International Journal of Computer Sciences and Engineering Open Access
Review Paper Volume-5, Issue-3 E-ISSN: 2347-2693

Digital World Bug: Y2k38 an Integer Overflow Threat-Epoch

S. Harshini[1*], K.R. Kavyasri[2] , P. Bhavishya[3], T. Sethukkarasi[4]

[1*]Department of Computer Science and Engineering, R.M.K Engineering College, Chennai ,India
[2]Department of Computer Science and Engineering, R.M.K Engineering College, Chennai ,India
[3]Department of Computer Science and Engineering, R.M.K Engineering College, Chennai ,India
[4]Department of Computer Science and Engineering, R.M.K Engineering College Chennai ,India

[*]*Corresponding Author: hars14137.cs@rmkec.ac.in*

Available online at: www.ijcseonline.org

Received:12/Feb/2017 Revised: 20/Feb/2017 Accepted: 10/Mar/2017 Published: 31/Mar/2017

***Abstract*-: Digital universe has been menaced by plenty of bugs but only a few seemed to pose a great hazard. Y2K,Y2K10 were the most prominent bugs which were blown away. And now we have Y2K38. The Y2K38 bug, if not resolved, it will get hold off the predictions that were made for the Y2K bug would come face reality this time. Y2K38 bug will affect all the system applications and most of the embedded systems which use signed 32 bit format for representing the internal time. The number of seconds which can be represented using this signed 32 bit format is 2,147,483,647 which will be equal to the time 19, January, 2038 at 03:14:07 UTC(Coordinated Universal Time),where the bug is expected to hit the web. After this moment the systems will stop working correctly. This could wipe out programs that rely on the internal clock to make measurements. There have been some solutions which delayed this problem, that we can have some more time to find a universal solution and so does our proposed solution.**

Keywords-Date,Time,Y2K,Y2K38 bug

I. INTRODUCTION

Y2K bug was the result of practice of abbreviating a four digit year to two digits.The practice of representing years in two digits creates a problem when one rolls over from x99 to x00.However, the number of computer failures due to Y2K is not known.[1][2] Another bug which is threatening to cause great damage is the Y2K38 bug. The Y2K38 problems arise because of the similar reason which was responsible for the Y2K problems. As the Y2K problems resulted from not allocating enough bits for the representation of year, the Y2K38 problems are a result of not allocating enough bits for the representation of internal time. C and C++ programs use a time_t data type for the representation of dates an time internally. And 4 bytes are allocated for this representation. In these 32 bits, one bit is assigned for showing the positive/ negative integer value and hence, 31 bits are left for representing the time value. The highest value that can be represented using these 31 bits is 2,147,483,647 and what this number will represent in time_t is equal to 19 January, 2038 at 03:14:07 UTC. At this moment, every time_t used in a 32 bit C or C++ program will reach its upper limit and after that moment, the 32 bit clocks will overflow and will return some erroneous value. January 19,2038 is a date which will result in the failure of many computer platforms, softwares and embedded systems as these systems will run out of time. On 19, January, 2038, as soon as the clock will strike 03:14:07, all these systems will stop working properly.[3]Try setting the date on your phone to a day beyond this. iPhones will only allow you to reach January 1, 2038, and the rest of the dates are grayed out. In the past, some Android devices have been reported to crash and fail to reboot when set to this date [6].And the result will be massive power outages, hospital life, support system failures, bank problems, satellites falling out of orbit, et cetera. The Y2K38 bug is also called the Unix Millennium bug because most 32-bit Unix-like systems store and manipulate time in this format and this is usually called UNIX time. Most programs which are written in the C programming language will suffer from use a library of routines called the standard time library (time.h)

Date & time	time_t (decimal) – seconds after epoch	time_t (32-bit binary) representation
1-Jan-1970, 00:00:00 UTC	0	00000000 00000000 00000000 00000000
1-Jan-1970, 00:00:01 UTC	1	00000000 00000000 00000000 00000001
1-Jan-1970, 00:01:00 UTC	60	00000000 00000000 00000000 00111100
1-Jan-1970, 01:00:00 UTC	3 600	00000000 00000000 00001110 00010000
2-Jan-1970, 00:00:00 UTC	86 400	00000000 00000001 01010001 10000000
1-Jan-1971, 00:00:00 UTC	31 536 000	00000001 11100001 00110011 10000000
1-Jan-2038, 00:00:00 UTC	2 145 916 800	01111111 11101000 00010111 10000000
19-Jan-2038, 03:14:07 UTC	2 147 483 647	01111111 11111111 11111111 11111111
13-Dec-1901, 20:45:52 UTC	-2 147 483 648	10000000 00000000 00000000 00000000

II. ABOUT Y2K38: THE BUG

This erroneous value may be 1, January, 1970 or 13, December, 1901 which is a Friday and hence, y2k38 bug is also called Friday, the 13th bug. C is the most widely used programming language due to its compactness and embedded softwares mainly use C language due to which the fields in which such softwares are being used will also be adversely affected. And, hence, these protocols too will be affected from this bug. Another early manifestation of the Y2K38 bug was diulged when the Mars rover Opportunity had a software crash: "The rover was babbling, doing things like sending nonsensical communications that it date-stamped as being from the year 2038 [9]."

