Network Layer Security and Virtual Private Networks Using SKIP

Teodora Ngo, William Soley

SunSoft Networking Security Group
teodora.ngo@sun.com, william.soley@sun.com

ABSTRACT

End-System SKIP implements the Simple Key-Management for Internet Protocols (SKIP) protocol. Any two or more systems running End-System SKIP will have the ability to encrypt all traffic between them, without having to modify applications.

This paper presents an architecture for using SKIP to implement Virtual Private Networks (VPNs). VPNs allow a closed group of hosts to communicate with each other as if they were attached to a physically isolated network. In fact, the VPN in this architecture is created on top of a publicly shared network such as the Internet. It provides isolation, privacy, and authentication, respectively, through encapsulation, encryption and digitally signed certificates. If desired, an administrator can add a firewall between the VPN and the underlying network to allow selected traffic to pass through.

INTRODUCTION

Commercial organizations are joining the Internet to take advantage of its global resources and services. At the same time, they do not want to allow external access to all of their internal resources and information. There are security risks on the Internet which can come from different sources. For example, hackers (or crackers) may try to have malicious fun at the expense of users whose computers are on the network. Industrial spies may try to steal trade secrets of other companies. Sometimes, well-intentioned users might accidentally expose corporate data or services from within a network. Even within an enterprise network (intranet), various levels of sensitive corporate data need restricted access. Examples are strategic plans, marketing plans, sales data, salary information, and employee evaluation.

A secure virtual private network based on the Internet offers an economical alternative for building enterprise networks compared to the traditional approach of using private leased lines. SKIP is an encapsulating protocol which is ideally suited for implementing secure VPNs. This paper provides some background on cryptography and the SKIP protocol. Then it examines the requirements of a virtual private network and describes an architecture for building secure VPNs using SKIP.

INTRODUCTION TO CRYPTOGRAPHY

Privacy ensures that when a message is sent, no one but the intended receiver can read or interpret it. *Authentication* means that the recipient of a message is assured that the sender is whom the message claims the sender to be. *Integrity* ensures that data received is the same as the data that was sent, without alteration.

Encryption is the transformation of data into a form unreadable by anyone except the sender and receiver. Its purpose is to ensure privacy by keeping the information hidden from anyone for whom it is not intended, even those who can see the encrypted data. The original message is called the *plaintext*, The resulting message is called *ciphertext*. Encryption is a reversible process. The inverse operation, *decryption*, transforms the ciphertext back to the original plaintext. Encryption and decryption is done by using a *key* or pair of keys.

Symmetric Cryptography

In traditional cryptography, the sender and receiver share a key in common and keep this key secret from everyone else, an arrangement called *symmetric cryptography*. A well-known example of this is the U.S. Data Encryption Standard (DES). The sender and receiver agree on the cryptographic algorithm and the key. The plaintext is encrypted with the key, and the ciphertext is decrypted with the same key or a key derived from the encrypting key. The process is rather simple, as follows:

1. Alice and Bob agree on the cryptographic algorithm and key.
2. Alice takes the plain text and encrypts it with the agreed cryptographic algorithm and key, resulting in the ciphertext.
3. Alice sends the ciphertext to Bob.
4. Bob decrypts the ciphertext with the same algorithm and key, and converts it back into plaintext.

The problem in this approach is that the sender and receiver must make arrangements to share a secret key, and keep this key secret from everyone else. If they are in separate physical locations, the sender and receiver must trust a courier or some communications system not to disclose the secret key. For *n* nodes to communicate, (n) (n-1)/2 pairs of secret keys must be distributed. This is quite cumbersome, considering that these secret keys need to be changed periodically.

Public Key Cryptography

A modern approach to this problem is *public-key cryptography*. Public-key cryptography was invented in 1976 by Whitfield Diffie and Martin Hellman at Stanford University. (Whitfield Diffie is now at Sun Microsystems.) It bypasses the problem of arranging shared secret keys by using two mathematically complementary keys, one known to everyone (*public key*), and one known only to its owner (*private key*). The two keys together are called a *key pair*. The keys are mathematically related, but even with knowledge of one key, it is computationally infeasible to figure out the other key. This permits one key, the *public key*, to be made widely known. Another popular example of a public key system was developed at MIT in 1977 by Ron Rivest, Adi Shamir, and Len Adleman. It is known by their initials, RSA.

