UNIVERZITA PARDUBICE

Fakulta elektrotechniky a informatiky

Porovnání výkonu softwarových diskových polí

Jan Růžička

Bakalářská práce

2014

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2014/2015**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Jan Růžička**
Osobní číslo:	**I10195**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Porovnání výkonu softwarových diskových polí**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

V práci budou popsány technologie zápisu dat na disková pole (RAID) včetně doporučení jednotlivých zapojení pro konkrétní účely. V praktické části bude provedena instalace systému Linux na systém s alespoň třemi volnými disky, které budou zapojeny do softwarových polí různých typů. Následně budou provedeny výkonnostní testy nad diskem a softwarovými zapojeními diskových polí (přístupová doba, sekvenční a náhodné čtení a zápis) a bude provedeno vyhodnocení a porovnání výkonu jednotlivých zapojení.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1) **VESELSKÝ, Jiří, et al. Linux - Dokumentační projekt. 4. aktualiz. vyd. Brno: Computer Press, 2008. 1336 s. ISBN: 978-80-251-1525-1. Dostupný též z WWW: http://www.root.cz/knihy/linux-dokumentacni-projekt-4-vydani/stahnout/.**
2) **Shah, S.: Administrace systému Linux - podrobný průvodce začínajícího administrátora. Praha: Grada, 2002. ISBN 80-7169-586-6.**
3) **Benchmarking sw: http://www.acnc.com/benchmarks.html.**
4) **Bonnie: http://www.textuality.com/bonnie/.**
5) **Bonnie++: http://www.coker.com.au/bonnie++/.**
6) **Dokumentace jádra Linuxu: http://www.kernel.org/.**
7) **Phoronix Test Suite: http://www.phoronix-test-suite.com/.**

Vedoucí bakalářské práce: **Mgr. Tomáš Hudec**
Katedra informačních technologií

Datum zadání bakalářské práce: **24. listopadu 2014**
Termín odevzdání bakalářské práce: **5. prosince 2014**

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Ing. Lukáš Čegan, Ph.D.
vedoucí katedry

V Pardubicích dne 24. listopadu 2014

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 5. 12. 2014

Jan Růžička

Poděkování

Tímto bych rád poděkoval Mgr. Tomáši Hudcovi za věcné rady a trpělivost v průběhu zpracování této bakalářské práce. Poděkování také patří mé rodině za podporu a vytvoření kvalitního zázemí během studia na vysoké škole.

Anotace

Hlavní náplní této práce je porovnání výkonu různých typů softwarových diskových polí v operačním systému Linux. Je zaměřena na nejpoužívanější typy diskových polí RAID, na kterých byly vytvořeny souborové systémy EXT4 a ZFS. Jsou uvedeny rozdíly mezi různými diskovými poli a zhodnoceno jejich praktické využití. Výkonnostní testy byly provedeny pomocí benchmarku Bonnie++ a výsledky byly porovnány pomocí grafů.

Klíčová slova

RAID, Pole, Souborový systém, Benchmark

Title

Performance comparison of software arrays

Annotation

The main goal of this thesis is comparison between different types of software disk arrays in Linux. It is focused on the most used types of RAID arrays which were formated as ext4 and zfs. The differencies between arrays are stated as well as their practical use. The performance benchmarks were conducted using the Bonnie++ benchmarking program and the results were compared graphically.

Keywords

RAID, Arrays, File Systems, Benchmark

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek a pojmů

Benchmark	Testovací počítačový program.
Buffer	Část paměti, je určena pro dočasné uchování dat před přesunem na jiné místo.
CSV	Comma-separated values (hodnoty oddělené čárkami) je jednoduchý souborový formát určený pro výměnu tabulkových dat.
Defragmentace	Je proces zpětného skládání celku z dílčích částí – fragmentů.
Duplexing	Technika připojení jednotlivých disků k samostatným diskovým řadičům.
ECC	Error Checking and Correcting (kontrola a oprava chyb).
Extent	Rozsah navazujících fyzických bloků (resp. alokačních jednotek).
Fdisk	Počítačový program, slouží k vytvoření diskových oddílů na pevném disku.
Filesystem	Souborový systém.
Fsck	File System Check, příkaz řešící kontrolu souborového systému a jeho opravy.
GNU/Linux	Označení pro operační systém založený na principech unixových systémů.
HDD	Hard Disk Drive, fyzický pevný disk.
Inode	I-uzel, je datová struktura uchovávající metadata o souborech a adresářích.
Live CD	Je operační systém uložený na bootovatelném CD, není potřeba instalace OS.
MD	Multiple Devices (mnohonásobné zařízení).
Mdadm	Nástroj pro správu MD a tvorbu RAID polí.
Metadata	Strukturovaná data o datech.
Mirror	Typ diskového pole (zrcadlo).
Mkfs	Příkaz používaný k vytvoření souborového systému v OS založeném na UNIX.
Mount	Příkaz používaný k připojení souborového systému, který je umístěn na zařízení.
Mountpoint	Přípojný bod souborového systému do OS.
Nabootovat	Zavést operační systém do paměti počítače.

OS	Operační systém.
Pool	Uložiště, skládající se z mnoha virtuálních datových zařízení.
Portace	Znamená obecně přenos, přesun či přemístění mezi dvěma subjekty.
Posix	Portable Operating System Interface, standard hlavně používaný unixovými OS.
RAID	Redundancy Array of Inexpensive Disks (Redundatní pole levných disků) nebo Redundancy Array of Independent Disks (Redundantní pole nezávislých disků).
Raidz	Typ diskového pole.
Read/write cache	Čtecí/zapisovací cache (cache – mezi-paměť).
Stream	– proud, technologie kontinuálního přenosu audiovizuálního materiálu.
Stripe	Základní jednotka pole (stripe - pruh), blok dat určité velikosti.
Superblok	Hlavička souborového systému.
Swap	Swapovací oddíl, úložiště dat používané pro realizaci virtuální paměti.
Top-level	Nejvyšší zařízení u ZFS diskových polí.
Update time	Aktualizace času, slouží pro zjištění, kdy byl čas změněn.
Utilita	Pomocný program, slouží k zjednodušení práce s počítačem.
Vdevs	Virtuální datová zařízení, která tvoří ZFS pool.
XOR	Logický člen - exkluzivní disjunkce (také nonekvivalence).
ZFS	Zettabyte File Systém, souborový systém a správce logických svazků.
Zfsutils	Nástroj pro vytváření RAID polí na souborovém systému ZFS.
Zpool	Příkaz pro konfiguraci ZFS diskových polí.
Žurnálovat	Žurnálovací systém souborů zapisuje změny, které mají být v počítačovém systému souborů provedeny, do speciálního záznamu nazývaného žurnál.

0 Úvod

Cílem této práce je otestování výkonu softwarových diskových polí, porovnání různých typů zapojení mezi sebou a doporučit vhodná zapojení pro konkrétní situace. Motivací této práce by mělo být usnadnění firmám, podnikům, ale i domácnostem v rozhodování nad zvažovanou investicí.

V této práci bude popsáno fungování a konfigurace diskových polí RAID s pomocí dvou různých souborových systémů, konkrétně ext4 a zfs v operačním systému Linux, vzhledem k maximálnímu výkonu, dostatečné bezpečnosti a minimální ceně.

První kapitola je zaměřena na historii vzniku diskových polí a popsání jejich vlastností. Představí různé druhy implementací diskových polí a rozdíly mezi nimi.

Ve druhé kapitole je stručně popsán operační systém Linux, výběr linuxových distribucí a instalace samotného operačního systému, na kterém byly disková pole vytvářeny a měřen jejich výkon.

Třetí kapitola pojednává o architektuře pevných disků, vytváření oddílů a o souborových systémech, které jsou nezbytnou součástí pro provozování diskových polí.

Ve čtvrté kapitole je uvedena konfigurace různých typů zapojení diskových polí na souborových systémech ext4 a zfs.

Poslední pátá kapitola je věnována představení testovacího programu Bonnie++. Je popsána konfigurace jednotlivých testů a vyhodnocení výsledků měření výkonu diskových polí.

1 Disková pole RAID

Vlastní motivace k vytvoření této práce byl fakt, že ceny fyzických pevných disků už nejsou tak závratné, jako tomu bylo v minulosti. Každý si tak může dovolit vytvořit diskové pole, ale ne každý zná tento pojem a dokáže takové pole sestavit, tato práce by měla objasnit danou problematiku a pomoci i méně zdatným uživatelům k vytvoření diskového pole a využívání jeho výhod.

1.1 Historie RAID

Historicky byla situace spíše opačná. Fyzické pevné disky měly malou kapacitu, byly dost drahé, pomalé a spolehlivost nebyla skoro žádná. V roce 1988 byla vydána publikace Univerzitou California *A Case For Redundant Arrays of Inexpensive Disks*. Objevoval se termín **RAID** (Redundancy Array of Inexpensive Disks). Na trhu však byly k dostání dva typy, takzvaných spolehlivých a levných disků. Spolehlivé disky nabízely o něco vyšší spolehlivost, avšak za ceny několikrát vyšší. Dnes se termín RAID přeformuloval na význam v souvislosti s rozvojem technologií ukládání dat na *Redundancy Array of Independent Disks*.

„Původní ovladač **meta-disku**, odtud název ovladače **md**, napsal Marc Zyngier. Implementace zpočátku delší dobu podporovala pouze typy Linear, RAID 0 a RAID 1. Později ovladač vyvíjeli zejména Ingo Molnár, Gadi Oxman a nyní Neil Brown. Přibyla podpora dalších typů polí a podpora bootování z RAIDu 0 a 1. K provozování softwarového RAIDu je zapotřebí kromě jaderného ovladače také sada utilit *raidtools* (dříve mdtools), které musejí odpovídat verzi jaderného ovladače (raidtools 0.42 a raidtools 0.90).“ (1)

1.2 Teorie, vlastnosti a porovnání různých typů RAID polí

Definice RAID

Termín RAID je v informatice metoda zabezpečení dat proti selhání fyzického pevného disku a metoda jak zvýšit výkon diskového systému bez extrémních pořizovacích nákladů. Zabezpečení je realizováno specifickým ukládáním dat na více nezávislých pevných disků. Pro některá zapojení vystačí vlastnit dva pevné disky, některé zapojení vyžadují tři a více pevných disků. Jednotlivé typy zapojení RAID polí jsou označovány čísly, nejčastěji RAID 0, RAID 1, RAID 5. Patří všude tam, kde je výpadek disku nebo dokonce ztráta dat nepřípustná. Příklady použití najdeme na webových serverech, databázových úložištích, nebo v multimediálních centrech.

Úrovně RAIDu se liší různou úrovní zabezpečení dat a výkonností. V praxi se nepoužívají všechny definované úrovně, ale jen ty nejvýhodnější.

Stavy

Při provozování diskových polí se můžeme dostat do takzvaného degradovaného stavu, při kterém dojde k výpadku některého disku, typicky se sníží výkon, avšak stále jsou všechna uložená data k dispozici. Obsluha musí vyměnit havarovaný disk za nový a ten se začlení zpět do pole. Čímž se dostáváme do dalšího ze stavů, kdy probíhá rekonstrukce pole a jsou dopočítávány chybějící data a ty jsou následně zapsána na nový disk. Při tomto stavu jsou data stále přístupná. Po dokončení rekonstrukce je RAID pole opět ve stavu synchronizováno. Ovšem neplatí to pro všechna zapojení, příkladem je RAID 0. Jelikož zapojení RAID 0 není redundantní, nelze u něj provést rekonstrukci dat. RAID pole vytváří logický (virtuální) úložný prostor, který se navenek tváří jako jediný pevný disk. Jednotlivé disky v poli nazýváme členy pole. (2)

Rekonstrukcí je míněna buď synchronizace obsahu nového disku s ostatními aktivními disky v případě RAID 1, anebo rekonstrukce obsahu původního havarovaného disku na základě redundantní informace, jedná-li se o RAID 4 nebo 5. Po dobu, než rekonstrukce proběhne, se pole nachází v tzv. degradovaném režimu, kdy v závislosti na konfiguraci nemusí být redundantní a v případě RAIDu 5 se to projeví sníženým výkonem (odtud název **degradovaný režim**). Pro lepší automatizaci je lepší mít trvale k dispozici rezervní disk, takže rekonstrukce pole může být zahájena zcela automaticky na pozadí bez vědomí uživatele.

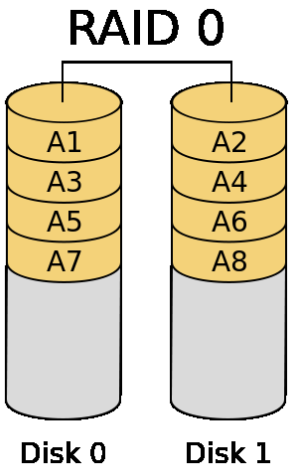
RAID neznamená zálohování

Je nutné však upozornit na fakt, že ačkoli se technologie RAID polí může zdát jako by šlo o zálohu dat, není tomu tak. Skutečná záloha však vyžaduje doplňující operace (uložení dat na bezpečné místo, jejich fyzické zabezpečení, šifrování zálohy, možnost návratu ke starší verzi dat apod.). Proto není možné při používání RAID pole samotné zálohování vyloučit. Redundantní disková pole jsou odolná pouze vůči výpadkům určitého počtu disků. Neochrání před výpadkem napájení, poškozením souborového systému při pádu celého systému nebo chybou administrátora. Proto je potřeba myslet i na další metody ochrany dat - např. na záložní zdroje napájení (UPS), žurnálovací souborové systémy apod. V každém případě pravidelně zálohovat. (1)

Populární typy RAID

RAID 0 (Nonredundant striped array)

„Tento typ je určen pro aplikace, které vyžadují maximální rychlost. Jistá nevýhoda může být, že není redundantní. Naopak je potřeba vzít v úvahu, že pravděpodobnost výpadku takového pole roste s počtem disků. Ideální použití představují audio/video streamingové aplikace, eventuálně databáze a obecně aplikace, při kterých čteme sekvenčně velká množství dat. Základní jednotkou pole je tzv. **stripe**, což je blok dat určité velikosti (běžně 4 až 64 kB v závislosti na aplikaci). Po sobě jdoucí data jsou pak v poli rozložena střídavě mezi disky do *stripů* takovým způsobem, aby se při sekvenčním čtení/zápisu přistupovalo ke všem diskům současně. Tím je zajištěna maximální rychlost jak při čtení tak i zápisu, ale současně je tím dána také zranitelnost pole. Při výpadku kteréhokoliv disku se stávají data v podstatě nečitelná, respektive nekompletní. RAID 0 bývá označován rovněž jako **Striping**. Protože není redundantní, má nejvýhodnější poměr cena/kapacita. Počet disků je libovolný. Je ovšem třeba pamatovat na to, že s rostoucím počtem disků v poli roste i pravděpodobnost výpadku pole, protože výpadek libovolného disku znamená havárii celého pole. RAID 0 je tedy velmi rychlý, ale méně bezpečný než samostatný disk.“ (1) Princip diskového pole je znázorněn na Obrázek 1.



