# *A Simple Introduction To Computer Networking*

by Kalid Azad · 22 comments 

Most networking discussions are a jumble of acronyms. Forget the configuration details — what are the insights?

- **Networking is about communication**
- **Text is the simplest way to communicate**
- **Protocols are standards for reading and writing text**

Beneath the details, networking is an IM conversation. Here's what I wish someone told me when learning how computers communicate.

## *TCP: The Text Layer*

The Transmission Control Protocol (TCP) provides the handy illusion that we can "just" send text between two computers. TCP relies on lower levels (http://en.wikipedia.org/wiki/Internet_Protocol) and can send binary data, but ignore that for now:

- **TCP lets us Instant Message between computers**

We IM with Telnet, the 'notepad' of networking: telnet sends and receives plain text using TCP. It's a chat client peacefully free of ads and unsolicited buddy requests.

Let's talk to Google using telnet (http://support.microsoft.com/kb/279466) (or putty (http://www.chiark.greenend.org.uk/~sgtatham/putty/), a better utility):

```
telnet google.com 80
[connecting...]
Hello Mr. Google!
```

We connect to google.com on port 80 (the default for web requests) and send the message "Hello Mr. Google!". We press Enter a few times and await the reply:

```
<html>
...
<h1>Bad Request</h1>
Your client has issued a malformed or illegal request
...
</html>
```

Malformed? Illegal? *The mighty Google is not pleased*. It didn't understand us and sent HTML telling the same.

But, we had a conversation: text went in, and text came back. In other words:

## *Protocols: The Forms To Fill Out*

Unstructured chats are too carefree — how does the server know what we want to do? We need a *protocol* (standard way of communicating) if we're going to make sense.

We use protocols all the time

- Putting "to" and "from" addresses in special places on an envelope
- Filling out bank forms (special place for account number, deposit amount, etc.)
- Saying "Roger" or "10-4" to indicate a radio request was understood

Protocols make communication clear.

## *Case Study: The HTTP Protocol*

We see HTTP in every url: http://google.com/ (http://google.com/). What does it mean?

- Connect to server google.com (Using TCP, port 80 by default)
- Ask for the resource "/" (the default resource)
- Format the request using the Hypertext Transport Protocol

HTTP is the "form to fill out" when asking for the resource. Using the HTTP format, the above request looks like this:

```
GET / HTTP/1.0
```

Remember, *it's just text*! We're asking for a file, through an IM session, using the format: [Command] [Resource] [Protocol Name/Version].

This command is "IM'd" to the server (your browser adds extra info, a detail for another time). Google's server returns this response:

```
HTTP/1.0 200 OK
Cache-Control: private, max-age=0
Date: Sun, 15 Mar 2009 03:13:39 GMT
Expires: -1
Content-Type: text/html; charset=ISO-8859-1
Set-Cookie: PREF=ID=5cc6…
Server: gws
Connection: Close

<html>
(Google web page, search box, and cute logo)
</html>
```

Yowza. The bottom part is HTML for the browser to display. But why the junk up top?

Well, suppose we just got the raw HTML to display. But what about errors: if the server crashed, the file wasn't there, or google just didn't like us?

Some *metadata* (data about data) is useful. When we order a book from Amazon **we expect a packing slip** describing the order: the intended recipient, price, return information, etc. You don't want a naked book just thrown on your doorstep.

Protocols are similar: the recipient wants to know if everything was OK. Here we see infamous status codes like 404 (resource not found) or 200 (everything OK). These headers aren't the real data — they're the packing slip from the server.

## *Insights From Protocols*

Studying existing, popular systems is a great way to understand engineering decisions. Here are a few:

**Binary vs Plain Text**

Binary data (http://betterexplained.com/articles/a-little-diddy-about-binary-file-formats/) is more efficient than text, but more difficult to debug and generate (how many hex editors do you know to use?). Lower-level protocols, the backbone of the internet, use

binary data to maintain performance. Application-level protocols (HTTP and above) use text data for ease of interoperability. You don't have religious wars about endian issues with HTTP.

**Stateful vs. Stateless**

Some protocols are stateful, which means the server remembers the chat with the client. With SMTP, for example, the client opens a connection and issues commands one at a time (such as adding recipients to an email), and closes the connection. Stateful communication is useful in transactions that have many steps or conditions.

Stateless communication is simpler: you send the entire transaction as one request. Each "instant message" stands on its own and doesn't need the others. HTTP is stateless: you can request a webpage without introducing yourself to the server.

**Extensibility**

We can't think of everything beforehand. How do we extend old protocols for new users?

HTTP has a simple and effective "header" structure: a metadata preamble that looks like "Header:Value".

If you don't recognize the header sent (new client, old server) just ignore it. If you were expecting a header but don't see it (old client, new server), just use a default. It's like having an "Anything else to tell us?" section in a survey.

