JavaCard alkalmazások biztonsági tulajdonságainak formális verifikációja

Verók István
Budapesti Műszaki és Gazdaságtudományi Egyetem, Híradástechnika Tanszék
Email: vi@shamir.ebizlab.hit.bme.hu

Az intelligens chipkártyás rendszerek (és ezek közt is főként a JavaCard) újfent érdeklődést keltettek a formális módszerek iránt. Habár a nagy Java programok még mindig meghaladják a legtöbb formális eszköz képességeit, a JavaCard egyszerre elég kicsi és elég használható ahhoz, hogy felkeltse a teljes formalizálhatóság reményét, és mégis hasznos legyen valós életbeli problémák megoldásában. Erőforrás- és képességkorlátozottsága (kis memóriaméret, egyszálú programfuttatás, kevés adattípus, szemétgyűjtés hiánya) jó egyensúlyt teremt a két ellentétes szempont közt.

PACAP

A formalizálási kutatások célja már az első lépésektől elsősorban a megbízhatóság és az adatbiztonság vizsgálata volt. Ennek egyik első eredménye volt a PACAP-projekt. Egy olyan automatikus eszköz lett a végterméke, amely appletek lefordított bájtkódjában ellenőrzi a “biztonságos függés” tulajdonságot [1]. Ez a tulajdonság minden információtároló memóriarekeszhez biztonsági címkéket rendel, az ezek közti adatáramlás pedig csak akkor engedélyezett, ha a cél-rekesz rendelkezik minden, a forrásrekeszben is meglevő címkével.

Az engedélyezett címkék specifikációja és a vizsgálandó kisalkalmazás bájtkódja egyaránt a PACAP bemenetei. Maga a PACAP egy előfeldolgozó, amely az SMV modell-ellenőrző formátumában gyárja le a metódusok és objektumok modelljeit, valamint a címkespecifikáció alapján ellenőrzendő állításokat is, melyeket az SMV ellenőriz [2].

JML

Az Iowa State University-n Gary Leavens és csoportja megalkották a Java Modeling Language nevű “viselkedésleíró felületspecifikációs nyelvet" [4]. Ez az Eiffel nyelvből ismert “szerződéses tervezés” (design by contract) alkalmazása Javában, tehát a programozó betartandó elő- és utófeltételeket adhat metódusoknak. Ezek boolean-értékű Java-kifejezések (azonban használhatók bennük az elsőrendű `\forall` és `\exists` kvantorok is). Az esetlegesen kivételekkel visszatérő metódusok viselkedését külön `exceptional_behavior` ágakkal is meg lehet adni.

A holland nijmegeni egyetemről származó LOOP eszköz a legfejlettebb JML-ellenőrző. Fordításkor, statikusan képes tulajdonságokat ellenőrizni.

Verificard

A Verificard 2000-ben kezdődött, és jelenleg a legtöbb fontos európai formális chipkártyás kutatás az égisze alá tartozik. A kutatások három fő, egymásra épülő irányban folynak: a platform, az API, és alkalmazói programok verifikációja. Ezek közül itt az elsőt fejtem ki részletesebben.

Ide tartozik a Java fordító helyességbizonyítása (ez lényegében a kétféle formátum kiértékeléséből kapott hatások ekvivalenciáját mondja ki), ugyanígy történt a CAP konverter helyességvizsgálata is.

A kártyán található bájtkód-ellenőrző (Bytecode Verifier, BCV) kutatása több irányban is zajlik, a legáltalánosabb irány Klein és Nipkow nevéhez fűződik [3], ők egy absztrakt virtuálisgép-leírásból egy adatfolyam-analizátor segítségével automatikusan generáltak bájtkód-ellenőrzőt (ez a megközelítés a jövőben akár más nyelvek, például a Microsoft-féle MSIL formalizálásához is újrafelhasználható lehet).

A bájtkód-ellenőrzők mellett a Java virtuális gép (Java Virtual Machine, JVM) modellezésére fordítják a legtöbb erőfeszítést. A LOOP-projekt [5] részeként készült JVM-modell része a memóriamodell, az objektumorientált öröklés, a kivételkezelés, a vezérlési szerkezetek. A Jive-projekt érdekessége, hogy egy praktikusan választott JavaCard-részhalmazzal foglalkozik: a figyelmen kívül hagyott részek még egy esetben sem hiányoztak nekik valós programok elemzésében. A JavaCard-forrásokat a már említett "szerződéses tervezés" révén egészítik ki specifikációkkal, ebből az SJIVE eszközzel állítanak elő tételbizonyítókkal bizonyítandó állításokat. A KeY-projekt viszont mintegy "mellékesen" tartalmaz egy JVM-modellt: a célja ugyanis a szoftvertervezésben, mint mérnöki folyamatban használható specifikációs és verifikációs eszközök fejlesztése.

Jövőbeli célok és fejlesztések

Teljes JavaCard szemantika: a meglevő részek összeállításával a JavaCard platform minden részletét teljesen lefedő formális modell.

Jobb eszközök: temporális logikai kifejezések bevezetése a JML meglevő elsőrendű kijelentéslogikája mellé.

Több kisalkalmazás kölcsönhatásainak leírása: vagy platformszintű módosításokkal, vagy magasabbszintű protokollok bevezetésével (pl. egyfajta challenge-response), cél a PACAP-hoz hasonló bizalmi viszonyok megteremtése.

JavaCard túlnövése: a teljes Java nyelv egyszálú szekvenciális viselkedésének leírása (igen előrehaladott állapotban).

Többszálú végrehajtás: a szinkronizáció, kölcsönös kizárás és hasonló fogalmak elsősorban a többi nyelv esetében játszanak szerepet, a JavaCard platform még ma is túl szükös ehhez.

Irodalomjegyzék

[1] P. Bieber, F. Cuppens: A Logical View of Secure Dependencies, 1992. http://www.cert.fr/francais/deri/bieber/papers/jcs.ps.gz

[2] P. Bieber, J. Cazin, P. Girard, J.-L. Lanet, V. Wiels, G. Zanon: Checking Secure Interactions of Smart Card Applets, 2000. http://www.cert.fr/francais/deri/bieber/papers/2000esorics_long.ps.gz

[3] G. Klein, T. Nipkow: Verified Bytecode Verifiers, 2001. http://www4.informatik.tu-muenchen.de/~nipkow/pubs/fossacs01.ps.gz

[4] G. T. Leavens, A. L. Baker, C. Ruby: Preliminary Design of JML: A Behavioral Interface Specification Language for Java, 2001. ftp://ftp.cs.iastate.edu/pub/techreports/TR98-06/TR.ps.gz

[5] G. Barthe, G. Dufay, L. Jakubiec, B. Serpette, S. M. de Sousa: A Formal Executable Semantics of the JavaCard Platform, 2001. ftp://ftp-sop.inria.fr/lemme/personnel/Gilles.Barthe/esop01.ps.gz