Beveiligingsplan

"Virtueel medicijnkastje voor een Android smartphone"

Lennart Schellingerhout
Versie 1.1

Inhoudsopgave

Managementsamenvatting

Wanneer er omgegaan wordt met privacygevoelige gegevens, is het verstandig om een goede beveiliging te hebben om de privacy te kunnen waarborgen.
Omdat er bij de applicatie "Virtueel medicijnkastje voor de Android smartphone" sprake is van privacygevoelige gegevens, dient er voor de applicatie een goed passende beveiliging ontwikkelt te worden.

In de eerste 'sprint' van de ontwikkeling van de applicatie wordt er qua beveiliging vooral gekeken naar de verbinding tussen de telefoon en de back-end server. Er kan vanaf de telefoon een verzoek komen om gegevens te tonen van de medicijnen die er op de Nederlandse markt aanwezig zijn. Nu kunnen er gevallen zijn, waarbij de gebruiker van de applicatie wil dat een derde (mogelijk malafide) partij niet kan inzien welke medicijnen de gebruiker opzoekt.

Er zijn globaal gezien drie breed geaccepteerde maatregelen die geïmplementeerd kunnen worden in de applicatie. Het gebruik van symmetrische encryptie, het gebruik van asymmetrische encryptie of het gebruik van SSL (Secure Socket Layers).

Symmetrische encryptie heeft als voordelen dat een derde (mogelijk malafide) partij de gegevens niet kan lezen. Ook wordt één van de meest gebruikte algoritmes voor symmetrische encryptie, AES, door de Amerikaanse overheid gebruikt om zowel geclassificeerde als niet geclassificeerde te coderen. Dit algoritme is nog niet gekraakt.
Nadelen van symmetrische encryptie is het oversturen van de gedeelde geheime sleutel. Hiervoor moet een los protocol (Diffie-Hellman) gebruikt worden, die veel tijd kost om te implementeren. Bij het oversturen van gegevens dienen deze elke keer opnieuw gecodeerd en gedecodeerd te worden, wat de performance niet ten goede komt.

Asymmetrische encryptie heeft, dankzij het gebruik van openbare als geheime sleutels bij zowel de server als client, de mogelijkheid om unieke sleutelcombinaties te creëren. Hierdoor kan een derde (mogelijk malafide) partij niet tussen beide komen en de gecodeerde data begrijpen. Een ander voordeel is dat de server en client zich kunnen authenticeren naar elkaar.
Het grote nadeel van asymmetrische encryptie is de implementatie. Ook aan de kant van de client moet er een openbare en geheime sleutel aanwezig zijn. Dit zal standaard niet het geval zijn, dus moet deze eerst aangemaakt worden. Dit is een grote hoeveelheid werk en kan ook nadelen hebben op de performance.

SSL heeft als voordelen dat het uitwisselen van de certificaten geautomatiseerd is. Een ander voordeel is dat er een veilige 'tunnel' wordt gecreëerd, waar een derde (mogelijk malafide) partij geen toegang tot heeft. Daarnaast is SSL een veelgebruikte beveiligingstechniek. Onder andere grote banken gebruiken het voor het internetbankieren.
Enkele nadelen van SSL zijn de kosten voor het aanschaffen van een officieel certificaat. Dit moet gebeuren bij een CA (Certificate Authority) en kost geld. Dit hoeft echter niet wanneer er gebruik gemaakt wordt van een zelfgetekend certificaat. Een ander, relatief klein, nadeel is men wel afhankelijk is van de Android implementatie van SSL. Er is dus een afhankelijkheid van Android en Google bij de beveiliging. Nu is er op dit moment nog geen sprake van grote wantrouwen richting Google, maar wanneer daar veranderingen in komen, moet overwogen worden om van de Android SSL implementatie af te stappen.

Deze voor- en nadelen naast elkaar leggend is er tot de conclusie gekomen om voor de techniek SSL te kiezen. Dit is een techniek die zich bewezen heeft en gebruikt wordt door onder andere banken. Deze techniek is relatief eenvoudig te implementeren. Er kan gebruik gemaakt worden van een zelf getekend certificaat. Mocht er persoonsgebonden informatie op de backend server komen te staan, dan is het aan te raden om een certificaat aan te schaffen bij een Certificate Authority (CA).

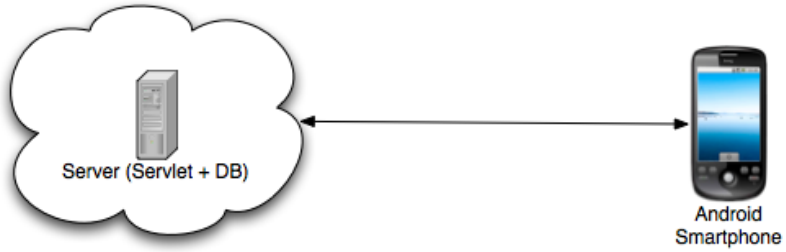
Er is bij de tweede sprint een database aangemaakt, waarin privacygevoelige data opgeslagen zit. Het is verstandig om gebruik te maken van encryptie. Wanneer encryptie over de gehele database zit, zorgt dit voor verminderde performance. Het is aan te raden alleen de date te coderen die privacygevoelig is.

Naast het gebruik van encryptie over de database is het verstandig om tijdens de verder ontwikkeling van de applicatie onderzoek te doen naar de volgende aspecten:

- Database overdracht tussen twee smartphones of naar een computer.
- Juridisch onderzoek naar de wetgeving omtrent het opslaan van persoonsgegevens.
- Onderzoek naar zwakheden in de gebruikte platformen (JBoss SEAM en Google Android).
- Onderzoek naar fysieke beveiliging.

Sprint 1: Communicatie van de telefoon met de backend server

De applicatie 'Virtueel medicijnkastje' voor de Android smartphone communiceert met een server die in de 'cloud' staat. Hiermee kan deze de door de gebruiker gevraagde gegevens uit de G-Standaard database ophalen. Hierbij moet gekeken worden welke beveiligingsmaatregelen zijn om de privacy van de gebruiker optimaal te garanderen en de integriteit van de data te kunnen garanderen.



Afbeelding 1: Communicatie tussen de server en de gebruiker

Risicoanalyse

Om een goede beveiliging op te kunnen zetten, moet er eerst een risicoanalyse gedaan worden. In een risicoanalyse wordt er gekeken of er informatie, systemen of processen zijn waarbij risico gelopen wordt. Zo kan het zijn dat bepaalde informatie altijd beschikbaar moet zijn om een essentieel bedrijfsproces door te laten gaan. Hierbij zou het risico de beschikbaarheid zijn.
Zo kan informatiebeveiliging in drie categorieën gedeeld worden: Beschikbaarheid (B), Integriteit (I) en Vertrouwelijkheid (V).
Per proces kan er bekeken worden of er op één van die drie categorieën risico wordt gelopen, dit gebeurt in de afhankelijkheidsanalyse.
Hierna wordt in de kwetsbaarheidsanalyse bekeken wat de kwetsbaarheden zijn per object van het systeem. Hiermee wordt in kaart gebracht welke bedreigingen relevant zijn en welke niet.
Wanneer dit in kaart gebracht is kan er een goede afweging gemaakt worden welke risico's zo goed mogelijk bestreden moeten worden. Dit wordt gedaan in de maatregelanalyse. Een overdaad aan beveiliging is net als een tekort aan beveiliging niet goed. Een overdaad kan een systeem trager maken en zal uiteraard extra kosten met zich mee brengen.

