NanoTorrent: Plaatsbewuste peer-to-peer bestandsdistributie in draadloze sensornetwerken

Mattias Buelens, *KU Leuven*

***Samenvatting*—Draadloze sensornetwerken bestaan uit vele kleine computers uitgerust met sensoren en actuatoren die met elkaar kunnen communiceren dankzij hun draadloze antenna. Om deze netwerken operationeel te houden voor lange tijdspannes, moeten ze zich kunnen aanpassen en evolueren ook nadat ze uitgezet zijn in hun omgeving. Dit betekent dat ze grote bestanden zoals nieuwe programma's of configuraties snel en efficiënt via het netwerk moeten kunnen verkrijgen.**

NanoTorrent is een peer-to-peer protocol dat snelle bestandsdistributie toelaat in draadloze sensornetwerken. Het introduceert een hybride mechanisme om peers te vinden, door zowel met een gecentraliseerde tracker te werken en in de lokale omgeving naar buren te zoeken. Dit laat toe om zelfs in zeer heterogene netwerken te opereren, waarbij verschillende knopen verschillende programma's uitvoeren. De peer-to-peer aanpak helpt om de belasting door de bestandsdistributie te verdelen over het netwerk, en maakt gebruik van link-local multicast berichten om delen van het bestand tegelijkertijd naar meerdere buren te distribueren.

De evaluatie van het protocol toont aan dat NanoTorrent een snelle bestandsoverdracht kan realiseren in verschillende netwerkconfiguraties. De hybride aanpak om peers te vinden komt wel met een trade-off waarbij de toegenomen overdrachtssnelheid ook extra transmissies met zich meebrengt. Deze berichten moeten in meerdere hops over het netwerk om afgelegen knopen te bereiken, waardoor de tussenliggende knopen moeten helpen om deze door te sturen. Het protocol kan ook op nog verschillende plaatsen verbeterd worden om minder verkeer over het netwerk te sturen.

***Keywords*—Draadloze sensor netwerken, internet der dingen, peer-to-peer, bestandsdistributie, draadloze netwerken**

I. Inleiding

DRAADLOZE sensornetwerken zijn een opkomende vorm van computernetwerken bestaande uit vele kleine computers met beperkte mogelijkheden en capaciteiten die uitgerust zijn met allerlei sensoren en actuatoren en met elkaar draadloos kunnen communiceren. Deze computers of 'knopen' werken samen om metingen te verrichten, gegevens te verzamelen en te verwerken en acties te plannen en uit te voeren. Draadloze sensornetwerken kunnen uitgezet worden in voor mensen moeilijk bereikbare of gevaarlijke omgevingen, zoals in metrotunnels of op vulkanen [1]. Hier kunnen ze wetenschappelijke metingen doen, gevaren in de omgeving detecteren of zelf beslissen om actie te ondernemen.

Omdat deze netwerken voor lange tijd moeten meegaan en mogelijk niet meer fysiek toegankelijk zijn nadat ze zijn geïnstalleerd, moeten ze kunnen aangepast worden op afstand. Nieuwe toepassingen of nieuwe taakconfiguraties moeten kunnen verspreid worden naar alle knopen in het netwerk, en deze moeten zich zelfstandig kunnen herconfigureren met de nieuwe data. Deze herprogrammering of herconfiguratie van het netwerk moet liefst zo snel mogelijk gebeuren, zodat de knopen kunnen verder gaan met hun eigenlijke taak. Omdat de kleine computers vaak op batterijen werken, moet dit ook energie-efficiënt gebeuren met zo weinig mogelijk uitgewisselde berichten tussen knopen.

NanoTorrent is een peer-to-peer protocol dat zulke grote bestanden kan verspreiden over een draadloos sensor netwerk. Hiervoor gebruikt het een hybride mechanisme om peers te vinden, door zowel met een gecentraliseerde tracker te werken en in de lokale omgeving naar buren te zoeken. De peer-to-peer bestandsoverdracht helpt om de belasting van de gecommuniceerde berichten te verdelen over het hele netwerk.

Het vervolg van dit artikel is als volgt gestructureerd. Sectie II geeft een overzicht van de probleemcontext van internet der dingen en draadloze sensornetwerken. Sectie III bespreekt het ontwerp van het NanoTorrent protocol. Sectie IV beschouwt de implementatie van het gemaakte prototype. Sectie V evalueert het ontwerp en het prototype. Sectie VI bespreekt het onderzochte gerelateerde werk. Sectie VII eindigt het artikel met de conclusies.

