



W5

**PODSTAWY BUDOWY I KORZYSTANIA
ZE ŚRODOWISKA DOCKER**

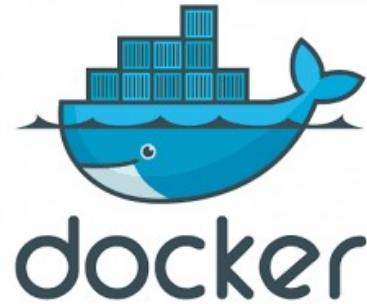
W5 Środowisko Docker

Definicja i podstawowe komponenty

Docker to platforma służąca tworzeniu, dostarczaniu oraz uruchamianiu aplikacji w zvirtualizowanym środowisku kontenerów.

Komponenty Docker:

- Docker Engine
- Docker Hub
- Docker Swarm
- Docker Compose
- niezależne narzędzia (otwarte API)



<https://www.docker.com/>

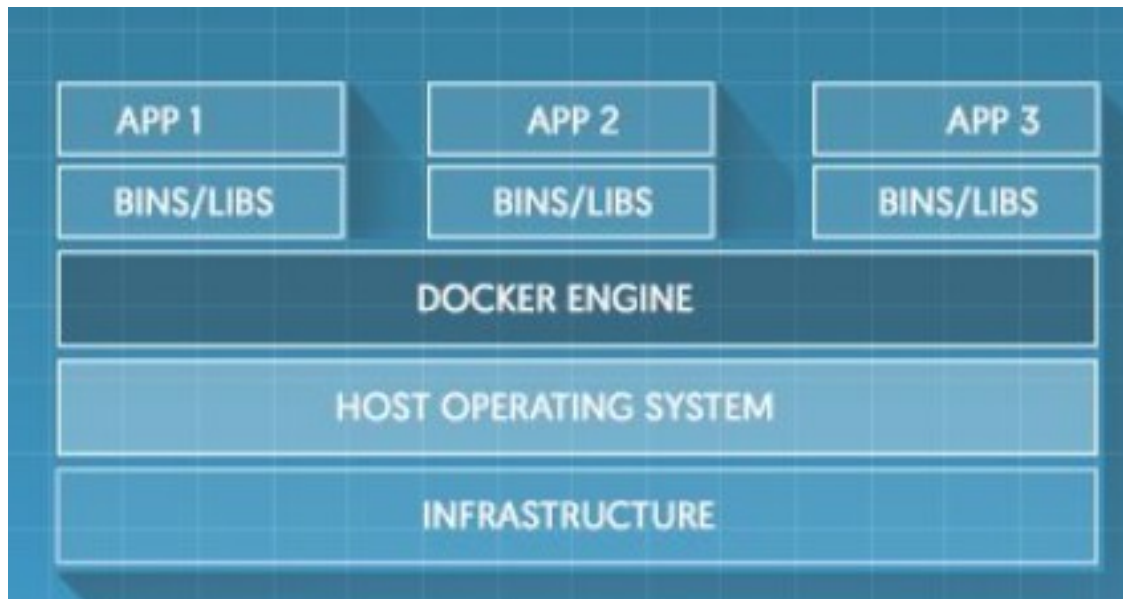
W5 Środowisko Docker

Pojęcie kontenera

Wirtualizacja oparta o kontenery używa kernela systemu operacyjnego hosta do uruchamiania wielu instancji aplikacji gości.

Każdy kontener posiada:

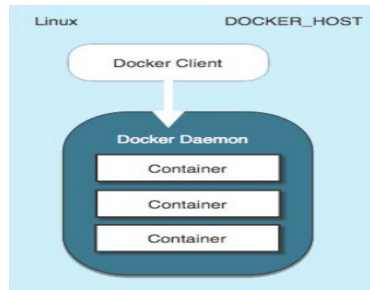
- system plików (rootfs)
- procesy
- pamięć
- urządzenia
- porty sieciowe



W5 Środowisko Docker

Architektura środowiska

- Klient (komenda docker) wysyła polecenia do demona
- Demon wykonuje pracę
- Klient i demon mogą znajdować się
 - na tym samym serwerze
 - na różnych serwerach
- Oprócz oficjalnego klienta - 3rd party
- automatyzacja
- orkiestracja
- GUI

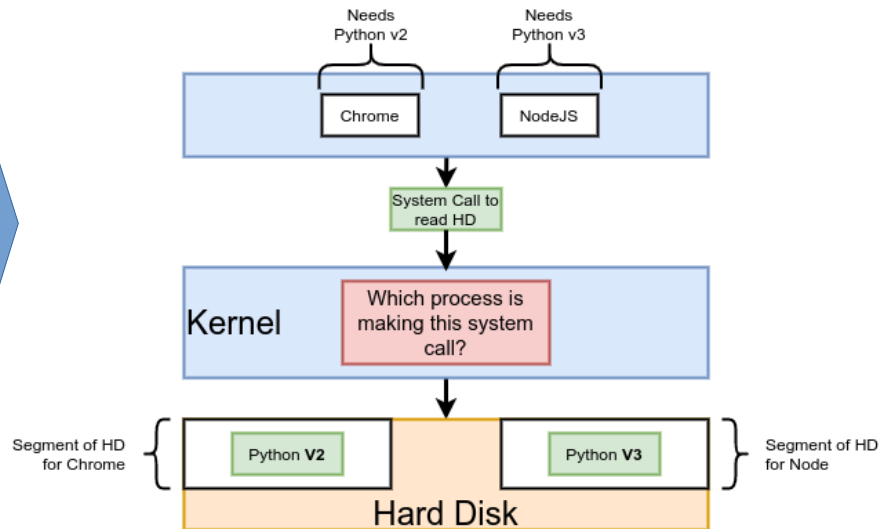
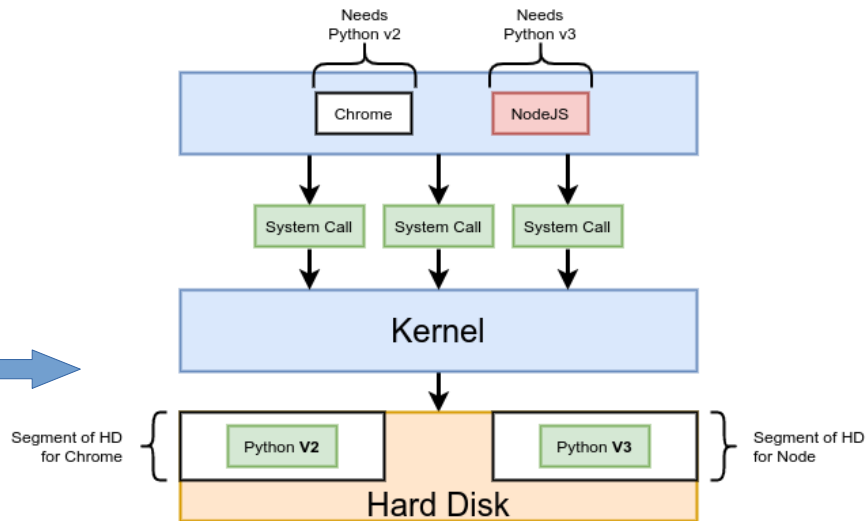
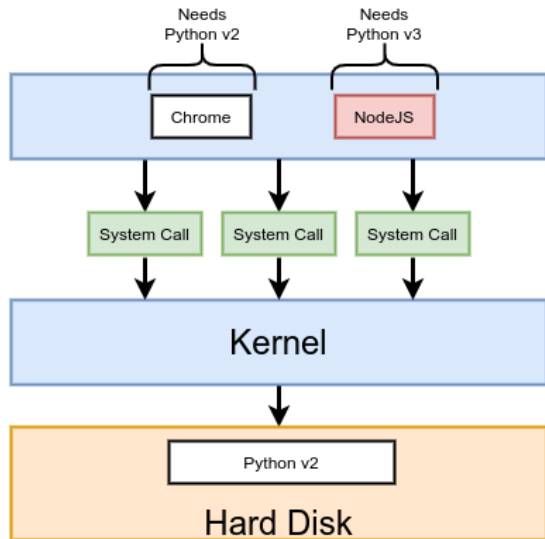


```
student@PwCh0-VB:~$ docker version
Client: Docker Engine - Community
 Version:      19.03.13
 API version:  1.40
 Go version:   go1.13.15
 Git commit:   4484c46d9d
 Built:        Wed Sep 16 17:02:52 2020
 OS/Arch:     linux/amd64
 Experimental: false

Server: Docker Engine - Community
 Engine:
  Version:      19.03.13
  API version:  1.40 (minimum version 1.12)
  Go version:   go1.13.15
  Git commit:   4484c46d9d
  Built:        Wed Sep 16 17:01:20 2020
  OS/Arch:     linux/amd64
  Experimental: false
 containerd:
  Version:      1.3.7
  GitCommit:    8fba4e9a7d01810a393d5d25a3621dc101981175
 runc:
  Version:      1.0.0-rc10
  GitCommit:    dc9208a3303feef5b3839f4323d9beb36df0a9dd
 docker-init:
  Version:      0.18.0
  GitCommit:    fec3683
```

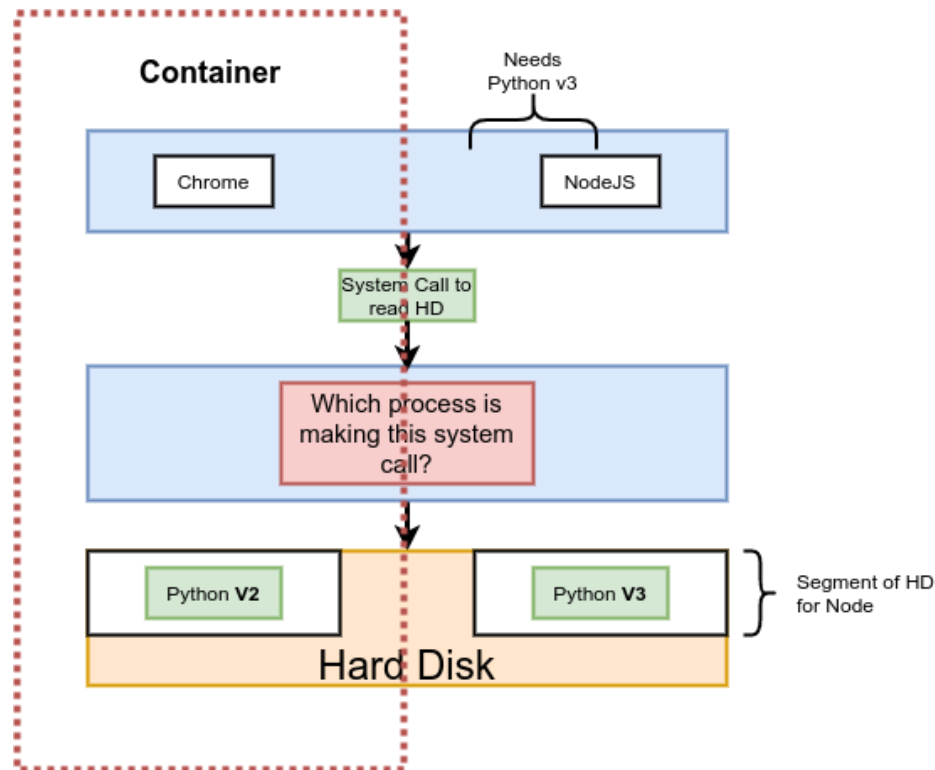
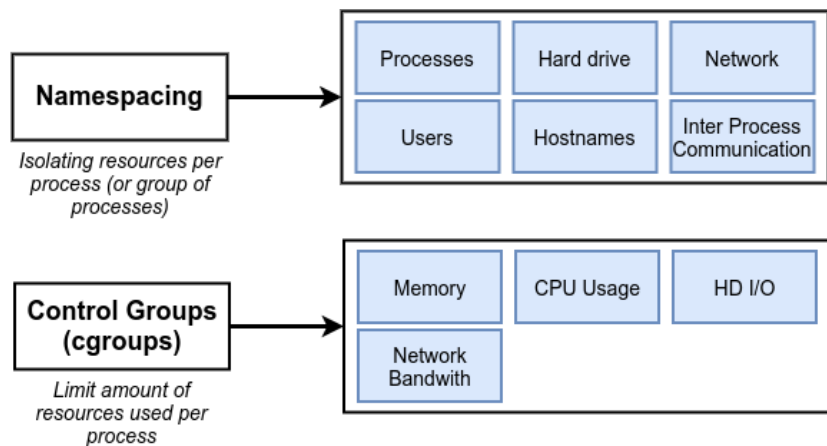
W5 Środowisko Docker

Co oferują kontenery Docker ? (1)



