Great Expectations

Protocol Design and Socioeconomic Realities

Dirk Kutscher <ietf@dkutscher.net>

University of Applied Sciences Emden/Leer

The Internet & Web as a whole qualify as wildly successful technologies, each of which empowered by wildly succesful protocols per RFC 5218's definition [1]. As the Internet & Web became critical infrastructure and business platforms, most of the originally articulated design goals and features such as global reach, permissionless innovation, accessibility etc. [5] got overshadowed by the trade-offs that they incur. For example, global reach – intended as enabling global connectivity – can also imply global reach for infiltration, regime change and infrastructure attacks by state actors. Permissionless innovation – motivated by the intention to overcome the lack of innovation options in traditional telephone networks – has also led us to permissionless surveillance- and mass-manipulation-based business models that have been characterized as detrimental from a societal perspective.

Most of these developments cannot be directly ascribed to Internet technologies alone. For example, most user surveillance and data extraction technologies are actually based on web protocol mechanisms and particular web protocol design decisions. While it has been documented that some of these technology and standards developments have been motivated by particular economic interests [2], it is unclear whether different *Internet* design decisions could have led to a different, "better" outcome. Fundamentally, economic drivers in different societies (and on a global scale) cannot be controlled through technology and standards development alone.

This memo is thus rather focused on specific protocol design and evolution questions, specifically on the question how technical design decisions relate to socio-economic effects, and aims at providing input for future design discussions, leveraging experience from 50 years of Internet evolution, 30 years of Web evolution, observations from economic realities, and from years of Future Internet research.

IP Service Model

The IP service model was clearly designed to provide a minimal layer over different link layer technologies to enable inter-networking at low implementation cost [3]. Starting off as an experiment, looking for feasible initial deployment strategies, this was clearly a reasonable approach. The IP service model of packet-switched end-to-end best-effort communication between hosts (host interfaces) over a network of networks, was implemented by:

- an addressing scheme that allows specifying source and destination host (interface) addresses in a topologically structured address space; and
- minimal per-hop behavior (stateless forwarding of individual packets).

The minimal model implied punting many functions to other layers, encapsulation, and/or "management" services (transport, dealing with names, security). Multicast was not excluded by the architecture, but also not very well supported, so that IP Multicast (and the required inter-domain multicast routing protocols) did not find much deployment outside well-controlled local domains (for example, telco IP TV).

The resulting system of end-to-end transport over a minimal packet forwarding service has served many applications and system implementations. However, over time, technical application as well as business requirements have led to additional infrastructure, extensions and new way of using Internet technologies, for example:

- in-network transport performance optimization to provide better control loop localization in mobile networks;
- massive CDN infrastructure to provide more scalable popular content distribution;
- (need for) access control, authorization based on IP and transport layer identifiers;
- user-tracking based on IP and transport layer identifiers; and
- usage of DNS for localization, destination rewriting, and user tracking.

It can be argued that some of these approaches and developments have also led to some of the centralization/consolidation issues that are discussed today – especially with respect to CDN that is essentially inevitable for any large-scale content distribution (both static and live content). Looking at the original designs, the later understood commercial needs and the outcome today, one could ask the question, how would a different Internet service model and different network capabilities affect the tussle balance [5] between different actors and interests in the Internet?

For example, a more powerful forwarding service with more elaborate (and more complex) per-hop-behavior could employ (soft-) stateful forwarding, enabling certain forms of in-network congestion control. Some form of caching could help making services such as local retransmissions and potential data sharing at the edge a network service function, removing the need for some middleboxes.

Other systems such as the NDN/CCNx variants of ICN employ the principle of accessing named-data in the network, where each packet must be requested by INTEREST messages that are visible to forwarders. Forwarders can aggregate INTERESTs for the same data, and in conjunction with in-network storage, this can implement an implicit multicast distribution service for near-simultaneous transmissions.

In ICN, receiver-driven operation could eliminate certain DoS attack vectors, and the lack of source addresses (due to stateful forwarding) could provide some form of anonymity. The use of expressive, possibly application-relevant names could enable better visibility by the network – however potentially enabling both, more robust access control *and* (on the negative side) more effective hooks for censoring communication and monitoring user traffic.

This short discussion alone illustrates how certain design decisions can play out in the real world later and that even little changes in the architecture and protocol mechanisms can shift the tussle balance between actors, possibly in unintended ways. As Clark argued in [3], it is important to understand the corresponding effects or architectural changes, let alone bigger redesign efforts.

The Internet design choices at a time were motivated by certain requirements that were valid at the time – but may not all still hold today. Todays networking platforms are by far more powerful, more programmable. The main applications are totally different as are the business players and the governance structures. This process of change may continue in the future, which adds another level of difficulty for any change of architecture elements and core protocols. However, this does not mean that we should not try it.

Network Address Translation

Network Address Translation (NAT) has been criticized for impeding transport layer innovation, adding brittleness, and delaying IPv6 adoption. At the same time NAT was deemed necessary for growing the Internet eco system, for enabling local network extensions at the edge without administrative configuration. It also provides a limited form of protection against certain types of attacks. As such it addressed shortcomings of the system.

The implicit client-initiated port-forwarding (the technical reason for the limit attack protection mentioned above) is obviously blocking both unwanted and wanted communication, which makes it difficult to run servers at homes, enterprise sites etc. in a sound way (manual configuration of port forwarding still comes with limitations). This however could be seen as one of the drivers for the centralization of servers in data centers (“cloud”) that is a concern in some discussions today. [4]

What does this mean for assessing and potentially evolving previous design decisions? The NAT use cases and their technical realization are connected to several trade-offs that impose non-trivial

challenges for potential architecture and protocol evolution: 1) Easy extensibility at the edge vs. scalable routing; 2) Threat protection vs. decentralized nature of the system; 3) Interoperability vs. transport innovation.

