AFRL-IF-RS-TR-2002-313
Final Technical Report
December 2002

PRIVACY ANALYSIS OF THE INTERNET PROTOCOL

BBN Technologies

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. ARPS

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2002-313 has been reviewed and is approved for publication.

APPROVED: Bradley J Harnish

BRADLEY J. HARNISH
Project Engineer

FOR THE DIRECTOR:

WARREN H. DEBANY, Technical Advisor
Information Grid Division
Information Directorate

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 074-0188*

1. AGENCY USE ONLY (Leave blank)	**2. REPORT DATE** DECEMBER 2002	**3. REPORT TYPE AND DATES COVERED** Final Jun 01 – Aug 02

4. TITLE AND SUBTITLE PRIVACY ANALYSIS OF THE INTERNET PROTOCOL	**5. FUNDING NUMBERS** C - F33615-01-C-1974 PE - 69199F PR - ARPS TA - NZ WU - ON
6. AUTHOR(S) William Quentrille, Ronald Watro, Charles Lynn, Jennifer Mulligan, Tushar Saxena and John K. Zao	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) BBN Technologies 10 Moulton Street Cambridge Massachusetts 02138	**8. PERFORMING ORGANIZATION REPORT NUMBER** N/A
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Defense Advanced Research Projects Agency AFRL/IFGA 3701 North Fairfax Drive 525 Brooks Road Arlington Virginia 22203-1714 Rome New York 13441-4514	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER** AFRL-IF-RS-TR-2002-313

11. SUPPLEMENTARY NOTES

AFRL Project Engineer: Bradley J. Harnish/IFGA/(315) 330-1884/ Bradley.Harnish@rl.af.mil

12a. DISTRIBUTION / AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.	**12b. DISTRIBUTION CODE**

13. ABSTRACT ***(Maximum 200 Words)***

This document is the final report for Privacy Analysis of the Internet Protocol (IP), a BBN Technologies project in the DARPA Fault Tolerant Networking (FTN) program. The project was funded through the Advanced Technology for Information Assurance and Survivability (ATIAS) program. This project was proposed under ATIAS Focused Research Topic #7, which specifically addressed privacy protection for network traffic at the IP level.

The project produced three specific results:

1. Design of a Privatized Datagram Service (PDS) - A traffic protection concept called PDS is described along with the motivating attack model and cost concerns. A realization of PDS in IPv4 using IPsec tunnels is described. The possible privacy implications of IPv6 and potential follow-on IPs are also discussed.
2. Simulation of a PDS Example - A simulation of a sample network using PDS in IPv4 was developed in ns-2, a network simulation tool. The simulation demonstrates that PDS provides traffic protection against recently developed attacks. The simulation includes a new class of ns-2 objects that can support future experimentation with the PDS concept.
3. Proposed IPsec ESP Revisions - The IPsec protocol is being revised by the Internet Engineering Task Force (IETF). This project interacted with the team working on the Encapsulating Security Payload (ESP) specification to develop ESP changes that support TFC.

14. SUBJECT TERMS Privatized Datagram Service, PDS, Traffic Flow Analysis Internet Protocol, IPV4, IPv6, IPsec, Encapsulating Security Payload, ESP, Traffic Flow Confidentiality, TFC, network simulation, ns-2.	**15. NUMBER OF PAGES** 57
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Privacy Analysis of the Internet Protocol (IP)

1. INTRODUCTION AND OVERVIEW

This document is the final report for ***Privacy Analysis of the Internet Protocol (IP)***, a BBN Technologies project in the DARPA Fault Tolerant Networking (FTN) program. The project was funded through the Air Force Research Laboratory (AFRL) Advanced Technology for Information Assurance and Survivability (ATIAS) program. This project was proposed under ATIAS Focused Research Topic (FRT) #7, which specially addressed privacy protection for network traffic at the IP level.

1.1 Statement of the Problem

The motivating problem for this research is the situation in which a group of US military facilities exchange encrypted communication over public networks. The traffic flow on such networks can be a source of covert information flow to eavesdroppers that monitor the public networks. Even simple measurements of the quantity of traffic arriving at a remote location can be indicative of imminent activity at that site. More sophisticated traffic analysis has been used to compromise passwords in the secure shell protocol [Son01]. The potential information leakage from network traffic is similar to the covert information channels between security levels in a multilevel operating system.

The simplest solution to this problem is to deploy a fully secure, private network, thus preventing any illicit access to the network traffic. This solution is indeed used for highly classified communications, but it is too expense to deploy for the large volume of sensitive but unclassified information that directs most DoD activity today.

This project focused on designing a standards-based approach to providing ***traffic flow confidentiality*** (TFC). TFC is the protection of traffic flow patterns against adversary analysis that identifies communicating parties and draws inferences about the communication based on publicly assessable traffic characteristics.

1.2 Significance of the Problem

Adversary analysis of traffic patterns is one of several communication security issues that confront the DoD today. Among these issues, maintaining availability of communication is clearly a current priority. It is important that solutions to the traffic protection problem do not worsen other problems. For example, some traffic protection schemes route traffic through a fixed series of intermediate hosts, adding to the denial of service problem since the failure of any single host impacts traffic for many hosts.

Research in traffic protection has been pursued in the commercial sector due its connection to providing anonymity. Traffic protection is a necessary part of any complete anonymity solution. For our motivating problem, where DoD enclaves communicate with each other, anonymity is not of interest. Instead, we expect that clients and servers in our traffic-protected network will commonly execute strong authenticate protocols as part of their communication.

1.3 Proposed Solution

To provide traffic protection, this report defines the Privatized Datagram Service (PDS) as a means of securing the external traffic between a group of trusted enclaves. Each enclave is expected to have a gateway router that will apply the protection. The PDS concept as described here does not address traffic analysis performed inside a trusted enclave. External traffic is protected using a collection of paced IPsec tunnels between trusted host systems. By paced, we mean that traffic in the tunnels is set at a fixed rate, with dummy traffic inserted as required to maintain the traffic rate. Packet size is also maintained at a fixed level. Efficiency of this approach remains a concern, but will be supported by our topology, routing structure, and recommended changes to the ESP that allow for efficient padding of packets and other improvements.

1.4 Contributions of this Project

The project produced three specific results:

- **Design of a Privatized Datagram Service (PDS) -** A traffic protection concept called PDS is described along with the motivating attack model and cost concerns. A realization of PDS in IPv4 using IPsec tunnels is described. The possible privacy implications of IPv6 and potential follow-on IPs are also discussed.
- **Simulation of a PDS Example -** A simulation of a sample network using PDS in IPv4 was developed in ns-2, a network simulation tool. The simulation demonstrates that PDS provides traffic protection against recently developed attacks. The simulation includes a new class of ns-2 objects that can support future experimentation with the PDS concept.
- **Proposed IPsec ESP Revisions - T**he IPsec protocol is being revised by the Internet Engineering Task Force (IETF). This project interacted with the team working on the Encapsulating Security Payload (ESP) specification to develop ESP changes that support TFC. The simplest of the changes are expected to easily gain IETF approval, as they are easy to implement and provide important utility for traffic protection. Other changes, such as packet archives, will need further investigation into their impact before the IETF can consider adoption.

Another valuable aspect of this project has been its coordination with other network security work at BBN. For example, this project has interacted with the BBN team working on the FastJam project in the Wolfpack program. FastJam applies techniques from signal processing to wireless communication and BBN is applying similar techniques to wired networks [Par02]. These attack projects are a useful source of information. The key issue for this project is to stay aware of the traffic features that the attack projects find exploitable. It is also necessary for us to understand the extent of obscuring or mixing of these features that is required to defeat the analysis. To date, packet timing has been the critical feature exploited by these attacks, and pacing packets by releasing them at constant time intervals is an effective countermeasure.

1.5 Report Organization

There are six sections in this report. Section 2 covers technology background, including discussion of several other projects that consider network traffic privacy and anonymity issues. Section 3 discusses our technical approach. It considers attacker capabilities and defender costs.

It also outlines the fundamental concepts of our PDS system, including the notion of an overlay network with paced traffic behavior. Section 4 covers the PDS approach to overall network design and routing. Section 5 reports on the ns-2 simulation that was built to test the security of the PDS concept. Section 6 covers conclusions and recommendations for future work. Appendix A contains a sample Internet Draft developed by this project. Appendix B contains a summary of the new simulation capability that was added to ns-2 to perform the PDS testing.

2. TECHNOLOGY BACKGROUND

This section reviews several of the past efforts in privacy protection and anonymity.

2.1 Onion Routing

Onion Routing is a traffic protection system developed and prototyped by the Naval Research Laboratory [Syv97, Syv00]. It uses a forwarding-and-mixing approach first proposed for e-mail by David Chaum [Cha81]. The Onion Routing system maintains a set of mixing centers called onion routers. When a packet enters the onion routing network, the initial entry router wraps several layers of encryption around the original packet (hence the name onion routing). The initial router determines a random path through the onion network ending at the appropriate router, and encrypts the packet using the public keys of the onion routers with the last hop router key used first. As the packet travels through the network, each router will unwrap its portion of the packet, mix the packets it holds, and then forward the packet to the next hop maintaining the packet size. Each router only knows the router that precedes it and follows it. In this way, the communication is scrambled, thus providing protection against traffic analysis. A freely available prototype of the Onion Routing system operated for several years on the Internet.

There are several known limitations to the Onion Routing system. The technique of forcing traffic through a specific sequence of routers creates a series of single points of failure. If any one router in the network goes down, then all traffic through that router must be restarted along a new path. Another limitation is that traffic entering and exiting the Onion Network is vulnerable. This threat is reduced by the use of entrance ands exist policies on the amount of traffic that a single router handles. It is also possible to position onion routers at the entry points of enclaves, using the assumption that traffic inside the enclave is protected. The layered encryption of onion routing provides confidentiality protection against a malicious routing node. However, denial of service protection requires identification of the malicious router, and this may be difficult.

The Onion Routing prototype did not control the level of traffic between the mixing centers. An assessment performed by the NRL team showed that traffic analysis could exploit these varying traffic levels. As a simulation experiment, link-level padding was added based on a sine wave over a 24-hour period. The sine wave was chosen to avoid the expense of static padding. Their results showed that the attack under consideration was closed by deployment of the sine-wave padding.

2.2 Crowds

Crowds [Rei98] is a system for providing anonymity for client systems on the World Wide Web. The concept of Crowds is very simple: a group of web client systems band together to act as proxies for each other. The group of systems is called a crowd. When a member of a crowd wishes to access a web server, the member's request is routed through a random number of crowd members, as proxies, before finally reaching the intended host. The random routing of traffic is similar to an Onion Routing approach, but Crowds does not do mixing of traffic at the proxies. The path from original client to server in Crowds is fairly static, selected at random when a host joins the crowd and only changed due to the failures of a system on the path or other significant event. The rate at which the path changes is a delicate parameter, as an attack can be crafted by malicious crowd members that force path recreations and track the results [Wri01].

Crowds provides varying degrees of anonymity against local attacks and faulty or malicious crowd members. Crowds does not attempt to define protection against a group of attackers that monitor traffic at numerous points in the network. Such an attack group could potentially follow the full path of a transaction and thus defeat the anonymity service. The initial release of Crowds suffers from several limitations (pointed out in [Rei98]), such as incompatibility with both SSL and firewalls and also susceptibility to denial of service attacks.

Crowds provides a useful example of protection technology. It isolates the concept of employing a sequence of intermediate hosts and provides an analysis of the protection that this provides. This concept can then be combined with additional mechanisms to provided better protection.

2.3 Tarzan

Tarzan [Fre02] is a newly announced peer-to-peer anonymizing network service that appears to provide the best current version of a mixing-based system. Tarzan improves over Crowds by providing some protection against global traffic analysis and improves on Onion Routing by using dynamically selected mixing centers from a large group of peer nodes. Tarzan includes a provision for generating cover traffic to help deflect the global analysis threat.

As Tarzan is just announced, there has not been much time to analyze its effectiveness. Initial discussions have focused on situations where large numbers of peer nodes are malicious. This can easily occur as a single malicious user can form numerous virtual nodes. Under these conditions, there are claims that the anonymity service fails. In any event, the Tarzan design seems to provide the best current privacy protection scheme from a mixing approach.

2.4 PipeNet

PipeNet [Dai98] is an unpublished abstract model that employs a form of virtual link encryption. It is generally considered unimplementable in its full detail but it is a useful benchmark for designing more practical solutions. True link encryption reduces all traffic between two neighboring nodes to a stream of bits. Traffic analysis against link-encrypted channels would be very difficult, as there is little header information and traffic timing or quantizing that can be detected in the channels.

In PipeNet terms, a connection is made from a caller through a set of intermediate hosts (called switches) and terminating at a receiver. The connection setup requires that all hosts on the path share a key with the caller. Nodes also must know their position in the path and must share a link key with each of their neighbors. An Onion-like application of encryption is used.

To thwart traffic analysis, PipeNet employs a static traffic pattern. Constant size packets are emitted at a constant rate from the original sender. PipeNet thus uses a combination of mixing and static traffic shaping. This combination is an effective defense but must be enhanced to provide reasonable performance. The PDS concept builds on the PipeNet example.

2.5 NetCamo

NetCamo (for Network Camouflaging) is a traffic protection system from Texas A&M University [Gua00]. The intended application space of the NetCamo project closely matches our own approach. The NetCamo implementation is a modification of the Free S/WAN IPsec gateway. The NetCamo gateways perform the standard functions to support IPsec tunnels, but they also support custom-designed shaping of the size and timing of the tunnel traffic. The size of packets in the tunnel can be set to a fixed length, and the gateways pad or fragment packets to meet the size specification. The timing of traffic can be controlled to match a specified cover pattern. For example, packets can be required to be transmitted at a constant rate. Dummy packets are inserted into connections as necessary to match the selected cover mode.

