



Същност на Виртуализацията

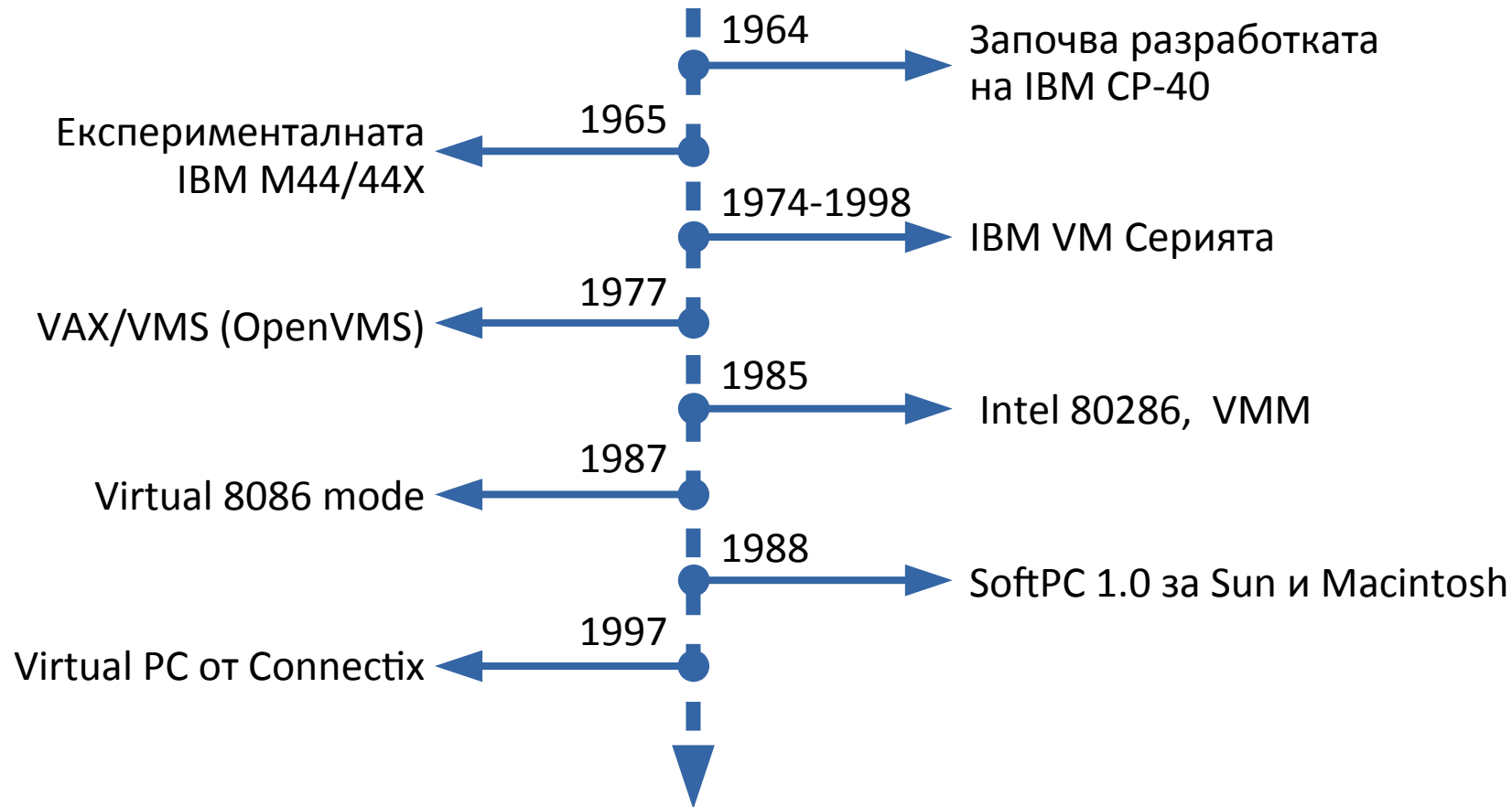
Въведение
Виртуализация

Съдържание

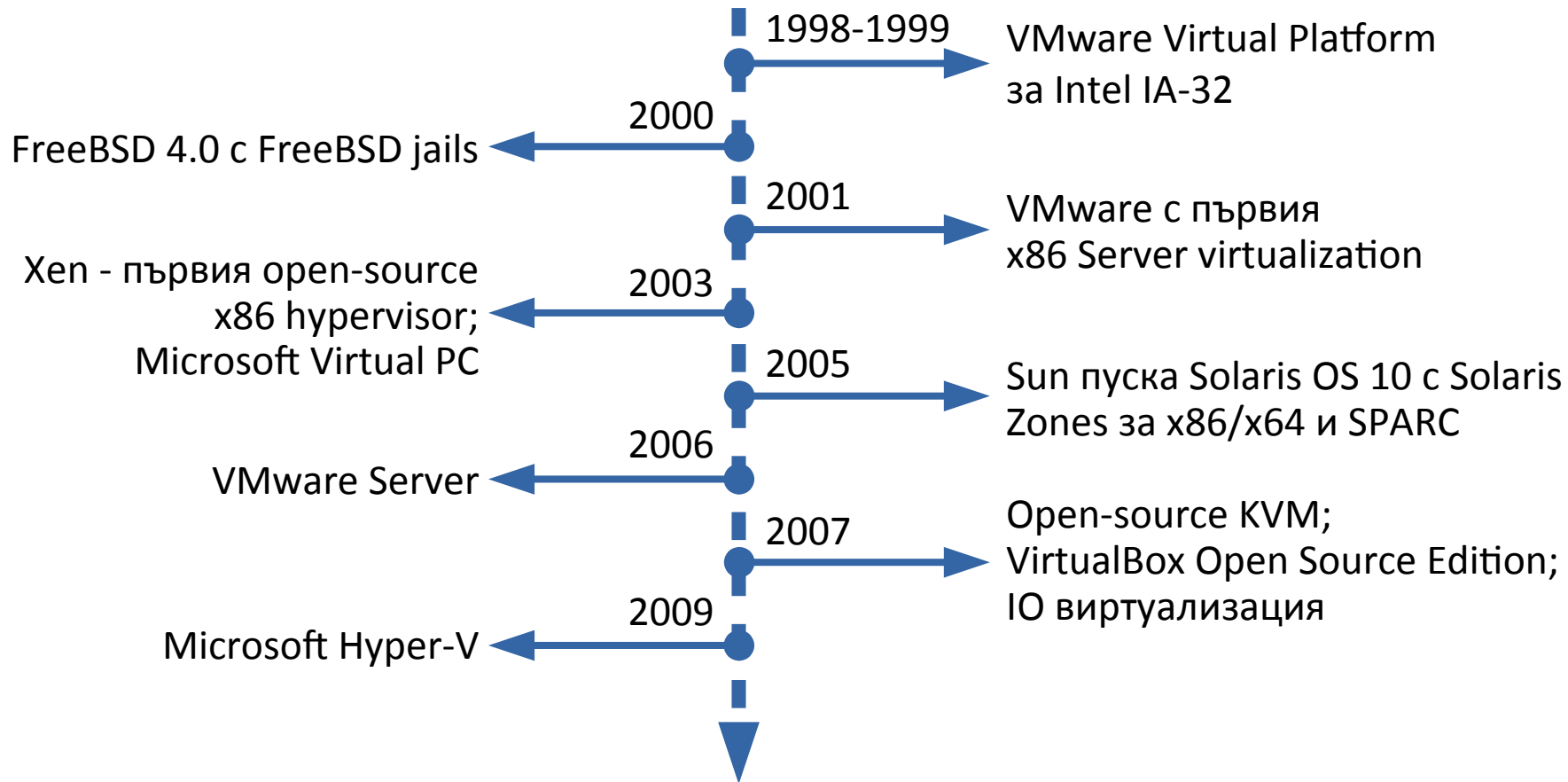
1. Кратка история
2. Видове виртуализация. Процесна/Application и Системна (OS и Machine-Level виртуализация)
3. Хипервайзори. Тип 1 (Bare-metal) и Тип 2 (Hosted) хипервайзори
4. Системи за виртуализация
5. Приложение. Предимства и недостатъци

