UNIVERSITY OF CALIFORNIA,
IRVINE

Localize Earthquake Detection Through Low-cost Wireless Ad-Hoc Network

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

Master of Science

in Networked System

by

Phu Huu Nguyen

Dissertation Committee:
Professor Nalini Venkatasubramanian, Chair
Chancellor's Professor Nikil Dutt
Assistant Professor Marco Levorato

2015

Dedication

First, I would like to dedicate this thesis to my mother, Mrs. My Thi XuanTran, for having been giving me endlessly support and motivation to pursue higher education.

I also would like to devote this work to all my relatives, friends, co-workers and UCI professors who have offered me countless help to finish program.

Table of Contents

Page

List of Figures v
List of Tables vi
List of Algorithms vii
List of Equations viii
Acknowledgments ix
Abstract of the Dissertation x
Chapter 1: Introduction 1
Chapter 2: Motivation and Background 3
2.1. Earthquake Phenomenon and Characteristics 3
2.2. Current EEW Systems 4
2.2.1. Urgent Earthquake Detection and Alarm System (UrEDAS) 4
2.2.2. Earthquake Early Warning System for High-Speech Railways (HR-EEW) 6
2.2.3. Community Seismic Network (CSN) System 8
Chapter 3. Research Problem and Approach 10
3.1. Research Problem 10
3.2. Potential Routing Protocols for New EEW System 12
3.2.1. Flat-Routing Protocols 12
3.2.2. Location-based Routing Protocols 14
3.2.3. Hierarchical Routing Protocols 15
3.3. Issues with Integrating Current Routing Protocols for EEW Systems 17
3.4. Forest-Based Dissemination Scheme for Wireless Ad-hoc Network (FaNet) 19
3.4.1. Reasoning of Disseminating Data through a Forest-Based Network 19
3.4.2. FaNet Tree Components and Properties 20
3.4.3. How to Form a Forest-based Network (FaNet) for Data Collection and Dissemination 20
3.4.4. Forest-based Network Dissemination Issues and Solutions 23
Chapter 4: Implementation 31
4.1. SCALE Network for Earthquake Detection and Early Warning Purpose 31
4.2. SCALE Seismic Node Hardware Components 32
4.3. SCALE Client Design 34

4.4. Aggregated STA/LTA for Decentralized Detection Process 35
4.4.1. STA/LTA Algorithm 35
4.4.2. Apply STA/LTA for FaNet System to Decentralize Earthquake Detection.....36
4.5. Passive Network Polling Algorithm to Increase Detection Accuracy.....39
Chapter 5. Test Results and Analysis 40
5.1. Multicast vs. Forest-Based Data Dissemination Evaluation.....40
5.2. Experiments on Data Transmission in Wireless Ad-hoc Network.....45
Chapter 6. Conclusion and Future work.....49
6.1. Conclusion.....49
6.2. Future works.....49
Preferences 51

List of Figures

PAGE

FIGURE 1: CALIFORNIA SAN ANDREAS FAULT [3] 3
FIGURE 2: GENERAL SEISMIC WAVE PATTERNS OF EARTHQUAKE [5] 4
FIGURE 3: UREDAS P-WAVE VARIATION FUNCTION [8] 5
FIGURE 4: UREDAS EARTHQUAKE OBSERVATION UNIT [8] 5
FIGURE 5: BIRD'S EYE VIEW OF HR-EER SYSTEM [9] 7
FIGURE 6: CSN SYSTEM ARCHITECTURE [10] 8
FIGURE 7: BIRD'S EYE VIEW OF OUR HYBRID DECENTRALIZED EEW NETWORK 11
FIGURE 8: DIRECTED DIFFUSION DATA PROPAGATION PATH [11] 14
FIGURE 9: DISSEMINATION DIRECTIONS IN HYBRID DECENTRALIZED EEW SYSTEMS 18
FIGURE 10: SNAPSHOT OF A FOREST-BASED NETWORK STRUCTURE 20
FIGURE 11: M-K-ARY TREE STRUCTURE 20
FIGURE 12: SINGLE PRIMARY PARENT NODE FAILED ISSUE 23
FIGURE 13: FANET STRUCTURE IMPROVEMENT (ADAPT FARECAST APPROACH) [16] 24
FIGURE 14: LOOPING DISSEMINATION PROBLEM IN FANET 24
FIGURE 15: MULTI PARENT BACKUP DISSEMINATION PROCESS 25
FIGURE 16: PARENT NODE FAILURE IN FANET TREE 26
FIGURE 17: FANET ROOT NODES INITIATE THE DISSEMINATION 28
FIGURE 18: FANET LEAF NODES INITIATE THE DISSEMINATION 29
FIGURE 19: FANET INTERMEDIATE NODES INITIATE THE DISSEMINATION 29
FIGURE 20: DATA DISSEMINATION PROCESS IN FANET NETWORK 30
FIGURE 21: SCALE ECOSYSTEM [17] 31
FIGURE 22: RASPBERRY PI [18] 33
FIGURE 23: PHIDGETS USB ACCELERATOR SENSOR 33
FIGURE 24: DETAILED SEISMIC CLIENT DESIGN 34
FIGURE 25: STA/LTA SEISMIC WAVES ANALYSIS [20] 36
FIGURE 26: MULTICAST NETWORK TOPOLOGY 41
FIGURE 27: FOREST-BASED (FANET) NETWORK TOPOLOGY 42
FIGURE 28: TOTAL DISSEMINATION TRAFFIC BETWEEN MULTICAST AND FANET 43
FIGURE 29: DELIVERY CAPACITY OF FANET V.S. MULTICAST DISSEMINATION SCHEME 43
FIGURE 30: DUPLICATED TRAFFIC OF FANET V.S. MULTICAST DISSEMINATION SCHEME 44
FIGURE 31: DISSEMINATING POWER CONSUMPTION OF FANET VS. MULTICAST DISSEMINATION SCHEME 45
FIGURE 32: EXPERIMENTAL EEW MESH NETWORK SETTING 46
FIGURE 33: LIVE SEISMIC DATA FROM EXPERIMENTAL EEW MESH NETWORK WITHOUT INTERNET INTERRUPTION 47
FIGURE 34: LIVE SEISMIC PICKS FROM EXPERIMENTAL EEW MESH NETWORK WITH INTERNET INTERRUPTION 48

List of Tables

	PAGE
TABLE 1: NETWORK SEISMIC PICKS COLLECTION MATRIX	38
TABLE 2: NODE EARTHQUAKE POLLING TABLE	39

List of Algorithms

	PAGE
ALGORITHM 1: ACCEPTING NEW CHILDREN LOGIC FOR FANET NODES	21
ALGORITHM 2: GREEDY PARENTS SELECTION	22
ALGORITHM 1: JOINING TREE LOGIC FOR FANET NODES	22

List of Equations

PAGE

EQUATION 1: SHORT TERM AVERAGE CHANGE SEISMIC PICKS OF PEER NODES 37
EQUATION 2: REFACTORED SHORT TERM AVERAGE SEISMIC PICKS CHANGE OF PEER NODES 37
EQUATION 3: SHORT TERM AVARAGE SEISMIC PICKS OF PEER NODES SIMILARS TO THAT OF EACH NODE 37
EQUATION 4: SIMPLIFIED SHORT TIME AVERAGE SEISMIC PICK CHANGE OF PEER NODES 38
EQUATION 5: SIMPLIFIED LONG TERM AVERAGE SEISMIC PICK CHANGE OF PEER NODES 38

Acknowledgments

It was my great honor to have the opportunities to work with Dr. **Venkatasubramanian** and her research team under the SCALE project during the time I study for my master program in Networked System at University of California, Irvine (UCI). Dr. **Venkatasubramanian** has given me tremendous motivations and inspirations to challenge myself to advance my research in the areas that I did not feel familiar and comfortable with. Her patience, detailed instructions, insightful comments and deep knowledge in the network embedded system and middleware play important factors in the success of this thesis. I have learned a lot from Dr. **Venkatasubramanian** not only about the networked system, especially in wireless sensor network and the internet of thing, but also her attitude and professional manner toward research and her commitment in applying her specialty and experience on serving the community.

I also would like to express my great gratitude toward Dr. **Levorato** for giving great support and advices which helped me get back to the right direct for this thesis. His lectures on wireless network topics and Markov Chain process have enlightened my understanding about the wireless communication area. However, the most valuable thing I have learn from him is his philosophy in studying and researching. He has taught me that it is worth to conduct a research project in the area that is new to me and I have fewer chances to succeed rather than selecting something that I am familiar with and can easily achieve. I could be failed when going to a new study field but I can learn a lot of new things while doing it. I am still applying this principle in my academic activities as well as my professional career.

Lastly, I want to thank Dr. **Dutt** for accepting to be in my dissertation thesis committee, even though I had to reschedule my dissertation day many times. I cannot thank you enough for his support and flexibility. I did not have an opportunity to study with Dr. **Dutt** during the time I am at UCI. Dr. **Venkatasubramanian** suggested me to ask Dr. **Dutt** to be a member of my thesis committee. I got his acceptance within the first day I emailed him. He also recommended me to use Doodle online event scheduling software to pole my final defense day, which I found extremely useful.

Many of my fellow graduate students offered great help during my study and research at UCI School of Computer Science and Engineering. I would like to thank **Kyle Benson** for helping me with many issues regarding to the SCALE and CSN clients as well as his advices in many other topics. Last but not least, I would like to give special thanks to **Qiuxi Zhu** and **Guoxi Wang** for sharing their experiences in Python programming, Raspberry pi and sensor configuration. You all have given me a great life experience of being a graduate student at UCI campus.

Finally I would like to thank my mom for all that she has done for me. Her sacrifice and restlessly financial and spiritual supports have eased the path to pursue my higher education.

Abstract of the Dissertation

Localize Earthquake Detection Through Low-cost Wireless Ad-Hoc Network

By

Phu Huu Nguyen

Master of Science in Networked System

University of California, Irvine, 2015

Professor Nalini Venkatasubramanian, Chair

Seismic activities can cause significant damages to infrastructures and human lives. An early earthquake alert system (EWW) could help save our lives and protect properties. Constructing a reliable and affordable earthquake detection and alert system is a challenging job. Seismic activities do not have any specific statistical patterns that we can observe and study over the time but we only have a couple seconds to detect and trigger alert. Traditionally, earthquake can be detected by either one big seismic station or a set of intermediate size seismic observation units, which are often costly and hard to scale. Researchers in the seismic community have presented new EEW system that can detect earthquake with a lower budget by using crowdsources. However, the performance of the system is relied on the internet of its deployment area, which could be interrupted in big earthquakes. In this paper, we are proposing a hybrid decentralized EEW architecture in which the earthquake detection process is taken off the cloud. Seismic activities are observed and analyzed locally by each seismic node to reduce the impact of infrastructure failure during the earthquake. The capacity of seismic node is empowered by a forest-based network and forest-based dissemination scheme. The new network structure allows low quality electric devices to collaborate with each other to accomplish their earthquake detection and early alert mission.

Chapter 1: Introduction

Nepal earthquake in April 2015, which killed more 9,000 people and injured more than 23,000 [1], shows that seismic activities could cause significant impacts on the community where EEW systems are not available. The number of the fatalities and injuries would have been less if the residents of the Nepal's affected region were alerted before the big strike arrived. Nepal does not have many earthquakes like Japan or California in the USA. Therefore, its government has less motivation to build and manage an earthquake detection system to reduce the potential destructions. EEW systems have been built and deployed in many countries, such as United State, Japan, Mexico, and China. Most of the current systems are implemented with the centralized approach where high power computer servers are used to collect seismic data from multiple observing stations within the monitoring regions to detect the earthquake. The cost of building and maintaining the system discourages some governments or communities in the earthquake zones to have it in place. According to the survey of U.S. geological science report, the capital investment for a West Coast EEW system is about $38.3 million plus an additional annual maintenance and operations of $16.1 million [2]. This spending is a burden for the nations with a small gross domestic product (GDP) or low income communities.

The question is whether we can apply the computer technologies to lower the building and maintaining cost of the EEW systems and make them more affordable for every community while guaranteeing their reliability and accuracy in detecting and alerting earthquake ability.

Due to its cost, traditional EEW systems are often only implemented to detect the earthquake at critical places, such as railroad lines or big commercial buildings. The system contains either one super seismic station or a set of intermediate quake detector units. The seismic station or seismic boxes often include an accelerometer, a processing unit, and an alarm unit. The processing unit collects and analyzes the seismic data from the accelerometer to identify the arrival of any quakes within its monitoring region. When the observation stations recognize a potential earthquake, they send alerts to an earthquake control center where appropriate actions are taken place, including slowing down the trains and broadcasting the earthquake alerts over different media channels to notify residents. This earthquake detecting and warning model has been proved effective. However, it is expensive and difficult to expand it for a large region.

The internet provides a mean of communication that we can utilize to develop the low-cost earthquake detection systems. It allows the end users to integrate with the application to enhance its content and performance. Community Seismic Network (CSN) is one of the earthquake warning systems that take advantage of the internet crowdsourcing concept to detect the earthquakes. People can volunteer to participate in the network to detect the potential strikes in their areas. Volunteers' personal computers or smartphones are turned into an seismic sensor node by installing and running CSN client. The CSN nodes sense and send the seismic picks to remote CSN servers where the earthquake detection is conducted. CSN is one of the unconventional EEW systems which attempt to identify the seismic activities through the public contribution over the internet. Although it is a centralized architecture system, which has some issues with the reliability and scalability in earthquake detection application domain.

The internet of things (IoT) and open source software ideology provide us even more options to build efficient networked systems that can handle a specific task with a constrained budget. Low-cost electronic devices can be programed to collaborate with each other over their own network to accomplish their assigned jobs. Therefore, it is practical to develop a decentralized EEW system that is fueled and managed by the members in the community it servers.

The thesis attempts to propose a platform in which residents of the earthquake sensitive regions can establish and run a decentralized EEW system by themselves to increase the safety for their community. Our main objective is to come up with a clever way to build a resilience wireless seismic sensor mesh network which can detect potential seismic activities in a specific target region using low-cost electrical devices. In order to achieve the goal, we have designed and integrated the following algorithms:

- Forest-based Accelerated Dissemination Scheme for Wireless Ad-hoc Sensor Network.
- Network Adaptive STL/LTA algorithm to ensure the accuracy of earthquake detection.
- Consensus Network voting algorithm to enhance the certainty of peer node's decision.

The rest of the paper will be organized as following: Chapter 2 will discuss about earthquake, its impacts on our society and how it is currently detected. Chapter 3 will talk about challenges in detecting earthquake and giving early warning and our approach to resolve it. Chapter 4 show how to implement our proposal to construct an affordable and reliable EEW system. Chapter 5 is dedicated for evaluation and analysis of our new techniques in localizing earthquake detection process. And lastly, chapter 6 will conclude our study and indicate some future works to improve our proposal.

Chapter 2: Motivation and Background

2.1. Earthquake Phenomenon and Characteristics

Earthquake is one of the unavoidable natural disasters that human society has to deal with. It is the recognizable shakings of the earth surface, which can be strong enough to destroy the man-made structures such as bridges, cruise ships, buildings and houses. Destructive earthquakes can take away lives of thousands of people and put millions of others into catastrophes. Earthquake is a natural mechanism to help the earth to reconstruct its fault planes so that they can get to a more stable state. These activities release the frictional force to the earth surface which could crack the rock formation and generate the violent ground shakings. Earthquakes with magnitude 3.0 or lower are imperceptible and considered weak shakes. Magnitude 8.0 or above could cause serious damages for a large area.

Figure 1: California San Andreas Fault [3]

Earthquakes occur every year around the world. The current earthquake detection instruments and system report around 500,000 seismic shakes with different magnitudes per year. Among of these, about 100,000 can be felt by human beings [4]. Some regions have more earthquake activities than the others. California and Alaska in the U.S., as well as in El Salvador, México, Guatemala, Chile, Peru, Indonesia, Iran, Pakistan, the Azores in Portugal, Turkey, New Zealand, Greece, Italy, India, Nepal and Japan are those places with high risk of having a big earthquake with magnitude of 7.0 or more.

An earthquake creates P-waves and S-waves. The P-waves are primary waves that can travel through liquids and solids with a speed of 5 to 7 kilometer per second [13]. They are less destructive than the S-waves, which arrives later. Most of current EEW systems attempt to capture the P-waves and send out the alerts before the S-waves arrive. If the epicenter is far away from the protected area, the time interval between the detection of the P-waves and the arrival of the S-waves will be sufficient enough for us to shut down or stop critical operations such as nuclear plant, moving trains and elevators or gas pipelines.

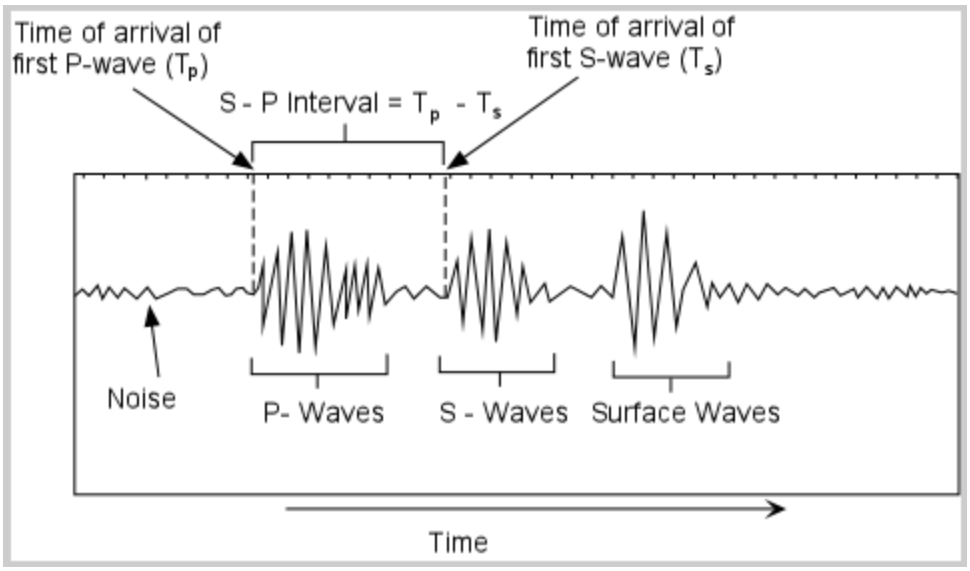


Figure 2: General Seismic Wave Patterns of Earthquake [5]

2.2. Current EEW Systems

2.2.1. Urgent Earthquake Detection and Alarm System (UrEDAS)

UrEDAS is an EEW system that was developed by System and Data Research Co. Ltd in Japan in 1984. It was the only EEW system that provided the real time P-wave alarm at that time [6, 7]. Since then, UrEDAS has been evolved and deployed wisely over Japan. It detected successfully the 1994 Northridge earthquake, the 1995 Kobe earthquake, the 2003 Miyagiken-Oki earthquake, and the 2004 Niigataken Chuetsu earthquake [6]. UrEDAS includes two types: “UrEDAS” and “Compact UrEDAS”. UrEDAS is used to identify the location and the magnitude of the earthquake while Compact UrEDAS is used to checks whether it is destructive or not and whether alerts should be strigged. In case of earthquakes, UrEDAS can issue the alarm within the radius of 200 km (124.274 mile). Compact UrEDAS’ coverage areas is much smaller than that of UrEDAS, about 20 km or 12.5 mile radius.

