Pierwsze kroki

Co to serwer, Linux a Windows, pierwsze połączenie z serwerem.

- Co to jest serwer i czym różni się od laptopa
- Linux a Windows: te same zadania, dwa światy
- Połączenie z serwerem z Windowsa: SSH, Windows Terminal i PuTTY

Co to jest serwer i czym różni się od laptopa

Poziom: podstawy **Dotyczy: podstaw infrastruktury** **Czas: ~15 min**

Ta strona tłumaczy, czym właściwie jest serwer, zanim zobaczymy pierwszą komendę. Poznajemy różnicę między sprzętem a maszyną wirtualną, sens dwóch dysków w RAID i to, dlaczego serwer w ogóle nie ma monitora. Wracamy tu, gdy zapomnimy, co dokładnie znaczy nazwa, adres IP albo port.

Skróty użyte na tej stronie (9)

Skrót	Co znaczy
DNS	Domain Name System: książka telefoniczna internetu, zamienia nazwy na adresy IP
IP	Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)
MSSQL	Microsoft SQL Server: silnik bazy danych Microsoftu, na nim w Zielonej stoi księgowość Optima
RAID	Redundant Array of Independent Disks: sposób łączenia kilku fizycznych dysków w jeden, widziany przez system jako pojedynczy dysk, zwykle z dodatkową odpornością na awarię jednego z nich
RDP	Remote Desktop Protocol: protokół pulpitu zdalnego Windows, uruchamiany poleceniem mstsc
SSH	Secure Shell: szyfrowane zdalne logowanie do serwera
VLAN	Virtual LAN: wydzielona logicznie sieć na tym samym sprzęcie
VPN	Virtual Private Network: szyfrowany tunel do sieci firmowej
WWW	World Wide Web: strony i aplikacje otwierane w przeglądarce po HTTP albo HTTPS

Serwer to komputer, który obsługuje innych

> **Pojęcie: serwer.** Serwer to zwykły komputer, tyle że jego zadaniem jest odpowiadać na prośby innych komputerów, a nie służyć jednej osobie przed ekranem. Laptop pani Haliny czeka, aż ona coś kliknie. Serwer `sql01` czeka cały czas na pytania z sieci: podaj fakturę numer 4521, zapisz nową pozycję w magazynie, sprawdź hasło użytkownika jan.nowak.

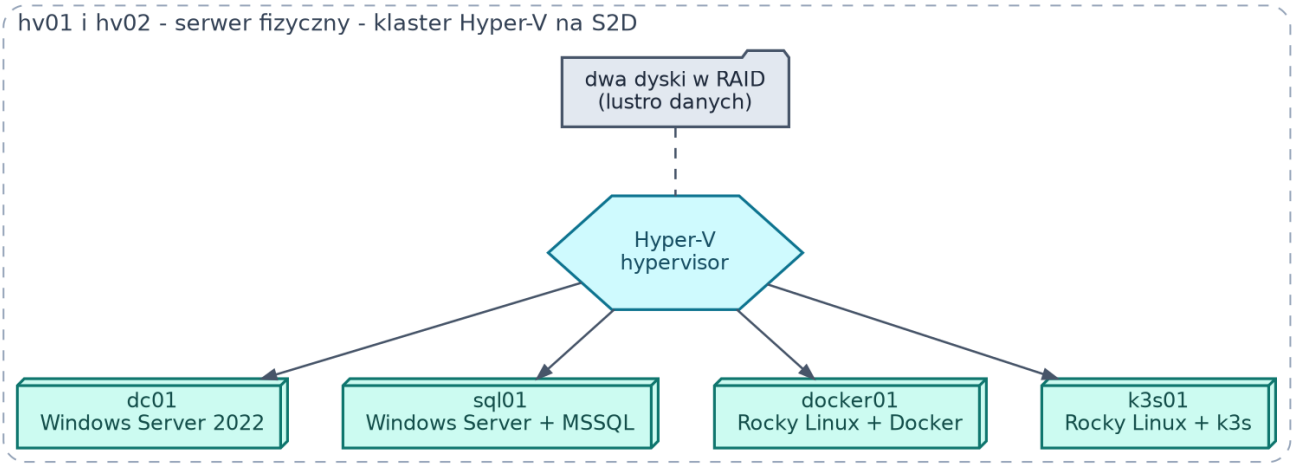
W tym rozróżnieniu nie chodzi o wygląd skrzynki. To, że komputer stoi w szafie w serwerowni, samo z siebie nie czyni go serwerem, tak samo jak biurko nie czyni kogoś księgową. Serwerem czyni go rola: obsługuje żądania od innych, zwykle wielu naraz, zwykle bez przerwy.

W Zielonej rolę tę pełni kilkanaście maszyn. `dc01` loguje ludzi do domeny, `sql01` trzyma księgowość, `docker01` trzyma Gitea i kilka innych aplikacji. Każda z nich robi jedną rzecz dla wielu osób naraz, dwadzieścia cztery godziny na dobę.

Sprzęt kontra maszyna wirtualna

Coraz rzadziej jeden serwer fizyczny odpowiada za jedną rolę. W Zielonej stoją dwa potężne serwery fizyczne, `hv01` i `hv02`, a na nich żyje prawie cała reszta jako maszyny wirtualne.

> **Pojęcie: maszyna wirtualna (VM).** Maszyna wirtualna to udawany komputer, który działa w środku prawdziwego. Program zwany hypervisorem, w Zielonej to Hyper-V, dzieli procesor, pamięć i dyski fizycznego serwera na kawałki i z każdego kawałka robi osobny, w pełni sprawny komputer, ze swoim systemem operacyjnym i adresem IP.



Serwer fizyczny hv01 i hv02 dzieli się na maszyny wirtualne przez Hyper-V (kliknięcie otwiera pełny rozmiar)

Dzięki temu jeden potężny serwer fizyczny udaje kilkanaście mniejszych. `dc01`, `sql01`, `docker01` i `k3s01` to osobne systemy operacyjne, każdy z nich zachowuje się tak, jakby miał własny komputer, choć fizycznie mieszkają razem na `hv01` albo `hv02`. Gdy potrzebny jest nowy serwer, nie kupujemy nowej skrzynki, tylko zakładamy kolejną maszynę wirtualną.

To nie znaczy, że sprzęt fizyczny znika z rozmowy. Gdy `hv01` traci prąd albo pada mu dysk, tracimy naraz wszystkie maszyny wirtualne, które na nim mieszkały. Dlatego `hv01` i `hv02` pracują razem jako klaster: jeśli jeden z nich odmówi posłuszeństwa, jego maszyny wirtualne wstają na drugim.

