P6 Companion Standard

Standaardisering communicatieprotocol P1-devices

Versie 1.1.0

Dit rapport is geschreven door Diderik van Wingerden en Frans van der Schrier en is eigendom van Enexis.

‘s Hertogenbosch, 24 oktober 2014

Inhoudsopgave

Versiebeheer

Dit hoofdstuk bevat informatie over het versiebeheer van dit specificatiedocument en de aan de P6 Companion Standard-gerelateerde hardware- en softwareproducten. Elk product heeft in principe zijn eigen versiebeheer. Dit omdat de beheerder, het updateproces en de updatefrequentie per product verschilt.

In dit hoofdstuk is te lezen welke versies van de producten relateren aan de specificaties in dit document.

Tabel 1: versiebeheer van producten P6 Companion Standard

Product	Versie	Beheerder	Versiemethode
P6 Companion Standard	1.0.0	Diderik van Wingerden	Zie verder (1)
Open API	1.0	Diderik van Wingerden	Zie verder (2)
iPhone (iOS) Referentieapp	1.1.17	Total Active Media	Zie verder (3)
Android Referentieapp	1.1.17	Total Active Media	Zie verder (3)
Windows Referentieapp	0.97	Fflowing	+0.01 bij een update
Simulator	0.97	Fflowing	+0.01 bij een update
SDK	0.5	Diderik van Wingerden	Zie verder (4)

(1) Versiemethode P6 Companion Standard
De methode van versienummers voor dit document, de P6 Companion Standard, is “x.y.z”, waarbij x een definitieve vastgestelde versie representeert. De y geeft incrementele specificatiewijzigingen aan binnen de vastgestelde versie die relateren aan de Open API. De z geeft een volgnummer aan voor versies van het document die niet relateren aan specificatiewijzigingen van de Open API (bijvoorbeeld: lay-outverbeteringen).

(2) Versiemethode Open API
De specificatie van de Open API is het belangrijkste onderdeel van de P6 Companion Standard. De versienummering voor de Open API is “x.y” en loopt gelijk met de “x.y” van de P6 Companion Standard. De x geeft een zogenaamde ‘major release’ en y een ‘minor release’ aan. Een major release geeft aan dat er grote veranderingen hebben plaatsgevonden (Open API niet meer compatibel met vorige versie), zoals wijzigingen in de bestaande API en toevoegingen van grote functionaliteiten. Een minor release geeft aan dat er fixes, patches, kleine aanpassingen en functionele wijzigingen met beperkte impact zijn toegevoegd (Open API nog compatibel met vorige versie).

Hierdoor kan een app die bijvoorbeeld gemaakt is voor Open API versie 1.0 nog blijven werken op Open API versie 1.1, maar niet op Open API versie 2.0. Schematisch zie dit er als volgt uit, zie tabel 1:

Tabel 2: major en minor versies Open API en compatibiliteit van apps

Versie Open API	App gemaakt voor versie Open API	Functioneert de app?
1.0	1.0	Ja
1.1	1.0	Ja
1.0	1.1	Ja
2.x	1.x	Nee
1.x	2.x	Nee

(3) Versiemethode iPhone en Android Referentieapps
Het versienummer van deze apps is in principe identiek omdat de apps op hetzelfde ontwikkelplatform (Titanium) worden gemaakt.

De methode van versienummers voor de iPhone en Android apps is “x.y.z.i”:
x = major version

y = feature release
z = sprint
i = hotfix

Hierin is te beredeneren dat x grote wijzigingen betreft, zoals een compleet nieuwe UI. Extra features (y), zijn kleinere zaken, zoals bijvoorbeeld inzicht in de opbrengst van zonnecollectoren. Z zijn onderdelen van features of refactoring, optimalisatie en/of bugfixrondes; i wordt gebruikt voor hotfixes. Deze worden gedaan wanneer een app in productie een kritische fout bevat.

(4) Versiemethode SDK
De methode van versienummers voor de SDK is “x.y”: x voor een ‘major release’ (grote wijziging) aan de SDK en y voor ‘minor releases’ (kleine veranderingen of verbeteringen). Op dit moment is de SDK nog in pre-release, daarom heeft deze een nummering met major release ‘0’.

Zie bijlage H voor de volledige versiehistorie van dit document en de oorspronkelijke specificatiedocumenten.

1 Inleiding

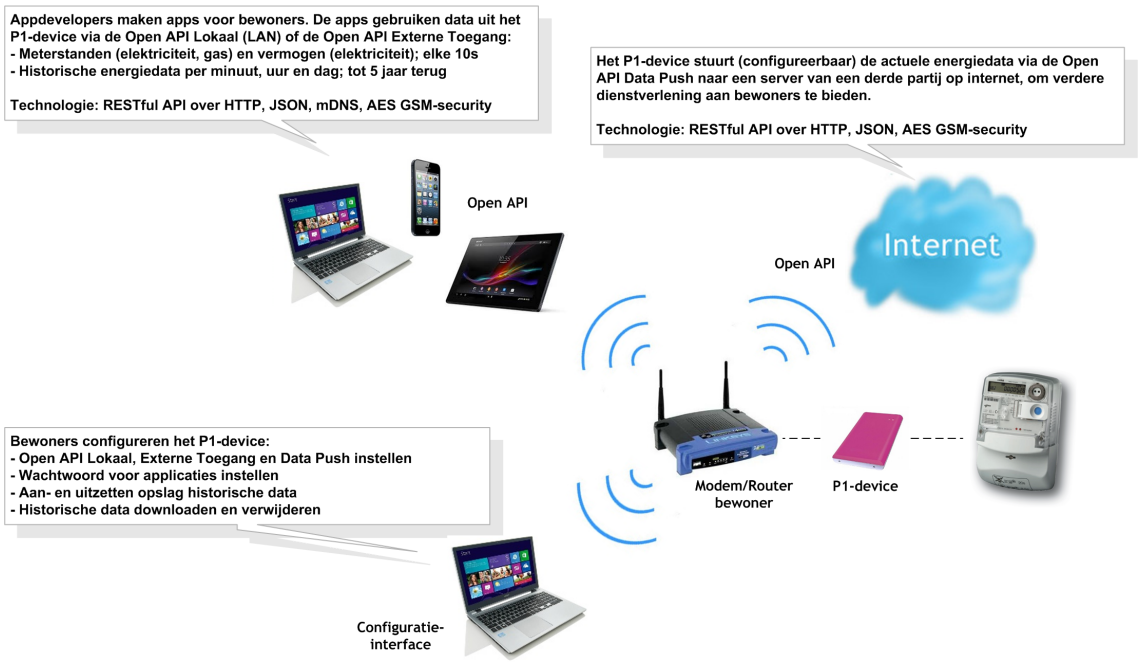
Dit document bevat een beschrijving van de P6 Companion Standard: in essentie zijnde specificaties van een beveiligd communicatieprotocol genoemd de “Open Application Programming Interface” of “Open API” als extensie van de P1 Companion Standard, dat onderdeel is van de Dutch Smart Meter Requirements van Netbeheer Nederland.

De P6 Companion Standard beoogt met de Open API een eenvoudig HTTP-gebaseerd communicatieprotocol voor de P1-poort van Nederlandse Slimme Meters tot stand te brengen, waardoor P1-devices die de P6 ondersteunen en toepassingen (applicaties/apps) die de Open API gebruiken, interoperabel worden. Met P1-device bedoelen we: een elektronisch apparaatje dat bewoners aan de P1-poort aansluiten, dat de data uit de P1-poort leest en dat deze data direct of indirect nuttig maakt voor de bewoner.

De specificaties van de Open API zijn ontstaan na verschillende iteraties tijdens ontwikkeling van het P1-device, het P1-device van Enexis. Op een bepaald moment is gemeend om de specificaties van de Open API los te trekken van de specificaties van het P1-device, zodat deze toegepast kunnen worden door andere partijen dan Enexis. Hieruit is de P6 Companion Standard ontstaan.

De P6 Companion Standard is meer dan alleen een document: naast dat het P1-device gezien kan worden als referentie-implementatie van de P6 Companion Standard in een P1-device, zijn er referentie-applicaties voor iOS, Android en Windows en is er een Software Development Kit (SDK) met simulatiesoftware en broncode.

De werking van een P1-device met een implementatie van de P6 ziet er schematisch als volgt uit, zie afbeelding 1:



Afbeelding 1: schematische weergave werking P1-device met P6-functies

Dit document is als volgt opgebouwd: hoofdstuk 2 bevat de specificaties van de hardware van het P1-device, hoofdstuk 3 geeft een lijst van de specificaties van de configuratie-interface die bewoners kunnen gebruiken om het P1-device te beheren, hoofdstuk 4 specificeert de Open API, vervolgens staat in hoofdstuk 5 een korte beschrijving van de Software Development Kit die ontwikkelaars helpt bij het maken van applicaties en bevat hoofdstuk 6

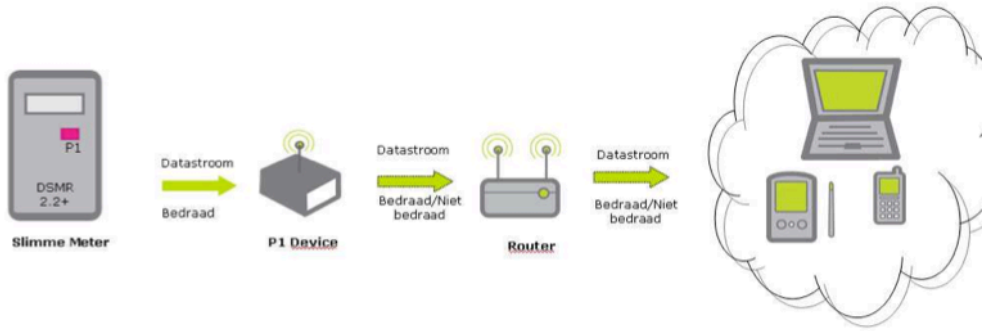
een overzicht van de Simulator en referentieapplicaties voor iPhone, Android en Windows die ter ondersteuning onderdeel zijn van de P6 Companion Standard.

Hierna volgen enkele bijlagen: bijlage A bevat een schematisch overzicht van de functionaliteiten van het P1-device, bijlage B geeft in detail de encryptie- en authenticatiebewerkingen van de gehanteerd AES GCM-beveiliging in de Open API, in bijlage C en D staan procesdiagrammen die weergeven hoe de referentieapplicaties verbinding maken met het P1-device, bijlage E bevat de gehanteerde standaarden en best-practices in de P6 Companion Standard, bijlage F somt de gehanteerde definities op, bijlage G bevat enkele bronnen en referenties en tenslotte bevat bijlage H een tabel met een gedetailleerde versiehistorie van dit document.

Ter info: is het mogelijk om tegelijk elektriciteit te leveren aan en gebruiken uit het net?
Als je zonnepanelen op je dak hebt, dan lever je elektriciteit aan het net als je netto meer produceert dan je verbruikt (in je huis). Dat betekent dus dat (bij hoog tarief) T1 0 is (verbruik) en T3 positief is (levering). Produceer je minder dan je gebruikt, dan is T1 positief en T3 0. Echter, in sommige huizen kan het voorkomen dat zowel T1 positief is en T3 ook. Er wordt dus zowel verbruikt als geleverd. Hoe kan dat? Dit doet zich voor in een situatie met “3-fases”, onder andere het geval als men een inductiekookplaat heeft. Er zijn dan logisch gezien 2 elektriciteitsnetten in huis aanwezig en op 1 daarvan zijn de zonnepanelen aangesloten. Als T1 en T3 beide positief zijn, dan is het verschil van deze de netto productie of consumptie. Ja dus.

2 Specificaties hardware

Dit hoofdstuk bevat een beknopt overzicht van de specificaties van de hardware van een P1-device dat de P6 Companion Standard kan ondersteunen.



Afbeelding 2: schematische weergave hardware

Slimme meter:

- De slimme meter voorziet de P1-poort met het P1-telegram. De P1-poort is in DSMR 2.2+ een niet gevoede seriële data poort (9K6 Baud) met een zogenaamde RJ11-connector. In DSMR 4.0 is deze seriële data poort (115K2 Baud) poort wel gevoed.

P1-device:

- Het P1-device is een kastje dat de data van de P1 poort (= RJ11 aansluiting) vertaalt naar een werkomgeving waar een consument gebruik van kan maken om de data inzichtelijk te krijgen in het belang van bewustwording betreffende energieverbruik in relatie tot verbruiksgedrag.
- In principe kan vanuit het P1-device een verwerkingseenheid aangesproken worden die de data direct kan verwerken of via apparaten (bv router) die de data kan doorzetten naar meerdere verwerkingseenheden.
- Uiteindelijk voegt de uitvoering van additionele software de uiteindelijke kracht toe om zelfs naar bijv. een mobiele telefoon toe te communiceren en of analyses te maken.
- De klant bepaalt uiteindelijk zelf of de meterstanden lokaal zichtbaar komen en omgezet worden in verbruiksgegevens en al dan niet buiten het thuisnetwerk beschikbaar komen. De software zal daar dus rekening mee moeten houden.

Router:

- De router is apparatuur van de klant. De router is aangesloten op het P1-device. Hiermee komt de data van de slimme meter beschikbaar in het zogenaamde lokale netwerk (LAN) en voor zover gewenst ook op internet.

Verwerkingseenheid, bijvoorbeeld een laptop:

- Een verwerkingseenheid kan een telefoon, computer of een tablet zijn. Een applicatie op de verwerkingseenheid kan dan de data omzetten in informatie voor de gebruiker.

Eisen aan het P1-device

Hardware:

1. Aansluitbaar door middel van een mee te leveren kabel bij de P1-poort van de slimme meter van Enexis (RJ11).
2. Koppelfunctionaliteit met (eventuele) router met DHCP bij voorkeur zeer laagdrempelig (/plug&play) voor de gebruiker.
3. DSMR 4.0 compatible, eventueel downwards compatible met DSMR 2.2+

NB:

- Bij DSMR 2.2+ meters de mogelijkheid het device apart te voeden (bv. middels een batterij compartiment of adapter).
- Voor DSMR 4 bij voorkeur geen externe voeding nodig; in dat geval rekening houden met maximale mogelijk te leveren stroom vanuit de P1-poort.
- Zie voor nadere informatie hiervoor P1 Companion Standaard van Netbeheer Nederland.

Overige kaders:

1. P1-telegram databuffer/bewaartijd van device: minuutwaarden E+G -> maand, uurwaarden E/G -> 1 jaar, dagwaarden E/G -> 5 jaar (alleen de meterstanden hoeven opgeslagen te worden, niet de volledige P1-telegrammen).
2. De actuele/momentane datastroom (10-secondenwaarden E en uurwaarden G) doorgeven.
3. Het P1-device moet uitbreidingen in de P1-berichten begrijpen (niet op crashen), ook in DSMR2.2+, zowel in de hardware als software van het device.

3 Specificaties configuratie-interface

Dit hoofdstuk bevat de specificaties voor hoe bewoners het P1-device kunnen beheren middels een configuratie-interface.

De configuratie-interface moet in elk geval de volgende functies bevatten:
1. Aan- en uitzetten van de Open API Lokaal, Externe Toegang en Data Push. Dit zijn de drie manieren van toegang tot de energiedata voor applicaties.
2. Instellen van het wachtwoord voor applicaties.
3. Configureren van maximaal 5 servers voor de Open API Data Push-functie.
4. Aan- en uitzetten van de opslag van historische energiedata.
5. Downloaden van de historische energiedata in CSV-formaat.
6. Verwijderen van de historische data.
7. Inzien van de Open API-versie die het P1-device ondersteunt.

Daarnaast wordt aanbevolen om de volgende functies op te nemen:
8. Wachtwoord om toegang tot de configuratie-interface te krijgen, met mogelijkheid om dit wachtwoord te veranderen en terug te zetten naar 'default' (fabrieksinstelling).
9. Algemene voorwaarden en disclaimer die bewoners eenmalig en bij wijzigingen moeten accepteren alvorens gebruik te kunnen maken van de configuratie-interface en daarmee het P1-device.
10. Indicaties van de correcte aansluiting en werking van het P1-device.
11. Inzien en/of downloaden van P1-berichten.
12. Inzien van systeeminformatie om foutopsporing en hulp op afstand te vereenvoudigen.

