Colofon

|  |  |
| --- | --- |
| Locatie | R:\Mijn Documenten\dictu\crypto\corona app\Covid-19 App Crypto Raamwerk 0.90.docx |
| Contactpersoon | Erik Holkers  E.H.Holkers@DICTU.nl  DICTU Security Centre (DSC)  Bezuidenhoutseweg 73 | 2594 AC Den Haag  Postbus 20401 | 2500 EK Den Haag |
| Auteurs | Dirk-Willem van Gulik  Erik Holkers  Ivo Jansch |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Inhoud

Colofon—2

Inhoud—3

1 Inleiding—5

1.1 Achtergrond—5

1.2 Doelstelling en uitgangspunten—5

1.3 Overzicht—7

1.4 Dreigingen, aanvallen en beschermende maatregelen op basis van cryptografie.—8

1.5 Definities—9

2 TEK life cycle managment—10

2.1 Key generation and derivation—10

2.2 Key Storage—11

2.3 Key Usage—12

2.3.1 Encryptie, generatie van RPI en AEM—12

2.3.2 Encryptie, generatie van potentially contacted RPIs—12

2.4 Key Backup—13

2.5 Key Archival—13

2.6 Key Destruction—14

2.7 Key Transport, upload van TEKS—15

2.7.1 Requirements—15

2.7.2 Functioneel flow—16

2.7.3 privacy preserving—17

2.7.4 Voorkom brute force uploads—17

2.7.5 Authenticiteit van de uploaded TEKs (alleen echte TEKs)—17

2.7.6 Gebruiksvriendelijk en veilig—17

2.7.7 Authenticiteit van de receiving server—18

2.8 Key Transport, downloads van TEKs—19

2.8.1 requirements—19

2.8.2 Authenticiteit van diagnose keys—19

2.8.3 Authenticiteit van Fake keys—20

2.8.4 Authenticiteit van diagnose keys Internationaal—21

2.8.5 Authenticiteit van download / Distribution Server / CDN—22

2.8.6 Verificatie op Client van de gedistribueerde bestanden met diagnosed keys—22

2.9 Transport, downloads van config files—23

2.9.1 Signing en verificatie van Config Files—23

2.10 Crypto Specs, algorithms, schema’s and protocols—24

Bijlage A Dreigingsoverzicht—25

Bijlage B Overwegingen rond upload van TEKs—26

Bijlage C Fake Keys—27

**Documentbeheer**

**Versiehistorie**

| Versie | Datum | Status | Auteur |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Versie goedkeuring**

| Versie | Datum | Naam | Functie/afdeling |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Wijzigingen**

| Versie | Wijzigingen |
| --- | --- |
|  |  |
|  |  |
|  |  |

**referenties**

| document | | | Auteur |
| --- | --- | --- | --- |
|  |  |  |  |
| Exposure Notification Cryptography Specification April 2020 v1.2 | | | GACT |
| SECURITY ANALYSIS OF THE COVID-19 CONTACT TRACING SPECIFICATIONS BY APPLE INC. AND GOOGLE INC | | | Yaron Gvili |
| Decentralized Privacy-Preserving  Proximity Tracing  Overview of Data Protection and Security. Version: 3rd April 2020. | | |  |

# Inleiding

*Dit document is in bewerking en ter review. Er zijn in deze versie ook nog de rode tekstdelen die nog te verwerken metadata bevatten.*

## Achtergrond

Binnen het Covid aanvalsprogramma zijn er een aantal werkstromen die allemaal bijdragen aan het hoofddoel, Nederland sneller en beter door deze crisis krijgen. Relevant in deze context zijn de technische werkstromen:

1. De GGD Covid19 notificatieapp - een breed uitgezette mobile app voor het publiek dat versneld informatie verstrekt aan de contacten van een door een laboratorium test besmet gevonden persoon.
2. Een Thuis Rapportage App -- een ondersteunden app voor positief geteste personen die actief begeleid worden door de GGD.
3. Een directe ondersteuning van de bestaande B&C opsporingsonderzoeken middels een portaal of website die een deel van de huidige formulier/telefoon gebaseerde processen automatiseert.
4. Een eventuele kleinschalige epidemiologische app die zeer complete en accurate input levert voor de RIVM modellen op basis van een kleine groep (vergelijkbaar met de Nivel peilstations en Pienter onderzoeken)

Dit document beperkt zich tot optie 1 - “GGD Covid19 notificatieapp”.