Afhankelijkheidsanalyse

De afhankelijkheidsanalyse heeft tot doel te bepalen hoe afhankelijk de processen zijn van de betrouwbaarheid van een informatiesysteem. Allereerst wordt het informatiesysteem beschreven en worden de/het proces(sen) die er gebruik van maken geïnventariseerd. Hierna wordt het belang van de/het geïnventariseerde proces(sen) voor de gebruikers van het systeem in kaart gebracht. Dan wordt het belang dat het informatiesysteem vertegenwoordigt voor de geïnventariseerde processen bekeken.
Als laatste wordt de informatie van de twee laatste stappen gecombineerd om zo het betrouwbaarheidsaspect te kunnen bepalen voor het systeem.

Het systeem bevat, zoals afbeelding 1 weergeeft, een server waar een database met medicijninformatie op draait. Daarnaast is er een applicatie voor een Android smartphone, die verbinding legt met deze server via het internet. Hiermee kan informatie uit de medicijndatabase opgehaald worden op de Android smartphone.
Er is in deze eerste sprint dus maar één proces, het ophalen van medicijninformatie. Dit wordt gedaan door de gebruiker op de Android smartphone die de applicatie heeft geïnstalleerd.

Nu het proces van de applicatie in kaart is gebracht, kan er gekeken worden welk belang de applicatie heeft voor de gebruiker. Dit zal worden gedaan worden voor de drie punten die de betrouwbaarheid aangeven: beschikbaarheid (B), integriteit (I) en vertrouwelijkheid (V).

Doordat de applicatie nu alleen ter ondersteuning dient voor de gebruiker, is het niet van essentieel belang, maar wenselijk, dat de informatie altijd beschikbaar is. De beschikbaarheid is hierdoor geschaald op ‘gemiddeld’.
Het is van groot belang dat de informatie die overgezonden worden wordt correct is. Een derde, mogelijk malafide, partij mag deze informatie niet bewerken voordat deze bij de gebruiker van de applicatie aankomt. Doseringen en bijwerkingen als invloed op rijvaardigheid mogen niet aangepast worden, omdat dit invloed kan hebben op de gezondheid van de gebruiker en in sommige situaties ook de omgeving van de gebruiker. De integriteit staat hierom ook geschaald op ‘zeer hoog’.
Als laatste wordt er gekeken naar de vertrouwelijkheid. Het is niet wenselijk dat iedereen mee kan kijken naar welke medicijnen er opgezocht worden door de gebruiker. Zeker wanneer er in een volgende fase gebruik gemaakt gaat worden van een medicijnkastje. Een derde, mogelijk malafide, partij kan, door middel van opgezochte medicijnen, patronen gaan herkennen en daarmee schatting

te doen van medicijngebruik van de gebruiker. In geval van chronisch zieken is het absoluut niet wenselijk dat een derde partij ter beschikking komt van deze medicijninformatie. Zeker met oog op de toekomst, waarbij het medicijnkastje regelmatig medicijninformatie ophaalt van de medicijnen uit het kastje, wordt patroonherkenning in medicijnen een stuk makkelijker. Sommige medicijnen zullen veel vaker dan andere opgevraagd worden en daaruit kan de conclusie getrokken worden dat de gebruiker van de applicatie het medicijn gebruikt. De vertrouwelijkheid is hierdoor geschaald op 'hoog'.
Zoals al genoemd is, is de applicatie nuttig voor de gebruiker en niet van essentieel belang. Dit heeft invloed op de te nemen maatregelen.

Wanneer het belang van het proces (BIV) gecombineerd wordt met het belang van het informatiesysteem (nuttig), kan men de betrouwbaarheidseisen opstellen. Met deze betrouwbaarheidseisen kan in kaart gebracht worden voor welke van de BIV factoren er beveiligingsmaatregelen genomen moeten worden.

Hiervoor wordt gebruik gemaakt van de onderstaande tabel.

	Vitaal	Nuttig	Support	Overbodig
Zeer Hoog	E	B	W	Gc
Hoog	B	W	Gc	Gc
Gemiddeld	W	Gc	Gc	Gc
Laag	Gc	Gc	Gc	Gc

Tabel 1: Matrix voor het bepalen van betrouwbaarheidseisen

Gc = Geen criterium B = Belangrijk W = Wenselijk

Uit de tabel kunnen we nu halen dat voor de factor beschikbaarheid 'geen criterium' geldt. Dit betekent dat er geen specifieke maatregelen voor de beschikbaarheid getroffen hoeven te worden.
De integriteit wordt geschaald op 'belangrijk'. Het is dus van belang dat bij het treffen van beveiligingsmaatregelen er gekeken wordt naar maatregelen die de integriteit helpen te bewaken.
De factor vertrouwelijkheid wordt gezet op 'wenselijk', wat inhoud dat het niet van essentieel belang is, maar zeker nuttig is om er beveiligingsmaatregelen voor te nemen.

Alle informatie uit de afhankelijkheidsanalyse kan nu uitgewerkt worden in één matrix.

	Proces	P1 *(medicijninforma ophalen)*
Informatiesysteem	Belang proces (B, I, V)	G ,Z ,H
Gebruikersproces	Belang informatiesyste voor het proces	Nuttig
	Betrouwbaarheid (B, I, V)	Gc, Bl, W

Tabel 2: Matrix met de resultaten van de afhankelijkheidsanalyse

B = Beschikbaarheid *I = Integriteit* *V = Vertrouwelijkheid*
G = Gemiddeld *Z = Zeer hoog* *H = Hoog*
Gc = Geen criterium *Bl = Belangrijk* *W = Wenselijk*

De gevonden betrouwbaarheidseisen vormen een set van drie waarden, die grafisch weergegeven kunnen worden zoals hieronder. De mate van oranje is een maat voor het belang van het betreffende betrouwbaarheidsaspect (beschikbaarheid, integriteit, vertrouwelijkheid).

Diagram 1: Diagram voor betrouwbaarheidseisen

Maatregelanalyse

In de maatregelanalyse wordt bepaald welke (beveiligings)maatregelen nodig zijn om het te beschouwen informatiesysteem zodanig te beveiligen dat alle risico's, die nog over zijn, acceptabel zijn.

