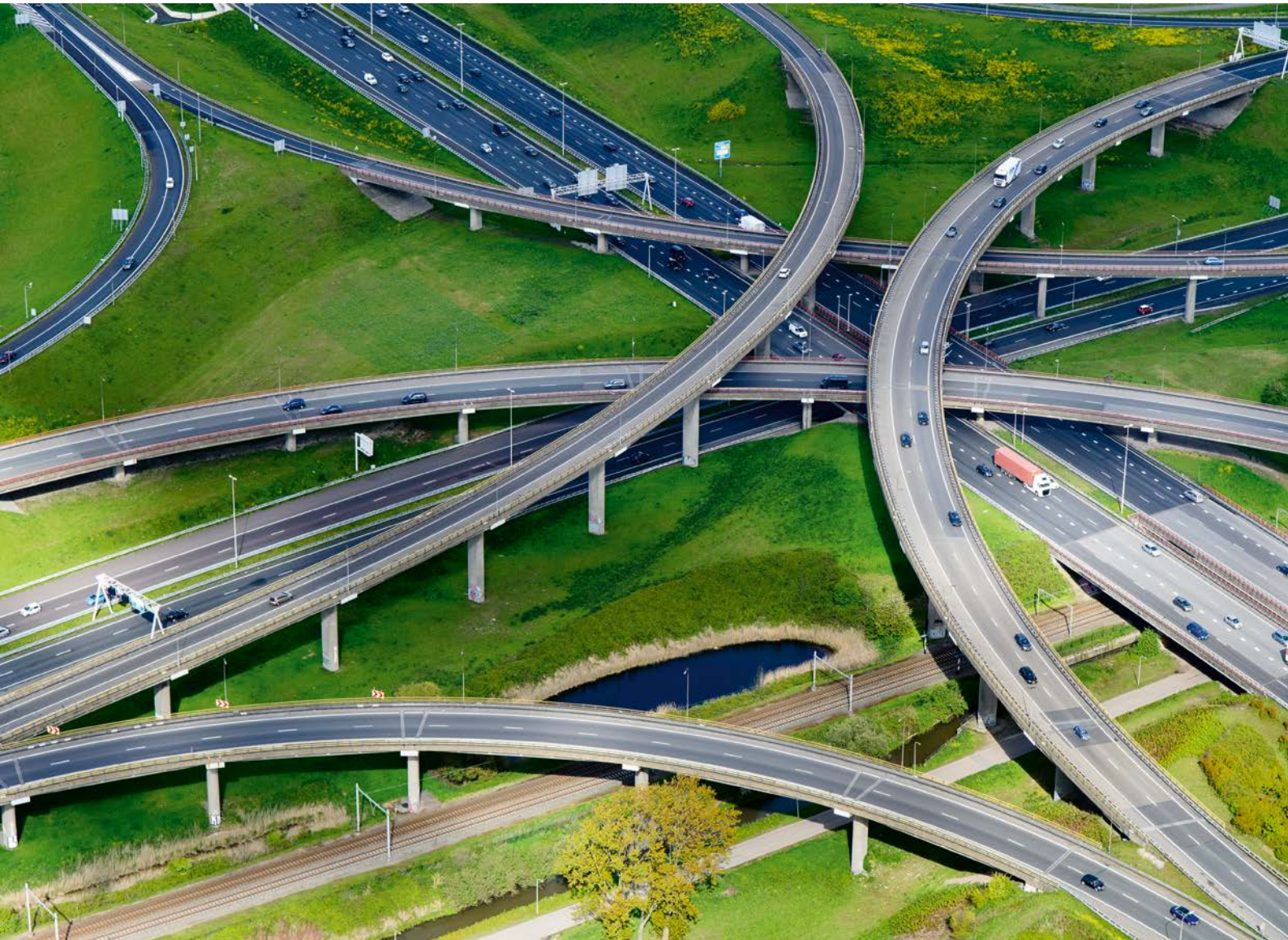
Nationaal Cyber Security Centrum
Ministerie van Justitie en Veiligheid

ICT-beveiligingsrichtlijnen voor Transport Layer Security (TLS)



National Cyber Security Centre

Het Nationaal Cyber Security Centrum (NCSC) draagt via samenwerking tussen bedrijfsleven, overheid en wetenschap bij aan het vergroten van de weerbaarheid van de Nederlandse samenleving in het digitale domein.

Het NCSC ondersteunt de Rijksoverheid en organisaties met een vitale functie in de samenleving met expertise, advies, respons op dreigingen en het versterken van de crisisbeheersing. Daarnaast biedt het NCSC informatie en advies voor burger, overheid en bedrijfsleven ten behoeve van bewustwording en preventie. Het NCSC is daarmee het centrale meld- en informatiepunt voor ICT-dreigingen en -veiligheidsincidenten.

Deze ICT-beveiligingsrichtlijnen voor Transport Layer Security zijn in 2014 voor het eerst door het NCSC gepubliceerd. De huidige update dateert van 2019. Zie de bijlage *Wijziging van deze richtlijnen* voor meer informatie hierover.

Deze publicatie is opgesteld in samenwerking met de volgende partners:
- het Nationaal Bureau voor Verbindingsbeveiliging (NBV) dat deel uitmaakt van de Algemene Inlichtingen- en Veiligheidsdienst (AIVD).

De volgende organisaties en personen hebben waardevolle bijdragen geleverd aan de totstandkoming van deze publicatie:
- Autoriteit Persoonsgegevens
- Belastingdienst
- Centric
- Dienst Publiek en Communicatie
- Forum Standaardisatie
- IBD
- KPN
- NLnet Labs
- Northwave
- Platform Internetstandaarden
- RDW
- SURFnet
- de Volksbank
- Z-CERT

- Daniel Kahn Gillmor, ACLU
- Tanja Lange, Eindhoven University of Technology
- Kenny Paterson, ETH Zurich
- Rich Salz, Akamai Technologies
- Nick Sullivan, Cloudflare

ICT-beveiligingsrichtlijnen voor Transport Layer Security (TLS)

Inhoud

Inleiding

Deze richtlijnen zijn bedoeld als advies bij het inkopen, opstellen en beoordelen van configuraties voor het Transport Layer Security-protocol (TLS). TLS is het meest gebruikte protocol voor het beveiligen van verbindingen op internet.

Doel

Deze richtlijnen bevatten geen stapsgewijze instructies voor het configureren van TLS.[1] Niettemin zijn zij technisch van aard. Dit document helpt een organisatie te kiezen tussen alle mogelijke TLS-opties om zo te komen tot een veilige configuratie. Een beheerder of leverancier past de configuratie vervolgens toe.

Gebruik bij inkoop

Organisaties die ICT-systemen inkopen, kunnen naar dit document verwijzen bij het stellen van eisen aan de te leveren producten. Wanneer een leverancier aan de richtlijnen in dit document voldoet, levert en onderhoudt hij een veilige TLS-configuratie.

Veiligheidsniveau

De beslissing over de juiste TLS-configuratie maakt elke organisatie uiteindelijk zelf. Het kiezen van een veilige configuratie is een complex karwei. Elke optie vereist een keuze tussen de beschikbare alternatieven en dat zijn er vaak heel veel. Bij deze keuze speelt veiligheid natuurlijk een rol, maar er moet bijvoorbeeld ook rekening worden gehouden met de compatibiliteit met de software van klanten of eindgebruikers. De richtlijnen in deze publicatie fungeren als gids bij deze taak.

Om de keuze voor een configuratie te vergemakkelijken, zijn de instellingen voor TLS-opties in deze richtlijnen in vier veiligheidsniveaus verdeeld:

- Een instelling die **Onvoldoende** is, moet niet worden gekozen. TLS-configuraties die deze instelling bevatten, zijn niet veilig.
- Van **Uit te faseren** instellingen is bekend dat ze fragiel zijn met oog op de doorontwikkeling van aanvalstechnieken. Dergelijke instellingen bieden slechts een geringe veiligheidsmarge. Daardoor lopen ze het risico dat ze in de toekomst de status **Onvoldoende** krijgen. Voor een aantal clients zijn die **Uit te faseren** instellingen (nog steeds) nodig, omdat zij hele oude applicaties gebruiken. Het gebruik van deze instellingen moet worden onderworpen aan schriftelijke voorwaarden voor de uitfasering ervan, waarmee de beëindiging van het gebruik wordt gepland.
- Is een instelling **Voldoende**, dan betekent dit dat die instelling 'nu nog voldoet'. Het is dus nog steeds mogelijk om deze instelling in een veilige TLS-configuratie te gebruiken. Vaak zijn **Voldoende** instellingen nodig voor de compatibiliteit met oudere client-systemen.
- De veiligste en meest toekomstbestendige instellingen hebben de kwalificatie **Goed**. Hebt u de vrijheid om uw eigen instellingen te kiezen, gebruik dan waar mogelijk alleen **Goede** instellingen.

Van tijd tot tijd worden er nieuwe of verbeterde aanvalstechnieken voor TLS ontwikkeld. Dergelijke aanvalstechnieken hebben meestal betrekking op **Uit te faseren** of **Voldoende** instellingen. Een instelling die als gevolg van een aanvalstechniek onveilig is geworden, verliest haar status van **Uit te faseren**, **Voldoende** of **Goed**. In dergelijke situaties wordt er een aanvulling op deze richtlijnen gepubliceerd. Zie voor meer informatie de bijlage *Wijziging van deze richtlijnen*.

Goede instellingen zijn waarschijnlijk toekomstbestendiger dan **Voldoende** instellingen. Hier kan echter geen garantie voor worden gegeven. Bovendien is geen enkele TLS-configuratie 'voor altijd' veilig. Zelfs TLS-configuraties die uitsluitend uit **Goede** instellingen bestaan, moeten op een gegeven moment geactualiseerd worden. Dat is bijvoorbeeld het geval wanneer de status van **Goede** instellingen tot **Onvoldoende** wordt afgewaardeerd.

1 Het boek 'Bulletproof SSL and TLS' van Ivan Ristic (ISBN 978-1907117046) biedt, naast veel achtergrondinformatie over TLS, wel stapsgewijze instructies voor het configureren van uiteenlopende software voor een veilig gebruik van TLS. Mozilla geeft op haar wiki-gedeelte configuratievoorbeelden voor populaire webserversoftware: https://wiki.mozilla.org/Security/Server_Side_TLS. De website https://bettercrypto.org/ geeft eveneens stapsgewijze instructies. NB: Het is mogelijk dat deze informatiebronnen nog niet zijn bijgewerkt sinds de introductie van TLS 1.3. Het advies in de genoemde publicaties kan enigszins afwijken van het advies in dit document.

De woorden 'onvoldoende', 'uit te faseren', 'voldoende' en 'goed' hebben ook een betekenis in dagelijks taalgebruik. Om het onderscheid duidelijk te maken, worden deze woorden in de richtlijnen in een **vet lettertype** weergegeven, wanneer ze naar een veiligheidsniveau verwijzen.

Hoofdboodschap

Een veilige TLS-configuratie is belangrijk voor het beveiligen van netwerkverbindingen. TLS kent veilige en minder veilige instellingen. Helaas ondersteunt oudere software niet altijd de meest veilige instellingen. Gebruik waar mogelijk **Goede** instellingen, en vul deze aan met **Voldoende** instellingen ter ondersteuning van oudere software. Beschikt u over veel oudere software die ondersteuning nodig heeft? Gebruik dan een breed palet aan **Voldoende** instellingen en vul deze waar mogelijk aan met **Goede** instellingen. Gebruik uitsluitend **Uit te faseren** instellingen wanneer dat nodig is met het oog op de client-compatibiliteit en formuleer duidelijke criteria voor de beëindiging van het gebruik ervan. Gebruik geen **Onvoldoende** instellingen.

Leeswijzer

De kern van deze richtlijnen wordt gevormd door de hoofdstukken *Gebruiksadvies*, *Richtlijnen en Versies, algoritmes en opties*. Het hoofdstuk *Gebruiksadvies* is bedoeld voor degenen die zelf een veilige TLS-configuratie moeten creëren. Dit hoofdstuk bevat richtsnoeren en adviezen om die veilige configuratie tot stand te brengen. Het hoofdstuk *Richtlijnen* is bedoeld voor degenen die TLS-configuraties moeten beoordelen, zoals auditors. Daarbij kan het zowel om configuraties op papier als in de praktijk gaan. Het hoofdstuk *Versies, algoritmes en opties* bevat relevante TLS-opties. Ook worden voor elke optie de veilige instellingen beschreven. In andere hoofdstukken wordt regelmatig naar het hoofdstuk *Versies, algoritmes en opties* verwezen voor nadere informatie.

Deze richtlijnen kunnen op drie manieren gelezen worden:
- Indien u zelf een TLS-configuratie ontwerpt, lees dan het hoofdstuk *Wat is Transport Layer Security?*, gevolgd door het hoofdstuk *Gebruiksadvies*. In het hoofdstuk *Gebruiksadvies* wordt u vervolgens doorverwezen naar de relevante passages in het hoofdstuk *Versies, algoritmes en opties*.
- Wilt u weten hoe bepaalde instellingen voor TLS-opties de veiligheid beïnvloeden? Ga dan naar het hoofdstuk *Versies, algoritmes en opties*.
- En wanneer u TLS-configuraties moet beoordelen: lees dan het hoofdstuk *Wat is Transport Layer Security?*, gevolgd door het hoofdstuk *Richtlijnen*. In het hoofdstuk *Richtlijnen* wordt u vervolgens doorverwezen naar de relevante passages in het hoofdstuk *Versies, algoritmes en opties*.

