University of Southampton

Faculty of Engineering and Applied Science
School of Electronics and Computer Science

Communities of Trust in Pervasive Wireless Networks

Mini Thesis

By Michael S Saywell

Supervisor: Prof. David De Roure

Supported by a Doctorial Training Award as part of the Equator Interdisciplinary Research Collaboration, Engineering and Physical Sciences Research Council,

Grant number GR/N15986/01

Table of Contents

Introduction....4
Initial Research....6
- Communities Wireless Networks....6
- Wireless Network Technologies....6
  - 802.11....7
  - Bluetooth....8
- Mesh Networking....9
  - Bridging....10
  - Traditional Routing Protocols....12
    - RIP....13
    - OSPF....13
  - Ad-Hoc Routing Protocols....14
    - Location Based Routing....14
    - Fisheye State Routing....16
    - Clustering....16

Test bed network....18
- Issues Encountered....18
  - Throughput....18
  - Transmitter Location....18
  - Background Noise....19
  - Malicous Nodes....19
  - Latency....19

Sharing Connectivity....20
- InterIP....20
  - Slave nodes....21
  - Multihoming....22
  - Drawbacks....22

Trust....23
- Existing Trust Mechanisms....23
  - X.509....23
  - PGP....25
  - SPKI/SDSI....25
  - KeyNote....27
- Components of Trust....28

Transitivity....28
Reputation....28
Wireless Security Mechanisms....30
802.11....30
MAC Filtering....30
WEP....31
802.1x....31
WPA and 802.11i....32
Bluetooth....32
Wireless Authentication Service....33
A Community Trust Service....34
Results....35
Proposed Improvements....36
SPKI Wireless LAN Authentication....36
Requirements....36
Initial Protocol Design....37
Limitations....38
Toward a distributed model....40
Preliminary Analysis of the PGP Web of Trust....41
Strongly Connected Components....42
Resilience to Attack....44
Conclusion and Future Work....45

Introduction

Pervasive networking capability is a requirement for complete physical-digital integration and wireless is the obvious medium for this. Existing wireless networks come in many forms, ranging from cellular networks spanning hundreds of kilometres, through enterprise WiFi networks, localised WiFi hot-spots and home networks covering a single building and finally personal area networks spanning tens of meters. In such networks security is typically only possible when either the entire infrastructure is owned and managed by a single organisation (cellular, Enterprise WiFi and hot-spots), or when there is a single or small number of users (home networks and PANs). In the former a central database is used containing all user credentials, in the latter a shared secret is typically used.

Neither of these approaches is suitable for sharing connectivity. For example, close friends and family members may want to give each other access to wireless networks in their home without running open networks which anybody can connect to. A shared password is not desirable, and a central database is not practical for the home network.

Although I have introduced my research as being primarily concerned with security and trust, my initial work was more concerned with routing protocols and radio issues arising from so called "meshed networks". A mesh network is one where access points forward packets to each other over wireless, allowing otherwise separate access points to combine into a single network providing seamless mobility and hand-overs.

This approach eliminates some of the security problems, for example one can envisage a mesh network where traffic between neighbouring nodes is forwarded freely but traffic off the mesh is subjected to local firewalling and authentication. In contrast to when an access point is stand-alone and must authenticate each and every user. Whilst this side-steps the security issues rather than addressing them, it is an interesting solution to the problem, with the added advantage of providing complete blanket wireless coverage.

The remainder of this report is structured as follows:

1. Background research in wireless networking technologies and mesh routing.
2. Testbed network deployment and results
3. Refocus of research into sharing connectivity

4. Background research in trust mechanisms and implementations in the wireless area.
5. Application of ad-hoc routing mechanisms to trust problems
6. Conclusions and future work

Initial Research

The desire for high speed, low cost, pervasive wireless data connectivity has never been greater. Indeed it is so widely recognised that people have set up community wireless network initiatives worldwide[1]. These networks vary in their goals, some intend to build wireless backbones whilst others aim to provide coverage by sharing private broadband connections. Such networks are typically run open (i.e. without security) or use MAC based filtering and a web portal such as NoCat[2], which will be discussed further later.

In this section I discuss the different technologies and research relevant to the area of pervasive wireless networks.

Wireless Network Technologies

There are essentially 2 approaches to providing wireless network connectivity, firstly the mobile phone cellular networks which span entire countries or continents and secondly the short-range, low power 802.11 home/business wireless network covering inside buildings and small outdoor areas. 802.11 networks have data rates far in excess of those currently offered by mobile networks, a trend that can only be expected to continue due to issues such as coverage area and battery life. In the pervasive world devices are likely to be just as interested in connecting to local services as those on the Internet, for example a wireless digital camera sending pictures to a nearby display and then uploading them to a website. Indeed many of the next generation mobile phone handsets are set to have integrated GSM, WiFi and Bluetooth.

It seems that if we could better utilise these smaller, local wireless networks that in urban areas especially we would be able to benefit from better, cheaper connectivity than is available with a cellular network. There are many different wireless protocols, the following provides an overview of the current most popular standards and their capabilities.

802.11

In 1997 the IEEE ratified the ANSI/IEEE 802.11 specification[1], specifying the MAC and Physical layers of a wireless communications protocol. This initial standard

1 FreeNetworks.org - http://freenetworks.org/

2 NoCatNet - http://nocat.net

supported data rates of 1 or 2Mbit only, deployment was fairly sparse, typically restricted to industrial and warehouse environments. It included 3 physical layer definitions, 2 in the 2.4GHz spectrum and 1 using IR.

In 1999 the standard was extended, adding 5.5 and 11Mbit data rates to the DSSS (Direct Sequence Spread Spectrum) physical layer, this standard is the now widespread 802.11b[2]. 802.11b has been hugely successful, with demand driving new faster standards running at up to 54MBit. All of the standards utilise license-free parts of the radio spectrum, either 2.4GHz (802.11, 11b and 11g) or 5Ghz (802.11a). These public frequencies are limited to low transmit power levels in many countries, for example 100mW in the UK[3]. Communication range stretches from under 100m indoors to several kilometres given line of sight and suitable directional antenna.

A typical 802.11 deployment consists of one or more access points connected to a common wired backbone, wireless clients connect to the access point with the strongest signal and can freely migrate between APs with no loss of connectivity. There is no direct communication between the access points via the wireless medium, handovers are handled via the wired backbone.

There are also two alternative modes of operation defined by the specification:

The first is Ad-Hoc mode. This is a standard feature present in client adapters, it is designed to allow communication to occur directly between clients without a co-ordinating access point. The radio layer assumes that all clients are within range of each other and thus does not re-transmit packets, although it is possible to over-come this restriction by forcing hosts to re-transmit at the IP or Network layer. Ad-Hoc mode is not ideally suited for large networks as the stations attempt to form a virtual network called an IBSS. Each IBSS has an ID taken from the MAC address of the station which created it, stations which join the IBSS use this ID to identify it and provided that at least 1 node remains running the ID will continue to exist. Problems arise when merging multiple IBSS networks, for example when 2 initially disconnected networks become linked by a new host then they will not merge as one might expect, instead the joining host must choose one network ID or the other.

---

3 UK Interface Requirement 2005:
http://www.ofcom.org.uk/static/archive/ra/publication/interface/word-pdf/ir2005.pdf

The second mode is wireless bridging or WDS, this feature is present in most access points and is often used to connected 2 wired networks together via a point to point wireless link. Implementations typically have 2 modes of operation, point to point and point to multi-point, only access points in point to point mode can connect to those in multi-point mode, giving rise to star topology networks. More advanced implementations allow arbitrary links between access points, with each access point running spanning tree to create a single Ethernet network.

