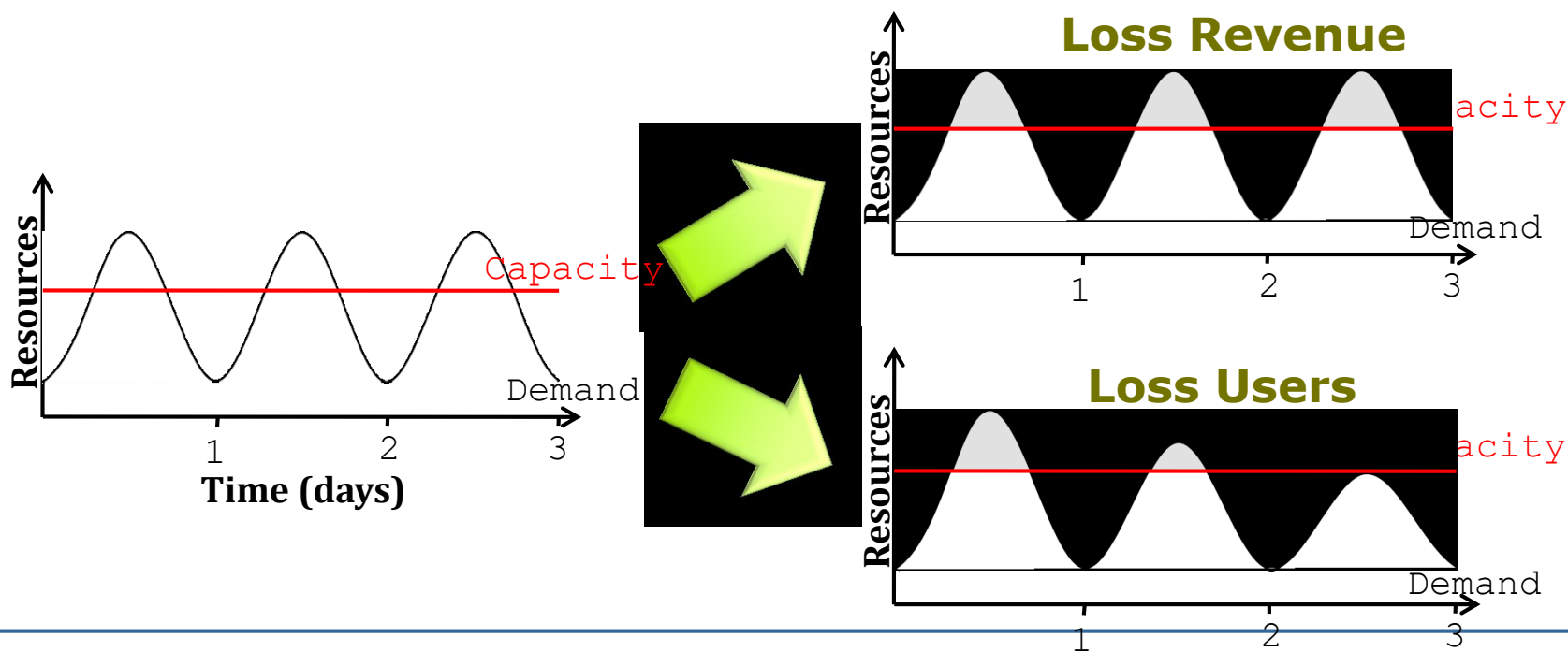
---

- SLA е договор между доставчика и клиента който описва в измервателни термини, какви мрежови услуги доставчикът ще предостави.
- Метрики, гарантиращи производителността:
  - Up-time, down-time
  - Системна пропускателна способност
  - Време за отговор
- Детайли по отстраняване на проблеми
- Наказания за лоша производителност
- Документирани параметри на сигурността

