De Logius-norm voor DigiD: perikelen bij technisch testen (uitgebreide versie)

Matthijs Koot, senior security consultant bij Madison Gurkha. Geschreven in mei/juni 2014, op persoonlijke titel.

Abstract

Het NCSC publiceerde in januari 2012 een leidraad met beveiligingsrichtlijnen voor webapplicaties. Een selectie van die richtlijnen werd een maand later door Logius verheven tot een beveiligingsnorm waar de ruim 550 organisaties die DigiD-gekoppelde systemen hebben aan moeten voldoen. In dit artikel bespreek ik enkele van de 'zuiver technische' onderdelen uit die norm die zonder een algemeen geaccepteerde interpretatie niet eenduidig toetsbaar zijn. Het valt daardoor te verwachten dat verschillende experts in identieke gevallen tot verschillende oordelen komen. Bij elk onderdeel geef ik uit de NCSC-leidraad de beschrijving, de doelstelling en de vereiste succescriteria; en vervolgens mijn reflectie en een werkwijze voor toetsing. Later dit jaar wordt een nieuwe versie van de NCSC-leidraad verwacht. Dit artikel is bedoeld als expositie van de huidige problematiek en als klankbord voor de nieuwe NCSC-leidraad, die nog in ontwikkeling is.

Introductie

In januari 2012, in het kielzog van Lektober, publiceerde het NCSC de leidraad "ICT-beveiligingsrichtlijnen voor webapplicaties" [NCSC1]. Het NCSC, Logius en de Rijksauditdienst hebben daaruit een selectie van 28 richtlijnen gemaakt die door Logius is verheven tot de "norm ICT-beveiligingsassessments DigiD" [Logius]. Deze bevat zowel organisatorische als technische richtlijnen en is van toepassing op alle "internet-facing webpagina's, systeemkoppelingen en infrastructuur die met DigiD gekoppeld zijn en betrekking hebben op het proces" [Logius]. (In dit artikel verwijst "norm" naar de Logius-norm en "richtlijn" naar de NCSC-richtlijnen die de bouwstenen zijn van de norm.)

Het "ICT-Beveiligingsassessment DigiD" is een jaarlijks terugkerend audittraject dat organisaties met DigiD-gekoppelde systemen in opdracht van Logius dienen uit te (laten) voeren onder verantwoordelijkheid van een RE. In het audittraject kan externe expertise worden ingeschakeld. Een voorbeeld daarvan is een gespecialiseerde technische beveiligingstester. Een externe beveiligingstester verzamelt bewijsmateriaal en kan bij elk van de technische richtlijnen aangeven of de applicatie hieraan voldoet. De RE kan voor diens oordeel steunen op het oordeel en de bewijsvoering van de beveiligingstester.

De kernactiviteit van mijn werkgever is technisch beveiligingsonderzoek. We hebben gekeken op welke onderdelen van de DigiD-norm we onze technische kennis kunnen inzetten. (Volledige DigiD-audits voeren we uit in samenwerking met dochtermaatschappij ITSX, maar dat valt buiten de scope van dit artikel.) Na het bestuderen van de richtlijnen kwamen we in 2012 tot de conclusie dat tien van de richtlijnen in de norm niet of nauwelijks afhankelijk zijn van organisatie of beleid: die richtlijnen beschouwen we daarom als 'zuiver technisch'. Het betreft de B3-richtlijnen (applicatiebeveiliging) en B5-2 (vertrouwelijkheid en onweerlegbaarheid). Bij die richtlijnen gaat het dus niet zozeer om het bestaan van intern beleid en een gehandhaafde praktijk, maar om het actuele technische beveiligingsniveau van de applicatie zelf. Kortom: is de applicatie in praktijk voldoende veilig?

De probleemstelling die ik in dit artikel behandel, is dat de richtlijnen als toetsbaar zijn bedoeld, maar niet alle richtlijnen dat zijn, of in elk geval niet zonder nadere interpretatie. Het valt daarom bij die richtlijnen te verwachten dat verschillende beveiligingstesters in identieke gevallen tot verschillende oordelen komen. Ik betoog dat die NCSC-richtlijnen of de Logius-norm aanpassing behoeven, en dat voor consistente(re) toetsing nader uitgewerkte toetsingscriteria wenselijk zijn. Omwille van leesbaarheid beperk ik me in dit artikel tot de mogelijke (expert)oordelen "voldoet" en "voldoet niet" (niet te verwarren met het oordeel van een RE). In dit artikel kan ik de onduidelijkheid niet volledig wegnemen, maar wel expliciet maken, en deel ik enkele gedachten over de werkwijze bij toetsing.