Public key cryptography solves the problem of securely distributing keys by removing the need to communicate secret keys. Only the public keys need to be transmitted. They need not be kept secret, but it is important to guarantee their authenticity and integrity. If a message is encrypted with the public key, it must be decrypted with the private key. This is how privacy is achieved. If a message is encrypted with the private key, it must be decrypted with the public key. This is how authentication is achieved.

Choices: Symmetric or Public Key Cryptography?

The disadvantage of using public-key cryptography for encryption is speed. Several symmetric encryption algorithms are orders of magnitude faster than public-key algorithms. Symmetric cryptography is best for encrypting data, while public-key cryptography provides increased security, in that the private keys do not need to be revealed. The best solution is to combine both systems to get the best of both worlds. A public-key algorithm can be used to encrypt a secret key. A symmetric algorithm can use this secret key to encrypt the bulk of data. This is the approach taken by the SKIP protocol.

Diffie-Hellman Key Exchange

Diffie-Hellman Key Exchange is the first public-key algorithm invented. Its security is based on the difficulty of calculating discrete logarithms in a finite field, as compared to the ease of calculating exponentiation. The mathematics is simple. First, Alice and Bob agree on two large primes, g and p, such that g is primitive mod p. These two primes don't have to be secret.

1. Alice chooses a large random integer i and sends Bob

   g^i mod p.

2. Bob chooses a large random j and sends Alice

   g^j mod p.

3. Alice takes what she receives from Bob, g^j and raises to exponent i

   $(g^j \bmod p)^i = (g^j)^i \bmod p.$

4. Bob takes what he receives from Alice, g^i and raises to exponent j

   $(g^i \bmod p)^j = (g^i)^j \bmod p.$

Since $g^{ji} = g^{ij}$, Alice and Bob now have a common value without knowing each other's private values, i and j. Note that only the private values i and j need to be kept secret. The two large primes g and p, and the two exponentials g^i and g^j can be widely known without compromising the shared value g^{ij} mod p.

SKIP PROTOCOL

SKIP was developed by Ashar Aziz of Sun Microsystems' Internet Commerce group, and is being proposed to the Internet Engineering Task Force (IETF) as a standard. This paper focuses on using the SKIP protocol for encryption. End-System SKIP is an end-to-end encryption software that implements the SKIP protocol at the network layer (IP). Any two or more systems running End-System SKIP will have the ability to encrypt all traffic between them, without having to modify applications.

To run SKIP, each IP-based source and destination node needs an authenticated Diffie-Hellman public value. This public value may be authenticated in numerous ways. For example, public values can be distributed in the form of certificates. The certificates can be signed using either an RSA or DSA signature algorithm.

SKIP uses the principles of Diffie-Hellman key exchange to generate a shared secret key that is known only to the source and destination nodes. As with Alice and Bob in the example above, suppose node A wants to communicate securely with node B. Node A has a secret value i and a public value g^i mod p. Node B has a secret value j and a public value g^j mod p. Then, nodes A and B have a shared secret key, g^{ij} mod p. SKIP derives a long term key, Kij, from the shared secret key, by taking the low order key-size bits of g^{ij} mod p.

An individual IP packet is encrypted or authenticated using a randomly generated packet key denoted as Kp. Kp is in turn encrypted using the derived long term key Kij. This allows the packet key to be changed frequently and limits the amount of data encrypted by the long term key.

The encrypted packet is then encapsulated inside another IP packet. The encapsulated packet contains a plaintext IP header, a SKIP header, and the encrypted IP datagram. Figure 1 on page 3 shows the format of a SKIP packet.

The SKIP header contains the packet key Kp encrypted with long-term key Kij. The field Kij Alg identifies the encryption algorithm used for encrypting the packet key Kp. The field crypt alg identifies the encryption algorithm used for encrypting the IP packet. With this information, the receiver of the encapsulated packet can use the long-term key to decrypt Kp, and then decrypt the IP packet.

