CYBERNETICA





Krüptoalgoritmid ning nende tugi teekides ja infosüsteemides

aruanne

Versioon 1.1

11. september 2026. a.

D-16-610

Avalik

©2026 Riigi Infosüsteemi Amet

Uuringu koostas Cybernetica AS Riigi Infosüsteemi Ameti tellimusel.

Projektijuhid:	Katrin Kivi (Riigi Infosüsteemi Amet)
	Martina Võrklaev (Riigi Infosüsteemi Amet)
	Tõnis Reimo (Riigi Infosüsteemi Amet)
	Sandhra-Mirella Valdma (Cybernetica)
	Peeter Laud (Cybernetica)
Autorid (Cybernetica):	Arne Ansper
	Alisa Pankova
	Calvin Pärn
	Aivo Kalu
	Kristjan Krips
	Peeter Laud
	Petr Muzikant
	Martin Suomalainen
	Taisto Treumann
	Jelizaveta Vakarjuk

Riigi Infosüsteemi Amet, Pärnu maantee 139a, 15169 Tallinn, Eesti
Email: ria@ria.ee, Web: https://www.ria.ee, Telefon: +372 663 0200

Cybernetica AS, Mäealuse 2/1, 12618 Tallinn, Eesti.
E-mail: info@cyber.ee, Web: https://www.cyber.ee, Telefon: +372 639 7991.

Sisukord

1 Sissejuhatus 8

2 Krüptoprimitiivid 9

2.1 Võtmepikkuste soovitused 9
2.2 Krüptoprimitiivide rakendamisest 13
2.3 Krüptograafilised räsifunktsioonid. 14
2.3.1 SHA-2 14
2.3.2 SHA-3 14
2.4 Parooliräsimise algoritmid 15
2.4.1 Argon2 16
2.5 Sümmeetrilise võtmega primitiivid 16
2.5.1 Plokkšifrid 16
2.5.2 Jadašifrid 16
2.5.3 Räsipõhised sümmeetrilised konstruktsioonid 17
2.6 Asümmeetrilised primitiivid 17
2.6.1 RSA 18
2.6.2 ElGamal krüptoskeem 18
2.6.3 Elliptkõveratel põhinevad süsteemid 18
2.6.4 Kvantturvalised skeemid 19
2.7 Hübriidrežiim 22

3 Protokollid ja protokollistikud 24

3.1 TLS 24
3.1.1 TLS 1.3 kirjeldus. 24
3.1.2 TLSi analüüsid ja ründed TLSi vastu 26
3.1.3 Kliendi autentimine TLS-seansis 29
3.1.4 Soovitused 31
3.1.5 Muud TLSi rakendamist kirjeldavad dokumendid 35
3.2 SSH 35
3.2.1 Kirjeldus 35
3.2.2 Analüüsid ja ründed 36
3.2.3 Soovitused 37

3.3 IPsec 37
3.3.1 Ülevaade 37
3.3.2 Analüüsid ja ründed 39
3.3.3 Soovitused 39
3.4 Virtuaalsed privaatvõrgud 42
3.4.1 OpenVPN 42
3.4.2 WireGuard 43
3.5 Bluetoothi ja NFC turvaaspektid 44
3.5.1 NFC 44
3.5.2 Bluetooth 45

4 Krüptokonteinerite vormingud 48
4.1 Andmete kodeerimise viisid 48
4.1.1 Kodeerimine biti- või baidijadana 48
4.1.2 Kodeerimine struktureeritud tekstina 48
4.2 Allkirjavormingud 48
4.3 Krüptograafiliste objektide kodeerimine 49
4.3.1 CMS 49
4.4 JSON Object Signing and Encryption 49
4.4.1 JSON Web Signature 50
4.4.2 JAdES 50
4.4.3 JSON Web Encryption 50
4.4.4 JSON Web Token 50
4.4.5 JSON Web Key 50
4.4.6 COSE 51
4.4.7 XMLi-põhised vormingud 51
4.5 ASiC konteinervorming 51
4.5.1 ASiC-S 51
4.5.2 ASiC-E 52
4.6 Mobiilne juhiluba ja kontrollitavad mandaadid 52
4.6.1 SD-JWT kontrollitavad mandaadid 53
4.6.2 ISO mDoc-vorming 55
4.6.3 W3C kontrollitavad mandaadid 57
4.7 CDOC ja CDOC2 59
4.7.1 CDOC 59

4.7.2 CDOC2 60

5 Autentimise, delegeerimise ja atribuutide esitamise protokollid 62

5.1 OpenID Connect 62
5.1.1 Protokolli kirjeldus 62
5.1.2 Liidendus 62
5.2 FIDO, WebAuthn ja pääsuvõtmed 64
5.2.1 WebAuthn 65
5.2.2 Pääsuvõtmed 66
5.3 Web eID 66
5.4 OAuth 2.0 67
5.4.1 Protokolli kirjeldus 68
5.4.2 Omamistõendus 69
5.5 Atribuutide esitamise protokollid 70
5.5.1 ISO 18013-5 ja ISO 18013-7 71
5.5.2 OpenID4VP 72
5.5.3 Digital credentials API 73

6 Teegid ja rakendused 75

6.1 Kõige levinumad ja olulisemad krüptoteegid: ülevaade 75
6.1.1 OpenSSL 77
6.1.2 wolfSSL 77
6.1.3 Mbed TLS 77
6.1.4 NSS 78
6.1.5 Botan 78
6.1.6 Bouncy Castle 79
6.1.7 Tink 79
6.2 Valikukriteeriumid 79
6.3 Arenduskeskkonnad 82
6.3.1 C ja C++ 82
6.3.2 Java 82
6.3.3 Go 82
6.3.4 Node.js 83
6.3.5 Rust 83
6.3.6 PHP 83

6.3.7 .NET 84
6.3.8 Python 84

7 Postkvant-krüptograafia kasutuselevõtt 85

7.1 Tugi turvaprotokollides 85
7.1.1 TLS 85
7.1.2 SSH 86
7.1.3 IPsec 87
7.1.4 WPA3 88
7.1.5 Kerberos. 88
7.1.6 WireGuard 88
7.1.7 OpenVPN 88
7.1.8 OAuth ja OpenID Connect 88
7.1.9 WebAuthn 89
7.1.10 Web eID 89
7.2 Tugi taristuprotokollides 89
7.2.1 IEEE 802.1X 89
7.2.2 Bluetooth 90
7.2.3 NFC ja RFID 90
7.2.4 NTLM 90
7.2.5 Azure 90
7.2.6 AWS. 91
7.3 Tugi andmestruktuurides ja -konteinerites 91
7.3.1 PKCS #11 91
7.3.2 X.509. 92
7.3.3 Mobiilne juhiluba 93
7.3.4 Kontrollitavad mandaadid 93
7.4 Tugi pika elueaga krüptokonstruktsioonides 93
7.4.1 Programmikoodi allkirjastamine 93
7.4.2 Püsivara allkirjastamine 93
7.5 Tugi sardsüsteemides 94
7.6 Tugi riistvaras 95
7.6.1 Biomeetrilised passid 95
7.6.2 Füüsilised turvamoodulid (HSM) 96
7.7 Krüptoinventuuri tegemise meetodid 96

7.8 Standardimine 97
7.8.1 Riiklike institutsioonide standardimisprotsessid 97
7.8.2 Rahvusvaheliste organisatsioonide standardid 98
7.9 Olemasolev krüptomaterjal 99

Summary in English 101

Kirjandus 102

1 Sissejuhatus

See uuringuaruanne on kaheksas omasugune, jätkates aastatel 2011–2021 [1, 2, 3, 4, 5, 6, 7] aruannete seeriat, mille seitsmendat versiooni uuendati aastal 2023 [8]. Uuringu eesmärk on anda ülevaade krüptograafiliste vahendite kasutamise hetkeseisust ja heast tavast.

Aruande struktuur on sarnane paari eelmisega [7, 8], mis pööravad tähelepanu enimkasutatavatele krüptograafilistele protokollidele, andmestruktuuridele ja teekidele. Aruanne algab krüptoalgoritmide kirjeldustega peatükis 2, mille juurest liigutakse edasi järjest keerulisemate turvafunktsioonide juurde: peatükk 3 käsitleb otspunktide vahelise liikluse turvamise protokolle, peatükk 4 kõrgema taseme turvafunktsioonides kasutatavaid andmestruktuure ja nende kodeeringuid ning peatükk 5 kõrgema taseme turvafunktsioone realiseerivaid protokolle. Üksikute peatükkide struktuuris võib olla muudatusi, kui oleme otsinud loomulikumat viisi materjali rühmitamiseks. Aruande viimane „regulaarne“ peatükk (ptk 6) kirjeldab krüptoteeke ja arenduskeskkondi selle kõige realiseerimiseks.

Selle aruande kirjutamise ajal on oluline teema kõikvõimalikes rakendustes kvantrünnetele avatud krüptoalgoritmide väljavahetamine postkvant-krüptograafia algoritmide vastu. Aruande viimases peatükis (ptk 7) püüame kirjeldada selle ülemineku seisu eri valdkondades. Me loodame, et üleminek läheb ilma suuremate tagasilöökideta ja aruande mõnes tulevases versioonis seda peatükki enam vaja ei ole.

Pärast eelmise aruande [8] avaldamist on suurenenud ühe turvafunktsiooni – digikukrute – olulisus. Seetõttu on aruandele lisandunud nende erinevaid detaile kirjeldavad alapeatükid. Kuivõrd digikukrute pinu on sügav, hõlmates nii andmete transporti, andmestruktuure kui ka (autentimise ja atribuutide esitamise) protokolle, leiab ka nende kirjeldusi nii peatükist 3, 4 kui ka 5.

Aruande lähtematerjaliks on eespool mainitud varasemaid Riigi Infosüsteemi Ameti tellitud uuringuid ning tuntud riiklike (NIST, BSI) ja rahvusvaheliste (ECCG) organisatsioonide samalaadsed uuringud ja soovitused. Lisaks toetub töö ETSI, NISTi ja IETFi standarditele ning eelretsenseeritud teadusartiklitele. Aruande tarkvaratehnilistes osades on rakendatud Cybernetica ASi pikaajalisi kogemusi krüptovormingute ja -teekide valikul ning nende kvaliteedi hindamisel. Aruande koosseis on määratud raamhanke “Krüptograafiliste algoritmide elutsükli uuringu“ alt teostatud hanke „Krüptograafiliste algoritmide elutsükli uuring 2025“ tööde kirjeldustega.

Aruanne on suunatud kõigile IT-taristu ning tarkvaraarenduse valdkondade juhtidele ja spetsialistidele.

2 Krüptoprimitiivid

2.1 Võtmepikkuste soovitused

Algoritmide turvatase määratakse bittides – mingi algoritmi b-bitine turvatase tähendab, et efektiivseim teadaolev rünne kasutab arvutusressurssi, mis kulub ligikaudu 2^b ploki krüptimiseks plokkšifriga. Turvataseme formaalsem definitsioon, vähemalt klassikaliste (st ilma kvantarvutita) ründajate jaoks on antud NISTi aruandes 800-57 [9, ptk. 5.6.1.1].

Kui aga arvestada ka kvantarvutitega, siis muutub turvataseme defineerimine pisut keerukamaks. Eelmises lõigus määratletud „b bitti turvalisust“ tähendab, et algoritmi ründamise keerukust võrreldakse plokkšifri võtme leidmisega, kus võtmed on b-bitised ja plokkšifril pole ühtegi nõrkust, mis muudaks võtme otsimise lihtsamaks kui kõigi võtmete läbiproovimine. Me võiksime võrrelda seda ka mõne teise ülesandega, mille sisuks võiks ikkagi olla mõne sümmeetrilise krüptograafia hulka loetava primitiivi mingil olulisel viisil murdmine, sest sümmeetriline krüptograafia on alati olnud turvatasemete defineerimise aluseks. Üks selline oluline ülesanne on räsifunktsioonile kollisiooni leidmine. Me teame, et kui räsifunktsioonil ei ole mingeid kollisiooni leidmist lihtsustavaid nõrkusi, siis vastavalt sünnipäevaparadoksile on tema kollisiooni leidmiseks vajaliku töö kogus proportsionaalne suurusega $2^{\ell/2}$, kus räsifunktsiooni väljund on ℓ-bitine. Seetõttu on tüüpiline, et kui meie eesmärk on saavutada b bitti turvalisust räsifunktsioonide kollisioonikindlusele toetuvas süsteemis, siis kasutame selleks $2b$-bitise väljundiga räsifunktsioone.

Kvantarvutite puhul ei ole enam korrektne lugeda b-bitise võtmega plokkšifrite ning $2b$-bitise väljundiga (kollisioonikindlate) räsifunktsioonide turvatasemeid „enam-vähem“ samaks, sest täieliku läbivaatuse algoritmid kvantarvutitel muudavad neid turvahinnanguid erinevalt. Groveri algoritm [10] vajab b-bitise võtmega plokkšifri võtmete läbivaatuseks kvantarvutil umbes $2^{b/2}$ sammu. Sellele algoritmile toetuv Brassard-Høyer-Tappi algoritm [11] leiab ℓ-bitise väljundiga räsifunktsiooni kollisiooni umbes $2^{\ell/3}$ sammuga. Kuigi võib tunduda et, kvantarvutiga vastast arvesse võttes võiks võrdseks lugeda b-bitise võtmega plokkšifri ja $1.5b$-bitise väljundiga räsifunktsiooni turvatasemeid, kusjuures selleks turvatasemeks on „$b/2$ bitti“, siis on see liialt lihtsustav, nagu me ka allpool, tabeli 3 seletuste juures välja toome.

Ameerika standardiorganisatsioon NIST määratles postkvant-krüptograafia standardimiskutses [12] oodatavate konstruktsioonide turvatasemed viisil, mis hoidus plokkšifrite võtmeotsingut ja räsifunktsioonide kollisiooniotsingut omavahel võrdsustamast. Kutses määratleti viis turvataset; konkreetsel turvatasemel oleva algoritmi murdmine pidi nõudma vähemalt sama palju ressursse kui:

1. 128-bitise võtmega plokkšifri võtme leidmine;
2. 256-bitise väljundiga räsifunktsiooni kollisiooni leidmine;
3. 192-bitise võtmega plokkšifri võtme leidmine;
4. 384-bitise võtmega räsifunktsiooni kollisiooni leidmine;
5. 256-bitise võtmega plokkšifri võtme leidmine.[1]

NIST eeldab, et iga järgmine tase on turvalisem kui eelmine. Ressursse võib mõõta viisil, mis on konkreetse ülesande ja ründestsenaariumi jaoks sobiv (näiteks klassikaliste elementaaroperat-

[1] Standardimiskutses [12] on toodud NISTi selleaegsed hinnangud AESi võtme leidmiseks või SHA3 kollisiooni leidmiseks vajalike klassikaliste ja kvantlülituste suurusele. Hinnang on antud ka SHA3-512 jaoks, kuid sellele vastavat „kuuendat turvataset“ ei ole määratletud.

sioonide arv, või kvantlülituse suurus). Nende tasemete intuitiivseks võrdluseks NISTi aruandes 800-57 [9] kirjeldatud „klassikaliste“ turvatasemetega võib öelda, et tasemed 1 ja 2 vastavad klassikalisele 128-bitisele, tasemed 3 ja 4 klassikalisele 192-bitisele ja tase 5 klassikalisele 256-bitisele turvatasemele.

Tabelites 1 ja 2 on toodud praegu kasutusel olevate või lähitulevikus kasutusele tulevate standardsete või kohe standarditud saavate signatuuriskeemide ja avaliku võtmega krüptosüsteemide turvatasemed. Nende skeemide hulka kuuluvad nii klassikaline RSA ja elliptkõverate põhised skeemid kui ka uued võre- ja koodipõhist krüptograafiat kasutavad skeemid. Iga skeemi jaoks toome parempoolses veerus ka välja, kas skeem on kvant-turvaline või ei. Kirjeldame neid kõiki täpsemalt alapeatükis 2.6.

Hetkeolukorras, kus mitu skeemi koos oma variantidega on äsja standarditud või standardiks saamas, võib olla keeruline otsustada, milline skeem vastab paremini ühe või teise rakenduse vajadustele. Seetõttu toome neis tabelites välja ka nende skeemide krüptomaterjali pikkuse ning püüame visandlikult kirjeldada põhiliste operatsioonide jõudlust. Postkvant-turvaliste skeemide jõudluse kirjeldus pärineb OpenQuantumSafe'i mõõtmistest[2] (v.a HQC, mille mõõtmistulemusi nad ei ole avaldanud; need pärinevad HQCd tutvustavalt lehelt[3]). Klassikaliste skeemide jõudlusnumbrid pärinevad strongSwani projekti mõõtmistest[4]. Tabelite 1 ja 2 jõudluse veerud iseloomustavad, mitu protsessoritakti vastav operatsioon (võtmegenereerimine, signatuuri verifitseerimine, signatuuri loomine, krüptimine, dekrüptimine) nõuab. Suurem arv täppe tähendab kiiremat operatsiooni; konkreetselt on kasutatud sümbolitel on järgmised tähendused:

- ●●●: protsessoritaktide arv jääb kümne ja saja tuhande vahele;
- ●●○: protsessoritaktide arv jääb saja tuhande ja ühe miljoni vahele;
- ●●: protsessoritaktide arv jääb ühe ja kümne miljoni vahele;
- ●○: protsessoritaktide arv jääb kümne ja saja miljoni vahele;
- ●: protsessoritaktide arv jääb saja miljoni ja ühe miljardi vahele;
- ○: protsessoritaktide arv ületab üht miljardit.

Toodud hinnangud on spetsiifilised ühele konkreetsele realisatsioonile ja arvutiarhitektuurile. Siiski võiksid nad anda mingi ettekujutuse ühe või teise operatsiooni arvutuslikust keerukusest. Konkreetsematest stsenaariumidest võimaldab saada aimu eBACSi võrdlusanalüüs[5], mis mõõdab paljude eri realisatsioonide jõudlust paljudel eri platvormidel.

Turvatase, krüptograafilise materjali suurus ning algoritmide jõudlus on olulised valikukriteeriumid kasutusele võetavatele krüptograafilistele algoritmidele. Neid kriteeriume on teisigi: mälukasutus, turvalise realiseerimise lihtsus, privaatvõtme lubatud kasutuskordade arv. Mälukasutusest me siin aruandes ei räägi, sest see on liialt sõltuv konkreetsest realisatsioonist. Samuti leidub sageli kompromisse jõudluse ja mälukasutuse vahel. Privaatvõtmeid võib kõigi tabelites 1 ja 2 nimetatud algoritmide juures kasutada vähemalt 2^{64} korda, mida võib põhimõtteliselt lugeda piiramatuks [14] (aga vaata ka alampeatükki 2.6.4.7). Turvaline realiseerimine on kõige lihtsam hästitoetatud krüptoteeki kasutades (ptk. 6). RSA ja ECDSA kõrvalkanalirünnetest, õigemini nende kasutusest protokollide ründamisel oleme me varasemates aruannetes kirjutanud [4, 8]. ML-DSA haavatavus kõrvalkanalirünnete vastu on hetkel alles midagi, millest püütakse täpsemalt aru saada; mõnda tulemust kirjeldame alampeatükis 2.6.4.5. SLH-DSA disain on oma olemuselt

[2] https://openquantumsafe.org/benchmarking/
[3] https://pqc-hqc.org/
[4] https://wiki.strongswan.org/projects/strongswan/wiki/PublicKeySpeed
[5] https://bench.cr.yp.to/

Tabel 1. Signatuuriskeemide turvatasemed

algoritm(id)	võtmepikkus		signatuuri	jõudlus			PQ?
	avalik	privaat-	pikkus	KGen	Ver	Sig	
NIST turvatase 1							
FALCON-512	897 B	1281 B	666 B	●○	●●○	●●○	jah
SLH-DSA-SHA2-128s	32 B	64 B	7856 B	●○	●●○	●	jah
SLH-DSA-SHAKE-128s	32 B	64 B	7856 B	●	●●	●	jah
SLH-DSA-SHA2-128f	32 B	64 B	17088 B	●●○	●●	●○	jah
SLH-DSA-SHAKE-128f	32 B	64 B	17088 B	●●	●●	●○	jah
ECDSA-256 ja Ed25519	33 B	32 B	64 B	●●○	●●○	●●○	ei
RSA-3k	384 B	384 B	384 B	○	●●○	●○	ei
NIST turvatase 2							
ML-DSA-44	1312 B	2560 B	2420 B	●●●	●●●	●●○	jah
NIST turvatase 3							
ML-DSA-65	1952 B	4032 B	3309 B	●●○	●●○	●●○	jah
SLH-DSA-SHA2-192s	48 B	96 B	16224 B	●○	●●	●	jah
SLH-DSA-SHAKE-192s	48 B	96 B	16224 B	●	●●	○	jah
SLH-DSA-SHA2-192f	48 B	96 B	35664 B	●●	●●	●○	jah
SLH-DSA-SHAKE-192f	48 B	96 B	35664 B	●●	●●	●○	jah
ECDSA-384	49 B	48 B	96 B	●●	●●	●●	ei
Ed448[a]	57 B	57 B	114 B	●●○	●●○	●●○	ei
RSA-8k	1024 B	1024 B	1024 B	○	●●○	●	ei
NIST turvatase 5							
ML-DSA-87	2592 B	4896 B	4627 B	●●○	●●○	●●○	jah
FALCON-1024	1793 B	2305 B	1280 B	●○	●●○	●●	jah
SLH-DSA-SHA2-256s	64 B	128 B	29792 B	●○	●●	●	jah
SLH-DSA-SHAKE-256s	64 B	128 B	29792 B	●	●●	○	jah
SLH-DSA-SHA2-256f	64 B	128 B	49856 B	●●	●●	●○	jah
SLH-DSA-SHAKE-256f	64 B	128 B	49856 B	●●	●●	●	jah
ECDSA-521	66 B	66 B	132 B	●●	●●	●●	ei

[a] Disaini eesmärk on, et murdmine nõuaks vähemalt sama palju ressursse kui 224-bitise võtmega plokkšifri võtme leidmine [13].

Tabel 2. Võtmekapseldusmehhanismide ja avaliku võtmega krüptosüsteemide turvatasemed

algoritm(id)	võtmepikkus		krüptoteksti	jõudlus			PQ?
	avalik	privaat-	pikkus	KGen	Enc	Dec	
NIST turvatase 1							
ML-KEM-512	800 B	1632 B	768 B	●●●	●●●	●●●	jah
HQC-128	2249 B	2305 B	4433 B	●●●	●●○	●●○	jah
FrodoKEM-640-AES	9616 B	19888 B	9720 B	●●○	●●	●●○	jah
FrodoKEM-640-SHAKE	9616 B	19888 B	9720 B	●●	●●	●●	jah
ECDH-256	33 B	32 B	33 B	●●○		●●○	ei
NIST turvatase 3							
ML-KEM-768	1184 B	2400 B	1088 B	●●●	●●●	●●●	jah
HQC-192	4522 B	4586 B	8978 B	●●○	●●○	●●○	jah
FrodoKEM-976-AES	15632 B	31296 B	15744 B	●●	●●	●●	jah
FrodoKEM-976-SHAKE	15632 B	31296 B	15744 B	●●	●●	●●	jah
ECDH-384	49 B	48 B	49 B	●●		●●	ei
NIST turvatase 5							
ML-KEM-1024	1568 B	3168 B	1568 B	●●●	●●●	●●●	jah
HQC-256	7245 B	7317 B	14421 B	●●○	●●○	●●	jah
FrodoKEM-1344-AES	21520 B	43088 B	21632 B	●●	●●	●●	jah
FrodoKEM-1344-SHAKE	21520 B	43088 B	21632 B	●○	●○	●○	jah
ECDH-521	66 B	66 B	66 B	●●		●●	ei

kõrvalkanalikindlam, juhul kui kasutusel oleva räsifunktsiooni realisatsioonis ei ole kõrvalkanaleid.

Tabelis 3 toome välja sümmeetriliste krüptoskeemide (plokkšifrid ja räsifunktsioonid, vt alapeatükke 2.3 ja 2.5) turvatasemed. Nende jõudlust me siinkohal kirjeldada ei püüa: need algoritmid peaksid olema piisavalt kiired igal pool kasutamiseks, välja arvatud ehk mõnel üliväikesel platvormil. Kõik tabelis 3 toodud algoritmid on teadaolevalt turvalised ka kvantarvutite vastu, nii nagu NISTi turvatasemed seda kirjeldavad.

Leiame, et algoritmid, mille NISTi turvatase on vähemalt 1, on praegusel hetkel ja ka nähtavas tulevikus piisavalt turvalised. Küll aga tuleks hakata loobuma algoritmidest, mis ei ole kvantturvalised (veerg „PQ?“ tabelites 1 ja 2). Soovitame uutes arendustes kõikjal kus võimalik kasutada postkvant-turvalisi algoritme. Võimatu võib see olla juhul, kui rakendus nõuab keeruliste omadustega krüptoprimitiive, näiteks mingite täiendavate omadustega nullteadmustõestusi. Kui krüptoprimitiividest kasutatakse ainult signeerimist ja krüptimist, siis leiab sobiva algoritmi siintoodud tabelitest. Sarnaselt Võrgu- ja infoturbe koostöörühma (*Network and Information Security (NIS) Cooperation Group*) soovitustele [16] soovitame ka meie kvantkaitsetute algoritmide kasutamisest järk-järgult ja ettevaatlikult loobuda:

Tabel 3. Sümmeetriliste krüptoalgoritmide turvatasemed

NIST turvatase	Plokkšifrid	Räsifunktsioonid	ECCG ACM v2 [15]
1	AES-128	SHA-256, SHA3-256, SHAKE128[a]	ei
2	AES-192		ei
3		SHA-384, SHA3-384	jah
4	AES-256		jah
5		SHA-512, SHA3-512, SHAKE256[a]	jah

[a] SHAKE-funktsiooni väljundi väikese pikkuse korral võib turvatase olla madalam. Vaata ptk. 2.3.2

- pärast 2035. aastat mitte enam kasutada kvantkaitsetuid algoritme;
- pärast 2030. aastat mitte enam kasutada kvantkaitsetuid algoritme kõrge riskiga kasutusmallides, mille iseloomustuse võib leida allikatest [17, 18];
- postkvant-turvalisi algoritme kasutada hübriidrežiimis (ptk 2.7).

Meie soovitused ei puuduta sümmeetrilisi krüptoalgoritme. See võib näida üllatavana, sest Groveri algoritm võiks vähendada võtmeruumi täielikuks läbivaatuseks vajaliku töö hulka. Kuid Groveri algoritm teeb täielikku läbivaatust teisiti[6] kui need praegu kasutatavad viisid, mis on viinud meie soovitusteni kasutada vähemalt 128-bitise turvatasemega algoritme.

Nagu tabel 3 näitab, erineb meie hinnang Euroopa küberturvalisuse sertifitseerimise rühma (*The European Cybersecurity Certification Group, ECCG*) hinnangutest, mis käesoleva aruande avaldamise ajal kehtivad [15, ptk. 2.1]: ECCG paneb suuremat rõhku Groveri algoritmi väiksemale sammude arvule. Seda tuleb silmas pidada, kui kooskõla ECCG hinnangutega on oluline. Samas NIST ei piira AES-128 ja SHA-256 kasutust juhul, kui ründajal võib kasutada olla kvantarvuti [9, ptk. 5.6.1]. Samuti on käesoleva aruande avaldamise ajal kommenteerimiseks avatud ECCG hinnangute uus versioon [19], mis samuti AES-128 ja SHA-256 kasutust ei piira, kuid möönab, et kvantarvutiga vastase vastu võib nende turvatase olla väiksem kui 128 bitti.

Meie praeguste teadmiste kohaselt kehtivad kõik tabelites 1–3 toodud turvatasemete hinnangud kvantkindlate algoritmide jaoks *tähtajatult*: ei ole ette näha teoreetilisi edusamme arvatavalt raskete võrepõhiste või räsipõhiste arvutusülesannete lahendamisel, mis neid hinnanguid madaldama sunniks. Kvantkaitsetute algoritmide jaoks kehtivad ülalnimetatud tähtajad.

2.2 Krüptoprimitiivide rakendamisest

Aruande eelmises versioonis [8] selgitasime, et standardseid krüptoprimitiive kasutatakse nii standardsetes kui ka rakendusespetsiifilistes protokollides. Esimene kasutusviis tavaliselt probleeme ei valmista, sest kui probleemid ilmnevad, siis puudutavad nad korraga paljusid kasutajaid, mis tähendab, et ka probleemide lahendamisega tegelevad paljud võimekad organisatsioonid. Teisel juhul on aga oht, et hoolimata krüptoalgoritmi sobivast turvatasemest ning korralikult realiseeritud algoritmist ja võtmehaldusest ei pruugi protokoll saavutada rakendaja soovitud turvaomadusi. Seetõttu tuleks krüptoprimitiivide rakendamisel väljaspool standardprotokolle või standardseid vorminguid plaanitavale tööle küsida pädeva krüptograafi või krüptograafiliste protokollide spetsialisti hinnangut: neil on olemas meetodid nõrkuste tuvastamiseks.

[6] https://words.filippo.io/128-bits/

2.3 Krüptograafilised räsifunktsioonid

Räsifunktsioonid (*hash function, hash algorithm, digest function*) on funktsioonid, mis võtavad oma sisendiks vaid sõnumi ning väljastavad (tavaliselt fikseeritud pikkusega) sõnumilühendi ehk räsi.

Krüptograafilise räsifunktsiooni kohustuslikeks omadusteks on kollisioonikindlus (*collision resistance*), mittepööratavus (*pre-image resistance*) ning lisaoriginaalikindlus (*second pre-image resistance*) – Rogaway ja Shrimpton seletavad need mõisted täpsemalt lahti krüptograafiliste räsifunktsioonide aluseid tutvustavas artiklis [20].

Tavaliselt on arvutuskiirus krüptograafilise räsifunktsiooni puhul omaette väärtus. Räsifunktsioonidel on aga rakendus, kus on vajalik, et räsifunktsiooni arvutamise algoritm oleks ressursinõudlik. Selleks rakenduseks on lõppkasutaja-paroolide räsimine. Krüptograafilised räsifunktsioonid nende paroolide räsimiseks ei sobi, parooliräsimiseks on kasutusel eraldi funktsioonid, mida kirjeldame alampeatükis 2.4.

2.3.1 SHA-2

2001. aastal avaldas USA Riikliku Julgeoleku Amet (NSA) SHA-2 esimese spetsifikatsiooni. Praegu spetsifitseerib funktsiooni SHA-2 standard FIPS 180-4 [21]. SHA-2 on üks võimalikest räsifunktsioonidest, mille kasutamine on standarditud räsipõhise signatuuriskeemi SLH-DSA (ptk 2.6.4.7) jaoks.

SHA-2 puhul on standarditud funktsioonid SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224 ning SHA-512/256. Kõigis neis nimedes näitab (viimane) kolmekohaline arv väljundi pikkust bittides. Meie praeguste teadmiste kohaselt on SHA-2 variantide turvatase võrdne selle teoreetilise maksimumiga: n-bitise väljundi korral n bitti lisaoriginaalikindluse ja $n/2$ bitti kollisioonikindluse jaoks (vt ka tabel 3). Kuna kasutada soovitatakse ainult algoritme, mis pakuvad vähemalt 128-bitist turvalisust, siis on SHA-224 ja SHA-512/224 kasutamine ebasoovitav rakendustes, kus kollisioonikindlus on oluline. Me soovitame SHA-224 ja SHA-512/224 algoritme kõikjal vältida, sest nii mõnigi kord on raske aru saada, kas rakenduse turvalisus põhineb räsifunktsiooni kollisioonikindlusel või mõnel lihtsamini saavutataval omadusel [22]. Teadustöö SHA-2 krüptoanalüüsi alal on viimastel aastatel aktiveerunud, uusi (teoreetilisi) tulemusi on saadud nii kollisioonikindluse [23, 24, 25, 26] kui ka lisaoriginaalikindluse [27] vallas.

2.3.2 SHA-3

2015. aastal avaldas NIST täiendusena algoritmidele SHA-1 ja SHA-2 räsialgoritmi SHA-3 standardi [28]. Standard pakub nelja standardset fikseeritud väljundipikkusega räsifunktsiooni – SHA3-224, SHA3-256, SHA3-348 ja SHA3-512. Viimane arv funktsiooni nimes näitab väljundi pikkust bittides ning pikkused on valitud nii, et nad langeksid kokku SHA-2 perekonna funktsioonide väljundipikkustega. Sarnaselt SHA-2 perekonnale soovitame ka siin SHA3-224 algoritmi mitte kasutada.

SHA-3 perekonda kuuluvad ka muutuva (piiramatu) väljundi pikkusega räsifunktsioonid SHAKE128 ja SHAKE256 – neid saab kasutada näiteks juhuarvugeneraatorina. Muuhulgas kasutatakse neid ML-DSA signatuuriskeemi (ptk 2.6.4.5) osana ja nende kasutamine on standarditud SLH-DSA (ptk 2.6.4.7) jaoks.

SHA-3 perekonna spetsifikatsioon [28] ei sea piiranguid SHAKE-funktsioonide väljundi pikkusele, see võib olla täpselt nii lühike või pikk nagu rakendus vajab. Lühikese väljundi korral hakkab SHAKE'i kui räsifunktsiooni turvataset piirama väljundi pikkus: kui see on b-bitine, siis saab

räsifunktsioon pakkuda ülimalt $b/2$ bitti turvalisust[7]. Kui väljund on piisavalt pikk, siis on SHAKE128 disainitud pakkuma 128 bitti turvalisust (kui $b \geq 256$) ja SHAKE256 512 bitti turvalisust (kui $b \geq 512$).

SHA-3 ja SHAKE krüptoalgoritme, ennekõike nende aluseks olevat *käsnakonstruktsiooni*, kirjeldasime põhjalikumalt 2016. aasta krüptoalgoritmide elutsükli uuringus [4].

2.4 Parooliräsimise algoritmid

Turvaintsidentide statistika[8] näitab selgelt, et viimasel viiel aastal on toimunud erinevatest teenustest ja infosüsteemidest vähemalt paarsada infoleket, millel on järgmine muster.

- Infosüsteemis kaitstakse lõppkasutajate paroole liiga nõrgalt (paroole hoitakse kas avatekstina või räsituna mõne lihtsa meetodiga – tavapärane on standardse räsifunktsiooni ühekordne rakendamine).
- Sissetungija varastab kasutajate andmebaasi, mis sisaldab kasutajatunnuseid (tüüpiliselt e-posti aadress) ja paroole või parooliräsisid ning nõrkade parooliräside korral arvutab neist jõurünnetega paroolide väärtused[9].
- Murdunud paroolide abil saab teenuses kasutajakontosid üle võtta ning kuivõrd inimesed kasutavad sama parooli väga tihti paljudes teenustes, saab neid paroole tarvitada kasutaja kontode ülevõtmiseks teisteski teenustes.

Selliste rünnete vältimiseks on mõistlik rakendada parooliräsimise algoritme, mis kasutavad räsi genereerimiseks plokkšifreid ja räsifunktsioone sellisel viisil, et lõplik algoritm oleks sihipäraselt aeglane. Jõurünnete muutuvad seetõttu teostamatuks, kuid ühekordsel algoritmi rakendamisel probleeme ei teki.

Parooliräsimise algoritmid kuuluvad laiemasse võtmetuletusfunktsioonide klassi – need on sellised algoritmid, mille eesmärk on deterministlikult genereerida etteantud sisendist raskesti ennustatav baidijada. Üldiselt ei pea võtmetuletusfunktsioonid olema aeglased ning neil leidub peale parooliräsimise veel teisigi rakendusi, nagu Diffie-Hellmani võtmevahetuse tulemusena saadud ühisest väärtusest sümmeetrilise võtme genereerimine. Üks populaarsemaid üldkasutuslikke võtmetuletusfunktsioone on HKDF, mis tugineb HMACil ning mida parooliräsimiseks kasutada ei soovitata [29].

Paroolide korralik räsimine võimaldab rakendada ja/või efektiivsemaks muuta NISTi paroolide kohta antud nõudeid ja soovitusi [30, ptk. 3.1.1.2]:

- paroolide regulaarset vahetamist ei tohiks nõuda;
- kui paroolidele on kehtestatud maksimumpikkus, siis peaks see olema vähemalt 64 tähemärki;
- paroolidele ei tohiks kehtestada muid nõudeid peale miinimumpikkuse (näiteks nõue kasutada suur- ja väiketähti segamini ja muud sellised piirangud);
- kasutaja valitud paroole peaks võrdlema sõnastike või reeglistikega, et hoida ära levinud või kompromiteeritud paroolide kasutamist.

[7] See turvatase on maksimaalne, mida räsifunktsioon suudab pakkuda *kollisioonikindluse* omadust rikkuda püüdvate rünnete vastu. *Mittepööratavuse* ja *lisaoriginaalikindluse* rikkumise vastu võib funktsioon pakkuda kuni b bitti turvalisust. Alampeatükis 2.3.1 toodud põhjusel soovitame me aga rakenduste turvaanalüüsides seda erisust mitte teha.

[8] https://haveibeenpwned.com/PwnedWebsites

[9] https://hashcat.net/hashcat/

2.4.1 Argon2

Tänapäeval on enim soovitatud parooliräsimise algoritm Argon2, mis võitis aastal 2015 Password Hashing Competition[10] võistluse. 2021. aastal valmis Argon2-st informatiivne RFC 9106 [31] ning seda soovitab ka OWASP[11].

Argon2 on tegelikult algoritmide perekond, kus iga üksiku algoritmi omadused on pisut erinevad:

Argon2d
– optimeeritud GPU-põhiste rünnete vastu, kuid võib sisaldada kõrvalkanaliründe id lihtsustavaid nõrkusi;

Argon2i
– optimeeritud kõrvalkanalirünnete vastu;

Argon2id
– eelmise kahe hübriid. RFC 9106 [31] soovitab kasutada Argon2id algoritmi kõigil juhtudel kui ülejäänud variantide eelistamiseks ei ole häid põhjuseid, samuti soovitab seda kasutada BSI [32].

Soovitatud parameetrid Argon2 perekonna algoritmidele on toodud RFC 9106 peatükis 4 [31].

NIST on nimeliselt soovitanud kasutada vaid PBKDF2 [33], kuid seda soovitust sisaldav dokument on läbivaatamisel[12] ning selle käigus on otsustatud lisada dokumenti uusi soovituslikke algoritme. Teiste levinumate parooliräsimise algoritmide kohta on võimalik rohkem infot saada 2023. aasta aruandest [8].

2.5 Sümmeetrilise võtmega primitiivid

Sümmeetrilise võtmega krüptoprimitiivid on sellised primitiivid, mille puhul kõigi operatsioonide jaoks kasutatakse sama võtmematerjali. Sümmeetrilised krüptoprimitiivid jagunevad kolme suurde kategooriasse: plokkšifrid, jadašifrid ning räsipõhised konstruktsioonid.

2.5.1 Plokkšifrid

Plokkšiffer on olemuselt n-bitiste plokkide üks-ühene teisendus. Konkreetse teisenduse määrab k-bitine võti. Plokipikkus n ja võtmepikkus k ei pea olema võrdsed. Selle aruande kirjutamise ajal on ainus laialt kasutatav standardne plokkšifrialgoritm AES (*Advanced Encryption Standard*) [34].

Plokkšifreid ei kasutata kunagi nende puhtal kujul, vaid mingis konkreetses režiimis (AES-GCM, AES-CBC) või mingi teise konstruktsiooni (näiteks jadašifri või juhuarvude generaatori) ehituselemendina.

2.5.2 Jadašifrid

Jadašiffer võimaldab suvalise L-bitise avateksti krüptimist, genereerides k-bitisest võtmest L-bitise bitijada, mis kombineeritakse avatekstiga bitthaaval XOR operatsiooni abil (ehk *modulo* 2 liitmistehtega). Üht ja sama võtit mitu korda kasutades tuleb hea seista selle eest, et genereeritav bitijada oleks igakord erinev (ja eelmistel kordadel genereeritud bitijadadest sõltumatu). Selle jaoks võtab bitijada genereerimise algoritm lisaks võtmele argumendiks veel *nonsi*, mis peab iga

[10] https://www.password-hashing.net/
[11] https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html
[12] https://csrc.nist.gov/News/2023/decision-to-revise-nist-sp-800-132

kord olema erinev. Nonss ei pea olema aga sõltumatu eelmistest nonssidest; tegemist võib olla näiteks loenduriga, mida igal krüptimisel suurendatakse. Samuti võib nonss olla avalik; kui teate vastuvõtja ei tea nonsi väärtust, siis võib selle lisada krüptotekstile.

Jadašifrite saamiseks on mitmeid viise: jadašifrina kasutamiseks projekteeritud algoritmid, muutuva pikkusega räsifunktsioonid (SHAKE) või sobivas režiimis plokkšifrid (AES-CTR).

Jadašifrid pakuvad selle aruande kontekstis huvi ennekõike oma kasutuse tõttu TLSi šifrikomplektides, ChaCha20 algoritmi näol. Üldisemalt on neil eelis plokkšifrite ees sellistes rakendustes, mis nõuavad ettemääramata suurusega andmesegmentide krüptimist ja edastamist piiratud oludes – näiteks raadiosides, kus plokkšifri kasutamine šifri ploki suurusest väiksemate segmentide edastamisel oleks vajaliku täidise tõttu raiskavam.

2.5.2.1 ChaCha20

ChaCha on Daniel J. Bernsteini 2008. aastal publitseeritud [35] jadašifrite perekond, number variandi ChaCha20 nimes tähistab iteratsioonide arvu (vähema kui 20 iteratsiooniga ChaCha variandid ei ole piisava turvatasemega).

ChaCha20 on kasutusel TLS 1.2 ja TLS 1.3 jaoks soovitatud šifrikomplektides. ChaCha20 kasutatakse nendes komplektides koos Poly1305 autentimiskoodiga [36].

Hinnatakse, et ChaCha20 on ilma spetsiifilise AESi toeta riistvaral kolm korda kiirem kui AES [36]. Seetõttu on tema kasutusvaldkond TLS-is ennekõike väheste ressurssidega arvutisüsteemides (esemevõrgu seadmed, sardsüsteemid).

ChaCha20 võtmepikkus on 256 bitti ning selle turvatase on sama kui Salsa20/20 jadašifril, millel ChaCha põhineb – 256 bitti [37].

2.5.3 Räsipõhised sümmeetrilised konstruktsioonid

Räsipõhiste konstruktsioonide peamine näide on sõnumiautentimiskoodid, mille levinuimaks konstruktsiooniks on HMAC (*hash-based message authentication code*) [38]. Standardsete HMAC-koodide turvatase sõltub ennekõike kasutatavast võtmepikkusest. Euroopa küberturvalisuse sertifitseerimise rühm (ECCG) soovitab kasutada HMACi võtmepikkusega vähemalt 125 bitti [15, ptk. 3.3]. Tüüpiliselt on rakendustes HMAC-võtme pikkus seotud kasutatava räsifunktsiooni väljundipikkusega.

Üldjuhul soovitatakse tänapäeval kasutada sõnumite autentimiseks AEAD (Authenticated Encryption with Associated Data) [39] lähenemist, kus sõnumiautentimiskood ja krüptimisalgoritm on kombineeritud ühte terviklikku protokolli ning kasutaja ei pea muretsema, kuidas algoritme turvaliselt ja efektiivselt kombineerida. Sellist lähenemist kasutab ka eelmises alapeatükis mainitud ChaCha20/Poly1305 [36], kus Poly1305 autentimiskoodi eraldiseisvalt kasutada oleks ebapraktiline, sest iga võtit saab turvaliselt kasutada vaid korra.

2.6 Asümmeetrilised primitiivid

Asümmeetrilised primitiivid on sellised, kus eri operatsioonide jaoks kasutatakse erinevat võtmematerjali. Asümmeetrilist primitiivi realiseeriva algoritmistiku kasutus algab sellest, et üks süsteemi pooltest kasutab võtmegenereerimisalgoritmi, et luua võtmepaar – avalik võti ja privaatvõti. See pool avalikustab avaliku võtme, aga privaatvõtme hoiab ainult enda teada. Avalikust võtmest sellele vastava privaatvõtme leidmine on eeldatavasti arvutuslikult raske.

Põhilised asümmeetrilised primitiivid on avaliku võtmega krüptoskeemid ja signatuuriskeemid. Avaliku võtmega krüptosüsteemides kasutatakse avalikku võtit sõnumi krüptimiseks ning privaatvõtit dekrüptimiseks. Signatuuriskeemides kasutatakse privaatvõtit mingile sõnumile signatuuri loomiseks ning avalikku võtit selle signatuuri kontrollimiseks. Näeme, et krüptimine ja signatuurikontroll on tegevused, mida saavad teha kõik, kes teavad vastavat avalikku võtit. Seevastu tundlikud tegevused – dekrüptimine ja signeerimine – on võimalikud ainult võtmepaari loojale.

2.6.1 RSA

RSA [40] on asümmeetriline krüptograafiline algoritm, mida saab kasutada nii signatuuriskeemina kui ka krüptimisskeemina. RSA turvalisus põhineb täisarvude tegurdamise probleemi eeldataval keerukusel. Soovitatavad võtmepikkused on toodud tabelis 1; mingi konkreetse võtmepikkusega RSA signeerimise ja krüptimise turvatasemed on võrdsed.

Eestis on RSA-signeerimine Smart-ID autentimis- ja allkirjastamisteenuse aluseks. Tuletame meelde, et Smart-ID ei ole puhas RSA, vaid signeerimisprotokolli mõlemal poolel (kasutaja telefon ning Smart-ID server) on oma isiklik RSA-võti, mis kombineeritakse üheks võtmeks; samuti kombineeritakse mõlema poole antud signatuurid [41]. RSA-signeerimist kasutatakse ühtlasi X-tees (signeerimiseks ja autentimiseks), ajatemplites ja kehtivuskinnituse vastustes. RSA-krüptimist (koos OAEP täidisega) kasutatakse CDOC2s sümmeetrilise võtme krüptimiseks.

RSAst ja rünnetest tema vastu oleme kirjutanud aruande varasemates versioonides [1, ptk 2.2.1][2, ptk 2.2.1][4, ptk 3].

2.6.2 ElGamal krüptoskeem

ElGamal [42] on avaliku võtmega krüptimisskeem, mille turvalisus põhineb Diffie ja Hellmani otsustusprobleemi (DDH probleem) eeldataval keerukusel. Skeemi on võimalik kasutada igas rühmas, kus DDH probleem on eeldatavalt keeruline, sealhulgas elliptkõverate punktidel defineeritud rühmadel. Eestis kasutatakse ElGamali krüptimisskeemi internetihääletusel häälte krüptimiseks [43], kasutatud on seda nii jäägiklassirühmades kui ka elliptkõverarühmades.

2.6.3 Elliptkõveratel põhinevad süsteemid

Ka elliptkrüptograafia põhineb suuresti Diffie ja Hellmani otsustusprobleemi eeldataval keerukusel kõverapunktidest moodustatud rühmades. Võrreldes aruande eelmise versiooniga [8, ptk 2.6.3] ei ole neis süsteemides olulisi uuendusi toimunud, seega tuletame siin ainult meelde, milliste skeemidega on tegemist.

Iga elliptkõverarühma kirjeldab mittetriviaalne komplekt parameetreid: baaskorpus, kõverat määrav võrrand, rühma genereeriv kõvera punkt (mis määrab ka rühma suuruse). Detailsemalt kirjeldame neid parameetreid selle aruande varasemas versioonis [3, ptk. 3]. Kuna nende parameetrite valik on krüptograafiliselt oluline ja samas mittetriviaalne, siis on kasutusel teatud standardsed elliptkõverad (ja rühmad). Tabel 4 loetleb kõveraid (ühes viidetega nende kirjeldusele), mida soovitavad NIST, IETF ja ECCG [15].

Tabel 4. Standardsed elliptkõverad

Kõvera nimi	Standard	Kuju	Märkused
NIST P-256 NIST P-384 NIST P-521	NIST FIPS186-4 [44]	Weierstraß	P-256 ja P-384 kasutatakse Eesti Mobiil-ID ja ID-kaardi võtmepaaride jaoks
Curve25519 Edwards25519 Curve448 Edwards448	RFC 7748 [45]	Montgomery Edwards Montgomery Edwards	omavahel algebraliselt seotud omavahel algebraliselt seotud
Brainpool P256r1 Brainpool P348r1 Brainpool P512r1	RFC 5639 [46]	Weierstraß	

2.6.3.1 ECDH

ECDH (*Elliptic Curve Diffie-Hellman*) on elliptkõveratel põhinev võtmevahetusprotokoll, mis lubab kahel poolel turvaliselt kokku leppida ühise võtme. Tänu elliptkõverate eeldatavatele turvaomadustele on ECDH-võtmed ja teated sama turvataseme juures oluliselt lühemad kui traditsioonilises Diffie-Hellmani protokollis [47]. ECDH on laialdaselt kasutusel ja tänapäevases krüptograafias üks peamisi võtmevahetuse meetodeid. See on osa paljudest protokollidest, näiteks on see kasutusel TLS 1.3 ning Signali protokollides ja muudes läbivalt turvalistes süsteemides.

2.6.3.2 ECDSA ja EdDSA

ECDSA ja EdDSA on elliptkõveratel põhinevad signatuuriskeemid. Nende skeemide turvalisus põhineb diskreetse logaritmi probleemi eeldataval keerukusel elliptkõverarühmades.

EdDSA on deterministlik signatuuriskeem, mis töötab Edwardsi kõveratel. Deterministlikkus tähendab, et ühe ja sama sõnumi signeerimiseks kasutatakse alati üht ja sama juhuslikku väärtust (kui on kasutatud üht ja sama privaatvõtit). See vähendab riski, et signeermiseks kasutatakse nõrka juhuslikkust – see on eriti oluline, kuna ECDSA skeemi puhul on nõrk juhuslikkus üks levinumaid skeemi murdumise põhjuseid. Seetõttu pakub EdDSA paremat turvalisust kõrvalkanalirünnete (nt juhuslikkuse vigade) vastu [48].

ECDSA on laialt kasutatav tõenäosuslik signatuuriskeem (st kasutab juhuslikku väärtust iga signatuuri loomisel). NIST-i digitaalsignatuuristandardi viimane versioon [13] (erinevalt eelviimasest [44]) spetsifitseerib ka deterministliku ECDSA signeerimisalgoritmi. ECDSA skeem on kasutusel Eesti ID-kaardis kasutaja autentimiseks ja dokumentide digiallkirjastamiseks.

2.6.4 Kvantturvalised skeemid

Selle aruande eelmise versiooni ilmumise järel on standarditud mitmeid avaliku võtmega krüptograafia skeeme, mis on eeldatavalt turvalised nii kvant- kui ka klassikalise vastase

olemasolul. Nende skeemide hulgas on võrepõhiseid, koodipõhiseid ja räsipõhiseid skeeme, mida järgnevalt kirjeldame.

2.6.4.1 ML-KEM

ML-KEM [49] (*Module-Lattice-Based Key Encapsulation Mechanism*, varasema nimega Crystals-Kyber, vaata ka selle aruande eelmist versiooni [8, ptk 2.8.1]) on (struktureeritud) võrepõhine võtmekapseldusmehhanism (st funktsionaalsuselt sarnane asümmeetrilisele krüptimisele), mille NIST on standardinud. Selle turvalisus põhineb moodulis vigadega õppimise (*module Learning with Errors*) probleemi keerukusel. ML-KEM on lisatud ka Euroopa küberturvalisuse sertifitseerimise rühma (ECCG) avaldatud juhisesse „Agreed Cryptographic Mechanisms“ [15]. Skeemi parameetrid on esitatud tabelis 2.

2.6.4.2 HQC

HQC [50] (*Hamming Quasi-Cyclic*) on koodipõhine võtmekapseldusmehhanism, mida NIST on standardimas. Skeemi turvalisus põhineb paarsusega QCSD (*Quantum Charge-Coupled Devices with parity*) probleemil. Kvaasitsüklilise struktuuri tõttu on HQC avaliku võtme ja krüptogrammi suurused väiksemad kui teistes koodipõhistes skeemides, kuid suuremad kui struktureeritud võrepõhistes skeemides. Skeemi parameetrid on esitatud tablis 2.

2.6.4.3 FrodoKEM

FrodoKEM [51] on võrepõhine võtmekapseldusmehhanism, mis põhineb vigadega õppimise (*learning with errors*, LWE) probleemil. FrodoKEM ei kuulu NISTi standarditud skeemide hulka, küll aga standardib teda ISO [52]. Kuna FrodoKEM põhineb struktureerimata võredel, peetakse seda konservatiivsemaks skeemiks kui HQC. Skeemi struktureerimatuse tõttu on aga tema võtmed ja krüptogrammid suuremad kui näiteks ML-KEMil. FrodoKEM on lisatud Euroopa küberturvalisuse sertifitseerimise rühma (ECCG) avaldatud juhisesse „Agreed Cryptographic Mechanisms“ [15].

