Journal of Theoretical and Applied Information Technology
30th June 2014. Vol. 64 No.3
© 2005 - 2014 JATIT & LLS. All rights reserved·

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SECURE EMBEDDED WEBSERVER WITH INTERNET CONTROL MESSAGE PROTOCOL

[1]J.JAYACHANDRA BENSAM, [2]Dr.T.JAYASINGH, [3]Dr.S.RAVI
[1]Research Scholar, Department of Electronics & Communication Engineering,
Dr. M.G.R. Educational and Research Institute University, Chennai, Tamil Nadu, India
[2]Supervisor,
Dr. M.G.R. Educational and Research Institute University, Chennai, Tamil Nadu, India
[3]Professor & Head, Department of Electronics & Communication Engineering,
Dr. M.G.R. Educational and Research Institute University, Chennai, Tamil Nadu, India
E-mail: [1]bensam004@gmail.com, [2]drtjayasingh@rediffmail.com, [3]ravi_mls@yahoo.com

ABSTRACT

Embedded system based real time process control and monitoring requires several critical issues to be intrinsically taken care to avoid fatal error. Issues of concern include avoiding OS crash, avoiding fragmented frames, place proper section of code in critical section, implement the different activities as tasks, etc. In this paper, an embedded webserver using LPC1768 is implemented using Internet Control Message Protocol (ICMP) and the algorithms are written such that, OS crash due to ICMP echo message limit violation, is avoided. The ICMP frame length is monitored and only unfragmented frames are received.

Keywords: *ICMP, Secure Transmission, Network Speed, Error Reduction, Ping of Death*

1. INTRODUCTION

The network forms an essential component in computer generation and it is imperative to create a technique to monitor and manage a computer network for available resources in a network. A network system is basically a map in which various network devices are interconnected and communicate with each other. The interconnection among devices can be based on any of the network topologies. To monitor these devices by manually pinging the information in console window is a tedious process particularly if the network devices are many and the polling interval is small. The research on ICMP and SNMP protocols helps in automating the desired task assumptions such as, (i) A tool that allows user to either create the desired network topology based on the prevalent network infrastructure and enter network devices configuration details or (ii) Allow the network monitoring tool to self discover the underlying network topology is essential. Monitoring process is then started which takes the IP list and sends ICMP echo request packets and accepts ICMP echo reply to determine the device status. ICMP is an integral part of any IP implementation. ICMP messages are sent in IP packets and it uses IP as a higher-level protocol and is implemented in every IP module. ICMP messages are classified into two main categories,

(i) ICMP Error Messages
(ii) ICMP Query Messages

To provide / send error messages for non-transient error conditions, and to provide a way to probe the network in order to determine general characteristics about the network. A number code, also known as the “message type”, is assigned to each ICMP message; it specifies the type of the message. Another number code represents a “code” for the specified ICMP type and acts as a sub-type and interpretation is dependent upon the message type.

1.1.Objective Of The Work

The main objective of this work is to implement an embedded system based webserver, make it secure and prevent OS crash due to long echo message in ICMP. Eliminating the fragmented frames at regular intervals intrinsically in the algorithm is done to achieve the objective.

2. ICMP MESSAGE ENCAPSULATION

The Internet Control Message Protocol (ICMP) is used by routers and hosts to send network control information to each other. It is a separate protocol that sits on IP and uses IP to transport messages. It is considered the part of the IP layer that communicates error message and other conditions that require attention. The ICMP messages are acted on by either the IP layer or the higher layer protocol and messages cause error to be returned to user processes that are transmitted within IP datagrams as shown in Figure 1.

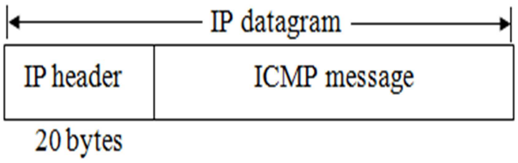


Figure 1: ICMP messages encapsulated within an IP datagrams

ICMP messages are encapsulated in IP datagrams, as is shown in Figure 2.

