Athens Institute for Education and Research
ATINER

ATINER's Conference Paper Series
COM2013-0528

Automatic IP-Address Configuration Converting For Public Network

Phongphan Danphitsanuphan
Department of Computer and Information Science
King Mongkut's University of Technology North Bangkok, Thailand

Pathompong Hirunpong
Department of Computer and Information Science
King Mongkut's University of Technology North Bangkok, Thailand

Athens Institute for Education and Research
8 Valaoritou Street, Kolonaki, 10671 Athens, Greece
Tel: + 30 210 3634210 Fax: + 30 210 3634209
Email: info@atiner.gr URL: www.atiner.gr
URL Conference Papers Series: www.atiner.gr/papers.htm

Printed in Athens, Greece by the Athens Institute for Education and Research.

ISSN **2241-2891**
16/09/2013

An Introduction to ATINER's Conference Paper Series

ATINER started to publish this conference papers series in 2012. It includes only the papers submitted for publication after they were presented at one of the conferences organized by our Institute every year. The papers published in the series have not been refereed and are published as they were submitted by the author. The series serves two purposes. First, we want to disseminate the information as fast as possible. Second, by doing so, the authors can receive comments useful to revise their papers before they are considered for publication in one of ATINER's books, following our standard procedures of a blind review.

Dr. Gregory T. Papanikos
President
Athens Institute for Education and Research

This paper should be cited as follows:

Danphitsanuphan, P. and Hirunpong, P. (2013) "**Automatic IP-Address Configuration Converting For Public Network**" Athens: ATINER'S Conference Paper Series, No: COM2013-0528.

Automatic IP-Address Configuration Converting For Public Network

Phongphan Danphitsanuphan
Department of Computer and Information Science
King Mongkut's University of Technology North Bangkok
Thailand

Pathompong Hirunpong
Department of Computer and Information Science
King Mongkut's University of Technology North Bangkok
Thailand

Abstract

to be able to connect to the public WIFI networks, automatic DHCP function is needed to be enabled. This is not applicable to devices, which users do not have authorization to make change their network configuration, such as office notebook or fixed IP-address devices. This paper introduces Zero-configuration software module to cope with the problem at the network gateway by editing Ethernet package header to the required IP address parameters of a particular public network both of incoming and outgoing packages. The Experimental results show that the Zero configuration module can convert Ethernet package with above 99% of throughput in various testing scenarios. It can take up to 7,100 packages per second (about 80 Mbps) and its efficiency is 96.27% compared against normal data transfer (Zero-configuration module disabled) through the network gateway. Moreover, It can handle all well-known ports and services including VPN, etc.

Keywords: zero configuration, IP-address conversion, IP-address blinding

Corresponding Author:

Introduction

At present, WIFI Internet is widely used for hotels or any public areas. Many users need to set their network configurations in order to connect to the public network environment by mostly enabling automatic DHCP service. When they bring their computer back to home or office, network configuration is needed to be set back to previous setting. Also, some users do not have authorization to change network setting.

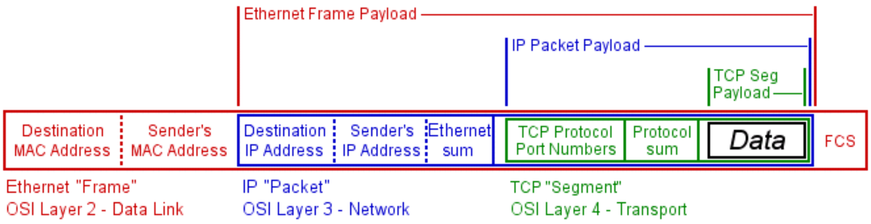
This paper presents Zero-configuration (ZC) software module to help computer to connect to the Internet regardless of which IP-Address, DNS, PORXY and Gateway address which were formerly set up in that computer. As a result, when a computer connected to the public Internet service, which equipped with the ZC module, client can connect to the Internet without any changes required.

Current technology can do such thing by using several brands of hardware boxes, but such hardware boxes are quite expensive for the function we discussed earlier so we develop software instead. It reduces cost and can be adjusted to make it more suitable for widely uses.