Designers of UrEDAS discover that we can examine the rate of the changes in the initial P-wave to detect earthquakes. The rate is proportional to the distance between the seismological observation stations, where the P-wave is studied, and the epicenter. Therefore, seismic P-wave movement can be formulized as a function of Bt*exp(-At) where B and A are unknown coefficients and t is the original time that is taken when the P-wave first arrives at the observation station. A and B coefficients are determined by the least-squares method by fitting the y(t) = Bt*(-At) into the observed seismic waveform.

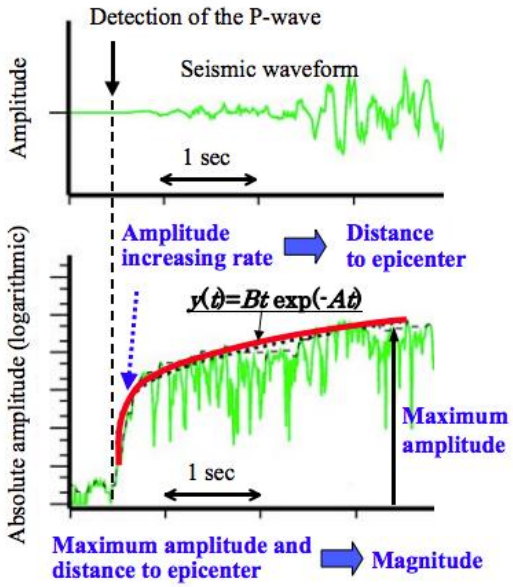


Figure 3: UrEDAS P-wave Variation Function [8]

Each UrEDAS observation station observes and detects the earthquake on its own. They are designed specifically for the seismic detection purposes. They have a mechanical alarm seismograph, an accelerator, an alarm seismometer and a seismic processing unit. Their built-in PC has parallel processing and remote operation with a real-time OS. Their circuits are made so that the electromagnetic noise is counteracted [8]. This allows deploying the seismograph to high electromagnetic areas such as wayside substations.

Figure 4: UrEDAS Earthquake Observation Unit [8]

UrEDAS seismographs have been installed not only in Japan but in other countries as well, such as United States and Mexico, for testing and researching purposes. UrEDAS was

deployed in Pasadena (PAS) and Berkeley (BKS) in 2000 and 2001 in a joined effort in the seismic research project between University of California and California Tokaido Shinkansen to build an UrEDAS alarm system against tsunami earthquake at the Pacific region as well as to improve the accuracy of the earthquake detection at the fault area [7]. PAS and BKS ErEDAS send email to those people that are interested in receiving the real-time data after the earthquakes happen. MEX UrEDAS has also installed in Mexico with the cooperation of the Centro de Instrumentacion y Registro Sismico (CIRES). Detected data will be sent to UrEDAS center at CIRES headquarter office where it will be emailed to the people who are concerned.

2.2.2. Earthquake Early Warning System for High-Speech Railways (HR-EEW)

Seismic researchers not only try to construct the system that can detect earthquakes and give warnings as early as possible but they also try to minimize the cost of the system. As we have a discussion in the above session, earthquake can be detected by a network of the low-quality electronic equipment. In the traditional wired sensor network, we need to spend an additional budget for running and maintaining a cable system between the nodes. Wireless communication allows us to place the sensor nodes in the earthquake regions and let them communicate with each other through the wireless channel. For these reasons, seismic WSN network with the low-cost sensor nodes is a promising architecture for EEW system with constrained budget. However, it raises a question about how many nodes a EEW system needs to handle its desired job effectively. Does the density of the nodes in the deployment area increase the accuracy of the earthquake detection for the system? Caltech computer scientists in [8] show that the sensor networks can still achieve its desired tasks with a reasonable amount of sensor nodes. The main idea is that we can aggregate the measurements from a group of sensors that are co-active to detect the rare events such as earthquakes.

Inspiring by the same sparsification concept, authors in [9] present an earthquake detection architecture that can be built to protect the important facilities from the earthquake damages. The earthquake early warning system is constructed to prevent the potential catastrophes from the seismic activities along Jin-Hu high speed railway systems in China. The primary goal of the system is detecting and giving early earthquake alerts to the train operators so that they can slow down or stop the trains before the high energy S-wave arrives. The earthquake alerts are also sent to the railway control centers so that the train dispatchers can adjust the train schedules or apply the auto break remotely to prevent the trains from being derailed during the earthquake.

HR-EEW system is based on the wireless sensor network. It utilizes the wireless sensors to long range the communication in the outdoor environment a long high-speed raid-road to form a seismic network for earthquake detection and warning purposes. However, the cost of HR-EEW system is minimized by its optimized deployment schema (number of sensor nodes and their locations), which is derived from a constrained budget and predefined performance requirements. The developers of HR-EEW define a set of parameters and use them to find the minimum cost of constructing and deploying the system. They denote SD as the seismic district where the system is setup. E is a set of all points in SD that could be an earthquake's epicenter. R_1, R_2…, R_n stand for the railway lines. V_s, V_p and V_{wsn} are the transmission speed of S-wave, P-wave and wireless sensor network respectively. S is a set of m locations S_1, S_2…, S_m where sensor nodes are installed. sci is the cost of of deploying a sensor node to location S_i for the system. *Time performance requirement* Te and *accuracy performance requirement* k are also defined to as the main performance constraints when

maximizing the cost of the HR-EEW system. Te is the interval between the time that the train control center receives earthquake warning reports and the time that the destructive S-waves hits the rail-road infrastructures, which are R_1, R_2, …, Rn in this case. From the defined set of parameters, [11] provides mechanism to identify the subset of S that has the lowest cost while satisfy all variable SD, R, S, k and T_e.

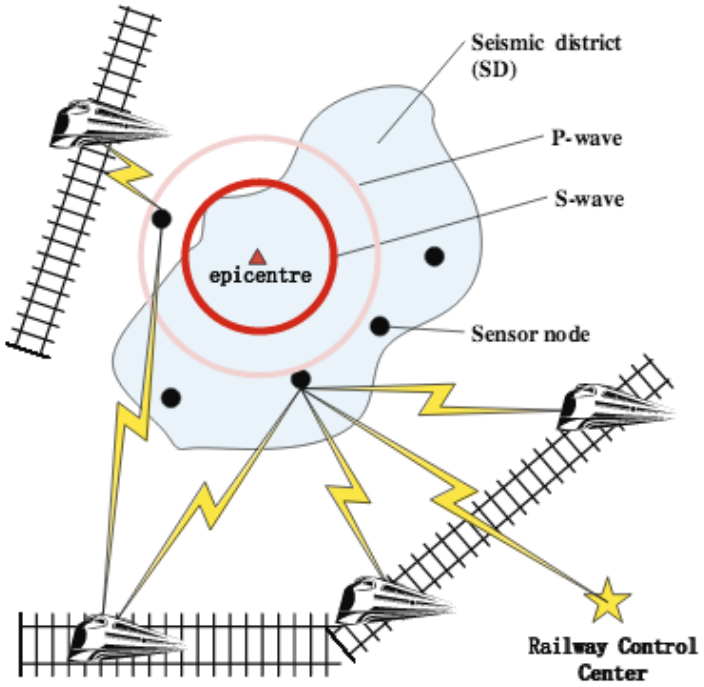


Figure 5: Bird's Eye View of HR-EER System [9]

Different from other wireless EWW system in which the member nodes communicate with each other through the short range radio frequency channel and the sensory data is relayed to the sink nodes by intermediate nodes, HR-EEW sensor nodes do not collaborate with each other in detecting and triggering earthquake warning messages. When an earthquake is verified, HR-EEW seismic stations send earthquake alerts directly to the receivers at the protecting regions. The location of the seismic stations is very critical to the efficiency and performance of the HR-EEW. Because the number of sensor nodes are reduced, HR-EWW needs to identify perfect locations in the earthquake sensitive area to install the sensor nodes. They are between the epicenter and the target area such that the alerts can arrive to the railroad control centers, railroad stations and train operators before the S-wave strikes. The longer the distance between the seismic stations and the railroad lines, the more time people can have to prepare and respond to the incoming of an earthquake because the earthquake alert messages travel almost 40,000 times faster than P-waves do. However, we do not want this distance to be too long so that the communication cannot happen due to out of wireless radio coverage.

2.2.3. Community Seismic Network (CSN) System

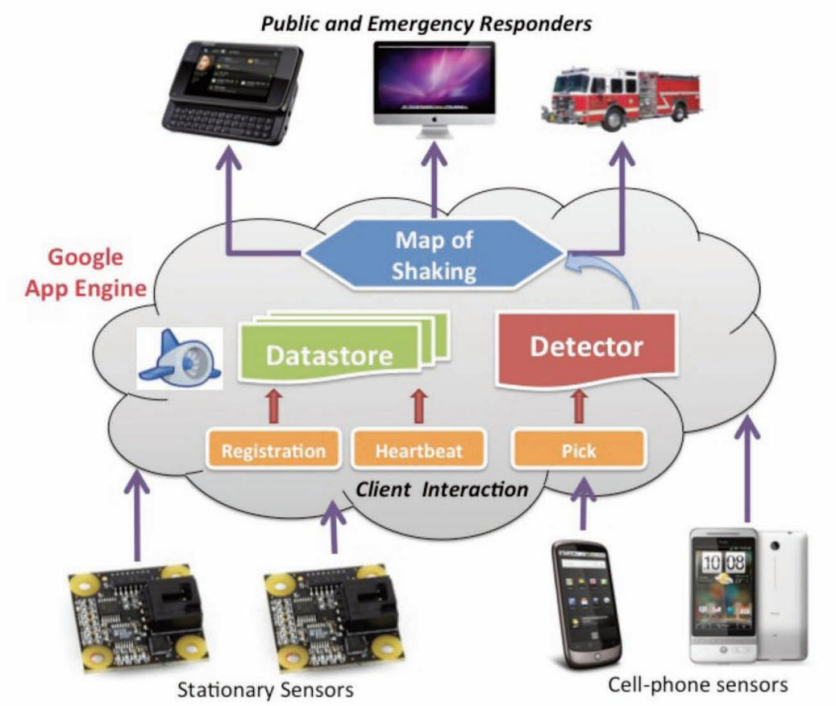


Figure 6: CSN System Architecture [10]

CSN is one of the first attempts to detect the seismic shakes using crowdsourcing by the searchers at California of Institute of Technology, Caltech. The main objective of the CSN network is constructing a shaking map of the affected region within a couple seconds after an earthquake occurs so that the first responders can have a better view of the damages; and therefore, can provide better assistances to the victims.

The CSN network is comprised of CSN clients and CSN servers. CSN clients are installed and run on volunteers' personal computers, smartphones and plugin computers. CSN clients register its geographic location and address with the CSN server when they join the network. They collect picks generated by the potential earthquake shakes and apply STA/LTA algorithm to filter out the non-seismic signals. The seismic picks are then submitted to the CSN server to be processed.

Each seismic pick contains the latitude and longitude of its source node. The CSN server aggregates the data to produce a map that reflects the ground motion of a potential seismic event. CSN server also analyzes the collected picks to identify whether they are from a real earthquake. Basically, CSN server creates a virtual grid map for the area it is monitoring. The map contains the geo-cells that represent the deployment location of CSN clients. Each geo-cell holds the current earthquake detection result of the node it associates with. When the amount of the geo-cells (CSN clients) is large enough, the CSN server can count the number of the geo-cells with the earthquake detection report to identify whether an earthquake has occurred. An earthquake is confirmed when the number of detections within the geo-cells in a

particular timeline exceeds a specific threshold that has been defined in the CSN server configuration [10].

To increase the robustness and the scalability of the CSN system, the CSN server is hosted on Google App Engine, which is a cloud-based infrastructure. Because earthquake does not happen frequently, the system often receives small traffic under normal conditions. However, when a strike is hit, CSN clients will produce unpredictable amount of traffic, which could be too much to process for a single server. Hardware failure and data lost are also big concerns of a self-hosted system. Google App Engine runs off Google data house centers which locate over the global with a well-designed load balancing mechanism. The data is also replicated and backed up in many places. Therefore, the risk of data lost due to crashed or malfunctioned servers is minimized. Hosting CSN servers on Google App Engine also avoids single point of failure when the CSN server is happening to be in the earthquake striking area. We want to get the seismic picks out of the affected region as quickly as possible so that the data can be processed for useful decisions. All Google data house centers are operating in the safe zone areas where the probability of being damaged by an earthquake is very low.

The significant achievement of the CSN system is the use of crowdsource to obtain necessary measurements for earthquake detection. However, the accuracy and effectiveness for this approach relies on the number of participants who are willing to run and maintain a CSN client at their own place. Not only that, CSN clients passively collect the seismic picks when an earthquake happens. They cannot declare and trigger the earthquake alerts by themselves based on the data they collect from their surrounding neighbors. Only CSN server can send out alerts, which could be delayed due to the network condition during the earthquake. In the worst scenario, CSN servers do not receive enough detections from CSN clients to claim an earthquake. It is not because earthquake is not happening but it is because it was so destructive that it had disrupted the internet or the power supply of the whole region it stroke. We need a more resilient seismic network that can survive through the big quakes to give us the warnings as earlier as possible before the S-wave arrives. In addition, we need the network to give us some brief estimations of the damages and the signs of survivors after the strike.

Chapter 3. Research Problem and Approach

3.1. Research Problem

Through our survey on the current implementation of the EEW systems, we recognize that there is still space for the improvement in the seismic detection and early warning area. Most of the EEW systems we found are built for commercial use and hard to scale. We want to install the EEW system not only at critical places, such as business buildings or bridges, but also in residential areas to increase the safety of our living environment. For this reason, it is necessary to have an affordable EEW system that can be deployed widely.

A centralized architecture like the CSN system to detect and alert the earthquake promises to bring down the construction and the operating cost. We can deploy the low-cost seismic sensor nodes to the earthquake sensitive regions. The seismic nodes then observe and submit the potential seismic signals to a remote server, where the data is aggregated and analyzed to detect the possibility of an earthquake. When an earthquake is detected, the remote server broadcasts the alerts to all the end users in the network. However, the proficiency of this approach relies on the communication through the backbone internet, which could experience some slowness or interruptions when a strong earthquake strikes the target region. The remote server could not receive enough the seismic data from the sensor nodes in the striking area to confirm that an earthquake is happening. Even if the remote server could identify the ongoing earthquake and send out the alerts to the end users, the transmission time over the internet through the earthquake impacted areas could make the alert messages become less valuable when they get to the users. For these concerns, the client/server architecture cannot guarantee the liability of the EEW system.

We can resolve this issue by moving the earthquake detection process from a remote server to a local node when alerts are expected to be delivered. By doing this, we can cut short the detection and alert process because we do not need to ship the seismic data to the remote server to be processed and the alert messages do not have to go through the internet to get to their destinations. Each node can make its own analysis based on the data that it has been collected and trigger the alerts when it needs.

To increase the accuracy of the local detection as well as preventing the false alerts, it is very important that nodes can gather a sufficient amount of data from the accelerator sensor of its own and other neighbors. Therefore, a network through which seismic nodes can exchange data is required. Obviously, we cannot construct this network over the backbone network; otherwise the new system will have the same issue that we discuss in the centralized approach. A wired network to connect all seismic sensor nodes would remove the dependence of the internet infrastructure; however it is impractical due to the complexity, cost and scalability. A wireless ad-hoc network seems more suitable for our new EEW system. We can configure the seismic nodes so that they can communicate with other nodes within their wireless coverage ranges. This will allow the nodes to form a local wireless ad-hoc network around their deployment location so that they can share and collect the sensory data from other surrounding nodes. A drawback of this approach is the isolation problem. In the real world deployment, we cannot always enforce the nodes to be deployed adjacent to each other. It is more likely that some nodes will be out of the coverage range of the big group. Therefore, they cannot achieve enough data from other local nodes to do their own detection.

After looking into the pros and cons of many possibilities, we conclude that a combination of the centralized and decentralized architecture is the final choice for our proposing EEW system. Besides linking with other neighbors, each seismic node connects to a cloud server. The seismic nodes disseminate their own or neighbors' sensory data into their local ad-hoc network so that the local member nodes can get a copy of each other data. At the same time, they also send a copy of the collected data to the remote server if the data has not been submitted. The remote server does its own detection and disseminates the earthquake alerts to the nodes that fail to detect the ongoing earthquake in their area.

Nevertheless, the proposing hybrid decentralized architecture for the EEW system give us new challenges to solve. We need to make sure that seismic nodes in our new EEW system are able to gather sufficient amount of seismic data from other sources for their own detection. They only have a couple seconds to process the detection and deliver the warning messages to the subscribers based on the seismic noise from the P-waves. The earthquake notifications have to get to the end users before the S-waves hit the region; otherwise the detection result would become less useful. Therefore, It is essential that the seismic data can be transmitted quickly and reliably to the remote server and the close-by nodes. For the scope of this page, we just focus on resolving this requirement locally. The fastest path for the seismic data to travel from its source node to the cloud server is left for another paper.

In some respects, our proposing hybrid decentralized EEW system has the same structure of a wireless sensor network (WSN). It contains a large amount of the sensor nodes. Sensor nodes relay the sensing data from the source nodes to a sink node (sensor nodes with internet connection) when the data is shipped to a remote server. It seems that we can implement some available techniques of WSN for our proposing EEW system.

