

BEVEILIGINGSEISEN WEBDIENSTEN

Gemeente 's-Hertogenbosch



Versiebeheer:

Datum	Versie	Toelichting
29-07-2026	0.8	Conceptversie
03-09-2026	0.9	Opmerkingen verwerkt. Tekst aangescherpt en definities robuuster omschreven.
08-09-2026	1.0	Definitief

INHOUDSOPGAVE

1.	Inleiding	3
1.1.	Doel en reikwijdte	3
1.2.	Beheer	3
2.	Transport Layer Security.....	4
2.1.	Checklist Transport Layer Security	4
3.	Cryptografie en sleuteluitwisseling	5
3.1.	Checklist cryptografie en sleuteluitwisseling	5
4.	PKI, sleutels en certificaten	6
4.1.	Checklist PKI, sleutels en certificaten	6
5.	HTTP security headers	7
5.1.	Checklist HTTP security headers.....	8
6.	Same-Origin Policy.....	9
6.1.	Checklist Same-Origin Policy	9
7.	API-specifieke beveiligingsmaatregelen	10
7.1.	Checklist API-specifieke beveiligingsmaatregelen.....	10
8.	Cookies en sessiebeveiliging.....	11
8.1.	Checklist cookies en sessiebeveiliging.....	11
9.	Security.txt.....	12
9.1.	Checklist security.txt.....	12
10.	Resource Public Key Infrastructure	13
10.1.	Checklist Resource Public Key Infrastructure	13
11.	Niet-productieomgevingen	14
11.1.	Checklist niet-productieomgevingen.....	14
12.	Algemeen.....	15
12.1.	Checklist algemeen.....	15

1. Inleiding

Gemeente 's-Hertogenbosch maakt intensief gebruik van webdiensten, waaronder websites, webportalen en API-gebaseerd berichtenverkeer. Dit document — de vervanger van *Gebruik van TLS en HTTP response headers* — beschrijft de vereisten die door de gemeente zijn opgesteld in relatie tot transportbeveiliging (TLS), HTTP security headers, API-specifieke en overige relevante beveiligingsmaatregelen. Als niet aan de in dit document genoemde eisen kan worden voldaan, wordt via het intakeproces van de gemeente bepaald of, en waar van toepassing voor welke termijn, akkoord kan worden gegaan met eventuele uitzonderingen.

1.1. Doel en reikwijdte

De in dit document genoemde maatregelen gelden voor alle webdiensten waarvoor gemeente 's-Hertogenbosch enige vorm van verantwoordelijkheid of aansprakelijkheid draagt. Bovendien dienen de genoemde toegepaste instellingen individueel, maar ook als geheel resultaat te hebben dat deze een optimale beveiliging realiseren die de goede werking van de applicatie niet in de weg staat. Applicatiebeveiliging zelf valt buiten de scope van dit document.

1.2. Beheer

DB/ICT en DB/ICT/INFRA waarborgen dat dit document periodiek wordt geëvalueerd en waar nodig wordt herzien of aangepast op maatschappelijke en technologische ontwikkelingen.

2. Transport Layer Security

Transport Layer Security (TLS) is een gestandaardiseerd protocol voor het opzetten en gebruiken van een versleutelde verbinding tussen twee computersystemen of applicaties. De beveiligde verbinding garandeert de vertrouwelijkheid en integriteit van data die uitgewisseld wordt tussen client en server. Het gebruik van TLS is verplicht in de volgende gradatie: TLS 1.0/1.1 zijn niet toegestaan, TLS 1.3 moet worden aangeboden. TLS 1.2 mag uitsluitend worden aangeboden wanneer dit aantoonbaar noodzakelijk is voor compatibiliteit en moet in dat geval sterk worden geconfigureerd. TLS 1.3-only geniet de voorkeur. Wanneer gebruik wordt gemaakt van een niet-beveiligde verbinding is deze alleen bedoeld om te verwijzen naar de met TLS beveiligde variant.

TLS-compressie, renegotiation en 0-RTT

TLS-compressie is een techniek waarmee gegevens in een compactere vorm worden overgedragen, waardoor minder communicatieoverhead ontstaat. TLS-compressie wordt niet gebruikt om via methodes zoals CRIME het lekken van gevoelige data tegen te gaan.

