AJATEMPLID digitaaldokumentidel

Ahto Buldas,
Helger Lipmaa

Aja templisambad sirguvad, suursugused nagu saatus.
(A. Alliksaar)

Seoses tehnikasajandi lõppemise ning informatsioonisajandi algamisega on üle kogu maailma täheldada nn komputeriseerumisprotsesse. Näost–näkku suhtlemise asemel istutakse jututubades, ajalehti loetakse veebist ning eriti eesrindlikud kodanikud ostavad arvuti abil ka oma hommikuse piima. Järjest rohkem kasvab ka Interneti kaudu tehtavate äritehingute arv. Nagu omal ajal, kui mindi naturaalmajandamiselt üle vana hea kulla peale, on ka nüüd, üleminekul kullalt bittidele, vaja täiesti uue infrastruktuuri väljatöötamist. Kohendamist vajavad inimeste harjumused ning muutuvad nende vajadused (alamklassid enam ei taha...) ning muudatusi vajavad ka seadused (ülemklassid enam ei saa...). Nagu alati, võib ühiskonda muuta mitmel viisil, näiteks seadusliku menetluse läbi või vägivalda kasutades. Üks on aga kindel — pea alati on muudatuste aeg ka segaduste ja kaose aeg. Järgnevas tutvustame mõningaid rahumeelseid vahendeid kaoste ja segaduste vältimiseks järgmisesse sajandisse minekuks.

Pikaealised digitaaldokumendid

Tuleb hetk ja koos hetkega see,
kes kõik ehtsalt võltsitu teeb võltsilt ehtsaks.
(A. Alliksaar)

Üks kõige tõenäolisemaid ja aktuaalsemaid võimalusi korratuse ja kaose tekkimiseks bitiühiskonnas on pikaealiste juriidilise jõuga elektrondokumentide kasutuselevõtmine. Valdavalt on elektrondokumentide haldussüsteemide turvameetmete projekteerimisel lähtutud turvalise sõnumivahetuse süsteemidest (näiteks turvaline elektronpost), kus tegeldakse üksnes lühiealiste sõnumite turvamisega. Turvalises sõnumivahetuses kasutatav turvatehnika pole piisav pika elueaga dokumentide halduseks. Oletame, et Alice'i privaatvõti D_A pole teatud hetkest alates enam privaatne, st keegi teine on saanud selle võtme oma valdusse. Kui selline fakt ilmsiks tuleb, siis klassikaline lahendus on võtme kandmine kõigile kasutajatele kättesaadavasse tühistuslisti (*revocation list*). Kui Bob saab sõnumi $D_A(X)$, siis kontrollib ta muuhulgas ka seda, kas võti D_A pole kantud tühistuslisti. Kui on, siis ei saa ta sõnumi sisu usaldada. Sellise lahenduse lihtsus ja turvalisus on kahjuks näiline. Kui X on näiteks laenuleping, mis kinnitab, et Alice on võtnud pangast pikaajalist laenu, siis peale võtme D_A tühistuslisti kandmist muutub kaheldavaks ka laenulepingu ehtsus, ehkki see sõlmiti tunduvalt varem kui võti D_A tühistati. Arvake ära, kas Alice on peale laenu saamist huvitatud oma privaatvõtme hoolikast hoidmisest?!

Toodud näide peaks ilmekalt selgitama vajadust mehhanismide järgi, mis võimaldaksid tuvastada ja ka tõestada, millal mingi elektrondokument loodi.

Usaldatav kolmas osapool

Kõrges rohus suures rahus
tillukene mees
kuulab aja voolamist, ta pilk on
pilvedes.
...
Kõik küsivad siis, kui vaja
üht küsimust
üle aja.
(J. Viiding)

Kõige lihtsam viis dokumendi loomisaja tõestamiseks on kasutada usaldatavat kolmandat osapoolt, kes registreerib talle esitatud dokumendid ja lisab neile nn digitaalse ajatempli, st metaosa, mille abil saab tõestada dokumendi signeerimise aega. Ajatempli teenuse pakkujal (*Time–Stamping Service* — TSS) on oma privaatvõti D_{TSS}, millele vastava avaliku võtme abil saab ajatemplite ehtsust kontrollida. Samuti on vajalik, et TSS–i valduses oleks kell, mida kõik kasutajad usaldaksid kui ajaetaloni. Ajatempli küsimine toimub järgmise protokolli abil:

- Alice saadab TSS–le sõnumi Y, millele soovib saada ajatemplit. Näiteks erijuhul võib Y olla sõnumi X digitaalsignatuur, st $Y=D_A(X)$.
- TSS lisab sõnumile momendi aja t, signeerib liitsõnumi ja saadab Alice'le sõnumi $S=D_{TSS}(Y,t)$, mida võibki kasutada kui signeeritud sõnumi ajatemplit.

