

How PNA Works: The Future of Networking

An Overview

John Day

4 Feb 2010

Architecture is maximizing the invariances and minimizing the discontinuities.

Executive Summary

We have made a fundamental breakthrough in networking architecture that solves the major problems of Internet infrastructure while enabling critical new capabilities. No other approach has the simplicity and breadth with major practical implications. The breakthroughs arise from four major innovations that extend conventional Internet architecture: separating mechanism and policy in protocols, unifying connectionless and connection-mode communication and topological addresses, embedded in a Distributed Interprocess Communication model. These innovations mutually reinforce to yield a dramatic reduction in complexity, rather than the current piece meal approach which only increases complexity with each new problem attacked. Rather than the traditional architectures of 7, 5, or 4 layers, we have found that there is a single parameterized layer, really a distributed application that provides IPC, or a Distributed IPC Facility (DIF). PNA is compatible with conventional internetworks. There is no requirement for a “flag day;” adoption may be accomplished incrementally. Taken in combination, these innovations dramatically improve robustness, transmission in chaotic environments, performance, security, delivery of time-sensitive information, modes of operation (such as multicasting and multihoming), management complexity, engineering, and most importantly scaling.

Introduction

Beginning around the turn of the century, there was growing awareness that the “Internet was broken” that the Internet architecture was reaching its limits. While the best researchers have been working at solving the problem ever since, the effects of 20 years of group think have taken their toll and to date, they have come up dry. For some, by the mid-70s the experience of the ARPANET had uncovered fundamental questions that remain unanswered to this day. By the early 90s, I began to make significant breakthroughs on these issues, with major new insights that have continued. To make these breakthroughs, it was necessary to go back to basics and question what we thought we knew. Some of it was right; some were insights we hadn’t noticed; some were misconceptions that were barriers to a simpler approach.

The fundamental insight derives from the realization that networking is a distributed application that does InterProcess Communication (IPC). This yields a single recursive Distributed IPC Facility (DIF) that generalized to cover the breadth of networking. Such broad, deep results are only possible by reconsidering the fundamental structures of networking rather than taking an incremental approach. A complete explanation of how this new architecture works would take considerable space. This document is limited to a general overview. The book, **Patterns in Network Architecture, A Return to Fundamentals**, Prentice-Hall, 2008 provides a more complete exposition. We begin by describing the basic machinery of PNA, followed by a description of the new principles that define and motivate how it works. This is followed by an overview of the operational advantages of our approach. We close with a description of the status of the work.

1. Four Fundamental Innovations and The Recursive Layer

What we have uncovered about network architecture is based on four fundamental innovations:

- separating mechanism and policy in protocols,
- unifying connection-mode, connectionless, and multicast communication,
- a new approach to topological addresses, yielding
- a recursive distributed IPC Facility (DIF) that is repeated to create the full breadth of networking from the application level to the core network.

Also two little known but important results of previous work prove to be key:

- Richard Watson's proof that for synchronization it is necessary and sufficient to bound three timers. This leads to the separation of port allocation and synchronization, conflated in most protocols, and that all data transfer protocols are soft state. This not only leads to much simpler protocols but also has major security implications.
- The distinction between application process and application entity uncovered in the mid-80s by the OSI work.

The first two innovations along with revisiting the fundamentals of operating systems lead to the realization that a layer is a distributed application providing IPC, rather than the fixed hand-crafted layers of the conventional Internet. While this may at first appear obvious, its implications are far-reaching and far from obvious. The third of these innovations further leverages the capabilities of the DIF. These innovations solve multiple problems while yielding a dramatic reduction in complexity, instead of the increased complexity of the current incremental approach.

The Distributed IPC Facility

Our analysis extending IPC from a single system to a distributed system tells us that all networking layers provide the same basic functions, but they do so for different ranges of bandwidth and qualities of service. A layer is a Distributed InterProcess Communication (IPC) Facility composed of cooperating applications called IPC Processes.

Separating mechanism and policy reveals that an IPC Process is comprised of three sets of functions: **data transfer**, **data transfer control**, and **layer management**, characterized by progressively increasing cycle times decoupled through a state vector (or information base) which

match the complexity of the processing to the cycle time, i.e. simpler, faster; more complex slower. Precisely the structure, one wants to see for problems like this. (see Figure 1).