# Динамично обезпечване

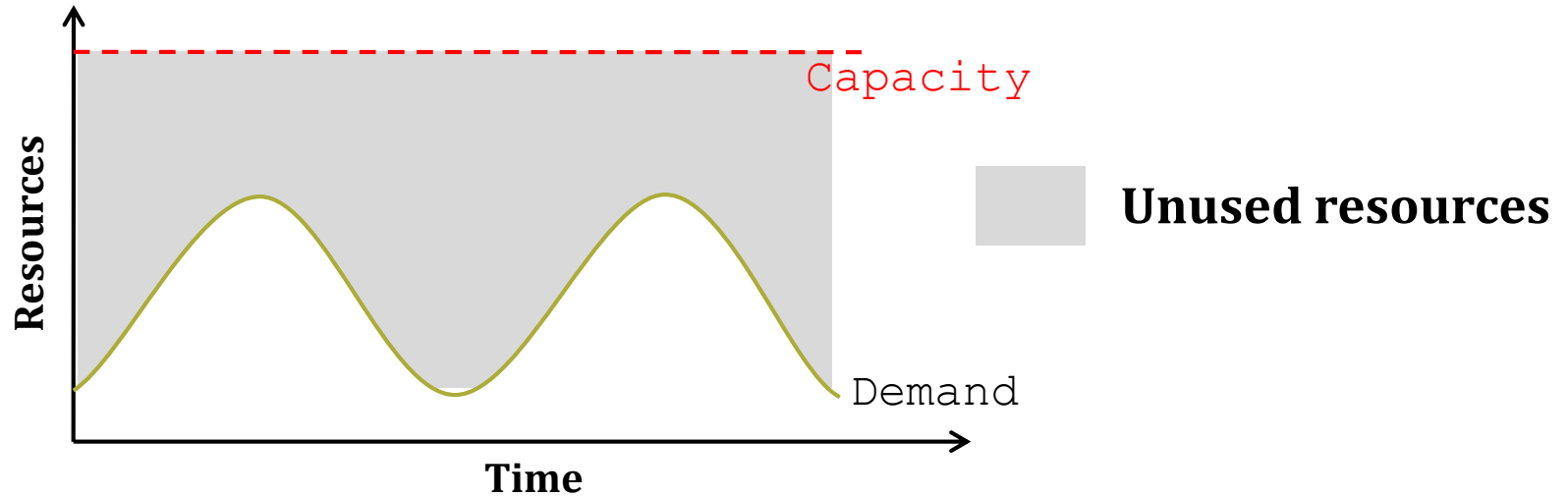
При традиционния модел на изчисления съществуват два общи проблема, свързани с използването на системните ресурси:

- **Подценяване на използването на системата, което води до недостатъчно осигуряване**