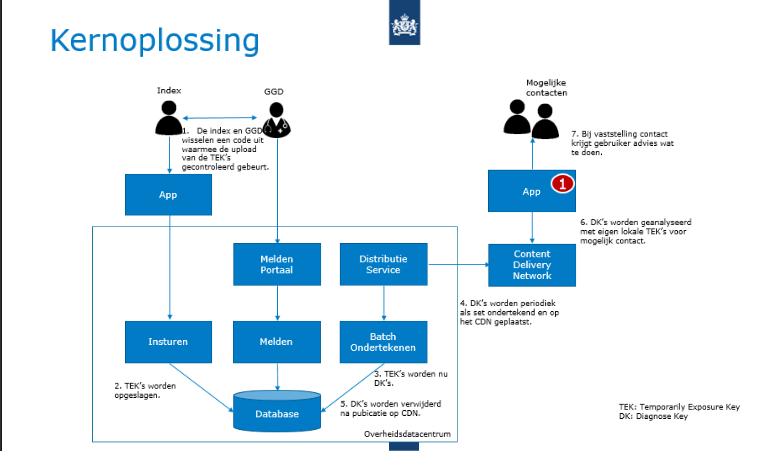
## Doelstelling en uitgangspunten

Doelstelling van dit document is om uitgaande van het Google Apple Exposure Notification (GAEN) raamwerk te komen tot een daarop aansluitend cryptosysteem voor de Covid19 notificatieapp ondersteuning van de bescherming van de privacy van deelnemende personen en de integriteit van de verwerking. Het GAEN raamwerk beoogt anonimiteit van deelnemers te waarborgen op basis van een cryptografische oplossing waarbij een zogenaamde Temporary Exposure Key (TEK) centraal staat. Iedere deelnemer krijgt t.b.v. de anonimiteit een willekeurige (“random”) cryptografische sleutel toegewezen. Het raamwerk moet zorgdragen voor behoud van de anonimiteit en de integriteit van die sleutel. Het model dat daarbij gehanteerd wordt is de algemeen gebruikelijke fasering binnen het lifecycle management van cryptografische sleutels. Voor cryptografische standaarden worden de standaarden gebruikt van:

1. COMMISSION RECOMMENDATION (EU) 2020/518 of 8 April 2020
2. GACT Exposure Notification Cryptography Specification
3. NCSC (NL)
4. ENISA-Sogis (EU)
5. Relevante nationale, internationale en industry de-facto standaarden (met name die waarop het GACT document terugvalt).

De keuze voor de cryptografische oplossingen is tot stand gekomen tegen de achtergrond van andere, niet cryptografische beveiligingsmaatregelen en het belang van gebruikersvriendelijkheid van de app. Het cryptoraamwerk is onderdeel van een breder pakket aan maatregelen en dekt niet alle risico’s af. Bekende (rest-) risico’s worden apart benoemd. Deze afwegingen (inclusief de noodzaak van compenserende maatregelen en eisen aan de governance) komen samen in het Data Protection Impact Assessment (DPIA) document.  
In de eerste versie van dit document wordt een vereenvoudigd model gehanteerd van verwerkingslocaties bestaande uit mobiele telefoons (MT) , upload server (US) en download server (DS). Naamgeving is verder op basis van GAEN en Dp-3T.

## Overzicht



Verdere detaillering en toelichting is terug te vinden in het infra architectuur document: Exposure Notificatie Infrastructuur Overview.ppt.

## Dreigingen, aanvallen en beschermende maatregelen op basis van cryptografie.

Het cryptoraamwerk beschermt tegen een subset van dreigingen en of aanvallen op

* de authenticiteit van de TEKs
* de anonimiteit van de positief geteste persoon

Het raamwerk biedt beveiliging tegen de volgende dreigingen:

1. Vervalste TEKs door upload van TEKs die niet afkomstig zijn van de telefoon als meegebracht door een positief getest persoon. Gevolg risico bestaat dan uit onterechte exposure notifications, gemiste notifications, overbelasting en daardoor uitval van het systeem.
2. Vervalste TEKs (diagnosed keys, van positief getesten) bij download. Het gevolg risico is als één. Het cryptoraamwerk beveiligt tegen downloads van fake sites of aanpassingen tijdens transport. Wijzigingen in het centrale server die vooraf gaan aan de cryptografische beveiliging vallen buiten de scope van dit document.
3. Doorbreken van de anonimiteit. GAEN kiest een cryptografische oplossing op basis van random getallen (TEKs) die per definitie (privacy by design) uit zich zelf niet terug leiden naar de telefoon of positief geteste persoon. Voor de beoogde authenticiteit bij upload is tijdelijke authenticatie en juist wel een koppeling noodzakelijk. Het raamwerk adresseert dit door de gebruikte authenticators slechts tijdelijk (niet langer dan noodzakelijk) te bewaren.