Skeemi parameetrid on toodud tabelis 2. Skeemist on olemas kaks versiooni, mis erinevad sümmeetrilise algoritmi poolest, mida kasutatakse seemnest suurema pseudojuhusliku maatriksi genereerimiseks. Selleks algoritmiks võib olla kas AES või SHAKE. Esimene neist on üldiselt kiirem platvormidel, kus AESi arvutamine on riistvaraliselt kiirendatud; ka tabelis 2 toodud jõudlustulemused pärinevad selliselt platvormilt. Kui AESi riistvaraline kiirendamine puudub, siis võib eeldada, et SHAKE'i kasutav variant on kiirem.

2.6.4.4 Classic McEliece

Classic McEliece [53] on koodipõhine võtmekapseldusmehhanism. Kuna skeemi esialgne versioon pakuti välja 1978. aastal, peetakse seda konservatiivseks ja väga põhjalikult analüüsitud skeemiks. Matemaatilise struktuuri tõttu on Classic McEliece-skeemil suhteliselt suured avalikud võtmed, kuid väga lühikesed krüptogrammid. Classic McEliece ei kuulu NISTi standarditud skeemide hulka, kuid selle on standardinud ISO [52]. Tabelis 2 me teda praegu ei kajasta, sest ei pea tõenäoliseks tema lähiajal praktikas rakendamist.

2.6.4.5 ML-DSA

ML-DSA [54] (*Module-Lattice-Based Digital Signature Algorithm*, varasema nimega Crystals-Dilithium, vaata ka selle aruande eelmist versiooni [8, ptk 2.8.2]) on NISTi standarditud võrepõhine signatuuriskeem. Skeemi turvalisus põhineb moodulis vigadega õppimise probleemil ja moodulis lühikese täisarvlahendi ülesande probleemi variandi keerukusel. Kuna ML-DSA on efektiivne ja suhteliselt lihtsa teostusega skeem, on NIST valinud selle üldotstarbeliseks signatuuriskeemiks. ML-DSA on lisatud Euroopa küberturvalisuse sertifitseerimise rühma (ECCG) avaldatud juhisesse „Agreed Cryptographic Mechanisms“ [15]. Skeemi parameetrid on esitatud tabelis 1.

ML-DSA kõrvalkanalikindlate realisatsioonide loomist on uuritud; olemas on realisatsioonid, mille tööaeg ei sõltu salajastest andmetest[13]. Kirjeldatud on kõrvalkanaleid, mis ei põhine tööaja mõõtmisel:

- Kui signeerimise realisatsioon järgib täpselt standardis [54] toodud efemeerse saladuse juhuslikust seemnest genereerimise protseduuri, siis võib leiduda voolutarbe mõõtmisel põhinev kõrvalkanal [55], mis selle efemeerse saladuse leiab. Piisavat arvu signatuure ja neile vastavaid efemeerseid saladusi teades on võimalik leida privaatvõti.
- Standard [54] kirjeldab, kuidas privaatvõti enne signeerimist lahti pakkida. See lahtipakkimise protseduur võib sisaldada kõrvalkanalit [56].
- ML-DSA signeerimisel luuakse rohkesti vaheväärtusi, mida hiljem ei kasutata; selliste väärtuste lekkimine võib osutuda kõrvalkanaliks, mille kaudu privaatvõti lekkida võib [57, 58].

2.6.4.6 FN-DSA

FN-DSA [59] (*Fast-Fourier Transform over NTRU-Lattice-Based Digital Signature Algorithm*, varasema nimega Falcon, vaata ka selle aruande eelmist versiooni [8, ptk 2.8.2]) on võrepõhine signatuuriskeem, mida NIST on standardimas. Süsteemi turvalisus põhineb lühikese täisarvlahendi ülesande (*Short Integer Solution*, SIS) probleemil NTRU võredel. Skeemil on standardsete postkvantturvaliste signatuuriskeemide hulgas väikseim avaliku võtme ja allkirja suurus, kuid allkirjastamine ja võtme genereerimine on aeglasem kui ML-DSA skeemi puhul. Skeemi teostus eeldab ujukoma-aritmeetika kasutamist, mis teeb raskemaks skeemi turvalise teostuse, olgugi et see ei ole võimatu [60][14]. Skeemi parameetrid on esitatud tabelis 1.

2.6.4.7 SLH-DSA

SLH-DSA [14] (*Stateless Hash-based Digital Signature Algorithm*, varasema nimega Sphincs+, vaata ka selle aruande eelmist versiooni [8, ptk 2.8.2]) on NISTi standarditud olekuta räsipõhine signatuuriskeem. Skeemi turvalisus põhineb kasutatava räsifunktsiooni kollisioonikindlusel; seda turvaeeldust võime lugeda konservatiivsemaks kui võrepõhiste arvutusprobleemide eeldatavat keerukust. SLH-DSA skeemil on lühike avalik võti, kuid pikad signatuurid. Selle tõttu sobib skeem kasutuseks erijuhtudel, kus on vaja konservatiivsemat turvalisust ja signatuuri suurus ei ole takistus. SLH-DSA on lisatud Euroopa küberturvalisuse sertifitseerimise rühma (ECCG) avaldatud juhisesse „Agreed Cryptographic Mechanisms“ [15].

[13] https://blog.trailofbits.com/2025/11/14/how-we-avoided-side-channels-in-our-new-post-quantum-go-cryptography-libraries/
[14] https://keymaterial.net/2026/05/13/so-you-want-to-deploy-fn-dsa/

Lisaks turvatasemele on standardsel SLH-DSAl [14] veel kaks parameetrit. Esimene neist on kasutatav räsifunktsioon, mis võib olla kas SHA2 või SHAKE. Teine neist on kasutatavate räsipuude ja signatuuripuude struktuuri iseloomustav parameetrikomplekt [14, tabel 2], mis võib anda kas suhteliselt väikseid (***s**mall*) signatuure või lubada suhteliselt kiiret (***f**ast*) signeerimist. Skeemi parameetrid on esitatud tabelis 1.

SLH-DSA parameetristikud [14] on valitud arvestusega, et ühe võtmega oleks võimalik anda vähemalt 2^{64} signatuuri. Lisaks on selle aruande avaldamise ajal kommentaare ootamas parameetristikud [61], mis peaksid lubama vähemalt 2^{24} signatuuri andmist ühe võtmega.

Tänu oma disainile on SLH-DSA hästi kaitstud tööaega mõõtvate kõrvalkanalirünnete vastu. Kaitsmaks teda rünnete vastu, mis mõõdavad käitusaegset voolutarvet või loevad üksikuid vahetulemusi, saab kasutada standardseid, ühissalastusel põhinevaid meetodeid. SLH-DSA jaoks võivad need meetodid olla üsna kulukad, seetõttu on uuritud, millal neist osaliselt loobumine veel turvaline võiks olla [62]. Kui ründaja saab aga mõjutada mälus olevate bittide väärtust SLH-DSA signeerimise ajal, siis on tal võimalik muuta valeks SLH-DSA korrektsel kasutamisel kehtiv väide, et ühekordseid võtmeid [8, ptk 2.8.2] kasutatakse suure tõenäosusega ainult üks kord. Sellise ründe käigus loodud signatuurid sisaldavad informatsiooni, mida kasutades on ründajal endal võimalik uusi signatuure luua [63]. Mälus olevate bittide väärtuse mõjutamiseks saab kasutada *Rowhammer*-rünnet, vt. alampeatükki 3.1.2.

2.6.4.8 Olekuga räsifunktsioonidel põhinevad signatuuriskeemid

XMSS ja LMS on olekuga räsifunktsioonidel põhinevad signatuuriskeemid, mille NIST on lisanud soovitatud skeemide hulka [64]. Sarnaselt SLH-DSA skeemile põhineb nende turvalisus skeemi aluseks oleva räsifunktsiooni kollisioonikindlusel. Olekuga räsifunktsioonidel põhinevad signatuuriskeemid koosnevad mitmest ühekordse signatuuriskeemi võtmepaarist. Seetõttu on oluline jälgida, milliseid võtmepaare on juba kasutatud signeerimiseks. Selleks on vaja hoida meeles mitte ainult võtmepaari, vaid ka skeemi olekut. Olekuga skeeme soovitatakse kasutada ainult olukordades, kus on võimalik tagada turvaline ning veakindel olekuhaldus. LMS ja XMSS on lisatud Euroopa küberturvalisuse sertifitseerimise rühma (ECCG) avaldatud juhisesse „Agreed Cryptographic Mechanisms“ [15].

XMSS ja LMS loovad võtmegenereerimise ajal teatud kahendpuu, mille tippudes on ühekordsed võtmepaarid. Kui selle puu kõrgus on h, siis saab loodud võtmega anda kuni 2^h signatuuri. NIST soovitab LMS-i jaoks parameetrikomplekte, kus h väärtus on kas 5, 10, 15, 20 või 25. XMSS-i jaoks soovitatakse aga parameetrikomplekte, kus h on kas 10, 16, 20, 40 või 60 [64].

2.7 Hübriidrežiim

Paljud organisatsioonid soovitavad üleminekuperioodil kasutada korraga nii klassikalisi kui ka postkvant-turvalisi krüptoalgoritme, nii et kogu skeemi turvalisuseks piisaks neist üheainsa algoritmi turvalisusest. Sarnaselt sümmeetrilise võtmega krüptograafiale (ptk 2.5) tuleb algoritmide kombineerimiseks fikseerida mingi konkreetne kombineerimisviis ehk *režiim*. Selliseid režiime, mis kombineerivad mitut sama krüptograafilist primitiivi realiseerivaid algoritme ja mille turvalisus järeldub vähemalt ühe kombineeritava algoritmi turvalisusest, nimetame *hübriidrežiimideks*. Tuleb välja, et hübriidrežiimid signeerimise ja võtmekapselduse jaoks on üsna erineva keerukusega.

- Signatuuriskeemide lihtne kombinatsioon $\sigma = (\sigma_1, \sigma_2)$ on võltsimatu ründaja poolt valitud sõnumi korral (EUF-CMA), kui üks neist signatuuridest on võltsimatu [65]. Hübriidskeemidelt

võime nõuda ka tugevamaid omadusi, näiteks *non-separability* erinevaid versioone, mis nõuavad, et kaht signeerimisalgoritmi üheskoos rakendades loodud hübriidsignatuurist ei oleks võimalik leida üksiku algoritmiga loodud signatuuri. Teadaolevad kombineerimisviisid ei pruugi olla rakendatavad suvalistele signatuuriskeemidele, vaid nõuavad näiteks, et kombineeritavate skeemide konstruktsioon järgiks Fiat-Shamiri heuristikat [66].

- Võtmekapseldusmehhanismide turvalise hübriidrežiimi valimine on keerulisem kui signatuuriskeemide puhul. Kui kombineerida kahe võtmekapseldusmehhanismi väljundid k_1 ja k_2 võtmetuletusfunktsiooniga lihtsaimal viisil $k = KDF(k_1, k_2)$ (siin KDF on võtmetuletusfunktsioon, *Key Derivation Function*), siis see kombinatsioon ei pruugi olla turvaline valitava krüptogrammiga rünnete (*Chosen-Ciphertext Attack*, CCA) vastu [67]. Kuna NIST soovitab võimalusel alati kasutada valitava krüptogrammiga rünnete vastu turvalist võtmekehtestusmehhanismi, on vaja saladuste kombineerimiseks võtmetuletusfunktsiooni korrektselt kasutada [68].

Enne hübriidrežiimi valimist tuleb tähele panna, et hübriidskeemide struktuur on kahe algoritmi kombineerimise tõttu tavalisest keerulisem. Selle võrra keerulisem on omakorda ka hübriidskeemide teostus, mistõttu vale hübriidrežiimi teostuse või valiku korral võib tekkida madaldusrünnete oht või teised turvanõrkused.

Hübriidrežiimi kasutamisel on järgmised eelised:

- **mitmekihiline kaitse**. Kui süsteemis kasutakse korraga nii klassikalisi kui ka kvantturvalisi krüptoalgoritme, jäävad andmed kaitstuks rünnete eest isegi siis, kui üks neist osutub kaitsetuks (kvantarvuti või mõne muu ründeviisi vastu);
- **regulatiivne ja koostalitlusvõime paindlikkus**. Olemasolevad NISTi standardid ja juhised võimaldavad hübriidrežiimi kasutamist postkvant-krüptograafia ülemineku kontekstis [69]. See aitab sujuvamalt üle minna postkvant-krüptograafiale süsteemides, mis peavad vastama kindlatele standarditele ja regulatsioonidele, kasutades NISTi kinnitatud klassikalist skeemi.

3 Protokollid ja protokollistikud

3.1 TLS

TLS (*Transport Layer Security*) on valdava osa Interneti-liikluse turvamiseks kasutatav protokollistik. Protokollistiku eesmärgiks on realiseerida „turvalise kanali“ abstraktsioon. TLS-protokolli kasutatakse valdavas osas turvalisi kanaleid vajavates rakendustes. Ta on osa HTTPS-protokollist, aga teda kasutatakse ka näiteks DNS-päringute, andmebaasiühenduste, privaatvõrkude ja muu turvamiseks. Selles peatükis toodud soovitused kehtivad kõigi kasutusmallide jaoks. TLS-protokolli viimane versioon on 1.3 [70, 71]. Enne seda oli pikalt kasutusel versioon 1.2 [72], mida kirjeldasime põhjalikult 2016. aasta aruandes [4].

Sarnase nimega ja sarnaste krüptograafiliste protokollidega, kuid üsna erineva eesmärgiga on DTLS (*Datagram TLS*), mille viimane versioon on samuti 1.3 [73]. Selle protokollistiku eesmärgiks on realiseerida „turvalise pakettkommutatsiooni“ abstraktsioon klient-server rakenduste vahel. Siinses aruandes me DTLSil põhjalikumalt ei peatu, olgugi et pakettkommutatsiooni olulisus eri kasutusmallides võiks ajas pigem kasvada.

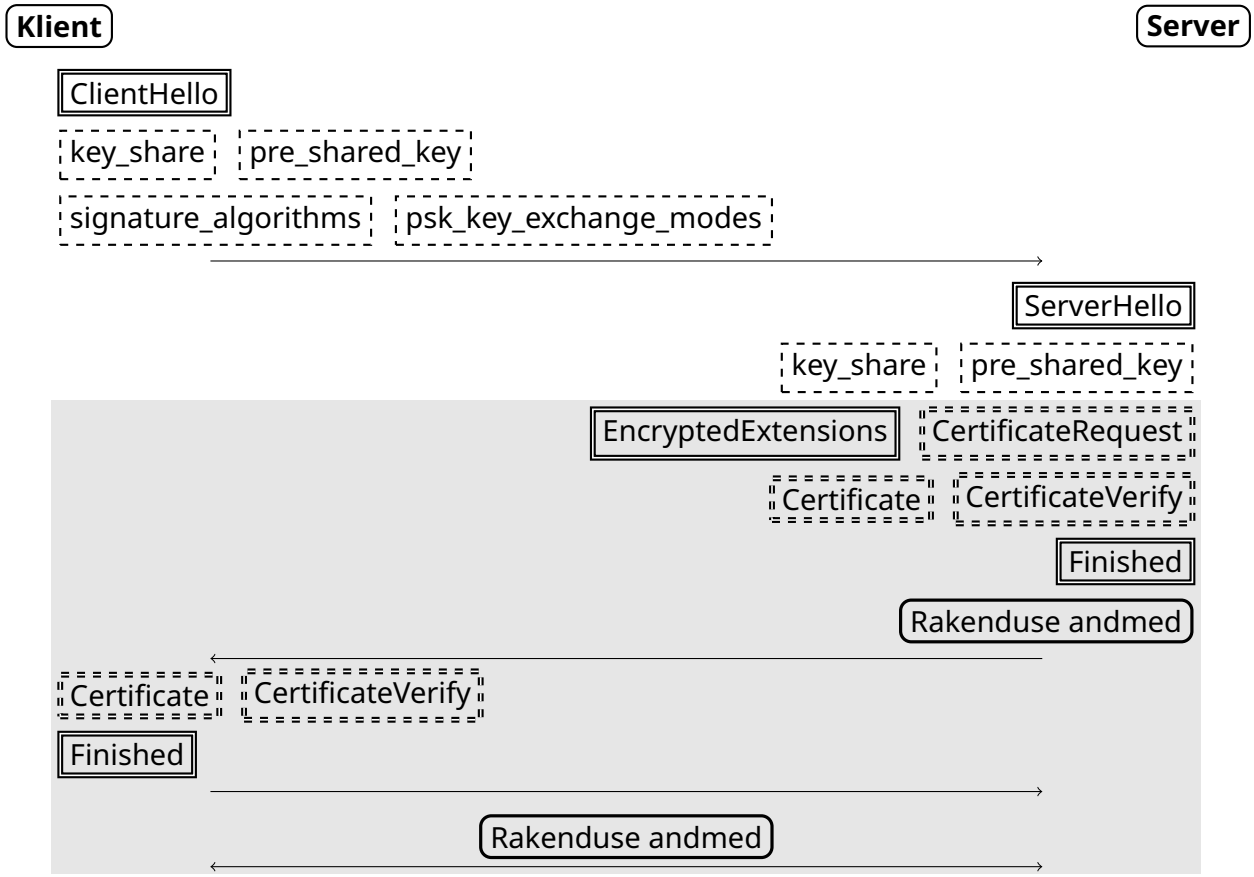
3.1.1 TLS 1.3 kirjeldus

TLS koosneb kahest peamisest protokollist – võtmekehtestusprotokollist ja transpordiprotokollist. Neist esimene kasutab (enamasti) asümmeetrilist krüptograafiat, et kokku leppida sümmeetriline võti, millega kanali sisu kaitstakse. Teine kasutab seda võtit, et sümmeetrilise krüptograafia abil kanali sisu krüptida ja autentida. Võtmevahetuse *peamine variant* põhineb Diffie-Hellmani võtmevahetusel (ptk 2.6.3.1), mille autentimiseks kasutatakse signatuure. Transpordiprotokoll põhineb mõnel AEAD (Authenticated Encryption with Authenticated Data) turvafunktsionaalsusega plokk- või jadašifri töörežiimil. Krüptimisalgoritmi sisendiks on lisaks võtmele k veel avatekst P ja kaasandmed A; väljundiks on krüptotekst C, millest ei ole võimalik avateksti P ilma võtit teadmata tuletada. Võtit k teadvale olemile autendib C ära nii P kui ka A.

TLSi võtmekehtestusprotokolli peamine teadetejada on toodud joonisel 1. Mõlema poole saadetavad teated on loogiliselt jagatud mitmeks „sõnumiks“, mis saadetakse aga üheskoos, nii et võtmevahetuseks piisab kolmest teatest, millest esimese ja viimase saadab klient ja keskmise server. Peale nende teadete vahetamist on kliendil ja serveril kokku lepitud saladus, mida ainult nemad kahekesi teavad.

`ClientHello`-sõnumi oluline osa on kliendi genereeritud juhuslik nonss, mis tagab serverilt saabuvate sõnumite värskuse. Teine oluline osa on nende sümmeetriliste krüptoalgoritmide loend, mida klient transpordiprotokollis toetab; see loend on eelistatuse järjekorras. `ServerHello`-sõnum sisaldab samuti juhuslikku nonssi; lisaks ütleb server seal, millist kliendi poolt toetatud sümmeetrilist krüptoalgoritmi kasutada.

TLSi sõnumitel võivad olla laiendused. Üks oluline laiendus on `key_share`, mida kasutatakse ühise saladuse kokkuleppimiseks. TLS 1.3 spetsifikatsioon [70] nägi ette, et kokkuleppimine toimub ECDH-d (ptk 2.6.3.1) kasutades ning `key_share` laienduse sisu on põhiliselt Diffie-Hellmani efemeerne avalik võti ühes kasutatava elliptkõvera identifikaatoriga. Hiljem, seoses postkvantkrüptograafia kasutuselevõtuga kasutatakse sedasama laiendust ka koos võtmekapseldusmehhanismidega [74]. Kui ühise saladuse kokkuleppimiseks vajalikud teated on vahetatud, siis ülejäänud osa võtmevahetusprotokollist viiakse läbi turvatud kanali sees.



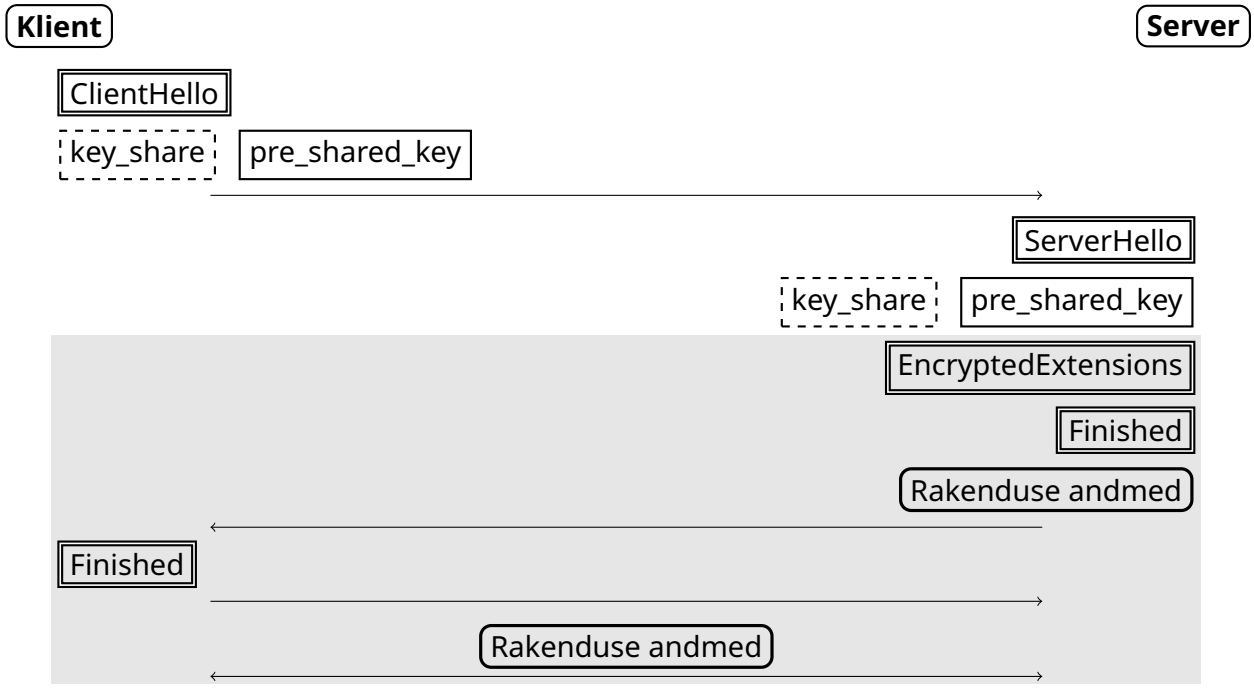
Joonis 1. TLSi võtmevahetusprotokolli teated. Sõnumid on kahekordse joonega, nende olulised laiendused ühekordse joonega kastides. Pideva joonega on kujutatud kohustuslikud, katkendjoonega valikulised osad. Hallil taustal olevad teated saadetakse krüptitud ja autenditud kanalis.

Kliendi võtmevahetusprotokollis autentimiseks on eri viise, samuti võib klienti (võtmevahetusprotokolli osana) mitte autentida. Server võib kliendilt nõuda sertifikaadiga autentimist, saates sõnumi `CertificateRequest`. Vastuseks saadab klient `Certificate`-sõnumis oma sertifikaadi, mille mingi sertifitseerimiskeskus on väljastanud, ning sõnumis `CertificateVerify` allkirja senivahetatud sõnumitele. Allkiri näitab, et klient teab privaatvõtit, mis vastab sertifikaadis olevale avalikuke võtmele.

Võtmevahetusprotokoll lõppeb `Finished`-sõnumitega, mille sisuks on sõnumiautentimiskood kõigile senivahetatud sõnumitele, kasutades selleks vastkokkulepitud ühist saladust. Niimoodi kinnitab üks pool teisele, et ta tõepoolest on välja arvutanud sama saladuse nagu teine pool.

Kui server ei toeta kliendi poolt valitud saladuse kokkuleppimise algoritmi või parameetreid, siis võib server vastata sõnumiga `HelloRetryRequest`, milles pakub välja oma algoritmi ja parameetrite valiku nende seast, mille klient `ClientHello`-teate osana saatis. See lisab protokollile ühe pendelduse.

TLS 1.3 toetab eelnevalt kokkulepitud võtmete (taas)kasutamist uute seansivõtmete loomisel ja/või poolte autentimisel. Kokkuleppimine võib toimuda väljaspool TLSi seansse (näiteks seadme initsialiseerimisel) või mõne eelmise seansi ajal. Seansiaegse kokkuleppimise otsustab server, saates kliendile `NewSessionTicket`-sõnumi. Sõnumi sisse paneb server *seansivõtme* – informatsiooni, mis aitab tal tulevases seansis, kui klient selle talle tagasi saadab, võtmed ja sinna juurde kuuluva konteksti üles leida. Sageli ongi selleks informatsiooniks needsamad võtmed ja nende juurde kuuluv kontekst, mille server on krüptinud ainult talle teadaoleva võtmega (*STEK:*



Joonis 2. TLSi võtmevahetusprotokolli teated eelnevalt kokkulepitud võtmeid kasutades. Tähistused on samad, mis joonisel 1.

Session Ticket Encryption Key).

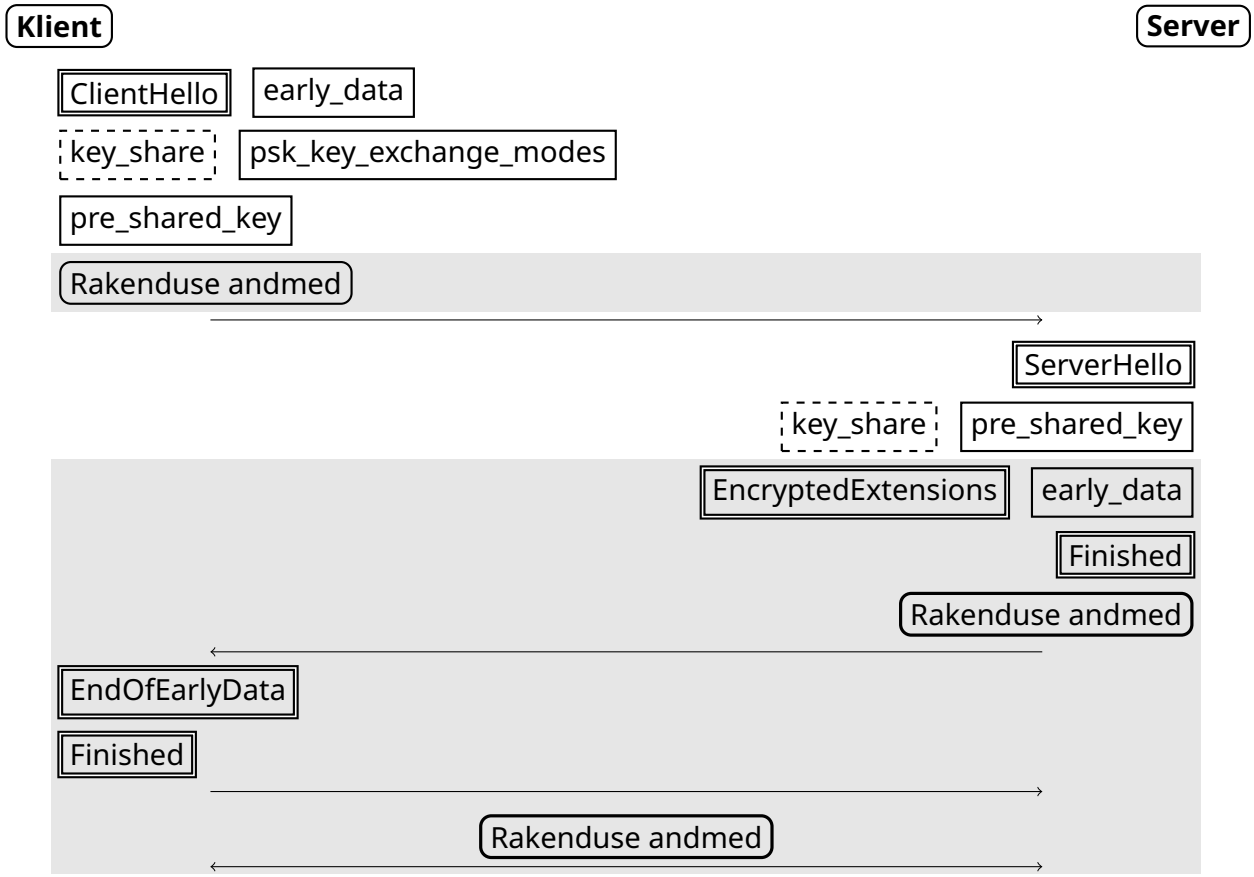
Võtmekehtestusprotokolli sõnumid eelnevalt kokkulepitud võtmete kasutamisel ja nende võtmete abil poolte autentimisel on toodud joonisel 2. `ClientHello`-teate osana saadab klient serverile nimekirja eelnevalt kokkulepitud võtmete nimedest, mida ta oleks nõus kasutama, ja server valib neist ühe välja. Väljavalitud võtit võib kasutada transpordiprotokollis liikluse kaitsmiseks. Sellele võib ühtlasi lisada täiendava ühise saladuse, mis `key_share`-laiendusi kasutades kokku lepitakse. Igal juhul on `Finished`-sõnumites olev sõnumiautentimiskood arvutatud võtmega, mis sõltub välja valitud eelnevalt kokkulepitud võtmest.

Näeme, et võtmekehtestusprotokoll oma joonisel 2 kujutatud variandis ei pea üldse kasutama avaliku võtmega krüptograafiat. Sellisena on ta kasutatav väga madala jõudlusega arvutusplatvormidel (sardsüsteemid, esemevõrk jms). Võrreldes muude protokollidega, mida samadel platvormidel kasutatakse, on TLS 1.3 eeliseks kõik talle rakendatud (formaalsetel meetoditel põhinevad) analüüsid (vt allpool).

TLS 1.3 toetab nn vastussõnumita (*0-RTT*) võtmekehtestust, kus klient saadab kaitstud andmeid kohe esimese teate koosseisus, vahetult pealt `ClientHello`-sõnumit. Nende andmete terviklus ja konfidentsiaalsus tagatakse *ainult* eelnevalt kokkulepitud võtmete abil, mis tähendab, et saavutatavad turvaomadused on natuke nõrgemad. Nende andmete jaoks jääb saavutamata tulevikuturvalisus, samuti ei ole nad kaitstud taasesitusrünnete vastu. Joonisel 3 on toodud sõnumite järjekord vastussõnumita võtmekehtestusel. Server võib esimese teate koosseisus olevad kaitstud andmed ka tagasi lükata, jättes `EncryptedExtensions`-sõnumisse `early_data`-laienduse lisamata.

3.1.2 TLSi analüüsid ja ründed TLSi vastu

TLS 1.3 loodi tihedas koostöös arvutiteadlastega, kes olid kogenud krüptoprotokollide analüüsis. Selle koostöö käigus tekkisid detailsed protokollimudelid ja -analüüsid [75] ning edasi arenes



Joonis 3. TLSi võtmevahetusprotokolli teated vastussõnumita võtmevahetusel. Tähistused on samad, mis joonisel 1.

ka krüptograafiliste protokollide analüüs kui teadusvaldkond. Tänu sellele leiti protokolliloogika vead üles enne protokolli spetsifikatsioon avaldamist. Ainus teadaolev avaldamisjärgne nõrkus protokolliloogikas on nn selfi rünne [76], mis puudutab eeljaotatud saladusi, mida me aruande eelmises versioonis [8] kirjeldasime ja mille neutraliseerimist kirjeldab RFC 9257 [77].

TLSi võtmevahetusprotokolli (ja transpordiprotokolli) peamised soovitavad turvaomadused on klassikalised: konfidentsiaalsus ja autentsus [78]. Neist esimene tähendab, et kui mingis protokollisessioonis rikkumata klient ja server omavahel teateid vahetavad, siis nende teadete sisu jääb tundmatuks liiklust pealtkuulavale vastasele, kes kliendi ja serveri privaatvõtmeid või eelnevalt kokkulepitud võtmeid ei tea. Tugevam omadus on *tulevikusalastatus*, mis nõuab, et lõpetatud seanssides vahetatud teated jääksid salajaseks ka peale privaatvõtmete lekkimist (kuid mitte peale kasutuselolevate krüptoalgoritmide ebaturvaliseks muutumist). Autentsus tähendab, et kui üks pool arvab, et ta sai mingi teate mingi seansiidentifikaatoriga protokolliseansis, siis teine pool arvab, et ta on sama teate sama identifikaatoriga seansis välja saatnud. Täiendav autentsusomadus on taasesituse vältimine, mis tähendab, et üht väljasaadetud teadet võetakse vastu ülimalt üks kord.

TLSi juurutuses on aga **protokolliloogika** üksainus kiht, mida rünnata. Selle alla ja peale jääb mitu teist kihti. Erinevaid ründeid, mis sihtisid TLSi eelmiste versioonidega seotud nõrkusi eri kihtides, kirjeldasime põhjalikumalt 2016. aasta aruandes [4]. Tuletame meelde, milliste kihtidega ja võimalike nõrkustega meil tegemist võib olla.

Krüptograafilised konstruktsioonid

TLSi protokollianalüüsid eeldavad, et sümmeetrilised ja asümmeetrilised konstruktsioonid,

mida protokollis kasutatakse (näiteks signatuuriskeemid, sümmeetrilised ja asümmeetrilised krüptosüsteemid) on „korrektselt“ realiseeritud, st rahuldavad mingeid kindlaid turvaomadusi.

Krüptograafilised primitiivid
Selleks, et krüptograafilised konstruktsioonid oleksid turvalised, on tarvilik (aga mitte piisav), et kasutatavad primitiivid (plokk- ja jadašifrid, räsifunktsioonid, tagauksega räsifunktsioonid) oleksid turvalised. Muuhulgas peavad kasutatavad võtmed või räsifunktsioonide väljundid olema piisavalt pikad.

Realisatsioonid
Protokolliloogika tuleb täpselt ja vigadeta realiseerida, sealjuures vältides ründeid võimaldavate kõrvalkanalite loomist. „Posteli seadus“ – ole konservatiivne oma tegemistes, aga liberaalne teistelt tulevate sisendite aktsepteerimises – [79, jaotis 2.10] turvaprotokollide realiseerimisel ei kehti.

Riistvara
Protokollirealisatsioon käivitatakse mingil riistvaral. Selle nõrkused, näiteks võimalikud arvutusvead või protsessidevahelised spetsifitseerimata sõltuvused võivad olla protokollirünnete aluseks.

Arusaamine
Täpsed garantiid, mida protokoll annab, ei pruugi olla niisama lihtsalt arusaadavad (näiteks: kui me räägime teise poole identiteedi kindlakstegemisest, siis kus täpselt jooksevad poole ja identiteedi piirid?). Protokolligarantiidest arusaamine on oluline protokolli suuremasse süsteemi integreerimisel.

Hiljutised ründed ja analüüsid uurivad nõrkusi mõnel neist kihtidest. Realisatsioonide analüüsid uurivad tavaliselt suurt hulka realisatsioone, et tuvastada, kui paljudel ja/või millistel neist on mingi konkreetne nõrkus.

Bleichenbacheri rünne [80] põhineb RSA-põhise võtmekehtestuse kõrvalkanalitel ning pärineb juba aastast 1998, kuid nende kõrvalkanalite sulgemine valmistab krüptoteekidele endiselt raskusi. Mõne aasta vanune analüüs [81] leiab, et mitmes teegis on need kõrvalkanalid ikka veel piisavalt suured, et Bleichenbacheri rünne oleks võimalik. Kindlalt likvideerib selle kanali ainult TLS 1.3-le üleminek, millega kaob RSA-põhise võtmekehtestuse tugi (vt. alapeatükki 3.1.4).

TLSi teostused võivad olla ka endiselt kaitsetud *madaldusrünnete* suhtes, kus kliendi ja serveri vahel paiknev ründaja veenab neid kasutama madalama versiooniga TLS-protokolli kui nad mõlemad oleksid suutelised toetama. TLSi võtmekehtestusprotokollis on olemas meetmed kõrgeima ühiselt toetatud versiooni kokkuleppimiseks, kuid mõnes serverirealisatsioonis ei ole need korrektselt teostatud. Selleks, et selliste serveritega ikkagi suhelda, võivad kliendirakendused omakorda realiseerida funktsionaalsuse (nn *downgrade dance*), mida protokoll ette ei näe. See täiendav funktsionaalsus annab ründajale täiendavaid võimalusi. Hiljutine paarisaja äritarkvararakenduse uuring [82] leidis kaheksa rakendust, mis tegid *downgrade dance*'i ning ei rakenda ka teisi meetmeid madaldusrünnete avastamiseks.

Seansipileteid (*session ticket*) kirjeldasime juba eespool. Seansipilet on krüptograafiline materjal, mida server üksinda loob ja kasutab (aga klient vahepeal salvestab). Seetõttu ei ole tema vorming standarditud (olgugi, et varasemate TLSi versioonide kontekstis on antud sellekohaseid soovitusi [83], sealhulgas krüptida AES-128-CBC-ga ning terviklust kaitsta HMAC-SHA-256-ga) ja seetõttu võib serverirealisatsioonides leiduda nendega seotud nõrkusi. Mõne aasta vanune suuremahuline analüüs [84] tuvastas, et levinud avatud lähtekoodiga realisatsioonides iseenesest ei ole olulisi seansipiletitega seotud nõrkusi, küll aga leidub juurutusi, kus

seansivõtme krüptimise võti (STEK) on liiga väikese entroopiaga. Samuti on mõnedes teekides seansipiletite krüptimiseks kasutusel selline režiim, mis lubab ainult umbes 2^{32} pileti krüptimist.

Rowhammer-rünnetest oleme kirjutanud mitmes eelmises aruandes [4, 8] ning kahjuks ei ole põhjust neid puudutava aruteluga praegu lõpetada. Tuletame meelde, et *Rowhammer*-i aluseks on tänapäevase dünaamilise muutmälu ebatäiuslikkus, kus ühe konkreetse biti väga sage lugemine või kirjutamine võib mõjutada ka kõrvalasuvate bittide väärtust [85]. Tegemist on riistvaralise nõrkusega, mistõttu tarkvaralist kaitset on selle vastu leida üsna keeruline. Viimaste aastate jooksul tundub, et see rünneteklass on muutunud küpsemaks, millele viitab teoreetilisemate ründeraamistike ilmumine [86], mida on järgmiste rünnete juures kasutatud; muuhulgas hõlmavad need ründeid TLSi realisatsioonide vastu [87]. Küpsemine annab ka lootust kaitsemeetodite süstemaatiliseks arenguks, vähemasti raamistikest tuletatavate rünnete vastu.

3.1.3 Kliendi autentimine TLS-seansis

TLS-CCA (TLS Client Certificate Authentication) on standardne viis kliendi autentimiseks TLS-seansis. Olukorda, kus autenditakse nii server kui ka klient, mis on levinud praktika, nimetatakse ka vastastikuseks TLSiks (mTLS, Mutual TLS).

Server võib paluda kliendilt autentimist, näidates, millist kliendisertifikaati ta ootab (kriteeriumiteks võivad olla toetatud CAde eraldusnimed, toetatud algoritmid, TLS 1.3 puhul sertifikaadi laiendused ja tingimused nende väärtustele).

Klient esitab sertifikaadi ning signeerib sama, sertifikaadis näidatud võtmepaariga senised kätlussõnumid (ja TLS 1.3 korral sertifikaadipäringu kontekstiinfo). Klient võib sertifikaadi esitamisest ka loobuda, seda näiteks juhul, kui sobivatele tingimustele vastav sertifikaat puudub. Sellisel juhul on serveri otsustada, kas seansiga saab jätkata või mitte.

TLSi versioonides kuni 1.2-ni edastatakse kliendi sertifikaat esmase kätluse ajal kaitsmata kujul ning on sellisena pealkuulajale nähtav. TLS 1.2 korduskätlusel ja TLS 1.3 kätluse ajal on sertifikaat kaitstud. TLS 1.3 korral toimub sertifikaatide edastamine krüptitult ja kliendisertifikaat edastatakse vaid eelnevalt edukalt autenditud serverile.

Eestis levinud TLS-CCA kasutus on ID-kaardiga autentimine, aga TLS-CCAd kasutatakse ka süsteemidevahelises kommunikatsioonis, nt X-tee turvaserverite vahel[1].

TLS-CCA võimaldab kliendi võtmepaari krüptograafiliselt siduda TLS-seansiga. Seda võimaldab ka standard RFC 8471 (Token Binding-protokoll), kuid levinud brauserites, nagu Firefox, Chromium, Microsoft Edge selle tugi puudub (viimastes oli Token Binding-protokolli tugi varasemates versioonides olemas, kuid uutes versioonides on sellest loobutud).

Tänu kahepoolsele ja kätlusega seotud autentimisele on TLS-CCA praktiliselt kasutatav viis, mille abil vältida infovahetuse krüptograafilise kaitse märkamatut nõrgenemist või kadu keskkondades, kus kasutatakse TLS-liikluse inspekteerimise vahendeid (tulemüürid, antiviirused) või rakendatakse liikluse vahendamist rakenduskihis (õngitsusründed, PSD2 *embedded* ja *decoupled* liidestused[2]).

[1] https://docs.x-road.global/UseCases/uc-mess_x-road_member_communication_use_case_model.html

[2] https://www.europeanpaymentscouncil.eu/sites/default/files/kb/file/2018-05/APIEG30-18Authenticationguidance(SCA).pdf

3.1.3.1 TLS-CCA ja Eesti ID-kaardiga autentimine

Kuivõrd ID-kaardi kasutuse algusaegadel olid ID-kaart ja temaga tehniliselt samaväärsed digiallkirjavahendid ainsad eksisteerivad digiallkirjavahendid, läks käibele lihtsustatud oskusteave ID-kaardi jaoks TLS-CCA konfigureerimise kohta, mis kattis vaid üht vajaliku konfiguratsiooni aspekti – usaldatud sertifitseerimisahelate määramist.

Novembris 2025 vahetus ID1-formaadis dokumentide (st ID-kaartide) tootja. Senise kaarditootja IDEMIA asemel hakkas Eesti ID-kaarte tootma Thales. Uue ID-kaardi olulisemad omadused ja vajalikud muudatused on toodud ID-kaardi veebilehel[3]. Koos uute kaartidega on lisandunud uus usaldusteenuse osutaja Zetes. See tähendab, et ID-kaardiga autentimiseks ja/või allkirjastamiseks peavad süsteemid toetama kahte usaldusteenust. Üleminekuperioodil on paralleelselt kasutusel nii senise IDEMIA kui ka uue tootja Thalese ID-kaardid. Seoses uue usaldusteenuse osutaja (Zetes) lisandumisega tuleb lisaks SK ID Solutions sertifikaadiahelatele lisada Zetese sertifitseerimisahela tugi ning uue (Thalese) ID-kaardi sertifikaatide kehtivuse kontrollimiseks kasutada Zetese OCSP-teenust. Aruande kirjutamise hetkel on vastavate usaldusteenuse pakkujate veebilehtedel antud järgmiste sertifikaadiahelate kirjeldused:

SK ID Solutions:

- SK ID Solutions ROOT G1E (Elliptic Curve CA, aktiivne, väljaandmine)
- EE-GOVCA2018 (aktiivne, väljaandmine)
- SK ID Solutions ROOT G1R (RSA CA, ei ole aktiivne, varund)
- EE Certification Centre Root CA (ei ole aktiivne, ainult tühistamine)

Zetes:

- EEGovCA2025 (aktiivne, väljaandmine)

TLS-CCAd kasutades tuleb teenuses konfigureerida uue ID1 kaardi usaldusahel (sertifikaadidid ja tühistatud sertifikaatide loendid on leitavad usaldusteenusepakkujate veebilehtedelt[4,5,6]) ja teha vajalikud muudatused kasutaja sertifikaadi kehtivuse kontrollimiseks.

Selle aruande kirjutamise hetkel soovitab Riigi Infosüsteemi Amet e-teenustes ID-kaardiga isikutuvastuseks eelistada Web eID (vt ptk 5.3) lahendust[7], põhjuseks selle parem töökindlus ja kasutajasõbralikkus. Vastavalt Web eID arhitektuuri dokumentatsioonile[8] toimub autentimine vastavalt Web Authentication API põhimõtetele, mitte kasutades TLS-CCAd. Veebiserverites ei ole enam vaja spetsiaalset TLS-CCA konfiguratsiooni, mis lihtsustaks ID-kaartide kasutamist pilveteenustes. Samas soovitatakse endiselt kasutada TLS-CCA lahendust rangete turvanõuetega e-teenustes, mille hulka kuuluvad näiteks e-teenused, kus talletatakse riigisaladusi[9].

Seoses üleminekuga uuele veebi autentimise ja allkirjastamise lahendusele (Web eID[10]) on lõppenud eelmise veebilehitsejate ja kiipkaardilugejate draiverite vahelisi päringuid vahendava laienduse Token Signing arendamine ning seda ei ole enam ka Chrome'i veebipoes ja Microsoft Edge'i laienduste poes[11]. Juhul kui Token Signing on ikka veel veebibrauseri laienduste valikus,

[3] https://www.id.ee/artikkel/thales-id-kaart/
[4] https://repository.eidpki.ee/crt/
[5] https://repository.eidpki.ee/crl/
[6] https://www.skidsolutions.eu/resources/certificates/
[7] https://www.id.ee/artikkel/isikutuvastus-id-kaardiga/
[8] https://github.com/web-eid/web-eid-system-architecture-doc
[9] https://www.id.ee/artikkel/isikutuvastus-id-kaardiga/
[10] https://web-eid.eu/
[11] https://github.com/open-eid/chrome-token-signing

tuleb see eemaldada[12]. Mobiilseadmete (telefon, tahvelarvuti jne) brauserites Web eID laiendus puudub.

Web eID kasutab eID-kaartidega otse suhtlemiseks PC/SC APIt[13]. See on platvormist sõltumatu API ligipääsuks kiipkaardilugejatele, mis on spetsifitseeritud madalamal tasemel kui operatsioonisüsteemi krüptograafilised APId ning on stabiilne. PKCS#11 APIt kasutatakse ainult juhul, kui PC/SC APDU (Application Protocol Data Unit) protokolli spetsifikatsioon ei ole eID-kaardi jaoks kättesaadav. Elektroonilised ID-kaardid on jagatud kahte astmesse:

- 1. astme kaartide puhul (Eesti, Läti, Soome) on PC/SC APDU protokolli spetsifikatsioon olemas ja kaartidega otse suhtlemiseks kasutatakse PC/SC APIt;
- 2. astme kaartide puhul (Horvaatia, Leedu) ei ole PC/SC APDU protokolli spetsifikatsioon kättesaadav ning selle asemel kasutatakse PKCS#11 APIt, mis eeldab kolmanda poole PKCS#11 mooduli olemasolu.

3.1.4 Soovitused

Selles alapeatükis anname soovitusi TLSi konfigureerimiseks. Need soovitused põhinevad meie teadmistel TLSi eri võtmevahetusviiside ja andmete kodeerimise turvalisuse kohta, samuti meie soovitustel kasutatavate krüptoprimitiivide ja võtmepikkuste kohta peatükis 2.

Aruanne ei sätesta soovituste kehtivusaega. Postkvant-krüptograafiale üleminekul tuleks lähtuda riiklikust postkvant-krüptograafia tegevuskavast. Samuti on relevantsed BSI soovituste kehtivusajad[14].

3.1.4.1 TLSi versioonid

Soovitame tungivalt kasutada kõigil võimalikel juhtudel TLS versiooni 1.3. TLS 1.2 on õigustatud ainult väga üksikutel juhtudel, näiteks siis kui klientrakendus on aastast 2020 või veel vanem. Veel vanemaid TLS versioone kasutada ei tohi.

3.1.4.2 TLS 1.2 ja TLS 1.3 raames kasutatavad krüptograafilised primitiivid

TLS 1.2 šifrikomplektid määravad algoritmid võtmekehtestuse, (autentimisel) signeerimise ja sümmeetrilise krüptimise jaoks ning fikseerivad kasutatava räsifunktsiooni.

TLS 1.2 puhul ei ole RSA-võtmekehtestuse[15] kasutamine enam lubatud, sest seda pole siiani õnnestunud turvaliseks muuta. Tõepoolest, alates sellest ajast, kui Bleichenbacher kirjeldas rünnet [80] RSA PKCS#1 v1.5 standardis [88] välja pakutud avatekstide polsterdusmehhanismi (enne nende RSAga krüptimist) vastu, pole suudetud välja pakkuda viise, mis võtaksid ründajalt praktilise võimaluse seda rünnet läbi viia. Pigem on leitud uusi võimalusi ründe läbiviimiseks [89], kus arvutiriistvara ja operatsioonisüsteemide poolt rakendustele pakutavate abstraktsioonide mittetäielikkus loob kanaleid, mille abil ründaja saab talle vajaliku informatsiooni kokku koguda.

Oluline on ka, et kuna rünnakud RSA-võtmekehtestuse vastu võimaldavad võltsida RSA-signatuuri, siis saab ebaturvalist võtmekehtestust pakkuva serveri abil rünnata ka servereid, mis kasutavad sedasama RSA-võtmepaari ainult autentimiseks (nt TLS 1.3 servereid). Seepärast

[12] https://www.id.ee/artikkel/veebibrauserite-seadistamine-id-kaardi-kasutamiseks/
[13] https://github.com/web-eid/web-eid-system-architecture-doc
[14] https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-2.pdf?__blob=publicationFile&v=11
[15] TLS 1.3 ei sisalda RSA-võtmekehtestust.

on võtmekehtestuseks lubatud kasutada vaid efemeersete[16] Diffie-Hellmani võtmepaaridega võtmekehtestust (ECDHE ja DHE).

RSA-võtmekehtestuse kasutamine pole soovitatav. Juhul kui koostalitlusvõime tagamiseks on TLS 1.2 jaoks RSA-võtmekehtestuse kasutamine kasvõi ajutiselt vältimatu, peab hoolitsema, et kasutataks unikaalseid, vaid selle serveriga seotud võtmeid. Võimaluse korral tuleb sellise serveri kasutamist piirata ka kliendi IP-aadressi järgi või vähendada võimalikku ründepinda mingi muu meetodiga. Olenemata TLSi versioonist ei soovitata sama võtmepaari kasutada mitmes iseseisvas TLS-serveris. Eriti ettevaatlik peab olema võtmetega, mille jaoks on olemas *wildcard*-sertifikaadid (nimedega kujul `*.example.com`) või mitut nime sisaldavaid sertifikaadid ning mida sellisena on mugav kasutada mitmes, sageli erineva TLS-protokolli teostuse või konfiguratsiooniga serveris.

Nii TLS 1.2 kui TLS 1.3 puhul on lubatud RSA-PSSi kasutamine suhtluspoolte autentimiseks (st võtmetüübina serveri või kliendi sertifikaadis). RSA PKCS #1 v1.5 kasutamine pole soovitatav. BSI keelas RSA PKCS #1 v1.5 kasutamise pärast 2025. aastat.

Efemeersete võtmetega Diffie-Hellmani võtmekehtestuseks astendamisega jäägiklassirühmas (DHE) jaoks peab kasutama piisavalt suure järguga rühma. DHE-võtmekehtestuse korral sõltub võtmekehtestusprotokolli abil kokkulepitud peasaladuse turvalisus otseselt ja ainult kasutatud rühma turvaomadustest ja mitte näiteks serveri autentimissertifikaadis kajastatud võtmepaari omadustest[17]. TLS 1.2 ja vanemad lubavad kasutada isegenereeritud rühmi, mille turvaomadusi on raske hinnata. TLS 1.3 lubab kasutada vaid valitud hulka, teadaolevalt turvalisi rühmi ning RFC 7919 [91] lisab nimega rühmade toe ka varasematele TLSi versioonidele. TLS 1.2 ja TLS 1.3 jaoks tohib kasutada vaid rühmi, mis on piisava turvatasemega vastavalt tabelile 4. Kui DHE kasutamine pole vajalik tagasiühilduvuse jaoks, soovitame seda mitte kasutada, sest elliptkõveratepõhine ECDHE on parema jõudlusega ja seda toetab suurem osa tänapäevaseid TLS-teostusi (sh värskemad brauserid).

Nii TLS 1.2 kui TLS 1.3 korral on lubatud sümmeetrilise krüptimise jaoks kasutada vaid autenditud töörežiime. Plokkšifritest võib kasutada vaid AESi, jadašifritest toob TLS 1.3 sisse ChaCha20. Lubatud šifrid/töörežiimid on AES-GCM, AES-CCM ja CHACHA20-POLY1305. AES-CCMi kasutamine on lubatud, aga vaid siis, kui teised kaks pole kasutatavad. Kui kasutusel on AES-CCM, siis peab sõnumiautentimiskood olema 16-baidine, mitte 8-baidine, st „CCM_8“ [92] töörežiimid ei ole lubatud.

