An Adaptable Node Architecture for Future Wireless Networks

D. O'Mahony, L. E. Doyle

Introduction

Technological advances have made possible the production of low-cost computing devices equipped with wireless digital communications. Considerable research is taking place at present both in the academic and commercial research communities and also in the marketplace to determine the form of the wireless networks of the future. So many questions are still open. What radio spectrum will be best? What kind of modulation is most appropriate? How will access to the spectrum be regulated? What will be the overall architecture of the system – cellular or ad-hoc? How will nodes be identified? How will information be routed to its destination? What kinds of security measures need to be taken?

It is clear that there will be multiple answers to these questions depending on the intended application. What is appropriate for communication between a swarm of sensors collaborating to process information may be quite different to that used in a future mobile phone replacement. At Trinity College, Dublin we have been focussing on a software and hardware architecture that will allow us to experiment with many different facets of wireless networking from mobile applications through to core radio issues in a component based approach. A key theme in our research is adaptability. The ultimate goal is to produce a general-purpose mobile node capable of running a large range of network applications that can adapt its mode of operation to the radio environment and prevailing wireless network architecture.

In the following sections, we will first outline the core architecture and then go on to show how it has been used to realize our 4th Generation Mobile Node.

The Layer Construct

The use of layered reference models as a means of de-composing complex networked systems has been with us since IBM's Systems Network Architecture (SNA) and the ISO Reference Model for Open Systems Interconnection (OSI) [1]. The layer boundary allowed an abstraction barrier to be constructed around the functionality of the layer and, in the case of OSI, the precise primitives used for inter-layer communication and their associated parameters could be precisely specified. The use of abstract primitives in OSI allowed implementers a choice of what real programming constructs (e.g. subroutines, tasks, processes) to use to implement the layers themselves and also what mechanisms (e.g. sub-routine calls, inter-process messages) area used to communicate between them. The final choice was generally determined by the native facilities available in the operating systems and language chosen for implementation. In logical terms, layers maintained their own state and implicitly executed their own state machines in parallel with all other layers that were present. One of the common problems for implementers

was to map this implicit assumption into an environment supporting a single thread of control.

In the UNIX streams architecture [2], a very simple inter-layer interface was formulated that basically involved passing a 'message-block' between layers. Most of the time, this block contains data to be sent on the link but it could also contain control directives and parameters. Clark [3] argued that the overhead associated with data copying of these kinds of structure may be very large in relation to that used in implementing the control operations of a particular protocol.

In order to avail of the large numbers of commercially written applications, we wanted to choose a commodity computing architecture. Since we were interested in communicating between palm-top and hand-held devices as well as fixed workstations, the logical software environment to use was Microsoft Windows. Different variants of this base operating system ran on devices with all popular form-factors, and a subset of the Win32 application programming interface (API) was available across all platforms.

The Win32 operating environment has a native implementation of threads available on all relevant platforms and this allows the implicit multitasking inherent in logical layered architectures to be naturally represented in code. Thus each layer is represented by (at least) one thread and since the operating system will pre-emptively schedule each thread to receive a fair share of processor resources, the networking programmer need not worry about scheduling issues.