Renegotiation houdt in dat client en server tijdens de applicatie-fase een nieuwe handshake-fase opstarten. Renegotiation wordt sinds TLS 1.3 om veiligheids- en performanceredenen niet meer ondersteund.

Zero Round-Trip Time (0-RTT) is een optie die geïntroduceerd is in TLS 1.3. Het doel van 0-RTT is om client en server (nog) sneller hun sessie te laten hervatten. 0-RTT wordt niet gebruikt omdat een goede configuratie ervan niet eenvoudig is en het de nodige veiligheidsrisico's met zich meebrengt.

AEAD-algoritmes

Met bulkversleuteling wordt de data versleuteld door een symmetrisch versleutelingsalgoritme. Gebruik de algoritmen AES-GCM en ChaCha20-Poly1305, zodat authenticatie en encryptie effectief zijn geïntegreerd in de operatiemodus (AEAD).

SSL Labs

Om te testen of de configuratie van TLS voldoet, kan gebruik worden gemaakt van een dienst van SSL Labs: <https://www.ssllabs.com/ssltest>. Let op: Hierbij is de score een indicatie. De minimale verplichte score kan in de praktijk mogelijk niet gelijk staan aan de vereiste en juiste configuratie.

2.1. Checklist Transport Layer Security

- ☐ TLS 1.3 wordt aangeboden (verplicht);
- ☐ Als TLS 1.2 wordt aangeboden, is deze sterk geconfigureerd (verplicht);
- ☐ TLS 1.0 en TLS 1.1 worden niet aangeboden (verplicht);
- ☐ Er wordt geen gebruik gemaakt van TLS-compressie (verplicht);
- ☐ Er wordt geen gebruik gemaakt van renegotiation (verplicht);
- ☐ Er wordt geen gebruik gemaakt van 0-RTT (verplicht);
- ☐ Er wordt alleen gebruik gemaakt van AEAD-ciphers: AES-GCM, ChaCha20-Poly1305 (verplicht);
- ☐ De score bij SSL Labs is minimaal "A" (verplicht).

3. Cryptografie en sleuteluitwisseling

Het sleuteluitwisselingsalgoritme bepaalt hoe client en server gezamenlijk cryptografisch sleutelmateriaal afleiden voor de beveiliging van de verbinding. Wanneer een aanvaller de private key van de server verkrijgt, kan deze zich mogelijk als de server voordoen zolang het bijbehorende certificaat nog wordt vertrouwd. Bij correct gebruik van ephemeral sleuteluitwisseling en forward secrecy kunnen eerder opgenomen TLS-sessies niet met alleen de private key van het servercertificaat worden ontsleuteld.

Forward secrecy en finite field groups

Sleuteluitwisselingsmethoden met forward secrecy gebruiken tijdelijk, sessiespecifiek sleutelmateriaal. Daardoor kan eerder opgenomen communicatie niet achteraf worden ontsleuteld wanneer alleen de langdurige private key van het servercertificaat uitlekt.

Finite Field Diffie-Hellman Ephemeral (FFDHE) biedt forward secrecy door gebruik te maken van tijdelijke sleutels in finite-fieldgroepen. Binnen de gemeentelijke standaard is het gebruik van FFDHE niet toegestaan; elliptische-krommegroepen en waar mogelijk hybride post-quantumgroepen hebben de voorkeur.

Sleuteluitwisselingsalgoritme

De cryptografische sterkte is een maat om de beveiligingswaarde van een cryptografisch algoritme in uit te drukken. Verschillende algoritmes kunnen met andere sleutellengtes dezelfde cryptografische sterkte behalen. X25519MLKEM768, SecP256r1MLKEM768 en SecP384r1MLKEM1024 zijn hybride sleuteluitwisselingsgroepen die klassieke ECDHE combineren met ML-KEM. Deze combinatie is bedoeld om bescherming te bieden tegen zowel klassieke als toekomstige cryptografisch relevante quantumaanvallen. X25519MLKEM768 geniet de voorkeur in verband met de dreiging van een CRQC (cryptografisch-relevante quantumcomputer). Zie volgende paragraaf.