Bluetooth

Bluetooth is primarily aimed at personal area networks, the specification[3] includes various profiles such as headset, phone and computer which devices can assume and then communicate directly with each other. For example any Bluetooth headset should work with any other Bluetooth phone, provided they both comply with the specification. Bluetooth was designed to run on small low-power devices and, consequently has a very low throughput compared to recent 802.11 standards, the total capacity of a Bluetooth link being just 1 MBit. Transmit powers are also typically very low, the specification defines 3 classes of device operating at 1, 2.5 and 100mw, however only PC powered dongles or access points typically operate in the latter range.

All Bluetooth connections are made via Piconets, a Piconet is simply 2 or more devices which occupy the same physical channel. 1 device in the Piconet is the master and the rest are slaves, all communication takes place via the master device (i.e. slaves cannot communicate directly). A device can participate in multiple Piconets, but can only ever be the master of 1. Multiple Piconets linked in this way are termed Scatternets.

Although the Bluetooth design acknowledges these ad-hoc Scatternets, the specification does not include details of how they should operate, thus there are few if any implementations. The low data rates and short range offered by most Bluetooth devices, has resulted in 802.11 becoming the dominant technology for wireless networking at the IP layer. However Bluetooth continues to be popular as a "cable replacement" technology on phones, PDA's and other personal devices.

Mesh Networking

In his paper[4] Shepard introduces the concept of a rooftop network, such networks use low power wireless equipment placed on the rooftops of many buildings to communicate with other nearby nodes as shown in Figure 1.

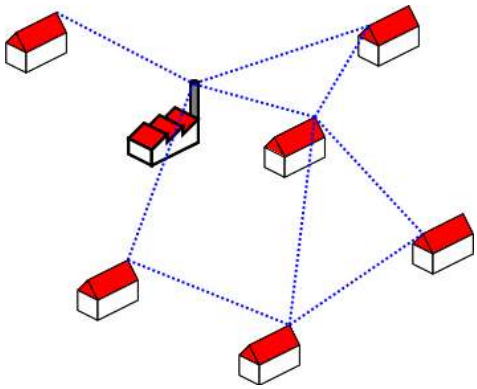

Figure 1: A typical mesh network

Nodes forward traffic co-operatively for each other, thus facilitating communication between all nodes on the network even though they may not have a direct connection with each other.

This is the area in which I initially focused my research, I intended to see if such networks were practical, what routing protocols would be suitable and how they scaled. The goal was to create an open infrastructure which could provide wide area wireless coverage.

Security would be managed on a per service basis much like the current Internet. For example to get Internet access you could setup a secure VPN connection across the network to a trusted gateway, perhaps located in your own home or business. In this way you have wireless coverage across a large area without having to authenticate users on a global scale.

Based on the community wireless model no central organisation would deploy this network, rather individuals and businesses would put up nodes on their own property to gain access. Thus when a user gains access they also simultaneously expand coverage allowing more people to connect. This approach requires a cheap, license-free radio technology and a routing protocol with good scalability.

Bridging

The simplest approach is not to use routing at all but rely on standard Ethernet layer 802.1d bridging[5], in fact many smaller mesh networks take this very approach. Bridging has the benefit of allowing clients to migrate between access points since it is effectively just a single flat network, however bridging introduces introduces in-efficiencies in all but the simplest network topologies.

The problems arise from the fact that loops in Ethernet are prohibited, therefore a mesh network that wishes to present itself as a flat network must also be loop free. The spanning tree protocol used in 802.1d will disable links in a network such that a topology with loops becomes a tree instead. If one of the links in use should go down the algorithm runs again and a previously disabled link will become active to maintain connectivity.

Spanning tree is a tried and tested mechanism for allowing redundant links between core bridges in a network, however it does not apply well to the wireless scenario. Consider the fully-connected topology shown in Figure 2.

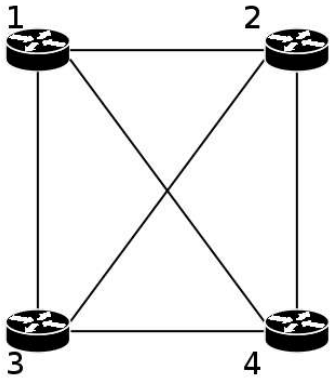

Figure 2:
Fully Connected Topology

We cannot run this topology as a flat network since it contains loops therefore we must enable spanning tree, this could create several different topologies, of which Figure 3 is an example.

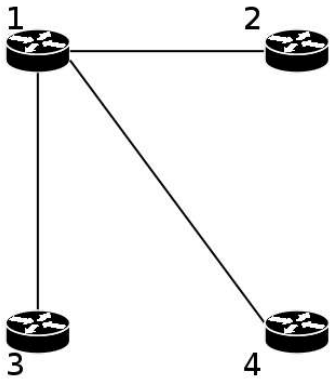

Figure 3:
Bridged Topology

Now each of the access points can be safely bridged together into a single Ethernet segment, however this topology will restrict the available bandwidth between some of the access points, for example nodes 3 and 4 previously had a direct link but now traffic must be relayed via node 1.

In a wired network this is of little consequence, it will use up some of the switching capacity on node 1 and latency will be marginally higher but neither of these are likely to cause problems. However in the wireless case this relaying will halve the throughput of the network. This is due to wireless being a shared medium like an Ethernet Hub, since all the nodes are operating on the same frequency (which they must in order to exchange data) only 1 may transmit at any time.

Thus node 4 will actually receive traffic from node 3 twice, the first time it will be discarded (as it is addressed to node 1), and then secondly once it has been relayed by node 1.

Hence bridging is a very inefficient solution, to better it we must move higher in the network stack and make use of IP layer routing protocols.

Traditional Routing Protocols

Traditional large scale routing requires a strict hierarchical architecture in order to scale efficiently, as shown by the aggregation typically found between Department, Organisation, ISP and the Internet. Without such aggregation the Internet could not function, as of August 2004 there are 140,606 routing prefixes announced[4] compared to

4 Source: http://www.cidr-report.org/

an estimated 233,101,481 hosts[5]. Thus on average there are approximately 1,600 hosts for each prefix announced, an aggregation of more than 3 orders of magnitude.

It should also be noted that many still consider the current routing table to be un-acceptably large, consequently with IPv6 additional levels of aggregation are encouraged to incur less routing overhead in the Internet core.

However in a meshed network very little aggregation is possible as addresses are usually allocated at random and the hierarchical nature of physical connectivity found on the Internet is not present. I have provided a brief overview of two popular routing protocols, RIP and OSPF.

RIP

Wired network routing protocols typically work using the Bellman-Ford[6][7] or "Distance Vector" algorithms, one of the simplest protocols based on this algorithm is RIP[8].

In RIP each router keeps a table containing for each destination:

- Gateway: the first gateway along the route to the destination.
- Interface: the physical network which must be used to reach the first gateway.
- Metric: a number, indicating the distance (number of hops) to the destination.
- Timer: the amount of time since the entry was last updated.

Routers send updates to each other by exchanging these tables. If a router receives an update for a destination with a lower metric than it is currently using it will replace that entry in it's table, so as to always use the shortest path.

RIP works well for small networks, however as the size increases 2 problems are encountered. Firstly the size of the tables exchanged hosts grows unacceptably large and secondly whenever the topology changes (such as following a link failure) a routing update is triggered. These two issues combined means that in larger and more dynamic networks the routing table can consume a sizeable amount of the total bandwidth available.

5 Based on the number of DNS hostnames, Source: http://www.isc.org/ops/ds/reports/2004-01/

RIP is also limited by design to networks with a maximum diameter of 15 hops, this is to prevent routing loops from forming during updates.