Verwijzingen

In deze publicatie worden meerdere soorten verwijzingen gebruikt:
- De richtlijnen zijn genummerd (bijv. B2-1) en opgenomen in het hoofdstuk *Richtlijnen*.
- Technische termen worden niet altijd bij het eerste gebruik meteen uitgelegd. Wanneer een term op deze manier is gemarkeerd, dan wordt deze in de *Verklarende woordenlijst* aan het eind van deze publicatie beschreven.
- Voor het verstrekken van achtergrondinformatie worden voetnoten[2] gebruikt.
- De *Referenties* aan het eind van deze publicatie vormen de onderbouwing voor het gegeven advies. Indien een bepaalde referentie de onderbouwing vormt voor een advies, wordt de betreffende referentie op de volgende manier aangeduid: (**1**).

2 Op deze manier.

1 Wat is Transport Layer Security?

Transport Layer Security (TLS) is een protocol voor het opzetten en gebruiken van een cryptografisch beveiligde verbinding tussen twee computersystemen: een client en een server. Nadat met het TLS-protocol een beveiligde verbinding tot stand is gebracht, kunnen applicaties de verbinding gebruiken om data uit te wisselen tussen de client en server. TLS wordt in een breed scala van toepassingen gebruikt. Bekende voorbeelden zijn webverkeer (https), e-mailverkeer (IMAP en SMTP na STARTTLS) en bepaalde typen Virtual Private Networks (VPN).

Waarom TLS?

TLS beschermt de communicatie tussen een client en een server. Het beschermen van communicatie is vooral belangrijk als er gevoelige informatie via een verbinding wordt verstuurd. Informatie kan gevoelig zijn vanwege de vertrouwelijkheid (zoals inloggegevens) en vanwege de integriteit (zoals een financiële transactie).
In sommige gevallen is het gebruik van versleutelde verbindingen verplicht. Deze verplichting kan opgenomen zijn in het beleid van een organisatie, maar kan ook in de wet- en regelgeving zijn vastgelegd.

- Op grond van de 'pas-toe-of-leg-uit'-lijst van het Forum Standaardisatie is het gebruik van TLS verplicht voor communicatie tussen onderdelen van de Nederlandse overheid (zoals een veilige uitwisseling van e-mails).[3] Het gebruik van https wordt voor alle overheidswebsites verplicht gesteld.[4]
- De PCI Data Security Standard (PCI DSS) is een beleidsvoorschrift in de financiële sector dat het gebruik van een versleutelde verbinding verplicht stelt, wanneer gegevens van kaarthouders via open, publieke netwerken worden verstuurd.[5]
- De Autoriteit Persoonsgegevens verplicht het gebruik van https voor websites die persoonsgegevens verzamelen.[6] Deze eis vloeit voort uit de Algemene Verordening Gegevensbescherming.

In elk voorbeeld waarborgt het gebruik van TLS dat de verstuurde gegevens tijdens het transport niet door derden bekeken of gewijzigd kunnen worden. Omdat gevoeligheid gebruikersafhankelijk is, is een versleutelde verbinding (en TLS) tegenwoordig in veel omgevingen de regel en niet de uitzondering.

TLS beveiligt echter alleen de inhoud van de communicatie. Informatie over het datatransport wordt niet beschermd. In dat opzicht verschilt TLS van IPsec. TLS werkt op de transport layer, IPsec werkt op de internet layer.[7]

Op dit moment zijn er zeven verschillende TLS-versies. Drie van die versies hebben nog hun oude naam: Secure Sockets Layer (SSL) 1.0, 2.0 en 3.0. Zij zijn ontwikkeld door Netscape. Vervolgens kwamen TLS 1.0, 1.1, 1.2 en 1.3 op de markt. Zij zijn gestandaardiseerd door de Internet Engineering Task Force (IETF). De IETF biedt TLS als een open standaard aan. De meest recente TLS-versie is 1.3.[8]

Een client of server kan meerdere versies van TLS ondersteunen. Die versies zijn echter niet onderling compatibel. Elke versie hanteert zijn eigen opties, bijvoorbeeld op het gebied van bulkversleuteling, authenticatie en sleuteluitwisseling.

3 https://www.forumstandaardisatie.nl/standaard/tls

4 https://www.rijksoverheid.nl/ministeries/ministerie-van-binnenlandse-zaken-en-koninkrijksrelaties/documenten/kamerstukken/2018/10/16/kamerbrief-over-verhogen-informatieveiligheid-bij-de-overheid

5 PCI-DSS v3.2.1, Eis 4, zie https://www.pcisecuritystandards.org

6 https://autoriteitpersoonsgegevens.nl/nl/onderwerpen/avg-europese-privacywetgeving/verantwoordingsplicht#wat-zijn-voorbeelden-van-technische-beveiligingsmaatregelen-6385

7 De transport layer en de internet layer zijn onderdeel van de internet-protocol-suite, een model waarmee het netwerkverkeer beschreven kan worden. Dit model wordt nader toegelicht in RFC 1122, en is beschikbaar via https://datatracker.ietf.org/doc/rfc1122/

8 De specificaties van TLS 1.3 zijn vastgelegd in RFC 8446 en zijn beschikbaar via https://datatracker.ietf.org/doc/rfc8446/

Werking van TLS

Een verbinding tussen een client en server die met TLS beveiligd is, heet een TLS-sessie. Een TLS-sessie bestaat uit twee fasen: de handshake en de applicatiefase. Tijdens de handshake spreken client en server af op welke manier de TLS-sessie wordt opgezet. Tijdens de handshake worden onder andere de volgende zaken overeengekomen:

- Welke versie van TLS wordt er gebruikt?
- Welke sleutel wordt er gebruikt voor de uitwisseling van toekomstige data en hoe wordt die geselecteerd (sleuteluitwisseling)?
- Welk certificaat gebruikt de server om zijn identiteit aan de client te bewijzen?
- Toont de client ook een certificaat? Zo ja, welk?
- Welke cipher suite wordt er gebruikt voor de versleuteling van de data tijdens de applicatiefase?

De handshake wordt geïnitieerd door de client. Tijdens die handshake komen de client en de server vier cryptografische algoritmes overeen: een algoritme voor de sleuteluitwisseling, een algoritme voor digitale handtekeningen, een algoritme voor bulkversleuteling en een algoritme voor hashing. In deze richtlijnen worden deze vier algoritmes samen verder aangeduid als een Algoritmeselectie.[9] Vervolgens controleert de client de authenticiteit van het certificaat dat de server laat zien. Indien de client ook een certificaat gebruikt,[10] wordt de authenticiteit ervan door de server gecontroleerd.

Nadat de handshake is afgerond, begint de applicatiefase. Tijdens de applicatiefase fungeert de TLS-sessie als een beveiligde tunnel voor het dataverkeer. Applicaties kunnen deze tunnel gebruiken om hun eigen dataverkeer te verzenden tussen de client en de server. De applicaties hoeven zich verder niet te bemoeien met de werking van de tunnel: zij kunnen deze gebruiken als abstract communicatiekanaal dat de vertrouwelijkheid en integriteit van de informatie garandeert.

9 Zie het kader *De gewijzigde betekenis van cipher suite in TLS 1.3* in het hoofdstuk *Richtlijnen* voor de reden voor deze benaming.

10 NB: De TLS-versies die ouder zijn dan TLS 1.3, versleutelen de informatie die tijdens de handshake wordt uitgewisseld niet volledig.

Softwarebibliotheken

TLS wordt in veel verschillende applicaties gebruikt. Het programmeren van alle functionaliteiten van TLS is veel werk en vergt specialistische kennis. Daarom bevat de meeste software geen eigen code voor TLS, maar wordt er gebruik gemaakt van een TLS-softwarebibliotheek.

Er zijn verschillende softwarebibliotheken beschikbaar. Sommige bibliotheken zijn vrije software, andere zijn als gesloten product beschikbaar. Ze kunnen in besturingssystemen geïntegreerd worden, maar kunnen ook afzonderlijk worden geleverd. Bekende TLS-bibliotheken zijn o.a. OpenSSL[11], SChannel[12], NSS[13] en mbed TLS[14].

Deze richtlijnen geven geen oordeel over de veiligheid van specifieke TLS-softwarebibliotheken. Alle software bevat bugs, dus ook de TLS-bibliotheken. Bugs kunnen op hun beurt weer tot kwetsbaarheden leiden. Elke bibliotheek heeft haar voor- en nadelen. Overigens zijn niet alle TLS-instellingen in elke bibliotheek beschikbaar.

Stel de leverancier van de TLS-bibliotheek die u gekozen hebt (of de vendor die de bibliotheek in uw systeem heeft geïntegreerd) de volgende vragen om een eerste inzicht te krijgen in de betrouwbaarheid ervan. Beschikt de gekozen TLS-softwarebibliotheek:

- Over een openbaar beleid over de wijze waarop melders kwetsbaarheden kunnen rapporteren?
- Over ontwikkelaars met toegang tot adequate middelen om (veiligheids)ondersteuning te verlenen?
- Over een goede staat van dienst wat betreft het reageren op aanvallen op TLS in het verleden en op kwetsbaarheden in de implementatie van de bibliotheek?
- Over een manier om gebruikers te informeren over veiligheidsupdates, op een manier die duidelijk te onderscheiden is van de reguliere updates?
- Wordt de bibliotheek regelmatig op onafhankelijke wijze gecontroleerd en geëvalueerd?
- Maakt de bibliotheek gebruik van constant time-implementaties om beter bestand te zijn tegen timing side channels?

11 https://www.openssl.org/

12 https://docs.microsoft.com/en-us/windows/desktop/secauthn/secure-channel

13 https://developer.mozilla.org/en-US/docs/Mozilla/Projects/NSS

14 https://tls.mbed.org/

Het NCSC adviseert om bewuste keuzes te maken wat het gebruik van TLS-bibliotheken betreft:

- Gebruik altijd de meest recente versie van de gekozen bibliotheek. De ontwikkelaar heeft bij deze versie namelijk de meeste tijd gehad om kwetsbaarheden te verhelpen.
- Gebruik uitsluitend instellingen die voor uw bedrijfsvoering noodzakelijk zijn. Dit voorkomt dat programmeerfouten in niet-noodzakelijke functionaliteiten tot kwetsbaarheden in het systeem leiden. Zie het hoofdstuk *Gebruiksadvies* voor meer informatie over de keuze van softwarebibliotheken.

Het belang van random numbers

In veel cryptografische toepassingen, waaronder TLS, spelen random numbers (toevalsgetallen) een belangrijke rol. TLS maakt op meerdere plekken in het protocol gebruik van random numbers.

De kwaliteit van de gebruikte random numbers is cruciaal voor de betrouwbaarheid van TLS. Het kiezen van de juiste instellingen voor TLS is belangrijk, maar als er random numbers van een slechte kwaliteit worden gebruikt, kunnen er ondanks die instellingen grote risico's ontstaan.

Alle besturingssysteem en TLS-softwarebibliotheken bevatten methoden om random numbers te genereren. Daarnaast zijn er hardwaremodules verkrijgbaar voor het genereren van random numbers. Dergelijke hardwaremodules produceren sneller random numbers van hogere kwaliteit dan strikt softwarematige methoden.

U kunt voor meer achtergrondinformatie en adviezen over methoden om random numbers te genereren terecht bij het onderdeel *Het genereren van random numbers* in de bijlage *Verdere overwegingen*.

2 Gebruiksadvies

De richtlijnen in dit document zijn bedoeld om een veilige TLS-configuratie te creëren. In de praktijk speelt bij de keuze van een configuratie echter niet alleen de veiligheid een rol. De TLS-configuratie van een server moet ook compatibel zijn met de TLS-configuratie van alle clients die verbinding moeten maken met de server. Daarnaast moeten de configuraties van client en server op bepaalde aspecten overeenkomen.

Bij sommige keuzes kunnen verschillende selecties aan client- en serverzijde op elkaar afgestemd worden. Dit geldt onder andere voor ondersteunde versies, algoritmeselecties en groepen. Om een client en server te laten communiceren, moet er bij elk van deze keuzes minstens één keuze zijn die zowel door de client als de server worden ondersteund. Als de client de versies TLS 1.0, TLS 1.1 en TLS 1.2 ondersteunt, kan deze wel communiceren met een server die de versies TLS 1.0 en TLS 1.1 ondersteunt, maar niet met een server die alleen de versie TLS 1.3 ondersteunt. Ditzelfde principe is ook van toepassing op algoritmeselecties en ondersteunde groepen.

Andere keuzes bepalen hoe lang een bepaalde cryptografische sleutel of andere parameter is. Dat geldt onder andere voor de RSA- en ECDSA-sleutels. Om een client en server te laten communiceren, moeten zowel de client als de server de gekozen lengte van de sleutel of parameter ondersteunen. Oudere software ondersteunt niet altijd sleutels en parameters van voldoende lengte en nieuwere software sluit soms juist sleutels en parameters met een te korte lengte uit.

Tot slot zijn er ook nog opties: configuraties die alleen maar 'aan' of 'uit' kunnen staan. Bijvoorbeeld een server die de TLS-compressie wel of niet ondersteunt. Er zijn geen compatibiliteitsproblemen met clients bekend indien een server aan de hand van de hier beschreven opties wordt geconfigureerd.

Overigens kent TLS, naast de opties die hier worden besproken, nog een stortvloed aan andere opties. Wij bespreken uitsluitend de opties die van invloed zijn op de veiligheid van TLS en die soms niet standaard op veilig staan.

Scenario 1: Controle over client en server

In sommige situaties heeft de partij die zeggenschap heeft over de configuratie van een server, ook zeggenschap over de configuratie van de client. Een voorbeeld hiervan is een webserver waarop een interne webapplicatie wordt gehost. Deze webserver is alleen bereikbaar voor clients van de organisatie zelf. Het NCSC adviseert om in zulke gevallen **Goede** instellingen te gebruiken: dit is de veiligste en meest toekomstbestendige optie. In het hoofdstuk *Versies, algoritmes en opties* wordt beschreven welke instellingen **Goed** zijn.

