CYBERNETICA

Krüptograafiliste algoritmide kasutusvaldkondade ja elutsükli uuring

Aruanne
Versioon 3.1
31. detsember 2013. a.
83 lk
Dok. A-77-5

Sisukord

1 Sissejuhatus

See aruanne on koostatud eesmärgiga koondada teaduskirjanduses ja rahvusvahelistes uuringutes esitatud soovitused krüptograafiliste süsteemide ja -algoritmide kasutamiseks riigi infosüsteemides. Aruanne on suunatud tarkvaraarhitektidele, IT-audiitoritele jt tehnilistele spetsialistidele, kes osalevad uute infosüsteemide loomisel ja olemasolevate süsteemide turbe tagamisel.

Aruande esimene versioon[1] valmis 2011. a ja toona hindasid autorid tema ajahorisondiks ligikaudu viis aastat [40]. Selle aja jooksul jõuab arvutustehnikas ja krüptoanalüüsi meetodites toimunud areng muuta kaugemaleulatuvad ennustused ebausaldusväärseks. Seetõttu soovitati toonases aruandes ta uuesti läbi vaadata ja ajakohastada hiljemalt aastal 2016. Tegelikkuses aga soovis tellija (Riigi Infosüsteemi Amet) vahepealsete arengute ning uute vajaduste valguses aruande uuendamist juba 2013. aastal. Ehkki olulised krüptograafilised primitiivid ei ole vahepeal murdunud, areneb kogu valdkond nii kiiresti, et selline tähelepanu infoturbe alustehnoloogiatele on kahtlemata õigustatud.

Ka selle aruande ajahorisondiks võib pidada ligikaudu viit aastat, mis tähendab, et järgmine läbivaatus tuleks teha hiljemalt aastal 2018. Võrreldes aruande eelmise versiooniga on soovituste osas eraldi välja toodud primitiivid, mis on küll veel laialt kasutusel, kuid tuleks välja vahetada juba lähema kahe aasta jooksul, sest nende murdmine on juba lähiajal võimalik isegi võrdlemisi tagasihoidliku eelarvega.

Aruanne ei sisalda kõigi krüptograafiliste mõistete seletusi ja eeldab krüptograafia algteadmiste olemasolu. Kindlasti piisab ülikoolides loetavatest sissejuhatavatest krüptograafia kursustest või ka praktilisest töökogemusest käsitletavas valdkonnas. Parimaks allikaks eestikeelsele terminoloogiale on Andmekaitse ja infoturbe seletussõnastik [139], mis on kättesaadav ka veebis[2].

Aruanne ei anna täielikku ülevaadet kogu maailmas kasutatavatest krüptograafilistest algoritmidest. Arvestades töö ajalisi ja mahulisi piiranguid, keskendutakse Eestis kasutatavatele krüptoalgoritmidele ja protokollidele.

1.1 Vajadus

Krüptosüsteemid murduvad enamasti mitte üleöö, vaid järkjärgult. Samuti juhtub sageli, et krüptoalgoritm muutub ebaturvaliseks ainult mõnes kasutusvaldkonnas, jäädes samas täiesti turvaliseks teistes valdkondades. Klassikaline näide on räsifunktsioonid. Toimiva kollisiooniründe leidmine ei tähenda veel ebaturvalisust kasutusvaldkonnas, kus piisab juba ühesuunalisusest või turvalisusest lisaoriginaaliründe suhtes.

[1] Uuring koostati Euroopa Liidu struktuurifondide programmi "Infoühiskonna teadlikkuse tõstmine" raames Riigi Infosüsteemi Ameti ning Majandus- ja Kommunikatsiooniministeeriumi riigi infosüsteemide osakonna tellimusel.

[2] `http://akit.cyber.ee/`

Sageli ei ole aga tarkvaraarendajad piisavalt kompetentsed otsustama, milliseid kasutusvaldkondi uued ründed ohustavad ja milliseid mitte. Samuti ei saa arendajad alati aru, et loodav süsteem vajab turvalisuse tagamiseks krüptograafilisi meetodeid. Sageli jäetakse krüptograafilised meetmed realiseerimata arenduskiiruse huvides. Halvimal juhul otsustab arendaja korrektse lahenduse asemel kasutada omaloodud krüptoalgoritme. Sellistele probleemidele on ka raske jälile jõuda, sest arendaja vaatepunktist on kõik justkui korras. Organisatsioonid ei ole ka alati huvitatud sõltumatust koodiauditist, tuues ettekäändeid alates ärihuvidest ja lõpetades riigisaladusega.

Krüptograafiakogukonnas on juba enam kui sada aastat tagasi omaks võetud lingvisti ja krüptograafi Auguste Kerckhoffsi [156] aastal 1883 sõnastatud põhimõte, et **krüptosüsteemi turvalisus peab sõltuma ainult võtmete salastusest, mitte aga tema kirjelduse salajasusest**. Seda põhimõtet on ajaloos korduvalt eiratud ning sageli jõutud olukorrani, kus nõrka krüptosüsteemi on üritatatud kaitsta varjamisega (*security by obscurity*), kuid pärast süsteemi avalikustamist on ta kiiresti murtud. Hea näide on DVD-plaatide kaitsemehhanism CSS (*Content Scrabmling System*), mille murdmine sai pärast pöördprojekteerimist tavalisel arvutiriistvaral võimalikuks lausa reaalajas [255]. Hoiatavad praktilised näited on ka raadiovõrkude turvamisel kasutatavad algoritmid A5 ja RC4 (vt jaotised 2.1.2 ja 2.1.3).

Infoturbespetsialistide kaasamine suuremate infosüsteemide loomisesse on tänapäeval vältimatu. Ka Eesti Vabariigi andmekogudele kehtestatud kolmeastmeline etalonturbe süsteem ISKE näeb ette krüptograafilisi lahendusi ning organisatsioonilisi mehhanisme nende kasutamiseks (vt [35], meede M 2.161 "Krüptokontseptsiooni väljatöötamine", meede M 4.90 "Krüptoprotseduuride kasutamine ISO/OSI etalonmudeli eri kihtides" jt).

1.2 Krüptograafia reaalse maailma kontekstis

Infosüsteemide turvataset hinnates tuleb arvestada, et krüptograafiliste primitiivide ja protokollide ründamine ei ole ainus võimalus infosüsteeme kahjustada. Pigem vastupidi – viimase paarikümne aasta jooksul on loodud niivõrd tugevad krüptograafilised algoritmid, et enamasti ei tasu nende ründamine end ära, mistõttu on sagedasemad ründeobjektid krüptoalgoritmide konkreetsed teostused ja vananenud krüptograafiat kasutav tarkvara ja riistvara.

Üheks nõrgemaks lüliks reaalse maailma süsteemides on inimene, kes oma teadmatuse, füüsilise piiratuse või liigse usaldavuse tõttu võib suure osa infotehnoloogilistest turvameetmetest kasutuks muuta. Levinud näide on manipuleerimisründed (*social engineering attacks*), mille korral ründaja petab lihtsameelselt kasutajalt välja ründeks vajalikku teavet, või veenab teda süsteemi kahjustavalt käituma. Tüüpilistest manipuleerimisvõtetest võib pikemalt lugeda klassikalisest teosest *The Art of Deception* [197].

Teine probleem, mis inimese kergesti rünnatavaks muudab, on tema põhimõtteline võimetus arvutis toimuvat vahetult tajuda. Inimene vajab arvutiga suhtlemiseks välisseadmeid (klaviatuur, monitor, hiir, kõlarid jmt) ning kõiki neid seadmeid on võimalik rünnata. Nii piisab arvutil kirjutatud teksti taastamiseks vaid klaviatuuriklõbina salvestamisest [275].

Ka Eestis on esinenud laia kõlapinda leidnud ründeid välisseadmete vastu. Näiteks 2011. aasta parlamendivalimiste ajal avalikustati kahjurvara prototüüp, mis oleks ekraanipilti muutes potentsiaalselt suutnud heauskse kasutaja poolt antud elektroonilist häält mõjutada [141]. Sisuliselt taandub probleem kasutatava töötluse baasi usaldusväärsusele (vt. lk. 7), mida ei ole võimalik tagada üksnes infotehnoloogiliste meetoditega. Kasuta-

jad peavad mõistma, et arvutite heakord on suuresti nende endi teha. Vajadus autoga regulaarselt tehnoülevaatusel käia on ühiskonnas juba ammu omaks võetud. Niisama elementaarseks peaks saama ka kahjurvaratõrje kasutamine arvutis.

Krüptograafiliste primitiivide kasutamiseks kõrgema taseme protokollide ehitamisel on sageli mitmeid võimalusi, kuid kõik nad pole ühtviisi turvalised. Keerukust lisab asjaolu, et osaliselt murdunud primitiiv võib olla teatud kontekstis jätkuvalt turvaline. Otsus selle primitiivi väljavahetamiseks on suuresti majanduslik ning eeldab turvariskidest tuleneda võivate kahjude hindamist. Aruande eesmärk on soovitada kõigil asutustel ja ettevõtetel teha selline analüüs, et hiljem mingi krüptoalgoritmi murdudes olla valmis kiiresti tegema eelnevalt läbimõeldud otsuseid. Selle aruande eesmärk ei ole ega saagi olla kahjuhinnanguite andmine kõigi Eesti Vabariigi infosüsteemide jaoks. Aruanne annab vaid alusmaterjali ning mõned lähtekohad vajalikuks analüüsiks, mis tuleb igal üksikjuhul eraldi läbi viia, sh uute süsteemide arhitektuuri kavandamisel.

1.3 Usaldatava töötluse baas

Inimese võimetus arvutis toimuvat vahetult tajuda ja kontrollida ning ise krüptosüsteemi kasutamiseks vajalikke arvutusi läbi viia tähendab, et ta peab usaldama riist- ja tarkvarakomponente, st uskuma, et need töötavad korrektselt. Samuti tuleb võib-olla uskuda, et teatud andmed on õiged või loodud õigesti läbiviidud protseduuridega. Kõigi usaldatavate andmete, riist- ja tarkvara, protsesside ning neid protsesse läbiviivate isikute kogumit nimetatakse *usaldatava töötluse baasiks*. Sellesse kuuluvad tavaliselt järgmised komponendid.

- *Arvuti riistvara.* Protsessor peab töötama vastavalt oma spetsifikatsioonile ja tegema neid ja ainult neid arvutusi, mida arvutiprogramm määrab, ekraan peab näitama õiget pilti, jne. Samuti tuleb eeldada, et klaviatuur, ekraan ja hiir ei kopeeri ega edasta informatsiooni ründajaile.
- *Tarkvara.* Tuleb eeldada, et ta teeb just seda, mida me arvame, et ta teeb.
- *Lisariistvara.* Eesti kontekstis tuleb kindlasti usaldada ID-kaarti, mille vigadeta töötamist tuleb eeldada. Samuti tuleb usaldada SIM-kaarte, mis osalevad mobiil-ID protokollides. Eeldada tuleb ka, et neile kaartidele paigaldatud tarkvara töötab korrektselt.
- *Sertifikaadid.* Turvaliseks suhtluseks teise isiku või süsteemiga tuleb teada suhtluspartneri kohta andmeid, mida süsteem kasutab turvalise krüptograafiliselt kaitstud suhtluskanali loomiseks ja ülalhoidmiseks. Kui neid andmeid eelnevalt olemas ei ole, loetakse nad sageli suhtluspartneri sertifikaadist. Siis tuleb eeldada, et sertifikaadi väljastamise käigus on suhtluspartner piisavalt hästi tuvastatud.

Sõltuvalt kasutatavatest primitiividest, protokollidest ja süsteemidest võib usaldatud arvutusbaasil olla muidki komponente. Usaldatava töötluse baasi teadvustamine aitab hinnata süsteemide tegelikku turvalisust. Vajaliku usaldatava töötluse baasi maht ja koosseis on sageli väga oluline kriteerium ühe tehnilise lahenduse eelistamiseks teisele.

1.4 Välisriikide kogemus

Krüptograafiliste primitiivide hetkeseisu seirab regiooni juhtivaid spetsialiste kaasates Euroopa Liidu kompetentsikeskus ECRYPT II. Keskus üllitab igal aastal aruande, kus kajastatakse põhiliste krüptograafiliste algoritmide ja protokollide turvalisust ja antakse

üldisi soovitusi nende kasutamiseks. Hetkel kehtivad soovitused on avaldatud 30. septembril 2012 [62]. Käesolev aruanne tugineb suures osas sellele dokumendile.

Amerika Ühendriikides koostab sarnaseid aruandeid National Institute of Standards and Technology (NIST) Computer Security Resource Center. Hetkel kehtivaid soovitusi Ameerika Ühendriikide valitsusasutuste andmete turvamiseks sisaldab 2012. aasta juulis koostatud dokument [70], mida siinse aruande koostamisel on samuti kasutatud. Aastal 2005 avaldas USA National Security Agency (NSA) omapoolsed soovitused valitsusasutustes kasutatavate avalike krüptoalgoritmide kohta, mida tuntakse koondnimetuse Suite B all [15]. Eksisteerib ka eriti tundlike andmete käitlemiseks kasutatav Suite A, mis ei ole avalik. Suite B algoritmid on standarditud ka mitmete protokollistike (IPsec [176], TLS [231] jt.) osana.

Jaapani valitsuse poolt kasutatavate krüptograafiliste primitiivide kohta annab soovitusi CRYPTREC projekt [8]. Kahjuks pärineb nende viimane inglisekeelne aruanne aastast 2002 [29]. Jaapani teadurid on küll oma uuringutest ja soovitustest hiljem ka ingliskeelseid lühikokkuvõtteid avaldanud. Selle aruande seisukohast on neist kõige huvitavam Shiho Moriai ülevaateartikkel [201].

1.5 Taust ja metoodika

Aruande esimese versiooni koostamisel 2011. aastal oli üks eesmärk ka Eesti olulisemates infosüsteemides kasutatavate krüptograafiliste meetodite kaardistamine. Paraku selgus, et vajalikku teavet on raske koguda. Vastavasisuline järelepärimine tehti 51 asutusele; nende hulka kuulusid maavalitsused, ministeeriumid, ametid ja inspektsioonid, Riigikantselei, Rahvusarhiiv ning IT-tugiasutused, lisaks 6 eraettevõtet. Sisulist vastust ei õnnestunud aga saada neist üheltki.

Et vahepealse kahe aasta jooksul ei ole toimunud suuri õiguslikke muudatusi, mis suunaksid asutusi suuremale koostööle, loobuti seekord samasugusest järelepärimisest ning otsustati keskenduda väiksemale arvule asutustele. Samuti muudeti veidi uurimuse fookust. Konkreetsete krüptomeetodite kaardistamise asemel uuriti pigem, kas kasutatavad meetodid on murdumise korral kergesti vahetatavad, ja kas asutuses on olemas isik, kelle vastutusalasse kuulub krüptoanalüüsi hetkeseisuga kursis olemine. See aga ei tähenda, et kogu Eestit katva kaardistuse ideest tuleks loobuda – ka see on pikas perspektiivis kindlasti vajalik, kuid enne tuleb luua kaardistuseks vajalik õiguslik baas.

Uuringu üks oluline eesmärk on selgitada elliptiliste kõverate krüptograafia kasutuselevõtu võimalusi Eestis. Kuna üks suurim potentsiaalne takistus on vastavate meetodite patenteeritus, viidi uuringu käigus läbi ka patendianalüüs, kuhu kaasati Eesti Patendiameti spetsialiste.

2 Krüptograafilised algoritmid

2.1 Sümmeetrilised krüptoalgoritmid

Sümmeetrilised krüptoalgoritmid jagunevad kahte suurde klassi: plokk- ja jadašifriteks. Tänu jadašifrite ajalisele tõhususele on nende peamised kasutusvaldkonnad madala taseme multimeediumiprotokollid. Teistes rakendustes kasutatakse valdavalt plokkšifreid. Teisest küljest on plokkšifrid jadašifritega võrreldes krüptoanalüüsile paremini vastu pidanud. Sellest johtuvalt keskendutakse ka siinses aruandes peamiselt plokkšifritele ning käsitletakse jadašifreid eeskätt esituse täielikkuse huvides.

2.1.1 DES, TDEA/3DES

Ajalooliselt oli DES esimene laialt kasutust leidnud standardne plokkšiffer. NIST standardis ta esimest korda 1976. aastal nimega FIPS-46. Standardit on hiljem kolmel korral uuendatud; viimane versioon FIPS-46-3 pärineb aastast 1999 [9]. Oma 56-bitise efektiivse võtmepikkuse tõttu peetakse DESi tänapäeval ebaturvaliseks ning ka standard FIPS-46 on 2005. aastast ametlikult tühistatud. USA valitsusasutustel on lubatud DESi kasutada ainult kolmekordse krüpteerimisalgoritmi TDEA (tuntud ka tähise 3DES all) komponendina kuni aastani 2030 [72]. Kuigi TDEA/3DESi võtmepikkus on 168 bitti, on tema iteratiivse konstruktsiooni tõttu tema ründamiseks vajalik reaalne töömaht suurusjärgus ülimalt 2^{112} operatsiooni (mõnede ründestsenaariumite puhul isegi pigem 2^{100} operatsiooni) [62]. Just selle erinevuse tõttu tuleks ka TDEAd võimalusel kõigis rakendustes vältida, seda enam, et AESi näol on olemas väga hea alternatiiv.

2.1.2 A5, Kasumi, SNOW 3G

Kuigi 2. põlvkonna GSM-mobiilsideplatvormil kasutatud algoritme A5/1 ja A5/2 ei ole kunagi ametlikult avalikustatud, pöördprojekteeriti nende ehitus 1990ndate aastate lõpus ning peatselt leiti mõlemast olulisi turvanõrkusi [89, 69]. Aastal 2010 demonstreerisid saksa uurijad Nohl ja Munaut praktilist seadet, mille abil on võimalik GSM-kõnesid ka reaalselt pealt kuulata [208]. Nad ei avalikustanud küll kogu rakenduse lähtekoodi, kuid kirjeldasid kogu ründevektorit piisava põhjalikkusega, nii et motiveeritud ja piisava tehnilise taustaga ründaja suudaks nende seadme sõltumatult uuesti luua.

Juba esimeste nõrkuste leidmise järel hakkasid standardiorganisatsioonid GSMA ja 3GPP algoritme A5/1 ja A5/2 aktiivsest kasutusest kõrvaldama. Samuti õpiti suletud lahenduse ohtudest ning järgmise generatsiooni (3G/UMTS) mobiilside krüptoalgoritmi loomise protsess oli avatum kui GSMi korral. Valituks osutus MISTY šifri modifikatsioon Kasumi [19], mida kasutab võtmejada genereerimiseks ka uus GSMi standard A5/3. 2010. aastal leiti Kasumi vastu lähisvõtmerünne (*related-key attack*), mis võimaldab süsteemis kasutatava võtme täielikult taastada, nõudes ainult nelja lähedast võtit ja ühe lauaarvuti arvutusjõudlust [122]. Hetkel ei ole selge, kas see rünne võib viia uue põlvkonna mobiil-

side krüptograafia murdumisele mõnes praktilises olukorras, kuid on kindel, et Kasumi on nõrgem kui tema loojad algselt kavandasid (paradoksaalselt isegi nõrgem kui algne šiffer MISTY). Samuti on 3. põlvkonna UMTS-protokolli vastu leitud vahendusründeid (*man-in-the-middle attack*), mis võimaldavad ära kasutada GSM-võrgust pärinevaid nõrkusi [192].

Nende nõrkuste kõrvaldamiseks on 4. põlvkonna mobiilsidestandardis 4G/LTE (*Long Term Evolution*) võetud kasutusele uued konfidentsiaalsuse ja tervikluse tagamise algoritmid UEA2 ja UIA2, mis tuginevad jadašifrile SNOW 3G [20]. Hea ülevaade SNOW 3G turvalisusest ja teostuse detalidest on Molina-Gili jt artiklis [198]. SNOW 3G vastu on pakutud ründeid [113, 99], mis eeldavad madala taseme juurdepääsu krüptograafilisi operatsioone sooritavale kiibile. Niisiis võib hetkel pidada šifrit SNOW 3G üldotstarbelise mobiilside krüpteerimiseks piisavalt turvaliseks.

Mobiilteenuste kasutajal ei ole alati võimalik valida, millise šifriga tema sideseanssi turvatakse. Mobiilsideprotokollid võimaldavad automaatset tagasilangust nõrgemate algoritmide kasutamisele ja seda võimalust keelates võib klient sidevõimalusest hoopis ilma jääda. Niisiis tuleb teha valik teenuse käideldavuse ning konfidentsiaalsuse ja tervikluse vahel. Eriti tundlikku teavet on ehk õigem mobiilvõrkude kaudu üldse mitte edastada.

2.1.3 RC4

Jadašifri RC4 töötas 1987. aastal RSA Security juures välja Ronald Rivest (ka RC šifri nimes tähendab mitteformaalselt "Ron's code"). Šifri kirjeldust ei ole kunagi ametlikult avaldatud, kuid tema väidetav lähtekood lekkis 1994. aastal Internetti ning kuna lekkinud koodi väljundjada vastab ametliku RC4 binaarrealisatsiooni omale, on alust uskuda, et ta on autentne. Tänu oma tõhusale tarkvaralisele teostatavusele on RC4 kasutusel paljudes IT-süsteemides ja standardites (SSL/TLS [116], IEEE802.11 standardipere (WiFi) [14], Microsofti protokoll MPPE [211] jpt).

Jadašifrina peaks RC4 ideaalis genereerima juhuslikust bitijadast eristamatu jada, kuid kahjuks on RC4 väljundjadast leitud arvukalt sõltuvusi, mis võimaldavad teha järeldusi šifri sisemise oleku ning kasutatava võtme kohta [183, 214, 184, 215, 129, 158]. Kleini analüüs [158] viis 2007. aastal utiliidi aircrack-ptw väljatöötamiseni; see suudab WEP-protokollis kasutatud 104-bitise RC4 võtme leida vähem kui minuti jooksul [261]. Sarnase jõudlusega ründe pakkusid samal aastal välja ka Vaudenay ja Vuagnoux [264].

Kokkuvõtteks võib öelda, et RC4 on murtud nii teoorias kui ka praktikas ning teda ei ole soovitav kasutada ühegi turvaeesmärgi taotlemiseks. Samuti tuleb turvalisust nõudvates keskkondades hoiduda WiFi-võrkude turvamisest WEP-protokolliga ning kasutada selle asemel protokolli WPA2 (vt ka jaotist 3.5.3).

2.1.4 Blowfish

Šifri Blowfish töötas 1993. aastal välja Bruce Schneier [234] ning see oli omal ajal üks esimesi patendivabu plokkšifreid, mis on sellest ajast peale leidnud realiseerimist paljudes krüptoteekides ja -protokollistikes. Blowfishi lahendus sobib kasutamiseks nii tark- kui ka riistvaras ja vaatamata pikaajalisele analüüsile ei ole tema vastu olulisi ründeid leitud. Seega sobib Blowfish (sobiva võtmepikkusega) kasutamiseks praktiliselt kõikjal, kus vajatakse turvalist plokkšifrit. Tõsi, pärast AESi standardimist on Blowfishi populaarsus teataval määral vähenenud.

2.1.5 AES

Standardiorganisatsioon NIST kuulutas 1997. aastal välja võistluse uue põlvkonna plokkšifristandardi (Advanced Encryption Standard, AES) loomiseks ning valikuprotsess jõudis lõpule 2001. aastal, mil võitjaks kuulutati Belgia krüptograafide loodud šiffer Rijndael [207]. Peale *de jure* standardistaatuse on AES kujunenud ka *de facto* kõige laiemalt kasutatud (ja rünnatud) plokkšifriks. Kuna tugevus klassikalise lineaarse ja diferentsiaalse krüptoanalüüsi vastu oli juba AESi väljatöötamise eesmärk, ei ole need meetodid tema vastu ka efektiivseks osutunud. 2002. aastal leiti teoreetilisi võimalusi AESi algebralise struktuuri nõrkuste ärakasutamiseks [110, 202], kuid need ei ole viinud praktiliste rünneteni [106, 178].

2009. aastal leidsid Biryukov ja Khovratovich lähisvõtmeründe AES-192 ja AES-256 vastu [88]. Ründe keerukus on üllatuslikult kõige väiksem AES-256 puhul (suurusjärgus $2^{99,5}$ operatsiooni), kuid ründe õnnestumiseks on vaja, et ründaja suudaks veenda süsteemi krüpteerima tema poolt etteantud lähedaste (Hammingi kauguse mõttes) võtmetega. Kui reaalne selle eelduse täidetus on, sõltub konkreetsest süsteemist ning nõuab eraldi analüüsi. AES-128 korral ei ole lähisvõtmeründe keerukus enam täielikust võtmeruumi läbivaatusest väiksem.

Kõige suuremat ohtu kujutavad AESi analüüsi seisukohast külgründed (*side channel attacks*), eriti protsessori vahemälu ajastuse mõõtmine (*cache timing attacks*), [82, 210]. Ka nende rünnete praktilisus sõltub suuresti konkreetsest rakenduskeskkonnast, nõudes paljude eelduste üheaegset täidetust.

Põhjaliku ülevaate teadaolevatest rünnetest AESi vastu leiab projekti ECRYPT aruandest [105].

Kokkuvõtteks võib öelda, et viimase 10 aasta jooksul on AESi põhjalikult analüüsitud, kuid vaatamata rahvusvahelise krüptograafiakogukonna jõupingutustele ei ole seda siiani sisuliselt murda suudetud. Seega võib AESi nii võtmepikkusega 128, 192 või 256 bitti pidada selle aruande ajahorisondi ulatuses kasutamiseks piisavalt turvaliseks.

2.2 Asümmeetrilised krüptoalgoritmid

Asümmeetrilised krüptoalgoritmid tuginevad teatavate algebraliste ja arvuteoreetiliste ülesannete lahendamise raskusele. Seetõttu eeldab sobivate algoritmide valik vastavate ülesannete lahendamise meetodite hetkeseisu analüüsi. Analüüsi käigus keskendutakse järgmistele probleemidele:

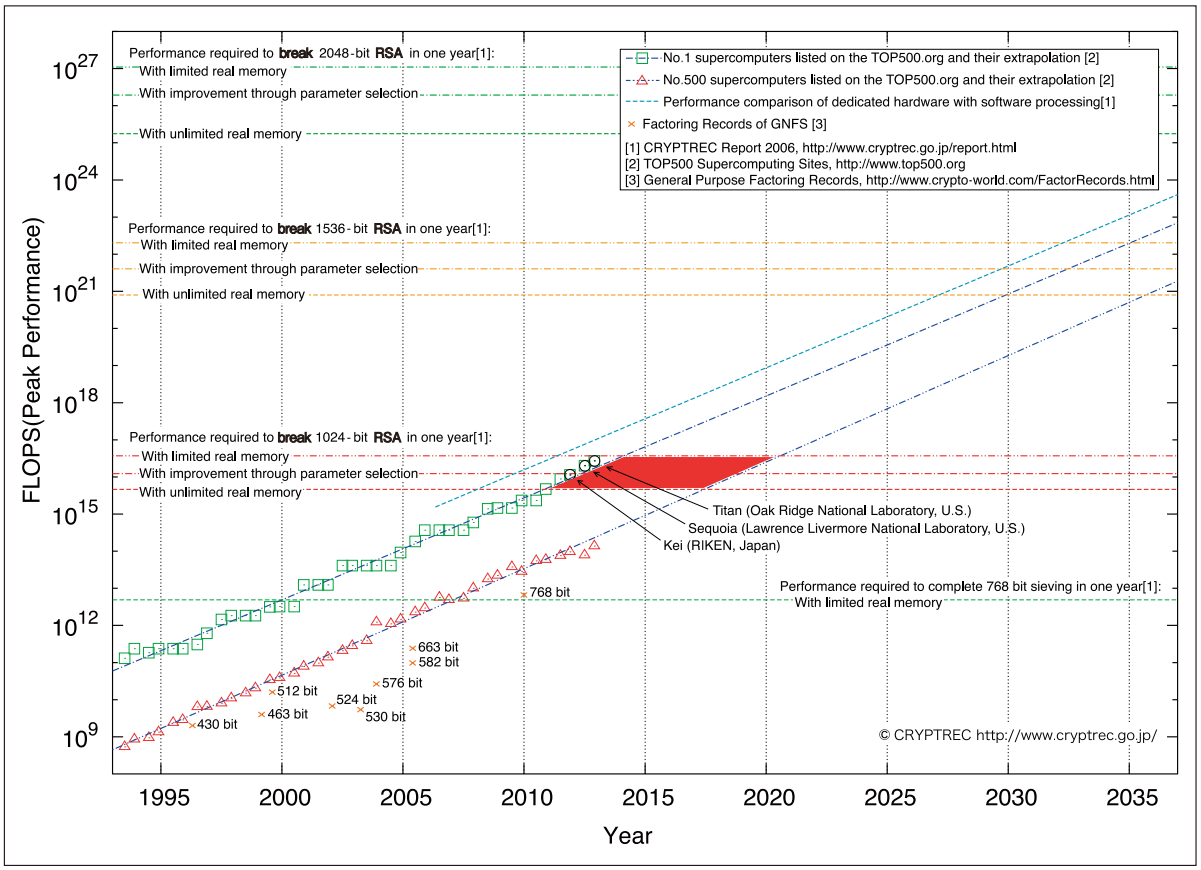
- täisarvude tegurdamine, RSA probleem;
- diskreetne logaritm ja Diffie-Hellmani probleem jäägiklassiringide multiplikatiivsetes rühmades;
- jagamise probleem elliptiliste kõverate rühmades.

2.2.1 RSA

RSA on üks vanemaid ning praktikas kõige laiemalt levinud avaliku võtme krüptograafilisi algoritme. Selle avaldasid 1978. aastal Ron Rivest, Adi Shamir ja Leonard Adleman [225].

2.2.1.1 Mooduli tegurdamine

RSA krüptosüsteemi turvalisus tugineb suurte kordarvude tegurdamise raskusele ja seetõttu on tegurdamine ka kõige rohkem läbiuuritud meetod RSA ründamiseks.



Joonis 1: Jaapani teadlaste analüüs RSA mooduli tegurdamiseks vajaliku töömahu kohta.

Parim hetkel teadaolev suurte kordarvude tegurdamise meetod, arvkorpuste sõelumine (*number field sieve*), võimaldas rahvusvahelisel teadlaste rühmal 2009. aastal tegurdada 768-bitise arvu [159]. Aastal 2012 tegurdati 1061-bitine Mersenne'i arv $2^{1061} - 1$ (tänu Mersenne'i arvude eriomadustele on see lihtsam kui suvaliste arvude tegurdamine) [104]. 768-bitise arvu tegurdamiseks kuluks umbes 2000 aastat arvutiaega; tänu rööptöötlusele jäi füüsiline ajakulu suurusjärku 2 aastat. Ründe autorite hinnangul on 1024-bitise RSA ründamine sama meetodiga umbes 1000 korda aeglasem, mistõttu esimese RSA-1024 mooduli murdumist ennustasid uuringu autorid toimuvat mitte varem kui viie, aga mitte hiljem kui kümne aasta pärast.

Asjaolu, mille teadurid 2009. aastal tähelepanuta jätsid, oli kättesaadava arvutusvõimsuse pidev kasv. 2012. aastal eeldasid Jaapani teadlased, et nende käsutuses olev superarvuti Kei suudab 1024-bitise RSA mooduli tegurdada ühe aastaga. Nad analüüsisid ka teiste arvutite võimsusi ja mitmesuguste tegurdusülesannete keerukust ning nende töö tulemuse võtab ilmekalt kokku joonis 1 [201].

Graafikult on samuti näha, et maailma 500. superarvuti jõuab jõudluseni, mis on vajalik RSA-1024 tegurdamiseks ühe aastaga, hinnanguliselt aastaks 2017. Kokkuvõttes tuleb 1024-bitiste RSA moodulite kasutamisest loobuda lähema kahe aasta jooksul, sest pärast seda muutub mooduli murdmiseks vajalik arvutusvõimsus kättesadavaks juba liiga laiale ründajate ringile.

2013. aasta septembris jõudis avalikkuse ette väide, et USA luureorganisatsiooni NSA (*National Security Agency*) võimekus krüptoalgoritmide murdmisel võib olla suurem, kui senini arvatud [172]. Eri allikates on antud spekulatiivseid hinnanguid summadele, mida

NSA eriotstarbelise riistvara arendusse investeerida suudab. Nii näiteks võib 1 miljardi dollari eest ehitada seadmed, mis võimaldavad 1024-bitiseid RSA võtmeid murda mõne tunniga [137].

Küsimusele, kas niisugune perspektiiv ohustab Eesti riigi julgeolekut, ei ole ühest vastust. Ühest küljest annab 1024-bitiste RSA võtmete murdmise võimekus tõepoolest võimaluse lugeda selle algoritmi abil krüpteeritud sõnumeid. Teisalt aga on selge, et NSA suudab soovi korral mõjutada USA-s asuvaid tarkvaratootjaid (Microsoft, Apple jt) andma agentuurile vahetut juurdepääsu nende tootjate platvormidel töödeldavatele andmetele, nii et krüptoalgoritmide murdmise järele kaob igasugune vajadus. Riskianalüüs selle ohu mõjust Eesti riigi ning äriühingute turbetasemele jääb käesoleva uuringu raamidest välja.

RSA-2048 jääb suure tõenäosusega turvaliseks käesoleva aruande ajahorisondi (st 5 aasta) jooksul. Samas tasub märkida, et Ecrypt II aruanne [62] soovitab uutes arendustes kasutamiseks juba vähemalt 2432-bitiseid RSA mooduleid.

Stsenaarium, kus ründajal on RSA ründamiseks kasutada vaid üks konkreetne avalik moodul, on küll praktikas kõige lihtsamini saavutatav, kuid sugugi mitte ainuvõimalik. Järgnevates jaotistes vaatamegi ka teisi potentsiaalseid ründeid.

2.2.1.2 Korduvate teguritega moodulite ründamine

Kui kujutada ette graafi, mille tipud on algarvud ja kahe tipu vahel on serv parajasti siis, kui leidub RSA-moodul, mis on moodustatud vastavate algarvude korrutisena, peaks see graaf ideaalis koosnema ainult isoleeritud tippudest ja üksikutest, ilma ühise otspunktita servadest. Arjen Lenstra jt demonstreerisid 2012. aastal, et reaalne maailm erineb ideaalsest oluliselt [177]. Nad kogusid Internetis saadaolevatest avaliku võtme sertifikaatidest ca 6.4 miljonit RSA-moodulit ning otsisid neis ühiseid tegureid. Siinjuures tuleb märkida, et kuigi etteantud RSA-mooduli tegurdamine eraldiseisvana on raske, on kahe kordarvu ühise teguri leidmine (juhul kui see eksisteerib) Eukleidese algoritmi abil väga lihtne. Kokkuvõttes leidis Lenstra töörühm, et nende poolt moodustatud graafis oli 1995 sidusat komponenti, milles oli kolm või rohkem tippu; kõige kummalisem sidus komponent oli isomorfne 9-tipulise täisgraafiga K_9.

