Sebastian **Biedroń**

UNIX, WIRTUALIZACJA, BEZPIECZEŃSTWO

PODRĘCZNIK ADMINISTRATORA

Helion

Redaktor prowadzący: Małgorzata Kulik
Projekt okładki: Paweł Kowalski
Opracowanie okładki: Studio Gravite / Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie/aixpow
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-283-3672-8

Printed in Poland.

Spis treści

Rozdział 3.
Wirtualizacja systemu AIX

Początki masowej wirtualizacji systemów mamy już daleko za sobą. Dziś zdecydowana większość systemów operacyjnych podlega wirtualizacji. Prawie każdy z nas ma w swoim laptopie jakieś maszyny wirtualne, które może w każdej chwili uruchomić zarówno w celach edukacyjnych, jak i po to, by wykonać część swojej pracy w izolowanym środowisku. Podobnie jest w przypadku zastosowań komercyjnych i wirtualizacji serwerów.

Na rynku umownie nazywanym „x86" mamy wiele potencjalnych produktów do wirtualizacji serwerów. Możemy wybierać pomiędzy VMware ESXi hypervisor, Hyper-V, KVM czy XEN. Natomiast na rynku platform UNIX-owych (jak IBM POWER czy Oracle SPARC) wybór mamy ograniczony i możemy użytkować jedynie wirtualizatory dostarczane przez dostawcę platformy.

Każde z wyżej wymienionych rozwiązań ma wady i zalety. Oczywistą wadą w przypadku platformy Power jest ograniczony wybór metod wirtualizacji. Główną i przeciwstawną zaletą dla wymienionej wady jest lepsza integracja wirtualizatora z platformą. Przekłada się to na wyższą wydajność i dostępność systemów oraz wyższy poziom ich bezpieczeństwa.

Platforma Power dostarcza dwa rodzaje wirtualizatorów. Zaawansowany i dostępny od wielu lat wirtualizator PowerVM oraz proste rozwiązanie będące ukłonem w stronę świata linuksowego, czyli PowerKVM. Dodatkowo można wykorzystać wirtualizację na poziomie systemu operacyjnego AIX, używając WPAR-ów (*Workload Partitions*). W tym rozdziale będzie omawiany najbardziej popularny rodzaj wirtualizacji na platformie Power, czyli PowerVM.

Do zrozumienia niniejszego rozdziału wskazane jest posiadanie podstawowej, ogólnej wiedzy dotyczącej zagadnień związanych z wirtualizacją systemów operacyjnych. W przypadku zainteresowania głównie aspektami dotyczącymi systemu operacyjnego można ten rozdział pominąć i wrócić do niego w terminie późniejszym.

PowerVM

PowerVM to najbardziej rozpoznawana i dająca największe możliwości metoda wirtualizacji spośród wszystkich trzech wymienionych powyżej. Swoje początki jako metoda wirtualizacji systemów AIX miała kilkanaście lat temu. Jeśli spojrzeć głębiej na genezę tej metody, to czerpała ona z rozwiązań istniejących na platformach mainframe.

Opisywana platforma wirtualizacji cechuje się bardzo dobrą integracją zarówno z platformą sprzętową, jak i z systemem operacyjnym. Niewątpliwą przyczyną takiej głębokiej integracji jest tworzenie wszystkich tych warstw w jednej firmie, gdzie przepływ informacji i wzajemne interesy są zbieżne.

Bardzo ważną cechą tej wirtualizacji jest dowolnie skalowalna wydajność. Jest ona do osiągnięcia dzięki jednej z ważnych cech — możliwości przydzielania zwirtualizowanym systemom zasobów zarówno wirtualnych, jak i fizycznych. Dzięki temu jeżeli potrzebujemy bardzo dużej przepustowości sieci LAN czy SAN, możemy przydzielić dowolną liczbę fizycznych kart do obsługi tego ruchu.

Na chwilę obecną (rok 2017) wirtualizacja sprzedawana jest w dwóch wersjach: *standard* i *enterprise*. Różnią się one tylko kilkoma dodatkowymi zaawansowanymi cechami dostępnymi w wyższej wersji: *Active Memory Sharing*, *Live Partition Mobility* i *PowerVP Performance Monitor*. Różnice pomiędzy wersją standard i enterprise przedstawia tabela 3.1.

Tabela 3.1. *PowerVM standard kontra enterprise*

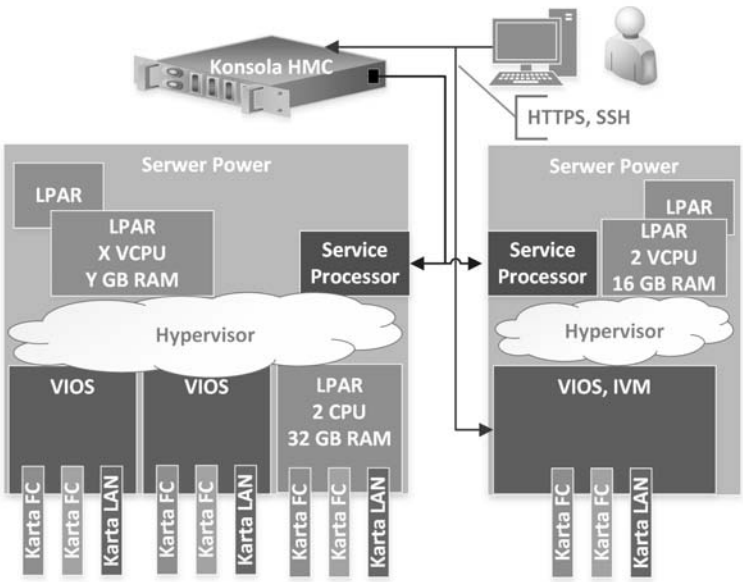
Cecha	**Wersja PowerVM**	
	Standard	***Enterprise***
Virtual I/O Server	Ok (DUAL VIOS)	Ok (DUAL VIOS)
Suspend/Resume	OK	OK
NPIV	OK	OK
Shared Processor Pools	OK	OK
Shared Storage Pools	OK	OK
Thin Provisioning	OK	OK
Active Memory Sharing	**BRAK**	OK
Live Partition Mobility	**BRAK**	OK
PowerVP Performance Monitor	**BRAK**	OK
SR-IOV	OK	OK

Funkcjonalności przedstawione w tabeli są opisane w dalszej części rozdziału dotyczącego wirtualizacji.

Części składowe wirtualizacji

Kluczowym dla zrozumienia sposobu działania wirtualizacji PowerVM jest rozebranie całego mechanizmu na części bazowe pełniące różne funkcje w całym rozwiązaniu. Ukazane jest to na rysunku 3.1.

Rysunek 3.1. *Części składowe wirtualizacji*

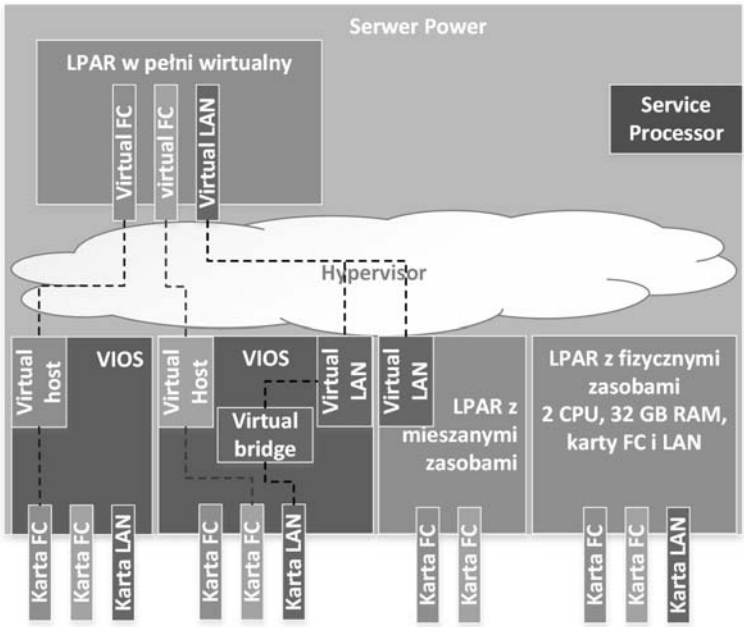


Podstawowe części służące do skutecznego działania wirtualizacji to:

LPAR, ***logiczna partycja***, ***partycja***, ***mikropartycja***, ***SPLPAR*** (***Shared Processor Logical partition***) — wymienione określenia często używane są zamiennie. Oznaczają wirtualny wycinek serwera uruchomiony na platformie Power, czyli to samo, co w świecie x86 oznacza określenie „wirtualna maszyna". Dany LPAR posiada określone zasoby wirtualne: procesor, pamięć, urządzenia I/O (dyski, LAN, SAN), może posiadać urządzenia fizyczne: fizyczny procesor, fizyczne karty IO. Taki LPAR można uruchomić przy użyciu interfejsu zarządzającego (konsoli) i zainstalować na nim system operacyjny. Z punktu widzenia logiki VIOS opisywany poniżej jest LPAR-em z zainstalowanym odpowiednim oprogramowaniem.

Historycznie mikropartycja to nowsze pojęcie niż LPAR i wskazujące na zmniejszenie granulacji zasobów przydzielanych do wirtualnego tworu. W pierwszych implementacjach wirtualizacji granulacja była na poziomie całego procesora. Później zaczęła schodzić na poziom ułamków. W dniu dzisiejszym wymienione wyżej pojęcia oznaczają potocznie to samo i w taki sposób będą dalej używane w tej publikacji.

Unikalną cechą wirtualizacji platformy Power jest to, że LPAR może być w pełni zwirtualizowany, jak i może posiadać komponenty fizyczne (dedykowane core'y, karty I/O do obsługi LAN i SAN). Różne rodzaje LPAR-ów ukazuje rysunek 3.2. Ważną cechą jest możliwość instalacji innych niż AIX systemów, takich jak IBM i — historycznie znany pod nazwą OS/400 — czy Linux (Red Hat, SLES, Ubuntu).



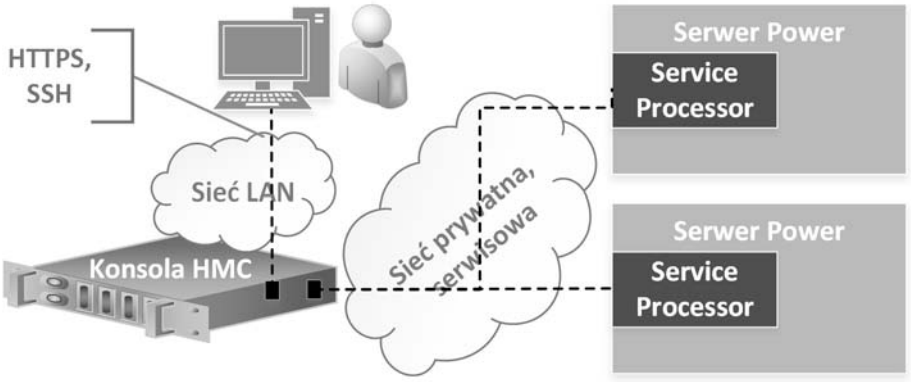
Rysunek 3.2. *Różne rodzaje LPAR-ów*

Warto nadmienić, że konfiguracja LPAR-a przechowywana jest w dwóch miejscach: w profilu partycji i w jej „bieżącej konfiguracji". Informacja jest o tyle istotna, że parametry z jednego i z drugiego miejsca nie muszą się pokrywać. Gdy na przykład wykonana zostanie operacja dołożenia procesora, bieżąca konfiguracja będzie odzwierciedlała tę operację, a profil nie, dopóki jawnie go nie zaktualizujemy.

Konsola HMC (*Hardware Management Console*) to konsola „sprzętowa", służąca do zarządzania jednym lub wieloma serwerami zarówno pod względem ich konfiguracji, utrzymania, jak i z punktu widzenia części funkcji wirtualizacji. Tego typu konsola nie jest elementem koniecznym do posiadania zwirtualizowanego środowiska Power, jednak gdy konieczne staje się zarządzanie wieloma serwerami i użycie bardziej zaawansowanych funkcji wirtualizacyjnych, staje się elementem pożądanym.