A.Time and Date Representation:

Time_t is actually an integer which is counting the number of seconds that have passed since January 1, 1970 at 00:00:00 UTC. it will calculate the time by subtracting as many seconds from 1, January, 1970 and this will give a date and time due to which many applications will fail. And the value of this time_t will be Friday, 13th December, 1901. This date is called the 'wrap around' date. There are many data structures which will result in this problem. some are File systems (many file systems use only 32 bits to represent times in inode) · Databases (that have 32-bit timefields) ·Embedded factory, refinery control and monitoring subsystems
-assorted medical services.
-assorted military services.

Table2**:** time_t representation of date,time

B.Below is a piece of code that will simulate the bug Y2K38

```
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <time.h>
int main (intargc, char **argv)
{
time_t t;
t = (time_t) 1000000000;                    printf ("%d, %s",(int)t,asctime(gmtime(&t)));
t = (time_t) (0x7FFFFFFF);
printf ("%d, %s", (int) t, asctime (gmtime (&t)));
t++;
printf ("%d, %s", (int) t, asctime (gmtime (&t)));
return 0;
}
```

Output-

1000000000, Sun Sep 9 01:46:40 20012147483647, Tue Jan 19 03:14:07 2038-2147483648, Fri Dec 13 20:45:52 1901

III. RELATED WORK

The related work done for Y2k and Y2k38 are:
A. for Y2K bug there were many solutions.Few of them are –

1. Date Expansion- Inclusion of the century in the two digit representation was done in programs, files and databases
2. Windowing- Two-digit years were retained, and programs determined the century value only when needed for particular functions
3. Date Re-partitioning- In legacy databases whose size could not be economically changed, six-digit year/month/day codes were converted to three-digit years .

IV. CONTRAST BETWEEN Y2K AND Y2K38

Y2k bug was a computer imperfection that made the common practice of representing year in four digit numbers abbreviating with only last two digits.In 1960s – 1980s, dates were stored in YY MM DD format, where each character occupied a whole byte of of computer storage i.e, 8 bits.Saving two bytes per data entry was a significant amount,. As a result, the number 19 has to be omitted. For instance, the year 1980 would simply be stored as two digit number "80."This broached a serious problem as the year 2000 approached. Instead of showing 2000, it would simply read "00"and most of the programs would mistake this to 1900, causing a plethora of problems – primarily, calculations being off by *negative* 99 years [11]. In retrospect, it's very easy to say that programmers should have forecasted Y2K bug, but computers then were a new technology and programmers did not anticipate that their codes will be still in use after 50 years.

As a result, atmost 300 billion US Dollars were spent to correct problems that might arise due to this Y2K. The number of actual failures due to Y2K is unpredictable. However, the number of reported problems was much lower than the hype predicted, leading many to question whether this was a result of excellent preparation or if the problem had been grossly exaggerated [12].

While both Y2K AND Y2K38 bugs are time/date-related issues, they occur in completely different aspects. Everything that Y2K was supposed to affect (or did affect) will also be affected by the Y2K38 bug. But Y2K38 occurs on a basic level, the actual representation of time itself – which is a problem because it affects just about any program or device that needs to know about time. While Y2K was relatively easy to understand, Y2K38 is a little more difficult in that it requires a deeper understanding of how time is digitally managed in computers, which may contribute to its obscurity. In addition, because of the perception of overhype and unnecessary alarm associated with Y2K since "nothing happened"and "the world didn't end,"it's easy to see why Y2K38 is being so easily dismissed as a problem that will fix itself. However, this optimism may not be enough to get through to 2038 unscathed. It is likely that the overwhelming media attention toward Y2K forced businesses to proactively address it, mitigating the issue almost fully prior to the start of the millennium [13].

