Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Solidity okosszerződések ellenőrzése

Horváth Márk (BSc) mérnökinformatikus szakos hallgató
Konzulensek:
Kocsis Imre tanársegéd, MIT
Hajdu Ákos doktorandusz, MIT
Szoftverfejlesztés szakirány
Önálló laboratórium Összefoglaló
18/19. II. félév

Az Önálló laboratórium során elsősorban a Solidity okosszerződések ellenőrzésével, verifikációjával foglalkoztam. Mivel az előző félév során a Témalaboratórium tárgy keretein belül megismerkedhettem a blockchain rendszerek és technológiák működésével, ezért számomra könnyebb volt a kutatás, hiszen már alapból volt fogalmam az okosszerződések mechanizmusairól.

Az önállólabor célja az volt, hogy megismerjem milyen módszerek vannak Solidity okosszerződések ellenőrzésére és verifikációjára, majd az ezeket segítő eszközök használatával, működésével tisztában legyek. Végül pedig ezeknek a felhasználásával egy okosszerződés ellenőrzés analízist készítsek, amelyből kiderül, hogy melyik ellenőrzési módszer mennyire hatásos. Ezek számos új fogalmat és módszert mutattak be amiknek segítségével az ellenőrzés alaposabb, kimerítőbb és könnyebben alkalmazható lesz.

A félév első részében megismerkedtem az okosszerződés ellenőrzésének a fogalmával, módszereivel illetve alapból az ellenőrzések fontosságával. Ezek alapján a solc-verify nevű automatikus formális verifikációs eszközzel kezdtem az ismerkedést. Ehhez hozzátartozik az általa használt SMT solverek és a Boogie programnyelv megismerése. Ezek ismereteinek segítségével már képes voltam saját okosszerződés ellenőrzésére, egyes hibák okának felderítésére. Következőnek egyéb verifikációs eszközöket próbáltam ki, melyek közül a MythX statikus analízist végző eszköz bizonyult a leghatékonyabbnak így később ezt használtam a komparációhoz. Továbbá megismerkedtem a Solidity AST (abstract syntax tree) szerkezetével, amely segítségével egy okosszerződésből többfajta mutációt lehet létrehozni (Python script).

Ezt felhasználva képesek vagyunk egy adott okosszerződésbe mutáció segítségével tetszőleges hibákat generálni, majd a kapott okosszerződéseket lefuttatni a használt verifikációs eszközökön. Így tesztelhetünk különböző hibákat, az eszközök hatékonyságát, a hibák kritikusságát.