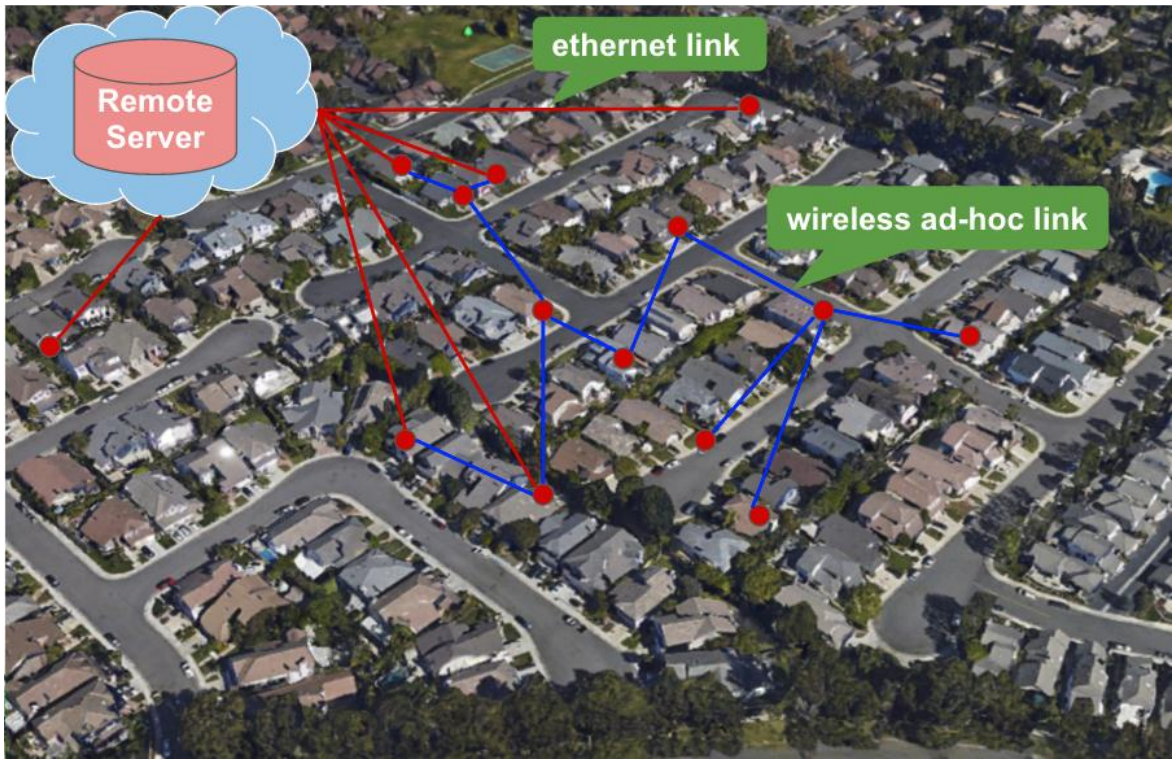


Figure 7: Bird's Eye View of Our Hybrid Decentralized EEW Network

3.2. Potential Routing Protocols for New EEW System

WSN is not a new topic in computer literature, though its application domain has not been involved significantly, especially under the growth of the internet of things. Constructing a wireless network of connected devices to collect the sensing data for military purposes, environment monitoring, healthcare service and industrial needs has become very popular. Many routing protocols have been proposed to increase the efficiency of data collection through WSN. Some of them will be discussed briefly in the following sections.

3.2.1. Flat-Routing Protocols

SPIN routing protocols are well-known in the flat-routing category for its efficiency in saving energy and eliminating redundant data. Heinzelman, who invented SPIN, recognizes that the sensor nodes do not need to share their sensing data to everyone else in the network. They instead only need to distribute the data to those who want to possess it. He is also aware that the traditional routing protocols, such as flooding and gossiping, consume a significant amount of system energy and bandwidth because the duplicated copies of data are populated over the network. Therefore, he constructs a negotiation algorithm that allows the nodes to achieve the data they desire for. The negotiation involves 3 steps where the sensor nodes send or receive either ADV, REQ or DATA messages. When a SPIN node obtains new sensing data that it wants to share with other nodes, it broadcasts an ADV message which only contains the description of the data. Neighbor nodes who are interested in the new data will send a REQ message to the broadcasting node to request for a copy of the data. Finally DATA message is sent to the requesting nodes. The data diffusion mechanism reduces a lot of unnecessary transmission. Nodes communicate with each other through the small data packages (ADV and REQ) before the big packages (DATA) are transmitted. The number of packages to be transmitted is also relatively small. The source nodes only send the sensing data to the neighbor nodes who request it.

SPIN-1, SPIN-2, SPIN-BC, SPIN-EC and SPIN-RL are all under the SPIN routing protocols family. They all share the three-stage basic negotiation process as being discussed above. However, an additional extension is added to SPIN-2 to conserve energy for each individual node when they are operating under the battery mode. A power threshold is set for the SPIN-2 nodes to adjust their sharing behavior. When the energy of a node is getting close to the threshold, it reduces broadcasting the ADV messages to other neighbors to reserve energy. The node only participates into the network if it can ensure that its participation would not drain its power below the minimum power level [11,12]. SPIN-BC, SPIN-PP, SPIN-EC and SPIN-RL are derived from SPIN but they are specially designed networks. For instance, SPIN-BC is used for broadcasting channels while SPIN-PP is implemented for point-to-point communication.

The main drawback of SPIN routing protocols is their data delivery ability. The SPIN's data advertisement mechanism does not guarantee that the sensory data is delivered to where it is needed, especially in a larger network [11,12]. Data from the east side of the network cannot reach to the west side nodes if the middle nodes are not interested in having it. The speed of the populating data over the network is also an issue in the SPIN protocols. For those applications that are time-sensitive like EEW system, the SPIN three-stage negotiation process data dissemination adds extra time it needs for the nodes to achieve the sensory data from other nodes in the network. Therefore, their view about the real time traffic in the local network may be out of date and cannot be used to make any good decisions and notifications.

Directed Diffusion is also well-known algorithm in the flat routing protocol category for WSN. Unlike SPIN, which the sensing data is advertised by its source sensor nodes and is sent to other sensor nodes in the network when they request, Directed Diffusion lets the sink node to initial the data transmission process. The sensor nodes collect and aggregate the sensing data around its neighbors into the data objects that the application is designed to observe. Data objects could include data type, instance associated with its data type, location where it is created, confidence of the match when querying the data again, time the data is recorded and time interval that it is still considered valid. For instance, seismic sensors' data can have the following structure:

Type: seismic shake
Instance: 6.0 earthquake
Location: Latitude: 40.712784 | Longitude: -74.005941
Confidence: .85
Recorded timestamp: 06-10-2015 12:16:20
Duration: 5s

The source node assigns a name for the data it creates. The pair value is the node data list table to be looked up and diffused to the sink nodes on request. The BS nodes broadcast the request for data they are looking for into the network in the form of interests. An interest describes a specific redefined task that the WSN needs to achieve. When a node receives an interest, it checks to see its data list table to see if it has the data that matches with the interest's description. It sends the data to the node where the interest comes if the data is available at its location. If the data is not, the node forms a gradient from its current location toward the neighbor from which it receipts the interest package. A gradient contains attribute value and its direction. The gradient creating process is continued until an path is established from the current node to the base station. Beside this, the node also propagates the interest to other neighbor nodes which have not received it. The interest propagation mechanism is repeated until the requesting data is found. The source node picks the best gradient path to transmit the data back to the BS node or the requester in its local network [11, 12].

Loops could occur among gradients during the interest propagation step. However, a cache mechanism is applied during the data dissemination to make sure that data packages are not looping in the network. Nodes store a copy of the data pair value (date name and content) on local for an interest it that it relays. Therefore, it can respond to a similar request on behalf on the source nodes in the future.

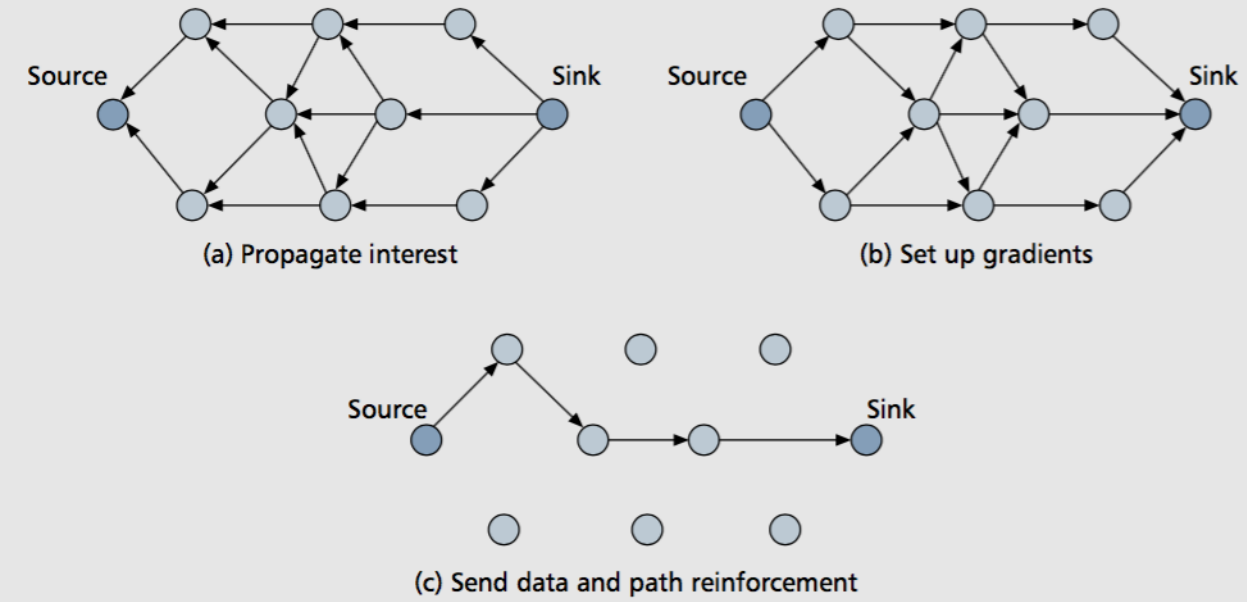


Figure 8: Directed Diffusion Data Propagation Path [11]

Directed Diffusion improves WSN network lifetime by eliminating redundancy transmissions. Only data the sink needs is travelled through the network. The path reinforcement guarantees only nodes on the chosen path are involved in the data transmission process. The rest of the network can stay inactive to conserve energy. However, this mechanism causes some weaknesses for the routing protocol itself. Package lost can happen among intermediate nodes during the interest broadcasting or when the source node sends data to the sink in response to its broadcasting interest messages. The sink node, for the reason, does not get the data as it expects. The author of Directed Diffusion routing protocol suggests that sink nodes periodically rebroadcast the previous interests to make sure the interest's source node can get their requests. The approach can prevent package lost but it adds additional traffic into the system due to interest storm. Another weakness of flat-routing protocol family is that it does not utilize node's location to prolong network lifetime. Nodes in critical position such as those that are close to the sink nodes or on the bridge between 2 subnetworks often receive more traffic. They have a high probability of being drained out of energy, which could interrupt or stop the communication between the sink nodes with a portion or the rest of the network. The issues are tackled in location-based routing protocol family.

3.2.2. Location-based Routing Protocols

Location-based routing takes a different approach from other routing protocols. It exploits sensor nodes' positions to find the best routes to transmit data within the network. The challenge of the location-based approach is the way to obtain nodes' location. In some location-based routing topology, each node uses incoming signal strengths to estimate the distance between itself and its neighboring nodes. It then exchange information with its neighbors to obtain their relative coordinates; however, the location of each node may be not be accurate because the wireless signal strengths could be affected by interference and obstacles. [11] The alternative solution is to equip each node with a Global Positioning System (GPS) unit so that it can communicate with a satellite to identify its current position. This technique can be expensive

in terms of energy consumption and financial cost. Low powered GPS receivers with active and sleep cycles are often implemented to save energy [11]. In order to reduce the cost of the whole network, only a portion of the nodes have the capability of GPS communication, and these nodes share their coordinates with their neighbors.

One big advantage of location-based routing protocol is energy conservation, and Geographic Adaptive Fidelity (GAF) is famous for this advantage. GAF divides the network area into fixed zones in a grid fashion. Nodes within each specific zone of the virtual grid collaborate with each other to route traffic when the traffic goes through their region. Only one node is kept awake for a certain period of time in order to monitor and forward data to the base station (BS) for other sleeping nodes in this zone. Active nodes go back to sleep after their duty time expires. Sleeping nodes wakes up after their sleeping period, and they have to stay awake if it is their turn. Each node adjust their sleeping and active time based on the routing demand and condition of the zone it belongs to. This mechanism allows the network to utilize less nodes to transmit data to the BS; therefore, the energy of the whole network can be conserved.

Geographic and Energy Adaptive Routing (GEAR) is another energy-friendly routing protocol for WSN. It takes the location and the current energy status of each sensor node into consideration when setting up routes for the WSN. The GEAR restrict the number of packets sent to a specific intended region of the network instead of the whole network. By doing this, it helps to reduce the amount of unnecessary transmissions and conserve energy for the overall system. Each node in the GEAR keeps track of an estimated cost and a learning cost for reaching the destination through its neighbors. The estimated cost is calculated based on the power level of the node and its distance to the destination. The learning cost is an adjustment of the estimated cost that accounts for routing around holes in the network. When a node does not have any neighbor that is closer to the target region than itself, it encounters a routing hole. In this case, it uses the learning cost to pick a neighbor that can forward packets to the target region. It then decreases the learning cost by one when the packets are successfully delivered so that it can be used to set up route for other packets when they arrive. If the node does not have any routing hole, it picks a neighbor with the smallest estimated cost to be the next node. When packets reach their destination regions, a recursive geographic forwarding or restricted flooding is applied to diffuse them to other nodes within the area [11].

3.2.3. Hierarchical Routing Protocols

In hierarchical or cluster-based routing architectures, wireless sensor nodes are divided into smaller groups and assigned different roles based on their energy level. Nodes with high power are in charge of processing and sending the information to the target destination while low power nodes are responsible for sensing data only [11, 12]. The routing protocols involve into two main phases. Clusters and their leader or cluster heads (CH) are established in phase one. Routing traffic is conducted in phase two where CHs will act on the behalf of the member nodes within their region to transmit the sensory data to the designated destinations. LEACH, PEGASIS, TEEN and ATEEN are the names of a few WSN routing protocols that organize nodes in hierarchy structure to lower the power consumption for the whole network.

Low Energy Adaptive Clustering Hierarchy (LEACH) protocol was proposed to lower the power consumption and increase lifetime for WSN. LEACH attempts to distribute the traffic load evenly over the network to prevent the important nodes being exhausted due to overload. To achieve this goal, cluster head (CH) nodes are selected randomly; and their roles are rotated based on the energy distribution among the sensor nodes in the network [31]. CHs broadcast

the advertising messages to the nearby nodes to announce their roles. Non-CH nodes connect and send data to the CH node whose advertisement messages have the strongest signal strength. To reduce the traffic load within the network, CH nodes aggregate data from other nodes before submitting it to the base station. The data combination and compression are done periodically. Therefore, the protocol works for those applications where the sensing data can be collected after a specific period of time such as habitat study or monitoring.

LEACH operation involves the setup phase and the steady state phase. During the setup phase, clusters are formed and CHs are elected so that data can be transferred to BS in the steady state phase. The time LEACH network in the steady state phase is set to be longer than that of the setup phase to minimize the overhead cost. After the network stays in the steady state phase for a pre-defined period of time, it goes back to the setup phase to restructure its clusters and CHs. The clustering mechanism allows prolonging the network lifetime because every node eventually can contribute to data transmission back to BS for other nodes when it becomes a CH. However, its performance is based on the assumption that CH nodes are distributed uniformly over the network and that each node has enough energy and computational power to handle CH tasks. In reality, there is high possibility that CH nodes are concentrated in a certain part of the network while none of them can be found in others. Each node may have the same initial power but it is consumed differently after it joins the network. Therefore, energy level at each node is not always equal and its contribution to the network cannot be expected the same. Periodically reconstructing the clusters and re-selecting CHs help to spread the traffic load among the nodes and extend the system active time. In the meantime, the processes themselves create extra overhead due to the advertising messages and CHs selection.

Power-Efficient Gathering in Sensor Information Systems (PEGASIS) is an improvement for LEACH. It tries to expand the network lifetime even longer by utilizing the chain architecture for WSN. Nodes communicate with its closest neighbors to form a chain. The distance from each node to its all neighboring nodes is measured by the signal strength that it can hear from them. Unlike LEACH, where multiple nodes are in charge of transmitting sensing data to the BS, PEGASIS only uses one node in the chain for this task [11, 12]. Each node in the network takes turn to communicate with the BS to submit the data that it has collected from other nodes. A new round is started again when all the nodes have finished their turn. This approach allows every node to participate in gathering and sending the sensory data to the BS, and therefore spreading out the powder draining uniformly over the network.

Experiment result shows that PEGASIS network lifetime is twice longer than that of LEACH. This performance enhancement is as the result of eliminating the dynamic clustering formation and reducing the number of data transmission and reception. However, PEGASIS introduces new overhead of the network due to its dynamic topology adjustment. Each node needs to identify the power status of its neighboring nodes to decide where it should route the traffic. This process can be exhausted and cost nodes a significant amount of energy. PEGASIS is also not applicable for high time-constrained applications such as natural disaster early warning system. Only one node is allowed to transmit data per time. Other nodes have to wait for their turn even though they have data being queued up. The delay time is getting bigger when the size of the network increases because it takes longer to get to their turn.

Threshold-Sensitive Energy Efficient Sensor Network Protocol (TEEN) and Adaptive Periodic (APTEEN) were proposed to deal with the delay issue that LEACH and PEGASIS have. The two new protocols focus on the network energy sustainability but they provide us a

way to adjust the level power consumption and the speed of data transferring among the nodes within the network. Like LEACH, TEEN also organizes nodes into clusters and CHs. However, in TEEN, CH sensor nodes send their members a hard and soft threshold to control the number of transmissions the nodes can submit [11]. The hard threshold is the value of sensed attribute; and the soft threshold is the change in the sensed attribute value. TEEN member nodes only transmit data to their CH when the sensed data value is above the hard threshold value, therefore reducing the amount of unnecessary transactions. The soft threshold value is updated more often by CH nodes to reflect the changes in application interest. It is used to reduce further the number of transmissions when the sensed attribute value is modified. It actually also increases the accuracy of the collected data in depicting the network picture because it allows the member nodes to actively filter out the unwanted data based on the instruction from the application.

APTEEN is derived from TEEN protocol but its CHs broadcast more parameters to optimize performance of the member nodes as well as their sensed data transmission. Beside the attributes and thresholds set, APTEEN CHs also send the TDMA schedule and count time parameter value. The TDMA schedule let each member node know its assigned slot so that their transmission does not interfere or is not interfered by other nodes. The count time is the maximum time period between two successive data submission. If a node does not send any data to its CH with the count time period, it is forced to sense and retransmit again. Compared to TEEN, ATEEN provides a more flexibility in collecting the sensing data by allowing the user to set the value of the filtering attributes. However, calculating TDMA schedule and the count time value is an expensive operation, which adds extra load to the CHs.

3.3. Issues with Integrating Current Routing Protocols for EEW Systems

Our research on wireless routing protocols shows some limitations of using them for our proposing EEW system. Our seismic sensor nodes are not ensured to achieve the necessary data for their local detection through the data sharing and disseminating techniques that have been proposed. WSN routing protocols, such as LEACH, GAF and GEAR, aim to minimize the power consumption of each sensor node to prolong its network lifetime. The sensor nodes are set to switch back and forth between the active and inactive state to conserve energy. These saving mechanisms would prevent a relatively large number of the sensor nodes in the network to be able to observe the real-time seismic signals when an actual earthquake is coming. The success of our decentralized earthquake detection architecture really much relies on the collaboration among the nodes. All seismic nodes need to be in active mode all the time to sense, share its sensing data, collect sensing data from other peers, and analyze the collected sensing data to detect and alert any potential earthquake. Furthermore, we propose the decentralized EEW system to be deployed to the residential areas. The seismic nodes will be installed and run at the volunteers' house or apartment where the access to the power supply is not an issue. Therefore, energy saving is not a challenge for us but being able to capture as much as seismic data from an active earthquake is very valuable.