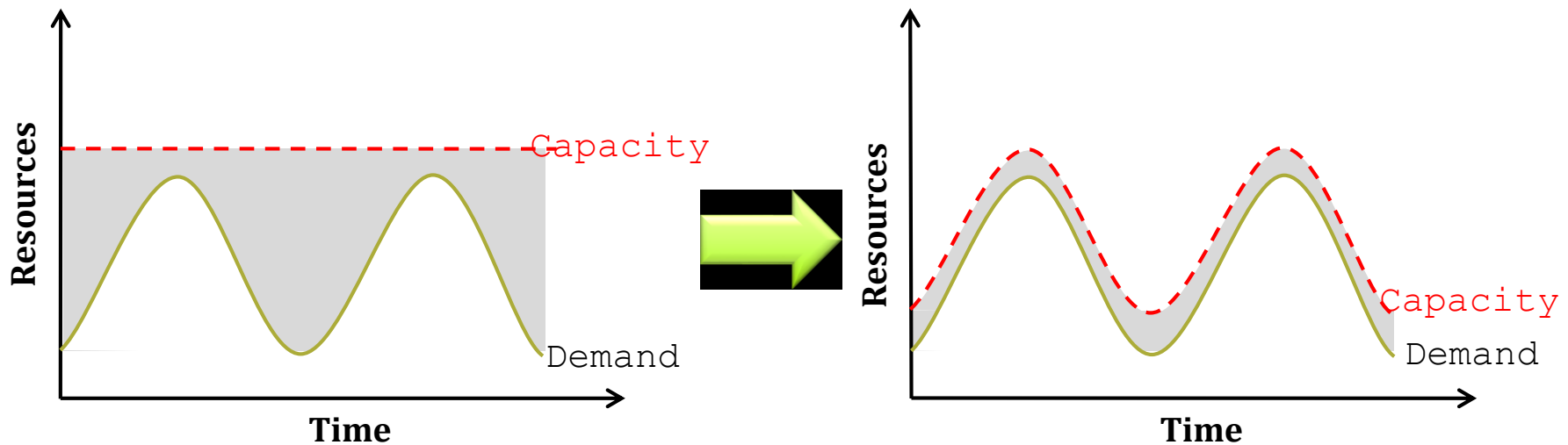
# Динамично обезпечване

- Надценяване на използването на системата, което води до ниско използване

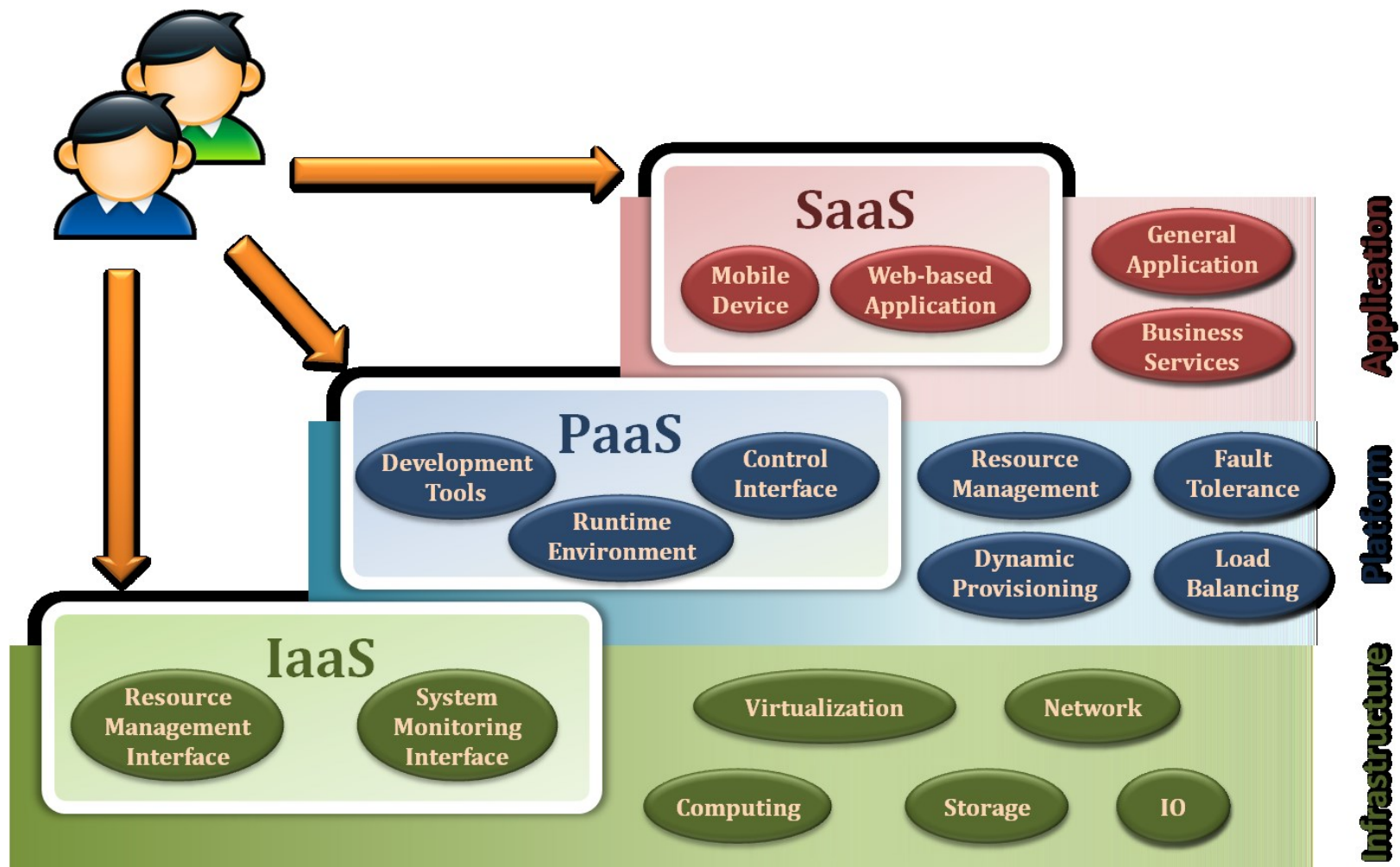


# Динамично обезпечване

Решение: динамично обезпечване на ресурси



# Cloud Service Модели



# Infrastructure as a Service

---



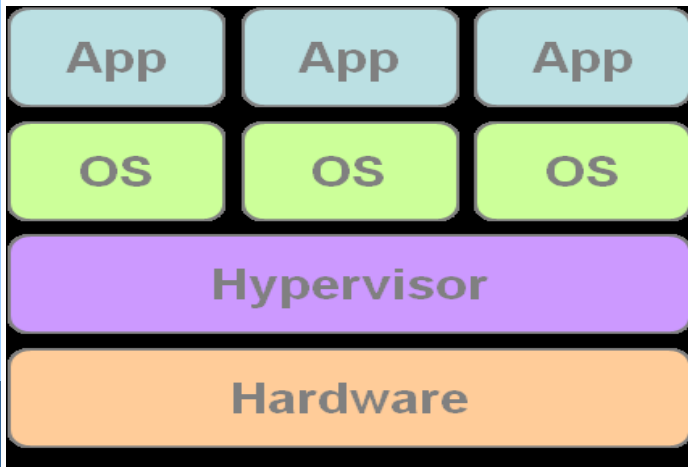
Модел на доставяне, при който организацията наема външно оборудване използвано за осигуряване на различни дейности (памет, хардуер, сървъри и мрежови компоненти). Доставчикът притежава оборудването и е отговорен за неговата поддръжка. Клиентът обикновено заплаща за използване на това оборудване.

- Amazon EC2
- Eucalyptus