Kui TSS–i avalik võti on kõigile teada, siis on võimalik lihtsalt kontrollida, millal dokument TSS–ile esitati. Kirjeldatud süsteemi võib võrrelda juba ammu kasutatava nipiga, kus sõnum saadetakse lahtisel postkaardil iseenda aadressil, et saada sinna postiteenuse templit, kus teatavasti on ka aeg. Postiteenus tegutseb seljuhul TSS–i rollis. Sellel lihtsal ajatempli süsteemil on aga kaks olulist puudust:

- Pole selge, mis saab siis, kui võti D_{TSS} pole ühel hetkel enam privaatne. Tekkinud olukord on analoogiline sellega, mida kirjeldati eelmises peatükis. Sellest hetkest alates muutub kõigi seni välja antud ajatemplite ehtsus küsitavaks.
- TSS peab olema absoluutselt usaldatav, st kõik kasutajad peavad uskuma, et TSS ei anna tahtlikult, näiteks kasu saamise eesmärgil, välja võltsajatempleid.

Arvestades sellega, kuidas asjad tegelikus elus käivad, on nendest põhjustest rohkem kui küll, et teha vaadeldud süsteemi praktiline kasutamine võimatuks.

Linkimine ja ajatemplite usaldatavus

Sed quis custodiet ipsos
Custodes?[1]
(Juvenalis 47–138 a d)

Ajatemplite süsteem paberdokumentide jaoks ühe asutuse või orga-

[1] Kuid kes hakkab valvama valvureid?

nisatsiooni piires on juba ammu välja mõeldud. Selleks on dokumentide register, st vihik, milles on nummerdatud read dokumentide registreerimiseks. Kui vihikut täidetakse järgimööda, siis on hiljem väga raske sinna dokumente lisada ilma nähtavaid jälgi jätmata. Sama ideed saab kasutada ka elektrondokumentide korral. Ridade järgnevus registris tagatakse räsifunktsiooni H kasutamisega. Ajatempli teenuse pakkuja (TSS) on kohustatud pidama välja antud ajatemplite registrit, mille korrektset täitmist saavad kõik süsteemi kasutajad kontrollida. Ajatempli väljaandmise protokoll oleks sel juhul järgmine:

- Alice saadab TSS–ile mingi digitaalse infokogumi Y.
- Teades, et Alice'i taotlus on järjekorranumbrilt n–s, leiab TSS sõnumilühendi
  $H_n = h(n,ID_A,t,Y)$,
  kus ID_A on Alice'i identifikaator, t on momendi aeg ja h on mingi räsifunktsioon, mis ei tarvitse olla sama mis H. Seejärel arvutab TSS nn linkimisinfo
  $L_n = H(H_n, L_{n-1})$
  ja saadab Alice'ile tagasi sõnumi $(n,ID_A,t,L_{n-1},D_{TSS}(L_n))$, mida kasutatakse kui ajatemplit.

Lisaks ajatempli väljaandmise teenusele võib iga klient küsida TSS–i käest ka mistahes H_n–i ja L_n–i, mis on jäädvustatud TSS–i poolt. Kasutatavate räsifunktsioonide ühesuunalisuse tõttu on väga raske tekkinud ahelat võltsida ja sinna uusi ajatempleid juurde lisada mujale kui ahela lõppu. Ei ole enam eriline katastroof, kui TSS–i privaatvõti kompromiteerub, sest kogu ajatemplite süsteemi turvalisus on seotud pigem kasutatavate räsifunktsioonide turvalisusega. Räsifunktsiooni ühesuunalisuse omadust võib jämedalt väljendada järgmiselt:

- Kui $H(X)$ oli teada mingil ajahetkel t, siis suurus X pidi olema teada enne ajahetke t.

