RESOURCE DISCOVERY IN AD HOC NETWORKS

Diane Tang
Chih-Yuan Chang
Kei Tanaka
Mary Baker

Technical Report No.: CSL-TR-98-769

August 1998

This project is in part supported by FX Palo Alto Laboratories and the National Physical Science Consortium

Resource Discovery in Ad hoc Networks

by
Diane Tang, Chih-Yuan Chang, Kei Tanaka, and Mary Baker
{dtang, mgbaker}@cs.stanford.edu
{chang, kei}@pal.xerox.com

Technical Report: CSL-TR-98-769
August 1998

Computer System Laboratory
Departments of Electrical Engineering and Computer Science
William Gates Computer Science Building, 4A
Stanford University
Stanford, California 94305-9040

Abstract

Much of the current research in mobile networking investigates how to support a mobile user within an established infrastructure of routers and servers. Ad hoc networks come into play when no such established infrastructure exists. This paper presents a two-stage protocol to solve the resource discovery problem in ad hoc networks: how hosts discover what resources are available in the network and how they discover how to use the resources. This protocol does not require any established servers or other infrastructure. It only requires routing capabilities in the network.

Key Words and Phrases: Resource Discovery, Ad hoc networks

Resource Discovery in Ad hoc Networks

Diane Tang[*], Chih-Yuan Chang[+], Kei Tanaka+, Mary Baker[*]

[*]Stanford University, [+]FX Palo Alto Laboratories

Abstract **Much of the current research in mobile networking investigates how to support a mobile user within an established infrastructure of routers and servers. Ad hoc networks come into play when no such established infrastructure exists. This paper presents a two-stage protocol to solve the resource discovery problem in ad hoc networks: how hosts discover what resources are available in the network and how they discover how to use the resources. This protocol does not require any established servers or other infrastructure. It only requires routing capabilities in the network.**

1.0 Introduction

The combination of the growth of the Internet and the popularity of portable computing devices such as laptops has led to an increasing interest in mobile computing and networking; laptop users wish to connect to the network as they move around and visit different locales. Most research in mobile networking has focused on the case where the mobile user has a "home" network that he usually uses, in his office perhaps, and tries to solve the problem of how to keep the user connected to the network when he visits some foreign network, perhaps at a different company. Mobile IP [Per], DHCP (Dynamic Host Configuration Protocol) [Dro], and Dynamic DNS (Domain Name Service) [Vix-TRB] all try to deal with the issues that come up in this type of situation. However, all of these protocols assume that they have an established infrastructure within which they may work, namely the Internet, consisting of its routers and previously established servers.

A different situation arises when there is no such established infrastructure on which to depend; such a network is called an *ad hoc network*. An ad hoc network is a temporary network that is built on the fly, and has the following characteristics:

- A complete lack of established infrastructure: there are no pre-existing base stations to depend on, no established or well-known servers, and perhaps no internet connection.
- A network spanning multiple hops: a packet sent from node A may not be able to get directly to node B. Instead, a packet might have to traverse several intermediate nodes to reach B from A.
- The nodes in the network are mobile.
- A dynamic structure: since the network consists of mobile nodes, the topology of the network is not static, but is instead changing, perhaps rapidly.

To make the concept of an ad hoc network more concrete, such a network might be used in the following examples:

- A conference: people go to conferences, and want to network their laptops together to trade data, paper drafts, etc. The size of the network and the type of hosts brought are not known beforehand.
- A military situation: the army goes on a mission and needs computer support.
- Disaster relief: the red cross goes to a disaster site and needs a network to keep track of completed tasks, supplies, etc.
- A construction site: a construction company goes to a site to build a building and needs to keep track of equipment, workers, etc.

The problem of how to construct an ad hoc network has recently become a topic of interest at several research institutions. Some of the research problems include the following:

- Host naming: in some cases, such as the conference scenario, nodes may come in with a pre-defined name, such as an IP address in their home network. Is the node given a temporary name, or can the node use its own IP address? Does the node get a new name every time it moves within the network?
- Routing: traditional IP routing does not mean much in this network, unless the ad hoc network is connected to the Internet itself. Even so, since the network is a multi-hop network, a host will need to discover how to send packets to another host.
- Security: can anyone join an ad hoc network? Also, how much of the communication is visible by every participant?
- Resource discovery: even though there may not be any established servers, some nodes may choose to provide a service (files, web, printing, etc.) to other nodes. How do other nodes discover what services are out there and how to use them?

In this paper, we address the specific problem of resource discovery in an ad hoc network. Existing solutions, such DHCP [Dro] and dynamic DNS [VixTRB], will not work in an ad hoc network since they require that some sort of server be set up, and as men-

tioned above, an ad hoc network cannot depend on such established resources or infrastructure.

2.0 Proposed solution

The solution we propose is fully distributed with each node advertising the services it provides. There are two stages in our proposed solution: resource discovery, in which nodes discover what resources exist in the network, and resource location, during which nodes locate and discover how to communicate with the servers providing the desired particular resource server. Note that a service is a specific instance of a resource. One node may be looking for a printer resource, and another node may provide the printing service that the first node can discover and use.