NetCamo also takes on the additional challenge of providing Quality of Service (QoS) requirements. The system is controlled centrally to determine the route availability and the overall delays. The traffic is routed according to several constraints. The stabilization constraint ensures that new and existing connections can be sent according to their traffic plans without exceeding the (network determined) camouflaged traffic pattern. The conservation constraint ensures that the correct amount of traffic is re-routed at each node; any source or destination node must have equal amounts of incoming and outgoing traffic. The delay constraint ensures that real time requirements are met for delays. Not all of these functions appear to be implemented in the current NetCamo prototype.

An assessment of the effectiveness of NetCamo's privacy protection was done under the auspices of the DARPA FTN experimentation process [The02]. BBN served as the white team for the NetCamo experiment. The testing was limited to a two-gateway configuration. In this configuration, NetCamo was very successful at preventing traffic analysis by the experiment's Red Team. While NetCamo was very successful in the experiment, there remain two important research issues about this approach to traffic protection. First, can the protection be made practical for large groups of communicating gateways? The multiple gateway problem is clearly much harder than the two-gateway problem. The NetCamo papers discuss this issue but no experimental verification has been performed. The second key issue is finding an effective manner for integrating traffic shaping as performed by the NetCamo gateways into an Internet Standard. The current IPsec specification does not easily permit padding in packets and does not efficiently permit the introduction of dummy packets. Our work in this report addresses both of these questions as well as other issues.

2.6 ANON: An IP-layer Anonymizing Infrastructure

ANON is a research project at Harvard University [Kun02]. ANON is motivated by the example of a server that wants to hide its true IP address from its clients and likewise a client that wants to hide its true IP address from a server. ANON requires a group of trusted forwarding hosts that will implement the mixing aspect of the anonymity. It also requires a public initialization server that provides the necessary information to initiate connections with hidden hosts.

The ANON procedures for creating a hidden server are as follows. The server selects a chain of forwarders. The true address of the first forwarder in the chain and an encrypted form of the address of the server are provided to the initialization server. The initialization server provides the forwarder address and the encrypted server address to clients. The first forwarder in the chain can decrypt the server's address and re-encrypt it for transmission to the second forwarder. The final forwarder then sends the packet to the server.

The ANON forwarding network uses address encryption as a content protection scheme, but it must also address traffic protection in order to guarantee anonymity. In this respect, ANON proposes conventional rate-limiting and link padding as anonymity measures on the traffic between forwarders. A unique feature of ANON is that it proposes that traffic levels alternate between a low threshold level and a high threshold level. This approach helps support Denial of Service protection.

2.7 NAT, DYNAT, and other Address Manipulation Schemes

This section discusses techniques that emphasize manipulation of IP addresses as the primary tool for traffic protection. A basic address manipulation capability is provided by Network Address Translation (NAT). NAT is widely used in the current Internet, sometimes to alleviate shortages of IPv4 addresses and sometimes just to simplify address maintenance in protected enclaves. The traffic analysis protection provided by NAT is limited to aggregating connections, meaning that traffic from multiple hosts is multiplexed onto a single address. This protection avoids the simplest attacks but is vulnerability to spectral analysis techniques that can separate the aggregated threads. The protection provided by NAT is only useful if the data in the packets is encrypted, as otherwise the individual connections could be trivially located. If encryption is provided by tunnel model IPsec, then NAT is superfluous, as address aggregation is part of this IPsec mode. Also, the nature of NAT prevents effective use of the host-to-host IPsec transport mode, since the NAT gateway acts as a man-in-the middle attacker. Overall, NAT cannot be considered an effective tool to prevent traffic analysis.

DYNAT [Fin01] is an address-hopping scheme that was developed at BBN Technologies. By address hopping, we mean that the effective IP addresses of a pair of communicating hosts are rapidly changed according to a cryptographic pattern to prevent attack. Other organizations have suggested similar ideas. There is a loose analogy between DYNAT and spread spectrum communication, where the frequency of the radio communication is changed to avoid attack. A detailed comparison of DYNAT with previously completed by BBN. In red team experimentation, DYNAT was shown to dramatically increase the time required to analyze a network [Kew01]. From the viewpoint of general traffic analysis, DYNAT provides some additional protection beyond traditional NAT, but DYNAT brings additional problems, such as potential conflicts with IPsec, Domain Naming Service (DNS), and Address Resolution Protocol (ARP). BBN improved DYNAT interoperability by designing Anonymous Unicast DYNAT (AUD), which employees IP-in-IP encapsulation. Overall, DYNAT does not seem to be the best solution to the traffic protection problem.

While address manipulation as a general tool is only of limited use for traffic protection, it does play a role to support the mixing approach. For example, entry and exit into a mixing system will sometimes employ address manipulation to provide camouflage for these sensitive traffic gateways.

2.8 S-DATM – Secure Dynamic Adaptable Traffic Masking

Secure Dynamic Adaptable Traffic Masking (SDTM) is a technique developed by Brenda Timmerman that uses network sensors and formal specifications to drive a traffic-masking scheme [Tim96, Tim99]. The SDTM research carefully defines the process of traffic masking as a set of states and possible state transitions. Statistics are kept in a limited manner by keeping averages instead of individual data points. At each time interval, the system, based on the current state (queue length, recent input, recent output, etc…), probabilistically chooses to send nothing, a real packet, or a padding packet. The probability of sending data can be varied widely to give different levels of protection, from a completely constant rate to sending data whenever it is available. The protection is controlled by two constants that bound the likelihood of the next output based on the current state. In this way the traffic pattern is bounded within a constant of the desired secure traffic pattern. If sending data would vary the traffic too far from the bounds, it will not be sent in that time interval.

Further into the research, different methods of traffic masking are examined and their efficiencies, delays, and protection strength are assessed in a simulation. The work compares a non-adaptive interval scheme (constant bit rate), an adaptive interval scheme (send when data available), and differently parameterized S-DATM. It compares the schemes for delay, where the non-adaptive scheme produced the worst delay but used the least padding. S-DATM was the best in efficiency for sending, but also used the most padding. Another key measurement is the correlation constant which measures how the input affects the output in terms of throughput. Using this measurement, S-DATM again was the best for producing the least correlated data (other than the inefficient non-adaptive scheme which leaks no data ever). This work shows the effects of the S-DATM and how it compares to other schemes.

The S-DATM concept is interesting and aspects of it would be useful to combine with our PDS approach to improve its efficiency. One concern, however, is that if traffic adaptation is too effective, then it may create enough of a signal for the traffic analysis tools to detect and interpret. Our belief is that traffic adaptation must be done on a non-real-time basis to avoid these attacks.

3. TECHNICAL APPROCHES AND EVALUATION

This section discusses attack and defense capabilities and costs as well as describing the initial framework for our PDS system.

3.1 Attack/Cost Analysis

The design of a traffic protection scheme must balance the extent and effectiveness of the protection provided against the costs of providing the protection. This is a difficult compromise to make and there are a variety of reasonable approaches to pursue. In this section, we first discuss attacker capabilities.

3.1.1 Attacker Capabilities

In network security, there is a hierarchy of attacker capability levels:

- Passive Attack: Observes traffic but does not interfering with it.
- Traffic Manipulation Attack: Packets are observed, delayed, and possibly deleted.
- Replay Attack: Specific packets are captured and subsequently replayed.
- Full Attack: Packets can be arbitrarily generated. For example, a router may be under the complete control of an adversary, able to see all traffic that passes through, misroute the traffic, destroy the traffic, and initiate its own traffic.

In addition to these network capability levels, attackers must be categorized by their privilege or trust level in the system. In our model, an external attacker has access to only the unprotected network segments that form the public network between two communicating enclaves. It is assumed that external attackers do not possess private credentials or other key material that would let them generate encrypted traffic that would appear to come from a trusted source.

Insider attacks are assumed to have access to the internal networks of some protected enclave. On the internal networks, traffic may not be encrypted. Also, an internal attacker may have access to private credentials and may violate the local processing rules of a proposed traffic protection scheme.

3.1.2 Attack Prevention Costs

There are many costs for providing traffic analysis protection on a network, depending on the chosen method of protection. Primarily the costs are in dollars, bandwidth, and latency in addition to costs from schemes assuming a public key infrastructure. Each individual traffic protection scheme introduces different costs and provides different levels of protection relative to their cost.

(A) Equipment Costs

Additional hardware may be required to implement a traffic protection scheme, such as specific routers or mixing centers. Memory may need to be added to machines to ensure adequate buffering space. There may be costs to acquire or license the software that implements the scheme. In this paper, we focus on techniques that would have no software costs.

(B) Administrative Costs

Schemes that impose a burden on system administration personnel, beyond nominal installation, are expensive in the network lifecycle and will be avoided.

(C) Bandwidth

Most traffic analysis protection schemes utilize excess bandwidth in some fashion or other. Bandwidth is negatively impacted when individual packets are padded to achieve uniform or statistically acceptable packet size patterns and when extra IP headers are used to hide information. There is a bandwidth cost when the traffic data stream is padded with extra packets to conform to a packet-timing requirement. Finally, bandwidth is wasted as packets travel along routes that are not the shortest possible, using up more of the network capacity than necessary. Protection from traffic analysis techniques almost assures wasted bandwidth in several capacities, but careful design can control these costs and minimize their impact.

(D) Latency

Latency refers to delays in the communication link. For example, there may be a latency at the start-up of communication while the network connection is negotiated. Secure connections commonly take extra time to negotiate. There may also be general packet latency. In a traffic protection scheme, each packet may be slowed at mixing hosts that decrypt and re-encrypt the data, requiring more processing than a standard packet. Minimizing packet latency is important for time-sensitive transmissions such as videoconferencing. A useful traffic protection scheme will want to add very little latency to the connection to ensure timely delivery.

(E) Fragility

Security protection may increase the fragility of a network connection. For example, a tunnel-mode IPsec connection may fail when one of its gateways fails even if an alternate gateway is available. As a second example, the Onion routing system (see section 2.1) wraps a packet in layers of encryption that correspond to the planned route for the packet. In this case, the failure of any hosts on the planned path can cause the packet to be dropped.

3.1.3 Attack/Cost Model for this Project

The initial attack model for our work is full external attack. As will be discussed in the next sections, this attack model allows our initial design to use iterated tunnels to protected traffic rather than nested tunnels. Iterated or end-to-end tunnels will provide protection against external attack but not against insider attack at the mixing centers. The protection provided by iterated tunnels is adequate for many defense environments and is more flexible and efficient than nested tunnels. By ignoring insider attack at this point, we can also provide the ability for trusted nodes to monitor traffic for authorized purposes. This last capability could provide a "law-enforcement override" to the privacy features of IP.

For a cost model, our focus will be on providing robust connections that minimize administrative overhead. Latency will be increased but this increase will be controllable by adjusting the level of traffic protection. Finally, bandwidth is a major cost in our design, as network traffic will enter and leave systems purely as a camouflaging activity. The exact analysis of costs will require simulation activity that is beyond the scope of this initial design.

3.2 PDS Design Examples

3.2.1 Point-to-Point Traffic Tunnels

In this section we begin the description of the PDS concept through a series of more complex configurations. In Figure 1, the TFC protection provided by a standard IPsec tunnel is illustrated. The source addresses of the packets are hidden by the tunnel encapsulation, and this is clearly a useful piece of protection. On the other hand, the size and timing of packets will be largely unaffected.

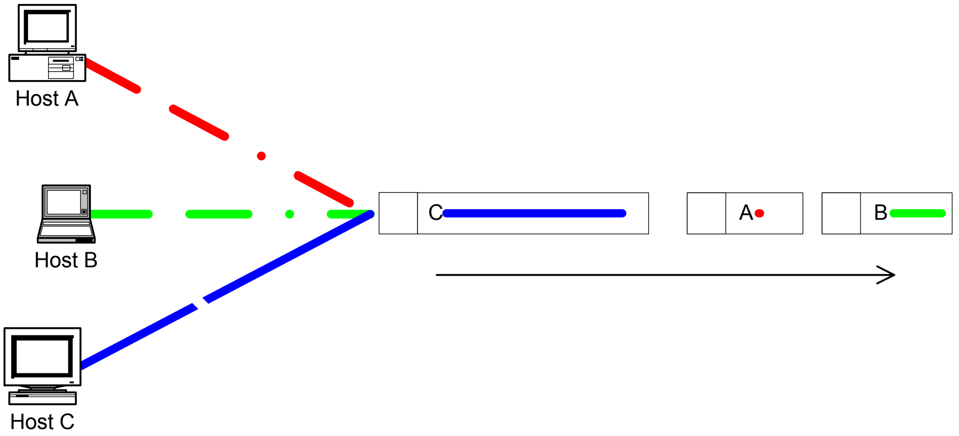

Figure 1 Tunneled Traffic without Padding

3.2.2 Point-to-Point Paced Tunnel

A paced tunnel is an enhancement of a traditional IPsec tunnel. The paced tunnel pads packets to a fixed length and issue packets at a constant rate. These changes block the information that is most commonly exploited by traffic analysis. The NetCamo approach implemented paced tunnels as a point-to-point solution, and the FTN experimentation program found this approach to be effective. Section 5 provides more discussion of how paced tunnels are created and why they provide the desired protection.

3.2.3 Multiple Party Paced Tunnels

The challenge that PDS addresses is how to extend the paced tunnel concept to multiple parties. For small groups, a complete graph of paced traffic tunnels can be used, although efficiency concerns may arise in the configuration of the traffic limits on each of the paced tunnels. For large networks, a topology must be selected and a routing scheme must be deployed. These issues are discussed in section 4.

