Progresivní technologie v informatice II

Vysoce dostupné systémy

Clusterová řešení odolná proti poruchám

Fakulta informačních technologií
České vysoké učení technické v Praze

Financováno Evropskou unií
NextGenerationEU
Národní plán obnovy
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Databázová řešení vysoké dostupnosti

- Aby byly schopné zlepšit parametry vysoké dostupnosti, moderní databázové stroje používají tzv. redo logy (After Image nebo Before Image).
- Tyto techniky jsou užitečné pro databáze běžící na jednom nebo více serverech.
- Databázový stroj obvykle pracuje se dvěma typy transakčních logů (žurnálů):
  - on-line redo logy obsahují oboje = uzavřené (commit) i neuzavřené transakce (Before Image).
  - archivní redo logy obsahují jen uzavřené transakce (zpravidla After Image).

Vysoká dostupnost jediného serveru

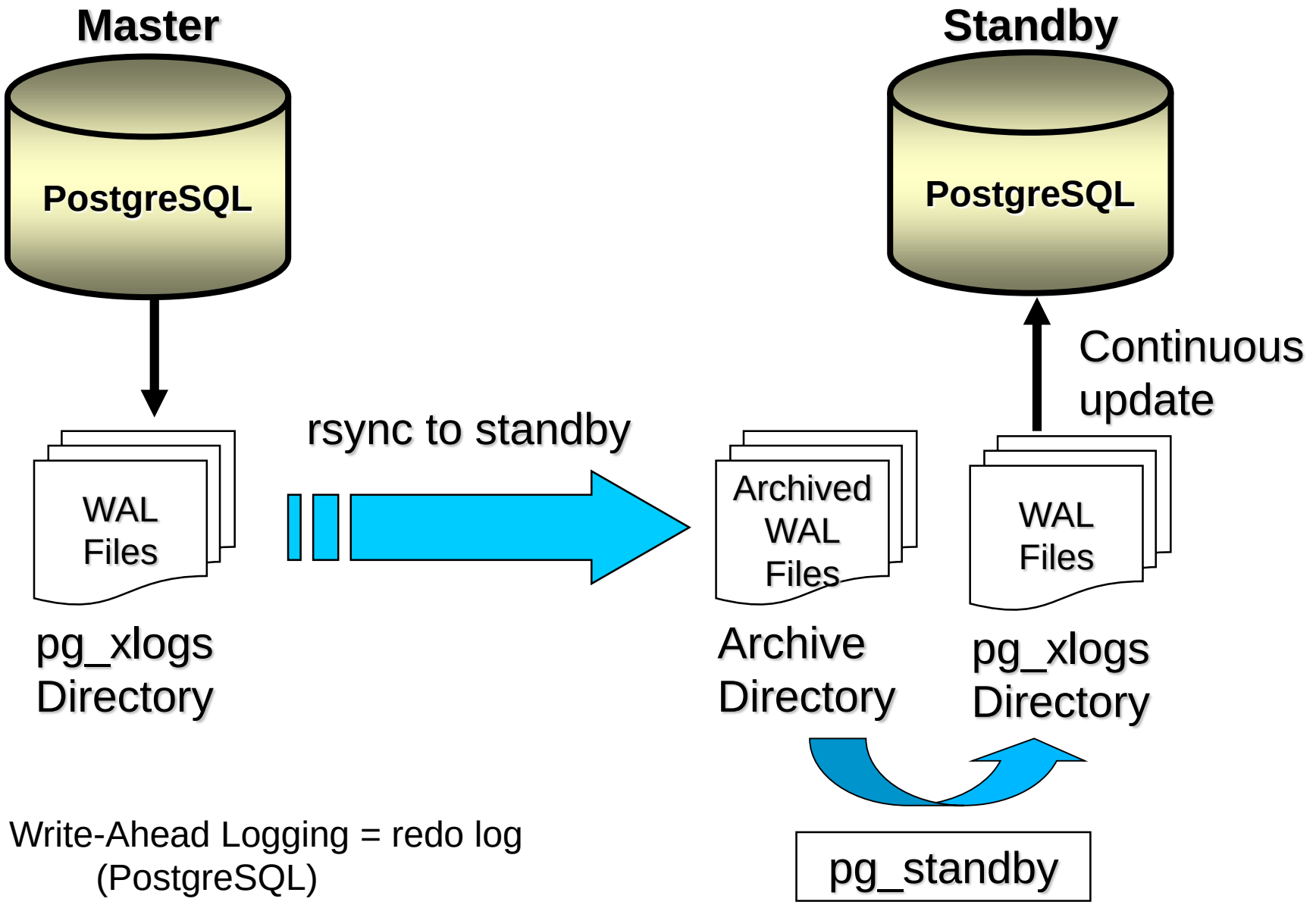
- On-line redo logy se používají pro rychlé zotavení z výpadku systému pro zajištění konzistence dat:
  - uzavřené transakce se aplikují znovu,
  - neuzavřené transakce jsou „odrolovány“ (rolled back).
- Protože on-line redo logy obsahují i stará data (Before Image Journal), mouhou být použité i pro opravu dat po chybách aplikace nebo databáze.
- Archivní redo logy jsou de-facto diferenční inkrementální zálohou databáze, mohou být aplikovány i na obnovenou plnou kopii databázových souborů (After Image Journal).

