A Clean-Slate Design for the Next-Generation Secure Internet

Steven M. Bellovin, Columbia University
David D. Clark, MIT
Adrian Perrig, CMU
Dawn Song, CMU

1 Introduction

This is the report on a workshop held at CMU on July 12–14, 2005. The workshop is part of the planning process initiated by NSF to explore potential architectures for a next generation secure network designed to meet the needs of the 21st century. In considering future architectures, we ignore issues of backward compatibility with the current Internet but seek to benefit from the experience gained by analyzing both the strengths and weaknesses of the current design. Specifically, this workshop looks at the fundamental interplay between security and underlying network architecture and seeks to chart a preliminary course for future work in this crucial research area.

This workshop focused on initiating a productive dialog between experts from the network security and network architecture communities. The agenda was arranged to stimulate initial consideration of the security goals for a new Internet, the design space of possible solutions, how research in security and network architecture could be integrated so that security is included as a first-tier objective in future architectures, and to explore methods for identifying and considering the social consequences of these architecture and security design choices.

1.1 Why do we need a clean-slate design for the next-generation secure Internet?

Today's Internet is a clear success. However, some aspects of the current Internet fall short of both current expectations for a reliable communication infrastructure and future demands that we would like to be able to put on such a network. Perhaps the attributes most critically lacking are those relating to security, including highly resilient and dependable availability, and a trustworthy environment for people (and their computers) to communicate.

The NSF initiative is fundamentally based on the premise that to achieve a substantive improvement in what the Internet can offer its users, the research community must accept a challenge—not to make the Internet a little better through incremental changes—but instead, to envision an end goal – what communication infrastructure we want in 10 years – and take bold steps toward creating and evaluating potential designs. The process of incremental change, without an overall vision for where we are going, runs the risk of giving an appearance of progress without actually moving us any closer to our specific long-term objective of a

This work was supported by NSF Grant CNS-0540274, "Collaborative Research: Planning Grant: A Clean-Slate Design for the Next-Generation Secure Internet"

secure yet flexible network architecture. As Yogi Berra said, "You've got to be very careful if you don't know where you're going, because you might not get there."

Despite the Internet's critical importance, portions of the its architecture are surprisingly fragile and the entire network suffers under the weight of incessant attacks ranging from software exploits to denial-of-service. One of the main reasons for these security vulnerabilities is that the Internet architecture and its supporting protocols were primarily designed for a benign and trustworthy environment, with little or no consideration for security issues. This assumption is clearly no longer valid for today's Internet, which connects millions of people, computers, and corporations in a complex web that spans the entire globe. Given this level of vulnerability, industry and researchers alike have moved to counter these threats, yet the difficulty they have faced making significant gains in this area is a demonstration that the source of many of the Internet's most critical security problems are rooted in a now highly ossified and fragile network architecture. Seeing this fundamental problem, we seek to redesign the Internet architecture from the ground up with security and robustness as primary requirements.

Adding a security layer to the protocol stack may at first glance appear to be a viable approach to securing the Internet architecture. Unfortunately, securing a single layer is insufficient, as an attacker can exploit vulnerabilities in the layers that remain unsecured. To briefly illustrate this problem, consider the issue of BGP routing security. To secure BGP, researchers proposed securing the communication between two BGP speakers by having them share a secret key and adding a key-dependent MD5 checksum to each packet exchanged. Unfortunately, this does not secure the semantics of the exchanged information, and a malicious router with appropriate credentials can still send malicious routing information. Conversely, securing only the semantics of the routing information is also insufficient, since an attacker could now exploit vulnerabilities in TCP to tear down the connection between two BGP hosts (the TCP MD5 checksum option prevents this attack at the transport layer). Hence, we need to holistically consider security at *every* layer of the protocol stack.

While simply "bolting on" security to current protocols may also seem appealing, it too also has important but often subtle drawbacks. For example, a primary challenge in securing (or even altering) individual network protocols is the complex and often unanticipated interaction between multiple protocols. Often, changing one protocol will introduce incompatibilities with many others or limit some type of desirable functionality, resulting in a highly over-constrained design problem. This is one of the main reasons the Internet has reached its current ossified and highly fragile state.

Additionally, adding security to an individual protocol often results in increased protocol complexity and an increased number of assumptions required for the protocol to operate in a valid manner. Such additional complexity can further increase overall system vulnerability as by introducing new failure modes.

1.2 A Fresh Approach

In order to envision a next generation network that may be vastly different from today's Internet, we must avoid the trap of taking any portion of the current network architecture as a necessary requirement for a future design. As a result, this NSF program is focused on the question of how we would design an internetwork if we could design it "from scratch" today. In doing so we will set our sights on forward-looking requirements and mechanisms, taking what is valuable from past designs, proposing new approaches when needed, and fitting the resulting ideas into a fresh overall architecture. In this respect, the process of design has been called "clean slate", meaning that the research community is encouraged not to be constrained by features of the existing network. The challenge is not change simply for the sake of change, but to make an

informed speculation about future requirements, to reason carefully about architectural responses to these requirements, to stimulate creative research in innovative and original networking concepts to meet these requirements, and to produce a sound and defensible argument in support of the architecture(s) proposed.

How might the success of this project be measured? One answer might be that a new Internet architecture is developed and then deployed wholesale in place of the current Internet. There are some who believe that this ambitious goal is the only way for us to shift to a materially improved Internet. The current network, which was initially architected in the 1970's and has evolved incrementally since then, may now carry enough baggage from its three decades of evolution that it cannot continue to evolve to meet the requirements which we expect it will face in the next decade. Another reasonable and successful goal for this research would be that by setting a long term vision for where the Internet should go, this research will help shape and inspire a more creative evolution of the existing Internet toward that goal. If the research community can set a vision of what the Internet should be in ten years, and provide a viable path to get to that point, we will likewise consider this project a success.

There are a number of planning activities considering both the requirements for this new network as well as approaches to achieving these requirements. While this report focuses only on the space of security, this workshop group interpreted security in a broad sense, and as a result this report discusses a wide range of architectural issues.

2 Goals for a Next Generation Secure Internet

In this section, we describe at a high level several of the most important goals of the next generation secure Internet.

Availability: If a set of users or applications on systems desire to communicate and these systems are both well-behaved and allowed to communicate by the policies of the interconnecting networks, then they should be able to do so.

This is a deceptively simple goal. First, it begs a clarification of "well-behaved". Second, it begs a discussion of connectivity and the economics and payment for service, which was outside the scope of this workshop. More importantly, it implies an open-ended quest to identify and remedy any barrier to availability, ranging from transient routing instability to denial of service attacks. This ambitious interpretation of this goal is, in fact, the intended one. We believe that availability, and addressing all the issues that might impair it, should be the highest priority objective for a next generation network.

Network functionality to assist in end-host security: One of the most vexing issues today is the poor state of end-host security and its implications for the overall state of the network. While a network purist might say that the security of the end-host is not the responsibility of the network, if we pose "good security" as an overall goal for a next generation Internet, this promise must make sense to the lay audiences — the public, Congress, and so on. For us to claim good security and not acknowledge the problems faced by average people, such as zombies, phishing, spam, spyware, worms and viruses — all of which involve the end-nodes — would leave the claim of "good security" vacuous to all but a small group of researchers. As such, part of this project must be to take a considered and defensible position on the role of the network in supporting the end-host security, and to propose a consistent division of responsibility between the network and the end-host system in achieving good security. Our eventual architecture must take into account the

expected progress in the state of end-node security, as well as realistic limitations on our ability to modify or replace the base of operational end-host systems. Despite this, we must remember to take an approach to building a secure network architecture that will make sense in the larger context of the user's experience.

