A biztonságos információkezelés (secure processing) alapjai

SZŐLLŐSI LORÁND, MAROSITS TAMÁS

Budapesti Műszaki és Gazdaságtudományi Egyetem, Távközlési és Médiainformatikai Tanszék
Nagysebességű Hálózatok Laboratórium
{szollosi, marosits}@ttt-atm.tmit.bme.hu

Kulcsszavak: informatikai biztonság, secure processing, titkosítás, doboz-struktúráltság

A megbízható, bizalmas feldolgozás technikai háttere kulcskérdéssé vált. Annál is inkább foglalkoznunk kell ezzel, mivel a digitális hordozó közegeken a szerzői jogok tarthatósága megbízható platform és eszközkészlet nélkül megkérdőjeleződik. Ugyanakkor problémát jelent a felhasználói jogok (anonimitás, biztonsági másolat, forrás hitelességének ellenőrizhetősége stb.) biztosítása is. A matematikai háttér adott, a kvantumprocesszorok korának beköszöntéig gyakorlati válasz adható az elméleti kérdéskörre. Cikkünkben magas szintről indulva szeretnénk összefoglalni a secure processing elméleti kérdéseit és útmutatást adni olyan konkrét rendszerek kidolgozásához, melyek alkalmazása esetén egyik fél érdekei sem sérülnek.

1. Bevezetés

A biztonságos feldolgozást nyújtó rendszerek kutatásának egyik legfontosabb mozgatórugóját a DRM (Digital Rigths Management) rendszerek jelentik. A témakör iránti érdeklődést jól mutatja, hogy a Conference on Communications and Multimedia Security [6] három vitaindító előadása közül az egyik a DRM rendszerek architektúrájával foglalkozik.

Tágabb és nehezebben kezelhető terület a PC alapú biztonságos környezet megteremtése, erre egy – sokat vitatott – megoldás a TCPA/TPM. [7] bemutatja a TPM egy, a felhasználók érdekeit előtérbe helyező lehetséges esetét: egy, a felhasználó személyi adatait és pénzügyi információit kezelő, a világhálóra kapcsolt szerver esetén kívánatos, hogy ez a szerver igazolni tudja a felhasználó felé az adatkezelési szabályzatban foglaltak betartását. Ugyanez a jelentés bemutat több elméletileg kivitelezhető támadást (többek között visszajátszásos támadást) a jelenlegi rendszer ellen. Arbaugh [8]-ban bemutatja a jelenlegi TCPA specifikáció olyan kibővítésének lehetőségét, mely a személyiségi jogokhoz kapcsolódó érdekeket is figyelembe veszi.

Ritter széleskörű gyűjteményét adja a felhasznált elméleti alapoknak, egyúttal közöl több, a témába vágó szabadalmi leírást [5].

2. Klasszikus információ és új követelmények

2.1. A klasszikus információ tulajdonságai

Témánk keretein belül információnak nevezünk minden olyan adatot, amely jelentéssel, jelentőséggel bír, így feldolgozásra alkalmas. A klasszikus információ teljesen más tulajdonságokkal rendelkezik, mint a klasszikus anyag. Így az arra vonatkozó törvényi szabályozásnak egészen másnak kell lennie, különben a jogbiztonság kérdésessé válik.

Az információ tulajdonságai:

1. *Oszthatatlan egységekre bontható,* melyek további darabolása vagy fizikailag lehetetlen, vagy az adat általa elveszti eredeti tartalmát, jelentését és így már nem tekinthető információnak. Minden általunk tekintett információ véges sok részre osztható.
2. Tökéletesen *replikálható* (lemásolható) anélkül, hogy az eredeti példány használhatósága romlana.
3. *A replikáció* ténye *nem bizonyítható* és *nem cáfolható*.
4. Önmagában nem azonosítja a forrást, a felhasználót, nem hitelesíti az adat integritását, nem hordozza a saját történetét; azaz a hordozó struktúra *keretezése memóriamentes*.

Fontos megjegyeznünk, hogy ezen tulajdonságok nem korlátozzák az informatikát – így nem jelentenek korlátokat az informatika felhasználhatóságára sem –, csak egy konkrét technológia ismérvei. Ezen jellegzetességek miatt azonban korunk informatikai rendszereiben a biztonság és az adatkezelés bizalmassága nem mindig tartható követelmények.