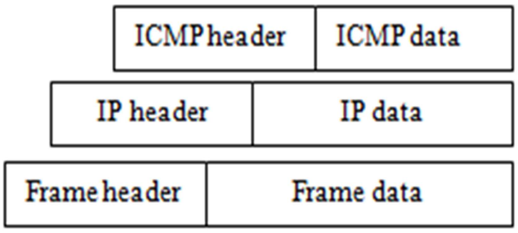


Figure 2: ICMP messages encapsulation

ICMP frames are identified by IP Protocol field value and used by IP to send error and control messages. The ICMP message format is shown in Figure 3.

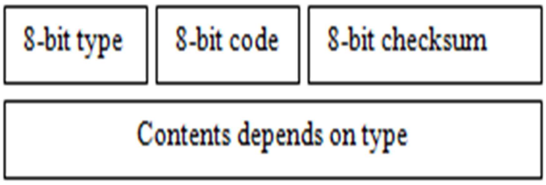


Figure 3: ICMP message format

It is used to communicate IP status and error messages between hosts and routers and implemented with IP as just a packet delivery system. It transmits and routes datagrams from sources to destinations through a series of interconnected networks. It has a checksum in the IP header to detect lost bits and no error detection on the datagram payload, but has no native mechanism for source host notification, where ICMP comes in used to report IP errors to the source host and ICMP data is carried as the payload of an IP datagram that specifies additional message formats.

3. TYPES OF ICMP MESSAGES

There are different types of ICMP messages, which suits unique purposes. They are briefed below and the types are given below in Table 1.

3.1 Information Messages

The sender sends a query to another machine (either host or router) and expects an answer. For e.g. a host might want to know if a router is alive.

3.2 Error Indication Messages

The IP software on a host or router has encountered a problem processing an IP datagram. For e.g. it may be unable to route the datagram to its destination.

Table 1: ICMP Message Types

Category	Information
Error Reporting Messages	Destination unreachable
	Source quench
	Parameter problem
	Redirection
Query Messages	Echo request/reply
	Timestamp request
	Address mask
	Router solicitation

4. ICMP ERROR REPORTING

ICMP always transmit error messages to the original source as shown in Figure 4.

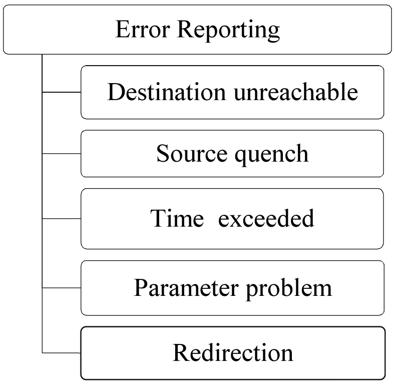


Figure 4: ICMP Error Message

4.1 Features of ICMP

The following points are,

(i) No ICMP error message will be generated in response to a datagram carrying an ICMP error message

(ii) No ICMP error message will be generated for a fragmented datagram that is not the first fragment

(iii) No ICMP error message will be generated for a datagram having a multicast address

(iv) No ICMP error message will be generated for a datagram having a special address such as 127.0.0.0 or 0.0.0.0.

(v) Echo messages are limited 2^{16} i.e.65536bytes data in the data part of the packet

(vi) Prominent information is in the header

4.2 ICMP Data for Error Messages

When a gateway cannot route a datagram (e.g., an appropriate route in its local table and bit is set) it discards the message and returns an ICMP destination unreachable message to the sending host (refer Figure 5).

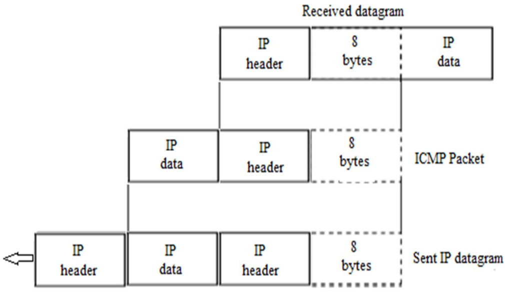


Figure 5: ICMP destination message to sending host

When a gateway becomes congested and runs out of a buffer space, it may discard a datagram and return a source quench message. It requests to the sender to reduce the rate of sending datagram.

