Testing Internet Applications - Terminology and Applicability

Mazen Malek * Roland Gecse *

Abstract

This paper examines the applicability of OSI conformance test methodology to Internet protocols. It summarizes the differences between them and introduces the Internet Reference Model along with a new abstract test method, which was designed for the practical purposes of conformance testing of TCP/IP protocols. Some interesting test cases and points, that were chosen from RIP, demonstrate the facilities of the model and give impression of testing Internet protocols.

1 Introduction

Up to now, in the Internet community, conformance testing was an unknown concept. However, the need for recommendation conforming TCP/IP implementations grows, as the application of Internet protocols in business telecommunication systems is becoming reality. It is probable that more and more vendors are going to provide Internet products, whose reliability and interoperability with other products have to be assured.

Although conformance testing methodology [1] was originally intended for OSI based systems, there are ongoing discussions about its applicability to the TCP/IP protocol stack. Numerous articles and conference contributions justify that these questions present a current topic. [2] founds theoretical base of relay system testing, which is then used, among others, for the testing of Simple Mail Transfer Protocol [3] and IP router. [4] and [5] focus on detailed analysis of Transmission Control Protocol's flow control algorithms that are expected to be used in measuring and fixing the majority of implementation problems listed in [6]. On the other hand, [7] deals with interoperability test suite derivation that may be used for the purpose of Internet testing.

The following issues, beside others, will be argued in this paper. Sections 2-5 give an overview of the Internet protocol structure, introduce the Internet Reference Model and suggest a new abstract test method. Also, similarities and differences in layering, data flow and configuration are fetched in comparison to the OSI Basic Reference Model (BRM) [8]. After the presentation of a possible test realization

*Conformance Center, Ericsson Ltd., Laborc u. 1., H-1037 Budapest, Hungary

(section 6) and a short overview of the Routing Information Protocol and related Internet routing concepts (section7), sections 8 and 9 give some practical testing experience.

2 Comparing Internet and OSI architecture

The OSI BRM has 7 layers, each of which with a well-defined task. OSI protocol stacks are designed to fit to this model. The protocol entities (PEs) of a particular protocol suite are associated to the appropriate layers. Peer-to-peer communication between two PEs of the same layer takes place in abstract protocol data units (PDUs) while physical communication with upper and lower layers' PEs is only possible via service primitives (SPs).

Unfortunately, Internet was not planned to have such a detailed abstract model. The structure of TCP/IP, which represents the actual state of Internet, has evolved gradually from the beginning [9]. Internet has only four layers: link, network, transport and application. Although the general functions of these layers are not as well defined as OSI's, they provide almost the same functionality. Disregarding that reliable service appears first only in the transport layer, network and transport layers map to their OSI counterparts. Internet link layer maps, in general, to OSI physical and data-link layers. Since the application layer holds all remaining functionality (OSI layers 5-7), applications may gain enormous complexity. Internet protocols do not have standardized SPs, thus in contrast to open systems; the communication between neighboring layers is implementation specific. This, besides the loosely specified layer characteristics, results that layer boundaries are flexible. Another feature that must be kept in mind when talking about Internet is the whole TCP/IP protocol stack should be considered as a single unit together with a set of alternative protocols. The transport layer for example consists of two protocols: the transmission control protocol (TCP), which is a connection-mode service and the user datagram protocol (UDP) that provides a connectionless service. In a particular communication process, at most one of these services is used.

From the configuration point of view a real open system can act as end system, relay system or both simultaneously. Internet systems have also this kind of configurations with noting that relay systems are called also intermediaries. Intermediaries are further subdivided according to working aspects to proxy, gateway and tunnel. In our paper we limit our discussion to relay systems and call them routers.

3 Conformance testing of Internet protocols

From the conformance testing perspective it is worth to distinguish between hardware and software implementations. Hardware implementations (e.g. IP router) neither implement the whole TCP/IP protocol stack, nor provide interface to protocol layers. Accordingly, they could be examined only by an external test system. Software implementations (e.g. FTP client, httpd programs), on the other hand, have numerous advantages over hardware systems. Besides the existing test methods [12], they imply the possibility of designing more effective new test methods. For the understanding of these methods, a particular TCP/IP implementation should be examined.

