A comparative study of mesh network protocols

SIMON MÅNSSON & FILIP NILSSON
BACHELOR´S THESIS
DEPARTMENT OF ELECTRICAL AND INFORMATION TECHNOLOGY
FACULTY OF ENGINEERING | LTH | LUND UNIVERSITY

9 juni 2017

A comparative study of mesh network protocols

Simon Månsson & Filip Nilsson

Datateknik LTH

Examensarbete handlett av Sven Robertz Gestegård från LTH och Einar Vading och Daniel Andersson från Axis Communications AB

Sammanfattning

Meshnätverk är högst relevanta och intressanta idag med tanke på alla anslutna enheter som finns. Dessa enheter syftar ofta till att ha låg strömförbrukning för att kunna drivas av mindre strömkällor som solceller eller batterier. Den hårdvara som används för att skapa meshnätverk har ofta samma målsättning, att dra så lite ström som möjligt.

Syftet med examensarbetet är att ge ökad förståelse kring meshnätverk och en del av de olika protokoll som finns tillgängliga.

Examensarbetet har undersökt och utvärderat ett antal av de meshnätverksprotokoll som finns tillgängliga utifrån följande frågeställningar. Skulle ett meshnätverk kunna utgöra en del av en lösning för att skapa sig en uppfattning om var personer befinner sig i byggnader? Och vilket protokoll lämpar sig bäst för att vidare undersöka denna möjlighet?

För att utvärdera protokollen undersöktes följande aspekter för varje protokoll, strömförbrukning, latens, räckvidd, utvecklingsmiljö och säkerhet.

I den inledande fasen av examensarbetet samlades information in kring ett antal meshnätverk för att i en förundersökning besluta kring vilka av dessa som var mest intressanta att undersöka vidare. Hårdvara med stöd för de mest intressanta protokollen köptes in för att ge möjlighet att utvärdera frågorna i problemformuleringen.

Resultatet av utvärderingen visade på att protokollen ZigBee, Thread och SmartMesh visat sig mest intressanta under förundersökningen och lämpade sig för vidare undersökning.

Nyckelord

Meshnätverk, Thread protocol, ZigBee, SmartMesh

Abstract

Mesh networks are highly relevant and interesting today considering all the connected devices that exists. These devices often aim to have low power consumption to be able to be powered by smaller power sources such as solar cells or batteries. The hardware used by mesh networks often has the same purpose, to minimize power consumption.

The purpose of this thesis is to provide an increased understanding of mesh networks and some of the various protocols that are available.

A number of the mesh network protocols available were examined and evaluated based on the following questions. Could a mesh network be part of a solution to provide an estimation of the location of people in large buildings? And which protocol is best suited for further investigation regarding the previous question?

To evaluate the protocols, the following aspects were examined for each protocol: power consumption, latency, range, development environment and security.

In the initial phase, information about a number of mesh network protocols was gathered in order to determine which ones were the most interesting to investigate further. Hardware with support for the most interesting protocols was purchased to provide the opportunity to evaluate the issues in the problem formulation.

The result of the evaluation showed that the three protocols ZigBee, Thread och SmartMesh were found to be the most interesting during the preliminary investigation are suitable for further investigation.

Keywords

Mesh network, Thread protocol, ZigBee, SmartMesh

Innehåll

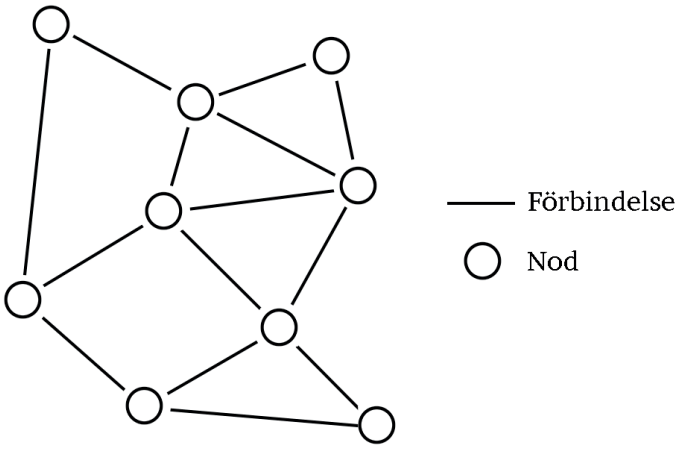
Figurer

Tabeller

1 Inledning

Meshnätverk har blivit mer aktuellt i takt med att begreppet “Internet of Things” spridit sig. Meshnätverk bygger på ett antal noder som är sammankopplade med trådlösa länkar.

Ett exempel på en tillämpning av meshnätverk skulle kunna vara ett uppkopplat hem där till exempel kylskåp, tvättmaskin, med mera förses med utrustning för att kommunicera i ett meshnätverk. Dessa produkter skulle sedan kunna skicka små meddelanden till användaren via nätverket. Inga sladdar behöver dras mellan produkterna, istället hoppar meddelandena fram genom nätverket. Figur 1 ger en bild av hur noderna i ett meshnätverk skulle kunna kommunicera.



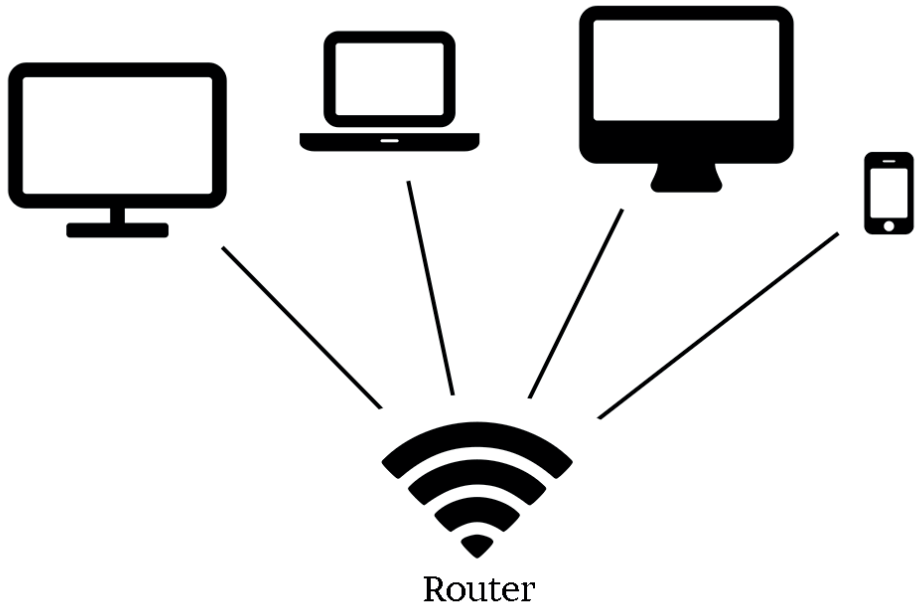
Figur 1: Exempel på struktur i mesh-baserat nätverk

Ett annat användningsområde för meshnätverk skulle kunna vara en fabrik i behov av att övervaka temperaturer runt om i anläggningen. Temperatursensorer installeras på de platser som behöver övervakas och ansluts till noder som kopplas samman till ett nätverk. Var nod läser av temperaturen periodvis och skickar iväg de avlästa meddelandena. Nodernas låga strömförbrukning och förmåga att gå i viloläge gör att de kan drivas på batteri under

långa perioder. Nätverket är inte beroende av någon annan infrastruktur för att kunna skicka sin information, dess räckvidd begränsas bara av antalet noder och avståndet mellan dem. Figur 2, sid 9 visar ett exempel på ett traditionellt WiFi-nätverk. En router eller motsvarande hårdvara försedd med antenner sprider ett trådlöst nätverk i huset. Olika slags enheter kan ansluta till detta trådlösa nätverk för att kommunicera med varandra eller ut mot internet.

Till skillnad från ett meshnätverk måste all kommunikation mellan de anslutna enheterna gå via routern vilket gör routern till en "single point of failure". Om routern förlorar sin strömförsörjning försvinner även det trådläsa nätverket vilket innebär att noderna inte kan kommunicera med varandra. Detta riskerar att göra nätverket sårbart. I industriella tillämpningar är det inte acceptabelt.

Meshnätverk minskar risken för "single point of failure" genom att nätverket är självläkande. Skulle en nod försvinna så kommer noder som tidigare skickat via den att hitta en ny väg så att deras information når fram.



Figur 2: Exempel på struktur i ett WiFi-nätverk. Alla enheter ansluts trådlöst med länkar till en central router eller accesspunkt. Nätverket är beroende av routern då all kommunikation går via den. Detta gör att routern är en "Single-point-of-failure"

1.1 Bakgrund

Axis har en mängd olika typer av nätverksanslutna enheter. Bland annat övervakningskameror, nätverksanslutna dörrstationer med mera. En stor del av systemen är Linux- och IPbaserade. Axis finner det intressant att se vad ett meshnätverk kan bidra med för ökad användarnytta. Ett sådant nätverk skulle till exempel kunna vara ett bra sätt att distribuera sensordata i systemet.

1.2 Syfte

Syftet med examensarbetet är att ge ökad förståelse kring meshnätverk och en del av de olika protokoll som finns tillgängliga. Med hjälp av informationen som tas fram kan Axis undersöka möjligheten att använda meshnätverk i någon av deras produkter. Rapporten ska även utgöra underlag för en prototyp.

1.3 Målformulering

Målsättningen med examensarbetet är att undersöka ett antal protokoll för meshnätverk. Dessa protokoll bedöms utifrån ett antal frågeställningar. Undersökningen resulterar avslutningsvis i en rekommendation kring vilka av dessa protokoll som är lämpade för vidare arbete.

En intressant tillämpning av meshnätverk skulle vara ett positioneringssystem som i exempelvis nödsituationer kan ge en uppfattning om varje personer befinner sig i större byggnader. En indikation på om någon person finns kvar i byggnaden skulle vara till värde för räddningstjänst. En lösning skulle kunna vara att använda Axis passersystem. Tyvärr är det dock inte alltid möjligt att förlita sig på data från passersystemet då en person skulle kunna öppna och släppa in/ut flera andra utan att deras passage registreras.

1.4 Problembeskrivning

Skulle ett meshnätverk kunna utgöra en del av en lösning för att skapa sig en uppfattning om var personer befinner sig i byggnader? Vilket protokoll lämpar sig bäst att vidare undersöka möjligheten?

För att besvara dessa frågor kommer nedanstående punkter undersökas.

- Strömförbrukning - Kan enstaka noder drivas på batteri under en längre tidsperiod?
- Latens - Är svarstiderna tillräckligt låga för att vara tillförlitliga?
- Räckvidd - Vilken räckvidd kan förväntas?
- Utvecklingsmiljö - Vilka krav ställs på utvecklingsmiljö?
- Säkerhet - Vilken typ av säkerhet erbjuder protokollet?