Verder moet de configuratie-interface voldoende beveiligd zijn om te voorkomen dat hackers eenvoudig toegang krijgen. Dit kan bijvoorbeeld door dezelfde AES GCM-beveiliging te gebruiken als de Open API.

De configuratie-interface heeft de volgende minimale performance en laadtijden:
- Bij (optisch) wisselen van een scherm duurt het laden van het (optisch) nieuwe scherm maximaal 1,5 seconden. Met 'optisch' wordt bedoeld: ook al wordt er technisch geen nieuwe scherm geladen, voor de bewoner lijkt dit wel zo.
- Het uitvoeren van een functie zoals wijzigen van het wachtwoord duurt maximaal 3 seconden.
- Het downloaden van alle energiedata per minuut voor een maand duurt maximaal 10 seconden.

Ad 5. Downloaden van de historische energiedata in CSV-formaat
De bewoner kan alle aanwezige data per dag, uur, minuut en 10 seconden downloaden in CSV-formaat.

Output per dag:
- Datum en tijd (dd-mm-jjjj uu:mm:ss)
- Meterstand begin van de dag elektriciteit telwerk I (00:00, kWh in drie decimalen)
- Meterstand begin van de dag elektriciteit telwerk II (00:00, kWh in drie decimalen)
- Meterstand begin van de dag teruggeleverde elektriciteit telwerk I (00:00, kWh in drie decimalen)
- Meterstand begin van de dag teruggeleverde elektriciteit telwerk II (00:00, kWh in drie decimalen)
- Meterstand begin van de dag gas (00:00, m^3 in drie decimalen)

```
Datum;Meterstand Tarief I (Laag) (kWh);Meterstand Tarief II (Hoog) (kWh);Teruglevering
Tarief I (kWh);Teruglevering Tarief II (kWh);Meterstand Gas(m3)
21-01-2014 00:00:00;441,055;453,345;0,000;0,000;483,966;
22-01-2014 00:00:00;441,446;455,985;0,000;0,000;486,404;
23-01-2014 00:00:00;443,112;462,626;0,000;0,000;494,791;
...
```

Output per uur:

- Datum en tijd (dd-mm-jjjj uu:mm:ss)
- Meterstand begin van het uur elektriciteit telwerk I (kWh in drie decimalen)
- Meterstand begin van het uur elektriciteit telwerk II (kWh in drie decimalen)
- Meterstand begin van het uur teruggeleverde elektriciteit telwerk I (kWh in drie decimalen)
- Meterstand begin van het uur teruggeleverde elektriciteit telwerk II (kWh in drie decimalen)
- Meterstand begin van het uur gas (m^3 in drie decimalen)

```
Datum;Meterstand Tarief I (Laag) (kWh);Meterstand Tarief II (Hoog) (kWh);Teruglevering
Tarief I (kWh);Teruglevering Tarief II (kWh);Meterstand Gas(m3)
21-01-2014 17:00:00;441,055;453,345;0,000;0,000;483,966;
21-01-2014 18:00:00;441,055;453,396;0,000;0,000;483,966;
21-01-2014 19:00:00;441,055;453,903;0,000;0,000;484,399;
...
```

Output per minuut:

- Identiek aan per uur, met meterstanden aan het begin van de minuut

```
Datum;Meterstand Tarief I (Laag) (kWh);Meterstand Tarief II (Hoog)
(kWh);Teruglevering Tarief I (kWh);Teruglevering Tarief II (kWh);Meterstand Gas(m3)
25-02-2014 10:21:00;538,158;574,701;0,000;0,000;694,467;
25-02-2014 10:22:00;538,158;574,705;0,000;0,000;694,467;
25-02-2014 10:23:00;538,158;574,710;0,000;0,000;694,467;
...
```

Output per 10 seconden:

- Bevat ook het elektrisch vermogen en teruggeleverde vermogen

```
Datum;Meterstand Tarief I (Laag) (kWh);Meterstand Tarief II (Hoog) (kWh);Teruglevering
Tarief I (kWh);Teruglevering Tarief II (kWh);Meterstand Gas(m3);Vermogen (W);Teruglever
vermogen (W)
18-03-2014 12:51:50;600,782;660,302;0,000;0,000;801,396;410;0;
18-03-2014 12:52:00;600,782;660,303;0,000;0,000;801,396;350;0;
18-03-2014 12:52:10;600,782;660,304;0,000;0,000;801,396;540;0;
...
```

Als er een meterwaarde ontbreekt in de reeks, dan bevat de CSV enkel een spatie op de plek van de waarde.

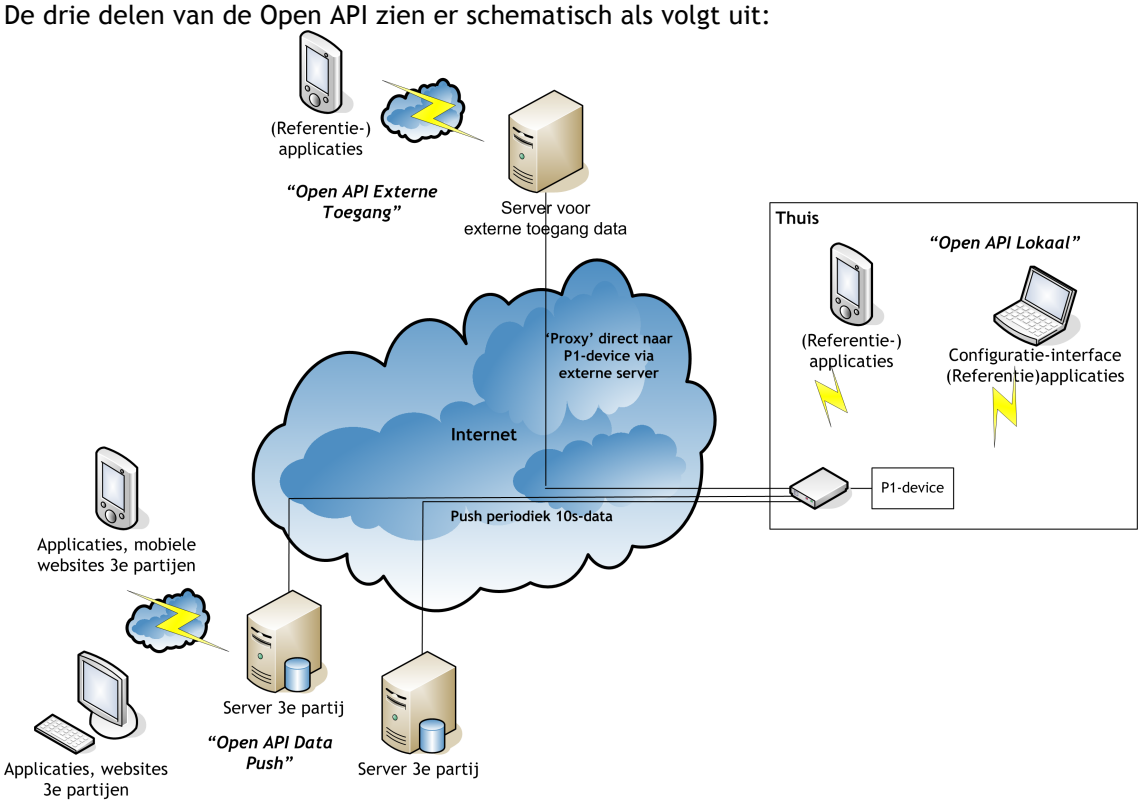
Het gekozen CSV-formaat lijkt zoveel mogelijk op het CSV-formaat van The Energy Stick (TES, zie http://energystick.nl/). Het grootste verschil is dat TES de verbruikswaarden geeft en P1-device de meterstanden.

4 Specificaties Open API

Dit hoofdstuk bevat de specificaties van de zogenaamde Open Application Programming Interface (Open API). We hebben het over een 'open' API omdat de API vrij beschikbaar zal zijn voor iedere applicatieontwikkelaar (appdeveloper) om in te zien en bij toestemming van bewoners te gebruiken. De Open API is opgesteld naar bestaande standaarden en best-practices, zodat applicatieontwikkelaars snel in staat zijn om applicaties (apps) te maken.

De Open API bestaat uit drie delen:
1. Open API Lokaal: het direct benaderen van het P1-device in het thuisnetwerk (LAN).
2. Open API Externe Toegang: toegang via een 'proxy' (externe server) tot de Open API Lokaal. Deze proxy is in essentie transparant voor de applicaties en zorgt ervoor dat applicaties ook buitenshuis toegang hebben tot het P1-device.
3. Open API Data Push: het pushen van 10-secondedata naar maximaal 5 servers van een derde partij.

De drie delen van de Open API zien er schematisch als volgt uit:



Afbeelding 3: schematische weergave Open API

De paragrafen 4.1, 4.2 en 4.3 bevatten de specificaties van elk deel van de Open API. Elke paragraaf begint met een functionele beschrijving vanuit het perspectief van de gebruiker, gevolgd door een of meer paragrafen met technische specificaties en details.

4.1 Open API Lokaal

Deze paragraaf bevat functionele en technische specificaties van de Open API Lokaal: het gedeelte van de Open API zoals deze werkt op het thuisnetwerk van de bewoner.

Paragraaf 4.1.1 bevat een functionele beschrijving van de Open API Lokaal vanuit het perspectief van de gebruiker. De overige paragrafen bevatten technische specificaties en details.

4.1.1 Functioneel: vanuit perspectief bewoner

Deze paragraaf bevat een functionele beschrijving van hoe een bewoner de Open API via apps gebruikt.

Paragraaf 4.1.1.1 geeft aan hoe een bewoner het P1-device instelt om apps toegang te geven tot zijn energiedata. Paragraaf 4.1.1.2 bevat het proces van downloaden en gebruiken van apps. En paragraaf 4.1.1.3 beschrijft hoe een bewoner de toegang van apps tot energiedata kan beperken.

4.1.1.1 Openstellen energiedata voor applicaties

De bewoner gebruikt de configuratie-interface om apps toegang te geven tot zijn energiedata. Dit door het instellen van een wachtwoord voor applicaties dat de bewoner vrij kan kiezen.

4.1.1.2 Downloaden en gebruiken van apps

Om apps te gaan gebruiken, zullen bewoners eerst te weet moeten komen dat er apps bestaan en welke dat zijn. Dit kunnen ze vernemen via communicatie van de P1-deviceverkoper, via vrienden, familie of buren, via de appstore (Apple Store of Google Play), via zoekmachines (Google) en via uitingen in de media.

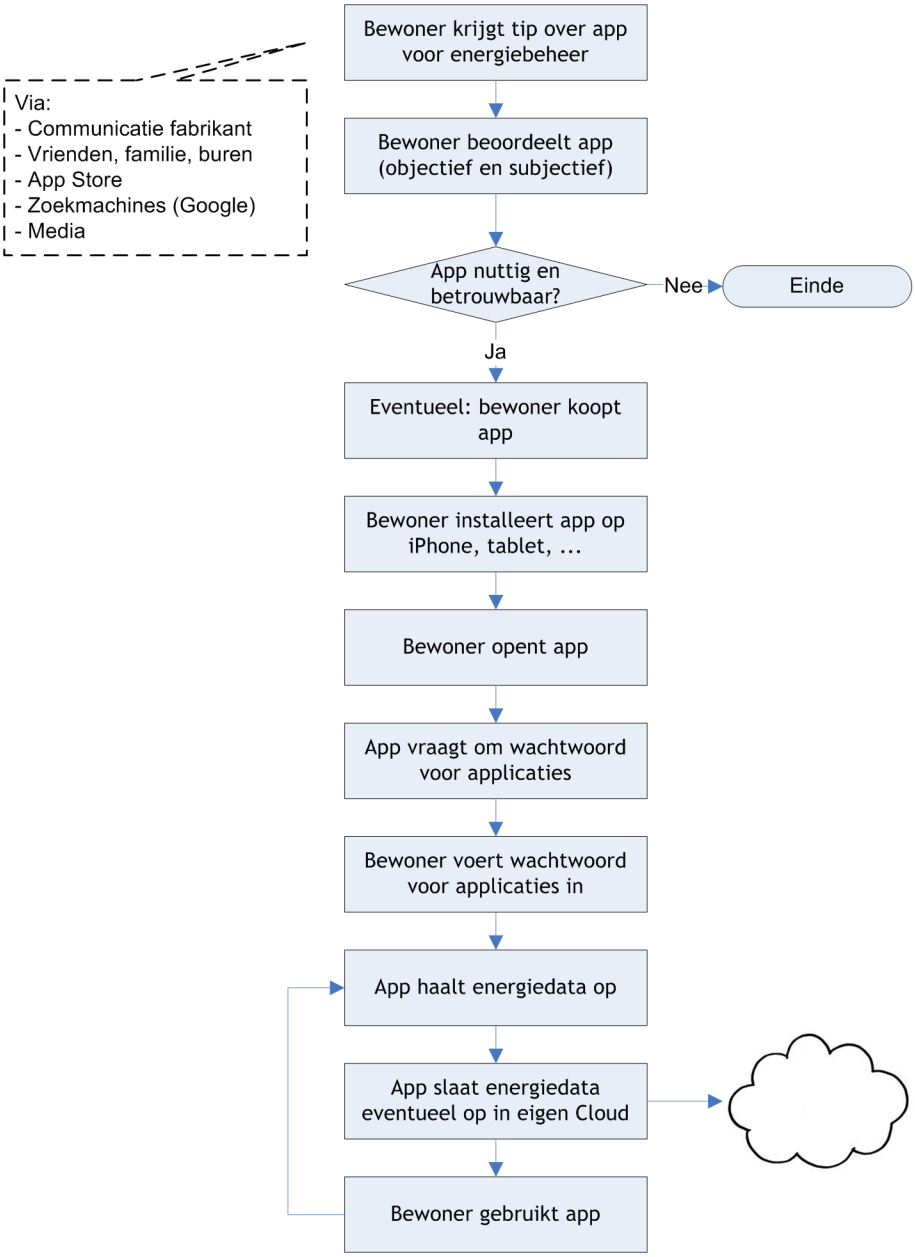
Als een bewoner een app heeft gevonden, dan schat hij in of hij de app(developer) vertrouwt en of de app interessant genoeg is om te gebruiken (kopen of gratis downloaden). Nadat de bewoner de app op zijn device (computer, laptop, tablet of smartphone) heeft geïnstalleerd, start hij deze op voor eerste gebruik.

De app zal bij eerste gebruik automatisch het P1-device vinden, mits het device van de bewoner is verbonden met het thuisnetwerk. De app zal nu (eenmalig) vragen om het “wachtwoord voor applicaties” dat de bewoner in de configuratie-interface heeft ingesteld. De app kan vervolgens naar believen energiedata van het P1-device ophalen.

De app zal zolang deze geïnstalleerd is op het device van de bewoner energiedata op kunnen halen. Sommige apps zullen dat alleen doen als de app actief is (geopend is door de bewoner), andere halen de data mogelijk ‘op de achtergrond’ periodiek op.

Het is mogelijk dat apps de energiedata via de internetverbinding van de bewoner opslaan in een ‘cloud’ (eigen database verbonden aan internet) om interessante toepassingen te kunnen bieden aan de bewoner. Hierbij wordt de app dus gebruikt als ‘gateway’ naar internet. Externe servers op internet kunnen ook direct energiedata van het P1-device ontvangen, zie daarvoor paragraaf 4.3.

Zie afbeelding 4 voor een schematische weergave van het proces hoe bewoners applicaties vinden, downloaden en in gebruik nemen:



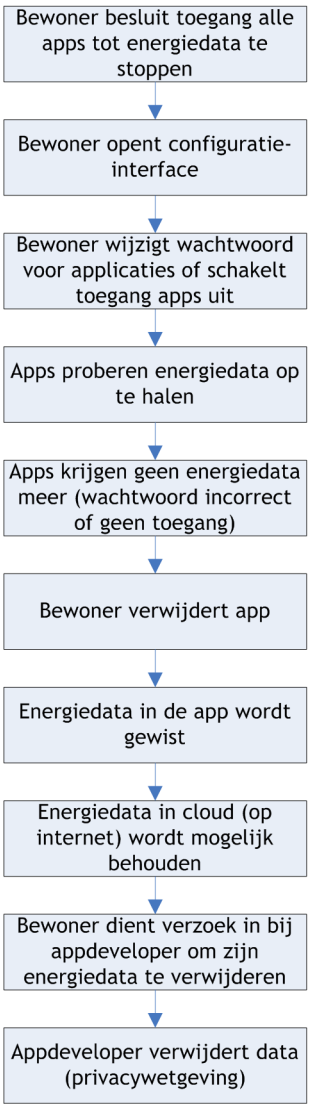
Afbeelding 4: proces van apps downloaden en gebruiken

4.1.1.3 Toegang tot energiedata beperken

De bewoner kan op elk moment besluiten dat hij apps geen verdere toegang wil geven tot zijn energiedata. Daarvoor wijzigt hij het wachtwoord voor applicaties of zet hij de toegang voor applicaties uit in de configuratie-interface.