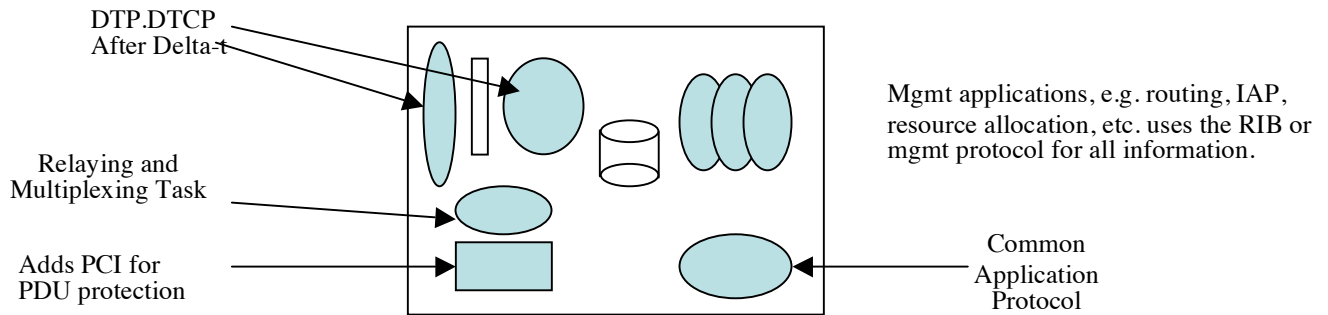


Figure 1. An IPC Process consist of data transfer (left) with a fast cycle time, decoupled through a state vector with data transfer control (middle) with a longer but still fast cycle time, both which interact with the much slower management tasks (right) through an information base.

Separating mechanism and policy also reveals that there is only one error and flow control protocol that cleaves naturally into data transfer and data transfer control decoupled through a state vector. This protocol covers the entire range from UDP to TCP. Separately it has become clear that there is only one Application Protocol. There are only 6 operations that can be performed remotely: create/delete, read/write, and start/stop on objects. The variety of applications is in the objects being manipulated, not the protocol. This has the effect of making the difficult to change protocol very stable, while the more easily changed object models implement the value. This common distributed application protocol is used within the DIF for all layer management to provide enrollment and security management, port allocation and access control, QoS monitoring, flow management, and routing. Enrollment is the process of initializing the DIF and creating the minimal shared state necessary for IPC.

The scaling problems encountered in the Internet are due to its wide operating range, static structure and incomplete addressing architecture, i.e. it only has point of attachment addresses. These problems are readily overcome by repeating, or recursing, the DIF, as required depending on the range of bandwidth in the network: The greater the range, the more instances of the DIF, the more layers. At each level of recursion, traffic is multiplexed to form flows that can take advantage of the policies of the lower layer. This allows us to "divide and conquer" the management of an internet into tractable tasks, yielding dramatic improvements in operating efficiency. In practice, any one system would typically have 2 or 3 layers, but not all systems in the network would have all layers. (see Figure 2)

The closest similar technique in conventional internets is simple hand-crafted tunneling. However, tunneling merely encapsulates the data transfer protocol and the rest of the layer machinery is left in place. Hence, there are no advantages for scaling and many disadvantages in managing the resulting configuration. When a DIF is instantiated all of the machinery of the layer is also instantiated, yielding advantages for scaling and providing all of the machinery for managing the recursion. This is a key element of how PNA works and scales.

Using the DIF drastically reduces development and operational complexity. Once the single DIF has been created and tested, there are no other layers to test – only the combinations of the DIF with different policies. Security analysis is similarly simplified. Furthermore, changes can be made one parameter or policy at a time, allowing stepwise testing at minimal cost, rather than requiring a laborious and expensive process of specification, design, code writing and testing. And of course, the same stepwise process can be used in tuning and managing an operational network.

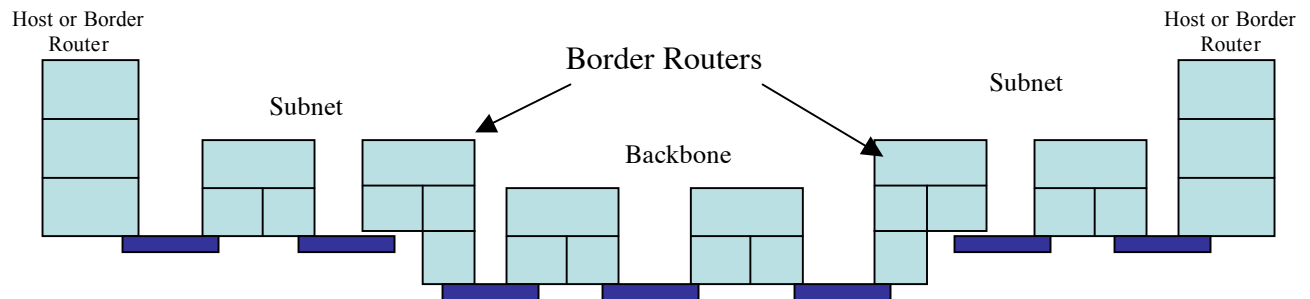


Figure 2. Routing occurs both in the DIFs at the subnet level and at the backbone. But due to the recursion the backbone appears as a single hop to the subnets.

