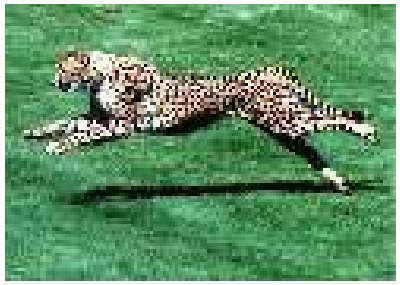
IBM – Informix Dynamic Server



Informix Dynamic Server 11.50
Multi-instance Active Cluster pro vysokou dostupnost

Jan Musil
IT Specialist SWG IBM

Přehled prezentace

- Co je to “MACH-11” ?
- Architektura řešení
- Nové typy sekundárních serverů
  - Remote Standalone Secondary (RSS)
  - Continuous Log Restore (CLR)
  - Shared Disk Secondary (SDS)
- Zpřístupnění sekundárních serverů pro zápisy
- Connection Manager
- Connection Manager Arbitrator

Co je to “MACH-11”?

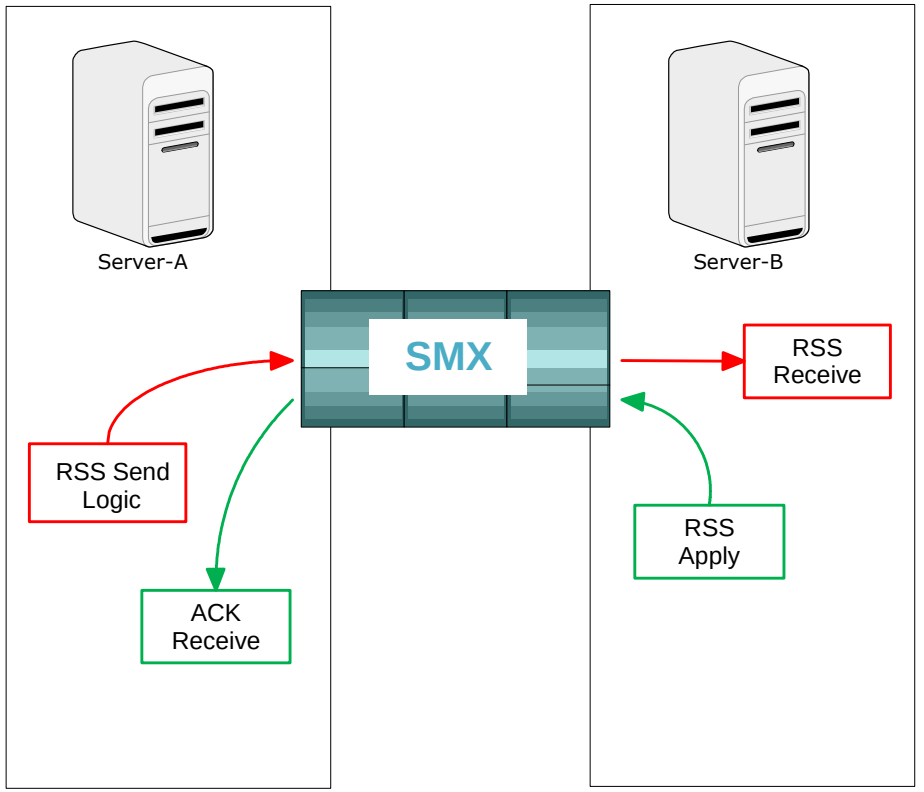
- Multi-instanční aktivní cluster pro vysokou dostupnost (Multi-instance Active Cluster for High Availability)
  - Zásadním způsobem rozšiřuje funkcionalitu HDR o nové typy sekundárních serverů
  - Tři nové typy sekundárních instancí:
    - Remote Standalone Secondary (RSS)
    - Shared Disk Secondary (SDS)
    - Continuous Log Restore (CLR) resp. “near-line” standby
  - Technologie HDR, RSS, CLR & SDS se mohou použít společně v libovolné kombinaci
    - Navíc lze transparentně provozovat v prostředí Enterprise Replication (ER)
- “MACH-11” není:
  - pouze 1-ku-N HDR….
  - … ale se všemi novými sekundárními servery poskytuje funkcionalitu třívrstvé architektury vysoké dostupnosti

Architektura řešení

- Dvě nové komponenty
  - Server Multiplexer (SMX):
    - Interně zajišťuje TCP/IP komunikaci mezi všemi MACH-11 instancemi
    - Podporuje vícenásobná logická připojení prostřednictvím jednoho fyzického TCP/IP připojení
  - Index Page Logging:
    - Vytváření indexů se zapisuje do logických žurnálů
    - Povinné pro RSS a SDS, volitelné pro HDR

Server Multiplexer (SMX)

- Vytváří multiplexované síťové propojení mezi dvěma servery
- Prostřednictvím jednoho SMX propojení komunikuje více interních vláken
- Vlákna na jednom uzlu mohou jednoduše navázat logické spojení s vlákny na druhém serveru
- Plně duplexní protokol:
  - HDR komunikují pouze prostřednictvím half-duplex protokolu
- Podpora šifrované komunikace
- Velmi jednoduchá komunikace mezi instancemi
- Aktivuje se automaticky
- Není třeba žádná konfigurace (vyjma šifrování)