Data die al opgeslagen was in de app (historische data) blijft dan wel behouden. Als de gebruiker de app van zijn device wist, dan zal de historische data in de app ook gewist worden. Tenzij de app de data in de cloud heeft opgeslagen. De bewoner kan een verzoek indienen bij de appdeveloper om alle persoonlijke data te verwijderen (privacywetgeving).

Zie afbeelding 5 voor een schematische weergave van het proces waarmee bewoners de toegang tot hun energiedata kunnen beperken:



Afbeelding 5: proces van toegang tot energiedata beperken

De volgende subparagrafen van paragraaf 4.1 bevatten technische specificaties en details.

4.1.2 P1-device vinden met Device Discovery of externe server

Deze paragraaf beschrijft hoe de Open API-interface gebruik maakt van ‘device discovery’-protocollen om het P1-device te kunnen vinden in het thuisnetwerk (LAN).

Als een softwareapplicatie (app) voor het eerst energiedata wil ophalen van het P1-device, dan zal de app het device eerst moeten vinden. Hiervoor hanteert het P1-device de volgende protocollen met hostname “P1-device.local”:

- Multicast DNS (mDNS): alleen name resolution, geen Service Discovery; het device reageert alleen op A-requests (niet op ANY)
- NetBIOS
- LLMNR
- DHCP Hostname

Uit praktijktests is gebleken dat bij ongeveer 10% van de bewoners een applicatie het P1-device niet kan vinden met genoemde protocollen. In veel van deze gevallen komt dit omdat

de netwerkconfiguratie van de bewoner de UDP-pakketten niet goed doorlaat tussen het P1-device en de applicatie.

Daarom kan een applicatie een request doen naar een externe detectieserver om het lokale IP-adres van het P1-device te achterhalen. Dit gaat als volgt:

- App stuurt een HTTP GET-request naar http://detect.[p1device].nl/localip
- De detectieserver controleert of het WAN IP-adres bekend is in de database.
- Zo ja, dan stuurt de server een JSON-response met het serienummer en IP-adres:

```
{
   "ipAddress": "192.168.1.100",
   "serialNumber": "12345678"
}
```

- Zo nee, dan is de JSON-response leeg:

```
{
   "ipAddress": null,
   "serialNumber": null
}
```

- In dat geval kan de gebruiker een serienummer (bijvoorbeeld 12345678) invoeren en kan een app het lokale IP-adres opvragen met: http://detect.[p1device].nl/localip/12345678
- Het is ook mogelijk om http://detect.[p1device].nl/redirect aan te roepen: dan wordt de applicatie doorverwezen naar het IP-adres van het P1-device, wat in de browser betekent dat (bijvoorbeeld) een bewoner de configuratie-interface kan openen.

Theoretisch kan het voorkomen dat er meerdere P1-devices in het netwerk van de bewoner zijn, dus dat de externe database meer dan 1 P1-device op een WAN IP-adres registreert. We gaan er vanuit dat dit in de praktijk niet voorkomt.

Zie de procesdiagrammen in bijlagen C en D hoe een applicatie te werk gaat om het P1-device te vinden.

Als het device geen DHCP-server kan benaderen, dan kiest het device een IP-adres in de 169.254.x.y-range, te beginnen bij 169.254.3.3. Dit volgens de Automatic Private IP Addressing-methode van de Internet Assigned Numbers Authority (IANA).

Motivatie voor gemaakte keuzes:

- Hoewel mDNS geen officiële onafhankelijke open standaard is, is het op dit moment de meest gangbare open standaard die voorhanden is.
- Door naast mDNS ook LLMNR en NetBIOS name aan te bieden, hoeven Windowsgebruikers geen aanvullende software op hun computer te installeren (Apple iTunes of Apple Bonjour).
- Door een aantal van de meest gangbare protocollen te ondersteunen, is geprobeerd in zoveel mogelijk gevallen het device snel en eenvoudig te laten vinden in het thuisnetwerk.

4.1.3 Web API voor het ophalen van data

Het P1-device beschikt over een webserver die kan reageren op HTTP-requests van een app over TCP/IP. De web server luistert op poort 80. Een app stuurt een HTTP-GET request om data op te halen uit het P1-device. Dit resulteert in een Web API die appdevelopers gebruiken om apps te ontwikkelen voor het P1-device.

Het P1-device zet in de Server-header van elke HTTP-response “P1-device/0.1” (compleet dus de header: “Server: P1-device/0.1”). Dit om de webserver in het device te identificeren.

Motivatie voor gemaakte keuzes:
- Een Web API is momenteel een gangbare en moderne manier voor communicatie op applicatieniveau om appdevelopers de mogelijkheid te geven apps te ontwikkelen voor thuiscomputers, laptops, tablets en smartphones, etc.
- Om een Web API te bieden, is een webserver op het P1-device nodig. Poort 80 is de meest gangbare en standaardpoort voor een webserver om over te communiceren voor normaal dataverkeer.

4.1.3.1 Requests die apps kunnen sturen om data op te halen

Het P1-device slaat energiedata (meterstanden) op van de energietypen Elektriciteit I, Elektriciteit I retour, Elektriciteit II, Elektriciteit II retour en Gas. En van elk type:
- Per minuut, 1 maand opslagcapaciteit (44.640 entries: 31 dagen * 24 uur * 60 minuten)
- Per uur, 1 jaar opslagcapaciteit (8.784 entries: 366 dagen * 24 uur)
- Per dag, 5 jaar opslagcapaciteit (1.827 entries: 3 * 365 dagen + 2 * 366 dagen)
- Actueel inclusief vermogen elektriciteit, elke 10 seconden (1 entry)
- Per 10 seconden, inclusief vermogen elektriciteit, 15 minuten opslagcapaciteit (90 entries: 15 minuten * 6 10-seconden)

Met een URL en parameters specificeert de app in het GET-request welke energiedata te krijgen:

URL, http://[p1device].local	**Queryparameters**	**Toelichting**
/accessinfo	(geen)	Geeft configuratie-informatie over toegang tot het P1-device; is aan te roepen zonder beveiligde AES GCM-sessie
/energynow	(geen)	Geeft actuele standen en vermogen, zoals in laatste P1-bericht (ververst elke 10 seconden)
/energyhistory	?interval=(tenseconds,minute,hour,day) &timestamp=(UNIX-timestamp) &entries=(aantal) &format=(json,csv)	Geeft historische energiedata
/status	(geen)	Geeft statusinformatie over het P1-device
/p1	(geen)	Geeft het volledige P1-bericht (ververst elke 10 seconden)
/login	(geen)	Authentication endpoint voor authenticatie en opbouwen AES GCM-sessie, zie paragraaf 4.4
/logout		Ruimt de bestaande beveiligde AES GCM-sessie op

Het P1-device geeft de energiedata van alle energietypen (Elektriciteit I, II, retour I, II en Gas) altijd bij elkaar, dit is niet te specificeren met een parameter.

Het ophalen van historische energiedata gaat dan bijvoorbeeld met de URL:
http://[p1device].local/energyhistory?interval=minute&timestamp=1364988259&entries=10&format=json

Toelichting op de queryparameters bij /energyhistory:

- Interval: geeft aan of de energiedata per 10 seconden, minuut, uur of dag opgehaald wordt.
  - Default (bij weglaten) is: tenseconds.
- Timestamp: een UNIX-timestamp die aangeeft vanaf welk moment data opgehaald wordt (het aantal seconden sinds 1 januari 1970 middernacht UTC, Coordinated Universal Time).
  - Let op: hoewel het communicatieformaat UTC is, slaat het device de data wel volgens de Nederlandse tijdzone op (of de tijdzone waar het device zich bevindt). Dus het begin van een dag is 0:00 Nederlandse tijd, vertaalt naar UTC.
  - Truncating: als de timestamp ergens gedurende (halverwege) een 10-secondeinterval, minuut, uur of dag is, dan neemt het P1-device de hele 10 seconden, minuut, uur of dag voorafgaande (de rest wordt 'getruncated'), bijvoorbeeld: dagwaarde maandag 13:20 = dagwaarde maandag 0:00.
  - Default (bij weglaten):
    - Als de entries-parameter aanwezig is: startmoment rekent het P1-device zelf uit (als entries=60 en detail=minute, dan is timestamp dus 1 uur geleden)
    - Als de entries-parameter ontbreekt: alle data voor zover aanwezig in het P1-device
  - Timestamp te ver in het verleden (er is geen of deels data beschikbaar vanaf dat moment):
    - Als er GEEN data is, dan null
    - Als er DEELS data is, dus vanaf 4 juni als vanaf 1 juni wordt opgevraagd met 30 entries, dan geeft ie timestampFirst 4 juni, timestampLast 30 juni, met 30-4 = 26 entries
  - Timestamp in de toekomst:
    - Als de entries-parameter aanwezig is: zelfde werking als bij default
    - Als de entries-parameter ontbreekt: de entry met de meest recente waarden
- Entries: geeft aan van hoeveel momenten (10-seconden, minuten, uren, dagen) de historie opgehaald wordt.
  - Default (bij weglaten):
    - Als de timestamp aanwezig is: alle entries tot nu
    - Als de timestamp ontbreekt: alle aanwezige data
- Format: geeft aan of de data in JSON of CSV-formaat wordt gegeven.
  - Default (bij weglaten): JSON

Motivatie voor gemaakte keuzes:

- Met de API haalt een app de data voor alle soorten energie in 1 keer op. Dit is niet verder gesplitst (niet specificeerbaar met een parameter) om de benodigde processor- en opslagcapaciteit op het P1-device te beperken.
- De API bevat verschillende parameters om een beperkt aantal entries op te halen uit het P1-device. Dit om de grote hoeveelheid mogelijk onnodige data die keer op keer over het netwerk gaat en in de app verwerkt moet worden, te beperken. Met deze parameters is een balans gevonden tussen de nodige processor- en opslagcapaciteit op het P1-device en onnodig veel data sturen.
- Het P1-device bevat een klok om apps ook in geval van een of meer stroomstoringen van accurate data te voorzien ("ik trek even de stekker eruit" is ook een stroomstoring). Hoewel appdevelopers mechanismen toe kunnen voegen om niet-aanwezige of niet-accurate data eruit te filteren, moet wel bekend zijn welke data al dan niet aanwezig of accuraat is. De klok synchroniseert met de timestamp in de P1-berichten bij DSMR4+ of via een NTP-server bij DSMR2.2+.

4.1.4 Specificatie van de JSON-berichten

Het /accessinfo-endpoint geeft het volgende JSON-bericht:

```
{
   “serialNumber”: “12345678”,
   “versionAPI”: “1.0”,
   “applicationAccess”: true,
   “externalAccess”: true
}
```

Dit endpoint kan een applicatie gebruiken zonder eerst een beveiligde sessie op te hoeven bouwen. Dit zodat een applicatie kan zorgen voor een goede gebruikerservaring, het geven van juiste opties en feedback (zie ook de procesdiagrammen in bijlagen C en D).

Het P1-device kan energiedata geven in JSON en CSV-formaat. Zie hoofdstuk 3 voor het CSV-formaat.

Het /energyhistory-endpoint geeft het volgende JSON-bericht, wanneer een applicatie 10-secondehistorie opvraagt:

```
{
   “request” : {
      "interval": "tenseconds",
      "timestamp": 1364988259,
      "entries": 5,
      "format": "json"
      },
   “response” : {
      “interval”: “tenseconds”,
      "timestampFirstEntry": 1364988200,
      "timestampLastEntry": 1364988300,
      “entries”: 5,
      "unitElectricity": "kWh",
      "unitElectricityPower" : "W",
      "unitGas": "m3",
      "highTariffElectricity": "electricityMeterI",
      "electricityMeterI": [12.345, 12.345, 12.345, 12.345, 12.345],
      "electricityMeterIReturn": [0.223, 0.233, 0.232, 0.232, 0],
      "electricityMeterII": [12.345, 12.345, 12.345, 12.345, 12.345],
      "electricityMeterIIReturn": [0.045, 0.045, 0.045, 0.045, 0.045],
      "electricityPower": [0.34, 0.34, 0.34, 0.34, 0.34],
      "electricityPowerReturn": [0.12, 0.12, 0.12, 0.12, 0.12],
      "gasMeter": [2320.122, 2320.122, 2319.871, 0, 0]
   }
}
```

Toeliching:

- Het requestobject bevat de waarden zoals deze aanwezig waren het HTTP-request om de data op te vragen. Als een parameter in het request niet aanwezig is, dan staat er de waarde null, bijvoorbeeld:

```
“timestamp”:  null,
```

- De eenheid van de meterwaarden Elektriciteit (unitElectricity) is altijd kWh
- De eenheid van de meterwaarde Gas (unitGas) is altijd m3
- Welk telwerk het hoge tarief elektriciteit heeft (highTariffElectricity): electricityMeterI of electricityMeterII. In de praktijk is meter I altijd laagtarief, maar door een fout in een van de DSMR-specificaties, bevat het JSON-bericht deze informatie expliciet.
- De timestamp van de eerste meterstand

- De timestamp van de laatste meterstand
- Per soort meterstanden een lijst, in 3 decimalen nauwkeurig:
    - Elektriciteit I
    - Elektriciteit I retour
    - Elektriciteit II
    - Elektriciteit II retour
    - Gas
- De array met meterstanden is van oud naar nieuw (oudste meterstand links, nieuwste meterstand rechts).
- Wanneer een app de energiedata in gedeeltes ophaalt, dan kan deze de waarde “timestampLastEntry” gebruiken voor een nieuw request.
- Als er geen meterstanden aanwezig zijn, bijvoorbeeld door een (tijdelijke) storing, dan bevat het JSON-bericht voor elke waarde die ontbreekt ‘null’ (zonder de apostroffen). Bijvoorbeeld:
    - `“electricityMeterII”: [12345, null, 12345, null, 12345],`
- Als alle waardes in een array ontbreken (bijvoorbeeld omdat er geen gasmeter is), dan kan null weergegeven worden in plaats van de array, zonder de haken. Bijvoorbeeld:
    - `“gasMeter”: null`
- Als alle waardes in alle arrays ontbreken (er is helemaal geen historiedata), dan zijn timestampFirstEntry en timestampLastEntry beide null. Als er slechts één waarde is, dan zijn de timestampFirstEntry en timestampLastEntry gelijk aan elkaar.

Het JSON-bericht van het /energyhistory-endpoint voor de data per minuut, uur en dag ziet er hetzelfde uit als voor de 10-secondedata, behalve dat er geen waarden voor vermogen van elektriciteit aanwezig zijn:


```
{
   “request” : {
      "interval": "minute",
      "timestamp": 1364988259,
      "entries": 5,
      "format": "json"
      },
   “response” : {
      “interval”: “minute”,
      "timestampFirstEntry": 1364988200,
      "timestampLastEntry": 1364988300,
      “entries”: 5,
      "unitElectricity": "kWh",
      "unitGas": "m3",
      "highTariffElectricity": "electricityMeterI",
      "electricityMeterI": [12.345, 12.345, 12.345, 12.345, 12.345],
      "electricityMeterIReturn": [0.223, 0.233, 0.232, 0.232, 0],
      "electricityMeterII": [12.345, 12.345, 12.345, 12.345, 12.345],
      "electricityMeterIIReturn": [0.045, 0.045, 0.045, 0.045, 0.045],
      "gasMeter": [2320.122, 2320.122, 2319.871, 0, 0]
   }
}
```

Het /energynow-endpoint geeft een JSON-bericht voor de actuele data veel lijkt op dat van de 10-secondedata en minuut-, uur- en dagdata en ziet er als volgt uit:

```
{
   “request” : {},
   “response” : {
      "interval": "now",
      "timestampLastEntry": 1352761963,
      "unitElectricity": "kWh",
      "unitElectricityPower" : "W",
      "unitGas": "m3",
      "highTariffElectricity": "electricityMeterI",
      "electricityMeterI": 12.345,
      "electricityMeterIReturn": 0.223,
      "electricityMeterII": 12.345,
      "electricityMeterIIReturn": 0.045,
      "electricityPower": 0.34,
      "electricityPowerReturn": 0.12,
      "gasMeter": 2320.126
   }
}
```

Het /status-endpoint geeft een JSON-bericht voor informatie over de status van het P1-device en ziet er als volgt uit:

```
{
   “history”: true,
   “timestampNow”: 1352761963,
   “timestampFirstTensecondsData”: 1352761963,
   “timestampFirstMinuteData”: 1352761963,
   “timestampFirstHourData”: 1352761963,
   “timestampFirstDayData”: 1352761963
}
```

Dit /status-endpoint is alleen beschikbaar via een beveiligde AES GCM-sessie.