Seda omadust võib väljendada ka nii, et kui Alice usub, et X on värske ajahetkel (pole teada kellelegi enne ajahetke) t, siis ta usub, et ka $H(X)$ on värske ajahetkel t. Ajatemplite vahelist ühesuunalist andmesõltuvust iseloomustab järgmine diagramm:

…	L_{n-2}	L_{n-1}	L_n	L_{n+1}	…
…	H_{n-2}	H_{n-1}	H_n	H_{n+1}	…

Kui Alice usub, et L_{n+1} eksisteeris ajahetkel t, siis ta usub ka, et L_n, H_n, L_{n-1}, … eksisteerisid ajahetkel t. Kui lisaks sellele Alice usub, et L_{n-1} oli värske ajahetkel t', siis ta usub, et ka L_n, L_{n+1}, … olid värsked ajahetkel t'. Seega ta usub, et L_n moodustati millalgi ajahetkede t' ja t vahel.

Nagu võis tähele panna, ei läinud eelnevas arutelus kusagil vaja TSS–i privaatvõtme turvalisust ega ka ühtegi eeldust TSS–i korrektse käitumise kohta, mistõttu võib järeldada, et saadud süsteem on tunduvalt töökindlam ja raskemini rünnatav kui eelmises punktis kirjeldatud usaldatavat kolmandat osapoolt nõudev süsteem.

Räsifunktsioonide turvavajadused

Koledad kollisioonid kollitavad kolisedes nende kooseksistentsi kohal.
(A. Alliksaar)
Kuskil on üks aeg,
kus kõik on väga vale.
(A. Aule)

Nagu juba öeldud, sõltub kogu ajatemplite süsteemi turvalisus kasutatavate räsifunktsioonide h ja H turvalisusest, kusjuures oluliselt erineval määral. Funktsioonist H sõltub dokumentide ajaline järgnevus, funktsioonist h aga tembeldatavate dokumentide seos vastavate ajatemplitega.

Mis juhtub siis, kui ühel hetkel hakkab olema lihtne leida funktsiooni H kollisioone, st kui antud X –i korral on lihtne leida teist argumenti $X' \neq X$, nii et $H(X)=H(X')$? Selgub, et siis saab võltsida mistahes eelnevaid ajatempleid ja nende ajalist järjekorda. Näiteks, kui TSS soovib võltsida n–ndat ajatemplit, siis tuleb tal esmalt arvutada enda jaoks sobilik
$H_n = h(n,ID_n,t,Y)$.
Edasi on kaks võimalust. TSS võib leida mingi H_{n+1}, nii et
$L_n = H(H_n,L_{n-1})$
$L_{n+1} = H(H_{n+1},L_n)$.
Teine võimalus n–ndat ajatemplit võltsida on leida L_{n-1} ja H_{n-1}, nii et
$L_{n-1} = H(H_{n-1},L_{n-2})$
$L_n = H(H_n,L_{n-1})$.
Loomulikult pole selline võltsimine võimalik, kui funktsioon H on ühesuunaline. Paneme tähele, et funktsiooni H murdmisel tuleb kummalgi juhul võltsida vähemalt üks linkimisinformatsioon ning seega võib murdjaks olla vaid TSS ise, mitte aga kurikael Edgar. Erinev lugu on siis, kui räsifunktsioon h lahti murtakse. Sellisel juhul saab leida sobiva neliku (n,ID_n,t,Y), nii et $h(n,ID_n,t,Y)=h(n,ID_A,t,Y)$, jättes puutumata linkimisinformatsioonid. See tähendab, et funktsiooni h murdumise korral võib ajatemplite süsteemi rünnata suvaline kolmas, küllalt nutikas, osapool (muuhulgas ka TSS ise) ning funktsiooni H turvalisusest pole enam kasu. See on üks põhjusi, miks peab räsifunktsioon h olema turvalisem kui H.

Peab märkima, et tänapäeval ei teata ühtegi räsifunktsiooni, mille turvalisus oleks kindel näiteks järgmise 10 aasta jooksul (üks esimesi laiemalt kasutusele võetud räsifunktsioone, MD4, murti lahti umbes 5 aastat pärast leiutamist). On olemas vaid kiired räsifunktsioonid, mille lahtimurdmiseks puudub piisav teadmus (nimetada võiks funktsioone SHA–1 ja RIPEMD–160) ning aeglased räsifunktsioonid, mille turvalisus põhineb mõnel hästituntud raskesti arvutataval matemaatilisel funktsioonil (näiteks diskreetsel logaritmil). Kuid ka viimaste räsifunktsioonide turvalisuses ei ole veendumust, kuna järjest mõeldakse välja uusi ja kiiremaid diskreetse logaritmi arvutamise algoritme.