Vysoká dostupnost více serverů

Několik úrovní vysoké dostupnosti:

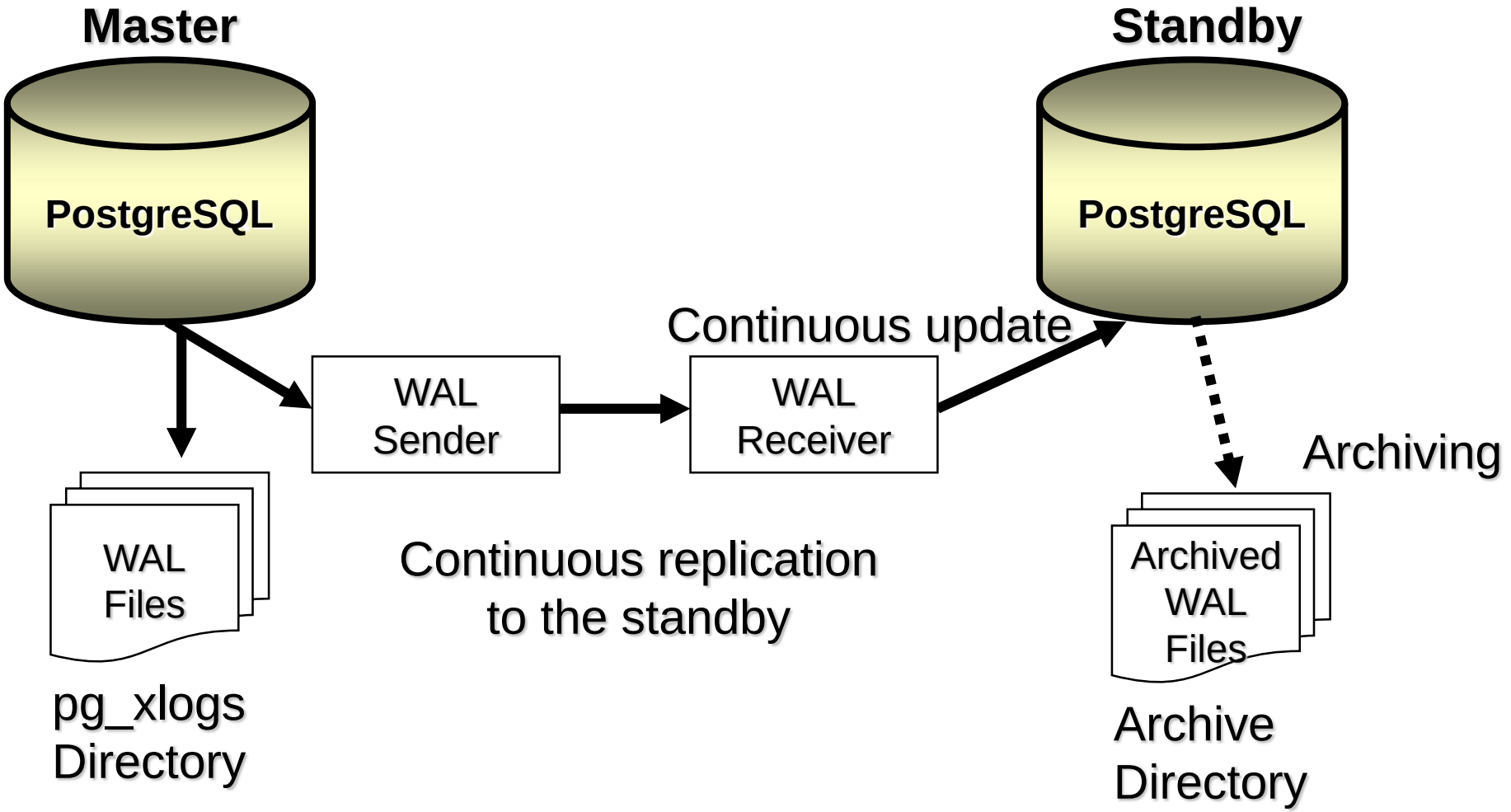
- **Cold Standby**: přesun uzavřených archivních logů na záložní systém a aplikace těchto logů na obnovenou plnou kopii databáze (v případě potřeby).
- **Warm Standby**: automatizovaný přesun archivních logů a jejich aplikace na záložní (standby) databázi.
- **Hot Standby**: automatizovaná asynchronní on-line replikace uzavřených transakcí do záložní databáze. Hot a Warm Standby databáze se mohou v některých řešeních používat pro read-only dotazy.
- **Database Mirror**: synchronní on-line replikace uzavřených transakcí do záložní databáze.

