IEICE TRANSACTIONS on Communications

VOL. E98-B NO. 8
AUGUST 2015

A PUBLICATION OF THE COMMUNICATIONS SOCIETY
The Institute of Electronics, Information and Communication Engineers
Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

IEICE TRANS. COMMUN., VOL.E98–B, NO.8 AUGUST 2015

PAPER

Estimation of the Port Number Consumption of Web Browsing

Gábor LENCSE[†a)], ***Member***

SUMMARY Due to the depletion of the public IPv4 address pool, Internet service providers will not be able to supply their new customers with public IPv4 addresses in the near future. Either they give private IPv4 addresses and use carrier grade NAT (CGN) or they move towards IPv6 and provide NAT64 service to the IPv6 only clients who want to reach IPv4 only servers. In both cases they must use a stateful NAT/NAT64 solution. When dimensioning a NAT/NAT64 gateway, the port number consumption of the clients is a key factor as the port numbers are 16 bits long and a unique one has to be provided for every session (when using traditional type NAPT, which does not include the destination IP address and port number in the tuple for the identification of TCP sessions) and a single web client may use several hundred sessions and an equal number of port numbers according to literature. In this paper, we present a method for the estimation of the port number consumption of web browsing. The method is based on the port number consumption measurements of the most popular web sites and their combination using the number of the visitors of the web sites as weight factors. We propose the resulting curve as an approximation of a general profile of the average port number consumption of web browsers after the first click, but without taking into consideration the effect of the web users' browsing behavior. We also discuss the case of the extended NAPT, which can reuse the source port numbers towards different destination IP addresses and/or destination port numbers. We propose a formula and give measurement results for the extended NAPT gateways, too. We disclose the measurement method in detail and provide the measurement scripts in Linux, too.

key words: *IPv6 transition, NAPT, NAT64, port number consumption, port shortfall, web browsing*

1. Introduction

In the forthcoming years more and more internet service providers (ISP) will not be able to supply their new customers with public IPv4 addresses due to the depletion of the public IPv4 address pool. They have two main alternatives. Either they distribute private IPv4 addresses and use carrier grade NAT (CGN) or they move towards IPv6. The second option is better for the long run. However, the IPv6 only clients must somehow reach the IPv4 only servers, which are in majority today. In our opinion, the best available solution for this problem is the application of DNS64+NAT64 [1]. Thus both alternatives (CGN and IPv6 only clients) use stateful NAT. Stateful NAT replaces the source port numbers (besides the source IP addresses) with unique port numbers in order to identify the connections. This is why it is called NAPT: Network Address and Port Translation (see the details of its operation and the two implementation alternatives later). The TCP or UDP port numbers are 16 bits long so they have 2^{16} possible values, from which the ones from 1024 to 65535 are actually used for replacing the source port numbers. This seems to be a large enough range for a high number of clients but one client may require several hundreds of ports depending on the type of applications used and a port shortfall may result in improper operation [2]. Thus an ISP or a network operator should carefully determine the number of clients per NAT64 device (or more exactly per public IPv4 address assigned to its outbound interface). They would need statistical input for a well-founded design, but there are only few research papers dealing with the port number consumption of various applications ([2]–[5]).

From among the NAT64 compatible applications [6], HTTP and FTP are the most port consuming ones while the others generally use only one port [5]. In this paper, we focus on the port number consumption of HTTP only.

The aim of this paper is twofold: we would like to give numerical data that can be immediately used and also to disclose our measurement method that can be used by other researchers to validate our results. Therefore we give the details of our measurements and also include our scripts.

The remainder of this paper is organized as follows. First, the port number consumption problem is explained and discussed, including an overview of existing results and the reasons for their inapplicability as well as a basic overview of the two possible ways of the estimation of the port number consumption of web browsing. Second, our port number consumption estimation method (based on the port number consumption and market share of some of the most popular web sites) is detailed explaining the considerations of the method and the exact measurement scripts. Third, our results are presented and discussed. Finally, our conclusions are given.

We consider this topic to be important in dimensioning NAT/NAT64 gateways, which we expect to be a significant issue in the upcoming years of IPv6 transition.

2. Initial Discussion of the Problem

2.1 Port Number Consumption of Stateful NAT/NAT64

If the clients use private IPv4 addresses the NAT (more precisely NAPT: Network Address and Port Translation) [7] gateway makes it possible for them to communicate with the rest of the IPv4 world. If the clients use IPv6 addresses,

Manuscript received November 3, 2014.
Manuscript revised March 5, 2015.

[†]The author is with the Dept. of Telecomm., Széchenyi István University, 1. Egyetem tér, H-9026 Győr, Hungary.
a) E-mail: lencse@sze.hu

DOI: 10.1587/transcom.E98.B.1580

Table 1 Traditional NAPT translation table.

Source IP Address	Source Port Number	External IP Address	Temp. Port Number	Transport Protocol
10.1.2.2	5001	192.0.2.1	10001	TCP
10.1.2.3	5001	192.0.2.1	10002	TCP
10.1.3.5	5002	192.0.2.1	10003	TCP

Table 2 Extended NAPT translation table.

