=

Media Transmission over Coupled Wired/Wireless Networks using Application Level Active IPv6 Networks

Subtitle

Peter T. Kirstein, K.Carlberg, K. Hasler and P. O'Hanlon
University College London

Abstract: A general architecture for Edge services in the IPv6 environment has been developed in several active networks projects. In a series of multimedia, multicast, conferencing projects, it became clear that an edge media adaptation service is needed if the participating networks are heterogeneous. This need became more acute with wireless access to the Internet. This paper describes both the general architecture, and the functionality of the components, to achieve active edge services. It considers also how the IPv6 functionality assists in setting up a generic service, which may be of interest to wireless operators.

Key words: IPv6, multimedia conferencing, active networks, policy-based services, wireless access, edge services.

1. INTRODUCTION

Wireless networks have many interesting properties. For the sake of this paper, some particular ones are the following:

- The capacity of the link varies dynamically as a function of the distance from the Access Point (AP) to the wired infrastructure and other factors;
- The speed of many wireless links will stay modest, less than a few 100 Kbps, over 3rd generation systems – e.g. UMTS. Others, like Wireless LANs, may have systems speeds exceeding 10s of Mbps.

- A single terminal may have wireless connectivity to several APs with one technology – but the intelligence in the terminal and the APs will allow only one to be active at a time;
- There may be several wireless technologies overlapping, so that a terminal may be attached to several at the same time (e.g. Wireless LAN and UMTS);
- If several access points are accessible at the same time, policies can be defined specifying the best one to use – with or without signalling elsewhere which is being used.
- The wireless WAN services Operators of (e.g. UMTS) will not put in special code to deal with specific applications unless a good revenue stream is assured.
- The UMTS operators have stated that they will embrace IPv6 for 3rd generation services.

These properties of wireless services provide application developers a number of challenges. The operators need new services, but are not really prepared to take commercial risks to provide for them. Operators would like a share of current LAN applications, but do not know what Quality of Service (QoS) they will be able to offer such applications.

These challenges are a clear opportunity for Active Networks. Here we do not mean networks where each packet carries a programme, as was proposed in the original DARPA ideas [tenn97], but rather where active adaptation is done at the boundary between the wireless and the tethered worlds – i.e., active gateways or servers providing new edge services.

In this paper we describe activities which were done originally in a number of different projects, but which are coming together to provide potential multimedia services. The services are IPv6 enabled, where the components and management infrastructure are policy controlled. Provided that a basic skeleton is running in a server and is well-connected to the wireless AP, the software to service the application need be loaded onto the server only when revenue-paying customers wish to invoke the service. The main example that we present will be multimedia conferencing, with media adaptation at the boundary is on demand; the technique, however, has a much broader applicability.

In Section 2, we provide the background to the project; this comes from three different sets of projects with an IPv6 flavour: multimedia conferencing, active networks, and wireless access. In these projects an active service for edge networks has been developed, which is being extended. This service may be invoked dynamically, and so may be very suitable for next generation wireless networks, where the killer applications

are still being sought. The concept of this service is developed in Section 3. Finally some conclusions are presented in Section 4.

2. BACKGROUND

We at University College London have been part of three streams of projects – all heavily reliant on IPv6:

- Multimedia Deployment – e.g. MECCANO [kirst00], [meccano]
- Application Level Active Networks - e.g. ANDROID [ANDROID]
- Wireless Internet Access – e.g. 6WINIT [6winit]

Our activities in these three projects have come together to provide an interesting concept for an experimental service in which an active server assists the provision of multicast multimedia. This includes seamless operation over wireless networks.

2.1 Multimedia Projects

In MECCANO we piloted the deployment of the Mbone multimedia tools. For multimedia deployment over heterogeneous networks with widely varying bandwidths, it is often necessary to provide media adaptation at the boundary between the networks. In MECCANO we developed early versions of a UCL Transcoding Gateway (UTG), which provided media and protocol mediation at that boundary. The UTG could do media adaptation like video packet filtering, media transcoding (like ensuring that an audio coder robust to wireless errors would be used on the wireless network), and unicast/multicast protocol conversion. The parameters of the UTG were fixed partly by the system with knowledge of the networks either side, and partly by the user.