The required speed of disseminating and collecting data process for the decentralized EEW nodes also does not allow us to be able to apply the existing WSN routing techniques. SPIN provides a lightweight data-centric routing protocol which could be the good candidate for a decentralized EEW application. SPIN nodes do not send data to other nodes in the network. They instead broadcast the advertising message to describe the data they have. Nodes that are

interested in the new data will need to send a request to the advertising node to get a copy of it. This data dissemination process help reduce the amount of the unnecessary traffic. However, it takes extra time for the nodes to get a copy of their neighbors' data. For EEW application, saving a couple seconds is very critical. Therefore, we want to have a dissemination scheme through which data from the source nodes can get to its destination nodes as quickly as it can.

Last but not least, the structure of our hybrid decentralized EEW system is a little bit different from that of the traditional WSN system. In most of the time, our seismic nodes are also the sink nodes for themselves. Therefore, they do not often need to relay the seismic data to a designated sink node to transmit it to our remote server. However, during earthquake and post-earthquake periods, we expect that some nodes in our decentralized EEW system could lose their connection with our remote server. A relay mechanism is still essential in these cases. The challenge of relaying data within the network of our proposing system is that the relaying data could have multiple sink nodes to go forward. Most of relaying schemes are designed to work with a single or a few designated sink nodes. Furthermore, the remote server of our new EEW system not only collects but also injects data into its network. The application data travels in both directions, from seismic sensor nodes to the remote server and the other way around. At the same time, the data is also required to relay to other nodes surrounding its original source node.

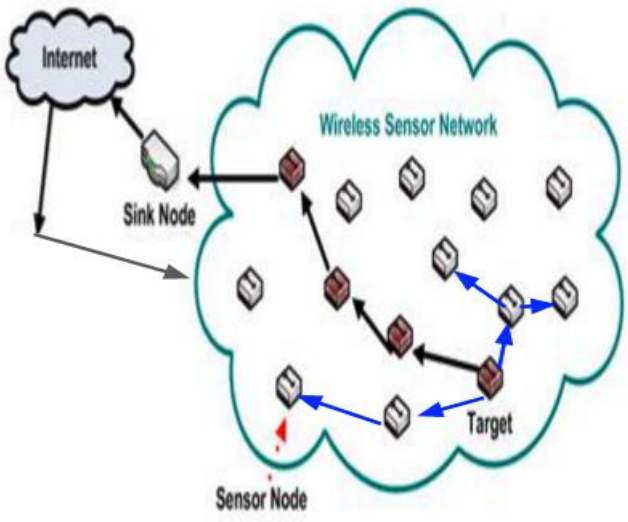


Figure 9: Dissemination Directions in Hybrid Decentralized EEW systems

Unfortunately, sharing and relaying through broadcasting and multicasting mechanism cause some issues to the application such as bigger delay time and low data rate due to the congestion and the package lost. Interference is also a big concern when a large number of adjacent wireless nodes are sending at the same time. In the meantime, sharing and relaying data mechanisms provided by current WSN routing protocols mostly attempt to conserve node energy, adding additional time in the process. EEW system can only tolerate a couple seconds delay or its output is less useful. For these reasons, we need a robust routing and dissemination algorithm in which data can be propagated over the wireless ad-hoc network of proposing EEW system as fast as possible, regardless the changes in its network topology due to the noises from the deployment environment.

Furthermore, we project our local distributed quake system work reliably in three different scenarios in which the network topology could be changed completely: Pre Earthquake, During Earthquake and Post Earthquake. Before an earthquake occurs, most of the nodes in the network are in good condition. Their internet connection and power status are sufficient. Traffic in the network is light because nodes do not have any sensory data to share. During the quake, nodes in the impacted region will observe a lot of shakes, and therefore will inject a tremendous number of the seismic picks into their local wireless ad-hoc network. The accelerator sensing data from each node is only valid for the local quake detection if it is available at other local nodes in less than a few second after it is sensed. This is because we want to be able to send out the earthquake alert before the S-wave arrives so the residents can adjust their current activities to be safe through the shake. Fortunately, study has shown that earthquake detection can be done with a small data set from a large seismic client network [8, 9]. Each seismic node in the EEW system is only required to gather seismic data from a decent amount of its peer nodes instead of from the entire network to carry out its own detection. This tells that an efficient routing and disseminating mechanism in which seismic sensor nodes can share and receive sensing data from other nodes within their deployment radius defines the performance of our hybrid decentralized EEW system.

3.4. Forest-Based Dissemination Scheme for Wireless Ad-hoc Network (FaNet)

3.4.1. Reasoning of Disseminating Data through a Forest-Based Network

The key element of a decentralized EEW system is that the detection can be done locally by individual nodes. To achieve this, each node needs to collect a decent amount of the seismic data from other nodes that are within a certain radius as we have discussed in section 3.3. A simple approach is to have nodes broadcast the seismic data they observe or receive from other nodes. However, this could create a bad impact for the network due to the broadcast storm, congestion and duplicated packages. Multicast could help us to prevent these problems but its dissemination speed is not as fast as our proposing EEW system needs. To accelerate the dissemination process among the nodes, we can construct a spanning tree that allows data to be transmitted from its source node to anywhere else in the network. Each node then can disseminate its data to other nodes along the tree. Though it is expensive to construct and maintain one big spanning tree for the whole wireless ad-hoc network, especially when nodes are deployed randomly. Also, this approach seems impractical to implement for the network of low-cost devices because they will be overwhelming with constructing and maintaining their communicating channel; and therefore, have no resource left to execute their designed functionalities. The reliability of the dissemination through a spanning tree is also a concern. It could be disrupted when a link between two nodes on the dissemination path is broken, which is likely to happen more often in a wireless network. In the meantime, nodes do not need data from the whole network for their local detection. We found that rare events can be detected by a small data set that is selected randomly from the system [8, 9]. Accurate detections can be derived from a small subset of nodes that locates in the target region. Therefore, one big spanning tree approach wastes system resource and does not provide the reliability for our data dissemination process. To make it less expensive and more efficient to disseminate through the tree structure network, we suggest to use a set of multiple small connected trees instead. The main idea is that the adjacent nodes collaborate with each other to form a small tree whose size

is defined by the application. Adjacent trees connect to each other through their branches so that data can be transmitted from one tree to other. Together, all the trees form a forest that reflects the data dissemination paths among the nodes in a specific region.

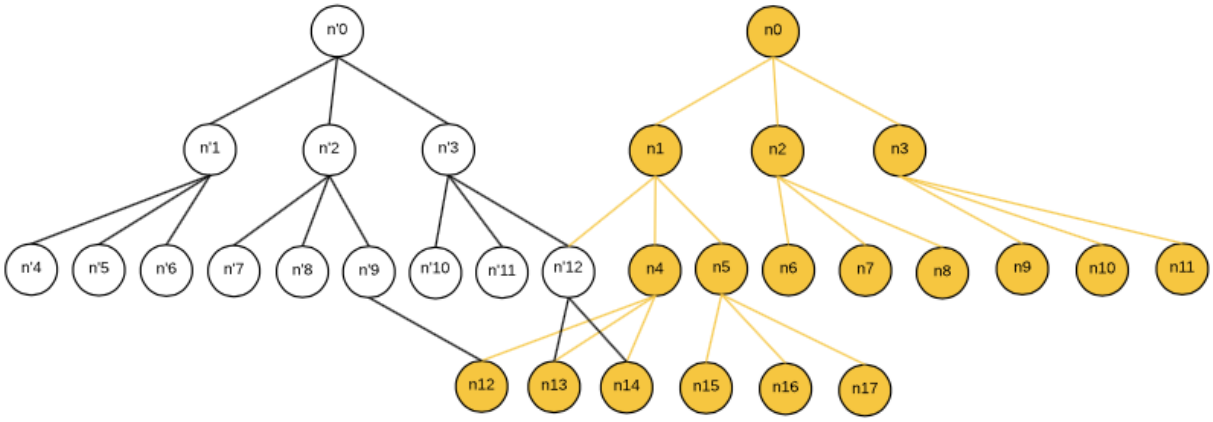


Figure 10: Snapshot of a Forest-based Network Structure

3.4.2. FaNet Tree Components and Properties

Every tree in the forest-based network has the structure of a m-k-ary tree, a rooted tree that can have no more than m levels and its nodes have no more than k children. We use m-k-ary structure to control the size of the tree as well as the number of direct neighbors that each node has to spread out the traffic load over the network. Each tree contains a root node and non-root nodes. Root nodes are initially selected by the remote server based on their location. They help maintain the tree hierarchy for the nodes that want to be their descendants. They also disseminate data from the remote server to all member nodes when they are required. Non-root nodes are intermediate and are leaf nodes. Intermediate nodes relay data between their parents and children while leaf nodes allow data to jump from one tree to another nearby tree.

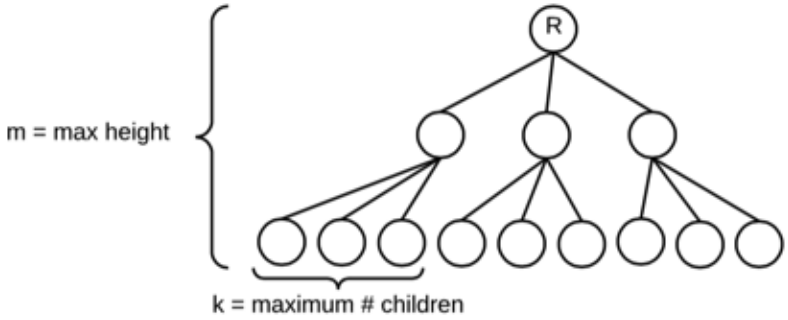


Figure 11: M-k-ary Tree Structure

3.4.3. How to Form a Forest-based Network (FaNet) for Data Collection and Dissemination

When a node is deployed to a specific area, it first registers with a remote server to submit its location and achieves an identification number to communicate with other members. The remote server then checks to see if the area associated with the new node has been assigned a root node. The remote server promotes the new node to be the root node if its deployment area does not have any root node yet.

Next, non-root nodes broadcast a discovery messages. When nodes receive a discovery message, it will return an offer-to-join message to the sender if its current number of children nodes and level in the tree are smaller than the maximum limit. The pseudo logic for existing nodes to handle join requests from other nodes can be seen in Algorithm 1.

Algorithm 1: Accepting New Children Logic for FaNet Nodes

```
While a node is active do
        // Remove inactive children nodes
        for each child node do
                if node has not heard from child node with m seconds then
                        remove the child node
                end if
        end for

        for each pending offer-to-join message do
                if node does not receive confirm-join message then
                        reallocate the reserved child plot for new discovery messages
                end
        end for

        listen for discovery messages from other nearby nodes

        num_children = total current number of its children
        max_children = maximum number of children the node can have
        num_offer-to-join messages = total number of pending offer-to-join messages

        if node received a discovery message for a node then
                if the discovery message is from its peer node then
                        if node belongs to a tree and
                          (num_children + num_offer-to-join messages)  < max_children then
                                reserve a child plot for the requesting node for n seconds
                                send an offer-to-join message to the request node
                        end if
        end if
end while
```

The offer-to-join message includes three main elements: the offering node's address, its primary tree ID and its level in the tree. A node could receive multiple offer-to-join messages from different neighbors after it sent out discovery messages. These neighbors can be in the same tree or different trees. Upon receiving offer-to-join messages, the new node needs to select the best offer-to-join messages to reply in order to claim the spots before they are deallocated for other request. Basically, nodes prefer to belong to the network which can offer

the greatest benefits. Therefore, nodes apply the strategies that are described in Algorithm 2 when picking its parent after broadcasting discovery message.

Algorithm 2: Greedy Parents Selection

- *Nodes accept the offer from a parent node that belongs to a big tree and is closest to its root.*
- *If nodes have offers from different trees, they accept the best one from each tree.*
- *Nodes choose the parent that is closest to the root to be their primary parent.*
- *Where the offers are tied, nodes pick a random one.*
- *Nodes periodically look for better alternative parents.*

After sending out discovery message to look for potential parent nodes to join, nodes waits to hear from other nearby node. Algorithm 3 shows the pseudo logic that each node applies to join a tree in FaNet when it is first deployed or seek for a better alternative parent node.

Algorithm 3: Joining Tree Logic for FaNet Nodes

```
while node is active do
        if node is deployed for the first time then
                register with a remote server to submit its location, get its identification and its initial role.
        end if

        if node is not the root and wants to connect to another node then
                broadcast discovery message

                wait_time = W seconds
                offer_messages = array()

                while (wait_time > 0 ) do
                        // Listen to all nearby nodes for offer-to-join messages
                        new_offer_message = listen_from_all_neighbors();

                        if new_offer_message is not empty then
                                offer_messages[] = new_offer_message;
                        end if
                        wait_time--;
                end

                if new_offer_messages is not empty then
                        apply algorithm 2 to pick the best offer-to-join message to respond
                end if
        end if
end while
```

To prevent the unauthorized members and malicious nodes to connect to the network, we include an application token into the discovery messages. It is actually a hash key that is generated from a global secret key (which is hard coded in the application codes). When a neighbor node receives a broadcast discovery message, it first checks to see if the message is

from the member nodes of its network by comparing its encrypted global secret key with the application token in the message. If they do not match, they will drop the messages and ignore the join request. If these strings match, indicating that the discovery message is legitimate, the neighbor node will send back the requesting node an offer-to-join message if it currently belongs to an unsaturated tree. It then reserves the open spot for the joining node within W seconds. After the reservation time, it will reallocate the spot for other requests if it does not receive any heartbeat or data packages from the requesting node.

3.4.4. Forest-based Network Dissemination Issues and Solutions

3.4.4.1. Node Failure during the Dissemination Process

Our greedy join algorithm allows organizing the wireless sensor nodes into a forest-based structure in which the sensory data can exchanged speedily. However, we found out that the dissemination process could be interrupted due to the single primary parent restriction in our design. We only allow nodes to have one primary parent from each tree they belong to. Therefore, the disseminating data can be blocked when it reaches to a failed node like we depict in Figure 12. If the failed node is the node that connects multiple trees together, the cost of its failure is even higher. It will prevent the disseminating data from diffusing from one tree to other.

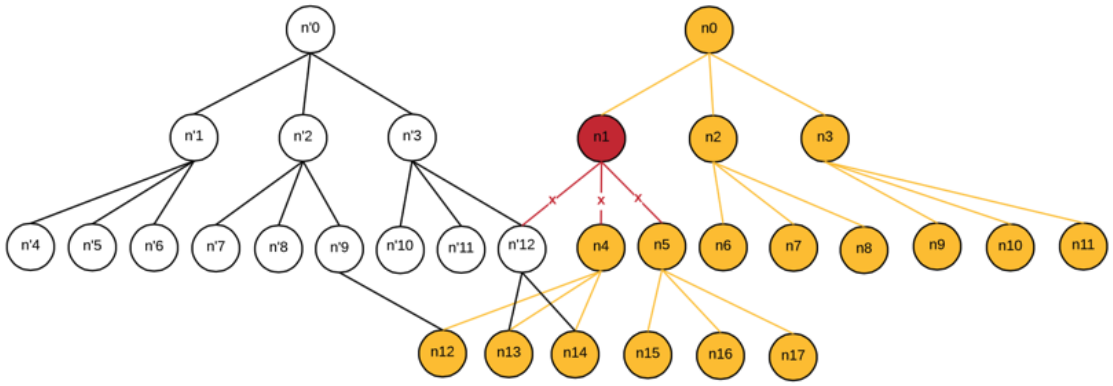


Figure 12: Single Primary Parent Node Failed Issue

To solve this problem, we adopt the multi fan-in multi fan-out dissemination technique from FareCast [16] when constructing our forest-based network. Instead of allowing non-root nodes to have only one primary parent, we modify the design so that the nodes can connect to multiple parents from its primary tree like depicting in Figure 13 below. By enforcing multi-parent multi-child relationship in our forest-based ad-hoc network, we can reduce the package lost due to intermediate node failure during the data dissemination process. The new design ensures that the intermediate nodes have more than one path to share and receive data with other members in their local network.

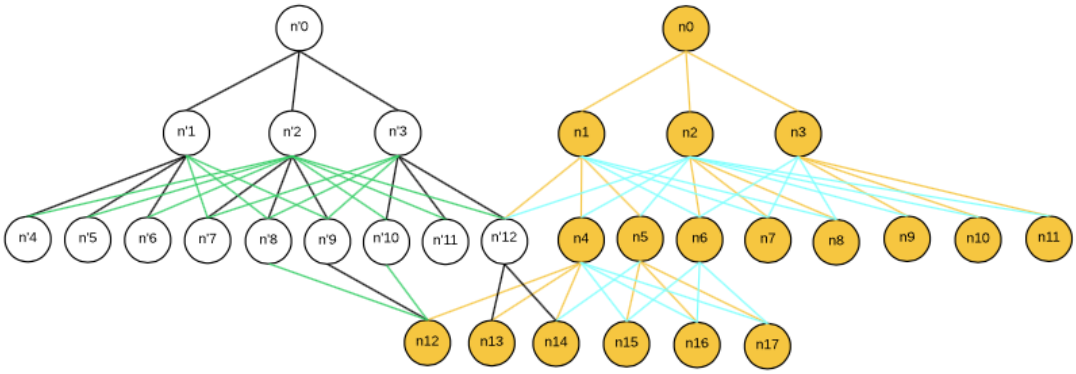


Figure 13: FaNet Structure Improvement (Adapt FaReCast Approach) [16]

3.4.4.2. Looping During the Dissemination and Solution

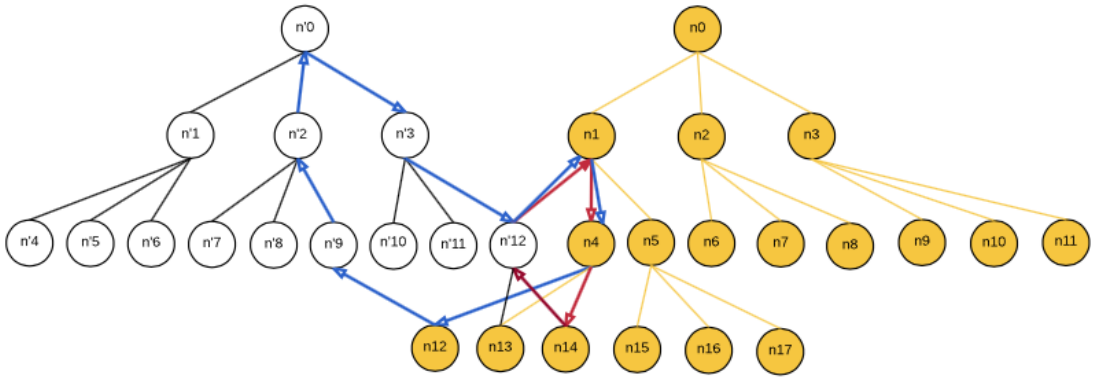


Figure 14: Looping Dissemination Problem in FaNet

As we mentioned above, data is not sent to the nodes that it comes from. However, this mechanism does not completely solve the looping problem, especially when nodes belong to multiple trees. For instance, when node 14 in the right tree in Figure 14 has the sensing data to share, it sends the data to its parent node n'12 in the left tree where the data is relayed to node n1 in the right tree as a part of the dissemination process. Node n1 then sends a copy of this data down the right tree. Eventually, a copy of the data will get back to node n14 where it was originally created. Node n14 will treat the data as a relaying message and forward it to n'12, causing an infinite loop in its local network. Adding ttl to the data packages help prevent them being looped infinitely in the network. Unfortunately, small loops still exist like we show in Figure 14.

