electronic signaling specifications, and data-link, transport, and application layer protocols. For the purposes of simplicity, let's just agree that a "network" is a collection of devices that can communicate in some fashion, and can successfully transmit and receive data amongst themselves.

The term "**private**" is fairly straightforward, and is intricately related to the concept of "virtualization" insofar as VPN's are concerned, as we'll discuss in a moment. In the simplest of definitions, "private" means that communications between two (or more) devices is, in some fashion, secret – that the devices which are not participating in the "private" nature of communications are not privy to the communicated content, and that they are indeed completely unaware of the private relationship altogether. Accordingly, data privacy and security (data integrity) are also important aspects of a VPN which need to taken into consideration when considering any particular VPN implementation.

Another means of expressing this definition of "private" is through its antonym, "public." A "public" facility is one which is openly accessible, and is managed within the terms and constraints of a common public resource, often via a public administrative entity. By contrast, a "private" facility is one where access is restricted to a defined set of entities, and third parties cannot gain access. Typically, the private resource is managed by the entities who have exclusive right of access. Examples of this type of private network can be found in any organizational network which is not connected to the Internet, or to any other external organizational network, for that matter. These networks are private due to the fact that there is no external connectivity, and thus no external network communications.

Another important aspect of "privacy" in a VPN is through its technical definition, as describing the privacy of addressing and routing system, meaning that the addressing used within a VPN community of interest is separate and discrete from that of the underlying shared network, and from that of other VPN communities. The same holds true for the routing system used within the VPN and that of the underlying shared network. The routing and addressing scheme within a VPN should, for all intents and purposes, be self-contained, but this degenerates into a philosophical discussion on the context of the term "VPN." Also, it is worthwhile to examine the differences between the "peer" and "overlay" models of constructing VPN's – both of which are discussed in more detail in Section 3.1, "Network Layer VPN's."

"**Virtual**" is a concept that is slightly more complicated. The *New Hacker's Dictionary* (formerly known as the Jargon File) **[2]** defines *virtual* as –

> **virtual /adj./** [via the technical term "virtual memory", prob. from the term "virtual image" in optics] **1**. Common alternative to {logical}; often used to refer to the artificial objects (like addressable virtual memory larger than physical memory) simulated by a computer system as a convenient way to manage access to shared resources. **2**. Simulated; performing the functions of something that isn't really there. An imaginative child's doll may be a virtual playmate. Oppose {real}.

Insofar as VPN's are concerned, the definition in **2.** above is perhaps the most appropriate comparison for virtual networks. The "virtualization" aspect is one that is similar to what we briefly described above as "private," however, the scenario is slightly modified – the private communication is now conducted across a network infrastructure that is shared by more than a single organization. Thus, the private resource is actually constructed by using the foundation of a logical partitioning of some underlying common shared resource, rather than by using a foundation of discrete and dedicated physical circuits and communications services. Accordingly, the "private" network has no corresponding "private" physical communications system. Instead, the "private" network is a virtual creation which has no physical counterpart. The virtual communications between two (or more) devices is due to the fact that the devices which are not participating in the virtual communications are not privy to the content of the data, and that they are also altogether unaware of the private relationship between the virtual peers. The shared network infrastructure could, for example, be the global Internet and the number of organizations or other users not participating in the virtual network may literally number into the thousands, hundreds of thousands, or millions.

A VPN can also said to be a ***discrete*** network **[3]** –

> **discrete \dis*crete"\, a.** [L. discretus, p. p. of discernere. See Discreet.] **1**. Separate; distinct; disjunct.

The discrete nature of VPN's allow both privacy and virtualization. While VPN's are not completely separate, per se, the distinction is that they operate in a discrete fashion across a shared infrastructure, providing exclusive communications environments which do not share any points of interconnection.

The combination of these terms produces ***VPN*** – a **private network** , where the privacy is introduced by some method of **virtualization**. A VPN could be built between two end-systems or between two organizations, between several end-systems within a single organization or between multiple organizations across the global Internet, between individual applications, or any combination of the above.

> *As an aside, it should be noted that there is really no such thing as a non-virtual network, when considering the underlying common public transmission systems and other similar public infrastructure components as the base level of carriage of the network. What separates a VPN from a truly "private" network is whether the data transits a shared versus a non-shared infrastructure. For instance, an organization could*

> *lease private line circuits from various telecommunications providers and build a private network on the base of these private circuit leases, however the circuit switched network owned and operated by the telecommunications companies are actually circuits connected to their DACS (Digital Access Cross-Connect Systems) network and subsequently their fiber optics infrastructure, and this infrastructure is shared by any number of organizations through the use of multiplexing technologies. Unless an organization is actually deploying private fiber and layered transmission systems, any network is layered with "virtualized" connectivity services in this fashion.*

A VPN doesn't necessarily mean communications isolation, but rather the controlled segmentation of communications for communities of interest across a shared infrastructure.

The common and somewhat formal characterization of the VPN, and perhaps the most straightforward and strict definition, is:

> ***A VPN is a communications environment in which access is controlled to permit peer connections only within a defined community of interest, and is constructed though some form of partitioning of a common underlying communications medium, where this underlying communications medium provides services to the network on a non-exclusive basis***.

A simpler, more approximate, and much less formal description is:

> ***A VPN is private network constructed within a public network infrastructure, such as the global Internet***.

It should also be noted that while VPN's may be constructed to address any number of specific business needs or technical requirements, a ***comprehensive*** VPN solution provides support for dial-in access, multiple remote sites connected by leased lines (or other dedicated means), the ability of the VPN service provider to "host" various services for the VPN customers (e.g., web hosting), and the ability to support not just intra-, but also inter-VPN connectivity, including connectivity to the global Internet.

2. VPN Motivations

There are several motivations for building VPN's, but a common thread in each is that they all share the requirement to "*virtualize*" some portion of an organization's communications – in other words, make some portion (or perhaps all) of the communications essentially "*invisible*" to external observers, while taking advantage of the efficiencies of a common communications infrastructure.

The base motivation for VPN's lies in the economics of communications. Communications systems today typically exhibit the characteristic of a high fixed-cost component, and smaller variable cost components which vary with the transport capacity, or bandwidth, of the system. Within this economic environment, it is generally financially attractive to bundle a number of discrete communications services onto a common high capacity communications platform, allowing the high fixed-cost components associated with the platform to be amortized over a larger number of clients. Accordingly, a collection of virtual networks implemented on a single common physical communications plant is cheaper to operate than the equivalent collection of smaller physically discrete communications plants, each servicing a single network client.

So, if aggregation of communications requirements leads to a more cost-effective communications infrastructure, why not pool all these services into a single public communications system? Why is there still the requirement to undertake some form of partitioning within this common system that results in these "*virtual private*" networks?

In response to this, the second motivation for VPN's is that of communications privacy, where the characteristics and integrity of communications services within one closed environment is isolated from all other environments which share the common underlying plant. The level of privacy depends greatly on the risk assessment performed by the subscriber organization – if the requirement for privacy is low, then the simple abstraction of discretion and network obscurity may serve the purpose. However, if the requirement for privacy is high, then there is a corresponding requirement for strong security of access and potentially strong security applied to data passed over the common network.

This paper can't do justice to the concept of VPN's without some historical perspective, so we need to take a quick look at why VPN's are an evolving paradigm, and why they will continue to be an issue of confusion, contention, and disagreement. This is important, since you will indeed discover that opinions on VPN solutions are quite varied, and everyone seems to be deeply religious on how they should be approached.

Historically, one of the precursors to the VPN was the Public Data Network (PDN), and the current familiar instance of the PDN is the global Internet. The Internet creates a ubiquitous connectivity paradigm, where the network permits any connected network entity to exchange data with any other connected entity. The parallels with the global Public Switched Telephone Network (PSTN) are, of course, all too obvious – where a similar paradigm of ubiquitous public access is the predominate characteristic of the network.

The public data network has no inherent policy of traffic segregation, and any modification to this network policy of permitting ubiquitous connectivity is the responsibility of the connecting entity to define and enforce. The network environment is constructed using a single addressing scheme and a common routing hierarchy, which allows the switching elements of the network to determine the location of all connected entities. All of these connected entities also share access to a common infrastructure of circuits and switching.

However, the model of ubiquity in the "Internet PDN" does not match all potential requirements, especially the need for data privacy. For organizations who wish to use this public network for private purposes within a closed set of participants (for example, connecting a set of geographically separated offices), the Internet is not always a palatable possibility. There are a number of factors behind this mismatch, including issues of quality of service (QoS), availability and reliability, use of public addressing schemes, use of public protocols, site security, and data privacy & integrity (the possibility of traffic interception). Additionally, a corporate network application may desire more stringent levels of performance management than is available within the public Internet, or indeed may wish to define a management regime which differs from that of the underlying Internet PDN.

It is worthwhile at this point to briefly examine the importance of Service Level Agreements (SLA's) in regards to the deployment of VPN's. SLA's are negotiated contracts between VPN providers and their subscribers, which contain the service criteria to which the subscriber expects specific services to be delivered. The SLA is arguably the only binding tool at the subscriber's disposal with which to ensure that the VPN provider delivers the service(s) to the level and quality as agreed, and it is in the best interest of the subscribers to monitor the criteria outlined in the SLA for compliance. However, Service Level Agreements present some challenging technical issues both for the provider and the subscriber. For the subscriber, the challenge is to devise and operate service measurement tools which can provide a reasonable indication as to what extent the SLA is being honored by the provider. Also, it should be noted that a subscriber may use a SLA to bind one or more providers to a contractual service level, but if the subscriber's VPN spans multiple provider's domains, the SLA must also encompass the issue of provider interconnection and the end-to-end service performance. For the provider, the challenge lies in honoring multiple SLA's from a number of service providers. In the case of an Internet PDN provider, the common mode of best effort service levels, is not conducive to meeting SLA's, given the unpredictable nature of the host's resource allocation mechanisms. In such environments, the provider either has to ensure that the network is very generously engineered in terms of the ratio of subscriber access capacity to internal switching capacity, or the provider can deploy service differentiation structures to ensure that minimum resource levels are allocated to each SLA subscriber. It must be noted that the former course of action does tend to reduce the benefit of aggregation of traffic, which in turn does have an ultimate cost implication, while the latter course of action does have implications in terms of operational management complexity and scalability of the network.

The alternative to using the Internet as a VPN today is to lease circuits, or similar dedicated communications services, from the public network operators (the local telephone company in most cases), and create a completely private network. It is a layering convention which allows us to label this as "completely private," as these dedicated communications services are (at the lower layers of the protocol stack) again instances of virtual private communications systems constructed atop a common transmission bearer system. Of course, this is not without precedent, and it must be noted that the majority of the early efforts in data networking, and many of the current data networking architectures, do not assume a deployment model of ubiquitous public access.