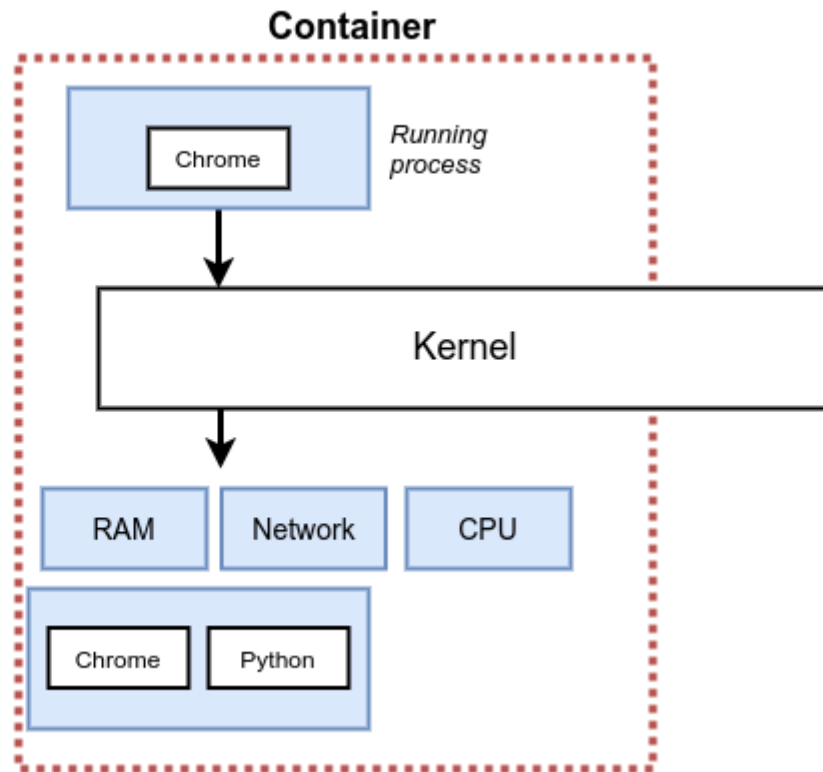
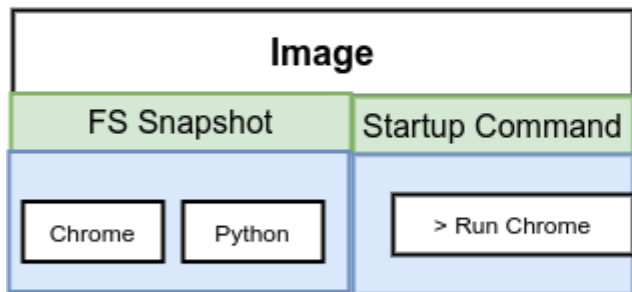
W5 Środowisko Docker

Co oferują kontenery Docker ? (2)



W5 Środowisko Docker

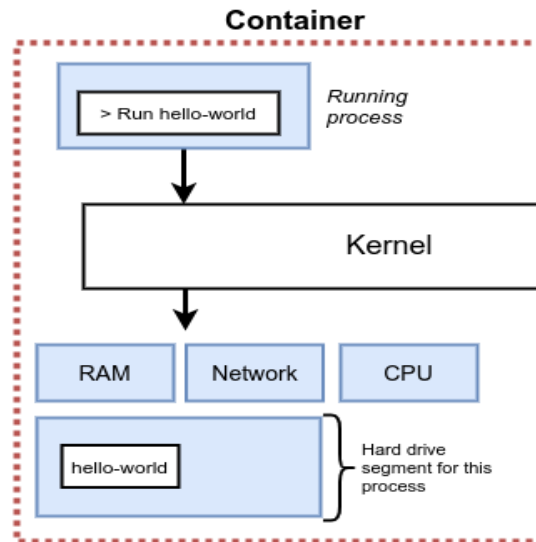
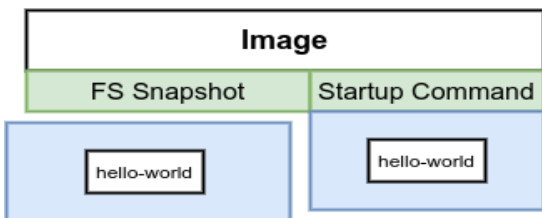
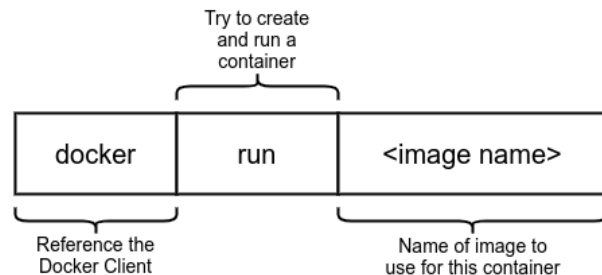
Kontener a obraz Docker



W5 Środowisko Docker

Wybrane polecenia w środowisku Docker (1)

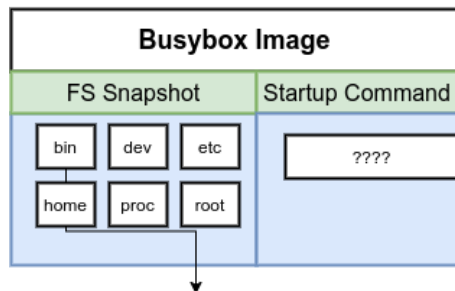
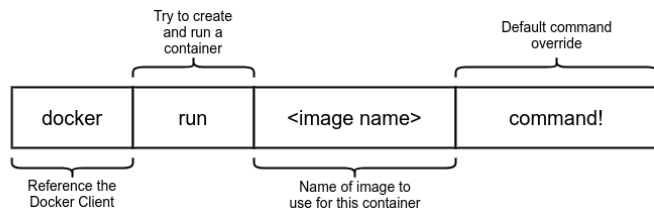
Creating and Running a Container from an Image



W5 Środowisko Docker

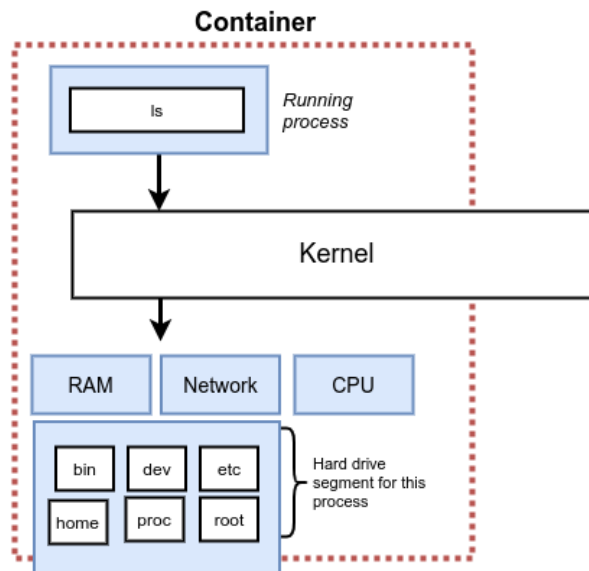
Wybrane polecenia w środowisku Docker (2)

Creating and Running a Container from an Image



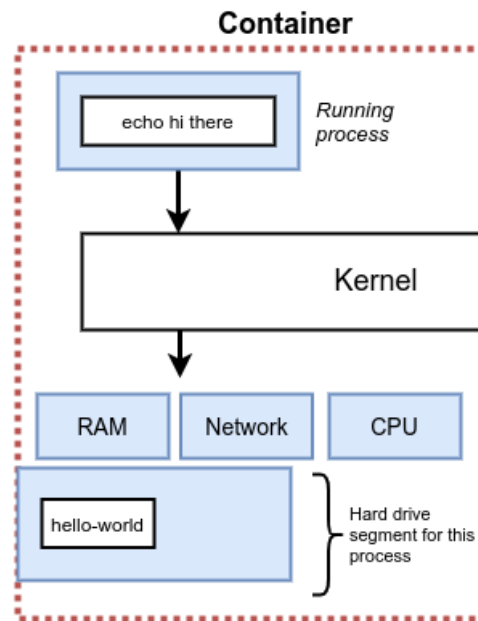
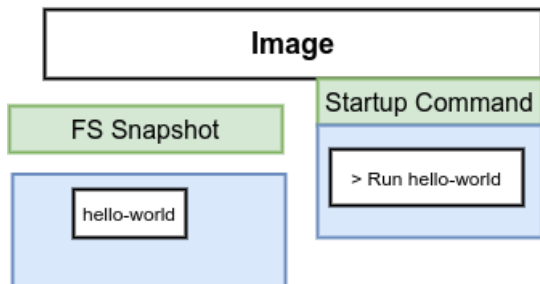
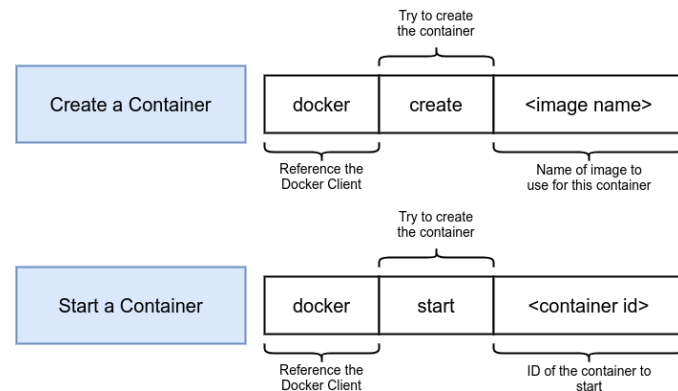
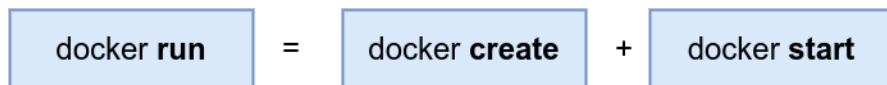
```
docker run busybox
```

```
docker run busybox ls
```



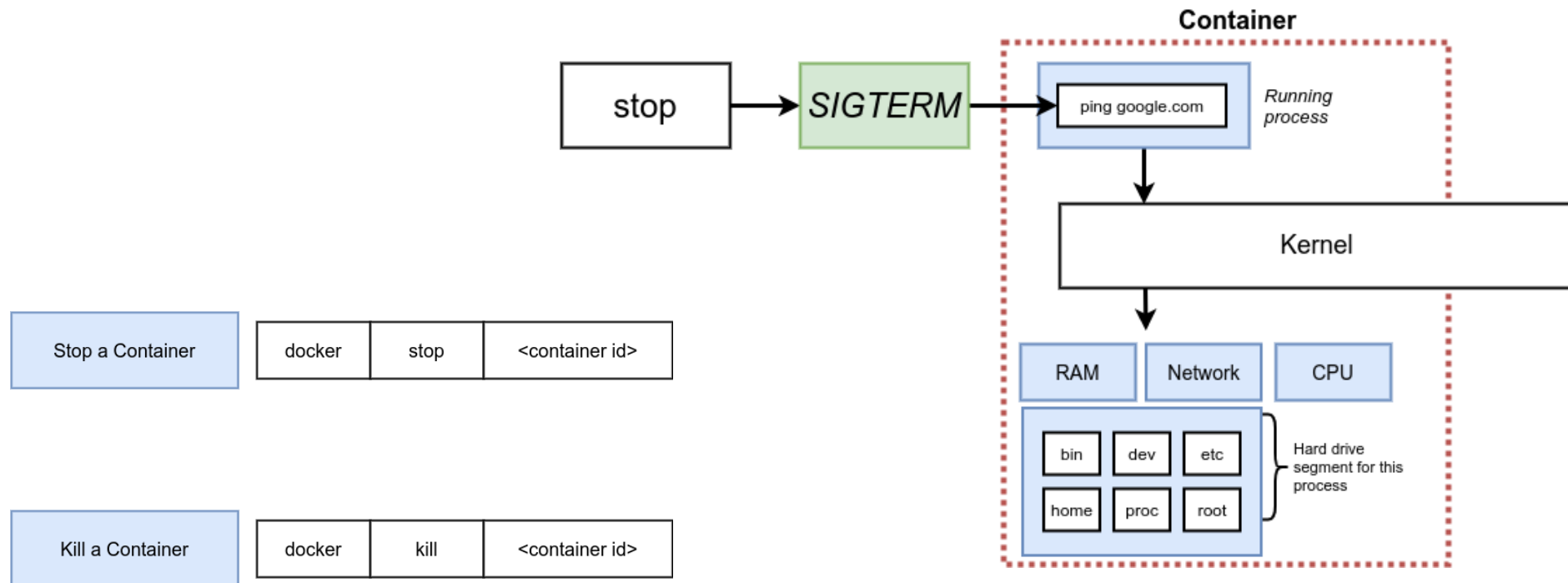
W5 Środowisko Docker

Wybrane polecenia w środowisku Docker (3)



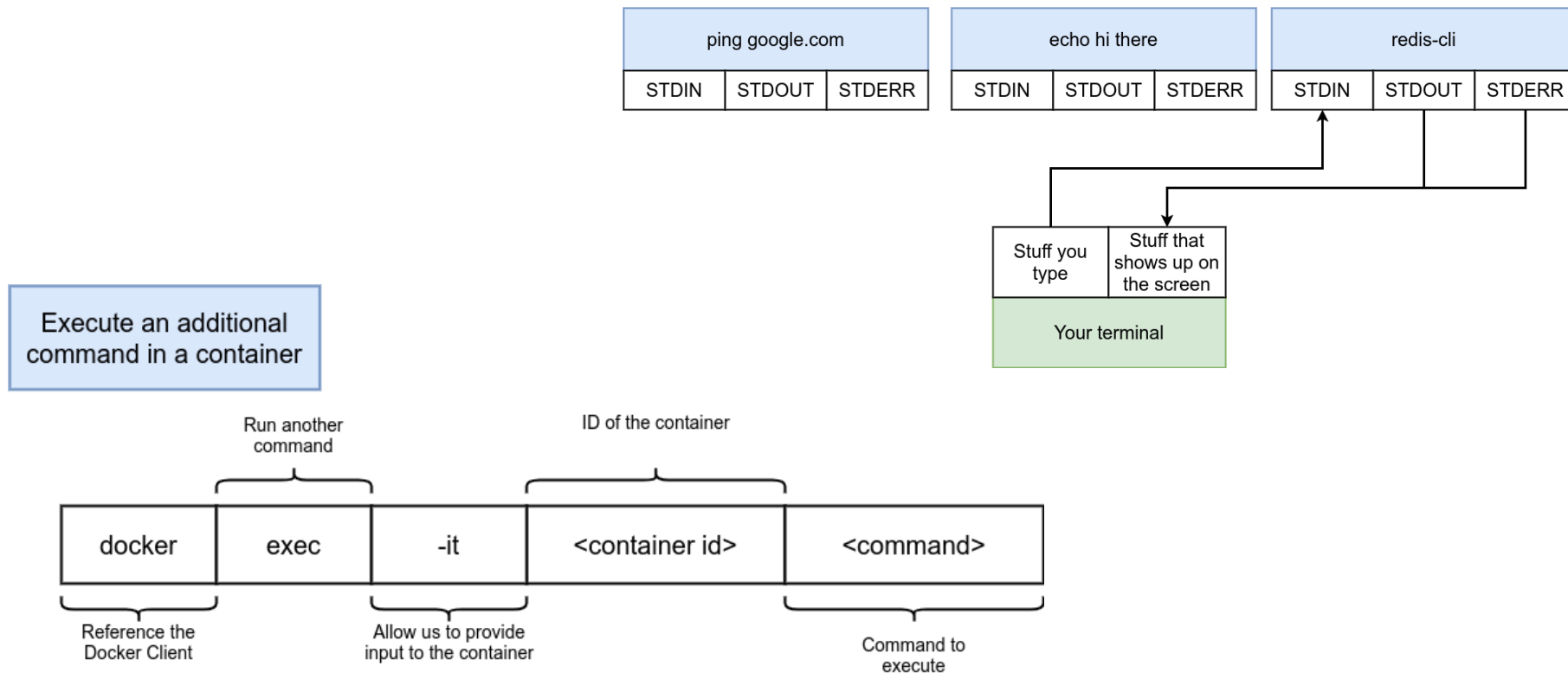
W5 Środowisko Docker

Wybrane polecenia w środowisku Docker (4)