Voor overige dreigingen zie Bijlage A Dreigingsoverzicht

## Definities

|  |  |  |
| --- | --- | --- |
| begrip | synoniem | omschrijving |
| anoniem |  | Niet herleidbaar tot identiteit van persoon en / of telefoon of daaraan bijdragend in samenhang met andere data. |
| integriteit |  | Hier beperkt tot controleerbaarheid op niet gewijzigd zijn van echte TEKs. |
| identiteit |  | Set van eigenschappen met bepaalde waarde die gebruikt kan worden om een element uit een verzameling (persoon of machine) uniek re selecteren. |
| Authenticatie |  | Technisch bewijzen van een identiteit van mens of machine |
| Authenticiteit | Echtheid | Combinatie van integriteit en authenticatie |
| MT |  | Mobiele Telefoon |
| TEK | Temporary Exposure Key | Dagelijkse random warde van 128 bit als basis voor afleiding van dagelijkse RPI-Key en AEM-Key |
| TEKRollingPeriod | Roll over time | Max Life Time of TEK, “a day” |
| RPI-Key |  | Dagelijkse 128 bit sleutel als basis voor encryptie van een string van vaste en op basis van interval variabele data tot de RPI |
| ENIntervalNumber | 10 minutes Number | Number of 10 minute periodes elapsed since |
| RPI | Rolling Proximity Idenfier | Anonieme “identifier” welke gebroadcast wordt door bluetooth op de MT. |
|  |  |  |
| AEM-Key |  | Dagelijkse 128 bit sleutel als basis voor encryptie van een string van variabele data tot de AEM |
| AEM | Associated Encrypted Metadata | Versleutelde metadata welke in combinatie met RPI gebroadcast wordt. |
| DK | Diagnosed Key Infected Key |  |
| DS | Diagnose(d) Server Publisher Server Download Server Distribution Server | Centrale server waarvandaan diagnose keys gedownload worden |
| CDN | Content Delivery Network | voor verdere propagatie en distributie |
| RS | Receiving Server UpLoad Server | Centrale server waarnaartoe diagnose keys ge-upload worden. |
| DRS | Distributed RS | Meerdere “locale” front end receivers. |

# TEK life cycle managment

Het Cryptoraamwerk wordt opgebouwd rond de lifecycle van de Temporary Exposure Keys (TEKs). De TEK keys van een positief geteste deelnemer worden ook wel Diagnose(d) Keys genoemd.

## Key generation and derivation

Het GAEN raamwerk beoogt de privacy van deelnemers te borgen door dagelijks een ieder een anonieme identifier toe te kennen op basis van een random sleutel, de **TEK** waarvan 2 voor encryptie te gebruiken sleutels afgeleid worden, de **RPIK** en **AEMK**, zie tabel.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **generation** | | | | | |
| Freq | name | | size | source | schema |
| daily | **TEK** | Temporary Exposure Key | 128 |  | CRNG |
|
| **derivation** | | | | | |
|  |  |  |  | schema | |
| daily | **RPIK** | Rolling Proximity Identifier Key | 128 | HKDF(**TEKi**, NUL L, UTF8("EN-RPIK"),16) | |
| daily | **AEMK** | Associated Encrypted Metadata Key | 128 | AEMK ← HKDF(**TEK**i , NULL, UTF8("EN-AEMK"),16) | |

Afleiding van de 2 encryptie sleutels is deterministisch en zonder toevoeging van een random waarde. Deze oplossing levert alleen werkelijk anonimiteit als er sprake is van een Truly Random Number voor de TEK. Daarbij spelen 2 eigenschappen, statistisch random èn non predictability.

Apple geeft aan te werken op basis van

1. A True random number generator als bron.   
   *The /dev/random generator is* ***a true random number generator*** *that obtains entropy from interrupts generated by the devices and sensors attached to the system and maintains an entropy pool.* ***The NDRNG provides 128-bits of entropy.***
2. *The* ***NDRNG*** *feeds entropy from the pool into the DRBG on demand. A FIPS 140-2 approved deterministic random bit generator based on a block cipher as specified in NIST SP 800-90A is used.*  
   Deze tweede functie heeft de non predictability eigenschap niet meer maar door de 128 bits input (=output) blijft de 128 bits entropie gehandhaafd.

Google geeft aan dat:  
*Android's SecureRandom is backed by BoringSSL, BoringSSL provides much more than just TLS but also most of the crypto primitives on Android. Under the hood the SecureRandom is backed directly by calls to BoringSSL's RAND\_bytes, which provides* ***cryptographically secure random*** *numbers and is itself backed by urandom on Android*.

## Key Storage

De TEKs en afgeleide RPIK en AEMK keys zijn eigendom van de gebruiker en liften mee op out of the box aanwezige beveiligingsmaatregelen op de mobiele telefoons.