Warm Standby: Automatizovaná aplikace logů



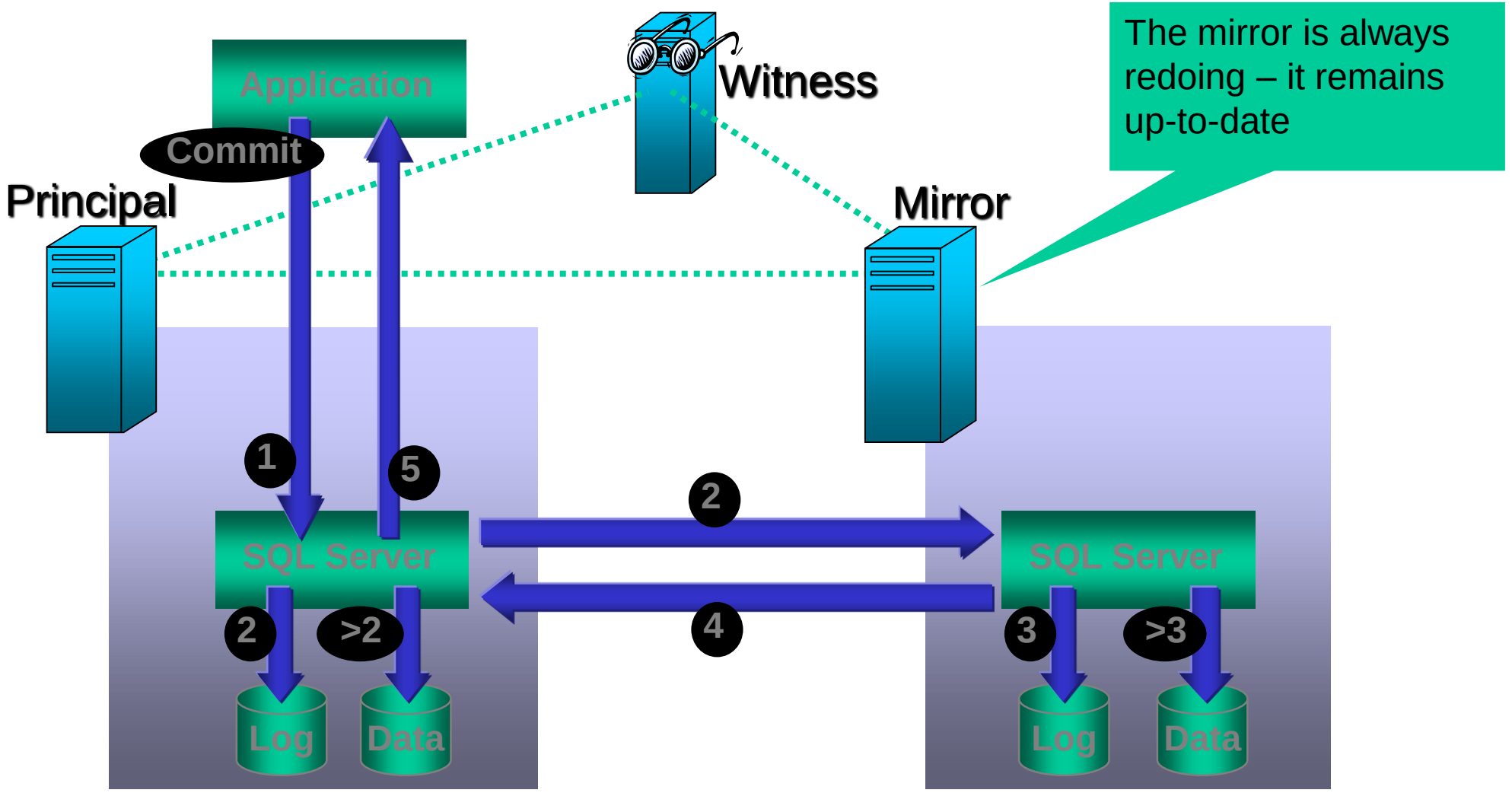
WAL = Write-Ahead Logging = redo log (PostgreSQL)

Hot Standby: Přenos logu komunikačním kanálem



Zrcadlení databáze (Database Mirroring)

Zrcadlení databáze je synchronní replikace uzavřených transakcí do záložní (zrcadlené) databáze.