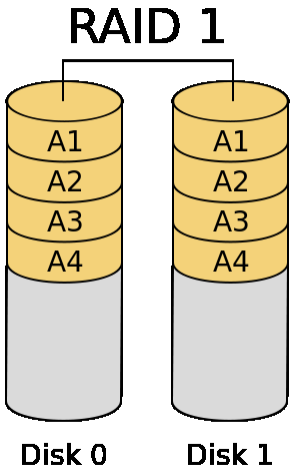
Obrázek 1 – Diskové pole RAID 0

Zdroj: (2)

Opak *Stripingu* (prokládání) je **Linear** (zřetězení). „Ve zřetězení jsou data postupně ukládána na několik disků. Jakmile se zaplní první, ukládá se na druhý, poté na třetí atd. Výhodou je snadné zvětšení kapacity přidáním dalšího členu a skutečnost, že při výpadku členu mohou být některé soubory nedotčeny.“ (2)

RAID 1 (Mirrored array)

„RAID 1 je naopak maximálně redundantní. Rychlost čtení může být oproti samostatnému disku výrazně vyšší, rychlost zápisu je stejná jako u samostatného disku. Funguje tak, že data jsou při zápisu *zrcadlena* na všechny disky v poli (tedy v případě RAIDu 1 tvořeného dvěma disky jsou data duplikována apod.). Při čtení lze využít vícero kopií dat a podobně jako u RAIDu 0 číst ze všech disků současně. Tento typ pole je určen pro aplikace s důrazem na maximální redundanci. Výhodou tohoto redundantního řešení je stabilní výkon i v případě výpadku disku, nevýhodou je poměr cena/kapacita. Počet disků bývá buď dva anebo libovolný, čím větší počet disků, tím větší redundance a odolnost proti výpadku.“ (1) Princip zapojení je znázorněn na Obrázek 2.

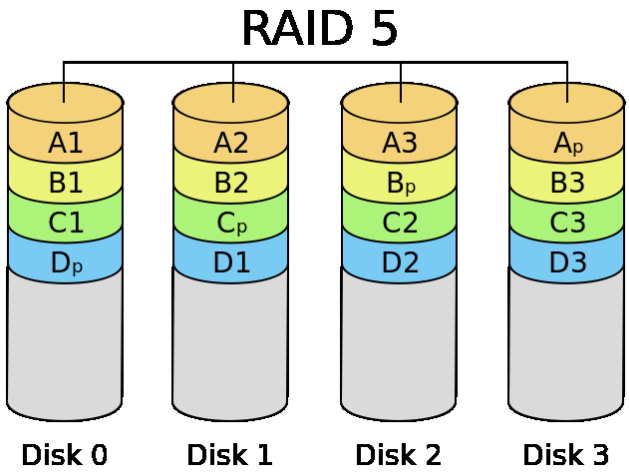


Obrázek 2 – Diskové pole RAID 1
Zdroj: (2)

„Podobná technika může být uplatněna o úroveň výše, kdy jsou použity dva samostatné řadiče. Tato technika se nazývá **duplexing** a je odolná i proti výpadku řadiče. Teoreticky se může výrazně zvýšit rychlost čtení a o něco snížit odezva, avšak záleží na konkrétním řadiči. Zápis může být pomalejší, protože se ukládají stejná data na dva disky. Technika výrazně zvyšuje bezpečnost dat proti ztrátě způsobené poruchou hardware. Nevýhodou je potřeba dvojnásobné diskové kapacity.“ (2)

RAID 5 (Striped array with rotating parity)

„Tento typ poskytuje redundanci vůči výpadku libovolného jednoho disku s dobrým poměrem cena/kapacita a výkonem. RAID 5 je vylepšená varianta RAIDu 4 v tom, že parita není uložena na jednom vyhrazeném disku, ale je rozmístěna rovnoměrně mezi všemi disky pole, čímž se odstraní úzké hrdlo architektury RAIDu 4.“ (1) Princip zapojení je znázorněn na Obrázek 3.



Obrázek 3 – Diskové pole RAID 5

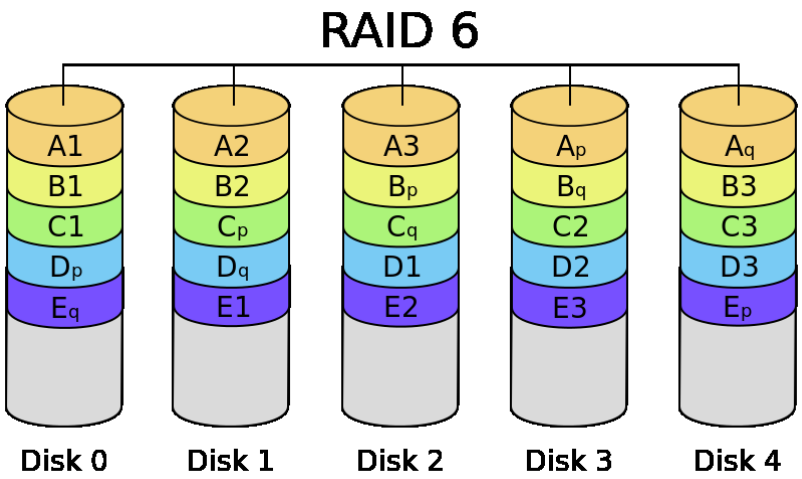
Zdroj: (2)

„Paritu lze spočítat buď tak, že skutečně načteme a *XORujeme* data z odpovídajících datových *stripů* všech disků (takto se parita počítá při inicializaci pole, vyžaduje to tedy přístup ke všem diskům). Ve druhém případě načteme původní data z datového stripu, která se mají změnit, provedeme XOR s novými daty a výsledek ještě XORujeme s původní hodnotou parity (takto se parita počítá na již inicializovaném běžícím poli). Zápis dat tedy představuje dvoje čtení (dat a parity), výpočet parity a dvojí zápis (opět dat a parity). Počet přístupů na disk při zápisu je v tomto případě konstantní bez ohledu na počet disků v poli, přistupuje se vždy ke dvěma diskům a to také má za následek nižší výkon tohoto typu pole ve srovnání s redundantním RAIDem 1.“ (1)

„V *degradovaném režimu* (degradovaný režim znamená stav, kdy je některý z disků z pole vyřazen kvůli hardwarové chybě) se pak musejí data uložená na havarovaném disku odvodit z dat zbývajících disků a parity. Na rozdíl od redundantního RAIDu 1, kde výpadek disku obvykle neznamená výrazný pokles výkonu, vykazuje RAID 5 v degradovaném režimu výrazně snížený výkon zejména při čtení. Minimální počet disků pro tento typ diskového pole jsou 3.“ (1)

RAID 6

„Obdoba RAID 5, používá dva paritní disky, přičemž na každém z nich je parita vypočtena jiným způsobem. Opět kvůli přetížení paritních disků jsou paritní data uložena střídavě na všech discích. Výhodou je odolnost proti výpadku dvou disků. Rychlost čtení je srovnatelná s RAID 5, ale zápis je pomalejší než u RAID 5, právě kvůli výpočtu dvou sad paritních informací. RAID 6 je možno sestavit z minimálně čtyř disků (pokud bychom neuvažovali rozprostření paritních informací na všechny disky, pak by se dalo zjednodušeně říci, že dva jsou datové a dva paritní). V této minimální konfiguraci se však s ohledem na výslednou kapacitu příliš nepoužívá, neboť kapacita pole je poloviční, tedy stejná jako v konfiguraci zrcadlení RAID 1 dvou párů disků. Při zrcadlení navíc není třeba počítat dvě sady paritních informací, je tedy mnohem rychlejší při zápisu a nepotřebuje vysoký výpočetní výkon. RAID 6 je tedy výhodné při použití pěti a více disků.“ (2) Princip diskového pole je znázorněn na Obrázek 4.



Obrázek 4 – Diskové pole RAID 6
Zdroj: (2)

Nestandardní a méně používané typy RAID

RAID 2 (Parallel array with ECC)

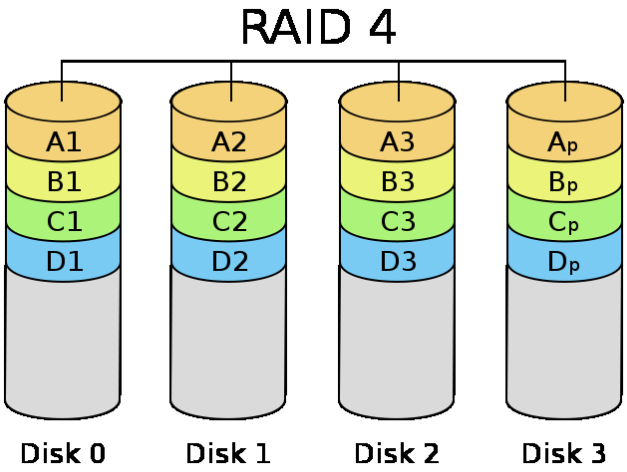
„Pole tohoto typu jsou dnes již historií, protože dnešní disky mají vlastní opravné mechanismy a uchovávají *ECC* informace pro každý sektor samy. V polích tohoto typu se stripovalo po jednotlivých sektorech a část disků pole byla vyhrazena pro ukládání ECC informací. Jakékoliv čtení i zápis proto zpravidla zahrnovalo přístup ke všem diskům pole, což bylo překážkou vyššího výkonu zejména u aplikacích pracujících se většími kusy dat. Tento typ není redundantní.“ (1)

RAID 3 (Parallel array with parity)

„Také tento typ polí se již nepoužívá, jedná se o předchůdce RAIDu 4. Stripovalo se po sektorech, ale jeden disk byl vyhrazen jako paritní, což zajišťovalo redundanci (princip zajištění redundance je stejný jako u RAIDu 4 a 5). Protože i v tomto případě se stripovalo po sektorech, jakékoliv čtení i zápis zpravidla zahrnovalo přístup ke všem diskům pole.“ (1)

RAID 4 (Striped array with parity)

„RAID 4 je redundantní pole, které se dnes již používá málo. Funkčně je podobné RAIDu 5, který je ale výkonnější. Funguje tak, že jeden disk je vyhrazen jako tzv. *paritní disk*. Na paritním disku je zaznamenán kontrolní součet (operace XOR přes data stejné pozice jednotlivých disků). Pokud tedy dojde k výpadku některého z datových disků, lze data rekonstruovat z dat zbylých disků a parity uložené na paritním disku. RAID 4 je odolný vůči výpadku libovolného jednoho disku a má tedy příznivý poměr cena/kapacita. Paritní disk ale představuje úzké hrdlo této architektury při zápisech, protože každý zápis znamená také zápis na paritní disk. Minimální počet disků je 3.“ (1) Princip zapojení je znázorněn na Obrázek 5.



Obrázek 5 – Diskové pole RAID 4

Zdroj: (2)

RAID 7

„RAID 7 bylo vytvořeno firmou Storage Computer Corporation. Je odvozené od RAID 3 a RAID 4 a liší se zejména tím, že k nim přidává vyrovnávací paměť.“ (2)

Kombinace více typů polí

„Z definic výše popsaných typů polí vyplývá, že redundantní pole nejsou tak rychlá, jak bychom si mohli přát a naopak u RAIDu 0 nám chybí redundance. Existuje ovšem možnost, jak výhody jednotlivých typů diskových polí spojit. Tato metoda spočívá ve vytvoření kombinovaných polí, kdy disky v poli určitého typu jsou samy tvořeny poli jiného typu. Příkladem může být např. *RAID 1+0*, kdy jsou pole typu RAID 1 dále sloučeny do RAIDu 0. Takové pole je pak redundantní (toleruje výpadek až dvou disků), rychlejší zejména v zápisech než samotný RAID 1, a má lepší poměr cena/kapacita než RAID 1, velikost je zde n/2 · disk a minimální počet disků je pak 4. Další možností je třeba *RAID 0+1*, *RAID 5+0*, *RAID 5+1* apod.“ (1)

- „Pole **RAID 0+1** (stripování) je kombinací RAID 0 a RAID 1. Data uložíme *prokládaně* (stripováním) na dva disky (A, B), poté totéž uděláme s dalšími dvěma disky (C, D). Získáme tak dva logické disky AB, CD, které mají redundantní obsah. Máme-li soubor, který se při stripování rozdělí na dvě poloviny, první část souboru máme na disku A a C, druhou na disku B a D. Výhodou tohoto způsobu je, že nejen rozkládáme zátěž mezi více disků při čtení a zápisu, ale data jsou také uložena redundantně, takže se dají

po chybě snadno obnovit. Mezi nevýhody patří využití pouze 50 % celkové diskové kapacity, a při výpadku jednoho ze čtyř disků ztrácíme redundanci dat.

- Pole **RAID 1+0** (stripování) je opět kombinací RAID 0 a RAID 1, ale postupujeme obráceně. Nejdříve uložíme stejná data na disk A, B, poté na disk C, D. Získáme tak dva logické disky AB, CD, na nichž jsou data uložena stripovaně. Máme-li soubor, který se při stripování rozdělí na dvě poloviny, první část souboru je na disku A a B, druhá část je na disku C a D, na rozdíl od RAID 0+1. Výhody jsou podobné RAID 0+1, navíc je RAID 1+0 odolnější proti výpadku více disků a po chybě je obnova dat mnohem rychlejší. Nevýhodou je opět využití pouze 50 % kapacity.“ (2)

Tabulka 1 – Přehled diskových polí

Zdroj: (1)

Typ	Počet disků	Kapacita	Redundance	Shrnutí
Linear	2 a více	n× disk	Ne	Nejedná se o RAID, pouze sloučení disků
RAID 0	2 a více	n× disk	Ne	Maximální výkon při čtení i zápisu
RAID 1	2 a více	disk	Ano	Maximální redundance, bez výrazné degradace výkonu při výpadku, rychlejší čtení, vyšší cena
RAID 4	3 a více	(n-1)× disk	Ano	Nepoužívá se, nahrazen RAID 5
RAID 5	3 a více	(n-1)× disk	Ano	Výhodný poměr cena/kapacita

1.3 Existence SW a HW implementací RAID polí

Společnou vlastností všech diskových polí je, že pro služby operačního systému se tváří jako jeden disk. Existují tři typy jakým způsobem docílit diskového pole. Nabízí se varianta použít hardwarově řešené diskové pole. Tato varianta je pro tvorbu diskového pole nejdražší, jelikož se skládá ze speciálního řadiče. O něco levnější způsob je použít speciální řadič pro RAID pole, který je ve skutečnosti spíše na bázi softwarového řešení diskového pole. Poslední typ je samotné softwarové řešení, které nevyžaduje speciální řadič. Všechny zmíněné typy si probereme podrobněji.