1.5 Prototyp

Den tänkta prototypen som examensarbetet ligger som underlag för har som syfte att visa vad ett meshnätverk skulle kunna tänkas användas till. I detta fall är det tänkt att utveckla en enkel prototyp vars syfte är att positionera personer i byggnader. Detta kommer göras genom att installera enheter på utvalda platser i byggnaden. Dessa enheter bygger tillsammans upp ett meshnätverk som täcker byggnaden. Utöver dessa enheter bär personer i byggnaden personliga, batteridrivna enheter, som ansluts till meshnätverket. För att positionsbestämma personer rapporterar deras individuella enheter periodvis vilken den närmaste fast installerade noden är. På så vis kan en uppfattning skapas om var i byggnader personer befinner sig.

1.6 Motivering av examensarbete

Meshnätverk är högst relevant och intressant idag med tanke på alla anslutna enheter som finns. Meshnätverk lockar med låg strömförbrukning och små enheter. Användningsområdet för meshnätverk är stort, allt från trådlösa strömbrytare till larmenheter skulle kunna tänkas ha nytta av tekniken. Examensarbetet syftar till att undersöka tekniken meshnätverk och ge en större uppfattning kring hur den fungerar.

1.7 Avgränsningar

I den fördjupade undersökningen analyserades endast tre protokoll. Det vore önskvärt vore att kunna gå in på detaljer kring hur fler av protokollen kommunicerar och fungerar, men det visade sig tidigt att protokollen fungerar olika och att det inte skulle finnas tidsutrymme för att undersöka dem i detalj. Av denna anledning kommer de tester som görs av protokollen hållas på en grundläggande nivå och resultaten skall ses som representativa för en normal kontorsmiljö. Det har inte heller utförts några mätningar i kontrollerade miljöer.

1.8 Terminologi

- **Latens** - Tiden mellan exempelvis en fråga och ett svar. Kan även kallas svarstid.
- **API** - Application programming interface.
- **DSSS** - Direct Sequence Spread Spectrum, är en typ av modulationsteknik.
- **AES** - Advanced Encryption Standard, är en symmetrisk krypteringsalgoritm.
- **IDE** - Integrated development environment, en programvaran som används för att utveckla programvaror.
- **GPIO-portar** - General-purpose input/output, ett interface för lågnivåkommunikation. Ofta på kretskort eller liknande hårdvara.
- **UART** - Universal asynchronous receiver/transmitter, krets för seriell kommunikation.
- **SEGGER J-Link** - Debug-adapter för mikrokontroller.

2 Metod

Arbetet delades in i två olika delmoment. Först utfördes en förundersökning för att ta fram tre lämpliga kandidater för det andra delmomentet, en djupare undersökning. Dessa delmoment beskrivs nedan.

2.1 Förundersökning

Arbetet inleddes med en förundersökning. Information och specifikationer för de olika protokollen samlades in. Källorna utgjordes av den officiella dokumentation för de olika protokoll som fanns att tillgå på internet. Den insamlade informationen sammanställdes för varje protokoll. Därefter bedömdes varje protokoll av rapportens författare utifrån de tre följande kriterierna.

Lätt att hitta hårdvara innebär möjligheten att få tag på den hårdvara som krävs för att kunna testa protokollet.

Lätt att hitta exempel innebär möjligheten att hitta andra projekt som använt sig av protokollet. Detta kan vara enskilda individers projekt eller företag som förevisar exempel på implementationer. Detta kan även visa på hur utbrett protokollet är.

Lätt att hitta dokumentation innebär möjligheten att hitta dokumentation som beskriver protokollet.

Dessa kriterier togs fram i samråd med handledarna på Axis. Resultaten från undersökningen användes sedan för att välja protokoll som var lämpliga att undersöka vidare.

2.2 Material

Efter förundersökningen införskaffades den hårdvara som krävdes för att genomföra tester av de olika protokollen. NXP, Digi och SmartMesh erbjuder utvecklingskort där användaren får möjlighet att använda sig av protokollen. Lista över hårdvara finns nedan.

- **Thread: NXP FRDM-KW41Z Development kit**
  Paketet innehåller två stycken noder. Dessa noder kan kopplas till en dator via dess USB port.

- **ZigBee: Digi XBee ZigBee Mesh Kit XKB2-Z7T-WTZM**
  Detta paket innehåller tre stycken noder och tre stycken Digi XBee Grove Development Board. Utvecklingskorten ger möjligheten att koppla noderna till en dator och de kan därefter konfigureras med önskade inställningar.

- **SmartMesh: DC9000B - SmartMesh IP Starter Kit**
  Detta paket innehåller fem stycken noder, ett Interface Board (IB) och en nätverksförvaltare (*eng. network manager*). Med hjälp av det IB som följer med kan man koppla en nod till en dator via USB porten som finns på IB:et. Nätverksförvaltaren kopplas via USB och ger möjligheten att kommunciera med nätverket via en dator.

Utöver detta användes oscilloskop, multimeter, spänningsaggregat och diverse kablage.

2.3 Undersökning

Ett antal parametrar som reflekterar protokollens prestanda valdes för att undersöka protokollen mer ingående. Dessa parametrar togs fram i samråd med handledarna på Axis. De parametrar som valdes är strömförbrukning, latens, räckvidd, säkerhet och krav på utvecklingsmiljö.

2.3.1 Strömförbrukning

Strömförbrukningen mättes med ett oscilloskop och noderna fick energi från batteri eller labbaggregat. Utvecklingskorten för protokollen Thread och SmartMesh erbjöd möjlighet att slå av lysdioder för att minimera den genomsnittliga förbrukningen, detta gjordes med hjälp av byglar på utvecklingskorten. Målet med mätningarna var att mäta strömförbrukningen då en liten mängd data skickades. Detta för att de olika mätningarna skulle vara någorlunda jämförbara.

SmartMesh I fallet SmartMesh anslöts ocilloskopet till SmartMeshkitets Interface Board på pinnarna markerade VSENSE.

ZigBee Inför undersökningen konfigurerades två noder. En av noderna konfigurerades så att den sov en tidsbestämd period för att sedan vakna och

då skicka en bestämd mängd data och därefter gick den direkt in i sovläge igen. Under mätningen strömförsörjdes noden med hjälp av ett spänningsaggregat inställt på 3.1V. Mellan strömkällan och noden anslöts en resistor med 12 ohms motstånd, med hjälp av oscilloskopet kunde spänningen mätas över resistorn. Eftersom motståndets resistans var känd kunde Ohms lag användas för att avläsa strömförbrukningen.

Thread På utvecklingskortet avlägsnades byglarna J8 och J20 för att minska strömförbrukningen. Utöver det togs lödpunkten SH503 bort. Noden strömförsörjdes med hjälp av ett spänningsaggregat inställt på 3V. Mellan strömkällan och noden anslöts en resistor med 12 ohms motstånd.

2.3.2 Latens

Latens mättes genom att skicka en liten mängd data till en nod och därefter vänta på att noden skickar tillbaka en bekräftelse på att datan blivit mottagen. Tiden mellan sändning och mottagning av svar mättes. Protokollen Thread och SmartMesh erbjöd möjligheten att skicka så kallade pings, den möjligheten fanns inte för protokollet ZigBee. Därför skickades en så kallad "Transmit Request" till noden. Noden svarar efter mottagande automatiskt med en bekräftelse.

Thread och ZigBee var inställda på att inte inta sovläge till skillnad från SmartMesh. Mätningar gjordes då flera noder kopplats i följd. Mätvärden för upp till tre noder i följd uppmättes och dokumenterades. Målet med dessa mätningar var att mäta svarstider för de olika protokollen i förhållande till antalet noder som meddelandet behöver skickas via.

2.3.3 Räckvidd

Tre olika test togs fram för att undersöka de olika protokollens räckvidd. För att genomföra testen behövdes två noder. En av dem anslöts till en dator. Syftet med testerna var att mäta det maximala avståndet mellan två noder innan de tappar förbindelsen.

Det första testet utfördes i en typisk kontorsmiljö. Noderna hade under testet fri sikt och en liten mängd potentiella hinder. Det andra testet involverade ett kontorslandskap på Axis. Noderna hade under testet inte fri sikt, men var inte begränsade av några kraftigare väggar. Det tredje testet genomfördes utomhus. Noderna hade under testet fri sikt.

I de tre ovanstående testerna ökades avståndet mellan noderna tills dess att de inte längre kunde kommunicera med varandra.*

2.3.4 Säkerhet

Så mycket information som möjligt samlades in för att skapa en uppfattning kring hur de olika protokollen hanterar kryptering, anslutning till nätverket med mera.

2.3.5 Utvecklingsmiljö

Delundersökningen ger en uppfattning om vilka krav som ställs på den utvecklingsmiljö som används. De olika protokollen ställer egna specifika krav på till exempel vilka programvaror som behöver installeras för att kunna kommunicera med noderna med hjälp av en dator.

3 Förundersökning

Syftet med förundesökningen var att på ett övergripande vis undersöka protokollen och deras egenskaper. Förundersökningen resulterade i att tre protokoll valdes ut för vidare undersökning.

Det är viktigt att ha i åtanke att ingen hårdvara fanns tillgänglig under förundersökningen. Först då förundersökningen avslutades köptes hårdvara in och utvärderades. Detta medförde att beslutet under förundersökningen togs baserat på teoretisk grund.

Tabell 1 listar de mest mest grundläggande specifikationerna för de olika protokollen. En redogörelse för varje protokoll och dess för- och nackdelar följer efter tabell 1. Förundersökningen avslutas med en slutsats samt en rekommendation baserat på förutsättningarna i problemformuleringen.

3.1 ZigBee

ZigBee använder sig av frekvenserna 2,4GHz eller 868MHz (Europa) för att skicka data [2]. All data skickas krypterat med 128-bitars AES och den maximala överföringshastigheten är 250Kb/s [2]. Förundersökningen visar att det finns goda möjligheter att hitta hårdvara som stödjer ZigBee.

ZigBee-protokollet använder sig av DSSS [10] för att minska risken för störningar från andra enheter som sänder i 2,4GHz ISM-bandet. ZigBee-protokollet ger noder möjlighet att gå in i sömnläge då de inte behöver skicka data, vilket minskar deras strömförbrukning. Användaren specificerar hur lång en sömncykel är [2]. ZigBee tillåter minst ett dussintal noder i ett nätverk, begränsningen ligger i mängden data som skickas och de ingående nodernas minneskapacitet [2].

Ett nätverk som använder sig av ZigBee-protokollet består av noder, dessa noder kan ha en av tre olika roller. Rollerna är coordinator, router och end device [2].

