Service Discovery in Wireless Mesh Networks

DISSERTATION

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF TECHNOLOGY IN INFORMATION TECHNOLOGY

(WIRELESS COMMUNICATION & COMPUTING)

Under the Supervision of
Dr. Shekhar Verma
Asst. Professor
Mr. Vijay Chaurasiya
Lecturer
IIIT Allahabad

Submitted By
Nihara Patil
M. Tech. IT (WCC),
Fourth Semester,
MW200508

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY
ALLAHABAD, U.P., 211012

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY

Allahabad

(Deemed University)

(A Centre of Excellence in Information Technology Established by Govt. of India)

WE DO HEREBY RECOMMEND THAT THE THESIS WORK DONE UNDER OUR SUPERVISION BY NIHARA PATIL (MW200508) ENTITLED "SERVICE DISCOVERY IN WIRELESS MESH NETWORKS" BE ACCEPTED IN PARTIAL FULFILMENT OF THE REQUIREMENT OF THE DEGREE OF MASTER OF TECHNOLOGY IN INFORMATION TECHNOLOGY PROGRAM SPECIALIZATION IN WIRELESS COMMUNICATIONS AND COMPUTING.

COUNTERSIGNED

Prof. Shekhar Verma
THESIS ADVISOR

DEAN (ACADEMIC)

Mr Vijay Kr. Chaurasia
THESIS ADVISOR

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY

Allahabad

(Deemed University)

(A Centre of Excellence in Information Technology Established by Govt. of India)

CERTIFICATE OF APPROVAL*

The foregoing thesis is hereby approved as a creditable study in the area of Information Technology carried out and presented in a manner satisfactory to warrant its acceptance as a pre-requisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein but approve the thesis only for the purpose for which it is submitted.

COMMITTEE ON
FINAL EXAMINATION
FOR EVALUATION
OF THE THESIS

_____

_____

_____

*Only in case the recommendation is concurred in

Abstract

Service Discovery in Wireless Mesh Networks

By

Nihara Patil

Wireless Mesh networks have received a lot of attention lately. They offer the flexibility of wireless access, combined with a high coverage area; they also offer communication between heterogeneous domains. Wireless mesh networks (WMNs) consist of mesh routers and mesh clients, where mesh routers have minimal mobility and form the backbone of WMNs. They provide network access for both mesh and conventional clients. The integration of WMNs with other networks such as the Internet, cellular, IEEE 802.11, IEEE 802.15, IEEE 802.16, sensor networks, etc., can be accomplished through the gateway and bridging functions in the mesh routers.

For accessing services which don't lie in the same domain as the user, it must be able to interface and co-ordinate with its surroundings domains without the user's intervention. For this to happen, the service discovery protocol must be able to discover remote resources and use them. Thus, service discovery is the ability to discover the services in the same domain and the neighboring domains without explicit user direction. The method of Service Discovery in Wireless Mesh Networks will be slightly different than in fixed networks and in mobile ad-hoc networks.

In self-organized mobile networks, the need to know identifiers in order to establish connections is a burden to the users. The focus can be shifted from the nodes to the services they provide. Service discovery enables simple browsing and location of services available in the network and can offer support to the creation and advertisement of user communities.

Acknowledgements

I would like to thank my advisor, Mr. Vijay Kr Chaurasia, for his support and subtle guidance. His updated knowledge on the various wireless technologies helped me in the basic structuring of my work. He always helped in getting any resources needed in the initial stages, when things were required the most and I had a vague direction towards my thesis progress.

I am also very grateful for the advice and support from my advisor, Dr Shekar Verma, for his frank opinion and for motivating me in the most difficult phase of my thesis work. His guidance in the final stage was invaluable. His suggestions were very useful and helped me reach the completion of my thesis.

I would also like to thank my batch mate Sachin Kanaujia for his suggestions and help during the thesis. His clear views always gave me a new approach to ponder upon. I want to thank my parents for helping me in always keeping focused on my studies and for the supports when needed.

Lastly all the thanks belong to the Almighty.

Table of Contents

List of Figures

1

Introduction

1.1 Background

Wireless LAN (WLAN) Technology is currently experiencing tremendous growth in popularity, offering secure, seamless mobile access into corporate environments, homes and residential areas, and public spaces. WLAN technology is not new; however, with the increasing awareness of the need for security, the advanced adoption rate of mobile user devices such as PDAs, cellular phones and WLAN-enabled laptops, and the availability of new interactive applications, WLANs are becoming a common option [2]. As a realization among enterprises for the value mobility has come, WLAN technology has come up as a mainstream technology and as a strategic, integrated platform. From the view of service providers, WLAN technology provides a new business and revenue opportunities to cater to the need of always on, anytime, anywhere access required by their subscribers [13].

Wireless Mesh Network solution offers a different solution that can be deployed as an integrative solution to existing infrastructure to extend and expand WLAN access beyond traditional hotspot areas, enhancing coverage and offering seamless mobility [6].

The Applications that have increased with growth in wireless technology [1]:

- Broadband home networking
- Community networking
- Enterprise networking

- Metropolitan area networks
- Transportation systems
- Peer to peer applications

These aim at providing seamless communications between heterogeneous devices and heterogeneous networks; they have the capability of integrating seamlessly with its surroundings, to form a ubiquitous cooperative network [5]. A requirement has emerged, for a Technology that can make these Applications cheaper. The Technology must be compatible with the existing solutions.

Few scenarios and applications where wireless mesh technology is more suitable and can act as a more versatile or feasible solution than other technologies wired or wireless include, but are not limited to, the following [6]:

• Areas where extensive coverage is required, like, offices, on-campus networking, stadiums, or spanning a Sprawling facility

• Areas where up till now no wired facility is there or which are, under-wired, or difficult to wire, such as highways, conduits, or rural areas.

• in emergencies like fire fighting, disaster recovery, and military operations.

With increase in Applications of Wireless Mesh Networks a need arises for a Service Discovery Mechanism typically suited for them. As the Service Discovery allows a device to seamlessly communicate with its environments, device mobility and the heterogeneous nature doesn't hinder its utilizing the network resources and services [5].

Wireless Mesh Networks (WMNs) are dynamically self-organized and self-configured wireless systems, consisting of two main device categories: mesh clients and mesh routers. The later have a minimal mobility and they form a backbone of WMN [1]. Because of their advantages over other wireless networks, WMNs are emerging as an optimal solution showing rapid progress and inspiring many applications. The possible areas of application of mesh networks include broadband home networking, community networking, enterprise networking, metropolitan area networks, transportation systems, Peer-to-Peer applications and others [1].

1.2 Problem

As the purpose of Wireless mesh networks are that they are dynamically self-organized and self-configured wireless systems, and consisting of mesh clients and mesh routers, several issues exist which are not in other networks [3]. Service discovery systems aim at seamless discovery of resources and services that a network provides. The main idea is that users or other programs can locate resources, services and files efficiently from the known network [5]. A Service Discovery mechanism for wireless mesh network must address all specific needs and satisfy the criteria to work in a network with heterogeneity along with being effective.

1.3 Goals and Scope

To propose a Service Discovery mechanism that can work with WMNs .As the application scenarios for WMN are numerous, the Service Discovery Mechanism must support that. Also it must work well with conventional clients. WMNs comprises of Mesh Clients and Mesh Routers. Mesh Routers are generally fixed [1]. I propose to make the maximum use of the Mesh Router's stability in Service Discovery procedure. Heterogeneity of the networks is supported by Mesh Networks [1]. My Mechanism mainly applies to scenarios when multiple WLANs are joined together to extend the network. The WLANs may or may not be using the same radio technologies.

2

Wireless Mesh Networks

2.1 What are Wireless Mesh Networks?

Wireless mesh networks (WMNs) are dynamically self-organized and self-configured, with the nodes in the network automatically establishing an ad hoc network and maintaining the mesh connectivity [1]. WMNs are comprised of two types of nodes:

(1) Mesh routers and

(2) Mesh clients.

Along with the routing capability for gateway/bridge functions existing in a conventional wireless router, a mesh router supports additional routing functions to provide a platform for mesh networking. Using multi-hop communications, the coverage can be extended by a mesh router with much lower transmission power requirements [19]. To further enhance the adaptability of mesh networking, a mesh router is normally equipped with multiple wireless interfaces built on either the same or different wireless access technologies [1].

Mesh and conventional wireless routers are usually built based on a same hardware platform in spite of all the differences between them. Mesh routers generally have minimal mobility and their purpose is basically formation of mesh backbone for the mesh clients [1]. Even though mesh clients can also work as a router, the hardware platform and software for them can be made simpler than those for mesh routers. For instance, communication protocols for mesh clients can be light-weight, as gateway or bridge functions are not needed by mesh clients, only a single wireless interface is often needed in a mesh client [3].

The gateway/bridge functionalities in mesh routers enable the integration of WMNs with various other networks. Wireless mesh routers enable conventional nodes equipped with wireless network interface cards (NICs) to connect directly to WMNs [1]. Ethernet can be used to access WMNs by connecting to wireless mesh routers when wireless NICs are not available. WMN caters to the need of the users to be always on line anywhere, anytime [13]. Instead of being another type of ad-hoc networking, WMNs diversify and enhance the capabilities of ad-hoc networks .The differences between WMNs and MANETs are stated in section 2.6.

In many ways WMNs have become preferable over MANETs, they have advantages such as low installation costs, easy network maintenance, robustness, service coverage that can be relied on, etc [4]. Today, WMNs are a widely accepted technology in the traditional application areas of ad hoc networks, and they are also undergoing rapid commercialization application scenarios such as broadband home networking, community networking, building automation, high-speed metropolitan area networks, and enterprise networking etc [1].

2.2 WMN Architectures

2.2.1 Infrastructure/Backbone WMNs.

This type of WMNs consists of mesh routers for forming an infrastructure for clients that connect to them. The clients can be Mesh clients or Conventional clients. The WMN infrastructure/ backbone may be built using various types of radio technologies, in addition to the mostly used IEEE 802.11 technologies (as shown in Fig 1 [1]). The mesh routers form a mesh of self-configuring, self-healing links among themselves, which acts as a backbone. The gateway functionality enables mesh routers to connect to the Internet [1].

This approach, also referred to as infrastructure meshing, provides backbone for conventional clients and its main purpose is integration of WMNs with existing wireless networks, which is achieved through gateway/bridge functionalities present in mesh routers. Conventional clients with Ethernet interface are connected to mesh routers via Ethernet links, so the mesh routers have multiple interfaces. The

conventional clients with the same radio technologies as mesh routers can directly communicate with mesh routers and don't need any Ethernet link between them [1].

When different radio technologies are used, clients must communicate with the base stations that have Ethernet connections to mesh routers. Infrastructure/Backbone WMNs are the most commonly used type because of its ease of deployment.