Šifrikomplektis määratud räsifunktsiooni kasutatakse sõnumiautentimiskoodi (HMAC) koosseisus. Sümmeetriliste šifrite autenditud töörežiimides (AEAD – *Authenticated Encryption with Additional Data*) kasutatakse sõnumiautentimiskoodi vaid pseudojuhusliku jada generaatori (PRF – *Pseudorandom Function*) sisendi komponendina. Kui šifrikomplekt räsifunktsiooni eraldi ei maini, kasutatakse TLS 1.2 ja uuemate juures vaikimisi SHA-256. See on ka soovitus – kasutada SHA-256 või tugevamat räsifunktsiooni.

3.1.4.3 TLS 1.2 jaoks lubatud šifrikomplektid

TLSis kasutatavaid šifrikomplekte ja nende kahebaidiseid identifikaatoreid registreerib IANA[18]. Samas registris on toodud ka väli iga konkreetse šifrikomplekti soovitatavuse kohta.

Lubatud šifrikomplektide nimekiri lähtub IANA soovitustest ja eelmises jaotises kirjeldatud

[16] Ühekordsete, lühikese elueaga. Täpsemat käsitlust vt [90, jaotis 1.5.7].
[17] https://weakdh.org/
[18] https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml#tls-parameters-4

reeglitest.

TLS 1.2 jaoks on lubatud järgnevad IANA soovitatavad šifrikomplektid:

- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256

Vajaduse korral on lubatud ka järgnevad **IANA ebasoovitatavaks märgitud** šifrikomplektid:

- TLS_DHE_RSA_WITH_AES_128_CCM (see ja kõik järgmised üksnes turvalise DHE rühmaga! Vaata ka lõiku eespool DHE võtmekehtestuse kohta – see on lubatud, kuid võimaluse korral peaks seda vältima.)
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_DHE_RSA_WITH_AES_256_CCM
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_DHE_RSA_WITH_CHACHA20_POLY1305_SHA256

TLS 1.2 kasutamise korral tuleb keelata kätluse kordamine juba loodud seansis (*renegotiation*).

3.1.4.4 TLS 1.3 jaoks lubatud šifrikomplektid

TLS 1.3 šifrikomplektid määravad vaid sümmeetrilise šifri ja räsifunktsiooni ning seetõttu neid TLSi eelmiste versioonidega kasutada ei saa.

Lubatud TLS 1.3 šifrikomplektid on:

- TLS_AES_256_GCM_SHA384
- TLS_CHACHA20_POLY1305_SHA256
- TLS_AES_128_GCM_SHA256
- TLS_AES_128_CCM_SHA256 (vaid juhul, kui AES-GCM või ChaCha20-Poly1305 pole kasutatavad)

Hübriidne võtmevahetus TLS 1.3-s Võimaluse korral tuleks TLS 1.3 klient konfigureerida kasutama `supported_groups` laiendusväljal esimese rühma väärtusena `X25519MLKEM768`. Võimaluse korral uuendage TLS-servereid, nii et need toetaks `X25519MLKEM768` hübriidrühma[19] ning seadistage see prioriteetseks. Võtme hübriidkehtestus kaitseb süsteemi *harvest-now-decrypt-later* rünnete eest.

3.1.4.5 Andmete pakkimine

TLS 1.2 kasutamise korral on oluline keelata ka andmete pakkimine (*compression*) TLS-kihis, sest see muudab TLSi rünnatavaks. Võimaluse korral tuleb keelata ka andmete pakkimine

[19] Määratlus leitav IETFi portaalist, https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/.

TLS-seansi sees kasutatavas rakendusprotokollis (nt HTTP), sest selline pakkimine võimaldab andmete lekkimist sarnaselt pakkimisele TLS-kihis [93]. Kui pakkimise keelamine pole võimalik, peab rakendama muid leevendavaid meetmeid nagu päringuvõltsimise (*Cross-Site Request Forgery*) vastane kaitse ja väljundi pikkuse juhuslik muutmine, et peita pakkimise tekitatud pikkuseerinevusi, mida seda tüüpi ründed kasutavad.

3.1.4.6 Autentimine Eesti ID-kaardiga

Laialt kasutusel olev konfiguratsioon, mille sisuks on õigesti paigaldatud sertifikaadihierarhiad, ei anna tegelikult soovitud turvaomadust – kindlust, et krüptograafilisel tasemel oleks TLS-CCA autentimine võimalik vaid ID-kaardiga. Selleks, et saavutada korrektset konfiguratsiooni, on vaja pidada silmas järgmist.

- Sertifikaadi kuulumine kindlasse sertifikaadihierarhiasse näitab seda, et sertifikaadi sees olev informatsioon on usaldusväärne. Seejuures tuleb aga silmas pidada, et vaadata tuleb hierarhiat tervikuna, mitte ainult juursertifikaati. Sama juursertifikaadi all võib olla hierarhia kõrvalharusid, mida ei tohi etteantud otstarbeks usaldada. Oluline on kontrollida sertifikaadi väljastamise aluseks olevate poliitikate identifikaatoreid (OID-sid).
- Informatsioon sertifikaadi esitaja ning lubatud kasutusotstarbe kohta on saadaval üksnes sertifikaadis endas.

Selleks, et tuvastada, kas kliendi autentimisel kasutatud X.509 sertifikaat on ID-kaardi autentimissertifikaat ning usaldusväärne, tuleb lisaks sertifikaadi väljastanud CAle kontrollida ka sertifikaadi laiendust `X509v3 CertificatePolicies`[20], mis sisaldab sertifikaadi väljastamise aluseks olevate poliitikate identifikaatoreid (OIDsid). Viited kehtivatele sertifitseerimispoliitikatele leiab id.ee veebilehelt[21] ja usaldusteenuse pakkujate (SK ID Solutions, Zetes) veebilehtedelt[22,23,24].

Sertifikaadihierarhiate konfigureerimise osas soovitame pöörduda vastavate tehniliste juhendmaterjalide poole[25].

3.1.4.7 OCSP kasutamine

Võtmekehtestusprotokolli „peamises“ voos (joonis 1) toimub serveri autentimine sertifikaadiga. Kliendil on vaja veenduda, kas sertifikaat on veel kehtiv; selle info saab ta serveri sertifikaadi väljastanud CAlt. Kaks peamist kehtivusinfo levitamise viisi on sertifikaatide tühistusnimekirjad (*Certificate Revocation List*, CRL) ja võrgusertifikaadi staatuse protokoll (*Online Certificate Status Protocol*, OCSP). CRLi leidmise või OCSP-päringu tegemise aadress on sertifikaadis kirjas; kliendirakendus peab mõnda neist kasutama.

OCSP-päringu tegemine riivab kliendi privaatsust, sest CA saab teada, millise serveriga ta ühenduda tahab. Seetõttu on nii mõnigi CA lõpetanud OCSP-päringutele vastamise[26]. Privaatsusriive vähendamiseks saab kasutada TLS-sertifikaadi olekupäringu laiendit [70, jaotis 4.4.2.1], mis lubab serveril endal saata kliendile OCSP-vastus, mille server on CAlt hiljuti

[20] https://datatracker.ietf.org/doc/html/rfc5280#section-4.2.1.4
[21] https://www.id.ee/en/article/certificate-policy-for-identity-card-digital-identity-card-residence-permit-card-and-diplomatic-identity-card-2/
[22] https://www.skidsolutions.eu/resources/certificate-policies/
[23] https://www.skidsolutions.eu/resources/profiles/
[24] https://repository.eidpki.ee/repository/
[25] https://www.id.ee/artikkel/isikutuvastus-id-kaardiga/
[26] https://letsencrypt.org/2024/12/05/ending-ocsp

küsinud. Kui serverile sertifikaadi väljastanud CA toetab OCSPd, siis soovitame server seadistada toetama seda laiendit.

3.1.5 Muud TLSi rakendamist kirjeldavad dokumendid

TLSi turvalise konfigureerimise soovitusi annavad ka muud organisatsioonid. Nende avaldatavate dokumentide uuendamistsükkel võib olla siinse aruande omast erinev. Seetõttu on kasulik neid soovitusi regulaarselt kontrollida. Järgnevalt toome mõned viited materjalidele, mida jälgida.

IETF haldab heade tavade kirjeldust, sellest on väga hiljuti avaldatud uus versioon: RFC 9325 [94].

NIST on avaldanud juhiseid TLSi valimiseks, konfigureerimiseks ja kasutamiseks [95]. Oma soovitused annavad ka BSI (Saksamaa) [96] ja Madalmaade rahvuslik küberturbekeskus [97].

OWASPil on spikker (*cheat sheet*) TLSi turvaliseks konfigureerimiseks[27], muuhulgas leiab sealt konfiguratsiooni testimiseks mõeldud tööriistade nimekirja.

Brauseritootja Mozilla haldab soovitusi TLS-serverite turvaliseks konfigureerimiseks[28]. Mozilla soovitused jagunevad kolme gruppi vajaliku tagasiühilduvuse taseme järgi: moodsad brauserid/kliendid (TLS 1.3 toega, tagasiühilduvust pole vaja), keskmine/mõõdukas tase (soovituslik konfiguratsioon üldkasutatavale serverile) ja vana (erandlik tagasiühilduvus; kasutamiseks vaid erandkorras).

Ka NSA on avaldanud soovitused [98] vanematest TLSi versioonidest ja nõrkadest algoritmidest loobumise kohta. Üheks oluliseks erinevuseks selle aruande ja NSA soovituste vahel on, et meie soovitame RSA-võtmekehtestusest loobuda.

Veebibrauserites kasutatava TLSi ja PKI osas lepivad brauseritootjad ja CAd kokku omavahelises foorumis[29]. Selle organisatsiooni kaudu kehtestatakse alusnõuded (*baseline requirements*)[30], mis on kohustuslikud CAdele, kes soovivad oma juursertifikaatide levitamist brauserites.

Postkvant-krüptograafia kasutust TLSis kirjeldab detailsemalt alapeatükk 7.1.1.

3.2 SSH

3.2.1 Kirjeldus

Secure Shell (SSH) on krüptoprotokoll, mis on projekteeritud võrguteenuste turvamiseks. SSH esimene rakendus oli kaugterminal, turvamata protokollide nagu *telnet* ja *rexec* asendusena. Praktikas võib SSHd kasutada igasuguste TCP-põhiste võrguprotokollide turvamiseks.

Tarkvarapaketi SSH esmaversiooni, mis toetas vaid protokolli SSH-1, avaldas Tatu Ylönen 1995. aastal. Protokolli versioon SSH-1 sisaldab mitmeid vigu [99], mille tõttu ei soovitata seda enam kasutada.

IETF standardis 2006. aastal [100] protokolli SSH-2. Protokoll on krüptograafiliselt parema lahendusega kui SSH-1 ning sisaldab tuge suurema hulga kasutusmallide jaoks (näiteks rakenduse-ühenduste multipleksimine üle ühe TCP/SSH-ühenduse[31]).

[27] https://cheatsheetseries.owasp.org/cheatsheets/Transport_Layer_Security_Cheat_Sheet.html
[28] https://wiki.mozilla.org/Security/Server_Side_TLS
[29] https://cabforum.org
[30] https://cabforum.org/baseline-requirements-documents/
[31] https://en.wikibooks.org/wiki/OpenSSH/Cookbook/Multiplexing

SSH-2 koosneb transpordiprotokollist [101], (kasutaja) autentimise protokollist [100] ja kommunikatsiooniprotokollist [102]. Neist viimane spetsifitseerib kaugterminali jooksutamiseks vajaliku protokolliloogika ja ei ole olemuselt turvaprotokoll. Transpordiprotokolli abil lepivad klient ja server kokku ühise saladuse, kasutades selleks Diffie-Hellmani võtmevahetust (ptk 2.6.3.1). Seejuures server signeerib oma teate, mis tähendab, et pärast võtmevahetust on server autenditud. Transpordiprotokoll [101] spetsifitseerib ka kanali krüptimisviisi. Kasutaja autentimise protokolli kasutades saab klient ennast serverile autentida, kasutases selleks kas salasõna või signatuuri.

SSH-2 krüptograafilised omadused määratakse kolme komponendiga: šiffer (*Encryption Algorithm*), autentsustõendi algoritm (*MAC Algorithm*) ja võtmekehtestuse meetod (*Key Exchange Method*). Normeeritud väärtused nendele parameetritele on kirjas IANA vastavas registris[32], kuid realisatsioonid ei pea piirduma loetletud algoritmidega, vaid võivad lisada omi. Seetõttu on raske anda üksikasjalikku juhist SSH šifrite konfigureerimiseks (erinevalt näiteks TLSist). On võimalik anda mõned järgimiseks sobivad suunised, kuid pärandsüsteemide ühilduvusnõuded võivad seada omad nõuded.

Andmevahetuse kaitsmisest sõltumatult võidakse krüptograafilisi meetodeid kasutada ka kasutaja autentimiseks. Süsteemide kaugpääsu võimaldavate võtmete turvataset peab regulaarselt kontrollima ja kui see pole enam piisav, siis nõrgaks osutunud võtmed asendama või eemaldama. Lisaks võtmetüübile on oluline ka kasutatav autentimisalgoritm. Näiteks on kõik esimeses selleteemalises RFCs [100] kirjeldatud algoritmid praeguseks kasutuskõlbmatud, sest kasutavad SHA-1 räsi.

Konkreetse SSH realisatsiooni toetatavate algoritmide kohta leiab teavet vastava tarkvara juhendmaterjalidest.

3.2.2 Analüüsid ja ründed

Tabel 5. Teadaolevate nõrkuste riski hindamine

CVE	Mõju	Rakendatavus	Olek
CVE-2025-26465 [103]	serveri identiteedi võltsimine, vahendusrünne (MITM)	**keskmine** (vajab VerifyHostKeyDNS sisselülitamist)	Parandatud versioonis 9.9p2
CVE-2025-26466 [104]	teenusetõkestus (CPU/mälu)	**keskmine** (SSH2_MSG_PING)	parandatud versioonis 9.9p2
CVE-2024-6387 [105]	„regreSSHion“ autentimata kaugkäitus	**kõrge**	parandatud versioonis 9.8p1
CVE-2023-48795 [106]	„Terrapin Attack“ potentsiaalne vahendusrünne	**keskmine**	parandatud versioonis 9.6
CVE-2023-38408 [107]	koodi kaugkäitus (RCE) läbi PKCS#11 moodulite	**kõrge** (vajab agendi edastamist)	parandatud versioonis 9.3p2

Tabel 5 esitab tuntud ohtude riskihinnangud. Kõik nimetatud vead on uuemates versioonides ära

[32] https://www.iana.org/assignments/ssh-parameters/ssh-parameters.xhtml

parandatud. Vanemaid OpenSSH versioone kasutavad süsteemid on jätkuvalt ohustatud. Tabeli allikaks on OpenSSH koduleht [108].

3.2.3 Soovitused

Järgnevad soovitused tuginevad BSI soovitustele SSH turvalise kasutamise kohta [109]. Samuti tuginevad need meie soovitustele krüptoprimitiivide kohta, mille andsime eelmises peatükis. Vastavad seadistused tuleks sisse viia nii SSH-serveri kui -kliendi konfiguratsiooni, et vältida madaldusrünnet.

Lisaks väärib mainimist, et BSI [32] kiidab pikaajaliseks konfidentsiaalse informatsiooni kaitse tarbeks heaks postkvant-algoritmi ML-KEM-768, mis on saadaval ja soovituslik viimases OpenSSH versioonis hübriidses vormis.

- Šifrite hulgast tuleks kasutada ainult ChaCha20-Poly1305 ning AES-GCM variante, nende puudumisel AES-CTRi. Kui kasutusel on ChaCha20-Poly1305, siis tuleb tagada, et nii klient kui ka server kasutavad Terrapin-ründe suhtes immuunset versiooni tarkvarast.
- Autentimiskoodide hulgast tuleks kasutada ainult HMAC-algoritme SHA-2 perekonna räsidega.
- Võtmekehtestusalgoritmide hulgast tuleks valida sellised, mis kasutavad DH- või ECDH-võtmekehtestust postkvant-võtmekapseldusmehhanismiga, SHA-2 või SHA-3 perekonna räsialgoritme ning selliseid Diffie-Hellmani rühmi või elliptkõveraid, mis vastavalt tabelile 1 võimaldavad vähemalt 128-bitist turvataset (näiteks DH-rühmad *group1* ja *group14* ei kvalifitseeru, sest nende järk ei ole piisavalt suur).
- Võtmekehtestuse kvantründekindluse jaoks tuleks kasutada ECDH-võtmekehtestust hübriidrežiimis postkvant-turvalise võtmekapseldusmehhanismiga, nagu `mlkem768x25519-sha256`.
- Serveri ja kasutaja (kui kasutatakse kasutaja autentimist võtmepaariga) võtmetena tuleb kasutada piisava turvatasemega (vt tabel 1) elliptkõveratel baseeruvaid võtmeid ja neile vastavaid autentimisalgoritme.

Internetis leidub hulgaliselt materjale[33] SSH turvalise konfigureerimise kohta, kuid nendega tutvudes tuleb alati veenduda, et juhised pole aegunud ning kehtivad lugejat huvitava SSH realisatsiooni ja versiooni kohta.

3.3 IPsec

3.3.1 Ülevaade

IPsec (RFC 4301 [110] jt) on IETFi defineeritud turvaarhitektuur ja protokollistik, mis lisab krüptograafilise kaitse OSI-mudeli võrgukihis (vastandina nt TLSile, mis teeb seda transpordikihis). IPSeci abil turvatakse näiteks asutustevahelist sidet (kahe paljudest arvutitest koosneva IP-võrgu ühendamine üle üldkasutatava sidevõrgu) aga ka kaugpääsu, mille korral ühenduvad üksikud arvutid turvaliselt kaitstud võrku või serverisse.

IPsec turvaparameetrite konfigureerimise kohta on soovitusi andnud ka BSI [111] ja NIST [112].

IPsec kirjeldab:

- turvaühendi (*security association* ehk SA) mõistet,

[33] Näiteks: https://linux-audit.com/audit-and-harden-your-ssh-configuration/

- protokolle turvaühendi (SA) kokkuleppimiseks ja haldamiseks ning
- protokolle turvaliseks andmevahetuseks kokkulepitud turvaühendit kasutades.

IKEv2 (*Internet Key Exchange version 2*) [113] on protokoll poolte vastastikuseks autentimiseks ning turvaühendite tekitamiseks ja haldamiseks. Praegune versioon 2 on edasiarendus varasematest IKEv1 ja ISAKMP (*Internet Security Association and Key Management Protocoll*) protokollidest. IKEv2 pakub võrreldes IKEv1-ga hulka mugavusteenuseid (laiendatava autentimise tugi, IP-rändluse tugi, seansi elutukse jms) ning on projekteeritud olema turvalisem kui IKEv1. Turvanõuetele vastava seadistuse korral võib kasutada turvaühendi kokkuleppimiseks ka IKEv1, kuid IKEv1 turvaline seadistamine on keerukam kui IKEv2 turvaline seadistamine. Eelistatud on IKEv2 kasutamine.

IP-pakettide krüptograafilise kaitsega tegelevad protokollid AH (*Authentication Header*) [114] ja ESP (*Encapsulating Security Payload*) [115]. AH tagab tervikluse paketi lastile (*payload*) ja valitud päiseväljadele saatja identiteedi kontrolli. ESP tagab lisaks AH pakutavale ka paketi sisu konfidentsiaalsuse ja piiratud kaitse liiklusvoo analüüsi vastu. IPsec arhitektuur võimaldab AH ja ESP kasutamist ükshaaval või kombinatsioonis, konkreetne kasutus sõltub vajadusest.

Mõlemad protokollid toetavad transportrežiimi ja tunnelirežiimi. Transportrežiimis lisatakse algsele paketile, algse päise järele, lisapäis, mis seob paketiga turvaühendi ja sisaldab (ESP korral valikuliselt) tervikluse kontrolliks vajalikke andmeid. Marsruutimiseks kasutatakse algse paketi päist ning selles olnud aadresse. Transportrežiimis ei saa tagada IP-paketi muutuvate väljade terviklust ning ESP korral on paketi päis pealtkuulajale nähtav (võimalik on liiklusvoo analüüs).

Tunnelirežiimis on kogu algne pakett uue, turvalüüside vahel saadetava paketi lastiks (*payload*) ning seda ei saa paketi edastamisel muuta. Terviklust kontrollitakse kogu (algse) paketi ulatuses. ESP korral krüptitakse kogu pakett – pealtkuulaja ei näe algse paketi lasti ega ka päist.

Soovitame kasutada ESP-protokolli tunnelirežiimis.

AH ja ESP kasutatavad krüptoalgoritmid, parameetrid ja võtmed määratakse turvaühendiga (SA) ning need lepitakse kokku IKEt kasutades. Võtmete käsitsi konfigureerimine ilma IKEt kasutamata on tehniliselt võimalik, kuid seda ei peaks lubama, kuivõrd seda on keerulisem hallata ning siis jäävad pooled ilma võtmevärskendamise võimalusest.

IKE protokolli esimese sammuna lepivad IKE pooled kokku IKE turvaühendis ja teise sammuna autendivad üksteist. Esimeses sammus tuletatakse DH-kehtestatud saladusest ja vahetatud nonssidest põhisaladus, millest tuletatakse edaspidi kõik kasutatavad võtmed. IKE turvaühendis kokkulepitud algoritme kasutatakse IKE liikluse (sh ka autentimissõnumite) kaitsmiseks ja vastastikuseks autentimiseks.

Autentimiseks tuletatakse esimestest vahetatud pakettidest ja kokkulepitud saladustest autentimissõnumid, mille kumbki pool peab signeerima. Jagatud saladuse (parooliga) autentimisel kasutatakse sõnumiautentimiskoodi, mille sisenditeks on tuletatud autentimissõnum ja paroolist tuletatud väärtus. IKEv2 jagatud saladusega autentimisprotokoll ei sisalda meetmeid parooli jõuga murdmise vastu. Kasutaja peab tagama parooli piisava keerukuse. Alternatiivselt kasutatav EAP autentimisprotokoll sisaldab meetmeid ka jõuründe vastu.

Pärast IKE turvaühendi kokkuleppimist loovad pooled vähemalt ühe alam-turvaühendi (*child security association*) konkreetsete liikluse-selektorite jaoks. Liikluse-selektor (*traffic selector*) on IP-aadresside vahemik, IP-protokoll ja pordivahemik, millele turvaühendiga määratud kaitset rakendatakse.

Olemasoleva IKE turvaühendi kontekstis saavad pooled luua uusi turvaühendeid ja juba

kokkulepitud turvaühendite võtmeid värskendada (*rekey*). Mõnede šifrite töörežiimide korral on sama võtmega krüptitav andmemaht piiratud (nt AES-GCMi korral on see umbes 64 GB) ja nõutav on võtmete regulaarne uuendamine. Mõned IPseci teostused ei uuenda võtmeid pärast ettenähtud andmemahu täitumist. Sellisel juhul peab kasutama šifreid, millel mahupiirangut pole või lühendama võtmevärskenduse perioodi nii, et võti uuendataks enne, kui kriitiline andmemaht täituda saab.

Kehtivad IKEv2 standardid ei käsitle postkvant-krüptograafia kasutamist. BSI 2026. aastal avaldatud IPsec soovituste dokument ei kirjelda IKEv2 jaoks postkvant-krüptograafia kasutamist, kuna vastav standardimisprotsess on veel pooleli [111]. BSI soovitab kasutada võtmekehtestuse jaoks hübriidmehhanisme, klassikalisi võtmekehtestusmehhanisme lubab BSI kasutada kuni 2031. aasta lõpuni.

3.3.2 Analüüsid ja ründed

IPseci protokollistiku turvaanalüüse oleme kirjeldanud varasemas aruandes [2], kus leidsime, et protokolliloogikas nõrkusi ei ole. IPseci teostuste Libreswan ja strongSwan mõningate versioonide analüüsid (kasutades kas hägutestimist või olekumasina õppimist) näitavad aga, et puudu on teatud kontrollid serverinimede või protokollisõnumipäiste parsimisel [116, 117]. Värskem artikkel [118], kus samuti rakendati kuuel erineval IPSeci realisatsioonil hägustestimist, näitab, et need teostused töötlevad mõningaid vigaseid teateid leebemalt kui spetsifikatsioon ette näeb. Pisut varasem artikkel [119] kirjeldab, kuidas nii mõneski IKEv1 realisatsioonis leidub nn Bleichenbacheri oraakel (vt ptk 3.1.2). Samuti näitab see artikkel, et nii IKEv1 kui ka IKEv2 lubavad eeljagatud võtmetele rakendada jõurünnet (vallasrežiimis), seega on tarvis eeljagatud võtmete kasutamise korral (kui paralleelselt ei ole mingit muud võtmevahetusmeetodit kasutatud) hoolitseda selle eest, et need võtmed oleksid suure entroopiaga.

3.3.3 Soovitused

IKE on protokoll turvaühendite (SA) kokkuleppimiseks. Kõik kokkulepitavad algoritmid on nummerdatud, vastavaid registreid haldab IETF. Allpool viidatakse lubatud algoritmidele IETFi registrites näidatud nimede ja numbritega. Meie soovitused algoritmide valikuks või mittevalikuks lähtuvad krüptoprimitiividele peatükis 2 esitatud soovitustest (tabel 4). Järgnevad soovitused kattuvad suures osas BSI omadega[34].

Kõik IKE abil kokkulepitavad võtmed tuletatakse pseudojuhusliku funktsiooni (*Pseudo Random Function, PRF*) abil esimeste sõnumitega kehtestatud Diffie-Hellmani saladusest ja vahetatud nonssidest. See tähendab, et turvalisus sõltub oluliselt DH-parameetritest ja kehtestatud saladuse turvatasemest.

3.3.3.1 DH-rühmad

DH-rühma abil kehtestatud saladust kasutatakse peasaladuse tuletamisel (IKE turvaühendis) ja võidakse kasutada lisasisendina alamturvaühendite (*child* SA) loomisel.

Lubatud rühmad[35] on esitatud järgnevas loetelus. Kirjete juures sulgudes olev arv on rühma

[34] BSI avaldas 2026. aasta jaanuaris uue versiooni juhendist, mis annab soovitusi IPSec ja IKEv2 kasutamise ja seadistamise kohta [111].

[35] DH rühmade register: https://www.iana.org/assignments/ikev2-parameters/ikev2-parameters.xhtml#ikev2-parameters-8

identifikaator IETFi registris.

- 3072-bitine MODP-rühm (15)
- 4096-bitine MODP-rühm (16)
- 256-bitine juhuslik ECP-rühm (19). See on IETFi nimi ja identifikaator rühmale NIST P-256.
- 384-bitine juhuslik ECP-rühm (20). See on IETFi nimi ja identifikaator rühmale NIST P-384.
- 521-bitine juhuslik ECP-rühm (21). See on IETFi nimi ja identifikaator rühmale NIST P-521.
- brainpoolP256r1 (28)
- brainpoolP384r1 (29)
- brainpoolP512r1 (30)

Soovitame kasutada ECDHd koos Brainpooli elliptkõveraga, viimasel peab olema sobiv turvatase[36].

Alamturvaühendite saladused (kasutatavad võtmed) tuletatakse peasaladusest ja turvaühendi loomisel vahetatud uutest nonssidest. Protokoll lubab ka (valikulist) uut DH-võtmekehtestust, mille abil tekitatakse lisasaladus alamturvaühendi saladuse tuletamiseks. Kui kasutatav IPseci teostus võimaldab sellist täiendavat võtmekehtestust, siis soovitame seda kasutada. Kasutama peaks sama või tugevamat DH-rühma kui see, mida kasutati algse (IKE-)turvaühendi kokkuleppimisel.

Ilma täiendava saladuseta sõltub alamturvaühendi saladus ainult peasaladusest ning peasaladuse murdumisel on kõik alamsaladused võimalik tuletada pealtkuulatud sõnumitest – täiendav DH-võtmekehtestus annab alamturvaühenditele tulevikuturvalisuse.

3.3.3.2 Pseudojuhuslik funktsioon

Pseudojuhuslikku funktsiooni kasutatakse Diffie-Hellmani protokolliga kehtestatud jagatud saladusest krüptovõtmete pärimiseks.

Lubatud funktsioonid[37] on:

- `PRF_AES128_XCBC` (4)
- `PRF_AES128_CMAC` (8)
- `PRF_HMAC_SHA2_256` (5)
- `PRF_HMAC_SHA2_384` (6)
- `PRF_HMAC_SHA2_512` (7)

Pseudojuhusliku funktsiooni väljundi pikkus peab olema vähemalt sama pikk kui on kasutatava šifri võti. See tähendab, et 256-bitise võtmega šifreid kasutades ei tohi kasutada AES128-põhiseid funktsioone IETFi registrinumbritega 4 ja 8.

[36] Meie soovitusel eelistada Brainpooli NIST-i elliptkõverarühmadele pole väga tugevat põhjendust, kuid see soovitus kordab meie varasemaid soovitusi [7, 8], kui Brainpooli kõveraid soovitas (ja soovitab praegugi) BSI [111], samas kui NIST-i mainet mõjutas veel Dual_EC_DRBG skandaal [120].

[37] Pseudojuhuslike funktsioonide register: `https://www.iana.org/assignments/ikev2-parameters/ikev2-parameters.xhtml#ikev2-parameters-6`

3.3.3.3 Tervikluse kontrolli algoritmid

Tervikluse kontrolli algoritme[38] kasutatakse IKE-, AH- ja ESP-protokollides sõnumi tervikluse tagamiseks. Nende kasutamine IKE ja ESP korral on vajalik vaid siis, kui kasutatav šiffer ise terviklust ei taga.

Lubatud sõnumiautentimise algoritmid on:

- `AUTH_AES_XCBC_96` (5)
- `AUTH_HMAC_SHA2_256_128` (12)
- `AUTH_HMAC_SHA2_384_192` (13)
- `AUTH_HMAC_SHA2_512_256` (14)

3.3.3.4 Šifrid

Šifreid kasutatakse protokollides IKE ja ESP lasti (*payload*) krüptimiseks (konfidentsiaalsuse saavutamiseks). Mõnedel šifritel/töörežiimidel on tervikluse tagamise omadus.

Lubatud šifrid[39] on:

- `ENCR_AES_CBC` (12) (koos sellega peab kasutama eraldi tervikluse tagamise algoritmi!)
- `ENCR_AES_CTR` (13) (koos sellega peab kasutama eraldi tervikluse tagamise algoritmi!)
- `ENCR_AES_GCM_16` (20)
- `ENCR_AES_GCM_12` (19)
- `ENCR_AES_CCM_16` (16)
- `ENCR_AES_CCM_12` (15)
- `ENCR_CHACHA20_POLY1305` (28)

Kõigi loetletud šifritega võib kasutada võtmeid pikkusega 128 ja 256 bitti (vastavalt šifri võimalustele).

3.3.3.5 Autentimisalgoritmid

Autentimisalgoritme kasutatakse IKE-protokollis tõestusena, et identiteediväidet esitav pool teab mingit jagatud saladust või deklareeritud avalikule võtmele vastavat privaatvõtit.

Lubatud algoritmid[40] on:

- ECDSA kõveral secp256r1 (NIST P-256), SHA-256 (9)
- ECDSA kõveral secp384r1 (NIST P-384), SHA-384 (10)
- ECDSA kõveral secp521r1 (NIST P-521), SHA-512 (11)
- ECDSA kõveral brainpoolp256r1, SHA-256 (14)
- ECDSA kõveral brainpoolp384r1, SHA-384 (14)

[38] Tervikluse kontrolli algoritmide register: https://www.iana.org/assignments/ikev2-parameters/ikev2-parameters.xhtml#ikev2-parameters-7

[39] Šifrite register: https://www.iana.org/assignments/ikev2-parameters/ikev2-parameters.xhtml#ikev2-parameters-5

[40] https://www.iana.org/assignments/ikev2-parameters/ikev2-parameters.xhtml#ikev2-parameters-12

- ECDSA kõveral brainpoolp521r1, SHA-512 (14)
- RSASSA-PSS, kasutades 4096-bitist RSA-võtit, SHA-384 (14)

3.3.3.6 Turvaühendi eluiga

IKE-turvaühendi maksimaalne soovituslik eluiga on 24 tundi ja alamturvaühendi eluiga maksimaalselt 4 tundi [111]. Arvestama peab ka kasutatavast šifrist tingitud võimalikke andmemahupiiranguid.

3.4 Virtuaalsed privaatvõrgud

Virtuaalne privaatvõrk (VPN) on mingi organisatsiooni arvutivõrk, mis on esiteks privaatne, st välised pooled ei näe selles võrgus liikuvaid andmed, ja teiseks kasutab avalikku võrgutaristut, näiteks internetti. Privaatsus saavutatakse krüptograafiliste meetodite abil, võrguliiklust tööjaamade ja serverite (mis võivad füüsiliselt paikneda väga erinevates kohtades) vahel jooksvalt krüptides ja dekrüptides. Võrguliiklust, aga ka -topoloogiat ja võrku kuuluvate tööjaamade asukohti varjates on VPN oluline privaatsustehnoloogia.

VPN-teenused jagunevad mitmesse eri kategooriasse: on teenused, mis võimaldavad kaugligipääsu ettevõtte sisevõrgule (kaugjuurdepääsu VPN), on teenuseid, mille eesmärk on ühendada omavahel mitme erineva asutuse sisevõrgud (saitidevaheline VPN) ning mobiilne VPN, mille eesmärk on võimaldada VPN-ühendust seadmele, mis vahetab regulaarselt WiFi-võrke või mille puhul pole garanteeritud pidev internetiühendus. Samuti saab VPN-teenuseid eristada protokollide järgi, millel nad tuginevad.

3.4.1 OpenVPN

OpenVPN on üks populaarsemaid avatud lähtekoodiga VPN-protokolle, mille esimene versioon ilmus juba aastal 2001. Tänaseks on teostusi muuhulgas Windowsi, macOSi, Linuxi, OpenBSD, FreeBSD, Androidi ja iOSi platvormide jaoks.

OpenVPNis ei ole eraldi krüptoalgoritmide teostusi, vaid selles kasutatakse meetodeid OpenSSL krüptoteegist. Seetõttu on võimalik OpenVPNi seadistada väga mitmel erineval moel ning erinevate primitiivide jaoks on palju erinevaid valikuvariante. OpenVPN kasutab võtmevahetuseks vaikimisi TLS/SSL-protokolle, kuid ka seda on võimalik soovile vastavalt seadistada. Siiski tuleb panna tähele, et OpenSSList sõltuvuse tõttu pärib OpenVPN kõik OpenSSList pärinevad turvanõrkused[41].

OpenVPNi saab kasutada kahes režiimis: kas kasutades TCP (Transmission Control Protocol) või UDP (User Datagram Protocol) protokolli. TCP-protokoll garanteerib andmepakettide terviklikkuse ning õiges järjekorras saabumise, UDP loobub nendest omadustest, et saavutada kiirem ühendus. Vaikimisi kasutab OpenVPN TCP-režiimi, kuid jääb ka UDP-režiimis uuematele VPN-protokollidele kiiruse osas alla[42].

OpenVPN on kõrge küpsusastmega privaatvõrguprotokoll, mistõttu on seda võimalik kasutada paljudes eri seadistustes ning olukordades, nii kaugligipääsu loomiseks kui ka erinevate sisevõrkude ühendamiseks, kuigi mobiilsetel seadmetel soovitatakse kasutada teisi, kiiremaid protokolle. Samuti tasub meeles hoida, et kui seadistada OpenVPNi ise, tekib oht seda valesti üles

[41] https://www.techradar.com/vpn/what-is-openvpn
[42] https://nordvpn.com/blog/protocols/

seades turvanõrkuseid tekitada, sest seadistusprotsess nõuab nii aega kui ka pädevust. Paljudel juhtudel on võimalik seda riski maandada mõne VPN-teenuse kasutamisega[43].

3.4.2 WireGuard

WireGuard[44] on OpenVPNist värskem VPN-protokoll, mis sisaldub alates 2020. aasta märtsi lõpust Linuxi tuumas (tuuma versioon 5.6). Sarnaselt OpenVPNiga leidub WireGuardi teostusi mitmel platvormidel sh Android, Windows, macOS, iOS, FreeBSD ja OpenBSD.

Kuigi ühest küljest kannab WireGuardi protokolli dokumentatsioon[45] märkust „*Draft revision*“, on protokoll siiski formaalselt verifitseeritud [121][46]. Sellele esialgsele kõrgetasemelisele analüüsile on nüüd järgnenud arvukalt põhjalikumaid analüüse [122, 123, 124], kusjuures on kasutatud protokollide automaatverifitseerimise vahendite abi. Olulisi nõrkusi ei ole avastatud. Tehtud on ettepanekuid turvalisuse tugevdamiseks. Ühes hiljutises analüüsis [124] hoiatatakse, et üks võimalik optimeerimine (seotud efemeervõtmete eelgenereerimisega) suurendab oluliselt ründepinda.

Kas WireGuardi kasutada või mitte, oleneb tugevalt ümbritsevast keskkonnast, võrgutopoloogiast ja kättesaadavast oskusteabest – WireGuard on projekteeritud ennekõike kakspunkt-võrgutopoloogias kasutamise jaoks ning keerulisemate võrkude ülesseadmine võib olla komplitseeritud. Ei tohi ka unustada, et konservatiivsemate organisatsioonide jaoks ei ole värske eesliinitehnoloogia peaaegu kunagi hea valik. Kui OpenVPN kasutab vaikimisi TCP-protokolli, siis WireGuard tugineb UDP-protokollil, mis tagab küll suurema kiiruse, kuid ei taga andmepakettide terviklikkust ning õiget saabumise järjekorda. Tuleb ka märkida, et WireGuard on OpenVPNist kiirem ka siis, kui OpenVPNi kasutatakse UDP-režiimis[47]. Samuti koosneb WireGuard OpenVPNiga võrreldes palju vähematest koodiridadest ning on seetõttu kergemini auditeeritavam ja teostatavam.

WireGuardi krüptoprimitiivide konfiguratsiooni osas ei saa selle aruande kirjutamise ajal soovitusi anda, kuna WireGuardi konfiguratsioon on fikseeritud, aga see sisaldab siiski krüptograafilisi primitiive, mida aruande muudes osades on soovitatud (X25519, ChaCha20, Poly1305). Ühtegi mittesoovitatavat või keelatud primitiivi kasutuses ei ole. Samas eemaldab konfiguratsiooni fikseeritus võimaluse WireGuardi valesti seadistada – risk, mille teostumine on OpenVPNi puhul võrdlemisi reaalne (kuigi on võimalik kasutada ka mõne VPN-teenusepakkuja poolt pakutavat teenust, kus see risk on väiksem).

Teiselt poolt võib vaadelda fikseeritud šifrikonfiguratsiooni ka WireGuardi potentsiaalse nõrkusena. Mõne šifri murdumine ei võimalda sujuvat üleminekut, vaid tarkvara tuleb uuendada mõlemas otspunktis korraga. Suuremate installatsioonide puhul võib see olla problemaatiline. Seda probleemi on mainitud ühes varasemas tulemüüriplatvormi IPFire blogis[48], kus WireGuardi peeti ebaküpseks, kuid nüüdseks on IPFire juba saanud WireGuardi toe [49].

[43] https://nordlayer.com/learn/vpn/openvpn/
[44] https://www.wireguard.com/
[45] https://www.wireguard.com/papers/wireguard.pdf
[46] https://www.wireguard.com/formal-verification/
[47] https://www.wireguard.com/papers/wireguard.pdf
[48] https://blog.ipfire.org/post/why-not-wireguard
[49] https://www.ipfire.org/blog/ipfire-2-29-core-update-195-released-wireguard-inside

3.5 Bluetoothi ja NFC turvaaspektid

Selles alapeatükis vaatame NFC- ja Bluetooth-protokolle digikukrute kontekstis. Nimelt vaatleme kasutusmalli, kus NFCd või Bluetoothi kasutatakse esitaja ja kontrollija lähenduses olevate seadmete vaheliseks suhtlemiseks. NFC kasutamist ID-kaardi kontekstis käsitleb põhjalikumalt 2022. aasta aruanne [125]. Siin me sarnast riskianalüüsi digikukrute kontekstis ei esita, vaid anname üldisema ülevaate võimalikest rünnetest ja nõrkustest, andmata hinnangut eri tüüpi rünnetele. Juhime tähelepanu, et selle alapeatüki käsitlusalast jäävad välja ka ründed konkreetsete *krütoprimitiivide* vastu, mida NFC või Bluetooth võib omakorda kasutada.

3.5.1 NFC

Lähiväljaside (*Near Field Communication, NFC*) on traadita side kokkupuutes või lähedaste seadmete vahel, tavaliselt teineteisest mitte üle 10 cm kaugusel. NFC töötab sagedusel 13,56 MHz ja kiirusega vahemikus 106 kbit/s kuni 424 kbit/s. NFC põhineb raadiosagedustuvastuse (*Radio Frequency IDentification*, RFID) süsteemidel, võimaldades kahesuunalist sidet seadmete vahel. NFC-tehnoloogiat on laialdaselt rakendatud ning see on saadaval mobiiltelefonides, sülearvutites ja paljudes teistes seadmetes.

NFC-seadmed liigitatakse aktiivseteks ja passiivseteks:

- **aktiivne (NFC terminal/lugeja):** toitega seade, mis on võimeline algatama interaktsiooni teiste NFC seadmetega. Aktiivne seade genereerib raadiosagedusvälja, mis toidab passiivset seadet;
- **passiivne (NFC pääsmik/silt):** seade, mida toidab lähedusvahemikku sattunud aktiivne NFC-seade. Selline seade võib võtta väga lihtsaid vorme, nagu näiteks silt, kleebis, võtmehoidja või kaart, vältides vajadust akude järele.

NFC-side kahe seadme vahel on võimalik, kui vähemalt üks neist on aktiivne NFC-seade. Ründaja valduses võib olla mõlemat tüüpi seadmeid. Mõlemad vajavad kaitset.

Järgnevalt loetleme peamisi ründemeetodeid NFC-seadmete vastu, mis on rakendatavad digikukrute kontekstis, kus NFCd kasutatakse esitaja ja kontrollija vaheliseks suhtlemiseks.

Pealtkuulamine, segamine, andmete muutmine. Nendes stsenaariumites sekkub ründaja andmevahetusse NFC-seadmete vahel. Kuna NFC-side toimub vaid üksteise vahetus läheduses asuvate seadmete vahel, on seda tüüpi rünne praktikas keeruline, kuid teostatav [126]. Selliseid ründeid on lähemalt uuritud ID kaardi kontekstis [125], järeldusega, et ID-kaardi kasutamise riskitase NFC-liidese kaudu ei ole oluliselt kõrgem kui ID-kaardi kasutamine USB-liidese kaudu.

Peamine meetod pealtkuulamise ja andmete muutmise vältimiseks on NFC-seadmete vahele turvalise kanali loomine (kõrgemal tasemel), kasutades selleks näiteks krüptograafilisi meetodeid. Segamise vastu krüptograafia ei aita; üks võimalik vastumeede on RF-signaali kontroll andmeedastuse ajal. Kuna segamise võimsus on suurem kui andmete saatmisel kasutatav võimsus, suudab saateseade ründe tuvastada ja andmeedastuse automaatselt peatada.

Vahendusründed, teesklus ja topeltteesklus. Väga lühikese suhtlusvahemiku tõttu on vahendusründed keerulised, sest ründaja peaks asuma rünnatavate seadmete lähedal. Siiski on teesklus võimalik näiteks tarkvarapõhiselt [127] (rünne Google Walleti vastu) või suhtlusrünnete

kaudu[50].

Topeltteesklust ei saa ära hoida rakenduse tasemel krüptograafiaga, sest ründaja võib lihtsalt vahendada sõnumeid kahe ausa poole vahel. Põhilised vastumeetmed topeltteeskluse vältimiseks on:

- NFC-pääsmiku varjestamine kasutaja kõrval olevate elektromagnetväljade eest;
- lisatoimingud NFC-pääsmiku omaniku poolt (tehingu kinnitamine), kaksikautentimine (*two-factor authentication*);
- RFID-süsteemi kaugpiiristamise protokollide (*distance bounding protocols*) kasutuselevõtt. NFC-lugeja saab teada, kas NFC-pääsmik on piisavalt lähedal. See võib aidata ka siis, kui ründajal õnnestub veenda kasutajat NFC-pääsmikku lahti lukustama.

3.5.2 Bluetooth

Bluetooth on standardil IEEE 802.15.1 põhinev traadita võrgu protokoll ja meetod andmevahetuseks (kuni 100m, siseruumides umbes 10m). See töötab 2,4GHz ISM-sagedusalal (*Industrial, Scientific and Medical*). Bluetooth Classic toetab kiirust kuni umbes 24 Mb/s. Väikese energiatarbega Bluetooth LE (*Low Energy*) kiiruse piir on umbes 1 Mb/s, kuid energiakulu on väiksem. Siin aruandes pöörame tähelepanu Bluetooth LEle kui digikukrutes kasutatavale tehnoloogiale.

Bluetooth LE (BLE) on suunatud seadmetele, mis töötavad väga väikese võimsusega (mündi)patareide või minimalistlike toiteallikatega, mis tähendab, et väike energiatarbimine on hädavajalik. Bluetooth LE on eriti atraktiivne seadmevõrgu rakenduste jaoks, nagu tervisemonitorid, keskkonnaseire, lähedusrakendused ja paljud teised. Järgnev kirjeldus pärineb Bluetooth Core'i spetsifikatsioonist [51], mis on avalikult tasuta saadaval.

Käesolevas dokumendis käsitleme järgmisi BLEs määratletud rolle:

- **välisseade (alluv)** (*Peripheral*) saadab kuulutusi läheduses asuvatele seadmetele ja ootab ühendusi;
- **kekseade (ülem)** (*Central*) võtab vastu välisseadmete kuulutusi ja võib mõnega neist ühenduse luua. Ülem võib ühenduda mitme alluvaga. Moodustuvat juhuvõrku kutsutakse pikovõrguks.

Seadmed teavitavad üksteist oma olemasolust, saates pakette kuulutuskanalite kaudu (*advertising channels*, kindlat tüüpi füüsilised kanalid). Ühenduse algataja võib esitada kuulutajale (*advertiser*) ühenduse taotluse. Ühendus algab, kui kuulutaja võtab taotluse vastu. Kui ühendus on loodud, saab algatajast kekseade ja kuulutajast väliseade. Ühendussündmusi kasutatakse andmepakettide saatmiseks kesk- ja väliseadmete vahel. Kekseade algatab iga ühendussündmuse ja võib selle igal ajal lõpetada.

Aruande kirjutamise hetkel on Bluetooth-seadmete vaheliseks suhtlemiseks kasutusel olnud mitu lahendust, soovitatav on uusim neist (Secure Connections). See kasutab järgmiseid krüptoprimitiive: [52]

- krüptimine: AES-CCM

[50] https://www.kaspersky.com/blog/apple-google-nfc-carding-theft-2025/53267/
[51] https://www.bluetooth.com/wp-content/uploads/Files/Specification/HTML/Core-61/out/en/index-en.html
[52] https://www.bluetooth.com/wp-content/uploads/Files/Specification/HTML/Core-54/out/en/br-edr-controller/security-specification.html

- autentimine: HMAC-SHA256
- võtme genereerimine: P-256 ECDH ja HMAC-SHA-256

Järgnevalt käsitleme erinevaid ründeid Bluetooth LE vastu, mis põhinevad hiljutistel ülevaateartiklitel [128, 129]. Kuigi Bluetoothiga seotud ründeid nimetatakse sageli *Blue*-millekski, kasutame siin sarnaselt NFCle üldisemat ründeklassifikatsiooni.

Ehkki ründed võivad olla väga erinevad, kuulub oluliste ohutusmeetmete hulka Bluetooth-funktsiooni väljalülitamine selle mittekasutamise ajal ja tundmatu seadmega mittesidumine. Olemas on ka täiendavad turbevõimalused, nagu BLE-Guardian, mis võimaldab kasutajal kontrollida, kes saab teda avastada, skaneerida ja ühendada oma seadmega.

Jälitus. Mainime seda rünnet enne pealtkuulamist, kuivõrd see rünne ei pruugi (veel) anda ligipääsu rünnatavate seadmete vahetatavatele andmetele. Siiski suudab ründaja koguda teavet seadme omaduste kohta, nagu MAC-aadress, seadme mudel, kordumatu identifikaator jne (*BluePrinting*). Selline info võib identifitseerida seadme üheselt. Sellised ründed rikuvad kasutajate privaatsust (nt võimaldavad jälgida seadme liikumist) ja lihtsustavad mõningate seadmete teadaolevate nõrkuste ärakasutamist (*BlueStumbling*).

BLE-protokolli piirangute tõttu ei ole sellele probleemile head lahendust. Seadmeid tuleks hoida avastamist mittevõimaldavas režiimis, kui BLE-ühendust ei ole vaja.

Pealtkuulamine. Juhtmeta side on pealtkuulamisele eriti avatud, sest andmeid edastatakse läbi õhu ja ründaja vajab ainult seadet raadioside salvestamiseks. Kuigi ühendus on krüptitud, ei ole vanemad BLE paaristusmeetodid nagu *Just Connect* piisavalt tugevad, et seda rünnet vältida. Näiteks on võimalikud jõuründed PIN-koodi vastu (*PIN cracking*). Võtmevahetuse ja autentimise pealtkuulamine võimaldab paljastada kasutaja pikaajalise võtme.

Vastumeetmena tuleks kasutada tugevamaid paaristusmeetodeid, nagu *BLE Secure*. Juhime tähelepanu, et kuna BLE Secure põhineb Diffie-Hellmani võtmevahetusel, ei ole see kvantturvaline.

Segamine. Ründaja võib saata ohvrile lühikese aja jooksul suure hulga pakette, põhjustades kliendi puhvrite ületäitumise (*BlueSmacking*). Samuti võib pakettide sees saata müra, mis võib põhjustada BLE-seadme krahhi või väärkäitumist. Ründaja võib saata seadmele sagedasi päringuid või sundida seda tegema raskeid arvutusi. See pidev tegevus tühjendab kiiresti seadme aku. Tagajärjed on teenusetõkkest kuni ettenägematu käitumiseni.

Arendajad peaksid läbi viima saadud andmete nõuetekohase valideerimise. Selliseid ründeid on siiski raske tuvastada, sest ründajate taotlused võivad olla legitiimsetest eristamatud ja pahatahtlikud andmed saab korrektselt vormindada. Akuseire võib tuvastada ühendusi, mis kulutavad liiga palju energiat.

Teesklus. Teesklusründes (*spoofing*) teeskleb ründaja, et ta on mõni teine seade. Ründaja võib muuta oma MAC-aadressi ja esitada ennast kui õiget seadet. Kui kasutaja seade lubab suvaliste ühenduste vastuvõtmist, saab ründaja seade kasutaja seadmele saata suvalise kuulutussõnumi (*BlueJacking*), mis ei pruugi olla mitte ainult tüütu ja segav, vaid võib meelitada kasutaja paaristusse. Pärast ohvri paaristusse meelitamist võib ründaja kasutada teadaolevaid nõrkusi, et saada andmeid ohvri seadmest (*BlueSnarfing*) või paigaldada tagauks seadme täielikult enda kontrolli alla võtmiseks (*BlueBugging*).

Kui kaks seadet on juba paaristatud ja ründaja ei suuda pikaajalist võtit ära arvata, võib see petta seadmeid ühenduse loomist uuesti läbi tegema. Ründaja saab nüüd kasutada MAC-aadressi teesklust või kasutada paaristusprotokolli nõrkusi.

Vastumeetmeks ei tohiks kasutaja aktsepteerida meelevaldseid ühendusi. Ettevaatlik tuleks olla isegi pealtnäha legitiimsete ühendustega. Mustade ja valgete nimekirjade kasutamine võib aidata MAC-aadressi teeskluse ja ebaseaduslike uute paaristumistaotluste vastu.

Vahendusründed. Paigutades ennast Kesk- ja Välisseadmete keskele, võib ründaja nende vastu lihtsasti vahendusründe korraldada. **Taasesitusründe** (*replay attack*) puhul ei pea ründaja isegi sõnumite sisu lugema ega muutma. Nagu NFC, on ka Bluetooth vastuvõtlik **topeltteesklusele** [129].

Turvaline ühendus ja tugevamad sidumismeetodid nagu BLE Secure on olulised, et takistada ründajal vahetatud sõnumeid muuta, kuid sellest ei piisa. Taasesitusrünnete vältimiseks tuleks kasutada autentimismehhanisme (nonss, seansivõti või ajatempel). Topeltteeskluse vältimiseks võib analüüsida sõnumite ajalisi viivitusi. Samuti on võimalik lisada sõnumitesse mõningaid konkreetse pikovõrgu spetsiifilisi andmeid, kuid juhul kui ründaja tegutseb kesksеadmena ühes pikovõrgus, saab ta jõustada parameetreid, mis pärinevad teisest pikovõrgust, kus ründaja on väliseade. Ainus erand on see, kui need kaks pikovõrku on üksteise lähedal ja täpselt samade parameetrite kasutamine põhjustaks suhtlemishäireid.