4.2.1 Time exceeded

The routers decrement its time-to-live (TTL) field. If the TTL value reaches 0, the gateway discards the datagram and sends a time exceeded message to the sender. It is used by a destination host to show that not all fragments have arrived within a set time.

4.2.2 Parameter problem

When a host encounters problem with an IP datagram, it returns a parameter problem message to the datagram's sender and sent from a router to a local host on the same network. It informs the source of a better route to the destination. A host usually starts with a small routing table that is gradually augmented and updated.

5. ICMP QUERY MESSAGES

This is used to diagnose network problems, this type of ICMP message is answered in a specific format by the destination node. The query message process is shown in Figure 6.

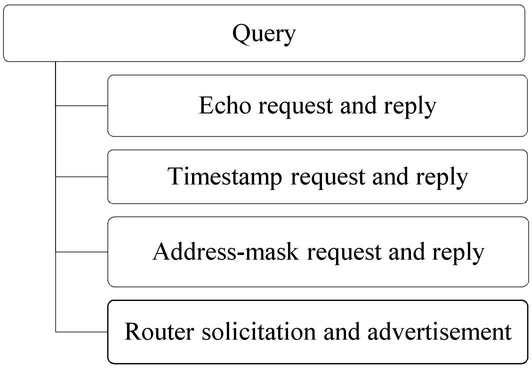

Query
Echo request and reply
Timestamp request and reply
Address-mask request and reply
Router solicitation and advertisement

Figure 6: ICMP Query Messages

5.1 Echo Request / Reply

If a network system sends an ICMP echo request message to another node network, then another node in the system is required to respond with an ICMP echo reply. The program ping allows a user to check whether a machine is reachable and functioning. The identifier and sequence number are used by the echo sender to aid in matching the replies with the echo requests.

5.2 ICMP Attacks

ICMP is a protocol used in internet layer of TCP/IP protocol suite to send error messages and carryout network management tasks. It is a 'Ping' tool which is used to send echo message to know the live status of the destination. The ICMP protocol does not have any authentication built-in and attacker can intercept ICMP packets. 'Ping' is used to launch denoal of service attacks to defenses against ICMP attacks may include the following key points are,

(i) Checking if the packet belongs to the same connection or not.
(ii) Changes in routes should be authorized to a particular connection.
(iii) Reply packet should be accepted at a particular time.

6. LIMITATIONS OF ICMP BASED EMBEDDED SYSTEM INTERFACE

Operating systems running on embedded target board can get crashed if echo messages exceeding 2^{16} bytes is specified. This ICMP echo message of gargantuan size will crash the code running in embedded target. To overcome this problem in ICMP based embedded system interface the incoming ICMP frame length needs to be checked and only unfragmented frames need to handled by the embedded system. The code development should be done to avoid this limitation.

In this paper, an existing embedded system (LPC1768) based webserver to display the process value from a plant using ICMP protocol is used for experimentation. The existing code was not optimized with respect to ICMP problem and the code crashed. The code is optimized to handle

(i) Fragmented frames

(ii) To take only 36 bytes even if echo message exceeds 2^{16} bytes.

This is done by including the following in the code.

a. if(!(val&(IP_FLAG_NOREFRAG| IP_FRAGOFS_MASK)))
The above line of code checks for unfragmented frames and whenever fragmented frames are sent, it will be discarded. This, however, does not alter the protocol type, source ip and destination IP address.

b. if(RecdIPFrameLength>MAX_ETH_TX_DATA_SIZE)
ICMP Data count = MAX_ETH_TX_DATA_SIZE-IP_HEADER_SIZE_ICMP_HEADER_SIZE;
else
ICMP Data count = RecdIPFrameLength-IP_HEADER_SIZE_ICMP_HEADER_SIZE;

The above line of code ensures that if more than the maximum Ethernet data size (i.e. 60 bytes) is received then only 36 bytes are assigned to the ICMP Data Count. Thus IP_HEADER_SIZE of 20 bytes and ICMP_HEADER_SIZE of 4 bytes along with the accepted 36 bytes will not violate the maximum Ethernet data size. Thus, buffer overflow is avoided and the code will not crash.

7. IMPLEMENTATION DETAILS / RESEARCH METHODOLOGY

The research methodology details are shown in the block diagram in Figure 7. It consists of the embedded target using LPC1768, the webserver lunched by 1768 in host and the Linux kernel port exploiting the ICMP limitation.

