Poznajemy wysoką dostępność

Opowieść, pojęcia, poziomy rezerwy i wzorce projektowania.

- Wysoka dostępność: usługa, która nie ma jednego serwera
- Wysoka dostępność: pojęcia, liczenie dostępności i poziomy rezerwy
- Wzorce projektowania wysokiej dostępności: jak dobrać sposób do rodzaju usługi

Wysoka dostępność: usługa, która nie ma jednego serwera

Poziom: podstawy

Dotyczy: docker01, klaster hv01 i hv02, serwerownia Zielonej w Pamiątkowie

Czas: ~15 min

Ta strona otwiera książkę o tym, jak sprawić, żeby awaria jednego serwera nie zatrzymywała usługi. Zaczynamy od komputerów z lat siedemdziesiątych, które pierwsze wzięły to na serio, a kończymy na piątkowym popołudniu w Zielonej, gdy `docker01` przestał wstawać. Na końcu pokazujemy, jak ułożona jest książka.

Skróty użyte na tej stronie (4)

Skrót	Co znaczy
RPO	Recovery Point Objective: ile danych wolno stracić przy awarii
RTO	Recovery Time Objective: jak długo wolno nie działać przy awarii
VAX	Virtual Address eXtension: rodzina minikomputerów firmy Digital Equipment Corporation z lat 70. i 80.
VRRP	Virtual Router Redundancy Protocol: protokół, w którym kilka hostów uzgadnia, który z nich trzyma wspólny adres IP (opisany w RFC 5798)

Skąd się wzięło

> **Pojęcie: wysoka dostępność (HA, High Availability).** Takie zaprojektowanie usługi, żeby awaria pojedynczego elementu (serwera, dysku, zasilacza, kabla) nie zatrzymywała jej albo zatrzymywała na czas, który właściciel usługi z góry zaakceptował. Nie oznacza usługi, która nigdy się nie psuje, tylko taką, która psuje się w kontrolowany sposób.

Firma Tandem Computers powstała w 1974 roku, a pierwszy system z linii NonStop trafił do Citibanku w maju 1976 roku [1]. Pomysł był prosty: nie ma w nim jednego wspólnego elementu, bo każdy procesor ma własną pamięć, a procesory ustalają sprawy, wymieniając komunikaty. Po awarii jednego pracę przejmuje drugi [1].

W 1985 roku Jim Gray, jeden z inżynierów Tandema, opublikował raport „Why Do Computers Stop and What Can Be Done About It?” [2]. Z danych o ponad tysiącu lat pracy systemów wynikało, że sprzęt psuł się rzadziej niż ludzie i oprogramowanie, a największą część przestojów powodowały błędy administracji i błędy programów. To wniosek, do którego wracamy w tej książce: drugi serwer nie leczy złego `dnf update`.

Równolegle Digital Equipment Corporation zapowiedziało w maju 1983 roku VAXcluster, a pierwsza wersja z VAX/VMS 4.0 wyszła w 1984 roku [3]. Każdy komputer miał własny system, a dyski i blokady były wspólne dla całego klastra, więc użytkownik widział jeden duży komputer, a administrator mógł wyłączać jego części po kolei [3]. Ten wzorzec widzimy dziś w innych odmianach we Windows Failover Clustering, w Pacemakerze i w Kubernetesie.

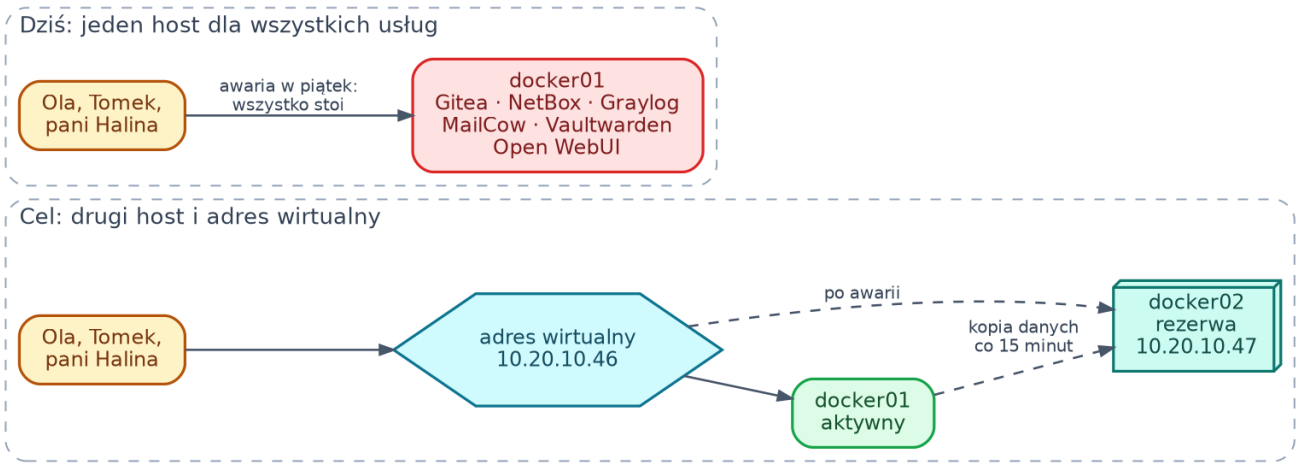
> **Pojęcie: pięć dziewiątek.** Dostępność 99,999 procent, czyli około 5 minut 15 sekund przestoju w roku. Nazwa wywodzi się z telekomunikacji, gdzie centrale telefoniczne miały działać prawie bez przerw [4].

Dokładnej daty powstania nazwy nie znamy. Najstarsze użycie „five nines” w sensie niezawodności, jakie znaleźliśmy, pochodzi z 1992 roku [5], więc nie łączymy jej z Tandemem ani z konkretnym rokiem lat siedemdziesiątych.

Jak to działa w jednym obrazku

Zamiast jednej maszyny, która robi wszystko, mamy dwie, które znają się nawzajem. Aplikacja ma jeden adres, pod którym użytkownicy ją znajdują, a ten adres przechodzi z maszyny na maszynę. Dane są na tyle świeże na obu, żeby przejęcie nie kosztowało nas pracy z całego dnia.

W Zielonej ten adres nazywamy adresem wirtualnym, a przenosi go protokół VRRP. Szczegóły są na kolejnych stronach, tu wystarczy zapamiętać trzy elementy: dwa hosty, adres, który skacze, i dane skopiowane z jednego na drugi.



Rysunek 1. Docker01 dziś i po zmianie. Dziś wszystkie aplikacje stoją na jednym hoście, a docelowo drugi host `docker02` ma świeżą kopię danych i przejmuje adres wirtualny `10.20.10.46`, gdy `docker01` przestaje odpowiadać. (kliknięcie otwiera pełny rozmiar)

Dlaczego Zielona

> **Z ŻYCIA WZIĘTE**
>
> Piątek, 15:20. Tomek zakończył na `docker01` aktualizację pakietów i zrestartował maszynę, żeby załadować nowe jądro. Klaster Hyper-V pokazuje, że maszyna jest uruchomiona, ale do konsoli wirtualnej nie dociera nic poza monitem rozruchowym, który prosi o naprawę systemu plików. Gitea, Vaultwarden i poczta z mailcow stoją, a Marek z magazynu w Swarzędzu dzwoni, że nie otwiera się jedno hasło potrzebne do drukarki etykiet.

Maszyna wirtualna dla Hyper-V żyje, więc klaster nie ma czego przenosić. Uszkodzony system plików gościa przechodzi razem z nią na drugi węzeł, jeśli zajdzie potrzeba. Odtworzenie z kopii restic, wzięcie haseł z kopii poza Vaultwardenem i start stosów po kolei zajmuje Tomkowi i Oli do późnego wieczoru, a poczta wraca dopiero po kolacji.

W poniedziałek pan Zbigniew pyta o jedno: skoro mamy klaster, to po co jeszcze drugi serwer. Odpowiedź ma dwie części. Klaster chroni przed utratą sprzętu, ale nie przed uszkodzonym systemem, złą aktualizacją czy błędem człowieka, a my chcemy, żeby w takim piątku usługa wróciła w kilkanaście minut, a nie w kilka godzin. Za tym idzie koszt, więc ta książka uczy też, kiedy drugi serwer się opłaca, a kiedy wystarczy dobra kopia.

Mapa tej książki

Książka idzie od teorii poziomów rezerwy przez przenoszenie kontenerów i replikację danych do klastrów Windows i Linux, równoważenia i orkiestracji, a kończy się macierzą wszystkich usług Zielonej.

Strona	Na jakie pytanie odpowiada
Wysoka dostępność: pojęcia, liczenie dostępności i poziomy rezerwy	Ile minut w roku to 99,9 procent, czym są RTO i RPO i jak dobieramy poziom do usługi
Przenoszenie kontenerów z docker01 na docker02: zimna i ciepła rezerwa, przełączenie	Jak z jednego hosta zrobić dwa i jak przełączyć usługę adresem wirtualnym
Replikacja danych między hostami: DRBD, GlusterFS, Ceph, ZFS i replikacja w aplikacji	Którą metodę kopiowania danych wybieramy i kiedy
Wzorce projektowania wysokiej dostępności: jak dobrać sposób do rodzaju usługi	Jak rodzaj usługi (plik, baza, usługa bezstanowa, maszyna wirtualna) wyznacza sposób utrzymania wysokiej dostępności i jakie zasady są wspólne dla wszystkich
Klaster Windows dla własnych usług: rola, zasób, zależności, kworum i testy przełączenia	Jak zrobić z własnej usługi na Windows rolę klastra, która przechodzi między węzłami
Klaster Windows od zera: budowa, sieć, świadek i S2D	Jak zbudować klaster Windows krok po kroku, od sieci po wspólny magazyn
Klaster Windows w utrzymaniu: aktualizacje CAU, monitoring i problemy	Jak utrzymujemy działający klaster Windows i co robimy, gdy coś w nim zawodzi
Pacemaker na Linuksie: własna usługa z adresem wirtualnym	Jak Pacemaker pilnuje usługi, przenosi ją na drugi węzeł i odcina węzeł, który przestał odpowiadać
Pacemaker w utrzymaniu: zarządzanie, testy i aktualizacje	Jak na co dzień zarządzamy klastrem Pacemaker, testujemy go i aktualizujemy
Proxmox VE: klaster trzech węzłów i HA	Jak zbudować klaster Proxmox VE i włączyć w nim wysoką dostępność maszyn
Proxmox VE w utrzymaniu: kopie PBS, aktualizacje, użytkownicy	Jak utrzymujemy klaster Proxmox VE, kopie i konta
Proxmox VE a Hyper-V z Failover Cluster: porównanie i przenosiny	Czym Proxmox VE różni się od Hyper-V z Failover Cluster i jak przenieść maszyny
Inne hypervisory z HA: oVirt, Oracle Linux Virtualization Manager i XCP-ng	Jak wysoką dostępność robią pozostałe platformy wirtualizacji
Równoważenie i adres wirtualny: keepalived i HAProxy na parze lb01 i lb02	Jak działają VRRP i kontrole stanu oraz jak ustawić je bez split brain
Docker Swarm: klaster z trzema menedżerami, na którym przenosimy kolorowy nginx	Co daje Swarm w 2026 roku i dlaczego nie jest naszym pierwszym wyborem

Strona	Na jakie pytanie odpowiada
k3s z trzema węzłami serwerowymi: etcd, adres wirtualny API i Longhorn	Jak zbudować klaster Kubernetesa, który przeżyje awarię węzła
Porównanie mechanizmów HA: Windows, Linux i hypervisory	Czym różnią się mechanizmy wysokiej dostępności na różnych systemach i kiedy który wybieramy
Macierz wysokiej dostępności usług Zielonej: poziom rezerwy każdej usługi, zależności i kolejność prac	Jaki poziom rezerwy ma każda usługa Zielonej, co od czego zależy i w jakiej kolejności to poprawiamy
Perforce i Lore w klastrze: cztery drogi, rekomendacja dla Zielonej i test przełączenia (dodatek: przykład praktyczny)	Jak zbudować klaster pod Perforce i Lore, żeby awaria węzła nie zatrzymywała zespołu
Perforce i Lore: porównanie dróg do wysokiej dostępności na Windowsie (dodatek: przykład praktyczny)	Które drogi do wysokiej dostępności Perforce i Lore porównaliśmy i którą wybieramy na start
Perforce i Lore na Linuksie: odpowiedniki dróg z Windowsa, pułapka wielkości liter i rekomendacja dla Zielonej (dodatek: przykład praktyczny)	Jak to samo zrobić na Linuksie i dlaczego wielkość liter jest pułapką przy przenosinach

Do kopii zapasowych i odtwarzania sięgamy do Kopia całego docker01: /docker/apps, /docker/compose i bazy w kontenerach, a o RPO i RTO baz piszemy w RPO i RTO: dwie liczby, od których zaczyna się każdy plan backupu.