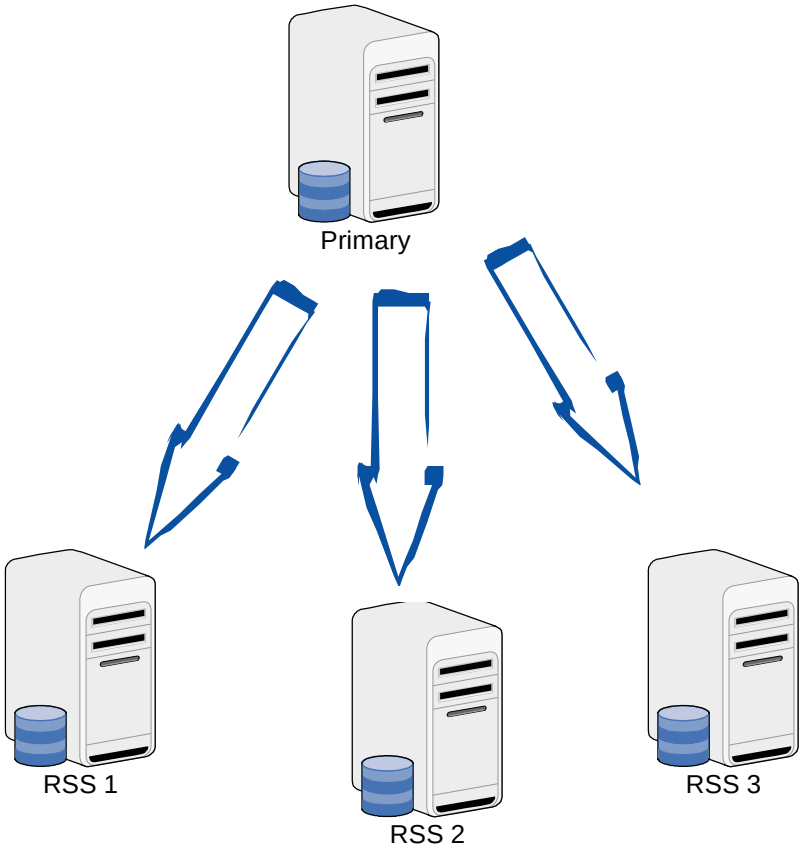


Index Page Logging

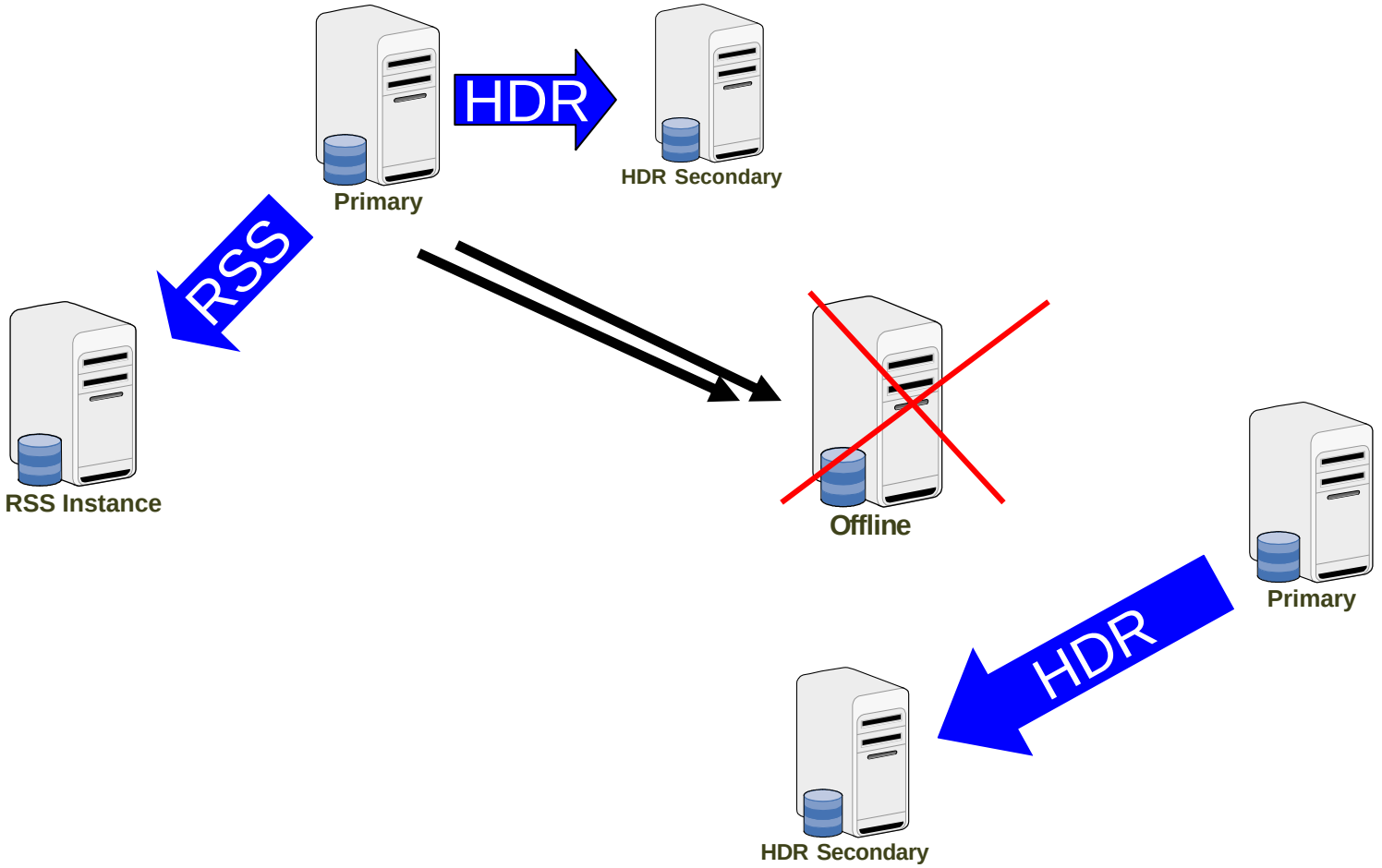
- Technologie vytváření indexů používaná u HDR není vhodná
  - Sekundární instance musí být dostupná
  - Použití indexů na primáru je opožděno v důsledku čekání na potvrzení od sekundáru
  - U RSS instance nevyžadujeme on-line dostupnost, vytváření indexů by tak blokovalo jejich použití
- Řešení prostřednictvím Index Page Logging
  - Při vytváření indexu se indexové stránky nezasílají na sekundár přímo, ale zapisují se do logických žurnálů
  - Povinné pro RSS a SDS
  - Žurnálování indexů může být rozděleno do několika transakcí a je nezávislé na uživatelské transakci, uživatelská transakce a interní „indexová“ transakce jsou plně koordinované
- Aktivace Index Page Logging
  - LOG_INDEX_BUILDS 1
  - onmode –wf LOG_INDEX_BUILDS=1.
- Pokud existují RSS a SDS instance, index page logging nelze vypnout
- Pokud je Index Page Logging aktivní, HDR přenáší indexy novým způsobem prostřednictvím logických žurnálů

