

Project ‘beveiligde pasjes’

Hotelbeveiliging

Door Joël Ledelay, Jeroen Weener, Jan-Willem Nijhuis, Jetse Schoffelmeer, Menno Schober en Dennis Cai

Projectgroep 18 | Parels der Informatica | 27 oktober 2014 | Universiteit Twente, Enschede

Inhoudsopgave

[Requirements 2](#_Toc402303580)

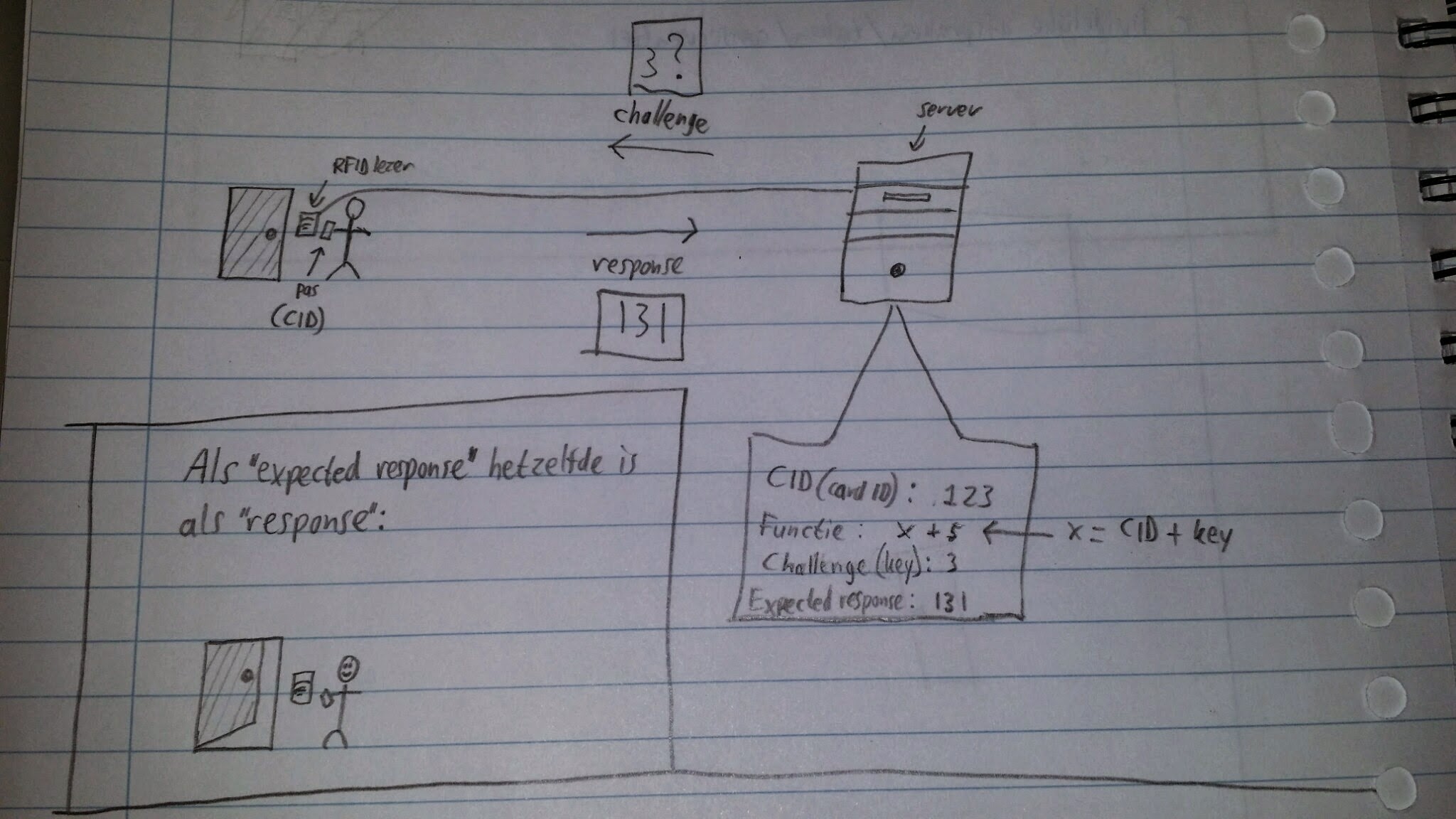
[Ontwerp 3](#_Toc402303581)

[Ontwerpkeuzes 3](#_Toc402303582)

# Requirements

* De veiligheid van iedereen die aanwezig is in het gebouw blijft gewaarborgd. Bij gevaar moeten alle deuren automatisch opengaan, bijvoorbeeld in geval van brand. Tevens mag het systeem niet gemakkelijk te kraken zijn.
* Er bestaan verschillende soorten autoriteit voor de pasjes, zodat er alleen bevoegde mensen op bepaalde plaatsen kunnen komen.
* De data op de pasjes is beveiligd, omdat eventuele kwaadwillenden de informatie op de pasjes eventueel kunnen misbruiken.
* Pasjes kunnen worden uitgeschakeld/geblokkeerd in het geval dat deze kwijt raken of gestolen worden.
* Er kan worden bijgehouden wie er op welk tijdstip incheckt, om deze data te kunnen gebruiken in geval van diefstal, fysiek geweld, etc.
* M~~edewerkers krijgen alleen tijdens werktijden toegang tot de geautoriseerde ruimtes.~~
* De data op de pasjes kan snel gelezen worden. Dit houdt de klant tevreden.
* Pasjes moeten geactiveerd kunnen worden door de gebruiker, en er kan een bepaalde graad van autoriteit op geplaatst worden.
* De database heeft een hoge capaciteit.