Stappenplan

1. Inventariseer welke TLS-opties er op de server beschikbaar zijn.
2. Inventariseer de verschillende type clients die verbinding gaan maken met de server.
3. Inventariseer voor elke clienttype welke instellingen er ondersteund worden.[15]
4. Kies **Goede** instellingen voor de volgende opties:
   a. TLS-versie voor de server. Zorg ervoor dat elke client een versie ondersteunt die de server ook ondersteunt.
   b. Algoritmeselecties voor de server. Zorg ervoor dat elke client een algoritmeselectie ondersteunt die de server ook ondersteunt.
   c. Sleutellengtes voor de server. Zorg ervoor dat de gekozen lengte door elke client wordt ondersteund.
   d. Ondersteunde elliptische krommen voor de server. Zorg ervoor dat elke client een elliptische kromme ondersteunt die de server ook ondersteunt.
   e. Finite field-groepen die door de server worden ondersteund, indien oudere clients geen elliptische krommen ondersteunen. Zorg er in dat geval voor dat elke client een **Voldoende** finite field-groep ondersteunt die de server ook ondersteunt.
   f. Overige opties voor de server, tenzij er overtuigende redenen zijn om niet voor **Goede**, maar voor **Voldoende** instellingen te kiezen. Die redenen vloeien voort uit de client-inventarisatie die is uitgevoerd.

15 Een overzicht van TLS-configuraties voor uiteenlopende categorieën clients is beschikbaar via https://www.ssllabs.com/ssltest/clients.html.

5. Ondersteunt de server voor een bepaalde optie geen **Goede** instelling om de clients te ondersteunen? Dan hebt u drie opties:
   a. Vervang de clients dan door een categorie die wel een **Goede** instelling voor deze optie ondersteunt.
   b. Vervang de server door een categorie die wel een **Goede** instelling voor deze optie ondersteunt.
   c. Kies voor de betreffende optie een **Voldoende** instelling die zowel door clients als de server wordt ondersteund.
6. Ondersteunt de server voor een bepaalde optie geen **Goede** en **Voldoende** instelling om de clients te ondersteunen? Dan hebt u drie opties:
   a. Vervang de clients dan door een type die wel (andere) **Goede** of **Voldoende** instellingen voor deze optie ondersteunt.
   b. Vervang de server door een type die wel **Goede** of **Voldoende** instellingen voor deze optie ondersteunt.
   c. Selecteer voor deze optie een **Uit te faseren** instelling die zowel door de clients als de server wordt ondersteund. Deze optie is een laatste redmiddel, omdat hier extra risico's aan verbonden zijn en dit de minst toekomstbestendige configuratie is.
7. Configureer de server met de geselecteerde instellingen. De TLS-configuratie maakt deel uit van de software die de TLS-verbinding gebruikt. Wilt u bijvoorbeeld https aanbieden, dan wordt TLS als onderdeel van de webserversoftware geconfigureerd.
8. Test vervolgens of deze configuratie inderdaad met alle type clients werkt. Ondervindt u compatibiliteitsproblemen? Ga dan terug naar stap 5.
9. Documenteer de gekozen instellingen. Besteed daarbij in ieder geval aandacht aan de volgende punten:
   a. Selecteer en documenteer de voorwaarden voor de geplande beëindiging van het gebruik van alle gekozen **Uit te faseren** instellingen. Zie *Planning van de beëindiging van het gebruik van Uit te faseren configuraties* in het hoofdstuk Richtlijnen, voor enkele voorbeelden van duidelijke voorwaarden voor die beëindiging.
   b. Leg ook de redenen vast waarom er voor **Voldoende** in plaats van **Goede** instellingen is gekozen.

Scenario 2: Alleen controle over de server

In andere situaties heeft de partij die de controle heeft over de configuratie van de server, geen controle over de configuratie van (alle) clients die verbinding maken met de server.[16] Een voorbeeld hiervan is een webserver die een publieke website aanbiedt. Het NCSC adviseert om waar mogelijk voor **Goede** instellingen te kiezen en om die in dit scenario met **Voldoende** instellingen aan te vullen. In sommige gevallen kan het noodzakelijk zijn om ook **Uit te faseren** instellingen te gebruiken in de periode waarin clients bezig zijn met het uitfaseren van deze onveiligere configuraties. In het hoofdstuk *Versies, algoritmes en opties* wordt beschreven welke instellingen respectievelijk **Goed**, **Voldoende** en **Uit te faseren** zijn.

Stappenplan

1. Inventariseer welke categorieën clients verbinding (moeten) maken met de server.[17] Dit is een activiteit die meestal in overleg met de bedrijfseigenaar wordt uitgevoerd.
   a. Maak de afwegingen zichtbaar: het belang van de compatibiliteit versus het risico en de daaraan verbonden ondersteuningskosten van configuraties die steeds kwetsbaarder worden.
2. Inventariseer welke TLS-opties er op de server beschikbaar zijn.
3. Kies **Goede** en **Voldoende** TLS-versies voor de server.
4. Kies **Goede** en **Voldoende** algoritmes voor de server. Zorg voor ondersteuning van een breed scala aan **Voldoende** algoritmes. Deze zijn vaak nodig met het oog op de compatibiliteit met oudere clients. Ondersteun echter niet meer **Voldoende** algoritmes dan nodig zijn om de compatibiliteit te waarborgen.
5. Kies **Voldoende** lengtes van de sleutels. Kies voor **Goede** lengtes wanneer u er zeker bent dat die door alle clients ondersteund worden.
6. Kies **Goede** elliptische krommen voor de server. Zijn er redenen die erop wijzen dat deze krommen niet door alle clients ondersteund worden? Selecteer dan ook **Voldoende** elliptische krommen. Ondersteun echter niet meer krommen dan nodig zijn om de compatibiliteit te waarborgen.
7. Sommige oude clients ondersteunen geen elliptische krommen. Moet u dergelijke clients ondersteunen? Selecteer dan **Voldoende** finite field-groepen. Ondersteun echter niet meer finite field-groepen dan noodzakelijk zijn om de compatibiliteit te waarborgen.

16 Deze kop kan ook gelezen worden als 'Alleen controle over de client', hoewel de clients in deze richtlijnen niet centraal staan. Een voorbeeld hiervan is een e-mailservice die als een TLS-client fungeert bij het verzenden van een e-mail.

17 Idealiter gebeurt dit door gebruik te maken van statistische gegevens over de TLS-verbinding op de server (of een soortgelijke service).

8. Configureer de server met de geselecteerde instellingen. De TLS-configuratie maakt deel uit van de software die de TLS-verbinding gebruikt. Wilt u bijvoorbeeld https aanbieden, dan wordt TLS als onderdeel van de webserversoftware geconfigureerd.
9. Breng de software in kaart die clients vermoedelijk gaan gebruiken om verbinding te maken. Test of clients die de betreffende software gebruiken ook daadwerkelijk verbinding kunnen maken.
10. Hebt u last van compatibiliteitsproblemen?[18] Zoek dan uit door welke opties die problemen veroorzaakt worden.
    a. Meestal zijn dergelijke problemen op te lossen door **Goede** instellingen te vervangen door of aan te vullen met **Voldoende** instellingen.
    b. Indien uit de inventarisatie in de stappen 1 tot en met 9 blijkt dat er ook sprake is van bijzonder oude client-software, moet u wellicht tijdelijk **Uit te faseren** instellingen in uw configuratie opnemen. Van **Uit te faseren** instellingen is bekend dat ze fragiel zijn met het oog op de doorontwikkeling van aanvalstechnieken. Dergelijke instellingen bieden slechts een kleine veiligheidsmarge. Ondersteun niet meer **Uit te faseren** algoritmes dan nodig zijn om de compatibiliteit te waarborgen.
11. Selecteer en documenteer de duidelijke criteria en termijnen voor de geplande vervanging van alle gekozen **Uit te faseren** instellingen. Zie *Planning van de beëindiging van het gebruik van Uit te faseren configuraties* in het hoofdstuk Richtlijnen voor enkele voorbeelden van duidelijke voorwaarden voor die beëindiging. Indien de verwijdering van invloed is op de voorwaarde zoals die in stap 1 is gedefinieerd, moet er doorgaans opnieuw overleg met de bedrijfseigenaar plaatsvinden.

Aandachtspunten

- De richtlijnen in dit document zijn van invloed op de selectie van een certificaatleverancier: niet elke certificaatleverancier kan namelijk elk type certificaat leveren. Bespreek uw TLS-configuratie dan ook met de certificatenbeheerders en Public Key Infrastructures (PKI's) binnen uw organisatie.
- Het controleren van TLS-configuraties kan onderdeel zijn van het kwetsbaarheidsmanagement, penetratietests en de reguliere auditcyclus in uw organisatie. Er bestaan tools en websites waarmee u ook zelf een dergelijke controle kunt uitvoeren.[19] De resultaten van een dergelijke controle kunt u vervolgens vergelijken met de aanbevelingen in deze richtlijnen. Op deze manier kunt u onaangename verrassingen in het kader van penetratietests of audits voorkomen.
- Besturingssystemen beschikken doorgaans over meer dan één TLS-softwarebibliotheek. Zorg ervoor dat u weet welke softwarebibliotheek de serversoftware gebruikt en dat deze continu up-to-date is.

Afwijken van het gebruiksadvies

Het NCSC adviseert om TLS-configuraties te allen tijde op basis van deze richtlijnen samen te stellen. Dit kan echter in sommige situaties niet haalbaar zijn. Houd in dergelijke situaties rekening met de volgende aspecten:

- Voer in ieder geval een risicoanalyse uit wanneer u afwijkt van de richtlijnen. Dat afwijken zal een negatief effect hebben op de beveiliging. Waarom is dat in dit geval aanvaardbaar? Hoe bent u tot die conclusie gekomen? Welke aanvullende maatregelen worden er genomen om de gevolgen van die extra risico's te beperken? Documenteer de afwijkingen en de resultaten van bovenstaande afwegingen.
- Iets is meestal beter dan niets. Zelfs een verbinding die door een **Onvoldoende** TLS-configuratie wordt beschermd, kan voor sommige aanvallers ondoordringbaar zijn. Het feit dat u niet volledig aan de richtlijnen kunt voldoen, kan nooit een reden zijn om helemaal geen TLS te gebruiken.
- Het beoordelen van een TLS-configuratie vereist veel specialistische kennis. Indien u van dit gebruiksadvies afwijkt, verdient het aanbeveling om uw TLS-configuratie en de daaruit voortvloeiende risico's met een expert op dit gebied te bespreken.

18 Maakt uw organisatie gebruik van TLS-interceptie, hetzij lokaal bij de client hetzij op het netwerk? Dan zou dat de oorzaak van het compatibiliteitsprobleem kunnen zijn. In de factsheet "TLS-interceptie" van het NCSC worden overwegingen en randvoorwaarden beschreven voor het gebruik van TLS-interceptie. (https://www.ncsc.nl/actueel/factsheets/factsheet-tls-interceptie.html)

19 Voorbeelden van dergelijke tools zijn testssl.sh (https://testssl.sh/) en sslyze (https://github.com/nabla-c0d3/sslyze). Via de website Internet.nl kunt u online testen of uw webservers en e-mailservers aan deze richtlijnen voldoen (https://www.internet.nl/). Via de website Qualys SSL labs kan een soortgelijke online controle voor webservers worden uitgevoerd (https://www.ssllabs.com/ssltest/).

3 Richtlijnen

In deze richtlijnen wordt regelmatig verwezen naar instellingen die **Goed**, **Voldoende** of **Uit te faseren** zijn. Alle configuraties waarnaar wordt verwezen, zijn opgenomen in het hoofdstuk *Versies, algoritmes en opties*.

Versies

Recente versies van TLS zijn veiliger dan oudere versies. De oudere TLS-versies bevatten kwetsbaarheden die niet kunnen worden gerepareerd. Het gebruik daarvan moet dan ook vermeden worden. Een TLS-configuratie kan meerdere versies ondersteunen.

Nummer	Richtlijn
B1-1	Alle ondersteunde TLS-versies zijn **Goed**, **Voldoende** of **Uit te faseren**
Zie Hoofdstuk 4 – Versies, algoritmes en opties, Tabel 1 – Versies.	

Algoritmeselecties

Voor elke verbinding komen de client en server het gebruik van vier cryptografische algoritmes overeen. Een algoritme voor sleuteluitwisseling, een algoritme voor digitale handtekeningen in de certificaatverificatie, een algoritme voor bulkversleuteling en een algoritme voor hashing. De vier geselecteerde cryptografische algoritmes worden gezamenlijk een algoritmeselectie genoemd. De twee cryptografische algoritmes voor bulkversleuteling en hashing worden gezamenlijk aangeduid met de term cipher suite, die gebruikt wordt voor de bescherming van records.

Voorbeelden van deze algoritmes zijn:
- Certificaatverificatie: RSA, ECDSA, etc.
- Sleuteluitwisseling: ECDHE, DHE, RSA, etc.
- Bulkversleuteling: AES-GCM, ChaCha20-Poly1305, 3DES-CBC, etc.
- Hashing: SHA-1, SHA-256, etc.