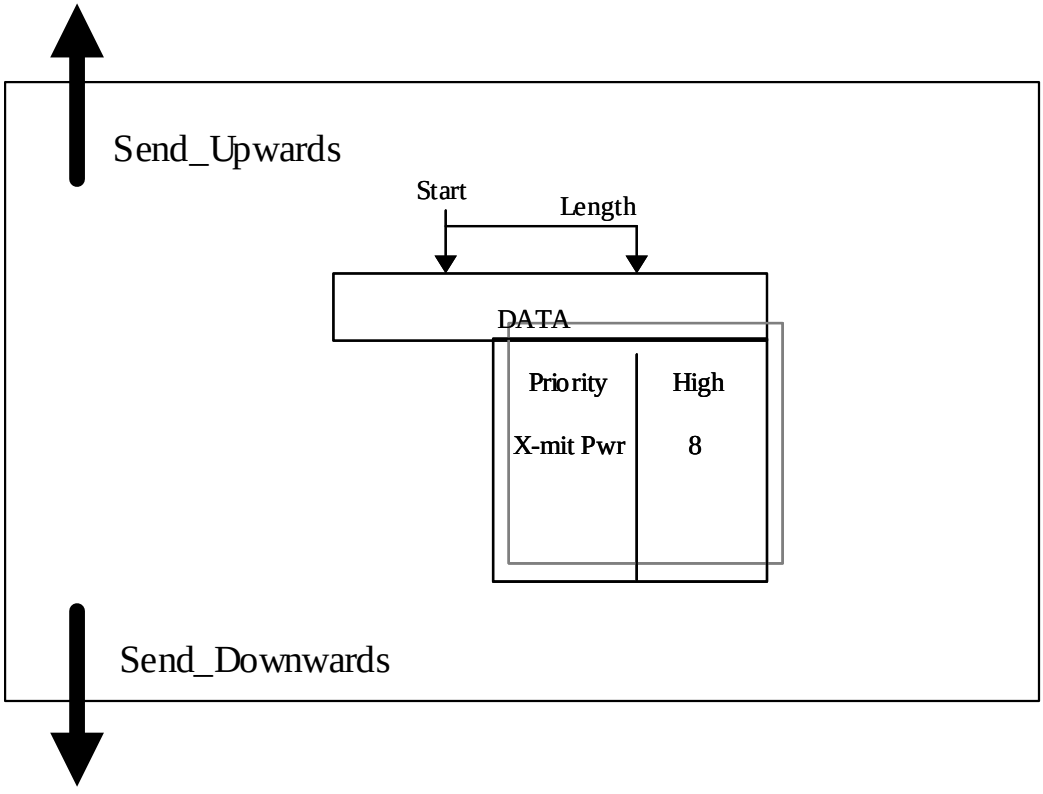

Figure 1: Generic Layer Structure

We have adopted a highly simplified version of the UNIX streams method of inter-layer communication. Our message blocks are dynamically allocated at the source (usually in the application or at the network interface) and a reference to them passed from layer-to-layer without the need for copying. Data primitives tend to grow as they progress downwards through a protocol stack as they acquire header and trailer fields and undergo corresponding shrinkage as they rise up a stack. Our message blocks are sized to deal with the maximum sized data unit with pointers to indicate the beginning and size of the currently active area of the buffer. This allows individual layers to add and strip-off headers and trailers without requiring data movement. Having a single size for the message block simplifies memory allocation and disposal. When a layer has processed a message block, it invokes one of two primitives: *send_upwards* or *send_downwards*, which pass the block to a neighbouring layer. A message block is automatically de-allocated when it 'falls off the end' of a stack.

Data giving information about the circumstances of the data reception or directives on how it should be processed are contained in a 'blackboard' type structure attached to each message block. This is structured as an arbitrary set of attribute value pairs and can be added-to, modified or interrogated by any layer. In this way, it is possible for an application to specify information that may be acted upon by a layer much lower down the stack providing extra facilities for Application Level Framing [3].

In our initial implementations, each layer in the stack had two queues that held message blocks coming in from either above or below. Inter-layer communication was accomplished by moving blocks into these queues and by using the Windows *Event* inter-process communications mechanism to signal their arrival. Performance benchmarks showed that the time taken in queue-manipulation was significant and these were abandoned in favour in favour of using Windows *Messages* that yielded a considerable performance improvement.

In order to preserve modularity, each layer should have minimum coupling with its neighbours. As part of the stack assembly process, a layer is *pushed* onto the top of a (possibly null) existing stack. This process allows a component approach to be taken both to stack development and assembly. If necessary, layers providing different functionality can be added dynamically.

Realizing the 4th Generation Node

The wireless node of the future will be flexible, adaptive, intelligent, capable of functioning within standard and proprietary networks, capable of functioning in structured and ad-hoc environments and suitable for use with multimedia applications. The layered stack architecture is an essential component that enables realisation of this 4th Generation Node.