To minimize the unneeded transmission due to looping, besides having nodes to check packages' ttl before sharing them again, we also add a logic to make sure that the relaying nodes are not the original source of the messages. We include additional attribute called original_source_id in each diffusing message. Nodes compare their own id with the original_source_id of the messages they get from other nodes. They disregard the messages that were derived from themselves.

3.4.4.3. Link Failure during the Dissemination

We are aware that link failures are common issues in the wireless ad-hoc network. Nodes can temporarily fail to communicate with their parent nodes for a short period of time. When this occurs, the node that connects to the failed link will not be able to directly receive the disseminating data from their parent or children from the other end. Fortunately, this link failure problem does not have a significant impact on the dissemination process when we enforce multi fan-in multi fan-out architecture for our tree. If a node cannot get a copy of the disseminating data from a parent due to its link to the parent is broken, it would likely receive the data from other parent.

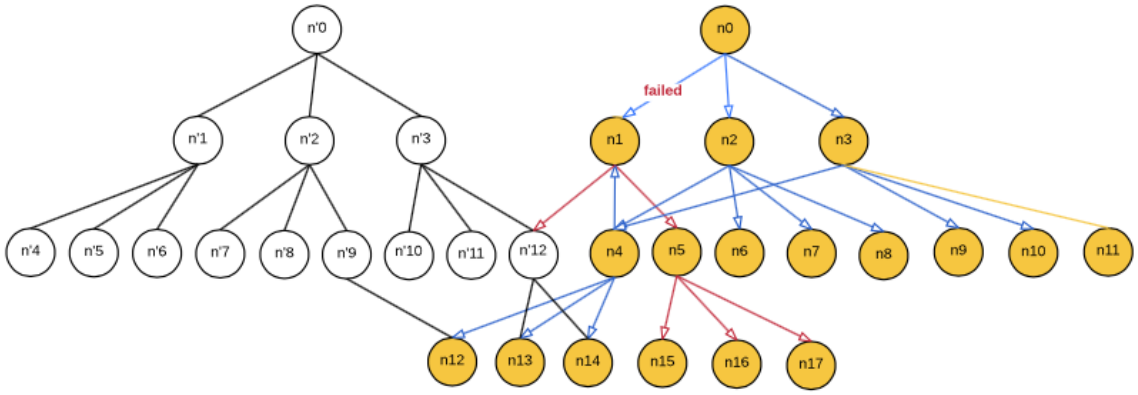


Figure 15: Multi Parent Backup Dissemination Process

3.4.4.4. Node Failure in the Pre-dissemination

Beside the link failures, we also design our data dissemination schema to handle the node failures as well. Our objective is providing a seamless data sharing protocol that works for low-quality devices network in which the node failures are common. We make sure that a solid forest-based network is always preserved so that its member nodes can carry out the dissemination process when it is necessary. For that reason, it is critical that we can identify and eliminate the malfunctioning nodes out of the network. We also need to organize the descendants of the failed nodes so that they can remain connected with other nodes in the forest. In order to accomplish this requirement, nodes periodically exchange the heartbeat message with their parent and children when the dissemination is not in process. If a node does not receive any disseminating or heartbeat messages from a parent or a child node within a certain period of time, it will assume that the relative node has gone. It then stops forwarding the disseminating message to the failed node. If the failed node is a child node, it will deallocate its spot for other incoming join request. If the failed node is the only primary parent, the affected node will promote one of its secondary parents to be its new primary parent. If the affected node does not have any parent left, it will nominate itself to be a temporary root node and announce its new role to all its current descendants. At the same time, it broadcasts a discovery message to look for new parents. When the node found a parent to attach to, it will send out a message to all its descendants to announce its new root node. Other than that, no further action is executed. The descendants of the affected node will decide to stay or leave the tree based on the update information they receive from their parents.

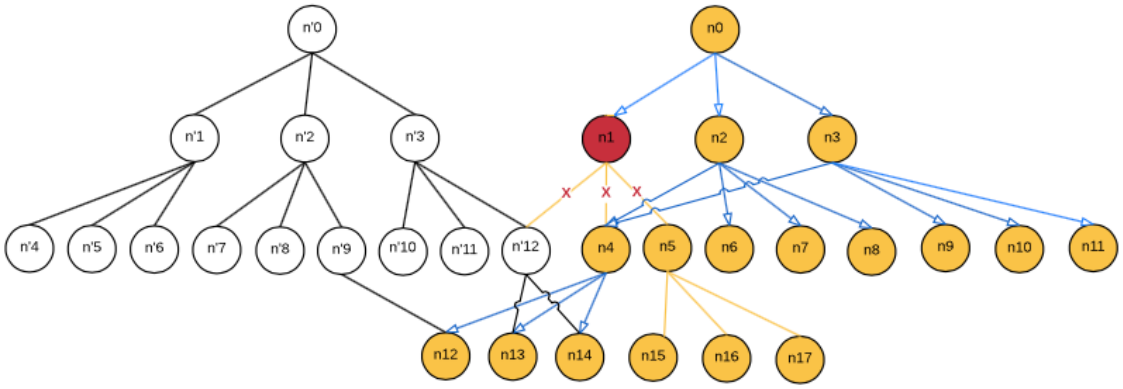


Figure 16: Parent Node Failure in FaNet Tree

3.4.5.5. Parent Node Failure and Solution

Parent node failure is not a critical issue in FaNet network. Each node is allowed to have up to P primary parents. It also actively seeks for new primary parents if the number of its current primary parents is less than P. Therefore, the probability that all primary parents of a specific node are gone at the same time is very small. When this happens, nodes first loop through their current tree list and pick a secondary parent with the highest hierarchy (lowest level) to be their new primary parent. They then send the heartbeat messages with the new selected parent information to intermediate child nodes to let them know that a new primary parent has been elected. These messages will be propagated down the tree until every node in the branch of the tree is informed. If the failed parent nodes do not have any secondary parent to be promoted to its primary parent, they will start a discovery process to find a new parent to join. If no new primary parent is found during the process, the failed parent nodes will decide to break apart the branch below it by announcing that their level in the current tree is now infinite.

Although parent node failure does not have a big impact on FaNet performance, root node failure could cause some problems for nodes under its tree. A root node is the gateway to the internet of its member nodes. It also maintains the hierarchy of nodes in the trees as well as help them collaborate to each other. Data dissemination within a tree will be interrupted if its root node is broken. Thus, nodes need to find a new tree to join or to promote a node to be its new root right after they are disconnected with their current root node. Fortunately, FaNet contains small trees that can be expanded no more than m levels and member nodes can have no more than k primary children. The value of m and k are kept small so that the cost constructing or reconstructing a tree is minimized.

3.4.5.6. Root Node Failure and Solution

Root nodes communicate with its children through the heartbeat messages when the network has low traffic. The messages allow the child nodes (and nodes below them) to identify their hierarchy as well as the status of their parent. When level 1 nodes have not received traffic and heartbeat messages from the root (level 0 node) for a period of time t, they loop through their current tree list and pick the tree whose secondary parent has the highest hierarchy (lowest level) to be its new primary parent. It then sends the heartbeat messages with updated primary parent to its children nodes to let them know that the new primary parent has been elected. These messages will be propagated down the tree until they reach the leaf nodes. If the node does not have any secondary parent to be promoted to its primary parent, it will send parent discovery messages to its direct children which the same discovery primary parent

process will be repeated recursively down the tree. If no new primary parent is found during the process, the original node with failed parent will decide to break up the branch below it by announcing to its children that its level in the current tree is now infinite. When receiving the message, nodes in the branch will detach from its primary parent and start looking for new tree to join.

In the worst case scenario, the root node is failed and none of its member nodes can find a new tree to join. All nodes in the tree will be broken apart and start to form a new tree again. However, none of these nodes or its neighbors can be the root node because they have lost the access to the cloud. We want to maintain the mesh network for the local nodes so that they still communicate and exchange data even though their connection has been interrupted. Another criteria is considered when selecting the root node for a tree. It could be the node with internet connection or in good status in terms of power and computational capacity. Therefore, a tree will be established eventually no matter what. The size of the tree is kept small (m level x k children) so that the cost of constructing and reconstructing the tree is not too expensive.

3.4.5.7. Unsatisfied Nodes

In our forest-based ad-hoc network, nodes are allowed to attach or detach from a tree for their best interest. Each node applies the greedy join algorithm to select and maintain its parent list. Fundamentally, nodes always seek for new parents through which they can communicate with a large group. As the result, nodes periodically broadcast the discovery messages. To reduce the negative impact of broadcasting and allow the network to be converged, we divide nodes into three groups: root nodes, *satisfied* nodes and *unsatisfied* nodes. Root nodes are dedicated to organize and manage their tree structure. They also disseminates data from the remote server to other nodes in the tree. To be picked as the root for a tree, the node requires to have the internet connection. Therefore, the root nodes do not connect to other tree to reduce the traffic load to their way. *Satisfied* nodes connects directly to a root node. They are in a good position because they are closest to the system resources. Every non-root node tries to reach to this spot in which it satisfies with its status and stops looking for alternative options.

Regular nodes periodically broadcast the join messages to its neighbors. The broadcasting period depends on the distance of a node to the root of its primary tree. Nodes that are far away from the root node send discovery messages more often than those that are nearby. After broadcasting the discovery messages, if a node receives an offer-to-join message from node that has higher level than one of its current primary parents, it will accept the join offer and reorder its parent list. If the tree list exceeds the maximum allowed value (n), it will drop the tree with lowest level and notify the parent node on the tree that it is leaving. When the parent receives a leaving message from its child, it will remove the child node from its children list and give this spot to another node that wants to be its child. By gradually moving closer to the gateway nodes, the greedy parent selection method helps each node get the best connection to the cloud. It also allows each node to be able to collect the most amount of data that is shared by others within its local area.

Broadcasting the discovery messages also allows nodes to reattach to the network when they are disconnected with their current parents. To parent-children relationship, parent nodes periodically send the *update-status* messages to their children nodes. They also forward messages from their parents to their children nodes. If a child node does not receive any *update-status* or replaying messages from a parent node, it will assume that the parent node

has crashed and cannot participate in the network. The node, therefore, starts to look for a new parent node to join. It will repeat the same joining a tree process as we discussed above.

3.4.5. How data is disseminated through FaNet

When a node has data to share with other nodes in the network, it will send a copy of the data to its direct connected nodes, where the data is recursively relayed to other nodes until every node in the network receives a copy of it or it is told to stop. To avoid looping and abundant transmissions, the data is only forwarded to the nodes that it is not coming from. Additional fields are also added into the disseminating packages for these purposes. The goal of our forest-based dissemination scheme is rapidly spreading the data from the source nodes to nearby nodes in a specific area for local consumption. However, we want to control the distance that the disseminating data can reach out. Therefore, a time to live (ttl) attribute is added to every message before it is sent out. The field defines the dissemination radius so that stale data are not transmitted over the network. Data packages' ttl can be adjusted to meet its application time-constrained requirement. For those applications that are very time-sensitive like EEW system, ttl is set to be only a couple seconds.

When nodes receive a message from a parent or child node, they first check its ttl. They will ignore and not forward the message to other nodes if the message's ttl has been expired. To prevent relaying duplicated messages, a unique identification number (ID) is added to each original message. This allows nodes to identify and eliminate consuming and forwarding the data packages they have received. Each node also manages its own received message table. New disseminating messages will be saved while duplicated ones will be disregarded.

There are 4 types of dissemination that can happen in FaNet. The first type is when the root nodes initiate the dissemination. This happens when the root nodes want to share their own sharing with other local nodes or they are required to forward data from the remote server to other member nodes. To start the dissemination, the root nodes loop through its children and send a copy of the disseminating data to each of them. The children nodes could receive multiple copies of the disseminating data package because of multi primary parent structure. But each child node will only forward one copy of the disseminating data to its children. Duplicated data packages will be dropped. The same receiving and forwarding process will be repeated until the disseminating data gets to the leaf nodes or its ttl is expired.

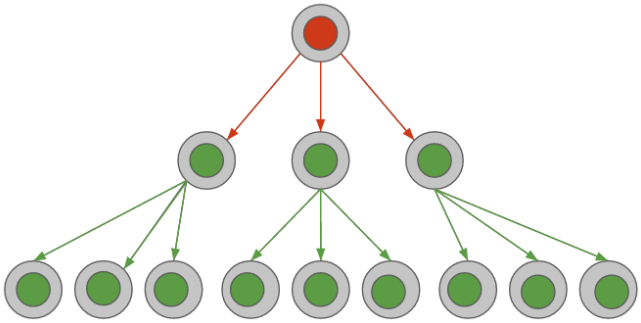


Figure 17: FaNet Root Nodes Initiate the Dissemination

The dissemination can be started from a leaf node when it has data to share with other member nodes within the tree it belongs to. Leaf nodes are the least busy nodes in the network. In most cases, they do not have to relay traffic for any node. Leaf node usually disseminate its

own data. When leaf nodes have data to disseminate, they will pack the data into a disseminating package and send a copy of it to each of its parents. The disseminating package is continued to relay to other nodes until it reaches to other leaf nodes or is expired.

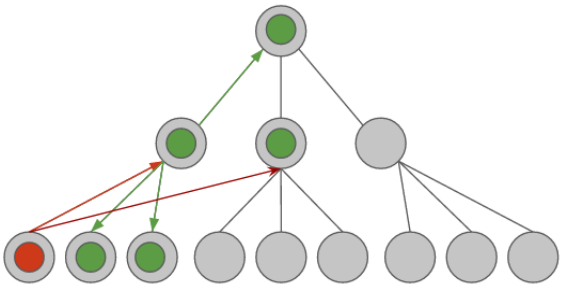


Figure 18: FaNet Leaf Nodes Initiate the Dissemination

Intermediate nodes also can initiate or carry on the dissemination process when they have data to share. The data is either generated by the nodes themselves or from their direct neighbors (parents and children). Same as leaf and root nodes, intermediate nodes only send the disseminating data to its parent and children whom the data did not come from. The disseminating package will also be dropped if the intermediate nodes figure out that it has been disseminated by themselves.

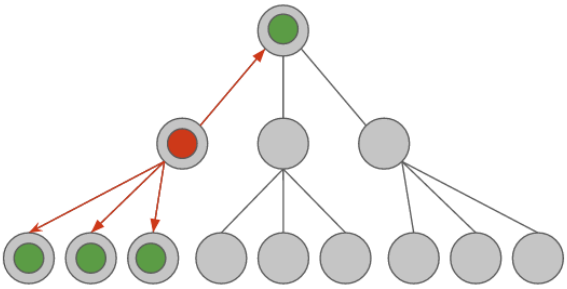


Figure 19: FaNet Intermediate Nodes Initiate the Dissemination

The most important Feature of FaNet dissemination scheme is that the nodes can share their data with not only other nodes within their primary tree but also the nearby tree in the forest as we depict in Figure *20.* The disseminating data can be propagated into a new tree when it is forwarded to a node which has multiple parents from different trees.

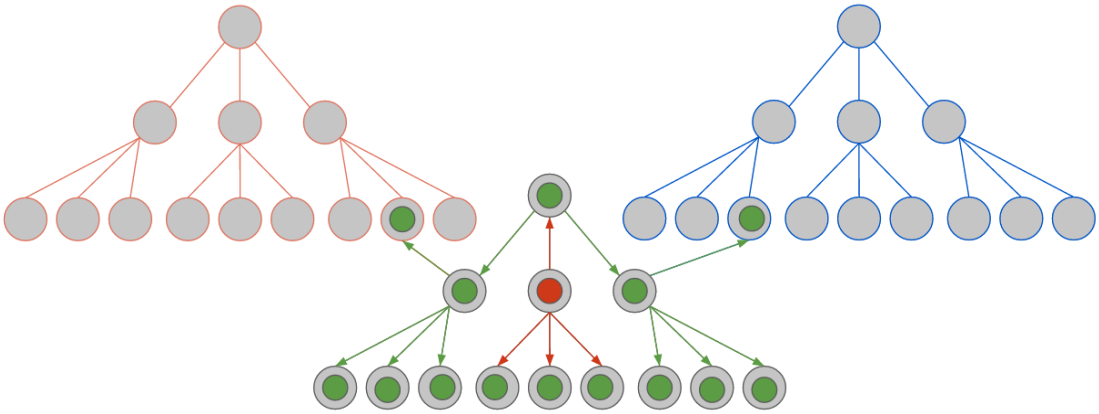


Figure 20: Data Dissemination Process in FaNet network

Chapter 4: Implementation

4.1. SCALE Network for Earthquake Detection and Early Warning Purpose

The driven inspiration of this paper is adding earthquake detection and early warning alert capacity for safe community alert network (SCALE), which is derived from an ongoing project sponsored by University of Irvine, Montgomery County and Global Cities Challenge Partners. The project seeks a sustainable platform in creating the safe living environment by using the advantages of networking, computing and sensing technology. SCALE's main goal is a low-cost multi-sensor network that can detect the hazard events such as house fire, gas leaking, illegal home invasion, flood and earthquake. The sensing data is collected and shared. Public service agencies such as Ambulance Dispatcher, Police, and Fire departments can subscribe the data to provide a better assistance and support to the community. Technical companies can also archive the data to develop the safety enhancement softwares, such as house intrusion notification or gas leaking alert.

Figure 21: Scale Ecosystem [17]

The current architecture of SCALE fulfills its primary objectives though it still has some significant limitations, especially when we want to exploit its infrastructure to detect earthquake and give its beneficiaries early warning. The reliability of the SCALE network becomes a problem when the internet or power supply of SCALE deployment region is disconnected due to violent ground shakes of a destructive earthquake. In these situations, we may lose a portion of the SCALE network in the affected region; its system becomes dysfunctional as a result -- which contradicts the main design objective of the SCALE project. Collaboration among the SCALE client nodes is a less critical problem in the client/server architecture, but it is very important in terms of the accuracy of the application outcome when we want to decentralize the earthquake detection process.

In the current design, each SCALE client sense and submit its sensing data to a remote SCALE message broker to be archived by other community safety enhance applications. The transmission is conducted through the Ethernet link. SCALE nodes do not collect and process sensing data on their end. They also do not have any mechanism to synchronize with other nodes to accomplish the same assigned task. Therefore, SCALE architecture is not suitable for earthquake detecting and early warning purposes. In order to allow SCALE system to have a functionalities of an EEW system, we need to redesign its network, the hardware and software of its clients and the remote server.

We propose to construct a hybrid decentralized EEW system on top of SCALE system to inherit its facilities and improve the weaknesses in detecting and alerting earthquake that the centralized approach has. To achieve this, SCALE client nodes still need to submit and receive data from SCALE server through the backbone internet. The SCALE server collects seismic data from SCALE clients and analyzes the data to find any possibility of an earthquake. At the same time, each SCALE clients are designed to collect and do its own detection at its deployment location. SCALE clients collaborate with each other to form a forest based wireless ad-hoc network for the area they are deployed so that they can gather more data from other member nodes for local detection. We let SCALE nodes to collect and share data through wireless ad-hoc network on local to avoid network interruptions as the consequences of destructive earthquakes. Wireless ad-hoc network also make it easy to deploy and scale the SCALE system.