История

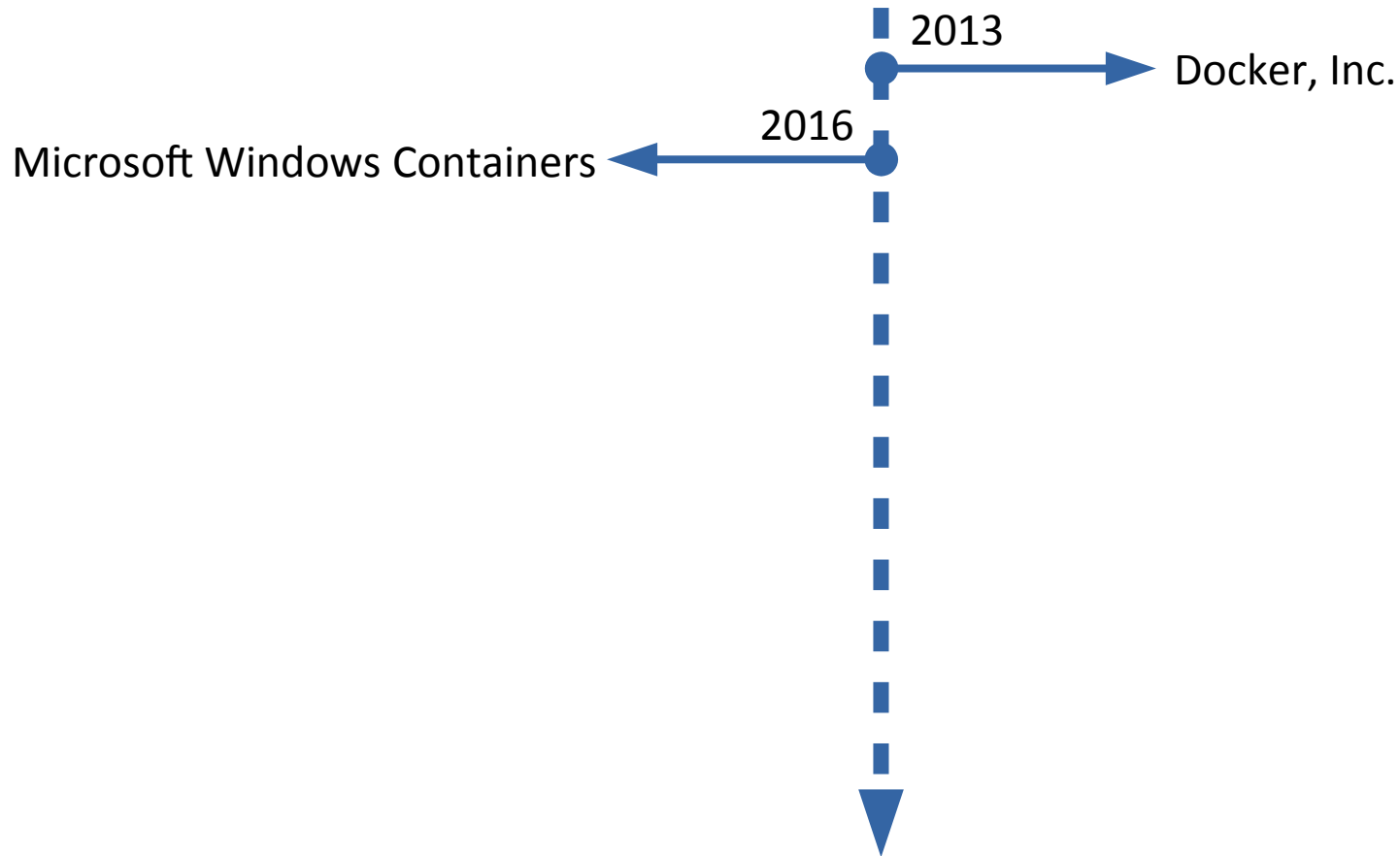
Кратка история



Кратка история

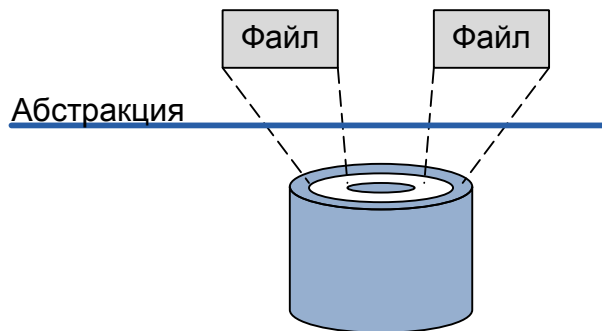


Кратка история



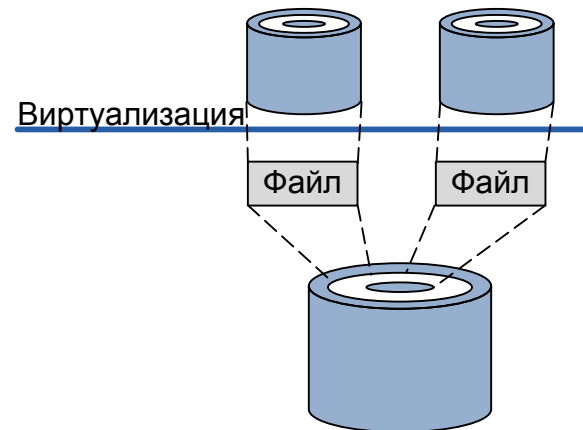
Пример

Абстракция и Виртуализация



Абстракция

- ❖ ОС “скрива” детайлите за адресиране в хард диска т.е. скрива това, че той е съставен от сектори и пътечки
- ❖ За приложния софтуер диска изглежда като множество от файлове

