Active Networks and Security

Vijay Varadharajan, Rajan Shankaran, Michael Hitchens
Distributed System and Network Security Research
University of W.Sydney, Australia

July 2, 1999

1 Introduction

The main ingredient of an active network is that packets in these networks contain code that can be executed when they arrive at a node. That is, active networks introduce the idea of user controlled computing functionality in packets which when executed can modify and extend the behaviour of the network infrastructure and software. An important main objective of an active network is to use such a mechanism to increase flexibility and customizability of the network. Such networks are referred to as active in the sense that nodes can perform computations on, and modify, the packet contents. In addition, this processing can be customized on a per-user or per-application basis. In contrast, in traditional networks such as the Internet, although the routers may modify a packet's header, they pass the user data opaquely without examination or modification. Furthermore, the header computation and associated router actions are specified independent of the user process or application that generates the packet.

Given the flexibility of active networks and the ability of the nodes not only to modify the packet contents but also take actions based on the packet contents, security issues become very significant when it comes to the practical operation of active networks. Not only do we need to consider the standard network security issues, which by themselves can pose some important challenges in a large network environment, but we also need to understand the new types of threats that can arise in active networks and devise appropriate solutions.

The organization of the paper is as follows. Section 2 gives a brief background on the active network environment and mentions a range of projects on active networks that are currently being carried out. Section 3 discusses the security issues in active networks and looks at the different types of security threats that can arise in such an environment. Section 4 considers the provision of security services and mechanisms in active networks. It concentrates on authentication, integrity and access control services. Finally, section 5 briefly discusses some sample applications where the use of active networks can be advantageous and where security issues are significant.

2 Active Networks Overview

The concept of active networking emerged from DARPA research community in 1995 on future directions of networking systems. Several issues with existing networks were discussed including the difficulty of integrating new technologies and standards into shared network infrastructure, reduction in performance due to multiple protocol layers and the difficulty of adding new services into the existing architectural model. The idea of allowing messages to carry procedures and data emerged, which can be used to adapt the network to changing requirements. In practice, there can be several approaches as to how the active networks can be realized. Two approaches have been discussed in detail in recent times. One is the so called programmable switch approach which maintains the existing packet/cell format and provides a discrete mechanism that supports the downloading of programs. Separation of the downloading of programs from message processing can be useful especially when the selection of programs is done by one group of people (such as the network adminsitrators) different from the end users. For instance, this can allow router operators to dynamically load code into the routers for the purposes of router extensibility. In the other approach, sometimes referred to as the smart packet or capsule approach, the packets are replaced with small programs that are encapsulated in transmission frames and executed at each node along the path. User data can also be embedded within these capsules. With both these approaches, as network programs need to be transmitted and loaded into a range of platforms, there is a need for common models for encoding of network programs, the primtives required at each node and allocation of node resources.

Work on active networks is underway at a number of institutions on a range of topics [1] including capsule and programmable switch architectures, spcification techniques, end system issues and applications including mobility and congestion management. At MIT, an active network architecture based on capsule approach is being investigated [2]; at Pennsylvania University, the SwitchWare project is using a programmable switch approach that allows digitally signed type-checked modules to be loaded into the nodes of a network [3]; at Columbia, a language to script the processing of packet streams is being developed [7]; and CMU is looking at resource management mechanisms. There are also other work on programmability and data dictionaries at BBN [6], on congestion on Georgia Tech and several aspects of Pennsylvania design at Bell.

Though there has been some work on security as part of the SwitchWare project, there has not been an effort to do a comprehensive investigation of security issues in active networks. The SwitchWare project's security architecture is for the programmable switch approach based active networks which is primarily concerned with secure downloading of programs. There is a need to understand better the types of security threats and design choices in smart packet based active networks. This is the main objective of this paper. Furthermore, it seems that the typical reasons for deferring security considerations have again been evident in the case of active networks. These include, apart from the simple difficulty, the perceived negative consequences of making a system secure has on system's flexibility, usability and performance. In fact, in contrast given that the flexibility of active

networks makes it extremely susceptible to security threats, it is critical to take into account the security concerns early in the design process.