Softwarová řešení

Pro softwarová řešení, operační systém mezi ovladače řadičů vloží další vrstvu, která má na starosti obsluhu služeb RAID a připojené diskové pole se do systému hlásí jako jeden disk s kapacitou odpovídající typu zapojení pole. Základem diskového pole je utilita **mdadm** využívaná pro vytváření diskových polí na souborovém systému *ext4*. Ta vytváří v systému *md* (multiple devices) zařízení, které se skládají z jednotlivých blokových zařízení (především fyzických disků a jejich oddílů). Tato utilita umí vytvářet: RAID 0, RAID 1, RAID 4, RAID 5, RAID 6 a RAID 10 a další speciální režimy LINEAR, MULTIPATH a FAULTY. Mnou otestovaná zapojení jsou: RAID 0, RAID 1, RAID 5. Dále utilita **zfsutils** pro řešení diskových polí na souborovém systému *zfs*. Která umí vytvořit typ diskových polí: **ZFS Mirror** (obdoba RAID 1) a **Raidz** (obdoba RAID 5). Tyto typy jsem taktéž testoval.

Výhodou softwarových diskových polí je veliká flexibilita, nízká cena. Nevýhodou je zatížení systémového CPU při obsluze diskových polí.

Hardwarová řešení

Pro hardwarová řešení diskových polí slouží speciální hardware, který můžeme nazývat *řadič diskových polí*. Vytvoření nebo úpravu konfigurace RAIDu, provedeme buď při startu operačního systému, kdy vstupujeme do konfiguračního menu řadiče. Další možností jak nastavit pole je přes speciální dodávané aplikační vybavení řadiče. Toto vybavení nám také pomáhá hlídat stav pole.

Výhodou hardwarových diskových polí je fakt, že řadiče mají vlastní procesor pro výpočet paritních dat, což znatelně ulehčuje práci systémovému procesoru. Řadič zajišťuje rozložení výkonu v rámci zvolené úrovně RAID na fyzické disky. Dále řeší sestavení, obnovu a rozšiřování diskového pole v případě přidání fyzického disku. Také obsahují speciální paměť takzvanou **read/write cache** obvykle o kapacitě 64 MB až 512 MB, do které jsou ukládána data před fyzickým zápisem na disky a v případě čtení jsou zde uloženy před-načtené bloky dat, u kterých se předpokládá, že o ně systém požádá v rámci čtení většího bloku dat. Navíc paměť může být zálohována speciální baterií pro případ výpadku napájení. Operační systém tedy není zatížen žádnou ze služeb provozu pole a tak za diskové pole neplatíme výkonem systémového procesoru a takový řadič se pak stává zcela nezávislým na operačním systému a je možné jej provozovat bez nutnosti podpory ovladačů od výrobce hardware.

Nevýhodou však bývá, že dané disky jsou omezené na jeden typ diskového pole. Další nevýhodou může být cena, která se pohybuje na spodní hranici kolem 10 tisíc Kč. (3)

HW akcelerátory

Dále se prodávají speciální karty do PCI slotu základní desky počítače, které jsou jakési **hardwarové akcelerátory** pro RAID pole. Toto hardwarové diskové pole je ve skutečnosti spíše softwarové, jelikož nabízejí pouze jednoduchou RAID utilitu na definici RAID pole, nicméně bez ovladačů a podpory operačního systému z toho opravdové diskové pole nikdy nevznikne. Integrovaný RAID přímo na desce tedy není nic jiného než marketingový tah.

2 Operační systém Linux

V této kapitole bude představen samotný operační systém Linux, stručně popsány vlastnosti systému a vybrány populární linuxové distribuce.

„GNU/Linux nebo jen krátce Linux je v informatice označení pro operační systém založený na principech unixových systémů (původně pouze jeho jádro). Linux je šířen v podobě distribucí, které je snadné nainstalovat nebo přímo používat (tzv. Live CD). Zároveň se díky použitým licencím jedná o volně šiřitelný software, takže je možné ho nejen volně používat, ale i dále upravovat a distribuovat (kopírovat, sdílet). Tím se odlišuje od proprietárních systémů (např. Microsoft Windows či Mac OS X), za které je nutné platit a dodržovat omezující licence.“ (4)

„Operační systém Linux používá unixové jádro, které vychází z myšlenek Unixu a respektuje příslušné standardy *POSIX* a *Single UNIX Specification*. Název je odvozen z křestního jména jeho tvůrce Linuse Torvaldse a koncovka písmenem *x* odkazuje právě na Unix, podobně jako XENIX, Ultrix, IRIX, AIX a další.“ (4)

Linuxové distribuce

Operační systém Linux je šířen pomocí *distribucí*, které obsahují vše potřebné pro fungování systému. Distribucí existuje hned několik, v minulosti platilo, že skoro každá větší firma vyvíjela svojí verzi operačního systému Linuxu a to vedlo později k takzvaným *linuxovým válkám*. Každá firma chtěla prosadit svoje standardy, což bylo obtížné, proto vznikl standard POSIX. Jeho úkolem bylo vytvořit jednotné rozhraní, které mělo zajistit přenositelnost programů (aplikací) mezi různými hardwarovými platformami.

V současnosti jsou distribuce sestavovány jednotlivci, týmy dobrovolníků, ale i komerčními firmami. Distribuce zahrnuje jádro, další systémový a aplikační software, grafické uživatelské rozhraní (X.org, KDE, GNOME atd.). Distribuce mají různá zaměření, například výběr obsažených programů, podpora určité počítačové architektury, použití ve vestavěných systémech atd. V současné době existuje kolem 450 různých distribucí. (4)

„Mezi nejznámější distribuce Linuxu patří: Arch Linux, Danix (česká distribuce), Debian, Fedora (nástupce Red Hat Linuxu), Red Hat Enterprise Linux (vychází z Fedory), Gentoo, Greenie, (slovenská distribuce), Knoppix, Mandriva (dříve Mandrake), Linux Mint (vychází

z Ubuntu), Slackware, Slax (česká live distribuce), Source Mage, SUSE, Ubuntu, Kubuntu (derivát Ubuntu).“ (4)

2.1 Instalace OS Linux a příprava RAID polí

Instalace Linuxu se liší podle zvolené linuxové distribuce. Většina distribucí nabízí textovou i grafickou verzi instalace, kterou obvykle zvládne i začátečník.

Instalaci můžeme provést přímo z instalačního média (při této variantě se ovšem nainstalují aplikace ve verzi, která byla k dispozici v době vydání distribuce). Případně v průběhu instalace můžeme z internetu doinstalovat doplňující software, či stáhnout nejnovější verzi systému. Některé distribuce lze také instalovat z jiného běžícího systému (jiné distribuce Linuxu). (4)

Na výběr máme tři typy, jakým způsobem lze nainstalovat operační systém Linux:

1. Instalovat Linux jako jediný operační systém.
2. Instalovat Linux na jiný disk, než je současný operační systém.
3. Instalovat Linux na disk společně s jiným operačním systémem.

V této práci byla použita instalace na samostatný disk. Pevný disk byl rozdělen na primární diskový oddíl a následně provedeno formátování oddílu. Jako souborový systém byl vybrán žurnálovací souborový systém ext4. Nakonec byl vytvořen odkládací virtuální diskový oddíl **swap** ve velikosti dvojnásobku fyzické velikosti RAM paměti, která činila 4 GB. Mezi další kroky instalace patřily: výběr jazyka, výběr země, typ rozložení klávesnice. Další volbou je nastavení středoevropského času a možnost výběru některých aplikací. Velice důležitou volbou je zadání jména uživatele a jeho hesla, popř. hesla superuživatele. Tato položka je v Linuxu velmi důležitá, zejména pro správu systému.

První co musíme udělat, je vybrat některou z linuxových distribucí. Pro účely této práce byla vybrána distribuce *Ubuntu* ve verzi *14.04*, přesný výpis verze zjistíme pomocí příkazu:

```
uname -a [1]
Linux jan-ruzicka 3.13.0-37-generic #64-Ubuntu SMP Mon Sep 22 / 21:28:38
UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
```

[1] Příkaz je vyznačen tučným písmem a výpis je normálním písmem.

Parametry testovací sestavy, na které byly provedeny testy měření výkonu softwarových diskových polí, je následující:

Hardware:

Processor: AMD Phenom II X4 940 @ 3.00GHz (4 Cores), Motherboard: ASUS M4A78-E, Chipset: AMD RS780 + SB7x0/SB8x0/SB9x0, Memory: 4096MB, Disk: 640GB Western Digital WD6401AALS-0 + 80GB Seagate ST380013AS, Graphics: Sapphire AMD Radeon HD 4870 512MB, Audio: VIA VT1708S, Monitor: S24C350, Network: Qualcomm Atheros AR8121/AR8113/AR8114

Software:

OS: Ubuntu 14.04, Kernel: 3.13.0-37-generic (x86_64), Desktop: Unity 7.2.2, Display Server: X Server 1.15.1, Display Driver: radeon 7.3.0, Compiler: GCC 4.8.2, File-System: ext4, Screen Resolution: 1920x1080

3 Architektura HDD, vytváření oddílů a souborové systémy

Architektura fyzických pevných disků

„Pevný disk (zkratka HDD, anglicky Hard Disk Drive) je zařízení, které se používá v počítačích a ve spotřební elektronice k dočasnému nebo trvalému uchovávání většího množství dat pomocí magnetické indukce. Pevný disk je dnes běžnou součástí počítačů, kapacity se pohybují obvykle od 20 do 2 000 GB. Jde o pevně uzavřenou jednotku, která uvnitř obsahuje několik nad sebou umístěných rotujících záznamových ploten. Tyto plotny se otáčejí konstantní rychlostí po celou dobu, co je pevný disk HDD připojen ke zdroji elektrického napájení. Aby se plotny točily, nemusí se na ně přitom neustále číst nebo zapisovat. Rychlé otáčení v okolí ploten vytváří tenkou vzduchovou vrstvu, na níž se pohybují čtecí/zapisovací hlavy. Vzdálenost hlav od disku je asi 0,3 až 0,6 mikronu. Pro každý povrch plotny má pevný disk elektromagnetickou čtecí a zápisovou hlavu s mikroskopickou cívkou. Hlavy jsou umístěny na jednom společném rameni a pohybují se zároveň.“ (5)

Nejčastější příčiny poruch HDD

Pevný disk je velmi náchylný k poruchám a chybám, kvůli složité konstrukci, vysoké provozní rychlosti a náchylnosti ke změně polohy. Při používání pevného disku je nutné dodržet důležitou zásadu – nehýbat s ním. (5)

- „Otřesy a nevhodná manipulace s diskem v zapnutém stavu.
- Náraz po pádu disku na zem.
- Elektrický zkrat, výpadky, přepětí elektrické energie v síti.
- Fyzické poškození pouzdra disku.
- Mechanická závada materiálu v konstrukci disku.
- Otřesy zapnutého notebooku po nečekaném probuzení z režimu spánku.
- Prudké pootočení nebo posunutí zapnutého disku, externího disku, notebooku.
- Zakopnutí o kabel a převrhnutí zapnutého (externího) disku.
- Zestárnutí magnetického povrchu disku.
- Smazání dat z disku omylem, nechtěné zformátování disku.
- Porušení, popraskání protiprachové ochrany.
- Odpojení disku od počítače v průběhu kopírování.
- Polití disku vodou, čajem, kávou nebo jinou tekutinou.“ (5)

Diskové plotny

„Plotny se rychle otáčejí (udává se počet otáček za minutu). V běžných discích plotny rotují rychlostí 7 200 otáček za minutu, vyšší třída disků do pracovních stanic se točí rychlostí 10 000 otáček za minutu a u některých serverových disků i 15 000 otáček za minutu. Při 7 200 otáček za minutu je obvodová rychlost plotny přes 100 km/h (pro 3,5 palcový disk). Otáčky disku společně s hustotou záznamu a rychlostí vystavovacího mechanismu určují celkový výkon disku. Čím rychleji se plotny otáčí tím víc na ně působí odstředivá síla a proto se někteří výrobci u disků s 10–15 000 otáček za minutu uchylují k 2,5" verzím, kde je síla menší, a tak jsou materiály méně namáhány. Průměry ploten u 3,5" HDD jsou standardně 9,5 cm, ale mohou být u serverových disků 6,5 cm, 7 cm i 8,4 cm. Plotny se liší i svojí výškou, od cca 0,4 mm po 2,0 mm.“ (6)

Operace nutné pro čtení nebo zápis dat:

1. „vystavit čtecí hlavu na správnou pozici,
2. vyčkat na utlumení rozkmitu způsobeném setrvačností hlav (vystavení trvá řádově milisekundy [ms]),
3. vyčkat na pootočení disku na místo, od kterého začne čtení nebo zápis (tzv. latence).“ (6)

„Průměrný střední čas, za který je disk připraven číst nebo zapisovat data, se označuje jako *přístupová doba*. V současné době je okolo 8,5 ms. U disků s 15 000 otáček za minutu je to pod 4 ms.“ (6)

Partition tabulka

„Na začátku pevného disku (v jeho úplně prvním sektoru) je uložena oblast, označovaná jako *Partition tabulka* (někdy také jako *Master Boot sektor*). První polovinu této tabulky zabírá krátký program, který slouží k zavedení operačního systému do paměti po zapnutí počítače. Právě díky existenci tohoto kódu se *Partition tabulka* stává cílem útoku některých počítačových virů a místem umístění viru. Ve druhé části jsou pak data, která popisují, jak je fyzický disk rozdělen na logické disky. Partition tabulka existuje na fyzickém disku vždy pouze jedna – nezávisle na tom, na kolik logických disků je fyzický disk rozdělen (výjimkou jsou tzv. *extended partition tabulky*).“ (7)

Typy diskových oddílů

Diskové oddíly lze rozdělit na tři typy: **primární**, **rozšířený** a **logický**. Toto rozdělení vyplívá z použití *MBR* (master boot record) oblasti disku.