Dlaczego serwer ma dwa dyski i RAID

Laptop ma zwykle jeden dysk. Gdy on się psuje, tracimy dane i szukamy kopii zapasowej. Serwer nie może sobie na to pozwolić, bo nie obsługuje jednej osoby, tylko całą firmę.

> **Pojęcie: RAID.** RAID to sposób łączenia kilku fizycznych dysków tak, żeby razem zachowywały się jak jeden, ale przetrwały awarię pojedynczego z nich. Najprostszy wariant, który spotykamy najczęściej, to RAID 1, czyli lustro: dwa dyski trzymają dokładnie te same dane. Kiedy jeden z nich umiera, drugi pracuje dalej bez żadnej przerwy, a zepsuty dysk po prostu wymieniamy.

Lustro nie jest kopią zapasową. Jeśli ktoś przypadkiem skasuje plik, RAID skasuje go na obu dyskach naraz, bo obie kopie są identyczne w każdej chwili. RAID chroni przed awarią sprzętu, a kopia zapasowa przed pomyłką człowieka i przed awarią całego serwera. Potrzebujemy obu naraz, nie jednego zamiast drugiego.

Dlaczego serwer nie ma monitora

Serwery w serwerowni Zielonej stoją bez podłączonego ekranu i klawiatury. To nie oszczędność, tylko konsekwencja tego, jak z nich korzystamy.

Administrator nie siedzi przed serwerem tak jak przed laptopem. Łączymy się z nim zdalnie, przez sieć, poleceniem SSH na Linuksie albo pulpitem zdalnym RDP na Windowsie, z własnego biurka albo z domu. Monitor i klawiatura przy samym serwerze przydają się właściwie tylko w awarii, gdy sieć akurat nie działa, dlatego serwerownia ma jeden wspólny ekran na wózku, podłączany w razie potrzeby do dowolnej maszyny.

System operacyjny, usługa, port, adres IP i nazwa DNS po ludzku

Zanim zaczniemy cokolwiek konfigurować, warto rozumieć pięć słów, które wracają na niemal każdej stronie tej bazy.

> **Pojęcie: system operacyjny.** To program, który zarządza samym komputerem: włącza dysk i kartę sieciową, pozwala uruchamiać inne programy. W Zielonej spotykamy dwa systemy: Windows Server na maszynach księgowości i domeny, oraz Rocky Linux na serwerach programistów i narzędzi.

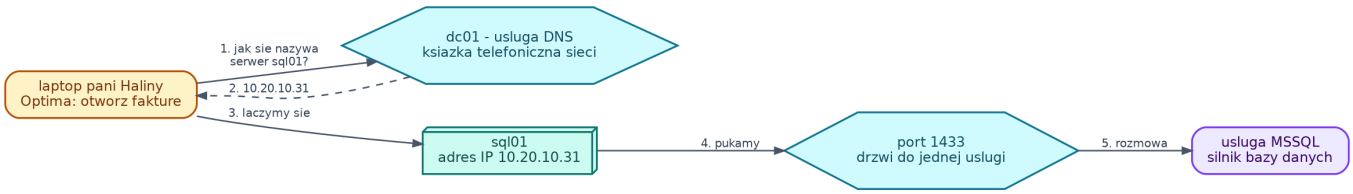
> **Pojęcie: usługa.** Usługa to program, który działa w tle serwera i czeka na żądania, bez okna i bez człowieka przed ekranem. `sshd` czeka na próby połączenia po SSH, MSSQL czeka na zapytania Optimy, DNS czeka na pytania o nazwy. Usługę można zatrzymać i uruchomić niezależnie od reszty systemu.

> **Pojęcie: adres IP.** Adres IP to numer, pod którym komputer jest widoczny w sieci, podobny do numeru domu. Serwer `sql01` ma adres `10.20.10.31`. Bez tego numeru żaden inny komputer nie wie, dokąd wysłać dane.

> **Pojęcie: nazwa DNS.** Ludzie zapamiętują nazwy lepiej niż numery, więc zamiast wpisywać `10.20.10.31` za każdym razem, wpisujemy `sql01`. Usługa DNS na `dc01` działa jak książka telefoniczna sieci: dostaje nazwę i odpowiada adresem IP.

> **Pojęcie: port.** Jeden serwer potrafi prowadzić wiele usług naraz, więc sama nazwa i adres IP nie wystarczą, żeby trafić do właściwej z nich. Port to numer drzwi w tym samym budynku: MSSQL nasłuchuje na porcie `1433`, SSH na porcie `22`, strona WWW zwykle na porcie `80` albo `443`.

Cała droga od kliknięcia do odpowiedzi wygląda tak samo za każdym razem, niezależnie od aplikacji.

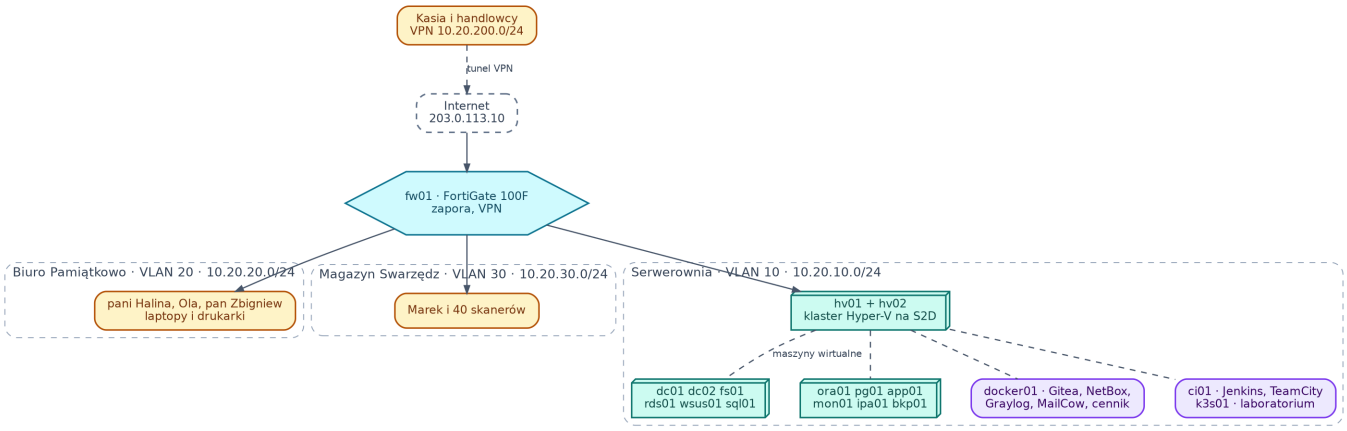


Laptop pani Haliny pyta o fakturę: od nazwy serwera przez DNS, adres IP i port aż do usługi MSSQL (kliknięcie otwiera pełny rozmiar)

Optima na laptopie pani Haliny zna tylko nazwę `sql01`. Zanim wyśle jedną porcję danych, system operacyjny laptopa pyta DNS o adres, łączy się na ten adres, puka do konkretnego portu i dopiero wtedy usługa MSSQL zaczyna rozmowę. Cały ten łańcuch dzieje się w ułamku sekundy, ale każde jego ogniwo może się zepsuć osobno, i wtedy szukamy, które.