3 Security Issues in Active Networks

To understand the security threats in active networks, it is first necessary to consider the operations in an active network environment in more detail. The architecture we consider is a simplified version of a smart packet based active network. The network system has active packets called smart packets which can trigger specific processing when they pass through active network nodes. Smart packets can contain program and data wrapped within a smart packet header and encapsulated within the active network encapsulation protocol. There is a type field in the smart packet header which identifies what the smart packet contains in terms of programs and data. The Program section carries the code to be executed at the appropriate hosts. The Data section carries for instance the results of the execution; it may also include some informational messages. In addition to the smart packet header, the active network protocol header contains information such as the source and destination addresses. We will consider the required security related header information and the design of security mechanisms in detail in the next section after considering the security threats.

There are several security threats possible in such an environment. Consider the situation where a smart packet containing code to execute arrives at an active node C from B.

First, there may be a need to identify the sender of the smart packet. This can be either in terms of the identification of the active node B that is sending the packet and/or the identity of the client in the active node that is involved. Knowing that a smart packet comes from a particular node may not be adequate when it comes to determining the level of trust that can be placed on the packet. The trust can be dependent on the client and/or the node that originally created the packet. For instance, the smart packet might have been originally created by node A and it has passed through say node B to node C; in this case, there may be a need to identify both the sender B as well as the creator A in making authorization decisions. Hence an authentication service may be required that can authenticate both the sender as well as the creator of the packet. Furthermore, there may be a need to identify the application or the client in the node A that actually created the packet. For instance, this could be a network management application at node A.

Secondly, there is a need to ensure that the contents of the smart packet have not been illegally tampered with. In general, this can be counteracted using some standard data integrity service. However in the case of active networks, this raises some additional issues. Consider again, for instance, the situation where a smart packet travels from node A to B to C. Assume that the smart packet contains some program and some data sections. Several cases can occur.

- In the first case, assume that the packet is executed by node B and some of the data section is modified but the programs section remains the same. Then the packet

moves to node C. From C's point of view, it is important to ensure that the packet that it receives is the same as the one that was sent by B. Hence the need for an integrity service between B and C. C may also wish to know that the program code originally created by A has not been modified by B. If this is required, there is a need for integrity service between A and C, as far as the program code is concerned. With respect to the data section, the original data (created by A) may not be useful to C as there has been a genuine modification due to execution of the program at B. We will make this assumption in this case. Hence in this case, the security architecture needs to provide some integrity service between both B and C as well as some integrity service between A and C, covering parts of the smart packet. However

- Now consider another case where the program code itself is genuinely modified by B. If this happens, from C's point of view, it may be sufficient to know that the packet is integrity protected between B and C, given that the original program and the data created by A is not legally valid for C anymore. However it is important for C to know that what it is receiving is only integrity protected with respect to B and not with respect to A.

- Consider the case where C is interested in identifying the part of the data that has originally come from A. Given that execution of the packet could lead to modification of the data section, if this is required, then the data needs to be appended and not overwritten and a separate integrity protection will be required.

Hence we can see that the security architecture should be able to support several options in the provision of integrity service in active networks.

The next security threat is concerned with the actions that can occur as a result of the execution of the smart packet in a node. For instance, there will be a need to ensure that the smart packet execution does not abuse or misuse resources at the node. For instance, smart packet may not be allowed to make general system calls or file system access calls. As in the case of authentication and integrity, different options are possible as to what type of access control service is to be provided. For instance, access control can be provided based on the identity of the creator of the smart packet and/or on the identity of the sender of the smart packet and/or on the identities on the path taken by the smart packet in reaching the node. This requires an access policy to be maintained to determine which packet (e.g. which sender and/or creator) can access what in the system. Such an access policy can be maintained at the node. If a finer level of granularity of access control is needed, then it would require authorisation decisions based on the privileges associated with the program code to access a certain service at the node. To achieve this, it is necessary to specify and store "somewhere" the privileges associated with different programs. In general, two options exist namely the "push" approach or the "pull" approach. In the push approach, the privileges needed to access the service are carried by the smart packet itself. In this case, these privileges need to be protected for integrity and authenticity in the usual manner; this is in many ways similar to passing the authenticated identity information though at a finer level of granularity. In the case of the pull approach, the privileges associated with

the programs are stored say in the node as part of the access policy. The main issues associated with this approach are the management of these finer granularity privileges and their dynamic nature.

