DDT — A Versatile Tunneling Technology

Noritoshi Demizu[1]
Suguru Yamaguchi[2]
Nara Institute of Science and Technology

Abstract

Current internet technology doesn't solve the following three configuration problems for corporate networks: insulating a network located in the depths of a private network from the Internet, wasting Internet address space behind a firewall, and the lack of security in communication between corporations. This paper proposes a versatile tunneling model called "DDT" in order to give solutions for these configuration problems. This paper also gives solutions to other problems of tunneling, for example, explosion of packets, ignoring QoS, and veiling of packets. We then describe the structure of our DDT implementation on BSD. Finally, we discuss routing and a layering model for tunneling.

I. Introduction

In recent years, the tunneling technique has become popular in some internet protocol technologies in order to make various routing problems simple.

One of the reasons why we use tunneling is to provide a virtual link in order to carry minor protocols such as experimental protocols, new or historical protocols, and protocols of different families. For example, the current multicast routing mechanism uses tunneling to skip routers which can't handle multicast datagrams[3]. This tunneling protocol is "IP in IP". As another example, tunneling will be used in the transition from the current IP to the next generation of IP[1][2]. The other example is to carry different protocol families such as "OSI CLNP over IP"[3][4], "AX.25 over IP"[5], "IPX over IP"[6] and "SNA/IP"[7]. The Generic Routing Encapsulation (GRE)[8][9] was designed for this purpose.

Another use of tunneling is to support mobile hosts in the Internet. The Columbia method[10] uses tunneling to pass datagrams between Mobile Support Routers (MSR). MSRs construct a virtual network to which all the Mobile Hosts (MH) are connected. The Columbia method defines its own tunneling protocol called "IP within IP", and the tunneling provides the network with additional functions.

The other use for tunneling is to add auxiliary paths in a routing topology in order to simplify routing problems based on the network topology. The auxiliary paths can be considered as a kind of source routing[4], or a virtual link. For example, IDRP considers tunneling in order to carry transit datagrams between border gateways[11]. This method will make routing control simple by separating inter-domain routing control from intra-domain routing control. As another example, tunneling can be used to avoid limitations of packet forwarding technology[12] and routing protocol[13]. In yet another example, tunneling creates a private link between the same enterprise across an external service provider[14]. Internet Encapsulation Protocol[15] was designed to provide this kind of virtual network. "PPP over TCP" can also be used for this purpose[16].

However, tunneling has problems such as explosion of packets[17], ignoring Quality of Service (QoS), and veiling of packets. Many internet researchers consider tunneling dangerous and that it should be dealt with carefully. In addition to those problems, there doesn't exist a conventional model to deal with tunneling. Those who want to use tunneling should design tunneling implementations for each case.

In this paper, we propose a versatile tunneling model and give solutions to the problems described above.

This paper is organized as follows: Section II proposes our tunneling model which is called "DDT". Section III shows applications with DDT. Section IV identifies the problems of tunneling, and sections V and VI propose solutions. Section VII describes the structure of our implementation of DDT on BSD Unix System. Section VIII discusses routing and a layering model for tunneling, and proposes topics for a future study.

1 nori-d@is.aist-nara.ac.jp. He also works for OMRON Corporation and can be reached at demizu@nff.ncl.omron.co.jp.

2 suguru@is.aist-nara.ac.jp.

3 "Datagram" is a term for a packet or a Protocol Data Unit (PDU) in the internet protocol suite.

4 Or it could be called "gateway routing" because the demand comes from intermediate gateways.

II. Tunneling Model "DDT"

Tunneling is essentially a means to construct a virtual network. Since ordinary networks are provided through network interfaces, we believe that the virtual networks should be provided through virtual network interfaces. Consequently, virtual networks can be treated the same as ordinary networks, for example, in routing control. For these reasons, we propose a tunneling model called *DDT*[5].