Lineaarse linkimisviisi puudused

Kõigi teede pikkus ajas on võrdne
(K. Ristikivi)

Eeltoodud linkimisskeemi juures oli iga ajatempel seotud ühe, talle vahetult eelneva, ajatempliga. Nagu nägime, tagas toodud meetod küll oluliste turvanõuete täitmist, kuid samas on tegu väga ebapraktilise skeemiga, millel on kaks olulist puudust:

- Vajadus, et ajatempli teenuse pakkuja säilitaks mälus terve ajatemplite ahela. Kui oletada, et et reaalselt funktsioneerivas süsteemis on ajatempleid *ca* 10^{11}, siis on selge, et mälust kipub puudu jääma.
- Ajatemplite võrdlemisel tehtav töö on võrdeline ajatemplite järjekorranumbrite vahega, tehtava töö hulga ülemtõke on $O(n)$. See teeb aga ajatemplite võrdlemise äärmiselt ebapraktiliseks, kuna üheainsa ajatempli kontrollimiseks tuleks halval juhul teha sama palju tööd kui TSS on teinud kogu oma eksistentsi vältel.

Üks võimalus seda olukorda parandada on jagada TSS–i tegevus

raundideks, mille lõpus väljastatakse koond–ajatempel. Mingis raundis väljastatud ajatempli linkimisinfo sõltub samas raundis eelnevalt väljastatud ajatemplitest ning eelneva raundi koond–ajatemplist. Käesoleva raundi jooksul väljastatud dokumendid paigutatakse täieliku kahendpuu lehtedeks. Puu suvalise sisemise tipu väärtuseks on $H_k:=H(H_L,H_R)$, kus H_L ja H_R on selle tipu vastavalt vasaku ja parema „lapse" väärtused. Sellise skeemi korral peab TSS säilitama vaid koond–ajatemplid $s_n:=H(H_r,s_{n-1})$, kus H_r on selle raundi puu juure väärtus. Kõik ülejäänu antakse kaasa individuaalsetele ajatemplitele. Kui dokumendid räsitakse enne ajatempli löömist, on iga ajatempli pikkus $O(\log n)$, mille saab vähendada konstandiks, kasutades ühesuunalisi akumulaatoreid.

Kahjuks on koond–ajatemplitel ka puudusi, neist märkimisväärseim on võimatus võrrelda ajaliselt ühe raundi jooksul antud ajatempleid. Teise puudusena võib märkida seda, et kliendid peavad enne kehtiva ajatempli saamist ootama raundi lõpuni.

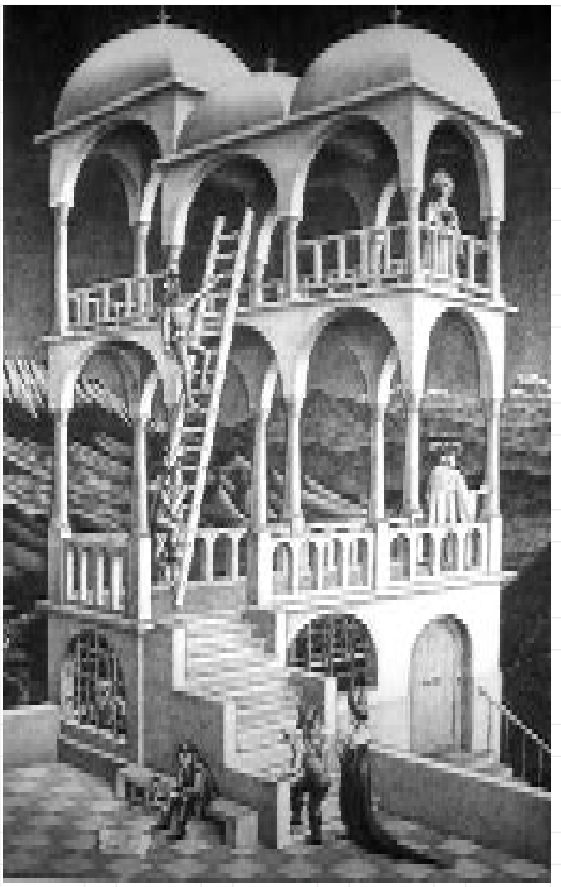
Binaarsed linkimisviisid

Kui palju harusid peab olema jõel,
et ta oleks haruldane?
(A. Alliksaar)