Ethernet Frame package

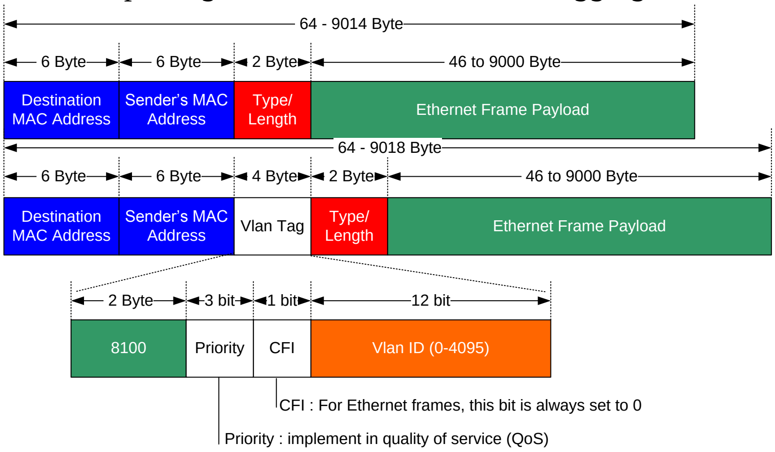
Ethernet Frame as in figure 1 compounds of Destination MAC Address, Sender's MAC Address, Destination IP Address, Sender's IP Address, Ethernet sum and Protocol sum

Figure 1. *Ethernet frame package*



In the complex network environment, VLAN [8] is used to isolated groups of network and Ethernet frame has an extra "Tag Vlan" slot as presented in figure 2.

Figure 2. *Ethernet package with and without VLAN tagging*

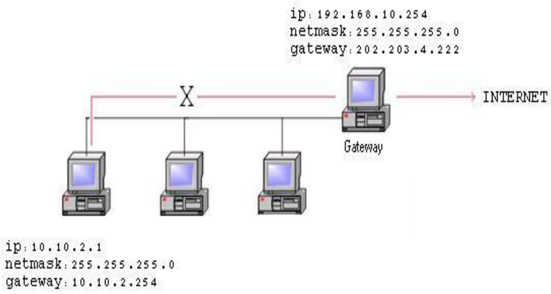


Ethernet package conversion design

This experiment used Linux Cent OS 5.2 as a network gateway server which has 2 Ethernet ports (ETH0, ETH1). Its functions were to allocate private IP address to the clients and to route all client's Ethernet packages to the Internet.

- *Normal scenario:* Clients (DHCP enable) physically/wirelessly connects to ETH0 (LAN side) and get assigned IP-address from the gateway server. Then, Clients can connect to the Internet through ETH1 (WAN side). Unrecognizably fixed IP-address client cannot access to the Internet as depicted in figure 3

Figure 3. *Unrecognizably fixed IP-address client cannot to the Internet through the Gateway server*



- *ZC module enabled scenario:* Clients (DHCP enable or unrecognizably fixed IP address) can connect to the Internet. In order to convert Ethernet package header to required format of the network gateway, virtual interface and ZC software module are added to gateway server as depicted in figure 4. Following configurations were used in the experiment.

Ethernet interface configuration

ETH0 (Local IP – LAN interface)

Link encap:Ethernet HWaddr 00:22:2D:C1:98:22

inet addr:192.168.200.248 Bcast:192.168.200.255 Mask:255.255.255.0

ETH1 (Public IP – WAN interface)