De gewijzigde betekenis van *cipher suite* in TLS 1.3

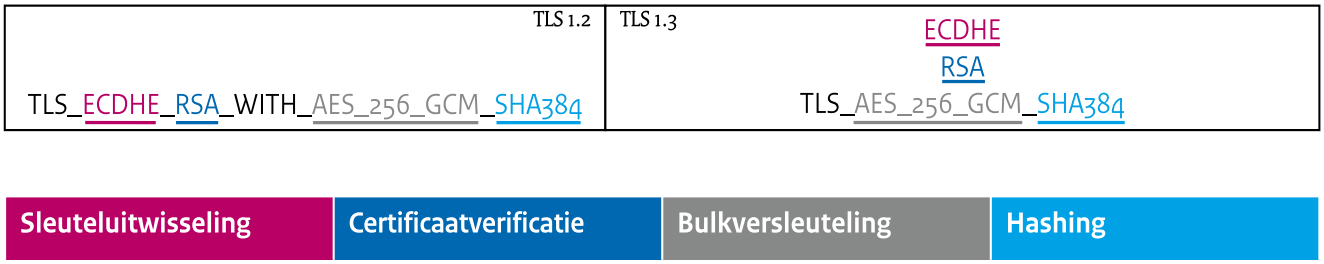
In de vorige versie van deze richtlijnen werd in plaats van algoritmeselectie de term cipher suite gebruikt. Wij hebben die terminologie aangepast in overeenstemming met een wijziging in TLS 1.3.

Tot en met TLS 1.2 bestond een cipher suite ook uit de algoritmes voor sleuteluitwisseling en digitale handtekeningen. Vrijwel alle honderden resulterende combinatiemogelijkheden voor cipher suites zijn opgenomen in het protocolregister.

Om een dergelijk namenexplosie te voorkomen, bestaan de cipher suites in TLS 1.3 alleen nog maar uit de algoritmes voor bulkversleuteling en hashing.

Figuur 1 toont de gewijzigde notatie van cipher suites in TLS 1.2 ten opzichte van TLS 1.3.

Om een verbinding op te zetten onderhandelt TLS over het gebruik van een algoritme voor elk van deze doelen. Er zijn honderden toegestane combinaties beschikbaar. Een TLS-configuratie kan meerdere Algoritmeselecties ondersteunen.



Figuur 1 – Notatie van cipher suites in TLS 1.2 en TLS 1.3. De kleuren geven de verschillende algoritmes en hun functie aan. In TLS 1.3 maken de cryptografische algoritmes voor sleuteluitwisseling en certificaatverificatie geen deel meer uit van de cipher suite.

Nummer	Richtlijn
B2-1	Alle ondersteunde algoritmeselecties bevatten een **Goed**, **Voldoende** of **Uit te faseren** algoritme voor certificaatverificatie.
Zie Hoofdstuk 4 – Versies, algoritmes en opties, Tabel 2 – Algoritmes voor certificaatverificatie.	
B2-2	Alle ondersteunde algoritmeselecties bevatten een **Goed**, **Voldoende** of **Uit te faseren** algoritme voor sleuteluitwisseling.
Zie Hoofdstuk 4 – Versies, algoritmes en opties, Tabel 4 – Algoritmes voor sleuteluitwisselingen.	
B2-3	Alle ondersteunde algoritmeselecties bevatten een **Goed**, **Voldoende** of **Uit te faseren** algoritme voor bulkversleuteling.
Zie Hoofdstuk 4 – Versies, algoritmes en opties, Tabel 6 – Algoritmes voor bulkversleuteling.	
B2-4	Alle ondersteunde algoritmeselecties bevatten een **Goed**, **Voldoende** of **Uit te faseren** algoritme voor hashing.
Zie Hoofdstuk 4 – Versies, algoritmes en opties, Tabel 7 – Hash-functies voor bulkversleuteling en het genereren van random numbers.	
B2-5	De algoritmeselecties worden op basis van de voorgeschreven ordening door de servers gekozen.
Zie Hoofdstuk 4 – Versies, algoritmes en opties, onder *Geef de voorkeur aan snellere en veiligere algoritmes*.	

Certificaten

Via TLS kan de server zijn identiteit aantonen met behulp van een X.509-certificaat. De client kan uitsluitend via een certificaat vaststellen dat hij met de server communiceert en niet met een derde die van plan is om de communicatie af te luisteren of te manipuleren. Het verwerven en beheren van certificaten is geen onderdeel van deze richtlijnen. Voor suggesties op dit gebied verwijzen wij naar het onderdeel *Beheer van certificaten* in de bijlage *Verdere overwegingen*.

Nummer	Richtlijn
B3-1	De server biedt een certificaat aan ter authenticatie.
B3-2	De ondertekende vingerafdruk is gecreëerd middels een **Goed**, **Voldoende** of **Uit te faseren** algoritme voor hashing (zie *Hash-functies voor certificaatverificatie*).
Zie Hoofdstuk 4 – Versies, algoritmes en opties, Tabel 3 – Hash-functies voor certificaatverificatie.	
B3-3	Als de server een certificaat aanbiedt met een RSA-sleutel, is de lengte van deze sleutel **Goed** of **Voldoende**.
Zie Hoofdstuk 4 – Versies, algoritmes en opties, Tabel 8 – Sleutellengte RSA.	
B3-4	Als het aangeboden certificaat niet direct door de root CA is ondertekend, biedt de server tussenliggende CA's aan die het pad authenticeren tussen de root CA en het aangeboden certificaat.

Sleuteluitwisseling

Het algoritme voor sleuteluitwisseling specificeert de wijze waarop een client en een server een sleutel overeenkomen voor een versleutelde communicatie. Ephemeral Diffie-Hellman is een methode om een tijdelijke gedeelde sleutel tussen de partijen te creëren. Diffie-Hellman kent meerdere varianten gebaseerd op finite fields (DHE) en elliptische krommen (ECDHE). Meer informatie over het gebruik van tijdelijke sleutels is te vinden in het gedeelte *Forward Secrecy* in de bijlage *Verdere overwegingen*.

In deze richtlijnen wordt geen minimale omvang voorgeschreven voor de geheime parameter die in een ephemeral Diffie-Hellman wordt gebruikt. Deze parameter is doorgaans 'hard-coded' in de TLS-softwarebibliotheek of afgeleid uit een geselecteerde groep. Het is geen instelling die een beheerder kan configureren.

Nummer	Richtlijn
B4-1	Wanneer er voor de sleuteluitwisseling gebruik wordt gemaakt van DHE, dan is de geheime parameter tijdelijk van aard, willekeurig gekozen uit een uniforme verdeling[20] en van een adequate omvang voor het gekozen finite field.
B4-2	Wanneer er voor de sleuteluitwisseling gebruik wordt gemaakt van ECDHE, dan is de geheime parameter tijdelijk van aard, willekeurig gekozen uit een uniforme verdeling[20] en van een adequate omvang voor de gekozen elliptische kromme.

20 Het NCSC adviseert tegen het gebruik van variaties op het TLS-protocol die de veiligheidsanalyse van TLS 1.3 en haar forward secrecy-eigenschap nietig maken doordat de DH-parameter op de een of andere manier wordt gefixeerd. Een van de voorbeelden op dit gebied ten tijde van het schrijven van deze publicatie is de ETSI 'Enterprise Transport Security (ETS)'- specificatie (ETSI TS 103 523-3), voorheen bekend onder de naam 'eTLS'. De NCSC-factsheet 'TLS-interceptie' bevat afwegingen en randvoorwaarden voor het inzetten van TLS-interceptie door middel van TLS-proxy's.

Elliptische krommen

Van berekeningen met elliptische krommen wordt gezegd dat ze plaats vinden 'op' een elliptische kromme. De kromme vormt de context waarin gerekend wordt. Met het oog op een veilige communicatie moet er een adequate kromme worden gebruikt. Niet alle krommen bieden echter eenzelfde veiligheid.

Nummer	Richtlijn
B5-1	Alle gebruikte elliptische krommen zijn **Goed**, **Voldoende** of **Uit te faseren**.
Zie Hoofdstuk 4 – Versies, algoritmes en opties, Tabel 9 – Ondersteunde Elliptische Krommen.	

NB: Richtlijn B5-1 is zowel van toepassing op een sleuteluitwisseling gebaseerd op ECDHE als op de digitale handtekeningen middels ECDSA en EdDSA.

Finite Fields

Van berekeningen met finite fields wordt gezegd dat ze plaats vinden 'in' een finite field. Het finite field vormt de context waarin gerekend wordt. Met het oog op een veilige berekening moet er een adequaat finite field worden gebruikt. Niet alle finite fields bieden echter eenzelfde veiligheid, interoperabiliteit of efficiëntie.

Nummer	Richtlijn
B6-1	Alle gebruikte finite fields zijn **Goed**, **Voldoende** of **Uit te faseren**.
Zie Hoofdstuk 4 – Versies, algoritmes en opties, Tabel 10 – Ondersteunde finite field-groepen.	

Overige opties

Overigens kent TLS, naast de opties die hier worden besproken, nog een stortvloed aan andere opties. Wij bespreken uitsluitend de opties die van invloed zijn op de veiligheid van TLS en die soms niet standaard op veilig staan.

Compressie

Het gebruik van compressie kan een aanvaller informatie bieden over geheime componenten van een versleutelde communicatie. Omdat de data eerst gecomprimeerd en daarna pas versleuteld worden, kan de mate van compressie informatie verschaffen over de data die verzonden wordt.

Nummer	Richtlijn
B7-1	De instellingen voor compressie zijn **Goed**, **Voldoende** of **Uit te faseren**.
Zie Hoofdstuk 4 – Versies, algoritmes en opties, Tabel 11 – Compressie.	

Renegotiation

In de oudere versies van TLS (voor TLS 1.3) is het tot stand brengen van een nieuwe handshake toegestaan. Dit heet renegotiation (opnieuw onderhandelen).

Nummer	Richtlijn
B8-1	De instellingen voor renegotiation zijn **Goed**, **Voldoende** of **Uit te faseren**.
Zie Hoofdstuk 4 – Versies, algoritmes en opties, Tabel 12 – Insecure renegotiation; en Tabel 13 – Client-initiated renegotation.	

0-RTT

0-RTT is een optie in TLS 1.3 waarmee applicatiedata wordt getransporteerd tijdens het eerste handshake-bericht. 0-RTT biedt echter geen bescherming tegen replay-aanvallen op de TLS-layer en is daardoor moeilijk veilig te gebruiken in een applicatie-agnostische context.

Nummer	Richtlijn
B9-1	De instellingen voor 0-RTT zijn **Goed**, **Voldoende** of **Uit te faseren**.
Zie Hoofdstuk 4 – Versies, algoritmes en opties, Tabel 14 – 0-RTT.	

Planning van de beëindiging van het gebruik van Uit te faseren configuraties

Van **Uit te faseren** instellingen is bekend dat ze fragiel zijn met het oog op de doorontwikkeling van aanvalstechnieken. Dergelijke instellingen bieden slechts een kleine veiligheidsmarge vergeleken met **Voldoende** of **Goede** alternatieven. Daardoor lopen ze een groter risico dat ze in de nabije toekomst de status **Onvoldoende** krijgen.

Het gebruik van **Uit te faseren** instellingen moet onderworpen worden aan schriftelijke voorwaarden voor de beëindiging van het gebruik ervan.

Nummer	Richtlijn
B6-1	Voor alle ondersteunde configuraties die **Uit te faseren** zijn, worden voorwaarden vastgelegd voor de geplande beëindiging van het gebruik ervan.
B6-2	Alle **Uit te faseren** configuraties worden niet meer gebruikt nadat de termijn voor de geplande beëindiging is verstreken.

Het verdient aanbeveling om **Uit te faseren** configuraties te verwijderen zodra de mogelijkheid zich daartoe aandient, omdat verwijderde configuraties niet langer aangevallen of gebruikt kunnen worden om andere TLS-componenten aan te vallen. Tegelijkertijd kan het uit compatibiliteitsoverwegingen voor sommige applicaties noodzakelijk zijn om ze te blijven ondersteunen, totdat de client-ondersteuning verbeterd is.

Het NCSC adviseert om **Uit te faseren** instellingen niet eindeloos te blijven gebruiken.[21] Breng in kaart waarvoor deze **Uit te faseren** instellingen nog gebruikt worden, inclusief duidelijke voorwaarden voor de beëindiging van het gebruik ervan, zodra er aan die voorwaarden wordt voldaan. Dat betekent dat verwijdering gepland wordt op het moment van ingebruikname.

Het kiezen van het moment waarop de ondersteuning wordt beëindigd, is afhankelijk van de betreffende applicatie. Het web-ecosysteem faseert oude TLS-versies en -configuraties eerder uit dan het e-mail-ecosysteem. Deze richtlijnen richten zich niet op afzonderlijke applicaties en bevatten dan ook uitsluitend algemene adviezen.

Hierna treft u een aantal beëindigingsvoorwaarden aan die u schriftelijk kunt vastleggen. **Uit te faseren** configuraties A, B en C worden verwijderd …

- … op <datum>;
- … na de release van <webbrowser> versie X in het 'stabel release channel';
- … wanneer het relatieve aantal gebruikers minder bedraagt dan Y%;
- … wanneer het absolute aantal gebruikers minder bedraagt dan Z per maand.

21 Ondersteuning van **Uit te faseren** (client) software kan ook problematisch zijn om redenen die geen verband houden met TLS: bij dergelijke software is de kans namelijk groter dat deze bekende veiligheidskwetsbaarheden bevat die in latere versies hersteld zijn.

4 Versies, algoritmes en opties

In dit hoofdstuk komen TLS-versies; cryptografische algoritmes; sleutellengtes en keuze van groepen; en opties aan de orde. De veiligheid van een configuratie is afhankelijk van de keuzes die er in deze categorieën wordt gemaakt.

De getallen tussen haakjes verwijzen naar de referenties achterin.

