

Honours Project Proposal  
**Resource Management and Access Control in  
Suburban Ad Hoc Networks**

**Stanley Gunawan**  
School of Computer Science and Software Engineering,  
Monash University

May 6, 2003

**Contents**

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Research Context</b>	<b>5</b>
<b>3</b>	<b>Research Methods and Plan</b>	<b>6</b>
3.1	Research Methods . . . . .	6
3.2	Timetable . . . . .	7
<b>4</b>	<b>Relevance of the Research</b>	<b>7</b>
	<b>References</b>	<b>8</b>

# 1 Introduction

A Suburban Ad Hoc (Area) Network [1] is a self-organizing, quasi-static ad hoc (typically wireless) network. It provides high speed connectivity in a suburban community of cooperating networked nodes. Besides being a community network, a gateway to the Internet can be attached, providing a home broadband internet service to the users.

What differentiates SAN from existing wireless community networks (like [wireless.org.au](http://www.wireless.org.au)<sup>1</sup>) and home broadband internet (like Bigpond Broadband<sup>2</sup>) is its independence from a third-party installed and maintained infrastructure to provide connectivity. The nodes in SAN combine their resources together to build a functional network. For example, the most typical network activity is the sending of packets from one node to another. Without a direct link existing between those two nodes, the sender relies on intermediate nodes to pass on the packets until they reach the recipient.

Cooperativeness is necessary for SAN to work, but in this type of community, each user has their own selfish needs. So if possible, each user would like to maximise the benefit they can get from the network, and minimise the work they need to do. A simple scheme is to turn off the device when they don't need it. A more difficult one is to use a device with selfish configuration. For example, such device would refuse to forward any packet from its neighbours, so it wouldn't waste resources (bandwidth and power). This can be achieved by reverse engineering the protocol and building a custom made one, or by modifying the behaviour of an existing one. Even though this is not something that most people can do, after one person has done it, it's very likely that they would share this knowledge. Consider the growing business of game console modification chips that can allow such consoles to play pirated games.

Attempting to build a tamper-proof SAN device to protect it from modification is not a realistic solution. A good overview on tamper resistance, and why solely relying on it for protection is problematic is given in [2]. We should assume that once a user has total unsupervised physical access to a device, he can do almost anything to it. It's essential then to have a system in the community that each node has to adhere for it to participate, that discourage selfish behaviours. This project proposes a virtual economy system in which nodes get charged by other nodes for using their resources.

In SAN, the most valuable resource is the network connection. There-

---

<sup>1</sup><http://www.wireless.org.au>

<sup>2</sup><http://www.bigpond.com/broadband/>

fore the system needs to be designed with this in mind. There are many other resources that a node can share. Examples are processing power (e.g. encryption, security level), storage and authentication of nodes.

In this system, each node has their own bank account. Some amount of money is deposited to it when the node provides resources to others, and some is withdrawn when it needs to pay for using resources owned by other nodes. In this system, users who never share their resources would soon run out of money. Without any money they wouldn't be able to do anything in the network. So although users are still selfish, they need to do work for others in order to fulfill its own needs. This system also prevents resource abuse (e.g. denial of service attack), because each user's resource usage is limited by how much money (s)he has.

We can also extend the economy analogy to create a flexible pricing scheme. That is, the price for each packet doesn't have to be fixed. It can follow the law of supply and demand. So for example, the price for bandwidth in peak time should be higher than off-peak. There is also a load balancing application: nodes which are not very popular (e.g. because it has just joined the network), can offer cheaper price. In effect, in this scheme, no nodes should get too worked up, because as it gets higher demand for its resource(s), it would start increasing the price. Customers that have no particular need to go with that node would find a cheaper alternative. Customer nodes can also express the importance of having a particular packet being sent, by offering a larger reward.

To avoid complicated price negotiation every time a node needs to use a resource, it can sign up a contract with another node, ensuring a fixed agreed price and the availability of the resource. This is especially useful for packet forwarding, since a lot of packets need to be passed around all the time. Performing price negotiations with all the intermediate nodes for every packet would incur too much overhead. So for example, a customer node can sign up a contract with all intermediary nodes guaranteeing availability and fixed pricing for a session.

An obvious problem with this scheme is the storage of money. Where should the information of how much money a node has, be placed? The problem is that in a decentralised network, there is no commonly trusted server where critical information can be stored. In this project, we propose the use of a Distributed Hash Table (DHT) (commonly used in Peer to Peer (P2P) applications) to store each node's money information. A DHT distributes a <key, value> pair over some chosen nodes in the network, and provides efficient lookup of *value* based on *key*. Since DHT distributes the data, there is redundancy. If there is a discrepancy in the returned *value*

(either caused by unintentional corruption or an attempt of cheating), a “voting” can be performed to reliably get the correct data, given that there is enough replication. We can then stop replicating the data on the suspicious node.