Link encap:Ethernet HWaddr 1C:6F:65:92:71:F5

inet addr:192.168.10.248 Bcast:192.168.10.255 Mask:255.255.255.0

Virtual interface

Link encap:Ethernet HWaddr BE:AD:56:E6:9A:59

inet addr:10.0.0.1 Bcast:10.0.0.255 Mask:255.255.255.0

**** Virtual interface's Hwaddr (mac address) is added to ARP table []*

Client device interface

MAC Address: 00-16-D3-4C-F9-CB

IP Address: 192.168.20.233

Subnet Mask: 255.255.255.0

Default Gateway: 192.168.20.254

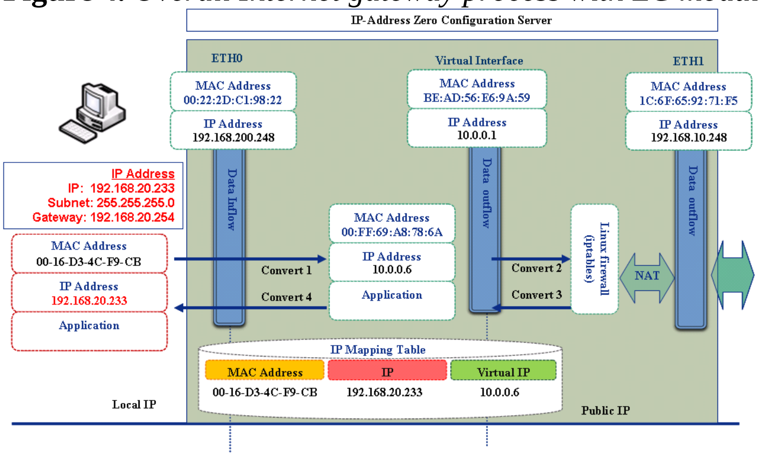
DNS Servers: 8.8.8.8

Ethernet package conversion by ZC module

Out-going package:

- "Convert 1", in figure 4, changes client's Mac-address and IP-address to virtual interface's recognized IP-format which assigned by ZC module and record values in IP Mapping table.
- "Convert 2" changes Virtual interface's Mac-address and IP-address to ETH1 interface's recognized format by using standard Linux firewall mechanism.

Figure 4. *Overall Internet gateway process with ZC module*



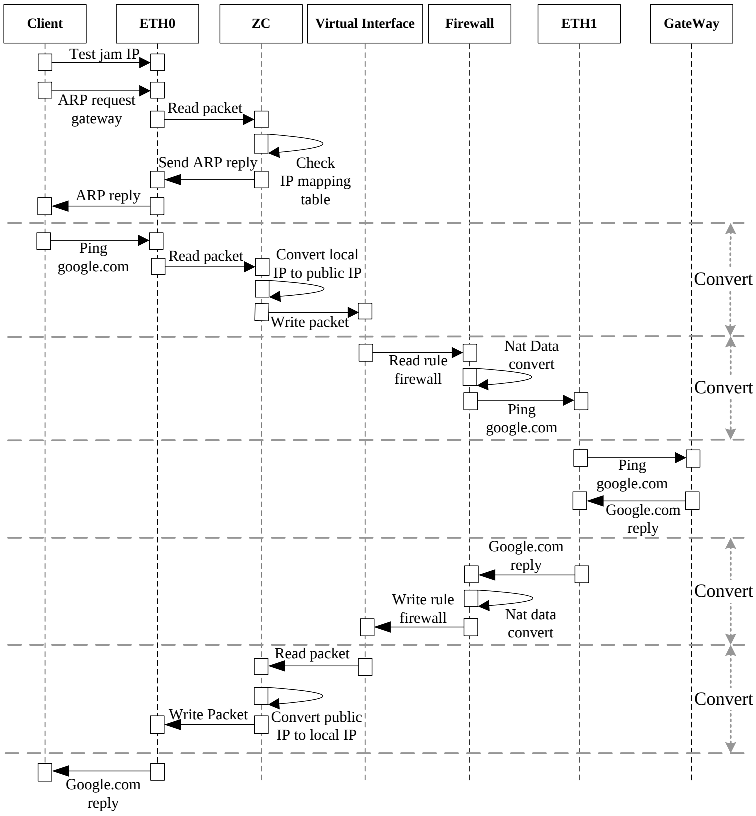
In-coming package:

- "Convert 3" changes ETH1 interface's Mac-address and IP-address to Virtual interface's recognized format by using standard Linux firewall mechanism.

- "Convert 4" changes Virtual interface's Mac-address and IP-address to Client's IP-address and Mac's address which recorded in IP mapping table.

IP mapping table is created and maintain by zero configuration module to record and match between real client's IP address and assigned Virtual IP address using in "Convert 1 and 4".

