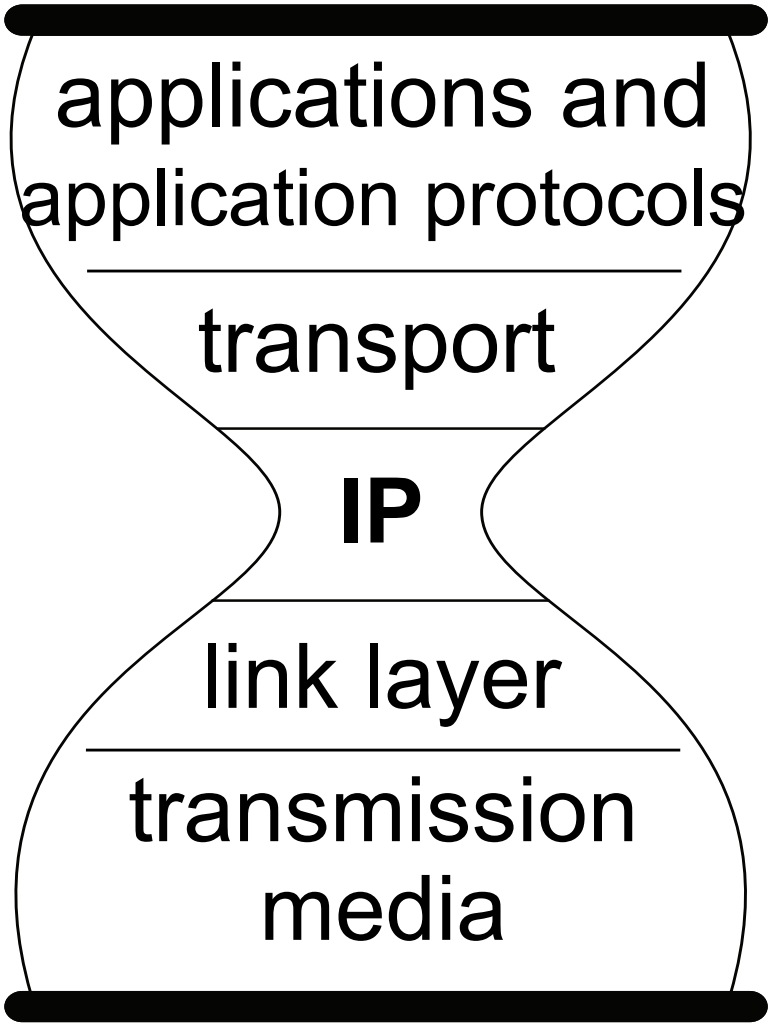
Session 1: Reconsidering Internet Architecture

"Internet Architecture": the definition ?

The well-known hourglass protocol stack



The narrow waist: the centerpiece in the architecture

Layers of Success

Oliver Spatscheck • *AT&T Labs — Research*

Even though it should be self-evident to everyone at this point, it's difficult to avoid repeating the fact that the Internet has grown beyond belief during my lifetime. Taking this as given, I'd like to focus on some of the Internet's increasing growing pains. I don't propose a solution to the problems I highlight, but being aware of a problem is often the first step toward fixing it.

The problem I want to focus on is layering. Before diving deeper into the topic, let me first define what I mean by "layering." Traditionally, the term has been used to define protocol layers; for example, the OSI protocol stack. However, here I use a broader definition that includes the protocol stack as well as every functional module that provides a well-defined interface or service on the Internet. For example, using this definition, the interfaces in a service-oriented architecture define boundaries between layers, too.

Layering has clearly contributed hugely to the Internet's success by allowing functionality to be introduced basically overnight; however, increasingly, it's hampering the reliability of the network it created.