Storage Fail-over clustery

- Aplikace běží na jednom uzlu (uzel má přístup k datům aplikace), ostatní uzly jsou připraveny aplikaci převzít (take over).
- **Clusterový balíček** (cluster package) je prostředí potřebné pro běh aplikace, obsahuje všechny potřebné prostředky (logické disky, systémy souborů a IP adresy). Může být:
  1. v případě nedostupnosti automaticky převeden na jiný uzel (fail-over) nebo
  2. přepnut na jiný uzel administrátorem manuálně (push-button řešení).
- Pro tento typ clusterů se používá i název Active-Passive clustery.

Clusterový balíček (Cluster Package)

- Clusterový balíček se skládá z:
  - aplikačního kódu,
  - logických disků a jejich mount pointů,
  - IP (nebo symbolické) adresy, na které je aplikace přístupná,
  - skriptů pro ovládání aplikace ze strany clusterového softwaru:
    - start aplikace,
    - stop aplikace,
    - kontrola, zda aplikace běží.
  - seznamu uzlů, kde může běžet včetně preferovaného,
  - atributů definujících chování balíčku v clusteru:
    - auto-start (zda má být automaticky nastartován),
    - fail-over (zda má být automaticky restartován na jiném uzlu),
    - fail-back (zda má být automaticky vrácen na preferovaný uzel, když se opět stane dostupným).

Start clusterového balíčku

Při startu clusterového balíčku (tj. aplikace) clusterový SW:

- zkontroluje, zda aplikace již neběží na jiném uzlu,
- provede mount registrovaných logických disků balíčku na jejich mount pointy,
- registruje IP adresy v TCP/IP stacku (nebo symbolické jméno aplikace v DNS),
- spustí startovací skript,
- monitoruje dostupnost aplikace.

Zastavení clusterového balíčku

Při zastavení clusterového balíčku (tj. aplikace) clusterový SW:

- spustí skript pro zastavení aplikace,
- čeká, až se aplikace zastaví,
- zruší registraci IP adresy v TCP/IP stacku (nebo symbolického jméno aplikace v DNS),
- odpojí (dismount) všechny logické disky balíčku.

Fail-over a Fail-back clusterového balíčku

- Detekuje-li clusterový SW nedostupnost aplikace, nejprve zkouší restart aplikace na stejném uzlu.
- Neřeší-li restart problém (aplikace je stále nedostupná), clusterový SW převede aplikaci na jiný uzel (package fail-over):
  - spustí skript pro zastavení, zruší registraci IP adresy, odpojí disky,
  - vybere další uzel ze seznamu a spustí balíček na něm.
- Detekuje-li clusterový SW nedostupnost uzlu, restartuje balíček na jiném uzlu (fail-over).
- Navrátí-li se preferovaný uzel aplikace zpět do clusteru, vrátí se aplikační balíček na preferovaný uzel (package fail-back).

Příklady Storage Fail-over clusterů

- HP Serviceguard
- Oracle Solaris Cluster
- IBM PowerHA SystemMirror for AIX
- Microsoft Cluster Server
- Veritas Cluster Server (unix, Linux, Windows)
- Linux-HA
- Red Hat Cluster Suite