> *As an aside, it should be noted that this is quite odd, when you consider that the inherent value of an architecture where ubiquitous public access over a chaotic collection of closed private networks had been conclusively demonstrated in the telephony marketplace since the start of the 20th century. While the data communications industry appears to be moving at a considerable technological pace, the level of experiential learning, and consequent level of true progress as distinct from simple motion, still leaves much to be desired!*

Instead of a public infrastructure deployment, the deployment model used has been that of a closed (or private) network environment where the infrastructure, addressing scheme, management, and services were dedicated to a closed set of subscribers. This model matched that of a closed corporate environment, where the network was dedicated to serve a single corporate entity as the sole client. This precursor to the VPN can be called the private data network, and was physically constructed using dedicated local office wiring and dedicated leased circuits (or private virtual circuits from an underlying switching fabric such as X.25) to connect geographically diverse sites.

However, this alternative does have an associated cost, in that the client now has to manage the network and all it's associated elements, invest capital in network switching infrastructure, hire trained staff, and assume complete responsibility for the provisioning and on-going maintenance of the network service. Such a dedicated use of transport services, equipment, and staff is often difficult to justify for many small-to-medium sized organizations, and while the functionality of a private network system is required, the expressed desire is to reduce the cost of the service through the use of shared transport services, equipment, and management. There are a number of scenarios which can address this need, ranging from outsourcing the management of the switching elements of the network (managed network services), to outsourcing the capital equipment components (leased network services), to outsourcing of the management, equipment, and transport elements to a service provider altogether.

In the simple example illustrated in [Figure 1], Network “A” sites have established a VPN (depicted by the red lines) across the service provider’s backbone network, where Network “B” is completely unaware of it’s existence. Both Network “A” and Network “B” can harmoniously coexist on the same backbone infrastructure.

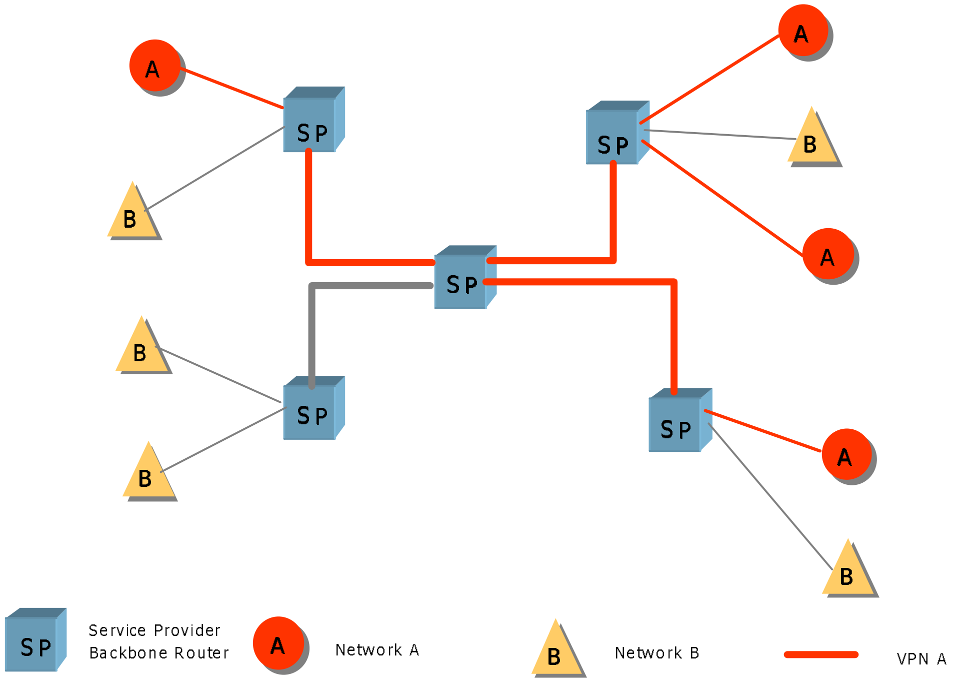


Figure 1

This is, in fact, the most common type of VPN – one in which there are geographically diverse subnetworks which belong to a common administrative domain, interconnected by a shared infrastructure outside of their administrative control (such as the global Internet or a single service provider backbone). The principle motivation in establishing a VPN of this type is that perhaps the majority of communications between devices within the VPN community may be sensitive in nature (again, a decision on the level of privacy required rests solely on a risk analysis performed by the administrators of the VPN), yet the total value of the communications system does not justify the investment in a fully private communications system which uses discrete transmission elements.

On a related note, the level of privacy a VPN may enjoy depends greatly on the technology used to construct the VPN. For example, if the communications between each VPN subnetwork (or between each VPN host) is securely encrypted as it transits the common communications infrastructure, then it can said that the privacy aspect of the VPN is relatively high.

In fact, the granularity of a VPN implementation can be broken down further to a single end-to-end, one-to-one connectivity scenario. Examples of these types of one-to-one VPN’s are single dial-up users establishing a VPN connection to a secure application, such as an online banking service, or a single user establishing a secure, encrypted session between a desktop and server application, such as a purchasing transaction conducted on the World Wide Web. This is type of one-to-one VPN is becoming more and more prevalent as secure electronic commerce applications become more mature and further deployed in the Internet.

It is interesting to note that the concept of virtualization in networking has also been considered in regard to deploying both research and production services on a common infrastructure. The challenge in the research and education community is one where there is a need to satisfy both network research and production requirements. VPN’s have also been considered as a method to segregate traffic in a network such that research and production traffic behave as “ships in the night,” oblivious to one another’s existence, to the point that major events (e.g. major failures, instability) within one community of interest are completely transparent the other. This concept is further documented in MORPHnet **[4]**.

It should also be noted that VPN’s may be constructed to span more than one host communications network, so that the “state” of the VPN may be supported on one or more VPN provider networks. This is perhaps at its most robust when all the providers explicitly support the resultant distributed VPN environment, but other solutions which do not necessarily involve knowledge of the overlay VPN are occasionally deployed with mixed results.

3. Types of VPN’s

The confusion factor comes into play in the most basic discussions regarding VPN’s. This is principally due to the fact that there are actually several different types of VPN’s, and depending on the functional requirements, several different methods of constructing each type of VPN is available. The process of selection should include consideration of what problem is being solved, risk analysis of the security provided by a particular implementation, issues of scale in growing the size of the VPN, and the complexity involved in both implementing the VPN, as well as ongoing maintenance and troubleshooting.

To simplify the description of the different types of VPN’s, they have been principally broken down in this paper into categories which reside in the different layers of the TCP/IP protocol suite [Figure 2].

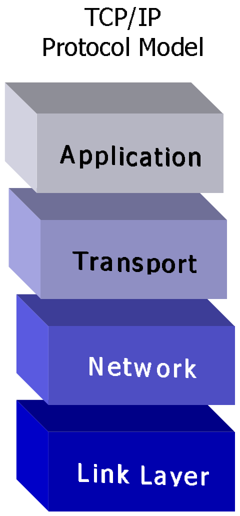


Figure 2

3.1 Network Layer VPN’s

The network layer in the TCP/IP protocol suite consists of the IP routing system – how reachability information is conveyed from one point in the network to another. There are a few methods to construct VPN’s within the network layer – each are examined below. A brief overview of non-IP VPN’s is provided in Section 4.0.

It is perhaps noteworthy at this point to provide a brief overview of the differences in the “peer” and “overlay” VPN models. Simply put, the “peer” VPN model is one in which the network layer forwarding path computation is done on a hop-by-hop basis, where each node in the intermediate data transit path is a peer with a next-hop node. Traditional routed networks are examples of “peer” models, where each router in the network path is a peer with their next-hop adjacencies. Alternatively, the “overlay” VPN model is one in which the network layer forwarding path is not done on a hop-by-hop basis, but rather, the intermediate link layer network is used as a “cut-through” to another edge node on the other side of a large cloud. Examples of “overlay” VPN models are ATM, Frame Relay, and tunneling implementations.

Having drawn these simple distinctions between the peer and overlay models, it should be noted that the overlay model introduces some serious scaling concerns in cases where large numbers of egress peers are required. This is due to the fact that the number of adjacencies increase in direct relationship with the number of peers – the amount of computational and performance overhead required to maintain routing state, adjacency information, and other detailed packet forwarding and routing information for each peer becomes a liability in very large networks. If each egress node in a cut-through network become peers, in an effort to make all egress nodes one “Layer 3” hop away from one another, this limits the scalability of the VPN overlay model quite remarkably.

For example, as the simple diagram [Figure 3] illustrates, the routers surrounding the interior switched infrastructure represent egress peers, since the switches in the core interior could be configured such that all egress nodes are one “Layer 3” hop away from one another, creating what is commonly known as a “cut-through.” This is the foundation of an overlay VPN model. Alternatively, if the switches in the interior were replaced with routers, then the routers positioned at the edge of the cloud now become peers with their next hop router nodes, not other egress nodes. This is the foundation of the peer VPN model.

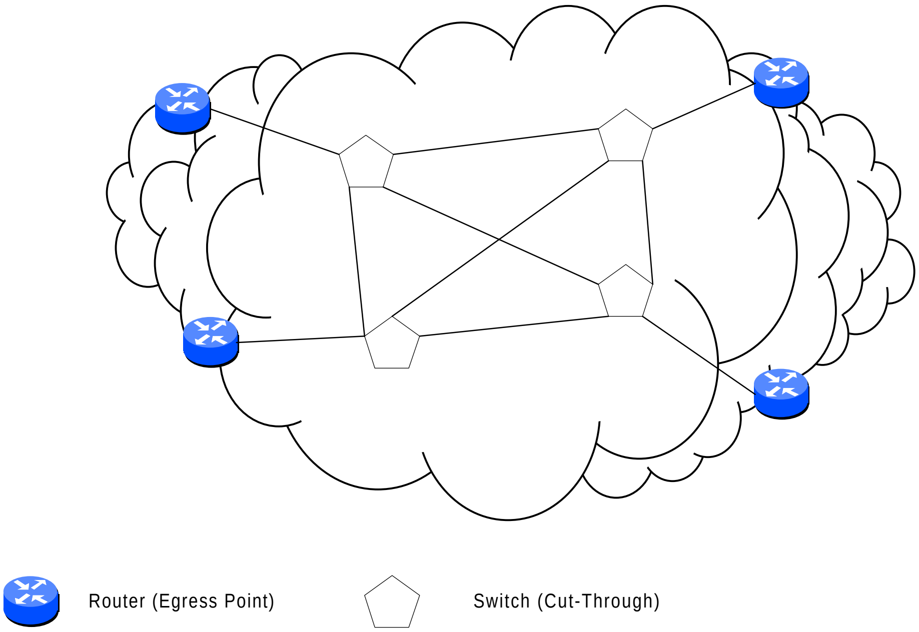


Figure 3

3.1.1 Controlled Route Leaking

"*Controlled route leaking*" (or *route filtering*) is a method which could also be called "privacy through obscurity," since it consists of nothing more than controlling route propagation to the point that only certain networks receive routes for other networks which are within their own community of interest. This model can be considered a "peer" model, since a router within a VPN site establishes a routing relationship with a router within the VPN provider's network, instead of an edge-to-edge routing peering relationship with routers in other sites of that VPN. While the common underlying Internet generally carries the routes for all networks connected to it, this architecture assumes that only a subset of such networks form a VPN.. The routes associated with this set of networks are filtered such that they are not announced to any other set of connected networks, and that all other non-VPN routes are not announced to the networks of the VPN. For example, in [Figure 1] above, if the service provider (SP) routers "leaked" routing information received from one site in Network "A" to only other sites in Network "A", then sites not in Network "A" (e.g., sites in Network "B") would have no explicit knowledge of any other networks which where attached to the service provider's infrastructure   [Figure 4]. Given this lack of explicit knowledge of reachability to any location other than other members of the same VPN, privacy of services is implemented by the inability of any of the VPN hosts to respond to packets which contain source addresses from outside the VPN community of interest.

