Eötvös Loránd Tudományegyetem
Informatikai Kar

Webes alkalmazások fejlesztése

1. előadás

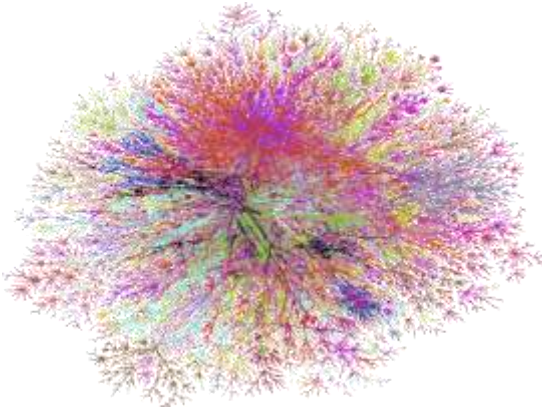
Webes alkalmazások és biztonságuk

Cserép Máté
mcserep@inf.elte.hu
http://mcserep.web.elte.hu

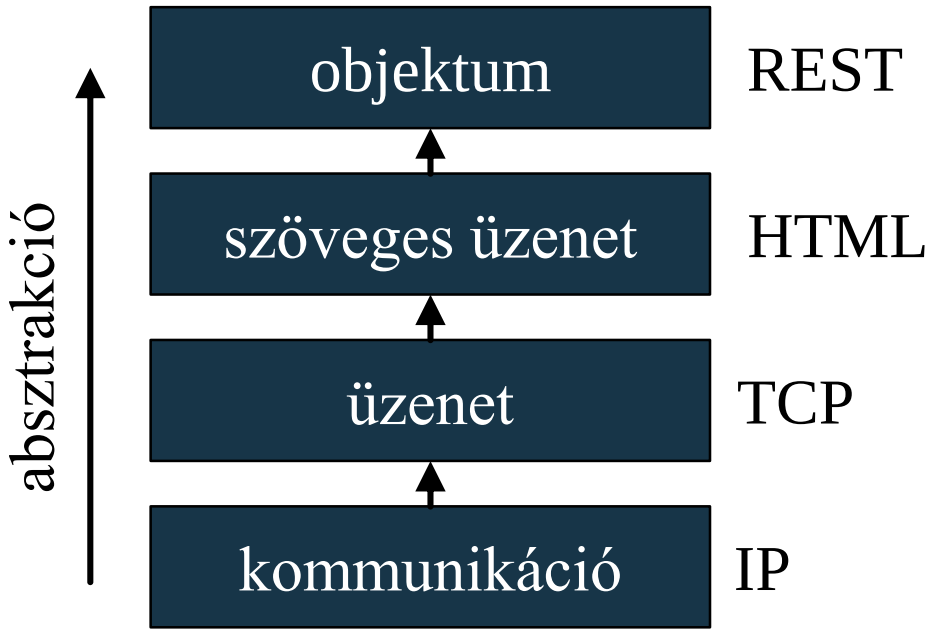
Webes alkalmazások és biztonságuk

Kommunikáció

- Alkalmazások sok esetben folytatnak kommunikációt hálózaton keresztül, általában az *OSI modell*re épülve
  - különböző kommunikációs megoldásokat nyújt egy többrétegű architektúrában
    - *alkalmazás*: TLS, SSH, SMTP, …
    - *megjelenítési*: HTML, CSS, JSON, …
    - *szállítási*: UDP, TCP
    - *hálózati*: IP
  - a modellre további kommunikációs és szolgáltatási *architektúrák* (MVC, REST), és *keretrendszerek* épülnek (ASP.NET, WCF)



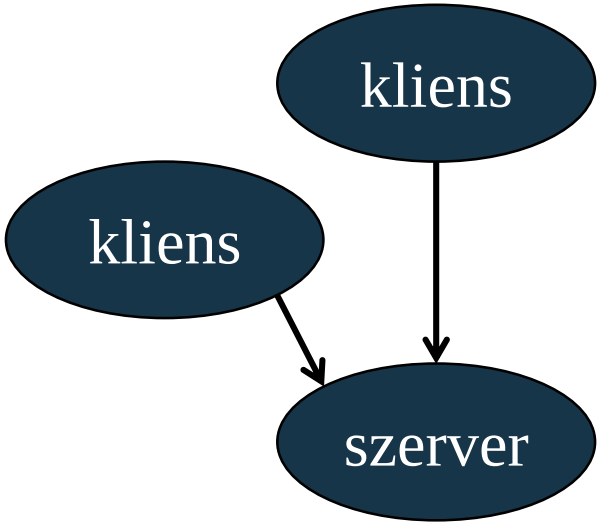
- a felsőbb rétegek absztrakciót nyújtanak, elfedik az alsóbb szinteket
  - növelik a *biztonság*ot, illetve a *megbízhatóság*ot
  - biztosítják az összetett tartalommal való kommunikációt



