Project Number:	**IST-2001-32603**
Project Title:	**6NET**
CEC Deliverable Number:	**32603/UCL/DS/4.3.1/A1**
Contractual Date of Delivery to the CEC:	December 31st 2002
Actual Date of Delivery to the CEC:	March 28th 2003
Title of Deliverable:	First set of IPv6-enabled Dynamic VPNs running
Work package contributing to Deliverable:	WP4
Type of Deliverable*:	R
Deliverable Security Class**:	PU
Editors:	Piers O'Hanlon, Peter Kirstein, Manish Lad
Contributors:	Felix Garcia Clemente, Panos Gevros, Antonio F. Gomez Skarmeta, Kristian Hasler, Peter Kirstein, Manish Lad, Piers O'Hanlon, Gregorio Martinez, Joe Spagnolo

* Type: P - Prototype, R - Report, D - Demonstrator, O - Other

** Security Class: PU- Public, PP – Restricted to other programme participants (including the Commission), RE – Restricted to a group defined by the consortium (including the Commission), CO – Confidential, only for members of the consortium (including the Commission)

Abstract:

VPN technology is widely deployed within IPv4, however the provisioning of dynamic VPN technology still remains very much an open research issue. This document outlines the investigation carried out into the issues surrounding the deployment of dynamic VPN technology within the specific context of IPv6. A number of VPN infrastructures aiming to provide dynamic VPNs have been analysed providing a status and a review of suitability. Consideration is also given to applications that may use an underlying dynamic VPN infrastructure, with specific focus on the context of conferencing environments. These have been supported by Active Networking, though other options are considered.

Keywords:

VPN, Dynamic, IPv6

Executive Summary

Virtual Private Network (VPN) technology has been, and increasingly continues to be, widely deployed within IPv4 inter-network environments. However, such deployment usually involves a high degree of static configuration. Provisioning for dynamic VPN set-up is still very much an open research issue. This Deliverable was intended to set up a technology for use of IPv6-enabled Dynamic VPNs, based on work done in other projects. It was expected, of course, that the technology chosen would also be deployable over the 6NET pilot network.

Two technologies for such dynamic VPNs were developed to a demonstrable state in the ANDROID and RADIOACTIVE projects. They both used Application Level Active Network techniques in Edge Servers. The ANDROID one then used a normal, but proprietary, VPN management system. The RADIOACTIVE one used a novel hierarchic architecture called X-Bone. According to early statements in the Description of Work, it was the X-Bone version that we were to show running and then deploy.

While the X-Bone version was demonstrated publicly in June 2002 on a three-site basis, it became clear from this early work that the technology would not scale in its current format. The form of routing used precluded a relation between VPN and physical topology. For the following four months, we pursued the ANDROID topology inside that project; again we achieved a demonstrable status, but were not convinced that this would produce a satisfactory VPN technology for 6NET – partly because of performance issues, and partly because the management system was proprietary.

Whatever was said in the DoW, the clear aim of this task is to provide an IPv6-enabled technology that can be deployed simply, and on a large scale, over the 6NET network. Hence we decided that to follow blindly the DoW, and merely describe a VPN that had been deployed, would not be a suitable base on which to build this task. We also decided to carry out an analysis both of the concepts behind the technologies and of their implementation. In addition, we explored two other technologies that appeared during the year. This report concentrates on the analysis of these technologies, and lays the foundation for our second year work. In fact as a result of this analysis, it is clear that the second year work will be re-directed to a different technology from that envisaged in the DoW. It is only these analyses, and the actual writing of this report, which are charged to the 6NET project.

In this report we will first consider some of the basic considerations that arise in the technology of VPNs. We describe both the two technologies we have been pursuing, and three others, which have emerged during the year. In each case, we describe the basis of the technology, give its status, and review its suitability. We study also applications that would use these basic technologies, and consider how they would cope with specific components that we have been using. Finally we outline our further plans to provide deployable VPNs.

Table of Contents

1. INTRODUCTION 5

2. DYNAMIC VPN OVERVIEW 7

2.1. VPN MANAGEMENT 7
2.2. SECURITY 7
2.3. TOPOLOGY 8
2.4. ROUTING 8
2.5. HIERARCHICAL VPNS 8
2.6. PUBLIC KEY INFRASTRUCTURE (PKI) 9
2.7. IPV6 DEPLOYMENT 9

3. CURRENT INFRASTRUCTURES 10

3.1. INTRODUCTION 10
3.2. NETCELO/6WIND 10
3.2.1. Mechanism 10
3.2.2. Status 11
3.2.3. Review 12
3.2.3.1 Topology consideration criteria 12
3.2.3.2 Routing 13
3.3. THE ISI X-BONE 13
3.3.1. Mechanism 13
3.3.2. Status 14
3.3.2.1 Address Space Scarcity 14
3.3.2.2 Extending the X-Bone for IPv6 Overlay Deployment 14
3.3.3. Review 16
3.4. THE UMU-PBNM SYSTEM 17
3.4.1. Mechanism 17
3.4.2. Status 19
3.4.3. Review 19
3.5. DVC VPN 19
3.5.1. Mechanism 19
3.5.2. Status 20
3.5.3. Review 20
3.6. THE CISCO IPV4 VPN 20
3.6.1. Introduction 20
3.6.2. Mechanism 20
3.6.3. Status 22
3.6.4. Review 22

4. APPLICATIONS 23

4.1. INTRODUCTION 23
4.1.1. Programmable Networks 23
4.2. APPLICATION LEVEL ACTIVE NETWORKS 24
4.3. APPLICATION LEVEL MULTICAST SERVICE 24
4.4. THE CONFERENCE APPLICATION 25
4.4.1. Overview 25
4.4.2. Use of the IPv6 X-Bone 25
4.4.3. Use of the Netcelo/6WIND 27
4.4.4. Extension to the UMU-PBNM System 28
4.4.5. Extension to the DVC System 29

5. FUTURE PLANS 30

6. CONCLUSION 31

7. REFERENCES 32

1. Introduction

Virtual Private Network (VPN) technology has been, and increasingly continues to be, widely deployed within IPv4 inter-network environments. However, such deployment usually involves a high degree of static configuration. Provisioning for dynamic VPN setup is still very much an open research issue.

According to the Description of Work (DoW), the aim of this task is:

> To look at the issues surrounding the provision of IPv6 dynamic VPN technology. While the provisioning of IPv4-based VPNs is already standard, the provisioning of dynamic VPNs is still very much an open research issue, and will be considered in the context of IPv6 within this Activity. Other VPN technologies like MPLS based VPNs will also be evaluated.

According to the same source, we should have had a first set of IPv6-enabled Dynamic VPNs running by month 11, with this report a description of what we had done. On month 24, we should have had a particular technology, the X-Bone, running in an expanded form. By month 36, we should have extended this to several other technologies, and write an evaluation report.

The work on VPNs done over the first nine months of the project was actually carried out entirely under the auspices of some other projects; the technology being used there was dictated by the needs of those projects, and could not be changed. In neither case was it of paramount importance to provide a technology that would be suitable for the deployment of dynamic VPNs within the context of IPv6. Whatever was said in the DoW, the clear aim of this task is to provide an IPv6-enabled technology that can be deployed simply, and on a large scale, over the 6NET network.

It should be emphasised, however, that it was always the aim of the first year to work with technologies developed in other projects. It was our hope that these could be deployed later, without much change, in the 6NET environment. We did indeed demonstrate two technologies for VPNs; these are discussed later in this report. This was done with no manpower charged to the 6NET project; it was all done under other projects. We decided that to follow blindly the DoW, and merely describe a VPN that had been deployed, would not be a suitable base on which to build this task. We decided to carry out also an analysis both of the concepts behind the technologies and of their implementation. In addition, we explored three other technologies that appeared during the year. This report concentrates on the analysis of these technologies, and lays the foundation for our second year work. In fact as a result of this analysis, it is clear that the second year work will be re-directed to a different technology from that envisaged in the DoW. It is only these analyses, and the actual writing of this report, which are charged to the 6NET project.

Two technologies were pursued in our search for deployable IPv6-enabled dynamic VPNs:

- The set-up of VPNs using active networks and conventional management systems
- The use of hierarchical active networks

Work done within the Active Network Distributed Open Infrastructure Development (ANDROID) project at University College London (UCL) can contribute substantially toward an initial dynamic VPN infrastructure for deployment within 6NET. This involves a dynamically configured, fully meshed set of VPNs, set-up and controlled by a VPN Manager. Within an application-level context of video conferencing, sessions are initiated by Active Servers located at each site, and managed by a Reflection Manager. A complementary approach was adopted in the RADIOACTIVE project; here an architecture of hierarch IPv4-enabled VPNs was developed at the Information Sciences Institute in Santa Monica, USA. This was ported to IPv6, and linked to a similar set of Active

Servers and Reflection Managers by UCL. Some lessons from this work are discussed here, but we show that the extension of neither approach is really suitable.

In this report we will first consider, in Section 2, some of the basic considerations that arise in the technology of VPNs. In Section 3 we describe both the technologies we have been pursuing, and three others that have emerged during the year. In each case, we describe the basis of the technology, give its status, and review its suitability. In Section 4, we consider applications that would use these basic technologies, and consider how they would cope with specific components that we have been using. Then in Section 5 we present our immediate plans, and in Section 6 present some early conclusions.

2. Dynamic VPN Overview

A VPN is a network built over the shared public IP infrastructure that operates with the security, management and Quality of Service (QoS) policies of a private network. A VPN is a cost-effective means of building and deploying private communication networks for multi-site interconnection. The VPN service provider connects multiple IP addresses at geographically dispersed sites so that they appear to be within the same private network. It is important to note that the term VPN may be interpreted in different ways, referring some cases to a private network just in terms of private address space, or QoS provision, and in other cases the use of encryption is the defining factor. In the context of this activity we will be more concerned with secured private networks.

VPNs can be considered a special case of what is known as *overlay* networks; these are isolated virtual networks created over an existing network. They are composed of *nodes* (hosts or routers) and *tunnels* (paths i.e. multiple hops, on the underlying network that appear as links in the overlay).