2.2. Alkalmazások és követelmények

Kényelem: Felhasználói szemmel nézve a legfontosabb döntési szempont, hogy az eszköz használata ne legyen bonyolult, a felhasználó ne ütközzön nehézségekbe. Nem tételezhető fel, hogy egy kényelmetlen rendszert bárki hosszú távon használni fog, legyen az bármennyire biztonságos.

Anonimitás: Két irányban vizsgálhatjuk: egyrészt bizonyos esetekben a felhasználó (például mint vásárló) rejtve akar maradni a tulajdonos/szolgáltató (például eladó) előtt. Másrészt sokkal általánosabb probléma, hogy harmadik fél ne szerezzen tudomást a tranzakció paramétereiről, különösen a felhasználóhoz kapcsolható bizalmas jellegű adatokról.

Tranzakciók letagadhatatlansága: Mindkét fél érdeke, hogy a tranzakciók letagadhatatlanok legyenek. Ehhez nem elég, hogy a felek kölcsönösen ellenőrizni

tudják, hogy a másik jóváhagyta-e a szóbanforgó tranzakciót, arra is szükség van, hogy ezt harmadik fél felé bizonyítani tudják titkos információ kiadása nélkül. A nyilvános kulcsú aláírás erre tökéletesen megfelelő eljárást ad.

Áteresztőképesség: A rendszer egyes elemeinek, például egy kliens-szerver modellben a szervergép(ek)nek jelentős terhelést kell elviselniük, ezáltal gyakran szűk keresztmetszetet jelenthetnek. Az áteresztőképesség a végrehajtható tranzakciók számát korlátozza, túllépése rosszabb esetben rendszerösszeomlást okozhat. Profitorientált szereplők ezért csak megfelelően nagy áteresztőképességű rendszerek alkalmazásában érdekeltek.

Költség: Elsősorban az egy tranzakcióra eső költség a döntő. A szolgáltató ezt vagy közvetlenül a felhasználóra terheli (növelve ezzel az árat, csökkentve a keresletet), vagy az előre megszabott árból próbálja kigazdálkodni a tranzakció költségét. A tranzakciós költség gazdasági megszorítást jelent egy rendszer alkalmazhatóságára és a technológia fejlődésével gyorsan csökken.

Tranzakciós adatok integritása: A információ sértetlenségének biztosítása nélkül a szolgáltató jogi kötelezettségeinek sem tud eleget tenni, például nem tud könyvelni. Szerencsére általában az integritás-problémák még a hibásan tervezett rendszerek esetén is a tranzakciók egy kis hányadát érintik; ezek kezelése azonban hosszú időt vehet igénybe és a rendszerbe vetett bizalmat is megingathatja (gondoljunk csak a bankkártyaszám-lopásokra vagy a szavazat-újraszámlálásra).

Az információ tulajdonságai adottak, azokat nem tudjuk megváltoztatni. Tudunk azonban olyan reprezentációt, olyan protokollokat kifejleszteni, amelyek során adott információcsomagok feldolgozása a kívánt működést mutatja, beleértve a biztonsági követelmények teljesítését is. Több ilyen rendszert láthatunk a gyakorlatban működni, melyek tulajdonságait a lenti *táblázat* foglalja össze.

A TCPA (Trusted Computing Platform Alliance) a Microsoft, Intel, IBM, HP és AMD szövetsége melynek célja a gyártók és fejlesztők számára biztonságosabb PC létrehozása. Ennek elérése érdekében egy erős kriptográfiát használó ellenőrző chipet építenének először az alaplapra, majd a processzorba. Sajnos a technológia felhasználók számára az eddiginél is kevesebb kontrollt biztosít a számítógépük felett. Bővebb információ a technológiáról az irodalomban [2] olvasható.

2.3. Az informatikával szemben támasztott új követelmények összegzése

Az előző szakasz alapján látható, hogy az informatikával szemben mind az új piacok, mind a környezet radikálisan új igényeket támasztanak. Ugyanakkor a jelenlegi informatikai megoldások a fenti problémákat csak külön-külön kezelik, vagyis minden, a biztonsággal vagy a bizalmassággal kapcsolatban felmerülő kérdésre egyedi válaszaink vannak. Szükség van egy olyan integrált megoldásra, amely egyszerre képes az összes felmerülő kérdésre megnyugtató választ adni. Feltételezhetjük, hogy egy ilyen integrált megoldás olcsóbb, egyszerűbben menedzselhető és kisebb erőforrásigényű lehet, mint az egyedi megoldások összessége.