4.2. SCALE Seismic Node Hardware Components

In order for SCALE nodes to serve as seismic nodes for our proposing hybrid decentralized EEW system, they must have an accelerometer to collect seismic signals, a radio transceiver to transmit and receive seismic data to and from other nodes and a processing unit to process raw seismic signal.

Ideally, we should build a seismic node from a personal computer (PC) or laptop because these devices are almost available at every house in most of developed countries. However, it is a waste of resource in terms of energy and money to run residential PCs as seismic nodes. Earthquake does not happen all the time but seismic nodes of a crowdsourcing EEW system must stay active and scrutinize any change from their surrounding environment to detect the occurrence of an earthquake. Our primary goal of this thesis is providing an affordable EEW system. It is important that the total cost of each seismic node is reduced while its performance is maintained. Board computers have been improved remarkably lately and shown that they are able to handle the same tasks as traditional PC with a much smaller cost. In our experiment, we choose The Raspberry Pi 2 Model B as the main processing unit for our seismic sensor node. At the time we wrote this page, it is the latest model of the Raspberry Pi Foundation, which is registered as an educational charity in UK. The Raspberry Pi costs $35 each but It has the capability of a decent desktop computer. It is powered with a ARMv7 multi core processor and 1 Gigabyte of RAM. I also have 4 USB ports, 40 GPIO pins, ethernet port and micro SD card lot. These features make the Raspberry Pi a perfect choice for the low-cost decentralized EEW system.

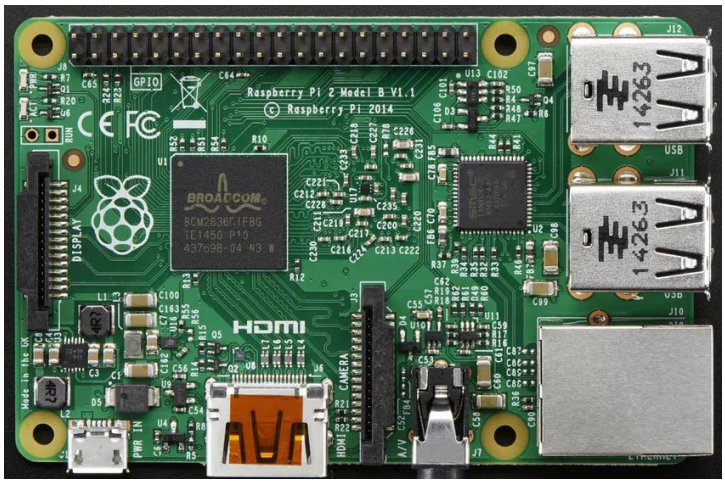


Figure 22: Raspberry pi [18]

Our goal is creating plug and play seismic sensor nodes. The end users should not do any additional configuration when installing the seismic node at their house so that we can encourage more people to join the EEW network. Furthermore, we also need deployed seismic nodes to be able to communicate with each other through wireless media to form an overlay mesh network for local collecting and sharing seismic data. For these purposes, we equip each Raspberry Pi with 1 wireless USB adapter. It is built with the latest wireless technology and comply with wireless IEEE 802.11b/g/n standards. It operates on universal radio wireless channel (1-14) with frequency band from 2.4 to 2.4835GHz. Its data transmission rate can reach up to 150Mbps. The radio frequency power of the wireless adapters can get up to 20 dBm which allows its radio signal to coverage up to 66 feet in door and 330 feet outdoor.

The most import piece of a seismic sensor node is its accelerometer. It is used to measure the acceleration in a given axis of the sensor node due to the ground movement of its surrounding environment. The measurement is reported in terms of g-force, which reflects the increments of the earth's standard gravitational acceleration (9.81 m/s^2). Phidget USB accelerometer has more strength in measuring the earth movement because it contains 3 axis accelerometers which make it capable of observing the change in the acceleration in all 3 spatial axes [19].

Figure 23: Phidgets USB Accelerator Sensor

4.3. SCALE Client Design

Figure 24 is a brief data flow diagram of the SCALE client software for seismic nodes of our proposing hybrid decentralized EEW system. Each SCALE client runs 5 separated threads to seek for potential seismic waves, filter out non seismic noise, store and share its collected seismic signal with other member nodes in its local network and a remote server, aggregate seismic data from other nodes with its own data to scan for potential earthquake and trigger alert if one is identified. Seismic client also notifies the remote server when it detected an earthquake.

As the same time, the remote server also does it down detection based on the data it has received from seismic client nodes. The detection result is used to verify the accuracy of the earthquake alerts coming from seismic clients. The remote server can confidently confirm that an earthquake is happening at a specific region when its own detection result matches with the majority of seismic nodes' report that it has heard from the region. When this happens, the remote server will send an alert to those seismic nodes that were failed to report earthquake incidence to it. This client/server communication logic ensures that an alert is actuated at every seismic node right after the first seismic waves are detected so that appropriate responses can be carried out by the time the destructive seismic waves arrive.

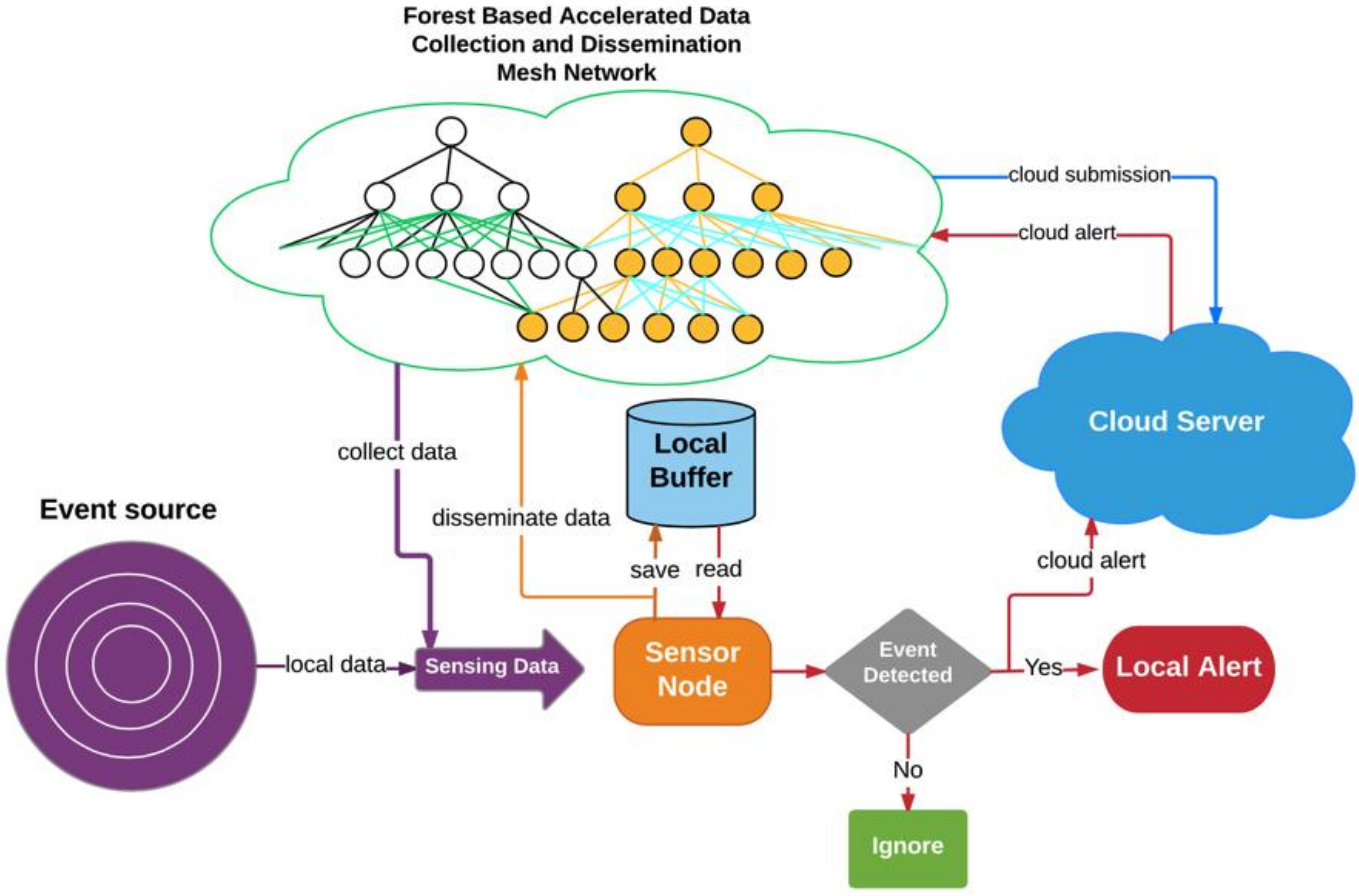


Figure 24: Detailed Seismic Client Design

4.4. Aggregated STA/LTA for Decentralized Detection Process

4.4.1. STA/LTA Algorithm

Earthquakes are rare events which do not have any specific patterns and are really hard to predict. Many approaches have been proposed in literature to detect the ground shakes based on seismic data. STA/LTA algorithm has been shown the most effective method to detect earthquake from seismic noise [10]. It has been implemented in many EEW systems such as EEWS and CSN network. STA/LTA is able to record the weak motions that other threshold triggering algorithms often fail to accomplish. STA/LTA continuously keeps track of the changes in the seismic noise amplitude in two moving time windows. The short time average (STA) window measures the average amplitude of seismic signal within a short time interval. STA value could be used to compare with an earthquake predefined threshold to watch for a possible earthquake [20]. However, using only STA value to declare an earthquake is not sufficient enough because the system cannot distinguish the difference between the actual seismic noise and the man-made noise. Therefore, long time average (LTA) window needs to be taken into consideration as well. LTA holds the average value of the seismic signal over a long time interval. The ratio STA/LTA is instead examined to identify the possibility of an earthquake.

Whenever a new wave of the seismic signal is coming, its absolute amplitude is measured before STA and LTA are recalculated. Finally, the ratio STA/LTA is compared with a predefined threshold. A channel trigger is declared if the ratio is greater than the threshold. A channel trigger cannot indicate that an earthquake is occurring and the system should respond immediately to it. Most of the seismic networks collect data from multiple channels for that decision through a voting algorithm. The trigger voting mechanism defines how many channels have to be in the “trigger” state in order to consider that the system is sensing real seismic waves. A channel is in the trigger state until the seismic signal starts to terminate gradually [20]. The channel de-triggers when the current ratio of STA/LTA falls below another user predefined threshold or STA/LTA de-trigger threshold.

Depending on the application, additional parameters such as pre-event time (PEM) and post-event time (PET) are also defined. PEM is the time interval that the application records the seismic data before the system enters the trigger active state. PET is the recording time interval after the system is de-triggered.

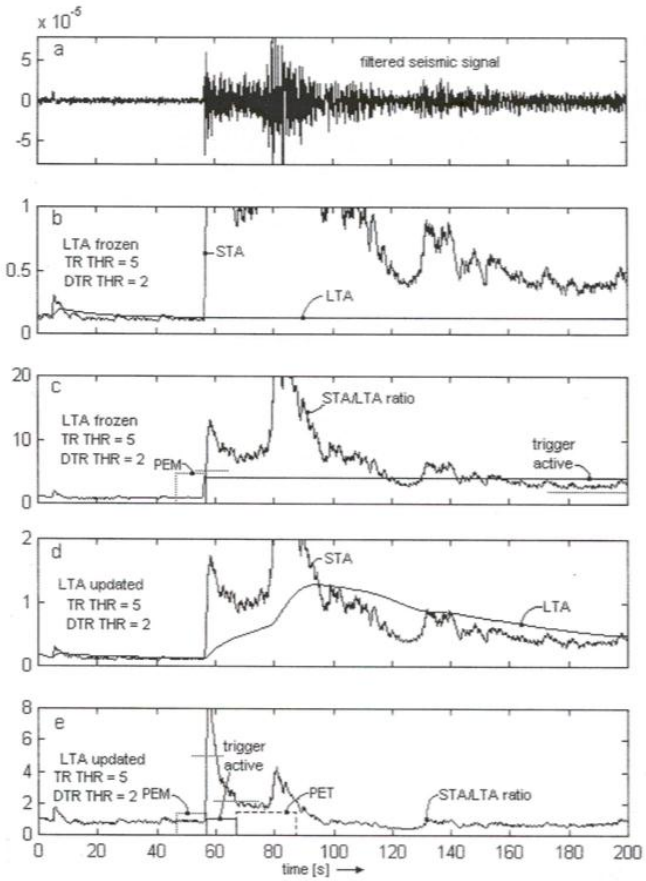


Figure 25: STA/LTA Seismic Waves Analysis [20]

4.4.2. Apply STA/LTA for FaNet System to Decentralize Earthquake Detection

In FaNet decentralized earthquake detection system, seismic nodes run their own STA/LTA algorithm and record seismic signals in terms of seismic picks that the application can digest. The stronger the shake is, the more number of picks a seismic node reports. Beside this, seismic nodes are also allowed to share their observed seismic picks with other nodes in local. Therefore, each node always has real-time seismic data that reflects the actual seismic noise within its deployment location. In the centralized architecture like CSN system we discuss above, an earthquake is detected when its central server recognizes the number of nodes that are reporting the seismic shakes has passed a threshold. In the decentralized approach, nodes have a better view of the state of local seismic noise. Therefore, they can make their own decisions on whether to declare an earthquake alert or not, assuming that sensor nodes are distributed uniformly over the target area with a proper density.

We figure out that there is a correlation between the local STA/LTA and the global STA/LTA ratios. If we can detect an earthquake at an individual seismic node by scrutinizing the STA/LTA ratio of the seismic wave it observes, then we can use the average value of STA/LTA ratios of all the seismic nodes within a location to detect earthquakes in that location. Assuming that all FaNet nodes run their own local STA/LTA algorithm to report earthquake in the form of digital picks or seismic picks. All nodes share their collected data with their neighbors through

FaNet. If $l_i(t)$ is the function that reflects the seismic picks in node i at time t, then the local seismic picks at time t in the node can be depicted as

$$g(t) = \frac{a_1 l_1(t) + a_2 . l_2(t) + \ldots + a_n l_n(t)}{n}$$

Equation 1: Short Term Average Change Seismic Picks of Peer Nodes

where n is the maximum numbers of neighbors the node can communicate. a_1, a_2… a_n are propagation delays in transmitting seismic picks from node 1, 2… n to the node. Because FaNet allows disseminating rapidly the data sharing over the local network, the propagation delay a_1, a_2…, and a_n are not significant. Nodes also only share their seismic data with nodes that a couple hops away, we therefore can consider that a_1, a_2…, and a_n are the same and $g(t)$ can be written as

$$g(t) = \frac{a}{n}[l_1(t) + l_2(t) + \ldots + l_n(t)]$$

Equation 2: Refactored Short Term Average Seismic Picks Change of Peer Nodes

When a real earthquake occurs, all sensor nodes seem likely to collect the same amount of seismic picks at any given time t.

$$l_1(t) \simeq l_2(t) \simeq \ldots \simeq ln(t) = l(t)$$
$$\Rightarrow g(t) \simeq a.l(t)$$
$$\Rightarrow \frac{g(t_2) - g(t_1)}{(t_2 - t_1)} \simeq \frac{l(t_2) - l(t_1)}{(t_2 - t_1)}$$
$$\Rightarrow \Delta g(t) \simeq \Delta l(t)$$

Equation 3: Short Term Average Seismic Picks of Peer Nodes Similar to that of Each Node

Therefore, if *l(t)* depicts the strength of the seismic noise at each individual node, $g(t)$presents the view of all local nodes toward the incident. Nodes can identify the occurrence of an earthquake at a specific location on the behalf of their local network by observing the total seismic picks of all neighbors that they have received over the network. To increase the accuracy of the detection, we apply the core concept of STA/LTA algorithm in digesting and interpreting the local collected seismic data for every node. Fundamentally, each node keeps track of the short and long time average of its neighbors' picks rate and its own pick rates. When the radio between the short-term average and long-term average is greater than the predefined earthquake trigger threshold at a specific node, it can consider that an earthquake is happening at its local area.

In order to achieve this goal, each node will construct a matrix to keep track of the seismic activities within its location. It is called Picks Collection Matrix which has the size of (n +1) x (m + 1). Table 1 presents the content of seismic picks each node has collected from its peers at time interval t. The value of n is identified by the routed value of LTI/STI ratio where STI is the short time interval in which seismic picks are collected and examined and SLI is the long time interval in which collected seismic picks are examined. m is the number of neighbors that have been sharing sensing data with the node. Each node will run Algorithm 4 to see if its collected data reflects a potential earthquake in its area. If we call sta_i and sla_i are the aggregated short term and long term change of seismic picks each peer node has at time interval t respectively, then these values can be achieved by the following formula:

$$sta_i = \frac{\sum_1^m l_m(t_i)}{m}$$

Equation 4: Simplified Short Time Average Seismic Pick Change of Peer Nodes

$$lta_i = \frac{\sum_1^m \frac{\sum_{i-n}^{i} l_m(t_i)}{n}}{m}$$

Equation 5: Simplified Long Term Average Seismic Pick Change of Peer Nodes

	t_{i-n}	$t_{i-(n-1)}$	...	t_{i-1}	t_i
Node 1	$l_1(t_{i-n})$	$l_1(t_{i-(n-1)})$		$l_1(t_{i-1})$	$l_1(t_i)$
Node 2	$l_2(t_{i-n})$	$l_2(t_{i-(n-1)})$		$l_2(t_{i-1})$	$l_2(t_i)$
...					
Node m-1	$l_{m-1}(t_{i-n})$	$l_{m-1}(t_{i-(n-1)})$		$l_{m-1}(t_{i-1})$	$l_{m-1}(t_i)$
Node m	$l_m(t_{i-n})$	$l_m(t_{i-(n-1)})$		$l_m(t_{i-1})$	$l_m(t_i)$

Table 1: Network Seismic Picks Collection Matrix

Algorithm 4: Aggregated STA/LTA for Individual Detection

```
STI = short time interval in which seismic picks are collected and examined
SLI = long time interval in which collected seismic picks are examined
Aggregated STA = average seismic picks each peer node has within STI
Aggregated SLA = average seismic picks each peer node has within SLI
clock = 0
While node is active do
        Gather seismic picks from other peer nodes
        Store collected seismic picks

        if clock > STI then
                calculate Aggregate STA
                calculate Aggregated Peer LTA

                If Aggregated STA/ Aggregated LTA > Earthquake_Threshold
                        disseminate suspecting-earthquake message
                end if
        else
                clock = 0;
        end if
end while
```

4.5. Passive Network Polling Algorithm to Increase Detection Accuracy

To increase the confidentiality of a local detection, it is essential to utilize the information that the node has collected from multiple sources. Like the way we usually use to confirm an earthquake in our daily lives. When we feel something like an earthquake, we often ask or observe the reaction of the people around us to see if it is actually an earthquake. The more people confirm that they have felt an earthquake, the more confident we are to say that an earthquake has occurred. In our proposing decentralized earthquake detection algorithm that we discussed in the above section, a seismic node believes that an earthquake is occurring when it figures out that its local aggregated average STA/LTA ratio has passed a defined earthquake threshold. When a node can verify that, it will disseminate a ballot message into its local network to vote yes for an ongoing earthquake polling topic. The message contains a binary value to indicate that an earthquake is suspected at the sender and identify an expiration time to prevent old votes being counted.

