MODULÁRIS VÉDELMI MODELL KIDOLGOZÁSA XML ADATOKHOZ

MODULAR SECURITY MODEL FOR XML

Kovács László, Gergely Ádám*, Pribula Péter Balázs*,Török András**

ABSTRACT

The XML file format provides a flexible storage for data. The XML standard uses a semi-structured text format with text annotations. The XML format enables an automated processing of the file content. There are many applications areas of the XML files and there are many efficient processing standards for XML. Despite the large popularity, only few security features are available for XML files. The usual option is to protect the whole XML file as a unit. The second option is to use the XML Security framework where a subtree in the file can be encoded with a symmetric key. The proposed security model enables a segmentation of the XML tree where the same segment may have different access attributes for the different users. The model corresponds to a multi-user multi-object security system.

1. XML ADATMODELL

Az XML (kiterjeszthető jelölő nyelv) [5] alapvető tulajdonsága, hogy az adat és a meta adat együtt kerül tárolásra. Az XML tárolás egyik fontos jellemzője, hogy az adatok mellé társíthatóak a kiegészítő információt hordozó annotációk is. A jelölő elemekre épülő szöveges formátummal tetszőleges hierarchikus szerkezet alakítható ki. Az XML modell szerkezetét szabályozó modell az XDM [4], mely az XML dokumentumot egyetlen gyökér elemet tartalmazó hierarchiának tekinti. A fában néhány csomóponttípus értelmezett: jelölő elem, annak attribútuma, névtér, megjegyzés és szövegelem. Az XDM grafikus elemekkel jelöli a különböző csomópontokat és azok kapcsolatát. Egy XDM csomópont lehet többek között elem vagy attribútum. Az elem csomópontnak több típusa is létezik. Ezek szerint megkülönböztetünk üres elemet, szöveg elemet, gyerek elemet és vegyes elemet. Attribútumnak nevezzük a megnevezés és érték párosát. Az attribútumok az elemek tulajdonságait határozzák meg. Egy elemhez tetszőleges számú attribútum rendelhető. Az XML struktúra fő előnyi közé tartozik a rugalmas struktúra; az automatikus feldolgozhatóság; a jelentés és érték együttes tárolása; valamint a platform függetlenség.

Az XML széles körben elterjedt az információs rendszerekben. A leggyakoribb felhasználási területek közé tartozik a konfigurációs adatok tárolása; alkalmazási adatok perzisztens tárolása univerzális formában; valamint az adatok továbbítása az Interneten heterogén csomópontjai között. Az XML jelentősége az univerzális tárolásban és az automatizált, tartalom alapú feldolgozhatóságban rejlik. Az XML mint közvetítő nyelv alkalmazásával automatizáltan konvertálható át egymásba az adattábla és a vektorgrafikus kép; a Word dokumentum és relációs adattábla.

Az XML nyelvnek az előnyei mellett néhány negatív vonással is számolnunk kell. Az egyik ilyen jellemző a bőbeszédűség, a nagyobb terjedelem. Ez a bőbeszédűség a szemantikai annotáció esetében hasznos, de az adatátvitel gyorsasága szempontjából hátrány. A további negatív tulajdonság az adatelemek fán belüli keresésének lassúsága és az adatok nagyfokú nyíltsága. Ez utóbbi az adatok védelmi mechanizmusához kapcsolódik.