The main difference between this scheme and the other proposed resource discovery schemes is that there is no dependence on a central server that clients query to discover which services are present in a network. See Section 5.0 for a more detailed description of the related work.

2.1 Naming

Before we discuss the two phases of the solution, we need to resolve the name mapping issue: each type of resource, e.g., printers or file servers, has its own multicast group. We do the multicast address assignment by hashing the name of the resource (“Printers”, “File Servers”) to the last byte of the multicast address, leading to an address of 232.0.0.x. This type of hashing is also used by the Appletalk Name-Binding Protocol [App]. We do assume that common resources have a well-known resource name, so that different nodes providing the same type of service use the same multicast group. The advantage gained by using a hashing scheme is that users have a more intuitive grasp of what “Printers” might be than what 232.0.0.93 might represent.

This hashing mechanism might is result in an address collision problem, which occurs when two different types of resources end up using the same multicast group address. There are two possible problems. The first one is whether only one resource will be advertised or both. For both to be advertised, the servers advertising only need to check that the resource name as well as the multicast address are the same. Only if both match do the servers check to see who continues advertising and who stops.

The second problem occurs when a client sends a request to the multicast group asking for more information. There are two possible solutions in this case. In the first case, the client can send the resource name in the request packet, and the servers can check to see whether that matches, and if so, then send the information. The other solution is that all the servers can reply, and the client can discard any responses in which it is not interested. The former solution, given today’s mobile nodes with more limited bandwidth than CPU power, is a better choice since it uses less bandwidth at the expense of an extra string comparison.

2.2 Phase I: Resource Discovery

This phase addresses how nodes discover what type of resources exist in the network; it is the bootstrapping mechanism. Every node that provides a service sends an advertisement, which includes the multicast address of that service, to a well-known multicast address and port. Any node interested in seeing which resources exist in this network joins this multicast group to receive advertisements. Essentially, we have traded a centralized server for a well-known multicast address.

Currently, the advertisement follows the form described in the Session Description Protocol (SDP) [HanJa], and advertisements are sent out according to the Session Announcement Protocol (SAP) [Han]. As a result, any host trying to discover resources needs only to run `sdr` [SDR], a session directory tool that receives and parses SDP announcements on the SAP multicast address and port.

Note that if a node provides multiple types of services, it needs to send a separate advertisement for each service. Also, the client need not run `sdr` if it knows the resource multicast group, or the resource name to hash to the associated multicast address – sdr is merely a bootstrapping mechanism for resource discovery.

One refinement we make to prevent multiple nodes from advertising the same resource, and thereby wasting bandwidth, is to have each node that advertises a service also listen for other advertisements; if it sees another node advertising the same service and multicast address, it checks to see which node has the lower address. The node with the lower address keeps on advertising; the other node stops sending advertisements. This assumes that both nodes are participating and behaving properly.

Even if the node stops sending advertisements, it still listens to keep track of when advertisements are sent. If an announcement has not been sent within a small multiple of the expected time, then this node starts advertising the resource again.

2.3 Phase II: Resource Location

By this phase, the client node knows which type of service it is looking for and the multicast address for that type of service. What the client wants to do now is to get more information about the specific service provided by servers in this group, so that it can choose which servers it wants to use.

The client does this by sending a request for information to the multicast address it found in the previous phase and a well known port, currently port 5959. Every node providing this service is a member of the multicast group, and therefore receives the request. Upon receiving the request, every server responds with details about the service (a printer might provide details such as paper size, color, and dpi), as well as information on how the client should access the server, such as the unicast address, port, and protocol the client should use to contact the server. The server may also respond with a URL with more information, or even a script the client could execute.

Once the client chooses a particular server, all further communication with that server takes place via whatever mechanism the server specifies, most likely unicast-based, rather than using the resource multicast group.

3.0 Dependencies

Having presented the protocol, we can see this resource discovery protocol depends only on having multicast routing. However, for the network to be viable at all, it must have unicast routing. Multicast routing is built on top of unicast routing, and therefore multicast must also work.

Further note that this resource discovery protocol does not depend on any properties inherent to ad hoc networks, and could, in fact, be used in any network. The only aspect of ad hoc networks manifest in this protocol is the lack of dependence on any established servers to distribute the resource information. However, we do assume that services have well-known names.

4.0 Open Issues

While this protocol does not depend on anything that is not inherent to any network, there are several open issues, both with the underlying function of ad hoc networks and with the resource discovery protocol itself:

- The stability of multicast routing
- Scalability
- Security

First, let us look at the stability of multicast routing. An ad hoc network may have a rapidly changing network topology. Current multicast routing algorithms may not be able to keep up with the rate of change, especially if the changes affect which nodes are parents and children in its routing trees. Simulating how multicast routing algorithms behave in a changing topology is one method of determining their stability; different underlying unicast routing algorithms also need to be simulated to see how quickly they converge in an ad hoc network, since this could affect multicast performance as well.