Flexibility and Extensibility: The next generation secure Internet architecture should be flexible and extensible and capable of supporting innovative applications. The current Internet architecture has proven rigid and difficult to modify, leaving the network less able to cope with unanticipated and large-scale changes in network behavior. We hope a next generation secure Internet will learn from past lessons and be be designed with adaptibility specifically in mind, leaving it better prepared to suport new applications. Moreover, security mechanisms should be robust to normal extensions of the technology by users and operators, e.g., by tunneling/encapsulation inside the network.

3 Workshop Structure

Approximately 40 people, from government, industry, and academe, were invited to the workshop; a list is given in Appendix A. We looked for a wide range of expertise. Obviously, we wanted networking specialists, but networking is a broad field. We also invited a wide variety of of security specialists.

Less obviously, we invited people with expertise in law, public policy, and sociology. A new network design *will* have social implications; it is better to make such decisions explicitly, with full knowledge of the consequences. To give just one example, increasing traceability of network activity might have a chilling effect on legitimate anonymity and privacy, and hence on free speech. Where should the boundary be drawn?

All participants were requested to submit white papers; those may be found at `http://www.cs.columbia.edu/~smb/ngsi-whitepapers`. A list of the white papers is in Appendix B; the workshop agenda is shown in Appendix C

There was a keynote address, by Steven Kent; there was were also presentations of several of the white papers. Following that, tehre were a series of plenary sessions followed by breakout groups to discuss naming and addressing, host versus network responsibilities, and network management; these groups reported back in a plenary session. The workshop concluded with discussion of infrastructure needs and recommendations to NSF and the research community.

Although cryptography was not an explicit agenda item, various cryptographic issues were mentioned during a number of the sessions. The authors' understanding of these concerns are given in Appendix D.

4 The Sessions

4.1 Workshop Outline

The workshop started with a presentation by Darleen Fisher, from NSF, who made the point that security is anything but a new problem, and that studies (CSTB reports, for example) have been calling for a fresh assault on the problem for over 15 years. She noted that there has been (to some extent) a evolution in thinking, from security as an absolute all-or-nothing objective to an approach based on acceptable insecurity and security as risk management.

David Clark talked about new architecture and clean-slate design, and offered a challenge to the security community to be a partner at the table during the design process, as opposed to coming in after the fact for

a security review and telling the architects what they did wrong. He asserted that a future Internet design project might focus much more on security and manageability as fundamentals, rather than just data delivery.

Adrian Perrig provided a framework of requirements and approaches (Appendix E). He challenged the workshop with a difficult formulation of the problem, in which any element (of any sort) can potentially be compromised, and where compromised nodes can collude. He offered a design space of approaches to security: prevention, detection and recovery, resilience and deterrence, which triggered considerable discussion. During the course of the workshop, there were doubts expressed about all of these. In the mind of at least one of the attendees, each of the following is true: Prevention is seen as brittle and inflexible, detection and recovery (e.g. intrusion detection) is ineffective, resilience is too hard to be practical, and deterrence is not effective. So we lack agreement even as to what fundamental approaches we should take, and what emphasis we should assign to each of them. (In fact, there seem to be good examples of each approach, but these are isolated examples, and not part of any architectural framework.) It was also asserted that in the real world, resilience is the only thing that actually works.

Steve Kent gave a keynote talk on security topics for a future Internet. He looked at the history of security in the present Internet. He noted that we were concerned from the beginning about malign end-nodes (as well as multi-level secure end-nodes), but that there was little attention to denial of service, non-repudiation, and secure communication between end-nodes and network infrastructure. He offered a list of what we have now figured out:

- We cannot depend on the discipline or sophistication of users.
- We cannot depend on correct user configuration of controls such as ACLs and firewalls.
- We cannot depend on security models based on managed trust assertions.
- We cannot depend on intrusion detection.
- We cannot depend on application designers paying attention to security.
- We cannot depend on ISPs to perform ingress filtering or other security checks.
- We cannot depend on legal deterrence.

After this rather pessimistic assessment, he offered some design lessons or approaches:

- We must clearly articulate assumptions about context and security goals of mechanisms.
- We must not compromise assurance for increased functionality.
- Protocol design is very, very hard.
- Crypto is rarely the weak link.
- Technologies for security will be used in ways they were not intended.

To evaluate a new proposal:

- Think about the infrastructure needed.
- Consider the management requirements.

- Beware of further DoS threats.
- Pay attention to alternatives for design of identifiers (see below).
- Don't give attackers a means to influence the behavior of the target.
- Evaluate a device in relation to the perceived threat.

After this talk, there were several short talks on architecture, and then a series of topical discussion. The following summary combines the content of these talks with the various discussions.

4.2 Naming and Addressing

There was a great deal of confusion about this topic, because the discussion started with mechanism (names, etc), and not with requirements. The conversation floundered because different speakers were exploring different requirement spaces. Listing requirements would be an excellent exercise, but really hard. However, it would more directly bring out the issues.

To some, naming was about search, and this triggered a discussion about top-down assigned meaningful names vs. a world of search (Google-like) to find things. These two world-views have very different security implications.

To others, naming is about intrinsic identity, and this leads to questions about the need to reveal names, anonymity, and so on. In contrast to this, others felt that names play all sorts of roles from long-term continuity (e.g. archived information), short-term continuity (maintaining and validating sessions among mobile machines), names for aggregates of entities and so on. It was clear that to some people, things only have one real name, which is associated with deep existential identity. Others saw names as reflections of different aspects of identity, and assume that the same entity can have many names, including names that are built up out of attributes. There was a lot of uncertainty as to whether one can talk in general about any security properties of a name. It was noted that programming languages have created very precise frameworks for naming, with rigorous semantics. It was asked whether this had anything to do with the way we use names in a system such as the Internet. In particular, compilers tell us when we make a "naming error". The Internet may not. Is there any analogy to the role of the compiler here?

One use of names is to solve the "eq" problem: is this object that I found via one means the same as the one I found by another means, or is this item today the same as that item yesterday? This use raises very different issues than names as a means of traceability, which implies some link back to some accountable entity associated with the name. The goal of traceability triggers interest in mechanisms that give you "conditional privacy", which can be revoked if you "do bad things". (Some digital cash has this design.) This begs a set of larger social questions about who controls the decision to revoke the privacy, of course. This comment reflects a larger theme of the workshop: security, as we need to address it in a future Internet, is as much a social as a technical design space.

A fundamental aspect of naming and identity is how entities come to know each other—the problem of introduction. There was considerable debate about the utility of different approaches, described as "top-down" and "bottom-up". This discussion reflected both technical issues and social concerns. When users get to know each other, they often don't make use of a third party to introduce themselves. They get to know each other by a process of exchange of identifiers, which are then used over time to allow a continuity of the interaction. This was called the bottom-up approach. In contrast, the group used the example of how a user gets an association with a bank. In this case, there were those who thought that this association is so

important (and getting it right is so important) that it would be necessary to have some trusted authority—some third party that would introduce the user to the bank and assure the identity of each to the other. The user might need to be linked to some well-known identity such as a SSN or passport. On the other hand, a user might get some identifier or credential for a bank by physically going to a branch of the bank and downloading some information onto a device, and this pattern feels more bottom-up. It was asserted that credit cards are a great example of the third-party top-down pattern, where the authority that issues the identities also manages the trust and risk in the system. This raised the speculation that the "top down" and "bottom-up" imagery was actually about the direction in which trust flowed. There was no clear sense that the group agreed on what patterns would make the most sense in different circumstances, and would be most usable and robust.