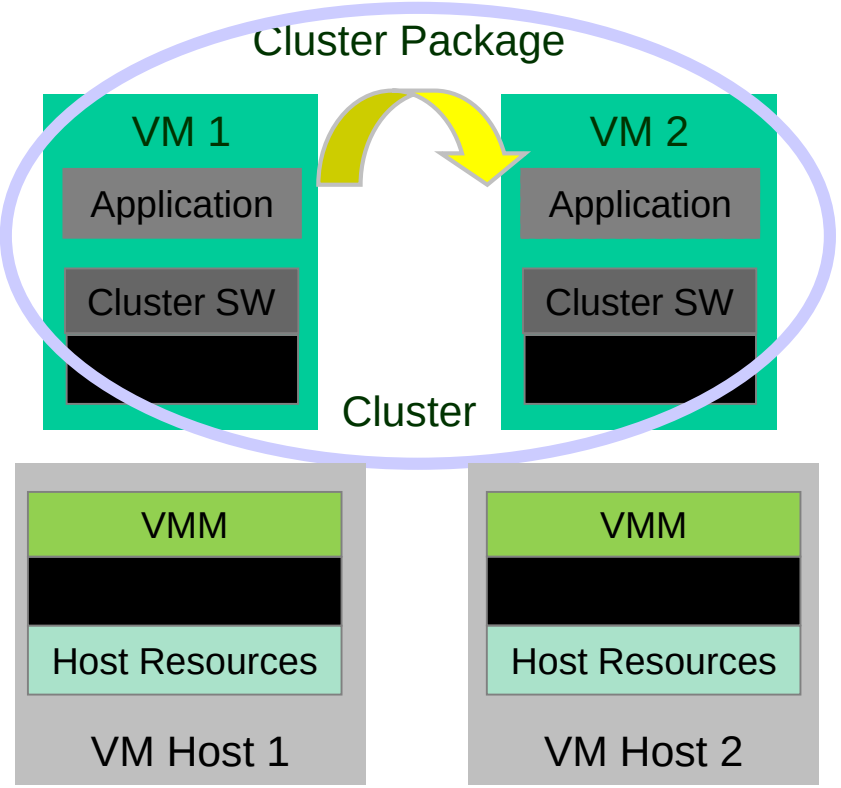
Virtuální stroje a clustrování

V zásadě existují dvě možnosti kombinování virtuálních strojů s clustery:

- Virtuální stroj jako uzel clusteru (A VM as a cluster node). Cluster je složený z virtuálních počítačů – clusterovaný je operační systém virtuálního počítače (guest OS).
- Virtuální stroj jako „cluster package“ (A VM as a cluster package). Cluster je na úrovni VMM, tzn. že clusterovaný je operační systém fyzického počítače (host OS).