II. Context

A. Internet der dingen

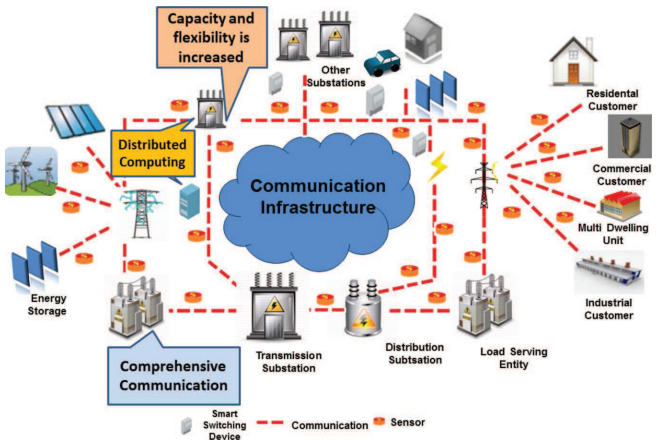
In de visie van het 'internet der dingen' worden fysieke objecten uitgerust met kleine ingebouwde computers die het mogelijk maken om deze dingen te monitoren en te besturen op elk moment van eender waar ter wereld. Dankzij IPv6 kan ieder object een uniek adres toegewezen worden dat via het internet bereikbaar is [2].

De mogelijke toepassingen van het internet der dingen beslaan bijna alle domeinen, gaande van het verbeteren van het comfort thuis tot het controleren van grote essentiële infrastructuren:

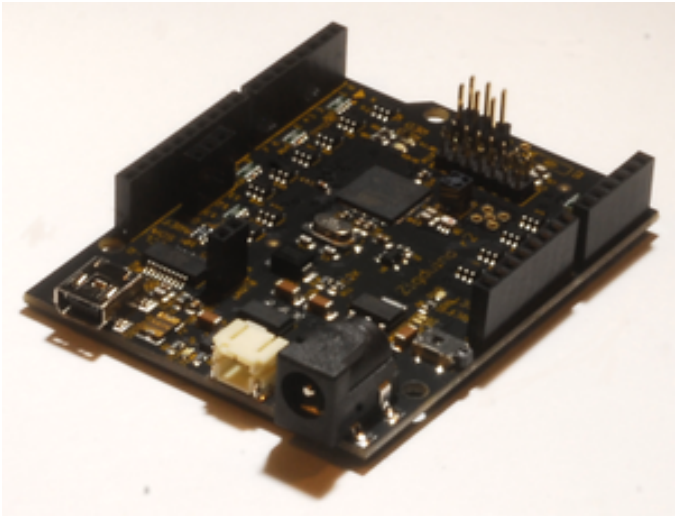
Slimme huizen. Door huishoudtoestellen, temperatuursensoren en beveiligingscamera's te verbinden met het internet, kan een huiseigenaar op ieder moment zijn leefcomfort controleren met elk toestel [3]. Met een enkel dashboard krijgt hij een overzicht van alle toestellen thuis, en kan hij bijvoorbeeld zowel de airconditioning als de rolluiken aansturen.

Slimme elektriciteitsnetten. Met de verschuiving van fossiele brandstoffen naar hernieuwbare energiebronnen wordt energieproductie steeds minder voorspelbaar. Terwijl kool- en kerncentrales elektriciteit kunnen produceren op ieder moment, hangt de productie van zonnepanelen en windmolenparken af

M. Buelens is een masterstudent aan de KU Leuven, Leuven, België.



Figuur 1. Voorbeeld van een architectuur voor een slim elektriciteitsnet. [5]



Figuur 2. Het AVR Zigduino r2 platform. [7]

van de tijd en de weercondities. Dit kan leiden tot tekorten of blackouts, of overproductie. Om te kunnen omgaan met deze fluctuaties, moeten slimme elektriciteitsnetten de mogelijkheid hebben om het energieverbruik nauwkeuriger te voorspellen op korte termijn, en om verbruikspieken te verschuiven naar momenten waarop er meer productiecapaciteit is [4]. Huizen kunnen uitgerust worden met slimme elektriciteitsmeters die continue het verbruik meten en doorsturen naar de producenten, en wasmachines kunnen geprogrammeerd worden om te werken wanneer er weinig verbruik op het net is. Dit biedt veel uitdagingen om een gesofisticeerde IT infrastructuur te ontwikkelen die deze slimme apparaten kan ondersteunen [5], zoals getoond in figuur 1.

B. *Draadloze sensornetwerken*

Vele toepassingen in het internet der dingen worden geïmplementeerd met een draadloos sensornetwerk. Deze bestaan uit een aantal kleine computers (ook wel ‘knopen’ of ‘motes’ genoemd) met sensoren, actuatoren en een draadloze antenne. De knopen meten veranderingen in hun omgeving, communiceren met elkaar en de buitenwereld en werken in op hun omgeving.