De maatregelanalyse bevat een opsomming, waarin voor ieder object ia aangegeven tegen welke bedreigingen het betreffende object beveiligd dient te worden en welke beveiligingsniveaus daarbij nodig zijn.

Type object = Applicatie
- Bedreiging = Aanpassen data
  - Beveiligingsniveau = Hoog
  - Maatregelen: Encryptie/SSL
- Bedreiging = Afluisteren van data
  - Beveiligingsniveau = Gemiddeld
  - Maatregelen: Encryptie/SSL

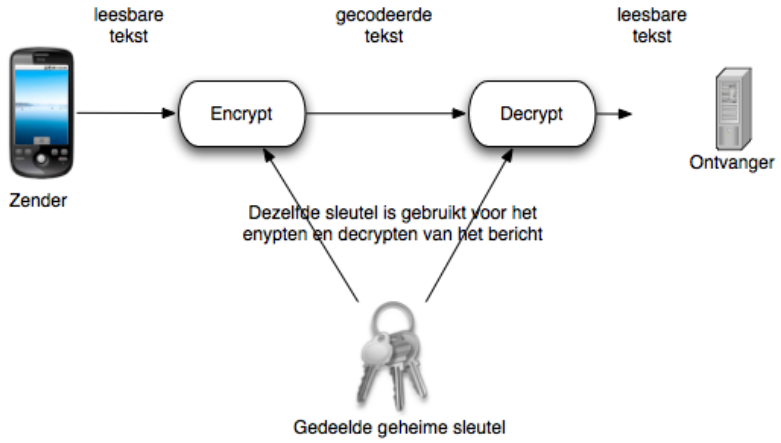
De maatregel die genomen moet worden is encryptie of SSL. Deze maatregelen zijn beide mogelijk, maar één kan geïmplementeerd worden. Hiervoor zullen beide maatregelen naast elkaar worden gezet en bekeken welke het beste is.

Encryptie

Encryptie is het coderen (versleutelen) van gegevens met behulp van een algoritme. Om de gecodeerde gegevens weer terug te krijgen moet deze gedecodeerd worden (decryptie). De twee meest gebruikte vormen van encryptie zijn symmetrische encryptie en asymmetrische encryptie.

Symmetrische encryptie

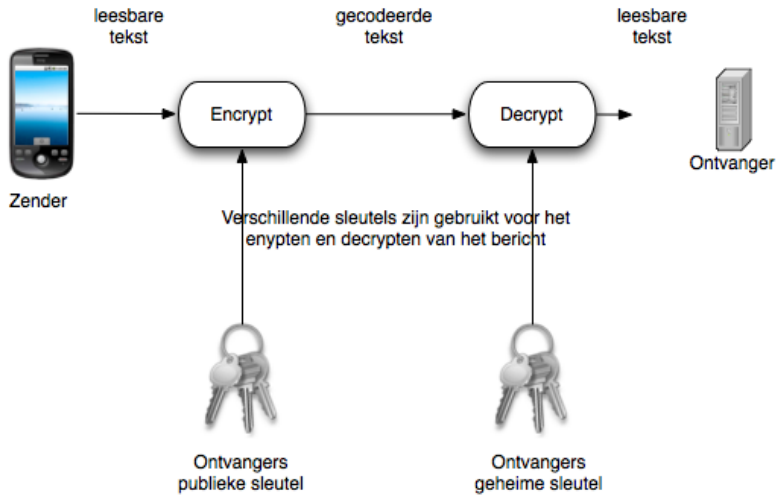
Bij symmetrische encryptie wordt de sleutel die gebruikt is om het bericht te coderen ook gebruikt om het bericht te decoderen. Dit betekent dat de ontvanger de sleutel al een keer van de zender moet hebben ontvangen. Dit moet gebeurt zijn op een 'veilige' manier, waarbij de zender heeft kunnen controleren of identiteit van de ontvanger juist is en of er geen onderschepping van de sleutel mogelijk is. Enkele voorbeelden van symmetrische encryptie zijn AES, DES, IDEA en RC4.



Afbeelding 2: De werking van symmetrische encryptie

Asymmetrische encryptie (public key encryption)

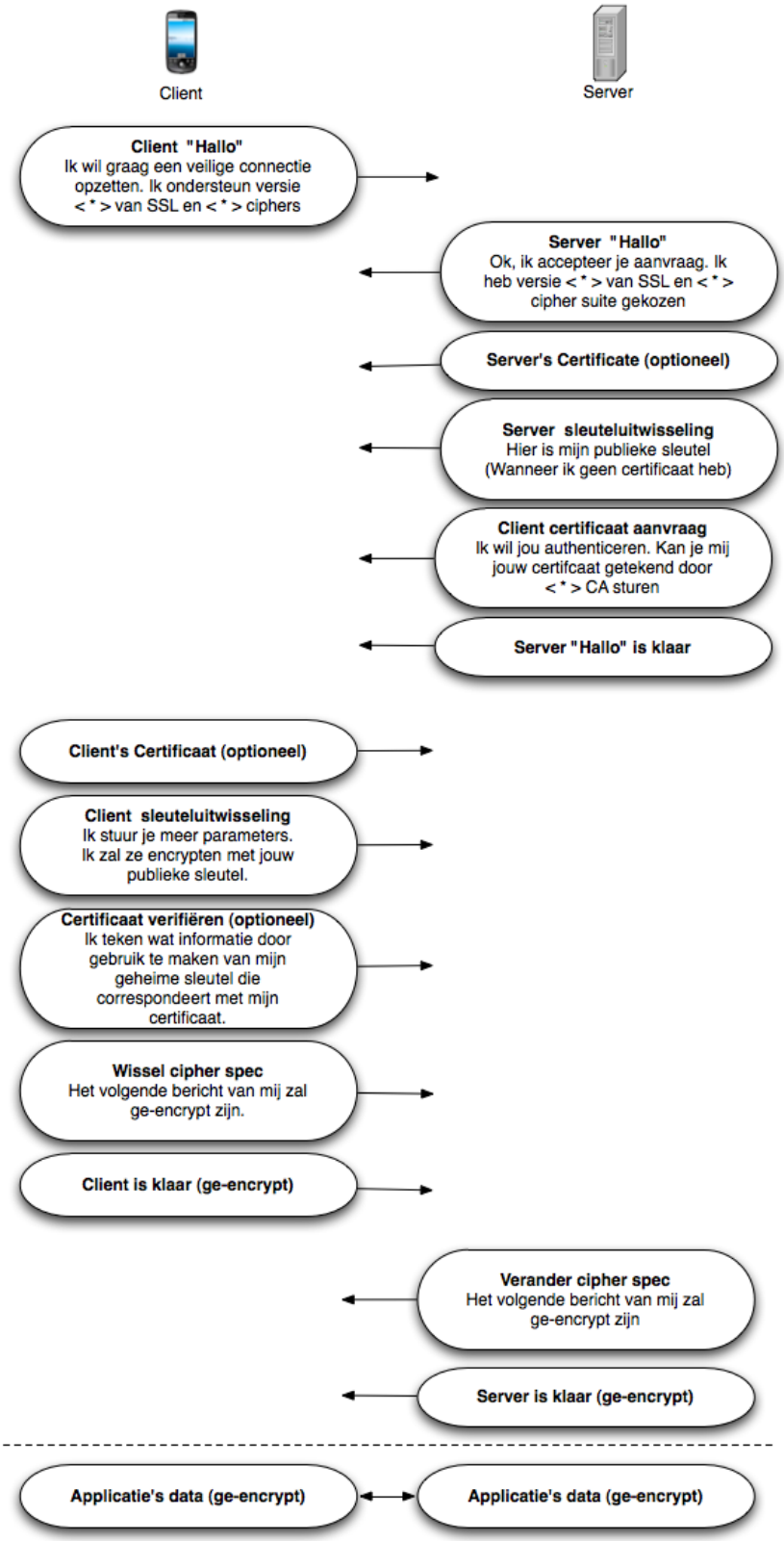
Bij asymmetrische encryptie wordt niet gebruikt gemaakt van één, maar van twee sleutels (publieke en geheime) bij zowel de zender als de ontvanger. De publieke sleutels (die voor iedereen toegankelijk zijn) van de zender en ontvanger worden uitgewisseld. De zender versleutelt de gegevens met behulp van de publieke sleutel van de ontvanger en zijn geheime sleutel. De ontvanger ontvangt de gecodeerde gegevens en kan deze decoderen met zijn geheime sleutel en de publieke sleutel van de ontvanger. Hierdoor hoeft de geheime sleutel niet meer overgestuurd worden, waardoor een groot beveiligingsrisico weg wordt genomen. De combinatie van de publieke sleutel van de ontvanger en de geheime sleutel van de zender is uniek, waardoor een derde partij niet zonder de geheime sleutel van de ontvanger de gegevens kan decoderen. Asymmetrische encryptie kan ook gebruikt worden om te controleren of de gegevens echt van de zender en niet van een derde partij afkomstig zijn. Hierbij worden de gegevens versleuteld door de zender met zijn eigen geheime sleutel. Hij stuurt met de gegevens ook zijn publieke sleutel mee naar de ontvanger. De ontvanger kan de gegevens met de publieke sleutel openen en daarmee controleren of de gegevens echt van de zender afkomstig zijn. Een voorbeeld van asymmetrische encryptie is PGP, die gebruik maakt van het RSA algoritme.