We made first attempts in MECCANO, to ensure that the Mbone tools were IPv6-enabled. The operating system support for IPv6 has improved recently; we have tracked the changes, and the Mbone tools VIC for video, RAT for audio and NTE for shared whiteboard are now available for a broad range of operating systems [mbone]

2.2 Application Level Active Networks

Application level active network (ALAN) [Fry99] provides an environment in which developers can engineer applications through the network by utilising platforms on which 3rd party software can be

dynamically loaded and run [Mars99]. The ALAN system consists of peer-to-peer or client and server applications that are located in the existing Internet. These communicate through Application Servers (ASs) that are located in strategic points between them. An AS provides an Execution Environment for Proxylets (EEP). Proxylets are downloaded to the AS from a code server, and are executed on behalf of the users. Those applications provide functionalities that enhance the level of service or introduce new services to the final user. End-to-end active services are provided by one or more ASs executing one or more proxylets. Messaging is done in XML [XML01a] and [XML01b]] and is carried over HTTP [Mars01]. These concepts are being developed further in the IST project ANDROID. Here we have provided a two-level system of Proxylets, which have the code loaded from code servers, and policies (written in XML) loaded from a Policy Server. The general run-time environment envisaged is shown in Fig. 1. The particular implementation of ALAN software being used is the FunnelWeb software from ITS [fry].

The Active Application (AA) proxylets are fetched dynamically from the Active Applications Depository (AAD), and the Policy Modules (PM) from the Policy Depository (PD) of Fig. 1. FunnelWeb ensures that these modules can be loaded dynamically and bound together. Of course one Active Service may require several AA proxylets and PMs. Moreover, the AA Proxylet may be generic, while the policy module is related to the specific user request – or organisation request; the latter case might apply, for example, if the application is the set up of a dynamic Virtual Private Network (VPN).

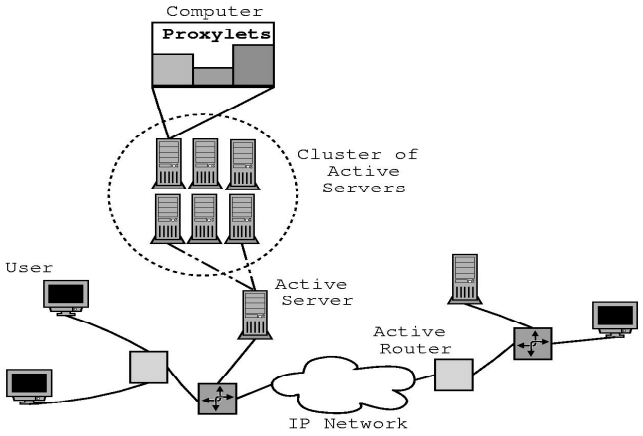

Figure 1. Schematic of Active Network Environment

Figure 1 shows a cluster of active servers, though they may be single systems. The active server can be considered as an end system with a full protocol stack. It also provides an execution environment capable of running user-provided processes that are unrestricted above the transport layer [Mars99]. Multiple Execution Environments for Proxylets (EEPs) are allowed to run on each active server. Each EEP is allowed to run one or more proxylets. The EEP is a Java Virtual Machine (JVM) – FunnelWeb – with rudimentary control primitives (e.g., stop, start, control, load). Each proxylet runs on its own JVM. Active server security and resources (consumed by the proxylets) need to be managed locally. Proxylet thread resource consumption is managed by application providers or the users and is out of the scope of these projects.