Motivatie voor gemaakte keuzes:

- JSON is momenteel de best practice voor eenvoudige cross-platform data-uitwisseling in Web API’s.
- Het ontworpen JSON-formaat voor energiedata is een gemaakte afweging tussen leesbaarheid (semantiek) van de data, beschikbare processor- en opslagcapaciteit op het P1-device en compactheid van het bericht.

4.2 Open API Externe Toegang

Deze paragraaf bevat de specificaties van de externe toegang, een functie in de Open API die applicaties in staat stelt data uit het P1-device te halen via een speciale proxyserver, wanneer het apparaat van de bewoner niet verbonden is met het thuisnetwerk.

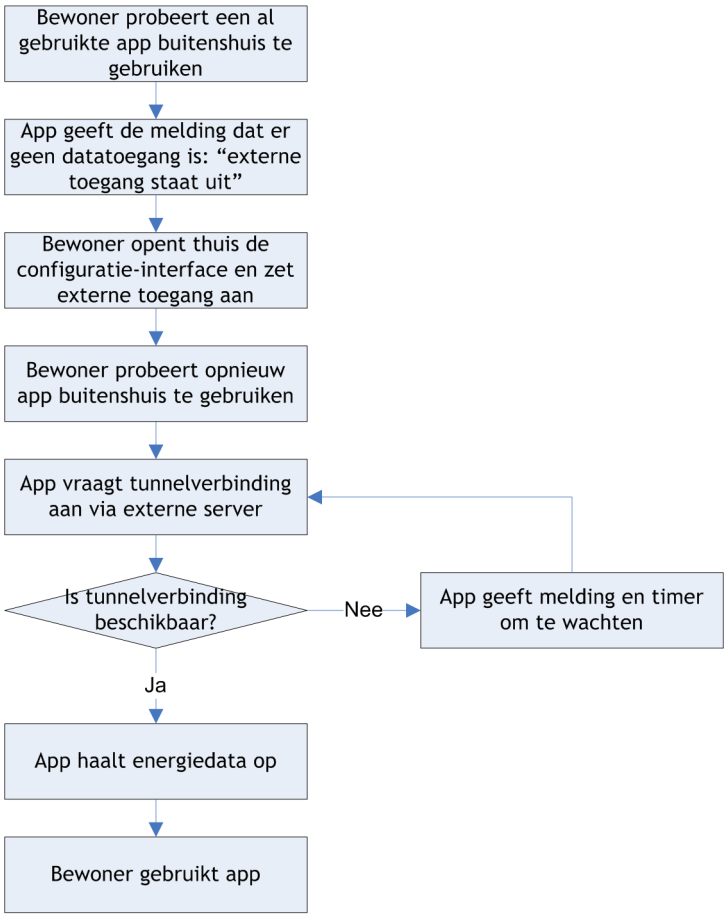
Paragraaf 4.2.1 bevat een functionele beschrijving van de Open API Externe Toegang vanuit het perspectief van de gebruiker. Paragraaf 4.2.2 bevat technische specificaties en details.

4.2.1 Functioneel: vanuit perspectief bewoner

Wanneer een bewoner besluit dat apps ook bij de energiedata in het P1-device moeten kunnen als de bewoner met client device (smartphone, laptop etc.) niet thuis is, dan zet de bewoner de externe toegang voor applicaties aan in de configuratie-interface. Nadat de bewoner ‘externe toegang applicaties’ aan heeft gezet in de configuratie-interface, dan zal de ‘proxy’ voor externe toegang openstaan.

Echter, na een aantal minuten van inactiviteit zal het P1-device periodiek bij de externe server checken of er een proxyverbinding nodig is. Dit om resources op de server te besparen. Hierdoor kan het voorkomen dat een applicatie die het P1-device extern wil benaderen, moet wachten totdat er een proxyverbinding is. De bewoner krijgt hier een melding van in de applicatie.

Zie afbeelding 6 voor een schematische weergave van het proces hoe bewoners applicaties buitenshuis toegang geven via de externe server:



Afbeelding 6: toegang applicaties via externe server

Er zijn meerdere variaties mogelijk op dit proces. Bijvoorbeeld als een bewoner een app buitenshuis voor het eerst gebruikt of als het wachtwoord onjuist blijkt te zijn. Zie bijlagen C en D voor volledige procesdiagrammen.

De volgende subparagraaf van paragraaf 4.2 bevat technische specificaties en details.

4.2.2 Endpoint /accessrequest en statussen voor externe toegang

De externe toegang is in essentie transparant: applicaties kunnen op dezelfde manier als via de Open API Lokaal een beveiligde sessie opbouwen en de energiedata, accessinfo en status ophalen. De externe server fungeert als een ‘doorgeefluik’: requests van de applicatie naar het P1-device en responses van het P1-device naar de applicatie worden 1-op-1 doorgegeven. Omdat de energiedata altijd versleuteld wordt verzonden, is het voor de externe server onmogelijk om energiedata te lezen en/of op te slaan.

Applicaties moeten om gebruik te maken van de externe toegang wel een aanvraag voor verbinding doen via het endpoint ‘/accessrequest’ en moeten er rekening mee houden dat de tunnelverbinding niet altijd actief is tussen het P1-device en de externe server. Met een

aantal statussen kan de applicatie de juiste informatie geven aan de gebruiker tot er wel een verbinding is.

Een applicatie die de externe toegang wil gebruiken, stuurt een HTTP-request naar:
http://data.[p1device].nl/proxy/[serienummer]/accessrequest

Hierop volgt bijvoorbeeld de volgende response:

```
{
   "status": "expected",
   "waitSeconds": 60
}
```

Dit geeft aan dat de app een minuut (60 seconden) moet wachten tot er een proxyverbinding beschikbaar is.

Om dit HTTP-request te doen, moet de applicatie dus het serienummer van het P1-device weten. Dit kan de applicatie eerder uit het /accessinfo-bericht opgeslagen hebben (bij lokaal gebruik) of aan de gebruiker gevraagd hebben het serienummer in te voeren.

Status en waitSeconds kunnen de volgende waardes hebben:
- Status: "connected", waitSeconds: 0
  - Het P1-device heeft een verbinding met de server, een app kan direct requests gaan sturen.
- Status: "expected", waitSeconds: (aantal seconden)
  - Het P1-device heeft zich recent nog gemeld bij de server, maar er is momenteel geen verbinding. De app moet een aantal seconden wachten tot het P1-device zich weer meldt bij de server en er een verbinding opgezet kan worden.
- Status: "disconnected", waitSeconds: (aantal seconden)
  - Het P1-device had zich al moeten melden bij de server, maar dat is niet gebeurd. Wellicht is er een onderbreking in de internetverbinding (geweest), heeft de gebruiker de externe toegang voor applicaties uitgezet in de configuratie-interface, of is het P1-device niet goed meer aangesloten ("stekker eruit"). Omdat het een tijdelijk probleem kan zijn, geeft de server aan de app aan om een aantal seconden te wachten en het dan nogmaals te proberen. Het is aan de app om de gebruiker hier melding van te geven.
- Status: "unknown", waitSeconds: null
  - Dit P1-device heeft zich (nog) niet gemeld bij de server. Mogelijk is externe toegang voor applicaties niet aangezet in de configuratie-interface, is het P1-device niet goed aangesloten en/of heeft deze geen internetverbinding.

Voor dit request is geen AES GCM-sessie nodig (welke logischerwijs ook niet gemaakt kan worden, omdat daarvoor een (proxy)verbinding met het P1-device nodig is).

Motivatie voor gemaakte keuzes:
- De reden voor de Externe Toegang is om bewoners de mogelijkheid te geven om met een app toegang te krijgen tot de data in het P1-device, ook als de bewoner niet thuis is (dus als het client device wel verbonden is met internet, maar niet met het LAN thuis) en zonder gebruik te hoeven maken van een server van een derde partij (zie ook 4.3 Data Push).
- De manier waarop de Externe Toegang-functie werkt, zorgt ervoor dat bewoners met minimale inspanning gebruik kunnen maken van de functie. Ze hoeven bijvoorbeeld geen port-forwarding te configureren in hun modem/router.
- Door end-to-end een beveiligde AES GCM-sessie op te bouwen, heeft de server nooit de beschikking over privacygevoelige data (zoals energiedata) van een bewoner.
- Het initiatief voor het opbouwen van een verbinding ligt bij het P1-device, dus deze functie geeft geen mogelijkheid tot het zomaar verbinding maken naar binnen het lokale netwerk.
- De Externe Toegang-functie kan alleen gebruikt worden door een bewoner die het serienummer en het wachtwoord voor applicaties weet. Bovendien moet 'externe

toegang' aan zijn gezet in de Configuratie-interface.

4.3 Open API Data Push

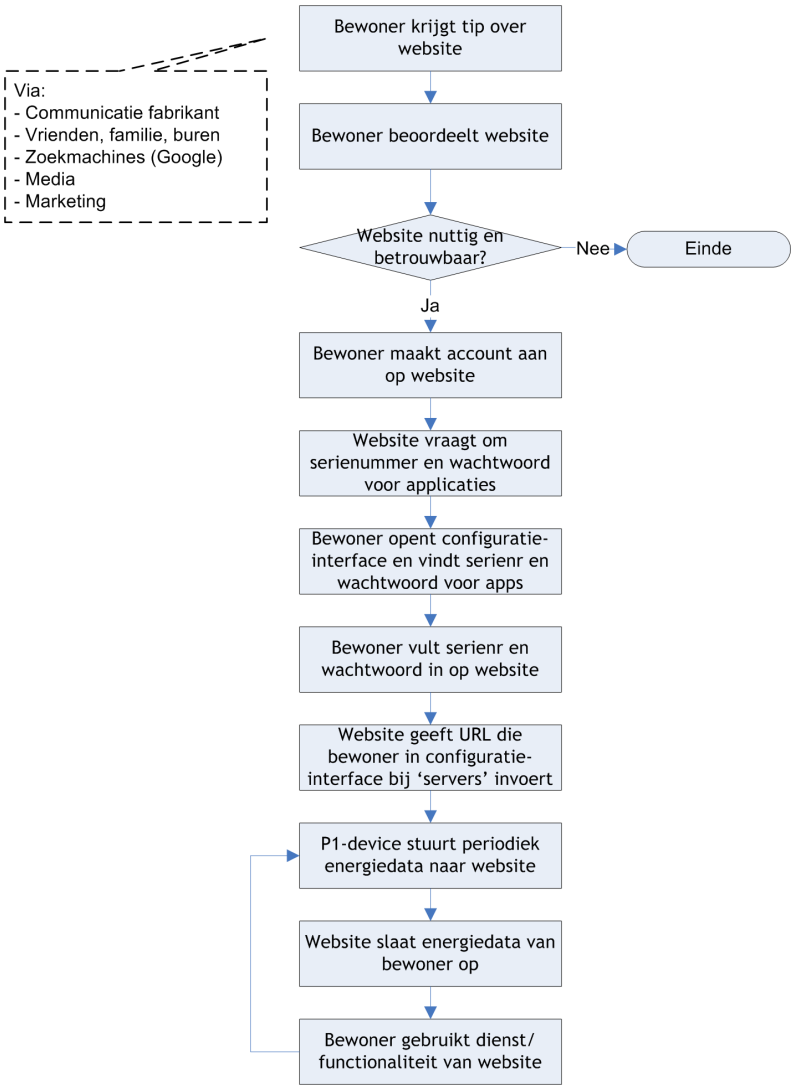
De Open API Data Push-functie geeft applicatie-ontwikkelaars de mogelijkheid om met toestemming van bewoners energiedata continu te ontvangen op een eigen server. Applicatie-ontwikkelaars kunnen de energiedata dan opslaan in een database en er diensten op aanbieden aan de bewoner. Deze paragraaf bevat een functionele en technische beschrijving van de Open API Data Push.

Paragraaf 4.3.1 bevat een functionele beschrijving van de Open API Data Push vanuit het perspectief van de gebruiker. Paragraaf 4.3.2 bevat technische specificaties en details.

4.3.1 Functioneel: vanuit perspectief bewoner

Typisch gezien zal een bewoner onder andere via vrienden, de media of (online) marketing op het spoor komen van een 'interessante website die werkt met het P1-device'. De bewoner bezoekt vervolgens deze website, schat in of hij deze wil gebruiken en als dit het geval is, volgt een aantal stappen om zijn P1-device te verbinden aan de website.

Zie afbeelding 7 voor een schematische weergave van het proces hoe bewoners een website toegang geven tot energiedata:



Afbeelding 7: gebruik websites met energiedata P1-device

De volgende subparagraaf van paragraaf 4.3 bevat technische specificaties en details.

4.3.2 Technisch: hoe de Data Push werkt

De Open API Data Push werkt als volgt:

- De bewoner kan maximaal 5 serveradressen instellen. Elke server krijgt 1 keer per minuut de actuele 10-secondewaarden.
- Het P1-device houdt per server(adres) bij tot welke timestamp de 10-secondedata succesvol is verzonden. Op die manier kan bij overbelasting van het P1-device of bij een andere (connectie)fout de server toch alle data ontvangen. Dit tot een maximale foutperiode van 15 minuten.
- Bij de eerste verbinding gaat de timestamp naar 'nu' en stuurt het P1-device alleen de actuele waardes (dus niet de al aanwezige 15 minuten aan data).
- Let op: als de gebruiker de opslag van historische data uitzet met de configuratie-interface, dan blijft het P1-device wel het register van 15 minuten aan 10-secondewaarden opslaan en doorgeven aan de server (maar deze data is niet meer opvraagbaar via /energyhistory).
- Het P1-device stuurt het serienummer (uniek nummer per device) naar de server in een custom HTTP-header "[P1device]-DeviceID". Dit zodat een externe server de P1-devices van elkaar kan onderscheiden. De header wordt dan bijvoorbeeld:
  - [P1device]-DeviceID : 34f3r325ff34
- De URL van de externe server die de bewoner instelt in de configuratie-interface is de basis-URL die de Open API gebruik om het security-entrypoint en data-entrypoint te construeren. Deze zijn:
  - Security-entrypoint: <server-URL>/login
  - Data-entrypoint: <server-URL>/energynow
- De Open API Data Push gebruikt AES GCM op dezelfde manier als de Open API Lokaal. Dit met hetzelfde wachtwoord voor applicaties.
- Het JSON-bericht dat een externe server ontvangt, ziet er als volgt uit:

```
{
   "response": {
      "interval": "tenseconds",
      "timestampFirstEntry": 1364988200,
      "timestampLastEntry": 1364988300,
      "entries": 5,
      "unitElectricity": "kWh",
      "unitElectricityPower" : "W",
      "unitGas": "m3",
      "highTariffElectricity": "electricityMeterI",
      "electricityMeterI": [12.345, 12.345, 12.345, 12.345, 12.345],
      "electricityMeterIReturn": [0.223, 0.233, 0.232, 0.232, 0],
      "electricityMeterII": [12.345, 12.345, 12.345, 12.345, 12.345],
      "electricityMeterIIReturn": [0.045, 0.045, 0.045, 0.045, 0.045],
      "electricityPower": [0.34, 0.34, 0.34, 0.34, 0.34],
      "electricityPowerReturn": [0.12, 0.12, 0.12, 0.12, 0.12],
      "gasMeter": [2320.122, 2320.122, 2319.871, 0, 0]
   }
}
```

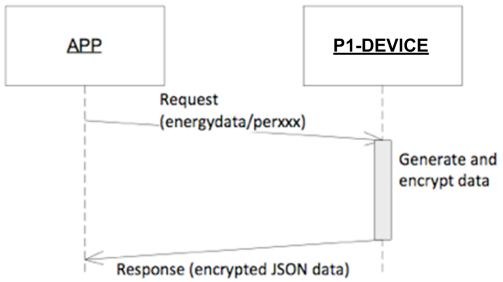
4.4 Beveiliging: gebruik van AES GCM met sessies

Randvoorwaarde voor de beveiliging is dat wanneer de bewoner zijn WiFi-netwerk niet (goed) heeft beveiligd, de energiedata daarmee niet op straat komt te liggen. Het P1-device moet er daarom voor zorgen dat onder normaal gebruik de verdiensten die een 'hacker' (iemand die de data wil stelen) krijgt, niet opwegen tegen de inspanning die hij moet doen om de data te bemachtigen.

