

Define an application protocol

<https://github.com/heig-vd-dai-course>

[Web](#) • [PDF](#)

L. Delafontaine and H. Louis, with the help of GitHub Copilot

This work is licensed under the [CC BY-SA 4.0](#) license.

Objectives

- Refresh on networking
- Learn where to find information about application protocols
- Understand application protocol
- Define application protocols
- How to use an application protocol
- **While quite short, one of the most important chapters of the course!**

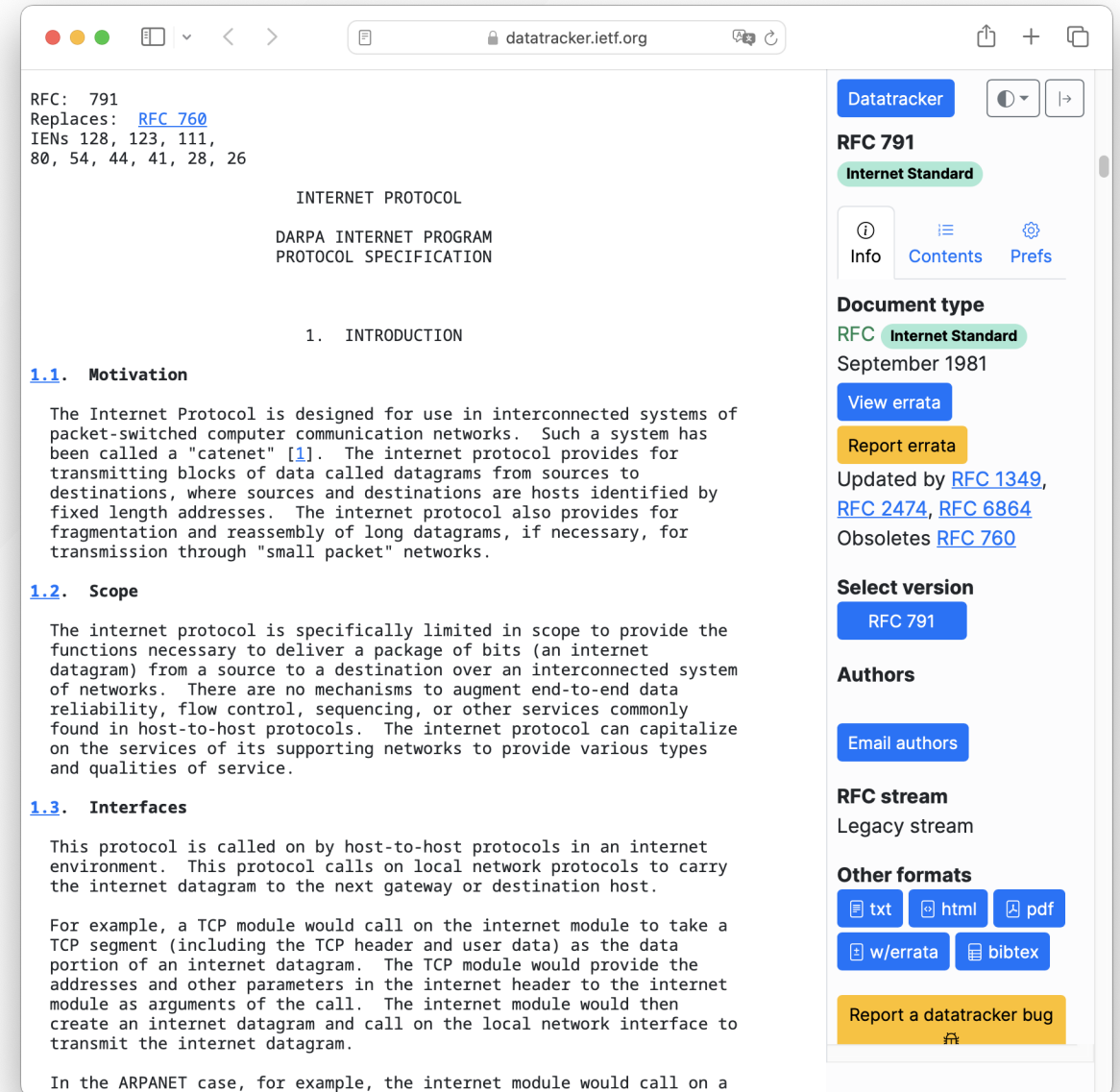


A quick reminder about networking

More details for this section in the [course material](#). You can find other resources and alternatives as well.