Source IP Address	Source Port Number	External IP Address	Temp. Port Number	Dest. IP Address	Dest. Port Number	Transp. Prot.
10.1.2.2	5001	192.0.2.1	5001	198.51.100.2	80	TCP
10.1.2.3	5001	192.0.2.1	5001	203.0.113.3	80	TCP
10.1.3.5	5001	192.0.2.1	5001	198.51.100.2	443	TCP
10.1.3.6	5001	192.0.2.1	10001	198.51.100.2	80	TCP

they need a NAT64 [8] gateway to be able to communicate with IPv4 only servers. In both cases the gateway sends out an IPv4 packet with its public IPv4 address as source address. Thus the reply packet(s) will be able to arrive back to the gateway. Then the gateway has to be able to forward the packets to the appropriate client using the correct IPv4 or IPv6 address and port number. To be able to do this, the stateful NAPT devices use a tuple for the identification of the session. Let us recall that a TCP/UDP session can be identified by a 5-tuple: source IP address, source port number, destination IP address, destination port number and the protocol number (expressing TCP or UDP). Traditional stateful NAPT implementations omit the destination IP address and destination port number and use only the entries in Table 1 for connection tracking [9]. They always replace the source port number with a unique one for each source IP address + port number pair. This algorithm has the consequence that the clients with private IP addresses behind the NAPT device may only have altogether 63k number of sessions per public IP address used at the external network interface of the gateway. Paper [9] suggested a solution for this problem in 2007. Their solution is to include the destination IP address and destination port number in the translation table and replace the source port number only if it is necessary — which means keeping the old one would result in a 5-tuple for the IP packet after the translation that is identical with an existing one in the translation table. We illustrated this in Table 2 (the 5-tuple in the rightmost 5 columns must be different for each session).

From here on we call the first method *traditional* and the second one *extended* NAPT implementation type. Which of the two types of implementation should we consider when thinking of the port number consumption of web browsing? Whereas traditional implementations with the 63k limit must undoubtedly exist, we should decide if there exists at least one extended NAT/NAT64 implementation. Therefore we checked the well-known Netfilter framework [10] of the Linux system (also called `iptables` from the name of its user interface program). Its source code was accessed as the part of the Linux kernel on GitHub [11]. We found the `get_unique_tuple()` function in the `/net/netfilter/nf_nat_core.c` file. It uses a `struct` of type `nf_conntrack_tuple`, which is defined in the `/include/net/netfilter/nf_conntrack_tuple.h` file and this `struct` contains the destination IP address and destination port number. Thus it can reuse the source port numbers when the clients access web servers with different IP addresses. This is also good news concerning NAT64 implementations because the TAYGA [12] stateless NAT64 solution is used together with `iptables` under Linux to provide stateful NAT64 service. We have already tested the performance and stability of the combination of TAYGA+`iptables` and we found it stable, but the other tested NAT64 implementation, namely the PF of OpenBSD seriously outperformed it, see our papers [13] and [14]. For this reason, we also checked the source code of PF [15] but the analysis of the more than 6700 lines long single file C code is beyond our time limits.

However, the port numbers are not unlimited even if we use an extended NAT/NAT64 implementation. Why? The HTTP protocol always uses TCP and the destination port number is either 80 (HTTP) or 443 (HTTPS). Thus practically, the destination IP address is the only factor of freedom. And though an IPv4 address is 32 bits long (thus it may take potentially 2^{32} possible values) the destination IP addresses of the web clients have a very much uneven distribution for at least two reasons:

1. There are some very popular web sites which may be visited by a significant proportion of the clients behind the NAT/NAT64 device.
2. Web hosting companies provide name based virtual hosting [16] to spare IP addresses. It means that they use the same IP addresses for a possibly high number of different web sites.

If it theoretically happens that a very popular destination IP address combined with a high port number consumption content exhausts all the possible external IP address plus temporary port number combinations at the gateway then its clients may suffer a port shortfall, but all those external IP address plus temporary port number combinations may be reused for other destination IP addresses. Therefore, if an extended type NAT/NAT64 gateway is used then it is enough to check the port consumption of the *most popular and port hungry* destination IP address, and there is no need to deal with all the others.

2.2 Related Work: What Exists and What is Needed?

Both papers [2] and [3] focus on the upper bound of the possible port number consumption of internet applications because the so-called A+P (Address plus Port) approach [17] requires that. This approach uses Stateless A+P Mapping (SMAP) and to support this, it divides the 64k port range to smaller, fixed sized ranges. Therefore the authors of papers [2] and [3] are interested in the upper bound of the port number consumption of internet applications. Both papers suggest that this upper bound is in the order of several hundreds.

The authors of [4] also deal with the upper bound of the port number consumption of a single web browser. They simulate how the TCP protocol stack reacts to a port shortfall situation and they find that it results in about a 3 second-long delay without any feedback. Finally, they recommend 500 ports per client.

As for a traditional NAT or NAT64 device, it does not need to pre-divide its port range (like A+P does) but it can dynamically allocate ports for each application. Thus its design needs something like the average port number consumption rather than the maximum port number consumption of the applications. Of course, since only the average as a single number may not be enough, one may also need some more detailed characterization of the port number consumption of internet applications (e.g. a curve as a function of time).

As for an extended NAT or NAT64 device, it can reuse the port numbers for web servers with different IP addresses, thus it is enough to find the *most popular and port hungry* destination IP address. We will refine this concept later on.