The use of dynamic VPNs essentially involves the establishment of each tunnel in an "on-demand" fashion as and when they are required and their ultimate teardown when no longer required. In the ideal scenario, this would mean that all such configuration, including requesting, negotiation, key exchange, security policy application, set-up and tear down would be carried out dynamically with no prior configuration or requirements apart from perhaps the assumption that all parties have been issued with valid keys for relevant negotiation.

However, a number of issues arise that must be taken into account during the establishment of dynamic VPNs.

2.1. VPN Management

Each VPN requires some sort of management system. The management system must set up the tunnels, and keep track of the tunnels that have been set up. It must provide the parameters for the topology that has been implemented; in the case of fixed routing, it must even provide the routing information. It must check the authentication and authorisation of nodes joining a VPN, and determine when nodes should be forced to leave the VPN. When a node leaves, the management system must also shut down the relevant tunnels. In most current VPNs, this management system is centralised, though some of the components may be distributed; examples of the latter are the time-out procedures, the re-routing in case of failures, and the way part of the set-up of secure tunnels is carried out. It is important to move towards a more distributed management system, where many of the decisions can be carried out in a distributed manner according to clearly definable management policies.

The VPN management system has a close analogy to a network management system, and very similar considerations apply to both.

2.2. Security

The initial security concern arises during the request for establishment of any given VPN, where a requesting user is required to be authenticated and authorised before any action should be permitted. This also implies the requirement of some form of policy infrastructure responsible for accepting, rejecting and controlling each VPN. One of the differentiators between different VPNs is the nature and flexibility of this policy infrastructure.

There also exists a need for an underlying security infrastructure providing mechanisms for authentication, authorisation, leading to establishment of secure key exchange and encryption. This would usually be provided by a Public Key Infrastructure (PKI).

Additionally, some form of monitoring and audit trail may need to be employed in order to preserve the dynamic nature of the VPNs, ensuring that they remain up for as long as required, yet providing a timely expiration when no longer in use.

2.3. Topology

The choice as to what topology a VPN takes is another important consideration. Additionally, in the case of a dynamic scenario, once a VPN has been established the way in which further nodes are added of is also of interest. The topology is an important factor in ensuring the most efficient use of network resources and bandwidth, and maintaining consistency in quality of service throughout the network, across multiple separate connection 'uptimes' over long periods of time.

There are a number of ongoing projects examining this general area, which provide differing approaches. We have examined the following:

- Netcelo VPN Management – provides a fully-meshed VPN topology
- ISI X-Bone – provides a dynamic discovery mechanism for topology construction
- UMU Policy Based Network Management (UMU-PBNM) system – focuses on dynamic and transparent VPN management, based on the UMU-PKIv6 structure [Gomez03b]
- Defence R&D Canada DVC – establishment of 'coalitions' maintained by user-defined security policies
- Cisco/Entrust VPN Connect System – VPNs with Cisco routers using Entrust PKI

Each of these projects will be discussed in more detail in the sections that follow.

2.4. Routing

Another important differentiator is the nature of routing within the topology. In general the VPN is an overlay network; thus it has no control impact on the underlying routing of packets through the network. Nevertheless, the VPN is constructed through a series of *tunnels*, which overlay the underlying network. These tunnels, together with their end-points, comprise the VPN. Packets can traverse from one end-point ***A*** to another end-point ***B*** only through one or more such tunnels. For example in the case of a fully-meshed VPN, each end-point is fully connected to each other; in that case routing decisions are only taken at the edge routers. However, for even moderate-sized VPNs, the number of tunnels required becomes excessive. For this reason, other topologies and routing polices are required, which may utilise static or dynamic routing.

For a deployable VPN, it is desirable that this topology has some relation to the underlying physical topology. While this is obviously desirable, and clearly normally occurs at the network layer, it does not always occur at the VPN layer. It requires that there be a measure of *distance* or *cost of transit* to determine the route. These measures are normally put in during the construction of the VPN, since it is only the VPN management system, which has the knowledge of the underlying parameters. Whether thereafter the routing between intermediate points is static or dynamic is another consideration; clearly the latter is more flexible and robust.

2.5. Hierarchical VPNs

The term "Hierarchical VPN" within this context refers to the creation of VPN tunnels using IPsec in a multi-level overlay structure by setting up tunnels within tunnels. Thus it would be possible to set up a number of separate VPNs overlaying one base VPN. These in turn may act as a base VPN for more VPNs. This mechanism has been implemented in systems such as the X-Bone described below. However, these aspects are beyond the scope of the current milestone.

2.6. Public Key Infrastructure (PKI)

The basic job of a Public Key Infrastructure (PKI) is to define the mechanisms used both to allow a recipient of a signed message to trust a digital signature and to allow a sender to find the encryption key for a recipient. It is comprised of the elements needed to manage and enable the effective use of public key encryption technology, particularly on a medium and large scale. The base components are a Certification Authority (CA), one (or several) Registration Authorities, (RA) and a directory server. Some other extra components, like smart cards, time stamping servers, OCSP servers, and so on, can be used depending on the services offered by a particular implementation of a PKI.

In the specific context of Dynamic VPNs, PKIs provide the means for authentication of users and nodes prior to their participation in a VPN. It should be noted that certain aspects of this procedure could also be achieved through static configuration. Two PKIs have been installed and tested: the Entrust Authority 5.0/6.0 and the University of Murcia Public Key Infrastructure (UMU-PKIv6).

The first, Entrust Authority 5.0/6.0, is a commercial PKI which currently has no support for IPv6. It consists of an LDAP server that publishes the certificates of the CA, users and processes. It also publishes the PKI CRL which checks if a certificate is valid at a particular time. A VPN Connector is also included to issue certificates to CISCO routers. It makes use of the Informix software as a trusted database for the whole system and has an easy to use AutoRA system for the generation of the user key pairs and certification request.

The second, the University of Murcia Public Key Infrastructure (UMU-PKIv6), is characterised by the use of open-source software, like Apache, Postgresql, OpenLDAP or OpenSSL, and the use of the Java language for the development of every internal component. This PKI has support for LDAP directories, Java Cards and RSA smart cards. It also provides a policy-based management and a web-based registration method for final users. In addition to this, three advanced services are implemented in this infrastructure: a Time-Stamping Authority, an OCSP Authority and a SCEP server, able to accept certificate request from secure gateways. The UMU-PKIv6 is operates fully over IPv6 and is thus the logical choice for 6net.

The planned cross-certification linking of UCL's installation of the UMU-PKIv6 with the University of Murcia's PKI will be a collaborative activity between 6NET and Euro6IX. This will provide wider opportunities for VPN testing and implementation.

2.7. IPv6 Deployment

All the infrastructures examined have, in the first instance, arisen from within an IPv4 context and so their main implementation effort has been to design and enable dynamism in VPN infrastructures within an IPv4 environment. Some of these, notably the Netcelo VPN Management, UMU-PBNM and the X-Bone have been ported to the IPv6 environment; although they provide support for IPv6 to the transport layer and above they still rely to some degree on an underlying IPv4 infrastructure to be present for certain management functions. Most of the management functionalities are not properly IPv6 enabled, an exception being the case of the UMU-PBNM system.

3. Current Infrastructures

3.1. Introduction

There are many infrastructures for the implementation of VPNs; almost all of them are only IPv4-enabled at this time. Since most are proprietary, it is only possible to consider their modification if we have a special relationship with their suppliers. For this reason, we present here a selection of such VPNs with which UCL has been working, and which could have a future for the project.

The first of these, the Netcelo/6WIND one discussed in Section 3.2, was the basis of the ANDROID project, and was deployed in its IPv6 version, in late 2002 for a five-site WAN demonstration at IST2002. The second, the X-Bone, was ported to IPv6 by UCL, and was used in the RADIOACTIVE project; again this was deployed over a three-site WAN demonstration at DANCE 2002. It was largely developed by the Information Sciences Institute, and would be available in source code. The third, described in Section 3.4, has been developed at the University of Murcia; UCL has collaborated in small parts of its PKI. It would be available to UCL under the collaboration between 6NET and Euro6IX. The fourth, developed at NRNS Inc. for Defence R&D Canada is available under an International Collaboration Board (ICB) agreement with UCL; it is still only IPv4-enabled, but discussions are in progress for this being IPv6 enabled. The fifth, described in Section 3.6, is based on standard Cisco/Entrust technology releases. UCL worked with this two years ago, and we believe that there are still no plans to provide IPv6-enabled versions of the ENTRUST software. By a combination of the University of Murcia and the Cisco system, it might, however, be a good basis for the 6NET VPN.

3.2. Netcelo/6WIND

3.2.1. Mechanism

During the ANDROID project, extensive use was made of the Netcelo/6WIND VPN management infrastructure. This is essentially a client-server based system consisting of a single "VPN Manager" that is responsible for co-ordinating each site requiring connection to the VPN.

Each site has a "Requestor" client that communicates with the VPN Manager whenever that particular site requires connection to the VPN. The communication is in the form of XML text messages that signify a Join, Leave, Acknowledgement or Update. The first two messages are sent by the Requestor and indicate whether a site chooses to join or leave the VPN due to the presence (or lack thereof) of traffic requiring transport over the VPN. An Acknowledgement is sent by the VPN manager to verify that a Join/Leave XML message was received. Finally, the Update message contains the current list of sites that are members of the VPN. This list indicates which nodes should receive the encapsulated packets.

The VPN Manager then communicates with the router at the requesting site in order to configure and activate tunnels to each of the sites that are currently members of the VPN. This is done for each new request and creates a fully meshed VPN structure.