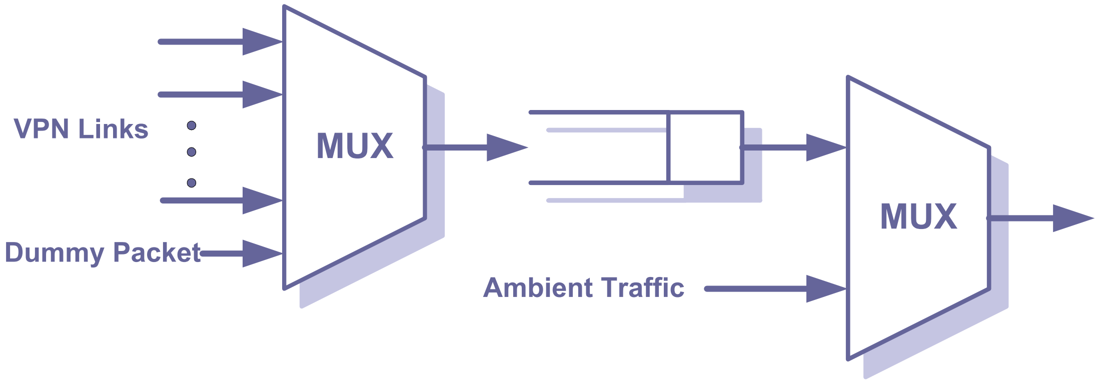

Figure 2 PDS Traffic Multiplexing

Figure 2 shows the multiplexing (MUX) of traffic at a gateway host that operates a group of paced tunnels. Traffic is queued for each VPN link. A traffic pattern is established for the links. A dummy packet is sent any time a VPN link is scheduled to send a packet but has none available. The VPN MUX then feeds into the general traffic MUX, where ambient traffic is mixing in a fixed pattern with VPN traffic.

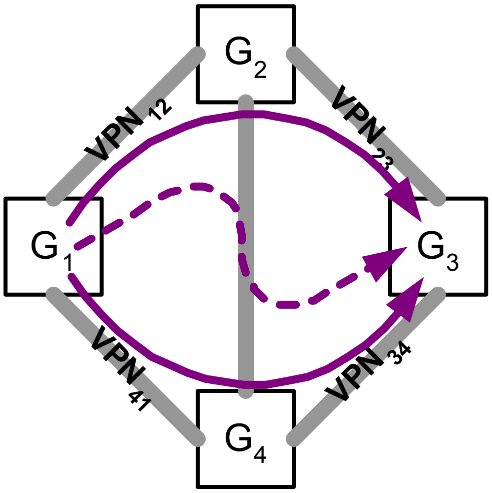

Figure 3 Illustration of PDS Routing Issue

Figure 3 represents the routing problem in a partially connected tunnel network. The dotted arrow is traffic from G1 to G3 and there is no explicit tunnel for this traffic. Section 4 proposes an OSPF and a MLSP option for handling this routing.

4. PROPOSED SOLUTION

4.1 Updates to the Encapsulating Security Payload (ESP)

The IPsec Encapsulating Security Payload (ESP) is currently being revised and a draft version of the ESP v3 specification is available as an IETF Internet Draft [Ken02]. Two important changes to ESP are proposed in the Internet Draft to provide better support for traffic flow confidentiality. These changes are:

- Provision for additional padding in packets
- Provision for a null packet

A short description of each change is given below. It is interesting to note that current tools such as NetCamo that pad packets and use null packets must do so in a non-standard fashion by making custom modifications to public domain IPsec software. The new ESP v3 will finally permit a standards-based solution.

For padding of packets, the previous ESP specification allowed only 255 bytes of padding. This amount of padding was adequate to support block encryption modes and to allow manipulation of data to selected 8-byte or 16-byte boundaries. For TFC purposes, the specification now suggests that unlimited padding be allowed, and that this be implemented in the usual fashion, i.e., by tracking the length of the valid data in the packet.

Adding “dummy” or null packets to a connection is a basic operation in the PDS concept. The ESP v3 draft specification uses the next header field protocol value 59 to designate a "dummy" packet. The draft states that transmitters and receivers must support this dummy next header value and that data in the null packet need not be well formed and is to be ignored.

The above-discussed changes are part of the IETF revision process for ESP v3 and acceptance of these changes seems likely. An IETF Internet Draft (ID) attached as Appendix A to this report describes another possible change to ESP. The ID describes an approach to forming packet archives inside a single packet. This could reduce the wasted bandwidth in a system that uses fixed packet lengths, but there are interesting efficiency issues for this scheme when used in high-speed routers. Further discussion of the packet archive concept will be needed before it can be expected to gain IETF acceptance.

4.2 Virtual Private Network (VPN) Realization of IP Traffic Flow Confidentiality

In the previous sections, we discussed the technique of using a new version (v.3) of IPsec Encapsulating Security Payload protocol (ESP) to realize traffic flow confidentiality (TFC) service between two Internet subnets. In this section, we proceed to discuss the necessary techniques of offering the same service to a non-trivial set of subnets that may be scattered over the global Internet and managed under different Autonomous Systems. We call this service, *Enterprise PDS*, and intend to discuss its realization by (1) specifying its functional/operational requirements, (2) outlining the general approach of establishing a partially connected IPsec virtual private network (VPN), and (3) disclosing the details of techniques such as VPN design/management, traffic routing/dispersion and source packet pacing.

4.2.1 Requirements of Enterprise PDS

As in the point-to-point TFC service, the primary goal of the Enterprise PDS is to prevent adversaries existing *inside* the Internet including both the data forwarding and the network management infrastructure but *outside* the premises of the client subnets from learning the presence/absence as well as the characteristics of Internet communication among the client subnets. Following is a list of the more refined requirements of the service.

1. The Enterprise PDS should obscure the *identity of source(s) and destination(s)* of individual connectionless or connection-oriented network services among the client subnets. This requirement implies that the service should hide IP addresses as well as other identifiers including next protocol identifiers and port numbers of communicating end-points from observers outside the client subnets.

2. The Enterprise PDS should obscure the *identity and characteristics of the applications* that use the network services among the client subnets. This requirement implies that the service should obscure the IP protocol header fields such as next protocol identifiers and port numbers, the upper-layer protocol headers as well as certain properties of the IP packets such as their sizes, transmission intervals and transmission patterns.

   Note that the service is not required to obscure any explicit identity of the applications and their users. Although *user/application anonymity* is deemed important for preventing the adversaries from learning about communications between target entities/sites, such a service should be provided by communication applications and/or middleware with the use of upper-layer protocols.

3. The Enterprise PDS should amortize the *fluctuation of short-term traffic load* among the client subnets. Although the amount of traffic flowing among the subnets results from the aggregation of large number of communication sessions established between different users, applications and hosts, its short-term fluctuation, nevertheless, may reveal sudden increase or decrease of communication activities, which by itself may be a valuable piece of information to the adversaries. Hence, the service is expected to present outside observers with a quasi-static flow of traffic among the client subnets.

   Note that the service can only maintain a quasi-static level of traffic load because it must accommodate gradual variation of data throughput among the subnets as well as occasional but necessary changes of subnet membership and connectivity.

In addition to fulfilling the requirements of protection, the Enterprise PDS must also maintain a satisfactory level of *efficiency* while using the communication resources of the underlying Internet. In particular, it should not unduly decrease the data throughput and/or increase the communication latency or the packet loss rate of any end-to-end communication. These requirements become more important if certain quality-of-service (QoS) guarantees are expected to be upheld among the client subnets. In those cases, the Enterprise PDS should provide a certain level of *QoS transparency* between the service infrastructure and the underlying Internet. The following requirements assert a tight relationship between the effective data throughput and the cumulative communication latency between the TFC service and the underlying Internet.

1. The Enterprise PDS should make efficient use of the communication capacity of the underlying Internet that is allocated to support the service. Since the service is expected to fill the unused capacity with meaningless bits and packets in order to maintain a quasi-static traffic profile, it must minimize the *average amount* of bit/packet filled during the steady provision of the service.
2. The Enterprise PDS should assert tight deterministic upper bounds of the additional delays introduced to end-to-end communication among the client subnets. Since the service employs IPsec VPNs and tunnel-based packet pacing, it must choose VPN topologies with small network diameters and packet-pacing mechanisms with tightly bounded transmission delays.

4.2.2 Operation of Enterprise PDS

In order to serve a set of Internet subnets, the Enterprise PDS establishes a *partially connected virtual private network (VPN)* among the subnets by connecting those using IPsec ESP v.3 tunnels, which offer site-to-site TFC protection. The topology of the VPN and the capacity of individual IPsec tunnels are determined through careful tradeoffs between the efficiency in using the Internet bandwidth reserved for the tunnels and the maximum number of tunnels traversed by VPN traffic, also known as the *diameter* of the VPN. Hiding of traffic source, destination, load and characteristics is accomplished by using TFC protection offered by IPsec ESP v.3 tunnels, and by equalizing the sizes and transmission intervals of the packets traveling within each tunnel. VPN topology, tunnel capacity and packet pacing can all be revised periodically and/or on demand to accommodate changes in subnet membership and traffic profile. The operation cycle of the service can be divided into *four* distinct stages. Their procedures are described in this section, and their technologies are detailed in the subsequent sections.

(A) Formation of PDS Communities

A group of trusted hosts must be established to participate in the PDS. An application level protocol might be useful here to specify the maintenance functions for group membership. The traffic flow design of the PDS community will need to change as membership and usage patterns change. Thus functionality could also be included in the application protocol.

(B) Deployment of TFC VPNs

After a TFC Community is created for the subnets that require TFC protection, a partially connected IPsec VPN can be established over these subnets based on the following information:

- *IPsec Selectors for TFC Protected Traffic* — source/destination IP addresses, protocol identifiers and port numbers of the inter-subnet traffic that requires TFC protection must be specified so that IPsec tunnels can be properly configured to transfer the protected traffic.
- *(Optional) Quality-of-Service Specification of TFC Protected Traffic* — if the expected throughput or maximum latency of protected traffic between some client subnets are known, these QoS requirements must be taken in consideration when the VPN is laid out.
- *IP addresses of IPsec Tunnel End-points* — selected external network interfaces on the edge gateways of the client subnets must function as the end-points of IPsec ESP tunnels. The IP addresses of these interfaces must be specified along with the TFC protected traffic they permit to pass. Note that these interfaces may need to transfer traffic from subnets other than

the ones they belong because the IPsec tunnels do not provide direct connections between all the subnets.

Formally, the above information can be expressed as a set of 3-tuples, each of which specifies the packet selectors, QoS parameters and tunnel end-points of a *stream* of TFC protected traffic. Together, these tuples gives the *service profile* of a TFC Community.

In the trivial case that the PDS Community has only a small number of client subnets, the service can be provided by a fully connected VPN with an IPsec tunnel implementing each 3-tuple. In the more general cases, partially connected IPsec VPNs are deployed. These VPNs are specified by 3-tuple sets, known as the *implementation profile*, similar to those that prescribe the TFC service. However, instead of specifying the expected streams of protected traffic flowing among the client subnets, the tuples specify the actual IPsec tunnels connecting the edge gateways of the subnets. The projection of the service profile onto the implementation profile is a non-trivial task. It must take into account the capacity of the physical Internet and the ambient load of the un-protected traffic.

Once the topology and link capacity of a TFC VPN are obtained by interpreting its implementation profile, the IPsec ESP tunnels that serve as the virtual links of the VPN can be established individually by conducting IKE negotiations [Har98] between pairs of edge gateways. In addition, a management overlay may be imposed onto the VPN to coordinate the episodic changes to the TFC community membership and service profile.

(C) Traffic Routing and Dispersion within a TFC VPN

After the successful establishment of a TFC VPN, we must devise schemes to move TFC protected traffic through the VPN. *Efficient use* of network resources and *transparency* of QoS services were the main concerns when we developed the traffic routing and dispersion schemes. We also used standard Internet routing protocols as far as possible to assure easy acceptance of our schemes.

(1) Packet Forwarding in Fully Connected TFC VPN

In the trivial case that a fully connected VPN is used to serve 2 – 6 client subnets, traffic flows from one subnet to another should be passed through an IPsec ESP tunnel directly connecting the source and destination subnets. IP packets are encapsulated according to ESP v.3 protocol by an IPsec gateway at the edge of the source subnet and decapsulated by a corresponding IPsec gateway at the edge of the destination subnet. The encapsulated packets may be passed between multiple routers (and even multiple autonomous systems) in the underlying Internet, but they need not be decapsulated and re-encapsulated by any other IPsec gateway.

(2) Packet Routing in Partially Connected TFC VPN

In the more general cases that partially connected VPN are established among large sets of client subnets, IPsec gateways at the edges of the subnets should play the role of routers in the TFC VPN. If the source and destination subnets of the packets are directly connected by IPsec tunnels, then packets may be passed between the IPsec gateways located at the edge of the subnets. Otherwise, the packets must be forwarded to some intermediate IPsec gateways located in other client subnets. In the intermediate gateways, the packets are decapsulated, placed into subsequent IPsec tunnels and encapsulated/forwarded again to other IPsec gateways until they reach the edge gateways at their destination subnets and are thus extracted from the VPN traffic. Conversely, the IPsec gateways at the edges of the client subnets must process the outbound traffic in the IPsec tunnels as described. In addition, they must decapsulate the ESP packets coming from the IPsec tunnels and pass them onto network links. If the decapsulated packets are destined for hosts in their client subnets, the packets are forwarded to the links connected to the subnets. Otherwise, the edge gateways must pass the packets to outgoing IPsec tunnels and forward them onto their "next hop". Similar to the routers in the underlying Internet, the IPsec gateways make their packet forwarding decision by consulting the *VPN routing tables* maintained in the gateway by a routing protocol.

We adopted the version 2 of Open Shortest Path First (OSPF v.2) protocol [Moy98] as the routing protocol of TFC VPN. OSPF is a link state routing protocol recommended by Internet Administrative Board (IAB) as the standard Intra-domain routing protocol. We chose to use the protocol not only because of its wide acceptance by the Internet community, but also because of its desirable properties, such as rapid, loop-free convergence of routing tables during network connectivity changes and support of multiple routing metrics as well as management of multiple routing paths. Since OSPF is an intra-domain routing protocol, we must regard the entire TFC VPN as a single routing domain — i.e. treating it as one autonomous system even though its client subnets may belong to multiple autonomous systems in the underlying Internet. This single-domain management of the TFC VPN should not interfere with the multiple-domain management of the client subnets in the underlying Internet because the VPN is treated as an *overlay network* consisting of IPsec gateways as its nodes and IPsec ESP tunnels as its virtual links. Concerning the choice of routing metrics, we recommend the use of *route throughput* and *route delay* (as alternatives to traditional hop count) since the efficiency of network capacity utilization and the upper bounds of packet delays are the important figures of merit with respect to VPN performance.