The optional fields src NSID and dst NSID (source and destination name space identifier) indicate that Master Key-IDs will be used to look up authenticated public values instead of the source and/or destination IP addresses. For a list of NSID assignments, see [2]. For example, NSID 5

IP_p	version	service	length	
	ident		offset	
	ttl	nxt=SKIP	cksum	
	src address			
	dst address			
SKIP	version	src NSID	dst NSID	nxt=AH
	counter N			
	Kij alg	crypt alg	MAC alg	comp alg
	Kp encrypted in Kijn			
	src Master Key-ID			
	dst Master Key-ID			
AH	nxt=ESP	length	(reserved)	
	SKIP_SPI			
	AH MAC computed using A_kp			
ESP	SKIP_SPI			
	ESP transform data			
	payload encrypted using E_kp			

Figure 1 SKIP Packet

indicates the name space using MD5 of DNS names, and Master Key-ID of 128-bits. The use of NSIDs allows dynamic assignment of IP addresses.

The description uses the number theoretic construction of classic Diffie-Hellman. However, the SKIP protocol can be generalized to any public key agreement algorithm. Examples include constructions that employ elliptic curves over finite fields.

SKIP Protocol for Multicast IP

The traditional approach to key distribution for multicast applications is to have a multicast Key Distribution Center (KDC) distribute the multicast traffic key to all authorized group members. This one shared key is then used by all group members to encrypt multicast traffic.

There are at least two problems with this approach. First, key change does not scale well to a large number of nodes. Key change policies need to be a function of the amount of data encrypted with any given key, and not just a function of time. This means that for high speed links, the keys need to be updated far more frequently than for slower speed links. For a large number of nodes in a multicast group, it is difficult for a multicast KDC to rapidly supply updated keys to all group members.

The second problem with using the same key for encryption by all members of a multicast group is that it precludes the use of certain stream ciphers that are very efficient. This is because using the same key will result in the same key-stream to be used to encrypt different plaintext. Since key-stream reuse is catastrophic to the security of a stream cipher, it must always be avoided.

A simple extrapolation of SKIP for unicast IP described above solves both problems. Instead of distributing a traffic encryption key, a group owner distributes a Group Interchange Key (GIK). The GIK is then used as a key-encrypting key, similar to the way Kij is used for unicast IP.

To send encrypted traffic to a multicast group, a member first requests the GIK from the group owner. The group owner's identity needs to be known to the requesting node. This request is made using the unicast SKIP procedure described above, which includes source origin authentication. Once the group owner determines that the requesting node is on the group's access control list, it will provide the GIK to the requesting node. The requesting node will then encrypt the multicast traffic using a randomly generated traffic key, Kp. The traffic key is in turn encrypted using the GIK and sent with the packet.

This solves both problems noted earlier. Changing the multicast traffic encryption key is simple. Each source of multicast traffic can do this by randomly generating a new traffic key and sending it in-line with the multicast data packet. The multicast traffic encryption key can be updated very rapidly, even with every packet, if desired.

Since each source of encrypted traffic generates random traffic keys, it is natural that all senders of multicast traffic use different keys. This allows a stream cipher to be used for multicast traffic encryption. It is important to allow stream ciphers to be used with IP multicast because a common use of IP multicast is conferencing, especially video-conferencing. Since video is a demanding application in terms of speed and throughput, it is important to allow use of ciphers that are efficient in software.

VIRTUAL PRIVATE NETWORKS

A data communication *network* is made up of systems (nodes/routers) that are interconnected by a communication medium. In wide area networks, this medium is typically a transmission line. The service provided by the line is enhanced by the data-link layer to create a data-link connection (link). The service provided by the link is further enhanced by the network layer and higher layers. Figure 2 shows an example of a "real" link in terms of the OSI Reference Model.[6]

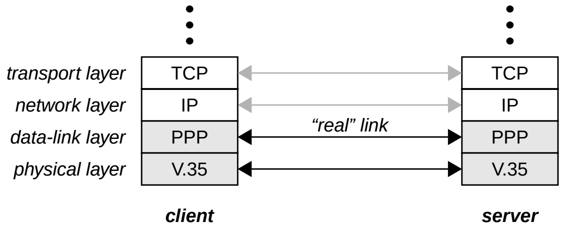

Figure 2 "Real" Link based on Transmission Line

A *private network* is a network that is dedicated to a single user group, for example, an enterprise network. A *public network* is a network that is shared by multiple user groups, for example, the Internet.