OSPF

In OSPF[9] each router maintains a database describing the entire network topology. From the topology database each router constructs a tree of the shortest path to each router using the Dijkstra algorithm[10]. This is used to select the next-hop router when forwarding packets.

OSPF is a Link State protocol, meaning that routing updates are triggered by network links being operational or not. When a change occurs it is propagated to all other routers.

The drawbacks of OSPF are that it has a high router overhead, in unstable or frequently changing topologies a router may find itself continuously re-building the shortest-path tree. On a graph with m edges and n vertices Dijkstra (and thus OSPF) has been shown to run in either $\Theta(n^2)$ or $\Theta(m + n \log n)$ for smaller graphs[11].

Ad-Hoc Routing Protocols

These traditional protocols would quite plainly not be adequate since I was interested in protocols that could scale to cover an entire city and beyond, so I began to look at alternative approaches.

The majority of recent routing protocol research has taken place in the MANET[6] area, with an emphasis on scalability and good response to rapid topology changes.

Ad-hoc routing protocols fall into 2 camps, pro-active or table driven and re-active or on-demand. A pro-active router knows all the possible destinations in the network in advance, RIP and OSPF are both pro-active. A re-active router only seeks a destination once it receives a packet for it, this can help scalability at the expense of increased initial latency.

---

6 IETF Mobile Ad-hoc Networks Working Group:
http://www.ietf.org/html.charters/manet-charter.html

Location Based Routing

The use of geographic data for scalable routing in large scale networks has been considered before. It was first discussed by Gregory Finn in 1987[12] as a mechanism for dealing with Metropolitan scale networks by using latitude and longitude as components of a hosts network address. Routers in such networks are aware of their position and that of their neighbours, when they receive a packet the destination address tells them the destination host's physical location.

Finn envisaged a scenario where city blocks were linked to each other directly rather than via the Internet and an ISP as is commonplace today. Although Finn was considering wired links the basic principles of geographic forwarding that he outlines are equally applicable to wireless medium.

His proposal was that packets are forwarded greedily, that is to a neighbour nearer to the destination than the current node, until no such node exists or the packet has reached it's destination. If no closer node exists then it is assumed that there is an obstacle and the current node floods the packet to all of it's neighbours until a node with a neighbour closer to the destination is reached and greedy forwarding can be resumed, as shown in Figure 4.

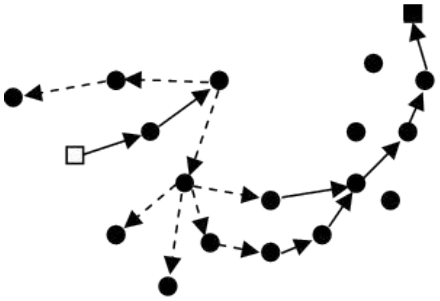

Figure 4: Cartesian Flooding

Solid lines represent greedy forwarding and dotted lines flooding. Duplicate packets can occur when a flooded packet arrives at 2 closer nodes.

Work since then, such as GPSR[13] has worked in the constraints of traditional 32 bit IP addressing which is too small to encode geographic data, instead protocols have relied on a dedicated location server which routers query to find the physical location of a

destination. Convergence of geographic addressing with IP and in particular IPv6 is briefly discussed in GeoCast[14], IPv6 addresses are sufficiently large that it's possible to encode the location data in the address itself. In his Internet Draft[15] Tony Hain demonstrates this very principle, showing how geographic locations can be used to form a provider independent address. As an example 20 bits of address space can be used to cover a region of approx 26 square km at a resolution of 6.4m square. The mechanism detailed also makes use of bit interleaving, meaning that shorter prefixes cover large areas, as more bits are used the accuracy increases. For example 16 bits refer to an area 104km^2 in size, 32 bits 407m^2 and 44 bits just 6.4m^2.

GPSR is neither truly pro-active or re-active, whilst it is not table driven it does not have to search for each new route.

Fisheye State Routing

Fisheye State Routing (FSR)[16] is based on a link state protocol, enhanced by propagating updates less quickly as the distance from the source increases. If mobility is high then obsolete data will be dropped at outer nodes due to these forwarding delays, consequently routing updates do not flood the entire network. Although distant nodes may have outdated routing information as packets approach their destination the accuracy of the routing tables increases, compensating for this.

Fisheye is primarily designed as a mechanism to reduce routing traffic in networks with high mobility rates which is not a large issue in rooftop mesh networks, however the design philosophy is an elegant one which may be of use in different contexts.

Clustering

Clustering has been used in several protocols[17],[16] and is an extension of the clubs algorithm[18]. Clustering is the process by which individual nodes group themselves dynamically, there are several mechanisms to do this, including simply emulating the wireless topology (where clusters represent a group of directly connected nodes), or by some algorithmic method such as that described in clubs.

In HSR the clustering process is repeated recursively to form multiple layers of "clusters", Level 0 is the physical layer, Level 1 clusters comprise solely of the Level 0

leaders (cluster heads), Level 2 of the Level 1 leaders and so on, 2 levels of clustering are shown in Figure 5. Clusters are comprised of 2 basic node types, gateway nodes (hollow circles) and internal nodes (solid circles) and cluster heads (solid squares). Gateway nodes are present wherever 2 clusters meet and are responsible for forwarding traffic between the 2 cluster heads.

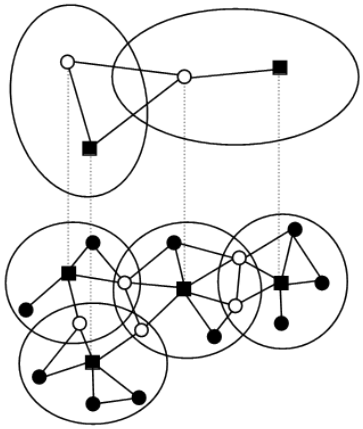

Figure 5:
Hierarchical Clustering

Addressing in HSR is directly linked to the hierarchical structure and is defined as being the sequence of MAC addresses of nodes that must be traversed to reach the destination node from the highest level.

Test bed network

We have deployed a small test bed network as part of a local community wireless initiative called SOWN[7]. The network has provided us with invaluable data and hands-on experience of the problems one can expect to encounter at the wireless layer.

Issues Encountered

- **Throughput**

The biggest realisation was the poor levels of throughput one can expect from a meshed network, when a packet is being forwarded across many hops only 1 node in 3 can transmit at once, see Figure 6.

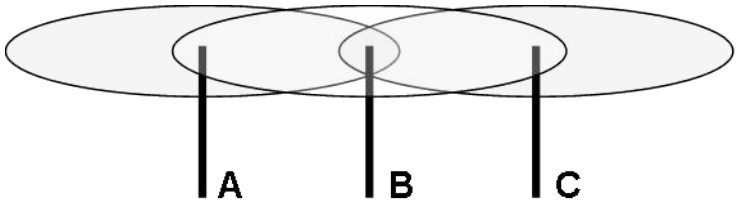

Figure 6: Overlapping Transmissions

The net effect of this is that with standard 802.11b throughput is reduced to approximately 1Mbit of TCP traffic across many hops, and that bandwidth is shared across surprisingly large regions of a mesh network.

By using multiple radio's in each node one can reduce or even eliminate these problems, however when using multiple radios the antenna must be positioned some distance from each other to avoid cross-channel interference due to physical proximity.

- **Transmitter Location**

Since the transmit powers used in the 2.4GHz range are heavily limited to get long-distance communications requires line of sight between endpoints. This typically requires external aerials and a significant installation process, far more than is reasonable if wide-scale deployment is required.

- **Background Noise**

As the number of transmitters in the system increases so will the background noise,

7 Southampton Open Wireless Network – http://www.sown.org.uk

making connections between nodes less reliable and reducing throughput.