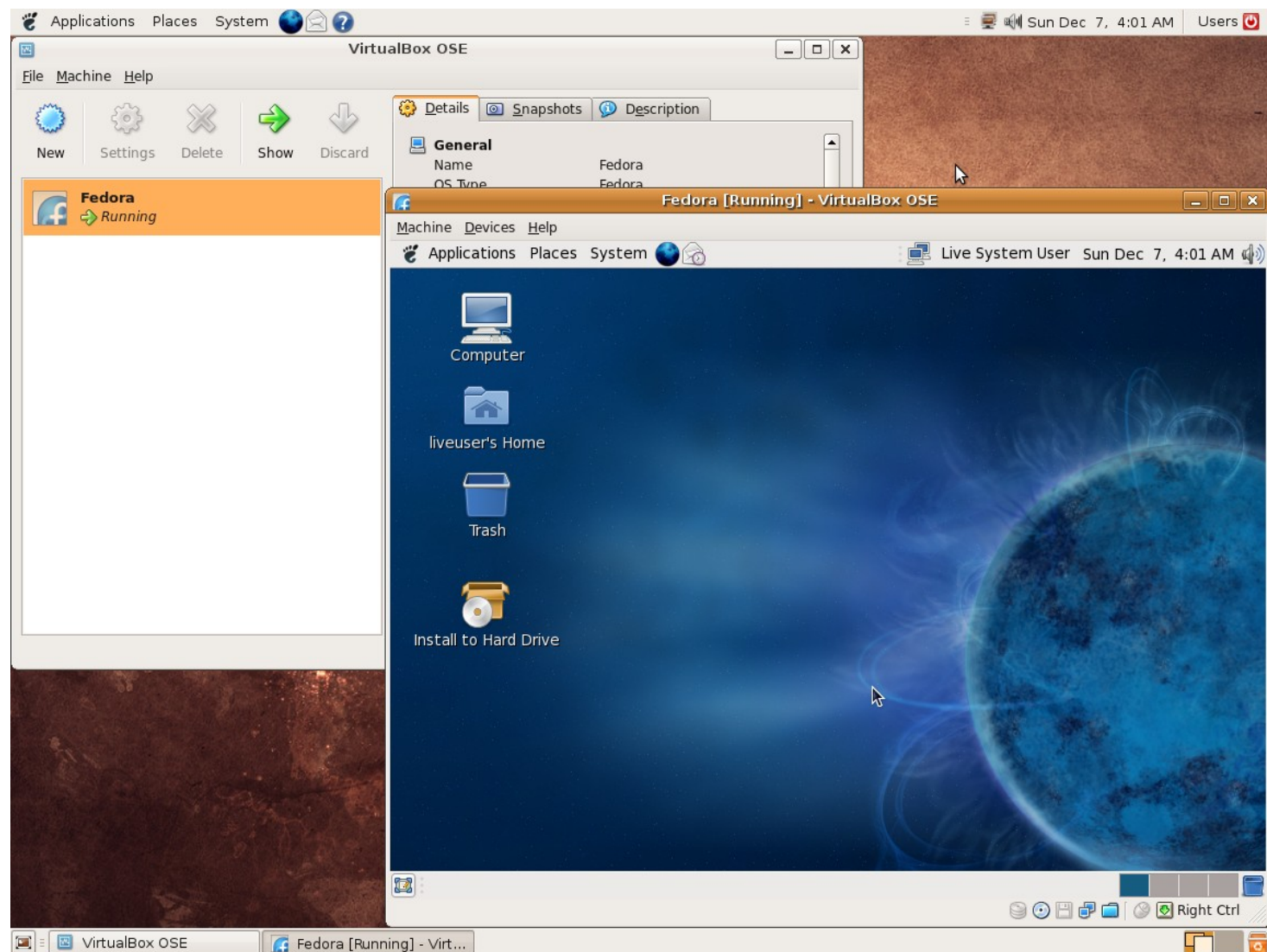


Виртуализация

- ❖ Преобразува един физически диск в два по-малки виртуални диска, всеки от които изглежда, че има свои собствени сектори и пътечки
- ❖ Не е задължително да скрива или опростява детайли

Пример

Fedora Linux стартиран
във виртуална
машина под
управлението на
Ubuntu Linux и
VirtualBox



Пример

Windows 7 и OS X Lion
стартирани във
виртуални машини
под управлението на
OS X Sierra и Parallels



Виртуализацията е рамка (framework) или методология за разделяне на ресурсите на компютър в множество среди за изпълнение, чрез прилагане на една или повече концепции или технологии, като хардуер и софтуерно разделяне, споделяне на време, частична или пълна машинна симулация, емуляция, качество на услуги и много други.