The communications medium that provides data-link services for a *virtual private network* (VPN) is an underlying network. The underlying network may be the same technology as the VPN or may be different.

The most common underlying network technologies used to build IP VPNs are X.25, Frame Relay, and IP. X.25 is often used because, in some countries, it is the least expensive or only tariff available. Frame Relay is popular because of its low overhead and ability to deliver a Committed Information Rate (CIR). IP is popular especially when public IP facilities are already available and may be shared.

The discussion here is limited to the case where the VPN and the underlying network are both IP networks.

Virtual Data-Links

The term *virtual private network* can be misleading. The network layer of a VPN is completely intact and just as real as it ever is. No modification of the network layer is required to implement a VPN.

The virtual part of a VPN actually takes place at the data-link layer. It is really the links of a VPN that are virtual. Figure 3 shows an example of a "virtual" link in terms of the OSI Reference Model.[6] It is constructed from two concatenated subsystem hierarchies. Virtual links may be point-to-point or multi-point. Too many point-to-point links can have a serious impact on performance and scalability.[8]

Protocol data units of the upper hierarchy are said to be *encapsulated* within those of the lower hierarchy. The lower hierarchy provides services to the data-link layer of the upper hierarchy that would normally be provided by a physical layer. To avoid confusion, the word *underlying* is used herein to refer to the lower hierarchy, for example, underlying network layer.

A virtual link must provide the same services at its interface to the network layer as would a real link, such as PPP, to allow a network entity, such as IP, to use them interchangeably. Likewise, a virtual link must use the underlying network interface, for example IP, as would a transport entity, to allow a single underlying network entity to serve them both.

Operational Advantages

The advantages of VPN can be numerous.[10] Very high demand is expected for software systems that satisfy these requirements. Some of the major advantages of VPN compared to leased lines are:

- Lower operating cost – Internet provides tremendous economy of scale for buying bandwidth and for network management.
- Topology agility – Virtual links can be created, re-routed and deleted without the long lead time or expense of ordering leased lines.
- Vendor flexibility – Services may be obtained from a variety of ISPs without getting "locked in" to one vendor.

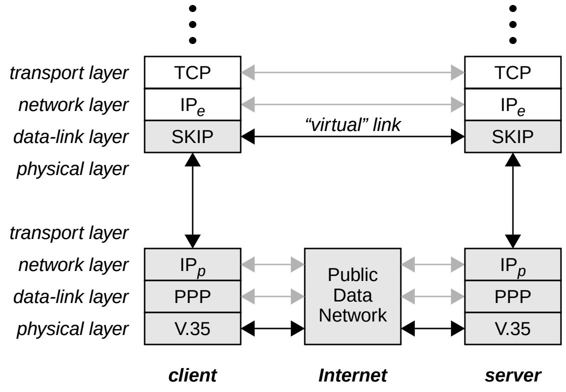

Figure 3 "Virtual" Link based on Public IP Network

Requirements

A VPN must provide the same services to the layer above it as would a real private network of the same technology. The users must not be able to tell the difference. The application software must not require modification.

Every user group or enterprise is likely to have different needs. But in general, VPN must meet the following specific requirements to satisfy the overall objective of behaving exactly like a real network:

Privacy, Authentication and Integrity

Enterprise security officers are often comfortable with sending confidential information over leased lines without authentication or encryption. The comfort is drawn on the (usually faulty) assumption that the effort and risk required to tamper with leased lines exceeds any value that might be derived.

An IP network is much more complex than the cross connects or channel banks used to provide leased line services. Its failure modes and potential for misconfiguration can easily result in misdelivered traffic, even without intentional tampering. The complexity also increases the chance of an attacker being able to find and exploit a bug or misconfiguration.

A basic requirement for VPN services should be strong privacy, authentication, and integrity. User groups or enterprises that are not concerned about these security issues probably do not need a VPN. They can just use the public network directly. An exception would be if the security aspects are handled by a higher layer in the stack, which is beyond the scope of this paper.