Nadia Heninger jt uurisid täpsemalt, millest see probleem tekib. Artiklis [143] jõudsid nad järeldusele, et süüdi on madala entroopiaga meetodite kasutamine RSA-mooduli jaoks vajalike juhuarvude genereerimisel (näiteks Linuxis jt UNIXilaadsetes operatsioonisüsteemides `/dev/urandom`'i kasutamine `/dev/random`'i asemel). Seejuures kerkib probleem enamasti marsruuterite, tulemüüride jt seadmete puhul, millel puuduvad välisseadmed, millest piisaval hulgal entroopiat koguda.

2.2.1.3 Teostuse ründamine

Kuna oma algse matemaatilise definitsiooni alusel teostatud RSA-krüptosüsteem oleks semantiliselt ebaturvaline, kasutatakse seda peaaegu eranditult kapseldatuna PKCS#1 vormingusse. 1998. aastal leidis Daniel Bleichenbacher ründe, mis võimaldas standardi PKCS#1 versiooniga 1.5 krüpteeritud avatekste taastada [93]; Bardou jt. esitasid 2012. aastal paar suurusjärku kiirema versiooni sellest ründest [68]. Peale Bleichenbacheri ründe avaldamist töötati välja tõestatavate turvagarantiidega versioon 2 ning hetkel kehtib standardi versioon 2.1 [150].

Sellest näitest lähtudes anname kaks soovitust. Esiteks, krüptograafilistest standarditest on alati soovitatav kasutada uusimaid versioone. Teiseks, krüptograafiliste algo-

ritmide rakendamisel tuleb alati tugineda laialt tunnustatud ja testitud teostustele, nagu OpenSSL [16] või Bouncy Castle [5].

2.2.1.4 RSA ründamine lisateabe olemasolul. Külgründed

Alati ei kehti eeldus, et ründajal on krüptosüsteemi ründamiseks kasutada ainult protokolli väljatöötaja poolt avalikena teatatud väärtused (RSA puhul näiteks avalik moodul ja avalik võti). Välise vaatleja jaoks võivad teavet lekitada näiteks aeg, mis kulub avaliku mooduli jaoks vajalike algarvude genereerimiseks või salajase eksponendiga astendamiseks, töötluse voolutarve jne (vt Paul Kocheri ja tema töörühma artiklid [167, 165, 166]). Seega on realistlik eeldada, et ründajal võib olla osaline juurdepääs salajastele väärtustele.

Esimene uurimus selles vallas pärineb juba 1985. aastast RSA loojate Ron Rivesti ja Adi Shamiri sulest [227], kes tõestasid, et jämedalt $n/3$ biti teadmine n-bitise RSA-mooduli ühest tegurist võimaldab mooduli polünomiaalses ajas tegurdada.[3] Hilisemad arvukad uurimused on seda tulemust oluliselt parandanud, lisatud on teistsuguseid eeldusi ja ründevektoreid. Näiteks tõestas Michael J. Wiener 1990. aastal, et kui RSA salajane eksponent d on (näiteks dekrüpteerimise kiirendamiseks) valitud nii väike, et $d < N^{0.25}$ (kus N on avalik moodul), siis on RSA süsteem polünomiaalses ajas murtav [270]. Hiljem parandasid Boneh ja Durfee seda tulemust kuni piirini $N^{0.292}$ [96].

Häid ülevaateid RSA ründamise hetkeseisust ning viimastest tulemustest leiab näiteks Alexander May ja tema töörühma artiklitest [142, 186].

Kõigist rünnetest ülevaate andmine ning analüüs, millised neist on iga konkreetse infosüsteemi jaoks ohtlikud, jääb siinse aruande raamidest kaugele välja. Üldine soovitus võimalike ohtude vastu on sama, mis jaotises 2.2.1.3 – kasutada läbitestitud ja aktiivse kogukonna poolt toetatud teeke, mida uute rünnete ilmnemisel keskselt uuendatakse, mitte aga tugineda kohalikule teostusele. Jaotises 2.2.1.2 kirjeldatud ründe neutraliseerimiseks tuleb tagada, et moodulite genereerimisel kasutatav juhuarvude generaator kasutaks piisavat entroopiat. Kaitse külgrünnete (nt voolutarbe mõõtmise või ajastusrünnete) vastu sõltub suuresti konkreetsest rakendusest ning hõlmab varjestusmeetodeid ja sama võtme erinevate kasutuskordade arvu piiramist. Head ülevaadet külgrünnetest ning võimalikest kaitsemeetmetest võib lugeda Paul Kocheri jt artiklist [166].

2.2.2 Diskreetsel logaritmil põhinevad süsteemid

2.2.2.1 DSA, ElGamal

RSA kõrval teise olulise avaliku võtmega krüptosüsteemi esitas 1985. aastal Taher ElGamal [127]. Võrreldes RSA-ga olid ElGamali süsteemil mõned olulised eelised. Esiteks oli ta juba definitsiooni tasemel randomiseeritud (leevendades seega oluliselt RSA puhul kerkivat semantilise ebaturvalisuse probleemi) ning teiseks polnud ta patentidega kaetud. Need asjaolud motiveerisid USA valitsust võtma ElGamali süsteemi aluseks oma digitaalallkirjaalgoritmi (*Digital Signature Algorithm*, DSA) standardimisel digitaalallkirjastandardi DSS osana [13]. ElGamali süsteem on defineeritud üle suvalise (tsüklilise) rühma, seetõttu võib tema realiseerimisel kasutada näiteks elliptiliste kõverate rühmi (ning see on standardi DSS praeguses versioonis ka ette nähtud).

[3] Paneme tähele, et ühe RSA-mooduli teguri pikkus on keskmiselt $n/2$ bitti.

2.2.2.2 Diffie-Hellmani võtmevahetus

Oma algsel kujul, nagu Whitfield Diffie ja Martin Hellman selle 1976. aastal esitasid, lahendab Diffie-Hellmani (DH) võtmevahetusprotokoll probleemi, kuidas kaks osapoolt võivad autentse, kuid mittekonfidentsiaalse kanali kaudu leppida kokku ühise saladuse, mida pealtkuulaja ei suuda tuvastada [117]. Tänaseks on DH oma eri teostustes saanud valdavas enamikus Interneti-protokollistikes kasutatavaks võtmevahetusmeetodiks. Protokolli DH on standardinud näiteks IETFi dokumendiga RFC2631 [222] ning standardi *Internet Key Exchange* (IKEv2) osana dokumendiga RFC5996 [153], NSA aga Suite B osana [15] jt.

2.2.2.3 Diskreetse logaritmi ründed

On võimalik tõestada, et ElGamali krüptosüsteemi ründamine on arvutuslikult samaväärne Diffie-Hellmani võtmevahetuse murdmisega, mis omakorda ei ole raskem kui diskreetse logaritmi ülesande lahendamine vastavas rühmas [256]. Seetõttu on peamised krüptograafilised ründed mõlema süsteemi vastu seotud eeskätt just ajaliselt tõhusate diskreetse logaritmi arvutamise meetodite arenguga. Leidub rida klassikalisi meetodeid, mis töötavad kas suvalise või mõne kindla struktuuriga (nt jäägiklassi-)rühmade korral, nt beebisammhiigelsamm (*baby-step-giant-step*), Pollardi ρ-, Silver-Pohlig-Hellmani jt algoritmid. Hea ülevaate peamistest algoritmidest annab Alfred Menezese, Paul van Oorschoti ja Scott Vanstone'i monograafia [191]. Lähtudes klassikaliste rünnete keerukusest hinnatakse n-bitise mooduliga jäägiklassiringis diskreetse logaritmi leidmist umbes niisama keeruliseks, kui n-bitise RSA mooduli tegurdamist.

2013. aastal publitseerisid Antoine Joux jt tulemused, mis vähendavad teatud algebralistes struktuurides (täpsemalt väikese karakteristikuga lõplikes korpustes) parimate teadaolevate diskreetse logaritmi leidmise algoritmide asümptootilist keerukust [65]. Arvestades ajaloolist seost diskreetse logaritmi leidmise ja tegurdusalgoritmide arengu vahel, ilmus kiiresti ka spekulatsioon, et see tähendab paari lähema aasta jooksul kõigi klassikaliste asümmeetriliste algoritmide (sh RSA) kasutuskõlbmatuks muutumist [244]. Esialgu ei ole need spekulatsioonid siiski piisavalt põhjendatud.

2.2.3 Elliptiliste kõverate süsteemid

Elliptiliste ning hüperelliptiliste kõverate omaduste kasutamise krüptograafias pakkusid 1980. aastate teisel poolel sõltumatult Neil Koblitz [161, 162] ja Victor Miller [195]. Võrreldes RSAga on süsteemi kirjeldus keerulisem ning ühe osa avalikest parameetritest moodustab kasutatav elliptiline kõver, mille valimine on mittetriviaalne. Rida USA valitsusasutustes rakendamiseks sobivaid elliptilisi kõveraid spetsifitseerib NISTi standard FIPS 186-4 [13]. Soovitused standardi rakendamiseks annab juhend [38], mis täpsustab paljusid reaalsete kõverate ja võtmete genereerimise, hoidmise ning kasutamisega seotud üksikasju. Matemaatiliste primitiivide realiseerimist kirjeldab juhend [36]. Ka teised organisatsioonid on üllitanud elliptiliste kõverate krüptograafiat puudutavaid standardeid, neist olulisemad on SECG [18] ja ANSI X9.62 [1]. Ka Saksamaa infoturbeametil BSI on olemas oma standard [42], mis kõverate valiku osas viitab edasi nn Brainpool'i standardile RFC 5639 [179]. Väärib tähelepanu, et NSA Suite B kasutab avaliku võtme krüptograafiast ainult elliptilistel kõveratel baseeruvaid algoritme [15].

Abstraktse algebra seisukohast kasutavad elliptiliste kõverate süsteemid samuti diskreetse logaritmi leidmise raskust teatud rühmades, konkreetsemalt elliptiliste kõverate

punktide rühmades. Sellest tulenevalt saab neile tuginedes ehitada DSA/DSS-põhise allkirjaskeemi [13] ning Diffie-Hellmani põhise võtmevahetusprotokolli [71]. L-bitiste võtmete puhul töötavad parimad üldiste rühmade jaoks välja töötatud diskreetse logaritmi arvutamise meetodid (nt Pollardi ρ-algoritm või *baby-step giant-step* algoritm, vt [191] peatükk 3.6) ajakeerukusega $2^{L/2}$. Daniel Brown on esitanud argumente, mis näitavad, et elliptiliste kõverate rühmad käituvad paljuski üldiste rühmadega sarnaselt, millest tulenevalt võib loota, et oluliselt paremaid meetodeid nende kasutamisel saadavate krüptosüsteemide ründamiseks ei leidugi [98]. Seda seisukohta on kritiseerinud Stern, Pointcheval, Malone-Lee ja Smart [251] ning viimaste seisukohta omakorda Koblitz ja Menezes [163]. Aktiivne teaduslik diskussioon selles vallas alles areneb, kuid kindlalt võib öelda, et jäägiklassiringide kohta on teada märksa rohkem sisemist struktuuri, mis aitab ründeid tunduvalt tõhusamalt realiseerida.

Elliptiliste kõverate algebralist struktuuri on edukate rünnete realiseerimiseks õnnestunud ära kasutada vaid mõnel üksikul erijuhul. 1993. aastal näitasid Menezes, Okamoto ja Vanstone, et kui kasutatava elliptilise kõvera rühma elementide arv jagab suurust q^k-1 mingi algarvu q ja väikese astendaja k korral, saab elliptilise kõvera diskreetse logaritmi ülesande taandada diskreetse logaritmi ülesandele korpuse $GF(q^k)$ multiplikatiivses rühmas ning selle ülesande jaoks on omakorda teada subeksponentsiaalse ajalise keerukusega algoritmid [190]. Semaev [236], Smart [247] ning Satoh ja Araki [233] näitasid sõltumatult, et kui elliptilise kõvera punktide arv langeb kokku põhikorpuse elementide arvuga (kõver on *anomaalne*), siis saab vastava diskreetse logaritmi ülesande lahendada polünomiaalses ajas. Mõlema ründe rakendumise kriteeriumid on parameetrite genereerimisel lihtsasti kontrollitavad ja neid ründeid on seega lihtne ära hoida. Süstemaatilise ülevaate ellptiliste kõverate krüptosüsteemide ründamisest ning hetkeseisust annavad Koblitz, Menezes ja Vanstone artiklis [164].

2.2.3.1 Patendid

Kuigi elliptiliste kõverate krüptosüsteemid pakuvad RSA ja diskreetse logaritmi süsteemidega võrreldes sama taseme turvalisust palju lühemate võtmetega ja seega tunduvalt efektiivsemalt, ei ole elliptilised kõverad praktilistes rakendustes sugugi laialt levinud. Peamine põhjus on siin juriidilised asjaolud, sest paljud vastavate süsteemide teostamiseks vajalikud meetodid on kaetud patentidega. Patendid on laiali firmade käes (sh Sun Microsystems, Apple Computer, Certicom, Cylink) ning pole selge, millised neist firmadest ja milliste väidetega võivad sekkuda. Näiteks Sun Microsystems on välja kuulutanud "patendirahu" ning annetanud oma elliptiliste kõverate meetodite lähtekoodi kasutamiseks teegis OpenSSL. Samal ajal kaebas Certicom, kellel on üle 130 vastavasisulise patendi, 2007. aastal Sony kohtusse intellektuaalomandi õiguste rikkumise pärast. Hagi lükati 2009. aastal küll tagasi, kuid küsimus sellest, milliste õiguslike järelmitega peab elliptiliste kõverate süsteemi realiseerija arvestama, on endistviisi lahtine. Miinimumina tuleb arvestada rahvusvahelises kohtus käimise kuludega. Certicomi õigust elliptiliste kõverate süsteemide intellektuaalomandile pretendeerida tunnustas ka NSA, kes 2003. aastal litsentsis firma tehnoloogiat 25 miljoni dollari eest USA valitsusasutustes kasutamiseks.

Käesoleva uuringu üks eesmärk oli selgitada, milline on elliptiliste kõverate patendistaatus Eestis, Euroopas ning kogu maailmas üldisemalt. Selleks pöörduti Patendiameti spetsialistide poole. Läbiviidud küsitlustest selgus, et patendid jagunevad oma katvuspiirkonnalt mitmesse klassi.

- **Eesti patendid**, mis on taotletud ning kehtestatud Eesti Vabariigi territooriumil.

- **Euroopa patendid**, mis on taotletud ning kehtestatud territoriaalsel põhimõttel Euroopas. Euroopa patent võib Eestis kehtida või mitte kehtida, sõltuvalt taotleja soovist.
- **Euroopa Liidu patendid** on uus patendiliik, mille täpne sisseviimise aeg on praeguse kirjutamise hetkel (sügis 2013) veel selgusetu. Esialgsete plaanide kohaselt pidi Euroopa Liidu ühine patendisüsteem rakenduma 2014. aasta algusest, kuid kuna mitmeid õiguakte ei ole veel suudetud vastu võtta ega vastavat patendikohust luua, lükkub see protsess ilmselt tulevikku. Kindalt on teada see, et ükski olemasolev patent automaatselt Euroopa Liidu patendiks ei muutu ning Euroopa Liidu patente saab hakata taotlema ainult uudsetele leiutistele. See tähendab muuhulgas, et USAs hetkel kehtivad elliptiliste kõverate krüptograafia patendid ei või ELi patentideks saada.
- **Teiste riikide patendid** puudutavad Eestit ainult siis, kui mõni Eesti ettevõte soovib oma tootega vastavale turule siseneda. Selleks otstarbeks tuleb kindlasti teha täiendav patendiuuring, kogu maailmas kehtivate patentide analüüs jääb siinse aruande raamidest enamasti välja. Erandina toome ära vaid kokkuvõtte Dan Bernsteini uuringust mõnede USA patentide kohta ning Eesti Patendiameti analüüsi kahe standardse ellptilise kõvera kohta.

Patentide mõju hindamisel tuleb arvestada veel teistegi asjaoludega peale territoriaalse põhimõtte.

Näiteks võivad leiutised olla kaitstud mitte üksnes juba välja antud patentidega, vaid ka sisseantud patendiavaldustega, mis hiljem võivad patentideks saada. Et aga pärispatendi saamise kulud ületavad tavaliselt oluliselt patendiavalduse menetlemise kulusid, siis on levinud patendi saamise tahtlik venitamine, kasutades näiteks nn. jätkutaotluste (*continuation*) mehhanismi, kus autor saab lisada olemasolevale patenditaotlusele uusi sellesama leiutise "kehastusi". Patendiavaldused ei ole erinevalt patentidest avalikud ja seega võivad nad tänu määramatusele olla pärispatentidega võrreldes veelgi tõhusamad konkurentide tõrjujad. Seega kui patendiuuringut teha, tuleks adekvaatse pildi saamiseks saada juurdepääs ka menetluses olevatele patenditaotlustele.

Teiseks küsimusi tekitavaks aspektiks on uudsus. Kuigi uudsus on leiutise patenditavuse üheks eelduseks, ei ole selle tingimuse kontroll sugugi lihtne – põhimõtteliselt tuleks läbi uurida kõik, mis inimkonnale teada on; see ülesanne aga ei tarvitse jõukohane olla. Nii võidaksegi välja anda patente, mis uudsuse nõuet rikuvad ja seetõttu kohtuvaidluses alati vastu ei pea. Enne kohtuvaidluse lõppu ei ole aga võimalik patendi kehtivust hinnata, mistõttu ka potentsiaalselt kehtetut patenti saab kasutada näiteks konkurendi ähvardamiseks.

Senised standardid käsitlevad patenditemaatikat väga ettevaatlikult. Näiteks BSI dokument [42] ütleb:

> *The algorithms described in this guideline have been carefully selected to allow patent-free and/or license-free implementations. Nevertheless, some of the described algorithms or its particular implementations may be subject of patent rights. The BSI shall not be held responsible for identifying any or all such patent rights.*

SECG standard [18] toob samaväärse lahtiütlusklausli:

> *The reader's attention is called to the possibility that compliance with this document may require use of an invention covered by patent rights. By publication of this document, no position is taken with respect to the validity of*

this claim or of any patent rights in connection therewith. The patent holder(s) may have filed with the SECG a statement of willingness to grant a license under these rights on fair, reasonable and nondiscriminatory terms and conditions to applicants desiring to obtain such a license. Additional details may be obtained from the patent holder and from the SECG website, www.secg.org.

Järgnevas esitame elliptiliste kõverate patendianalüüsi, toetudes Patendiameti ja Dan Bernsteini uuringutele.

Eesti patendid Eesti Patendiametisse elliptiliste kõverate krüptograafia patente või kasulikke mudeleid tänaseks (oktoober 2013) esitatud ei ole.

Euroopa patendid Eesti Patendiameti registris elliptiliste kõverate krüptograafia osas Euroopa patente ei ole. Küll aga eksisteerib hulk Euroopa patenditaotlusi, milles Eesti on ära märgitud ja mis võivad siin kehtima hakata. Eesti Patendiameti poolt koostatud taotluste täielik nimekiri on toodud käesoleva aruande eraldiseisva lisana.

USA patendid USAs kehtivatest elliptilise krüptograafia meetodeid katvatest patentidest on krüptograaf Dan Bernstein teinud ülevaate [80], mida järgnevas refereerime. Bernsteini analüüsi eesmärk on uurida, kas tema enda pakutud kõver `Curve25519` [83] võiks mõnda patenti rikkuda, ja ta on jätnud vaatluse alt välja terve rea patente, mis seda ilmselgelt ei tee. Seega ei saa Bernsteini tulemusi kasutada kõigi võimalike kõverate patendikaetuse hindamiseks, kuid see annab siiski kogu temaatikast hea ülevaate.

Kõigepealt toob Bernstein viite kolmele USA patendile (nr 5159632, 5271061 ja 5463690), mis katavad taandamist modulo p, kus p on algarv, mis on teatud lihtsal viisil esitatav arvu 2 astmete kaudu. Niisugune esitus võimaldab taandamise realiseerida elementarsete bitioperatsioonide (nihete ja liitmiste) abil ning just niisugust moodust need patendid katavadki. Kuna paljude NIST FIPS 186-4 [13] kõverate moodulid on valitud sarnases mugavas esituses, tekib nende kasutamisel oht vastavaid patente rikkuda. Muuhulgas mõjutab see oht kõveraid `P-192`, `P-224`, `P-256`, `P-384` ja `P-521`.

Need on USA patendid ja nad ei kehti automaatselt Eesti territooriumil. Dan Bernstein tõstatab ka nende patentide kehtivuse küsimuse üldisemalt [81]. Sarnased optimeerimismeetodid olid tema väitel publitseeritud juba enne patendi väljaandmist, 1990. aastal tegid seda Bender ja Castagnoli [79]. Lõpliku vastuse niisugustes küsimustes saab aga anda ainult kohus. Bernsteini enda loodud kõver `Curve25519` ei kasuta modulaartaandamisel potentsiaalselt patenteeritud mehhanisme ning on selles valguses turvalisem valik.

Teiseks toob Bernstein potentsiaalselt olulisena ära viite USA patendile number 6141420, mis nõudlusis katab ka elliptilise kõvera punkti pakkimismeetodit, kus punkti kahe koordinaadi asemel saadetakse ainult üks (kuivõrd teine koordinaat on temast algebraliselt tuletatav). Bernsteini hinnangul ei taluks ka see väide kohtuvaidlust, sest see idee on publitseeritud juba 1986 aastal Milleri esialgses artiklis [195]. Kui seda väidet aga õnnestuks kohtuvaidluses kaitsta, tähendaks see tõsist probleemi väga paljudele elliptiliste kõverate krüptograafia rakendustele, sest punktide pakkimine on vajalik väikeste krüptotekstide moodustamiseks ja seega ka oodatava efektiivsuse saavutamiseks.

Kolmandaks viitab Bernstein USA patendile number 5299262, mille nõudlus katab teatud meetodeid fikseeritud alusel astendamiseks. Bernsteini hinnangul ei rakendu need ei tema kõverale `Curve25519` ega NISTi kõveratele, samuti ei tohiks nad taluda kohtuvaidlust, sest patendis kirjeldatud meetodid on mitmetes teadusartiklites varem avaldatud.

2.2.3.2 Kasutuselevõtt Eestis

Iga krüptograafilise primitiivi turvalisus sõltub tema analüüsiks tehtud jõupingutuste hulgast. Näiteks on matemaatikud suurte kordarvude tegurdamise probleemiga tegelenud üle 2000 aasta ning seetõttu võib tegurdamise raskusele tuginevat RSA-süsteemi võrdlemisi turvaliseks pidada. Elliptiliste kõverate ajalugu on kahe suurusjärgu võrra lühem, kuid ka nende puhul kehtib reegel, et kasutada tuleks niisuguseid kõveraid, mida on kõige rohkem uuritud. Võrreldes standardeid FIPS 186-4 [13], SECG SEC2 [18] ja ANSI X9.62 [1], võib näha, et nad kattuvad mõnede soovitatavate primitiivide osas. Hea võrdlustabeli toob RFC4492 lisa A [90]. Sellest võrdlusest nähtub, et on olemas kaks kõverat, mida need standardid eri nimede all spetsifitseerivad, nimelt `prime192v1` ja `prime256v1` (NISTi tähistuses `P-192` ja `P-256`). Seega on need kõverad laialt tunnustatud ning väärivad Eestis kasutamise kaalumist.

Esimene probleem nende kõverate juures on patendid. Kõverate `prime192v1` ja `prime256v1` algarvulised moodulid on valitud nii, et taandamine nende moodulite järgi oleks realiseeritav lihtsate bititehetega. See moodus võib aga olla patenteeritud.

Teine probleem on seotud elliptiliste kõverate krüptoprimitiivide parameetrite valikuga. Neid parameetreid on rohkem kui näiteks RSA korral, nad on RSA-ga võrreldes algebraliselt keerukamad ning keerukam on ka nende valimise protseduur. Põhimõtteliselt on seda protseduuri kontrollival osapoolel võimalik valida parameetreid nii, et nende vahel kehtib veel mingeid ainult talle teada olevaid seoseid, mis lubavad selle teabe valdajal vastavat krüptoalgoritmi kergemini murda.

Seoses NSA paljastustega on krüptograafilises kogukonnas tõusnud küsimused ka NISTi standarditud primitiivide kohta. Näiteks on küsimärgi alla sattunud pseudojuhuarvude generaator `Dual_EC_DRBG` ning kõver `P-256` [46]. Täpsema analüüsi võimaluse kohta, et erinevatesse kõveratesse on paigaldatud salajasi tagauksi, koostasid Dan Bernstein ja Tanja Lange [84]. Nende hinnang kõveratele `prime192v1` ja `prime256v1` on tagaukseohu seisukohast hävitav, kõvera `Curve25519` struktuur on nende väitel aga täielikult läbipaistev. Üldisemaid kaalutlusi NSA tagauste kohta on ka aruande jaotises 2.2.1.1.

Üldised soovitused elliptiliste kõverate kasutuselevõtuks sõltuvadki sellest, kas NSA tagaust loetakse ohuks või mittte. Kui ei loeta, on mõistlik kasutada võimalikult standardseid kõveraid, nt `P-192` ja `P-256`, mis kuuluvad olulisemate standardite ühisossa.

Kui aga NSA tagauste olemasolu peetakse oluliseks riskiks, on mõistlikum kasutada Dan Bernsteini kõverat `Curve25519` või mõnd BSI poolt standarditud kõverat, mille valimise protseduur on läbipaistvam [42].

Hetkel ükski elliptiliste kõverate krüptograafia patent Eestis ei kehti, sest patendiomanikel ei ole senini olnud motivatsiooni neid Eestis kehtestada. Kui elliptilised kõverad aga Eestis laialdaselt kasutusele läheks, võib oodata rea Euroopa patentide katvusala kiiret laiendamist ning see tähendab vajadust litsentsitasudega arvestada.

2.3 Räsifunktsioonid

Võrreldes üsnagi hästi läbiuuritud võtmepõhise krüptograafia algoritmidega on turvaliste räsifunktsioonide loomise meetodite kohta teada oluliselt vähem. Peaaegu kõik praktikas levinud räsifunktsioonid kasutavad mingil määral *ad-hoc*-konstruktsioone ning nende formaalseid turvatõestusi on tunduvalt raskem anda. Seetõttu on räsifunktsioonid viimastel aastatel olnud ühed enimmurtud krüptograafilised primitiivid.

2.3.1 MD5

Nõrkustega funktsiooni MD4 asendamiseks töötas Ronald Rivest 1991. aastal välja räsifunktsiooni MD5 ning selle standardis IETF 1992. a [226]. Esimesed nõrkused leiti uuest standardist juba 1996, mil Hans Dobbertin demonstreeris MD5 raundifunktsiooni kollisiooni [118]. 2004. aastal leidis professor Wangi töörühm kollisiooni ka kogu räsifunktsioonile [268]. Hiljem on selle ründe efektiivsust oluliselt tõstetud, nii et nüüdseks on MD5 kollisioone hariliku lauaarvuti abil võimalik leida sekunditega [252, 272]. Kuigi leitavad kollisioonid ei ole täiesti vabalt valitavad, vaid peavad vastama üsna paljudele kitsendustele, on teadaolevatest aluskollisioonidest lähtudes leitud kollisioone ka reaalsete semantikaga dokumentidele nagu PostScripti failid [112] või X.509 sertifikaadid [254].

Kõik ülalöeldu kehtib spetsiifiliselt kollisioonide leidmise kohta. Originaaliründe (*preimage attack*) mõttes on MD5 endiselt tugev. Parim teadaolev rünne suudab originaali leida $2^{123.4}$ operatsiooniga, sellal kui MD5 väljundil on üldse 2^{128} teoreetiliselt võimalikku väärtust [232].

Kuid MD5 kasutamist ei saa kollisioonide leidumise tõttu enam soovitada, sest krüptoprotokolle realiseerivad programmeerijad ei mõista sageli, millistel eeldustel on see funktsioon turvaline. Tegelikult ei anna infosüsteemide ehitajad endale sageli aru isegi sellest, et nad krüptoprotokolli loovad. Seega soovitame üldise turvataseme tõstmiseks kasutada ainult niisuguseid primitiive, mis on turvalised praktiliselt kõigis kontekstides.

Kui aga mingitel (näiteks eelarvelistel) põhjustel ei ole MD5 täielik väljavahetamine võimalik, tuleks tööde planeerimisel juhinduda järgmistest põhimõtetest.

- Räsifunktsiooni MD5 ei tohi kasutada olukorras, kus ohtlikuks osutub stsenaarium, mille korral ründaja võib saavutada edu kaht räsitavat faili ise ette valmistades (näiteks lepingute või sertifikaatide allkirjastamisega, sõnumi moodustamisega ajatembelduseks või räsipõhiseks logimiseks jms).
- Räsifunktsiooni MD5 võib selle aruande ajahorisondi (5 aastat) ulatuses kasutada olukorras, kus turvaomadused tuginevad räsiväärtusest originaali või lisaoriginaali leidmisele.
- Räsifunktsiooni MD5 võib ka edaspidi kasutada mittekrüptograafilistel eesmärkidel (näiteks ühekordsete identifikaatorite moodustamisel või failide kontrollkoodide arvutamisel juhuslike vigade avastamiseks).

Samas on süsteemi projekteerides väga raske kindlustada, et seda ei hakata kunagi kasutama stsenaariumis, kus see primitiiv enam turvaline ei ole. Seega soovitame käesolevas aruandes üldreeglina MD5 kasutamisest üldse loobuda.

2.3.2 SHA-1

MD5 potentsiaalne nõrkus oli selge juba 1990. aastate alguses, sest tänu suhteliselt lühikesele väljundjadale (128 bitti) on temas sünnipäevaründe abil kollisiooni leidmise keerukus suurusjärgus 2^{64} räsiarvutust.

1993. aastal kuulutas NIST välja uue standardi FIPS 180, mida hakati nimetama SHA (*Secure Hash Algorithm*) ja hiljem SHA-0. NISTi väitel avastati sellest peatselt nõrkusi, mille parandamiseks töötati välja kergelt modifitseeritud SHA-1 ning avaldati see 1995. aastal standardina FIPS 180-1. Hiljem on seda standardit veel kolmel korral uuendatud (2002, 2008 ja 2012). Hetkel kehtib FIPS 180-4 [44].

Käesoleva kirjutamise hetkel on parimaks SHA-1 kollisiooni leidmiseks meetodiks Marc Stevensi rünne, mis nõuab ligikaudu 2^{60} räsiarvutust [253] (märgime, et sünnipäe-

varünne nõuaks ca 2^{80} räsiarvutust). Hindamaks, mida see praktikas SHA-1 turvalisuse jaoks tähendab, toome siinkohal ära Bruce Schneieri ja Jesse Walkeri arvutused [235].

Nende hinnangul nõuab SHA-1 ühe ploki arvutamine umbes 2^{14} protsessoritsüklit, mis teeb Stevensi ründe keerukuseks ligikaudu 2^{74} tsüklit. Ühe serveriaasta loevad Schneier ja Walker aastal 2012 võrdseks 2^{61} tsükliga ning serveriaasta hinnaks \$350 (see on Amazoni virtuaalserveri aastase rentimise kulu). Seega aastal 2012 nõudnuks Stevensi rünne

$$\frac{2^{74}}{2^{61}} \cdot \$350 \approx \$2.800.000\,.$$

Ennustades Moore'i seaduse alusel, et iga kolme aastaga muutuvad arvutid 4 korda võimsamaks[4], ning eeldades, et serveriaasta hind jääb ligikaudu samaks, võib eeldada, et see arvutus annab iga kolme aasta järel neli korda odavama kulu. Niisiis maksaks SHA-1 kollisiooni leidmine aastal 2015 umbes \$700.000, aastal 2018 umbes \$175.000 ning aastal 2021 umbes \$44.000. Lisaks tuleb märkida, et see arvutus võtab arvesse ainult üldotstarbeliste protsessorite jõudlust, spetsialiseeritud riistvara kasutamisel on nende hinnapiirideni jõudmist oodata juba tunduvalt varem.

Kokkuvõttes soovitame loobuda SHA-1 kasutamisest krüptograafiliselt kriitilistes rakendustes hiljemalt aastaks 2015. Sama hinnangu annab ka Ecrypt II aruanne, soovitades loobuda SHA-1 kasutamisest nii kiiresti kui võimalik [62].

2.3.3 SHA-2

Üsna peatselt pärast SHA-1 spetsifitseerimist standardis FIPS 180-1 sai selgeks, et 160-bitine väljundjada ei ole arvutustehnika jõudluse kiiret kasvu arvestades ettenähtavas tulevikus enam turvaline juba kasvõi seetõttu, et ühel hetkel saab võimalikuks 2^{80} räsioperatsiooni nõudev sünnipäevarünne. Seetõttu lisati standardile 2002. aasta redaktsioonis funktsioonid SHA-256, SHA-384 ja SHA-512, 2008. aastal SHA-224 ning 2012. aastal SHA-512/224 ja SHA-512/256. Neid kuut funktsiooni tuntaksegi ühise nime SHA-2 all.

SHA-2 pere algoritmide vastu on leitud vaid ründeid, mis töötavad piiratud raundide arvuga. Täpsemalt osatakse leida kollisiooni 24 raundi korral, samas kui SHA-224 ja SHA-256 kasutavad 64 raundi, SHA-384 ja SHA-512 aga 80 raundi [145]. Räsiväärtuse originaali on võimalik leida 43-raundilise SHA-256 ja 46-raundilise SHA-512 korral [57].

Kokkuvõttes võib SHA-2 pere algoritme käesoleva aruande ajahorisondi ulatuses turvalisteks pidada; vt ka [62, 29].

2.3.4 SHA-3

Kuna kahest kõige laiemalt kasutatavast räsifunktsioonist (MD5 ja SHA-1) üks on juba murdunud ning teine iga hetk murdumas, kuid asenduseks pakutav SHA-2 perekond on võrdlemisi *ad-hoc*-ehitusega, kuulutas NIST 2007. aastal sarnaselt edukaks osutunud AES võistlusele välja järgmise põlvkonna räsifunktsiooni võistluse. 2010. aasta detsembris avalikustati viis finalisti, nende seast valiti 2012. aasta oktoobris võitaks Keccak [43].