Driving Innovation

Let's first examine some positives. Layering started early in the Internet. The IP layer provides only basic, host-to-host, best-effort connectivity. Protocols such as UDP and TCP addressed some of these limitations pretty much from day one. It didn't stop there, though. Layers such as the Secure Sockets Layer (SSL), Transport Layer Security (TLS), and IPSEC provide security, FTP and HTTP provide remote object access, SOAP and RPC offer remote code execution, and the GPRS Tunneling Protocol (GTP) provides cellular network support. These are only a few of the protocols used on top of the IP layer; we shouldn't forget the layers below, such as Multiprotocol Label Switching (MPLS) and the Ethernet Protocol (ETH). I could easily fill my allotted space just listing protocols; however, the important part to take away from this section is that many layers exist, and the ease at which we can quickly add layers to address a new need, often without any standards committee, is a major force behind the Internet's success. To highlight this point, note that many of the protocols I've listed were standardized in the IETF after they were already in widespread use.

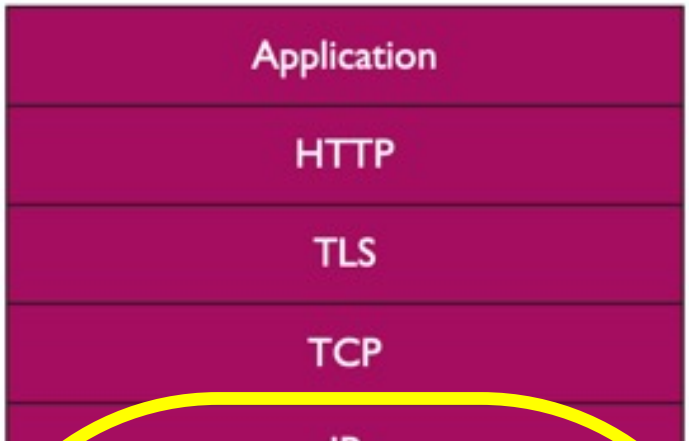
What's Wrong?

Let's now turn to the other side of the coin. The main issue with layering is that layers hide information from each other. We could see this as a benefit, because it reduces the complexities involved in adding more layers, thus reducing the cost of introducing more services. However, hiding information can lead to complex and dynamic layer interactions that hamper the end-to-end system's reliability and are extremely difficult if not impossible to debug and operate. So, much of the savings achieved when introducing new services is being spent operating them reliably. To make this point more clear, let's look at a case in which I was recently involved.

Company A was using a wireless machine-to-machine service for a mission-critical application and was experiencing a two-minute outage in one or two of their machine-to-machine systems every day – an unacceptable disruption considering the application's importance. This application had been built by third-party B, was hosted by a specialized cloud service provider C in yet another party D's data center, and was communicating with machines in the field using a cellular network E as the primary communication channel. Note that, in this case, every provider

