第36卷 第11期 计 算 机 工 程 2010年6月
Vol.36 No.11 Computer Engineering June 2010
·安全技术· 文章编号：1000—3428(2010)11—0154—02 文献标识码：A 中图分类号：TP301.6

基于GPU的MD5高速解密算法的实现

乐德广[1]，常晋义[1]，刘祥南[2,3]，郭东辉[2]
(1. 常熟理工学院计算机科学与工程学院，苏州 215500；2. 厦门大学信息科学与技术学院，厦门 361005；
3. 厦门美亚柏科资讯科技有限公司，厦门 361008)

摘 要：MD5快速碰撞算法由于不支持逆向过程而无法在MD5密码攻击中得到实际应用。针对上述问题，通过分析基于图形处理单元(GPU)的MD5密码并行攻击算法原理，设计基于GPU的MD5高速解密算法，在此基础上实现一个MD5高速密码攻击系统。测试结果证明，该算法能有效加快MD5密码破解速度。

关键词：MD5算法；密码学；图形处理单元

Implementation of MD5 Fast Decryption Algorithm Based on Graphic Processing Unit

LE De-guang[1], CHANG Jin-yi[1], LIU Xiang-nan[2,3], GUO Dong-hui[2]
(1. School of Computer Science and Engineering, Changshu Institute of Science and Technology, Suzhou 215500;
2. School of Information Science and Technology, Xiamen University, Xiamen 361005; 3. Xiamen Meiah Pico IT Co., Ltd., Xiamen 361008)

【Abstract】Fast MD5 collision algorithm falls to be used in real application of cracking MD5 password because it does not support reverse cracking. According to the issue, by analyzing the principle of MD5 password parallel cracking algorithm based on Graphic Processing Unit(GPU), this paper proposes a fast MD5 decryption algorithm based on GPU, and implements a fast MD5 password cracking system. Test result proves that the algorithm can accelerate the cracking of MD5 password.

【Key words】MD5 algorithm; cryptography; Graphic Processing Unit(GPU)

1 概述

最近，文献[1]提出的基于模减差分分析的MD5[2]碰撞攻击方法重新引起了广大学者对MD5的研究兴趣。然而，由于该MD5碰撞攻击方法不能直接由MD5散列值逆向得出明文，因此这种攻击方法在实际MD5密码破解中还无法得到很好的应用。本文分析了当前MD5密码攻击中存在的安全性问题，提出一种MD5密码并行攻击算法，并基于图形处理单元(Graphic Processing Unit, GPU)设计和实现了MD5高速密码破解系统。该系统能充分利用GPU的高性能计算能力，对MD5散列值进行大规模并行密码分析，从而快速破解出经MD5加密的密码，因此，在计算机安全与取证[3]中具有一定的实际应用意义。

2 MD5算法安全性分析

一般破解MD5需要3个基本条件：(1)对于任意y，找x，使得MD5(x)=y。(2)给定x1，找x2，使得MD5(x1)=MD5(x2)。(3)找x1、x2，使得MD5(x1)=MD5(x2)。任何能够满足上述3个条件之一的方法称为MD5破解。目前还没有一个算法能满足条件(1)。文献[1]给出了符合条件(2)或条件(3)的碰撞算法，即可以很快找到2条信息，使之产生相同的MD5散列值。这种基于碰撞的MD5密码破解方法是在知道x1的情况下可以找到x2，但是x2的内容具有不确定性。也就是说，即使知道了x1的内容，仍然无法将x1的内容篡改为有意义的x2，使x2和x1的MD5值相同。因此，它无法直接威胁当前MD5密码的安全体系。

目前对MD5密码的直接攻击有穷举法或彩虹法[4-5]。穷举法是把可能出现的密码明文用MD5运算后，把得到的散列值直接与需要破解的MD5散列值进行比较，判断该明文是否是已知MD5散列值所对应的密码明文。当密码空间确定时，穷举法的破解效率取决于计算机的运算性能。彩虹法是在穷举法的基础上把得到的MD5散列值和原始的明文数据构成一对一的映射表，然后通过搜索和匹配的方式从映射表中找出破解密码所对应的原始明文。当明文数据的随机性很强时，彩虹法要求海量的存储设备来存储映射表和高效的搜索引擎，因此，其破解效率取决于计算机的存储空间和搜索方式。

