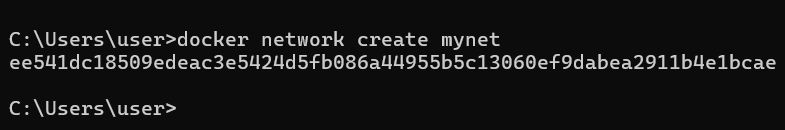
Джумантурлы Сержан

**Lab 1: Docker арқылы интернет-протоколдарды зерттеу**

Мақсат

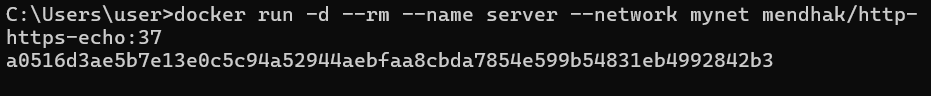
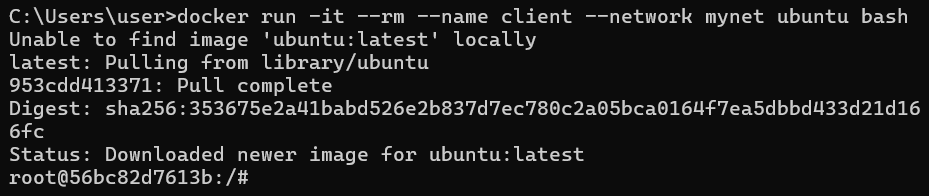
Бұл жұмыста сіз әртүрлі интернет-протоколдармен танысасыз. Ол үшін бір Docker желісінде екі контейнер іске қосылады: біреуі сервер, екіншісі — клиент рөлін атқарады. Сіз әртүрлі протоколдарды қолданып тәжірибе жасап, олардың айырмашылықтарын салыстырасыз.

**Ортаны дайындау**

1. Арнайы Docker желісін құрыңыз.  
   

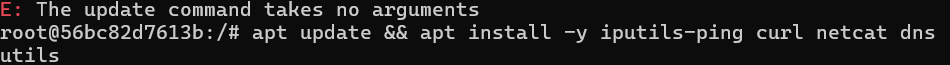
Командалық жолда Докер желісін құрдым. Сәтті құрылды айди шықты.

1. Осы желіде екі контейнерді іске қосыңыз (server және client).

Server контейнері mendhak image базасында құрылды. Ал Client контейнері ubuntu image базасында құрлды.

1. Екі контейнерге де қажетті желілік құралдарды орнатыңыз.

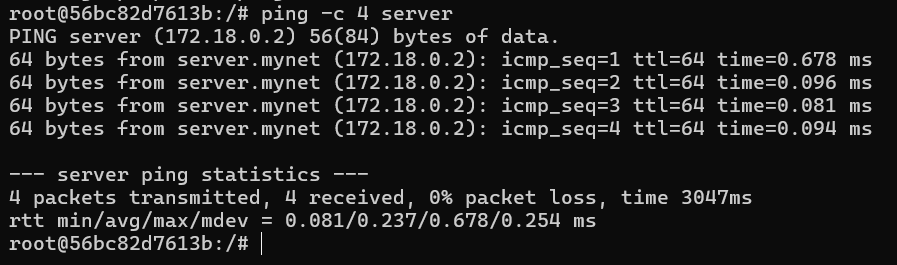




Бұл жерде мен контейнерлерге осы лабароториялық жұмыста берілген протоколдармен жұмыс жасау үшін арнайы құралдар пакетін жаздым.

**1. ICMP (Ping)**

* Клиент контейнерінен сервер контейнерінің қолжетімділігін тексеріңіз.
* Пакеттердің жеткізілуіне және жауап уақытына назар аударыңыз.



Ping тексеру арқылы қолжетімлік сәтті өтті. 4 пакетте жіберілді, 4 пакетте қабылданды, процессінде пакет жоғалу пайызы 0% ге тең.

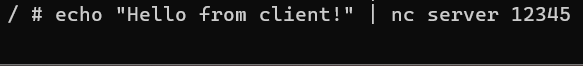
**2. TCP (Netcat)**

* Сервер контейнерінде TCP тыңдағышты іске қосыңыз.



Nc -l -p 12345 командасы арқылы TCP тыңдатқышты іске қостым. Енді ол клиент контейнерінен хабар күтеді.

* Клиент контейнерінен хабарлама жіберіңіз.



Echo “Hello from client!” команда арқылы хабар жіберіледі

Бұл жерде nc server 12345 мәтіне server контейнеріне аддрес болап келеді.