Voor beveiliging van de Open API zijn de volgende keuzes gemaakt:

- De bewoner gebruikt 1 wachtwoord voor applicaties dat hij kan wijzigen in de configuratie-interface. Dit wachtwoord is hetzelfde voor alle apps en websites die een bewoner gebruikt.
- Authenticatie en encryptie met AES GCM (Advanced Encryption Standard, Galois/Counter Mode). AES is een block encryptiemethode die 16 bytes (128bit) per keer encrypt, GCM is een implementatie om meer dan 16 bytes te encrypten en wordt aanbevolen door het National Institute of Standards and Technology (NIST). GCM voorziet ook in authenticatie.
- Gebruik van een challenge/response-mechanisme met sessies om een key te genereren.
- Time-out: om enigszins beschermd te zijn tegen brute-force-attacks op het wachtwoord, zit er in de vierde en volgende mislukte inlogpogingen een time-out van 3 seconden voordat er een nieuwe poging gedaan kan worden. Deze time-out is systeembreed en niet per client.

Alleen de data van het P1-device naar apps bevat gevoelige informatie en dient te worden versleuteld, zie afbeelding 8:



Afbeelding 8: data-encryptie alleen response

Bij een verbinding van een app naar het P1-device moet de app een sessie bij het P1-device aanvragen. In de sessie wordt een sessiekey bepaald waarmee binnen de sessie de data wordt versleuteld. Het P1-device bepaalt wanneer een sessie is afgelopen.

De procedure voor de authenticatie en het genereren van een sessiekey is als volgt:

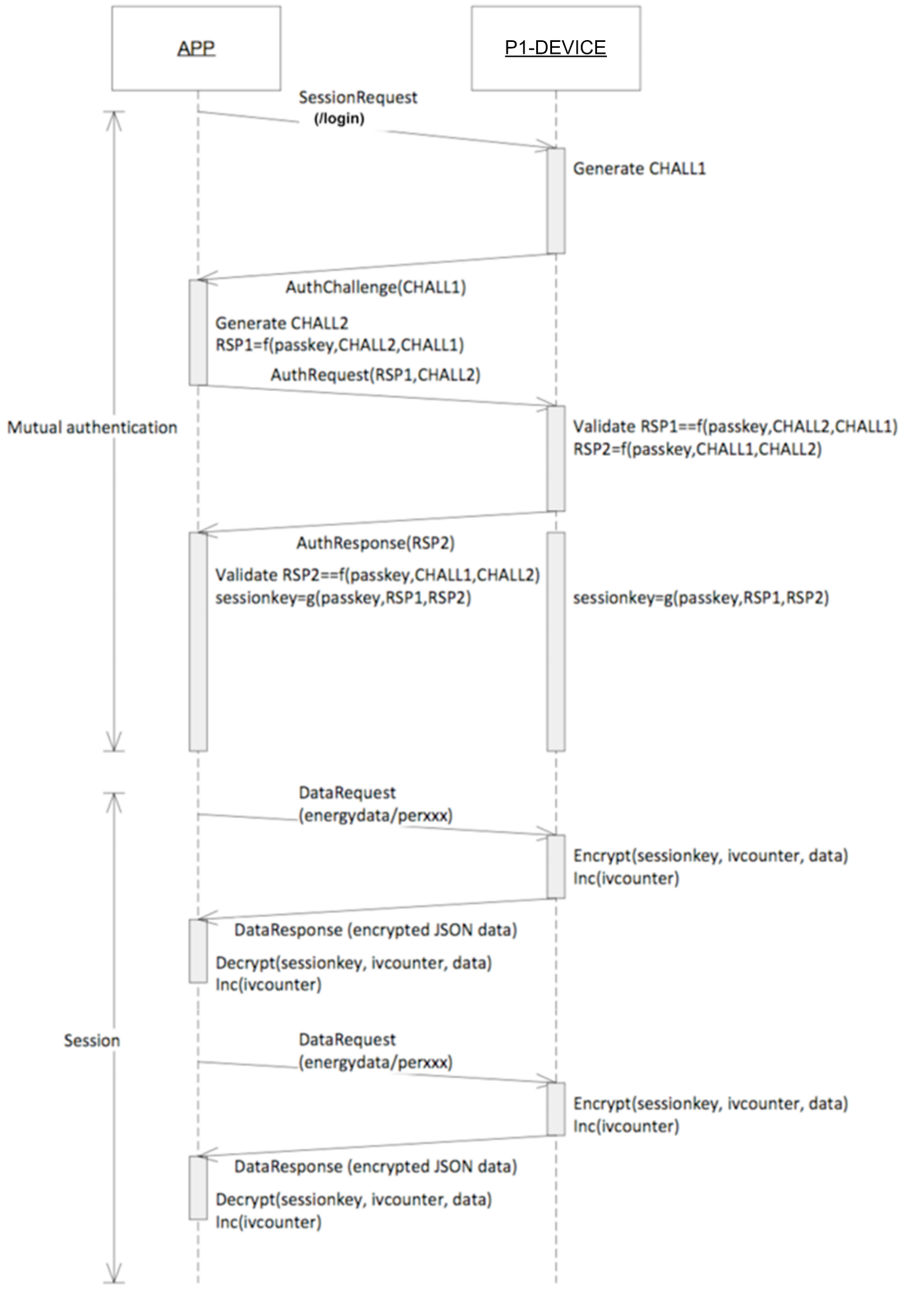
1. De app stuurt een SessionRequest naar het P1-device.
2. Het P1-device genereert een challenge (AuthChallenge) met een random getal (CHALL1) naar de app.
3. De app stuurt een autorisatie verzoek (AuthRequest) met hierin:
   a. Een random getal als challenge (CHALL2) voor het P1-device.
   b. Het antwoord op de challenge (RSP1) door de GMAC op CHALL1 te berekenen met de (van het wachtwoord afgeleide) key en CHALL2 als IV.
4. Het P1-device valideert RSP1. Bij een fout wordt de authenticatie gestopt door het P1-device.
5. Het P1-device antwoordt (AuthResponse) met response (RSP2) op de challenge van de app door de GMAC op CHALL2 te berekenen met de (van het wachtwoord afgeleide) key en CHALL1 als IV.
6. De app valideert RSP2. Bij een fout wordt de authenticatie gestopt door de app.
7. Het P1-device en de app genereren beide een sessiekey aan de hand van de originele key, RSP1 en RSP2 met een one-way functie.
8. De datarequests van de app worden door het P1-device beantwoord met de sessiekey versleutelde JSON-data.

Het P1-device stuurt een non-persistent sessiecookie met de naam “session” naar clients om deze te kunnen identificeren. Clients zullen deze sessiecookies zoals gebruikelijk bij elk request naar het P1-device sturen.

Als een client geen secured sessie heeft opgebouwd, dan stuurt deze de requests voor het opbouwe van een secured sessie naar een apart authentication-endpoint. Dit endpoint is: http://[p1-device].local/login

N.B.: Als de bewoner in de configuratie-interface een optie aanpast die invloed heeft op de Open API, dan verwijdert het P1-device alle lopende secured sessies. Bijvoorbeeld bij: wijzigen wachtwoord voor applicaties, uitzetten Open API en uitzetten opslag historische data.

Zie afbeelding 9 voor een schematische weergaven van het proces voor authenticatie en genereren van een sessiekey:

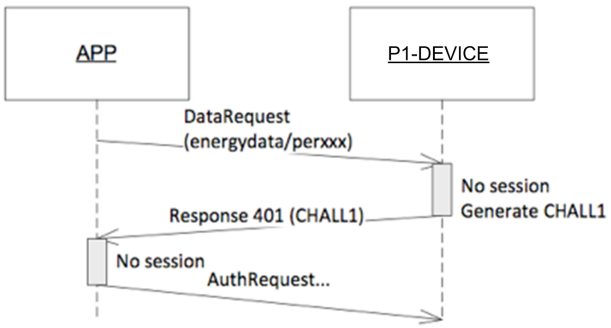


Afbeelding 9: proces van authenticatie en genereren sessiekey

Het P1-device beheert de sessies. Een sessie is per app en heeft een levensduur van 60 minuten. Het P1-device kan een of meerdere sessies tegelijkertijd toestaan (afhankelijk van beschikbare resources).

Het P1-device communiceert naar de app dat deze een nieuwe sessie moet aanvragen door op een datarequest te antwoorden met een 401-response. De app zal dan de procedure om een sessiekey te genereren opnieuw moeten uitvoeren.

Zie afbeelding 10 voor een schematische weergave van het proces wanneer er geen sessie beschikbaar is:



Afbeelding 10: proces wanneer er geen sessie beschikbaar is

Zie bijlage D voor de details van de encryptie- en authenticatiebewerkingen nodig voor het opbouwen van een sessie en verzenden van data.

Motivatie voor gemaakte keuzes:
- Het wachtwoord voor applicaties is een ander wachtwoord dan het wachtwoord voor de configuratie-interface. Hierdoor kunnen apps geen ongeautoriseerde toegang krijgen tot de configuratie-interface (en vervolgens de configuratie aanpassen).
- Door slechts 1 wachtwoord voor applicaties te gebruiken voor de Open API (en niet 1 wachtwoord per app) wordt de beheerlast van de bewoner beperkt. Ook kan de bewoner hiermee in 1 keer de toegang tot energiedata van alle apps uitschakelen (door het wachtwoord in de configuratie-interface te wijzigen).
- Advanced Encryption Standard (AES) is een wereldwijd geaccepteerde standaard van U.S. National Institute of Standards and Technology (NIST) voor het encrypten van gegevens en wordt bijvoorbeeld ook gebruikt in de Wireless Meter Bus (Wireless MBus) specificatie.
- Er is voor gekozen om met sessies te werken en niet het wachtwoord voor applicaties (of afgeleide daarvan) als key in de encryptie te gebruiken. Dit omdat een kwaadwillende de versleutelde data zou kunnen opvragen en met een ferme inspanning de key behorende bij het wachtwoord zou kunnen achterhalen.

4.5 Afhandeling van fouten

De webserver hanteert HTTP-statuscodes om het slagen of falen van een uitgevoerd HTTP-request aan te geven. Dit betekent dat de Open API voldoet aan de HTTP/1.1 Status Code Definitions zoals vastgesteld door het W3C.

De Open API geeft informatie (toelichting/uitleg) in de responsebody bij fouten, zodat appdevelopers sneller kunnen vaststellen wat er mis is gegaan en eventuele fouten in de appcode corrigeren. In de W3C-specificatie is dat aangegeven met: “the server SHOULD include an entity containing an explanation of the error situation, and whether it is a temporary or permanent condition.”

Als het HTTP-request voor ophalen van energiedata een fout bevat, bijvoorbeeld een 'out of bounds'-bereik, dan reageert het P1-device hierop door de data terug te geven die wel

beschikbaar is. Bijvoorbeeld: opvragen van entries tot 2.000 voor de data per jaar, geeft de entries tot 1.827 of tot zover beschikbaar op dat moment.

Zie Tabel 3 voor de HTTP-statuscodes die het P1-device in bepaalde gevallen geeft:

Tabel 3: HTTP-statuscodes Open API

#	Code	Body	Toelichting
1	200 OK		Request is succesvol afgehandeld, de body bevat de gevraagde data.
2	200 OK	OpenAPI session freed	Bij succesvolle afsluiting van de sessie na aanroep /logout.
3	400 Bad Request	Invalid JSON: [explanation]	Als het P1-device het JSON-bericht niet kan interpreteren. Geef uitleg/details in de body.
4	401 Unauthorized	Unable to decrypt data	Als het P1-device de data niet kan decrypten.
5	401 Unauthorized	Unable to start session: [explanation]	Als het starten van een secured sessie mislukt. Geef uitleg/details in de body.
6	401 Unauthorized	Session expired or non-existent	Als de secured sessie is verlopen of als er nog geen secured sessie is opgebouwd.
7	403 Forbidden	User has switched off Open API	Als de bewoner de Open API uit heeft staan
8	404 Not Found		Bij het opvragen van een niet bestaande URI (niet bestaande query parameters worden genegeerd)
9	404 Not Found	User has switched off energy history	Als de bewoner de opslag van historie uit heeft staan
10	408 Request Timeout		De app doet er te lang over om een request te sturen.
11	500 Internal Server Error		Als er iets onverwachts mis is gegaan in het P1-device.
12	500 Internal Server Error	Request expired	Bij Open API Externe Toegang: het request is meer dan 3 keer opnieuw geprobeerd of heeft langer dan 1 minuut in de queue gestaan.
13	500 Internal Server Error	Tunnel queue full (keep-alive)	Bij Open API Externe Toegang: als er gedurende een reconnect tussen Ectual en externe server een OpenAPI request van de applicatie binnenkomt terwijl de tunnel request queue vol zit.
14	500 Internal Server Error	Tunnel disconnected	Bij Open API Externe Toegang: als de externe server (om wat voor reden dan ook) besluit de verbinding naar de applicatie te verbreken en er nog requests in de queue stonden. Redenen hiervoor zijn onverwachte foutsituaties waar verder niets aan te doen valt.
15	503 Service Unavailable		Als het P1-device niet genoeg resources beschikbaar heeft om het request af te handelen. Onder andere als het maximum aantal HTTP-sessies of secured sessies is bereikt.
16	503 Service Unavailable	Tunnel queue full	Bij OpenAPI Externe Toegang: als de externe server een Open API-request van de applicatie ontvangt terwijl de tunnel request

			queue vol zit.
17	504 Gateway Timeout	Tunnel not active	Als een applicatie via de Open API Externe Toegang een endpoint opvraagt anders dan /accessrequest en er nog geen verbinding met het P1-device (tunnel) beschikbaar is.

4.6 Richtlijnen en best practices voor de performance

Deze paragraaf bevat richtlijnen voor de performance van de Open API Lokaal en Externe Toegang en best practices voor applicatie-ontwikkelaars hoe de performance maximaal te benutten.

4.6.1 Performance en best practices Open API Lokaal

De Open API Lokaal voldoet minimaal aan de volgende performance:
- Het ophalen van de actuele energiedata (energynow) duurt maximaal 1,5 seconden.
- Het ophalen van alle energiedata per minuut voor een maand duurt maximaal 10 seconden. Dit betekent een doorvoersnelheid van historische energiedata (energyhistory) van ongeveer 4.500 entries per seconde.

Apps mogen maximaal 10 keer per seconde (of: elke 100 ms) een request sturen naar het P1-device. Dit om de load op het device te beperken en deze niet te overladen. Als het device onvoldoende resources heeft om een request te beantwoorden, dan zal er een Connection Reset plaatsvinden op TCP-niveau. Dat is een teken voor een app om het ‘rustiger aan’ te doen of om te detecteren dat er momenteel veel apps om data vragen.

De best practice is dat apps in de GUI een handeling niet 1-op-1 vertalen naar een datarequest voor het P1-device, maar daar als volgt mee omgaan:
- Request 1: minimaal 100ms na het vorige request
- Bij geen response (Connection Reset) nog 4 pogingen met oplopende pauzes: 200, 300, 400 en 500 ms.
- Als het device dan nog niet benaderbaar is, een melding aan de gebruiker geven. Bijvoorbeeld: "P1-device heeft het heel druk momenteel, nog een keer proberen?"