At the same time, each node actively listens to its parent and children nodes for earthquake ballot messages from other nodes in its area. It stores the voting messages in a polling result table, where the messages are grouped together based on their arrival time interval. Each time slot is chosen small enough so that the polling result within it can be announced by the time the S-wave hits the area. Each node creates a record in its polling result table for every node that it has been communicating with. When it receives a voting message from a node at the time interval t_i, it will consider that this node has voted YES for the ongoing earthquake polling. At the end of the time interval t_i, each node will count the number of YES ballots and the total number of neighbor nodes in the polling result table.

	t_{i-n}	$t_{i-(n-1)}$	...	t_{i-1}	t_i
Node 1	*YES*	*NO*		*NO*	*YES*
Node 2	*NO*	*NO*		*NO*	*YES*
...					
Node m-1	*NO*	*YES*		*YES*	*YES*
Node m	*NO*	*NO*		*NO*	*NO*

Table 2: Node Earthquake Polling Table

Chapter 5. Test Results and Analysis

5.1. Multicast vs. Forest-Based Data Dissemination Evaluation

We wrote a Math Lab simulator to compare the performance of sharing data through our FaNet scheme with the broadcasting approach. Our testing network topology includes 25 wireless sensor nodes which are deployed randomly within an area of 250-foot-wide by 250-foot-long. The wireless coverage range is up to 70 feet. Each node is set to generate one seismic data package at the same time. Each node then uses multicast and FaNet dissemination scheme, respectively, to diffuse its generated package to other nodes within the network. The data package's expiration time is also provided to determine how far it can travel from its source node in each case. To simplify the simulation, we consider that the number of hops a package can be relayed equals to its time to live (tt). Nodes will not forward the packages with an expired ttl.

To evaluate the overhead of FaNet dissemination scheme with the multicast approach, we assume that it costs each node 1 mAh to send 1 Mb data to another node. The power each node spent during the dissemination is recorded in each run for each dissemination scheme. We set our simulator to run 10 times with the package's expiration time varying from 1 to 10 hops. Each time, Nodes in the network apply 2 different dissemination techniques to diffuse its generated package: multicast dissemination and FaNet dissemination. After the simulation, we calculate the average packages that each node received, the average duplicated packages and the average number of the packages that nodes can use for earthquake detection. Because the EEW system is very time-sensitive, its member node cannot use the old seismic data for the detection. The collected data needs to be a couple seconds old; otherwise the outcome of the EEW application becomes less useful.

Figure 26 describes the network topology that all 25 testing sensor nodes formed when we run the simulation with the multicast data dissemination scheme. The black dots represent the sensor nodes. The blue dots indicate the location of the access point. The pink lines depict the wireless ad-hoc link between 2 nodes. The network does not have any specific structure. It is actually an unstructured mesh network in which each node builds and keeps track of a list of nodes within its wireless coverage range or neighbor nodes. Each node sends a copy of the disseminating data, either from itself or other node, to all of its direct neighbors. The disseminating data is multicasted again when it reaches to a nearby node until its ttl is expired. First look in the network, we see many unnecessary links are established. The network also contains many loops, which could cause the broadcast storms during the dissemination process.

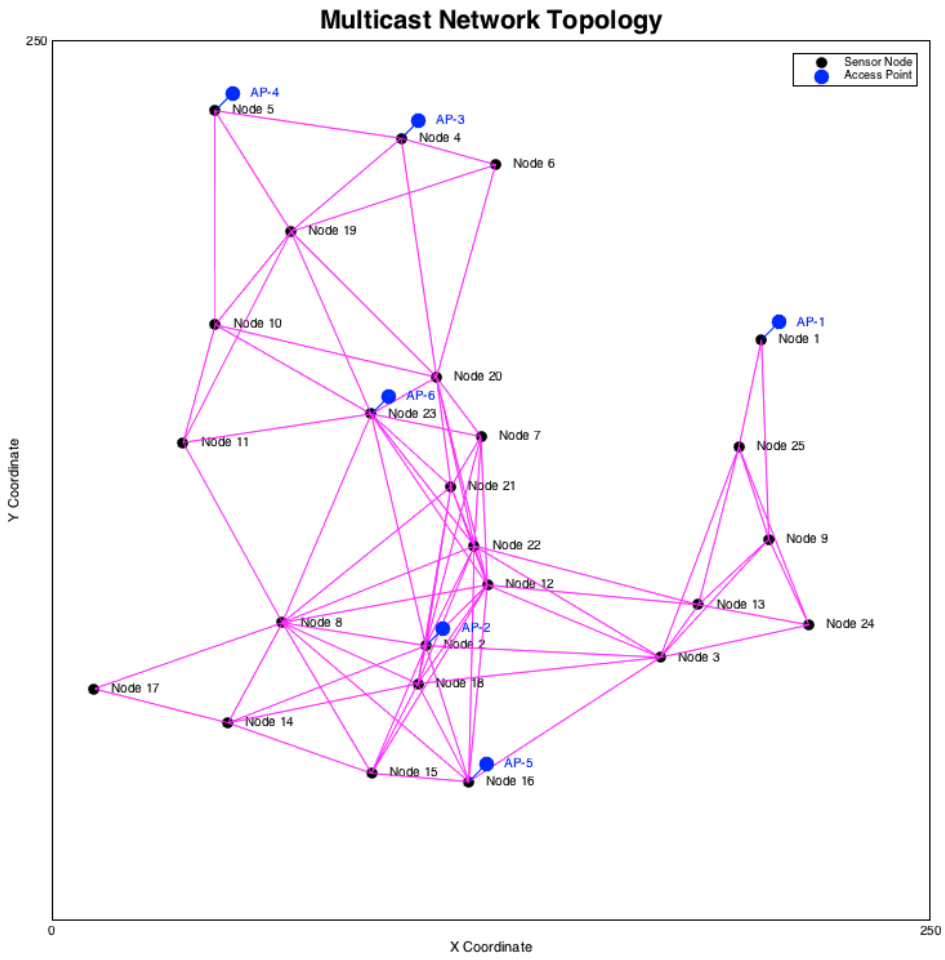


Figure 26: Multicast Network Topology

Figure 27 is the network's 25 testing sensor nodes formed when we applied FaNet algorithm for each of them. It is a forest-based network in which small trees are constructed and linked to each other. Unlike the network described in Figure 26, a minimal number of links between the nodes is established and maintained. Loops among nodes are also removed. The network now has new node type and link type. The red dots are the root node of the tree. The green link is the connection between trees. It plays as a bridge to transmit data from one tree to another tree in the forest.

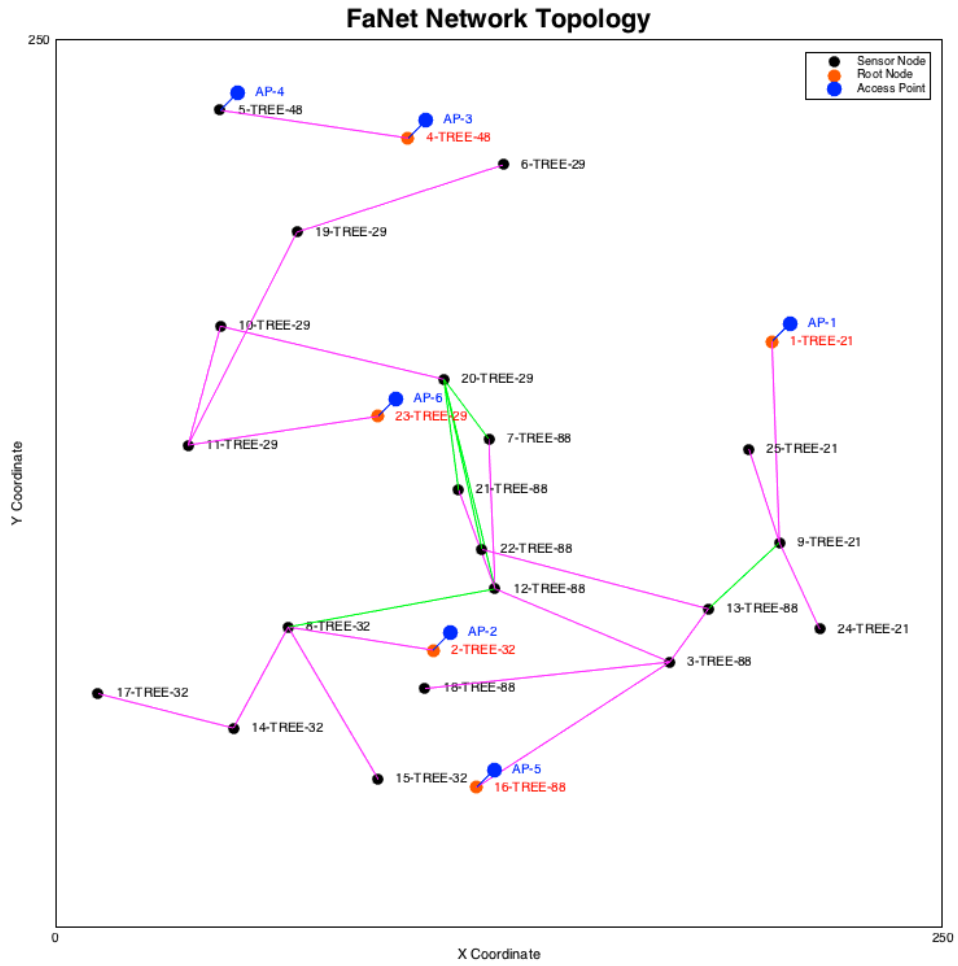


Figure 27: Forest-based (FaNet) Network Topology

In our simulation, we assume that the seismic nodes will sense the same shake when an earthquake strikes its deployment area. Therefore, they will generate and disseminate the same seismic data package. To compare the performance of the FaNet dissemination technique with multicast approach, we set the data package to be disseminated 5 hops away from its source node and calculate the total number of packages that all the nodes received in each dissemination scheme. We observe that nodes receive more packages when the original data is disseminated through multicast scheme. Though, almost 50% of received packages in multicast are duplicated and cannot be used for local detection. Meanwhile, nodes get less packages with FaNet dissemination mechanism but they contains high percentage of unique packages.

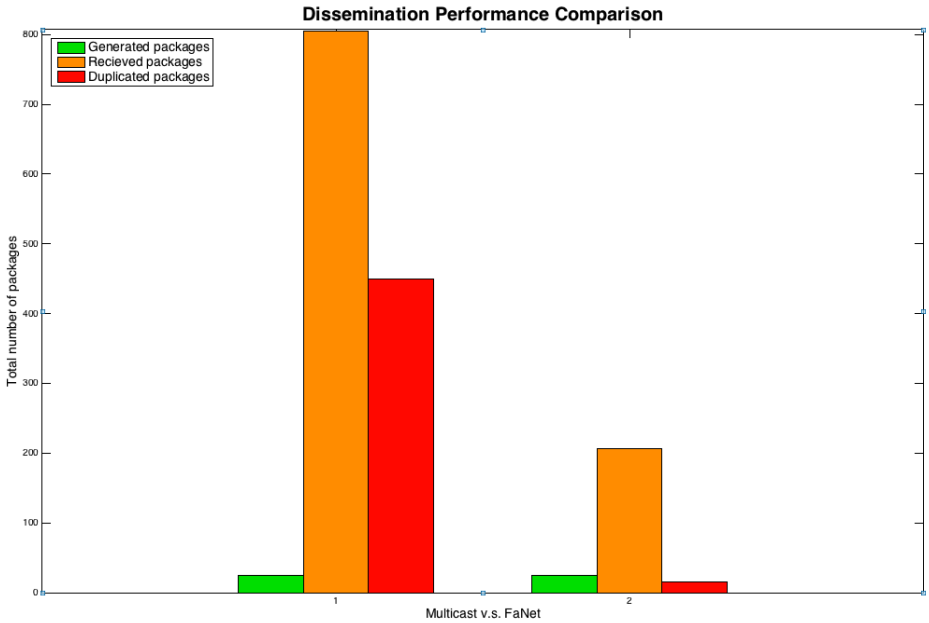


Figure 28: Total Dissemination Traffic between Multicast and FaNet

We try to find out if nodes can receive more disseminating data when we increase the radius of the dissemination, meaning that the disseminating data can travel farther away from its source node. We set the data package of each node to be disseminated up to 1 hop, 2 hops …, and 10 hops respectively. We then calculated the average received packages each node got in each dissemination radius when FaNet and Multicast technique are enforced. The results are plotted in the graph we show in Figure 29. We see that when the dissemination distance is getting larger, the average number of received packages each node got in multicast dissemination network is accelerated significantly. However, this value goes up slightly in FaNet.

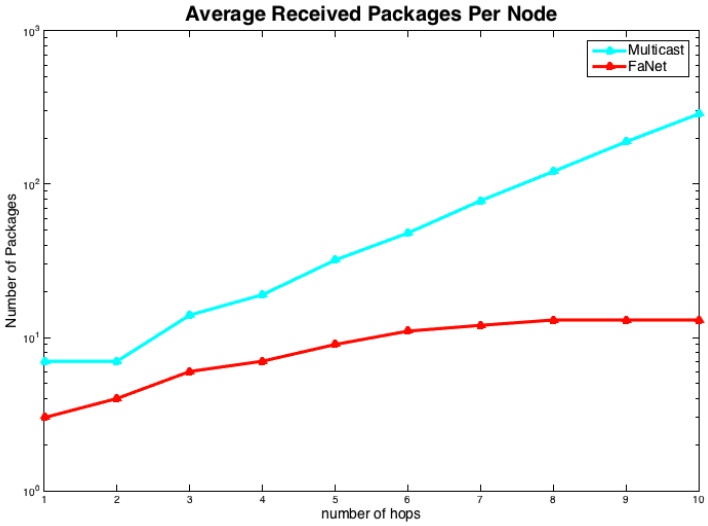


Figure 29: Delivery Capacity of FaNet v.s. Multicast Dissemination Scheme

First look, it seems that multicast dissemination scheme has better performance than our proposing mechanism when we want to increase the data diffusion area for each node. However, the average number of received packages that present in Figure 29 includes duplicated packages. Figure 30 shows that the increment in the total received packages per node in multicast dissemination is due to the escalation of duplication message. Because nodes in multicast do not know which direction the disseminating data came from, the disseminating data needs to be re-disseminated multiple times to reach to nodes that are more hops away from its source. Each time the disseminating data is multicasted, its previous relayers or senders could encounter it again. This explains why the number of duplicated packages of each multicast node raises rapidly when we expand the area in which it data can be spread out. When nodes organized in a forest based network and apply FaNet dissemination scheme to disseminate their own and other members' data, they only forward the disseminating data to nodes that are opposite direction of where the data came from. The dissemination strategy ensures that the disseminating data will not come back to previous disseminators. As a result, duplicated data occurs less with FaNet nodes and it does not increase much when the expiration time (number of hopes in our simulation) of each disseminating package increase.

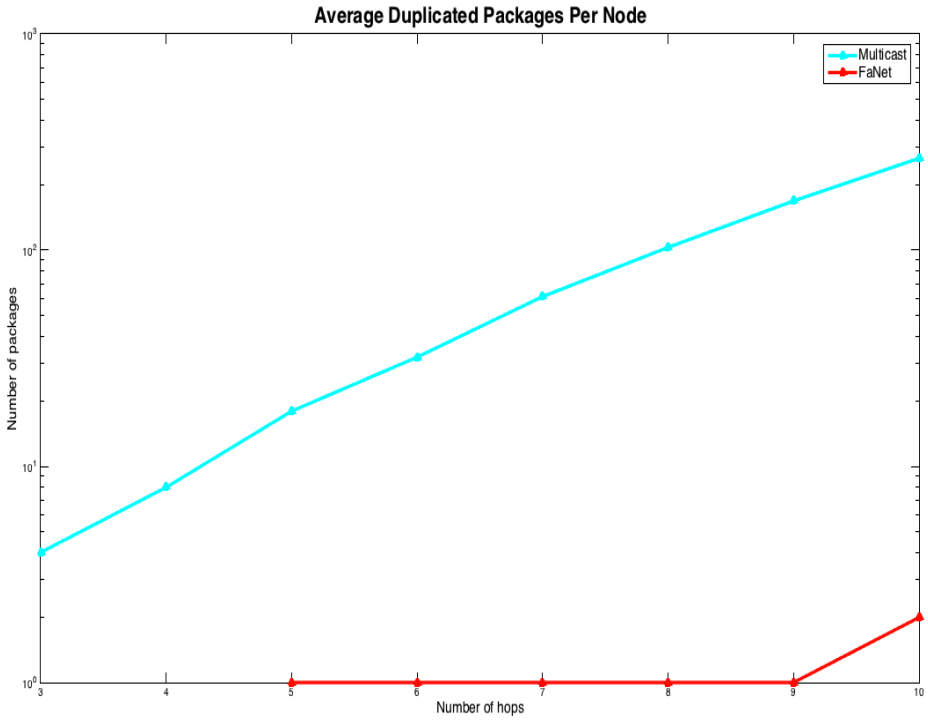


Figure 30: Duplicated Traffic of FaNet v.s. Multicast Dissemination Scheme

Besides being able to prevent transmitting unnecessary traffic in the network, our simulation result also reveals that FaNet nodes consume less system resources to disseminate data. Figure 31 shows that the average amount energy each node in FaNet spent to carry out the dissemination of itself and other member nodes is smaller than the amount it needed with multicast approach. But the big achievement of FaNet in term of network overhead is that its nodes do not use