The Internet Protocol (IP)

- Each computer has a unique IP address
- IPv4 addresses are limited; NAT routers share IP addresses
- IPv6 fixes this issue
- IP addresses are used to route packets



The screenshot shows a web browser displaying the Datatracker page for RFC 791, titled "INTERNET PROTOCOL DARPA INTERNET PROGRAM PROTOCOL SPECIFICATION". The page includes a sidebar with navigation options like "Info", "Contents", and "Prefs", and a main content area with sections for "1. INTRODUCTION", "1.1. Motivation", "1.2. Scope", and "1.3. Interfaces". The sidebar also features a "Document type" section, "View errata", "Report errata", "Updated by" links, "Obsoletes" link, "Select version" dropdown, "Email authors" button, "RFC stream" information, "Other formats" buttons, and a "Report a datatracker bug" button.

RFC: 791
Replaces: [RFC 760](#)
IENs 128, 123, 111, 80, 54, 44, 41, 28, 26

INTERNET PROTOCOL
DARPA INTERNET PROGRAM
PROTOCOL SPECIFICATION

1. INTRODUCTION

1.1. Motivation

The Internet Protocol is designed for use in interconnected systems of packet-switched computer communication networks. Such a system has been called a "catenet" [1]. The internet protocol provides for transmitting blocks of data called datagrams from sources to destinations, where sources and destinations are hosts identified by fixed length addresses. The internet protocol also provides for fragmentation and reassembly of long datagrams, if necessary, for transmission through "small packet" networks.

1.2. Scope

The internet protocol is specifically limited in scope to provide the functions necessary to deliver a package of bits (an internet datagram) from a source to a destination over an interconnected system of networks. There are no mechanisms to augment end-to-end data reliability, flow control, sequencing, or other services commonly found in host-to-host protocols. The internet protocol can capitalize on the services of its supporting networks to provide various types and qualities of service.

1.3. Interfaces

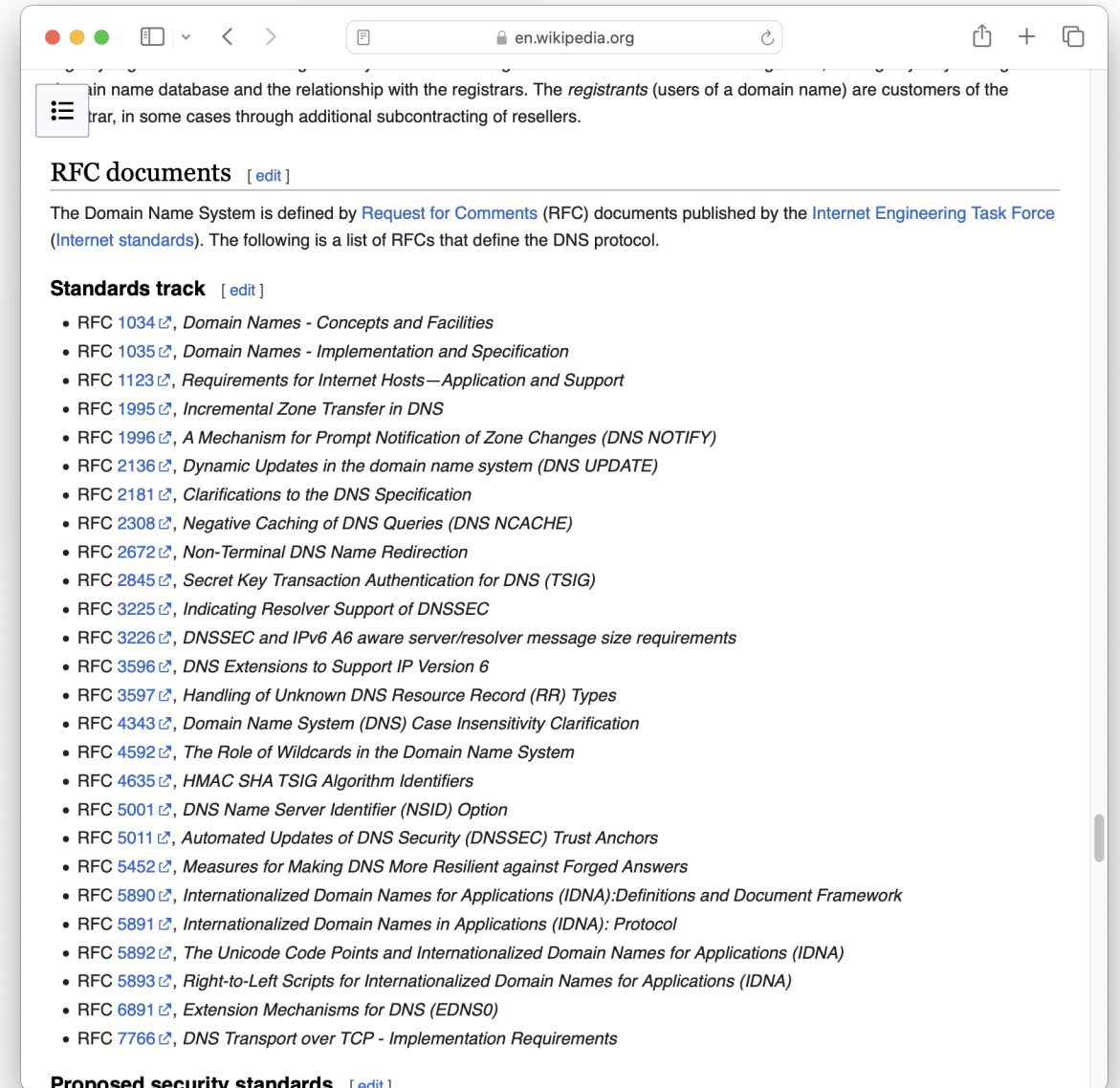
This protocol is called on by host-to-host protocols in an internet environment. This protocol calls on local network protocols to carry the internet datagram to the next gateway or destination host.