Az adatok hozzáférés elleni védelmének biztosítása az operációs rendszerek alapvető szolgáltatásai közé tartozik. Az adatvédelem több szinten valósul meg. Az egyik szint a felhasználók azonosítása (authentikáció). A második szint az adatokra vonatkozó műveletek engedélyezése vagy tiltása (authorizáció). A védelmi rendszer további elemei közé tartozik a következtetés ellenőrzés; az adatok elrejtése és az elvégzett műveletek naplózása. Ezen funkciók a hagyományos, adatbázis alapú tárolási rendszerekben megszokott szolgáltatások. Az egyes objektumok és a szubjektumok közötti kapcsolat leírására alkalmazott reláció hozzáférési mátrixszal (acces matrix) írható le, melynek sorai az objektumok, oszlopai pedig az egyes szubjektumok. A mátrix celláiban pedig az engedélyezett műveletek találhatóak. Az objektumok és a szubjektumok is egyedi azonosítóval rendelkeznek. Az XML rendszer esetében napjainkban még sokkal kevesebb védelmi szolgáltatás áll rendelkezésre. Az XML-ben tárolt adatok védelmére leggyakrabban az állomány szintű OS védelmet használják. Ezen a szinten a teljes XML dokumentum egyetlen egységnek tekinthető. Napja-

* *Miskolci Egyetem, Általános Informatikai Tanszék*

inkban egy másik út is kínálkozik, az XMLSecurity csomag, melynek segítségével a dokumentum egyes részei egy megadott kulcsot alkalmazva titkosítható. Ez a fajta titkosítás már lehetőséget ad a dokumentum fregment szintű védelemre, viszont minden fregment csak egyetlen felhasználóhoz rendelhető. Nagyobb rugalmasságot adnak azok a rendszerek, ahol egy fregmenst több felhasználó is tud párhuzamosan használni.

2 .XML VÉDELMI MECHANIZMUSOK

A XML dokumentumok állomány szintű védelménél az egyik leghatékonyabb rendszer a Windows Encrypted File System [3], mely egy kibővített NTFS fájlrendszernek tekinthető. A rendszerben minden egyes felhasználó egy úgynevezett saját SID-vel (System ID) rendelkezik. Az első EFS használatkor az operációs rendszer létrehoz egy bizonyítványt, amelyet a Registry nevű mappában tárol. A bizonyítvány által létrejön automatikusan egy kulcspár, amely egy aszimmetrikus titkosítási eljáráshoz alkalmazható. A rendszer minden egyes fájl titkosításakor generál egy véletlenszerű 128 bit hosszúságú számot, amelyet File Encrypt Key-nek (FEK) nevezünk. Ezzel a FEK kulccsal egy szimmetrikus kódolással titkosítja a fájlt. Ezután a rendszer aszimmetrikus módon kódolja a File Encrypt Key-t a felhasználó publikus kulcsának segítségével. Ennek az aszimmetrikus kódolásnak az eredménye lesz a Data Decryption Filed, a DDF. Az operációs rendszer a DDF-et tárolja a titkosított fájl header részében. A következő lépésben a FEK kódolásra kerül a Data Recovery Agent nyilvános kulcsával is. Ezáltal kapjuk meg a Data Recovery Field-et, amely szintén a fájl header részébenében kerül letárolásra. Amennyiben a felhasználó hozzá akar férni a titkosított fájlhoz, úgy a bizonyítványban tárolt privát kulcsával vissza kell fejtenie a DDF-et. A visszafejtés utána megkapja a File Encrypt Key-t. A felhasználó az FEK segítségével visszafejtheti a titkosított dokumentumot

Az egyes XML fregmentek titkosítására szolgál az XML Security szabvány [7]. Az adatrejtés megvalósításához szimmetrikus kulcson alapuló titkosítási mechanizmust használnak DES, DES3 vagy akár AES algoritmus segítségével. A Java forráskódban implementálásra került egy olyan program csomag, amely képes elemekre szeparálni egy XML dokumentumot, ezáltal tetszőleges részt jelölhetünk ki és titkosíthatunk egy XML dokumentumon belül.