Remote Standalone Secondary

- Podobné jako u HDR
  - Plná kopii celé instance
  - Lze použít pro reportování (Read-Only)
  - Vytváří se fyzickou obnovou ze zálohy primáru
- Odlišné od HDR:
  - Používá plně duplexní komunikační protokol (SMX) (lepší propustnost u pomalejších linek)
  - Žádná operace se neprovádí synchronně (ani kontrolní bod)
  - Nelze převést do primáru, ale lze převést na HDR sekundár
  - Může existovat libovolný počet RSS instancí
  - Vyžaduje Index Page Logging
- RSS lze použít v kombinaci s HDR sekundárem
  - RSS lze převést na HDR sekundár
  - HDR sekundár lze převést na RSS



Příklad použití HDR a RSS



Konfigurace RSS instance

Primární instance

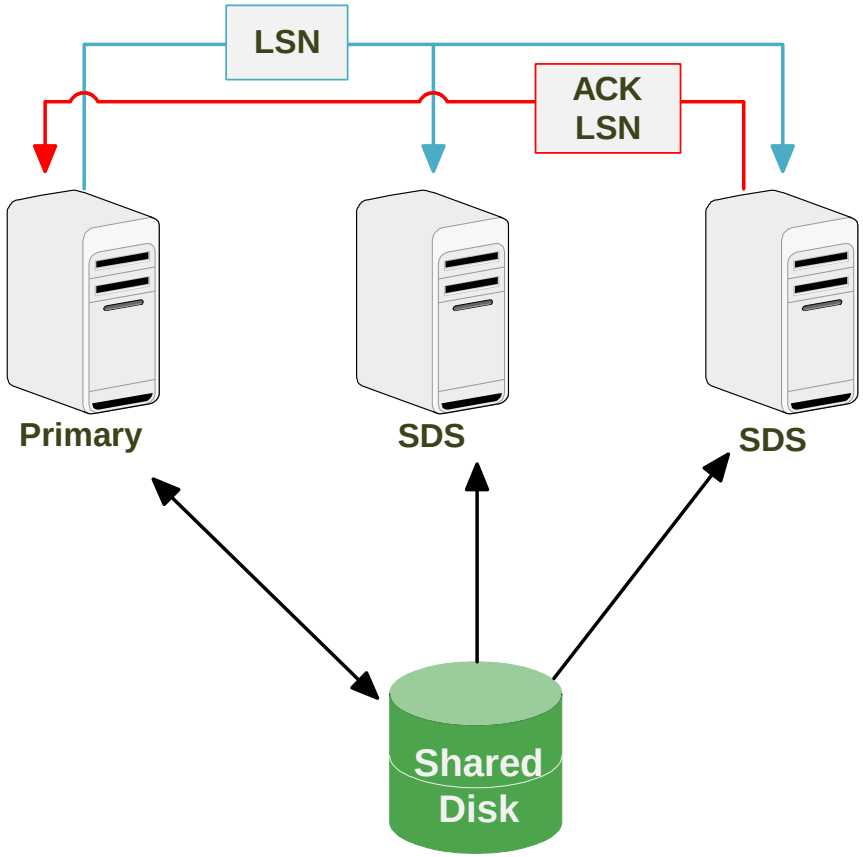
- Zapnout Index Page Logging
  - LOG_INDEX_BUILD 1 nebo
  - onmode –wf LOG_INDEX_BUILDS=1
- Definice RSS instance/instancí
  - onmode –d add RSS <RSS instance Name> <optional password>
  - ….
- Provést zálohu úrovně 0
  - ontape –s –L 0

RSS instance

- Zapnout Index Page Logging
  - LOG_INDEX_BUILD 1 nebo
  - onmode –wf LOG_INDEX_BUILDS=1
- Provést fyzickou obnovu na RSS instanci
  - ontape –p
- Aktivovat RSS instanci
  - onmode –d RSS <source instance> <optional password>

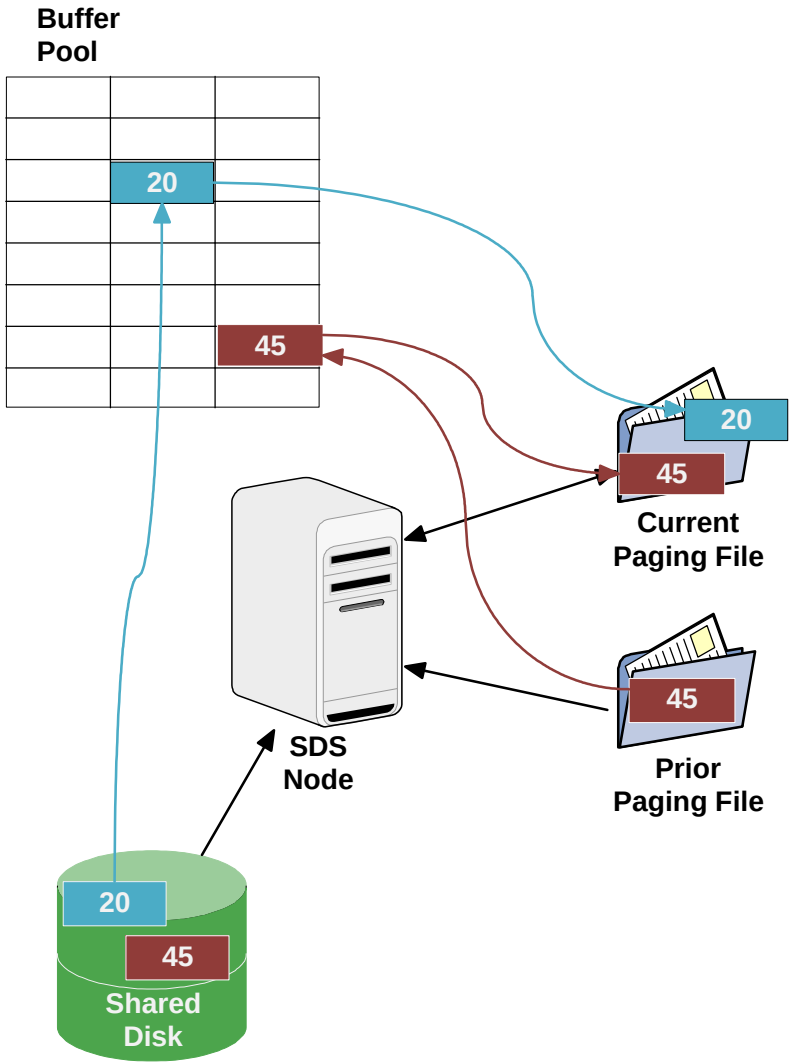
Shared Disk Secondary (SDS)