Cryptografisch-relevante quantumcomputer

Een cryptografisch-relevante quantumcomputer (CRQC) bestaat (nog) niet, maar kan nu al een risico vormen. Zo kunnen aanvallers met een store-now-decrypt-later aanval de versleutelde data nu al verzamelen en deze op een later moment ontsleutelen (wanneer zij beschikken over een CRQC). Cryptografie die veilig is tegen klassieke computers, maar niet tegen quantumcomputers, wordt quantumkwetsbaar genoemd. Cryptografie die ook veilig is tegen quantumaanvallen staat bekend als post-quantumcryptografie (PQC).

3.1. Checklist cryptografie en sleuteluitwisseling

- ☐ Forward secrecy is aanwezig (verplicht);
- ☐ Er wordt geen gebruik gemaakt van FFDHE (verplicht);
- ☐ X25519MLKEM768 is aanwezig (verplicht);
- ☐ Er wordt gebruik gemaakt van post-quantumcryptografie (PQC)-key exchange (voorkeur).

4. PKI, sleutels en certificaten

Een Certificate Authority (CA), of certificaatautoriteit, is een vertrouwde entiteit die digitale certificaten uitdeeft binnen een Public Key Infrastructure (PKI). Deze certificaten koppelen een identiteit (bijvoorbeeld een persoon, organisatie of server) aan een publieke sleutel. Het doel is om vertrouwen te creëren in digitale communicatie door authenticiteit, integriteit en vertrouwelijkheid te waarborgen.

Mutual Transport Layer Security

Mutual Transport Layer Security (mTLS) is een uitbreiding van het standaard TLS-protocol waarbij client én server elkaar authentifieren met digitale certificaten. TLS beveiligt communicatie door encryptie en authenticatie van de server; mTLS voegt hieraan toe dat ook de client een certificaat moet tonen, waardoor wederzijdse authenticatie ontstaat.

Gebruik gemeente 's-Hertogenbosch

Voor de domeinen die in het bezit zijn van gemeente 's-Hertogenbosch levert de gemeente zelf het benodigde PKI-materiaal aan. Hierbij wordt geen PKI-materiaal aangeleverd op basis van een aangeleverd Certificate Signing Request (CSR), maar enkel en alleen op PKCS#12 (ook wel PFX genoemd) gebaseerde certificaatarchieven (certificate stores) waarbij de private key en de bijbehorende certificaten afdoende zijn versleuteld zonder gebruik te maken van de OpenSSL-optie “-legacy”.

Elliptic Curve Digital Signature Algorithm (ECDSA) en Rivest-Shamir-Adleman (RSA)

ECDSA en RSA zijn beide algoritmen die kunnen worden gebruikt voor het maken en verifiëren van digitale handtekeningen. ECDSA maakt gebruik van elliptische-krommencryptografie en biedt bij een vergelijkbaar beveiligingsniveau aanzienlijk kleinere sleutels en handtekeningen dan RSA. Daardoor is ECDSA vooral aantrekkelijk in omgevingen waar rekenkracht, opslagruimte of bandbreedte beperkt zijn. RSA is ouder en wordt historisch gezien zeer breed ondersteund. Voor een vergelijkbaar beveiligingsniveau zijn echter grotere sleutels nodig. De prestaties verschillen bovendien per bewerking. RSA-verificatie kan zeer snel zijn, terwijl het genereren van RSA-handtekeningen doorgaans meer rekenkracht vereist. Welke methode het meest geschikt is, hangt af van onder meer compatibiliteit, beschikbare cryptografische bibliotheken, prestatie-eisen, sleutelbeheer en de gebruikte protocollen.

4.1. Checklist PKI, sleutels en certificaten

- ☐ Diensten die iedereen kan gebruiken, maken gebruik van een publieke CA (verplicht);
- ☐ Diensten die enkel door medewerkers van de gemeente worden gebruikt, maken gebruik van een publieke of interne CA (verplicht);
- ☐ Voor mTLS API-berichtenverkeer wordt gebruik gemaakt van PKI-overheid CA (verplicht);
- ☐ Voor domeinen in het bezit van gemeente 's-Hertogenbosch levert de gemeente zelf het benodigde PKI-materiaal (verplicht). Gemeente voorziet in dezen zelf in de van toepassing zijnde randvoorwaarden;
- ☐ Certificate stores op basis van PKCS#12 kunnen worden verwerkt (verplicht);
- ☐ Geen OpenSSL “-legacy” toegepast (verplicht);
- ☐ Volledige certificaatketen is aanwezig (uitgezonderd root-certificaten) (verplicht);
- ☐ Er wordt gebruikgemaakt van een passend ECDSA- of RSA-servercertificaat. ECDSA heeft in deze de voorkeur wanneer de vereiste clientcompatibiliteit dit toelaat. In geval van RSA bedraagt de sleutellengte minimaal 3072 bits.