In sum, the Distributed IPC Facility enables the recursive decomposition of the problems of the conventional Internet.

The integration of connection and connectionless networking, and the separation of mechanism and policy, are the two main concepts behind the creation of the DIF. They are described in the following two sections.

2. The Integration of Connection and Connectionless Networking

The dichotomy between connection and connectionless has impaired architectures for the last 25 years. Although both are useful in the appropriate environments, both have severe shortcomings outside those environments. Connection-based networks avoid congestion but are brittle to failure; while connectionless is resilient to failure but is prone to congestion. Attempts to integrate them have not met with much success. The two have been viewed traditionally as extremes of the amount of shared state required in data transfer protocols. That is, connectionless-mode protocols require very little state, and connection-mode protocols require much more state. The so-called “dumb network.” Integration of the two modes was especially difficult because there did not seem to be any cases between the two extremes.

What we discovered was quite the opposite, consider: Why are connection-based networks so brittle? When there is a failure no one knows what to do. All the necessary information for repairing the failure is elsewhere at the edge. On the other hand, with connectionless when there is a failure every router knows how to route the packet. Failures are not a problem. Connectionless is maximal state information, not minimal!! The dumb network is not so dumb! Using our insights the IPC model as a guide, we have discovered how to achieve this integration and to exploit it based on three observations:

- 1) Port-ids must be allocated for both connections and connectionless thus decoupling the choice from the interface, yielding a single API to the user for both capabilities;
- 2) as one moves down in the layers and in toward the backbone, traffic tends to become more dense and more connection-oriented; conversely, as one moves up in the layers and out toward the periphery, traffic becomes less dense and more connectionless; and
- 3) integration can be achieved by using the definition of connectionless to create a range of capabilities – rather than by using the implementation view based on the amount of shared state.

In conventional networks, an application¹ determines the choice of connection or connectionless. Because the two services have different interfaces, the implementation of the application must be different depending on which service it is using. In PNA, the user requests communications with a given quality of service – there is only one API. The DIF then determines which mechanism should be used based on the QoS the DIF is receiving from the lower DIFs and the QoS required by the request. This creates a single consistent interaction with the application for both connectionless and connection-mode.

Integrating connection and connectionless functionally within the DIF is achieved by going back to the definition of connectionless: the routing of each protocol data unit (PDU) is independent of the routing of any previous PDU, i.e. the probability that the routing is dependent on previous action is 0. Hence, a connection is when the routing of each PDU is determined by the routing of the previous PDU, i.e. the probability is 1. We can create a range of capabilities by simply allowing this probability to have values other than 0 or 1. This shifts the focus from the shared state of the data transfer protocols to the choices of resource allocation and routing and allows us to construct a range of routing behaviors that can vary between being more or less connectionless along a path. In essence, this means that connection-like services could use multiple paths as long as the requested QoS was maintained. Furthermore, the failure of a link would have little impact on a connection passing over it. Transfer would not have to halt while the connection was re-established as with traditional connection methods. The failure would cause a re-allocation of resources and a new routing of the traffic. Or, if the “connection” were using multiple paths, the remaining paths may pick up the slack with only a slight drop in QoS while the routing worked around the problem and brought the QoS back to normal. This flexibility is impossible to achieve in today’s Internet, without tedious, error-prone, manual configurations using multiple protocols not designed to work together.

We have also discovered that multicast and anycast are two forms of the same construct. An anycast address is the name of a set of addresses with a rule, such that when the name is referenced, the rule is evaluated and returns one member of the set that satisfies the rule. A multicast address is a set where the rule returns all members. We call the general form a “whatevercast” name, which is a set of addresses the returns whatever addresses satisfies the rule. The normal multicast API combines functions such that when they are properly disentangled, the application can use the same API for unicast (connectionless or connection) and whatevercast. Furthermore, specialized multicast protocols are not required, multicast distribution becomes a management function and multicast itself is integrated into the generation of forwarding tables. This is a huge simplification.

¹ Since a DIF is a distributed application, its elements are also applications and hence an application using a DIF may be a higher layer DIF.