3 基于GPU的MD5高速解密算法设计与实现

3.1 基于GPU的MD5密码攻击算法

在传统的MD5密码破解中，基于CPU的密码遍历是串行的循环计算过程，因此，其效率和计算速度都非常低。本文为此提出一种MD5密码并行攻击算法。图1显示了基于GPU的MD5密码并行攻击算法原理。在CPU中调用GPU内核程序后，GPU通过其内部的线程执行管理器控制密码生成器同时产生多个可能的不同密码。将这些同时产生的多个不同的密码输入GPU的不同线程后，这些线程就可以对不同的密码进行MD5并行计算，并与已知的MD5散列值进行分析比较，最后输出比较结果。通常GPU的线程数量能够达到十万的数量级，因此，GPU的密码破解具有很高的并行计算效率。

作者简介：乐德广(1975－)，男，博士，主研方向：网络通信，信息安全；常晋义、刘祥南、郭东辉，教授

收稿日期：2009-11-24 **E-mail：**ledeguang@gmail.com

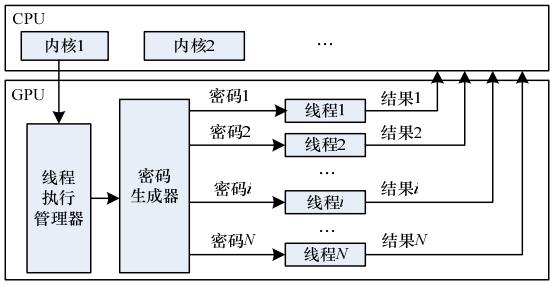

图 1　基于 GPU 的 MD5 密码并行攻击算法原理

3.2　基于 GPU 的 MD5 密码破解系统实现

基于上述算法原理，本文开发了 GPU 高速 MD5 密码破解系统。该系统采用 NVIDIA G80 GPU 芯片，软件开发平台使用 CUDA[6]。其实现过程如下：

(1)设置 GPU 内核环境变量

在 MD5 密码破解中，需要在 GPU 内核中设置密码相关变量，包括：密码字符集，密码字符集长度，密码长度，待破解的密码散列数量，密码变量指针。这些环境变量的具体设置如下：

```
__constant__ unsigned char g_CharSet[255];
__constant__ unsigned long g_CharSetLength;
__constant__ unsigned long g_PasswordLength;
__constant__ unsigned long g_Md5HashCountr;
__constant__ unsigned long g_Plain[96];
```

(2)GPU 初始化

不同的设备具有不同的 GPU 硬件资源，如 GPU 数量、流处理器数量以及内存大小。为了能够充分利用 GPU 硬件资源，需要在 GPU 初始化中获取设备的 GPU 内核信息。本文的破解系统通过以下方式获取设备的 GPU 内核信息：

```
int GPU_N;
cudaGetDeviceCount(&GPU_N);
cudaDeviceProp DeviceProp;
cudaGetDeviceProperties(&DeviceProp, 0);
```

其中，变量 GPU_N 用来存储当前系统中 GPU 的数量；变量 DeviceProp 用来显示 GPU 内核的相关信息，如流处理器数量、存储器容量。

根据已获得的 GPU 硬件信息，为 GPU 内核函数的参数变量分配相应的 GPU 内存空间，包括已知的 MD5 散列值、比较结果变量和每轮内核调用中新的密码变量指针：

```
cudaMalloc((void**)&d_CompMd5Hash, Md5HashCountr*16 );
cudaMalloc((void**)&d_Result, 4 );
cudaMalloc((void**)&d_Next, sizeof(Next));
```

最后通过函数 cudaMemcpyToSymbol()把密码字符集、密码字符集长度、密码长度以及散列数量等数据从 CPU 拷贝到 GPU：

```
cudaMemcpyToSymbol(g_CharSet, &CharSet, CharSetLength);
cudaMemcpyToSymbol(g_CharSetLen, &CharSetLength, 4);
cudaMemcpyToSymbol(g_PasswordLength,&PasswordLength, 4);
cudaMemcpyToSymbol(g_Md5HashCountr,&Md5HashCountr, 4);
```

(3)产生测试密码

由于 GPU 线程是并行执行的，因此需要为每个线程分配不同的测试密码。本文的破解系统通过一个密码生成器来实现，其实现伪代码如下：

```
p=t_id;
for i 从 0 到(g_PasswordLength-1){
  tmp = p + g_plain[i];
  p =取整(tmp/g_CharSetLength);
  q =取余(tmp/g_CharSetLength);
  Password[i] = g_CharSet[q];}
```

其中，变量 t_id 为 GPU 中的线程 ID 号；g_PasswordLength 为密码的长度；g_plain[i]为密码的第 i 位对应于密码字符集的指针位置；变量 Password 为线程 t_id 所对应的密码。

(4)实现 MD5 密码并行攻击算法

本文在 GPU 的内核代码中实现了 3.1 节的 MD5 密码并行攻击算法，其伪代码如下：

```
password=password+0x80;
data[0-63]=password;
for i 从 0 到 63
{   if 0≤i≤15       f (data);
   else if   16≤i≤31       g(data);
   else if   32≤i≤47       h(data);
   else if   48≤i≤63       i(data);
}
```