Logically Separate Networks

The VPN model shown in Figure 3 has two network layers. In the where both network layers are IP, it may be tempting to allow the boundary between these layers to become fuzzy. This must be avoided to provide maximum security, specifically to:

- Minimize value of traffic analysis.
  The specific endpoints of the traffic must be hidden as much as possible from observation within the Internet.
- Permit unregistered addresses in the VPN.
  An IP address, X, has no implied relationship to the same IP address, X, in the underlying network.
- Prevent routes to VPN from being advertised in Internet.
  First, this would disclose information about the internal topology of the VPN, and second, this would increase the risk of a software or configuration failure resulting in a breach.
- Prevent routes to Internet from being advertised in VPN.
  This would increase the risk of a software or configuration failure resulting in a breach.

The VPN network must behave as if it is not connected to the underlying network. Traffic must not be allowed to flow between any endpoint in the VPN and any endpoint in the underlying network. Traffic may only flow from an endpoint in the VPN through the underlying network to another endpoint in the VPN.

Dynamic IP Address Support

This is most likely to be an issue for traveling users, telecommuters and small offices that use dial-up access (analog or ISDN) to access the underlying network. In these cases, the service provider generally assigns the IP address from a pool each time the dial-up connection is established.

Even with statically assigned addresses, a particular user may connect to different service providers (or locations) at different times, producing the same effect as a dynamically assigned address (but it won’t change as often).

Protocols such as NFS, rsh, etc., which rely on the client IP address for access control, require static client addresses. In addition, network management and security auditing are simplified by static client addresses.

The VPN implementation should support static address assignment at the VPN network layer, even when the underlying network layer provides dynamic address assignment.

ARCHITECTURE FOR SKIP-BASED VPNS

SKIP is ideally suited for implementing VPNs. The SKIP design objectives explicitly satisfy the following requirements:

- authentication
- encryption
- encapsulation

As a side effect of SKIP’s protocol encapsulation, the following requirements can be met if SKIP is carefully implemented within the proper framework:

- logically separate networks
- dynamic IP address support

Remote Access Example

Figure 4 shows a sample topology composed of a client workstation at a remote site accessing an enterprise network server using a SKIP-based VPN. As shown, the virtual link is terminated directly on the remote client workstation. (A configuration typical of a telecommuter or traveler.)

A similar configuration for a remote site with several workstations, such as a branch office, would mirror the enterprise site in Figure 4 by having a SKIP gateway and enterprise LAN at the remote site, as well.

Network Addresses

The virtual link endpoint addresses are allocated from the enterprise network address space. These addresses need not be registered. Routes to the enterprise network are not advertised to the public Internet.

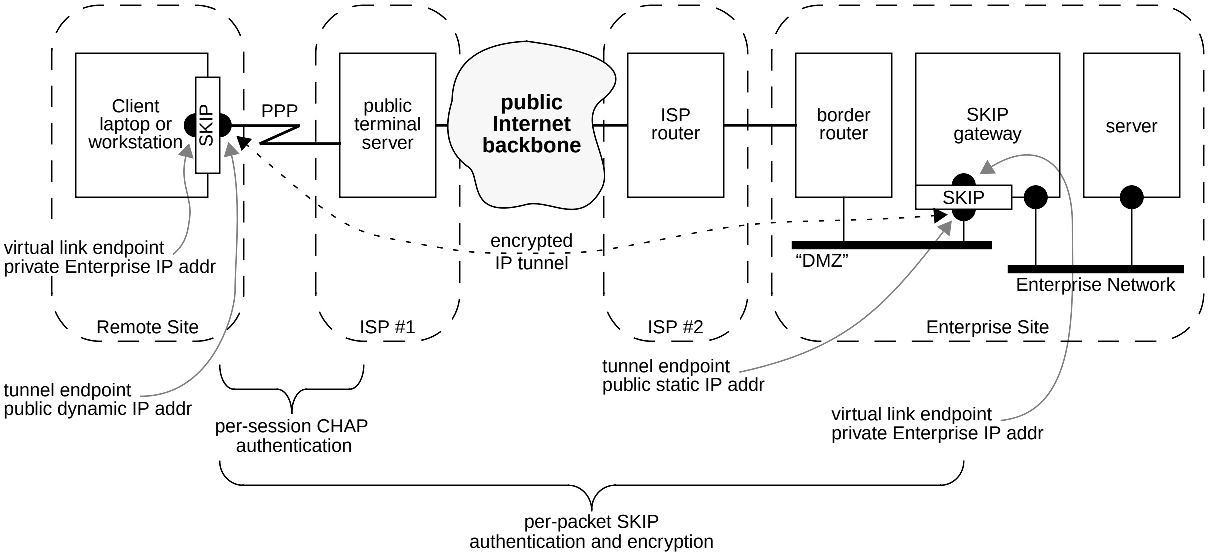
Client laptop or workstation
SKIP
PPP
public terminal server
public Internet backbone
ISP router
border router
SKIP gateway
server
SKIP
encrypted IP tunnel
“DMZ”
Enterprise Network
virtual link endpoint private Enterprise IP addr
Remote Site
ISP #1
ISP #2
Enterprise Site
tunnel endpoint public static IP addr
tunnel endpoint public dynamic IP addr
per-session CHAP authentication
virtual link endpoint private Enterprise IP addr
per-packet SKIP authentication and encryption