In some cases, there may be a need to protect the contents of the smart packet against unauthorised eavesdropping. In this case, some form of confidentiality mechanism is required to counteract this threat.

There will also be a need to maintain at the node an audit log of key security relevant events that have occured as a result of execution of the smart packet. Such an audit log will need to maintain the identity information associated with the smart packet and the action that was carried out. The node itself requires an audit policy which specifies what security relevant events need to be audited and what information are to be recorded in audit logs.

The denial of service threat is aggravated in the case of active networks. For instance, attacks can now be made against a variety of resources such as the CPU cycles, output link bandwidth and storage since these are exposed either wholly or in part to the smart packets.

4 Security Infrastructure and Mechanisms

From above, we can see that a comprehensive security architecture for active networks would need to support at least the following security services:

- authentication mechanisms to determine the identity of the sender and creator active nodes
- integrity mechanisms to protect the smart packets against illegal modification
- confidentiality mechanisms to protect the smart packets against unauthorized disclosure
- access control mechanisms to determine what actions can be done by the smart packet and what resources can it access and modify
- auditing mechanisms at the node to record the types of events that have occurred and to determine whether they conform to the specified security policy
- mechanisms to detect and contain denial of service attacks, given that in general it is not possible to eliminate them.

In this paper, we will concentrate on authentication, integrity and access control services. In particular, we will discuss authentication and integrity services together as they have several issues that are in common. We will briefly mention how confidentiality can be achieved using well known standard techniques. We will not be addressing auditing and denial of service in any great detail in this paper.

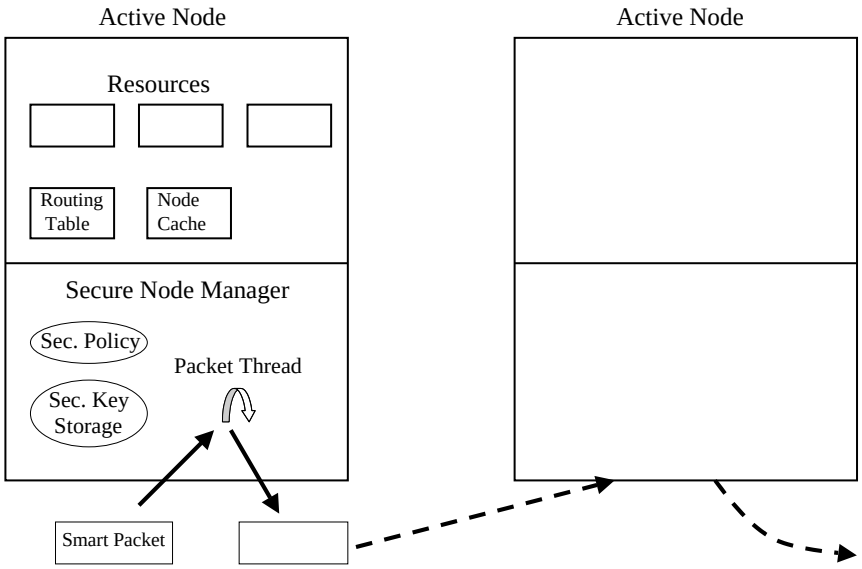

Figure 1: Secure Active Network Environment

4.1 Secure Active Network Environment

Let us first assume that every active node and user in our system has a public key - private key pair. We will also assume that secure storage available in the active nodes if private keys need to be stored. These keys and the corresponding certificates are to be used in the authentication service. At this stage, we will assume that there is a single trusted certificate authority CA which signs the certificates of the active nodes. We will relax this constraint later in our discussion. The public key of the CA is assumed to be known to each of the active nodes in the network. We will assume that an active node has some trusted component that is concerned with the security of the execution environment in the node. We will refer to this logical component as Secure Node Manager (SNM). The SNM also provides a logical place to store and maintain the security policy associated with the node with respect to smart packets. Consider the situation where a smart packet arrives at a node and is run within the execution environment offered by the node. The SNM with its security policy monitors and constrains the behaviour of the smart packet based on the security policy. We will return to the issue of security policy when we discuss access control below.