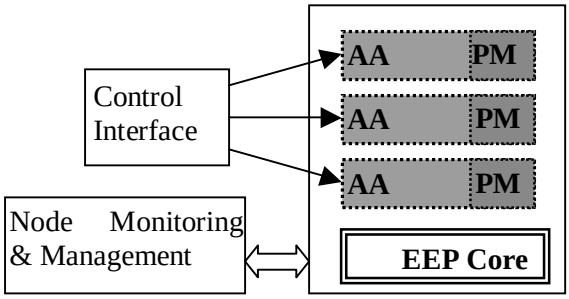

Figure 2. AS Architecture

An example of the structure of an AA Server is shown in Fig. 2. The AAs are written in pure JAVA, and run over the FunnelWeb Execution Environment (EE) – with each instance of FunnelWeb running in a separate JAVA Virtual Machine (JVM). A node monitoring/management system can control the ASs, and each AA has a well-defined control interface. In ANDROID we have ensured that the whole environment of EEP, Node, AA and all servers operate in the IPv6 environments. Because of the heavy use of JAVA, this required the JDK1.4 release from Sun which was properly IPv6 enabled. The full ANDROID system is described in [android].

2.3 The TAG

As part of the ANDROID project, the MECCANO UTG was modified into a Transcoding Active Gateway (TAG), which could operate as a normal Proxylet in the ANDROID system. Now the parameters of the TAG could be changed dynamically, as a function of the networks either end, the

performance of the networks, and the short term behaviour of specific media applications like audio or video streams. While some of the TAG functionality is coded in, others are loaded dynamically as other proxylets, or modified as a result of XML-expressed policies. Hence, the media stream adaptation can be optimised by the TAG as a function of traffic and instantaneous user perception of performance. A schematic of the TAG is shown in Fig. 3.

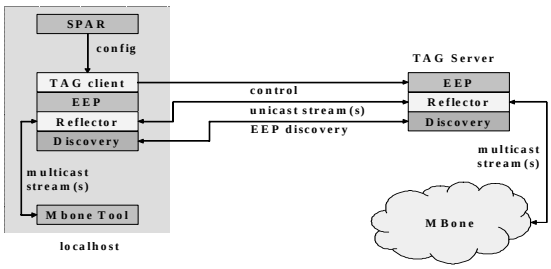

Figure 3. Schematic of the TAG

The TAG is designed for two purposes: acting as multicast-unicast reflector for VPNs and as a media adaptation device. The key points of the current TAG design are:

- automatic configuration of the Relay components using the Secure Conference Store [kirst00];
- use of Active Networking for locating and positioning reflection and media adaptation points;
- re-multicasting of reflected media streams on the client (for VPN);
- multicast-unicast conversion for media streams (in wireless nets);
- modular approach of the TAG components, implemented as Java proxylets, dynamically loaded as required and as a function of policy.

Each media reflector is a separate *proxylet*, written in pure Java. The Proxylets contain additional information for execution on the EEP. The added metadata policy file could limit control of the proxylet to a range of hosts. The Reflector proxylets processes both the incoming unicast stream originating from the opposite end point and the outgoing multicast streams originating from the multicast group/tools. In the case of video and audio, reflectors also process the additional RTCP stream. Simple rate-limiting cuts off forwarding after a pre-set rate limit (in packets per second) has been reached. The rate can be altered from the client using the RMI connection to the server, which then in turn passes the new rate to the proxylet.

For the VPN application, TAG has can connect a client into a conference VPN and subsequently provide access to the VPN media streams. Using this mode, a client will automatically request to join the VPN once local multicast activity is detected – this latter point making the request transparent to the user. The join request takes the form of a registration of a local EEP to a Reflector Manager (RM). The registration triggers the RM to notify each node within the VPN to forward media streams back to the registered EEP. In addition the RM can be used to convey rate and access information to each remote node. Several extensions are still under development:

- Media conversion (transcoding) and rate control
- Quality of service improvements to the reflection
- Improvements to the discovery and location of Active Servers/EEPs
- EEP access control to prevent unauthorised user access.
- Services offered by an EEP
- Support for Foreign Agent functions
- Secure communications and authentication of client

2.4 Policy Syntax

The overall structure of a policy is shown in Figure 4. The general approach is consistent with [damn]. A policy is written to be interpreted by a subject, which is then expected to perform specified actions on targets, possibly dependent on some conditions.

