TCP/IP Traffic Classification Based on Port Numbers

Patrick Schneider
Division of Applied Sciences, Harvard University, 29 Oxford Street, Cambridge, MA 02138, USA
Patrick.Schneider@Switzerland.ORG

Abstract

A new approach called TCP/IP Traffic Classification Based on Port Numbers is presented here. This approach is primarily based on the correlation between the transport layer's port number and the corresponding application or group of applications. We further suggest that advanced classification models are used in combination with the port number based traffic classification. Future routers can provide different quality of service (QoS) to flows depending on the port number or range of port numbers they are using. A combination of multiple characteristics enables a router to select real-time flows and handle them with a higher priority. The hit-rate can be increased by using additional characteristics such as packet interarrival time distribution and packet size distribution in addition to the port numbers. Measurements of real-world traffic show the percentage of Internet traffic.

1. Introduction

1.1. Port Numbers

While IP addresses determine the physical endpoints of a network connection, port numbers [1] determine the logical endpoints of the connection. Port numbers are 16-bit integers with a useful range from 1 to 65535.

1.1.1. Well-Known Port Numbers

In a client-server application, the server usually provides its service on a well-known port number. Well-known port numbers are a subset of the numbers which are assigned to applications. According to RFC1700 [5], well-known port numbers are managed by the Internet Assigned Numbers Authority (IANA). They used to be in the range from 1 to 255, but in 1992 the range was increased up to 1023.

Each Unix system holds a list containing all the well-known port numbers known to the hosts. This list is stored in the file `/etc/services`.

1.1.2. Registered Port Numbers

In addition to the well-known ports below 1024 there are more port numbers assigned to applications but are located anywhere from 1024 to 65535. While the IANA can not control uses of these ports it does register or list uses of these ports in RFC1700 [5] as a convenience to the community. On most systems the registered ports can also be used by ordinary user processes or programs executed by ordinary users.

1.1.3. Reserved Port Numbers

A port number in the range of 1 to 1023 is called a Reserved Port Number. In UNIX operating systems only a process with superuser privileges can assign itself a reserved port. Some applications, notably Rlogin, Klogin, Login and SSH, use the concept of reserved ports as part of their authentication between the client and server (server on well-known port while client on reserved port).

1.1.4. Ephemeral Port Numbers

Ephemeral port numbers are for short lived connections, as used usually by clients. This is because a client typically exists only as long as the user running the client needs its service, while servers typically run as long as the host is up. In Unix ephemeral ports are in the range from 1024 to 5000. The Solaris 2.2 operating system represents a notable exception. By default it places ephemeral ports in the range from 32768 to 65535 (the physical limit of the 16 bit port number range).

Most PC based TCP/IP cards provide their ephemeral ports in the range of 1024 to 5000, since their software is based on the freely available and usable FreeBSD source code [6].

1.2. Classifying Traffic

In order to provide optimal quality of service (QoS) three basic types of traffic can be distinguished. Interactive data, bulk data and data for real time applications. The former two classes are considered as elastic data. While real-time

applications do not wait for late data to arrive, elastic applications will always wait for data to arrive. It is not that these applications are insensitive to delay; to the contrary, significantly increasing the delay of a packet will often harm the application's performance. An example of classifying flows is given in [4], definig six classes of traffic flow.

1.2.1. Interactive Data

Interactive data is of a bursty character with small packets, single or in small groups, being transmitted sporadically. Since each bit of interactive data is of high value in order to achieve correct performance, the reliable TCP connection is usually preferred over the simpler UDP protocol. The delay added to interactive data by the transmission must be as low as possible.

1.2.2. Bulk Data

If the flow is bulk data, large amount of data - in the range of kilobytes to typically several megabytes - are transferred. The delay requirements are rather lax for asynchronous bulk transfer such as electronic mail (unattended data transfer) or news ('filler' traffic), and of intermediate importance for interactive bulk transfer such as FTP data (attended data transfer). However, in order to achieve high performance, a high throughput should be aspired to. It is crucial for most bulk data applications to provide a reliable service.