So the discussion of naming and identity was also about trust and where it comes from. If there is only one source of trust, then identity schemes have to be top-down, from that source, but in real life there are many sources of trust, and many ways for parties to get to know each other. Governments will play a role here, as well as third-parties such as credit card companies, certificate authorities, and so on. On the one hand, it seems awkward if every trust model generates a new name space, but it is also confusing if different names cannot be uniformly trusted in different circumstances, especially if this variation is not obviously visible.

One architecture question concerns what should be visible "in the network". Many functions for names require that they have meaning only at the end points. But especially as we separate the concept of name and location (to facilitate mobility) the question arises as to what sort of identification, if any, should be visible in the packet, so that elements in the network can observe and use it. One view is that there is no single answer to this question—that different situations in different parts of the network will require different degrees of visible identity in the network, so that the packet format might include an identity field, but not a fixed specification of what that field means. Different sorts of identifiers might be used in different circumstances.

The important roles for identity "in the network" seem to include accounting, access control and traceability. The implications of accounting are not at all clear. Accounting might seem to imply robust (fraud-proof) identifiers in packets, but may actually imply more "setup" protocols and state in the network, rather than more information in each packet. And traceability may be accomplished to some extent based on location information and identity information provided by the end nodes. There was concern about building tracing tools that are "too good", because of fear of abuse. Tracing, especially when it does not involve cooperation and consent of an end-node, should perhaps require a high work factor.

Another requirement in this space is service location and binding. If services are bound to fixed (or long-lasting) addresses of physical servers, then the address can be a proxy for the service. But in many cases, application designers want a more abstract way of naming a service, rather than associating a service with a single server. Today we can do this at several levels. We can use anycast to name a service, in which case the client thinks the service is "named" by its address but there can still be multiple copies. Or we can use an extension to the DNS, so the service is "named" by its domain name and the DNS maps this name to one or another instance of the service. These alternative (and others that might be invented as part of a new architecture), have both functional and security implications that have not been well mapped.

A specific issue in this space is the relationship between private address spaces and NAT, which some see as a security enhancement, and the desire to make servers with private addresses globally available. There are many examples today of applications that struggle to deal with NAT, including peer to peer schemes and SIP. A new architecture will probably include a new mechanism for rendezvous and service connection, and

this will have to be designed taking security considerations and private address spaces into account.

At a higher level, there was a brief discussion of the structure of URLs, which contain both a location and a name. This can be seen as both a good thing or a bad thing, but there has been little overall analysis of the security implications of different alternatives.

The discussion of naming brought up classic CS considerations such as early vs. late binding (DNS lookup vs. routing, for example) and the lack of a rigorous security framework to compare different designs.

For any name space, the security analysis should include a review of what infrastructure is required to validate the names, and how the security of this infrastructure matches the assumed threat model.

4.3 Routing

One important mechanism discussion concerned source routing. Source routing shifts the whole landscape in unfamiliar ways. It is seen as a tool to give control to the user (e.g. to keep traffic from certain unwelcome parts of the net or employ user choice as an economic incentive), but it still depends to some extent on the user getting valid information from the net. To the extent that this information can be falsified, this creates new and unfamiliar vulnerabilities. However, user control is still real—the user knows what he is trying to accomplish, and can see (to some extent) whether he is succeeding. (He can see if his packets are not getting through. He cannot see if they are going along the path he expected, unless we add new mechanism. He cannot see if they are going places he did not want them to go.) So if a path is failing, the user can detect this and try another, even if the user does not know what is wrong with the path. There is a range of design options from total knowledge of route alternatives to the limited idea of saying "try some other path." Some folks asserted that the end-user would have to know which nodes in the net were under attack; others thought that the user could just hunt for working paths, which is an example of resilience—a less efficient but still functional mode when the system is under attack.

We don't have a comparative model of the amount and distribution of information in traditional vs. source routing—source routing requires a (perhaps partial) connectivity graph, which is more than routing computes or reveals today. On the other hand, if connectivity information is gathered and validated in a distributed (and potentially redundant) way, there may be less opportunity for a corrupted node to have a widespread effect. And there is no framework for the source to determine what parts of the network are more or less trustworthy. With BGP routing, ISPs that don't trust each other will not tend to use each other. (The current bilateral agreements are a weak proxy for bilateral trust assertions.) But this information is missing (unless we add it back in) for source-selected routes. So there is a different trust framework, one that we have not explored.

We identified a number of security-related objectives for source routing: avoidance of degraded performance, avoiding content filtering, avoid regions or countries that violate some end-node policy, avoid paths that drop packets, and avoid impersonation of other end-points.

This conversation was closely related to one on tunneling and encapsulation. Again, the discussion started with the mechanism, and struggled backwards toward requirements, which (not surprisingly) looked a lot like source routing. Encapsulation can hide more information (the ultimate source and/or destination), so it can meet other requirements such as forms of anonymity and increased protection from traffic analysis. It was noted that tunnels can be used by multiple parties—the user can make a tunnel back to a home network (a VPN), an ISP can make a tunnel (a level 2 network), or a third party (some sort of overlay network, perhaps) can make a tunnel. The objectives and the balance of power is very different in the three cases, but the mechanism may be the same. Depending on which actor deploys the tunnel, different parties

will benefit. This raises the question of tussle, and what constitutes a level playing field (or a properly tilted one). It also raises questions of what happens if you shift control (power) to people who do not have the ability to use this control. Perhaps the answer is that the user will delegate this control, and will exercise power indirectly—by picking the agent that makes the detailed decisions.

The discussion of tunnels led to a discussion of overlays. If a new architecture is designed to support overlays, then what is the responsibility (with respect to security) of the overlay and the "underlay". It was noted that overlays have been proposed to control DoS attacks, but DoS attacks directed at the network itself (link capacity) have to be solved at the underlay. It seems messy that DoS has to be managed at multiple levels, but perhaps DoS is a phenomenon that exists (and must be managed) at all layers. This is a good question for thought.

There may be very different architectures for overlays themselves, depending on who deploys them. One option is that the overlay is a single, global system run by one entity, so that trust is not an issue among the parts of the overlay. Another option is that the overlay itself is built up out of parts managed by different entities, so that issues of trust, regional boundaries, and so on will be issues for overlays. Will overlay designers end up inventing an equivalent for BGP?

4.4 Network Management

Before the group could discuss the interaction of management and security, it first considered the state of management itself. An initial presentation by Hui Zhang framed the issues. In the original Internet design, there was little attention paid to network management. In addition, the fundamental design principles have been violated. The Internet designers initially assumed that the IP architecture was stateless, but there is state and logic everywhere. There is no consistent approach to managing or maintaining this information, no overall security architecture for this information, and no uniform discipline for dealing with inconsistency in this state. We need an organized set of methods to deal with this state, and a view of the security implications of different methods.