(3) Flow-based Traffic Dispersion

In addition to packet routing, we also introduced the mechanism of traffic dispersion to the operation of TFC VPN. Its purpose is to enhance the VPN efficiency in using the communication bandwidth of the underlying Internet. Since IPsec ESP v.3 protocol uses payload padding and dummy packet insertion to equalize the sizes of different IP packets, and the inter-packet transmission intervals of different Internet applications, substantial amount of meaningless bits and packets would be added into each IPsec tunnel in order to conceal the fluctuations of actual packet size and traffic load. The added bits and packets amount to a net waste of network bandwidth. As such, their *average rates* of insertion must remain low over long duration of VPN operation.

We proposed to reduce the rate of dummy packet insertion into each IPsec tunnel by both "flattening" the actual traffic load of the tunnel and sending the packets at a matching pace through the Internet links. With this arrangement, dummy packets would only be inserted when the appearance of outbound packets and the controlled transmission of the packets are temporarily out of synchronization and thus caused the egress queue to be empty at the necessary time for packet transmission. We further proposed to achieve the flattening of tunnel traffic load by dispersing *traffic flows* between each subnet pair onto multiple routing paths between the two subnets and then transmitting subsequent flows through different IPsec tunnels. In this context, we define a traffic flow as a one-way stream of TFC protected packets that have the same traffic selectors, source and destination IPsec gateways. We chose to disperse VPN traffic based on short-term flows rather than individual packets because we want to eliminate the effect of out-of-order packet delivery onto TCP session throughput. Assuming the traffic flows in a TFC VPN are random in both their occurrence and duration, by sending different traffic flows between the same subnet pair along different VPN routes, we manage to amortize the site-to-site traffic loads (which may exhibit significant and persistent differences) over different IPsec tunnels and hence different paths along the underlying Internet. Similar to its effects on Internet routing, the multi-path mechanism is expected to lower both level and fluctuation of traffic load of IPsec tunnels [Jea92, Jea93].

(4) Implementation of Traffic Dispersion with OSPF Equal Cost Multi-path vs. MPLS Forwarding

There are likely many acceptable approaches to implement traffic dispersion in a VPN using standard Internet protocols (rather than proprietary technology). We mention here the use of the multi-path routing capability of OSPF v.2 standard and the Multi-Protocol Label Switching (MPLS) technology onto the IPsec VPN overlay.

The OSPF v.2 protocol has two useful capabilities for our overlay dispersion:

1. It is capable of determining the "shortest" route(s) between two end points based on several distance metrics including maximum throughput, cumulative delay and packet loss probability.
2. It is capable of discovering multiple routes between two end points that exhibit the same lowest value of the chosen routing metric. OSPF can maintain some or all of these routes in its routing table and permits the routing engine to distribute the traffic (in equal or different fractions) among these routes.

In order to implement OSPF routing based on multiple metrics, the routers must be able to (1) associate several metric values to every network link, (2) maintain a separate routing table for each metric and (3) forward specific class of packets consistently using a chosen routing table. The routers must also be able to compute metric values according to a natural or artificial definition of equality. Since it is unlikely to have exactly equal values of certain metrics (e.g. cumulative delays) along different routes, one may divide the metric values into ranges (or more generally, equivalent classes) and treat the values in the same range or equivalent class as being equal to one another.

Multi-Protocol Label Switching (MPLS) is an emerging technology that enables data packets to be forwarded in link and/or network layers based on pre-assigned sequences of packet labels. Traffic can be programmed to pass along specific paths by distributing relevant labels to the switching elements. Also, QoS constraints can be satisfied by preserving networking resources for specific traffic flows along the labeled paths. MPLS can be used in a TFC VPN to implement flow-based traffic dispersion by labeling multiple paths that can be used to forward TFC protected packets between each pair of client subnets. Furthermore, bandwidth and packet transmission pace along each IPsec tunnel can be programmed using resource reservation protocols. An MPLS overlay must then be used to forward traffic through the IPsec tunnels. One drawback is that MPLS is currently only consistently deployed inside a single vendors network infrastructure.

4.3 Topology Considerations for VPN Links

When large numbers of hosts are to participate in a PDS community, a topology selection will be needed that balances the node degree versus the diameter of the host VPN graph. Table 1 shows graph properties for some common examples.

Table 1 Connectivity Properties of Common Graphs

	Ring	2-dim Torus	Complete Graph
Diameter (D)	N/2	sqrt(N)	1
Node Degree (d)	2	4	N-1
Total Links	N	2N	(N^2-N)/2

The goal of our network topology is to consider a small fixed node degree *d* and then to examine graphs with minimum possible diameter *D*. This problem has been considered at length for designing Asynchronous Transfer Mode (ATM) networks and deciding how virtual paths should be laid out to create optimal virtual circuits. The solution we choose is a combination of research in the area.

The general layout problem for network nodes has been studied by Bermond, Marlin, Peleg and Perennes [Ber99]. They looked at nodes arranged in paths, cycles, a torus, a mesh, an arbitrary tree, and a complete k-ary tree. The most effective pattern with restricted degree is the torus. In two dimensions this is a pattern of nodes as if they were laid out on graph paper with a node at each intersection, with additional connections between the nodes at the bottom of the page an the top, as well as between the leftmost and rightmost nodes (see Figure 5). In this way the pattern has no edges and no inherent “bad” spots. The basic figure gives all nodes a degree of 4, but this could be expanded to consider additional dimensions and all degrees that are powers of two. For the most balanced torus with degree 4, we assume that the number of nodes *n*, is a perfect square and thus that both directions would have sqrt(n) nodes. The diameter of the network is then sqrt(n), the largest distance between two nodes being sqrt(n)/2 in both dimensions. Using a torus, we have satisfied our requirements for a small diameter with a restricted degree on each node.

Toruses of higher dimensional also have useful proprieties. For instance, a 3-dimensional torus would have node degree 6, and have $n^{1/3}$ nodes in each direction. Therefore the diameter of such a network would be at most 3/2 * $n^{1/3}$, traveling half of the length in every dimension. In the *k*th dimension with degree 2**k* the torus would have a diameter of *k*/2 * $n^{1/k}$. The following graph demonstrates the growth rate of diameter as the number of nodes increases for different diameters. The dimension of the torus should be selected according to the number of nodes in the torus to produce roughly the smallest diameter.

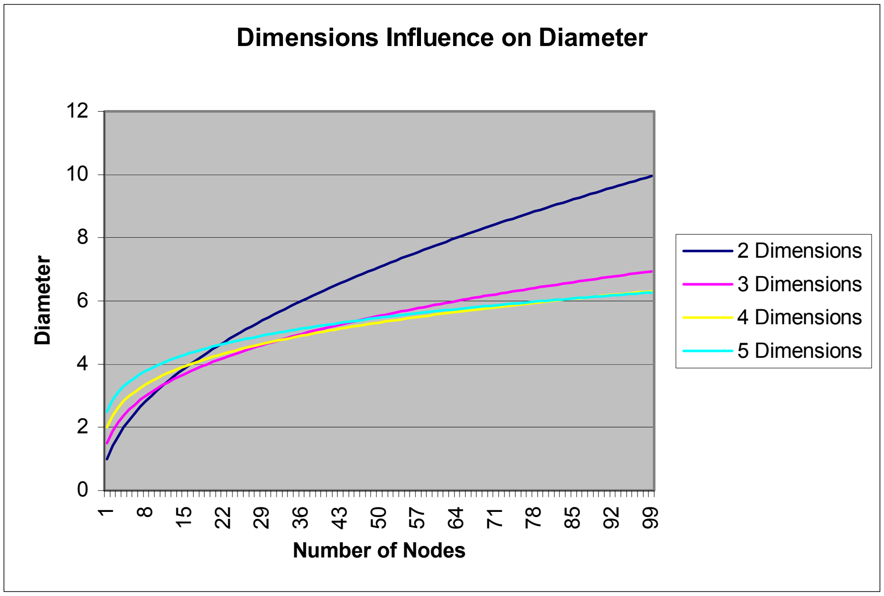

Figure 4 Diameter Graph for Torus Networks

Next we need to allocate the specific hosts in the network to particular spots in the torus. With no further information about the capacities needed on these links, we can do no better than to randomly assign each node to each position. This can be accomplished by randomly permuting the nodes and then filling up each column of the torus in this order. The resulting graph will be a uniformly random 2-dimensional torus with an average path length of sqrt(n)/4 and maximum diameter of sqrt(n).

While this algorithm will work in theory, in reality the number of nodes in a network is not likely to be a perfect square. In this case, to form the torus we need to make some small changes to the above algorithm. First we create dummy nodes as placeholders and pretend as if they were regular nodes in the above algorithm. We can create as many dummy nodes as necessary to get to the next power. Then once the nodes have been distributed into the torus we can remove the dummy nodes by creating new edges. The method through which we will remove the dummy nodes may depend on which requirement is more stringent, a small diameter or a maximum node degree. Two options present themselves as demonstrated by Figure 5. Either we can remove the dummy node and then:

A. connect the nodes on the paths that travel straight through the node (solid circular segments on the inner sphere), or

B. connect each node to the node that is next connected to the dummy node looking clockwise at the outgoing edges (dashed circular segments on the inner sphere).

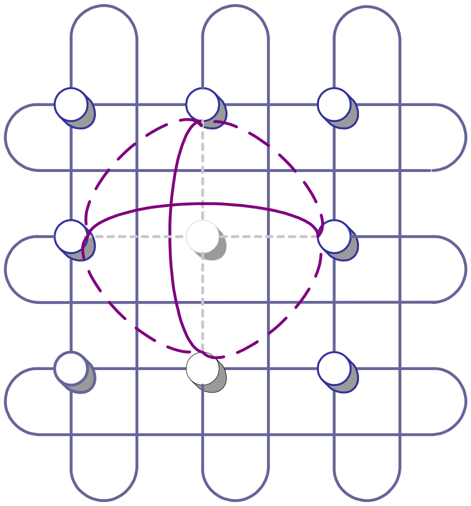

Figure 5 Eliminating a Dummy Node from a Torus

Each reconnection is well defined and can be repeated for any arrangement of dummy nodes to recreate a structure similar to a torus with good properties. In method A, we preserve the restraints of degree of each node, but the diameter may increase as we remove nodes. In method B, the diameter will remain the same, but the degree of each node may increase dramatically. Therefore, depending on the precise goals and restrictions of the network layout, one of the two strategies will produce an efficient layout in an efficient amount of time.

In a real world situation, we expect the network to be growing and changing over time. New members will wish to join, and others will depart. Therefore the network must be flexible enough to handle new members without disrupting the entire preset network already.

One simple idea is to add any new nodes onto the outside of an already constructed full torus. This may lead to a poor distribution of nodes because they are not randomly distributed. Newly joining nodes are likely to be related to one another as an organization may participate with several nodes. The non-random nodes may disrupt the PDS traffic flows if they are situated near nodes with which they often communicate. Therefore we need a method that will more randomly distribute nodes and not leak as much information.

It seems reasonable to prepare a graph for additions by ensuring that there are enough holes in the torus to accommodate expected growth. When a new node desires to join the network, it can take the place of a random dummy node and make new connections to the nodes that are adjacent to it along the axes of the torus. These nodes are more randomly distributed in the network and will only affect at most four other nodes. When the torus becomes full as all dummy nodes have been replaced with new nodes and another new node looks to join, the entire set of nodes can be redistributed in the initial fashion. Thus, we again have a “non-full” torus that can be filled in with any joining nodes. The rare-time cost of relaying out the entire network should be fairly minimal, especially as the network grows larger, as it will occur with at least quadratically decreasing frequency.

4.4 Privacy Protection and IPv6

This section covers traffic analysis issues in IP version 6 (IPv6). Currently the transition from IP v4 to IP v6 appears to be stalled and there is some risk that large-scale deployment of IPv6 may not occur. In any event, it is useful to consider whether IPv6 offers any benefits or pitfalls to privacy issues.

The most obvious feature of IPv6 is the large size of the address space. This has advantages and disadvantages for privacy protection. The primary advantage is the extended capability to generate new addressees for the same interface, thus making traffic flow to a single interface more difficult to correlate. This advantage is more valuable to traffic mixing schemes than to the PDS concept, where we create the illusion of a fixed traffic pattern, at least in the overlay network. A possible disadvantage of the large address space occurs in the stateless address autoconfiguration process, and this is discussed below (section 4.4.1).

The IPv6 feature that may best support the PDS concept is "anycast" addressing. An anycast address represents a group of hosts with the provision that a packet sent to an anycast address is to be delivered to one of the address's hosts. Further discussion of PDS using anycast addressing is include below (section 4.4.2).

4.4.1 Privacy in IPv6 Address Autoconfiguration

IPv6 network addresses are 128 bits long and support unicast, multicast, and anycast address types. Each IPv6 unicast address contains a 64-bit interface identifier that specifies the unique local interface associated with the address. The other 64 bits in a unicast address contain flags of various sorts and the network identifier for the interface. While IPv6 interface identifiers are only required to be locally unique (on the local link segment), it has been suggested in the IPv6 addressing architecture [Hin98] that globally unique identifiers (such as Ethernet MAC addresses) be used. Indeed, the fact that such a large number of potential interface identifiers is supported by the IPv6 design shows that globally unique identifiers can be used. It is proposed that a singly homed host can autoconfigure an IPv6 address by concatenating the network identifier for its current network with its unique host identifier. A host that is newly attached to a network can transmit a local message requesting the network prefix for the attached segment. Once the host receives a reply from a local router, the host can create its IPv6 address. This process is referred to in IPv6 specifications as stateless address autoconfiguration [Tho98]. This approach to network address construction was widely deployed by Novell in NetWare networks using the old IPX protocol. The IPv6 details are of course different but the concepts are the same. Stateless address autoconfiguration eliminates much of the need for the Dynamic Host Configuration protocol (DHCP).