1.2.3. Real-Time Data

In a real-time application, the source takes some signal, packetizes it, and then transmits the packets over the network. The network inevitably introduces some variation in the delay of the delivered packets. The receiver depacketizes the data and then attempts to faithfully play back the signal. This is done by buffering the incoming data and then replaying the signal.

The performance of a playback application is measured along two dimensions: latency and fidelity. Some playback applications, in particular those that involve interaction between the two ends of a connection such as a phone call, are rather sensitive to the latency; other playback applications, such as transmitting a movie, are not. Similarly, applications exhibit a wide range of sensitivity to loss of fidelity. Intolerant applications require an absolutely faithful playback, while tolerant applications can tolerate some loss of fidelity. It is expected that the vast bulk of audio and video applications will be tolerant.

If the data buffer exceeds a certain size (in seconds of material played back in real time) the real-time character of the data turns into bulk character. A large buffer enables a real-time application to wait for late packets (within the limits given by the buffer size) which is considered non real-time behavior.

It is hard to set an upper limit for the buffer that is still considered real time data. We will call traffic that produces a continuous output (audio or video) but allows any amount of buffering bulk data rather than real-time data. Namely audio transmissions over HTTP are considered bulk data.

2. Selected TCP/IP Applications

2.1. The File Transfer Protocol (FTP)

The commonly used application FTP is the Internet standard for file transfer. The FTP TCP protocol differs from most other applications because it uses two TCP connections to transfer a file.

A control connection is established on well-known TCP port 21 (FTP-control) for commands from the client to the server and for the server's replies. A dedicated data connection is created each time bulk data is transferred between two points. Two different modes can be entered alternatively. In the default port mode, the client does a passive open on an ephemeral port and communicates the port number via the control connection to the server. The server establishes the data connection by performing an active open on well known TCP port 20 (FTP-data) to the client's ephemeral port.

The PASV command toggles to the passive mode requesting the server to listen on an ephemeral port and to wait for a connection to be established by the client. The server tells the client its ephemeral port number. In the passive mode, both port numbers are ephemeral.

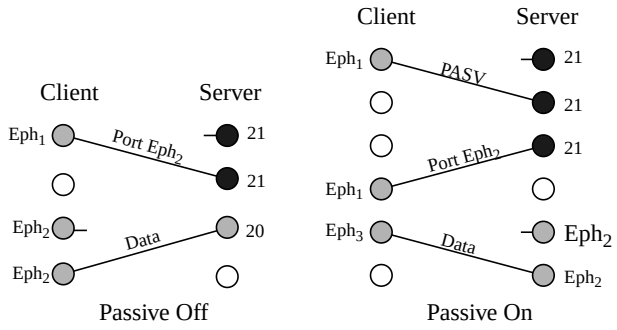

Figure 1: Simplified model for FTP using default port mode (left) and passive mode (right).

2.2. Remote Procedure Calls (RPC)

The Sun RPC protocol is based on the remote procedure call model, which is similar to the local procedure call model. One could think of the RPC protocol as the jump-to-subroutine instruction ("JSR") of a network.

RPC servers typically bind an ephemeral port and some a reserved one. After the kernel has allocated a port, it is

registered at the port mapper in order to keep track of which RPC programs are using which ports. The port mapper provides its services on the well-known port 111 (UDP and TCP). Section 29.4 of [1] describes the port mapper in detail. When a client wishes to make an RPC call to a given service, it will first contact the port mapper daemon on the server machine to determine the port number to which RPC messages should be sent. The port mapper makes dynamic binding of remote programs possible. Recently, the port mapper has become *rpcbind* due to the fact that it works not only in a TCP and UDP environment but over any transport layer.

To obtain information on the port numbers currently used by RPC servers, one can invoke the PMAPPROC_DUMP procedure by typing `rpcinfo -p` which returns a list containing all entries with program number, version number and corresponding port number in the port mapper database.

