COMP312-09A Communications and Systems Software

Lecture 2 – Internet Standards

Matthew Luckie
mluckie@cs.waikato.ac.nz

THE UNIVERSITY OF WAIKATO

Protocols and Standards

- Protocols
  - Define the format and the order of messages exchanged between two or more communicating entities, as well as the actions taken on the transmission and/or receipt of a message
  - The language computers use to talk over a network.
- Technical standards
  - Documents that define the characteristics (for example dimensions, safety aspects, performance requirements) of a product, process, or service according to the technical state of the art
  - The rules everybody agrees to.

Why do we care about protocols and standards?

- Interoperability
  - The ability of software and hardware on multiple machines from multiple vendors to communicate
    - Source: FOLDOC definition
  - In practice this means software is of higher quality
- Avoid being locked to a particular vendor

Protocols and Standards

- ITU: International Telecommunications Union
- GSM: Global System for Mobile communications
- Ethernet : IEEE 802: Local Area Networks
  - 802.3 – Ethernet
  - 802.11 – Wireless Ethernet
- Internet Standards : IETF Request for Comments (RFC)
  - RFC 791 – Internet Protocol (IPv4) : 1981
  - RFC 792 – ICMP : 1981
  - RFC 793 – TCP : 1981
- Web standards: W3C
  - HTML 4.01

Standards

- ANSI
  - C
  - C++
- Standards New Zealand
  - www.standards.co.nz
  - AS/NZS 2063:2008 Bicycle Helmets
  - NZS 2604:1999 Timber Framed Buildings
- The nice thing about standards is that there are so many of them to choose from
  - Andrew S. Tanenbaum

Not all network protocols have standards

- Bittorrent and most P2P protocols
- Most instant messengers

COMP202

- Ignored the fact standards existed
  - Focused on the fact various technologies existed
  - the problems they solved
  - how they worked, etc.
- Implementation of basic HTTP server by example, rather than standard
  - Saved heart-ache caused by prescribing a 176 page document to students for reading and partial implementation for a 5% lab task.

COMP312

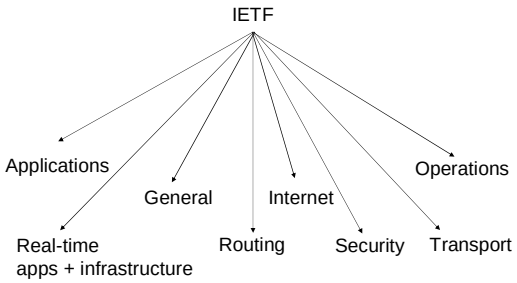
- Most focus will be on the Request for Comments (RFC) documents series produced by the Internet Engineering Task Force (IETF)

What is the IETF?

The Internet Engineering Task Force is a loosely self-organized group of people who contribute to the engineering and evolution of Internet technologies. It is the principal body engaged in the development of new Internet standard specifications. The IETF is unusual in that it exists as a collection of happenings, but is not a corporation and has no board of directors, no members, and no dues.

- RFC 3160: The Tao of IETF

The IETF: divided into areas



Each area has typically two area-directors. Directors form Internet Engineering Steering Group (IESG)

IESG

- Internet Engineering Steering Group
- Formation of all area directors (representing experts from wide-ranging expertise areas)
- RFC standards documents ultimately require approval from all IESG members.
  - Standards are therefore reviewed from multiple perspectives

The Internet Standards Process

In outline, the process of creating an Internet Standard is straightforward: a specification undergoes a period of development and several iterations of review by the Internet community and revision based upon experience, is adopted as a Standard by the appropriate body (see below), and is published. In practice, the process is more complicated ….