A titkosítási eljárást a W3C szervezet a http://www.w3.org/Encryption/2001 néven szabványosította. A fő címke az EncryptedData, amely egy összetett struktúrából áll [1]. Az EncryptionMethod segítségével adható meg az elemtitkosítás algoritmusa, amely szimmetrikus elven működik. A KeyInfo titkosított formában tárolja a titkosító kulcsot. Fontos, hogy titkosítva legyen a kulcs, hiszen ennek hiányában bárki visszafejthetné az adatokat. A titkosított kulcsot a feladónak más csatornán kell eljuttatnia a címzetthez. Egyetlen kulcsról tároló információt az EncryptedKey. Az EncryptedKey két részből tevődik össze: a CipherData és a CipherValue részekből. Itt kerül elhelyezésre az elem tartalma titkosított formában. A struktúrában a következő rész szintén az EncryptionMethod, amely most a szimmetrikus elven titkosított kulcs aszimmetrikus titkosításáért felel.

3. MODULÁRIS VÉDELMI MODELL XML-RE

A javasolt védelmi modellben az adott x XML dokumentumot logikai szegmensekre bontjuk. Minden szegmens egy azonos védelmi tulajdonságokkal rendelkező terület. Minden szegmens egy részfából képezhető annak zérus vagy néhány részfájának eltávolításával. Minden szegmenshez pontosan egy gyökér elem rendelhető. A gyökér elemek relációja alapján a szegmensek között is bevezethető egy hierarchia:

$$S_1 > S_2, \text{ ha } r(S_1) > r(S_2);$$

ahol a S_i a szegmensek jelöli és 'a > b' reláció teljesül, ha az a csomópont a b csomópontnak egy őse.

Az egyes szegmensekhez a hozzáféréseket egy ACL alapú mátrixban tárolhatjuk. A hozzáférési műveletek halmazában két műveletet definiálunk: olvasás (R) és írás (W). Minden felhasználóhoz egy saját ACL lista tartozik. Egy adott felhasználó esetén az egyes kapcsolódó szegmensekre vonatkozó védelmi beállítások nem lehetnek függetlenek egymástól. Ugyanis az alábbi ellentmondásokat meg kell akadályozni:

- a felső szegmensben elvégzett módosítás nem törölhet ki nem látott adatokat alsó szegmensekből

- egy alsó szegmenshez való hozzáférési jog esetén látni kell az odavezető utat.

Ezen irányelvek miatt az alábbi integritási szabályokat kell bevezetni az $S_1 > S_2$ szegmensek vonatkozásában:

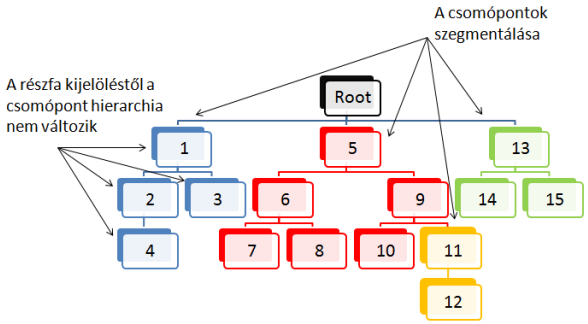
$w(S_1) \Rightarrow w(S_2)$

$w(S_2) \Rightarrow r(S_1) \mid w(S_1)$

$r(S_2) \Rightarrow r(S_1)$

A megadott moduláris védelmi rendszer az egyik implementációs modellben az XML Security szabvány fölé épül. Mivel egyazon XML dokumentum a különböző felhasználóknál különböző szegmensekre bontható, a logikai szegmensek egy finomabb felbontású fizikai szegmentációt eredményeznek. Az egyes fizikai szegmensek az XML Security védelmi modell eszközrendszerével kerülnek titkosításra.