Protokoll	Frekvens	Överförings-hastighet	Open Source	Krypterings-algoritm
ZigBee	2,4GHz, 868MHz [2]	250Kb/s [2]	Nej	128-bit AES [2]
Z-Wave	868,40MHz, 869,85MHz [3]	100 Kb/s [4]	Nej	128-bit AES [4]
SmartMesh	2,4GHz [5, sida 3]	– **	Nej	128-bit AES [5, sida 2]
DigiMesh	900MHz, 2,4GHz [6, sida 4]	150Kb/s, 250Kb/s [6, sida 4]	Ja	AES* [6, sida 4]
Thread	2,4GHz [7, sida 4]	250Kb/s [7, sida 4]	Nej	AES-CCM* [8]
OpenThread	2,4GHz [7, sida 4]	250Kb/s [7, sida 4]	Nej	AES-CCM* [8]
DASH7	433MHz, 868MHz, 915MHz [9, sida 55]	167Kb/s [9, sida 55]	Ja	AES-CCM* [9, sida 57]

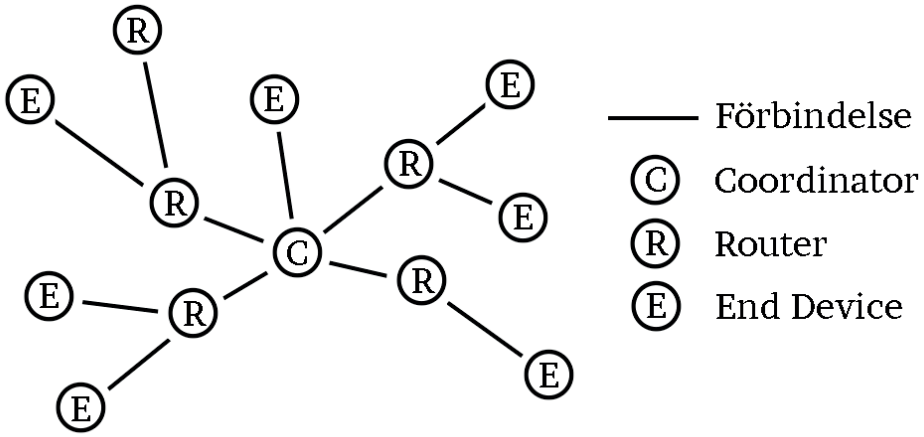
* *Nyckellängd har ej kunnat fastställas.*
** *Överföringshastighet har ej kunnat fastställas.*

Tabell 1: Översikt specifikationer av meshnätverk

- **Coordinator:** Har ansvar för att etablera nätverket [6]. Samt ansvarig för strukturen och säkerheten i nätverket [2]. Det kan endast finnas en coordinator i nätverket vid en specifik tidpunkt [6]. Coordinatorn utses manuellt vid konfigurering av noden. Beroende på nätverkets struktur skulle potentiellt coordinatorn kunna ses som en "Single-point-of-failure" [11, sida 2-3].

- **Router:** Ansvarar för att vidarebefodra data mellan andra noder [6] och utökar därmed nätverkets räckvidd [2].

- **End device:** Samlar in data eller reglerar processer. [2]. En end device



Figur 3: Översikt ZigBee-baserat nätverk

kan bara kommunicera med en "förälder" (en coordinatornod eller en routernod) [6].

3.2 Z-Wave

Till skillnad från ZigBee så krävs det en licens för att utveckla till Z-Wave [12]. Z-Wave använder sig av frekvensen 868,42MHz i Europa [4]. Detta gör att kommunikationen i meshnätverket potentiellt kommer störas mindre av exempelvis wifi och bluetooth som sänder i 2,4GHz ISM-bandet [4]. Z-Wave har en maximal överföringshastighet på 100Kb/s [4] och likt de andra protokollen använder även Z-Wave 128-bitars AES kryptering [4].

Noder av typen Z-Wave kan anta tre olika roller.

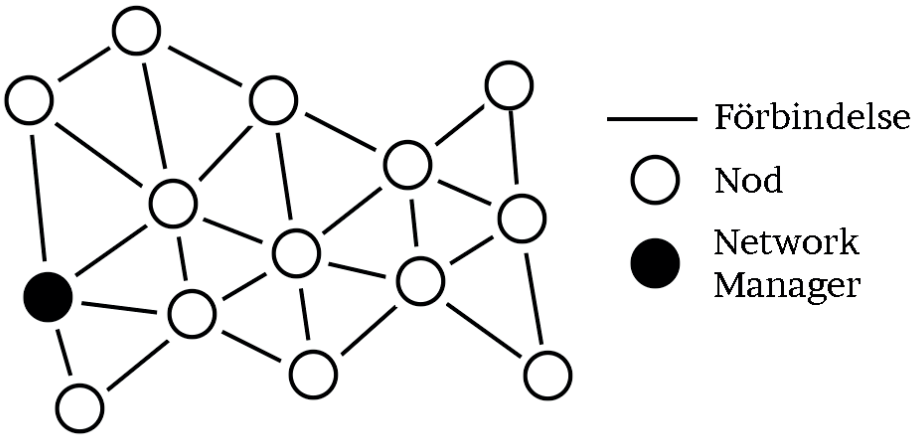
- **Controller:** Används för att styra och samordna andra noder i nätverket och ger tillåtelse att bli en del av nätverket [13, sida 16]. Controllern har också möjlighet att vidarebefodra data från noder. Det har inte framgått under förundersökningen om controllern kan anses vara en "Single-point-of-failure".

- **Slave:** Kan inte skicka data utan att ha fått en förfrågan från en annan nod. Om en nod försöker skicka ett paket men ej har direkt kontakt

med destinationsnoden kan den skicka informationen via en slavnod. [13, sida 18].

- **Routing slave:** Liknar slave men har utökad funktionalitet. Utöver funktionaliten hos en slave har en routing slave tillgång till en del av routingtabellen [14]. Den kan även initiera överföring av data till de noder i nätverket som controllern har specificerat [14].

3.3 SmartMesh



Figur 4: Översikt SmartMesh-baserat nätverk

Likt Z-Wave är SmartMesh ett proprietärt protokoll som utvecklas av Linear Technology och marknadsför sig främst mot industrin [5, sida 2]. Protokollet SmartMesh använder sig av 2,4GHz ISM-bandet för att skicka data [5, sida 3]. Den maximala överföringshastigheten har inte framgått av förundersökningen.

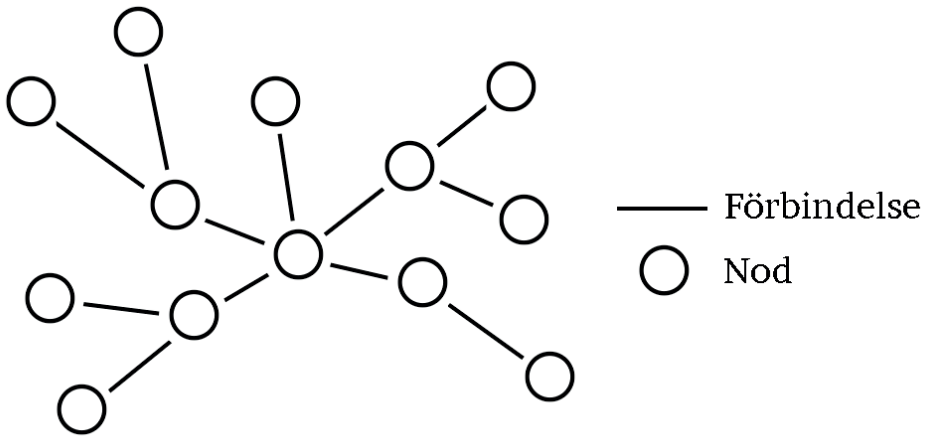
Ett SmartMeshnätverk består av en eller flera nätverksförvaltare (*eng. network manager*) och noder kallade "Motes" [15]. Nätverksförvaltaren har två uppgifter, den första är att agera acesspunkt mellan nätverket och annan programvara. Den andra är att underhålla nätverket genom diverse algoritmer [15]. Under förundersökningen har det inte framgått om nätverksförvaltaren kan anses vara en "Single-point-of-failure".

Utöver nätverksförvaltaren och Motes finns ytterligare en roll i ett Smart-Meshnätverk. Denna roll kallas “AP Mote”. Då nätverket skapas kommer nätverksförvaltaren att instruera denna nod att skicka ut så kallade “Advertisements”. De innehåller information som gör det möjligt för noder att synkronisera med nätverket och be om att få gå med. Detta är en del i den handskakning som görs för att vidhålla säkerheten i nätverket. [16, sida 3] Noderna utgör själva nätverket, de kan skicka och ta emot data från andra noder [15]. De har även möjlighet att samla in data via sensorer och skicka den vidare [15].

3.4 DigiMesh

DigiMesh är framtaget av företaget Digi som även tillverkar hårdvara. Protokollet har en maximal hastighet på 250Kb/s vid användning av frekvenser i 2,4Ghz ISM-bandet och 150Kb/s vid 900Mhz fre[6, sida 4]. Datan som skickas är krypterad med AES kryptering [6, sida 4]. Då DigiMesh är ett proprietärt protokoll stödjer det inte hårdvara från andra tillverkare [6, sida 3].

Det som får DigiMesh att sticka ut gentemot de andra protokollen är att alla noder har samma roll och kräver alltså inte att vissa noder måste vara konfigurerade som router [6, sida 3], se figur 5. Detta innebär att alla noder har möjlighet att både skicka egen data och skicka vidare data från en annan nod. Detta gör att DigiMesh inte är känsligt för “Single point of failure”[6, sida 3] eftersom nätverket självläker om en nod skulle sluta fungera. Det krävs dock att det finns andra noder i närheten som informationen kan skickas genom. Noderna har möjlighet att sova [6] för att spara ström, vilket är fördelaktigt för de noder som är batteridrivna.



Figur 5: Översikt DigiMesh-baserat nätverk

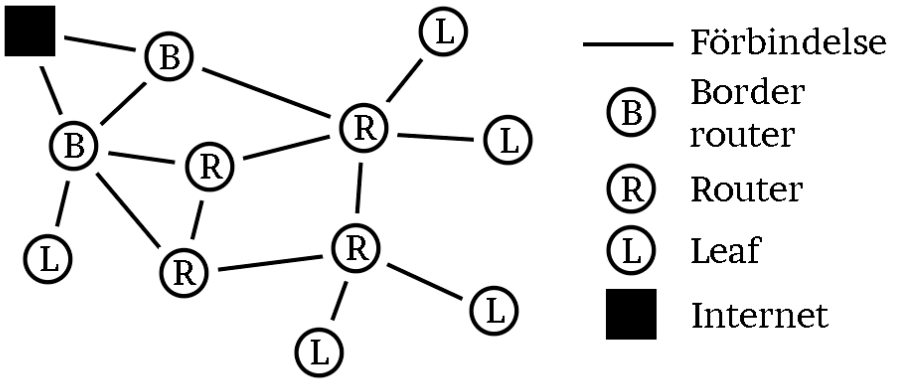
3.5 Thread

Thread Group är den organisation som står bakom utvecklingen av protokollet Thread. För att lansera en produkt med Thread behöver produkten certifieras.

Thread använder sig av 2,4GHz ISM-bandet [7, sida 4]. Thread använder sig av AES-CCM-kryptering [8], dock har det inte av förundersökningen framgått vilken nyckelstorlek som använts. Thread har en maximal överföringshastighet på 250Kb/s [7, sida 4].

Ett Threadnätverk kan hantera att en eller flera noder försvinner ur nätverket utan att kommunikationen störs [7, sida 3]. Enligt Thread Group har en nod så pass låg strömförbrukning att den kan drivas under en längre tid enbart på AA-batterier [7, sida 3].