De TEKs en hun afgeleiden zijn in combinatie met de telefoon en dus de identiteit van de gebruiker een medisch gegeven. TEKs en alle afgeleiden moeten der halve beveiligd (b.v. versleuteld) worden om toegang buiten de bedoelde functie van de app te voorkomen.

Apple geeft aan dat:

*We state it’s stored securely but I do not have additional details. (vraag 67 nog open)  
Only EN apps have access to TEKs and only with prior content by the user (permission dialog shown by getDiagnosisKeys()).*

En specifiek over RPI’s verder:

*Eavesdropping EphIDs is not possible on the most recent versions of Apple API, only on Android. Apple sets the EphID to 0 if any app tries to access BLE beacons used for proximity tracing, while Google allows accessing it.*Daarnaast is er generieke / platform beveiliging als beschreven in:

* <https://www.apple.com/business/docs/site/AAW_Platform_Security.pdf>
* <https://www.apple.com/business/docs/resources/Managing_Devices_and_Corporate_Data_on_iOS.pdf>

Google geeft aan dat:  
*TEKs are protected by the application sandbox and Android's disk encryption.  
Practically this means that only Google Play Services can access this data, no other application or the OS itself can access the data. Google Play Service alone controls access.*

Er is geen beveiliging voor RPI’s zoals beschreven bij Apple.

Wel is er ook de generieke / platform beveiliging als beschreven in:

* <https://source.android.com/security>

Op de receiving server en de gehele backend wordt geen encryption in rest toegepast maar vertrouwd op de fysieke en logische beveiliging in het rekencentrum.

## Key Usage

### Encryptie, generatie van RPI en AEM

De RPIK en AEMK (beiden afgeleid van de TEK) worden iedere interval (10-20 minuten) gebruikt voor encryptie waarbij een voor het interval unieke Rolling Poximity Identifier (RPI) en versleutelde metadata, Associated Encrypted Metadata (AEM) ontstaan. Beide worden gezamenlijk met bluetooth gebroadcast. Er is vanuit GAEN afstemming op de eveneens gebroadcaste temporary mac adressen ter voorkoming van linking van de diverse PRIs. Dit gebeurt overigens niet ‘in’ de app; maar in een proprietary library die onderdeel uitmaakt van het operating system van de mobile telefoon leverancier. Deze informatie is dus niet beschikbaar voor de App (voor het besmet&consent moment).  
  
De huidige versie van het GAEN raamwerk kent nog wel de incorporatie van datum tijd in de RPI waardoor uiteindelijk toch een verband met volgorde te maken is. Wijzigingsvoorstel staat uit bij Apple en Google.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Usage:** | | | | |
| **encryptie** | | | | |
| freq | naam |  | size | Schema/algoritme |
| 10 m | **RPI** | Rolling Proximity Identifier | 128 🡺128 | AES128(RPIK , PaddedData) |
| 10 m | **AEM** | Associated Encrypted Metadata | 32🡺128 | AES128−CTR(AEMK, RPI, Metadata) |

### Encryptie, generatie van potentially contacted RPIs

Na ontvangst van de diagnose keys wordt het voor de eigen RPI gebruikte proces van key derivation en 10-min-interval encryption hiermee herhaald ten einde een match te kunnen maken met opgeslagen RPIs waarmee een contact geweest is. De app heeft ook hier geen toegang tot deze data - ze wordt beheert door een bibliotheek/Operating Systeem.

## Key Backup

Ad 1, vanaf de mobiele telefoon.

Google geeft aan dat:  
*Nothing is shared into the cloud; There are no backups.*

Apple geeft aan dat:  
*Exposure keys never leave the device and are not backed up to iCloud.*

Ad2, op de receiving server

Er wordt uitgegaan van een HA storage oplossing waarvoor geen backup gemaakt wordt.

De positief geteste keys (diagnosed keys) worden in the clear gepubliceerd. Er is geen greep op het bewaren van de keys in het publieke domein (na download). De gepubliceerde sleutels worden wel voorzien van een digitale handtekening (die de app verifieerd).

## Key Archival

Key archival is niet van toepassing voor de Corona App. Key Archival in het publieke domein is niet onder controle van de app. Archival bestaat wel rond voor auditing relevante handelingen rond de positief geteste keys maar dat is out of scope voor dit document.

## Key Destruction

De app verwijdert TEKs ouder dan 14 dagen van de telefoon. Op de receivingserver worden authentiek bevonden diagnosed keys verwijderd na publicatie. Afgewezen keys worden eveneens verwijderd. Gepubliceerde diagnosed keys worden na 14 dagen weer verwijderd.

Apple doet op de telefoon een key reset / rollover na upload van de op die dag gebruikte key. Uploaded TEKs worden dus nooit actief gebruikt.

TEK key management wordt uitgevoerd in een bibliotheek of het operating system. De app heeft hier zelf geen controle over.