- OpenNebula

# Infrastructure as a Service

---

## Реализация на IaaS: Виртуализация



- Абстракция на изпълнима среда, която може динамично да се предоставя на клиенти;
- Реализира се на базата на виртуални машини (VM);
- Хипервайзор емулира инструкциите от VM и управлява хардуерните ресурси динамично и прозрачно.

# Platform as a Service

---

Начин за наемане на операционни системи, памет и мрежови капацитет през Интернет. Моделът на доставка позволява потребителят да наема виртуализирани сървъри за изпълнение на приложения или за тяхното разработване и тестване.

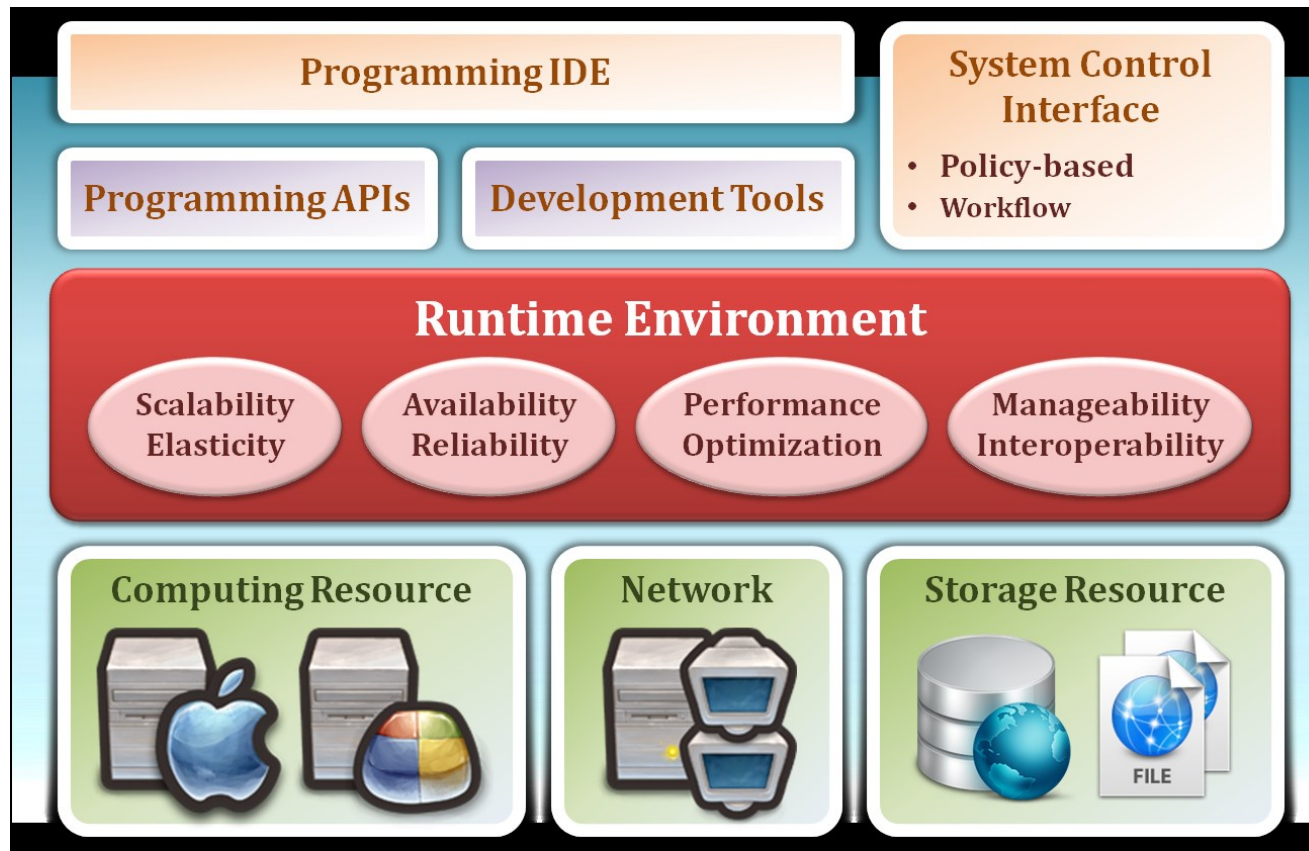
- Microsoft Windows Azure
- Google App Engine
- Hadoop