Knopen in een draadloos sensornetwerken verschillen sterk van gewone computers, en hebben verschillende ontwerpvereisten voor hun programma’s:

- Door hun kleine grootte zijn ze beperkt in hun mogelijkheden. Terwijl gewone computers multi-core gigahertz processors hebben met RAM geheugen in de orde van GB, hebben knopen slechts megahertz microprocessoren en een paar KB aan RAM. De TMote Sky heeft bijvoorbeeld een kloksnelheid van 8 MHz en 48 KB aan geheugen [6], terwijl de AVR Zigduino r2 (figuur 2) aan 16 MHz draait met 128 KB aan flash RAM [7].
- De levensduur van een knoop is beperkt door zijn batterij. Dit maakt energieverbruik een belangrijke ontwerpvereiste voor programmeurs.
- Draadloze sensor netwerken zijn onbetrouwbaar. Knopen kunnen falen doordat hun batterij opgebruikt zijn of doordat de omgeving hen fysiek vernielt (overstromingen, branden, aardbevingen,...) Programmeurs moeten rekening houden met falende knopen.
- Draadloze netwerken zijn erg gedistribueerd. Een netwerk kan bestaan uit honderden knopen, met mobiele knopen die binnen en buiten het bereik van de antenne vallen. Gedistribueerde programma’s moeten kunnen schalen naar zulke grote aantallen van knopen.

C. *BitTorrent*

Het voorgestelde protocol haalt veel inspirate van het BitTorrent protocol [8]. Dit is een peer-to-peer bestandsdistributieprotocol voor het internet, dat het bestand opsplitst in gelijke delen (‘pieces’) die afzonderlijk verspreid kunnen worden tussen peers. Gebruikers moeten eerst een `.torrent` bestand downloaden om de eigenlijke distributie van de ‘torrent’ te starten in hun BitTorrent client.

Peers melden zich aan bij een centrale tracker, die hun deelname registreert in de ‘swarm’ en een lijst van andere peers terug geeft. Peers kunnen vervolgens verbinding maken met de ontdekte peers, en ontdekken welke pieces de andere peers al hebben gedownload. Een peer kan dan een piece dat hij ontbreekt aanvragen bij een andere peer, het piece ontvangen en vervolgens zelf verspreiden naar andere peers.

BitTorrent zorgt ervoor dat peers ook moeten helpen met pieces te uploaden, en niet enkel mogen downloaden. Hiervoor gebruikt het een tit-for-tat strategie, waarbij peers die weigeren om pieces aan te bieden gestrafd worden doordat andere peers ook zullen weigeren om pieces aan te bieden aan deze slechte peer. Dit maakt dat alle peers verplicht worden te helpen bij de distributie, en zo de belasting over de verschillende peers eerlijker verdeeld wordt.

NanoTorrent neemt de concepten van torrents, trackers, swarms en pieces over in het ontwerp. Peers worden op een gelijkaardige manier gevonden via een tracker, en bestanden worden als pieces verspreid over het peer-to-peer netwerk.

III. Ontwerp

Het ontwerp van het NanoTorrent protocol bestaat uit twee grote delen: het vinden van peers om mee te verbinden, en het uitwisselen van het bestand met deze gevonden peers.

A. Vinden van peers

NanoTorrent gebruikt twee parallele mechanismen om peers te vinden.

Peers vinden met tracker. Een eerste manier bestaat erin om peers te laten communiceren met een centrale 'tracker', die bijhoudt welke peers er momenteel deelnemen aan de distributie van de torrent. Wanneer de tracker een aanvraag ontvangt van een peer, voegt hij deze peer toe aan de swarm en geeft als antwoord een aantal andere peers uit dezelfde swarm terug. De peer kan dan verbindingen beginnen maken met peers uit de ontvangen lijst. Opdat de tracker geen peers teruggeeft die al lang niet meer reageren en dus geen verbindingen zullen accepteren van peers, moeten peers periodiek hun deelname bij de tracker vernieuwen. De tracker kan dan oude peers na een tijd verwijderen, zodat deze niet meer voorkomen in de antwoorden.

Lokale peers. Een tweede manier gebruikt de nabijheid van andere peers om deze te vinden. Peers sturen periodiek een bericht naar alle buren in hun omgeving met informatie over de torrent die zij momenteel aan het downloaden zijn. Hiervoor maken zij gebruik van IPv6 link-local multicast berichten. Wanneer een peer zo'n multicast bericht ontvangt, kan hij zien van welke buur dit bericht kwam en zelf een verbinding beginnen met deze buur. Doordat berichten over een draadloos medium hoe dan ook hoorbaar moeten zijn voor alle omringende buren, kost het verzenden van zo'n multicast bericht even veel als een gewoon unicast bericht gericht naar slechts één ontvanger.