Biztonsági követelmények teljesülése a különböző alkalmazásoknál

Rendszer	Használók érdekeinek teljesülése				Szolgáltatók vagy jogtulajdonosok érdekeinek teljesülése			
	Kényelem	Anonimitás tulajdonosok felé	Anonimitás harmadik fél felé	Tranzakciók letagadhatatlansága	Áteresztőképesség	Költség	Tranzakciós adatok integritása	Tranzakciók letagadhatatlansága
Home banking	+	–	–	+(?)	+	+	+	+
Internetes választások	–	–	+	–	+	–	+(?)	+/–
Jegyrendelés	+	–	+	–	+	–	–	–
Internetes vásárlás bankkártyával	+	–	+/–	–	+/–	–	+/–	+/–
TCP/TPM	+	–	–	–	–	+	+	+
Egyedi processzorazonosító	+	–	–	–	–	+	–	–

Jelmagyarázat:
+: Az adott érdek teljesülése a jelenlegi rendszerekben biztosított
–: Az adott érdek teljesülése a jelenlegi rendszerekben nem biztosított
+/–: Egyes rendszerekben nem, másokban biztosított az adott érdek teljesülése
+(?): Az adott érdeket a rendszer figyelembe veszi, de kérdéses vagy ellenőrízhetetlen a teljesülése

Ugyanakkor lehetséges, hogy bizonyos esetekben ennek az integrált rendszernek nem tudjuk az összes lehetőségét kihasználni és bevezetése nem gazdaságos.

Az új követelmények több kategóriába sorolhatóak. Egyrészt fontos egy új információfogalom (melyet a továbbiakban SPI-nek – secure processing information – nevezünk) kialakítása, amely valamilyen anyagszerűen kezelhető entitást kell fedjen. Ennek kapcsán az alábbi feltételeknek való megfelelést kell vizsgálnunk:

- Az információ véges sok, tovább nem osztható egységből kell álljon.
- Az információ *képes legyen magát azonosítani* és az *integritása* biztosított legyen. (Azaz „látszik", hogy mit tartalmaz.)

Másrészt olyan tulajdonságokat is megkövetelünk, melyeket az anyaggal kapcsolatban nem a fizika, hanem a jogrend biztosít a felhasználóknak, azaz

- Definiáljon egy *interfészt*, melyen át – és csak azon át – elérhető és feldolgozható az információ. (adatvédelemmel kapcsolatos törvények hétköznapokban)
- A *replikációjára tett kísérletek eredménytelenek* legyenek mindaddig, amíg ezt az interfész nem engedélyezi. (Lopás vagy visszajátszásos támadás elleni védelem.)
- *Forrása és összes módosítója azonosítható* legyen. (Legyen története, vagyis a keretszerkezet rendelkezzen memóriával. Hétköznapi példa erre a könyvelés.)

Az egyes szempontok persze különböző mértékben fontosak az informatika különböző szereplőinek, mivel azok különböző nézőpontokból vizsgálják a kérdéskört. Alapvetően három érdekcsoportot tudunk elkülöníteni:

- A *szellemi termékek* (szoftverek, tervek) *szerzőijog tulajdonosainak* érdekeit kell biztosítanunk. Erre megfelelő, ha egyrészt a *felhasználó bizonyítja jogosultságát* (de nem azonosítja saját magát) az SPI felé, másrészt csak meghatározott interfészen át érhetjük el a programokat.
  Program alatt a továbbiakban olyan információt értünk, melyet a felhasználónak joga van futtatni. Természetesen a felhasználó nem feltétlen pusztán a gép előtt ülő személy, hanem lehet egy másik program is (például adatbetöltés), vagy másik információ (például hivatkozás). Ezeknek a programoknak és információkereteknek is át kell esniük egy hasonló azonosítási procedúrán.
- A *rendszerüzemeltetők* (rendszergazdák, adminisztrátorok) célja a hatékony karbantartás lehetőségének a fenntartása. Szükségük van valamilyen eszközre, amivel a felhasználók jogosultságait állíthatják. Adott esetben szükséges lehet olyan jogosultságok kiadására is egyes felhasználók számára, amelyekkel a rendszergazda maga sem rendelkezik, tehát a rendszergazda nem csak a saját jogait engedheti át másoknak; továbbá a saját jogkörét nem tágíthatja.
  Fontos a felhasználók által végzett műveletek naplózása, követése.
- A *felhasználók* azt szeretnék, ha a rendszer iránt érzett – és a korábbi rendszerek használata során már megszokott – komfortérzetük nem csökkenne, ugyanakkor nem akarnak az információgyűjtés áldozatává sem válni. Ezt azért sem lenne szerencsés, mivel így monopóliumhelyzettel való visszaélés válik lehetségessé (információ programhoz kötése). Továbbá így – különösen üzleti felhasználók esetén – titkos adatok, információk megszerzése lenne lehetséges. Egyes felhasználócsoportok (például cégek, vagy azok osztályai) tovább szeretnék szűkíteni, személyre szabni a tagjaik számára elérhető funkciókat.

Látható, hogy a szereplők érdekei eltérőek. Ezt a helyzetet felfoghatjuk egy új evolúciós kalitkának. Evolúciós kalitkának nevezünk minden olyan környezetet, ahol a mintahalmaz (jelen esetben az informatikai megoldások halmaza) egy eleme el tud helyezkedni anélkül, hogy versenyhelyzetbe kerüljön valamely társával. Biológiai környezetünkben az új fajok kialakulását eredményezi egy evolúciós kalitka megnyílása, hasonlóan az informatikában új technológiák jelenhetnek meg. A jövő informatikájának tényleges szabályait az fogja meghatározni, hogy az informatika szereplőiből létrejövő érdekcsoportok mennyire tudják érvényesíteni elvárásaikat; egyáltalán: milyen hamar kapcsolódnak be ennek a szabályrendszernek a kialakításába.

3. A jelenlegi eszközök felhasználhatósága

A jelenleg rendelkezésünkre álló informatikai eszközkészlet (matematikai törvényszerűségek, algoritmusok, számítógépes technológiák, konkrét alkalmazások) a fentiekben vázolt problémák egy részét már képes megoldani. Ezek közül a legfontosabbak:

1. **Hashképzés (MD5, SHA1)**: Hashfüggvénynek nevezünk egy olyan dimenziószűkítő függvényt, mely könnyen számolható, de nehezen invertálható, és nehéz két azonos függvényértéket eredményező bemenetét találni. Az MD5 hash ennek tipikus példája [1,3]. A lenyomatkészítő függvényeket azért alkalmazzuk, hogy általuk az adat szándékos vagy véletlen módosítása felismerhető legyen. Azaz, ha az információnk MD5 hash-e rendelkezésre áll, ellenőrizhetjük annak *integritását*. Ez azonban nem nyújt módot a forrás azonosítására. Általában aláírás előtt alkalmazzák, hogy kisebb méretű szövegen kelljen végrehajtani a számításigényes RSA algoritmust.
2. **Nyilvános kulcsú aláírás (RSA, DSA)**: A nyilvános kulcsú aláírás lényege, hogy minden szereplő kap egy hitelesítő központtól egy kulcspárt: E (publikus) és D (titkos) kulcsokat, melyek – mint függvények – egymás inverzei, továbbá E-ből D nehezen számítható. Az E kulcsot nyilvánosságra hozza a résztvevő, míg a D kulcsot titokban tartja. Ez utóbbit használja aláírásra. (Az algoritmus pontos leírása meg-

található [3]-ban). Bárki, aki birtokában van az adott személy nyilvános kulcsának, ellenőrizni tudja, hogy egy információ tőle érkezett-e, tehát ez az eljárás lehetőséget nyújt a *hitelesség* biztosítására is. Az elektronikus aláírás egy további problémát is megold, nevezetesen a *letagadhatatlanságot*: azaz, ha valaki egyszer aláírt egy dokumentumot, akkor többé nem tudja letagadni az aláírás tényét, mert bárki könnyedén tudja ellenőrizni a dokumentumot. Felmerül, hogy hogyan tudjuk biztosítani az E kulcs hitelességét.