Sellest ajast alates on NIST Keccaki kallal töötanud ning seda mitmes osas muutnud. Pärast 2013. aastal avaldatud teateid, et NSA on üritanud krüptograafilisi algoritme

[4] Kuigi tänapäeval sõnastatakse Moore'i seadust üldise arvutusvõimsuse terminites ning hinnatakse arengukiiruseks kahekordistumine 18 kuuga, ei ole see ajalooliselt siiski päris täpne. Gordon Moore'i algses artiklis [199] kõneldakse loogikaelementide arvust kiibi (pinnaühiku) koha ning pakutakse kahekordistumise ajaks ühte aastat. Hiljem muutis Moore ise oma ajahinnangut kahe aasta peale [200]. Praeguseks folkloorsena kokku lepitud hinnangut 18 kuud võib pidada paljude ekspertide arvamuste keskmiseks.

meelega nõrgestada, on ilmunud spekulatsioone, et ka SHA-3 ei paku enam vajalikku turvalisuse ja usaldusväärsuse taset [138].

Käesoleva kirjutamise hetkel (septembris 2013) ei ole SHA-3 standard veel kommentaarideks avatud. Niisiis soovitame hetkel üleminekut SHA-3-le mitte alustada ja kasutada SHA-2 perekonna räsifunktsioone, kuni SHA-3 standarditakse ning saab rahvusvahelise kogukonna heakskiidu.

2.3.5 RIPEMD pere

Esimene RIPEMD pere räsifunktsioon RIPEMD töötati välja EL projekti RIPE (*RACE Integrity Primitives Evaluation*) käigus aastatel 1988–1992. Juba 1995. aastal leidis Hans Dobbertin ründe, mis andis kollisiooni RIPEMD-le, mille raundide arv oli piiratud 3-lt 2-le [119]. Kuigi täisversiooni kollisiooni leidis professor Wangi töörühm alles aastal 2004 [267], hakati algset RIPEMD-d kohe täiendama. 1996. aastal pakkusidki Hans Dobbertin, Antoon Bosselaers ja Bart Preneel välja uusi räsifunktsioone RIPEMD-128, RIPEMD-160, RIPEMD-256 ja RIPEMD-320 [120]. Viimased kaks ei lisa esimesele kahele muud turvalisust peale kaks korda pikemate väljundjadade; see aitab vaid sünnipäevaründe-laadsete jõumeetodite vastu. Kuna sünnipäevaründega nõuab RIPEMD-128 vaid umbes 2^{64} operatsiooni ning lisaks on piiratud raundide arvuga RIPEMD-128-st leitud olulisi nõrkusi [189], ei soovitata RIPEMD-128-t enam kasutada. Kuna sarnane rünne RIPEMD-160 vastu ei toimi, võib RIPEMD-160 käesoleva aruande ajahorisondi ulatuses pidada turvaliseks.

2.4 Sõnumiautentimiskoodid (MAC-koodid)

Sõnumiautentimiskoodidest (*Message Authentication Codes (MAC)*) võib lihtsustatult mõelda kui digitaalallkirjade sümmeetrilistest analoogidest, mida saab kasutada olukorras, kus sõnumi autentsuse ja tervikluse tagamine on oluline ainult kahe osapoole vahel. Seljuhul ei ole tavaliselt vaja kasutada tehnoloogiliselt nõudlikumat asümmeetrilist krüptograafiat, vaid saab lähtuda ka ühiselt jagatavast sümmeetrilisest võtmest.

MAC-koodid ei ole enamasti iseseisvad primitiivid, vaid tuginevad teistele primitiividele, tüüpiliselt plokkšifritele ja räsifunktsioonidele. Krüptograafilises kogukonnas on pakutud võimalikke lähenemisi, milledest osa kohta on võimalik anda ka formaalseid turvatõestusi. Näiteks on HMAC konstruktsiooni [168] kohta tõestatud, et ta on turvaline, kui aluseks võetud räsifunktsiooni raundifunktsioon käitub pseudojuhusliku funktsioonina [75]. Huvitav on märkida, et see turvatõestus ei sõltu alloleva räsifunktsiooni kollisioonivabadusest ja seega ei välista MD5-s leitud kollisioonid otseselt tema kasutamist HMAC-i konstrueerimisel. Küll aga on näidatud, kuidas kasutada räsifunktsiooni teadaolevaid nõrkusi HMAC-i eristamisel juhuslikust funktsioonist [157]. Sellest johtuvalt on mõistlik ehitada HMAC teadaolevalt turvalistele räsifunktsioonidele (nt SHA-2).

Teine laialt levinud meetod MAC-ide saamiseks on kasutada plokkšifrit mõnes sobivas töörežiimis, näiteks CBC-režiimis [17], saades tulemusena konstruktsiooni, mida tuntakse nimega CBC-MAC [7]. On võimalik tõestada, et see konstruktsioon osutub turvaliseks, kui autenditava dokumendi pikkus on ette teada ja fikseeritud [76]; enamasti ei ole see eeldus praktikas täidetud. Algne CBC-MAC-i definitsioon kasutab üht plokkšifrit kahe erineva võtmega. Süsteemi praktilisel teostamisel on kiusatus võtta need kaks võtit võrdseks, kuid see tooks endaga kaasa olulise turvanõrkuse, võimaldades luua kasutaja poolt mõeldust erineva dokumendi, mille MAC valideerub. Niisuguste probleemide enne-

tamiseks on soovitav kasutada CBC-MAC-i mõnd modifikatsiooni, näiteks CMAC-i [250] või OMAC-i [146].

2.5 Võtmepikkuste võrdlused ja soovitused parameetrite valikuks

2.5.1 Võtmepikkuste võrdlused

Ükskõik kui hästi krüptograafilised algoritmid ka ei oleks lahendatud, on neid alati võimalik rünnata jõumeetodil, mis näiteks sümmeetrilise krüptograafia korral tähendab sisuliselt võtme äraarvamist võtmeruumi täieliku läbivaatuse teel või räsifunktsiooni kollisiooni leidmist sünnipäevaründe abil. Asümmeetrilise krüptograafia korral kasutatakse valdavalt algebralisi konstruktsioone, mille sisestruktuuri kohta on reeglina palju teada, ja nii peame asümmeetrilisel juhul lisaks arvestama ründeid aluseks võetud struktuuri vastu.

Krüptograafiliste primitiivide kavandamise üks olulisi eesmärke on saavutada nende käitumine "ideaalilähedaste" objektidena, st nii, et nende vastu ei leiduks jõumeetodist oluliselt tõhusamaid ründeid. Sümmeetrilise primitiivi turvamiseks piisab läbivaatuseks vajaliku võtmeruumi küllaldase suuruse tagamisest.

Algoritme, mille vastu jõumeetodist paremad ründed leitakse, peetakse nõrkadeks. Nõrku sümmeetrilisi algoritme ja räsifunktsioone soovitatakse mitte kasutada. Asümmeetrilisel juhul on teatavad struktuursed nõrkused enamasti paratamatud ja seetõttu tuleb krüptosüsteemi parameetrid valida nii, et parimad teadaolevad ründed oleksid liiga keerukad. Praktiliselt kõik võtmega krüptoalgoritmid on kavandatud võimaldama mitmesuguse pikkusega võtmeid; see suurendab läbivaatuseks vajalikku võtmeruumi ja muudab algoritmi sisestruktuuri ärakasutamise praktikas liiga keeruliseks.

Kui keeruline on "praktikas liiga keeruline" ülesanne? 2006. aastal hinnati kogu maailmas tol hetkel olemasolnud arvutusvõimsust 2^{85} operatsioonile aastas [105]. Arvestades Moore'i seadust, mille üks võimalik tõlgendus ütleb, et olemasolev arvutusvõimsus kahekordistub pooleteise aastaga, võib kirjutamishetkel (august-september 2013) pidada realistlikuks hinnanguks 2^{94} operatsiooni aastas.

Kuigi on ebareaalne eeldada, et kogu maailma arvutusvõimsus oleks suunatud ühe algoritmi või võtme ründamisele, annab see siiski mõistliku ülemise hinnangu praktilisele turvavajadusele.

Teine parameeter, millega peab turvavajaduse hindamisel arvestama, on potentsiaalse ründaja motivatsioon ning sellest tulenev valmisolek investeerida ründeressurssidesse. ECRYPT2 aruanne [62] kasutab tabelis 1 esitatud ründajate ning nende motivatsioonist tulenevate vajalike turvatasemete klassifikatsiooni. Turvavatase on määratud ideaalse sümmeetrilise šifri võtmebittides, st ründamaks šifrit turvatasemega b peaks ründaja tegema 2^b operatsiooni (sisuliselt kogu võtmeruumi läbi vaatama). Kuna aruande [62] soovitused olid kirjutamishetkel umbes aasta vanused, võib Moore'i seadust arvestades kõigile turvavajadustele tänase seisuga lisada umbes ühe biti.

Siinkohal on huvitav võrrelda ECRYPT2 metoodikat jaotises 2.3.2 tooduga. Schneier ja Walker hindasid 2^{61} operatsiooni hinnaks umbes \$350, mis tähendab, et aastal 2012 oleks näiteks 2^{69} arvutusoperatsiooni tegemine maksnud $2^8 \cdot \$350 \approx \$100.000$. Näeme, et need kaks metoodikat on suurusjärkude osas ühel nõul.

Nagu eelpool mainitud, tuleb asümmeetrilise krüptograafia võtmepikkuste soovitamisel arvesse võtta algebralisest struktuurist lähtuvaid ründeid. Tabel 2 esitab ECRYPT2 [62] aruande hinnangud asümmeetriliste ja sümmeetriliste võtmete pikkus-

Tabel 1: ECRYPT2 ründajate klassifikatsioon

Ründaja	Eelarve	Vajalik turvatase
Lihthäkker	0	58
Lihthäkker	$400	63
Kahjurvara looja	0	77
Väike organisatsioon	$10k	69
Keskmine organisatsioon	$300k	69
Suur organisatsioon	$10M	78
Luureagentuur	$300M	84

Tabel 2: ECRYPT2 võtmepikkuste ekvivalentsuse tabel.

Turvatase	RSA / DLOG	EC
48	480	96
56	640	112
64	816	128
80	1248	160
112	2432	224
128	3248	256
160	5312	320
192	7936	384
256	15424	512

Tabel 3: ECRYPT2 asümmeetriliste võtmepikkuste turvatasemete tabel.

RSA/DLOG võtmepikkus	Turvatase
512	50
768	62
1024	73
1536	89
2048	103

te võrdluseks. RSA ja diskreetse logaritmi põhiste süsteemide (ElGamal, DSA, Diffie-Hellmani võtmehavetus) turvataset võib sama võtmepikkuse juures pidada praktiliselt võrdseteks. Sealjuures on elliptilistele kõveratele tuginevate süsteemide vajadused võtmepikkuse mõttes tunduvalt madalamad, jäädes turvataseme b saavutamiseks suurusjärku $2b$.

Tabel 3 esitab ECRYPT2 RSA ja diskreetse logaritmi süsteemide enamlevinud võtmepikkuste ning ideaalse sümmeetrilise šifri turvataseme vastavuse hinnangud. Selle tabeli alusel saame leida näiteks RSA-1024 murdmiseks vajaliku investeeringu suuruse. Kasutades jaotises 2.3.2 toodud hinnangut, maksis 2^{61} arvutusoperatsiooni tegemine 2012. aastal ligikaudu $350. RSA-1024 murdmiseks vajalikud 2^{73} operatsiooni maksaksid siis

$$\frac{2^{73}}{2^{61}} \cdot \$350 \approx \$1.400.000 \; .$$

2.5.2 Soovitused

Aruande üks eesmärke on anda ka soovitusi sobivate krüptoalgoritmide kasutamiseks. Järgnevas esitatakse neli loetelu, kuhu on paigutatud selles jaotises käsitletud algoritmid. Esimene loetelu sisaldab primitiive, mille kasutamine on ebaturvaline juba täna. Teises ja kolmandas on toodud primitiivid, millest tuleks loobuda vastavalt kahe ja viie aasta aasta jooksul, ning neljanda loetelu primitiivide kasutamine on suure tõenäosusega turvaline ka üle aruande ajahorisondi ajahorisondi (st viie aasta). Aruande autorid mõistavad aga, et teatud primitiivide korral (näiteks mobiilsides kasutatavates krüptoalgoritmides) on nende kasutamist Eestis raske lõpetada vaid Eesti-siseste organisatsiooniliste meetmete abil.

Ebaturvalised primitiivid: DES, A5/1, A5/2, RC4, RSA-512, RSA-768 (ja diskreetse logaritmi põhised süsteemid mooduli pikkusega kuni 768 bitti), MD5, RIPEMD-128.

Primitiivid, mille kasutamine tuleks lõpetada 2 aasta jooksul: SHA-1, RSA-1024 (ja diskreetse logaritmi põhised süsteemid mooduli pikkusega 1024 bitti).

Primitiivid, mille kasutamine tuleks lõpetada 5 aasta jooksul: TDEA/3DES, Kasumi.

5 aasta jooksul turvalised primitiivid: Blowfish, AES (kõik standardsed võtmepikkused), RSA-2048 (ja diskreetse logaritmi põhised süsteemid mooduli pikkusega 2048 bitti), SHA-2, SHA-3, RIPEMD-160.

3 Krüptograafilised protokollid

Krüptograafilistest primitiividest turvaliste süsteemide ehitamiseks ei piisa. Neid primitiive tuleb ka õigesti kasutada, sest muidu võib juhtuda, et süsteemi saab rünnata primitiividest mööda minnes. Oluline krüptograafia kasutamise koht on süsteemikomponentide vaheline suhtlus ning eeskirjadeks, mis ütlevad, kus mingit krüptoalgoritmi rakendada, on krüptograafilised protokollid.

3.1 Protokollide turvagarantiide olemus

Krüptograafilised protokollid on märksa keerulisemad objektid kui eelmises peatükis vaadeldud primitiivid. Ka krüptograafias konstrueeritakse keerulisemaid objekte lihtsamatest. Lisaks konstruktsiooni loomisele püütakse ka tõestada, et kui konstruktsiooni osadena kasutatud primitiivid on turvalised, on ka konstrueeritav protokoll turvaline. Oluline on, et sellise tõestuse andmiseks pole tarvis midagi teada tegelikult kasutatavate lihtsamate objektide turvalisusest; turvatõestus (ehk *taandus*) annab ainult implikatsiooni.

Seetõttu peaks ideaalis iga krüptograafilise protokolli juurde kuuluma ka tema turvalisuse taandus protokollis kasutatavate primitiivide turvalisusele. Tegelikkuses on kõik see, mida protokoll lisaks primitiividele sisaldab – side osapoolte vahel, ründaja interaktsioon toimuva sidega, teostuse iseärasused – aga detailirikas, raskesti formaliseeritav ja käsitletav ning seetõttu on paljud protokollid esitatud ilma turvatõestusteta. Heal juhul on protokolli turvatõestus esitatud kunagi hiljem, halvemal juhul üksnes arvatakse, et protokoll on turvaline.

Küll aga järeldub eelmisest lõigust, et võrreldes primitiividega on protokollide jaoks võimalike turvaotsuste ulatus laiem. Kui primitiivi kohta saab teha ainult otsuse "ebaturvaline" või "pole teada, et ebaturvaline" (võimalik, et kvalifitseeritult selle primitiivi kasutusparameetrite võimalike väärtustega), siis krüptograafilise protokolli kohta võib otsustada ka seda, et ta on "turvaline, kui kasutatavad primitiivid on turvalised" või "ebaturvaline isegi siis, kui kasutatavad primitiivid on turvalised".

Krüptograafiliste protokollide turvatõestuste kasutamise üks põhjusi on krüptoprimitiivide turvadefinitsioonide keerukus. Nn. *arvutuslikus mudelis* [132] spetsifitseerib turvadefinitsioon selle ründe, mis ründajal peab ebaõnnestuma; sellel definitsioonil põhinevad turvatõestused peavad näitama, kuidas krüptoprotokolli vastu sooritatav rünne on teisendatav primitiivi turvadefinitsioonis kirjeldatavaks ründeks. Krüptoprotokollide turvalisuse põhjendusi peab lihtsustama *krüptograafia sümbolmudel* [121]. Mudel keskendub mitte sellele, mida ründaja teha ei suuda, vaid sellele, mida "mõistlikku" ta teha suudab. Mõned näited ründaja jaoks "mõistlike" operatsioonide kohta:

- genereerida uus juhuslik väärtus;
- olles omandanud (s.t. genereerinud, kinni püüdnud või arvutanud) kaks teadet, moodustada neist paar (või rakendada neile või neist ühele mõnda teist operatsiooni);

- omades krüptogrammi ja sellele vastavat võtit, leida vastav avatekst.

"Mõistlik" operatsioon ei ole näiteks ainult krüptogrammist ilma võtit teadmata teise krüptogrammi loomine, nii et vastavad avatekstid oleksid omavahel mittetriviaalsel viisil seotud. Teatud tegevuste klassifitseerimine "mõistlikeks" põhineb eeldusel, et kõik muud operatsioonid viivad ainult tähenduseta sõnumite tekkimiseni, mis on sellise sõnumi saaja poolt lihtsasti väljafiltreeritavad. Krüptograafia sümbolmudel eeldab, et ründaja sooritab ainult "mõistlikke" arvutussamme.

Krüptograafia sümbolmudel on osutunud mugavaks ja väga viljakaks abstraktsioonitasemeks krüptograafiliste protokollide analüüsil, mida tõendab lähenemisviiside paljusus. Analüüsimetooditena on pakutud paljude meetodite kohandusi, näiteks mudelikontrolli [181, 73, 196], andmevooanalüüsi [94, 205], formaalloogikat [100, 259, 111], tüübisüsteeme [50, 134, 135], resolutsiooni [91]. Väga oluline on ka see, et suur osa neist metoodikatest on osutunud piisavalt lihtsaks, et konstrueerida automaatseid või siis vähemalt poolautomaatseid analüüsivahendeid [187, 182, 243, 249, 58, 92], mida saab rakendada reaalsetel protokollidel. Selliste analüüsivahendite kasutamine tagab, et protokolli turvalisuse põhjendus ei sisalda hooletusvigu, kuid seal võivad olla vead, mis on omakorda tingitud vigadest analüsaatori lähtekoodis. Seetõttu tuleb loota, et analüsaatoril on piisavalt kasutajaid vigade kiireks avastatamiseks.

Kui protokoll on turvaline sümbolmudelis, siis mida tähendab see arvutuslikus mudelis? Esiteks, kui protokoll ei oleks arvutuslikus mudelis turvaline, peaks mõne kasutusel oleva krüptoprimitiivi või nende kombinatsiooniga olema võimalik sooritada mõni "ebamõistlik" operatsioon. Kuigi primitiivide turvatõestused seda ei välista, oleks see ikkagi väga märkimisväärne avastus. Küsimust, kas ja millal protokolli turvalisusest sümbolilises mudelis järeldub tema turvalisus arvutuslikus mudelis, on viimase kümne aasta jooksul uurinud mitmed tööd [53, 173, 194, 193, 109, 149, 108, 63, 64]. Tänaseks on saadud tulemused piisavalt põhjalikud ja võimaldavad väita, et kui krüptoprotokolli kasutab ainult enamlevinud primitiive (krüpteerimine, allkirjastamine, räsimine) ja ainult "mõistlikul" viisil, siis järeldub sümbolilisest turvalisusest arvutuslik turvalisus.

3.2 Autentimis- ja võtmevahetusprotokollid

Autentimisprotokolli kasutatakse juhul, kui kaks või enam olemit (isikut / arvutit / ...) soovivad kindlaks teha, et teine osapool (või -pooled) on hetkel olemas. Võtmevahetusprotokolli kasutatakse, kui olemid soovivad kokku leppida värske (seansi)võtme, mida ei tea keegi teine peale nende. Enamasti on kasutatavatel protokollidel nii autentimise kui ka võtmevahetuse eesmärk. See tähendab, et protokollis osalev olem saab nii teadmise, et teised olemid on elus (s.t. protokolli töö ajal aktiivsed), kui ka võtme, mida teavad ainult need teised olemid.

Et väita midagi teise olemi kohta, on tarvis tema kohta midagi teada. Tüüpiliselt on selleks teadmuseks olemi avalik võti, mis leitakse tema sertifikaadist. Sertifikaatide levitamise viisid jäävad selle aruande käsitlusalast välja. Siinkohal eeldatakse, et olemi nime järgi on võimalik leida tema kehtiv sertifikaat (mis on üks ja ainus, vähemalt käesolevas rakenduses) ning sellest tema avalik võti. Mõnes rakenduses võib olemi kohta teada olla midagi muud – näiteks tema parool või midagi, mis sellest sõltub.

Turvaline võtmevahetusprotokoll tagab, et võrguliiklust pealtkuulav või seda muutev ründaja ei leia seansivõtit, mida parajasti vahetatakse. *Tulevikuturvaline* protokoll tagab veel, et toimunud protokolliseansi logist ja osapoolte pikaajalistest saladustest (tüüpiliselt on need sertifikaatides olevatele avalikele võtmetele vastavad salajased võtmed) ei

ole võimalik leida seansivõtit. Tulevikuturvalisus on oluline näiteks juhul, kui kardetakse, et võrguliiklust logitakse ning tulevikus tekib raskusi pikaajaliste saladuste hoidmisega (näiteks toimub mingi matemaatiline läbimurre, mille tõttu osutuvad kasutatavad asümmeetrilised krüptoalgoritmid tulevikus oodatust nõrgemaks). Mitte kõik laialt kasutatavad võtmevahetusprotokollid ei ole tulevikuturvalised. Üldiselt põhinevad tulevikuturvalised protokollid Diffie-Hellmani võtmevahetusel (jaotis 2.2), kus sõnumite autentsust tagatakse digitaalallkirjadega.

3.2.1 SSL/TLS

Transport Layer Security (TLS) (värskeim versioon 1.2 [116]) ja tema eelkäija *Secure Sockets Layer (SSL)* on ilmselt levinuimad protokollistikud kahe osapoole vaheliseks autentimiseks ja võtmevahetuseks ning sellele järgnevaks turvaliseks suhtluseks (vt. jaotis 3.5). TLS spetsifitseerib võtmevahetusprotokolli (mille käigus toimub ka ühe või mõlema osapoole autentimine) ja transpordiprotokolli. Siinkohal vaatame võtmevahetusprotokolli, transpordiprotokolli käsitleb jaotis 3.5.

Võtmevahetusprotokollis (*handshake protocol*) saadavad osapooled teineteisele esmalt oma sertifikaadid ning lepivad kokku kasutatavates algoritmides. Seejärel saadab klient serverile ühe sõnumi ning võib-olla server kliendile ühe sõnumi (kui lepiti kokku Diffie-Hellmani võtmevahetuses), millest mõlemad osapooled suudavad arvutada ühe ja sama, kõigi ülejäänud isikute jaoks salajase väärtuse (*pre-master secret*). Võtmevahetus on sellega saavutatud ja server on kliendile autenditud. Klient on serverile autenditud vaid siis, kui kliendi sertifikaat võimaldab digitaalallkirjastamist ja server aktsepteerib seda sertifikaati.

Lisaks eelmises lõigus kirjeldatud struktuurile on TLS-protokollil veel palju valikuliselt rakendatavaid osi ning nende kõigi arvessevõtmine turvatõestuste juures on töömahukas (olgugi, et kontseptuaalselt võrdlemisi lihtne). Paulson [217] on kasutanud TLS-võtmevahetusprotokolli turvaomaduste tõestamiseks sümbolmudelis [121] formaalloogilisi meetodeid [216], tõestused on läbi viidud teoreemitõestusassistendi Isabelle/HOL [206] abil. Paulson uurib protokolli põhilist (kuni kaheksast sõnumist koosnevat) voogu ning näitab, et selle voo juures on võtmevahetusprotokoll turvaline. He jt. [140] näitavad protokollide komponeerimise loogika abil [111], et umbes samasuguse sõnumivoo turvaomaduste säilimist ka protokolli käitamisel suurema süsteemi alamprotokollina. Gajek jt. [130] näitavad, et TLS-i võtmevahetus- ja transpordiprotokoll koos võetuna võib abstraheerida turvalisi sideseansse; see abstraktsioon on korrektne krüptograafia arvutuslikus mudelis. Kahjuks erineb TLS-i transpordiprotokoll vähesel, kuid olulisel määral selles artiklis esitatud skeemist, ja selle erinevuse tõttu ei ole see abstraktsioon praktikas alati korrektne. Bhargavan jt. [86] analüüsivad üht konkreetset TLS-i teostust, mis on kirjutatud keeles F# [258], ning leiavad, et nii protokoll kui ka teostus on turvalised.

TLS-võtmevahetusprotokolli ja tema teostuste kohta on teada ka mitmeid nõrkusi, mis demonstreerivad viidatud turvaanalüüside piiri. Bleichenbacheri rünnet [93] kirjeldati jaotises 2.2.1.3. Klima jt. [160] näitavad, kuidas seda rünnet laiendada juhule, kus server küll realiseerib Bleichenbacheri ründe vastumeetmed, kuid kontrollib nähtavalt, kas kliendi saadetud *pre-master secret* sisaldab õiget versiooninumbrit. Need ei ole ründed protokolli enda, vaid seal kasutatava krüptoprimitiivi vastu. Augustis 2009 leiti aga rünne TLS-protokolli enda vastu, seoses sõnumijadaga, mida eelnevad analüüsid ei uurinud. TLS-protokollis on mõlemal osapoolel võimalik algatada uue *pre-master secret*'i kokkuleppimine. Sel hetkel on võimalik ründajal lisada enda valitud sõnum kliendi ja serveri

vahelisse kanalisse ning jätta mulje, et see tuli kliendilt [221]. Selle ründe vältimiseks lisati TLS-protokollile täiendus, mis selle autentsusprobleemi parandab [223]. See täiendus on realiseeritud OpenSSL-is alates versioonist 0.9.8m.

TLS-i võtmevahetus- ja transpordiprotokollide seni põhjalikema ja reaalsete kasutusviiside suhtes täpseima analüüsi on läbi viinud Krawczyk jt. [170]. Nad uurivad üheaegselt nii võtmevahetus- kui ka transpordiprotokolli (mis on teineteisega põimunud, sest võtmevahetusprotokolli viimased sõnumid vahetatakse loodud transpordikanali sees; see asjaolu muudab analüüsi märksa raskemaks). Nende analüüs võtab arvesse olemasolevaid vastumeetmeid näiteks Bleichenbacheri ründe [93] vastu. Nad leiavad, et kui transpordikihis on kasutusel turvaline krüptosüsteem ja rakendatud on meetmed, mis välistavad käesoleva dokumendi jaotises 3.5 kirjeldatud ründed, siis on TLS tõestatavalt turvaline. See tulemus on ehk isegi pisut üllatav, arvestades nõrkusi võtmevahetusel, kui kasutusel on krüpteerimine RSA-ga viisil, mis on tundlik Bleichenbacheri ründele. Leitakse, et protokolli teised osad on juhuslikult loodud sellistena, et nad varjavad neid nõrkusi. Selle „juhuslikkuse“ tõttu soovitavad tolle analüüsi autorid eelistada TLS-i võtmevahetusprotokollis Diffie-Hellmani protokollil põhinevaid variante. Käesoleva aruande autorid leiavad siiski, et ka RSA-l põhinevad variandid on piisavalt turvalised. Diffie-Hellmani protokollil põhinevad variante (EC)DHE-RSA, DHE-DSS, ECDHE-ECDSA [116, 90] tuleb aga kasutada siis, kui eesmärk on verifitseeritult [148] tulevikuturvaline võtmevahetus.

3.2.1.1 ID-kaardi kasutamine TLS-protokollis

Olemasolevat standardit PKCS #11 toetavad krüptoteegid lubavad TLS-võtmevahetusel kliendi võtmevahetussõnumi digitaalallkirjastamist kiipkaardis oleva salajase võtmega. Nii kasutatakse ka Eesti ID-kaarti.

Väärib rõhutamist, et eelkirjeldatud, kliendi autentimise vastu toimiva ründe [221] võimalikkus on ei sõltu vähimalgi määral sellest, kas ID-kaarti kasutatakse või mitte. Ründe ärahoidmiseks on tarvilik, et kliendi ja serveri kasutatav TLS-i teostus oleks selle ründe vastu kaitstud [223].

3.2.1.2 Mobiil-ID

Kui võtmevahetuse ajal jääb klient serverile autentimata, võib kliendi autentimine olla osa edasisest suhtlusest; levinud viis selleks on paroolide kasutamine. Kuna server on kliendi jaoks autenditud, siis võib klient loodud ühenduse kaudu saata parooli, mida ainult serveril on lubatud teada. Selles skeemis on nõrkused juhul, kui klient ei ole tähelepanelik ning on ühenduse loonud ründajaga, mitte soovitud serveriga. Sel juhul võib ründaja ise serveriga ühendust võtta ning kliendilt saadud parooli serverile edasi saata (klassikaline vahendusrünne). ID-kaarti kasutades ei ole selline rünne võimalik, sest ründaja ei ole võimeline kliendi nimel digitaalallkirju koostama ning kliendi koostatud allkirju (mis on mõeldud võtmevahetuseks ründajaga) server ei aktsepteeri.

Mobiil-ID on samuti protokoll kliendi autentimiseks. Selles protokollis püütakse klienti autentida sellega, et serveri genereeritud pretensioon (*challenge*) digitaalallkirjastatakse kliendi sertifikaadis olevale avalikule võtmele vastava salajase võtmega, mis asub kasutajale kuuluva mobiiltelefoni SIM-kaardil. Seega peab protokoll viima serveri väljakutse mobiiltelefonini ning seejärel veenduma, et klient soovis tõepoolest end sellele serverile autentida. Serveri ja mobiiltelefoni vahel suhtlemiseks kasutatakse veebiteenust DigiDocService [147], mida haldab AS Sertifitseerimiskeskus. Mobiiltelefon arvutab saadud

pretensioonist (millest ühe poole genereeris server ja teise poole DigiDocService) kontrollkoodi, mis koosneb neljast kümnendnumbrist. Sama kontrollkoodi saadab server ka kliendile, mis seda kasutajale näitab. Kasutaja võrdleb kaht kontrollkoodi ning nende kokkulangemisel annab mobiiltelefonile korralduse allkirjastada. Tervet protokolli kirjeldab DigiDocService'i spetsifikatsioon [147].

Mobiil-ID autentimisprotokolli turvaanalüüs [174] näitas, et tema tehnilises lahendustes on mitmeid nõrkusi, mis teatud juhtudel võivad ründajal lasta end autentida mõne teise kasutajana. Leiti, et DigiDocService, mis peaks olema kõigest vahendaja serveri ja mobiiltelefoni vahel, on protokolli praeguses versioonis usaldatud osapool. Probleem tuleneb järgmistest asjaoludest.

- Pretensiooni allkirjastamisel allkirjastatakse ainult pretensioon. Krüptoprotokollide ettevaatliku kavandamise põhimõtted [52] soovitavad juhul, kui digitaalallkiri on mõeldud kontrollimiseks mingile konkreetsele osapoolele, lisada allkirja alla selle osapoole nimi.
- Osa pretensioonist genereerib DigiDocService. Kuna kontrollkoodidel on ainult 10000 võimalikku väärtust, saab DigiDocService'i kontrollkoodide väärtusi vabalt valides kollisioone tekitada.
- Kontrollkoodi, mida klient kasutajale näitab, arvutatab välja veebiteenus DigiDocService, mitte server.

Lisaks leidis turvaanalüüs [174], et Mobiil-ID protokoll on vahendusrünnetele niisama haavatav kui parooliga autentimine.

Nende nõrkuste kõrvaldamine vajab muudatusi DigiDocService'i ning Mobiil-ID-d kasutavate serverite töös. Et muuta autentimiseks kasutatavat allkirjastamisoperatsiooni, tuleb välja vahetada ka SIM-kaartide peal jooksev tarkvara, mis töötleb Mobiil-ID protokolli teateid.

Kui mingi infosüsteem pakub Mobiil-ID protokolliga autentimist, siis peab infosüsteemi omanik täielikult usaldama DigiDocService'it. Autentimisel tuleb igal juhul usaldada ka sertifitseerimisteenuste andjat, kelle väljastatud sertifikaati infosüsteemi klient kasutab. Usaldusbaasi suurusest lähtudes soovitame, et kuni Mobiil-ID protokollis leitud nõrkuste kõrvaldamiseni tuleks kasutajate autentimisel aktsepteerida ainult AS Sertifitseerimiskeskuse väljastatud sertifikaate. Sel juhul ei suurenda vajadus usaldada DigiDocService'it autentimiseks vajalikku usaldatava töötluse baasi.

3.2.1.3 iPizza pangalink

Pangalink on 1990ndate aastate lõpul Eesti pankade poolt kasutusele võetud firmapärane standard, mis võimaldab teenuse andjail autentida kasutajaid internetipanga kaudu [212]. Pangalingi kasutamine on Eestis üsna laialt levinud, väga tihti kasutatakse seda teise võimalusena ID-kaardiga autentimise kõrval. Pangalingi tehnilist standardit kirjeldab näiteks Swedbanki dokument [257]. Pangalinki kasutamisel kuuluvad panga autentimise infosüsteemid (ja kogu nende usaldatava töötluse baas) teenuse usaldatava töötluse baasi hulka.

Protokollis iPizza vahetatakse sõnumeid HTTPS-protokolli kaudu. Edastatavad sõnumid on veel omakorda digitaalallkirjastatud ning selleks kasutatakse räsialgoritmi SHA-1 ning RSA PKCS #1 versiooni 1.5 [24] allkirjatäidist (ingl. k. *padding*). RSA võtmetena on toetatud kuni 4096-bitised võtmed.

iPizza protokolli erinevates teostustes on leitud vigu [180, 212], näiteks klassikaline vahendusrünne, mis lubab pahatahtlikul teenuseandjal enda autenditud kliendi nimel teise

teenusepakkujaga sessiooni algatada. Seetõttu soovitame Riigi Infosüsteemi Ametil nõuda teenuse pakkujailt ja kasutajailt nende rakendustest leitud probleemide kõrvaldamist.

Pikemas perspektiivis soovitame aga hakata kasutajaid autentima vaid ID-kaardi ja Mobiil-ID vahenditega, sest nõnda on usaldatava töötluse baas väiksem.

3.2.1.4 TUPAS pangalink

TUPAS on Soome finantsettevõtete liidu *Finanssialan Keskusliitto* loodud protokoll [128], mille abil saavad teenuseandjad autentida kasutajaid internetipanga kaudu. Protokolli kasutab Eestis Nordea pank.

Protokollis TUPAS edastatavatele sõnumitele on lisatud sõnumiautentimiskood (MAC), mis võimaldab teenuseandjal ja pangal kontrollida, kas sõnumit ei ole vahepeal muudetud ning sõnum on saabunud salajast võtit jagavatelt osapooltelt (teenuseandja ja pank). Ei kasutata krüptograafilist sõnumiautentimiskoodi HMAC, vaid salajane võti lisatakse räsitava stringi lõppu ning seejärel kasutatakse tavalist räsialgoritmi. Sellise MAC-funktsiooni nõrkused on teada [219, jaotis 4.2] ja ta võib ebaturvaliste räsialgoritmide kasutamisel võimaldada protokolli liiklust vahendaval kliendil panga ja teenuseandja vahelisi sõnumeid modifitseerida.