Powiązane: Jak trzymamy aplikacje w Dockerze: konwencje Zielonej · S2D: wymiana dysku bez przestoju, gdy pula jest prawie pełna

Źródła:

1. en.wikipedia.org: Tandem Computers
2. Jim Gray, Why Do Computers Stop and What Can Be Done About It?, Tandem Technical Report 85.7, czerwiec 1985
3. en.wikipedia.org: VMScluster
4. networkworld.com: Five nines by the book
5. listserv.linguistlist.org: wątek listy ADS-L o pochodzeniu „five nines”

Wysoka dostępność: pojęcia, liczenie dostępności i poziomy rezerwy

Poziom: podstawy
Dotyczy: wszystkie usługi Zielonej, docker01, hv01 i hv02, pg01, fw01 **Czas: ~60 min**

Zanim ktoś zbuduje drugi serwer, trzeba wiedzieć, ile minut przestoju w roku firma jest w stanie znieść i ile danych wolno jej stracić. Na tej stronie tłumaczymy pojęcia (SPOF, RTO, RPO, kworum, split brain), pokazujemy, jak przeliczać dziewiątki na minuty, i opisujemy trzy poziomy rezerwy: zimną, ciepłą i gorącą. Wracamy tu przy każdej rozmowie z właścicielem usługi, który mówi, że usługa ma działać zawsze.

ZAŁOŻENIA SCENARIUSZA

Zielona ma pojedynczy `fw01` z jednym łączem (`wan1`, 203.0.113.10), co jest jednym z opisanych tu SPOF-ów; para FortiGate to dopiero rekomendacja: Para FortiGate w klastrze: FGCP aktywny-pasywny. Klaster `hv01` i `hv02` zaczynamy bez świadka (luka, którą strona wskazuje), w jednej serwerowni w Pamiątkowie, a kopie trzyma jeden fizyczny `bkp01`.

Skróty użyte na tej stronie (17)

Skrót	Co znaczy
API	Application Programming Interface: sposób, w jaki programy rozmawiają ze sobą
CLI	Command Line Interface: praca przez wpisywanie komend w terminalu
DNS	Domain Name System: książka telefoniczna internetu, zamienia nazwy na adresy IP
etcd	etc distributed: rozproszony magazyn par klucz-wartość, w którym Kubernetes trzyma cały stan klastra, a zmiany uzgadnia protokołem Raft
HA	High Availability: wysoka dostępność, czyli działanie mimo awarii jednego elementu
IETF	Internet Engineering Task Force: organizacja publikująca standardy internetowe jako dokumenty RFC

Skrót	Co znaczy
IP	Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)
MTBF	Mean Time Between Failures: średni czas pracy między awariami
MTTR	Mean Time To Repair: średni czas od awarii do powrotu usługi
RFC	Request for Comments: oficjalny dokument opisujący standard internetowy, na przykład sposób działania danego protokołu; w ITIL także Request for Change, czyli wniosek o zmianę
RPO	Recovery Point Objective: ile danych wolno stracić przy awarii
RTO	Recovery Time Objective: jak długo wolno nie działać przy awarii
S2D	Storage Spaces Direct: programowa macierz Windows Server z dysków wielu serwerów
SPOF	Single Point of Failure: element, którego awaria zatrzymuje całą usługę, bo nie ma dla niego zamiennika
STONITH	Shoot The Other Node In The Head: odcięcie węzła, co do którego nie mamy pewności, że jest martwy, najczęściej przez wyłączenie jego zasilania
UPS	Uninterruptible Power Supply: zasilacz awaryjny, podtrzymuje zasilanie urządzeń na czas zaniku napięcia z sieci
VRRP	Virtual Router Redundancy Protocol: protokół, w którym kilka hostów uzgadnia, który z nich trzyma wspólny adres IP (opisany w RFC 5798)

Fiszka

Komendy wykonujemy z CLI bash na dowolnym hoście z Linuksem (`docker01` albo stacja administratora), a polecenie PowerShell na `hv01` lub `hv02` w sesji administratora. Żadna z nich niczego nie zmienia.

TYLKO ODCZYT

Obliczenie dopuszczalnego przestoju w roku: awk

```
awk -v p=99.9 'BEGIN { printf "%.1f min rocznie\n", 365*24*60*(100-p)/100 }'
```

Zmienna `p` to dostępność w procentach, a wynik to minuty przestoju w roku liczonym jako 365 dni. Dla 99,9 procent wychodzi 525,6 minuty, czyli 8 godzin 45 minut i 36 sekund.

Przykład:

```
525.6 min rocznie
```

□□ **awk** = od nazwisk autorów Aho, Weinberger, Kernighan: język do danych w kolumnach

□□ TYLKO ODCZYT

Obliczenie dostępności z MTBF i MTTR: awk

```
awk -v mtbf=1000 -v mttr=4 'BEGIN { printf "%.3f %%\n", 100*mtbf/(mtbf+mttr) }'
```

MTBF to średni czas między awariami w godzinach, a MTTR to średni czas naprawy w godzinach. Przy 1000 godzin i 4 godzinach wychodzi 99,602 procent, a przy skróceniu naprawy do 15 minut (0,25 godziny) 99,975 procent.

□□ **MTBF** = Mean Time Between Failures: średni czas pracy między awariami · **MTTR** = Mean Time To Repair: średni czas od awarii do powrotu usługi · **awk** = od nazwisk autorów Aho, Weinberger, Kernighan: język do danych w kolumnach

□□ TYLKO ODCZYT

Sprawdzenie, kiedy host był restartowany: last -x reboot

```
last -x reboot | head -n 10
```

Wypisuje ostatnie uruchomienia systemu z datą, czasem trwania poprzedniej pracy i wersją jądra. Historia pochodzi z pliku `/var/log/wtmp`, który logrotate zwykle obraca co miesiąc, więc głębokość sprawdzamy w `/etc/logrotate.conf`.

Przykład z `docker01`:

```
reboot   system boot  5.14.0-570.el9.x Fri Sep 25 15:21   still running
reboot   system boot  5.14.0-570.el9.x Tue Sep  8 06:02 - 15:19 (17+09:17)
```

Lista pokazuje restarty, a nie przerwy w działaniu usługi. Prawdziwy przestój mierzy monitoring, na przykład Zabbix, a ta komenda podpowiada, ile razy w roku planowaliśmy przerwę sami.

□□ **last** = pokaż historię logowań użytkowników do systemu

□□ TYLKO ODCZYT

Sprawdzenie świadka klastra Hyper-V: Get-ClusterQuorum

```
Get-ClusterQuorum | Format-List Cluster, QuorumResource, QuorumType
```

Pokazuje, czy klaster ma świadka (dysk, udział plikowy albo Cloud Witness). Gdy w wyniku nie ma żadnego zasobu świadka, klaster liczy głosy tylko z `hv01` i `hv02`, a Microsoft zaleca świadka zawsze [6]. Nazwy typów kworum sprawdzamy w dokumentacji Microsoftu.

Z życia wzięte

Z ŻYCIA WZIĘTE

Poniedziałek, 9:05. Pan Zbigniew przychodzi do pokoju administratorów z kartką: „Vaultwarden ma działać zawsze, a ja nie chcę słuchać o oknach serwisowych". Ola pyta, ile to jest zawsze, i dopisuje na tablicy 99,99 procent. Tomek liczy w terminalu: 52 minuty w roku, a jedno comiesięczne uruchomienie `docker01` po aktualizacji trwa 3 minuty, czyli 36 minut tylko na samych planowanych restartach.

Pan Zbigniew nie chce słuchać o oknach serwisowych, ale też nie wie, ile kosztuje każda dziewiątka więcej. Ola proponuje, żeby najpierw ustalić dwa fakty: ile minut przerwy w Vaultwardenie firma naprawdę zniesie i ile zmian w skarbcu wolno jej stracić. Z tych dwóch liczb wynika poziom rezerwy, a z poziomu koszt. Tę rozmowę prowadzimy z każdym właścicielem usługi, a ta strona jest do niej ściągą.