> *Tunneling services are provided as a virtual network through a virtual interface.*

In our DDT model, pairs or groups of DDT virtual interfaces construct a virtual network. A packet thrown into a DDT virtual interface by a routing module will be encapsulated and sent to the peer of the DDT virtual interface. At the opposite gateway, the packet will then be decapsulated and come up to the routing module through the DDT virtual interface

Figure 1 shows an example of a DDT model applied to "IP over IP" tunneling. In this figure, the host H_a and the host H_b have both virtual and physical network interfaces, and each of the DDT virtual (physical) interfaces on H_a and H_b have an IP address V_a and V_b (P_a and P_b), respectively. The network N_v is a DDT virtual point-to-point network, and has a network address. Note that N_v can be treated the same as an ordinary network with these numbering in the current Internet architecture.

Now H_a selects the DDT virtual network N_v as the appropriate network to send an IP datagram to H_b according to its routing table. The datagram sent to N_v has the source address V_a and the destination address V_b, and it contains a TCP segment in its data part. In the real network, the IP datagram is encapsulated by another IP datagram and goes through a physical network. The encapsulated datagram has the source address P_a and the destination address P_b, and its data part contains the datagram which conceptually runs on N_v.

In the rest of this section, we discuss each of the DDT constituents: tunneling, virtual network, and virtual interface.

II.A. Tunneling

Broadly speaking, *tunneling* is defined as follows: Protocol Data Units (PDUs) in the network layer protocol x are sent by the network

[5] DDT stands for "doki doki tunneling" in Japanese, or "delightfully dangerous tunneling" in English, for historical reasons.

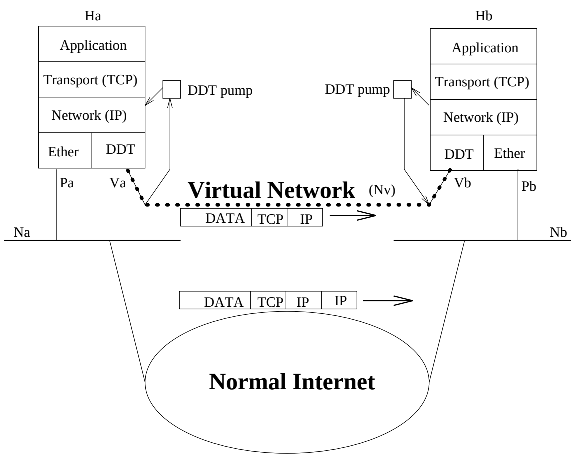

Figure 1: DDT applied to IP over IP

layer (or higher) protocol entity y instead of using a datalink layer protocol. This configuration of tunneling is often referred to as "x *over* y". In the narrow definition, x and y belong to the same protocol suite. In this paper, we focus on tunneling in the narrow definition, particularly tunneling of the Internet protocol suite such as "IP over IP", which encapsulates an IP datagram in another IP datagram, and "IP over TCP/IP", which serializes IP datagrams in a TCP stream.

II.B. Virtual network

The virtual network can be classified into two categories. The first category basically provides datalink functions like auxiliary paths in the routing topology. We call this type of virtual network a *pseudo network*. In this category, the tunneling mechanism can add powerful and useful functions to the PDU delivery, such as PDU encryption and reliability control.

The second category provides Network with Additional Functions (NAF) like supporting mobile hosts in the Internet. In this paper, we focus solely on the pseudo network.

II.C. Virtual interface

The virtual interface resides between the routing and the datalink modules. It exists as if it was the same as other ordinary "physical" network interfaces, such as an ethernet interface. As such the virtual interface is required to have Maximum Transfer Unit (MTU).

When the virtual interface receives a packet from the routing module, it encapsulates the packet and redirects it up to the routing module in order to send out the packet to the physical medium. Therefore, the DDT model violates the regulations of current layering models.

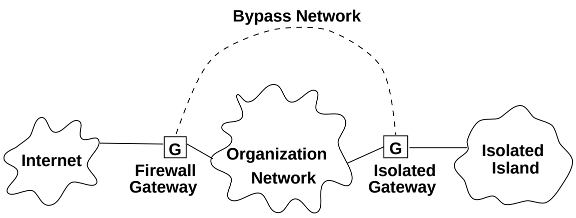

Figure 2: Isolated network

III. Application

In this section, we introduce tunneling applications provided with DDT, in addition to the applications described in section I.