4 Krüptokonteinerite vormingud

4.1 Andmete kodeerimise viisid

4.1.1 Kodeerimine biti- või baidijadana

Base64 [130] on üks lihtsamaid andmete kodeerimisviise, mis väljendab 6 bitti infot ühe ASCII sümbolina. Selleks kasutatakse 65 erinevast ASCII sümbolist koosnevat tähestikku, millest üht kasutatakse täidistamiseks. See võimaldab teisendada 3 baiti ehk 24 bitti infot 4 sümboliga. Levinud on sarnased base64 ja base64url tähestikud, millest teine ei kasuta URLide tähenduses erisümboleid. HTTP valdkonnas on kasutusel ka token68 [131] sümbolite hulk, mis sisaldab endas base64 ja base64 sümbolite ühendit koos kahe URIdes kasutatava kirjavahemärgiga. Token68 pole aga kodeerimisskeem, kuivõrd sel puuduvad reeglid sümbolite ja bittide vastavuse kohta.

Distinguished Encoding Rules (DER) [132] on ASN.1-andmestruktuuride kodeerimisformaat mis tagab, et iga väärtust on võimalik ainult üheselt kodeerida. DER kodeerib iga väärtuse *Tag-Length-Value* kolmikuna, mille esimene liige on tüüpi märkiv silt, teine liige on väärtuse pikkus ning kolmas liige on väärtus ise. DER on laialdaselt kasutusel X.509-serifikaatides ja võtmetes.

Concise Binary Object Representation (CBOR) [133] on JSONi-laadne kahendandmete jadastusformaat. Sarnaselt JSONile sisaldab CBOR võtme ja väärtuse paare, kuid seda lühemal ning kiiremini masinloetaval kujul. CBOR sobib ka hästi voollugemiseks (*streaming*).

4.1.2 Kodeerimine struktureeritud tekstina

JSON [134] ja XML [135] on levinumad viisid andmete kodeerimiseks struktureeritud tekstina. W3C soovitus JSON-LD [136] lisab JSONile linkandmete esitamise toe ning on leidnud kasutust digikukrute valdkonnas[1].

4.2 Allkirjavormingud

Euroopa Liidu e-identimise ja e-tehingute määrus eIDAS [137] defineerib elektroonilise allkirja neli taset, kõrgemaist madalamani:

- kvalifitseeritud elektrooniline allkiri (QES);
- elektroonilise allkirja kvalifitseeritud sertifikaadil põhinev täiustatud elektrooniline allkiri (AdES/QC);
- täiustatud elektrooniline allkiri (AdES);
- muud elektroonilised allkirjad.

E-identimise ja e-tehingute usaldusteenuste seadus (EUTS) [138] võrdsustab eestikeelse mõiste „digiallkiri“ eIDASes määratud kvalifitseeritud elektroonilist allkirjaga (QES). Muul juhul tuleb kasutada terminit „elektrooniline allkiri“. Nii EUTS kui ka eIDAS võrdsustavad QES-taseme allkirja omakäelise allkirjaga. Rohkem infot eIDASe rakendamisest Eestis leiab Eesti e-identiteedi ökosüsteemi ülevaatest.[2]

[1] https://www.w3.org/TR/vc-overview/

[2] https://www.ria.ee/sites/default/files/documents/2025-10/eID-okosusteem-lisadega-2025.pdf

Täiustatud elektrooniline allkiri hõlmab endast järgnevaid välju:

- Allkirjastatav dokument või selle räsi
- Allkirjastatud atribuudid
- Allkirja väärtus
- Allkirjastamata atribuudid (valikuline)

ETSI standard EN 319 102-1 [139] defineerib neli tõeväärtuse profiili (Basic, Timestamped, Long-Term, Long-Term Archival), mis määravad allkirja vajalikud elemendid. Profiil Basic on baastase, Timestamped nõuab ajatempli olemasolu, tasemed Long-Term ja Long-Term Archival nõuavad atribuute allkirja pikaajaliseks valideerimiseks. Allkirja valideerimishorisont ei ole ilma edaspidiste hooldustegevusteta küll piiramatu; valideerimiseks on muuhulgas vaja, et mõni ajatemplitest veel kehtiks (või signeerimiseks kasutatav krüptoalgoritm oleks ikka veel turvaline).

Eestis loodavad digiallkirjad on vaikimisi Long-Term tõeväärtuse profiiliga; allkirjastamata atribuutide seas on seetõttu ajatempel ning kehtivusinfo ja täielik sertifikaat nii allkirja kui ka ajatempli kohta. Atribuutide täpsed väärtused olenevad valitud vormingust, millest Eestis levinum on XAdES.

4.3 Krüptograafiliste objektide kodeerimine

4.3.1 CMS

Cryptographic Message Syntax (CMS) [140] on ASN.1 ja PKCS#7 põhine vorming, mida kasutakse ka standardsete CAdES [141] ja PAdES [142] signatuuride alusena. Probleemiks on vähene tuntus ning ASN.1-kodeeringu keerukus koos sellest omakorda tulenevate turvaprobleemidega. Samuti on CMSi vanem versioon [143] avatud asendusrünnetele [140, *Abstract*], mis on nüüdseks teekides parandatud. CMSi alaspetsifitseerimisest tulenevate nõrkuste vältimise hea tava standard on mustandi faasis [144].

CMSi kasutamine rakenduste ehitusprimitiivina ei ole otseselt ebasoovitav, kuid arendaja peaks hoolikalt jälgima, kas tal on olemas kogu vajalik kogemus ja oskusteave niivõrd keerulise vormingu korrektseks rakendamiseks.

4.4 JSON Object Signing and Encryption

JSON Object Signing and Encryption (JOSE) [145] on üldnimetaja JSONi-põhiste krüptograafiliste sõnumivormingute perekonnale[3] – JWS, JWE ja JWT.

Kuigi JOSE vorminguperekond on moodne ja lihtsasti kasutatav, on tema suureks puuduseks (JSONist pärinev) skeemikirjeldusvahendite puudumine, mis sunnib igal platvormil programmeerijat väga hoolikalt läbi mõtlema, kuidas JOSE-struktuure turvaliselt parsida ning töödelda. Samuti tekib probleeme ühilduvusega, kui on vaja vahetada JOSE-struktuure erinevate platvormide vahel. On teada, et olemasolevad teegid ei taga kõiki vajalikke turvakontrolle, võimaldades näiteks asendusründeid konteinerite metaandmete vastu. Teiseks puuduseks on JOSE standardiga määratud krüptoalgoritmide suletud nimekiri, mis ei võimalda JOSE-struktuure kasutada standardiga määratud registris[4] loetlemata algoritmidega (näiteks lävikrüptograafiat kasutades). Määratud krüptoalgoritmide hulgas on ka valikud kasutada mitte mingit krüptimist või valitud krüptogrammiga rünnaku suhtes ebaturvalist RSA PKCS#1 v1.5 [80].

[3] Hoolimata sellest, et vormingu aluseks on JSON, võib kaitstav avatekst olla ka muule süntaksile vastav baidijada.
[4] https://www.iana.org/assignments/jose/jose.xhtml

JOSE kasutamine (väljaspool avalikke standardeid, mis seda ette kirjutavad) ei ole otseselt ebasoovitav, kuid tuleb arvestada, et tegu on keerulise ja kõrge riskiastmega vorminguga, mille korrektseks rakendamiseks on vaja kaasata arendustegevusse turvainsener, kes koostab konkreetse rakenduse jaoks sobivad profiilid ning töötlusreeglid.

Riiklikes infosüsteemides kasutamisel (eriti kehtib see *OpenID Connecti* ja JWT kohta) soovitame killustumise vähendamiseks asutustevahelist kooskõlastamist ja profiilide koostamist.

4.4.1 JSON Web Signature

JSON Web Signature (JWS) [146] on vorming andmete tervikluse krüptograafiliseks kaitsmiseks kas digitaalsignatuuri või autentimiskoodi abil.

Oluline on märkida, et JWS ei kuulu eIDASe poolt tunnustatud allkirjavormingute hulka.

4.4.2 JAdES

JAdES on JSON-formaadis AdES-signatuur, mis põhineb JSON Web Signature vormingul. JAdESi vormingut kirjeldab standard ETSI TS 119 182-1 [147].

JAdES täiendab JWSi vormingut täiendavate signeeritud ja signeerimata atribuutidega. JAdESi signeeritud atribuutide hulgas on signeerimise aeg, sertifikaadi lühend, signeeritavate andmete ajatempel ja signeerija atribuutid. Need täiendavad atribuudid kaitsevad mitmete rünnete nagu näiteks sertifikaadiasendusründe eest [148].

JAdES-signatuure saab luua näiteks Digital Signature Service teegiga[5].

4.4.3 JSON Web Encryption

JSON Web Encryption (JWE) [149] on vorming andmete krüptimiseks (sh autentiva krüptimise abil).

4.4.4 JSON Web Token

JSON Web Token (JWT) [150] annab standardse viisi väidete (*claims*) edastamiseks kahe poole vahel JWS- või JWE-kujul ning määrab kindlaks mõned standardsed väited.

Erinevalt JWS ja JWE standarditest määrab JWT ära avateksti struktuuri (JSON-objekt) ja jadastusreeglid, samuti reeglid vormingu lõppkujule, et seda oleks võimalik kasutada piiratud mahu ja tähestikuga vormingutes – näiteks HTTP URL-osisena.

Kõige tuntum ja levinum JWT standardrakendus on OpenID Connect protokollistik [151].

4.4.5 JSON Web Key

JSON Web Key (JWK) [152] on standard ühe krüptograafilise võtme hoidmiseks JSON-formaadis. *JSON Web Key Set* (JWKS) on standard mitme võtme hoidmiseks JSON-formaadis, mis koosneb ühest JWK standardis olevate võtmete massiivist.

[5] https://ec.europa.eu/digital-building-blocks/wikis/display/DIGITAL/Digital+Signature+Service+-++DSS

4.4.6 COSE

COSE [153] on JOSE-laadne vorminguperekond, mis kasutab andmete kodeerimiseks JSONi asemel CBORi. COSE põhineb küll JOSEl [154], kuid nende detailides leidub piisavalt erinevusi, et neid tuleks lugeda sarnasteks vorminguperekondadeks, mitte samaks. CB-AdES [155] on CBOR-formaadis AdES-signatuur mis põhineb COSEl ning on CBORi analoog JAdESile. CB-AdES signatuuride kasutamist näeb ette ETSI standard atribuutide mDoc-vormingus (alampeatükk 4.6.2) elektroonilise atesteerimise kohta [156], kuid eIDASe allkirjastamisvormingute seast neid signatuure ei leia[6].

4.4.7 XMLi-põhised vormingud

XML Encryption [157] ja *XML Signature* [158] on W3C soovitused XML-vormingus krüptitud andmete ja elektrooniliste allkirjade hoidmiseks. XML Signature pakub mitut viisi kuidas allkirjastatud andmeid ja allkirja hoida:

- *Enveloped* XML-allkirjade puhul allkirjastatakse alati XML dokumenti, mille allkiri pakendatakse XML elemendina ta sisse. Allkirja XML-elementi ei allkirjastata, selle välja jätmiseks kasutatakse vastavat XML-teisendust. Sellist pakendamist kasutab näiteks SAML 2.0.
- *Enveloping* XML-allkirjade puhul pakendatakse allkirjastatavad andmed allkirja XML-elementi. Sellised allkirjad näevad vähe kasutust.
- *Detached* XML-allkirjade puhul pakendatakse allkirjastatavad andmed ja allkiri eraldi failidena. Sellised allkirjad on Eestis laialdaselt kasutusel, kus allkirjastatud failid pannakse ASiC-E konteinerisse koos XAdES-allkirjaga, milles iga allkirjastatud faili kohta lisatakse URI-viide konteineris ning faili räsi.

XAdES [159] on XML-formaadis AdES-signatuur, mis põhineb XML Signature vormingul ning laiendab seda AdESi atribuutidega.

4.5 ASiC konteinervorming

Associated Signature Container (ASiC) on ETSI standarditud [160] konteinervorming andmefailide seostamiseks signatuuride või ajatemplitega. ASiC oma erinevates variatsioonides ei ole midagi muud kui etteantud struktuuriga ZIP-fail. ASiCi standard ise krüptograafilisi omadusi ei sätesta. ASiCi standard sisaldab lihtsutatud ASiC-S konteinerivormingut ja laiendatud ASiC-E vormingut. Mõlemad konteinerivormingud võivad sisaldada kas XML-(XAdES) või CMS-vormingus (CAdES) signatuure või ajatempli tookenit. Lisaks võib ASiC-E konteiner sisaldada ka tõenduskirjeid. JAdES-vormingus signatuure need konteinerivormingud käesoleva aruande kirjutamise hetkel ei toeta.

4.5.1 ASiC-S

Associated Signature Container Simple (ASiC-S) on krüptokonteiner, mis sisaldab üht allkirjastatavat andmefaili ja üht signatuurifaili (mis võib sisaldada mitut signatuuri) XAdES- või CAdES-formaadis või ajatemplit RFC 3161 (õigemini selle uuenduse RFC 5816 [161]) formaadis. Allkirjastatav andmefail võib olla ka muuhulgas ZIP-fail, mis sisaldab mitut faili.

ASiC-S krüptokonteiner on võimekuselt alamhulk ASiC-E krüptokonteinerist. ASiC-S tugi on lihtsamini teostatav, see formaat on sobilik dokumentide digiallkirjastamiseks.

[6] https://eur-lex.europa.eu/eli/reg_impl/2024/2979/oj/eng

Eestis on ASiC-S ajatemplit sisaldav versioon kasutusel ka DDOC-vormingus digiallkirjade ajatembeldamiseks. Eestis varem levinud DDOC-digiallkirjad põhinesid SHA-1 räsialgoritmil, mis muutus aja jooksul ebaturvaliseks ning seetõttu DDOC-digiallkirjade loomise tugi katkestati.[7] Ajaloolised dokumendid viidi üle ASiC-S konteineritesse, pakendades esialgse DDOC-digiallkirja koos värske ajatempliga. ASiC-S konteinerites olevate digiallkirjade valideerimiseks saab kasutada DigiDoc4j[8] teeki või SiVa[9] või DigiDoc tarkvara.

4.5.2 ASiC-E

Associated Signature Container Extended (ASiC-E) krüptokonteiner sisaldab üht või mitut allkirjastatavat andmefaili ja CAdES- või XAdES-formaadis signatuure. ASiC-E konteiner võib sisaldada ka RFC 5816 [161] formaadis ajatempli tookenit ja RFC 4998 [162] või RFC 6283 [163] formaadis tõenduskirjet. Seejuures iga signatuur, ajatempel või tõenduskirje käib mingisuguse osahulga (k.a kõigi) konteineris olevate andmefailide kohta.

4.5.2.1 ASiC-E koos XAdES-signatuuriga

Eestis on ilmselt kõige levinumaks krüptokonteineriks ASiC-E koos XAdES (*XML Advanced Electronic Signatures*) signatuuriga, mille baasprofiili annavad ETSI EN 319 162-1 [160] ja ETSI EN 319 132-1 [159]. Seda baasprofiili varem defineerinud standardeid [164, 165] täpsustab Eesti kohaliku BDOCi spetsifikatsioon [166].

Eestis kasutatakse seda vormingut kõigi igapäevaste digiallkirjade korral.

Kuivõrd tegu on tehniliselt keeruka vorminguga, on oluline ASiC-E digiallkirjadega töötamisel kasutada mõnd head teeki valmisolevate hulgast[10], vastavalt oma platvormile.

4.5.2.2 ASiC-E koos CAdES-signatuuriga

ASiC-E koos CAdES-signatuuriga on vastava XAdES-vorminguga analoogne, kuid XMLi asemel kasutatakse seal signatuuri tehnilise kandjana vormingut CMS. ETSI EN 319 162-1 [160] annab baasprofiili ka ASiC-E konteinerile koos CAdESi (*CMS Advanced Electronic Signatures*) signatuurivorminguga. Signatuurivormingut kirjeldab ETSI EN 319 122-1 [167].

Eestis ei ole CAdES laialdaselt kasutusel, kuid nõue teistes Euroopa Liidu riikides kasutatavate digisignatuuri-vormingute aktsepteerimiseks tähendavad, et ka Eesti infosüsteemid peavad olema valmis sellised allkirju vähemalt valideerima, kasutades heakvaliteedilist valmisteeki, näiteks SiVa.

4.6 Mobiilne juhiluba ja kontrollitavad mandaadid

EUDI digikukru kontekstis relevantsete standardite ja tehniliste spetsifikatsioonide nimekiri on leitav avalikust GitHubi hoidlast[11]. Selles alapeatükis kirjeldame mõningaid selle nimekirja dokumentide vorminguid, mis on kasutusel sõnumite edastamisel esitaja ja kontrollija vahel

[7] https://www.id.ee/artikkel/id-kaardi-tarkvara-uuendus-toob-kaasa-mitu-olulist-muudatust/
[8] https://www.id.ee/en/article/digidoc-java-library-digidoc4j/
[9] https://open-eid.github.io/SiVa/
[10] https://github.com/open-eid/
[11] https://github.com/eu-digital-identity-wallet/eudi-doc-standards-and-technical-specifications

atribuutide avaldamise kontekstis (selle kontekstiga saab täpsemalt tutvuda analüüsis „Tõend kui platvorm“ [168]). Eesmärk on anda lühiülevaade nendest vormingutest ja neis kasutatud krüptograafilistest meetoditest. Selles alapeatükis kasutame järgmiseid mõisteid:

- **kontrollitav mandaat** (*verifiable credential*) on krüptograafiliselt turvatud väide **subjekt**i (*subject*) kohta;
- **väljaandja** (*issuer*) on pool, kellel on õigus välja anda kindlaid mandaate;
- **esitaja** (*holder*) on pool, kes esitab mandaati. Esitaja võib, kuid ei pruugi olla mandaadi subjekt;
- **kontrollija** (*verifier*) on pool, kes kontrollib mandaati ja veendub selle õigsuses;
- **valikuline avaldamine** (*selective disclosure*) on mandaadi esitamise viis, mis võimaldab näidata kontrollijale mandaadi vaid mõned valitud atribuudid;
- **seostamatus** (*unlinkability*) on mandaadi esitamise viis, mis ei võimalda kontrollijal tuvastada, kas ja kellele seda mandaati on juba varem esitatud.

Euroopa Liidu digiidentiteedirakendustesse, nagu sätestatud komisjoni rakendusmääruses 2024/2977[12], väljastatakse isikutuvastusandmed kas ISO/IEC 18013-5:2021 või W3C Verifiable Credentials Data Model 1.1 esitatud vormingus.

4.6.1 SD-JWT kontrollitavad mandaadid

JSON Web Token (JWT, vt ptk 4.4) struktuure saab kasutada kontrollitavate mandaatide väljendamiseks viisil, mida on lihtne mõista ja töödelda, kuna see põhineb väljakujunenud veebiprimitiividel. IETF RFC 9901 [169] spetsifikatsioon kirjeldab struktuuri SD-JWT (Selective Disclosure JWT). mis toetab JWT-struktuuride valikulist avaldamist: esitaja saab otsustada, millised atribuute näidata kontrollijale ja milliseid mitte (väljaandja määratletud piirides).

SD-JWT kontrollitavaid mandaate (*SD-JWT Verificable Credentials*, SD-JWT-VC) kirjeldab spetsifikatsioon [170], mis on käesoleva dokumendi kirjutamise ajal veel mustandi staatusega. See spetsifikatsioon kirjeldab, kuidas väljendada kontrollitavaid mandaate JSON-formaadis (vt ptk 4.1.2), mis põhineb SD-JWT vormingul.

Vormingu struktuur. SD-JWT on sisuliselt signeeritud JSON-dokument, mis sisaldab valikuliselt avaldatavate väidete räsisid, koos dokumendiväliste (valikuliselt) avaldatud väidetega. Avaldatud väiteid võib ära jätta ilma signatuuri rikkumata, kuid väidete muudatused on tuvastatavad tänu nende räside signatuuridele.

SD-JWT VC kontrollitavaid mandaate kodeeritakse SD-JWT formaadis. Mandaat võib kasutada kõiki IETF RFC 7519 [150] spetsifikatsioonis määratletud *JSON Web Token Claims* registris toodud väiteid. Järgmiseid JWT-väiteid *ei saa* avaldada valikuliselt (st ei saa kontrollija eest varjata):

- kontrollitava mandaadi väljaandja
- kehtivuse alguse aeg
- kehtivuse lõpu aeg
- kinnitusmeetod (*confirmation method*), millega tuvastatakse esitajat autentiv võti
- kontrollitava mandaadi tüüp, nt `https://credentials.example.com/identity_cred`
- teave kontrollitava mandaadi staatuse/kehtivuse kohta

[12] https://eur-lex.europa.eu/eli/reg_impl/2024/2977/oj/eng

Selles nimekirjas on eriline komponent *kontrollitava mandaadi tüüp*. Selle spetsifitseerib SD-JWT VC ja see on kontrollitava mandaadi jaoks kohustuslik. See võimaldab interpreteerida mandaati üheselt.

Lisaks ülaltoodud väidetele võib mandaat sisaldada erinevaid valikuliselt avaldatavaid väiteid, nagu nimi või sünniaeg.

Privaatsus. SD-JWT valikuline avaldamine põhineb soolaga räsimisel. Mis tahes andmeelemendi puhul, mis peaks olema valikuliselt avaldatav, sisaldab SD-JWT objekt andmeelemendi väärtuse asemel selle räsi. Selleks, et kontrollija ei suudaks ära arvata avaldamata andmeelementide väärtusi, kasutatakse räside loomisel soola ehk värsket juhuslikkust. Kontrollijale esitamiseks saadab esitaja väljaandja poolt signeeritud räsid koos valitud andmete avaldamiseks vajaliku infoga (andmete väärtuse ja kasutatud soolaga). Seejärel saab kontrollija ise arvutada avaldatud andmete räsi ja kinnitada, et see on lisatud signeeritud räside hulka.

Iga valikuliselt avaldatud väite kohta loob väljaandja avalduse järgmiselt:

- loo JSON-massiiv kolmest elemendist:
  - soola väärtus. Soola soovitatava entroopia saavutamiseks saab kasutada `base64url-encode`'i abil saadud 128 bitti krüptograafiliselt turvatud juhuslikke andmeid. Soola väärtus peab olema kordumatu iga valikuliselt avaldatud väite puhul;
  - väite nimi või võti;
  - väite väärtus;
- kodeeri saadud JSON-massiivi UTF-8 baidijada `base64url-encode`'i abil.

Saadud objekti nimetatakse avalduseks (*disclosure*). SD-JWT koosneb väljaandja signeeritud JWTst ja nullist või enamast avaldusest.

Iga avaldus räsitakse spetsiaalses andmeelemendis määratud räsialgoritmi, vaikimisi SHA-256 abil. Räsialgoritmi saab valida registrist IANA Named Information Hash Algorithm Registry[13], kuid iga registris sisalduv räsialgoritm ei pruugi olla SD-JWT kontekstis kasutamiseks sobiv. Selleks, et tagada valikuliselt avaldatavate väidete privaatsust, peab räsifunktsioon olema mittepööratav ja ei tohiks võimaldada vaatlejal saada osalist teavet avaldamata sisu kohta. Räsifunktsioon peab olema ka kollisioonikindel ja lisaoriginaalikindel, kusjuures kollisioonikindlus peaks vastama signeerimisskeemis kasutatava räsifunktsiooni kollisioonikindlusele.

Sama räsiväärtus ei tohi SD-JWTs esineda rohkem kui üks kord. Väiteid võib lisada ka avatekstina, näiteks juhul, kui konkreetsete väidete võimalik varjamine kontrollija eest ei ole kavandatud kasutusviisi puhul nõutav.

Tasub mainida, et räsipõhine valikuline avaldamine ei saavuta kahe avalduse omavahelist seostamatust (*unlinkability*).

Autentimine. Kontrollijad saavad kontrollida esitatud andmete ehtsust ja valikuliselt paluda esitajal tõendada, et ta on kontrollitava mandaadi kavandatud omanik, näiteks tõendades mandaadis viidatud krüptograafilise võtme omamist. Kui võtme sidumine on kasutusmalliga nõutav, peab SD-JWT sisaldama teavet võtmematerjali kohta. Esitaja avaldab objekti SD-JWT+KB, mis koosneb objektist SD-JWT ja sellega seotud võtmesiduvast JWTst (*Key Binding* JWT). Võtmesiduv JWT sisaldab signatuuri üle järgmiste objektide:

- SD-JWT räsi;

[13] https://www.iana.org/assignments/named-information/named-information.xhtml

- nonss, mis tagab signatuuri värskuse;
- kavandatud kontrollija, kellele dokument esitatakse.

Määrata tuleb signeerimisalgoritmi identifikaator, mille leiab JSON Object Signing and Encryption (JOSE) registrist[14].

Krüptimine. SD-JWT objekte peab säilitama ainult krüptitud kujul. Võimaluse korral tuleb dekrüptimisvõtit hallata riistvaralises turvamoodulis. Tsentraliseeritud salvestamisele tuleks eelistada andmete detsentraliseeritud salvestamist (nt kasutajaseadmetes). Aegunud SD-JWTd tuleks kustutada niipea kui võimalik.

SD-JWT või SD-JWT+KB krüptimiseks andmete edastamisel potentsiaalselt ebakindlate või lekkeohtlike kanalite kaudu võivad rakendajad kasutada JWE-meetodit (*JSON Web Encryption*, vt ptk 4.4). Seda meetodit tuleks kasutada eriti juhul, kui SD-JWT edastatakse URLi kaudu ja teavet võidakse salvestada brauseris või veebiserveri logides.

Tühistamine. Kontrollitav mandaat võib sisaldada atribuuti `status`, mis ütleb, kust saab järgi vaadata mandaadi kehtivust. Selles osas viidab SD-JWT VC spetsifikatsioon TSL-i (*Token Status List*) spetsifikatsioonile [171], mis on aruande kirjutamise ajal veel mustandi staatuses. See spetsifikatsioon määrab mehhanismi, andmestruktuurid ja töötlemiseeskirjad JOSE või COSE abil turvatud objektide staatuse avaldamiseks. Spetsifikatsioonis määratud mehhanism võimaldab küsida korraga paljude objektide staatust, vähendades märkimisväärselt interaktsioone väljaandjaga, tagades sellega parema mastaabitavuse ja privaatsuse, kuna väljaandja ei saa teada, millise konkreetse objekti staatust kontrollitakse (kollektiivne anonüümsus).

4.6.2 ISO mDoc-vorming

Andmemudel `mdoc`[15] määratleb struktureeritud viisi andmete esitamiseks mobiilses dokumendis, mis on peamiselt mõeldud juhiloa tõendina, kuid on kasutatav ka teistes kontekstides. Seda vormingut kasutatakse nii päringusõnumite (mida saadab kontrollija) kui ka vastussõnumite (mida saadab esitaja) puhul. Aruandes keskendume vastussõnumile, mis sisaldab väidetavaid atribuute.

Vormingu struktuur. Mobiilne dokument sisaldab järgmiseid parameetreid:

- `DocType` tuvastab mDoc-dokumendi tüübi, praegu `org.iso.18013.5.1.mDL`. Väli, mis praegu on `1`, võib vormingu tulevastes versioonides saada suurema väärtuse;
- `NameSpace` annab nimeruumi, milles on määratletud mDoci andmeelemendid. Koos dokumendi tüübiga võimaldab see esitatud andmeid üheselt interpreteerida;
- lisaks sisaldab dokument andmeelemente. Igal andmeelemendil on identifikaator `DataElementIdentifier`, mis tuvastab andmeelemendi, ja `DataElementValue`, mis sisaldab andmeelemendi väärtust.

Standard sisaldab kohustuslike ja valikuliste andmeelementide loetelu. Kohustuslikud elemendid on:

- ainulaadne dokumendi number;

[14] https://www.iana.org/assignments/jose/jose.xhtml
[15] https://www.evs.ee/et/iso-iec-18013-5-2021

- identifitseeriv teave (nagu täisnimi, sünniaeg, näokujutis);
- väljaandja (riik ja väljaandev asutus);
- väljaandmise ja kehtivuse lõppkuupäev;
- kontrolliks oluline teave, mis mobiilse juhiloa puhul on juhiõiguste loetelu.

Valikulised andmeelemendid hõlmavad üksikasjalikumaid isikuandmeid, nagu kodakondsus või elukoha aadress, mis on rakendusepõhised.

Andmeelemendid ja nendega seotud teave on kaitstud mobiilse turvaobjekti (*Mobile Security Object, MSO*) abil, mis muuhulgas sisaldab esitaja seadet autentivat avalikku võtit. MSO on kodeeritud CBOR-vormingus (vt ptk 4.1.1). MSO on kapseldatud ja signeeritud märgistamata COSE (CBOR Object Signing and Encryption) struktuurina, mida täpsemalt kirjeldab alapeatükk 4.4.6.

Privaatsus. Privaatsuse tagamiseks kasutatakse valikulise avaldamise mehhanismi, mis sarnaneb SD-JWT omaga. See võimaldab mobiilse dokumendi andmeelemente osaliselt varjata ja näidata kontrollijale ainult tehingu jaoks vajalikke andmeelemente. Väljaandja signeerib MSO-objekti, mis sisaldab andmeelementide väärtuste asemel nende soolatud räsisid. Iga MSO-andmeelemendi kohta arvutatakse eraldi räsi. Räsifunktsiooni sisend koosneb väljadest `digestID` (mis tuvastab seda räsi üheselt), `random` (ettearvamatu juhuslik või pseudojuhuslik väärtus) ning `elementIdentifier` ja `elementValue` (mis sisaldavad tegelikke andmeid). Väärtus `random` on iga elemendi puhul erinev ja selle pikkus on vähemalt 128 bitti. Selle eesmärk on tagada, et signeeriud räsi iseenesest ei annaks teavet selle sisu kohta. Andmeelemendi avaldamiseks peab kontrollijale selle väärtuse avaldama. Toetatud räsifunktsioonid valikulise avaldamise mehhanismi jaoks on `SHA-256`, `SHA-384` ja `SHA-512`.

Esitaja privaatsust saab veelgi tugevdada nullteadmuse (*zero-knowledge, ZK*) tõendiga. Selleks peab vastus sisaldama krüptograafilist tõestust bitijadana. Spetsiaalne andmeväli `ZkSystemID` määrab, millist nullteadmusskeemi kasutatakse. Lubatud `ZkSystemID`-de kogum pärineb kontrollija päringust, kus igale identifikaatorile on lisatud nullteadmusskeemi kirjeldus ja parameetrid. Konkreetsed lubatud nullteadmusskeemid ja parameetrid ei kuulu spetsifikatsiooni kohaldamisalasse.

Autentimine. Autentimisvõtit kasutatakse esitaja autentimiseks ühel kahest viisist: kas ECDH (*Elliptic Curve Diffie-Hellman*)-kokkuleppega sõnumiautentimiskoodi (*Message Authentication Code, MAC*) või ECDSA/EdDSA signatuuri abil. MAC-autentimiseks arvutavad esitaja ja kontrollija ECKA-DH (*Elliptic Curve Key Agreement*) võtmekehtestusalgoritmi abil efemeerse MAC-võtme. Kontrollija seade peaks toetama mõlemat lähenemist. Signatuuridest on lubatud ES256 (ECDSA ja SHA-256), ES384 (ECDSA ja SHA-384), ES512 (ECDSA ja SHA-512) ja EdDSA. Standard nimetab ka, millised konkreetsed elliptilised kõverad on kasutatavad. EdDSA-d võib kasutada nii kõveraga Ed25519 kui ka kõveraga Ed448.

Krüptimine. Dokumendi andmeid saab krüptida hübriidvõtmega krüptimise (*hybrid public key encryption, HPKE*) abil. See omakorda kasutab võtmevahetusmehhanismi `DHKEM_P256`, võtmetuletusfunktsiooni `HKDF_SHA256` ja sümmeetrilist krüptimist `AES_128_GCM`.

Tühistamine. MSO võib sisaldada struktuuri `Status`, mis sisaldab MSO tühistamisteavet. See struktuur sisaldab viidet MSOde tühistamisloendile: see on COSE-struktuur, mis näitab, kas konkreetne MSO on tühistatud või mitte. See sisaldab kogu teavet, mida kontrollija vajab, et

teha kindlaks, kas MSOde tühistamisloend on autentne.

4.6.3 W3C kontrollitavad mandaadid

See alapeatükk põhineb W3C *soovitustel*. W3C soovitus on spetsifikatsioon või nõuete kogum, mis on saanud W3C heakskiidu. W3C soovitused on ametlikult läbi vaadanud W3C liikmed, tarkvaraarendajad ning W3C rühmad ja huvitatud isikud. W3C *soovitab* soovituse laialdast kasutuselevõttu veebi standardina. Neid dokumente võib nimetada ka W3C standarditeks.

Kontrollitav mandaat on võltsimiskindel mandaat, mille autorsust saab krüptograafiliselt kontrollida. Kontrollitavaid mandaate saab kasutada kontrollitavate esituste loomiseks, mis võivad olla ka krüptograafiliselt kontrollitavad (nt sisaldab väljaandja signatuuri). Kontrollitavaid mandaate kasutatakse nii ühe või mitme subjekti omaduste kui ka mandaadi enda omaduste esitamiseks.

Keskses rollis on soovitus *W3C Verifiable Credential Data Model* v2.0[16] (edaspidi W3C VC DM 2.0), mis määratleb põhimõisted, mis on aluseks kõigile teistele seotud W3C soovitustele. Mudel on määratletud abstraktselt ja rakendused väljendavad oma konkreetseid mandaate, kasutades andmemudeli jadastust (*serialization*). Praegused soovitused kasutavad enamasti JSON-jadastust (vt ptk 4.1.2). Jadastamisel on oluline veenduda, et kõik pooled tõlgendavad mandaadi struktuuri ühtemoodi.

Vormingu struktuur. Kontrollitav mandaat sisaldab järgmiseid kohustuslikke parameetreid:

- kontekst (`context`): URLi spetsifikatsioon, mis tagab et kõik pooled tõlgendavad mandaati ühtemoodi;
- tüüp (`type`): mandaadi kavandatud kasutusmall. Selle struktuur põhineb JSON-LD 1.1 vormingul;
- väljaandja (`issuer`): pool, kes väidab, et mandaat on tõene. See on kas URL või objekt, mis sisaldab atribuuti `id`, mille väärtus on URL;
- subjekt (`credential subject`): pool, kelle kohta väide esitatakse. Tehniliselt on see kogum objekte, kus iga objekt peab omakorda olema ühe või mitme mandaadi subjekt. Iga objekt võib sisaldada parameetrit `id` subjekti tuvastamiseks (kuid tasub mainida, et üldiselt ei ole väli `id` mandaadi puhul kohustuslik). Mitme subjektiga seotud teabe väljendamine kontrollitavas mandaadis on täiesti võimalik;
- kehtivusaeg (`validity period`): mandaadi kehtivusperiood.

Mandaati saab kaitsta kahe erineva mehhanismi abil: **ümbrikutõendid** (*enveloping proofs*) või **manustõendid** (*embedded proofs*). Mõlemal juhul on mandaadi autentsus tagatud krüptograafiliselt (näiteks signatuuri abil). Ümbrikutõendi korral saab mandaadist tõestuse osa. Manustõend on lisatud jadastusse koos mandaadiga.

Soovituses W3C VC DM 2.0 määratletud **ümbrikutõendid** põhinevad JSON-objekti signeerimisel ja krüptimisel (JOSE, vt ptk 4.4), CBOR-objekti signeerimisel ja krüptimisel (COSE, vt ptk 4.4.6) või JWTde valikulisel avaldamisel (SD-JWT, vt ptk 4.6.1). Need on kõik IETFi spetsifikatsioonid, kuid olemas on ka eraldi W3C soovitus[17], mis defineerib seosed IETFi spetsifikatsioonide ja W3C kontrollitavate mandaatide põhimudeli W3C VC DM 2.0 vahel.

[16] https://www.w3.org/TR/vc-data-model-2.0/
[17] https://www.w3.org/TR/vc-jose-cose

Manustõendite üldine struktuur on määratletud eraldi W3C soovituses[18]. Mõningad selle üldise struktuuri instantsieerimised põhinevad EdDSA, ECDSA ja BBS (Boneh-Boyen-Shacham) signatuuridel.

BBS (vastava RFC mustand on koostamisel[19]) võimaldab signeerida mitu sõnumit, mida saab avaldada valikuliselt, kasutades nullteadmustõestusi. Selline tõestus varjab ära mitte ainult avaldamata sõnumid, vaid ka signatuuri enda, aidates sellega kaasa seostamatusele.

Privaatsus. Vormingus kaalutakse järgmisi privaatsusmehhanisme:

- atribuutide valikuline avaldamine. See võimaldab esitajal anda kontrollijale täpselt vajalikku teavet ja mitte midagi muud. Soolatud räsil põhinevat mehhanismi kirjeldab alapeatükk 4.6.1.

  Alternatiivselt saab välja anda iga atribuudi jaoks eraldi mandaadi, kombineerides need ühes avalduses. Seejuures peab aga olema ettevaatlik, et atribuute poleks võimalik kombineerida tahtmatul viisil (näiteks esitades erinevate dokumentide andmevälju ühise dokumendina);
- mittekorrelatsiooniline tuvastamine. See võimaldab esitajal tõestada, et talle on välja antud mandaat (või subjektil tõendada, et tema kohta on välja antud mandaat), seejuures jagamata kontrollijaga esitaja (subjekti) unikaalset identifikaatorit;
- atribuutide seostamatu avaldamine. See saavutatakse BBS-signatuuride ja nullteadmustõestuste abil. Valikuline avaldamine ja identifikaatori varjamine ei ole piisavad, kui kontrollija näeb mitmel esitamisel ühte ja sama signatuuri.

Autentimine. Esitatud kontrollitavates mandaatides avaldab atribuut `credentialSubject` kontrollijale selle mandaadi subjekti, mis omakorda võib olla seotud identifikaatoriga `id`. Kui esitaja on mandaadi subjekt, võib kontrollija ta autentida, kui tal on olemas kõik seleks vajalikud metaandmed. Sel juhul saab kontrollija esitaja autentimiseks kasutada kontrollitavas avalduses sisalduvat esitaja signatuuri. Parameeter `id` on valikuline. Kontrollijad võivad kasutada subjekti üheseks tuvastamiseks ka teisi atribuute.

Võtmehalduse osas viidatakse NISTi ja FIPSi soovitustele. Samuti mainitakse, et missioonikriitiliste süsteemide rakendajaid julgustatakse tungivalt konsulteerima turvalisuse ja krüptograafia spetsialistidega, et saada põhjalikke juhiseid.

Krüptimine. Kuna kontrollitav mandaat sisaldab sageli isikut tuvastavat teavet, on rakendajatel tungivalt soovitatav kasutada kontrollitavate mandaatide säilitamiseks ja transportimiseks mehhanisme, mis kaitsevad andmeid isikute eest, kellel ei peaks neile juurdepääsu olema. Mehhanismid, mida võiks kaaluda, on transpordikihi turve (TLS) või muud edastatavate andmete krüptimise vahendid, samuti krüptimis- või juurdepääsukontrollimehhanismid, mis kaitseks andmeid kontrollitavas mandaadis jõudeolekus.

Tühistamine. Spetsiaalne parameeter `credentialStatus` aitab leida kontrollitava mandaadi staatusega seotud teavet, näiteks selle peatamise või tühistamise kohta. Selle täpne sisu võib olla mitmesugune. Näiteks võib see sisaldada linki välisele dokumendile, mis märgib, kas mandaat on peatatud või tühistatud. Rakendajaid hoiatatakse, et mitme olekukirjega mandaadid võivad

[18] https://www.w3.org/TR/vc-data-integrity/
[19] https://www.ietf.org/archive/id/draft-irtf-cfrg-bbs-signatures-10.html

sisaldada vastuolulist teavet. Selliste konfliktide lahendamine on osa valideerimisprotsessist, mis ei kuulu W3C VC DM 2.0 käsitlusalasse.

Spetsifikatsiooni paindlikkus ja mõned hoiatused. W3C VC DM 2.0 sisaldab hoiatusi, et spetsifikatsioon on paindlik ja seega ei kaitse võimaliku väärkasutuse eest. Näiteks võimaldab see luua mandaate ilma signatuuride või tõestusteta. Rakendajad peaksid olema teadlikud, et need mandaadi tüübid ei ole kontrollitavad, sest autorsus kas ei ole teada või seda ei saa usaldada.

Teine oluline aspekt on eri mandaatide kombineerimine avalduses, eriti juhtudel, kui mitu seotud atribuuti on väljastatud eraldi mandaatidena. Väljaandja ja kontrollija peavad veenduma, et avaldus ei sisaldaks mitme erineva subjekti andmeid ühe subjekti nime all, või et see ei kombineeriks eri mandaatidest tulenevaid väiteid sobimatul viisil.

Üldine soovitus on olla ettevaatlik, et privaatsusega mitte liialdada. Kontrolliprotseduur peaks olema üldjuhul hoolega läbi mõeldud. Üks toodud näide illustreerib, miks vanusekontrollist ei piisa ning on oluline ka muu info, nagu näokujutis.

4.7 CDOC ja CDOC2

4.7.1 CDOC

CDOC on XML Encryptioni [172] baasil koostatud Eesti kohalik vorming krüptitud andmete edastamiseks. CDOC-konteineriformaadi spetsifikatsioon loodi 2005. aastal ning selle viimane versioon pärineb aastast 2012 [173].

Krüptimiseks kasutatakse avaliku võtmega krüptimist (RSA PKCS#1 v1.5, ECC *Integrated Encryption Scheme* (IES)). Selliselt krüptitud sõnumite dekrüptimiseks vajalikke liideseid pakuvad vaid ID-kaart ja krüptopulgad (Mobiil-ID ja Smart-ID tugi puudub).

CDOC-vorming ei tõenda sõnumi autori autentsust. Kui autentsuse tuvastamine on vajalik, tuleb edastatavad materjalid kas enne või pärast krüptimist digitaalselt allkirjastada. Samal ajal tuleb tähelepanu pöörata sellele, et selline kaheastmeline krüptograafiliste meetodite rakendamine ei pruugi anda soovitud kaitset: ründajale jääb tihtipeale võimalus tulemust manipuleerida [174, p. 4.3] [175]. Täpseid vastumeetmeid on keeruline ette kirjutada, kuivõrd need sõltuvad konkreetsest rakendusest kus CDOCi kasutatakse, kuid üldkehtiv soovitus on allkirjastada sisu enne krüptimist ning hoolitseda selle eest, et allkirjastatud sisu hulgas oleks ka info selle kohta, kellelt kellele sõnum (dokument) on mõeldud liikuma [176].

CDOC on mõeldud krüptitud failide transpordiks. Oluline on märkida, et CDOC vorming ei paku tulevikusalastust (*forward secrecy*), mistõttu ei tohi CDOCi kasutada krüptitud materjalide säilitamise eesmärgil. Tulevikusalastuse puudumise ja teabe salastusajaga tuleb arvestada ka transpordiks krüptides. Hoiatav näide sellest, et tulevikuturvalisuse puudumine on probleem, oli 2017. aasta ROCA intsidendi kriis, mille käigus selgus, et ID-kaartide RSA-privaatvõtmed olid genereeritud viisil, mis tegi nende avalikust võtmest väljaarvutamise piisavalt lihtsaks. Muuhulgas tähendab see, et sellistele võtmetele saadetud CDOC-konteinerid on piisavalt võimekate ründajate poolt avatavad [5].

4.7.2 CDOC2

CDOC2 on koodnimetus Eesti uuele failikrüptimisstandardile, mis kasutab CDOC2 failivormingut[20] ning lahendab järgmised CDOCi spetsifikatsioonide ja teostustega seotud probleemid:

- CDOC ei paku tulevikuturvalisust;
- CDOC ei erista krüptimist transpordiks ning krüptimist säilitamiseks;
- CDOC ei võimalda krüptida vastuvõtjale, kes kasutab vaid mobiil-IDd või Smart-IDd.

CDOC2 väljatöötamist alustati 2020. aastal Riigi Infosüsteemi Ameti projekti „CDOC2.0 analüüs (info pikaajalise salastamise lahenduse arhitektuur)“ käigus [177]. Siinse aruande kirjutamise ajaks (2026. aasta II kvartal) on CDOC2 integreeritud DigiDoci tarkvarasse[21] ning on loodud CDOC2 käsurearakenduse etalonteostus[22] ja CDOC2 kapsliserveri etalonteostus[23]. Lisaks on valmimas iOSi ja Androidi tugi, mis võimaldab dokumente krüptida ja dekrüptida ka mobiil-iD ja Smart-ID abil[24]. CDOC2 dokumentatsioon on kättesaadav veebilehel https://open-eid.github.io/CDOC2/.

CDOC2 võimaldab kasutada dekrüptimisvõtme haldamiseks riistvaralist vahendit (ID-kaarti või krüptopulka), kuid võtmeserveri arhitektuur toetab ka teisi lahendusi.

CDOC2 versioon 1.7 võimaldab kasutada kuut erinevat krüptimisskeemi[25]:

- SC01: krüptimisskeem vastuvõtjatele, kellel on EC-võtmepaar. Jagatud saladuse kokkuleppimiseks kasutatakse Diffie-Hellmani võtmekehtestust;
- SC02: krüptimisskeem vastuvõtjatele, kellel on RSA-võtmepaar. Kasutatakse RSA-OAEP krüptimisskeemi;
- SC03: võtmeserveriga krüptimisskeem vastuvõtjatele, kellel on EC-võtmepaarid;
- SC04: võtmeserveriga krüptimisskeem vastuvõtjatele, kellel on RSA-võtmepaarid;
- SC05: krüptimisskeem vastuvõtjatele, kellel on eeljaotatud sümmeetriline saladus (*pre-shared symmetric secret*);
- SC06: eeljaotatud paroole kasutav krüptimisskeem.

CDOC2 krüptimisskeemid SC03 ja SC04 kasutavad uudse lähenemisena võtmeedastusserverit. Sellise lähenemise korral kasutatakse sümmeetriliselt krüptitud sõnumi (mis liigub otse vastuvõtjale näiteks e-posti vahendusel) ning võtmekapsli (mis saadetakse läbi serveri) transportimiseks erinevaid kanaleid. See tähendab, et pool, kes jälgib otsekanalit, ei saa ligipääsu asümmeetrilisele krüptogrammile, millega põhisõnumi krüptimiseks kasutatud sümmeetrilist võtit kaitstakse. See omakorda tähendab, et isegi kasutatava asümmeetrilise krüptograafia murdumisel säilib sellise ründaja vastu sõnumi salajasus.

Krüptimisskeemid SC05 ja SC06 on mõeldud pikaajaliselt talletatavate dokumentide krüptimise jaoks. Nendes krüptimisskeemides võib krüptija määrata vastvõtjaks iseennast. CDOC2 võimaldab pärast dekrüptimist faile säilitamise eesmärgil rekrüptida.

[20] https://open-eid.github.io/CDOC2/1.7/02_protocol_and_cryptography_spec/ch03_container_format/
[21] https://www.id.ee/artikkel/cdoc-2-0/
[22] https://github.com/open-eid/cdoc2-java-ref-impl
[23] https://github.com/open-eid/cdoc2-capsule-server
[24] https://cyber.ee/resources/news/cdoc2/
[25] https://open-eid.github.io/CDOC2/1.7/02_protocol_and_cryptography_spec/ch02_encryption_schemes/#cdoc2-encryption-schemes

CDOC2 versioon 2.0 lisab krüptimisskeemi SC07, mis kasutab (N-N)-tüüpi saladusejaostust[26]. CDOC2 krüptimisskeem SC07 võimaldab jagada dekrüptimisvõtit ühissalastuse abil mitmeks osaks ning levitada võtmeosakuid erinevate CDOC2 serverite kaudu. Sellisel juhul piisab dekrüpterimiseks vaid autentimisest, mistõttu saab dekrüptida ka Smart-ID ja mobiil-ID abil.

Loomulikult tuleb CDOC2 serverile ligipääsu eraldi turvata, nii näiteks saab võtmekapsli allalaadimist autentida Eestis kasutusel olevate eID-vahenditega.

Teine vajadus, millega CDOC2 pikemas perspektiivis arvestab, on üleminek postkvant-krüptoalgoritmidele. Hetkel pole veel turul riistvaralisi postkvant-võtmeseadmeid, aga kui need ilmuvad, on nende lisamine CDOC2 spetsifikatsiooni lihtne. Samuti on võimalik postkvant-krüptoalgoritmide võtmeid ühissalastuse abil tükkide kaupa erinevate võtmeedastusserverite kaudu levitada.

CDOC2 kasutab võrreldes oma eelkäijaga ka teistsuguseid andmevorminguid. Edastatavad failid arhiveeritakse, kasutades POSIX tar-vormingut [178]. Arhiveeritud failid pakitakse ZLIB-meetodil, mille standardis IETF RFC 1950 [179]. Metainfoga varustatud konteiner jadastatakse lõpuks FlatBuffers-vormingu[27] abil.

[26] https://open-eid.github.io/CDOC2/2.0-Draft/02_protocol_and_cryptography_spec/ch02_encryption_schemes/#sc07-encryption-scheme-with-n-of-n-secret-shared-decryption-key
[27] https://google.github.io/flatbuffers/

5 Autentimise, delegeerimise ja atribuutide esitamise protokollid

5.1 OpenID Connect

5.1.1 Protokolli kirjeldus

OpenID Connect[1] on OAuth 2.0 karkassile (vt ptk 5.4) ehitatud identiteediinfo kiht. OpenID Connect spetsifitseerib viisid, kuidas delegeerida kasutaja autentimine autentimisteenusele ja autentimise tulemus klientrakendusele tagastada. OpenID Connect defineerib ka viisid kasutaja profiili-info standardseks edastamiseks.

Abstraktsel tasemel järgib OpenID Connecti protokoll järgmisi juhiseid:

- klient (RP) saadab päringu OpenID tarnijale (OP);
- OP autendib lõppkasutaja;
- OP vastab identiteeditõendi (*ID token*) ja tavaliselt ka pääsutõendiga (*access token*);
- RP võib saata pääsutõendiga päringu kasutajainfo teenusele (*UserInfo Endpoint*);
- kasutajainfo teenus tagastab väiteid lõppkasutaja kohta.

Identiteeditõend on esitatud *JSON Web Token* (JWT) struktuurina (vt ptk 4.4). Pääsutõend on määratletud OAuthi spetsifiktsioonides (vt ptk 5.4), mis pakuvad üldist raamistikku kolmandate poolte rakendustele, kuidas saada piiratud ligipääsu HTTP-ressurssidele.

Autentimine võib toimuda ühel kolmest viisist: volituskoodiga voog (*Authorization Code Flow*), kaudne voog (*Implicit Flow*) või hübriidvoog (*Hybrid Flow*). Vood määravad, kuidas identiteedi- ja pääsutõend kliendile tagastatakse.

OpenID Connect protokollistiku põhjalikuma kirjelduse leiab 2023. aasta krüptoaruandest. Võrreldes aastaga 2023 ei ole selles olnud olulisi muudatusi. Digikukrute kontekstis on oluline uus asi OpenID4VP, mida käsitleb alapeatükk 5.5.2.

5.1.2 Liidendus

OpenID Federation 1.0[2] kirjeldab viisi, kuidas kaks omavahel suhelda soovivat poolt saavad kolmanda poole (nn usaldusankru) abil luua omavahelise usaldusseose. See viis on üks võimalikest viisidest, kuidas OpenID Connect ja OAuth 2.0 protokollid kliendi (nt veebiteenuse) ja identiteeditarnija vahel usaldust loovad. Usaldusseos tähendab usku, et teine pool kuulub samasse **liidendisse** (*federation*) ja et teist poolt kirjeldavad metaandmed (mida kasutatakse OpenID Connect protokolli kontekstis) on tema saadetud. Liidend kujutab endast teineteist usaldavate poolte vahelist kokkulepet, mis määrab, kuidas vahetatakse ja hallatakse identiteediteavet.

OpenID spetsifikatsioon käsitleb pooli kui olemeid. **Olem** on miski, millel on eraldiseisev ja eristatav eksistents ning mida saab tuvastada teatud kontekstis. Olem liidendis peab saama olla kindel, et teised olemid, kellega ta suhtleb, kuuluvad samasse liidendisse. Samuti peab tal

[1] https://openid.net/specs/openid-connect-core-1_0.html, OpenID Connect Core 1.0 incorporating errata set 2, Last modified December 15, 2023

[2] https://openid.net/specs/openid-federation-1_0.html

olema võimalik uskuda, et teavet, mida teised olemid enda kohta avaldavad, ei ole transpordi ajal rikutud ja et teave järgib liidu poliitikat. Organisatsiooni võib esindada mitu olemit. Olem võib kuuluda ka rohkem kui ühte liidendisse.

