st

Network Working Group G. Scott
INTERNET DRAFT Defense Information Systems Agency
August 1996

Guide for Internet Standards Writers
<draft-ietf-stdguide-ops-00.txt>

Status of this Document

This Internet Draft expires 21 February 1997.

Abstract

This document is a guide for Internet standard writers. It defines those characteristics that make standards coherent, unambiguous, and easy to interpret. Also, it singles out usage believed to have led to unclear specifications, resulting in non-interoperable interpretations in the past.

This version of the document is a draft. It's intended to generate further discussion and addition by the STDGUIDE working group. Please send comments to stdguide@midnight.com or to the author.

Table of Contents

1 Introduction

This document is a guide for Internet standard writers. It offers guidelines on how to write a protocol specification with clarity, precision, and completeness. These guidelines are based on both prior successful and unsuccessful IETF specification experiences. Note that some guidelines may not apply in certain situations.

The goal is to increase the possibility that multiple implementations of a protocol will interoperate. Writing specifications to these guidelines will not guarantee interoperability. However, a recognized barrier to the creation of interoperable protocol implementations is unclear specifications.

Many will benefit from having well-written protocol specifications. Implementors will have a better chance to conform to the protocol specification. Protocol testers can use the specification to derive unambiguous testable statements. Purchasers and users of the protocol will have a better understanding of its capabilities.

2 General Guidelines

It is important that multiple readers and implementors of a standard have the same understanding of a document. To this end, information should be orderly and detailed. The following are general guidelines intended to help in the production of such a document.

2.1 Protocol Description

Standards must include a description of the purpose or context of a protocol's use. The author of a protocol specification will have a great deal of knowledge as to the purpose of a protocol. However, the reader is more likely to have general networking knowledge and experience, rather than expertise in a particular protocol. Without an explanation of the purpose behind a protocol interpreting it is far more difficult, and a reader is more prone to error.

This also applies to the algorithms used by a protocol. A detailed description of the algorithms or citation of readily available references that give such a description is necessary.

2.2 Discussion of Security

If the Internet is to achieve its full potential in commercial, governmental, and personal affairs, it must assure users that deliveries of their information transfers are free from tampering or compromise. Well-written security sections in standard protocol documents can do much to achieve that condition. Implementors will find it easier to comply and do security. Users can understand the security measures in place, and so have faith in the Internet.

The security section should address several topics. Very important is a description of the security issues the protocol solves, and what issues remain unsolved. The effects the security measures have on the protocol's use and performance. If possible, the discussion should address how much insurance the implementation of the security measures achieves.

An author may not include security measures or considerations in the protocol standard. If so, a detail explanation why they did not is necessary. This discussion could present the reasons why the security issues are unresolvable at this time. Alternatively, the author could present a case why security is unneeded when using the protocol.

These security sections should be complete and stand alone. If security measures are part of the general protocol text, they will be difficult to find. If the security measures are not clear they may not be implemented, nor will a user be assured that they exist.

Finally, it is no longer acceptable that security sections consist solely of statements similar to: "Security issues are not discussed in this RFC."

2.3 Level of Detail

The author should consider whether concise or verbose text best conveys the protocol's intent. Concise text has several advantages. It makes the document easier to read. Such text reduces the chance for conflict between different portions of the specification. The reader can readily identify the required protocol mechanisms in the standard. Also, it makes it easier to identify the requirements for protocol implementation. A disadvantage of concise descriptions is that a reader may not fully comprehend the reasoning behind the protocol, and thus make assumptions that will lead to implementation errors.

Longer descriptions may be necessary, however, to explain purpose, background, rationale, implementation experience, or to provide tutorial information. This permits explanations at sufficient depth to insure understanding of the protocol. Yet several dangers exist with lengthy text. Finding the protocol requirements in the text is difficult or confusing. An increased risk that the same mechanism may have multiple descriptions, which leads to misinterpretations or conflict. Lengthy text is a challenge to the attention span of some readers. Finally, it is more difficult to comprehend, a consideration as English is not the native language of the many worldwide readers of IETF standards.

One approach is to divide the standard into sections: one describing the protocol concisely, while another section consists of explanatory text. The STD 3/RFC 1122/RFC 1123 1812 provides examples of this method.