Webes alkalmazások és biztonságuk

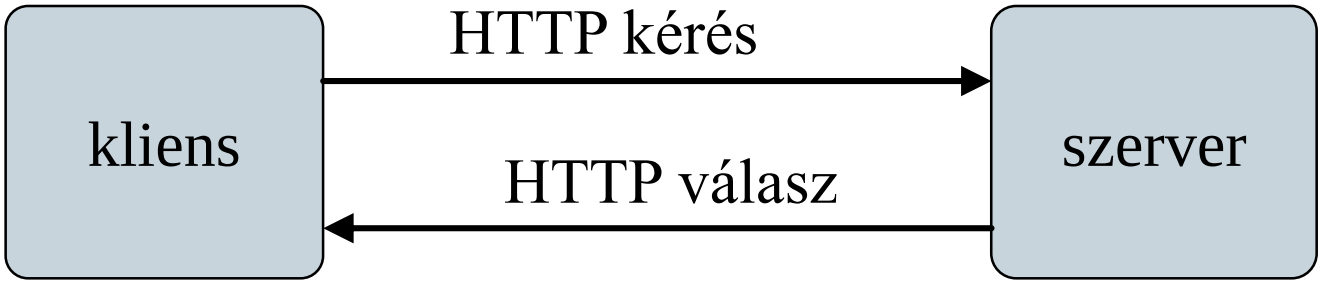
Rendszerek

- A hálózaton kommunikáló alkalmazás-rendszerek alapvetően két kategóriába sorolhatóak:
  - *decentralizált*, közvetlen kapcsolatú (P2P)
  - *kliens-szerver*: egy központ gép látja el a funkciókat, és szolgál ki tetszőleges sok csatlakozó gépet, amely lehet:
    - *vastagkliens* (*thick client*): a végrehajtást a kliens végzi, a szerver főleg kapcsolattartásra, adatkezelésre szolgál
    - *vékonykliens* (*thin client*): a végrehajtást a szerver végzi, a kliens interakcióra szolgál



A HTTP protokoll

- A webes adatközlés alapvető protokollja a *HTTP* (*Hypertext Transfer Protocol*)
  - a kérés/válasz paradigmára épül, vagyis a kliens elküld egy kérést, amelyre a szerver válaszol
    - a választ értelmezi és megjeleníti a kliens
    - a válasz első sora a státuszsor, amely visszajelez a kérés végrehajtásáról (pl. 200 sikeres, 404 nem található, 503 szolgáltatás nem elérhető)