## Key Transport, upload van TEKS

### Requirements

1. Privacy Preserving
2. Voorkom brute force uploads
3. authenticiteit van de uploaded TEKs:
   1. integriteit / onveranderd tijdens transport
   2. bron integriteit (kort), controleerbaar afkomstig van de telefoon van de positief geteste persoon, 🡺 “authenticatie”. Niet blijvend herleidbaar tot de telefoon.
4. doel integriteit, upload naast de juiste server, authenticatie van de receiving server.

De geschetste oplossing beoogt zowel recht te doen aan authenticiteit van de keys als privacy by design toe passen. De voor authenticiteit opgebouwde vertrouwensband wordt zo kort als mogelijk in stand gehouden. De voor de vertrouwensband gebruikte authenticators, waarlangs in principe herleidbaarheid van diagnosed keys tot mobiele telefoon of persoon zou kunnen ontstaan worden niet eerder dan noodzakelijk gegenereerd of aangeleverd en direct na gebruik vernietigd. De link is éénmalig en tijdelijk.

De oplossing beoogt privacy by design toe te passen binnen het eigen functie gebied. Beneden wordt b.v. ter aanvulling een opmerking gemaakt over de registratie van ip adressen wat echter buiten de hier beschreven cryptografische functies valt.

Gebruikersvriendelijkheid is noodzakelijk, zeker als een code ingetoetst moet worden door een zojuist positief geteste persoon. De gezochte oplossing beoogt dan ook niet te te leunen op de cryptografische sterkte van “TAN” codes wat hen ondoenlijk lang en complex zou maken.

De cryptografische oplossing moet ook geen wiel uitvinden; kans op fouten bij cutting edge implementaties is te groot en een risico voor de beoogde doelen op zich.

Er worden 2 flows uitgewerkt, details zijn te vinden hier:

Flow 1

:<https://github.com/minvws/nl-covid19-notification-app-coordination/blob/master/architecture/Solution%20Architecture.md>

Flow 2:  
<https://docs.google.com/document/d/1JZYEL4zpX02dE1Qi09LXZho4bRfi84k30hsbfTqOI-c/edit?ts=5eddf2e0#heading=h.pss2ecxwcpvp>

### Functioneel flow

Op hoofdlijnen bestaat de upload bij Flow1 één uit de volgende stappen:

1. Uitwisseling authenticators / keys:  
   De App op de mobiele telefoon maakt verbinding over TLS met de backend en ontvangt een gebruikersvriendelijke LabConfirmationCode (LCC) en 256 bits shared secret (ConfirmationKey, CK) van de backend. LCC en CK zijn nu op 2 plekken.
2. Initiatie van Upload:  
   De positief geteste persoon vertelt de LCC aan een operator. De operator zet, bij een positieve test, de CK op groen voor gebruik m.b.v. de opleesbare LCC. Hiervoor is een over het internet bereikbare portal (beheerpagina) waar de operator met 2-factor authenticeert. Hij zij logt aan met 2-factor authenticatie.
3. Beveiligde upload:  
   De uploaded TECs worden door de mobiele telefoon voorzien van een MAC (HMAC) op basis van de CK; de backend verifieert.
4. Verwijderen van de authenticators en keys:  
   Nadat alle TEKs uploaded zijn (2 batches) of zoveel eerder als mogelijk worden alle authenticators (LCC, CK etc) verwijderd.

### privacy preserving

1. Beperking rond ip logging. Logs zo kort mogelijk bewaren en toegang beperken tot need-to-know.
2. Z.s.m. na upload en verificatie van de diagnosed keys verwijderen van gebruikte codes en keys (LCC,CK) en cryptografische sleutels, zowel op het mobiel als in het centrale systeem om linkability naar de gebruiker te doorbreken.
3. Blinderen van upload van TEKs d.m.v. decoy traffic (zie verder) zodat een patroon van netwerkverkeer niet de informatie ontsluit dat iemand positief getest is.
4. Bij zeer lage aantallen (weinig positief geteste personen) worden fake keys toegevoegd aan de distributie server om herleiden van TEKs tot positief geteste persoon te verhinderen.

### Voorkom brute force uploads

1. SOC / Firewall oplossing ter voorkoming brute force attack. Afweging hierbij is dat het aantal uploads relatief laag is (in theorie de testcapaciteit) en dat telefoons maar zelden contact maken. Dit maakt brute-force “DoS” mitigatie makkelijk en het dus mogelijk de cryptografische oplossing minder zwaar te maken ten einde de gebruikersvriendelijkheid tegemoet te komen.