For example, a TCP module would call on the internet module to take a TCP segment (including the TCP header and user data) as the data portion of an internet datagram. The TCP module would provide the addresses and other parameters in the internet header to the internet module as arguments of the call. The internet module would then create an internet datagram and call on the local network interface to transmit the internet datagram.

In the ARPANET case, for example, the internet module would call on a

The Domain Name System (DNS)

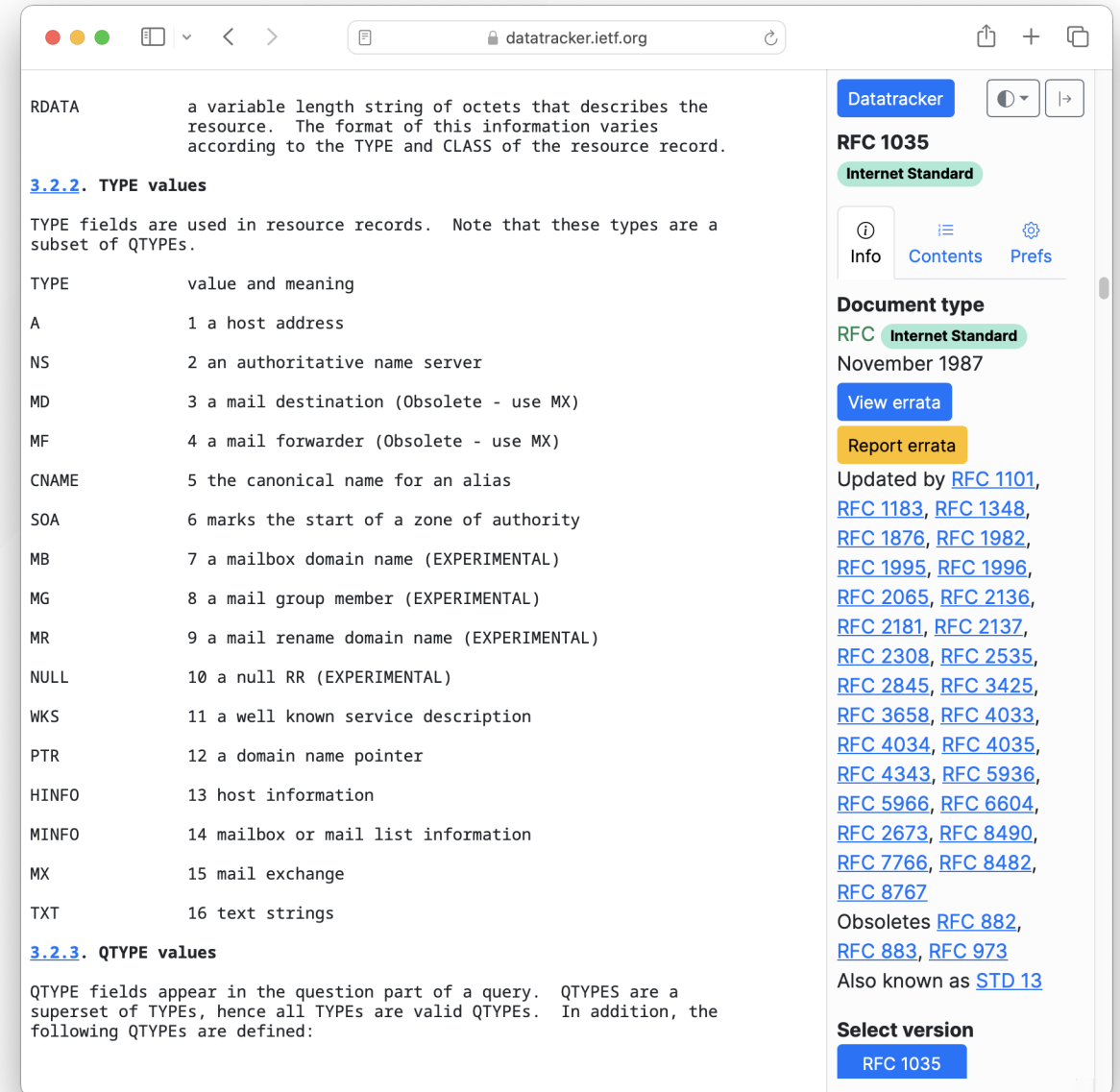
- DNS maps domain names to IP addresses
- Example: `heig-vd.ch` → `193.134.223.20`
- `dig` and `nslookup` are useful tools to query DNS servers



Common DNS records

Records map a domain name to an IP address.

- **NS** : Name Server
- **CNAME** : Alias
- **A** : IPv4 address
- **AAAA** : IPv6 address



The screenshot shows the Datatracker website for RFC 1035. The main content area is titled "RDATA" and describes a variable length string of octets. Below this, there is a section "3.2.2. TYPE values" which lists various DNS record types and their meanings. The list includes:

TYPE	value and meaning
A	1 a host address
NS	2 an authoritative name server
MD	3 a mail destination (Obsolete - use MX)