# Platform as a Service

## Системна архитектура



# Platform as a Service

---

## Реализация на PaaS: **Runtime Environment Design**

- Колекция от налични софтуерни услуги. Типично е формирана като съвкупност от програмни библиотеки.
- Потребителите могат да използват IDE за разработване на приложения.

# Software as a Service

---

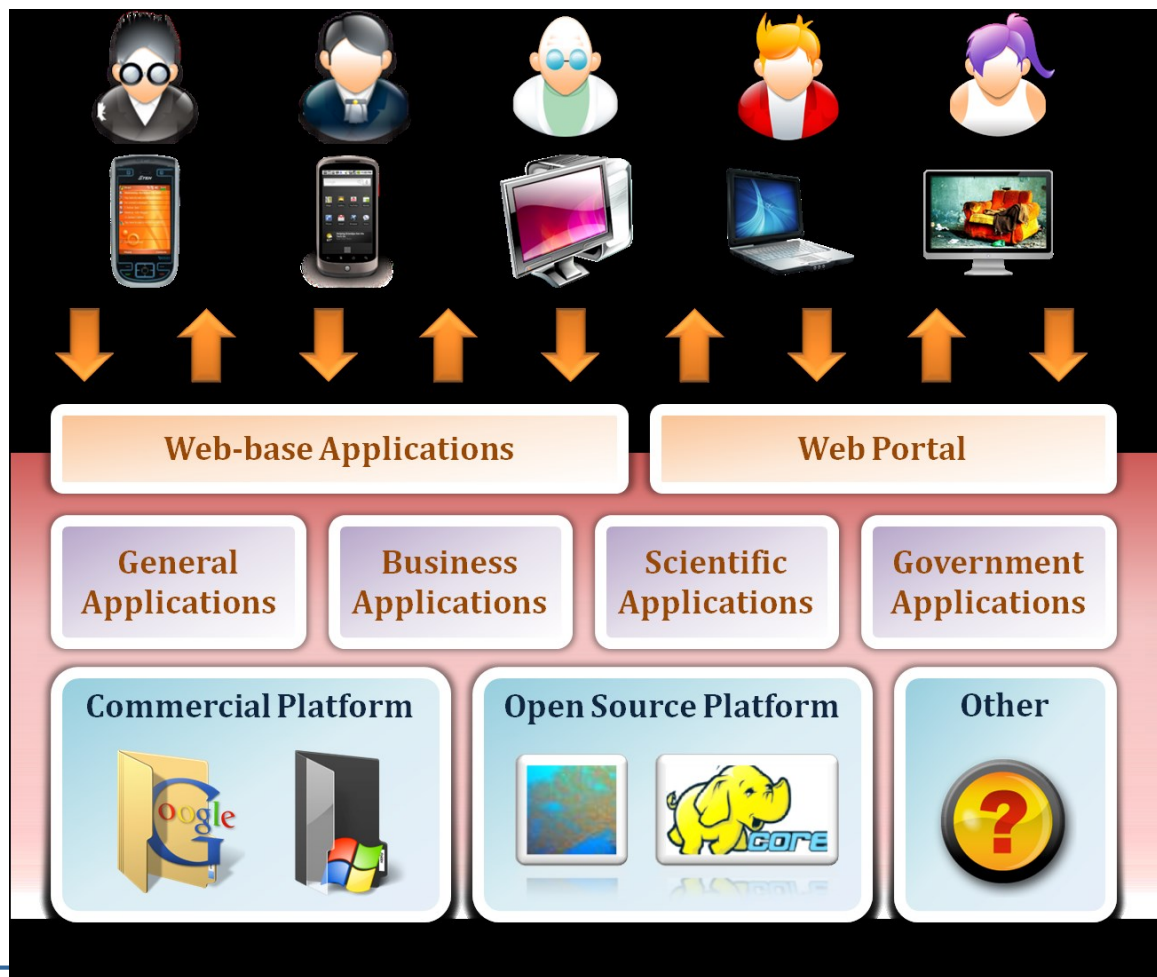
Софтуерен разпределен модел, при който приложенията се поддържат от доставчик и са достъпни за потребителите през мрежови достъп, типично Интернет.