- **Malicious Nodes**

As with many distributed systems it would be very hard to detect and isolate a malicous node (for example advertising a false default route) automatically.

- **Latency**

The process of encoding and de-coding a wireless signal takes time, approximately several ms before transmission delays are considered. This is because checksums must be calculated on the entire packet before it is transmitted. Thus in a large network with many nodes communications over many hops are likely to be high in latency.

Sharing Connectivity

My interests lie primarily in creating a de-centralized "organic" style network providing ubiquitous connectivity and it was apparent that a well performing wireless mesh would require a high level of design and careful deployment. Thus I began to look at alternative approaches to achieve this goal.

Until now I had ignored the increasingly widespread availability of broadband Internet connectivity since providing other users with access is often against the contract used by an ISP. What's more you are held personally responsible for the actions of that user, for example if your connection was used to send spam email, it is likely your ISP would hold you in breach of contract.

I devised a system called InterIP which allowed users to connect via the Internet to their home network, from which they can use their own Internet connection. Thus if a user does send spam email it will be via their own connection, and they will be the liable.

InterIP

First we must reserve some IP address space for InterIP, for example 10.0.0.0/8. This space is split, let us assume into 256 class C networks. There must also exist a reverse DNS infrastructure for 10.0.0.0/8, although it can appear anywhere in public DNS, e.g. 10.in-addr.sown.org.uk

For each class C network there exists one or more InterIP lease servers. An InterIP lease server is effectively a DHCP server for an entire network block, we shall assume the blocks are /28 networks (16 IPs). As with DHCP leases must be renewed periodically or they may be assigned to other nodes.

A node is simply a router running InterIP, there are 2 types of node, those directly connected to the Internet with at least 1 public IP address are gateway nodes, those without a public IP are slave nodes and must be connected to at least one gateway node via a LAN. Slave nodes will be discussed in more depth later, until then we shall use the term "node" to mean a "gateway node".

When a new node is installed it selects an InterIP lease server, this could be random or

based on some property such as geographic location, the discovery and selection mechanism is not important provided it evenly distributes nodes between lease servers.

The node requests a network allocation from the InterIP lease server, if granted server publishes a forward DNS entry corresponding to the public IP of that node for each of the reverse hostnames in that block.

For example if the lease was for 10.1.2.0/28 and the node resided at the public IP 123.123.123.123 then the entries would be:

```
0.2.1.10.in-addr.sown.org.uk.       IN A  123.123.123.123
1.2.1.10.in-addr.sown.org.uk.       IN A  123.123.123.123
2.2.1.10.in-addr.sown.org.uk.       IN A  123.123.123.123
...
15.2.1.10.in-addr.sown.org.uk.      IN A  123.123.123.123
```

When another node receives a packet destined to a non-local InterIP address it performs a forward DNS lookup corresponding to the destination, this will return the public IP of that node. The InterIP packet is then encapsulated into a normal IP packet and sent to that public address. Upon receipt of such an encapsulated InterIP packet it is first checked to ensure that the destination of the InterIP packet is in a valid and local InterIP range, if this requirement is met then it is simply de-encapsulated and forwarded according to the local routing table.

Slave nodes

If a node does not have a public IP then it is deemed to be a slave node and must be connected to one or more gateway nodes e.g. via a local area network. The slave node will request the gateway node for transit, if granted the gateway node will contact an InterIP lease server on behalf of the slave, and an additional InterIP block will be assigned and routed via the gateway.

Multihoming

If gateway nodes are connected via some other means than the Internet such as a community wireless network they may choose to perform multi-homing for each others

blocks. Gateway 1 requests transit from gateway 2, if granted then gateway 1 will contact the InterIP lease server instructing it to add additional DNS entries for the public IP of gateway 2. Reusing the previous example, if the public IP of gateway 2 is 99.98.97.96 the DNS zone file would now look like:

```
0.2.1.10.in-addr.sown.org.uk.       IN A  123.123.123.123
                                    IN A  99.98.97.96
1.2.1.10.in-addr.sown.org.uk.       IN A  123.123.123.123
                                    IN A  99.98.97.96
2.2.1.10.in-addr.sown.org.uk.       IN A  123.123.123.123
                                    IN A  99.98.97.96
```

Now when a node in a different IP network receives packets to 10.1.2.2 it will perform the lookup as before but this time receive multiple gateways, encapsulated packets can then be distributed between these gateways increasing the overall bandwidth available between the 2 sites.

Drawbacks

A prototype implementation (without the lease server) worked well, building a virtual network over the Internet in the 10.0.0.0/8 address space. The drawbacks are an inefficient use of bandwidth and high complexity for users as they would require an InterIP aware router and VPN server on their home network.

Trust

It was at this time that I decided I had perhaps been avoiding the main issue , namely of allowing people to share their Internet connectivity directly with other trustworthy individuals.

To demonstrate the security issues let us assume that everybody runs their wireless network "open" (without authentication) such that any client can connect. The first problem is that you are allowing unknown users direct access to your own internal network, a wireless LAN could be deployed outside a firewall, but typically you want to be able to access internal resources from your own wireless network so this is counter-productive. The second problem is that of providing anonymous Internet access, if somebody uses your Internet connection to download illegal content, hack into a server, send spam email or any number of other uses prohibited by your ISP then you are likely to be held responsible and would be unable to provide little conclusive evidence otherwise.

We want to avoid locking down the network totally though, we want our friends, family and colleagues to be able to connect without having to setup individual devices or give out passwords, in general we want to provide access to people that we trust not to abuse the network. Ideally the system could authenticate people that are trustworthy despite them being not directly known, friend's of a friend and friend's of a relative are obvious examples.

Existing Trust Mechanisms

The two most common trust mechanisms in use today are X.509 and PGP, a more recent development is SPKI/SDSI however this is yet to see wide-spread deployment. In this section I will consider these and other security mechanisms.

X.509

X.509[19] is the traditional Public Key Infrastructure, first developed in 1980's as part of the X.500 specification which was an attempt to build a global "Internet Phone book". X.509 is based around a global name space known as the Directory Information Tree (DIT) which is comprised of many unique Distinguished Names (DNs). The purpose of an X.509 certificate is to bind a Distinguished Name to a public key, this binding is

signed by a trusted third party known as the Certifying Authority (CA). Only a single signature is supported.

An X.509 certificate includes:

- Distinguished Name
- Public Key
- Expiry Date
- Digital Signature of the Issuer
- Distinguished Name of the Issuer

To find another user in X.500 you would simply consult the directory. However the global X.500 initiative failed, thus there is no standardised way to discover another users certificate. X.509 is generally considered to be overly complex and incomplete for the following reasons:

1. The certificates are not human-readable and there are several different standards for encoding them, this makes them annoying to handle and use.

2. CA's periodically produce Certificate Revocation Lists. These are lists of Certificates which have been revoked, typically either because their associated private key has been compromised or the information it contained was found to be incorrect. Clients should check CRLs before accepting a certificate, however this is often poorly implemented. For example certificates issued by Verisign do not include the CRL Distribution Point field defined in RFC2459[19] required for automatic checking.

3. Certifying Authorities typically do a minimal amount of checking to verify a users identity. Thus limiting the usefulness of a personal certificate.

PGP

PGP[20] differs from X.509 in that it is a totally distributed system, anybody can sign anybody else's key thus everybody is in effect a type of Certifying Authority. PGP relies on having a large numbers of digital signatures, known as the web of trust, to compensate for the fact that users are not trained to check or verify each others identity.