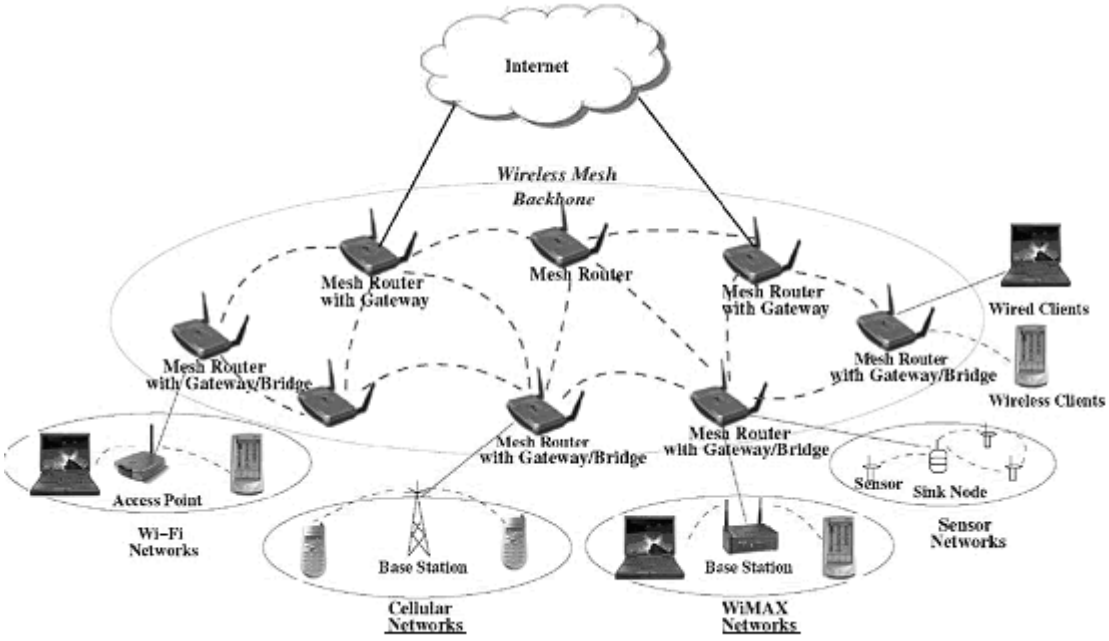


FIG: 1 Infrastructure/Backbone Mesh Network

2.2.2 Client WMNs.

Client WMNs (FIG 2 [1]) have almost the same characteristics as a Mobile Ad-Hoc Network. Client meshing is used to enable peer-to-peer networking among client devices. In this type of architecture, client nodes constitute the actual network to perform routing and configuration functionalities as well as providing end-user applications to customers. This type of WMNs does not constitute mesh routers. In Client WMNs, a packet destined to a node in the network hops through multiple nodes to reach the destination [1]. Client WMNs are formed using one type of radios

on devices. Moreover, the requirements on end-user devices are increased due to the added routing and configuration functions.

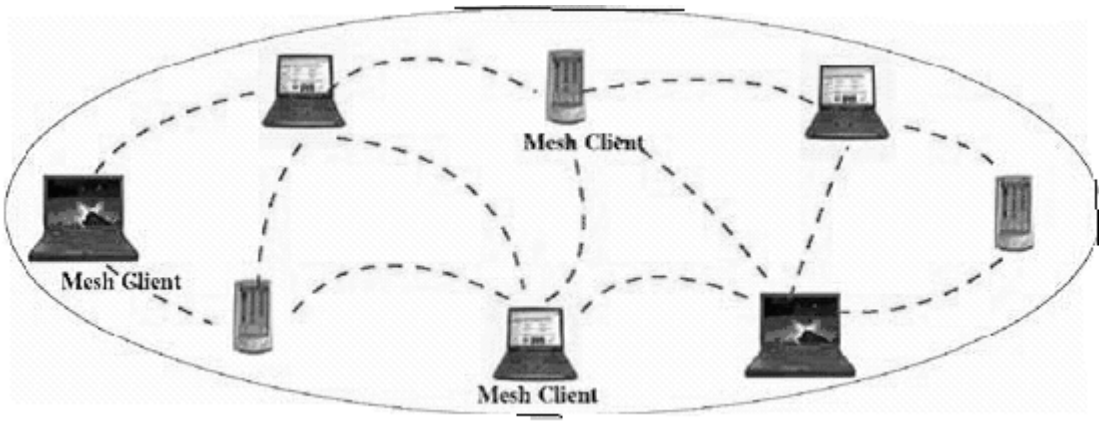


FIG: 2 Client WMN

2.2.3 Hybrid WMN

Hybrid WMN (FIG 3 [1]) is the combination of infrastructure and client meshing. Mesh clients are able to access the network through mesh routers as well as directly meshing with other mesh clients. While the infrastructure provides connectivity to other networks such as the Internet, Wi-Fi, WiMAX, cellular, and sensor networks. The routing capabilities of clients provide improved connectivity and coverage inside the WMN [1]. The hybrid architecture provides full advantage of the WMN. Of all the Architectures it is best and has maximum applicability.

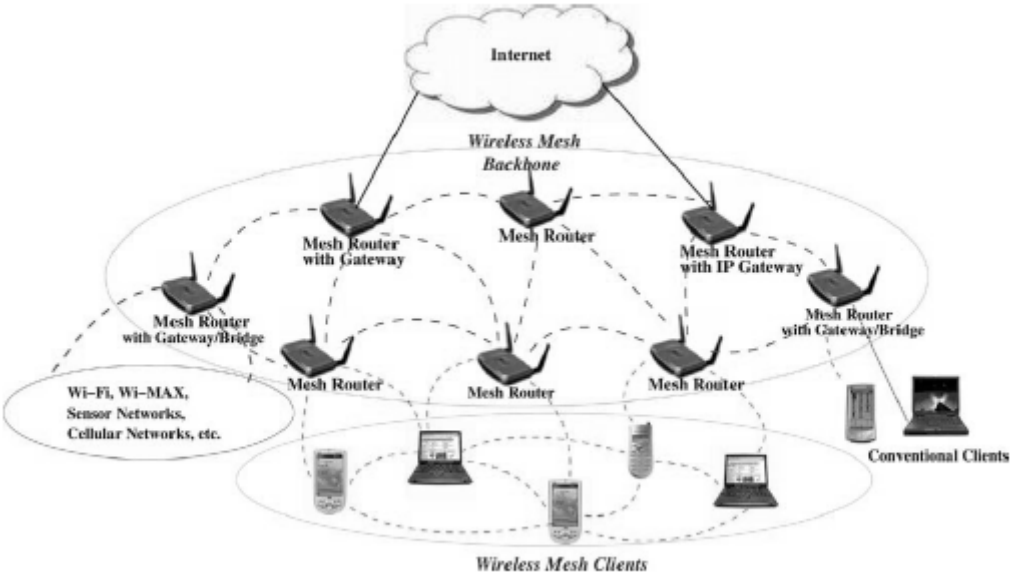


FIG: 3 Hybrid WMN

2.3 WMN Advantages

- WMNs support ad hoc networking, and have the capability of self-forming, self- healing, and self-organization [7].
- WMNs use multi-hop communications; they have a wireless infrastructure/backbone provided by mesh routers [7].
- Mesh routers have minimal mobility and perform dedicated routing and configuration, significantly reducing the load off the mesh clients and other end nodes [1].
- Wireless infrastructure easily supports the mobility of end nodes [3].
- Mesh routers integrate heterogeneous networks, including both wired and wireless. Thus, multiple types of network access exist in WMNs [3].
- Power-consumption constraints are different for mesh routers and mesh clients [1].

2.4 Characteristics of WMNs

• WMNs are Multi-hop in nature.

One purpose that WMNs serve is extension of the coverage range of current wireless networks without sacrificing the channel capacity. Another purpose is to provide non-line-of-sight (NLOS) connectivity among the users [1]. To meet these requirements, the mesh-style multi-hopping is indispensable. Multihop networks achieves higher throughput without sacrificing effective radio range via shorter link distances, less interference between the nodes, and more efficient frequency re-use.

•WMNs are self-forming, self-healing, and self-organizing.

The network performance in WMNs enhances, because of flexible network architecture, easy deployment and configuration, fault tolerance, and mesh connectivity, i.e., multipoint-to multipoint communications. It is attributed to these features, that WMNs have low upfront investment requirement, and the network can grow gradually as the situation demands [7].

• Mobility depends on the type of mesh node.

The design of a WMN requires that mesh routers have minimal mobility, while mesh clients can be stationary or mobile nodes. This requirement simplifies the implementation of a WMN [1].

• Power-consumption constraints depend on the type of mesh node.

Power consumption constraints for Mesh routers are usually not strict. However, due to the mobility aspect of mesh clients, they may require power efficient protocols. For example, consider a mesh-capable sensor. It certainly requires its communication protocols to be efficient in terms of power [1]. So we may derive that, the MAC or routing protocols suited for mesh routers may not be the optimal choice for mesh clients, such as sensors, as power efficiency is a significant issue in wireless sensor networks.

• WMNs are Compatible and interoperable with existing wireless networks.

WMNs built on IEEE 802.11 technologies must be compatible with IEEE 802.11 standards in the sense of supporting both mesh capable and conventional Wi-Fi clients. These WMNs should also be inter-operable with other wireless networks and cellular networks [1]. Due to the lack of wired infrastructure that exists in cellular or Wi-Fi networks through deployment of base stations or access points, WMNs are generally considered as a type of ad-hoc networks. Additional capabilities necessitate more sophisticated algorithms and design principles for the realization of WMNs [3] along with the ad hoc networking techniques required by them. In other words we can say that, instead of being a type of ad-hoc networking, WMNs aim to diversify the capabilities of ad hoc networks. Consequently, ad hoc networks can actually be considered as a subset of WMNs [4].

• WMNs support multiple types of network access.

In WMNs, backhaul access to the Internet and peer-to-peer (P2P) communications are supported. In addition, the integration of WMNs with other wireless networks and providing services to end-users of the integrated networks can be proficiently done through WMNs [1].

• WMNs usually have Wireless infrastructure/backbone.

As discussed before, WMNs consist of a wireless backbone with mesh routers. The wireless backbone provides large coverage, connectivity, and robustness in the wireless domain. However, the connectivity in ad hoc networks depends on the individual contributions of end-users, which may not be reliable [3].

• WMNs Integrate Conventional and Mesh networks

WMNs easily integrate with conventional clients that use one of the radio technologies used by a mesh router, which is implemented through a host-routing function in mesh routers [3]. Integration of various existing networks such as Wi-Fi, the Internet, cellular and sensor networks can be done by WMNs as the mesh routers have gateway/ bridge functionalities in them [1]. As a result, users in one network can access services in other networks, by making use of the wireless infrastructure.