W5 Środowisko Docker

Wybrane polecenia w środowisku Docker (5)



W5 Środowisko Docker

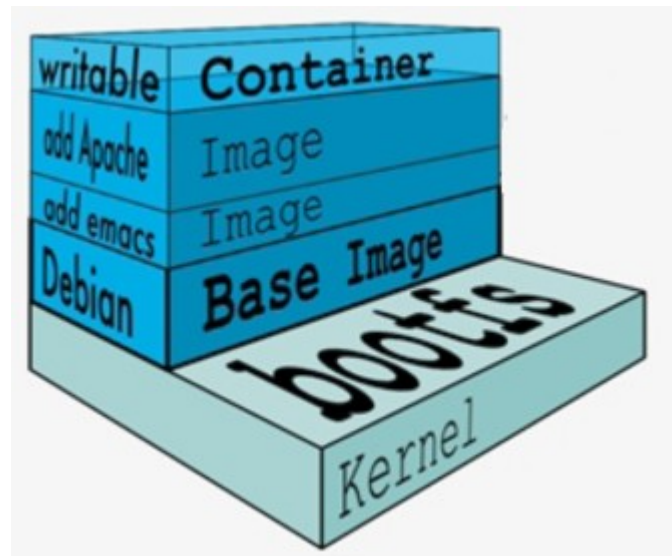
Obrazy w środowisku Docker (1)



Pobranie obrazu składało się z kilku etapów, każdy zawierał proces pobrania tzw. warstwy obrazu oraz jej rozpakowanie na lokalnym systemie plików. Wynika z tego, że obrazy posiadają strukturę warstwową. Dokładniej rzecz ujmując, cały obraz składa się z tzw. manifestu oraz kolejnych warstw.

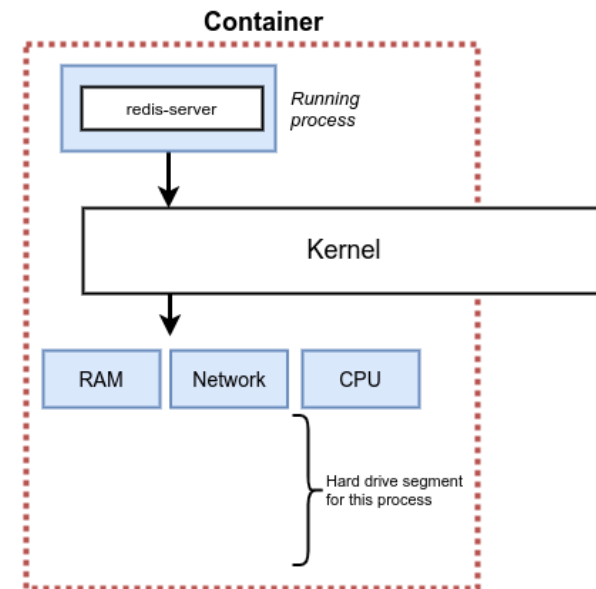
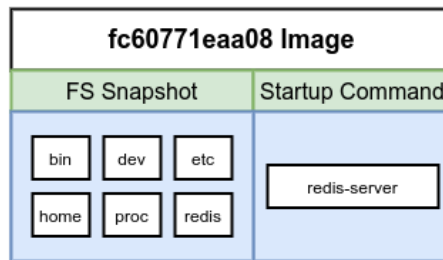
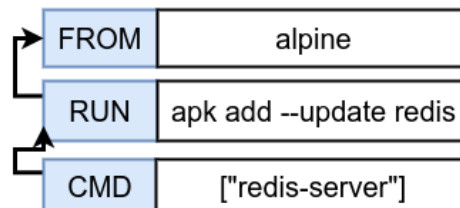
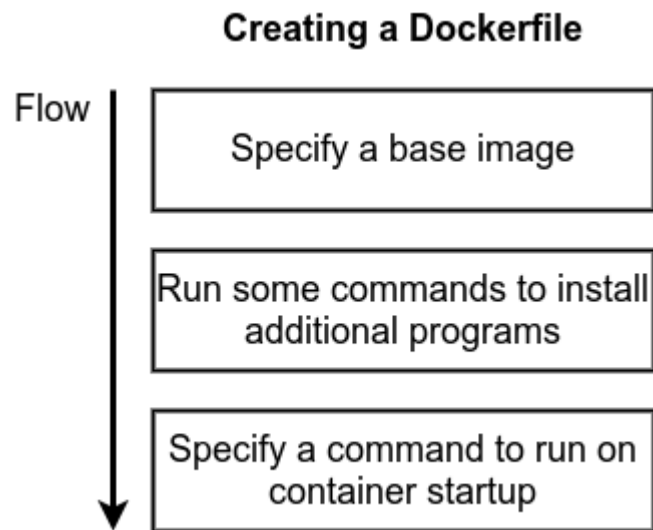
<https://docs.docker.com/registry/spec/manifest-v2-2/>

Warstwy powstają w wyniku kolejnych działań, które mają doprowadzić do zapewnienia dostępności niezbędnych komponentów programowych dla aplikacji uruchamianej w kontenerze opartym o ten obraz.



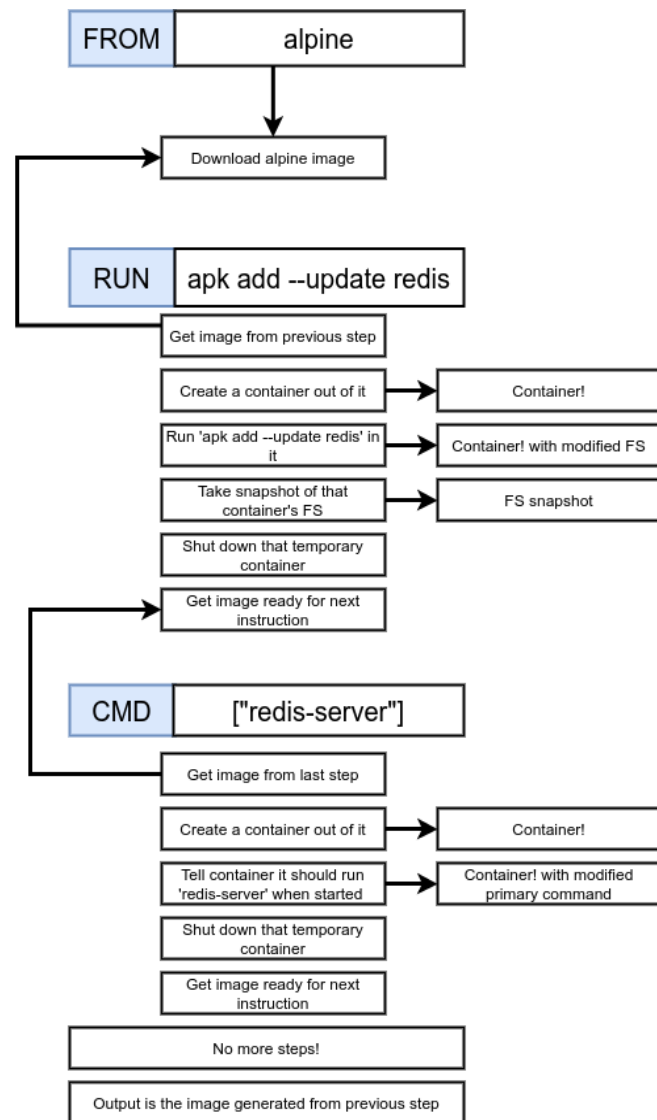
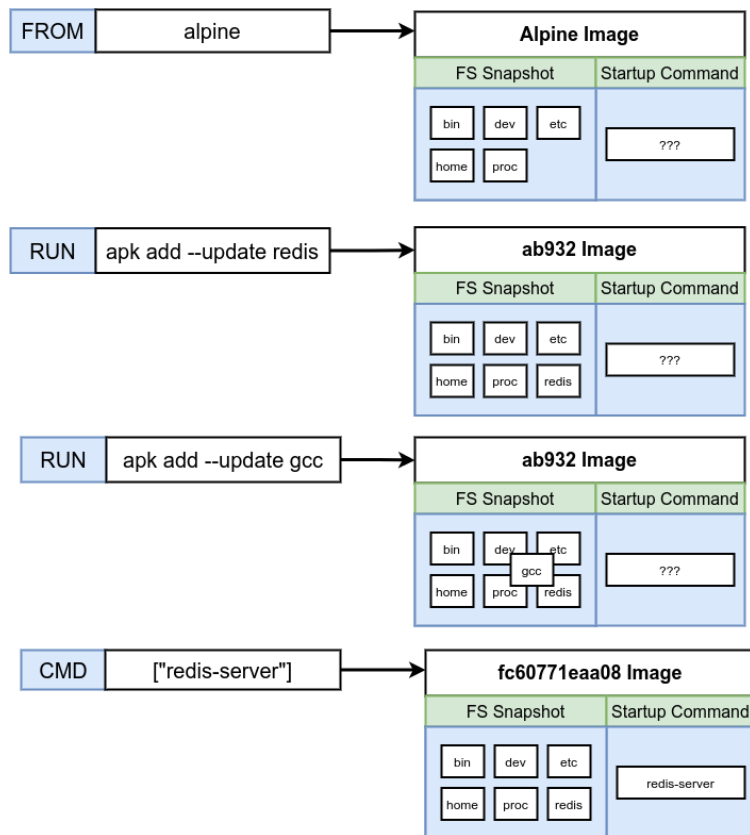
W5 Środowisko Docker

Budowanie obrazów Docker (1)



W5 Środowisko Docker

Budowanie obrazów Docker (2)



W5 Środowisko Docker

Budowanie obrazów Docker (3)

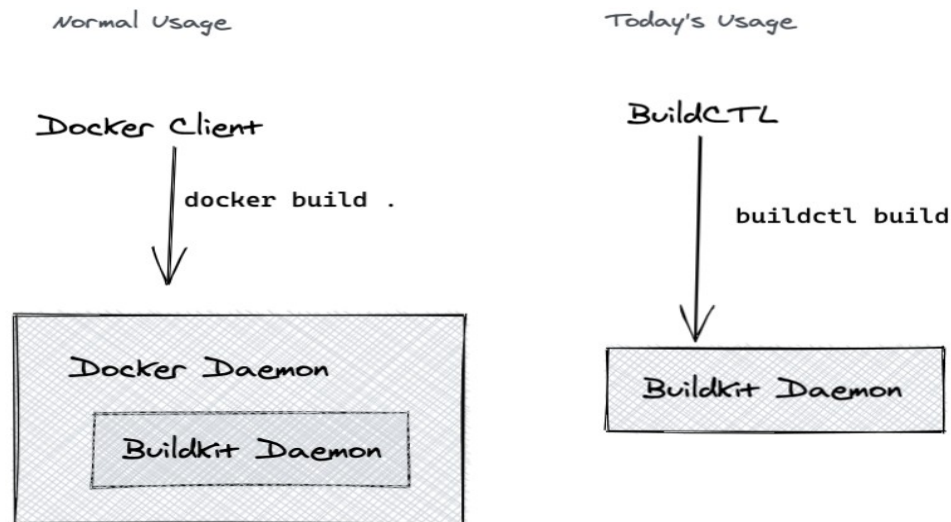
BuildKit

Docker Docs:

https://docs.docker.com/develop/develop-images/build_enhancements/

GitHub:

<https://github.com/moby/buildkit>



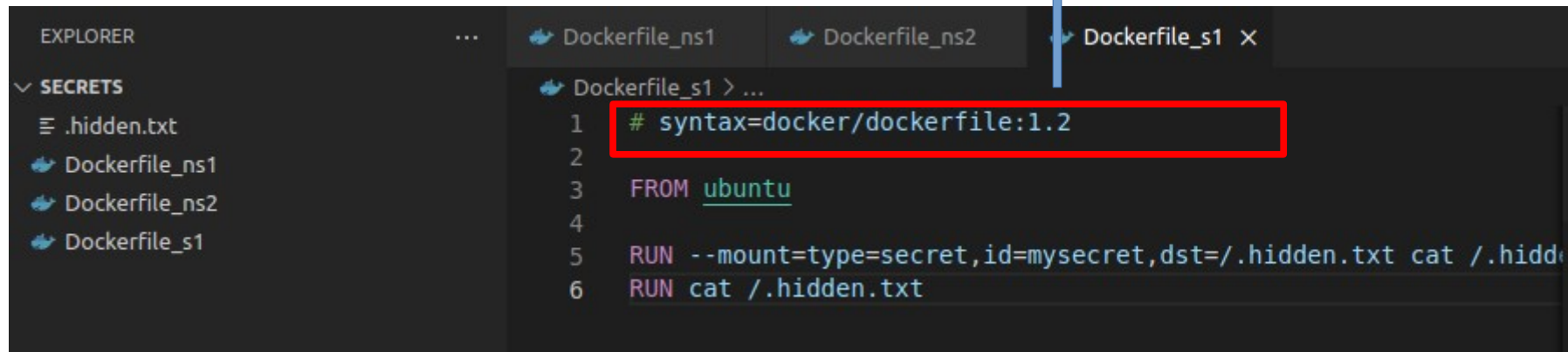
W5 Środowisko Docker

Budowanie obrazów Docker (4)