2.3. The Secure Shell (SSH)

SSH is a program to log into another computer over a network, to execute commands in a remote machine, and to move files from one machine to another. It provides strong authentication and secure communications over insecure channels. It is intended as a replacement for rlogin, rsh, and rcp. Additionally, SSH provides secure X11 forwarding and, as an option, secure forwarding of arbitrary TCP connections.

By default, SSH listens for connections on port 22 which has been officially registered for SSH [8]. The client binds the highest possible port in the reserved range. If the user does not change the DISPLAY environment variable, the connection to the X11 display is automatically forwarded to the remote side in such a way that any X11 programs started from the shell (or command) will go through the encrypted channel, and the connection to the real X server will be made from the local machine.

2.4. X11 Forwarding

By default, all the X11 forwarding is done on a port in the range from 6000 to 6063 (registered port range) and an ephemeral port. No encryption is performed on this data.

2.5. Hypertext Transfer Protocol (HTTP)

The Hypertext Transfer Protocol (HTTP) is an application-level protocol with the lightness and speed necessary for distributed, collaborative, hypermedia information systems. RFC 1945 [5] provides detailed information about HTTP/1.0.

On the Internet, HTTP communication generally takes place over TCP/IP connections. The default port is TCP 80, but according to RFC 1945 [5] other ports can be used. Ports different from 80 are sometimes chosen in order to allow more than one HTTP server to run at the same time on the same host. Port 81 is a ‘hot standby’ port since it is not assigned to a different server. Multiple concurrent servers on the same host are sometimes considered necessary in order to improve the overall performance for hefty frequented hosts.

2.6. Real Time Audio And Video

Some real time applications send their data over TCP port 80. They are HTTP/1.0 compliant clients (see RFC 1945 [5] for details), thus they get their data from a www server using the HTTP/1.0 GET method. That is done through the TCP protocol at port 80 just like a Web browser would do.

While only a few samples of real time applications are shown here, the two major classes are covered: Real-time data over UDP or HTTP (TCP). Table 1 summarizes the port numbers used by the real-time applications we investigated.

Table 1: Summary of Real Time Applications

Application	Control	Data
Free Radio Ethernet	-	UDP 5002
Televox (player)	TCP 12380	UDP 12380
Televox (phone)	TCP 9010	-
StreamWorks	UDP 1558	UDP ephemeral
GSM Direct Audio	TCP 80 (HTTP)	dito
WinPlay3	TCP 80 (HTTP)	dito
TrueSpeech	TCP 80 (HTTP)	dito
Internet Wave	TCP 80 (HTTP)	dito
RealAudio/Video	TCP 7070	UDP 6800 - 7200
VDOLife Player	TCP 7000	UDP 7001

3. Traffic Classification Based on Port Numbers

To give information about the type of classic on a specific port number or range of port numbers we introduce a two-stage classification model which is presented in this section. Basically the first stage outputs the application for a given port number, respectively range of port numbers, while the following stage computes the type of traffic for the given application.

3.1. Basic Ranges by Definitions

Five basic ranges of port numbers provide port number spaces for commonly used and new servers, clients, and Unix processes requiring a special authentication. These five basic ranges are listed in table 2.

Table 2: Basic Ranges of Port Numbers

Name	Application	Port Numbers
Well-known	Server	1-1023
Registered	Server	1024-65535
Free	Server	5001-32767
Ephemeral	Client	1024-5000 32768-65535
Reserved	Superuser	1-1023

3.2. Mapping port numbers to applications

Every assigned port number found in RFC1700 [5] can be easily mapped to its corresponding application. The reverse process is possible, too but there are certain limitations regarding the reliability. Applications that have registered port numbers assigned to them will always use these numbers. Nevertheless different applications may use the same ports as long as they are available. Thus a reliable identification of an application is only guaranteed in the well-known port range from 1 to 1023. A problem arises with the RPC services because they may use ports in the reserved range as well as in the ephemeral range. Fortunately RPC services are mostly used within a LAN and do not appear on the Internet itself, thus they are only a negligible percentage of the traffic found on a WAN.