•In WMNs mesh routers perform dedicated routing and configuration.
Additional functionalities of routing and configuration of all other nodes are performed by end-user devices in ad hoc networks along with other functions. WMNs consist of mesh routers for these functionalities [1]. This reduces the load on end-user devices significantly, which in turn provides lower power consumption and high-end application capabilities to possibly mobile end-users with energy constrains [1]. This also leads to decrease in the cost of devices that can be used in WMNs due to the limited end-user requirements.

•WMNs can work on multiple radios.
Routing and access functionalities can be assigned to different radios in mesh routers, as they can be equipped with multiple radios. Thus enabling segregation of two main types of traffic in the wireless domain [7]. The access to the network by end users can be separately carried out on a different radio, whereas routing and configuration can be performed between mesh routers on some other radio. This leads to a radical improvement in the capacity of the network [30].

2.5 Critical Design Factors

1. Multiple Radios
Many approaches have been proposed to increase the capacity and flexibility of Wireless systems. These include directional and smart antennas, MIMO systems, and multi-radio/multi-channel systems, few more advanced radio technologies such as reconfigurable radios, frequency agile/cognitive radios, and even software radios [1]. These advanced wireless radio technologies all require an innovative design in higher layer protocols, especially MAC and routing protocols [3].

Mesh routers can be equipped with multiple radios to perform routing and access functionalities. This enables separation of two main types of traffic in the wireless domain. While routing and configuration are performed between mesh routers, the access to the network by end users can be carried out on a different radio [3]. This significantly improves the capacity of the network.

2.Scalability

WMNs support multihop communication, so as it is true for any multihop network that when the size of network increases, the network performance degrades significantly [31]. WMNs also suffer from scalability issues. The impact is that Routing protocols may not be able to find a reliable routing path, transport protocols may loose connections, and MAC protocols may experience significant throughput reduction. Hence, scalability is a critical requirement of WMNs. To ensure the scalability in WMNs, all protocols from the MAC layer to the application layer need to be scalable [3].

3. Mesh Connectivity

Many advantages of WMNs derive from mesh connectivity. To ensure reliable mesh connectivity, network self-organization and topology control algorithms are needed [1]. A significant improvement in the performance of WMNs can be achieved through implementing topology-aware MAC and routing protocols.

4. Broadband and QoS

Different from classical ad hoc networks, most applications of WMNs are broadband services with heterogeneous QoS requirements. Thus, in addition to end-to-end transmission delay and fairness, more performance metrics, such as delay jitter, aggregate and per-node throughput, and packet loss ratios, must be considered by communication protocols [1].

5.Security

Although many security schemes have been proposed for wireless LANs in recent years, they are still not fully applicable for WMNs. For instance, there is no centralized trusted authority to distribute a public key in a WMN due to the distributed system architecture [32]. The existing security schemes proposed for ad hoc networks could be adopted for WMNs. However, most of the security solutions for ad hoc networks are still not mature enough to be implemented practically [1]. Moreover, the different network architectures between WMNs and ad hoc networks usually render a solution for ad hoc networks ineffective in WMNs.

6.Ease of Use

Protocols must be made to allow the network to be as independent as possible. In addition, network management tools need to be designed to proficiently maintain the function, monitor the performance, and configure the parameters of WMNs [3]. Rapid deployment of WMNs can be achieved through the autonomous mechanisms in networking protocol together with the network management tools.

7.Compatibility and Inter-operability

In WMNs it is a default requirement to support network access for both conventional and mesh clients. Therefore, WMNs need to be backward compatible with conventional client nodes. This demands that mesh routers be capable of integrating heterogeneous wireless networks [7].

2.6 Comparison between MANETs and WMNs

It is important to know the differences between MANETs and WMNs, as these two are closely related and many are not able to distinguish a WMN from MANET.

1. WMNs are a way to diversify the capabilities of ad-hoc networks. Implementation of dedicated mesh routers along with mesh clients in WMNs introduces a hierarchy in the network architecture whereas ad hoc networks have flat architecture [4].

2. In WMNs along with mesh clients we also have mesh routers, which provide connectivity to several other networks and among the mesh clients [1]. Clients in ad hoc network operating with a particular radio technology cannot access a client in a different radio technology network (i.e. they have compatibility problems). While in case of mesh networks different networks can be integrated with the help of mesh routers employing multiple standard radios [31].

.

3. Devices forming ad hoc networks are either mobile or portable or both, so they are power constrained [4]. In such a scenario the load of routing decisions and other network configuration functions needs to be offloaded from such devices. In WMNs

this problem is solved through mesh routers, which are mostly fixed with no power constraints, suitable to perform tasks on behalf of all mesh clients relieving clients from such processing and hence enhancing overall performance [1].

4. The effect of mobility is different in WMNs and in MANET due to different network architecture. In ad hoc networks clients' mobility completely change the network shape, which affects the overall routing decisions, and network performance [4]. In case of WMNs the clients mobility has limited effect on the overall routing decisions, as the mesh routers are fixed and responsible for routing and network configuration [2].

5. Use of multiple radios, and multiple channels per radio, increases the effective network capacity and throughput. Diverse network paths may be followed by implementing different routing metrics [31].

2.7 Application Areas of WMNs

The characteristics of WMNs make them an efficient solution in many applications especially in the area of Community and Municipal networks. Some of the applications, which motivate the research, and development of WMNs are [4]:

- Cellular Backhaul
- Campuses
- Municipalities, including downtown cores and parks
- VoIP Interoperability
- Broadband Residential Networking
- Real-Time Multicast Video
- GPS Resource Tracking
- Multimedia Instant Messaging
- Military operations, disaster recovery and temporary installations
- Intelligent Transportation Systems and Logistics
    - Large warehouses, shipping yards, construction sites and

  - Commuter rail lines.
  - Airport terminals and hangers.
  - Video Surveillance and traffic monitoring cameras.
  - Traffic and environmental sensor monitoring.
  - Fixed and portable variable message signs.
  - Adaptive traffic signals.
  - Automatic Vehicle Location (AVL).
  - Remote reporting and database access.
  - Fleet management and communications.

- Community and Neighborhood networking
- Law Enforcement Agencies
  - Instant incident communications
  - Mobile data, video and voice services
  - Real time video surveillance feeds
  - Interdepartmental communications
  - High bandwidth data connectivity for officers on the street

- Enterprise networking
- Metropolitan area networks
- Sensor Networks
  - Chemical, bio, nuclear detection and monitoring
  - Traffic observation, detection and management
  - Industrial Control Systems
  - Portable and mobile site monitors
  - Security and Video monitoring systems

- Health and medical systems

3

Service Discovery

3.1 What is Service Discovery?

In the last 10 years, the trend in technology has been towards creating a ubiquitous computing environment. Essentially, this is where a myriad of devices interconnect in an ad-hoc fashion, across heterogeneous domains [11]. These pervasive computing systems are composed of small-embedded devices, communicating in a wireless network and essentially independent of any global management. This is where the field of service discovery fits in. For a device to be truly mobile, it must be able to interface and co-ordinate with its surroundings without the user's intervention. For this to happen, the service discovery protocol must be able to discover local resources and form an ad-hoc network [5].

Thus, service discovery is the ability to discover and form an ad-hoc network without explicit user direction. It facilitates devices and services to properly discover, configure, and communicate with each other. Service discovery minimizes administrative overhead and increases usability [11]. So the main purpose of service discovery is to detect services and devices offered by devices and computers in a network and to announce offered services to devices and computers

3.2 Service Discovery system components:

- **Service**: It abstracts a set of functionalities offered by a networked entity [34].
- **Client**: These are entities that expect to discover available services in an unknown network with no or little configuration [34].

- **Directory**: Directories are responsible for caching advertisements from available services and carry out lookup to cater to discovery requests from clients. Explicit directory agents are used by some approaches. Few approaches implement the directory as a part of the service/client implementation. [9]

3.3 Service Discovery Steps

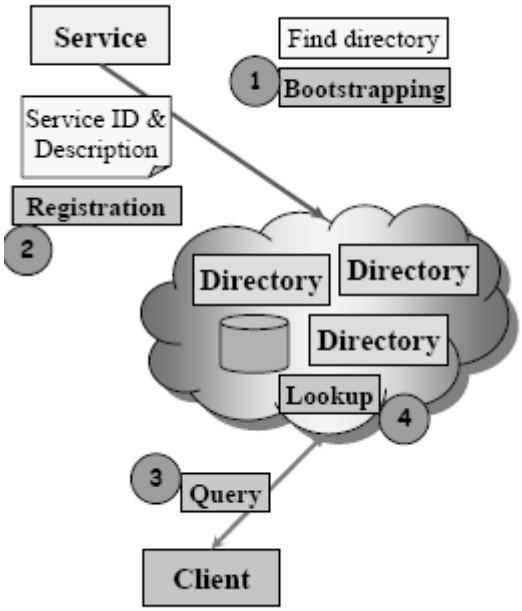


FIG: 4 Service Discovery Steps

Service discovery steps (FIG 4 [9]) involves the following steps:

- Bootstrapping,
- Querying, and
- Obtaining service handle(s) as a result of the query (done by Lookup)

3.3.1 Bootstrapping

Bootstrapping (FIG 5 [9]) is the first step in the discovery process and some a priori knowledge or preconfiguration is required by most discovery approaches [9].
Three basic techniques used for the bootstrapping process are:

• Unicast communication [25], in this method the client/service simply communicates on a particular network address, on which a directory agent or other bootstrap information provider is listening. In few methods DHCP is used to broadcast the address of the directory agents.

• Multicasting [25] from the client/service to a directory agent is also used in few of the approaches. With this method, the bootstrapping unit declares its existence using a distinguished multicast address on which directory agents are listening. After receiving the message, the directory agent usually react to the client/service with essential information or authentication request on a different unicast channel opened by the client/service [9].

•Yet another method is to listen for the periodic multicast of directory advertisements on a distinguished well known multicast address. This is the reverse process of the previous technique and is implemented by a few approaches [9].

The disadvantage associated with this method is that multicasting from client/service to directory agents consumes more bandwidth. The advantage is its flexibility.

From the clients' perspective, service discovery involves bootstrapping, querying, and obtaining service handle as a result of the query. A service first bootstraps and then registers itself with a directory agent if present [35].

Bootstrapping

- Requires some pre-configured knowledge
- Three major approaches:

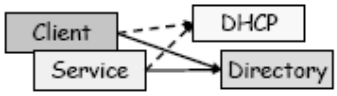
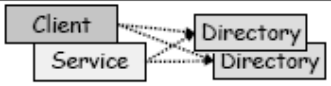
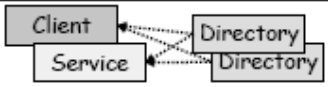
Approach	Figure	Bandwidth requirement	Fault-tolerance	Network Dependent
Unicast to directory. (IP obtained from DHCP server/ out of band)	 INS,Twine,SLP,P2P,Grid	Low	Low	No
Multicast by client/service	 SLP, Jini,Splendor	High	High	Yes
Periodic **multicast advertisement** by directory	 SLP, Jini, Splendor	Moderate	High	Yes

FIG 5: Bootstrapping Approaches

3.3.2 Registration

Service providers register their services at directory/directories, which can be searched and found by Service Requestors. Service Registration refers to the representation of the directory information [9].