- RFC 2026 : The Internet Standards Process (1996)

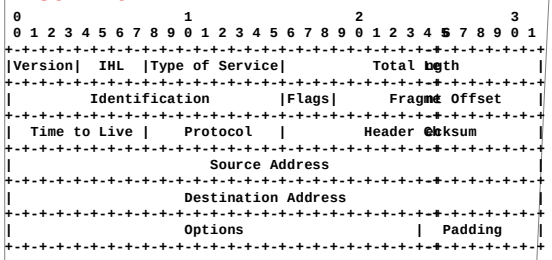
The Internet Standards Process

- Author(s) write an Internet Draft (a document formatted like an RFC)
  - Find, or create, a working group in one of the 8 available areas interested in pursuing the work towards standardisation
  - Iteratively refine the document in response to feedback
  - Obtain working-group consensus, running code from multiple implementers
- Forward to IESG for approval as RFC
- IESG consults, and approves if it determines the standard to be acceptable
- RFC assigned a number.

RFC documents

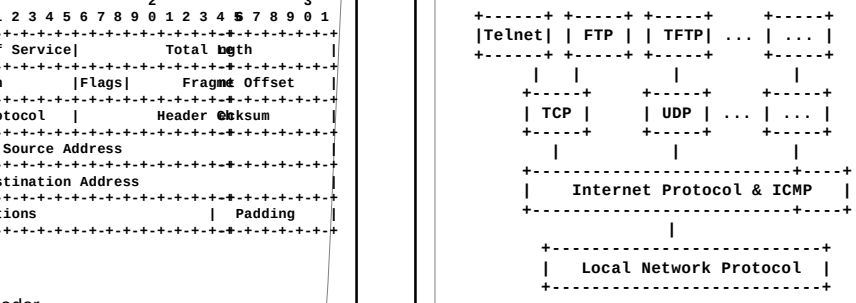
- Series began 7 April 1969.
- Plain text documents; ascii art as necessary
- Specify requirements, typically with upper-case words such as:
  - MUST (or REQUIRED, SHALL)
  - MUST NOT (SHALL NOT)
  - SHOULD (RECOMMENDED)
  - SHOULD NOT (NOT RECOMMENDED)
  - MAY (OPTIONAL)
- RFC 2119: Key words for use in RFCs to Indicate Requirement Levels

Ascii Art



RFC 791, IPv4 Header

More Ascii Art



RFC 791, Protocol Relationships

Typical structure of an RFC document

- Title
- Status of the memo
- Copyright notice
- Abstract
- Table of Contents
- Introduction
- <Protocol Definition>
- Security Considerations
- References
- Authors' Addresses
- Full Copyright Statement

Overview of RFC 2229: dictionary server

```
Network Working Group                                          R. Faith
Request for Comments: 2229                  U. North Carolina, Chapel Hill
Category: Informational                                        B. Martin
                                                  Miranda Productions
                                                          October 1997

                    A Dictionary Server Protocol


Status of this Memo

   This memo provides information for the Internet community.  It does
   not specify an Internet standard of any kind.  Distribution of this
   memo is unlimited.

Copyright Notice

   Copyright (C) The Internet Society (1997).  All Rights Reserved.
```

- http://www.rfc-editor.org/rfc/rfc2229.txt

Overview of RFC 2229: dictionary server

Abstract

The Dictionary Server Protocol (DICT) is a TCP transaction based query/response protocol that allows a client to access dictionary definitions from a set of natural language dictionary databases.

Overview of RFC 2229: dictionary server

1. Introduction

For many years, the Internet community has relied on the "webster" protocol for access to natural language definitions. The webster protocol supports access to a single dictionary and (optionally) to a single thesaurus. In recent years, the number of publicly available webster servers on the Internet has dramatically decreased.

...

The webster protocol is not suitable for providing access to a large number of separate dictionary databases, and extensions to the current webster protocol were not felt to be a clean solution to the dictionary database problem.

Shorter: webster was good for its time, but we need something better.

Overview of RFC 2229: dictionary server

1. Introduction

(continued)

The DICT protocol is designed to provide access to multiple databases. Word definitions can be requested, the word index can be searched (using an easily extended set of algorithms), information about the server can be provided (e.g., which index search strategies are supported, or which databases are available), and information about a database can be provided (e.g., copyright, citation, or distribution information). Further, the DICT protocol has hooks that can be used to restrict access to some or all of the databases.