MF	4 a mail forwarder (Obsolete - use MX)
CNAME	5 the canonical name for an alias
SOA	6 marks the start of a zone of authority
MB	7 a mailbox domain name (EXPERIMENTAL)
MG	8 a mail group member (EXPERIMENTAL)
MR	9 a mail rename domain name (EXPERIMENTAL)
NULL	10 a null RR (EXPERIMENTAL)
WKS	11 a well known service description
PTR	12 a domain name pointer
HINFO	13 host information
MINFO	14 mailbox or mail list information
MX	15 mail exchange
TXT	16 text strings

Below the table, there is a section "3.2.3. QTYPE values" which explains that QTYPE fields appear in the question part of a query and are a superset of TYPEs.

The right sidebar contains navigation and document information, including "Datatracker", "RFC 1035", "Internet Standard", "Info", "Contents", "Prefs", "Document type", "RFC", "Internet Standard", "November 1987", "View errata", "Report errata", "Updated by RFC 1101, RFC 1183, RFC 1348, RFC 1876, RFC 1982, RFC 1995, RFC 1996, RFC 2065, RFC 2136, RFC 2181, RFC 2137, RFC 2308, RFC 2535, RFC 2845, RFC 3425, RFC 3658, RFC 4033, RFC 4034, RFC 4035, RFC 4343, RFC 5936, RFC 5966, RFC 6604, RFC 2673, RFC 8490, RFC 7766, RFC 8482, RFC 8767", "Obsoletes RFC 882, RFC 883, RFC 973", "Also known as STD 13", and "Select version RFC 1035".