4.6.2 Performance en best practices Open API Externe Toegang

De verbinding met het P1-device via Externe Toegang is anders dan via Lokaal: de externe server serialiseert alle requests naar de Ectual. Dit betekent dat een applicatie effectief maar 1 request tegelijk kan hebben uitstaan. De externe server heeft een queue van minimaal 3 requests, overeenkomend met de drie sessies die het P1-device minimaal tegelijk aan kan.

Een applicatie kan daarmee in praktijk wel 3 requests hebben uitstaan, maar dit is niet de beoogde use case van de queue. De use case is dat 3 applicaties tegelijkertijd 1 request kunnen hebben uitstaan (waarbij dus mogelijk de latency van de twee requests van de andere tunnel gebruikers bovenop de latency van het eigen request komt!).

Een queue-slot blijft in gebruik zolang het request uitstaat. Pas als het volledige antwoord van het P1-device is afgeleverd aan de applicatie, dan wordt het slot weer vrijgegeven.

Omdat te zorgen dat de verbinding tussen het P1-device en de externe server in stand blijft en om te zorgen dat een weggevallen verbinding snel gedetecteerd wordt, stuurt de externe server om de 4 seconden een ‘keep alive’-bericht naar het P1-device.

De best practices voor gebruik van de Externe Toegang zijn:
- Zet maximaal 1 request tegelijk uit: wacht altijd tot je een reply hebt gekregen voordat je een nieuw request uitzet.

- Doe alleen een request als je de informatie hiervan ook daadwerkelijk gaat gebruiken (voorkom bijvoorbeeld dat scrollen door de historie resulteert in een stapeling van requests).
- Zorg dat je app goed omgaat met de mogelijk hoge latencies en lage bandbreedte die een verbinding via de Externe Toegang met zich mee kan brengen.

5 Software Development Kit (SDK) voor appdevelopers

Het doel van een Software Development Kit (SDK) is om het de appdeveloper makkelijker te maken apps te ontwikkelen voor het P1-device. Dit maakt de drempel lager om aan de slag te gaan met het P1-device. En hoe sneller een app gemaakt is, met een hogere kwaliteit.

Op dit moment bevat de SDK:

- Terms & Conditions: onder welke voorwaarden mag een softwareontwikkelaar de SDK en het P1-device gebruiken.
- CryptoJS Library met AES GCM-uitbreiding: zoals ontwikkeld voor de iOS en Android apps
- Referentie-applicatie in Titanium (iOS and Android): de broncode en documentatie van de iOS en Android apps
- Referentie-applicatie Windows: de broncode en documentatie van de Windowsapp
- Simulator: de ontwikkelde simulator met broncode en documentatie
- Voorbeeldcode voor het gebruiken van de Open API Data Push
- Specificaties: dit document

Daarnaast heeft de ondersteunende website een ‘support’-gedeelte waar appdevelopers een Forum en Wiki kunnen gebruiken.

6 Simulator en Referentieapplicaties

Dit hoofdstuk bevat een beknopte samenvatting van de software die ondersteunend is aan de P6 Companion Standard.

6.1 Simulator P1-device

De Simulator is een softwarematige simulatie van het P1-device. Dit om appdevelopers de kans te geven om apps te ontwikkelen voor de Open API. Ook appdevelopers die (nog) geen beschikking hebben over het ontwikkelde device. De Simulator draait op de computer van een appdeveloper. Hiermee kan deze verschillende gebruiksscenario's (parameters) testen op een app in ontwikkeling.

De Simulator heeft de volgende eigenschappen (samengevat):
- Draait op Windows, Mac OS en Linux
- Ondersteunt mDNS
- Ondersteunt de volledige Open API, inclusief AES GCM en HTTP-foutcodes
- Is geparameteriseerd betreffende de instellingen die een gebruiker in de Management Tool kan doen: wel/geen elektriciteit retour, 1 of 2 telwerken, historie-opslag aan/uit, etc.
- Heeft een grafische en een XML-interface om de instellingen aan te passen
- Genereert geparameteriseerd een historische database
- Toont relevante debugdata

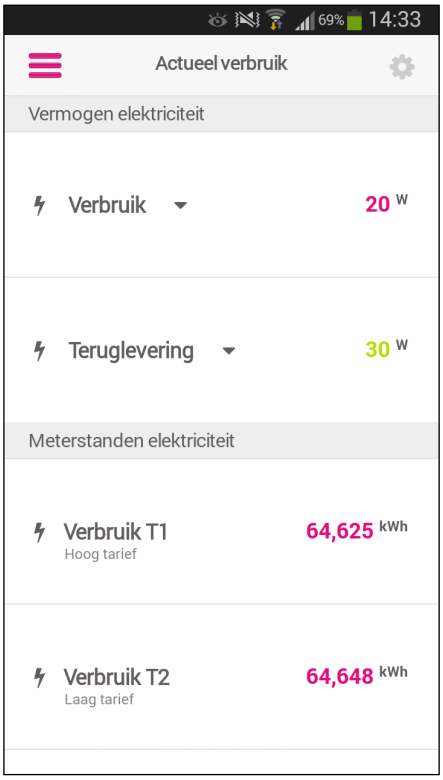
6.2 Mobiele Referentie-applicaties: iPhone en Android

De mobiele referentieapplicaties zijn functioneel eenvoudige apps voor iOS en Android om de werking van Open API te demonstreren. Ook kunnen appdevelopers de broncode bekijken en hergebruiken in nieuwe te ontwikkelen toepassingen.

De apps hebben de volgende functionaliteiten (samengevat):
- Vinden van het P1-device in het thuisnetwerk met Device Discovery en via de detectieserver
- Demonstratiemode als er geen P1-device beschikbaar is
- Laten invoeren en verifiëren van het wachtwoord voor applicaties
- Verbinden via de Open API Lokaal en Externe Toegang
- Tonen van actuele gegevens: meterstanden en vermogen
- Tonen van historie: per uur, dag en maand
- Omrekenen van kWH/m3 naar euro's.
- Tonen van de status van het P1-device
- Tonen van debuginformatie
- Hulpfunctie en links naar de gebruikerswebsite

Zie bijlage C voor een procesdiagram van hoe de mobiele referentieapps verbinding maken met het P1-device.

Zie hieronder afbeelding 11 voor een schermvoorbeeld van de mobiele apps:



Afbeelding 11: schermvoorbeeld mobiele referentie-applicaties

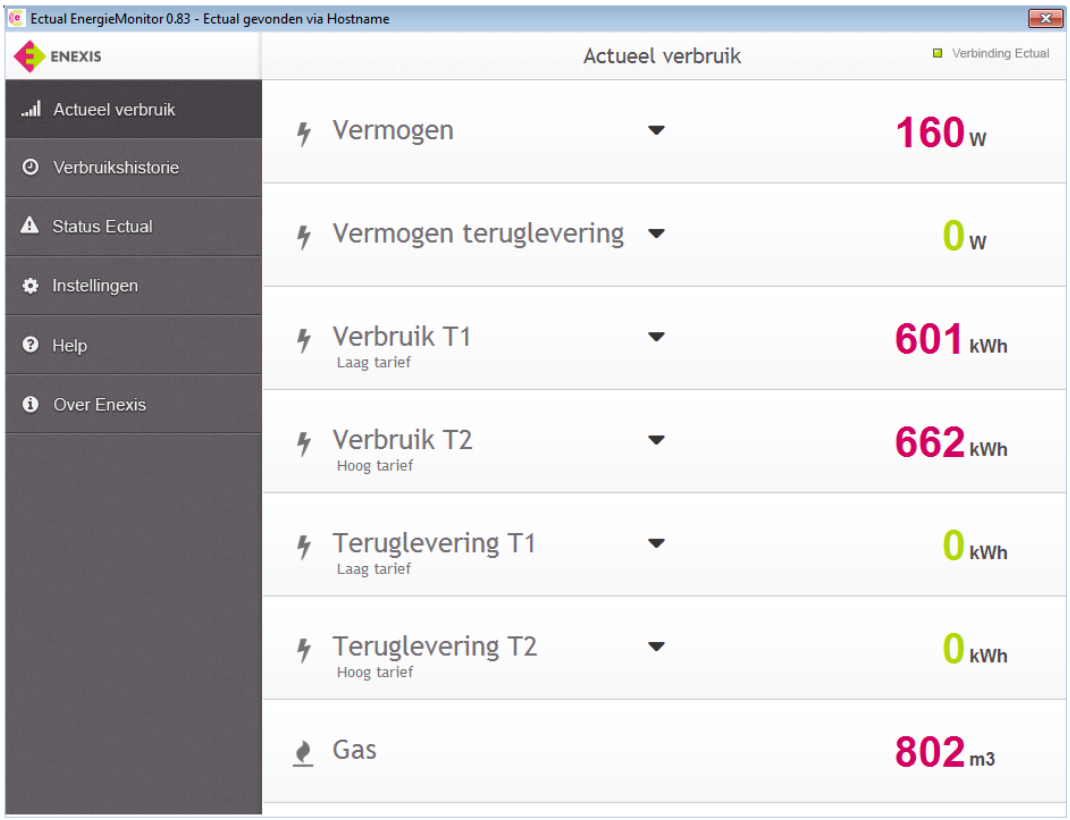
6.3 Referentie-applicatie voor Windows

Naast de mobiele referentieapps is er ook een referentieapp voor Windows. Deze heeft in basis dezelfde functionaliteiten als de mobiele versies en heeft verder de volgende functies:

- Meer grafieken (omdat er meer schermruimte beschikbaar is)
- Continue opslaan van de actuele energiedata in een CSV-bestand
- Uitgebreide status- en debuginformatie inclusief exportmogelijkheid

Zie bijlage D voor een procesdiagram van hoe de Windowsapplicatie verbinding maakt met het P1-device.

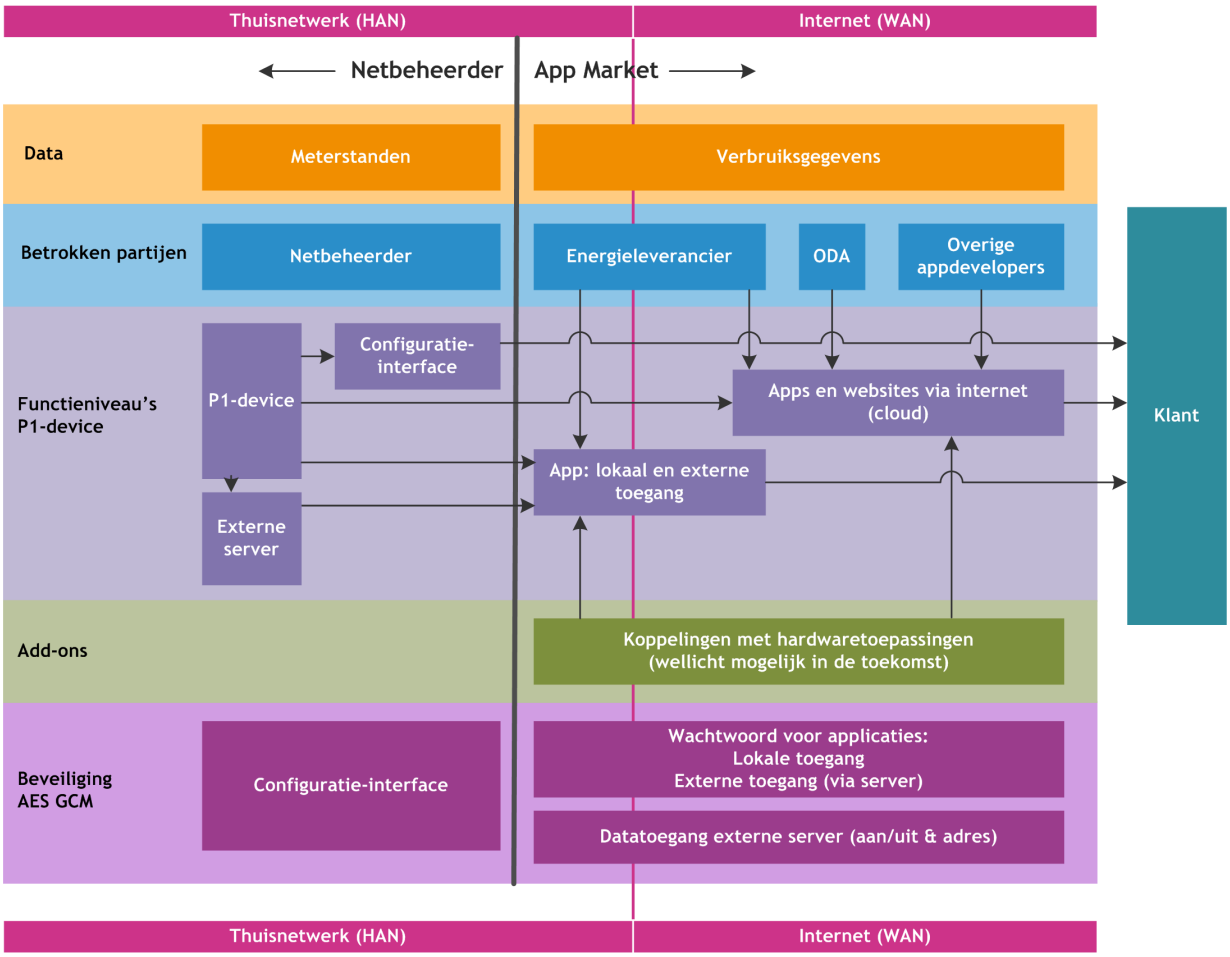
Zie hieronder afbeelding 12 voor een schermvoorbeeld van de Windowsapp:



Afbeelding 12: schermvoorbeeld referentie-applicatie Windows

Bijlage A: overzicht functionaliteiten P1-device

Afbeelding 13 bevat een overzicht van de functionaliteiten van het P1-device:

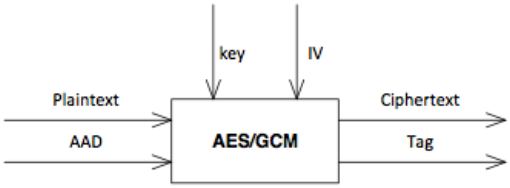


Afbeelding 13: overzicht functionaliteiten P1-device

Bijlage B: encryptie- en authenticatiebewerkingen AES GCM

Deze bijlage bevat de detailspecificaties van de encryptie- en authenticatiebewerkingen om een sessie op te bouwen (genereren sessiekey) en versturen van data via AES GCM.