The dynamic layered structure enables a communication stack to be constructed from a suite of protocols. Different routing protocols, transport protocols, MAC protocols and

Radio interfaces etc. can be inserted and removed as desired. The criteria for choosing the components of the communication stack are varied.

- The communication stack can be created to suit the underlying network or can be adapted to changes in the underlying network. For example if the medium of physical communication changes from infrared to packet radio the MAC layer module that handles the infrared medium can be removed and replaced it with a MAC layer module that handles the radio packet medium.

- The communication stack can be configured to suit the size and topology of the network. For example in the case of ad-hoc networks the most appropriate routing protocol can depend of the number of nodes in the network.

- The communication stack can be configured to suit the underlying communication conditions. For example a group of nodes in a wireless network could decide to use more robust but complex channel encoding scheme under noisy communications conditions at the expense of speed.

- The communication stack can be configured to deal with non-homogenous networks. For example to ensure that all nodes can communicate a configuration based on the capabilities on the most primitive node in the network can be selected.

- The communication stack can be configured to suit the application. For example the chosen transport protocol can be altered for real time communications.

- The communication stack can be configured based on security issues. This can be as simple as choosing a particular kind of encryption for use in the wireless group to as complex as changing the whole communication stack structure if under threat of infiltration.

- The communication stack can be configured based on payment or QoS. Certain layers of the communication stack may be 'cheaper' than others.

Whatever the chosen configuration, the appropriate layers are placed in each node of the network and assembled at runtime to form a complete protocol stack.

Wireless Links

The use of the layered architecture makes it easy to experiment with a variety of different wireless links. Conceptually, a radio layer simply accepts packets from the layer above and transmits these packets to every node within range. We have constructed such layers to support infra-red links, short-range UHF radio and also Bluetooth spread spectrum links.

In the Infra-Red case, we envisaged a scenario where a public space could be equipped with a number of infra-red access points to which mobile nodes could 'dock' whenever they needed to connect with other nodes and services on the fixed network or with other mobile nodes that were in a'docked' state themselves. Transmission is very simple in this scenario as there are typically only two parties involved.

More mobile communications can be achieved using our UHF short-range radios. These make use of an FM radio module on amateur frequencies giving data rates of approximately 40Kbps over distances of 100 metres or so. These radios were chosen to allow full control over what is transmitted and what media access control (MAC) algorithm was adopted.

More recently, we have additionally begun to use radios based on Bluetooth technology that offer rates of up to 1Mbps over short range, but only do so in the context of a picocell constructed using control dialogues. Packets sent on a Bluetooth link are directed at a destination node, and thus this is one of the parameters that should be added to the 'blackboard' associated with each message block that is transmitted. The layer interface hides many of the bluetooth idiosyncrasies from higher levels and allows us to construct real ad-hoc networks consisting of multiple hops over such radio links.

There are multiple possibilities for realising the physical wireless link. To date, in our implementation options are available for choosing IrDa, Bluetooth, and proprietary UHF radios as the frontend hardware for the system. The generic nature of the stack however means that interfaces to other hardware front-ends are possible.

To extend the size and scope of the network system beyond the number of available physical nodes, a means of simulating an unlimited number of extra nodes has been designed. A *datagram layer*, based on the use of sockets, enables the simulation of physical media such as radio broadcasting and IrDA. This layer offers the same interface to upper levels as a wireless link, but communication actually takes place using IP datagrams. Configured into each nodes datagram layer is a list of IP addresses of nodes that are within the simulated radio range. A frame that is send downwards through such a a stack will be transmitted as a datagram to each of the listed nodes just as if it were broadcast on a radio channel.

We have also constructed a 'relay' layer that can interconnect nodes with real radio interfaces together with a population of nodes that care just connected to the fixed network and making use of the simulated radio layer as illustrated in Figure 2. This allows large populations of nodes to be networked together, some of which are real wireless nodes and some that are not This assists greatly with the development of upper level protocols and also serves as an easy means to give wireless nodes access to resources only available in the fixed network.