Räsimiseks saab protokollis kasutada algoritme MD5, SHA-1 ja SHA-256. Standardi kohaselt pidid pangad ja teenuseandjad 2011. aasta jooksul loobuma MD5 ja SHA-1 algoritmide kasutamisest ning võtma kasutusele SHA-256, kuid praktikas ei ole Eesti pankade teenustes sellist üleminekut või selle plaanimist täheldatud. Sõnumiautentimiskoodi salajane võti on vähemalt 256-bitine.

Sarnaselt iPizza protokolli realisatsioonidega on ka TUPAS protokolli realisatsioonidest leitud vigu [212]. Näiteks ei kontrolli paljud teenuseandjad sõnumite ühekordsust ning see võimaldab ründajal taasesitada salvestatud TUPASe sõnumit.

Soovitame teenuseandjatel nii kiiresti kui võimalik oma realisatsioon üle kontrollida ning vead parandada. Lisaks soovitame teenuseandjatel nõuda pankadelt turvaliste räsifunktsioonide kasutamist ning MD5 ja SHA-1 algoritmidest loobumist. Pikemas perspektiivis soovitame loobuda pangalingi teenuse kasutamisest ning autentida kasutajaid ID-kaardi või Mobiil-ID vahenditega, sest nendel juhtudel on usaldatava töötluse baas väiksem.

Soovitame pankadel parandada TUPAS protokoll ning võtta kasutusele krüptograafiline sõnumiautentimiskood HMAC.

3.2.2 IPsec (IKE)

IPsec on protokollistik võrguliikluse turvamiseks võrgukihis. See jaotis käsitleb IPsec-protokollistiku võtmevahetusprotokolle, peamiselt protokolli IKE (*Internet Key Exchange*).

IKE [153] avaldati esmakordselt 1998. aastal ja selles kasutatakse tüüpilist autenditud Diffie-Hellmani võtmevahetust, mille osapooled rakendavad omavahelistele sõnumitele (vt. jaotis 2.2.2.2) mingit meetodit nende tervikluse tagamiseks. Selleks meetodiks võib olla digitaalallkirjastamine või sõnumiautentimiskoodide kasutamine. Esimesel juhul peavad osapooltel olema sertifikaadid, teisel juhul peab neil juba varasemast olema mingi ühissaladus.

Võtmevahetusprotokolli IKE on formaalsete meetoditega uurinud näiteks Meadows [188] ning Canetti ja Krawczyk [102]. Protokoll ei ole kuigi keeruline. IKE järglaseks pakutud protokoll *Just Fast Keying (JFK)* [55] on samuti formaalsete meetoditega (proto-

kollianalüsaatoriga ProVerif [91]) analüüsitud [51]. Nende analüüside käigus on tõendatud ka IKE ja JFK tulevikuturvalisus.

3.2.3 Kerberos

Kerberos [204] on autentimis- ja võtmevahetusprotokoll, mis põhineb piletitel ning järgib ühekordse sisselogimise põhimõtteid. Kui klient soovib ühenduda mingi serveriga mingist domeenist, võtab ta esmalt ühendust selle domeeni autentimisserveriga. Autentimisserver tuvastab kliendi ning saadab talle pileti, milles sisaldub (sümmeetriline) seansivõti soovitud serveriga suhtlemiseks. Pileti abil autendib server kliendi. Autentimisserver võib kliendi tuvastada mitmeti: protokolli esimestes versioonides kasutati selleks jagatud võtmeid, aga protokolli laiendustes lubatakse kasutada ka avaliku võtme sertifikaate [274].

Butler jt. [101] analüüsisid Kerberost krüptograafia sümbolmudelis, leides, et üldjoontes on protokoll turvaline. Võib küll väita, et see analüüs ei ole otseselt Kerberosele kohaldatav, sest Kerberos kasutab krüptoprimitiive [220] viisil, mida ei loeta üldjoontes turvaliseks. Siiski on Boldyreva ja Kumar [95] näidanud, et protokollis Kerberos kasutatav *lihtsustatud profiil* CBC-töörežiimis plokkšifri kasutamiseks koos võtmelaiendusega [220, jaotis 5] konstrueerib turvalisest plokkšifrist turvalise autenditud krüptosüsteemi. Seega võib Kerberoses selle profiili kasutamist lugeda turvaliseks (erinevalt *üldisest* krüpteerimis- ja kontrollsummaprofiilist [220, jaotised 3 ja 4], mille turvalisuse üle otsustamiseks on meie teadmised hetkel puudulikud). Loomulikult tuleb seejuures kasutada turvalisi krüptoprimitiive.

3.2.4 SKIP

SKIP (*Simple Key-Management for Internet Protocol*) [61] oli üks võimalikke võtmevahetusprotokolle IPsec-protokollistikus enne IKE kasutusele võtmist. Protokolli SKIP kasutab X-tee [175] monitoorimisfunktsionaalsus.

Protokolli kasutamisel eeldatakse, et kõigi osapoolte Diffie-Hellmani võtmevahetuse avalikud võtmed on teistele osapooltele autentsel viisil teatavaks tehtud (näiteks sertifikaatide abil). Sel viisil on iga kahe osapoole jaoks määratud nende ühissaladus ning seda (või sellest tuletatud väärtust) saab kasutada pikaealise sümmeetrilise võtmena nende kahe osapoole vahel. Pikaealist sümmeetrilist võtit kasutatakse lühiealiste võtmete kokkuleppimiseks.

SKIP-võtmevahetusprotokoll on seega väga lihtsa struktuuriga ning lühiealise võtme kokkuleppimiseks polegi tarvis eraldi sõnumeid vahetada. Lihtsuse hind on aga vajadus lisada täiendavat teavet kõigisse transpordiprotokolli kaudu saadetavatesse IP-pakettidesse. Selline liiasus oli ka põhjus, miks SKIP-i ei valitud IPSec-protokollistikku [28].

3.2.5 WiFi võtmevahetusprotokollid

Kui WiFi võrgu [14] pääsupunkti ja lõppseadme vaheline ühendus on krüpteeritud, algab nendevaheline suhtlus mõne võtmevahetusprotokolliga autentimisraamistikust EAP (*Extensible Authentication Protocol*) [54]. Neid protokolle on sümbolmudelis analüüsitud [32] ning on leitud, et nad on turvalised võtmevahetusprotokollid. Selle protokolli rakendamise tulemusel lepivad pääsupunkt ja lõppseade kokku pikaajalise sümmeetrilise võtme, mida kasutades lepitakse kokku lühiealised võrguliikluse kaitsmise võtmed.

EAP peresse kuuluvad protokollid vajavad pääsupunktide ja lõppseadmete eelnevat konfigureerimist (sõltuvalt kasutatavast protokollist). Praktikas enim kasutatav, eeljagatud võtit kasutav protokoll EAP-PSK [85] nõuab, et pääsupunktis ja lõppseadmes oleks määratud üks ja sama paroolfraas. See paroolfraas peab olema piisavalt suure entroopiaga, et tema äraarvamine oleks ebatõenäoline. Selliste paroolfraaside jagamise hõlbustamiseks on välja pakutud protokoll *Wi-Fi Protected Setup* (WPS) [22], mida paljud pääsupunktid ja lõppseadmed toetavad. Protokolli turvalisus põhineb seitsmest kümnendnumbrist koosneva paroolfraasi (10 000 000 varianti) äraarvamise keerukusel, kui oletatava fraasi õigsuse kontrolliks tuleb läbi viia üks WPS-protokolliseanss.

Kahjuks on WPS teostatud nii, et kuni 10 000 000 variandi läbiproovimise asemel tuleb läbi proovida ainult kuni 11 000 varianti [265]. Seega on WPS ebaturvaline ja teda kasutada ei tohiks.

Disaini käigus tehtud eelduste või hiljem leitud rünnete tõttu saavutab nii mõnigi vanem autentimisprotokoll oma turvaeesmärgid (kliendi ja serveri identifitseerimise) ainult juhul, kui suhtluskanal kliendi ja serveri on turvatud. Mõni selline protokoll võib olla laialdases kasutuses ning tema väljavahetamine oleks väga kulukas. Selliste protokollide turvaliseks kasutamiseks on välja pakutud tunneldusmehhanism PEAP (*Protected EAP*) [152], kus kõigepealt moodustatakse TLS-ga turvatud tunnel ja seejärel vahetatakse pärandprotokolli teateid selle tunneli sees. Suuremates WiFi võrkudes on levinud lahendus, kus autentimisprotokollina on kasutuses MSCHAPv2 [276]. See protokoll on ebaturvaline, kui ründaja suudab kliendi ja serveri vahelist liiklust pealt kuulata [185]. Seetõttu jooksutatakse seda protokolli PEAP-i abil moodustatud tunnelis.

Tunneli otspunktiks oleva seadme jaoks on turvatud tunneli loomisel mõte ainult siis, kui ta kontrollib tunneli teise otsseadme identiteeti. See kontroll võib olla teostatud läbi sertifikaadi verifitseerimise. Vastasel juhul võib olla võimalik vahendusrünne, kus Wi-Fi pääsupunkt ja lõppseade ei loo tunnelit mitte omavahel, vaid kumbki loob turvatud tunneli ründajani [60]. Hiljuti on demonstreeritud, et sageli on WiFi lõppseadmes tõepoolest puudu tunneli teise otspunkti identiteedi kontroll [273, 248] ning sellised ründed on realistlikud. Rünnete teostuse autorid soovitavad PEAP-i kasutamine lõpetada, sest mõne teise autentimismeetodi (näiteks EAP-TLS) kasutuselevõtt on praktikas lihtsam, kui PEAP-i kasutamine turvalisel viisil. Käesoleva aruande autoritel ei jää üle muud, kui nende soovitustega ühineda.

3.2.6 Soovitused tulevikuks

Krüptograafiliste protokollide kavandamise hea tava järgi alustatakse taotletavate turvaeesmärkide (formaalsest) kirjeldamisest ning pärast protokolli loomist tõestatakse, et kõik seatud eesmärgid on saavutatud. Samuti tulevad kogu protsessile kasuks läbipaistvus ja avatus.

Kuivõrd aruande koostajatel pole olnud ligipääsu Mobiil-ID protokolli loomise üksikasjadele, ei saa me anda hinnangut tulemuse vastavusele taotletud turvaeesmärkidele. Seetõttu ei saa aruande autorid ka väita, et protokollil pole omadusi, mida tema loojad ette ei näinud.

Soovitame Mobiil-ID protokolli ümberkavandamist. Alustada tuleb soovitud turvaeesmärkide formaalsest kirjeldamisest, jätkata protokollisõnumite fikseerimisega ning lõpuks tõestada formaalselt (näiteks Dolev-Yao mudelis), et protokoll need turvaeesmärgid tõepoolest ka saavutab. Allikas [174] pakub ka võimalikke viise kõigi nende toimingute

läbiviimiseks. Selles allikas pakutud protokollimuutused eeldavad aga ka muudatusi kõigis Mobiil-ID-d kasutavates serverites.

Protokollide iPizza ja TUPASe kasutajatel soovitame võimalikult kiiresti parandada protokollides ja realisatsioonides olevad vead. Pikemas perspektiivis soovitame pangalingi protokollide kasutamisest loobuda ning autentida kasutajaid ID-kaardi ja Mobiil-ID lahendustega.

Protokolli SKIP vähese kasutatavuse tõttu pole uuringu autoreil õnnestunud leida selle protokolli turvaanalüüse. Seetõttu soovitame teha üht kahest:

- vahetada X-tees SKIP-protokoll mõne levinuma võrgukihi turvaprotokolli vastu;
- viia läbi SKIP-protokolli (X-tees kasutatavate valikute mahus) formaalne analüüs.

Samas ei ole see soovitus kõrge prioriteediga, sest SKIP ei ole X-tee jaoks missioonikriitline. Kui protokollis SKIP mingi tõsine nõrkus leitaks (mis iseenesest ei ole tõenäoline), siis oleks selle abil võimalik kõigest teada saada, milline asutus millist teenust kasutas, või siis X-tee haldajale teenusekasutuse kohta valeandmeid saata.

3.3 Arvuti ja ID-kaardi vaheline suhtlus

PKCS #11 [31] on standard, mis kehtestab rakendusprogrammide liidese kiipkaartidele ja teistele krüptograafilistele riist- ja tarkvaraelementidele. Standard kehtestab loetelu krüptograafilistest operatsioonidest, mida kiipkaart võib osata sooritada. Võimalike operatsioonide liike on palju, nende hulka kuuluvad

- uute võtmete loomine nii sümmeetriliste kui ka asümmeetriliste krüptosüsteemide jaoks,
- krüptograafiliste objektide (võtmete) sisestamine kaarti,
- andmete räsimine, krüpteerimine, dekrüpteerimine, digitaalallkirjastamine, verifitseerimine, sõnumiautentimiskoodide arvutamine,
- ühe võtme krüpteerimine teisega,
- krüpteeritud võtme dekrüpteerimine ja selle edaspidine kasutamine kaardis,
- kaardis sisalduva võtme modifitseerimine teatud viisil (XOR-operatsiooniga) ja muud operatsioonid kaardis salvestatud võtmetega.

Iga operatsiooni saab läbi viia erinevate krüptoalgoritmide ning nende parameetritega. Parameetrite hulka kuuluvad näiteks plokkšifri töörežiim, avateksti täidis jms. Iga krüptograafiline element võib realiseerida ainult osa standardist. Näiteks Eesti ID-kaart realiseerib väga väikese osa – ta on võimeline saadud sisendit teatud algoritmidega räsima ning kaardis sisalduvate RSA-võtmetega dekrüpteerima / allkirjastama / verifitseerima. Seejuures võib allkirjastamine / dekrüpteerimine toimuda kas ilma täidiseta (mehhanism RSA_X_509) või PKCS #1 standardi versioonis 1.5 [24] spetsifitseeritud täidistega (mehhanism RSA_PKCS) krüpteerimiseks (täidis RSAES-PKCS1-v1_5) ja digitaalallkirjastamiseks (täidis RSASSA-PKCS1-v1_5).

Standardi PKCS #11 rikkalikkus annab palju võimalusi rünneteks krüptograafiliste elementide vastu. Näiteks, kui mõni võti K kaardis on märgitud kasutatavaks nii dekrüpteerimisel kui ka teiste võtmete krüpteerimisel, on võimalik teised kaardis olevad võtmed leida, krüpteerides nad kõigepealt võtmega K (kaart tagastab krüptogrammi) ning seejärel saadud krüptoteksti võtmega K dekrüpteerides (kaart tagastab avateksti). Nende rünnete tulemuslikkust teatud kaartidel on ka praktikas kontrollitud [107, 114, 97].

Eesti ID-kaart realiseerib väikese osa PKCS #11-liidesest. Seetõttu ei ole kirjeldatud ründed talle rakendatavad. Aleksei Gornõi bakalaureusetöös [136] läbiviidud rünnetes rünnatakse mitte ID-kaarti, vaid arvutit. Küll aga lubab Eesti ID-kaart digitaalallkirjastamist ja dekrüpteerimist ühtede ja samade võtmetega (koos PKCS #1 standardis spetsifitseeritud täidise lisamise või eemaldamisega). Seetõttu on teoreetiliselt mõnes kontekstis, kus ründajal on õnnestunud panna ID-kaart dekrüpteerima tuhandeid teateid, võimalik Bleichenbacheri rünne [93, 68]. Ründaja jaoks ilmselt lihtsaim võimalus sellised dekrüpteerimised läbi viia on saada ligipääs ID-kaardile, mis on juba PIN1-ga aktiveeritud, või mille PIN2-e ründaja teab. Sel juhul on aga rünne mõttetu, sest selliselt ligipääsetava ID-kaardiga saab soovitud digitaalallkirja kohe luua. Siiski võib ette kujutada ka stsenaariume, kus rünnatav ID-kaart on mõne ründajast erineva agendi kontrolli all, kes millegipärast on nõus neid dekrüptimisi selle ID-kaardiga läbi viima.

ID-kaart toetab mehhanismi RSA_X_509, mis lubab sõnumeid ilma täidist lisamata või selle olemasolu kontrollimata digitaalallkirjastada või dekrüpteerida. Selline täidise mittekasutamine võib lihtsustada teatud ründeid kaardi vastu, mille olemasolust me hetkel küll midagi ei tea, kuid mille mitteleidumist ei ole võimalik ka tõestada (erinevalt näiteks OAEP-polsterduse [27] kasutamisel võimalikest tõestustest). Kui võimalik, siis soovitame RSA_X_509 mehhanismi toetamise ID-kaartides lõpetada.

PKCS #11-standard spetsifitseerib mitmeid krüptoalgoritme, mida kiipkaart või muu krüptograafiline element võib sisaldada. Uute algoritmide (näiteks SHA-3 või elliptilistel kõveratel põhinevate krüptosüsteemide) lisandumisel tuleb seda standardit täiendada.

3.4 Digitaalallkirjaprotokollid

Digitaalallkirjastamise üks eesmärk on *salgamistõrje* (*non-repudiation*): kui üks isik on mingi dokumendi allkirjastanud, peab allkirjastatud dokumendi saanud isikutel olema võimalus kolmandaid osapooli veenda, et esimene isik selle dokumendi tõepoolest allkirjastas. Ühtlasi on oluline ka *korrektsus* – kui esimene isik ei ole tegelikult mingit dokumenti allkirjastanud, siis ei tohi teised osapooled uskuma jääda, et ta seda tegi.

Et inimene ise ei ole tehniliselt suuteline digitaalallkirja koostama, delegeerib ta selle toimingu arvutile. Inimene peab arvutit usaldama, et see digitaalallkirjastaks õige dokumendi ja ei midagi muud. Samuti peab inimene usaldama seadet, mis talle dokumente näitab ja mille kuva põhjal ta teeb otsuse mingi dokument allkirjastada. Selline *usaldatud arvutusbaasi* küsimus kerkib digitaalallkirjastamisel alati. Järgnevad jaotised selgitavad, kui suur on see usaldatud baas eri protokollides. Eelistada tuleks protokolle, kus usaldatud baas on väiksem.

Eesti kontekstis on peamised digitaalallkirja andmise vahendid ID-kaart ja Mobiil-ID ning kasutatavad protokollid põhinevad DigiDoc-teegil [239]. Seetõttu eeldabki järgnev analüüs, et nii digitaalallkirja konstrueerimisel kui ka kontrollil kasutatakse nimetatud vahendeid. Eesti kontekstis võib olla oluline ka välisriigis moodustatud digitaalallkirja kontrollimine DigiDoc-vahenditega. Sel juhul on turvaotsuste tegemiseks siiski tarvis teada ka viisi, kuidas digitaalallkiri moodustati. Võib arvata, et kui moodus on sarnane Eestis levinutega, on ka saadavad turvagarantiid sarnased.

3.4.1 Allkirjastamine ID-kaardiga

Allkirjastamine arvutis ID-kaarti on võimalik digitaalallkirjastamiseks kasutada mitmel viisil. Kontseptuaalselt lihtsaim on DigiDoc-teegi kasutamine oma arvutis. Sellisel

juhul kutsub DigiDoc-funktsioone välja rakendus, mille ülesannete hulka kuulub allkirjastatava dokumendi näitamine kasutajale, selle dokumendi esitamine õiges vormingus ning edastamine DigiDoc-teegile. Too rakendus kuulub seega usaldatud arvutusbaasi.

DigiDoc-teek arvutab digitaalallkirjastatava dokumendi sõnumilühendi. Sõnumilühend edastatakse ID-kaardile, mis viib läbi astendamise RSA salajase astendajaga. Saadud väärtus lisatakse digitaalallkirja objektile. Samuti saab teeki kasutada OCSP kinnituste saamiseks moodustatud digitaalallkirjale. OCSP kinnituse küsimine toimub pärast digitaalallkirja moodustamist.

ID-kaardiga allkirjastamise protokoll on väga lihtne. Tema usaldatud arvutusbaasi kuuluvad DigiDoc-teek ning rakendus, mis seda teeki kasutab. Loomulikult kuuluvad usaldatud arvutusbaasi ka kasutaja arvuti ning ID-kaart. Arvuti ning ID-kaardi vahelist suhtlust reguleerib PKCS #11 [31] standard. Protokoll OCSP on samuti standarditud [203].

Protokoll OCSP See protokoll on määratud allkirjale kehtivuskinnituste küsimiseks. Moodustatud allkiri saadetakse sertifitseerimiskeskusele, mis tagastab allkirjastatud vastuse, kus öeldakse, kas esialgse allkirja moodustanud isiku sertifikaat on hetkel kehtiv. Kehtivuskinnitust võib küsida nii allkirja moodustaja kui ka keegi teine, näiteks verifitseerija. Protokoll on lihtne, koosnedes vaid päringust ja päringuvastusest. Protokoll võib olla tundlik taasesitusrünnete suhtes – kui allkirja verifitseerija teeb OCSP-päringu, siis võib talle vastuseks esitada mõne varasema vastuse, mis on sama sertifikaadi kohta käinud päringu kohta antud. Ründe vastu aitab nonsi (*nonce*) kasutamine päringus ja vastuses; see on samuti standardis [203] spetsifitseeritud ja DigiDoc-teekides realiseeritud.

Allkirjastamine DigiDoc-portaalis DigiDoc-portaalis digitaalallkirjastamiseks laadib kasutaja allkirjastatava(d) dokumendi(d) portaali. Seejärel käivitatakse kasutaja arvutis brauseriplugin (vt. jaotist 4.1.2.4), mille üks sisendparameeter on allkirjastatava dokumendikonteineri räsi. Plugin lisab räsile kasutatud räsialgoritmi algoritmiidentifikaatori ning rakendab tulemusele RSA-astendamist ID-kaardil oleva salajase võtmega. Astendamise tulemus saadetakse tagasi DigiDoc-portaalile, mis sooritab ülejäänud sammud allkirjastatud dokumendi loomiseks, sh sertifikaadi kehtivuskinnituse küsimise.

Kuna portaal sooritab enamiku allkirjastamiseks vajalikest sammudest, sh allkirjastatava räsi arvutamise, lisandub ta usaldatava töötluse baasile. Kui ründaja saavutab portaali üle kontrolli, on ta võimeline mis tahes dokumente allkirjastama, andes Javapluginale ette endavalitud räsiväärtuse.

3.4.2 Allkirjastamine Mobiil-ID-ga

Mobiil-ID-ga allkirjastamiseks saadetakse allkirjastatavad failid või nende räsid teenuseandja veebiteenusele, mis edastab allkirjastatavate andmete räsi allkirjastaja mobiiltelefonile ja kontrollkoodi teenusepakkujale või kasutajale. Näiteks DigiDoc3 rakendus edastab allkirjastatavate andmete räsi DigiDocService'i veebiteenusele, mis omakorda edastab selle allkirjastaja mobiiltelefonile ning DigiDoc3 kuvab kasutajale DigiDocService'i poolt tagastatud kontrollkoodi. Mobiiltelefoni SIM-kaardis töötav SIMToolkit rakendus näitab kasutajale räsi põhjal arvutatud kontrollkoodi, mida kasutaja peab (allkirjastatava dokumendi samasuses veendumiseks) võrdlema numbriga, mida näitab talle veebiteenuse poole pöördunud rakendus (näiteks DigiDoc3 klient, internetipank, eesti.ee portaal).

Nagu on kirjeldatud protokollianalüüsis [174], sisalduvad usaldatud arvutusbaasis sel juhul kasutaja arvuti, veebiteenus DigiDocService, kasutaja mobiiltelefon, SIM-kaart ning mobiilsideteenuse osutamiseks kasutatavad seadmed. Kui ründaja kontrollib mõnd arvutusbaasis sisalduvat seadet, on tal võimalik allkirjastamist häirida või keerukama ründega võltsida rünnatava isiku digiallkiri ründaja poolt määratud dokumendile.

3.4.3 Digitaalallkirja kontrollimine

Digitaalallkirja on võimalik kontrollida kas kasutaja enda arvutis või DigiDoc-portaalis. Kontrollimine võib sisaldada allkirjastamisel kasutatud sertifikaadi kohta OCSP-päringute tegemist. Portaalis allkirja kontrollimise protokoll on lihtne – kasutaja laadib allkirjastatud dokumendi portaali ning portaal vastab, kas allkiri kehtib või mitte. Loomulikult tuleb sealjuures portaali usaldada.

3.4.4 Soovitused tulevikuks

Allkirjastajail on soovitatav kasutada allkirjastamisviisi, mille puhul usaldatud arvutusbaas on võimalikult väike, arvestades muidugi ka rakendusele seatud teiste funktsionaalsete ja mittefunktsionaalsete nõuetega. See tähendab võimaluse korral dokumendi allkirja loomist omaenda arvutis oleva rakenduse abil. Muuhulgas oleks positiivne ka selline areng, et mingis veebikeskonnas (näiteks internetipangas) loodud dokumenti allkirjastades oleks kasutajal olemas valik, millise rakendusega (veebikeskkonnas või kasutaja arvutis töötava) dokument allkirjastatakse. Eriti teravalt ilmneb see probleem just internetipangas maksekorraldusi luues, sest kasutajal puudub kontroll selle üle, milline dokument tegelikult allkirjastatakse.

AS Sertifitseerimiskeskus soovitab, et digitaalallkirjastamist pakkuvad veebikeskkonnad võimaldaksid kasutajal tutvuda allkirjastatava dokumendiga enne ning loodud digitaalallkirjaga pärast allkirjastamist [39]. Kui veebikeskkond seda soovitust järgib (mitte kõik ei tee seda) ning kasutaja kontrollib, kas digitaalallkirjastati õige dokument, siis märkab ta, kui keskkond allkirjastatava dokumendi ära vahetas. Kuid ka see soovitus ei eemalda veebikeskkonda usaldatava töötluse baasist, sest valele dokumendile antud digitaalallkiri on olemas ka juhul, kui kasutaja pärast märkab, et ta soovis allkirjastada hoopis teist dokumenti. Valele dokumendile antud digitaalallkirja kehtetuks tunnistamine on ilmselt seotud õiguslike raskustega.

Eesti digitaalallkirjastamise standardite ja tarkvara tugi tuleks muuta modulaarsemaks. Toetatud räsi- ja allkirjastamisalgoritmide ning võtmepikkuste nimekirja muutmine tuleb teha lihtsamaks nii DigiDoc-standardis kui ka DigiDoc-teegis ja ilmselt siis ka veebiteenuses DigiDocService. Alternatiiv on loobuda nii DigiDoc-standardist kui ka -teegist võimalikult ruttu ning kasutada edaspidi ainult BDOC-i [2], kus kasutatavate algoritmide nimekiri ei ole kinnine.

Räsialgoritmide valiku osas on soovitav loobuda algoritmi SHA-1 toetamisest ning võtta kasutusele kas SHA-2 pere algoritmid. Hetkel ei ole alust arvata, et ühegi selle pere räsifunktsiooni vastu leitaks lähemate aastate jooksul tõsiseid ründeid (vt jaotis 2.3.3), seetõttu ei ole alust anda soovitust selle pere mingi konkreetse algoritmi eelistamiseks. Kui SHA-3 räsifunktsioonide pere saab rahvusvahelise kogukonna poolt tunnustatud standardiks, võib kaaluda üleminekut ka mõnele selle perekonna esindajale (vt jaotis 2.3.4).

Iga turvamehhanism on nii nõrk kui nõrk on tema nõrgim lüli. DigiDoc vormingus allkirjastatud dokumendi koostamisel kasutatakse räsialgoritme kõigepealt allkirjastatavate dokumentide räsimiseks ning seejärel DigiDoc-konteineri räsimiseks. Kui nendes kohtades

kasutada erinevaid algoritme, on allkirjade võltsimiskindlus piiratud neist kahest algoritmist nõrgema kollisioonikindlusega.

Lisaks soovitame, et kui allkirja verifitseerimisel kasutatakse sertifikaadi kehtivuse kindlakstegemiseks protokolli OCSP, siis tuleks taasesitusrünnete (*replay attacks*) vältimiseks lisada päringule nonss, nii nagu vastav standard seda nõuab ja nagu DigiDoc-teek seda ka teeb.

3.5 Transpordiprotokollid

Pärast turvalise võtmevahetusprotokolli rakendamist on selle protokolli osapooltel olemas ühine saladus, millest tuletada võti või võtmed edaspidise osapooltevahelise liikluse kaitsmiseks. *Transpordiprotokoll* kirjeldab, milliseid krüptograafilisi operatsioone tuleb rakendada saadetavatele andmetele nende konfidentsiaalsuse ja tervikluse tagamiseks.

Kuigi teoreetiliselt on teada, kuidas krüptoalgoritmidest ja sõnumiautentimiskoodidest konstrueerida turvalisi kanaleid [77, 78, 229], esineb transpordiprotokollides palju erinevaid konstruktsioone. Enamik neist on turvalised, kuid mingitel täiendavatel eeldustel, mida teoreetilised konstruktsioonid ei vaja.

Turvalised kanalid on paljude protokollide osad. Mõned neist protokollidest (näiteks HTTPS) on küllaltki keerulised ja sisaldavad optimeerimisi osapoolte vahelise suhtluse kiirendamiseks. Üks sellistest optimeerimistest on vahetatavate sõnumite automaatne pakkimine. Pakkimisel on mõtet ainult enne sõnumi sisenemist turvalisse kanalisse, sest hea krüptosüsteemi krüptogrammid ei ole eristatavad juhuslikust baidijadast (s.t. neid ei saa enam pakkida). Pakitud sõnumi pikkus sõltub tavaliselt selle sõnumi sisust. Transpordiprotokollid tavaliselt ei ürita varjata vahetatavate sõnumite pikkust. See loob süsteemidesse kõrvalkanali (mis tegelikult ei sõltu üldse kasutatavatest krüptograafilistest lahendustest), mis võib ründajale anda teavet sõnumi sisu kohta. See kõrvalkanal on eriti võimas siis, kui ründaja, kes huvitub sõnumite teatud osast, saab nende sõnumite mingeid teisi osi ise muuta [154]. Hiljuti on ilmunud ka selle ründe praktilised realisatsioonid HTTPS-protokolli (HTTP protokoll, mis on kaitstud SSL/TLS transpordiprotokolliga) jaoks [124, 74, 131].

Neist rünnetest lihtsamad [124] uurivad krüpteeritud HTTP-päringute pikkusi. Ründaja, olles saanud võimaluse teha kasutaja veebibrauseri kaudu päringuid serveritele (klassikaline päringuvõltsinguründe (*cross-site request forgery*) stsenaarium), lisab päringutesse oletuse selle kohta, milline võiks olla veebibrauseri ja serveri poolt kasutatav prääник (*cookie*), mis seda kasutajat identifitseerib. Brauser lisab päringule ka selle prääniku enda; päring pakitakse ja krüpteeritakse. Kui ründaja suutis ära arvata mingi osa prääникust, on pakitud päring lühem, kui siis, kui äraarvamine ei õnnestunud. Ründaja jälgib liiklust brauseri ja serveri vahel ning HTTP-päringute pikkuse järgi teeb järeldusi prääniku väärtuse kohta. Keerulisemate rünnete puhul[74, 131] jälgitakse serveri saadetud päringuvastuste pikkust. Sel juhul peab server päringuvastusesse panema ka ründaja oletuse, et see saaks mõjutada pakitud vastuse pikkust.

Nende rünnete eest pakub täielikku kaitset ainult pakkimisest loobumine nii SSL/TLS-transpordiprotokollis kui ka selle peal olevas rakendustaseme protokollis. Ainult HTTP-päringuid uurivate rünnete korral piisab päringute pakkimata jätmisest; päringute pakkimisest on loobunud brauserid Chrome ja Firefox: Internet Explorer pole neid kunagi pakkinud [133]. HTTP-päringuvastuseid uurivate rünnete vastu nii head kaitset ei ole, sest erinevalt päringutest peetakse päringuvastuste pakkimata jätmist enamasti liiga suureks hoobiks efektiivsusele. Gluck jt. ning Ristic [131, 224] pakuvad meetodeid ründe

edukuse vähendamiseks; nende rakendatavust tuleks käesoleva aruande autorite arvates hinnata juhtumipõhiselt.

3.5.1 SSL/TLS

SSL/TLS-i transpordiprotokollis (*record protocol*) [116] rakendatakse saadetavale paketile ja tema järjekorranumbrile kõigepealt sõnumiautentimiskoodi. Pakett ja kood konkateneeritakse ning neile lisatakse täidis, et teate kogupikkus oleks kasutatava plokkšifri plokipikkuse kordne. Täidisbaitide väärtus ja nende arv on omavahel seotud. Seejärel sõnum krüpteeritakse ja saadetakse. Krüpteerimiseks kasutatakse plokkšifrit (näiteks AES) CBC-režiimis või jadašifrit (näiteks RC4). Seejuures aga ei kasuta SSL/TLS protokollid kuni versioonini TLS v1.0 CBC-režiimi vastavalt tema definitsioonile, mis nõuab iga paketi krüpteerimisel uue initsialiseerimisvektori genereerimist ning sellega suurendab krüptogrammi pikkust. Selle asemel aheldatakse järjestikuste pakettide krüptogrammid kokku, võttes iga järgmise paketi krüpteerimisel initsialiseerimisvektoriks (IV) eelmise paketi krüptogrammi viimase ploki. Nagu allpool näeme, on sellisel muudatusel kaugeleulatuvad tagajärjed.

Selline autentimise ja krüpteerimise järjestus ei vasta täpselt teooriale, mis soovitab kõigepealt krüpteerida ning seejärel autentida. Teadaolevad juhud [77], kus see konstruktsioon nõrgaks osutub, on küll puhtteoreetilised. Samuti on näidatud, et kui krüptoalgoritmiks on plokkšiffer CBC-režiimis või jadašiffer, siis on ka kõigepealt autentiv ja seejärel krüpteeriv konstruktsioon turvaline [169]. See turvagarantii kehtib aga ainult „õige“ CBC-režiimi jaoks.

Kaua on teada olnud realistlik rünne [263], juhul kui transpordiprotokolli teostus vigaselt vormindatud paketi saamisel avalikustab, mis nurjas tema dekodeerimise – kas oli täidis väära pikkusega või ebaõnnestus sõnumiautentimiskoodi kontroll. Avalikustamine võib potentsiaalselt aset leida mitmeti. Otseseim viis on veateadete tagastamine. Kuid avalikustada võib ka aeg, mis kulub vigase paketi töötlemiseks: väära pikkusega täidise korral pole enam põhjust sõnumiautentimiskoodi kontrollida ja seega on töötlusaeg lühem. OpenSSL (alates versioonidest 0.9.6i ja 0.9.7a) realiseerib kaitse selle ründe vastu. Kahjuks näitavad hiljutised tulemused [56], et selle ajakanali korralik sulgemine ei olegi niisama lihtne. OpenSSL alates versioonidest 1.0.1d, 1.0.0k ja 0.9.8y sisaldab koodi, mille eesmärgik on tagada, et kõigi saabunud pakettide töötlus võtaks täpselt ühepalju aega; tänapäevaste protsessoriarhitektuuride keerukuse tõttu on see vägagi keerukas ülesanne.