(5)GPU 内核调用

由于 GPU 的程序是以内核方式运行的，因此需要在 CPU 中调用 GPU 内核函数。与普通的 C 语言函数只执行一次的方式不同，在调用 GPU 的内核函数时，它将由 N 个不同的线程并行执行 N 次。执行内核的每个线程都会被分配一个唯一的线程 ID。其调用方式如下：

```
crackmd5<<<block,thread>>>(hash);
```

其中，crackmd5 为 GPU 内核函数名。参数 block 用于指定内核的栅格维数和大小，即每个栅格内线程块的数量。为了不使流处理器组在线程同步过程中出现空闲，设置线程块的数量是 GPU 中流处理器组数量的 4 倍。这样线程块在线程同步等待时就能使其他线程块继续运行，充分利用流处理器组的计算资源。参数 thread 用于指定每个线程块的维数和大小，即每个线程块内的线程数量，它取决于每个流处理器组的存储器资源。假设每个流处理器组的寄存器总数是 R，每个线程需要占用的寄存器数量是 M，B 是每个流处理器组线程块的数量，那么每个线程块的线程数量 $T=32\times\mathrm{int}(R/(B\times M))$，其中，int 为取整函数。因此，每个 GPU 内核程序的并行线程总数等于每个线程块的线程数量乘以线程块的数量。内核函数 crackmd5 的参数 hash 表示已知的 MD5 散列值。

(6)分析比较及结果输出

根据第(4)步测试密码的 MD5 运算结果及第(5)步 GPU 内核调用中提供的已知 MD5 散列值，GPU 线程通过比较分析可以判断其测试密码是否为已知 MD5 散列值所对应的密码。其实现伪代码如下：

```
if( data[0-15] = hash[0-15] )
  if( data[16-31] = hash[16-31] )
    if( data[32-47] = hash[32-47] )
      if( data[48-63] = hash[48-63] )
            *cracked_password = password;
```

3.3　性能测试及分析

下面对 2 种 MD5 破解系统进行测试和比较分析。在本文的 MD5 破解系统中，CPU 采用最新的 Intel Core 2 Quad Q9300 四核中央处理器，内存为 4 GB，操作系为 Windows XP。此外配置了 4 张 GeForce 9800 GX2 的 GPU 显卡。图 2 显示了 MD5 密码破解速度的比较结果。其中，Core2 Quad Q9300 表示采用 Intel 四核 CPU 的工作站，其 MD5 破解速度为每秒 9.6×10^6 个密码；4*9800GX2 表示采用本文算法的 GPU 工作

（下转第 158 页）

用 RSA 算法分别对 512 位的数据进行加密、解密、签名和验证，测得所需的时间为 0.03 s, 0.16 s, 0.16 s 和 0.02 s。在该认证机制中，设消息的长度为 512 位，对称密钥算法为 DSA，不计 DSA 运算时间(DSA 比 RSA 快 100 倍)，完成一次跨域认证所需的时间大概为 1.1 s，远小于 DAA 认证的时间开销，有效地减少了跨域认证所需的时间。

2.5 仿真分析

本文使用了 Matlab 绘制了 2.2 节中信任值公式中 Trust(B→A)的变化过程。图 3 为 Trust(B→A)在不同 α, β, γ 比例下的变化曲线。

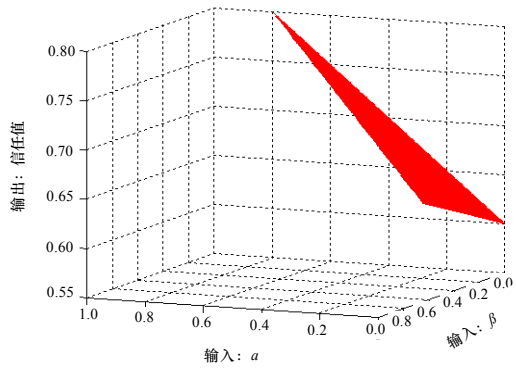

图 3　Trust(B→A)在不同分配比例下的变化曲线

假设在正常情况下，域 B 对 TPM 用户的信任值为 0.8，对域 A 的直接信任值为 0.65，其他域对域 A 的推荐信任值为 0.6。图 3 中的深色区域即为 *Trust*(B→A)在不同比例下的变化曲线图。当 α=1 时，Trust(B→A)取得最大值 0.8；当 γ=1 时，Trust(B→A)取得最小值 0.6。所以远程域可以根据侧重点的不同，动态地调节 α, β, γ 的比例，根据 Trust(B→A)的变化来制定合适的信任阈值。