We addressed the port number consumption of HTTP and FTP in [5]. As for FTP, we showed that its port number consumption is somewhat higher than the number of the downloaded files and directory listings (as separate data connections are used for each of them) and it also depends on the type of the FTP client. As for HTTP, we showed that the port number consumption depends not only on the web sites (the URLs) but also on the web browsers and also on the operating system of the client.

From the many types of internet applications, now we will focus on web browsing only. Concerning the port number consumption of web browsing, the following statements can sum up the essence of the results above:

- The port number consumption of the different websites can be in the range of one to several hundreds.
- The port number consumption of the different websites must be measured carefully as they may depend on several factors such as the web browser and the operating system of the client.
- A "similar to average" but probably more detailed statistical characterization of the port consumption of web browsing is needed for the traditional type NAPT gateways.
- The port number consumption of the *most popular and port hungry* server (destination IP address) is needed for the extended type NAPT gateways.

Unfortunately, we could not find this kind of results in the literature.

2.3 What can be Measured?

On the one hand the methods we used in [5] are very promising as it is only question of time and computing power to measure the port number consumption of the top 500 [18] (or even much more) web sites of the world and the results can be combined as weighted average where the weight of a given web site is proportional to its market share. However, there are several problems with this approach:

1. A script based automated test like the one we used in [5] can only measure the port number consumption of the front pages of the sites efficiently. However the port number consumption of the other pages (especially of those that can be accessed only after logging in with user name and password) can be different.
2. Alexa [18] gives the up-to-date rank list of the sites, but without weighting factors (market share). The market share lists we could find on the web are either obsolete (e.g. [19] is from 2012 and [20] is from 2010) or refer to US sites only (e.g. [19]) or only provide lists within countries instead of global list (e.g. [21]) and all of them list only up to the top 100 sites or even less.
3. We showed in [5] that the port number consumption depends on the client's browser and operating system too.
4. There are different rank lists provided for the different countries (e.g. [18]) and also for mobile clients (e.g. [21]). But the port number consumption may be different in the network of a given network operator from that of the general one in the given country (e.g. in the case of a university, a corporate network, a mobile or a wired ISP).

Thus the results of this method are surely not accurate, and they can only be used as a hint concerning the order of magnitude of the port number consumption of web clients. However, this method is still considered to be useful as there is a 2-3 order or magnitude difference between the old naive 1 port per client and the 500 ports per client rules.

A different approach can be that measurements are taken on live networks (monitoring the traffic of real users). This solution can produce accurate results if the measurements are taken on the same network where the NAT/NAT64 devices are to be placed. The disadvantage of this approach is that if the results are published then their validity is rather limited to the given network (or to similar ones).

In this paper, we follow the first approach. We believe that our contribution can be the most useful if it is twofold: we publish both our results and our detailed measurement method including measurement scripts.

By following this approach, we can characterize the port number consumption of web browsing for the traditional type NAPT devices and also for the extended type ones. But for the latter, we can deal with the popular web sites problem only, but we cannot address the problem of the name based virtual hosting due to lack of data. We return to this subject at the end of this paper when discussing the results.

3. Measurement Methodology

3.1 General Considerations

The method described in [5] could now also be followed. Then we used eight operating system (OS) plus browser combinations and showed that these combinations give different results. But our results also showed that the port number consumption of the most port hungry OS + browser combination was not higher than two times the port number consumption of the less port consuming one. Taking into account that the problems listed in the previous chapter may cause larger difference than the factor of two, we used only one OS + browser combination choosing Debian Linux + Iceweasel (a Firefox clone).

The measurements in [5] lasted 60 seconds. This time was very likely long enough to download the web pages, but the port number consumption displayed on the graphs did not decrease to zero until the end of the 60 second-long interval. This could happen for two reasons: either the TCP connections were still alive or the TCP connections were already closed but the timeout has not expired yet in the NAT64 device and thus the connections were still present in its connection tracking state table.

As different NAT/NAT64 devices may have different (default) timeout settings we are interested in the time while the TCP connections are open. (The extra time for the port usage can be considered later on the basis of the timeout values of the given devices.) Note that the time while the TCP connections are open does not depend on the NAT/NAT64 device but on the client and the server only, thus we do not have to use NAT/NAT64 for the measurements.

The aim of our first series of measurements was to determine the length of the measurement interval that is large enough to see the length of all the TCP connections, while we do not waste execution time.

3.2 Measurement Environment and Parameters

An average home desktop computer was chosen with the following parameters: AMD Athlon 64 X2 Dual Core 4200+ 2200 MHz CPU, 2 GB DDR2 667 MHz RAM, 320 GB HDD, Internet access through a Linksys E3000 router using cable net with 60 Mbps download and 6 Mbps upload speeds. Debian 7.6 operating system with KDE was installed on the computer.

The version of Iceweasel was: 24.6.0. Iceweasel was set to start with an empty page (i.e. not try to load a home page or restore the previous page(s) that were open last time). The caching was switched off (by limiting cache to 0MB of space) but it was observed that the disk usage of the `~/.cache/mozilla/firefox` directory was still growing thus the contents of this directory were deleted in every single measurement right before opening the web pages.

Flash player from the non-free repository was also installed manually so that the browser may load flash portions of the pages. (The pop-up windows were left blocked as this was the default setting of the browser.)