A service request from the client is transformed into a query. Likewise, a service from the service provider has to be abstracted and formally represented, this is achieved through directory information; directory information is stored in a directory, which may reside at one or many places in a network [35]. Two important aspects related to directory information are: expressiveness and formalism.

Expressiveness defines the versatility of directory information and what content it can contain, and Formalism of the directory information look at how it is structured. In the process of query matching the query is matched against the directory information,

these aspects of directory information have a direct impact on the effectiveness, correctness, and scalability of the discovery service [9].

Most common Directory Information used for service discovery is as follows:

• **Service Identity**:

For distinguishing a service from all other advertised services, all the service discovery methods use a unique identifier associated with each service. A globally unique flat identifier generated locally by the service may be used in some cases. In other cases the identity is composed of different pieces of information, for instance locally unique identifiers combined with the network address (IP:port pair) of the service [9].

• **Service Description**:

Service lookup based on a given description is offered by most of the discovery methods. Service description is usually stored as a set of attribute-value pairs. Some discovery approaches store attribute-value pairs as lists and do not relate attributes with each other [34]. In some approaches attributes are related based on dependency relationships and stored in a tree-like hierarchy. A predefined set of attributes for each service type may be used. Most widespread used way of service description is XML-based representation, implemented by many discovery approaches for obvious advantages of extensibility and openness [36].

• **Service Handle**:

Service handle provides access to a service. The outcome of service discovery is a service handle. The service handle should enable the client to access a service by providing any form of reference to it [9].

Some examples of service handle obtained after a service lookup are:

- Service identifier,
- Network address of the service,
- URL for the service,
- Service description as a set of attribute-value pair
- Service description in some standard language like XML, and

- Proxy stub

Network address of a service is the minimum information provided by a discovery approach.

• **Expiry Time**:

Directory information update and consistency maintenance can be done following two approaches: hard-state and soft-state registration. Both methods have their pros and cons, hard-state registration has a lower bandwidth needs but provides lower fault-tolerance. Soft-state registration offers higher tolerance to service failures, but at the cost of higher bandwidth consumption [11]. Consequently, there is a tradeoff between fault tolerance and bandwidth consumption. A mix of these two approaches can also be used.

3.3.3 Query Matching

The process of querying involves matching the client's request (expressed as a query) against all services (directory information) to obtain a satisfying subset. The querying language implemented by a discovery system, enables the client to impose different types of constraints on this satisfying subset and eases discovery of more relevant services [35]. Diverse levels of expressiveness are followed by various service discovery approaches:

• **Search by category**:

Many service discovery approaches, classify services into categories and allow clients to discover services by category [9].

• **Attribute-value matching**:

Many approaches structure queries as union of a set of attribute value pairs that should be matched against all the advertised service descriptions. For more flexibility Wildcard matching in the value side of an attribute-value pair can also be supported [9].

• **Comparison function**:

Few approaches provide for more than string matching. In these systems, data-type for each attribute can be specified implicitly or explicitly. The precise data-type is used in choosing suitable comparison function during query processing [9].

• **Compound queries**:

Allowing logical, relational and set membership operators in queries forms a Compound query [9].

• **Attribute relationship**:

Allowing dependency relationship between attributes, resulting into tree-like service descriptions and queries [35].

• **SQL like Query**:

A Query can be most naturally related to SQL Query [9].

Among these different levels of query semantics, SQL like Query and compound queries are the most powerful and flexible. Tree-like query and description as adopted are useful in narrowing down search scope for large systems [9].

3.3.4 Service Lookup

A Service Lookup depends on the directory architecture (FIG 6 [9]).
The directory architecture embraced by different service discovery approaches can be broadly grouped as centralized or decentralized [34]. In centralized architectures, the directory information is stored in a central location in the network.

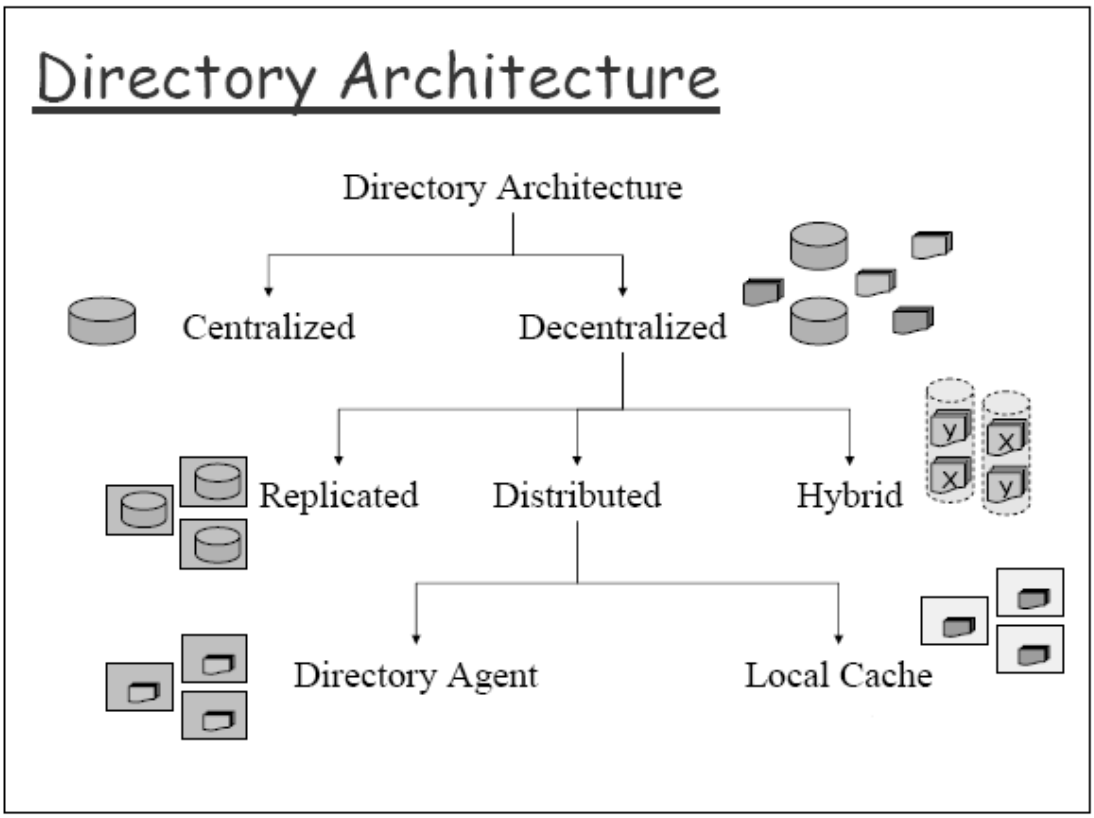


FIG 6: Directory Architecture Types

Directory information is stored at multiple network locations when decentralized architectures are implemented. Decentralized directory systems can further be grouped as replicated, distributed and hybrid [9].

The entire directory information is stored at different network locations in case of replicated architecture. The directory information is partitioned and the parts are stored at different network locations in distributed case. In the hybrid architecture, both replication and distribution are implemented. For distributed directory architectures dedicated servers can store the directory information [35]. Centralized directory architecture is not suitable for large systems, due to scalability issues, like the directory becomes a performance bottleneck and a single point of failure. On the other hand, consistency of the various replicas is a major concern in the replicated architecture. Distributed and hybrid directory architectures have a good scalability with respect to the growth of network and they also provide better level of fault-tolerance [34].

4

Evaluation and Exploration Environment

4.1 Network Simulator version 2

The Network Simulator version 2 (ns-2) is a deterministic discrete event network simulator, initiated at the Lawrence Berkeley National Laboratory (LBNL) through the DARPA funded Virtual InterNetwork Testbed (VINT) project. The VINT project is collaboration between the Information Sciences Institute (ISI) at the University of Southern California (USC), Xerox's Palo Alto Research Center (Xerox PARC), University of California at Berkeley (UCB) and LBNL [27].

Ns-2 was initially created in 1989 to be an alternative to the REAL Network Simulator. Since then the uses and width of the ns project has grown significantly. Although there are several different network simulators available today, ns-2 is one of the most common. Ns-2 differs from most of the others by being open source software, supplying the source code for free to anyone that wants it. Whereas most commercial network simulators will offer support and a guarantee but keeping the moneymaking source code for themselves. The release of the source code helps users to create their own functions and subprograms, but also makes it easier to implement them into the ns-2 environment. One of the main benefits for the ns project group releasing the source code is that independent researchers can help in the development of ns-2. It is fairly common that a researcher contributes with the code of a non-implemented protocol or algorithm, after constructing it for his studies.

It is worth noting that ns-2 is a research effort and not a commercial software release. The difference is that there are very few people in the ns project group compared to ordinary software, leading to difficulties in supporting all the users. That problem has lead to the solution of having a huge mailing list (http://mailman.isi.edu/mailman/listinfo/ns-users) for anyone interested, as well as a complete archive of all the mails ever been sent to this mailing list. The mailing list is based on the idea of user helping user, taking the load of the ns project group. The mailing list and the archives are a huge help for all users of ns-2, no matter old or new, since usually someone else has had the same problem before.

Another important thing to remember is that ns-2 is an ongoing project and hence not completed product. This being the reason why it is free and offers no support except the mailing lists. The people that are in charge of the project heavily rely on the users to find bugs and faults and reporting these when discovered. This also leaves the validating of results to the user, but the user is not alone so help is just an email away. The most common protocols are so well used and checked so the main worries are the new implementations. New implementations usually start out as a research assignment not linked to the ns project group. Since the project group does not have a full company helping them in verification and implementation they have no possibility to do everything themselves thus encouraging any help they can get.

4.2 The ns-2 structure

Ns-2 is made up of hundreds of smaller programs, separated to help the user sort through and find what he or she is looking for. Every separate protocol, as well as variations of the same, sometimes has separate files. Though some are simple, still dependent on the parental class [27].

4.2.1 C++

C++ is the predominant programming language in ns-2. It is the language used for all the small programs that make up the ns-2 hierarchy. C++, being one of the most common programming languages and specially designed for object- oriented coding, was therefore a logical choice what language to be used. This helps when the user wants to either understand the code or do some alterations to the code. There are

several books about C++ and hundreds, if not thousands, of pages on the Internet about C++ simplifying the search for help or answers concerning the ns-2 code.

4.2.2 OTcl

Object Tcl (OTcl) is object-oriented version of the command and syntax driven programming language Tool Command Language (Tcl). This is the second of the two programming languages that ns-2 uses. OTcl is used as a front-end interpreter in ns-2. Linking the script type language of Tcl to the C++ backbone of ns-2. Together these two different languages create a script controlled C++ environment. This helps when creating a simulation, simply writing a script that will be carried out when running the simulation. These scripts will be the recipe for a simulation and is needed to set the specifications of the simulation itself. Without a script properly defining a network topology as well as the data-rows, both type and location, nothing will happen. For a more in depth presentation of these scripts have a closer look at the introduction and related chapters in the ns-2 manual.