Serwerownia Zielonej

Wszystkie te serwery stoją fizycznie w jednym miejscu, w serwerowni w centrali w Pamiątkowie, podłączone do tej samej sieci VLAN 10.



Mapa sieci i serwerów Zielonej: biuro, magazyn, serwerownia i handlowcy przez VPN (kliknięcie otwiera pełny rozmiar)

Biuro w Pamiątkowie i magazyn w Swarzędzu łączą się z serwerownią przez sieć firmową, a handlowcy w terenie przez VPN. Z punktu widzenia laptopa pani Haliny nie ma to znaczenia: jej Optima i tak pyta tylko o nazwę `sql01`, a resztą drogi zajmuje się sieć.

Z życia wzięte

Z ŻYCIA WZIĘTE

Tomek, drugi tydzień w pracy, dostaje pierwsze samodzielne zadanie: sprawdź, czy na docker01 jest miejsce na nową aplikację. W głowie ma obraz laptopa, bo z takich komputerów korzystał do tej pory, więc pyta, gdzie jest ekran do tego serwera. Odpowiadamy, że łączymy się zdalnie przez SSH, a monitor w serwerowni jest jeden, na wózku, na wypadek awarii sieci. Dwa tygodnie później Tomek sam tłumaczy to następnej nowej osobie.

Częste pomyłki

Myślimy	W rzeczywistości
Serwer to po prostu mocniejszy komputer	Serwerem czyni go rola: obsługuje wielu naraz i bez przerwy, moc jest tylko konsekwencją tej roli
Maszyna wirtualna to coś słabszego niż prawdziwy serwer	VM to pełny system operacyjny, ograniczony tylko tym, ile zasobów przydzielił mu hypervisor
RAID, czyli lustro, zastępuje kopię zapasową	RAID chroni przed awarią dysku, nie przed skasowanym plikiem ani przed awarią całego serwera
Serwer bez monitora jest zepsuty albo źle skonfigurowany	Brak monitora jest celowy, administrujemy zdalnie przez SSH albo RDP

Ćwiczenia

1. Laptop programistki Oli i serwer `docker01` mają swoje adresy IP. Do czego w takim razie służy nazwa DNS, skoro adres IP już wystarczy do połączenia?

Odpowiedź

Nazwa DNS jest dla nas, ludzi, żeby nie trzeba było pamiętać ciągów cyfr takich jak `10.20.10.42`. Komputer i tak w tle zamienia nazwę na adres IP przez usługę DNS, zanim się połączy.

1. `hv01` pada w środku nocy z powodu awarii zasilania. Co dzieje się z maszynami wirtualnymi, które na nim mieszkały, i dlaczego w Zielonej stoi obok niego `hv02`?

Odpowiedź

Maszyny wirtualne z `hv01` tracą zasilanie razem z nim, bo fizycznie mieszkały w jego procesorze i pamięci. `hv02` istnieje właśnie po to, żeby klaster Hyper-V mógł podnieść te same maszyny wirtualne na drugim serwerze fizycznym.

1. Optima na laptopie pani Haliny łączy się z `sql01` na porcie `1433`. Czym różni się port od adresu IP?

Odpowiedź

Adres IP wskazuje, do którego komputera w sieci mamy trafić, tak jak numer domu. Port wskazuje, do której usługi na tym komputerze mamy trafić, tak jak numer mieszkania w tym samym budynku.

Linux a Windows: te same zadania, dwa światy

Poziom: podstawy **Dotyczy: Windows Server 2022, Rocky Linux 9** **Czas: ~25 min**

Ta strona pokazuje te same codzienne czynności administratora wykonane najpierw w PowerShellu, potem w bashu. Nie uczymy się tu nowego systemu od zera, tylko przekładamy to, co już umiemy z Windows, na słownictwo Linuksa. Wracamy tu za każdym razem, gdy szukamy linuksowego odpowiednika znanej komendy Windows.

Skróty użyte na tej stronie (5)

Skrót	Co znaczy
CMD	skrót od polecenia command, dawna nazwa wiersza poleceń Windows sprzed PowerShella
EL	Enterprise Linux: rodzina RHEL i jej klonów (Rocky, Alma, Oracle Linux)
IP	Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)
PID	Process ID: numer procesu w systemie
SSH	Secure Shell: szyfrowane zdalne logowanie do serwera

Fiszka

TYLKO ODCZYT

Sprawdzenie folderu, w którym jesteśmy, i listy plików: Get-ChildItem oraz ls

```
Get-Location
Get-ChildItem
```

```
pwd
ls -la
```

Obie komendy najpierw pokazują pełną ścieżkę bieżącego folderu, a potem listę plików i podfolderów w nim. Flaga `-la` w bashu pokazuje też pliki ukryte i dodatkowe informacje o prawach dostępu.

Przykład:

```
PS C:\Users\tomek> Get-Location

Path
----
C:\Users\tomek

[tomek@docker01 ~]$ pwd
/home/tomek
[tomek@docker01 ~]$ ls -la
-rw-r--r-- 1 tomek tomek 512 27 wrz 10:15 notatki.txt
```

□□ **pwd** = print working directory: pokaż, gdzie jestem · **ls** = list: wypisz zawartość katalogu

□□ **TYLKO ODCZYT**

Wyświetlenie zawartości pliku: Get-Content oraz cat

```
Get-Content C:\Users\tomek\notatki.txt
```

```
cat /home/tomek/notatki.txt
```

Obie komendy wypisują cały plik tekstowy na ekran, od pierwszej do ostatniej linii. Dla bardzo długich plików na Linuksie wygodniejszy bywa `less`, który pozwala przewijać tekst zamiast wypisywać go od razu w całości.