3. The Separation of Mechanism and Policy in Protocols

Protocols (and systems generally) are comprised of sets of mechanisms, such as acknowledgement or error-correction. Over the last 30 years, no new mechanisms have appeared. There are approximately 15 mechanisms, from which all networking protocols are constructed using appropriate subsets of these mechanisms. We define policy as the variant aspect of a mechanism. For example, acknowledgement is a mechanism, when to acknowledge is policy. Data corruption detection is a mechanism, what error code is used is policy. As one might expect, policies generally come in pairs, one for the sending side of the mechanism and one for the receiving side. For some mechanisms, there may be a policy to initialize the mechanism. Greater complexity is rare.

We have found that separating mechanism and policy in protocols exposes patterns in the structure of protocols that increases the range of operation while reducing complexity, with major implications for implementation both in hardware and software.

Mechanisms fall into two categories: **tightly bound mechanisms** have header fields that must be associated with the Transfer or Data PDU, such as sequencing or CRCs; **loosely bound mechanisms** have header fields that may or may not be bound to the Transfer PDU, such as flow control or acknowledgement. Processing loosely bound mechanisms is more general-purpose in nature and more complex, while processing tightly bound mechanisms has much less range and is simpler. There is also only minor interaction between the two types. This means that all error and flow control protocols can be cleaved into data transfer and data transfer control, which leads to a much more efficient implementation and considerable asynchrony and pipelining.

The policies for a protocol operating close to the physical medium, e.g. a data link protocol, tend to be dominated by the characteristics of the medium, while those operating closer to the user applications tend to be dominated by the requirements of the applications. This explains why data link protocols have been successful while there has always been dissatisfaction with transport protocols. Data link protocols were serving one master, i.e. the physical medium, while transport protocols had many, i.e. the applications. Not separating mechanism and policy in transport protocol designs implicitly assumes that there is a single point in an eight dimensional space (the number of mechanisms in a typical protocol) that meets the requirements for all applications. Once stated this way, it is no wonder the result has been unsatisfactory.

The only other difference between protocols is their syntax, and this is not an essential difference. The procedures of the protocol vary little if sequence numbers are 8, 16, or 32 bits, or if addresses are 8, 16, or 32, etc. If an abstract syntax is used that compiles to a specific concrete syntax, it is possible to define a single protocol with multiple concrete syntaxes. High-speed protocols would have longer sequence numbers, protocols intended for small networks or lower layers would have shorter addresses, etc. This may affect the modulus of the arithmetic or the length of strings to match, but the procedures for the protocols remain the same.

We have determined that all conventional data transfer protocols can be reduced to this structure, with a small number of concrete syntaxes and mechanisms and a wider range of policies. Any of

the capabilities of existing protocols can be achieved by the appropriate choice of policy sets or by using these in combination in different layers. And the resulting implementation is simpler. This is an important source of the power of our architecture to simplify, scale, and provide new capabilities. Hence, there are only two protocols: an **Error and Flow Control**, which cleaves naturally into data transfer and data transfer control. These modify state internal to the protocol. And a **Common Distributed Application Protocol**, which perform the six basic operations and modify state external to the protocol. As strange as it may sound, “protocols” such as IP, CLNP, Ethernet MAC, etc. are merely a fragment of common header in the error and flow control protocol. In fact, treating it as a separate protocol would create inefficiencies in the implementation. In essence, 30 years ago we should not have split TCP horizontally to get IP, but vertically to split data transfer from control.

4. Naming and Addresses

The source of most problems in the Internet architecture is the lack of a complete naming and addressing architecture. It has been well known that at least 3 levels of names were necessary: location-independent application names, location-dependent node addresses, and route-dependent point of attachment addresses. For historical reasons, the Internet never got beyond the last of these. This is the equivalent of a computer system with physical memory addresses but not virtual memory or a file names, although the Internet has provided domain names as macros for the physical memory addresses. This has lead to untold problems and is currently at the root of a crisis in the Internet today. Without a full addressing architecture, it is impossible to support multihoming in a manner that scales and many things are either very hard or impossible. To see this, consider what it would be like to try to create re-locatable code in a system without virtual memory. That is today’s Internet. PNA has a complete naming architecture, so the solution to these problems is inherent in the structure and accrue at no cost.

Another fundamental shortcoming in conventional network addressing is making routing decisions given an address. Addresses are supposed to be location-dependent, but route-independent. They tell you "where" something is without specifying how to get there. In daily life it’s well understood how, if you know where you are, given a destination street address one can use addresses along the way to find a route (e.g., lettered and numbered streets in Washington D.C.), but how it could be done for networks has been a major unknown.