W rzeczywistości konsola HMC jest zazwyczaj tzw. appliance'em, czyli fizycznym serwerem x86 z odpowiednią ilością portów LAN i zainstalowanym oprogramowaniem zarządzającym. Konsola HMC może być również instalowana jako maszyna wirtualna na hypervisorze na platformie x86. Sposób współdziałania użytkownika z konsolą oraz połączenia konsoli z serwerami prezentuje rysunek 3.3.

IVM (*Integrated Virtualization Manager*) to konsola „programowa" instalowana jako część VIOS-a w przypadku nieposiadania konsoli HMC. Jest wygodna w użyciu i wystarczająca dla małych, niewymagających środowisk. Użycie IVM sprawdza się w takich środowiskach pomimo posiadania pewnych ograniczeń w stosunku do konsoli HMC. Różnice w funkcjonalnościach dostarczanych przez konsole przedstawia tabela 3.2.



Rysunek 3.3. *Konsola HMC, schemat połączeń*

Tabela 3.2. *Różnice w funkcjonalności HMC i IVM*

Funkcjonalności	HMC	IVM
Wirtualizacja, tworzenie wielu LPAR-ów na serwerze	TAK	TAK
Wiele serwerów podłączonych pod jedną konsolę	TAK	NIE
Przydzielanie bezpośrednio komponentów fizycznych (karty LAN, SAN...) do LPAR-a	TAK	NIE
Instalacja typu DUAL VIOS	TAK	NIE
Funkcjonalność Live Partition Mobility	TAK (HMC-HMC)	TAK (IVM-IVM)
Wykorzystanie technologii NPIV	TAK	TAK
Konfiguracja Shared Ethernet Adaptera (SEA) z opcją HA	TAK (dual VIOS)	NIE

„Podstawowy" hypervisor — podstawowe funkcje hypervisora pełni lekki kod wbudowany w serwer, będący częścią jego oprogramowania „sprzętowego", tzw. firmware. Zadaniem tej części jest wirtualizacja zasobów procesora, pamięci i wewnętrznej części sieci LAN. Można tu użyć słowa „podstawowy", gdyż niecała wirtualizacja bazuje na tej części hypervisora. „Podstawowy" hypervisor nie wirtualizuje urządzeń I/O. Do wirtualizacji I/O używana jest oddzielna część nazywana Virtual I/O Serverem (VIOS).

Odseparowanie części wirtualizacji procesora i pamięci od I/O ma duży sens w kontekście wydajności i dostępności serwera. Dzięki krótkiemu kodowi znajdującemu się bliżej „serca" maszyny, czyli w firmware, serwer jest w stanie wydajniej zarządzać zasobami, będąc jednocześnie mniej podatnym na awarię. Dzięki zaawansowanym funkcjom niektórych modeli serwerów możliwe jest włączenie mirroringu hypervisora na poziomie pamięci RAM, co również może znacząco wpłynąć na jego dostępność. Z drugiej strony, natura operacji I/O pozwala na zdublowanie VIOS-ów, czyli części odpowiedzialnej za wirtualizację I/O.

VIOS (*Virtual I/O Server*) to część odpowiedzialna za wirtualizację operacji I/O. VIOS-y udostępniają LPAR-om sieć LAN i przestrzeń dyskową. W rzeczywistości twór ten jest LPAR-em z zainstalowanym oprogramowaniem, które z kolej można określić mianem customizacji systemu AIX dla pełnienia funkcji serwera I/O. VIOS-ów na jednym serwerze może być wiele, chociaż z reguły stosowane są 1 albo 2, gdy

chcemy zapewnić maksymalną dostępność systemów i umożliwić sobie aktualizację Virtual I/O Serverów bez konieczności wyłączania LPAR-ów klienckich. Funkcjonalność VIOS-a zastanie opisana szerzej w dalszej części rozdziału.

Wirtualizacja procesora

Wirtualizacją przydziału procesora i pamięci zarządza część potocznie nazwana „podstawowym" hypervisorem. Jak wspomniano wcześniej, hypervisor jest częścią integralną serwera, dostarczaną wraz z jego oprogramowaniem sprzętowym, tzw. firmware. Dzięki bliskości kodu wirtualizacji i sprzętu, a jednocześnie stosunkowo niewielkim rozmiarom tego kodu może działać on wydajniej i stabilniej od rozwiązań działających bez tej bliskości.

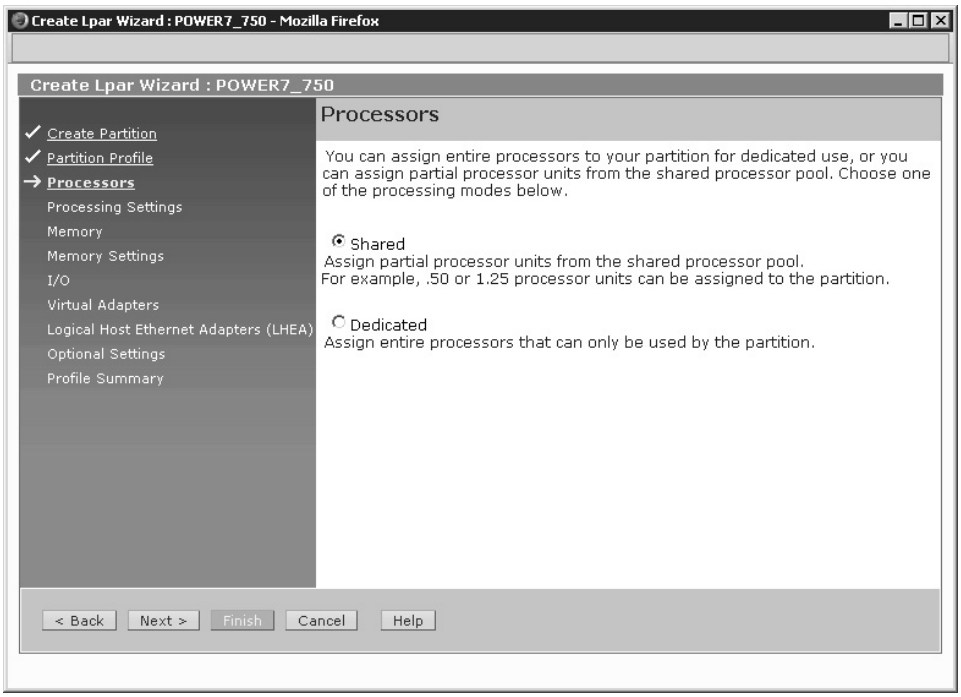
Wirtualizacja procesora ma tę przewagę nad zastosowaniem procesora fizycznego, że można lepiej zutylizować całą platformę. Obciążenie dużego serwera, posiadającego wiele core'ów, na którym działają dziesiątki LPAR-ów, często dochodzi do 60 – 80%, przy czym zachowuje on stabilność wydajnościową. W przypadku zastosowania odpowiedników w postaci serwerów fizycznych ich wykorzystanie często oscyluje w granicach 10 – 20%. Ważną cechą jest również możliwość dynamicznej reakcji na zwiększające i zmniejszające się obciążenie. W większości przypadków odebranie lub dołożenie kolejnych procesorów do systemu to kilka kliknięć niewymagających jego restartu. Taka sama czynność w przypadku systemów fizycznych oznacza konieczność rozbudowy serwera lub migracji na inny, wydajniejszy.

Ważną i unikalną cechą wirtualizacji PowerVM jest możliwość przypisywania LPAR-owi procesora „fizycznego". Tryb działania w tym przypadku nosi nazwę Dedicated. Polega on na przypisaniu całej mocy procesora/procesorów do LPAR-a. Wpływ na to, który fizyczny procesor ma zostać przypisany, jest ograniczony z punktu widzenia administratora, decyduje o tym hypervisor. Podejmuje on decyzję w taki sposób, aby optymalizować wydajność pracy LPAR-a i eliminować potencjalne problemy, np. dużą odległość procesora od pamięci (tzw. *affinity*).

Warto przy tej okazji uspójnić definicję procesora, jaką posługuje się IBM. Otóż w przypadku serwerów Power termin „procesor" oznacza coś innego niż np. w świecie x86. W świecie x86 wyraźnie rozróżnia się procesor (*socket*) i rdzeń, więc idąc za tym tokiem rozumowania, mówimy, że jeden procesor Intel I7 posiada 4, 6, 8 rdzeni. W przypadku platformy Power, mówiąc „procesor", mamy na myśli rdzeń procesora, o ile jawnie nie jest powiedziane, że chodzi o *socket* lub chip (w jednym sockecie może być wiele chipów, czyli patrząc funkcjonalnie — wiele procesorów). Należy zawsze mieć ten fakt na uwadze, czytając dokumentację IBM.

Parametry procesorów wirtualnych

LPAR posiada szereg parametrów, które można definiować podczas jego tworzenia i zmieniać później podczas jego działania. Parametry te mają duży wpływ na sposób pracy systemu i przydział mocy do LPAR-ów. Dwa podstawowe tryby działania to tryb dedykowany (*Dedicated*) i współdzielony (*Shared*). Ekran wyboru trybów podczas tworzenia LPAR-a prezentuje rysunek 3.4.



Rysunek 3.4. *Tryb przydziału procesora z konsoli HMC: Shared/Dedicated*

Tryb dedykowany (Dedicated)

Tryb dedykowany oznacza przypisanie całych „fizycznych" procesorów (core'ów) do LPAR-a. Jeżeli LPAR nie używa procesora, to moc procesora jest tracona. Z poziomu monitoringu systemu operacyjnego widoczne to będzie jako bezczynne cykle procesora (`%idle`). Jeżeli nie chcemy tracić mocy procesora dedykowanego, możemy określić jego tryb działania jako *donating*. Takie ustawienie oznacza, że niewykorzystany czas pracy procesora będzie oddawany do użytku innych LPAR-ów na maszynie, o ile LPAR nie jest mocno obciążony. Domyślnie LPAR obciążony w 80% nie będzie oddawał czasu procesora do puli. Sterować tą wartością można za pomocą ustawienia `ded_cpu_donate_thresh`, które można zmienić poleceniem `schedo` z poziomu systemu operacyjnego.

W trybie *Dedicated* określamy jeszcze trzy szczegółowe parametry działania:

1. ***Minimum processors*** — minimalna liczba procesorów, z jaką może działać system. Jeżeli podczas próby uruchomienia LPAR-a nie będzie dostępna minimalna liczba procesorów, to LPAR się nie uruchomi i zgłosi błąd.
2. ***Desired processors*** — wartość pożądana. LPAR zawsze uruchomi się z określoną tutaj liczbą procesorów. Jedynym wyjątkiem jest brak wolnych zasobów na serwerze w ilości „desired" w momencie uruchamiania partycji. W tym przypadku LPAR uruchomi się z najwyższą dostępną i możliwą do zaalokowania liczbą procesorów powyżej wartości minimum. Wartość tę można zmieniać dynamicznie podczas pracy systemu w zakresie od minimum do maksimum.

3. ***Maximum processors*** — maksymalna liczba procesorów, które mogą zostać przypisane do LPAR-a bez restartu systemu operacyjnego. Obrazowo rzecz ujmując, jeżeli partycja ma 2 procesory (*desired*) widoczne w systemie i 4 procesory ustawione jako wartość maksymalna, to administrator może z konsoli zarządczej dołożyć kolejne 2 procesory, zwiększając bieżącą moc partycji (bez restartu, a cała czynność zajmuje 30 sekund). Natomiast w przypadku konieczności zwiększenia mocy LPAR-a powyżej wartości maximum należy:
   - Zmienić wartość maximum w profilu.
   - Zamknąć LPAR.
   - Uruchomić LPAR ze zmienionego profilu z nową wartością maximum.