MBR

„Master Boot Record je prvních 512 bytů na začátku každého disku. Obsahuje zavaděč systému a tabulku rozdělení disku (partition table). Někdy také volitelně identifikátor disku. Hlavní úlohou MBR je načíst do paměti *boot* sektor oddílu, který je označen jako aktivní v partition table.“ (8)

Primární diskové oddíly

„Kvůli omezení BIOSu počítače lze nastavit pouze 4 primární diskové oddíly. Jsou to oddíly zapsané v MBR (master boot record) oblasti disku.“ (8)

Rozšířené diskové oddíly

„Rozšířené (extended) oddíly byly zavedeny kvůli potřebě dělení disků na více než 4 oddíly se zachováním zpětné kompatibility. Rozšířený diskový oddíl nelze přímo používat, ale může obsahovat větší množství logických oddílů. Je to v podstatě *kontejner* pro logické oddíly.“ (8)

Logické diskové oddíly

„Logický oddíl je oddíl umístěný v rozšířeném diskovém oddílu. Logické oddíly musí na sebe navazovat. Každý logický oddíl obsahuje odkaz na jeho následující oddíl.“ (8)

Oddíl SWAP

„Každý proces operačního systému potřebuje pro svůj běh určité množství operační paměti (RAM). V případě nedostatku této paměti se momentálně nepotřebná data z RAM (např. blokované procesy) přesouvají na disk, aby uvolnili místo běžícímu procesu. Data se odkládají na speciálně vyhrazený oddíl – *swap*.“ (8)

Důvody dělení disků na oddíly

Výhody:

- „Správně rozdělený disk na oddíly je více přehledný.
- Šetření místem při výběru správného souborového systému (např. pro malé soubory). Velký oddíl u FAT32 zvětšuje clustery (nejmenší jednotka disku).
- Lepší administrace a údržba disku (defragmentace, správa místa, formátování).
- Bezpečnost dat, ochrana a izolace uživatelských dat při havárii operačního systému a oddílů.
- Možnost koexistence více operačních systému.
- Možnost využívat výhody jednotlivých souborových systémů.“ (8)

Nevýhody:

- „Velmi obtížné přerozdělování velikosti oddílů.
- Problém s nedostatkem místa např. na systémovém disku.“ (8)

Rozdíly ve fyzickém umístění diskových oddílů

V průběhu měření byl zjištěn fakt, který ovlivňuje rychlost čtení a zápisu pevného disku a tím je fyzická poloha diskových oddílů. Pokud byl vytvořen diskový oddíl na začátku pevného disku, tak rychlost čtení i zápisu byla rychlejší než při tvorbě diskového oddílu na konci pevného disku. Může za to obvodová rychlost, která je větší u středu, než na vnějšku pevného disku. Jelikož diskové plotny mají kruhový tvar. Tento fakt zapříčinil, že všechny typy zapojení diskových polí, byly prováděny samostatně. Tedy diskové oddíly jakož to členy RAID, byly vždy vytvářeny na začátku pevného disku.

3.1 Popis a vlastnosti souborových systémů ext4 a zfs

„Souborový systém nebo také **filesystem** je v informatice označení pro způsob (systém) organizace dat ve formě souborů tak, aby bylo možné k těmto datům snadno přistupovat. Souborové systémy jsou vlastně takové sklady pro data, které jsou uloženy na vhodném typu elektronické paměti. Tyto paměti mohou být v podobě pevného disku, či SSD disku uloženy přímo v počítači, nebo ve formě přenosných paměťových karet a USB disků.“ (9)

„Souborový systém zajišťuje ukládání a čtení dat, která mohou být hierarchicky ukládána v podobě adresářů a souborů, které jsou určeny svým jménem. Z toho vyplývá, že dva soubory nebo podadresáře umístěné ve stejném adresáři nesmí mít stejné jméno.“ (9)

„Informace uložené v systému souborů dělíme na *metadata* a *data*. *Metadata* popisují strukturu systému souborů a nesou další doplňující informace, jako je velikost souboru, čas poslední změny souboru, čas posledního přístupu k souboru, vlastník souboru, oprávnění v systému souborů, seznam bloků dat, které tvoří vlastní soubor atd. Pojmem *data* pak míníme vlastní obsah souboru, který můžeme přečíst, když soubor otevřeme.“ (9)

Omezení souborových systémů

Různé souborové systémy mohou mít různá omezení, například:

- „velikost paměťového média, kterou je daný systém schopen pokrýt,
- délka souboru,
- délka jména souboru,
- počet zanořených podadresářů,
- podporovaná znaková sada.“ (10)

Žurnálovací souborové systémy

„Zápis dat a metadat do systému souborů probíhá v několika krocích. Proto nejsou data a metadata v každém okamžiku *konzistentní*. Dojde-li v takové chvíli k havárii počítače (např. výpadek elektrického proudu, chyba hardware, software a podobně), zůstane systém souborů v *nekonzistentním* stavu. Z tohoto důvodu je při dalším startu operačního systému vhodné, aby byla provedena kontrola a nekonzistentní data byla opravena. K tomu může dojít automaticky (např. v Linuxu nebo ve Windows 95 a novějších systémech) nebo je nutné spustit kontrolu ručně (systémy DOS).“ (10)

„Celková kontrola systému souborů a všech vazeb mezi daty a metadaty je časově velmi náročná operace, při které navíc může dojít ke zbytečné ztrátě již částečně zapsaných informací. Proto jsou moderní systémy souborů rozšířeny o *žurnálování*, které umožňuje po havárii rychlou opravu eventuálních nekonzistencí. Žurnál je obvykle realizován jako cirkulární **buffer** a jeho účelem je uchovávání chronologického záznamu prováděných operací, do kterého se zapisují všechny prováděné činnosti. Pokud dojde např. k výpadku napájení, je po restartu nekonzistence opravena návratem do předchozího zaznamenaného stavu za pomoci záznamů

z žurnálu. Mezi žurnálovací souborové systémy patří např. NTFS, HFS+, ext3, ext4, XFS nebo ReiserFS.“ (10)

Souborový systém Ext4

„*Ext4* (fourth extended filesystem) jedná se o žurnálovací souborový systém vyvinutý pro linuxové jádro, jehož vývoj začal 10. října 2006 jako zpětně kompatibilní nástupce *ext3* se všemi jeho výhodami. Do jádra byl začleněn nejprve jako vývojový a dne 11. prosince 2008 byl v jádře verze 2.6.28 označen za stabilní a vhodný k běžnému užívání.“ (10)

„Ext4 přinesl mnoho novinek typických pro moderní souborové systémy, může podporovat svazky až o velikosti *1 EiB* a soubory s maximální velikostí *16 TiB*, podporuje **extenty** a odstraňuje limit původního systému ext3, jenž mohl obsahovat v adresáři maximálně 32 768 podadresářů. Tento limit byl v ext4 navýšen na 64 000 a pomocí rozšíření **dir_nlink**, může tuto hranici dále prolomit (ačkoliv to zastaví zvyšování počtu odkazů z rodiče). Tato vlastnost je implementována do jádra Linuxu od verze 2.6.23. S původním ext3 je kompatibilní zpětně i dopředně (pod typem ext4 je možné připojit souborový systém ext3, a naopak pod typem ext3 lze připojit ext4 svazek, pokud nepoužívá *extenty*).“ (9)

„*Extenty* nahrazují tradiční schéma blokového mapování, které využívají předchozí systémy ext2 a ext3. Extent je rozsah navazujících fyzických bloků (resp. alokačních jednotek), který zlepšuje výkon při práci s velkými soubory a zmenšující fragmentaci. Jeden extent se tak může stát v systému souborů ext4 alokační jednotkou o velikosti až 128 MiB souvislého místa na disku, místo mnoha jednotlivých datových bloků o standardní velikosti 4 KiB.“ (10)

„Mezi další vlastnosti patří podpora nanosekundových časových razítek a pre-alokace i zpožděná alokace místa pro soubory. Ovladač ext4 obsahuje (stejně jako předchozí verze souborového systému) nástroje, které omezují fragmentaci již při ukládání dat na disk. Nově však ext4 umožňuje nasazení online *defragmentátoru* na úrovni souborů nebo celého souborového systému. Díky označování nepoužitých oblastí disku mohou nástroje pro opravu systému (fsck) pracovat rychleji než na ext3.“ (10)

Souborový systém ZFS

„*ZFS* (Zettabyte File System) je kombinovaný souborový systém a správce logických svazků vyvinutý společností Sun Microsystems pro operační systém Solaris. ZFS obsahuje funkce pro ověřování integrity dat, podporu pro uchovávání velkých objemů dat, integraci konceptů

souborového systému a správy svazků, zaznamenávání a ukládání aktuálního stavu systému (jako bod obnovy u Windows), ověřování integrity dat a jejich opravy za chodu, RAID-Z a přirozená podpora NFSv4 ACLs. ZFS umí uložit až 256 kvadrilionů Zettabytů (1 ZB = 270 bytů). ZFS je implementováno jako open-source software, licencováno pod Common Development and Distribution License (CDDL).“ (11) ZFS byl do 20. září 2011 registrovanou obchodní značkou firmy Oracle.

„ZFS je běžnou součástí *OS xBSD* a pokračovatelů projektu OpenSolaris. ZFS je založena na několika důležitých technologiích, které ho předurčují spíš k využití na serverech a datových úložištích. Mnohé z toho je ale možné výhodně využít i na desktopových systémech. ZFS zachází s disky odlišně od běžných souborových systémů tím, že využívá celkovou kapacitu všech přiřazených disků – tzv. *pool*. V poolu je možné vytvářet tzv. *datasety*, což je určitá obdoba adresářů. Tím se vlastně chová podobně jako LVM, ale pro svou činnost nevyžaduje a nevytváří žádné diskové oddíly. V systému je možné vytvořit více poolů, které mohou pracovat v různých módech (módem je myšleno ZFS Mirror (zrcadlení*)* nebo obdoba RAID 5 – RAIDZ). Dataset může využít celou kapacitu příslušného poolu. Pokud je to nutné nebo vhodné, lze omezit kapacitu datasetů pomocí kvót.“ (11)

„ZFS umožňuje transparentní kompresi jednotlivých datasetů. ZFS používá pro zápis dat metodu *COW* (copy-on-write). To znamená, že systém nikdy nepřepisuje starou verzi dat. Pokud je třeba data změnit, zapíšou se do nového bloku. Pokud zápis proběhne bez problémů, změní se ukazatel a pak teprve dojde ke smazání staré verze. Pro ukládání dat se používají bloky s proměnlivou velikostí. To je vhodné třeba při zapnuté kompresi. Naopak je nutné dbát, aby zaplnění daty nepřekročilo 80 % celkové kapacity. V tomto případě by mohlo dojít k drastickému zpomalení zápisů na disk. Pro zachování konzistence dat používá ZFS kontrolní součty uložené přímo na disku. Pokud se narazí na nekonzistentní data, provede systém automaticky jejich opravu bez zásahu administrátora. Jednoduchým způsobem lze v systému periodicky provádět kontrolu a případnou opravu nekonzistencí.“ (11)

Charakteristika:

- „automatická kontrola a oprava konsistence zapsaných dat,
- téměř nepřekonatelná horní kapacitní hranice (až 16 EB = 16 mld. GB),
- zvýšení maximálního počtu souborů v jedné složce na úroveň > 7,2×1 016,
- proměnlivá velikost jednotlivých bloků,
- implementace transparentní komprese dat, což výrazně urychluje zápis.“ (12)

„Aby ZFS optimalizoval svou práci, používá dynamické rozdělování zátěže na všechny jednotky. Jakmile je přidán další disk, dojde automaticky k přerozdělení dat mezi disky, takže je pak ihned používán naplno i nový disk. Každému svazku ve stromu je pak přidělena taková kapacita, kterou potřebuje. Zároveň ale používá výkon celé diskové kapacity. Aby nedocházelo k přeplňování disků, je k dispozici samozřejmě také systém kvót. Protože je ZFS připraven na velké zátěže, implementuje I/O reordering, který dovoluje optimalizovat jednotlivé operace. To má ohromný vliv na celkový výkon ZFS.“ (13)

4 Disková pole a souborové systémy v Linuxu

Instalace balíčků mdadm a zfsutils

Pro vytvoření softwarových diskových polí na souborovém systému *ext4* byla použita utilita *mdadm*, kterou lehce přidáme do systému pomocí terminálu příkazem:

```
sudo apt-get install mdadm [2]
```

Pro souborový systém *zfs*, byla využita utilita *zfsutils*. Kterou přidáme do systému obdobně.

4.1 Disková pole na souborovém systému EXT4

RAID pole lze v Linuxu vytvářet pomocí MD subsystému. Pro administraci RAID polí slouží v Linux příkaz *mdadm*. Jádro Linuxu udržuje informace o aktuálním stavu MD subsystému v souboru */proc/mdstat*. V níže uvedeném příkladu je v systému pole RAID 0, které se skládá ze dvou disků (sdc1 a sdd1). Pole je plně funkční (oba disky jsou ve stavu *U*, což znamená *Up*).

```
cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 sdc1[0] sdd1[1]
19542976 blocks [2/2] [UU]
unused devices: <none>
```

Konfigurace RAIDu verze 0.42 používala konfigurační soubor */etc/mdtab.* Na rozdíl od verze 0.90, kde se konfigurace nachází v */etc/raidtab* a obsahuje následující direktivy:

- „**raiddev**: touto direktivou definice pole začíná, následuje označení pole. Svazky softwarového RAIDu se označují md1 až mdX.
- **raid-level**: následuje direktivu raiddev, uvádíme zde typ pole (-1 = linear, 0 = RAID 0, 1 = RAID 1, 5 = RAID 5).
- **persistent-superblock**: tato direktiva se týká kompatibility se starší verzí RAIDu.
- **chunk-size**: velikost *stripu*, maximální velikost je 4 MB (což je dáno konstantou MAX_CHUNK_SIZE ovladače), udává se v kB.
- **nr-raid-disks**: zde uvádíme kolik diskových oddílů bude součástí pole, maximální počet oddílů je 12 v jádrech 2.2.x, ale jsou k dispozici záplaty, které tento limit podstatně zvyšují.
- **nr-spare-disks**: počet rezervních disků v poli.

[2] Příkaz bez výpisu je naformátován normálním písmem.