5. HTTP security headers

HTTP security headers zijn als onderdeel van HTTP response headers cruciale serverinstellingen die browsers instrueren hoe ze zich moeten gedragen tijdens het bezoeken van een website, waardoor ze worden beschermd tegen veelvoorkomende aanvallen. Ze worden toegevoegd aan de HTTP-response en verbeteren de beveiliging en privacy.

X-Content-Type-Options

De X-Content-Type-Options header met de waarde nosniff zorgt ervoor dat een user agent niet het formaat van de ontvangen gegevens raadt (via een proces dat MIME-type-sniffing wordt genoemd).

Strict-Transport-Security

HTTP Strict Transport Security (HSTS) is een beveiligingsmechanisme dat websites beschermt tegen zaken zoals protocol-downgrade-aanvallen en cookie-kaping.

Content Security Policy (CSP)

Content Security Policy (CSP) is een beveiligingsstandaard die is ontworpen om een extra beveiligingslaag toe te voegen aan webapplicaties. Hiermee kunnen ontwikkelaars beperken welke bronnen (zoals JavaScript, CSS, afbeeldingen en andere) geladen mogen worden. Unsafe-inline betekent dat de CSP het gebruik van inline scripts of inline CSS toestaat. Unsafe-eval betekent dat de CSP functies zoals eval toestaat, waardoor de site code kan uitvoeren die is opgeslagen als een string, en zodoende misbruikt kan worden.

Permissions-Policy

Permissions-Policy is een HTTP security header waarmee bepaald kan worden welke browserfuncties en API's (zoals toegang tot de camera, microfoon, of geolocatie) wel of niet gebruikt mogen worden op een webpagina en in ingesloten (third-party) iframes.

Referrer-Policy

De Referrer-Policy is een HTTP security header die bepaalt welke informatie de browser meestuurt in de Referer header wanneer een gebruiker van de ene naar de andere pagina of website navigeert.

Cross-origin

Cross-origin verwijst in webontwikkeling naar een situatie waarin een webpagina, geladen vanuit één domein (oorsprong/origin), resources (zoals afbeeldingen, API-data, scripts) opvraagt van een *ander* domein, protocol of poort. Hierbij doen zich de volgende situaties voor:

- Cross-Origin Resource Sharing (CORS) is een mechanisme waarmee een server aangeeft welke andere origins browsercode toegang mogen geven tot een cross-origin response. CORS voorkomt niet noodzakelijk dat een verzoek wordt verzonden en is geen vervanging voor authenticatie of autorisatie;
- Cross-Origin-Opener-Policy (COOP) isoleert browsing contexts;
- Cross-Origin-Embedder-Policy (COEP) bepaalt onder welke voorwaarden een document cross-origin resources mag insluiten, en;
- Cross-Origin-Resource-Policy (CORP) laat een resource aangeven vanaf welke origins deze mag worden geladen.

Server en X-Powered-By

De Server en X-Powered-By HTTP response headers vormen een beveiligingsrisico omdat ze informatie over de gebruikte webserversoftware en frameworks prijsgeven. Bij misbruik kan deze informatie worden gebruikt om gericht te zoeken naar bekende kwetsbaarheden in specifieke softwareversies.

Securityheaders

Om te testen of aan de configuratie van HTTP response headers wordt voldaan, kan gebruik worden gemaakt van een dienst van Scott Helme: <https://securityheaders.io>. Let op: Hierbij is de score een indicatie. De minimale verplichte score kan in de praktijk mogelijkwerwijs niet gelijk staan aan de vereiste en juiste configuratie.