Code-inspectie

NOREA heeft eind 2012 een "Handreiking DigiD ICT-beveiligingsassessments voor RE's" gepubliceerd [NOREA]. Uit bijlage 1 blijkt dat de auteurs broncode-inspectie geen noodzakelijk onderdeel vinden om aan de Logius-norm te voldoen: zelfs niet waar het de richtlijn over het gebruik van geparametriseerde SQL-queries betreft (B3-5). Op dat punt ben ik het niet met de auteurs eens, omdat de ervaring leert dat bij broncode-inspectie kwetsbaarheden worden ontdekt die zonder de broncode waarschijnlijk niet zouden worden gevonden. Vooral SQL-injectie die zich in de uithoeken van de applicatie bevindt en/of niet zichtbaar is aan de buitenkant voor tools als Sqlmap; maar ook fouten in authenticatie- of autorisatielogica. Indien de applicatie ook voldoende veilig moet zijn indien geconfronteerd met een misnoegde insider, dan is broncode-inspectie erg belangrijk. (Dat zeg ik uit overtuiging, niet als WC-Eend.) En ja, dat vraagt tijd en eist vaardigheid in broncode-inspectie. Waar het NCSC als succescriterium noemt dat de broncode beschikbaar moet zijn voor onderzoek, citeer ik dat criterium in dit artikel.

B3-1: invoervalidatie

Beschrijving:	"De webapplicatie valideert de inhoud van een HTTP-request voor die wordt gebruikt."
Doelstelling:	"Voorkomen [sic] het verlies, wijziging of misbruik van gegevens door onbetrouwbare (malafide) invoer. Voorkom dat de applicatielogica wordt beïnvloed."
Vereiste succescriteria:	"1) Beschikken over de broncode van de programmatuur. 2) Validatie vindt plaats op in ieders [sic] geval dynamische onderdelen van de URL, query parameters, form parameters, cookies, HTTP-headers, XML en bestanden. 3) De webapplicatie voert deze validatie uit op basis van: typecontrole (bijvoorbeeld string of integer); lengtecontrole; formaatcontrole (op basis van bijvoorbeeld een reguliere expressie); controle op valide karakters (bijvoorbeeld alleen 'A-Z' en 'a-z'). 4) In het geval de invoer niet voldoet aan één of meerdere van bovenstaande controles, weigert de webapplicatie deze invoer. 5) De webapplicatie filtert de invoer op basis van: malafide sleutelwoorden (bijvoorbeeld 'DROP' of ' rm '); malafide tekens (bijvoorbeeld '" of '"); malafide patronen (bijvoorbeeld '/**/' of '..\..\') (...)"

Reflectie

De doelstelling "voorkom dat de applicatielogica wordt beïnvloed" is weinig specifiek. Onder "vereiste succescriteria" staan verschijningsvormen van invoervalidatie, geen succescriteria. Het kan immers niet zo zijn dat een applicatie niet aan richtlijn B3-1 voldoet enkel en alleen omdat niet elke invoer expliciet op lengte wordt gecontroleerd; een goed ingerichte database en goed opgezette foutafhandeling vangt corpulente invoer prima af. En indien een postcodeveld invoer van "<" of ">" niet weigert maar wel normaliseert, bijvoorbeeld door die tekens te verwijderen of te vervangen door spaties (zie B3-3): dan wordt, strikt genomen, ook niet aan deze richtlijn voldaan. Voor een applicatie die fatsoenlijk omgaat met vreemde tekens, is controle op valide karakters onnodig: het biedt weliswaar een extra beschermingslaag, maar het is geen verstandige minimumeis. Invoervalidatie is een essentieel onderdeel van de beveiliging, maar waar dien je als beveiligingstester, in de context van de Logius-norm, de streep te trekken tussen "voldoet" en "voldoet niet"? Gaat het alleen om afwezigheid van kwetsbaarheden die praktisch en onmiddellijk uitbuitbaar zijn (zoals stored XSS) of om meer dan dat? Ook in het eerste geval blijft de vraag: wanneer is een "voldoet niet" gerechtvaardigd? Het is belangrijk dat verschillende experts in gelijke gevallen (voldoende) gelijk oordelen.