- Google Apps (Gmail, Google Docs, Google sites)
- SalesForce.com
- EyeOS



# Software as a Service

## Системна архитектура



# Software as a Service

---

## Реализация на SaaS: **Web services**

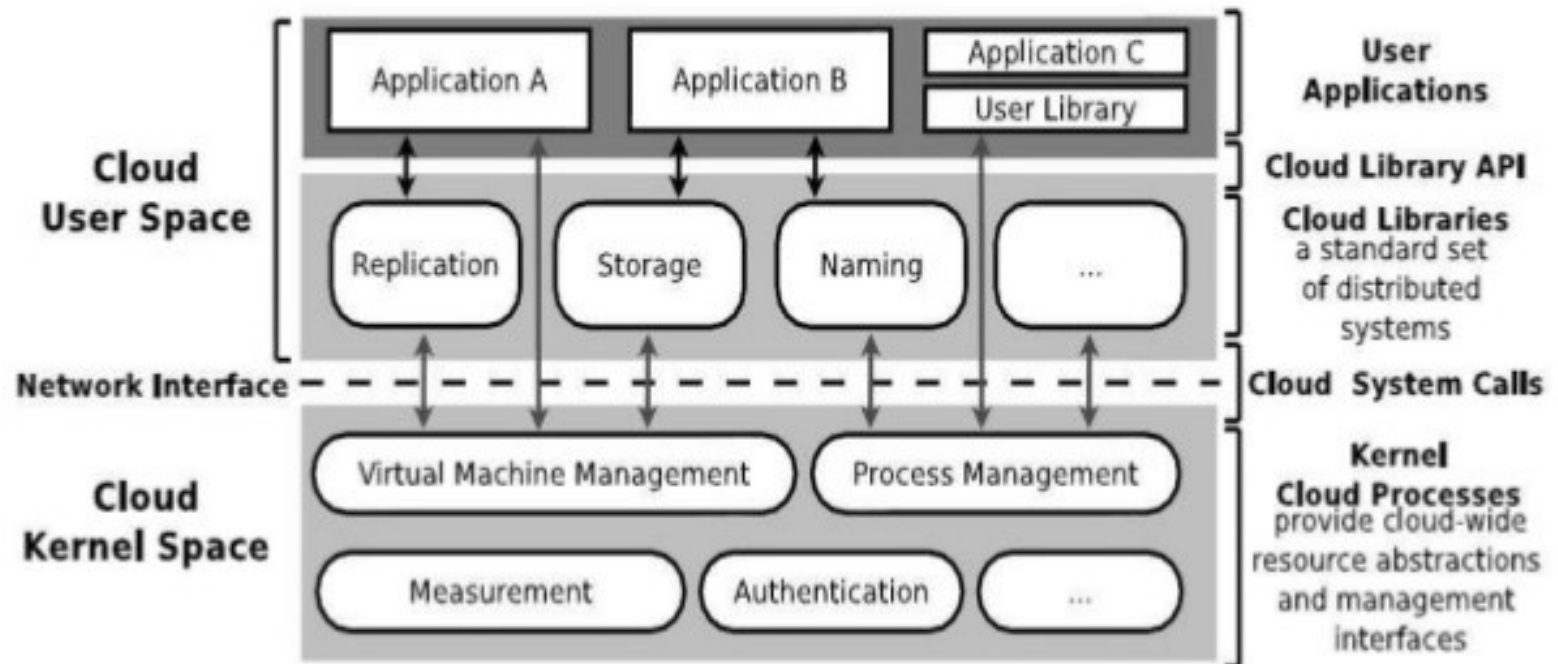
- Web 2.0 – тенденцията в използването на WWW.
- Използване на различни типове web-приложения.
- Web-портали, предоставящи услуги като e-mail, борсови цени, бази данни, развлечения:
  - iGoogle
  - MSNBC
  - Netvibes
  - Yahoo!