A PGP certificate includes:

- A unique ID
- A name (typically an email address)
- A PGP public key
- Digital signature of the users private key
- Digital signatures made by other users who have validated this certificate

Users have a keyring which is used to store the public keys of others. Each key has two associated attributes, whether or not it's authentic and the level of trust the user places in that key.

The levels of trust and default behaviours are as follows:

- Yes - Accept and use certificates signed by this public key
- No - Don't use this key
- Maybe - Prompt
- Marginal - Requires several (by default 2) marginal keys to make the authentication

PGP was designed solely as a means to provide secure email, it is concerned only with proof of identity and not more generic problems such as trustworthiness.

SPKI/SDSI

As its name suggests, SPKI/SDSI[21][22] is the combination of 2 specifications. SDSI [23] was a proposal by Butler and Lampson, which took a "clean slate" approach to security, attempting to deal with some of the problems discovered with X.509. Their work was subsequently integrated into the IETF Simple Public Key Infrastructure Working Group[8] and the resulting specification termed SPKI/SDSI or SPKI for short.

Perhaps the most defining aspect of SPKI is it's use of local name spaces, each user maintains their own mapping from principals (public keys) to names. A similar mapping exists in PGP, however in PGP it is the key owner who defines the name rather than the list owner. In SPKI it is the responsibility of each individual to assign names much like one would specify the names in a personal phone book. These "phone books" may be made public, in which case it is possible to build relative names, for example "the person

8 IETF Simple Public Key Infrastructure Working Group: http://www.ietf.org/html.charters/spki-charter.html

whom Alice refers to as Bob".

By using local name spaces a CA is no longer required to manage the global name space to avoid clashes and more meaningful names can be assigned by users to their to keys. The removal of the CA means that there is no central point of failure, and individuals are free to set their own criteria before considering another key to be trustworthy.

SPKI defines Authorisation Certificates which allow permissions to be passed from one key to another, the certificates have a delegation bit which denotes whether the permission may be passed on to other keys. These certificates bind a Tag to a public key, a tag can be almost anything and simply represents a particular permission. ACL's can be used to reason about authorisation certificates, although they are not part of the standard.

SPKI also permits groups to be defined within the certificate structure, allowing a user to apply ACL's to multiple keys quickly and easily.

SPKI is oriented toward being online, that is it favours on-demand re-validation over static CRL lists. Users may provide online directories containing their namespace and current Authorisation Certificates, these can be traced back to the originating key to confirm legitimacy.

SPKI contains a fault tolerance method known as "K of N" or "Threshold", this can provide extra security by requiring that a key has been delegated an authorisation via a number of independent sources. For example if K is 2 and N is 3 and the subjects are Alice, Bob and Carol then the user will have to present 2 different certificate chains each of which pass through a different one the subjects specified.

KeyNote

KeyNote[24] is an evolution of PolicyMaker[25], which unlike the previous mechanisms defines a framework or language in which permissions can be defined, delegated, queries and evaluated.

Although SDSI has the scope for such a language it does not specify it, allowing implementations to craft their own. The KeyNote approach is precisely the opposite,

allowing any KeyNote aware tool to evaluate and validate securtity permissions.

KeyNote consists of several core components:

- A language for describing actions which are controlled by the system
- A mechanism for identifying principals which are entities that can be authorised to perform actions
- A language to describe policies which govern the actions a principal may perfom
- A language to specify credentials which allow principlas to delegate authorisations to other principals.

These components are wrapped by a compliance checker which applications interface with to verify whether a requested action should be granted.

A web of trust could be created in KeyNote in a similar fashion to SDSI, by requiring a delegation of permission from one or more well known keys and relying on unknown intermediate keys in the web.

The specification does not address the means by which credentials and principals are obtained. It uses a traditional global model for identities.

Components of Trust

Transitivity

The transitivity of trust has been an area of much discussion, the traditional perspective being that trust is not transitive, in other words if Alice trusts Bob and Bob trusts Cathy then Alice cannot trust Cathy. I consider this definition to be too clear cut as very often trust is transitive, indeed the entire structure of x.509 certificates is built upon this fact (although some may consider this to be a flaw). For example in x.509, when a client trusts a Certificate Authorities they are also indicating trust of every certificate issued by that CA.

Quite often if we are venturing into something unknown we will rely on the recommendations of those we already trust. In the trust domain transitivity is conditional, it depends on the number of recommendations received, the trustworthiness

of those doing the recommendations and the importance associated with this relationship. For example users may consider a large number of recommendations from less well known individuals the same as a small number from close friends or family.

Reputation

Reputation as a mechanism for determining trust in distributed systems is a well proven method for establishing trust as websites such as EBay have shown. EBay provides a "Feedback Forum" where after completing an auction the buyer and seller can give each other either a positive or negative point, thus trustworthy sellers quickly gain a high positive score allowing bidders to buy with greater confidence. Sellers with low quality goods or that provide bad service quickly gain negative feedback rendering their account useless as few people are prepared to bid from a seller with poor reputation.

The Ebay system is not without it's flaws, there is nothing preventing a malicious seller from setting up a new account after each auction to avoid negative feedback. Likewise they can also create false reputations by running several low value auctions, bidding for these items from other factious accounts and giving themselves positive feedback. Although this would cost a small amount in listing fees it would be easily recouped by setting up several simultaneous high-value auctions.

These problems and others stem from the fact that in reputation in Ebay is anonymous, there is no mechanism to trace the path between buyer and seller. For example if you knew several other people who had bought from a buyer with success your confidence would be much higher. Taking this a step further it is not necessary to directly know the previous bidders, if a user you have dealt with before has also dealt with this buyer that still provides multiple independent paths between yourself and the bidder can be identified then additional trust can be derived from this.

Several Certificate Authorities have begun programmes to support this type of notion. Both Thawte[9] and CACert[10] offer free personal email certificates, however the certificates are provided with the name field left blank. To get these details filled in they offer a public assurance program, this works by relying on members of the public to certify each others identity, this works in much the same way as PGP key signing.

9 Thawte web of trust, http://www.thawte.com/html/COMMUNITY/wot/index.html

10 CAcert assurance programme: http://www.cacert.org/index.php?id=3

Users are encouraged to meet face to face with some photographic ID to certify each others identity, they can then log on to the CA's website and record this. After several successful meetings the users identity is assumed valid and their certificate is updated to include personal details.

In comparison the security of PGP is based around this concept from the ground up, its certificates reflect this with their ability to accept multiple signatures. Thus each time a user verifies the identity of another they sign the others certificate directly. With the personal certificates this information is only available via central website and not stored on the certificate itself.

Reputation in general has been well discussed in [26][27], and is generally considered to be an excellent means of gaining trust. However neither PGP nor Personal Certificates offer any further information as to the nature of any meeting or signing. For example there is no way to discriminate between close friends and family or somebody you meet on the train. You may be equally sure of their identity (assuming the train passenger is carrying ID) so it would be appropriate to assign them equal ratings, however you would not share the same level of trust with them.

Wireless Security Mechanisms

In this section I will consider how security has implemented in different wireless technologies.

802.11

802.11 includes several security mechanisms, which I will discuss below.

MAC Filtering

This is the most basic form of security, access points can be configured with ACL lists restricting the MAC addresses which can connect. This is relatively easy to overcome as an attacker can sniff traffic to find a working MAC and hijack it. Despite these flaws MAC filtering is widely used as it requires no client side support or setup.

In many environments MAC filtering is typically deployed alongside a web login which dynamically adds allowed MAC addresses after a successful login. This is worrying

since if a malicious user hijacks another's connection then all the logs will point to the innocent party thus falsely incriminating him.

An open source web login known as NoCat is commonly used in community wireless networks, it too provides dynamic MAC address filters. There are several different back-end authentication mechanisms including RADIUS and their own protocol implemented using GnuPG[11] and SSL.