Jak to działa

Rozwiń: jak to działa pod spodem

Pojedynczy punkt awarii

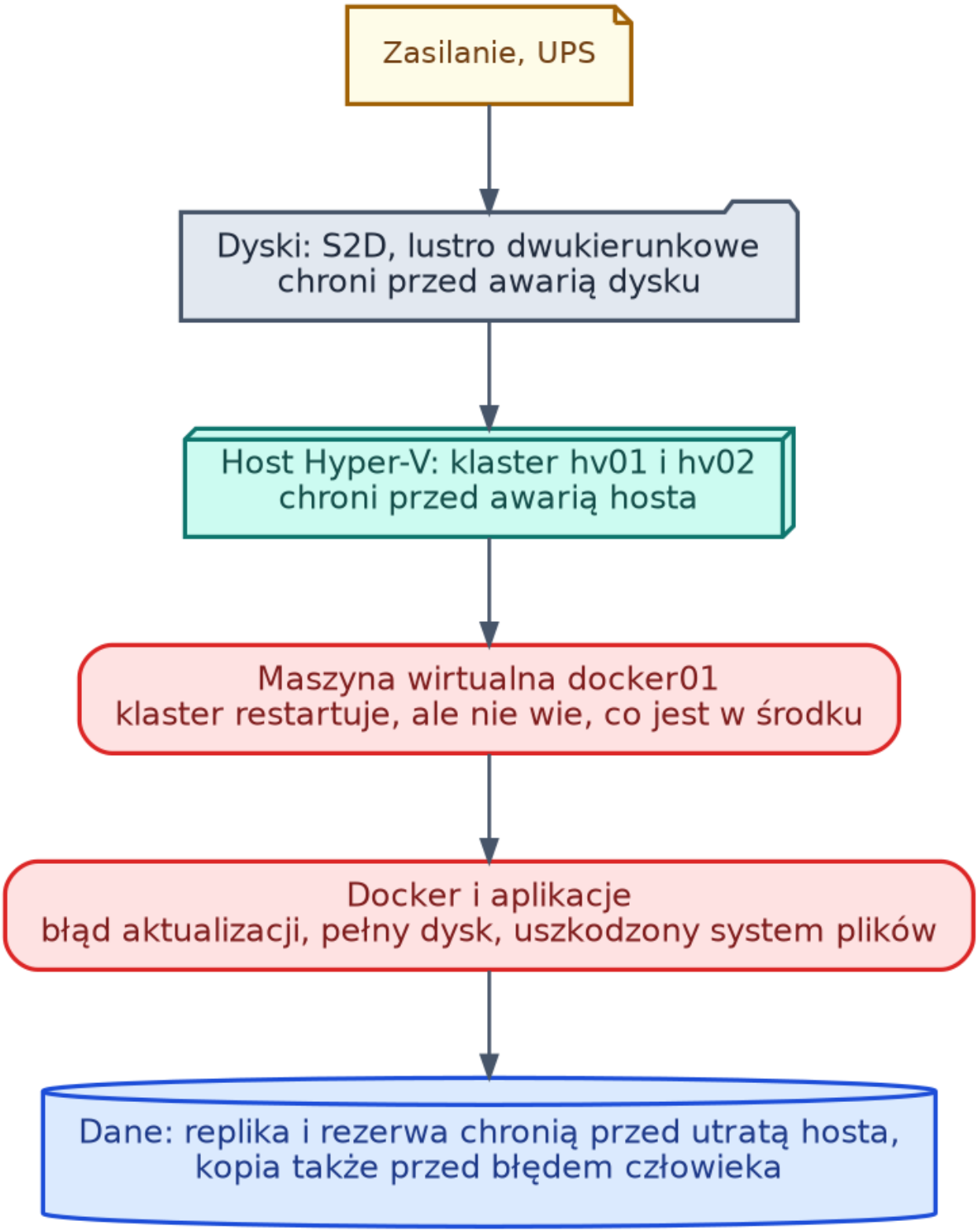
> **Pojęcie: SPOF (single point of failure).** Element, którego awaria zatrzymuje całą usługę, bo nie ma dla niego zamiennika. Może to być serwer, ale też zasilacz, kabel, przełącznik, zapora albo jeden człowiek, który zna hasło.

Wysoką dostępność zaczynamy od wypisania SPOF-ów, a dopiero potem wybieramy narzędzia [1]. W Zielonej zapora `fw01` (FortiGate 100F) jest jedna, `pg01` obsługuje Gitea, Zabbix i

aplikacje Oli, a kopie zapasowe trzyma jedna maszyna `bkp01`. Klaster Hyper-V z `hv01` i `hv02` usuwa SPOF, jakim jest pojedynczy host, ale nie usuwa wszystkich pozostałych.

Warstwy awarii

Usługa składa się z warstw, a każda ma własny mechanizm ochrony i własne słabe miejsce. Zielona ma dziś porządnie zabezpieczone warstwy dolne: UPS, dyski w S2D i hosty w klastrze. Nie ma zabezpieczenia wnętrza maszyny `docker01` ani aplikacji, a tam według Jima Graya występują najczęstsze przyczyny przestojów [2].



Rysunek 1. Warstwy awarii usługi na docker01. Klaster Hyper-V restartuje maszynę, ale nie wie, co jest w środku, więc zła aktualizacja, pełny dysk albo uszkodzony system plików nie są objęte jego ochroną. (kliknięcie otwiera pełny rozmiar)

Z rysunku wynika zasada, którą stosujemy w całej książce: poziom usługi nie może być wyższy niż poziom jej najsłabszej zależności. Gitea z gorącą rezerwą na `docker01`, ale z bazą na `pg01` bez rezerwy, ma w praktyce poziom `pg01`.

Dostępność w liczbach

> **Pojęcie: dostępność.** Ułamek czasu, w którym usługa działa tak, jak obiecaliśmy. Zapisujemy ją w procentach, a w mowie potocznej liczymy dziewiątkami: „trzy dziewiątki” to 99,9 procent.

Dziewiątki łatwo wypowiedzieć, ale trudniej wyobrazić sobie, ile to minut. Poniższa tabela liczy rok jako 365 dni, miesiąc jako 30 dni, a wyniki zaokrągla do sekundy. Wikipedia podaje 8,77 godziny dla 99,9 procent, bo liczy rok jako 365,25 doby, a różnica nie ma znaczenia dla naszych decyzji [1].

Dostępność	Rocznie	Miesięcznie	Tygodniowo
99%	3,65 doby (87 h 36 min)	7 h 12 min	1 h 40 min
99,5%	43 h 48 min	3 h 36 min	50 min
99,9%	8 h 45 min 36 s	43 min 12 s	10 min
99,95%	4 h 22 min 48 s	21 min 36 s	5 min
99,99%	52 min 34 s	4 min 19 s	1 min
99,999%	5 min 15 s	26 s	6 s

Dostępność zależy od dwóch liczb, które inżynierowie nazywają MTBF i MTTR.

> **Pojęcie: MTBF i MTTR.** MTBF (Mean Time Between Failures) to średni czas pracy między awariami. MTTR (Mean Time To Repair) to średni czas od awarii do powrotu usługi. Dostępność wynosi MTBF dzielone przez sumę MTBF i MTTR.

Przy MTBF równym 1000 godzin i MTTR równym 4 godziny dostępność wynosi 99,60 procent. Gdy MTTR skrócimy do 15 minut, wynosi 99,975 procent, a MTBF się nie zmieniło. W małej firmie taniej skrócić naprawę (gotowy host, sprawdzone kopie, skrypt przełączenia) niż wydłużyć czas między awariami.

Dwie reguły liczenia pomagają uniknąć obietnic bez pokrycia. Gdy usługa zależy od dwóch elementów po kolei, mnożymy ich dostępności: dwa razy 99,9 procent daje 99,8 procent. Gdy dwa niezależne elementy się zastępują, mnożymy ich niedostępności: dwa razy 1 procent daje

0,01 procent, czyli 99,99 procent. Ten drugi wynik jest teoretyczny, bo `docker01` i `docker02` mają ten sam system, tę samą konfigurację, ten sam klaster i ten sam administrator, więc ich awarie nie są niezależne.

RTO i RPO

> **Pojęcie: RTO (Recovery Time Objective).** Najdłuższy czas, przez jaki firma godzi się, żeby usługa nie działała po awarii. Mierzymy go od chwili awarii do chwili, gdy użytkownik znów może pracować.

> **Pojęcie: RPO (Recovery Point Objective).** Najdłuższy czas wstecz, za jaki firma godzi się stracić dane. Przy RPO równym 15 minut po awarii wolno nam stracić zmiany z ostatniego kwadransa.

Obie liczby ustala biznes, a nie administrator, bo to właściciel usługi wie, ile kosztuje minuta przerwy i godzina utraconych zapisów. Administrator mówi tylko, ile kosztuje technika, która te liczby dotrzymuje. Dla baz danych Zielonej omawiamy to na stronie RPO i RTO: dwie liczby, od których zaczyna się każdy plan backupu, a replikację samych baz w książce o PostgreSQL i pokrewnych.

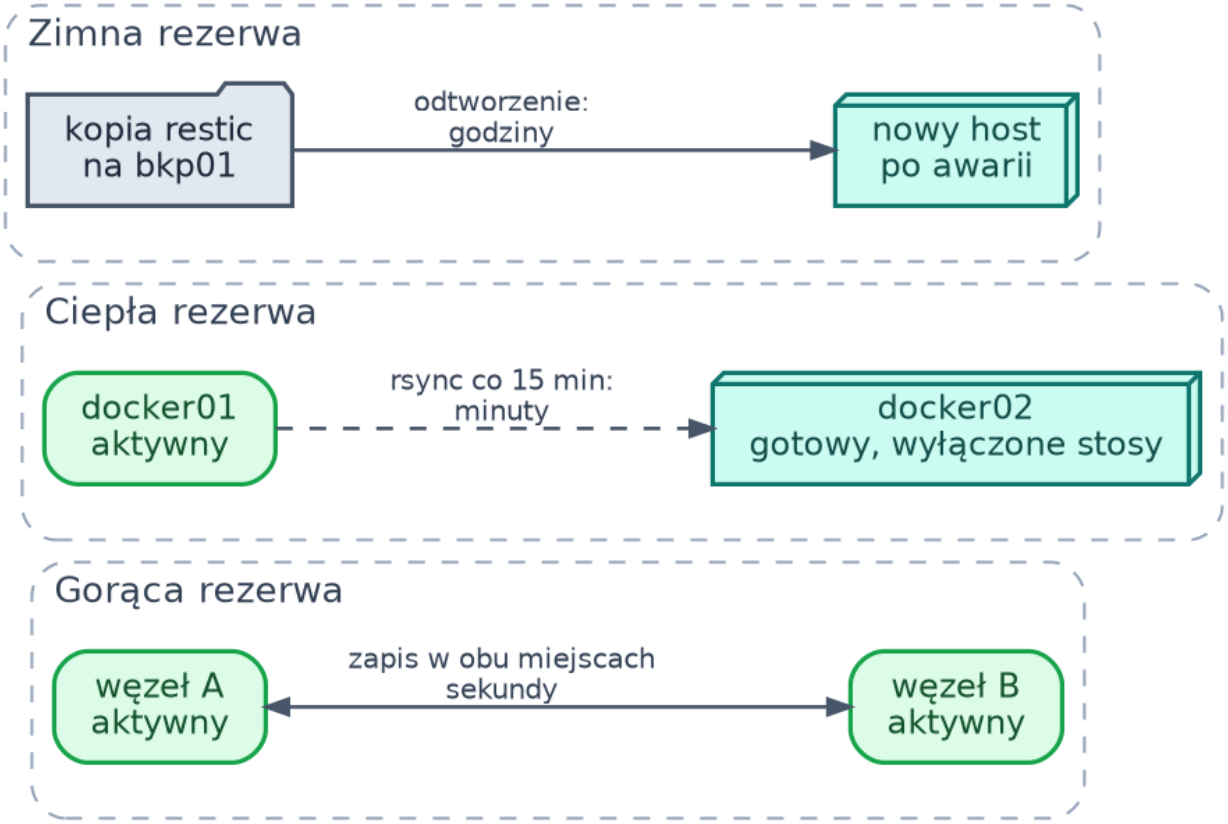
Trzy poziomy rezerwy

> **Pojęcie: zimna rezerwa.** Mamy kopię danych i przepis na usługę, ale nie mamy działającego drugiego hosta. Po awarii przygotowujemy host i odtwarzamy dane z kopii, więc czas liczymy w godzinach, a stratę danych w czasie od ostatniej kopii.

> **Pojęcie: ciepła rezerwa.** Drugi host stoi przygotowany, ma ten sam przepis (Compose, konfigurację) i dane skopiowane kilkanaście minut temu, ale usługa na nim nie działa. Po awarii włączamy ją na drugim hoście i przenosimy adres, więc czas liczymy w minutach, a stratę w minutach od ostatniej synchronizacji.

> **Pojęcie: gorąca rezerwa.** Drugi węzeł ma dane bieżące, bo zapis trafia na oba węzły jednocześnie albo replika baz podąża za głównym o sekundy. Przejęcie trwa sekundy do minut, a strata danych jest zerowa albo bliska zeru. Cena to dwa działające węzły, sieć

między nimi i mechanizm kworum.