# Облачни операционни системи

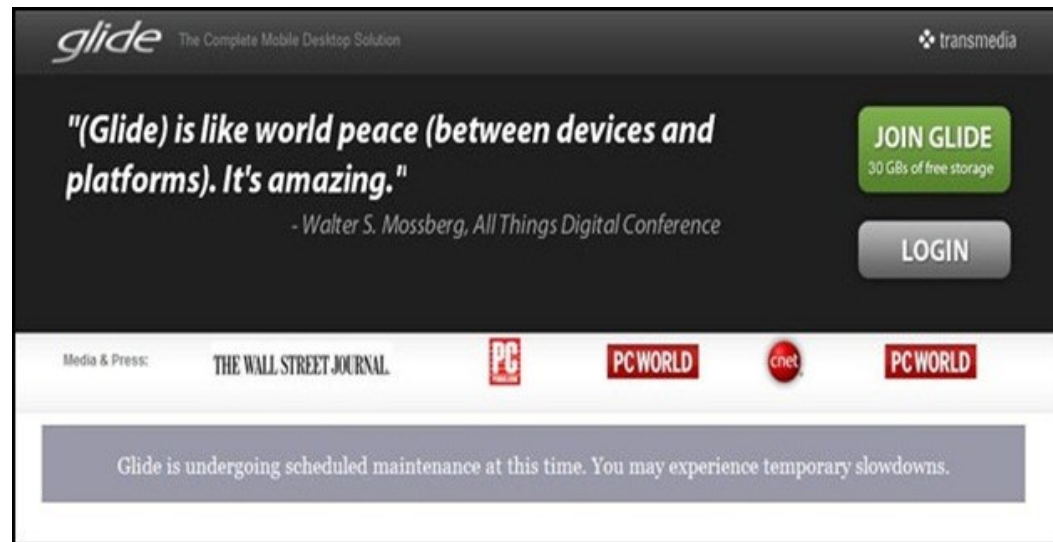
---

Cloud OS управлява операциите, изпълнението и процесите на виртуални машини, виртуални сървъри и виртуални инфраструктури.

# Облачни операционни системи

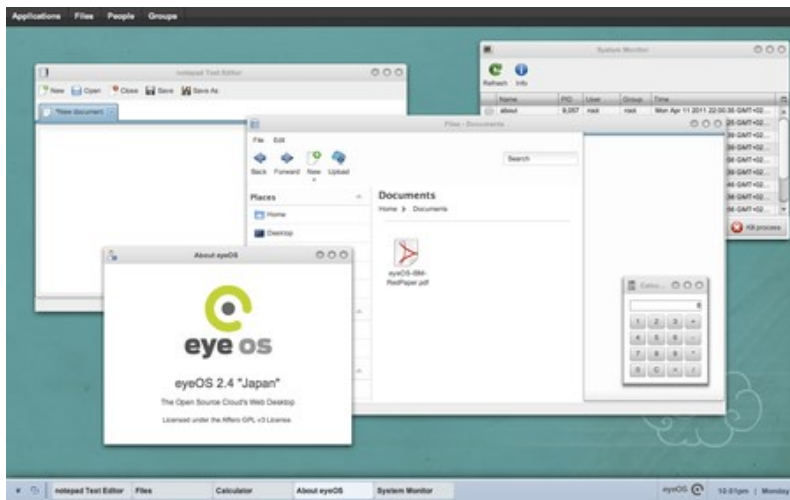


# Облачни операционни системи





# Облачни операционни системи



**Въпроси?**