* Хабарламаның қалай жеткізілетінін және қабылданатынын бақылаңыз.

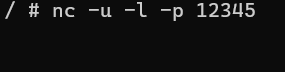
Изображение выглядит как текст, Шрифт, снимок экрана, типография

Содержимое, созданное искусственным интеллектом, может быть неверным.

Хабар сәтті жетті server контейнерінде exec те “Hello from client!” жазуы шықты

**3. UDP (Netcat)**

* Сервер контейнерінде UDP тыңдағышты іске қосыңыз.



* Клиент контейнерінен хабарлама жіберіңіз.



Изображение выглядит как текст, Шрифт, снимок экрана, Графика

Содержимое, созданное искусственным интеллектом, может быть неверным.

* TCP-мен салыстырғандағы айырмашылықты байқаңыз.

TCP арнайы “қол алысу” қослуы арқылы орындалады. Жіберілген әр пакет жеткені расталуы керек. Жоғалған пакеттер қайтып жіберіледі. Пакеттер ретпен жетеді. Жылдамдығы баяу. Пакеттер міндетті түрде жіберілуі керек, пакеттер жоғалса қайта жіберіледі.

UDP ешқандай косылу жоқ (қосылусыз жұмыс істейді). Сенімділік аз, растау жоқ пакеттер жетпей қалуы мүмкін. Жылдамды жоғары. Пакеттер ретсіз келеді.

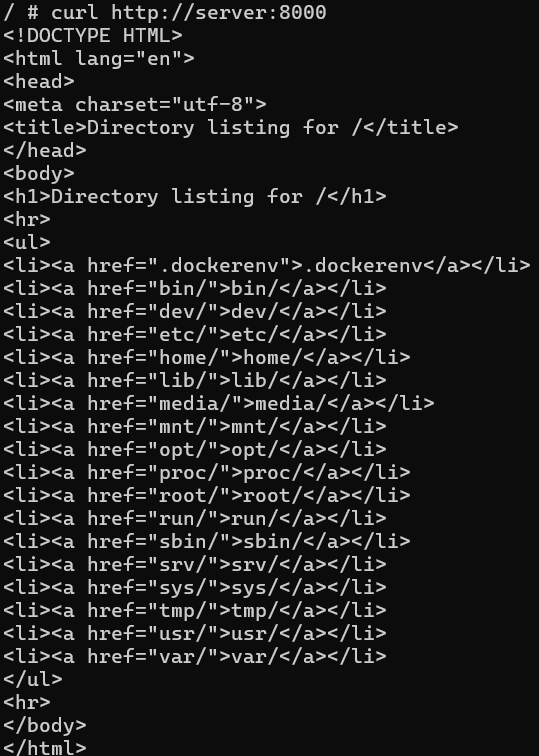
**4. HTTP**

* Сервер контейнерінде шағын HTTP-сервер орнатыңыз.  
  



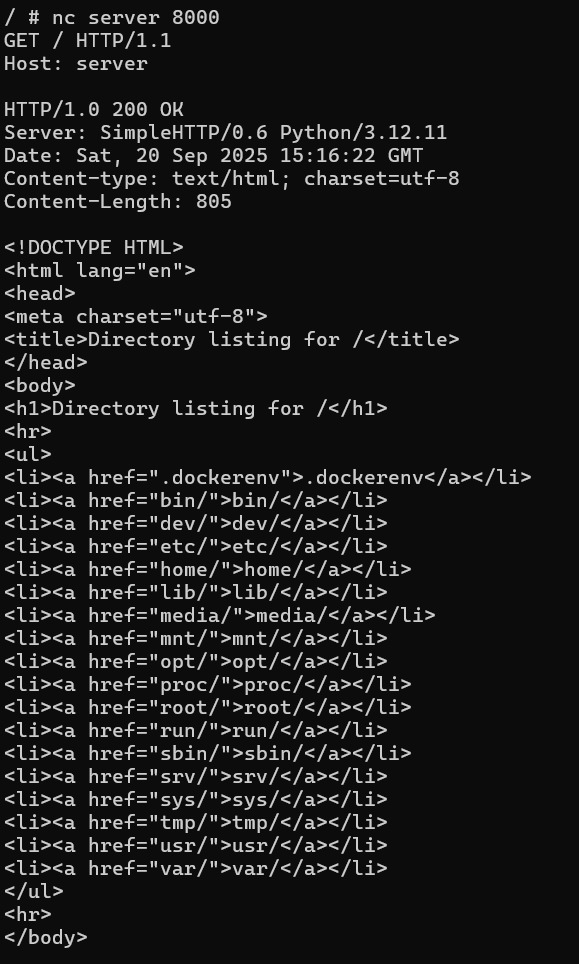
Бұл команда арқылы шағын HTTP сервер орнаттық. Host енді сұрау күтіп тұр.