Figure 4 Sample Topology showing Secure Remote Access to Enterprise Network via Internet

The tunnel endpoint addresses are allocated from the public Internet address space. In the case of dial-up access, the remote tunnel endpoint address is likely to be dynamically allocated. Routes to the Internet are not advertised to the enterprise network.

Entity Relationships

Figure 5 shows an OSI entity relationship diagram for the same topology introduced in Figure 4. (The ISP routers are omitted for brevity.) SKIP authentication and encryption protect all header and data fields of layers above it when passing them through the public Internet (shown shaded).

The IP$_p$ header source and destination address fields contain the tunnel endpoint addresses (in the public network address space). The IP$_e$ header source and destination address fields contain the client and server addresses (in the enterprise network address space). The SKIP gateway appears as a router in the VPN network layer, and as an endpoint in the underlying public network layer.

Non-SKIP Traffic

The SKIP modules in the figures will pass only authenticated SKIP packets between the enterprise network and the Internet. In some cases selected traffic may be allowed to bypass SKIP and flow freely between the enterprise network and the Internet. For example, clients inside the enterprise network may be permitted to initiate TCP sessions to web servers in the public Internet. In such cases, some kind of firewall would be required to filter the internetwork traffic.

Solaris 2 Implementation

The Solaris 2 networking code is STREAMS-based. The Solaris STREAMS framework provides many possible designs for implementing VPN using SKIP. Two are briefly described below.

Wishful Design

Figure 6 shows a wishful design of SKIP for VPN within the STREAMS framework. In this design, the SKIP module plays a dual role. It is pushed on top of IP in the role of a transport module. It also is a pseudo DLPI* driver on which IP is pushed. This design elegantly mimics the OSI entity relationships (see Figure 5).

Unfortunately this design is impossible with the current Solaris implementation of IP. It requires two autonomous instances of IP, shown in the figure as IP$_e$ and IP$_p$. Each instance would require an independent IP address space, ARP cache, and routing table, requiring major changes to the Solaris IP implementation.

A compromise is possible if the enterprise network uses only registered addresses. One IP instance could serve both roles. Some filtering would be needed to prevent traffic from flowing directly into the public network without being processed by SKIP. There is considerable risk that minor misconfigurations could allow traffic to bypass SKIP. Therefore, this compromise is not recommended.

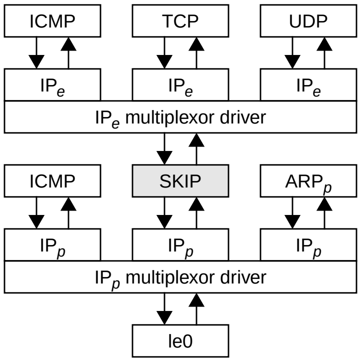

Figure 6 Wishful SKIP STREAMS Module Design

Realistic Design

A more realistic design for SKIP streams modules is shown in Figure 7. This design is outwardly identical to that used in the Sun Internet Commerce Group (ICG) implementations of end-node SKIP. The SKIP module interposes itself between the network device driver and IP. The difference between the existing ICG implementation and this design is new functionality to maintain a separate address space for the