Figure 5. *Sequence diagram ZC working step when unknown client connect to testing network and send Ping command to www.google.com*

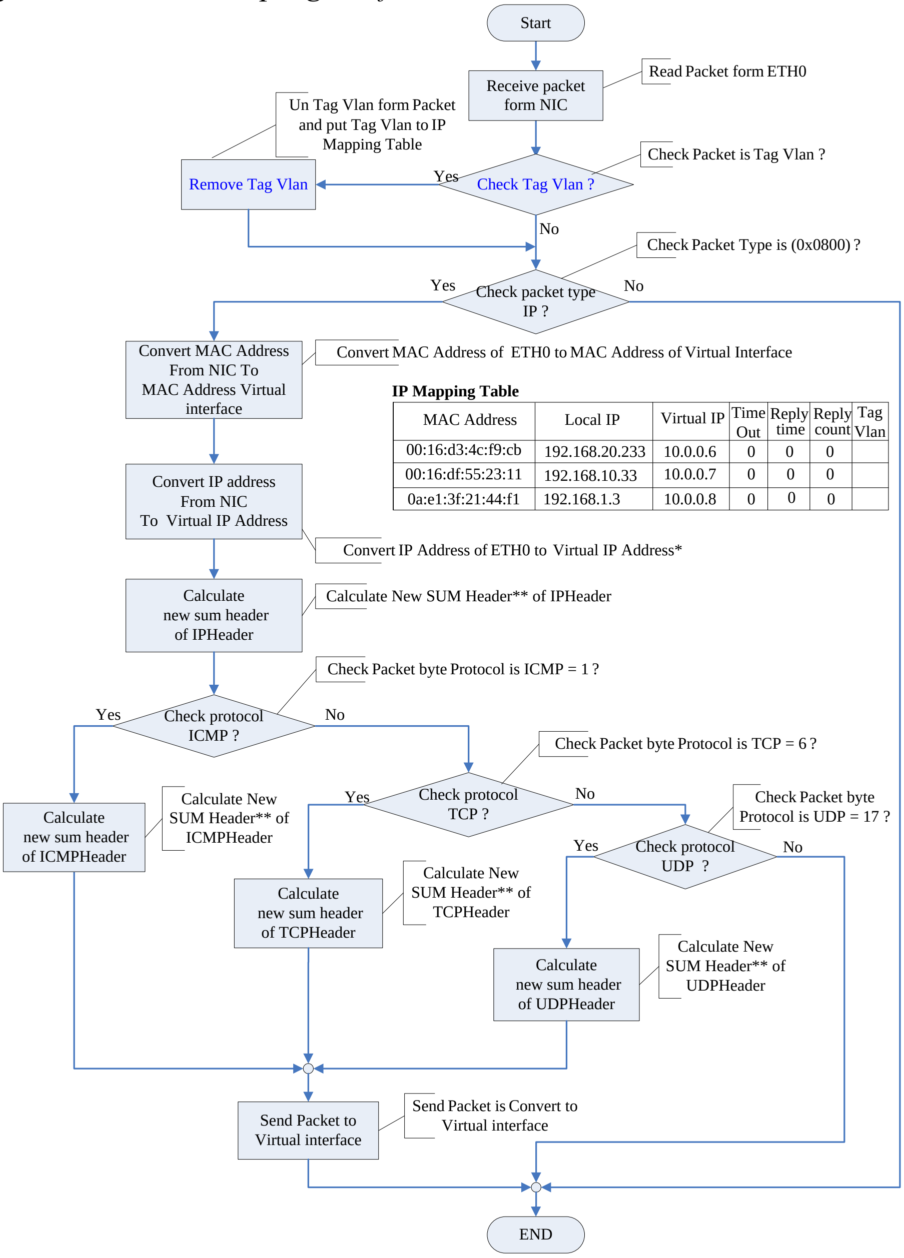


Zero-configuration software module

In figure 6, ZC module does "convert 1" and "convert 4" and maintain IP-mapping table (solely used by ZC).

First, ZC module detects if the package contains VLAN tagging. Second, ZC checks if it is Internet Protocol version 4 (IPv4) since ZC cannot works with IPv6. Third, package's original IP-address and MAC address are converted to virtual IP-address and MAC address and they are stored in IP-Mapping table as a record. Fourth, ZC calculates and imprints new check sum to the package regarding to its protocol (ICMP/TCP/UDP). Finally, edited package is sent to Virtual interface and to ETH1 respectively.