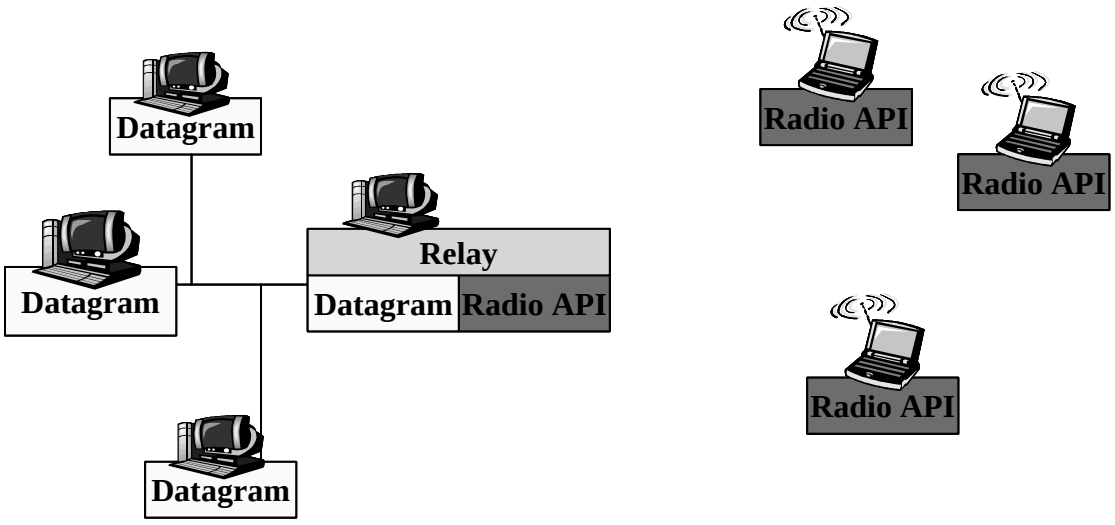

Figure 2: Wireless nodes connected to fixed network via an Access Point. The Access Point is configured with a radio layer, a relay layer and the datagram layer. *The higher layers of the communication stack on each node are not shown*.

The simulation of the radio broadcast environment inherent in our datagram layer does not allow us to simulate mobility adequately. The list of nodes that receive a given transmission correctly is fixed and the reception is either total or non-existent. In order to improve on this, we have begun development of a reality emulator that allows us to experiment with different mobility scenarios.

The reality emulator works by replacing the radio layer in a stack with an emulator layer. The emulator layer in each node connects via a stream socket to a common server. A schematic of this system is shown in Figure 3. The transmissions from all nodes are fed via these streams to the server which then emulates the transmission characteristics of the broadcast medium in real-time.

The server can be fed with a map representing a given area and it then places each emulated node at a point on this map. Mobility scenarios involving node direction and speed can be input to the system and repeatedly played. Based on the relative positioning of nodes, the server's simple model of the radio channel can cause nodes not to hear each other, for transmission to collide and for error rates to rise. Figure 4 shows 8 mobile nodes positioned against a map of Trinity College showing circles indicating their transmission ranges. Since the emulator is configured to emulate our relatively low-speed UHF radios, and the model of the radio channel is quite simplistic, a single server is quite capable of emulating a large number of nodes simultaneously in real-time.

The layers that sit on the reality emulator are unchanged from those version that run on the real radio layers, and the emulator simply makes it much more convenient to exercise them in a wide variety of mobility scenarios.