Some performance optimized FTP implementations send their bulk data over an ephemeral port rather than using well-known port 20. The only reliable way to determine which flows contain FTP data is by matching their IP addresses against the IP addresses of the concurrent FTP-control flows.

The use of Proxy servers has been spirallingly increased during the last few years. Proxy servers act as an interface between a certain application (such as HTTP) and the Internet. They change the original service's port number to any number depending on the proxy server and its settings. Many HTTP proxies convert the original port 80 to port 8080 but different ports are frequently used, too. As long as there is no standard for proxy servers on what port numbers to use, there is no way to guess the application from the used port number.

The upper part of figure 2 shows the mapping of port numbers to application classes.

3.3. Mapping Applications to Traffic Classes

For most applications we can define a traffic class. An FTP-data connection will always carry bulk data while TELNET handles interactive data. However, this mapping is aggravated or turns out to be impossible when it comes to applications such as SSH, multiplexing several independent and unpredictable sources over one TCP connection.

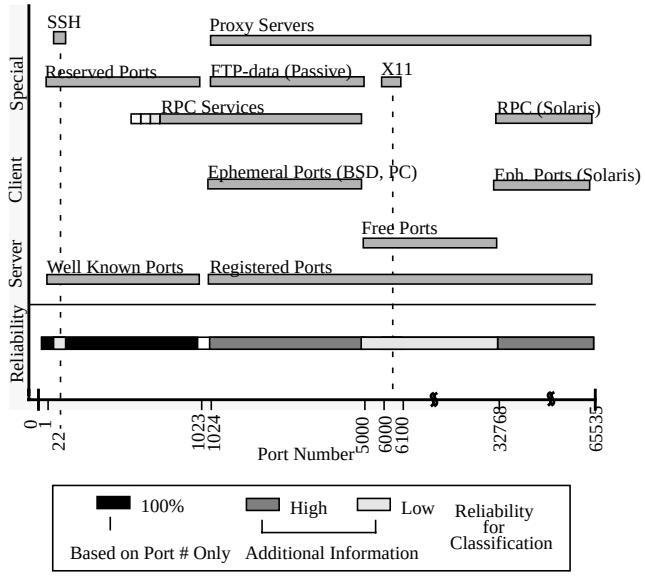

Figure 2: The port number map and reliability for classification.

In figure 2, the black part of the bar (100%) means that we can consider traffic on these ports to belong to their corresponding traffic class. This 100% reliability is obtained by evaluating port numbers only.

Without further investigation (i.e. by only watching the port number of a packet) there is no way to decide what traffic occurs on ports other than the ones in the black-marked range.

Additional means such as source and destination address matching for possible FTP-data packets against FTP-control flows (multiple concurrent connections), increase the reliability to detect FTP-data packets from an application running in the passive mode.

3.4. Limitations of the Port Number Based Classification Method

Port numbers were not intended to indicate the traffic class. Thus it is often hard or even impossible to give reli-

able information on the traffic by watching the mere port numbers. There are several major drawbacks:

- Multiple traffic over some applications (such as SSH) fades all chances to determine traffic character by watching the port number.
- A unique mapping is not possible if two or more applications or classes of applications use the same port or range of ports. Namely every registered port resides in a non-unique port number space.
- Dynamic allocation for server ports (to be found with RPC services) disables the possibilities to build a static look-up table to map these ports to the applications. Dynamic generation and up-dating of look-up tables is non-trivial and produces additional traffic (by contacting every host's port mapper).
- FTP can run in either, its default port mode (well-known port 20), or in the performance optimized passive mode (ephemeral port). Matching IP addresses against FTP-control IP addresses increases the probability to locate passive FTP-data traffic, but needs some additional CPU power and memory.