Figure 6. *"Convert 1" program flow chart*

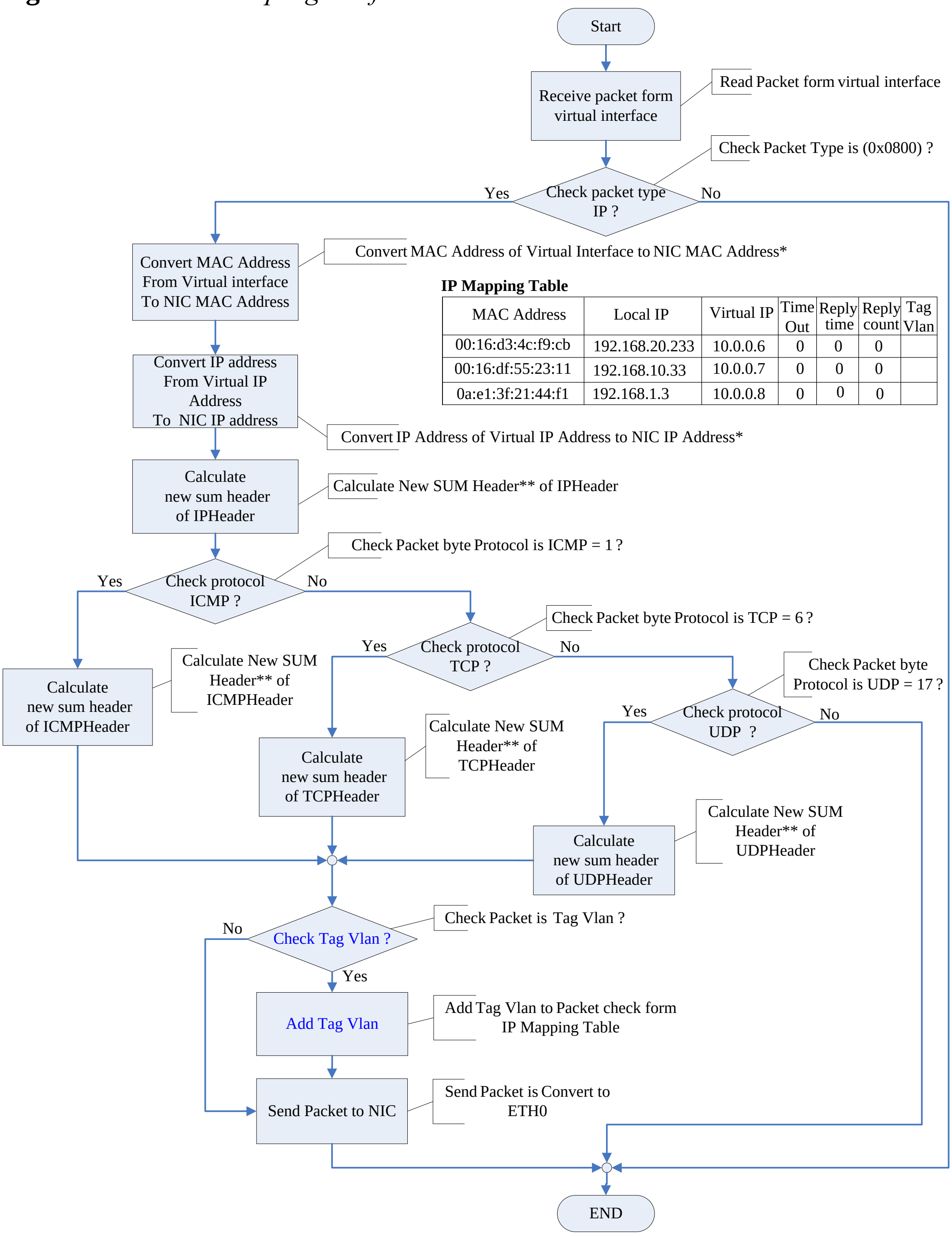


IP Mapping Table

MAC Address	Local IP	Virtual IP	Time Out	Reply time	Reply count	Tag Vlan
00:16:d3:4c:f9:cb	192.168.20.233	10.0.0.6	0	0	0	
00:16:df:55:23:11	192.168.10.33	10.0.0.7	0	0	0	
0a:e1:3f:21:44:f1	192.168.1.3	10.0.0.8	0	0	0	

Once ETH 1 receives reply data package from the Internet. It sends the package to the Virtual interface and ZC module start process "Convert 4" (reversed steps of "Convert 1") as depicted in Figure 7. ZC module extracts Virtual IP address and MAC address and searches them in IP Mapping table. Then convert data package's IP and MAC addresses back to the original. Finally, ETH0 is able to send the package back to the client.