**Error Correction & Reliability**

It's the job of lower-level protocols like TCP to make sure data is transmitted reliably. But higher-level protocols (like HTTP) need to make sure it's the *right* data. How are errors handled and communicated? Can the client just retry or does the server need to reset state?

HTTP comes with its own set of error codes to handle a variety of situations.

**Availability**

The neat thing about networking is that works on one computer. Memcached is a great service to cache data. And guess what? It uses plain-old text commands (over TCP) to save and retrieve data.

You don't need complex COM objects or DLLs – you start a Memcached server, send text in, and get text out. It's language-neutral and easy to access because any decent OS supports networking. You can even telnet into Memcached to debug it.

Wireless routers are similar: they have a control panel available through HTTP. There's no "router configuration program" — you just connect to it with your browser. The router serves up webpages, and when you submit data it makes the necessary configuration changes.

Protocols like HTTP are so popular you can *assume* the user has a client.

**Layering Protocols**

Protocols can be layered. We might write a resume, which is part of a larger application, which is stuffed into an envelope. Each segment has its own format, blissfully unaware of the others. Your envelope doesn't care about the resume — it just wants the to: and from: addresses written correctly.

Many protocols rely on HTTP because it's so widely used (rather than starting from scratch, like Memcached, which needs efficiency). HTTP has well-understood methods to define resources (URLs) and commands (GET and POST), so why not use them?

Web services do just that. The SOAP protocol crams XML inside of HTTP commands. The REST protocol embraces HTTP and uses the existing verbs as much as possible.

## *Remember: It's All Made Up*

Networking involves *human conventions*. Because plain text is ubiquitous and easy to use, it is the basis for most protocols. And TCP is the simplest, most-supported way to exchange text.

**Remembering that everything is a plain text IM conversation** helps me wrap my head around the inevitable networking issues. And sometimes you need to jump into HTTP to understand compression (http://betterexplained.com/articles/how-to-optimize-your-site-with-gzip-compression/) and caching (http://betterexplained.com/articles/how-to-optimize-your-site-with-http-caching/).

Don't just memorize the details; see protocols as strategies to solve communication problems. Happy networking.

Posted in Guides (http://betterexplained.com/articles/category/guides/), Programming (http://betterexplained.com/articles/category/programming/)



## *Questions & Contributions*

Ask a Question (http://betterexplained.com/articles/a-simple-introduction-to-computer-networking/#tab-faq)

Contribute an Insight (http://betterexplained.com/articles/a-simple-introduction-to-computer-networking/#tab-aha)

## Ask Question

Name (optional)

Reference URL (optional)

Have feedback? Just enter it above. I'm making a curated set of questions and insights for the article. Thanks!

Continue Discussion 23 replies

Mar '09

T

### trose

Hey Kalid,
Good overview. Thanks for keeping it simple. Problem with simple is that all us cs geeks will nitpick. I hope that doesn't happen and that the comments stay high level like your article. A quick follow up on how address resolution works would be fun. In the parlay or your article when I IM the screen name 'penguinlips' how does that name get resolved to my window on my laptop. That might bring the whole networking thing to light for a whole level of folks.
take care,
stop by and visit soon
T.

Mar '09

P

### prateeksharma

Good to see an article on something different from maths. And I like your analogy of telnet to IM. Should make stuff easier to understand for most people.

I am not sure about the target audience but most novice people I have explained basics of n/w to are more interested in knowing:

a. What does IP Address mean?
b. What are Software Ports?
c. How does DNS Resolution work?
d. How does email work?

I understand that N/W is a very vast topic and cannot be summed in one blog post but you did a great job in introducing the basics. Its a good reference for newbies!

As always, great work! keep it up! 

Mar '09

A

### Anonymous_User

[...] A Simple Introduction To Computer Networking | BetterExplained Most networking discussions are a jumble of acronyms. Forget the configuration details — what are the insights? [...]

Mar '09



### Anonymous_User

[...] 前几天，BetterExplained上有一篇文章，很通俗地解释了这个模型。我读后有一种恍然大悟的感觉，第一次感到自己理解了互联网的总体架构。 [...]

Mar '09

### kalid Founder

@Mr. Rose: Yeah, the nitpicking is definitely the issue when talking about computer topics -- there's always a deeper level to talk about. I think the big thing for me is realizing what level of abstraction to stay at (just thinking about text) and then moving on once that layer is understood. Address resolution will definitely be on the list -- also, planning on coming back to town in mid-April :).

@Prateek: Thanks for the support! Yep, it's fun to vary up the topics sometimes, gives me an excuse to explore new things. I think IP address, DNS resolution and the like would be great topics to cover :).