2.4 Protocol Versions

Often the standard is specifying a new version of an existing protocol. In such a case, the authors should detail the differences between the previous version and the new version. This should include the rationale for the changes, for example, implementation experience, changes in technology, responding to user demand, etc.

2.5 Decision History

In standards development, reaching consensus requires making difficult choices. Including a discussion history and rationales for a decision can prevent future revisiting of these disagreements later, when the original parties have moved on. Occasionally, the alternative not taken may have been simpler to implement, so including the logic behind the choice may prevent future implementors from taking nonstandard shortcuts.

2.6 Response to Out of Specification Behavior

Recommend that detail description of the actions taken in case of behavior that is deviant from or exceeds the specification be included. This is an area where implementors often differ in opinion as to the appropriate response. By specifying a common response, the standard author can strike a blow against the law of unintended consequences.

The standard should describe responses to behavior explicitly forbidden or out of the boundaries defined by the specification. Two possible approaches to such cases are discarding, or invoking error-handling mechanisms. If discarding is chosen, detailing the disposition may be necessary. For instance, treat dropped frames as if they never were received, or reset an existing connection or adjacency state.

The specification should describe actions taken when critical resource or performance scaling limits are exceeded. This is not necessary for every case. It is necessary for cases where a risk of network degradation or operational failure exists. In such cases, a consistent behavior between implementations is necessary.

2.7 The Liberal/Conservative Rule

A rule, first stated in RFC 791, recognized as having benefits in implementation robustness and interoperability is:

> "Be liberal in what you accept, and
>   conservative in what you send."

Or establish restrictions on what a protocol transmits, but have few restrictions on what it will receive. To avoid any confusion between the

two, recommend that standard authors specify send and receive behavior separately.

The effect of this approach is that the description of reception will require the most detailing. For implementations will be expected to accept any packet from the network without failure or malfunction. Therefore, the actions taken to achieve that result, need to be laid out in the protocol specification. Standard authors should consider not just how to survive on the network, but achieve the highest level of cooperation possible to limit the amount of network disruption. The appearance of undefined information or conditions must not cause a network or host failure. This requires specification on how to attempt acceptance of most of the packets. Two approaches are available, either using as much of the packet's content as possible, or invoking error procedures. Specify a dividing line on when to take which approach.

A case for consideration is that of a routing protocol, where acceptance of flawed information can cause network failure. For protocols such as this, the specification should identify packets that could have differing interpretations and mandate that they be ignored. For example, routing updates contain more data than the tuple count shows.

2.8 Handling of Protocol Options

Standards with many optional features increase the chance of non-interoperable implementations. The danger is that different protocol implementations may specify some optional combinations that are unable to interoperate with each other. Ideally, implementation experience purges options from the protocol while the document moves along the standard track.

Options should only be present in cases where the protocol has an item that a particular marketplace requires, or because it enhances the product. The protocol specification must explain the full implications of either using the option or not, and the case for choosing either course. However, omission of the optional item should have no interoperability consequences for the implementation that does so.

Certain cases will require the specifying of mutually exclusive options within a protocol. That is, the implementation of an optional feature precludes the implementation of the other optional feature. For clarity, provide details on when to implement one or the other, what the effect of choosing one over the other is, and what problems the implementor or user may face. The choice of one or the other options should have no interoperability consequences between multiple implementations.

The most prevalent current practice in the specification of Internet standards is to identify mandatory protocol features by the term "MUST," and optional features by "MAY" or "SHOULD."

2.9 Notational Conventions

Formal syntax notations can be used to define complicated protocol concepts or data types, and also to specify values of these data types. This permits the protocol to be written with out concern on how the implementation is constructed or how the data type is represented during transfer. The specification is simplified because it can be presented as "axioms" that will be proven by implementation.

The formal specification of the syntax used should be referenced in the text of the standard. Any extensions, subsets, alterations, or exceptions to the formal syntax should be defined.

The STD 11/RFC 822 provides an example of this. In RFC 822 (Section 2 and

Appendix D) the Backus-Naur Form (BNF) meta-language was extended to make its representation smaller and easier to understand. Another example is STD 16/RFC 1155 (Section 3.2) where a subset of the Abstract Syntax Notation One (ASN.1) is defined.