Erre megoldást nyújt egy mindenki által ismert CA, azaz hitelesítő központ, mely a kulcs eredetiségét és tulajdonosának kilétét saját aláírásával garantálja. Vegyük észre, hogy bár eddig személyekről beszéltünk, valójában egy processzor is részt vehet az aláírásban, amennyiben képes a szükséges műveleteket elvégezni és aláírását titokban tartani. Azaz ellenőrizheti például a felhasználó processzora (vagy egy alkalmazás, amire már most is számtalan példát mondhatnánk az internetes banki rendszerektől a különböző plug-inekig.), hogy az általa kapott programot olyan személy vagy cég írta-e, akinek a programjait a felhasználó engedi futtatni; így pedig kiküszöbölhetőek a vírusok.

3. **Nyilvános kulcsú titkosítás (RSA)** [3]: Hasonló az elve az aláíráséhoz, csak itt az adatok *titkosságának* biztosítása a cél. Azaz a partner nyilvános kulcsával titkosítjuk az adatot, melyet csak ő tud majd dekódolni, mert csak ő ismeri a saját titkos kulcsát. A titkosság egyúttal azt is jelenti, hogy a kommunikáció *lehallgathatatlanná* válik, mivel a titkos információ (cyphertext) az azt lehallgató személy számára nem szolgáltat információt. Szintén lehet egy processzor is szereplője az eljárásnak, ekkor a processzormagba úgy juttathatunk kódot és adatot, hogy az külső felek (lehallgatók) számára értelmetlen. Ezzel megoldhatjuk a *másolhatatlanságot* is, ami az anyag és a klasszikus információ közötti legszembetűnőbb eltérés: ugyanis, ha a programot a fentiekben vázolt módon juttatjuk a processzorba, akkor azt hiába másoljuk le, nem tudja egy másik processzor végrehajtani.

   Természetesen a processzorunk lemásolhatja azt, illetve ha kiadja saját titkos kulcsát, akkor sem garantálható tovább a másolhatatlanság. Azonban, ha csak azon processzorok kaphatnak megfelelő minősítést, melyek ezen szabályokat betartják, és az információ előállítója kéri a processzortól ezt a megfelelőségi tanusítványt, mielőtt számára lekódolná az adatokat, akkor garantálható a zavartalan működés. Vegyük észre, hogy ezzel sem a felhasználó, sem a processzora nem kellett, hogy azonosítsa magát; pusztán a hozzáférés jogosságát igazolta a processzor.

Van azonban egy Achilles-sarka a jelenlegi technológiák csokránának: ez pedig a rosszindulatú programokkal szembeni védtelenség. Vírusok ugyan nem juthatnak a felhasználó gépére, de csak azt tudhatja jelenleg, hogy *kitől származik* a kód, amit futtat, és nem azt, hogy *mit csinál,* vagy hogy *mihez fér hozzá*. Olyan környezetet kell tehát alkotnunk, amelyben eleve lehetetlen a jogosultsági határokon adatokat átszivárogtató programok írása. Erre nyújt megoldást a javasolt hierarchikus jogosultsági szinteken alapuló, általunk „doboz-struktúrált”-nak nevezett információfeldolgozási környezet.

4. Doboz-struktúrált feldolgozási környezet

A hagyományos informatikai környezetekben megszoktuk, hogy a gyermek-folyamatok a szülőjüktől viszonylag független életet élnek: saját maguk foglalhatnak memóriát, írhatnak és olvashatnak állományokat, foglalhatnak le processzoridőt (rosszabb esetekben ez utóbbi automatikusan és megállíthatatlanul történik, működésképtelenné téve a rendszert).

Ha teljes biztonságot szeretnénk a felhasználó számára nyújtani, akkor ezt a programozó-barát megközelítést fel kell, hogy adjuk. Cserébe egyrészt jól lokalizálhatóvá válnak a nem megfelelően működő modulok, azaz amelyek nem felelnek meg az interfészüknek, vagy nem komformak a felhasználó szándékával, másrészt jól menedzselhetővé válnak a rendszer erőforrásai.