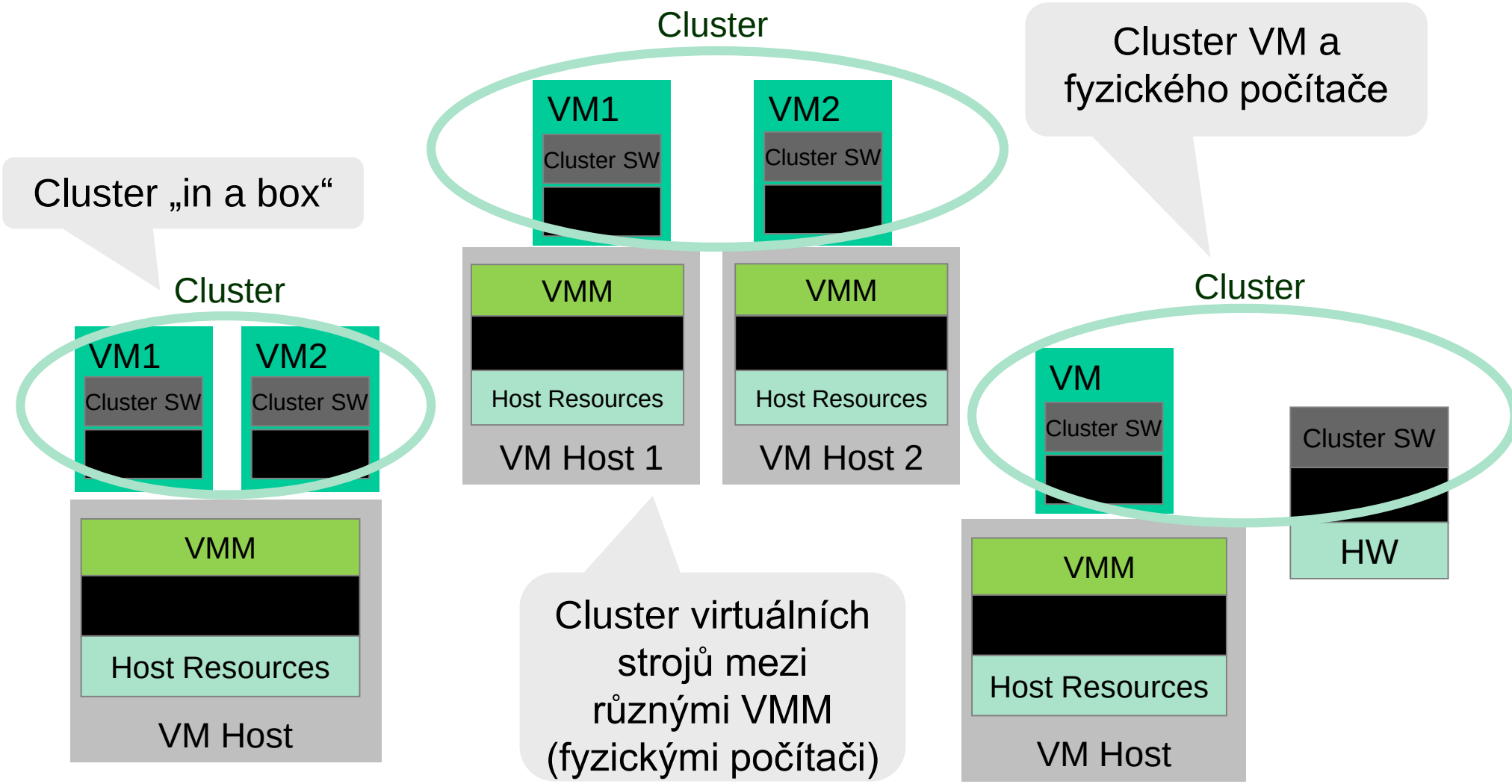
Virtuální stroj jako uzel clusteru

- clusterový SW běží na instancích OS na virtuálních počítačích.
- OS virtuálních počítačů tvoří cluster – tedy všechny VM mohou běžet souběžně.
- Je to stejné jako u tradičních clusterů, jen používají místo fyzických počítačů virtualizované.
- Aplikace v operačním systému virtuálního počítače je „cluster package“.



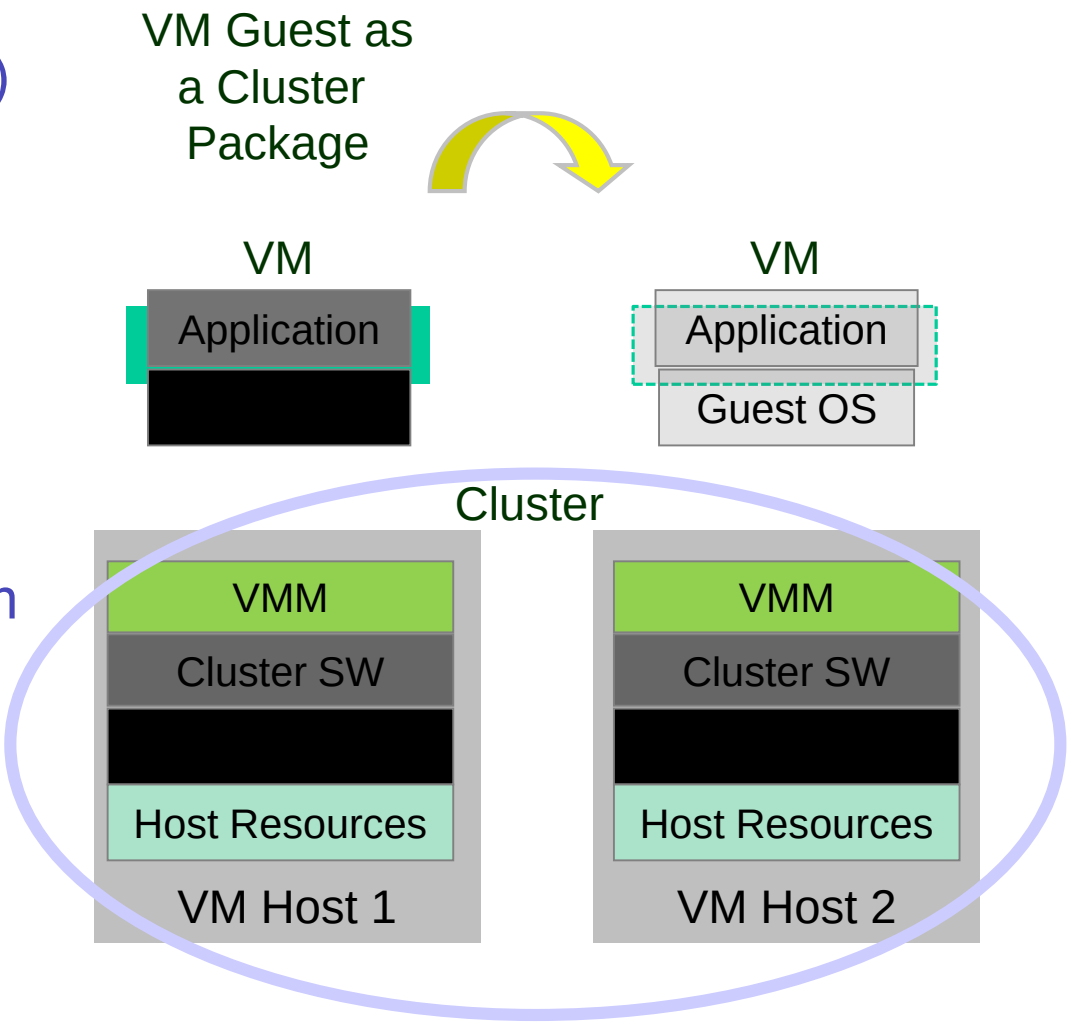
Virtuální stroj jako uzel clusteru - architektury