4.2.3 Nodes

A node is exactly what it sounds like, a node in the network. A node can be either an end connection or an intermediate point in the network. All agents and links must be connected to a node to work. There are also different kinds of nodes based on the kind of network that is to be simulated. The main types are node and mobile node, where node is used in most wired networks and the mobile node for wireless networks. There are several different commands for setting the node protocols to be used, for instance what kind of routing is to be used or if there is a desire to specify a route that differs from the shortest one. Most commands for node and mobile node can be found in the ns manual. Nodes and the closely connected link creating commands, like simplex link and duplex link, could be considered to simulate the behavior of both the Link Layer.

4.2.4 Agents

An agent is the collective name for most of the protocols you can find in the transport layer. In the ns-2 manual they are defined as the endpoints where packets are created and consumed. The agents in ns-2 are all connected to their parent class, simply

named Agent. This is where their general behavior is set and the offspring classes are mostly based on some alterations to the inherent functions in the parent class. The modified functions will overwrite the old and thereby change the performance in order to simulate the wanted protocol.

4.2.5 Applications

The applications in ns-2 are related to the Application Layer in the TCP/IP suite. The hierarchy here works in the same fashion as the in the agents case. The ns-2 applications are used to simulate some of the most important higher functions in network communication. Since the purpose of ns-2 is not to simulate software, the applications only represent some different aspects of the higher functions. Only a few of the higher layer protocols has been implemented, since some are quite similar when it comes to using the lower functions of the TCP/IP stack. For instance there is no use adding both a SMTP and a HTTP application since they both use TCP to transfer small amounts of data in a similar way. The only applications incorporated in the release version of ns-2 are a number of different traffic generators for use with UDP and telnet and FTP for using TCP. All the applications are script controlled and when concerning the traffic generators, you set the interval and packet-size of the traffic. FTP can be requested to send a data packet whenever the user wants to, or to start a transfer of a file of an arbitrary size. If starting an FTP transmission and not setting a file-size the transmission will go on until someone calls a stop.

4.2.6 NAM

The Network Animator NAM is a graphic tool to use with ns-2. It requires a nam-tracefile recorded during the simulation and will then show a visual representation of the simulation. This will give the user the possibility to view the traffic packet by packet as they move along the different links in the network. NAM offers the possibility of tracing a single packet during its travel and the possibility to move the nodes around for a user to draw up his network topology according to his own wishes. Since the simulation has already been performed there is no possibility for the user to change the links or any other aspect of the simulation except the representation. NAM is dependant on the existence of an X-server in order to be able to open a graphical window. Therefore there has to be a version of X running if NAM is to work.

4.3 MW-NODE for NS-2

The wireless and mobile networking support in ns-2 is monolithic and not entirely coherent with basic design principles so with the development of these technologies simulator evolution to simulate the new technologies has become difficult [27]. One of the new features i.e. multiple wireless interfaces cannot be simulated without additional modifications. The MobileNode object have several limitations, it is justifiable to replace the MobileNode object with a new, more flexible and better-integrated layout of existing components. This step will provide better support for wireless and mobile networking. This approach as presented in [24] uses modules (MW-Node i.e. Module-based Wireless Node), which is defined as:

A Module-based Wireless Node (MW-Node) is a Node with wireless and possibly mobile and energy-support capabilities added by means of modules. It is not a new node object derived from Node. Rather it is a new layout of mostly existing components. This new design enables to support new features and in particular it aims to provide:

• Support for multiple interfaces/multiple channels, and

• A common basis for the implementation of wireless routing protocols [22].

The current technologies have grown in terms of hardware to support multiple interface and multiple channels. Multiple interface support means support for several interfaces (possibly using different technologies) on any node, where each interface is connected to a channel/link and each maybe running its own instance of a routing protocol, the protocols may be same of different for every interface [22]. It also includes the possibility to manage the operation of each interface independently. Conversely, multiple channels refer to a single routing/forwarding or MAC entity, which decides channel to use.

4.3.1 Justification for MW-NODE design

Wireless networking covers a large variety of technologies using different protocols and having roles. It ranges from power-constrained mobile wireless sensors using short-range radio access to WLAN home access and wireless broadband long-range directional communication between fixed WiMAX base-stations. In addition, wireless nodes are becoming multimodal [24]. They are equipped with multiple interfaces possibly using different technologies, including wired access, and may use them concurrently or not. Furthermore, nodes capabilities may vary over time. A node may be fixed and grid-powered for some time and later becomes mobile and battery-powered.

To be able to simulate efficiently this wide variety of scenarios, a flexible, modular, well-integrated and easily configurable design is needed [24]. The current design is neither flexible nor really integrated and that it makes it tedious, if not impossible, to add new features such as support for multiple interfaces on a single node .The Shortcomings of the MobileNode object are presented as under.

4.3.2 Shortcomings of the MobileNode object

1) The MobileNode is a monolithic and inadaptable object:
All the components needed for wireless and mobile simulation whether they are required for the given scenario are included. So the design of a MobileNode is rather inflexible. MobileNode is power-constrained with a velocity and energy model [27]. For simulating a fixed grid-powered wireless node setting its velocity to null and setting its energy model to none is necessary, i.e. mobility support and support for energy consumption will be included even though they are not enabled. Considering the case of infrastructure mode where packets are sent by a single hop between the stations and an access point. And stations are not in charge neither for routing or relaying packets [24]. Even so they will be simulated for performing such functions.

2) The wireless and mobile networking support has inconsistent design:
MobileNode object is linked with the wireless routing and the mobility support, while the support for energy related components are provided through the base Node object itself [24]. As seen practically only MobileNode needs the energy support.

3) An ad-hoc routing protocol is essential for the configuration of MobileNode API:
The "node-config" command is used for configuration of a MobileNode and what makes a Node a MobileNode is if an ad-hoc routing protocol is provided or not [27]. A wireless node operating in an infrastructure mode does not require any routing functionalities. No routing agent should be defined in such case.

4) The MobileNode configuration API uses a flat approach:
Support for hierarchical addressing, and setting up wireless specific components both use a unique command, namely node-config. The node-config command should be used exclusively to enable/disable modules and each module configured independently using dedicated commands [24].

5) The use of agents for routing and forwarding does not agree with basic design principles: Agents should represent endpoints where packets are either formed or consumed. Using agents for forwarding packets is thus in disagreement with the basic function of an agent. Routing agents should only be used to generate packets related to routing for example route request packets, but should not handle data traffic [27].

6) Currently a common basis for implementing routing protocols for wireless and mobile networks is not supported: It involves defining and implementing a new node type. Besides, when implementing a new protocol, one must set a new case in the wireless node creation procedure (create-wireless -node) and at least write a specific Tcl procedure for setting up the routing support, e.g. create-aodv-agent [24].

4.3.4 MW-NODE Modules

Three main capabilities for the MobileNode translate into three new types of modules, namely wireless, mobility, and energy modules. At most one module of each type may be enabled at a time, and mobility and energy modules require a wireless module [22].

node-config API is used to enable/disable wireless, mobility and energy modules. If a module of same type has already been enabled it is disabled when enabling a new one.

Wireless module

Wireless capability to a Node is provided by wireless module, which includes:

• Wireless communication support between nodes. This is basically related to positioning the Node on Topography, since the communication properties are dependent on the distance between the transmitter and the receiver. For this Tcl commands are provided (e.g. set-position) but to make it more practical and intuitive the same commands are also provided for a Node, which calls in turn the commands on the WirelessModule [27].
• Support for multiple wireless network interfaces on a node. A WirelessModule keeps a list of all the WirelessNetwork Interface attached to a Node and wireless-config provides an API for per interface configuration. Two wireless modules are provided: BaseWirelessModule and PortalWirelessModule [22].

Commands at a glance
$node is-wireless
This command returns 1 if a WirelessModule is registered, 0 otherwise.
$node has-wired-interface
These command returns 1 if the node may have one or more wired interface, 0 otherwise.

Energy module

An energy module provides energy-support to a Node using an EnergyModel. The energy level decreases with time and packet transmissions/receptions [22].

4.4 Cygwin

Cygwin is a Linux-like environment for Windows. It consists of a DLL (cygwin1.dll), which acts as an emulation layer providing substantial POSIX (Portable Operating System Interface) system call functionality, and a collection of tools, which provide a Linux look and feel. The Cygwin DLL works with all x86 versions of Windows since Windows 95. The API follows the Single UNIX Specification as much as possible, and then Linux practice. Two other major differences between Cygwin and Linux are the C library (newlib instead of glibc) and default **/bin/sh**, which is **ash** on Cygwin but **bash** on most Linux distributions [23].

With Cygwin installed, users have access to many standard UNIX utilities. They can be used from one of the provided shells such as **bash** or from the Windows Command Prompt. Additionally, programmers may write Win32 console or GUI applications that make use of the standard Microsoft Win32 API and/or the Cygwin API. As a result, it is possible to easily port many significant UNIX programs without the need for extensive changes to the source code. This includes configuring and building most of the available GNU software (including the development tools included with the Cygwin distribution).

Cygwin began development in 1995 at Cygnus Solutions (now part of Red Hat Software). The first thing done was to enhance the development tools (**gcc**, **gdb**, **gas**, etc.) so that they could generate and interpret Win32 native object files. The next task was to port the tools to Win NT/9x. They could have done this by rewriting large portions of the source to work within the context of the Win32 API. But this would have meant spending a huge amount of time on each and every tool. Instead, they took a substantially different approach by writing a shared library (the Cygwin DLL) that adds the necessary UNIX-like functionality missing from the Win32 API (fork, spawn, signals, select, sockets, etc.). They call this new interface the Cygwin API. Once written, it was possible to build working Win32 tools using UNIX-hosted cross-compilers, linking against this library [23].

From this point, they pursued the goal of producing native tools capable of rebuilding themselves under Windows 9x and NT (this is often called self-hosting). Since neither OS ships with standard UNIX user tools (fileutils, textutils, bash, etc...), they had to

get the GNU equivalents working with the Cygwin API. Most of these tools were previously only built natively so they had to modify their configure scripts to be compatible with cross-compilation. Other than the configuration changes, very few source-level changes had to be made. Running bash with the development tools and user tools in place, Windows 9x and NT look like a flavor of UNIX from the perspective of the GNU configure mechanism [23]. Self-hosting was achieved as of the beta 17.1 release in October 1996.

The entire Cygwin toolset was available as a monolithic install. In April 2000, the project announced a New Cygwin Net Release, which provided the native Win32 program **setup.exe** to install and upgrade each package separately. Since then, the Cygwin DLL and **setup.exe** have seen continuous development [23].