WEP

The original 802.11 specification includes a Wired Equivalent Privacy (WEP) algorithm which is widely acknowledged to have several flaws[28]. The most relevant of these is that the protocol relies on a shared secret configured on both client and access point, thus restricting it's usefulness in deployments of more than a few people. There are also cryptographic flaws, the details of which are not relevant here, suffice to say that given adequate data (which can be gathered by simply listening to traffic), it is possible to determine the shared secret.

802.1x

In June 2001 the IEEE approved the 802.1x[12] standard which defines "Port Based Network Access Control", a generic authentication system for networks. Originally designed for use in wired networks it has proven workable in the wireless realm. 802.1x defines extensions to EAP[29] (Extensible Authentication Protocol) called EAPOL (EAP Over LAN) allowing it to work directly at the network layer. Authentication is handled by a back end RADIUS server, allowing each user to have a unique user name and password.

EAP itself supports several numerous different authentication mechanisms, including MD5, TLS, TTLS, PEAP. Encryption in wireless networks is provided by a rotating WEP key, with each client having a different key thus preventing snooping between clients.

---

11 GNU Pricay Guard, an open source OpenPGP implementation: http://www.gnupg.org/

12 IEEE 802.1x: http://standards.ieee.org/getieee802/download/802.1X-2001.pdf

WPA and 802.11i

WEP and 802.1x are soon to be replaced with 802.11i[13], which at the time of writing has recently been ratified. WPA[14] (Wi-Fi Protected Access) was conceived as a stop-gap solution by the WiFi Alliance[15], it was effectively a snapshot and subset of an earlier draft of 802.11i. WPA has now been superseded by WPA2 which fully implements the 802.11i specification. 802.11i specifies encryption methods superior to those used by WEP and includes support for 802.1x as standard.

802.11i defines several new security algorithms under the heading of "Robust Security Network Association" or RSNA. An RSNA can be established using either pre-shared keys or by using IEEE 802.1x. 802.11i is primarily concerned with the definition of new encryption algorithms to achieve the RSNA and address the flaws found in the RC4 based WEP methods.

802.11i uses Open System authentication, this means that any station can connect to the access point at the most basic level although they will not be able to exchange traffic until an additional authentication mechanism has been completed. Thus it is possible for denial of service attacks to be launched, whereby an attacker masquerades as a legitimate user and sends a de-association to the access point, resulting in the legitimate user being disconnected. A flood of such de-association requests would prevent use of that access point. Whilst some people consider this a major shortfall others point out that an equally effective denial of service attack would be to flood the radio channel with noise against which there is no protection.

Bluetooth

Bluetooth security is based on a challenge-response scheme. One party (the verifier) sends a challenge (a random number) to the other party (the claimant). The claimant calculates a response which is a function of the challenge, it's own Bluetooth address and a secret key. The secret key comprises of a user defined PIN, a random number and the other devices Bluetooth address. This process is termed pairing.

This initial key is used to exchange 2 more random numbers (generated by each device)

13 IEEE 802.11i: http://standards.ieee.org/getieee802/

14 WPA: http://www.wi-fi.com/OpenSection/protected_access.asp

15 The WiFi Alliance: http://www.wi-fi.com/

which is combined with the respective devices Bluetooth address, finally these are combined in an XOR to give the 128-bit link key for this pairing.

There are some relatively minor flaws in Bluetooth security surrounding the initial key exchange, due to pin-numbers typically being only 4 digits long. If an attacker is able to snoop this exchange then they can easily iterate through all the possible pin numbers and thus decode the final 128 bit key. By spoofing their device address they would then be able communicate directly with either device.

A second flaw is that the Bluetooth device ID could be used to track users, however a similar criticism could be levelled at MAC addresses in Ethernet cards. Many phone users upgrade to new models every few years, thus this tracking information would be of limited use and accuracy.

The Bluetooth solution is totally distributed but at the same time very rigid. For any 2 devices to communicate with each other they must undergo pairing, which requires control of both devices.

Wireless Authentication Service

All the authentication mechanisms currently available on wireless devices are based on either shared pass phrase (e.g. WEP) or a global database (e.g. 802.1x). Neither approach is suitable for creating communities of trusted users as different scenarios require different levels of trust. WEP

Individuals naturally have differing views on who they consider trustworthy, thus traditional global password databases are not useful.

A good system would provide users with credentials (be they a user ID and password or digital certificate) which work with all access points that they have permission to use. This creates an interesting problem, as we require globally unique identifiers with localised access control, something currently not offered by any wireless authentication mechanism.

Thus whilst a global system is needed, whether or not a user is authenticated will depend on the owner of the device they are trying to use.

A Community Trust Service

As a preliminary project in this area I decided to implement a community oriented 802.1x service. Designed for use in the home, it allows people to specify groups of users who can use their access point. Ultimately the goal is to allow any reputable user to use any access point registered in the system, thus enabling people to share connectivity with relatively good security.

One of my main goals for this service was to maintain compatibility with existing client implementations, I decided to use 802.1x as the basic authentication type as it requires no additional hardware at end-users sites beyond the access point and it allows me to place the enhanced functionality I required into a central RADIUS server rather than having to client implementations.

In order to use the service a user must first register a user account, this process is anonymous with the only check being an optional email address verification. The rational behind this being that if you have an email address at a well known corporation this provides a fairly good indicator of your identity. Next the user registers their access points with the system, this includes a public IP address and a shared secret used to encrypt the RADIUS communication. Finally they may specify which communities can use their access points, they can use groups defined by other people or create their own, each access point can permit several different communities.

A community is essentially a group of people who have mutual trust for each other. Communities have administrators who are users with extra privileges allowing them to control who belongs to the community, they have a responsibility for checking the identity of users joining the community and evicting those who have abused the trust.

The service itself works using some simple modifications to the back end RADIUS server, when the server receives an authentication request it checks the source IP address against its database of registered access points, if it finds a match then it examines the user ID requesting to connect, these 2 items are cross referenced against a list of communities to see if both node and user are present in the same community. Following this authentication proceeds as it would with a traditional RADIUS server (i.e. Password/certificate checks), if any stage of the process fails the request is denied.

Results

The service worked quite well, but highlighted several problem areas, discussed below.

- How can users verify that other users are who they claim to be online? If the other user is unknown, short of having a face to face meeting this is hard to achieve
- Even small communities pose management problems, for example when a user applies to join a community which administrator should check their identity etc. This is mainly a problem of communication between administrators
- Too much responsibility is placed on the administrators (ID checking etc), in addition to this users have to accept their standards of verification
- Users potentially have to re-prove their identity for each community they join
- The system is still centralized, thus prone to scalability problems and a directed attack from parties wishing to gain access.
- The trust relationships are only imposed by the server and are not part of the certificate structure, weakening the overall security.

Proposed Improvements

I intend to simplify the service to more closely mimic the trust relationships found in PGP or SPKI. Users will be able to nominate other users whom they trust, and specify transitivity of that trust on a per-user basis. They will also be able to specify a general hop-limit on the maximum distance a user may be from themselves. This should remove the complexity and overhead of the community approach whilst allowing users to have control over who may use their access point.

SPKI Wireless LAN Authentication

An alternative to introducing the concepts of reputation and transitivity into a back end RADIUS server is to use a security protocol with these concepts built in. Both SPKI/SDSI and PGP are good candidates for this.

I choose SPKI primarily due to it's ACL support, whereas PGP only allows users to vouch for each others identity and not whether you consider them to be trustworthy.

Requirements

Any wireless authentication scheme must support mutual authentication, that is to say that the server must also prove its identity to the client as well as the other way around. This is to prevent malicious entities from setting up false authentication servers and harvesting user credentials.