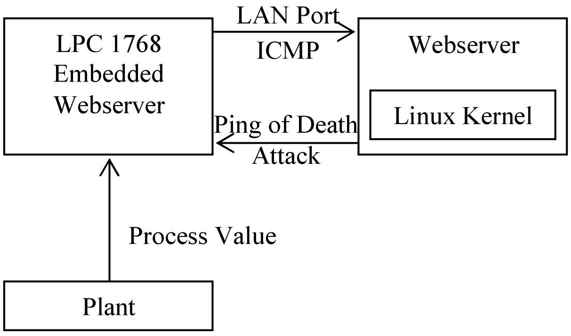


Figure 7: Block Diagram of Embedded System based Webserver using ICMP

The embedded system based webserver interface for monitoring the process includes the following steps:

1) Get the IP address

2) Initialize buffer size (Ex:1500) for storing the message i.e. IP header and ICMP data. This is done using code line.

```
Char buf[1500];
bzero(buf, sizeof buf);
```

3) Create a socket for communication. This should include domain name, type and protocol. In this work AF_NET is set to IPv4 Internet Protocol, SOCK__RAW provides raw network protocol access and IPPROTO_ICMP selects the ICMP protocol

4) Manipulate the option for socket. This contains the IP header i.e. IP_HDRINCL

5) Extract the structure of the given IP address and copy the number of bytes

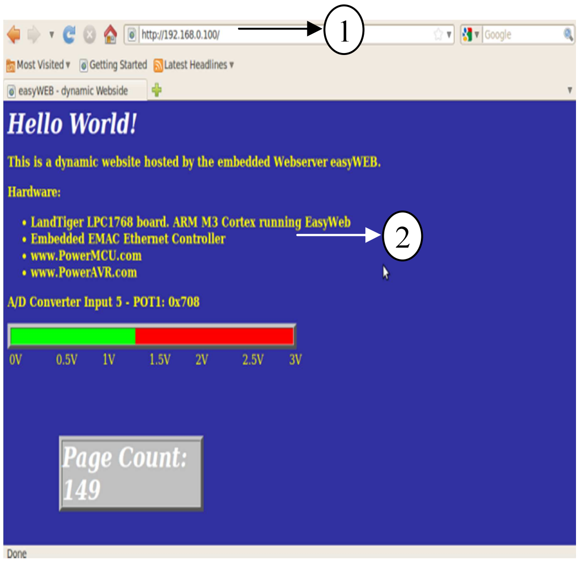
6) Represent the IP Header and ICMP payload. This needs the declaration of following information;
   (a) IP Version
   (b) Header length
   (c) Types of service
   (d) Length
   (e) Identification
   (f) Flags
   (g) Offset
   (h) Time to live
   (i) Protocol
   (j) Checksum
   (k) Source IP Address
   (l) Destination IP Address

   ICMP also requisites the following order
      (i) Types of message
      (ii) Code
      (iii) Checksum

7) After declaring IP header, the packets are sent to destination IP Address.

The code details are listed in Appendix -I

8. RESULT AND DISCUSSION



Hello World!

This is a dynamic website hosted by the embedded Webserver easyWEB.

Hardware:

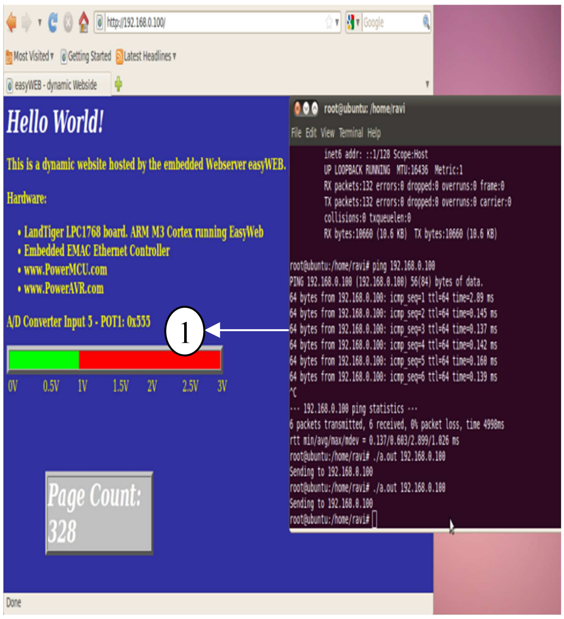
- LandTiger LPC1768 board. ARM M3 Cortex running EasyWeb
- Embedded EMAC Ethernet Controller
- www.PowerMCU.com
- www.PowerAVR.com