Figur 6 visar en möjlig struktur i ett Thread-baserat nätverk. Nätverket består av tre olika typer av noder “Leaf”, “Router” och “Border Router”.



Figur 6: Översikt Thread-baserat nätverk

- **Border Router:** En specifik roll som har möjlighet att kommunicera med andra nätverk som till exempel WiFi och Ethernet [7, sida 5]. Det kan finnas en eller flera border routers i samma nätverk vilket kan säkerställa kommunikationen utåt även om någon av dem försvinner.

- **Router:** Kan ta emot data från närliggande leaf-noder och skicka det vidare mot dess destination. Alla routrar i meshnätverket har tillgång till en routingtabell som tas fram med hjälp av en algoritm kallad Routing Information Protocol (RIP). Noderna uppdaterar periodiskt denna routingtabell och sprider den vidare i nätverket [7, sida 9].

  Om en router inte har några anslutna leafnoder kan den byta läge till REED (Router-Eligable End Device)

- **Router-eligable End Device:** Dessa noder är inte tänkta att vidarebefordra data. Skulle meshnätverket förändras och behovet finnas kan dessa noder dock ändra typ till router utan användarens påverkan [7, sida 5].

- **Leaf:** Dessa noder är lämpliga att koppla sensorer eller ställdon till. För att leaf-noder ska ha möjlighet att kommunicera med varandra måste det finnas en mellanliggande router eller borderrouter, en så kallad förældernod [7, sida 9]. Om en leaf-nod skulle tappa kontakt

med vald föräldernod kommer den per automatik försöka söka upp en ny föräldernod inom räckhåll.

Leaf-noderna kan till skillnad från routers och border routers gå in i sömnläge [7, sida 9].

Nya noder i ett Threadbaserat meshnätverk ansluter alltid som antingen leaf- eller REEDnoder. Den första noden som blir router utses även till ledare [17, sida 10]. Routers eller border routers kan ta rollen som ledare i meshnätverket, vilket innebär att de kan ta beslut rörande adresser till routers och förfrågningar om att lägga till nya routers [7, sida 5]. Ledarrollen röstas fram inom nätverket. Skulle den aktuella ledaren försvinna kommer en ny ledare röstas fram per automatik. På så vis säkerställs "No single point of failure" [7, sida 5].

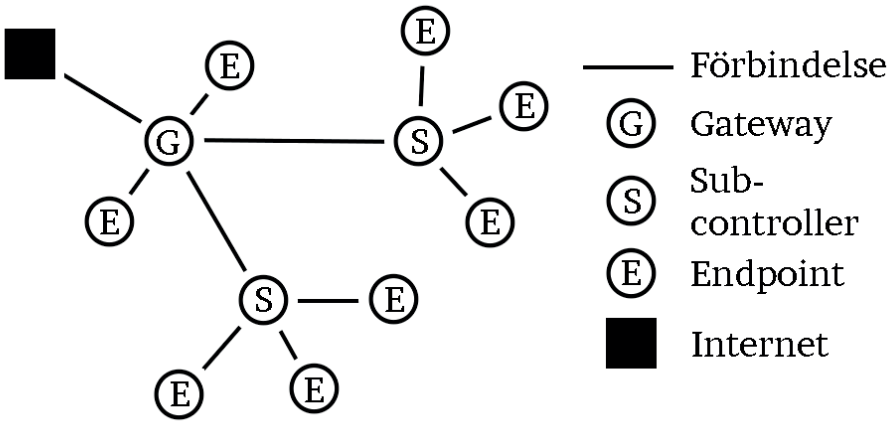
3.6 OpenThread

OpenThread har sitt ursprung i Thread och delar därför specifikationer med Thread. I dagsläget finns ingen hårdvara framtagen specifikt för OpenThread.

3.7 DASH7

DASH7 är ett öppet protokoll [18] vilket innebär att det är gratis att använda. Protokollet använder sig av 433MHz, 868MHz och 915MHz-banden [9, sida, 55] och har en maximal hastighet på 167Kb/s [9, sida, 55]. De låga frekvenserna möjliggör en bättre räckvidd och bättre penetration av hinder [19, sida 1]. Datan som skickas är krypterad med AES-CCM-kryptering [9, sida 57].

- **Gateway:** En nod med möjlighet att kommunicera med andra typer nätverk. Noden går aldrig in i sömläge [19, sida 2] och lyssnar alltid efter meddelanden om den inte själv sänder. [19, sida 2]
- **Subcontroller:** En subcontroller behöver inte alltid vara aktiv, utan vaknar periodvis och utför en sökning över sina kanaler [19, sida 2]. Noden kan skicka och ta emot data [19, sida 2].
- **Endpoint:** Dessa noder kan ta emot och skicka data och har möjlighet att gå in i sömnläge och vakna vid så kallade "wake-up events" [19, sida 2].



Figur 7: Översikt DASH7-baserat nätverk

- **Blinker:** Dessa noder tar aldrig emot utan skickar endast data. [19, sida 2]. Det finns inte med någon blinker i exemplet som syns i figur 7.

3.8 Bedömning

Baserat på den information som samlats in under förundersökning och det arbete som utförts för att samla in informationen kunde en uppfattning skapas kring följande frågeställningar: "Lätt att hitta hårdvara", "Lätt att hitta exempel" och "Lätt att hitta dokumentation". Med hjälp av uppfattningen kring frågeställningarna kunde de poängsättas utifrån fyra olika svarsalternativ. Dessa alternativ hade ett numeriskt värde för att enkelt kunna summera och ta fram ett resultat. Alternativen var "Stämmer mycket bra" med värdet 4, "Stämmer bra" med värdet 3, "Stämmer dåligt" med värdet 2 och det sista alternativet "Stämmer inte alls" med värdet 1.

Tabell 2 sammanfattar resultatet från dessa frågeställningar.

3.9 Slutsats av förundersökning

Rent teoretiskt erbjuder alla de undersökta protokollen de specifikationer som söks. Frågeställningarna "Lätt att hitta hårdvara", "Lätt att hitta exempel" och "Lätt att hitta dokumentation" i kapitel 3.8 användes för att underlätta

Protokoll	Lätt att hitta hårdvara	Lätt att hitta exempel	Lätt att hitta dokumentation	**Resultat**
ZigBee	4	4	4	**16**
Thread	2	1.5	3.5	**7**
Z-Wave	1.5	1.5	3.5	**6.5**
DASH7	1	1	2	**4**
SmartMesh	3.5	2	3.5	**9**
OpenThread	1	1.5	2.5	**5**
DigiMesh	3.5	1	3.5	**8**

Tabell 2: Författarnas uppskattade värden av frågeställningarna

urvalsprocessen. För vissa av protokollen fanns det flera exempel på projekt och det gick att hitta mycket information kring dem. Två protokoll som utmärkte sig var ZigBee och SmartMesh vilket kan ses i tabell 2.

Till skillnad från ZigBee och SmartMesh hittades få exempel på projekt där Thread eller DigiMesh användes. De bedömda värdena i tabell 2 visar på att DigiMesh skulle vara att föredra framför Thread.

Trots detta valdes Thread framför DigiMesh. Detta på grund av att strukturen på meshnätverket som Thread har ansågs intressant. DigiMesh är låst till hårdvara som levereras av tillverkaren av protokollet. Precis som Thread ansågs DigiMesh ha en intressant struktur på meshnätverket, det valdes dock bort då inga exempel på projekt kunde hittas. För andra protokoll fanns det i stor utsträckning endast teoretisk information. Inga exempel på projekt med dessa protokoll hittades. Det beslutades att inte gå vidare med något protokoll som saknade exempel, detta gäller framförallt DASH7 och OpenThread. OpenThread ersätts av Thread då skillnaden mellan dem är att OpenThread har öppen källkod. Förundersökningen visade att färdig hårdvara för Thread fanns att tillgå. Under förundersökningen hittades ytterst få exempel på projekt där Z-Wave använts. Vilket är anledningen till att det valdes bort från vidare undersökning.

Vår förundersökning kom fram till att ZigBee, Thread och SmartMesh lämpade sig bäst för vidare undersökning. Protokollet Thread är intressant då dess noder kan ändra roll i meshnätverket automatiskt. SmartMesh är ett beprövat protokoll inom industrisektorn och protokollets noder har låg strömförbrukning. ZigBee har funnits med länge och redan letat sig ut på konsumentmarknaden.

4 Resultat

Undersökningen tog fram värden för strömförbrukning, latens och räckvidd. Utöver det undersöktes och utvärderades protokollens säkerhet och deras krav på utvecklingsmiljö. Resultatet från strömförbrukningen visar på att SmartMesh är att föredra då batteridrift är aktuellt. Hårdvaran som användes av protokollet Thread hade under mätningarna en konstant hög strömförbrukning. Detta resulterade i att mätvärdena inte gick att jämföra med de värden som uppmätts för hårdvaran för de två övriga protokollen.

Mätningar utfördes för att bedöma räckvidden i olika miljöer. Resultaten visade att SmartMesh och Thread gav bäst resultat vid mätningar inomhus. I utomhusmiljö visade sig Thread prestera bättre än SmartMesh och ZigBee.

De mätningarna som gjordes med avseende på latens visade att svarstiden för SmartMesh var betydligt högre än för de andra två protokollen. Detta kan förklaras genom att noderna i SmartMesh sover i perioder till skillnad från ZigBee och Thread.

Undersökningen visade att de tre protokollen har god säkerhet vid kommunikation inom nätverket. Säkerheten skiljer sig åt mellan protokollen men ingen av deras säkerhet bedöms vara undermånlig.