Rysunek 2. Trzy poziomy rezerwy. Zimna rezerwa to kopia restic na bkp01 i odtworzenie po awarii. Ciepła to docker02 z danymi z rsync co 15 minut i wyłączonymi stosami. Gorąca to dwa aktywne węzły, między którymi zapis idzie w sekundach. (kliknięcie otwiera pełny rozmiar)

Poniższa tabela zbiera rzędy wielkości przyjęte w tej książce. To nie są gwarancje producentów, tylko punkt wyjścia do rozmowy, a prawdziwe liczby poznajemy z testu przełączenia.

Poziom	Typowy RTO	Typowe RPO	Koszt	Złożoność	Przykład w Zielonej
Zimna	godziny	do doby (od ostatniej kopii)	niski	niska	restic z `docker01` na `bkp01` o 02:30

Poziom	Typowy RTO	Typowe RPO	Koszt	Złożoność	Przykład w Zielonej
Ciepła	minuty do kilkunastu	kilkanaście minut	średni (drugi host)	średnia	`docker02` z rsync co 15 minut i adres wirtualny
Gorąca	sekundy do minut	zero albo sekundy	wysoki (węzły, sieć, kworum)	wysoka	replika natywna baz, klaster z trzema węzłami

Poziomy się nie wykluczają. Kopia zapasowa zostaje także tam, gdzie mamy gorącą rezerwę, bo replika powiela również pomyłkę: skasowanie danych i błąd logiczny docierają do drugiego węzła w sekundy, a w ciepłej rezerwie w kilkanaście minut.

Aktywny-pasywny i aktywny-aktywny

> **Pojęcie: aktywny-pasywny.** Jeden węzeł obsługuje ruch, drugi czeka gotowy i przejmuje po awarii pierwszego. Jest prostszy, bo w danej chwili tylko jeden zapisuje dane.

> **Pojęcie: aktywny-aktywny.** Oba węzły obsługują ruch naraz, a awaria jednego zmniejsza moc, ale nie zatrzymuje usługi. Wymaga, żeby aplikacja i dane to znosiły: wspólna baza, sesje poza pamięcią procesu, bezpieczne zapisy z dwóch miejsc.

Dla większości aplikacji z `docker01` wybieramy aktywny-pasywny. Gitea, Vaultwarden i mailcow nie są zaprojektowane do pracy dwóch instancji nad tymi samymi plikami, a jeśli ktoś to wymusi, dostaje uszkodzone dane. Aktywny-aktywny zostawiamy bazom z natywną replikacją i usługom bezstanowym.

Adres wirtualny

> **Pojęcie: VIP (adres wirtualny).** Adres IP, który nie należy do jednego serwera, tylko do roli. Trzyma go ten host, który jest aktualnie aktywny, a po awarii adres przechodzi na drugi.

Użytkownik łączy się z jednym adresem lub nazwą i nie wie, który host odpowiada. W naszym układzie adres wirtualny przenosi protokół VRRP, który opisuje RFC 5798 [3], a realizuje go

demon keepalived. Pozostałe możliwości przełączenia (zmiana wpisu DNS, przekierowanie na zaporze) porównujemy na stronie Przenoszenie kontenerów z docker01 na docker02: zimna i ciepła rezerwa, przełączenie.

Kworum, split brain i fencing

> **Pojęcie: kworum.** Zasada większości: grupa węzłów działa, jeśli widzi więcej niż połowę głosów. Dzięki temu po zerwaniu sieci tylko jedna strona ma większość i może zapisywać.

Dla etcd kworum wynosi liczbę węzłów podzieloną przez 2, plus 1, zaokrągloną w dół, a nieparzysta liczba węzłów jest zawsze lepsza od parzystej [4]. Docker Swarm stosuje to samo do menedżerów i zaleca nieparzystą liczbę, nie więcej niż siedem [5].

Liczba węzłów	Potrzebna większość	Tolerowane awarie
1	1	0
2	2	0
3	2	1
4	3	1
5	3	2

Dwa węzły nie tolerują żadnej awarii, bo utrata jednego zostawia drugiemu połowę głosów, a nie większość. Dlatego Microsoft zaleca zawsze świadka w klastrze, także dwuwęzłowym [6]. Dla Zielonej oznacza to, że klaster `hv01` i `hv02` powinien mieć świadka, a w istniejących stronach o S2D nie znaleźliśmy o nim wzmianki, więc sprawdzamy to komendą z Fiszki.

> **Pojęcie: split brain.** Dwa węzły tracą łączność, każdy uznaje się za aktywny i niezależnie zmienia dane. Po połączeniu zwykle nie da się ich automatycznie scalić, więc ktoś musi wybrać wersję i ją stracić [7].

W układzie dwóch hostów z adresem wirtualnym split brain może wystąpić, gdy zerwie się sieć między `docker01` i `docker02`, a oba widzą swoją bramę `fw01`. Oba uznają się wtedy za MASTER i oba mogą przyjmować zapisy. Zapobiegamy temu regułą kierunku synchronizacji, kontrolą, czy partner nie trzyma tego samego adresu, i skryptem sprawdzającym, który obniża priorytet hosta, gdy ten nie widzi Dockera, bramy albo Traefika. Szerzej piszemy o tym w Replikacja danych między hostami: DRBD, GlusterFS, Ceph, ZFS i replikacja w aplikacji.

> **Pojęcie: fencing (STONITH).** Odcięcie węzła, co do którego nie mamy pewności, że jest martwy, najczęściej przez wyłączenie jego zasilania, zanim ktoś inny przejmie jego zasoby. Skrót STONITH znaczy „Shoot The Other Node In The Head” [8].

Fencing jest najpewniejszą ochroną przed split brain, ale wymaga sprzętu, który potrafi wyłączyć drugi węzeł (zarządzana listwa zasilająca, karta zarządzania, API hypervisora). W Zielonej nie budujemy fencingu dla `docker01` i `docker02`, bo to dwie maszyny wirtualne, a koszt błędu jest mniejszy niż koszt układu. Zamiast tego opieramy się na kontrolach w skryptach i na tym, że człowiek decyduje o powrocie starego hosta.

Pytania do właściciela usługi

Poziom rezerwy dobieramy z rozmowy z właścicielem usługi, a nie z własnych przekonań. Poniżej osiem pytań, które zadajemy po kolei, i cel każdego z nich.

Pytanie	Po co pytamy
Kto korzysta z usługi i kiedy jej potrzebuje najbardziej?	ustalamy godziny, w których przerwa boli (zamknięcie miesiąca, start zmiany o 6:00)
Co się dzieje w firmie, gdy usługa stoi godzinę, cztery godziny, dobę?	zamieniamy odczucie na koszt, a z tego wynika RTO
Ile pracy z ostatniej godziny da się odtworzyć ręcznie?	z tego wynika RPO
Czy ktoś może pracować bez usługi, jeśli ma obejście (papier, plik, inna aplikacja)?	tanie obejście bywa lepsze niż drugi serwer
Czy planowane przerwy (aktualizacje, wymiana sprzętu) wliczają się do przestoju?	przestoje planowane mogą zjeść cały roczny budżet
Od czego usługa zależy: baza, DNS, uwierzytelnianie, zapora, internet?	poziom usługi nie przekroczy poziomu zależności
Kto podejmuje decyzję o przełączeniu, jeśli nie ma automatu, i czy ktoś odbierze telefon w piątek o 20:00?	najlepszy mechanizm jest bezużyteczny bez osoby, która go uruchomi
Ile firma jest gotowa zapłacić za każdą kolejną dziewiątkę?	rozmowa o pieniądzach zamyka rozmowę o technice

Odpowiedzi zapisujemy na stronie usługi, a w przebiegu z macierzą HA zbieramy je w jedną tabelę. Dziś mamy tylko wstępne propozycje: zimną rezerwę dla NetBoksa, ciepłą dla Gitea, Vaultwardena i poczty, gorącą (replika natywna) dla Active Directory, a dla `pg01` ciepłą lub gorącą po rozmowie z autorem. Wszystkie wymagają potwierdzenia przez właścicieli usług.

Częste błędy

Błąd	Skutek	Jak unikamy
Mówimy „dostępność 99,9” bez podania, czy planowane przerwy się liczą	po roku okazuje się, że 8 godzin zjadły aktualizacje, a nie awarie	w umowie i w liczbach zapisujemy, czy okna serwisowe są wyłączone
Liczymy dostępność dwóch hostów jako 1 minus kwadrat niedostępności	obiecujemy 99,99 procent, a obie maszyny psują się od tej samej aktualizacji	przyjmujemy, że awarie nie są niezależne, i mierzymy w teście
Poziom rezerwy usługi wyższy niż jej bazy	Gitea ma gorącą rezerwę, ale `pg01` jest jeden	rozpisujemy łańcuch zależności i bierzemy najsłabsze ogniwo
Replika zamiast kopii zapasowej	skasowanie danych dociera do rezerwy w sekundy	kopia restic zostaje na każdym poziomie rezerwy
Dwa węzły bez trzeciego głosu i wiara w automatyczne przełączenie	przy zerwanej sieci oba uznają się za aktywne	trzeci głos (świadek, tiebreaker) albo reguły w skryptach i decyzja człowieka
Rezerwa, której nikt nie testował	w dniu awarii wychodzi brak obrazu kontenera, reguły zapory albo wpisu w bazie	test przełączenia w kalendarzu, na przykład raz na kwartał
Tylko jedna osoba umie przełączyć usługę	awaria w czasie urlopu kończy się czekaniem	instrukcja na stronie, drugi administrator przeszkolony i test z jego udziałem

□□ Ćwiczenia

1. Ile minut w miesiącu dopuszcza dostępność 99,95 procent i czy wystarczy, jeśli aktualizacja `docker01` trwa 3 minuty i robimy ją co miesiąc?

Odpowiedź

Miesięcznie (30 dni) wychodzi 21 minut 36 sekund, więc 3-minutowy restart mieści się z zapasem, o ile awarii nie ma. W rocznym rachunku 12 restartów to 36 minut z 4 godzin 22 minut rocznego budżetu, czyli około 14 procent. Pozostałe niecałe 4 godziny muszą pokryć wszystkie awarie, więc sama aktualizacja nie jest problemem, ale każda godzina awarii jest.

1. Usługa działa na `docker01` z dostępnością 99,9 procent i zależy od `pg01`, który ma 99,5 procent. Jaka jest łączna dostępność?

Odpowiedź

Przy zależności szeregowej mnożymy: 0,999 razy 0,995 daje około 0,994, czyli 99,4 procent, co odpowiada około 52 godzinom przestoju w roku. Usługa jest więc słabsza od swojej bazy, a nie

od własnego hosta. Poprawa hosta nie pomoże, dopóki nie poprawimy `pg01`.

1. Dlaczego klaster z dwoma węzłami i bez świadka nie toleruje żadnej awarii?

Odpowiedź

Większość z dwóch głosów to dwa głosy, a po utracie jednego węzła drugi ma tylko jeden głos, czyli połowę. Połowa nie jest większością, więc węzeł nie wie, czy to on jest odcięty, czy partner nie żyje, i musi się zatrzymać (albo ryzykować split brain). Trzeci głos, na przykład świadek w postaci udziału plikowego lub Cloud Witness, zamienia sytuację w 2 z 3 i pozwala węzłowi, który widzi świadka, działać dalej.