- Jakmile primár zapíše do logických žurnálů, posílá všem SDS Log Sequence Number (LSN)
- SDS obdrží LSN z primáru a přečte logické žurnály ze sdílených disků
- SDS aplikuje všechny změny z logických žurnálů do své sdílené paměti
- SDS potvrdí primáru resynchronizaci na základě LSN
- SDS instance nikdy nezapisuje do sdílených disků (ani při kontrolním bodě)
  - Pokud SDS potřebuje uvolnit buffer sdílené paměti, dočasně ho zapíše do strínkovacího souboru (paging file)
- Primár nepřepíše původní verzi stránky na disku, dokud si není jistý, že některá SDS nebude tuto stránku potřebovat



SDS stránkovací soubory

- Slouží k dočasnému uložení stránek, které by se u „normální“ instance zapsaly na disk
- Stránky se zde uchovávají do dalšího kontrolního bodu
- Z důvodů podpory Non-Blocking Checkpoints jsou dva stránkovací soubory
  - Aktuální (current paging file)
    - Modifikované stránky od posledního kontrolního bodu
  - Předchozí (Prior Paging File)
    - Stránky modifikované v průběhu posledního kontrolního bodu
  - SDS čte požadované stránky v následujícím pořadí
    1. current paging file
    2. prior paging file
    3. chunk.
  - Zápisy se provádí pouze do aktuálního stránkovacího souboru



Dočasné databázové prostory SDS instancí

- SDS nemůže používat existující dočasné databázové prostory primáru
- Při startu SDS jsou existující dočasné prostory primáru pro SDS znepřístupněny
- SDS si vytvoří dynamicky své vlastní dočasné prostory na základě nastavení v konfiguračním souboru (SDS_TEMPDBS)

Inicializace SDS

Primární instance

1. Nakonfigurovat prostředí sdílených disků

2. Nastavit SDS_ENABLE na primáru

   onmode –d set SDS primary
   <instance_name_of_SDS_primary>

SDS instance

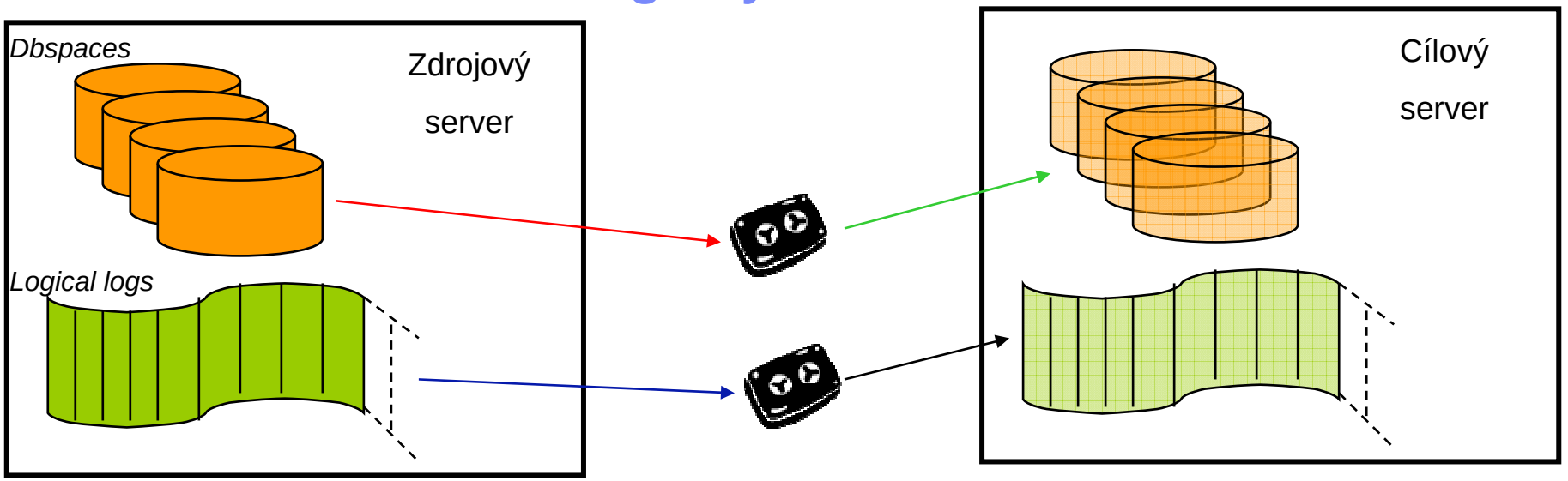
1. Vytvořit stránkovací soubory a soubor pro SDS dočasný databázový prostor
2. Upravit SDS $ONCONFIG soubor
   - SDS_ENABLE 1
   - SDS_PAGING *dva stránkovací soubory*
   - SDS_TEMPDBS *name, location (file created), page size, dbspace size, offset*
3. Následující konfigurační parametry musí být stejné, jako na primáru

   ROOTNAME ROOTPATH
   ROOTOFFSET ROOTSIZE
   PHYSDBS PHYSFILE
   LOGFILES LOGSIZE
4. Následující parametry musí být unikátní pro SDS instanci

   DBSERVERALIASES, DBSERVERNAME
5. oninit

   POZOR! Nesmí se spustit jako oninit –i !!!