Mivel a fizikai szegmensek transzparensek a felhasználók számára és egy szegmenshez csak egy titkosítási kulcs tartozhat, a fizikai szegmensekhez tartozó kulcsok nyilvántartását egy külön modul végzi. E külön modul feladata a felhasználókhoz rendelt kulcsok és a fizikai szegmensekhez tartozó kulcsok összerendelése. Ez az összerendelés egy külön táblázatban történik. Mivel arra törekszünk, hogy az XML dokumentum maga hordozza a hozzáféréshez szükséges szabályozást, a dokumentumot szükséges átalakítani



1. Ábra XML dokumentum szegmensei

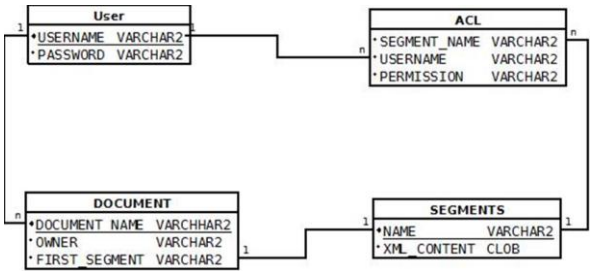
Egy ilyen dokumentum két részből áll: a titkosításhoz használt információkból (User Meta Table, UMT) és a titkos dokumentumból (Secure Document, SD). Az UMT tartalmazza a felhasználókhoz tartozó szegmenseket tikosító szimmetrikus kulcsot. Minden felhasználóhoz tartozik egy bejegyzés, amit a felhasználó a saját nyilvános kulcsával titkosít. így megakadályozva az illetéktelen hozzáférést. Amennyiben egy szegmensnek több tulajdonosa is van, a szegmens titkosító kulcsát minden tulajdonos bejegyzésébe meg kell jeleníteni.

Önhordozó architektúra esetén a szegmenseket egyesével titkosítjuk. Amennyiben egy felhasználó jogot kap egy mélyebben elhelyezkedő, másik szegmensben található szegmensre, így annak perem elemeit látnia kell. A perem elemek magukba foglalják a gyökérig vezető utat valamint a testvér elemeket. A perem elemekből elég csak a tag részt látni, nincs szükség a tartalmára is.

A felhasználók azonosítása aszimmetrikus kulcsú PKI mechanizmuson alapszik. A PKI (Public Key Infrastucture), a nyilvános kulcsú infrastruktúra lehetővé teszi egy nem védett hálózaton belüli biztonságos kommunikációt. A kommunikációs felek digitális tanúsítványok segítségével biztosítják a megbízhatóságot. A PKI a nyilvános kulcsú titkosítási módszeren alapszik. Technikai kivitelezése egy kulcspár tárolásával történik, melyben a kulcspárok egy összetartozó nyílt és egy titkos kulcs összerendelését tartalmazza. Egy PKI technológiát biztosító szervezetnek minden esetben garantálnia kell a felhasználók számára a titoktartás, az integritás, a hitelesség és a megbízhatóság elvét. [9].

A rendszer másik tesztelt implementációs környezete az adatbázis központú megvalósítás. Ekkor a dokumentumok szegmenseit egy táblában tároljuk. A szegmensek közti kapcsolatot egyszerű PCR elv alapján oldhatjuk meg. A dokumentum fáját indorder bejárva a szegmenseket letároljuk a táblába. Amennyiben egy szegmens (S1) egy másikat (S2) tartalmaz, az alacsonyabb szintű (S2) szegmensnek egy új id-t adunk, majd a magasabb szintű (S1) szegmensbe elhelyezünk egy linket, amely az S2-re mutat.

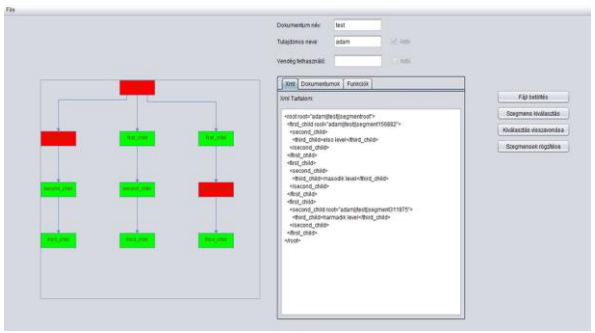
A védelmi rendszernek nyilván kell tartania a felhasználókat, a dokumentumokat, és az ezekhez tartozó szegmenseket, valamint a szegmensekhez tartozó hozzáférési listát. Ebben az esetben az adatbázis táblákban tárolódik minden információ, közte az XML dokumentumok tartalma is. Emiatt nincs szükség az XML dokumentumrészek titkosítására és nincs szükség a kulcsok kezelésére sem. A felhasználók jogosultságát, a logikai és fizikai szegmensek kapcsolatát az adatbázis tárolja. A relációs modell a jelenlegi információk a következőképpen néz ki.