5.1. Checklist HTTP security headers

- ☐ X-Content-Type-Options: nosniff is aanwezig (verplicht);
- ☐ Strict-Transport-Security is aanwezig (verplicht);
- ☐ Content Security Policy is aanwezig (verplicht, vermijd unsafe-inline/eval);
- ☐ DigiD: Content Security Policy is aanwezig zonder unsafe-inline/eval (verplicht);
- ☐ Permissions-Policy is aanwezig (verplicht);
- ☐ Referrer-Policy is aanbevolen;
- ☐ Bij gebruik van CORS bevat Access-Control-Allow-Origin uitsluitend expliciet toegestane, expliciete origins (verplicht);
- ☐ DigiD: COOP, COEP en CORP zijn aanwezig (verplicht);
- ☐ Server, X-Powered-By en andere HTTP response headers die een product en/of versie tonen zijn niet toegestaan (verplicht);
- ☐ De score op securityheaders is minimaal "A" (verplicht).

6. Same-Origin Policy

De Same-Origin Policy (SOP) is een fundamenteel beveiligingsmechanisme in webbrowsers. Deze policy beperkt de mogelijkheid van scripts en documenten van de ene origin om gegevens van een andere origin te lezen of te wijzigen. Een origin wordt bepaald door de combinatie van protocol, hostnaam en poort.

De Same-Origin Policy bepaalt niet rechtstreeks of een webpagina in een <frame>, <iframe>, <embed> of <object> mag worden opgenomen. Hiervoor bestaan aparte beveiligingsmechanismen die een pagina in staat stellen te bepalen welke andere pagina's haar mogen insluiten.

X-Frame-Options

De oudere methode hiervoor is de HTTP security header X-Frame-Options. Bijvoorbeeld: X-Frame-Options: SAMEORIGIN. Met de instelling uit dit voorbeeld mag de betreffende pagina alleen worden ingesloten door een bovenliggende pagina met dezelfde origin. Een andere mogelijke waarde is DENY, waarmee insluiten volledig wordt geblokkeerd.

CSP met Frame-ancestors

De moderne en flexibelere methode is de CSP-richtlijn frame-ancestors. Bijvoorbeeld: Content-Security-Policy: frame-ancestors 'self'. Deze instelling komt functioneel ongeveer overeen met X-Frame-Options: SAMEORIGIN. Met frame-ancestors kunnen daarnaast specifieke vertrouwde origins worden toegestaan, bijvoorbeeld: Content-Security-Policy: frame-ancestors 'self' <https://www.s-hertogenbosch.nl>.

Voor moderne browsers heeft frame-ancestors de voorkeur, omdat deze richtlijn meer flexibiliteit biedt dan X-Frame-Options. Voor aanvullende compatibiliteit met oudere clients kunnen beide headers tegelijk worden gebruikt, mits zij hetzelfde beleid uitdrukken:

Voorbeeld:

Content-Security-Policy: frame-ancestors 'self';

X-Frame-Options: SAMEORIGIN

Deze maatregelen helpen vooral bij het voorkomen van clickjacking, waarbij een aanvaller een pagina onzichtbaar of misleidend in een andere pagina insluit.

6.1. Checklist Same-Origin Policy

- ☐ CSP met frame-ancestors aanwezig (verplicht);
- ☐ X-Frame-Options aanwezig in geval van oudere clients (optioneel).

7. API-specifieke beveiligingsmaatregelen

Cache-Control: no store

Cache-Control is een HTTP response-header die instructies (richtlijnen) bevat voor browsers en tussenliggende caches zoals CDN's over hoe lang en waar webcontent moet worden opgeslagen. Het optimaliseert de prestaties door onnodige serververzoeken te verminderen. Cache-Control: no-store is een HTTP-header die aan browsers en tussenliggende servers laat weten dat een webpagina of bestand nooit mag worden opgeslagen in de cache.

RateLimit headers

Deze headers informeren clients over de geldende limieten en het moment waarop opnieuw verzoeken kunnen worden gedaan. De limieten moeten aan de serverzijde worden afgedwongen; de headers zelf voorkomen geen overmatig of ongewenst gebruik.

API-authenticatie

API-authenticatie en -autorisatie kunnen onder meer worden ingericht met OAuth 2.0-toegangstokens, JWT's of API-keys. OAuth 2.0 is een autorisatieframework, JWT is een formaat waarin claims kunnen worden vastgelegd en een API-key is een rechtstreeks aan een client toegekend geheim. De passende methode hangt af van het gebruiksscenario en de vereiste beveiliging.