* Клиент контейнерінен HTTP-сұрау жіберіңіз.



HTTP сұрау curl арқылы нәтижесінде сервер директориясындаға файлды қайтарды.

* Нәтижені таза TCP-мен салыстырыңыз.

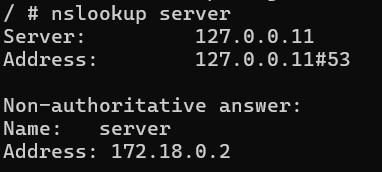


TCP арқылы nc арқылы хосттан өзім команда жазып сервердің директориясындаға файлды қайтардым. Командаларды өзің жазу ға тура келеді.

HTTP curl арқылы тез автоматты түрде қайтарады. GET / HTTP/1.1 және Host: server сияқты тақырыптарды автоматты жібереді.

**5. DNS**

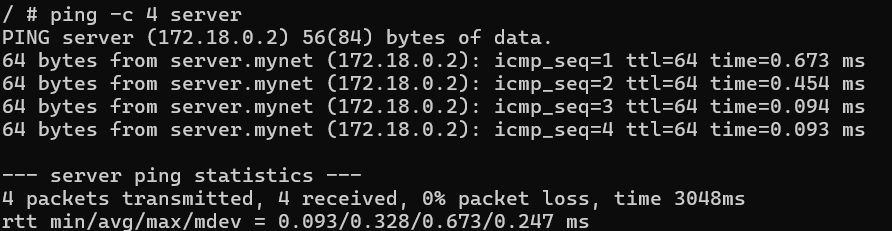
* Клиент контейнерінен сервер контейнерінің атын шешіңіз (resolve).



* Docker-дің кірістірілген DNS-і контейнерлердің өзара байланысын қалай қамтамасыз ететінін зерттеңіз.

Доекердің ішкі DNS қызметі желі ішіндегі контейнерлерге IP аддресін автоматты береді. Егер --name server деген команда арқылы контейнер ашсақ. Докердің ішкі DNS жүйесі server атын server контейнерінің IP не ауыстырады. Контейнердің IP-ін ойда сақтау керек емес тек атын білсек болды, басқа контейнерлер server деп қосыла алады.

Мысалдар:



Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

Көріп тұрғандай client контейнері ішінен server контейрінен тек атын жазу арқылы сұрау алдық.

**Ойлануға арналған сұрақтар**

1. Қай протоколдар хабарламаның жеткізілуіне кепілдік береді?

TCP протоколы хабарламаның жеткізілуіне кепелдік береді.

* Қол алысу косылуы   
  жүреді
* Пакеттер ретпен жіберіледі
* Баяу жетікізіледі (көптеген тексерулерге байланысты)
* Процессте жоғалған пакеттер қайта жәберліеді.

1. HTTP TCP-дің мүмкіндіктерін қалай пайдаланады?

Ең алдмен TCP арқылы косылу орнатылады. Содан кейін HTTP сұраулар жіберіледі. HTTP сұраулар TCP сегентеріне бөлінеді, TCP оларды жоғалтпай, дұрыс ретпен серверге жібереді. Жауап алу кезінде HTTP жауаптар TCP арқылы тексеріліп, жауаптың бір бөлігі жоғалса TCP оны қайта жібереді. HTTP сұраудан кейін TCP жұмыс жасауы аяқталады.

1. Неліктен DNS контейнерлердің желілік әрекеттесуінде маңызды рөл атқарады?

Әр контейнерлердің IP адрессін біліп жүруге міндет етпейді. Құрған кезде берілшен атын докердің ішкі DNS жүйесі контейнер атын IP адресске айналдырады. Контейнер арасында қосылу жасау үшін тікелей контейнердің атын жазу арқылы сұрау жасауға болады. Яғни контейнерлер бір-бірін аттары арқылы таба алады. Бұл жүйені сенімді әрі ыңғайлы етеді.

1. Қандай жағдайларда TCP орнына UDP қолдану тиімді болады?

TCP → деректер жоғалмауы керек болса (веб, пошта, банкинг, файл жіберу).

UDP → жылдамдық пен latency басты болса (ойындар, видео/аудио, DNS, VoIP).