Figure 7. *"Convert 4" program flow chart*



Experiment

Following controlled environment was built to capture the ZC's performance, throughput and reliability by comparing clients who connected to the IP-Address Zero Configuration server, with the clients who directly connected to the Internet.

Internet gateway server specification (with ZC module)
CPU: AMD Phenom II X6 3.2GHz
Motherboard: GIGABYTE GA-890GPA-UD3H (rev. 2.0)
RAM: 12GigaByte
Hard disk: 160GigaByte
Ethernet card: 1Gigabit x 2
OS: CentOS 6.2 (linux)

Performance testing:

Figure 9. Local FTP downloading and uploading testing configuration
Scenario 1: Local FTP downloading and uploading

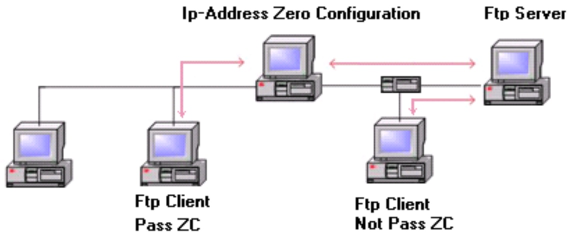


Table 1. *Local FTP downloading result*

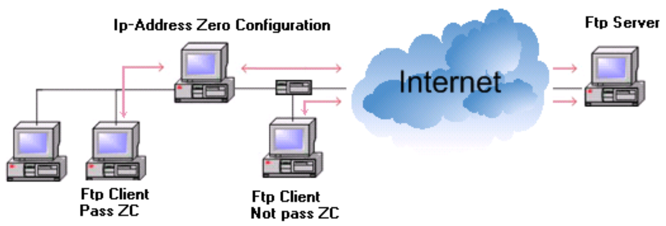
Download data (Byte)	Client who not pass ZC Server		Client who pass ZC Server		%Throughput (Pass ZC/ Non pass ZC)
	Time (Sec)	Speed (Byte/Sec)	Time (Sec)	Speed (Byte/Sec)	
1,024	0.004	256,000	0.006	170,667	50.00%
102,400	0.014	7,314,286	0.016	6,400,000	85.71%
524288	0.045	11,650,844	0.05	10,485,760	88.88%
1,048,576	0.1	10,485,760	0.1	10,485,760	100.00%
104,857,600	8.854	11,842,964	8.856	11,840,289	99.97%
536,870,912	45.32	11,846,225	45.32	11,846,225	100.00%
1,073,741,824	90.62	11,848,839	90.74	11,833,170	99.86%

The table 1 and 2 show downloading and uploading results comparing between pass and not-pass ZC server. ZC achieves better performance at those bigger among of data transfer transaction. Some ZC's process overhead obviously affects transactions which have smaller among data transfer.

Table 2. *Local FTP uploading result*

Download data (Byte)	Client who not pass ZC Server		Client who pass ZC Server		%Throughput (Pass ZC/ Non pass ZC)
	Time (Sec)	Speed (Byte/Sec)	Time (Sec)	Speed (Byte/Sec)	
1,024	0.006	170,667	0.007	146,286	83.33%
102,400	0.017	6,023,529	0.022	4,654,545	70.58%
524288	0.051	10,280,157	0.076	6,898,526	50.98%
1,048,576	0.096	10,922,667	0.109	9,619,963	86.45%
104,857,600	8.84	11,856,355	8.931	11,740,858	98.97%
536,870,912	45.269	11,859,571	45.598	11,774,001	99.27%
1,073,741,824	90.597	11,851,847	91.07	11,790,291	99.47%

Figure 10. *Internet FTP downloading and uploading testing*
Scenario 2: Internet FTP downloading and uploading



The table 3 and 4 show downloading and uploading results comparing between pass and not-pass the ZC server. Throughput difference is very slight when the size of data transfer is more than 1 MByte.