Door deze twee mechanismen samen te gebruiken, kan het protocol meer heterogene draadloze netwerken ondersteunen bestaande uit een aantal clusters van knopen. Binnen een cluster kunnen knopen elkaar vinden met lokale berichten, terwijl ze knopen uit andere clusters kunnen vinden via het antwoord van de tracker.

B. Bestandsdistributie

De peer-to-peer bestandsdistributie is erg gelijkaardig aan dat van BitTorrent. Het heeft wel enkele belangrijke aanpassingen met het oog op de beperkte capaciteiten van knopen in een draadloos sensornetwerk.

Peers proberen verbindingen op te zetten met alle peers die ze weten te vinden. Voor elke verbinding moeten ze een beetje geheugen reserveren om de toestand bij te houden. Deze toestand bestaat uit: het IPv6 adres van de andere peer, de set van pieces die beschikbaar zijn bij de andere peer, het tijdstip van het laatste bericht ontvangen van deze peer, en informatie over de huidige aanvraag voor een piece die onlangs verstuurd is naar de peer. Peers onderhouden hun verbindingen door periodiek `have` berichten te versturen, waarin ze hun huidige set van beschikbare pieces meedelen aan verbonden peers. De ontvangende peers werken dan hun beeld op de toestand van die peer bij.

Wanneer een peer merkt dat een ontbrekende piece beschikbaar is geworden bij een van zijn verbonden peers, kan deze een aanvraag versturen voor die piece. De peer verstuurt als antwoord de inhoud van die piece, zodat de ontvanger deze data kan wegschrijven en zelf de piece kan aanbieden aan zijn eigen connecties.

Als optimalizatie kunnen peers die een aanvraag voor een piece ontvangen van een lokale peer, hun antwoord met de data versturen naar *alle* buren als een multicast bericht. Dit laat lokale peers toe om data te ontvangen voor pieces die zij nog niet expliciet hadden aangevraagd, maar wel in geïntereseerd zijn. Zo kan een lokale piece uitwisseling meerdere lokale peers tegelijk bedienen.

IV. Implementatie

De systeemarchitectuur van de implementatie van het prototype is weergegeven in figuur 3. Hierin is de tracker verantwoordelijk voor het beheer van de swarm, de peer verantwoordelijk voor het vinden van andere peers en het uitwisselen van pieces, en de border router verantwoordelijk voor de verbinding tussen het draadloze sensornetwerk en het externe netwerk waar de tracker zich bevindt.

A. Tracker

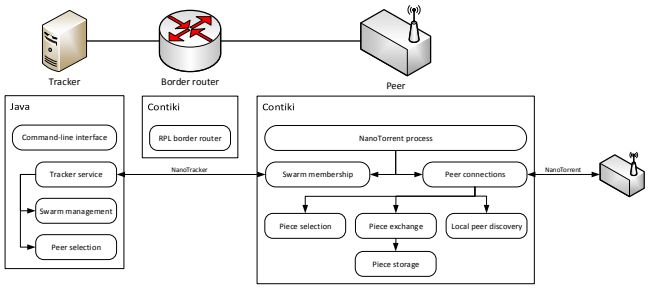
De tracker is geïmplementeerd als een stand-alone Java applicatie. Peers communiceren met de tracker over het NanoTracker UDP protocol om deel te nemen aan een swarm en peers te vinden in de swarm. De tracker onderhoudt de toestand van alle swarms en alle peers die hij opvolgt, en gebruikt deze informatie om een lijst van andere peers in de swarm aan te bieden aan nieuwe peers. In de huidige versie van het prototype kiest de tracker gewoon een aantal peers uit de swarm willekeurig, zonder een intelligent selectie algoritme. Dit kan indien nodig aangepast worden om meer informatie in rekening te houden bij deze keuze, zodat peers betere verbindingen kunnen opzetten.

B. Peer

De peer is geïmplementeerd in C als een Contiki [9] applicatie. Deze moet o.a. de verbinding met de tracker onderhouden, verbindingen met peers maken en accepteren, pieces kiezen om aan te vragen bij andere peers en op zijn beurt pieces aanbieden aan andere peers.

Om te vermijden dat peers te veel connecties tegelijkertijd zouden openen, zijn het aantal tegelijk geopende connecties beperkt. Wanneer een peer een nieuwe connectie probeert te openen, kan het dus zijn dat dit mislukt. De peer stuurt een `close` message sturen naar de andere peer om te laten weten dat hij de connectie niet kon openen, zodat deze aan zijn kant de eigen toestand ook kan opkuisen.