1. Właściciel mówi: „RPO zero, RTO pięć minut, ale budżet na jeden serwer”. Co mu odpowiadamy?

Odpowiedź

Pokazujemy, że RPO zero wymaga zapisu w dwóch miejscach naraz, czyli drugiego węzła i sieci, a RTO pięć minut wymaga drugiego hosta z gotowym przepisem. Przy jednym serwerze możemy zaoferować zimną rezerwę z kopii (RTO w godzinach, RPO do doby) i podać, ile kosztuje przejście na ciepłą (drugi host) albo gorącą rezerwę (replikacja i węzły). Decyzja zostaje po stronie właściciela, ale podejmuje ją, znając cenę.

Powiązane: Wysoka dostępność: usługa, która nie ma jednego serwera · Przenoszenie kontenerów z docker01 na docker02: zimna i ciepła rezerwa, przełączenie · Replikacja danych między hostami: DRBD, GlusterFS, Ceph, ZFS i replikacja w aplikacji · RPO i RTO: dwie liczby, od których zaczyna się każdy plan backupu · Hyper-V na co dzień: przełącznik, maszyny, punkty kontrolne i migracja · S2D: wymiana dysku bez przestoju, gdy pula jest prawie pełna

Źródła:

1. en.wikipedia.org: Single point of failure i High availability
2. Jim Gray, Why Do Computers Stop and What Can Be Done About It?, Tandem Technical Report 85.7, czerwiec 1985
3. IETF: RFC 5798, Virtual Router Redundancy Protocol (VRRP) Version 3
4. etcd.io: FAQ, dokumentacja wersji 3.6
5. docs.docker.com: Administer and maintain a swarm of Docker Engines
6. learn.microsoft.com: Manage the cluster quorum

7. en.wikipedia.org: Split-brain (computing))
8. clusterlabs.org: Pacemaker Explained, Fencing

Wzorce projektowania wysokiej dostępności: jak dobrać sposób do rodzaju usługi

Poziom: średnio

Dotyczy: Hyper-V Failover Cluster, Windows Failover Cluster, Pacemaker i Corosync, Proxmox VE HA, keepalived, HAProxy, Docker Swarm, k3s, magazyny S2D, Ceph, DRBD i ZFS

Czas: ~50 min

Ta strona uczy dobierać sposób utrzymania wysokiej dostępności do rodzaju usługi, a nie do konkretnej aplikacji. Opisujemy pięć typów usług, siedem wzorców i zasady, które obowiązują w każdym z nich. Wracamy tu, gdy ktoś mówi „zróbmy klaster” i trzeba odpowiedzieć, jaki i po co.

ZAŁOŻENIA SCENARIUSZA

przyjmujemy jedną serwerownię w Pamiątkowie z dwoma hostami `hv01` i `hv02`, pojedynczym `fw01` i jednym fizycznym `bkp01`. Klaster hostów jest na początku bez świadka (luka), a świadek ma stać poza nim: na `bkp01` albo w chmurze.

Skróty użyte na tej stronie (20)

Skrót	Co znaczy
B2B	business to business: sprzedaż i systemy skierowane do innych firm, nie do klientów indywidualnych
CNAME	Canonical Name: rekord DNS będący aliasem, drugą nazwą wskazującą na już istniejący rekord A
Corosync	usługa komunikacji klastra, która pilnuje członkostwa węzłów i kworum; Proxmox VE i Pacemaker używają jej do sprawdzania, które węzły żyją
DR	Disaster Recovery: plan przywrócenia działania po poważnej awarii

Skrót	Co znaczy
DRBD	Distributed Replicated Block Device: replikowane urządzenie blokowe w jądrze Linuksa, w którym każdy zapis trafia na dwa węzły
HA	High Availability: wysoka dostępność, czyli działanie mimo awarii jednego elementu
HAProxy	High Availability Proxy: program, który rozdziela połączenia między kilka serwerów i omija te, które nie odpowiadają
IETF	Internet Engineering Task Force: organizacja publikująca standardy internetowe jako dokumenty RFC
IP	Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)
RPO	Recovery Point Objective: ile danych wolno stracić przy awarii
RTO	Recovery Time Objective: jak długo wolno nie działać przy awarii
S2D	Storage Spaces Direct: programowa macierz Windows Server z dysków wielu serwerów
SPOF	Single Point of Failure: element, którego awaria zatrzymuje całą usługę, bo nie ma dla niego zamiennika
TTL	Time To Live: czas życia rekordu lub danych
UPS	Uninterruptible Power Supply: zasilacz awaryjny, podtrzymuje zasilanie urządzeń na czas zaniku napięcia z sieci
VIP	Virtual IP: adres wirtualny, który należy do roli, a nie do jednego serwera, i przechodzi na drugi host, gdy pierwszy przestaje odpowiadać
VRRP	Virtual Router Redundancy Protocol: protokół, w którym kilka hostów uzgadnia, który z nich trzyma wspólny adres IP (opisany w RFC 5798)
WMS	Warehouse Management System: system zarządzania magazynem
WWW	World Wide Web: strony i aplikacje otwierane w przeglądarce po HTTP albo HTTPS
ZFS	system plików z migawkami i replikacją przyrostową zbudowany w jednym narzędziu z menedżerem wolumenów (dawniej Zettabyte File System)

Z życia wzięte

Z ŻYCIA WZIĘTE

Czwartek, 9:30. Pan Zbigniew pyta, czy klaster, który ma chronić serwer wersji plików zespołu Oli, ochroni też WMS Marka i serwer WWW sklepu B2B. Tomek odpowiada, że nie, bo każda z tych usług jest inna: serwer wersji plików trzyma dane w jednym miejscu i pisze do nich jeden proces, baza WMS ma własną replikację, a serwer WWW nie trzyma niczego, co by

się zmieniało. Ola rysuje na tablicy trzy różne strzałki i dopisuje, że wspólna jest tylko jedna rzecz: każdy z tych układów trzeba sprawdzić testem przełączenia. Pan Zbigniew prosi o tabelę, z której będzie widać, co wybieramy dla jakiej usługi i co zobaczą użytkownicy.

Jak to działa

Rozwiń: jak to działa pod spodem

Pięć typów usług

> **Pojęcie: usługa stanowa i bezstanowa.** Usługa stanowa trzyma dane, które się zmieniają i które muszą przeżyć awarię: pliki, bazę, kolejkę. Usługa bezstanowa nie trzyma niczego ważnego we własnym procesie, więc dowolną jej kopię można wyłączyć i zastąpić inną.

> **Pojęcie: jeden zapisujący.** Zasada, że w danej chwili do tych samych danych pisze jeden proces na jednym węźle. Wiele systemów nie znosi dwóch piszących naraz (zwykły system plików, baza bez replikacji), a próba obejścia tego kończy się uszkodzeniem danych.

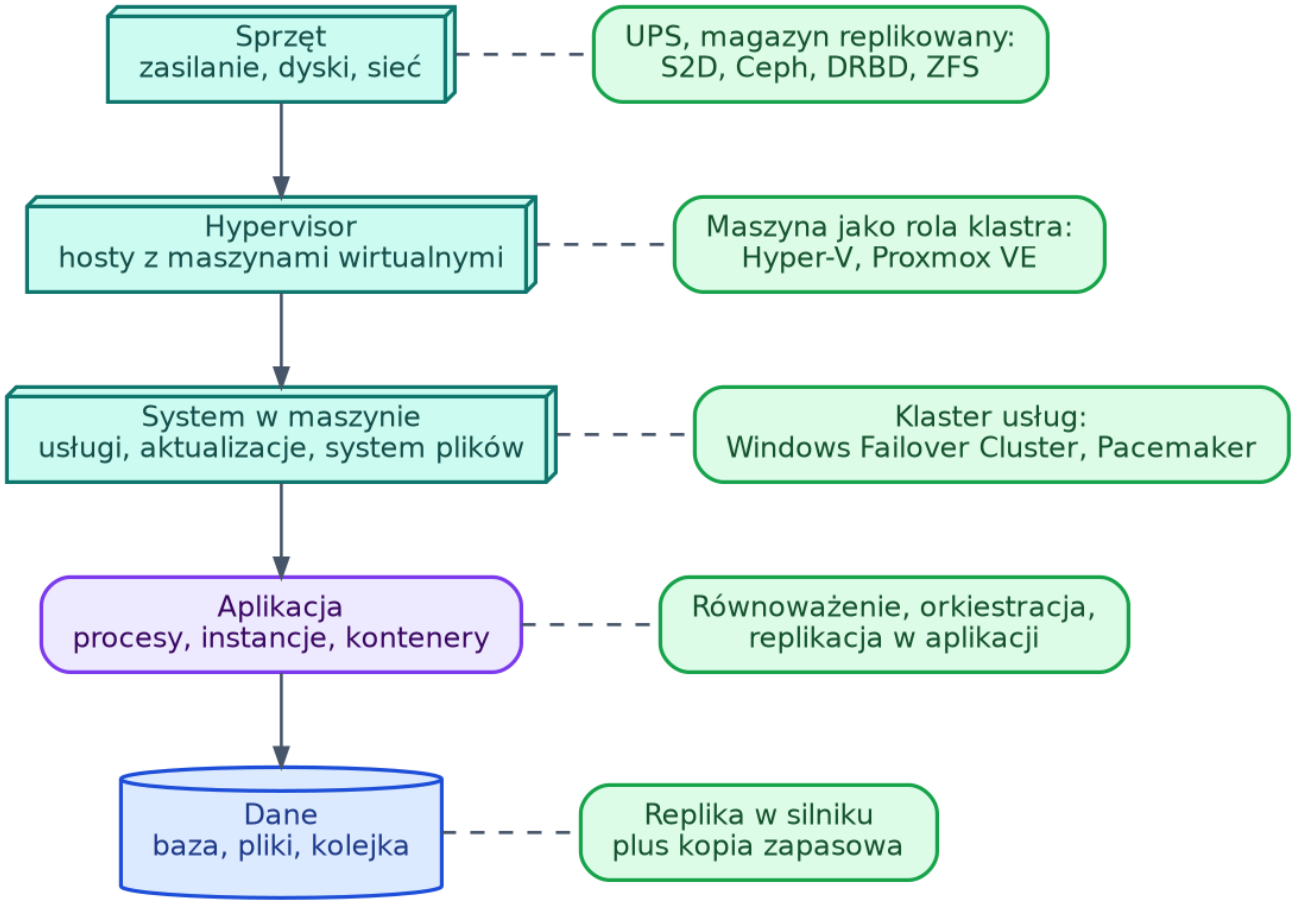
Zielona ma pięć typów usług i każdy ma inny naturalny sposób ochrony:

- stanowa z jednym zapisującym: serwer wersji plików, poczta, baza bez replikacji; dane leżą w jednym miejscu i otwiera je jeden proces,
- baza z replikacją: silnik sam kopiuje zmiany na drugi węzeł (PostgreSQL, MongoDB, Active Directory),
- bezstanowa usługa za równoważeniem: kilka identycznych instancji, dane są gdzie indziej (w bazie),
- usługa z kolejką: instancje pobierają zadania z kolejki, a stan trzyma kolejka,
- plik współdzielony: udział plikowy, z którego korzysta wielu użytkowników.

Poziom rezerwy usługi (zimna, ciepła, gorąca) i liczby RTO i RPO tłumaczy strona Wysoka dostępność: pojęcia, liczenie dostępności i poziomy rezerwy. Tu zakładamy, że czytelnik je zna, i pokazujemy, jakim wzorcem te poziomy osiągamy.