Kontinuální obnova logických žurnálů



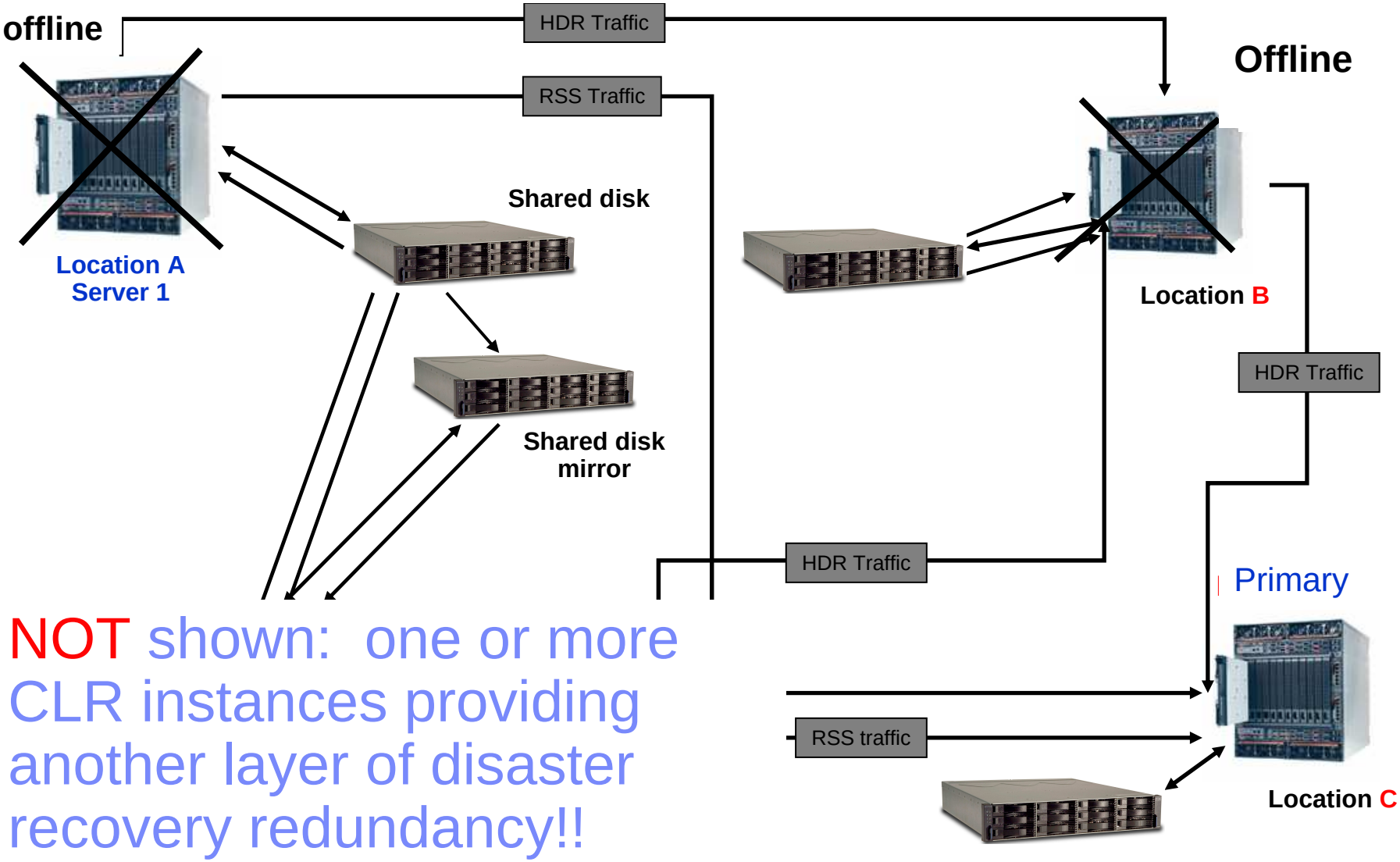
- ontape –s –L 0
- ontape –a
- ….
- ontape –a

- ontape –p ...fast recovery
- ontape –l –C ...fast recovery
- ontape –l –C ...fast recovery
- ….
- ontape –l –X ...quiescent
- onmode -m ...on-line

- onbar –b –L 0
- onbar –b -l
- ….
- onbar –b -l

- onbar -r –p
- onbar –r –l –C
- onbar –r –l –C
- ….
- onbar –r –l –X
- onmode -m

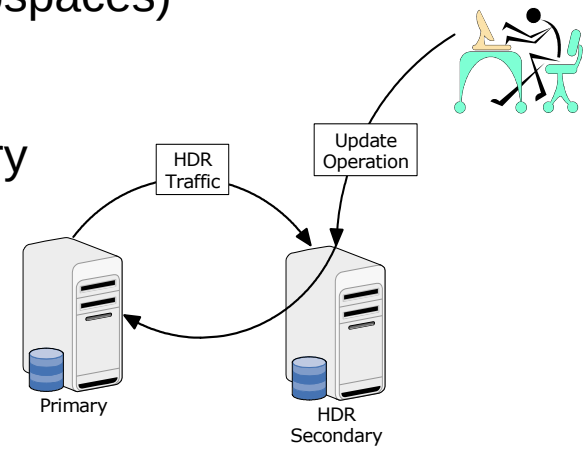
The server at location B fails



NOT shown: one or more CLR instances providing another layer of disaster recovery redundancy!!

Zpřístupnění sekundárních serverů pro zápisy

- Klientské aplikace mohou modifikovat data na sekundárních serverech s použitím tzv. *redirected writes* (přesměrované zápisy)
- Modifikace sekundárních serverů prostřednictvím přesměrovaných zápisů poskytuje dojem, že ke změně dochází skutečně na sekundárním serveru
- Ve skutečnosti je transakce přenesena na primární server, kde se fyzicky provede a odkud se všechny provedené změny rozdistribuují na všechny sekundární servery
- Lze použít pro modifikaci všech základních datových typů, uživatelsky definovaných typů a blobů uložených buď v žurnálovaných smart blob prostorech nebo v tabulkových databázových prostorech (nikoliv blobspaces)
- Data sekundárního serveru se nemodifikují přímo
- Je možné použít pro HDR, SDS a RSS sekundární servery
- Na sekundárních serverech je možné vytvářet jak implicitní, tak explicitní dočasné tabulky
- Podpora ER