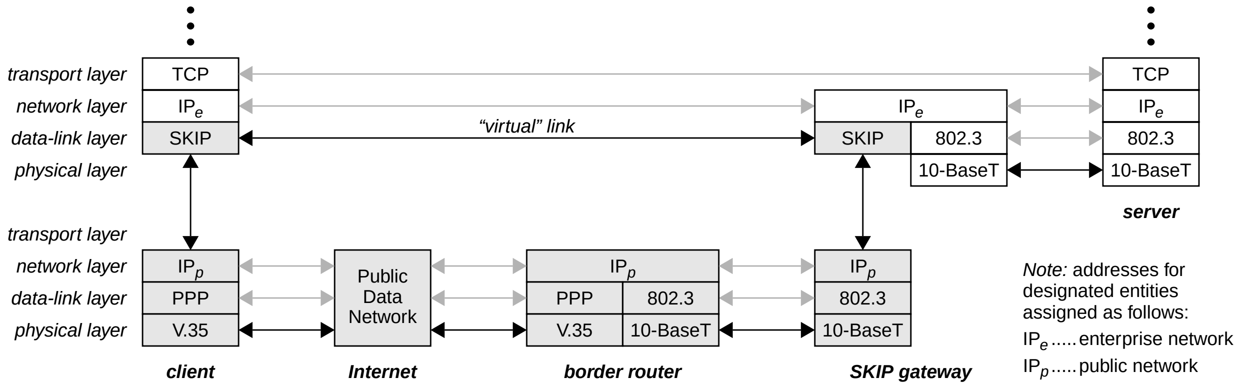

Figure 5 OSI Entity Relationships for Secure Remote Access to Enterprise Network via Internet

* Data Link Provider Interface

underlying network. This creates routing and key naming issues.

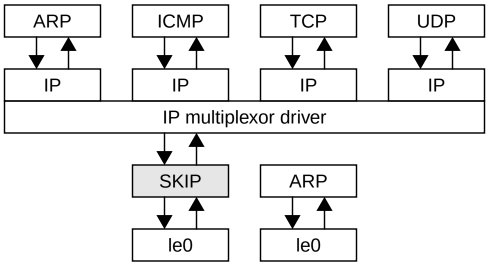

Figure 7 Realistic SKIP STREAMS Module Design

No IP module exists to provide underlying network services for the virtual link beneath SKIP. Therefore, the implementation of the SKIP module must duplicate much of the functionality of IP. Packet fragmentation, reassembly, and the associated MTU logic must be duplicated. A primitive routing mechanism must also be included.

Even though this design is not as elegant in terms of the implementation, it does have some advantages. It works with the existing implementation of IP in Solaris. Moreover, because it is much simpler to configure, there is less chance of configuration error resulting in traffic bypassing SKIP.

ACKNOWLEDGMENTS

- Catoe, Randy, internetMCI Network Security Architect, MCI, Reston, VA.
- Danielson, Bill, Internet Commerce Group, Sun Microsystems, Paolo Alto, CA.
- Gross, Phill, internetMCI Director of Engineering, MCI, Reston, VA.

REFERENCES

1. Ashar Aziz, “Simple Key-Management For Internet Protocols (SKIP)”, draft-ietf.ipsec-aziz-skip-00.txt, Internet Draft, May 1995.
2. Ashar Aziz, “Simple Key-Management For Internet Protocols (SKIP)”, draft-ietf-ipsec-skip-06.txt, Internet Draft Work in Progress, Dec 1995.
3. Ashar Aziz and Martin Patterson, “Design and Implementation of SKIP”, INET ’95 paper, Jun 1995.
4. Sun Microsystems, “Cryptography in Public Internetworks with SunScreen”, Internet Commerce Group, ICG-95-0002, 1995.
5. Whitfield Diffie and Martin Hellman, “New Directions in Cryptography”, IEEE Transaction on Information Theory, IT-22, pp. 644-654, Nov 1976.
6. ITU, “Reference Model of Open Systems Interconnection for CCITT Applications”, Rec. No. X.200, Geneva, 1985.
7. Bruce Schneier, *Applied Cryptography, Second Edition, Protocols, Algorithms, and Source Code in C*, John Wiley & Sons, Inc., 1996.
8. William Soley, “Virtual Link Models for Protocol Encapsulation”, BT Tymnet, San Jose CA, Feb 1991.
9. Andrew Tanenbaum, *Computer Networks, Second Edition*, Prentice-Hall, 1988.
10. U.S.Computer, “Internet-Based Secure Virtual Private Networks: The Cost of Ownership”, Saratoga CA, Jan 1996.