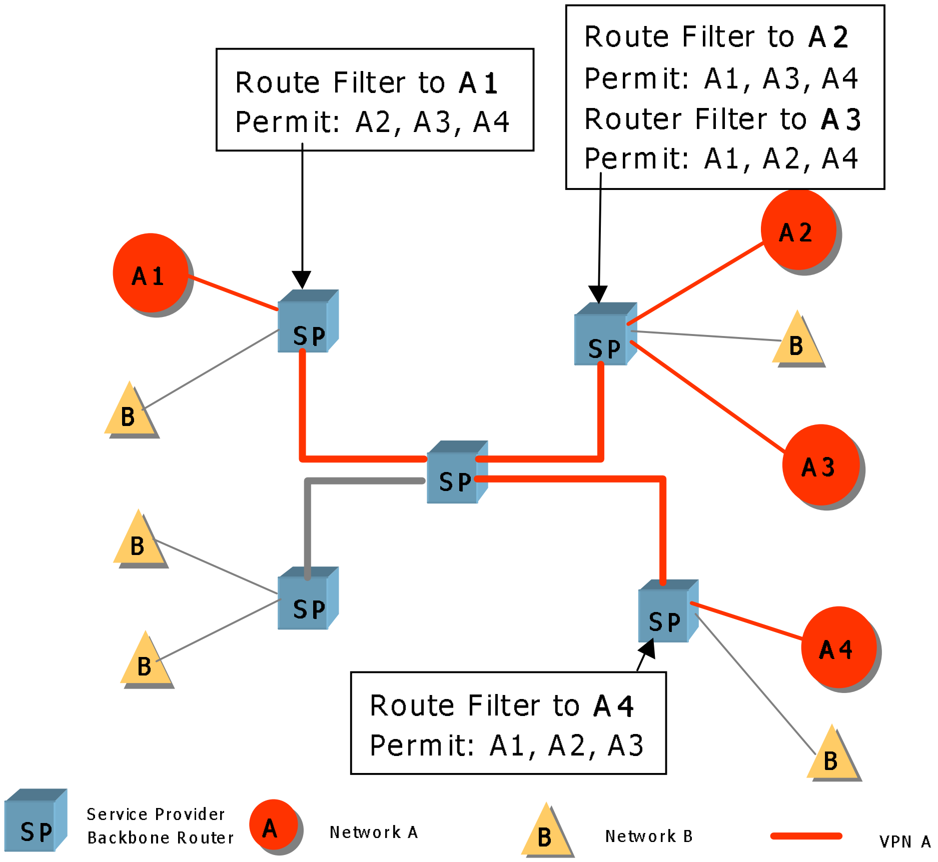


Figure 4

This use of partial routing information is prone to many forms of misconfiguration. One potential problem with route leaking is that it is extremely difficult, if not impossible, to prohibit the subscriber networks from pointing default to the upstream next-hop router for traffic destined for networks outside of their community of interest. From within the VPN subscriber's context, this may be a reasonable action, in that "default" for the VPN is reachability to all other members of the same VPN, and pointing a default route to the local egress path is, within a local context, a reasonable move. Thus, it is no surprise that this is a common occurrence in VPN's where the customer configures and manages the CPE (customer premise equipment) routers. If the service provider manages the configuration of the CPE routers, then this is rarely a problem. Otherwise, it may be wise on the part of the service provider to place traffic filters on first-hop router to prohibit all traffic destined for networks outside of the VPN community of interest.

It should also be noted that this environment implicitly assumes a common routing core. This, in turn, implies that each VPN must use addresses which do not clash with those of any other VPN on the same common infrastructure, and cannot announce arbitrary private addresses into the VPN. There is also another, perhaps less obvious, side effect of this form of VPN structure – it is not possible for two VPN's to have a single point of interconnection, nor is it possible for a VPN to operate a single point of interconnection to the public Internet in such an environment. A so-called "*gateway*" where all external traffic is passed through a control point which can both enforce some form of access policy and record a log of external transactions. The common routing core uses a single routing paradigm, based solely on destination address.

> *As an aside, it should also be noted that this requirement highlights one of the dichotomies of VPN architectures. VPN's must assume that they operate in a mutually hostile environment, where any vulnerability which exposes the private environment to access by external third parties may be exploited in a hostile fashion. However, VPN's rarely are truly isolated communications environments, and typically all VPN's do have some form of external interface allowing controlled reachability to other VPN's and to the broader public data network. The tradeoff between secure privacy and the need for external access is a constant feature of VPN's.*

To implement inter-VPN connectivity requires the network to route externally originated packets to the VPN interconnection point, and if they are admitted into the VPN at the interconnection point, the same packet may be passed back across the network to the ultimate VPN destination address. Without the use of Network Address Translation (NAT) technologies at the interconnection point of ingress into the VPN, this kind of communications structure is insupportable within this architecture [Figure 5].

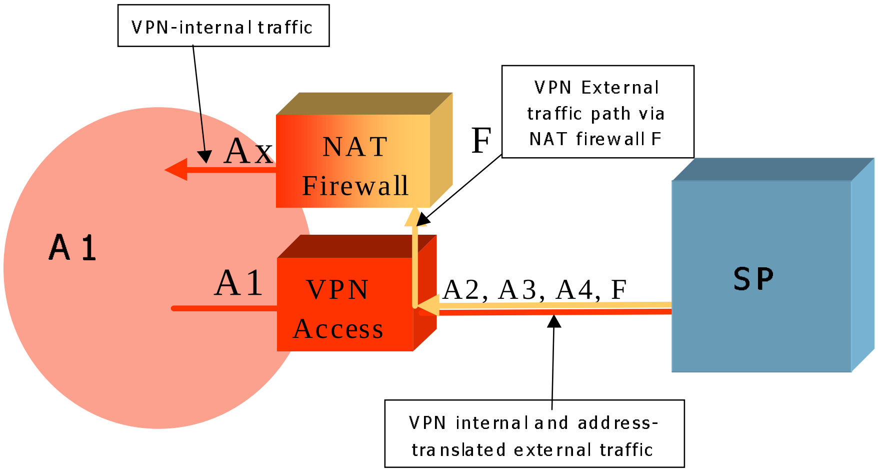

VPN-internal traffic
Ax
NAT Firewall
F
VPN External traffic path via NAT firewall F
A1
A1
VPN Access
A2, A3, A4, F
SP
VPN internal and address-translated external traffic

Figure 5

In general, the technique of supporting private communities of interest simply by route filtering can at best be described as a primitive method of VPN construction, which is prone to administrative errors, and admits an undue level of insecurity and network inflexibility. Even with comprehensive traffic and route filtering, the resulting environment is not totally robust. The operational overhead required to support complementary sets of traditional routing and traffic filters is a relevant consideration, and this approach does not appear to possess the scaling properties desirable to allow the number of VPN's to grow beyond the bounds of a few hundred, using today's routing technologies.

Having said that, however, a much more scaleable approach is to use BGP communities **[5]** as a method to control route propagation. The use of BGP communities scales much better than alternative methods insofar as controlling route propagation and is less prone to human misconfiguration. Briefly, the use of BGP communities attribute allows a VPN provider to “mark” BGP NLRI's (Network Layer Reachability Information) with a community attribute, such that configuration control allows route information to propagated in accordance with a community profile [Figure 6].

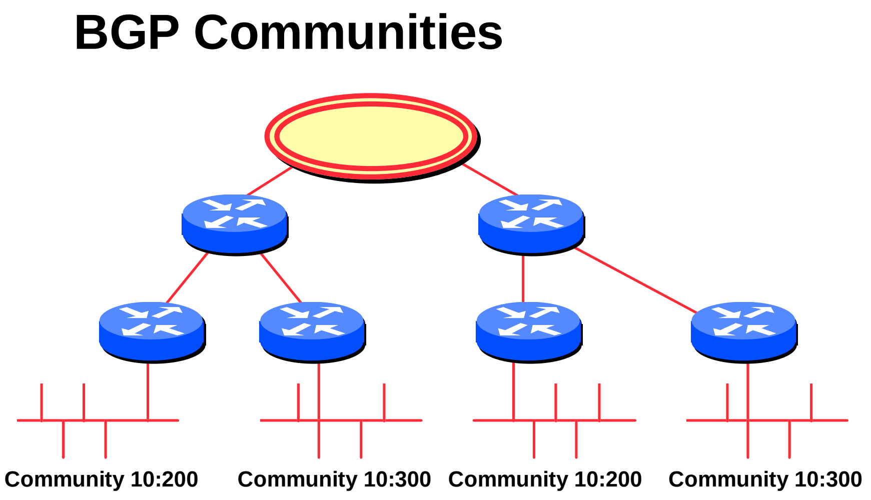


Figure 6

Due to the fact that traffic from different communities of interest must traverse a common shared infrastructure, there really is no data privacy (to speak of) in the portion of the network where traffic from multiple communities of interest share the infrastructure. Therefore, it can be said that while connected subnetworks – or rather, subscribers to the VPN service – may not be able to detect the fact that there are other subscribers to the service, multiple interwoven streams of subscriber data traffic pass unprotected in the core of the service provider's network.