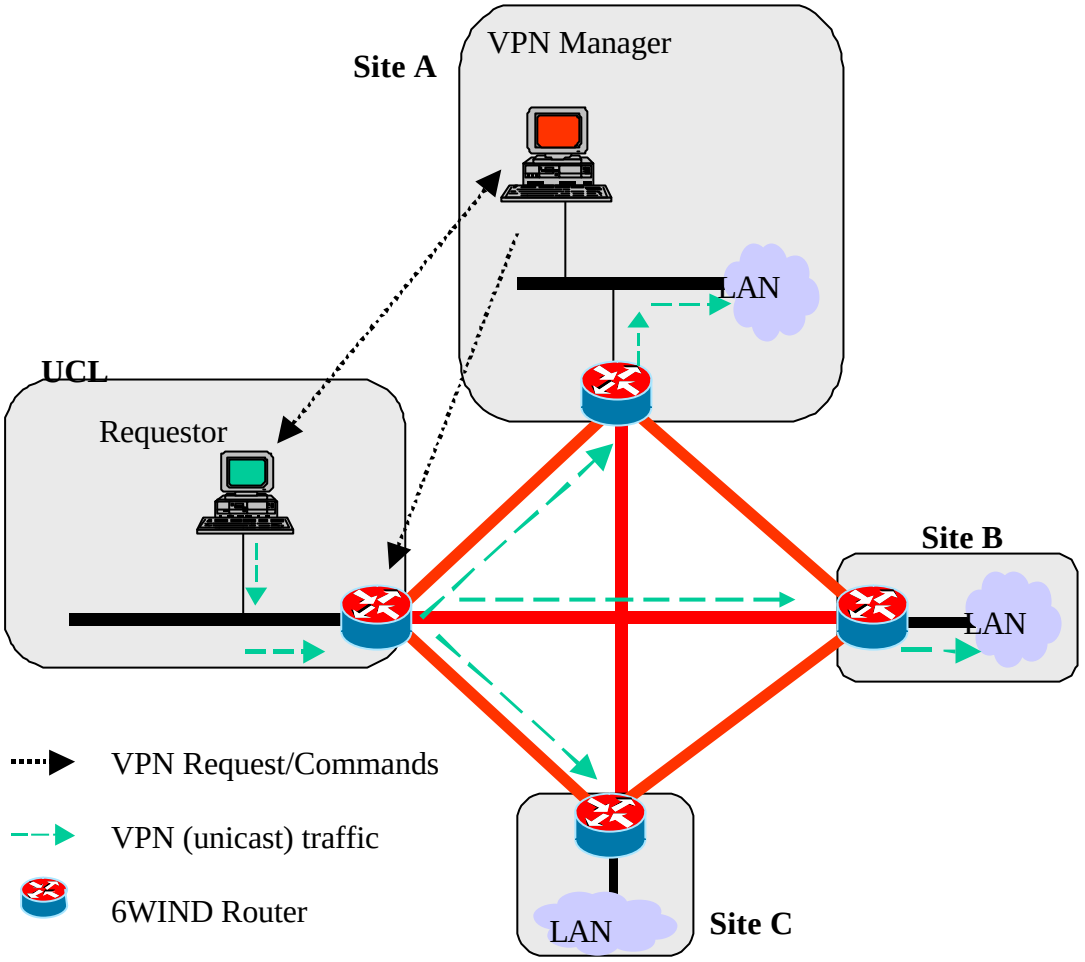

Figure 1

Figure 1 above shows a simple scenario of the overall mechanism of this approach. Once the newly configured site has become part of the VPN, traffic is forwarded to all other member sites as necessary.

Each "Requestor" repeats the transmission of Join messages at regular intervals in order maintain the soft-state and to keep the tunnel up, otherwise the VPN Manager will initiate the removal of the site from the VPN.

3.2.2. Status

The final network set-up used within the ANDROID project involved the above infrastructure extended to a number of sites:

- 6WIND, France
- BT, UK
- Netcelo, France
- UCL-CS, UK
- UCL-EE, UK

3.2.3. Review

Essentially, the infrastructure can be extended further than that used within the ANDROID project for example to ten sites. However, the underlying fully meshed nature of the configuration system that is employed by the VPN Manager does pose a very serious restriction on the scalability of such an infrastructure on the wider scale.

It very quickly becomes infeasible and inefficient for each member site to be configured with tunnels in a fully meshed manner to every other member site in the VPN. Each site must duplicate each packet it transmits 'n-1' times where n is the number of members. In fact, the strain on the member sites and the overall VPN itself could even be observed with as few as the ten sites here, depending on the type and amount of traffic generated. Bandwidth intensive traffic such as high volume multimedia streams will quickly strain such a system.

Therefore, the topology structure needs to be reconsidered. For example, of the sites within the ANDROID VPN infrastructure listed above, it may be more efficient for a single "wide area" tunnel to be set-up between say UCL-CS into the main VPN, and another (local) tunnel set-up between UCL-EE and UCL-CS with UCL-CS relaying traffic from UCL-EE into the main VPN. This would reduce the overhead of a duplicated set-up, maintenance and tear down of tunnels between all other member sites and the two UCL sites that are geographically located within the same complex (and hence may take the same underlying path to each member anyway). This could even be applied at the next level to ensure that the UCL-CS connection is not duplicated with another near-by site, thus creating a hierarchical system. This does however introduce new issues.

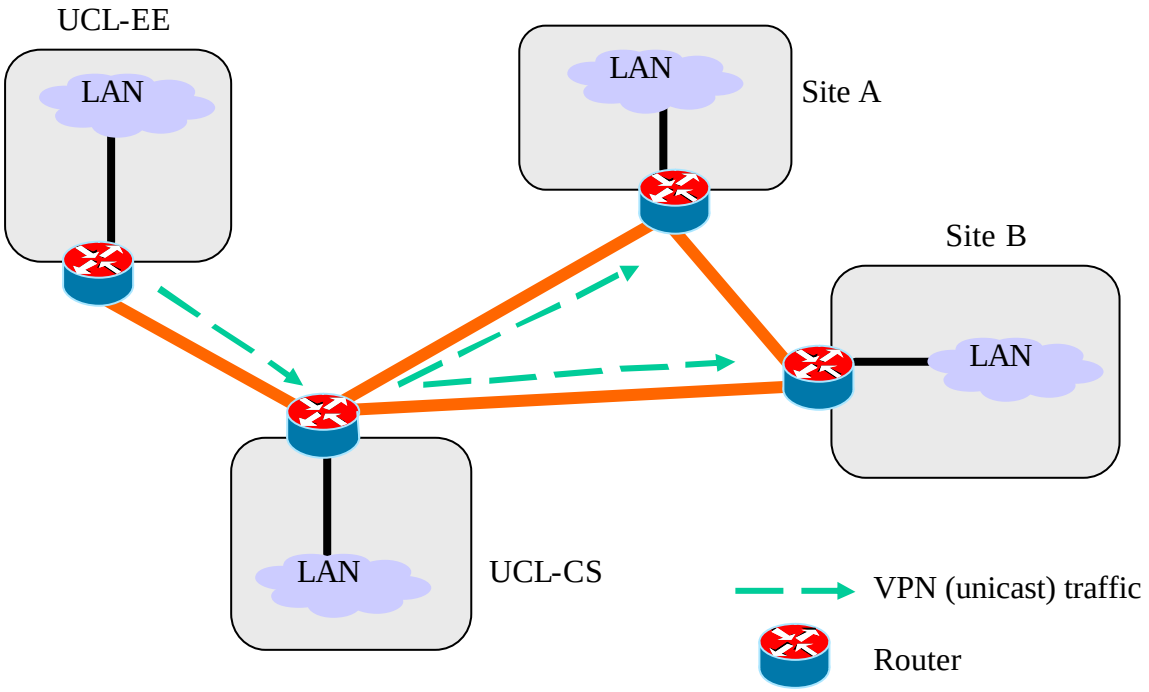

Figure 2

3.2.3.1 Topology consideration criteria

In the current infrastructure, the VPN Manager has the sole responsibility to maintain the VPN connections between all members of the VPN. The fully meshed nature of the infrastructure makes

the actual operation a relatively straightforward task: that of ensuring all member sites are configured to connect to all other member sites.

The removal of the fully-meshed nature introduces the need for a more sophisticated topology construction system. Who decides where a new member should be connected within the network? There could be several possible solutions to this:

- the VPN Manager
- the new member
- a third party server
- a distributed querying solution
- a self-correcting system based on lowest latency analysis

Each of the solutions, and any others that are not listed, will introduce an added complexity within the whole infrastructure.

3.2.3.2 Routing

The removal of the fully-meshed structure also imposes the need for some routing mechanism to be employed within the system, since there is no longer a direct VPN connection to each member.

3.3. The ISI X-Bone

The *X-Bone* [Touc01] is a system developed at USC/ISI for managing and deploying IPv4 overlays. Its goal is to reduce configuration effort and increase network component sharing. The X-Bone extends current overlay management by introducing dynamic resource discovery, monitoring and component reuse since the nodes (hosts or routers) can simultaneously participate in multiple overlays. It does not require OS specific or application specific modifications, only basic IP in IP encapsulation and existing implementations of dynamic routing and the Domain Name Service (DNS). Its key features include the support of recursive overlays (overlays built on top of other overlays) and the integration of IPSec and dynamic routing.

3.3.1. Mechanism

The X-Bone allows users to deploy overlays without human network manager participation. It manages inter-overlay resource contention by providing a uniform coordination point for overlays. This provides a framework for coordinating reservations, even between different mechanisms that manage a single class of resource. By making overlay establishment a fast, common function, the X-Bone enables new uses for overlays, such as for distributed applications without cumbersome application-level service location and routing support.

The X-Bone is a distributed system composed of *Resource Daemons (RDs)* and *Overlay Managers (OMs).* Users create overlays by sending a request to an OM using a Web-based GUI or sending a message directly to the OM API. The OM then uses multicast expanding ring search to discover available RDs and subsequently uses TCP (with SSL) connections to configure and monitor resources.

The RDs are daemon processes, which configure and monitor the resources on the nodes. They listen for invitations on a well-known IP multicast address (using S/MIME authenticated UDP) and respond according to their capabilities and available resources indicating willingness to participate in the overlay.

The OM selects an arbitrary subset from those RDs that responded and uses X.509 encrypted TCP/SSL connections to each chosen RD in order to transmit configuration information. The OM is responsible for determining the tunnel endpoint addresses and the routing table entries to be sent to each RD.

The basic X-Bone mechanism is *tunnelling* (or IP in IP encapsulation); tunnels allow incremental deployment when the routers in the underlying (*base*) network are lacking specific protocol capabilities. The X-Bone uses a *two-layer tunnelling* mechanism for each level of the overlay. This results in three IP headers in the case of an overlay on top of the base network. The additional layer permits the use of multicast, dynamic routing and IPsec inside the overlay since these are intrinsically network layer mechanisms.

The innermost header (*internal*) indicates the endpoints in the overlay and the next (*external*) header acts as a link layer for the overlay indicating the endpoints of the tunnel over which the packet is traversing, the final header (*base*) indicates the tunnel endpoints in the base network.