Versies

Recentere versies van TLS zijn veiliger dan oudere versies. De oudste drie versies van TLS (SSL 1.0, SSL 2.0 en SSL 3.0) zijn niet veilig in het gebruik. De beste bescherming wordt geboden door de meest recente versie van TLS: TLS 1.3.

Versie	Status
TLS 1.3	**Goed** (3)
TLS 1.2	
TLS 1.1	**Uit te faseren** (3)
TLS 1.0	
SSL 3.0	**Onvoldoende** (3)
SSL 2.0	
SSL 1.0	

Tabel 1 – Versies

Cryptografische algoritmes

De veiligheid van een TLS-verbinding is afhankelijk van de geconfigureerde algoritmes. De richtlijnen om algoritmeselecties vast te stellen hebben betrekking op vier domeinen:

1. Certificaatverificatie
2. Sleuteluitwisseling
3. Bulkversleuteling
4. Hashing

Aan de eerste drie domeinen wordt in een afzonderlijk gedeelte aandacht besteed. Hashing wordt als een bouwsteen gebruikt en wordt in het kader van de andere drie domeinen nader toegelicht.

Figuur 2 bevat een samenvatting van de algoritmeselecties en hun veiligheidsniveau en toont de gelijkenis tussen algoritmeselectie en de cipher suite-notatie in TLS 1.2 en TLS 1.3. De veiligheidsniveaus zijn als volgt van toepassing op de algoritmeselecties.

Goed, Voldoende en Uit te faseren

Een **Goede** algoritmeselectie bestaat voor alle domeinen uitsluitend uit **Goede** keuzes. **Goede** algoritmes bieden een security-equivalent van ten minste 128 bits. **Goede** algoritmes voldoen per definitie aan de richtlijnen B2-1 tot en met B2-4. Zie de rij met het opschrift **Goed** in Figuur 2 voor voorbeelden van combinaties die tot **Goede** algoritmeselecties leiden.

Een **Voldoende** algoritmeselectie bestaat uit **Voldoende** keuzes en eventueel ook **Goede** keuzes voor alle domeinen. **Voldoende** algoritmes voldoen per definitie aan de richtlijnen B2-1 tot en met B2-4. Zie de rij met het opschrift **Voldoende** in Figuur 2 voor voorbeelden van combinaties die tot **Voldoende** algoritmeselecties leiden (eventueel gecombineerd met keuzes uit de rij **Goed**).

Zowel **Goede** als **Voldoende** algoritmeselecties bevatten uitsluitend algoritmes voor sleuteluitwisseling die voor forward secrecy zorgen.

Een **Uit te faseren** algoritmeselectie bestaat uit **Uit te faseren** keuzes en eventueel ook **Goede** of **Voldoende** keuzes voor alle domeinen. Zie de rij met het opschrift **Uit te faseren** in Figuur 2 voor voorbeelden van combinaties die tot **Uit te faseren** algoritmeselecties leiden (eventueel gecombineerd met keuzes uit de rijen **Goed** of **Voldoende**).

TLS 1.2	TLS 1.3
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	ECDHE RSA TLS_AES_256_GCM_SHA384

	Sleuteluitwisseling	Certificaatverificatie	Bulkversleuteling	Hashing
Goed	ECDHE	ECDSA RSA	AES_256_GCM CHACHA20_POLY1305 AES_128_GCM	(HMAC-)SHA-384 (HMAC-)SHA-256
Voldoende	DHE		AES_256_CBC AES_128_CBC	(HMAC-)SHA-1
Uit te faseren	RSA*		3DES-CBC*	
	* Geschreven als TLS_RSA_WITH_...		* Geschreven als 3DES_EDE_CBC or DES_CBC3	

Figuur 2 – Notatie van cipher suites in TLS 1.2 en TLS 1.3. In de tabel wordt een overzicht gegeven van de algoritmeselecties en hun veiligheidsniveau. De volgende aspecten zijn niet opgenomen in de (oude) cipher suite-notatie: versies, hashfuncties voor certificaatverificatie, hashfuncties voor de sleuteluitwisseling, sleutellengtes en keuze van groepen, en opties. Die aspecten zijn te vinden in de betreffende secties. Wat de volgorde betreft, wordt verwezen naar het gedeelte Geef de voorkeur aan snellere en veiligere algoritmes.

Geef de voorkeur aan snellere en veiligere algoritmes

Het NCSC adviseert om de algoritmeselecties in de navolgende volgorde te configureren. Daarbij gaat de voorkeur uit naar de snelste en veiligste algoritmes. Prefereer **Goede** boven **Voldoende** en dan pas **Uit te faseren** algoritmeselecties. Ga op een bepaald veiligheidsniveau als volgt te werk: kies in de eerste plaats algoritmes die de sleuteluitwisseling uitvoeren op basis van elliptische krommen en als dat niet mogelijk is, dan pas de algoritmes die finite fields gebruiken. Beiden verdienen de voorkeur boven algoritmes voor statische sleuteluitwisseling.
In de tweede plaats verdienen algoritmes die bulkversleutelingen uitvoeren op basis van AEAD-algoritmes de voorkeur boven andere algoritmes (zie *Algoritmes voor bulkversleuteling*). In de derde plaats gaat de voorkeur uit naar algoritmes die de certificaatverificatie uitvoeren op basis van ECDSA boven algoritmes die RSA gebruiken (hoewel dit het gebruik vereist van een certificaat voor een ECDSA-sleutel).[22] In de vierde plaats moeten algoritmes gekozen worden op basis van een aflopende volgorde van hun sleutel en dan pas op basis van hun hash-grootte. Tot slot verdient AES-256 de voorkeur boven ChaCha20.[23]

Algoritmes voor certificaatverificatie

Bij de verificatie van certificaten wordt gebruik gemaakt van digitale handtekeningen. Om de authenticiteit van de verbinding te garanderen, moet een betrouwbaar algoritme voor certificaatverificatie gekozen worden. Het algoritme dat gebruikt wordt om een certificaat te ondertekenen, wordt door de certificaatleverancier geselecteerd.

Het certificaat specificeert het algoritme voor digitale handtekeningen dat tijdens de sleuteluitwisseling door zijn eigenaar wordt gebruikt. Het is mogelijk om meerdere certificaten te configureren, zodat er ook meer dan één algoritme ondersteund kan worden.

Algoritme	Status
ECDSA	**Goed (2; 3)**
RSA	
DSS[24]	**Onvoldoende**
EXPORT-varianten	
PSK	
Anon	
NULL	

Tabel 2 – Algoritmes voor certificaatverificatie

Het algoritme EdDSA is weliswaar **Goed**, maar is (nog) niet goedgekeurd voor gebruik door leveranciers van certificaten (**1**) en is om die reden niet in de tabel opgenomen.

22 ECDSA krijgt vanwege prestatieredenen de voorkeur boven RSA.

23 AES is een ouder algoritme dat langduriger geëvalueerd is door de (wetenschappelijke) cryptologische gemeenschap dan het nieuwere ChaCha20-algoritme. AES biedt ook meer snelheid op platforms die over hardware-acceleratie beschikken. AES is echter minder efficiënt dan ChaCha20 op (mobiele) platforms die niet over die acceleratiemogelijkheid beschikken. Soms wordt er toch voor ChaCha20 boven AES gekozen, omdat men de langere levensduur van de client-batterijen belangrijker vindt dan de serverprestaties.

24 Het algoritme DSS wordt al zeer lange tijd nauwelijks gebruikt. Het is **Onvoldoende**, omdat code die nauwelijks gebruikt wordt, minder vaak getest wordt en dus een groter risico loopt dat het verborgen kwetsbaarheden bevat.

Hashfuncties voor certificaatverificatie

De digitale handtekeningen op certificaten maken gebruik van hashfuncties. Daarbij is de veiligheid van de gekozen hashfunctie van groot belang.

Algoritme	Status
SHA-512	Goed (1; 3)
SHA-384	
SHA-256	
SHA-1	Onvoldoende (1; 3)
MD5	

Tabel 3 – Hashfuncties voor certificaatverificatie

Algoritmes voor sleuteluitwisseling

Een TLS-verbinding begint met een sleuteluitwisseling om een sessiesleutel te genereren. Algoritmes voor een sleuteluitwisseling met forward secrecy waarborgen ook de vertrouwelijkheid van de communicatie in het verleden in situaties waarin de geheime sleutel wordt gecompromitteerd. Algoritmes voor statische sleuteluitwisseling gebruiken de publieke sleutel die in het certificaat geïntegreerd is om een versleuteld exemplaar van de sessiesleutel te transporteren. Sleuteluitwisselingen op basis van ECDHE en DHE bieden forward secrecy. Sleuteluitwisselingen op basis van statisch RSA, ECDH en DH-sleutels in certificaten beschikken niet over die functionaliteit.[25]

Algoritme	Status
ECDHE	Goed (3)
DHE[26]	Voldoende
RSA	Uit te faseren (2; 3)
DH[26]	Onvoldoende
ECDH[27]	
KRB5	
NULL	
PSK	
SRP	

Tabel 4 – Algoritmes voor sleuteluitwisselingen

25 De vertrouwelijkheid bij het gebruik van statisch RSA, ECDH en DH is gebaseerd op het op de lange termijn geheimhouden van de sleutels. Een aanvaller kan een dergelijke langetermijnsleutel in de toekomst stelen en inbreuk maken op de vertrouwelijkheid van het communicatieverkeer in het verleden. Bij ECDHE en DHE worden de sleutels slechts zeer kort bewaard en vervolgens vernietigd, zodat er niets te stelen valt.

26 DHE heeft de status **Voldoende** en niet **Goed**, omdat dit algoritme traag is, wanneer er gebruik wordt gemaakt van sterke parameters.

27 Voor statisch (EC)DH zijn speciale certificaten vereist en daarom wordt dit algoritme zelden gebruikt in TLS. Het gebruik ervan wordt als **Onvoldoende** aangemerkt, omdat code die nauwelijks gebruikt wordt, minder vaak getest wordt en dus een groter risico loopt dat het verborgen kwetsbaarheden bevat.

Gebruik liever ECDHE dan DHE

Er wordt tegenwoordig op grote schaal gebruik gemaakt van cryptografie op basis van *elliptische krommen* (ECC). De snelste keuze voor TLS-servers is cryptografie op basis van elliptische krommen (ECDSA, ECDHE).

Voordat ECC op grote schaal beschikbaar was, kon forward secrecy in TLS alleen maar met het DHE-mechanisme tot stand worden gebracht. Sinds de ontwikkeling van Elliptic Curve Diffie-Hellman Ephemeral (ECDHE –tijdelijke sessiesleutels met Diffie-Hellman op basis van elliptische krommen) is dit niet langer het geval. In deze richtlijnen gaat vanwege prestatieredenen de voorkeur uit naar het gebruik van ECDHE boven DHE.

Hashfuncties voor sleuteluitwisseling[28, 29]

De certificaateigenaar maakt tijdens de sleuteluitwisseling gebruik van een digitale handtekening om het eigenaarschap te bewijzen van de geheime sleutel die bij het certificaat hoort. De eigenaar creëert die digitale handtekening door het ondertekenen van de output van een hashfunctie. In dit verband is het gebruik van een veilige hashfunctie van belang.

SHA2-ondersteuning voor handtekeningen[30]	Status
Ja (ondersteuning van SHA-256, SHA-384 of SHA-512)	Goed (3)
Nee (***geen*** ondersteuning van SHA-256, SHA-384 of SHA-512	Uit te faseren (3)

Tabel 5 – Hashfuncties voor sleuteluitwisseling

Veranderingen in RSA op het gebied van digitale handtekeningen

Het mechanisme voor digitale handtekeningen om het eigenaarschap van een geheime RSA-sleutel te bewijzen, is in TLS 1.3 aangepast. Het oude RSA-PKCS#1 v1.5 padding-mechanisme is vervangen door een modern padding-mechanisme (RSA-PSS). Deze verbetering is met terugwerkende kracht van toepassing: het is verplicht voor servers die zowel TLS 1.2 als TLS 1.3 ondersteunen.
Gebruik de meest recente versie van uw TLS-softwarebibliotheek om deze verbeteringen te implementeren.

28 In dit gedeelte wordt aandacht besteed aan het gebruik van hashfuncties voor key confirmation en handshake-integriteit. Het gebruik van hashfuncties voor het genereren van sleutels wordt nader toegelicht in het gedeelte *Hashfuncties voor bulkversleuteling en het genereren van random numbers*.

29 Dit zijn de meest gangbare algoritmes voor hashing. SHA-224 wordt ook als **Voldoende** aangemerkt, maar dit algoritme wordt niet veel gebruikt.

30 NB: SHA2-ondersteuning voor handtekeningen maakt meestal deel uit van een TLS-softwarebibliotheek en niet van een configuratie. Als uw configuratie het gebruik van SHA2 voor sleuteluitwisseling niet ondersteunt, is het wellicht noodzakelijk om uw softwarebibliotheek te actualiseren.

Algoritmes voor bulkversleuteling[31]

Tijdens de applicatiefase worden (data)records door een symmetrisch encryptie-algoritme in bulkvorm versleuteld. Een symmetrisch encryptie-algoritme bestaat meestal uit een cipher en een operatiemodus. Algoritmes die de authenticatie en encryptie effectief in een operatiemodus integreren, verdienen de voorkeur. Tot deze zogeheten AEAD-algoritmes behoren AES-GCM en ChaCha20-Poly1305. Het meest gebruikte symmetrische encryptie-algoritme is AES. TLS ondersteunt AES met twee sleutellengtes (128 en 256 bits), terwijl ChaCha20-Poly1305 maar één sleutellengte ondersteunt (256 bits).