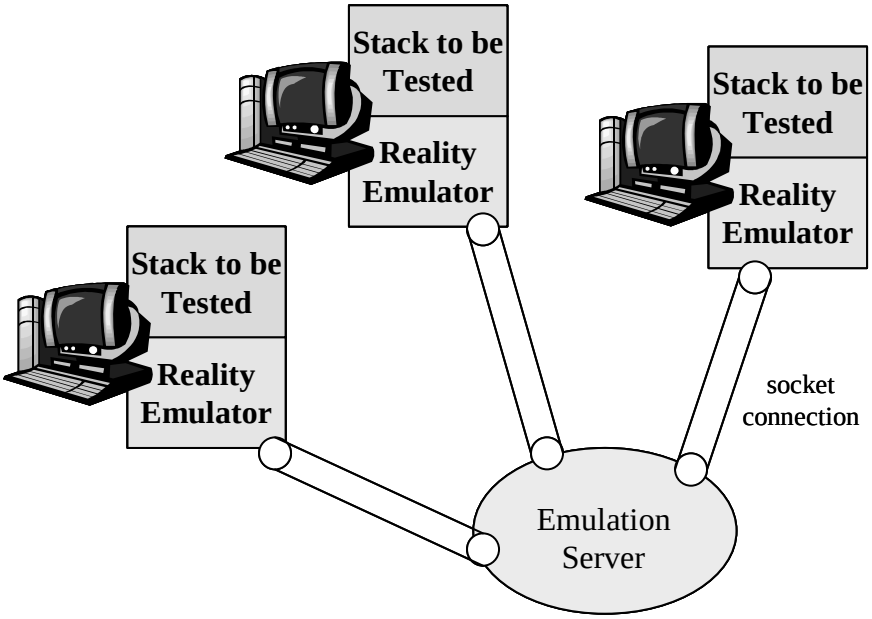

Figure 3: The communication stack of interest sits on top of the reality emulator layer. These layers connect via sockets to the emulation server.

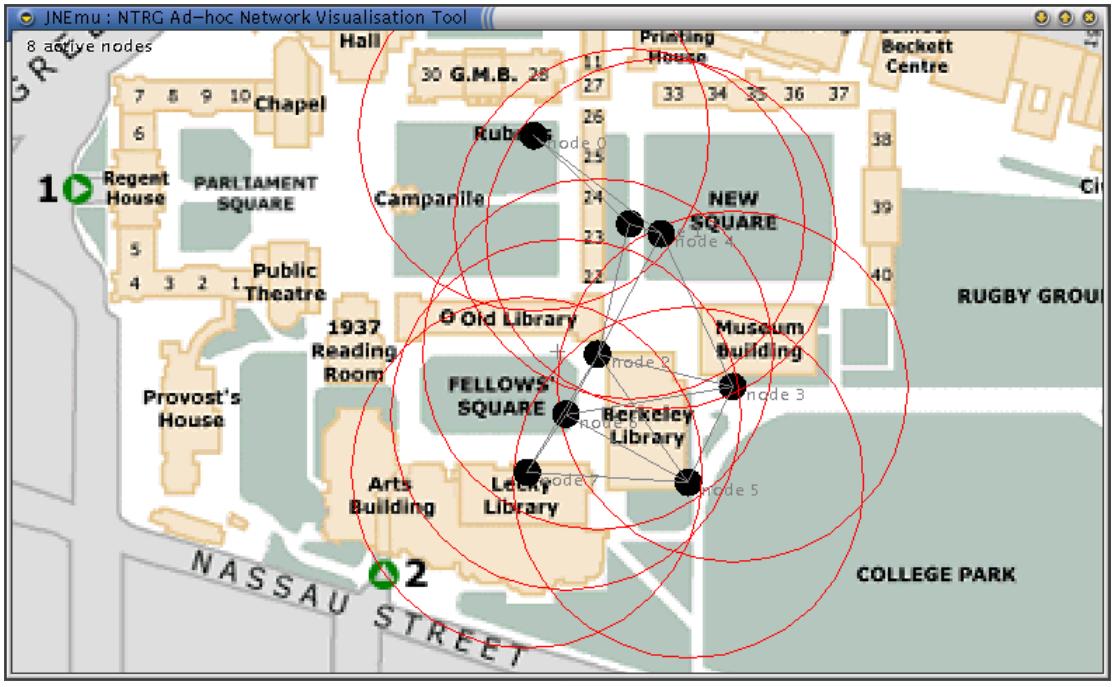

Figure 4: Sample Screen Dump from the 'Reality Emulator' Tool

Software Radio Layers

Typical radio front-ends perform baseband processing and IF and RF communication functions. In the last five years interest in designing re-programmable radios has increased and software defined radio employing adaptable hardware devices such as Field Programmable Gate Arrays (FPGAs) and Digital Signal Processors (DSPs) are being designed to enable multiple radios to be implemented on a single hardware configuration.