V zásadě můžeme použít 3 typy architektur:



Virtuální stroj jako „cluster package“

- Cluster SW běží na všech fyzických počítačích.
- Celý virtuální stroj může být přesunut (spolu s OS a aplikacemi) mezi VMM na různých fyzických počítačích.
- Cluster package obsahuje prostředky potřebné pro VM a skripty pro start/zastavení VM.
- Toto řešení poskytuje vysokou dostupnost pro aplikace na jediném serveru (nepřizpůsobené pro běh na clusteru).
- VM běží nejvýše na jednom VMM (fyzickém počítači).



Virtuální stroj jako „cluster package“: Migrace

- Většina současných virtualizačních řešení poskytuje clusterové funkce:
  - fail-over virtuálního stroje na jiný uzel clusteru (tj. stop & restart na jiném VM hostu) a
  - migraci běžícího virtuálního stroje na jiný uzel (bez přerušení běhu).
- Existují i další zajímavé vlastnosti:
  - VM snapshoting, tj. uschování a obnova stavu celého virtuálního stroje (CPU registry, obsah paměti a disků).

Živá migrace VM

- VMM na cílovém uzlu spustí “shadow VM”, aby na něj přenesl stav běžícího virtuálního stroje (prefetching phase). VMM na uzlu, kde dosud běží, požádá OS virtuálního počítače o redukování použité paměti (swap na disk) a následně monitoruje přístupy běžící aplikace do paměti a replikuje ji do „shadow VM“ na cílovém uzlu.
- Když je paměť (téměř) celá replikována, spustí fail-over virtuálního stroje. VM se pozastaví, přenese se zbytek obsahu paměti a obsah registrů, I/O zařízení (disky a síť) se připojí na novém uzlu a VM pokračuje v běhu zde.

Příklady: VMWare, IBM Live Partition Mobility, HPVM, ...

Implementace živé migrace VM

- Dává dobrý smysl nechat operační systém virtuálního počítače rozhodnout, které paměťové stránky ponechat v použití a které odsunout na disk.
- VMM používá pro tento účel koncept **balónového** procesu/driveru, který běží v operačním systému virtuálního počítače.

Balónový proces/driver

- Potřebuje-li VMM odsunout paměťové stránky virtuálního stroje na disk, požádá balónový proces/driver o alokaci více fyzické paměti od operačního systému virtuálního stroje.
- Aby mohl OS přidělit balónovému procesu/driveru více fyzické paměti, musí ji odebrat jiným procesům nebo uvolnit z diskové paměti cache. OS tedy odsune nepoužívané části adresního prostoru procesů na stránkovací/swapovací disk či soubor.
- VMM nyní má informaci, které paměťové rámce virtuálního počítače nejsou používané, protože je vlastní balónový proces/driver.
- Tyto paměťové rámce se nemusí migrovat na shadow VM.
- Stejný mechanismus VMM používá při realizaci požadavku administrátora na zmenšení objemu paměti virtuálního stroje.

Další možné vylepšení virtualizace paměti ve VMM

- VMware sleduje duplicity v obsahu paměti různých virtuálních strojů (de-duplikace obdobná mechanismu použitému ve virtuálních páskových knihovnách: viz přednáška Datová úložiště II.).
  - Používá se pro kód operačního systému (jádra, driverů, sdílených knihoven, ...) a aplikací.
  - Uchovává se jen jedna kopie stránky.
  - Copy-on-write policy.

