Network Working Group
Request for Comments: 3607
Category: Informational
M. Leech
Nortel Networks
September 2003

Chinese Lottery Cryptanalysis Revisited: The Internet as a Codebreaking Tool

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Abstract

This document revisits the so-called Chinese Lottery massively-parallel cryptanalytic attack. It explores Internet-based analogues to the Chinese Lottery, and their potentially-serious consequences.

1. Introduction

In 1991, Quisquater and Desmedt [DESMEDT91] proposed an esoteric, but technically sound, attack against DES or similar ciphers. They termed this attack the Chinese Lottery. It was based on a massively-parallel hardware approach, using consumer electronics as the "hosts" of the cipher-breaking hardware.

In the decade since Quisquater and Desmedt proposed their Chinese Lottery thought experiment, there has been considerable growth in a number of areas that make their thought experiment worth revisiting.

In 1991, the Internet had approximately 8 million reachable hosts attached to it and in 2002, the number is a staggering 100 million reachable hosts. In the time since the Chinese Lottery paper, computer power available to the average desktop user has grown by a factor of approximately 150.

2. Dangerous Synergy

The combined growth of the Internet, and the unstoppable march of Moore's Law have combined to create a dangerous potential for brute-force cryptanalysis of existing crypto systems.

In the last few years, several widescsale attacks by so-called Internet Worms [SPAFF91] have successfully compromised and infected surprisingly-large numbers of Internet-attached hosts. In 2001, The Cooperative Association for Internet Data Analysis [CAIDA2001] reported that the Code Red v2 worm was able to infect over 350,000 hosts in its first 14 hours of operation. The payload of the Code Red worm was mischief: the defacement of the host website with a political message. It was bold, brash, and drew attention to itself nearly immediately.

Consider for a moment, an Internet worm with a darker and ultimately more dangerous purpose: to brute-force cryptanalyse a message, in order to determine the key used with that message. In order for the worm to be successful, it must avoid detection for long enough to build up a significant level of infected systems, in order to have enough aggregate CPU cycles to complete the cryptanalysis. Furthermore, our worm would need to avoid detection for long enough for the cracked key to be useful to the owners of the worm. Recent research [USEN2002] on stealthy worms paints a very dark picture indeed.

Even after such a worm is detected it would be nearly impossible to tell whose key the worm was attacking. Any realistic attack payload will have one or two pieces of ciphertext, and some known plaintext, or probable-plaintext characteristics associated with it; hardly enough data to determine the likely victim.

3. Winner phone home

When a given instance of the worm determines the key, it needs to contact the originator in order to give them the key. It has to do this in such a way as to minimize the probability that the originator will get caught.

One such technique would be for the worm to public-key encrypt the key, under the public key(s) of the originator(s), and place this in some innocuous spot on the website of the compromised host. The worm could also back-propagate so that a number of compromised websites in the topological neighborhood of the worm will also contain the data. The file containing the key would be identified with some unique keyword which the originators occasionally look for using Internet search engines. The worm could make the process more efficient by using the "keyword registry" services of various Internet search engines.

Another approach would be to post a (possibly PGP-encrypted) message to several newsgroups, through an anonymous posting service. Similarly, Internet "chat" services like IRC could be used; indeed there's an emerging tradition of using IRC and similar services for real-time, anonymous, control of worms and viruses.

Any of these methods can be used both to allow the "winning" worm instance to send results and for the originator to send new work for the amassed army of compromised systems.

4. Evaluating the threat

Both Internet growth and CPU performance follow a reasonably predictable doubling interval. Performance of computing hardware appears to still be following Moore's Law, in which performance doubles every 1.5 years. Internet growth appears to be following a doubling period of 3 years.

By establishing a common epoch for both performance and Internet growth, we can easily determine the likely attack time for any given year, based on a purely arithmetic approach.

A simplifying assumption is that it is indeed possible to construct a suitably-stealthy worm and that it can achieve a steady-state infection of about 0.5% of all attached hosts on the Internet.

In 1995, J. Touch, at ISI, published a detailed performance analysis of MD5 [RFC1810]. At that time MD5 software performance peaked at 87mbits/second, or an equivalent of 170,000 single-block MD5 operations per second. In the same year, peak DES performance was about 50,000 setkey/encrypt operations per second.

In 1995, the Internet had about 20,000,000 attached hosts. In 2002, there are a staggering 100,000,000 attached hosts.

A simple C program, given in Appendix A can be used to predict the performance of our hypothetical worm for any given year. Running this program for 2002 gives:

```
Year of estimate: 2002
For a total number of infected hosts of 503968
aggregate performance: MD5 9.79e+11/sec DES 2.88e+11/sec
DES could be cracked in about 1.45 days
```

DES with 8 character passwords could be cracked in 16.29 minutes
MD5 with 64-bit keys could be cracked in about 218.04 days
MD5 with 8 character passwords could be cracked in 4.79 minutes

The numbers given above suggest that an undetected attack against MD5, for a reasonable key length, isn't likely in 2002. A successful attack against DES, however, appears to be a near-certainty.