III.A. Connecting isolated networks

Corporations often limit direct access between enterprise networks and the Internet for security reasons. It is often restricted to a small number of hosts near the firewall[6] gateway, even if the firewall gateway has a packet filtering capability. It is difficult for any department located far from the firewall gateway to access the Internet directly because it requires all the intermediate routers between such departments and the firewall gateway to have the Internet routing information. DDT can help them by making a virtual network between the department and the firewall gateway. Note that DDT doesn't require intermediate routers to have the Internet routing information.

This method can also be applied to the case that some of the subnets of a network are separated by a network with different network numbers.

III.B. Saving address space

Corporations often require a big address space even if only a few hosts are reachable from the Internet. One of the reasons is that they want to provide for the future when direct Internet access will be safely available via some future technology. So they allocate Internet reachable addresses to all their subnets, because they cannot tell which subnets will need access to the Internet. Even if they could tell exactly which hosts need access to the Internet, they would have to allocate Internet reachable addresses not only to the subnets to which the hosts are connected, but to all of the intermediate subnets as well.

In this case, it is possible to reduce the address space by making a DDT virtual network between the hosts and the firewall gateway in the same manner as described in section III.A. In this scheme, only the hosts and the firewall gateway require the Internet address, so that the use of the Internet address space is very small.

It is easier to save address space with DDT than with the Network Address Translator (NAT)[18]. NAT resides in the firewall router, and it translates any network addresses on every inbound datagrams from pseudo addresses to real addresses, and it also translates any network addresses on every outbound datagrams from real addresses to pseudo addresses. Therefore, NAT provides an image of pseudo hosts with Internet reachable addresses to the Internet. The address translations have to be performed on any protocol of any layer carrying network address information. Thus NAT requires the firewall gateway to know the details of all of such protocols and it makes it difficult to implement a complete NAT, while DDT requires no such knowledge.

III.C. Secure virtual network

As a countermeasure against eavesdropping, encryption functions are going to be embedded into several layers such as the application layer. DDT provides a method to build encryption support into a network layer by making a DDT virtual network with an encrypted tunneling protocol. Such a virtual network can be created dynamically between any hosts supporting DDT. Application programs can designate the DDT secure virtual networks by secure TOS[19].

This method can be applied to communications between two organizations across the Internet. In this case, the DDT secure virtual network should be created between their respective firewall gateways, and all traffic between them should be routed via the DDT secure virtual network. Note that this method doesn't require the communicating hosts to have an Internet reachable address, if there are no conflicts.

III.D. Experimental network

Researchers can use DDT to build their experimental network, because any topology can be easily emulated with a DDT virtual network.

IV. Problems

In this section, we address the Problems of Tunneling (PoT) within our DDT model. They are referred to as *PoT-num* in this paper.

1. Packet explosion, or the *big bang*
   In the case that an encapsulated packet is encapsulated again to enter a tunnel where the packet has been encapsulated before, the packet will become larger and larger

[6] "firewall" is a facility to protect an organization network from intruders.

with encapsulation headers infinitely[17]. We call it the *big bang*.

2. Poor control of QoS
   Users can attach Quality of Service (QoS) requests to each PDU. Since tunneling often hides part or all of the QoS information, encapsulated PDUs may be improperly routed.

3. Unidentified packet
   Encapsulation veils the real information, such as source and destination, of the encapsulated packets. Therefore, the packet filter cannot handle them correctly and it causes serious security problems. It is also hard to know current network usage to evaluate network configuration and performance.

4. Error reporting
   If a packet meets trouble in a tunnel, the error will be reported to the encapsulator. There needs to be a criterion for the encapsulator to decide whether the report should be ignored or forwarded to the actual source host. There also needs to be a method to forward the report to the actual source host correctly[15].