Secondly since the authentication is performed over the wireless medium it must be secure against replay attacks, i.e. recording a successful authentication and then re-sending the packets to gain access and man in the middle attacks where an attacker masquerades as a legitimate access point.

Lastly since this authentication mechanism could potentially run on access points it should have a minimal server-side overhead.

Initial Protocol Design

Based on these requirements I designed a proof of concept authentication protocol using SPKI. This is a 2 stage protocol, in the first stage the client and server verify each others identity, in the second stage the authentication itself is performed. The design of the protocol borrows heavily from TLS[29] and related work on integrating TLS and SPKI[30]. indeed a full integration with TLS is under consideration and would appear feasible based on other work, but not relevant to this proof of concept implementation.

The client begins communications by sending a "ClientHello" message to the server.

The server responds with a "ServerHello" followed by a random challenge string. The client signs this string with it's SPKI private key and forwards the signature (including it's public key) to the server which verifies it's authenticity.

The client then generates its own random challenge which is sent to the server. Now it is the servers turn to sign the string and send the signature to the client, which in turn verifies the signature.

At this point mutual authentication has occurred, both client and server can be sure that the other holds a valid public/private key pair. The public keys exchanged must be the

same as those in the following stages of authentication.

Now the Server sends the client a Tag, this represents the permission token required to authenticate with the server. For the client to authenticate it must present a certificate chain running from the Servers public key to it's own, with each certificate containing the permission tag.

This certificate sequence is sent to the server which verifies it and allows or denies the client as appropriate.

Limitations

The design requires that the access point itself has a certificate and that clients can present a certificate chain terminating with that certificate. This would appear to have drawbacks, firstly when a user changes their access point they will have to re-configure all their devices with the new certificate and secondly if you have multiple access points then you will need to be able to create a certificate chain for each one.

One could address this problem by configuring the access point to authenticate based on some other Principal, this would require modification to the second half of the protocol, with the access point now having to provide both Tag and Principal.

However, this creates problems of it's own, allowing an attacker to setup a rogue access point in order to gain information regarding the certificate structures used. By masquerading as a legitimate access point and requiring the normal Tag and principal pair clients would present their complete certificate chains. Whilst such chains are of little use themselves they will highlight which individuals can delegate the tag required for access, marking them as targets.

To prevent such attacks an additional stage could be introduced requiring the access point to prove it's legitimacy by providing a certificate chain from the owners principal to its own. Although this would require that the client has a local copies of every key used in the chain in order to verify it.

Since it is the clients responsibility to produce a certificate chain and the fact that it must do this before having connectivity means that clients need to have a local copy of

the certificate chain. This is plausible in a restricted environment where you know the owner of an access point in advance, but of little use in a more general setting when connecting to an unknown access points.

Toward a distributed model

To counter the "unknown access point" problem the requirement to produce a certificate chain needs to be moved away from the client to either the access point or an authentication server. SPKI encourages users to make their keyring available online, thus the information required to produce a certificate chain may be scattered across several servers.

The process of discovering and building a certificate chain is a shortest path problem. If one builds a graph of the trust relationships with keys as vertexes and signatures as edges then we can map the possible different routes from one key to another. Of course such a graph would be distributed over many key servers in reality, meaning that the searches required to produce a chain will be very bandwidth intensive.

For example to fully search every key in 5 hop path, with each key having 10 signatures would require examining 5^{10} or 9,765,625 keys. Assuming 256 bytes per query, this would result in ~2.3GB of traffic! Thus more efficient or certainly more distributed querying mechanisms must be found.

A standard Authorisation Tag will have to be used. For example, a tag containing the string "FreeNetworks.org" could signify that one entity may use the access point belonging to another entity.

Recall that SPKI Tags represent permissions which are passed between entities using Authorisation Certificates and that SPKI allows these certificates to be delegated in a binary fashion (i.e. full delegation permissions or not at all). When a tag has been delegated the entity can sign other keys using the same tag and it is as if the original entity signed it. The entity can also choose whether or not to delegate the permission further. By using a common tag and repeated delegation it's possible to build up a web of trust similar to that created by PGP key signings.

Strongly Connected Components

A strongly connected graph is defined as a directed graph that has a path from each and every vertex to every other vertex.

Applied to SPKI it means that a tag has been delegated such that a path exists between all entities. In the PGP web of trust the strongest set is the largest strongly connected set.

This raises an interesting property which is that if two entities believe they are in the strongest set, do they need to find a route between them to establish trust or just prove they are each a member of the strongest set? If both entities are in the strongest set then any path between them must also be in that set, including the shortest path. If either entity is not in the strongest set then this approach fails, otherwise unless we wish to impose trust "hop limits" it is valid.

Perhaps the simplest method of identifying whether an entity is a member of the strongest set is to require them to provide a signature chain to one or more well known members. By “well known” I refer to entities which are well connected, that is they have a high number of signatures, potentially making them easier to locate due to searches having to go less deep. These well known entities must not be static since they will invariably become compromised or redundant at some point, for example maintaining a list of the 10% most connected entities from which 3 are randomly picked should be sufficient.

Such a scheme simplifies our task greatly as not only are our possible destinations limited but due to their highly connected nature should be easier to locate. From my earlier work in wireless routing I believed that some of the MANET protocols could possible be adapted to this task. Many of the MANET protocols rely geographic/short range nature of wireless communications (For example FSR, GPSR, Clubs, etc). Naturally the graphs generated by key signings and tag delegations do not strictly follow this pattern, although geographic bunching may be present I do not expect it to be a fundamental feature.

Another consideration is that the protocol must support asymmetric links, in MANET networks this refers to a link in which traffic passes in one direction only, in this context it refers to when one entity has signed another's key but there is no reciprocal signature. Few MANET protocols have this capability.

Further analysis on the suitability of various routing protocols is still required, however

a protocol which can recognise and take advantage of some nodes being better connected than others should be suitable. Of particular interest is a greedy approach, where the best connected neighbouring entity is always queried first. This relies on the most well connected members having a high degree of connectivity with each other.

In addition since the two endpoints can communicate with alternative approaches may be applicable, for example co-operatively negotiating a route, or flooding from both ends at once.

Preliminary Analysis of the PGP Web of Trust

Whilst I am aiming to use SPKI as my authentication scheme the underlying tag delegation has similar properties to PGP public key signatures. Thus to test different approaches for this problem I am using a worldwide keyring obtained from http://www.pgp.net.

For my initial analysis I used a reduced set of approximately 100,000 keys, of this roughly 10,000 were fully connected with each other. For each of these keys I traced the path to the 2 best connected keys, that is the keys with the most inbound and outbound signatures.

Figure 7 shows the average number of signatures a key on that path has in relation to it's distance from the target. It clearly shows the number of signatures increasing as the destination is reached, although this is quite subdued after 4-5 hops.