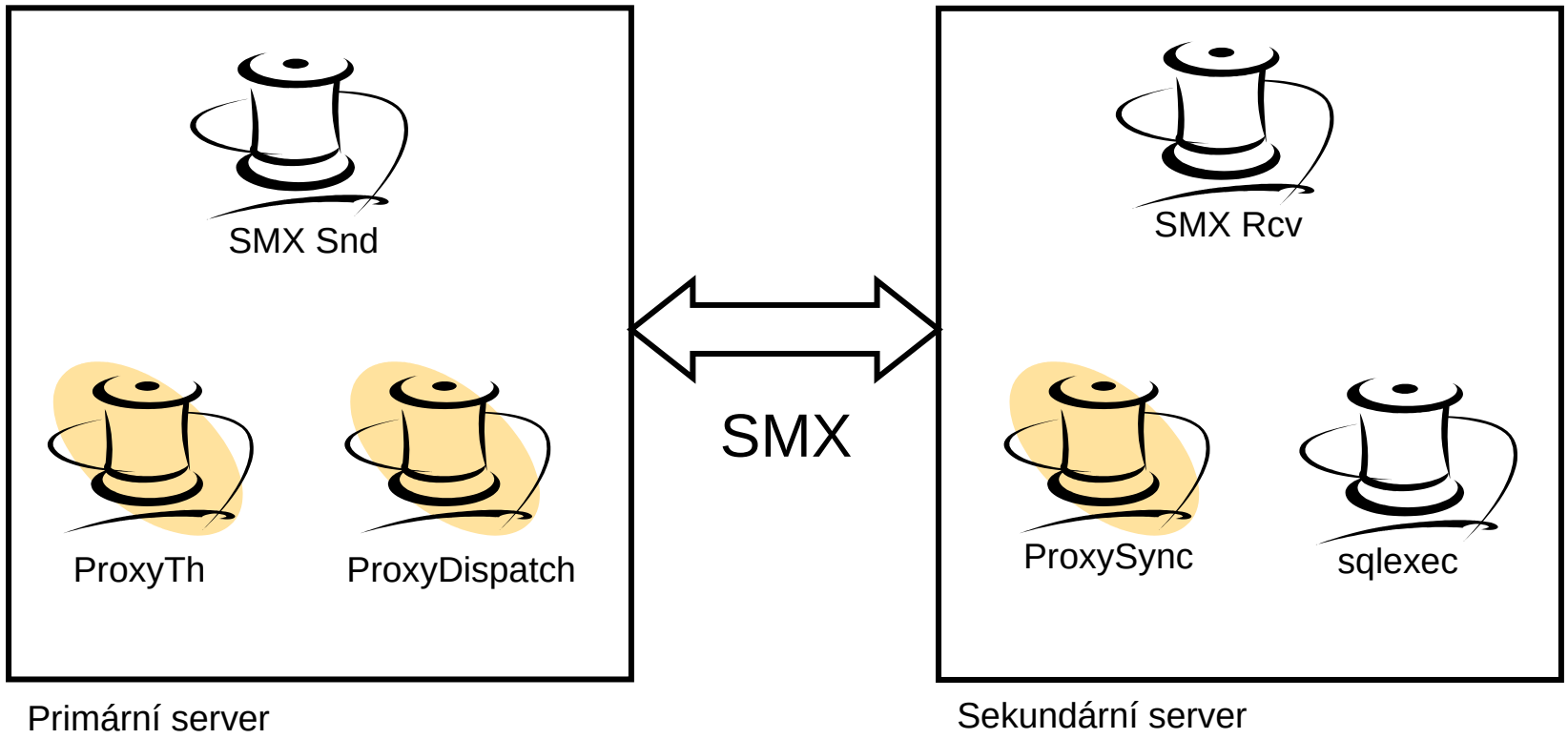


Používání přesměrovaných zápisů

- Ze sekundárního serveru se přesměrují na primární server všechny zápisy typu INSERT, UPDATE a DELETE
- Lze přesměrovat pouze DML operace
- Pro víceuživatelský přístup k datům se používá metoda optimistické konkurence (optimistic concurrency)
  - Přečtení dat do svého paměťového prostoru
  - Před ukončením transakce provedení kontroly, zda jiná transakce mezitím nemodifikovala stejná data
  - Pokud ano, transakce se zruší
  - Pokud ne, transakce se ukončí a zapíše na disk
- Na primárním serveru snižuje kolize způsobené zamykáním
- V případě pádu primárního serveru dojde k automatickému přesměrování prováděných transakcí na nový primární server
- Triggers a Constraints se aplikují na primárním serveru

Konfigurace a architektura přesměrovaných zápisů

- ONCONFIG: REDIRECTED_WRITES
  - Nastavená hodnota uvádí počet komunikačních rour a dispečerů pro zajištění provádění operace mezi sekundárním a primárním serverem
  - Nastavená hodnota by neměla být větší než dvakrát počet CPU VP



Aplikace optimistické konkurence pro přesměrované zápisy

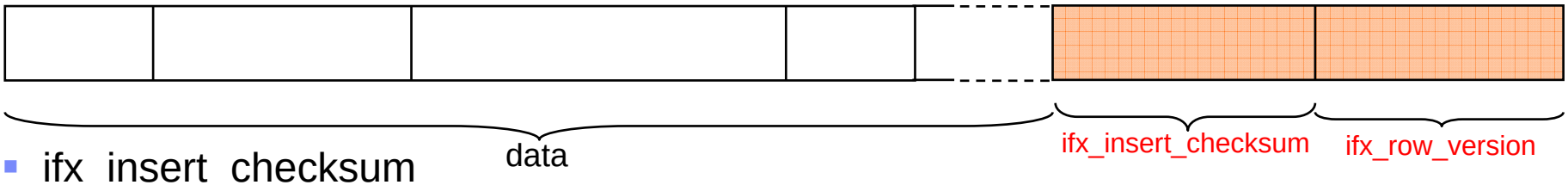
- (S) Vybere se záznam pro modifikaci a uchová se jeho obraz (*before image*)
- (S) Spustí se operace změny, která se přesměruje na (P)
- (P) Porovná se aktuální obraz záznamu s jeho původním obrazem ze (S)
- (P) Obrazy jsou různé -> Chyba: EVERCONFLICT (-7350) -> Zápis se neprovede

  (P) Obrazy jsou stejné -> Dokončí se operace s ukončením transakce
- (P) Rozešle provedenou změnu na všechny sekundární servery

- Problém: Původní obraz záznamu ze (S) se musí poslat k porovnání na (P)
- Jak optimalizovat síťovou komunikaci ?
- Řešení: verzování záznamů

Verzování záznamů

- Slouží k určení, zda se záznam změnil, a zda dochází ke konfliktu
- Pro provádění přesměrovaných zápisů není sice verzování záznamů vyžadované, snižuje však zatížení sítě a zvyšuje výkonnost
  - Pokud není verzování aktivní, sekundární server musí poslat primárnímu serveru pro porovnání celý záznam
  - V případě verzování se posílá pouze aktuální verze záznamu



- ifx_insert_checksum
  - Kontrolní číslice určená při vložení záznamu, která se nemění po celou jeho dobu existence
- Ifx_row_version
  - Verze záznamu, která se mění po každé modifikaci záznamu
- ifx_row_id
  - *Partition Number*:*rowid*:*ifx_insert_checksum*:*ifx_row_version*

```
1048928:257:741480809:1    7324334:258
```