Table 3. *Internet FTP downloading result (bandwidth = 10 Mbps)*

Download data (Byte)	Client who not pass ZC Server		Client who pass ZC Server		%Throughput (Pass ZC/ Non pass ZC)
	Time (Sec)	Speed (Byte/sec)	Time (Sec)	Speed (Byte/sec)	
1,024	0.06	17,066.60	0.063	16,253	95%
102,400	0.129	793,798.40	0.206	497,087	40.31%
524288	0.542	967,321.00	0.56	936,228	96.67%
1,048,576	0.966	1,085,482.40	0.97	1,081,006	99.58%
104,857,600	85.95	1,219,870.10	86.427	1,213,250	99.45%
536,870,912	439.84	1,220,599.40	440.64	1,218,375	99.81%
1,073,741,824	881.06	1,218,684.80	881.09	1,218,642	99.99%

Table 4. *Internet FTP uploading result (bandwidth = 1 Mbps)*

Download data (Byte)	Client who not pass ZC Server		Client who pass ZC Server		%Throughput (pass ZC / not pass ZC)
	Time (Sec)	Speed (Byte/sec)	Time (Sec)	Speed (Byte/sec)	
1,024	0.059	17,355.93	0.068	15,058.82	84.74%
102,400	1.18	86,779.66	2.35	43,574.47	84.74%
524288	4.373	119,892.06	4.57	114,648.59	95.42%
1,048,576	8.147	128,707.01	8.19	127,984.38	99.43%
104,857,600	778.48	134,693.92	779.95	134,441.96	99.81%
536,870,912	3,999.75	134,225.85	4,015.81	133,689.29	99.59%
1,073,741,824	7,976.88	134,606.67	7,983.08	134,502.20	99.92%

Scenario 3: Shooting large number of packages to ZC

Large number of packages was generated by Packit application and shouted to the ZC server. Each package has size of 1460 Bytes. A number of packages which ZC can take per second were recorded as in table 5. Maximum throughput of the ZC server is about 7,000 packages or 10 Megabytes per second or about 80 Mbps which more than enough for public Internet service.

Table 5. *Package shooting result to the ZC server*

#Packages	100,000	200,000	500,000	1,000,000
Packets/Sec	7,692.4	7,407.11	7,246.26	7,103.94
Bytes/Sec	11,230,769	10,814,814	10,579,710	10,313,725

Functional testing (diversity of applications)

Following applications were tested to ensure that ZC can work with all well-known applications.

SecureCRT:	Protocol Telnet, SSL, SSH1,SSH2
FileZilla Client:	Protocol FTP
FileZilla Server:	Protocol FTP
IE, Firefox, Opera, Chome:	Protocol HTTP, HTTPS
VNC Viewer, TeamViewer:	Protocol TCP
Outlook, Outlook Express:	Protocol SMTP, POP3
Windows Live:	Protocol TCP
MultiPing:	Protocol ICMP
Samba:	Protocol UDP
CounterPath-eyeBeam:	Protocol SIP, RTP, RTCP
P2P Torrent:	Protocol UDP

Reliability or stability testing

ZC module was tested against a public WIFI service which had client about 30-40 clients concurrently connected to the internet. ZC's CPU and memory usage were captured for a week to see stability of the service as shown in picture 10, 11 and 12.