All the other settings of the browser were left to have their default values. The port number related settings were checked by opening the `about:config` URL. They can be found after the `network.http.` prefix. Their actual values were also their default ones:

```
max-connections 256
max-persistent-connections-per-proxy 32
max-persistent-connections-per-server 6
```

The first one is the most important for us, as this one can limit the peak port number consumption rate of the browser.

The `netstat` Linux command was chosen for monitoring the open connections. As it has been mentioned before, we were interested in the time of the open TCP connections thus Linux kernel version 3.12.6 was compiled by gcc 4.7.2-5 in order to be able to decrease the time out value of the TCP protocol stack. The value of the TCP_TIMEWAIT_LEN parameter defined in `include/net/tcp.h` was decreased from 60 seconds to 1 second.

The list of the global 1 million top sites was downloaded from [18]. The date of the file was July 18, 2014. (Some other parameters concerning the estimation of the wait factors of the web sites will be provided later.)

3.3 Measurement Scripts

The main measurement script is called `measure`. It takes the URLs to be tested from the file named `sites` and calls an external script named `ns` to count and log the number of active TCP connections. The `measure` script opens the URLs one by one and calls the `ns` script in every second. The measure script contains two parameters at the beginning of the file. They are used in the following way: `Seconds` defines the length of the measurement interval in seconds. `Iterations` defines how many times every URL is tested. The script supplies the URL as a positional parameter to the `ns` script.

```
#!/bin/bash
# Main script for port number consumption measurements.
# It MUST be executed as root!
Seconds=600 # length of one test in seconds
Iterations=11 # perform each test so many times
killall -9 iceweasel 2>/dev/null # might be running
rm -rf results # delete earlier results
```

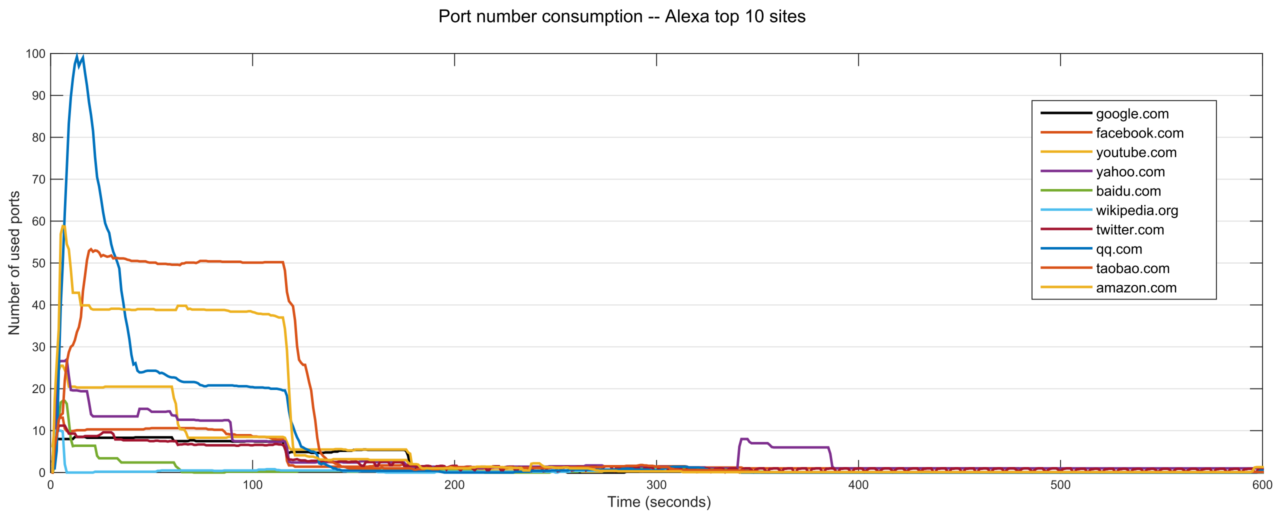

Fig. 1 Port number usage of the Alexa top 10 sites as a function of time.

```
mkdir results  # create the directory
for URL in $(cat sites) # test all the URLs
do
    for i in {1..$Iterations}
    do
        rm -rf ~/.cache/mozilla/firefox/* # iceweasel
        iceweasel "http://$URL" & # open the URL
        for t in {1..$Seconds}
        do
            sleep 1 # wait 1 second
            ./ns $URL & # count and log
        done
        sleep 1 # ./ns must finish
        echo "" >> results/$URL # line feed
        killall -9 iceweasel 2>/dev/null
        sleep 3 # let the browser stop completely
    done
done
```

The `ns` script takes the URL as a positional parameter: `$1`. It counts the TCP connections in the ESTABLISHED state for the `iceweasel` browser and logs their number in `results/URL`. Note that the `-p` option of `netstat` is used to display the PID and the name of the program that the TCP connection belongs to. To be able to access this information, the `ns` script and thus the `measure` script must be executed as root.

```
#!/bin/bash
# It is called by the main script, run it as root!
# Counts open TCP connections for $1 and logs them
PORTS=$(netstat -tpn|grep "ESTABLISHED.*$1"|wc -l)
echo -n "$PORTS " >> results/$1
```

3.4 Determining the Length of the Measurement Interval

Preliminary measurements were performed in order to determine the length of the measurement interval. The global top 10 sites were selected from [18]. After some tests, the measurement interval of 600 seconds for the preliminary measurements was chosen to be large enough and all these sites were tested 11 times to produce reliable results. The results of the 11 measurements were averaged as follows (M denotes the number of the measurement):

$$Ports(URL, t) = \frac{1}{11} \sum_{M=1}^{11} Ports(URL, t, M) \qquad (1)$$

Figure 1 shows the port number consumption of the 10 tested sites as a function of time. It can be seen that 1 minute is way too short for the measurement interval, but the port number consumption drops at about two minutes (or sometimes earlier) and after 3 minutes there is only a very small number of ports in use (in general). Thus the length of the measurement interval was chosen to be 200 seconds to leave some guard interval.