Kahe olemi tõendatud kuulumine ühte liidendisse on aluseks nendevahelise usalduse tekkimisele. Kahepoolses (*bilateral*) liidendis on võimalik luua otsene usaldus kahe samasse liitu kuuluva organisatsiooni vahel. Mitmepoolses (*multilateral*) liidendis ei pruugi kahepoolsed lepingud olla praktilised, sellisel juhul võib usaldust vahendada kolmas isik. OpenID Federation 1.0 spetsifikatsioon kirjeldab, kuidas kaks olemit saavad luua omavahelise usalduse usaldatava kolmanda poole abil. **Usaldusankur** (*trust anchor*) on usaldatav olem, mille peamine eesmärk on väljastada avaldusi olemite kohta.

Identiteediliitu saab realiseerida, kasutades ühte või mitut hierarhilist tasandit. Usaldushierarhias võib olemil olla ülemaid olemeid (kes esitavad väiteid olemi kohta) ja alamolemeid (kelle kohta olem esitab väiteid). Madalaima taseme olemeid nimetatakse **lehtolemiteks** (*leaf entity*) ja kõrgema taseme olemeid, välja arvatud usaldusankur, **vaheolemiteks** (intermediate entity). OpenID Connect-identiteediliidus on lehtolemiteks klient (RP) ja OpenID tarnija (OP).

Olemi identifikaator (*Entity Identifier*) on globaalselt unikaalne identifikaator, mis on seotud ühe olemiga. Kõik spetsifikatsiooniga määratletud identifikaatorid on URL-id, mis kasutavad https-skeemi, millel on hosti komponent ja mis võivad sisaldada pordi- ja teekomponente.

Iga olem saab luua vastastikuse usalduse mis tahes teise olemiga, kui nende vahel leidub vähemalt üks ühine usaldusankur. Usalduse põhikomponent on **olemideklaratsioon** (*Entity Statement*), mis on krüptograafiliselt signeeritud JWT struktuur (vt ptk 4.4). See struktuur sisaldab deklaratsiooni väljaandjat, subjekti, kehtivusaega ning spetsiaalset struktuuri JWKS (*JSON Web Key Set*, vt ptk 4.4), mis kujutab subjekti signeerimisvõtme avalikku osa. Vastavat salajast võtit kasutab olem iseenda kohta väidete signeerimiseks, ning usaldusankrud ja vaheolemid oma vahetute alamate kohta väidete signeerimiseks. JWT võib sisaldada ka täiendavat teavet nagu olemi tüüp, mida saab kasutada Open ID Connecti ja OAuth protokollides.

Usalduse loomine. Olemideklaratsioonid võivad olla järgmist tüüpi:

- kõik olemid liidendis avaldavad enda kohta olemideklaratsiooni, mida nimetatakse **olemi konfiguratsiooniks** (*Entity Configuration*). Leht- või vaheolemite konfiguratsioon sisaldab üht või mitut viidet vahetutele ülematele olemitele;
- ülemad olemid liidendis (usaldusankur ja selle vahendajad) avaldavad olemideklaratsioonid oma vahetute alamate kohta, mida nimetatakse **alamdeklaratsioonideks** (*subordinate statements*).

Usaldusahel (*Trust Chain*) on olemite avalduste jada, mis algab ahela subjekti konfiguratsioonist (tema väidetest enda kohta) ja lõpeb usaldusankruga. Olemi konfiguratsioonide ja ülema/alama vahelist suhet kinnitatavate deklaratsioonide järjestus usaldusankru suunas tõendab, et usaldusahela subjekt (tavaliselt lehtolem) on usaldusankrule tugineva liidendi liige. Ahela kontrollimiseks tuleb kontrollida kõiki selles sisalduvaid olemideklaratsioone, st signatuure, väljaandja/subjekti välju ning kehtivuskuupäevi.

Usaldusankrud ja vaheolemid võivad määrata täiendavaid kontrollipõhimõtteid. Näiteks võivad nad seada piiranguid maksimaalsele lubatud usaldusahela pikkusele, olemite tüüpidele või olemite nimedele.

Eelnevaid mehhanisme saab kasutada usalduse loomiseks OpenID tarnija (OP) ja kliendi (RP) vahel, kellel puudub varasemast olemasolev konfiguratsioon või registreering. Spetsifikatsioon

kirjeldab seda protsessi detailsemalt.

Aegumine ja tühistamine. Igal usaldusahelal on aegumisaeg. Seega peavad liidendis osalejad toetama usaldusahela värskendamist. Kui tihti osaleja usaldusahelat ümber hindab, sõltub sellest, kui kiiresti ta tahab teada saada, et midagi on muutunud.

Spetsifikatsioon soovitab, et usaldusankrute avalikud võtmed peaksid olema saadaval meetodil, mis ei sõltu nende olemite konfiguratsioonist. Selle eesmärk on pakkuda liiasust juhul, kui avaliku võtme infrastruktuur [180], millel põhineb avalike võtmete hankimine olemite konfiguratsioonidest, osutub rikutuks.

Kuna liidendis osalejatelt eeldatakse usaldusahela regulaarset kontrollimist, ei määratleta selles spetsifikatsioonis kindlat tühistamisprotsessi. Konkreetsed liidendid võivad vajadusel määratleda oma tühistamisprotsessi.

5.2 FIDO, WebAuthn ja pääsuvõtmed

FIDO Alliance[3] töötab välja standardeid, mis võimaldavad kasutajatel ühe autentimistegurina (*authentication factor*) kasutada asümmeetrilise krüptosüsteemi võtmepaari. Kogu protokollistiku oluline erinevus näiteks Eestis harjumuspärasest ID-kaardi taristust seisneb selles, et igal kasutajal on palju võtmepaare ning iga võtmepaar on kasutusel autentimiseks vaid ühe rakendusega. See on oluline meede mitmesuguste asendus- ja vahendusrünnete vältimiseks, samuti teeb see raskemaks kasutaja tegevuste korreleerimise erinevate tuginevate isikute (*relying party*) vahel.

Võtmepaare haldab autentur (*authenticator*), milleks võib olla iseseisev riistvaraline seade või kasutaja seadmesse integreeritud platvormi autentur (*platform authenticator*). Platvormi autenturid on laialdaselt kasutusel, kuna need on integreeritud operatsioonisüsteemi ja tuginevad seadmes olevale turvamoodulile (nt TPM, Apple Secure Enclave, Trusted Execution Environment). Tüüpiline riistvaraline autentur on USB- ja NFC-liidesega „pulk“[4]. *FIDO Alliance* standardid kirjeldavad kommunikatsiooni selliste seadmetega ning kaugautentimiseks kasutatavaid autentimis- ning atesteerimisprotokolle, mille käigus noid võtmepaare kasutatakse. Lisaks iseseisvale riistvaralisele autenturile ja platvormi autenturile toetab FIDO2 ka mitmeseadmelisi pääsuvõtme tüüpi mandaate (*multi-device passkeys*), mida saab mitme erineva seadme vahel sünkroniseerida.

Senised FIDO autentimise spetsifikatsioonid jagunevad kolme rühma:

- FIDO U2F (*FIDO Universal 2nd Factor*; FIDO2 osana teise nimega CTAP1) – vanem pärandprotokollistik, millele tuleks võimalusel alati eelistada FIDO2 kasutamist. Defineerib ennekõike krüpotvõtmeid haldava füüsilise autentimisseadme ja selle kasutuse tugeva teise autentimistegurina;
- FIDO UAF (*FIDO Universal Authentication Framework*) – vanem protokollistik, mis pole FIDO2 ja WebAuthniga ühilduv. Kui U2Fi eesmärk oli pakkuda turvalist teist autentimistegurit, siis UAF kirjeldab suhtlust ka mitmikautentimist võimaldava autenturiga, mis eemaldab vajaduse parooliga autentimise järele. Lisaks U2F-autenturitele võimaldab see kasutada autentimisseadmeid, mis enne võtme kasutamist kontrollivad kasutajasamasust ka muul viisil, näiteks biomeetrilise kontrolliga.;

[3] FIDO Alliance (https://fidoalliance.org/) on tööstusühendus, mille missiooniks on autentimisstandardite abil vähendada maailma sõltuvust paroolidega autentimisest. Organisatsioon edendab autentimis- ja atesteerimisstandardite loomist, kasutamist ja nende kasutamise ühilduvuskontrolli.

[4] https://www.yubico.com/ee/product/yubikey-5-nfc/

- FIDO2 (katab standardeid CTAP[5] ja WebAuthn) – täna kehtiv standard tugeva autentimise toetamiseks veebirakendustes. WebAuthn defineerib autentimise veebirakenduses koos asjakohasete protokollide ja APIdega. CTAP2 võimaldab kasutada autentimiseks välist autenturit või platvormipõhist autenturit. Lisaks võimaldab FIDO2 tagasiühilduvuse eesmärgil kasutada ka protokolli CTAP1. FIDO2 põhise autentimislahenduse näiteks on pääsuvõti (*passkey*), mis kasutab WebAuthni APIt ja CTAP2-protokolli.

5.2.1 WebAuthn

WebAuthn on W3C standard [181], mis defineerib autentimise veebirakenduses koos asjakohaste protokollide ja APIdega. WebAuthni spetsifikatsioon on valminud FIDO Alliance'i ja World Wide Web Consortiumi (W3C) koostöös ja on avaldatud W3C soovitusena. WebAuthni abil on võimalik autentimiseks kasutada FIDO2 (CTAP2) seadmeid ja tagasiühilduvuse raames ka FIDO U2F (CTAP1) tüüpi autentimisseadmeid.

WebAuth võimaldab kasutada paroolivaba autentimislahendusena pääsuvõtmeid, mida kirjeldame täpsemalt alapeatükis 5.2.2.

Uutes süsteemides tuleb järgida standardit WebAuthn Level 2 [182]. Ettevalmistamisel on ka standard WebAuthn Level 3 [183].

WebAuthni toe lisamine infosüsteemidele ei ole triviaalne, see nõuab nii tuge veebirakenduse raamistiku autentimisfunktsioonidelt kui ka rakenduse kasutajate halduse meetodite täiendamist WebAuthni kasutatavate avalike võtmete haldusega. Valmis raamistike korral võib liidestamine olla lihtsam.

WebAuthn defineerib oma autenturi mudeli, CTAPiga ühilduv autentur on selle mudeli üks võimalik teostus. WebAuthn jagab autentureid:

- ühendatuse järgi: platvormiautentur on seotud klientseadmega ega pole sellest eraldatav. Platvormiülene autentur (*cross-platform authenticator*) on iseseisev autentur, mida saab kasutada mitme erineva klientseadmega. Kommunikatsiooni vahendava sobiva kihi olemasolu korral on võimalik (nt mobiiltelefoni sisseehitatud) platvormiautenturit kasutada mingis teises kontekstis platvormiülese autenturina;
- privaatvõtmete võtmete hoidmise viisi järgi: kohalike mandaatidega (*resident credential*, võtit ja sellega seotud atribuute hoitakse autenturi kohalikus hoidlas) autentur ja võtmeid võtmepidemesse (*key handle)*) kodeeriv autentur. Autentur võib toetada mõlemat mandaatide (võtmete) hoidmise viisi, mandaadi loomisel saab valida, kas mandaatide hoidmine autenturis on kohustuslik;
- kasutajasamasuse kontrollimise (*user verification*) toetatuse järgi: mitmikautentimist (*multi-factor authentication*) toetavad autenturid ja seda mittetoetavad autenturid. Mitmikautentimist toetav autentur võib toetada ka rohkem kui kahte autentimistegurit.

CTAP omakorda defineerib sõnumid FIDO-kliendi (ehk lõppkasutajaseadmes oleva FIDO-tarkvarapinu) ja tegeliku autenturi vahel toimuvaks suhtluseks. CTAP1 on U2F ühilduv, CTAP2 on FIDO2 muude standarditega kooskõlaline uuem protokoll. Reeglina võimaldavad FIDO2-kliendid kasutada ka U2F-autentureid. Madala taseme transpordiprotokoll jääb mõlema CTAPi versiooni jaoks samaks, kirjeldatakse USB, NFC ja BLE (*Bluetooth Low Energy*) kasutamist.

[5] https://fidoalliance.org/specs/fido-v2.2-ps-20250714/fido-client-to-authenticator-protocol-v2.2-ps-20250714.pdf

5.2.2 Pääsuvõtmed

Pääsuvõti[6] on paroolivaba autentimismehhanism, mis kasutab WebAuthni APIt ja CTAP2-protokolli. Sisuliselt on tegemist avaliku võtmega krüptograafiat kasutava autentimismehhanismiga, mis kasutab autenturiga suhtlemiseks CTAP2-protokolli. Sõltuvalt seadistusest võib autentimist võimaldav server lubada kasutajal lisada sama konto külge mitu erinevat pääsuvõtit.

WebAuthni spetsifikatsioon [182] ütleb, et privaatvõtit ei tohi kunagi paljastada kolmandale poolele ega ka autenturi omanikule. Samas võimaldavad osad platvormid pääsuvõtmete privaatvõtmeid pilve varundada ja seadmete vahel sünkroniseerida (näiteks Apple iCloud Keychain[7], Google Password Manager[8]). Pääsuvõtmete privaatvõtmeid saab hallata ja sünkroniseerida ka pilvepõhiste paroolihaldurite abil. Vastav protsess pole standarditud ning sõltub platvormist. Enne niisuguse sünkroniseerimis- ja varundusfunktsiooni kasutamist tuleks kontrollida, kas platvormi spetsifikatsioonis on kirjeldatud privaatvõtmete krüptimise viisi. Kui tegemist pole avaliku protokolli ja lähtekoodiga, siis on keeruline kontrollida, kas privaatvõtmed on enne varundamist ja sünkroniseerimist krüptitud viisil, mis ei võimalda teenuseandjal neid dekrüptida.

5.3 Web eID

Web eID on üldnimetus 2021. aastal kasutusele võetud tarkvarakomponentidele, mis võimaldavad veebibrauseris kiipkaardi vahendusel autentida ja signeerida. Web eID koosneb kolmest komponendist: JavaScripti teegist, veebilehitseja laiendusest ja omarakendusest (*native application*). Web eID arhitektuuri kirjeldus on avaldatud GitHubis[9]. Web eID portaal ja proovikeskkond asuvad veebilehel `https://web-eid.eu/`.

Web eID võimaldab kasutada lisaks Eesti riigi poolt väljastatud kiipkaartidele ka Läti, Leedu, Soome, Horvaatia, Tšehhi ja Belgia ID-kaarte[10].

Web eID kasutamiseks on vaja uuendada nii serveri kui kliendi tarkvara. Kui e-teenus kasutab ID-kaardi liidestamiseks varasema põlvkonna tarkvara, siis tuleb teha mõningaid muudatusi Web eID kasutuselevõtuks. Web eID integreerimiseks e-teenustesse on loodud teegid koos juhiste ja näidistega Java, .NET ning PHP platvormidele[11].

Kliendi poolel on Web eID kasutamiseks vaja uuendada ID-kaardi tarkvara, mille käigus paigaldatakse Web eID brauserilaiendus ja Web eID kasutamiseks vajalik omarakendus. Aruande kirjutamise ajal on Web eID veel tagasiühilduv eelmise põlvkonna ID-kaardi tarkvaraga, mistõttu on kuni 2026. aasta novembrini võimalik autentida ka e-teenustesse, mis nõuavad autentimist TLS-CCA abil. Web eID arendamise ajakava näeb ette, et tagasiühilduvus kaob 2026. aasta novembris[12].

Võrreldes varem kasutusel olnud tarkvaraga muudab Web eID peamiselt autentimisega seonduvat. Web eID autentimisprotokollil on ühiseid jooni nii TLS-CCA kui WebAuthniga, kuid erinevalt TLS-CCAst toimib Web eID rakenduskihis ja autentimistõend edastatakse REST API kaudu. Seetõttu ei nõua Web eID veebiserverilt TLSi eriseadistust.

[6] `https://fidoalliance.org/passkeys/`
[7] `https://developer.apple.com/passkeys/`
[8] `https://developers.google.com/identity/passkeys`
[9] `https://github.com/web-eid/web-eid-system-architecture-doc`
[10] Riikide nimekiri pärineb aruande kirjutamise ajal Web eID veebilehel (`https://www.id.ee/artikkel/web-eid/`) toodud andmetel. Tehnilises plaanis on olemas võimalus võtta kasutusele ka teiste Euroopa riikide ID-kaarte.
[11] `https://www.id.ee/artikkel/web-eid/#mida-pean`
[12] `https://www.id.ee/artikkel/web-eid/#ajakava`

Sarnaselt varem kasutatud autentimisprotokollile loob Web eID-d kasutav e-teenus juhuslikult genereeritud pretensiooni (*challenge*), mille klient peab ID-kaardil oleva autentimiseks kasutatava privaatvõtmega signeerima. Erinevalt eelmise põlvkonna tarkvarast ei puhverda Web eID enam ID-kaardi PIN1-te, mistõttu igal autentimisel küsitakse seda uuesti; seetõttu ei ole pärast e-teenusest väljalogimist vaja brauserit sulgeda. Teise uuendusena küsib Web eID autentimisprotsessi alguses kliendilt luba sertifikaadi edastamiseks e-teenusele. Täpsem protokollikirjeldus on toodud Web eID dokumentatsioonis [184, joonis 5].

Web eID muudab mõnevõrra turvagarantiisid. Varem kasutusel olnud TLS-CCA erines paljudest levinud autentimisprotokollidest, nagu OpenID Connect, Mobiil-ID ja Smart-ID selle poolest, et TLS-CCA takistas seansi kaaperdamist ja vahendusründeid. Kuna Web eID ei kasuta TLS-CCAd, siis pole Web eID TLS-seanss krüptograafiliselt seotud kliendi võtmepaariga ning seetõttu tuleb vahendusrünnete ja seansi kaaperdamise takistamiseks kasutada teisi turvameetmeid. Näiteks vahendusrünnete tuvastamiseks laseb Web eID kliendil allkirjastada lisaks e-teenuse poolt genereeritud pretensioonile (*challenge*) ka veebisaidi allika (*origin*). Täpsema ülevaate seansi kaaperdamise ja vahendusrünnete riskist leiab Web eID turvaanalüüsist [185], analüüsi võtab kokku ülevaade ID.ee lehel[13].

5.4 OAuth 2.0

OAuth 2.0 [186, 187] on karkass (*framework*), mis defineerib volitusprotokolli (kuigi täpsem oleks öelda „delegeerimisprotokolli“). OAuth võimaldab kolmanda poole klientrakendusel saada piiratud pääsuõiguse HTTP-teenustele. Kõige tavalisem kasutus on selline, kus kasutaja annab ühele teenusele piiratud päringuõiguse mingis teises teenuses hallatavale ressursile.

Näiteks: kasutaja annab fototöötlusrakendusele piiratud ajaks õiguse lugeda ja muuta fotohoidlas asuvaid ja talle kuuluvaid pilte. Kasutaja on ressursi omanik. Fototöötlusrakendus on klientrakendus. Õigus lugeda ja kirjutada on delegeeritav piiratud õigus. Fotohoidla on teenus, millele klientrakendus juurdepääsu vajab. Pildid hoidlas on ressurss, mille kasutamise jaoks õiguseid delegeeriti (volitus anti).

On oluline silmas pidada, et OAuth 2.0 ei ole universaalne volitustehalduse protokoll, vaid mõeldud ainult konkreetse kasutaja õiguste täielikuks või osaliseks delegeerimiseks mõnele rakendusele.

OAuth-karkassile on ehitatud palju mitmesuguseid laiendusi, nagu OpenID Connect[14] autentimisinfo standarditud edastamiseks ja User Managed Access[15] pääsupoliitikate haldamiseks kasutajakeskselt, aga ka laiendused protokolli erinevatele aspektidele nagu parameetrite edastamine, klientrakenduse autentimine, volitusserveri parameetrite avastamine, reeglid pääsutõendite kasutamise kohta jms.

OAuthi volitusprotokolli nüansside ja kasutuste erinevates kontekstides kirjeldamisega tegeleb IETFi avatud volitusprotokolli töörühm[16].

OAuthi turvamudelit, st ohtusid millega on protokolli projekteerimisel arvestatud ning seda kuidas nende vastu kaitstakse, kirjeldavad IETF RFC 6819 [188] ja IETF RFC 9700 [189]. Oluline on märkida, et OAuth ei ole autentimisprotokoll[17] ja selle kasutamine autentimiseks ilma

[13] https://www.id.ee/artikkel/sessiooni-kaaperdamise-runnaku-ja-vahendusrunde-risk-web-eid-s/
[14] https://openid.net/specs/openid-connect-core-1_0.html
[15] https://kantara.atlassian.net/wiki/spaces/uma/overview
[16] https://datatracker.ietf.org/wg/oauth/about/
[17] https://oauth.net/articles/authentication/

sobivate täiendusteta (nagu seda enne protokolli OpenID Connect standardimist sageli tehti) võib põhjustada mitmesuguseid turvanõrkusi[18]. Näiteks ei ole klientrakendusel üldjuhul võimalik kontrollida, et pääsutõend väljastati just talle või sama seansi vältel.

OAuthi abil väljastatud pääsutõendeid saab kasutada ka muudes protokollides peale HTTP, kuid OAuth-protokolliga kirjeldatud sõnumite vahendamine üle mingite muude protokollide peale HTTP pole OAuthi turvamudelis kirjeldatud ning sellised kasutusmallid vajaksid eraldi turvaanalüüsi.

OAuth 2.0 kavandamisel on arvesse võetud varasemate versioonide (1.0 ja 1.0a) kogemusi, kuid ta erineb teostuselt neist oluliselt. OAuth 2.0 parandas mitmed varasemates versioonides esinenud nõrkused; varasemad versioonid ei ole enam kasutamiseks turvalised.

OAuth 2.0 turvaliseks kasutamiseks tuleb OAuth 2.0 õigesti seadistada[19][20]. Praegu veel kavandi kujul olev OAuth 2.1 peaks koondama OAuth 2.0 head tavad põhistandardisse ning muutma mõned turvalise seadistuse aspektid kohustuslikuks.

5.4.1 Protokolli kirjeldus

OAuth2 protokolli järgi väljastab volitusserver klientrakendusele pääsutõendi(d) (*access token*), mida klientrakendus saab kasutada (HTTP) ressursiserverile päringute tegemiseks.

Abstraktsel tasemel koosneb protokoll kolmest sammust:

1. volituskinnituse (*authorization grant*) hankimine. Klientrakendus hangib kasutajalt (võimalik, et volitusserveri vahendusel) volituskinnituse. Volituskinnituse hankimise viise kirjeldas täpsemalt 2023. aasta aruande alapeatükk 5.1.4 [8];
2. pääsutõendi (*access token*) hankimine. Volitusserver väljastab volituskinnituse alusel klientrakendusele pääsutõendi. Pääsutõendite olemust ja struktuuri kirjeldas täpsemalt 2023. aasta aruande alapeatükk 5.1.5 [8];
3. pääsutõendi kasutamine. Klient kasutab pääsutõendit päringus ressursiserverile.

Erinevate omaduste ja võimalustega klientrakenduste teenindamiseks defineerib OAuth erinevad viisid volituskinnituse ja pääsutõendi hankimiseks.

Tavaliselt kasutatakse volituskoodiga kinnitamist (*Authorization Code Grant*), kus pääsutõendi väljastamine algab veebibrauseri vahendusel edastatava volitustaotlusega. See võimaldab volitusserveril korraldada kasutaja autentimist ja/või küsida kasutaja nõusolekut pääsutõendi väljastamise kohta.

Volitusserver võib defineerida ja pääsutõenditega siduda skoobid. OAuth standard ei defineeri skoobi tähendust ega kasutusi, see on jäetud konkreetsete teostuste ülesandeks – skoop on silt, mille tähenduse defineerib volitusserver. Skoope saab kasutada ja on kasutatud näiteks OAuth protokolli laienduste kokkuleppimiseks – OpenID Connecti kasutamiseks peab klient kasutama skoopi `openid`. Skoope kirjeldas täpsemalt 2023. aasta aruande alapeatükk 5.1.3 [8].

[18] https://portswigger.net/web-security/oauth
[19] https://cheatsheetseries.owasp.org/cheatsheets/OAuth2_Cheat_Sheet.html
[20] https://www.rfc-editor.org/rfc/rfc9700

Pooled

Loogilises vaates on igas OAuth voos vähemalt kolm poolt: klientrakendus, volitusserver ja ressursiserver. Lisaks neile osaleb protokollis tavaliselt ka neljas pool (ressursi omanik), enamasti inimkasutaja.

Klientrakendus (*Client*) on pool, keda kasutaja volitab enda õigustes tegutsema. Klientrakenduste ehitamisel peab arvestama, et protokoll jagab need avalikeks (*public*) ja saladusevõimelisteks (*confidential*). Esimese olek on selle kasutajatele (ja võimalikele ründajatele) täielikult nähtav ja volitamisserveri võimalused sellist klienti turvaliselt autentida seega piiratud. OAuthi põhistandard kirjeldab olukorda, kus klientrakendus on veebirakendus, hilisemad laiendused [187, 190] spetsifitseerivad ka kasutamise omarakendustes (*native application*, sh mobiilirakendused) ja tervenisti brauseris käivates rakendustes.

Ressursiserver (*Resource Server*) tarnib klientrakendusele kaitstud (HTTP) ressursi. OAuthi kontekstis peab ressursiserveri arendaja mõtlema läbi viisid, kuidas selgitada välja ressursiserverile esitatud pääsutõendi tegelik tähendus ja kehtimine – kelle kinnitusel ja mille jaoks see tegelikult väljastati ning kas see on piisavalt värske, et selle alusel pääsuotsuseid teha. Turvakriitilistes süsteemides võib olla vajalik kehtivusinfo sage või igakordne kontrollimine volitusserverist. Arendamisel peaks arvestama pääsutõendite kasutamise spetsifikatsioonidega, nagu RFC 6750 [191].

Pääsutõendi tähendus on tavaliselt see, et ressursi omanik delegeerib klientrakendusele mingid oma õigused ressursi suhtes. Pääsuotsuse tegemisel peab ressursiserver muuhulgas kontrollima, kas ressursiomanikul endal ikka olid delegeeritavad õigused.

Volitusserver (*Authorization Server*) väljastab klientrakenduse soovil ja vajalike tingimuste täidetuse korral klientrakendusele pääsutõendi (*access token*). Tavaliselt on üheks tingimuseks ressursiomaniku kinnitus (*authorization grant*).

Klientrakendus peab olema volitusserveris eelnevalt registreeritud. Leiduvad protokollilaiendused, mille abil saab klientrakendus end dünaamiliselt registreerida [192] ja sellist registreeringut hallata [193]. Klientrakenduste ja nendega seotud naasmisaadresside (URL) registreerimine ja kontrollimine on vajalik vältimaks kasutaja võimalikku suunamist ründaja ettesöödetud aadressile. Olukord, kus kasutaja saadetakse pärast edukat ja korrektsel aadressil toimunud autentimist ja volituskinnituse küsimist võltssaidile, lisab sellisele võltssaidile teenimatut usaldusväärsust. Selle probleemiga peab arvestama ka klientrakenduste dünaamilise registreerimise võimaldamisel.

Pääsutõendite (ja muude volitusserveri väljastatud piletite ja tõendite) kehtivuse ja sisu kontrolliks ja edastamiseks on laiendus Token Introspection, RFC 7662 [194].

Ressursiomanik (*Resource Owner*) on kasutaja või süsteem, kellel või millel on õigus ja võime otsustada kaitstud ressursi kasutamise üle. Kui ressursiomanik on inimene, siis võib tema kohta öelda ka lõppkasutaja (*End-user*).

5.4.2 Omamistõendus

OAuth esitajatõendit (*bearer token*) saab kasutada ilma identiteeditõestuse (*proof of identity*) ja muude piiranguteta, mistõttu valedesse kätesse sattunud esitajatõendi abil on võimalik saada

juurdepääs vastava tõendiga seotud ressurssidele [188]. Traditsioonilised OAuth-pääsutõendid on olemuselt esitajatõendid, mistõttu eelnevalt kirjeldatud probleem kehtib ka pääsutõendite kontekstis.

Selle probleemi lahendamiseks saab rakendada OAuth 2.0 omamistõestust (*Proof of Possession (DPoP)*), mis defineeriti 2023. aastal avaldatud RFC 9449 standardikavandiga [195]. OAuth-omamistõestus kasutab avaliku võtmega krüptograafiat, et siduda pääsutõend seda tõendit kasutava rakendusega. Sisuliselt kasutab klientrakendus privaatvõtit, et koostada signatuur ning pakendada see JWT-struktuuri (*JSON Web Token*). Vastavat signeeritud JWT-struktuuri käsitletakse omamistõendina. Ressursiserver võtab omamistõestust sisaldava JWT vastu ning kontrollib signatuuri, verifitseerides, et signatuuriga seotud avalik võti kattub pääsutõendiga seotud avaliku võtmega.

Klientrakenduse, volitusserveri ja ressursiserveri vahel toimuv omanditõestusega seotud pääsutõendi koostamise ja verifitseerimise protokoll sisaldab nelja põhisammu.

1. Klientrakendus saadab volitusserverile volituskinnitust (*authorization grant*) sisaldava tõendipäringu (*token request*), eesmärgiga saada pääsutõend (või värskendamispilet (*refresh token*). Klientrakendus lisab vastava HTTP-päringu päisesse omamistõestuse.
2. Volitusserver seob pääsutõendi kliendi omamistõestuses oleva avaliku võtmega, mistõttu tuleb pääsutõendi kasutamiseks tõendada vastava privaatvõtme omamist. Juhul kui volitusserveri poolt väljastatakse värskendamispilet, siis see seotakse samuti omamistõestuses oleva avaliku võtmega. Volitusserver saadab pääsutõendi/värskendamispileti kliendile.
3. Pääsutõendi kasutamiseks peab klient tõendama, et tal on võimalik kasutada pääsutõendiga seotud privaatvõtit. Selleks peab klient koostama omamistõestuse ning lisama selle ressursipäringu päisesse. Ressursipäringuga tuleb kaasa panna pääsutõendiga seotud avalik võti (selle võib edastada JWT osana). Klient edastab ressursipäringu ressursiserverile.
4. Ressursiserver kontrollib, kas pääsutõendiga seotud avalik võti kattub päringu omamistõestuse avaliku võtmega. Samuti kontrollib ressursiserver, kas omamistõestuses olev pääsutõendi räsi kattub esitatud pääsutõendi räsiga ning kas omamistõestuses olev URI kattub päringus oleva URIga. Kui verifitseerimine õnnestub ja pääsutõend kehtib, siis edastab ressursiserver kliendile vastava ressursi.

Iga HTTP-päringu jaoks tuleb koostada uus unikaalne omamistõestus. Oluline on aru saada, et JWT-kujul omamistõestus ei ole pääsu reguleerimiseks piisav. Omamistõestuse põhist autentimist kirjeldab täpsemalt RFC 9449 jaotis 7.1 („The DPoP Authentication Scheme“) [195].

Enne omamistõestuse rakendamist tuleb hoolikalt läbi töötada RFC 9449 11. peatükk, mis käsitleb omamistõestuse turvalisusega seotud aspekte ning annab juhiseid omamistõestuse turvaliseks kasutamiseks [195].

5.5 Atribuutide esitamise protokollid

Nagu sätestatud komisjoni rakendusmäärustes 2024/2982[21] ja 2024/2977[22], on standardid 18013-5:2021 ja 18013-7:2024[23] Euroopa Liidu digiidentiteedirakendustele kohustuslikud. ISO-18013-5 standard kirjeldab atribuutide esitamist juhul, kui esitaja ja kontrollija on teineteisele lähedal (*proximity verification*). ISO-18013-7 standard kirjeldab atribuutide esitamist juhul,

[21] https://eur-lex.europa.eu/eli/reg_impl/2024/2982/oj/eng
[22] https://eur-lex.europa.eu/eli/reg_impl/2024/2977/oj/eng
[23] Aruande kirjutamise ajaks on ilmunud standardide uuemad versiooonid ISO/IEC 18013-5:2025 ja ISO/IEC 18013-7:2025

kui esitaja ja kontrollija on teineteisest kaugel (*remote verification*). See omakorda kasutab OpenID for Verifiable Presentations (OpenIDVP) ja DC API-põhiseid vooge, nagu on täpsustatud OpenID4VC HAIP (High Assurance Interability Profile) spetsifikatsioonis. Selles alapeatükis anname ülevaate nendest spetsifikatsioonidest.

Järgnevas kirjeldame mõningaid protokolle atribuutide esitamiseks kontrollijale. See alapeatükk on tugevalt seotud alapeatükiga 4.6, milles kirjeldatakse atribuutide esitamise *vorminguid*. Kasutame järgmisi termineid:

- **tõend** (*certificate*) on kinnitus või dokument, mis vastavalt oma tüübile kinnitab tõendi subjekti kohta mingeid väiteid. Kasutame seda mõistet digitaalse juhiloa ja erinevat tüüpi kontrolitavate mandaatide üldistusena
- **tõendi esitus** (*certificate presentation*) on ühe või mitme tõendi andmetest koosnev kogum, mis kinnitab tõendi(te) omaniku privileegi või õigust
- **digikukkur** (*wallet*) on tõendi omaniku (nuti)seadmesse paigaldatud rakendus, mis lubab tõendi omanikul esitada kontrollijale oma tõendi(te) andmetest koosneva tõendi esituse
- **seadmesisene voog** (*same-device flow*) on andmevoog, milles digikukkur ja brauser asuvad samas seadmes
- **seadmeväline voog** (*cross-device flow*) on andmevoog, milles digikukkur ja brauser asuvad eri seadmetes
- **väljaandmispäring** (*issuing request*) on sõnum, mille kasutaja saadab väljaandjale tõendi saamiseks
- **väljaandmisvastus** (*issuing response*) on sõnum, millega väljaandja saadab kasutajale taotletud tõendi
- **esituspäring** (*presentation request*) on sõnum, mille kontrollija saadab kasutajale tõendi esituse saamiseks
- **esitusvastus** (*presentation response*) on sõnum, millega kasutaja saadab kontrollijale tõendi esituse

5.5.1 ISO 18013-5 ja ISO 18013-7

Standard ISO/IEC 18013-5[24] spetsifitseerib eelkõige elektroonilise juhiloa kasutamist, kuid standardi defineeritud tõendiformaat toetab igasuguste andmete hoidmist ja esitamist. Formaati tutvustab lähemalt alapeatükk 4.6.2. Peale tõendi andmeformaadi ja esituse defineerimise keskendub standard tõendite esitusele kahe füüsilise seadme vahel.

Suhtlus esitaja ja kontrollija vahel algab seadmete kaasamisest (*device engagement*). Selleks saadab esitaja kontrollijale spetsiaalse andmeobjekti `DeviceEngagement`, mis sisaldab edaspidiseks suhtlemiseks vajalikku infot. See struktuur edastatakse NFC kaudu või QR-koodi abil ja see vajab füüsilist lähedalolemist. Selle abil loovad esitaja ja kontrollija suhtluskanali. Kontrollija saadab esitajale esituspäringu `DeviceRequest`. Esitaja saadab kontrollijale esitusvastuse, kasutades struktuuri `DeviceResponse`. Kõik struktuurid kodeeritakse CBOR-formaadis. Järgneva seadmelt-seadmele andmevahetuse puhul on kolm alternatiivi: Bluetooth low energy, Wifi-Aware või NFC.

Eraldi on olemas standard ISO/IEC 18013-7[25], mis käsitleb tõendi esitamist juhul, kui esitaja ja

[24] https://www.evs.ee/et/iso-iec-ts-18013-7-2025
[25] https://www.evs.ee/et/iso-iec-18013-7-2025

kontrollija pole füüsiliselt teineteisele lähedal. Selle aruande kirjutamise seisuga pakub standard kolme lahendust:

1. esitaja loob kontrollijaga suhtluskanali, kasutades infot (nt URI) spetsiaalsest struktuurist `ReaderEngagement` (CBOR-kodeeritud), mille ta on eelnevalt kontrollija käest saanud;
2. esitaja kasutab tõendi esitamiseks OID4VP-protokolli (vt ptk 5.5.2);
3. esitaja kasutab tõendi esitamiseks Digital credentials API-protokolli (vt ptk 5.5.3).

Esitaja ja kontrollija seadmed peavad toetama vähemalt ühte neist meetoditest.

ISO/IEC 18013-7:2025 mainib, et kaugesituse korral on võimalik vahendusrünne. See juhtub siis, kui ründaja algatab suhtluse ausa kontrollijaga, et luua link, mis seejärel edastatakse ohvrile (teeseldes ausat kontrollijat), et ohver viiks protsessi ründaja nimel lõpule. Sellisel juhul annab kontrollija privileegi ründajale, kes kasutab selleks ära ausa kasutaja õigusi. Standard arutab, kuidas vältida selliseid ründeid eri lahenduste korral.

Lahenduse 1 korral peab kasutaja edastama oma domeeni allika (*domain origin*), kuid mõned brauserid seda kahjuks ei võimalda. Lahenduse 2 korral peab kontrollija siduma kasutaja seansi vastava OID4VP seanssiga, kuid see paneb kontrollijale suurema vastutuse. Lahenduse 3 korral peab kasutaja edastama oma domeeni allika, mis on selle lahenduse korral täiesti tehtav.

5.5.2 OpenID4VP

OpenID for Verifiable Presentations (OpenID4VP)[26] on OAuth 2.0 protokollil baseeruv tõendite esituste pärimist ja edastamist kirjeldav spetsifikatsioon. Kontrollija, enamasti veebileht, pärib tõendit digikukrult, mis võib asuda brauseriga nii samas kui ka eraldi seadmes.

Selle aruande kirjutamise seisuga toetab protokoll kolme tüüpi tõendeid: W3C Verifiable Credential Data Model (vt ptk 4.6.3), ISO/IEC 18013-5 mdoc (vt ptk 4.6.2) ja SD-JWT-based Verifiable Credentials (vt ptk 4.6.1).

Protkoll eristab kahte voogu, sõltuvalt digikukru paiknemisest:

- seadmesisese voo korral (digikukkur ja brauser asuvad samas seadmes) on tõendi päring URL, mis käivitab digikukru. Digikukkur saadab vastuse, käivitades kontrollija saadetud URLi läbi sama brauseri, mille kaudu oli loodud esialgne ühendus. See võimaldab kontrollijal veenduda, et algne seanss oli loodud samas seadmes, kus asub digikukkur;
- seadmevälise voo korral (digikukkur ja brauser asuvad eri seadmetes) esitab kontrollija tõendi päringu QR-koodi abil. Kasutaja skaneerib QR-koodi digikukru seadmega. Vastuse saadab digikukkur otse kontrollijale.

Oluline on märkida, et kontrollija saadetud QR-koodi on ründajal võimalik taasesitada ohvrile (esitajale) teises kontekstis ja sedasi esitada tõendeid ohvri nimel. Selle ründe eest hoiatab ka protokolli kirjeldus.

OpenID4VC HAIP (*High Assurance Interability Profile*)[27] määrab kindlaks meetodid, mida kasutada OpenID4VP ümbersuunamisel (nt URLi edastamine kukru ja brauseri vahel), et kindlustada seda õngitsusrünnete vastu. Kontrollmeetmeid peavad rakendama nii digikukkur kui ka kontrollija. Dokumendis on mainitud, et kaitsemeetmed toimivad ainult seadmesisese voo puhul.

[26] https://openid.net/specs/openid-4-verifiable-presentations-1_0.html
[27] https://openid.net/specs/openid4vc-high-assurance-interoperability-profile-1_0-final.html

Seadmevälise voo puhul ei ole võimalik vajalikke meetmeid kohaldada. Siiski on seadmesisene voog soovitatav, kuid mitte kohustuslik, sest kindlus õngitsemise vastu on saavutatav muude vahenditega.

Seadmevälise voo jaoks vahendusründeid täielikult leevendavaid meetmeid ei ole. Spetsifikatsioonis kirjeldatud alternatiivne meetod, mis kasutab brauseri Digital Credentials API-liidest (vt ptk 5.5.3), välistab vahendusründed ka seadmevälises voos.

5.5.3 Digital credentials API

Digital Credentials API (DC API) spetsifikatsioon[28] on selle aruande kirjutamise ajal kavandi staatuses (W3C Working Draft). DC API defineerib liidese, mis vahendab digitaalsete tõendite esitamist ja väljaandmist. Vastavalt spetsifikatsioonile on API eesmärk toetada järgmisi eesmärke:

- Hoida taotlemise ja väljaandmise aktid eraldi konkreetsetest esituse ja väljaandmise protokollidest, seega võimaldades protokollide ja mandaadivormingute laiendatavust.
- Nõuda esitus- ja väljaandmispäringute *krüptimata* jätmist, võimaldades riskianalüüsi.
- Eeldada läbipaistmatuid, st *krüptitud*, esitus- ja väljaandmisvastuseid, mis võimaldavad väljaandjatel, kontrollijatel ja esitajatel valida, kus avaldada potentsiaalselt tundlikku teavet.
- Nõuda esitus- või väljaandmispäringute esitamiseks mööduvat aktiveerimist (*transient activation*), tagades, et toimingud ei ole võimalikud ilma kasutaja aktiivse osalemise ja iga toimingu kinnitamiseta.
- Kui esituspäringule vastab mitu erinevat mandaati, võimaldada kasutajal ise valida sobiv mandaat.
- Kui väljaandmispäringut võib esitada kasutades mitu erinevat digikukrut (digikurku rakendust), võimaldada kasutajal ise valida sobiv digikukkur.
- Võimaldada platvormidel pakkuda turvalisi seadmetevahelisi (*cross-device*) esitus- ja väljaandmispäringuid, koos läheduskontrolliga.

Toetatud esituse ja väljaandmise protokollide seas on mainitud OpenID4VP ja ISO-18013-7.

DC API on sisuliselt brauseri liides, mis vahendab digikukru ja veebilehe vahel nii tõendi väljastamist kui ka esitamist. Liides parandab nii turvaomadusi kui ka kasutajakogemust võrreldes URLidel põhinevate lahendustega, seda eriti seadmevälises voos, kus brauser ja digikukkur ei paikne ühes ja samas seadmes. Tegemist on uue liidesega, mille tugi saabus brauseritesse alles 2025. aasta teises pooles. Mobiilisetest brauseritest toetavad liidest kirjutamise hetkel Apple Safari, Microsoft Edge ja Google Chrome. Viimase tugi tekkis alles 2026. aasta jaanuaris. Mozilla loobus DC API arendusest 2024. aastal negatiivse seisukohaga[29,30]. Sellest hoolimata ilmusid 2025. aasta lõpus ka Mozilla probleemihaldussüsteemi Bugzilla DC API toetusega seotud tööülesanded[31].

DC API on kontseptuaalselt sarnane Web Authentication APIga, kuid on spetsiaalselt kavandatud Digital Identity Walleti kasutusmalli silmas pidades. DC API spetsifikatsioon ei nõua seadmetevaheliste ühenduste loomiseks konkreetse protokolli kasutamist. Sarnaselt Web

[28] https://www.w3.org/TR/digital-credentials/
[29] https://github.com/WebKit/standards-positions/issues/332#issuecomment-2019400609
[30] https://github.com/mozilla/standards-positions/issues/1003#issuecomment-2111403209
[31] https://bugzilla.mozilla.org/show_bug.cgi?id=2005077,https://bugzilla.mozilla.org/show_bug.cgi?id=2004828

Authentication APIga (vt ptk 5.2.1) võib DC API kasutada kahe lähestikku oleva seadme (digikukru ja brauseri) vahel[32] ühenduse loomiseks CTAP- ja Bluetooth-protokolle. Tehingu õnnestumiseks peavad olema rahuldatud järgmised tingimused:

- kasutajaseansid peavad tulenema kas ühest ja samast seadmest või Bluetooth-läheduses olevatest seadmetest;
- kasutaja brauseri edastatud domeeni nimi on võrdne nimega, mida tegelikult kasutab teenuseandja.

Isegi kui kasutaja kinnitab (nt õngitsemise käigus) võltstehingu, tuvastatakse andmete mittevastavus automaatselt ja tehing katkestatakse.

Ründeid ja nende vastaseid kaitsemeetmeid kirjeldab lähemalt digikukru III etapi analüüsi [196] 7. peatükk.

[32] https://developer.chrome.com/blog/digital-credentials-cross-device-ot

6 Teegid ja rakendused

Kui rakenduses on tarvis kasutada mingeid krüptograafilisi primitiive, struktuure või protokolle, siis tuleb selleks alati valida rakenduse keskkonda sobiv kõrge kvaliteediga krüptoteek – keerulisematel juhtudel ka sobitada omavahel krüptoteegi ning arendus- ja käidukeskkondade valikut, sest keskkonnad ei ole krüptograafiliste teekide kättesaadavuses osas võrdsed. Paljudel juhtudel ei ole valitud keskkonna jaoks piisavalt universaalseid krüptoteeke üldse olemas ning sel juhul tuleb valida mõni teek, millel on olemas sobiv mähis (tavaliselt on selleks teegiks OpenSSL).

Krüptograafiliste primitiivide või konstruktsioonide teostamine äriprotsesse toetava rakenduse arenduse käigus ei ole üldiselt hea plaan, kuivõrd selline arendus kipub keskenduma vaid ärirakenduse funktsiooniga seotud vajadustele, mitte aga niivõrd teostatud krüptograafia turvalisuse nõuetele (jättes näiteks tähelepanuta algoritmide kõrvalkanalikindluse).

Kindlasti tuleb arendusprojekti jaoks krüptoteegi valikul ja kasutamisel teha järgnevat.

- Tutvuda enne kasutuselevõttu teegi versioonidega ning võimalike paralleelselt eksisteerivate peaversioonidega (*major version*). Käesolev aruanne ei soovita meelega ühtegi kindlat tarkvaraversiooni, sest olukord muutub väga kiiresti ning põhjapanevaid teadustöid sel teemal ei ole.
- Hinnata, kui suur on teegi üldine populaarsus (*Google Trends* on selleks väga hea vahend), vähepopulaarsetel teekidel on kalduvus muutuda jäätvaraks.
- Hinnata, millise kiirusega on arendusmeeskond seni teegile turvaparandusi väljastanud.
- Hinnata, kui suur meeskond teegi arendamisega tegeleb ning kas võib eeldada meeskonna jätkusuutlikkust arendatava projekti eluea piires. Paratamatult on siinkohal eelistatud teegid, millel on mõne suurkorporatsiooni, valitsuse või suurema MTÜ tugi.
- Kasutada alati teegi mõne toetatud haru viimast versiooni ning hoolitseda, et oleks olemas protseduurid kasutatud teekide korraliseks ülevaatuseks ja uuendamiseks. Tähelepanu tuleb pöörata ka sellele, et süsteemne pakihaldus (näiteks APT, RPM) ei rikuks ära arendaja plaane ega paigaldaks toodangukeskkonda dünaamiliselt lingituna vanemat ning vigast versiooni krüptoteegist.
- Pidada silmas, et teegid sisaldavad ühilduvuse huvides ka mitmesuguste selliste primitiivide ja protokollide teostusi, mida ei peaks üheski uues ega uuendatavas süsteemis enam kasutama. Mingi konstruktsiooni olemasolu teegis ei ole põhjus selle kasutamiseks, iga kasutamine tuleb otsustada pädevatele ja värsketele turvahinnangutele tuginedes.

Selles peatükis vaatleme eelkõige teekide ja arenduskeskkondade viimaseid stabiilseid versioone.

6.1 Kõige levinumad ja olulisemad krüptoteegid: ülevaade

Tabel 6 annab ülevaate kõige levinumatest ja olulisematest krüptoteekidest.

Aruande eelmise versiooniga [8] võrreldes on tabelit 6 täiendatud tulpadega „Postkvant-krüptograafia tugi“ ja "Kasutusala"ning tabelist on välja jäetud tulp „RSA-PSSi tugi“. Aruande järgmistes versioonides võivad lisanduda sarnased tulbad ka muude krüptograafiliste konstruktide jaoks. Üks või teine tulp võib samamoodi tabelist kaduda, kui vastava konstrukti toetamise küsimus minetab päevakajalisuse.

Tabel 6. Krüptoteekide ülevaade (2026. aasta alguse seisuga)

Nimi	Keel(ed)	Kasutusala	PQC tugi[a]	Soovitatud
OpenSSL (libcrypto)	C	kõige mitmekülgsem	jah	jah
LibreSSL (libcrypto)	C	alternatiiv OpenSSLile	ei[b]	ei
BoringSSL	C	Google'i tooted	jah	ei
GnuTLS	C	alternatiiv OpenSSLile	osaline[c]	ei
WolfSSL (wolfcrypto)	C	alternatiiv OpenSSLile	jah	jah
Mbed TLS	C	sardsüsteemid	ei	jah
NSS	C	internetiprotokollid	osaline[d]	jah
Botan	C++	tootmisrakendused	jah	jah
s2n	C	TLSi protokollistik	jah	ei
BouncyCastle	Java, C#	Java ja C# omateek	jah	jah
pyca/cryptography	Python	Pythoni omateek	ei	jah
Libgcrypt	C	üldotstarbeline	osaline[e]	ei
Crypto++	C++	üldotstarbeline	ei	ei
Nettle	C	üldotstarbeline	ei	ei
TinkCrypto	Java, C++, Objective-C, Go, Python	algoritmiagnostilised primitiivid	ei	jah
libsodium	C	algoritmiagnostilised primitiivid	jah	ei
libcrux	Rust	kõrgkindel krüptoprimitiivide teostus	jah	ei
libjade	C	kõrgkindel krüptoprimitiivide teostus	jah, ainult PQC	ei

[a] ML-DSA, ML-KEM, hübriidrežiimid ja X.509
[b] ML-KEM on imporditud, aga API kaudu kättesaadavaks ei ole tehtud.
[c] Eksperimentaalsed postkvant-sertifikaadid.
[d] Puudub postkvant-sertifikaatide teostus.
[e] PQC teostused näivad olevat vananenud.

6.1.1 OpenSSL

1998. aastal selleks hetkeks juba kolm aastat vana teegi SSLeay baasil loodud OpenSSL[1] on kasutuselolevatest laiatarbe-krüptoteekidest pikima ajalooga ning tuntuim krüptograafiat pakkuv teek maailmas. OpenSSL teostab suure hulga krüptoprimitiive (räsialgoritmid, sümmeetrilised ja asümmeetrilised krüptoalgoritmid), vorminguid (näiteks X.509 ja CMS) ja protokolle (SSL, TLS, DTLS). Kuigi OpenSSL oli vaba tarkvarana pikalt väga laialt kasutusel, püsis selle arendusaktiivsus vähese välise toetuse ja vähese rahastuse tõttu 2014. aastani suhteliselt tagasihoidlikuna. Häirekellaks kasutajatele sai mitmete turvanõrkuste (nagu näiteks Heartbleed[2]) avastamine, pärast mida on märkimisväärselt kasvanud nii leitud turvanõrkuste arv, OpenSSLi arendustempo kui ka projekti rahaline toetus, sealhulgas paljude suurfirmade poolt. Mitmete uute täiskohaga arendajate abiga anti 2021. aasta septembris välja ka uus versioon 3.0, mis on Apache 2.0 litsentsi all, ent jätkuvalt toetatakse veel ka vanemat OpenSSL 1.1.1 seeriat.

Kuigi üleminek OpenSSLi versioonile 3.0 ei tähenda ilmtingimata palju muudatusi seda kasutavate programmide lähtekoodis, on siiski muutunud mitmed programmeerimisliidesed, masinkoodiliidesed ja ka mitmed olulised vaikekäitumised. Seetõttu tuleks uuele versioonile üleminekul OpenSSL-teeki kasutavad lähtekoodid vastavalt OpenSSLi migreerimisjuhendis[3] kirjeldatud muudatustele kriitiliselt üle vaadata ja vajaduse korral korrigeerida. Eriliselt tähelepanelik tuleb olla nende muudatuste osas, mis ei pruugi ehitus- ja käitusaegselt otseseid vigu anda, kuid millest tulenevalt võib programmi käitumine keeruliselt tuvastatavalt ebakorrektseks muutuda. Näiteks on oht võimaldada mäluturvalisuse rikkumist, kui ignoreerida OpenSSLi versiooniga 3.0 lisatud nõuet mõningate `EVP_PKEY_` nimeruumi funktsioonide tagastusväärtusi mitte muuta. Soovitame võimaluse korral lõpetada kõikide taunitud liideste kasutamine, mida ka OpenSSLi versiooniga 3.0 rohkesti lisandus.

6.1.2 wolfSSL

wolfSSL[4] on nii vabavaralise kui ka kommertslitsentsi all pakutav kergekaaluline TLSi ja DTLSi teostus, mis on orienteeritud kasutusele manusseadmetes. wolfSSL on ristkompileeritav paljudele riistvaraarhitektuuridele ja operatsioonisüsteemidele ning pakub ka OpenSSL-teegiga ühilduvat liidest. wolfSSLi (algselt CyaSSL) on arendatud alates aastast 2004, tema arendusaktiivsus on vähemalt viimase kümne aasta jooksul järjepidavalt kasvanud ning tema arendajad on olnud kiired lisama tuge uutele standarditele ja algoritmidele, sealhulgas postkvant-krüptograafiale. Siin nimekirjas olevatest teekidest pakub ta hetkel ainsana DTLS 1.3 tuge. wolfSSL madalama taseme krüptograafiliste primitiivide moodulil wolfCrypt on ka FIPS 140-2 sertifikaat. wolfSSLi nõrkuseks võib siiski lugeda tema suhteliselt vähest populaarsust, mille üheks põhjuseks võib olla ka see, et wolfSSL teeki pakutakse vaid GPL 2.0+ ja kommertslitsentsi all.