- **device** *jméno_oddílu*: tato direktiva následovaná jednou z direktiv *raid-disk*, *spare-disk*, *parity-disk* nebo *failed-disk*. Tyto direktivy deklarují příslušné oddíly, které budou součástí pole.
- **raid-disk**: tento oddíl bude aktivním oddílem.
- **spare-disk**: tento oddíl bude sloužit jako rezervní.
- **parity-disk**: tento oddíl bude sloužit jako paritní disk (u zapojení RAID 4).
- **failed-disk**: tento oddíl bude při inicializaci pole přeskočen, má význam pouze při sestavování pole v degradovaném stavu.
- **parity-algorithm**: specifikuje schéma rozložení parity u RAIDu 5. Možnosti jsou:
    - *left-symmetric* (které je obecně ze všech nejrychlejší),
    - *right-symmetric*,
    - *right-asymmetric*.“ (1)

Konfigurace uložená v souboru **/etc/raidtab**. Příklad konfigurace pole *Linear* Tabulka 2. Mějme pole typu Linear složené ze dvou oddílů sda1 a sdb1:

Tabulka 2 – Konfigurace pole Linear

Zdroj: (1)

Raiddev	**/dev/md0**	**Popis**
raid-level	-1	úroveň RAIDu
persistent-superblock	1	povinný parametr pro autodetekci pole jádrem
nr-raid-disk	2	počet disků pole
nr-spare-disks	0	počet záložních disků
device	/dev/sda1	první diskový oddíl tvořící pole
raid-disk	0	číslo oddílu v poli
device	/dev/sdb1	druhý diskový oddíl tvořící pole
raid-disk	1	číslo oddílu v poli

Příklad konfigurace *RAID 1* Tabulka 3. Mějme diskové pole RAID 1 složené ze dvou oddílů sda1 a sdb1:

Tabulka 3 – Konfigurace RAID 1

Zdroj: (1)

Raiddev	/dev/md1	Popis
raid-level	1	úroveň RAIDu
nr-raid-disks	2	počet disků pole
nr-spare-disks	0	počet záložních disků
persistent-superblock	1	povinný parametr pro autodetekci pole jádrem
device	/dev/sda1	první diskový oddíl tvořící pole
raid-disk	0	číslo oddílu v poli
device	/dev/sdb1	druhý diskový oddíl tvořící pole
raid-disk	1	číslo oddílu v poli

Konfigurace RAID 0

Vytvoření diskového pole a souborového systému

K vytvoření diskového pole s dvěma zařízeními *sdc1* a *sdd1*, použijeme příkaz:

```
sudo mdadm –create /dev/md0 –level=0 –raid-devices=2 /dev/sdc1 /dev/sdd1
```

Na vzniklém diskovém poli vytvoříme souborový systém ext4:

```
sudo mkfs –t ext4 /dev/md0
```

Pole lze zastavit pouze v případě, že se nepoužívá, nesmí být připojeno příkazem **mount**. V případě zrušení pole aplikujeme příkazy pro vymazání *metadat* na všechny členy diskového pole. Metadata jsou zapsána na oddíl při vytváření pole. Pokud je potřeba, aby pole již nebylo automaticky detekováno, je možné následujícími příkazy metadata odstranit (vynulovat):

Zastavení a zrušení diskového pole

Pro zastavení diskového pole, slouží příkaz:

```
sudo mdadm --stop /dev/md0
```

Pro zrušení celého diskového pole použijeme příkazy:

```
sudo mdadm --zero-superblock /dev/sdc1
sudo mdadm --zero-superblock /dev/sdd1
```

Konfigurace RAID 1

Vytvoření diskového pole a souborového systému

K vytvoření diskového pole s dvěma zařízeními sdc1 a sdd1, použijeme příkaz:

```
sudo mdadm –create /dev/md1 –level=1 –raid-devices=2 /dev/sdc1 /dev/sdd1
```

Na vzniklém diskovém poli vytvoříme souborový systém ext4:

```
sudo mkfs –t ext4 /dev/md1
```

Zastavení a zrušení diskového pole

Pro zastavení diskového pole, slouží příkaz:

```
sudo mdadm --stop /dev/md1
```

Pro zrušení celého diskového pole použijeme příkazy:

```
sudo mdadm --zero-superblock /dev/sdc1
sudo mdadm --zero-superblock /dev/sdd1
```

Konfigurace RAID 5

Vytvoření diskového pole a souborového systému

Zde se jedná o vytvoření RAID 5 pole se třemi zařízeními (sdc1, sdd1 a sde1), můžeme použít následující příkaz:

```
sudo mdadm --create /dev/md5 --level=5 --raid-devices=3 /dev/sdc1 \
/dev/sdd1 /dev/sde1
```

Pokud chceme vytvořit diskové pole, ale nemáme k dispozici všechny disky. Můžeme na místo chybějícího disku použít klíčové slovo **missing** (chybějící): (14)

```
sudo mdadm --create /dev/md5 --level=5 --raid-devices=3 /dev/sdc1 \
missing /dev/sde1
```

„Ačkoliv je možné sestavit pole v degradovaném režimu s chybějícími redundantními aktivními zařízeními, tento postup se nedoporučuje. Zejména pak, pokud je potřeba na takové pole přesunout data ještě před tím, než obsluha doplní zbývající zařízení.“ (14)

„Na vytvořeném poli je možné vytvořit souborový systém. Není nutné čekat, až proběhne rekonstrukce pole. V případě vytváření souborového systému pro běžnou instalaci (tj. cca 30 GB) je možné nechat počet vytvořených **inodů** na příkazu **mkfs**. V případě, že vytváříme souborový systém pro domácí adresáře, ve kterých nebude mnoho malých souborů, je možné počet *inodů* zmenšit (přepínačem -N). Kontrola pomocí příkazu **fsck** pak probíhá mnohem rychleji. Tuto negativní vlastnost odstraňuje souborový systém ext4.“ (15)

Pro tvorbu souborového systému použijeme příkaz:

```
sudo mkfs -t ext4 /dev/md5
```

Zastavení a zrušení diskového pole

Pro zastavení diskového pole, slouží příkaz:

```
sudo mdadm --stop /dev/md5
```

Pro zrušení celého diskového pole použijeme příkazy:

```
sudo mdadm --zero-superblock /dev/sdc1
sudo mdadm --zero-superblock /dev/sdd1
sudo mdadm --zero-superblock /dev/sde1
```

Výpis vlastností pole

Výpis vlastností diskového pole, můžeme zjistit pomocí příkazu *mdadm,* s použitým přepínačem –D. Výpis může být následující:

```
mdadm -D /dev/md0/dev/md0:
Version : 1.0
Creation Time : Thu Aug 18 11:36:07 2011
Raid Level : raid1
Array Size : 20980784 (20.01 GiB 21.48 GB)
Used Dev Size : 20980784 (20.01 GiB 21.48 GB)
Raid Devices : 2
Total Devices : 2
Persistence : Superblock is persistent
Intent Bitmap : Internal
```

```
Update Time : Sat Sep  3 16:37:23 2011
State : active
Active Devices : 2
Working Devices : 2
Failed Devices : 0
Spare Devices : 0
Name : localhost.localdomain:0
UUID : f8665378:b79d25a3:90cd2d67:750c7a5b
Events : 338168
Number   Major   Minor   RaidDevice State
0        8       1        0         active sync   /dev/sda1
2        8       17       1         active sync   /dev/sdb1
```

Ve výše uvedeném výpisu je vypsána informace o poli typu RAID 1, které se skládá ze dvou oddílů (/dev/sda1, /dev/sdb2). **Major-minor** (čísla speciálních zařízení jednotlivých oddílů). Pokud **minor number** je nula, tak je pole automaticky sestaveno jako /dev/md0. Položka **update time** je používána při sestavování pole, kdy je potřeba zjistit, které členy pole obsahují platná data (novější záznam o poslední aktualizaci pole). Pokud jsou tedy u dvou členů RAID 1 časy různé, použije se při sestavení pouze novější člen a starší člen je poté nutno s polem synchronizovat. (15)

„Tyto informace se zapisují do *superbloku* každého členu pole (dříve se superbloky nepoužívaly, takže bylo například obtížné pole po změně zapojení disků správně složit). Informace o poli musí být konzistentní ve všech členech pole.“ (15)

„Nástroj *mdadm* umožňuje monitorovat stav pole a v případě, že dojde k rozpadnutí pole (které však může dále pokračovat v činnosti), odešle upozornění administrátorovi systému. Administrátor se obvykle pokusí zahájit rekonstrukci pole pouhým přidáním vypadlého oddílu znovu do pole, případně musí vyměnit havarovaný disk, vytvořit znovu oddíly a teprve pak pole nechat zrekonstruovat.“ (15)

4.2 Disková pole na souborovém systému ZFS

„Práce s disky se díky ZFS velmi zjednodušuje, už nejsou potřeba diskové oddíly, formátování disků, vytváření nových souborových kapacit a podobně. Vše se zjednodušuje na dva příkazy: **zpool** a **zfs**.“ (13)

„Bohužel *ZFS* zatím neumí fungovat jako kořenový souborový systém, a tak z něj nemůžeme bootovat. Podle *Sunu* se této funkce ale také v dohledné době dočkáme. ZFS je už dnes velmi

použitelný souborový systém s mnoha výhodami proti běžným souborovým systémům. Jeho možnosti jsou široké, ale zároveň klade minimální nároky na obsluhu a administraci. Zdá se, že většina vývojářů už pochopila jeho sílu a portování na další systémy snad na sebe nenechá dlouho čekat.“ (13)

Pro dosažení stejných výchozích podmínek souborových systémů *ext4* a *zfs*, musíme přenastavit parametr **relatime**[3] na hodnotu **on**. Jelikož ve výchozím nastavení je parametr nastaven na *relatime=off*.

ZFS pool s jedním HDD

V tomto typu zapojení byl použit pouze jeden pevný disk. Konfigurace a základní příkazy budou popsány níže.

Vytvoření diskového pole a souborového systému

K vytvoření nového *poolu* budeme potřebovat jeden jediný příkaz:

```
sudo zpool create myPool sdc1
```

Tímto byl na disku vytvořen jeden pool s názvem *myPool*. Můžeme si na něm vytvořit další souborové systémy s vlastní hierarchií. Například vytvoříme adresář */home*:

```
sudo zfs create myPool/home
```

Ten pak nastavíme tak, aby se automaticky připojoval do námi zvoleného *mountpointu* (přípojného bodu):

```
sudo zfs set mountpoint=/media/home myPool/home
```

Takhle jednoduché to je. Takto si můžeme vytvořit další a další souborové systémy. Například můžeme nastavit kvótu na 20 GB, čímž nastavíme pevnou velikost poolu. Dále můžeme také zapnout kompresi:

```
sudo zfs set quota=20g myPool/home
sudo zfs set compression=on myPool/home
```

[3] Nastavení typu aktualizace přístupového času k souboru relativně k času změny.

Výpis diskového pole a jeho stavu

Pro přehled informací o stavu vytvořeného poolu, můžeme využít příkaz:

```
sudo zpool list
NAME SIZE USED AVAIL CAP HEALTH ALTROOT
myPool 35.2G 6.39G 28.9G 18% ONLINE -
```

Jednotlivé položky ve výpisu: název poolu (*NAME*), celkovou velikost poolu (*SIZE*), součet alokované velikosti všech datasetů a interních metadat (*USED*), součet nealokovaného prostoru v poolu (*AVAILABLE*), procentuální vyjádření naplněnosti poolu (*CAP*) a aktuální stav poolu (*HEALTH*). (16)

„Důležitým příkazem je i **zpool status** pro vypsání stavu poolu. Stav poolu je určen ze stavu jeho *top-level* zařízení. Platí, že pokud jsou všechny virtuální zařízení (*vdevs*) **Online**, tak je celý pool ve stavu *Online*. Pokud je některé virtuální zařízení ve stavu **Degraded** (degradovány) nebo **Unavailable** (nedostupné), tak pool jako takový je označen jako *Degraded* (degradován). Pokud je top-level zařízení označené jako **Faulted** (vadné) nebo **Offline**, tak celý pool je označen jako *Faulted* (vadný).“ (16)

Stav celého našeho poolu zjistíme příkazem:

```
sudo zpool status –v myPool
```

Zrušení diskového pole

Pro zrušení celého poolu, použijeme příkaz:

```
sudo zpool destroy myPool
```

ZFS Mirror

Jedná se principiálně o totožné zapojení jako RAID 1. Následující příkazy představí vytvoření diskového pole, vytvoření souborového systému, zrušení pole a sledování stavu diskového pole. Pro zapojení ZFS Mirror použijeme dvě zařízení a to *sdc1* a *sdd1*.

Vytvoření diskového pole a souborového systému

Pro vytvoření diskového pole použijeme příkaz:

```
sudo zpool create myPool mirror sdc1 sdd1
```

Dále vytvoříme souborový systém *zfs*:

```
sudo zfs create myPool/mirror
```

Nakonec nastavíme přípojný bod, kam se bude automaticky připojovat diskové pole:

```
sudo zfs set mountpoint=/media/mirrorZFS myPool/mirror
```

Výpis diskového pole a jeho stavu

Pro výpis diskového pole slouží příkaz:

```
sudo zpool list
```

Pro výpis stavu a jeho statistik, použijeme příkaz:

```
sudo zpool stat
```

Zrušení diskového pole

Pro zrušení celého poolu, využijeme příkaz:

```
sudo zpool destroy myPool
```

ZFS Raidz

Toto zapojení se nejvíce funkčně podobá zapojení RAID 5. Následující příkazy představí vytvoření diskového pole, vytvoření souborového systému, zrušení pole a sledování stavu diskového pole. Pro zapojení ZFS Raidz použijeme tři zařízení *sdc1*, *sdd1* a *sde1*.

Vytvoření diskového pole a souborového systému

Pro vytvoření diskového pole použijeme příkaz:

```
sudo zpool create myPool raidz sdc1 sdd1 sde1
```

Dále vytvoříme souborový systém *zfs*:

```
sudo zfs create myPool/raidz
```

Nakonec nastavíme přípojný bod, kam se bude automaticky připojovat diskové pole:

```
sudo zfs set mountpoint=/media/raidzZFS myPool/raidz
```

Výpis diskového pole a jeho stavu

Pro výpis diskového pole slouží příkaz:

```
sudo zpool list
```

Pro výpis stavu a jeho statistik, použijeme příkaz:

```
sudo zpool stat
```

Zrušení diskového pole

Pro zrušení celého poolu, využijeme příkaz:

```
sudo zpool destroy myPool
```

5 Benchmark

Testovací program je v informatice chápán jako počítačový program, či sada programů nebo speciální operace, kterými je možno posoudit relativní výkonnost testovaného objektu. Obvykle provozuje řadu standardních testů, které opakuje pro dosažení co možná nejvěrnějších výsledků. **Benchmarking** je obvykle spojován s posuzováním charakteristik výkonu počítačového hardwaru, například výkon CPU při výpočtu matematické operace s plovoucí desetinou čárkou, výkon pevných disků, či výkon grafických karet a dalších. Srovnávací testy poskytují způsob porovnání vlastností různých subsystémů v odlišné systémové architektuře.