V. PROPOSED SOLUTION

For Y2K38 bug, we don't have any universal solution yet. But some of the proposed solutions are-

A. Change the definition of time_t- this is one of the quick –fixes that have been suggests for the existing 32-bit software is to redefine time_t as an unsigned integer instead of a signed integer. An unsigned integer doesn't have to waste one of its bits to store the plus/minus signed for the number it represents. This will double the range of numbers it can store. Whereas a signed 32-bit integer can only go upto 21 47 483 647, an unsigned 32-bit integer can go all the way up to 4 294 967 295.A time_t of this magnitude could represent any date and time from 12:00:00 AM 1-Jan-1970 all the out to 6:23:15 AM 7-Feb-2106.Firmly giving us more than a year for 64 bit software to dominate the planate [15]. This would result in code compatibility problems as there are certain applications which are dependent on the signed 32 bit representation of time

B.Shift from 32 bit systems to 64 bit systems- Shifting from 32 bit systems to 64 bit systems gives us a new wrap around date which is over 20 times greater than the estimated age of universe i.e. about 292 billion years from now.[16]

C. NetBSD's solution- Starting with its 6th major release, NetBSD is using a 64 bit time_t for both 32 bit and 64 bit architectures which supports 32 bit time_t in applications.

VI. CONCLUSION

We want to suggest that if we can slow down the time of these 32 bit systems then we will have some more time to find a universal solution or might be able to replace most of the 32 bit systems with 64 bit systems in that extra interval of time. Then, we will get 25 more years to resolve the Y2K38 problem.

Y2K38 being a serious problem needs to be solved properly and within time. But, a solution that can be applied universally is not yet there. Most of the solutions are either for a particular software, system or for delaying this problem. Our suggested solution too tries to delay the problem so that we can have some extra time. Using the given solution, the year 2038 problem will become the year 2063 problem.

REFERENCES

[1]. M. Schumacher,"Year 2000, Y2K, millennium bug", *Power Engineering Society Summer Meeting*, 1999. IEEE, ISBN: 0-7803-5569-5
[2]. Zijiang Yang,J.C.Paradi,"*DEA evaluation of a Y2K software retrofit program*", IEEE Transactions on Engineering Management , Vol.51(3),ISSN: 0018-9391,2004.
[3]. Vikas Chandra Sharma,"*YK-2028 Bug*",Journal Of Image Processing And Artificial Intelligence, *Vol .1(3),*2015.
[4]. Diomidis Spinellis,"*Code quality: the open source perspective. Effective software development series in Safari Books Online*". Adobe Press. ISBN 0-321-16607-8.
[5]. Margaret Ross,"*The Y2K Legacy—The Time Bomb of Tomorrow*", Kluwer Academic Publishers, Vol 10(4), pp 281–283,2002 Print ISSN:0963-9314
[6]. Carrington, Damia "*Was Y2K bug a boost?"*BBC News. Archived from the original on 22 April 2004.
[7]. Raymond B.Howard,*"The Case for Windowing: Techniques That Buy 60 Years*", Year/2000 Journal, Mar/Apr 1998."
[8]. "Issue 16899: Year 2038 problem,"Android, 16 May 2011.
[9]. J. L. Smoot and P. J. Meyer, "*What is UTC?*,"NASA - Marshall Flight Space Center, 29 March 1995.
[10]. Jet Propulsion Laboratory, "*Mission Fantastic to Mars (Part 4),*"California Institute of Technology,
[11]. The Open Group and IEEE, "*The Open Group Base Specifications Issue 6*, IEEE Std 1003.1, 2004 Edition (definition of epoch),"The Open Group; IEEE;, 2004
[12]. "*Y2K Bug,"*National Geographic Society 1996-2013
[13]. Dutton, Denis , "*It's Always the End of the World as We Know It*", The opinion Pages,The New York Times. *31 December 2009.*
[14]. R. M. Wilcox, "*The Year 2038 Problem*,"23 October 2003.
[15]. Rae Zimmerman,"*Y2K readiness helped NYC on 9/11*", MIT News, 19 November 2002.
[16]. Arnd Bergmann,"*The End Of Time(32bit edition)*",Embedded Linux Conference,San Jose,CA,2015.

Authors Profile

Ms.Harshini.S currently pursuing her Bachelor of Engineering, in department of computer science and engineering in R.M.K engineering college(affiliated to Anna university).Member of CSI, Her Research work mainly focuses on bigdata analytics,cloud security,IOT,and advanced computing.She is a member of CSI.

Ms K.R.Kavya Sri is currently pursuing her Bachelor of Engineering, in department of computer science and engineering in R.M.K engineering college (affiliated to Anna university). Member of CSI, Her Research work mainly focuses on bigdata analytics and IOT.She is a member of CSI.

Ms P.Bhavishya is currently pursuing her Bachelor of Engineering, in department of computer science and engineering in R.M.K engineering college (affiliated to Anna University).Member of CSI, Her Research work focuses on data mining, cloud security, IOT, and Network Security .She is a member of CSI,ISTE.

Mrs. Sethukkarasi is a research scholar, pursued Ph.D. Currently working as professor in department of Computer Science and Engineering, R.M.K engineering college. Her research work focuses on data mining, cloud computing, cyber security, network security. She has 8 years of teaching experience.