3.1.2 Tunneling

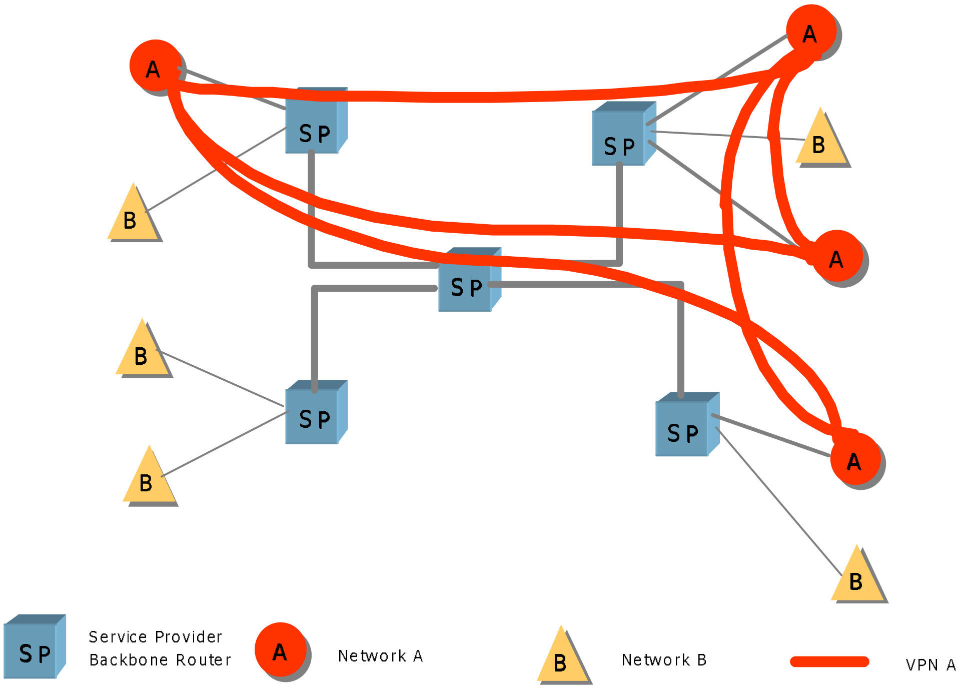
Sending specific portions of network traffic across a tunnel is another method of constructing VPN's – some more effective than others. The most common tunneling mechanisms are GRE (Generic Routing Encapsulation) **[6]** tunneling between a source and destination router, router-to-router or host-to-host tunneling protocols such as L2TP (Layer 2 Tunneling Protocol) **[7]** and PPTP (Point-to-Point Tunneling Protocol) **[8]**, and DVMRP (Distance Vector Multicast Routing Protocol) **[9]** tunnels.

Tunneling can be considered an overlay model, but the seriousness of the scaling impact relies on whether the tunnels are point-to-point or point-to-multipoint. Point-to-point tunnels, have lesser scaling problems than do point-to-multipoint tunnels, except in situations where a single node begins to build multiple point-to-point tunnels with multiple end-points. While there is a linear scaling problem introduced at this point, the manageability of point-to-point tunnels lies solely in the administrative overhead and the number of the tunnels themselves.. On the other hand, point-to-multipoint tunnels that use "cut-through" mechanisms to make greater numbers of end-points one hop away from one another and subsequently introduce a much more serious scaling problem.

While the Mbone (Multicast Backbone) itself could quite literally be considered a global VPN, and while DVMRP tunnels are still widely used by organizations to connect to the Mbone, it really is not germane the central topic of VPN's, since the focus of this paper is on unicast traffic.

3.1.2.1 Traditional Modes of Tunneling

GRE tunnels, as mentioned previously, are generally configured between a source (*ingress*) router and a destination (*egress*) router, such that packets designated to be forwarded across the tunnel (already formatted with an encapsulation of the data with the "normal" protocol-defined packet header) are further encapsulated with a new header (the GRE header), and placed into the tunnel with a destination address of the tunnel end-point (the new next-hop). When the packet reaches the tunnel endpoint, the GRE header is stripped away, and the packet continues to be forwarded to the destination as designated in the original IP packet header [Figure 7].



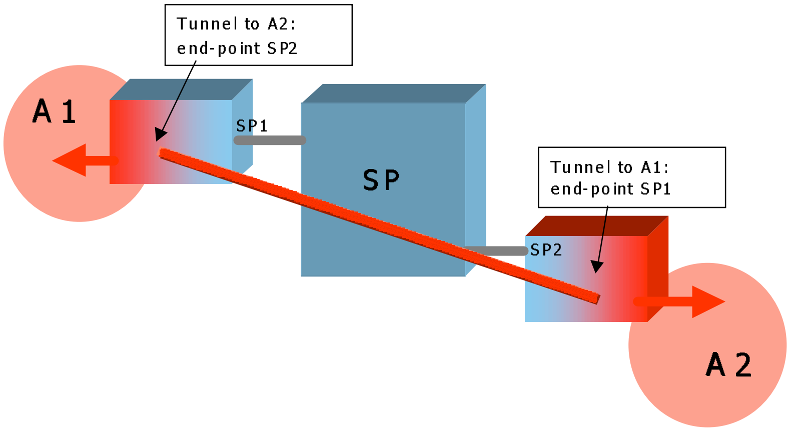


Figure 7

GRE tunnels are generally point-to-point – that is, there is a single source address for the tunnel and usually only a single destination tunnel endpoint. However, there are some vendor implementations that allow the configuration of point-to-multipoint tunnels – that is, a single source address and multiple destinations. While this implementation is generally used in conjunction with NHRP (Next-Hop Routing Protocol) **[10]**, the effectiveness and utility of NHRP is questionable and should be tested prior to deployment. It is also worthwhile to point out that NHRP is known to produce steady state forwarding loops when used to establish shortcuts between routers. In the above scenario, NHRP is used for establishing shortcuts between routers.

Tunnels, however, do have a number of very compelling attractions when used to construct VPN's. The architectural concept is to create VPN's as a collection of tunnels across a common host network. Each point of attachment to the common network is configured as a physical link which uses addressing and routing from the common host network, and one or more associated tunnels. Each tunnel endpoint logically links this point of attachment to other remote points from the same VPN. The technique of tunneling uses a tunnel egress address defined within the address space of the common host network, while the packets carried within the tunnel use the address space of the VPN, which in turn does constrain the tunnel endpoints to be collocated to those points in the network where the VPN and the host network interconnect. The advantage of this approach is that the routing for the VPN is isolated from the routing of the common host network. The VPN's can reuse the same private address space within multiple VPN's without any cross impact, which provides considerable independence of the VPN from the host network. This is a key requirement for many VPN's in that private VPN's typically may not use globally unique or coordinated address space, and there is often the consequent requirement to support multiple VPN's which independently use the same address block. Such a configuration is not supportable within a controlled route leakage VPN architecture. The tunnel can also encapsulate a number of different protocol families, so that it is possible for a tunnel-based VPN to mimic much of the functionality of dedicated private networks. Again, the requirement to support multiple protocols in a format which preserves the functionality of the protocol is a critical requirement for many VPN support architectures. This is a requirement where an IP common network with controlled route leakage cannot provide such services, whereas a tunneling architecture can segment the VPN-private protocol from the common host network. The other significant advantage of the tunneled VPN is the segregation of the common host routing environment with that of the VPN. To the VPN, the common host network assumes the properties of a number of point-to-point circuits, and the VPN can use a routing protocol across the virtual network which matches the administrative requirements of the VPN. Equally, the common host network can use a routing design which matches the administrative requirements of the host network (or collection of host networks), and is not constrained by the routing protocols used by the VPN client networks.

Perhaps these advantages would be sufficient to conclude that GRE tunneling is the panacea for VPN design. However, there are several drawbacks to using GRE tunnels as a mechanism for VPN's, mostly in regard to administrative overhead, scaling to large numbers of tunnels, and quality of service and performance. Since GRE tunnels must be manually configured, there is a direct relationship to the number of tunnels that must be configured and amount of administrative overhead required to configure and maintain them – each time the tunnel endpoints must change, they must be manually reconfigured. Also, while the amount of processing required to encapsulate a packet for GRE handling may appear to be small, there is a direct relationship to the number of configured tunnels and the total amount of processing overhead required for GRE encapsulation. Of course, tunnels can be structured to be triggered automatically, but there are a number of drawbacks to such an approach which dictate careful consideration of related routing and performance issues. The worst end-state of such automatic tunnel generation is that of a configuration loop where the tunnel passes traffic over itself. It is important, once again, to reiterate the impact of a large number of routing peering adjacencies resulting from a complete mesh of tunnels, which can result in negative effect on routing efficiency.

An additional concern with GRE tunneling is the ability of traffic classification mechanisms to identify traffic with a fine enough level of granularity, and not become a hindrance to forwarding performance. If the traffic classification process used to identify packets (which are to be forwarded across the tunnel) interferes with the router's ability to maintain acceptable packet-per-second forwarding rates, then this becomes a performance liability.

Privacy of the network remains an area of concern as the tunnel is still vulnerable – privacy is not absolute. Packets which use GRE formatting can be injected into the VPN from third party sources. To ensure a greater degree of integrity of privacy of the VPN, it is necessary to deploy ingress filters that are aligned to the configured tunnel structure. It is also necessary to ensure that the CPE routers are managed by the VPN service provider, as the configuration of the tunnel endpoints are a critical component of the overall architecture of integrity of privacy. This is a problem with using GRE tunnels in this fashion, since most VPN service providers do not wish to manage CPE routers. Arguably, one might suggest that having a dedicated CPE router defeats one of the basic premises of constructing a VPN – the use of shared infrastructure as a way to reduce the overall network cost.

It should be noted that VPN's can be constructed using tunnels without the explicit knowledge of the host network provider, and the VPN can span a number of host networks without any related underlying agreements between the network operators to mutually support the overlay VPN. Such an architecture is little different from provider-operated VPN architectures, and the major difference lies in the issue of traffic and performance engineering, and the administrative boundary of the management of the VPN overlay. Independently configured VPN tunnels can result in injection of routes back into the VPN in a remote location, which can cause traffic to traverse the same link twice, once in an unencapsulated format and secondly within a tunnel, which in turn can lead to adverse performance impacts.

It is also the case that the overlay VPN model has no control over which path is taken in the common host network, or the stability of that path, which in turn can lead to adverse performance impacts on the VPN. Aside from the technology aspects of this approach, the major issue is one of whether the VPN management is outsourced to the network provider, or undertaken within administrative functions of the VPN. One of the more serious considerations in building a VPN on tunneling is that there is virtually no way to determine the cost of the route across a tunnel, since the true path is masked by the cut-through nature of the tunnel. This could ultimately result in highly suboptimal routing, meaning that a packet could take a path determined by the cut-through mechanism which is excessively suboptimal, while native per-hop routing protocols might find a much more efficient method to forward the packets to their destinations.