5.1 Popis a funkčnost benchmarku Bonnie++

Tato jednoduchá *utilita* testuje zvolený *souborový systém* tím, že na něj vytvoří různými způsoby poměrně velké soubory (ve výchozím stavu 2× velikost RAM po 1 GB, tedy 8 GB RAM = 16× 1 GB souborů), které následně čte a přepisuje. Toto všechno měří a počítá rychlosti jednotlivých operací, navíc v dalších testech umí vyvářet různé (volitelné) množství souborů a adresářů, ke kterým různě přistupuje (mazání a čtení). Otestuje tedy i schopnost souborového systému jak rychle umožňuje přistupovat k adresářům a souborům. Program je založen na předchůdci Bonnie, který napsal Tim Bray. Bonnie++ je běžně k dispozici z linuxových repozitářů, nebo přímo ze zdroje na domovské stránce programu. Na systémech vycházejících z operačního systému Debian, v případě této práce na linuxové distribuci Ubuntu, se lehce nainstaluje pomocí příkazu: (17)

```
sudo apt-get install bonnie++
```

„Pro minimalizaci vlivu ukládání souborů do mezi-paměti počítače, je třeba použít zkušebních datových souborů větších než je velikost paměti RAM. Někteří odborníci navrhují používat datové soubory až ve dvacetinásobku velikosti RAM, jiní navrhují dvojnásobnou velikost RAM paměti.“ (18) V této práci byly použity datové soubory ve velikosti 10 GB při velikosti RAM paměti 4 GB. Doporučení je používat velikosti datových souborů vždy stejné pro testování různých typů zapojení diskových polí.

Bonnie++ provádí několik testů, v závislosti na použitých argumentech. Dokud nejsou všechny testy hotové, skoro nic se nezobrazuje. Hlavním výstupem je prostý text v 80 sloupcích, který je navržen tak, aby se vešel i po vložení do e-mailu a byl dobře zobrazen i na Braillových displejích. Druhým typem výstupu je **CSV** (čárkou oddělené hodnoty). To lze snadno importovat do tabulkových procesorů (MS Excel) nebo databázových programů. (19)

Pro přehlednější vyjádření výsledků, můžeme použít program **bon_csv2html**, který uloží výsledky z formátu CSV do HTML tabulky. Naopak program **bon_csv2txt** generuje textovou tabulku. Pokud ve výpise výsledků narazíme místo naměřené hodnoty na řadu „++++“, znamená to, že rychlost vykonání daného testu byla tak rychlá, že program nemohl naměřit věrohodné hodnoty, konkrétně méně jak 500 ms. Aby zobrazené hodnoty nebyly nepřesné, program raději vypíše řadu „++++“. Řešení je snadné, navýšíme počet testovaných souborů, či velikost testovaných souborů. (17)

Ukázka použití programu bon_csv2html:

```
echo \ [4]
1.97,1.97,testPrvniOddil,1,1414331631,10000M,,597,98,44722,13,19531, \
7,3266,87,54361,9,156.8,8,500,,,,,32436,37,746812,99,885,1,33257, \
38,980859,96,817,1,38830us,524ms,3264ms,87743us,20346us,1834ms, \
1577ms,343us,15308ms,1470ms,24us,13425ms | bon_csv2html > vystup.htm
```

Pro každý test dostaneme množství práce vykonané v kB za sekundu a procentuální vytížení procesoru. U výsledků výkonnosti vyšší čísla jsou lepší, zato pro vytížení procesoru, jsou lepší hodnoty nižší. Výsledek výkonu 2 000 kB za sekundu s vytížením procesoru 90 % jsou lepší hodnoty, než výkon 1 000 kB za sekundu s vytížením procesoru 60 %. K dispozici jsou dva druhy činnosti programu. Prvních šest testů, které se provádějí, pochází z původního Bonnie. Zpočátku se provádí série testů na soubor nebo soubory o známé velikosti. Ve výchozím nastavení je velikost 200 MB, ale ta ve většině případů nestačí. Pro každý test Bonnie++ zobrazuje počet kB zpracovaných za sekundu a vytížení procesoru. Tyto testy mají za úkol otestovat propustnost vstupních a výstupních operací a jsou navrženy tak, aby simulovaly některé typy databázových aplikací. Druhým cílem je otestování: vytváření, čtení a mazání mnoha malých souborů, které simulují práci web serveru. (19) (20)

Detaily jednotlivých testů

Prvních šest testů:

1. **Sekvenční výstup** (Sequential Output)
    1. **Po znaku** (Per Character): Soubor je zapsán pomocí metody **putc()** stdio macro. Smyčka, která zajišťuje zápis musí být dostatečně malá, aby se vešla do jakékoli přiměřené l-cache.

[4] Tento dlouhý příkaz převede hodnoty do HTML tabulky.

    2. **Blok** (Block): Soubor je vytvořen pomocí metody **write()**. Procesor by se měl starat pouze o alokování prostoru pro soubor.
    3. **Přepis** (Rewrite): Každý soubor je přečten pomocí metody **read()**, je smazán a následně přepsán pomocí metody **write()**. Vzhledem k tomu, že se nemusí znovu alokovat prostor pro data, tak by měl tento test efektivně vyzkoušet účinnost cache souborového systému a rychlost přenosu dat.
2. **Sekvenční vstup** (Sequential Input)
    1. **Po znaku** (Per Character): Soubor je načten pomocí metody **getc()** stdio macro. Opět platí, že vnitřní smyčka je velmi malá. Ta by měla vykonávat pouze stdio a sekvenční vstup.
    2. **Blok** (Block): Soubor je čten pomocí metody **read()**. Tento test by měl být velmi ryzím testem výkonu sekvenčního vstupu.
3. **Náhodné vyhledávání** (Random seeks): Tento test spustí ve výchozím nastavení 3 *SeekProcCount* procesy paralelně a provede náhodné vyhledávání souborů pomocí metody **random()** v *bsd* systémech, **drand48()** na systémech *sysV*. (19)

Posledních šest testů:

Testy, které vytvářejí soubory, používají názvy souborů obsahující sedm číslic a náhodný počet (od 0 do 12) alfanumerických znaků.

- Pro **sekvenční testy** (Sequential tests) se náhodné znaky ve jméně souboru nacházejí za číslicemi.
- V **náhodných testech** (Random tests) jsou náhodné znaky ve jméně souboru na prvním místě.

Sekvenční testy zahrnují vytváření souborů v číselném seřazení, čtení souborů a mazání souborů. Náhodné testy obsahují totéž. (19)

Zde je uvedeno několik parametrů, které můžeme použít pro testování:

```
-d používá se pro specifikaci, kde je souborový systém umístěn,
-u používá se k volbě konkrétního uživatele, který spustí testy,
například spuštění pod uživatelem root,
-g výběr konkrétní skupiny, která spustí program,
-r zde specifikujeme velikost RAM paměti systému,
-s zde nastavíme velikost testovaných datových souborů v MB,
-n určuje počet testovaných souborů,
```

-m zde můžeme pojmenovat konkrétní test,
-x zde můžeme určit počet opakování testů. (18)

5.2 Popis konfigurace měření výkonu diskových polí

Pro všechny typy zapojení diskových polí, bylo použito stejné výchozí nastavení testů. Nejprve byl naformátován pevný disk a vytvořen na něm primární oddíl ve velikosti 20 GB. Dále byl vytvořen konkrétní typ diskového pole a na něm vytvořen souborový systém *ext4*, či *zfs*. Příkaz, který slouží ke spuštění testů je následující:

```
sudo bonnie++ -u root –d /media/přípojnýBod –m názevTestu –s 10000 \
–n 500
```

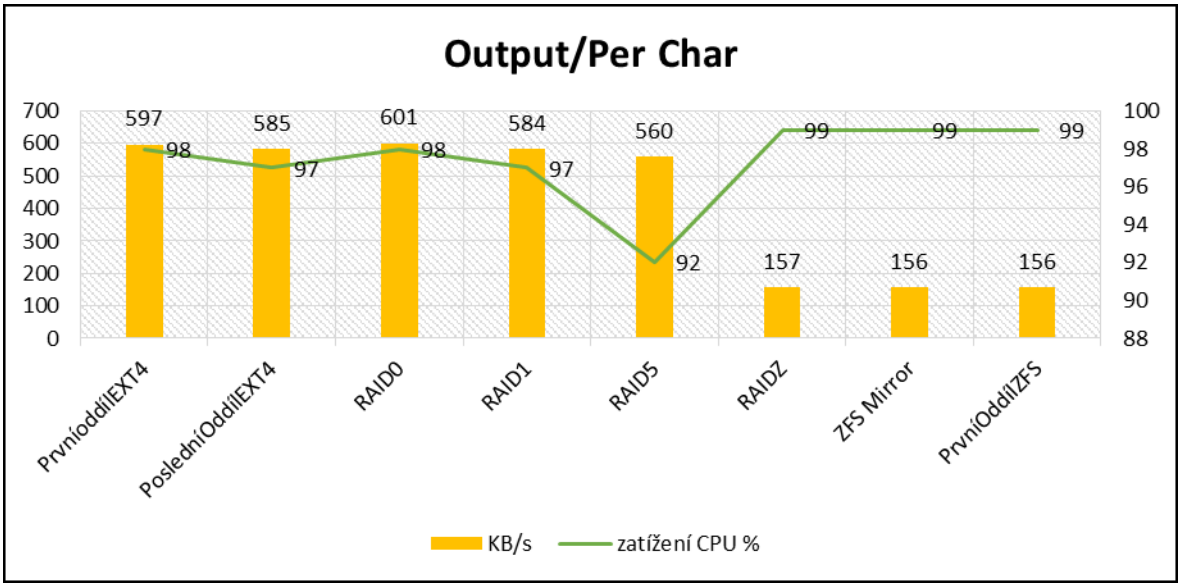
V příkazu je specifikován přípojný bod souborového systému, velikost testovacích datových souborů a počet datových souborů.

5.3 Vyhodnocení výsledků měření výkonu diskových polí

Pro jednotlivé typy měření byly vytvořeny přehledné tabulky a grafy v tabulkovém procesoru MS Excel. Následně byly porovnány otestované typy zapojení diskových polí mezi sebou pomocí grafů.

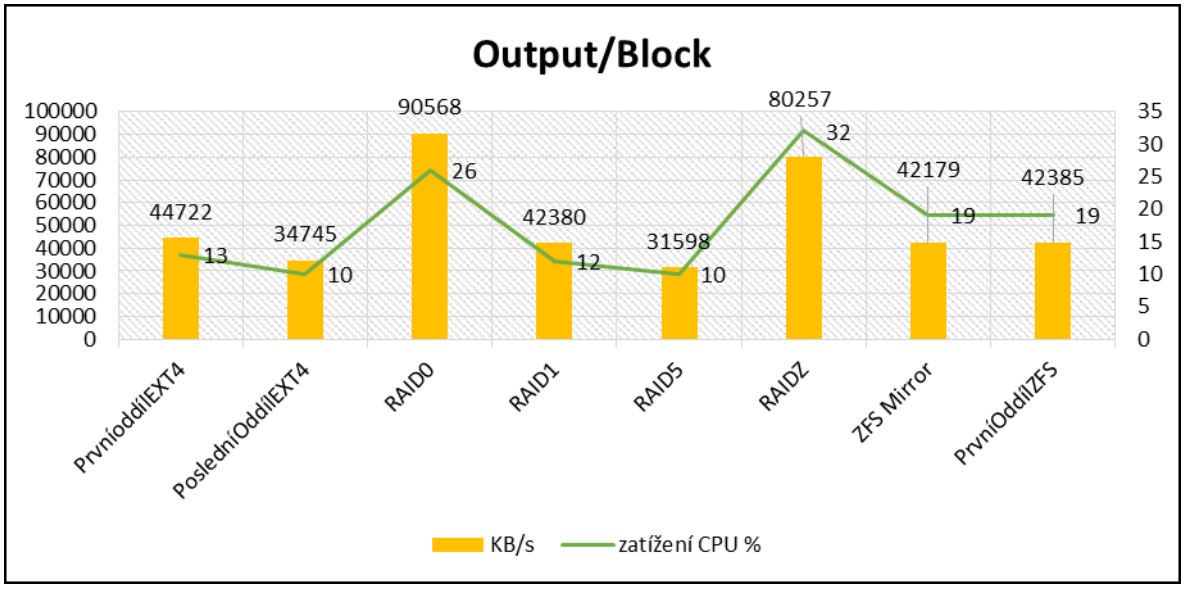
Pro tvorbu diskových polí se více hodí souborový systém ext4, který dosahuje ve většině testů lepšího výkonu a menší zátěže procesoru, než při použití souborového systému zfs. Souborový systém zfs dosahuje lepšího výkonu pouze při provádění sekvenčního a náhodného mazání. Typ RAID 0 dosahuje dvojnásobného výkonu vůči použití jednoho pevného disku v případech: sekvenčního výstupu po blocích, sekvenčního přepisu, sekvenčního vstupu po blocích a při náhodném vyhledávání. V ostatních testech je výkon RAID 0 a jednoho pevného disku srovnatelný. Použití softwarových diskových polí bych doporučil, jelikož výkon v mnoha testech byl leckdy vyšší při téměř srovnatelném zatížení systémového procesoru, nežli bez použití diskových polí. Pro dosažení co možná nejlepšího výkonu, je důležité vytvářet primární oddíly na začátcích pevných disků, jelikož výkon je závislý na fyzickém umístění oddílů na pevném disku.