BuildKit - secrets

https://docs.docker.com/develop/develop-images/build_enhancements/#overriding-default-frontends

<https://docs.docker.com/engine/reference/builder/#syntax>



The screenshot shows a code editor with three tabs: Dockerfile_ns1, Dockerfile_ns2, and Dockerfile_s1. The left sidebar shows a file explorer with a 'SECRETS' section containing .hidden.txt, Dockerfile_ns1, Dockerfile_ns2, and Dockerfile_s1. The main editor area shows the content of Dockerfile_s1, which includes a red box highlighting the line `# syntax=docker/dockerfile:1.2`. A blue arrow points from this line to the URL <https://docs.docker.com/engine/reference/builder/#syntax> above it. The rest of the Dockerfile content is as follows:

```
1 # syntax=docker/dockerfile:1.2
2
3 FROM ubuntu
4
5 RUN --mount=type=secret,id=mysecret,dst=/.hidden.txt cat /.hidden.txt
6 RUN cat /.hidden.txt
```

```
$ DOCKER_BUILDKIT=1 docker build --progress=plain --secret  
id=mysecret,src=.hidden.txt -f Dockerfile_s1 -t local/us1 .
```

W5 Środowisko Docker

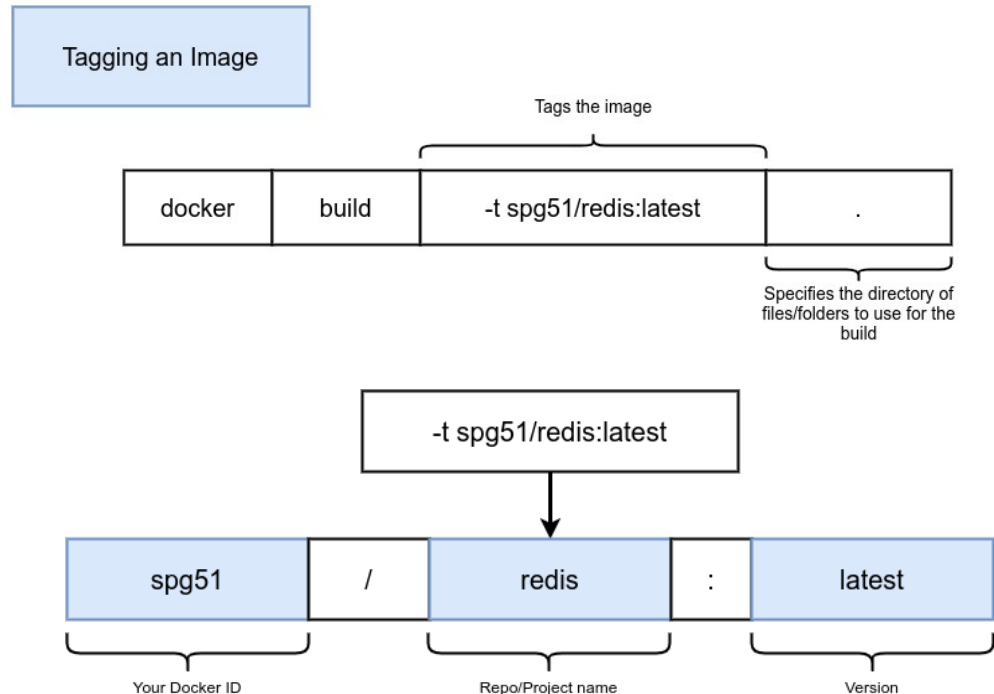
Nazwy obrazów



DockerHub



<https://hub.docker.com/>



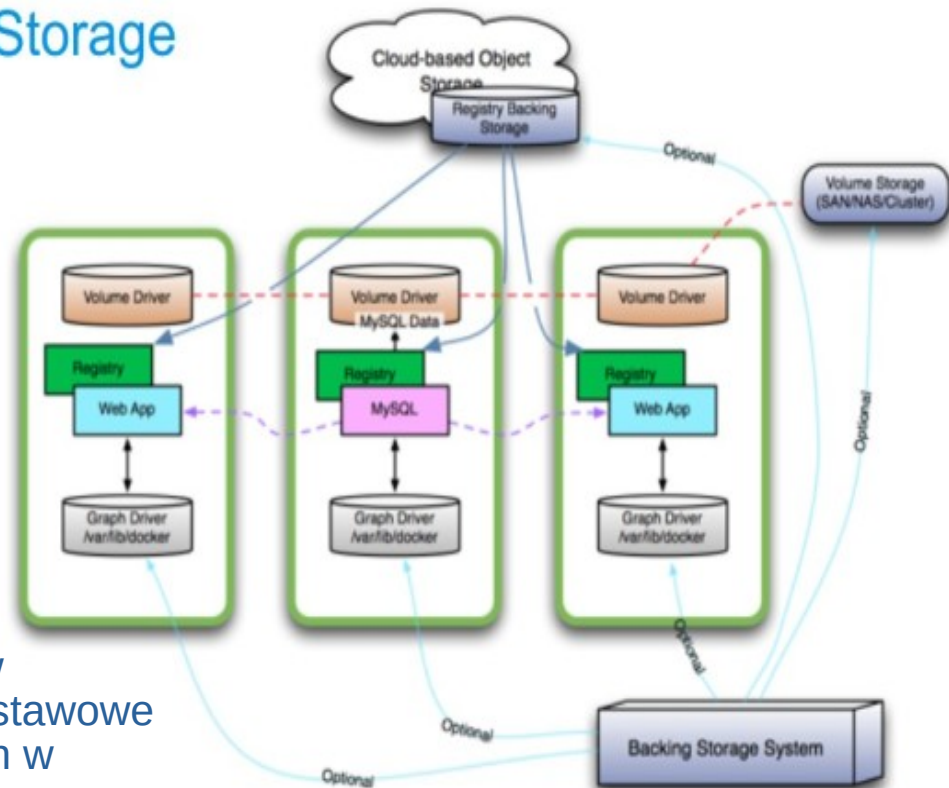
W5 Środowisko Docker

Przechowywanie danych w środowisku Docker (1)

Przechowywanie danych wewnątrz kontenera jest możliwe w najwyższej warstwie stosu systemu plików kontenera (warstwa RW). Należy jednak pamiętać, w kontenery Docker są ulotne co oznacza, że, działają tak długo jak długo wykonywane jest zadanie (działa aplikacja) zlecona do wykonania w kontenerze. Po jej zakończeniu, dane zapisane w tej warstwie systemu plików kontenera są bezpowrotnie tracone.

Najprostsze rozwiązanie polega na wykorzystaniu plików Dockerfile oraz komendy VOLUME. Generalnie trzy podstawowe mechanizmy odpowiedzialne za przechowywanie danych w środowisku Docker

Docker Storage



W5 Środowisko Docker

Przechowywanie danych w środowisku Docker (2)

Dobór sterownika przechowywania danych zależy od funkcjonalności tak systemu macierzystego, zainstalowanego na nim systemu operacyjnego jak o specyficznych wymagań określonego projektu.

<https://docs.docker.com/storage/storagedriver/select-storage-driver/>

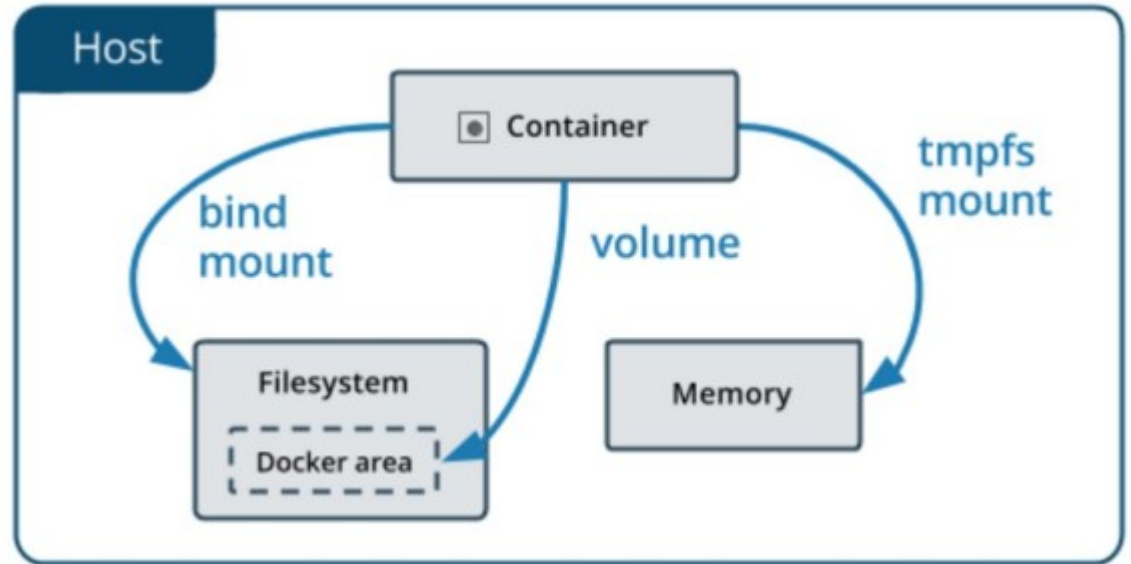
Storage driver	Supported backing filesystems
<code>overlay2</code> , <code>overlay</code>	<code>xfs</code> with <code>ftype=1</code> , <code>ext4</code>
<code>aufs</code>	<code>xfs</code> , <code>ext4</code>
<code>devicemapper</code>	<code>direct-lvm</code>
<code>btrfs</code>	<code>btrfs</code>
<code>zfs</code>	<code>zfs</code>
<code>vfs</code>	any filesystem

W5 Środowisko Docker

Pojęcie wolumenów i sposób ich wykorzystania

Wolumen to określony katalog w kontenerze, który zostaje przeznaczony na przechowywanie danych trwałych, niezależnie od cyklu życia konteneru.

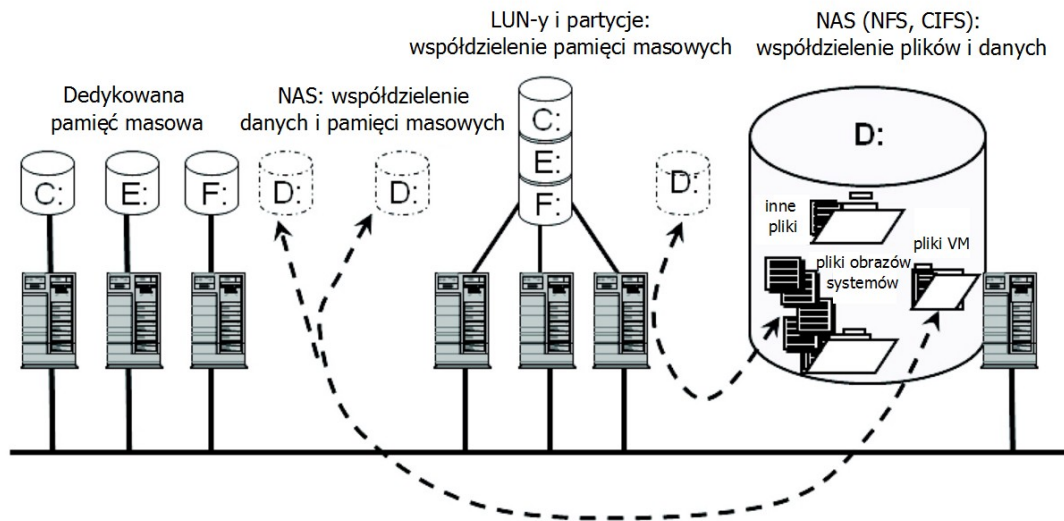
- Wolumeny nie są częścią obrazów
- Są nadal utrzymywane, jeśli obrazy zostaną skasowane
- Mogą być zamapowane na katalog znajdujący się na host
- Mogą być zamapowane w wielu kontenerach (współdzielone)
- Wolumeny są deklarowane w Dockerfile i montowane przy starcie kontenera
- Ścieżki muszą być absolutne (od /)



W5 Przechowywanie danych

Współdzielenie danych i pamięci masowych (1)

W przypadku **współdzielenia pamięci masowych** (ang. shared storage), poszczególne serwery otrzymują dostęp do części pamięci, typowo zorganizowanej jako partycja, dysk lub wolumen logiczny lub LUN (ang. Logical Unit Number).



Współdzielenie danych (ang. shared data) związane jest z odczytem lub zapisem danych przez wiele serwerów z pojedynczego pliku. Realizacja tego rodzaju współdzielenia odbywa się za pośrednictwem oprogramowania i dedykowanych protokołów, np. NFS (ang. Network File System) czy CIFS (ang. Common Internet File System).

W5 Przechowywanie danych

Struktura a dostęp do danych

Dane o zdefiniowanej strukturze (ang. structured data) posiadają przypisany zestaw atrybutów, które upraszczają różne operacje na danych, np. przeszukiwanie. Taka struktura zdecydowanie komplikuje zadania zmiany organizacji danych, np. dodawanie, grupowanie itp.