De encryptie- en authenticatiebewerkingen kunnen we beschouwen als een proces, zie afbeelding 14:



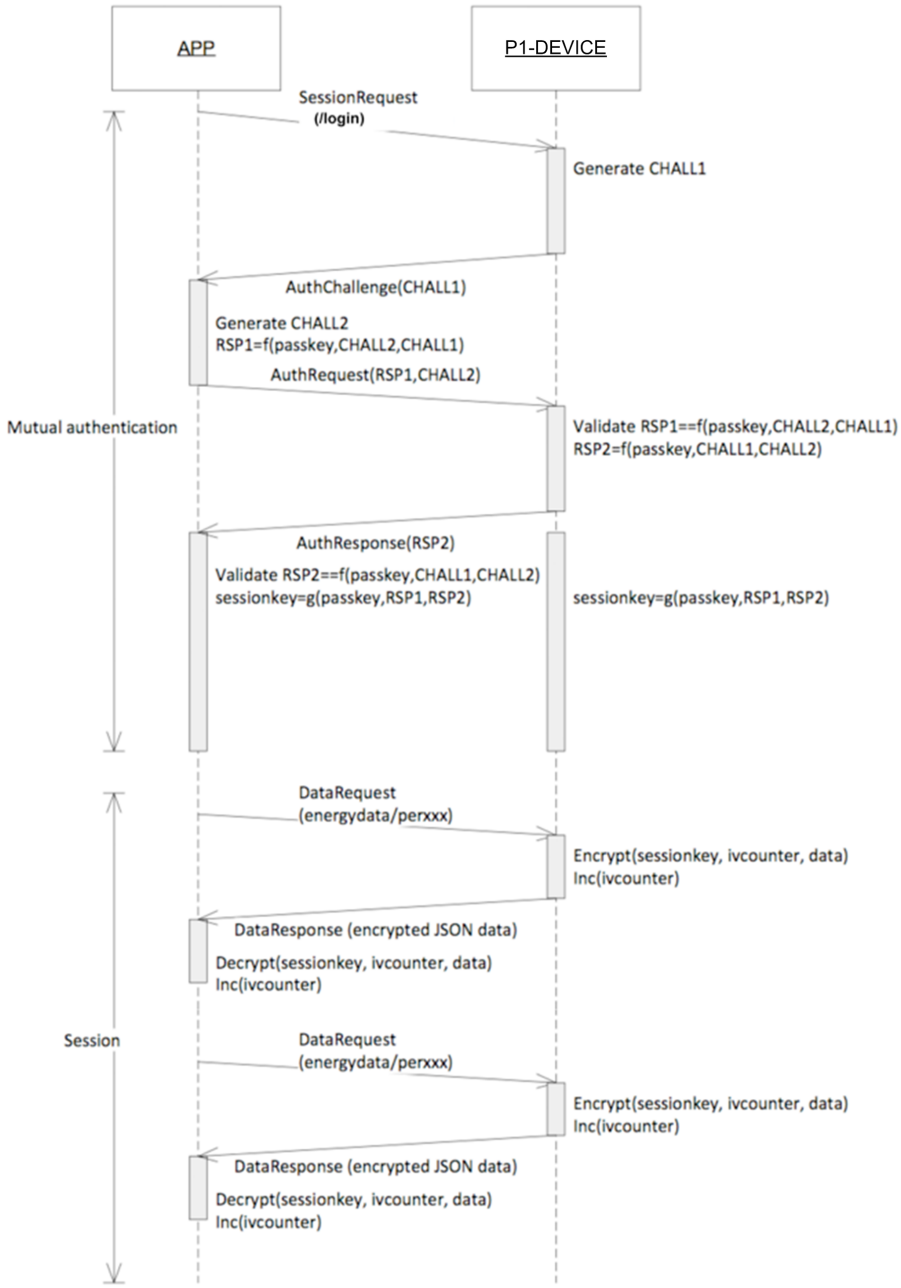
Afbeelding 14: weergave AES GCM als proces

De volgende tabel bevat en toelichting op de ingangen en uitgangen van dit proces:

Naam	Type	Grootte	Omschrijving
Key	Parameter	128-bit	Een binaire sleutel gebruikt voor het encrypten.
IV	Parameter	128-bit	Initialisation Vector, een random getal dat wordt gebruikt bij encryptie om het herkennen van patronen moeilijk te maken.
Plaintext	Invoer	*	Data dat geëncrypt en geauthentiseerd moet worden.
AAD	Invoer	*	Aditional Authenticated Data, data dat alleen geauthentiseerd moet worden.
Ciphertext	Uitvoer	*	Geëncrypte data
Tag	Uitvoer	128-bit	Authenticatietag

Hierbij opgemerkt:
- Voor de decryptie zijn Plaintext en Ciphertext verwisseld.
- Voor de encryptie blijft ADD leeg.
- Voor de GMAC-berekening blijft Plaintext leeg en is juist AAD gevuld.

De volgende 9 paragrafen (9 stappen) bevatten een uitwerking van afbeelding 9 (proces van authenticatie en genereren sessiekey) in paragraaf 4.4. Voor de volledigheid is deze figuur hieronder nogmaals opgenomen:



Afbeelding 15: proces van authenticatie en genereren sessiekey (herhaling afbeelding 9)

De volgende tabel bevat de variabelen die nodig zijn:

Naam	Grootte	Omschrijving
passphrase	>18 Chars	De passphrase die de gebruiker heeft gekozen. Dit is een string die speciale tekens mag bevatten.
passkey	128-bit	Een van de passphrase afgeleide key.
CHALL1	128-bit	Een binair getal dat het P1-device genereert.
CHALL2	128-bit	Een binair getal dat de APP genereert.
RSP1	128-bit	De uitkomst van een bewerking op CHALL1 door de APP.
RSP2	128-bit	De uitkomst van een bewerking op CHALL2 door het P1-device.
sessionkey	128-bit	Een key die voor de sessie gemaakt wordt, afhankelijk van RSP1 en RSP2.
ivcounter	128-bit	Een IV die begint bij 0 en na elk data request wordt opgehoogd.

1. Genereren passkey
De passkey wordt gegenereerd door een SHA-256 hash te nemen van de passphrase. Een SHA-256 levert een 256-bit (32-byte) getal op waarvan we de eerste 128-bit (16- byte)gebruiken als key.

```
hash = SHA-256(passphrase) # genereren een hash van de passphrase
passkey = hash(0:15) # nemen de eerste 16 bytes van de hash
```

2. Genereren CHALL1
CHALL1 wordt gegenereerd door een 128-bit (16-byte) random getal.

```
CHALL1 = Random(16) # 16-byte random getal
```

3. Genereren CHALL2
CHALL2 wordt gegenereerd door een 128-bit (16-byte) random getal.

```
CHALL2 = Random(16) # 16-byte random getal
```

4. RSP1 = f(passkey,CHALL2,CHALL1)
De RSP1 wordt berekend door de GMAC te berekenen over CHALL1 met de volgende eigenschappen:

Naam	Omschrijving
Key	passkey
IV	CHALL2
Plaintext	leeg
AAD	CHALL1
Ciphertext	leeg
Tag	Uitvoer naar RSP1

5. RSP2 = f(passkey,CHALL1,CHALL2)
De RSP2 wordt berekend door de GMAC te berekenen over CHALL2 met de volgende eigenschappen:

Naam	Omschrijving
Key	passkey
IV	CHALL1
Plaintext	leeg
AAD	CHALL2
Ciphertext	leeg
Tag	uitvoer naar RSP2

6. Sessionkey=g(passkey,RSP1,RSP2)
De sessionkey wordt berekend door de GMAC te berekenen over RSP1 en RSP2 met de volgende eigenschappen:

Naam	Omschrijving
Key	passkey
IV	0x00000000000000000000000000000000
Plaintext	leeg
AAD	RSP1 xor RSP2[1]
Ciphertext	leeg
Tag	uitvoer naar sessionkey

RSP1 xor RSP2: binaire XOR-operatie. Invoer twee keer een 128-bit getal, uitvoer een 128-bit getal.

[1] Binaire XOR operatie. Invoer 2x 128-bit getal, uitvoer een 128-bit getal

7. Encrypt(sessionkey,ivcounter,data)
De data wordt door middel van AES/GCM geëncrypt met de volgende eigenschappen:

Naam	Omschrijving
Key	sessionkey
IV	ivcounter
Plaintext	de JSON-data zoals gespecifieerd in de OpenAPI
AAD	leeg
Ciphertext	binaire uitvoer
Tag	binaire uitvoer (als toevoeging op Ciphertext)

8. Decrypt(sessionkey,ivcounter,data)
De data wordt door middel van AES/GCM gedecrypt met de volgende eigenschappen:

Naam	Omschrijving
Key	sessionkey
IV	ivcounter
Ciphertext	geëncrypte data, zonder tag
AAD	leeg
Plaintext	binaire uitvoer ongeëncrypte data
Tag	binaire uitvoer als verificatie van de ontvangen tag

9. Inc(ivcounter)
De IV parameter moet per sessionkey uniek zijn. Dit wordt gedaan door de ivcounter bij het tot stand komen van een sessie op nul te zetten en na elke data transport te verhogen met 1.

```
ivcounter = ivcounter + 1
```

De ivcounter wordt aan de kant van de app en het P1-device na elk datatransport opgehoogd. Mocht (door bijvoorbeeld een fout tijdens transport) de ivcounter uit sync lopen, dan zal de app een fout ontdekken tijdens het decrypten en zal een nieuwe sessie moeten starten.

Berichtdefinities
De berichten die gebruikt worden voor het maken van een sessie zijn HTTP POST requests naar http://[p1-device]/login. De volgende paragrafen bevatten de uitwerking hiervan.

Hierbij opgemerkt:
- De eventuele inhoud van de POST-berichten is in JSON-formaat.
- Bij gebruik van de Open API Externe toegang moet het content-type van het POST-request juist zijn: “application/json”. De externe server controleert hierop en geeft een foutmelding als het content-type niet klopt. Bij de Open API Lokaal en Open API Data Push wordt hier niet op gecontroleerd.
- De binaire data is als Base64 gecodeerd.

SessionRequest (POST)

Richting	APP → P1-device
Volgt op	1. Het opstarten van de APP 2. Een fout bij het decoderen
Doel	Een nieuwe sessie tot stand brengen
Inhoud	`{“action”: “sessionrequest”}`
Voorbeeld	`{“action”: “sessionrequest”}`

AuthChallenge

Richting	P1-device → APP
Volgt op	1. SessionRequest 2. AuthRequest die niet geaccepteerd is
Doel	De APP uitdagen
Inhoud	{"action": "authchallenge", "challenge1": [challenge1 in base64 formaat]}
Voorbeeld	{"action": "authchallenge", "challenge1": "BkSGGqIqSv90s9QQiYSQkXVDvPg\u003d"}

AuthRequest (POST)

Richting	APP → P1-device
Volgt op	AuthChallenge
Doel	1. Op de uitdaging van het P1-device reageren 2. Het P1-device uitdagen
Inhoud	{"action": "authrequest", "response1": [response1 in base64 formaat], "challenge2": [challenge2 in base64 formaat]}
Voorbeeld	{"action": "authrequest", "response1": "BkSGGqIqSv90s9QQiYSQkXVDvPg\u003d", "challenge2": "BkSGGqIqSv90s9QQiYSQkXVDvPg\u003d"}

AuthResponse

Richting	P1-device → APP
Volgt op	AuthRequest
Doel	Op de uitdaging van het P1-device reageren
Inhoud	{"action": "authresponse", "response2": [response2 in base64 formaat]}
Voorbeeld	{"action": "authresponse", "response2": "BkSGGqIqSv90s9QQiYSQkXVDvPg\u003d"}

DataRequest (GET)

Richting	APP → P1-device
Volgt op	1. AuthResponse 2. DataRequest
Doel	Ophalen van data
Inhoud	
Voorbeeld	

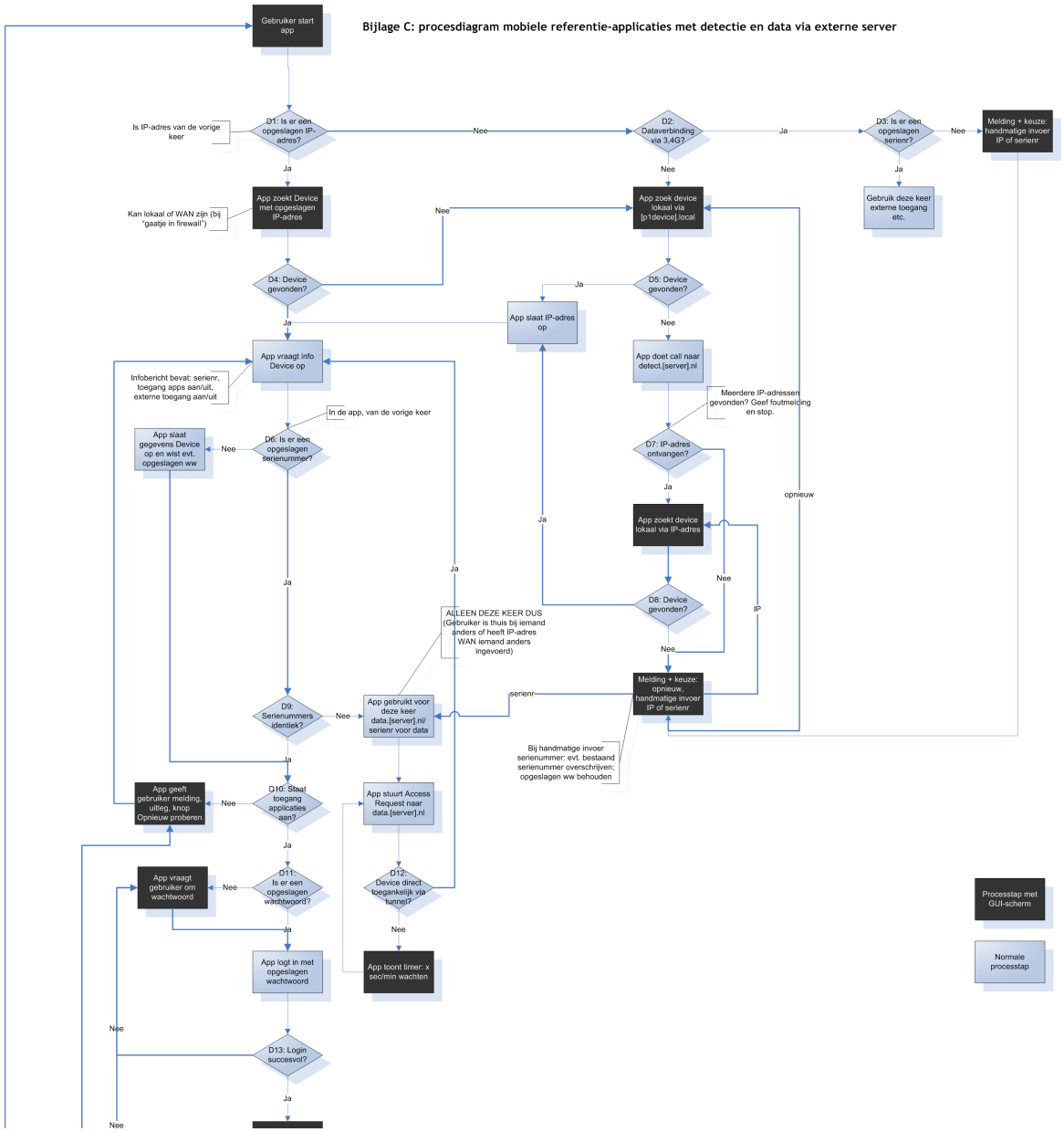
DataResponse

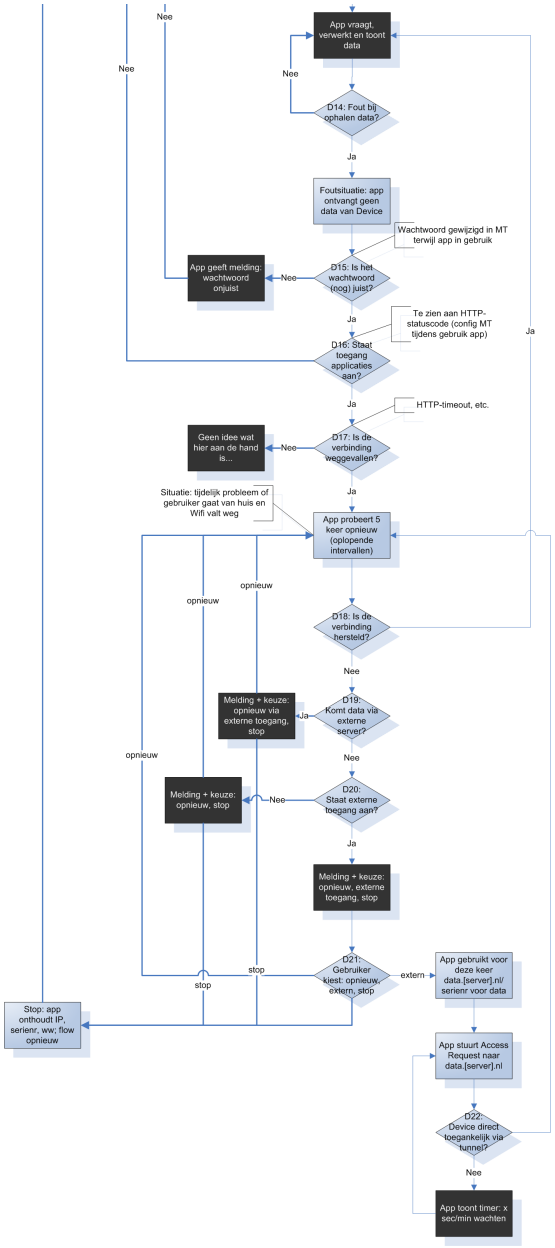
Richting	P1-device → APP
Volgt op	DataRequest
Doel	Opsturen van data
Inhoud	[geëncrypte data + tag in base64 formaat]}
Voorbeeld	tMSy2PrbEXZwKOyZwqTY5rkFImNC7y/hPzvuaw132qW7vxxle8m+Yndsdgc5Kk9tua8R3R+OKRbtZcb9pwTIe1ndczEkyV/+hKz3V4pKKVccjusqmMsf2TlnSXGnucElHDI02AW4UphAq9BEsDdGhk+a3xake+z48KHCKbsDlN2lAhLbK1jxE6mbp/2AWZ3fui/6B0JWYRVgJnGiZCDoEVueouclVtBpspZaudX3iyzar+anaSnQddhsdbfNUVRNlzY3DqBmdsXnkj3YEpMdu5Kcq5Guvod5/mT+gPG/SbS+zZj1GxaAiJ3QsiUf3RlRTa/s0BTSJ+

	JcRJl3ZiElZs/prxdA7gzRNvY

Bijlage C: procesdiagram mobiele referentie-applicaties met detectie en data via externe server

Deze bijlage bevat een procesdiagram dat weergeeft hoe de mobiele referentie-applicaties verbinding maken met het P1-device: direct en via de externe detectie- en dataserver. Het diagram is als apart document bijgevoegd voor de leesbaarheid (PNG-formaat).