Conceptually, if one considers network control as defining the acceptable behaviors of the underlying network, then network control is akin to "programming". For example, one can argue that activities such as configuring firewalls or setting up weights of BGP graphs are programming exercises. The difficulty in programming the network control plane is that what is being programmed (the Internet) is amorphous due its scale as well as to the emergent behaviors that come about due to its open nature. Today, and to a large extent, network control suffers from the same lack of organizing principles as did programming of stand-alone computers some thirty years ago. Primeval programming languages were expressive but unwieldy; software engineering technology improved not only through better understanding of useful abstractions, but also by automating the process of verification of safety properties both at compile time (e.g., type checking) and run times (e.g., memory bound checks). What programming languages have done to software engineering is to force programmers to adopt a disciplined approach to programming that reduces the possibility of "bugs" and by making it possible to mechanically check (whether at compile time or run-time) for unacceptable specifications/behaviors. In many ways, this was done at the expense of reducing the expressive power given to programmers. High-level programming languages do not afford to programmers the same expressive power that assembly language does (i.e., there are programs that one can write in assembly language that one cannot write in Java). High-level abstractions that restrict expressiveness are not unique to programming languages, they are certainly the norm in Operating Systems, whereby programmers are allowed to interact with (say) system code and resources in prescribed (less expressive) ways. This loss of expressive power is

precisely what has enabled us to deal with issues of scale of software artifacts and systems. Yet, the same has not yet materialized for network management and control. Along these lines, the same kinds of benefits in dealing with issues of scale and complexity could find their way into network management and control if we adopt a more "disciplined" approach – an approach that confines the ability of network managers to "program" the network. Of course finding the "right" balance of expressive power and the resulting safety/security guarantees, and mapping such expressive powers to trust levels is the challenge. What we need is a "disciplined" approach for trading off expressive power for security/safety.

The group tried to define the difference between management and control, where an example of control would be dynamic routing. One hypothesis is that control is automatic, and management involves humans, but today there is a goal to automate more and more of what we call management, and as well to automate more of diagnosis, which is somewhere between control and management. Another hypothesis is that the distinction is one of time-scale: control operates in a time scale of seconds and fractions of a second, while management acts in minutes and hours. Another distinction is that control seems to involve distributed algorithms that run on the elements that make up the data plane, while management seems to be more centralized. One current research direction is to move the computation of routing tables out of the routers and into a more centralized element. This is described as moving the computation from the control plane to the management plane.

Another viewpoint is that management is about dealing with rare events, and control is about dealing with ongoing "normal" events. Rare events may not merit the effort of automated diagnosis and correction, and may be intrinsically complex and difficult to sort out.

One comment was that since the control functions may not have been designed to be managed, it may be necessary to "reverse engineer" or "tweak" the control plane to allow management to occur. There is a basic research question in how to design control algorithms that themselves can be controlled or managed. What are the "control knobs" that should stick out of a control plane? This problem was also framed as one of visibility-control algorithms, exactly because they operate automatically, may not be designed to reveal what is happening, with the result that operators cannot see or understand the workings of their network. So there is a need on the one hand to give more visibility into the network, and at the same time to make sense of all the data that various sensors are gathering. Monitoring tools have to infer data they should just be able to measure, while we drown in perhaps irrelevant data. There does not seem to be any framework or model to help shape this area.

It was noted that network operation and management is a major source of operating costs, and as well a source of failures and disruptions due to operator error. So we can expect major changes in this area, and we can expect this to be a major focus of a future architecture design.

Cross layer management was brought up as a significant problem. Today, operators use very complex layer 2 mechanisms (e.g. MPLS), but there is no visibility from the layer 3 (IP) tools into layer 2. Different layers often have their own separate control mechanisms (and may attempt independently to correct the same fault), and they may have separate management systems with different operators. On the other hand, it was noted that MPLS, because it is "less dynamic" with respect to control (e.g. configuration is based on off-line traffic engineering computations), that it may be less a source of security problems and instability.

So an architecture for management (and control) will deal with issues of control and visibility, with decentralization vs. centralization, with intra and inter-regional flow of data and control, with abstraction, and with the issue of which parties have control and how different parties interact. Future management will do a better job of allowing managers to ask "what if" questions about proposed changes and alternative

configurations. The architecture will deal with extensibility.

The discussion of management and security was again framed by Hui Zhang. Management is seen as the source of many security problems. Management is ad hoc, which always creates problems. Management depends on state that is spread over many entities, and we don't have good models to reason about the security of the resulting system. Management systems may contain their own loop-holes, which attackers can exploit.

Using the taxonomy of security approaches provided by Perrig, we can see that management has a role in all of them. Prevention implies the need to configure security protocols, access control lists and policy enforcement elements. Detection and recovery involve continued data gathering and analysis. Resilience includes automated control algorithms such as routing. Deterrence involves auditing and forensic analysis.

Since failures and misconfigurations are seen as a major challenge for availability, there was discussion about stability and how the network (and its control and management) recover. There was some speculation that the Internet now has enough unrecognized interlocking dependencies that it might not recover properly from a massive failure (e.g. a global power failure). More generally, there was a sense that network protocols and algorithms should have the general property that they have a homing sequence that takes them to some stable state from any starting condition. This might be a primitive state that just allows the full control and management mechanisms to come into play, and this concept (that the network might have more than one operating "level") might be part of a future architecture. This idea, in turn, raised problems of how a highly distributed system can ever be in one or another consistent state or level.

There is some relationship between management and the earlier discussion of identity. It was proposed that all elements in the system should have some form of identity. Elements that today are "lower layer", such as switches or interface cards, should become visible to some extent and have an address or identity. Management and security may challenge us to remove the rigid boundaries that today separate layer 2 and layer 3 to the extent that they live in different name and address spaces.

4.5 End-node Security

The group recognized that it will be necessary to deal with end-node security issues if we can make the claim that "the Internet" is secure. This reality begs the architectural question of what the role of the different Internet elements should be in building an overall consistent approach to security.

Today, the firewall is the most obvious element "in the network" that tries to protect the end-node, so there was considerable discussion of firewalls. There were mixed views on the overall utility of firewalls in today's world. On the one hand, it may be no harder to configure an end-node than to configure a firewall. Firewalls provide only imperfect protection, and do not deal with issues such as insider attacks. On the other hand, firewalls can prevent a range of simple attacks, and thus raise the bar for attacks from outside the perimeter. This seems like a useful contribution. As a specific example, a firewall "in the network" may be able to filter out some malicious traffic that would otherwise clog the access circuit to the host. Physical placement of an element such as a firewall relates to what sorts of protection it can offer.

Since this workshop is about new architecture, the discussion about firewalls should be turned into a discussion about design: if there are going to be elements in the network that perform security related functions, what should those functions be and how can the architecture improve the robustness with which these functions are performed?

It was also noted that there are a wide range of devices that might be called firewalls, which differ with respect to how (and by whom) they are managed and controlled. At one extreme might be a device

under the control of an end-node, fully trusted by the end-node, which has access to private information about identities and trust relationships of that end node. At the other extreme might be a box located in the network that tries to detect and block unsuitable activities without any ability to interact with the end-nodes to determine what they are trying to accomplish. The difference in the locus of control is as fundamental as any functional difference.

From a social perspective, this raises questions of the user vs. the control. Should a user be able to override the protections in a firewall if the user wants to do something that is potentially dangerous? Who has the final say over what applications the user can run, and what policies should be installed in the firewall? This discussion, again, is about power and control in the larger social context.

This point relates to the increasing use of encryption in the network. If end-to-end encryption becomes the norm, observation and filtering in the network will become less and less effective and relevant. But outsourcing of certain checks to trusted nodes that perhaps share encryption keys may become more useful. This again illustrates that the issue of trust and control may be the most important factors in defining a future firewall.

The issue of control also arises in the concern about how to deal with hosts that have become infested with malware. If the end-node controls the firewall, once the end-node is infested the malware can reprogram the firewall, and this become easier if we architect the protocols by which this control is implemented. This reality raises the question of whether there needs to be some trusted path between the firewall and some trustworthy control point, whether this is a human or a third-party control element. But there is a tension between the idea (which was generally accepted) that the host should have control over what traffic it gets, and the idea that the host needs to maintain some protection for itself in the case that it becomes infested with malware.