Reserved ports

- Ports identify processes or services
- Analogy: an IP address is like a street address, a port is like an apartment number
- Ports are 16-bit unsigned numbers, maximum 65535
- Reserved ports: 0-1023
- Other ports: wild west

The screenshot shows the Wikipedia page for 'Reserved ports'. It includes a table legend for TCP and UDP protocol table cells, a section for well-known ports with a disclaimer, and a detailed table of well-known ports.

Table legend [edit]

Legend of TCP and UDP protocol table cells for port numbers

Cell	Description
Yes	Described protocol <i>is</i> assigned by IANA for this port, and <i>is</i> : standardized, specified, or widely used for such.
Unofficial	Described protocol <i>is not</i> assigned by IANA for this port, but <i>is</i> : standardized, specified, or widely used for such.
Assigned	Described protocol <i>is</i> assigned by IANA for this port, ^[2] but <i>is not</i> : standardized, specified, or widely used for such.
No	Described protocol <i>is not</i> : assigned by IANA for this port, standardized, specified, or widely used for such.
Reserved	Port is reserved by IANA, ^[2] generally to prevent collision having its previous use removed. ^{[3][4]} The port number may be available for assignment upon request to IANA. ^[3]

Well-known ports [edit]

This is a dynamic list and may never be able to satisfy particular standards for completeness. You can help by adding missing items with reliable sources.

The port numbers in the range from 0 to 1023 (0 to $2^{10} - 1$) are the *well-known ports* or *system ports*.^[3] They are used by system processes that provide widely used types of network services. On **Unix-like** operating systems, a process must execute with **superuser** privileges to be able to bind a **network socket** to an **IP address** using one of the well-known ports.^[5]

Well-known ports [hide]

Port	TCP	UDP	SCTP	DCCP	Description
0	Reserved				In programming APIs (not in communication between hosts), requests a system-allocated (dynamic) port ^[6]
1	Yes	Assigned			TCP Port Service Multiplexer (TCPMUX). Historic. Both TCP and UDP have been assigned to TCPMUX by IANA, ^[2] but by design only TCP is specified. ^[7]
2	Assigned				compressnet (Management Utility) ^[3]
3	Assigned				compressnet (Compression Process) ^[3]
5	Assigned				Remote Job Entry ^[8] was historically using socket 5 in its <i>old socket form</i> , while MIB PIM has identified it as TCP/5 ^[9] and IANA has assigned both TCP and UDP 5 to it.
7	Yes				Echo Protocol ^{[10][11]}

What is an application protocol

More details for this section in the [course material](#). You can find other resources and alternatives as well.

What is an application protocol

- Defines application communication
- RFCs on the IETF website:
 - Relies on transport and network protocols
 - Multiple revisions exist

The screenshot shows the IETF Datatracker page for RFC 5321, titled "Simple Mail Transfer Protocol". The page is from the Network Working Group, authored by J. Klensin in October 2008. It is categorized as a "Draft Standard" and is part of the "Standards Track".

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

This document is a specification of the basic protocol for Internet electronic mail transport. It consolidates, updates, and clarifies several previous documents, making all or parts of most of them obsolete. It covers the SMTP extension mechanisms and best practices for the contemporary Internet, but does not provide details about particular extensions. Although SMTP was designed as a mail transport and delivery protocol, this specification also contains information that is important to its use as a "mail submission" protocol for "split-UA" (User Agent) mail reading systems and mobile environments.

Document type

RFC **Draft Standard**
October 2008
[View errata](#)
[Report errata](#)
Updated by [RFC 7504](#)
Obsoletes [RFC 2821](#)
Updates [RFC 1123](#)
Was [draft-klensin-rfc2821bis](#) (individual in app area)

Select version

00	01	02	03	04
05	06	07	08	09
10	11	RFC 5321		

Compare versions

...nsin-rfc2821bis-10 ▾
RFC 5321 ▾
[Side-by-side](#) [Inline](#)

Author
[Dr. John C. Klensin](#) ✉
[Email author](#)

How is structured an application protocol

More details for this section in the [course material](#). You can find other resources and alternatives as well.

