Novel Approach for Worm Detection using Modified Crc32 Algorithm

Avinash Wadhe
M-Tech(CSE)
G.H.R.C.E ,Nagpur

Rahul Suryawanshi
M-Tech(CSE)
G.H.R.C.E ,Nagpur

Nikita Mahajan
M-Tech(CSE)
G.H.R.C.E ,Nagpur

ABSTRACT
Computers are designed to execute instructions one after another. Those instructions normally do useful calculations, maintain databases and communicate with users on other systems. Sometimes, the instruction execution can be damaging or malicious in nature. When that happens by accident, we call it a code involved a software bug or cause an unexpected program behavior. If the instructions source are individual who intended that the occurrence of abnormal behavior, then we consider this as infected coding; such a code authorities sometimes referred as worm. There are lot of distinct forms of such software which are characterized by there behavior, how they are fired, and how they spread. Now-a-days, the media almost uniformly described occurrences of worms as computer viruses. One of the most high profile threats to information integrity is the computer virus and worms. This paper presenting how to mitigate the worms using the improved CRC32 approach and comparing the result with existing technique available for worm detection.

Keywords
Worms, CRC32, Modified CRC32

1. INTRODUCTION

The "virus" is a term used to refer other types of malware, adware, and spyware programs commonly but erroneously. The specific term that should be used is “Malware”. Malware includes computer viruses, Trojan horses, most root kits, spyware, worms and other unwanted software, including true viruses [6].Thus, when discussing about worms, a common question arises What is the difference between a worm and a virus? Both are considered to be malware and can perform the same malicious actions. Typically, Virus are not self-propagated. They are depended on user to activate and transfer to new destination [7].A computer worm is a program of self-propagation across a network exploiting policy or security flaws in widely-used services and it may do with or without any user intervention [4]. Unlike a computer virus, it does not need to attach itself to an existing program. There are a number of techniques through which a worm discovers new machines to exploit. Some are: scanning, external target lists, pre-generated target lists, internal target lists, and passive monitoring. Worms may use a combination of these strategies [4].
To detect viruses, this system will first use integrity checking technique, which will give all the infected files. Now to get more details about the infection, which implies that it is very difficult to generate a different message having the same checksum. Modified CRC-32 is more efficient than normal CRC-32. The Internet connects a vast number of personal computers, most of which run Microsoft Windows operating systems on x86-compatible architectures. Recent global security incidents have shown that this monoculture is a very attractive target for e-mail worms, self-replicating malicious programs that rely on users opening e-mail attachments out of curiosity. Spreading with this rather unsophisticated method.

Various versions of NetSky, MyDoom, and Bagle have dominated the malware statistics of 2007 and are still regularly seen in top 10 threat lists in 2010. MyDoom alone caused a total economic damage of around 3 billion US Dollars during the phase of its initial outbreak. Despite the high economic damage, it is relatively easy to develop e-mail worms: in contrast to the “classic” viruses of the pre-Internet era, which spread by infecting executable files, e-mail worms that infect hundreds of thousands of computers can be created without knowledge of system programming or assembly language. Today’s e-mail worms are commonly written in high-level programming. A worm is a program very similar to a virus; which has the same ability to self-replicate and can lead negative effects on system and most importantly they are detected and eliminated by antivirus. However, worms are not strictly viruses. They do not require to infect other files to reproduce. It reproduce rapidly without damaging file, saturate networks and cause to collapse them. They can also spread within the memory of a computer.

Effects of worm
1) Growth in traffic volume
2) Rise in number of scans and sweeps
3) Change in traffic patterns for some hosts
4) Predicting scans by analyzing the scan engine of the worm
5) Occupy the system memory by making replica unnecessarily

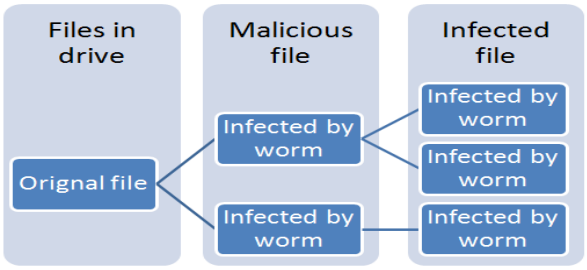

Figure 1: Working of worm

Figure 1 shows how worm spread in system .Commercial antivirus use various worm detection technique and for scanning purpose use CRC-32 algorithm match with existing databases and classify the worm infected files but issue is speed of worm infection is very fast than complexity of CRC-32 algorithm ,require powerful algorithm with high speed to detect worm in initial stage . Modified CRC-32 implementation in antivirus packages will work effectively.