5

Experimental Study and Implementation

5.1 Simulation of Wireless Mesh Network

Wireless Mesh Networks have been discussed in the Chapter 2 of this thesis. In a WMN a mesh access point normally has two or more interfaces. One interface acts locally as a standard base station, and another to run a mesh protocol, which takes part in the wireless backhaul operation, sending and receiving packets and forwarding packets on behalf of other mesh points. Currently the MobileNode object does not support multiple wireless interfaces on a single node, and even hinders the simulation of such functionality, which is a major limitation of the current design and doesn't meet the requirements of a WMN simulation.

It includes simulating mesh routers and mesh clients other simulations are the same. For implementing different channels the physical and MAC layer parameters should be set. I have used a scenario where a WLAN is to be extended. The WLAN uses 802.11 a and 802.11b networks, which are not compatible together i.e. by default they cannot communicate with each other due to the different modulation schemes they use. By using a Mesh router we can solve this problem and the 802.11a clients and 802.11b clients can communicate with each other.

Version of NS-2 simulator used for testing proposed service discovery mechanism: ns-2.30 with mw-node patch to support multiple interfaces and multiple channels.

5.1.1 Configuration of mesh router.

If we want to have a wired link from the internet gateway to the mesh router where mesh router is a Portal node, it is defined using command :

```
$ns_ node-config -wireless +Portal ;
```

set node [$ns_ node];

So node will at least need 1 wired connection to function properly. The portal nodes cannot be mobile but they have wireless capabilities. They can have multiple interfaces as mentioned earlier.

To add an NetworkInterface2 of a given type to a Node [22]
$node add-interface <networkInterfaceType name>

To add a Wireless interface
Set wif [$node add-interface “wireless”];
Where wif is the name of a wireless interface;

A NetworkInterface2 may be added to a Node at any time of the simulation. Once an interface has been added it cannot be removed. Available types are Link and Wireless. To bring all the interfaces of a node, or only the interfaces of a specified type (optional), up or down, respectively [22].
$node bring-up-interfaces [<networkInterfaceType name>]
$node bring-down-interfaces [<networkInterfaceType name>]

The state of each interface can also be controlled individually
$interface up
$interface down

Similarly to reset interfaces
$node reset-interfaces [<networkInterfaceType name>]
$interface reset

5.1.2 Configuration of mesh client

A mesh client is a mobile node which may have one or more interfaces [22].

```
$ns_ node-config -wireless +Base -mobility +Base
```

set node [$ns_ node];

To set up interface on different channel we configure the wireless channel for the interface using the command

```
$ns_ wireless-config -channel  [new Channel/Wireless] \
                     -mac      $val_b(mac) \
                     -phy      $val_b(phy)
```

For the Mesh components to communicate they must have a common channel between themselves.

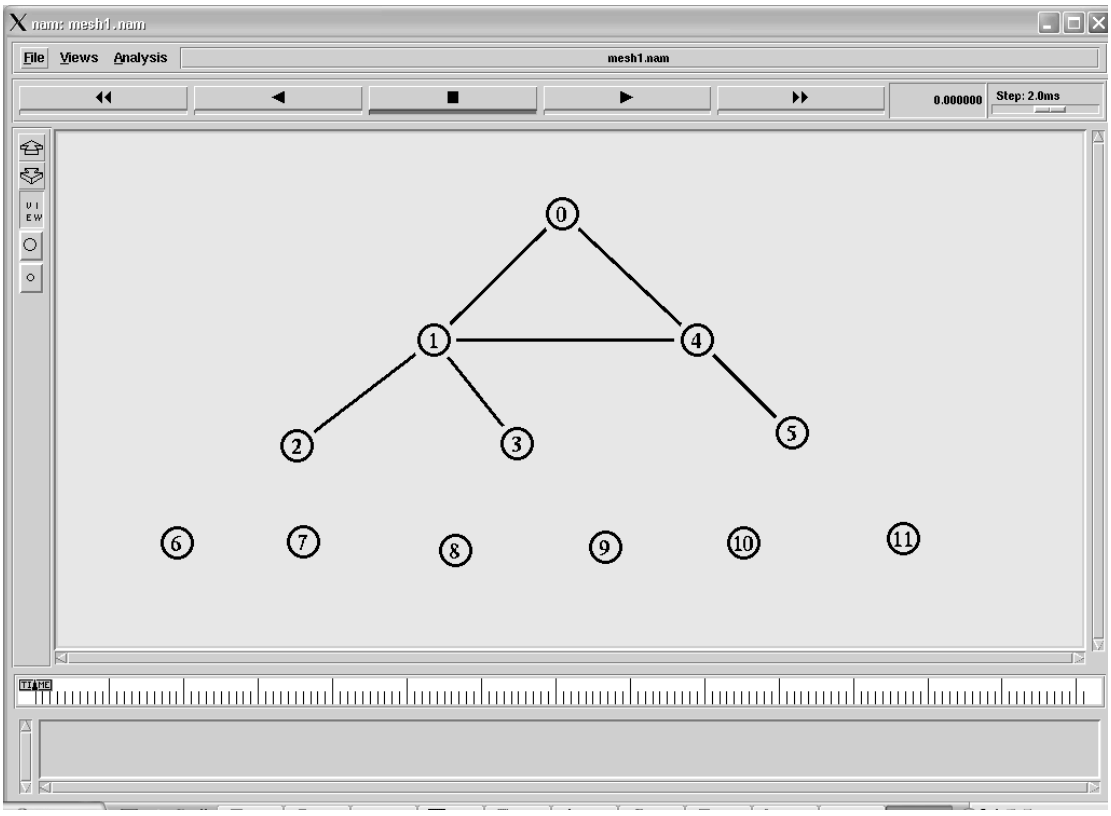


FIG 7: Simulation of Wireless Mesh Network

The above snapshot represents the wireless mesh network simulation.

Node 0 is the Internet gateway.

Nodes 1 and 4 are acting as Mesh Routers.
Nodes 2 and 5 are the 802.11a access point.
Node 3 is the 802.11b access point.
Nodes 6 and 7 are the 802.11a nodes under access point 2.
Nodes 8 and 9 are the 802.11b nodes under access point 3.
Nodes 10 and 11 are the 802.11a nodes under access point 5.

5.2 Service Discovery mechanism proposed

Depending on the service discovery steps as described in chapter 3, a mechanism that can satisfy the requirements for a wireless mesh network is proposed. The mesh network comprises of mesh routers, mesh clients, clients from other wireless networks and conventional clients. It is simulated as a protocol in NS-2 called MDP i.e. Mesh Discovery Protocol.
The proposed service discovery mechanism in context with the service discovery steps:

Bootstrapping is the first step in the discovery process and some a priori knowledge or preconfiguration is required by most discovery approaches.

In my approach I am making use of multicasting, a node joins a group and when it sends a service request, not only does the requesting node get the service reply, but also the nodes which are part of the same multicast group get service reply message enabling all the nodes to cache the availability of the service in their remote caches.

I made the choice of multicast over unicast, as delivering content to many people at the same time is an effective way to utilize network resources. This can be achieved by using multicast technologies. Since Wireless medium is used where the radio spectrum is a rare resource and is typically considered as the bottleneck in communication systems. The benefits of multicast on the air interface is that many users can receive the same data on a common channel, thus not clogging up the air interface with multiple transmissions of the same data [25].

Registration is the process where a service from one node is registered on another node. Two commands are there which perform registration in my approach. One is "register" used explicitly to register a service; another is "search" which performs registration of the service on all other nodes of the multicast group even when a service is requested by one node only.

In the process of registration we consider the information that is to be put in the directory (which is implemented as local and remote cache). I have implemented Directory on every MDP agent, which enables every node to store information about the available services. The Directory information stored is the service name, which is not unique. To identify a service a unique service Id called XID is assigned to every service. From the client's perspective he requires a service handle as a result of the service query, this XID acts as service handle. Another Directory information used is TTL value associated with every service, which specifies Expiry time of a service.

Query matching, used to serve a service request. In my mechanism it is done using a simple way Attribute-value matching. This matched the service attribute with the cache information to find the service; the service name is the value it is matched against.

Service Lookup, used to find a service .As stated earlier a Service Lookup depends on the directory architecture.

Directory Architecture used is distributed, for the purpose of making the mechanism a flexible one, where every node can act as a client or a server. As we have the stability advantage in mesh router, I have kept a considerable directory size in mesh routers as compared to the mesh clients When excessive number of services try to register with a mesh clients a Cache overflow situation will prevent this, hence preventing the overburden on the mesh clients.

In a way Replicated architecture is also used along with the distributed one as a particular service is added on to the remote cache of all the nodes in a multicast group if ALL option is used.

5.3 Implementation Details:

When a mesh router or a mesh client wants to participate in a service discovery mechanism an agent of type MDP is attached with it. The stability of mesh router allows in proposing a scheme in which energy constraints are not of much importance.

The services that this mechanism uses for service discovery are as under:

1) conf
2) enable
3) register
4) add
5) search
6) deregister
7) showcache

To enable a client to participate in the service discovery mechanism it is attached with an agent MDP which enables the node to perform functions for the service discovery.

1) conf

The commands used for attaching a MDP agent are :

```
set node_(1) [$ns_ node];
set p0 [new Agent/MDP];

$ns_ attach-agent $node_(1) $p1;
```

conf usage is :

```
$p0 conf [getRandom] 1000
```

The conf function is

```
void MDPAgent::config ( int random, int ttl )

{
```

```
    srand ( random );

    xid = rand ( ) % 0xFFFFFF;

    cache.config ( here_.addr_, ttl );

}
```

conf attaches a cache with every MDP agent and assigns a unique xid to it.
cache is MDPCacheMemory that is associated with every MDP agent .
Every MDP agent has two caches associated with it

1) Local Cache, where it stores the service information that it hosts locally.
2) Remote Cache, where it stores the service information of service providers that it has cached or the service providers who have explicitly opted to register their service with that particular node.

2) enable
The usage is:
$p0 enable
Every MDP agent is having a status associated with it, by using the above command we enable the agent and the status is set to ENABLED . So the agent is fully functional, and the agent can actively take part in service discovery .

3) register
The usage is :
$node <node no> register <service> <ttl>
This command registers <service> from <node no> to node with the given <ttl>. So this command is analogous to the service registration function. As discussed a MDP agent has two caches, with this function we add the service in the remote cache of the node .For uniquely identifying the service we have service ID which can be used as a Service handle too.
When a MDP agent is attached to a node it can act as a server as well as a client. The fact that mesh router should be more resourceful is also considered.

4) add
The usage is :

$node add <service>

This command adds the service in the local cache of the node signifying that it is a service that the node itself serves. For identifying the service we associate a service ID with it.

5) search

The usage is :

$node search <service> ANY/ALL

This enable the node to initiate a service query process .Here two options ANY or ALL are provided , enabling a node to make choices, if a node wants choices in choosing the service from the same service provided by many nodes is can use ALL option.