Przykład:

```
PS C:\Users\tomek> Get-Content C:\Users\tomek\notatki.txt
Backup dc02 zaplanowany na 22:00

[tomek@docker01 ~]$ cat /home/tomek/notatki.txt
kontener gitea restartowany 26.09, dziala poprawnie
```

□□ **cat** = concatenate: sklej i wypisz pliki

□□ TYLKO ODCZYT

Szukanie tekstu w pliku: Select-String oraz grep

```
Select-String -Path C:\logs\app.log -Pattern "blad"
```

```
grep "blad" /var/log/app.log
```

Obie komendy przeszukują wskazany plik linia po linii i wypisują tylko te linie, w których pojawia się szukany tekst. Dopasowanie działa też na kilku plikach naraz, wystarczy podać więcej ścieżek albo wzorzec z gwiazdką.

Przykład:

```
PS C:\> Select-String -Path C:\logs\app.log -Pattern "blad"


app.log:214:14:32:05 blad polaczenia z baza CDN_Zielona


[tomek@docker01 ~]$ grep "blad" /var/log/app.log
14:32:05 blad polaczenia z kontenerem gitea
```

□□ **grep** = global regular expression print: szukaj wzorca w tekście

□□ TYLKO ODCZYT

Sprawdzenie działających procesów: Get-Process oraz ps

```
Get-Process
```

```
ps aux
```

Obie komendy pokazują listę wszystkich uruchomionych programów razem z ich numerem procesu i zużyciem pamięci. `Get-Process` domyślnie sortuje po nazwie, `ps aux` po numerze PID, ale obie da się posortować inaczej.

Przykład:

```
PS C:\> Get-Process | Select-Object -First 2
```

```
Handles  NPM(K)  PM(K) Name
-------  ------  ----- ----
    412      21  45200 lsass
    198      12  18400 dns

[tomek@docker01 ~]$ ps aux | head -2
root  812  0.1  1.2 812400 45000 ?  Ssl  10:12  0:03 dockerd
tomek 2104 0.0  0.1  12800  3200 pts/0 R+  10:20  0:00 ps
```

PID = Process ID: numer procesu w systemie · **ps** = process status: pokaż procesy

TYLKO ODCZYT

Sprawdzenie stanu usługi: Get-Service oraz systemctl status

```
Get-Service -Name Spooler
```

```
systemctl status sshd
```

Obie komendy mówią, czy dana usługa jest uruchomiona, zatrzymana albo w ogóle nie istnieje na tym systemie. `systemctl status` dodatkowo pokazuje kilka ostatnich linii dziennika tej usługi, czego `Get-Service` nie robi.

Przykład:

```
PS C:\> Get-Service -Name Spooler

Status   Name     DisplayName
------   ----     -----------
Running  Spooler  Print Spooler

[tomek@docker01 ~]$ systemctl status sshd
● sshd.service - OpenSSH server daemon
     Active: active (running) since Sun 2026-09-27 08:02:11 CEST
```

systemctl = system control: sterowanie usługami systemd

TYLKO ODCZYT

Przeglądanie dziennika zdarzeń: Get-WinEvent oraz journalctl

```
Get-WinEvent -LogName System -MaxEvents 5
```

```
journalctl -n 5
```

Obie komendy pokazują najnowsze wpisy z dziennika systemowego, czyli zapis tego, co ostatnio działo się z systemem i usługami.

> **Pojęcie: systemd.** Systemd to program w Linuksie, który uruchamia usługi, pilnuje ich i prowadzi ich wspólny dziennik. `journalctl` czyta właśnie ten dziennik, tak jak Podgląd zdarzeń czyta dziennik Windows.

Przykład:

```
PS C:\> Get-WinEvent -LogName System -MaxEvents 2

TimeCreated       Id LevelDisplayName Message
-----------       -- ---------------- -------
27.09.2026 10:05   7036 Informacja   Usluga Spooler przeszla w stan dzialania

[tomek@docker01 ~]$ journalctl -n 2
wrz 27 10:05:02 docker01 sshd[2201]: Accepted publickey for tomek
```

journalctl = journal control: czytanie dziennika systemd

TYLKO ODCZYT

Sprawdzenie adresu IP i testowanie połączenia: Get-NetIPAddress, ping oraz ip, ping

```
Get-NetIPAddress -AddressFamily IPv4
ping dc01
```

```
ip addr show
ping -c 3 dc01
```

Pierwsza komenda w każdej parze pokazuje adresy IP przypisane do kart sieciowych tego komputera. Druga wysyła kilka pakietów testowych do wskazanej nazwy i mówi, czy tamten komputer odpowiada.

Przykład:

```
PS C:\> ping dc01


Odpowiedz z 10.20.10.11: bajtow=32 czas=1ms TTL=128


[tomek@docker01 ~]$ ping -c 3 dc01
64 bytes from 10.20.10.11: icmp_seq=1 ttl=128 time=0.912 ms
```

□□ **IP** = Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection) · **ping** = Packet InterNet Groper: sprawdź, czy host odpowiada w sieci i jak długo trwa odpowiedź · **ip** = narzędzie do adresów, tras i interfejsów sieciowych

□□ TYLKO ODCZYT

Sprawdzenie miejsca na dysku: Get-Volume oraz df

```
Get-Volume
```

```
df -h /docker
```

Obie komendy pokazują, ile miejsca zajęte jest na dysku, a ile zostało wolnego. `df -h` w Linuksie pokazuje wynik w megabajtach i gigabajtach zamiast surowych bajtów, dzięki fladze `-h`.

Przykład:

```
PS C:\> Get-Volume -DriveLetter C


DriveLetter FileSystem Size    SizeRemaining
----------- ---------- ----    -------------
C           NTFS       200 GB  74 GB


[tomek@docker01 ~]$ df -h /docker
/dev/mapper/vgDocker-lvDocker  100G  61G  40G  61% /docker
```

□□ **df** = disk free: ile miejsca wolnego na dyskach

□□ ZMIENIA SYSTEM

Instalacja programu: winget oraz dnf

```
winget install NazwaPakietu
```

```
sudo dnf install NazwaPakietu
```

> **Pojęcie: menedżer pakietów.** Menedżer pakietów to program, który pobiera, instaluje i aktualizuje gotowe programy z zaufanego repozytorium, zamiast szukać instalatora w internecie samodzielnie. Winget robi to w Windows, dnf w Rocky Linux i innych systemach z rodziny EL.