Po co ta wartość w ogóle istnieje? Czy nie można by zastosować reguły, że każdy LPAR ma domyślnie ustawioną wartość maximum na 256 (maksymalna liczba procesorów obsługiwana przez AIX)? Otóż problem ma dwie natury:

1. **Wydajnościową** — system operacyjny, aby obsłużyć dynamiczne dodanie zasobów takich jak procesor czy pamięć, musi być do tego gotowy i muszą istnieć odpowiednie struktury w systemie operacyjnym. Struktury te budowane są podczas startu systemu operacyjnego (proces bootowania). Tymczasem dla zapewnienia optymalnego działania nie jest celowe budowanie struktur do obsługi procesorów i pamięci, jeżeli one nigdy nie zostaną wykorzystane. Takie podejście powodowałoby marnotrawstwo zasobów i miało negatywny wpływ na wydajność systemu.
2. **Licencyjną** — niektórzy dostawcy oprogramowania mogą wymagać ustawienia wartości maximum danego LPAR-a na liczbę odpowiadającą posiadanym licencjom danego oprogramowania. Częściowo utrudniłoby to swobodne dodawanie zasobów i łamanie tym samym praw licencyjnych. Na chwilę obecną najwięksi dostawcy oprogramowania nie przykładają wagi do tej wartości. Jednak polityka licencyjna jest „polityką" w pełnym znaczeniu tego słowa. Nie podlega logice i może w każdej chwili ulec zmianie.

Wartość maximum należy ustawiać z rozsądkiem — jeśli jest to możliwe, trzeba zapewniać zapas na modyfikację zasobów w przypadku problemów wydajnościowych.

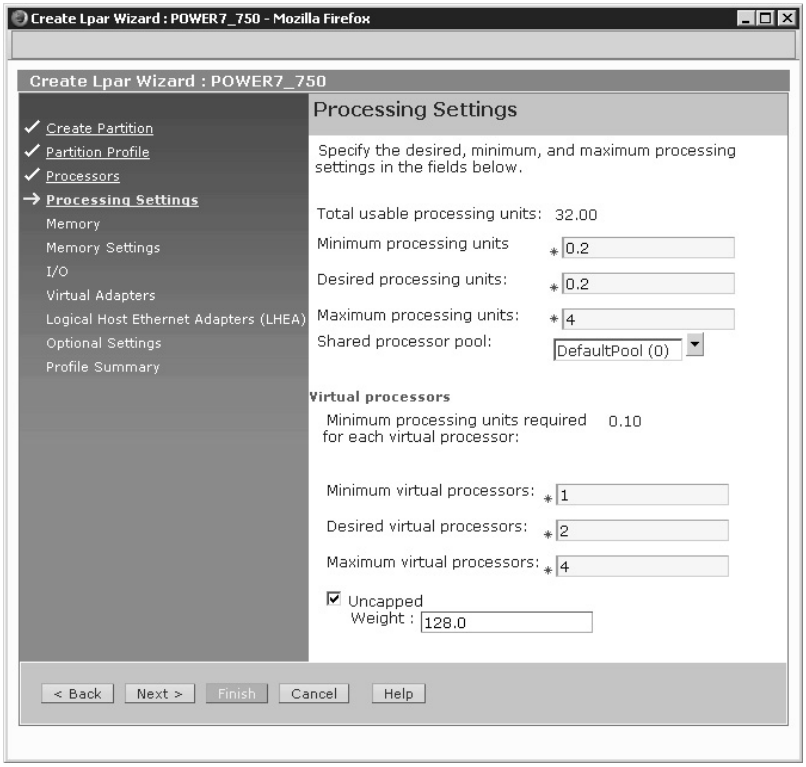
Tryb dedykowany ma podstawową wadę w postaci zawłaszczania i marnowania niewykorzystanych zasobów. Wadę tę częściowo redukuje możliwość wywłaszczania nieużywanych dedykowanych procesorów (tryb donating). Funkcjonalność tę można włączyć w dowolnym momencie, jednak nawet w tym przypadku część cykli procesora będzie niewykorzystywana. Jeśli chodzi o zalety trybu Dedicated, to gwarantuje on stałą wydajność niezakłóconą mechanizmami wirtualizacji. Można odczuć poprawę wydajności w przypadku działania w tym trybie, szczególnie jeżeli serwer jest bardzo dynamicznie używany i wiele LPAR-ów było tworzonych, usuwanych i modyfikowanych. Według dokumentacji różnica w wydajności pomiędzy systemem używającym procesorów dedicated a shared może wynosić około 7 – 10%. Natomiast należy wziąć pod uwagę fakt, że osiągnięty wynik w dużym stopniu zależy od liczby LPAR-ów na serwerze, jego całkowitego obciążenia i poprawności rozłożenia LPAR-ów (tzw. affinity).

Tryb ten jest do wykorzystania głównie dla systemów, które charakteryzują się dużym i mniej więcej stałym użyciem zasobów procesora. Zazwyczaj w dużych instytucjach można znaleźć do kilku procent takich systemów. Chcąc optymalnie utylizować całą platformę, należy go stosować ostrożnie i z wyczuciem (raczej rzadko).

Tryb współdzielony (Shared)

Tryb współdzielony oznacza przypisanie wirtualnych procesorów do LPAR-a. Mogą one używać cykli różnych procesorów z puli procesorów dostępnych, niemniej jednak maksymalna moc procesora wirtualnego nigdy nie przekroczy mocy procesora fizycznego.

Ten tryb pracy daje najwięcej korzyści w postaci dobrej utylizacji całej platformy. Jednocześnie udostępnia sporo parametrów, które można wykorzystać do sterowania jego pracą i przydziałem zasobów. Parametry możliwe do modyfikacji prezentuje rysunek 3.5.



Rysunek 3.5. *Konsola HMC, parametry trybu shared*

Opis parametrów:

- ***Minimum virtual processors***, ***Desired virtual processors***, ***Maximum virtual Processors*** — parametry zbliżone funkcjonalnie do ich odpowiedników z trybu dedykowanego, z tą różnicą, że w tym przypadku mówimy o procesorach wirtualnych, a nie fizycznych.

- ***Minimum processing units*** — minimalna ilość mocy procesora, jaką może posiadać ten LPAR. Oznacza to, że jeżeli na serwerze nie ma dostępnej tej ilości mocy procesora, to LPAR się nie uruchomi. Z drugiej strony, odbierając dynamicznie moc procesora, nie odbierzemy jej poniżej tego poziomu. Poziom jest definiowalny z dokładnością do 1/100 mocy procesora (minimum 1/10 CPU na 1 wirtualny procesor). Parametr może być zmieniany, jednak jego zmiana wymaga ponownego wczytania profilu LPAR-a, czyli jego zamknięcia i ponownego uruchomienia.
- ***Desired processing units*** — wartość pożądana, LPAR zawsze uruchomi się z określoną tutaj mocą procesora. Jedynym wyjątkiem jest brak wolnych zasobów na serwerze w ilości *desired* w momencie uruchamiania partycji. W tym przypadku LPAR uruchomi się z najwyższą dostępną i możliwą do zaalokowania mocą procesora, nie mniejszą niż wartość minimum. Wartość tę można zmieniać dynamicznie podczas pracy systemu w zakresie od minimum do maximum z dokładnością do 1/100 mocy procesora (minimum 1/10 CPU na 1 wirtualny procesor).

  Wartość ta ma duże znaczenie, gdyż LPAR, nawet działając w trybie *uncapped*, nigdy nie dostanie mniej mocy, niż ona wynosi. Jest to gwarancja mocy obliczeniowej dla partycji. Z drugiej strony, nie należy myśleć, że jest ona na stałe przypisana do LPAR-a. Należy ją rozumieć jako gwarancję mocy, którą LPAR dostanie na 100%, jeżeli będzie miał takie zapotrzebowanie. Natomiast jeżeli nie będzie tej mocy potrzebował, zostanie ona przekazana przez hypervisor do puli wolnych zasobów i w razie potrzeby przydzielona innemu LPAR-owi.
- ***Maximum processing units*** — maksymalna moc procesorów, która może zostać dołożona do LPAR-a bez restartu systemu operacyjnego. W przypadku konieczności zwiększenia mocy partycji powyżej wartości maximum należy zmienić wartość maximum w profilu, zamknąć LPAR i uruchomić LPAR ze zmienionego profilu z nową wartością maximum. Wartość tę można zmieniać z dokładnością do 1/100 mocy procesora (minimum 1/10 CPU na 1 wirtualny procesor).
- ***Uncapped/Capped*** — o ile nie zaznaczy się trybu *uncapped*, LPAR działa w trybie *capped*. Różnice najlepiej przedstawić na przykładzie dwóch LPAR-ów. Załóżmy, że posiadamy dwie partycje założone w trybie shared o parametrach przedstawionych w tabeli 3.3.

Tabela 3.3. *Przykładowe LPAR-y capped/uncapped*

	Virtual Processors			***Processing Units***			
	min	***desired***	***max***	***min***	***desired***	***max***	***cap/uncap***
LPAR1	1	2	4	0,1	**0,5**	4	***Capped***
LPAR2	1	2	4	0,1	**0,5**	4	***Uncapped***

Do pewnego stopnia oba LPAR-y zachowują się podobnie, można im dynamicznie (bez konieczności restartu) dodawać lub odbierać zasoby procesorowe. W obu przypadkach zmiana parametrów minimum i maximum wymaga ponownego wczytania profilu (zamknięcie LPAR-a, zmiana profilu,

uruchomienie partycji ze zmienionego profilu). Jednakże podstawowa różnica polega na tym, że:

- LPAR1 działający w trybie *capped* i mający wartość *Desired processing units* 0,5 nigdy nie zutylizuje więcej mocy niż 0,5 CPU, chyba że jawnie zmienimy jego tryb pracy na uncapped lub jawnie dodamy mu mocy procesora.
- LPAR2 działający w trybie *uncapped* i posiadający wartość *Desired processing units* 0,5 w miarę zapotrzebowania będzie mógł zutylizować moc odpowiadającą wartości *Desired virtual processors*, czyli 2 CPU. Oczywiście wyjście partycji LPAR2 powyżej wartość 0,5 będzie możliwe, ale tylko jeżeli serwer posiada dostępny zapas mocy procesora, a pula zasobów (*Shared procesor pool*), do której należy LPAR, posiada nieprzydzielone zasoby procesora.

Można zadać pytanie, czy w takim razie nie lepiej tworzyć wszystkie LPAR-y w trybie uncapped, żeby zostawić furtkę w przypadku nietypowego i trudno przewidywalnego zapotrzebowania na moc? Odpowiedź zazwyczaj jest twierdząca. Dla optymalizacji wykorzystania platformy jest to kluczowy mechanizm. Jedynym ważnym czynnikiem przeciw wykorzystaniu trybu *uncapped* jest aspekt licencyjny. W przypadku LPAR1 wystarczy licencja na 0,5 CPU (o ile dany dostawca oprogramowania wspiera licencjonowanie cząstkowe — *subcapacity*), natomiast w przypadku LPAR2 zapotrzebowanie na licencję wykorzystywanego oprogramowania wynosi 2 CPU, gdyż tyle wynosi maksymalna utylizacja, jaką może osiągnąć LPAR2 bez zmian konfiguracyjnych.

Warto przy okazji zwrócić uwagę, że konfiguracja LPAR1, wykorzystana w celu pokazania mechanizmu działania, jest ustawieniem niezgodnym z najlepszymi praktykami w kontekście wydajności. W przypadku trybu capped należy stosować minimalną liczbę procesorów *desired*.

- ***Weight*** — waga, priorytet. Priorytet jest wartością nieistotną, o ile utylizacja serwera nie osiąga 100%. Do tego momentu każdy LPAR działający w trybie uncapped ma przydzielone tyle mocy, ile potrzebuje. Dopiero wtedy, gdy w danej chwili serwer utylizuje 100% zasobów procesora, uwidaczniają się algorytmy hypervisora, które powodują przydział mocy procesora proporcjonalnie do przydzielonej LPAR-owi wagi. Priorytetyzacja, obok liczby wirtualnych procesorów i wartości desired, jest najważniejszym sposobem wpływania na podział mocy w przypadku dojścia obciążenia serwera do 100%.