3.1.2.2 Virtual Private Dial Networks (VPDN's)

While there are several technologies (vendor-proprietary mechanisms as well as open, standards-based mechanisms) available for constructing a virtual private dial network (VPDN), there are two principle methods of implementing a VPDN which appear to be increasing in popularity – L2TP and PPTP tunnels. From an historical perspective, L2TP is the technical convergence of the earlier L2F **[11]** protocol specification and the PPTP protocol. However, one might suggest that since PPTP will be included in the desktop operating system of the majority of the world's personal computers, it stands to be quite popular.

At this point it is worthwhile to distinguish the difference between "client initiated" tunnels and "NAS-initiated" ((Network Access Server, or dial access server, initiated). The former is commonly referred to as "voluntary" tunneling, while the latter is commonly referred to as "compulsory" tunneling. Voluntary tunneling is where the tunnel is created at the request of the user for a specific purpose – compulsory tunneling is where the tunnel is created without any action from the user, and without allowing the user any choice in the matter.

L2TP, as a "compulsory" tunneling model, is essentially a mechanism to "off load" a dial-up subscriber to another point in the network, or to another network altogether. In this scenario, a dial-up subscriber dials into a NAS (Network Access Server), and based on a locally configured profile (or a NAS negotiation with a policy server) and successful authentication, a L2TP tunnel is dynamically established to a predetermined end-point, where the subscriber's PPP session is terminated [Figure 8].

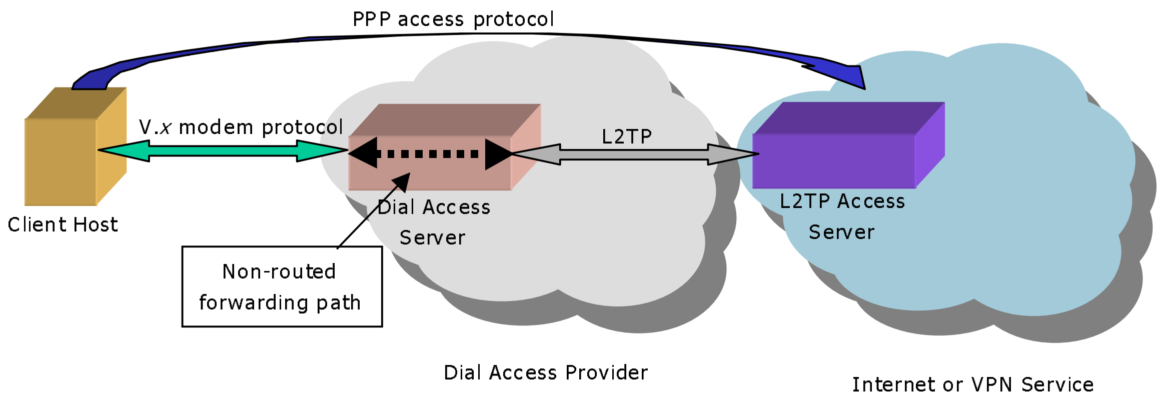


Figure 8

PPTP, as a “voluntary” tunneling model, on the other hand, allows end-systems (e.g. desktop computers) to configure and establish individual discrete point-to-point tunnels to arbitrarily located PPTP servers, without the intermediate NAS participating in the PPTP negotiation and tunnel establishment. In this scenario, a dial-in subscriber dials into a NAS, however, the PPP session is terminated on the NAS as in the traditional PPP model. The subsequent PPTP session is then established between the client end-system and any arbitrary upstream PPTP server that the client desires to connect to, given that it can reached via traditional routing information, and that the user has been granted the appropriate privileges on the PPTP server [Figure 9].

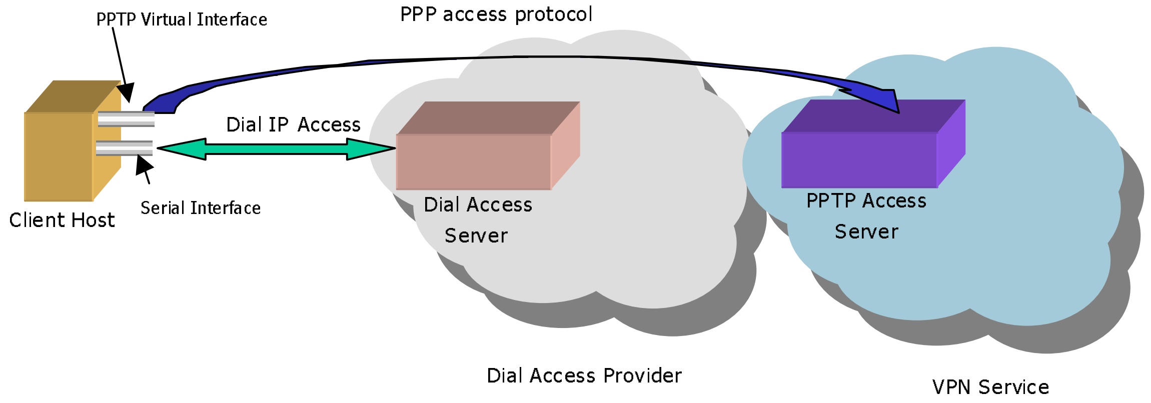


Figure 9

While L2TP and PPTP may sound extraordinarily similar, there are subtle differences which deserve further examination. The applicability of either protocol is very much dependent on what problem is being addressed. It is also about control – who has it, and why it is needed. It also depends very much on how each protocol implementation is deployed – in either the “voluntary” or “compulsory” tunneling models.

With PPTP in a “voluntary” tunneling implementation, the dial-in user has the ability to choose the PPTP tunnel destination after the initial PPP negotiation. This is important if the tunnel destination will change frequently, and no modifications are needed by mechanics in the transit path. It is also a significant advantage that the PPTP tunnels are transparent to the service provider, and no advance configuration is required between the NAS operator and the overlay dial access VPN. In such a case, the service provider does not house the PPTP server, and simply passes the PPTP traffic along with the same processing and forwarding policies as all other IP traffic. In fact, this could indeed be considered a benefit, since configuration and support of a tunneling mechanism within the service provider network would be one less thing that the service provider has to operationally manage, and the PPTP tunnel can transparently span multiple service providers without any explicit configuration. However, the economic downside to this for the service provider, of course, is that a “VPDN-enabled” network service could be marketed and yield an additional source of revenue. From the subscriber’s perspective, this is a “win-win” situation, since he is not reliant on the upstream service provider to deliver the VPDN service – at least no more than he is for basic IP-level connectivity. The other “win” is that the subscriber does not have to pay a higher subscription fee for a VPN service. Of course, the situation changes dramatically if the service provider houses the PPTP servers, or if the subscriber resides within a subnetwork in which the parent organization wants the network to make the decision concerning where tunnels are terminated. The major characterization of PPTP-based VPDN is one of a roaming client base, where the clients of the VPDN use a local connection to the public Internet data network, and then overlay a private data tunnel from the client's system to the desired remote service point. Another perspective is to view this approach as "on-demand" VPDN virtual circuits.

With L2TP in a “compulsory” tunneling implementation, the service provider controls where the PPP session is terminated. This can be extremely important in situations where the service provider to whom the subscriber is actually dialing into (let’s call it the “modem pool provider” network) must transparently hand off the subscriber’s PPP session to another network (let’s call this network the “content provider”). To the subscriber, it appears as though he is directly attached to the content provider’s network, when in fact he has been passed transparently through the modem pool provider’s network to the service to which he is subscribed. Very large content providers, for instance, may outsource the provisioning and maintenance of thousands of modem ports to a third-party provider, who in turn agrees to pass the subscriber’s traffic back to the content provider. This is generally called “wholesale dial.” The major motivation for such L2TP-based wholesale dial lies in the typical architecture of the PSTN network. The PSTN network is typically constructed in a hierarchical fashion, where a local PSTN exchange directly connects a set of PSTN subscribers, which is in turn connected via a trunk bearer to a central office or metropolitan exchange, which may be connected to a larger regional office or major exchange. A very efficient means of terminating large volumes of data PSTN calls is to use a single common call termination point within the local or central exchange to terminate all local data PSTN calls, and then hand the call data over to a client service provider using high volume data transmission services. This cost efficiencies that can result from this architecture form a large part of the motivation for such L2TP-based VPDN's, so a broad characterization of the demand for this style of VPDN can be characterized as a wholesale/retail dial access structure. Another perspective is to view this approach as "static" VPDN access.

Of course, if all subscribers connecting to the modem pool provider’s network are destined for the same content provider, then there are certainly easier ways to hand this traffic off to the content provider’s network – such as simply aggregating all of the traffic and handing the content provider a “big fat pipe.” However, in situations where the modem pool provider is providing a wholesale dial service for multiple upstream “next-hop” networks, the methods of determining how each subscriber’s traffic needs to be forwarded are somewhat limited. Packet forwarding decisions could be made at the NAS based on the source address of the dial-up subscriber’s computer, allowing for traffic to be forwarded along the appropriate path to its ultimate destination, providing a virtual connection, per se. However, the use of assigning static IP addresses to dial-in subscribers is highly discouraged due to the inefficiencies in IP address utilization policies, and the critical success of DHCP (Dynamic Host Configuration Protocol) **[12]** has made static IP address allocation to dial-up subscribers essentially a relic of earlier days.

There are, however, some serious scaling concerns in deploying a large-scale L2TP network, which revolve around the issue of whether large numbers of tunnels can actually be supported with little or no network performance impact. Since there have been no large-scale deployments of this technology to date, there is no empirical evidence to support or invalidate these concerns.

In some cases, however, appearances are everything – some content providers do not wish for their subscribers to know that when they connect to their service, that they have instead been connected to another service provider’s network, and then passed along ultimately to the service to which they are subscribed. In other cases, it is merely designed to be a matter of convenience, such that subscribers do not need to log into a device more than once.

Regrettably, the L2TP draft does not detail all possible implementations or deployment scenarios for the protocol. The basic deployment scenario is quite brief when compared to the rest of the document, and is arguably biased towards the "compulsory" tunneling model. Nonetheless, there are implementations of L2TP which follow the voluntary tunneling model. To the best of our knowledge, there has never been any intent to exclude this model of operation. We have also been told that recently, at various interoperability workshops, there have several different implementations of a voluntary L2TP client. There is nothing in the L2F protocol that would prohibit deploying it in a voluntary tunneling manner, just that it (to date) has not been widely implemented. Further, PPTP has been also been deployed using the compulsory model in a couple of specific vendor implementations.

In summary, the argument of whether PPTP or L2TP is more appropriate for deployment in a VPDN depends on whether the determination is made that control needs to lie with the service provider, or with the subscriber. Indeed, the difference can be characterized as to the client of the VPN, where the L2TP model is one of a “wholesale” access provider who has a number of configured client service providers who appear as VPN’s on the common dial access system, while the PPTP model is a one of distributed private access where the client is an individual end-user and the VPN structure is that of end-to-end tunnels. One might also suggest that the difference is also a matter of economics, since the L2TP model allows the service provider to actually provide a “value added” service, beyond basic IP-level connectivity, and charge their subscribers accordingly for the privilege of using it, thus creating new revenue streams, whereas the PPTP model enables distributed reach of the VPN at a much more atomic level, enabling corporate VPN’s to extend access capabilities without the need for explicit service contracts with a multitude of network access providers.