Teoreetiliselt on SSL/TLS transpordiprotokolli kasutatava IV-aheldatud CBC-režiimi nõrkused teada olnud juba mitu aastat [66, 67]. Paari viimase aasta jooksul on aga ilmunud ka tegelikud ründed [123], mis neid nõrkusi ära kasutades suudavad leida avateksti, mida transpordiprotokoll kaitsma peaks. Kõige mõistlikum viis ennast selle ründe vastu kaitsta on mitte kasutada plokkšifrit IV-aheldatud CBC-režiimis. TLS alates versioonist v1.1 [115] seda režiimi enam ei toeta, seega soovitame mitte kasutada TLS-i vanemaid versioone. Samuti tuletame meelde, et kuigi BEAST-rünne [123] ei ole rakendatav jadašifritele, siis vaikimisi jadašifriks TLS-is on RC4, mida me samuti kasutada ei soovita (vt. jaotis 2.1.3).

3.5.2 IPsec-protokollistiku transpordiprotokoll

Encapsulating Security payload (ESP) [155] on võrgupakettide kodeerimisviis IPsec-protokollistikus. Muuhulgas kasutab seda kodeerimisviisi X-tee [175]. Sel viisil IP-paketi

sisu (või terve pakett, sõltuvalt kasutusrežiimist) kõigepealt krüpteeritakse ning seejärel rakendatakse krüpteeritud paketile sõnumiautentimiskoodi. See vastab teoreetilistele soovitustele [77]. Võtmevahetusprotokolli SKIP kasutamisel peab kodeeritud pakett täiendavalt sisaldama selle paketi krüpteerimiseks kasutatud võtit (mis on krüpteeritud pikaealise võtmega).

3.5.3 WiFi transpordiprotokollid

IEEE 802.11 (WiFi) traadita võrgu standard spetsifitseerib võrgutaseme transpordiprotokollid võrguliikluse kaitsmiseks. Neist esimeses – *Wired Equivalent Privacy (WEP)* [34, jaotis 8.2.1] – on mitmeid nõrkusi ning tänaseks on ta murtud. Leidub aktiivne rünne, mille abil võib kasutatava transpordivõtme leida vähem kui minutiga [261].

Protokolli WEP nõrkusteks olid muuhulgas jadašifri RC4 (jaotis 2.1.3) kasutamine, tema võtmejada ebakorrektne lähtestamine ja kasutu mehhanism andmepakettide tervikluse tagamiseks. Protokoll TKIP (*Temporal Key Integrity Protocol*) [34, jaotis 8.3.2] parandab WEP-protokolli mitmel viisil. Iga paketi jaoks kasutatakse eraldi krüpteerimisvõtit, mis genereeritakse pikaealisest võtmest. Samuti on terviklusmehhanismi parandatud, olgugi et see on ikka veel nõrk. Krüpteerimisalgoritmina on endiselt kasutusel RC4. TKIP on standarditud kui *Wi-Fi Protected Access (WPA)*.

WPA vastu on teada rünne [260, 209], mille abil on võimalik leida terviklusmehhanismis kasutatav võti, misjärel on ründajal võimalik teeselda suhtluse osapooleks olemist ning teisele osapoolele endavalitud sisuga pakette saata. Rünne ei võimalda krüpteerimisvõtmeid leida ega osapooltevahelist liiklust dekrüpteerida.

Protokolli CCMP (*Counter Mode with Cipher Block Chaining Message Authentication Code Protocol*) [34, jaotis 8.3.3] nimi sisaldab juba suure osa sellenimelise plokkšifrist andmepakettide kodeerimisviisi konstruktsiooni detailidest. Tema põhiosa on plokkšifri kasutamine CCM-režiimis [269]; lisaks kehtestab standard, kuidas teatud autentimisandmeid arvutada. CCMP, kus plokkšifriks on 128-bitise võtmega AES, on standardiseeritud kui *Wi-Fi Protected Access II (WPA2)*.

CCM-režiim on tõestatavalt turvaline [151] ning WPA2-kodeerimisviisi vastu teadaolevaid ründeid ei leidu. Küll aga on CCM-režiimi kritiseeritud tema vähese efektiivsuse, kontseptuaalse keerulisuse ja detailide peensuse tõttu [228].

WiFi liikluse turbeks tuleb alati kasutada mehhanismi WPA2, sest varasemad mehhanismid on ebaturvalised.

3.6 Soovitused tulevikuks

Selle aruande põhiline soovitus on kasutada tõestatavalt turvalisi protokolle, sest protokollide turvatõestusi osatakse esitada ning neist tulenevate tagatiste ulatusest saadakse aru. Kui infosüsteemis kasutatavatel protokollidel on turvaomadused, mida need infosüsteemid vajavad, siis ei ole tarvis neid protokolle tulevikus välja vahetada. Seega puudub vajadus infosüsteeme nii ulatuslikult modulariseerida, et näiteks terve võtmevahetusprotokoll koos oma loogikaga oleks lihtsasti vahetatav. Küll aga on tarvis, et protokollis kasutatavad krüptograafilised primitiivid ning nende võtmepikkused oleksid vahetatavad, mistõttu protokollide teostused peavad olema modulaarsed ning kasutatavad primitiivid ei tohi olla lähtekoodis püsistatud (*hard-coded*), vaid peavad jääma koodist sõltumatult konfigureeritavaks.

4 Krüptorakendused

4.1 Tarkvara

4.1.1 Protokollide teostused

See jaotis kirjeldab krüptoprimitiivide kasutust mõnes Eesti jaoks olulises protokollistikus ning analüüsib krüptograafiliste algoritmide vahetuse mõju vastavatele süsteemidele. Analüüs käsitleb üldkasutatavaid avalikke protokollistikke ning ID-kaarti ja sellega seotud tehnoloogiaid.

4.1.1.1 Krüptoteegid OpenSSL ja Bouncy Castle

OpenSSL [16] on protokollide SSLv2, SSLv3 ja TLSv1 ning vajalike primitiivide C-keeles kirjutatud teostus. OpenSSL-i arendab suur rahvusvaheliselt tunnustatud kogukond ning enam kui kümneaastase arendusperioodi tulemusena on saavutatud väga stabiilne, optimeeritud ja hästi modulariseeritud lähtekood. Tänu paindlikele Apache'i-laadsetele litsentsitingimustele on OpenSSL lihtsasti kasutatav nii äri- kui ka priivaratoodetes, ning C-põhistes süsteemides tuleb teda kindlasti eelistada krüptograafiliste primitiivide kohalikule teostamisele. OpenSSL sobib (vastavalt seadistatuna) ka sellistesse tarkvarasüsteemidesse, mille turvapoliitika näeb ette FIPS 140-2 [26] järgi valideeritud krüptomoodulite kasutamist.

Java- ja C#põhiste süsteemide puhul tuleb iseleiutamisele samuti eelistada suurema kogukonna arendatavat teeki. Üks selline on Bouncy Castle, mille litsentsitingimused (MIT-laadne litsents), stabiilsus ja suur krüptoprimitiivide hulk rahuldab enamiku praktikas esinevatest vajadustest. Toetatud primitiivide hulka kuuluvad AES, Blowfish, DES, IDEA, RSA, ElGamal, SHA-1, SHA-2 pere räsifunktsioonid jpt primitiivid; täieliku loetelu võib leida spetsifikatsioonist [6].

4.1.1.2 Apache SSL soovitatav konfiguratsioon

Veebiserveriga Apache kasutatakse enamasti moodulit mod_ssl [45], mis teostab TLS-protokolli, võimaldab teenindada HTTPS-põhiste URL-idega veebisaite ning autentida X.509 sertifikaatidega kasutajaid. Mooduli mod_ssl konfiguratsioonis on võimalik reguleerida, milliseid algoritme SSL-protokolli kokkuleppimisel server aktsepteerib ning seeläbi juhtida kliendiga kokkulepitava SSL-seansi turvalisust.

Vaikekonfiguratsioonis kasutatakse parameetri `SSLCipherSuite` väärtusena stringi "`DEFAULT`", mille puhul võivad mõned brauserid kokku leppida ka näiteks DES-krüpteerimisalgoritmi, mille kasutamist see aruanne ei soovita. Debian 7.0 distributsioonis kasutatakse vaikimisi mõnevõrra turvalisemat stringi "`HIGH:MEDIUM:!aNULL:!MD5`", kuid ka sellisel juhul lubatakse näiteks krüpteerimisalgoritmi RC4, mille kasutamist see aruanne heaks ei kiida.

Soovitame kasutada parameetri väärtusena stringi "`HIGH:!ADH:!SHA1`", mis lubab järgmisi algoritme:

1. võtmevahetusalgoritmid DH, RSA, ECDH, ECDSA,
2. krüptoalgoritm AES 128- ja 256-bitise transpordivõtmega,
3. räsialgoritmid SHA-256 ja SHA-384 HMAC funktsioonina.

Spetsiaalselt keelatakse anonüümne Diffie-Hellmani võtmevahetusprotokoll ning SHA1 räsialgoritmi kasutamine HMAC funktsioonina. Soovitame perioodiliselt lubatud algoritmide nimekirja üle vaadata, ebaturvalised algoritmid jõuga keelata ning järjestada lubatud algoritmid, arvestades serverite jõudlust ja turvaeesmärke.

Paršovs [213] analüüsib mod_ssl'i tüüpkonfiguratsioone ja Eesti teenuseandjate veebiserverite konfiguratsioone kasutajate autentimisel X.509 sertifikaatidega ning annab soovitusi teenuseandjatele oma serverite konfigureerimiseks (jaotis V-A viidatud analüüsis).

1. Konfiguratsiooniparameeter `SSLCACertificateFile` viitab failile, kus on toodud sertifitseerimiskeskuste endi allkirjastatud juursertifikaadid, mida veebiserver usaldab. Konfigureerimisel tuleb hoolitseda selle eest, et selles failis ei oleks liigseid sertifikaate. Eesti ID-kaartide toetamiseks piisab käesoleva aruande kirjutamise ajal neljast sertifikaadist: *ESTEID-SK 2007*, *ESTEID-SK 2011*, *Juur-SK* ja *EE Certification Centre Root CA*.
2. Failis, millele viitab konfiguratsiooniparameeter `SSLCADNRequestFile`, olgu need sertifitseerimiskeskuste sertifikaadid, mida otseselt kasutatakse kliendisertifikaatide väljastamiseks. Selle faili olemasolu on oluline, sest vastasel juhul saadab veebiserver kliendile TLS-võtmevahetusprotokolli `CertificateRequest`-sõnumi oma usaldatud juursertifikaatide omanike nimed, avalikustades seega selle nimekirja.
3. Veebirakenduse sees tuleb kontrollida, kas kliendi sertifikaat, millega edukas võtmevahetus läbi viidi, on tõepoolest väljastatud sertifitseerimiskeskuse sertifikaadiga, mis on mõeldud kliendisertifikaatide väljastamiseks. Seda kontrolli mod_ssl ise ei tee; ta kontrollib ainult, kas leidub usaldusahel kliendisertifikaadist usaldatud juursertifikaadini.
4. Konfiguratsiooniparameeter `SSLVerifyDepth` määrab, kui pikki usaldusahelaid kliendisertifikaadist usaldatud juursertifikaadini mod_ssl lubab. Eesti ID-kaartide toetamiseks piisab selle parameetri väärtusest 2, mida ka soovitatakse. Selle parameetri suuremad väärtused ei ole küll otsene oht, kuid võimaldavad mõningaid valesti väljastatud sertifikaatidel põhinevaid ründeid [171]. Selle parameetri eriti suured väärtused võimaldavad teenusetõkestusründeid veebiserverite vastu.
5. Tuleb tagada, et võtmevahetusprotokolli seanss oleks võimalikult lühike. Selleks tuleks kasutada Apache'i moodulit mod_reqtimeout ning konfigureerida ta nii, et liiga pikad protokolliseansid kindlasti katkestataks, isegi kui selles seansis mingisugune võrguliiklus toimub.
6. Kliendisertifikaadi kehtivust tuleb kontrollida. Kui sertifitseerimiskeskus annab välja sertifikaatide tühistusloendeid, siis tuleb neid ka kasutada. Seevastu AS Sertifitseerimiskeskuse LDAP-teenuse kasutamine sertifikaadi kehtivuse kontrolliks oleks selle teenuse väärkasutus [218] ning samuti ebaturvaline, sest suhtlus LDAP-serveriga on turvamata.
7. Kui kasutaja logib end välja, tuleb tema brauserile teada anda, et seanss on lõppenud ja järgmisel võtmevahetusel tuleb sertifikaadi kasutamiseks kasutajalt uuesti

nõusolekut küsida. See teavitus toimub eri brauserites erinevalt, vt. [213, jaotis III-I].

8. Tundlike teenuste korral nõuda kliendi sertifikaadiga autentimist kogu kasutajaseansi vältel, mitte ainult kliendi esialgsel tuvastamisel (mille õnnestumisel seotaks seanss ainult HTTP-präänikuga).
9. Kui veebiserveri kasutamisel on oluline klientide anonüümsus võrguliikluse pealtkuulajate eest, siis tuleb kliendi sertifikaati küsida mitte esialgsel võtmevahetusel, kus see sertifikaat saadetaks serverile krüpteerimata kujul, vaid uue seansivõtme kokkuleppimise ajal. Selleks tuleb konfiguratsiooni parameeter `SSLVerifyClient` määrata mitte üleserverilises kontekstis, vaid kataloogi kohta.

4.1.2 ID-kaardi baastarkvara

ID-kaardi baastarkvara [49] koosneb järgmistest komponentidest.

- libdigidoc – DigiDoc-tüüpi failihalduse teek. Sisaldab funktsioone [245] failide allkirjastamiseks, allkirjastatud failidele OCSP-kinnituse võtmiseks, allkirjade kontrollimiseks, andmete krüpteerimiseks ja dekrüpteerimiseks vastavalt XML-ENC standardile [144]. Uutes rakendustes tuleks selle teegi asemel kasutada teeki libdigidocpp [47].
- libdigidocpp – BDOC- ja DigiDoc-tüüpi failihalduse teek. Sisaldab funktsioone failide allkirjastamiseks ja allkirjade kontrollimiseks [262]. Krüpteerimisfunktsioone ei ole. On kirjutatud keeles C++.
- JDigiDoc – DigiDoc- ja BDOC-tüüpi failihalduse teek, mis on kirjutatud Javas. Teek võimaldab faile nii allkirjastada kui ka krüpteerida (samuti dekrüpteerida ning digitaalallkirju verifitseerida) [246].
- Brauseriplugin esteid-plugin dokumentide allkirjastamiseks brauseri kaudu.
- ID-kaardi draiver Windows-i uuemate versioonide jaoks.
- ID-kaardi utiliit.

Loetletud komponendid kasutavad järgmisi üldlevinud teeke:

- OpenSC [41] on standardi PKCS #11 [31] teostus. See standard kirjeldab rakendusliidest kiipkaartidega suhtlemiseks. OpenSC ei kasuta niivõrd ise krüptograafiat, kuivõrd vahendab krüptograafilisi päringuid rakenduse ja kaardi vahel.
- PC/SC spetsifikatsiooni [37] teostus. See komponent korraldab arvuti ja kaardi(lugeja) vahelist suhtlust. Komponendil ei ole krüptograafilisi võimeid.
- OpenSSL.
- Teegid XML-i loomiseks ja parsimiseks.
- Teegid pakitud failide loomiseks ja parsimiseks.
- Teegid käivitatava koodi dünaamiliseks laadimiseks.

Järgnevas kirjeldame krüptoprimitiivide kasutamist neis komponentides ja nende konfigureeritavust. Samuti püüame anda soovitusi, mida neid komponente kasutades või edasi arendades võiks silmas pidada. Ülevaate alus on komponentide lähtekood [49]; libdigidoc- ja libdigidocpp-teekide ning esteid-plugina analüüs põhineb tarkvaraversioonil 3.7.2, JDigiDoc-i analüüs versiooni 3.8 beeta. Koodi on analüüsi käigus küll loetud, kuid mitte käivitatud.

4.1.2.1 libdigidoc

See teek kasutab krüptograafilisi primitiive järgmistel digitaalallkirjastamisega seotud juhtudel.

OCSP-kontrollipäringute koostamine. Päringu koostamiseks kasutatakse OpenSSL-teeki, kuid algoritmide identifikaatorid antakse ette libdigidoc-teegis. Konkreetselt antakse ette räsialgoritmi identifikaator (SHA-1), mida kasutatakse päringukonteineri räsimiseks enne selle allkirjastamist. Allkirjastamiseks ning võtme lugemiseks PKCS #12 [25]-standardile vastavast konteinerist kasutatakse samuti OpenSSL-i, nii et võimalike digitaalallkirjaalgoritmide tugi sõltub ainult OpenSSL-ist.

OCSP-päringuvastuste kontroll. Digitaalallkirja kontrolliks kasutatakse OpenSSL-i. Kontroll, kas saadud päringuvastus üldse kehtib vastava sertifikaadi kohta, sooritatakse libdigidoc-is. Selleks räsitakse sertifikaat fikseeritud räsialgoritmiga (SHA-1), seejuures kontrollimata, kas päringuvastuses on kasutatud seda algoritmi või mõnda teist.

Digitaalallkirjastamine. Digitaalallkirjastamisel moodustatakse kõigepealt konteiner, kuhu lisatakse allkirjastatavate failide sõnumilühendid (räsid) ning vajadusel teised allkirja juurde kuuluvad atribuudid. Sellest konteinerist arvutatakse enne allkirjastamist omakorda sõnumilühend. Teek võimaldab esimesena mainitud sõnumilühendit arvutada räsialgoritmiga SHA-1 või SHA-256. Viimast sõnumilühendit arvutab ta algoritmiga SHA-1. Viimane sõnumilühend edastatakse ID-kaardile RSA-astendamise läbiviimiseks.

Digitaalallkirjade verifitseerimine. Teek eeldab, et verifitseeritakse RSA-allkirju, kus dokumentide räsimise algoritm on SHA-1 või SHA-256 ning konteineri räsimiseks kasutatakse algoritmi SHA-1. RSA-allkirja verifitseerides kasutatakse OpenSSL-i madala taseme funktsioone, mis allkirja väärtusest lähtudes taastavad dokumendi räsi. Seda räsi võrreldakse dokumendi räsiga. Teek toetab ka elliptilistel kõveratel põhinevaid allkirju (ECDSA [13, 1]), mida kontrollitakse OpenSSL-i funktsioonidega.

Fikseeritud algoritmide identifikaatorid on programmikoodi sisse kirjutatud, nii et nende muutmine või konfigureeritavaks tegemine nõuab koodi läbivaatust. Samuti on mitmes kohas programmikoodi sisse kirjutatud krüptograafilise algoritmi (peamiselt räsifunktsiooni) väljundjada eeldatav pikkus. Seetõttu ei tarvitse teegi uuendamisel piisata kõigi nende kohtade läbivaatamisest, kus esineb string `sha1` või `SHA1` või `SHA256`.

Teek libdigidoc kasutab krüptograafilisi primitiive ka järgmistel krüpteerimisega seotud juhtudel.

Transpordivõtme(te) genereerimine. Teek libdigidoc kasutab standardset hübriidse krüpteerimise meetodit, kus dokument krüpteeritakse sümmeetrilise võtmega, see aga dokumendi vastuvõtja avaliku võtmega. Transpordivõti tuleb iga kord uuesti genereerida. Teegis libdigidoc on fikseeritud see, et sümmeetrilise krüptoalgoritmina kasutatakse 128-bitise võtmega AES-i ahelarežiimis (*Cipher Block Chaining (CBC) mode*) [17, 271]. Võtme genereerimiseks küsitakse operatsioonisüsteemilt juhuslike baite, mis segatakse OpenSSL-i pseudojuhuarvude generaatori olekusse. Seejärel küsitakse pseudojuhuarvude generaatorilt juhuslikke baite koguses, millest jätkuks nii võtmele kui ka ahelarežiimi lähtestusvektorile. Neid baite ei kasutata aga otse võtme ja lähtestusvektorina, vaid hoopis sisendandmetena OpenSSL-teegi

funktsioonile `EVP_BytesToKey`, mille ülesanne on konstrueerida võti argumentidena etteantud paroolist ja soolast (*salt*). Konstruktsioon kasutab ka räsifunktsiooni, milleks libdigidoc'is on fikseeritud MD5.

Krüpteerimine. Krüpteerimisel kasutatakse kõigepealt genereeritud transpordivõtit, et krüpteerida dokument, ning seejärel saaja sertifikaadist saadud avalikku võtit, et krüpteerida transpordivõti. Dokumendi krüpteerimiseks kasutatakse 128-bitise võtmega AES-i ahelarežiimis. Transpordivõtme krüpteerimiseks kasutatakse RSA-d. libdigidoc ei sea piiranguid RSA võtme pikkusele. Transpordivõtme krüpteerimisel kasutatakse standardi PKCS #1 versioonis 1.5 [24] spetsifitseeritud täidist.

Dekrüpteerimine. Dekrüpteerimisel leitakse kõigepealt transpordivõti ja seejärel dokument, tehes krüpteerimisel läbiviidavate operatsioonide pöördoperatsioone.

Transpordivõtmete genereerimise lahenduses on mitmeid puudusi.

1. Pseudojuhuarvude generaatori väljund on juba otse kasutatav võtmena (ja lähtestusvektorina), nii et `EVP_BytesToKey` kasutamine on tarbetu ja kasutu (kuid mitte kahjulik).
2. Funktsioon `EVP_BytesToKey` instrueeritakse kasutama MD5-räsifunktsiooni, mida peaks igasugustes krüptograafilistes rakendustes vältima.
3. `EVP_BytesToKey` teostab PKCS #5 standardis [33] esitatud võtme- ja lähtestusvektori genereerimise meetodit PBKDF1, mis eeldab, et räsifunktsiooni väljundjada pikkus on vähemalt niisama suur kui võtme ja lähtestusvektori pikkused kokku. Siintoodud juhul see nii ei ole (MD5 väljundjada pikkus on 128 bitti, AES-i võtme pikkus on kirjeldatud juhul samuti juba 128 bitti ning lähtestusvektori pikkus samuti 128 bitti). Sellisel juhul on `EVP_BytesToKey` käitumine ebastandardne ning teaduskogukonna poolt korralikult analüüsimata. Siiski, käesoleva aruande autorid arvavad, et `EVP_BytesToKey` kasutamine selle koha peal sellisel viisil ei põhjusta ebastandardsusest tulenevaid nõrkusi, sest `EVP_BytesToKey` on kasutusel ainult krüptimisvõtme, mitte aga lähtestusvektori loomiseks. Krüptimisvõti pole aga pikem kui kasutatava räsifunktsiooni väljundi pikkus.

PKCS #1 v1.5 täidis on teatud olukordades ebaturvaline [93, 68]. Olgugi et dokumenditranspordi kontekstis need riskid ilmselt ei realiseeru, peab dekrüpteerimisrakenduses olema ettevaatlik, et pahatahtlikule saatjale mitte teada anda, kas vastuvõtja leidis krüpteeritud dokumendist õige täidisega transpordivõtme. Kui vastuvõtja kasutab dekrüpteerimiseks oma ID-kaarti, siis on selle võimaluse välistamine ID-kaardi ülesanne (vt. ka jaotist 3.3). On küllalt tõenäoline, et ID-kaart seda ei tee – kui transpordivõtme täidis ei ole korrektne, siis väljastatakse veateade, mille libdigidoc logib.

4.1.2.2 libdigidocpp

See teek kasutab krüptograafilisi primitiive samadel digitaalallkirjastamisega seotud juhtudel, mis ülalvaadeldud libdigidoc. Räsifunktsioonide kasutamine teegis on tunduvalt paindlikum: teegi lähtestusel laaditakse konfiguratsioonifail, kus on määratud, millist räsifunktsiooni dokumentide allkirjastamisel kasutada. Digitaalallkirjade verifitseerimisel toetatavate räsifunktsioonide ja allkirjastamismeetodite valik on suurem, lisandunud on SHA-2 perekonna räsifunktsioonid. Kõik toetatud räsifunktsioonid on ühes päisefailis loetletud; see teeb algoritmide nimekirja haldamise lihtsaks. Räsifunktsioonide teostused tulevad OpenSSL-teegist. Käesoleva aruande autorite arvates on see lahendus päris hea.

Nimetatud konfiguratsioonifaili kasutatakse küll ainult BDOC-tüüpi failide loomisel. DigiDoc-tüüpi failide töötluseks kasutatakse libdigidoc-teeki, kus kasutatavad räsifunktsioonid on programmikoodis fikseeritud.

Digitaalallkirjade verifitseerimisel antakse märku, kui dokumendi sõnumilühend on arvutatud räsifunktsiooniga, mille turvalisuses võiks kahelda. Loetelu sellistest räsifunktsioonidest ja allkirjastamismeetoditest on peidus ühe meetodi teostuses. Et see loetelu on ainult ühes kohas, on selle haldamine lihtsam; oleks aga veelgi parem, kui see loetelu oleks mõnes päisefailis. Ka ei ole see loetelu täielik – temasse kuulub küll RSA-SHA1 allkirjastamismeetod, kuid puudub ECDSA-SHA1.

OCSP-päringute koostamine ja verifitseerimine toimub OpenSSL teegi abil. Päringu räsimiseks kasutatav räsifunktsioon SHA-1 on programmikoodis fikseeritud.

4.1.2.3 JDigiDoc

JDigiDoc [246] on Java-keeles teostatud DigiDoc- ja BDOC-tüüpi failide haldamise teek, mis kasutab krüptograafilisi operatsioone läbi `javax.crypto` liidese. Ajatemplite ning OCSP-päringute ja vastuste töötlemiseks kasutab JDigiDoc teeki Bouncy Castle [5]. Sarnaselt libdigidoc-teegiga on ka JDigiDoc kasutatav failide allkirjastamiseks, digitaalallkirjade kontrolliks, failide krüpteerimiseks ja dekrüpteerimiseks. Sarnaselt libdigidocpp-teegiga määratakse kasutatavad räsifunktsioonid konfiguratsioonifailis.

JDigiDoc-teek sisaldab ka dokumentide krüpteerimise funktsioone. Sarnaselt libdigidoc-iga kasutatakse transpordivõtme krüpteerimiseks samuti PKCS #1 v1.5 täidist, mis võib teatud tingimustel osutuda nõrkuseks. Teisi täidiseid ei lubata, aga koodis on olemas kohad, kus seda saab konfigureerida. Transpordivõti genereeritakse standardsete vahenditega krüptoteegist. Koodis on fikseeritud, et transpordivõti on 128-bitine. Kasutatav võtme genereerimise algoritm ja krüpteerimise algoritm on fikseeritud koodis klassiväljade väärtustena ning on vajadusel kergesti muudetavad. Millegipärast ei saa neid algoritme määrata konfiguratsioonifailis.

4.1.2.4 esteid-plugin

C-keeles kirjutatud brauseriplugin *esteid* lubab brauseri kaudu dokumente allkirjastada [218]. Plugina funktsiooniks on brauserilt saadud dokumendiräsile ID-kaardiga digitaalallkirjastamine. Plugin lisab saadud räsile räsifunktsiooni identifikaatori, edastab selle ID-kaardile ning tagastab brauserile ID-kaardi arvutatud digitaalallkirja.

Seega plugin ise räsifunktsioone välja ei kutsu, küll aga peab ta nende identifikaatoreid teadma. Samuti peab ta aru saama, millise räsifunktsiooniga on moodustatud allkirjastatav dokumendiräsi. Räsifunktsiooni identifitseerib plugin räsi pikkuse järgi, eeldades et kasutatud on SHA-1 või SHA-2 peresse kuuluvat funktsiooni. Muude räsifunktsioonide lisandumisel võib selline meetod olla ebapiisav. Käesolev aruanne soovitab lisada allkirjastamismeetodile täiendava parameetrina räsifunktsiooni identifikaator.

4.1.2.5 ID-kaardi utiliit

Teek sisaldab funktsioone mitmesuguste operatsioonide sooritamiseks kiipkaardiga; krüptograafiline võime teegis peaaegu puudub. Nende operatsioonide seas on ka kaardil olevate RSA-võtmetega RSA-astendamised. Need operatsioonid saavad argumendina dokumendi räsi ning tagastavad digitaalallkirjastatud räsi. Teegis olevad funktsioonid eeldavad, et see

räsi on moodustatud SHA-1 või mõne SHA-2 peresse kuuluva räsifunktsiooniga. Muude räsifunktsioonide lisandumisel võib see koht teegi lähtetekstis vajada läbivaatust.

4.2 Riistvara

4.2.1 ID-kaart

Eesti ID-kaart põhineb firma Infineon toodetaval tüüpkiibil "JCLX80JTOP20ID", millele Trüb Baltic AS on kirjutanud (Eesti ID-kaardile spetsiifilise) Java rakendusprogrammi.[5] Lähitulevikus kavatsetakse üle minna firma Infineon uuemale tüüpkiibile "jTOP ID on SLE 78", kus on teostatud turvalisemad krüptoalgoritmid.

Tüüpkiibis on teostatud krüptoalgoritmid DES, 3-DES, RSA ja korpusel $GF(p)$ elliptiliste kõverate krüptograafia. Krüptoalgoritmid on rakendusprogrammidele kättesaadavad Java Card 2.2.2 operatsioonisüsteemi vahendusel. Eesti ID-kaardile vajalikud funktsioonid on teostatud Java programmina, mille Trüb Baltic AS saab laadida kaardi muutmällu (75 kilobaiti).

Uuemas (lähituleviku) kiibis on 2048-bitine RSA, kuni 512/521-bitised elliptilised kõverad, TDEA/3DES, 256-bitine AES ning kuni 512-bitised SHA-2 pere räsifunktsioonid.

Krüptoalgoritmid on teostatud riistvaras ja neid muuta ei saa. Muuta (uuesti laadida) saab vaid Java-rakendust. Rakenduse hoolduseks tuleb teada iga kaardi kohta eraldi kaardispetsiifilist haldusvõtit, mis on Trüb Baltic AS käes ja on kuulutatud Eesti riigisaladuseks. Seega saab ID-kaardi rakendusprogrammi muuta vaid Trüb Baltic AS osalusel. Täpsemalt kirjeldab haldustoiminguid ning nies kasutatavaid krüptograafilisi mehhanisme ID-kaardi spetsifikatsioon [59].

Kiibis on teostatud rohkem algoritme kui ID-kaardi rakendus tegelikult kasutab, seega on võimalik ID-kaardi poolt kasutatavaid algoritme (teostatud hulga piires) muuta ka lihtsalt Java-rakendust ümber programmeerides. Uue rakenduse laadimine on tehtud võimalikuks ainult Politsei- ja Piirivalveametis (mis loob selleks Trüb Baltic AS-iga RSA-2048 abil turvatud SSL/TLS ühenduse) ning selleks toiminguks kulub 5-7 minutit kaardi kohta.

Digitaalallkirja ja autentimise privaatvõtmed genereeritakse ID-kaardi sees, kasutades kiibisisest juhuarvude generaatorit, mis on sertifitseeritud ja saab vajaliku entroopiahulga kiibi sees toimuvatest olemuslikult juhuslikest füüsikalistest protsessidest. Tehasest tulnud kiibid personaliseerib Trüb Baltic AS vastavalt Politsei- ja Piirivalveametilt saadud korraldustele ning kaardi sees genereeritakse vajalikud RSA võtmed. See protseduur võtab aega kuni 5 minutit kaardi kohta. Avalikud võtmed lähevad Sertifitseerimiskeskus AS-le sertifikaadi väljaandmiseks. See toimub AES256-ga krüpteeritud SSL/TLS kanali vahendusel.

Trüb Baltic AS enda poolt hallatavat krüptoalgoritme sisaldavat koodi on väga vähe. Trüb Baltic AS saab ajakohast ründeteavet kiibi tootjalt (firmalt Infineon), kes jälgib uut teavet külgrünnete kohta ja informeerib pidevalt Trüb Baltic AS-i, jagades vajadusel ka soovitusi nende rünnete kahjuliku mõju vähendamiseks (näiteks rakenduskoodi muutmiseks). Infineon on Trüb Baltic AS-ile andnud ka rakendusprogrammide kirjutamise juhised, mille eesmärk on külgrünnete mõju vähendamine. Rakenduskood (Java) ei ole küll otseselt salastatud, kuid ka mitte avalik dokument.

Trüb Baltic AS ise informeerib pidevalt Eesti riiki võimalikest uutest turvaohtudest.

[5] Kogu jaotis põhineb intervjuul Trüb Baltic AS esindajaga.

Krüptograafiliste algoritmide murdumise mõju

Kui mõni ID-kaardil aktiivses kasutuses olev krüptograafiline primitiiv muutub ebaturvaliseks enne kaardi kehtivusaja lõppu, tuleb selle kasutamine lõpetada. Eeldusel, et kaardi riistvara pakub ka turvalist alternatiivi, tähendab see kaardil asuva Java-programmi vahetamist. Kuna seda tuleb teha füüsiliselt kontrollitud keskkonnas Politsei- ja Piirivalveameti teeninduspunktis, toob see endaga kaasa ameti potentsiaalse ülekoormuse aktiivse uuendamise perioodil ning ebamugavusi ID-kaardi omanikele. Samuti tuleb arvestada vajadusega piirata ID-kaarti kasutavate teenustele juurdepääsu nende kaartide jaoks, millel on vajalikud uuendused tegemata. See toob endaga kaasa nende teenuste käideldavuse languse.

4.2.2 Mobiil-ID

Mobiil-ID ei põhine ainulaadsel kiibil – reeglina muutub kiip umbes iga 2-3 aasta järel, olenevalt tarnijast. Põhjused on nii majanduslikud (kui nõudlus konkreetsele kiibile väheneb, siis hind tõuseb ning tarnijad vahetavad kiibi välja) kui ka tehnilised – enamasti on uuematel kiipidel rohkem võimalusi (näiteks NFC, rohkem mälu, kiirem protsessor jne). Aruande koostamise ajal (sügisel 2013) on välja töötamisel järgmise põlvkonna Mobiil-ID ning seetõttu võib termin Mobiil-ID siinses aruandes tähistada kaht lahendust – vana ja uut.

Krüptograafiliselt arhitektuurilt sarnaneb Mobiil-ID kiip ID-kaardi kiibiga: on olemas riistvaras teostatud primitiivid, mida saab välja kutsuda Java-rakendusest. Põhimõtteliselt on võimalik krüptograafilisi algoritme realiseerida ka rakenduse tasemel ja seda on ka kaalutud, kuid kiibitootjad on oma arendustöödes nõudlusele järele andnud ning laiendanud riistvaras pakutavate primitiivide hulka. Hetkel kasutab Mobiil-ID kaart asümmeetrilist algoritmi RSA1024, uue põlvkonna kaartidel on valikus RSA1024, RSA2048 ja ECC256.