Parametru Weight warto rozsądnie używać, np. mocno dystansując priorytet systemów testowych i produkcyjnych czy też dystansując systemy pod względem ich krytyczności i ewentualnych kosztów, jakie niesie ze sobą niedotrzymanie terminów i tempa przetwarzania.

Pule procesorowe (Shared processor pool)

Shared processor pool — pula procesorowa. Wirtualny twór określający zbiór procesorów, a w zasadzie określający moc wyrażaną w liczbie procesorów, gdyż nie jest związany z konkretnymi procesorami fizycznymi. Każdy tworzony LPAR należy do jakiejś puli procesorowej. Jeżeli nieistotne jest to, do której puli ma trafić, to trafia do puli *default* określającej całe zasoby serwera.

Pula procesorowa jest tworem umożliwiającym optymalizację wykorzystania licencji. Liczni dostawcy oprogramowania, a szczególnie najwięksi z nich, uwzględniają pule procesorowe w swoich modelach licencjonowania produktów. Działanie puli procesorowej najlepiej omówić na przykładzie. Załóżmy, że posiadamy serwer wyposażony w 32 core'y (w nomenklaturze serwerów Power 1 procesor równa się 1 core). Na tym serwerze chcemy uruchomić 4 LPAR-y po 4 procesory z zainstalowaną bazą danych DB2, a posiadamy licencje tylko na 8 core'ów. Opisany przykład przedstawia tabela 3.4.

Tabela 3.4. *Przykładowe LPAR-y w puli procesorowej*

	Virtual Processors			***Processing Units***					
	min	***desired***	***max***	***min***	***desired***	***max***	***Cap/ uncap***	***Weight***	***Pool***
LPAR1	1	**4**	4	0,1	**1**	4	*Uncapped*	64	**8 CPU**
LPAR2	1	**4**	4	0,1	**1**	4	*Uncapped*	64	
LPAR3	1	**4**	4	0,1	**1**	4	*Uncapped*	128	
LPAR4	1	**4**	4	0,1	**1**	4	*Uncapped*	128	

Bez zastosowania puli procesorowej na taką konfigurację potrzebnych byłoby 16 procesorów, a przy zastosowaniu puli potrzebnych jest 8. W przypadku konfiguracji takiej jak w tabeli powyżej wymienionych LPAR-ów dotyczą następujące zasady:

- Sumaryczne obciążenie wszystkich LPAR-ów nigdy nie przekroczy 8 procesorów.
- Każdy LPAR może utylizować w dowolnym momencie do 4 procesorów (*Desired virtual processors*), o ile dostępna jest moc procesorowa na serwerze i utylizacja puli nie osiągnęła 8 procesorów.
- Minimalną gwarantowaną na poziomie całego serwera mocą, jaką otrzyma LPAR, jest 1, czyli wartość parametru *Desired processing units*.
- Jeżeli wszystkie LPAR-y w tym samym czasie będą wykazywały maksymalne zapotrzebowanie na moc przetwarzania i serwer będzie obciążony w 100%, to zacznie działać parametr *Weight* (priorytet). Skutek będzie taki, że (w przybliżeniu) hypervisor podzieli moc między LPAR-y proporcjonalnie do priorytetu (z dokładnością do działania algorytmów hypervisora). Spodziewany efekt będzie następujący:
  - LPAR1 = 1,67 (+0,67 CPU ze względu na *weight=64*),
  - LPAR2 = 1,67 (+0,67 CPU ze względu na *weight=64*),
  - LPAR3 = 2,33 (+1,33 CPU ze względu na *weight=128*),
  - LPAR4 = 2,33 (+1,33 CPU ze względu na *weight=128*).

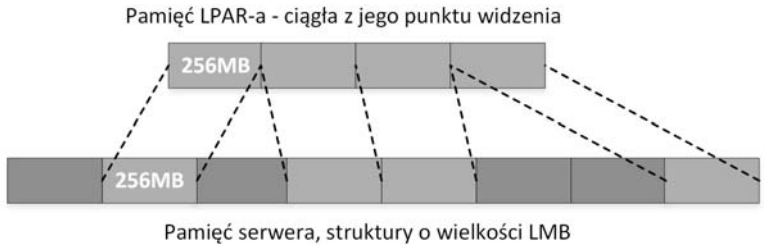
Wirtualizacja pamięci

Podobnie jak jest w przypadku procesora, wirtualizacją przydziału pamięci zarządza część potocznie nazywana „podstawowym" hypervisorem. Jest on częścią integralną serwera, dostarczaną wraz z jego oprogramowaniem sprzętowym, tzw. firmware.

Z punktu widzenia serwera pamięć jest przydzielana każdemu LPAR-owi w trybie dedykowanym. Czyli pamięć fizyczna jest przydzielona LPAR-owi na wyłączność. Odbywa się to blokami o wielkości LMB (*Logical Memory Block*) definiowanymi przy inicjalizacji serwera. Typową wielkością tego bloku jest 256 MB. Jest to również blok, którym operuje się, dodając pamięć do LPAR-a lub odbierając mu ją. Wyjątkiem jest AMS (*Active Memory Sharing*). W tym trybie pamięć przydzielana jest blokami o wielkości 4 KB i może być współdzielona pomiędzy wiele LPAR-ów. AMS będzie opisany w dalszej części rozdziału.

Przydzielona konkretnemu LPAR-owi pamięć w trybie dedykowanym może nie być ciągła z punktu widzenia serwera. Natomiast z punktu widzenia LPAR-a jego pamięć jest tworem ciągłym. Aby można było osiągnąć taki stan, adresy pamięci są wirtualizowane dla LPAR-a. Zjawisko obrazuje rysunek 3.6.

Rysunek 3.6. *Liniowa adresacja pamięci LPAR-a*

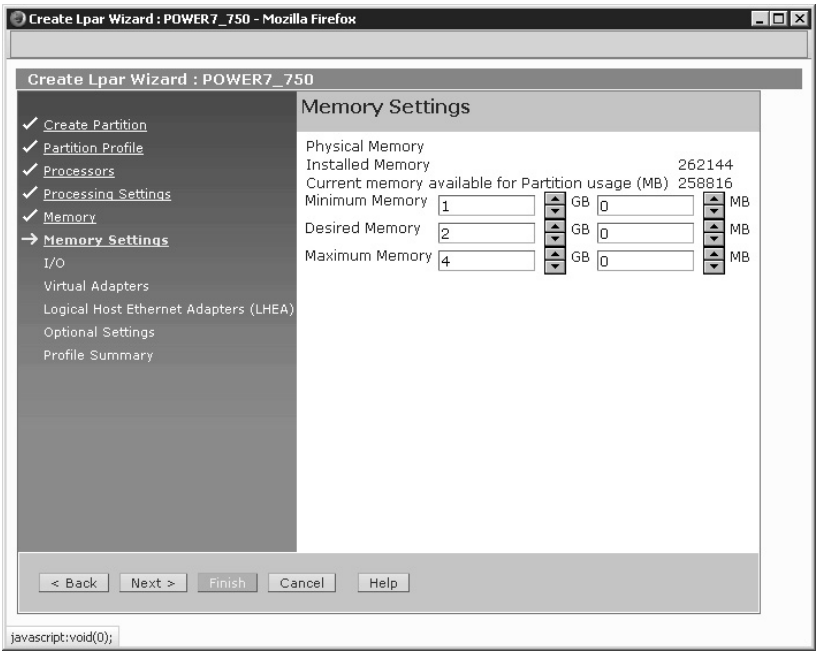


Przy tworzeniu LPAR-a hypervisor dba o to, żeby pamięć stanowiła jednolity blok również z punktu widzenia jej fizycznego położenia. Niemniej jednak wiele operacji zmiany ilości pamięci (dołożenie, odebranie) przeprowadzonych na wielu LPAR-ach na serwerze może prowadzić do dużej niejednolitości. Nie stanowi to problemu, dopóki pamięć znajduje się „blisko" przydzielonego procesora w konstrukcjach wieloprocesorowych.

Odmienną rzeczą w kontekście wirtualizacji pamięci jest pamięć samego hypervisora. Hypervisor korzysta z części pamięci nieudostępnionej dla samych LPAR-ów. W niektórych modelach serwerów istnieje możliwość zwiększenia dostępności rozwiązania poprzez mirroring pamięci hypervisora. Funkcja ta nosi nazwę *Active Memory Mirroring*, można też spotkać nazwę *Hypervisor Memory Mirroring*. Należy zwrócić uwagę, że mirroring pamięci hypervisora dotyczy „podstawowego" hypervisora, a nie dotyczy części odpowiedzialnej za wirtualizację I/O (*Virtual I/O Server*), o której będzie mowa w dalszej części rozdziału. Natomiast operacje I/O mogą być równie dobrze zabezpieczone poprzez utworzenie pary Virtual I/O Serverów obsługujących ten sam ruch sieciowy i dyskowy.

Podstawowe parametry pamięci LPAR-a

Podstawowe parametry pamięci dedykowanej LPAR-a definiowane podczas jego tworzenia prezentuje rysunek 3.7.



Rysunek 3.7. *Podstawowe parametry pamięci — konsola HMC*

Minimum Memory — minimalna wielkość pamięci, z jaką może uruchomić się LPAR w przypadku, gdy nie ma dostępnej pamięci w ilości reprezentowanej parametrem *Desired memory*. Jest to również wielkość pamięci, do której można dynamicznie (bez restartu) zmniejszyć pamięć LPAR-a, jeżeli jest on uruchomiony.

Desired Memory — pożądana wielkość pamięci dla LPAR-a. Z taką wielkością pamięci uruchomi się LPAR, o ile serwer będzie dysponował odpowiednią pulą wolnego RAM-u.

Maximum Memory — maksymalna wielkość pamięci, jaką można przypisać partycji bez konieczności jej restartu. Wartość tego parametru nie powinna być definiowana ze zbyt dużą nadwyżką, gdyż w takim przypadku tracimy realną pamięć w dwóch miejscach:

- Z perspektywy całego serwera można zauważyć, że gdy zwiększamy parametr *Maximum*, część pamięci utylizowanej przez hypervisor ulega zwiększeniu. Tym samym maleje nam wielkość pamięci dostępna dla LPAR-ów (tracimy około 1/64 pamięci zdefiniowanej jako maximum LPAR-a).
- Z perspektywy LPAR-a dostępna pamięć jest zajmowana przez struktury służące do obsługi maksymalnej jej wielkości. Struktury te są tworzone w pamięci operacyjnej podczas procesu bootowania przy uwzględnieniu wielkości *Maximum memory*.

Active Memory Expansion (AME)

Funkcjonalność *Active Memory Expansion* polega na kompresji pamięci. Pomimo że funkcjonalność działa na poziomie systemu operacyjnego, to jest ona definiowana na poziomie konsoli zarządzającej i wymaga posiadania odpowiedniej licencji.

Została ona wprowadzona wraz z rodziną procesorów Power7. Uruchomiona na procesorach Power7+ lub nowszych działa znacznie wydajniej, gdyż procesory te wspierają sprzętowo proces kompresji i dekompresji. Posiadając ograniczoną ilość pamięci, dzięki tej technologii można utworzyć większą liczbę partycji lub partycje o większych zasobach. Oczywiście odbywa się to kosztem procesora, gdyż musi on wykonywać dodatkową pracę. Przy odpowiednio dobranych parametrach kompresji narzuty mogą być nieznaczne i wynosić od kilku do kilkunastu procent.