Algoritme	Status
AES-256-GCM	Goed (3)
ChaCha20-Poly1305	
AES-128-GCM	
AES-256-CBC	Voldoende (2)
AES-128-CBC	
3DES-CBC	Uit te faseren[32] (2; 3)
AES-256-CCM_8[33]	Onvoldoende
AES-128-CCM_8[32]	
IDEA	
DES	
RC4	
NULL	

Tabel 6 – Algoritmes voor bulkversleuteling

Hashfuncties voor bulkversleuteling en het genereren van random numbers

Sommige algoritmes voor bulkversleuteling gebruiken hashfuncties om voor authenticiteit te zorgen (MAC).[34] Daarbij is de veiligheid van de gekozen hashfunctie van groot belang.

TLS gebruikt de geselecteerde hashfunctie ook als een component bij het genereren van random numbers (PRF). In dat verband is de veiligheid van de gekozen hashfunctie dan ook van belang.

Algoritme[35, 36]	Status
HMAC-SHA-512	Goed (2; 3)
HMAC-SHA-384	
HMAC-SHA-256	
HMAC-SHA-1	Voldoende (3)
HMAC-MD5	Onvoldoende (2; 3)

Tabel 7 – Hashfuncties voor bulkversleuteling en het genereren van random numbers

Let op: SHA-1 is uitsluitend **Voldoende** voor bulkversleuteling en als een component bij het genereren van random numbers. Het algoritme is **Onvoldoende** voor gebruik in digitale handtekeningen in certificaten, zoals ook is toegelicht in het gedeelte *Hashfuncties voor certificaatverificatie*. Het gebruik ervan als onderdeel van de sleuteluitwisseling zonder ondersteuning van nieuwere alternatieven wordt eveneens als **Onvoldoende** aangemerkt, zoals beschreven in het gedeelte *Hashfuncties voor sleuteluitwisseling*.

Sleutellengtes en keuze van groepen

Sleutellengte RSA

De veiligheid van RSA voor de versleuteling en digitale handtekeningen houdt verband met de sleutellengte van de publieke sleutel.[37]

Lengte van RSA-sleutels	Status
Minimaal 3072 bit	Goed (2; 3)
2048 – 3071 bit	Voldoende (2)
Minder dan 2048 bit	Onvoldoende

Tabel 8 – Sleutellengte RSA

Ondersteunde elliptische krommen[38]

Niet alle elliptisch krommen zijn geschikt voor gebruik in TLS. De veiligheid van de ECDSA- en EdDSA-digitale-handtekeningen en de ECDHE-sleuteluitwisseling zijn afhankelijk van de geselecteerde kromme. In de tabel zijn de meest gangbare krommen opgenomen.

31 Dit zijn de meest gangbare algoritmes voor bulkversleuteling. Andere **Goede** algoritmes zijn AES-{256,128}-CCM. CAMELLLIA is **Voldoende**. SEED en ARIA maken deel uit van de **Uit te faseren** algoritmes. Deze algoritmes worden zelden gebruikt. Wanneer een systeem gebruikmaakt van deze algoritmes, verdient het aanbeveling om na te gaan of dat echt noodzakelijk is.

32 3DES-CBC is in TLS 1.0 het enige beschikbare algoritme voor bulkversleuteling dat niet **Onvoldoende** is. Wanneer u de ondersteuning van TLS 1.0 beëindigt, moet 3DES-CBC buiten werking worden gesteld.

33 AES-CCM_8 is een variant van AES-CCM met een ingekorte authenticatie-tag die een lager security-equivalent heeft voor de bescherming van de integriteit.

34 AEAD-algoritmes integreren een authenticiteitsmechanisme en gebruiken geen aparte hashfunctie voor de bescherming van records. Wanneer een cipher suite die een AEAD-algoritme bevat, naar een hashfunctie verwijst, wordt die uitsluitend gebruikt als component in het genereren van random numbers.

35 Dit zijn de meest gangbare algoritmes voor hashing. SHA-224 wordt ook als **Voldoende** aangemerkt, maar dit algoritme wordt niet veel gebruikt.

36 In andere documenten kan naar deze algoritmes worden verwezen zonder het voorvoegsel HMAC.

37 Meer specifiek met de publieke modulus, die deel uitmaakt van de publieke sleutel.

38 Dit zijn de meest gangbare elliptische krommen in TLS. Secp521r1 is **Goed**. De krommen brainpoolP512r1, brainpoolP384r1 en brainpoolP256r1 zijn **Voldoende**. Deze elliptische krommen worden in TLS zelden gebruikt en ze zijn om die reden niet opgenomen in de tabel. Wanneer een systeem gebruikmaakt van deze krommen, verdient het aanbeveling om na te gaan of dat echt noodzakelijk is.

Elliptische kromme	Status
secp384r1	Goed (3)
secp256r1	
x448	
x25519	
secp224r1	Uit te faseren (3)
Andere krommen	Onvoldoende

Tabel 9 – Ondersteunde elliptische krommen

Ondersteunde finite field-groepen

De veiligheid van de Diffie-Hellman Ephemeral (DHE) sleuteluitwisseling is afhankelijk van de lengte van de publieke en geheime sleutels die in de geselecteerde finite field-groep wordt gebruikt. De grotere sleutellengtes die noodzakelijk zijn voor het gebruik van DHE gaan ten koste van de prestaties. Maak een zorgvuldige afweging en kies waar mogelijk ECDHE boven DHE.

Gestandaardiseerde finite field-groepen

In deze richtlijnen wordt het gebruik van gestandaardiseerde groepen aanbevolen. Er wordt voor grotere groepen gekozen om het risico van vooraf gemaakte berekeningen door aanvallers te beperken.

Deze conservatieve aanpak zorgt er wel voor dat de serverprestaties nog meer achteruitgaan bij het gebruik van DHE. Maak een zorgvuldige afweging en kies waar mogelijk ECDHE boven DHE.

In het verleden was de complexiteit die verbonden was aan de vrije keuze van finite field-groepen een bron van kwetsbaarheden. Om die complexiteit te verminderen bevat de TLS 1.3-specificatie slechts een beperkt aantal finite field-groepen voor DHE. In deze richtlijnen zijn de **Voldoende** groepen voor TLS beperkt tot de groepen die in TLS 1.3 worden gebruikt (en die in RFC 7919 nader gespecificeerd worden).

Finite field-groep	Status
ffdhe4096 (RFC 7919)	Voldoende[39]
ffdhe3072 (RFC 7919)	
ffdhe2048 (RFC 7919)	Uit te faseren
Overige groepen	Onvoldoende

Tabel 10 – Ondersteunde finite field-groepen

39 Deze groepen hebben uitsluitend de status **Voldoende** in plaats van **Goed**, omdat ze zo traag zijn. Een DHE-sleuteluitwisseling met grotere groepen doet een significant groter beroep op resources dan een security-equivalente ECDHE-uitwisseling. De groepen ffdhe6144 en ffdhe8192 (RFC 7919) zijn weliswaar ook **Voldoende**, maar die zijn zelfs nog trager.

Opties

Compressie

Het gebruik van compressie kan een aanvaller informatie bieden over geheime delen van versleutelde communicatie. Een aanvaller die in staat is om een deel van de verzonden data te achterhalen of te beïnvloeden, kan door middel van een groot aantal verzoeken stukje bij beetje de oorspronkelijke data reconstrueren. Data die meer herhalingen bevatten, zijn beter te comprimeren dan data die geen herhalingen bevatten. Een aanvaller kan zo reconstrueren of een reeds bekend gedeelte vaker voorkomt in de verzonden data. In dergelijke gevallen kan het gebruik van compressie een negatief effect op de veiligheid hebben.

TLS-compressie wordt zo weinig gebruikt dat het geen kwaad kan om deze optie uit te schakelen. Voor applicatiespecifieke compressie ligt dat anders. Bij het http-protocol wordt compressie bijvoorbeeld vaak gebruikt om de beschikbare bandbreedte efficiënter te gebruiken.

Weeg de voors en tegens van het compressiegebruik op applicatieniveau zorgvuldig tegen elkaar af. Wanneer u kiest voor het gebruik van compressie op applicatieniveau, ga dan na of het mogelijk is om hieruit voortvloeiende potentiële applicatie-aanvallen te beperken. Een voorbeeld van een dergelijke maatregel is het beperken van de mate waarin een aanvaller de respons van de server kan beïnvloeden.

Compressie-optie	Status
Geen compressie	Goed
Compressie op applicatieniveau	Voldoende[40]
TLS-compressie	Onvoldoende[41]

Tabel 11 – Compressie

Renegotiation

In de oudere versies van TLS (vóór TLS 1.3) is het tot stand brengen van een nieuwe handshake toegestaan. Dit zogeheten 'opnieuw onderhandelen' (renegotiation) was in het oorspronkelijke ontwerp onveilig. Inmiddels is de standaard gerepareerd en is er een veiliger renegotiation-mechanisme beschikbaar. De oude versie wordt sindsdien als *insecure renegotiation* aangeduid en deze moet derhalve uitgeschakeld worden.

40 Het gebruik van applicatiespecifieke compressie (zoals http-compressie) leidt tot een TLS-configuratie die kwetsbaar is voor BREACH-aanvallen. Het uitschakelen van de applicatiespecifieke compressie kan een negatief effect op de systeemprestaties hebben.

41 Het gebruik van TLS-compressie leidt tot een TLS-configuratie die kwetsbaar is voor CRIME-aanvallen.

In het algemeen is het niet nodig om clients de mogelijkheid te bieden om een renegotiation te initiëren (client-initiated renegotiation). Bovendien komt een server hierdoor binnen een TLS-verbinding bloot te staan aan DoS-aanvallen. Een aanvaller kan ook zonder renegotiation op initiatief van een client soortgelijke DoS-aanvallen uitvoeren door veel parallelle TLS-verbindingen te openen. Dergelijke aanvallen zijn echter eenvoudiger te traceren en met de standaardmaatregelen te migiteren.

Insecure renegotiation	Status
Uit[42] (of n.v.t. voor TLS 1.3)	Goed
Aan	Onvoldoende

Tabel 12 – Insecure renegotiation

Client-initiated renegotiation	Status
Uit (of n.v.t. voor TLS 1.3)	Goed
Aan	Onvoldoende

Tabel 13 – Client-initiated renegotiation

0-RTT

In oude versies van TLS worden minimaal twee 'roundtrips' gemaakt tussen de client en de server voordat applicatiedata verzonden kan worden. Deze overhead wordt in TLS 1.3 gehalveerd, omdat er nog maar één roundtrip nodig is. Met de 0-RTT-optie in TLS 1.3 kan deze overhead nog verder gereduceerd. Door deze optie worden applicatiedata al tijdens het eerste handshake-bericht getransporteerd. Het nadeel is dat 0-RTT echter geen bescherming biedt tegen replay-aanvallen op de TLS-layer en is daardoor moeilijk veilig te gebruiken in een applicatie-agnostische context.

0-RTT	Status
Uit (of n.v.t. voor TLS 1.3)	Goed
Aan	Onvoldoende

Tabel 14 – 0-RTT

OCSP stapling

De TLS-cliënt kan de geldigheid van het X.509-certificaat van de server via het OCSP-protocol controleren bij de certificaatleverancier. Dat OCSP-protocol verschaft de certificaatleverancier informatie over clients die met de betreffende server communiceren. Dit kan echter een privacy-risico vormen. Een server kan de OCSP-informatie ook zelf verstrekken (OCSP stapling). Dit lost niet alleen het privacy-risico op, maar vereist ook geen connectiviteit tussen de client en certificaatleverancier en dit gaat dan ook sneller.

OCSP stapling	Status
Aan	Goed
Uit	Voldoende

Tabel 15 – OCSP stapling

42 Dit wordt soms ook wel 'secure renegotiation' genoemd.

Bijlage A – Verdere overwegingen

Forward secrecy

Forward secrecy is een mechanisme om de vertrouwelijkheid van eerdere TLS-communicatie te blijven waarborgen als de geheime sleutel van een certificaat van een server gestolen wordt. Configuraties die ECDHE of DHE voor de sleuteluitwisseling gebruiken, bieden forward secrecy.

Bij forward secrecy gebruiken client en server niet meteen hun eigen sleutels voor de bulkversleuteling. In plaats daarvan wordt een tweede tijdelijke (ephemeral) sleutel overeengekomen, die alleen voor die sessie geldt. Vervolgens worden alle gebruikte waarden verwijderd. De gebruikte tijdelijke sleutel kan niet afgeleid worden uit de geheime sleutel van het certificaat. Zonder forward secrecy worden de sleutels van de server (die corresponderen met het bijbehorende certificaat) gebruikt om de sessiesleutels meteen uit te wisselen.

Forward secrecy biedt bescherming tegen een aanvalsscenario dat uit twee stappen bestaat. Allereerst moet een aanvaller erin slagen om de door TLS beschermde communicatie 'af te luisteren'. Daarna moet hij de geheime sleutel die correspondeert met de publieke sleutel in het certificaat achterhalen, bijvoorbeeld door hacking of via een gerechtelijk bevel. Met toegang tot die geheime sleutel, kan een aanvaller de sessiesleutel in zijn bezit krijgen en het versleutelde communicatieverkeer ontsleutelen, zodat er sprake is van een schending van de vertrouwelijkheid van die communicatie.

Sessietickets

Bij veel applicaties is het gebruikelijk dat de client en server nadat er een eerste TLS-verbinding tot stand is gebracht, meer verbindingen maken of opnieuw verbindingen tot stand brengen. TLS beschikt over mogelijkheden waarmee een client en server een sessie kunnen hervatten zonder een nieuwe handshake. Zij komen dan meteen in de applicatiefase terecht, zodat de kosten voor het opzetten van een TLS-sessie voor aanvullende verbindingen worden gereduceerd.