3.5 Determining the Weight Factors

From the sources [19]–[21] only [21] is up to date. Unfortunately, this one is not really good for our purposes either, for the following reasons:

1. It gives "monthly people" and not "monthly visits". For ranking the sites, it is a perfect measure, but for estimating the weight factor of a site in the global web traffic, one would need the "monthly visits" value. (E.g. the spreadsheet downloadable from [20] contains two fields "unique audience" and "total visits" but being more than 4 years old, it is way too obsolete.)
2. It is a country-wise top site list, not a global one. In practice, this is an advantage for a network administrator from a given country, but it may be a disadvantage when writing an international paper.
3. Twelve of the US top 100 websites are denoted as "Hidden profile" (giving no data).

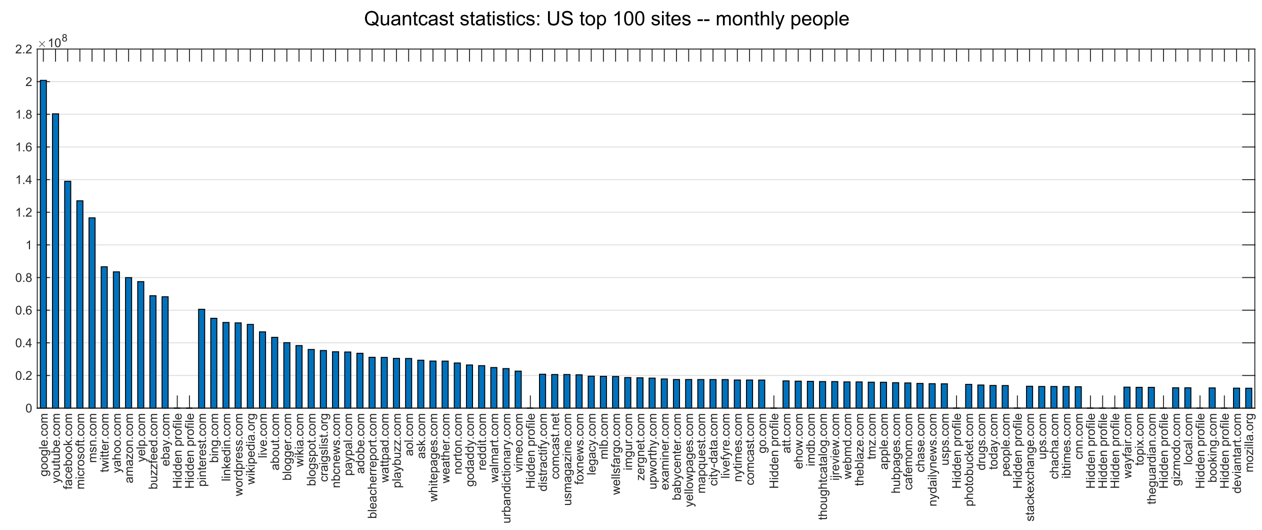

Fig. 2 Number of visitors of the US top 100 websites — based on the data downloaded from [21] on July 22, 2014.

To be able to prepare at least some rough estimation, we handle these problems in the following way:

- Having no better hints, we estimate the proportion of the total visits of the sites with the proportion of their unique visitors.
- We use the data for the US top 100 websites. (It could be any other country but the US profile was hopefully measured precisely enough.)
- We simply leave out the missing 12 websites. (As this US top website list differs from that of Alexa, there is no ground for guessing the URLs even if the number of their visitors could be approximated by linear interpolation from that of their neighbors.)

As a consequence, instead of the most well-known and complete Alexa list, the first 88 existing URLs from [21] (downloaded the US list July 22, 2014) were used. Figure 2 shows the number of visitors. The envelope of the columns reminds us of an exponential curve.

The weight factors were calculated as follows:

$$Weight(i) = \frac{Visitors(i)}{\sum_{j=1}^{88} Visitors(j)} \tag{2}$$

Where the sites are indexed from 1 to 88 (leaving out the ones with a hidden profile).

3.6 Quantities for Characterization

The traditional type NAPT devices are sensitive to the average port number consumption of web browsing, thus the weighted sum of the port number consumption of the 88 public profile sites was calculated as follows:

$$Ports(t) = \sum_{i=1}^{88} Weight(i) Ports(i,t) \tag{3}$$

Where $Ports(i,t)$ denotes the result of our measurements and $Ports(t)$ denotes our estimated port number consumption profile (see below) that aims to approximate the average port number consumption of an arbitrary web browsing session as a function of time.