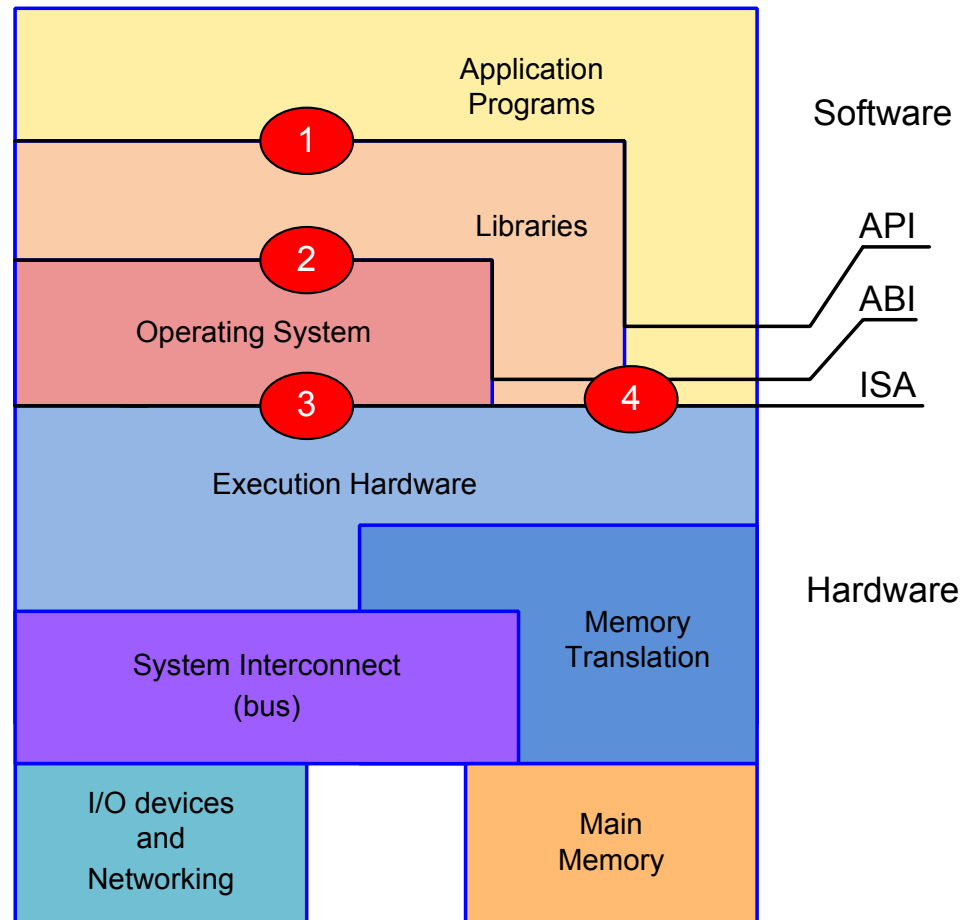
Видове Виртуални Машини

(Процесни/Application, ОС-Level и Системни)

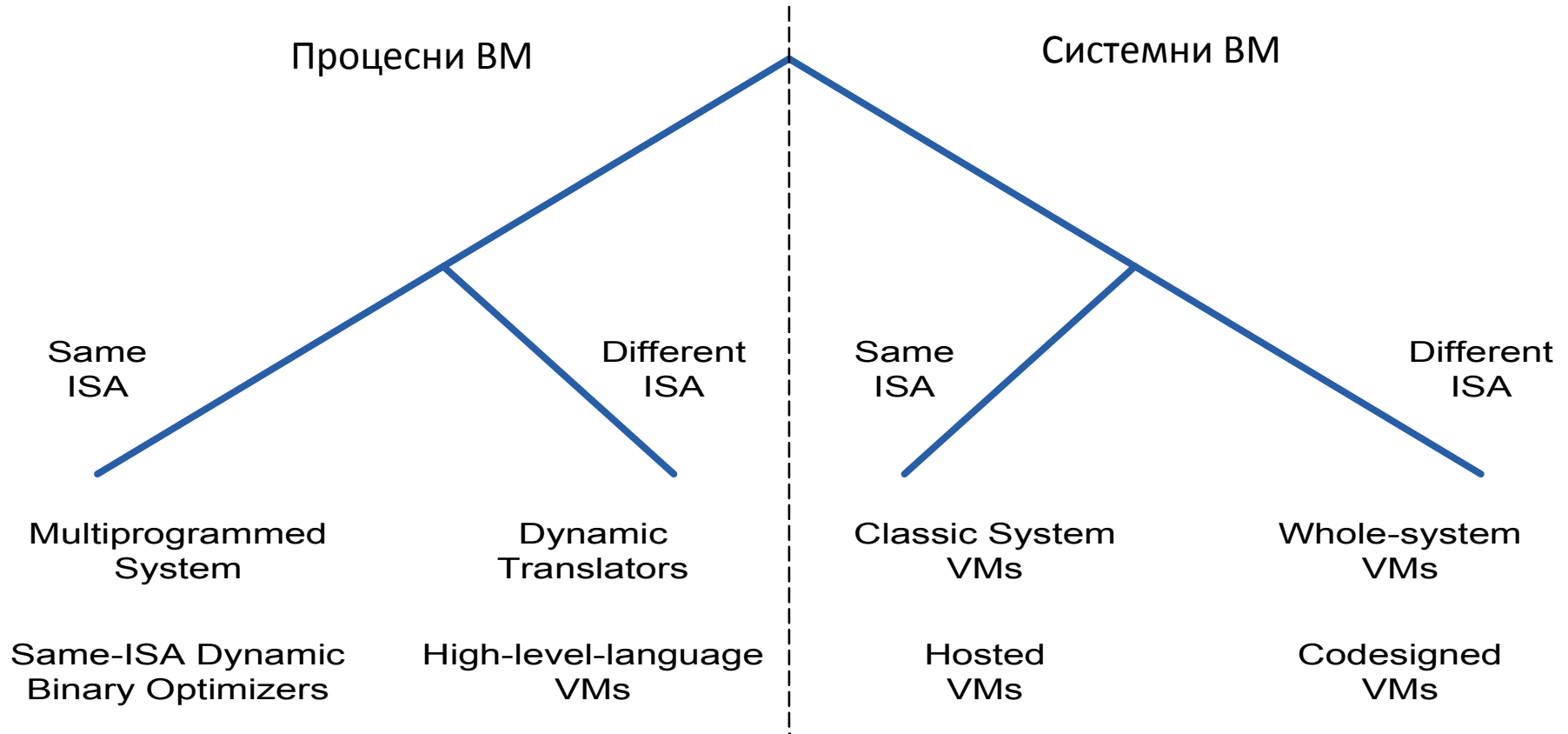
Виртуални машини. Принципи

Архитектурни интерфейси, близо до границата хардуер-софтуер

- ❖ **Instruction set** – маркира границата между хардуер и софтуер (❸ и ❹). ❹ представлява потребителския набор от инструкции. ❸ са системният набор от инструкции (супер-множество на 4);
- ❖ **Application Binary Interface (ABI)** – дава достъп на програмите до хардуерните ресурси – интерфейс ❹ или чрез системни извиквания – интерфейс ❷;
- ❖ **Application Programming Interface (API)** – дава достъп до хардуерните ресурси и услуги – интерфейс ❹, допълнени с извиквания на библиотеки от езици за програмиране от високо ниво. Използването му позволява лесното пренасяне на програмите между платформите чрез рекомпилация;