Definiując LPAR z *Active Memory Expansion*, mamy do dyspozycji dwa parametry:

- ***Active Memory Expansion*** — włączenie/wyłączenie funkcjonalności. Włączenie bądź wyłączenie kompresji pamięci wymaga restartu LPAR-a.
- ***Active Memory Expansion factor*** — współczynnik kompresji pamięci danego LPAR-a mieszczący się w widełkach 1,00 – 10,00. Zmiana współczynnika przy włączonym AME jest wykonywana dynamicznie. Współczynnik określa, ile pamięci będzie widoczne dla systemu operacyjnego. Dla uzyskania tej informacji należy go przemnożyć przez ilość pamięci przydzielonej do systemu z konsoli zarządzającej (*Desired memory*). Przykłady LPAR-ów z różnymi współczynnikami kompresji zawiera tabela 3.5.

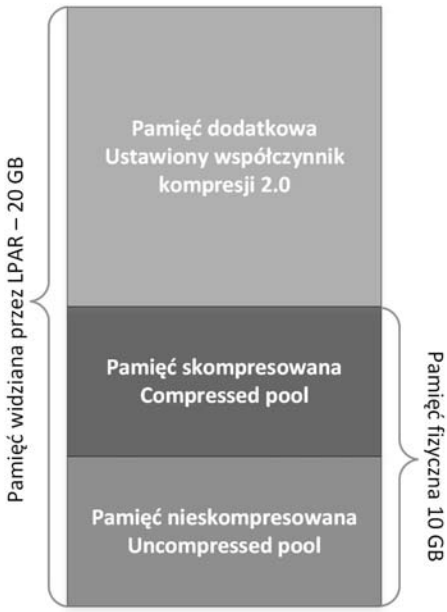
Tabela 3.5. *AME — ilość pamięci przy różnych współczynnikach kompresji*

	Desired Memory (GB)	AME	AME — współczynnik kompresji	Pamięć widziana przez system operacyjny (GB)
LPAR1	**4**	Włączone	1.0	**4·1 = 4**
LPAR2	**4**	Włączone	1.5	**4·1.5 = 6**
LPAR3	**4**	Włączone	1.8	**4·1.8 = 7,2**
LPAR4	**4**	Włączone	3.0	**4·3 = 12**

Aplikacje uruchamiane w systemie operacyjnym nie są świadome działania mechanizmu kompresji pamięci, zatem jej użycie jest dla nich całkowicie przezroczyste. Oczywiście założenie to jest prawdą w przypadku rozsądnego dobrania współczynnika kompresji. Przy aplikacji korzystającej z całej dostępnej pamięci ustawienie wysokiego współczynnika może mieć dramatyczne konsekwencje wydajnościowe. W takim przypadku, chociaż aplikacja nie wie, że system używa kompresji pamięci, to spowolni swoje działanie, czekając na cykle procesora zużywane na ciągłą kompresję i dekompresję.

Od strony systemu operacyjnego praca z kompresją pamięci wymaga podziału dostępnej pamięci na dwie pule. Dla dobrego zobrazowania rozwiązania załóżmy, że LPAR posiadający 10 GB pamięci ma włączone AME ze współczynnikiem kompresji 2.0. Sytuację przedstawia rysunek 3.8.

Rysunek 3.8.
Kompresja pamięci



Uncompressed Pool — pamięć nieskompresowana, na której system operacyjny działa w tradycyjny sposób. Jej wielkość jest zmienna w czasie. Zależy od współczynnika kompresji i ilości pamięci wykorzystywanej przez system operacyjny.

Compressed Pool — pamięć skompresowana, jej wielkość podobnie jak w przypadku Uncompressed Pool jest zmienna w czasie i zależy od ilości skompresowanych danych.

System operacyjny, nawet z bardzo wysokim współczynnikiem kompresji, może nie wykorzystywać *Compressed Pool*, o ile wykorzystywana aktualnie pamięć mieści się w *Uncompressed Pool*.

W przypadku gdy *Uncompressed Pool* nie jest w stanie pomieścić używanej pamięci, AME kompresuje najrzadziej używane strony pamięci i przerzuca je do *Compressed Pool*, uwalniając tym samym pamięć z puli nieskompresowanej.

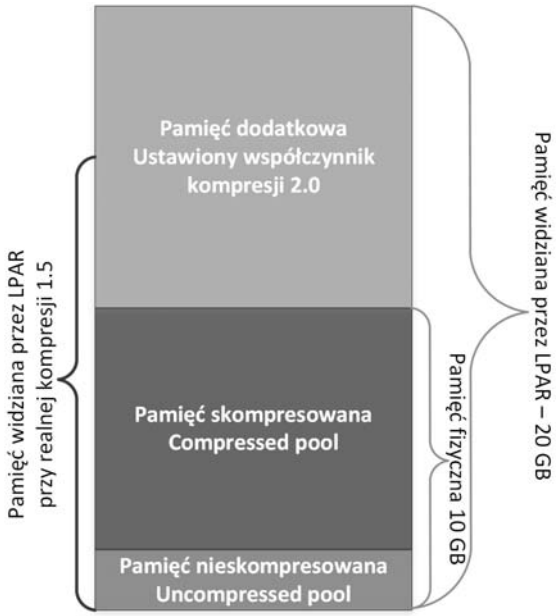
W przypadku gdy aplikacja odnosi się do pamięci będącej w stanie skompresowanym, następuje tzw. *Page Fault*, który polega na dekompresji tej pamięci, i przerzucenie jest do *Uncompressed Pool*. Dopiero wtedy aplikacja wykonuje pożądane operacje, nie będąc świadoma operacji dekompresji i przeniesienia pamięci przez system operacyjny.

Fakty dotyczące AME, na które należy zwrócić uwagę:

- AME nie kompresuje pamięci tzw. pinned (jest to pamięć oznaczona jako nieprzenośna i nigdy nie może zostać przeniesiona na *paging space* ani skompresowana przez AME).
- AME nie kompresuje pamięci cache systemu plików. Dobrą praktyką jest ograniczenie maksymalnej wielkości cache plików w przypadku użycia AME (parametr `maxclient%` polecenia `vmo`).

- Systemy obsługujące aplikacje pracujące w pamięci RAM na skompresowanych lub zaszyfrowanych danych nie skorzystają z kompresji pamięci.
- AME kompresuje wyłącznie strony o wielkości 4 KB.
- Wykorzystanie kompresji wymaga dodatkowych cykli procesora, chociaż przy odpowiedniej konfiguracji obciążenie jest znikome.
- W przypadku użycia zbyt dużego współczynnika kompresji może nastąpić tzw. deficyt pamięci. Oznacza to, że zadeklarowana ilość pamięci, którą widzi system operacyjny, nie może być osiągalna ze względu na zbyt niski współczynnik kompresji lub zbyt dużą ilość danych niepodlegających kompresji. W takim przypadku struktury, które nie mieszczą się w pamięci operacyjnej, są przerzucane na paging space, co grozi dużym spowolnieniem pracy. Przykład deficytu pamięci prezentuje rysunek 3.9.

Rysunek 3.9. *Kompresja pamięci — deficyt pamięci*



Pomimo wielu wymienionych powyżej ograniczeń większość systemów jest w stanie skorzystać z powodzeniem z tego mechanizmu przy jednoczesnej niewielkiej utracie mocy procesora. Łatwo to zweryfikować przy użyciu narzędzia `amepat`, które jest dostępne w systemie AIX niezależnie od posiadanych licencji.

Praca z Active Memory Expansion

Uruchomieniu AME towarzyszą potencjalne zagrożenia w postaci wpływu na wydajność oraz pytania dotyczące tego, jaki współczynnik kompresji będzie odpowiedni dla naszego systemu. Odpowiedzią na to jest narzędzie `amepat`. Jest ono dostępne z poziomu systemu operacyjnego niezależnie od tego, czy nasz serwer posiada licencję na tę funkcjonalność, czy nie. Narzędzie jest w stanie wykonać analizę i przedstawić kilka

propozycji konfiguracji systemu. Posiadając te propozycje i mając informację, jak wpłyną one na obciążenie systemu, można dokonać wyboru opcji najbardziej interesującej w naszym przypadku. Przykładowe wykorzystanie `amepat`:

```
# amepat 1 9    # Uruchomienie dziewięciu jednominutowych próbek.
...wycięte zbędne informacje
Active Memory Expansion Modeled Statistics         :
-------------------------------------------
Modeled Expanded Memory Size :   4.00 GB
Achievable Compression ratio   :2.42    # Średni współczynnik kompresji pamięci LPAR-a.

Expansion    Modeled True      Modeled             CPU Usage
Factor       Memory Size       Memory Gain         Estimate
---------    -------------     ------------------  -----------
     1.00          4.00 GB         0.00 KB [  0%]  0.00 [  0%]
     1.07          3.75 GB       256.00 MB [  7%]  0.00 [  0%]
     1.15          3.50 GB       512.00 MB [ 14%]  0.00 [  0%]
     1.24          3.25 GB       768.00 MB [ 23%]  0.00 [  0%]
     1.34          3.00 GB         1.00 GB [ 33%]  0.00 [  0%]
     1.46          2.75 GB         1.25 GB [ 45%]  0.00 [  0%]
     1.60          2.50 GB         1.50 GB [ 60%]  0.00 [  0%]

Active Memory Expansion Recommendation:
---------------------------------------
The recommended AME configuration for this workload is to configure the LPAR
with a memory size of 2.50 GB and to configure a memory expansion factor
of 1.60.  This will result in a memory gain of 60%. With this
configuration, the estimated CPU usage due to AME is approximately 0.00
physical processors, and the estimated overall peak CPU resource required for
the LPAR is 0.80 physical processors.
...wycięte zbędne informacje
```

Jak widać na powyższym wydruku, rezultat uruchomionej analizy daje kilka propozycji ustawień. Ostatnia z propozycji sugeruje zastosowanie współczynnika kompresji 1,6 i ograniczenie fizycznej pamięci z 4 GB do 2,5 GB. Przy takich ustawieniach dodatkowe użycie procesora do kompresji wciąż będzie wynosić 0%. Podczas przeprowadzonego testu LPAR używał niewielkiej ilości posiadanej pamięci i jej utylizacja nie przekroczyła 2,5 GB, więc kompresja, mimo że włączona, nie spowodowałaby utraty cykli procesora.

W tym przypadku analiza wykazała, że LPAR posiada więcej pamięci, niż jest mu potrzebne, zatem propozycje ograniczają się do zmniejszenia fizycznej pamięci i włączenia kompresji. W rezultacie uwolniona zostanie pamięć, którą będzie można spożytkować do innych celów, natomiast z punktu widzenia działających aplikacji nic się nie zmieni.

Kluczowym elementem jest wybór odpowiedniego czasu, w którym będzie przeprowadzana analiza. Należy ją przeprowadzać wtedy, gdy spodziewamy się większej utylizacji danego systemu. Dobrze jest uruchomić analizę na dłuższy okres, tak by była jak najbardziej miarodajna. Niewłaściwie dobrane parametry mogą się odbić na wydajności systemu.

Choć kompresja pamięci jest przezroczysta dla aplikacji, to mając dostęp wyłącznie do systemu operacyjnego, jesteśmy w stanie odczytać jej ustawienia. Poleceniem dającym nam odpowiednie informacje jest `lparstat`:

```
# lparstat -i    # Zbędne informacje zostały wycięte z tego wydruku.
Online Memory                              : 6144 MB
Memory Mode                                : Dedicated-Expanded
Target Memory Expansion Factor             : 1.50
Target Memory Expansion Size               : 9216 MB
```

Polecenie pokazuje, że system fizycznie posiada 6144 MB pamięci z włączoną kompresją o współczynniku 1,5, co w sumie daje systemowi do dyspozycji 9216 MB.