Using a long-lived global identifier such as an interface Ethernet address in the IP address is a concern from the privacy perspective. The problem is primarily for mobile hosts that move from network to network. With the Ethernet address appearing in the IP address, the location of a mobile host would be identifiable by any entity that can view the network packets. To counter this problem, a privacy extension of IPv6 was proposed in RFC 3041 [Nar01]. This RFC was called out in the references to the FRT 7 request for proposals. In RFC 3041, the link-identifier of a client is referred to as the private identifier for the interface. A process is defined for generating a series of public local interface identifiers by applying a hash function to a combination of the private interface address value and a history value. The privacy extension also discusses the logistics of managing the public address of the interface. There are several obstacles to the deployment of such an address-changing scheme, such as potential conflicts with routing, network debugging activity, and DNS-based authentication schemes.

From the perspective of the BBN research into privacy protection, the RFC 3041 suggestions appear to be short sighted. First, a much simpler solution would be for a client with privacy concerns to simply choose a random local identifier for autoconfiguration. The Ethernet address or any other unique local information could be used to seed the local random selection. Duplicate address detection must be performed if random address selection is used, but this is always required even when Ethernet addresses are used. Furthermore, the RFC 3041 suggestions only attempt to hide local addresses, and this functionality is part of the minimal privacy protection that is applied whenever a tunnel-mode gateway is employed. Thus, the RFC 3041 changes would generate little benefit in a situation where minimal privacy protection steps are in place.

4.4.2 Supporting PDS in IPv6

The IPv6 anycast addressing appears to provide a useful tool for supporting both PDS and mixing schemes for privacy protecting. Many of the privacy protection approaches involve a group of hosts that are sharing or proxying traffic for each other. These groups could be represented by a number of distinct anycast addresses. Each anycast address would represent the communicating group of hosts and some additional state information about the traffic, such as the number of mixing rounds previously applied to the traffic. Storing the state information in the address will allow better control of mixing behavior, for example, as compared to randomly selecting a next hop.

It should be noted that the applicability of anycast messaging may be limited by several factors, for example, the interaction of anycast addressing with IP-level encryption. In the PDS example, we would want an anycast capability that could be secured using IPsec. The details of how this will be supported remain to be fully defined.

4.5 Privacy Protection and a New IP

This section takes a brief look at future prospects for traffic protection. The first subsection below considers conventional networking technology beyond IPv6 while the second part considers advanced networking technology, such as quantum-base networks. Overall, near-term prospects for Internet-wide deployment of new IP technology are small, as is illustrated by the delayed acceptance of IPv6. For this reason, the project placed little emphasis on the area of developing an entirely new approach to IP. In the long term, however, there are exciting prospects for solutions to the traffic protection problem based on new IP technology.

4.5.1 Conventional Network Technology

On of the limitations of the current TCP/IP suite for traffic protection is the limited tolerance that current TCP provides for out-of-order packets from the IP level. Traditional TCP includes only a very simple acknowledgement mechanism that operates by tracking the sequence number of the next expected packet. If packets are vigorously mixed at the IP level, current TCP can be expected to hang and drop connections. Several groups have investigated developing a TCP with a more robust acknowledgement service. The mixing approach to traffic protection can make good use of this change to TCP. For PDS, the packets should remain largely in the same order, but a longer timeout may be useful to maintain connections through changes in the overlay network and its routing.

In addition to network protocol changes, there is hope that advances in network hardware, such as Asynchronous Transfer Mode (ATM), will benefit privacy protection. The key to these advanced technologies would be to eliminate the visible IP packet boundaries in large portions of the network. Without the IP packet boundaries, traffic analysis cannot obtain the timing and sizing data it needs as input. The use of small fixed-sized frames (as in ATM) provides the type of networking functionality that has the potential to hide IP packet boundaries. Of course the actual mapping of IP data into the packets will need to be designed to reflect the privacy goal. The current nature of the Internet backbone suggests that traffic protection can be applied to large network hops that use a single technology and are located within a single providers range.

4.5.2 Advanced Network Technology

New technology based on quantum mechanics shows promise to create networks that are immune from many forms of traffic analysis. The transmission of data in quantum form involves the use of polarized photons. This technology is attractive for a number of reasons, including the currant abundance of preexisting dark fiber that can provide the networking media. The famous uncertainty principle of quantum mechanic has a superficial positive application to privacy protection, but the full story of quantum encryption and quantum networking is very complex. Suffice it to say that should quantum networks be realized, the issue of privacy protection would take on an entirely new form that cannot be predicted today.

Active Networks is a technology that we have listed as advanced (implying futuristic) but might easily be viewed as straddling the conventional and advanced boundary. The concept of active networks is packets that carry a form of internal executable context. Some forms of active networking technology can be deployed within the framework of the current Internet infrastructure. For traffic protection, there are some interesting potential ways to exploit active networks. Active packets might be used to spawn other packets to complicate the task of tracking packet flow. An active packet might provide a control mechanism for the management of a mixing scheme, or for the traffic level management of a fixed traffic scheme.

5. RESULTS AND DISCUSSION

This section examines the simulation testing of the security properties of the PDS design.

5.1 Simulation of a Sample PDS Network

As described in the previous sections, PDS uses paced network tunnels in VPNs in order to hide key flow characteristics and prevent flow identification by snooping agents on the core network. Our proposal requires implementation of pacing only at the routers on customer premises, thus eliminating the need for software changes at routers owned by service providers.

Two key topics are discussed below:

1. *Flow Detection in VPNs*: Signal processing can be used to detect flows and gain useful information about communication patterns. We describe the ideas behind these signal processing techniques and the implementation of the attacks in the simulation.
2. *Using Paced Tunnels to Hide Flows:* In this part we examine the solution to the problem of flow detection in VPNs by using paced tunnels. We discuss how pacing can be used to alter the time of arrivals in such a fashion so as to prevent a snooper from gathering useful information in the backbone. We also discuss our experiences from the simulation experiments with paced tunnel VPNs. In particular, we describe the following three key observations:

   2.1. With paced tunnel VPNs, the only frequencies discovered on the backbone links correspond to the pacing frequencies of all the tunnels sharing that link.

   2.2. Pacing also conceals increased communication activity during high-volume events (e.g. beginning of a war). Our proposed pacing scheme ensures that during high-volume events, the frequency response remains unchanged, thus preventing discovery of the event.

   2.3. Pacing must use dummy packets; otherwise one can reveal as much information as in the unpaced scenario.

5.2 ns-2 Network Simulation Tool and BBN Extension

Our simulation work was performed using ns-2, a publicly available discrete event simulator for networking research [McC]. ns-2 is an object-oriented simulator, written in C++, with an OTcl interpreter as a front end. It contains built-in support for a variety of protocols including TCP/IP and many routing protocols. The DARPA Virtual InterNetwork Testbed (VINT) project provided past support for the development of ns-2. ns-2 is currently maintained jointly by UC Berkeley, LBL, and ISI.

In order to develop the experiments for paced data tunnels in ns-2, a new class called *Frt7Pacer* was added to ns-2. This class allows one to build VPNs with paced tunnels and with a variety of useful tunnel configurations. Frt7Pacer is derived from the Connector class and can be installed in each node. Frt7Pacer allows the data at egress interfaces to be aligned at boundaries of periodic intervals. Free intervals may optionally be filled with dummy packets. Summary and installation information is included in Appendix B.

The FRT7Pacer class was developed for internal BBN use in order to test our PDS design. The code can be made available to other researchers (assuming government approval) but there is no end-user documentation.

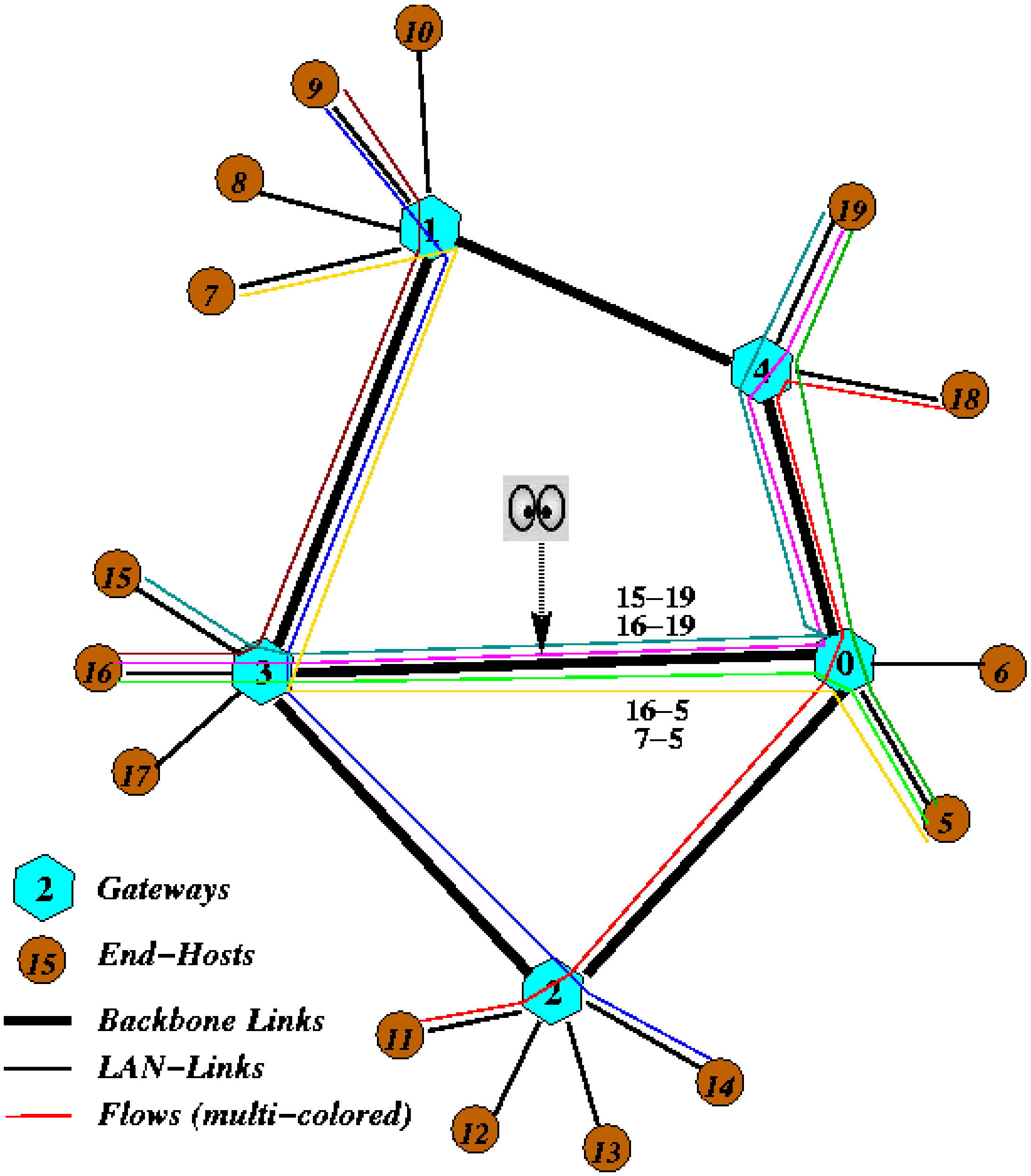

Figure 6 Simulation Sample Topology

The simulation network that was used in this report for experimentation. It has 5 core gateways, each with between 2 and 4 end-hosts. Eight (8) flows were set up between various end-host pairs – 7 FTP flows and 1 CBR/UDP flow. A snooping tap was placed at the midpoint of all core links. This figure shows the snooping tap on the 0-3 link. The tap can only record the times at which packets pass on the link – the tap is not allowed any information about the contents of the packets, or the headers. Four FTP flows (7-5, 16-5, 16-19 and 15-19) share the 0-3 link.

5.3 Flow Detection on VPNs

The tests of PDS were conducted on the test network shown in Figure 6. In this figure, Nodes 0, 1, 2, 3 and 4 are core routers, whereas nodes 5, ...,17 are end-hosts attached to core routers. We set up a combination of FTP and CBR (UDP-based) flows between end-hosts on this system. We also set up snooping taps on the core links (0-2, 0-3, 0-4, 1-3, 1-4, 2-3). In order to restrict the amount and nature of information these taps can access, the only data they were allowed to collect were the timings of the packets on the link. No access was allowed into the actual data inside the packet.

Our aim is to demonstrate examples of the kind of information the taps can infer about the communication patterns in the network, even under such restrictive conditions, wherein the taps cannot read even one byte of the packet.

The *Lomb Periodogram* method was used to analyze the timing traces collected by the taps [Par02]. Lomb periodograms have been shown to be very useful in extracting the underlying flow characteristics from a variety of timing signals. We will demonstrate how we used Lomb Periodograms to analyze VPN traffic and its potential for revealing key flow information, without requiring any information about the packet contents, or even its header.

As was mentioned, six taps were placed on the six core links, and they recorded (i) the times at which packets traversed the link, and (ii) the direction of the packet. For example, the tap on the 0-3 link recorded the times of all packets that traversed the 0-3 link (as they passed near the tap), and the direction in which the packets traveled (0 to 3 or from 3 to 0). Let ***T1*** be the time-series representing the times the packets going from 0 to 3 passed the tap, and let ***T2*** represent the time-series representing the times the packets going from 3 to 0 passed the tap. A signal encoding is constructed from these two time-series by assigning an amplitude of +1 to samples in ***T1***, and -1 to samples in ***T2***. The Lomb periodogram analysis (as described in [Par02]) was carried out on such signal encodings for each of the six taps. The Lomb periodogram for the tap on the 0-3 link is shown in Figure 7. It shows that the prominent frequencies are 214, 58, 233, 92, 116, 184 and 150 Hz.