How is structured an application protocol

- Defined by a set of rules to follow in a RFC
- Rules define:
 - Transport protocol
 - Messages order
 - Examples and errors

4. The SMTP Specifications

4.1. SMTP Commands

4.1.1. Command Semantics and Syntax

The SMTP commands define the mail transfer or the mail system function requested by the user. SMTP commands are character strings terminated by <CRLF>. The commands themselves are alphabetic characters terminated by <SP> if parameters follow and <CRLF> otherwise. (In the interest of improved interoperability, SMTP receivers SHOULD tolerate trailing white space before the terminating <CRLF>.) The syntax of the local part of a mailbox MUST conform to receiver site conventions and the syntax specified in [Section 4.1.2](#). The SMTP commands are discussed below. The SMTP replies are discussed in [Section 4.2](#).

A mail transaction involves several data objects that are communicated as arguments to different commands. The reverse-path is the argument of the MAIL command, the forward-path is the argument of the RCPT command, and the mail data is the argument of the DATA command. These arguments or data objects must be transmitted and held, pending the confirmation communicated by the end of mail data indication that finalizes the transaction. The model for this is that distinct buffers are provided to hold the types of data objects; that is, there is a reverse-path buffer, a forward-path buffer, and a mail data buffer. Specific commands cause information to be appended to a specific buffer, or cause one or more buffers to be cleared.

Several commands (RSET, DATA, QUIT) are specified as not permitting parameters. In the absence of specific extensions offered by the server and accepted by the client, clients MUST NOT send such parameters and servers SHOULD reject commands containing them as having invalid syntax.

4.1.1.1. Extended HELLO (EHLO) or HELLO (HELO)

These commands are used to identify the SMTP client to the SMTP server. The argument clause contains the fully-qualified domain name of the SMTP client, if one is available. In situations in which the SMTP client system does not have a meaningful domain name (e.g., when its address is dynamically allocated and no reverse mapping record is

Klensin Standards Track [Page 32]

[RFC 5321](#) SMTP October 2008

Aliases
3.9.1. Alias
3.9.2. List
4. The SMTP Specifications
4.1. SMTP Commands
4.1.1. Command Semantics and Syntax
4.1.1.1. Extended HELLO (EHLO) or HELLO (HELO)
4.1.1.2. MAIL (MAIL)
4.1.1.3. RECIPIENT (RCPT)
4.1.1.4. DATA (DATA)
4.1.1.5. RESET (RSET)
4.1.1.6. VERIFY (VRFY)
4.1.1.7. EXPAND (EXPN)
4.1.1.8. HELP (HELP)
4.1.1.9. NOOP (NOOP)
4.1.1.10. QUIT (QUIT)
4.1.1.11. Mail-Parameter and Rcpt-Parameter Error Responses

How to define an application protocol

More details for this section in the [course material](#). You can find other resources and alternatives as well.

How to define an application protocol

- Lot of work and thinking
- A protocol is never perfect
- The more you take your time to think and design it, the less you will have to change it later

Appendix D. Scenarios

This section presents complete scenarios of several types of SMTP sessions. In the examples, "C:" indicates what is said by the SMTP client, and "S:" indicates what is said by the SMTP server.

Klensin Standards Track [Page 87]

[RFC 5321](#) SMTP October 2008

D.1. A Typical SMTP Transaction Scenario

This SMTP example shows mail sent by Smith at host bar.com, and to Jones, Green, and Brown at host foo.com. Here we assume that host bar.com contacts host foo.com directly. The mail is accepted for Jones and Brown. Green does not have a mailbox at host foo.com.

```
S: 220 foo.com Simple Mail Transfer Service Ready
C: EHLO bar.com
S: 250-foo.com greets bar.com
S: 250-8BITMIME
S: 250-SIZE
S: 250-DSN
S: 250 HELP
C: MAIL FROM:<Smith@bar.com>
S: 250 OK
C: RCPT TO:<Jones@foo.com>
S: 250 OK
C: RCPT TO:<Green@foo.com>
S: 550 No such user here
C: RCPT TO:<Brown@foo.com>
S: 250 OK
C: DATA
S: 354 Start mail input; end with <CRLF>.<CRLF>
```