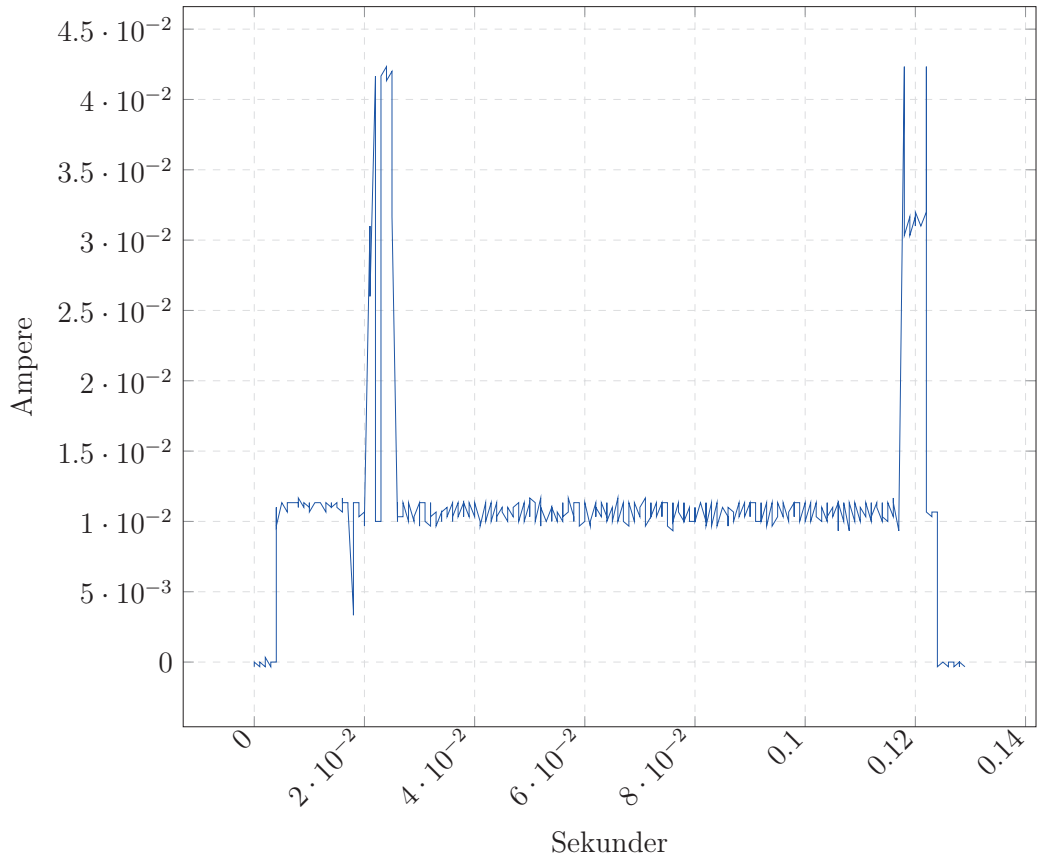
4.1 Strömförbrukning

Målet med dessa mätningar var att ge en bild av strömförbrukning då en viss uppgift utfördes av varje protokoll.

En liten mängd data skickades och var nods strömförbrukning mättes med ett oscilloskop under tiden. Resultaten från mätningarna är jämförbara då det handlar om en liknande mängd data. Tidsperioderna kan jämföras för att skapa en uppfattning kring den energi som kommer förbrukas då datan skickas.

4.1.1 ZigBee

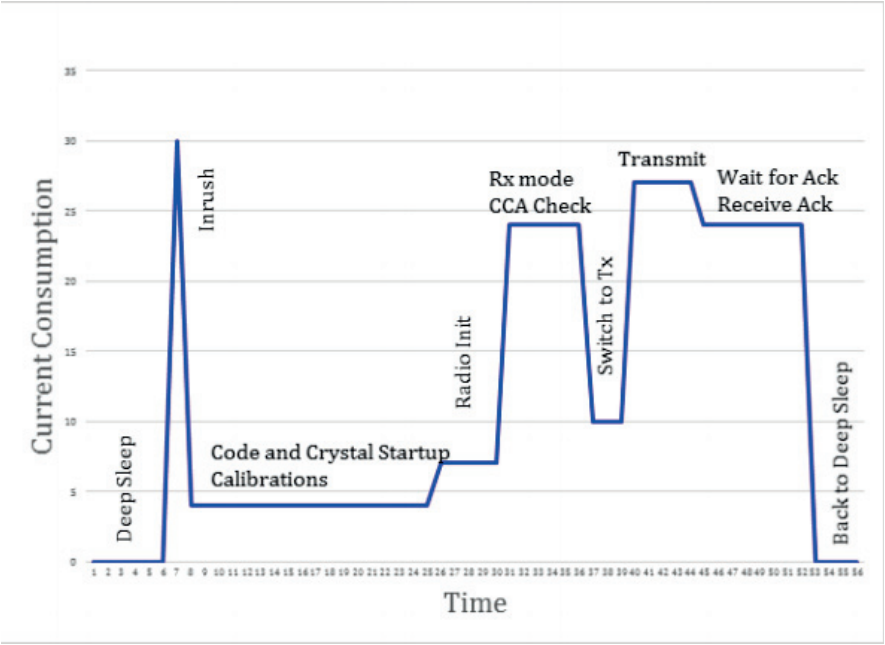
Figur 8 visar strömförbrukningen då XBee noden skickar en frame med data. Diagrammet visar hur noden vaknar upp ur sömnläge för att kort därefter slå igång radion och skicka den avsedda datan. Noden väntar sen på svar för att därefter återgå till sömnläge.



Figur 8: Strömförbrukning ZigBee

4.1.2 Thread

Resultatet av mätningarna visade att strömförbrukningen av hårdvaran var konstant hög. En följd av detta var att några variationer i strömförbrukningen inte gick att se när data skickades. En jämförelse gjordes mot figur 9 som visar en typisk sov/sändningsykel. Jämförelsen visade att de mätningar som gjorts skiljde sig så mycket åt att beslutet togs att inte använda mätningarna som resultat.



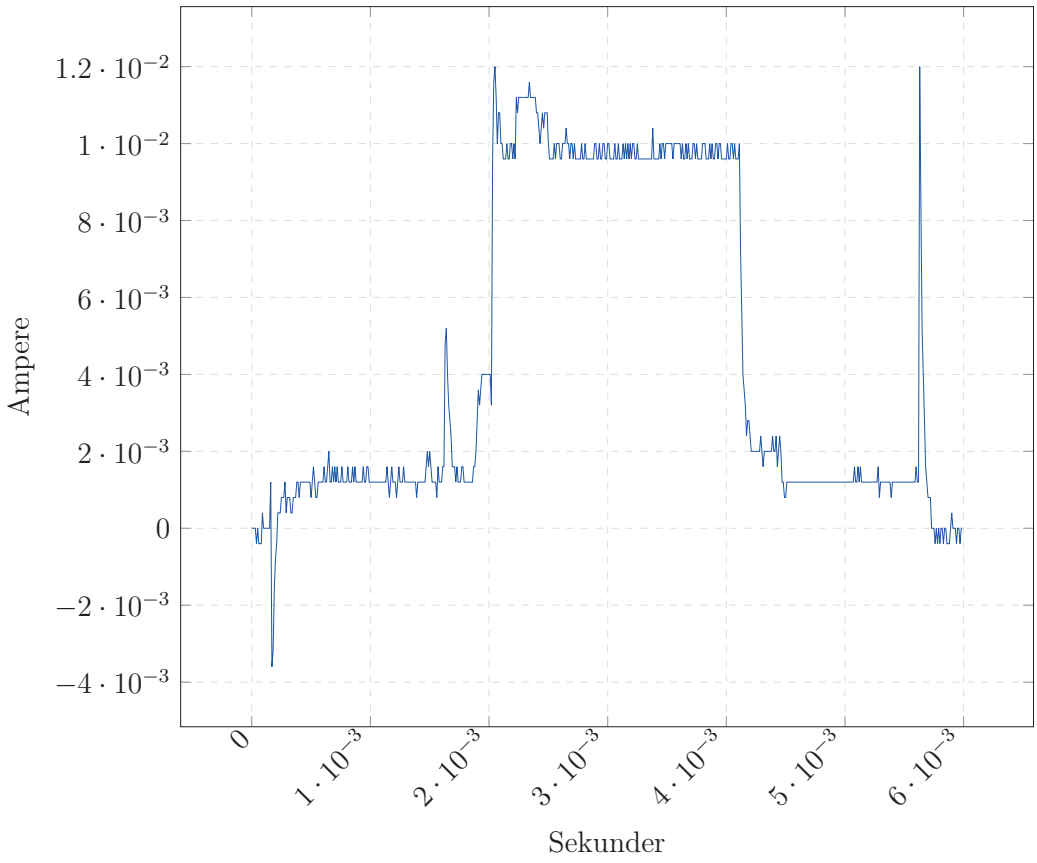
Figur 9: Exempel på en typisk sov/sändningscykel för Thread protokollet. Diagrammet förevisar de olika stegen som protokollet utför under cykeln.[1, sida 3]. Diagrammets värden är inte representativa då syftet med diagrammet är att visa de olika stegen.

4.1.3 SmartMesh

Figur 10, sid 30 visar mätvärden då en SmartMesh mote skickar data.

Det visade sig att noden skickade olika typer av data periodvis. Det var svårt att tolka vad denna data fyllde för syfte och det låg utanför examensarbetets tidsram att undersöka dessa detaljer närmre. Problemet som uppkom var att den data som användes för att testa strömförbrukningen inte gick att skilja från övrig data som noden skickade.

Diagrammet visar hur noden vaknar upp ur sömnläge för att kort därefter starta radion och skicka den avsedda datan. Noden väntar därefter på svar för att sen återgå till sömnläge.

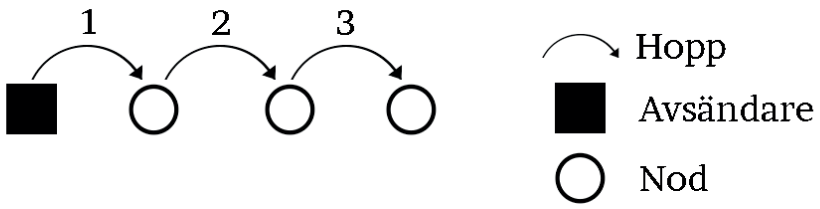


Figur 10: Strömförbrukning SmartMesh

4.2 Latens

I de tre följande tabellerna presenteras resultatet av mätningar med avseende på latens. De noder som användes var uppställda enligt topologin i figur 11, sid 31. Enligt den strukturen måste ett meddelande gå via nod ett och två om den har kontrollern som avsändare och nod 3 som mottagare. På så sätt kan man mäta svarstiden för olika scenarion.

Resultat presenteras för mätningar på upp till tre hopp från avsändare. I tabellerna presenteras 10 värden, medelvärdet av dessa samt maximum- och minimumvärde.



Figur 11: Nätverkets topologi vid latensmätning

4.2.1 ZigBee

I tabell 3, sid 32 kan man se en genomsnittlig ökning på 16% i svarstiden då man jämför resultaten av mätningarna vid ett respektive två hopp. Man kan se en genomsnittlig ökning på 48,4% i svarstiden då man jämför resultaten av mätningarna vid två respektive tre hopp. Under iteration sju då tre hopp mättes blev resultatet 216 ms. Detta bedöms vara en tillfällighet. Om värdet utesluts från medelvärdet blir det resulterande medelvärdet 62.9 ms.

4.2.2 Thread

I tabell 4, sid 32 kan man se en genomsnittlig ökning på 104% i svarstiden då man jämför resultaten av mätningarna vid ett respektive två hopp. Man kan se en genomsnittlig ökning på 15% i svarstiden då man jämför resultaten av mätningarna vid två respektive tre hopp.

4.2.3 SmartMesh

I tabell 5, sid 33 kan man se en ökning på 49,7% i svarstiden då man jämför resultaten av mätningarna vid ett respektive två hopp. Man kan se en ökning på 13,3% i svarstiden då man jämför resultaten av mätningarna vid två respektive tre hopp.