Mogelijke toetsing

Concreet testen: XSS; LDAP-injectie; XXE, XSL/XML/XPath-injectie; OS/shell-injectie; JSONP-hijacking; Linq-injectie; blacklist/whitelist-evasion; path traversal; XML Signature wrapping, et cetera (n.o.t.k.).

Oordeel "voldoet niet" indien ten minste één gemiddelde of hoge kwetsbaarheid is gevonden die (ook) door invoervalidatie kan worden weggenomen. (n.o.t.k.; voor het bepalen van "laag", "gemiddeld" en "hoog" is een algemeen geaccepteerd beslissingsmodel wenselijk: hiertoe zouden beveiligingstesters het inschalingsmodel kunnen gebruiken dat het NCSC hanteert voor haar beveiligingsadviezen [NCSC2]. Indien alle beveiligingstesters tijdens DigiD-beveiligingsassessments hetzelfde inschalingsmodel gebruiken, vermindert de kans dat verschillende beveiligingstesters in gelijke situaties verschillend oordelen.)

Oordeel "voldoet niet" indien er structurele gebreken of inconsistenties in de invoervalidatie zijn, ook als er geen sprake is van een onmiddellijk uitbuitbare kwetsbaarheid: denk aan blacklist/whitelist-evasion zonder dat daarmee op dit moment een uitbuitbare kwetsbaarheid kan worden aangetoond; en aan het accepteren van "<" en ">" in een context waar dat niet passend is, ook al worden de tekens netjes gecodeerd in de uitvoer (zie B3-4).

Oordeel "voldoet" in alle andere gevallen.

B3-2: authenticatie en autorisatie

Beschrijving:	"De webapplicatie controleert voor elk HTTP verzoek of de initiator geauthenticeerd is en de juiste autorisaties heeft."
Doelstelling:	"Valideer de initiator van een HTTP-request teneinde te voorkomen dat kwaadwillenden transacties uit naam van een valide gebruiker uitvoeren."
Vereiste succescriteria:	"1) Beschikken over de broncode van de programmatuur. 2) De waarde van cookies is gekoppeld aan het IP-adres waarnaar deze waarde is verstuurd. 3) Voor onderdelen van de webapplicatie waarmee transacties door een gevalideerde gebruiker kunnen worden uitgevoerd: zijn formulierpagina's voorzien van een dynamisch token; accepteert de webapplicatie alleen verzoeken waarbij de inhoud van de Referer-header overeenkomt met

	de URL van de betreffende webapplicatie."

Reflectie

Met de huidige succescriteria is richtlijn B3-2 goed toetsbaar. Maar de keerzijde van specifiek maken is meteen zichtbaar: de vereiste succescriteria zijn te beperkt. Alleen IP-binding van sessies en bescherming tegen CSRF worden genoemd, terwijl authenticatie- en autorisatiekwetsbaarheden zich ook kunnen bevinden in de verwerking van GET- en POST-parameters (denk aan een parameter "gebruiker=1" die wordt aangepast naar "gebruiker=2"; idem bij gebruikerspecifieke gegevens indien daaraan wordt gerefereerd met absolute volgnummers). Wellicht hebben de opstellers gemeend dat die variaties reeds onder B3-1 zijn afgedekt? Hoe dan ook, deze richtlijn is in beginsel nuttig en kan worden verbeterd. Hoe lang we kunnen blijven eisen dat sessies IP-gebonden zijn, is trouwens ook de vraag: door toename van mobiele communicatie (3G/4G) waar IP-adressen kunnen veranderen en protocollen zoals Happy Eyeballs voor dual-stacked (IPv4+IPv6) devices is deze eis op termijn misschien niet meer vol te houden.

Mogelijke toetsing

Concreet testen: zijn sessies IP-gebonden; CSRF; clickjacking; autorisatiecontroles; server-side invalidaties van sessies na uitloggen (n.o.t.k.).

Oordeel "voldoet niet" indien de applicatie kwetsbaar is voor CSRF, voor clickjacking, geen IP-gebonden sessies heeft, de autorisatiecontroles gebrekkig zijn en/of sessies bij het uitloggen niet worden geïnvalideerd op de server (n.o.t.k.).

Oordeel "voldoet" in alle andere gevallen.

B3-3: normalisatie

Beschrijving:	"De webapplicatie normaliseert invoerdata voor validatie".
Doelstelling:	"Normaliseer alle invoerdata voor deze te valideren om te voorkomen dat filteringmechanismen ongewenste patronen niet herkennen."
Vereiste succescriteria:	"Beschikken over de broncode van de programmatuur."