Forwarding
7.8. Resistance to Attacks
7.9. Scope of Operation of SMTP Servers
8. IANA Considerations
9. Acknowledgments
10. References
10.1. Normative References
10.2. Informative References
A. TCP Transport Service
B. Generating SMTP Commands from RFC 822 Header Fields
C. Source Routes
D. Scenarios
D.1. A Typical SMTP Transaction Scenario
D.2. Aborted SMTP Transaction Scenario
D.3. Relayed Mail Scenario
D.4. Verifying and Sending Scenario
E. Other Gateway Issues
F. Deprecated Features of RFC 821
F.1. TURN
F.2. Source Routing
F.3. HELO
F.4. #-literals
F.5. Dates and Years
F.6. Sending versus

- Simplified structure for this course:

- Section 1 - Overview
- Section 2 - Transport protocol
- Section 3 - Messages
- Section 4 - Examples

The screenshot shows a web browser window with the URL `datatracker.ietf.org`. The page title is "Appendix D. Scenarios". The main content area contains the following text:

This section presents complete scenarios of several types of SMTP sessions. In the examples, "C:" indicates what is said by the SMTP client, and "S:" indicates what is said by the SMTP server.

Below the text, there is a metadata line: "Klensin Standards Track [Page 87]".

The document identifier is "RFC 5321" and the date is "October 2008".

The section title is "D.1. A Typical SMTP Transaction Scenario".

The main text of the scenario is: "This SMTP example shows mail sent by Smith at host bar.com, and to Jones, Green, and Brown at host foo.com. Here we assume that host bar.com contacts host foo.com directly. The mail is accepted for Jones and Brown. Green does not have a mailbox at host foo.com."

The SMTP transcript is as follows:

```
S: 220 foo.com Simple Mail Transfer Service Ready
C: EHLO bar.com
S: 250-foo.com greets bar.com
S: 250-8BITMIME
S: 250-SIZE
S: 250-DSN
S: 250 HELP
C: MAIL FROM:<Smith@bar.com>
S: 250 OK
C: RCPT TO:<Jones@foo.com>
S: 250 OK
C: RCPT TO:<Green@foo.com>
S: 550 No such user here
C: RCPT TO:<Brown@foo.com>
S: 250 OK
C: DATA
S: 354 Start mail input; end with <CRLF>.<CRLF>
```

On the right side of the browser, there is a table of contents for the document, listing sections from "Forwarding" to "F.6. Sending versus".

Section 1 - Overview

This section defines the purpose of the protocol:

- What is the goal of the protocol?
- What is the problem that it tries to solve?
- What the application protocol is used for?

Section 2 - Transport protocol

This section defines the transport protocol used by the application protocol:

- What protocol(s) is/are involved? On which port(s)?
- How are messages/actions encoded?
- How are messages/actions delimited?
- How are messages/actions treated (text or binary)?
- Who initiates/closes the communication?
- What happens on an unknown message/action/exception?

Section 3 - Messages

This section defines the messages that can be exchanged between the client and the server.

- What are the messages/actions?
- What are the parameters?
- What are the return values?
- What are the exceptions?

Try to describe these for a given context, not from each point of view. It makes it way easier to understand and to implement.

Section 4 - Examples

This section defines examples of messages that can be exchanged between the client and the server and the exchange order:

- What are the examples of messages/actions?
- What are the examples of exceptions?

It is important to define these examples to illustrate the protocol and to help the reader to understand the protocol using sequence or state diagrams.

Example - The SMS protocol

More details for this section in the [course material](#). You can find other resources and alternatives as well.

Example - The SMS protocol

“ You are working for a startup that wants to create a new communication app.

The app is simple: it allows users (with unique usernames) to send small text messages (maximum 100 characters) to each other. The server is in charge of sending the messages to the recipients.

You are asked to define the application protocol that will be used by the clients and the server. ”

Questions

Do you have any questions?

Practical content

What will you do?

- Define two custom application protocols:
 - *"Guess the number"* game
 - *"Temperature monitoring"* application

These application protocols will be used in the next chapters to implement them!



Find the practical content

You can find the practical content for this chapter on [GitHub](#).



Finished? Was it easy? Was it hard?

Can you let us know what was easy and what was difficult for you during this chapter?

This will help us to improve the course and adapt the content to your needs. If we notice some difficulties, we will come back to you to help you.