5. Encapsulation overhead
   Compared to ordinary packet delivery, tunneling requires additional CPU power for encapsulation/decapsulation, and additional bandwidth for encapsulation headers.

6. Packet redirection loop
   Packets sent to a DDT virtual interface are redirected up to the routing module to be sent out to a physical medium. It is possible that such packets are sent to the virtual interfaces again and again, and never reach the physical interface.

7. Optimum MTU of the virtual interface
   Although a DDT virtual interface is not based on a physical medium, it is required to have a proper Maximum Transfer Unit (MTU).

8. Inefficient routing, or the *black hole*
   Tunneling creates a conceptual network, whose availability relies on other networks. However, there is no routing protocol to handle a conceptual network. In the worst case, it is possible that there appears to be an illegal gateway which announces routing information without any guarantees. We call it the *black hole*.

9. Routing loop
   Due to the difficulty of routing control, a routing loop can be configured. In such a case, there needs to be a method to break the routing loop.

10. Protocol architecture model
    DDT doesn't fit the current protocol layering model because it involves layer violations.

11. Uncontrollable topology
    DDT gives an ability to create private links freely among users which makes network administration difficult.

To solve these problems, we partition DDT into two classes; connectionless DDT and connection-oriented DDT, whose respective solutions are often different. In the next two sections, we discuss how to solve each of these problems.

V. Connectionless DDT

In this section, we discuss how the problems, discussed in section IV, can be solved for connectionless DDT (CL-DDT), which encapsulates datagrams one by one.

In this scheme, we believe that encapsulation and decapsulation should be considered as a kind of transformation of a packet. Therefore attributes of each datagram should be inherited between the datagrams of before the transformation and of afterwards. In the case of the Internet protocol suite, the attributes include TTL, TOS and IP security option[7] in the IP header[15].

The inheritance of TTL solves the packet explosion (PoT-1) and routing loop (PoT-9) through TTL expiration. If TTL is decremented at each encapsulation, the redirection loop (PoT-6) can also be diminished in finite steps. The inheritance of QoS keeps the routing appropriate even in a tunnel (PoT-2). Because of the inheritance of TTL, the packet's TTL is decremented by the number of routes in the tunnel even when the packet just goes to the next hop router through the tunnel. ICMP[8] will be generated if TTL expires in the tunnel, so that CL-DDT tunnels are transparent.

Although the inheritance of TTL can break

[7] TTL stands for "Time To Live" and TOS stands for "Type of Service". Both TOS and IP security option are kinds of QoS.

[8] ICMP[20] stands for "Internet Control Message Protocol" and is a protocol to report errors and other network information.

opt	len	type	proto
Source Address			
Destination Address			
First eight octets of data			
part of IP option if possible			

Figure 3: IPOPT_TUNINFO

4	2	Dep	HL	Identifier
DF,MF,and Offset				IP opt if any
src/dst pairs if any				
IP data without first eight octets				

Figure 4: An example of IPPROTO_DDT

routing loops, all the packets sent into loops will circulate until the TTL has expired. Therefore it is necessary to detect routing loops to block packets from entering into the loops at the CL-DDT source gateway.

To solve unidentified packets (PoT-3), we propose a new IP option called IPOPT_TUNINFO defined as in Figure 3. Note that the first eight octets of the data section often contain important information such as port numbers. This IP option makes it possible to tell real information, such as an originator's address and the destination port of an encapsulated datagram, so that packet filtering can be performed.

We also propose a new tunneling protocol called IPPROTO_DDT whose behavior is defined in the first half of this section. It also plays an important role to report a proper ICMP to the actual source host. When a packet meets trouble in a tunnel, an ICMP will be sent to the CL-DDT gateway. Although the ICMP contains only an encapsulation header and first eight octets of an original datagram, the CL-DDT gateway has to generate a proper ICMP, which includes the copy of the IP header and the first eight octets of data part of the original datagram, from the received ICMP. In such a case, the proper ICMP can be generated from IPOPT_TUNINFO and IPPROTO_DDT with the method shown in Table 1. Note that IPOPT_TUNINFO is included in the encapsulation header and the first eight octets of IPPROTO_DDT is exactly the data part of encapsulated datagram. Figure 4 shows an example of IPPROTO_DDT. It is designed to have the information necessary to generate an ICMP in the first eight octets, and it throws away duplicated and computable fields to minimize encapsulation overhead (PoT-5).