Reflectie

Normalisatie is een optionele voorfase van invoervalidatie (B3-1) en hoort daaronder thuis. Je moet zorgen dat de applicatie bestand is tegen alle soorten invoer, dus ook path traversal-pogingen (die trouwens al zijn afgedekt door B3-7), injectie van NULL-bytes, onverwachte encoding, et cetera. Het enige succescriterium is dat de broncode beschikbaar moet worden gesteld, maar daaruit kan natuurlijk niet automatisch het oordeel "voldoet" volgen. Waar dien je als beveiligingstester de streep te trekken tussen het oordeel "voldoet" en het oordeel "voldoet niet"? De enige manier die ik kan bedenken, blijft vaag: bevat de applicatie een kwetsbaarheid die (gedeeltelijk) aan een gebrek aan normalisatie kan worden toegeschreven? In dat geval zou ook XSS meestal resulteren in een "voldoet niet" op deze richtlijn, omdat de XSS-kwetsbaarheid voorkomen had kunnen worden door normalisatie. Of normalisatie daartoe de beste maatregel is, blijft dan buiten beschouwing.

Mogelijke toetsing

Concreet testen: zie B3-1.

Oordeel "voldoet niet" indien er ten minste één gemiddelde of hoge kwetsbaarheid is gevonden die (ook) kan worden weggenomen door normalisatie (n.o.t.k.).

Oordeel "voldoet" in andere gevallen.

B3-4: uitvoercodering

Beschrijving:	"De webapplicatie codeert dynamische onderdelen in de uitvoer."
Doelstelling:	"Codeer dynamische onderdelen van de uitvoer zodat er geen ongewenste tekens in de

	uitvoer terecht komen."
Vereiste succescriteria:	"Beschikken over de broncode van de programmatuur."

Reflectie

Ook aan deze richtlijn lijkt te kunnen worden voldaan uitsluitend door broncode beschikbaar te stellen. Maar de doelstelling lezende en daar eigen interpretatie aan toevoegend, wordt de richtlijn wel helder: indien gebruikersinvoer wordt gebruikt in een uitvoer, zorg dan voor codering (bijvoorbeeld het vervanging van "<" door de HTML-gecodeerde variant "<") om te voorkomen dat invoer als, bijvoorbeeld, de invoer als JavaScript-code wordt geïnterpreteerd door de browser. XSS is bijna altijd (ook) een probleem van uitvoercodering.

Mogelijke toetsing

Concreet testen: zie B3-1.

Oordeel "voldoet niet" indien er ten minste één gemiddelde of hoge kwetsbaarheid is gevonden die (ook) door uitvoercodering kan worden opgelost (n.o.t.k.).

Oordeel "voldoet" in alle andere gevallen.

B3-5: geparametriseerde SQL

Beschrijving:	"Voor het raadplegen en/of wijzigen van gegevens in de database gebruikt de webapplicatie alleen geparametriseerde queries."
Doelstelling:	"Door databasequeries samen te stellen op basis van geparameteriseerde queries (in tegenstelling tot dynamische strings), wordt de kans op SQL-injectie aanzienlijk verkleind."
Vereiste succescriteria:	"1) Beschikken over de broncode van de programmatuur. 2) De webapplicatie maakt gebruik van geparameteriseerde queries bij het benaderen van databases."

Reflectie

Het tweede succescriterium kan worden aangescherpt: "de webapplicatie maakt **uitsluitend** gebruik van geparameteriseerde queries bij het benaderen van databases." In de beschrijving is dit al wel verwoord ("alleen geparametriseerde queries"), maar het hoort in precieze bewoording thuis bij de succescriteria. Een SQL-injectiekwetsbaarheid kan aanvullend worden opgevat als een probleem van invoervalidatie (B3-1) en/of normalisatie (B3-3). Eén SQL-injectiekwetsbaarheid kan dus het oordeel "voldoet niet" opleveren bij B3-1, B3-3 en B3-5; net zoals één XSS-kwetsbaarheid het oordeel "voldoet niet" kan opleveren bij B3-1, B3-3 en B3-4.

Mogelijke toetsing

Concreet testen: SQL-injectiekwetsbaarheid en broncode-inspectie.

Oordeel "voldoet niet" indien ten minste één SQL-injectiekwetsbaarheid is gevonden of ten minste één SQL-statement wordt opgebouwd via concatenatie met gebruikersinvoer (dus: ongeacht of sprake is van SQL-injectiekwetsbaarheid).