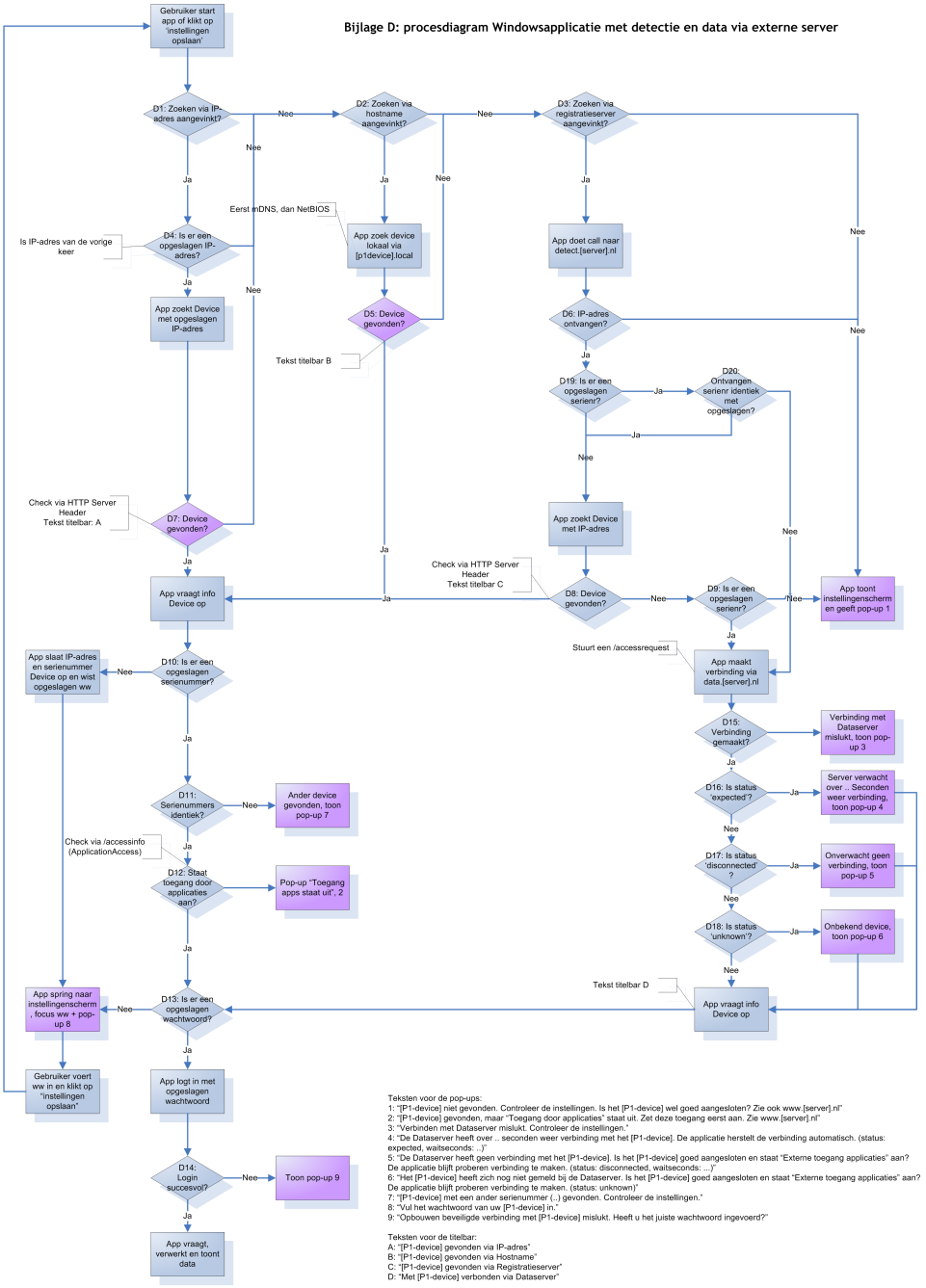




Afbeelding 16: procesdiagram verbinding mobiele referentie-applicaties

Bijlage D: procesdiagram Windowsapplicatie met detectie en data via externe server

Deze bijlage bevat een procesdiagram dat weergeeft hoe de referentie-applicatie voor Windows verbinding maakt met het P1-device: direct en via de externe detectie- en dataserver. Het diagram is als apart document bijgevoegd voor de leesbaarheid (PNG-formaat).



Afbeelding 17: procesdiagram Windowsapplicatie

Bijlage E: ideeën en wensen voor volgende versies

Deze bijlage bevat ideeën en wensen voor volgende versies van het P1-device. Deze ideeën en wensen komen van verschillende partijen, ontstaan tijdens gebruik van het P1-device Ectual:

1. Optie aan Open API toevoegen waarmee het mogelijk is om slechts een beperkt aantal datapunten op te vragen in een reeks, via het aangeven van een interval, een zogenaamde ‘step’. Bijvoorbeeld: opvragen van minutendata om de 10 minuten, de step is dan 10. Voor minutendata van een uur krijg je dan 60/10 = 6 entries (0, 10, 20, 30, 40 en 50). Use case: grafiek tekenen met intervallen van 10 minuten. Het opvragen van minutendata geeft een onnodige impact op het P1-device en de performance.
2. (nog verder aan te vullen uit gebruikservaringen)

Bijlage F: gebruikte standaarden en best-practices

De API maakt gebruik van de volgende standaarden en best-practices:

HTTP	Hypetext Transfer Protocol, de standaard voor applicatiedatauitwisseling op het World Wide Web. Zie: http://www.w3.org/Protocols/ http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html
JSON	JavaScript Object Notation, een veelgebruikte tekstgebaseerde standaard voor data-uitwisseling. Het voordeel is dat het formaat eenvoudig voor mensen te lezen is. Ook is het (meestal) compacter dan data in XML. Zie: http://www.json.org/
mDNS	Multicast DNS (mDNS) is een veel- (en steeds meer) gebruikte manier om apparaten in een netwerk eenvoudig te ontdekken, wanneer er geen DNS server beschikbaar is. Bijvoorbeeld bij een lokaal thuisnetwerk. Zie: http://www.multicastdns.org/
NetBIOS	NetBIOS is een ouder protocol om computers (apparaten) elkaar te laten vinden in een netwerk. Het is vooral relevant voor Windowscomputers. Zie: http://en.wikipedia.org/wiki/NetBIOS
LLMNR	LLMNR is een modern protocol voor apparaatherkenning in Windows en is de opvolger van NetBIOS. Zie: http://en.wikipedia.org/wiki/Link-local_Multicast_Name_Resolution
DHCP Hostname	DHCP Hostname is een manier om de Router in het netwerk de naam van een device te laten registreren. Zie: http://en.wikipedia.org/wiki/Dynamic_Host_Configuration_Protocol
Unix time	Unix time is een veelgebruikte standaard om timestamps op te slaan en te communiceren. Het voordeel is dat er in de software eenvoudig en snel mee te rekenen is. Zie: http://en.wikipedia.org/wiki/Unix_time

Bijlage G: gehanteerde definities in dit document

Deze bijlage bevat definities van een aantal belangrijke begrippen in dit document.

AES	Advanced Encryption Standard, een methode om blokken van 16 bytes te encrypten. Gespecificeerd door het NIST in FIPS PUB 197.
App	Een stuk software dat een bewoner kan installeren op een van zijn devices (zie devices van bewoners) en dat gebruik maakt van de energiedata om de bewoner een interessante functionaliteit te bieden. De bewoner installeert een app normaalgesproken uit een App Store, bijvoorbeeld de Apple Store of Google Play.
Appdeveloper	Een persoon of bedrijf die een app maakt, onderhoudt en beschikbaar stelt aan bewoners.
Bewoner	Een persoon die in een huis in Nederland woont welke een slimme meter en het P1-device heeft.
Challenge/Response	Methode om te controleren of de andere kant over de benodigde key beschikt. De Challenge is een random getal dat door de andere kant bewerkt moet worden om zo te valideren dat de andere kant over de benodigde key beschikt. De Response is de uitkomst van de bewerking op de Challenge. In dit document zijn de challenges en responses altijd 128-bit (16 bytes) groot.
P1-device	Het apparaat dat een bewoner in zijn huis krijgt om energiedata uit de slimme meter aan hem beschikbaar te stellen, via een Open API.
Devices van bewoners	Computers en mobiele apparaten van bewoners waarop zij apps kunnen gebruiken en een webbrowser starten, zoals: PC, laptop, tablet, iPhone, Android Phone, etc.
GCM	Galois/Counter Mode, een methode om meerdere blokken met AES te encrypten en authentiseren. Gespecificeerd in NIST Special Publication 800-38D.
GMAC	Een methode om data te authentiseren, het is een sub set van GCM. Gespecificeerd in NIST Special Publication 800-38D.
IV	Initialisation Vector, een random getal dat wordt gebruikt bij encryptie om het herkennen van patronen moeilijk te maken. In dit document is een IV altijd 128-bit (16 bytes) groot.
Key	Een binaire sleutel gebruikt voor het encrypten. In dit document is een key altijd 128-bit (16 bytes) groot.
Configuratie-interface	De software die bewoners in staat stelt om het P1-device te configureren.
NIST	National Institute of Standards and Technology.
Open API	De voor apps te gebruiken data-interface om energiedata uit het P1-device te halen.
P1-device	Een elektronisch apparaatje dat bewoners aan de P1-poort aansluiten, dat de data uit de P1-poort leest en dat deze data direct of indirect nuttig maakt voor de bewoner

Bijlage H: bronnen en referenties

De volgende bronnen zijn mede bekeken voor het opstellen van dit document:

Green Button	Een gestandaardiseerd open dataformaat voor het opslaan en uitwisselen van energiedata in de Verenigde Staten. Zie: http://www.greenbuttondata.org/ http://www.data.gov/energy/page/welcome-green-button
Tendril API	Tendril is een bedrijf dat onder andere een oplossing (software) biedt aan energiebedrijven om hun grote hoeveelheden data van het 'smart grid' op te slaan. Zij hebben een API voor appdevelopers gemaakt waarvan de specificatie vrij beschikbaar is. Zie: http://dev.tendrilinc.com/
Twitter API	De Twitter API is een aardig voorbeeld van een Web API die door heel veel appdevelopers gebruikt wordt om nieuwe apps te ontwikkelen. Zie: https://dev.twitter.com/

Bijlage I: volledige versiehistorie

Deze bijlage bevat de versiehistorie van dit document.

Versie	Beschrijving
0.96.1	Na de keuze om het standaardgedeelte van het communicatieprotocol voor P1-devices zoals ontworpen voor de Ectual in een aparte P6 Companion Standard op te nemen, is dit document ontstaan als subset van versie 0.95 van de “Specificaties IZIE Hardware en Software”, samengevoegd met een aantal specificatiewijzigingen die na versie 0.95 zijn bepaald. De standaardgedeeltes opgenomen uit de specificaties versie 0.95 zijn: ▪ Versienummers en methode versiebeheer van producten die vallen onder de P6 Companion Standard: Open API, referentie-applicaties, Simulator en SDK (zie hoofdstuk Versiebeheer) ▪ Hardwarematige eisen aan het P1-device (hoofdstuk 2) ▪ Specificaties van de configuratie-interface voor bewoners (hoofdstuk 3) ▪ Specificaties van de Open API (hoofdstuk 4) ▪ Overzicht van de inhoud van de Software Development Kit (hoofdstuk 5), de SDK is onderdeel van de P6 Companion Standard ▪ Overzicht van de functionaliteit van de referentie-applicaties (hoofdstuk 6) ▪ Overzicht van de functionaliteit van het P1-device (bijlage A) ▪ Encryptie en authenticatiebewerkingen van AES GCM (bijlage B) ▪ Gebruikte standaarden en best-practices (bijlage E) ▪ Gehanteerde definities (bijlage F) Het volgende is gewijzigd: ▪ De inleiding is herschreven (hoofdstuk 1) ▪ Doorvoeren algemene naam van het device: “P1-device” ▪ Uitbreiding van de Open API met een mogelijkheid om applicaties van buitenshuis toegang te laten hebben tot het P1-device; de Open API heeft daarmee drie delen gekregen: Open API Lokaal, Open API Externe Toegang en Open API Data Push ▪ Uitbreiding van de Open API Data Push waardoor het sturen van 10-secondedata naar externe servers robuuster is en minder resources vraagt van het P1-device ▪ Toevoeging /p1-endpoint in Open API om P1-bericht om te vragen ▪ Toevoeging van HTTP-statuscodes nodig voor de Externe Toegang ▪ Toevoeging Device Discovery via externe server als mDNS en andere lokale discoveryprotocollen niet werken (10% van de gevallen) ▪ Toevoeging van een substructuur in de JSON-berichten met energiedata, om het verschil tussen request- en responseparameters duidelijker te maken. ▪ Uitbreiding van de energiehistorie met 15 minuten aan 10-secondedata ▪ Uitbreiding van de Open API met een /accessinfo-endpoint ▪ Aanpassing van het JSON-bericht van de /status-endpoint ▪ Toevoeging procesdiagram voor detectie en data via externe server voor referentie-applicaties mobiel en Windows (bijlage C en D) ▪ Toevoeging kader in inleiding met uitleg hoe sommige woningen op hetzelfde moment elektriciteit kunnen leveren aan en gebruiken uit het net.
0.96.2	▪ Toevoeging Automatic Private IP Addressing-methode bij ontbreken DHCP-server ▪ Toevoeging time-out bij mislukte inlogpogingen om brute-force-attack te vertragen. ▪ Toevoeging gebruik juiste content-type van POST-request bij gebruik Externe Toegang
0.96.3	▪ Definitie “P1-device” toegevoegd in Inleiding en Definities ▪ Kleine aanpassingen figuur 13, Bijlage A ▪ Motivatie voor gemaakte keuzes van Externe Toegang toegevoegd aan paragraaf 4.2
1.0.0	▪ Besluit om de specificaties en Open API vast te stellen als eerste definitieve versie: daarom van 0.96 naar 1.0.

	▪ Tabel 1 aangepast naar laatste versies deelproducten ▪ H4 Open API: duidelijker onderscheid aangegeven in de tekst tussen algemene functionele beschrijvingen en technische specificaties en details ▪ /logout-endpoint toegevoegd in de Open API-specificatie ▪ HTTP-statuscodes 500 voor Open API Externe Toegang aangepast na praktijkervaringen met wegvallende verbinding ▪ Best practices en toelichting benaderen Externe Toegang toegevoegd na praktijkervaringen (zie paragraaf 4.6.2) ▪ Kleine aanpassingen in afbeelding 13 ▪ Kleine tekstuele aanpassingen
1.1.0	▪ Update van versienummers in hoofdstuk 'Versiebeheer' voor de laatste versies van de iOS en Android-apps, Windows-applicatie, Simulator en SDK. ▪ Uitbreiding statuscodes in tabel 3 van paragraaf 4.3: 200 OK "OpenAPI session freed", 503 Service Unavailable "Tunnele queue full" en 504 Gateway Timeout "Tunnel not active" ▪ Toevoeging hoofdstuk 5, SDK, met voorbeeldcode voor de Open API Data Push ▪ Nieuwe bijlage E met ideeën en wensen voor volgende versies van de P6, hernummering van volgende bijlagen.