Przykład:

```
PS C:\> winget install Git.Git
Znaleziono Git.Git [Git]
Pobrano.

[tomek@docker01 ~]$ sudo dnf install git
Installed:
  git-2.43.0-1.el9.x86_64
```

EL = Enterprise Linux: rodzina RHEL i jej klonów (Rocky, Alma, Oracle Linux) · **winget** = Windows Package Manager: menedżer pakietów do instalacji programów w Windows · **dnf** = Dandified YUM: menedżer pakietów RHEL/Rocky

ZMIENIA SYSTEM

Ponowne uruchomienie usługi: Restart-Service oraz systemctl restart

```
Restart-Service -Name Spooler
```

```
sudo systemctl restart sshd
```

Obie komendy zatrzymują usługę i od razu uruchamiają ją ponownie, bez ponownego uruchamiania całego systemu. Po restarcie warto sprawdzić stan usługi tą samą komendą, którą sprawdzaliśmy go wcześniej.

Przykład:

```
PS C:\> Restart-Service -Name Spooler
PS C:\> Get-Service -Name Spooler
```

```
Status   Name
------   ----
Running  Spooler

[tomek@docker01 ~]$ sudo systemctl restart sshd
[tomek@docker01 ~]$ systemctl is-active sshd
active
```

systemctl = system control: sterowanie usługami systemd

Z życia wzięte

Z ŻYCIA WZIĘTE

Tomek dostaje zadanie sprawdzenia obciążenia na docker01 i odruchowo szuka ikony w prawym dolnym rogu ekranu, tak jak otwierał Menedżera zadań na swoim laptopie. Na serwerze bez monitora nie ma paska zadań ani ikon, jest tylko czarne okno terminala po SSH. Podpowiadamy mu polecenie `top`, które w Linuksie pokazuje to samo, co Menedżer zadań w Windows: listę procesów i ich zużycie zasobów na żywo. Po tym dniu Tomek zaczyna prowadzić własną notatkę z odpowiednikami komend.

Jak to działa

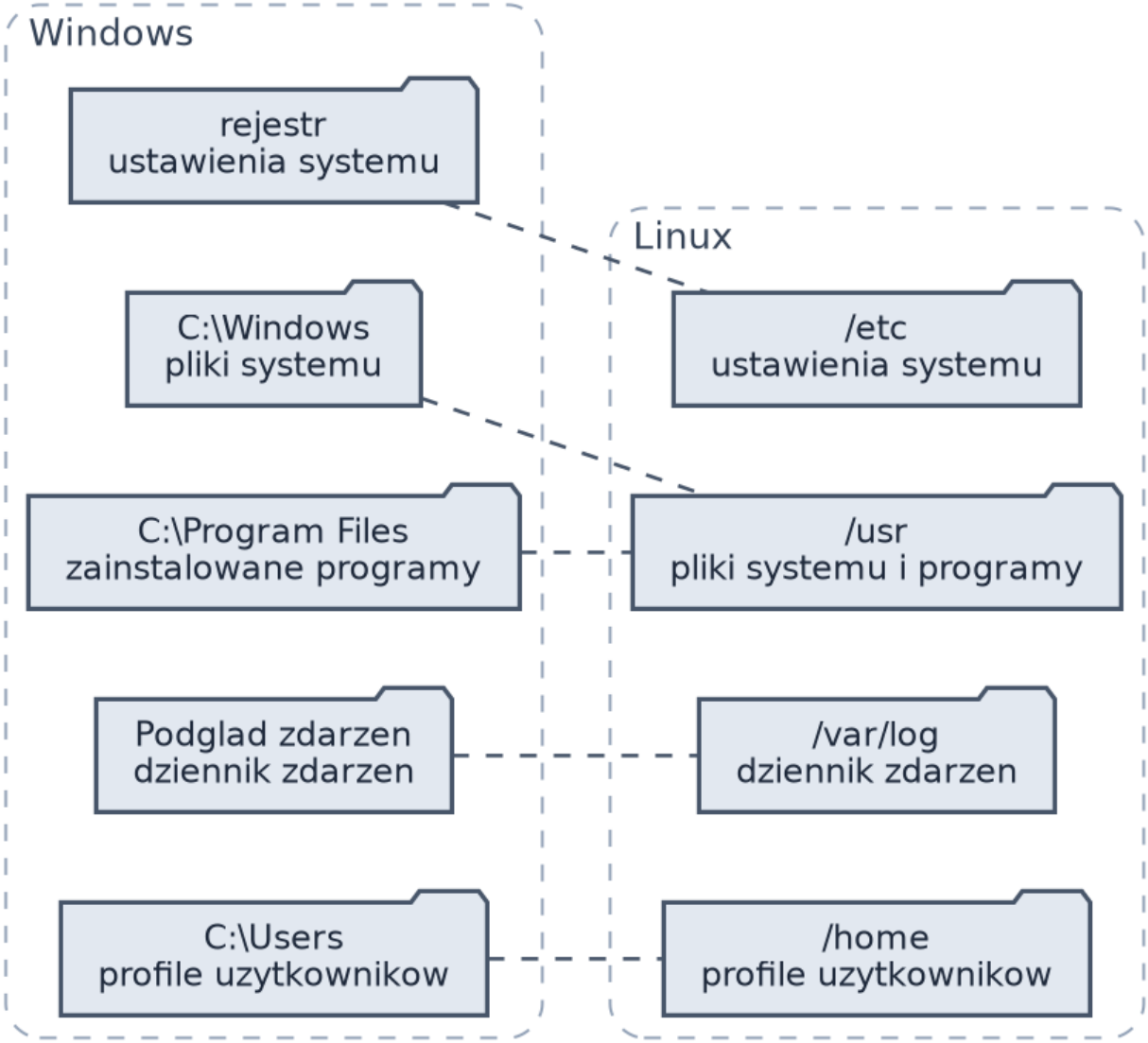
Rozwiń: jak to działa pod spodem

Poniższa tabela zbiera odpowiedniki najczęstszych czynności administratora w obu systemach. Nie trzeba się jej uczyć na pamięć, wracamy do niej, gdy zapomnimy jednej strony równania.