# Ontwerp



Als een iemand in het hotel zijn/haar pasje scant, dan wordt er een “handshake” naar de server verstuurd. De server stuurt vervolgens een “challenge” naar de RFID-scanner (Arduino). Deze challenge is een willekeurig integer tussen twee vaste waarden. Vervolgens berekent de Arduino de “response” op de challenge, zodat de server kan verifiëren dat het pasje de juiste autorisatie heeft. Deze response wordt berekent door middel van een functie met als variabelen de Card-ID en de challenge (key). De server decrypt vervolgens de ontvangen ciphertext en checkt of de Card-ID in de database voorkomt, en welke rechten hiermee verbonden zijn.

# Ontwerpkeuzes

Er zijn talloze eisen die gesteld kunnen worden bij dit project, maar veiligheid heeft daarvan de hoogste prioriteit. Het is essentieel dat de veiligheid van de gasten alsmede de medewerkers van het hotel gewaarborgd blijft. In het interview dat gehouden is met een belangrijke stakeholder, een hotelbewaker, is dit dan ook sterk naar voren gekomen.

Als authenticatiemethode is er gekozen voor het “challenge and response” protocol, vanwege het feit dat dit een zeer veilige manier is om er voor te zorgen dat de server zeker weet dat het met de scanner communiceert, en niet met een hacker bijvoorbeeld. Er wordt een “challenge” verstuurd vanaf de server (Python), waarna de RFID-lezer (Arduino) aan de hand van de Card-ID van het pasje de “response” berekend en terugstuurt naar de server. Dit systeem is veilig omdat er elke keer een nieuwe challenge wordt verstuurd vanaf de server en de key veel mogelijke waarden kan aannemen, waardoor de response steeds zal verschillen. Het heeft voor een eventuele aanvaller dus geen nut om de response op te nemen en weer af te spelen, aangezien deze elke keer weer verschillend is. Zolang de functie waarmee versleuteld wordt onbekend is zal dit systeem veilig zijn.

Er zijn verschillende lagen van autoriteit. Het is niet de bedoeling dat gasten overal kunnen komen, zoals in kantoren, opslagruimtes en dergelijke. Deze ruimtes zijn alleen voor de medewerkers toegankelijk. Hetzelfde geldt voor bewakers/receptiemedewerkers en dergelijke. Deze kunnen elk hun eigen gebieden wel in maar niet per se elke kamer. Ook dit wordt geïmplementeerd met het oog op veiligheid.

Snelheid is ook een factor om rekening mee te houden. Voor de directie is het uiteraard belangrijk dat de hotelgasten tevreden blijven. Niemand houdt ervan om te moeten wachten, zelfs als het om enkele seconden extra gaat. Er moet echter allereerst gezorgd worden dat het systeem überhaupt werkt, daarna wordt er gekeken naar de stroomlijning van het systeem.



Als extra uitdaging is er een GUI gemaakt, om het openen en sluiten van deuren te kunnen simuleren. Er is hierboven een screenshot van de alfaversie toegevoegd. Tevens kan er hier worden gezocht in de database naar bestaande gegevens, en er kunnen hiermee gegevens worden toegevoegd.

Tot slot moeten de tijd en plaats waarop mensen inchecken opgeslagen kunnen worden. Dit is handige informatie mocht er zich een ongeval of misdrijf of iets dergelijks voordoen. Dit is echter niet essentieel, dus deze functionaliteit wordt alleen geïmplementeerd indien er voldoende tijd over is.

Mankementen:

* Hoewel er sprake is van encryptie, is er nog steeds sprake van een vrij gelimiteerde key, wanneer men het vaak genoeg probeert zul je herhaling gaan zien aangezien de key tot nu toe slechts 80000 opties heeft.
* De pasjes kunnen relatief gemakkelijk gekopieerd worden aangezien we de data op de pasjes niet aanpassen, waarna de gebruiker zolang het Card ID actief is altijd een deur binnen zal kunnen komen.
* De database is niet beveiligd of encrypted, dus wanneer iemand in deze database terecht komt kan deze zo de ID’s van alle gebruikers aflezen evenals de geschiedenis.
* De Arduino kan worden uitgelezen en daar staat de formule in voor de key die wordt toegepast op de Card ID, dan kan iemand met de key en de ciphertext het Card ID makkelijk uitvinden.
* Ook kan iemand met toegang tot de Python-code de formule terug vinden. Tevens kunnen ze ook het wachtwoord tot het programma in de code vinden aangezien deze in de code staat.
* Wanneer iemand het brandalarm af laat gaan zullen alle deuren automatisch open gaan en kan toegang worden verkregen tot alle ruimtes.

Exceptionele situaties:

Bij een brand moeten alle deuren automatisch open gaan, veiligheid gaat voor beveiliging vinden wij.

De gebruiker kan een backup van de database op een andere locatie neerzetten om de gegevens te beschermen, maar voor de rest zijn er geen bijzondere situaties waarbij ons systeem blijft functioneren.