Firewalls can only be a part of an overall security solution. There was significant agreement that "defense in depth" is a necessary approach for a future architecture. The goal of an architecture must be to guide users and operators in the process of combining firewalls and other security elements into an overall solution.

Firewalls in the network have the effect of blocking certain traffic, which raises issues of visibility (as well as control) with respect to what the firewall does. If a firewall blocks traffic silently, this triggers a failure that may have to be debugged by network management systems or network managers. Some firewalls may insist on the silent failure—they may not even want to admit that they are present in the system, or what their policies are. But this raises problems that to the user look like network availability problems. One workshop participant asserted that end-nodes should be able to tell what sort of elements are in their communication path, and to detect if they are being "protected" by some sort of filter. So there needs to be a thoughtful balance of these issues, as part of a new architecture.

There was also a discussion of virtual networks as a security tool. Virtual networks provide a very rigid isolation of communities of users. They define the points (if any) at which traffic can enter the virtual net, which may make it easier to deploy firewalls and "border guards" at the right place. However, it was noted that a compromised (or malicious) node can inject unwelcome traffic into a virtual net, so the protection of a virtual network is not absolute, and perhaps no greater than a single network with firewalls. If we see a virtual network as a manifestation of a uniform locus of control and trust, then virtual networks are a helpful construct in understanding what resources are part of that locus of trust. A virtual network may be as much an organizing tool as a protection. But if a component inside the virtual network turns out to be untrustworthy, we still need a tool to isolate ourselves from that component.

Another topic of discussion was application design. Some applications today, such as email, have an

architecture that routes traffic through servers on its way from sender to receiver. This has allowed receivers to out-source spam and virus checking to a trusted server. The design of email gives the user (in most cases, in principle) choice over which server to use, which allows users to express their trust preferences. This is an example that we might study (for its strengths and weaknesses) in order to offer guidance to application designers and to look for common application services that we might provide as part of a future architecture. It was noted that many application designers are not especially sensitive or well-trained in issues of security, and to the extent that we can move security-related services down to a support layer, we may improve the overall situation. Again, the question of trust seems to be central to this design. Some participants seem to like the idea that protection (or other application services) should be outsourced "to the network", to capture the idea that the function may be distributed and not at a specific physical location, but other people preferred the image of outsourcing "to a service", not because of its physical specificity, but because they want to capture the idea of picking among a choice of service providers.

One of the architectural implications of "outsourcing" is that protocols at all levels might well be re-designed from the perspective of facilitating various sorts of checking. For example, the initial handshake of TCP might be redesigned to allow a node in the network (one trusted by the receiver) to perform an initial validation of source identity and so on, before any state in the destination is created. This same principle may apply across the levels from packet to application.

And of course, any element onto which a function has been outsourced itself becomes a target for attack. An architecture might try to deal with this by limiting the visibility (e.g. the addressability) of the elements, or otherwise preventing attackers from sending targetted traffic to them. There is an old saying that computer scientists can solve any problem except performance by adding a layer of indirection. This discussion of outsourcing of services is a form of indirection, and we need to think hard about whether the use of services (and the design of applications to facilitate the use of services) shifts the landscape of security in any fundamental way.

The discussion then shifted from prevention to detection and recovery. What should the response of "the network" be to the discovery of a machine that has become a zombie, for example? Several participants asserted that it is practical for a network operator to detect that an attached end-node has become a zombie. In the university environment, there is a tradition of cutting infected machines off the network, but in the consumer market, this approach is not usually followed, perhaps because the ISP would not be able to deal with the flood of resulting calls to their help desk. And if they cut off a machine, they have to deal with the question of when to reconnect it, and so on. Actions in the network to protect the host may be more effective (and more usable) if they are embedded in a larger context of management and control. Thus, for example, if a host that is cut off because it is infested with a zombie is connected to a restricted virtual network that gives the user access to information and to tools to clean up the system, the process of cutting off and then reconnecting a end-node to the network can be transformed from a frustrating help-desk call to a positive opportunity to "help the user with his problems", an opportunity that might even be monetized.

4.6 Lessons from IPv6

Steve Bellovin spoke on the lessons to be learned from the IPv6 development and deployment process. The effort started in 1992–1993. The problem IPv6 was intended to solve was the shortage of IPv4 addresses. There were several competing proposals on the table; there was no consensus on which direction to take.

Several other things affected the situation. Internet security—which is not the same as cryptography—was starting to become an issue. There was no host autoconfiguration; the first DHCP RFC was issued in

late 1993. There were no NATs. Routing table size was becoming an issue. And the OSI versus TCP/IP war was in full swing.

The IETF responded by forming the IPng directorate. It was decided to keep the basic semantic model of IPv4. Routing table size, though known to be important, was not to be the focus of the group; if renumbering was easy enough, mandating CIDR-style addressing would solve that problem. Mobility, multicast, and security (or rather, IPsec) were mandated as part of any solution.

The basic decisions were easy enough, but *engineering* the protocols was a lot harder than expected. Many features were added; some caused complexity in other areas. Scoped addresses changed the socket API, Neighbor Discovery replaced ARP and included basic autoconfiguration, flow labels were added (though their usage wasn't specified), and the early decision on autoconfiguration froze part of the address format.

The ultimate claims for IPv6 did not attract many people. It does have autoconfiguration, but in the interim, DHCP deployment for IPv4 became universal. IPsec exists for IPv4. And, though IPv6 does have bigger addresses, that doesn't attract end-users.

There were unanticipated interactions. Neighbor Discovery could not be secured with IPsec. Site-local addresses had to be replaced with a different scheme, because they interacted poorly with the DNS. Multihoming is still an unsolved problem. Renumbering, though easier, isn't easy; there are too many address in ACLs, configuration files, etc. Besides, developments such as NATs, tight limits on address allocation, and the evolution of IPv4 solved many of the problems IPv6 was aimed at.

In short, development took longer than anticipated and the rest of the world did not stand still.

4.7 Breakout Sessions

There were three breakout sessions, on host versus network responsiblities, naming and addressing, and network management. The conclusions reported in this section are those of the breakout group, rather than of the workshop as a whole.

4.7.1 Host versus Network Responsibilities

We assume that there will always be some malicious hosts and routers, either compromised machines or ones actually run by malefactors. There will be enough such machines to overload bottleneck links, because the net will remain heterogenous in bandwidth, technology, and computational capacity. The net will also remain open, decentralized, and international.

The primary responsibility of the network is availability for critical applications, even under emergency conditions. This may require some form of bandwidth reservation, which in turn requires suitable authentication and authorization.

Some form of generalized policy enforcement idea seems to be a good idea. There are many sources of policies — users, machine and network owners, and governments. This creates complex authentication and authorization scenarios, and the potential for bad societal effects on such values as privacy and anonymity. Furthermore, even if such boxes do perform filtering, it may be simply an optimization; hosts still need to protect themselves.

Based on the success of NAT, it seems like a new architecture must provide the perceived and real benefits of NAT boxes, though hopefully without their present architectural disadvantages. That will require

redesigning rendezvous and separation of location and identity. However, the entire concept of identity, especially on a global scale, needs to be re-examined.

4.7.2 Naming, Addressing, Forwarding, Routing, and QoS

The primary responsibility of the network is to deliver data to the correct destination, despite failures or malicious activity. To this end, it must be able to verify routing, at all layers. Some form of resource control is needed in the forwarding path, for QoS, isolation, etc. Hosts should have the ability to influence path selection, but not at the expense of security.