Field	From where
VERS	IPPROTO_DDT
HL	IPPROTO_DDT
TOS	Encapsulation header
LEN	Encapsulation header
ID	IPPROTO_DDT
OFFSET	IPPROTO_DDT
TTL	Encapsulation header
PROTO	IPOPT_TUNINFO
Check sum	Computable
SRC, DST	IPOPT_TUNINFO
IP options	IPOPT_TUNINFO if possible
first 64bits	IPOPT_TUNINFO

Table 1: How to generate a proper ICMP

There is another approach to generate proper ICMP: putting an index of the information stored in the CL-DDT gateway in each encapsulated packet. We didn't take this approach because it doesn't seem to be able to deal with gigabit traffic.

VI. Connection Oriented DDT

In this section, we discuss how the problems, discussed in section IV, can be solved for the connection oriented DDT (CO-DDT), which serializes datagrams into a stream type tunnel.

In this scheme, the attributes of each datagram are ignored in a tunnel, so that all the datagrams in a tunnel are handled equally. Therefore it is important to select tunnels for each datagram properly, and many sorts of tunnels should be prepared to satisfy the policy of tunnel selection (PoT-2). The routers between CO-DDT gateways will not generate an ICMP for each datagram. Therefore CO-DDT tunnels are opaque. And it is difficult to identify each datagram in a CO-DDT tunnel (PoT-3).

To avoid routing loops (PoT-9) and redirection loops (PoT-6), the flow control plays an important role. If a flow is established and data ran in it, there should be no trouble between the CO-DDT gateways. Otherwise, the CO-DDT gateway should report an error such as "network unreachable" to the actual source host (PoT-4). Although there is a possibility that a control packet of a flow meets a black hole or a big bang, the packet will diminish in finite steps as described in section V. It is easier to reduce meaningless big bangs with CO-DDT than with CL-DDT, because the occurrence of control packets is much fewer than that of data packets.

The CO-DDT stream can be compressed (PoT-5) or encrypted, so that secure virtual networks can be configured. In the internet protocol

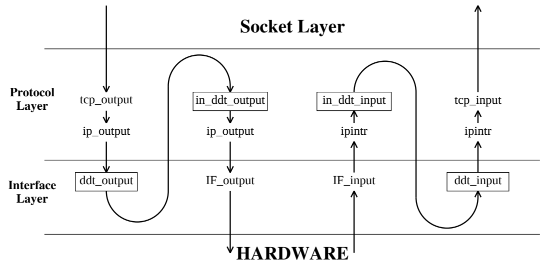

Figure 5: DDT implementation on BSD

suite, "PPP over TCP" or "SLIP over TCP" can be used for this purpose.

VII. Implementation on BSD

We implemented CL-DDT on several operating systems whose network code is based on BSD tahoe or BSD Networking Software Release 2 (NET2). Our implementation goals are: (1) to run CL-DDT on both BSD tahoe and NET2 with the same source code, (2) not to modify the vendor provided object files, and (3) not to depend on any particular protocol suite. Figure 5 shows the structure of our CL-DDT implementation on BSD. It consists of two parts; one is a virtual interface module (ddt_input/ddt_output, 484 lines) which provides network interface functions to the routing module, and the other is a *pump up module* (in_ddt_input/in_ddt_output, 471 lines) which performs encapsulation and decapsulation of datagrams and brings about layer violation.

To avoid redirection loops (PoT-6) between virtual interfaces, the MTU of a virtual interface is set to the MTU of the next hop interface minus the size of the encapsulation header. When a redirection loop is formed, the MTUs affect each other and converge to 0. The Path MTU discovery algorithm[21][22][23] can be used to find the optimum MTU if the next hop is through the physical interface. This method can also be applied to the virtual interface of CO-DDT (PoT-7).