Co chcemy	Windows	Linux
Gdzie jesteśmy	`Get-Location`	`pwd`
Lista plików	`Get-ChildItem`	`ls -la`
Podgląd pliku	`Get-Content`	`cat`, `less`
Szukanie tekstu w pliku	`Select-String`	`grep`

Co chcemy	Windows	Linux
Kopiowanie pliku	`Copy-Item`	`cp`
Przenoszenie i zmiana nazwy	`Move-Item`, `Rename-Item`	`mv`
Usuwanie pliku	`Remove-Item`	`rm`
Tworzenie katalogu	`New-Item -ItemType Directory`	`mkdir`
Procesy	`Get-Process`	`ps aux`, `top`
Usługi	`Get-Service`	`systemctl status`
Restart usługi	`Restart-Service`	`systemctl restart`
Dziennik zdarzeń	`Get-WinEvent`	`journalctl`
Adres IP	`Get-NetIPAddress`	`ip addr`
Test połączenia	`ping`	`ping`
Miejsce na dysku	`Get-Volume`	`df -h`
Instalacja programu	`winget install`	`dnf install`
Edytor tekstu w terminalu	brak odpowiednika w CLI, zwykle Notatnik	`nano`, `vim`
Użytkownicy systemu	`Get-LocalUser`	`getent passwd`, `id`
Uprawnienia do pliku	zakładka Zabezpieczenia (NTFS, ACL)	`chmod`, `chown`
Konfiguracja systemu	rejestr	pliki w `/etc`

> **Pojęcie: rejestr systemu Windows.** Rejestr to jedna wielka baza ustawień całego systemu i zainstalowanych programów, ukryta pod programem `regedit`. Linux nie ma jednego takiego miejsca, każdy program trzyma swoje ustawienia we własnym pliku tekstowym w katalogu `/etc`.



Ustawienia systemu, pliki i dzienniki zdarzeń: te same rzeczy pod innymi nazwami w Windows i Linuksie (kliknięcie otwiera pełny rozmiar)

Poza nazwami komend te dwa systemy różnią się też sposobem myślenia, nie tylko słownictwem. Windows nie rozróżnia wielkości liter w nazwach plików, więc `Raport.txt` i `raport.txt` to ten sam plik. Linux rozróżnia je zawsze, więc te dwie nazwy to dwa osobne pliki, co bywa źródłem pomyłek dla kogoś, kto zaczynał na Windows.

Windows dzieli ścieżki ukośnikiem wstecznym `\`, jak w `C:\Program Files`, Linux zwykłym ukośnikiem `/`, jak w `/usr/bin`. PowerShell w praktyce toleruje też zwykły ukośnik, ale bash nigdy nie zrozumie odwrotnego.

> **Pojęcie: powłoka.** Powłoka to program, przez który wpisujemy komendy i dostajemy odpowiedzi, w Windows to PowerShell albo starszy CMD, w Linuksie najczęściej bash.

Linux trzyma się zasady „wszystko jest plikiem": dysk, drukarka, a nawet lista procesów w `/proc` dają się odczytać tymi samymi komendami co zwykły plik tekstowy. Windows takiej jednej zasady nie ma, dysk czyta się inaczej niż rejestr, a rejestr inaczej niż dziennik zdarzeń.

> **Pojęcie: potok.** Potok, zapisywany znakiem `|`, przekazuje wynik jednej komendy jako wejście do drugiej. W PowerShellu w potoku płyną gotowe obiekty z nazwanymi polami, w bashu płynie zwykły tekst, który trzeba dopiero rozciąć komendami takimi jak `grep` czy `awk`.

Częste błędy

Objaw	Przyczyna	Co robimy
`cd Dokumenty` w bashu nie działa, choć folder Documents istnieje	Linux rozróżnia wielkość liter, `Dokumenty` to nie to samo co `documents`	wpisujemy nazwę dokładnie tak, jak pokazuje `ls`
Ścieżka z Windows wklejona do basha nie działa	ukośniki wsteczne `\` nie znaczą w bashu tego samego co w Windows	zamieniamy `\` na `/`
`ls` w PowerShellu działa, ale niektóre flagi Linuksa nie	`ls` w PowerShellu to alias `Get-ChildItem`, nie prawdziwy `ls`	używamy natywnych parametrów PowerShella albo Windows Terminal z prawdziwym bashem
Zmiana w rejestrze nie widoczna od razu	niektóre klucze rejestru wymagają restartu usługi albo systemu	sprawdzamy w dokumentacji, czy dana zmiana wymaga restartu
Brak uprawnień mimo bycia administratorem	w Linuksie zwykłe konto nie ma praw roota, trzeba `sudo`	dopisujemy `sudo` przed komendą wymagającą uprawnień

□□ Ćwiczenia

1. Kolega z zespołu Oli mówi, że na Linuksie zgubił się plik `Raport.txt`, bo `ls` pokazuje tylko `raport.txt`. Co się właściwie stało?

Odpowiedź

Nic się nie zgubiło, oba pliki to na Linuksie dwie różne nazwy, bo system rozróżnia wielkość liter. W Windows `Raport.txt` i `raport.txt` byłyby tym samym plikiem, na Linuksie trzeba wpisać nazwę dokładnie tak, jak pokazuje `ls`.

1. Chcemy sprawdzić, czy usługa SSH działa na `docker01`, tak jak sprawdzalibyśmy usługę drukowania na Windows przez `Get-Service`. Jakiej komendy użyjemy i co dodatkowo zobaczymy w jej wyniku?

Odpowiedź

Użyjemy `systemctl status sshd`. Oprócz samego stanu usługi (uruchomiona albo zatrzymana) zobaczymy też kilka ostatnich linii jej dziennika, czego `Get-Service` nie pokazuje.

1. Dlaczego wynik `Get-Process` w PowerShellu łatwiej dalej przetworzyć niż wynik `ps aux` w bashu?

Odpowiedź

PowerShell przekazuje w potoku gotowe obiekty z nazwanymi polami, więc można od razu odwołać się na przykład do pola z zużyciem pamięci. Bash przekazuje zwykły tekst, więc żeby wyciągnąć jedną kolumnę, trzeba go dodatkowo pociąć komendami takimi jak `awk` czy `cut`.

Połączenie z serwerem z Windowsa: SSH, Windows Terminal i PuTTY

Poziom: podstawy **Dotyczy: OpenSSH (Windows Client), PuTTY** **Czas: ~20 min**

Ta strona pokazuje, jak z laptopa z Windows połączyć się z serwerem linuksowym takim jak `docker01`. Pokazujemy dwie drogi: wbudowanego klienta OpenSSH w Windows Terminal oraz starszy, wciąż spotykany program PuTTY. Wracamy tu, gdy zapomnimy dokładnej komendy albo znaczenia komunikatu o zmienionym odcisku klucza.

Skróty użyte na tej stronie (3)

Skrót	Co znaczy
DNS	Domain Name System: książka telefoniczna internetu, zamienia nazwy na adresy IP
IP	Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)
SSH	Secure Shell: szyfrowane zdalne logowanie do serwera

Fiszka

TYLKO ODCZYT

Sprawdzenie wersji klienta SSH: ssh -V

```
ssh -V
```

Ta komenda mówi, czy w systemie jest w ogóle zainstalowany klient SSH i w jakiej wersji.