Overview of RFC 2229: dictionary server

2. Protocol Overview

2.1. Link Level

The DICT protocol assumes a reliable data stream such as provided by TCP. When TCP is used, a DICT server listens on port 2628.

This server is only an interface between programs and the dictionary databases. It does not perform any user interaction or presentation-level functions.

Shorter: uses TCP port 2628

Overview of RFC 2229: dictionary server

2.2. Lexical Tokens

Commands and replies are composed of characters from the UCS character set [ISO10646] using the UTF-8 [RFC2044] encoding. More specifically, using the grammar conventions from [RFC822]:

Document is referring to Section 2 of RFC 822: Augmented Backus-Naur Form (BNF)
http://www.rfc-editor.org/rfc/rfc822.txt

Overview of RFC 2229

```
                                          ; (  Octal, Decimal.)
CHAR        =  <any UTF-8 character (1 to 6 octets)>
CTL         =  <any ASCII control           ; (  000,  0.- 31.)
               character and DEL>           ; (  177,  127.)
CR          =  <ASCII CR, carriage return>  ; (   15,   13.)
LF          =  <ASCII LF, linefeed>         ; (   12,   10.)
SPACE       =  <ASCII SP, space>            ; (   40,   32.)
HTAB        =  <ASCII HT, horizontal-tab>   ; (   11,    9.)
<">         =  <ASCII quote mark>           ; (   42,   34.)
<'>         =  <ASCII single quote mark>    ; (   47,   39.)
CRLF        =  CR LF
WS          =  1*(SPACE / HTAB)

dqstring    =  <"> *(dqtext/quoted-pair) <">
dqtext      =  <any CHAR except <">, "\", and CTLs>
sqstring    =  <'> *(dqtext/quoted-pair) <'>
sqtext      =  <any CHAR except <'>, "\", and CTLs>
quoted-pair =  "\" CHAR
```

ABNF Repetition

- The character "*" preceding an element indicates repetition. The full form is:
  - **<l>*<m>element**
- indicating at least <l> and at most <m> occurrences of element. Default values are 0 and infinity so that "*(element)" allows any number, including zero; "1*element" requires at least one; and "1*2element" allows one or two.

Quoted Pair

- quoted-pair = "\" CHAR
- i.e if you want to literally use \ in a string, then you have to escape it with \
  - i.e. \\

Overview of RFC 2229

```
atom        =  1*<any CHAR except SPACE, CTLs, <'>ƒ">, and "\">
string      =  *<dqstring / sqstring / quoted-pair>
word        =  *<atom / string>
description =  *<word / WS>
text        =  *<word / WS>
```


Overview of RFC 2229

When a client initially connects to a DICT server a code 220 is sent if the client's IP is allowed to connect:

```
220 text capabilities msg-id
```

The code 220 is a banner, usually containing host name and DICT server version information.

The second-to-last sequence of characters in the banner is the optional capabilities string, which will allow servers to declare support for extensions to the DICT protocol. The capabilities string is defined below:

```
capabilities =  ["<" msg-atom *("." msg-atom) ">"]
msg-atom     =  1*<any CHAR except SPACE, CTLs,
                   "<", ">", ".", and "\">
```


Available commands

- DEFINE
- MATCH
- SHOW
  - DATABASES
  - STRATEGIES
  - INFO
  - SERVER
- CLIENT
- STATUS
- HELP
- QUIT

The define command

```
3.2.  The DEFINE Command

   DEFINE database word

3.2.1.  Description

   This command will look up the specified word in
   the specified database.  All DICT servers MUST
   implement this command.
```

Assignment 1: TFTP client and server

- Not all network protocols are text based.
- You will implement one of these for assignment 1:
  - Trivial File Transfer Protocol (TFTP)
  - Stop-and-wait file transfer protocol, using UDP
  - Designed to be simple to implement
  - Part of your task is to test interoperability of both your client and server implementations with a classmate's client and server implementations.
- Verification due Fri March 20th

Further reading

- Computer Networks, 4th edition
  - Pages 71-77
- Assignment 1: Trivial File Transfer Protocol (TFTP) client and server