4.4BSD-Lite's Net/3 networking code [10] can be considered as a reference implementation of the Internet protocol suite (Besides TCP/IP, it also supports Xerox Network Systems (XNS), OSI communication protocol families and the Unix domain protocols that are provided for Inter Process Communication (IPC)).

The structure of the Net/3 networking code is presented in figure 1. Application level protocols (FTP, Telnet, and RIP) are distinguished from the underlying TCP/IP stack. They are running as processes in the device's user space while underlying layer protocols used to be implemented as a single unit in the operating system space. The internal structure of this unit consists of three layers: application programming interface (API) or socket layer, protocol layer and interface layer. The public functions of this unit can be reached at the kernel entry points using system calls (SCs) which represent the operating systems' service primitives. API, in addition to separating the application layer, provides a protocol independent interface to the entities of the underlying protocol layer. It offers a set of different networking features of the kernel that can be reached uniformly via SCs. The protocol layer holds the Internet transport (UDP, TCP) and network (IP, ICMP, IGMP) layer protocols [11]. The protocol layer does not provide SCs to application layer entities. The interface layer consists of various device drivers implementing link layer protocols (e.g. Ethernet) and procedures that are used for address conversion between the protocol layer and it. The code for different pseudo devices (loopback interface, BSD packet filter (BPF)) can also be found there. Interface layer functions are accessible through SCs. The packet filtering functions are further applicable for control and observation. Now, having a global picture of the overall structure of TCP/IP, the Internet Reference Model will be introduced.

4 The Internet Reference Model

It can be stated that all of today's software TCP/IP implementations are based upon the architecture of Net/3. By considering this, a model will be introduced that is suitable for conformance testing and incorporates the listed features of software implementations. In the Internet Reference Model (IRM), In Figure 1, the functions of SPs are replaced by SCs of API (In this context, API is used as a general term, which in a particular implementation (e.g. Net/3) stands for both socket API and BPF. That is, because the socket API does not provide access to the interface layer). These SCs allow applications to send PDUs directly to each layer protocol entity. The API itself should be considered as a switch that connects applications to the selected underlying service via SCs. The functions of the API are provided at kernel entry points (rhombus). The semicircles present the possible destination protocol layers to which SCs provide access. The dashed line expresses that API itself is not a protocol. Although the IRM has some minor differences from OSI BRM, which are coming from design aspects, the applicability of existing conformance test methodology is straightforward.

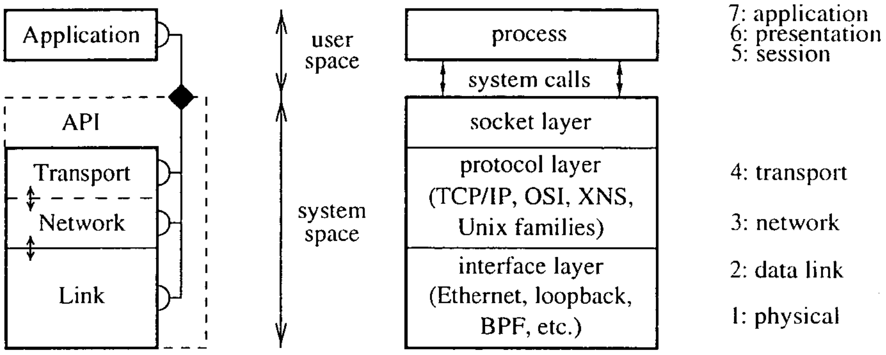

Figure 1: The Internet Reference Model (left), general organization of Net/3 networking code (right)

5 Abstract Test Methods

Considering the open structure of software implementations, the new Joint Test method (JT) will be defined, which can be uniformly applied to testing of all protocols of IRM. JT can be applied both in Single Party Testing (SPyT) and in Multi Party Testing (MPyT) context. When used in SPyT, it resembles to the local [1] test method, whereas the MPyT variant has similarities to the local transverse test method in [2].

JT is shown in (Figure 2), and uses the graphical notation of [12].