RELATED WORK

Michael J. Wiacek, David J. Albanese, Jeffrey A. Six, Christopher M. Salter, [5], depending on characteristics that how worms infect, gives a method of classifying them and collect large set of *attack attributes*, which represent classes of actions taken by worms to ensure successful infection, but it fails to detect worm, regardless of their creation. It classifies the worm and detects them according to their propagation. They matched defensive technologies against attack attributes and presented it into a *defence matrix method*.

Cliff C. Zou, Don Towsley, Weibo Gong [8] provide only theoretical guideline for monitoring worm behavior and an analytical model "*hit-list routing worm*", through which they reveal the underlying similarity and relationship between different worm scanning strategies.

Parbati Kumar, Manna Shigang, Chen Sanjay Ranka [9] proposes a mathematical model that *precisely* characterizes the propagation patterns of the permutation-scanning worms. Also gives an analytical model which assumes full network connectivity, no delays and no host failures, practically those are not possible. So, we proposed a modified CRC32 algorithm which can detects worms with high scanning rate more effectively.

2. METHODOLOGY AND IMPLIMENTATION

2.1 Problem Definition:

Worm Mostly undetectable as they make replica of a file in the system or in the network very quickly. It has ability to self replicate and makes duplicate files or link shortcut of a file. Worm spread so quickly that human response was ineffective. First upon how many types of worm are present in network

Worm	Year	Strategy	Victims	Other Notes
Morris	1988	Topological	6000	First major autonomous worm. Attacked multiple vulnerabilities.
Code Red	2001	Scanning	~300,000	First recent "fast" worm, 2nd wave infected 360,000 servers in 14 hours
Nimda	2003	Scanning IIS, Code Red 2 backdoor, etc	~200,000	Local subnet scanning. Effective mix of techniques
Scalper	2005	Scanning	<10,000	Released 10 days after vulnerability revealed
Slammer	2009	Scanning	>75,000	Spread worldwide in 10 minutes

```
        /*basic crc32 algo [14]*/

Unsigned int crc32 (unsigned char *message)
{

  int a,b;
  unsigned int byte, crc;
  a = 0;
  crc = 0xFFFFFFFF;

  while (message[a] != 0)
 {
    byte = message[a]; // for next byte.
    byte = reverse(byte); // reversal 32 -bit.

    for (b = 0; b<= 7; b++)
    { // eight times.

      if ((int)(crc ^ byte) < 0)
         crc = (crc << 1) ^ 0x04C11DB7;
      else crc = crc << 1;
         byte = byte << 1; // Ready for next msg bit.

    }

   a = a + 1;
  }

  return reverse(~crc);
}
```

Spread through email, network, USB device Require efficient algorithm to detect, recent antivirus use CRC-32 algorithm to detect replicas of file created by worm. A cyclic redundancy check (CRC) is commonly used in storage devices and digital network as an error-detecting to detect accidental changes in raw data. Based on the reminder of polynomial division. The block of data entering in the system get a short *check value* attached. The basic CRC32 algorithm is a conventional cyclic redundancy check algorithm with the greater complexity of execution. But in improved bit a time CRC32 algorithm the complexity of algorithm is reduced substantially by removing the inner for loop and bit reversing process which will facilitate the faster worm presence detection.

The checksum of known clean files are gathered in a so-called white list. Then, file is checked whenever it scanned, against the white-list. Thus it reduces the chances for false positives and speeding up process for the clean. However, because of an attacker, it is a bad idea who easily craft modified virus. Such a virus can bypass the anti-virus detection. Adding the file's size to each clean file checksum does not significantly complicate the attack: crafting a malware with the same CRC32 and size as a clean file is no more than a few dozen

Minutes of computing. But to detect 1st growing worm, the complexity of CRC-32 is not enough hence to match infected file and to compute the CRC of original file, required efficient algorithm. Figure 2 showing the CRC-32 algorithm which

commonly use for detecting infected file as worm. But regarding complexity is not up to the mark henceforth most of the commercials antivirus only promise for detection of worm but

```
/* Table lookup CRC algorithm. */

unsigned int crc32(unsigned char *message)
{
   int a,b;
   unsigned int byte, crc, mask;
   static unsigned int table[256];

  /* Set up the table, if necessary. */

  if (table[1] == 0)
  {
     for (byte = 0; byte <= 255; byte++)
     {
          crc = byte;

          for (b = 7; b >= 0; b--)
          { //eight times done.
                mask = -(crc & 1);
               (crc >> 1) ^ (0xEDB88320 & mask);
          }

          table[byte] = crc;
      }
  }

/* Through with table setup, now calculate the CRC. */

  a = 0;
 crc = 0xFFFFFFFF;

  while (message[a] != 0)
  {
       byte = message[a]; // for  next byte.
       crc = (crc >> 8) ^ table[(crc ^ byte) & 0xFF];
       a = a + 1;
  }

  return ~crc;
}
```