3.1.3 Network Layer Encryption

Encryption technologies are extremely effective in providing the segmentation and virtualization required for VPN connectivity, and can be deployed at almost any layer of the protocol stack. The evolving standard for network layer encryption in the Internet is IPSec (IP Security) **[13]**. While the IPSec architecture and its associated protocols are being finalized in the IETF (Internet Engineering Task Force) **[14]**, there is relatively little network layer encryption being done in the Internet today. However, there are some vendor proprietary solutions which are currently in use.

While IPSec has yet to be deployed in any significant volume, it is worthwhile to review the two methods in which network layer encryption is predominantly implemented. The most secure method for network layer encryption to be implemented is end-to-end, between participating hosts. This allows for the highest level of security. The alternative is more commonly referred to as "tunnel mode," whereas the encryption is only performed between intermediate devices (routers), and traffic between the end-system and the first-hop router is in plaintext. This is considerably less secure, since traffic intercepted in transit between the first-hop router and the end-system could be compromised. Where the VPN architecture is based on tunnels, the addition of encryption to the tunnel still leaves the tunnel ingress and egress points vulnerable, since these points are logically part of the host network as well as being part of the unencrypted VPN network. Any corruption of the operation, or interception of traffic in the clear, at these points will compromise the privacy of the private network.

In the end-to-end encryption scheme, VPN granularity is to the individual end-system level. In the tunnel mode scheme, the VPN granularity is to the subnetwork level. Traffic which transits the encrypted links between participating routers, however, is considered secure.

Network layer encryption, to include IPSec, is merely a subset of what a VPN is.

3.2 Link-Layer VPN's

One of the most straightforward methods of constructing VPN's is to use the transmission systems and networking platforms for the physical and link-layer connectivity, yet still be able to build discrete networks at the network layer. A link-layer VPN is intended to be a close (or preferably exact) functional analogy to a conventional private data network.

3.2.1 ATM and Frame Relay Virtual Connections

A conventional private data network uses a combination of dedicated circuits from a public carrier, together with an additional private communications infrastructure, to construct a network which is completely self-contained. Where the private data network exists within private premises, the network generally uses a dedicated private wiring plant to carry the VPN. Where the private data network extends outside the private boundary the dedicated circuits, it is typically provisioned for a larger public communications infrastructure using some form of time-division and/or frequency-division multiplexing to create the dedicated circuit. The essential characteristic of such circuits is the synchronization of the data clock, such that the sender and receiver pass data at a clocking rate which is fixed by the capacity of the dedicated circuit.

A link-layer VPN attempts to maintain the critical elements of this self-contained functionality, while achieving economies of scale and operation by utilizing a common switched public network infrastructure. Thus, a collection of VPN's may share the same infrastructure for connectivity, and share the same switching elements within the interior of the network, but explicitly must have no visibility, either direct or inferred, of one another. Generally, these "networks" operate at Layer 3 (the network layer) or higher in the OSI reference model, and the "infrastructure" itself commonly consists of either a Frame Relay or ATM network [Figure 10]. The essential difference here between this architecture of virtual circuits and that of dedicated circuits is that there is now no synchronized data clock shared by the sender and receiver, nor necessarily is there a dedicated transmission path which is assigned from the underlying common host network. The sender generally has no a priori knowledge of the available capacity of the virtual circuit, as the capacity varies in response to the total demand placed on it by other simultaneous transmission and switching activity. Instead, the sender and receiver can use adaptive clocking of data, where the sender can adjust the transmission rate to match the requirements of the application and any signaling received from the network and the receiver. It should be noted that a dedicated circuit system using synchronized clocking cannot be oversubscribed, while the virtual circuit architecture (where the sender does not have a synchronized end-to-end data clock) can indeed be oversubscribed. It is the behavior of the network when it transitions into this over-subscribed state which is of most interest here.

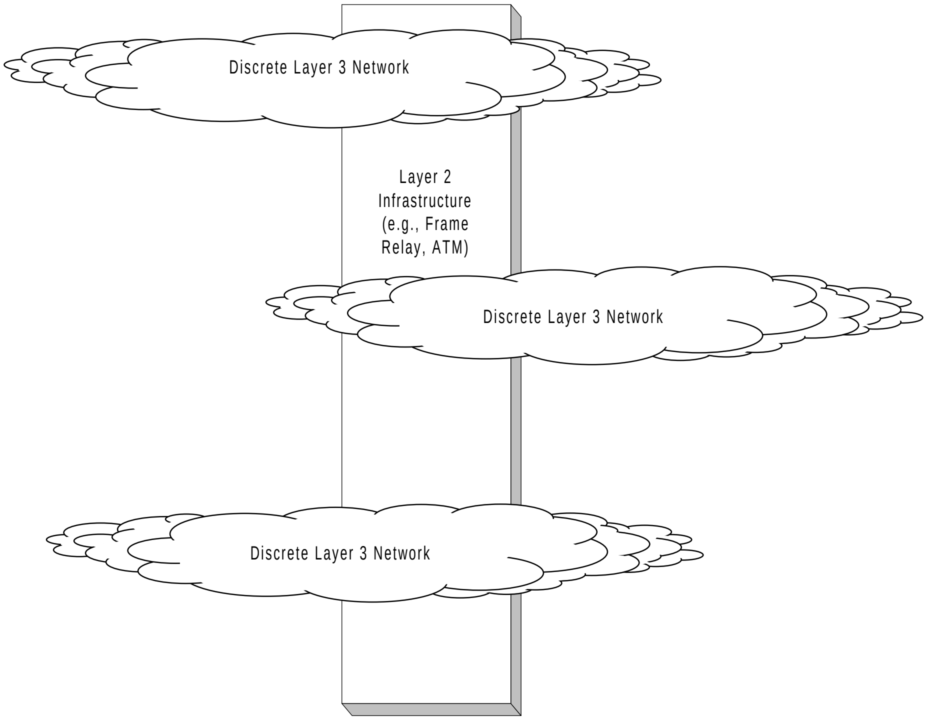


Figure 10

One of the nice things about a public switched Wide Area Network which provides virtual circuits is that it can be extraordinarily flexible. Most subscribers to Frame Relay services, for example, have subscribed to the service for economic reasons – it is cheap, and the service provider will usually throw in an SLA (Service Level Agreement) that "guarantees" some percentage of frame delivery in the Frame Relay network itself.

> *The remarkable thing about this service offering is that the customer will generally be completely unaware whether the service provider can actually deliver the contracted service at all times and under all possible conditions. The layer 2 technology is not a synchronized clock blocking technology where each new service flow is accepted or denied based on the absolute ability to meet the associated resource demands. Each additional service flow is accepted into the network and carried on a best effort basis. Admission functions provide the network with a simple two level discard mechanism which allows a graduated response to instances of overload, however once the point of saturated overload is reached within the network all services will be affected.*

This brings up several other important issues. The first one which is worthy of mention is concerning the engineering practices of the Frame Relay service provider. If the Frame Relay network is poorly engineered, and is constantly congested, then this may very well be reflected in the service quality delivered to the subscribers. Frame Relay uses a notion of a per-virtual circuit CIR (Committed Information Rate). This is an ingress function associated with Frame Relay which checks the ingress traffic rate against the CIR. Frames which exceed this base rate are still accepted by the Frame Relay network, but are marked as DE (Discard Eligible). Because the network can be over-subscribed there will be times where the data rate within a switch will exceed both the egress transmission rate and the local buffer storage. At this point in time, the switch will begin to discard data frames, and will initially do so for frames with the DE marker present. This is essentially a two level discard precedence architecture. It is an administrative decision by the service provider as to the relative levels of provisioning of core transmission and switching capacity, and the ratio of network ingress capacity used by subscribers. The associated CIR's of the virtual circuits against this core capacity is a critical determinant of the resultant deliverable quality of performance of the network and the layered VPN's. For example, at least one successful (and popular) Frame Relay service provider provides an economically attractive Frame Relay service which permits 0 CIR on PVC's, combined with an SLA (Service Level Agreement) ensuring that at least 99.8% of all frame-level traffic presented to the Frame Relay network will be delivered successfully. If this SLA is not met, then the subscriber's monthly service fee will be appropriately prorated the following month. The Frame Relay service provider provides frame level statistics to each subscriber every month, culled from the Frame Relay switches, to measure the effectiveness of this SLA "guarantee." This particular Frame Relay service provider is remarkably successful in honoring the SLA's due to the fact that they conduct ongoing network capacity management each week, provisioning new trunks between Frame Relay switches when trunk utilization exceeds 50%, and ensuring that trunk utilization never exceed 75%. In this fashion, traffic on PVC's with a 0 CIR can generally avoid being discarded in the Frame Relay network.

Having said that, the flexibility of PVC's allow discrete VPN's to be constructed across a single Frame Relay network. And in many instances, this scenario lends itself to situations where the Frame Relay network provider also manages each discrete VPN via a telemetry PVC. Several service providers have *Managed Network Services* (MNS) which provide exactly this type of service.

While the example above revolves around the use of Frame Relay as a link-layer mechanism, essentially the same type of VPN mechanics hold true for ATM. As with Frame Relay, there is no data clock synchronization between the sender, the host network, and the receiver. As with Frame Relay, the sender's traffic is passed into the ATM network via an ingress function, which can mark cells with a CLP (Cell Loss Priority) indication. And, as with Frame Relay, where a switch experiences congestion, the switch will attempt to discard marked (CLP) cells as the primary load shedding mechanism, but if this is inadequate, the network must shed other cells which are not so marked. Once again, the quality of the service depends on proper capacity engineering of the network, and there is no inherent guarantee of service quality as an attribute of the technology itself.

The generic observation is that the engineering of Frame Relay and ATM common carriage data networks are typically very conservative. The inherent capabilities of both of these link layer architectures do not permit a wide set of selective responses to network overload, so that in order for the network to service the broadest spectrum of potential VPN clients, the network must provide high quality carriage and very limited instances of any form of overload. In this way, such networks are typically positioned as a high quality alternative to dedicated circuit private network architectures, which are intended to operate in a very similar manner (and, not surprisingly, are generally priced as a premium VPN offering). Technically, the architecture of link layer VPN's is almost indistinguishable from the dedicated circuit private data network – the network can support multiple protocols, private addressing and routing schemes, as the essential significant difference between a dedicated circuit and a virtual link layer circuit is the absence of synchronized clocking between the sender and receiver. In all other aspects, the networks are very similar.