Peers gebruiken net zoals BitTorrent een *zeldzaamste eerst* policy om pieces te selecteren voor aanvragen. Hierbij verkiest de peer om eerst de piece aan te vragen die het minst voorkomen bij zijn verbonden peers. Dit helpt om deze zeldzame pieces sneller te verspreiden over de rest van het netwerk,



Figuur 3. NanoTorrent systeemarchitectuur

zodat ze sneller minder zeldzaam worden en de belasting op het netwerk weer beter verdeeld kan worden.

Peers zorgen er ook voor dat ze eenzelfde piece niet bij meerdere peers tegelijk aanvragen. Dit is inefficiënt, omdat de peer beter verschillende pieces in parallel kan aanvragen. Een uitzondering op deze regel is wanneer de peer bijna alle pieces heeft gedownload. In deze *end-game modus* tracht de peer zo snel mogelijk de paar ontbrekende pieces te downloaden met meerdere aanvragen bij verschillende peers, zodat de peer sneller het volledige bestand heeft en andere peers optimaal kan helpen met hun distributie.

Pieces worden opgeslagen in het Contiki File System (CFS). Afhankelijk van het platform waarvoor de Contiki applicatie wordt gecompileerd, kan dit belanden op een harde schijf, een EEPROM geheugen of in RAM geheugen. Op het AVR Zigduino platform kan CFS naar 4 KB aan EEPROM geheugen schrijven [7]. CFS maakt abstractie van deze verschillende opslagmedia, en biedt een uniforme interface om bestanden te lezen en te schrijven.

V. Evaluatie

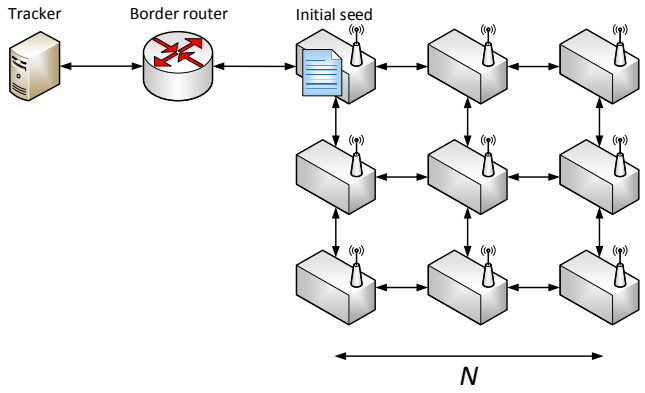
Het prototype werd geëvalueerd op zijn schaalbaarheid en op zijn bruikbaarheid in een heterogene netwerkconfiguratie door middel van de COOJA simulator.

Het doel van NanoTorrent is om bestanden te distribueren naar vele knopen in een draadloos netwerk, in een aanvaardbare tijd en met zo weinig mogelijk berichten om energie-efficiënt te zijn op fysieke knopen. Het moet bruikbaar zijn in verschillende netwerkconfiguraties bestaande uit verschillende soorten knopen met verschillende programma's, die mogelijk niet allemaal deelnemen aan de NanoTorrent distributie.

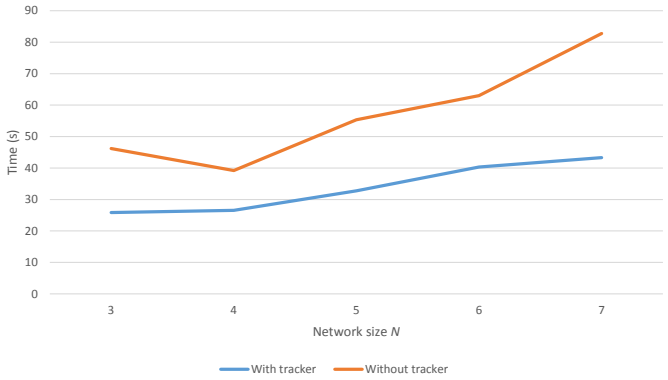
COOJA [10] maakt het mogelijk om een volledig Contiki platform te simuleren zonder wijzigingen aan de code van de geteste applicatie. In de simulator kan eender welke configuratie van knopen getest worden, met verschillende parameters voor het communicatiemedium. Tijdens de simulatie kan alle uitvoer en verzonden pakketten weergegeven en opgeslagen worden, wat zeer nuttig is om later analyses te maken.

A. Schaalbaarheid

De schaalbaarheid van het protocol werd getest door een bestand te distribueren in een netwerk met N^2 knopen in een



Figuur 4. Netwerkconfiguratie voor schaalbaarheidsexperiment. Het netwerk bestaat uit $N \times N$ knopen uitgelijnd in een raster, met één initiële seed.



Figuur 5. Tijd nodig om het volledige bestand te distribueren naar alle knopen voor een netwerkdiameter N.