3.3.2. Status

3.3.2.1 Address Space Scarcity

Currently the X-Bone operates using separate, private IPv4 address spaces for the internal and external addresses of an overlay. Overlay addresses can be re-used among overlays that do not overlap. This is determined by the OM during the initial negotiation phase.

The problem of scarce IP addresses has been solved with the introduction of the new Internet Protocol; IP version 6. IPv6 provides bigger address space, seamless support for mobility and mandatory security features.

With Virtual Private Networks and/or overlay networks that are tightly controlled and centrally managed, like the X-Bone, address scarcity is generally not an issue. However there other reasons why an IPv6 overlay may be preferable compared to an IPv4 one. The difference is in the unique features of IPv6, better support for mobile nodes, mandatory security features, auto-configuration and better support for QoS. For instance with regards to QoS, in the current IPv4 networks it may be difficult for a router to determine the QoS Class of a packet, based on information found in the network layer packet header, especially when multiple classes of QoS are involved and encryption is used. However this is not the case with IPv6-based VPNs, which have the advantage that the class of service can be specified outside the VPN envelope of the IP packets.

3.3.2.2 Extending the X-Bone for IPv6 Overlay Deployment

For the RADIOACTIVE project's purposes, it was necessary to provide the modifications to the X-Bone in order to run using IPv6. The goal was to extend the X-Bone in order to facilitate the deployment of IPv6 overlays on top of existing IPv4 networks. The nodes participating in the IPv6 overlay had IPv6 support built into their network stacks.

With the X-Bone each level of tunnelling (IP encapsulation) is specified in a separate **"***XboneNodeCommand***"** command sent from the OM to the RD. Usually there are more than one "*tunnel*" command in each message and there are special keywords which control the order of execution.

Since IPv6 is not ubiquitous (although this is changing fast), minimal assumptions were made about the connectivity both between the participating RDs and between the RDs and the OM. This it was assumed that all X-Bone nodes were reachable using IPv4 and particularly that the OM used IPv4 multicast in order to communicate with the RDs.

Moreover the X-Bone is written in Perl and at the time of writing there were no TCP/SSL libraries that would operate over IPv6. However the capability of deploying IPv6 overlays is particularly important when the X-Bone components are potentially isolated by IPv4 clouds (see Figure 3).

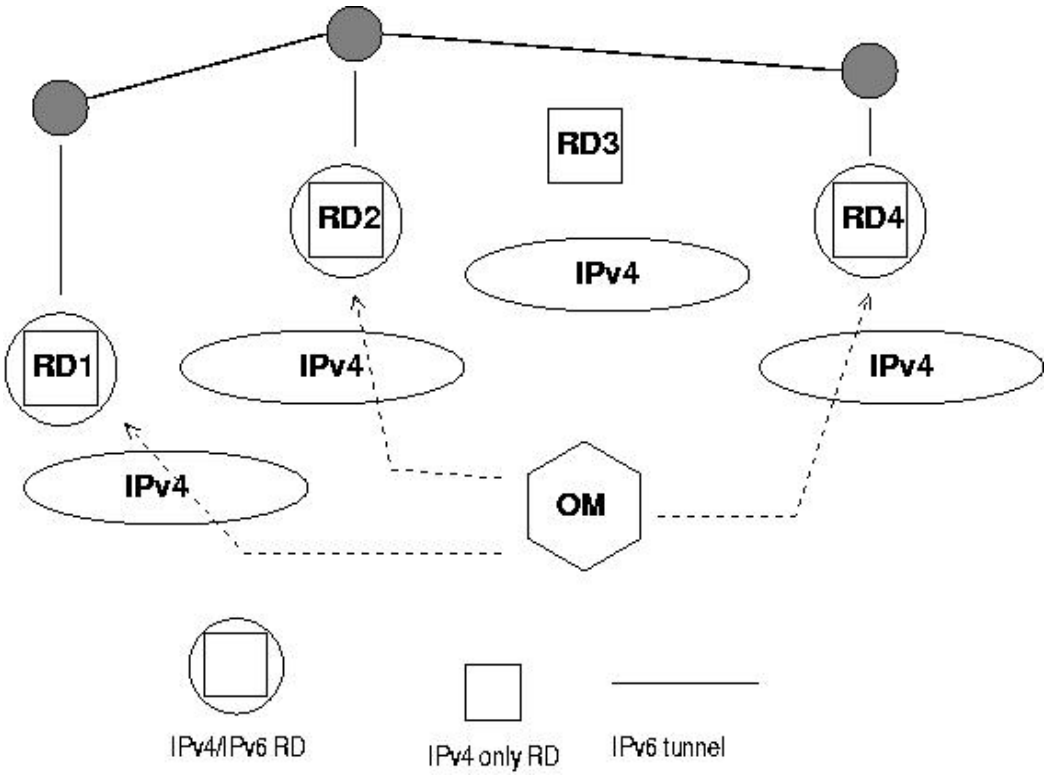

Figure 3: Deploying IPv6 overlays over IPv4 networks with the X-Bone

In order to achieve this, the creator of an X-Bone overlay should first indicate to the OM the IP version number required for that particular overlay (IPv4 or IPv6). Also the RDs should respond to the invitations by the OM for participation in an IPv6 overlay *only if* their network stack supports IPv6. The response is then sent back to the OM using UDP/IPv4.

After this initial negotiation and discovery phase the procedure followed by the OM is almost identical to that followed by the creation of IPv4 overlays, allowing for substantial code reuse and avoiding the duplication of address allocation procedures in the IPv6 case. This is achieved by treating the IPv4 addresses provided by the allocator (OM) as 32-bit blocks. Subsequently these blocks are used in the construction of IPv6 addresses by combining pre-pending a common IPv6 prefix to the IPv4 allocated block. Thus the X-Bone logic required for the creation of overlay networks becomes orthogonal to the IP version number.

The price paid for the simplicity is the requirement for all the RDs (those on the IPv6 capable nodes) to *share a common, pre-configured IPv6 prefix* to be used for the creation of overlays. The length of the prefix has to be less than 96 bits, given that IPv4 addresses are 32 bits long.

Alternatively the OM could provide them with the IPv6 prefix in order to ensure that all the RDs use the same prefix. However at the initial stage, the solution of the RDs being pre-configured with a common IPv6 prefix was chosen. This has the advantage that it does not require changes in the format of the protocol messages between the OM and the RDs. However this is a small price to pay

compared to the flexibility, the simplicity and the independence of the X-Bone protocol logic from the IP version number of the overlay.

Instead of requiring an additional IPv6 address allocation procedure or different types of messages, the IPv4 addresses are used with the pre-configured IPv6 prefix shared by the RDs (running on the v6 capable nodes) in order to construct appropriate IPv6 addresses for use in the IPv6 overlay. The OM could then use the same v4-to-v6 translation procedure for the DNS updates. Thus the Xbone protocol communicates tunnel endpoint numbering that can be extended in the IPv6 case.

The X-Bone can be used to bootstrap and manage an Active Networks (AN) infrastructure, deploying them on their own overlays. X-Bone also provides a platform to demonstrate the benefits of AN; although the X-Bone can be deployed prior to the availability of AN support, it can be implemented itself in AN technology.

3.3.3. Review

The X-Bone is a system for VPN deployment primarily intended for network research and network lab classes. It is a prototype implemented in Perl that has not been designed to address the requirements of wide area VPN deployment.

At the time of investigation, there was much untested/unused code in the distribution particularly that related to dynamic routing which could be left out of the official release. Because the X-Bone depends on so many components (WWW, DNS server, multicast, X.509 certificates, Perl modules, kernel modifications) the installation procedure is complex. For running a small test one requires a machine for the overlay manager (which usually hosts the DNS and the WWW server as well) and at least three machines (two hosts and a router) for running a modest test.

The biggest drawback of the X-Bone is considered to be the fact that there is no explicit way for determining which nodes get connected as neighbours ("routing adjacency") on a specific overlay. This could be achieved implicitly if the nodes were carrying specific attributes and the OM could decide based on these attributes how to better much user specifications. This limitation leads to sub-optimal topologies and results in poor performance (increased packet loss and delay) when the nodes are distributed over the wide-area public Internet.

Another drawback, which prevents the X-Bone from being a general-purpose VPN system, is the fact that it relies on multicast between the Overlay Manager and the Resource Daemons. This can be rather restrictive given that multicast is still not widely deployed in the Internet; in order to connect multicast capable islands over unicast-only public infrastructure another overlay (called the M-Bone – Multicast Backbone) has to be created using the **mrouted** IP multicast routing daemon. It would be an interesting combination to have the X-Bone automatically deploy an M-Bone overlay in case this is needed.

The X-Bone at the time of investigation supports only PC-based routers. Resource Daemons are effectively hosts and so must be running on PCs. It is not possible to include in the overlay a CISCO router for example. This could be done using the SNMP MIB. Furthermore, at the time of investigation, the X-Bone was untested with the Linux operating system and also IPSec was unavailable for Linux, so we were thus forced to use FreeBSD router systems.

The notion that X-Bone is mostly geared towards "local" environments is re-enforced by the fact that the whole process is centrally managed by the Overlay Manager which normally runs under a single administrative authority.

There is a whole area of research in establishing overlay networks in a distributed (or peer-to-peer) fashion.

Despite its shortcomings the X-Bone provides some useful features such as IPv4 IPsec and concurrent overlays. In summary, based on our experience with X-Bone and taking into account general requirements for VPN systems, we would like to see future versions of the X-Bone support in order of priority (preference):

- supporting IPv4 and IPv6 overlays concurrently,
- supporting multiple IPv6 overlays concurrently (fix the case where the same nodes need to be connected as neighbors in more than one overlays)
- ability for incremental adds to the overlay. The X-Bone is used to create overlays which provide VPN service. Currently the overlay topology is determined at set-up time and it is not possible to include new nodes in the overlay once it is established. The only way to address this is to temporarily bring down the overlay and create a new one but this will cause disruption,
- specification of topological, policy (or other) constraints in the role selection process (which nodes are allowed to be connected via tunnels to which on a given overlay),
- full-fledged IPv6 support. Currently the X-Bone requires IPv4 connectivity between the OM and the RDs. The configuration of the RDs uses the TCP Secure Sockets Layer Perl module (**Net::SSLeay.pm**).
- Another interesting extension would be to use the Application Deployment capabilities of the X-Bone for deploying the application-level active network infrastructure (FunnelWeb EEPs) to the appropriate nodes.
- Also useful and simple would be an extension of the command emulation program to handle not only the overlay set-up procedure (currently supported) but also the the overlay tear-down procedure.