Iteration	1 hopp	2 hopp	3 hopp
1	31	46	78
2	54	69	61
3	34	48	52
4	66	67	42
5	48	39	65
6	29	52	94
7	61	45	216
8	42	34	57
9	34	61	60
10	55	66	57
Medelvärde	**45.4**	**52.7**	**78.2**
Minimum	**29**	**34**	**42**
Maximum	**66**	**69**	**216**

Tabell 3: Latensmätvärden ZigBee

Iteration	1 hopp	2 hopp	3 hopp
1	18	53	59
2	22	38	54
3	29	57	61
4	25	58	56
5	22	51	54
6	26	53	63
7	25	43	69
8	23	50	63
9	28	52	44
10	30	51	59
Medelvärde	**24.8**	**50.6**	**58.2**
Minimum	**18**	**38**	**44**
Maximum	**30**	**58**	**69**

Tabell 4: Latensmätvärden Thread

Iteration	1 hopp	2 hopp	3 hopp
1	3143	6643	6574
2	2292	5706	11635
3	5233	13092	10793
4	4305	8167	3056
5	7294	7843	10764
6	3613	4835	7350
7	3236	6337	12320
8	8366	3889	6391
9	3472	9918	7117
10	8299	7309	7546
Medelvärde	**4925.3**	**7373.9**	**8354.6**
Minimum	**2292**	**3889**	**3056**
Maximum	**8366**	**13092**	**12320**

Tabell 5: Latensmätvärden SmartMesh

4.3 Räckvidd

Resultatet av de mätningar som utfördes på protokollen avseende räckvidd presenteras i tabell 6. Tabellen innehåller resultaten från mätningar för varje protokoll i tre olika miljöer.

Scenario	ZigBee	Thread	SmartMesh
Fri sikt inomhus	37 m	>47 m	>47 m
Kontorslandskap	26 m	35 m	39 m
Utomhus	56 m	172 m	49 m

Tabell 6: Räckvidd för de olika protokollen

4.4 Utvecklingsmiljö

Med utvecklingsmiljö syftas det på nedanstående aspekter.

- IDE
- Drivrutiner
- Verktyg/program
- Dokumentation
- Öppen källkod
- Befintliga kodexempel
- I/O

Varje protokoll får en kort redogörelse kring var punkt. Syftet är att ge läsaren en uppfattning kring vilka krav som ställs för att få igång den utvecklingsmiljö som behövs för att använda och utveckla för den valda hårdvaran.

4.4.1 ZigBee

Den programvara som användes för att konfigurera varje nod heter XCTU och utvecklas av Digi. Den hårdvara som använts för att utvärdera ZigBee-protokollet tillverkades också av Digi. Programvaran XCTU stöds av följande platformar Linux, Windows och MacOS.

XCTU kan bland annat konfigurera noder, övervaka nätverket, skicka meddelanden eller data till noder och ladda upp mjukvara till noden. Den mjukvara som används på noderna är proprietär och tillhandahålls av Digi. Undersökningen har inte hittat några alternativ till den mjukvara Digi erbjuder.

Det finns diverse bibliotek bestående av öppen källkod som kan användas för att kommunicera med noderna. Undersökningen har visat att det bland annat finns bibliotek skrivna i Python [20], C/C++ [21], Java [22]. Dessa bibliotek är utvecklade av tredje part.

Om användaren vill läsa eller skicka information via en nod finns GPIO eller UART tillgängligt på noderna.

Dokumentation finns att hämta på Digis hemsida. Dokumentation beskriver hur noderna konfigureras för att exempelvis sova periodvis. Dokumentationen beskriver även hur användaren går tillväga för att konfigurera en nod så att den blir "Coordinator" eller "Router".

4.4.2 Thread

Programvaran Kinetis Design Studio användes för att uppdatera mjukvaran på hårdvaran från NXP som körde Threadprotokollet. Programvaran har stöd för följande operativsystem Linux, Windows och MacOS [23]. För att kunna kommunicera med hårdvaran krävs drivrutiner för SEGGER J-Link.

Den mjukvara som användes på noderna för att kunna använda Thread tillhandahölls av NXP. Med mjukvaran följde det även med kodexempel i form av några mindre exempelporgram.

För att interagera med nätverket behövs en terminal som har stöd för seriell kommunikation. Under arbetets gång användes terminalklienterna Terra Term och Cutecom.

NXP tillhandahåller dokumentation som rör hårdvaran. Där finns detaljer kring hur användaren uppdaterar programvaran på noderna med mera.

Den hårdvara som införskaffades för Thread har fysiska GPIO-portar och UART-portar.

4.4.3 SmartMesh

Det finns ett antal olika programvaror som används för att konfigurera och kommunicera med nätverksförvaltaren (*eng. network manager*) i ett SmartMeshnätverk. De drivrutiner som krävs finns tillgängliga för följande operativsystem Windows, Linux och MacOS.

Det finns en programvara som heter Stargazer vilken används för att konfigurera och visualisera SmartMeshnätverket. Ett användbart verktyg för att få en uppfattning kring hur noderna är ansluta till varandra. Stargazer finns endast tillgängligt för Microsoft Windows.

Nätverksförvaltaren har endast en seriell port som är dedikerad till dess API. Det innebär att endast en process kan ha en förbindelse med nätverksförvaltarens API. För att tillåta fler förbindelser finns det en programvara kallad Serial Mux. Programvaran fungerar på både Linux och Windows, dock inte på MacOS.

Utöver dessa programvaror finns ett bibliotek av Pythonfiler som underlättar integrationen av en SmartMesh produkt [24]. Detta bibliotek kallas SmartMesh SDK.

I SmartMesh SDK finns testapplikationer som visar exempel på hur noderna kan användas. Det går att köra SmartMesh SDK på Windows, Linux och MacOS då det är skrivet i Python [24]. Värt att nämna är dock att tillverkaren varnar för att några komponenter av SmartMesh SDK ej har testats för Linux och MacOS [25, sida 24-25].

På Linear Technologies hemsida finns det dokumentation för SmartMesh. Bland annat finns dokumentation kring hur protokollet fungerar och hur man går till väga för att börja använda det.

4.5 Säkerhet

De olika protokollens säkerhet undersöktes övergripande för att ge en uppfattning kring vad för säkerhet som erbjuds.

4.5.1 ZigBee

ZigBee protokollet har tre olika nycklar som används för att säkerställa säker kommunikation.

- **Link Key** Är den nyckel som delas mellan noder, så kallade "End-to-end devices". Nyckeln möjliggör säker kommunikation mellan två noder [26, sida 2].
- **Trust Center Link Key (TCLK)** Nyckeln som delas mellan "Trust Center" och en nod. Denna nyckel är en Link Key och skapas under proceduren då enheten ansluter till nätverket. Nyckeln används för att skydda meddelanden på applikationslagret och "Stack commands" [27, sida 3].
- **Network Key** Denna nyckel delas av alla enheter på nätverket och används för att skydda "Management and control communications" [27, sida 3].
- **Transport Key** Alla noder på nätverket delar en Transport Key med Trust Center. Det främsta syftet med nyckeln är att säkerställa att kommunikationen av Network Key sker säkert [27, sida 3].

Kryptoalgoritmen i ZigBee är AES och dess meddelandeintegritet säkerställs med hjälp av CBC-MAC(Cipher block chaining message authentication code) [26, sida 2].

När en nod försöker ansluta sig nätverket är det första steget att informera nätverket om noden som vill gå med. Detta görs via extern påverkan som till exempel administration via en hemsida. Det nätverket får reda på då är nodens ID och relevant säkerhetsinformation rörande noden [27, sida 3]. När nätverket mottagit den informationen går den in i "Permit joining ON state". Därefter behöver personen som installerar noden interagera med noden så att den försöker gå med i nätverket, detta görs till exempel genom att trycka på en knapp på noden. När noden sen försöker ansluta till nätverket autentiseras den och får därefter de olika nycklar den behöver. Den första som händer är att noden skickar sin TCLK till nätverkets Trust Center. Sedan kan båda parterna få tag på en Transport Key utifrån TCLK [27, sida 4]. Med hjälp av Transport Key kan nu Trust Center skicka Network Key till noden. Därefter är anslutningsproceduren avklarad och då upprättar Trust Center en ny TCLK.

4.5.2 Thread

Thread använder sig av J-PAKE (EC-JPAKE) [28, sida 8] för att kommunicera säkert.

För att en ny nod ska kunna ansluta till nätverket måste nätverket ha en "Commissioner" [28, sida 9]. En Commissioner har som roll att autentisera nya noder och är även "Authorizer". Vilket gör att den kan ge "Network Credentials" till noder som ansluter till nätverket.

En Commissioner måste inte vara en nod på nätverket utan kan vara till exempel en mobiltelefon [28, sida 2]. Detta ger användaren möjligheten att ta besluta kring vilka noder som får ansluta till nätverket. En Commissioner skulle också kunna vara en nod i nätverket som var ansvarig för att ta de ovan beskrivna besluten.

Då en nod ska ansluta till ett nätverk utför den ett "DTLS (Datagram Transport Layer Security) handshake" med nätverkets Commissioner [28, sida 23]. I detta handshake delar de nycklar med varandra så att de kan kommunciera säkert [28, sida 24]. Därefter skickar nätverkets Commissioner de nätverksparametrar noden behöver för att kunna ansluta till nätverket. Därefter avslutar de sin kommunikation och noden ansluter till nätverket [28,

sida 24].

Alla noder som vill gå med skickar sina "DTLS handshakes" över en osäker 802.15.4 radiolänk. Denna trafik är inte krypterad och skickas därmed i klartext och utan integritetskontroll. Nodens "DTLS handshake" skickas via UDP på en specifik port vilket låter nätverket skilja paketet från övrig trafik. Om inte Noden har en direkt förbindelse med nätverkets Commissioner måste deras DTLS Handshake gå via andra noder i nätverket.

Thread använder sig av en nyckel som är känd för alla noder på nätverket. Den används på MAC-lagret för att skydda nätverket från bland annat avlyssning [28, sida 8]. Detta är dock ett enkelt sorts skydd då det räcker att en nod på nätverket blir "Compromised" för att denna nyckel potentiellt kan bli känd för utomstående. Därav används ofta ytterligare former av säkerhet.

Nyckeln används för att skilja på en nod som är autentiserad och en del av nätverket och från en nod som vill gå med i nätverket. Denna nätverksnyckel skickas till en nod som nyss gått med i nätverket med hjälp av KEK (Key Encryption Key) [28, sida 8].

4.5.3 SmartMesh

Innan en nod kan gå med i ett nätverk kommer den lyssna efter "Advertisement" meddelanden, se förklaring på sida 21, från antingen en AP mote eller en nod som redan är deltagare i nätverket. Meddelandet används bland annat för att synkronisera tiden.

Efter att tiden synkroniserats skickar noden ett "Join Request" meddelande innehållandes information kring nodens strömkälla, routingmöjlighet och kända grannar. Detta är en del i "Security Handshake" som kommer etablera en krypterad kanal mellan nätverksförvaltaren (*eng. network manager*) och moten. Join Request skickas krypterat med hjälp av en "Join Key" som programmerats in hos noderna.

Managern svarar med en "Run-Time MAC Layer Authentication Key", en session till managern, en "Short Adress" som den ska använda i framtiden när den kommunicerar och en route till managern. Utöver detta får även noden fyra olika sessionsnycklar som används vid kommunikation i nätverket. Se förklaring på sida 40.