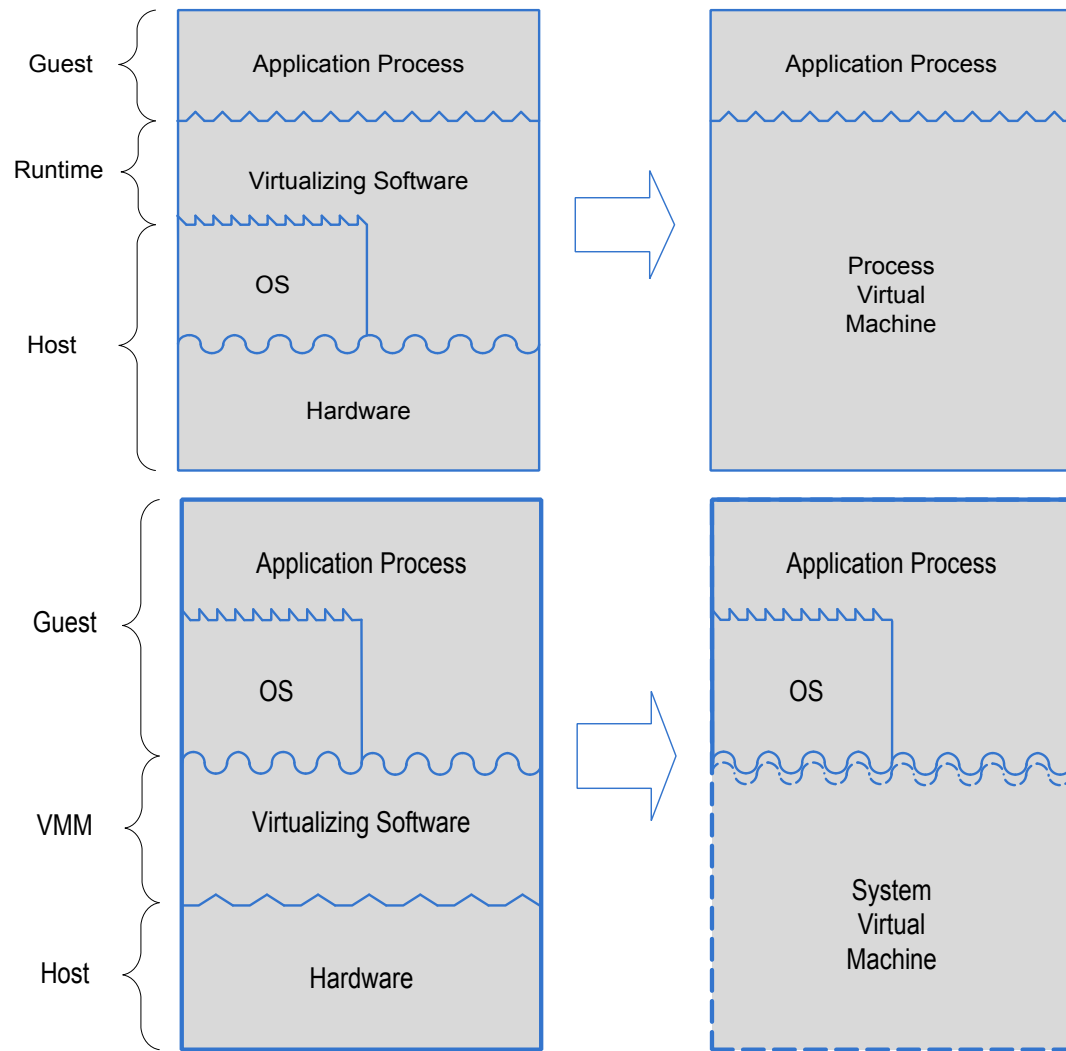
Класификация на ВМ



Процесни и Системни ВМ

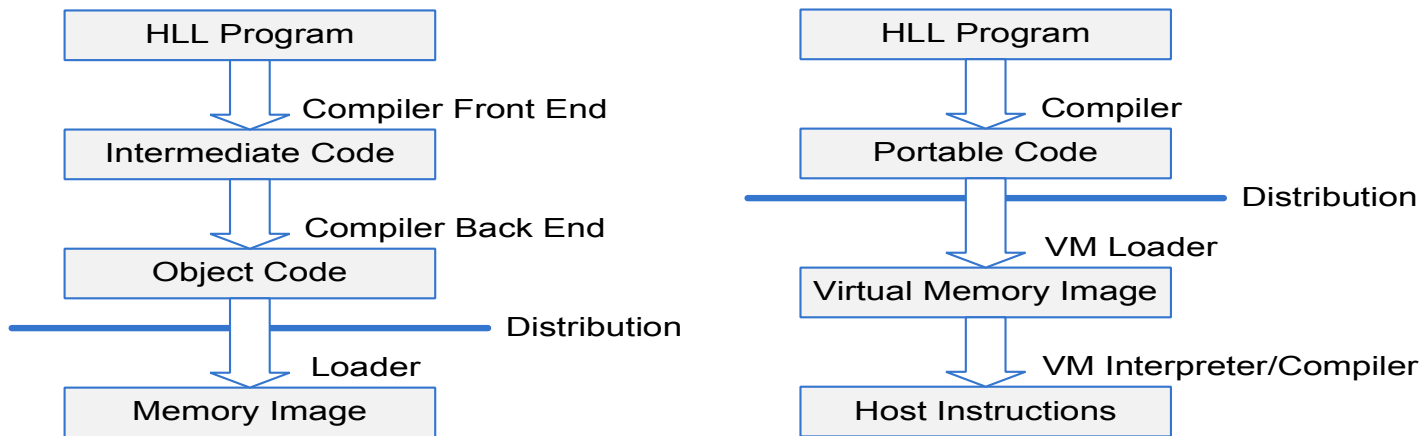
Основни разлики

- ❖ При **процесните ВМ**, виртуализираният софтуер е на ниво ABI или API и е върху комбинацията OS/HW. Runtime емулира едновременно потребителските инструкции и едно от извикванията на ОС или извикванията на библиотеки
- ❖ При **системните ВМ**, виртуализираният софтуер е между хардуера-хост и софтуера-гост. VMM (Virtual Machine Monitor или Hypervisor) емулира хардуерните ISA, така че софтуера-гост може да изпълнява различен ISA. На практика, обаче, VMM се използва само основно за предоставяне на виртуализация на хардуерните ресурси



Видове

- ❖ **Multiprogrammed systems** – съвременните ОС с време деление;
- ❖ **Emulators and dynamic binary translators** – Intel IA32-EL, който позволява IA-32 приложение да се изпълнява на Intel Itanium;
- ❖ **Same-ISA binary optimizers** – Dynamo, SolidOpt и др.;
- ❖ **High-Level-Language ВМ** – CLR, JVM, LLVM;