As for the extended type NAPT devices, they are sensitive to the maximum port number consumption per destination IP address. For them, the port number consumption of web browsing can be characterized as follows:

$$Ports(t) = \max_{i=1}^{88} (Weight(i) Ports(i,t)) \tag{4}$$

4. Measurement Results and Their Discussion

The measurements were performed with the 88 public URLs from the Quantcast US top 100 list [21] using the above detailed `measure` script. The length of the measurement time was 200 seconds and all of the URLs were downloaded 11 times and the results were averaged according to Eq. (1) to produce reliable results.

The weighted sum of the measured port number consumption values of the 88 sites was calculated according to Eq. (3) and it is shown in Fig. 3. As expected, the curve rises steeply and its fall at about two minutes is also definite. Two other less definite falls can be observed around one minute and three minutes. It is rather tempting to state that 50 ports are enough instead of 500, and they are used for about 2 minutes, but we believe that more investigation is needed before setting up this kind of rule of thumb.

The maximum of the weighted port number consumption curves of the 88 sites was calculated according to Eq. (4) and it is shown in Fig. 4. The port number consumption is always less than 3 due to the small values of the weight factors (they start from 0.066873 for `google.com` at the first place and end at 0.004063 for `mozilla.org` at the last place of the Quantcast popularity list).

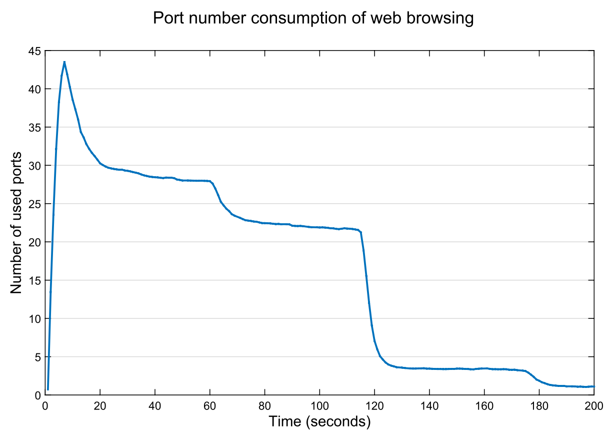

Fig. 3 Estimated port number consumption profile for **traditional type NAPT devices** — based on the *weighted sum* of the port number consumption of the 88 public profile sites from among the Quantcast US top 100 sites.

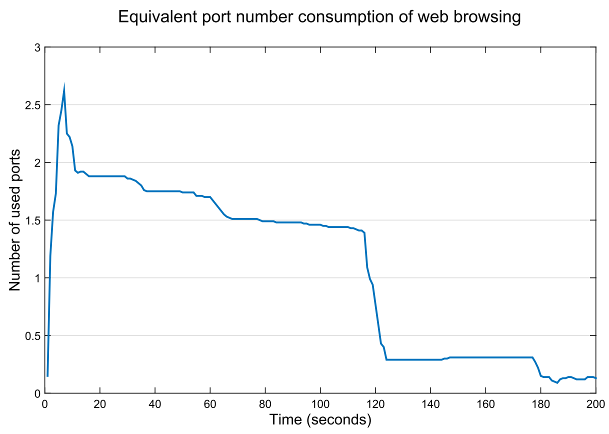

Fig. 4 Estimated port number consumption profile for **extended type NAPT devices** — based on the *weighted maximum* of the port number consumption of the 88 public profile sites from among the Quantcast US top 100 sites.

Now, let us come back to the problem of the name based virtual hosting. We cannot prove the following considerations, but it is likely that:

- The sites listed among the Quantcast top 100 websites are hosted on their own and do not share IP addresses with other sites.
- The virtual hosted sites sharing a common IP address have different port number consumption (some high and some low).
- None of the shared IP addresses for web sites in the USA has higher traffic than 1% of the total USA web traffic.

Thus it is very likely that they do not have the chance to influence the maximum value calculated according to Eq. (4), due to their relatively small weight factor and not expressly high port number consumption.

We emphasize that all our results are *order of magnitude estimations* only and we encourage other researchers to repeat our measurements with other web browsers and different and possibly better top site lists.

The profiling of the different websites and preparing a weighted average (or maximum) of their port number consumption is a kind of synthetic approach. The produced profiles in Fig. 3 and in Fig. 4 cannot be directly used for dimensioning the NAT/NAT64 gateways, but they need to be combined with the web users' browsing behavior: for example, how often they click on a link within the active web page or open a new page by typing in its URL. This is a popular research area, with both novel and matured publications, such as [22]–[24], to mention only a few. But the discussion of this topic is beyond the scope of our paper.

4.1 How Does Caching Influence the Port Number Consumption?

Three sites with a high port number consumption were selected from the Alexa top 10 sites list. Their port number consumption was tested with and without caching. In both cases, the length of the measurement time was 200 seconds and all of the URLs were downloaded 11 times and the results were averaged according to Eq. (1) to produce reliable results. When caching was on, an initial downloading was done to fill the cache and the contents of the `/.cache/mozilla/firefox` directory were not deleted by the `measure` script. The cache size was set to 100 MB in the browser, in order to be large enough. The results are shown in Fig. 5. For `amazon.com`, the port number consumption results of the experiments with and without caching are so close to each other that the red line nearly completely hides the blue one. The situation is very similar for `taobao.com`, too. For `qq.com`, there is a visible difference: the maximum value of the port number consumption without caching is 103.9 and it is only 94.1 with caching. However, this difference is less than 10%. Thus our order of magnitude estimations are not influenced significantly by caching.