### Authenticiteit van de uploaded TEKs (alleen echte TEKs)

1. HMAC-SHA256 op basis van CK (shared secret) t.b.v. de authenticiteit van de diagnosed keys. Hiermee is de oorsprong van de TEKs gegarandeerd (die éne telefoon), kunnen keys niet aangepast of tijdens transport toegevoegd worden en veilig in meerdere batches (2) verstuurd worden. Het laatste punt is onmogelijk bij alleen meesturen van b.v. de LCC, daarbij zou een herbruikbare authenticator over het netwerk verstuurd worden.

### Gebruiksvriendelijk en veilig

1. De authenticiteit van de TEKs (alleen echte TEKs) is gebaseerd op de HMAC functie op basis van de CK. De CK wordt geactiveerd aan de hand van de LCC die door de patiënt opgelezen moet kunnen worden van af de telefoon en door de GGD operator ingevoerd moet kunnen worden. De GGD operator activeert de LCC op een internet gebaseerd portal. De LCC moet gebruikersvriendelijk zijn. Veiligheid kan daarom niet gebaseerd zijn op een cryptografisch afdoende sterkte van de LCC alleen.
2. Het activatie portal is tevens een aan het internet gekoppelde beheerpagina. De beveiliging komt dan tot stand door voor de GGD operator authenticatie in te regelen op basis van 2-factor authenticatie: userid/wachtwoord en een tweede factor, geïmplementeerd met de google / microsoft authenticator OTP app (of gelijkwaardige app). Daarmee blijft het veilig en gebruikersvriendelijk.  
   In die context kan de complexiteit LCC beperkt blijven.  
   Keuze uit 23 letters/cijfers (alfabet minus op elkaar lijkende getallen) en dat 6 keer als in XXX-YYY. Dat geeft 150.000.000 mogelijkheden en daarmee een kans van slechts 1 op 10.000 van raden bij 15.000 openstaande LCCs.

### Authenticiteit van de receiving server

PKI-O TLS certificaat. Toegestane TLS strings:

|  |
| --- |
| ECDHE\_RSA TLS\_AES\_256\_GCM\_SHA384 |
| ECDHE\_RSA TLS\_AES\_128\_GCM\_SHA256 |
| TLS\_ECDHE\_RSA\_WITH\_AES\_256\_GCM\_SHA384 |
| TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256 |
| TLS\_ECDHE\_RSA\_WITH\_AES\_256\_CBC\_SHA384 |
| TLS\_ECDHE\_RSA\_WITH\_AES\_128\_CBC\_SHA256 |

Bewust is afgezien van het gebruik van de DeviceCheck API. Implementatie bleek bewerkelijk en er is onduidelijkheid rond de privacy. Mogelijk leidt het tot linkability naar de telefoon. Authenticiteit van de TEKs is ook zonder deze API al afgedekt.   
  
TLS eenzijdig, certificate pinning op Staat der Nerderlanden RootCA (of een lager gelegen certificaat). Dit geeft aanvullende integriteit tijdens transport met een werkbare implementatie van het sleutelbeheer. Een hard coded check op een vast end entity certificaat is operationeel bewerkelijk en foutgevoelig.

Zie verder **Bijlage B Overwegingen rond upload van TEKs**

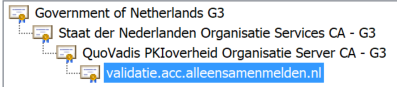
## Key Transport, downloads van TEKs

### requirements

1. Authenticiteit voor diagnose keys:  
   bron integriteit + integriteit tijden transport en (kortstondig) na download.
2. Authenticiteit voor download / Distribution Server / CDN
3. Vertrouwelijkheid (+ integriteit) tijdens download
4. Authenticiteit van config files

### Authenticiteit van diagnose keys

Er wordt gebruikt gemaakt van 2 keer een geavanceerde digitale handtekening.   
De eerste conform GAEN vanwege interoperabiliteit; hierbij zijn uiteindelijk Apple en Google in control. De tweede op basis van een eigen pki-certificaat gebaseerde handtekening. De app zelf is hier in control en uiteindelijke bepaalt de verificatie van deze handtekening of de app de TEKs als authentiek beschouwt.