Warstwy: kto chroni co

Usługa składa się z warstw, a każdy wzorzec chroni inną. Wzorzec z niższej warstwy nie widzi, co dzieje się wyżej: klaster hostów restartuje maszynę, ale nie wie, że aplikacja w niej przestała odpowiadać.



Rysunek 1. Warstwy usługi i wzorce, które je chronią. Sprzęt chroni UPS i magazyn replikowany, hypervisor maszyna jako rola klastra, system w maszynie klaster usług, aplikację równoważenie, orkiestracja i replikacja w aplikacji, a dane replika w silniku i kopia zapasowa. (kliknięcie otwiera pełny rozmiar)

Warstwa	Co ją chroni	Czego nie chroni
Sprzęt: zasilanie, dyski, sieć	UPS, magazyn replikowany (S2D, Ceph, DRBD, ZFS)	błędów w warstwach wyżej, skasowania danych
Hypervisor	maszyna jako rola klastra (Hyper-V, Proxmox VE)	zawieszonego systemu, wadliwej aktualizacji w maszynie

Warstwa	Co ją chroni	Czego nie chroni
System w maszynie	klaster usług (Windows Failover Cluster, Pacemaker)	uszkodzonych danych na wspólnym dysku
Aplikacja	równoważenie, orkiestracja, replikacja w aplikacji	błędu logicznego powielonego na rezerwę
Dane	replika w silniku bazy, kopia zapasowa	replika nie cofa skasowania ani ransomware, to potrafi tylko kopia

Z tabeli wynika zasada z pierwszej strony książki: poziom usługi nie jest wyższy niż poziom jej najsłabszej zależności. Wzorce łączymy, a nie zastępujemy jednego drugim: klaster hostów pod spodem, a nad nim to, co chroni wnętrze.

Maszyna wirtualna jako rola klastra

Hypervisor traktuje maszynę jak czarną skrzynkę. Gdy host pada, uruchamia ją na innym hoście, a przy pracach planowych przenosi ją w migracji na żywo [12]. Hyper-V Failover Cluster i Proxmox VE HA robią to samo, a inne hypervisory (oVirt, Oracle Linux Virtualization Manager, XCP-ng) mają własne odpowiedniki, które opisuje osobna strona w przygotowaniu.

Po awarii hosta system w maszynie startuje od zera, jakby ktoś wyrwał wtyczkę, więc użytkownik widzi przerwę rzędu minut, a połączenia i operacje w toku giną. Dokumentacja Proxmox VE podaje typowy czas wykrycia awarii i przełączenia około 2 minut i wymaga co najmniej trzech węzłów, żeby kworum było wiarygodne, oraz wspólnego magazynu dla maszyn [2]. Hyper-V wymaga klastra hostów ze świadkiem [4].

To najprostszy wzorzec i dlatego jest naszym wyborem domyślnym dla usług stanowych z jednym zapisującym. Nie leczy za to niczego, co dzieje się wewnątrz maszyny: zła aktualizacja, pełny dysk i uszkodzony system plików przechodzą razem z maszyną. Budowę klastra Hyper-V opisuje strona Hyper-V na co dzień: przełącznik, maszyny, punkty kontrolne i migracja, a przenoszenie całego hosta z kontenerami strona Przenoszenie kontenerów z docker01 na docker02: zimna i ciepła rezerwa, przełączenie.

Klaster usług

Klaster usług pilnuje nie maszyny, tylko usługi w jej wnętrzu. Zasoby (adres, nazwa, dysk, usługa) tworzą rolę albo grupę, którą klaster uruchamia na jednym węźle naraz, a po awarii na drugim [1]. W Windows Failover Cluster własną usługę dodajemy jako Generic Service, Generic Application albo Generic Script, a w Pacemakerze z Corosync jako zasób z agentem, na przykład jednostką systemd.

Dane leżą na dysku wspólnym dla obu węzłów, ale otwartym tylko na jednym. Przełączenie to zatrzymanie i start, a nie migracja na żywo, więc nawet planowa zmiana węzła kończy się krótką przerwą, za to po awarii odpada rozruch systemu, bo drugi węzeł już działa. W Pacemakerze fencing jest obowiązkowy: klaster bez niego kiedyś uszkodzi dane [3].

Własnych ról nie dodajemy na hostach z hiperkonwergentnym S2D, bo Microsoft kieruje takie usługi do maszyn wirtualnych [8]. Zostaje klaster gościnny z dwóch maszyn. Pojęcia roli, zasobu i zależności opisuje strona Klaster Windows dla własnych usług: rola, zasób, zależności, kworum i testy przełączenia, a odpowiednik linuksowy strona Pacemaker na Linuksie: własna usługa z adresem wirtualnym.

Replikacja w aplikacji

Aplikacja, która sama kopiuje swoje dane na drugi węzeł (replika standby, repliki baz, zestawy replik, replikacja Active Directory), wie, które zapisy są potwierdzone. Dlatego nie ma problemu niespójnego obrazu pliku, a każdy węzeł ma własne dane, więc nie potrzebujemy wspólnego dysku ani zwykle fencingu. To jedyny wzorzec, w którym mamy dwie niezależne kopie danych.

Planowe przełączenie (awans repliki z udziałem starego serwera) zwykle nie traci niczego. Po awarii czas zależy od tego, kto awansuje replikę: człowiek, który najpierw upewnia się, że stary węzeł nie żyje, albo narzędzie z kworum lub arbitrem. Przy replikacji asynchronicznej tracimy zmiany, które nie zdążyły dotrzeć, a przy synchronicznej płacimy opóźnieniem każdego zapisu. Szczegóły i porównanie technik jest na stronie Replikacja danych między hostami: DRBD, GlusterFS, Ceph, ZFS i replikacja w aplikacji, a replikację baz opisują strony Replikacja: synchroniczna, asynchroniczna, logiczna, fizyczna i Replikacja strumieniowa.

Bezstanowe instancje za równoważeniem

Kilka identycznych instancji stoi za wspólnym adresem. Adres wirtualny (VIP) przenosi keepalived protokołem VRRP [11], a HAProxy rozdziela ruch i kontrolami stanu wyłącza instancje, które nie odpowiadają. Przy pracach planowych instancję wyłączamy z rotacji i użytkownik niczego nie widzi, a po awarii kontrola stanu potrzebuje kilku do kilkunastu sekund, w zależności od ustawień, zanim ruch przestanie iść do martwej instancji.

Wzorzec działa tylko dla usług bez własnych danych albo z danymi w zewnętrznej bazie. Sama para maszyn równoważących ma własne ryzyko: gdy zerwie się łączność między nimi, obie mogą trzymać ten sam adres. Odpowiedź na to jest na stronie

Równoważenie i adres wirtualny: keepalived i HAProxy na parze lb01 i lb02.

Orkiestracja kontenerów

Docker Swarm i Kubernetes (u nas k3s) pozwalają zadeklarować, ile kopii usługi ma działać. Po awarii węzła zadania startują na pozostałych, a aktualizacje idą kroczące, po jednej kopii. Węzły sterujące potrzebują kworum, więc zalecana jest nieparzysta liczba, co najmniej trzy [9][10], a trzy węzły na dwóch hostach nie chronią przed awarią hosta, który trzyma dwa z nich.

Orkiestracja nie rozwiązuje problemu danych: wolumen lokalny nie podąża za zadaniem, więc aplikacje stanowe potrzebują magazynu replikowanego albo bazy poza klastrem. Strony o wzorcu to Docker Swarm: klaster z trzema menedżerami, na którym przenosimy kolorowy nginx i k3s z trzema węzłami serwerowymi: etcd, adres wirtualny API i Longhorn. Czytelnik, który nie zna samych kontenerów, zaczyna od Docker: kontenery, które zmieniły serwerownie i Docker Compose, a Kubernetesa poznaje przez Kubernetes: sternik, który przypłynął z Google i Architektura klastra: płaszczyzna sterowania i węzły.

Magazyn współdzielony lub replikowany

S2D, Ceph, DRBD i ZFS nie są osobnym wzorcem dla usługi, tylko warstwą pod wzorcami powyżej: dają dane, które przeżywają utratę dysku albo węzła. Różnią się tym, co kopiują (bloki, obiekty, migawki) i ile węzłów wymagają: Ceph co najmniej trzech, DRBD trzeciego głosu dla kworum, a replikacja ZFS przez migawki jest asynchroniczna. Porównanie jest na stronie Replikacja danych między hostami: DRBD, GlusterFS, Ceph, ZFS i replikacja w aplikacji, a S2D opisuje strona S2D: wymiana dysku bez przestoju, gdy pula jest prawie pełna.

Magazyn replikowany chroni przed utratą sprzętu, ale replikuje także skasowanie i uszkodzenie zawartości. Jedna kopia danych na wspólnym dysku, choćby zdublowanym lustrem, nie daje więc drugiej niezależnej kopii.

HA a odtwarzanie po katastrofie

> **Pojęcie: DR (disaster recovery, odtwarzanie po katastrofie).** Odtworzenie usługi po utracie całego miejsca (pożar, zalanie), zwykle w innej lokalizacji i z ręcznym startem. Wysoka dostępność chroni przed utratą węzła, DR przed utratą miejsca, a kopia zapasowa przed utratą danych.

Hyper-V Replica kopiuje maszynę asynchronicznie co 30 sekund, 5 minut albo 15 minut, a przełączenie na replikę inicjuje człowiek, dlatego Microsoft nazywa ją technologią DR [5]. Storage Replica replikuje wolumeny synchronicznie albo asynchronicznie [6]. Obie dają drugą kopię w innym miejscu, a żadna nie jest ani wysoką dostępnością, ani kopią zapasową. Zielona nie ma dziś drugiego miejsca, więc to decyzja o sprzęcie i łączu, nie o samej technice.

Wybór wzorca

Tabela zestawia typ usługi z wzorcem i pokazuje, co zobaczą użytkownicy. Czasy to rzędy wielkości do zmierzenia testem, nie obietnice.