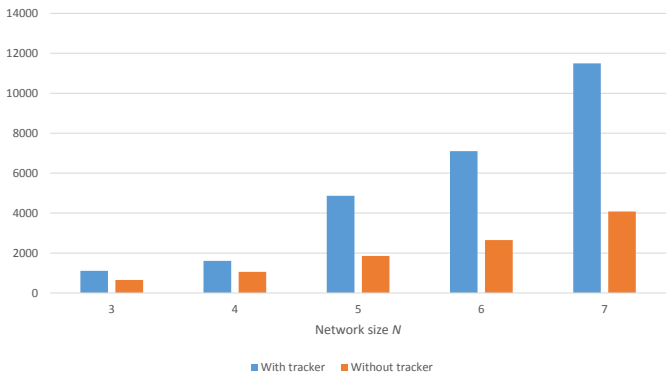
$N \times N$ rasterconfiguratie, zoals in figuur 4. De resultaten zijn getoond in figuur 5 en 6.

Het gebruik van een tracker zorgt duidelijk voor een snellere distributie, doordat peers meer verbindingen kunnen maken en zo sneller pieces kunnen aanvragen. Zonder een tracker moeten peers wachten totdat hun onmiddellijke buren pieces beginnen te ontvangen.

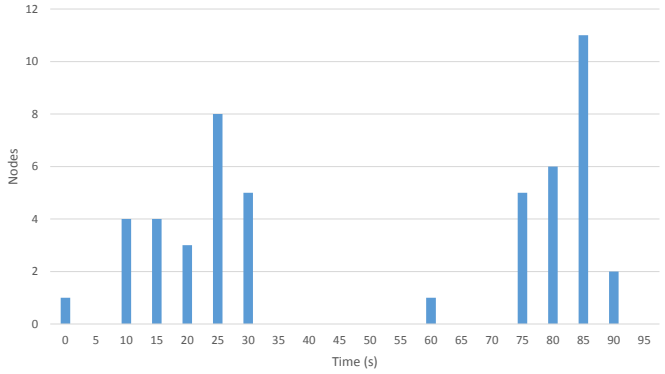
Echter, de toegenomen distributiesnelheid zorgt voor een drastische toename in het aantal verzonden berichten. Omdat peers communiceren met ver afgelegen peers ontdekte via de tracker, moeten hun berichten langs meerdere tussenliggende knopen passeren vooraleer ze hun bestemming bereiken. Deze tussenliggende knopen moeten hun eigen energie gebruiken om deze extra berichten door te sturen naar de juiste bestemming, wat zorgt voor meer verkeer op het netwerk. Wanneer de tracker niet gebruikt wordt, blijft het aantal verzonden berichten veel meer beperkt en schaalt het beter naarmate het netwerk groter wordt.

B. Heterogeniteit

Terwijl andere protocols zoals Deluge [11] vereisen dat alle knopen in het netwerk hetzelfde protocol draaien, kan Nano-



Figuur 6. Total aantal transmissies verzonden tijdens de distributie voor een toenemende netwerkdiameter N.



Figuur 7. Voltooiingstijden in netwerk met twee 5×5 clusters.

Torrent opereren in heterogene netwerken waarbij sommige knopen geen NanoTorrent implementatie uitvoeren.

Om deze toepassing te valideren, werd een experiment uitgevoerd met twee clusters van 5×5 knopen die enkel verbonden worden door een border router. De initiële seed werd geplaatst in één van de twee clusters, zodat peers in de andere cluster moeten verbindingen maken met deze cluster om het bestand te kunnen verkrijgen.

De verdeling van de voltooiingstijden van de verschillende peers is getoond in figuur 7. Hieruit blijkt dat het tot een minuut duurt vooraleer de eerste knoop in de andere cluster het volledige bestand heeft ontvangen. Zodra de eerste knoop in deze cluster de download heeft voltooid, kan het bestand wel sneller verspreiden doorheen de cluster, zodat alle knopen na in totaal anderhalve minuut het volledige bestand hebben.

C. Mogelijke verbeteringen

Er is duidelijk nog ruimte voor verbetering aan het protocol. Zeker het aantal verzonden berichten bij het gebruik van een tracker moet nog efficiënter gemaakt worden om bruikbaar te zijn in echte draadloze netwerken.

De configuratieparameters van het protocol moeten nog beter fijngesteld worden. Deze bepalen o.a. de waarden van de timers en het aantal tegelijk geopende connecties, die een grote impact kunnen hebben op de performantie. Periodieke `have` berichten kunnen initieel sneller verstuurd worden om sneller te kunnen beginnen met pieces te verspreiden. Momenteel is er ook nog weinig coördinate tussen knopen bij het versturen van aanvragen voor pieces, in vergelijking met Deluge [11] dat redundante aanvragen zoveel mogelijk tracht te vermijden.