X-Correlation-ID

Een correlation ID is een identificatiecode waarmee een request en bijbehorende berichten door meerdere systemen kunnen worden gevolgd. Inkomende identifiers worden gevalideerd of bij de vertrouwensgrens vervangen door een nieuw gegenereerde waarde. De identifier bevat geen persoonsgegevens of andere betekenisvolle gegevens. Het wordt gebruikt om de volledige "reis" van een transactie of de keten van gebeurtenissen door verschillende systemen, services of microservices heen te traceren.

7.1. Checklist API-specifieke beveiligingsmaatregelen

- ☐ Cache-Control: no store is aanwezig bij gevoelige data zoals persoonsgegevens (verplicht);
- ☐ RateLimit headers is aanwezig (aanbevolen);
- ☐ Voor machine-to-machine-API's wordt gebruikgemaakt van een passende token- of certificaatgebaseerde authenticatiemethode (verplicht);
- ☐ Voor browsergebaseerde toepassingen mogen beveiligde sessiecookies worden gebruikt wanneer dit past bij het gekozen architectuur- en beveiligingsmodel (verplicht).
- ☐ X-Correlation-ID is aanwezig (aanbevolen).

8. Cookies en sessiebeveiliging

Het correct instellen van Secure, HttpOnly en SameSite op cookies beperkt de gevolgen van verschillende aanvallen. HttpOnly kan voorkomen dat scripts een sessiecookie rechtstreeks uitlezen, maar voorkomt XSS zelf niet. SameSite kan bescherming bieden tegen bepaalde vormen van cross-site request forgery (CSRF). Secure zorgt ervoor dat de cookie alleen via een HTTPS-verbinding naar de server wordt gestuurd. HttpOnly voorkomt dat client-side scripts (zoals JavaScript, document.cookie) toegang krijgen tot de cookie en SameSite ten slotte bepaalt of de cookie wordt meegestuurd bij verzoeken die afkomstig zijn van een andere website (cross-site requests).

SameSite

In geval van SameSite zijn er drie mogelijke instellingen:

- SameSite=Strict:
Dit cookie wordt alleen meegestuurd in verzoeken die afkomstig zijn van de eigen site (first-party). Klikte de gebruiker op een link van een externe site die naar de gewenste site leidt dan wordt het cookie niet meegestuurd.
- SameSite=Lax (Standaardinstelling sinds Chrome 80/84):
Dit cookie wordt bij cross-site verzoeken alleen meegestuurd bij bepaalde top-level navigaties met veilige HTTP-methoden, zoals het volgen van een gewone link. Bij verzoeken binnen dezelfde site wordt de cookie eveneens meegestuurd. Het cookie wordt niet meegestuurd bij cross-site POST-verzoeken (zoals het invullen van een formulier op een externe site). Dit om cross-site request forgery (CSRF) aanvallen te voorkomen.
- SameSite=None:
Dit cookie wordt meegestuurd bij alle verzoeken, inclusief cross-site verzoeken. Belangrijk: Als SameSite=None wordt gebruikt, dient het cookie ook het attribuut Secure te bevatten. Dit betekent dat het cookie alleen via HTTPS verstuurd wordt.

8.1. Checklist cookies en sessiebeveiliging

- ☐ Cookies bevatten Secure, HttpOnly en SameSite (verplicht);
- ☐ SameSite=Strict en SameSite=Lax zijn aanwezig (aanbevolen);
- ☐ SameSite=None wordt alleen gebruikt in combinatie met Secure (verplicht).

9. Security.txt

Een security.txt-bestand is een tekstbestand (RFC 9116) waarmee organisaties hun contactgegevens voor het melden van beveiligingslekken (Coordinated Vulnerability Disclosure) publiceren op hun website, meestal via /.well-known/security.txt. Sinds mei 2023 is dit verplicht voor Nederlandse overheden via “pas toe of leg uit”. Ethische hackers en onderzoekers kunnen op deze manier sneller en makkelijker beveiligingslekken melden bij de juiste persoon, wat de digitale weerbaarheid verhoogt.

Een 302-redirect is een HTTP-statuscode die aangeeft dat een webpagina tijdelijk is verplaatst naar een andere locatie.