Pärast kiibi füüsilist valmimist on selle püsimälu täidetud vastavalt kiibi tarnija spetsifikatsioonile. Tavaliselt asuvad püsimälus SIM-kaardi operatsioonisüsteem ning süsteemsed rakendused. Isikustamise käigus salvestatakse kiibile muu vajaminev informatsioon – failistruktuur, spetsiifilised rakendused (sh Mobiil-ID rakendus) –, seatakse ligipääsuõigused jne. Pärast isikustamist on võimalik isikustatud informatsiooni uuendada vastavalt algsel isikustamisel määratud õigustele, sh OTA (*over-the-air*) protokolliga.

Isikustamise käigus luuakse ka võtmepaarid. Vanas Mobiil-ID-s on käigus kaks RSA1024 võtmepaari, üks autentimiseks ja teine allkirjastamiseks. Uude lahendusse kavandatakse kuut võtmepaari:

- autentimise võtmepaar, RSA1024,
- allkirjastamise võtmepaar, RSA1024,
- autentimise võtmepaar, RSA2048,
- allkirjastamise võtmepaar, RSA2048,
- autentimise võtmepaar ECC256,
- allkirjastamise võtmepaar ECC256.

Isikustamise käigus loodud võtmepaaride kasutuselevõtt ja vahetamine on lihtne, piisab teenuseandja päringus võtmepaari identifikaatori määramisest. Olukorras, kus kaart toetab füüsiliselt näiteks elliptiliste kõverate krüptograafiat, aga isikustamise käigus on loodud ainult RSA1024 võtmepaar, on ECC võtmeid võimalik genereerida hiljem ka OTA

kaudu, nii et klient ei pea kuhugi minema. Vajalik on ainult mobiilioperaatori poolne tegevus.

Kui kõik kaardi poolt toetatud krüptograafilised primitiivid murduvad, tuleb halvemal juhul välja vahetada Mobiil-ID kiibid, mis tähendab klientidele vajadust mobiilioperaatori esindusse minna. Põhimõtteliselt on võimalik OTA kaudu kiipidesse laadida ka uus rakendus, mis realiseerib uued primitiivid tarkvaras, kuid selline teostus jääks kardetavasti liiga aeglaseks.

Praegu kasutusel oleva Mobiil-ID korral genereeritakse võtmed isikustamise käigus välises riistvaralises turvamoodulis. Avalikud võtmed kirjutatakse faili ning edastatakse mobiilioperaatorile. Mobiilioperaator saadab vastavad autentimise ja allkirjastamise avalikud võtmed Mobiil-ID teenuse avamisel Sertifitseerimiskeskusse, mis genereerib sertifikaadid.

Uues Mobiil-ID lahenduses on võimalik ka võtmepaaride genereerimine otse kiibil.

Praeguses Mobiil-ID-s on krüptograafilised algoritmid kasutusel veel mitmes elutsükli järgus.

- Väljundfailid (avalike võtmete ja OTA võtmetega) saadetakse mobiilioperaatorile, kasutades 4096-bitise võtmega PGP krüpteeringut.
- OTA platvormi andmebaasis on OTA võtmed krüpteeritud sümmeetrilise algoritmiga.
- Mobiil-ID rakendusega suhtlemisel (autentimis- ja allkirjastamipäringute saatmisel) OTA platvormi kaudu kasutatakse 3DES-i. Kõik uuendused, mis Mobiil-ID rakendusele saadetakse (näiteks tekstimuudatused vms), kasutavad hiljemalt 2014 aasta lõpuks AES-krüpteeringut (OTA kaudu).
- Mobiilsideoperaatorite ja Sertifitseerimisekeskuse vahelises ühenduses, mille kaudu saadetakse sertifitseerimis- ning allkirjastamis ja autentimispäringuid, kasutatakse krüpteerivat VPN-i (L2TP/IPSEC).

Uue põlvkonna Mobiil-ID kiibi platvormi nõue on tugi ka plokkšifrile AES.

4.2.3 Digi-ID

Digitaalne isikutunnistus ehk Digi-ID on riiklik digitaalne dokument, millega saab elektroonilises keskkonnas oma isikut tõendada ja anda digitaalallkirja [12].[6] Erinevalt ID-kaardist ei trükita Digi-ID-kaardile isiku nime, fotot ega muud vajalikku, et seda ei saaks kasutada visuaalse isikut tõendava dokumendina.

Krüptograafia seisukohast sarnaneb Digi-ID mõneti ID-kaardile ning Digi-ID sertifikaatide väljastamisel kasutab Sertifitseerimiskeskus samu üldpõhimõtteid [240], ESTEID-kaardi sertifitseerimispoliitikat [242] ning ESTEID sertifikaatide profiili [238].

Digi-ID haldusprotseduurid on samuti lähedased ID-kaardi vastavatele protseduuridele. Rakenduste laadimiseks ja personaliseerimiseks kasutatakse 3DES krüpteerimisalgoritmi.

Digi-ID kiip erineb siiski ID-kaardi kiibist ja tema tootja on Keycorp. Kiibis on MultOS operatsioonisüsteem IE4. Kiibis on teostatud ainult 1024-bitine RSA allkirjastamisalgoritm.

Digi-ID kaartide personaliseerimine on võimalik vaid Politsei- ja Piirivalveametis. Trüb Baltic AS personaliseerib kiipkaardid vaid osaliselt. Politsei- ja Piirivalveamet personaliseerib kaardid täielikult ja vahetab PIN-koodid.

[6] Kogu jaotis põhineb intervjuul Trüb Baltic AS esindajaga.

4.3 Teenused ja taristu

4.3.1 DigiDocService

DigiDocService on protokollil SOAP põhinev veebiteenus, mida pakub AS Sertifitseerimiskeskus [147]. Teenus on oluline eelkõige seetõttu, et tema kaudu toimub isikutuvastus ja digitaalallkirjastamine Mobiil-ID-ga. Sama teenuse abil on võimalik ka ID-kaardiga digitaalallkirju anda ning allkirju kontrollida.

Teenuse teostuse lähtekood on Sertifitseerimiskeskuse kontrolli all ning seal uuendatav. Aruande autorid ei ole seda lähteteksti näinud. Teenuse spetsifikatsioonist [147] selgub aga, mida selle teenuse kasutajad peaksid silmas pidama, kui kasutatavad krüptoalgoritmid muutuvad.

Mobiil-ID-ga autentimisel allkirjastab kasutaja telefon 160-bitise juhusliku sõnumi, millest poole on genereerinud autentimist nõudev server ning teise poole DigiDocService veebiteenus. Digitaalallkirja pikkus võib muutuda, kui kasutusele võetakse pikemad võtmed. Juhusliku sõnumi pikkus (160 bitti) on selles rakenduses piisav.

Allkirjastamisel tuleb veebiteenusele saata päringuga kas allkirjastatavad failid või nende räsid. Failide räsimisel on ainus toetatud räsialgoritm SHA-1. DigiDoc-konteineri räsimise algoritm ei ole DigiDocService'i spetsifikatsioonis täpsustatud. See veebiteenus tekitab aga dokumente DigiDoc-vormingule vastavalt, mis lubab vaid SHA-1 räsialgoritmi.

Veebiteenuse DigiDocService kohandamise enamate krüptoalgoritmidega peab ära tegema selle teenuse haldaja. Selle aruande kirjutajail ei ole juurdepääsu DigiDocService'i lähtekoodile, seega ei ole võimalik hinnata, kui keeruline see töö on. Kuna teenus on veebiteenus, piisab muutuste tegemisest ainult ühes kohas.

DigiDocService'is uute (räsi)algoritmide kasutuselevõtu keerukuse kontekstis tuleb arvestada, et DigiDocService'i spetsifikatsioon ei täpsusta, millist räsialgoritmi kasutab sisemiselt tema teostus, kui enne ID-kaardi või Mobiil-ID-ga RSA astendamist on tarvis leida DigiDoc-konteineri räsi. Seega võiks DigiDocService'i uuendamisel olla uute räsialgoritmide kasutuselevõtt üpris valutu.

4.3.2 Digitaalne tempel

Digitaalne tempel [11] on Sertifitseerimiskeskuse (SK) teenus, mille abil saavad ettevõtted anda dokumentidele digitaalallkirju. Sellega lisatakse dokumendile kinnitus, et dokument on pärit allkirjastanud asutusest ning et dokumenti ei ole vahepeal muudetud. Tüüpiliselt kasutatakse seda massiliste dokumentide tarbeks (arved, maksekorraldused, kinnitused, tunnistused jne.) Teenuse tellimisel väljastab SK ettevõttele kiipkaardil või USB-krüptopulgal X.509-sertifikaadi. Digitaalse templi kasutamisel luuakse andmete allkirjastamisel DigiDoc- või BDOC-vormingus konteiner täpselt samamoodi nagu ID-kaardi või Digi-ID kasutamisel andmete allkirjastamiseks.

Uurides aruande koostamise käigus SK poolt välja antud digitaalse templi sertifikaate, tuvastati järgmist:

- sertifikaatide kehtivusaeg on umbes 1 kuni 3 aastat,
- sertifikaatide allkirjastamisel kasutatakse algoritmi sha1withRSA ja 2048-bitist RSA võtit,
- väljaantavate sertifikaatide RSA võtmete pikkus on 1024 või 2048 bitti, sertifikaate 1024-bitisele võtmetele on väljastatud veel 2012. aastal.

Siinne aruanne soovitab algoritmide SHA-1 ja RSA-1024 kasutamisest loobuda nii kiiresti kui võimalik. Kindlasti ei tohi 1024-bitistele RSA võtmetele enam uusi sertifikaate väljastada. SK on avaldanud lubaduse tühistada kõik kehtivad 1024-bitistele võtmetele antud sertifikaadid 2013. aasta lõpuks [241]; käesoleva aruande koostamise hetkel (oktoobris 2013) ei ole seda veel tehtud.

4.3.3 Asutus pangandussektorist

Kuna pangandussektoris liigub suurel hulgal informatsiooni, on Asutusel selle haldamiseks terve rida infosüsteeme, mille arv (sõltuvalt sellest, mida täpselt infosüsteemiks lugeda) ulatub sadadesse. Valdav osa süsteemidest kasutab krüptograafilisi mehhanisme kõigis standardsetes komponentides – sidekanalite kaitsmiseks, kasutajate autentimiseks jne. Päris krüpteerimata on vaid mõned ühendused printeriserverite ja printerite vahel.

Hindamaks krüptoalgoritmide murdumise ja vahetamise mõju Asutusele, tuleb vahet teha asutusesiseste ja välisliidestusega süsteemide vahel.

Kui peaks tekkima vajadus mõni krüptograafiline baaskomponent kõigis asutusesisestes süsteemides täielikult välja vahetada, võib see halvemal juhul tähendada kuni pool aastat kõigi sisesüsteemide tarkvaraparandusi, testimist ja juurutamist. Asutusesisesed süsteemid pole aga krüptograafiliste primitiivide murdumise seisukohast nii kriitilised. Sisevõrgu kaitseks on olemas täiendavad mehhanismid, siseosalusega rünnete puhul pole aga krüptograafilistest meetmetest nagunii eriti palju kasu. Sellest johtuvalt võib Asutus probleemi ilmnemisel tegeleda sellega võrdlemisi rahulikult.

Välisliidestusega süsteemide puhul on olukord teine. Kui peaks selguma vajadus vahetada näiteks TLS šifrikomplekti, suudetakse see läbi viia umbes paari tunni, maksimaalselt ühe tööpäeva jooksul. Uuem TLS pole aga alati parem, näiteks on Asutusel esinenud olukord, kus uuendatud TLS-i versioon tuli taas vanema vastu vahetada, sest brauser Chrome ei suutunud uue versiooni abil ID-kaarti toetada.

Kliendiseansside turvamisel on kasutusel RSA2048, kuid Asutuse esindaja sõnul peavad serverid seepärast tegema keerukamaid krüptograafilisi arvutusi kui RSA1024 kasutamisel. See kergendab oluliselt teenusetõkestusrünnete läbiviimist, mistõttu võiks lühiajaliste sessioonide turvamisel piisata ka RSA1024 poolt pakutavast turvatasemest. Käesolev aruanne niisugust lahendust siiski ei soovita.

Asutuses ei ole eraldi inimesi, kelle vastutusalas oleks järelevalve konkreetsete krüptoalgoritmide ja nende kasutamise üle. Küll on olemas turbeosakond, keda kaasatakse vajadusel arendustöödesse ning kerkivate probleemide lahendamisse. Ka krüptonõrkuste teavitus pole süsteemselt korraldatud, kuid selle järele pole ka otsest vajadust, sest informatsioon uute rünnete kohta liigub erialaportaalides väga kiiresti ning asutuse spetsialistid puutuvad sellega tahes-tahtmata kokku. Probleemseks võivad osutuda unikaalsed süsteemid, mida Eestis on võibolla vaid mõni üksik ning mille tarnija ei suhtu oma toote kohta käiva turbeteabe levitamisse piisavalt hoolikalt. Sellised juhtumid on siiski pigem erandlikud ning näitavad vähenemistrendi.

Asutus kasutab ka mitut majasisest süsteemi, kus kasutatav krüptograafiline lahendus on välja töötatud kohapeal. Ka need lahendused on pigem erandlikud ning vajadus nende järele kahaneb sedamööda, kuidas standardsed komponendid vajalikke kasutusjuhtumeid järjest paremini katavad.

4.3.4 Veebiteenust pakkuv asutus

Käesolevas jaotises vaatleme ühe konkreetse veebiteenust pakkuva asutuse juhtumit. Asutuse identiteeti pole konfidentsiaalsusnõuete tõttu võimalik avaldada, kuid aruande autoritel on põhjust arvata, et krüptograafiliste meetodite kasutamise seisukohast on tal võrdlemisi tavaline veebiinfosüsteem.

Süsteem autendib kasutajaid ID-kaardi, mobiil-ID ja pangalinkide kaudu ning kasutab ühenduste krüpteerimiseks protokolli HTTPS. HTTPS seansside loomise ja turbega tegeleb Apache veebiserveri moodul, mille jaoks lubatavate šifrikomplektide kirjeldused võetakse vastavast konfiguratsioonist. Konfiguratsiooni muutmine on võrdlemisi lihtne, vajades ühe süsteemiadministraatori tööd tunni-paari ulatuses, lisaks tuleb arvestada teenuse taaskäivitamisest tuleneva lühikese katkestusega.

Hetkel lubatud šifrikomplektide hulk on küllalt suur, ja nende hulgas on ka potentsiaalselt nõrku krüptoprimitiive. Peamine põhjus, miks päevapealt ei saa üle minna ainult tugevatele primitiividele, peitub klientide brauserites. Kõigi klientide juurdepääsu tagamiseks tuleb arvestada väga vanade brauserite (nt Internet Explorer 6), aga ka kõikvõimalike marginaalsete brauserite ja nende beetaversioonidega. Kui lubada turvapoliitikaga ainult tugevaid šifreid, ei pruugi mõned brauserid neid toetada, mis omakorda tähendab, et osa kliente ei saa pakutavat e-teenust tarbida.

Lahendusena võib riik kehtestada e-teenustele minimaalsed turvanõuded. Sel juhul võiks asutus neile viidates ebaturvalised ühendused blokeerida; omaalgatuse korras pole niisugune klientide eemaletõrjumine aga mõeldav.

Enamik asutuse sideturbe tarkvara-komponente on standardsed. Samuti on ID-kaardi ja mobiil-ID osas kogu tarkvara standardne, autentimisteenus ja põhirakendus asuvad eri serverites.

Eraldi töötajat, kelle ülesannete hulka kuuluks regulaarselt rakenduste krüptokonfiguratsiooni ülevaatamine, ei ole. Küll hooldatakse tarkvara pidevalt ja tehakse vajalikke uuendusi muu töö käigus. Ka eraldiseisvat turvanõrkustest teavitamise mehhanismi ei ole kehtestatud, kuid süsteemiadministraatoritel on ametijuhendist tulenev kohustus püsida kursis oluliste erialaste arengutega, mis kaudselt katab ka krüptoalgoritmide hetkeolukorda.

Krüptonõrkusi puudutav informatsioon levib süsteemiadministraatorite seas suuliselt ja meililisti vahendusel, ka turvajuhi üks ülesandeid on seda teavet vajadusel levitada. Vestlusel asutuse esindajatega koorus muuhulgas välja mõte, et riik võiks (näiteks CERT-i kaudu) pakkuda meilina iganädalast või igakuist ülevaadet olulisematest uudistest krüptograafia ja infoturbe vallas.

4.3.5 X-tee

X-tee turvaserverite vahelise suhtluse krüpteerimiseks kasutatakse OpenSSL-i šifrikomplekti `EDH-RSA-DES-CBC3-SHA`, mille komponendid on

- Diffie-Hellmani võtmevahetusprotokoll,
- 2048-bitine RSA autentimiseks,
- 168-bitise võtmega 3DES side krüpteerimiseks,
- SHA-1 sõnumiautentimiskoodi arvutamiseks (vt ka jaotist 4.3.6).

Seiresõnumite edastamisel on võtmevahetusprotokollina kasutusel SKIP.

Kuigi esialgu on teadaolevad ründed SHA-1-põhiste sõnumiautentimiskoodide vastu teoreetilist laadi, soovitavad nii käesolev aruanne kui Ecrypt II 2012. aasta väljala-

se [62] SHA-1 kasutamisest loobuda nii kiiresti kui võimalik. Kuna käesoleva kirjutamise ajaks (sügis 2013) on krüptoteegid TLS 1.2 standardi teostamisega lõpule jõudnud, soovitame X-tee järgmises versioonis kasutada TLS 1.2 standardi šifrikomplekti `TLS_DHE_RSA_WITH_AES_256_CBC_SHA256`.

Krüptograafiliste algoritmide murdumise mõju

X-tees kasutatakse krüptograafilisi algoritme mitmesugustes komponentides ja sellest tulenevalt on ka algoritmide vahetamiseks vajalik arendustöö erinev. Kui tehtav muudatus piirdub ainult TLS šifrikomplekti vahetusega, saab tarkvara paranduse välja lasta mõne tunniga. Kõigi komponentide läbivaatus võib võtta suurusjärgus mõned kuud (vt juhtumianalüüsi jaotises 4.3.6).

Uute algoritmide kasutuselevõtuks ei piisa pelgalt tarkvarauuendustest, need tuleb ka installeerida. Kuna X-tees on umbes 200 turvaserverit ning nende haldurite võimekus on väga erinev, kestab X-tee uusversioonile üleminek enamasti mitu aastat. Hetkel on ainus sunnimeetod asutuse X-teest välja lülitamine, see aga ei tarvitse olla võimalik asutuse poolt pakutavate teenuste kriitilisuse tõttu.

Soovitused kahjuliku mõju vähendamiseks

Põhiline probleem X-tee üleviimisel turvalisemate krüptoalgoritmide kasutamisele on organisatsiooni inerts. Kui asutus teab, et teda ei ole ilma tema teenuseid tõkestamata võimalik X-teest välja lülitada, puudub tal motivatsioon turvaserverit uuendada. Niisiis on X-tees uute krüptoalgoritmide kasutuselevõtu kiirendamiseks vaja tõsta eeskätt organisatsioonilist võimekust ja rakendada väljalülitamisest pehmemaid sunnimeetmeid, nt sätestada uuendamiskohustus lepinguliselt, viia sisse rahalised sanktsioonid jms.

4.3.6 Juhtumianalüüs: X-tee räsifunktsioonide vahetamine

Professor Wangi töörühm avaldas 2005. aastal oma artikli räsifunktsioonide ründamisest, kus MD5 kollisioon oli ilmutatud kujul välja arvutatud ning samuti oli selge, et sarnase konstruktsiooni tõttu ei paku ka SHA-1 enam maksimaalset algses tehnilises lahendustes kavandatud turvataset [268]. Kuna mõlemad räsifunktsioonid on infosüsteemides väga laialdaselt kasutusel, sattus nende süsteemide turvalisus automaatselt küsimärgi alla. Soovides maandada räsifunktsioonide nõrkustest tulenevaid riske Eesti ühes olulisemas e-riigi infrastruktuuri komponendis X-tee, planeeriti X-tee 5. versiooni arendustööde hulka nõrkade räsifunktsioonide väljavahetamine. Üksikasjaliku ülevaate planeeritud tööde sisust ning nende täitmisele kulunud ressurssidest leiab käesoleva aruande 2011. aasta versioonist [40].

Räsifunkstioon SHA-1 jäi endiselt kasutusse SSL/TLS-sessiooni sõnumiautentimiskoodide loomisel. Kuigi protokollistik TLS toetab tugevamate räsifunktsioonidega šifrikomplekte juba 2008. aastast standardi versioonis 1.2 [116], jõudis OpenSSL need teostada alles 2012. Kuna SHA-1 kasutamise vastu sõnumiautentimiskoodide moodustamisel pole tänaseni teada ühtegi praktilist rünnet, otsustati X-tee räsifunktsioonide vahetamise projekti käigus 2010. aastal see komponent muutmata jätta.

4.3.7 Soovitused

Siin uuritud tarkvarateegid tuleks ümber kirjutada viisil, mis lubaks kasutatavaid krüptoalgoritme ning nende võtmepikkusi kiiresti vahetada, kasutades selleks täitmis- või kompileerimisaegseid konfiguratsioonifaile. Sel viisil peaks spetsifitseerima nii neid algoritme, mida teek uute krüptograafiliste objektide loomisel kasutab, kui ka neid algoritme, mida ta kontrollimisel aktsepteerib. Sel viisil on võimalik operatiivselt lisada uute algoritmide tuge ja samas eemaldada vanu, ebaturvaliseks tunnistatud algoritme. Eriti oluline on selline refaktoreerimine räsifunktsioonide toe juures, sest toetada tuleb kõigepealt SHA-2 pere funktsioone ning mõne aja pärast SHA-3 räsifunktsiooni. Lisaks algoritmidele ja võtmepikkustele soovitame, et ka RSA täidiseid oleks hõlbus valida.

Koos ülalkirjeldatud täienduste tegemisega on mõistlik ja soovitatav ka lähtekoodi refaktoreerimine, sest digitaalallkirjastamise ja verifitseerimise erinevad juhud (digitaalallkirjade, sertifikaatide ja kehtivuskinnituste loomine/kontroll) sisaldavad sarnaseid koodilõike.

Krüpteerimisel tuleb plaanida üleminek RSA PKCS #1 v1.5 täidiselt RSA OAEP täidisele [27].

Peale selle tuleb läbi mõelda, kust on võimalik saada entroopiat Windowsis transpordivõtmeid genereerides. OpenSSL-i meetod `RAND_screen` ei pruugi anda piisavalt head tulemust. Eelistatum vahend on Windowsi CryptoAPI.

Transpordivõtmete genereerimisel pole mõtet kutsuda välja funktsiooni `EVP_BytesToKey`. OpenSSL-i juhuarvugeneraatorist saadud väärtused on otse võtmena kasutatavad, samuti ei pea sel juhul toetuma ebaturvalisele räsifunktsioonile MD5.

4.4 Andmevormingud

4.4.1 Digitaalallkirjastatud andmete vormingud

Eestis on kasutusel andmevormingud BDOC [4], BDOC 2.0 [3] ja DigiDoc [237]. Sügisel 2013 lasti välja ka andmevorming BDOC 2.1 [2].

4.4.1.1 Andmevorming DigiDoc

DigiDoc [237] on andmevorming, mida kasutavad DigiDoc-süsteemi kuuluvad rakendused ning mis põhineb rahvusvahelistel standarditel XML-DSIG [125] ja ETSI TS 101903 [23]. See andmevorming on olnud kasutusel juba aastast 2002 ning on praeguseks jõudnud versioonini 1.3. Aastal 2013 on DigiDoc jätkuvalt esmatähtis andmevorming allkirjastatud andmete esitamiseks. DigiDoc-vorminguga on võimalik esitada digitaalallkirjastatud andmeid ning nende andmetega seotud allkirju. Koos allkirjadega esitatakse ka nende aluseks olnud X.509 sertifikaatide kehtivust kinnitavad OCSP-kehtivuskinnitused. Praegune andmevorming määrab järgmised tingimused.

- Andmefailide räsimiseks tuleb kasutada algoritmi SHA-1.
- Räside allkirjastamisel tuleb kasutada RSA PKCS #1 versioonile 1.5 [24] vastavat RSASSA-PKCS1-v1.5 täidist.
- Allkirjastaja sertifikaadi kehtivuskinnituse saamiseks saadetakse OCSP-päringus nonssi väärtusena RSA allkirjast võetud räsiväärtus. Räsimiseks kasutatakse algoritmi SHA-1.

- Elemendis `CertDigest` olev räsi viitab andmefaili või näiteks OCSP-kehtivuskinnituse allkirjastamisel kasutatud sertifikaadile ning seal kasutatakse samuti algoritmi SHA-1.

Andmevormingus ei ole ette nähtud valikuvõimalusi ning kasutusel olevat algoritmi RSA-SHA-1 ei saa rakendused praegu muuta. Standard XMLDSIG näeb küll ette ka dgitaalallkirjaalgoritmi DSA-SHA-1 [13] kasutamisvõimalust, kuid DigiDoc-vormingusse ei ole seda võimalust üle toodud.

4.4.1.2 Vormingu DigiDoc muutmise keerukus

Andmevormingu DigiDoc praegu spetsifitseeritud krüptoalgoritmide muutmiseks tuleb teha järgmist.

- Uued krüptoprimitiivid, näiteks SHA-3 ja seda kasutavad RSA PKCS #1 või DSA skeemid peavad olema teostatud mõnes üldlevinud krüptoteegis, näiteks OpenSSL või Bouncy Castle'i teekides.
- OCSP kehtivuskinnitusteenus peab olema uuendatud, et päringutes saaks kasutada uusi primitiive.
- Tuleb luua ning publitseerida uus DigiDoc-andmevormingu kirjeldus, kasutades uute primitiivide kokkulepitud URI-identifikaatoreid (sarnaselt RFC4051 standardis spetsifitseeritud SHA-512 jms algoritmide identifitkaatoritele).
- Tuleb luua ning testida rakendustarkvara, mis kasutaks uut andmevormingu versiooni. Uuendada tuleb vähemalt DigiDoc-klient, jDigiDoc-teek ning DigiDoc-veebiteenus.

4.4.1.3 Andmevorming BDOC

BDOC on andmevorming, mis on loodud 2008. aastal foorumi `wpki.eu` töö käigus selleks, et asendada Eesti-spetsiifiline andmevorming DigiDoc ning Läti-spetsiifiline eDoc ühise vorminguga, mis võimaldaks esitada allkirjastatud andmeid ning nende andmetega seotud allkirju. BDOC-andmevormingu kirjeldus on standarditud 2009. aastal Eesti standardina [4].

BDOC-vormingus tuleb kasutada samu krüptoalgoritme, mis DigiDoc-vorminguski, räsialgoritmi SHA-1 ning RSASSA-PKCS1-v1.5 täidist. Selles standardis on märkus, mis tungivalt soovitab kasutada uuemaid räsialgoritme. Kuna standard kahjuks ei ütle, millist konkreetset algoritmi tuleks tegelikult kasutada, ning milliseid algoritme peavad kindlasti toetama allkirjastamis- ja valideerimisrakendused, tuleb nentida, et BDOC andmevorming ei paku krüptograafilises mõttes paremaid garantiisid kui DigiDoc-vorming ning praktikas tuleb uute krüptoprimitiivide kasutuselevõtmisel arvestada samalaadsete toimingutega (vt jaotist 4.4.1.2), nagu ka DigiDoc-vormingu korral.

4.4.1.4 Andmevormingud BDOC 2.0 ja BDOC 2.1

Andmevorming BDOC 2.0 oli loomulik jätk algsele vormingule BDOC. Selle eesmärk oli parandada kooskõla alusstandardites toimunud arengutega ning täpsustada mõningaid BDOC andmevormingu seni liiga üldiselt defineeritud aspekte - eelkõige protokolli OCSP spetsiifilist kasutust ajamärgendamisel. Sel eesmärgil võeti kasutusele XML-element xades:SignaturePolicyIdentifier, mis viitas elemendile NonceAlgo-

rithm (xmlns:nonce="http://www.sk.ee/repository/NonceAlgorithm") ja määratles sellega protokollis OCSP nonsi arvutamisel kasutatud räsifunktsiooni.

BDOC 2.0 standardikavand sai kriitika osaliseks [48], sest plaanitud muudatused ei võimaldanud üheselt eristada ajamärgendatud dokumente ajatembeldatud dokumentidest, samuti olid kriitikud eriarvamusel protokolli OCSP taseme küsimuste lahendamise suhtes XAdES vormingu tasemel. XML elemendi xades:SignaturePolicyIdentifier semantika on alusstandardis defineeritud hägusalt ning selle kohustusliku elemendina kasutuselevõtmises nähti võimalikku probleemi rahvusvahelise ühilduvuse saavutamisel.

BDOC 2.1 standard [2] võtab kriitikat arvesse, muutes kavandit 2.0 selliselt, et spetsiifiline allkirjastamispoliitika on kasutusel vaid OCSP-põhist ajamärgendamist kasutavates dokumentides. Samuti on parandatud nonsi arvutamiseks kasutatava räsialgoritmi esitamist, mis toimub nüüd juba OCSP vastuses ASN.1 vormingus. Peale selle täpsustab standardikavand arhiiviajatemplite kasutamist. BDOC 2.1 puhul tuleb andmefailide räsimiseks kasutada mõnda perekonna SHA-2 algoritmi.

Andmevormingu BDOC tulevik on hetkel lahtine. Sügisel 2013 valmis BDOC 2.1 starnardi lõppversioon [2] ja Riigi Infosüsteemi Amet avaldas ID-kaardi baastarkvara uue versiooni, mis täiendas BDOC-vormingu tuge. Tarkvara poolt vaikimisi pakutavaks vorminguks on aga endiselt DDOC [10].

Käesoleva aasta suvel erinevate allkirjavormingute baasteegi teostusest leitud vead [126] sundisid baastarkvara arendajat vormingu BDOC toe tarkvarast ajutiselt välja lülitama, praeguseks on kõik vead parandatud. Hetkel on veel aktiivne ka BDOCi vanimat versiooni väljastav DigiDocService.

4.4.2 Krüpteeritud andmete vormingud

Andmevorming CDOC ei ole spetsifitseeritud omaette standardina, vaid kasutatakse tavalist XMLENC [144] standardit. XMLENC on üsna lai ning lubab kasutada mitmeid krüptoalgoritme (kohustuslikud algoritmid on toodud paksus kirjas):

- **3DES-CBC**, **AES-128-CBC**, AES-192-CBC, **AES-256-CBC**,
- **RSAES-PKCS1-v1_5**, **RSA-OAEP**,
- **SHA-1**, **SHA-256**, SHA-512, RIPEMD-160.

Praktikas kasutab DigiDoc-i klienditarkvara [239] vaid alamosa nendest algoritmidest ega teosta kõiki kohustuslikke algoritme. Lähtekoodi on sisse kirjutatud algoritmide AES-128-CBC, RSAES-PKCS1-v1_5 ning SHA-1 kasutamine ning kasutajal ei ole võimalik neid muuta.

4.4.3 Suurte andmehulkade krüpteerimine. TrueCrypt

Suuri andmehulki pole enamasti mõistlik krüpteerida failide või andmebaasikirjete kaupa. Tüüplahenduseks on sel juhul terve failisüsteemi krüpteerimine, kusjuures failisüsteem võib asuda eraldi virtuaalsel kettal (mis moodustab välise süsteemi poolt vaadates ühe suure faili) või hõlmata kogu füüsilise ketta. Ketta krüpteerimiseks on olemas palju äri- ja priivaralahendusi (näiteks Microsoft Windowsi koosseisu kuuluv BitLocker ja Mac OSXi koosseisu kuuluv FileVault 2), millest ammendava ülevaate andmine jääb selle aruande raamidest välja. Seetõttu keskendume näiterakendusena ühele konkreetsele programmile TrueCrypt [21], mis on hea kompromiss avatuse, kasutusmugavuse ja võimalusterohkuse vahel ning on seetõttu Eestis võrdlemisi laia rakendust leidnud.

Failisüsteemi krüpteerimiseks kasutab TrueCrypt algoritme AES-256, Serpent [87] ja TwoFish [103] ning nende kombinatsioone XTS-režiimis. Paroolist krüptovõtme tuletamiseks saab kasutada räsifunktsioone RIPEMD-160, SHA-512 ja Whirlpool [266]. Kõiki neid algoritme võib käesoleva aruande ajahorisondi ulatuses pidada turvaliseks.

TrueCryptis moodustataval konteinerfailil pole eristatavat päist ning ainus võimalus oletada, et see on TrueCrypti konteiner, on tuvastada, et faili pikkus on 512 biti kordne ning et fail läbib statistilised juhuslikkustestid. Võimalikud on ka ründed, kus kasutatav parool/võti tuvastatakse mälu skaneerides või klaviatuuri pealt kuulates. Sellised ründed on aga ohtlikud praktiliselt kõigi kettakrüptolahenduste puhul, seega ei saa neid võtta argumentidena TrueCrypti kasutamise vastu.

TrueCrypti lähtekood on vabalt saadaval ning seega põhimõtteliselt täielikult kontrollitav. Ükski sõltumatu osapool pole aga teadaolevalt seda kontrolli läbi viinud. Kuna TrueCrypti arendajad eelistavad jääda anonüümseiks ning nende kasutatavat litsentsi pole OSI heaks kiitnud, ei saa TrueCrypti enne täielikku koodiauditit soovitada kõrget konfidentsiaalsustaset nõudvate dokumentide krüpteerimiseks.

Kui tekib vajadus muuta krüpteerimiseks kasutatavat plokkšifrit, tuleb luua uus konteiner ja andmed ümber tõsta. Parooli või võtme tuletamiseks kasutatava räsifunktsiooni vahetamine on võimalik ka ilma uut konteinerit loomata; seda funktsiooni pakub ka TrueCrypti kasutajaliides.

4.5 ISKE krüptomooduli täiendamine

ISKE on riigi infosüsteeme toetav infoturbesüsteem [35]. Krüptograafilisi meetodeid on käsitletud tema mitmes moodulis, kuid põhjalikuma tähelepanuta on jäänud krüptoalgoritmide vahetamise vajadusest tulenevad tegevused. Soovitame selles valguses üle vaadata ning täiendada järgmised ISKE moodulid:

- "B 1.7 Krüptokontseptsioon", sh jaotis 7 "Valmisolek hädaolukorraks",
- "B 1.8 Turvaintsidentide käsitlus",
- "B 1.9 Riist- ja tarkvara haldus",
- "B 1.13 Infoturbe teadlikkus",
- "B 1.14 Turvapaikade ja muudatuste haldus",
- "B 1.16 Nõuete haldus".

Tuleb välja töötada ka tarkvaraarenduse suunised, mis käsitleksid krüptoalgoritmide kiiret vahetatavust.