When a node uses search command a service request message is sent to ANY or ALL nodes depending on the user choice. After the other nodes receive the service request the nodes having the service will send the reply to the service requester.

6) deregister

the usage is :

$node deregister

This command sets the status of the node to DISABLED ,which in turn removes the service provided by the node from the remote caches of all other nodes. In other words the node does not participate in the service discovery any more.

7) showCache

the usage is :

$node showCache

This command shows the local and remote cache of the node ,it displays the service name and the node, which provides the service.

5.4 Simulation Results

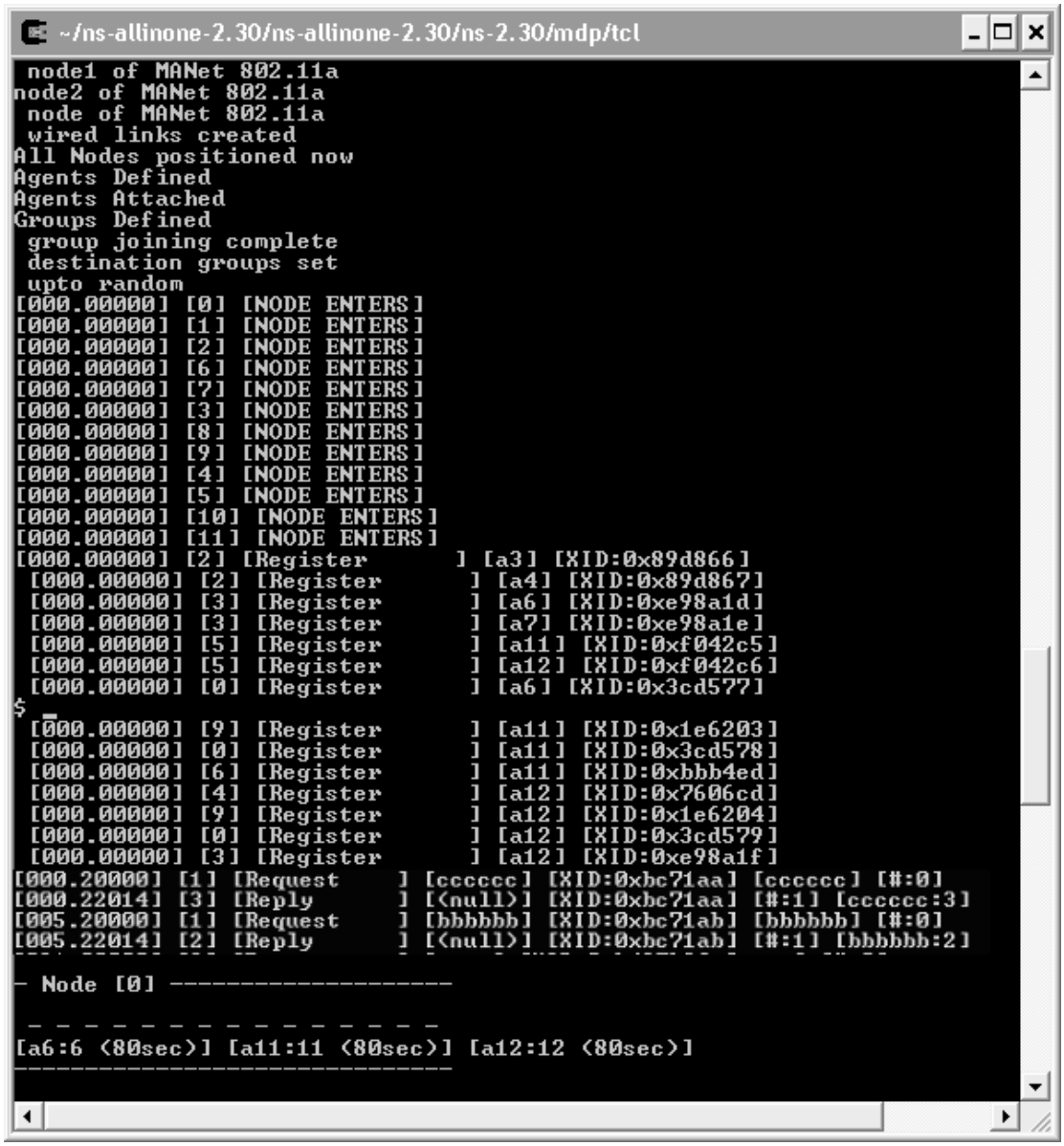


FIG 8: Simulation Result start screen

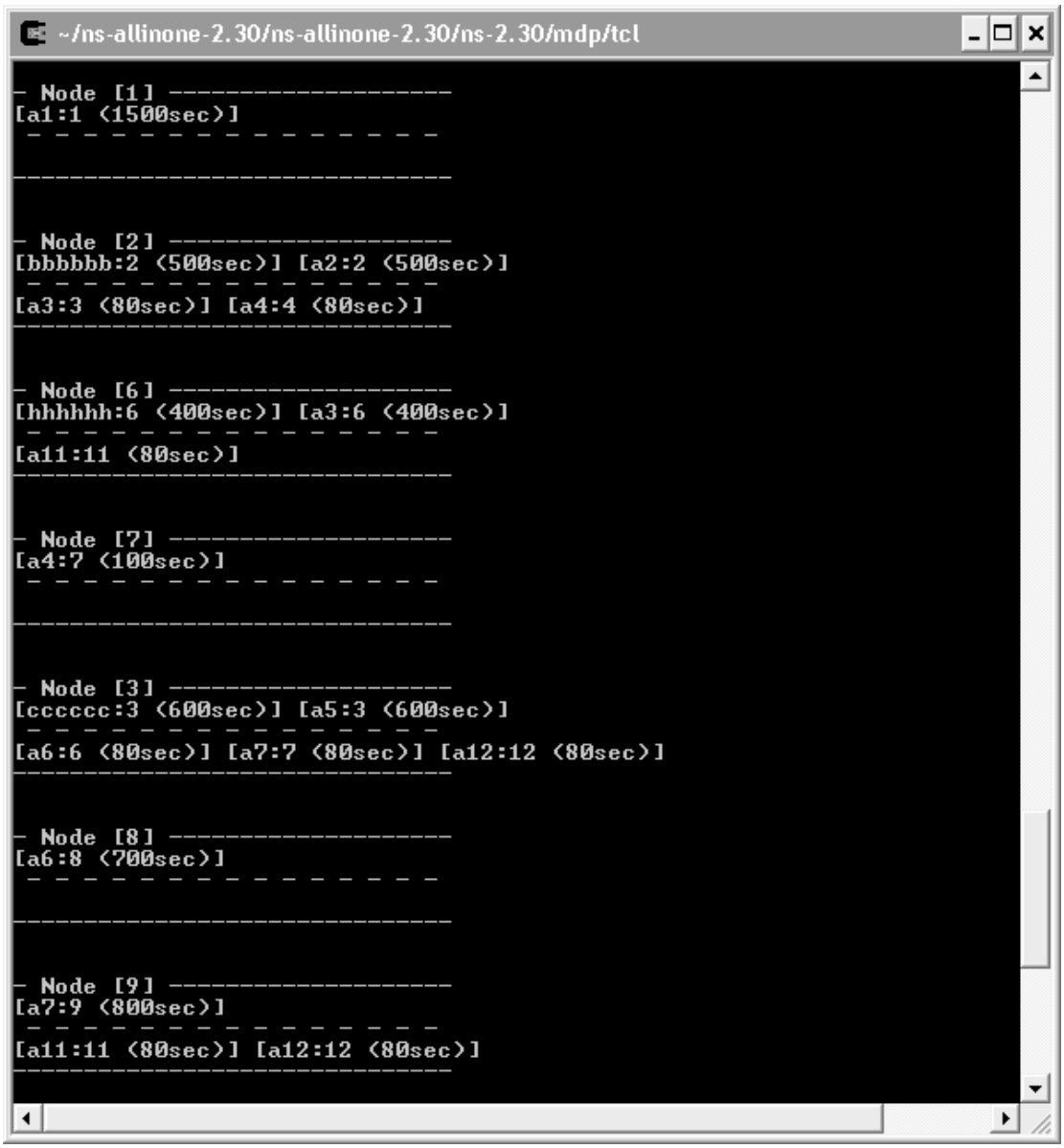


FIG 9 : Simulation Result second screen

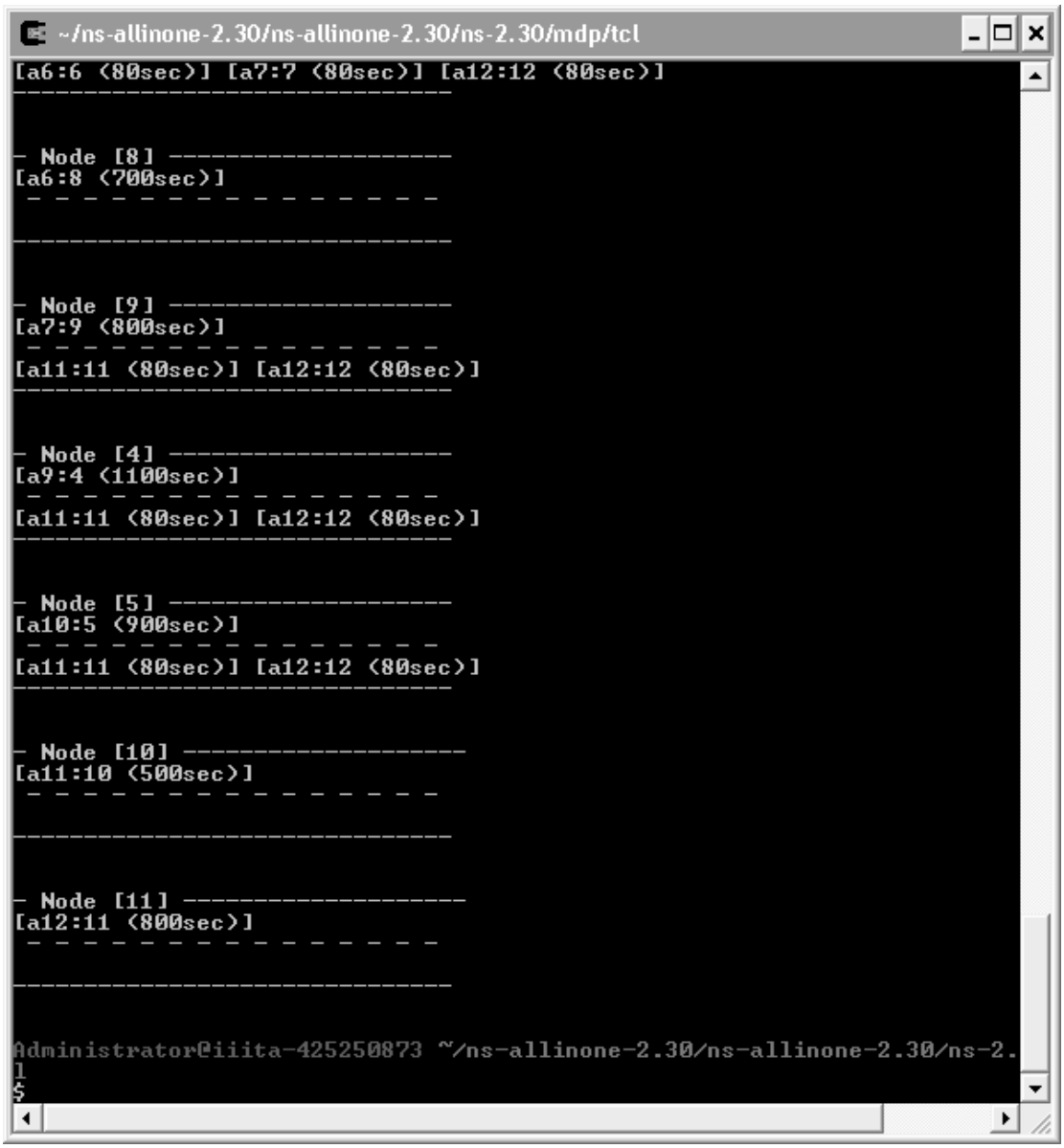


FIG 9 : Simulation Result end screen

6

Conclusion

This chapter concludes the thesis work in relation to the project of Service Discovery in Wireless Mesh Networks. In the conclusion I review the objective of thesis and summarize the main contribution of the thesis. During the thesis I also encountered some interesting problems that I do not have time to investigate. These problems are stated in the future work section.

6.1 Conclusion

The objective of the thesis is "To propose a Service Discovery mechanism that can work with WMNs". My Mechanism mainly applies to scenarios when multiple WLANs are joined together to extend the network. The WLANs may or may not be using the same radio technologies. To achieve this objective we designed several sub-tasks. This thesis has carried out the sub-tasks and completed the objective. The contribution of the thesis is as follows.

The basic knowledge about Wireless mesh network have been gathered, which can help in developing efficient and feasible service discovery mechanism for them. The prerequisites that a Service Discovery mechanism must have are also studied. A detailed analysis of all the Service Discovery Steps has been done. Steps and method to be used for service discovery are discussed , a simple and feasible approach for each step like bootstrapping, registration, query matching and service lookup have been chosen .Due to unavailability of multiple channel and multiple interface support in basic ns-2 versions , modifications have been done to ns2. This modification had a

significant impact on the project and helped greatly in simulating Wireless Mesh Networks.

Simulations have been performed to ensure that the proposed mechanism works and is feasible for wireless mesh networks. The first step was simulation of a wireless mesh network to demonstrate that networks with different radio technologies can communicate with each other. In the next step, developed mechanism named as Mesh Discovery Protocol (MDP) is added in ns-2. Simulations have been performed to test the working of MDP.

Finally it is verified that MDP works well with wireless mesh network. It carries out the main functions that are involved in a service discovery process namely, registration, service query, and service reply and deregistration.

8.2 FUTURE WORK

Many problems were faced while simulation of a mesh network in ns-2 . To evaluate certain performance criteria like , efficiency , service response time, and mainly simulation of Hybrid Mesh network, support for these features is to be added.
Other aspects, which need further investigation are:

- Support for sophisticated Query matching techniques; right now the simplest technique to maintain the simplicity of the mechanism has been used.
- Investigate method to make use of Mesh Routers more active in the service discovery process. Right now I have only been able to provide mesh routers with larger directory size.
- Investigate ns2 simulator regarding wireless links. The reason is that though the basic support for multiple interfaces have been added but it is still having problems when there are more number of interfaces while routing of packets. The problem is evident while using wireless links. The packets don’t use the interface that should be used.
- Investigate and compare a partially Centralized directory architecture with the present Distributed Replicated directory architecture for storing the Directory information of the services.

Bibliography

[1] A Survey on Wireless Mesh Networks, Ian F. Akyildiz, Georgia Institute of Technology, Xudong Wang, Kiyon, Inc, *IEEE Radio Communications* September 2005 page s23-s80

[2] Wireless Mesh Network Solution Brief, http://products.nortel.com/

[3] ABI, State of the Art Analysis of Wireless Mesh Technologies 2006, T. Huovila, P. Lassilay, J. Manner, and A. Penttinen, University of Helsinki, Helsinki University of Technology

[4] A comparison of MANETs and WMNs: commercial feasibility of community wireless networks and MANETs, ***A****CM International Conference Proceeding Series, 2006*, Sahibzada Ali, Sahibzada Ali, Shoiab khan, Hamed Al-Raweshidy, Brunel university, west London.

[5] Service Discovery in Pervasive Systems Steven R. Livingstone, 2003, *Unpublished honours thesis.* The School of Information Technology and Electrical Engineering. The University of Queensland.