Typ usługi	Zalecany wzorzec	Planowe wyłączenie	Awaria
Stanowa z jednym zapisującym (domyślnie)	maszyna jako rola klastra (Hyper-V, Proxmox VE)	migracja na żywo: niewidoczne albo pauza rzędu sekund	maszyna startuje od zera na innym hoście: minuty, połączenia zerwane, operacje do powtórzenia
Stanowa z jednym zapisującym, gdy przerwa po awarii ma być krótsza	klaster usług (Windows, Pacemaker)	przeniesienie roli z krótką przerwą	rola startuje na działającym węźle: dziesiątki sekund do minuty, połączenia zerwane
Stanowa z własną repliką (standby)	replikacja w aplikacji, maszyna w klastrze pod spodem	kontrolowane przełączenie: krótka przerwa, bez utraty danych	czas decyzji plus przełączenie, możliwa utrata niezreplikowanych zmian
Baza z replikacją	replikacja natywna plus arbiter lub świadek	przełączenie planowe: krótka przerwa, bez utraty danych	awans repliki: sekundy do minut zależnie od narzędzia, przy replikacji asynchronicznej możliwa utrata
Bezstanowa za równoważeniem	keepalived i HAProxy	niewidoczne: instancja wychodzi z rotacji	kilka do kilkunastu sekund, połączenia w toku zerwane, klient ponawia
Bezstanowa w kontenerach	orkiestracja (Swarm, k3s)	opróżnienie węzła i aktualizacja krocząca: niewidoczne	zadania startują na innych węzłach: sekundy do minut
Usługa z kolejką	konsumenci za orkiestracją albo równoważeniem, broker jak usługa stanowa	konsument kończy zadanie, nic nie ginie	komunikaty czekają w kolejce, rośnie opóźnienie, przerwane zadanie wraca i bywa wykonane drugi raz
Plik współdzielony	serwer plików w maszynie klastra na magazynie replikowanym	jak w pierwszym wierszu	otwarte pliki zamykają się, użytkownik otwiera je ponownie, niezapisane zmiany giną

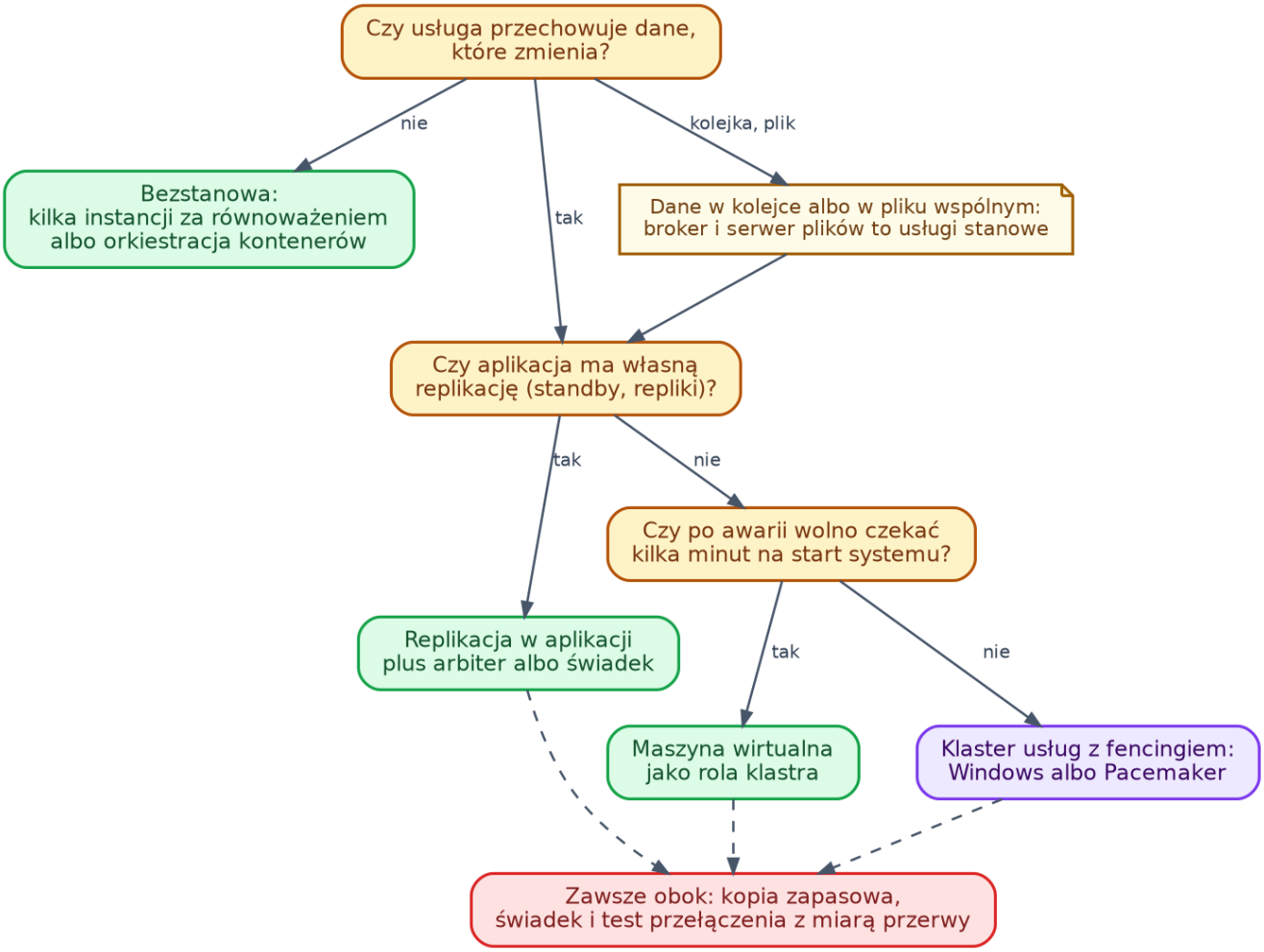
Typ usługi	Zalecany wzorzec	Planowe wyłączenie	Awaria
Dowolna, druga lokalizacja	DR: Hyper-V Replica, Storage Replica, kopie	nic się nie zmienia, replika nie przejmuje pracy	ręczny start repliki: godziny, dane sprzed 30 sekund do 15 minut (Hyper-V Replica)

Druga tabela mówi, czego wzorzec wymaga, żeby w ogóle zadziałał, i gdzie jest o nim strona w książce.

Wzorzec	Kworum i świadek	Fencing	Liczba węzłów	Strona
Maszyna jako rola klastra	świadek poza klastrem hostów; Proxmox VE co najmniej trzy węzły [2]	zajmuje się nim klaster hostów; w Proxmox VE potrzebny watchdog [2]	dwa hosty ze świadkiem albo trzy	Klaster Windows dla własnych usług: rola, zasób, zależności, kworum i testy przełączenia (kworum), Przenoszenie kontenerów z docker01 na docker02: zimna i ciepła rezerwa, przełączenie
Klaster usług	świadek (Windows) albo urządzenie kworum na trzecim hoście (Pacemaker)	Windows: rezerwacja dysku klastrowego, Pacemaker: obowiązkowy	dwa węzły plus trzeci głos	Klaster Windows dla własnych usług: rola, zasób, zależności, kworum i testy przełączenia, Pacemaker na Linuksie: własna usługa z adresem wirtualnym
Replikacja w aplikacji	arbiter lub trzeci głos dla automatycznego awansu, przy ręcznym decyduje człowiek	zwykle zbędny, bo każdy węzeł ma własne dane	dwa węzły, dla automatu trzy głosy	Replikacja danych między hostami: DRBD, GlusterFS, Ceph, ZFS i replikacja w aplikacji
Równoważenie	brak kworum danych, ryzykiem są dwa węzły z tym samym adresem	nie dotyczy	dwie instancje i dwie maszyny równoważące	Równoważenie i adres wirtualny: keepalived i HAProxy na parze lb01 i lb02

Wzorzec	Kworum i świadek	Fencing	Liczba węzłów	Strona
Orkiestracja	nieparzysta liczba węzłów sterujących, trzy tolerują utratę jednego [9][10]	nie dotyczy	trzy węzły sterujące, najlepiej na trzech hostach	Docker Swarm: klaster z trzema menedżerami, na którym przenosimy kolorowy nginx, k3s z trzema węzłami serwerowymi: etcd, adres wirtualny API i Longhorn
Magazyn replikowany	zależnie od techniki: Ceph trzy monitory, DRBD trzeci głos	zależnie od techniki	Ceph trzy węzły, DRBD dwa plus węzeł głosujący	Replikacja danych między hostami: DRBD, GlusterFS, Ceph, ZFS i replikacja w aplikacji
DR	nie dotyczy	nie dotyczy	drugi host w drugim miejscu	Perforce i Lore: porównanie dróg do wysokiej dostępności na Windowsie (dodatek)

Wzorca nie dobieramy według mody, tylko według odpowiedzi na kilka pytań. Drzewo pokazuje ich kolejność.



Rysunek 2. Drzewo wyboru wzorca. Pytamy najpierw, czy usługa przechowuje dane, potem, czy aplikacja ma własną replikację, na końcu, czy po awarii wolno czekać kilka minut. Kopia zapasowa, świadek i test przełączenia towarzyszą każdej odpowiedzi. (kliknięcie otwiera pełny rozmiar)

Budowę i utrzymanie poszczególnych wzorców opisują strony: Klaster Windows od zera: budowa, sieć, świadek i S2D; Klaster Windows w utrzymaniu: aktualizacje CAU, monitoring i problemy; Pacemaker w utrzymaniu: zarządzanie, testy i aktualizacje; Proxmox VE: klaster trzech węzłów i HA ; Proxmox VE w utrzymaniu: kopie PBS, aktualizacje, użytkownicy; Proxmox VE a Hyper-V z Failover Cluster: porównanie i przenosiny; Inne hypervisory z HA: oVirt, Oracle Linux Virtualization Manager i XCP-ng; Porównanie mechanizmów HA: Windows, Linux i hypervisory.

> Przykład: serwer wersji plików (Perforce, Lore). Serwer wersji dużych plików jest typem pierwszym: stanową usługą z jednym zapisującym, bo metadane i pliki wersjonowane otwiera jeden proces naraz. Zielona zaczyna od najprostszego wzorca, czyli maszyny wirtualnej jako roli klastra Hyper-V, a krótsza przerwa po awarii wymaga klastra usług w klastrze gościnnym. Perforce ma dodatkowo replikę standby, czyli replikację w aplikacji, a

> serwer zapasowy wymaga osobnej licencji [7]. Lore jest przed wersją 1.0 i nie ma opisanej lokalnej replikacji, więc zostaje maszyna jako rola klastra. Szczegóły, procedury i testy są w dodatku: Perforce i Lore w klastrze: cztery drogi, rekomendacja dla Zielonej i test przełączenia (Windows), Perforce i Lore: porównanie dróg do wysokiej dostępności na Windowsie i Perforce i Lore na Linuksie: odpowiedniki dróg z Windowsa, pułapka wielkości liter i rekomendacja dla Zielonej.

Zasady przekrojowe

Pojęcia SPOF, kworum, split brain i fencing tłumaczy strona Wysoka dostępność: pojęcia, liczenie dostępności i poziomy rezerwy. Poniższe zasady wynikają z nich i obowiązują niezależnie od wzorca.

Planowe wyłączenie a awaria

To dwa różne zdarzenia i dwie osobne obietnice dla właściciela usługi. Przy wyłączeniu planowym system zdąży przenieść pracę: migracja na żywo kopiuje pamięć działającej maszyny, a klaster usług zatrzymuje usługę czysto, więc użytkownik widzi najwyżej pauzę. Po awarii nic nie zdąży się zamknąć: drugi węzeł startuje z danych na dysku, a połączenia i operacje w toku giną.

Dlatego tę samą infrastrukturę użytkownicy widzą raz jako niewidoczną, raz jako przerwę. Mówimy to wprost: planowo niewidoczne, po awarii przerwa o zmierzonej długości. Dla klastra gościnnego na Hyper-V sprawdzamy też, czy pauza migracji nie wyrzuci węzła z klastra, bo próg sygnałów kontrolnych bywa krótki.

Nieczyste zatrzymanie i spójność danych

Po awarii usługa startuje tak, jakby ktoś wyrwał wtyczkę. Baza albo pliki zapisywane w chwili awarii mogą być niespójne, więc usługa musi umieć wstać po nagłym zatrzymaniu (dziennik transakcji, odtwarzanie). Tam, gdzie nie umie, dokładamy kontrolę przed startem i odtwarzanie z kopii, a nie wierzymy, że klaster sam to załatwi.

Z tego samego powodu nie kopiujemy plików żywej bazy zwykłym kopiowaniem. Robimy zrzut, checkpoint albo migawkę, które dają spójny obraz. Test sprawdza nie tylko, czy usługa wstała, ale też czy dane są w porządku.