5. Security Considerations

DES has been shown to be weak in the recent past. The success of the EFF machine, described in [EFF98] shows how a massively-parallel hardware effort can succeed relatively economically. That this level of brute-force cryptanalytic strength could be made available without custom hardware is a sobering thought. It is clear that DES needs to be abandoned; in favor of either 3DES or the newer AES [FIPS197].

The picture for MD5 (when used in simple MAC constructions) is much brighter. For short messages long keys with MD5 are effectively free, computationally, so it makes sense to use the longest architecturally-practical key lengths with MD5.

Operating system software is becoming more complex and the perpetrators of Internet Worms, Viruses, Trojan Horses, and other malware, are becoming more sophisticated. While their aim has largely been widescale vandalism, it seems reasonable to assume that their methods could be turned to a more focussed and perhaps more sinister activity.

As of February 2003, at least one worm, known as W32/Opaserv.A has a payload designed to implement a distributed DES cracker.

6. Acknowledgements

John Morris, of Nortel IS, contributed the idea of anonymous newsgroup posting.

Appendix A: Source Code

```
   /*
    * This program calculates the performance of a hypothetical
    *  "Chinese Lottery" brute-force cryptanalysis worm, based on
    *  the current date, estimates of Internet growth rate and
    *  Moores Law.
    *
    */ #include <stdio.h> #include <math.h> /*
    * EPOCH for the calculations
    */ #define EPOCH 1995.0 /*
    * Size of the Internet (ca 1995)
    */ #define INTERNET_SIZE 20000000.0

   /*
    * Software MD5 performance (ca 1995)
    */ #define MD5PERF 170000.0

   /*
    * Software DES performance (ca 1995)
    */ #define DESPERF 50000.0

   main (argc, argv) int argc; char **argv; {
        double yeardiff;
        double cryptoperf, multiplier;
        double cracktime;

        yeardiff = (double)atoi(argv[1]) - EPOCH;

        /*
         * Moores Law performance double interval is 1.5 years
         */
        cryptoperf = yeardiff / 1.5;
        cryptoperf = pow(2.0, cryptoperf);

        /*
         * Some fuzz here--not all hosts will be the fastest available
         */
        cryptoperf *= 0.450;

        /*
         * Internet size doubling interval is every 3 years
         */
        multiplier = yeardiff / 3.0;
        multiplier = pow(2.0, multiplier);
        multiplier *= (INTERNET_SIZE*0.0050);

        fprintf (stderr, "Year of estimate: %d\n", atoi(argv[1]));
```

```
         fprintf (stdout, "For a total number of infected hosts of %d\n",
              (int)multiplier);
        fprintf (stdout, "aggregate performance: MD5 %5.2e/sec DES
             %5.2e/sec\n",
             MD5PERF*cryptoperf*multiplier,
             DESPERF*cryptoperf*multiplier);

        cracktime = pow(2.0, 55.0);
        cracktime /= (DESPERF*cryptoperf*multiplier);
        fprintf (stdout,
             "DES could be cracked in about %3.2lf days\n",
             cracktime/86400.0);

        cracktime = pow(2.0, 8.0*6.0);
        cracktime /= (DESPERF*cryptoperf*multiplier);
        fprintf (stdout,
             "DES with 8 character passwords could be cracked in
             %3.2lf minutes\n",cracktime/60);

        cracktime = pow(2.0, 64.0);
        cracktime /= (MD5PERF*cryptoperf*multiplier);
        fprintf (stdout,
             "MD5 with 64-bit keys could be cracked in about
             %3.2lf days\n", cracktime/86400.0);

        cracktime = pow(2.0, 8.0*6.0);
        cracktime /= (MD5PERF*cryptoperf*multiplier);
        fprintf (stdout,
             "MD5 with 8 character passwords could be cracked in
             %3.2lf minutes\n", cracktime/60); }
```

Normative References

[DESMEDT91] "Chinese Lotto as an Exhaustive Code-Breaking Machine". J. Quisquater, Y. Desmedt, Computer, v. 24, n. 11, Nov 1991, pp. 14-22.

[RFC1810] Touch, J., "Report on MD5 Performance", RFC 1810, June 1995.

[EFF98] "Cracking DES: Secrets of Encryption Research, Wiretap Politics & Chip Design", Electronic Frontier Foundation, 1998.

[CAIDA2001] "CAIDA Analysis of Code Red" http://www.caida.org/analysis/security/code-red/

[SPAFF91] "The Internet Worm Program: An Analysis", Eugene Spafford, 1991.

[FIPS197] "Advanced Encryption Standard", US FIPS197, November, 2001.

[USEN2002] "How to Own the Internet in Your Spare Time", Proc. 11th Usenix Security Symposium, 2002.

Author's Address

Marcus D. Leech
Nortel Networks
P.O. Box 3511, Station C
Ottawa, ON
Canada, K1Y 4H7

Phone: +1 613-763-9145
EMail: mleech@nortelnetworks.com

Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.