Afbeelding 3: De werking van asymmetrische encryptie

SSL (Secure Sockets Layer)

Secure Sockets Layer (SSL) is een encryptie-protocol die communicatie op het internet beveiligd. Dit protocol levert door middel van cryptografie een beveiligde verbinding en de authenticiteit kan er mee gecontroleerd worden. SSL encrypt delen van netwerkverbindingen op de 'applicatielaag'.
De client kan gebruik maken van de 'certificate's authority' (CA) publieke sleutel om de digitale handtekening van het server certificaat te kunnen controleren. De combinatie van encryptie en certificaat authenticiteit zorgt ervoor dat een 'man-in-the-middle attack' en het 'sniffen' vrijwel onmogelijk gemaakt wordt.



Afbeelding 4: De werking van SSL

Voordelen en nadelen

Voordelen symmetrisch encryptie

Wanneer er gebruik gemaakt wordt van symmetrische encryptie, dan zal er gekozen worden voor AES. AES is een veelgebruikt algoritme en zal ook gebruikt worden in de voor- en nadelen van symmetrische encryptie.

- Bij een eventuele tussenkomst van een malafide derde partij kan deze alsnog de data niet lezen, omdat deze gecodeerd is.
- Het kraken van het AES algoritme is tot nu toe onsuccesvol geweest.
- De Amerikaanse overheid gebruikt AES voor zowel geclassificeerde als ongeclassificeerde berichten. AES wordt dus als veilig aangenomen.

Nadelen symmetrisch encryptie

- De geheime sleutel moet eerst overgezonden worden (bijvoorbeeld met behulp van Diffie-Hellman). Dit proces is niet geautomatiseerd en gaat meer tijd kosten bij het ontwikkelen.
- Het uitwisselen van de geheime sleutel en het encrypten kost tijd. Dit kan een nadelige invloed hebben op de zoektijd (performance). Het encrypten zal bij elke opdracht opnieuw moeten worden gedaan.

Voordelen asymmetrisch encryptie

- Er wordt een unieke combinatie van geheime en openbare sleutels gebruikt, waardoor een derde (mogelijk malafide) partij niet tussen beide kan komen.
- Doordat er een unieke combinatie van geheime en openbare sleutels gebruikt wordt, kunnen de identiteiten gecontroleerd worden.
- Wanneer er een up-to-date versie van RSA gebruikt wordt, dan is er sprake van veilig codering van de data.

Nadelen asymmetrisch encryptie

- Zowel de server als de client moet een geheime en openbare sleutel bezitten. Aangezien de client van zichzelf nog geen sleutels heeft, moeten deze eerst aangemaakt worden. Dit is lastig te implementeren en vereist veel werk om werkend te krijgen.

Voordelen SSL

- De handshakingprocedure is geïntegreerd in SSL, dus deze hoeft niet los geïmplementeerd te worden.
- Er is een veilige 'tunnel' gecreëerd tussen de zender en de ontvanger. Een derde, mogelijk malafide, partij kan er dus niet bij.
- Veelgebruikte manier om een 'veilige' verbinding te creëren. Het wordt onder andere gebruikt voor internetbankieren door vele banken.

Nadelen SSL

- Er is een certificaat nodig om de server te kunnen laten authenticeren. Dit is een kostenpost, die alleen van toepassing is bij het opslaan van persoonsgegevens op de server.
- Afhankelijk van de SSL implementatie van Android. Wijzigingen in SSL heeft men dus niet zelf in de hand.

Conclusie

De voor- en nadelen van symmetrische encryptie, asymmetrische encryptie en SSL afwegend, is het meest verstandig om te gaan voor een SSL implementatie.
Allen hebben zich bewezen qua veiligheid, maar er zijn toch punten waarop symmetrische en asymmetrische encryptie afvallen.

Voor deze situatie, waarbij er een veilige verbinding gewenst is, is het meest logisch om te gaan voor SSL.
SSL heeft de handshakingprocedure en het uitwisselen van de certificaten geautomatiseerd, waardoor het makkelijker te implementeren is binnen de bestaande code.
Bij symmetrische en asymmetrische encryptie moet deze handshakingprocedure zelf aangemaakt worden.
Ook moet bij asymmetrische encryptie de client zelf een geheime en openbare sleutel aan moet gaan maken. Dit is zeer omslachtig en lastig te implementeren.