6.1.3 Mbed TLS

Mbed TLS[5] on TLSi ja SSLi teostusi pakkuv teek. Mbed TLSi juured on Christophe Devine'i loodud teegis XySSL, mille arendus jätkus pärast soikumist eraldiseisva PolarSSLi nimelise projektina. 2014. aastal ostis PolarSSL-teegi ära Arm ning nimetas selle ümber oma Mbed manusseadmete

[1] https://www.openssl.org/
[2] https://heartbleed.com/
[3] https://www.openssl.org/docs/man3.0/man7/migration_guide.html
[4] https://www.wolfssl.com/products/wolfssl/
[5] https://www.trustedfirmware.org/projects/mbed-tls/

platvormi järgi. Mbed TLS sisaldab ka Armi seadmete turvaarhitektuuri PSA (Platform Security Architecture) API etalonteostust. Aastast 2020 on Mbed TLS Trusted Firmware avatud valitsemise projekti juhtimise all, mille juhatusse kuulub ka Arm. Kuigi Apache 2.0 litsentsiga Mbed TLS on populaarne eelkõige manusseadmetes, on ta kasutust leidnud ka mitmetes teistes avatud lähtekoodiga programmides ning on saadaval enimkasutatud Linuxi distributsioonides. Mbed TLSi kiiret arendust veavad eest eelkõige Armi enda arendajad, kuid arenduses osalevad ka mõned teised manusseadmete tarkvaraga tegelevad firmad. Aastatel 2021–2022 tehti Mbed TLSi koodihoidlasse enam kui 2000 koodikehtestust rohkem kui OpenSSLi koodihoidlasse.

6.1.4 NSS

NSS ehk Network Security Services[6] on TLSi, SSLi, PKCS#5, PKCS#7, PKCS#11, PKCS#12, S/MIME'i ja X.509 v3 teostusi pakkuv teek. NSSi lähtekoodi juured ulatuvad üheksakümnendatesse, Netscape'i veebilehitsejate lähtekoodi, mis pärast Netscape Communicator 4.0 väljalaset pakendati esmalt teegiks nimega HCL ("´Hard Core Library"), mille versioon 2.0 sai juba nimeks NSS ning tehti kättesaadavaks ka väljaspoole Netscape'i. Hiljem on NSS-teeki arendanud eelkõige Mozilla kogukond ning see on kasutuses muuhulgas veebilehitsejas Mozilla Firefox, meilikliendis Mozilla Thunderbird ning mitmetes dokumentide signatuure töötlevates avatud lähtekoodiga rakendustes. Kuigi NSS näib olevat kõrge kvaliteediga, pole ta laiemalt siiski väga populaarne ning ka NSSi pigem tagasihoidlikus tempos kulgevat arendust veavad eest suuresti ainult Mozillaga seotud arendajad.

6.1.5 Botan

Botan on Jack Lloydi loodud TLSi teostus, mille esimene avalik väljalase oli aastal 2001. Kuigi tegu pole väga populaarse teegiga, on Botan siiski saadaval enimkasutatud Linuxi distributsioonides. Botan on olnud BSI huviorbiidis vähemalt alates aastast 2007, mil FlexSecure GmbH abiga lisati teegile Card Verifiable Certificates (CVC) tugi esmalt InSiTo nimelise teegi näol[7]. Ka BSI projekti 197[8] raames, mille eesmärgiks oli kvaliteetse üldotstarbelise krüptoteegi loomine vastavalt BSI nõuetele, valiti 18 kandidaatteegi seest arenduse baasiks välja just Botan[9], mis viidi hiljemalt 2017. aasta esimeses kvartalis vastavusse BSI nõuetega. Projekti kolmas faas nägi ette ka Botani teegi hooldamist ja ajakohasena hoidmist mitmete aastate jooksul, järgides uusi suundumusi krüptograafias, standardites ja rakendusstsenaariumites. BSI projekti 197 alltöövõtja Rohde & Schwarz on jätkanud Botan teegi arendustöid ka uurimisprojekti KBLS[10] raames. Hoolimata BSI ja selle partnerite abist on Botani teegi arendus jätkuvalt suures osas Jack Lloydi õlgadel. Üheski stabiilses Botani teegi versioonis pole veel TLS 1.3 tuge, kuid see on lisamisel. Positiivseks võib pidada Botani teegi üleminekut uuematele C++ keele standarditele ning vastavate turvalise C++ programmeerimise praktikate järkjärgulist kasutuselevõttu.

[6] https://developer.mozilla.org/en-US/docs/Mozilla/Projects/NSS
[7] https://en.wikipedia.org/w/index.php?title=Botan_(programming_library)&oldid=1055456913
[8] https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Informationen-und-Empfehlungen/Kryptografie/Kryptobibliothek-Botan/kryptobibliothek-botan_node.html
[10] https://www.rohde-schwarz.com/de/loesungen/cybersicherheit/about-us/research/kbls_254244.html

6.1.6 Bouncy Castle

Bouncy Castle[11] on ühine nimi kahele vabavaralisele krüptoteegile vastavalt Javale ja C#-le, mis teostavad paljusid krüptoprimitiive, vorminguid ja protokolle. Kuigi funktsionaalsuse osas on mõlema Bouncy Castle'i teegi katvus võrreldav OpenSSL-teegiga, jääb Bouncy Castle populaarsuse poolest OpenSSL-teegile siiski tugevalt alla. Bouncy Castle'i C#-teek ei hõlma siiski päris kogu Java-teegi funktsioonistikku ning ka selle arendusaktiivsus on madalam. Mõlemas keeles või õigemini mõlemal platvormil (JVM ja .NET) on tegu *de facto* peamiste krüptoteekidega, kui vastavatesse platvormidesse sisse ehitatud funktsioonid välja arvata. JVM-platvormil töötab Bouncy Castle eelkõige JCA ja JSSE pistikteegina (vt ptk 6.3.2).

6.1.7 Tink

Google'i arendatav Tink[12] on kõige uuem projekt siin nimekirjas (versioon 1.0.0 avaldati septembris 2017), mis pakub programmeerimise headest tavadest lähtuvalt võimalikult turvaliseks kasutuseks abstraheeritud liideseid põhiliste krüptograafiliste operatsioonide (AEAD, MAC, PRF, digitaalsignatuurid, hübriidkrüptimine) ja võtmehalduse teostamiseks. Tinki käsitlusalasse ei kuulu madalama taseme primitiivid ega ka kõrgema taseme protokollid ja vormingud. Täpsemalt hõlmab Tink mitut erinevat teeki eri programmeerimiskeelte jaoks (Java, C++, Objective-C, Go, Python), mis vahetarkvarana kasutavad sisemiselt vastava platvormi krüptograafilisi primitiive OpenSSLi, BoringSSLi, BouncyCastle'i, Conscrypti, Oracle JDK, OpenJDK, Go crypto ja Web Crypto API teekidest või teostustest. Sedasi võivad viimaste nõrkused mõjutada ka Tinki teekide turvalisust. Tinki teekide suurimaks tugevuseks võib lugeda programmeerimisliideseid, mida on keeruline ebaturvaliselt kasutada ning mille kasutust on lihtne auditeerida.

6.2 Valikukriteeriumid

See alapeatükk annab ülevaate põhilistest teguritest, mida tuleks krüptoteegi valikul silmas pidada, võttes seejuures ühtlasi arvesse konkreetse kasutusmalli nõudeid ning pidevalt arenevaid krüptostandardeid.

Valikuprotsessi lihtsustamiseks soovitame kasutada abivahendina DeepWikit[13]. Arendajad saavad esitada sellele platvormile LLM-liidese kaudu päringuid, et hinnata konkreetseid GitHubi koodihoidlaid siin käsitletavate valikukriteeriumide vaatenurgast. DeepWiki tööriist annab kontekstuaalselt täpseid vastuseid esitatud küsimustele koos vahetute viidetega lähtekoodile, mis võimaldab süstemaatilist ja läbipaistvat lähenemist teegi valikule.

RSA-PSSi tugi. Üks põhilisi valikukriteeriume on RSA-PSSi (*Probabilistic Signature Scheme*) algoritmi toe olemasolu. RSA-PSS on hetkel levinuim RSA-põhiste digitaalsignatuuride standard. Enamik tabelis 6 loetletud teeke toetab küll RSA-PSSi, kuid enne lõpliku valiku tegemist tuleks selle toe olemasolu siiski kõigi kaalumisel olevate teekide puhul üle kontrollida.

Postkvant-krüptograafia tugi. 2026. aasta alguseks on kvantarvutustehnika järjepidev areng ning järjest uute postkvant-krüptograafiale (PQC) ülemineku strateegiate väljatöötamine teinud postkvant-krüptograafiast küberturbe valdkonna ühe kesksema uurimisobjekti. PQC-võimaluste

[11] https://www.bouncycastle.org/
[12] https://github.com/google/tink
[13] https://deepwiki.com

juurutamine või plaanitav juurutamine on selge märk teegi haldajate reageerimiskiirusest valdkondlikele suundumustele ning selle elujõulisusest pikemas perspektiivis. Krüptoteegid, millel täielikult puuduvad kavad PQC ülevõtmiseks, riskivad lähemas tulevikus kasutajate kaotamisega.

Tabel 6 sisaldab tulpa „PQC tugi“, mis kajastab PQC teostuste hetkeseisu käsitletud teekides. Ehkki hetkel PQC toeta teegid peaksid ootuste kohaselt selle sisse tooma oma tulevastes versioonides, tasub valikuprotsessi raames sõltumatult hinnata kõigi teegikandidaatide PQC-tegevuskavasid.

Siin loetletud teekidest paistavad eelkõige OpenSSL, WolfSSL ja Botan silma juba varakult küpsete ja funktsionaalselt täielike PQC teostustega.

Platvormid. Ehkki enamik tabelis 6 loetletud krüptoteeke toetavad üldotstarbelisi platvorme ja operatsioonisüsteeme, on mõned teegid kohandatud spetsiifiliselt piiratud ressurssidega keskkondade jaoks, nagu mikrokontrollerid ja sardsüsteemid. Niisuguste platvormide jaoks on optimeeritud nt Mbed TLS ja WolfTPM (WolfSSLi variant).

Üldtasandi ja detailtasandi primitiivid. Krüptoteegid võivad lähtuda erinevatest arhitektuurilistest põhimõtetest, mis panevad paika nende algoritmilise abstraktsuse astme. Osad teegid sisaldavad algoritmist sõltumatuid kõrgtaseme rakendusliideseid (API); selle eesmärk on suurendada teegi kasutajasõbralikkust ning vähendada konfiguratsioonivigade tõenäosust. Niisugused krüptoteegid sisaldavad sageli täiendavaid funktsioone, nagu automaatne võtmerotatsioon, objektide kodeerimine/dekodeerimine ning algoritmide sujuv migreerimine (oluline süsteemide kindlustamiseks tulevikus väljakujunevate ohtude vastu, näiteks rakendades tulevase kvantohu leevendamiseks postkvant-krüptograafiat). Nende tuntumad näited on Tink ja libsodium.

Teised teegid sisaldavad seevastu madaltaseme APIsid, mis võimaldavad vahetut juurdepääsu krüptoprimitiividele, nagu ECDSA- või RSA-operatsioonidele. Sellised teegid sobivad paremini krüptograafiliste mehhanismide täpset ohjet nõudvateks rakendusteks. Nende näidete hulka kuuluvad Libgcrypt, Botan, Crypto++ ja Nettle.

Ühe või teise paradigma valik oleneb sageli ühilduvus- ja koostalitlusnõuetest. Värskelt loodud isoleeritud süsteemides, kus kõige olulisem tegur on konsistentsus, eelistatakse reeglina algoritmist sõltumatut lähenemist. Seevastu olemasolevate spetsiifilistele algoritmidele tuginevate kolmanda poole loodud komponentide lõimimisel süsteemi võimaldavad madaltaseme primitiiviteegid suuremat paindlikkust ja vähendavad lõimimise keerukust.

Protokollide teostused. Krüptograafiliste tuumalgoritmide kõrval sisaldavad paljud teegid ka krüptograafiliste protokollide, eriti turvalise andmesidega seotud protokollide teostusi. Toetatavate protokollide hulgas on üldjuhul TLS/SSL, DTLS, QUIC ja avaliku võtme infrastruktuuriga seotud standardid nagu X.509 sertifikaatide töötlemine ja PKCS#1–12 ühilduvus. Mitmesuguste krüptoprotokollide ulatusliku toega paistavad silma näiteks OpenSSL, GnuTLS ja WolfSSL.

Märkigem, et algoritmist sõltumatud teegid (vt eespool) reeglina ei sisalda protokollide teostusi, keskendudes nende asemel krüptograafilistele baasoperatsioonidele.

C ja teiste programmeerimiskeelte omateegid. Paljud krüptoteegid on jõudluse, ühilduvuse ja ajaloolise relevantsuse huvides teostatud C-s. Samas on järjest kasvamas teistes kõrgkeeltes nagu Java, Python ja C# kirjutatud omateekide populaarsus, mille eeliseks on nende

integreerimise hõlpsus ning tarkvaraarenduse töövoogude lihtsustamine.

Omateegid võimaldavad vältida C-põhiste sõltuvuste integreerimise raskuskohti ning on seejuures sageli täiesti piisava jõudlusega. Üksikasjalikumalt käsitleb selliseid teeke alapeatükk 6.3. Tarkvara arendamisel teistes keeltes tuleks eelistada keele omateeke, välja arvatud juhul kui C-põhise teegi vahetuks integreerimiseks on mõjuv põhjus.

Java ja C# omateek on näiteks Bouncycastle, Pythonil pyca/cryptography.

C-seotised/ümbrised teiste programmeerimiskeelte jaoks. Omateekide alternatiiviks on kasutada C-seotisi või ümbriseid, mis võimaldavad juurdepääsu C-põhistele krüptoteekidele teistest programmeerimiskeeltest. Mõne teegi tarned sisaldavad ametlikke seotisi, teised toetuvad ühisarendatavatele lahendustele. Niisugused seotised on sageli piiratud võimalustega ning ei hõlma kogu teegi funktsioonistikku. Keeruliseks võib osutuda ka alusteegi ja selle seotiste sünkroniseerimine, mille tagajärjeks võib olla ümbriste iganenud või hooldamata versioonide kasutamine.

C-seotiste kasutamist tasub seetõttu kaaluda vaid juhul, kui garanteeritud on nende pikaajaline stabiilsus ja minimaalne kohandamisvajadus. Enamikul juhtudel on omateegid neist stabiilsem ja kestlikum lahendus.

Levinumate C-seotiste hulka kuuluvad OpenSSLi seotised, mida on olemas mitmete programmeerimiskeelte jaoks; osad neist on küll kohandatud konkreetsete kasutusmallide jaoks ning ei pruugi olla enam aktiivselt hallatavad.

Dokumentatsiooni loetavus ja kasutatavus. Selgelt kirjutatud ja kergesti kättesaadav dokumentatsioon on krüptograafia tõhusate ja turvaliste teostuste jaoks hädavajalik. Kehva dokumentatsiooni tagajärgedeks võivad olla konfiguratsioonivead, teegi väärkasutused ja turvanõrkused. OpenSSLil võib küll olla isegi hoolimata selle ulatuslikust kasutusest ja küpsusest halvasti navigeeritavaga dokumentatsiooniga teegi maine, kuid arendajate abistamiseks on saadaval hulgaliselt välisressursse ja kasutusõpetusi.

Teised teegid, nagu Tink, on samal ajal sihipäraselt kavandatud lähtuvalt kasutatavuse ja loetavuse põhimõttest, võimaldades arendajatele algusest peale juurdepääsu intuitiivsetele ja minimalistlikele APIdele ning põhjalikule juhendmaterjalile.

Soovitame kindlasti hinnata iga teegikandidaadi dokumentatsiooni ja kättesaadavate õppematerjalide kvaliteeti. Nõu saamiseks tasub pöörduda ka väljavalitud teegiga töötanud kogenud arendajate poole.

Kõrgkindlad teostused. Paralleelselt liikumisega postkvant-valmiduse suunas on hakatud välja töötama kõrgkindlaid krüptograafia teostusi, seda eelkõige akadeemiliste algatuste toel. Niisuguste teekide eesmärk on krüptograafiliselt ja formaalselt verifitseeritud ning põhjalikult testitud koodibaasi tarnimine, mis on mõeldud rakendusteks, kus kesksel kohal on korrektsus ja turvalisus.

Ehkki enamik loetletud teeke on endiselt väljalaskmise-eelses arendusfaasis, on see lähenemissuund tulevaste kõrgturvaliste teostuste vaatenurgast siiski paljutõotav. Olulisemad näited sellistest teekidest on libcrux ja libjade.

6.3 Arenduskeskkonnad

6.3.1 C ja C++

Programmeerimiskeeltesse C ja C++ ega nende standardteekidesse ei ole sisse ehitatud krüptograafilisi vahendeid, mistõttu tuleb neis keeltes kasutada väliseid krüptograafiat pakkuvaid teeke. Võrreldes teiste arenduskeskkondadega leidub selliseid samas programmeerimiskeeles arendatud teeke (omateeke) suhteliselt palju, ning mitmeid nende hulgast kirjeldab ja soovitab ka see aruanne peatükis 6.1.

6.3.2 Java

Krüptograafiliste meetodite kasutamine Java (JVM) platvormil toetub *Java Cryptography Architecture* (JCA) standardile, mis on Java platvormi normatiivne osa. JCA sätestab krüptoprimitiivide kasutamise liidesed ning sisaldab ka osa liideste teostusi.

Kuni Java versioonini 1.3 olid krüptimise ja võtmekehtestusega tegelevad liidesed defineeritud eraldi Java Cryptography Extensioni (JCE) nime all, kuid hilisemates versioonides seda lahusust enam ei ole. JCE nimi on endiselt kasutuses, kuid tihtipeale lihtsalt JCA sünonüümina.

JCA defineerib oluliselt suurema hulga primitiive kui ta ise teostab ning seetõttu on tihtipeale vaja kasutada kolmanda poole pistikteeke (*provider*), näiteks Bouncy Castle (vt ptk 6.1.6). Iga JCAs versioonis defineeritud primitiivide loetelu on võimalik leida JCA vastava versiooni dokumentatsioonist („Java Cryptography Architecture Standard Algorithm Name Documentation“, „Java Security Standard Algorithm Names).

JCA juurde käib eraldi arhitektuurse moodulina Java Secure Socket Extension (JSSE), mis sätestab SSLi, TLSi ja DTLSi kasutamise liidesed Java platvormil. JSSE projekteerimisprintsiibid on samad kui JCAl.

JCA ning JSSE dokumentatsioon on põhiline alguspunkt arendajale, kellel on vaja kasutada Java (JVM) rakendustes krüptograafilisi primitiive või turvalisi sideprotokolle.

Tuleb hoolitseda, et nii Java käidukeskkond (*runtime environment*) kui pistikteegid oleks alati uuendatud ja turvanõrkused eemaldatud.

6.3.3 Go

Go standardteek sisaldab paketti `crypto` koos selle alampakettidega, milles on realiseeritud väike valik krüptoprimitiive, X.509 sertifikaatide objektimine ja jadastus ning osaliselt ka TLS 1.2 ja TLS 1.3 protokollid. Lisaks on täiendav komplekt primitiive ja protokolle realiseeritud eraldi moodulis, mis asub nimeruumis `golang.org/x/crypto`. Neil põhineb ka Google'i Go jaoks arendatud krüptoteek Tink.

Go jaoks leidub mitmeid mähiseid ka OpenSSL-teegile, ent kahjuks katavad need vaid osi OpenSSLi funktsioonistikust ning aruande kirjutamise ajal ei saanud neist veel ühtegi pidada väga elujõuliseks. Samas on C-teekide liidestamine Go-programmidega suhteliselt lihtne (see on Go-keele osa), mistõttu on võimalik kasutada ka vahetult OpenSSLi või mõnda teist C-teeki. Muudel juhtudel soovitame jääda kasutama standardteegi paketti `crypto`, moodulit `golang.org/x/crypto` või teeki Tink.

6.3.4 Node.js

Node.js standardteek sisaldab mooduleid `crypto` ja `tls`, mis mähivad osa Node.js käivituskeskkonnaga vaikimisi paigaldatava OpenSSL-teegi krüptoprimitiividest ja TLSi teostusest. Mitmete algoritmide ja protokollide jaoks, mida standardteek ei paku, leidub pakihaldussüsteemi *npm* pakettidena levitatavaid mooduleid, ent nende kvaliteet võib olla kõikuv. Kuna JavaScript pole väga riistvaralähedane käituskeskkond, võib selles realiseeritud krüptograafilistel algoritmidel olla probleeme kõrvalkanalikindlusega.

Node.js pakub ka Node-API (varem N-API) nimelist laiendusliidest JavaScripti moodulite ja muude funktsioonide lisamiseks kasutades C++ keelt, mille abil on võimalik liidestada ka C ja C++ teeke, ent selle kasutamine on pigem keeruline. Kuivõrd OpenSSL on juba vaikimisi osaliselt liidestatud, ei ole selle järele krüptograafia kontekstis ka suurt vajadust.

6.3.5 Rust

Rusti standardteek ei sisalda krüptograafilisi algoritme ning nende teostamine on jäetud välistele teekidele, mis on üldiselt kättesaadavad ka pakihaldussüsteemi *Cargo* kaudu. Seetõttu on väliseid krüptoalgoritmide teostusi palju[14,15] ning nende kvaliteet ei ole ühtlane [197].

Eraldi võib ära märkida siiski OpenSSLi mähise[16] ning Rustlsi[17] nimelise TLS 1.2 ja TLS 1.3 teostuse. Jätkuvalt on arendusjärgus ka värske Tinki teegi (vt ptk 6.1.7) mitteametlik Rusti versioon[18], millega tehti algust septembris 2020.

Programmeerimiskeel Rust, mille projekteerimiseesmärkide hulga kuuluvad rangem tüübi- ja mäluturvalisus, võimaldab küll välistada mõned vigade klassid, ent pole siiski imevahend kõikide programmeerimisvigade vastu. Seetõttu ei tohi ka Rust-keeles kirjutatud koodilt alati automaatselt paremat kvaliteeti oodata. Ometi tasub hoolikalt kaaluda Rusti kasutamist komponentides, mille funktsionaalsusest olulise osa moodustab krüptograafiliste algoritmide kasutamine.

6.3.6 PHP

PHP pakub rakendustele mõningaid krüptograafilisi laiendusi[19], millest kõige märkimisväärsem on OpenSSLi mähis, mis võimaldab kasutada mõningaid OpenSSLi funktsioone. Juhul, kui PHP-rakenduses on vaja kasutada krüptograafilisi funktsioone, soovitame eelkõige pöörduda vastavate mähise poole.

Leidub ka mitmeid mittestandardseid PHP-krüptoteeke, ent nende kvaliteet on varieeruv. Neist üks populaarseim on `phpseclib`[20], mis pakub mõningaid sümmeetrilisi ja asümmeetrilisi krüptoalgoritme, X.509 ja SSH tuge. Kuna PHP pole väga riistvaralähedane arendusplatvorm, võib selles realiseeritud krüptograafilistel algoritmidel olla probleeme kõrvalkanalikindlusega.

[14] https://cryptography.rs/
[15] https://blog.logrocket.com/rust-cryptography-libraries-a-comprehensive-list/
[16] https://crates.io/crates/openssl
[17] https://github.com/rustls/rustls
[18] https://github.com/project-oak/tink-rust/
[19] https://www.php.net/manual/en/refs.crypto.php
[20] https://phpseclib.com/

6.3.7 .NET

.NET-platvormil (mis praktikas tähendab enamaltjaolt C#-keelt) on olemas süsteemne nimeruum `System.Security.Cryptography`, milles sisalduvad põhiliste krüptograafiliste primitiivide teostused. Eksisteerivad ka süsteemsed vahendid SSL/TLS- ja HTTPS-ühenduste loomiseks, näiteks kasutades `System.Net.Http.HttpClient`, `System.Net.HttpWebRequest`, `System.Net.Security.SslStream` jt klasse. Kui vaja on midagi enamat, tasub ennekõike uurida, kas selle toetus on juba olemas vabavaralises BouncyCastle'i teegis (vt ptk 6.1.6).

6.3.8 Python

Python-keele standardteek sisaldab mitmeid krüptograafilisi funktsioone pakkuvaid mooduleid.

- Moodul `hashlib` pakub räsifunktsioone ning `PBKDF2` ja `scrypt` võtmetuletusfunktsioone (ptk 2.4).
- Moodul `hmac` pakub räsipõhiste sõnumiautentimiskoodide [38] funktsioone.
- Moodul `secrets` pakub vahendeid krüptograafiliselt tugevate juhuarvude genereerimiseks.
- Moodul `ssl` pakub TLS/SSL-funktsioonidega sokleid.

Kuna Pythoni standardteegi pakutavad vahendid on piiratud, on alates aastast 2001 arendatud ka eraldiseisvat `pyOpenSSL`[21] teeki, mis on OpenSSLi mähis. Samal põhjusel lõi grupp Pythoni arendajaid 2013. aastal eraldiseisva `cryptography`[22] teegi, mis pakub mitmete krüptoprimitiivide teostusi, ning koondus organisatsiooniks Python Cryptographic Authority (PyCA)[23], mille alla on aruande kirjutamise ajaks liikunud ka `pyOpenSSL`-teegi arendus. Kui Pythoni standardteegi pakutavatest krüptograafilistest vahenditest ei piisa, soovitame kasutada eelkõige just `cryptography`-teeki, eelistades seda ka `pyOpenSSL`-teegile. Viimast tuleks selle arendajate tugeval soovitusel kasutada ainult TLS-ühenduste jaoks.

[21] https://pyopenssl.org/
[22] https://cryptography.io/
[23] https://github.com/pyca/

7 Postkvant-krüptograafia kasutuselevõtt

Kvantmehhaanika postulaadid lubavad arvutussamme, mis oluliselt erinevad sammudest, mida teevad praegu kasutusel olevad arvutid. Neid samme kombineerides on võimalik luua efektiivseid algoritme, mis lahendavad teatud arvutusülesandeid, mille eeldataval keerukusel põhinevad enimkasutatavad avaliku võtme krüptosüsteemid (RSA, ElGamal, elliptkõverapõhised süsteemid). Selliste kvantalgoritmide jooksutamiseks ja nende krüptosüsteemide murdmiseks on vaja piisavalt suuri ja töökindlaid kvantarvuteid. Arendustegevus selliste arvutite loomiseks on toimumas ja 2026. aastal on jõutud märkimisväärsete miilikivideni [198], mis viitavad sellele, et need eeldatavalt keerulised arvutusülesanded võivad varsti lahendatavad olla[1].

Selle aruande kirjutamise ajal on üle kogu maailma käimas või käivitumas kasutuselolevate krüptoalgoritmide väljavahetamine kvantturvaliste algoritmide vastu. Väljavahetamine nõuab kvantturvaliste algoritmide toe lisamist kõikvõimalike krüptograafiat kasutavatele tark- ja riistvarakomponentidele. Selles peatükis püüame lühidalt kirjeldada, milline on aruande kirjutamise aegne tugi kvantturvalistele krüptoalgoritmidele komponentides, mille vastu mõni aruande lugejatest huvi võiks tunda. Komponentide valiku aluseks on olnud aruande lähteülesande tehniline kirjeldus. Samuti toetab valikut meie oma kogemus ühe või teise komponendi olulisuse kohta Eesti infosüsteemides. Sellise valiku tõttu puudutab peatükk rohkem protokolle, tark- ja riistvarakomponente, protsesse, spetsifikatsioone ja standardeid kui kogu ülejäänud aruanne, kuid fookus on kitsalt postkvant-krüptograafia toel. Siin peatükis ei püüa me aga anda ülevaadet kvantarvutitest ning nendega või kvantturvalistele algoritmidele üleminekuga seotud riskidest ja ohtudest. Neid kirjeldame teekaardil [18], mille lõppversioon valmis kahjuks vahetult enne eelmises lõigus nimetatud kvantarvutuse miilikivideni jõudmise avalikustamist.

Märgime veelkord, et selle peatüki sisu peegeldab kvantturvaliste algoritmide tuge aruande kirjutamise ajal, st 2026. aastal. Siin olev informatsioon võib küllaltki kiiresti vananeda. Eelkõige tuleb siinkohal mainida viitamist RFC-kavanditele, mis võivad mingiks hetkeks saada kas IETFi standarditeks või hoopiski aeguda. Kuna üleminek postkvant-krüptograafiale on aruande valmimise ajal alles täit hoogu sisse saamas, siis on viidete eesmärk pakkuda lugejale võimalikult head ülevaadet ülemineku hetkeseisust ning võimalikest arengusuundadest, kuid kavandites esitatud sisuga ei tohiks arvestada tootmistasemel arenduses enne, kui need on avaldatud standarditena (RFC).

7.1 Tugi turvaprotokollides

7.1.1 TLS

TLSi (transpordikihi turve) kätlus kasutab mitmeid krüptograafilisi komponente. Mitmed neist kasutavad sümmeetrilist krüptograafiat ja nende kvantturvaliseks muutmiseks piisab vaid võtmepikkuse suurendamisest, kuid mõned asümmeetrilised primitiivid, mida kätlus kasutab, vajavad väljavahetamist.

Kätluse jooksul peavad server ja kasutaja leppima omavahel kokku ühises võtmes, mille jaoks peavad nad kasutama mõnda võtmevahetus- või võtmekapseldusmehhanismi. Samuti peab

[1] https://ecdsa.fail

klient ennast kätluse käigus serverile autentima, mille jaoks allkirjastab ta oma salajase võtmega kätluse senise protokolli [70, 72].

Üks levinumaid meetodeid kvantturvalise võtmevahetuse jaoks on kasutada ML-KEMi ja mõnda elliptkõverapõhist võtmevahetusskeemi (nagu X25519) hübriidselt [199, 200]. Sellist lähenemist kasutavad näiteks AWS[2], CloudFlare[3], Meta[4] ja Chrome[5]. Teostusi on võimalik leida OpenSSLilt[6], WolfSSLilt[7] ja go/crypto teegist[8].

Autentimisprotsessi kvantturvaliseks tegemine on saanud vähem tähelepanu, sest selle puhul pole ohtu *harvest-now-decrypt-later* rünneteks: isegi kui kvantründega hiljem salajane võti leida, saab selle abil võltsida vaid uusi allkirju.

Sellegipoolest leidub lahendusi, kus ML-DSA kombineeritakse hübriidselt mõne klassikalise signatuuriskeemiga [201] või kus autentimine seotakse ühise võtme kokkuleppimisel kasutatava võtmekehtestusmehhanismiga [202]. Samuti on ka soovitusi ML-DSA või SLH-DSA kasutamiseks mittehübriidselt [203, 204].

Kvantturvalise TLSi kohta leiduvad alljärgnevad RFC-d ja nende kavandid:

- „Hybrid Key Exchange in TLS 1.3“ [199]: kirjeldab üldisemalt, kuidas teostada TLSis hübriidset võtmekehtestust;
- „Post-Quantum Traditional (PQ/T) Hybrid Key Agreement Mechanisms for TLS 1.3“ [200]: kirjeldab kolme hübriidvõtmekehtestusmeetodit, mis ühendavad ML-KEM- erinevate elliptkõveratel põhinevate võtmevahetusskeemidega;
- celi-wiggers-tls-authkem [202] (2021): kirjeldab, kuidas viia läbi autentimist võtmekehtestuse ajal;
- reddy-tls-composite-mldsa [201] (2024): kirjeldab, kuidas ühendada ML-DSAd hübriidselt erinevate klassikaliste signatuuriskeemidega;
- ietf-tls-mldsa [203] (2024): kirjeldab, kuidas kasutada TLSis autentimiseks ML-DSAd;
- reddy-tls-slhdsa [204] (2024): kirjeldab, kuidas kasutada TLSis autentimiseks SLH-DSAd.

7.1.2 SSH

SSH (Secure Shell Protocol) on võrguprotokoll, mille peamine eesmärk on võimaldada kaugsisselogimist ja teenuste kasutamist turvamata võrgu kaudu. SSH kasutab avaliku võtmega krüptograafiat, et luua ühendus serveri ja kasutaja vahel ning et lasta serveril ja kasutajal end teineteisele autentida.

Nagu TLSi puhulgi (ja samadel põhjustel), on SSH postkvant-krüptograafiale üleminekul keskendutud eelkõige võtmevahetusprotokollile, mitte autentimisele. OpenSSH[9] on võtnud võtmevahetuseks kastutusele kaks meetodit: `sntrup761x25519-sha512` ja `mlkem768x25519-sha256`. Neist esimese puhul on hübriidselt ühendatud Streamlined NTRU Prime ja X25519, mille kasuta-

[2] https://aws.amazon.com/about-aws/whats-new/2025/11/network-load-balancers-post-quantum-key-exchange-tls/
[3] https://pq.cloudflareresearch.com/
[4] https://engineering.fb.com/2024/05/22/security/post-quantum-readiness-tls-pqr-meta/
[5] https://security.googleblog.com/2024/09/a-new-path-for-kyber-on-web.html
[6] https://openssl-foundation.org/post/2025-04-22-pqc/
[7] https://www.wolfssl.com/products/wolfcrypt-post-quantum/
[8] https://pkg.go.dev/crypto/tls
[9] https://www.openssh.org/pq.html

mist võimaldab ka GitHub[10]. Teine OpenSSH lisatud režiim `mlkem768x25519-sha256` ühendab hübriidselt ML-KEMi ja X25519, mis on nüüd vaikimisi OpenSSH poolt kasutatav protokoll. Alates OpenSSH versioonist 10.1 esitatakse kasutajale ka hoiatus, kui ta kasutab mittehübriidselt mõnda klassikalist võtmevahetusprotokolli.

Olulisemad RFCd ja RFC-kavandid kvantturvalise SSH kohta on:

- RFC 9941 [205] (2023): kirjeldab, kuidas ühendada hübriidselt Streamlined NTRU Prime ja X25519, et saavutada SSH jaoks kvantturvaline võtmevahetus;
- ietf-sshm-mlkem-hybrid-kex [206] (2020): kirjeldab, kuidas ühendada hübriidselt ML-KEM erinevate elliptkõveratel põhinevate võtmevahetusmehhanismidega, et saavutada kvantturvaline võtmevahetus.

7.1.3 IPsec

IPsec on protokollistik, mille abil korraldatakse turvalist suhtlust IP-võrgustikus. Eelkõige kasutatakse seda VPNides. Postkvant-krüptograafiale ülemineku vaatepunktist on olulisim IKEv2 protokoll, mille käigus toimub poolte autentimine ja krüptitud suhtluskanali loomine.

Jällegi on enamik tähelepanust läinud võtmekapseldusmehhanismi teostamise peale ning autentimine on saanud vähem tähelepanu. Näib, et laialt levinud teostusi kvantturvalisele IPsecile veel olemas ei ole, kuid väiksemad teenusepakkujad on asunud realiseerima kvantturvalisi võtmekapseldusmehhanisme[11] (nii hübriidselt kui ka puhtal kujul) ja mõnel juhul ka autentimismeetodeid[12].

Leidub mitmeid RFCsid, mis standardivad IKEv2 jaoks erinevaid võtmekehtestusmehhanisme ja signatuuriskeeme. Seetõttu on IKEv2 võrreldes teiste protokollidega lihtsamini kvantturvaliseks muudetav, sest üksikute alamprotokollide väljavahetamine on hõlpsam. Teisest küljest, kui arvestada, et IPseci alamprotokollid peavad kõik kooskõlaliselt töötama, siis peab üleminekul olema hoopis palju ettevaatlikum.

Tähtsaimad asjassepuutuvad RFCd ja RFC-kavandid on:

- RFC 7427 [207]: kirjeldab, kuidas kasutada ükskõik millist signatuuriskeemi IKEv2 autentimisprotseduuris (sh postkvant-algoritme);
- RFC 9370 [208]: kirjeldab, kuidas rakendada IKEv2s hübriidset võtmevahetust. Samuti kirjeldab, kuidas kasutada IKEv2 autentimisprotseduuris ükskõik millise avaliku võtmega krüptosüsteemi või signeerimisalgoritmi võtit;
- draft-ietf-ipsecme-ikev2-mlkem [209] (2023): kirjeldab, kuidas teostada IKEv2s hübriidset võtmevahetust, kasutades ML-KEMi;
- draft-ietf-ipsecme-ikev2-pqc-auth [210] (2024): kirjeldab, kuidas kasutada postkvant-algoritme IKEv2 autentimisprotseduuris.

[10] https://github.blog/engineering/platform-security/post-quantum-security-for-ssh-access-on-github/
[11] https://docs.fortinet.com/document/fortigate/7.6.1/administration-guide/229631
[12] https://www.rambus.com/blogs/bringing-ipsec-into-the-quantum-safe-era/

7.1.4 WPA3

Wi-Fi allianss ei ole avalikult üles näidanud huvi WPA3 kvantturvaliseks muutmise vastu, kuid arvatakse, et WPA4 võib juba toetada postkvant-algoritme[13].

7.1.5 Kerberos

Kerberos [211] tugineb põhiosas ainult sümmeetrilisele krüptograafiale, mis tähendab, et kvantarvutid seda otseselt ei ohusta. Kerberose laiendus võimaldab kasutajaid esmakordselt autentida X.509 sertifikaatide abil [212]; meil puudub info selle laienduse postkvant-krüptograafiale ülemineku kohta.

7.1.6 WireGuard

WireGuard on VPNi lahendus, mis kasutab kätluse käigus asümmeetrilist krüptograafiat, mis tuleks kvantturvaliseks muuta.

Üks olulisemaid väljapakutud lahendusi ilmus aastal 2021 [213], kus kirjeldati meetodit, kuidas ühendada kätluses omavahel kvantturvaline autentimine ja võtmevahetus. Artiklis soovitati kasutada McEliece'i ja Saberit, kuid tänaseks on need algoritmid saanud vähem tähelepanu kui mitmed teised väljapakutud kandidaadid.

WireGuardis on võimalik kasutada ka meetodit, kus võtmevahetuse ajal kasutavad mõlemad pooled eelnevalt kokkulepitud sümmeetrilist võtit, et lisada sellesse kvantturvalise sümmeetrilise krüptograafia element[14]. See lükkab reaalsuses muidugi sama probleemi edasi, sest kuidagi peab kokku leppima ka tolles sümmeetrilises võtmes. Selle probleemi on lahendanud näiteks Rosenpass,[15] realiseerides kvantturvalise võtmevahetuse, mis ühildub WireGuardiga, kasutades eelmainitud eelnevalt kokkulepitud võtmete meetodit. Praegu tundub Rosenpass olevat üks tõsiseltvõetavamaid kvantturvalise WireGuardi teostusi.

7.1.7 OpenVPN

OpenVPNi puhul on võimalik kasutada kätluseks OpenSSL 3.5, milles on realiseeritud ML-KEM, ML-DSA ja SLH-DSA[16]. Seetõttu on võimalik soovi korral kasutada OpenVPNi juba täna kvantturvalisena, kuid see vajab eraldi seadistamist. See tähendab, et OpenVPN kasutab küll vaikimisi klassikalist krüptograafiat ning *harvest-now-decrypt-later* ründed on võimalikud, kuid teadlikul kasutajal on võimalik neid vältida. Open Quantum Safe projekti raames on implementeeritud ka eksperimentaalne kvantturvaline OpenVPN[17].

7.1.8 OAuth ja OpenID Connect

OpenID Connect (ja laiemalt OAuth 2.0) kasutab avaliku võtmega krüptograafiat juurdepääsulubade allkirjastamisel ning toetub ka TLSile. Erinevalt teiste protokollide autentimisprotseduuridest kehtivad OpenID Connecti juurdepääsuload pikka aega ning seetõttu on olemas oht *harvest-now-decrypt-later* rünneteks. See tähendab, et ei piisa vaid TLSi postkvant-krüptograafiale ülemi-

[13] https://www.cwnp.com/wi-fi-ready-quantum-threat/
[14] https://www.wireguard.com/known-limitations/
[15] https://rosenpass.eu/about/
[16] https://blog.openvpn.net/quantum-safe-vpn
[17] https://github.com/open-quantum-safe/oqs-demos/tree/main/openvpn

nekust ja rünnete vältimiseks peab juba praegu tegutsema.

Aastal 2024 peeti OpenID sihtasutuse töötoa raames paneelarutelu postkvant-krüptograafiale ülemineku osas[18] ning jõuti järeldusele, et standardite muutmisega peab kohe tegelema hakkama. On olemas küll PoC-teostusi, nagu OpenID PoC [214], kuid selle aruande ilmumise ajaks ei ole veel midagi OpenID sihtasutuselt endalt ilmunud.

7.1.9 WebAuthn

Kuigi WebAuthn-protokolli haldab W3C, ei keskendu nad otseselt postkvant-krüptograafiale üleminekule. Küll aga on FIDO allianss, mis haldab koos W3Cga laiemat FIDO2 autentimisstandardit (mis tugineb suuresti WebAuthnil), väljendanud soovi lähimal ajal üleminekuga tegeleda [215].

Tegelikult võib öelda, et WebAuthnit on võimalik juba kasutada mõningal määral kvantturvalisena, kuna see kasutab CBOR Object Signing and Encryption (COSE) andmevormingut (ptk 4.4.6), mis lisas toe kolmele versioonile ML-DSAst, tuginedes seejuures IETFi standardikavandile RFC 9964 [216]. Laiem infrastruktuur veel siiski postkvant-algoritme ei toeta.

Kvantturvalist WebAuthni kirjeldab veel teinegi RFC-kavand [217] ning artikkel [218], mis kirjeldab laiemat, kvantturvalist FIDO2. Neile lisanduvad veel SandboxAQ PoC-teostus[19], mis toetab ka riistvaralisi autentimisvahendeid, ning FeitianTechi demokeskkond[20].

7.1.10 Web eID

Web eID[21] on autentimisprotokoll, mis võimaldab kasutada kiipkaarte autentimiseks ja digisignatuuride loomiseks veebibrauseris. Web eID raamistiku põhikomponentide (v.a kasutaja autentimisseadmed) ülesandeks on Web eID identsustõendi – mis koosneb pretensioonile antud digisignatuurist ning sertifikaadist – verifitseerimine. Ei ole alust pidada Web eIDd kvantkindlaks ning samuti puudub teave postkvant-krüptograafiliste funktsioonide väljatöötamise kohta Web eID raamistiku jaoks.

Siiski on praeguseks loodud Web eID eksperimentaalversioon[22], mis toetab Dilithium5 (ML-DSA) signatuuriga tookeneid ja sertifikaate.

7.2 Tugi taristuprotokollides

7.2.1 IEEE 802.1X

Kuigi IEEE 802.1 töörühm on postkvant-krüptograafiale üleminekut arutanud, siis tegelikke samme IEEE 802.1Xi üleminekuks ei ole astutud. On olemas RFC-kavandid [219, 220, 221], millest töörühm on teadlik, ning on artikleid, mis pakuvad välja alternatiivseid lahendusi IEEE 802.1X kasutamisele [222], kuid selgeid plaane ei tundu töörühmal veel üleminekuks olevat.

[18] https://openid.net/post-quantum-identity-standards/
[19] https://github.com/sandbox-quantum/pqc-fido2-impl
[20] https://github.com/FeitianTech/postquantum-webauthn-platform?tab=readme-ov-file
[21] https://web-eid.eu
[22] https://github.com/Muzosh/Post-Quantum-Authentication-On-The-Web

7.2.2 Bluetooth

Bluetooth-protokollide olulisemad komponendid, mis peavad olema postkvant-turvalised, on signatuuriskeem ja võtmekehtestusmehhanism. Oluline piirang on seadmete piiratud arvutusvõimsus, mis ei võimalda võtta kasutusele liiga mahukaid arvutusi nõudvaid krüptograafilisi algoritme.

Bluetoothi vaikimisi turvaline võtmekehtestusprotokoll (*secure key exchange*) põhineb elliptkõverate krüptograafial (Diffie-Hellmani võtmekehtestusel), mis ei ole postkvant-turvaline. Teadusartiklites [223, 224] võrreldakse olemasolevat võtmevahetusalgoritmi ECDH P-256 kvantturvaliste algoritmidega Kyber-512, Kyber-768 ja Kyber-1024 (nüüdseks standarditud kui ML-KEM, vt. alapeatükk 2.6.4.1), kus Kyber-512 peaks olema „klassikalises“ mõttes sama turvaline kui ECDH P-256. Vastava võtmevahetuse kontseptsiooni tõendus näitab, et Kyber-512 võtmevahetuse aeg on pikenenud mõnelt millisekundilt 10 sekundile, kuid praktikas on võimalik saavutada palju paremaid jõudlustulemusi (sõltuvalt seadme tüübist ja kaugusest). Samas peab arvestama oluliselt pikemate võtmete ja signatuuridega. Sobivateks signatuuriskeemideks on pakutud Falcon-512 ja Dilithium-2 (ML-DSA-44). Nüüdseks on Falconi skeemis avastatud külgkanaliründe id.

7.2.3 NFC ja RFID

Kuigi turvaline autentimine ei ole RFID/NFC jaoks üldiselt kohustuslik, võib see siiski olla mõne konkreetse rakenduse jaoks oluline. RFID-seadmete piirangute tõttu on raske kasutada isegi mitte-postkvantkrüptograafiat. Teadusartiklite hulgast leiab palju spetsiaalselt konstrueeritud protokolle, tihti ilma piisavalt formaalse turvatõestuseta. Need lahendused ei tundu piisavalt küpsed, et neist siinkohal kirjutada.

NFC foorumi allkirjakirje tüübimääratlus (*Signature Record Type Definition*, RTD) on turvaprotokoll, mida kasutatakse NDEF (NFC andmevahetuse formaat) sõnumite tervikluse ja autentsuse kaitsmiseks. Signeeritud sõnumite usaldus on seotud digitaalsete sertifikaatidega. Toetatud signatuuriskeemide hulgas on mainitud RSA, DSA, ja ECDSA variatsioone, kuid mitte kvantturvalisi signatuuriskeeme. Teadusartiklites [225] soovitatakse esemevõrkudes üldiselt kasutada Falconi või Dilithiumi signatuuriskeeme (räsipõhised skeemid nagu SPHINCS+ ei ole soovitatavad ülisuurte signatuuride tõttu).

7.2.4 NTLM

Microsoft ei soovita enam NTLMi kasutada ning selle asemel tuleks kasutada Kerberost[23]. Seetõttu ei ole Microsoftil ka plaane NTLMi kvantturvaliseks tegemiseks.

7.2.5 Azure

Microsoft teatas 2026. aasta juuni lõpus[24], et kiirendab aastal 2023 välja kuulutatud ajakavas Microsoft Quantum Safe Program[25] kirjeldatud protsesse, et aastaks 2029 oleks kõik Microsofti süsteemid täielikult postkvant-krüptograafiale üle viidud. Nende süsteemide sekka kuulub ka Azure. Microsofti põhilises krüptoteegis SymCrypt on kvantturvalistest

[23] https://learn.microsoft.com/en-us/windows/whats-new/deprecated-features
[24] https://www.microsoft.com/en-us/security/blog/2026/06/30/microsoft-advances-quantum-safe-security-as-the-risk-timeline-shifts/
[25] https://www.microsoft.com/en-us/security/blog/2025/08/20/quantum-safe-security-progress-towards-next-generation-cryptography/

võtmekapseldusmehhanismidest tänaseks teostatud ML-KEM ning signatuuriskeemidest XMSS, LMS, SLH-DSA ja ML-DSA[26]. See võimaldab arendajatel eri teenuste juures postkvant-krüptograafiaga eksperimenteerida, kuid selleks, et Azure'i komponendid oleks vaikimisi kvantturvalised, läheb veel aega.

7.2.6 AWS

AWS tegeleb aktiivselt postkvant-krüptograafiale üleminekuga vastavalt nende avaldatud üleminekuplaanile[27]. Kuigi AWSi üleminekuplaan ei sisalda tähtaegu konkreetsete eesmärkide täitmiseks, on näha, et palju plaanitust on juba ka ellu viidud. Nende põhiline krüptoteek AWS-LibCrypto on kooskõlas FIPS 140-3 standardiga. See on ühtlasi esimene avaliku lähtekoodiga FIPSiga kooskõlas krüptomoodul, kus on teostatud ka postkvant-algoritme[28]. Lisaks sellele võimaldab AWS paljudes oma teenustes kasutada hübriidse võtmevahetusega (ECDH+ML-KEM) TLSi: makseteenustes[29], koormusjaoturites[30] ning võtme-, sertifikaadi- ning saladuste halduses[31]. Failiedastusteenustes on realiseeritud hübriidse võtmevahetusega SSH[32] (ptk 7.1.2). Võtmehaldusteenuses on võimalik hoiustada ja kasutada ML-DSA-võtmeid[33] ning ML-DSA abil on võimalik luua ka sertifitseerimiskeskusi[34].

7.3 Tugi andmestruktuurides ja -konteinerites

7.3.1 PKCS #11

PKCS#11 on krüptograafiamoodulite liideste spetsifikatsioon, mis toimib omamoodi „ühise keelena" krüptograafiliste funktsioonide kasutajate ja allikate jaoks. Standardi esimene versioon avaldati 1995. aastal; selle põhifookus on krüptograafilistel riistvaramoodulitel ja tookenitel (nagu kiipkaardid).

Standardi värskeim versioon (2025. aasta lõpu seisuga) on versioon 3.1. Avaldatud on aga juba ka tulevase versiooni 3.2 kavand[35], mis sisaldab muuhulgas postkvant-krüptograafiaga seotud mehhanisme ja funktsioone (ML-DSA, ML-KEM jat SLH-DSA).

[26] https://techcommunity.microsoft.com/blog/microsoft-security-blog/microsofts-quantum-resistant-cryptography-is-here/4238780
[27] https://aws.amazon.com/blogs/security/aws-post-quantum-cryptography-migration-plan/
[28] https://aws.amazon.com/blogs/security/aws-lc-fips-3-0-first-cryptographic-library-to-include-ml-kem-in-fips-140-3-validation/
[29] https://aws.amazon.com/about-aws/whats-new/2025/11/aws-payments-cryptography-post-quantum-data-transit/
[30] https://aws.amazon.com/about-aws/whats-new/2025/11/network-load-balancers-post-quantum-key-exchange-tls/
[31] https://aws.amazon.com/blogs/security/ml-kem-post-quantum-tls-now-supported-in-aws-kms-acm-and-secrets-manager/
[32] https://docs.aws.amazon.com/transfer/latest/userguide/post-quantum-security-policies.html
[33] https://aws.amazon.com/about-aws/whats-new/2025/06/aws-kms-post-quantum-ml-dsa-digital-signatures/
[34] https://aws.amazon.com/about-aws/whats-new/2025/11/aws-private-ca-post-quantum-digital-certificates/
[35] https://docs.oasis-open.org/pkcs11/pkcs11-spec/v3.2/pkcs11-spec-v3.2.html

7.3.2 X.509

X.509 sertifikaatide postkvant-toe uurimisega tegeletakse hetkel päris laialdaselt. Pikemad krüptograafilised artefaktid (nagu avalikud võtmed ja digitaalsignatuurid) on juba niigi ressursinõudliku avaliku võtme taristu (PKI) jaoks oluline probleem. Praegusel hetkel võib paljude süsteemide puhul olla sobivaks lahenduseks avaliku võtme, signatuuri väärtuse ja signeerimisalgoritmi identifikaatori asendamine teistega.

Üks postkvant-sertifikaatide teostamise juures oluline küsimus on, kas kasutada neid hübriidrežiimis või mitte, kuna see suurendaks sertifikaatide mahtu veelgi. Üheks võimalikuks hübriidrežiimi realisatsiooniks on nn kameeleonsertifikaadid [226] (aegunud kavand), mida aga enam aktiivselt edasi ei arendata[36].

Välja on juba pakutud mitmeid ideid, kuidas optimeerida PKIde sertifikaadikasutust konkreetsete kasutusmallide puhul (nt mitme eri algoritmi kasutamine sertifitseerimisahela eri etappides olenevalt nende atribuutidest ja nõudmistest) või isegi neid täielikult ümber korraldada (loobumine vahesertifikaatidest, autentimiseks võtmekehtestusalgoritmide (*key agreement algorithms*) kasutamine või räsipuu-sertifikaatide kasutamine).