We have determined that an abstraction of the graph of the network is the missing element; that addresses need to be topologically dependent. (The mathematical definition of topology is used here, rather than the common usage in networking where “topology” is generally synonymous with the graph of a network. A topology is a mapping that maintains certain properties invariant, while a graph is the connectivity of nodes and arcs. A topology would represent common properties among many graphs, e.g. properties that remain when some arcs are missing.) In PNA, the address space has a topological structure: there is a topology between the address space and the elements of the layer. A topology between the address spaces of adjacent layers may also exist but is not required. The topology of the address space is chosen to be an abstraction of the expected graph of the network.

The use of topological addresses combined with recursive DIFs radically reduces the number of routes that must be stored. In fact, the number can be bounded. Thus, using topological addresses reduces the number of routes stored from hundreds of thousands to only those associated with the subnet in which the router resides (and whatever "shortcuts" might cross the topology). In a conventional national-scale backbone, we anticipate this number will not exceed a few hundred. PNA thus not only solves this, the most severe problem in routers today, but also enables much faster router table convergence.

We have constructed one such topological mapping based on the common hierarchical structure found in many networks. Other topologies are possible. This approach to addressing is leveraged by there being a distinct address space for every layer. In this architecture, all addresses are in effect private addresses. One of the more unexpected implications of this architecture has been that a global address space is not required. Even more unexpected, while a global application name space is very useful, it is not a requirement of the architecture.

5. General Features of this Architecture

A Generalization and Completion of Conventional Architecture It is a long story and not immediately germane here but early networking used operating systems as its guide. The development of the ideas and the shift to a new model of communication was in its infancy; there was still much to learn, when several events arrested that development and Moore's Law removed the need to finish the shift to the new paradigm. PNA has in a sense picked up where we left off and found that it lead to an immense simplification and complexity collapse. On the surface, PNA resembles traditional models, but the differences are subtle and multiply as one explores its properties. If the Internet were an operating system it would have more in common with DOS than UNIX – a rudimentary collection of functions lacking the basic elements a complete architecture.

Simplification PNA dramatically simplifies networking. Many capabilities such as multi-homing and mobility fall out as degenerate cases of the general architecture. There is no longer a different set of modules and protocols for each capability in each layer. Instead, the elements of the DIF combine to accomplish functions that have previously required a multitude of individual specifications. The implications of this are significant: rather than hundreds of handcrafted protocols each with their own quirks, there is a consistent repeating structure of three protocols and a common header and roughly a half dozen modules. In this approach there is thus far less to go wrong.

A single replicable layer of a half dozen or so modules can be much more effectively engineered to be free of bugs and security holes than the literally hundreds of protocols in conventional Internetworking. Furthermore, the inherently constrained nature of policies makes it possible to circumscribe their side effects and ensure their proper behavior. It is possible to define properties of policies that can be proved or tested that ensure proper behavior.

Flows Each layer manages a range of bandwidth. Flows within a layer handle a specific combination of ranges of QoS. At lower layers, flows between intermediate points aggregate higher layer flows of complementary QoS. This will lead to more easily managed flows and fewer flows in the backbone of the network. The mechanisms in any layer depends on whether they are

required to maintain the QoS expected by the layer above. In some cases, it may only provide flow control, in others it may be completely absent. The basic structure is repeated with connection-like flows across some subnets (at least with flow-control), and connectionless within subnets.

Applications are able to operate on whatever layer has sufficient scope to reach all their correspondents. For example, corporate applications may operate on top of a layer whose scope is limited to their corporate network, while the corporation's website may operate on a public global layer. Corporate applications would thus be invisible to the outside world. This has the effect of Network Address Translation (NAT) without explicitly requiring NATs. That is, the meaning of NAT is transformed; since every layer has its own address space, NATs are an integral part of the architecture.² The public Internet is merely another organizational network that one may choose to join or not. Provider's networks have their own address spaces on which these organizational networks (layers) float. Messaging, peer-to-peer, mail relaying, and transaction processing are essentially instances of a DIF with different policies and concrete syntaxes. Proxies and caching are part of normal layer operation, i.e. relaying and routing, with appropriate policies and parameters.

6. Operational Advantages of the PNA Model

Greater Robustness and More Effective Response to Change. PNA supports a dramatic improvement in operational robustness. Response to change is far faster thanks to load balancing, quicker convergence due to much smaller routing table sizes, more responsive flow management, and, on the manpower side, simpler and more effective operational management. PNA provides all of the flexibility and survivability of connectionless networking while supporting all the service capabilities of connection-oriented networking. Data flows under hostile environments are far more reliable due to highly tunable, situation-specific policies.