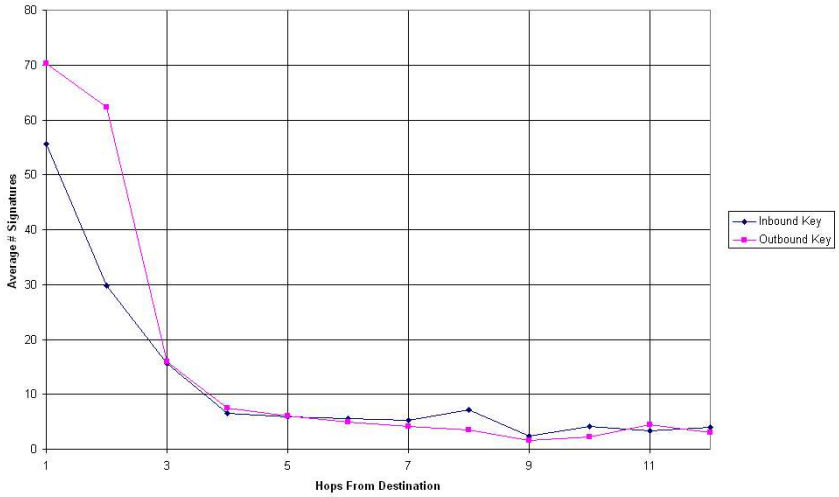

Figure 7: Signatures versus distance from target key

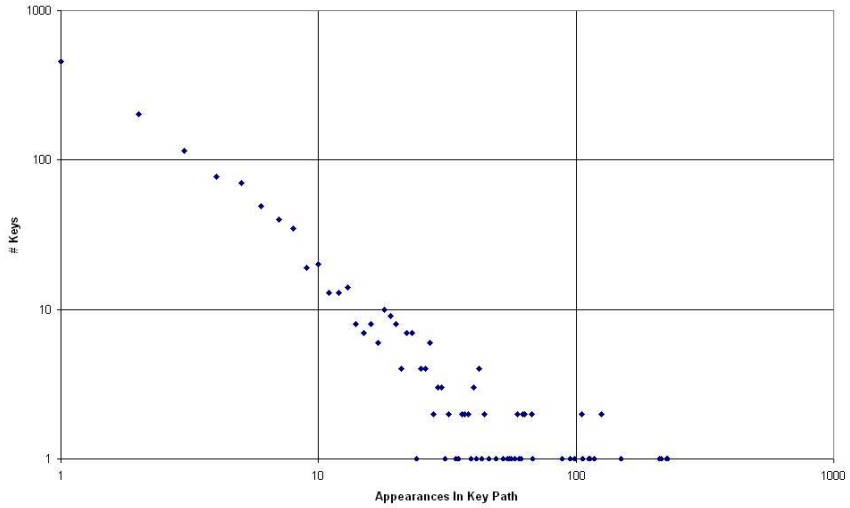

Figure 8: Total number of appearances made
by keys in all paths

Figure 8 shows how many times keys were used. This is startlingly logarithmic and shows that very few keys were re-used in different paths, we can conclude that the key-set is both dense and well connected thus the optimal routes are often unique.

Resilience to Attack

One potential problem with the strongest set is that as it's size increases it becomes more likely that an attacker can gain entry to the set. Not only is it more likely that somebody will be lax with regards to whom they delegate a permission to, but the sheer size of the set will make it a more appealing target and thus subject to more attempts of infiltration. In the worse case the set will cease to be trustworthy in the eyes of its members.

The K of N aspect of SPKI was designed to deal with this, however it is far from a water-tight solution. I believe that in a large set it would only provide marginal improvements in trustworthiness, at the cost of excluding many legitimate users. Further analysis of data is required to verify this.

However, I believe that the strongest set would be self-regulating provided the number of individuals able to delegate the permission is kept to a minimum. If an intruder is discovered then the people responsible for approving him can be petitioned to retract that delegation or face having their delegation permission revoked in turn. The ability

to revoke certificates is key to this and only SPKI provides a robust means in which to do this.

Conclusion and Future Work

Despite my current research being in a vastly different field to the one I started in many of the problems are similar. Establishing trust in large communities is a complex problem, I have shown that the structure of such communities bears some resemblance to that of ad-hoc wireless networks. Consequently the routing protocols and techniques devised in this field may be able to yield results superior to the shortest-path algorithms traditionally used to establish a chain of trust.

Further research is also required, particularly in the area of graph theory and social networks. It has been shown that many distributed networks display similar properties concerning the distribution of links, it would be interesting to see if the PGP web of trust displays similar properties.

I intend to use the full world-wide PGP keyring, containing over 2 million keys, to evaluate the effectiveness of different shortest path strategies and their scalability in a distributed environment. I anticipate that the main evaluation criteria will be time taken to establish a path, computational overhead and false-negatives (when a valid path is not found but one exists).

Bibliography

1: Telecommunications and Information Exchange between Systems -- Local and Metropolitan Area Network -- Specific Requirements -- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Institute of Electrical and Electronics Engineers, 1999, http://standards.ieee.org/getieee802/802.11.html

2: Supplement to 802.11-1999,Wireless LAN MAC and PHY specifications: Higher speed Physical Layer (PHY) extension in the 2.4 GHz band, Institute of Electrical and Electronics Engineers, 1999,

3: Bluetooth Core Specification v1.2, Bluetooth SIG, 2003, https://www.bluetooth.org/spec/

4: Shepard, Timothy J., A Channel Access Scheme for Large Dense Packet Radio Networks, , Volume 26, Pages 219-230, 1996

5: Media Access Control (MAC) Bridges, Institute of Electrical and Electronics Engineers, 1998, http://standards.ieee.org/getieee802/802.1.html

6: Bellman, R, On a routing problem, Quarterly of Applied Mathematics, Volume 16, Pages 87-90, 1958

7: Ford, L; Fulkerson, D, Flows In Networks, Princeton University Press, Princeton NJ, 1962

8: RFC1058: Routing Information Protocol, IETF, 1988, http://www.ietf.org/rfc/rfc1058.txt

9: RFC1583: OSPF Version 2, IETF, 1994, http://www.ietf.org/rfc/rfc1583.txt

10: Dijkstra, E, A note on two problems in connection with graphs, Nuerische Mathematik, Volume 1, Pages 269-271, 1959

11: Weiss, M, Data Structures and Algorithm Analysis in Java, 1999

12: Routing and Addressing Problems in Large MEtropolitan-scale Internetworks, , 1987,

13: Karp, B; Kung, H, GPSR: Greedy Perimeter Stateless Routing for Wireless Networks, 2000

14: T. Imielinski and J. Navas, GeoCast - Geographic Addressing and Routing, 1997

15: An IPv6 Provider-Independent Global Unicast Address Format, , 2003,

16: Nagpal, Radhika & Coore, Daniel, An Algorithm For Group Formation In An Amorphous Computer, 1998

17: C. Chiang and H. Wu and W. Liu and M. Gerla, Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel, 1997

18: , An Algorithm For Group Formation In An Amorphous Computer, 1998

19: Internet X.509 Public Key Infrastructure Certificate and CRL Profile, Internet Engineering Task Force, 1999, http://www.ietf.org/rfc/rfc2459.txt

20: Zimmermann, P, The Official PGP User's Guide, 1995

21: SPKI Requirements, , 1999, http://www.ietf.org/rfc/rfc2692.txt

22: SPKI Certificate Theory, , 1999, http://www.ietf.org/rfc/rfc2693.txt

23: Riverst, R; Lampson, B, SDSI - A Simple Distributed Security Infrastructure, 1996

24: The KeyNote Trust-Management System Version 2, , 1999, http://ietf.org/rfc/rfc2704.txt

25: M, Blaze; J, Feigenbaum; J, Lacy, Decentralized Trust Managment, 1996

26: Abdul-Rahman, A; Hailes, S, Supporting Trust in Virtual Communities, 2000
27: O'Hara, K, Trust - From Socrates To Spin, 2004
28: Borisov, N; Goldberg, I, Wagner, D, Intercepting Mobile Communications: The Insecurity of 802.11, 2001
29: PPP Extensible Authentication Protocol (EAP), , , http://www.ietf.org/rfc/rfc2284.txt
30: SPKI Certificate Integration with Transport Layer Security (TLS) for Client Authentication and Authorization, , 2001, http://www.tcs.tifr.res.in/~vishwas/pki/SPKI/draft-madhu-tls-spki-00.txt