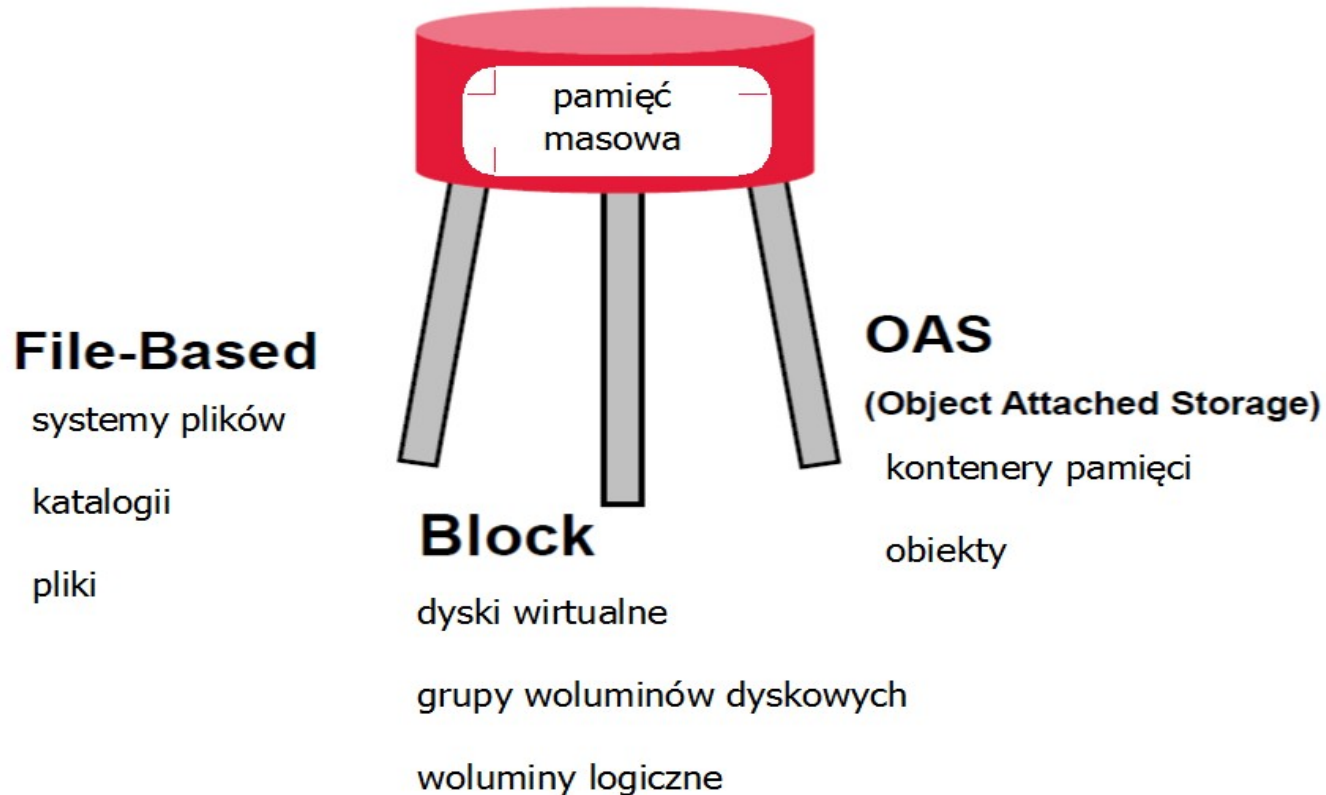
Alternatywą jest posługiwanie się danymi bez predefiniowanej struktury (ang. unstructured data), nazywanymi również jako dane o dostępie opartych o pliki (ang. file-accessed data). W tym przypadku jedyną formą organizacji danych są pliki przypisane do folderów lub katalogów w systemie plików.

Najbardziej ogólna klasyfikacja dostępu (lokalnego, zdalnego czy za pośrednictwem chmury) do danych i pamięci masowych obejmuje dostęp:

- za pośrednictwem API
- blokowy (ang. block-based access)
- plikowy (ang. file-based access)
- obiektowy (ang. object-based access)

W5 Przechowywanie danych

Metody dostępu do danych i pamięci masowych



W5 Przechowywanie danych

Dostęp blokowy

- Dostęp blokowy jest najniższym poziomem dostępu do danych i tym samym tworzy fundament dla wszystkich pozostałych tj, jest podstawą systemów dostępu do pamięci masowych dla struktur chmurowych jak i sieciowych pamięci masowych.
- W przypadku dostępu za pośrednictwem systemu plików, baz danych, systemów zarządzania dokumentami itp. szczegóły dostępu blokowego są ukrywane przez określony rodzaj abstrakcji. Abstrakcje te mogą przyjmować zróżnicowane formy, zależne od miejsca ich implementacji, np. systemy dyskowe ze wsparciem LVM, kontrolery RAID itd.

W5 Przechowywanie danych

Dostęp plikowy

- Dostęp plikowy jest obecnie najpopularniejszym sposobem dostępu. Dominuje w sieciowych systemach pamięci ale jest równie popularny w środowiskach zwirtualizowanych. W systemach chmurowych jest obecnie konkurentem dostępu obiektowego.
- Dostęp do danych tego typu wykorzystuje abstrakcję dla ukrycia dostępu blokowego w postaci nazwy pliku. Odnosi się zatem do danych bez predefiniowanej struktury.
- Operacje na danych wykorzystują określony typ semantyki poleceń, np. open, write itd. Dane z pliku pobierane są w blokach o określonym rozmiarze.

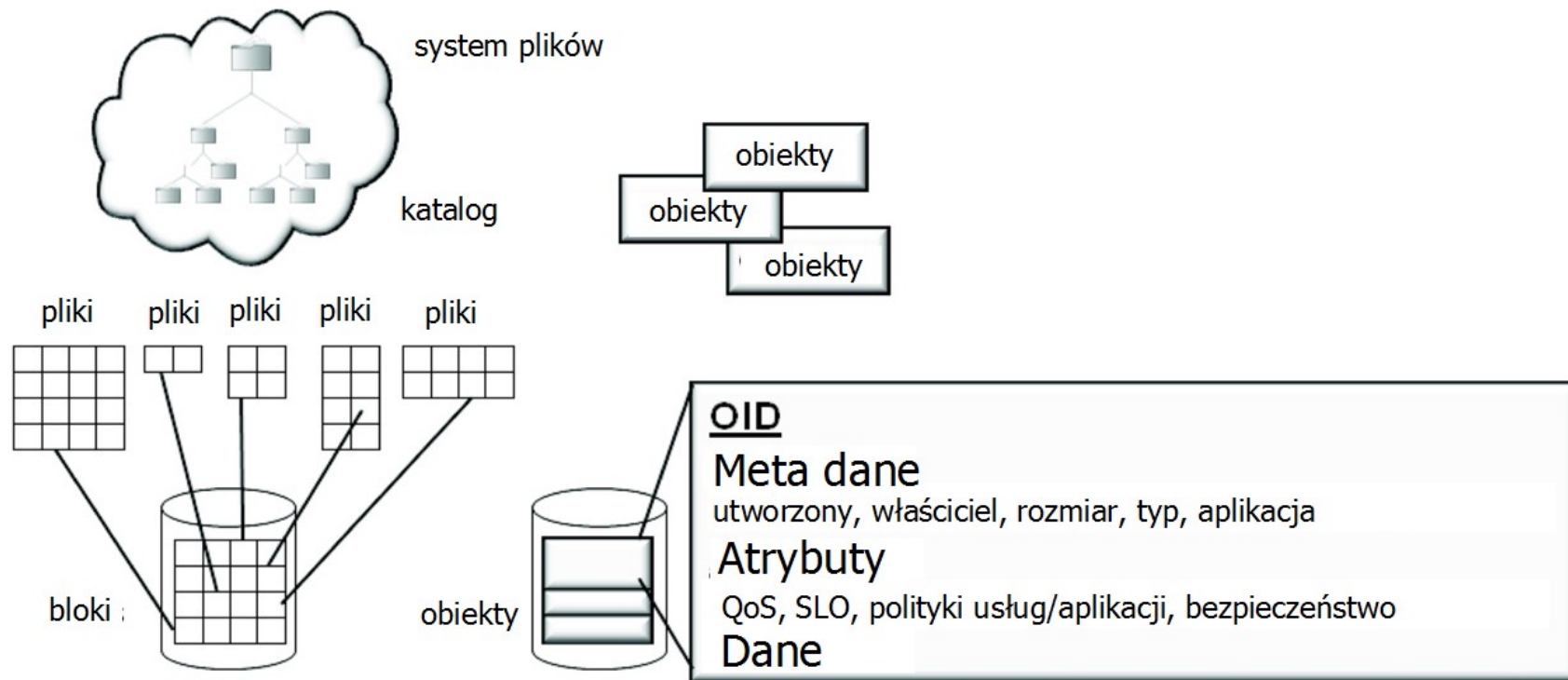
W5 Przechowywanie danych

Dostęp obiektowy

- Dostęp obiektowy dotyczy pamięci CAS (ang. content-addressable storage) czyli pamięci masowych o dostępie zbudowanych na bazie dwóch poprzednio omówionych rozwiązań. W tym mechanizmie dostępu dane są połączone z metadanymi (danymi opisującymi dane). Sposób i miejsce zapisu jest uzgadniane pomiędzy określoną aplikacją (procesem) a systemem plików a dalej, za jego pośrednictwem, z mechanizmami dostępu blokowego.
- Przykład meta danych to rozmiar pliku, informacja kto utworzył plik i ewentualnie dla kogo, atrybuty odczytu i zapisu, dane zabezpieczeń, typ danych

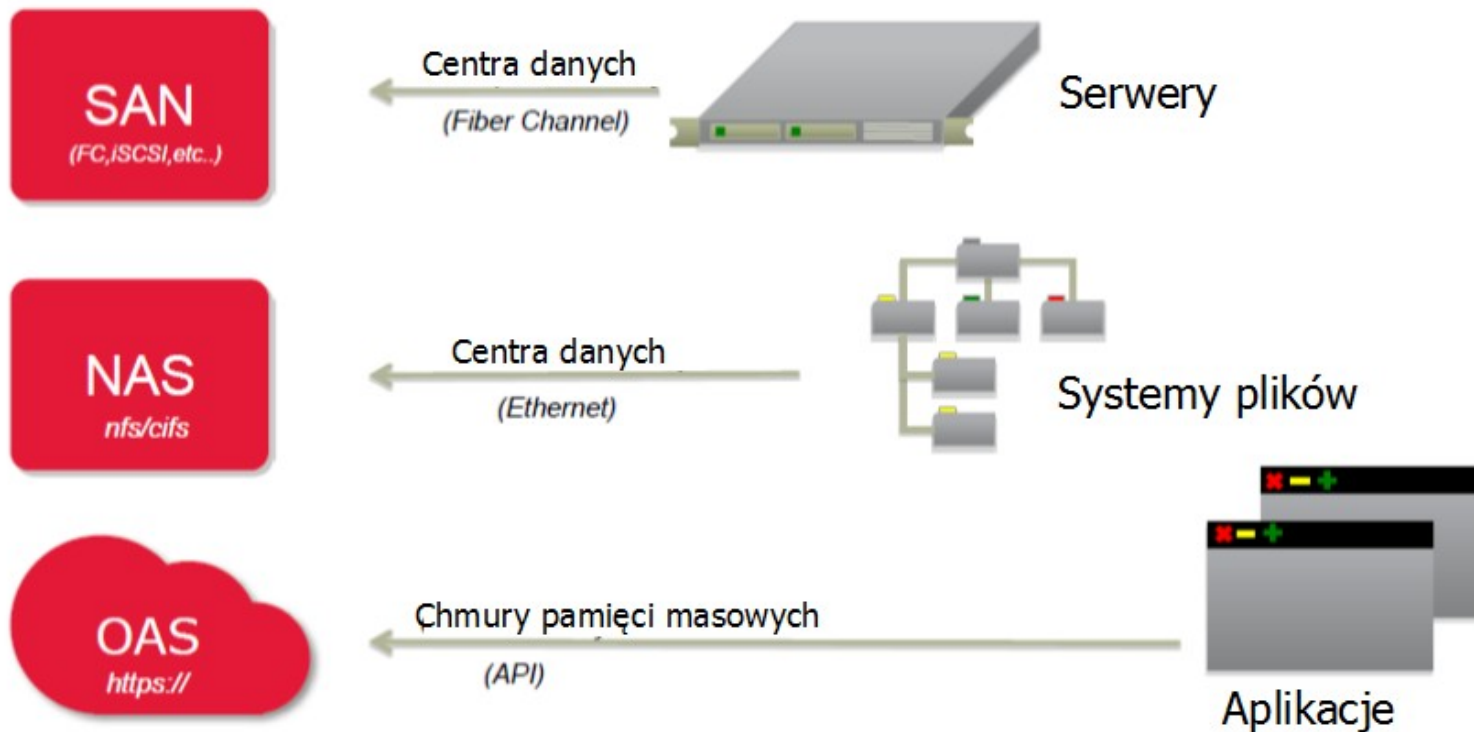
W5 Przechowywanie danych

Metody dostępu do danych i pamięci masowych - podsumowanie



W5 Przechowywanie danych

Inne spojrzenie na metody dostępu

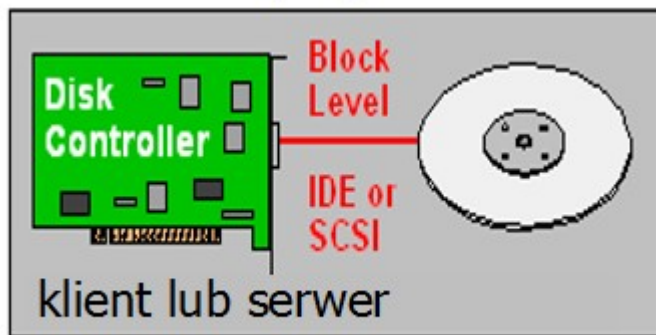


W5 Przechowywanie danych

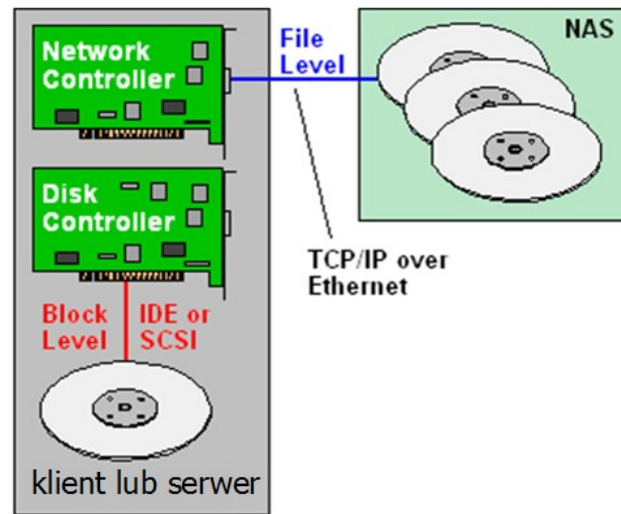
Trzy podstawowe struktury pamięci masowych (1)