Oordeel "voldoet" indien op basis van broncode is vastgesteld dat uitsluitend geparametriseerde queries worden gebruikt. (Testen zonder broncode-inspectie kan mijns inziens nooit tot "voldoet" leiden.)

B3-6: server-side validatie

Beschrijving:	"De webapplicatie valideert alle invoer, gegevens die aan de webapplicatie worden aangeboden, aan de serverzijde."
Doelstelling:	"Voorkom dat controles kunnen worden omzeild."
Vereiste succescriteria:	"1) Beschikken over de broncode van de programmatuur. 2) Voor elke controle die de webapplicatie uitvoert aan de clientzijde, is een equivalent aanwezig aan de serverzijde.

Reflectie

Om tegen deze richtlijn te toetsen zou je alle client-side controles in kaart moeten brengen en vervolgens toetsen of server-side dezelfde validaties worden uitgevoerd. Aangezien de client reeds als onvertrouwd dient te worden beschouwd, levert deze richtlijn een flinke hoeveelheid overbodig werk op. NOREA interpreteert deze richtlijn terecht als "reeds afgedekt door B3-1, B3-3, B3-4 en B3-5" [NOREA]. Richtlijn B3-6 zou wellicht obsoleet kunnen worden verklaard.

Mogelijke toetsing

Concreet testen: zie B3-1 en B3-5.

Oordeel "voldoet niet" indien bij B3-1, B3-3, B3-4 en/of B3-5 al "voldoet niet" staat. Verwijs naar de betrokken richtlijnen.

Oordeel "voldoet" in alle andere gevallen.

B3-7: dynamische file includes

Beschrijving:	"De webapplicatie staat geen dynamische file includes toe of beperkt de keuze mogelijkheid (whitelisting)."
Doelstelling:	"Voorkom dat ongewenste bestanden worden geïncorporeerd in een webapplicatie."
Vereiste succescriteria:	"1) Beschikken over de broncode van de programmatuur. 2) De webapplicatie maakt geen gebruik van dynamische file includes."

Reflectie

Het tweede succescriterium eist dat de webapplicatie geen gebruik maakt van dynamische file includes, terwijl de beschrijving van de richtlijn dat nuanceert ("of beperkt de keuze mogelijkheid"); dat is inconsistent. Verder vind ik dat uploadfunctionaliteit ook expliciet onder B3-7 mag worden opgenomen.

Mogelijke toetsing

Concreet testen: LFI, RFI, (inclusief path traversal, NULL-byte injectie etc., genoemd bij B3-1).

Oordeel "voldoet niet" indien er LFI of RFI is gevonden.

Oordeel "voldoet" in alle andere gevallen.

B3-16: Secure & HttpOnly cookies

Beschrijving:	"Zet de cookie attributen 'HttpOnly' en 'Secure'."
Doelstelling:	"Voorkom dat cookie communicatie kan worden afgeluisterd en voorkom dat cookies gestolen kunnen worden via cross site scripting."
Vereiste succescriteria:	"Het set cookie command bevat de secure flag en de HTTPOnly flag."

Reflectie

Wat is er met de boven- en onderkasten gebeurd tussen de beschrijving ('HttpOnly' en 'Secure') en het succescriterium ('secure' en 'HTTPOnly')? En er bestaat niet zoiets als "het set cookie command": ten eerste is het "Set-Cookie", ten tweede is het geen "command" maar een HTTP-header, en ten derde kunnen er meerdere "Set-Cookie"-headers zijn (het lidwoord "het" klopt dus niet). Deze richtlijn is niettemin goed toetsbaar: controleer of de cookieparameters Secure en HttpOnly worden gebruikt bij sessiecookies en andere cookies die authenticatie- en autorisatie-informatie bevatten.

Mogelijke toetsing

Concreet testen: of Secure en HttpOnly worden gezet voor alle sessiecookies en andere cookies die authenticatie- en autorisatie-informatie bevatten.

Oordeel "voldoet" indien een applicatie bij al deze cookies beide parameters zet.

Oordeel "voldoet niet" in alle andere gevallen.

B5-2: HTTPS

Beschrijving:	"Maak gebruik van versleutelde (HTTPS) verbindingen."
Doelstelling:	"Voorkom misbruik van (vertrouwelijke) gegevens die tijdens transport zijn onderschept."
Vereiste succescriteria:	"1) De inrichting is gebaseerd op een vastgesteld inrichtingsdocument/ontwerp, waarin is vastgelegd welke uitgangspunten gelden voor de toepassing van versleutelde verbindingen (SSL/TLS). 2) Er vindt een redirect plaats van HTTP naar HTTPS op het moment dat een (contact) formulier wordt opgevraagd."