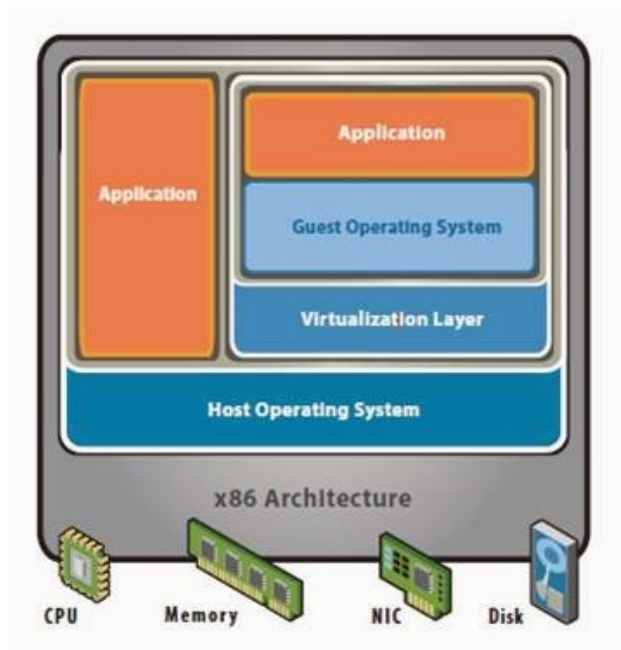
Видове

- ❖ **Класически системни ВМ** – Поставят VMM (хипервайзора) директно върху хардуера;
- ❖ **Хоствани системни ВМ** – Поставят VMM върху хост-ОС, която се изпълнява директно върху хардуера;
- ❖ **Whole-system VM** – VMM емулира различния набор инструкции хост инструкции;
- ❖ **Многопроцесорна виртуализация** – Физически или логически разделят големи ресурси на по-малки като използват специализиран софтуер;
- ❖ **Codesigned VM** – Реализират или свой собствен набор от инструкции или разширяват вече съществуващ. VMM е разположен на скрито място в паметта и служи за бинарна транслация и кеширане на инструкциите;

Хипервайзори

(Тип 1 – Bare-metal и Тип2 – Hosted)

Хипервайзор (Hypervisor) или VMM (Virtual Machine Monitor) се нарича компютърен софтуер, фирмуер или хардуер, който създава и стартира Виртуални машини.



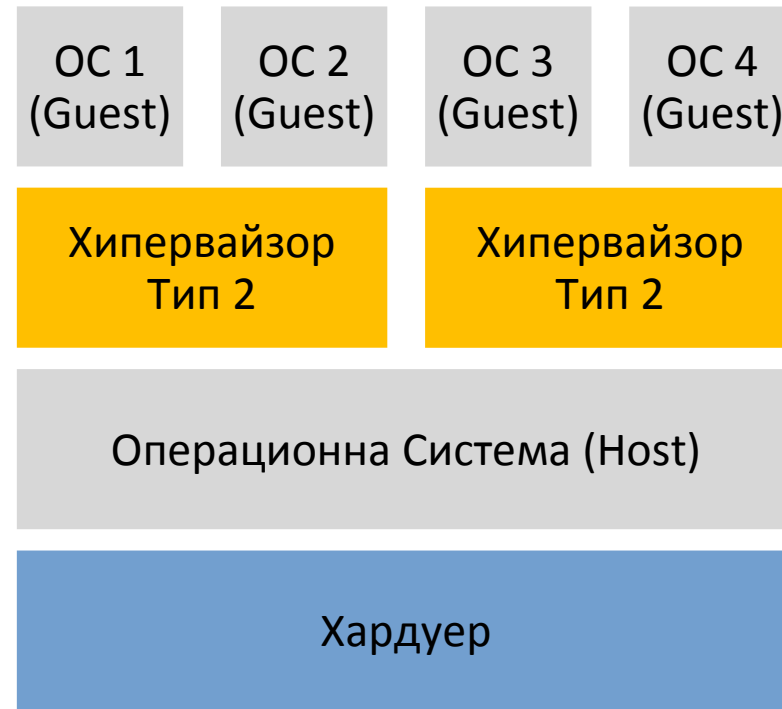
Host машина (Машина-Домакин) се нарича компютъра на който хипервайзора стартира една или повече виртуални машини, наречени **Guest машини** (Машини-Гости)

Видове

- ❖ **Type 1 – Bare-metal/Native** – Поставят хипервайзора (VMM) директно върху хардуера;
- ❖ **Type 2 – Hosted** – Поставят хипервайзора (VMM) върху хост-ОС, която е се изпълнява директно върху хардуера;

В този курс ще разглеждаме основно Type2 виртуални машини.

Хипервайзори (Hypervisors)



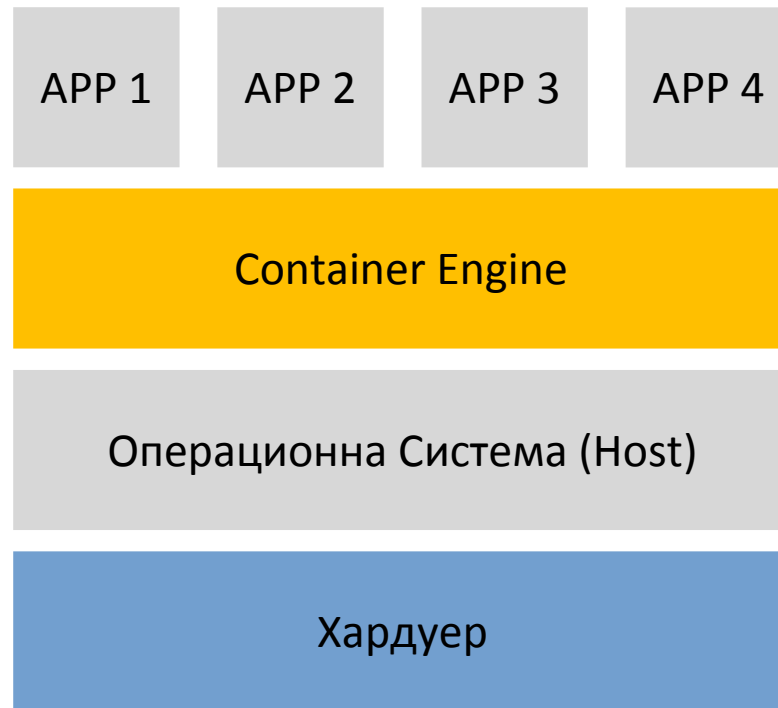
OS-Level Виртуализация

(Контейнери)

