Protokoły rozproszone w praktyce — Ściągamy linuxa

Zadania kwalifikacyjne

Grzegorz Uriasz — gorbak25@gmail.com

19 kwietnia 2026

Ostateczną wersję rozwiązań zadań proszę przesłać w systemie warsztatów, w terminie podanym na stronie WWW. Proszę dołączyć skrypty/programy powstałe podczas rozwiązywania zadań. Wyślij to co udało ci się wykonać — częściowe rozwiązania też są ok, ważna jest chęć pracy i rozwoju. Jeżeli coś nie jest jasne, napisz do mnie maila.

> **Mam jednak prośbę — nie korzystaj z AI** (chyba że jest tak wskazane w poleceniu). LLMy potrafią rozwiązać większość z tych zadań od ręki, ale nie o to w tym chodzi. Chodzi o to żebyś *Ty* się czegoś nauczył(a).

Środowisko

Warsztaty będą prowadzone w Pythonie. Upewnij się że masz zainstalowanego Pythona 3.10 lub nowszego. Nie potrzebujesz żadnego specjalnego środowiska — wystarczy twój laptop z ulubionym edytorem.

[Obowiązkowo, 1 pkt] Dlaczego chcesz uczestniczyć w tych warsztatach?

W maksymalnie 2 zdaniach odpowiedz na powyższe pytanie.

[Obowiązkowo, 1 pkt] Czy jesteś człowiekiem?

Przepisz tekst z powyższego obrazka. Opisz krótko jak rozwiązałeś(aś) captchę — podpowiedź: rozwiązanie jest związane z tematyką warsztatów :)

[Obowiązkowo, 3 pkt] Bajty na linii

Napisz w Pythonie program, który łączy się z serwerem TCP pod adresem `149.248.223.186` na porcie `1234`, odbiera dokładnie 64 bajty, odsyła je w odwrotnej kolejności, a następnie odbiera flagę w formacie WWW22{…} od serwera. Po otrzymaniu flagi zamyka połączenie.

Rozwiązaniem zadania jest:
1. Jaka jest flaga?
2. Krótki opis jak działa twój program (2-3 zdania)

Parsowanie binarne (5 pkt)

Plik .torrent to w zasadzie słownik zakodowany w formacie bencode (Bencoding). Bencoding definiuje cztery typy: bajty (np. `4:spam` = `"spam"`), liczby całkowite (np. `i42e` = 42), listy (np. `l4:spam4:eggse`) oraz słowniki (np. `d3:cow3:mooe`).

Zaimplementuj dekoder bencode

Zaimplementuj w Pythonie dekoder bencode — funkcję która przyjmuje bajty i zwraca zdekodowany obiekt Pythona (str, int, list, dict). Nie wolno korzystać z gotowych bibliotek do bencode.

Zdekoduj .torrent

Pobierz plik: alpine-minirootfs-3.23.3-x86_64.tar.gz.torrent Używając swojego dekodera, zdekoduj go i wypisz:
1. Jaki jest announce URL?

2. Jaka jest nazwa pliku?
3. Ile kawałków (pieces) ma plik?
4. Jaki jest rozmiar każdego kawałka?

Do rozwiązania dołącz ściągnięty plik .torrent.

Weryfikacja w sieci niezaufanych peerów (4 pkt)

W BitTorrencie pobierasz kawałki pliku od nieznajomych. Nie masz powodu im ufać — mogą wysłać ci zepsute dane. W zadaniu 5 widziałeś że plik .torrent zawiera pole `pieces` — ciąg haszy SHA1, po jednym na każdy kawałek pliku. To pozwala zweryfikować każdy kawałek osobno: haszujesz pobrany kawałek, porównujesz z haszem z .torrenta, jeśli się zgadza — kawałek jest ok.

Ale teraz wyobraź sobie sytuację: plik ma 10,000 kawałków, czyli 10,000 haszy SHA1 do przechowania. A co jeśli zamiast trzymać 10,000 osobnych haszy, połączylibyśmy je parami — hasz z hasza? Pierwszy hasz to SHA1(kawałek1), drugi to SHA1(kawałek2), a ich *rodzic* to SHA1(hasz1 ‖ hasz2). I tak dalej, aż dojdziemy do jednego hasza na samej górze.

Narysuj drzewo (2 pkt)

Narysuj taką strukturę dla 8 kawałków pliku (poziom 0: 8 haszy kawałków, poziom 1: 4 hasze, poziom 2: 2 hasze, poziom 3: 1 hasz na górze). Opisz w 2-3 zdaniach jak przy pomocy tej struktury zweryfikować że kawałek nr 3 jest poprawny, znając tylko hasz na samej górze i kilka haszy po drodze. Ile haszy (poza samym kawałkiem) musisz znać żeby to zrobić?

Pytania (2 pkt)

1. Co się stanie jeśli złośliwy peer zmieni jeden bajt w jednym kawałku — ile haszy trzeba przeliczyć żeby to wykryć?
2. Dlaczego ta struktura jest lepsza niż trzymanie 10,000 osobnych haszy, jeśli chcesz przekazać komuś sposób weryfikacji *jednego* kawałka?
3. (Opcjonalne) Czy znasz system lub protokół który używa podobnej struktury w praktyce? (podpowiedź: używasz go prawdopodobnie codziennie)

(Opcjonalne, +2 pkt) Dlaczego BitTorrent jest odporny?

Wyobraź sobie że tracker pada w trakcie pobierania pliku przez BitTorrenta. Czy pobieranie musi się zatrzymać? Jakie mechanizmy (istniejące lub hipotetyczne) pozwoliłyby kontynuować pobieranie nawet bez trackera? Odpowiedz w 3-5 zdaniach.

Powodzenia i do zobaczenia w Zabrzu!