9.1. Checklist Security.txt

- ☐ /.well-known/security.txt moet 302-redirecten naar <https://s-hertogenbosch.nl/.well-known/security.txt> (verplicht);
- ☐ Leveranciers mogen hun eigen security.txt gebruiken mits via deze constructie relevante meldingen worden doorgezet naar informatieveiligheid@s-hertogenbosch.nl (verplicht indien van toepassing).

10. Resource Public Key Infrastructure

Resource Public Key Infrastructure (RPKI) is een beveiligingskader dat Border Gateway Protocol (BGP)-routing op internet veiliger maakt door IP-adresblokken cryptografisch te koppelen aan geautoriseerde netwerken (ASN's). Het voorkomt route-hijacks en -lekken, waardoor onbedoelde omleidingen van internetverkeer worden tegengegaan. In Nederland is het gebruik van RPKI verplicht voor de rijksoverheid.

Route Origin Authorizations

Route Origin Authorizations (ROA) zijn digitaal ondertekende verklaringsdocumenten waarmee eigenaren van IP-adressen (IPv4/IPv6) autoritatief vastleggen welk autonoom systeem (ASN) hun netwerkblokken mag aankondigen op het internet.

10.1. Checklist Resource Public Key Infrastructure

- ☐ Publicatie Route Origin Authorizations door netwerkleverancier (verplicht);
- ☐ Validatie routes door netwerkleverancier (verplicht);
- ☐ Verwijderen van ongeldige routes door netwerkleverancier (verplicht).

11. Niet-productieomgevingen

Niet-productieomgevingen zijn veilige, afgeschermdde IT-omgevingen voor ontwikkeling, testen en acceptatie (OTAP), losgekoppeld van live gebruikersdata. Hiermee wordt voorkomen dat fouten de productieomgeving beïnvloeden. Daarnaast bieden niet-productieomgevingen ruimte voor het testen van softwarewijzigingen, prestaties en kwaliteitsborging.

11.1. Checklist niet-productieomgevingen

- ☐ Niet-productieomgevingen zijn niet publiek toegankelijk (verplicht);
- ☐ Niet-productieomgevingen bevatten geen publieke DNS-records (verplicht).

12. Algemeen

Op de site <https://internet.nl> kan de juiste implementatie van TLS en HTTP response headers worden gecontroleerd op “best practices” zoals aanbevolen door partijen uit de internetgemeenschap en de Nederlandse overheid.

Als hier 100% wordt gescoord, voldoe je aan de uitgangspunten van internet.nl zoals bijvoorbeeld het conformeren aan overheidsstandaarden. Dit betekent echter niet per definitie dat aan alle eisen van gemeente 's-Hertogenbosch wordt voldaan. Let dus op: De score is dus een indicatie. De minimale verplichte score kan in de praktijk mogelijk niet gelijk staan aan de vereiste en juiste configuratie. De voorwaarden voor het gebruik van IPv6 en DNSSEC staan vermeld in de gemeentelijke technische architectuur (TA). Voor TLS en HTTP response headers is het uitgangspunt dat de verbinding voldoende beveiligd is en dat alle applicatie-beveiligingsopties zijn ingesteld.

Als het testen van de configuratie van TLS en HTTP response headers niet mogelijk is via het internet, kan gebruik worden gemaakt van hulpprogramma's zoals curl (<https://curl.se>) en testssl.sh (<https://testssl.sh>). Hoewel deze programma's geen bindende scores opleveren, kunnen ze wel een goede indicatie geven over de kwaliteit van de desbetreffende configuratie.

Documentatie

Relevante documentatie voor de configuratie van TLS en HTTP response headers:

- <https://github.com/ssllabs/research/wiki/SSL-and-TLS-Deployment-Best-Practices>;
- <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers>;
- <https://github.com/OWASP/www-project-secure-headers>;
- <https://scotthelme.co.uk/tough-cookies>;
- <https://docs.tlsref.org/server-side-tls.html>;
- <https://www.ncsc.nl/transport-layer-security-tls/ict-beveiligingsrichtlijnen-voor-transport-layer-security>.

12.1. Checklist algemeen

- ☐ De score op internet.nl is 100% (verplicht).