Därefter kommer noden precis som övriga noder i nätverket broadcasta Advertisements och kontinuerligt lyssna efter andra noder i närheten. [29, sida 23-24]

Det finns flera alternativ för nätverksförvaltaren att hantera en Join Request på. Dessa beskrivs nedan.

- **Common Key** Det minst strikta alternativet och det som erbjuder lägst säkerhet. Nätverksförvaltaren kommer i detta fallet acceptera alla Join Requests som är krypterade med en nyckel som är känd för hela nätverket. noderna i nätverket skeppas från fabrik med denna nyckel. Det är rekommenderat att ändra denna nyckel [29, sida 24].

- **Access Control List (ACL)** Nätverksförvaltaren har en lista över de noder som är tillåtna att anslutna till nätverket. I listan finns nodernas MAC-adress och en unik Join Key för var nod.

  Då en nod ber att få ansluta till nätverket skickar den med sin MAC-adress och en Join Request som är krypterad med dess Join key. Nätverksförvaltaren kommer kontrollera så att nodens MAC-adress finns i listan över godkända noder och därefter använda den sparade Join Key för att avkryptera Join Request. Om båda dessa steg lyckas kommer noden kunna ansluta till nätverket [29, sida 24].

- **Common Key ->ACL** Ett nyligen installerat nätverk bestående av Common Key säkerhet ändras till ACL. Detta genomförs genom att låta var nod få en unik nyckel och därefter ändra deras Join Key med ett speciellt kommando som kan skickas från nätverksförvaltaren till noden över nätverket.

  Metoden anses vara osäker i det inledande skedet men övergår till avsevärt mycket säkrare då ACL aktiverats.

  Om fler noder ska läggas till i efterhand låter man dem bli tillagda på nätverksförvaltarens ACL lista. De kan därefter ansluta direkt till nätverket [29, sida 24].

SmartMesh tillhandahåller flera lager av säkerhet, dessa beskrivs kortfattat nedan.

- I länklagret autentiseras paketen vid varje hopp med hjälp av en runtime key och en tidsbaserad räknare. Detta resulterar i att endast accepterade synkroniserade noder kan skicka paket [29, sida 23].

- End-to-end, paketen krypteras och autentiseras med hjälp av sessionsnycklar och en delad räknare. Detta resulterar i att endast den tänkta

mottagaren kan dekryptera meddelandet. Det förhindrar även återupprepningar av meddelanden, korrupta meddelanden och "Man-In-The-Middle attacks" [29, sida 23].

Var nod tilldelas fyra nycklar då den ansluter till ett nätverk:

- Sessionsnyckel som är specifik för noden som används för network management traffic
- Sessionsnyckel som är specifik för noden som används för application traffic
- Broadcast Sessionsnyckel session som används för network management
- Broadcast Sessionsnyckel session som används för application traffic

Med hjälp av dessa fyra nycklar kan nätverket säkerställa att information skickad från en nod till nätverksförvaltaren endast kan dekrypteras av just nätverksförvaltaren. Man säkerställer även att endast noden som tar mot information från nätverksförvaltaren kan dekryptera informationen. Utöver detta säkerställs även att noder som vidarebefordrar medddelanden inte kan avläsa dem.

Nätverksförvaltaren kan välja att skicka ut meddelanden på hela nätverket med broadcast-nyckel. Dessa meddelanden kan alla noder i nätverket dekryptera [29, sida 23].

5 Diskussion

De uppmätta resultaten ger en översiktlig bild av protokollens egenskaper och hårdvarans prestanda. De mätningar som utfördes visar tydligt strömförbrukningen för hårdvaran använd av protokollen ZigBee och SmartMesh då en liten mängd data skickas. Dock skulle mätningarna för strömförbrukning behöva göras i större detalj under mer kontrollerade former. Resultatet från latensmätningarna ger en god indikation för hur svarstiderna ökar i förhållande till antalet noder som informationen skickas via. Det vore önskvärt att uföra undersökningen med ett större antal noder i serie. Den räckvidd som uppmätts ger en indikation för just den hårdvaran som använts. I vissa av fallen kommer hårdvaran se annorlunda ut då en eventuell implementation sker. Detta resulterar sannolikt i andra värden för räckvidd. Resultaten för räckvidd ger en god bild av räckvidden för de olika protokollen och deras hårdvara.

5.1 Strömförbrukning

Under mätningen av strömförbrukningen upplevdes problem med att få fram värden som var representativa. De olika protokollen fungerade olika och det var problematiskt att få en uppfattning om vad protokollen utförde för operationer under mätningarna. Vi kunde tydligt se i oscilloskopet att de utförde någon slags uppgift. Men det var svårt att få reda på vilka operationer de olika protokollen utförde vid en viss tidpunkt.

Önskvärt vore att ha mer tid att undersöka strömförbrukningen genom att till exempel i detalj studera vad de olika protokollen utför sett över en tidsperiod. Alternativt mäta förbrukningen över lång tid.

Det resulterade i att protokollen vara svåra att jämföra mot varandra då man inte visste om de utförde exakt identiska uppgifter.

Försöken att sänka strömförbrukningen för Thread lyckades inte. Trots att justeringar gjorts på utvecklingskortet för att minimera överflödig funktionalitet som drog ström. De utvecklingskort som undersöktes hade en ständig strömförbrukning på ungefär 30mA. Till skillnad från Xbee ZigBee som drog nära 0mA vid viloläge. Detta kan ha medfört att en del variationer i strömförbrukning för Thread kan ha dolts på grund av strömförbrukningen som låg på 30mA. Det är möjligt att en annan konfiguration av noden hade gett annorlunda resultat. Dock fanns inte tidsutrymme att undersöka ytterligare konfigurtioner.

Problemet med SmartMesh var att det inte fanns möjlighet att se vad som skickades. Det resulterade som tidigare nämnt i att det var svårt att dra paralleller till övriga protokoll där vi med säkerhet visste att en viss operation utfördes då vi mätte.

5.2 Latens

Önskvärt vore att kunna utföra tester på en större mängd noder för varje protokoll för att se hur det påverkar mätvärdena, dock ryms det inte inom examensarbetets budget. Därför kommer ett mindre antal noder införskaffas för varje protokoll som ska undersökas.

Protokollet SmartMesh låter sina noder sova enligt ett schema. Detta syns tydligt på de mätvärden som finns i resultatet för latenstesterna. SmartMesh har avsevärt mycket högre svarstid än övriga två protokoll. Tyvärr fanns det inte utrymme för att undersöka om schemat noderna sover efter går att justera. Schemat möjliggör lång drift på batteri men är till nackdel då det gäller svarstider.

ZigBee och Thread var inte inställda på att sova vilket gjorde att deras svarstider var mer konsekventa. I ett verkligt fall hade troligtvis noderna sovit enligt ett tidsintervall vilket hade påverkat svarstiderna.

Meshnätverk av typen Thread eller ZigBee kommer vara strukturerade på liknande sätt. En nod som är ansvarig för nätverket och en eller flera routers som vidarebefordrar meddelande. Dessa två sorters noder kommer inte drivas på batteri vilket gör att de inte sover. Detta medför att de har kort svarstid, likt de testet fått fram. Den sista noden i kedjan skulle däremot kunna sova i perioder likt noderna i SmartMesh. Vilket resulterar i att svarstiden i värsta fall blir tiden för perioden som noden sover plus den tid det tar att skicka meddelandet till slutmottagaren. Testet anses representativt för de noder som agerar routrar i ZigBee- och Threadprotokollen då de aldrig sover.

De långa svarstiderna medför nackdelen att det kan ta tid innan viktiga meddelanden når sin destination. Detta skulle kunna vara ett problem för tidskritiska system. Exempel på det skulle kunna vara larm eller passagesystem.

5.3 Räckvidd

Mätningarna kan ge en indikation på vilken räckvidd de olika protokollen har i vissa miljöer. Räckvidden är knuten till den använda hårdvaran, protokollet

i sig påverkar inte räckvidden i någon större grad. Önskvärt vore att göra dessa tester i en kontrollerad miljö då många faktorer spelar in i dessa tester. Vår tanke var att få fram mätvärden som var representativa för en normal arbetsmiljö.

5.4 Utvecklingsmiljö

Det läsaren bör ha i åtanke är att de krav som ställs på utvecklingsmiljön är baserade på den hårdvara som införskaffades under inför examensarbetets undersökning. Den hårdvara som införskaffades för ZigBee, SmartMesh och Thread är utvecklingskort. Dessa är tänkta att låta användaren utvärdera och testa protokollen på ett smidigt sätt. Vilka krav som ställs på en färdig produkt som ska utvecklas i Axis miljö behandlades inte i examensarbetet.

5.5 Reflektion över etiska aspekter

Sytet med den prototyp som examensarbetet ligger som underlag för var att positionera individer exempelvis i deras arbetsmiljö. Positionering av individer är ett känsligt ämne som väcker etiska frågetecken. Risken finns att individer upplever sig övervakade då exempelvis arbetsplatsen har en uppfattning om var individen befinner sig.

Målsättningen med prototypen är att ge företaget en uppfattning om i vilken del och eventuellt våning i en byggnad som individen befinner sig på. Att långtidslagra var personer befinner sig är inte en del av målsättningen.

Den fördel som följer prototypen är framförallt vid exempelvis brand, naturkatastrofer eller liknande händelser. Då kan arbetsplatsen snabbt ta fram underlag till eventuell räddningstjänst för att ge en uppfattning om var eventuella utsatta individer kan finnas.

5.6 Framtida utvecklingsmöjligheter

Det vore värdefullt att välja ut ett specifikt protokoll och göra djupare undersökningar. Det finns flera frågeställningar som skulle behöva undersökas alternativt undersökas djupare. Listan nedan ger några exempel.

- Noggrannare mätningar av strömförbrukning över tid för att ge en uppfattning om livslängd vid batteridrift.
- Undersöka det maximala antalet noder som kan kopplas till nätverket sett till olika strukturer på nätverket.
- Strömförbrukning då nätverket innehåller en större mängd noder.
- Vilka tider kan förväntas innan en förflyttande nod hunnit ansluta till en ny föräldernod.
- Finns möjligheten att kombinera prototypen med någon annan teknik för att hitta individer som inte registrerats av meshnätverket med målsättningen att hitta eventuella obehöriga.

6 Slutsats

De frågor som ställdes i problemformuleringen var “Skulle ett meshnätverk kunna vara en potentiell lösning för att skapa en uppfattning om var personer befinner sig i byggnader?” och “Vilket protokoll lämpar sig bäst att vidare undersöka möjligheten?”. Dessa två frågor besvaras i följande texter.

Skulle ett meshnätverk kunna vara en potentiell lösning för att skapa en uppfattning om var personer befinner sig i byggnader? För att besvara detta ställdes ett antal frågor i problembeskrivningen.