5 Kokkuvõte

5.1 Üldised tähelepanekud ja soovitused

Uuringu üks eesmärk oli saada ülevaade Eestis kasutatavate krüptograafilistest algoritmidest ja protokollidest ning meetmetest, mis peaksid tagama krüptograafiliste vahendite ajakohasuse. Eesmärgi saavutamiseks valitud personaalsete intervjuude strateegia andis paremaid tulemusi kui 2011. aastal sama uuringu käigus laiali saadetud ringkiri. Intervjuud nõudsid samas tunduvalt rohkem aega. Seetõttu ei saa ka praeguse aruande tulemusi lugeda täielikeks. Ühekordsete suhteliselt väikesemahuliste projektide abil polegi kogu Eestit hõlmava uuringu koostamine võimalik. Uuringu käigus on loodud esialgne metoodika ja näidisküsimustikud, millele tugunedes saab edaspidiseid intervjuusid läbi viia. Kui riigi huvi on selle töö jätkamine, soovitame luua vastav püsiv ametikoht või institutsioon, millele on tagatud ligipääs vajalikele potentsiaalselt konfidentsiaalsetele lähteandmetele.

2011. aasta aruandes soovitati mitmete nõrkade krüptoalgoritmide (nt RSA1024 ja SHA-1) väljavahetamist. Uuringu käigus selgus, et neid algoritme kasutatakse aktiivselt veel 2013. aasta lõpuski. Kuigi põhjuseid on erinevaid, kumab kõigist nõrga krüptograafia kasutusjuhtudest välja üks aspekt: Eestis puuduvad regulatiivaktid, mis sätestaksid krüptograafiliste meetodite miinimumnõuded. Seetõttu pole nõrkade algoritmide kasutajatele formaalselt midagi ette heita ega millelegi tuginedes olukorra parandamist nõuda. Niisiis soovitame krüptograafia miinimumnõuete sisseviimist riiklikul tasandil.

Aruande koostamise ajal ilmnes ka üks uus riskiklass, mida (vähemalt avalikus kogukonnas) Eestis seni väga aktiivselt käsitletud ei ole, nimelt välismaise superluureorganisatsiooni tegevuse poolt põhjustatud oht meie riigi ja inimeste privaatsusele. Uuringuaruande kirjutamise ajal (sügisel 2013) pole seda ohtu ega tema ulatust suudetud täielikult hinnata, seda ka mitte teistes riikides. Kindlasti on Eestil üksinda väga suure eelarvega luureorganisatsiooni(de) vastu võitlemisel vähe edulootust, kuid riskide paremaks mõistmiseks tuleb analüüsi kindlasti jätkata. Soovitame sellesse tegevusse kaasata teisi asjakohaseid institutsioone (nt Kaitsepolitsei ja Teabeamet) ning teha laiaulatuslikku koostööd rahvusvahelisel poliitilisel tasandil.

5.2 Plokkšifrid ja krüpteerimisrakendused

Algoritmi DES ei ole soovitatav üheski rakenduses kasutada ja algoritmi TDEA (3DES) tuleks, kui võimalik, kõigis rakendustes vältida.

Blowfish sobib kasutamiseks praktiliselt kõikjal turvalise plokkšifrina. AES võtmepikkusega 128, 192 või 256 bitti on aruande ajahorisondi ulatuses piisavalt turvaline. Andmete krüpteerimise rakendust TrueCrypt ei ole enne täielikku koodiauditit soovitav kasutada kõrget konfidentsiaalsustaset nõudvate dokumentide krüpteerimiseks.

5.3 Asümmeetriline krüpteerimine ja digitaalallkirjad

Algoritmi RSA 512- ja 768-bitised versioonid, samuti diskreetse logaritmi põhised süsteemid mooduli pikkusega kuni 768 bitti on ebaturvalised ja neid ei ole soovitatav enam kasutada. Algoritmi RSA 1024-bitiste versiooni kasutamisest tuleb loobuda nii kiiresti kui võimalik, kindlasti lähima kahe aasta jooksul. Algoritmi RSA 2048-bitist versiooni võib pidada turvaliseks käesoleva aruande ajahorisondi (st 5 aasta) piires.

Kui ei loeta riskiks võimalikke USA luureorganisatsioonide käsutuses olevaid tagauksi, on mõistlik kasutada võimalikult standardseid elliptilisi kõveraid, nt `P-192` ja `P-256`, mis kuuluvad olulisemate standardite ühisosse. Kui aga tagauste olemasolu peetakse oluliseks riskiks, on mõistlikum kasutada Dan Bernsteini kõverat `Curve25519` või mõnd BSI standarditud kõverat.

Krüpteerimisel on soovitav RSA PKCS #1 v1.5 täidiselt üle minna RSA OAEP täidisele.

5.4 Räsifunktsioonid ja sõnumiautentimiskoodid

Räsifunktsioonide MD5 ja RIPEMD-128 kasutamisest tuleks võimalikult kiiresti ja täielikult loobuda.

Räsifunktsiooni SHA-1 kasutamisest krüptograafiliselt kriitilistes rakendustes tuleks loobuda hiljemalt aastaks 2015. Räsifunktsioonide pere SHA-2 algoritme ja algoritmi RIPEMD-160 võib käesoleva aruande ajahorisondi ulatuses pidada turvalisteks. Räsifunktsiooni SHA-3 on soovitav mitte kasutada enne ta täielikku standardimist ning rahvusvahelise kogukonna heakskiitu. Seni on soovitatav kasutada SHA-2 perekonna räsifunktsioone.

Sõnumiautentimiskoodis HMAC tuleb kasutada turvalisi räsifunktsioone, nt SHA-2. Koodi CBC-MAC asemel on soovitav kasutada mõnd ta modifikatsiooni, näiteks CMAC või OMAC.

5.5 Mobiilside ja WiFi

Jadašifreid RC4, A5/1 ja A5/2 ei ole soovitav kasutada ühegi turvaeesmärgi taotlemiseks. Plokkšifri Kasumi kasutamisest tuleks loobuda hiljemalt 5 aasta jooksul. SNOW 3G on üldotstarbelise mobiilside krüpteerimiseks piisavalt turvaline. Väga tundlikku teavet on õigem mobiilvõrkude kaudu üldse mitte edastada.

Turvalisust nõudvates keskkondades tuleb hoiduda WiFi-võrkude turvamisest protokollidega WEP ja WPS ning kasutada nende asemel protokolli WPA2 (vt ka jaotist 3.5.3). Samuti tuleb sellistes keskkondades hoiduda võtmevahetusprotokolli PEAP kasutamisest, mille asemel võib kasutada näiteks protokolli EAP-TLS.

5.6 Autentimine ja digitaalallkirjad

Kerberos-protokollis kasutatav lihtsustatud profiil CBC-töörežiimis plokkšifri kasutamiseks koos võtmelaiendusega konstrueerib turvalisest plokkšifrist turvalise autenditud krüptosüsteemi. Seega võib Kerberoses selle profiili kasutamist lugeda turvaliseks, erinevalt Kerberose üldisest krüpteerimis- ja kontrollsummaprofiilist.

Järgmise põlvkonna Mobiil-ID protokollid on soovitatav kavandada, alustades soovitud turvaomaduste formaalsest kirjeldamisest ja jätkates protokollisõnumite fikseerimise ning formaalse tõestamisega, et protokoll neid turvaomadusi ka tõepoolest rahuldab.

Protokollide iPizza ja TUPAS kasutajatel soovitame võimalikult kiiresti paigata protokollides ja realisatsioonides olevad vead ajutiste lahendustega. Pikemas perspektiivis soovitame pangalingi protokollide kasutamisest loobuda ning autentida kasutajaid ID-kaardi ja Mobiil-ID lahendustega.

Eesti ID-kaartide haldusalgoritmina kasutatav 3DES on soovitatav võimalikult kiiresti välja vahetada turvalisema algoritmi, näiteks AES vastu.

Brauseripluginas esteid, mis lubab dokumente brauseri kaudu allkirjastada, on soovitav lisada allkirjastamismeetodile täiendava parameetrina räsifunktsiooni identifikaator.

Digitaalallkirjaseadme loomine on teenus, mille osutamise tingimused on standarditud 2003. aastal [30]. Nii ID-kaardi kui teiste digitaalset allkirjastamist ja autentimist võimaldavate seadmete loomise vastavus sellele standardile tuleb üle vaadata ning vajadusel seda vastavust parandada.

5.7 Veebiserverid ja protokoll SSL/TLS

SSL/TLS protokollis mitte kasutada plokkšifrit IV-aheldatud CBC-režiimis. TLS alates versioonist v1.1 seda režiimi enam ei toeta, seega soovitame mitte kasutada protokolli TLS vanemaid versioone. Samuti tuletame meelde, et kuigi BEAST-rünne ei ole rakendatav jadašifritele, on protokollis TLS vaikimisi jadašifriks RC4, mida me samuti kasutada ei soovita (vt. jaotis 2.1.3).

Põhjalikult on veebiserverite konfigureerimise tehnilisi detaile kirjeldatud punktis 4.1.1.2.

5.8 X-tee

Protokolli SKIP vähese kasutatavuse tõttu pole uuringu autoreil õnnestunud leida selle protokolli turvaanalüüse. Seetõttu soovitame kas vahetada X-tees protokoll SKIP mõne levinuma võrgukihi-turvaprotokolli vastu või viia läbi protokolli SKIP (X-tees kasutatavate valikute mahus) formaalne analüüs.

5.9 Krüptoteekide ja protokollide kasutamine

Krüptograafilistest standarditest tuleb alati kasutada uusimaid versioone. Krüptograafiliste algoritmide rakendamisel on soovitav tugineda laialt tunnustatud ja testitud teostustele, nagu on teegid OpenSSL või Bouncy Castle.

Tarkvarateegid tuleks ümber kirjutada viisil, mis lubaks kasutatavaid krüptoalgoritme ning nende võtmepikkusi kiiresti vahetada, kasutades selleks täitmis- või kompileerimisaegseid konfiguratsioonifaile. Seejuures peab rakendus konfiguratsioonifailide kasutamisel veenduma nende autentsuses, et välistada ründed pahatahtlike konfiguratsioonimuudatuste kaudu.

Soovitav on kasutada tõestatavalt turvalisi protokolle, sest protokollide turvatõestusi osatakse esitada ning neist lähtuvate garantiide ulatusest saadakse aru. Kui infosüsteemis kasutatavatel protokollidel on turvaomadused, mida need infosüsteemid vajavad, siis ei ole tarvis neid protokolle tulevikus välja vahetada. Seega puudub vajadus infosüsteeme nii ulatuslikult modulariseerida, et näiteks terve võtmevahetusprotokoll koos oma loogikaga

oleks lihtsasti vahetatav. Küll aga on tarvis, et protokollis kasutatavad krüptograafilised primitiivid ning nende võtmepikkused oleksid vahetatavad, mistõttu protokollide teostused peavad olema modulaarsed ning kasutatavad primitiivid ei tohi olla lähtekoodis püsistatud (*hard-coded*), vaid peavad jääma koodist sõltumatult konfigureeritavaks.

Eesti digitaalallkirjastamise standardite ja tarkvara tuge tuleks muuta modulaarsemaks. Toetatud räsi- ja allkirjastamisalgoritmide ning võtmepikkuste loetelu muutmine tuleb teha lihtsamaks nii DigiDoc-standardis kui ka DigiDoc-teegis ja ilmselt siis ka veebiteenuses DigiDocService. Alternatiiv on loobuda nii DigiDoc-standardist kui ka -teegist võimalikult ruttu ning kasutada edaspidi ainult BDOC-i, kus kasutatavate algoritmide loetelu ei ole kinnine.

5.10 Turbehaldus

Täiendada ISKE metoodikat nii, et saaks paremini võtta arvesse krüptograafiliste algoritmide vananemise ja vahetamisega seotud ohte.

Kirjandus

[1] ANSI X9.62:2005. Public Key Cryptography for the Financial Services Industry, The Elliptic Curve Digital Signature Algorithm (ECDSA).

[2] BDOC - Digitaalallkirja vorming. http://www.id.ee/public/bdoc-spec21-est.pdf.

[3] BDOC - Digitaalallkirja vorming. Standardikavand BDOC2.0:2013.

[4] BDOC. Digitaalallkirja vorming. EVS 821:2009, http://www.evs.ee/product/tabid/59/p-165934-evs-8212009.aspx.

[5] Bouncy Castle cryptographic toolkit. http://www.bouncycastle.org/.

[6] Bouncy Castle Specifications. http://bouncycastle.org/specifications.html.

[7] Computer Data Authentication. FIPS PUB 113.

[8] Cyptography Research and Evaluation Committees. http://www.cryptrec.go.jp/english/index.html.

[9] Data Encryption Standard (DES). FIPS PUB 46-3. http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf.

[10] Digiallkirju saab anda rahvusvahelistele standarditele vastavas vormingus BDOC. https://www.ria.ee/digiallkirju-saab-anda-rahvusvahelistele-standarditele-vastavas-vormingus-bdoc/.

[11] Digitaalne tempel. http://www.sk.ee/teenused/digitempli-teenus/.

[12] Digitaalse isikutunnistuse vormi, tehnilise kirjelduse ja digitaalsele isikutunnistusele kantavate andmete loetelu kehtestamine. https://www.riigiteataja.ee/akt/13353744.

[13] Digital Signature Standard (DSS). FIPS PUB 186-4. http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf.

[14] IEEE 802.11: Wireless Local Area Networks (LANs). http://standards.ieee.org/about/get/802/802.11.html.

[15] NSA Suite B Cryptography. http://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml.

[16] OpenSSL SSL/TLS toolkit. http://www.openssl.org.

[17] Recommendation for Block Cipher Modes of Operation. NIST Special Publication 800-38A, http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf.

[18] SEC 2: Recommended Elliptic Curve Domain Parameters. http://www.secg.org/download/aid-784/sec2-v2.pdf.

[19] Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 2: KASUMI Specification. http://www.3gpp.org/ftp/specs/html-info/35202.htm.

[20] Specification of the 3GPP Confidentiality and Integrity Algorithms UEA2 & UIA2. Document 1: UEA2 and UIA2 Specification. http://www.quintillion.co.jp/3GPP/Specs/etsi_sage_doc1_v1_1.pdf.

[21] TrueCrypt website. http://www.truecrypt.org/.

[22] Wi-Fi Protected Setup™. Wi-Fi Alliance. http://www.wi-fi.org/knowledge-center/articles/wi-fi-protected-setup%E2%84%A2.

[23] XML Advanced Electronic Signatures (XAdES). ETSI TS 101903, http://uri.etsi.org/01903/v1.4.1/.

[24] PKCS #1: RSA Encryption Standard, versioon 1.5, 1. november 1993. http://www.rsa.com/rsalabs/node.asp?id=2125.

[25] PKCS #12 - Personal Information Exchange Syntax Standard, versioon 1.0, 24. juuni 1999. http://www.rsa.com/rsalabs/node.asp?id=2138.

[26] Security Requirements for Cryptorgaphic Modules, 2001. http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf.

[27] PKCS #1: RSA Encryption Standard, versioon 2.1, 14. juuni 2002. http://www.rsa.com/rsalabs/node.asp?id=2125.

[28] CCSDS Next Generation Space Internet (NGSI) — End-to-End Security for Space Mission Communications. NASA Consultative Committee for Space Data Systems, aprill 2003. http://public.ccsds.org/publications/archive/733x5o1.pdf.

[29] CRYPTREC Report 2002. Technical report, Information-technology Promotion Agency, Japan, 2003. http://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/c02e_report2.pdf.

[30] Security Requirements for Trustworthy Systems Managing Certificates for Electronic Signatures - Part 1: System Security Requirements. ftp://ftp.cen.eu/CEN/Sectors/TCandWorkshops/Workshops/eSIGN_CWAs/cwa14167-01-2003-Jun.pdf, June 2003.

[31] PKCS #11 - Cryptographic Token Interface Standard, versioon 2.20, 28. juuni 2004. http://www.rsa.com/rsalabs/node.asp?id=2133.

[32] Specification of the Problems in the High-Level Specification Language. Deliverable 6.2 of AVISPA (Automated Validation of Internet Security Protocols and Applications) project, 27. juuli 2005. http://www.avispa-project.org.

[33] PKCS #5: Password-Based Cryptography Standard, versioon 2.1, 5. oktoober 2006. http://www.rsa.com/rsalabs/node.asp?id=2127.

[34] IEEE Standard for Information Technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 2007. http://standards.ieee.org/getieee802/download/802.11-2007.pdf.

[35] Infosüsteemide kolmeastmelise etalonturbe süsteem ISKE. Kataloogid. Versioon 6.00, 2010. https://www.ria.ee/public/ISKE/iske_kataloogid_6_00.pdf.

[36] Mathematical routines for the NIST prime elliptic curves, 2010. http://www.nsa.gov/ia/_files/nist-routines.pdf.

[37] PC/SC Workgroup Specifications, versioon 2.01.9, Aprill 2010. http://www.pcscworkgroup.com/specifications/overview.php.

[38] Suite B Implementer's Guide to FIPS 186-3 (ECDSA), 2010. http://www.nsa.gov/ia/_files/ecdsa.pdf.

[39] Digiallkirjastamine veebis. http://www.id.ee/10824, 20. aprill 2011.

[40] Krüptograafiliste algoritmide kasutusvaldkondade ja elutsükli uuring. http://www.riso.ee/sites/default/files/elfinder/article_files/kryptoalgoritmide_elutsykli_uuring.pdf, 2011. AS Cybernetica aruanne nr A-60-1.

[41] OpenSC - tools and libraries for smart cards, 2011. http://www.opensc-project.org/.

[42] Elliptic Curve Cryptography. https://www.bsi.bund.de/cae/servlet/contentblob/471398/publicationFile/30615/BSI-TR-03111_pdf.pdf, 2012.

[43] NIST Selects Winner of Secure Hash Algorithm (SHA-3) Competition. http://www.nist.gov/itl/csd/sha-100212.cfm, 2012.

[44] Secure Hash Standard (SHS) . FIPS PUB 180-4, 2012. http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf.

[45] *Apache Module mod_ ssl, Apache HTTP Server Versioon 2.4*, 2013. http://httpd.apache.org/docs/2.4/mod/mod_ssl.html.

[46] Are the NIST Standard Elliptic Curves Back-doored?, 2013. http://it.slashdot.org/story/13/09/11/1224252/are-the-nist-standard-elliptic-curves-back-doored.

[47] DigiDoc teegid - üldinfo ja teekide reliisimise ajakava, 6. märts 2013. http://www.id.ee/index.php?id=30290.

[48] Digitaalallkirjade jätkusuutlikkuse analüüs. Cybernetica, Riigi Infosüsteemide Ameti tellimusel, 19. aprill 2013. http://www.id.ee/public/Digitaalallkirjade_jatkusuutlikkuse_analyys1.0.pdf.

[49] ID-kaardi baastarkvara. http://id.eesti.ee/, oktoober 2013.

[50] Martín Abadi. Secrecy by Typing in Security Protocols. In Martín Abadi and Takayasu Ito, editors, *TACS*, volume 1281 of *Lecture Notes in Computer Science*, pages 611–638. Springer, 1997.

[51] Martín Abadi, Bruno Blanchet, and Cédric Fournet. Just fast keying in the pi calculus. *ACM Transactions on Information and System Security*, 10(3, Article 9):1–59, 2007.

[52] Martín Abadi and Roger M. Needham. Prudent Engineering Practice for Cryptographic Protocols. *IEEE Transactions on Software Engineering*, 22(1):6–15, 1996.

[53] Martín Abadi and Phillip Rogaway. Reconciling Two Views of Cryptography (The Computational Soundness of Formal Encryption). *Journal of Cryptology*, 15(2):103–127, 2002.

[54] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz. Extensible Authentication Protocol (EAP). IETF RFC3748, `http://tools.ietf.org/html/rfc3748`.

[55] William Aiello, John Ioannidis, and Patrick Drew McDaniel. Origin authentication in interdomain routing. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *ACM Conference on Computer and Communications Security*, pages 165–178. ACM, 2003.

[56] Nadhem J. AlFardan and Kenneth G. Paterson. Lucky Thirteen: Breaking the TLS and DTLS Record Protocols. In *IEEE Symposium on Security and Privacy*, pages 526–540. IEEE Computer Society, 2013.

[57] Kazumaro Aoki, Jian Guo, Krystian Matusiewicz, Yu Sasaki, and Lei Wang. Preimages for Step-Reduced SHA-2. In *Advances in Cryptology – ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 578–597, 2009.

[58] Alessandro Armando, David A. Basin, Yohan Boichut, Yannick Chevalier, Luca Compagna, Jorge Cuéllar, Paul Hankes Drielsma, Pierre-Cyrille Héam, Olga Kouchnarenko, Jacopo Mantovani, Sebastian Mödersheim, David von Oheimb, Michaël Rusinowitch, Judson Santiago, Mathieu Turuani, Luca Viganò, and Laurent Vigneron. The AVISPA Tool for the Automated Validation of Internet Security Protocols and Applications. In Kousha Etessami and Sriram K. Rajamani, editors, *CAV*, volume 3576 of *Lecture Notes in Computer Science*, pages 281–285. Springer, 2005.

[59] Trüb Baltic AS. EstEID. ID-card specification, 2012. `http://id.ee/public/TB-SPEC-EstEID-Chip-App-v3.4.pdf`.

[60] N. Asokan, Valtteri Niemi, and Kaisa Nyberg. Man-in-the-Middle in Tunnelled Authentication Protocols. In Bruce Christianson, Bruno Crispo, JamesA. Malcolm, and Michael Roe, editors, *Security Protocols*, volume 3364 of *Lecture Notes in Computer Science*, pages 28–41. Springer Berlin Heidelberg, 2005.

[61] Ashar Aziz, Martin Patterson, and Geoff Baehr. Simple Key-Management for Internet Protocol (SKIP). In *Internet Society's 1995 International Networking Conference (INET '95)*, 1995. `http://wayback.archive.org/web/*/http://skip.incog.com/inet-95.ps`.

[62] Steve Babbage, Dario Catalano, Carlos Cid, Benne de Weger, Orr Dunkelman, Christian Gehrmann, Louis Granboulan, Tim Güneysu, Jens Hermans, Tanja Lange, Arjen Lenstra, Chris Mitchell, Mats Näslund, Phong Nguyen, Christof Paar, Kenny Paterson, Jan Pelzl, Thomas Pornin, Bart Preneel, Christian Rechberger, Vincent

Rijmen, Matt Robshaw, Andy Rupp, Martin Schläffer, Nigel Smart, Serge Vaudenay, Fré Vercauteren, and Michael Ward. ECRYPT II Yearly Report on Algorithms and Keysizes (2011-2012). Technical report, European Network of Excellence in Cryptology II, September 2012. http://www.ecrypt.eu.org/documents/D.SPA.20.pdf.

[63] Michael Backes, Dennis Hofheinz, and Dominique Unruh. CoSP: a general framework for computational soundness proofs. In Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis, editors, *ACM Conference on Computer and Communications Security*, pages 66–78. ACM, 2009.

[64] Michael Backes and Dominique Unruh. Computational soundness of symbolic zero-knowledge proofs. *Journal of Computer Security*, 18(6):1077–1155, 2010.

[65] Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, and Emmanuel Thomé. A quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. Cryptology ePrint Archive, Report 2013/400, 2013. http://eprint.iacr.org/.

[66] Gregory V. Bard. The vulnerability of ssl to chosen plaintext attack. Cryptology ePrint Archive, Report 2004/111, 2004. http://eprint.iacr.org/.

[67] Gregory V. Bard. A challenging but feasible blockwise-adaptive chosen-plaintext attack on ssl. In Manu Malek, Eduardo Fernández-Medina, and Javier Hernando, editors, *SECRYPT*, pages 99–109. INSTICC Press, 2006.

[68] Romain Bardou, Riccardo Focardi, Yusuke Kawamoto, Lorenzo Simionato, Graham Steel, and Joe-Kai Tsay. Efficient padding oracle attacks on cryptographic hardware. In Safavi-Naini and Canetti [230], pages 608–625.

[69] Elad Barkan, Eli Biham, and Nathan Keller. Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 600–616. Springer Berlin / Heidelberg, 2003.

[70] Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid. Recommendation for key management – part 1: General (revision 3). Technical report, NIST, July 2012. http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57_part1_rev3_general.pdf.

[71] Elaine Barker, Don Johnson, and Miles Smid. Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography. Technical report, NIST, 2007. NIST Special Publication 800-56A, http://csrc.nist.gov/publications/nistpubs/800-56A/SP800-56A_Revision1_Mar08-2007.pdf.

[72] William C. Barker. Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher. Technical report, NIST, 2008. http://csrc.nist.gov/publications/nistpubs/800-67/SP800-67.pdf.

[73] David A. Basin, Sebastian Mödersheim, and Luca Viganò. An On-the-Fly Model-Checker for Security Protocol Analysis. In Einar Snekkenes and Dieter Gollmann, editors, *ESORICS*, volume 2808 of *Lecture Notes in Computer Science*, pages 253–270. Springer, 2003.

[74] Tal Be'ery and Amichai Shulman. A Perfect CRIME? Only TIME will tell. Technical report, Black Hat Europe, 2013. https://media.blackhat.com/eu-13/briefings/Beery/bh-eu-13-a-perfect-crime-beery-wp.pdf.

[75] Mihir Bellare. New Proofs for NMAC and HMAC: Security Without Collision-Resistance. In Cynthia Dwork, editor, *Advances in Cryptology – CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 602–619. Springer Berlin / Heidelberg, 2006.

[76] Mihir Bellare, Joe Kilian, and Phillip Rogaway. The Security of the Cipher Block Chaining Message Authentication Code. *Journal of Computer and System Sciences*, 61(3):362–399, 2000.

[77] Mihir Bellare and Chanathip Namprempre. Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In Tatsuaki Okamoto, editor, *Advances in Cryptology — ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 531–545. Springer Berlin / Heidelberg, 2000.

[78] Mihir Bellare and Phillip Rogaway. Encode-Then-Encipher Encryption: How to Exploit Nonces or Redundancy in Plaintexts for Efficient Cryptography. In Tatsuaki Okamoto, editor, *Advances in Cryptology — ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 317–330. Springer Berlin / Heidelberg, 2000.

[79] Andreas Bender and Guy Castagnoli. On the Implementation of Elliptic Curve Cryptosystems. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89 Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 186–192. Springer New York, 1990.

[80] Dan Bernstein. Irrelevant patents on elliptic-curve cryptography. http://cr.yp.to/ecdh/patents.html.

[81] Dan Bernstein. US patent 5159632. http://cr.yp.to/patents/us/5159632.html.

[82] Daniel J. Bernstein. Cache-timing attacks on AES, 2005. http://cr.yp.to/antiforgery/cachetiming-20050414.pdf.

[83] Daniel J. Bernstein. Curve25519: New Diffie-Hellman Speed Records. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography - PKC 2006*, volume 3958 of *Lecture Notes in Computer Science*, pages 207–228. Springer, 2006.

[84] Daniel J. Bernstein and Tanja Lange. SafeCurves: choosing safe curves for elliptic-curve cryptography. http://safecurves.cr.yp.to.

[85] F. Bersani and H. Tschofenig. The EAP-PSK Protocol: A Pre-Shared Key Extensible Authentication Protocol (EAP) Method. IETF RFC4764, http://tools.ietf.org/html/rfc4764.

[86] Karthikeyan Bhargavan, Cédric Fournet, Andrew D. Gordon, and Nikhil Swamy. Verified implementations of the information card federated identity-management protocol. In Masayuki Abe and Virgil D. Gligor, editors, *ASIACCS*, pages 123–135. ACM, 2008.

[87] Eli Biham, Ross Anderson, and Lars Knudsen. Serpent: A New Block Cipher Proposal. In Serge Vaudenay, editor, *Fast Software Encryption*, volume 1372 of *Lecture Notes in Computer Science*, pages 222–238. Springer Berlin / Heidelberg, 1998.

[88] Alex Biryukov and Dmitry Khovratovich. Related-Key Cryptanalysis of the Full AES-192 and AES-256. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 1–18. Springer Berlin / Heidelberg, 2009.

[89] Alex Biryukov, Adi Shamir, and David Wagner. Real Time Cryptanalysis of A5/1 on a PC. In Bruce Schneier, editor, *Fast Software Encryption*, volume 1978 of *Lecture Notes in Computer Science*, pages 37–44. Springer Berlin / Heidelberg, 2001.

[90] S. Blake-Wilson, N. Bolyard, V. Gupta, C. Hawk, and B. Moeller. Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS). RFC4492, `http://www.ietf.org/rfc/rfc4492.txt`.

[91] Bruno Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *CSFW*, pages 82–96. IEEE Computer Society, 2001.

[92] Bruno Blanchet. Automatic verification of cryptographic protocols: a logic programming approach. In *PPDP*, pages 1–3. ACM, 2003.

[93] Daniel Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin / Heidelberg, 1998.

[94] Chiara Bodei, Pierpaolo Degano, Flemming Nielson, and Hanne Riis Nielson. Flow logic for Dolev-Yao secrecy in cryptographic processes. *Future Generation Computer Systems*, 18(6):747–756, 2002.

[95] Alexandra Boldyreva and Virendra Kumar. Extended Abstract: Provable-Security Analysis of Authenticated Encryption in Kerberos. In *IEEE Symposium on Security and Privacy*, pages 92–100. IEEE Computer Society, 2007.

[96] Dan Boneh and Glenn Durfee. Cryptanalysis of RSA with private key d less than $N^{0.292}$. *IEEE Transactions on Information Theory*, 46(4):1339–1349, 2000.

[97] Matteo Bortolozzo, Matteo Centenaro, Riccardo Focardi, and Graham Steel. Attacking and fixing PKCS#11 security tokens. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM Conference on Computer and Communications Security*, pages 260–269. ACM, 2010.

[98] Daniel R. L. Brown. Generic Groups, Collision Resistance, and ECDSA. *Designs, Codes and Cryptography*, 35:119–152, 2005.

[99] Billy Bob Brumley, Risto M Hakala, Kaisa Nyberg, and Sampo Sovio. Consecutive S-box lookups: a timing attack on SNOW 3G. In *Information and Communications Security*, pages 171–185. Springer, 2010.

[100] Michael Burrows, Martín Abadi, and Roger M. Needham. A Logic of Authentication. In *SOSP*, pages 1–13. ACM, 1989.

[101] Frederick Butler, Iliano Cervesato, Aaron D. Jaggard, Andre Scedrov, and Christopher Walstad. Formal analysis of Kerberos 5. *Theoretical Computer Science*, 367(1-2):57–87, 2006.

[102] Ran Canetti and Hugo Krawczyk. Security Analysis of IKE's Signature-Based Key-Exchange Protocol. In Moti Yung, editor, *Advances in Cryptology — CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 27–52. Springer Berlin / Heidelberg, 2002.

[103] Christophe Cannière. Twofish. In Henk van Tilborg, editor, *Encyclopedia of Cryptography and Security*, pages 638–639. Springer US, 2005.

[104] Greg Childers. Factorization of a 1061-bit number by the Special Number Field Sieve. Cryptology ePrint Archive, Report 2012/444, 2012. `http://eprint.iacr.org/`.

[105] Carlos Cid, Henri Gilbert, Daniel Augot, Alex Biryukov, Anne Canteaut, Nicolas Courtois, Christophe De Cannière, Cédric Lauradoux, Matthew Parker, Bart Preneel, Matt Robshaw, and Yannick Seurin. AES Security Report. Technical Report D.STVL.2, European Network of Excellence in Cryptology, 2004. `http://www.ecrypt.eu.org/ecrypt1/documents/D.STVL.2-1.0.pdf`.

[106] Carlos Cid and Gaëtan Leurent. An Analysis of the XSL Algorithm. In Bimal Roy, editor, *Advances in Cryptology – ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 333–352. Springer Berlin / Heidelberg, 2005.

[107] Jolyon Clulow. On the Security of PKCS #11. In Colin Walter, Çetin Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2003*, volume 2779 of *Lecture Notes in Computer Science*, pages 411–425. Springer Berlin / Heidelberg, 2003.

[108] Hubert Comon-Lundh and Véronique Cortier. Computational soundness of observational equivalence. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM Conference on Computer and Communications Security*, pages 109–118. ACM, 2008.

[109] Véronique Cortier and Bogdan Warinschi. Computationally Sound, Automated Proofs for Security Protocols. In Mooly Sagiv, editor, *Programming Languages and Systems*, volume 3444 of *Lecture Notes in Computer Science*, pages 157–171. Springer Berlin / Heidelberg, 2005.

[110] Nicolas Courtois and Josef Pieprzyk. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In Yuliang Zheng, editor, *Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Springer Berlin / Heidelberg, 2002.

[111] Anupam Datta, Ante Derek, John C. Mitchell, and Arnab Roy. Protocol Composition Logic (PCL). *Electronic Notes in Theoretica Computer Science*, 172:311–358, 2007.

[112] Magnus Daum and Stefan Lucks. The Story of Alice and her Boss: Hash Functions and the Blind Passenger Attack, May 2005. Rump session presentation at Eurocrypt 2005.

[113] Blandine Debraize and Irene Marquez Corbella. Fault analysis of the stream cipher Snow 3G. In *Fault Diagnosis and Tolerance in Cryptography (FDTC), 2009 Workshop on*, pages 103–110. IEEE, 2009.

[114] Stéphanie Delaune, Steve Kremer, and Graham Steel. Formal Analysis of PKCS#11. In *21st IEEE Computer Security Foundations Symposium*, pages 331–344. IEEE Computer Society, 2008.

[115] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.1. RFC4346, `http://www.ietf.org/rfc/rfc4346.txt`.

[116] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol, Version 1.2, 2008. IETF RFC5246, `http://tools.ietf.org/html/rfc5246`.

[117] W. Diffie and M. E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22:644–654, 1976.

[118] Hans Dobbertin. Cryptanalysis of MD5 Compress, May 1996. Rump session presentation at Eurocrypt 1996.

[119] Hans Dobbertin. RIPEMD with Two-Round Compress Function is Not Collision-Free. *Journal of Cryptology*, 10(1):51–70, 1997.

[120] Hans Dobbertin, Antoon Bosselaers, and Bart Preneel. RIPEMD-160: A Strengthened Version of RIPEMD. In Dieter Gollmann, editor, *FSE*, volume 1039 of *LNCS*, pages 71–82. Springer, 1996.

[121] Danny Dolev and Andrew Chi-Chih Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–207, 1983.

[122] Orr Dunkelman, Nathan Keller, and Adi Shamir. A Practical-Time Related-Key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 393–410. Springer Berlin / Heidelberg, 2010.

[123] Thai Duong and Juliano Rizzo. Here Come The $\oplus$ Ninjas. `http://packetstormsecurity.com/files/105499/Browser-Exploit-Against-SSL-TLS.html`, May 13th 2011.

[124] Thai Duong and Juliano Rizzo. The CRIME Attack, Sep 2012. `https://docs.google.com/presentation/d/11eBmGiHbYcHR9gL5nDyZChu_-lCa2GizeuOfaLU2HOU/`.