Reflectie

Deze richtlijn is minder 'zuiver technisch' dan de andere: het eerste succescriterium spreekt immers over vastgestelde documentatie. Maar technisch testen is op twee manieren relevant: ten eerste vanwege de eis dat er een redirect moet plaatsvinden van HTTP naar HTTPS. En ten tweede omdat er in de configuratie van SSL/TLS allerlei technische details zijn die cruciaal zijn voor de werkelijke bescherming die met SSL/TLS wordt geboden. In de succescriteria worden geen eisen gesteld aan deze configuratie: zo bezien zou je met NULL-ciphers (SSL/TLS zonder versleuteling) en SSLv2 (een kwetsbare versie van het protocol) een "voldoet" kunnen krijgen, maar dat kan niet de bedoeling zijn. Op dit vlak is het aan de beveiligingstester om te onderzoeken of de opzet van de SSL/TLS-configuratie voldoende veilig is. Ook hier is de vraag: welk criterium hanteer je? Een strenge beveiligingstester zou al "voldoet niet" oordelen indien de SSL/TLS-configuratie geen ciphers ondersteunt die Perfect Forward Secrecy (PFS) bieden. (PFS voorkomt dat uit een in de toekomst gekraakte langetermijnsleutel sessiesleutels kunnen worden afgeleid die in het verleden zijn gebruikt voor versleuteling, en waarmee in het verleden onderschepte versleutelde SSL/TLS-comunicatie alsnog kan worden ontsleuteld.) Een minder strenge tester zal wellicht alleen "voldoet niet" oordelen indien er sprake is van hoge risico's zoals Heartbleed.

Mogelijke toetsing

Concreet testen: HTTP->HTTPS redirect; HSTS; ondersteunde ciphers en protocolversies; certificaat; kwetsbaarheden in de SSL/TLS-software; et cetera.

Oordeel "voldoet niet" indien gevoelige gegevens op ten minste één locatie in de applicatie over onversleuteld HTTP worden verstuurd (wat "gevoelig" is, is aan de beveiligingstester).

Oordeel "voldoet niet" indien er ten minste één gemiddelde of hoge kwetsbaarheid is gevonden in de opzet van SSL/TLS of de gebruikte software.

Oordeel "voldoet" in alle andere gevallen.

Denken vanuit kwetsbaarheden

Bijna alle XSS-kwetsbaarheden zijn een probleem van uitvoercodering (B3-4). Bijna alle XSS-kwetsbaarheden kunnen daarnaast ook worden opgevat als een probleem van invoervalidatie (B3-1) en/of normalisatie (B3-3). Neem bijvoorbeeld een XSS waarbij de gebruiker "<script>alert(1)</script>" invoert in een postcodeveld. De tekens "<" en ">" zijn nimmer passend in een postcodeveld: dan is er dus sprake van gebrekkige invoervalidatie, omdat er geen melding komt dat "<" en ">" niet zijn toegestaan. Maar ook van gebrekkige normalisatie, omdat "<" en ">" niet automatisch door de applicatie uit de invoer zijn gehaald.

Indien er onduidelijkheid bestaat over hoe het oordeel tot stand dient te komen, is te verwachten dat beveiligingstesters niet immer tot eenzelfde oordeel komen. We zouden met elkaar concrete afspraken kunnen maken: bijvoorbeeld dat XSS altijd resulteert in een "voldoet niet" op B3-4 (richtlijn over uitvoercodering), tenzij er goede gronden zijn om ervan af te wijken (er zijn allerlei verschijningsvormen van XSS). Daarmee is de norm niet verbeterd, maar wordt wel de beoordeling van verschillende experts meer gelijk. Maar goed, het is even afwachten hoe de nieuwe versie van de richtlijnen eruit zullen zien.