Figure 10. *ZC process ID monitoring*

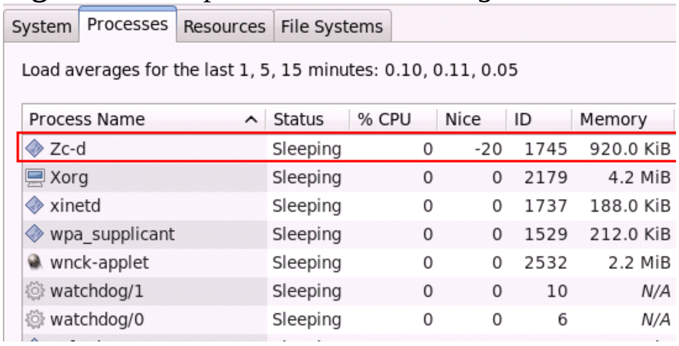


Figure 11. *ZC's CPU usage over a week*

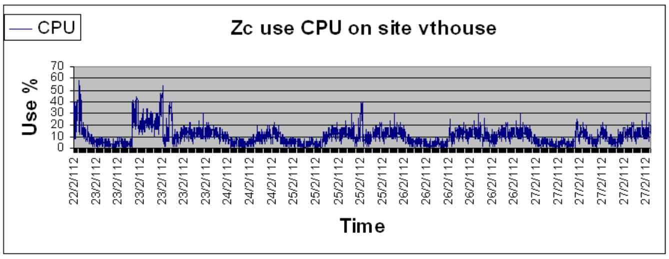
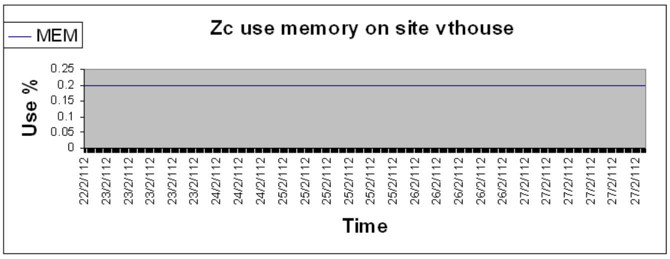


Figure 12. *ZC's CPU usage over a week*



Discussion

Ethernet frame:

ZC was designed to detect only normal and with VLAN-tagging Ethernet frame packages as depicted in figure 2. The ZC module extracts package in fix-length manner. ZC cannot cope with other formats of Ethernet frame such as IPv6.

Performance:

Indexing technique is used in IP-Mapping table in order to have a faster looking up records while processing "Convert 1" and "Convert 4" as in Figure

4, 5, 6 and 7. The ZC module cannot select to convert only packages of clients who have wrong IP-address configuration. Packages of all clients will be converted to virtual IP-address and MAC-address.

Reliability:

IP and MAC addresses mapping-record will be removed when clients disconnect from ZC server to maintain availability of virtual IP-address to the next clients and to reduce the size of the table to have a faster record lookup performance.

Conclusion

IP-Address zero configuration service uses the same principle as one of standard Linux's Network Address-Translator (NAT) [10] mixed with DHCP service. Its purpose is to covert Ethernet packages of clients to required format of the Internet gateway. Standard NAT does only conversion but not detect missed or wrong IP-address configuration like ZC does.

The Experimental results show that the ZC module can convert Ethernet packages with above 99% of throughput in various testing scenarios. It can take up to 7,100 packages per second (about 80 Mbps). Moreover, its efficiency is 96.27% compared against normal data transfer (Zero-configuration module disabled) through the Internet gateway. Besides, it can handle all well-known ports and services including VPN, VLAN. The IP-mapping table resides in physical memory so conversion processes should be faster (higher throughput) if higher CPU and bigger memory BUS are allocated to the experiment. Stability testing shows steady level of CPU and memory usages under the always-on public WIFI service which serves mixing controlled and real clients over a week.

References

Darren Bounds, *Packet analysis and injection tool*, http://linux.die.net/man/8/packit

IETF, *RFC 826: An Ethernet Address Resolution Protocol*. [cited 1982 November]. http://tools.ietf.org/html/rfc826

IETF, *RFC 1180: A TCP/IP Tutorial*. [cited 1991 January]. Available from: http://tools.ietf.org/html/rfc1180

IEEE, *IEEE std 802.1Q*. [cited 2003 May 7], http://standards.ieee.org/getieee802/download/802.1Q-2003.pdf.

Mark Mitchell, Jeffrey Oldham, and Alex Samuel, *Advanced Linux Programming*. ISBN 0-7357-1043-0. June 2001.

Maxim Krasnyansky, *Virtual Point-to-Point(TUN) and Ethernet(TAP) devices*, http://vtun.sourceforge.net/tun/

Suba Varadarajan, *Virtual Local Area Networks*, http://www.cse.wustl.edu/~jain/cis788-97/ftp/virtual_lans/index.htm

PJ Frantz, GO Thompson, VLAN frame format, US patent 5,959,990, 1999.

K. Egevang, P. Francis, the IP Network Address Translator (NAT), May 1994.