2. ábra: Védelem adminisztráció adatbázis sémája

4. MINTARENDSZER ARCHITEKTÚRA

Az elkészült mintarendszerben lehetőség nyílik a XML dokumentum logikai szegmenseinek GUI alapú kezelésre. A kezelő felületet látható a fa gráf és annak XML forrása. A fa gráf csomópontjai az elemek, a szegmensek gyökérelemei külön színnel vannak jelölve. A kezelő felületen a XML dokumentum logikai szegmensek bontása mellett elvégezhető a felhasználók adminisztrálása és az ACL lista előállítása is.



3. ábra. Védelmi modul kezelő felülete

A kódolt XML dokumentum váza az alábbi elemeket tartalmazza:

```
<?xml?>
 <secured-document-root>
  <user-meta-table>
  […]
  </user-meta-table>
  <secured-document>
    <document>
       […]
   </document>
  </secured-document>
 </secured-document-root>
```

A felhasználói adatokat tároló rész sémája:

```
<user-meta-table>
  […]
  <user publicKey=”12345678901…789”>
   <username>UserName</username>
   <dataEncryptionKey
    segmentId=”12345”>qE…M=
  </dataEncryptionKey>
    […]
  </user>
  […]
</user-meta-table>
```

A titkosított XML dokumentum csak a keretrendszeren keresztül fejthető vissza.

5. ÖSSZEFOGLALÁS

Az XML dokumentum kezelés egyik kevés szolgáltatást nyújtó szelete az adatok hozzáférés-védelmi rendszere. A kidolgozott moduláris, több felhasználó párhuzamos hozzáférését is támogató védelmi modell új lehetőséget biztosít az XML adatkezelés területén.

KÖSZÖNETNYÍLVÁNÍTÁS

A bemutatott kutató munka a TÁMOP-4.2.1.B-10/2/KONV-2010-0001 jelű projekt részeként az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

IRODALOMJEGYZÉK

[1] BARTLETT, R. G. ; COOK, M. W.; XML Security Using XSLT, Proc. of the 36th Hawaii International Conf. on System Sciences, Track 4, Vol 4, pp. 122.2, 2003

[2] CHANG, TAO-KU; HWANG, GWAN-HWAN: To secure XML documents with the extension function of XSLT, Software—Practice & Experience - Research Articles, Vol. 36 Issue 5, 2006, pp. 539 - 555.

[3] Sosinsky, Barrie: Microsoft Windows Server 2008: Implementation and Admisintration, Sybex, 2008

[4] HAROLD, ELIOTTE RUST: Hatékony XML, Kiskapu Kiadó, 2006

[5] BRADLEY, NEIL : Az XML kézikönyv , SZAK Kiadó, 2005

[6] RAVI S. SANDHU : Relational Database Access Controls using SQL, Handbook of Information Security Management, 1994

[7] DOUMEE, BLAKE , XML Security: RSA Security’s Official Guide, McGraw Hill Professional, 2002

[8] OBAIDAT, MOHAMMAD; BOUDRIGA, NOUREDDINE: Security of e-Systems and Computer Networks, Cambridge Univ. Press, 2007

[9] GUPTA, KAILASH N.; AGARWALA, KAMALESH N.; AGARWALA, PRATEEK AMAR: Digital Signature: Network Security Practices, PHI Learning Pvt. Ltd., 2005