Binaarseks linkimisviisiks nimetame ajatempli arvutamise skeemi, kus iga väljaantud ajatempel pole seotud mitte ainult oma vahetu eellasega, vaid ka veel ühe sobivalt valitud ajatempliga. Linkimisskeemi võib iseloomustada seosega:

$L_n = H(H_n, L_{n-1}, L_{n-f(n)})$,

kus f on mingi sobivalt valitud funktsioon. Näiteks, kui võtta $f(n) = 2$ ord n, kus ord n on suurim selline k, nii et arv n jagub jäägita arvuga $2k$, siis saame ajatempli kontrollimisel tehtava töö hinnanguks $O(\log_2 n)$, mis on oluliselt väiksem kui $O(n)$, kuid siiski liiga suur praktikas rakendamiseks. Osutub aga, et kui valida funktsioon f sobival viisil, siis on võimalik saada hinnanguks $O(\log n)$, mis teeb linkimisviisi juba praktikas kasutatavaks.



Koond–ajatemplid binaarse linkimisviisi korral

Hääletult kukub poolküpseid
päevi aegade ääretust puust.
(A. Alliksaar)

Selgub, et binaarne linkimisviis ei ole kasulik mitte ainult ajatempli kontrollimisel tehtava töö vähendamisel, vaid võimaldab lahendada ka koond–ajatemplite probleemi. On võimalik jagada arvutatud ajatemplite ahel võrdse pikkusega lõikudeks, nii et ühes lõigus olevad ajatemplid sõltuval otseselt üksnes selle sama lõigu ajatemplitest ja eelmiste lõikude viimastest, st koond–ajatemplitest. Sel juhul peavad *on–line* kättesaadavad olema üksnes koond–ajatemplid. Dokumendiga koos säilitatakse ajatemplite (logaritmilise pikkusega) ahelaid selle lõigu (raundi) koond–ajatemplini, kuhu dokument ise kuulub, ning ka eelmise lõigu koond–ajatemplini. TSS võib salvestada vanade raundide ahelad näiteks CD–dele ja hoida neid oma arhiivis, juhuks kui mõni ajatempel kuhugi ära kaob. Ühele CD–le mahub kümnetest miljonitest ajatemplitest koosnev ahel. Funktsiooni f võib aga valida nii kavalasti, et iga kahe dokumendiga kaasas olevate ahelate abil on võimalik koostada ühesuunaline ahel ühe dokumendi ajatemplist teise dokumendi ajatemplini, ja seda isegi siis, kui need dokumendid on tembeldatud ühes ja samas raundis.

Mida Eestis ja mujal tehakse ja mis on lähitulevikuplaanid?

Momendil puudub maailmas ühtne ettekujutus sellest, kuidas hakata ja kes hakkab ajatempleid välja andma. Puuduvad igasugused standardid sel alal. Veelgi enam, ajatempli alal tehtav teadustöö on nii meil kui ka mujal veel lapsekingades ja ega polegi peale luuletajate eriti kellelegi viidata. Tõsi küll, Haber ja Stornetta kirjutasid ajatemplitest juba 1993. aastal, kuid nii nende poolt pakutud kui ka hilisemad täiustatud variandid on kaugel sellest, et olla praktikas rakendatavad. Eelmisel aastal oli aga ajatempli süsteemide arendamise alal märgata liikumist ja seda just tänu digitaalsignatuuri reguleerivate seaduste massilise (ja kohati uisa–päisa) vastuvõtuga Ameerika Ühendriikides, Euroopas ja ka Aasia maades. On aga selge, et ilma funktsioneeriva ajatemplite süsteemita võib selline tegevus viia korratuse ja kaoseni, mida me niiväga tahaksime vältida. Eesti ja ka teised paljunäinud väikeriigid (näiteks Belgia) on tüdinud kaosest ja korralagedusest ning on asunud aktiivselt koostööd tehes ajatemplite tehnilist lahendust välja töötama. Eestis on käimas projekt, mille tulemusena pannakse loodetavasti juba sel kevadel käima esimene ajatempli server.

Ei ole kaduvaid, kõduvaid aegu.
Alles jääb hetk,
milles asume praegu.
Aeg, mis on tekkinud,
enam ei haju,
kui seda jäävust ka meeled ei taju.
(A. Alliksaar)

Ahto Buldas (ahtbu@ioc.ee) ja **Helger Lipmaa** (helger@cyber.ee) on Küberneetika AS–i vanemteadurid. Viidatud luuletused on kättesaadaval täies mahus aadressil http://www.cs.ut.ee/~helger/luule