To truly realise a flexible 4th generation wireless node we believe that adopting a general purpose processor software radio approach gives us more flexibility when it comes to linking services and communication systems. In the long-term, with increasingly powerful CPUs and with the availability of instruction set extensions such as Intel's MMX and SIMD that much of the software radio functions can be carried out using a general purpose CPU and minimise the need for external radio-related hardware. As the Win32 operating environment has a native implementation of threads available on all relevant platforms the multitasking inherent in implementing the software radio functionality can be easily represented in code.

The radio API layer of the dynamic stack is replaced with a set of software radio layer. This consists of a sequence of 'transformer' layers that will each carry out a single well-defined signal processing operation on the data passing through.

The possibility of being able to specify baseband processing functions as well as the higher layers of the communication stack means that a truly flexible and adaptable wireless node can exist.

Ad-Hoc Networking Protocol Layers

The adaptable node of the future will often be required to operate in an ad-hoc environment. Ad-hoc routing protocols seek to address the fundamental problems of route discovery and data trafficking in mobile networks. As ad-hoc networks are subject to unplanned growth, reduction or fragmentation, any routing algorithms employed must be able to respond to the changing nature of the network's topology. Changes to the network topology manifest themselves in many ways; link cost, link symmetry, route congestion, node mobility, and node density to name a few. These dynamic attributes of an ad hoc network must be accounted for in the design of any successful routing protocol [4].

Under the stewardship of the I.E.T.F.'s Mobile Ad Hoc Network (MANET)) working group, various routing protocols have been proposed and are at different stages of analysis and development [5]. These protocols fall into the two broad categorizations of proactive and reactive protocols. Proactive, table-driven protocols attempt to keep an up-to-date view of the network topology, and in the main are adapted and improved versions of the link-state and distance-vector approaches taken to routing in traditional wired high-bandwidth networks. Reactive protocols such as the Dynamic Source Routing (DSR) [6]

protocol will only seek out a specific route when the source requires it. More recent approaches have sought to leverage the optimal aspects of both reactive and proactive protocols. One example is the Zone Routing Protocol (ZRP) [7], a hybrid protocol that incorporates both proactive and reactive elements. In the ZRP, nodes proactively maintain information about neighbouring nodes up to a certain distance way, the zone radius. If routes to destinations outside this zone are sought, then a reactive protocol is employed to dynamically query nodes lying outside the node's zone.

Standard Ad-hoc routing protocols (reactive, proactive and hybrid) have been implemented as layers of the dynamic communication stack and can be tested on real networks [8], using the simulation facilities and using the Reality Emulator.

The work in this area has shown that it is clear that the creation of a routing protocol that suits all types of environments and operating conditions is not possible as different protocols excel in different scenarios. To this end adaptive routing protocols are being designed.

Our adaptive approach works on two main levels. Firstly our approach involves creating a system that can facilitate the choice of the best protocol (from a suite of available protocols) to suit the constraints of the scenario. Secondly the adaptive approach also extends to a sub-protocol level, i.e. the protocols themselves could have adaptive elements. To be able to deal with this level of adaptability, the system must has the following characteristics:

1. The system must have the capability to select the most appropriate protocol or change from one routing protocol to another based on stringent decision-making rules. The decision making process must be robust and dynamic.

2. Within any given protocol there are a wide number of controlling parameters. The system must have the ability to optimise individual protocols in a reliable fashion. The optimisation can be local or network wide.

3. The decision-making process will must be based must be dynamic. For example when a wide range of information is available to a node, then complex decisions can be taken. When less information is available (for example due to hardware failure, or due to a security alert that makes information from other nodes suspect) decisions must be made using whatever information is available. In the worst-case scenario a base level routing protocol must be used.

4. A mechanism for communicating the resulting decisions (which may also result in a change of the underlying operating parameters of the communication system) to the rest of the network.