Met sessie-ID's slaan zowel de client als server de sessiestatus op via een ID-referentie. De client maakt dat ID kenbaar bij het hervatten van een verbinding. Door het gebruik van dit sessie-ID activeren de client en server als het ware de corresponderende sessiestatus en komen zij meteen in de applicatiefase terecht.

Sessietickets vertonen veel overeenkomsten met sessie-ID's. Een sessieticket is een versleuteld exemplaar van de sessiestatus. Door de client te vragen om een sessieticket 'te bewaren', hoeft de server niet langer voor elke client het sessie-ID en de sessiestatus op te slaan. De client bewaart het sessieticket en toont het aan de server wanneer een verbinding hervat wordt. De server ontsleutelt vervolgens het sessieticket, herstelt de betreffende sessiestatus en de TLS-verbinding komt meteen in de applicatiefase terecht. Daarvoor moet de server wel over een 'encryptiesleutel voor sessietickets' beschikken.

Het ontwerp van die sessietickets is in TLS 1.0, 1.1 en 1.2 fragiel.[43] Een aanvaller die de encryptiesleutel voor sessietickets steelt, kan een passieve ontsleuteling uitvoeren van alle verbindingen die sessietickets uitwisselen of gebruiken. Hierdoor wordt tevens de eerder beschreven forward secrecy-functionaliteit gekraakt.

Deze tekortkomingen zijn in TLS 1.3 gecorrigeerd. De NCSC adviseert organisaties die het hervatten van sessies willen versnellen om TLS 1.3 te gebruiken. Wanneer er in oudere versies van TLS sessietickets worden gebruikt, moet de 'encryptiesleutel voor sessietickets' niet op een schijf worden opgeslagen en moet deze vaak geroteerd worden.

43 Drew Springall, Zakir Durumeric, and J. Alex Halderman. "Measuring the security harm of TLS crypto shortcuts." *Proceedings of the 2016 Internet Measurement Conference*. ACM, 2016. https://jhalderm.com/pub/papers/forward-secrecy-imc16.pdf

Het genereren van random numbers

Om cryptografische algoritmes veilig te gebruiken, is een goede entropiebron en een adequate generator voor random numbers (Pseudo-Random Number Generator, PRNG) nodig. Die entropiebron levert willekeurige data die als input voor de PRNG worden gebruikt. De PRNG zet deze willekeurige data om in uniform verdeelde random numbers. Dat random-vereiste is met name relevant voor (maar niet beperkt tot):

- het genereren van certificaatsleutels; en
- het genereren van tijdelijke sleutels en bewijs van eigendom van de geheime sleutels bij forward secrecy.

Het genereren van voldoende entropie kan een knelpunt vormen indien een TLS-server zwaar wordt belast. Het toevoegen van een hardwaremodule (een hardwarematige Random Number Generator) aan de server zorgt ervoor dat er altijd voldoende entropie beschikbaar is. Veel moderne processoren beschikken over een geïntegreerde hardwaremodule om entropie te verzamelen.
De meeste besturingssystemen en TLS-softwarebibliotheken bevatten een goede RNG. U kunt bij de leverancier van uw TLS-bibliotheek (of bij de vendor die deze bibliotheek in uw systeem integreert) navraag doen over de gebruikte RNG en de bijbehorende entropiebron.

Van de volgende RNG is bekend dat hij onveilig is:

- Dual EC DRBG[44, 45]

Het verdient aanbeveling om na te gaan of uw TLS-softwarebibliotheek deze onveilige RNG gebruikt.

Certificaatbeheer

Het verwerven en beheren van certificaten is geen onderdeel van deze richtlijnen. Een adequaat certificaatbeheer vormt echter wel een belangrijke voorwaarde voor het veilig gebruik van TLS. Daarom geven wij hier een overzicht van een aantal essentiële aandachtspunten. Voor meer instructies kunt u terecht in de NCSC-factsheet 'Veilig beheer van digitale certificaten'.[46]

- **Geheime sleutel genereren** Gebruik een goede RNG om de geheime sleutel te creëren. Zorg ervoor dat u deze sleutel op een betrouwbaar systeem genereert, bijvoorbeeld een Hardware Security Module (HSM) of op een computer die fysiek geen verbinding heeft met het internet. Een sleutel die op een niet-verbonden computer is gegenereerd, wordt vervolgens gebruikt op de server die het certificaat zal aanbieden.
- **Certificaatleverancier** Kies voor het leveren en ondertekenen van het certificaat een betrouwbare leverancier. Organisaties binnen de Nederlandse overheid kunnen gebruikmaken van certificaten die door PKIoverheid worden uitgegeven. Bij sommige applicaties zijn ze daartoe zelfs verplicht.
- **Domeinnamen** Het certificaat bevat een lijst met domeinnamen (Fully Qualified Domain Names, FQDNs) waarop dat certificaat van toepassing is. Zorg ervoor dat het certificaat alle domeinnamen bestrijkt waarvoor het gebruikt wordt, inclusief subdomeinen.
- **Uitgebreide validatie** Veel organisaties die certificaten uitgeven, verstrekken ook EV-certificaten (Extended Validation). Een EV-certificaat geeft wat extra zekerheid over de identiteit van de eigenaar. De ontwikkelaars van client-software hebben de karakteristieke zichtbare verschillen tussen normale en EV-certificaten in de loop der tijd echter verwijderd. EV-certificaten zijn doorgaans duurder dan normale certificaten. Op basis van een risicoanalyse kan bepaald worden of het zinvol is om een EV-certificaat te gebruiken.
- **Bestanden op de server** De beheerder van de server moet zorgen voor de aanwezigheid van tussenliggende CA's tussen de root CA en het uitgegeven certificaat op de server. De server biedt deze tijdens de TLS-verbinding aan. De geheime sleutel van het eigen certificaat moet op adequate wijze beschermd zijn. Een aanvaller die deze geheime sleutel in handen krijgt, kan namelijk het onderschepte communicatieverkeer lezen of manipuleren. Een geheime sleutel kan in een HSM worden opgeslagen. Een HSM is ontworpen om fysieke bescherming te bieden tegen het 'stelen' van een geheime sleutel.
- **Administratie** Houd een administratie bij van alle certificaten die binnen de organisatie worden gebruikt. Zo is het namelijk eenvoudiger om vast te stellen waar een certificaat in gebruik is indien het aan vervanging toe is. Noteer ook de verloopdatum van alle certificaten, zodat deze tijdig vervangen kunnen worden. Er mag nooit gebruik worden gemaakt van verlopen certificaten. Sommige organisaties die certificaten uitgeven, ondersteunen mechanismen voor het automatisch vervangen en uitrollen van certificaten, wat het risico op menselijke fouten kan reduceren.

Waar eindigt een TLS-verbinding?

Het model waarin een client verbinding maakt met een server, komt niet overeen met de configuratie die door veel organisaties wordt gebruikt. Het ontsleutelen van TLS-verkeer kan bijvoorbeeld gecentraliseerd plaatsvinden, waarna het verder binnen het interne netwerk wordt verstuurd. Deze opzet biedt de mogelijkheid om het netwerkverkeer ook achteraf te verwerken. Hou er bij een dergelijke opzet wel rekening mee dat TLS het verkeer slechts beschermt tot het punt waar de verbinding eindigt. Indien de geheimhouding en integriteit binnen uw organisatie ook na dit punt gewaarborgd moeten blijven, moet u aanvullende maatregelen nemen. Een mogelijk oplossing is om voor dit laatste gedeelte een nieuwe TLS-sessie te gebruiken.

44 https://csrc.nist.gov/publications/nistbul/itlbul2013_09_supplemental.pdf
45 https://projectbullrun.org/dual-ec/
46 https://www.ncsc.nl/dienstverlening/expertise-advies/kennisdeling/factsheets/factsheet-veilig-beheer-van-digitale-certificaten.html

Sommige organisaties die anti-DDoS-diensten aanbieden, vragen om de geheime sleutels van certificaten die in gebruik zijn voor TLS. Zij kunnen dan vervolgens uw verbinding beëindigen en filteren. Indien u van deze diensten gebruikmaakt, moet u niet eenvoudigweg uw geheime sleutel beschikbaar stellen. Dit kan namelijk in strijd zijn met interne beleid of de (sectorale) wet- en regelgeving. Overweeg in een dergelijke situatie om een leverancier te kiezen die deze informatie over uw geheime sleutels niet nodig heeft.[47] Breng de risico's in kaart die voortvloeien uit het bekend maken van uw geheime sleutels. Neem contractuele maatregelen ter compensatie van de verminderde technische controle en controleer regelmatig in welke mate de leverancier de afgesproken maatregelen ook in praktijk brengt.

Postkwantumveiligheid

Normaal gebruik van TLS biedt geen postkwantumveiligheid.[48] Het is echter wel mogelijk om TLS zodanig te configureren dat er sprake is van een beperkte[49] mate van postkwantumveiligheid. Het NCSC adviseert om specialistische ondersteuning in te schakelen in gebruikssituaties waarin aan deze veiligheidseis voldaan moet worden. Wat de Nederlandse overheid betreft, kan voor die ondersteuning een beroep worden gedaan op het Nationaal Bureau voor Verbindingsbeveiliging (NBV).

Authenticatie van clients met certificaten

TLS ondersteunt de wederzijdse authenticatie met certificaten. Bij die wederzijdse authenticatie gebruikt de client een certificaat om zichzelf bij de server te authenticeren, terwijl ook de server een certificaat gebruikt om zichzelf bij de client te authenticeren.

De certificaten van clients bevatten vaak gevoelige informatie, zoals persoonsgegevens. Een voorbeeld hiervan is de naam van de gebruiker. Vóór TLS 1.3 werden certificaten onversleuteld verstuurd als onderdeel van de handshake. Wanneer u certificaten met gevoelige gegevens voor de authenticatie gebruikt en u voor de geheimhouding op TLS vertrouwt, verdient het aanbeveling om TLS 1.3 te gebruiken.

Certificate pinning en DANE

Een client die een TLS-sessie met een server opzet, controleert het X. 509-certificaat van de server. De cliënt controleert de keten van digitale handtekeningen die het certificaat met het root CA verbindt. Dit PKI-systeem is echter kwetsbaar, omdat de meeste software honderden rootcertificaten vertrouwt. Indien een certificaatleverancier valse certificaten uitgeeft, komt de integriteit van het hele systeem in gevaar.

Hebt u de controle over de software van zowel de client als de server? Dan kunt u door middel van certificate pinning vastleggen welk/welke certificaat/certificaten de client moet accepteren. De client hoeft daardoor niet meer de gehele handtekeningenketen te controleren: een certificaat wordt herkend of niet. Een gecompromitteerde certificaatautoriteit vormt nu ook geen risico meer voor de verbinding. Daarnaast kan het gebruik van certificaten van een bepaalde certificaatautoriteit afgedwongen worden door het pinnen van het tussenliggende of het rootcertificaat die gebruikt worden voor de uitgifte van het certificaat. Op deze manier wordt het risico op het compromitterende acties van andere certificaatautoriteiten uitgesloten. De verbinding tussen een app op een mobiel platform en een server is een situatie waarin certificate pinning effectief kan zijn.

DNS-based Authentication of Named Entities (DANE) is een techniek waarmee clients een certificaat kunnen authenticeren op basis van het Domain Name System (DNS). De beheerder van een certificaat publiceert informatie over dat certificaat in een speciaal DNS-record, een TLSA-record. Cliënten kunnen de authenticiteit van het certificaat nu niet alleen via de PKI controleren, maar ook via het TLSA-record. NB: het traditionele DNS-systeem is niet betrouwbaar genoeg voor een veilig gebruik van DANE. Daarvoor moet DNSSEC worden gebruikt. Een voorbeeld van een DANE-toepassing is de authenticatie van TLS-verbindingen tussen e-mailservers[50]

47 Een voorbeeld van een methode om dit te bewerkstelligen is te vinden op: https://blog.cloudflare.com/keyless-ssl-the-nitty-gritty-technical-details/

48 Meer informatie over de gevolgen van kwantumcomputers voor organisaties is te vinden in de NCSC-factsheet "Postkwantumcryptografie". (https://www.ncsc.nl/actueel/factsheets/factsheet-postkwantumcryptografie.html)

49 Ten tijde van het schrijven van deze publicatie zijn dergelijke configuraties nog niet voorzien van postkwantum forward secrecy.

50 Meer informatie over DANE en het gebruik ervan is te vinden in de NCSC-factsheet 'Beveilig verbindingen van mailservers'. (https://www.ncsc.nl/actueel/factsheets/factsheet-beveilig-verbindingen-van-mailservers.html)

Bijlage B – Wijziging van deze richtlijnen

Deze richtlijnen zullen worden aangepast aan nieuwe versies van TLS en aan nieuwe inzichten over de (on)veiligheid van bepaalde configuraties. De veiligheid van TLS is een voortdurend onderwerp van onderzoek. De komende jaren zullen er dan ook nog meer kwetsbaarheden aan het licht komen. Daarnaast zullen er nieuwe TLS-versies of TLS-configuraties worden gestandaardiseerd.

Validiteit

De laatste versie van deze richtlijnen is altijd te vinden op de website van het NCSC. Het NCSC evalueert de validiteit van zijn adviezen op een periodieke basis. Deze richtlijnen kennen geen verloopdatum en zijn geldig totdat er een update is verschenen.

Acute wijzigingen

Indien er een acute wijziging van deze richtlijnen nodig is, dan zal deze worden uitgebracht als addendum bij de meest recente versie van de richtlijnen. Een dergelijke situatie kan zich bijvoorbeeld voordoen als uit onderzoek blijkt dat bepaalde TLS-configuraties niet meer veilig zijn.