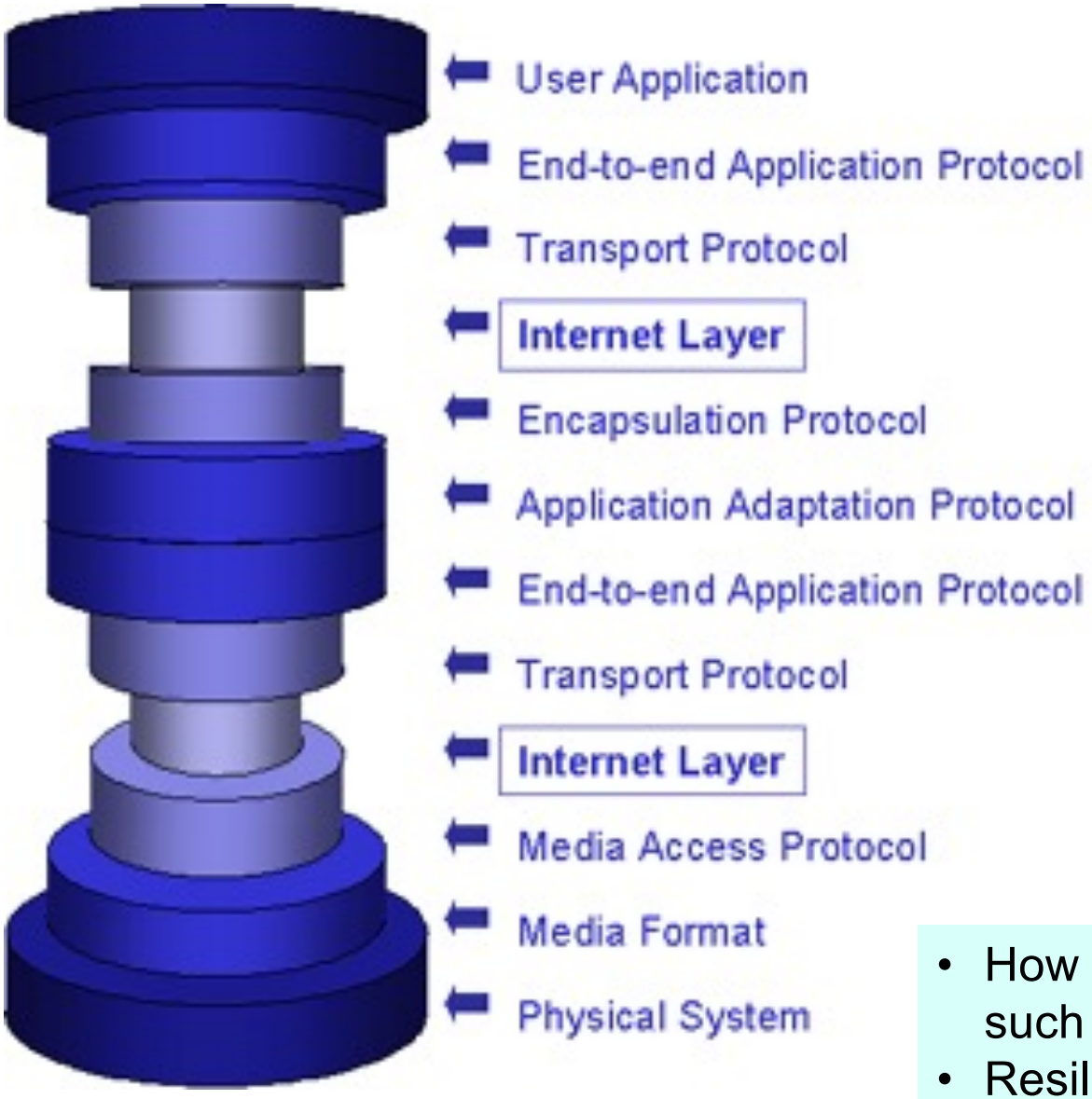
JANUARY/FEBRUARY 2013 1089-7801/13/$31.00 © 2013 IEEE Published by the IEEE Computer Society

2013



- Company A using a wireless M2M service for a mission-critical application
- experiencing a two-minute outage in one or two of their M2M systems every day
- the app is built by third-party B, hosted by a specialized cloud service provider C in yet another party D's data center
- communicating with machines in the field using a cellular network E as the primary communication channel
- Every party meeting its service-level agreement (SLA)

Consequence of virtualization?



- How easy to debug such a system?
- Resiliency?

Internet Engineering Task Force (IETF)
Request for Comments: 9298
Category: Standards Track
ISSN: 2070-1721

D. Schinazi
Google LLC
August 2022

Proxying UDP in HTTP

Abstract

This document describes how to proxy UDP in HTTP, similar to how the HTTP CONNECT method allows proxying TCP in HTTP. More specifically, this document defines a protocol that allows an HTTP client to create a tunnel for UDP communications through an HTTP server that acts as a proxy.

"UDP tunnels are commonly used to create an end-to-end virtual connection, which can then be secured using QUIC [QUIC] or another protocol running over UDP."

Internet Engineering Task Force (IETF)
Request for Comments: 9484
Updates: 9298
Category: Standards Track
Published: October 2023
ISSN: 2070-1721

T. Pauly, Ed.
Apple Inc.
D. Schinazi
Google LLC
A. Chernyakhovsky
Google LLC
M. Kühlewind
Ericsson
M. Westerlund
Ericsson

Proxying IP in HTTP

Abstract

This document describes how to proxy IP packets in HTTP. This protocol is similar to UDP proxying in HTTP but allows transmitting arbitrary IP packets. More specifically, this document defines a protocol that allows an HTTP client to create an IP tunnel through an HTTP server that acts as an IP proxy. This document updates RFC 9298.