There are certainly scaling concerns with these approaches to constructing VPN's, especially with regards configuration management of provisioning new VC's and routing issues. Configuration management still tends to be one of the sore points in VPN management – adding new subscribers and new VPN's to the network requires VC path construction and provisioning, a tedium which requires ongoing administrative attention by the VPN provider. Also, as we have already mentioned, full mesh networks encounter scaling problems, which in turn results in VPN's being constructed in which partial meshing is done to avoid certain scaling limitations. The liabilities in these cases need to examined closely, since partial meshing of the underlying link layer network may contribute to suboptimal routing (e.g., extra hops due to hub-and-spoke issues, redirects).

The problems described in the above paragraph applies to all types of VPN's build on the "overlay" model – not just ATM and Frame Relay. Specifically, the problem applies to GRE tunnels, as well.

3.2.2 MPOA (Multi Protocol Over ATM) and The "Virtual Router" Concept

Another unique model of constructing VPN's is the use of MPOA **[15]** , or Multi Protocol Over ATM, which uses RFC1483 encapsulation **[16]**. This VPN approach is similar to other "cut-through" mechanisms where a particular switched link-layer is used to enable all "Layer 3" egress points to be only a single hop away from one another.

In this model, the edge routers determine the forwarding path in the ATM switched network, since they have the ability to determine which egress point packets need to be forwarded to. Once a network layer reachability decision is made, the edge router forwards the packet onto a Virtual Connection (VC) designated for a particular egress router. However, since the egress routers cannot ARP (Address Resolution Protocol) for destination address across the cloud, they must rely on a external server for address resolution (ATM address to IP address).

The first concern here is a sole reliance on ATM – this particular model does not encompass any other types of data-link layer technologies, rendering the technology less than desirable in a hybrid network. While this may have some domain of applicability within a homogenous ATM environment, when looking at a broader VPN environment which may encompass a number of link-layer technologies, this approach offers little benefit to the VPN provider.

Secondly, there are serious scaling concerns regarding full mesh models of connectivity, where suboptimal network layer routing may result due to cut-through. And the reliance on address resolution servers to support the ARP function within the dynamic circuit framework brings this model to the point of excessive complexity.

The advantage of the MPOA approach is the use of dynamic circuits rather than more cumbersome statically configured models. The traditional approach to supporting private networks involves extensive manual design and operational support to ensure that the various configurations on each of the bearer switching elements are mutually consistent. The desire within the MPOA environment is to attempt to use MPOA to govern the creation of dynamically controlled edge-to-edge ATM VC's. While this may offer the carrier operator some advantages in reduced design and operational overhead, it does require the uniform availability of ATM, and in many heterogeneous environments this is simply not the case.

In summary, this is another overlay model, with some serious issues regarding scale.

There have also been "peer" VPN models introduced which allow the egress nodes to maintain separate routing tables – one for each VPN – effectively allowing separate forwarding decisions to be made within each node for each distinctive VPN. While this is an interesting model, this also introduces concerns about approaches where each edge device runs a separate routing process and maintains a separate routing information base (RIB, or routing table) process for each VPN community of interest. It is also necessary to note that the "virtual router" concept requires some form of packet labeling, either within the header or via some lightweight encapsulation mechanism, in order for the switch to be able to match the packet against the correct VPN routing table. If this is a global label, the issue of operational integrity is a relevant concern, while if it is a local label, the concept of label switching and maintenance of edge-to-edge label switching contexts is also a requirement.

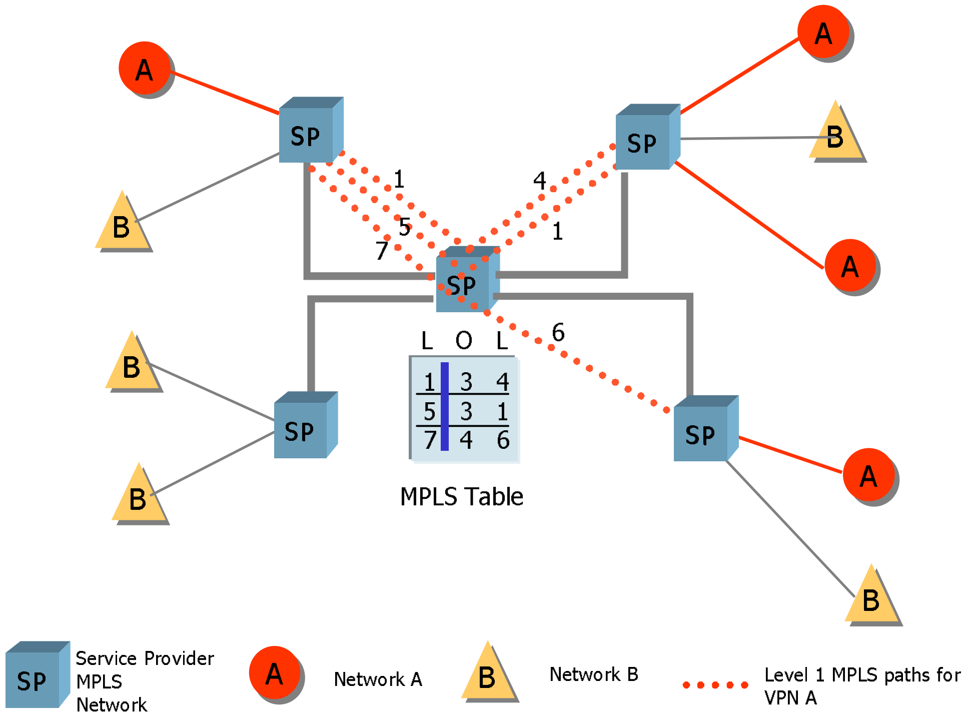
Among the scaling concerns are issues regarding the number of supported VPN's in relation to the computational requirements, and stability of routing system within each VPN (i.e. instability in one VPN affecting the performance of other VPN's served by the same device). The aggregate scaling demands of this model are also not inconsiderable. Given a change in the underlying physical or link-layer topology, the consequent requirement to process the routing update in a per-VPN basis becomes a significant challenge. Use of distance vector protocols to manage the routing tables would cause a corresponding sudden surge in traffic load, which grows in direct proportion to the number of supported VPN's. The use of link-state routing protocols would require the consequent link-state calculation to be repeated for each VPN, causing the router to be limited by available CPU capacity.

3.2.3 Multi-Protocol Label Switching

One method of addressing these scaling issues is to use VPN labels within a single routing environment, in the same way that packet labels are necessary to activate the correct per-VPN routing table. If local label switching is used, then this is effectively the architecture of an MPLS VPN. It is perhaps no surprise that when presented with two basic approaches to the architecture of the VPN – that of the use of network layer routing structures and per-packet switching, and the use of link-layer circuits and per-flow switching – that the industry would devise a hybrid architecture which attempts to combine aspects of these two approaches. This hybrid architecture is referred to as *Multi-Protocol Label Switching* (MPLS) **[17, 18]**.

The architectural concepts used by MPLS are generic enough to allow it to operate as a peer VPN model for switching technology for a variety of link-layer technologies, and in heterogeneous layer 2 transmission and switching environments. MPLS requires protocol-based routing functionality in the intermediate devices, and operates by making the inter-switch transport infrastructure visible to the routing. In the case of IP over ATM, each ATM bearer link becomes visible as an IP link, and the ATM switches are augmented with IP routing functionality. IP routing is used to select a transit path across the network, and these transit paths are marked with a sequence of labels which can be thought of as locally-defined forwarding path indicators. MPLS itself is performed using a label swapping forwarding structure. Packets entering the MPLS environment are assigned a local label and an outbound interface based on a local forwarding decision. The local label is attached to the packet via a lightweight encapsulation mechanism. At the next MPLS switch, the forwarding decision is based on the incoming label value, where the incoming label determines the next hop interface and next hop label, using a local forwarding table indexed by label. This lookup table is generated by a combination of the locally used IP routing protocol, together with a label distribution protocol, which creates end-to-end transit paths through the network for each IP destination. It is not our intention to go into any amount of detail on the MPLS architecture here, apart from noting the each MPLS switch uses a label-indexed forwarding table, where the attached label of an incoming packet determines the next hop interface and the corresponding outgoing label.

The major observation here is that this lightweight encapsulation, together with the associated notion of boundary-determined transit paths provides much of the necessary mechanisms for the support of VPN structures **[19]**. MPLS VPN's have not one, but three key ingredients – (1) constrained distribution of routing information as a way to form VPN's and control inter-VPN connectivity, (2) the use of VPN-ID's, and specifically the concatenation of VPN-ID's with IP addresses to turn (potentially) non-unique addresses into unique ones, and (3) the use of label switching (MPLS) to provide forwarding along the routes constructed via (1) and (2). The generic architecture of deployment is that of a label switched common host network and a collection of VPN environments which use label-defined virtual circuits on an edge-to-edge basis across the MPLS environment. An example is indicated in [Figure 9], where a table is illustrated indicating how MPLS virtual circuits are constructed.



L	O	L
1	3	4
5	3	1
7	4	6

Figure 9

A number of approaches are possible to support VPN's within an MPLS environment. In the base MPLS architecture, the label applied to a packet on ingress to the MPLS environment effectively determines the selection of the egress router, as the sequence of label switches defines an edge-to-edge virtual path. The extension to the MPLS local label hop-by-hop architecture is the notion of a per-VPN global identifier (or *Closed User Group* identifier as defined in **[19]**) which effectively is used within an edge-to-edge context. This global identifier could be assigned on ingress, and is then used as an index into a per-VPN routing table to determine the initial switch label. On egress from the MPLS environment, the CUG identifier would be used again as an index into a per-VPN global identifier table to undertake next hop selection. Routing protocols in such an environment need to carry the CUG identifier to trigger per-VPN routing contexts, and a number of suggestions are noted in **[19]** as to how this could be achieved.

It should be stressed that MPLS itself, as well as the direction of VPN support using MPLS environments are still within the area of active research, development, and subsequent standardization within the forum of the IETF, so that this approach to VPN support is still somewhat speculative in nature.

3.2.4 Link-Layer Encryption

As mentioned previously, encryption technologies are extremely effective in providing the segmentation and virtualization required for VPN connectivity, and can be deployed at almost any layer of the protocol stack. There are no industry standards, per se, for link-layer encryption, thus all link-layer encryption solutions are generally vendor specific and require special encryption hardware.

While this can avoid the complexities of having to deal with encryption schemes at higher layers of the protocol stack, it can be economically prohibitive, depending on the solution adopted. In vendor proprietary solutions, multivendor interoperability is certainly a genuine concern.

3.3 Transport and Application Layer VPN's

While VPN's can certainly be implemented at the transport and application layers of the protocol stack, it is not very common. The most prevalent method of providing virtualization at these layers is to use encryption services at either layer. For example, encrypted e-mail transactions, or perhaps authenticated DNS (Domain Name System) zone transfers between different administrative name servers, as described in DNSSec (Domain Name System Security) **[20]**.