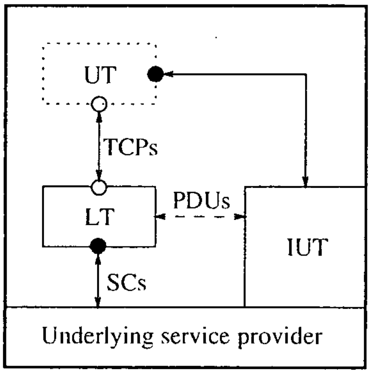

Figure 2: The joint test method

JT has the following characteristics:

- Test system and system under test (SUT) are on the same system.
- There is an optional Upper Tester (UT), and one Lower Tester (LT) in SPyT; no UT, an arbitrary number (usually 2) of LTs and a Lower Tester Control Function (LTCF) in MPyT. UT, LT(s) and LTCF are application layer processes.
- The Points of Control and Observation (PCOs) are at the LT and UT.
- Test coordination is done using Unix IPC.
- Test events are exchanged in PDUs using SCs of API. The control and observation is provided by means of API.

The most significant difference to the ancestor test methods, which is very advantageous in practical testing of software TCP/IP implementations, is that LT(s), UT and coordination procedures are placed in the application layer regardless of the layer which is occupied by IUT. Another good feature is that JT can be applied to both end systems (SPyT) and relay systems (MPyT), thus intermediaries can be tested out of their environment.

6 Test realization

Having an implementation to be tested and an abstract test suite (ATS), the means of testing should be provided. It consists of the implementation of tester functionality, the derivation of ATS into executable test suite (ETS) and the production of test documents. System Certification System (SCS) is a set of tools provided by Ericsson that can be used in a wide variety of testing: functional testing (white-box technique), conformance and interoperability testing (black-box) and performance testing (white/black-box). SCS is based on the following principles:

- Protocol independence. This means that different protocols can be tested on the same manner.
- Multiple simultaneous protocols. Not only one but also many protocols can be accessed from the same test.
- Distribution. One test may be distributed (over the Internet), making it possible for each part of the test most closely related to one interface to reside in the box containing that physical interface.
- Platform independence. SCS is independent of the platform in which the SUT executes. It can execute the same tests both against the physically real SUT and the SUT only simulated in a workstation (bypassing the lowest protocol layers).

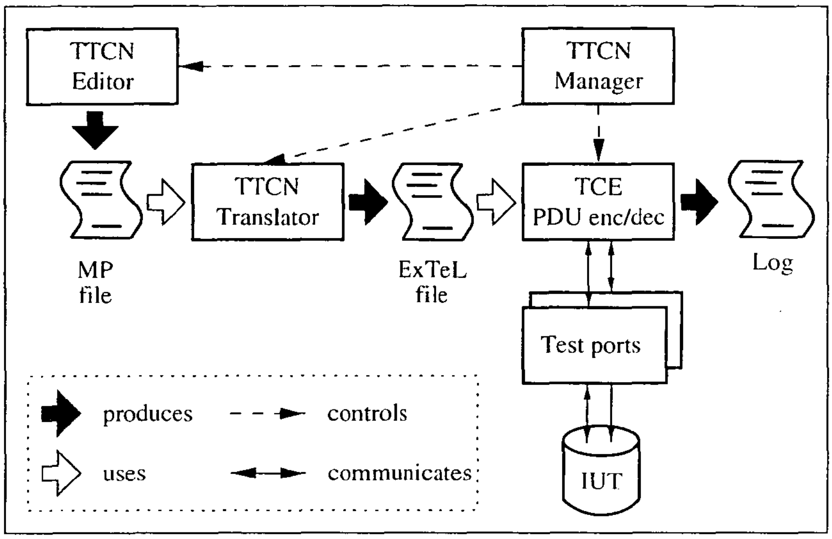

Figure 3: SCS structure

One of the main ideas in SCS is that it is an interpreting execution platform. This means that a TTCN test suite (an MP file) given as input to the Translator is first converted into an intermediate language, ExTeL (Executable Test Language), which then can be directly executed (interpreted) by the ExTeL Test Component Executor, TCE (see also Figure 3 above). With this method there is only one phase from a TTCN test suite to the final executable format which makes it different compared to the compiling methods, where an extra compilation and linking phase has to be performed. Another important feature in SCS is the Test Port concept. With this solution it is possible to develop the core functionality separately without affecting the existing test ports and vice versa. For this reason, a complete set of test ports where realized and worked out by the authors, particularly, IP, TCP and UDP test ports. There exist also two built-in PDU encoder/decoders: BER (Basic Encoding Rules) and a raw binary encoder/decoder. TTCN Manager is the front end in SCS. It has the control over execution and monitoring. The log files for different test components can be observed in real time.