Having a Distributed Hash Table in SAN means that we can use it to provide storage and lookup service for many types of information, not just money. For example, it can be used to support node authentication and trust metric by providing storage (for public key and trust value respectively) and lookup service.

For sensitive information like the public key of every node and how much money it has to be stored in a publicly accessible Distributed Hash Table, some access control mechanism has to be employed so that they can't be accessed by unauthorised users. Password Capability [3] is a well suited access control system to do this, since the access control needs to be fine grained. Take for example, the right for one node to draw 5 money unit from Node A's bank account.

Briefly, a capability is like an address to an object (analogous to memory references in programming). A capability also gives its possessor a set of access rights to the referenced object. Password capabilities provide protection from tampering by sparsity: a randomly generated password is used instead of a memory reference. Password capabilities can be derived, creating a child capability with a more restricted set of rights; they can also be passed around like normal system data and revoked.

So the proposed mechanism is:

- When a node joins the network, neighbouring nodes create a uniform bank account entry in the DHT for that node.
- For each of them, a capability is derived and returned to the new node.
- These capabilities have withdraw right, and a restricted deposit right. The deposit right is the right to execute a *deposit* code in the nodes, given a capability with a withdraw right as parameter.
- The new nodes then combine the capabilities to make a single capability. Operation on that capability effects the data on all the nodes hosting the replicated data.

Capabilities with very fine grained access control can be created, for example one which can make the capability act as a cheque: a capability which can only be used once, with the right to withdraw a certain amount of money from a certain node.

## 2 Research Context

Previous research in ad hoc network resource management concentrate mostly on Quality of Service (QoS) issues [4, 6]. The issue of selfish behaviours have been overlooked.

Existing research that addresses cooperativeness in decentralised network (ad hoc network or peer-to-peer system) generally use either of these mechanisms: monitoring of misbehaving nodes [12], trust metric [7] (a good survey is in [8]) and economy system [9, 13].

The approach in [12], and monitoring approaches generally rely on monitor nodes that are responsible for analysing the current state of the network, identifying nodes which aren't cooperating, and maintain some 'good' paths (or similarly isolate bad nodes). Although this maintains good throughput in the network, it doesn't punish or discourage uncooperativeness.

Trust metric systems discourage uncooperative nodes by assigning low trust value to them. Nodes with low trust value will be given low priority when requesting for a resource from others. But trust metric has the problem that nodes with high trust value tend to get over-popular. Furthermore, a node requesting resources can't express the importance of the request, since the only information it provides is how trustworthy it is.

Economy or digital cash systems addresses load balancing and expression of the value of a resource. The system described in [13] relies on a central server for money management, so not very suitable for SAN. On the other hand, for the one in [9], although it doesn't rely on any central server, it depends on the existance of a tamper-proof hardware in every node to ensure the integrity of a money counter stored in it.

Authentication for SAN described in [5] relies heavily on administration nodes which provide the storage of all nodes' public keys. Although delegation of authority is provided, minimal nodes can get that authority, and it relies on those nodes being trustworthy. Aside from the fact that administration nodes might not exist in a purely community SAN, the average degree of proximity between the delegated nodes and the nodes to be authenticated is not very high, potentially causing high number management packets forwarded. On the other hand, the proposed Distributed Hash Table scheme has high distribution and replication of information, with some nodes only responsible for part of them (those which have high degree of proximity to it). Note that in the case of authentication, the information is of type <NodeID, public-key> pair.

Another advantage of SAN as deployment system for secure mechanisms is that we can assume that potential attackers or reverse engineers will behave rationally. That means they would only work on circumvention techniques only if it would benefit them, not just for the sake of destabilising the network. SAN software will be installed in an embedded device with some degree of tamper resistance. So unlike most purely software based peer-to-peer nodes, reverse engineering it won't be easy, and they're not going to do it if they won't benefit.

### 3 Research Methods and Plan

#### 3.1 Research Methods

The scheme that has been mentioned so far are very abstract. Protocols need to be defined and analysed for correctness, implementation strategy need to be laid out, and the technical and social issues in introducing an economy system need to be analysed.

A low overhead protocol for exchanging money needs to be defined. This protocol then has to be analysed with game-theory based formal analysis [14] to ensure robustness in the face of selfish and misbehaving nodes. This means the exchange protocol must be at least *rational* [14]. A *fair* exchange protocol guarantees that it's not possible for a node to disadvantage correctly behaving nodes. In a *rational* protocol, it's possible for a correctly behaving node to be disadvantaged, but the cheater won't be able to benefit from it.

Extra features that supports the economy system also needs to be designed. Examples are the contract signing, and price bidding.