4.1. A doboz-struktúráltság feltételei

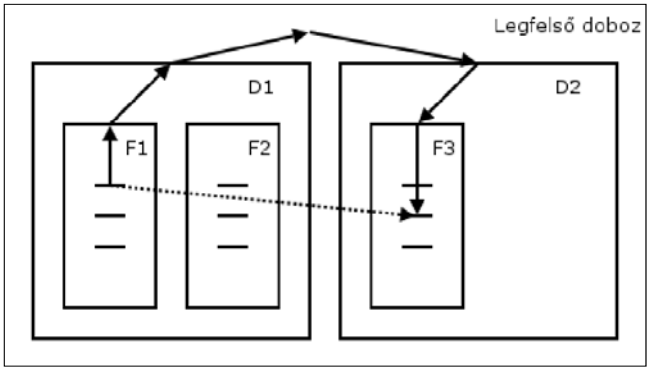
Tisztán doboz-struktúráltnak vagy SP-képesnek nevezünk egy rendszert, ha a következő feltételeket teljesíti:

1. A legfelső szinten egyetlen doboz áll.
2. Dobozai további, hierarchikusan egyenrangú dobozokra bonthatók, vagy végső dobozok, melyek a hierarchia alján állnak.
3. Egy doboz csak önmagán változtathat közvetlenül, az azt tartalmazó (szülője) és az általa tartalmazott (gyermeke) dobozokat specifikált interfészen át éri el.
4. Az interfészek tartalmazzák a függvények nevén, input és output paraméterein kívül a függvény gyártóját azonosító aláírást, továbbá a megvalósított szabvány (ha van ilyen) tanusítványát (ezáltal biztosítva, hogy a függvény valóban azt a funkciót látja el, amit a neve takar).
5. Alapértelmezésben a szülő minden lehetőségét öröklik a gyermekek, de a szülő ezt szűrheti.
6. Ha az interfész-specifikációban ellentmondás van a szülő és a gyerek között, akkor annak a szava a döntő, aki a feldolgozást végezni fogja. Tehát, ha a szülő hívja a gyerek egyik függvényét, akkor a gyereké, ha a gyermek a szülő egy függvényét, akkor a szülőé.

Ha a legfelső szinten SP-képes egység áll, akkor a rendszer SP-képes, tehát képes SPI kezelésére. Ehhez nemcsak az egyes gyártóknak kell a processzorok

kulcsainak védelmét biztosítaniuk, hanem gyártók közötti (és fölötti) összefogásra is szükség van a központ(ok) által kiadott tanusítványok karbantartása miatt. Ez nem jelenti természetesen a fejlesztési vagy technológiai információk cseréjét, az informatikai monokultúra kialakulását, csak a közös érdekek mentén történő egyeztetést (például szabványok, ajánlások megfogalmazása és betartásának ellenőrzése).

Lehetőség van akár több különböző CA használatára is, ha ezek között megfelelő szerződések jönnek létre, melyeknek természetesen anyagi vonatkozásai is lehetnek. Ekkor tulajdonképpen a pénzügyi-gazdasági életben megszokott viszontbiztosítást alkalmazzuk az informatikában. A jelenlegi CA-k esetében ezt keresztanusítványok kiállításával oldják meg.



Az *ábrán* látható, hogy a doboz-struktúrált folyamatok hívási modellje a hálózati rétegek egymásba ágyazásával analóg [4]. Az F1 folyamat látszólag közvetlenül üzen F3-nak, míg a háttérben ezt ellenőrzi F1, D1, a legfelső doboz, D2 és F3 interfésze. Ezért mondhatjuk, hogy az üzenetküldés biztonságos. Az SP-képes rendszert felfoghatjuk úgy is, hogy benne a folyamatok és a hozzájuk rendelt erőforrások egy hierarchikus fába szervezhetőek és az egyes folyamok csak ennek a fának az őket összekötő élein haladva érhetik el másik folyamatok erőforrásait. A fa gyökere a rendszer összes erőforrása, a levelei a folyamatok. Ahhoz, hogy a folyamatok erőforrásigényeikkel a fa ágaira kilépjenek, igazolniuk kell ehhez való jogukat.

4.2. A doboz-struktúráltság előnyei

A doboz struktúrában az összes erőforrás foglalásának lehetősége felülről lefelé adható tovább, tehát az SP-képes processzortól a processzek felé. Így az összes futó folyamat ellenőrzés alatt tartható. Legmagasabb szinten ezt az ellenőrzési lehetőséget az SP-képes processzor nyújtja, közvetlenül fölötte áll azonban a felhasználó, aki tilthatja, illetrve korlátozhatja egyes erőforrások elérését.