7 Routing techniques

Routing involves two basic activities: determination of routing paths and the transport of information groups (packets) through an internetwork. In some literatures the later is referred to as switching. Path determination, on the other hand, can be very complex. To aid the process of path determination, routing algorithms initialize and maintain routing tables, which contain route information. Usually, routing algorithms fill routing tables with destination/next hop associations. These associations tell a router that a particular "destination" can be gained optimally by sending the packet to the node identified in "next hop". Routers communicate with one another (and maintain their routing tables) through the transmission of a variety of messages. The routing update message is one such message. Routing updates generally consist of all or a portion of a routing table. They are the means by which routers communicate path information between one another. In this paper we limit our illustration on a very common routing technique, it is the RIP protocol. It is a distance vector, intra-domain routing protocol originally designed for PUP (Xerox PARC Universal Protocol, 1980) and used in XNS. RIP became associated with both UNIX and TCP/IP in 1982 when the Berkely Standard Distribution (BSD) implementation of UNIX was introduced.

8 Interesting issues in testing routing

When time comes and routing testing is to be performed, a lot of interesting points should be considered. As the basic representation of test cases is the stimuli-response pairs, one should define precisely the types and instances of such pairs. Accordingly, routing stimuli and responses were defined. We have found that we need to deal with two different layer concepts. Firstly, the Internet layer and its datagrams that is used to check the arrival of forwarded data. Secondly, the application layer, or the routing application in question. Within this context, majority of testing events will be written by applying types and instances of this application. Both stimuli and responses, at application level, are in a form of routing updates. These updates can be gotten at prescribed time intervals or synchronized to other testing events. Another point of interest is the alternatives to those responses. They may have the form of modified routing update, in terms of metrics and/or routes.

A conformance test case has, typically, three phases of actions, preamble, test case body and postamble. The first and third phases are for state transition, to initial testing state and to idle state respectively. In the case of routing application, state inquiry is only possible when the routing table is to be accessed, through the usual updates or by asking the table on demand.

In the Internet, routing is the main building block in its backbone. That's why they play a crucial role when we think of the enormous growth of the Internet. Another important issue here is that routing techniques are evolving more quickly than host ones.

9 Example

In this section we try to illustrate our method of testing on a simple configuration of three test components connected to the IUT. Each test component has the role of simulating another router in the network, while connections through the configuration tries to indicate the subnetworks involved in the testing. According to the standard [13]: "An internet router should be capable of choosing a next-hop destination for each IP datagram", and according to standard [14]: "After the validation process of a response finishes successfully, an update that changes the metric of an existing entry in the routing table should be a trigger for entry modification". This modification varies between recalculating the metric according to the following formula:

$$metric = MIN(metric + cost, 16)$$

where 16 indicates that a link is inaccessable, and adding a new entry. In our example, we have established the tested condition as if one route is substituted by the other. In particular, we have informed the IUT that the desired destination (address included in an IP datagram) is reachable with a lower cost through B router instead of C router. In terms of Tree and Tabular Combined Notation (TTCN) [15], the official test notation of the conformance testing, a preamble will contain the preliminary parts of testing; i.e. getting the configuration working, sending an update form B router (within the testing context it is a test component) with a cost of the destination less than what is sent by C, and sending an IP datagram from A with the destination address. Test case body, however, is a set of test events that is responsible for issuing a test verdict, indicating the correctness of behaviour. Here, it is the arrival of the sent IP datagram at B instead of C. Before ending the test case, an action should take place to get the implementation right to its original state. This is possible by refreshing the routing table with the original route update, in terms of RIP it means a new response from C with higher cost or form B with lower cost. Figure 4 demonstrates the overall distribution of testing functionality. In the laboratory environment, testing was handled with software components, called gatewas, and with the standard SCS interface together with the newly defined test ports. Accordingly, IP test port was used to transfer the sample IP datagram and UDP test port to send and receive routing updates. The two solutions differ in the appearance of test cases. In the first solution, test cases where included in the code part of the gateways, while the other used an interface to a TTCN editor. In short we list below the tasks associated to test components.

Tasks of test components:

- Simulate RIP
- Test coordination & management (remote/local)
- Execute Test Cases

According to this example, the following Test Purpose was defined.