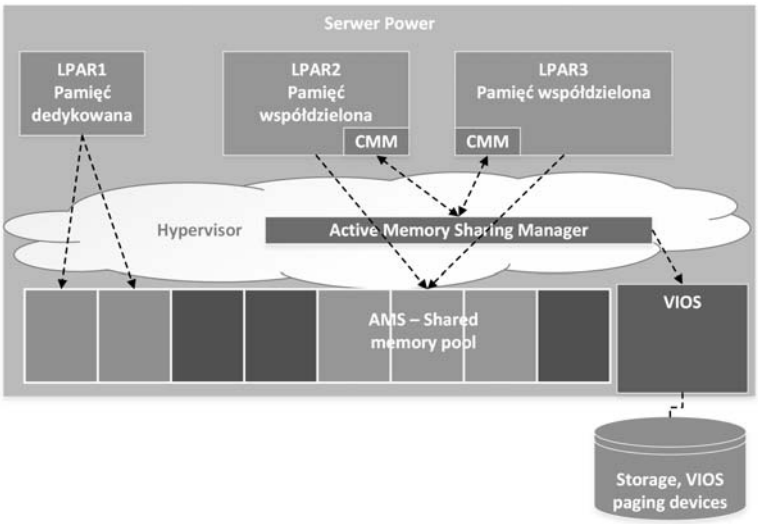
Do bieżącego monitoringu AME można użyć narzędzi takich jak `vmstat -c`, `lparstat -c` czy `topas`.

Active Memory Sharing (AMS)

W poprzedniej sekcji opisywany był mechanizm *Active Memory Expansion*. Mechanizm ten działa na poziomie LPAR-a i pozwala na wykorzystanie większej ilości pamięci, niż posiada LPAR.

Opisywany w tej sekcji *Active Memory Sharing* pozwala na uzyskanie podobnego efektu, czyli wykorzystanie większej ilości pamięci przez LPAR-y, niż fizycznie posiada serwer. Podstawowa różnica jest taka, że AMS działa na poziomie całego serwera, podczas gdy AME działa na poziomie LPAR-a.

Schemat działania AMS przedstawia rysunek 3.10.



Rysunek 3.10. *AMS — schemat działania*

Mechanizm AMS jest bardziej skomplikowany w działaniu niż AME. Wymaga dużo większych przygotowań do uruchomienia. Do działania *Active Memory Sharing* konieczne jest wydzielenie części pamięci na specjalną strukturę *AMS shared memory*

pool oraz posiadanie Virtual I/O Servera z wydzielonymi specjalnymi logicznymi woluminami lub dyskami na tzw. *paging devices*. Struktury przedstawione na rysunku:

- ***AMS Shared memory pool*** — wydzielona pula pamięci współdzielona przez grupę partycji. Na jednym serwerze może istnieć tylko jedna taka pula. LPAR-y albo w pełni korzystają z pamięci współdzielonej, albo w pełni korzystają z pamięci dedykowanej. Nie ma możliwości mieszania różnych typów pamięci w jednej partycji.

  Z punktu widzenia LPAR-a działającego w AMS przydział pamięci z puli i fizyczne adresy pamięci mu przydzielonej mogą się stale zmieniać. Oznacza to, że pamięć fizyczna przypisana w danej chwili do jednego LPAR-a w innym momencie może być przypisana do innego LPAR-a po uprzednim wyczyszczeniu jej zawartości przez hypervisor. Dzięki temu mechanizmowi partycja nie posiada nieużywanej pamięci, gdyż taka pamięć zwracana jest do puli. W przypadku nowego zapotrzebowania jest ona alokowana dla danego LPAR-a. Operacje tego typu działają bardzo efektywnie pod względem używanej pamięci, gdyż operują blokiem 4 KB, czyli blokiem typowej wielkości dla systemu operacyjnego.

- ***VIOS paging devices*** — zadeklarowana pamięć, której używają LPAR-y działające w *AMS shared memory pool*, może przekraczać całkowitą wielkość tej puli (*over-commitment*). W przypadku zastosowania tego rodzaju puli jest to efekt pożądany, gdyż chodzi właśnie o efektywne wykorzystanie pamięci. Niemniej jednak może zaistnieć sytuacja, w której wiele LPAR-ów równocześnie wykaże zapotrzebowanie na pamięć, które w rezultacie okaże się większe niż posiadana ilość pamięci w puli. Odpowiedzią na taki problem są specjalne urządzenia typu paging działające na wyznaczonym VIOS-ie (jednym lub dwóch dla zachowania wysokiej dostępności) na rzecz puli. Niestety opisywany przypadek prowadzi do wymiany stron pamięci z przygotowanym paging space'em, co jest dobrym mechanizmem obronnym, ale może prowadzić do znacznego spowolnienia pracy LPAR-ów dotkniętych tym procesem. Zaznaczyć przy tym trzeba, że dla każdego LPAR-a przypisywane jest jedno dedykowane urządzenie typu paging space. Zatem dla działania mechanizmu konieczne jest posiadanie tylu urządzeń, ile LPAR-ów ma pracować w puli pamięci.

  Jak było wspomniane wcześniej, mechanizm ten jest dosyć złożony. Wymaga współpracy ze sobą kilku elementów, takich jak:

  - ***Active Memory Sharing Manager (AMSM)*** — kod znajdujący się w hypervisorze będący sercem mechanizmu. Zarządza on pulą pamięci, przydziela i odbiera pamięć od LPAR-ów i współpracuje z VIOS-em, zlecając mu zapisanie lub odczyt stron pamięci z paging device. Odbieranie stron pamięci od LPAR-ów odbywa się na dwa sposoby.

    - **Poprzez *loaning*** — w tym przypadku system operacyjny sam wyznacza strony pamięci, które nie są mu potrzebne. Oddaje te strony w sposób dobrowolny — pożyczając je. Strony pozyskiwane w ten sposób nie powinny wpływać negatywnie na wydajność systemu.

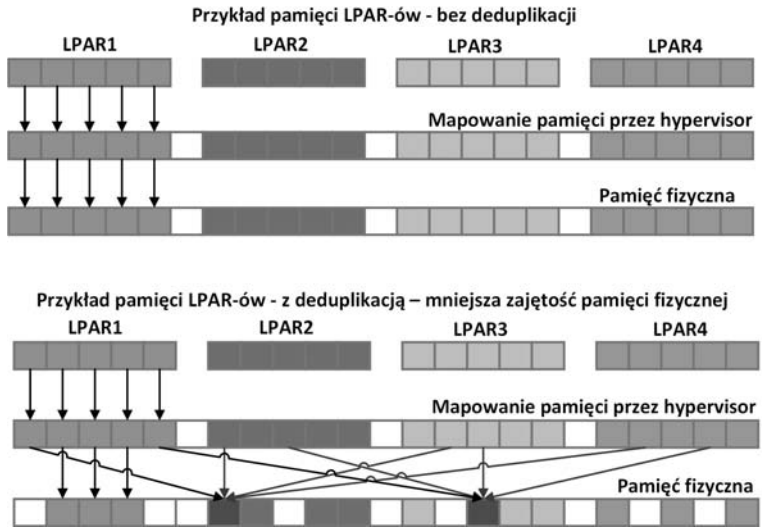
- **Poprzez *stealing*** — w tym przypadku AMSM odbiera strony w sposób „siłowy” („kradnąc” je), uprzednio zapisując ich zawartość za pośrednictwem VIOS-a na odpowiednim paging device. Odbierając strony od LPAR-a, zawsze odbiera te najmniej używane. Statystyki odnośnie do użycia stron pamięci udostępnia dla hypervisora jądro systemu operacyjnego będącego obiektem działań.

- ***Collaboration Memory Manager (CMM)*** — funkcjonalność jądra systemu operacyjnego. Jego rolą jest współpraca z AMSM. Przekazuje informacje o krytyczności poszczególnych stron pamięci, wyznacza strony do pożyczenia i dostarcza statystyki o wykorzystaniu stron, które są używane do „kradzieży” odpowiednich stron przez hypervisor.

Współdzielenie pamięci nie byłoby kompletne bez mechanizmu deduplikacji. Tutaj również jest do dyspozycji ten mechanizm (*Active Memory Deduplication* — AMD). AMD może być włączone lub wyłączone dla wszystkich LPAR-ów korzystających z AMS. Nie ma możliwości włączenia mechanizmu dla pojedynczych LPAR-ów.

Deduplikacja w sposób znaczący pomaga zredukować wielkość potrzebnej pamięci i tym samym zapewnić duży *overcommitment* pamięci. Można sobie wyobrazić prosty przykład, w którym uruchamiamy 4 LPAR-y z tą samą wersją systemu operacyjnego i podobną aplikacją działającą na tym samym oprogramowaniu. Po uruchomieniu pamięć każdego z nich ma wiele punktów wspólnych, gdyż załadowane są te same kody aplikacyjne. Zatem mechanizm deduplikacji nie musi przechowywać cztery razy tej samej informacji dla każdego LPAR-a osobno, natomiast może przechować tę informację raz, „mapując” ją pod odpowiedni adres pamięci każdego z LPAR-ów. Taką sytuację prezentuje rysunek 3.11.

Rysunek 3.11. *AMS — deduplikacja*



Mechanizm deduplikacji nie działa w sposób natychmiastowy. Nie można się spodziewać, że w momencie równoczesnego uruchomienia kilku identycznych LPAR-ów globalna utylizacja pamięci od razu będzie mniejsza. Mechanizm ten działa w tle.

Hypervisor buduje mapę pamięci, obliczając sygnatury każdej jej strony. Posiadając sygnatury, może tę mapę przeglądać w miarę wolnych zasobów i łączyć strony, przeprowadzając proces deduplikacji. Jeżeli któryś z LPAR-ów korzystających z tej samej strony modyfikuje ją, to następuje proces COW (*Copy On Write*). Strona pamięci zostaje skopiowana w nowe miejsce i podstawiona do LPAR-a ją modyfikującego, stając się jego własnością.

Jak każdy tego typu mechanizm AMS ma kilka ograniczeń, których trzeba być świadomym:

- LPAR-y korzystające z tej funkcjonalności muszą korzystać z procesorów w trybie *shared* (tryb *Dedicated* nie jest dopuszczalny).
- LPAR-y mogą być wyposażone wyłącznie w wirtualne urządzenia (virtual SCSI, virtual Fibre Channel — NPIV, virtual serial, virtual Ethernet). Urządzenia fizyczne są niedopuszczalne.
- System operacyjny nie może używać bloków innych niż 4 KB, co może mieć pewien negatywny efekt wydajnościowy w przypadku aplikacji, które masowo korzystają ze stron 64 KB. W przypadku AMS aplikacje będą musiały korzystać ze strony 4 KB, zatem ich obsługa może wymagać więcej cykli procesora niż w przypadku 64 KB.

Szczegółowe informacje na temat *Active Memory Sharing* dostępne są w publikacji *IBM PowerVM Virtualization Active Memory Sharing* dostępnej na stronach *http://www.redbooks.ibm.com*.

Virtual I/O Server (VIOS) — wirtualizacja I/O

Virtual I/O Server jest częścią składową wirtualizacji platformy Power. Jest on odpowiedzialny za wirtualizację I/O. Podział wirtualizacji na osobne części składowe wirtualizujące procesor i pamięć oraz osobne wirtualizujące operacje I/O może dziwić, szczególnie osoby zajmujące się wirtualizacją na platformie x86. Niemniej jednak ma to głęboki sens w postaci:

- Stabilnego, wewnętrznego, mało zmiennego świata wirtualizacji CPU i RAM, który może obsłużyć prosty kod hypervisora zintegrowany z maszyną.
- Skomplikowanego, zmiennego świata operacji I/O, który może obsłużyć specjalnie do tego celu skonstruowany twór — Virtual I/O Server.

VIOS okiem administratora AIX-a jest właśnie systemem AIX po dokonaniu dużej customizacji pod kątem współpracy z podstawowym hypervisorem serwera i optymalnej obsługi operacji I/O na rzecz innych LPAR-ów. Użytkując Virtual I/O Server, zazwyczaj działamy w restricted shellu z mocno ograniczonymi uprawnieniami i z dostępem do specjalnie przygotowanych poleceń. Polecenia te są w większości aliasami poleceń systemu AIX. Wykonując polecenie `oem_setup_env`, stajemy się rootem systemu VIOS i mamy dostęp do tradycyjnych poleceń systemu AIX.