SQL syntaxe pro aktivaci verzování záznamů

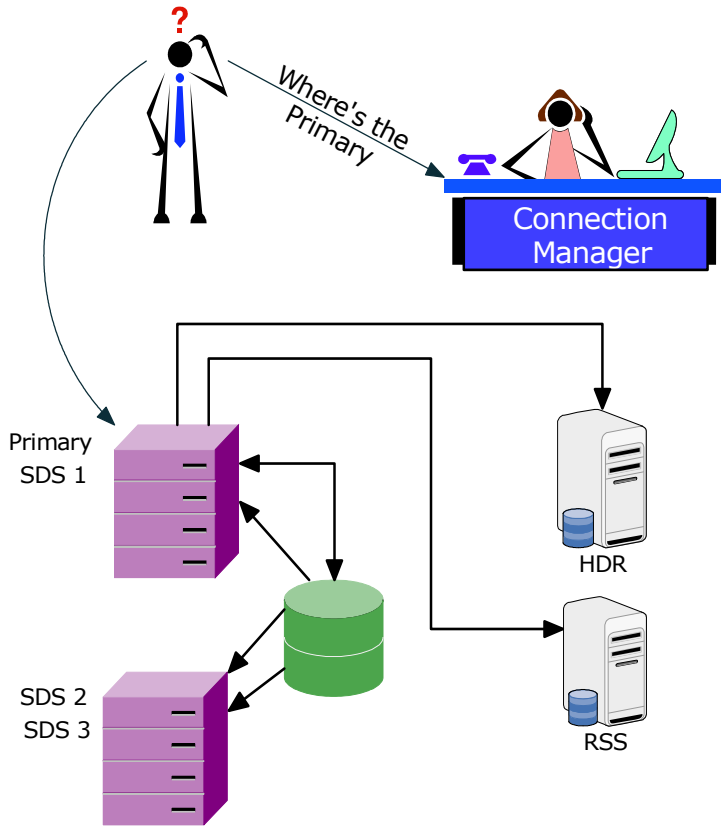
```
ALTER TABLE tablename add VERCOLS;

ALTER TABLE tablename drop VERCOLS;

CREATE TABLE tablename (
Column Name Datatype
Column Name Datatype
Column Name Datatype
) with VERCOLS;
```

Connection Manager (CM)

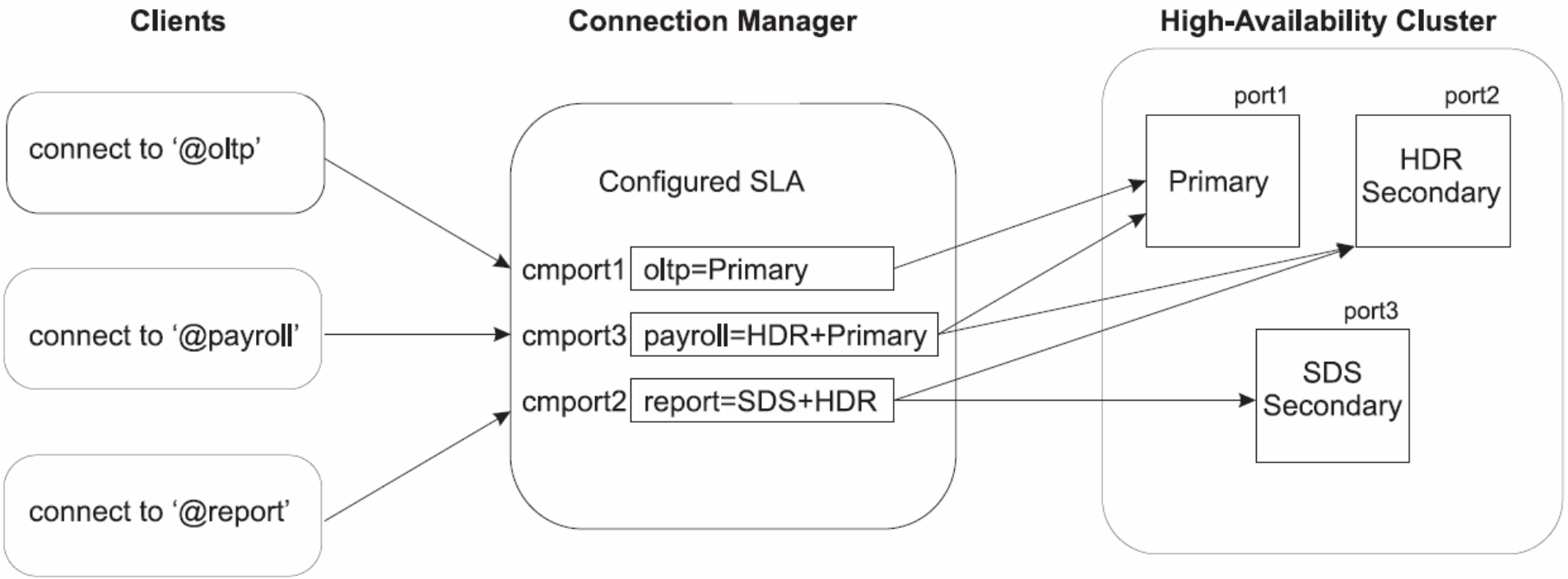
- Dynamicky přesměrovává požadavky na připojení od klientských aplikací na nejvhodnější server klástru vysoké dostupnosti
- Je připojen ke každému uzlu klástru a sbírá statistiky o typu uzlu, nevyužité kapacitě a jeho aktuálního stavu
- Na základě takto získaných statistik je schopen přesměrovat požadavek na připojení na nejvhodnější server
- Startuje se programem oncmsm (**ON**line **C**onnection **M**anager and **S**erver **M**onitor
- Vyžaduje instalaci ClientSDK 3.50 a vyšší
- Klientské aplikace musí být napsány v prostředí ClientSDK 3.50 a vyšší (
- Podporuje připojení prostřednictvím Distributed Relational Database Architecture (DRDA)



Service Level Agreements (SLA)