Test Purpose 1.1:

"To check if the router applies to changes in network topology after receiving the required update from other routers"

Abbriviations:

- IUT - Implementation Under Test
- A, B, C - gateways
- a, b, c, ai, bi, ci - networks

Roles of gateways

- Client
- server
- Listener

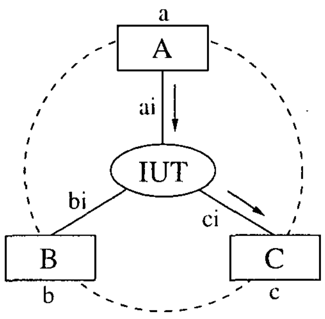

Figure 4: Illustration of a test application for IP routing

10 Conclusions

In this paper, differences between OSI and Internet systems were summarized. Then the Internet Reference Model was introduced together with the Joint Test method for conformance testing (which is major contribution to the base methodology). Afterwards, a practical application of the new concept was demonstrated on the testing of RIP routing protocol, which is a continuation of the progress reached in [16]. The Test Purpose viewed here is part of a complete set to provide complete testing for routing applications. We have shown a framework for testing Internet protocols, which was worked out on the basis of the conformance testing framework of [1]. Experiences with testing RIP showed that some extensions are necessary to TTCN for making it more suitable to describe test cases for testing Internet protocols. This is true especially for testing performance related features of the product. Our interest for the time being is to generate a complete test suite to test a routing application, RIP2 or OSPF. Future work can be for example the interoperability testing based on this concept of Internet protocols, and introduction of formal extensions that are more suitable for Internet testing.

References

[1] ITU-T X.290-X.296, OSI conformance testing methodology and framework for protocol recommendations for ITU-T applications, 1994-1995.

[2] Bi, J. and Wu, J.: Towards abstract test methods for relay system testing. Testing of Communicating Systems, Volume 10 pp 381-397, IFIP, 1997.

[3] Bi, J. and Wu, J.: Application of a TTCN based conformance test environment on the Internet email protocol. Testing of Communicating Systems, Volume 10 pp 324-330, IFIP, 1997.

[4] Kato, T., Ogishi, T., Idoue, A. And Suzuki, K.: Design of Protocol Monitor Emulating Behaviors of TCP/IP Protocols. Testing of Communicating Systems, Volume 10 pp 416-431, IFIP, 1997.

[5] Kato, T., Ogishi, T., Idoue, A. and Suzuki, K.: Intelligent Protocol Analyzer with TCP Behavior Emulation for Interoperability Testing of TCP/IP Protocols. Formal Description Techniques and Protocol Specification, Testing and Verification, FORTE X/PSTV XVII '97 pp 449-464, IFIP, 1997.

[6] Paxton, V. (editor), Allman, M., Dawson, S., Heavens, I. and Volz, B.: Known TCP Implementation Problems, ¡draft-ietf-tcpimpl-prob-02.txt¿. Internet Draft, IETF Network Working Group, May 1998.

[7] Malek, M. and Dibuz S.: A Pragmatic Method for Interoperability Test Suite Derivation. EUROMICRO'98, Proceedings of the 24th Euromicro Conference, Stockholm, Sweden, 1998. Aug 24-26.

[8] ITU-T X.200, Information Technology - Open Systems Interconnection Basic Reference Model: The Basic Model, 1994.

[9] Carpenter, B. (editor): Architectural Principles of the Internet, RFC 1958 Informational, IETF Network Working Group, 1996.

[10] Wright, G. R. and Stevens, W. R.: TCP/IP Illustrated, Volume 2, The Implementation. Addison-Wesley, 1995.

[11] Stevens, W. R. TCP/IP Illustrated Volume 1, The Protocols. Addison-Wesley, 1994.

[12] Baumgarten, B. and Giessler, A.: OSI conformance testing methodology and TTCN, North. Holland, 1994.

[13] P. Almquist:. Towards requirements for IP routers, Network working group, RFC1716, November 1994.

[14] C. Hedrick: Routing Information Protocol, Network working protocol, RFC1058 June 1988.

[15] ISO/IEC 9646-3, The Tree and Tabular Combined Notation (TTCN), 1998.

[16] Gecse Roland: Conformance testing methodology of Internet protocols, Internet application-layer, Protocol testing- the Hypertext Transfer Protocol. The IFIP 11th International workshop on Testing of Communicating Systems IWTCS'98, August 31- September 2, 1998, Tomsk, Russia