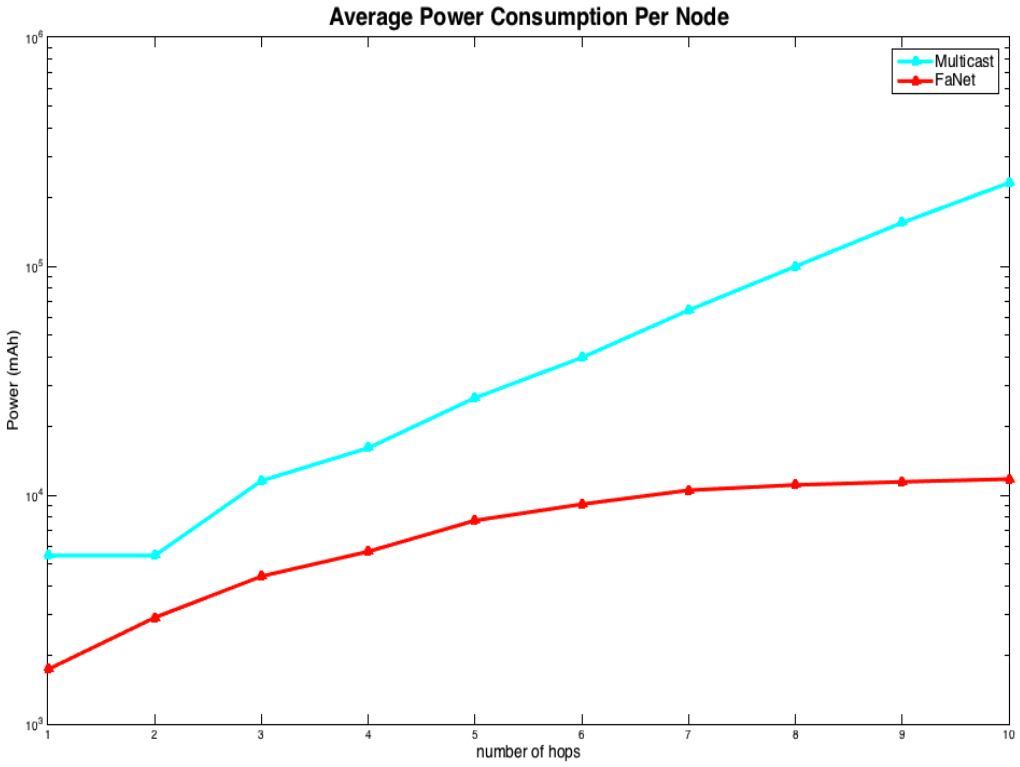


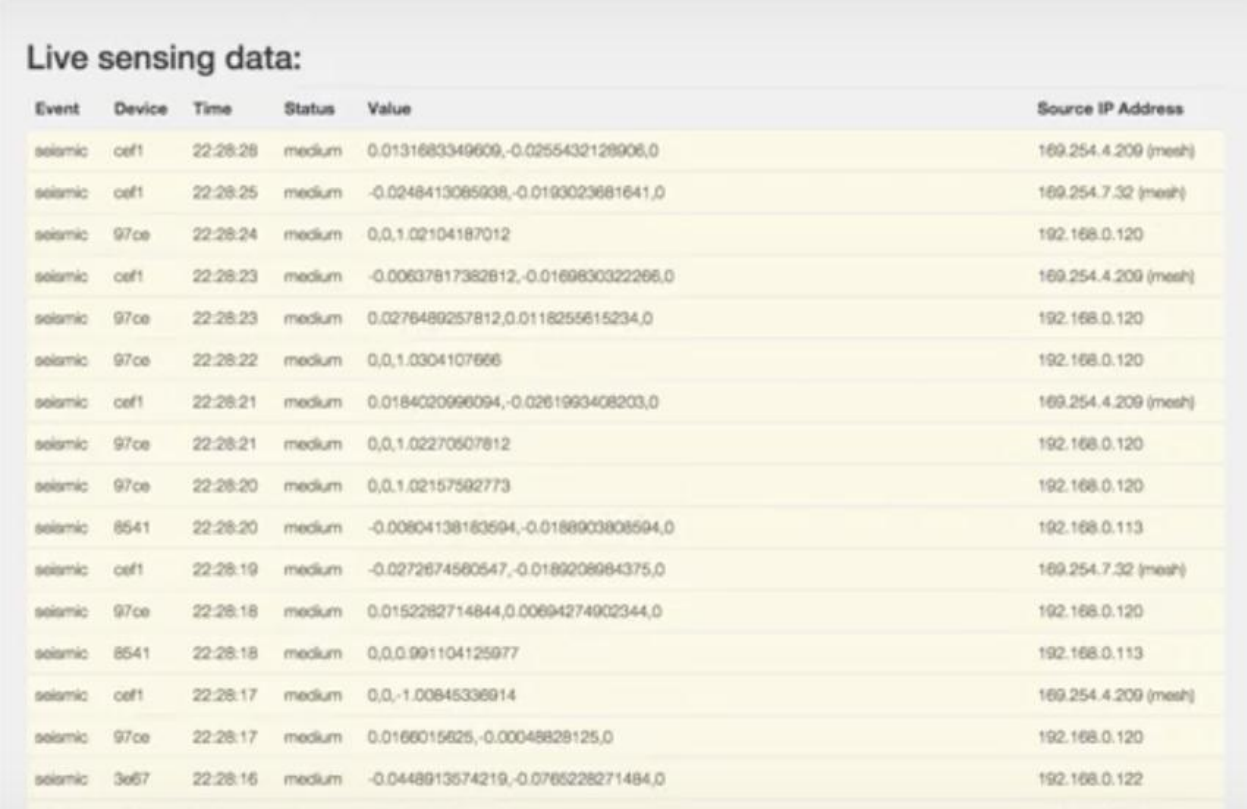
Figure 31: Disseminating Power Consumption of FaNet vs. Multicast Dissemination Scheme

5.2. Experiments on Data Transmission in Wireless Ad-hoc Network

For the proof of concept, we built a small EEW network with our proposing hardware and network design. On experiment EEW network has 4 seismic nodes as depicted in Figure 32. Each of them is configured to communicate with each other through wireless ad-hoc mode. They all initially are plugged into the internet. Three of them run a CSN client to collect seismic signals from their accelerator sensor and submit the data to a remote message queue broker. A simple html page with JavaScript is used to fetch the incoming data from our remote message queue broker and display it on a browser.

Figure 32: Experimental EEW Mesh Network Setting

We triggered an old Android Phone alarm that we set on vibrating modes to inject false seismic noises into testing seismic nodes and observed the streaming data from our experiment EEW system. We see that our message queue broker not only received seismic data directly from each seismic nodes through its Ethernet connection but also indirectly from its neighbor nodes. Figure 33 displays the seismic picks that came to our message queue broker when we ran the experiment. Each seismic data contains event type, seismic node's identification, the time event was detected by its source node, the source node Ip address and the value of the seismic signal. The source node Ip address can either be the Ethernet ip or the mesh ip address. The Ethernet address of each node is in 192.168.0.0/24 meanwhile the mesh ip address in 169.254.0.0/16.

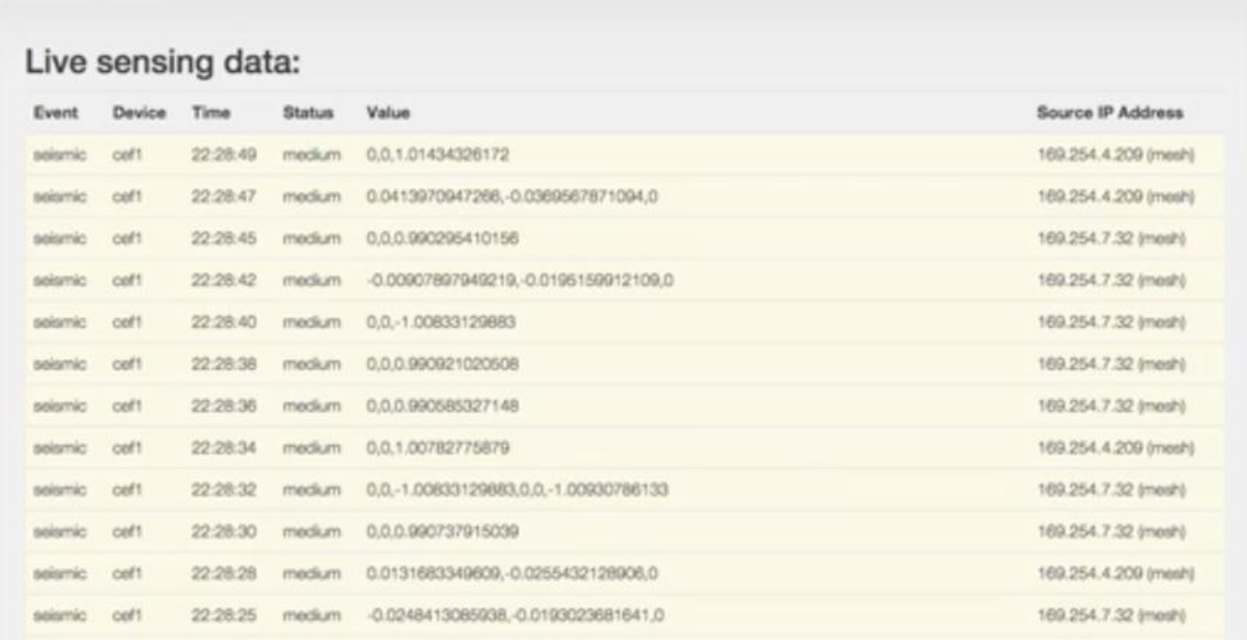


Live sensing data:

Event	Device	Time	Status	Value	Source IP Address
seismic	cef1	22:28:28	medium	0.0131683349609,-0.025543212890 6,0	169.254.4.209 (mesh)
seismic	cef1	22:28:25	medium	-0.0248413085938,-0.0193023681641,0	169.254.7.32 (mesh)
seismic	97ce	22:28:24	medium	0,0,1.02104187012	192.168.0.120
seismic	cef1	22:28:23	medium	-0.00637817382812,-0.016983032226 6,0	169.254.4.209 (mesh)
seismic	97ce	22:28:23	medium	0.027648925781 2,0.011825561523 4,0	192.168.0.120
seismic	97ce	22:28:22	medium	0,0,1.0304107666	192.168.0.120
seismic	cef1	22:28:21	medium	0.0184020996094,-0.026199340820 3,0	169.254.4.209 (mesh)
seismic	97ce	22:28:21	medium	0,0,1.02270507812	192.168.0.120
seismic	97ce	22:28:20	medium	0,0,1.02157592773	192.168.0.120
seismic	8541	22:28:20	medium	-0.0080413818359 4,-0.018890380859 4,0	192.168.0.113
seismic	cef1	22:28:19	medium	-0.0272674560547,-0.018920898437 5,0	169.254.7.32 (mesh)
seismic	97ce	22:28:18	medium	0.0152282714844,0.006942749023 44,0	192.168.0.120
seismic	8541	22:28:18	medium	0,0,0.991104125977	192.168.0.113
seismic	cef1	22:28:17	medium	0,0,-1.00845336914	169.254.4.209 (mesh)
seismic	97ce	22:28:17	medium	0.0166015625,-0.00048828125,0	192.168.0.120
seismic	3e67	22:28:16	medium	-0.0448913574219,-0.076522827148 4,0	192.168.0.122

Figure 33: Live Seismic Data from Experimental EEW Mesh Network without Internet Interruption

After we let our experiment ran for 3 minutes in which all 4 nodes were plugged in the Ethernet. We then unplugged the Ethernet cable from 3 seismic nodes that was equipped with an accelerator sensor. Observing our experimental dashboard as presenting here in Figure 34, we saw that seismic data were still coming to our message queue broker. However, its source node ip address was mesh ip address, which indicated that it was submitted to our message queue broker through its source node's mesh link.



Live sensing data:

Event	Device	Time	Status	Value	Source IP Address
seismic	cef1	22:28:49	medium	0,0,1.01434326172	169.254.4.209 (mesh)
seismic	cef1	22:28:47	medium	0.0413970947266,-0.0369567871094,0	169.254.4.209 (mesh)
seismic	cef1	22:28:45	medium	0,0,0.990295410156	169.254.7.32 (mesh)
seismic	cef1	22:28:42	medium	-0.00907897949219,-0.0195159912109,0	169.254.7.32 (mesh)
seismic	cef1	22:28:40	medium	0,0,-1.00833129883	169.254.7.32 (mesh)
seismic	cef1	22:28:38	medium	0,0,0.990921020508	169.254.7.32 (mesh)
seismic	cef1	22:28:36	medium	0,0,0.990585327148	169.254.7.32 (mesh)
seismic	cef1	22:28:34	medium	0,0,1.00782775879	169.254.4.209 (mesh)
seismic	cef1	22:28:32	medium	0,0,-1.00833129883,0,0,-1.00930786133	169.254.7.32 (mesh)
seismic	cef1	22:28:30	medium	0,0,0.990737915039	169.254.7.32 (mesh)
seismic	cef1	22:28:28	medium	0.0131683349609,-0.0255432128906,0	169.254.4.209 (mesh)
seismic	cef1	22:28:25	medium	-0.0248413085938,-0.0193023681641,0	169.254.7.32 (mesh)

Figure 34: Live Seismic Picks from Experimental EEW Mesh Network with Internet Interruption

Chapter 6. Conclusion and Future work

6.1. Conclusion

It is possible to construct and operate a low-cost EEW system through the crowdsourcing approach. It is also practical to increase the reliability and efficiency of the affordable system by localizing the detection. Unlike the conventional EEW systems in which the supper seismic stations or a set of intermediate seismic nodes is used for earthquake detection and early warning purpose, our proposing EEW system contains a large number of small nodes that can detect earthquake independently. The capacity is achieved through a resilient overlay wireless ad-hoc network and a lightweight data dissemination scheme. They enable the seismic nodes to share their sensing data with other members in their local network so that they can collaborate and assist each other to accomplish their earthquake detection and early alert mission. Because our proposing system is a highly time-sensitive application, sharing data is forwarded to other nodes along a forest-based structure to ensure that it can get to its consumers as fast as possible. The forest-based network is also built so that the common issues of the wireless networks, such as node failure, link failure and looping, are well addressed. The EEW architecture is also designed to be scalable and cost effective. We hope that the design can provide a reliable earthquake detection and alert system that can be deployed to any community which has a high risk of having big quakes.

6.2. Future works

Due to the scope of this project, we have not expanded the forest-based data dissemination technique to work in 3 dimensional networks, which can be seen when we want to deploy an EEW system for commercial buildings or bridges. In these deployment scenarios, seismic data is disseminated not only horizontally but vertically also. In the current design, we assume that all the nodes are in the same planar surface. Therefore, nodes in the same area seem likely to receive the same seismic shakes and the aggregated short average value of the seismic wave amplitude from all the nodes over its long time average can be used to detect earthquakes. We need a different algorithm or maybe even different network structure for the EEW nodes to collect and analyze the data to study for potential earthquakes.

We propose our Decentralized EEW to be working reliably in the pre-, during and post-earthquake. The system has no issues with the power supply in the pre-earthquake and when the earthquake has just occurred. A earthquake of a magnitude of 7.5 or above could damage badly the infrastructure of the region it went through. Consequently, the EEW nodes in the earthquake region could experience the power outage issue in the pre-earthquake. We can resolve the problem by equipping a rechargeable battery for each seismic node and switch it to run on battery mode when the electricity supply is gone. Nevertheless, this change in the design introduce a new challenge. Our proposing dissemination schema and decentralized earthquake detection mechanism based on the fact that the nodes have plenty of energy to handle its duty. Some optimizations need to be added to our design so that we can prolong the EEW network's lifetime when it operates in the power-constraint conditions.

Another improvement we can make for FaNet dissemination scheme is reducing the number of the disseminating packages in the network. All the nodes in their area do not need to bump their seismic data into the network to be shared with other node. Study in [24, 25] shows

that rare events can be detected by small amount of nodes. For this reason, we can pick a small subset of nodes in the area to share their seismic data with all other nodes in the network. Nodes can synchronize with each other to come up with a dissemination plan in which some nodes can stay out the process.

Although we attempt to propose a lightweight dissemination scheme, we have not achieved our objective completely. Nodes in our proposing EEW network still receive duplicated messages due to multi fan-in multi fan-out architecture. The overhead of the dissemination process could be smaller if we have a better method to reduce the amount of duplicated packages.

Last but not least, we have not fully implemented our decentralized EEW design for the SCALE system. We only evaluate our proposing dissemination scheme through our Math Lab simulation. In order to verify the accuracy of our new design, we need to test its performance in a real network in which the seismic nodes are installed in a real community. Seismic nodes may behave differently under the interferences of other noises from a real world environment. Also, we need to run our proposing decentralized earthquake detection technique against a data set that was collected from an identified earthquake to prove that it can detect the earthquake successfully.

Preferences

[1] “April 2015 Nepal earthquake,” https://en.wikipedia.org/wiki/April_2015_Nepal_earthquake, Dec 2015.

[2] D.D. Given, E.S. Cochran, T. Heaton, E. Hauksson, R. Allen, P. Hellweg, J. Vidale, and P. Bodin, “Technical Implementation Plan for the ShakeAlert Production System—An Earthquake Early Warning System for the West Coast of the United States.” USGS 2014, “http://pubs.usgs.gov/of/2014/1097/pdf/ofr2014-1097.pdf”, Apri 2015.

[3] “San Andreas Fault”, “https://en.wikipedia.org/wiki/San_Andreas_Fault”, May 2015.

[4] “Earthquake”, “https://en.wikipedia.org/wiki/Earthquake”, Apri 2015.

[5] Stephen A. Nelson, “http://www.tulane.edu/~sanelson/Natural_Disasters/eqcauses.htm”, Dec 2015.

[6] Yutaka Nakamura, “UrEDAS, Urgent Earthquake Detection and Alarm System,” 13th World Conference on Earthquake Engineering Vancouver, B.C., Canada August 1-6, 2004 Paper No. 908.

[7] Yutaka Nakamura, Jun Saita, “13 UrEDAS, the Earthquake Warning System: Today and Tomorrow.” “http://www.sdr.co.jp/papers/13.pdf”, Oct 2015.

[8] Matthew Faulkner, Annie H. Liu, Andreas Krause, “A Fresh Perspective: Learning to Sparsify for Detection in Massive Noisy Sensor Networks.”

[9] Yi Yuan, Dan Wang and Yi-Qing Ni, “Minimum Cost Deployment in Earthquake Early Warning System for High-Speed Railways.”

[10] Robert W. Clayton, Thomas Heaton, Mani Chandy, Andreas Krause, Monica Kohler, Julian Bunn, Richard Guy, Michael Olson, Mathew Faulkner, MingHei Cheng, Leif Strand, Rishi Chandy, Daniel Obenshain, Annie Liu, Michael Aivazis, “Community Seismic Network,” Annals of Geophysics, page 54, June 2011.

[11] Al-Karaki, J.N. and Kamal, A.E., “Routing Techniques in Wireless Sensor Networks: A Survey,” IEEE Wireless Communications, 2004, pp 2-28.

[12] Villalba, Luis Javier García, Orozco, Ana Lucila Sandoval, Cabrera, Alicia Triviño, Abbas, Cláudia Jacy Barenco, “Routing protocols in wireless sensor networks,” Sensors (Basel, Switzerland), 2009, pp 8399-421.

[13]Jens Nachtigall and Jens-Peter Redlich, “Wireless Alarming and Routing Protocol for Earthquake Early Warning Systems.”

[14] Richard E. Pattis, “Special Trees,” “http://www.cs.cmu.edu/~pattis/15-1XX/15-200/lectures/specialtrees/”, May 2015

[15] Antoine Bagula, “Applications of Wireless Sensors Network,” February 2012.

[16] Kyungbaek Kim, Sharad Mehrotra, and Nalini Venkatasubramanian, “FaReCast: Fast, Reliable Application Layer Multicast for Flash Dissemination”.

[17] Nalini Venkatasubramanian, “SCALE System Design Poster”, 2015.

[18] “Raspberry Pi 2 - Model B - ARMv7 with 1G RAM,” “http://www.adafruit.com/products/2358”, May 2015

[19] “1049 User Guide,” “http://www.phidgets.com/docs/1049_User_Guide,” May, 2015.

[20] Amadej Trnkoczy “Understanding and parameter setting of STA/LTA trigger algorithm”, September 1999.