Een nadeel van SSL is aanschaffen van een certificaat bij een CA (Certificate Authority). Dit kost geld, maar is alleen aan te raden wanneer er persoonsgegevens op worden geslagen op de server. Er kan zelf een certificaat uitgegeven worden, maar daarmee kan de ontvanger nooit de ware identiteit van de zender controleren.

Testen

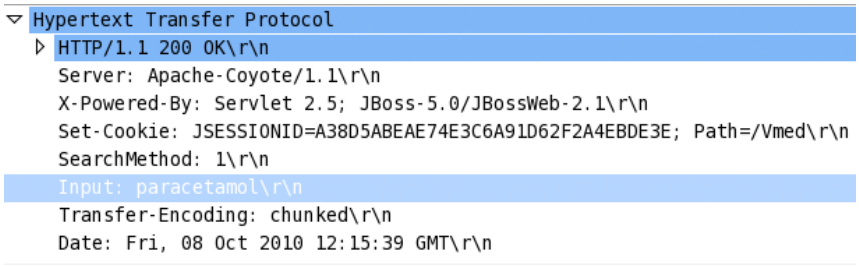
Wanneer SSL geïmplementeerd is in de applicatie en de server geconfigureerd is op het gebruikt van SSL, dan moet er gecontroleerd worden of het netwerkverkeer niet af te luisteren is. Hiervoor dient het netwerkverkeer bekeken te worden tijdens het gebruik van de applicatie. Wanneer er sprake is van SSL, dan zal de verstuurde data ge-encypt zijn en niet meer te achterhalen door de 'sniffer' (iemand die afluistert). Hieronder wordt zowel de situatie zonder SSL als de situatie met SLL beschreven, waarmee aangetoond wordt dat de verstuurde data niet meer begrijpbaar afgeluisterd kan worden.

Situatie voor de SSL implementatie

Allereerst moet er in de applicatie gezocht worden op een medicijn. In dit testgeval gaat er gezocht worden op een 'paracetamol' als 'medicijnnaam'.

Afbeelding 6: In de applicatie wordt gezocht op medicijnnaam 'paracetamol'

‘Medicijnnaam’ heeft als zoekmethode nummer ‘1’. In het netwerkverkeer moet dus ‘1’ en ‘paracetamol’ terug te vinden zijn.



Afbeelding 7: Netwerkverkeer op het moment dat er gezocht wordt

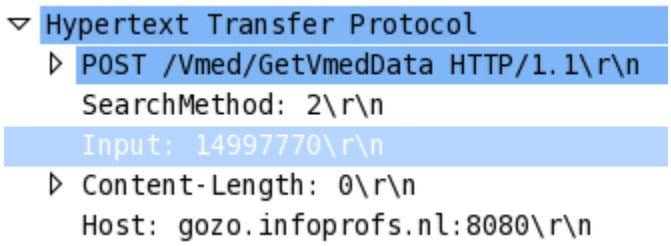
Uit de bovenstaande afbeelding uit Wireshark blijkt inderdaad de zoekmethode ‘1’ en de input ‘paracetamol’ te lezen zijn. De informatie van de telefoon naar de server toe is in de situatie zonder SSL dus leesbaar.

Nadat de zoekopdracht uitgevoerd is, is er data terug naar de telefoon gestuurd. De gebruiker kan nu het medicijn aanklikken waarvan deze de specifieke informatie wil verkrijgen.

Afbeelding 8: Medicijnresultaten wanneer er wordt gezocht op ‘paracetamol’

Nadat de gebruiker het specifieke medicijn heeft aangeklikt waarop gezocht moet worden, kan de telefoon doorgeven op welk specifiek medicijn moet worden gezocht. In dit testgeval gaat het medicijn ‘Daro Paracetamol vloeibaar v kind stroop 24Mg/Ml’. Er moet

daarvoor gezocht worden op het ZI-nummer. ZI-nummer is zoekmethode nummer '2' en de input zal daarvoor in dit geval '14997770' zijn.



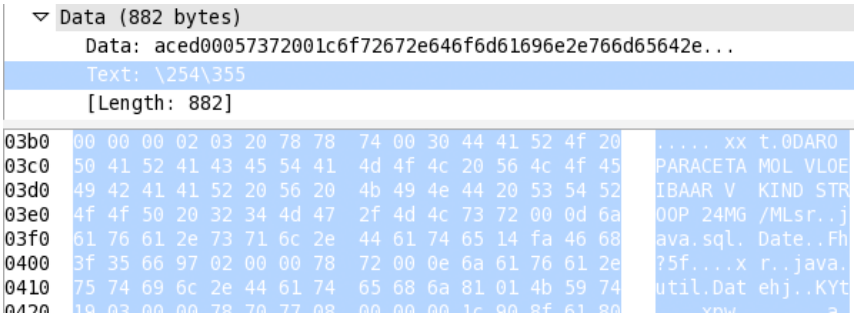
Afbeelding 9: Netwerkverkeer nadat er op een specifiek medicijn wordt geklikt

Ook deze data blijkt in het netwerkverkeer uitleesbaar te zijn. Dus alle data van de telefoon naar de server is leesbaar in het netwerkverkeer. Nu moet er nog gekeken worden of de data die vanaf de server naar de telefoon gaat ook leesbaar is.

De volgende stap in de applicatie is het laten zien van de informatie van het specifieke medicijn waarop gezocht is.

Afbeelding 10: Het scherm waarin een specifiek medicijn wordt getoond

Wanneer de data van de server naar de telefoon niet op een manier beveiligd is, dan zal de informatie die nu getoond wordt op het scherm uitleesbaar moeten zijn in het netwerkverkeer.



Afbeelding 11: Het netwerkverkeer van een specifiek medicijn vanaf de server

Er blijkt inderdaad informatie leesbaar te zijn die van de server naar de telefoon toe gaat. Er is in de afbeelding het medicijnnaam leesbaar waarop gezocht is.

Uit deze test kan geconcludeerd worden, dat bij het missen van beveiliging, de data uitleesbaar is met een programma waarmee netwerkverkeer bekeken kan worden. Dit geldt voor zowel data die van de telefoon naar de server toe gaat, als informatie die van de server naar de telefoon wordt gestuurd.

Situatie na de implementatie van SSL

Nu SSL geïmplementeerd is, moet er gekeken worden of er in het netwerkverkeer leesbaar is welke data er van de telefoon wordt verzonden naar de server en andersom.

Allereerst dient er dan weer gezocht te worden op een medicijnnaam. In dit geval gaan wij weer uit van 'paracetamol'.

Afbeelding 12: In de applicatie wordt gezocht op medicijnnaam 'paracetamol'

Nu deze informatie is verzonden, moet het netwerkverkeer bekeken worden.