In describing the security services, we will only consider the situation whereby the program section of the smart packet is not modified as it travels from one active node to another. In general, the program code can also legally change as it moves between active nodes. Though

this is possible and for some applications even useful and desirable, we will not address this case in this paper.

4.2 Authentication and Integrity

The placement of authentication information is a critical issue since it contains not only the "owner" and "origin" of the smart packet but it is also closely linked with the packet's integrity. We propose that each smart packet carries an Authenticator that has the following identifiers as part of its header:

- Smart Packet Identifier : This is just an identifier for the packet.
- Creator Identifier : This indicates the identity of the node where the smart packet has been created. This is in the form of an active node certificate.
- Creator Context : This identifies the principal in the active node which created the smart packet. This may also include other attributes such as the principal's role or the group. The role and group field can be made optional.
- Sender Identifier Set : This contains one or more identities of the sending active nodes. These are in the form of public key certificates.

Consider the situation where a smart packet created by node A moves from B to C. When it arrives at C, the Sender Identifier Set will indicate (A,B). The last sender is node B and the Creator Identifier will show that the creator node is A. Now extending this further, assume that the packet goes to D and then to E. When it arrives at E, the Sending Identifier Set will contain a chain of identities A,B, C and D.

The smart packet also carries the following security tags.

- Sec-Tag-C : This security tag is created by the creator node and it contains a cryptochecksum calculated over the following fields: Program Code (at this stage, we have assumed to be non-changing after creation), Creator Identifier and Creator Context. This can be implemented by hashing the above fields and then signing the hash value using the private key of the creator active node.
- Sec-Tag-S : This security tag is generated by each sending active node including the creator node. It contains a cryptochecksum which is calculated over the following fields: Program Code, Creator Identifier, Creator Context, Sender Identifier Set and Data. This can be implemented by hashing the above fields and then signing the hashed value using the private key of the sender active node.

Let us now consider the authentication and integrity services that can be achieved with the above authentication information in an active network environment.

Consider the situation where a smart packet is created in node A and then moves to B where the program code is executed and some of the data section of the packet is changed and then it travels to node C.

When the smart packet arrives at C from B, the Sender Identifier Set contains the certificate of B. The SNM at active node C uses Sec-Tag-S to determine from where the packet is coming from. This is achieved as follows: the public key of CA is used to validate the sender certificate B and then the public key of the sender B is used to validate the cryptochecksum in Sec-Tag-S which has been signed using the private key of B. The SNM at C is also able to authenticate the creator of the smart packet by verifying the Sec-Tag-C. The Creator Identifier in the packet contains the certificate the creator A. Using the public key of CA, the SNM at C validates the creator certificate and then uses the public key of the creator A to verify the cryptocheksum in Sec-Tag-C which has been signed using the private key of A. These computations enable the receiver of the packet to authenticate the creator and the sender of the smart packet.

If the smart packet has passed through a number of nodes, then the Sender Identifier Set will contain the set of identities through which the packet has travelled. The receiver is able to identify the path by verifying the identities in the Sender Identifier Set.

The cryptochecksums in the security tags also help to verify the integrity of the packet. Recall that the smart packet contains both program and data and at this stage, we are considering the case where the program part is intended to remain unchanged as it travels from one node to another. As the cryptochecksum in Sec-Tag-C is calculated over the program code, by verifying this cryptochecksum, the receiving node is able to determine whether there has been any illegal modification of the program code. This will be true for any receiving node. In fact, by checking the cryptochecksum in Sec-Tag-S, a receiving node will also be able to check whether the previous node which executed the program had the program that originally the creator had sent.