It turns out that each of these frequencies corresponds to some key timings (such as send-rates, round-trip times etc.) of each of the four flows (5-16 FTP, 16-19 FTP, 15-19 FTP and 5-7 FTP) sharing this link. The relative *spectral powers* of these frequencies give an indication of the amount or volume of communication happening using the corresponding flows.

These patterns of prominent frequencies, when analyzed across all taps, can potentially reveal some information about the end-to-end communications as well. For example, the frequencies 214, 58, 116 and 233 Hz show up on both taps – links 0-3, as well as link 0-4. This suggests the existence of one or more flows connecting 3 to 4, via 0. Indeed, this setup had two FTP flows – one between 15 ↔ 19, and another between 19 ↔ 16, both of which used the 3-0-4 path on the backbone.

Another potentially useful bit of information that can be gathered from the Lomb periodograms is the volume of traffic on a route. For example, after discovering the existence of flows connecting gateway 3 to gateway 4 using the 3-0-4 path, the variation, in time, of the *spectral power* of the frequencies 214, 58, 116 and 233 Hz can reveal the volume of traffic or communication between gateways 3 and 4. A high spectral power implies increased volume, whereas low spectral power of these frequencies would mean diminished communication between 3 and 4.

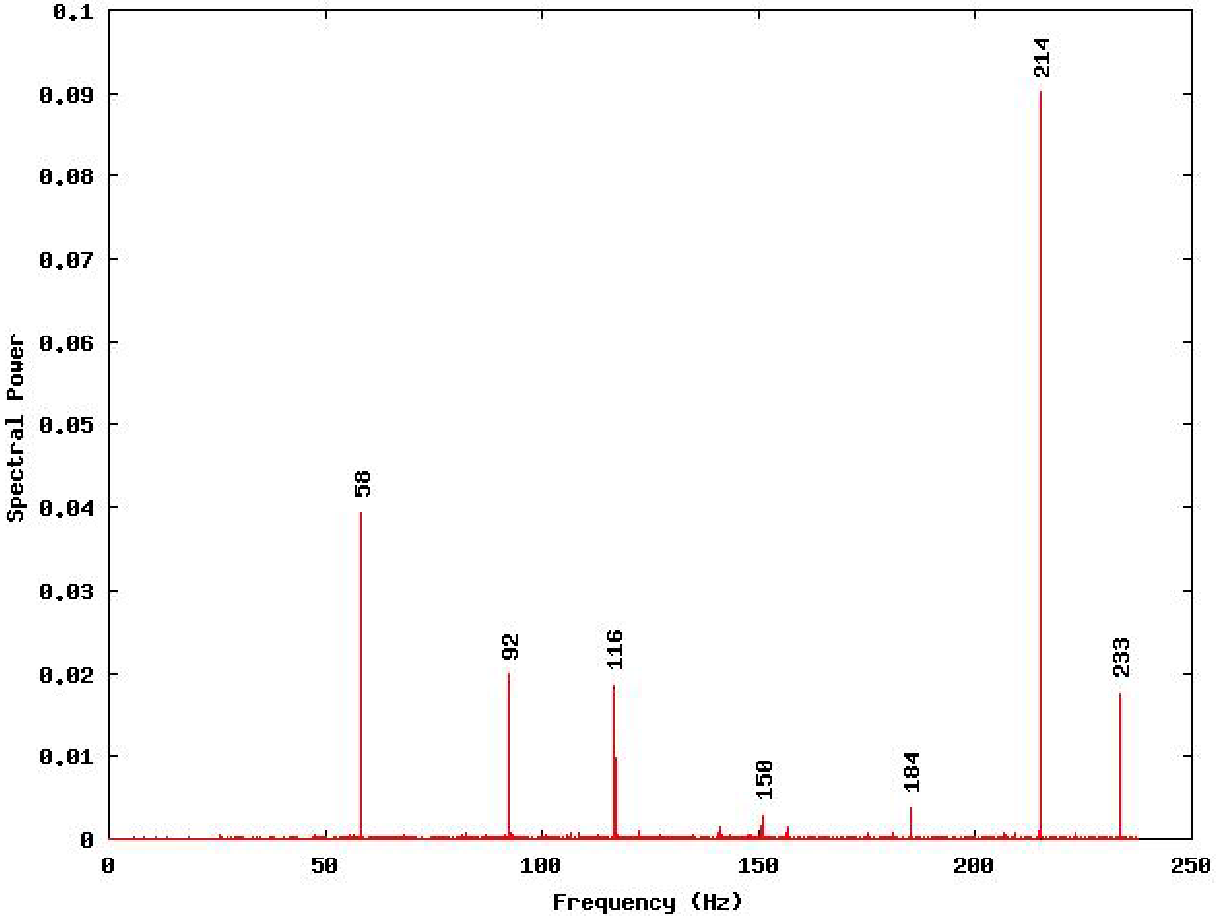

Figure 7 The 0-3 Lomb Results

The Lomb Periodogram of the signal collected by a probe tap on link 0-3. The prominent frequencies are 214, 58, 92, 233 and 116 Hz. Of these, four frequencies (214, 58, 116, 233) also show up on the 0-4 link, suggesting an active route of 3-0-4. Indeed, two of the four active routes using 0-3 link also used the 0-4 link.

Let us say that we are interested in identifying the sites, sharing the link 0-3, that are *"talking"* the most, or have the maximum communication. The fact that among all frequencies on link 0-3, the top 2 (viz. 214 Hz and 58 Hz) are shared with the link 0-4, suggests that the flows between 3 and 4 are contributing the most to the traffic on link 0-3.

In this section, we have demonstrated how even a simple tap, with limited signal capture capabilities can use some smart, sophisticated mathematical tools to discover important, and potentially damaging (to the adversary) information about the network communications and traffic. In the following sections, we will propose a technique that allows the communication parties on the VPN to actively control the signal, and the corresponding frequency response.

This can potentially allow them to *hide* the underlying flow characteristics from such snooping taps.

In order to decrease the effectiveness of time-series analysis of the type presented in the previous section, we used paced tunnels for implementing VPNs. The key idea is to use pacing of traffic at the VPN gateway routers so as to modify the timing characteristics of packets in the core network, and thus control the shape of the frequency response observed by snooping taps on the core-links. Such control can allow us to conceal our communication patterns, increase our security, and substantially decrease the effectiveness of snooping.

We will demonstrate the concept of paced tunnels using our previous example of Figure 6. Let's say we wish to establish a paced VPN tunnel between the subnetworks behind the gateways 3 and 4. We create a tunnel between the gateways 3 and 4 by specifying a pacing frequency. If the pacing frequency is 200 Hz, then gateway 3 establishes a paced tunnel to gateway 4 by essentially creating 200 slots per second, intended solely for gateway 4. We call these slots the slots on the 3→4 tunnel. Then, whenever gateway 3 has to forward any packet from end-hosts 15, 16 and 17, intended for end-hosts 18 and 19, it sends that packet in the earliest available (empty) slot on the 3-4 tunnel. When an empty slot expires (i.e. there was no "real" packet to be sent from 3 to 4 at that time), then gateway 3 sends a *"dummy"* packet to gateway 4. In other words, in this paced tunnel specification, gateway 3 always sends 200 packets per second to gateway 4 – some of these packets are "real" data packets, sent on a first come first serve basis, whereas others, when there is no communication, or real data to send, are "dummy" packets. If this is a bi-directional tunnel, then a similar tunnel is established from 4 to 3 as well.

Note that because of queuing at the intermediate routers (e.g., gateway 0), the receiving end (gateway 4) *may not receive* packets at 200 Hz, or every 5 ms. Also, the links 3-0 and 0-4 will typically be shared by other tunnels (connecting other gateway-pairs) as well. The interaction between these tunnels can cause substantial jitter in the inter-packet interval on the 3-4 tunnel.

Note also that many variations are possible in the tunnel configurations. Paced tunnels can be set up *without using dummy packets* as well. In this case, if a time slot does not have any real packet to send, then *nothing* is sent at all. We call these the *no-dummy paced tunnels.* Also, at least three different pacing frequency schemes are possible: (i) All tunnels use the same pacing frequency. (ii) All tunnels use different frequencies, but bi-directional tunnels use the same frequency in both directions. (iii) All tunnels use different frequencies, and moreover, bi-directional tunnels even use different frequencies in the two directions.

We implemented paced tunnels in the ns-2 simulator (see section 5.2). Our extensions can be used to set up paced tunnels between node-pairs, by specifying the (i) *pacing frequency* (possibly different across tunnels and directions)*, (ii) fraction of non-tunnel traffic* (in each tunnel)*, and (iii) whether the tunnel should send "dummy" packets or be a no-dummy paced tunnel.*

We conducted experiments using four configurations: (i) Uniform pacing frequency in all tunnels – with dummy packets; (ii) Uniform pacing frequency in all tunnels – without dummy packets; (iii) Different pacing frequency in all tunnels, in both directions as well – with dummy packets and (iv) Different pacing frequency in all tunnels, in both directions – without dummy packets.

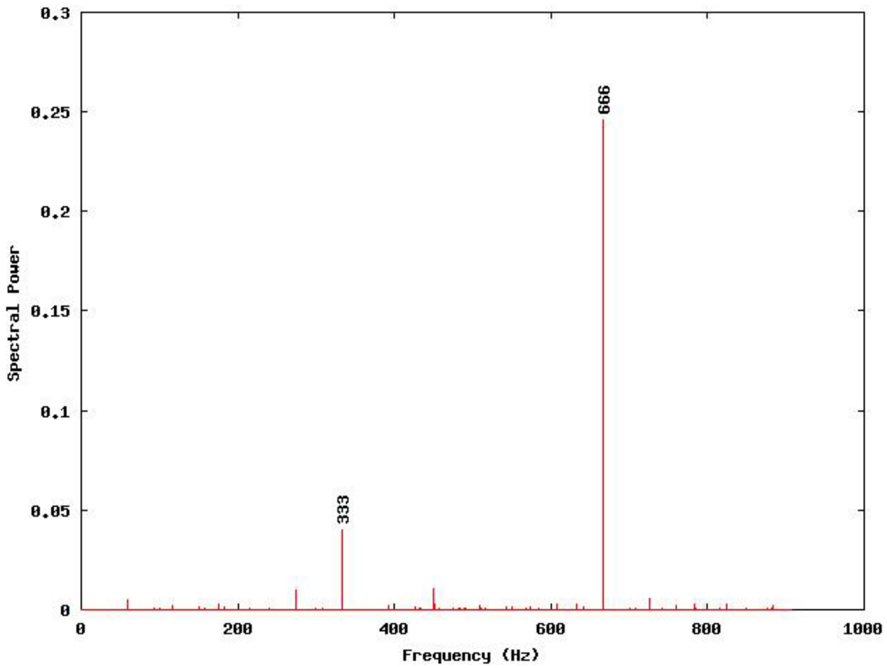

Figure 8 Uniform Pacing with Dummy Packets

Only the (uniform) pacing frequency (333 Hz) and its harmonics (666 Hz) dominate, hiding the characteristic frequencies of underlying flows.

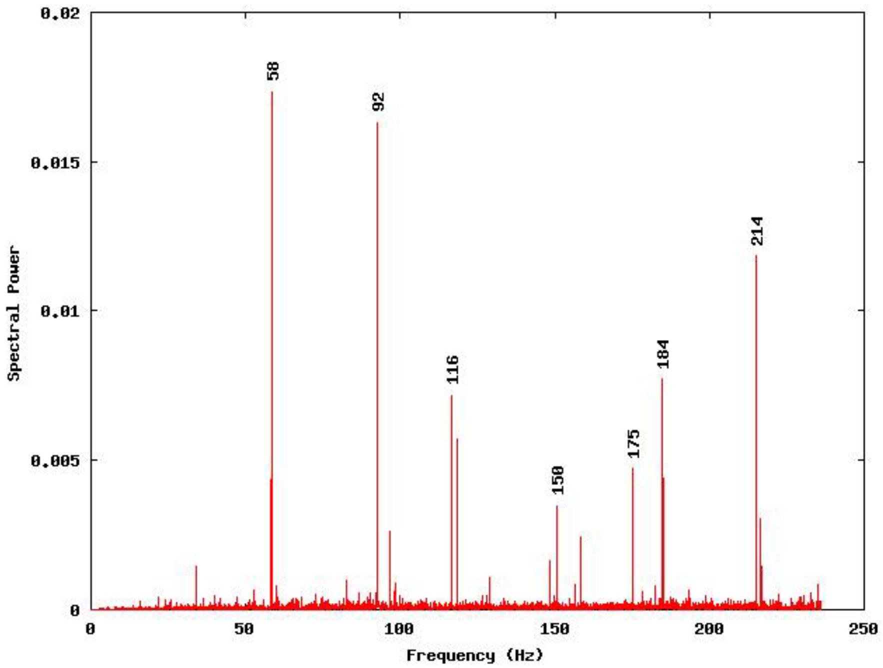

Figure 9 Uniform Pacing without Dummy Packets

Same as the Figure 8 case, except the no-dummy version – i.e., even though the real traffic is paced via time-slots, the empty time-slots are left vacant (no dummy packets are sent. The frequencies corresponding to the original flows (58, 92, 116, 150, 184 and 214 Hz – see Figure 8) return, thus revealing the flows again. This implies dummy packets are crucial for hiding flows.