Such an *adaptive ad-hoc routing system* can effectively deal with the routing issues and problems arising in a hostile and changing environment. The flexible layered architecture approach is a key component of this system.

Network Applications

One of the earliest application layers that we developed was designed to allow Internet access from the mobile nodes. Since all of our mobile nodes (both laptop and Palmtop) already run conventional Internet browser software, we were able to connect this to our protocol stack using the concept of a web proxy server. The applications layer in the mobile node listens on a local socket for web requests from a suitably configured locally running web browser. The lower layers in the protocol stack are then used to delver packets from this application layer to a corresponding proxy-server layer running on a node with fixed network access. This then issues the web requests to web servers on the Internet and forwards the results back to the mobile node. This allows us to easily delver web access to a population of nodes in a wireless ad-hoc network.

We see point-to-point voice communications as a key application for 4th generation mobile nodes. End-to-end voice transfer coupled with the ability to locate users as they move from node to node on the fixed network, or as their wireless node moves into and out of range of various fixed network access points will be essential if 4th generation systems are to take over the role that mobile phones occupy today.

The basic audio capability is provided in the layered architecture by a general purpose layer that captures audio and sends it down through the protocol stack. Audio packets coming from below are played to the hardware device. When this layer is coupled with an application layer that implements a signalling protocol such as the Session Initiation Protocol (SIP), we have the basis for a mobile telephony system. The issue of handover is not addressed as yet.

Not alone is the system suited to multiple applications, the flexible layered architecture can be exploited to enhance the performance of multimedia communications. For example to enable multimedia applications over ad-hoc networks the issue of error resilient transmission must be addressed. We have designed a new error detection and concealment technique to facilitate error resilient transmission [9]. This technique exploits information from the decoded image data itself and also uses information from the underlying network. The information from the underlying network is access by inserting appropriate layers in the communications stack to pass information of interest to the application layer. When applied to MPEG4, the method can localise errors to an even greater extent than with reversible variable length codes (RVLCs) alone.

The 4th Generation Security Architecture

Wireless networks, and ad-hoc networks in particular, derive part of their appeal from the fact that they enable all sorts of communication that was not previously possible. Such openness brings increased security risks. Where very large number of nodes share the same radio space and are coming into and out of contact with the fixed network, it will become very important to ensure that parties are authenticated before being allowed to communicate with each other and to access shared resources. In cases where the fixed

network is accessible, it will be possible to check back to a central server to verify the good standing of a node, but in a disconnected ad-hoc scenario compromises may have to be made in security to ensure the proper functioning of the ad-hoc cluster of nodes.

Nodes in our 4th generation system will authenticate themselves on the basis of a multi-faceted identity. At a very basic level, if a node can perform a simple friend-or-foe identification on a neighbour, it may be content to route packets through that node. Authentication of a much higher level may be required before they will enter into an application level dialogue involving sensitive information. Nodes can take on multiple identities simultaneously and these identities may also be hierarchical in nature. When authenticating, a node can progressively reveal more and more of their identity. This will start with an initial friend-or-foe check followed by perhaps further information on the persons rank within the organizational hierarchy.

Based on the identities revealed, multiple nodes will bind together to form groups. Once again, a node may be a member of several groups simultaneously. An example to illustrate this may involve multiple emergency services arriving at the scene of an accident. All nodes will join a basic group allowing them to route traffic for the others maximizing connectivity. Paramedic and fire-fighting personnel will each form sub-groups that use this connectivity to have end-to-end dialogues with each other without the possibility of crossover between the dialogues. One can also imagine that other groups could be present in the same space without the majority of nodes even being aware of their existence.

In the internet multicast community, much work has been done to facilitate the formation of groups of users that could be formed dynamically and allows users to join and leave these groups while preserving perfect forward and backward secrecy. An example of such system is the Versakey [10] system developed at the ETH in Zurich, Switzerland. These systems tend to rely on the fact that a single node takes on the role of overall controller of the system. In a highly dynamic wireless environment, it will be necessary to form groups in circumstances where this is not possible. It will also be necessary to create new groups on-the-fly and to pass the group leadership role from one node to another.