With respect to the data part, note that the cryptochecksum in Sec-Tag-S is calculated over the data part. Hence the verification of this checksum allows the receiver to verify whether the data that it has received as part of the data packet is the same one that the sender (the previous node) had sent. As the data part can legally change as the packet moves from node to node, it is only useful to consider the data integrity service over a single hop and not over the path.

If any of the authentication and integrity checks fails, then the packet is discarded. If the authentication cannot be completed for any reason, then the packet is forwarded. This may occur for several reasons. For instance, if we extend our scenario to include several trusted certification authorities (CAs), then an incomplete authentication situation can occur if the certificate is from a different CA. If all the authentication and integrity checks succeed then the packet proceeds to the access control phase.

4.3 Access Control

After authenticating the packet, the program code in the packet is executed in the active node execution environment. The SNM and its security policy are intended to monitor and constrain the behaviour of the packet within the packet. Access control forms a major part of this security policy. In formulating the access control, it is necessary to consider both

the access control attributes and rules used in the decision making as well as the authorities involved in the decision making and the enforcement of the access decisions.

Let us first consider the access control attributes. In general, there can be two parts to this, namely one part coming from the packet itself and the other coming from the active node SNM system. So far, from the authentication service, the attributes coming from the packet include the creator and sender identifiers and the creator context. These attributes can be used to specify access control rules which determine the privileges and restrictions that the program code should have. Consider first some form of identity based access policies. For instance, we can have access rules which are based on a combination of identities such as the identity of the creator and the context and/or the identity of the sender and/or the identities of the senders in the path taken by the packet. Hence we can have the following access rules:

- A smart packet sp-ID created by active node A can access resources R1,...,Rn in the receiving node.
- A smart packet sp-ID created by active node A with context identifier cx-ID can access resources R1,...,Rn in the receiving node.
- A smart packet sp-ID created by active node A and coming from a sender node B can access resources R1,...,Rn in the receiving node.
- A smart packet sp-ID created by active node A with context identifier cx-ID and coming from a sender B can access resources R1,...,Rn in the receiving node.
- A smart packet sp-ID created by active node A and coming from a sender node B via a node path (A1,...,Ak) can access resources R1,...,Rn in the receiving node.

The resources (R1, ..., Rn) include files, applications and objects in the active node. The cx-ID is the identifier of the principal which is involved with this smart packet. Such access control rules can be specified using Access Control Lists (ACL). They can also identify the limits on resources that are available to the programs in the packet such as the maximum number of instructions that can be executed per invocation.

The access rules can also be formulated in terms of overall service definition. At a simple level, we can have access control rules at the target active node such as a packet coming from creator X and sender Y can access services S1 and S2. Alternatively, we may include a service identifier within the smart packet. This essentially identifies the function of the program code in the packet and is used to convey to the recipient the purpose and intent of the packet. For instance, the service definition could be, say, FTP. So an access control rule at the target could be: a smart packet sp-ID with a service identifier ABC created by active node X and sender Y is allowed to execute. This field is inserted by the creator and in general can specify a list of services and the number of packets involved in the provision of the service(s). This field is particularly relevant when the program code in the packet is

allowed to change. In this case, it provides the recipient some assurance about the desired service.

The SNM in an active node plays an important role in terms of access decision making and enforcement. The access control rules are stored and managed by the SNM. This requires some form of interface to specify and update polices at the active node. If for a collection of nodes the access rules are to be the same, then the access policy can be specified by a central authority managing these nodes and then distributed to the nodes. For instance, this can occur in a "domain" with a collection of nodes where the access policy over the domain with respect to active nodes is intended to be consistent; in this case an authority such as the Secure Management Authority (SMA) responsible for the domain can specify the policy and then distribute it to the SNMs in the active nodes in the domain. The SNM in the active node enforces the access controls during runtime when a packet is executed at the node and determines whether a request is to be granted.

As a general security policy, one may have the following;

- programs whose authenticated origin which match the access control rules are given the appropriate privileges and accesses;
- programs whose authenticated origin do not match the access control rules can be executed with some standard or default privileges to specified resources;
- programs without authentication information can be discarded or forwarded or run in a limited resource environment.

