Rethinking the design of the Internet

David D. Clark

MIT CSAIL

CFP/CRN winter meeting

January, 2006

Topics:

- Description of FIND, a National Science Foundation initiative.
  - Why do this?
- Further motivation--security as an example.
- Relationship to another NSF project--a major infrastructure proposal called GENI.

FIND: A challenge question

1) What are the requirements for the global network of 10 or 15 years from now, and what should that network look like?

To conceive the future, it helps to let go of the present:

2) How would we re-conceive tomorrow’s global network today, if we could design it from scratch?

- This is not change for the sake of change, but a chance to free our minds.

Isn’t today’s net good enough?

Must start with serious discussion of requirements:

- It’s not just about cool new apps.

Security and robustness.

- As available as the phone system
- Been trying for 15 years--try differently?

Easier to manage.

- Really hard intellectual problem
- No framework in original design.

Recognize the importance of non-technical considerations

- Consider the economic landscape.
- Consider the social context.

What will be happening in 10 years

New network technology.

- Wireless
  - Mobility
  - Dynamic capacity allocation
  - Dynamic impairments
- Advanced optics
  - Dynamic capacity allocation (again!)

New computing paradigms

- Embedded processor, sensors, everywhere

Whatever computing is, that is what the Internet should support.

- The Internet grew up in a stable “PC” time.

FIND: it’s not a new IP

Perhaps a header format is not the defining piece of a new architecture.

Perhaps we focus on control, management and “other planes”.

- Data plane can fend for itself.

The “old” Internet

- Packet format.
  - Trying to replace that…
- Global addresses.
  - Broke that…
- Oblivious transport (end to end).
  - Eroding…
- Hosts are not routers (don’t run routing protocols)
  - Starting to break that…
- BGP (EBGP)
  - Talking about replacing that.
- DNS
  - Broke much of that…

The search for generality

(Or, the search for “open”.)

How do you make a “general” system?

Never commit to what it does.

- Commitment may “freeze” the system.

Design (architect) cool building blocks and hope someone can arrange them later.

- Run-time architecting.

We do this all the time.

QoS as an example

Two approaches to specification.

- Per-hop behavior (PHB), composable along a flow to get overall semantics.
  - How is behavior composed? (Flow setup?)
- Defined end-to-end behaviors
  - TCP-friendly rate adaptation.

This tension between approaches is basic.

Security--another example

- A firewall is a Per Hop Behavior.
  - It tells you nothing about how you achieve good overall security, or what security you can achieve.
- Alternative: some sort of “negative availability principle”.
  - If one set of nodes doesn’t want some other set of nodes to talk to them, the network should enforce that (or “help” enforce).
  - A bold, dangerous idea…

Consider economics

What does an ISP sell? What do I buy?

PHBs are (relatively) easy to create, but are they worth much?

Selling an end-to-end service seems like more value, but is hard.

- Have to agree on what the service is.
- Requires cooperation on service creation, revenue allocation, etc.

Consider the current work in ITU.

Design for today or tomorrow?

Design for today:

- Support the known apps.
- Make money.
- Embrace services.

Design for tomorrow:

- Don’t block innovation.
- Find cool building blocks.
- We have lots of honed tools for this purpose:
    - End-to-end arguments.
    - Run-time application adaptability.
    - Weak semantics.
    - Open interfaces.

My proposal for a design goal

We should design for tomorrow--design for change and for the application we have not seen yet.

We should pay more attention to how building blocks are composed into services.

We should think about how an architecture can itself survive change.

Security as an example

Define security generally.

- Not just disclosure control and integrity.
- Focus on availability
- Focus on user expectations of a safe and understandable experience.

Include the end node.

- Don’t assume that bad end-node security is someone else’s problem.
- Don’t assume the net will solve the whole problem.
- Need a reasoned division of responsibility.

More requirements

Security must be usable (so it gets used) and understandable.

There are different needs for security in different contexts.

We need an *architecture* for security.

- But note the point above.
- Architecture that supports late binding.

Security for tomorrow’s devices.

Sensitivity to social context and needs.

Resilience and availability

Security community has tradition of looking to resistance. Resilience may be a better path.

- Diverse failover modes
- Reduced interdependence under attack
- Integration with management
  - No silent failures
- Support for variability
- Resilient social structures
- Other disciplines?

Deterrence

Social form of question: what is the role of policing in the Internet?

Technical form of question: what should it be possible to see where?

Models of policing:

- Wait to be called.
  - Can end-node gather evidence? Witnesses?
  - Can application design prevent classes of crime?
- Feet on the street, cameras.
- CDC
- Contract law and arbitration.

Identity

At the packet level:

- If we cut loose from location, then what?
- Regional variation?
- Access to resources?

At the app (e2e) level.

- Cross application architecture?
- Bottom up or top down?
- Should “the net” play a part at the application level?
  - The “out-source” model of protection.
  - What about e2e at the application layer?

DoS

Proposal: distinguish “public” and “closed” servers.

For public: must diffuse.

- Speculation: diffusion will be key part of future.

For closed, outsource protection.

- Whom do you trust?

Possible research questions:

- Do private address spaces help?
- Virtual nets?
  - Must protect the real assets underneath…
- Re-architect protocols for these goals?

Protecting the end node:

The negative availability principle

- Architecting the firewall.
- Relate to identity (“Danger, danger…)

The virtual machine story

- Do virtual machines need virtual nets?

Helping the host protect itself.

- Trusted path, logging, reference monitor.

Avoid the two fates?

Virtualization and security

What do virtual networks really buy us?

- Will we have a few or millions?

What do private networks really buy us?

What do overlay networks really buy us?

- What are the necessary security requirements for the underlay?

Close links to management.

Protocol design for management and security.

No silent failure.

Make management systems secure.

- Hard--they must work when all else is failing…

Goof-proof tools

- Necessary to boost availability.

Security should slow events to the point where humans can intervene.

Application level security

Identify common services.

Provide design patterns for secure design.

- Design so that first interactions can be outsourced and diffused.
  - Validate identity, etc.
- Design to gather evidence consistent with risk.
- Give user control over selection of services “in the application”.
  - Outsourcing.

What is GENI?

The GENI project (also an NSF initiative) is not funding for research.

It is infrastructure.

GENI is a proposal to build a wide area platform to demonstrate new ideas in networking and distributed systems.

- (Big bucks.)

Critical features

Wide area (core) and edge.

- Wireless platforms.
- Advanced optics.

Virtualizable, programmable.

- The concept of “slices”.

Motivated by the success of PlanetLab.

Linkages

FIND is starting now.

GENI is being proposed now.

NSF is building research and infrastructure connections now.

- Other agencies
- Industry
- International

Participation in CFP can give you a window into this.