Suppose a DDT virtual network connects two separated networks N_a and N_b. If a subnet number of the DDT virtual network belonged to N_a, the DDT gateway at N_b should have routing information of all the subnets belonging to N_a after the DDT gateway at N_b has made a direct connection to N_a. To avoid this, a subnet default route, a CIDR, or an unnumbered link is desired.

The key issue to implement an unnumbered link is whether route information of the DDT virtual link in a routing table can be easily marked

Problems	CL-DDT	CO-DDT
1.Big bang	TTL	Flow
2.QoS	Copied	Variation
3.Identifying	IP option	(Impossible)
4.Report	IP option and New protocol	Encapsulator reports
5.Overhead	Reducible by new protocol	Reducible by compression
6.Redirection	TTL or MTU	Flow or MTU
7.MTU	Next hop or path MTU	
8.Black hole	(not yet)	(not yet)
9.Loop	TTL	Flow
10.Model	(not yet)	(not yet)
11.Topology	(not yet)	(not yet)
(Others)	Transparent	Opaque Processable

Table 2: Problems and their solutions

"unavailable" while looking up the real route in the routing table. It is easy to implement this function on BSD tahoe just by toggling the IFF_UP flags of the DDT interface. But it is hard on NET2.

There is another way to implement DDT by using the Pnet[24] approach. The advantage of the Pnet approach is that it is easy to add various functions into the tunneling protocol. The disadvantage is that it is hard to implement unnumbered links.

VIII. Discussion

Table 2 shows the summary of our approaches to the problems discussed in section IV. There still remain PoT-8, PoT-10 and PoT-11.

VIII.A. Routing

PoT-8 is that routing control is difficult with a conceptual network and therefore DDT virtual networks cannot yet be announced by routing protocols. Currently, DDT can be used for end-to-end communication because there is no need to announce a DDT virtual network. DDT can also be used if a routing configuration was made statically with careful consideration. Our experiments show that it is not difficult to find out the proper routing configuration if the topology is not too complex, and that a source route capability of the tunneling mechanism is helpful.

There are two key issues to solve the routing problem. The first issue is how to announce dependency information of a conceptual network in the routing protocol. In the case of BGP, our idea is to introduce a "DEPENDENCY" path attribute which lists all of the depending DDT gateways. This method can be applied for other routing protocols if it supports an extensible path/link attribute.

The second issue is how to compute a proper routing table when the dependency information exists. Our idea is as follows: first of all, make a direct graph of the route dependency. Then, find out all of the dependency loops in the graph and mark all of the paths/links in the loops "unavailable". Then, mark all of the paths/links depending on the unavailable routes "unavailable". Following that, add the route information for the hosts in the dependency lists of the available routes. Finally, apply normal route computation. The evaluation of cost/metric of DDT virtual networks should be done carefully. Verification of these methods is left for a future study.

VIII.B. Layering Model

PoT-10 is that DDT doesn't match a current protocol layering model, so that a new layering model is required to accommodate the DDT. The reason why DDT involves layer violation is not that layering exists, but that a *strict ordering* of layers exists. Current layering models force their own order and role of layers, while tunneling doesn't observe such regulations. Such kind of regulations should be deleted from our new layering model.

In our layering model, all the Service Access Points (SAP), the interface to its upper layer, should be plug compatible. Each layer entity requires the next lower layer to provide preferred features. For example, SMTP requires the next lower layer to provide a byte stream. As long as all such requirements are satisfied, users can pile up layers in their preferred order. There's no need to obey the order of ordinary layering model.

Imagine a graph whose horizontal axis is a route in a network, and whose vertical axis is a stack of layers. The top of the layers is information or data, and the bottom of the layers is the physical media. There can be any number of layers with network functions and/or transport functions between the top and the bottom. The depth of the stack will be varied along the route. Because the horizontal axis is actually a two dimensional network, the whole of the model is three dimensional. An ordinary layering model can be considered as one of the *view points* to understand layering structure in our layering model.