Our 4th generation security architecture also addressed the issue of resource usage. The co-operative environment of ad-hoc networking relies on a nodes willingness to relay traffic for its neighbours. This activity will deplete a nodes battery and also potentially reduce the bandwidth available for its own transmission. One way to address is for nodes to agree only to relay traffic for nodes that are part of an identified group and then to tightly restrict the membership of that group. A more flexible way to address resource usage is to find a means by which nodes can ‘pay’ each other in real time for any resources consumed.

We have taken the concept of a micropayment, which is a lightweight cryptographic technique for making repeated payments of very small amounts to work with multiple

parties [11]. This allows a node to communicate end-to-end using the services of many intermediate nodes. A single stream of payment tokens is injected at the source interspersed with normal data. Conceptually, this represents a single payment that passes through all nodes along the path. Each of the nodes then 'breaks-off' part of the payment commensurate with its contribution to maintaining the end-to-end link. We can use this to compensate mobile nodes for the use of their resources and also to reward organizations that provide wireless access points to the fixed network. This technology essentially enables anyone who has access to the fixed network to in effect become a network operator. The fact that nodes do not have to sign-up in advance with a single organization should allow for increased competition in networks of the future.

Reliable authentication, flexible and secure group formation capabilities and a secure means of payment for resource usage are key elements of the 4th Generation security architecture.

Summary

The ultimate goal of our work is to produce a general-purpose mobile node capable of running a large range of network applications that can adapt its mode of operation to the radio environment and prevailing wireless network architecture. To achieve this goal we have created a software and hardware architecture that allows us to experiment with many different facets of wireless networking. We have experimented with multiple wireless links, MAC protocols and ad-hoc routing protocols. We explored security issues and designed multimedia applications for the system. The resulting node architecture is one that embraces flexibility and adaptability at all levels from the hardware frontend (through software radio) to the application layer and is also an architecture that facilitates multiple levels of security.

References

[1] ISO, Information Processing Systems – Open Systems Interconnection – basic Reference Model, 1984, ISO-7498

[2] D. M. Ritchie, "A stream input-output system," Bell System Technical Journal, vol. 63, no. 8, pp. 1897-1910, 1984.

[3] Clark, David D. and D.L. Tennenhouse. "*Architectural Considerations for a New Generation of Protocols*." Proceedings of the SIGCOMM Symposium on Communication Architectures and Protocols (September 1990). Computer Communication Review, vol.20, no.4, Sept. 1990.

[4] Z.J. Haas and S. Tabrizi, "On Some Challenges and Design Choices in Ad-Hoc Communications," *IEEE MILCOM'98*, Bedford, MA, October 18-21, 1998

[5] Elizabeth Royer and C-K Toh " A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks ," *IEEE Personal Communications Magazine*, April 1999, pp. 46-55.

[6] Charles E. Perkins and Elizabeth M. Royer. “Ad-Hoc On-Demand Distance-Vector Routing.” *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, February 1999, pp. 90-100.

[7] Z.J. Haas, "The Routing Algorithm for the Reconfigurable Wireless Networks," *ICUPC'97*, San Diego, CA, October 12-16, 1997.

[8] T. Forde and L.E. Doyle and D. O'Mahony, "An Evaluation System for Wireless Ad-Hoc Network Protocols," Irish Signals and Systems Conference, June 2000, pp 219-224.

[9] L Doyle, A. Kokaram, D. O'Mahony, "Error-resilience in Multimedia Applications over Ad-hoc Networks", To be published ICASSP May 2001, Salt Lake City, Utah.

[10] Marcel Waldvogel, Germano Caronni, Dan Sun, Nathalie Weiler, Bernhard Plattner: The VersaKey Framework: Versatile Group Key Management, IEEE Journal on Selected Areas in Communications, Special Issue on Middleware, 17(8), August 1999

[11] Peirce, M. & O'Mahony, D., Flexible Real-Time Payment Methods for Mobile Communications, IEEE Personal Communications, Vol. 6, No. 6, December 1999, pp 44-55