图 4 为 Trust(B→A)在跨域交互过程中的变化曲线。其中，α=0.4, β=0.4, γ=0.2，其他域的推荐信任值为 0.6，图 4 中的深色区域即为该状态下 Trust(B→A)的取值变化范围。随着 T(B→A)的逐渐增加，在 TPM 用户完全可信时，Trust(B→A)达到的最大值为 0.92。

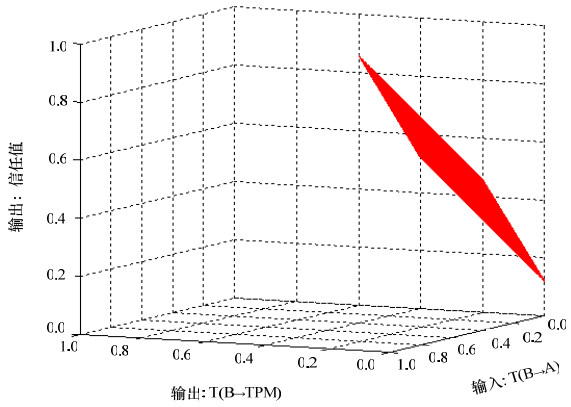

图 4　Trust(B→A)在交互过程中的变化曲线

3　结束语

本文提出了一种基于动态信任值的 DAA 跨域认证机制，通过建立域间的信任关系来实现 TPM 用户的跨域认证，同时又减少了认证的时间开销。下一步工作将使用协议仿真工具模拟实现该机制的认证流程。

参考文献

[1] Trusted Computing Group. Trusted Computing Platform Alliance (TCPA) Main Specification Version 1.1b[EB/OL]. (2002-02-10). https://www.trustedcomputinggroup.org.

[2] Brickell E, Camenisch J, Chen Liqun. Direct Anonymous Attestation[C]//Proc. of the 11th ACM Conference on Computer and Communications Security. New York, USA: ACM Press, 2004: 132-145.

[3] 黄辰林. 动态信任关系建模和管理技术研究[D]. 长沙: 国防科技大学, 2005.

[4] 裴俐春, 陈性元, 王　婷, 等. 一种基于信任度的跨异构域动态认证机制[J]. 计算机应用, 2008, 28(6): 1382-1384.

[5] 张艳群, 张　辰. 基于模糊理论的信任度评估模型[J]. 计算机工程与设计, 2007, 28(3): 532-534.

编辑　金胡考

(上接第 155 页)

站，其 MD5 平均破解速度超过每秒 30 亿个密码，速度是单 CPU 破解服务器的 300 倍以上。

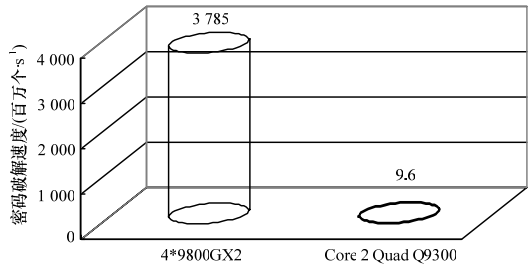

图 2　MD5 密码破解速度比较

4　结束语

针对 MD5 快速碰撞在密码攻击中存在的问题，本文提出一种 MD5 密码并行攻击算法，基于该算法设计实现了一个 GPU 高速 MD5 密码破解系统，测试结果证明该系统能有效提高密码破解速度。今后将进一步优化 GPU 的 MD5 密码破解系统，并研究其他加密算法(如 SHA-1、AES)的高速破解方法。

参考文献

[1] Wang Xiaoyun, Yu Hongbo. How to Break MD5 and Other Hash Functions[C]//Proceedings of CRYPTO'05. Aarhus, Denmark: [s. n.], 2005.

[2] 张建伟, 李　鑫, 张梅峰. 基于 MD5 算法的身份鉴别技术的研究与实现[J]. 计算机工程, 2003, 29(4): 118-119.

[3] 曾剑平, 郭东辉. 一种有效支持计算机取证的审计机制研究[J]. 计算机工程, 2006, 32(6): 148-150.

[4] Avoine G, Junod P, Oechslin P. Characterization and Improvement of Time-memory Trade-off Based on Perfect Tables[J]. ACM Trans. on Information and System Security, 2008, 11(4): 1-22.

[5] 张丽丽, 张玉清. 基于分布式计算的暴力破解分组密码算[J]. 计算机工程, 2008, 34(13): 121-123.

[6] VIDIA. CUDA[EB/OL]. (2009-03-30). http://www.nvidia.cn/object/cuda_home_cn.html.

编辑　张　帆