The original aim was to specifically implement X-Bone-based solution with the result of a deployable system, where the bulk would be from RADIOACTIVE. However, following the investigations during RADIOACTIVE, we decided that this was not an effectively deployable solution.

3.4. The UMU-PBNM system

3.4.1. Mechanism

Members of the Departamento de Ingenieria de la Information y las Comunicaciones at the University of Murcia (UMU-DIIC), Spain, have been working on an IPSec Policy-Based Network Management (UMU-PBNM) system [Gomez03a] (which uses their UMU-PKIv6 [Gomez03b]), whose main objective is to establish an infrastructure for distributed systems that is secure and scalable. It is based on the IETF approach for policies [Policy], [IPSP] and makes use of a full set of standards (e.g. PCIM [RFC3060], XML [XML01a], [XML01b], XML Signature [RFC3275], HTTPS [RFC2818], LDAP [RFC2251], COPS [RFC2748] and COPS-PR [RFC3084]). Its architecture, which is depicted in Figure 4, is split into two different processes:

- Policy Definition (from the policy console to the policy repository)
- Policy Recovery (from the policy repository to the policy enforcement point)

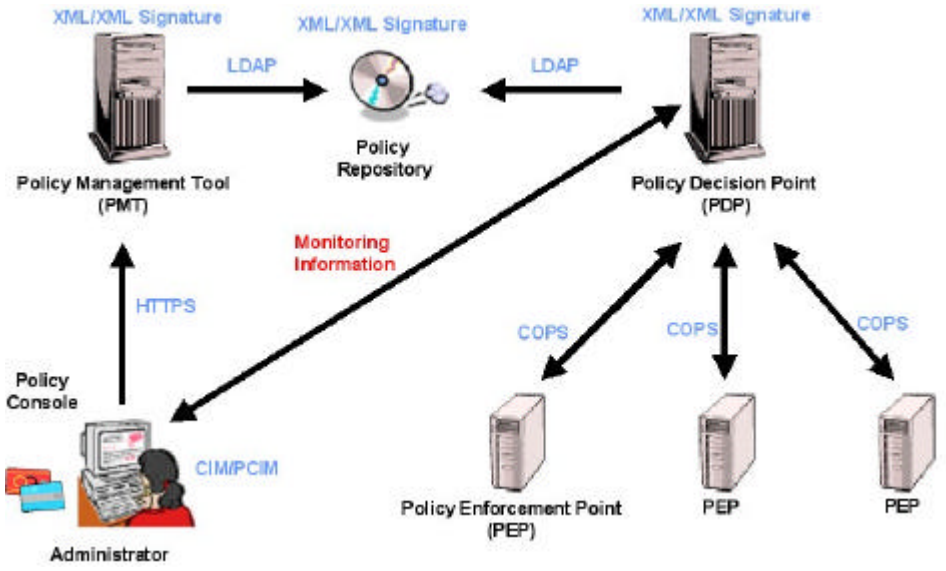

Figure 4: Architecture of the UMU-PBNM system

A short description of the main elements involved in this scenario is provided below.

- Policy Management Tool (PMT): defines a web interface to the network manager(s) – the administrator(s) of a particular security domain. This tool is in charge of:
  - Policy editing
  - Rule translation – from PCIM [RFC3060] to XML [XML01a], [XML01b]
  - Rule validation – to avoid possible conflicts with previously stored policies
  - Generation of the policy signature – done within a web client (named policy console) used by the adminisrator to access this tool. The private key of the administrator (which can be stored in a smart card or Java card) will be used. This private key and its associated certificate are managed by the UMU-PKIv6 [Gomez03b].
- Policy Repository: a database or directory based policy store, normally using LDAP [RFC2251] as the communication protocol
- Policy Decision Point (PDP): in charge of taking decisions on the behaviour (in the case of VPNs, security behaviour) based upon the policies defined by the network administrator(s). Its main tasks are:
  - Trigger detection and handling – for both the outsourcing and provisioning models
  - Rule location – according to the particular network conditions and the kind of node to be applied to
  - Resource specific rule validation
  - Device adaption – from XML (the format in which policies are stored in the policy repository) to vendor-specific commands (e.g Command Line Interfaces – CLIs)
- Policy Enforcement Point (PEP): in charge of enforcing the specific network-related policies. Its main tasks are:
  - Execution of actions
  - Optional device specific condition checking and validation – to avoid conflicts in the specific configuration of the final device

3.4.2. Status

Most of the aforementioned system is currently working at a lab level and is being tested before being released. This includes the recent deployment of the PDP and PEP using COPS [RFC2748] over IPv6. The currently missing features are the policy rule validation task of the PMT and the support of the COPS-PR [RFC3084] model.

A basic UMU-PBNM subsystem for the secure management of IPv6 VPNs was successfully tested in November 2002 during the IST Event 2002, with a VPN Enforcement Tool (VPN ETool) running as a secure web server, and a VPN Enforcement Point (VPN EPoint). Administrators are able to communicate with the VPN ETool via HTTPS and provide a policy description of the secure network domain. A VPN Enforcement Agent (VPN EAgent), effectively an SSH or SCEP client, running as part of the VPN EPoint, gets the policy definition from the VPN ETool and then communicates with the relevant routers in order to reconfigure them with the necessary tunnels, thus constructing the VPN. They currently work with 6WIND routers and support for Cisco routers is an ongoing task.

3.4.3. Review

The advantages of this system come from its strong focus on security and the secure establishment of the VPN infrastructure. It is possible to add new nodes using the system as necessary. Through the use of common trust hierarchies, it is possible for various nodes to be connected together using certificates assigned by their common PKI.

This system provides greater control over the network topology than either the X-Bone or the Netcelo VPN Management system. In such respects, scalability is less of an issue with regard to duplication of traffic in a fully meshed structure or the geographic connections in the overlay mechanism. However, this does make the system less autonomous in that an administrator/user is required to provide decisions in order to set-up the VPN infrastructure.

3.5. DVC VPN

3.5.1. Mechanism

The Dynamic VPN Controller (DVC) designed by NRNS Incorporated for Defence R&D Canada, "provides secure/authenticated out-of-band channels to establish, monitor and dismantle VPNs" [DVC]. It is based on the establishment of "coalition" partners who "need only maintain high-level information high-level information about each other – e.g. who they are and where they are on the network" [DVC].

The main concept and technology behind the system has been derived originally from the X-Bone but rather takes a more detailed look at the policy side of things. Like the X-Bone, it is largely written in Perl.

Essentially, each site runs a local DVC, which is connected to a common wide area network. Coalition partners connect their "participating network resources" through their local DVC. Each DVC maintains a local policy database and a web based CGI implemented user interface is provided. In order to establish a connection when a partner makes a request to join a coalition, the local DVC initiates a connection to a remote DVC via SSL. The initiating DVC provides its security policies, which may be passed up to the DVC Operators at the remote sites. If the remote

DVC Operator acknowledges the initiating DVC security policies, the remote DVC security policies are sent to the initiating DVC Operator. Upon acknowledging the remote DVC security policies, each DVC configures its local system in order to establish the required IPSec tunnel.

With the expanded policy support, when a coalition partner's access policies are modified within the local policy database, the DVC is notified and must re-negotiate the VPN connection terms. Similarly, if a coalition partner's access is terminated, the remote DVC is notified and the relevant VPN link is torn down.

3.5.2. Status

A DVC demonstrator is available with an implementation of the policy components, although its policy definitions and encoding mechanisms are still quite simplistic.

The DVC system is implemented on FreeBSD 4.4 making use of KAME IPSec implementation, the Zebra routing system, and ISAKMPD for dynamic key management.

Currently, the DVC system only supports IPv4 based connections, and so would need to be extended in order for deployment within an IPv6 environment.

3.5.3. Review

The DVC is the first such to system to succeed in examining the use of policies enforcement in a very thorough manner. This helps to move the policy management decisions to a more autonomous level than is currently possible elsewhere. The system could be improved however by standardising the manner in which it defines and encodes its policies. XML would be a good choice for policy encoding.

It provides dynamism in initiating and tearing down VPN connections through the use of the security policy negotiation system, but again at the cost that some level of static information is required to know where other coalition partners are within the network.

Like the Netcelo VPN Management system, the DVC establishes a fully meshed set of connections, and so for the same reasons described in section 3.2.3 above, faces problems of system scaling.

3.6. The Cisco IPv4 VPN

3.6.1. Introduction

UCL experimented in 2000/2001 with another system, which would seem to be a natural one to use in 6NET. It was based on the standard VPN releases from Cisco, coupled with a commercial VPN Connect software and a PKI from Entrust Technologies. This system is described below. Unfortunately no component of this system was IPv6-enabled at the beginning of 6NET – nor is yet. However, since we expect that the Cisco portion will soon be so enabled, and we see a way forward to use this technology, it is described here.

3.6.2. Mechanism

The mechanisms for the Cisco-based VPN set up in late 2001 consisted of a number of Cisco routers, running their VPN software, together with an Entrust PKI system.

The system was installed on a PC running Windows NT 4.0 Server. The various components (Entrust PKI/VPN Connector/Web Server) are as in Figure 5.