4. Interpretation of the Results

4.1. Traffic Classification by Port Numbers

The main goal of this thesis is to show the possibilities and limitations of classifying traffic based on the involved port numbers. Accurate traffic classification is possible for well-known port numbers in the range between 1 and 1023. It can be done by simply mapping port numbers to their corresponding traffic class. Table 3 summarizes the amount of traffic that can be classified by port numbers only (left block) and in combination with passive FTP detection (right block).

Table 3: Percentage of Traffic that can be Classified by Port Numbers

Layer	Ports only by Packets	Ports only by Bytes	incl. pass. by Packets	FTP by Bytes
Total	68%	76%	75%	86%
UDP	31%	24%	31%	24%
TCP	70%	77%	77%	87%

Although there is no intrinsic way to classify every packet on the Internet, more than three quarters of the overall traffic (by bytes) can be reliably classified by mere mapping of the well-known port numbers to their corresponding traffic classes. By including passive FTP detection, the percentage of TCP traffic that can be classified rises by 7 points to 77% when counting packets (improvement of 10%) and 10 points to 87% when counting bytes (improvement of 13%).

As of October 1996 the approach *Traffic Classification By Port Numbers* was an adequate way with a good performance-to-cost ratio. It is also implemented in commercially available routing products (e.g. Cisco Systems' NetFlow™ Switching software [19]). A powerful classification model that meets the future's needs must provide reliable detection of real-time traffic in order to be useful. The Internet and the vast number of services provided are growing, though and the tendency of the new applications transmitting data over the Net is going towards a dramatic increase of services that do not or not yet have a well-known port assigned by the IANA. Especially the up-coming real-time applications do not have dedicated ports at all. Since many of the new high-bandwidth applications use the simple UDP protocol, and also don't provide traffic congestion control on the application layer, their rapid growth calls for a rigorous routing policy in order to avoid a severe decay of performance due to heavy congestion. Since UDP packets are sent regardless of the traffic situation, too many packets increase not only the loss rate but also need resources in order to be handled, partially transmitted and finally discarded.

4.2. Suggestions for Further Research

Traffic can be classified by several means. A combination of several independent traffic classification means could lead to a satisfactory overall performance. We suggest here an algorithm that combines the relevant traffic classification methods in order to obtain a reliable probability for a given flow to be a specific class of traffic.

A brief outline for real-time evaluation is given as follows: Starting with an initial value Φ=1, each function f_i =[0..1] computes the probability of the input being real-time data. We assume that every function f_i finds all real-time data but also includes some non-real-time traffic and that different functions f_i produce false output for different traffic, thus the resulting Probability P_{RT} is significantly more reliable than any of the functions f_i by its own.

$$P_{RT} = \Phi \cdot f_1 \cdot f_2 \cdot f_3 \cdot f_{4\ldots} \cdot f_n$$

We name the algorithm *Subtractive Traffic Classification Cascade*. The term subtractive refers to the fact that it starts off with a preliminary probability of 1 while every function f_i can only decrease the probability. If one of the involved functions f_i outputs a probability of zero, the packet can never be considered real-time traffic again, independently of the other functions' opinions. Also, every function f_i can be weighted by limiting the lowest possible output for an input that does not match the pattern at all. For example if one considers function f_n to be a little uncertain, its output range

can be set to [0.5..1], thus limiting the possible damage for a fail to 0.5 instead of 0.