Fontos, hogy a dobozok között nincs átfedés, tehát egy erőforrás egy oszthatatlan atomja egyszerre csak egyetlen doboznál lehet. Ha mégis több blokkban van szükség a feldolgozására, akkor az információt a közös ősön át érhetik el. Ezáltal lehetővé válik az erőforrások korlátozott foglalása és felhasználása. Ez a mai rendszerekben még részlegesen van jelen és főleg a programok (illetve felhasználók) tárterületének egymástól való védelmét, valamint a kernel memória védelmét tartalmazza.

Nincs megoldva viszont általában például az időkeretek (processzoridő) védelme. Ennek köszönhetően egy féreg-program képes lehet pusztán osztódással a processzoridő legnagyobb részét lefoglalni, ami egy biztonságos rendszerben megengedhetetlen. Az általunk javasolt rendszerekben az időkeret (processzoridő) is ugyanolyan erőforrás, mint a többi, ugyanúgy kell foglalni, felszabadítani, és minden doboz csak az ő szülőjétől kérheti.

Ahhoz, hogy ne kelljen minden egyes szálnak processzoridőt kérnie magától a felhasználótól, nem kell más, mint a kontroll felületen belül egy újabb doboz, amely megkapja a rendszer szabad processzoridejét. Ha valamely processz többet akar foglalni, mint amennyi processzoridő még rendelkezésre áll, akkor a rendszernek a felhasználóhoz kell fordulnia a kontroll felületen keresztül, hogy döntse el, hogy a kérdéses folyamat kaphat-e a többi processz kárára többleterőforrást. Ezáltal zavartalan működés mellett biztosítható az összes erőforrás felügyelete és indokolt esetben elvétele.

5. Processzor és periféria az SP környezetben

Az informatikában a feldolgozásban résztvevő eszközöket két csoportba szokás sorolni: processzorok és perifériák. A perifériák szolgáltatják, tárolják és jelenítik meg az adatokat, a processzor(ok) végzi(k) a tényleges feldolgozást. Az SP környezetben ez a szétválasztás sokkal jelentősebbé válik, mivel biztosítani kell az információhoz való korlátozott hozzáférést. Erre csak az SP-képes processzorok és SP-képes perifériák képesek, a nem SP-képes eszközökkel való kommunikáció tehát nem minden esetben engedélyezett.

5.1. Processzor

Minden olyan egység, amely információ feldolgozására képes (azaz Turing-gépnek minősül), és részt vesz a feldolgozásban, szükségképpen SP-képes kell legyen. Azaz egyetlen SP-képes processzor sem adhat át SP igényű információt nem SP-képes processzornak, különben rés keletkezik a biztonsági hálón. A processzoroknak ezért igazolniuk kell egymás felé (a központtól kapott tanusítványukkal), hogy ők képesek SP-re, illetve az információnak biztonságosan (titkosítva) kell mozognia a két processzor között. A titkosítás történhet például nyilvános kulcsos rendszerben, vagy bármilyen más, azonosítást és titkosságot garantáló rendszerben. Nem SP-képes processzor csak olyan információt kaphat, amely nem igényel SP-et.

5.2. Periféria

Periféria minden egység, mely a processzorral kapcsolatba kerülhet, annak információt szolgáltat, vagy attól információt kér illetve fogad. A perifériák lehetnek

SP-képesek vagy nem SP-képesek; ameddig az SPI információ *feldolgozása* SP-képes processzorokban történik, addig nem jelent problémát egy vagy több nem SP-képes eszköz. Fontos azonban, hogy az információ *kódoltan* kerüljön a nem SP képes perifériákra (például winchester, hálózati eszköz), és ezt a kódot csak a címzett tudja dekódolni.

Ha a periféria tárolja az adatot, vagy abból bármilyen módon kinyerhető az, akkor nem elég, hogy a periféria interfésze SP-képes, a tárolt adat is titkosított kell legyen, hogy az SP környezet határfelületén belül maradjunk. Összefoglalva: SP-képes perifériából csak a címzett SP-képes eszköz nyerheti ki az információt még fizikai eszközök felhasználásával is, azonban nem SP-képes perifériák is jelen lehetnek a rendszerben mindaddig, amíg ezekben nem történik feldolgozás.