Full Support for Multiple Levels of Qualities of Service. In today's network, a single layer attempts to manage flows that range over six orders of magnitude of bandwidth and growing - a resource management nightmare. QoS solutions are tied to specific applications, making adding a new class of QoS difficult and cumbersome, and requiring major changes in most implementations. PNA provides multiple levels of qualities of service independent of applications. We take a more analytical approach that makes it simpler to accommodate new classes of QoS. Since many QoS parameters can be expressed as properties of the traffic pattern, the QoS of the flow can be derived from the flow. Our approach, along with the recursive structure of our architecture, provides a solution that leverages the burstiness of traffic rather than covering it up. Layers aggregate flows into higher bandwidth flows. Each layer manages flows in a given bandwidth range, multiplexing them onto higher bandwidth flows in lower layers. Thus, the number of flows to be managed at a given layer can be bounded or even held constant. Resource allocation becomes more efficient and scalable. Providers can finally do something about traffic engineering besides "order more bandwidth," while attaining greater efficiency. This approach greatly simplifies management and engineering a network. We also provide the means for service providers to collaborate on providing QoS without divulging sensitive information about their networks, removing a major barrier to interoperation. There is a range of long-sought services that conventionally require

² NATs don't break anything in this model, because it has a complete addressing architecture. NATs only break broken architectures.

significant overbooking of capacity, or separate networks. These include voice and video, including multicasting of both, and teleconferencing. PNA makes these capabilities available without special provisioning.

Improved Security. The recognition that networking is a Distributed IPC-Facility dictates where authentication, access control, integrity and confidentiality are positioned. This yields a well-defined bounded implementation structure that is inherently securable and presumes little trust of either upper or lower layers.³ The DIF forms a securable container.⁴ Security is done once for a DIF, which can be done with greater assurance as opposed to the current Internet where essentially every protocol does its own security. A typical implementation might have to implement security mechanisms 5 times or more. An application only has access to the destination application name and a local port-id, i.e. handle. There are no well-known ports or addresses that can be guessed or fabricated. In addition, the recursive structure isolates provider infrastructure so that it is immune to attack by hosts. Furthermore, since infrastructure on a given subnet, including security infrastructure such as key management, cannot be addressed from another subnet, it cannot be attacked by any device outside the subnet. The existence of a public global DIF, or e-mall, in our architecture functions to isolate hosts from all private infrastructure, so that they cannot mount attacks. Another implication of the structure that arises from Watson's results and from separating mechanism and policy is that confidentiality and integrity can be provided without an IPSec-like security connection.

Security is an integral part of layer operation, and is absorbed into the data transfer protocols. Security management is an embedded application and is part of layer management. Recursion provides considerable flexibility in the scope and form security can take. For example, VPNs are a degenerate case of layer operation and are far simpler to configure than current techniques. VPNs, being yet another layer, have all the facilities and capabilities available to any layer, including security and routing. Using authentication on address assignment prevents spoofing addresses. Intermediate flow management makes it simple to track attackers and narrowly squelch attacks without affecting other traffic. The very small module count of the DIF makes it possible to create high assurance implementations. Assurance may be improved further by hybrid hardware/software implementations.

Improved Modes of Operation. Conventionally, the performance of mobility, multihoming, and multicasting has been unsatisfactory, even while requiring large numbers of complex protocols or mechanisms.

Multihoming has never been adequately addressed. Routing protocols are unable to determine when two interfaces are connected to the same host. If there are two connections to a given host and one of them fails, the routing algorithms cannot know to move traffic to the other connection. While the Internet has developed a number of workarounds, they are all expensive and error-prone, and it has recently been recognized that the current approach (provider-independent addresses) does not scale to work in current or future routers, an on-going crisis. While major patches are

³ An IPC-Facility must assume only that a lower layer will attempt to deliver PDUs to something. There are no assumptions about to whom or about reliability.

⁴ The degree of security is determined by the strength of the security policies used. Hence it is possible to configure a DIF with null policies or very strong policies.

underway, these are known to be flawed as well. In PNA, multihoming is a part of the architecture – it is completely supported by the architectural structures and topological addresses; no special protocols nor mechanisms are required and no special burden is placed on the routers.