Den första av dessa frågor var “Kan enstaka noder drivas på batteri under en längre tidsperiod?” Batteridrift ger möjligheten att noder kan bäras med av personer. Nodernas ringa storlek i kombination med klockbatterier ger ett portabelt positioneringssystem. Av den anledningen är det viktigt att undersöka strömförbrukningen. Resultatet visar att de tre protokollen som undersökts har möjlighet att sova vilket ger en lägre strömförbrukning. De mätvärden som presenteras under resultat indikerar att det vore möjligt att driva noder på batteri och att det kan göras under en längre tidsperiod.

Nästa fråga behandlar latens, mätningar av latensen kan ge en indikation på hur lång tid som kan passera innan det skickade meddelandet kan förväntas vara framme. Då en persons position uppdateras kommer positionsinformationen skickas via kedjan av noder. Undersökningens syfte var att se hur tiden ökade i förhållande till antalet noder i kedjan. Resultatet visar att inget av de protokoll som undersökts har tillräckligt höga svarstider för att skapa problem. Det högsta värdet som uppmättes var 13 sekunder och det skulle kunna vara tillräckligt för att positionsbestämma personer i en byggnad med tillräcklig noggrannhet.

Utöver mätningarna för latens och strömförbrukning gjordes även mätningar av räckvidd för de tre protokollen. Räckvidden för de olika protokollen mättes upp och resultatet visar att protokollen och deras hårdvara erbjuder en tillräckligt god räckvidd i inomhusmiljöer. Syftet med dessa mätningar var att skapa en uppfattning kring vilken räckvidd man kan förvänta sig av de olika noderna i de miljöer de är tänkta att användas i. Räckvidden för de olika protokollen mättes upp och resultatet visar att protokollen erbjuder en tillräckligt god räckvidd i inomhusmiljöer då det lägsta avståndet som uppmättes var 26 meter.

Resultatet från undersökningen visar att det krävs någon av följande plattformar Microsoft Windows, Linux eller MacOS. Den mjukvara som be-

hövs för att interagera med eller utveckla för protokollen finns tillgänglig antingen hos tillverkaren eller på internet. Den sista frågeställningen behandlar säkerhet. Resultatet visar att de tre protokollens säkerhet är tillräckligt god och säkerställer att positioneringsinformationen inte blir påverkad av yttre faktorer eller kan avlyssnas.

Baserat på resultatet av dessa frågeställningar dras slutsatsen att meshnätverk kan vara lämpligt för att positionsbestämma individer i byggnader.

Vilket protokoll lämpar sig bäst att vidare undersöka möjligheten? För att svara på frågeställningen jämförs de olika protokollen nedan.

Resultatet från undersökningen av strömförbrukningen visar att då krav ställs på batteridrift är SmartMesh att föredra då dess noder kan drivas på batteri oavsett roll i nätverket. ZigBee och Thread erbjuder inte denna möjligheten bortsett från deras så kallade "end devices". Detta anses dock inte vara ett problem för den tänkta prototypen då endast de noder som behöver drivas på batteri kommer att vara "end devices". Dessa noder har möjligheten att gå i sömläge vilket gör att alla tre protokoll lämpar sig för den tänkta prototypen.

De mätningar som gjordes på protokollens latens visar en markant skillnad mellan Smartmesh och de två andra protokollen. Värstafallet för SmartMesh svarstider var 13092ms. Både ZigBee och Thread värstafall var däremot under 250ms. De olika protokollens resultat varierade vad gäller det maximala avståndet mellan två noder. Det längsta uppmätta avståndet i kontorsmiljö på 39 meter presterades av SmartMesh. Det kortaste uppmätta avståndet var 26 meter och presterades av ZigBee. Resultaten från latens- och räckviddsmätningarna visar att alla tre protokoll lämpar sig för den tänkta prototypen.

Utvecklingsmiljön för de olika protokollen tillåter de vanligaste operativsystemen Microsoft Windows, Linux och MacOS. De programvaror som används finns tillgängliga för nedladdning via de olika tillverkarnas hemsidor. Resultatet visar på att det inte finns några begränsningar för den prototyp som ska tas fram oavsett vilket av de tre protokollen som används.

De tre protokollen erbjuder alla god säkerhet vid kommunikation inom nätverket. Säkerheten skiljer sig åt mellan protokollen men ingen av deras säkerhet bedöms vara undermånlig.

Det finns en mängd olika områden där meshnätverk kan vara lämpliga för användning. De olika användningsområdena ställer alla unika krav på nätverken. Baserat på de resultat som examensarbetet tagit fram kan slutsatsen

dras att alla tre protokoll är lämpliga att använda till den tänkta prototypen. Det är dock upp till läsaren att själv bedöma vilket protokoll som lämpar sig till deras eventuella användningsområde.

Sammanfattningsvis har undersökningen visat att alla tre undersökta protokoll skulle kunna vara en möjlig lösning på frågeställningen "Skulle ett meshnätverk kunna vara en potentiell lösning för att skapa en uppfattning om var personer befinner sig i byggnader?".

Av de tre protokollen kan inte direkt någon slutsats dras kring frågeställningen "Vilket protokoll lämpar sig bäst att vidare undersöka möjligheten?". Bedömningen görs att alla tre protokoll lämpas för vidare undersökning och eventuell prototyp.

7 Källkritik

De referenser som använts i examensarbetet bedöms efter deras tillförlitlighet. Referenserna har delats upp i olika kategorier för att ge bättre översikt. Referenser från den vetenskapliga litteraturen som använts anses tillförlitliga då de är publicerade verk. Utöver detta användes dokument från tillverkare av protokoll eller hårdvara som referenser. Dessa referenser anses tillförlitliga då de publicerats på respektive tillverkares hemsidor.

Ett antal webbsidor har refererats och benämns nedan som webbresurser. Dessa källor anses tillräckligt tillförlitliga då informationen hämtats från tillverkarnas hemsidor.

- Vettenskaplig litteratur ([9], [10], [11], [19], [27])
- Dokument från tillverkare ([1], [3], [6], [7], [13], [16], [17], [25], [28], [29])
- Webresurs ([2], [4], [5], [8], [12], [14], [15], [18], [26])
- Mjukvarubibliotek ([20], [21], [22], [23], [24])

Referenser

[1] Thread battery operated devices, 2017-04-22. http://threadgroup.org/Portals/0/documents/whitepapers/Thread%20Battery-Operated%20Devices%20white%20paper_v1_public.pdf.

[2] Zigbee technical summary, 2017-02-22. http://www.zigbee.org/zigbee-for-developers/network-specifications/zigbeeip/.

[3] Z-wave frequency coverage, 2017-05-09. http://z-wave.sigmadesigns.com/wp-content/uploads/2016/08/Z-Wave_Frequency_Coverage.pdf.

[4] Z-wave alliance about page, 2017-02-17. http://z-wavealliance.org/about_z-wave_technology/.

[5] Smartmesh product sheet, 2017-05-08. http://cds.linear.com/docs/en/product-selector-card/smartmesh_Rev_D.pdf.

[6] Digimesh vs zigbee white paper, 2017-02-17. https://www.digi.com/pdf/wp_zigbeevsdigimesh.pdf.

[7] Thread technical white paper, 2017-02-14. https://www.threadgroup.org/Portals/0/documents/whitepapers/Thread%20Stack%20Fundamentals_v2_public.pdf.

[8] Thread security blog post, 2017-02-14. http://threadgroup.org/news-events/blog/ID/109/PART-TWO-Securing-the-Connected-Home-From-Outside-Threats#.WKMANdxieEJ.

[9] Dash7 alliance protocol 1.0: Low-power, mid-range sensor and actuator communication. *2015 IEEE Conference on Standards for Communications and Networking (CSCN), Standards for Communications and Networking (CSCN), 2015 IEEE Conference on*, page 54, 2015.

[10] Peizhong Yi, Abiodun Iwayemi, and Chi Zhou. Developing zigbee deployment guideline under wifi interference for smart grid applications. *IEEE Transactions on Smart Grid*, page 98, 2011.

[11] Transparent coordinator failure recovery for zigbee networks. *2009 IEEE Conference on Emerging Technologies Factory Automation, Emerging Technologies Factory Automation, 2009. ETFA 2009. IEEE Conference on*, page 1, 2009.

[12] Z-wave alliance information for developers page, 2017-02-17. http://z-wavealliance.org/z-wave-oems-developers/.

[13] Z-wave device class specification, 2017-02-22. http://z-wave.sigmadesigns.com/wp-content/uploads/2016/08/SDS10242-29-Z-Wave-Device-Class-Specification.pdf.

[14] Z-wave network layer, 2017-02-22. http://wiki.zwaveeurope.com/index.php?title=Z-Wave_Network_Layer.

[15] Linear technologyy - smartmesh ip. http://www.linear.com/products/SmartMesh_IP.

[16] Smartmesh ip access point (ap) mote 2.4ghz 802.15.4e wireless ap mote. http://cds.linear.com/docs/en/datasheet/5800ipaf.pdf.

[17] Thread technical overview, 2017-02-22. https://threadgroup.org/portals/0/documents/resources/Thread_Technical_Overview.pdf.

[18] Dash alliance "why dash7" page, 2017-02-17. http://www.dash7-alliance.org/why-dash7/.

[19] Weyn Maarten, Ergeerts Glenn, Wante Luc, Vercauteren Charles, and Hellinckx Peter. Survey of the dash7 alliance protocol for 433 mhz wireless sensor communication. *International Journal of Distributed Sensor Networks, Vol 2013 (2013)*, 2013.

[20] Xbee library python. https://pypi.python.org/pypi/XBee.

[21] libxbee3 - github. https://github.com/attie/libxbee3.

[22] Xbee java library. https://www.digi.com/resources/documentation/digidocs/90001438/#reference/r_xb_java_lib_ug_intro.htm%3FTocPath%3DUser%2520Guide%7C_____0.

[23] Kinetis® design studio integrated development environment , 2017-04-25. http://www.nxp.com/products/software-and-tools/software-development-tools/kinetis-design-studio-integrated-development-environment-ide:KDS_IDE.

[24] Smartmesh sdk. https://dustcloud.atlassian.net/wiki/display/SMSDK.

[25] Smartmesh ip tools guide. http://cds.linear.com/docs/en/software-and-simulation/SmartMesh_IP_Tools_Guide.pdf.

[26] Zigbee exploited - the good, the bad and the ugly, 2017-04-25. https://www.blackhat.com/docs/us-15/materials/us-15-Zillner-ZigBee-Exploited-The-Good-The-Bad-And-The-Ugly-wp.pdf.

[27] Considerations on security in zigbee networks. *2010 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC), 2010 IEEE International Conference on*, page 58, 2010.

[28] Thread security fundamentals, 2017-04-25. https://threadgroup.org/Portals/0/documents/whitepapers/Thread%20Commissioning%20white%20paper_v2_public.pdf.

[29] Linear technology - smartmesh ip user guide, 2017-04-25. http://cds.linear.com/docs/en/user-guide/SmartMesh_IP_User_s_Guide.pdf.

Printed by Tryckeriet i E-huset, Lund 2017

LUND
UNIVERSITY

Series of Bachelor's theses
Department of Electrical and Information Technology
LU/LTH-EIT 2017-579

http://www.eit.lth.se