4.2 Limitations on the Number of Connections

The number of the TCP connections between a client and a server can be limited by several factors:

- the settings both in the client and in the server[†]
- the administrative settings of certain network devices (e.g. firewalls, stateful NAPT devices) along the path between the client and the server
- the nature of the content of the web site: how many objects can be downloaded concurrently.

In our measurements, the settings of the client allowed maximum 256 connections, which was far higher than we experienced. The client computer was connected to an Ethernet port of a Linksys E3000 router which had a public IP address from the ISP. We did not set any limitation on the

[†]However, they may have a different effect on the port number consumption, see the next subsection.

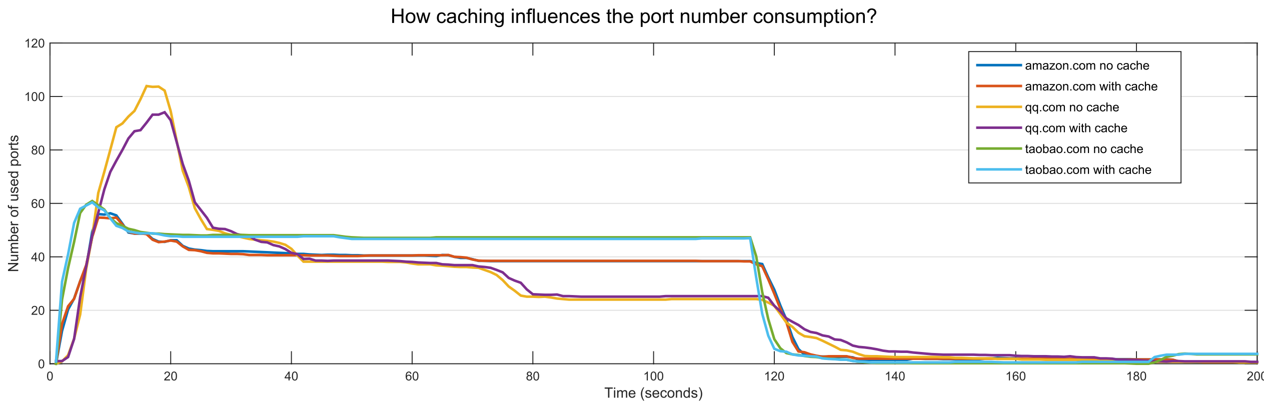

Fig. 5 The effect of the absence or presence of caching to the port number consumption.

client side and we do not know of any restrictions at our ISP. Therefore we believe our measurement results reflect the true behavior of the aforementioned internet sites. Under their 'behavior' we mean both their settings and their contents.

4.3 TCP Connections Rejected by the Server

If a client reaches its own limit of parallel sessions it will not try to open more TCP connections. However, if the limit is reached in the server, the client still tries to open more TCP connections and sends TCP SYN segments, which consume ports in the NAT/NAT64 device, but finally do not result in established TCP connections. As we counted the number of TCP connections in the ESTABLISHED state, the actual value of the port number consumption in a NAT/NAT64 device may be higher than we measured.

4.4 TCP Performance

During our measurements, the client computer was located in Hungary, Europe and the web servers were located in various continents all over the world. Our experiments lasted several days long: they were performed one after another during days and nights. Both the distance between the client and the server and the actual load of the network between them influenced the round trip delay between the clients and the server. The TCP performance could also affect the port number consumption.

4.5 Final Discussion

Several other factors may also influence the port number consumption of web browsing. Therefore we emphasize once again, that our results are only order of magnitude estimations and they should be repeated at some other part of the world. We encourage other researchers to do so and will be glad to hear about the results.

5. Conclusions

We explained and discussed the problem of the port number consumption of web browsing including the overview of the existing results and the reasons for their inapplicability for dimensioning NAT/NAT64 gateways.

We identified two types of NAPT behaviors: the *traditional* type does not use the destination IP address and port number for identifying TCP sessions and thus is limited to 63k source port numbers per external public IP address. The *extended* type NAPT devices include the destination IP address and the port number for the identification of TCP sessions and thus the 63k source port number limit per external public IP address is to be understood to be *per destination address*.

We have developed different formulas to assist the dimensioning of the two types of NAPT devices.

As for the traditional type NAPT devices, we suggested a method for the estimation of the average port number consumption of web browsing. The method is based on the weighted sum of the port number consumption of the 88 public profile websites from the Quantcast US top 100 sites list. Our proposed estimated port number consumption profile gives the average port number consumption of web browsing as a function of time and should be used together with a web users' browsing behavior model for dimensioning NAT/NAT64 gateways.

As for the extended type NAPT devices, we proposed the weighted maximum of the port number consumption of the most popular web sites.

By disclosing our measurement methods in detail including the most important scripts, we made it easy to validate our results using more complete top sites lists.

Acknowledgments

Author would like to thank Ágnes Varga and Ákos Kovács for English-language proofreading of the paper and for re-editing the figures in MATLAB, respectively.

References

[1] M. Bagnulo, A. Garcia-Martinez, and I. Van Beijnum, "The NAT64/DNS64 tool suite for IPv6 transition," IEEE Commun. Mag., vol.50, no.7, pp.177–183, July 2012.