Het is ook nog mogelijk dat peers twee verbindingen maken met dezelfde peer. Peers kunnen niet achterhalen of een peer gevonden via een tracker en via lokale multicast aankondigingen dezelfde fysieke node draaien.

D. Discussie

Uit de resultaten van het schaalbaarheidsexperiment volgt dat het gebruik van een tracker een trade-off met zich meebrengt. De distributiesnelheid kan verhoogd worden, ten koste van meer berichten over het netwerk.

NanoTorrent laat echter wel toe om in meer netwerkconfiguraties bestanden te distribueren doordat het IPv6 gebruikt als onderliggend netwerkprotocol. Het heterogeniteitsexperiment toont aan dat NanoTorrent een bestand kan distribueren tussen twee clusters gescheiden door een IPv6 border router, iets waar andere protocollen niet in slagen.

VI. Gerelateerd werk

A. Local Peer Discovery for BitTorrent

Local Peer Discovery [12] is een uitbreiding voor BitTorrent die peers toelaat om andere peers te vinden op het lokale netwerk (LAN). Hiervoor gebruikt het speciale multicast aankondigingen die niet buiten het lokale netwerk gerouteerd worden. Een peer kondigt aan welke torrent hij aan het downloaden is, zodat andere luisterende peers een verbinding kunnen openen met deze peer.

In de praktijk wordt deze uitbreiding weinig gebruikt in BitTorrent. Het is weinig waarschijnlijk dat twee gebruikers op een thuisnetwerk tegelijk dezelfde torrent zullen downloaden, en op een bedrijfsnetwerk wordt BitTorrent vaak niet toegelaten. In de context van draadloze sensornetwerken is het wel zeer relevant, en daarom is het opgenomen in aangepaste vorm in het NanoTorrent protocol.

B. Deluge

Deluge [11] is een protocol om bestanden te distribueren in een draadloos sensornetwerk.

Knopen kondigen periodiek aan welke versie van het bestand ze hebben, en luisteren naar meldingen van hun buren. Wanneer ze ontdekken dat een buur een nieuwe versie heeft, sturen ze een aanvraag voor de delen van dit bestand. Buren met delen van dit bestand kunnen dan hun data verspreiden naar geïnteresseerde buren.

Het gebruikt Trickle timers [13] om efficiënt lokale broadcasts te versturen naar buren. Iedere knoop luistert eerst een tijdje naar wat andere buren versturen alvorens zelf een bericht te sturen. Als ze merken dat verschillende buren reeds een identiek bericht hebben verstuurd, is het niet nodig dat de knoop dat bericht nog eens herhaald. De knoop kan zo besparen op zijn transmissie.

Deluge neemt dus gretig gebruik van de efficiëntie van lokale broadcasts in draadloze sensornetwerken. De ideeën kunnen ook gebruikt worden voor piece uitwisseling met lokale peers in NanoTorrent, maar vertalen moeilijk naar ver afgelegen peers gevonden via de tracker. Om het ontwerp van het prototype eenvoudig te houden, zijn de optimalisaties van Deluge voorlopig niet geïmplementeerd in NanoTorrent.

C. TinyTorrents

TinyTorrents [14] is een peer-to-peer protocol om torrents te verspreiden in een draadloos sensornetwerk. Het is sterk geïnspireerd door BitTorrent, en gebruikt ook torrent files, trackers en pieces.

De auteurs van TinyTorrents herkennen vele van de uitdagingen van draadloze sensornetwerken in de problemen die BitTorrent oplost. BitTorrent tracht de belasting op het netwerk evenwichtig te houden, verifieert de integriteit van pieces met checksums en maakt distributie snel en efficiënt met de zeldzaamste-eerste policy voor pieces. Mede daarom komen deze concepten ook terug in het ontwerp van NanoTorrent.

TinyTorrents biedt ook compatibiliteit met gewone BitTorrent door middle van een gateway die de brug spant tussen beide protocollen. Hoewel dit interessant is om sensormetingen beschikbaar te maken aan BitTorrent clients op het internet, is het niet echt nuttig om bestanden te sturen naar een draadloos sensornetwerk.

D. Uitdagingen en aanpakken voor het herprogrammeren van draadloze sensornetwerken

Wang et al. [15] schetsen een raamwerk om de verschillende functies nodig in een oplossing voor het herprogrammeren van sensornetwerken te onderzoeken. Hun framework omvat functies zoals versiebeheer, selectie van scopes, en aanvragen van code. Een protocol voor bestandsdistributie zoals NanoTorrent kan hierin de functie van code aanvragen en code distributie vervullen.