- Direct access storage (DAS)
- Network attached storage (NAS)
- Storage area network (SAN)

Direct Attached Storage (DAS)



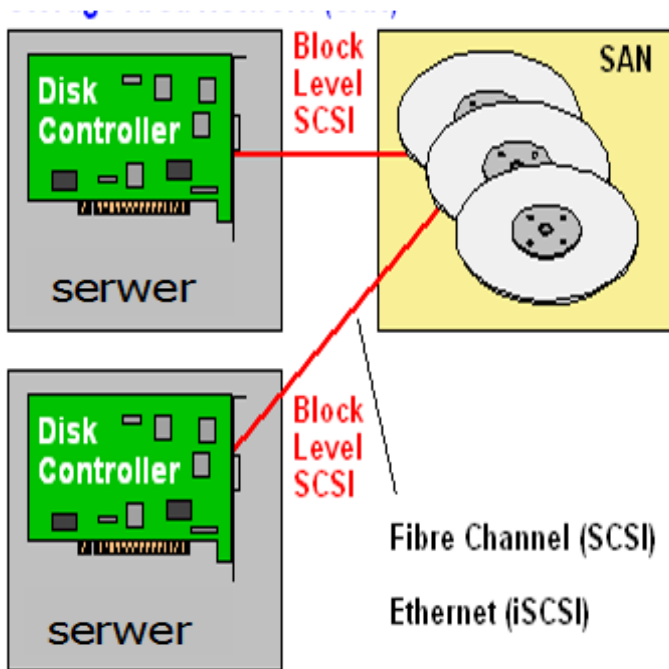
Network Attached Storage (NAS)



W5 Przechowywanie danych

Trzy podstawowe struktury pamięci masowych (2)

Storage Area Network (SAN)



	DAS	NAS	SAN
Typ współdzielenia pamięci	Sektory	Współdzielone pliki	Bloki
Transmisja danych	IDE/SCSI	TCP/IP, Ethernet	Fibre Channel
Dostęp	klienci lub serwery	klienci lub serwery	serwery
Pojemność (bajty)	10^9	$10^9 - 10^{12}$	$>10^{12}$
Złożoność	mała	średnia	wysoka
Koszt zarządzania (na 1 GB)	wysoki	średni	niski

W5 Przechowywanie danych

Trzy typy masowych pamięci sieciowych (1)

Direct Access Storage (DAS)

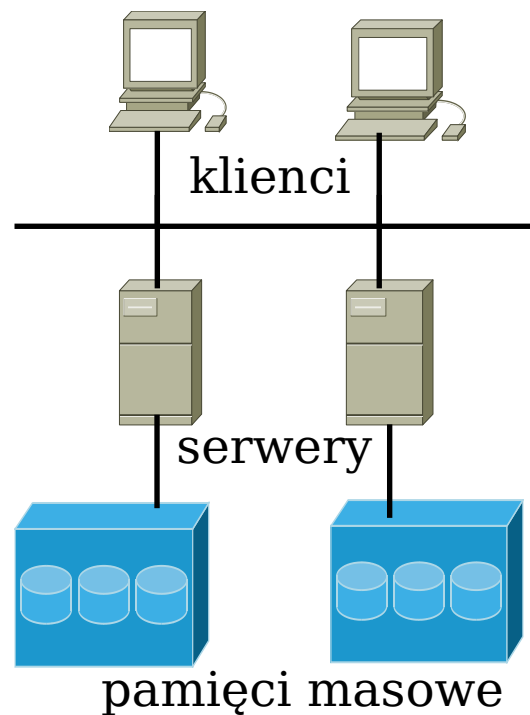
- SCSI
- RAID

Network Attached Storage (NAS)

Storage Area Network (SAN)

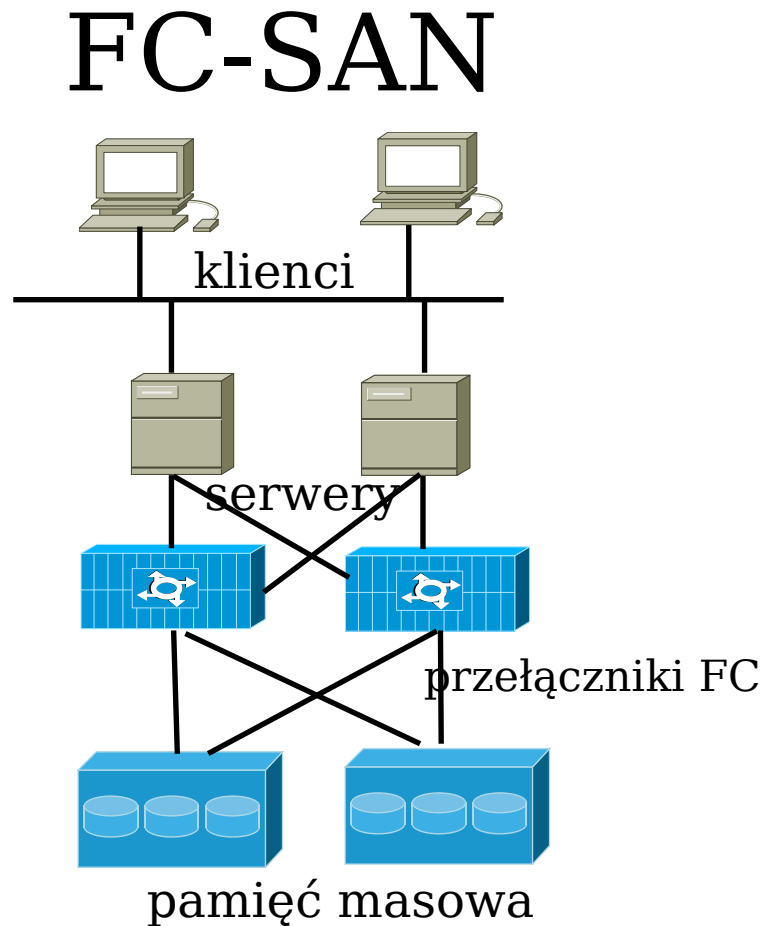
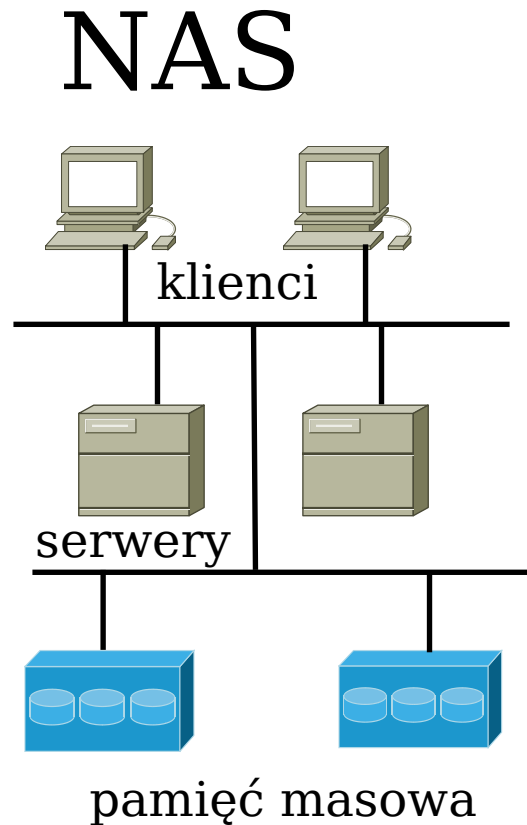
- Fiber Channel and
- Fiber Channel Switch

DAS



W5 Przechowywanie danych

Trzy typy masowych pamięci sieciowych (2)



W5 Przechowywanie danych

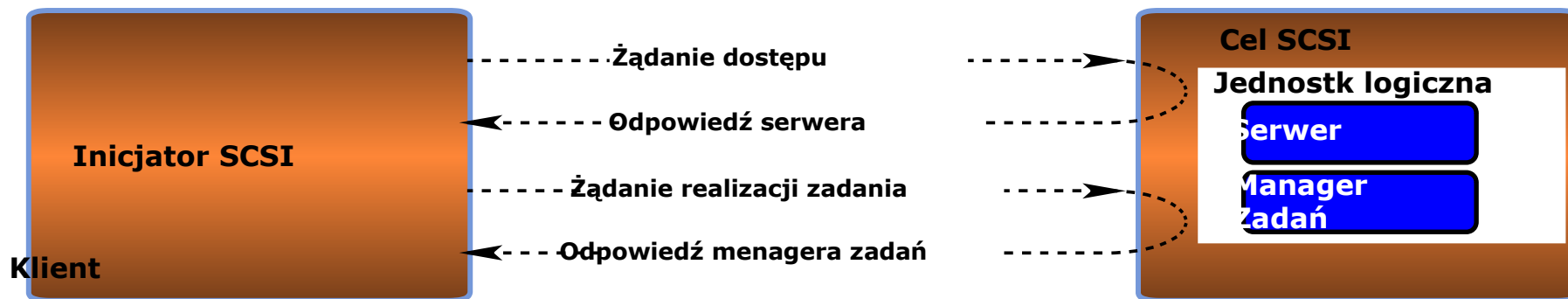
Interfejs SCSI - podstawy (1)

- Pierwotnie opracowany przez Shugart Associates i nazwany interfejsem SASI (ang. Shugart Associates System Interface)
- Wprowadzony przez ANSI jako międzynarodowy standard pod nazwą SCSI (ang. Small Computer System Interface)
- Najnowsza wersja standardu SCSI to
The Ultra standard 320 SCSI
(SCSI PARALLEL INTERFACES SPI-4)

SCSI wykorzystuje architekturę klient/server.

W5 Przechowywanie danych

Interfejs SCSI - podstawy (2)



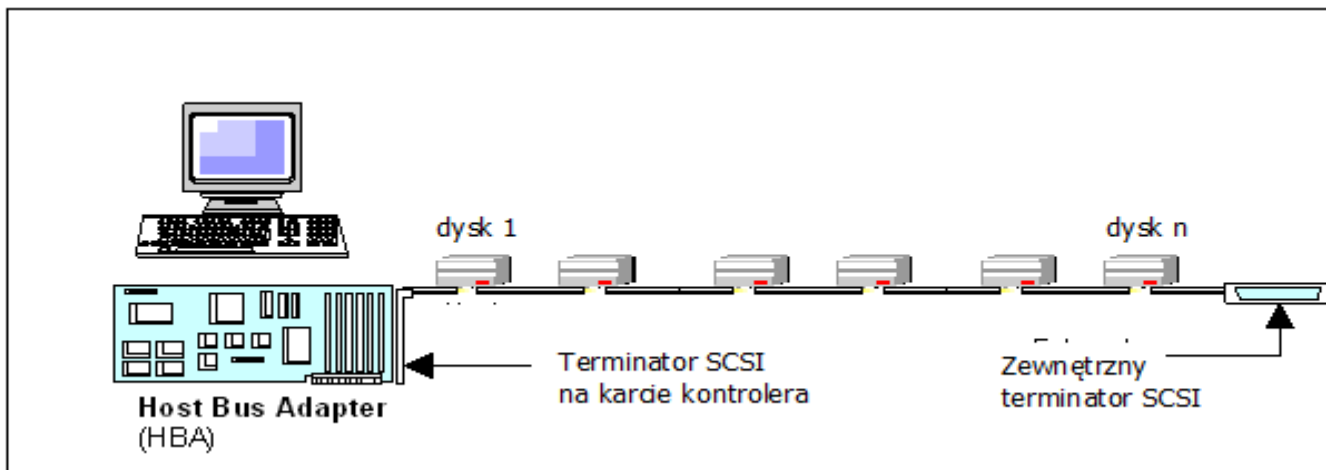
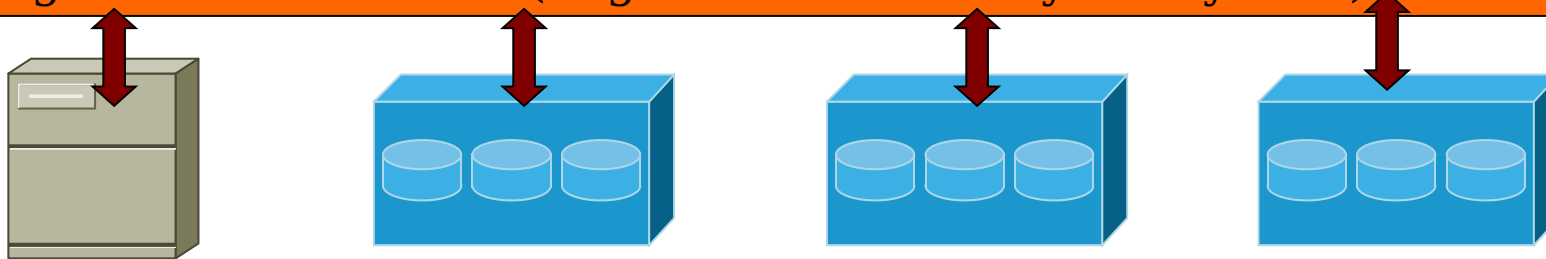
Klient nazywany jest inicjatorem (ang. **initiator**) i jest implementowany jako proces I/O nadzorowany przez system operacyjny.

Serwer jest nazywany celem (ang. **target**), zazwyczaj implementowany jako część oprogramowania kontrolera pamięci masowej.