[2] S. Miyakawa, "IPv4 to IPv6 transformation schemes", IEICE Trans. Commun., vol.E93-B, no.5, pp.1078–1084, May 2010. DOI:10.1587/transcom.E93.B.1078
[3] F. Fourcot, and B. Grelot, "Migrating to IPv6 with Address+Port translation", SLR Project Report, Telecom Bretagne, March 2010, https://svn.fperrin.net/v6fication/documentation/rapport-Fourcot-Grelot.pdf
[4] I. Kraemer, and F. Perrin, "Impact of the lack of ports in a DS-Lite architecture", Project Report, Telecom Bretagne, March 18, 2011, https://svn.fperrin.net/v6fication/article/simulation.pdf
[5] S. Répás, T. Hajas, and G. Lencse, "Port number consumption of the NAT64 IPv6 transition technology", Proc. 37th Internat. Conf. on Telecommunications and Signal Processing (TSP 2014), pp.66–72, Berlin, Germany, July 2014.
[6] S. Répás, T. Hajas, and G. Lencse, "Application compatibility of the NAT64 IPv6 transition technology", Proc. 37th Internat. Conf. on Telecommunications and Signal Processing (TSP 2014), pp.49–55, Berlin, Germany, July 2014.
[7] P. Srisuresh and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, August 1999.
[8] M. Bagnulo, P. Matthews, and I. Beijnum, "Stateful NAT64: Network address and protocol translation from IPv6 clients to IPv4 servers", RFC 6146, April 2011.
[9] M.S. Ferdous, F. Chowdhury, and J.C. Acharjee, "An extended algorithm to enhance the performance of the current NAPT", Proc. Internat. Conf. on Information and Communication Technology 2007 (ICICT'07), pp.315–318, Dhaka, Bangladesh, March 2007. DOI:10.1109/ICICT.2007.375401
[10] M. Boye, "Netfilter connection tracking and NAT implementation", Proc. Seminar on Network Protocols in Operating Systems, Dept. Commun. and Networking, Aalto Univ., pp.34–39, 2013. http://urn.fi/URN:ISBN:978-952-60-4997-7
[11] Linux Torvalds, "Linux kernel source tree", https://github.com/ torvalds/linux/, accessed Oct. 30. 2014.
[12] TAYGA: Simple, no-fuss NAT64 for Linux, http://www.litech.org/ tayga/, accessed Oct. 30. 2014.
[13] G. Lencse and S. Répás, "Performance analysis and comparison of the TAYGA and of the PF NAT64 implementations", Proc. 36th Internat. Conf. Telecomm. and Signal Processing (TSP 2013), pp.71–76, Rome, Italy, 2013. DOI:10.1109/TSP.2013.6613894
[14] S. Répás, P. Farnadi, and G. Lencse, "Performance and stability analysis of free NAT64 implementations with different protocols", Acta Technica Jaurinensis, vol.7, no.4, pp.404–427, Oct. 2014. DOI:10.14513/actatechjaur.v7.n4.340
[15] http://cvsweb.openbsd.org/cgi-bin/cvsweb/src/sys/net/pf.c, accessed Oct. 30. 2014.
[16] Virtual hosting, http://en.wikipedia.org/wiki/Virtual_hosting, accessed Oct. 30. 2014.
[17] R. Bush, Ed., "The address plus port (A+P) approach to the IPv4 address shortage", RFC 6346, Aug. 2011.
[18] "The top 500 sites on the web", http://www.alexa.com/topsites, accessed July 22, 2014.
[19] Statistic Brain, "Top websites by traffic", http://www.statisticbrain.com/top-us-websites-by-traffic/, accessed July 22, 2014.
[20] BBC News, "SuperPower: Visualising the Internet", http://news.bbc.co.uk/2/hi/technology/8562801.stm, accessed July 22, 2014.
[21] Quantcast, "Top sites", Quantcast Corporation, https://www.quantcast.com/top-sites, accessed July 22, 2014.
[22] P.E. Román, J.D. Velsquez, V. Palade, and L.C. Jain, "New trends in web user behaviour analysis" in Advanced Techniques in Web Intelligence-2: Web User Browsing Behaviour and Preference Analysis, pp.1–10, Springer Berlin Heidelberg, 2013. DOI:10.1007/978-3-642-33326-2_1
[23] C. Liu, R.W. White, and S. Dumais, "Understanding web browsing behaviors through Weibull analysis of dwell time", Proc. 33rd Internat. ACM SIGIR Conf. on Research and Development in Information Retrieval, Geneva, Switzerland, pp.379–386, July 2010. DOI:10.1145/1835449.1835513
[24] L.D. Catledgea, and J.E. Pitkowb, "Characterizing browsing strategies in the world-wide web", Computer Networks and ISDN Systems, vol.27, no.6, pp.1065–1073, April, 1995. DOI:10.1016/0169-7552(95)00043-7

Gábor Lencse received his M.Sc. and Ph.D. degrees in computer science from the Budapest University of Technology and Economics, Budapest, Hungary in 1994 and 2001, respectively. He works for the Department of Telecommunications, Széchenyi István University, Győr, Hungary since 1997. Now, he is an Associate Professor. He is also a part time Senior Research Fellow at the Department of Networked Systems and Services, Budapest University of Technology and Economics since 2005. His research interests include the performance analysis of communication systems, parallel discrete event simulation methodology and IPv6 transition methods.