Üks varane uuring [227] postkvant-krüptograafia (LMS-signatuuriskeemi [228]) mõjust PKI ressursikoormusele leidis, et TLS-võtmekehtestuse aeg võib viiekordistuda, seejuures vahetatud IP-pakettide arv kahekordistub. Axelspire'i blogipostituses[37] on kokku arvestatud, et tüüpilise kolmeelemendilise X.509-sertifikaadiahela kodeeringu suurus kasvab praeguselt 2,5–3,5 kilobaidilt 12 kilobaidini (kui ahelas on signeerimiseks kasutusel ML-DSA) või 53 kilobaidini (kui kasutusel on SLH-DSA). TLS-võtmekehtestusprotokolli ServerHello-teade kasvab u neli korda suuremaks (ML-DSA kasutamisel). Cloudflare'i blogipostituses[38] leitakse, et seda on liiga palju ja kasutada tuleks räsipuu-sertifikaate. Räsipuu-sertifikaatide kasutuselevõttu toetavad ka Let's Encrypt[39] ja Google oma Chrome-veebilehitsejas[40] ning neil mõlemal on selleks konkreetne ajakava.

Hiljutine uuring [229] mõõdab sertifikaadiahela suuruse mõju ajale, mis kulub TLS-võtmekehtestusprotokolli algatamisest kuni rakenduse andmeliikluse alustamiseni. Leitakse, et oluline on, kas vahetatavad teated mahuvad ühe IP-paketi sisse ära või ei. Samas sõltub võtmekehtestuseks kuluv aeg rohkem protokolli teostusest kui andmepakettidel võrgus liikumiseks kuluvast ajast. Räsipuu-sertifikaatide kasutamine aitab teadete suurust hoida allpool IP-paketi suuruse piiri.

Avaldatud on mitmeid RFCsid või RFC-kavandeid, mis käsitlevad postkvant-krüptograafiat X.509-sertifikaatides (ja toetavad seega eespool loetletud meetodeid ja tehnoloogiaid):

- *RFC 9802* [230]: räsipõhiste signatuurialgoritmide HSS ja XMSS kasutamine X.509 avaliku võtme taristus
- *RFC 9881* [231]: X.509 avaliku võtme taristu – algoritmi identifikaatorid ML-DSA jaoks
- *RFC 9935* [232]: X.509 avaliku võtme taristu – algoritmi identifikaatorid võtmekapseldusmehhanismi ML-KEM jaoks
- *RFC 9909* [233]: X.509 avaliku võtme taristu – algoritmi identifikaatorid SLH-DSA jaoks

[36] https://www.ejbca.org/resources/keymaster-the-state-of-hybrid-certificates-in-a-post-quantum-world/
[37] https://axelspire.com/business/pqc-tls-certificate-impact/
[38] https://blog.cloudflare.com/bootstrap-mtc/
[39] https://letsencrypt.org/2026/06/03/pq-certs
[40] https://blog.google/security/cultivating-a-robust-and-efficient-quantum-safe-https/

- *draft-ietf-lamps-fn-dsa-certificates* [234] (2025): X.509 avaliku võtme taristu – algoritmi identifikaatorid FN-DSA jaoks
- *draft-ietf-lamps-pq-composite-sigs* [235] (2019): hübriidne ML-DSA X.509 avaliku võtme taristus
- *draft-ietf-lamps-pq-composite-kem* [236] (2022): hübriidne ML-KEM X.509 avaliku võtme taristus
- *draft-ietf-plants-merkle-tree-certs* [237] (2023): räsipuu-sertifikaadid.

7.3.3 Mobiilne juhiluba

Selle aruande kirjutamise seisuga ei ole avalikustatud standardites (ISO/IEC DIS 18013-5:2025, ISO/IEC DTS 18013-7:2025) mainitud postkvant-krüptograafia kasutamist. Toetatud signatuurid ja võtmevahetusprotokollid põhinevad elliptkõveratel, mis ei ole postkvant-turvalised. Jooksvate arenduste kohta ei ole ka avalikku teavet.

7.3.4 Kontrollitavad mandaadid

W3C tegeleb hetkel postkvant-krüptograafia lisamisega kontrollitavate mandaatide raamistikku. W3C plaanides on lisada kvantturvaliste algoritmide tugi lähiajal[41]. Artiklis [238] käsitletakse kontrollitavate mandaatide postkvant-krüptograafiale ülemineku aspekte ning kirjeldakse järgmiseid samme ülemineku jaoks.

7.4 Tugi pika elueaga krüptokonstruktsioonides

7.4.1 Programmikoodi allkirjastamine

Aastal 2022 avalikustas NSA algoritmide komplekti CNSA 2.0 (Commercial National Security Algorithm Suite 2.0), milles soovitab programmikoodi ja püsivara allkirjastamiseks kasutada räsipõhist signatuuriskeemi LMS ühes SHA-256 või SHA-192 räsifunktsiooniga või alternatiivselt räsipõhist signatuuriskeemi XMSS[42]. Kuigi NSA soovib, et toetus nendele algoritmidele oleks olemas juba aastaks 2025, siis täielikku üleminekut nõutakse aastaks 2030.

Mitmed erinevad ettevõtted võimaldavad programmikoodi allkirjastamiseks ülalnimetatud algoritmide kasutamist, kuid kasutusele on võetud ka ML-DSA ning SLH-DSA. AWSi teenuste kaudu on võimalik allkirjastada koodi ML-DSAd kasutades[43], Encryption Consultingu CodeSign Secure toetab ML-DSAd ja LMSi[44] ning KeyFactori SignServer toetab ML-DSAd ja SLH-DSAd[45].

7.4.2 Püsivara allkirjastamine

Püsivara allkirjastamises on arenguid, kuid suuremad ettevõtted nagu Microsoft, Apple, Intel ja AMD ei tundu olevat selles osas veel aktiivsust üles näidanud. Sellegipoolest on zeroRISC implementeerinud oma OpenTitan kiipidel SLH-DSA[46] ning wolfSSL lisas oma turvalise

[41] https://www.w3.org/2024/10/vc-wg-charter.html
[42] https://media.defense.gov/2025/May/30/2003728741/-1/-1/0/CSA_CNSA_2.0_ALGORITHMS.PDF
[43] https://aws.amazon.com/blogs/security/post-quantum-ml-dsa-code-signing-with-aws-private-ca-and-aws-kms/
[44] https://www.encryptionconsulting.com/codesign-secure-v3-02-future-of-code-signing-with-pqc/
[45] https://www.signserver.org/use-cases/quantum-ready-code-signing/
[46] https://www.zerorisc.com/blog/post-quantum-secure-boot-on-opentitan

alglaadimise teenusele wolfBoot toe ML-DSA-le, LMS-ile ja XMSS-ile[47].

7.5 Tugi sardsüsteemides

Sardsüsteemides (nagu kiipkaardid, SIM-kaardid, turvaenklaavid) rakendatav krüptograafia on üliloluline igapäevaelus kasutatavate toodete ja nende turvalisuse jaoks. Reeglina on sardsüsteemide riistvaralised võimalused, arvutusvõimsus ja mälumaht võrdlemisi piiratud. Nende süsteemide üleviimine postkvant-krüptograafiale on seetõttu teistest (nagu personaalarvutid) keerulisem.

Postkvant-krüptograafia rakendamine sardsüsteemides nõuab seetõttu üht või teist laadi optimeerimismeetodite rakendamist, sardsüsteemi rolli minimeerimist krüptograafilistes toimingutes, spetsiaalselt postkvant-krüptograafia jaoks konstrueeritud riistvaraliste kiirendite kasutamist ning potentsiaalselt veel teistsugustegi lahenduste otsimist.

Esimene teadaolev postkvant-krüptograafia ulatuslik kasutusnäide sardsüsteemides on Saksa ID-kaart[48]. Suured kiipkaardikiipide loojad (Thales[49], NXP[50] ja Infineon[51]) on väljendanud valmidust tarnida postkvant-krüptograafia võimalusega komponente. Nende toodete kättesaadavus ja arendusplaan on selle aruande valmimise hetkel veel teadmata.

2024. ja 2025. aastal hakati mõningal määral juba tegelema sedalaadi süsteemide ametliku sertifitseerimisega. Paljudel juhtudel ei kuulunud aga kas PQC-võime hindamisobjekti hulka või siis jäeti see selgesõnaliselt sertifitseeringu käsitlusalast välja.

Mobiilplatvormide turvaelementide ja usaldatavate täitmiskeskkondade (*trusted execution environment*, TEE) kohta hetkel veel lähem teave puudub. Ehkki Apple'i sõnumsideplatvorm iMessage kasutab juba 2024. aastast saadik ML-KEM-võtmeid, pole siiani saadud ametlikku kinnitust, kas iPhone'i turvaenklaaviprotsessoril on olemas postkvant-krüptograafia võimalused (sama kehtib ka Apple silicon-kiipide ja nende riistvaraliste turvaelementide kohta)[52]. Samsung on samast ajast (2025. aasta veebruaris) pärit teadete kohaselt juba integreerinud PQC oma S3SSE2A-kiibi turvaenklaaviplatvormi[53,54]. Teiste mobiiliplatvormide turvaenklaavide (nagu Qualcomm Snapdragoni TEE ja Huawei Kirini iTrustee) kohta avalikustatud informatsioon puudub.

[47] https://www.wolfssl.com/wolfboot-secure-boot-now-with-support-for-fips-204-ml-dsa-post-quantum-signature-algorithm/
[48] https://www.gi-de.com/en/about-us/press/press-releases/germany-sets-new-standards-for-secure-id-documents-in-the-quantum-computing-era
[49] https://cpl.thalesgroup.com/sites/default/files/content/white-paper/post-quantum-cryptography-solutions-wp.pdf
[50] https://www.nxp.com/company/about-nxp/smarter-world-blog/BL-SECURING-TOMORROW-PQC-STRATEGY
[51] https://www.infineon.com/market-news/2025/INFCSS202510-004
[52] https://quantumsequrity.com/blog/apple-secure-enclave-pq
[53] https://news.samsung.com/global/the-first-step-to-a-quantum-safe-future-with-samsung-knox
[54] https://www.thelec.net/news/articleView.html?idxno=5167

7.6 Tugi riistvaras

7.6.1 Biomeetrilised passid

2025. aastal algatas Euroopa Liit PQC4eMRTD[55] projekti, mille eesmärgiks on kaitsta passe kvantrünnete eest. Projektis osalevad mitmed avalikud ja eraorganisatsioonid, sealhulgas ka Thales ja Infineon [56].

Alnahawi jt artiklis [239] analüüsitakse eMRTD-rakenduste ja taristu postkvant-krüptograafiale üleviimise eri aspekte. Tõdetakse, et postkvant-krüptograafia algoritmid on juba praeguse põlvkonna riistvaral realiseeritavad, ehkki silmas tuleb pidada nende platvormide piiratud jõudlust.

Thales on välja töötanud ja realiseerinud postkvant-krüptograafia toe kiipkaardis. ANSSI on selle kiipkaardi (MultiApp 5.2 Premium PQC) sertifitseerinud[57] Common Criteria raamistikus turvatasemele EAL 6+. Saksamaal on postkvant-krüptograafia toega kiipkaarti demonstreerinud Bundesdruckerei ja G+D, kaart toetab nii krüptimist kui ka autentimist. Projektis kasutati Infineoni uue disainiga kiipi, milles on kõrvalkanalirünnete eest kaitstud realisatsioonid postkvant-turvalistele algoritmidele.

ICAO on dokumentides Doc 9303-11 ja Doc 9303-12 [240] kehtestanud spetsifikatsioonid kaasaegsete krüptograafiliste meetodite rakendamiseks ja kasutamiseks, eelkõige avaliku võtme taristu (PKI) skeemide osas. Neid skeeme peavad kasutama dokumente väljastavad riigid või organisatsioonid reisidokumentides, mis on koostatud kooskõlas dokumendiga Doc 9303. Lisaks määratleb Doc 9303 osa 13 digipitseri kasutamise, mille eesmärk on tagada mitteelektrooniliste dokumentide autentsus ja terviklus kasutades asümmeetrilist krüptograafiat. Mitteelektroonilisel dokumendil esitatud teave signeeritakse kasutades digitaalsignatuuri ning saadud signatuur kodeeritakse kahemõõtmelise vöötkoodina, mis trükitakse dokumendile endale. Samas tuleb märkida, et praegusel kujul Doc 9303-11, Doc 9303-12 ja Doc 9303-13 ei käsitle postkvant-krüptograafiat.

ICAO uute tehnoloogiate töörühm (NTWG) ja ISO WG3 algatasid tegevused kvantarvutiga seotud riskide hindamiseks ja leevendamiseks. ICAO TAG/TRIP töödokument [241] tunnistab kvantarvuti ohtu ja analüüsib mõju järgmistele krüptograafilistele protokollidele:

- passiivne autentimine – elektroonilise reisidokumendi krüptograafiline kaitse on täielikult kompromiteeritud kvantarvuti olemasolul. Mõjutatud on nii dokumendi väljastamise PKI kui ka eMRTDs talletatud andmed;
- kiip/aktiivne autentimine – kaitse eMRTD-kiibi kloonimise või asendamise vastu ei oleks enam tagatud;
- PACE (turvalise sidekanali loomine kontrollisüsteemi ja eMRTD-kiibi vahel, et tagada kaitse nuhkimise ja pealtkuulamise eest) – eMRTD-kiibi kontrolliprotseduur ei oleks enam kaitstud nuhkimise ja/või pealtkuulamise eest;
- terminali autentimine – kiibile salvestatud väga tundlike biomeetriliste andmete kaitse ei oleks enam tagatud;
- turvaline sõnumivahetus – kvantoht puudub (kui kasutatakse piisava pikkusega võtit).

Töödokument käsitleb erinevaid kvantohu leevendamise võimalusi, sealhulgas mainides ka

[55] https://www.pqc4emrtd-project.eu/
[56] https://www.biometricupdate.com/202503/thales-joins-pqc4emrtd-project-to-protect-travel-documents-from-quantum-computing-attack
[57] https://cyber.gouv.fr/produits-certifies/multiapp-52-premium-pqc-version-52

hübriidrežiimi ja PKI paralleelse infrastruktuuri kasutust. Lisaks on dokumendis mainitud lühiajaline meede – riskide vähendamiseks võib lühendada eMRTD kehtivusaega (nt 10 aastalt 5 aastani). See võib vähendada aega, mis kulub kõigi kasutuses olevate eMRTDde uuendamiseks, andes väljaandnud asutustele suurema paindlikkuse ja reageerimisvõime olukorras, kus toimub märkimisväärne läbimurre kvantarvutite arengus.

ISO SC17/WG3 ja ISO SC27/WG2 vahelise postkvant-krüptograafiale ülemineku edendamise koostöö eesmärgiks on saada spetsifikatsioonid valmis 2027. aastaks[58].

7.6.2 Füüsilised turvamoodulid (HSM)

Mõnedel füüsiliste turvamoodulite tarnijatel on juba olemas postkvant-turvaliste krüptoalgoritmide toega tooted. Peamiselt on toetatud NISTi standardiseeritud algoritmid, kuid mõned tootjad pakuvad ka teiste algoritmide toetust. Hankimisel on oluline kontrollida, kas algoritmide teostus vastab NISTi standarditele, eriti kui teostus on loodud enne standardite kehtestamist.

- Utimaco (u.trust General Purpose HSM Se-Series), Quantum Protect lisab postkvant-turvaliste algoritmide toe olemasolevale turvamoodulile[59]. Algoritmide seas on ML-KEM ja ML-DSA ning olekuga räsifunktsioonidel põhinevad signatuuriskeemid LMS ja XMSS. See lahendus on disainitud krüptograafiliselt paindlikuks: HSMi vahetamist pole vaja. Utimaco pakub lisaks ka tasuta simulaatorit, mis võimaldab algoritmide integreerimist ja jõudlust testida enne paigaldamist;
- Entrust (nShield Hardware Security Module) toetab juba NISTi standarditud postkvant-krüptograafia algoritme – ML-DSA, ML-KEM, SLH-DSA[60];
- Securosys (Primus HSM CyberVault) on tekitanud postkvant-krüptograafia valmiduse oma Primus HSM CyberVault seerias, toetades NIST-i valitud algoritme – ML-KEM, ML-DSA, SLH-DSA, HSS-LMS; XMSS[61].
- Crypto4a (QxHSM) toetab nii NISTi standarditud postkvant-krüptograafia algoritme kui ka KDF-hübriidrežiimi [242] ja Classic McEliece krüptoalgoritmi[62].
- Thales (Luna HSM) [63] toetab LMSi signatuuriskeemi ning NISTi standarditud algoritme ML-DSA ja ML-KEM. ML-DSA ja ML-DSA teostus (HSM püsivara versioon 7.9.0) võib siin aga erineda NIST standardist. Samas võib Thales vajaduse korral teha muudatusi, et tagada ühilduvust ja sujuvat üleminekut.

7.7 Krüptoinventuuri tegemise meetodid

Krüptovarade avastamine ja inventuur (CADI) on üks postkvant-krüptograafiale ülemineku olulisimaid etappe, kuid see võib olla ka üks keerulisemaid, kuna krüptograafiat kasutatakse tänapäeva IT-maailmas sisuliselt kõikjal. Inventuuri peetakse üleüldiselt *no regret*-toiminguks, kuna reeglina parandab see inventeeriva organisatsiooni krüptograafiahaldust ning aitab hallata küberturvariske. Seetõttu soovitatakse seda alustada niipea kui võimalik. Inventuuri tulemuse

[58] https://www.icao.int/sites/default/files/Meetings/FALC2025/Presentations/4-RAJESHKUMAR-need-ppt-to-run.pdf, slaid 26
[59] https://utimaco.com/data-protection/gp-hsm/application-package/quantum-protect
[60] https://www.entrust.com/company/newsroom/entrust-nshield-hsms-achieve-validation-from-nist-cryptographic-algorithm-validation-program
[61] https://www.securosys.com/en/news/securosys-launches-next-generation-hsm-pqc-ready
[62] https://crypto4a.com/products/blade-modules/qx-hsm
[63] https://thalesdocs.com/gphsm/luna/7/docs/network/Content/sdk/extensions/pqc/post_quantum_algorithms.htm

esitamiseks võib kasutada CBOM-vormingut (*Cryptographic Bill of Materials*).

CADI-metoodikate kirjeldusi leiab erialastest allikatest esialgu vaid piiratud mahus, kuivõrd ühtse metoodika määratlemine kõigi kasutusmallide jaoks on tõendatavalt keeruline. Me ei ole ka teadlikud standardite olemasolust, mis CADI-protsessile konkreetseid nõudeid seaksid.

CADIt, soovitusi selle metoodika jaoks ja olemasolevaid IT-vahendeid kirjeldame pikemalt Eesti postkvant-krüptograafiale ülemineku riiklikus teekaardis [18, Lisa B].

7.8 Standardimine

Postkvant-krüptograafia algoritme standardivad erinevad riiklikud ja rahvusvahelised, avaliku ja erasektori standardiorganisatsioonid. Otsustamaks, milliseid standardeid Eestis eelistada, soovitame jälgida ennekõike NISTi, mis on juba avaldanud arvestatava hulga standardeid. Teised skeemid, mida on käsitletud ANSSI ja BSI materjalides, ei ole veel standarditud. Seetõttu soovitame neid mitte rakendada enne, kui vastavad standardid on kättesaadavad. Lisaks soovitame järgida ANSSI ja BSI lähenemist postkvant-krüptograafia kasutamisel hübriidrežiimis, kus PQC-mehhanisme rakendatakse koos traditsiooniliste krüptograafiliste algoritmidega, et tagada üleminekuperioodil suurem turvalisus.

7.8.1 Riiklike institutsioonide standardimisprotsessid

7.8.1.1 NIST

NIST alustas postkvant-krüptograafia standardimise projektiga juba 2016. aastal. Praeguseks on standarditud ML-KEM võtmekapseldusmehhanism ning ML-DSA ja SLH-DSA signatuuriskeemid. ML-DSA ja ML-KEM on standarditud üldkasutuseks nende jõudluse tõttu. FN-DSA signatuuriskeemi ja HQC võtmekapseldushehmanismi standardid on tulekul. Soovitatavate skeemide hulka on lisatud ka olekuga räsifunktsioonidel põhinevad signatuuriskeemid XMSS ja LMS.

7.8.1.2 BSI

BSI lähtub otseselt NISTi postkvant-krüptograafia standardimise tulemustest, kuid soovitatakse ka teisi algoritme, mis pakuvad konservatiivsemat turvataset. Soovitatud võtmekapseldusmehhanismid on ML-KEM, FrodoKEM ja Classic McEliece. Soovitatud signatuuriskeemid on ML-DSA, SLH-DSA ning XMSS ja LMS. BSI märgib, et räsifunktsioonidel põhinevaid signatuuriskeeme võib kasutada ka ilma hübriidrežiimita, näiteks pika elueaga süsteemides. Muid skeeme on soovitatud kasutada hübriidrežiimis.

7.8.1.3 ANSSI

ANSSI soovitused on BSIga sarnased. Soovitatud võtmekapseldusmehhanismid on ML-KEM ja FrodoKEM ning signatuuriskeemid on ML-DSA, SLH-DSA, FN-DSA, XMSS ja LMS. Postkvant-algoritme on soovitatud kasutada hübriidrežiimis.

7.8.1.4 NCSC (UK)

Ka NCSC järgib NISTi postkvant-standardeid ja rõhutab üleminekuperioodil hübriidlahendusi. Soovitatud algoritmid on ML-KEM, ML-DSA, SLH-DSA, LMS ja XMSS. Võrreldes teiste riikide soovitustega ei soovita NCSC mitte-standardseid postvant algoritme.

7.8.1.5 Hiina

Hiina on alles alustandud oma standardimisprotsessiga, olles välja kuulutanud oma konkursi[64]. Protsessi käigus otsitakse kvantturvalisi signatuuriskeeme, võtmekapseldusmehhanisme ja võtmevahetusprotokolle. Hiina ei kaalu läänemaailmas standarditud skeemide kasutuselevõttu kohalike standarditena.

7.8.1.6 Lõuna-Korea

Lõuna-Korea postkvant-krüptograafia standardimise protsessis valiti välja kaks signatuurskeemi – AlMer [243] ja HAETAE [244] ja kaks avaliku võtmega krüptosüsteemi NTRU+ [245] ja SMAUG-T [246].

7.8.2 Rahvusvaheliste organisatsioonide standardid

Standardimise liidu aastane tööprogramm (AUWP), mis avaldati 2025. aastal, määrab kindlaks kuus arenguprioriteeti [247]. Üks neist on standardide väljatöötamine, mis suunavad kvanttehnoloogia arengut ja postkvant-krüptograafia protokollide rakendamist.

7.8.2.1 ETSI

Ametlikult tunnustatud ETSI (European Telecommunication Standards Institute) loob ja haldab tuhandeid standardeid, tehnilisi kirjeldusi ja muid tulemusi, mis on aluseks info- ja sidetehnoloogia toega toodete, süsteemide, rakenduste ja teenuste arengule ja turuedule.

Kahjuks ei leidu ETSI veebilehel selle aruande kirjutamise hetkel postkvant-krüptograafiaga seotud *standardeid*. Kõige lähemad standarditele on mõningad *tehnilised spetsifikatsioonid*, mis on seotud hübriidrežiimis võtmevahetusega:

- ETSI TS 103 744 V1.2 – Quantum-safe Hybrid Key Establishment [248];
- ETSI TS 104 015 V1.1 – Efficient Quantum-Safe Hybrid Key Exchanges with Hidden Access Policies [249].

Tehnilised aruanded, mis käsitlevad postkvant-krüptograafiat, on palju mitmekesimad. Mõned neist on näiteks järgmised:

- ETSI TR 103 616 – Overview of PQ digital signatures [250];
- ETSI TR 103 823 – PQ encryption and KEM overview [251];
- ETSI TR 103 966 – Deployment Considerations for Hybrid Schemes [252];
- ETSI TR 103 967 – Impact of Quantum Computing on Symmetric Cryptography [253];
- ETSI TR 104 005 – Technical Report on impacts of the post-quantum cryptography on ETSI TC SET specifications [254],
- ETSI TR 103 619 – Migration strategies and recommendations to Quantum Safe schemes [255].

2026. aasta märtsis kiideti heaks ETSI TC ESI (*Technical Committee on Electronic Signatures and Trust Infrastructures*) postkvant-krüptograafia töörühm (ESI PQC). Rühm vastutab soovituste, suuniste ja strateegiate koostamise eest olemasolevate ESI väljundite ning PKI ja usaldusteenuste infrastruktuuride üleminekuks postkvant-krüptograafiale.

[64] https://www.niccs.org.cn/niccs/Notice/pc/content/content_1975896137741635584.html

7.8.2.2 CEN/CENELEC

CEN/CENELEC on rohkem keskendunud kvantstandarditele. 2022. aastal loodud CEN-CLC/JTC 22 tugineb CENi ja CENELECi kvanttehnoloogiate fookusgrupi abil loodud alusele, mis hõlmab kvanttehnoloogiate standardimise teekaarti [256] ja kvanttehnoloogiate kasutusmalle kirjeldavat dokumenti [257]. CEN-CLC/JTC 22 eesmärgiks on luua standardimisdokumentide tulemid kvanttehnoloogia valdkonnas (kaasa arvatud kvantkrüptograafia). Postkvant-krüptograafia on mainitud kvanttehnoloogia kontekstis, hübriidlahendusena QKD süsteemides [257].

CEN/CLC/JTC 13/WG 10 käsitleb riiklike ja rahvusvaheliste standardite ülevõtmist Euroopa standarditena krüptograafia valdkonnas, sealhulgas postkvant-krüptograafias[65].

7.8.2.3 ISO

ISO on teadlik kvantrünnete ohust ja liigub algoritmide standardimise suunas. Standardisse ISO/IEC 14888-4:2024 on juba lisatud olekuga räsifunktsioonidel põhinevad signatuuriskeemid LMS, XMSS [66]. Teised signatuuriskeemid ja võtmekapseldusmehhanismid ei ole veel standarditud, kuid on teada, et praegu menetletakse rahvusvahelise standardi kavandit (DIS), mis lisaks olemasolevale standardile ISO/IEC 18033-2 [52] ML-KEMi, FrodoKEMi ja Classic McEliece'i.

7.8.2.4 IETF

IETF tegeleb aktiivselt postkvant-krüptograafia standardimisega mitmes töörühmas. Koostatud on olekuga räsifunktsioonidel põhinevaid signatuuriskeeme käsitlevad RFCd – LMS [228] ja XMSS [258]. Lisaks on välja pakutud meetod, mille abil saab IKEv2s teha mitu järjestikust võtmevahetust. See meetod võimaldab integreerida postkvant-krüptograafia IKEv2-protokolli, säilitades samas tagasiühilduvuse, mis võimaldab tuletada kvantrünnete eest kaitstud IKE võtmepaare [259]. TLSi jaoks on olemas laiend sertifikaadipõhiseks autentimiseks välise eeljagatud võtmega [260].

7.8.2.5 W3C

W3C on alustanud postkvant-krüptograafiaga tegelemisega ja mitmed algatused on pooleli, kuid ükski postkvant-algoritm ei ole saanud ametlikuks standardiks. Näiteks dokumendikavandis „Cryptography usage in Web Standards" [261] mainitakse nii postkvant-võtmekapseldusmehhanisme (ML-KEM ja HQC) kui ka signatuuriskeeme (ML-DSA, SLH-DSA, FN-DSA). „Quantum-Safe Cryptosuites v0.3" [262], mis on W3C Credentials Community Group'i spetsifikatsioon (dokumendikavand), määratleb milliseid postkvant-skeeme tuleks kasutada verifitseeritavate tõendite jaoks tuues eraldi välja ML-DSA, SLH-DSA, FN-DSA ja SQISign signatuuriskeemid.

7.9 Olemasolev krüptomaterjal

Mida teha juba olemasoleva, kvantrünnetele avatud algoritme kasutades loodud krüptomaterjaliga, kui krüptograafiliselt märkimisväärse kvantarvuti loomise päev on lähenemas? Krüptomaterjal võib siinkohal hõlmata võtmeid, krüptotekste, signatuure, sõnumilühendeid ja võib-olla veel midagi muud (näiteks interaktsioonivabu nullteadmustõestusi?).

[65] https://standards.cencenelec.eu/ords/f?p=305:7:::::FSP_ORG_ID:3407680
[66] https://www.iso.org/standard/80492.html (Information security – Digital signatures with appendix. Part 4: Stateful hash-based mechanisms)

Võtmed on vaja välja vahetada. See on osa kasutusel olevate infosüsteemide uuendamisest vastavalt siin peatükis kirjeldatud uuendamisvõimalustele. Krüptotekstidega, juhul kui nad on juba üle kaitsmata võrguühenduste liikunud (ja selle liikluse võimaldamine ongi üks krüptimise eesmärke), pole midagi eriti mõistlikku peale hakata: kui ründaja on nad salvestanud, siis ootab ta, kuni kvantarvuti aitab tal need krüptotekstid lahti murda. Kui on alust arvata, et krüptotekst ei ole veel üle kaitsmata võrguühenduse liikunud, siis tuleks hoolitseda selle eest, et selline olukord püsiks. Peale infosüsteemi uuendamist tuleks see krüptotekst asendada krüptotekstiga, mis on loodud kvantkindla algoritmi abil.

Signatuurid ja muu *tervikluse* tagamiseks loodud krüptomaterjal on võimalik ette valmistada ajaks, mil krüptoalgoritme, millega nad loodud olid, enam turvaliseks ei peeta. Selle jaoks on vaja signatuurile, mille tõendusväärtust soovitakse pikaajaliselt säilitada, lisada tõendid, mis kinnitavad selle signatuuri ja seda konkreetse allkirjastajaga siduvate sertifikaatide olemasolu enne selliste (kvant)arvutite ilmumist, mis kasutatud signeerimisalgoritme murda suudavad. Signatuurid, mis neid tõendeid sisaldavad, vastavad täiustatud elektroonilise allkirja (ptk. 4.2) *Long-Term Archival (LTA)* profiilile.

Täiustatud elektroonilise allkirja standardid [139] võimaldavad funktsionaalsust, kus mingile profiilile vastavale signatuurile lisatakse täiendavaid tõendusatribuute, mille tulemusena saadakse LTA-profiilile vastav signatuur. Seejuures lisaja ei pea olema sama, mis esialgne signeerija. ETSI viis 2023. aasta sügisel läbi ühilduvustesti [263], kus selliseid lisamisi ja täiendatud signatuuride verifitseerimist katsetati; testis osales 122 organisatsiooni 38-st riigist.

E-identimise ja e-tehingute jaoks vajalike usaldusteenuste (eIDAS) määruse [137] 34. artikkel reguleerib kvalifitseeritud e-allkirjade kvalifitseeritud säilitamisteenuseid. Selle artikli alusel kehtestab määrus nr. 2025/1946 [264] standardid ETSI TS 119 511 signatuuride pikaajaliseks säilitamiseks [265] ja ETSI TS 119 172 selliselt säilitatud signatuuride verifitseerimiseks [266]. Säilitamine, mille eesmärgiks on signatuuri tõendusväärtuse pikaajaline hoidmine neid tõendeid vajadusel täiendades, on üks usaldusteenustest. Säilitusteenuse kvalifitseeritud pakkujad on leitavad eIDAS-e usaldusteenuste pakkujate nimekirjast; käesoleva aruande kirjutamise ajal (mai 2026) oli selles nimekirjas 32 kvalifitseeritud säilitusteenuse tarnijat, neist 29 aktiivset.

Summary in English

This is the eighth report in the series of reports on cryptographic algorithms and their life cycle. The report covers six topics. Its intended audience are managers and practitioners both in IT and in software development. As usual, the report describes several algorithms, protocols, formats and their secure usage. It takes a closer look at some of the protocols and the development tools for applications that use cryptography. The most important point is still that for practical evaluation of use of cryptography in any single context, one should hire a security engineer.

Section two focuses on cryptographic primitives and contains current recommendations on key length and other relevant parameters that need consideration to keep systems secure. It also contains an overview of the main types of cryptographic primitives and relevant algorithms that one could use. Compared to previous reports, the coverage of different cryptographic algorithms has been expanded, now also discussing the key lengths of postquantum algorithms (and the efficiency of all algorithms).

Section three describes several families of widely used communication protocols. Where relevant, it gives recommendations about their secure use. The section on TLS also describes other measures that are relevant when protecting one's website or other services with TLS. Other protocols covered include SSH, IPsec, and the VPN protocols OpenVPN and WireGuard. Section three also discusses the security aspects of Bluetooth and NFC, as part of the stack for digital identity wallets (DIW). DIWs are an addition to this report; parts of their stack are covered in Sections 3–5.

Section four describes several container formats used to cryptographically secure one-time messages between parties or documents at rest. Covered formats include CMS, ASiC and its variants, JOSE formats and Estonian local format CDOC. The wallet stack is represented by mDoc and W3C credential formats.

Section five, on authentication and authorisation protocols describes OAuth 2.0 and OpenID Connect, and several potential problems that might occur while implementing it. The Section also describes specifications produced by FIDO Alliance, such as WebAuthn, and the use of Passkeys. Considering the Estonian context, this Section describes Web eID, used for the authentication and signing in browsers with the help of identity cards. Finally, the Section describes protocols for credential presentation.

Section six describes tools to implement cryptographic measures in software. It begins with an overview of popular libraries that implement cryptographic primitives or protocols, such as OpenSSL. The rest of the Section describes support for cryptography in popular development environments and programming languages, such as Java, Go, Rust and PHP.

The topic of Section seven is very timely; it is about the state of migration to postquantum cryptography in all the protocols, data structures, and applications we have discussed. We also describe the state of standards documents.

Kirjandus

[1] *Krüptograafiliste algoritmide kasutusvaldkondade ja elutsükli uuring (Report on the life cycle of cryptographic algorithms)*. https://www.ria.ee/sites/default/files/content-editors/publikatsioonid/kryptoalgoritmide_elutsykli_uuring_15-07-2011.pdf. Cybernetica report no. A-60-1 (in Estonian). 2011.

[2] *Krüptograafiliste algoritmide kasutusvaldkondade ja elutsükli uuring (Report on the life cycle of cryptographic algorithms)*. https://www.ria.ee/sites/default/files/content-editors/publikatsioonid/kruptograafiliste_algoritmide_elutsukli_uuring_ii.pdf. Cybernetica report no. A-77-5 (in Estonian). 2013.

[3] *Krüptograafiliste algoritmide kasutusvaldkondade ja elutsükli uuring (Report on the life cycle of cryptographic algorithms)*. https://www.ria.ee/sites/default/files/content-editors/publikatsioonid/kruptograafiliste_algoritmide_uuring_2015.pdf. Cybernetica report no. A-101-1 (in Estonian). 2015.

[4] *Cryptographic algorithms lifecycle report 2016*. https://www.ria.ee/sites/default/files/content-editors/publikatsioonid/cryptographic_algorithms_lifecycle_report_2016.pdf. Cybernetica report no. A-101-3. 2016.

[5] *Krüptograafiliste algoritmide kasutusvaldkondade ja elutsükli uuring (Report on the life cycle of cryptographic algorithms)*. https://www.ria.ee/sites/default/files/content-editors/publikatsioonid/kruptograafiliste_algoritmide_elutsukli_uuring_2017.pdf. Cybernetica report no. A-101-9 (in Estonian). 2017.

[6] *Postkvant-krüptograafia ülevaade (Report on the post-quantum cryptographic algorithms)*. https://www.ria.ee/sites/default/files/content-editors/publikatsioonid/postkvant-kruptograafia-ulevaade-2018.pdf. Cybernetica report no. A-101-10 (in Estonian). 2018.

[7] *Krüptoalgoritmid ning nende tugi teekides ja infosüsteemides (Cryptographic algorithms and their support in libraries and information systems)*. https://www.ria.ee/sites/default/files/content-editors/publikatsioonid/cryptoreport2021.pdf. Cybernetica report no. T-184-7, v1.0 (in Estonian). 2021.

[8] *Krüptoalgoritmid ning nende tugi teekides ja infosüsteemides (Cryptographic algorithms and their support in libraries and information systems)*. https://www.ria.ee/sites/default/files/documents/2023-06/Cryptoreport%202023%20final.pdf. Cybernetica report no. T-184-7, v2.0 (in Estonian). 2023.

[9] Elaine Barker. *Recommendation for Key Management. Part 1: General*. NIST Special Publication 800-57 Part 1, Revision 5, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf. Mai 2020.

[10] Lov. K Grover. „A fast quantum mechanical algorithm for database search“. Teoses: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*. STOC '96. Philadelphia, Pennsylvania, USA: Association for Computing Machinery, 1996, lk-d 212–219. ISBN: 0897917855. DOI: 10.1145/237814.237866. URL: https://doi.org/10.1145/237814.237866.

[11] Gilles Brassard, Peter Høyer ja Alain Tapp. „Quantum cryptanalysis of hash and claw-free functions“. Teoses: *LATIN'98: Theoretical Informatics*. Toim. Cláudio L. Lucchesi ja Arnaldo V. Moura. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, lk-d 163–169. ISBN: 978-3-540-69715-2.

[12] National Institute of Standards ja Technology. *Submission Requirements and Evaluation Criteria for the Post-Quantum Cryptography Standardization Process*. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf. Accessed: 2022-12-09. Dets 2016.

[13] Dustin Moody. *Digital Signature Standard (DSS)*. FIPS PUB 186-5. 2023. DOI: 10.6028/nist.fips.186-5.

[14] National Institute of Standards and Technology (US). *Stateless hash-based digital signature standard*. Tehniline aruanne NIST FIPS 205. Washington, D.C.: National Institute of Standards ja Technology (U.S.), aug 2024, NIST FIPS 205. DOI: 10.6028/NIST.FIPS.205. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.205.pdf (vaadatud 29.09.2025).

[15] *Agreed Cryptographic Mechanisms*. 2025. URL: https://certification.enisa.europa.eu/document/download/a845662b-aee0-484e-9191-890c4cfa7aaa_en?filename=ECCG%20Agreed%20Cryptographic%20Mechanisms%20version%202.pdf.

[16] EU PQC Workstream. *A Coordinated Implementation Roadmap for the Transition to Post-Quantum Cryptography*. Technical Report Part 1, Version 1.1. Accessed: 2025-01-05. NIS Cooperation Group, juuni 2025.

[17] Alessandro Amadori *et al. The PQC Migration Handbook: Guidelines for Migrating to Post-Quantum Cryptography*. Technical Handbook. Accessed: 2025-10-03. AIVD, CWI ja TNO, dets 2024.

[18] *Eesti postkvant-krüptograafiale ülemineku riiklik teekaart (Estonian National Roadmap for Migration to Post-Quantum Cryptography)*. https://www.justdigi.ee/digi-side-ja-kuber/analuusid-ja-uuringud. Cybernetica report no. D-16-1248, v1.0 (in Estonian). Apr 2026.

[19] *Agreed Cryptographic Mechanisms. Working draft*. Version 3 submitted to public review until the end of July 2026. https://certification.enisa.europa.eu/document/download/9993abb8-5f27-47d0-bdb2-f6f284cd5141_en?filename=20260507-acm-draft.pdf. Apr 2026.

[20] Phillip Rogaway ja Thomas Shrimpton. „Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance“. Teoses: *Fast Software Encryption, 11th International Workshop, FSE 2004, Delhi, India, February 5-7, 2004, Revised Papers*. Toim. Bimal K. Roy ja Willi Meier. Kd 3017. Lecture Notes in Computer Science. Springer, 2004, lk-d 371–388.

[21] *Secure Hash Standard (SHS)*. FIPS PUB 180-4, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf. Aug 2015.

[22] Karthikeyan Bhargavan ja Gaëtan Leurent. „Transcript Collision Attacks: Breaking Authentication in TLS, IKE and SSH“. Teoses: *23rd Annual Network and Distributed System Security Symposium, NDSS 2016, San Diego, California, USA, February 21-24, 2016*. The Internet Society, 2016. URL: http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2017/09/transcript-collision-attacks-breaking-authentication-tls-ike-ssh.pdf.

[23] Yingxin Li, Fukang Liu ja Gaoli Wang. „New Records in Collision Attacks on SHA-2“. Teoses: *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part I*. Toim. Marc Joye ja Gregor Leander. Lecture Notes in Computer Science. Springer, 2024, lk-d 158–186. DOI: 10.1007/978-3-031-58716-0_6. URL: https://doi.org/10.1007/978-3-031-58716-0%5C_6.

[24] Yingxin Li *et al.* „The First Practical Collision for 31-Step SHA-256“. Teoses: *Advances in Cryptology - ASIACRYPT 2024 - 30th International Conference on the Theory and Application of Cryptology and Information Security, Kolkata, India, December 9-13, 2024, Proceedings, Part VII*. Toim. Kai-Min Chung ja Yu Sasaki. Lecture Notes in Computer Science. Springer, 2024, lk-d 237–266. DOI: 10.1007/978-981-96-0941-3_8. URL: https://doi.org/10.1007/978-981-96-0941-3%5C_8.

[25] Yingxin Li *et al.* „New Collision Attacks on Round-Reduced SHA-512“. Teoses: *Advances in Cryptology - CRYPTO 2025 - 45th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2025, Proceedings, Part V*. Toim. Yael Tauman Kalai ja Seny F. Kamara. Lecture Notes in Computer Science. Springer, 2025, lk-d 200–229. DOI: 10.1007/978-3-032-01901-1_7. URL: https://doi.org/10.1007/978-3-032-01901-1%5C_7.

[26] Zhuolong Zhang *et al.* „Collision Attacks on SHA-256 up to 37 Steps with Improved Trail Search“. Teoses: *Advances in Cryptology - EUROCRYPT 2026 - 45th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Rome, Italy, May 10-14, 2026, Proceedings, Part VI*. Toim. Joan Daemen ja Emmanuel Thomé. Lecture Notes in Computer Science. Springer, 2026, lk-d 91–120. DOI: 10.1007/978-3-032-25333-0_4. URL: https://doi.org/10.1007/978-3-032-25333-0%5C_4.

[27] Jian Guo *et al.* „Dual-Syncopation Meet-in-the-Middle Attacks: New Results on SHA-2 and MD5“. Teoses: *Advances in Cryptology - EUROCRYPT 2026 - 45th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Rome, Italy, May 10-14, 2026, Proceedings, Part VI*. Toim. Joan Daemen ja Emmanuel Thomé. Lecture Notes in Computer Science. Springer, 2026, lk-d 121–151. DOI: 10.1007/978-3-032-25333-0_5. URL: https://doi.org/10.1007/978-3-032-25333-0%5C_5.

[28] *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*. FIPS PUB 202, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf. Aug 2015.

[29] Hugo Krawczyk ja Pasi Eronen. *HMAC-based Extract-and-Expand Key Derivation Function (HKDF)*. RFC 5869. Mai 2010. URL: https://www.rfc-editor.org/rfc/rfc5869.txt.

[30] David Temoshok *et al. Digital Identity Guidelines: Authentication and Authenticator Management*. Digital Identity Guidelines, NIST Special Publication 800-63B-4, https://pages.nist.gov/800-63-4/sp800-63b.html. Aug 2025.

[31] Alex Biryukov *et al. Argon2 Memory-Hard Function for Password Hashing and Proof-of-Work Applications*. RFC 9106. Sept 2021. URL: https://rfc-editor.org/rfc/rfc9106.txt.

[32] *Cryptographic Mechanisms: Recommendations and Key Lengths*. BSI – Technical Guideline TR-02102-1, https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.pdf. Nov 2022.

[33] Meltem Sönmez Turan *et al. Recommendation for Password-Based Key Derivation Part 1: Storage Applications*. NIST Special Publication 800-132, https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-132.pdf. Dets 2010.

[34] *Advanced Encryption Standard (AES)*. FIPS PUB 197, https://doi.org/10.6028/NIST.FIPS.197-upd1. Mai 2023.

[35] Daniel J. Bernstein. *ChaCha, a variant of Salsa20*. https://cr.yp.to/chacha/chacha-20080128.pdf. Jaan 2008.

[36] Yoav Nir ja Adam Langley. *ChaCha20 and Poly1305 for IETF Protocols*. RFC 8439. Juuni 2018. DOI: 10.17487/RFC8439. URL: https://www.rfc-editor.org/info/rfc8439.

[37] Daniel J. Bernstein. *The Salsa20 family of stream ciphers*. https://cr.yp.to/snuffle/salsafamily-20071225.pdf. Dets 2007.

[38] Hugo Krawczyk, Mihir Bellare ja Ran Canetti. *HMAC: Keyed-Hashing for Message Authentication*. RFC 2104. Veebr 1997. URL: https://rfc-editor.org/rfc/rfc2104.txt.

[39] David McGrew. *An Interface and Algorithms for Authenticated Encryption*. RFC 5116. Jaan 2008. URL: https://www.rfc-editor.org/rfc/rfc5116.txt.

[40] Ronald L. Rivest, Adi Shamir ja Leonard M. Adleman. „A Method for Obtaining Digital Signatures and Public-Key Cryptosystems“. *Commun. ACM* 21.2 (1978), lk-d 120–126. DOI: 10.1145/359340.359342. URL: https://doi.org/10.1145/359340.359342.

[41] Ahto Buldas *et al.* „Server-Supported RSA Signatures for Mobile Devices“. Teoses: *Computer Security - ESORICS 2017 - 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part I*. Toim. Simon N. Foley, Dieter Gollmann ja Einar Snekkenes. Kd 10492. Lecture Notes in Computer Science. Springer, 2017, lk-d 315–333. DOI: 10.1007/978-3-319-66402-6_19. URL: https://doi.org/10.1007/978-3-319-66402-6_19.

[42] Taher El Gamal. „A public key cryptosystem and a signature scheme based on discrete logarithms“. *IEEE Trans. Inf. Theory* 31.4 (1985), lk-d 469–472.

[43] *IVXV protokollide kirjeldus*. https://www.valimised.ee/et/e-haaletamine/dokumendid. Vabariigi Valimiskomisjon IVXV-PR-1.10.0 (in Estonian). 2025.

[44] *Digital Signature Standard (DSS)*. FIPS PUB 186-4, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf. Juuli 2013.

[45] Adam Langley, Mike Hamburg ja Sean Turner. *Elliptic Curves for Security*. RFC 7748. Jaan 2016. URL: https://rfc-editor.org/rfc/rfc7748.txt.

[46] Johannes Merkle ja Manfred Lochter. *Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation*. RFC 5639. Märts 2010. URL: https://rfc-editor.org/rfc/rfc5639.txt.

[47] Whitfield Diffie ja Martin E. Hellman. „New directions in cryptography“. *IEEE Trans. Inf. Theory* 22.6 (1976), lk-d 644–654. DOI: 10.1109/TIT.1976.1055638. URL: https://doi.org/10.1109/TIT.1976.1055638.

[48] Simon Josefsson ja Ilari Liusvaara. *Edwards-Curve Digital Signature Algorithm (EdDSA)*. RFC 8032. Jaan 2017. DOI: 10.17487/RFC8032. URL: https://www.rfc-editor.org/info/rfc8032.

[49] National Institute of Standards and Technology (US). *Module-lattice-based key-encapsulation mechanism standard*. Tehniline aruanne NIST FIPS 203. Washington, D.C.: National Institute of Standards ja Technology (U.S.), aug 2024, NIST FIPS 203. DOI: 10.6028/NIST.FIPS.203. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.203.pdf (vaadatud 29.09.2025).

[50] Philippe Gaborit *et al. HQC specifications*. 2025. URL: https://pqc-hqc.org/doc/hqc_specifications_2025_08_22.pdf.

[51] Erdem Alkim *et al. FrodoKEM: Learning With Errors Key Encapsulation. Preliminary Standardization Proposal*. 2024. URL: https://frodokem.org/files/FrodoKEM_standard_proposal_20241205.pdf.

[52] *ISO/IEC 18033-2:2006/Amd 2:2026; Information technology - Security techniques - Encryption algorithms - Part 2: Asymmetric ciphers; Amendment 2*. https://www.iso.org/standard/86890.html. Juuni 2026.

[53] Daniel J. Bernstein *et al. Classic McEliece:conservative code-based cryptography:cryptosystem specification*. 2022. URL: https://classic.mceliece.org/mceliece-spec-20221023.pdf.

[54] National Institute of Standards and Technology (US). *Module-lattice-based digital signature standard*. Tehniline aruanne NIST FIPS 204. Washington, D.C.: National Institute of Standards ja Technology (U.S.), aug 2024, NIST FIPS 204. DOI: 10.6028/NIST.FIPS.204. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.204.pdf (vaadatud 29.09.2025).

[55] Vincent Quentin Ulitzsch *et al.* „Profiling Side-Channel Attacks on Dilithium - A Small Bit-Fiddling Leak Breaks It All“. Teoses: *Selected Areas in Cryptography - 29th International Conference, SAC 2022, Windsor, ON, Canada, August 24-26, 2022, Revised Selected Papers*. Toim. Benjamin Smith ja Huapeng Wu. Lecture Notes in Computer Science. Springer, 2022, lk-d 3–32. DOI: 10.1007/978-3-031-58411-4_1. URL: https://doi.org/10.1007/978-3-031-58411-4%5C_1.

[56] Ruize Wang *et al.* „Unpacking Needs Protection: A Single-Trace Secret Key Recovery Attack on Dilithium“. *IACR Commun. Cryptol.* 1.3 (2024), lk 12. DOI: 10.62056/A0FH89N4E. URL: https://doi.org/10.62056/a0fh89n4e.

[57] Alexandre Berzati *et al.* „Exploiting Intermediate Value Leakage in Dilithium: A Template-Based Approach“. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2023.4 (2023), lk-d 188–210. DOI: 10.46586/TCHES.V2023.I4.188-210. URL: https://doi.org/10.46586/tches.v2023.i4.188-210.

[58] Yuanyuan Zhou *et al.* „Rejected Signatures' Challenges Pose New Challenges: Key Recovery of CRYSTALS-Dilithium via Side-Channel Attacks“. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2025.4 (2025), lk-d 817–847. DOI: 10.46586/TCHES.V2025.I4.817-847. URL: https://doi.org/10.46586/tches.v2025.i4.817-847.

[59] Thomas Prest *et al. Falcon: Algorithm specifications and supporting documentation, Selected Algorithms of NIST's post-quantum cryptography standardization process*. https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022. Accessed: 2022-12-08. Nov 2022.

[60] Thomas Pornin. *New Efficient, Constant-Time Implementations of Falcon*. Cryptology ePrint Archive, Paper 2019/893. https://eprint.iacr.org/2019/893. 2019.

[61] Quynh Dang ja Dustin Moody. *Additional SLH-DSA Parameter Sets for Limited-Signature Use Cases*. NIST Special Publication 800-230, Initial Public Draft, https://doi.org/10.6028/NIST.SP.800-230.ipd. Apr 2026.

[62] Scott Fluhrer. *Side Channel Resistant Sphincs+*. Cryptology ePrint Archive, Paper 2024/500. 2024. URL: https://eprint.iacr.org/2024/500.

[63] Jeremy Boy *et al.* „SLasH-DSA: Breaking SLH-DSA Using an Extensible End-To-End Rowhammer Framework“. *CoRR* abs/2509.13048 (2025). DOI: 10.48550/ARXIV.2509.13048. arXiv: 2509.13048. URL: https://doi.org/10.48550/arXiv.2509.13048.

[64] David A. Cooper *et al. Recommendation for Stateful Hash-Based Signature Schemes*. 2020. DOI: 10.6028/nist.sp.800-208. URL: http://dx.doi.org/10.6028/NIST.SP.800-208.

[65] Nina Bindel *et al. Transitioning to a Quantum-Resistant Public Key Infrastructure*. Cryptology ePrint Archive, Paper 2017/460. https://eprint.iacr.org/2017/460. 2017. URL: https://eprint.iacr.org/2017/460.

[66] Julien Devevey, Morgane Guerreau ja Maxime Roméas. *Compact, Efficient and Non-Separable Hybrid Signatures*. Cryptology ePrint Archive, Paper 2025/2059. 2025. URL: https://eprint.iacr.org/2025/2059.

[67] Manuel Barbosa *et al.* „X-Wing". *IACR Communications in Cryptology* 1.1 (9. apr 2024). ISSN: 3006-5496. DOI: 10.62056/a3qj89n4e.

[68] Gorjan Alagic *et al. Recommendations for Key-Encapsulation Mechanisms*. Tehniline aruanne NIST SP 800-227. National Institute of Standards ja Technology, 2025, NIST SP 800–227. DOI: 10.6028/NIST.SP.800-227. URL: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-227.pdf (vaadatud 29.09.2025).

[69] NIST. *Post-Quantum Cryptography: FAQs*. https://csrc.nist.gov/Projects/post-quantum-cryptography/faqs. 2026.

[70] Eric Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.3*. RFC 8446. Aug 2018. URL: https://rfc-editor.org/rfc/rfc8446.txt.

[71] *The Transport Layer Security (TLS) Protocol Version 1.3*. RFC 9846. Juuli 2026. DOI: 10.17487/RFC9846. URL: https://www.rfc-editor.org/info/rfc9846.

[72] Eric Rescorla ja Tim Dierks. *The Transport Layer Security (TLS) Protocol Version 1.2*. RFC 5246. Aug 2008. DOI: 10.17487/RFC5246. URL: https://www.rfc-editor.org/info/rfc5246.

[73] Eric Rescorla, Hannes Tschofenig ja Nagendra Modadugu. *The Datagram Transport Layer Security (DTLS) Protocol Version 1.3*. RFC 9147. Apr 2022. DOI: 10.17487/RFC9147. URL: https://www.rfc-editor.org/info/rfc9147.