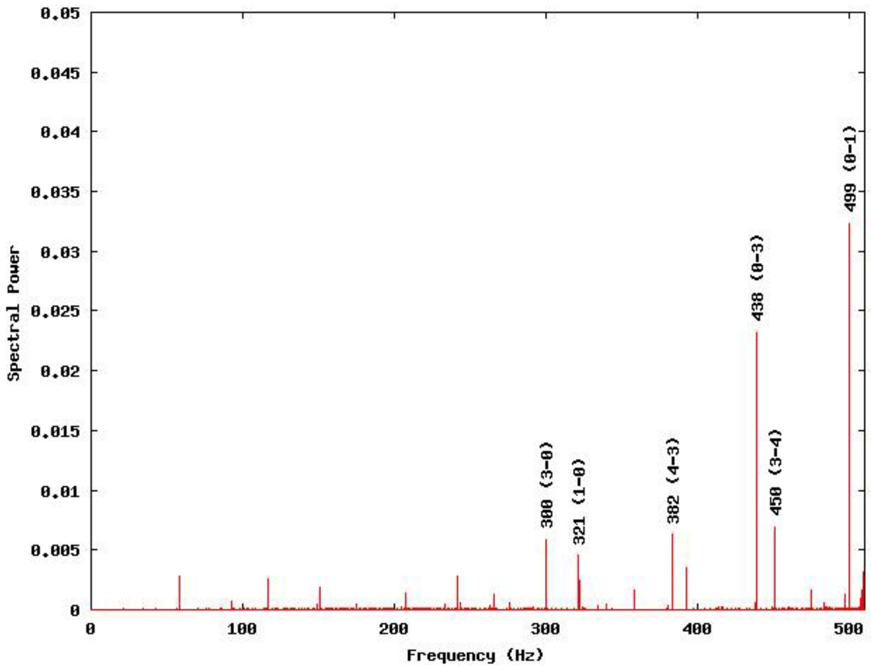

Figure 10 Non-Uniform Pacing with Dummy Packets

Again, only the pacing frequencies of the different tunnels (300 Hz for 3→0 tunnel, 321 for 1→0, 382 for 4→3, 438 for 0→3, 450 for 3→4 and 499 Hz for the 0→1 tunnel) dominate, hiding characteristic frequencies of individual flows.

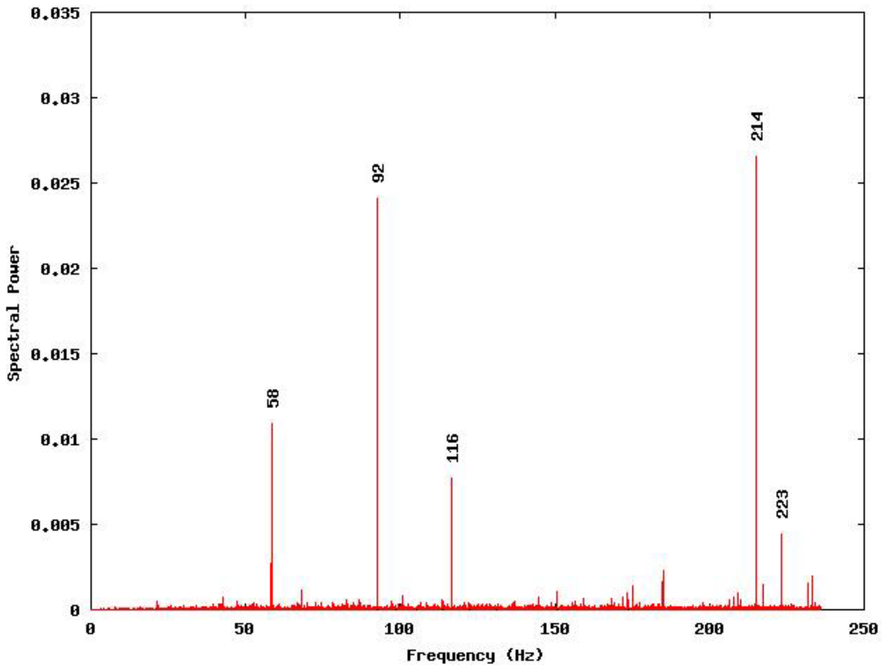

Figure 11 Non-Uniform Pacing without Dummy Packets

The “no-dummy” version of Figure 10. Again, if dummy packets are not sent in the empty time-slots, then the frequency response reveals the characteristic frequencies (5, 92, 116, 214 and 233 Hz) of the underlying flows – even though the data packets *were* paced via the time-slots.

As an example, we reran the simulation for the network in Figure 6 with paced tunnels between all gateway pairs, for all four configurations discussed above. All communication traffic was sent through the tunnels. The same flows were created as shown in Figure 6 and used in Figure 7. The Lomb periodogram on the 0-3 link, for all four cases are shown in Figure 8 (same pacing frequency in all tunnels, with dummy packets), Figure 9 (same pacing frequency in all tunnels, but *without* dummy packets), Figure 10 (different pacing frequency in all tunnels, with dummy packets) and Figure 11 (different pacing frequency in all tunnels, but *without* dummy packets).

As can be seen from Figure 8 and Figure 10, when tunnels are paced, and dummy packets are used, the tunnel pacing frequencies dominate the frequency response, while the characteristic frequencies of individual flows (which were visible in Figure 7) are now concealed. Therefore, in this example, paced tunnels are able to successfully suppress flow information from the snooping agent, thus making the network more secure against eavesdropping.

5.4 Experiences from Paced Tunnel Simulations

We now briefly discuss three key observations from our experiences with paced tunnel VPN simulations.

Paced tunnels conceal key flow characteristics*:* Our simulations suggest that if paced VPN tunnels are used, then the frequencies that dominate the core links are the pacing frequencies of the tunnels – not the frequencies corresponding to individual flow timings. This makes it difficult to distinguish the characteristics of individual flows, as they come on and off. Essentially, the pacing frequencies *mask* the flow characteristics by imposing pacing on packets corresponding to individual flows.

As can be seen from Figure 8 and Figure 10, the frequencies that dominate, correspond to the pacing frequencies of the various tunnels sharing that link. The flow characteristic frequencies are not visible in the Lomb periodogram, thus though one may be able to recognize the tunnels, the flows, or communications inside the tunnels are hidden, and their characteristics are invisible.

Paced tunnels conceal change in communication traffic volume*:* Paced tunnels also hide the volume of communications between gateways. This is because when tunnels are paced (with the dummy packets), the spectral power (see y-axis of the periodogram) will not change with increasing communications between gateways. Instead, when the volume of communications (or data packets) between two sites (say gateway 3 and 4) increases, the tunnel between gateways 3 and 4 will simply replace more dummy packets by data packets (i.e. the ratio of data packets/dummy packets on the tunnel goes up). However, the rate at which packets (both dummy and real) are sent between nodes 3 and 4 remains constant (and equal to the pacing frequency). Therefore, if a snooping agent is restricted only to detect the time of arrivals and not look inside the packets, then, since the timeseries is not affected, the periodogram remains unchanged. “Real” data packets, replace a larger number of dummy packets keeping the pacing frequency and the corresponding frequency response unchanged, thus preventing discovery of the event. The power of the spectrum does not change with traffic.

Dummy packets are required for effective security*:* Moreover, as can be seen from Figure 9 and Figure 11, if dummy packets are suppressed and the time-slots without real data packets left empty, then the frequency response becomes very similar to the unpaced case. This is because without the dummy packets, the paced tunnel effectively delays the data packets to coincide their transmission times with the time-slot boundaries. However, the long-term timings between successive data packets still remain approximately the same as in the unpaced tunnels (except for a very small jitter introduced by the time-slot alignment). Therefore, the long term flow characteristics such as the round trip times, or the send rate of applications will still show up on the Lomb periodograms. If the dummy packets were used, then the tap would perceive inter-packet timing to be the pacing frequency, and since it can't distinguish between real and dummy packets, the pacing frequencies will dominate the Lomb periodograms. This means that dummy packets must be sent by the pacer to ensure effective concealment.

5.5 Summary and Future Work in Simulation

There are many options, as discussed in section 2 and section 4, for adding privacy protection to current networks. To date, no single solution seems to be having a significant impact. The provision of simulation support as described in this section is a crucial tool for allowing the experimentation required to identify the most valuable technologies.

6. CONCLUSIONS AND FUTURE WORK

The project began with a review of the literature in the area of Traffic Flow Confidentiality. The past research was categorized into mixing schemes and paced link schemes. Our analysis determined that mixing schemes are reasonably efficient to implement but remain vulnerable to sophisticated attacks. The paced link schemes tend to provide good protection but are much more wasteful of resources. In order to provide a fully secure and reasonably efficient solution to the traffic protection problem, BBN designed the Privatized Datagram Service (PDS). PDS uses a paced overlay network to ensure adequate protection and relies on optimizations in network design and the Encapsulating Security Payload (ESP) specification to improve efficiency.

The protection capability of the PDS design was analyzed using a simulation of a small sample network. The DARPA-developed ns-2 tool was used to develop the simulations. The simulation results demonstrate that signal processing techniques were unable to identify traffic threads in the PDS approach.

The efficiency of PDS is supported by proposed changes to the ESP specification to allow more padding and to enable quick identification of dummy packets. Some of the changes are expected to easily gain IETF approval as they are easy to implement and provide important utility for traffic protection. Other proposed changes are more debatable, in that while they provide useful protection capability, they also have costs in other areas, such as performance.

The network structure for PDS is a fully connected graph for small sets of nodes, and a low-dimensional torus for larger sets of nodes. A scheme was developed to add and delete nodes from the architecture in an efficient fashion. The node graph is used to create a network of paced VPN tunnels to carry the protected traffic. OSPF and MPLS are recommended to perform routing and traffic dispersion in the overlay network.

Additional simulations and experiments are necessary to determine whether the PDS approach can be made efficient enough for general use. The simulation tools developed under this contract can be used in future experimentation. The FTN experimentation program would be good environment to perform this experimentation.

As an initial suggestion for a future experiment, we propose that the FTN traffic-protection tools (PDS, NetCamo, and ANON) all be configured to support a simulated network of about 20 communicating enclaves. This may require some simulation coding as NetCamo and ANON may not currently have ns-2 implementations. The testing would focus on the costs created by the traffic protection schemes and the resistance to outsider attack. Such an experiment would be a useful basis to identify one or more of these protection schemes for further development.

REFERENCES

1. [Ber99] Bermond, J-C., Marlin, N., Peleg, D., and Perennes, S., “Directed Virtual Path Layouts in ATM Networks,” INRIA Technical Report No. 3665.
2. [Bra96] Bradner, S., "The Internet Standards Process -- Revision 3", RFC 2026, BCP 00009, October 1996.
3. [Bra97] Bradner, S., "Key words for use in RFCs to Indicate Requirement Level", BCP 14, RFC 2119, March 1997.
4. [Bre00] Breslau, L., Estrin, D., Fall, K., Floyd, S., Heidemann, J., Helmy, A., Huang, P., McCanne, S., Varadhan, K., Xu, Y., and Yu, H., Advances in Network Simulation, IEEE Computer, vol. 33, no. 5, pp. 59-67, May, 2000.
5. [Cha81] Chaum, D., *Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms*, **Communications of the ACM**, v 24, no 2, February 1981.
6. [Cla01] Clarke, I., Sandberg, O., Wiley, B., and Hong, T. W., *Freenet: A Distributed Anonymous Information Storage and Retrieval System,* in **Designing Privacy Enhancing Technologies: Intl. Workshop on Design Issues in Anonymity and Unobservability**, H. Federrath, Ed., Springer, New York, 2001.
7. [Dai98] Dai, W., PipeNet 1.1. available at http://www.eskimo.com/~weidai/pipenet.txt
8. [Fin01] Fink, R., et al., A Comparison of DYNAT with Current Traffic Flow and Network Confidentiality Technologies, BBN Technical Memorandum No. 1288, 4 April 2001.
9. [Fre02] Freedman, M. J., and Morris, R., *Tarzan: A Peer-to-Peer Anonymizing Network Layer*, to appear, 9th ACM Conference on Computer and Communications Security (CCS), November 2002.
10. [Gol97] Goldberg, I., Wagner, D., and Brewer, E., “Privacy-enhancing Technologies for the Internet,” Proceedings of IEEE COMPCON, 1997.
11. [Gol99] Goldschlag, David M., Reed, Michael G., and Syverson, Paul F., “Onion Routing for Anonymous and Private Internet Connections,” Communications of the ACM, vol. 42, num. 2, February 1999.
12. [Gua00] Guan, Y., Fu, X., Bettati, R., and Zhao, W., Efficient Traffic Camouflaging in Mission-critical QoS-guaranteed Networks, Proceedings of IEEE Information Assurance and Security Workshop, pp. 143-149, West Point, June 2000.
13. [Har98] D. Harkins and D. Carrel, “The Internet Key Exchange (IKE)” Internet Engineering Task Force RFC-2409, November 1998.
14. [Hin98] Hinden, R. and Deering, S., IP Version 6 Addressing Architecture, IETF RFC 2373, July 1998.
15. [Jea92] Jean-Marie, A. and Liu, Z., “Stochastic Comparison for Queuing Models via Random Sums and Intervals”, *Journal of Advanced Applied Probabilities,* No.24, pp. 960–985, 1992.

16. [Jea93] Jean-Marie, A. and Gun, L., “Parallel Queues with Resequencing”, *Journal of ACM* 40(5), pp. 1188–1208, 1993.

17. [Ken98] Kent, S., and Atkinson, R., "IP Encapsulating Security Payload (ESP)", IETF RFC 2406, November 1998.

18. [Ken02] Private communication.

19. [Kew01] Kewley, D.; Fink, R.; Lowry, J.; and Dean, M. “Dynamic Approaches to Thwart Adversary Intelligence Gathering,” Second DARPA Information Security Conference and Exhibition (DISCEX II), June 2001.

20. [Kun02] Kung, H. T., Bradner, S., and Tan, K.-S., An IP-Layer Anonymizing Infrastructure IEEE MILCOM, October 2002.

21. [McC] McCanne, S. and Floyd, S., *ns Network Simulator,* http://www.isi.edu/nsnam/ns/.

22. [Mit99] Mitra, D., Morrison, J., and Ramakrishnan, K. G., "Virtual private networks: Joint resource allocation and routing design," Proc. IEEE INFOCOM '99, pp. 480--490, 1999.