Considering that sensitive information about a node is to be stored and replicated in many other nodes, we need assurance that no unauthorized alteration is possible, even by the host nodes. If this is unfeasible, we have to analyse if information replication and voting is enough to prevent misinformation.

Another question that needs to be addressed is if knowing which nodes host information that the corresponding user shouldn't be able to alter in some ways (e.g. the owner of the bank account can't deposit money out of nowhere) makes it possible for him/her to collude with host nodes, or direct some kind of attack to them.

Password capability system as described in [3] needs modification in order for it to work well in a decentralised system. Additionally, it needs to support necessary access rights, and be able to handle access to replicated objects for it to work in the described scheme.

Introducing an economy system into SAN can potentially create some problems which mirrors real world economy drawbacks, or some that only can happen in SAN. The worst cases need to be analysed, and the system should be able to handle it gracefully.

A simulation program that can mimic real ad hoc network usage pattern will be written in a high level language. The information storage and lookup system will be based on the Kademlia [11] DHT. Nodes will be simulated by communicating processes. Money exchange protocol and modified password capability system support will be written. The simulator will be used to compare the suitability of the different protocols and designs.

### 3.2 Timetable

Date	Activity
30 April	Research proposal submitted
1 May	Prepare draft of literature review
5 May	Work on money exchange protocol
3 June	Prepare for symposium
5 June	Symposium
6 June	Work on password capabilities modifications
11 June	First draft of literature review submitted
12 June	Do formal game theory and security analysis
30 July	Final literature review submitted
31 July	Work on simulation program and prepare thesis draft
10 September	Draft of the thesis submitted
24 October	Prepare for seminar
27 October	Seminar
4 November	Final thesis submitted
5 November	Work on project web site
11 November	Finalise project web site

## 4 Relevance of the Research

While SAN relies on the cooperation of its nodes to function, currently there is no incentive for them to cooperate. The aim of this research is to design a system that will promote cooperative behaviours, by introducing a virtual economy system. This gives the extra benefits of load balancing and accountability of resource usage.

The proposed system, unlike existing projects, avoids any centralised server and the use of special hardware protection. The result of this is that, if successful, the proposed system can not only be a reliable resource management and access control system in SAN, but in most decentralised systems.

## References

- [1] C. Kopp, R. Pose. Bypassing the Home Computing Bottleneck: The Suburban Area Network. *Third Australasian Computer Architecture Conference (ACAC)*, pages 87-100, February 1998.
- [2] R. Anderson, M. Kuhn. Tamper Resistance - a Cautionary Note. *In Proceedings of the Second Usenix Workshop on Electronic Commerce*, Oakland, California, November 1996.
- [3] M. Anderson, R D. Pose, C S. Wallace. A Password-Capability System. *The Computer Journal*, 29(1):1-8, 1 1986.
- [4] A. Bickerstaffe. Suburban Area Networks: Access Control and Management. Honours Thesis, School of Computer Science and Software Engineering, Monash University, Pages 50-62, 2001.
- [5] A. Bickerstaffe. Suburban Area Networks: Access Control and Management. Honours Thesis, School of Computer Science and Software Engineering, Monash University, Pages 45-49, 2001.
- [6] D A. Maltz. Resource Management in Multi-hop Ad Hoc Networks. Technical Report CMU CS TR00-150, School of Computer Science, Carnegie Mellon University, November 1999. Available from <http://www.monarch.cs.cmu.edu/papers.html>.
- [7] Li X., Ling L.. Building Trust in Decentralized Peer-to-Peer Electronic Communities. *Fifth International Conference on Electronic Commerce Research (ICECR-5)*, Montreal, Canada, 2002.
- [8] T. Grandison, M. Sloman. A Survey of Trust in Internet Applications. *IEEE Communications Surveys and Tutorials*, Q4 2000.
- [9] L. Buttyán, J-P. Hubaux. Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks. *ACM/Kluwer Mobile Networks and Applications (MONET)*, October 2003, Vol. 8 No. 5.
- [10] E. Sit, R. Morris. Security Considerations for Peer-to-Peer Distributed Hash Tables. In *IPTPS 2002*.

- [11] P. Maymounkov, D. Mazières. Kademlia: A Peer-to-peer Information System Based on the XOR Metric. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02)*, March 7-8, 2002, MIT.
- [12] S. Buchegger and J-Y. Le Boudec. Performance Analysis of the CONFIDANT Protocol (Co-operation of Nodes - Fairness in Distributed Ad-Hoc Networks). In *Proceedings of the ACM Symposium on MobiHOC*, Lausanne, Switzerland, 2002.
- [13] Mojo Nation, <http://www.mojonation.com>.
- [14] L. Buttyán, J-P. Hubaux. Rational Exchange - A Formal Model Based on Game Theory. *Proceedings of 2nd International Workshop on Electronic Commerce (WELCOM 2001)*, Heidelberg, Germany, 2001.