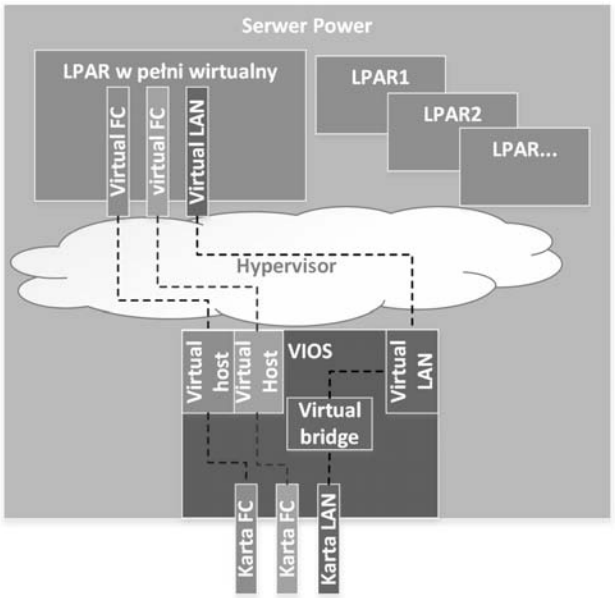
Virtual I/O Server jest tworem, któremu przypisujemy fizyczne urządzenia: dyski, karty LAN i karty SAN, po to, aby je wirtualizował i udostępniał dla LPAR-ów znajdujących się na serwerze. VIOS-ów na pojedynczym serwerze może być wiele. Istnieje możliwość wydzielenia konkretnych VIOS-ów dla konkretnych systemów, czy też wydzielenia konkretnych VIOS-ów do obsługi ruchu dyskowego, a innych do obsługi ruchu sieciowego. Zazwyczaj stosuje się jednak dwie typowe konfiguracje z jednym i z dwoma VIOS-ami. Te typowe konfiguracje i jedna mniej typowa z wieloma VIOS-ami zostaną pokazane na poniższych rysunkach.

- **Serwer z jednym VIOS-em** — typowa konfiguracja dla małych instalacji i małych serwerów. Pojedynczy VIOS nie ogranicza funkcji wirtualizacji. Na jednym VIOS-ie można wykorzystać wiele ścieżek dostępu do dysków i do sieci LAN, co ratuje nas w przypadku awarii pojedynczych adapterów. Niemniej jednak taka konfiguracja nie pozwala w pełni wykorzystać potencjału platformy w postaci maksymalizacji jej dostępności.

  W przypadku posiadania jednego VIOS-a jego awaria powoduje niedostępność wszystkich LPAR-ów z niego korzystających. Dodatkowo przed każdym jego restartem (np. z powodu patchowania) LPAR-y z niego korzystające powinny być wyłączone lub wyniesione z serwera, na przykład za pomocą mechanizmów *Live Partition Mobility*.

  Schemat działania wirtualizacji na serwerze z jednym VIOS-em przedstawia rysunek 3.12.

Rysunek 3.12. *Serwer z jednym VIOS-em*

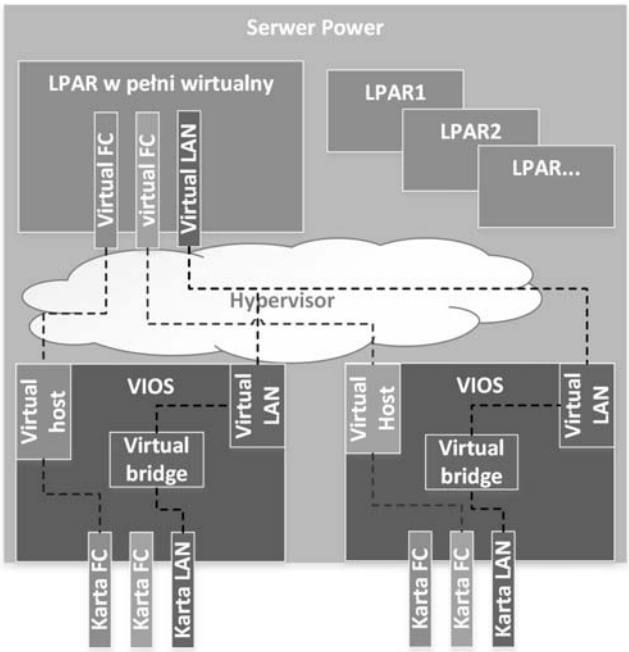


- **Serwer z dwoma VIOS-ami** — typowa instalacja pozwalająca zachować pełną redundancję. Oba VIOS-y jednocześnie pracują na rzecz LPAR-ów. Awaria jednego z nich lub planowany restart nie powodują niedostępności partycji na serwerze, gdyż z punktu widzenia dostępu do dysków i dostępu do sieci zachowane zostają ścieżki poprzez drugi VIOS.

Posiadanie dwóch VIOS-ów podnosi dostępność systemów i jednocześnie umożliwia stałe i systematyczne ich patchowanie w trybie naprzemiennym.

Schemat działania wirtualizacji na serwerze z jednym VIOS-em przedstawia rysunek 3.13.

Rysunek 3.13. *Serwer z dwoma VIOS-ami*

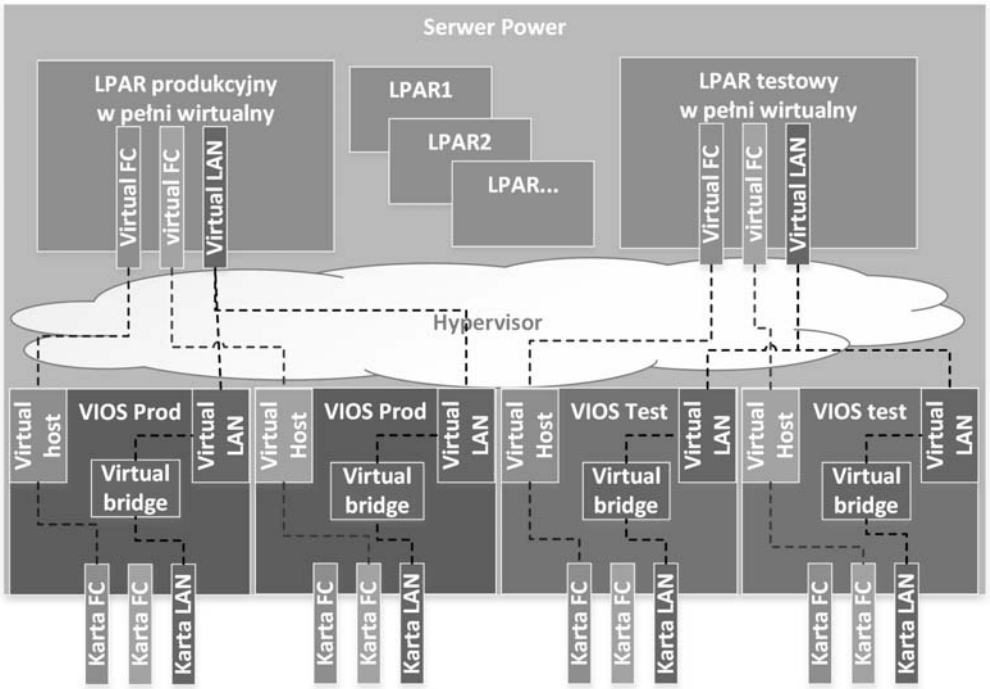


- **Serwer z czterema VIOS-ami** — rysunek 3.14 pokazuje przykład posiadania na jednym serwerze zarówno systemów produkcyjnych, jak i testowych przy jednoczesnym odseparowaniu VIOS-ów je obsługujących. W szerszym kontekście pokazuje on szerokie możliwości wykorzystania VIOS-ów i potencjalnie wiele scenariuszy, które można zastosować.

Planowanie instalacji VIOS-a

Sam proces instalacji Virtual I/O Servera jest dużo prostszy niż instalacja systemu operacyjnego AIX. Jako że VIOS jest systemem mocno wyspecjalizowanym, nie mamy zbyt dużo opcji podczas procesu instalacji. Niezwykle ważną kwestią jest odpowiednie zaplanowanie konfiguracji i określenie, na jakich komponentach fizycznych ma się odbyć instalacja.

VIOS może zostać zainstalowany zarówno na dyskach wewnętrznych podłączonych magistralą SAS (*Serial Attached SCSI*), jak i dyskach z zewnętrznej macierzy dyskowej. Zazwyczaj do instalacji wybiera się dyski wewnętrzne. Jest kilka powodów wyboru dysków wewnętrznych do instalacji VIOS-a:



Rysunek 3.14. *Serwer z czterema VIOS-ami*

- **Izolacja problemów z macierzami dyskowymi i siecią SAN/LAN** — wyobraźmy sobie sytuację, w której macierze dyskowe odmawiają posłuszeństwa i tracimy dostęp zarówno do systemów, jak i do VIOS-a. Takie zdarzenie mocno utrudnia diagnostykę i wydłuża czas przywrócenia funkcjonalności systemów.
- **Odtworzenie systemu VIOS po awarii** — wyobraźmy sobie sytuację, w której np. przez błąd administratora VIOS ulega awarii. Wówczas konieczne staje się odtworzenie z backupu. Problem uwidacznia się w momencie wyboru dysku, na którym należy odtworzyć system. W przypadku dysku wewnętrznego sprawa jest oczywista, gdyż z dużym prawdopodobieństwem jest jedynym dyskiem wewnętrznym bootowalnym. W przypadku zastosowania dysku z macierzy do odtworzenia można omyłkowo wybrać inne dyski, które wystawione były poprzez VIOS do LPAR-ów. Wybór odpowiedniego dysku dodatkowo utrudnia interfejs odtwarzania, dający ograniczone informacje o dyskach, na których możemy odtworzyć system.

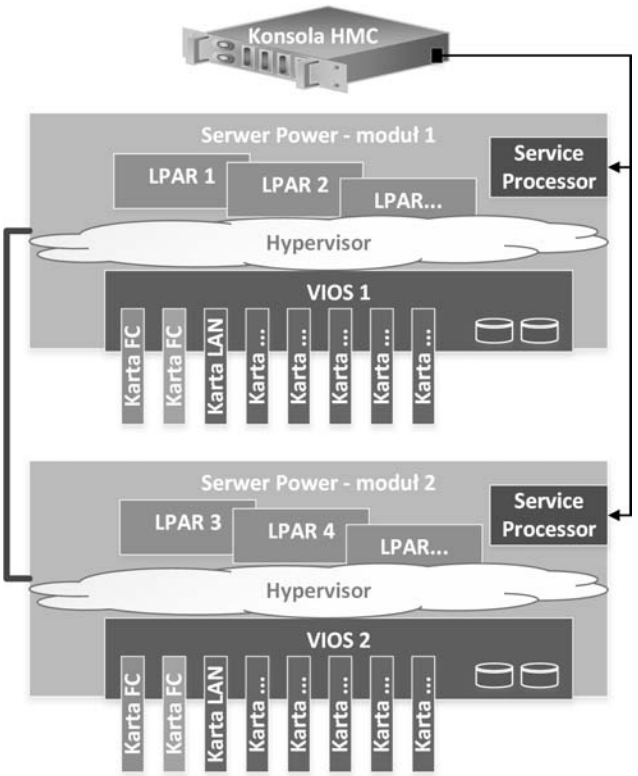
W przypadku wyboru dysków wewnętrznych przy instalacjach z dwoma VIOS-ami mającymi pełnić dla siebie rolę HA (High Availbility — wysoka dostępność) należy zadbać o to, aby dyski wewnętrzne nie miały części wspólnych. Czyli nie były obsługiwane przez ten sam kontroler dyskowy, a w przypadku dużych serwerów nie były umiejscowione w tym samym module.

Ważnym aspektem, szczególnie w przypadku instalacji dwóch i wielu VIOS-ów, jest wybór komponentów (kart fizycznych), które będą wirtualizowane. Czym należy się sugerować w przypadku serwerów modułowych, pokazuje poniższy przykład.