Přehled všech typů zapojení diskových polí:



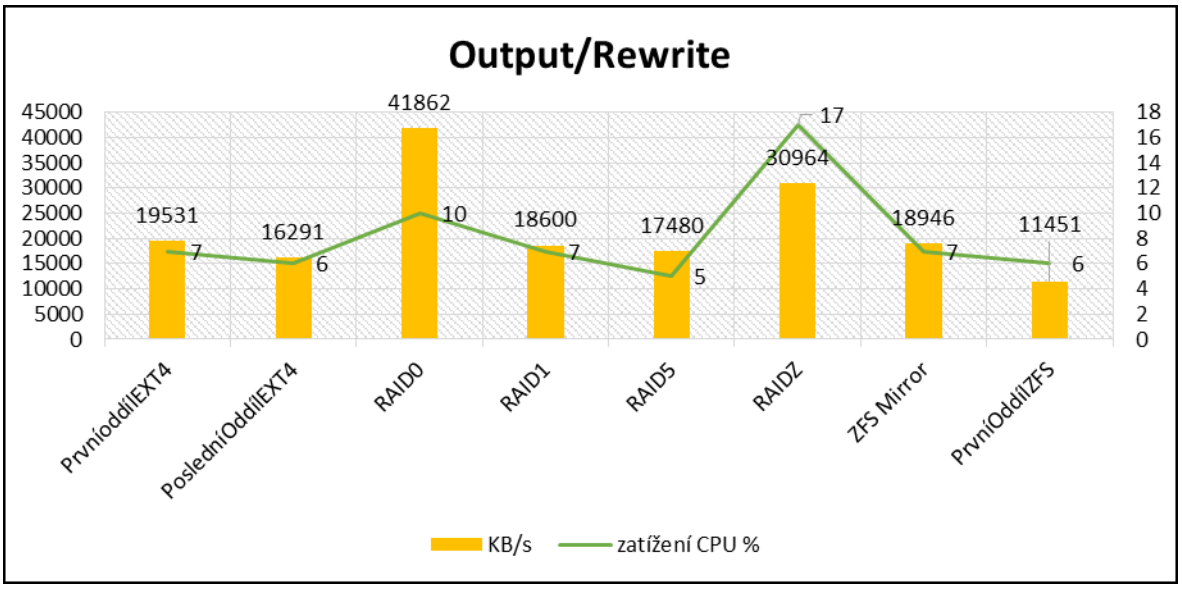
Obrázek 6 – Porovnání Output/Per Char

Zdroj: Autor



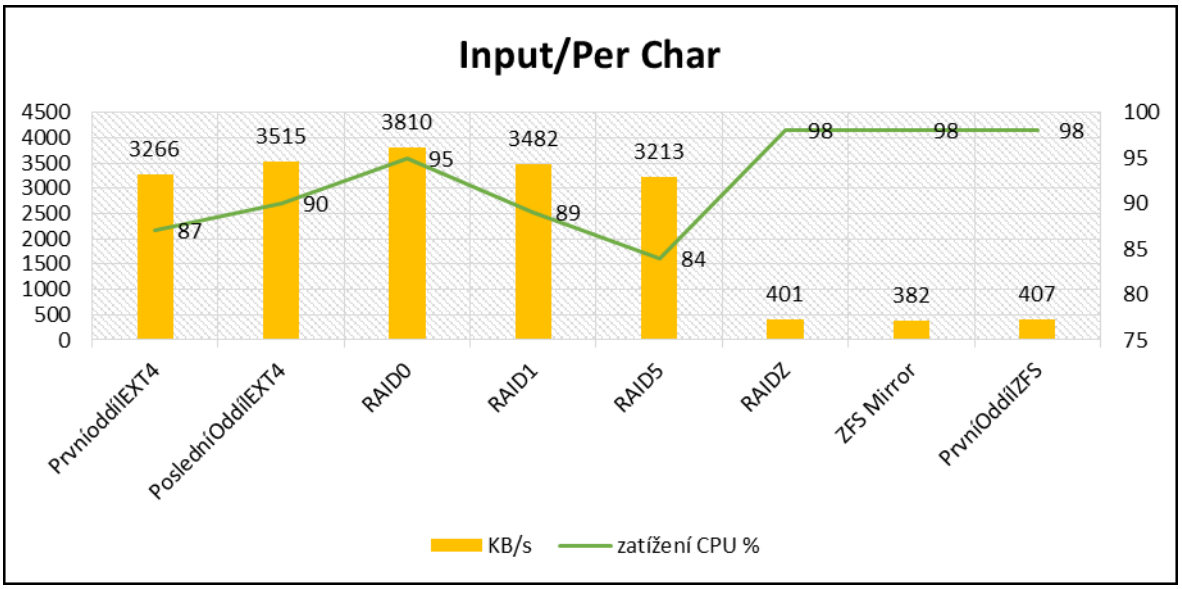
Obrázek 7 – Porovnání Output/Block

Zdroj: Autor



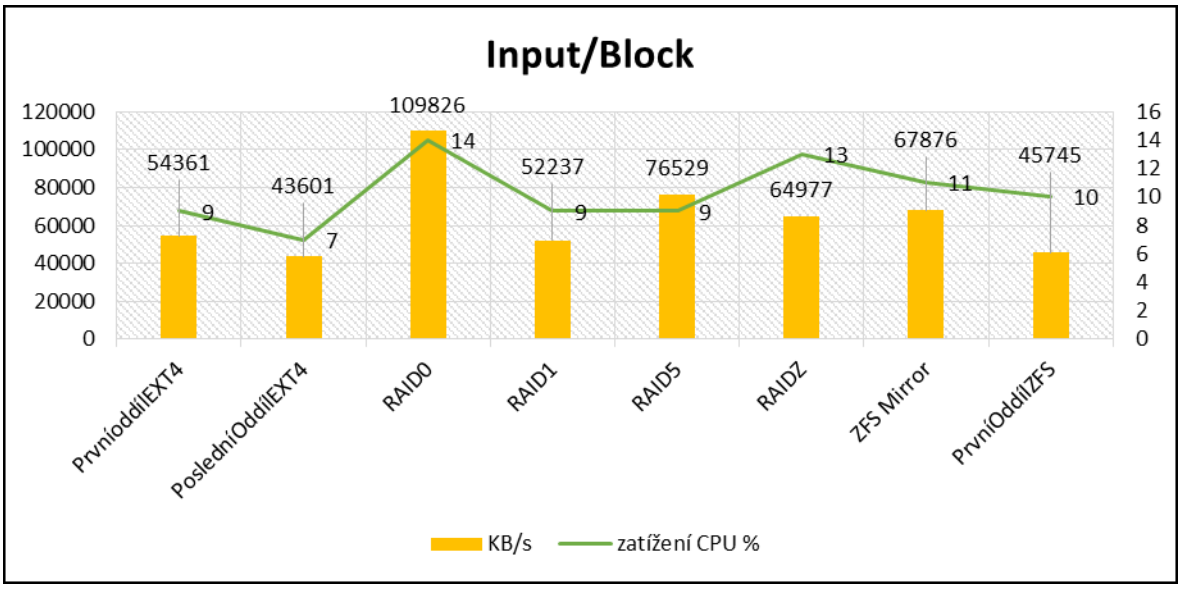
Obrázek 8 – Porovnání Output/Rewrite

Zdroj: Autor



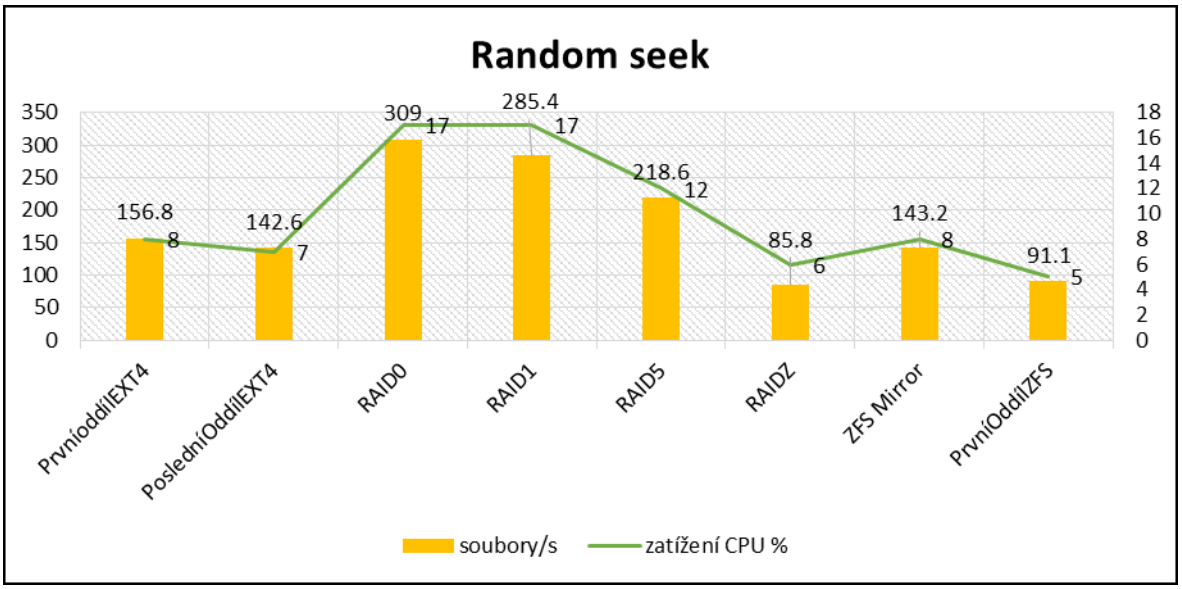
Obrázek 9 – Porovnání Input/Per Char

Zdroj: Autor



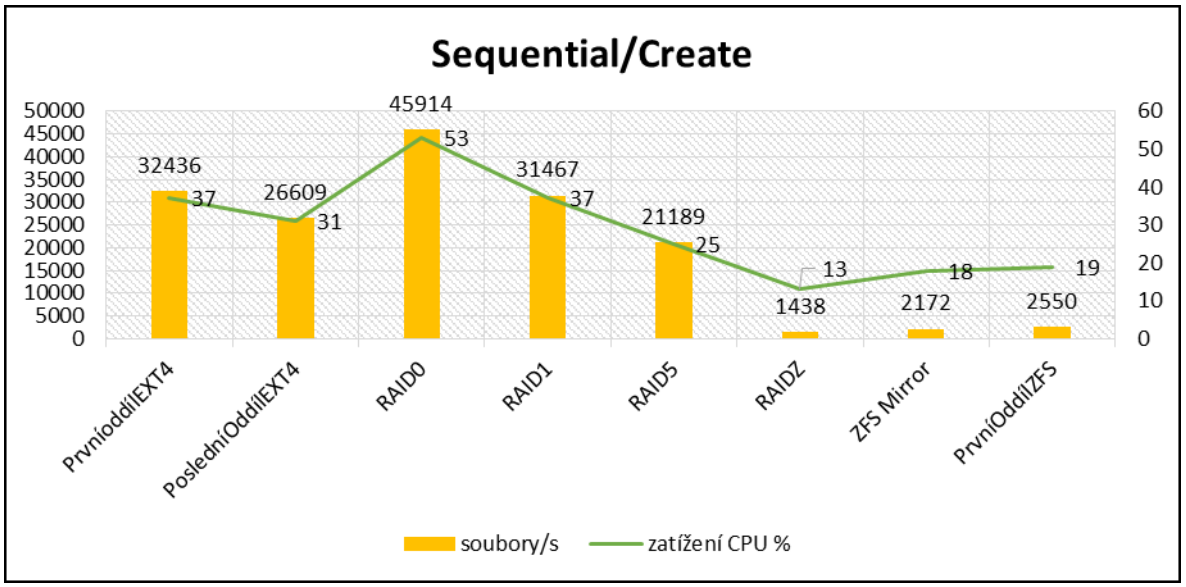
Obrázek 10 – Porovnání Input/Block

Zdroj: Autor



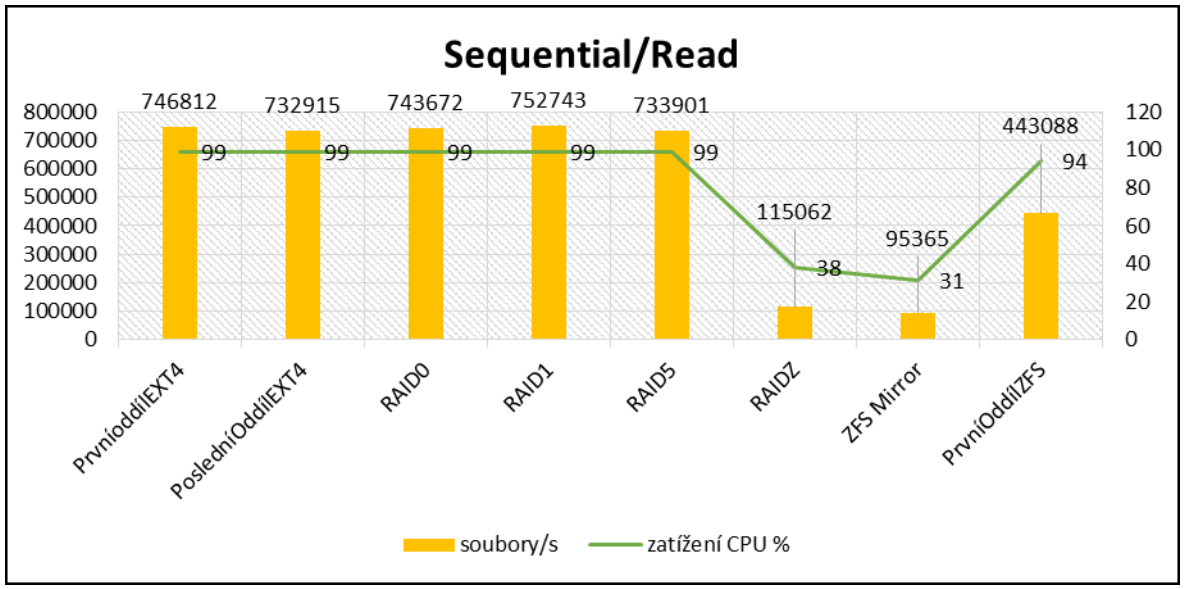
Obrázek 11 – Porovnání Random seek

Zdroj: Autor



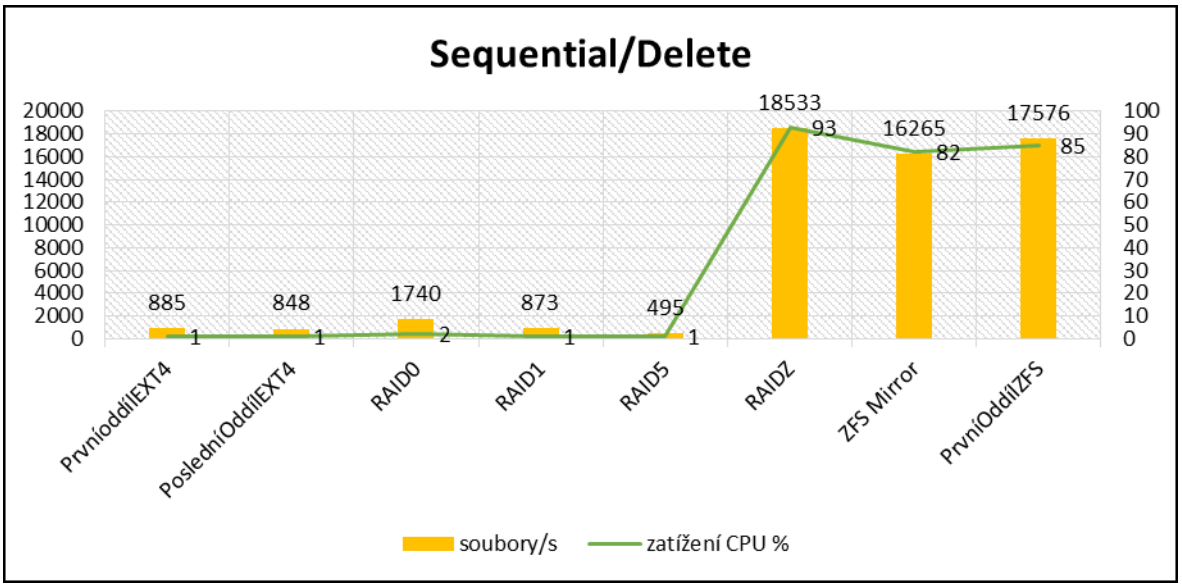
Obrázek 12 – Porovnání Sequential/Create