The last case may be useful and necessary as a smart packet can replicate unauthenticated services available today such as traceroute.

4.4 Confidentiality

In certain cirumstances, confidentiality of smart packets may be necessary. If this is the case, then the packet contents apart from the header information required for routing can be protected using encryption. Given that the packet may be executed at any of the nodes, it should be possible to to decrypt the packet at each node. Hence in this case, some form of link encryption is probably the most suitable scheme, whereby the packet content is encrypted using a temporary session key (generated by the sending node) and the session key itself is encrypted using the public key of the receiving node. However such a link encryption scheme increases the computation needed as it requires encryption and decryption of the packet at each node as it travels from one node to another. We do not envisage, in practice, extensive use of confidentiality service in smart packet based active network applications due to the performance overhead.

4.5 Security Implementation Issues

Let us now consider additional security related header information that is required in the provision of the security services described above. First, for authentication and integrity,

it is necessary to include the identifiers in terms of certificates. The main concern is that as the packet moves from one node to another, the general scheme proposed above requires that a certificate is added in each hop. This will reduce the amount of packet size that is available for program and data sections. In practice, it may be sufficient to include just the creator and the last sender certificates and omit the intermediary ones. This is probably sufficient for a number of applications where the access policy is not going to be based on the path the smart packet has taken but only on the creator and the last sender identities. The other issue with respect to certificates is the certificate of the trusted certificate authority. Once again, in practice, we can assume that within a domain, the certificate of the trusted authority is well known and hence it is not necssary for the smart packet to carry this. In the case of multiple certificate authorities, an approach could involve some form of short identifier to refer to the trusted authority rather than to carry its certificate. It is also necessary for the packet to carry the security tags C and S.

If a confidentiality service is to be provided, additional security information may be necessary depending on the method chosen. If a session key is used to encrypt the content, an encrypted version of the session key under the appropriate public key of the node needs to be added.

From the point of view of access control, it is necessary for the active nodes to have some secure component SNM which can maintain and manage the security policy and the keys as well as provide a runtime environment for secure execution of the smart packet.

In summary, it is not surprising to note that provision of security services requires additional header information, additional processing as far as the packets are concerned and maintenance and management of policy information at the nodes. Additional header information in the packets reduces the bandwidth available for carrying useful programs and data and additional processing at the nodes can have performance implications. However given the fact that the whole philosophy behind smart packets involves carrying of programs to do useful work at different nodes, it cannot be achieved successfully without taking into account the security considerations described above.

5 Active Network Applications Examples

In general, active networks can facilitate the provision of a diversity of services and applications. In this section, we briefly summarise some such applications and highlight some security-related issues. For more details on the applications, the reader is referred to [9, 8].

Application : Internet Stock Exchange

Many people use the Internet to get quotes on stocks and shares. Fast and secure access to up-to-date quotes is critical. In this application, caching is not very useful as the information is dynamic in nature. Furthermore, it is probably inappropriate to store entire Web pages as the granularity of the required objects is small. With an active network, the caching strategy may be customized to cater for different application needs. For instance, an active

protocol specializing in quotes on stocks and shares can cache quotes at network nodes using a per stock name granularity. This enables the requests for popular quotes to hit in the cache no matter what subset or order is requested. Also, the requests can specify a client controlled upto-dateness of the desired quotes, thereby allowing the client to trade response time against the timeliness of quotes. Hence with active networks, the quotes are cached at network nodes as they travel from the server to a client. Subsequent client requests are intercepted at the nodes where the local cache is checked to determine whether the desired quotes are available. If so, the quotes are sent to the client, where they are assembled into a viewable Web page. If not, the request is forwarded to the server.

In this application, we can see that several of the security issues considered above are relevant. For instance, the integrity of the quotes need to be protected during the transfer. It is also necessary to ensure that the quotes actually come from the genuine stock exchange server, which requires authentication.