This document describes a protocol for tunnelling IP through an HTTP server acting as an IP-specific proxy over HTTP. This can be used for various use cases, such as remote access VPN, site-to-site VPN, secure point-to-point communication, or general-purpose packet tunnelling.

Workgroup:	Multiplexed Application Substrate over QUIC Encryption
Internet-Draft:	draft-ietf-masque-connect-ethernet-01
Published:	19 October 2023
Intended Status:	Standards Track
Expires:	21 April 2024
Author:	A. R. Sedeño
	Google LLC

Proxying Ethernet in HTTP

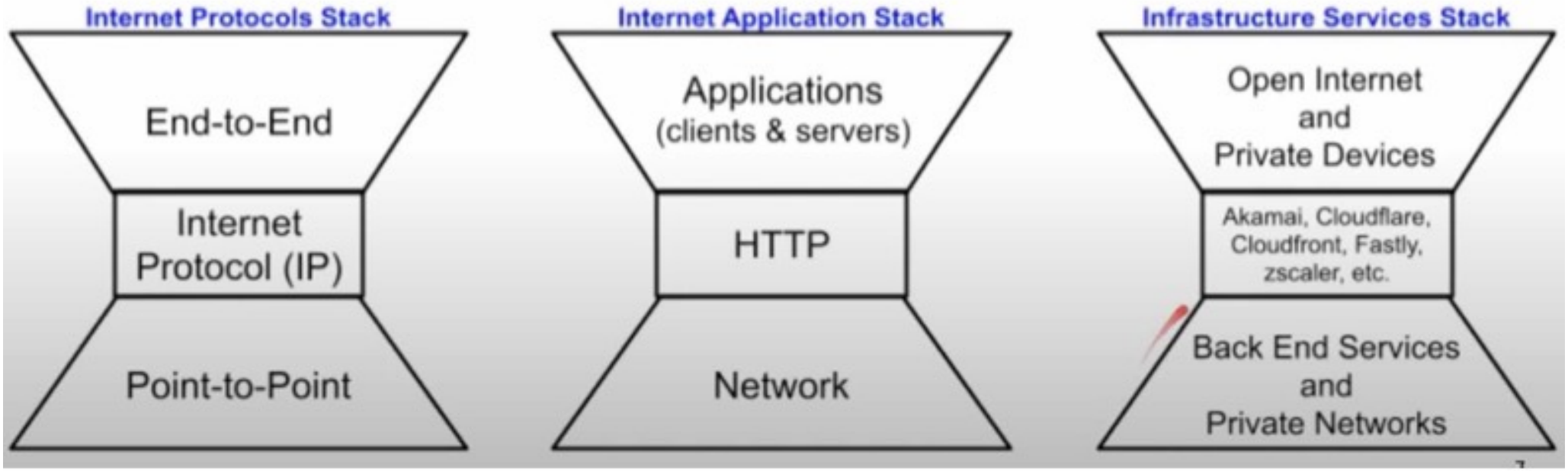
Abstract

This document describes how to proxy Ethernet frames in HTTP. This protocol is similar to IP proxying in HTTP, but for Layer 2 instead of Layer 3. More specifically, this document defines a protocol that allows an HTTP client to create Layer 2 Ethernet tunnel through an HTTP server to an attached physical or virtual Ethernet segment.

A recent drawing of the hourglass

Lookahead: establish a new stack and narrow waist ?

- Define interoperability for an "infrastructure services stack"
  - What services? e.g. caching, DoS/IDF, zero-trust, etc.
  - What interfaces and abstractions?



From the networking channel panel 11/8/2023 on "*Lessons learned from 40+ years of the Internet: an Industry Perspective*", https://youtu.be/ec4J7PwYPSA

Session 1: Reconsidering Internet Architecture

"Internet Architecture": *the definition* ?