Контейнери

Видове

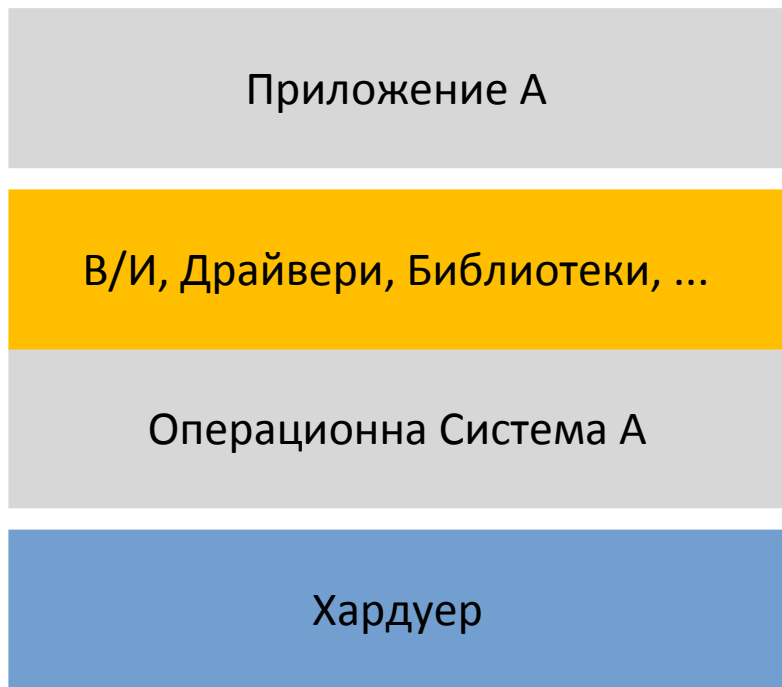
- ❖ Контейнери;
- ❖ Зони;
- ❖ Chroot / jail;



В този курс ще разглеждаме основно Контейнери.

Application-Level Виртуализация

Application-Level Виртуализация



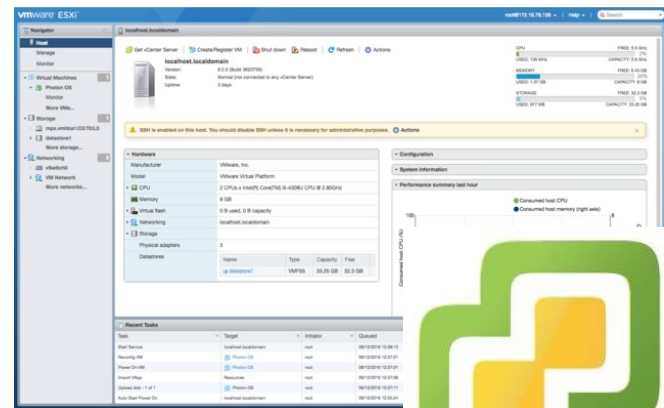
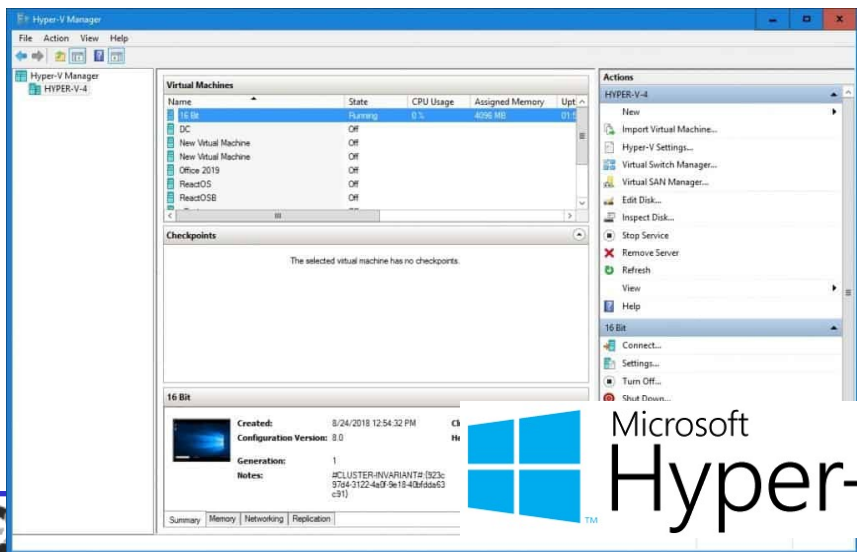
Application-Level Виртуализация

- ❖ Application-Level Виртуализация (Процесна виртуализация) – софтуерна технология за капсулиране/изолиране на компютърни приложения от ОС в която те се изпълняват;
- ❖ Така виртуализираните приложения не се инсталират по традиционния начин, но се изпълняват все едно това е така. Приложението се държи така както се изпълнява в оригиналната си ОС и всички налични там ресурси, но може да е изолирано напълно или в „sandbox“ в различна степен;

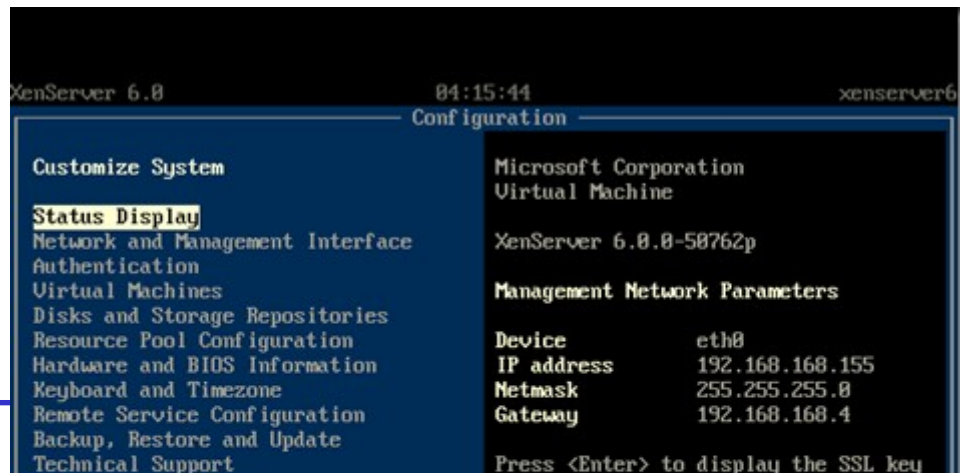
Системи за Виртуализация

Съвременни системи за виртуализация (Тип 1)

- ❖ VMware ESX and ESXi;
- ❖ Microsoft Hyper-V;
- ❖ Citrix XenServer;
- ❖ Oracle VM;
- ❖ и др.