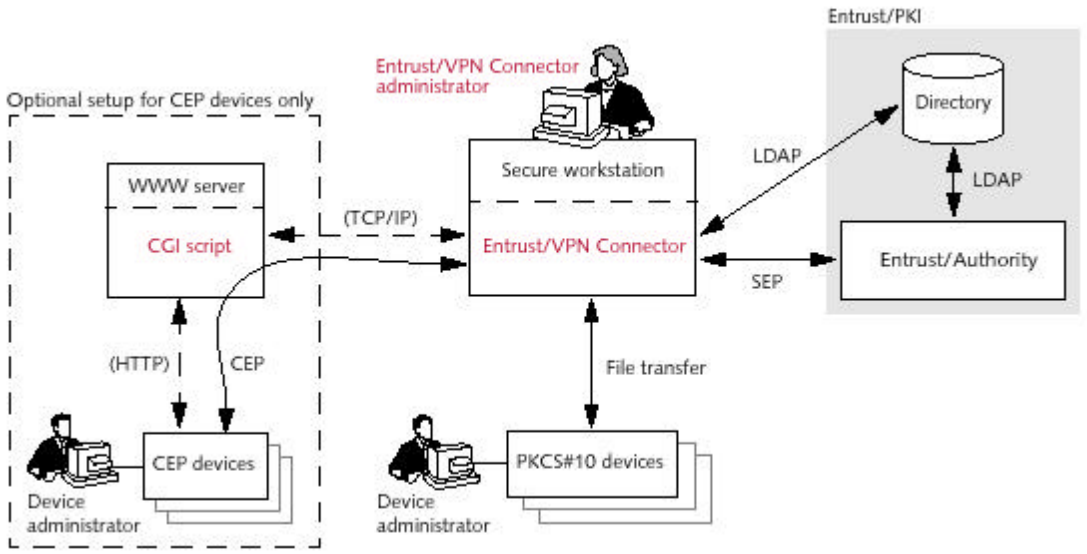

Figure 5: Architectural view of Entrust/VPN Connector (from the VPN Connector documentation)

The various components have all been installed on a single machine rather than being on separate machines as would be the case if this were a commercial environment rather than a development pilot. The architecture is as below. The Simple Certificate Enrolment Protocol (SCEP) is used to enrol devices for use in the VPN.

For enrolment of SCEP devices it was necessary to install a web server and for this we are using Microsoft Internet Information Server (IIS) 4.0. The following software was installed and configured:

- Entrust PKI Release 5.0
- Entrust VPN Connector Release 4.1d
- Microsoft IIS (web server) Release 4.0
- Cisco Release 11.3.5 (T)

The Cisco router wishing to be enrolled by the CA uses http to send its requests to a web server, which has a CGI script configured to talk to the UCL VPN Connector. Entrust requires that the end-entity transact with a Registration Authority (RA) which will then forward the requests to the CA. Consequently, the end-entity must retrieve the public key of both the RA and CA. In addition, Entrust also requires the generation and enrolment of signing and encryption keys even though the encryption keys are not used for IPsec.

It is necessary for the Cisco router to obtain the certificates for the CA and RA and request certificates for itself. When a certificate is requested it is possible to choose to have the IP address and serial number of the router in the certificate as part of the Distinguished Name.

It is necessary also to configure an IKE (ISAKMP/OAKLEY) SA and IPsec SA. The IKE SA is generated by two communicating end-points using Main or Aggressive mode. Once this is established IPsec SAs can be negotiated using Quick Mode.

One needs to ensure that the Access Lists are configured to be compatible with IKE. The IKE negotiation uses UDP on port 500 so this port must not be blocked. Router control uses access-list

100 to control traffic incoming from the Internet and access-list 101 to control outgoing traffic. Once the access-lists have been configured the IKE polices must be created. When the two end-entities start IKE negotiation they look for an IKE policy, which is the same on both peers. There can be several policies, which can be ordered numerically. The IKE SA includes the following:

- Encryption : des-cbc (default), 3des
- Authentication: Pre-shared keys, Encrypted Nonces (IOS only), RSA Signatures (IOS default)
- Hash Algorithm: SHA (IOS default), MD5 (Client default)
- Lifetime: 86,400 seconds (IOS default), unspecified (Client default)

In IOS, if no IKE policy is explicitly defined, a default policy is offered comprising all the default values listed above. If the default is used it is not displayed in the router configuration. Access lists need to be configured on the routers to detail how incoming and outgoing traffic is handled.

3.6.3. Status

The above system was used for experimentation in early 2001. A four-node configuration was tested, with one node in Canada. This was found to work perfectly. No work has been done on the system since that time. The Entrust documentation states that versions 11.3.5 (T) onwards should work with VPN/Connector. At UCL we attempted to use 12.0.5(T) but this did not work with VPN/Connector at the time.

3.6.4. Review

The system was quite satisfactory, within its previously known limitations. These included the fact that it worked only with IPv4, and the IPv6 IPsec had not been released (nor has it yet, we believe). One approach would be to IPv6 in IPv4 IPsec tunnels, though this obviously doesn't provide native functionality. The configuration set-up was very cumbersome, and could be done only from a Command Line. We were told that a radical re-think of the configuration interface was planned. Using the system with the Entrust software greatly eased the usage.

Clearly all the mechanisms, even for a dynamic VPN were there. However it was premature to consider using the Ciscos for 6NET, because they did not support IPv6 in this mode. Moreover, Entrust had no plans to support IPv6 in their work. It was possible to set-up configuration scripts, which allowed the Ciscos to be used in a dynamic VPN scenario.

We conclude that, while it was impractical to use the Ciscos for the first year of 6NET, it should be possible to use it in subsequent years. We would recommend not using the Entrust software; we still have no indication that they are supporting IPv6, and it would be unadvisable to use an expensive proprietary system. However, it would be quite feasible to replace the Entrust system by the University of Murcia VPN system, under the auspices of the 6NET-Euro6IX collaboration. This may turn out to be just the same as the approach of Section 3.4.

4. Applications

4.1. Introduction

The dynamic VPN systems described in the previous section can be used as the underlying infrastructure for a whole range of different applications that have the basic set of requirements of:

- requirement for security policies
- requirement for QoS policies
- geographic distribution
- requirement for segregation.

This encompasses a vast subset of applications. During past projects such as ANDROID and RADIOACTIVE, much work has been done within the context of Active Networks (AN). This forms the basis of the work described here. It is expected to deploy the Active Gateway technologies in the context of the Transcoding Active Gateway of Work Package 5. It is also expected that these may be deployed over VPNs for security purposes. However, it should be noted that the use of Active Networks is orthogonal to the deployment of a dynamic VPN infrastructure, and other mechanisms may be considered.

4.1.1. Programmable Networks

Active Networking (AN) is based on the dynamic deployment of new services. It has its roots in the Defense Advanced Research Projects Agency (DARPA) projects; the field was surveyed by Tennenhouse in 1997 [Tenn97]. The main emphasis is on enhancing the functionality of IP networks where a number of problems have been identified, related to the "end-to-end" model in which computation is performed predominantly at network endpoints. The role of computation in the network is restricted to simple header processing for routing user data, which is transported opaquely. This approach has worked very well in the development of the Internet; however some applications can be greatly enhanced by exploiting the characteristics of the network to optimise the way user data is processed on intermediate network nodes. Several intermediate devices already require specific computational capabilities. Examples include devices supporting application-specific functions such as packet filtering, differentiated Quality of Service, tunnelling, intrusion detection and security. In addition, Network Address Translators (NAT) and Application Level Gateways (ALG) are used to provide end-to-end transparency for applications across multiple network domains. The IETF Network Working Group has identified the need for such intermediate devices (termed middle-boxes) and the importance of managing their operations [Midd01].

There are two main approaches to active networking, discrete and capsule [Tenn96]. The discrete approach separates the mechanism for injecting programs into a programmable node from the actual processing of packets as they flow through a node, with a separate mechanism for each function. Users send a program to a node and this program would then be stored at the node. When a packet arrives at the node, its header is examined and a program is used to process the packet. The program actively processes the packet, possibly changing its contents, or the resources associated with a packet.

The capsule approach leads to a more dynamic form of active network. Each packet in such a network contains both data and a program fragment (of at least one instruction) that may include embedded data. The term *capsule* is used to describe these new types of packets. When a capsule arrives at an active node, its contents are processed using the program in the capsule. The traditional router or switch, which is responsible for routing and header processing, is replaced by an active

node that consists of three other major components: a code loading mechanism, a transient execution environment, and a more permanent storage area.

Much of the DARPA work in active networking [DARPA] has concentrated on the capsule approach. In the ANDROID and RADIOACTIVE projects, we adopted a discrete approach to active networking. Loading of programs can be more easily controlled, and the functionality can be more sophisticated without the size restriction imposed by the size of the capsule.

Performance is often regarded as a problem in application layer active networking since data always passes up and down a full protocol stack. However, processing power or specialized computing resources can be made available much more easily at the application layer than within a network-level service such as a router.

4.2. Application Level Active Networks

Application level active network (ALAN) [Fry99] provides an environment in which developers can engineer applications in the network by providing platforms on which third party software can be dynamically loaded and run [Mars99]. The ALAN system consists of active nodes that are located in the existing Internet. These nodes run an Execution Environment for Proxylets (EEP). Proxylets are Active Applications, which are downloaded to the EEP and executed on behalf of users. Those applications provide functionalities that enhance the level of service or introduce new services, or active services, to the final user. End-to-end active services are provided by one or more EEPs executing one or more proxylets. Messaging uses XML [XML01a],[XML01b] which is transported over HTTP [Mars00a],[Mars00b].

The FunnelWeb software is an implementation of an EEP and was used during the ANDROID project. In the current implementation of FunnelWeb, the EEP runs as a Java Virtual Machine (JVM), with each proxylet running in a separately spawned JVM. Active server security and resources (consumed by the proxylets) are managed locally.

4.3. Application Level Multicast Service

There is much work being undertaken in the area of application level multicast services. The various projects include, for example, ALMI project [ALMI] allowing multicast over unicast spread as a spanning tree, CMU Overlay and Peer-to-Peer Networks Research Projects [CMU], and the Yoid Project [Yoid] a set of protocols for host-based content distribution. These systems have not been utilised in the current approaches though some may analysed to provide input for future work.

Work at UCL has focussed on Active Network approaches. The Transcoding Active Gateway (TAG) was developed from an earlier Universal Transcoding Gateway (UTG) [Has00] and utilises the Reflector Proxylet, building upon the Active Networking FunnelWeb architecture to support its functionality. Within the context of ANDROID, it has four primary roles:

- Requesting to join or leave a VPN,
- Requesting to join or leave a reflection group,
- Reflection of multicast traffic through a VPN.
- Filtering of traffic by selective blocking.