In a positive light, use cases such local communication and dynamic Internet extension at the edge (with the associated security challenges) represent interesting requirements that can help finding the right balance in the design space for future network designs.

Encryption

Pervasive monitoring is an attack [7], and it is important to assess existing protocol and security frameworks with respect to changes in the way that the Internet is being used by corporations and state-level actors and to develop new protocols where needed. QUIC is encrypting transport headers in addition to application data, intending to make user tracking and other monitoring attacks harder to mount.

Economically however, the more important use case of user tracking today is the systematic surveillance of individuals on the web, i.e., through a massive network of tracking, aggregation and analytics entities [6]. Ubiquitous encryption of transport and application protocols does not prevent this at all – on the contrary, it makes it more difficult to detect, analyze, and, where needed, prevent user tracking. This does not render connection encryption useless (especially not because surveillance in the network and on web platforms complement each other through aggregation and commercial trading of personally identifying information (PII), but it requires a careful consideration of the trade-offs.

For example, perfect protection against on-path monitoring is only effective if it covers the complete path between a user agent and the corresponding application server. This shifts the tussle balance between confidentiality and network control (enterprise firewalls, parental control etc.) significantly. Specifically for QUIC, which is intended to run in user space, i.e., without the potential for OS control, users may end up in situations where they have to trust the application service providers (who typically control the client side as well, through apps or browsers, as well parts of the CDN and network infrastructure) to transfer information without leaking PII irresponsibly.

If the Snowden revelations led to a better understanding of the nature and scope of pervasive monitoring and to best current practices for Internet protocol design, what is the adequate response to the continuous revelations of the workings and extent of the surveillance industry? What protocol mechanisms and API should we develop, and what should we rather avoid?

DNS encryption is another example that illustrates the trade-offs. Unencrypted DNS (especially with the EDNS client option) allows for a detailled monitoring of individual users' and families' behavior by on-path monitoring. DNS encryption on the other hand can prevent on-path monitoring – but it could effectively make the privacy situation for users worse, if it is implemented by centralizing servers (so that application service provider, in addition to tracking user behaviour for one application, can now also monitor DNS communication for *all* applications). This has been recognized in current proposals, e.g., limiting the scope for DNS encryption to stub-to-resolver communication. While this can be enforced by architectural oversight in standards development, we do not yet know how we can enforce this in actual implementation, for example for DNS over QUIC.

Future Challenges: In-Network Computing

Recent advances in platform virtualization, link layer technologies and data plane programmability have led to a growing set of use cases where computation near users or data consuming applications is needed – for example for addressing minimal latency requirements for compute-intensive interactive applications (networked Augmented Reality, AR), for addressing privacy sensitivity (avoiding raw data copies outside a perimeter by processing data locally), and for speeding up distributed computation by putting computation at convenient places in a network topology.

In-network computing has mainly been perceived in four main variants so far: 1) Active Networking, adapting the per-hop-behavior of network elements with respect to packets in flows, 2) Edge Computing as an extension of virtual-machine (VM) based platform-as-a-service to access networks, 3) programming the data plane of SDN switches (leveraging powerful programmable switch CPUs and programming abstractions such as P4), and 4) application-layer data processing frameworks.

Active Networking has not found much deployment due to its problematic security properties and complexity. Programmable data planes can be used in data centers with uniform infrastructure, good control over the infrastructure, and the feasibility of centralized control over function placement and scheduling. Due to the still limited, packet-based programmability model, most applications today are point solutions that can demonstrate benefits for particular optimizations, however often without addressing transport protocol services or data security that would be required for most applications running in shared infrastructure today.

Edge Computing (just as traditional cloud computing) has a fairly coarse-grained (VM-based) computation-model and is hence typically deploying centralized positioning/scheduling though virtual infrastructure management (VIM) systems. Application-layer data processing such as Apache Flink on the other hand, provide attractive dataflow programming models for event-based stream processing and light-weight fault-tolerance mechanisms – however systems such as Flink are not designed for dynamic scheduling of compute functions.

Ongoing research efforts (for example in the proposed IRTF COIN RG) have started exploring this space and the potential role that future network and transport layer protocols can play. It is feasible to integrate networking and computing beyond overlays, potentially ? What would be a minimal service (like IP today) that has the potential for broad reach, permissionless innovation, and evolution paths to avoid early ossification?

Conclusions

Although the impact of Internet technology design decisions may be smaller than we would like to think, it is nevertheless important to assess the trade-offs in the past and the potential socio-economic effects that different decisions could have in the future. One challenge is the depth of the stack and the interactions across the stack (e.g., the perspective of CDN addressing shortcomings of the IP service layer, or the perspective of NAT and centralization). The applicability of new technology proposals therefore needs a far more thorough analysis – beyond proof-of-concepts and performance evaluations.

References

[1] D. Thaler, B. Aboba; What Makes for a Succesful Protocol?; RFC 5218; July 2008

[2] S. Greenstein; How The Internet Became Commercial; Princeton University Press; 2017

[3] David Clark; Designing an Internet; MIT Press; October 2018

[4] Jari Arkko et al.; Considerations on Internet Consolidation and the Internet Architecture; Internet Draft https://tools.ietf.org/html/draft-arkko-iab-internet-consolidation-01; March 2019

[5] Internet Society; Internet Invariants: What Really Matters; https://www.internetsociety.org/internet-invariants-what-really-matters/; February 2012

[6] Shosanna Zuboff; The Age of Surveillance Capitalism; PublicAffairs; 2019

[7] Stephen Farrell, Hannes Tschofenig; Pervasive Monitoring is an Attack; RFC 7258; May 2014