> **Pojęcie: SSH.** SSH to szyfrowany sposób logowania się do innego komputera przez sieć i wydawania mu poleceń, tak jakbyśmy siedzieli bezpośrednio przed nim. Usługa `sshd` na serwerze nasłuchuje domyślnie na porcie `22`.

Klient OpenSSH jest częścią systemu bez dodatkowej instalacji od Windows 10 w wersji 1809 (aktualizacja z października 2018) i od Windows Server 2019. Wszystkie stacje robocze i serwery Windows w Zielonej mają już te wersje albo nowsze, więc `ssh` po prostu działa w Windows Terminal.

Przykład:

```
PS C:\Users\tomek> ssh -V
OpenSSH_for_Windows_9.5p1, LibreSSL 3.8.2
```

SSH = Secure Shell: szyfrowane zdalne logowanie do serwera · **ssh** = Secure Shell: szyfrowane zdalne logowanie

TYLKO ODCZYT

Połączenie z serwerem po SSH: ssh nazwa_usera@NazwaSerwera

```
ssh nazwa_usera@NazwaSerwera
```

(gdzie `nazwa_usera` to konto istniejące na serwerze, a `NazwaSerwera` to jego nazwa DNS albo adres IP)

Po podaniu hasła albo po sprawdzeniu klucza dostajemy znak zachęty serwera zamiast własnego komputera. Od tej chwili każda wpisana komenda wykonuje się na serwerze, nie na naszym laptopie.

Przykład:

```
PS C:\Users\tomek> ssh tomek@docker01
tomek@docker01's password:
Last login: Sun Sep 27 09:58:03 2026 from 10.20.20.55
[tomek@docker01 ~]$
```

SSH = Secure Shell: szyfrowane zdalne logowanie do serwera · **DNS** = Domain Name System: książka telefoniczna internetu, zamienia nazwy na adresy IP · **IP** = Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection) · **ssh** = Secure Shell: szyfrowane zdalne logowanie

TYLKO ODCZYT

Wygenerowanie pary kluczy SSH: ssh-keygen -t ed25519

```
ssh-keygen -t ed25519
```

> **Pojęcie: klucz SSH.** Klucz SSH to para dwóch powiązanych ze sobą plików: klucza prywatnego, który zostaje tylko na naszym komputerze i nigdy nikomu go nie pokazujemy, oraz klucza publicznego, który możemy bezpiecznie skopiować na dowolny serwer. Serwer wpuszcza nas, gdy nasz klucz prywatny pasuje do klucza publicznego zapisanego wcześniej w jego pliku `authorized_keys`.

Przykład:

```
PS C:\Users\tomek> ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (C:\Users\tomek/.ssh/id_ed25519):
Your identification has been saved in C:\Users\tomek/.ssh/id_ed25519
Your public key has been saved in C:\Users\tomek/.ssh/id_ed25519.pub
```

SSH = Secure Shell: szyfrowane zdalne logowanie do serwera · **ssh-keygen** = wygeneruj parę kluczy SSH albo sprawdź odcisk klucza

ZMIENIA SYSTEM

Skopiowanie klucza publicznego na serwer bez ssh-copy-id: type oraz ssh

```
type $env:USERPROFILE\.ssh\id_ed25519.pub | ssh nazwa_usera@NazwaSerwera "mkdir -p
~/.ssh && chmod 700 ~/.ssh && cat >> ~/.ssh/authorized_keys && chmod 600
~/.ssh/authorized_keys"
```

Na Linuksie do tego zadania służy gotowe polecenie `ssh-copy-id`, którego Windows nie ma. Ten jeden wiersz robi dokładnie to samo ręcznie: wypisuje nasz klucz publiczny i przesyła go przez SSH prosto do pliku `authorized_keys` na serwerze.

Przykład:

```
PS C:\Users\tomek> type $env:USERPROFILE\.ssh\id_ed25519.pub | ssh tomek@docker01 "mkdir
-p ~/.ssh && chmod 700 ~/.ssh && cat >> ~/.ssh/authorized_keys && chmod 600
```

```
~/.ssh/authorized_keys"
tomek@docker01's password:
```

SSH = Secure Shell: szyfrowane zdalne logowanie do serwera · **type** = pokaż, czym jest dane słowo dla powłoki: poleceniem, aliasem czy funkcją

TYLKO ODCZYT

Zapisanie skrótu do serwera: plik ~/.ssh/config

```
Host docker01
    HostName 10.20.10.42
    User tomek
    IdentityFile ~/.ssh/id_ed25519
```

Plik `config` w katalogu `.ssh` pozwala zapisać alias dla serwera razem z jego adresem, kontem i kluczem, więc nie trzeba ich za każdym razem wpisywać. Plik zakładamy w `C:\Users\tomek\.ssh\config` dowolnym edytorem tekstu.

Przykład:

```
PS C:\Users\tomek> ssh docker01
Last login: Sun Sep 27 10:12:44 2026 from 10.20.20.55
[tomek@docker01 ~]$
```

TYLKO ODCZYT

Skopiowanie pliku na serwer: scp

```
scp .\raport.xlsx nazwa_usera@NazwaSerwera:/home/nazwa_usera/
```

Komenda `scp` kopiuje plik tą samą szyfrowaną drogą co zwykłe połączenie SSH, bez osobnego programu do transferu plików. Ten sam alias z pliku `config` działa też tutaj zamiast pełnej nazwy serwera.

Przykład:

```
PS C:\Users\tomek> scp .\raport.xlsx tomek@docker01:/home/tomek/
raport.xlsx                   100%   48KB   2.1MB/s   00:00
```

SSH = Secure Shell: szyfrowane zdalne logowanie do serwera · **scp** = secure copy: kopiowanie plików przez SSH

TYLKO ODCZYT

Połączenie pulpitem zdalnym z serwerem Windows: mstsc

```
mstsc /v:NazwaSerwera
```

SSH loguje nas do wiersza poleceń, ale serwery Windows takie jak `dc01` czy `sql01` obsługujemy zwykle przez pełny pulpit zdalny. Polecenie `mstsc` otwiera to samo okno Podłączania pulpitu zdalnego, które znamy z menu Start, tylko od razu z podanym adresem serwera.