Note the various tradeoffs. If we have ubiquitous end-to-end encryption, what security is needed in routing and forwarding? Alternatively, is good network security a strong-enough alternative to end-to-end encryption? What of encrypted tunnels? Do they cause other problems? Do forensics compromise security or privacy?

The notion of topology-based addressing should be re-examined. Included in that decision are the need for private or scoped addressing, and a separation of identity and location.

We also need to consider economic forces. Technical measures cannot override these. Many of today's abstractions come from today's practices; these include economic constructs such as policy routing by ISPs.

The entire design space should be re-examined. This may require changes to NSF's review process. We should explore a variety of design alternatives.

4.7.3 Network Management

Network management includes developing policy, implementing it via the control plane, and examining information retrieved from the control plane. Management occurs under both normal and abnormal conditions, and during transition due to reconfiguration or other change. All management data must be adequately secured; in addition, the control plane should include *safety* checks as well, to guard against clearly erroneous changes.

Management should be scalable; running a 10,000 node network should not be harder than running a 10 node network. Systems should reason about the future — what will happen? — and the past: what did happen? Information should flow between layers and devices, and should be adaptable to changes in the organization.

Research directions include enhancing the control plane to do more of the management work, with due attention to security and efficiency. Other areas of computer science, such as software engineering and model checking, may have useful insights.

The group suggest embracing network management as a critical research domain, with aid from network managers and equipment vendors. A testbed and research programs should be started.

4.8 Social Reflections on Security

Security can be seen as a technical problem, but in fact the concept of security is a social concept, and our view of objectives and threats arises in the larger context in which the Internet sits. Technical systems make design choices that express values in a larger social and economic sphere. For this reason, we need to embed the study of social values into a study of technical features. At this state of maturity for the Internet, there are few design decisions that are purely technical.

Helen Nissenbaum provided a framework to define the interplay between technologists and people from other disciplines. The process of embedding social considerations into technical design involves a process of discovery, translation and verification. We must work to discover and make explicit the larger values implied by design choices, we must translate between the social and the technical worlds—for example we need to operationalize social concepts such as privacy. We must design technology in accord with our social objectives, and finally, we must develop methods to validate that our designs will capture the desired social outcomes. Many methods of social science research assume that the artifacts have been built, and can be studied after the fact to discover their larger implications. Our goal here is to do the best we can to predict during the design process what the implications of design alternatives will be.

Nissenbaum then provided a list of topics that folks from other disciplines regularly study—topics that may be of direct relevant to a new architecture and to security in particular.

- Security as a social value, and the relationship of security to social engagement, and concepts of system robustness. Tradeoffs between security and functionality.
- Locus of control. Centralization vs. decentralization. Political science has been thinking about this for a long time with respect to governance.
- Tradeoffs between freedom and constraint. (Freedom is best expressed within constraints.)
- Privacy as a social concept.
- Security as a private vs. a public good.

The technical community has long understood that our artifacts (such as the Internet) are used in different contexts that have very different requirements. This discussion reminds us that the contexts are social, not just technical, and that our challenge is to design architectures that will yield the best technologies from a social perspective, not just from a technical one.

Deirdre Mulligan used privacy as an example of tradeoffs in the design space, looking at technology and law. She made the point that as new technologies are introduces, we must continuously evaluate the relationship between capability, expectation and legal rules. For example, people's expectation about what sort of copies they can make of music they own is being rapidly adjusted by today's legal, commercial and technical changes.

She offered some insights from the law (at least the U.S law) that we might keep in mind. Where information is stored is very material with respect to who can see it and under what circumstances. Management tools that focus on accurate and precise detection of illegal activity will be easier to justify as tools for policing than broadly-revealing tools. A novel tool built for use by police may be seen as invasive if it reveals new sorts of information, but if it enters into general use, then it will be deemed suitable for police use as well.

4.9 Some Final Thoughts on Architecture

John Wroclawski observed that the motivation for a future architecture will be on how the network can protect itself and be tolerant of new requirements. The problem is not to support "hot new applications"; the Internet today is pretty good at supporting applications. The redesign will address issues such as robustness, which will benefit all the applications, both current and future. While we may think of our architecture as

defining how the network itself is built, we must think in the context of an overall system that includes both the end-nodes and the humans that use and control them. Finally, the Internet is an economic playground now, and a future architecture must take this into account. We cannot design for a rigid outcome, but to allow a tussle among stakeholders, so that the tussle does not distort the architecture.

What is architecture? What survives over time, and what shapes progress? Here, for example, are two views of tunnels. One is that tunnels are a reflection of a failure of architecture—an architecture that cannot deliver what the user wants. Another view is that tunnels are a tool in a flexible architecture—a means to tailor and evolve an architecture without breaking it.

Two perspectives emerged several times. One is that it is very hard to talk about the security implications of an architecture that does not yet exist. The other is that a future architecture may not look at all like today's Internet. The "three-level" model of the Internet, the "hour-glass" construct, and so on, may all be missing from a future design. In this context, how does a "security expert" engage the process?

There was a recurring set of comments (not perhaps well developed, but recurring) about the relationship of architecture and power. It was noted that while mechanisms such as source routing could in principle shift power to the edge, today larger players such as the ISPs and third parties such as Akamai hold much of the power and are in a tussle over that control. How can architecture (if at all) be relevant to that tussle?

Many of the trends in the evolution of the Internet involve more state in the net. We have no security theory about how to manage, protect or reason about this state. But the model that the Internet is stateless, and can just be rebooted to clear up messes and get it into a stable state, is probably no longer true. If the state is all soft, then a reboot will still more or less work, but the meeting had little confidence that this is true, or that, in practical terms, the Internet would come up again after a massive power failure or other disruption.

We lack both metrics and test/validation methods for security, so we don't have any method (other than being smart) to compare alternative proposals, or to test an architecture to see how secure it is.

Several security-related elements were nominated as possible components of a new architecture. These include firewalls or other filtering devices, traffic monitors (such as intrusion detection devices), and exchange points between separate regions.

It was noted that lots of security problems arise at the application level, but that many application designers are not trained as security experts. So we should search hard for common application support services and building blocks, and make these secure. To the extent that application design is done by service composition, we improve the chances of secure applications.

Since systems such as the Internet will be used in new and unexpected ways, people may try to use it in circumstances for which it is not intended. In particular, with respect to security, we have a responsibility to describe the space of intended use and the security implications, but what do we do to deal with the inevitable event that it is used outside that design space. Is there any way we can help potential users understand the implications of deploying the technology in different contexts? One analogy offered was that of warranties—which become void if the device is improperly used. But we want to encourage the creative and innovative use of our new technology.

5 Infrastructure and Testbed Needs for NGSI

5.1 Timeline

The initial infrastructure needs for developing the NGSI are modest. As it develops, however, there will be increasing need for more significant resources such as dedicated testing facilities and custom-developed hardware.

- The initial testing will be done as an overlay network on the current Internet, using off-the-shelf computers as network control and switching elements.
- As testing proceeds, dedicated, high-speed capacity will be needed, possibly accompanied by dedicated router ports. Some test sites may need higher-speed connectivity to the Internet or to Internet 2.
- To move to native-mode operation, dedicated fibers or lambdas will be needed. The National LambdaRail network is a good starting point, but it may need to be enhanced, especially with additional geographic diversity.
- Concurrently, hardware-based NGSI network switching elements will need to be developed. This may be a major effort, as their architecture may be significantly different from today's routers.
- A dedicated network will require an operations center. This center can also serve as the testbed for new network management paradigms.
- In addition to direct infrastructure, a number of support systems will be necessary. The most important is a set of repositories for NGSI code, both network layer and application layer. There should be similar repositories for hardware designs, including machine-readable chip and board layouts. Significant storage for datasets resulting from analyzing the network will also be required.