[125] D. Eastlake, J. Reagle, and D. Solo. XML-Signature Syntax and Processing. IETF RFC3275, `http://tools.ietf.org/html/rfc3275`.

[126] Andres Einmann. ID-kaardi tarkvara ohtlik turvaauk sai paranduse. `http://www.postimees.ee/1359094/id-kaardi-tarkvara-ohtlik-turvaauk-sai-paranduse`.

[127] Taher ElGamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In George Blakley and David Chaum, editors, *Advances in Cryptology*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18. Springer Berlin / Heidelberg, 1985.

[128] Federation of Finnish Financial Service. TUPAS Identification Service for Service Providers, versioon 2.3c, jaanuar 2011. http://www.fkl.fi/en/themes/e-services/tupas/Pages/default.aspx.

[129] Scott Fluhrer and David McGrew. Statistical Analysis of the Alleged RC4 Keystream Generator. In Gerhard Goos, Juris Hartmanis, Jan van Leeuwen, and Bruce Schneier, editors, *Fast Software Encryption*, volume 1978 of *Lecture Notes in Computer Science*, pages 66–71. Springer Berlin / Heidelberg, 2001.

[130] Sebastian Gajek, Mark Manulis, Olivier Pereira, Ahmad-Reza Sadeghi, and Jörg Schwenk. Universally Composable Security Analysis of TLS. In Joonsang Baek, Feng Bao, Kefei Chen, and Xuejia Lai, editors, *Provable Security*, volume 5324 of *Lecture Notes in Computer Science*, pages 313–327. Springer Berlin / Heidelberg, 2008.

[131] Yoel Gluck, Neal Harris, and Angelo Prado. BREACH: reviving the CRIME attack. http://www.breachattack.com, July 12th 2013.

[132] Shafi Goldwasser and Silvio Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[133] Dan Goodin. Crack in Internet's foundation of trust allows HTTPS session hijacking. Ars Technica, Sep 13th 2012. http://arstechnica.com/security/2012/09/crime-hijacks-https-sessions/.

[134] Andrew D. Gordon and Alan Jeffrey. Authenticity by Typing for Security Protocols. *Journal of Computer Security*, 11(4):451–520, 2003.

[135] Andrew D. Gordon and Alan Jeffrey. Types and effects for asymmetric cryptographic protocols. *Journal of Computer Security*, 12(3-4):435–483, 2004.

[136] Aleksei Gorny. Analysis of Chip-card Based Authentication. Tartu Ülikool, bakalaureusetöö, 2009.

[137] Robert Graham. Tor is still DHE 1024 (NSA crackable). http://blog.erratasec.com/2013/09/tor-is-still-dhe-1024-nsa-crackable.html, Sep 2013.

[138] Joseph Lorenzo Hall. What the heck is going on with NIST's cryptographic standard, SHA-3?, 2013. https://www.cdt.org/blogs/joseph-lorenzo-hall/2409-nist-sha-3.

[139] Vello Hanson, Märt Laur, Ahto Buldas, and Imbi Nõgisto. *Andmekaitse ja infoturbe seletussõnastik*. Cybernetica AS, 2011.

[140] Changhua He, Mukund Sundararajan, Anupam Datta, Ante Derek, and John C. Mitchell. A modular correctness proof of IEEE 802.11i and TLS. In Vijay Atluri, Catherine Meadows, and Ari Juels, editors, *ACM Conference on Computer and Communications Security*, pages 2–15. ACM, 2005.

[141] Sven Heiberg, Peeter Laud, and Jan Willemson. The Application of I-voting for Estonian Parliamentary Elections of 2011. In *Proceedings of VoteID 2011*, LNCS. Springer, 2011.

[142] Wilko Henecka, Alexander May, and Alexander Meurer. Correcting Errors in RSA Private Keys. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 351–369. Springer Berlin / Heidelberg, 2010.

[143] Nadia Heninger, Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices. In *Proceedings of the 21st USENIX Security Symposium*, pages 205–220, 2012.

[144] Takeshi Imamura, Blair Dillaway, and Ed Simon. XML Encryption Syntax and Processing, 10. detsember 2002.

[145] Sebastiaan Indesteege, Florian Mendel, Bart Preneel, and Christian Rechberger. Collisions and Other Non-random Properties for Step-Reduced SHA-256. In Roberto Avanzi, Liam Keliher, and Francesco Sica, editors, *Selected Areas in Cryptography*, volume 5381 of *Lecture Notes in Computer Science*, pages 276–293. Springer Berlin / Heidelberg, 2009.

[146] Tetsu Iwata and Kaoru Kurosawa. OMAC: One-Key CBC MAC. In Thomas Johansson, editor, *Fast Software Encryption*, volume 2887 of *Lecture Notes in Computer Science*, pages 129–153. Springer Berlin / Heidelberg, 2003.

[147] Ahto Jaago, Liisa Lukin, and Ago Vesmes. DigiDocService spetsifikatsioon, versioon 2.125, 18. märts 2013. `http://www.sk.ee/upload/files/DigiDocService_spec_est.pdf`.

[148] Tibor Jager, Florian Kohlar, Sven Schäge, and Jörg Schwenk. On the security of tls-dhe in the standard model. In Safavi-Naini and Canetti [230], pages 273–293.

[149] Romain Janvier, Yassine Lakhnech, and Laurent Mazaré. Completing the Picture: Soundness of Formal Encryption in the Presence of Active Adversaries. In Mooly Sagiv, editor, *Programming Languages and Systems*, volume 3444 of *Lecture Notes in Computer Science*, pages 172–185. Springer Berlin / Heidelberg, 2005.

[150] J. Jonsson and B. Kaliski. Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1. RFC3447, Feb 2003. `http://tools.ietf.org/html/rfc3447`.

[151] Jakob Jonsson. On the Security of CTR + CBC-MAC. In Kaisa Nyberg and Howard Heys, editors, *Selected Areas in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 76–93. Springer Berlin / Heidelberg, 2003.

[152] Vivek Kamath, Ashwin Palekar, and Mark Wodrich. Microsoft's PEAP version 0 (Implementation in Windows XP SP1), 25. Oktoober 2002. `http://tools.ietf.org/html/draft-kamath-pppext-peapv0-00`.

[153] C. Kaufman, P. Hoffman, Y. Nir, and P. Eronen. Internet Key Exchange Protocol Version 2 (IKEv2). IETF RFC 5996, `http://tools.ietf.org/html/rfc5996`.

[154] John Kelsey. Compression and information leakage of plaintext. In Joan Daemen and Vincent Rijmen, editors, *FSE*, volume 2365 of *Lecture Notes in Computer Science*, pages 263–276. Springer, 2002.

[155] S. Kent. IP Encapsulating Security Payload (ESP). IETF RFC4303, `http://tools.ietf.org/html/rfc4303`.

[156] Auguste Kerckhoffs. La cryptographie militaire. *Journal des sciences militaires*, IX:161–191, 1883.

[157] Jongsung Kim, Alex Biryukov, Bart Preneel, and Seokhie Hong. On the Security of HMAC and NMAC Based on HAVAL, MD4, MD5, SHA-0 and SHA-1 (Extended Abstract). In Roberto De Prisco and Moti Yung, editors, *Security and Cryptography for Networks*, volume 4116 of *Lecture Notes in Computer Science*, pages 242–256. Springer Berlin / Heidelberg, 2006.

[158] Andreas Klein. Attacks on the RC4 stream cipher. In *Designs, Codes and Cryptography*, volume 48, pages 269–286. Springer Netherlands, 2008.

[159] Thorsten Kleinjung, Kazumaro Aoki, Jens Franke, Arjen Lenstra, Emmanuel Thomé, Joppe Bos, Pierrick Gaudry, Alexander Kruppa, Peter Montgomery, Dag Osvik, Herman te Riele, Andrey Timofeev, and Paul Zimmermann. Factorization of a 768-Bit RSA Modulus. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 333–350. Springer Berlin / Heidelberg, 2010.

[160] Vlastimil Klíma, Ondrej Pokorný, and Tomáš Rosa. Attacking RSA-Based Sessions in SSL/TLS. In Colin Walter, Çetin Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2003*, volume 2779 of *Lecture Notes in Computer Science*, pages 426–440. Springer Berlin / Heidelberg, 2003.

[161] Neal Koblitz. Elliptic Curve Cryptosystems. *Mathematics of Computation*, 48(177):203–209, 1987.

[162] Neal Koblitz. Hyperelliptic cryptosystems. *Journal of Cryptology*, 1:139–150, 1989.

[163] Neal Koblitz and Alfred Menezes. Another Look at Generic Groups. *Advances in Mathematics of Communications*, 1(1):13–28, 2007.

[164] Neal Koblitz, Alfred Menezes, and Scott Vanstone. The State of Elliptic Curve Cryptography. *Designs, Codes and Cryptography*, 19:173–193, 2000.

[165] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In Michael Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer Berlin / Heidelberg, 1999.

[166] Paul Kocher, Joshua Jaffe, Benjamin Jun, and Pankaj Rohatgi. Introduction to differential power analysis. *Journal of Cryptographic Engineering*, 1:5–27, 2011.

[167] Paul C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *CRYPTO*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, 1996.

[168] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-Hashing for Message Authentication. IETF RFC2104, `http://tools.ietf.org/html/rfc2104`.

[169] Hugo Krawczyk. The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?). In Joe Kilian, editor, *Advances in Cryptology — CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 310–331. Springer Berlin / Heidelberg, 2001.

[170] Hugo Krawczyk, Kenneth G. Paterson, and Hoeteck Wee. On the Security of the TLS Protocol: A Systematic Analysis. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013*, volume 8042 of *Lecture Notes in Computer Science*, pages 429–448. Springer Berlin Heidelberg, 2013.

[171] Adam Langley. Enhancing digital certificate security. Google Online Security Blog, 3. jaanuar 2013. http://googleonlinesecurity.blogspot.com/2013/01/enhancing-digital-certificate-security.html.

[172] Jeff Larson. Revealed: The NSA's Secret Campaign to Crack, Undermine Internet Security. http://www.propublica.org/article/the-nsas-secret-campaign-to-crack-undermine-internet-encryption, Sep 2013.

[173] Peeter Laud. Encryption Cycles and Two Views of Cryptography. In *NORDSEC 2002 - Proceedings of the 7th Nordic Workshop on Secure IT Systems*, pages 85–100. Karlstad University, 2002.

[174] Peeter Laud and Meelis Roos. Formal Analysis of the Estonian Mobile-ID Protocol. In Audun Jøsang, Torleiv Maseng, and Svein J. Knapskog, editors, *NordSec*, volume 5838 of *Lecture Notes in Computer Science*, pages 271–286. Springer, 2009.

[175] Märt Laur. *X-tee 5.0 turvaserveri kasutusjuhend. versioon 5.04*, 14. märts 2011. http://ee.x-rd.net/docs/est/turvaserveri_kasutusjuhend.pdf.

[176] L. Law and J. Solinas. Suite B Cryptographic Suites for IPsec. IETF RFC 4869.

[177] Arjen K. Lenstra, James P. Hughes, Maxime Augier, Joppe W. Bos, Thorsten Kleinjung, and Christophe Wachter. Ron was wrong, Whit is right. Cryptology ePrint Archive, Report 2012/064, 2012. http://eprint.iacr.org/.

[178] Chu-Wee Lim and Khoongming Khoo. An Analysis of XSL Applied to BES. In Alex Biryukov, editor, *Fast Software Encryption*, volume 4593 of *Lecture Notes in Computer Science*, pages 242–253. Springer Berlin / Heidelberg, 2007.

[179] M. Lochter and J. Merkle. Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation. RFC 5639, http://www.ietf.org/rfc/rfc5639.txt.

[180] Hans Lõugas. Läti tudeng leidis Eesti pangalinkidest turvaaugu. Eesti Päevaleht, 27. september 2012. http://www.epl.ee/news/eesti/lati-tudeng-leidis-eesti-pangalinkidest-turvaaugu.d?id=65024204.

[181] Gavin Lowe. Breaking and Fixing the Needham-Schroeder Public-Key Protocol Using FDR. In Tiziana Margaria and Bernhard Steffen, editors, *TACAS*, volume 1055 of *Lecture Notes in Computer Science*, pages 147–166. Springer, 1996.

[182] Gavin Lowe. Casper: A Compiler for the Analysis of Security Protocols. In *CSFW*, pages 18–30. IEEE Computer Society, 1997.

[183] Subhamoy Maitra and Goutam Paul. New Form of Permutation Bias and Secret Key Leakage in Keystream Bytes of RC4. In Kaisa Nyberg, editor, *Fast Software Encryption*, volume 5086 of *Lecture Notes in Computer Science*, pages 253–269. Springer Berlin / Heidelberg, 2008.

[184] Itsik Mantin and Adi Shamir. A Practical Attack on Broadcast RC4. In Mitsuru Matsui, editor, *Fast Software Encryption*, volume 2355 of *Lecture Notes in Computer Science*, pages 87–104. Springer Berlin / Heidelberg, 2002.

[185] Moxie Marlinspike. Divide and Conquer: Cracking MS-CHAPv2 with a 100% success rate. `https://www.cloudcracker.com/blog/2012/07/29/cracking-ms-chap-v2/`, 29. juuli 2012.

[186] Alexander May and Maike Ritzenhofen. Implicit Factoring: On Polynomial Time Factoring Given Only an Implicit Hint. In *Public Key Cryptography*, volume 5443 of *Lecture Notes in Computer Science*, pages 1–14. Springer, 2009.

[187] Catherine Meadows. The NRL Protocol Analysis Tool: A Position Paper. In *CSFW*, page 227, 1991.

[188] Catherine Meadows. Analysis of the Internet Key Exchange Protocol using the NRL Protocol Analyzer. In *IEEE Symposium on Security and Privacy*, pages 216–231, 1999.

[189] Florian Mendel, Norbert Pramstaller, Christian Rechberger, and Vincent Rijmen. On the Collision Resistance of RIPEMD-160. In Sokratis K. Katsikas, Javier Lopez, Michael Backes, Stefanos Gritzalis, and Bart Preneel, editors, *ISC*, volume 4176 of *LNCS*, pages 101–116. Springer, 2006.

[190] A.J. Menezes, T. Okamoto, and S.A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Transactions on Information Theory*, 39(5):1639–1646, 1993.

[191] Alfred Menezes, Paul van Oorschot, and Scott Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[192] Ulrike Meyer and Susanne Wetzel. A man-in-the-middle attack on UMTS. In *Proceedings of the 3rd ACM workshop on Wireless security*, WiSe '04, pages 90–97. ACM, 2004.

[193] Daniele Micciancio and Saurabh Panjwani. Adaptive Security of Symbolic Encryption. In Joe Kilian, editor, *TCC*, volume 3378 of *Lecture Notes in Computer Science*, pages 169–187. Springer, 2005.

[194] Daniele Micciancio and Bogdan Warinschi. Soundness of Formal Encryption in the Presence of Active Adversaries. In Moni Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 133–151. Springer, 2004.

[195] Victor Miller. Use of Elliptic Curves in Cryptography. In Hugh Williams, editor, *Advances in Cryptology – CRYPTO '85 Proceedings*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426. Springer Berlin / Heidelberg, 1986.

[196] John C. Mitchell, Mark Mitchell, and Ulrich Stern. Automated analysis of cryptographic protocols using Mur-phi. In *IEEE Symposium on Security and Privacy*, pages 141–151. IEEE Computer Society, 1997.

[197] Kevin D. Mitnick and William L. Simon. *The Art of Deception. Controlling the Human Element of Security*. Wiley Publishing, Inc., 2002.

[198] J Molina-Gil, P Caballero-Gil, C Caballero-Gil, and Amparo Fúster-Sabater. Analysis and Implementation of the SNOW 3G Generator Used in 4G/LTE Systems. In *International Joint Conference SOCO'13-CISIS'13-ICEUTE'13*, pages 499–508. Springer, 2014.

[199] Gordon E. Moore. Cramming More Components onto Integrated Circuits. *Electronics*, 8:114–117, 1965.

[200] Gordon E. Moore. Progress in digital integrated electronics. In *Electron Devices Meeting, 1975 International*, pages 11–13, 1975.

[201] Shiho Moriai. Security evaluation of cryptographic technology. *NICT News*, 426:2–3, March 2013.

[202] Sean Murphy and Matthew Robshaw. Essential Algebraic Structure within the AES. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 1–16. Springer Berlin / Heidelberg, 2002.

[203] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. X.509 Internet Public Key Infrastructure; Online Certificate Status Protocol — OCSP, juuni 1999.

[204] C. Neuman, T. Yu, S. Hartman, and K. Raeburn. The Kerberos Network Authentication Service (V5). IETF RFC4120, `http://tools.ietf.org/html/rfc4120`.

[205] Hanne Riis Nielson and Flemming Nielson. A flow-sensitive analysis of privacy properties. In *CSF*, pages 249–264. IEEE Computer Society, 2007.

[206] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL - A Proof Assistant for Higher-Order Logic*, volume 2283 of *Lecture Notes in Computer Science*. Springer, 2002.

[207] NIST. Announcing the Advanced Encryption Standard (AES). Technical report, CSRC, 2001. `http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf`.

[208] Karsten Nohl and Sylvain Munaut. Wideband GSM Sniffing. In *27th Chaos Communication Congress*, 2010.

[209] Toshihiro Ohigashi and Masakatu Morii. A Practical Message Falsification Attack on WPA. In *2009 Joint Workshop on Information Security*, Kaohsiung, Taiwan, August 2009.

[210] Dag Osvik, Adi Shamir, and Eran Tromer. Cache Attacks and Countermeasures: The Case of AES. In David Pointcheval, editor, *Topics in Cryptology – CT-RSA 2006*, volume 3860 of *Lecture Notes in Computer Science*, pages 1–20. Springer Berlin / Heidelberg, 2006.

[211] G. Pall and G. Zorn. Microsoft Point-To-Point Encryption (MPPE) Protocol. IETF RFC3078, `http://tools.ietf.org/html/rfc3078`.

[212] Arnis Paršovs. Security Analysis of Internet Bank Authentication Protocols and their Implementations. Master's thesis, Tallinna Tehnikaülikool, 2012.

[213] Arnis Paršovs. Practical issues with tls client certificate authentication. Cryptology ePrint Archive, Report 2013/538, 2013. `http://eprint.iacr.org/`.

[214] Goutam Paul, Siddheshwar Rathi, and Subhamoy Maitra. On non-negligible bias of the first output byte of RC4 towards the first three bytes of the secret key. In *Designs, Codes and Cryptography*, volume 49, pages 123–134. Springer Netherlands, 2008.

[215] Souradyuti Paul and Bart Preneel. Analysis of Non-fortuitous Predictive States of the RC4 Keystream Generator. In Thomas Johansson and Subhamoy Maitra, editors, *Progress in Cryptology – INDOCRYPT 2003*, volume 2904 of *Lecture Notes in Computer Science*, pages 67–70. Springer Berlin / Heidelberg, 2003.

[216] Lawrence C. Paulson. The Inductive Approach to Verifying Cryptographic Protocols. *Journal of Computer Security*, 6(1-2):85–128, 1998.

[217] Lawrence C. Paulson. Inductive Analysis of the Internet Protocol TLS. *ACM Transactions on Information and System Security*, 2(3):332–351, 1999.

[218] Tiit Pikma and Märt Laur. Elektroonilise identiteedi (eID) rakendusjuhend tarkvaraarendajatele, november 2012. `https://eid.eesti.ee`.

[219] Bart Preneel and Paul van Oorschot. MDx-MAC and Building Fast MACs from Hash Functions. In Don Coppersmith, editor, *Advances in Cryptology - CRYPT0' 95*, volume 963 of *Lecture Notes in Computer Science*, pages 1–14. Springer Berlin / Heidelberg, 1995.

[220] K. Raeburn. Encryption and Checksum Specifications for Kerberos 5. IETF RFC3961, `http://tools.ietf.org/html/rfc3961`.

[221] Marsh Ray and Steve Dispensa. Renegotiating TLS, 4. november 2009. `http://extendedsubset.com/?m=200911`.

[222] E. Rescorla. Diffie-Hellman Key Agreement Method. IETF RFC 2631. `http://tools.ietf.org/html/rfc2631`.

[223] E. Rescorla, M. Ray, S. Dispensa, and N. Oskov. Transport Layer Security (TLS) Renegotiation Indication Extension. IETF RFC5746, `http://tools.ietf.org/html/rfc5746`.

[224] Ivan Ristic. Defending against the BREACH Attack. Qualys Blogs, `https://community.qualys.com/blogs/securitylabs/2013/08/07/defending-against-the-breach-attack`, Aug 7th 2013.

[225] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120–126, February 1978.

[226] Ronald Rivest. The MD5 Message-Digest Algorithm, April 1992. RFC1321, `http://tools.ietf.org/html/rfc1321`.

[227] Ronald Rivest and Adi Shamir. Efficient Factoring Based on Partial Information. In Franz Pichler, editor, *Advances in Cryptology – EUROCRYPT' 85*, volume 219 of *Lecture Notes in Computer Science*, pages 31–34. Springer Berlin / Heidelberg, 1986.

[228] P. Rogaway and D. Wagner. A Critique of CCM. Cryptology ePrint Archive, Report 2003/070, 2003. `http://eprint.iacr.org/`.

[229] Phillip Rogaway, Mihir Bellare, and John Black. OCB: A block-cipher mode of operation for efficient authenticated encryption. *ACM Transactions in Information and System Security*, 6(3):365–403, 2003.

[230] Reihaneh Safavi-Naini and Ran Canetti, editors. *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*. Springer, 2012.

[231] M. Salter, E. Rescorla, and R. Housley. Suite B Profile for Transport Layer Security (TLS). IETF RFC 5430.

[232] Yu Sasaki and Kazumaro Aoki. Finding Preimages in Full MD5 Faster Than Exhaustive Search. In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 134–152. Springer Berlin / Heidelberg, 2009.

[233] T. Satoh and K. Araki. Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. *Commentarii Mathematici Universitatis Sancti Pauli*, 47:81–92, 1998.

[234] Bruce Schneier. Description of a New Variable-Length Key, 64-bit Block Cipher (Blowfish). In *FSE*, volume 809 of *Lecture Notes in Computer Science*, pages 191–204. Springer, 1993.

[235] Bruce Schneier. When Will We See Collisions for SHA-1? `http://www.schneier.com/blog/archives/2012/10/when_will_we_se.html`, October 2012.

[236] I. A. Semaev. Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p. *Mathematics of Computation*, 67:353–356, January 1998.

[237] Sertifitseerimiskeskus AS. *DigiDoc formaadi kirjeldus, versioon 1.3*, 2006. `http://www.id.ee/public/DigiDoci_vorming_1.3.2.pdf`.

[238] Sertifitseerimiskeskus AS. Sertifikaadid Eesti Vabariigi isikutunnistusel, versioon 3.3. `http://www.sk.ee/upload/files/ESTEID_profiil_et-3_3.pdf`, 1. jaanuar 2010.

[239] Sertifitseerimiskeskus AS. DigiDoc teegid. `http://id.ee/?id=28729`, 31. jaanuar 2011.

[240] Sertifitseerimiskeskus AS. Sertifitseerimispõhimõtted – CPS, versioon 2.5. `http://www.sk.ee/upload/files/SK_CPS_v2_5.pdf`, 1. jaanuar 2011.

[241] Sertifitseerimiskeskus AS. By the end of the year organisation certificates based on the 1024-bit key will not be used any longer, 2012. `http://sk.ee/en/News/by-the-end-of-the-year-organisation-certificates-based-on-the-1024-bit-key-will`

[242] Sertifitseerimiskeskus AS. ESTEID-kaardi sertifitseerimispoliitika, versioon 3.3. `https://www.sk.ee/upload/files/SK-CP-ESTEID-20120901v3_3.pdf`, 1. september 2012.

[243] Vitaly Shmatikov and Ulrich Stern. Efficient Finite-State Analysis for Large Security Protocols. In *CSFW*, pages 106–115, 1998.

[244] Tom Simonite. Math Advances Raise the Prospect of an Internet Security Crisis. `http://www.technologyreview.com/news/517781/math-advances-raise-the-prospect-of-an-internet-security-crisis/`, August 2013.

[245] Veiko Sinivee, Kersti Üts, and Kristi Uukkivi. CDigiDoc Programmer's Guide, versioon 3.7, 22. jaanuar 2013. https://svn.eesti.ee/projektid/idkaart_public.

[246] Veiko Sinivee, Kersti Üts, and Kristi Uukkivi. JDigiDoc Programmer's Guide, versioon 3.7, 22. jaanuar 2013. https://svn.eesti.ee/projektid/idkaart_public.

[247] N. P. Smart. The Discrete Logarithm Problem on Elliptic Curves of Trace One. *Journal of Cryptology*, 12:193–196, 1999.

[248] James Snodgrass and Josh Hoover. BYO-Disaster and Why Corporate Wireless Security Still Sucks, 3. august 2013. Ettekanne konverentsil DEF CON XXI, https://www.defcon.org/images/defcon-21/dc-21-presentations/djwishbone-PuNk1nPo0p/DEFCON-21-djwishbone-PuNk1nPo0p-BYO-Disaster-Updated.pdf.

[249] Dawn Xiaodong Song. Athena: A New Efficient Automatic Checker for Security Protocol Analysis. In *CSFW*, pages 192–202, 1999.

[250] JH. Song, R. Poovendran, J. Lee, and T. Iwata. The AES-CMAC Algorithm. IETF RFC4493, http://tools.ietf.org/html/rfc4493.

[251] Jacques Stern, David Pointcheval, John Malone-Lee, and Nigel Smart. Flaws in Applying Proof Methodologies to Signature Schemes. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 215–224. Springer Berlin / Heidelberg, 2002.

[252] Marc Stevens. On Collisions for MD5. Master's thesis, Eindhoven University of Technology, 2007.

[253] Marc Stevens. New collision attacks on SHA-1 based on optimal joint local-collision analysis. In *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 245–261. Springer, 2013.

[254] Marc Stevens, Alexander Sotirov, Jacob Appelbaum, Arjen Lenstra, David Molnar, Dag Osvik, and Benne de Weger. Short Chosen-Prefix Collisions for MD5 and the Creation of a Rogue CA Certificate. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 55–69. Springer Berlin / Heidelberg, 2009.

[255] Frank A. Stevenson. Cryptanalysis of contents scrambling system. http://www.cs.cmu.edu/~dst/DeCSS/FrankStevenson/analysis.ps, November 1999.

[256] Douglas R. Stinson. *Cryptography: Theory and Practice*. Chapman & Hall / CRC, 2002.

[257] Swedbank. Pangalingi tehniline spetsifikatsioon, versioon 1.2, juuni 2012. https://www.swedbank.ee/static/pdf/business/d2d/paymentcollection/info_banklink_techspec_2012_06_26_est.pdf.

[258] Don Syme. The F# 2.0 Language Specification, Aprill 2010. http://research.microsoft.com/en-us/um/cambridge/projects/fsharp/manual/spec.pdf.

[259] Paul Syverson and Paul C. van Oorschot. On unifying some cryptographic protocol logics. In *1994 IEEE Symposium on Research in Security and Privacy*, pages 14–28. IEEE Computer Society, 1994.

[260] Erik Tews and Martin Beck. Practical attacks against WEP and WPA. In David A. Basin, Srdjan Capkun, and Wenke Lee, editors, *WISEC*, pages 79–86. ACM, 2009.

[261] Erik Tews, Ralf-Philipp Weinmann, and Andrei Pyshkin. Breaking 104 Bit WEP in less than 60 seconds. In *Proceedings of the 8th international conference on Information security applications*, WISA'07, pages 188–202, Berlin, Heidelberg, 2007. Springer-Verlag.

[262] Kristi Uukkivi. Libdigidocpp Programmer's Guide, versioon 1.1, 30. mai 2013. `http://www.id.ee/public/SK-CPP-PRG-GUIDE.pdf`.

[263] Serge Vaudenay. Security Flaws Induced by CBC Padding — Applications to SSL, IPSEC, WTLS... In Lars Knudsen, editor, *Advances in Cryptology — EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 534–545. Springer Berlin / Heidelberg, 2002.

[264] Serge Vaudenay and Martin Vuagnoux. Passive-Only Key Recovery Attacks on RC4. In Carlisle Adams, Ali Miri, and Michael Wiener, editors, *Selected Areas in Cryptography*, volume 4876 of *Lecture Notes in Computer Science*, pages 344–359. Springer, 2007.

[265] Stefan Viehböck. Brute forcing Wi-Fi Protected Setup. `http://sviehb.files.wordpress.com/2011/12/viehboeck_wps.pdf`, 26. detsember 2011.

[266] V.Rijmen and P.S.L.M.Barreto. The Whirlpool hash function. `http://www.larc.usp.br/~pbarreto/WhirlpoolPage.html`.

[267] Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, and Xiuyuan Yu. Cryptanalysis of the Hash Functions MD4 and RIPEMD. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *LNCS*, pages 1–18. Springer, 2005.

[268] Xiaoyun Wang and Hongbo Yu. How to Break MD5 and Other Hash Functions. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 561–561. Springer Berlin / Heidelberg, 2005.

[269] D. Whiting, R. Housley, and N. Ferguson. Counter with CBC-MAC (CCM). IETF RFC3610, `http://tools.ietf.org/html/rfc3610`.

[270] Michael J. Wiener. Cryptanalysis of short RSA secret exponents. *IEEE Transactions on Information Theory*, 36(3):553–558, 1990.

[271] Jan Willemson. Sissejuhatus Krüptoloogiasse. Loengumärkmed, Tartu Ülikool, 2004. `http://home.cyber.ee/jan/intro/`.

[272] Tao Xie and Dengguo Feng. How To Find Weak Input Differences For MD5 Collision Attacks. Technical report, IACR, 2009. `http://eprint.iacr.org/2009/223.pdf`.

[273] Josh Yavor. BYOD PEAP Show, 4. august 2013. Ettekanne konverentsil DEF CON XXI, `https://www.defcon.org/images/defcon-21/dc-21-presentations/Yavor/DEFCON-21-Yavor-The-BYOD-PEAP-Show-Updated.pdf`.

[274] L. Zhu and B. Tung. Public Key Cryptography for Initial Authentication in Kerberos (PKINIT). IETF RFC4556, `http://tools.ietf.org/html/rfc4556`.

[275] Li Zhuang, Feng Zhou, and J. D. Tygar. Keyboard acoustic emanations revisited. *ACM Trans. Inf. Syst. Secur.*, 13(1):3:1–3:26, November 2009.

[276] G. Zorn. Microsoft PPP CHAP Extensions, Version 2. IETF RFC2759, `http://tools.ietf.org/html/rfc2759`.

A Küsimustikud

Käesolevas lisas on toodud ära uurimuse läbiviimiseks koostatud küsimustikud. Esimene küsimustik leidis kasutust erinevate infosüsteemide analüüsil, teine aga kiibipõhiste krüptolahenduste uurimisel.

A.1 Infosüsteemide küsimustik

1. Millistes asutuse infosüsteemi osades kasutatakse krüptoalgoritme?
2. Mida tähendaks nende muutmise vajadus, st kui töömahukas oleks nende väljavahetamine?
3. Kas kõik krüptoalgoritmide kasutusjuhud on kaetud vastutusega, st iga juhu jaoks leidub vastutav isik, kes vajadusel organiseerib krüptoalgoritmide väljavahetamise?
4. Kui palju on asutuse enda poolt hallatavat koodi/riistvara, kus krüptoalgoritme kasutatakse?
5. Kuidas jõuab vastutavate isikuteni info krüptoalgoritmide nõrkuse kohta?

A.2 Kiibipõhiste krüptolahenduste küsimustik

1. Üldküsimused kiibilahendusest ja krüptoalgoritmidest
   - 1.1 Kas kiibilahendus põhineb ainulaadsel füüsilisel kiibil või on see mingi tüüpkiibi konfiguratsioon?
   - 1.2 Kirjelda lühidalt kiibilahenduse üldist ülesehitust alates ränis realiseeritud algoritmidest, mikroprogrammidest, kaardi muutmälusse salvestatud koodist ja arvutis käivitatavast abikoodist.
   - 1.3 Millised programsed osad on muudetavad peale kiibi füüsilist valmimist? Millised peale isikustamist?
   - 1.4 Millised krüptoalgoritmid on realiseeritud kiibilahenduse aluseks olevas kiibis? Millist osa nendest algoritmidest kasutab konkreetne rakendus?
   - 1.5 Kas ja kuidas oleks võimalik kiibilahenduses kasutatud krüptoalgoritme muuta, s.t. kui tema aluseks olevas tüüpkiibis on realiseeritud rohkem algoritme kui tegelikult kasutatakse, siis kui keeruline oleks tegelikkuses kasutatavat algoritmi muuta, näiteks RSA $\rightarrow$ DSA, vms.
2. Küsimused taristust
   - 2.1 Kuidas ja kus genereeritakse kiibis kasutatavad krüptograafilised võtmed? Kas kiibi sees või sellest väljaspool? Kes kontrollib võtmete genereerimise protseduuri?
   - 2.2 Kui võtmed genereeritakse kiibist väjaspool, siis kuidas jõuavad võtmed kiipi, kas võtmete transpordil kasutatakse ka mingeid krüptoalgoritme?

2.3 Kirjelda lühidalt kiibilahenduse elutsüklit alates füüsilisest tootmisest ja lõpetades isikustamisega.

2.4 Kas elutsükli protseduuride juures on kusagil kasutatud ka krüptoalgoritme? Kui jah, siis milleks (lühidalt)?

2.5 Kui elutsükli juures kasutatakse krüptoalgoritme, siis mida tähendaks nende väljavahetamine, juhul kui see osutuks vajalikuks seoses nende ebaturvaliseks muutumisega?

3. Küsimused turvahaldusprotseduuridest

3.1 Mis juhtub, kui mõni kasutatud krüptoalgoritmidest muutub ebaturvaliseks? Kuidas (mis kanalite kaudu) kiibi valmistaja sellistest intsidentidest teada saab ja kuidas neile reageeritakse?

3.2 Teatavasti ei ole kiibilahenduse jaoks olulised mitte ainult krüptoalgoritmide nõrkused, vaid ka nende füüsilise/programse teostuse nõrkused (näiteks mitmesugused külgründed, tn ajastuse, voolutarbe, jms abil). Kuidas ja kas kaardi valmistaja jälgib uute avastatud külgrünnete olulisust kiibilahenduse turvalisusele? Kas loodetakse lahenduse aluseks oleva tüüpkiibi tootja sellekohasele tegevusele?

3.3 Mis saab siis, kui avastatakse külgrünne, mis on seotud kaardi füüsilise teostusega? Kas võib olla võimalik vahest näiteks parandada füüsilise teostuse nõrkust (näiteks ajastusel põhinevaid ründeid) rakendusprogrammide muutmisega?

3.4 Milline osa kiibilahendusega seotud programmidest on avalik, piiratud kasutusega, salastatud, riigisaladus, jne.?