Mobility is also not a special case – it is completely supported by the existing structures of the architecture and no special routing schemes are required, no “home-router,” and no “care-of” addresses. Mobility is equivalent to dynamic multi-homing and reduces to updating changes in the addresses of protocol state machines to reflect their position as they move with respect to the topology of the layer/subnet. There are no mobility-related scaling problems as with conventional, centralized solutions. The amount of traffic generated by mobile hosts is reduced and round trip time is reduced. Seamlessly migrating executing applications turns out to be easily supported by employing multicast naming along with the inherent addressing structure.

Distinct **Multicast** protocols are not required. Deploying multicast is fast and simple and only involves border routers. Multicasting is integrated seamlessly with unicast routing. Each border router merely calculates where it is in the distribution tree of the group and acts accordingly. This results in far lower operational costs and facilitates multiplexing of multicast groups. In PNA, multicasting is subsumed into the routing applications. As peculiar as it might sound, on the one hand unicast routing is a degenerate case of multicast routing, multicast routing becomes unicast in any one subnet.

Wireless Communication in PNA requires no special treatment - it is simply another medium. Currently wireless is an area of rapid technology development. This is driving the proliferation of a large number of new data link protocols. Using our architecture, these protocols can be created in weeks to months, not years, by simply using the appropriate policies. Furthermore, the new technology will have the full range of capability of the DIF, including security, routing, etc. from the get-go. Thus, we improve the speed of deployment and reduce costs by several orders of magnitude. Furthermore, normally adding a new data link protocol normally requires integration with the rest of the protocol stack - another expenditure of resources and time. Since the DIF always presents the same interface to the layer above, integration and deployment time are essentially nil. In combination with software radio technology, deploying new technologies would be a completely soft exercise that will reduce deployment time and make equipment far more flexible.

Scalability One of the most severe problems facing the Internet today is scalability. When the current protocols were designed in the early 1970's, no one planned for networks of the bandwidth, processing speeds, or numbers of users we have today. PNA scales with no upper bound over any range of users, resources, bandwidth, or distance. Any limitations are in the physics, not in the architecture. This scaling is enabled by the recursion of the layer. Requirements for router computation and storage capacities are structurally reduced. Router table size is orders of magnitude smaller, and bounded. Router capacity is dramatically increased by our ability to aggregate flows and cap the number of flows per layer. We change the way that routers are organized because we do the same things as before but in a different way with a different effect. For example: it is no longer necessary to store 200,000 routes or the projected millions; all of the calculation on full sets of routes are dramatically quicker, thus freeing up storage and computation resources to either do other things, or to apply more intelligence to the things being done.

Furthermore, the scaling is flexible. There is no requirement that a layer have no more than 100 or 10,000, or any other number of nodes, as long as it meets its operating requirements. This is facilitated not only by the recursion but by the use of topological addresses as well. Routes only need to be calculated to the nearest border router, not all the way to the destination. Also since multipath routing is inherently supported, the number of routes to be calculated is reduced. Equipment can be sized to fit economic price points and scale in those units. In addition, the policies of each layer can be more focused on a specific range of network operations and hence more effective in achieving its goals. Because these layers are private to the provider, there is greater flexibility to engineer the network. One is not constrained by the static structures today that require everyone to do the same thing. We expect that this will lead to a re-organization of routers and switches. We can also expect some layers to be quite large as equipment continues to exploit Moore's Law; however, this does not affect the architecture's ability to scale.

Improved Management. The repeating structure of the architecture yields much simpler management. With today's handcrafted protocols, there is no orthogonality to management; each protocol has its own peculiarities and interacts with other modules in unpredictable and unexpected ways. Earlier we noted that the current Internet approach of "incremental change" or constant patching served to increase the "parts count" and complexity. This ever-increasing complexity is not so much a concern of classic efficiency of processing and memory, but of manageability. The recent SmartGrid effort has determined that over 100 RFCs must be implemented for a device implementing the basic Internet capability. Patches that do not fit cleanly in an overall structure have unforeseen side-effects and require special procedures for configuration and management. These limit the flexibility of networks, increase the likelihood of errors, increase manpower requirements and generally cost untold dollars. At some point, it becomes (a perhaps unspoken) limitation on uses of the network. We must move to solutions that have less complexity and avoid these problems. PNA produces a considerable step towards simplicity. In our approach, interactions are consistent and predictable. With repeatability, far more sophisticated management is possible and with far cheaper personnel - the expertise required to operate a network is greatly reduced. We anticipate as much as an order of magnitude reduction in management costs through automation and de-skilling. This also provides significant advantages in difficult (e.g., combat) operating environments.