2.10 Implementation Experience

For a protocol to be designated a standard, it must go through the rigors of actual implementation. This implementation experience should be captured in the final document. For example, lessons learned from bakeoffs between multiple vendors.

3 Specific Guidelines

The following are guidelines on how to present specific technical information in standards.

3.1 Packet Diagrams

Most link, network, and transport layer protocols have packet descriptions. Recommend that packet diagrams be included in the standard, as they are very helpful to the reader. The preferred form for packet diagrams is a sequence of long words in network byte order, with each word horizontal on the page and bit numbering at the top:

```
       0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |Version| Prio. |                   Flow Label                  |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

In cases where a packet is strongly byte-aligned rather than word-aligned (e.g., when byte-boundary variable-length fields are used), display packet diagrams in a byte-wide format. Use different height boxes for short and long words, and broken boxes for variable-length fields:

```
                              0 1 2 3 4 5 6 7
                             +-+-+-+-+-+-+-+-+
                             |    Length N   |
                             +-+-+-+-+-+-+-+-+
                             |               |
                             +    Address    +
                                    ...
                             +   (N bytes)   +
                             |               |
                             +-+-+-+-+-+-+-+-+
                             |               |
                             +  2-byte field +
                             |               |
                             +-+-+-+-+-+-+-+-+
```

3.2 Summary Tables

The specifications of some protocols are particularly lengthy, sometimes covering a hundred pages or more. In such cases the inclusion of a summary table can reduce the risk of conformance failure by an implementation through oversight. A summary table itemizes what in a protocol is mandatory, optional, or prohibited. Summary tables do not guarantee conformance, but serve to assist an implementor in checking that they have addressed all protocol features.

The summary table will consist of, as a minimum, four (4) columns: Protocol Feature, Section Reference, Status, and References/Footnotes. Use additional columns if they further explain or clarify the protocol.

In the Protocol Feature column describe the feature, for example, a command word. Group series of related transactions under descriptive headers, for example, RECEPTION.

Section reference directs the implementor to the section, paragraph, or page that describes the protocol feature in detail.

Status indicates whether the feature is mandatory, optional, or prohibited. Provide a separate column for each possibility, or a single column with appropriate codes. These codes need to be defined at the start of the summary table to avoid confusion. Possible status codes:

```
        M  - must                        M - mandatory
        MN - must not                    O - optional
        S  - should                      X - prohibited
        SN - should not
```

Use the References/Footnotes column to point to other RFCs that are necessary to consider in implementing this protocol feature, or any footnotes necessary to further explain the implementation.

RFCs 1122 and 1123 provide examples of summary tables.

3.3 State Machine Descriptions

A convenient method of presenting a protocol's behavior is as a state-machine model. That is, a protocol can be described by as a series of states resulting from a command, operation, or transaction. State-machine models define the variables and constants that establish a state, the events that cause state transitions, and the actions that result from those transitions. Through these models, understanding the dynamic operation of the protocol as sequence of state transitions that occur for any given event. Detailed text description of the state machines is necessary. Also, recommend the use of diagrams, tables, or timelines to detail state transitions.

When using a state transition diagram, show each possible protocol state as a box connected by state transition arcs. Label each arc with the event that causes the transition, and, in parentheses, any actions taken during the transition. The STD 5/RFC 1112 provides an example of such a diagram. As ASCII text is the preferred storage format for RFCs, only simple diagrams are possible. Tables can summarize more complex or extensive state transitions.

In a state transition table, read events vertically and states horizontally. Represent state transitions and actions in the form action/new-state. Use commas to separate multiple actions, and go on succeeding lines as required. Present multiple actions in the order they must be executed, if relevant. Letters that follow the state indicate an explanatory footnote. The dash ('-') indicates an illegal transition. The STD 51/RFC 1661 provides an example of such a state transition table. The initial columns and rows of that table are below as an example:

	State					
	0	1	2	3	4	5
Events	Initial	Starting	Closed	Stopped	Closing	Stopping
Up	2	irc,scr/6	-	-	-	-
Down	-	-	0	tls/1	0	1
Open	tls/1	1	irc,scr/6	3r	5r	5r
Close	0	tlf/0	2	2	4	4
TO+	-	-	-	-	str/4	str/5
TO-	-	-	-	-	tlf/2	tlf/3

The STD 18/RFC 904 also presents state transitions in table format. However, it lists transitions in the form n/a, where n is the next state and a represents the action. The method in RFC 1661 is preferred as new-state logical follows action. Also, this RFC's Appendix C models transitions as the Cartesian product of two state machines. This is a more complex representation that may be difficult to comprehend for those readers that are unfamiliar with the format. Recommend that authors present tables as defined in the previous paragraph.

A final method of representing state changes is by a timeline. The two sides of the timeline represent the machines involved in the exchange. List the states the machines enter as time progresses (downward) along the outside of timeline. Within the timeline, show the actions that cause the state transitions. An example:

```
          client                                     server

             |                                          |
             |                                          |   LISTEN
 SYN_SENT    |-----------------------                   |
             |                       \ syn j            |
             |                        ----------------->|   SYN_RCVD
             |                                          |
             |                        ------------------|
             |        syn k, ack j+1 /                  |
 ESTABLISHED |<----------------------                   |
             |                                          |
```

4 Glossary

Internet standards are to use the following terms. Deviations from the definitions given are discouraged, as it will likely cause misinterpretations among readers.

MAY

This word defines the existence of an item that is optional.

MUST

This word defines the existence of an item that is an absolute requirement of the specification.

MUST NOT

This phrase prohibits the use of the item.

OPTIONAL

This word specifies that implementation of an item is discretionary.

RECOMMENDED

This word specifies an item that there may exist valid reasons in particular circumstances to ignore.

REQUIRED

This word specifies an item that is an absolute requirement of the specification.

SHOULD

This word defines the existence of an item that there may exist valid reasons in particular circumstances to ignore.

SHOULD NOT

This phrase means that there may exist circumstances when the described behavior is acceptable or even useful. Even so, describe the full implications so that the implementor can carefully weigh the pros and cons of the behavior.

The above definitions are of a "contractual" nature. This RFC does not define technical terms. These definitions have been evolving with technology. Extensive and detailed technical definitions in documents aid understanding.

5.0 Document Checklist

The following is a checklist based on these suggestions which can be applied to a document:

- o Does it explain the purpose of the protocol?
- o Does it reference or explain the algorithms used in the protocol?
- o Does it give packet diagrams in recommended form, if applicable?
- o Does it use the recommended meaning for any of the terms defined in the glossary above?
- o Does it separate explanatory portions of the document from requirements?
- o Does it describe differences from previous versions, if applicable?
- o Does it give examples of protocol operation?
- o Does it specify behavior in the face of incorrect operation by other implementations?
- o Does it delineate which packets should be accepted for processing and which should be ignored?
- o Does it consider performance and scaling issues?
- o How many optional features (MAY, SHOULD) does it specify? If more than [X], does it separate them into option classes?

- Have all combinations of options or option classes been examined for incompatibility?
- If multiple descriptions of a requirement are given, does it identify one as binding?

6. Author's Addresses

Gregor D. Scott
Director, Defense Information Systems Agency
ATTN: JIEO-JEBBD
Ft. Monmouth, NJ 07703-5613 USA

Phone: (908) 532-7726
Fax: (908) 532-7723
EMail: scottg@ftm.disa.mil

7. References

RFC 791 "Internet Protocol (IP)," J. Postel, September 1981.

RFC 904 "Exterior Gateway Protocol formal specification," D. Mills, April 1984

RFC1112 "Host extensions for IP multicasting," S. Deering, August 1989

RFC 1122 "Requirements for Internet Hosts -- Communication Layers," October 1989

RFC 1123 "Requirements for Internet hosts -- Application and Support," October 1989

RFC 1311 "Introduction to the STD Notes"

RFC 1602 "The Internet Standards Process - Revision 2"

RFC 1661 "The Point-to-Point Protocol (PPP)," W. Simpson, July 1994

This Internet Draft expires 21 February 1997.