A/D Converter Input 5 - POT1: 0x708

Page Count: 149

(1) Webserver IP Address (2) Process value displayed in the webserver (Both in value and Bar graph)

Figure 8: Embedded Web Server restored with code optimization to prevent code crash



Hello World!

This is a dynamic website hosted by the embedded Webserver easyWEB.

Hardware:

- LandTiger LPC1768 board. ARM M3 Cortex running EasyWeb
- Embedded EMAC Ethernet Controller
- www.PowerMCU.com
- www.PowerAVR.com

A/D Converter Input 5 - POT1: 0x555

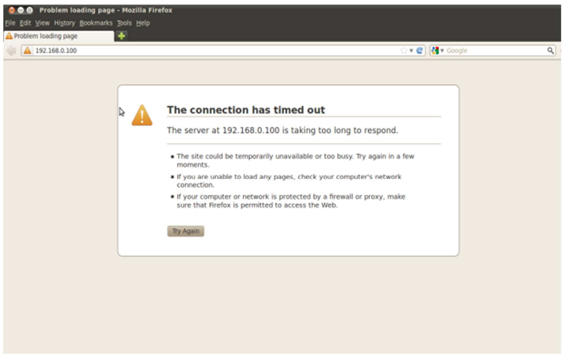
Page Count: 328

```
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:132 errors:0 dropped:0 overruns:0 frame:0
          TX packets:132 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:10660 (10.6 KB)  TX bytes:10660 (10.6 KB)

root@ubuntu:/home/ravi# ping 192.168.0.100
PING 192.168.0.100 (192.168.0.100) 56(84) bytes of data.
64 bytes from 192.168.0.100: icmp_seq=1 ttl=64 time=2.89 ms
64 bytes from 192.168.0.100: icmp_seq=2 ttl=64 time=0.145 ms
64 bytes from 192.168.0.100: icmp_seq=3 ttl=64 time=0.137 ms
64 bytes from 192.168.0.100: icmp_seq=4 ttl=64 time=0.142 ms
64 bytes from 192.168.0.100: icmp_seq=5 ttl=64 time=0.160 ms
64 bytes from 192.168.0.100: icmp_seq=6 ttl=64 time=0.139 ms
^C
--- 192.168.0.100 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 4998ms
rtt min/avg/max/mdev = 0.137/0.603/2.899/1.026 ms
root@ubuntu:/home/ravi# ./a.out 192.168.0.100
Sending to 192.168.0.100
root@ubuntu:/home/ravi# ./a.out 192.168.0.100
Sending to 192.168.0.100
root@ubuntu:/home/ravi# 
```

(1) Experimenting with Ping of Death Attack from Linux kernel

Figure 9: Embedded Web Server with Ping of Death Attack


Problem loading page - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Problem loading page
192.168.0.100
Google
Try Again

The connection has timed out

The server at 192.168.0.100 is taking too long to respond.

- The site could be temporarily unavailable or too busy. Try again in a few moments.
- If you are unable to load any pages, check your computer's network connection.
- If your computer or network is protected by a firewall or proxy, make sure that Firefox is permitted to access the Web.

Figure 10: Embedded Web Server crashed due to Ping of Death Attack in ICMP

8.1. LIMITATIONS OF THIS WORK

The embedded webserver is implemented in keil environment on LPC1768 ARM processor. In this experimented work, whenever the code is changed, it needs to be compiled in Keil and converted to a hex file and then downloaded into target. However, a more dynamic implementation, that permits on-the-fly code changes, could be using a userspace llike python under linux. This eliminates the need for compiing the changes in code everytime.