The top-level policy specification consists of six elements. The creator element allows the origin of a policy to be established. This policy specification is intended to be applicable in an environment where there are multiple points of control. Components should be able to accept control from users with different privileges. The administrator of the TAG, for example, will have ultimate control of its configuration, including the permitted extent of control by other users. End users, or the operators, may be allowed to control the way the router behaves for their own traffic.

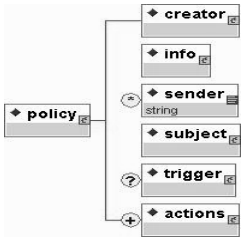

Figure 4. Schematic of a Policy

The info element contains most of the information about the policy that is not directly related to the policy rules. It includes a globally unique identifier for the policy and an indication of the modality of the policy (authorisation – what the subject may and may not do, obligation – what the subject must do, and should not do). The intention is that the modality should not be mixed within a single policy. The general policy handling components of the management system, which have no detailed knowledge of the specific rules contained in the policy, may then take advantage of this classification. It may also contain a textual description of the policy. The creation time, start time and expiry time of the policy can also be specified. Finally, a list of policies replaced by the current policy may be specified. Policies are considered to be immutable so different versions are not possible.

The sender element identifies the forwarding path the policy has taken. This information could be used as a check that the policy has followed an expected route, or to establish that the policy has already been received by other parts of the system. This element may be modified between creation and receipt of the policy and would therefore not be subject to digital signing.

The subject element identifies those entities in the system that are expected to respond to a policy. Identification of these entities is done by role. This is important so that a policy can refer to entities that are not present or not known at the time of creation.

The optional trigger element relates an event (via its unique event type identifier) to the policies that are supposed to handle it. When an event is detected, relevant policies must be activated. It is assumed that a policy is triggered by a single event. Correlation of events (aggregation, sequences, threshold rates etc.) is assumed to result in the generation of a single event, which then triggers an appropriate policy. A system that provides this functionality is described in [Nata01]. If a trigger element is not present, the policy is assumed to be activated as soon as received. This approach can be used in systems, which are not based on events. Triggerless policies can be used, for example, to effect immediate configuration changes that control subsequent behaviour.

Every policy includes one or more actions elements. These specify the behaviour that should result from the triggering of the policy. Each actions element contains an optional condition expression and a set of strings specifying actions to be taken on particular target components. These actions can use the open content features of XML schema to allow tags appropriate to any particular system to be included in a policy instance. All that is required is for the creator of the policy and the system on which the action is to be taken to agree on the syntax and semantics of the action.

2.5 Events

The structure of an event is shown in Figure 5. Its purpose is to provide sufficient information to allow generic components to be used in the distribution of events to all interested components but also to allow any additional information to be included to support specific circumstances. An event following the syntax specified here may be generated either directly or by an XML-aware component (such as ANDROID security and resource managers) or by a special monitoring component that obtains information using some other mechanism. Full details are given in

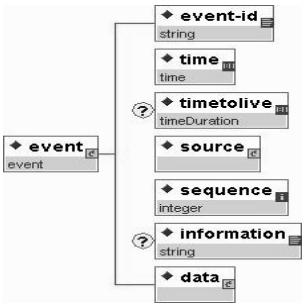

Figure 5. Event Schematic

The top-level event specification consists of seven elements. Each event type has a unique event-id, that may be used to trigger appropriate policies. The time element identifies when the event occurred while the optional time-to-live element specifies for how long the event is relevant. The source element identifies where the event originated. The sequence element is an integer, incremented with each event from a source. The optional information element is a text string intended to be read by people rather than processed automatically. The data element has an open content model and allows any well-formed XML to be included. Here any specific information about the event can be included, using whatever structure is most appropriate. It is only necessary that the event producer and the interested recipients share knowledge of the contents of this element.

The main customisation of the operation of the Proxylets, and of the control of their performance, comes from choice of suitable policies and event behaviour.