1. Conform specs GAEN  
   Handtekening zonder certificaat, conform GAEN, ECDSA, P-256 curve, SHA256  
   Private key beheren op basis van FIPS 140-2, L2+ in netHSM, OCS protected.  
   Public key aanleveren bij Apple / google.  
   De Public key wordt in PEM format gedeeld met Google na openen van een issue in de Google Bug Tracker, upload van de key en verificatie van het mailadres door Google. Apple ?  
   OTAP scheiding op basis van public / private pair per Bundle-ID waarbij:  
   App-ID=Team-ID || Bundle-ID  
   Key rollover (b.v. bij key compromise) op basis van:  
   -send new public key  
   -start signing with new private key,   
   -specify \_v2 in the verification\_key\_version field.  
   -Request Apple / Google to revoke first key when all downloaded TEKs have expired.
2. Ge-avanceerde, pki-certificaat-gebaseerde handtekening:  
   Algoritme conform PKI-O: minimal RSA2048SHA256.  
   signing certificaat aanvragen bij PKI-O leverancier van BS,   
   PKI-O certificaat voor Client Authentication en Server Authentication,  
   signing = critical, geen gekwalificeerd certificaat voor handtekening.  
   Handtekening op basis van RFC5652 (PKCS#7 / CMS detached), der encoded, signature op basis van een SHA256 hash. The SHA256 over precies de payload, d.w.z. de TEKs (multiple of 8 bit (i.e. complete bytes)). Het signing certificaat en de CA server certificaten (tot aan de RootCA) worden meegegeven met de handtekening. Private keys van de handtekeningen beheren op basis van FIPS 140-2, L2+ in netHSM, OCS protected. OTAP scheiding op basis van verschillend certificaat per omgeving. Key rollover (b.v. bij key compromise) op basis van certificate revocation procedure bij TSP (certificaat leverancier).   
   

### Authenticiteit van Fake keys

Bij lage aantallen (weinig positief getesten en dus een beperkte set uploaded TEKs ) wordt de set uitgebreid met fake keys. Signing is tweeledig en niet anders dan bij echte TEKs:

* Conform specs GAEN
* X.509 Certificaat (PKI-O) gebaseerde handtekening

Zie ook: **Bijlage C Fake Keys**

### Authenticiteit van diagnose keys Internationaal

De internationale afspraken liggen vast in het document:  
20200603\_CWA\_Interop\_Architecture.doc.  
Hier wordt een overzicht gegeven; het internationale document is leidend.

#### TLS verbinding met Federation GateWay (FGW) server

TLS 1.2, tweezijdige geauthentiseerd. Waarbij aan beide kanten certificate pinning plaats vindt op ca server certificaat. Client Certificaat eigenaar geeft het pin niveau (ca server certificaat uit de boven liggende PKI hiërarchie) door aan de FGW. De FGW zorgt voor communicatie van de pin niveaus van de PKI chains van de client certificaten en het eigen server certificaat naar allen.

#### Geavanceerde digitale handtekening over TEKs bij upload Conform CMS handtekening als nationaal gebruikt voor authenticiteit van de diagnose keys. Aan de handtekening worden de ca server certificates van de boven het signing certificate liggende PKI hiërarchie toegevoegd tot aan het RootCA certificaat.

Wederom certificate pinning en communicatie van niveau via de FGW.

### Authenticiteit van download / Distribution Server / CDN

De origin server (distribution server) en het CDN worden voorzien van een PKI Certificaat (overheids webside moet herkenbaar) en ondersteunt TLS.   
  
Aanvragen bij CIBG , in ieder geval PKI-O  
project specifieke naam en/of VWS in subject DN.

Zowel Android als iOS gaan uit van HTTPS, en op Android moet je je app expliciet configureren voor gebruik van HTTP, wat niet gebruikelijk is.

### Verificatie op Client van de gedistribueerde bestanden met diagnosed keys

Verificatie van de GAEN handtekening conform GAEN.  
De PKI certificaat gebaseerde handtekening verifieren comfom RFC 5652.   
Pinning op:

* Issuing CA (CA server die het signing certificaat uitgaf) d.m.v. verificatie van de **Subject Key Indentifier in het certificaat van de Issuing CA** (vaste waarde).
* **Subject naam in het signing certificaat**.

Op die manier is het mogelijk b.v. jaarlijks het signing certificaat te vervangen zonder update aan client zijde zolang het signing certificaat onder dezelfde subject naam uitgegeven wordt door dezelfde issuing ca.  
De verificatie gaat over de handtekening op de TEKs, het signing certificaat en alle bovenliggende CA certificaten tot en met een voorgeregistreerd RootCA certificaat op het device.

## Transport, downloads van config files

### Signing en verificatie van Config Files

De config files welke door de apps gedownload en verwerkt worden dienen eveneens gesigned en geverifieerd te worden (zelfde manier voor signing en verificatie als voor de diagnosed keys).   
Immers:

De App is signed (Apple, Google)

De TEKs zijn signed

🡺 voor config geen uitzondering

## Crypto Specs, algorithms, schema’s and protocols

|  |  |  |
| --- | --- | --- |
| HKDF | Output 🡸 HKDF(Key, Salt, Info, OutputLength) | IETF RFC 5869 |
| AES | Output 🡸 AES128(Key, Data) | 128 bits |
| AES-CTR | Ciphertext 🡸 AES−CTR(Key, IV, Data) | 128 bits |
| HMAC | MAC🡸HMAC(Key, Data) | 256 bits |
|  |  |  |