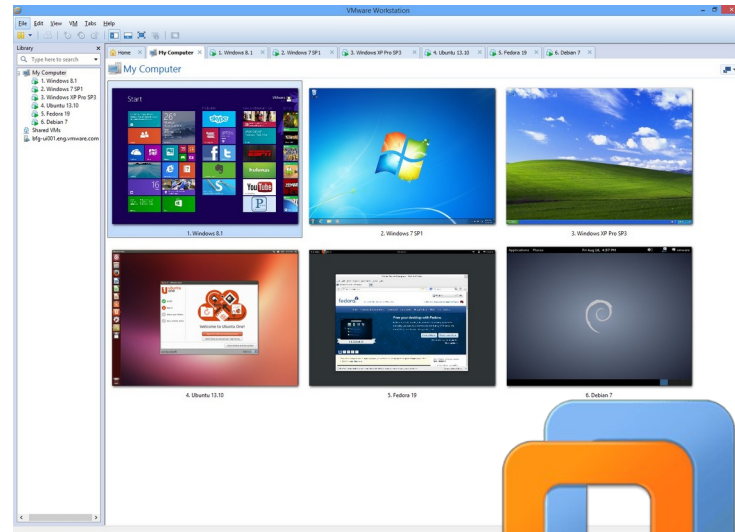
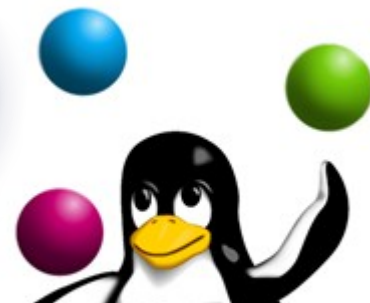
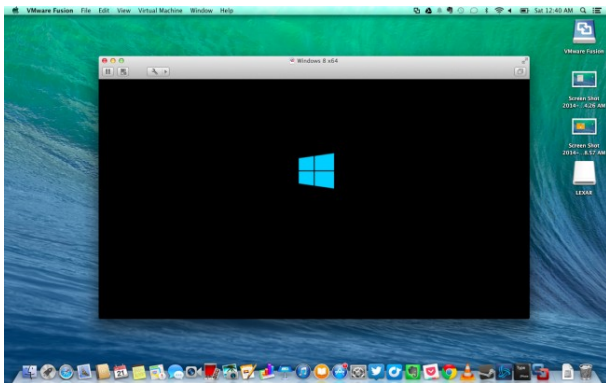
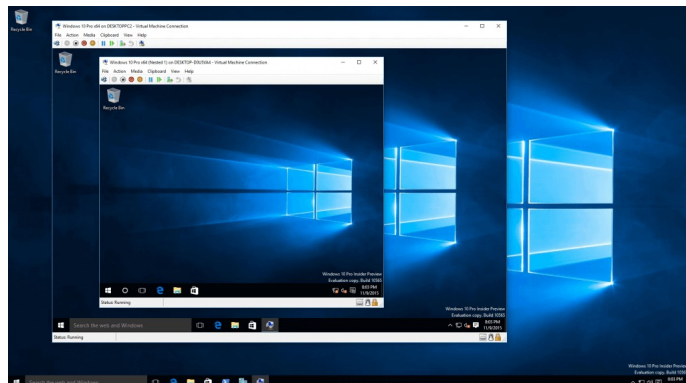


Microsoft
Hyper-V



Съвременни системи за виртуализация (Тун 2)

- ❖ VMware Workstation/Fusion/Player;
- ❖ Microsoft Virtual PC;
- ❖ Oracle VM VirtualBox;
- ❖ Red Hat Enterprise Virtualization;
- ❖ Kernel-based Virtual Machine (KVM);
- ❖ Parallels;
- ❖ и др.



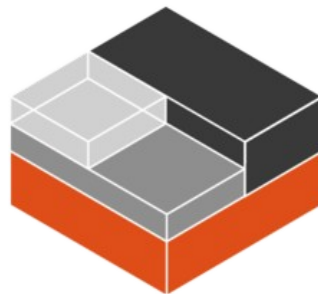
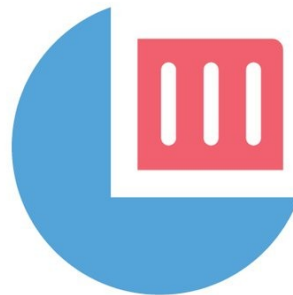
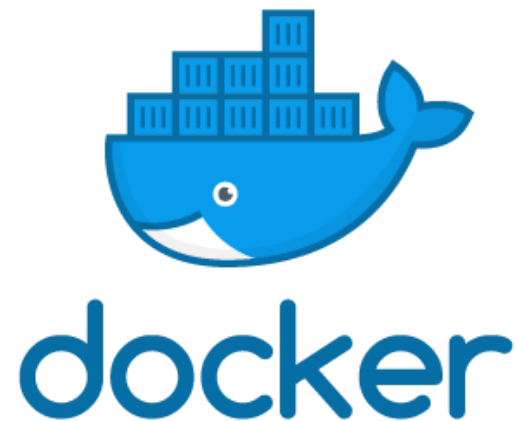
Други системи – емулатори, хипервайзори, специализирани...

- ❖ Basilisk II (емулатор)
- ❖ Bhyve (хипервайзор)
- ❖ Bochs (емулатор)
- ❖ Cooperative Linux (виртуализираща платформа)
- ❖ DOSBox (виртуална машина, емулатор)
- ❖ DOSEMU (слой за съвместимост – хардуерна вирт. и емуляция)
- ❖ PikeOS (Real time OS, хипервайзор)
- ❖ SheepShaver (емулатор)
- ❖ Windows on Windows (слой за съвместимост)
- ❖ и др.

Съвременни системи за виртуализация (OS-Level)

OS-Level Виртуализация:

- ❖ LXC;
- ❖ Docker;
- ❖ Container Linux / CoreOS Linux;
- ❖ Chroot, FreeBSD jail, sysjail;
- ❖ Solaris Containers (Zones);
- ❖ OpenVZ;



Съвременни системи за виртуализация (App-Level)

- ❖ VMware ThinApp;
- ❖ Citrix XenApp;
- ❖ Microsoft App-V;
- ❖ ZeroVM;
- ❖ Cameyo;
- ❖ Turbo;
- ❖ Vine;
- ❖ PortableApps;



Предимства и Недостатъци

Приложения, предимства и недостатъци

Предимства/Приложения:

- ❖ Консолидиране на сървъри – при много ненапълно натоварени физически машини;
- ❖ Пестене на ток – при консолидация разхода на ток може да намалее десетки пъти;
- ❖ Лесен дистанционен достъп – ВМ могат много по-лесно да бъдат контролирани;
- ❖ Нови ВМ могат да бъдат създавани без закупуване на нов хардуер;
- ❖ Лесно преместване на ВМ от една физическа машина на друга;
- ❖ Възстановяване от „бедствия“ или при пробиви в сигурността – бързо и лесно;
- ❖ Лесно дублиране/клонирание на машина;
- ❖ Може да се използват няколко ОС едновременно;
- ❖ За стартиране на стари или нови експериментални ОС или приложения;
- ❖ За обучение;

Недостатъци:

- ❖ Повишено натоварване заради изолацията и виртуализацията – ВМ са по-неефективни;
- ❖ При повече ВМ на един компютър има взаимно влияние на производителността;
- ❖ По-голяма сложност на системата;
- ❖ Рискове за сигурността;

Въпроси?

apenev@uni-plovdiv.bg