Zdroj: Autor



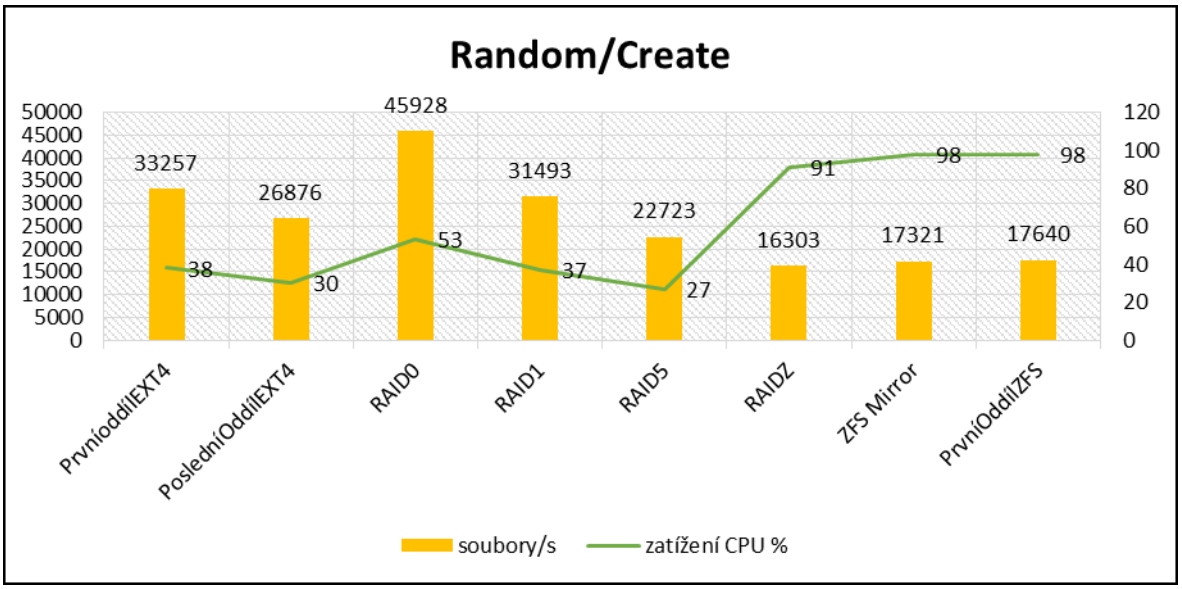
Obrázek 13 – Porovnání Sequential/Read

Zdroj: Autor



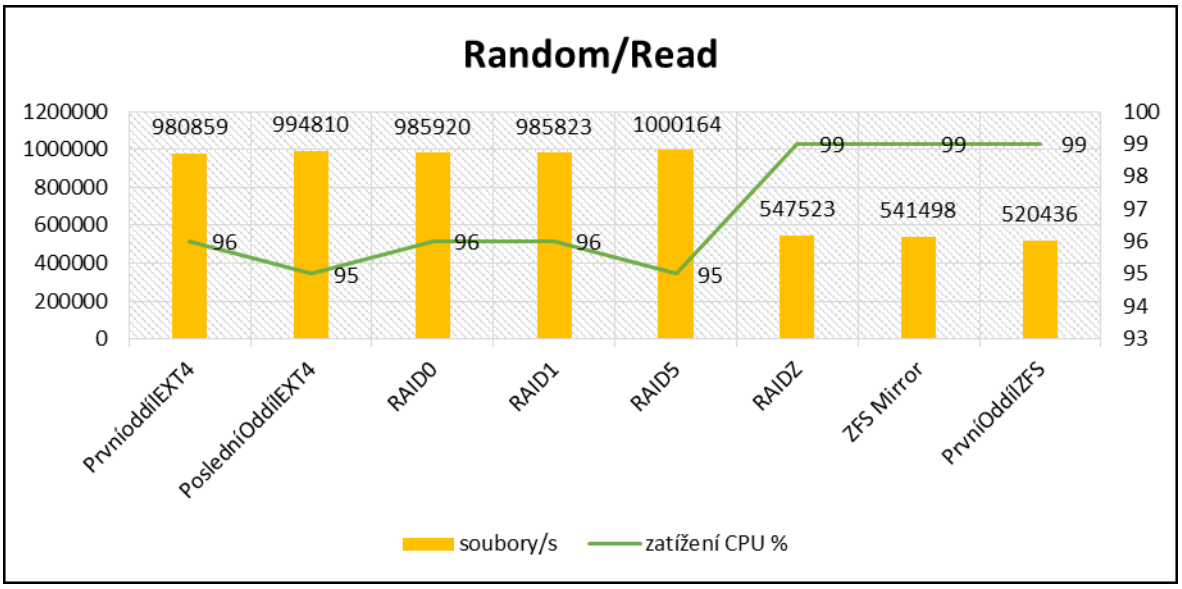
Obrázek 14 – Porovnání Sequential/Delete

Zdroj: Autor



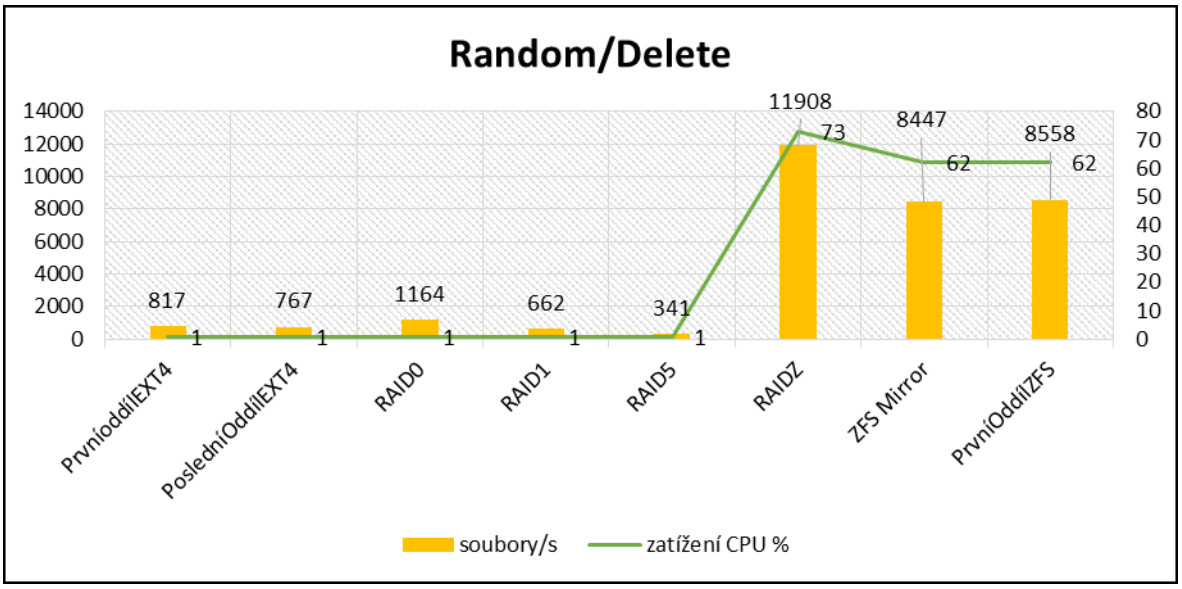
Obrázek 15 – Porovnání Random/Create

Zdroj: Autor



Obrázek 16 – Porovnání Random/Read

Zdroj: Autor



Obrázek 17 – Porovnání Random/Delete

Zdroj: Autor

Závěr

V této práci byly otestovány softwarová řešení diskových polí jako alternativa k nákladným hardwarovým řešením. Tyto pole mají jisté výhody jako je například flexibilita, která nabízí sestavení pole v degradovaném režimu. Bylo ověřeno, že zapojení RAID 0 má v některých typech operací opravdu dvojnásobný výkon oproti jednomu pevnému disku jak je vidět z grafů. V průběhu měření byl zjištěn fakt, který ovlivňuje rychlost čtení a zápisu pevného disku a tím je fyzická poloha diskových oddílů. Pokud byl vytvořen diskový oddíl na začátku pevného disku, tak rychlost čtení i zápisu byla rychlejší než při tvorbě diskového oddílu na konci pevného disku. Dále bylo vypozorováno z provedených testů, že vliv na výkon má nejen volba typu diskového pole, ale i souborový systém. Z použitých souborových systémů je lepší volbou ext4, který dosahoval ve většině testů vyššího výkonu a nižšího zatížení procesoru. Použití softwarových diskových polí bych doporučil, jelikož výkon v mnoha testech byl leckdy vyšší při téměř srovnatelném zatížení systémového procesoru, nežli bez použití diskových polí.

Seznam použitých informačních zdrojů

1. Häring, David. Softwarový RAID pod Linuxem. *Linuxové noviny.* [Online] 28. Červenec 2001. [Citace: 20. Říjen 2014.] http://www.linux.cz/noviny/2001-08/clanek04.html. ISSN 1213-3655.

2. RAID. *Wikipedie.* [Online] 8. září 2014. [Citace: 27. Říjen 2014.] http://cs.wikipedia.org/wiki/RAID.

3. Samek, Jan. Není RAID jako RAID – díl 1. Úvod. *Jens.cz.* [Online] 12. srpen 2009. [Citace: 26. Říjen 2014.] http://www.jens.cz/neni-raid-jako-raid-dil-1-uvod/.

4. Linux. *Wikipedie.* [Online] 26. říjen 2014. [Citace: 2. Listopad 2014.] http://cs.wikipedia.org/wiki/Linux.

5. Pevný disk, HDD. *Datahelp.* [Online] 20?? [Citace: 4. Listopad 2014.] http://www.zachrana-dat-hdd.cz/pevny-disk-hdd.

6. Pevný disk. *Wikipedie.* [Online] 30. září 2014. [Citace: 21. Říjen 2014.] http://cs.wikipedia.org/wiki/Pevný_disk.

7. Matejka. *Pevný disk.* [Online] 2005. [Citace: 21. Říjen 2014.] http://www1.osu.cz/home/matejka/soft/data/harddisk.htm.

8. Dytrych, Karel. Typy diskových oddílů. *school.kjn.cz.* [Online] 2008. [Citace: 21. Říjen 2014.] http://school.kjn.cz/operacni-systemy/partitions.html.

9. Hejda, Václav. Úschovna pro naše data – souborové systémy. *Linuxexpres.* [Online] 7. říjen 2013. [Citace: 23. Říjen 2014.] http://www.linuxexpres.cz/praxe/uschovna-pro-nase-data-souborove-systemy. ISSN 1801-3996.

10. —. Souborové systémy v Linuxu. *Linuxexpres.* [Online] 21. listopad 2011. [Citace: 23. Říjen 2014.] http://www.linuxexpres.cz/blog/souborove-systemy-v-linuxu. ISSN 1801-3996.

11. Vojtaj, Jaromír. PC-BSD: souborový systém ZFS. *root.cz.* [Online] 20. květen 2013. [Citace: 24. Říjen 2014.] http://www.root.cz/clanky/pc-bsd-souborovy-system-zfs/. ISSN 1212-8309.

12. ZFS. *Wikipedie.* [Online] 2. březen 2014. [Citace: 24. Říjen 2014.] http://cs.wikipedia.org/wiki/ZFS.

13. Krčmář, Petr. Co umí souborový systém ZFS. *root.cz.* [Online] 29. srpen 2006. [Citace: 3. Listopad 2014.] http://www.root.cz/clanky/co-umi-souborovy-system-zfs/. ISSN 1212-8309.

14. Dočekal, Michal. Správa linuxového serveru: Softwarový RAID prakticky. *Linuxexpres.* [Online] 14. leden 2010. [Citace: 2. Listopad 2014.] http://www.linuxexpres.cz/praxe/sprava-linuxoveho-serveru-softwarovy-raid-prakticky. ISSN 1801-3996.

15. Keršláger, Mgr. Milan. RAID v Linuxu. *pslib.cz.* [Online] 1. listopad 2013. [Citace: 3. Listopad 2014.] https://www.pslib.cz/ke/RAID_v_Linuxu.

16. Jamrich, Marián. ZFS: vytvárame a upravujeme základ pre naše dáta. *root.cz.* [Online] 26. leden 2011. [Citace: 4. Listopad 2014.] http://www.root.cz/clanky/zfs-vytvarame-a-upravujeme-zaklad-pre-nase-data/. ISSN 1212-8309.

17. Samek, Jan. Bonnie++: linux filesystem benchmark. *Jens.cz.* [Online] 22. březen 2009. [Citace: 25. Říjen 2014.] http://www.jens.cz/bonnie-linux-filesystem-benchmark/.

18. Coyle, James. Benchmark disk IO with DD and Bonnie++. *JamesCoyle.net.* [Online] 11. září 2013. [Citace: 26. Říjen 2014.] http://www.jamescoyle.net/how-to/599-benchmark-disk-io-with-dd-and-bonnie.

19. Coker, Russell. Bonnie++ Documentation. *coker.com.au.* [Online] 2007. [Citace: 26. Říjen 2014.] http://www.coker.com.au/bonnie++/readme.html.

20. bonnie++(8) - Linux man page. *die.net.* [Online] 20?? [Citace: 26. Říjen 2014.] http://linux.die.net/man/8/bonnie++.