# Bijlage A Dreigingsoverzicht

Kort overzicht van dreigingen op basis van eerdere studies van derden:

|  |  |
| --- | --- |
| **Privacy and Security Risk Evaluation of Digital Proximity Tracing Systems The DP-3T Project, 21 April 2020** | |
|  | Crypto mitigated |
| IR 1: Identify infected individuals | n |
| IR 2: Prevent notifications | n |
| GR 1: Cause false alarms through BLE range extensions | n |
| GR 2: Cause false alarms through active relays | n |
| GR 3: Identify location with infected people present | n |
| GR 4: Disrupt contact discovery | n |
| GR 5: Tracking a Bluetooth enabled device | n |
| GR 6: Reveal usage of the contact tracing app | n |
| NR 1: Network identities reveal data about infected patients | n |
| NR 2: Traffic analysis reveals data about infected patients | n |
| SR 1: Reveal social interactions | ? |
| SR 2: Recompute the risk score given the recorded observation, possibly using | n |
| SR 3: Location tracing after local phone access | n |
| SR 4: Location tracing of infected individuals | n |
| SR 5: Reveal social interactions to a central server | n |
| SR 6: Reveal colocation information about infected individuals to a central server | n |
| SR 7: Location tracing through access to a central server | n |
| SR 8: Reconstructing social interaction graphs | n |
| SR 9: Reveal at-risk status to a central server | n |
| **SECURITY ANALYSIS OF THE COVID-19 CONTACT TRACING SPECIFICATIONS  BY APPLE INC. AND GOOGLE INC.** | |
| Power and Storage Drain Attacks | n |
| Relay and Replay Attacks | n |
| Trolling Attacks | n |
| Tracking and Deanonymization Attacks | n |
|  |  |

* Server modification (aanvallen direct op de receiving of distribution servers)
* Noodzaak tot zaken als een ‘decoy’ om bepaalde zaken te verhullen / niet herleidbaar in tijd en plaats te maken.

# Bijlage B Overwegingen rond upload van TEKs

Het dilemma: anonimiteit lijkt haaks te staan op bron integriteit.

Technische oplossingen in de discussie

1. Code uitwisselen bij upload van TEKs geeft risico op hergebruik / replay.
2. MAC, Message Authentication Code op basis van HMAC-SHA256. Een MAC waarde is een cryptografisch controle getal voor:
   1. Verificatie van onveranderlijkheid van de data (b.v. tijdens transport)
   2. Verificatie van de bron, alleen partijen die de juiste sleutel hebben kunnen de MAC waarde berekenen. In ons voorbeeld is dat de app op de mobiele telefoon en de receiving server.
3. TLS tijdens transport, eenzijdig geauthentiseerd, alleen server certificate.  
   Biedt doel integriteit (server authenticatie), integriteit tijdens transport en vertrouwelijkheid (TEKs waren al anoniem, geen echte requirement).
4. TLS tijdens transport, tweezijdig geauthentiseerd, dus inclusief client authenticatie en bron integriteit waarbij:
   1. Alle app instanties zelfde certificaat en dan weinig waarde, geen bron integriteit
   2. Iedere app een eigen certificaat , wel bron integriteit maar leidt tot authenticatie en verlies van anonimiteit.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | AuthenticationCode /Tan | MAC | TLS-éénzijdig | TLS-mutual auth |
| anoniem | Herleidbaar (tijdelijk) | Herleidbaar  (tijdelijk) | ja | nee |
| Integriteit tijdens transport | nee | ja | ja | ja |
| Bron integriteit | ja | ja | nee | ja |
| Doel integriteit | nee | nee | ja | ja |
| Voorkom wilde uploads | nee | nee | nee | ja |

# Bijlage C Fake Keys

Fake keys worden 2 maal getekend, conform GAEN en certificaat gebaseerd:

#!/bin/sh

set -e

# ES256 (ECDSA p256)

openssl ecparam -name secp256k1 -genkey -noout -out ec.key

openssl ec -in ec.key -pubout -out ec.pub

echo Something to Sign > payload

# Sign something

#

openssl dgst -sha256 -sign ec.key payload > signature.bin

openssl dgst -sha256 -verify ec.pub -signature signature.bin payload

echo EC256 k1

echo pubkey:

cat ec.pub

echo Sginature:

cat signature.bin | base64

openssl req -new -x509 -nodes -keyout x509.key -out x509.pub -subj /CN=CoronaPub

openssl cms -sign -signer x509.pub -inkey x509.key -outform DER -out signature-cms.bin -in payload

echo

echo X509

cat x509.pub

echo

echo Signature using above cert:

cat signature-cms.bin | base64

zip fake.zip payload signature.bin signature-cms.bin