Next is the issue of how well ad hoc networks will scale, both with respect to an increasing number of nodes and to more rapid changes in the network topology. Time-to-live (TTL) or administrative scoping can be used so that the protocol will scale for use in large networks. In other words, these scoping mechanisms can be used to limit “how far away” a client looks for services, thus reducing protocol overhead to local neighborhoods rather than the entire network. The issue of scaling to more rapid changes in network topology is similar to the issue of multicast stability discussed above.

The final open issue considered here is security, specifically authentication and authorization. The server needs to be able to authenticate itself to the client so that the client

knows that it is not sending potentially sensitive information to a node impersonating a server. The client also needs to be able to authorize itself to the server, so that the server knows the client is allowed access to the service it provides. There is also a privacy issue, so that no one snooping on the network can get the data sent between the client and server. The security issue is complicated by the fact that there is no established key or certificate server.

5.0 Related work

Many proposals for how to do resource discovery are currently being proposed. Some of these, such as DHCP and dynamic DNS [Dro][VixTRB], do not apply to ad hoc networks, since they depend on using an established server to distribute the information.

Perhaps the most similar resource discovery protocol is from Appletalk [App]. In this protocol, a client wishing to find a service multicasts a request to a service-type-specific, ethernet-level multicast address. All servers providing a service of this type respond. In addition, a server broadcasts a gratuitous reply (identical to what it would send in response to a client request) when it first joins the Appletalk network. The main differences between the Appletalk Name Binding Protocol (NBP) and what we propose is the difference in addressing (IP address and port as opposed to the Appletalk named socket), and the different method in advertising resources.

More recently, the resource discovery working group in the Internet Engineering Task Force (IETF) has been developing the Service Location Protocol (SLP) [VeiGPK]. The main thrust of SLP is in using directory agents, which are established servers with which services register; multicast is used only in small networks or as a backup mechanism. Like our proposal, there is a “service-specific” multicast address, however, this address is not advertised in SLP as it is in our proposal. However, both proposals use a hash of the resource name to generate the appropriate multicast address. Our proposal is much more lightweight than the SLP proposal, with its directory agents, service agents, and user agents. SLP also includes more features, such as supporting queries for a service with specific properties. Our proposal leaves it to the client to parse the server replies to discover which properties each server provides and to choose appropriately.

6.0 Current Status

Currently, we have implemented a prototype. On the server side, a daemon reads in a configuration file listing all the services that host provides. For each service, the configuration file lists a unicast address, port number, resource name, resource description, and other things necessary for the SDP description. In addition, either a URL or a pointer to a script can be included. This daemon advertises these services, listens for other servers advertising the same service to determine whether to keep advertising, stop advertising, or restart advertising. This daemon also joins all appropriate multicast groups to listen for and reply to any requests for information from clients.

On the client side, we just use `sdr` [SDR] to discover the multicast addresses of resources available in the network. Once we have the multicast address, the client is started and sends out a query to the server; the query is resent up to three more times if no replies are received. Once the client starts receiving replies, the descriptions in the replies are listed for the user, and the user can just choose a particular server to use.

For source code, please contact Diane Tang at dtang@cs.stanford.edu.

7.0 Conclusion and Future Work

Much of the research that has been done so far in mobile computing and networking has focused on how to maintain connectivity while the user is moving around. This previous research depends on having an established infrastructure of routers and servers.

In this paper, we have described a resource discovery protocol for ad hoc networks that does not depend on having any established infrastructure; the protocol depends only on being able to route packets in the network. There are, however, several underlying open issues to make both ad hoc networks, and thus this resource discovery protocol, viable. These include the stability of multicast routing given frequent network topology changes, network scalability, and security.

Investigation into the first two issues is underway, and involves simulating ad hoc networks and the routing protocols used. The security issues are still being considered.

8.0 Acknowledgments

Much of this work was done at Fuji-Xerox Palo Alto Laboratories (FX-PAL). I'd like to thank all of the researchers at FX-PAL and the MosquitoNet group at Stanford University for listening and making great suggestions.

9.0 References

[App] Appletalk Name-Binding Protocol. http://adrm1.euro.apple.com/dev/tech-support/insidemac/Networking/Networking-61.html

[Dro] Droms, R. "Dynamic Host Configuration Protocol". RFC 1531, October, 1993.

[Han] Handley, M. "SAP: Session Announcement Protocol". Internet Draft, November 19, 1996. Work in Progress.

[HanJa] Handley, M. and Jacobsen, V. "SDP: Session Description Protocol". Internet Draft, March 26, 1997. Work in Progress.

[Per] Perkins, C., ed. "IP Mobility Support". RFC 2002, October 1996.

[SDR] `sdr`. ftp://cs.ucl.ac.uk/mice/sdr/

[VeiGPK] Veizades, J., Guttman, E., Perkins, C., Kaplan, S. “Service Location Protocol”. RFC 2165, June 1997.

[VixTRB] Vixie, P., ed., Thomson, S., Rekhter, Y., Bound, J. “Dynamic Updates in the Domain Name System (DNS UPDATE)”. RFC 2136, April 1997.