2.6 Wireless Access Networks

In 6WINIT, we are concerned also with the behaviour of media streams over wired/wireless boundaries. We make heavy use of Mobile IP, and are becoming increasingly concerned with situations where a specific terminal, e.g. a PDA, could have access either via WLAN or GPRS (and later UMTS). We would like to vary the media adaptation at the boundary depending on which of the networks was currently optimal for the user, and its current performance. Thus it is possible for the Care of Address server to control the media adaptation, and direct the streams to different wireless attachment points. We are now integrating the TAG of ANDROID with the mobile server functions of 6WINIT as part of the 6WINIT project.

3. ACTIVE NETWORKS IN WIRELESS SYSTEMS

3.1 Dynamic Edge Services

The sort of integrated service proposed here, based on active servers, could be of considerable interest to the wireless operators. This application, in the FunnelWeb environment, deals with many of the bullet points made in Section 1. Moreover, we can use specific bits in the IPv6 Header to indicate that this is a data stream destined for the active server; if these bits are not set, then the data packets are not active, and will pass through unchanged. If they are set, they are diverted to the active server, which carries out the requisite TAG functions.

One of the problems of the wireless operators is that they do not know which are the killer applications. For this reason they are reluctant to put up much resource for a new service, until they are convinced that it will be popular and that they will recover revenue. Under the Active network regime, it is only necessary for there to be an EEP-enabled servers well connected (on the tethered side) to any wireless Access Point (AP). Only when users request a service, will it be loaded dynamically into the server. Since this loading is as a direct consequence of a user action, even if the invocation is automatic, it is possible for the operators to charge for the Value Added Service. The ASs could deal with a significant number of different services, which could all be charged on demand.

A number of components are needed in a fully fledged system, and a number of optimisation decisions have to be made. For the application envisaged above, we would need to load also a charging Proxylet, a Resource Discovery neighbour search Proxylet – to discover the location of

the most suitable Code and Policy servers. For some variants, it will be necessary also to have a neighbour discovery Proxylet, which discovers the location of other ASs, and a Resource Monitoring Proxylet, to monitor the load on the Server. It would be quite possible for a further Active Server to be brought up dynamically, when the load on a server exceeds some (policy-based) value.

3.2 Value Added Services for Edge Networks

Most IP services are end-end, which makes it very difficult to address intermediate nodes inside the network. The availability of IPv6, with its extended packet format, large address space and fluid standardisation procedures, has an important role here. It is universally accepted that the bottom 64 bits of the address should denote a global address; the function of the other 64 bits is still under discussion in the IETF and elsewhere; moreover, there are other parts of the IPv6 packet format proposed for special functions. We propose reserving a few bits to indicate that the packet is part of an active service, which would be well served by assistance of an active server near a wireless boundary. The routing functions of IPv6 routers are also still under development. It would seem trivial to have edge routers look at the relevant part of the IPv6 address, and route packets that are part of an active service to the relevant server. The server will then process the data, modify the relevant “active service” bits in the address, and route the packet back to the router. Because of the setting of the services bits, the returned packet(s) will then be re-routed to the normal end host. It will require the usual IETF Working Group discussion to fix the nature of the functions required on the packets requiring active services. For the time being we will make some ad hoc decisions and possibly use the experimental diff-serv code space that already exist, but we will discuss these in the IETF.

The active server in this scenario has two aspects. One is packet processing at the edge of the network already mentioned; another is that if the relevant active application is not yet mounted, it is possible to pull in the relevant code and policies from a code and policy server. This initiation may not be trivial; the code may all exist on a preset server; it may also be necessary to run a Resource Location proxylet to locate the relevant server(s). The policies to be used may be common to all who run this application; they may, however, be specialised to the interests of a particular operator.

3.3 Experimental Plans

The FunnelWeb infrastructure, the Code and Policy Servers, Network Management and Security are all being pursued actively in the ANDROID project. These are largely completed to the extent of providing a usable experimental system, and will be demonstrated at the IST2002 Conference in Copenhagen in November 2002. In that project there a number of other issues are being studied: Resource Management, Mobility, Security Management and Policy Distribution. Moreover the application focus in that project is dynamic VPNs. The project finishes with the IST2002 demonstration, but most of the requisite infrastructure is already in place.