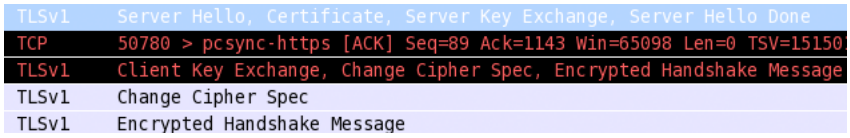
Protocol	Info
TCP	50780 > pcsync-https [ACK] Seq=1 Ack=1 Win=66240 Len=0 TSV=15150127
TCP	50780 > pcsync-https [PSH, ACK] Seq=1 Ack=1 Win=66240 Len=88 TSV=15
TCP	pcsync-https > 50780 [ACK] Seq=1 Ack=89 Win=5888 Len=0 TSV=11751992
TCP	pcsync-https > 50780 [PSH, ACK] Seq=1 Ack=89 Win=5888 Len=1142 TSV=
TCP	50780 > pcsync-https [ACK] Seq=89 Ack=1143 Win=65098 Len=0 TSV=1515
TCP	50780 > pcsync-https [PSH, ACK] Seq=89 Ack=1143 Win=66240 Len=158 T
TCP	pcsync-https > 50780 [PSH, ACK] Seq=1143 Ack=247 Win=6912 Len=6 TSV
TCP	pcsync-https > 50780 [PSH, ACK] Seq=1149 Ack=247 Win=6912 Len=45 TS

Afbeelding 13: Het SSL netwerkverkeer

Wanneer het netwerkverkeer nu bekeken wordt, is het niet duidelijk

leesbaar. Het enige wat er nu uit opgemaakt kan worden is dat er gebruik gemaakt wordt van https (SSL).

Wireshark bevat een functie die SSL kan 'decoden'. Hiermee kan Wireshark het SSL protocolproces leesbaar weergeven. Wanneer de data goed door SSL ge-encrypt is, moet deze niet te lezen zijn. Allereerst wordt er gekeken of de handshake procedure van SSL volledig doorlopen wordt.



Afbeelding 14: Het SSL netwerkverkeer nadat de decode functie in Wireshark is gebruikt

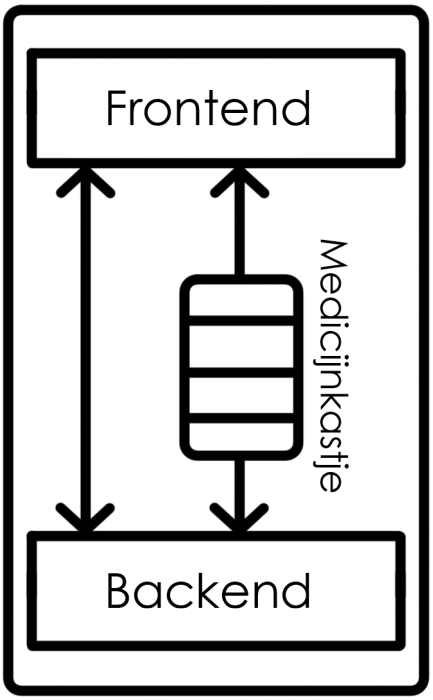
Uit de bovenstaande afbeelding is nu te lezen dat de handshake tussen de server en de telefoon goed doorlopen is. De server geeft zijn certificaat aan de telefoon (client) en de telefoon accepteer deze. Dit resulteert in de 'Encrypted Handshake Message'.
Nu moet alleen nog gecontroleerd worden of de data die de server naar de telefoon doorstuurt leesbaar is.

Afbeelding 15: De ge-encrypte data in SSL

Uit de bovenstaande afbeelding is niet te halen welke informatie er van de server naar de telefoon wordt gestuurd. Dit betekend dat de SSL-implementatie geslaagd is.

Sprint 2: Database op de smartphone

Nu er verbinding gemaakt kan worden van de telefoon met een server, moet er informatie opgeslagen gaan worden op de telefoon. Dit gebeurt door middel van een database die zich op de telefoon bevindt.
Omdat er persoonlijke informatie opgeslagen gaat worden in de database, moet er gekeken worden of er informatie in de database is die beveiligd moet worden. Hierbij zullen vooral de privacy en integriteit van groot belang zijn.
Er zal, net als bij sprint 1, een risicoanalyse uitgevoerd worden die bestaat uit een afhankelijkheidsanalyse en een maatregelanalyse. Deze analyse gaat alleen over de database 'medicijnkastje'.
Hieruit zal een advies volgen die in een eventuele volgende sprint geïmplementeerd kan worden. Daarnaast zullen er nog enkele andere adviezen en aanbevelingen volgen voor nog mogelijk komende sprints.



Afbeelding 16: Schematisch weergave van de smartphone met het medicijnkastje

Risicoanalyse

Om een goede beveiliging op te kunnen zetten, moet er eerst een risicoanalyse gedaan worden. In een risicoanalyse wordt er gekeken of er informatie, systemen of processen zijn waarbij risico gelopen wordt. Zo kan het zijn dat bepaalde informatie altijd beschikbaar moet zijn om een essentieel bedrijfsproces door te laten gaan. Hierbij zou het risico de beschikbaarheid zijn.
Zo kan informatiebeveiliging in drie categorieën gedeeld worden: Beschikbaarheid (B), Integriteit (I) en Vertrouwelijkheid (V).
Per proces kan er bekeken worden of er op één van die drie categorieën risico wordt gelopen, dit gebeurt in de afhankelijkheidsanalyse.
Hierna wordt in de kwetsbaarheidsanalyse bekeken wat de kwetsbaarheden zijn per object van het systeem. Hiermee wordt in kaart gebracht welke bedreigingen relevant zijn en welke niet.
Wanneer dit in kaart gebracht is kan er een goede afweging gemaakt worden welke risico's zo goed mogelijk bestreden moeten worden. Dit wordt gedaan in de maatregelanalyse. Een overdaad aan beveiliging is net als een tekort aan beveiliging niet goed. Een overdaad kan een systeem trager maken en zal uiteraard extra kosten met zich mee brengen.

Afhankelijkheidsanalyse

De afhankelijkheidsanalyse heeft tot doel te bepalen hoe afhankelijk de processen zijn van de betrouwbaarheid van een informatiesysteem. Allereerst wordt het informatiesysteem beschreven en worden de/het proces(sen) die er gebruik van maken geïnventariseerd. Hierna wordt het belang van de/het geïnventariseerde proces(sen) voor de gebruikers van het systeem in kaart gebracht. Dan wordt het belang dat het informatiesysteem vertegenwoordigt voor de geïnventariseerde processen bekeken.
Als laatste wordt de informatie van de twee laatste stappen gecombineerd om zo het betrouwbaarheidsaspect te kunnen bepalen voor het systeem.