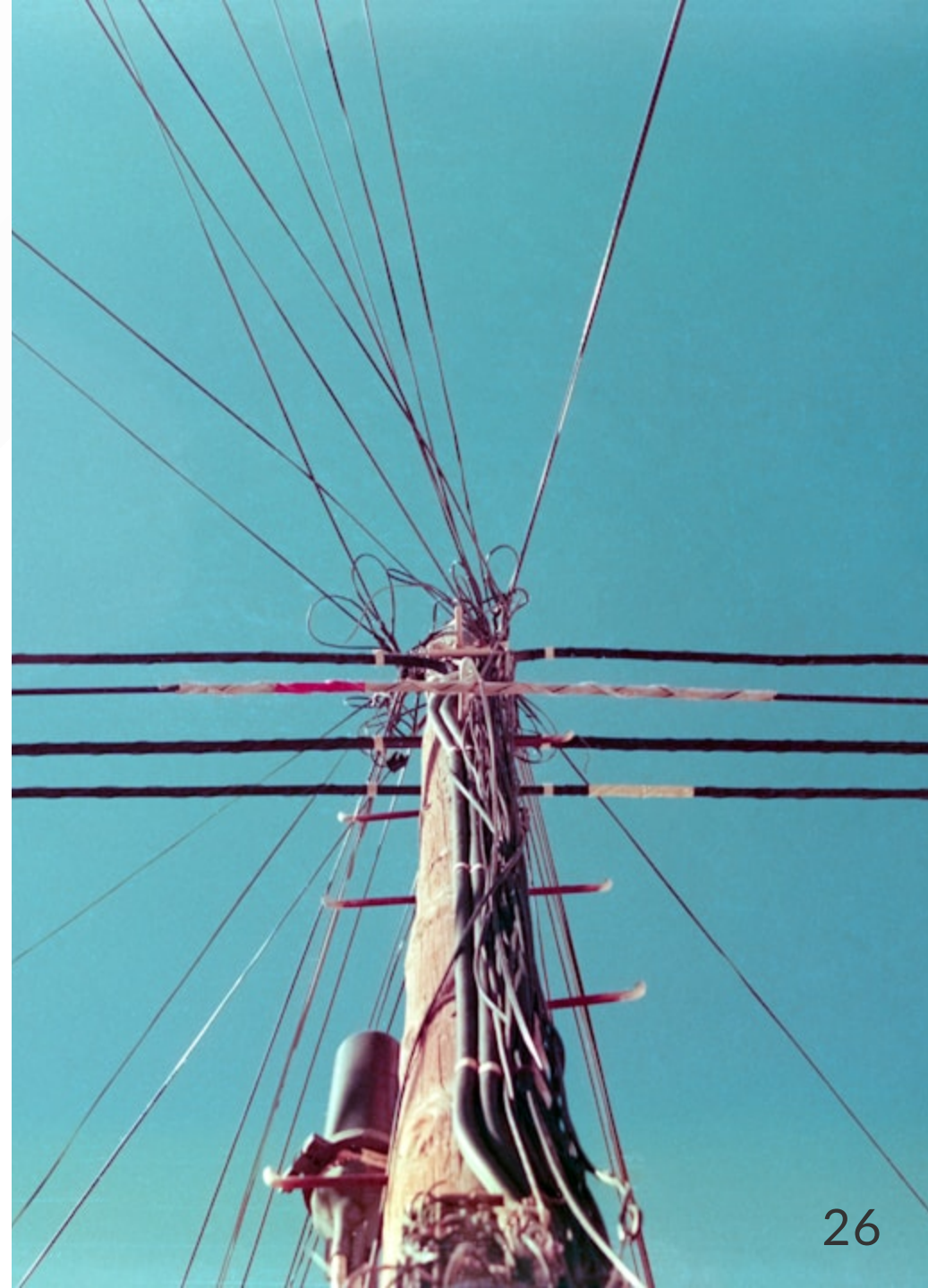
 [GitHub Discussions](#)

You can use reactions to express your opinion on a comment!

What will you do next?

In the next chapter, you will learn the following topics:

- Java TCP programming
 - How to create a TCP server
 - How to create a TCP client
 - Implement the "*Guess the number*" game using TCP (optional)



Sources

- Main illustration by [Iñaki del Olmo](#) on [Unsplash](#)
- Illustration by [Aline de Nadai](#) on [Unsplash](#)
- Illustration by [Henry Be](#) on [Unsplash](#)
- Illustration by [CHUTTERSNAAP](#) on [Unsplash](#)