There is some interesting, and perhaps extremely significant, work being done in the IETF to define a Transport Layer Security protocol (TLS) **[21]**, which would provide privacy and data integrity between two communicating applications. The TLS protocol, once finalized and deployed, would allow applications to communicate in a fashion that is designed to prevent eavesdropping, tampering, or message forgery.

It is unknown at the time of this writing, however, how long it may be before this work is finalized, or if it will be embraced by the networking community as a whole once the protocol specification is completed.

The significance of a “standard” transport layer security protocol, however, is that once implemented, it could provide a highly granular method for the virtualizing communications in TCP/IP networks, thus making VPN’s a pervasive commodity, and native to all desktop computing platforms.

4. Non-IP VPN’s

While this paper has focused on TCP/IP and VPN’s, it is recognized that multiprotocol networks may also have requirements for VPN’s. Most of the same techniques previously discussed here can also be applied to multiprotocol networks, with a couple of obvious exceptions – a number of the techniques described here are solely and specifically tailored for TCP/IP protocols.

Controlled route leaking is not suitable for a heterogeneous VPN protocol environment, in that it is necessary to support all protocols within the common host network. GRE tunnels, on the other hand, are constructed at the network layer in the TCP/IP protocol stack, however, most *routable* multiprotocol traffic can be transported across GRE tunnels (e.g. IPX, AppleTalk). Similarly the VPDN architectures of L2TP and PPTP both provide a PPP end-to-end transport mechanism which can allow per-VPN protocols to be supported, with the caveat that it is a PPP supported protocol in the first place.

The reverse of heterogeneous VPN protocol support is also a VPN requirement in some cases, where a single VPN is to be layered above a heterogeneous collection of host networks. The most pervasive method of constructing VPN’s in multiprotocol networks is to rely upon application layer encryption, and as such, are generally vendor proprietary, although some would contend that one of the most pervasive examples of this approach was the mainstay of the emergent Internet in the 1970's and 1980's – that of the UUCP network, which was (and remains) an open technology.

5. Quality of Service Considerations

As well as creating a segregated address environment to allow private communications, there is also the expectation that the VPN environment will be in a position to support a set of service levels. Such per-VPN service levels may be specified either in terms of a defined service level that the VPN can rely upon at all times, or in terms of a level of differentiation that the VPN can draw upon the common platform resource with some level of priority of resource allocation.

Using dedicated leased circuits a private network can establish fixed resource levels available to it under all conditions. Using a shared switched infrastructure, such as Frame Relay virtual circuits or ATM virtual connections, there is a similar intent to provide a quantified service level to the VPN through the characteristics of the virtual circuits used to implement the VPN.

When the VPN is moved away from such a circuit-based switching environment to that of a general Internet platform, is it possible for the Internet Service Provider to offer the VPN a comparable service level which attempts to quantify (and possibly guarantee) the level of resources which the VPN can draw upon from the underlying host Internet?

This is an area which is undergoing rapid evolution, and much of this area remains within the realm of speculation rather than a more concrete discussion about the relative merits of various Internet Quality of Service (QoS) mechanisms. Efforts within the Integrated Services Working Group of the IETF has resulted in a set of specifications for the support of guaranteed and controlled load end-to-end traffic profiles using a mechanism which loads per-flow state into the switching elements of the network **[22, 23]**. There are a number of caveats regarding the use of these mechanisms, in particular relating to the ability to support the number of flows which will be encountered on the public Internet **[24]**. Such caveats tend to suggest that these mechanisms will not be the which is ultimately adopted to support service levels for VPN’s in very large networking environments.

If the scale of the public Internet environment does not readily support the imposition of per-flow state to support guarantees of service levels for VPN traffic flows, the alternative query is whether this environment could support a more relaxed specification of a differentiated service level for overlay VPN traffic. Here, the story appears to offer more potential, given that differentiated service support does not necessarily imply the requirement for per-flow state, so that stateless service differentiation mechanisms can be deployed which offer greater levels of support for scaling the differentiated service **[25]**. However, the precise nature of these differentiated service mechanisms, and their capability to be translated to specific service levels to support overlay VPN traffic flows still remain in the area of future activity and research.

6. Conclusions

So what ***is*** a Virtual Private Network? As we have discussed, a VPN can take several forms. A VPN can be between two end-systems, or it can be between two or more networks. A VPN can be built using tunnels or encryption (at essentially any layer of the protocol stack), or both, or alternatively constructed using MPLS or one of the "virtual router" methods. A VPN can consist of networks connected to a service provider's network by leased lines, Frame Relay, or ATM, or a VPN can consist of dial-up subscribers connecting to centralized services, or other dial-up subscribers.

The pertinent conclusion here is that while a VPN can take many forms, there are some basic common problems that a VPN is built to solve, which can be listed as virtualization of services and segregation of communications to a closed community of interest, while simultaneously exploiting the financial opportunity of economies of scale of the underlying common host communications system.

To borrow a popular networking axiom, "When all you have is a hammer, everything looks like a nail." Every organization has its own problem that it must solve, and each of the tools mentioned within this paper can be used to construct a certain type of VPN to address a particular set of functional objectives. There is more than single "hammer" to address these problems, and network engineers should be cognizant of the fact that VPN's are an area where many people use the term generically – there is a broad problem set with equally as many possible solutions. Each solution has a number of strengths and also a number of weaknesses and vulnerabilities. There is no single mechanism for VPN's which will supplant all others in the months and years to come, but instead we will continue to see a diversity of technology choices in this area of VPN support.

7. Acknowledgments

Thanks to Yakov Rekhter (Cisco Systems), Eric Rosen (Cisco Systems), and W. Mark Townsley (Cisco Systems) for their input and constructive criticism.

8. References

[1] Wired Magazine, February 1998, Wired's "Hype List - Deflating this month's overblown memes," page 80. Ironically, number 1 on the *Hype List* is Virtual Private Networks with a life expectancy of 18 months.

[2] "The New Hacker's Dictionary," Third Edition. Compiled by Eric S. Raymond, published by MIT Press, 1993. The Jargon File online: http://www.ccil.org/jargon/

[3] Webster's Revised Unabridged Dictionary (1913). Hypertext Webster Gateway: http://work.ucsd.edu:5141/cgi-bin/http_webster

[4] "Architecture of the Multi-Modal Organizational Research and Production Heterogeneous Network (MORPHnet)," R. Aiken, R. Carlson, I. Foster, T. Kuhfuss, R. Stevens, L. Winkler, January 1997.

[5] RFC1997, "BGP Communities Attribute," R. Chandra, P. Traina, T, Li, August 1996. RFC1998, "An Application of the BGP Community Attribute in Multi-home Routing," E. Chen, T. Bates, August 1996.

[6] RFC1701, "Generic Routing Encapsulation," S. Hanks, T. Li, D. Farinacci, P. Traina, October 1994. Also RFC1702, "Generic Routing Encapsulation over IPv4 networks," S. Hanks, T. Li, D. Farinacci, P. Traina, October 1994.

[7] "Layer Two Tunneling Protocol 'L2TP'," draft-ietf-pppext-l2tp-10.txt, A. Valencia, K. Hamzeh, A. Rubens, T. Kolar, M. Littlewood, W. M. Townsley, J. Taarud, G. S. Pall, B. Palter, W. Verthein,, March 1998.

[8] "Point-to-Point Tunneling Protocol – PPTP," draft-ietf-pppext-pptp-02.txt, K. Hamzeh, G. Singh Pall, W. Verthein, J. Taarud, W. A. Little, July 1997. See also: http://www.microsoft.com/backoffice/communications/morepptp.htm

[9] RFC1075, "Distance Vector Multicast Routing Protocol," D. Waitzman, C. Partridge, S. Deering, November 1988. For historical purposes, see also ftp://ftp.isi.edu/mbone/faq.txt

[10] "NBMA Next Hop Resolution Protocol (NHRP)," draft-ietf-rolc-nhrp-15.txt, J. Luciani, D. Katz, D. Piscitello, B. Cole, N. Doraswamy , February 1998.

[11] "Layer Two Forwarding (Protocol) 'L2F'," draft-valencia-l2f-00.txt, A. Valencia, M. Littlewood, T. Kolar, October 1997.

[12] RFC2131, "Dynamic Host Configuration Protocol," R. Droms, March 1997.

[13] IPSec is actually an architecture – a collection of protocols, authentication, and encryption mechanisms. The IPSec security architecture is described in detail in “Security Architecture for the Internet Protocol,” draft-ietf-ipsec-arch-sec-04.txt, S. Kent, R. Atkinson, March 1998. Additional information on IPSec can be found on the IETF IPSec home page, located at http://www.ietf.org/html.charters/ipsec-charter.html

[14] The Internet Engineering Task Force – http://www.ietf.org/

[15] RFC1483, “Multiprotocol Encapsulation over ATM Adaptation Layer 5,” J. Heinanen, July 1993.

[16] The ATM Forum, “Multi-Protocol Over ATM Specification v1.0,” af-mpoa-0087.000, July 1997.

[17] "A Framework for Multiprotocol Label Switching," draft-ietf-mpls-framework-02.txt, R. Callon, P. Doolan, N. Feldman, A. Fredette, G. Swallow, A. Viswanathan, November 1997.

[18] "A Proposed Architecture for MPLS," draft-ietf-mpls-arch-01.txt, E. Rosen, A. Viswanathan, R. Callon, March 1998.

[19] "VPN Support for MPLS," draft-heinanen-mpls-vpn-01.txt, J. Heinanen, E. Rosen, March 1998.

[20] RFC2065, “Domain Name System Security Extensions,” D. Eastlake, C. Kaufman, January 1997. For further information regarding DNSSec, see: http://www.ietf.org/html.charters/dnssec-charter.html

[21] “The TLS Protocol – Version 1.0,” draft-ietf-tls-protocol-05.txt, T. Dierks, C. Allen, November 1997. For more information on the IETF TLS working group, see http://www.ietf.org/html.charters/tls-charter.html.

[22] RFC2211, "Specification of the Controlled-Load Network Element Service,", J. Wroclawski, September 1997.

[23] RFC2212, ":Specification of Guaranteed Quality of Service,", S. Shenker, C. Partridge, R. Guerin, September 1997.

[24] RFC2208, "Resource ReSerVation Protocol (RSVP) Version 1 – Applicability Statement, Some Guidelines on Deployment," A. Mankin, F. Baker, S. Bradner, M. O'Dell, A. Romanow, A. Weinrib, L. Zhang, September 1997

[25] " Differentiated Services Operational Model and Definitions," draft-nichols-dsopdef-00.txt, K, Nichols, S. Blake (editors), February 1998.