De smartphone heeft, zoals afbeelding 16 weergeeft, naast de frontend (GUI) en de backend (verbinding met server) ook een database gekregen: medicijnkastje. In dit medicijnkastje kan de gebruiker van de applicatie naast medische informatie ook persoonlijke informatie opslaan. Hierbij moet gedacht worden aan medicijnen die een persoon zich toedient. Van dit medicijn kan dan ook aangegeven worden hoeveel van een medicijn een persoon per toediening toedient en wat de frequentie daarvan is per dag. Er wordt een medicijnagenda bijgehouden.

Nu het proces van de applicatie in kaart is gebracht, kan er gekeken worden welk belang de applicatie heeft voor de gebruiker. Dit zal worden gedaan worden voor de drie punten die de betrouwbaarheid aangeven: beschikbaarheid (B), integriteit (I) en vertrouwelijkheid (V).

In tegenstelling tot de applicatie van sprint 1, wordt er nu een medicijnagenda bijgehouden. De beschikbaarheid van de database met daarin de medicijninformatie en wanneer een medicijn toegediend moet worden is nu zeer belangrijk. De beschikbaarheid is hierdoor geschaald op ‘zeer hoog’.
Net als bij de applicatie van sprint 1 is het van groot belang dat de informatie die in de database opgeslagen staat integer is. Fouten in deze database kunnen desastreuze gevolgen hebben voor de medicijngebruiker. De integriteit staat hierom ook geschaald op ‘zeer hoog’.
Als laatste wordt er gekeken naar de vertrouwelijkheid. Het is niet wenselijk dat iedereen mee kan kijken naar welke medicijnen er opgezocht worden door de gebruiker. Wanneer de smartphone kwijt raakt, dan moet de vinder van de smartphone op geen enkele manier bij privacygevoelige data kunnen komen. De vertrouwelijkheid is hierdoor geschaald op ‘zeer hoog’.

Wanneer het belang van het proces (BIV) gecombineerd wordt met het belang van het informatiesysteem (nuttig), kan men de betrouwbaarheidseisen opstellen. Met deze betrouwbaarheidseisen kan in kaart gebracht worden voor welke van de BIV factoren er beveiligingsmaatregelen genomen moeten worden.

Hiervoor wordt gebruik gemaakt van de onderstaande tabel.

	Vitaal	Nuttig	Support	Overbodig
Zeer Hoog	E	B	W	Gc
Hoog	B	W	Gc	Gc
Gemiddeld	W	Gc	Gc	Gc
Laag	Gc	Gc	Gc	Gc

Tabel 3: Matrix voor het bepalen van betrouwbaarheidseisen

Gc = Geen criterium B = Belangrijk W = Wenselijk

Uit de tabel kunnen we nu halen dat voor de factor beschikbaarheid 'belangrijk' geldt. Er moeten dus maatregelen genomen worden betreft de beschikbaarheid.
De integriteit wordt geschaald op 'belangrijk'. Het is dus van belang dat bij het treffen van beveiligingsmaatregelen er gekeken wordt naar maatregelen die de integriteit helpen te bewaken.
De factor vertrouwelijkheid wordt gezet op 'belangrijk'. Ook hier moet dus goed gekeken worden naar maatregelen.

Alle informatie uit de afhankelijkheidsanalyse kan nu uitgewerkt worden in één matrix.

	Proces	P1 *(medicijnkastje)*
Informatiesysteem	Belang proces (B, I, V)	Z ,Z ,Z
Gebruikersproces	Belang informatiesyste voor het proces	Nuttig
	Betrouwbaarheid (B, I, V)	Bl, Bl, Bl

Tabel 4: Matrix met de resultaten van de afhankelijkheidsanalyse

B = Beschikbaarheid I = Integriteit V = Vertrouwelijkheid
G = Gemiddeld Z = Zeer hoog H = Hoog
Gc = Geen criterium Bl = Belangrijk W = Wenselijk

De gevonden betrouwbaarheidseisen vormen een set van drie waarden, die grafisch weergegeven kunnen worden zoals hieronder. De mate van oranje is een maat voor het belang van het betreffende

betrouwbaarheidsaspect (beschikbaarheid, integriteit, vertrouwelijkheid).

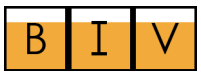


Diagram 2: Diagram voor betrouwbaarheidseisen

Maatregelanalyse

In de maatregelanalyse wordt bepaald welke (beveiligings)maatregelen nodig zijn om het te beschouwen informatiesysteem zodanig te beveiligen dat alle risico's, die nog over zijn, acceptabel zijn.

De maatregelanalyse bevat een opsomming, waarin voor ieder object ia aangegeven tegen welke bedreigingen het betreffende object beveiligd dient te worden en welke beveiligingsniveaus daarbij nodig zijn.

Type object = Applicatie
- Bedreiging = Aanpassen database
  - Beveiligingsniveau = Hoog
  - Maatregelen: Encryptie

- Bedreiging = Ongeautoriseerd lezen van de database
  - Beveiligingsniveau = Hoog
  - Maatregelen: Encryptie

De maatregel die genomen moet worden is encryptie. Er zal gekeken moeten worden welke data privacygevoelig is en op welke manier encryptie het beste geïmplementeerd kan worden.

Encryptie medicijnkastje database

Uit de risicoanalyse is gebleken dat er door het opslaan van privacygevoelige gegevens de beveiliging aanzienlijk groter moet zijn. Om de integriteit en vertrouwelijkheid te kunnen bewaken van de gegevens, is er voor gekozen om encryptie op de database te gaan gebruiken. Wanneer de smartphone kwijt raakt, dan kan deze privacygevoelige data niet door een derde gelezen worden. Ook niet wanneer men buiten de applicatie om de database weet te benaderen.
Het is echter niet verstandig om de gehele database te encrypten. Encryptie en decryptie kost extra tijd en het gaat ten koste van de performance van de applicatie, wanneer de gehele database onder encryptie zit.
Daarom is het verstandiger uit te zoeken welke velden in de database privacygevoelige informatie bevatten. Op deze velden kan encryptie gebruikt worden.

Veld (Tabel)
Wachtwoord (Gebruiker)
Dagboekentry (Dagboek)
Datumstart (Verzekeringscontract)
Datumeind (Verzekeringscontract)
OmschrijvingAllergie (Allergie)
ReactiePatient (PatientAllergie)
Naam (Patient)
Geboortedatum (Patient)
Lengte (Patient)
Gewicht (Patient)
Naam (Contact)
Postcode (Contact)
Huisnummer (Contact)
Toevoeging (Contact)
Straat (Contact)
Telefoonnummer (Contact)
TelefoonnummerRecept (Contact)
EmailContact (Contact)
EmailHerhaalrecept (Contact)

Tabel 5: Velden uit de medicijnkast database die privacygevoelig zijn

Wanneer er data toegevoegd wordt aan een veld waar encryptie vereist is, dan moet deze data door een encrypter heen. Wanneer deze toegevoegde data weer opgehaald dient te worden, moet deze gecodeerde data gedecodeerd worden.