5.3. Amit a dobozstruktúrált feldolgozás sem képes garantálni

Mint minden technológia, az SP is csak egy adott környezetben, adott feltételezések fennállása mellett nyújt garanciákat; ezért meg kell határozni azokat a kockázati tényezőket, amelyek kívül esnek a SP által vizsgált és megoldott problémák körén.

Ezek a következők:

– A központ kulcsának visszafejtése (minimális kockázat);
– A központ hitelességének megkérdőjelezhetővé válása (minimális kockázat);
– Kvantumprocesszorok vagy bármilyen más, NP-teljes problémákat belátható idő alatt megoldó eszközök elterjedése (egyelőre nem várható, jól előrejelezhető az áttörés);
– Magát SP-képesnek mondó, de valójában nem SP-képes processzorok tanusítványszerzése (kis kockázat megfelelő processzor-ellenőrzés mellett);
– Magát SP-képesnek mondó, de valójában nem SP-képes perifériák tanusítványszerzése (nagyon kis kockázat megfelelő periféria-ellenőrzés mellett);
– Interfész és jogi lehetőségek (licensz) eltérése (ez esetben kívánatos olyan állami szintű jogi szabályozás, mely a licensznek megfelelő interfész készítését írja elő);
– A megjelenítési rétegből (például képernyőről, hangkábelről) nyert információ újradigitalizálása, és visszahelyezése az informatikai környezetbe, ezáltal SPI nem SP-képes környezetbe juttatása titkosítatlanul (egyedi megítélést igényel, de az információ minősége az újradigitalizálás következtében valószínűleg romlik, információ vész el).

6. Összefoglalás

Cikkünk arra vállalkozott, hogy feltárja az információkezelés és -feldolgozás jelenlegi módszerei és az informatika fejlődése következtében megjelenő új követelmények közötti ellentmondásokat. Megvizsgáltuk az új kihívásokra adott válaszokat, illetve a rendelkezésünkre álló eszköztárat.

A mai, csak egyes kiválasztott részproblémák megoldására használatos rendszerek helyett egy új paradigma, a *biztonságos információfeldolgozás* (secure processing) alkalmazását tartjuk célravezetőnek, mivel ezáltal az információforgalom valamennyi szereplőjének érdekei biztonságosan és gazdaságosan teljesíthetőek. A doboz-struktúrált feldolgozási környezet cikkünkben bemutatott jellegzetességei lehetővé teszik, hogy a következő évtizedek infokommunikációs rendszereinek meghatározó eleme lehessen.

Meg kell még jegyeznünk, hogy bár az SP széleskörű elterjedése sem tudja teljesen kizárni az emberi tényezőben megjelenő kockázatot és azokat az egyéb veszélyforrásokat, amelyek az informatikai technológián kívülről fenyegetik a biztonságos információfeldolgozást (például fizikai sebezhetőség vagy biztonsági szempontból nem megfelelő ügyviteli-rendszabályi eljárások), de ezek némelyikének a kockázatát csökkenti, illetve korlátozza az ezen támadások által okozható kár mértékét.

Irodalom

[1] RFC 1321, http://www.faqs.org/rfcs/rfc1321.html
[2] Trusted Computing FAQ, http://www.cl.cam.ac.uk/users/rja14/tcpa-faq.html
[3] Buttyán Levente, Vajda István: Kriptográfia és alkalmazásai; Typotex, 2004
[4] Andrew S. Tannenbaum: Computer Networks, 3rd Ed.; Prentice-Hall Inc, 1996.
[5] Terry Ritter: Ciphers By Ritter, http://www.ciphersbyritter.com/
[6] Conference on Communications and Multimedia Security 2005, http://cms2005.sbg.ac.at/
[7] J. Marchesini, S. Smith, O. Wild, R. MacDonald: Experimenting with TCPA/TCG Hardware, Computer Science Technical Report TR2003-476
[8] B. Arbaugh: Improving the TCPA Specification, IEEE Trans. on Computer Science, 2002 August (Vol. 35, Issue 8), http://ieeexplore.ieee.org/xpl/abs_free.jsp?arNumber=1023792