The TAG Client Proxylet reads from a configuration file to determine which multicast groups are targeted for reflection and then joins them. The conference RTP and RTCP ports are monitored for traffic (e.g. generated by tools like VIC and RAT). The existence of multicast data indicates that a

site is a source. The existence of RTCP reports indicates that a site has receivers. Upon detection of multicast receivers or senders, the TAG Client Proxylet communicates with the VPN Manager in the manner described in section 3.2.1.

Following this, the TAG Client Proxylet sends a Join message to a designated Active Server running the Reflection Manager service. This is responsible for managing a list of all registered TAG Reflector Proxylets for particular multicast groups. Upon receiving a Join, the Reflection Manager sends the joining TAG Client Proxylet message for each of the other already registered TAG Reflector Proxylets within the session group. The TAG Client Proxylet then starts a new instance of a TAG Reflector Proxylet, which begins receiving the multicast data and reflecting it via unicast across the VPN architecture to each of the other registered TAG Reflector Proxylets within the session group. This reflection can take the form of IPv4 or IPv6 depending on support within the network and thus, IPv6 originated data can be reflected to remote sites that support only IPv4. On receiving a leave message from any TAG Client Proxylet, the Reflector removes the appropriate TAG Reflector Proxylet from its list and sends a message to all remaining registered TAG Reflector Proxylets informing them of the leave.

4.4. The Conference Application

4.4.1. Overview

In earlier projects [Kirst00] UCL has adapted the M-Bone conferencing tools running in workstations; these consist of audio (RAT), video (VIC), shared workspace (NTE). They also developed secure versions of the tools There are a number of basic applications that run in servers; these include the Secure Conferencing Store (SCS) for holding information on conferences, and a java applet (SPAR) [Kirst00], [Kirst99] for parsing Session Description Protocol (SDP) and starting conference tools. Others include the Multimedia Conference Recorder (MMCR) [Lam98] for media recording and playback, and the Transcoding Active Gateway (TAG) for media transformation [Has00]. The media tools themselves use a combination of C/C++ and TCL/TK. All the software running in the servers is written in Java, with the exception of the SCS, which is written in Perl and runs on an Apache server [APACHE].

The M-Bone tools can run in both IPv4 and IPv6. They can use unicast or multicast – though some organizations and routers do not support multicast. Some work was needed to track changes in the underlying IPv6 stacks on various operating systems, but this was a straightforward activity.

4.4.2. Use of the IPv6 X-Bone

A version of the X-Bone with IPv6 support (extended during the RADIOACTIVE project) was demonstrated at the DARPA Active Networks Conference and Exposition (DANCE) at San Francisco in May 2002. The demonstration scenario involved three different sites (University College London, ISI at Los Angeles and the demo site at San Francisco). The X-Bone was the enabling technology for establishing an IPv6 Virtual Private Network (VPN) between the three sites, the VPN was used for demonstrating IPv6 multicast videoconferencing with active applications handling the multicast to unicast translation at the edges of the VPN (X-Bone overlay) since the overlays do not support multicast.

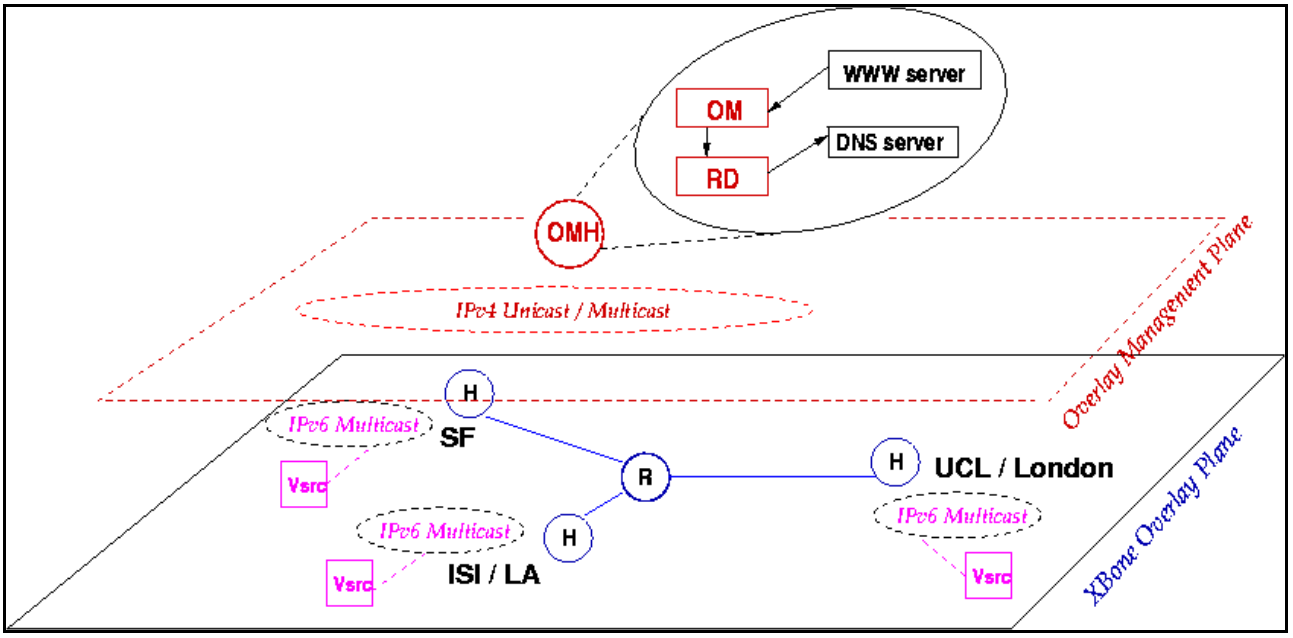

Figure 6: DARPA demonstration scenario

Figure 6 shows a logical diagram of the networks involved in the DANCE demonstration. Each one of the three sites has a local X-Bone host (H) where an IPv6 capable RD is running. There are vic video source(s) (Vsrc) running on different host(s) on each site. The Vsrc hosts are able to communicate using IPv6 multicast with the local X-Bone host (H).

IPv4 unicast connectivity is necessary between the nodes participating in the X-Bone overlay. IPv4 multicast is also necessary between the Overlay Management Host (OMH on the upper plane in Figure 6) and the X-Bone Overlay nodes (marked as circles in the lower plane). There was a multicast IPv4 tunnel (not shown in the Figure) established between two machines, one at UCL and the other at ISI, running the DVMRP multicast routing daemon (mrouted) in order to bypass the wide-area public Internet which does not support multicast. Nevertheless there was multicast between ISI and the demo site at San Francisco. IPv6 multicast is required only within the participating sites (in its simplest form this can be a LAN supporting native IPv6 multicast) connecting the Vsrc with the X-Bone Host (H).

The demonstration proceeded as follows; a human user initiated the creation of an overlay with a star topology using the Web GUI, which interacted with the OM at ISI. There was only one X-Bone router available, located at ISI. Therefore there was only one way in which the hosts at the three sites and the router at ISI could be connected in a star topology. However, this would not be true if we had decided to use another topology.

For instance had we decided to use a ring, each site would require at least one host and one router. Moreover, X-Bone does not provide topology optimisation based on certain constraints (such as locality in the underlying network topology). X-Bone does not provide any assurances about connecting the local X-Bone Host with the local X-Bone Router. Thus it is possible that a host at the demo site could have been connected (by tunnels) to a router at UCL. In this case the same packets would have been transmitted over the same underlying links several times before reaching their final destination as a result of the non-optimised topology.

The main purpose of the X-Bone is as an experimental vehicle for hierarchical VPNs. It clearly would need much more control over the topology relation between the physical and the logical nodes in order to use the X-Bone for a useful wide-area hierarchical VPN.

4.4.3. Use of the Netcelo/6WIND

As mentioned, the Netcelo/6WIND VPN management system was used as an underlying VPN infrastructure. The overall set-up involved a total of five sites:

- 6WIND
- BT
- Netcelo
- UCL-CS
- UCL-EE

Each site had an Active Server running an instance of the TAG Client Proxylet, sitting behind a 6WIND router.

Video streams transmitted using the M-Bone tool VIC were used to invoke the establishment of the VPN infrastructure via the VPN Manager as described in section 3.2.1. As with the X-Bone demonstration described in section 4.4.2 above, the active applications were used to handle the multicast to unicast translation at the edges of the VPN since the multicast traffic cannot be transmitted over the VPN.