We again define tunneling but within this model. By a broad definition, tunneling means piling up multiple layers, which have routing functions, onto the stack. By a narrower definition, some of the layers in the stack share the same address space. DDT can be normally described by this model. Tunneling is a way to use another network as a lower layer, and the network should be accessed through an interface to the lower layer.

On a route:	
Reach or die	PoT-1, PoT-6, PoT-9
QoS	PoT-2
Identity	PoT-3, PoT-4
Routing	PoT-8
MTU	PoT-6, PoT-7
Overhead	PoT-5
Layering Model	PoT-10
Uncontrollable	PoT-11

Table 3: Classification of PoT again

Ordinary "network layer" technologies are intended for two dimensional networks, but tunneling is a phenomenon in three dimensional networks. We believe the reason why tunneling is risky is that we don't yet have the necessary technology for three dimensional networks. Therefore we need to extend the current network technologies to three dimensional networks. In order to make it clear what we should consider, we reclassify the problems of tunneling discussed in section IV into Table 3.

First of all, we discuss the issues on a route. In CL-DDT, a packet goes around in three dimensional network space. When it goes down or up to the next layer, it will be encapsulated or decapsulated. Even in that case, we believe the packet should be considered identical, and TTL, QoS and Identity information should be passed along. In CO-DDT, the flow can be made only between the same layer, and contents of each flow is considered homogeneous. So CO-DDT gateway can leave "reach or die" issues to flow control, but should select a proper tunnel for each packet.

Second, we discuss the routing issue. In a three dimensional network, dependency relationship covers multiple layers. The solutions can be classified by the property of a direct graph of layer dependency. To make this graph, a layer which has a route depending on another layer's route should be considered to be dependent on the another layer, and layers with the same address space should be considered to be dependent on each other. In the case that there is no dependency loop in the graph, all the routes of the lower layers can be mapped to the upper layers, and the technology for a two dimensional network can be applied to each layer. In the case that all the loops in the graph are of the neighboring layers with the same address space, the method discussed in section VIII.A can be applied to the neighboring layers, and technology for a two dimensional network can be applied to the other layers. Otherwise, routing protocol should include the dependency information such as the protocol, the address space and the address information of each conceptual network. Or the exchange of routing information should be split so as not to form a dependency loop. Verification of these methods is left for a future study.

We believe our layering model is powerful enough to describe tunneling, and the technology for a three dimensional network is essentially the same as the technology for a two dimensional network.

IX. Conclusion

We have proposed a versatile tunneling model called DDT. It brings new tunneling applications such as connecting an isolated network, located in the depths of a private network, to the Internet, saving Internet address space behind a firewall gateway, and providing network level security between organizations. DDT has worked in some corporations to use these applications in their enterprise network.

Then, we have shown how problems of tunneling can be solved within our DDT model. It is useful to classify DDT into connectionless DDT (CL-DDT) and connection-oriented DDT (CO-DDT) to solve certain problems. with CO-DDT, it is easier to control risks than with CL-DDT, but packets can be identified only with CL-DDT.

Finally, we have discussed how to control the routing information of DDT virtual networks. We have also discussed a layering model which is required to accommodate the DDT, and shown how the problems of DDT can be expressed in that model. Verification of routing control methods is left for a future study.

Acknowledgment

Authors would like to thank Atsushi Onoe (SONY), Takahiro Kikuchi(ASTEM), Sei Kigoshi(Foretune), Jun Murai(Keio University) and the members of the WIDE Project for their active discussion and valuable suggestions. The Authors also thank Youki Kadobayashi(Osaka University), Nobuhiko Tada(Matsushita), Andy Krantz and Jeffrey Friedl (OMRON) for their great cooperation as proofreaders.

References

[1] R. E. Gilligan, E. Nordmark, and B. Hinden, "Internet Draft — IPAE: The SIPP Interoperability and Transition Mechanism," November 1993.