Przykład:

```
PS C:\Users\tomek> mstsc /v:dc01
```

SSH = Secure Shell: szyfrowane zdalne logowanie do serwera · **mstsc** = Microsoft Terminal Services Client: program otwierający pulpit zdalny (RDP) do innego Windows

KLIKAMY W INTERFEJSIE

Kliknięcia w PuTTY i PuTTYgen: interfejs

```
PuTTY: pole Host Name (or IP address) → NazwaSerwera → Port 22 → Connection type SSH → Open
PuTTYgen: Generate → poruszamy myszką nad pustym polem → Save private key → Save public key
```

PuTTY to starszy, samodzielny program do SSH, wciąż spotykany na starszych stacjach w firmach, które nie zdążyły przejść na Windows Terminal. PuTTYgen generuje klucze we własnym formacie, opisanym w sekcji niżej.

SSH = Secure Shell: szyfrowane zdalne logowanie do serwera

Jak to działa

Rozwiń: jak to działa pod spodem

> **Pojęcie: SSH i port 22.** SSH to protokół szyfrowanego zdalnego logowania, a jego usługa `sshd` na serwerze nasłuchuje domyślnie na porcie `22`. Cała droga połączenia z laptopa Tomka do serwera `docker01` przechodzi przez sieć firmową i zaporę `fw01`, zanim usługa sshd w ogóle zobaczy nasze żądanie.



Droga połączenia SSH od laptopa Tomka przez sieć i zaporę do powłoki na docker01 (kliknięcie otwiera pełny rozmiar)

Przy pierwszym połączeniu z nowym serwerem klient SSH pokazuje odcisk klucza tego serwera i pyta, czy mu ufamy. Za pierwszym razem to normalne pytanie, na które odpowiadamy `yes`, bo świeżo postawiony serwer jeszcze nie ma zapisanego odcisku po naszej stronie.

> **Pojęcie: odcisk klucza (fingerprint).** Odcisk klucza to krótki ciąg znaków jednoznacznie identyfikujący klucz danego serwera, zapisywany potem lokalnie w pliku `known_hosts`. Przy każdym kolejnym połączeniu klient po cichu porównuje aktualny odcisk z tym zapisanym, bez pytania nas o nic.

Zupełnie inaczej wygląda sytuacja, gdy klient nagle krzyczy dużymi literami „REMOTE HOST IDENTIFICATION HAS CHANGED". Ten komunikat pojawia się tylko wtedy, gdy odcisk klucza zmienił się od ostatniego połączenia, co zwykle oznacza przeinstalowany system na serwerze. Zanim wpiszemy cokolwiek dalej, warto zapytać kolegę z zespołu, czy faktycznie ktoś świadomie postawił ten serwer od nowa, bo to samo ostrzeżenie pokazuje też próba podszycia się pod serwer.

Logowanie kluczem jest bezpieczniejsze i wygodniejsze niż logowanie hasłem, bo klucza nie da się zgadnąć ani podejrzeć znad ramienia. Hasło wystarcza do jednorazowego dostępu albo do kont serwisowych bez pliku `authorized_keys`, ale w codziennej pracy administratora klucz jest normą. Wiele serwerów w Zielonej ma logowanie hasłem w ogóle wyłączone, żeby ktoś z zewnątrz nie mógł go zgadywać automatem.

> **Pojęcie: plik .ppk.** PuTTY nie rozumie standardowego formatu klucza OpenSSH i wymaga własnego formatu `.ppk`, generowanego programem PuTTYgen. Klucz

wygenerowany komendą `ssh-keygen` trzeba najpierw wczytać do PuTTYgen i zapisać ponownie jako `.ppk`, zanim PuTTY zechce go użyć.

Z życia wzięte

Z ŻYCIA WZIĘTE

Tomek pierwszy raz w życiu łączy się z serwerem przez SSH, celuje w `docker01`. Widzi pytanie o odcisk klucza i komunikat, którego nigdy nie widział, więc zamyka okno i zgłasza „awarię łączności". Pokazujemy mu, że to zwykłe pierwsze połączenie, wpisujemy `yes` razem i logujemy się bez przeszkód. Dwa tygodnie później Tomek sam tłumaczy nowej osobie różnicę między tym pytaniem a prawdziwym ostrzeżeniem o zmienionym kluczu.

Częste błędy

Objaw	Przyczyna	Co robimy
`Connection refused`	usługa SSH na serwerze nie działa albo nasłuchuje na innym porcie	sprawdzamy `systemctl status sshd` na serwerze albo pytamy administratora serwera
`Connection timed out`	serwer nie odpowiada wcale, zwykle zapora albo zła nazwa serwera	sprawdzamy `ping NazwaSerwera` i reguły na `fw01`
`Permission denied (publickey)`	serwer nie zna naszego klucza publicznego albo wymaga hasła, którego nie podaliśmy	sprawdzamy, czy klucz trafił do `authorized_keys`, próbujemy `ssh -v` dla szczegółów
`Host key verification failed`	zapisany wcześniej odcisk klucza serwera różni się od tego, co dostajemy teraz	potwierdzamy z zespołem, że serwer faktycznie zmienił klucz, dopiero potem usuwamy stary wpis z `known_hosts`
PuTTY nie widzi klucza wygenerowanego przez ssh-keygen	PuTTY rozumie tylko format `.ppk`, nie standardowy plik OpenSSH	wczytujemy klucz do PuTTYgen i zapisujemy jako `.ppk`

Ćwiczenia

1. Łączymy się pierwszy raz z nowo postawionym serwerem `k3s01` i klient pyta, czy ufamy jego odciskowi klucza. Czy to powód do niepokoju?

Odpowiedź

Nie, to normalne pytanie przy pierwszym połączeniu z serwerem, którego odcisku klucza jeszcze nie znamy. Niepokoić powinien dopiero komunikat „REMOTE HOST IDENTIFICATION HAS CHANGED" przy serwerze, z którym łączyliśmy się już wcześniej.

1. Kolega prosi o skopiowanie klucza publicznego na serwer, ale pracuje na Windows i nie ma `ssh-copy-id`. Jak inaczej zrobi to samo?

Odpowiedź

Może wypisać zawartość swojego pliku `id_ed25519.pub` komendą `type` i przekazać ją przez potok do `ssh`, które dopisze klucz do pliku `authorized_keys` na serwerze. To dokładnie to samo, co robi `ssh-copy-id` na Linuksie, tylko krok po kroku.

1. Klucz wygenerowany przez `ssh-keygen` nie działa w PuTTY. Dlaczego i co trzeba zrobić?

Odpowiedź

PuTTY rozumie tylko własny format klucza `.ppk`, a nie standardowy plik OpenSSH. Trzeba wczytać klucz do programu PuTTYgen i zapisać go ponownie jako plik `.ppk`.