Figure 3: Table lookup CRC-32 algorithm [14]

2.2 Using Modified CRC-32:

Figure 3 shows another lookup table CRC-32 algorithm improves some sort of fast detection of infected file .It is employs table lookup which is the usual way to calculated CRC-32. Although one bit at a time above program works, table lookup method works at a time one byte. A table of 256 full word constants is used, Mostly used by (QUICK HEAL, Macc Café).

While retaining one-bit-at-a-time character, still it can improved CRC32 algorithm. First, notice that the inner loop if statement used eight bit of the reversed byte and then discarded. Also, in inner loop, the higher-order eight bits of CRC are not altered loop (other than by shifting). By omitting the left shift byte to the bottom of loop, simplified the if-statement and thus can set crx=crc^ before the inner loop. Shifting right instead of left, the two reversals can be avoided. It can be achieved by reversing the hex constant and testing crc least significant crc-32. Finally, by replacing if-test with some simple logic, to save branches.

2.3 Implementation:

The unrolling of the inner loop by the full factor of eight is not unreasonable. Proposed work focus on per byte of input message, if this is done. This includes a load and a branch. (We depends on compiler to common the two loads of message and to transform while-loop so at the bottom of the loop there is only one branch.)

Proposed Algorithm:

```
/* improved bit-at-a-time CRC-32 algorithm.*/

unsigned int crc32(unsigned char *message)
{

  int a, b;
  unsigned int byte, crc, mask;
  a = 0;
  crc = 0xFFFFFFFF;

  while (message[a] != 0)

 {
     byte = message[a]; // Get next byte.
     crc = crc ^ byte;

    for (b = 7; b >= 0; b--)
    { // Do eight times.
        mask = -(crc & 1);
        crc = (crc >> 1) ^ (0xEDB88320 & mask);
     }
     a = a + 1;
  }

  return ~crc;
}
```

Figure 4: Modified CRC-32 algorithm

The register crc shifted right eight times by inner loop, while doing an operation or exclusive with a constant when a single right shift of eight position replaced by the low–order bit of crc, followed with mark or by single exclusive which depends on pattern of 1-bits of the crc register in the rightmost eight bits.

Propose algorithm substantialy imprvoes the efficiency of detection technique and promise to detect worm at initial stage.

3. RESULT

Experimental result shows comparative peformance of above algorithm .Infer the complexities difference between proposed algorithm.For this experiment sample worm infected file collect from worminstance site first it checke with CRC-32 algorithm and compre the result with praposed algorith n

simulteniously calculate the time required for detection and modified CRC-32 proves best among other algorithm .

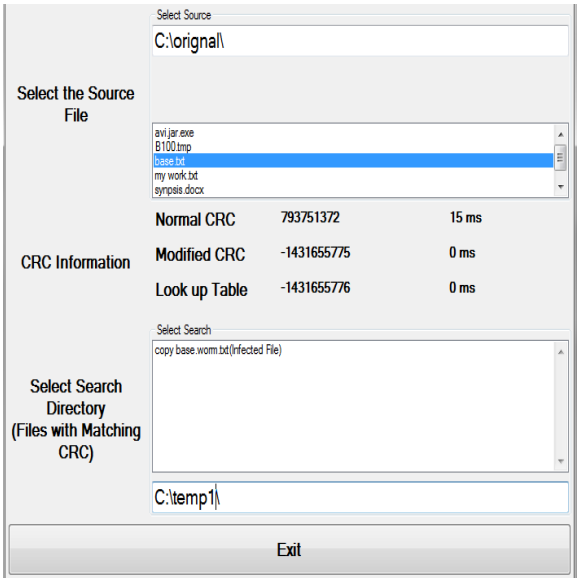

Figure 5. Screen shot of showing calculated time to detect sample worm by 3 algorithm

Comparative table indicate some popint of differnce between algorithjms so that result can be vieved at each level of executipon.