9. CONCLUSION

In this work, a secure webserver algorithm is embedded with the hardware core (ARM processor) to prevent code crash. Additionally, node can be protected in conjunction with security control (such as physical access, authentication, authorization or network controls) to adequately ensure the confidentiality, integrity and availability of the node link.

REFERENCES:

[1] Aman Mahajan ,Haresh Joshi ,Sahil Khajuria , Anil k Verma, “ICMP, SNMP: Collaborative Approach to Network Discovery and Monitoring”, *International Journal of Smart Sensors and Ad Hoc Networks (IJSSAN) ISSN No. 2248-9738* Vol.1, Issue-3, 2012, pp.8-12.

[2] HE Li-juan, WANG Dong-hong, "Topology discovery algorithm of the SNMP-based network--layer protocol”, *Journal of Shijiazhuang Vocational Technology Institute*, Vol.21 No.6, 2009.

[3] Li, Yanbing; Ma, Yue; Wang, Wei; Wan, Xiaoqiang, “A link layer topology discovery algorithm based on STP”, *Jisuanji Gongcheng/Computer Engineering,* Vol.32, No.18, Sep 20,2006 , pp.109-110+113.

[4] Yang Qiuxiang; “ Algorithm Research of Topology Discovery on SNMP”, *International Conference on Computer Application and System Modeling (ICCASM 2010)*,Vol.12, pp. 496-497.

[5] J. Wei-hua, Du jun "The application of ICMP protocol in network scanning", *Proceedings of International Conference on PDCAT*, Aug 2003, pp.904- 906.

[6] Madalina Baltatu, Antonio Lioy, Fabio Maino, Daniele Mazzocchi. “Security Issues in Control, Management and Routing Protocols”. 22-25 May 2000. *URL : http://www.terena.nl/tnc2000/proceedings/3A/3a2.pdf (10 Dec 2001).*

[7] Alex Peeters. “ICMP Header Format”. 4 October, 1999. *URL : http://citap.freeservers.com/publications/tcp-ip/tcpip012.htm (10 Dec 2001).*

[8] Lindsay van Eden. “The Truth About ICMP”. 17 May 2001 *URL : http:// www.sans.org /infosecFAQ/threats/ICMP.htm (10 Dec 2001).*

[9] Abdullah H. Alqahtani, Mohsin Iftikhar, “ TCP/IP Attacks, Defenses and Security Tools”, *International Journal of Science and*

Modern Engineering (IJISME) ISSN: 2319-6386, Vol.1, Issue-10, September 2013, pp:42-47.

[10] "Wireshark", online, *www.wireshark.org. (last accessed on 25 May 2013*)

[11] Bellovin, Steven M. "A look back at." *Computer Security Applications Conference*, 2004. 2*0th Annual. IEEE,* 2004.

[12] Trabelsi, Zouheir, and Khaled Shuaib, "NIS04-4: Man in the Middle Intrusion Detection." *Global Telecommunications Conference, 2006*. GLOBECOM'06. IEEE. IEEE, 2006.

[13] Yan, Boru, et al, "Detection and defence of DNS spoofing attack," *Jisuanji Gongcheng/ Computer Engineering 32.21* (2006), pp.130-132

[14] Zaraska, Krzysztof, "Prelude IDS: current state and development perspectives," *URL http://www. prelude-ids. org/ download / misc / pingwinaria/2003/paper. pdf*(2003)

[15] Yao, Xiaoyu, and Chen ZHAO, "Research on Implementation and Application of Linux Kernel Firewall Netfilter [J]," *Computer Engineering 8 (2003):* 042

APPENDIX –I

Step	Code	Remarks
1	int main(int argc, char *argv[])	-

2	Char buf[1500];
	bzero(buf, sizeof buf);
		Making zero valued bytes by using bzero() function

3	s=socket(AF_INET,SOCK_RAW,IPPROTO_ICMP	-
4	setsockops(s,IPPROTO_IP, IP_HDRINCL, &on, sizeof(on)	-
5	bcopy(hp->h_addr_list[0], &ip->ip_dst.s_addr, hp->h_lenght;	-
7	sendto(s, buf, sizeof buf, 0, (struct sockaddr *) & dst, sizeof dst)	(struct sockaddr *)&dst – Points to a sockadde structure containing the destination address.