Codevoorbeeld encryptie in Android

Aangezien er geen implementatie van gemaakt gaat worden van encryptie op de database door tijdgebrek, wordt hierbij een code snippet gegeven van encryptie in Android.

```
package net.sf.andhsli.hotspotlogin;

import java.security.SecureRandom;

import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;

/**
 * Usage:
 * <pre>
 * String crypto = SimpleCrypto.encrypt(masterpassword, cleartext)
 * ...
 * String cleartext = SimpleCrypto.decrypt(masterpassword, crypto)
 * &lt;/pre&gt;
 * @author ferenc.hechler
 */
public class SimpleCrypto {

	public static String encrypt(String seed, String cleartext) throws
Exception {
		byte[] rawKey = getRawKey(seed.getBytes());
		byte[] result = encrypt(rawKey, cleartext.getBytes());
		return toHex(result);
	}

	public static String decrypt(String seed, String encrypted) throws
Exception {
		byte[] rawKey = getRawKey(seed.getBytes());
		byte[] enc = toByte(encrypted);
		byte[] result = decrypt(rawKey, enc);
		return new String(result);
	}

	private static byte[] getRawKey(byte[] seed) throws Exception {
		KeyGenerator kgen = KeyGenerator.getInstance("AES");
		SecureRandom sr = SecureRandom.getInstance("SHA1PRNG");
		sr.setSeed(seed);
	   kgen.init(128, sr); // 192 and 256 bits may not be available
	   SecretKey skey = kgen.generateKey();
	   byte[] raw = skey.getEncoded();
	   return raw;
	}

	private static byte[] encrypt(byte[] raw, byte[] clear) throws Exception {
	   SecretKeySpec skeySpec = new SecretKeySpec(raw, "AES");
		Cipher cipher = Cipher.getInstance("AES");
	   cipher.init(Cipher.ENCRYPT_MODE, skeySpec);
```

```
        byte[] encrypted = cipher.doFinal(clear);
                return encrypted;
        }

        private static byte[] decrypt(byte[] raw, byte[] encrypted) throws
Exception {
        SecretKeySpec skeySpec = new SecretKeySpec(raw, "AES");
                Cipher cipher = Cipher.getInstance("AES");
        cipher.init(Cipher.DECRYPT_MODE, skeySpec);
        byte[] decrypted = cipher.doFinal(encrypted);
                return decrypted;
        }

        public static String toHex(String txt) {
                return toHex(txt.getBytes());
        }
        public static String fromHex(String hex) {
                return new String(toByte(hex));
        }

        public static byte[] toByte(String hexString) {
                int len = hexString.length()/2;
                byte[] result = new byte[len];
                for (int i = 0; i &lt; len; i++)
                        result[i] = Integer.valueOf(hexString.substring(2*i, 2*i+2),
16).byteValue();
                return result;
        }

        public static String toHex(byte[] buf) {
                if (buf == null)
                        return "";
                StringBuffer result = new StringBuffer(2*buf.length);
                for (int i = 0; i &lt; buf.length; i++) {
                        appendHex(result, buf[i]);
                }
                return result.toString();
        }
        private final static String HEX = "0123456789ABCDEF";
        private static void appendHex(StringBuffer sb, byte b) {

        sb.append(HEX.charAt((b&gt;&gt;4)&amp;0x0f)).append(HEX.charAt(
b&amp;0x0f));
        }

}

// see http://androidsnippets.com/encryptdecrypt-strings
```

Aanbevelingen betreffende onderzoek

Naast het gebruik van encryptie op de database als aanbeveling, zijn er nog meer aanbevelingen die gedaan kunnen worden. Deze zullen tijdens toekomstige sprints beter onderzocht moeten worden. Deze aanbevelingen betreffen niet alleen de verbinding tussen de smartphone en de server of het medicijnkastje database, maar de gehele omgeving.
Deze aanbevelingen tot onderzoek zullen kort aangestipt worden met uitleg/argumentatie.

Uitwisseling van gegevens uit de database

Er wordt aangeraden om gebruik te maken van encryptie in de database. Deze encryptie wordt in werking gesteld na autorisatie in de applicatie. Dit gebeurt door middel van een gebruikersnaam en wachtwoord in te voeren. Wanneer men de database wilt overzetten naar een andere smartphone of computer, dan zal de privacygevoelige informatie eerst gedecodeerd moeten worden voordat deze naar een andere database overgeplaatst kan worden. Wanneer de data gecodeerd in een andere database komt te staan, dan kan de data niet gedecodeerd worden. De applicatie op de smartphone kan door iemand anders worden beheerd en heeft waarschijnlijk een andere gebruikersnaam en wachtwoord. Om de transitie van privacygevoelige data niet in gevaar te brengen, moet er gekeken worden of er een manier is waarop ook de overdracht veilig kan. Hierbij kan gedacht worden aan een gezamenlijke encryptie van de twee partijen, waarbij ze een gezamenlijk wachtwoord hebben. Wanneer de data aankomt bij de nieuwe database, dan kan deze weer gecodeerd worden met de encryptie van de gebruiker van die database. Door een gezamenlijke encryptie te gebruiken tussen de twee partijen, wordt voorkomen dat een derde partij tijdens de overdracht privacygevoelige data kan lezen.

Juridisch onderzoek

Omdat er privacygevoelige informatie opgeslagen gaat worden, is het verstandig om gedegen onderzoek te doen naar de juridische kant van het opslaan van deze data. Er zijn wettelijke bepalingen betreft het opslaan van privacygevoelige gegevens. Het is dan ook verstandig om een expert op dit gebied te gaan interviewen. Deze kan aangeven of de data wel opgeslagen mag worden en welke mate van beveiliging voldoende is.

Onderzoek doen naar zwakheden in de platformen

Er moet gedegen onderzoek gepleegd gaan worden naar de zwakheden in de platformen die gebruikt worden. Dit zijn JBoss SEAM en Google Android.
In de media duiken er steeds regelmatiger berichten op over malware die zich installeert op Android smartphones. Aangezien de gebruikte platformen ook zwakheden hebben, is het verstandig deze goed in kaart te brengen. Hiermee kan bekeken worden of het wel verstandig is om het platform te gebruiken of dat er extra beveiligingsmaatregelen moeten komen.

Onderzoek naar fysieke beveiliging

Het is ook aan te raden om onderzoek te doen naar de fysieke beveiliging. Er moet goed bekeken worden waar de server(s) met de G-Standaard database geplaatst gaat worden. Dit moet, om de integriteit te kunnen waarborgen, zijn in een ruimte waar ook sprake is van fysieke beveiliging. Ook zal er gekeken moeten worden naar rollenscheiding en wie er bevoegd zijn om aanpassingen te mogen maken aan de database.