[6] Emerging standards for wireless mesh technology, Lee, M.J. Jianliang Zheng Young-Bae Ko Shrestha, D.M.City Univ. of New York, NY, USA; *Wireless Communications, IEEE, April 2006*, On page(s): 56- 63

[7] Wireless Mesh Networks For Residential Broadband, Dave Beyer, Nokia, *National Wireless Engineering Conference, San Diego*, 4 November 2002

[8] http://en.wikipedia.org/wiki/WLAN

[9] Service Discovery Protocols A Comparative Study: Reaz Ahmed, Raouf Boutaba, Fernando Cuervo, Youssef Iraqi, Dennis Tianshu Li, Noura Limam, Jin Xiao & Joanna Ziembicki School of Computer Science, *University of Waterloo, Canada*

[10] Liang Cheng, Service advertisement and discovery in mobile ad hoc networks, Workshop on Ad hoc Communications and Collaboration in Ubiquitous Computing Environments, in conjunction with the ACM 2002 *Conference on Computer Supported Cooperative Work*, New Orleans, November 16-20, 2002

[11] Service Discovery in Pervasive Computing Environments, Feng Zhu, Matt Mutka, and Lionel Ni, *IEEE Pervasive Computing*, Volume 4, Issue 4 (October 2005), Pages: 81 - 90, Year of Publication: 2005

[12] Yu Yang, Hossam Hassanein, and Afzal Mawji. Efficient Service Discovery for wireless Mobile Ad Hoc Networks. *Proceedings of the 4th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA-06), Dubai/Sharjah*, UAE, March 2006, pp. 571-578

[13] http://www.nortel.com/corporate/news/newsreleases

[14] Securing wireless mesh networks, Ben Salem, N.; Hubaux; *Wireless Communications, IEEE* [see also IEEE Personal Communications, Volume 13, Issue 2, April 2006 Page(s): 50 – 55

[15] "GSD: A Novel Group-based Service Discovery Protocol for MANETs", Dipanjan Chakraborty, Anupam Joshi, Tim Finin, and Yelena Yesha. *In Proceedings, 4th IEEE Conference on Mobile and Wireless Communications Networks (MWCN),* September 2002

[16] "Service Discovery in Mobile Ad Hoc Networks", Zhong Fan and Eduardo Guerreiro Ho .*Proceedings of the Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM'05)*

[17] "Service Rings – A Semantic Overlay for Service Discovery in Ad hoc Networks" Michael Klein Birgitta K¨onig-Ries Philipp Obreiter, *Proceedings of the 14th International Workshop on Database and Expert Systems Applications (DEXA'03)*

[18] Service Discovery Mechanism Over OLSR for Mobile Ad-hoc Networks Jose Luis Jodra, Maribel Vara, Jose Mª Cabero, Josu Bagazgoitia, *Proceedings of the 20th International Conference on Advanced Information Networking and Applications (AINA'06)*

[19] Architecture and Algorithms for an IEEE 802.11-based Multi-channel Wireless Mesh Network (A Raniwala, Prof Chiueh) - *Appears in proceedings of IEEE Infocom '05*

[20] http://www2.informatik.hu-berlin.de/~neukirch/hyacinth-dirk_neukirchen.pdf

[21] http://www.q2s.ntnu.no/~paquerea/ns.html

[22] A Module-based Wireless Node for *ns-2* Notes and Documentation, Laurent Paquereau

[23] www.cygwin.com

[24] A Module-Based Wireless Node for NS-2, Laurent Paquereau Bjarne E. Helvik Centre for Quantifiable Quality of Service in Communication Systems Norwegian University of Science and Technology, Trondheim, Norway *WNS2'06,* October 10, 2006, Pisa, Italy

[25] Multicasting vs. Unicasting in Mobile Communication Systems, Janne Aaltonen, Jouni Karvo Samuli Aalto, *WoWMoM'02*, September 28, 2002, Atlanta, Georgia, USA.

[26] Implementing a New Manet Unicast Routing Protocol in NS2, Francisco J. R, Pedro M. Ruiz Dept. of Information and Communications Engineering University of Murcia

[27] http://www.isi.edu/nsnam/ns/index.html

[28] Groups Hope to Avoid Mesh Standard Mess, Greg Goth, IEEE DISTRIBUTED SYSTEMS ONLINE 1541-4922 © 2005 Published by the *IEEE Computer Society Vol. 6*, No. 9; September 2005

[29] "MeshNetworks website." http://www.meshnetworks.com"

[30] Coverage and Capacity of A Wireless Mesh Network, Jane-Hwa Huang, Li-Chun Wang and Chung-Ju Chang, *2005 International Conference on Wireless Networks*, Communications and Mobile Computing Pages 458-463

[31] A novel channel assignment algorithm based on topology simplification in multi-radio wireless mesh networks, Leiming Xu; Yong Xiang; Meilin Shi *Performance, Computing, and Communications Conference*, 2006 IPCCC 2006. 25th IEEE International Volume, Issue, 10-12 April 2006 Page(s): 8 pp

[32] Wireless Mesh Networks with Seamless Mobility Application Note, http://www.belairnetworks.com/resources

[33] An Architecture for a Secure Service Discovery Service. Steven E. Czerwinski, Ben Y. Zhao, Todd Hodes, Anthony D. Joseph, Randy Katz, *Fifth Annual International Conference on Mobile Computing and Networks (MobiCOM '99)* , Seattle, WA, August 1999.

[34] R.Marin-Perianu, P.H.Harteland, J.Scholten, "A Classification of Service Discovery Protocols", Centre for Telematics and Information Technology, Univ. of Twente, The Netherlands, Technical report nr. TR-CTIT-05-25, Jun. 2005

[35] Feng Zhu, Matt Mutka, and Lionel Ni, "Classification of Service Discovery in Pervasive Computing Environments," *MSU-CSE-02-24,* Michigan State University, East Lansing, 2002.

[36] R. Chinnici et al., *Web Services DescriptionLanguage (WSDL) Version 2.0,*" W3C working draft, Aug. 2004; www.w3.org/TR/2004/WD-wsdl20-20040803.