W5 Przechowywanie danych

Domena SCSI

Magistrala SCSI - SDS (ang. Service Delivery Subsystem)



W5 Przechowywanie danych

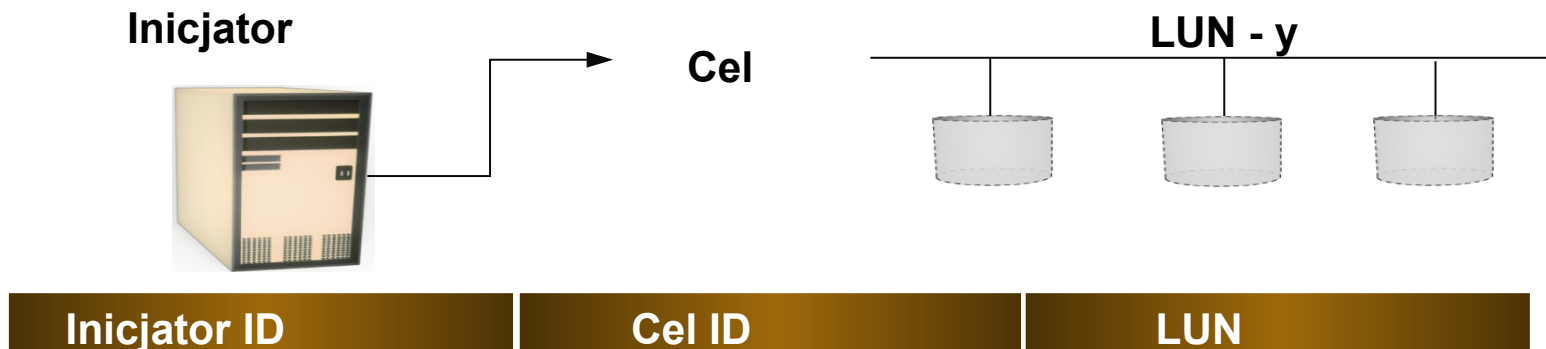
Interfejs SCSI – modele portów

- Urządzenia SCSI mogą zawierać porty inicjatora, celu lub port wielofunkcyjny cel/inicjator.
- Na podstawie struktury i funkcjonalności portów wyróżnia się modele urządzeń SCSI: initiator model, target model, target model z wieloma portami oraz target/initiator model.
- Treść komunikacji pomiędzy portami zależy od modelu portu ale zawsze wykorzystuje CDB (ang. Command Descriptor Block). Blok taki zawiera kod operacji, parametry polecenia i parametry kontrolne.

W5 Przechowywanie danych

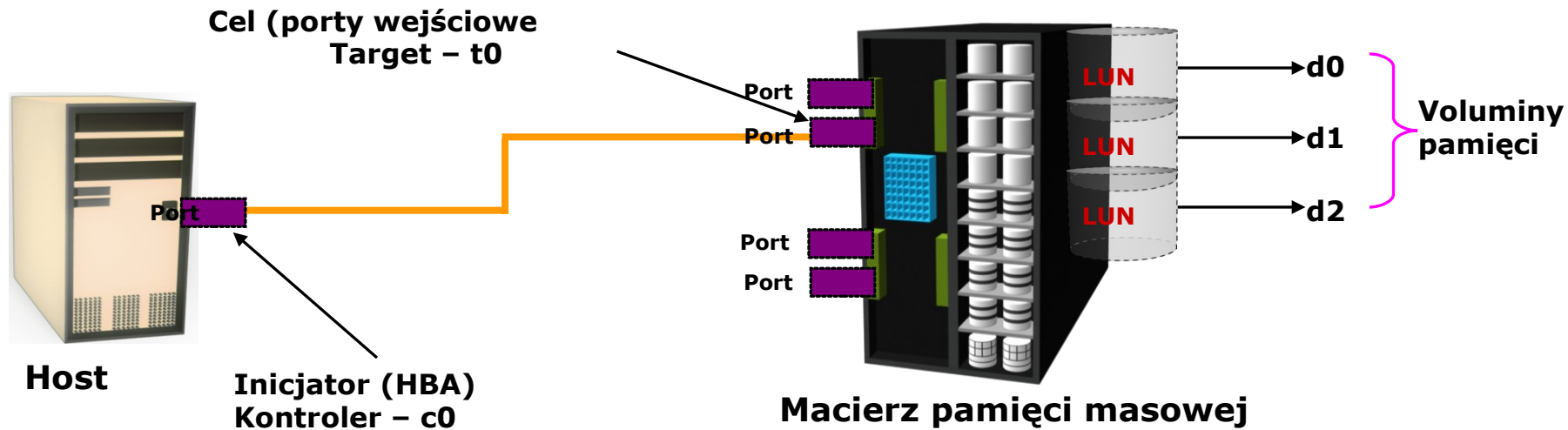
Interfejs SCSI – adresowanie (1)

- InitiatorID – liczba od 0 do 15, typowo równa 7.
- Cel ID – liczba od 0 do 15
- LUN – liczba, która określa adresowalne urządzenie (fizyczne lub wirtualne) obsługiwane przez cel.



W5 Przechowywanie danych

Interfejs SCSI – adresowanie (2)



Adresacja hosta

Volumin pamięci 1 - c0t0d0

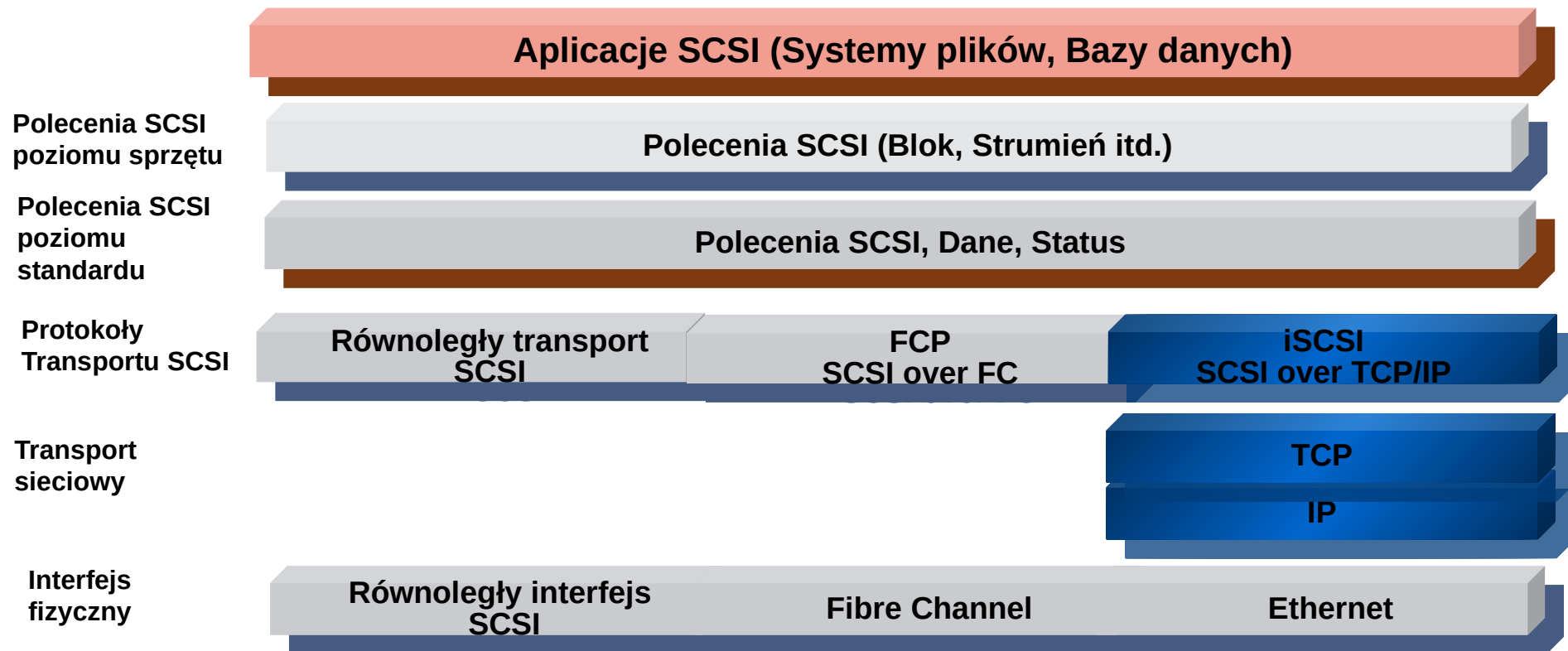
Volumin pamięci 2 - c0t0d1

Volumin pamięci 3 - c0t0d2

Inicjator ID	Cel ID	LUN
c0	t0	d0

W5 Przechowywanie danych

Interfejs SCSI – rozwiązania transportowe

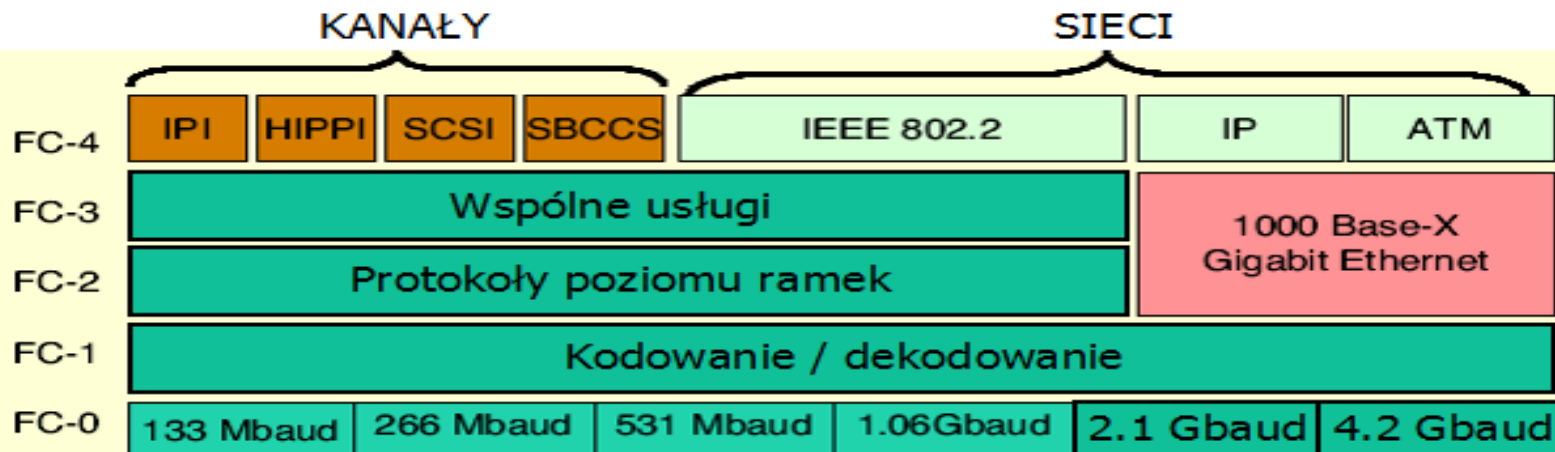


W5 Przechowywanie danych

Protokół Fibre Channel – podstawy

FC jest protokołem kanałowo-sieciowym

- Kanałowym: ponieważ potrafi zestawić kanały transmisji pomiędzy ograniczoną liczbą urządzeń. Ustanowiony kanał nie wymaga dalszej konfiguracji, co daje wysoką efektywność.
- Sieciowym: ponieważ potrafi obsłużyć złożone struktury połączeń urządzeń oraz ustalać trasy pomiędzy nimi.



W5 Przechowywanie danych

Protokół Fibre Channel – typy portów

- Każdy węzeł zazwyczaj posiada jeden fizyczny interfejs nazywany **N_Port**.

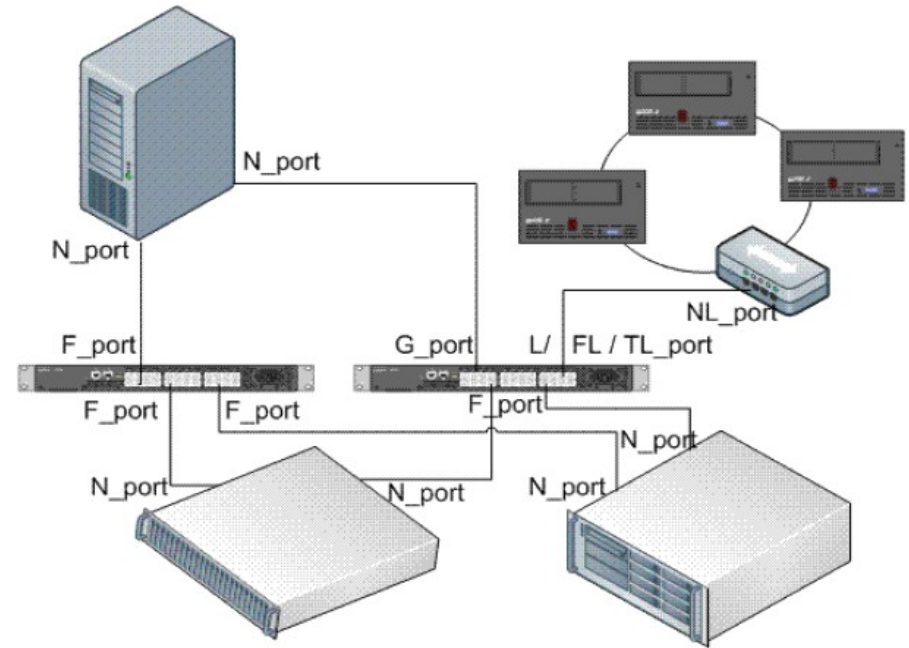
- Każdy węzeł ma przypisany 8-bajtową nazwę nadawaną przez wytwórcę.

Jeśli nazwa jest zarejestrowana w IEEE, określa się ją jako adres WWN (ang. World Wide Name). N_Port ID: 24-bitowy adres portu. N_

- Port może być połączony punkt – punkt z innym N_Portem lub portem strukturalnym (ang. fabric port), **F_port**.