Hun paper onderzoekt ook bestaande protocollen zoals Deluge, en categoriseert ze volgens hun functies en toepasbaarheid in verschillende configuraties. Ze merken op dat weinig protocols toelaten om slechts een deel van het netwerk te herprogrammeren. Dit is één van de punten waar NanoTorrent probeert op te verbeteren, door slechts een deel van de nodes te laten deelnemen aan een torrent distributie.

VII. Conclusies

NanoTorrent verkent de mogelijkheid om twee manieren om peers te vinden te combineren. Peers kunnen lokaal buren vinden die dezelfde torrent aan het downloaden zijn, en met de tracker kunnen ze verbinden met ver afgelegen peers om pieces te verspreiden naar andere delen van het netwerk. Helaas brengt deze hybride aanpak een trade-off met zich mee: de snelheid van de distributie kan inderdaad verhoogd worden, maar dit gaat ten koste van meer berichten over het netwerk.

Doordat NanoTorrent bovenop IPv6 is geïmplementeerd, kan het communiceren over meerdere tussenliggende knopen en zelfs meerdere netwerken zonder extra inspanningen in het ontwerp van het protocol. Het protocol maakt gebruik van de bestaande routing protocollen aangeboden door alle knopen in het netwerk, in plaats van een nieuw aangepast routing protocol uit te vinden. Naarmate draadloze sensornetwerken groter worden, zal het nodig zijn dat meer protocollen op IPv6 kunnen werken om mee te kunnen schalen met de toepassingen.

Het protocol kan nog op veel plaatsen verbeterd worden, en meer evaluaties zijn nodig om ten volle de impact van de hybride aanpak om peers te vinden in kaart te brengen.

Referenties

[1] H. S. N. Lab. (2008) Volcano monitoring. http://fiji.eecs.harvard.edu/Volcano.

[2] O. Vermesan *et al.*, "Internet of Things beyond the Hype: Research, Innovation and Deployment," in *Building the Hyperconnected Society - IoT Research and Innovation Value Chains, Ecosystems and Markets*. River Publishers, 2015, ch. 3, pp. 15–118.

[3] M. Chan, D. Esteve, C. Escriba, and E. Campo, "A review of smart homes - present state and future challenges," *Computer Methods and Programs in Biomedicine*, vol. 91, no. 1, pp. 55–81, 2008.

[4] V. C. Gungor, D. Sahin, T. Kocak, S. Ergut, C. Buccella, C. Cecati, and G. P. Hancke, "A Survey on Smart Grid Potential Applications and Communication Requirements," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 28–42, 2013.

[5] ——, "Smart Grid Technologies: Communication Technologies and Standards," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 4, pp. 529–539, 2011.

[6] M. Corporation. (2006) TMote Sky: Datasheet. http://www.eecs.harvard.edu/~konrad/projects/shimmer/references/tmote-sky-datasheet.pdf.

[7] L. Electromechanical. (2013, May) Zigduino r2 Manual. http://wiki.logos-electro.com/zigduino-r2-manual.

[8] B. Cohen. (2011, Oct.) The BitTorrent Protocol Specification. http://www.bittorrent.org/beps/bep_0003.html.

[9] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," in *29th Annual IEEE Conference on Local Computer Networks*, ser. Proceedings - Conference on Local Computer Networks. LOS ALAMITOS: IEEE Computer Soc, 2004, Conference Proceedings, pp. 455–462.

[10] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with COOJA," in *31st Annual IEEE Conference on Local Computer Networks*, ser. Proceedings - Conference on Local Computer Networks. NEW YORK: IEEE, 2006, Conference Proceedings, pp. 641–648.

[11] J. W. Hui and D. Culler, "The Dynamic Behavior of a Data Dissemination Protocol for Network Programming at Scale," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, ser. SenSys '04. New York, NY, USA: ACM, 2004, pp. 81–94.

[12] Wikipedia. Local Peer Discovery. https://en.wikipedia.org/wiki/Local_Peer_Discovery. Visited on 2015-07-25.

[13] P. Levis, N. Patel, D. Culler, and S. Shenker, *Trickle: a self-regulating algorithm for code propagation and maintenance in wireless sensor networks*. USENIX Assoc., 2004, pp. 15–28.

[14] C. Mc Goldrick *et al.*, "TinyTorrents - Integrating Peer-to-Peer and Wireless Sensor Networks," *Wons 2009: Sixth International Conference on Wireless on-Demand Network Systems and Services*, pp. 109–116, 2009.

[15] Q. Wang, Y. Y. Zhu, and L. Cheng, "Reprogramming wireless sensor networks: Challenges and approaches," *IEEE Network*, vol. 20, no. 3, pp. 48–55, 2006.