Jedna kopia danych to nie wysoka dostępność

Maszyna jako rola klastra i klaster usług ze wspólnym dyskiem mają jeden egzemplarz danych. Lustro dysków (na przykład S2D) chroni przed utratą dysku fizycznego, ale nie przed uszkodzeniem zawartości plików ani skasowaniem danych przez człowieka. Drugą niezależną kopię daje dopiero replikacja w aplikacji, a cofnięcie błędu daje wyłącznie kopia zapasowa.

Split brain i fencing

Zanim drugi węzeł przejmie dane otwarte do zapisu, mamy pewność, że pierwszy już nie pisze. Przy wspólnym dysku zapewnia to fencing, czyli odcięcie węzła o nieznanym stanie, przy replikach kworum albo arbiter, a przy ręcznym przełączeniu człowiek, który sprawdza, że stary węzeł naprawdę nie żyje. Wymuszone przełączenie z opcją „ignoruj partnera” wolno wykonać tylko wtedy, gdy partner na pewno nie wróci.

Stary węzeł po awarii nie wraca do roli aktywnej sam: wraca jako rezerwa, a decyzję o powrocie podejmuje człowiek. Pomagają w tym ustawienia, po których wracający węzeł nie odbiera roli (w keepalived opcja nopreempt), i wyłączenie usługi z rozruchu na starym węźle. Rysunek i szczegóły split brain są na stronie Replikacja danych między hostami: DRBD, GlusterFS, Ceph, ZFS i replikacja w aplikacji.

Świadek przy dwóch węzłach

Klaster z dwóch węzłów bez trzeciego głosu nie toleruje żadnej awarii, a Microsoft zaleca świadka zawsze, także dla klastra dwuwęzłowego [4]. Świadek ma stać fizycznie osobno od węzłów i służyć jednemu klastrowi [4]. Nie może leżeć na maszynie, która sama stoi na tym klastrze, bo znika razem z hostem.

W Zielonej wszystkie serwery Windows to maszyny na `hv01` i `hv02`, więc udział na `fs01` albo `dc01` odpada. Zostaje fizyczny `bkp01` albo Cloud Witness, a w Pacemakerze urządzenie kworum na trzecim hoście. Brak świadka w klastrze `hv01` i `hv02` jest luką, którą zamykamy przed jakąkolwiek obietnicą niewidocznego przełączenia.

Alias DNS zamiast nazwy hosta

Klienci łączą się z nazwą usługi, nie z nazwą maszyny. Po wymianie serwera zmieniamy cel aliasu, a nie ustawienia na każdej stacji. Gdy mamy adres wirtualny albo punkt dostępu klastra, nazwa wskazuje właśnie jego i niczego nie trzeba zmieniać.

Jeśli przełączenie robimy zmianą rekordu, krótki TTL (na przykład minuta) skraca czas, po którym klienci zobaczą zmianę. Rekord CNAME i rekord A o tej samej nazwie nie mogą współistnieć, więc przy zmianie wzorca najpierw usuwamy stary. W konfiguracji aplikacji piszemy nazwę usługi, nigdy nazwę hosta ani adresu IP, bo odnośniki zostaną wtedy przy martwym hoście.

Test przełączenia z miarą przerwy

Obietnica niewidoczności bez pomiaru jest obietnicą bez pokrycia. Test robimy poza zamknięciem miesiąca i poza zakazem zmian od 15 listopada do 6 stycznia, z komputera spoza klastra, pętlą, która co sekundę pyta usługę i zapisuje wynik z godziną. Pytamy nie tylko o port, ale prawdziwym zapytaniem do usługi, bo port bywa otwarty, gdy usługa już nie odpowiada.

Przerwą jest czas od pierwszego błędu do pierwszego sukcesu. Zapisujemy też czas do pierwszej udanej operacji zapisu i sprawdzamy dane po teście. Deklarowany czas przełączenia opieramy na najgorszym wyniku z kilku prób, nie na średniej.

Testujemy też awarie, nie tylko przeniesienie planowe: zabicie procesu, odcięcie sieci węzła, wyłączenie maszyny i awarię hosta. Gotowe pętle są na stronach Klaster Windows dla własnych usług: rola, zasób, zależności, kworum i testy przełączenia (PowerShell) i Pacemaker na Linuksie: własna usługa z adresem wirtualnym (bash).

Replikacja to nie kopia zapasowa

Replika odzwierciedla bieżący stan źródła, więc razem z danymi przenosi skasowanie, ransomware i błąd logiczny. Kopia zapasowa trzyma stan z przeszłości, do którego można wrócić, i zostaje na każdym poziomie rezerwy. Kopii, której nie próbowaliśmy odtworzyć, nie liczymy: odtwarzamy ją na maszynie pomocniczej co kwartał i zapisujemy czas. O kopiach hosta z kontenerami piszemy w Kopia całego docker01: /docker/apps, /docker/compose i bazy w kontenerach.

Częste błędy

Błąd	Skutek	Jak unikamy
Jeden wzorzec dla wszystkich usług, bo „klaster już mamy”	klaster hostów nie widzi wnętrza maszyny, a baza i serwer WWW mają inne potrzeby	dobieramy wzorzec do typu usługi według tabeli i drzewa
Klaster usług z danymi na wspólnym dysku bez fencingu	po zawieszeniu węzła oba mogą pisać na ten sam dysk i uszkodzić dane	fencing obowiązkowy albo wzorzec bez wspólnego dysku

Błąd	Skutek	Jak unikamy
Dwa węzły bez świadka i wiara w automatyczne przełączenie	po zerwaniu łączności żaden nie ma większości albo oba uznają się za aktywne	świadek poza klastrem, nie na maszynie z tego klastra
Replika zamiast kopii zapasowej	skasowanie dociera do rezerwy w sekundy	kopia zapasowa na każdym poziomie rezerwy
Trzy węzły sterujące na dwóch hostach uznane za odporne na awarię hosta	host z dwoma węzłami zabiera większość głosów	trzy hosty albo świadome przyjęcie ryzyka
W aplikacji wpisana nazwa hosta zamiast aliasu	po przełączeniu odnośniki wskazują martwy host	nazwa usługi i alias DNS, krótki TTL
Planowe przełączenie przetestowane, awaria nie	w dniu awarii wychodzi nieczysty start, brak obrazu, reguła zapory	test planowy i awaryjny z miarą przerwy w kalendarzu
Stary węzeł wraca do roli aktywnej sam	dwa węzły z różnymi danymi	powrót jako rezerwa, decyzja człowieka

□□ Ćwiczenia

1. Dlaczego dla serwera, który trzyma dane w jednym miejscu i pisze do nich jeden proces, domyślnie wybieramy maszynę jako rolę klastra, a nie klaster usług?

Odpowiedź

Maszyna jako rola klastra jest najprostsza: nie dokłada dysku wspólnego, fencingu ani drugiego systemu do aktualizacji, a przy pracach planowych przechodzi w migracji na żywo. Cena to przerwa rzędu minut po awarii hosta, bo system startuje od zera. Klaster usług skraca tę przerwę, ale kosztuje drugą maszynę, wspólny dysk i fencing, więc sięgamy po niego dopiero, gdy test pokaże przerwę dłuższą od dopuszczalnej.

1. Klaster z dwóch węzłów ma świadka na udziale na maszynie wirtualnej, która stoi na tym samym klastrze. Co się stanie po awarii hosta z tą maszyną i dlaczego?

Odpowiedź

Razem z hostem znika węzeł i świadek, więc zostaje jeden głos z trzech i klaster staje. Świadek ma stać poza klastrem: na fizycznym serwerze, który nie zależy od jego hostów, albo w chmurze.

1. Dlaczego replika w aplikacji nie zastępuje kopii zapasowej, choć daje drugą niezależną kopię danych?

Odpowiedź

Replika powiela zmiany ze źródła, więc skasowanie, ransomware i błąd logiczny docierają do niej w sekundy albo minuty. Jest niezależna pod względem sprzętu, ale nie pod względem treści. Kopia zapasowa trzyma stan z przeszłości i tylko ona pozwala do niego wrócić.

1. Pan Zbigniew pyta, czy po przełączeniu użytkownik zauważy przerwę. Co odpowiadamy dla maszyny jako roli klastra i skąd bierzemy liczbę?

Odpowiedź

Przy wyłączeniu planowym hosta zwykle niczego, najwyżej pauzę migracji na żywo. Po awarii hosta przerwę rzędu minut, bo system startuje od zera, a połączenia i operacje w toku giną. Liczbę podajemy dopiero z testu przełączenia: pętla co sekundę z komputera spoza klastra, najgorszy wynik z kilku prób.

1. Czym różni się Hyper-V Replica od maszyny jako roli klastra Hyper-V i gdzie wolno jej użyć?

Odpowiedź

Rola klastra to wysoka dostępność: maszyna startuje na drugim hoście tego samego klastra, na wspólnym magazynie. Hyper-V Replica to odtwarzanie po katastrofie: asynchroniczna kopia na innym hoście (co 30 sekund, 5 lub 15 minut), a start repliki inicjuje człowiek. Ma sens do drugiej lokalizacji, nie między hostami z tym samym magazynem.

Powiązane: Wysoka dostępność: usługa, która nie ma jednego serwera · Wysoka dostępność: pojęcia, liczenie dostępności i poziomy rezerwy · Przenoszenie kontenerów z docker01 na docker02: zimna i ciepła rezerwa, przełączenie · Replikacja danych między hostami: DRBD, GlusterFS, Ceph, ZFS i replikacja w aplikacji · Klaster Windows dla własnych usług: rola, zasób, zależności, kworum i testy przełączenia · Pacemaker na Linuksie: własna usługa z adresem wirtualnym · Równoważenie i adres wirtualny: keepalived i HAProxy na parze lb01 i lb02 · Docker Swarm: klaster z trzema menedżerami, na którym przenosimy kolorowy nginx · k3s z trzema węzłami serwerowymi: etcd, adres wirtualny API i Longhorn · Macierz wysokiej dostępności usług Zielonej: poziom rezerwy każdej usługi, zależności i kolejność prac · Docker: kontenery, które zmieniły serwerownie · Kubernetes: sternik, który przypłynął z Google · Architektura klastra: płaszczyzna sterowania i węzły

Źródła: (odczyt 3 października 2026)

1. learn.microsoft.com: Failover clustering overview
2. pve.proxmox.com: High Availability (co najmniej trzy węzły, wspólny magazyn, watchdog, typowy czas wykrycia i przełączenia około 2 minut)
3. clusterlabs.org: Pacemaker Explained, Fencing
4. learn.microsoft.com: Manage the cluster quorum (świadek zawsze, osobno od węzłów, tylko dla jednego klastra)
5. learn.microsoft.com: Hyper-V Replica overview
6. learn.microsoft.com: Storage Replica overview
7. help.perforce.com: General guidelines for multi-server deployments (każdy standby i replika, które mogą przejąć rolę, mają własny plik licencji)
8. learn.microsoft.com: Plan volumes, Storage Spaces Direct (ogólne usługi w maszynach wirtualnych)
9. docs.docker.com: Administer and maintain a swarm of Docker Engines
10. docs.k3s.io: High Availability Embedded etcd
11. IETF: RFC 5798, Virtual Router Redundancy Protocol (VRRP) Version 3
12. learn.microsoft.com: Suspend-ClusterNode