Application : Internet Auction

Web servers hosting online auctions are becoming popular in the Internet. A server running a live online auction collects and processes client bids for the available items. It also responds to requests for the current price of items. Current implementations of auction servers tend to perform all bid processing at the server. With an active network, it is possible to filter out low bids before they reach the server. This can help the server to achieve higher throughput during periods of heavy load. The nodes need to inform the client of rejected bids. The filtering active nodes can also keep a track of the number of rejected bids at each price and send those to the auction server at the end of the auction.
In this application, once again integrity service is required with respect to the content of the bids. Furthermore, there must be policy at each of the filtering nodes that contain the auction characteristics which allow the active node to carry out proper filtering upon execution of the bid packet. Hence access control mechanisms and policy management are needed at the active nodes. From a security point of view, this application has additional interesting features as the auction server is in fact delegating application specific tasks such as rejecting bids to the active network nodes.

Application : Multicasting

The final example that we mention here is the provision of some of the service elements required by multicast applications. It has been shown in [10] that if some intermediate points perform acknowledgement (ACK) aggregation and buffering, implosion problems can be reduced and protocol throughput can be increased. Furthermore, it is better to do local recovery from intermediate systems than from other end systems that participate in a multicast session. Considering the ACK implosion problem in a multicasting application involving a high number of receivers, active networks can perform fusion of ACKs at active nodes between the receivers and the source. Fusion of ACKs involves sending just one ACK from a given active node toward the source for each n ACKs received. With respect to negative ACKs (NACKs), active networks can perform filtering at active nodes between receivers and the source. Filtering consists of the recording in active nodes of the NACK

smart packets already sent toward the sender. They remember the data already requested and when a NACK is received, it is forwarded only if it asks for different data. Once again the integrity and authentication security services described above are important for these smart packets.

6 Concluding Remarks

Security is an important concern in the design and application of active networks. In this paper, we have considered some of the issues involved in the design of security services for active networks. In particular, we have looked at authentication, integrity and access control services in a smart packet based active network architecture. We have discussed the different types of authentication and access control mechanisms that need to be supported in a secure active network environment and the types of security header information needed in the provision of these services. Finally, we have briefly mentioned a sample of applications where active networks can be potentially useful and where security issues are significant. At present, we are developing the Active Network Header fields for IPv4 and IPv6 and considering the bandwidth and performance implications. In practice, trade offs need to be made between the bandwith and performance on the one hand and the extent of security mechanisms required on the other hand; however security needs to be an integral part of active networks design.

References

[1] D.Tennenhouse, J.Smith, W.D.Sincoskie, D.Wetherall, G.Minden, "A Survey of Active Network Research", IEEE Communication, Jan.1997, pp80-85

[2] D.Wetherall, J.Guttag and D.Tennenhouse "ANTS : A Toolkit for Building and Dynamically Deploying Network Protocols", OPENARCH'98, 1998.

[3] D.S.Alexander et al., "SwitchWare Active Network Architecture", IEEE Network, Vol.12, No.3, May/June 1998, pp29-37

[4] D.S.Alexander et al., "A Secure Active Network Environment Architecture: realization in SwitchWare", IEEE Network Vol.12, No.3, May/June 1998, pp37-46.

[5] D.S.Alexander et al., "Active Network Encapsulation Protocol (ANEP)", http://www.cis.upenn.edu/switchware/ANEP/docs/ANEP.txt, July 1997.

[6] B.Schwartz et al., "Smart PAckets for Active Networks", BBN Technologies Internal Report, 1998.

[7] Y.Yemini and S.Da Silva, "Towards Programmable Networks", Proc. of IFIP/IEEE DSOM Workshop, Oct.1996.

[8] M.Calderon et al., "Active Network Support for Multicast Applications", Vol.12, No.3, May/June 1998, pp46-53

[9] eBay Inc. AuctionWeb server; http://www.ebay.com/

[10] A.Azcorra et al, "A strategy for comparing reliable multicast protocols applied to RMNP and CTES", Proc. IEEE International Conference on Protocols for Mutimedia Systems, Mltimedia Networking'97, 1997.