[2] D. M. Piscitello, "Internet Draft — Transition Plan for TUBA/CLNP," March 1994.

[3] R. Hagens, N. Hall, and M. Rose, "Use of the Internet as a subnetwork for experimentation with the OSI network layer, RFC1070," February 1989.

[4] D. Katz, "Internet Draft — Tunneling the OSI Network Layer over IP (EON)," March 1994.

[5] B. Kantor, "Internet protocol encapsulation of AX 25 frames, RFC1226," May 1991.

[6] D. Provan, "Tunneling IPX traffic through IP networks, RFC1234," June 1991.

[7] W. Behl, B. Sterling, and W. Teskey, "Advanced SNA/IP : A Simple SNA Transport Protocol, RFC1538," October 1993.

[8] S. Hanks, T. Li, and P. Traina, "Internet Draft — Generic Routing Encapsulation (GRE)," September 1993.

[9] S. Hanks, T. Li, and P. Traina, "Internet Draft — Generic Routing Encapsulation over IPv4 networks," September 1993.

[10] J. Ioannidis, D. Duchamp, and G. Q. M. Jr., "IP-based Protocols for Mobile Internetworking," in *Proceedings of ACM SIGCOMM'91*, pp. 235–245, September 1991.

[11] S. Hares and J. Scudder, "Internet Draft — IDRP for IP," September 1993.

[12] K. Y. et al., "Autonomous System Partitioning and Policy Routing in the Japanese Internet," in *Proceedings of INET'93*, August 1993.

[13] Y. Rekhter, "Internet Draft — Selecting an Indirect Provider," December 1993.

[14] Y. Rekhter, B. Moskowitz, D. Karrenberg, and G. de Groot, "Address Allocation for Private Internets, RFC1597," March 1994.

[15] W. Woodburn and D. Mills, "A Scheme for an Internet Encapsulation Protocol: Version 1, RFC 1241," July 1991.

[16] M. S. T. Inc., *Morning Star PPP User Guide*, April 1993.

[17] P. Tsuchiya, "Mutual Encapsulation Considered Dangerous, RFC1326," May 1992.

[18] K. B. Egevang and P. Francis, "Internet Draft — The IP Network Address Translator (NAT)," May 1993.

[19] I. D. Eastlake, "Physical Link Security Type of Service, RFC1455," May 1993.

[20] J. Postel, "Internet Control Message Protocol, RFC792," September 1981.

[21] J. Mogul and S. Deering, "Path MTU Discovery, RFC1191," November 1990.

[22] S. Knowles, "IESG Advice from Experience with Path MTU Discovery, RFC1435," March 1993.

[23] C. Kent, K. McCloghrie, J. Mogul, and C. Partridge, "IP MTU Discovery options, RFC1063," January 1988.

[24] S. M. Bellovin, "Pseudo-Network Drivers and Virtual Networks," in *Proceedings of Winter USENIX Conference, Washington DC*, pp. 229–244, 1990.

Author Information

Noritoshi Demizu has been working for OMRON Corporation since 1988, where he has developed windowing software, maintained the network environment, and researched the computer networks. He has also been a student of the Graduate School of Information Science, Nara Institute of Science and Technology, Japan since 1993. He received his B.S. from Kyoto University in 1988. He has been a member of the WIDE Project since 1991.

Suguru Yamaguchi received the M.E. and D.E. degrees in computer science from Osaka University, Osaka, Japan, in 1988 and 1991, respectively. From 1990 to 1992 he was an Assistant Professor in Education Center for Information Processing, Osaka University. From 1992 to 1993, he was with the Information Technology Center, Nara Institute of Science and Technology, Nara, Japan, as an Associate Professor. Currently, he is an Associate Professor with the Graduate School of Information Science, Nara Institute of Science and Technology, Nara, Japan. He has been also a member of WIDE Project, since its creation in 1987, where he has been conducting research on network security system for wide area distributed computing environment. His research interests include technologies for information sharing, multimedia communication, network security and network management for the Internet.