23. [Moy98] J. Moy, “OSPF Version 2”, Internet Engineering Task Force RFC-2328, April 1998.

24. [Nar01] Narten, T., and Draves, R., “Privacy Extensions for Stateless Address Autoconfiguration in IPv6,” IETF RFC 3041, January 2001.

25. [Opp89] Oppenheim, A., and Schafer, R., *Discrete-Time Signal Processing*, Prentice Hall, 1989.

26. [Par02] *Using Signal Processing to Analyze Wireless Data Traffic*, Craig Partridge et. al., ACM Workshop on Wireless Security (WiSe), MobiCom 2002, Atlanta Georgia, U.S.A, September 28, 2002.

27. [Rei98] Reiter, M., and Rubin, A., Crowds: Anonymity for Web Transactions, *ACM Transactions on Information and System Security*, vol 1. no 1., pp. 66-92, November 1988.

28. [Son01] Song, D. X., Wagner, D., and Tian, X., Timing analysis of keystrokes and timing attacks on SSH, Tenth USENIX Security Symposium, 2001.

29. [Syv97] Syverson, P., Goldschlag, D., and Reed, M., “Anonymous Connections and Onion Routing,” Proceedings of the 1997 IEEE Symposium on Security and Privacy, May 1997.

30. [Syv00] Syverson, P. F., Tsudik, G., Reed, M. G., and Landwehr, C. E., “Towards an Analysis of Onion Routing Security,” Workshop on Design Issues in Anonymity and Unobservability, Berkeley, CA, July 2000

31. [The02] Theriault, K., Farrell, W., Collins, M., Bettati, R., and Graham, B., Final Report for NetCamo Experiment, BBN Technical Memorandum No. 1327, August 2002

32. [Tho98] Thomson, S. and Narten, T., IPv6 Stateless Address Autoconfiguration, IETF Request for Comments: 2462, December 1998

33. [Tim96] Timmerman, B. *Adaptive Traffic Masking Techniques for Traffic Flow Confidentiality on Internetworks*, USC CS Technical Report 96-641, available at http://www.usc.edu/dept/cs/tech.html .

34. [Tim99] Timmerman, B., *Secure dynamic adaptive traffic masking*, Proc. New Security Paradigms Workshop, Caledon Hills Canada, September 1999.

35. [Wri01] Wright, M., Adler, M., Levine, B., and Shields, C., "An analysis of the degradation of anonymous protocols," University of Massachusetts Amherst Technical Report, April 2001. See http://citeseer.nj.nec.com/wright01analysis.html .

APPENDIX A TFC PACKING INTERNET DRAFT

Internet Engineering Task Force — Charles Lynn
Internet Draft — BBN Technologies
draft-clynn-ipsec-tfcpacking-00.txt — August 2002

Traffic Flow Confidentiality Packing

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of [RFC2026].

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress".

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of current Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Distribution of this memo is unlimited.

Abstract

The revised ESP specification [RFC2406bis] enables additional traffic flow confidentiality (TFC) by allowing more than 255 octets of padding. Typical use of TFC will result in all ESP packets being the same size, and the size will likely be the maximum transmission unit (MTU) of the path. The ESP specification does not have a mechanism to allow more than one protected packet to be included in the ESP packet, thus reducing the maximum achievable throughput. This specification addresses this deficiency by specifying a generic packet framing header.

Please send comments on this draft to CLynn@BBN.Com.

Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this specification. Distribution of this memo is unlimited.

Table of Contents

Table of Figures

1. Introduction

The revised ESP specification [RFC2406bis] enables additional traffic flow confidentiality (TFC) by allowing more than 256 octets of padding. Typical use of TFC will result in all ESP packets being the same size, and the size will likely be the maximum transmission unit (MTU) of the path. The ESP specification does not have a mechanism to allow more than one protected packet to be included in the ESP packet, thus reducing the maximum achievable throughput. This specification addresses this deficiency by specifying a generic packet framing header. The framing header also enables TFC to be used when sending frames that do not specify their own length.

1.1. Terminology

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in [RFC2119].

2. Packet Framing Header

The format of the packet framing header is shown in Figure 1.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  | EncapProtocol |  Encapsulated Packet Length   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1. Framing Header Format

The fields are used as follows.

Next Header
This field identifies the protocol of the item following this frame. The protocol number that identifies the framing header is (IANA-TBD). The value of this field in the last instance of this header in a packet is 59 [IANA-Protocols], "No Next Header". The Next Header octet MUST be aligned on a 4-octet boundary relative to the beginning of the ESP header.

Encap Protocol
This field identifies the protocol of the immediately following encapsulated frame. When ESP is used in tunnel mode and the encapsulated frame is an IP packet (either v4 or v6), the value of this field would be 4, "IP in IP (encapsulation)". Other

protocols that have an assigned protocol number MAY appear in this field. Note that "dummy packets", using protocol number 59 MAY be used to insert additional padding in multiples of four octets. This can be used to cause the body of the first encapsulated packet to be aligned on an 8-octet boundary.

Encapsulated Packet Length

This field specifies the length of the following encapsulated frame. Implementation of this specification MUST permit the presence of one to three alignment padding octets following the encapsulated frame, before the header identified by the Next Header field. Implementations of this specification MUST support the inclusion of alignment padding octets. The length of any alignment padding is not included in the encapsulated packet length. Implementations of this specification MUST align subsequent framing headers on a 4-octet boundary.

3. Packet Packing Examples

Figure 2 illustrates the use of the framing header to encapsulate an IPv6 packet, other unspecified packets, and an IPv4 packet as the Payload Data of a tunnel mode ESP packet.

A dummy frame, EncapProtocol = 59, is used to align the body of the IPv6 packet on an 8-octet boundary relative to the beginning of the ESP header (SPI field).

The IPv6 packet's length is 1 mod 4, so 3 octets of alignment padding are inserted to align the next framing header on a mod-4 boundary.

No alignment padding is required after the final encapsulated frame. It is followed by the first octet any TFC padding that might be present.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |               Security Parameters Index (SPI)                 |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Sequence Number                          |
   +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
   |   (IANA-TDB)  |      59       |               0               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |   (IANA-TDB)  |       4       | IPv6 Packet Length = 4*n + 1  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  6  |                                                         :
   :  IPv6 Packet  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |               |           mod-4 alignment padding             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   :              ... additional encapsulated packets ...          :
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |       59      |       4       | IPv4 Packet Length = 4*m + 3  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  4  |                                                         :
   :                  IPv4 Packet                  +=+=+=+=+=+=+=+=+
   |                                               |  TFC padding  |
   +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+               |
   :                 ... additional TFC padding ...                :
   :                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   : ... Optional Encr Padding ... |Encr Pad Length|  (IANA-TBD)   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                 Authentication Data (variable)                |
   ~                                                               ~
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2. Example ESP Packet Using Framing Header

Figure 3 is a variation on Figure 2 that illustrates how the last framing header may be omitted when the last encapsulated frame specifies its own length.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |               Security Parameters Index (SPI)                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                      Sequence Number                          |
    +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
    |   (IANA-TDB)  |      59       |               0               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |     4         |       4       | IPv6 Packet Length = 4*n + 1  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |  6  |                                                         :
    :  IPv6 Packet  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |               |            mod-4 alignment padding            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |  4  |                                                         :
    :                  IPv4 Packet                  +=+=+=+=+=+=+=+=+
    |                                               |  TFC padding  |
    +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+               |
    :                 ... additional TFC padding ...                :
    :                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    : ... Optional Encr Padding ... |Encr Pad Length|  (IANA-TBD)   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                 Authentication Data (variable)                |
    ~                                                               ~
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3. Omission of Last Framing Header

4. IANA Considerations

IANA needs to assign a number from the protocol-numbers space to the framing header. All instances of (IANA-TBD) need to be replaced with the number assigned.

5. Security Considerations

The use of the framing header has no security relevant implications when used within an ESP packet payload. Note, however, that the IPsec rules about the processing of packets tunneled via ESP MUST be separately applied to each of the encapsulated frames.

If used in other contexts, e.g., as the body of an IP packet, it may adversely impact the ability of current generation packet filtering devices to apply their filtering rules to the encapsulated frames.

6. References

[IANA-Protocols]http://www.iana.org/assignments/protocol-numbers.

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", RFC 2026, BCP 00009, October 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Level", BCP 14, RFC 2119, March 1997.

[RFC2406bis]Kent, S., "IP Encapsulating Security Payload (ESP)", Draft-ietf-ipsec-esp-v3-03.txt, July 2002. (Work in progress)

Author's Addresses

Charles Lynn
BBN Technologies
10 Moulton St.
Cambridge, MA 02138
CLynn@BBN.Com

Intellectual Property Rights

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurance of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

Full Copyright Statement

APPENDIX B SUMMARY OF NS-2 SIMULATION EXTENSIONS

Description of the FRT7 mods to ns2

This document describes the changes made to the ns2 simulator software to support FRT7 simulations. In particular, they describe the following changes:

1. added the macbroadcast method to Agent/Message to use MAC broadcast (rather than unicast) for data packets
2. support for paced data tunnels for FRT7

....
...

In ns-allinone-2.1b8a/ns-2.1b8a/Makefile.in

Added frt7-pacer.o and tcl/lib/ns-frt7pacer.tcl to make dependencies.

....
...

In ns-allinone-2.1b8a/ns-2.1b8a/frt7-pacer.cc:

A new class called Frt7Pacer derived from Connector that can be installed in each node at entry_. Allows the data to be aligned at boundaries of periodic intervals. Free intervals may optionally be filled with dummy packets.

Useful for future research pursuits that may explore possible countermeasures against FRT7 paced tunnels.

....
...

In ns-allinone-2.1b8a/ns-2.1b8a/frt7-pacer.h:

The class declaration and methods for Frt7Pacer.

```
+#ifndef ns_frt7_pacer_h
+#define ns_frt7_pacer_h
+
+#include <assert.h>
+
+#include "packet.h"
+#include "queue.h"
+#include "ip.h"
+#include "connector.h"
+
+class Frt7Pacer : public Connector {
+ public:
+     Frt7Pacer();
+     void recv(Packet* p, Handler*);
+     void send(Packet* p, Handler*);
+     void handle(Event* e);
+ protected:
```

```
+     int command(int argc, const char*const* argv);
+     double when_;
+};
+
+#endif
```

....
...
In ns-allinone-2.1b8a/ns-2.1b8a/message.cc:

Support for sending out a message as a MAC-layer broadcast rather than as a typical unicast message.

Included mac.h, and the macbroadcast command which can be used in lieu of send. With macbroadcast, the packet's destination field is set to BROADCAST in both the MAC and IP headers.

....
...
In ns-allinone-2.1b8a/ns-2.1b8a/tcl/lib/ns-default.tcl:

Set Frt7Pacer defaults:

```
+Frt7Pacer set when_ 0.0
+Frt7Pacer set debug_ false
```

....
...
In ns-allinone-2.1b8a/ns-2.1b8a/tcl/lib/ns-frt7pacer.tcl:

Contains OTcl methods for the FRT7Pacer base class. Procedures that are typically called from a simulation outside :

```
+Frt7Pacer instproc init {node}
   Installs the FrtPacer object on <node>.

+Frt7Pacer instproc start-pacing {tid}
   Starts pacing on the tunnel <tid>.

+Frt7Pacer instproc add-tunnel {tunid endpt intvl dummysend}
   Creates a tunnel <tunid> terminating at node <endpt> with pacing
   interval <intvl>. Dummy packets are sent if <dummysend> is 1.

+Frt7Pacer instproc remove-tunnel {tunid} {
   Removes the tunnel <tunid>.

+Frt7Pacer instproc add-dst-to-tunnel {dst tunid} {
   Adds the destination <dst> to the tunnel <tunid>.  Any packet
   destined to <dst> that arrives at this node will be sent to the
   tunnel endpoint for furhter forwarding.

+Frt7Pacer instproc remove-dst-from-tunnel {dst tunid} {
   Removes the destination <dst> from the list of destinations to be
   forwarded on tunnel <tunid>.

+Frt7Pacer instproc get-tunid {dst} {
   Gets the tunnel to be used for destination <dst>.
```

....

In ns-allinone-2.1b8a/ns-2.1b8a/tcl/lib/ns-lib.tcl:

Called ns-frt7pacer.tcl from ns-lib.tcl

....

Software Installation Manual

This document describes how to install the FRT 7 patch to ns-2. This document does NOT describe how to setup or use the FRT 7 patch for ns-2. Please contact tsaxena@bbn.com for additional information. Note also that the patch includes many changes beyond the FRT 7 modifications. In particular, they include many changes required for FastJam project, which was DARPA funded.

....
System requirements:
....

- We assume a typical x86 PC workstation running the Linux operating system including the gcc compiler version 2.96 (or higher). For getting started on Linux, see http://www.linux.org or http://www.redhat.com

  For example, on our setup:

  ```
  $ uname -a
  Linux a 2.2.16-22 #1 Tue Aug 22 16:49:06 EDT 2000 i686 unknown

  $ cat /etc/redhat-release
  Red Hat Linux release 7.0 (Guinness)

  $ g++ --version
  2.96
  ```

  The rest of this document assumes that you are using tcsh. The syntax may be slightly different on other command shells.

```
.............................................................................
* ns-2 version 2.1b8a
  (Available from http://www.isi.edu/nsnam/ns/)

  ns-2 is a large package and is required to create new simulation scenarios
  and generate input data for FastJam (and FRT7).

  Download the allinone version ns-allinone-2.1b8a.tar.gz from the webpage
  above.  Locate the FastJam/FRT7 ns-2 modifications in this directory
  (fastjam-frt7-ns-allinone-2.1b8a-mods.patch).  Then perform the following
  steps:

  $ tar xfz ns-allinone-2.1b8a.tar.gz
  $ patch -u -p0 < fastjam-frt7-ns-allinone-2.1b8a-mods.patch
  $ cd ns-allinone-2.1b8a
  $ ./install

  For additional information on building, installing, and using ns-2, see
  the webpage above.
```