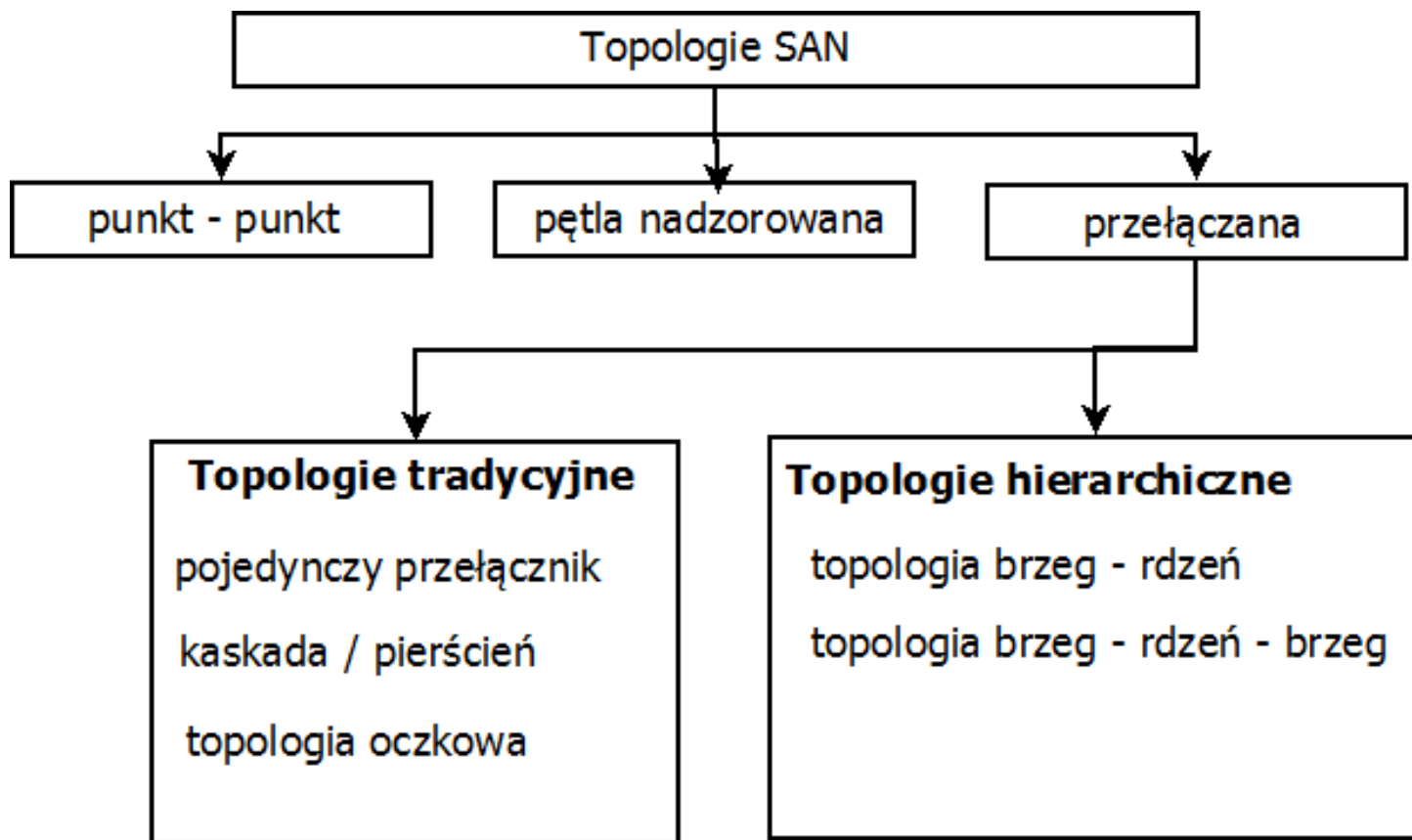
- Połączenia pomiędzy przełącznikami FC realizowane są przez porty rozszerzeń (ang. expansion ports), **E_ports**.

- Każdy port przełącznika FC jest tzw. **G_Port** (ang. generic port).



W5 Przechowywanie danych

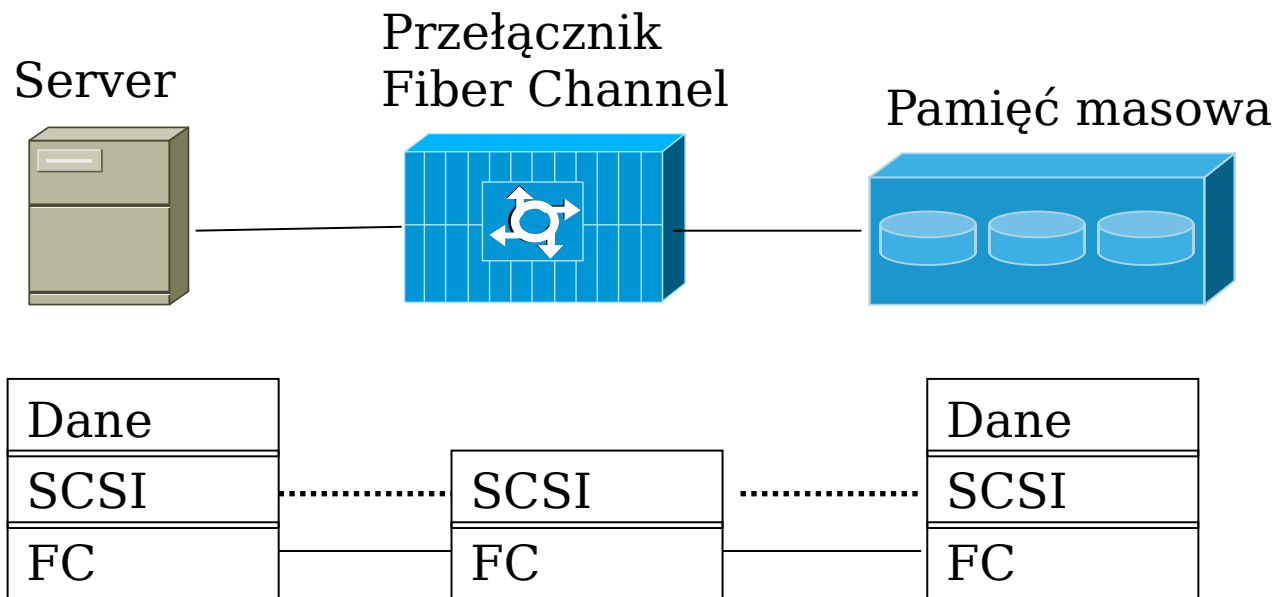
Protokół Fibre Channel – topologie



W5 Przechowywanie danych

Przełączane sieci FC-SW

Sieci SAN wykorzystujące przełączniki FC określane są mianem struktur FC-SW (ang. FibreChannel Switched Fabric).



W5 Przechowywanie danych

Klasyfikacja sieci FC-SW

Biorąc pod uwagę topologie połączeń, sieci FC-SW można przyporządkować do trzech kategorii:

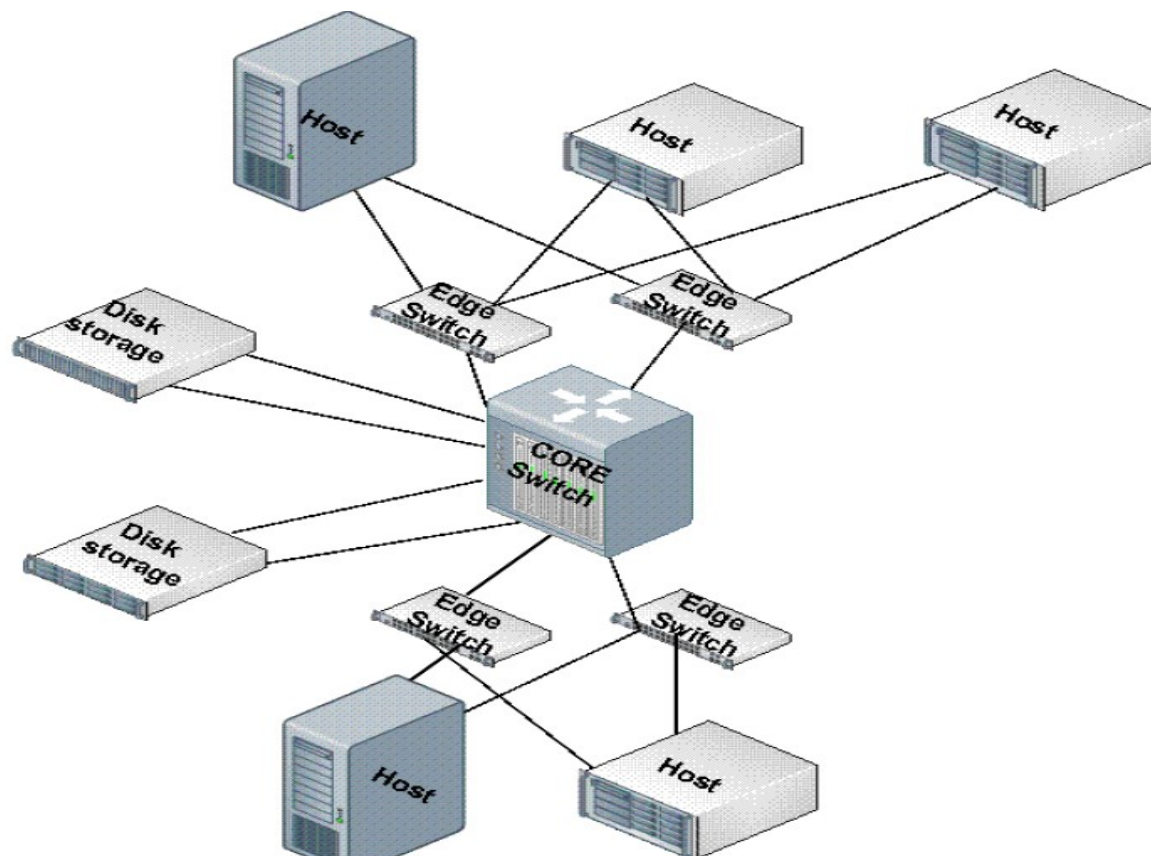
- Topologii z pojedynczym przełącznikiem
- Topologii kaskadowej
- Topologii oczkowych (ang. Mesh topology)

W praktyce wykorzystuje się bardziej złożone topologie warstwowe, a do najpopularniejszych z nich należy zaliczyć:

- Topologie rdzeń-brzeg (ang. Core-edge topology)
- Topologie brzeg-rdzeń-brzeg (ang. Edge-core-edge topology)

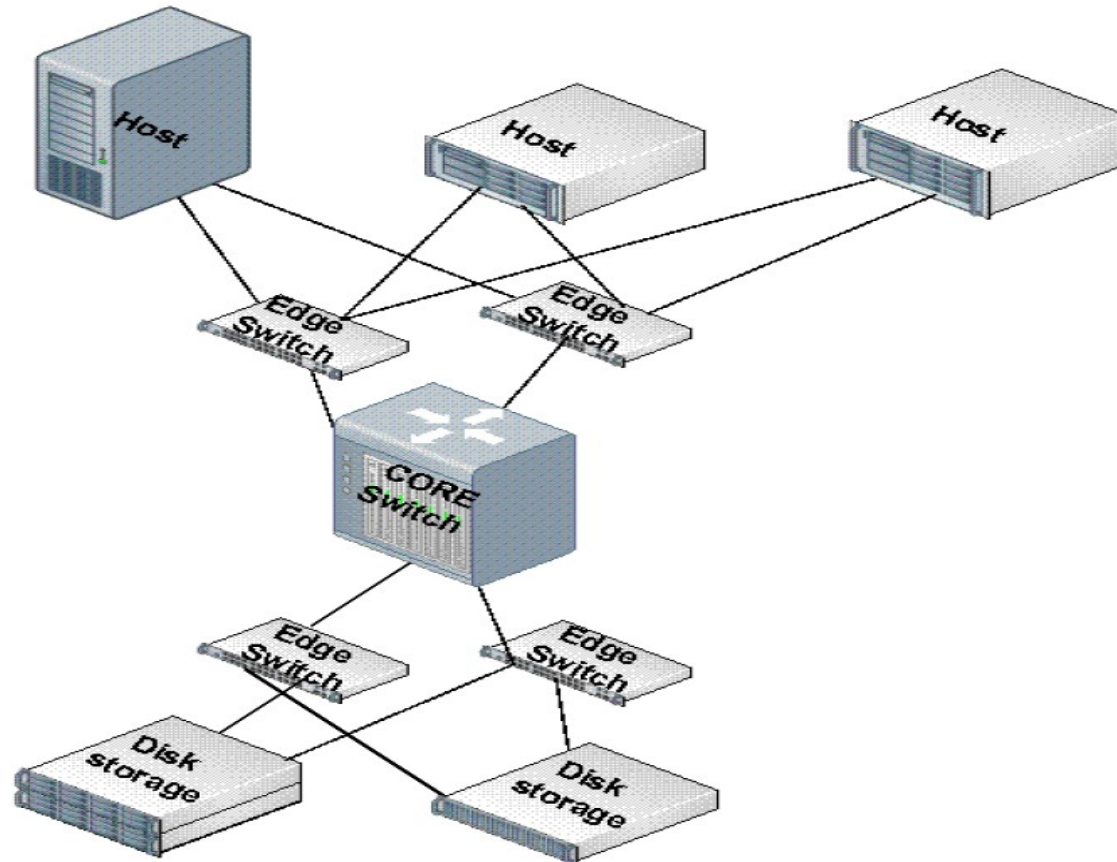
W5 Przechowywanie danych

Sieci FC-SW – topologia brzeg-rdzeń



W5 Przechowywanie danych

Sieci FC-SW – topologia brzeg-rdzeń-brzeg



W5

PYTANIA / UWAGI ?