Mar '09



### Anonymous_User

[...] A simple explanation of networking : http://betterexplained.com/articles/a-simple-introduction-to-computer-networking/ [...]

Apr '09



### Anonymous_User

[...] 前几天，BetterExplained上有一篇文章，很通俗地解释了这个模型。我读后有一种恍然大悟的感觉，第一次感到自己理解了互联网的总体架构。 [...]

Apr '09

### bhagavanjalli

As usual, one more fabulous article! Thanks Khalid. Yes, I wish you were my college professor in the first year at the engineering school.

Apr '09

### kalid Founder

@Bhagavan: Thanks!

Apr '09

### scientificchick

Thanks for an excellent overview. Very helpful for me since I'm generally clueless about computer stuff but willing to learn.

Apr '09

### kalid Founder

@Scientific Chick: You're welcome, glad it was helpful -- and being willing to learn is more important than knowing the answer :).

Apr '09

### kirantikare

Hi Khalid,

Nice post, I like your posts.

Even though I know these things, but learning it in some other point of view is always good.

Thanks for writing such things by simplifying them.

Regards,
Kiran Tikare

Jun '09

**jjumba**

hello i would like to be student of networking and lear how to use servers

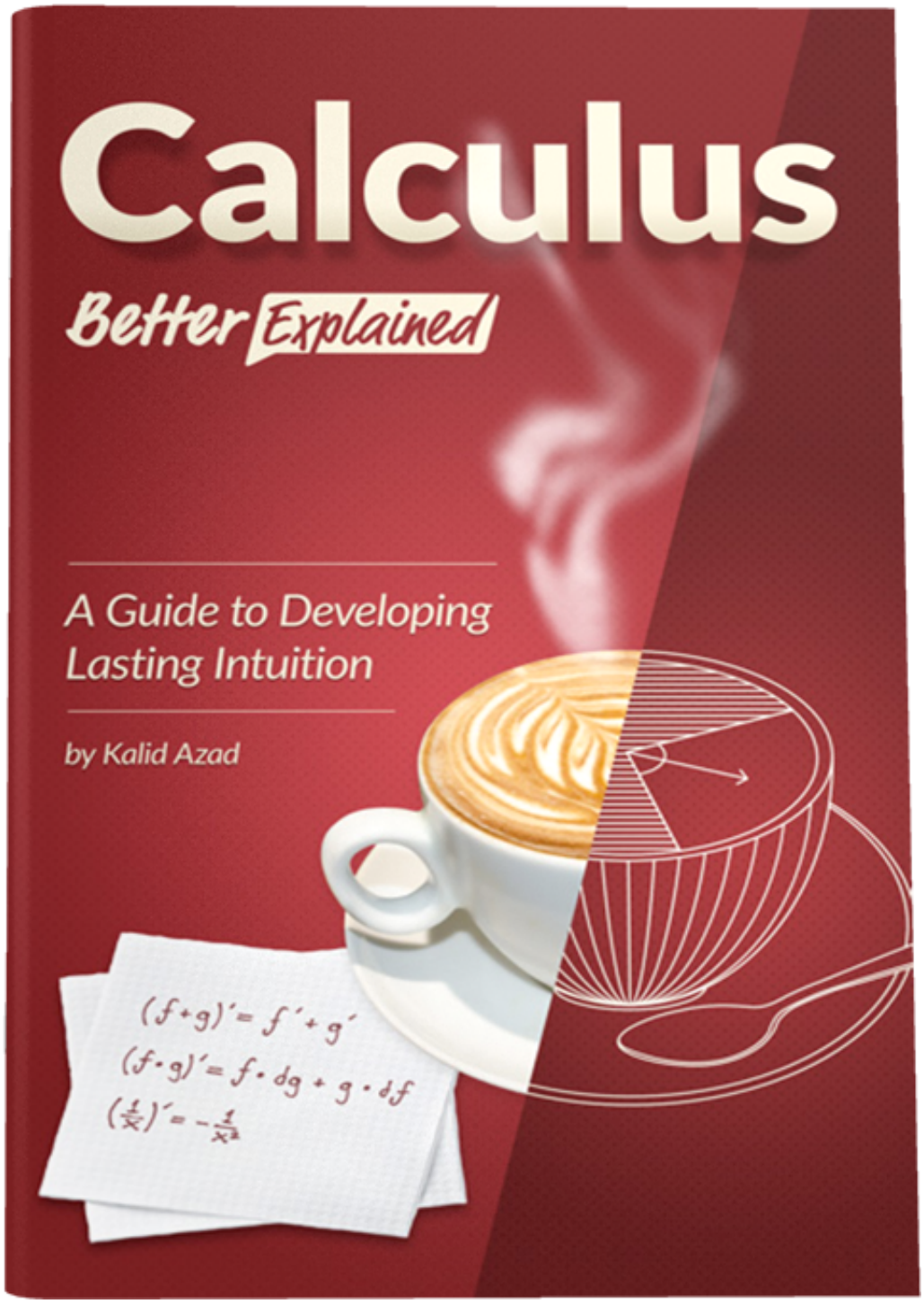
Feb '10

***In This Series***

***About The Site***

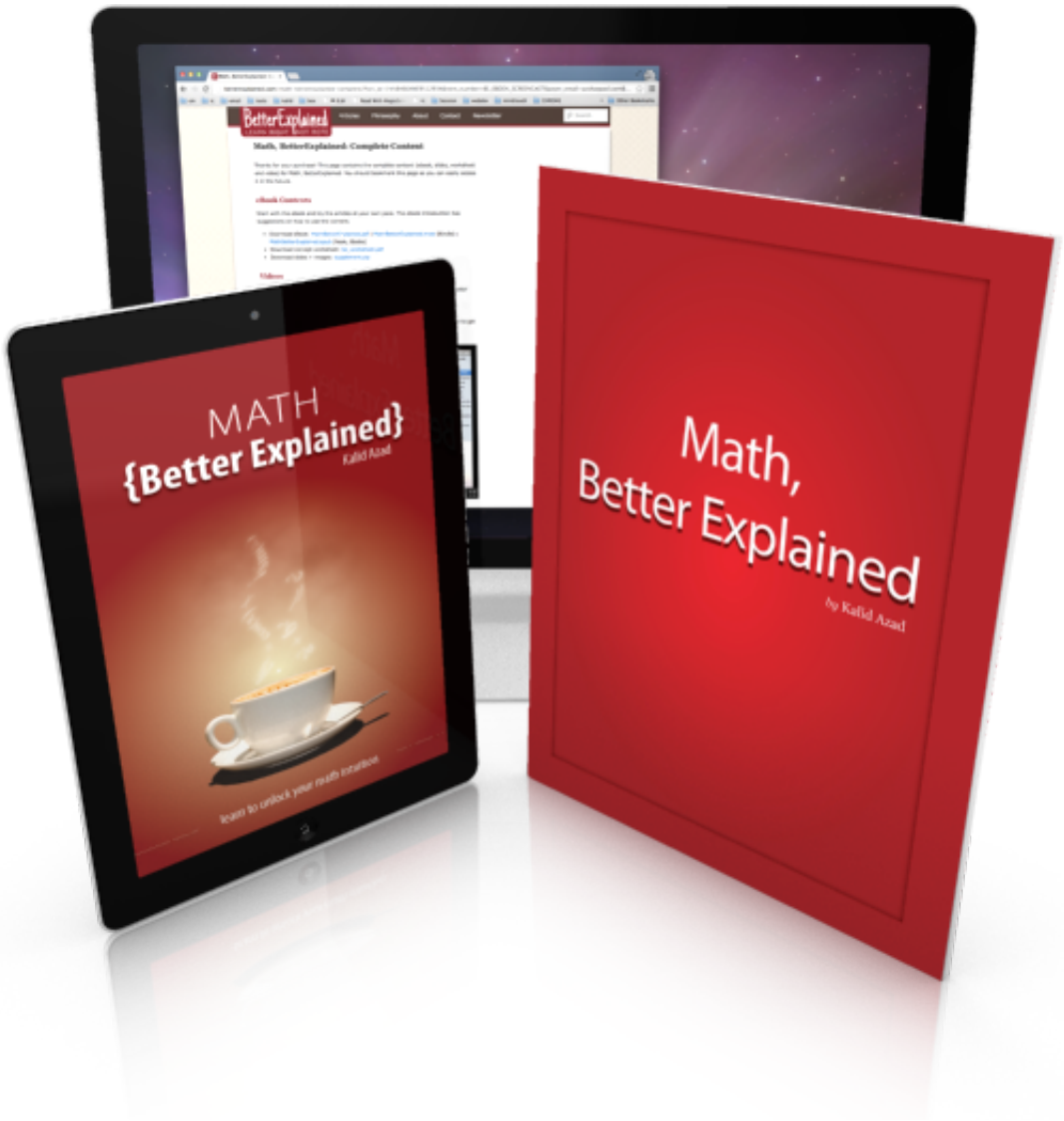
BetterExplained helps 400k monthly readers with clear, insightful lessons.

***Calculus Course***




(http://betterexplained.com/calculus)Calculus insights within minutes. Free to read online (http://betterexplained.com/calculus).

***Math, Better Explained***




(http://betterexplained.com/ebook/math) A dozen math essentials (http://betterexplained.com/ebook/math). Amazon bestseller.

---

"If you can't explain it simply, you don't understand it well enough." —Einstein (more (http://betterexplained.com/philosophy/))