The extension of the TAG to the wireless domain is being effected in the 6WINIT project. This will also be demonstrating at IST2002, the sort of applications considered above, in the context of the wireless networks available there: GPRS and WaveLan. Under that project, there is a planned extension to UMTS to be implemented in its last three months. We expect to show mobile applications based on hand-held wireless devices roaming between the three networks technologies using the active media streaming based on the TAG at the final 6WINIT review at the end of January 2002.

6NET a new IPv6 deployment project lasts from January 2002 – December 2004. 6NET is providing IPv6 applications deployment experience, Interconnection of wired and wireless networks is one of the tasks in that project. It is our intention to bring the completed TAG into the 6NET project. Here we plan a deployment using proper commercial routers (rather than the experimental FreeBSD Kame ones used hitherto). In this environment, the active edge services will be subjected to a more searching scrutiny of whether it is reasonable to put it into an IPv6 service environment.

Many of the modules mentioned above are still under development, but will be complete by the time of the deadline for submission of the full paper, and will be included. The many other components – like the resource manager, server replication management, and security management are still under development, but will be completed and integrated with the UCL-CS code by the time of the actual conference – and will be reported orally.

4. CONCLUSIONS

In this paper we have shown how the Application Level Active Networking can be used to provide dynamic self-adapting services, which may be very useful for the deployment of multimedia over wireless

networks. The work combined the results from three projects, originally unrelated, to produce a potentially useful outcome. While the original projects were IPv6 in nature, they did not really make much use of the properties of IPv6. This development does indeed use the potential offered by the IPv6 packet format. Whether we can really persuade wireless operators to support this type of system is not clear; their current financial constraints do not make them unduly adventurous at present.

ACKNOWLEDGEMENTS

This work has used the results of many colleagues in the ANDROID, 6WINIT and MECANO projects; most are identified on the web pages referenced below. We also acknowledge the support of the European Commission who supported these three projects, and DARPA who supported a companion project called RADIOACTIVE.

REFERENCES

[6winit] http://www.6winit.org/

[ANDROID] http://www.cs.ucl.ac.uk/research/android/

[damn00] Damianou N., Dulay N., Lupu E., Sloman M., Ponder: A Language for Specifying Security and Management Policies for Distributed Systems, Imperial College Research Report DoC 2001, Jan. 2000].

[fry99] Fry, M. and Atanu Ghosh: “Application level active networking”, Computer Networks, 31 (7) (1999) pp.655-667

[kirst00] Kirstein, PT, et al: "A Secure Multicast Conferencing", DISCEX 2000, pp 54-63, IEEE Computer Society, 2000.

[kirst02] Kirstein, PT et al: “The Radioactive Networking Architecture”, DARPA Active Networks Conference and Exposition, 29 – 30 May, 2002, San Francisco, USA, pp394-408, IEEE Computer Society

[mars01] Marshall, IW *et al.* “A Novel Architecture For Active Service Management”, IFIP/IEEE International Symposium on Integrated Network Management (IM 2001)

[mars99] Marshall, IW et. al., "Application-level Programmable Network Environment", BT Technology Journal, Vol. 17, No. 2, April 1999.

[mbone] http://www-mice.cs.ucl.ac.uk/multimedia/software/

[MECCANO] http://www-mice.cs.ucl.ac.uk/multimedia/projects/meccano/

[tenn97] D. Tennenhouse, J. M. Smith, W. D. Sincoskie, D. J. Wetherall and G. J. Minden, “A Survey of Active Network Research”. IEEE Comms Magazine, vol. 35, no. 1, 1997.

[XML01a] [XML Schema Part 0: Primer W3C Recommendation, 2 May 2001 http://www.w3.org/TR/xmlschema-0,

[XML01b] XML Schema Part 2: Datatypes W3C Recommendation 2 May 2001 http://www.w3.org/TR/xmlschema-2.