De richtlijnen B3-1 en B3-3 zijn niet gericht op specifieke kwetsbaarheden, maar op algemene maatregelen waarmee allerlei soorten kwetsbaarheden, waarvan XSS er één is, kunnen worden weggenomen. Andere richtlijnen, bijvoorbeeld B3-5 (richtlijn over geparametriseerde SQL) en B3-7 (richtlijn over dynamische file includes), zijn welbeschouwd verbijzonderingen van B3-1 die horen bij specifieke kwetsbaarheden (SQL-injectie resp. LFI, RFI). Misschien zou de Logius-norm meer van die verbijzonderingen kunnen bevatten waarbij wordt geredeneerd vanuit specifieke kwetsbaarheden in plaats van vanuit algemene maatregelen voor mitigatie. Denk bijvoorbeeld aan: "de applicatie is niet

kwetsbaar voor XSS" (want XSS verdient dan weer wel een eigen categorie --- iets dat OWASP al jaren heeft). Dat maakt het eenduidiger voor zowel beveiligingstesters als organisaties die aan de norm moeten voldoen. Immers, redenerend vanuit de bescherming van gegevens en functionaliteit, gaat het erom dat er geen XSS-kwetsbaarheid bestaat, niet om de manier waarop XSS is voorkomen--- invoervalidatie, normalisatie of uitvoercodering, het is allemaal goed.

Bij het redeneren vanuit kwetsbaarheden zou je de norm kunnen aanpakken vanuit het perspectief "voldoet, tenzij", dat de beveiligingstester uitdaagt om het tegendeel te bewijzen.

Leidraad is niet uitputtend

Beveiligingstesters kijken verder dan de Logius-norm. Ook de Logius-norm kijkt verder dan de Logius-norm: in het normdocument adviseert Logius organisaties ook andere maatregelen uit de NCSC-leidraad te "adopteren". Maar ook de NCSC-leidraad is niet uitputtend: zo komen clickjacking en open redirection er niet in voor, en natuurlijk ontbreken onderwerpen die pas na (of vlak voor) publicatie van de leidraad aan de orde zijn gekomen: zo gaat bij SSL/TLS onze aandacht tegenwoordig ook uit naar HSTS, BEAST, CRIME, Lucky13, Heartbleed, afwezigheid van OCSP stapling, enzovoorts. Een beveiligingstester kan en moet zich vrij voelen om op basis van die bevindingen, die niet expliciet zijn genoemd in richtlijn B5-2 maar passen in de achterliggende gedachte ervan, naar eigen inzicht "voldoet niet" te oordelen op B5-2 (mits goed onderbouwd). Uiteindelijk blijft het aan de RE te bezien wat met het oordeel en bewijsmateriaal van de beveiligingstester te doen. Ook staat het de beveiligingstester vrij om een krachtige aanbeveling te doen zónder dat die van invloed is op de DigiD-certificering; zoals thans, strikt genomen, de enige optie is wanneer je Heartbleed aantreft. Ook op dit punt --- welke SSL/TLS-bevinding telt nog mee voor beoordeling van B5-2 en welke niet --- is eenduidigheid nodig.

Hoe verder?

Moet het beleidseffect van de Logius-norm zijn: de afwezigheid van specifieke kwetsbaarheden? Of een veilig ontwerp dat veilig is uitgeprogrammeerd? Dat laatste is natuurlijk beter, maar daar is de norm in de huidige opzet simpelweg niet voor geschikt. Het certificeringsmodel van het Framework Secure Software dat recent het licht zag is daar waarschijnlijk beter voor geschikt [SSF]. (Disclaimer: de hoofdauteur daarvan is een bevriende oud-collega.) Wat mij betreft zou het beleidseffect van 'zuiver technische' onderdelen in de Logius-norm zich in eerste instantie moeten richten op afwezigheid van technische kwetsbaarheden en op aanwezigheid van specifieke, eenduidig toetsbare technische beveiligingsmaatregelen; eventueel met een gereserveerde catch-all categorie á la B3-1 waar thans onbekende kwetsbaarheden in kunnen worden opgevangen. Indien toch op generieke beveiligingsmaatregelen moet worden getest, zorg dan dat er een algemeen geaccepteerde interpretatie is die de werkwijze bij toetsing voldoende eenduidig maakt.

Betere aansluiting bij OWASP behoort tot de mogelijkheden: OWASP geeft een breed geaccepteerd vocabulaire en is beter hanteerbaar voor beveiligingstesters en de ontwikkelaars die de kwetsbaarheden moet oplossen (bijvoorbeeld dankzij specifieke categorieën voor XSS en injectie). Het vermindert het probleem van slechte toetsbaarheid en inconsistentie in oordelen. OWASP kan een gedeelte van de huidige technische richtlijnen goed afdekken, biedt verbreding, en geeft de beveiligingstester een duidelijker kader voor beoordeling, hoewel algemeen geaccepteerde toetscriteria ook dan nodig blijven voor de groepscategorieën (eigenlijk alles behalve A2 en A5). Tabel 1 geeft een kruisreferentie tussen OWASP en de Logius-richtlijnen. Nota bene: OWASP heeft veel uitgebreidere richtlijnen en volwassenheidsbeschrijvingen die wellicht zelfs beter gebruikt kunnen worden omdat ze al gericht zijn op webapplicaties en een betere dekking bieden dan de Logius-norm en de OWASP top-10. Tabel 1 dient slechts ter illustratie.