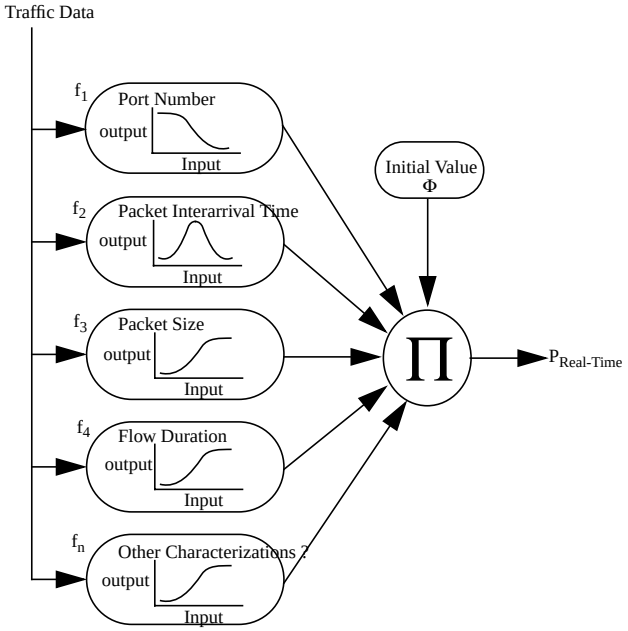

Figure 3: Block diagram of the subtractive traffic classification cascade.

In its basic version the *Subtractive Traffic Classification Cascade* only outputs the probability for one traffic class. Modifications could also include multiple traffic class detectors, one for interactive traffic, one for bulk traffic, and one for real-time traffic each.

Recall that this is not a complete implementation of an algorithm. Due to time limitations, only the basic idea could be presented here. We encourage everybody to do further research based on the results and ideas shown in this report in order to improve the performance of the Internet which is about to reach its limits in the current configuration. A powerful priority system based on a reliable traffic classification is able to relieve this distress.

5. Conclusion

The thesis presents *TCP/IP Traffic Classification Based on Port Numbers*, a new approach to distinguish between different types of TCP/IP traffic in order to provide priority oriented traffic routing in a overcrowded network such as the Internet. Based on fundamental TCP/IP theory, network specifications, and source code analysis, the theoretical possibilities and limitations are pointed out. A classification model based on port numbers is introduced and explained, using this information.

Finally, additional means to further improve the reliability of the classification model are discussed and the use of a new algorithm called *Subtractive Traffic Classification Cascade* is suggested.

Based only on the port number, completely reliable information about the service can never be given, because there are no restrictions on binding ports to applications that are compulsory to all operating systems, nor have all applications uniquely assigned port numbers. However most of the classic applications have got and make use of their well-known port numbers, enabling the classification based on port numbers to reliably identify the application. Many applications have only one kind of traffic (e.g. TELNET: Interactive), others such as HTTP may have different classes of traffic.

Additionally, the traffic of passive FTP-data connections can be detected by IP address matching.

References

[1] W. Richard Stevens, “TCP/IP Illustrated, Volume 1“, Addison-Wesley, Reading, MA, 1994.

[2] Gary R. Wright and W. Richard Stevens, “TCP/IP Illustrated, Volume 2“, Addison-Wesley, Reading, MA, ‘95.

[3] W. Richard Stevens, “UNIX Network Programming”, Prentice Hall, Englewood Cliffs, NJ, 1991.

[4] H.T. Kung and Alan Chapman, ”Automatic Quality of Service in IP Networks”

[5] RFCs: http://www.isi.edu/rfc-editor/status.html

[6] FreeBSD source code: http://minnie.cs.adfa.oz.au/FreeBSD-srctree/FreeBSD.html

[7] SunOS: http://cmgm.stanford.edu/man2html/index.html

[8] SSH (Secure Shell): http://www.cs.hut.fi/ssh/

[9] X11: ftp://crl.dec.com:/pub/DEC/xforward.tar.Z

[10] RealAudio homepage: http://www.realaudio.com/

[11] Televox homepage: http://www.voxware.com/

[12] GSM DirectAudio homepage: http://itre.ncsu.edu/gsm/

[13] VDOLife Player homepage: http://www.vdoguide.com/

[14] WinPlay3 homepage: http://www.iis.fhg.de/departs/amm/layer3/mmp/index.html

[15] DSP TrueSpeech homepage: http://www.dspg.com/

[16] Vocaltec Internet Wave: http://www.vocaltec.com/

[17] Xingtech StreamWorks: http://www.xingtech.com/

[18] IP Next Generation: http://playground.sun.com/pub/ipng/html/ipng-main.html

[19] Cisco NetFlow Switching: http://www.cisco.com/warp/public/733/packet/nfss_ov.htm