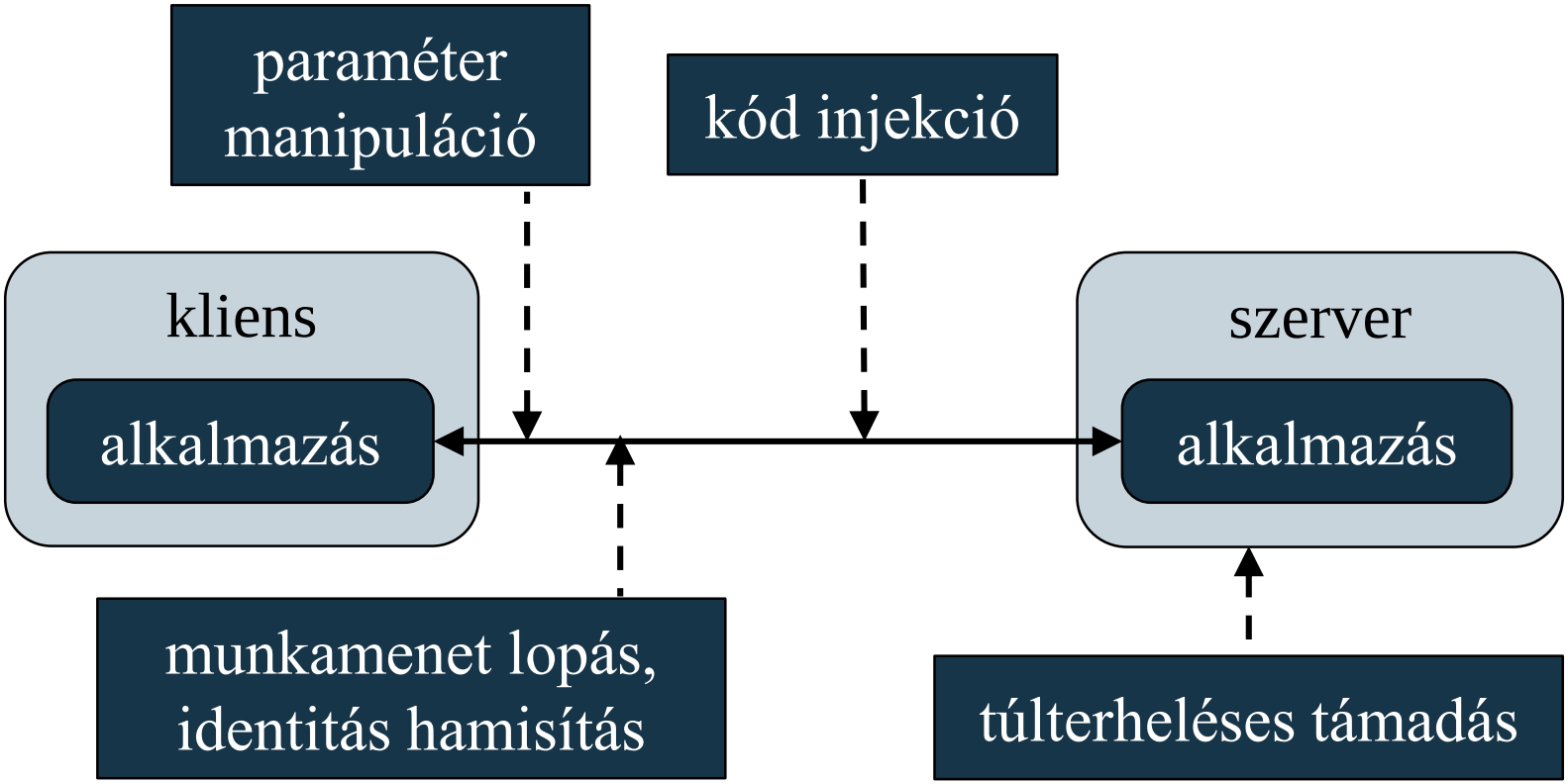


- a kérésnek több típusa lehet, pl.:
  - *GET*: adott (elérési útvonalhoz tartozó) erőforrás lekérése
  - *POST*: adatokkal ellátott tartalom küldése (pl. űrlap)
- a protokoll *állapotmentes*, azaz a szerverre érkező kérések egymástól függetlenek (két kérés között nincs állapotmegőrzés)
- ugyanakkor a szerver felépíthet egy *munkamenet*et (*session*) a kliens felé (az azonosító adatok kliens oldalon kerülnek mentésre)
- a protokoll biztonságossá tehető TSL titkosítással (HTTPS)

Webes alkalmazások és biztonságuk

Biztonság

- Egy webes alkalmazás esetén külön hangsúlyt kell helyezni a biztonságra, mivel számos ponton támadás érheti



Webes alkalmazások és biztonságuk

Biztonsági támadások

- A biztonsági támadások célja lehet:
  - *rendszerek megismerése, feltérképezése*
    - Spamhaus (2013)
  - *rendszerekbe történő beférkőzés, manipuláció*
    - titkosítások feltörése
    - felhasználás további támadásokhoz
    - Stuxnet (2010), BEAST (2011), Shellshock (2014), Heartbleed (2014)
  - *rendszerek megbénítása, leállítása*
    - túlterheléses támadások (*DoS*)
    - CloudFlare (2015), GitHub (2015), BBC (2016)