Een addendum wordt op de website van het NCSC gepubliceerd. Ook de partners van het NCSC worden hiervan op de hoogte gesteld. Daarnaast zal de publicatie van een addendum eveneens worden aangekondigd via het twitteraccount van het NCSC (@ncsc_nl) of via een op dat moment passend ander communicatiekanaal.

Nieuwe versies

Grotere wijzigingen worden in nieuwe versies van deze richtlijnen doorgevoerd. Een nieuwe versie van de richtlijnen bevat ook de informatie die in eerdere addenda is uitgebracht. Nieuwe versies worden op dezelfde manier verspreid als de addenda: ze worden op de website van het NCSC gepubliceerd, ze worden aan de NCSC-partners verstuurd en de publicatie ervan wordt via het twitter-account van het NCSC aangekondigd.

Bijlage C – Lijst met cipher suites

Zie Figuur 1 in het hoofdstuk *Gebruiksadvies* voor een uitleg over de cipher suite-notatie in TLS.

In de cipher suite-notatie missen: *versies; hash-functies voor certificaat-verificatie; hash-functies voor sleuteluitwisseling; sleutellengte en keuze van groepen* en *opties*. Hiervoor verwijzen we naar de gelijknamige gedeelten in het hoofdstuk *Versies, algoritmes en opties*. Zie voor de ordening van cipher suites het onderdeel *Geeft de voorkeur aan snellere en veiligere algoritmes* in hetzelfde hoofdstuk.

Goed

TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

Voldoende

TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
TLS_DHE_RSA_WITH_CHACHA20_POLY1305_SHA256
TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
TLS_DHE_RSA_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
TLS_DHE_RSA_WITH_AES_128_CBC_SHA

Uit te faseren

TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA
TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_AES_256_GCM_SHA384
TLS_RSA_WITH_AES_128_GCM_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA

Bijlage D – Verklarende woordenlijst

Verklarende woordenlijst	
3DES	Zie Bulkversleuteling
AEAD	Dit zijn algoritmes voor bulkversleuteling met Authenticated Encryption with Associated Data (AEAD): nauwe integratie tussen authenticatie en versleuteling. Tot de veel gebruikte AEAD-algoritmes voor bulkversleuteling behoren AES-GCM en ChaCha20-POLY1305.
AES	Zie Bulkversleuteling
Algoritmeselectie	Een Algoritmeselectie is een combinatie van vier elementen en bestaat uit een cryptografisch algoritme voor sleuteluitwisseling, een cryptografisch algoritme voor digitale handtekeningen, een cryptografisch algoritme voor bulkversleuteling en een cryptografisch algoritme voor hashing. In TLS 1.3 wordt een combinatie die alleen uit de laatste twee elementen bestaat, aangeduid als een cipher suite. Vóór TLS 1.3 verwees de term cipher suite naar wat in deze richtlijnen een Algoritmeselectie wordt genoemd. Bij het gebruik van TLS komen de client en server een Algoritmeselectie overeen die in de daaropvolgende versleutelde communicatie wordt gebruikt. Zie ook sleuteluitwisseling, digitale handtekening, bulkversleuteling, hash-functie en cipher suite.
Bulkversleuteling	Bulkversleuteling is het proces waarbij data tijdens de applicatiefase worden versleuteld met behulp van de tijdens de sleuteluitwisseling overeengekomen sleutel. De versleuteling vindt plaats met een algoritme voor symmetrische encryptie. Bekende voorbeelden van zulke algoritmes zijn AES-GCM, CHACHA20 en 3DES-CBC.
CA	Zie Certificaatverificatie
CAMELLIA	Zie Bulkversleuteling
CBC	Zie Operatiemodus
CCM	Zie Operatiemodus
Certificaat	Zie Certificaatverificatie
Certificaatverificatie	De server biedt de cliënt tijdens een TLS-sessie een certificaat aan. Dit certificaat is digitaal ondertekend door een certificaatautoriteit (CA). De certificaatautoriteit is een partij waarin de client vertrouwen stelt. De client controleert de digitale handtekening van de certificaatautoriteit. Zo wordt vastgesteld of het certificaat inderdaad door de certificaatautoriteit is uitgegeven. Het algoritme voor certificaatverificatie is het algoritme dat de certificaatautoriteit gebruikt om zijn digitale handtekening te plaatsen. Bekende voorbeelden zijn RSA en ECDSA.
ChaCha20-Poly1305	Zie Bulkversleuteling
Cipher suite	Een cipher suite bevat een algoritme voor bulkversleuteling en een algoritme voor hashing. Vóór TLS 1.3 bevatten cipher suites ook nog de algoritmes voor sleuteluitwisseling en digitale handtekeningen. In deze richtlijnen wordt gebruik gemaakt van de nieuwe, beperktere TLS 1.3-definitie van cipher suites. Zie Algoritmeselectie.
DANE	DNS-based Authentication of Named Entities (DANE) is een techniek waarmee clients een certificaat kunnen authenticeren op basis van het Domain Name System (DNS).

Verklarende woordenlijst	
(D)DoS-aanval	Een Denial of Service (DoS)-aanval is een aanval waarbij een computer met een stortvloed aan verzoeken buiten werking wordt gesteld. De verzoeken zijn afkomstig van één enkel computersysteem. Bij een Distributed Denial of Service (DDoS)-aanval zijn de verzoeken niet van één computersysteem afkomstig, maar van een groot aantal computersystemen.
DES	Zie Bulkversleuteling
DHE	Zie Sleuteluitwisseling
Diffie-Hellman	Zie Sleuteluitwisseling
DNS	Het Domain Name System (DNS) is een gedistribueerd systeem voor het beantwoorden van informatieverzoeken en vragen over domeinnamen. Een typisch vraag kan bijvoorbeeld zijn wat het IP-adres van een computer met een bepaalde domeinnaam is, of welke computer de e-mail voor een bepaalde domeinnaam afhandelt. DNS Security Extensions (DNSSEC) kan de betrouwbaarheid van de informatie in DNS verbeteren. DNSSEC zorgt er daarnaast voor dat DNS voor nieuwe toepassingen gebruikt kan worden, zoals DANE.
DNSSEC	Zie DNS
DSS	Zie Certificaatverificatie
ECC	Zie Elliptische Kromme
ECDHE	Zie Sleuteluitwisseling
ECDSA	Zie Certificaatverificatie
EdDSA	Zie Certificaatverificatie
Elliptische Kromme	Zie Mathematische structuur
Finite Field	Zie Mathematische structuur
Forward Secrecy	Forward secrecy is een mechanisme om de vertrouwelijkheid van eerdere TLS-communicatie te blijven waarborgen als de geheime sleutel van een certificaat gestolen wordt. Cipher suites die sleuteluitwisselingen gebruiken op basis van ECDHE of DHE bieden forward secrecy.
GCM	Zie Operatiemodus
Geheime sleutel	Zie sleutel
Handshake	De handshake is de fase van het TLS-protocol waarin de cliënt en server de wijze overeenkomen waarop ze gegevens uit gaan wisselen. Na de handshake volgt de applicatiefase, waarin cliënt en server versleutelde gegevens uitwisselen.
Hash-functie	Een hash-functie is een wiskundige functie die inputgegevens tot een digitale vingerafdruk verhaspelt. In het algemeen is die input niet meer uit het resultaat te herleiden. Hash-functies worden in TLS gebruikt als een component in digitale handtekeningen, voor het genereren van random numbers (PRF) en voor bulkversleuteling (MAC). Voorbeelden van hash-functies zijn MD5, SHA-1 en SHA-256.
Hashing	Zie Hash-functie
https	HTTP Secure (https) is een protocol dat een TLS-sessie opzet die vervolgens gebruikt wordt om HTTP-verkeer uit te wisselen. Communicatie met een webserver is op die manier niet 'af te luisteren' of te manipuleren.
IETF	De Internet Engineering Task Force (IETF) is een organisatie die verantwoordelijk is voor het ontwikkelen van internetstandaarden. Deze internetstandaarden worden gedocumenteerd in zogeheten Requests For Comments (RFC's). De IETF heeft geen bevoegdheid om het gebruik van de ontwikkelde standaarden te verplichten.
Mathematische structuur	Elliptische Krommen en Finite Fields (eindige lichamen) zijn mathematische structuren die voor berekeningen gebruikt kunnen worden. Een ander voorbeeld van zo'n mathematische structuur is de verzameling gehele getallen. Elliptische Krommen kunnen voor cryptografie worden gebruikt, de zogeheten Elliptic Curve Cryptography (ECC). EdDSA, ECDSA en ECDHE zijn op ECC gebaseerde algoritmes. Finite fields kunnen eveneens voor cryptografie worden gebruikt. DHE is een algoritme dat gebaseerd is op finite field-cryptografie.
MD5	Zie Hash-functie

Verklarende woordenlijst	
Operatiemodus	Een algoritme voor bulkversleuteling kan zowel gebruikmaken van datablokken (block cipher) als van datastromen (stream cipher). Bij gebruik van een block cipher moeten de versleutelde blokken op een veilige manier worden samengevoegd. De operatiemodus heeft betrekking op de wijze waarop deze blokken samengevoegd worden. Voorbeelden van operatiemodi zijn CBC en GCM.
PKI	Zie Public Key Infrastructure
Publieke sleutel	Zie sleutel
Public Key Infrastructure	Een Public Key Infrastructure (PKI) is een hiërarchische ordening van certificaten waarbij de hogere certificaten de authenticiteit van de lagere certificaten bevestigen met een digitale handtekening. Indien een cliënt de hoogste certificaten in de PKI vertrouwt, dan kan hij de lagere certificaten ook vertrouwen door de tussenliggende digitale handtekeningen te controleren. De certificaten die een certificaatautoriteit uitgeeft, vormen samen met het rootcertificaat een PKI.
RSA	RSA is een algoritme voor sleuteluitwisseling en certificaatverificatie. Zie beide onderwerpen.
Security-equivalent	Het security-equivalent is een maat om de cryptografische sterkte van versleutelingssystemen te vergelijken. Het security-equivalent wordt uitgedrukt in bit. De sterkte van een versleutelingssysteem is afhankelijk van het gebruikte algoritme, de sleutellengte en de stand van zaken wat de aanvalstechnieken betreft. Bijvoorbeeld: ECDSA met een sleutellengte van 256 bits en AES met een sleutellengte van 128 bits hebben allebei een security-equivalent van 128 bits op basis van het huidige inzicht in cryptologische aanvallen op deze algoritmes.
SHA-1	Zie Hash-functie
SHA-256, SHA-384, SHA-512	Zie Hash-functie
Sleutel	Een sleutel is een bepaalde hoeveelheid geheime data waarmee cryptografische berekeningen uitgevoerd kunnen worden. Versleutelde gegevens kunnen met behulp van de bijbehorende sleutel ontsleuteld worden. Bij symmetrische algoritmes voor versleuteling is de volledige sleutel geheim. Bij asymmetrische algoritmes voor versleuteling bestaat de sleutel uit twee delen, een publiek deel en een geheim deel. Het publieke deel van de sleutel heet ook wel de publieke sleutel. Dit deel is dus niet geheim. Het geheime deel van de sleutel heet de geheime sleutel.
Sleuteluitwisseling	In een TLS-sessie hebben de cliënt en server een sleutel nodig om met de bulkversleuteling van data te starten. Het uitwisselen van een sleutel gebeurt met behulp van een algoritme voor sleuteluitwisseling. Hiervoor is een speciaal algoritme nodig, omdat de verbinding tijdens de handshake nog niet versleuteld is. Voorbeelden van algoritmes voor sleuteluitwisseling zijn DHE, ECDHE en RSA.
Softwarebibliotheek	Een softwarebibliotheek bestaat uit software die bepaalde functionaliteiten beschikbaar stelt voor programmeurs van andere software. Door het gebruik van een softwarebibliotheek kan een programmeur voortbouwen op het werk van anderen. Op die manier hoeft die programmeur niet zelf alle functionaliteiten helemaal opnieuw te ontwikkelen. Het gebruik van TLS in software vind doorgaans plaats aan de hand van een softwarebibliotheek.
SSL	Secure Sockets Layer (SSL) is de oude naam voor Transport Layer Security (TLS). Hoewel TLS al sinds versie TLS 1.0 (1999) geen SSL meer heet, wordt de naam nog steeds veel gebruikt.
VPN	Een Virtual Private Network (VPN) is een netwerk dat bestaat uit computers die onderling verbonden zijn via niet-vertrouwde verbindingen. Door het toepassen van versleuteling kunnen de computers onderling toch op een vertrouwelijke wijze gegevens uitwisselen.

Referenties

1. **CA/Browser forum.** CA/Browser Forum Baseline Requirements. *CA-Browser Forum BR 1.6.2.* [Online] Januari 2019. https://cabforum.org/baseline-requirements-documents/
2. **Federal Office for Information Security (BSI).** Cryptographic Mechanisms: Recommendations and Key Lengths. *BSI TR-02102-1 v2018-02*, [Online] Mei 2018. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.pdf?__blob=publicationFile&v=8
3. **ECRYPT-CSA.** Algorithms, Key Size and Protocols Report (2018), *H2020-ICT-2014 – Project 645421, D5.4*, [Online] Februari 2018. http://www.ecrypt.eu.org/csa/documents/D5.4-FinalAlgKeySizeProt.pdf

Colofon

Publicatie
Nationaal Cyber Security Centrum (NCSC)
Postbus 117, 2501 CC Den Haag
Turfmarkt 147, 2511 DP Den Haag
070 751 55 55

Meer informatie
www.ncsc.nl
info@ncsc.nl
@ncsc_nl

versie 2.0 | april 2019