5.2 Security and Testbeds

Much of the security development necessary for an NGSI should not be carried out on the public Internet, nor on links shared with it. Newly-developed methods and (especially) code are likely to be faulty, and hence attackable. Tiger team attacks on, say, denial-of-service resistance may show that the mechanisms are flawed, thus flooding the link. However, if that link is in fact part of an overlay network, the underlying public link would be flooded, too. Accordingly, it is preferable to conduct such tests on a separate network.

6 Recommendations to NSF and the Research Community

The quest to build a secure next-generation Internet is not an easy one. There is no certain road-map for how to reach our destination and there are many possible paths. Navigating among them is itself a challenge. One of the outcomes of the workshop was guidance to NSF on how to proceed.

The primary question, of course, is architecture. The group felt that two or three collaborative teams should tackle the problem. Out of necessity, these teams would be quite sizable, with smaller teams addressing particular subtasks.

The design process involves three elements: free exploration of alternatives (with no constraints from the current Internet design); synthesis of the different ideas into a coherent architecture; and building real systems for deployment and evaluation.

Recommendation 1 *This last point must be stressed: the process should be based on running code.*

Although, as noted, the architectural teams will need to be large, arguably the process should start with smaller groups. This question merits further thought.

NSF probably needs to be heavily involved in the design process. It needs to be bold; its review panels are often far too conservative.

Although the system will be designed *de novo*, we obviously do not recommend that the process proceed in a vacuum.

Recommendation 2 *The lessons of the past* must *be studied.*

Among the questions to be answered in this vein are:

- What are the fundamental limits of the existing net?
- What are the new technologies that will help it address real problems?
- What characteristics should the new network have to help it take off?
- Why an incremental change approach is insufficient?

The lesson of IPv6 is particularly relevant to this last question.

Recommendation 3 *We must identify significant differentiators of this new network in order to provide a compelling case for the adoption of the new architecture.*

What are the security advantages of the new approach? Does it help eliminate large-scale attacks? Can we improve availability, especially in the face of an attack?

There are many things we cannot do today, or cannot do economically. Often, we know how to do them, but the set of incremental changes to enable them is too costly. For example, we cannot do remote surgery over the network. We can, in some sense, do online banking; however, the security risks are too significant for many potential users.

Network management today is very expensive, as much as half the cost of operating a network. Can we reduce the cost?

Today's network isn't that reliable. Can we make it so? People do not worry if water or power will be available; on the contrary, outages are notable precisely because they are so rare.

Many people would like more privacy and anonymity on the network. On the other hand, we would also like to hold malefactors responsible for their misdeeds. There is a tension between these different desires. Can a suitable compromise be found? Can anonymity be made conditional, but in a safe way?

We need a secure namespace. DNS is inadequate and it is unclear if DNSSEC will suffice.

We must retain today's ability to support new, innovative applications. A network that discourages innovation is at best useless and at worst harmless. However, it must be done without invalidating the security assumptions the new network is designed on. Reconciling these two points is a challenge. The network should also assist end-hosts in deflecting network-based attacks such as DDoS.

Some other aspects of the current network should be retained. Packets are probably a good idea; the ability for end-hosts to deploy new applications/services without the involvement of the network is definitely worth keeping.

Recommendation 4 *This effort should be broad in scope, and involve other communities in Computer & Information Science & Engineering (CISE), and other areas.*

Recommendation 5 *Privacy considerations should be designed in from the beginning.*

Even apart from the tension between privacy and accountability, privacy is too hard to achieve in some contexts because of other design decisions.

Recommendation 6 *The social implications of designs should be considered explicitly, from the very beginning.*

It is obviously impossible to foresee all of the societal effects of a design, however, we do not want to stumble into a preventable trap simply because of a design that was technically attractive.

Recommendation 7 *Lower the risk in the research community.*

This effort needs to be an open one with widespread community support. There should be frequent papers in conferences, journals, etc.

Recommendation 8 *Combine the understanding of existing systems with innovative "out of the box" ideas.*

We cannot turn a blind eye to what is happening in today's Internet, but we do not want to be constrained by it.

A Attendees

David Andersen	CMU	Deirdre Mulligan	U.C. Berkeley
Paul Barford	U. of Wisconsin	Helen Nissenbaum	NYU
Steven Bellovin	Columbia	Guru Parulkar	NSF
Azer Bestravros	Boston U.	Adrian Perrig	CMU
Matt Blaze	U. of Pennsylvania	Larry Peterson	Princeton
Dave Clark	MIT	Jennifer Rexford	Princeton
Richard Draves	Microsoft	Srinivasan Seshan	CMU
Darleen Fisher	NSF	Jonathan Smith	DARPA, Penn
Virgil Gligor	U. of Maryland	Dawn Song	CMU
Brian Hearing	DARPA	Ion Stoica	U.C. Berkeley
Dina Katabi	MIT	Gene Tsudik	U.C. Irvine
Stephen Kent	BBN	Jesse Walker	Intel
Carl Landwehr	NSF	John Wroclawski	USC ISI
Wenke Lee	Georgia Tech	Hui Zhang	CMU
Allison Mankin	Shinkuro	Lixia Zhang	UCLA
David McGrew	Cisco	Taieb Znati	U. of Pittsburgh

B White Papers

Below is a list of the submitted white papers; they all can be found at `http://www.cs.columbia.edu/~smb/ngsi-whitepapers`.

Paul Barford	*Measurement as a First Class Network Citizen*
Azer Bestavros	*Towards Trusted Adaptation Dynamics in Computing Systems and Networks*
Bob Braden, Terry Benzel, and John Wroclawski	*Some Considerations for Secure-Architecture Experimental Infrastructure*
David G. Andersen	*Defining The Secure Internet*
Dina Katabi	*Designing Defense Systems Against Unwanted Traffic*
Wenke Lee, David Dagon, and Guofei Gu	*Quality-of-Security Aware Internet*
David McGrew	
Deirdre Mulligan	
Helen Nissenbaum	*Security as a Technical Ideal and a Social, Political and Moral Value*
Jennifer Rexford	*Securing the Routing System at All Levels*
Srinivasan Seshan	*A Resilient Routing Infrastructure*
Ion Stoica	*Host-controlled Routing Architecture*
Gene Tsudik	*Privacy-Preserving Security Services and Protocols*
Jesse Walker	
John Wroclawski	*Redundancy and Location Information in a Robust Network Architecture*

C Agenda

Tuesday, July 12, 2005

Opening remarks that frame the course of the workshop

- NSF's Context and Charge to the Community (Darleen Fisher)
- Network Architecture Overview and Context (David Clark)
- Security Overview and Context (Adrian Perrig)

Architecture/security co-design (Stephen Kent)

Futuristic Architecture Ideas (John Wroclawski, Larry Peterson, Hui Zhang, and Ion Stoica)

Discussion Session 1: Addressing and Naming

Discussion Session 2: Endhost vs. Network's Responsibility for Security

Lessons Learned From IPv6 (Steven Bellovin)

Wednesday, July 13, 2005

Discussion Session 3: Routing, Forwarding, and QoS

Discussion Session 4: Network Management

Discussion Session 5: Recommendations to NSF and the Community

Breakout sessions

Report—Discussion and Initial Draft

Thursday, July 14, 2005

Infrastucture for Secure Network Architectures

- NSF plans for infrastructure acquisition (Guru Parulkar and Larry Peterson)

Discussion of NSF Initiative and Community Recommendations

Wrap up

D Cryptographic Challenges for NGSI

Although not discussed as a separate topic, a variety of cryptographic assumptions underly much of the discussion. Below we present the authors' understanding of these issues.