- Definice („smluvní poměr“ mezi klientem a serverem), podle které CM provádí přesměrování požadavků na připojení
- Založen na požadavku kvality dat a rychlosti jejich získání
- Typy požadavků na kvalitu dat
  - Aktuální data -> primární server
  - Určité zpoždění je možné, ovšem stále jsou vyžadovaná aktuální data -> SDS
  - Zpoždění v datech je možné a je akceptovatelné *dirty read* -> RSS, HDR
- SLA *jméno připojení = skupina uzlů*
  - *jméno připojení* – připojení, na kterém CM aktivuje *listener* vlákno
  - *skupina uzlů* – seznam uzlů, který definuje pořadí pro přesměrování požadavku na připojení
- Skupina uzlů může obsahovat klíčová slova primary, SDS, HDR a RSS nebo přímo jméno určitého serveru
  - Jednotlivé uzly jsou odděleny znakem ‘+‘
  - Pokud jsou uzly v závorce, pak CM z nich vybírá nejméně zatížený uzel
  - SLA onha1=SDS+HDR+primary SLA onha2=(SDS+HDR)+primary

Příklad konfigurace CM



sqlhosts file on Connection Manager and Client machines:

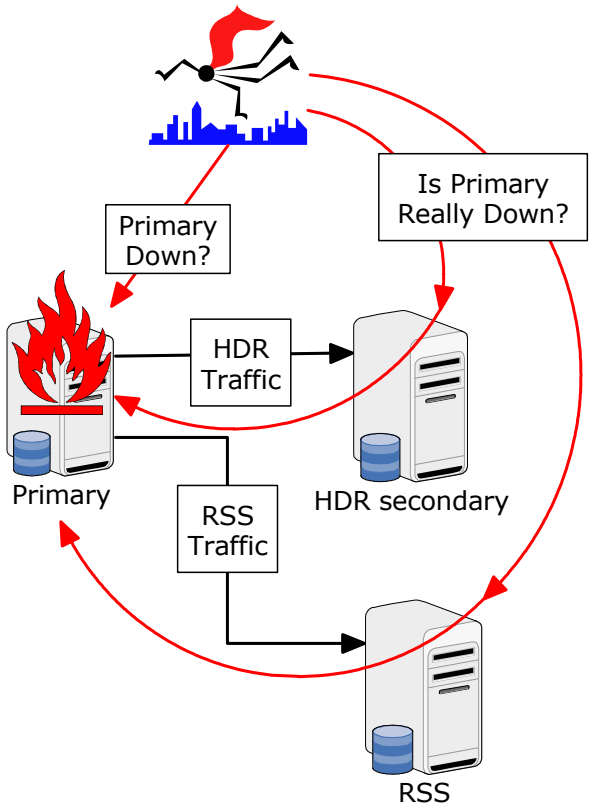
```
ids      onsoctcp host1 port1
oltp     onsoctcp cmhost1 cmport1
report   onsoctcp cmhost1 cmport2
payroll onsoctcp cmhost1 cmport3
```

Cluster sqlhosts file:

```
g_ha group - - i=10
ids onsoctcp host1 port1 g=g_ha
ids_hdr onsoctcp host2 port2 g=g_ha
ids_sds onsoctcp host3 port3 g=g_ha
```

Connection Manager Arbitrator (CMA)

- Zajišťuje automatické přepnutí některého ze sekundárních uzlů do *primary* (on-line) stavu
- Výběr nejvhodnějšího sekundárního uzlu pro přepnutí do primary stavu se provádí na základě FOC (Fail Over Configuration) definice
- FOC *pořadí uzlů*, *časový interval*
  - Pořadí uzlů – pořadí, ve kterém dochází k automatickému přepínání sekundárních uzlů na on-line primary
  - Časový interval – doba v sekundách, po kterou arbitrátor čeká, zda nedostane od serveru odpověď



Pravidla pro FOC

- Seznam sekundárních uzlů je oddělený znakem ‘+‘
- Pokud některé uzly jsou odděleny závorkami, fail over se provádí v pořadí
  - konkrétní uzel, SDS, HDR, RSS
- Příklad:

```
FOC node1+(SDS+node2+HDR+node3)+node4+RSS,10
```

- Pokud není FOC explicitně definovaný, platí pravidlo

```
FOC SDS+HDR+RSS,0
```

- Pokud primární server v časovém intervalu neodpoví, arbitrátor ověří jeho nedostupnost ještě dalšími alternativními komunikačními kanály klástru
- Nastavení `DRAUTO 3` zajišťuje, že v klástru bude pouze jeden primární uzel

Konfigurace oncmsm a příklady použití

- Konfigurační soubor

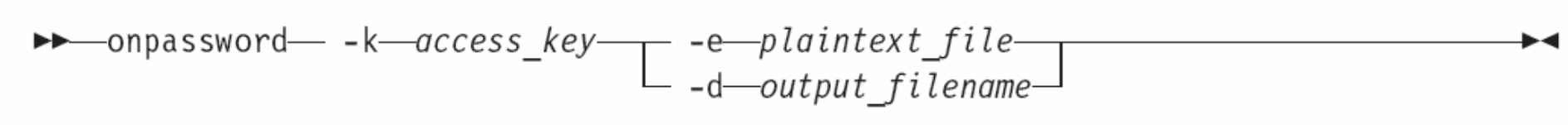
```
NAME ConnectionManagerName
SLA  name=value
SLA  name=value
FOC  failover_configuration,timeout_value
DEBUG        1/0
LOGFILE      <path to log file>
```

- Default jméno a umístění **$INFORMIXDIR/etc/cmsm.cfg**

```
% oncmsm
% oncmsm –c /path/to/config/file
% oncmsm  cm1 -s oltp_cm1=primary –s report_cm1=HDR+SDS
% oncmsm -s oltp=primary -s payroll=HDR+primary -s report=SDS+HDR -l cm.log
% oncmsm  cm1 -s oltp_cm1=primary –s report_cm1=HDR+SDS –f serv1+(serv2+SDS)+HDR+RSS,10
% oncmsm –k cm1
% oncmsm –k oltp
```

onpassword

- Zajišťuje šifrované uložení hesel uživatelů pro autentizaci mezi servery pro Connection Manager a ER




```
ServerName_1 AlternateServer_1        UserName_1       Password_1
ServerName_2 AlternateServer_2        UserName_2       Password_2
lx-rama              lx-rama                   ravi
    foobar
toru         toru                     usr2             fivebar
seth         seth                     fred             9ocheetah
cheetah              panther                   anup
    c0mpl1cate
```

- Příklad šifrování a dešifrování

```
onpassword –k 6azy78op –e my_passwd_file
onpassword –k 6azy78op –d my_passwd_file
```

- Šifrovaný soubor je uložen v `$INFORMIXDIR/etc/passwd_file`

Praktická ukázka