Tabel 1: Kruisreferentie Logius-norm en OWASP top 10 (versie 2013)

OWASP-categorie	**Voorbeelden van kwetsbaarheid**	**Richtlijnen uit Logius-norm**
A1: Injection	SQL-, LDAP-, Xpath-, Linq-injectie	B3-1, B3-3, B3-4, B3-5, B3-6, B3-7
A2: Cross-Site Scripting (XSS)	alle vormen van XSS	B3-1, B3-3, B3-4, B3-6
A3: Broken Authentication and Session Management	geen server-side invalidatie van sessies bij uitloggen; sessie-identifiers niet onvoorspelbaar; session fixation;	B3-2
A4: Insecure Direct Object References	autorisatiefouten	B3-2
A5: Cross-Site Request Forgery (CSRF)	geen Referer-check of geen nonce; indien wel nonce: nonce niet	B3-2

	onvoorspelbaar	
A6: Security Misconfiguration	TRACE-methode ingeschakeld, cookies niet Secure, cookies niet HttpOnly	B3-16, B5-2
A7: Insecure Cryptographic Storage	(buiten scope van dit artikel)	B5-1, B5-3, B5-4 (buiten scope van dit artikel)
A8: Failure to Restrict URL Access	autorisatiefouten	B3-2
A9: Insufficient Transport Layer Protection	SSLv2, Heartbleed, onvoldoende sterke ciphers, zwak sleutelmateriaal	B5-2, B3-16
A10: Unvalidated Redirects and Forwards	open redirection	B3-1

Afsluiting

Met Lektober nog vers in het geheugen, is het goed dat er een beveiligingsnorm is gekomen voor DigiD-gekoppelde systemen. Maar na twee jaar ervaring te hebben opgedaan met het toetsen van de technische normelementen, is duidelijk dat de eerste versie van de norm casu quo de richtlijnen waaruit de norm is opgebouwd aanpassing en nadere concretisering behoeven. De kinderziekten zouden zich wellicht gedeeltelijk kunnen laten verklaren door verhoogde tijdsdruk als gevolg van de politieke spanning die Lektober veroorzaakte. (Ter opfrissing: tientallen gemeenten werden acuut tijdelijk afgesloten van DigiD nadat tijdens Lektober hoge kwetsbaarheden zijn gevonden en volop media-aandacht kregen.) Waarschijnlijk zal de nieuwe versie van de NCSC-richtlijnen, die later dit jaar wordt verwacht, een opmaat zijn naar verbetering. Met dit artikel, waarvan een kopie is opgestuurd aan NCSC en Logius, hoop ik een kleine bijdrage te leveren aan deze ontwikkeling.

Dankwoord

De auteur dankt Ton van Deursen (Madison Gurkha), Jan Hendrickx (Madison Gurkha) en Maarten Hartsuijker (Classity) voor hun feedback op conceptversies van dit artikel.

[SSF] Framework Secure Software (SSF, 2014)
https://www.securesoftwarefoundation.org/FrameworkSecureSoftware.html

[Logius] Beveiliging webapplicaties (Logius, 2012)
http://www.logius.nl/producten/toegang/digid/logiusnlbeveiligingsassessments/

[NCSC1] ICT-beveiligingsrichtlijnen voor webapplicaties (NCSC, 2012)
https://www.ncsc.nl/dienstverlening/expertise-advies/kennisdeling/whitepapers/ict-beveiligingsrichtlijnen-voor-webapplicaties.html

[NCSC2] Inschalingsmatrix (NCSC, 2012) https://www.ncsc.nl/binaries/nl/dienstverlening/response-op-dreigingen-en-incidenten/beveiligingsadviezen-toelichting/1/Inschalingsmatrix.pdf

[NOREA] Handreiking ICT-beveiligingsassessments DigiD door RE's (NOREA, 2013)
http://www.norea.nl/Norea/Actueel/Nieuws/Handreiking+DigiD.aspx