Improved Engineering

- Faster Engineering Response to New Problems As new demands arise, systems can be quickly reconfigured to accommodate them simply by configuring new policies. Furthermore, research results will be much easier to evaluate and quickly incorporate into product. Today, new research takes the form of new protocols with new, unknown interactions and side effects; when new techniques are implemented from scratch it is difficult to determine how much of the change is due to the new technique and how much is "the other new stuff." In PNA, a new technique may consist of one or two new policies that can be configured and evaluated in isolation. Subsequent deployment is quicker and easier. We expect new businesses to emerge providing custom or specialized policy sets to customers and manufacturers, independent of the hardware manufacturer.

- Lower Engineering Costs Hundreds of engineers are required today for the development of a major new protocol, which normally takes four or more years, even without a silicon implementation. With PNA, two engineers can create a new DIF from concept to deployment in matter of months or even weeks.

- Improved Hardware Implementation Past attempts to put protocols into silicon have not been commercial successes, partly because there are so many protocols. Similarly, there are limitations to the effectiveness of conventional network processor products since there is only so much that can be done and still remain protocol-independent. In PNA, far more can be put into silicon without compromising capabilities. Our analysis indicates significant opportunities for asynchrony and pipelining in protocol processing. Also the reorganization represented by PNA eliminates the need for protocol “options.” Hence, there is no slow path. Everything is fast path. This will not only increase performance but could reduce hardware costs by an order of magnitude or more.

7. Adoption

The second question everyone asks is “What about Transition?” There is no transition, only adoption. We do not foresee a transition to PNA. Existing Internet Applications continue to use the Internet as it is. The Internet becomes a kind of public cyber-mall, which is close to what it is today. While business and new applications, at their own rate, deploy in the secure PNA environment. The legacy can be allowed to atrophy over time.

PNA can be used over IP, under IP, or along side IP. Deployment can begin with one pair of devices and proceed by adding one device at a time. There is a range of adoption strategies. The first and simplest deploys DIFs under IP, much as MPLS is deployed today. This allows the network to more effectively manage flows and provide better performance to users. Second, a common layer that emulates a Sockets API could be used to encapsulate existing applications to provide them with some of the benefits of layer operation without modifying legacy applications. This could take two forms: a pure Sockets API that maps to the DIF with policies and parameters that make it look like TCP, or a Sockets API with options that would allow it to accept directives and thus enable the DIF to provide an improved service and in turn enable the applications to make better use of the DIF. This would require only minor changes to the application. In either case, the number of legacy applications is sufficiently small that the layer could be aware of what the applications were and use policies appropriate to those applications. Third, new applications could use an API that directly accessed the layer's capabilities, to obtain all the benefits described here. Furthermore, a PNA structure could be wrapped around TCP/IP to yield some (but not all) of the benefits of PNA. This structure could exist in a subnet and interact transparently with traditional TCP/IP systems without them being any the wiser. And of course, a dual-stack approach could be continued used as long as desired to internetwork legacy applications over TCP and legacy applications over a DIF. This would be no more difficult than current dual stack/NAT transition plans. However, we do not foresee this as common, since it creates security problems.

Using a DIF to support applications has several benefits over current practice: a) a common application naming scheme, b) a consistent authentication regime, c) common security services, d) application layer routing and relaying to support messaging, proxies and caches, and e) one and two phase commit, etc. are all services of the layer. For example, an application developer of a new mail system would only need to create a few policies for routing with most of the effort concentrated on the problem of mail composition. Today, the lack of application layer addressing has limited conventional applications to the primitive client/server model. Using the DIF to support applications opens the door to true multi-party distributed applications, first envisaged in the early days of the ARPANet.

9. Status

These innovations are the result of 15 years of research. The specifications for the protocols are in place. A rudimentary prototype has been implemented and verified many properties of the architecture. Our next step is a full architecture implementation which we estimate will take roughly 12 months. Though there are many potential end uses for such an implementation. And many products that can incorporate it, the majority of the work only needs to be done once – another simplification wrought by the adoption of an inherently simply unifying architecture.

10. Conclusion

We have developed a breakthrough advance in Internet architecture, with all the characteristics required for quick adoption. Rooted in fundamental insights into the behavior of networks, PNA is an extension of existing technology and new insights that lead to a complexity collapse. It has a straightforward operational model that solves the major problems in Internet infrastructure. PNA enables critical, long-sought services, modes of use and operating characteristics, and improves security. It reduces time and cost of network engineering, development and operations. Finally, PNA has a straightforward adoption path that permits easy interoperation with conventional internets and maintenance of legacy equipment value. A world-class team is being assembled to lead the development this breakthrough architecture.