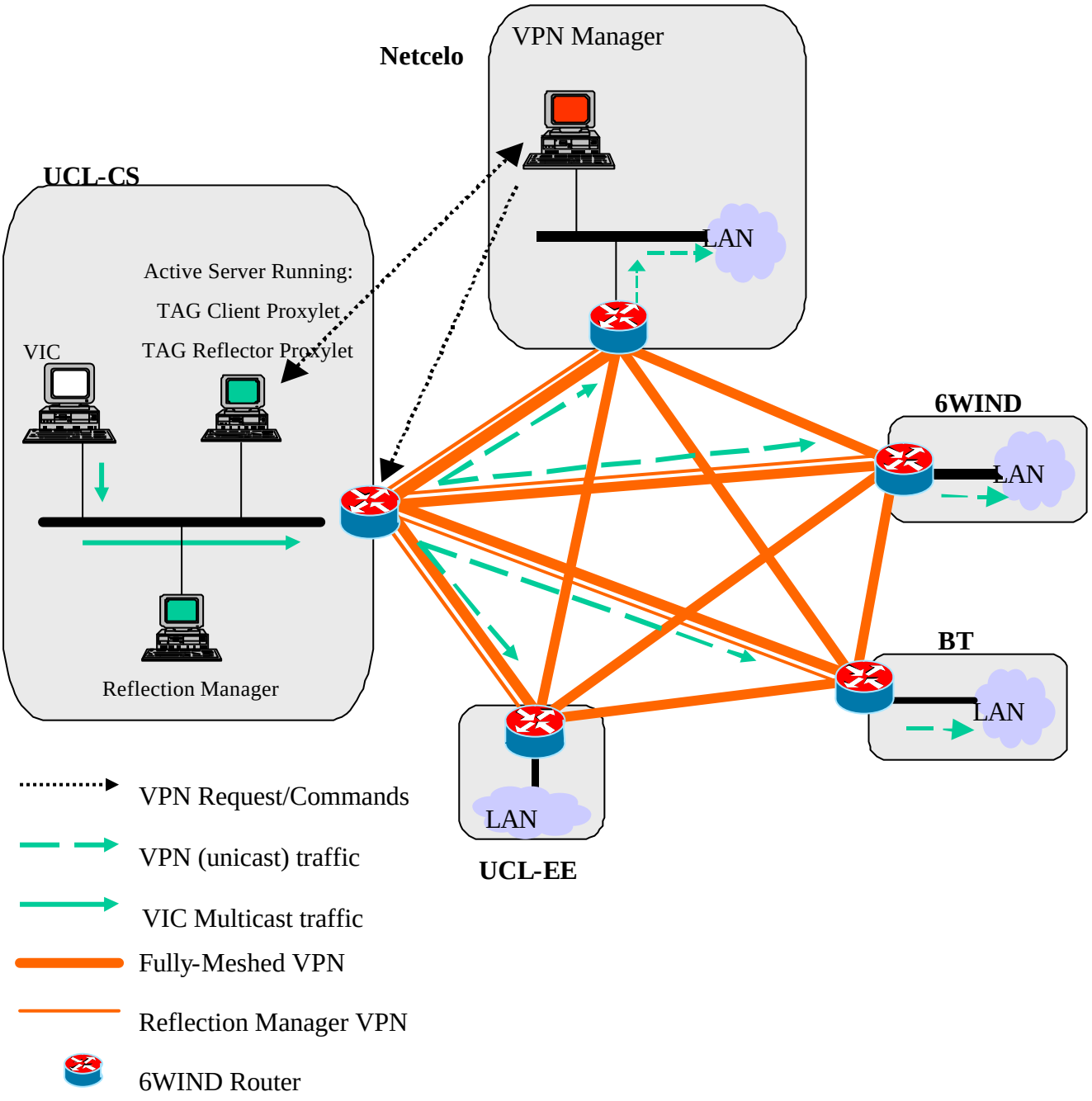

Figure 7

It should noted that during ANDROID, a BT system named the Management Information Distribution (MID) server was utilised in order to facilitate the distribution of XML defined Events and Policies. Since this is a proprietary and non-essential component, in this context, it is has been removed from future implementations.

4.4.4. Extension to the UMU-PBNM System

The main extensions required on the active applications side, in order to utilise the UMU-PBNM system would be to incorporate the ability of the TAG Client to communicate with the VPN ETool

(or VPN PDP in general) in much the same way as it does with the Netcelo VPN Manager. Initial expectations are that this will be quite conceivable, especially with the future availability of the Policy Definition Process in order to manage access control issues. This would rely upon the prior establishment of a trusted hierarchy between sites likely to participate within the VPN and also the prior configuration of the policies as required.

4.4.5. Extension to the DVC System

One of the main issues with regard to the DVC system is its lack of IPv6 support. This would be the first extension that would be required in order to utilise such a system.

Further extensions would be for each site's TAG Client to act as a "coalition" partner and registering with a local DVC to establish the VPN infrastructure. Again, this would rely upon the set-up of an initial set of security policies within the DVC architecture.

5. Future Plans

The first future task is to put together the components of the final chosen solution. This may be either the full UMU-PBNM system with Cisco routers, or the DVC system with the UMU-PKIv6, after looking into any problems in the DVC approach with respect to not tying topology to geographic location.

At the time of writing, the 6WIND routers were the only available routers supporting the necessary IPv6 and IPSec functionality required for the Netcelo VPN Management set-up. Cisco routers did not support IPSec for IPv6. A future aim is to increase the area of testing to encompass other routing devices for wider compatibility testing.

Another future goal is to examine the possibilities for greater integration with Road Warrior technology within the dynamic VPN infrastructure.

Another technology Multiprotocol Label Switching (MPLS) is widely deployed within ISPs and may be used for provision of VPNs on IPv4. MPLS is largely a layer two technology, utilising switch based infrastructures for traffic engineering of networks. MPLS provides for creation of Label Switched paths over the infrastructure, routing occurring at the edges. Full IPv6 support for MPLS is beginning to emerge, which we plan to investigate the suitability of, for the provision of dynamic VPNs.

Throughout the various scenarios described above, the one common limiting factor is that of multicast being limited to the internal network segments of each site. Externally, across the VPN infrastructure, traffic must be transmitted via unicast. One of the main future goals should be to investigate this in order to examine what needs to be done in order to enable the wider use of multicast to include transmission across the VPN infrastructure. This would remove the need for the reflection components within the ALAN architecture.

6. Conclusion

We have deployed two separate IPv6 enabled VPNs as required by the milestone. Further, we have gone beyond the requirements by analysing what would be desirable within the second year. However, there still remain a number of outstanding issues that need to be examined and solved before we can be precise as to which technologies we propose to carry forward into deployable entities.

Part of the decision on which technology to carry forward will depend upon the level of co-operation we are able to receive from other parties; for example, the flexibility of the UMU group in the directions they wish to pursue their system with VPNs, and Defence R&D Canada in moving their system over to IPv6.

As mentioned, the development of an Active Network system is orthogonal to the dynamic VPN system. Therefore, whether we will integrate the dynamic functionality of VPN with the AS FunnelWeb is yet to be determined.

7. References

[ALMI] http://alminet.org/

[APACHE] http://www.apache.org/

[Can00] Oscar Canovas, Antonio F. Gomez, Gabriel Lopez, Gregorio Martinez. "Dynamic Virtual Private Networks". 2000 SCS EuroMedia Conference. Antwerp (Belgium). May 2000.

[CisEnt] "Reference Guide – Deploying IPsec". Cisco, Entrust. http://www.cisco.com/warp/public/cc/techno/protocol/ipsecur/ipsec/prodlit/dplip_in.htm

[CMU] http://www-2.cs.cmu.edu/afs/cs/project/cmcl-yhchu/www.overlays/

[DARPA] DARPA Active Network Program. http://www.darpa.mil/ito/research/anets/projects.html

[Dieg02] Ciaran Diegnan. "R4.4.2: Active Router Management System". Version 1.0. Netcelo. 13 September 2002.

[DVC] Dynamic VPN Controller (DVC) Demonstrator Project Report. Version 1.2. NRNS Incorporated. Canada. October 2002

[DVCAdd] Dynamic VPN Controller (DVC) Demonstrator Addendum Report. Version 1.0. NRNS Incorporated. Canada. October 2002

[Fry99] Fry, M and A. Ghosh, "Application Layer Active Networking" Computer Networks, 31, 7, pp. 655-667, 1999

[Gev01] P.Gevros, "Deploying IPv6 Overlays with the Xbone" http://www.cs.ucl.ac.uk/staff/p.gevros/xb-v4v6-demo/

[Gomez03a] Antonio F. Gomez, Gregorio Martinez, Oscar Canovas. "New security services based on PKI". Future Generation Computer Systems. Elsevier Science vol 19 (2003) page 251-262

[Gomez03b] Antonio F. Gomez, Gregorio Martinez, Oscar Canovas, Gabriel Lopez. "PKI Services for IPv6 Networks in the context of the EU Euro6IX project". IEEE Internet Computing. To appear May-June 2003.

[Has00] Hasler, K, "The UCL Transcoding Active Gateway" http://www-mice.cs.ucl.ac.uk/multimedia/software/tag/

[ICBVPN] ICB Virtual Private Network. University College London. Version 1.06.

[IPSP] IP Security Policy (ipsp) Working Group. IETF. http://www.ietf.org/html.charters/ipsp-charter.html

[Kirst99] Kirstein, PT, et al, "Secure Multimedia Conferencing, A Secure Multicast Conferencing Architecture", Proc. 4th Int. Distributed Conf., Madrid, 1999.

[Kirst00] Kirstein, PT, et al, "A Secure Multicast Conferencing", DISCEX 2000, pp 54-63, IEEE Computer Society, 2000.

[Kirst01] RADIOACTIVE project. http://www.cs.ucl.ac.uk/research/radioactive

[Kirst02] 6WINIT project http://www.6winit.org

[Lam98] L. Lambrinos, P. Kirstein, and V. Hardman, "The Multicast Multimedia Conference Recorder", Proceedings of the 7th International Conference on Computer Communications and Networks, October 1998.

[Mars99] Marshall, IW et al, "Application-level Programmable Network Environment", BT Technology Journal, Vol. 17, No. 2, April 1999.

[Mars00a] W Marshall and M Banfield, "An architecture for application layer active networking", IEE Workshop on Application Level Active Networks: Techniques and Deployment", November 2000.

[Mars00b] Marshall, IW et. al., "Active management of multi-service networks", Proc. IEEE NOMS2000 pp981-3

[Mars01] Marshall, IW et al. "A Novel Architecture For Active Service Management", IFIP/IEEE International Symposium on Integrated Network Management (IM 2001), Seattle, 2001

[Midd01] "Middlebox Communication Architecture and framework", Internet Draft, Work in progress (draft-ietf-midcom-framework-06.txt) December 2001

[Policy] Policy Framework (policy) Working Group. IETF. http://www.ietf.org/html.charters/policy-charter.html

[Pries02] Richard Priestley. "UMU PKI Documentation". UCL. 2002.

[RFC2251] http://www.ietf.org/rfc/rfc2251.txt

[RFC2748] http://www.ietf.org/rfc/rfc2748.txt

[RFC2818] http://www.ietf.org/rfc/rfc2818.txt

[RFC3060] http://www.ietf.org/rfc/rfc3060.txt

[RFC3084] http://www.ietf.org/rfc/rfc3084.txt

[RFC3275] http://www.ietf.org/rfc/rfc3275.txt

[Touc01] J.Touch. et al, "Dynamic Internet Overlay Deployment and Management Using the X-Bone", Computer Networks, July 2001, pp. 117-135.

[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

[Tenn96] D. L. Tennenhouse and D. J. Wetherall. "Towards an active network architecture". Multimedia Computing and Networking '96, January 1996.

[Tenn97] D. Tennenhouse, J.M.Smith, W.D.Sincoskie, D.J.Wetherall and G.J.Minden, "A Survey of Active Network Research", IEEE Communications Magazine, vol. 35, no. 1, 1997

[UDcast] http://www.udcast.com/

[VPNDEF] http://www.vpnc.org/vpn-technologies.pdf

[Yoid] http://www.icir.org/yoid/