Webes alkalmazások és biztonságuk

Biztonsági támadások

- *adatlopás, adathalászat*
  - kulcsfontosságúak a személyes és pénzügyi adatok
  - az adatokat lehetőség szerint megfelelően titkosítani kell
    - nem visszafejthető kódok (MD5, SHA1, SHA512)
    - sózás (a tartalom módosítása, kiegészítése)
  - Playstation Network (2010), Epsilon (2011), Citigroup (2011), LinkedIn (2012), Adobe (2013), Snapchat (2013), World of Warcraft (2013), Carbanak (2014), Apple (2014), Sony (2014), JPMorgan Chase (2014), Mozilla (2014), Yahoo (2014), eBay (2014), Anthem (2015), British Airways (2015), US OPM (2015), Uber (2016), Bupa (2017), Equifax (2017)

Webes alkalmazások és biztonságuk

Kód injekció

- A *kód injekció* során kártékony kód kerül átküldésre az alkalmazáshoz, ennek módszerei:
  - *SQL injekció*: a szerveren futó SQL utasításokat manipulálja
  - *cross-site scripting* (*XSS*): szkript kerül feltöltésre a szerverre, amelyet a kliens böngészője futtat
  - *dinamikus kiértékelés*: a szerver által dinamikusan kiértékelt tartalmat manipulálja
  - *távoli fájl injekció*: fájl beküldése a szerverre, majd a tartalom futtatása
  - *parancssori injekció*: rendszerutasítások futtatása a szerveren

Webes alkalmazások és biztonságuk

Kód injekció

- Pl. (SQL injekció):

```
SqlCommand command = con.CreateCommand();
command.CommandText = "select * from user where
   userName = '" + textName.Text + "' and
   userPass = '" + textPass.Text + "';
command.ExecuteReader();
   // textPass.Text = "' or '1' = '1" betölti
   // a teljes táblát
```

- Pl. (XSS):

```
labelPost.Text = textInput.Text;
   // textInput.Text = "<script>…</script>" esetén
   // lefut a script a válaszban
```

Webes alkalmazások és biztonságuk

Kód injekció

- Sebezhetőségi pontok:
  - bemenő adat (kliens és szerver oldali) ellenőrzésének hiánya, vagy gyengesége (az alapértelmezetten lefutó automatikus ellenőrzés is okozhat sebezhetőségi pontokat)
  - SQL utasítások ellenőrzés nélküli, szöveg alapján történő létrehozása és futtatása, adatbázis-hozzáférés szabályozatlansága
  - gyenge kivételkezelés, és a kivétel információk megjelenése kliens oldalon (számos belső információt jeleníthet meg, amely tovább könnyíti a behatolást)

Webes alkalmazások és biztonságuk

Paraméter manipuláció

- A *paraméter manipuláció* (*parameter tampering*) során a kliens oldalon tárolt speciális információkat írják át, úgymint:
  - oldal argumentumok, űrlap tartalmak
  - süti tartalmak, rejtett mezők, nézetállapotok, HTTP fejlécek
- Pl. (oldal argumentum):

```
order.php?id=245601&disc=0
    // disc=100 esetén ingyenes a rendelés
```

- Pl. (süti tartalom):

```
user_id=3013
auto_login=true
    // a user_id átírásával másként jelentkezünk be
```

Webes alkalmazások és biztonságuk

Paraméter manipuláció

- Sebezhetőségi pontok:
  - oldal argumentumok túl nagy felelősséggel való felruházása, pl. elérési útvonalak megadása, felhasználói azonosítás (munkamenetek helyett)
  - felhasználói információk tárolása kliens oldalon, pl. felhasználónév és jelszó sütiben
  - információk (sütik) továbbítása nem biztonságos csatornán (HTTP)
  - rejtett mezők használata az oldalban kritikus információ tárolásra

Webes alkalmazások és biztonságuk

Munkamenet lopás

- A *munkamenet lopás* (*session hijacking*) során a munkafolyamat azonosítót lopják el, így hozzáférhetnek a felhasználói munkafolyamathoz
  - a munkamenetet általában sütik segítségével tárolják a kliens oldalon, ezért a süti megszerzése egyben a munkafolyamathoz való hozzáférést is biztosítja
- Sebezhetőségi pontok:
  - munkamenet sütik továbbítása nem biztonságos csatornán
  - munkamenet lejáratának korlátozatlansága
  - munkamenet sütin kívüli authentikáció hiánya