Table 2. Comparison of algorithms

CRC-32	Modified CRC-32	Lookup table CRC-32
Gives better computation for small files	Gives better computation for large file & different format files	Gives better computation for root directory files
Complexity is more than others	Complexity is substantially reduce	Complexity is equal but prior known CRC can reduce
Fail to retain calculated CRC for	Fail to retain CRC but avoid revisiting mask bit	Successful to retain CRC

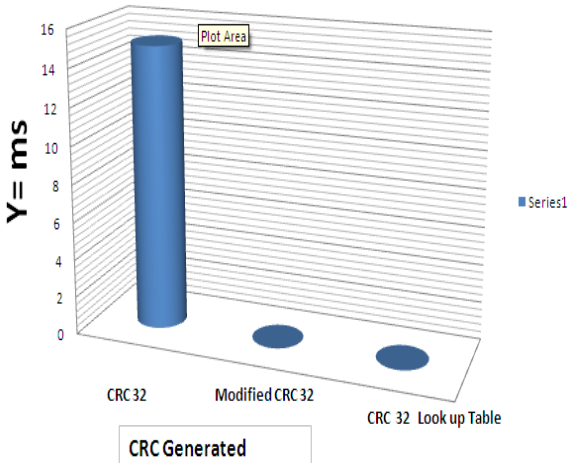

Figure 6. CRC calculated against time required

4. CONCLUSION AND FUTURE WORK

Proposed algorithm gives remarkable performance for detecting worm virus, in methodology experimental result proves that time complexity required for proposed algorithm sustain or stop fast propagation of worm in the system. By extending our algorithm to different varieties of permutation-scanning worms, we have shown that our algorithm is quite effective and holds promise for modeling even other kinds of worms .i.e. encrypted viruses. However, we believe that this algorithm should be effective in modeling most real world worms that are scanning at a very low rate to avoid detection even when such conditions do not always hold true. In future implementation of this algorithm, for detecting worms at the time of boot scan will detect worm at initial stage.

Henceforth implementation of this algorithm can be used in network exploiting security or policy flaws services.

5. REFERENCES

[1] Qi Lilei, Huang e Xiantong (2011), "Research on Worms Spread and Control Strategy in Complex Networks", IEEE

[2] Shouhuai Xu, Wenlian Lu, and Zhenxin Zhan (2012),"A Stochastic Model of Multivirus Dynamics", IEEE Computer Society.

[3] Filipe Freitas, Edgar Marques, Rodrigo Rodrigues, Carlos Ribeiro, Paulo Ferreira, and Lu´ıs Rodrigues(2009),"Verme: Worm Containment in Overlay Networks", IEEE.

[4] Nicholas Weaver, Vern Paxson, Stuart Staniford, Robert Cunningham(2003),"A Taxonomy of Computer Worms" *WORM'03*, Washington, DC, USA. Copyright 2003 ACM 1581137850/ 03/0010

[5] David J. Albanese, Michael J. Wiacek, Christopher M. Salter, Jeffrey A. Six (2004), "*The Case for Using Layered Defenses to Stop Worms*".

[6] Arun Bakshi, Vikas Dixit, Kaushal Mehta(2010),"Virus: A Menace for Information Security", Global Journal of Enterprise Information System.

[7] Craig Smith, Ashraf Matrawy, Stanley Chow, Bassem Abdelaziz(2009), "Computer Worms: Architectures, Evasion Strategies, and Detection Mechanisms", Journal of Information Assurance and Security 4.

[8] Cliff C. Zou, Don Towsley, Weibo Gong(2006),"On the performance of Internet worm scanning strategies", Performance Evaluation 63.

[9] Parbati Kumar Manna, Shigang Chen, Sanjay Ranka(2008),"Exact Modeling of Propagation for Permutation-Scanning Worms", *IEEE INFOCOM 2008.*

[10] Lidong Zhou, Lintao Zhang, Frank McSherry, Nicole Immorlica, Manuel Costa, Steve Chien "A First Look at Peer-to-Peer Worms: Threats and Defenses"

[11] Mathys Walma(2007),"Pipelined Cyclic Redundancy Check (CRC) Calculation" IEEE

[12] Zornitza Genova Prodanoff and Ronnie King(2004),"CRC32 Based Signature Generation Methods for URL Routing", IEEE.

[13] http://www.ross.net/crc/download/crc_v3.txt

[14] http://sobhan5968.blogfa.com/post-30.aspx

[15] David J. Stang, "Computer Virus Survival Guide",Page 35-39, Jun 1, 1991.