[74] Deirdre Connolly. *ML-KEM Post-Quantum Key Agreement for TLS 1.3*. Internet-Draft draft-ietf-tls-mlkem-07. Work in Progress. Internet Engineering Task Force, veebr 2026. 11 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-tls-mlkem/07/.

[75] Chris Brzuska *et al.* „Key-Schedule Security for the TLS 1.3 Standard". Teoses: *Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part I*. Toim. Shweta Agrawal ja Dongdai Lin. Kd 13791. Lecture Notes in Computer Science. Springer, 2022, lk-d 621–650. DOI: 10.1007/978-3-031-22963-3_21.

[76] Nir Drucker ja Shay Gueron. „Selfie: reflections on TLS 1.3 with PSK". *J. Cryptol.* 34.3 (2021), lk 27. DOI: 10.1007/s00145-021-09387-y.

[77] R. Housley *et al. Guidance for External Pre-Shared Key (PSK) Usage in TLS*. RFC 9257. Juuli 2022. URL: https://rfc-editor.org/rfc/rfc9257.txt.

[78] Karthikeyan Bhargavan, Bruno Blanchet ja Nadim Kobeissi. „Verified Models and Reference Implementations for the TLS 1.3 Standard Candidate". Teoses: *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*. IEEE Computer Society, 2017, lk-d 483–502. DOI: 10.1109/SP.2017.26. URL: https://doi.org/10.1109/SP.2017.26.

[79] *DoD standard Transmission Control Protocol*. RFC 761. Jaan 1980. DOI: 10.17487/RFC0761. URL: https://www.rfc-editor.org/info/rfc761.

[80] Daniel Bleichenbacher. „Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1“. Teoses: *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*. Toim. Hugo Krawczyk. Kd 1462. Lecture Notes in Computer Science. Springer, 1998, lk-d 1–12.

[81] Hubert Kario. „Everlasting ROBOT: The Marvin Attack“. Teoses: *Computer Security - ESORICS 2023 - 28th European Symposium on Research in Computer Security, The Hague, The Netherlands, September 25-29, 2023, Proceedings, Part III*. Toim. Gene Tsudik *et al.* Lecture Notes in Computer Science. Springer, 2023, lk-d 243–262. DOI: 10.1007/978-3-031-51479-1_13. URL: https://doi.org/10.1007/978-3-031-51479-1%5C_13.

[82] Ka Fun Tang, Ka Lok Wu ja Sze Yiu Chau. „Investigating TLS Version Downgrade in Enterprise Software“. Teoses: *Proceedings of the Fourteenth ACM Conference on Data and Application Security and Privacy, CODASPY 2024, Porto, Portugal, June 19-21, 2024*. Toim. João P. Vilela, Haya Schulmann ja Ninghui Li. ACM, 2024, lk-d 31–42. DOI: 10.1145/3626232.3653263. URL: https://doi.org/10.1145/3626232.3653263.

[83] Pasi Eronen *et al. Transport Layer Security (TLS) Session Resumption without Server-Side State*. RFC 5077. Jaan 2008. DOI: 10.17487/RFC5077. URL: https://www.rfc-editor.org/info/rfc5077.

[84] Sven Hebrok *et al.* „We Really Need to Talk About Session Tickets: A Large-Scale Analysis of Cryptographic Dangers with TLS Session Tickets“. Teoses: *32nd USENIX Security Symposium, USENIX Security 2023, Anaheim, CA, USA, August 9-11, 2023*. Toim. Joseph A. Calandrino ja Carmela Troncoso. USENIX Association, 2023, lk-d 4877–4894. URL: https://www.usenix.org/conference/usenixsecurity23/presentation/hebrok.

[85] Yoongu Kim *et al.* „Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors“. Teoses: *ACM/IEEE 41st International Symposium on Computer Architecture, ISCA 2014, Minneapolis, MN, USA, June 14-18, 2014*. IEEE Computer Society, 2014, lk-d 361–372. DOI: 10.1109/ISCA.2014.6853210.

[86] Youssef Tobah *et al.* „Go Go Gadget Hammer: Flipping Nested Pointers for Arbitrary Data Leakage“. Teoses: *33rd USENIX Security Symposium, USENIX Security 2024, Philadelphia, PA, USA, August 14-16, 2024*. Toim. Davide Balzarotti ja Wenyuan Xu. USENIX Association, 2024. URL: https://www.usenix.org/conference/usenixsecurity24/presentation/tobah.

[87] Andrew J. Adiletta *et al.* „LeapFrog: The Rowhammer Instruction Skip Attack“. Teoses: *10th IEEE European Symposium on Security and Privacy, EuroS&P 2025, Venice, Italy, June 30 - July 4, 2025*. IEEE, 2025, lk-d 1067–1081. DOI: 10.1109/EUROSP63326.2025.00065. URL: https://doi.org/10.1109/EuroSP63326.2025.00065.

[88] Burt Kaliski. *PKCS #1: RSA Encryption Version 1.5*. RFC 2313. Märts 1998. URL: https://rfc-editor.org/rfc/rfc2313.txt.

[89] Eyal Ronen *et al.* „The 9 Lives of Bleichenbacher's CAT: New Cache ATtacks on TLS Implementations“. Teoses: *IEEE Symposium on Security and Privacy*. San Francisco: IEEE, 20. mai 2019, lk-d 966–983.

[90] Colin Boyd, Anish Mathuria ja Douglas Stebila. *Protocols for Authentication and Key Establishment, Second Edition*. Information Security and Cryptography. Springer, 2020. ISBN: 978-3-662-58145-2.

[91] Daniel Kahn Gillmor. *Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS)*. RFC 7919. Aug 2016. URL: https://rfc-editor.org/rfc/rfc7919.txt.

[92] David McGrew ja Daniel Bailey. *AES-CCM Cipher Suites for Transport Layer Security (TLS)*. RFC 6655. Juuli 2012. DOI: 10.17487/RFC6655. URL: https://www.rfc-editor.org/info/rfc6655.

[93] Yoel Gluck, Neal Harris ja Angelo Prado. *BREACH: reviving the CRIME attack*. 2013. URL: http://breachattack.com/resources/BREACH%20-%20SSL,%20gone%20in%2030%20seconds.pdf.

[94] Y. Sheffer, P. Saint-Andre ja T. Fossati. *Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)*. RFC 9325. Nov 2022. URL: https://rfc-editor.org/rfc/rfc9325.txt.

[95] Kerry A. McKay ja David A. Cooper. *Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations*. NIST Special Publication 800-52 Revision 2. Gaithersburg, MD: National Institute of Standards ja Technology, aug 2019. DOI: 10.6028/NIST.SP.800-52r2. URL: https://csrc.nist.gov/pubs/sp/800/52/r2/final.

[96] *Cryptographic Mechanisms: Recommendations and Key Lengths – Part 2 – Use of Transport Layer Security (TLS)*. BSI – Technical Guideline TR-02102-2, https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-2.pdf?__blob=publicationFile&v=11. Jaan 2026.

[97] Nationaal Cyber Security Centrum (NL). *Transport Layer Security (TLS). Security guidelines version 2025-05*. Mai 2025. URL: https://www.ncsc.nl/en/transport-layer-security-tls/tls-security-guidelines.

[98] *Eliminating Obsolete Transport Layer Security (TLS) Protocol Configurations*. Jaan 2021. URL: https://media.defense.gov/2021/Jan/05/2002560140/-1/-1/0/ELIMINATING_OBSOLETE_TLS_UOO197443-20.PDF.

[99] National Institute of Standards and Technology (NIST). *CVE-2001-1473 Detail*. https://nvd.nist.gov/vuln/detail/CVE-2001-1473. Accessed: 2026-01-23.

[100] Chris M. Lonvick ja Tatu Ylonen. *The Secure Shell (SSH) Transport Layer Protocol*. RFC 4253. Jaan 2006. URL: https://rfc-editor.org/rfc/rfc4253.txt.

[101] Chris M. Lonvick ja Tatu Ylonen. *The Secure Shell (SSH) Authentication Protocol*. RFC 4252. Jaan 2006. DOI: 10.17487/RFC4252. URL: https://www.rfc-editor.org/info/rfc4252.

[102] Chris M. Lonvick ja Tatu Ylonen. *The Secure Shell (SSH) Connection Protocol*. RFC 4254. Jaan 2006. DOI: 10.17487/RFC4254. URL: https://www.rfc-editor.org/info/rfc4254.

[103] National Institute of Standards and Technology (NIST). *CVE-2025-26465 Detail*. https://nvd.nist.gov/vuln/detail/CVE-2025-26465. Accessed: 2025-12-02.

[104] National Institute of Standards and Technology (NIST). *CVE-2025-26466 Detail*. https://nvd.nist.gov/vuln/detail/CVE-2025-26466. Accessed: 2025-12-02.

[105] National Institute of Standards and Technology (NIST). *CVE-2024-6387 Detail*. https://nvd.nist.gov/vuln/detail/CVE-2024-6387. Accessed: 2025-12-02.

[106] National Institute of Standards and Technology (NIST). *CVE-2023-48795 Detail*. https://nvd.nist.gov/vuln/detail/CVE-2023-48795. Accessed: 2025-12-02.

[107] National Institute of Standards and Technology (NIST). *CVE-2023-38408 Detail*. https://nvd.nist.gov/vuln/detail/CVE-2023-38408. Accessed: 2025-12-02.

[108] OpenSSH Project. *OpenSSH: Release Notes*. https://www.openssh.org/releasenotes.html. Accessed: 2025-12-02.

[109] *Cryptographic Mechanisms: Recommendations and Key Lengths – Use of Secure Shell (SSH)*. BSI – Technical Guideline TR-02102-4, https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-4.pdf. Jaan 2023.

[110] Karen Seo ja Stephen Kent. *Security Architecture for the Internet Protocol*. RFC 4301. Dets 2005. URL: https://rfc-editor.org/rfc/rfc4301.txt.

[111] *Cryptographic Mechanisms: Recommendations and Key Lengths – Part 3 – Use of Internet Protocol Security (IPsec) and Internet Key Exchange (IKEv2)*. BSI – Technical Guideline TR-02102-3, https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-3.pdf?__blob=publicationFile&v=10. Jaan 2026.

[112] Elaine Barker, Quynh Dang ja Sheila Frankel. *Guide to IPsec VPNs*. NIST Special Publication 800-77, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-77r1.pdf. Juuni 2020.

[113] Charlie Kaufman *et al. Internet Key Exchange Protocol Version 2 (IKEv2)*. RFC 7296. Okt 2014. URL: https://rfc-editor.org/rfc/rfc7296.txt.

[114] Stephen Kent. *IP Authentication Header*. RFC 4302. Dets 2005. URL: https://rfc-editor.org/rfc/rfc4302.txt.

[115] Stephen Kent. *IP Encapsulating Security Payload (ESP)*. RFC 4303. Dets 2005. URL: https://rfc-editor.org/rfc/rfc4303.txt.

[116] Kai Tian *et al.* „Analysis of Vulnerability of IPsec Protocol Implementation Based on Differential Fuzzing“. Teoses: *Security and Privacy in New Computing Environments*. Toim. Wenbo Shi, Xiaofeng Chen ja Kim-Kwang Raymond Choo. Kd 423. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer International Publishing, 2022, lk-d 282–293. ISBN: 978-3-030-96791-8.

[117] Jiaxing Guo *et al.* „Automated State-Machine-Based Analysis of Hostname Verification in IPsec Implementations“. *Inf. Technol. Control.* 50.3 (2021), lk-d 570–587. DOI: 10.5755/j01.itc.50.3.27844.

[118] Jiaxing Guo *et al.* „Stateful black-box fuzzing for encryption protocols and its application in IPsec“. *Comput. Networks* 251 (2024), lk 110605. DOI: 10.1016/J.COMNET.2024.110605. URL: https://doi.org/10.1016/j.comnet.2024.110605.

[119] Dennis Felsch *et al.* „The Dangers of Key Reuse: Practical Attacks on IPsec IKE“. Teoses: *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*. Toim. William Enck ja Adrienne Porter Felt. USENIX Association, 2018, lk-d 567–583. URL: https://www.usenix.org/conference/usenixsecurity18/presentation/felsch.

[120] James Ball, Julian Borger ja Glenn Greenwald. *Revealed: how US and UK spy agencies defeat internet privacy and security*. The Guardian, https://www.theguardian.com/world/2013/sep/05/nsa-gchq-encryption-codes-security. 2013.

[121] Jason A. Donenfeld. „WireGuard: Next Generation Kernel Network Tunnel“. Teoses: *24th Annual Network and Distributed System Security Symposium, NDSS 2017, San Diego, California, USA, February 26 - March 1, 2017*. The Internet Society, 2017. URL: https://www.ndss-symposium.org/ndss2017/ndss-2017-programme/wireguard-next-generation-kernel-network-tunnel/.

[122] Benjamin Dowling ja Kenneth G. Paterson. „A Cryptographic Analysis of the WireGuard Protocol“. Teoses: *Applied Cryptography and Network Security - 16th International Conference, ACNS 2018, Leuven, Belgium, July 2-4, 2018, Proceedings*. Toim. Bart Preneel ja Frederik Vercauteren. Kd 10892. Lecture Notes in Computer Science. Springer, 2018, lk-d 3–21. DOI: 10.1007/978-3-319-93387-0_1. URL: https://doi.org/10.1007/978-3-319-93387-0%5C_1.

[123] Benjamin Lipp, Bruno Blanchet ja Karthikeyan Bhargavan. „A Mechanised Cryptographic Proof of the WireGuard Virtual Private Network Protocol“. Teoses: *IEEE European Symposium on Security and Privacy, EuroS&P 2019, Stockholm, Sweden, June 17-19, 2019*. IEEE, 2019, lk-d 231–246. DOI: 10.1109/EUROSP.2019.00026. URL: https://doi.org/10.1109/EuroSP.2019.00026.

[124] Pascal Lafourcade, Dhekra Mahmoud ja Sylvain Ruhault. „A Unified Symbolic Analysis of WireGuard“. Teoses: *31st Annual Network and Distributed System Security Symposium, NDSS 2024, San Diego, California, USA, February 26 - March 1, 2024*. The Internet Society, 2024. URL: https://www.ndss-symposium.org/ndss-paper/a-unified-symbolic-analysis-of-wireguard/.

[125] *Analysis of the Possibility to Use ID1 Card's NFC Interface for Authentication and Electronic Signing*. https://www.ria.ee/sites/default/files/content-editors/SF/analysis_of_the_possibility_to_use_id1_cards_nfc_interface_for_authentication_and_electronic_signing_d-26-7_1.1_0.pdf. Cybernetica report no. D-26-7. 2022.

[126] Ernst Haselsteiner ja Klemens Breitfuß. „Security in near field communication (NFC)“. Teoses: *Workshop on RFID security*. Kd 517. 517. sn. 2006, lk 517.

[127] Michael Roland, Josef Langer ja Josef Scharinger. „Applying relay attacks to Google Wallet“. Teoses: *2013 5th International Workshop on Near Field Communication (NFC)*. IEEE. 2013, lk-d 1–6.

[128] Arup Barua *et al.* „Security and privacy threats for bluetooth low energy in iot and wearable devices: A comprehensive survey“. *IEEE Open Journal of the Communications Society* 3 (2022), lk-d 251–281.

[129] Albert Levi *et al.* „Relay attacks on bluetooth authentication and solutions“. Teoses: *International symposium on computer and information sciences*. Springer. 2004, lk-d 278–288.

[130] Simon Josefsson. *The Base16, Base32, and Base64 Data Encodings*. RFC 4648. Okt 2006. DOI: 10.17487/RFC4648. URL: https://www.rfc-editor.org/info/rfc4648.

[131] Roy T. Fielding, Mark Nottingham ja Julian Reschke. *HTTP Semantics*. RFC 9110. Juuni 2022. DOI: 10.17487/RFC9110. URL: https://www.rfc-editor.org/info/rfc9110.

[132] ITU-T. *Information Technology – ASN.1 Encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*. Recommendation X.690. International Telecommunication Union, veebr 2021. URL: https://www.itu.int/rec/T-REC-X.690.

[133] Carsten Bormann ja Paul E. Hoffman. *Concise Binary Object Representation (CBOR)*. RFC 8949. Dets 2020. DOI: 10.17487/RFC8949. URL: https://www.rfc-editor.org/info/rfc8949.

[134] Tim Bray. *The JavaScript Object Notation (JSON) Data Interchange Format*. RFC 8259. Dets 2017. DOI: 10.17487/RFC8259. URL: https://www.rfc-editor.org/info/rfc8259.

[135] Tim Bray *et al. Extensible Markup Language (XML) 1.0 (Fifth Edition)*. W3C Recommendation. World Wide Web Consortium (W3C), nov 2008. URL: https://www.w3.org/TR/xml/.

[136] Gregg Kellogg, Pierre-Antoine Champin ja Dave Longley. *JSON-LD 1.1: A JSON-based Serialization for Linked Data*. W3C Recommendation. World Wide Web Consortium (W3C), juuli 2020. URL: https://www.w3.org/TR/json-ld11/.

[137] „Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC“. *OJ* L 257 (aug 2014), lk-d 73–114. URL: http://data.europa.eu/eli/reg/2014/910/oj.

[138] Riigikogu. *E-identimise ja e-tehingute usaldusteenuste seadus*. Riigi Teataja I, 30.12.2024, 6. Dets 2024. URL: https://www.riigiteataja.ee/akt/130122024006 (vaadatud 03.05.2026).

[139] *ETSI EN 319 102-1 V1.4.1 Electronic Signatures and Trust Infrastructures (ESI); Procedures for Creation and Validation of AdES Digital Signatures; Part 1: Creation and Validation*. URL: https://www.etsi.org/standards/#page=1&search=319%20102-1&content=0.

[140] Russ Housley. *Update to the Cryptographic Message Syntax (CMS) for Algorithm Identifier Protection*. RFC 8933. Okt 2020. URL: https://rfc-editor.org/rfc/rfc8933.txt.

[141] *ETSI TS 101 733. Electronic Signatures and Infrastructures (ESI); CMS Advanced Electronic Signatures (CAdES)*. URL: https://www.etsi.org/standards/#page=1&search=101%20733&content=0.

[142] *ETSI TS 102 778-1. Electronic Signatures and Infrastructures (ESI); PDF Advanced Electronic Signature Profiles; Part 1: PAdES Overview - a framework document for PAdES*. URL: https://www.etsi.org/standards/#page=1&search=102%20778-1&content=0.

[143] Russ Housley. *Cryptographic Message Syntax (CMS)*. RFC 5652. Sept 2009. URL: https://rfc-editor.org/rfc/rfc5652.txt.

[144] Daniel Van Geest ja Falko Strenzke. *Best Practices for Signed Attributes in CMS SignedData*. Internet-Draft draft-ietf-lamps-cms-euf-cma-signeddata-02. Work in Progress. Internet Engineering Task Force, veebr 2026. 15 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-lamps-cms-euf-cma-signeddata/02/.

[145] Richard Barnes. *Use Cases and Requirements for JSON Object Signing and Encryption (JOSE)*. RFC 7165. Apr 2014. URL: https://rfc-editor.org/rfc/rfc7165.txt.

[146] Michael Jones, John Bradley ja Nat Sakimura. *JSON Web Signature (JWS)*. RFC 7515. Mai 2015. URL: https://rfc-editor.org/rfc/rfc7515.txt.

[147] *ETSI EN 119 182-1. Electronic Signatures and Infrastructures (ESI); JAdES digital signatures; Part 1: Building blocks and JAdES baseline signatures*. URL: https://www.etsi.org/standards/#page=1&search=119%20182-1&content=0.

[148] Juan-Carlos Cruellas Ibarz. „Bringing JSON signatures to ETSI AdES framework: Meet JAdES signatures“. *Computer Standards & Interfaces* 71 (2020), lk 103434. ISSN: 0920-5489. DOI: https://doi.org/10.1016/j.csi.2020.103434.

[149] Michael Jones ja Joe Hildebrand. *JSON Web Encryption (JWE)*. RFC 7516. Mai 2015. URL: https://rfc-editor.org/rfc/rfc7516.txt.

[150] Michael Jones, John Bradley ja Nat Sakimura. *JSON Web Token (JWT)*. RFC 7519. Mai 2015. URL: https://rfc-editor.org/rfc/rfc7519.txt.

[151] N. Sakimura *et al. OpenID Connect Core 1.0 incorporating errata set 1*. https://openid.net/specs/openid-connect-core-1_0.html. Nov 2014.

[152] Michael B. Jones. *JSON Web Key (JWK)*. RFC 7517. Mai 2015. URL: https://rfc-editor.org/rfc/rfc7517.txt.

[153] Jim Schaad. *CBOR Object Signing and Encryption (COSE): Structures and Process*. RFC 9052. Aug 2022. DOI: 10.17487/RFC9052. URL: https://www.rfc-editor.org/info/rfc9052.

[154] Michael B. Jones *et al. CBOR Web Token (CWT)*. RFC 8392. Mai 2018. DOI: 10.17487/RFC8392. URL: https://www.rfc-editor.org/info/rfc8392.

[155] *ETSI TS 119 152-1. Electronic Signatures and Trust Infrastructures (ESI); CB AdES (CBOR-AdES) digital signatures; Part 1: Building blocks and CB-AdES baseline signatures*. URL: https://www.etsi.org/standards/#page=1&search=119%20152-1&content=0.

[156] *Electronic Signatures and Trust Infrastructures (ESI); Profiles for Electronic Attestation of Attributes; Part 1: General requirements*. Versioon 1.1.1. Dets 2025. URL: https://www.etsi.org/deliver/etsi_ts/119400_119499/11947201/01.01.01_60/ts_11947201v010101p.pdf.

[157] Donald Eastlake III *et al. XML Encryption Syntax and Processing Version 1.1*. W3C Recommendation. World Wide Web Consortium (W3C), apr 2013. URL: https://www.w3.org/TR/xmlenc-core1/.

[158] Donald Eastlake III *et al. XML Signature Syntax and Processing Version 1.1*. W3C Recommendation. World Wide Web Consortium (W3C), apr 2013. URL: https://www.w3.org/TR/xmldsig-core1/.

[159] *ETSI EN 319 132-1. Electronic Signatures and Infrastructures (ESI); XAdES digital signatures; Part 1: Building blocks and XAdES baseline signatures*. URL: https://www.etsi.org/standards/#page=1&search=319%20132-1&content=0.

[160] *ETSI EN 319 162-1. Electronic Signatures and Infrastructures (ESI); Associated Signature Containers (ASiC); Part 1: Building blocks and ASiC baseline containers*. URL: https://www.etsi.org/standards/#page=1&search=319%20162-1&content=0.

[161] S. Santesson ja N. Pope. *ESSCertIDv2 Update for RFC 3161*. RFC 5816. IETF, apr 2010. URL: http://tools.ietf.org/rfc/rfc5816.txt.

[162] T. Gondrom, R. Brandner ja U. Pordesch. *Evidence Record Syntax (ERS)*. RFC 4998. Aug 2007. URL: https://rfc-editor.org/rfc/rfc4998.txt.

[163] A. Jerman Blazic, S. Saljic ja T. Gondrom. *Extensible Markup Language Evidence Record Syntax (XMLERS)*. RFC 6283. Juuli 2011. URL: https://rfc-editor.org/rfc/rfc6283.txt.

[164] *Electronic Signatures and Infrastructures (ESI); XAdES Basline Profile*. Märts 2012. URL: https://www.etsi.org/deliver/etsi_ts/103100_103199/103171/02.01.01_60/ts_103171v020101p.pdf.

[165] *Electronic Signatures and Infrastructures (ESI); ASiC Baseline Profile*. Märts 2012. URL: https://www.etsi.org/deliver/etsi_ts/103100_103199/103174/02.01.01_60/ts_103174v020101p.pdf.

[166] *BDOC – digiallkirja vorming*. https://www.id.ee/wp-content/uploads/2020/01/bdoc-spec212-est.pdf. Versioon 2.1.2. OID: 1.3.6.1.4.1.10015.1000.3.2.3. 2014.

[167] *ETSI EN 319 122-1. Electronic Signatures and Infrastructures (ESI); CAdES digital signatures; Part 1: Building blocks and CAdES baseline signatures*. URL: https://www.etsi.org/standards/#page=1&search=319%20122-1&content=0.

[168] Alisa Pankova *et al. Tõend kui platvorm*. Dok. D-25-2, Cybernetica AS, Riigi Infosüsteemi Ameti tellitud analüüs, https://www.ria.ee/amet-uudised-ja-kontakt/uudised-pressikontakt/uuringud-ja-analuusid. Mai 2022.

[169] Daniel Fett, Kristina Yasuda ja Brian Campbell. *Selective Disclosure for JSON Web Tokens*. RFC 9901. Nov 2025. DOI: 10.17487/RFC9901. URL: https://www.rfc-editor.org/info/rfc9901.

[170] Oliver Terbu, Daniel Fett ja Brian Campbell. *SD-JWT-based Verifiable Digital Credentials (SD-JWT VC)*. Internet-Draft draft-ietf-oauth-sd-jwt-vc-15. Work in Progress. Internet Engineering Task Force, veebr 2026. 65 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-oauth-sd-jwt-vc/15/.

[171] Tobias Looker, Paul Bastian ja Christian Bormann. *Token Status List (TSL)*. Internet-Draft draft-ietf-oauth-status-list-19. Work in Progress. Internet Engineering Task Force, märts 2026. 78 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-oauth-status-list/19/.

[172] Takeshi Imamura *et al. XML Encryption Syntax and Processing Version 1.1*. https://www.w3.org/TR/2013/REC-xmlenc-core1-20130411/. Apr 2013.

[173] Veiko Sinivee ja Kristi Uukkivi. *Encrypted Digidoc Format Specification*. Versioon 1.1. AS Sertifitseerimiskeskus. Juuni 2012. URL: https://www.id.ee/wp-content/uploads/2020/06/sk-cdoc-1.0-20120625_en.pdf.

[174] Don Davis. „Defective Sign & Encrypt in S/MIME, PKCS#7, MOSS, PEM, PGP, and XML“. Teoses: *Proceedings of the General Track: 2001 USENIX Annual Technical Conference, June 25-30, 2001, Boston, Massachusetts, USA*. Toim. Yoonho Park. USENIX, 2001, lk-d 65–78. URL: http://www.usenix.org/publications/library/proceedings/usenix01/davis.html.

[175] Hugo Krawczyk. „The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?)“ Teoses: *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*. Toim. Joe Kilian. Kd 2139. Lecture Notes in Computer Science. Springer, 2001, lk-d 310–331.

[176] Ross J. Anderson ja Roger M. Needham. „Robustness Principles for Public Key Protocols“. Teoses: *Advances in Cryptology - CRYPTO '95, 15th Annual International Cryptology Conference, Santa Barbara, California, USA, August 27-31, 1995, Proceedings*. Toim. Don Coppersmith. Kd 963. Lecture Notes in Computer Science. Springer, 1995, lk-d 236–247.

[177] Mart Oruaas ja Jan Willemson. „Developing Requirements for the New Encryption Mechanisms in the Estonian eID Infrastructure“. Teoses: *Databases and Information Systems - 14th International Baltic Conference, DB&IS 2020, Tallinn, Estonia, June 16-19, 2020, Proceedings*. Toim. Tarmo Robal *et al.* Kd 1243. Communications in Computer and Information Science. Springer, 2020, lk-d 13–20.

[178] The IEEE and The Open Group. *Portable archive interchange*. https://pubs.opengroup.org/onlinepubs/9699919799/utilities/pax.html. 2018.

[179] L. Peter Deutsch ja Jean-loup Gailly. *ZLIB Compressed Data Format Specification version 3.3*. RFC 1950. Mai 1996. URL: https://rfc-editor.org/rfc/rfc1950.txt.

[180] Peter Saint-Andre ja Rich Salz. *Service Identity in TLS*. RFC 9525. Nov 2023. DOI: 10.17487/RFC9525. URL: https://www.rfc-editor.org/info/rfc9525.

[181] Christiaan Brand *et al. Web Authentication: An API for accessing Public Key Credentials*. Level 1 https://www.w3.org/TR/webauthn-1/. Märts 2019.

[182] John Bradley *et al. Web Authentication: An API for accessing Public Key Credentials Level 2*. Level 2 https://www.w3.org/TR/webauthn-2/. Apr 2021.

[183] John Bradley *et al. Web Authentication: An API for accessing Public Key Credentials Level 3*. W3C Candidate Recommendation Snapshot https://www.w3.org/TR/webauthn-3/. Jaan 2026.

[184] *Web eID: electronic identity cards on the Web*. https://github.com/web-eid/web-eid-system-architecture-doc/. 2022.

[185] *Analysis of planned architectural changes in Open-eID*. https://web-eid.github.io/web-eid-cybernetica-analysis/webextensions-main.pdf. Cybernetica report no. T-176-1. 2020.

[186] Dick Hardt. *The OAuth 2.0 Authorization Framework*. RFC 6749. Okt 2012. URL: https://rfc-editor.org/rfc/rfc6749.txt.

[187] William Denniss ja John Bradley. *OAuth 2.0 for Native Apps*. RFC 8252. Okt 2017. URL: https://rfc-editor.org/rfc/rfc8252.txt.

[188] Torsten Lodderstedt, Mark McGloin ja Phil Hunt. *OAuth 2.0 Threat Model and Security Considerations*. RFC 6819. Jaan 2013. URL: https://rfc-editor.org/rfc/rfc6819.txt.

[189] Torsten Lodderstedt *et al. Best Current Practice for OAuth 2.0 Security*. RFC 9700. Jaan 2025. DOI: 10.17487/RFC9700. URL: https://www.rfc-editor.org/info/rfc9700.

[190] A. Parecki, P. De Ryck ja D. Waite. *OAuth 2.0 for Browser-Based Applications*. RFC 10017. Aug 2026. DOI: 10.17487/RFC10017. URL: https://www.rfc-editor.org/info/rfc10017/.

[191] Michael Jones ja Dick Hardt. *The OAuth 2.0 Authorization Framework: Bearer Token Usage*. RFC 6750. Okt 2012. URL: https://rfc-editor.org/rfc/rfc6750.txt.

[192] Justin Richer *et al. OAuth 2.0 Dynamic Client Registration Protocol*. RFC 7591. Juuli 2015. URL: https://rfc-editor.org/rfc/rfc7591.txt.

[193] Justin Richer *et al. OAuth 2.0 Dynamic Client Registration Management Protocol*. RFC 7592. Juuli 2015. URL: https://rfc-editor.org/rfc/rfc7592.txt.

[194] Justin Richer. *OAuth 2.0 Token Introspection*. RFC 7662. Okt 2015. URL: https://rfc-editor.org/rfc/rfc7662.txt.

[195] Daniel Fett *et al. OAuth 2.0 Demonstrating Proof of Possession (DPoP)*. RFC 9449. Sept 2023. DOI: 10.17487/RFC9449. URL: https://www.rfc-editor.org/info/rfc9449.

[196] *Digiidentiteedikukru analüüside kogumik*. https://www.ria.ee/sites/default/files/documents/2026-05/Digikukru-III-etapi-analuus.pdf. Cybernetica report no. D-25-53 (in Estonian). 2026.

[197] Kai Mindermann, Philipp Keck ja Stefan Wagner. „How Usable Are Rust Cryptography APIs?“ Teoses: *2018 IEEE International Conference on Software Quality, Reliability and Security, QRS 2018, Lisbon, Portugal, July 16-20, 2018*. IEEE, 2018, lk-d 143–154.

[198] Madelyn Cain *et al.* „Shor's algorithm is possible with as few as 10,000 reconfigurable atomic qubits“. *arXiv* abs/2603.28627 (2026). DOI: 10.48550/ARXIV.2603.28627. arXiv: 2603.28627. URL: https://doi.org/10.48550/arXiv.2603.28627.

[199] D. Stebila, S. Fluhrer ja S. Gueron. *Hybrid Key Exchange in TLS 1.3*. RFC 9954. Juuli 2026. DOI: 10.17487/RFC9954. URL: https://www.rfc-editor.org/info/rfc9954/.

[200] K. Kwiatkowski *et al. Post-Quantum Traditional (PQ/T) Hybrid Key Agreement Mechanisms for TLS 1.3*. RFC 10024. Aug 2026. DOI: 10.17487/RFC10024. URL: https://www.rfc-editor.org/info/rfc10024/.

[201] Tirumaleswar Reddy.K *et al. Use of Composite ML-DSA in TLS 1.3*. Internet-Draft draft-reddy-tls-composite-mldsa-09. Work in Progress. Internet Engineering Task Force, veebr 2026. 13 lk-d. URL: https://datatracker.ietf.org/doc/draft-reddy-tls-composite-mldsa/09/.

[202] Thom Wiggers *et al. KEM-based Authentication for TLS 1.3*. Internet-Draft draft-celi-wiggers-tls-authkem-07. Work in Progress. Internet Engineering Task Force, mai 2026. 26 lk-d. URL: https://datatracker.ietf.org/doc/draft-celi-wiggers-tls-authkem/07/.

[203] Tim Hollebeek, Sophie Schmieg ja Bas Westerbaan. *Use of ML-DSA in TLS 1.3*. Internet-Draft draft-ietf-tls-mldsa-03. Work in Progress. Internet Engineering Task Force, mai 2026. 6 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-tls-mldsa/03/.

[204] Tirumaleswar Reddy.K *et al. Use of SLH-DSA in TLS 1.3*. Internet-Draft draft-reddy-tls-slhdsa-02. Work in Progress. Internet Engineering Task Force, nov 2025. 11 lk-d. URL: https://datatracker.ietf.org/doc/draft-reddy-tls-slhdsa/02/.

[205] Markus Friedl, Jan Mojzis ja Simon Josefsson. *Secure Shell (SSH) Key Exchange Method Using Hybrid Streamlined NTRU Prime sntrup761 and X25519 with SHA-512: sntrup761x25519-sha512*. RFC 9941. Apr 2026. DOI: 10.17487/RFC9941. URL: https://www.rfc-editor.org/info/rfc9941.

[206] Panos Kampanakis, Douglas Stebila ja Torben Hansen. *PQ/T Hybrid Key Exchange with ML-KEM in SSH*. Internet-Draft draft-ietf-sshm-mlkem-hybrid-kex-10. Work in Progress. Internet Engineering Task Force, veebr 2026. 15 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-sshm-mlkem-hybrid-kex/10/.

[207] Tero Kivinen ja Joel Snyder. *Signature Authentication in the Internet Key Exchange Version 2 (IKEv2)*. RFC 7427. Jaan 2015. DOI: 10.17487/RFC7427. URL: https://www.rfc-editor.org/info/rfc7427.

[208] C. Tjhai *et al. Multiple Key Exchanges in the Internet Key Exchange Protocol Version 2 (IKEv2)*. RFC 9370. Mai 2023. DOI: 10.17487/RFC9370. URL: https://www.rfc-editor.org/info/rfc9370.

[209] Panos Kampanakis. *Post-quantum Key Exchange with ML-KEM in the Internet Key Exchange Protocol Version 2 (IKEv2)*. Internet-Draft draft-ietf-ipsecme-ikev2-mlkem-05. Work in Progress. Internet Engineering Task Force, märts 2026. 12 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-ipsecme-ikev2-mlkem/05/.

[210] Tirumaleswar Reddy.K, Valery Smyslov ja Scott Fluhrer. *Signature Authentication in the Internet Key Exchange Version 2 (IKEv2) using PQC*. Internet-Draft draft-ietf-ipsecme-ikev2-pqc-auth-08. Work in Progress. Internet Engineering Task Force, apr 2026. 19 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-ipsecme-ikev2-pqc-auth/08/.

[211] Dr. Clifford Neuman *et al. The Kerberos Network Authentication Service (V5)*. RFC 4120. Juuli 2005. DOI: 10.17487/RFC4120. URL: https://www.rfc-editor.org/info/rfc4120.

[212] Brian Tung ja Larry Zhu. *Public Key Cryptography for Initial Authentication in Kerberos (PKINIT)*. RFC 4556. Juuni 2006. DOI: 10.17487/RFC4556. URL: https://www.rfc-editor.org/info/rfc4556.

[213] Andreas Hülsing *et al.* „Post-quantum WireGuard“. Teoses: *2021 IEEE Symposium on Security and Privacy (SP)*. 2021, lk-d 304–321. DOI: 10.1109/SP40001.2021.00030.

[214] Frederico Schardong *et al.* „Post-Quantum Electronic Identity: Adapting OpenID Connect and OAuth 2.0 to the Post-Quantum Era“. Teoses: *Cryptology and Network Security*. International Conference on Cryptology and Network Security. Springer International Publishing. Abu Dhabi, United Arab Emirates, 2022, lk-d 371–390. DOI: 10.1007/978-3-031-20974-1_20.

[215] FIDO Alliance. *Addressing FIDO Alliance's Technologies in Post Quantum World*. https://fidoalliance.org/wp-content/uploads/2024/02/FIDO-BTC-White-Paper_FIDO-Alliance-and-Post-Quantum-Cryptography.pdf. 2024.

[216] M. Prorock ja O. Steele. *ML-DSA for JSON Object Signing and Encryption (JOSE) and CBOR Object Signing and Encryption (COSE)*. RFC 9964. Mai 2026. DOI: 10.17487/RFC9964. URL: https://www.rfc-editor.org/info/rfc9964/.

[217] Aditya Mitra *et al. ML-DSA for Web Authentication*. Internet-Draft draft-vitap-ml-dsa-webauthn-04. Work in Progress. Internet Engineering Task Force, märts 2026. 16 lk-d. URL: https://datatracker.ietf.org/doc/draft-vitap-ml-dsa-webauthn/04/.

[218] Nina Bindel, Cas Cremers ja Mang Zhao. „ FIDO2, CTAP 2.1, and WebAuthn 2: Provable Security and Post-Quantum Instantiation “. Teoses: *2023 IEEE Symposium on Security and Privacy (SP)*. Los Alamitos, CA, USA: IEEE Computer Society, mai 2023, lk-d 1471–1490. DOI: 10.1109/SP46215.2023.10179454. URL: https://doi.ieeecomputersociety.org/10.1109/SP46215.2023.10179454.

[219] Tirumaleswar Reddy.K. *Post-Quantum Enhancements to TLS-Based EAP Methods*. Internet-Draft draft-reddy-emu-pqc-eap-tls-03. Work in Progress. Internet Engineering Task Force, apr 2026. 10 lk-d. URL: https://datatracker.ietf.org/doc/draft-reddy-emu-pqc-eap-tls/03/.

[220] Tirumaleswar Reddy.K ja Aritra Banerjee. *Post-Quantum Key Encapsulation Mechanisms (PQ KEMs) in EAP-AKA prime*. Internet-Draft draft-ietf-emu-pqc-eapaka-02. Work in Progress. Internet Engineering Task Force, märts 2026. 21 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-emu-pqc-eapaka/02/.

[221] Aritra Banerjee ja Tirumaleswar Reddy.K. *Enhancing Security in EAP-AKA' with Hybrid Post-Quantum Cryptography*. Internet-Draft draft-ietf-emu-hybrid-pqc-eapaka-01. Work in Progress. Internet Engineering Task Force, veebr 2026. 13 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-emu-hybrid-pqc-eapaka/01/.

[222] J.S. Buruaga, A. Bugler ja J.P. et al. Brito. „Versatile quantum-safe hybrid key exchange and its application to MACsec“. *EPJ Quantum Technol.* 12.84 (2025).

[223] Tao Liu, Gowri Sankar Ramachandran ja Raja Jurdak. „Towards Quantum Resilient IoT: A Backward-Compatible Approach to Secure BLE Key Exchange Against Quantum Threats“. Teoses: *Ninth IEEE/ACM International Conference on Internet-of-Things Design and Implementation, IoTDI 2024, Hong Kong, May 13-16, 2024*. IEEE, 2024, lk-d 170–180. DOI: 10.1109/IOTDI61053.2024.00019. URL: https://doi.org/10.1109/IoTDI61053.2024.00019.

[224] Jessica Bozhko *et al.* „Performance Evaluation of Quantum-Resistant TLS for Consumer IoT Devices“. Teoses: *20th IEEE Consumer Communications & Networking Conference, CCNC 2023, Las Vegas, NV, USA, January 8-11, 2023*. IEEE, 2023, lk-d 230–235. DOI: 10.1109/CCNC51644.2023.10060762. URL: https://doi.org/10.1109/CCNC51644.2023.10060762.

[225] Diana Ghinea *et al.* „Hybrid Post-quantum Signatures in Hardware Security Keys“. Teoses: *Applied Cryptography and Network Security Workshops - ACNS 2023 Satellite Workshops, ADSC, AIBlock, AIHWS, AIoTS, CIMSS, Cloud S&P, SCI, SecMT, SiMLA, Kyoto, Japan, June 19-22, 2023, Proceedings*. Toim. Jianying Zhou *et al.* Kd 13907. Lecture Notes in Computer Science. Springer, 2023, lk-d 480–499. DOI: 10.1007/978-3-031-41181-6_26. URL: https://doi.org/10.1007/978-3-031-41181-6_26.

[226] Corey Bonnell *et al. A Mechanism for Encoding Differences in Paired Certificates*. Internet-Draft draft-bonnell-lamps-chameleon-certs-07. Work in Progress. Internet Engineering Task Force, okt 2025. 52 lk-d. URL: https://datatracker.ietf.org/doc/draft-bonnell-lamps-chameleon-certs/07/.

[227] Panos Kampanakis *et al. The Viability of Post-quantum X.509 Certificates*. Cryptology ePrint Archive, Paper 2018/063. 2018. URL: https://eprint.iacr.org/2018/063.

[228] David McGrew, Michael Curcio ja Scott Fluhrer. *Leighton-Micali Hash-Based Signatures*. RFC 8554. Apr 2019. DOI: 10.17487/RFC8554. URL: https://www.rfc-editor.org/info/rfc8554.

[229] Matthew Chou ja Phuong Cao. „Network Impact of Post-Quantum Certificate Chain sizes on Time to First Byte in TLS Deployments“. *CoRR* abs/2604.24869 (2026). DOI: 10.48550/ARXIV.2604.24869. arXiv: 2604.24869. URL: https://doi.org/10.48550/arXiv.2604.24869.

[230] Daniel Van Geest *et al. Use of the HSS and XMSS Hash-Based Signature Algorithms in Internet X.509 Public Key Infrastructure*. RFC 9802. Juuni 2025. DOI: 10.17487/RFC9802. URL: https://www.rfc-editor.org/info/rfc9802.

[231] Jake Massimo *et al. Internet X.509 Public Key Infrastructure – Algorithm Identifiers for the Module-Lattice-Based Digital Signature Algorithm (ML-DSA)*. RFC 9881. Okt 2025. DOI: 10.17487/RFC9881. URL: https://www.rfc-editor.org/info/rfc9881.

[232] Sean Turner *et al. Internet X.509 Public Key Infrastructure - Algorithm Identifiers for the Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM)*. RFC 9935. Märts 2026. DOI: 10.17487/RFC9935. URL: https://www.rfc-editor.org/info/rfc9935.

[233] Kaveh Bashiri *et al. Internet X.509 Public Key Infrastructure – Algorithm Identifiers for the Stateless Hash-Based Digital Signature Algorithm (SLH-DSA)*. RFC 9909. Dets 2025. DOI: 10.17487/RFC9909. URL: https://www.rfc-editor.org/info/rfc9909.

[234] Jake Massimo *et al. Internet X.509 Public Key Infrastructure – Algorithm Identifiers for the Fast-Fourier Transform over NTRU-Lattice-Based Digital Signature Algorithm (FN-DSA)*. Internet-Draft draft-ietf-lamps-fn-dsa-certificates-00. Work in Progress. Internet Engineering Task Force, mai 2026. 17 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-lamps-fn-dsa-certificates/00/.

[235] Mike Ounsworth *et al. Composite Module-Lattice-Based Digital Signature Algorithm (ML-DSA) for use in X.509 Public Key Infrastructure*. Internet-Draft draft-ietf-lamps-pq-composite-sigs-19. Work in Progress. Internet Engineering Task Force, apr 2026. 234 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-lamps-pq-composite-sigs/19/.

[236] Mike Ounsworth *et al. Composite ML-KEM for use in X.509 Public Key Infrastructure*. Internet-Draft draft-ietf-lamps-pq-composite-kem-14. Work in Progress. Internet Engineering Task Force, märts 2026. 132 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-lamps-pq-composite-kem/14/.

[237] David Benjamin *et al. Merkle Tree Certificates*. Internet-Draft draft-ietf-plants-merkle-tree-certs-04. Work in Progress. Internet Engineering Task Force, mai 2026. 89 lk-d. URL: https://datatracker.ietf.org/doc/draft-ietf-plants-merkle-tree-certs/04/.

[238] Tim Wood *et al.* „Towards Post-Quantum Verifiable Credentials“. Teoses: *Proceedings of the 19th International Conference on Availability, Reliability and Security, ARES 2024, Vienna, Austria, 30 July 2024 - 2 August 2024*. ACM, 2024, 88:1–88:10. URL: https://doi.org/10.1145/3664476.3669932.

[239] Nouri Alnahawi *et al. Post-Quantum Cryptography in eMRTDs: Evaluating PAKE and PKI for Travel Documents*. Cryptology ePrint Archive, Paper 2025/812. 2025. URL: https://eprint.iacr.org/2025/812.

[240] International Civil Aviation Organization. *Doc 9303: Machine Readable Travel Documents*. https://www.icao.int/publications/doc-series/doc-9303. Accessed: 2026-06-03. 2026.

[241] International Civil Aviation Organization. *Quantum Safe Mechanisms for the Document Issuing PKI and Passive Authentication*. Working Paper TAG/TRIP/5-WP/18. Presented at the Fifth Meeting of the Technical Advisory Group on the Traveller Identification Programme (TAG/TRIP/5), 12–14 November 2025. Montréal, Canada: International Civil Aviation Organization (ICAO), nov 2025. URL: https://www.icao.int/sites/default/files/Meetings/TAG-TRIP/TAG-TRIP5/TAGTRIP.5.WP_.18.en_.FINALFULL.revised.pdf.

[242] Elaine Barker, Lily Chen ja Richard Davis. *Recommendation for Key-Derivation Methods in Key-Establishment Schemes*. NIST Special Publication 800-56C-2, https://doi.org/10.6028/NIST.SP.800-56Cr2. Aug 2020.

[243] Seongkwang Kim *et al. The AIMer Signature Scheme. Submission to the KpqC Competition*. 2024. URL: https://kpqc.or.kr/images/pdf/AIMer_Document.pdf.

[244] Jung Hee Cheon *et al. HAETAE: Shorter Lattice-Based Fiat-Shamir Signatures*. 2024. URL: https://aimer-signature.org.

[245] Jonghyun Kim ja Jong Hwan Park. *NTRU+. Algorithm Specifications And Supporting Documentation*. 2024. URL: https://kpqc.or.kr/images/pdf/NTRU+_Document.pdf.

[246] Jung Hee Cheon *et al. SMAUG-T: the Key Exchange Algorithm based on Module-LWE and Module-LWR*. 2024. URL: https://kpqc.or.kr/images/pdf/SMAUG-T_Document.pdf.

[247] European Commission. *Commission Notice on the 2025 Annual Union Work Programme for European Standardisation*. Official Journal of the European Union, C series, C/2025/1818. Accessed via EUR-Lex. 2025. URL: https://eur-lex.europa.eu/eli/C/2025/1818/oj/eng.

[248] *ETSI TS 103 744 (v1.2.2). CYBER; Quantum-Safe Cryptography (QSC); Quantum-safe Hybrid Key Establishment*. Veebr 2026. URL: https://www.etsi.org/deliver/etsi_ts/103700_103799/103744/01.02.02_60/ts_103744v010202p.pdf.

[249] *ETSI TS 104 015 (v1.1.1). Cyber Security (CYBER); Quantum-Safe Cryptography (QSC); Efficient Quantum-Safe Hybrid Key Exchanges with Hidden Access Policies*. Veebr 2025. URL: https://www.etsi.org/deliver/etsi_ts/104000_104099/104015/01.01.01_60/ts_104015v010101p.pdf.

[250] *ETSI TR 103 616 (v1.1.1). CYBER; Quantum-Safe Signatures*. Sept 2021. URL: https://www.etsi.org/deliver/etsi_tr/103600_103699/103616/01.01.01_60/tr_103616v010101p.pdf.

[251] *ETSI TR 103 823 (v1.1.1). CYBER; Quantum-Safe Public-Key Encryption and Key Encapsulation*. Sept 2021. URL: https://www.etsi.org/deliver/etsi_tr/103800_103899/103823/01.01.01_60/tr_103823v010101p.pdf.

[252] *ETSI TR 103 966 (v1.1.1). CYBER Security (CYBER); Quantum-Safe Cryptography (QSC); Deployment Considerations for Hybrid Schemes*. Okt 2024. URL: https://www.etsi.org/deliver/etsi_tr/103900_103999/103966/01.01.01_60/tr_103966v010101p.pdf.

[253] *ETSI TR 103 967 (v1.1.1). CYBER Security (CYBER); Quantum-Safe Cryptography (QSC); Impact of Quantum Computing on Symmetric Cryptography*. Jaan 2025. URL: https://www.etsi.org/deliver/etsi_tr/103900_103999/103967/01.01.01_60/tr_103967v010101p.pdf.

[254] *ETSI TR 104 005 (v1.1.1). Secure Element Technologies (SET); Technical Report on impacts of the post-quantum cryptography on ETSI TC SET specifications*. Juuli 2025. URL: https://www.etsi.org/deliver/etsi_tr/104000_104099/104005/01.01.01_60/tr_104005v010101p.pdf.

[255] *ETSI TR 103 619 (v1.1.1). CYBER; Migration strategies and recommendations to Quantum Safe schemes*. Juuli 2020. URL: https://www.etsi.org/deliver/etsi_tr/103600_103699/103619/01.01.01_60/tr_103619v010101p.pdf.

[256] CEN-CENELEC Focus Group on Quantum Technologies (FGQT). *Standardization Roadmap on Quantum Technologies (FGQT Q04, Release 1)*. Technical Report FGQT Q04. CEN ja CENELEC, märts 2023. URL: https://www.cencenelec.eu/media/CEN-CENELEC/AreasOfWork/CEN-CENELEC_Topics/Quantum%20technologies/Documentation%20and%20Materials/fgqt_q04_standardizationroadmapquantumtechnologies_release1.pdf.

[257] CEN-CENELEC Focus Group on Quantum Technologies (FGQT). *Quantum Technologies Use Cases (FGQT Q05, Release 1)*. Technical Report FGQT Q05. CEN ja CENELEC, märts 2023. URL: https://www.cencenelec.eu/media/CEN-CENELEC/AreasOfWork/CEN-CENELEC_Topics/Quantum%20technologies/Documentation%20and%20Materials/fgqt_q05_quantumtechnologiesusecases_release1.pdf.

[258] Andreas Huelsing *et al. XMSS: eXtended Merkle Signature Scheme*. RFC 8391. Mai 2018. DOI: 10.17487/RFC8391. URL: https://www.rfc-editor.org/info/rfc8391.

[259] C. Tjhai *et al. Multiple Key Exchanges in the Internet Key Exchange Protocol Version 2 (IKEv2)*. RFC 9370. Mai 2023. DOI: 10.17487/RFC9370. URL: https://www.rfc-editor.org/info/rfc9370.

[260] Russ Housley. *TLS 1.3 Extension for Certificate-Based Authentication with an External Pre-Shared Key*. RFC 8773. Märts 2020. DOI: 10.17487/RFC8773. URL: https://www.rfc-editor.org/info/rfc8773.

[261] W3C Technical Architecture Group and W3C Security Interest Group. *Cryptography Usage in Web Standards*. https://w3c.github.io/security-guidelines-cryptography. Editor's Draft. 2025.

[262] W3C Credentials Community Group. *Quantum-Safe Cryptosuites v0.3*. https://www.w3.org/di-quantum-safe/. Community Group Draft Report. 2025.

[263] Luigi Rizzo, Juan Carlos Cruellas ja Laurent Velez. *Technical Report of the LTA Signature Augmentation & Validation Plugtests Oct-Dec 2023*. Veebr 2024.

[264] „Commission Implementing Regulation (EU) 2025/1946 of 29 September 2025 laying down rules for the application of Regulation (EU) No 910/2014 of the European Parliament and of the Council as regards qualified preservation services for qualified electronic signatures and for qualified electronic seals“. *OJ* L (sept 2025). URL: http://data.europa.eu/eli/reg/2025/1946/oj.

[265] *ETSI TR 119 511. Electronic Signatures and Infrastructures (ESI); Policy and security requirements for trust service providers providing long-term preservation of digital signatures or general data using digital signature techniques*. URL: https://www.etsi.org/standards/#page=1&search=119%20511&content=0.

[266] *ETSI TR 119 172-4. Electronic Signatures and Trust Infrastructures (ESI); Signature Policies; Part 4: Signature applicability rules (validation policy) for European qualified electronic signatures/seals using trusted lists*. URL: https://www.etsi.org/standards/#page=1&search=119%20172-4&content=0.