Just as considering a NGSI requires a fresh look at networking protocols, a fresh look at cryptographic building blocks may be an important step to developing a more secure yet flexible next generation network. In particular, current cryptographic techniques may not apply to NGSI security goals due to: (1) computational inefficiency, (2) unexpected protocol interactions, (3) communication limitations, or (4) conflicts between requirements of different cryptographic components. Seemingly simple and cryptographically provable protocols often run into one or a combination of the above problems when deployed in real systems. For these reasons, it is essential to take a "clean-slate" approach not only to network design issues, but also to the development of cryptographic protocols and cryptographic primitives.

The primary areas are identity management, secure routing, secure name resolution, and support for end-host security. Examples of specific tasks to be explored include: (1) the use of novel cryptographic techniques to fight denial of service attacks; (2) low-cost revocable anonymity that can coexist with security; (3) the use of cryptography within well-designed economic mechanisms to encourage good behavior by non-adversarial but selfish users; (4) cryptographic protection of forensic data from tampering; (5) alternatives to PKI infrastructure that avoid single points of failure yet remain cryptographically secure and efficient. We stress that solutions must be developed not as stand-alone applications, but rather as tools that even with multiple invocations and concurrent use will remain secure and resilient. The specific requirements for these cryptographic primitives must be designed with the entire NGSI architecture in mind.

E Design Assumptions and Methodology

Near the beginning of the workshop, Adrian Perrig gave a talk to frame the debate. This section is derived from that talk.

Security assumptions: We assume that while not all future network services will have stringent security demands, the next generation network architecture must provide a foundation to support the deployment of so-called critical infrastructure applications. With this new architecture we seek to provide an economically viable platform that is capable of supporting the security components of availability, confidentiality, integrity to the degree these features are required by higher-level applications. Of these three considerations, we view the need for robust availability as the security requirement most directly dependant on network architecture.

When considering the diverse security needs of different network applications, we also recognize that security goals can at times be contradictory. As a simple example, source address confidentiality may make anonymous communication possible but it can also be abused to hide the source of an attack. To the degree possible, we want to create a flexible architecture capable of supporting varied levels of security demands. In this way, requirements of participants for either privacy or accountability can be negotiated for individual communication sessions depending on the scenario's specific benefits and costs.

Attacker assumptions: We assume a strong attacker model, where end–hosts, network switching elements, and even entire service providers or network domains may be compromised and collude with each other. Attackers have resources varying from those of teenage hackers to those of organized crime and nation-states. This is arguably the strongest attacker model, where attackers could potentially corrupt any node in the network and be adaptive to new settings and countermeasures, yet we feel it is warranted based on the likely demand for a future network to support/ highly critical services.

Heterogeneity assumptions: We assume that the network will be composed of heterogeneous nodes which may have different computational, storage and communication capacities. Support for such diversity has been a fundamental reason for the Internet's success and the increase in hand-held and embedded devices suggests that device diversity will similarly be important to the success of future networks.

Trusted hardware assumptions: The availability and deployment of cryptographically trusted hardware continues to increase dramatically as does the number of applications seeking to utilize its desirable security guarantees. Trusted hardware can serve as an important root of trust and can significantly simplify the design of security mechanisms. We assume the availability of trusted hardware solutions as a potential building block for end-hosts and network elements.

E.1 Design Space

In general, there are four categories of approaches to network security: *prevention*, *detection and recovery*, *resilience*, and *deterrence*.

Prevention: The prevention approach seeks to harden the network elements and protocols to render attacks infeasible or at least sufficiently difficult or costly to deter an attack. Many mechanisms achieve prevention by incorporating cryptography, but additional mechanisms including verification and isolation

can be utilized to effectively mitigate vulnerabilities. Preventive mechanisms are often highly attractive due to their low error margin and efficiency.

Detection and recovery: Detection involves monitoring of the real-time behavior of a protocol or device in order to recognize improper behavior. Once malicious behavior is detected, we resort to recovery techniques to curtail the malicious behavior and restore network order and functionality that may have been impaired.

Resilience: The resilience approach seeks to maintain a certain level of availability or performance even in the face of active attacks. For example, it is desirable for network performance to gracefully degrade in the presence of compromised network participants. Examples in this category include redundancy mechanisms, such as multipath routing or distributed lookup servers.

Deterrence: Historically, it has proved impossible to prevent all attacks. Notably, many security problems are due to buggy or misconfigured software; a problem that the network has only a limited ability to mitigate. If such attacks cannot be stopped by the network, it is at least desirable for the network to assist in deterring such attacks. Legal mechanisms can be used to provide disincentives for attackers. With effective attacker tracing and effective international laws and enforcement, risk-averse rational attackers may shy away from attacks. However, such deterrence implies a need to be able to attribute an attack to some party, which has obvious implications for the delicate balance between privacy and order. We would like to use cryptography or other technical mechanisms to help ensure an appropriate balance.

A solid security architecture for the next generation Internet will utilize a combination of all four approaches to achieve a more secure network architecture.

E.2 Design Guidelines

We propose an initial set of design principles that serve as guidelines for a design of a secure Internet architecture.

Minimal trust requirement: Information sharing is essential for coordination between network participants. However, a design trade-off exists, since the more heavily one participant's behavior relies on information from other participants, the more damage a single compromised participant can conceivably introduce. We are interested in security mechanisms that assume only minimal trust between network participants. Ideally, each node should trust only itself; hence, some proposed security mechanisms enable routers to make independent decisions without reference to other routers. When sharing information is inevitable, we can use cryptographic mechanisms to impose restrictions on the types of transformations a network participants can perform on shared information.

Minimal state in network elements: Even though the new generation of VLSI technology has enabled us to produce extremely powerful network devices such as switches and routers, requiring minimal state at each network element is still a desirable property. In particular, our goal is to use small or even constant state in network elements instead of storing per-flow or per-end-host state. This not only ensures the scalability of network elements, but also simplifies design and minimizes cost.

Minimal network layer functionality: In network design a well-established principle is the end-to-end principle, which advocates placing functionality at the end-host, particularly if the end-host has more complete information allowing it to make better decisions. Similarly, in designing security mechanisms for the future Internet, we seek to answer the following question: what is the minimal functionality the network layer should provide to ensure enforceable security?

Simplicity: A good design provides simplicity at several levels. Simplicity at the conceptual level facilitates dissemination and discussion of new architectures within a broad audience. Operational simplicity for configuration and setup makes network elements and protocols easier to use and more transparent for debugging while also lowering costs.

Economic viability: Many ISPs are currently facing rough economic times, and lower costs for managing and maintaining a network would represent an important reason for adopting a next-generation secure Internet. In fact, secure protocols provide robustness against benign misconfigurations in addition to those that are malicious, potentially increasing network uptime and lowering costs. Moreover, ISPs may be able to charge more for enhanced security services, and Internet connected businesses may benefit from a reduced exposure to attacks.

Provable security: Whenever possible, the security of important network protocols should be analyzed through formal method techniques. In some cases, the security can even be formally proved. These approaches provide a higher level of assurance than protocols analyzed in an ad-hoc fashion.