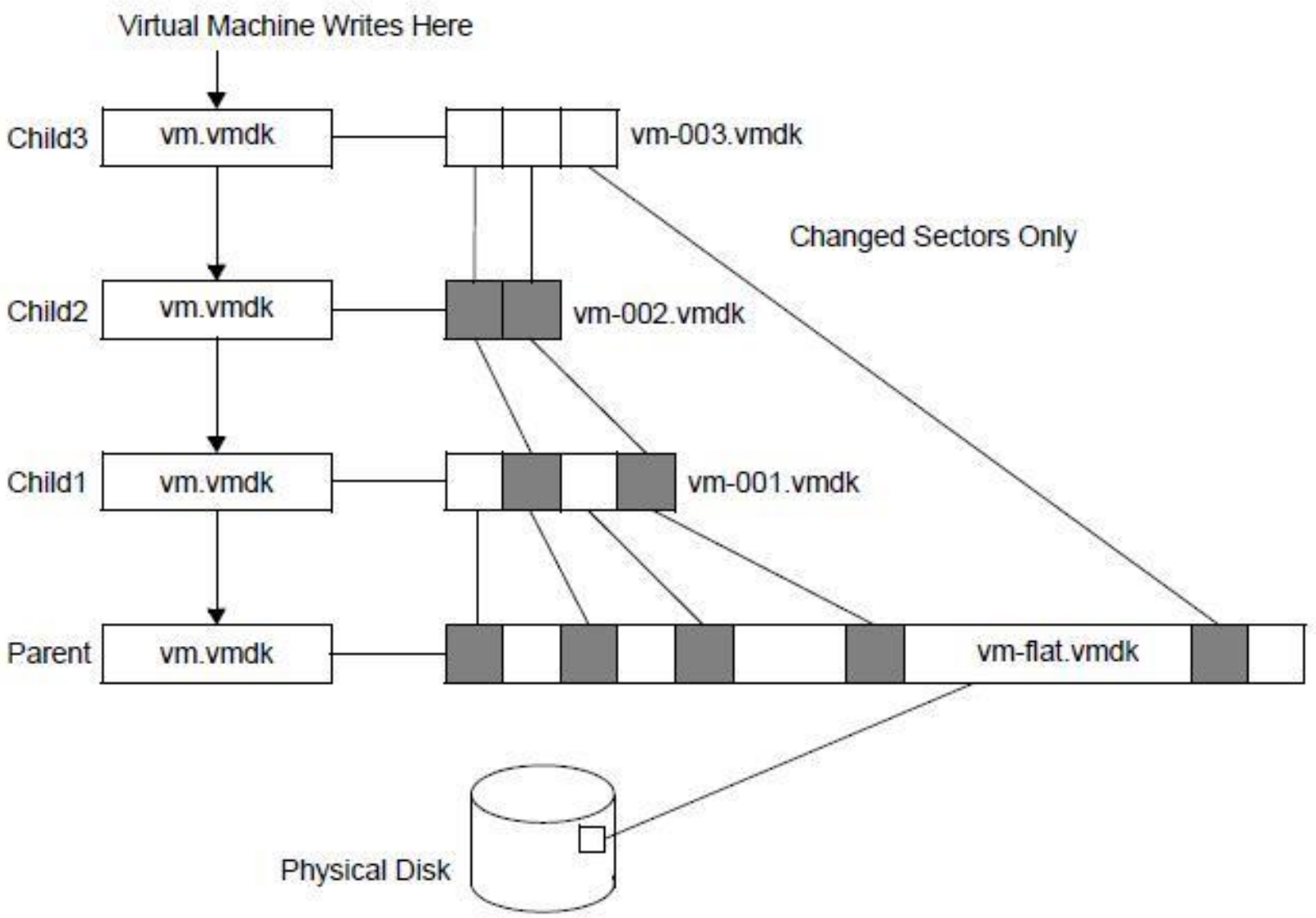
Snapshot virtuálního stroje (VM Snapshot)

- Některé VMM umožňují pořízení „snímku“ - snaphotu celého virtuálního stroje.
- **VM snapshot** zachovává stav a data virtuálního stroje ke konkrétnímu časovému okamžiku.
- Za běhu virtuálního stroje se dá vytvořit více snapshotů. Můžete se s virtuálním strojem vrátit ke kterémukoli zaznamenanému stavu tím, že na daný virtuální stroj aplikujete příslušný snapshot.
- Pro realizaci snapshotů VM je nutná virtualizace disků podporující snapshoty.

Snapshot virtuálního stroje

- **VM snapshot obsahuje následující informace**:
  - Nastavení virtuálního stroje a stav jeho napájení (zda běží nebo ne).
  - Obsah operační paměti (pokud běží)
  - Obsahy registrů všech procesorů (pokud běží).
  - Stav všech virtuálních disků virtuálního stroje (viz přednáška 3, snapshot).
    - Stejná technika (Copy on Write), ale implementovaná ve VMM.
    - Trochu zdržuje běh VM.

Příklad: Řetězení snapshotů VM ve VMware



Příklady VM

- VMWare
- PowerVM
- HPVM
- Oracle VM
- Oracle VirtualBox
- Microsoft Hyper-V
- Xen
- KVM