Załóżmy, że mamy do dyspozycji serwer z rodziny IBM Power System 870. Jest to serwer modułowy. W naszym przykładzie zakładamy dysponowanie serwerem składającym się z dwóch modułów, bez zewnętrznych komponentów I/O. W tym przypadku VIOS-y mają być dla siebie zastępstwem w razie awarii. Oto przykład, w jaki sposób możemy zaplanować instalację i przydział komponentów dla zapewnienia najwyższej dostępności serwera.

Rysunek 3.15 pokazuje kilka reguł, którymi warto się kierować. Poniżej znajduje się opis tych reguł z wyjaśnieniem powodów ich zastosowania.

Rysunek 3.15. *VIOS-y na serwerach modułowych*



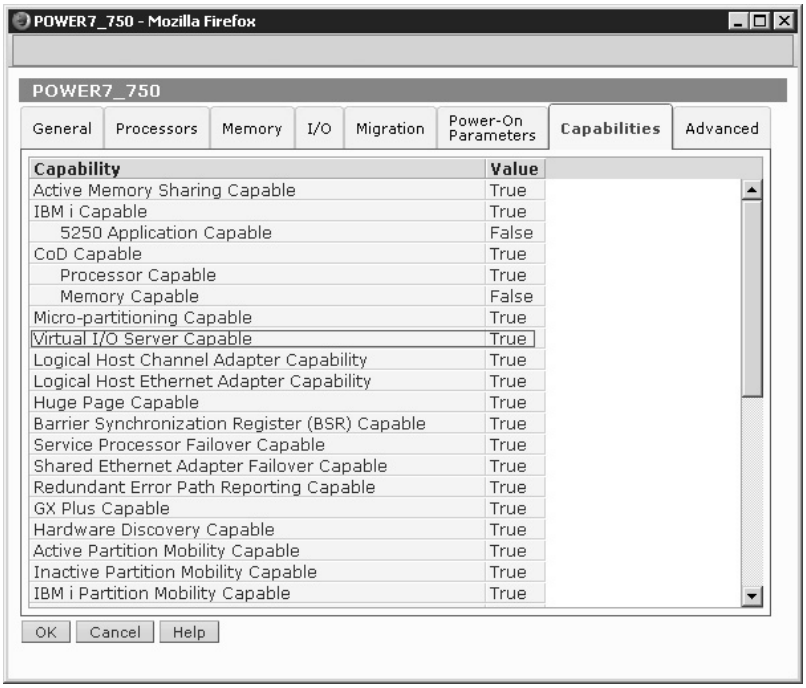
- **Dyski systemowe w różnych modułach** — każdy Virtual I/O Server jest fizycznie instalowany w innym module serwera. Jego dyski są między sobą mirrorowane. W miarę możliwości można pokusić się o to, aby każdy z dysków znajdował się na osobnym kontrolerze SAS.

- **Interfejsy zewnętrzne w tym samym module co dyski systemowe** — wszystkie interfejsy zewnętrzne, którymi zarządza VIOS, znajdują się w tym samym module co dyski systemowe do niego przypisane. Wskazane jest, aby Virtual I/O Server dysponował dedykowanym, szybkim portem sieciowym w przypadku wykorzystywania funkcjonalności *Live Partition Mobility*. Dzięki temu procesy migracji będą mogły przebiegać szybko oraz bez wpływu na ruch sieciowy innych systemów. Przed instalacją należy zapoznać się z dokumentacją konfigurowanego serwera i rozmieścić karty w odpowiednich slotach w taki sposób, aby nie ograniczać ich wydajności.
- **Przypisanie wszystkich komponentów VIOS-a do pojedynczego modułu** — na pytanie, dlaczego jest to tak istotne, odpowie kilka poniższych przykładów:
  - **Nastąpiła „duża" awaria jednego z modułów** — w tym przypadku „duża" oznacza niedostępność całego modułu. Jako że serwer składał się z dwóch modułów, jeden z nich może dalej pracować. Jeżeli były dwa dobrze rozmieszczone VIOS-y, to jest duża szansa, że jeden z nich pracuje dalej (podczas tworzenia VIOS-ów hypervisor powinien zadbać o rozmieszczenie ich jak najbliżej posiadanych komponentów fizycznych). Prawdopodobnie również mniej więcej połowa LPAR-ów mogła przeżyć tę awarię. Nawet jeżeli awaria spowodowała wyłączenie wszystkich systemów, to w dalszym ciągu jeden z Virtual I/O Serverów może zostać uruchomiony i w kolejnych krokach mogą zostać uruchomione LPAR-y z dokładnością do posiadanych, zmniejszonych o połowę zasobów.
  - **Nastąpiła „mała" awaria w jednym z modułów** — w tym przypadku „mała" oznacza przezroczystą dla działania LPAR-ów awarię np. procesora lub pamięci (platforma Power posiada szereg rozwiązań RAS — *Reliability Availbility Serviceability* — które pozwalają na wczesne wykrycie awarii czy też powtórzenie instrukcji na innym procesorze w przypadku awarii jednego z nich). Mała awaria, chociaż nie wpłynęła na dostępność systemów, może wymagać akcji serwisowej połączonej z wyłączeniem modułu. W przypadku odpowiedniej konfiguracji można pokusić się o wykonanie takiej operacji na gorąco, wyłączając jeden z VIOS-ów i część LPAR-ów, które nie zmieszczą się na jednym sprawnym module.

Instalacja VIOS-a

Przed przystąpieniem do instalacji Virtual I/O Servera należy się upewnić, że posiadamy odpowiednie licencje. Wiele funkcjonalności serwerów Power kupowanych i aktywowanych jest oddzielnie, włączając w to wirtualizację. Sprawdzenia, jakie licencje posiada serwer, można dokonać na stronie *http://www-912.ibm.com/pod/pod* po wpisaniu danych serwera. Uzyskane w ten sposób kody można aktywować na serwerze (o ile nie były aktywowane wcześniej). To, czy dane funkcjonalności serwera są na nim aktywne, można zweryfikować z poziomu właściwości serwera na konsoli HMC. Ekran z właściwościami serwera pokazuje rysunek 3.16.

Rysunek 3.16. *HMC — Capabilities, VIOS*



Jak widać powyżej, funkcjonalność Virtual I/O Servera jest aktywowana. Kolejne kroki instalacji można opisać w kilku punktach:

- **Tworzenie LPAR-a** — na początek należy utworzyć LPAR, wskazując, że będzie on pełnił rolę VIOS-a. Czynność ta jest analogiczna jak w przypadku tworzenia LPAR-a klienckiego z systemem AIX. Różnica jest taka, że VIOS musi posiadać fizyczne komponenty, na których zostanie zainstalowany. Przy tworzeniu partycji należy zwrócić szczególną uwagę na przydział odpowiednich urządzeń fizycznych, zgodnych z opracowanym wcześniej planem. Często popełnianym niedopatrzeniem jest ustawienie zbyt niskiej wartości przy opcji *Maximum virtual adapters*. Domyślnie wartość ta wynosi 10. Pozostawienie jej bez zmiany skutkuje szybkim osiągnięciem wartości maksymalnej i w rezultacie powoduje konieczność restartu VIOS-a i zwiększenia tej wartości.
- **Instalacja VIOS-a** — instalacji VIOS-a można dokonać lokalnie, uruchamiając LPAR z płyty DVD, za pośrednictwem konsoli HMC (wykorzystując mechanizm NIMOL — NIM on Linux) lub z poziomu posiadanej instalacji Network Installation Managera — poprzez sieć LAN. Network Installation Manager został szerzej opisany w rozdziale dotyczącym instalacji systemu AIX. Sam proces instalacji nie daje zbyt wielu opcji do wyboru. VIOS jest tworem spełniającym ściśle zdefiniowane funkcje, zatem zawsze jest to instalacja powtarzalna z niewielką ilością opcji customizacji.
- **Patchowanie** — w przypadku instalacji każdego oprogramowania konieczna jest instalacja najnowszych poprawek i stałe kontrolowanie, czy nie pojawiają się ich nowsze wersje. Do poprawek wydawane są instrukcje instalacji i zawsze należy postępować zgodnie z nimi.

- **Konfiguracja** — po instalacji i patchowaniu następuje konfiguracja. Podlega jej wiele aspektów systemu, takich jak konfiguracja użytkowników, monitoringu, instalacja odpowiednich sterowników do macierzy, z którymi VIOS będzie współpracował, hardening, konfiguracja współdzielonej sieci itp. Temat konfiguracji jest na tyle rozległy, że warto przekierować w tym momencie do dokumentacji dostawcy, chociaż część zagadnień będzie również omówiona w dalszej części rozdziału. Warto przytoczyć kilka wskazówek wstępnych:
  - Użytkownikiem nadrzędnym, posiadającym pełne uprawnienia, jest użytkownik `padmin`. Pracuje on w restricted shellu. W związku z tym ma pełne uprawnienia konfiguracyjne, ale tylko z perspektywy przygotowanych dla niego poleceń. Polecenia dostępne dla użytkownika można podejrzeć za pomocą polecenia `help`. Są one widoczne również w pliku *.profile* jako aliasy:

```
# head -15 /home/padmin/.profile
export PATH=/usr/ios/cli:/usr/ios/utils:/usr/ios/lpm/bin:/usr/ios/oem:/usr/ios
/ldw/bin:$HOME

if [ $LOGNAME = "padmin" ]
  then ioscli license -swma
fi

export SHELL=/usr/bin/ksh
alias backup="ioscli backup"
alias restore="ioscli restore"
alias chbdsp="ioscli chbdsp"
alias mvbdsp="ioscli mvbdsp"
alias rmsp="ioscli rmsp"
alias mkrep="ioscli mkrep"
alias chrep="ioscli chrep"
```

  - VIOS wywodzi się z systemu AIX. Przy użyciu polecenia `oem_setup_env` można uzyskać autoryzację użytkownika `root` systemu operacyjnego, w którym można działać jak w klasycznym systemie AIX. Jest to często niezbędne do instalacji sterowników do obsługi macierzy dyskowych czy do instalacji klientów monitoringu.
  - Należy działać zawsze z konta dedykowanego użytkownika za pomocą poleceń VIOS-a. Konto użytkownika `root` należy wykorzystywać tylko, gdy jest to niezbędne.

Skorowidz

E

F

G

H

I

J

O

P

Q

R

S

Ś

T

U

V

W

Z

Poznaj system AIX z bliska!

Systemy operacyjne z rodziny UNIX znane są z wysokiej niezawodności i wydajności. Właśnie z tego powodu w wielu firmach są one wykorzystywane do zarządzania serwerami kluczowych aplikacji. Jednym z systemów należących do tej grupy jest AIX, który zyskał popularność dzięki bardzo dużym możliwościom wirtualizacji i konfiguracji zabezpieczeń spełniających nawet najsurowsze wymogi bezpieczeństwa.

Z tej książki dowiesz się, jak działa ten system operacyjny i jak z nim pracować jako administrator. Nauczysz się przy tym wykorzystywać najlepsze praktyki w branży. Poznasz sposób działania PowerVM, które jest jednym z najbardziej elastycznych, a jednocześnie najbardziej niezawodnych rozwiązań wirtualizacyjnych. Dowiesz się też, jak w praktyce wykorzystać liczne możliwości zapewnienia bezpieczeństwa systemu operacyjnego i aplikacji działających pod jego kontrolą.

- **Platforma IBM Power**
- **Podstawy systemu AIX**
- **Wirtualizacja elementów systemu**
- **Instalacja i utrzymanie systemu**
- **Zarządzanie użytkownikami, dyskami i systemem plików**
- **Tworzenie kopii bezpieczeństwa i diagnostyka systemu**
- **Zarządzanie siecią, bezpieczeństwem i wydajnością**

Zostań administratorem serwerów Power!

ISBN 978-83-283-3672-8

cena: 89,00 zł