MD5碰撞攻击实验室

1介绍

一个安全的单向哈希函数需要满足两个特性：单向特性和抗碰撞特性。单向属性确保给定一个哈希值h，找到一个输入M在计算上是不可行的，这样的哈希(M)=h。抗碰撞特性保证了找到两个不同的输入在计算上是不可行的M1和M2，从而生成散列（M1）=散列（M2）。

一些广泛使用的单向哈希函数很难保持抗碰撞性能。在2004年的加密货币的臀部会议上，王晓云和合著者展示了对MD5[1]的碰撞攻击。2017年2月，CWI阿姆斯特丹和谷歌研究公司宣布了冲击攻击，打破了SHA1[3]的抗碰撞特性。虽然许多学生在理解单向属性的重要性方面没有什么困难，但他们不容易理解为什么抗碰撞特性是必要的，以及这些攻击会造成什么影响。

本实验室的学习目标是让学生真正了解碰撞攻击的影响，并亲眼了解，如果一个广泛使用的单向哈希函数的抗碰撞特性被破坏，会造成什么损害。为了实现这个目标，学生需要对MD5哈希函数发起实际的碰撞攻击。使用这些攻击，学生应该能够创建两个不同的程序，共享相同的MD5散列，但有完全不同的行为。本实验室涵盖了以下所描述的一些主题：

单向哈希函数，MD5
抗碰撞性能
沙痂碰撞袭击

阅读对单向哈希函数的详细介绍如下：

《种子书》第22章，《计算机与互联网安全：一个实际动手的方法》，第二版，作者：杜文亮。
详情请访问https://www。handsonsecurity.网

实验室环境。这个实验室已经在我们预构建的Ubuntu20.04VM上进行了测试，可以从SEED网站下载。该实验室使用了一个名为“快速MD5碰撞生成”的工具，这是由马克·史蒂文斯编写的。二进制文件的名称在虚拟机中称为md5colgen，它安装在/usr/bin文件夹中。如果它不在那里，你可以从实验室的网站上下载它(在实验室设置内。zip).如果您有兴趣将该工具安装到您自己的机器上，您可以直接从https://www下载源代码。赢tue.nl/hashclash/.

这个实验室是在雪城大学电气工程和计算机科学系的研究生维什塔斯普·Jokhi的帮助下开发出来的。

2实验室任务

2.1任务1：使用相同的MD5散列生成两个不同的文件

在此任务中，我们将生成具有相同MD5哈希值的两个不同文件。这两个文件的开头部分需要是相同的，即，它们共享相同的前缀。我们可以使用md5colgen程序来实现这一点，它允许我们提供任何任意内容的前缀文件。该程序的工作方式如图1所示。下面的命令生成两个输出文件，输出1。箱和出2。bin，对于一个给定的前缀文件前缀。txt:

```
$md5聚合前缀。txt -o out1 .退出2。垃圾桶
```

图1：从前缀生成的MD5碰撞

我们可以使用diff命令来检查输出文件是否不同。我们还可以使用md5sum命令来检查每个输出文件的MD5哈希。请参阅以下命令。

```
$diff1。退出2。垃圾桶
$ md5sum out1 .垃圾桶
$ md5sum out2 .垃圾桶
```

自1。箱和出2。bin是二进制的，我们不能使用文本查看器程序来查看它们，比如cat或更多的程序；我们需要使用二进制编辑器来查看（和编辑）它们。我们已经在虚拟机中安装了一个名为buluse的十六进制编辑器软件。请使用这样的编辑器来查看这两个输出文件，并描述您的观察结果。此外，您还应回答以下问题：

- **-问题1。**如果前缀文件的长度不是64的倍数，那么会发生什么呢？
- **-问题2。**创建一个恰好为64字节的前缀文件，然后再次运行碰撞工具，看看会发生什么。
- **-问题3。**md5colgen为这两个输出文件生成的数据（128字节）是否完全不同？请识别所有不同的字节。

.22任务2：了解MD5的属性

在这个任务中，我们将尝试理解MD5算法的一些特性。这些特性对我们在这个实验室进行进一步的任务很重要。MD5是一个相当复杂的算法，但从非常高的层面，它不是那么复杂。如图2所示，MD5将输入数据划分为64个字节的块，然后在这些块上迭代计算散列。MD5算法的核心是一个压缩函数，它接受两个输入，一个64字节的数据块和前一次迭代的结果。压缩函数产生一个128位的IHV，它代表“中间哈希值”；然后将此输出输入到下一个迭代中。如果当前的迭代是最后一个迭代，则IHV将是最终的哈希值。第一次迭代的IHV输入(IHV_0)是一个固定的值。

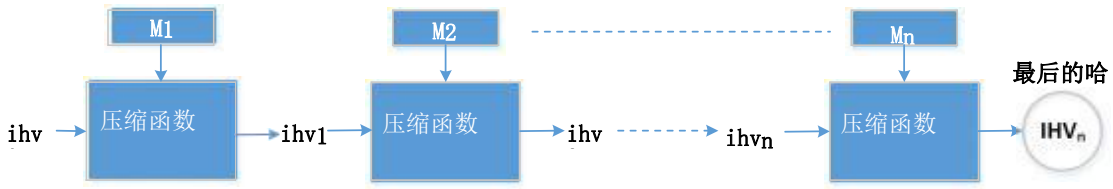

图2：MD5算法的工作原理

基于MD5的工作原理，我们可以得出以下属性的MD5算法：给定两个输入M和N，如果MD5(M)=MD5(N)，也就是说，MD5哈希的M和N是相同的，那么对于任何输入T，MD5(M|T)=MD5(N|T)，其中|表示连接。

也就是说，如果输入M和N具有相同的散列值，那么向它们添加相同的后缀T将得到两个具有相同散列值的输出。这个属性不仅适用于MD5哈希算法，也适用于许多其他哈希算法。你的 工作 在 这 任务是设计一个实验来证明这一性质
持有MD5。

您可以使用cat命令将两个文件（二进制文件或文本文件）连接为一个文件。下面的命令将文件2的内容与文件1的内容连接起来，并将结果放在文件3中。

```
$猫文件1文件2>文件3
```

2.3任务3：生成使用相同的MD5散列生成两个可执行文件

在此任务中，您将得到以下C程序。您的工作是创建这个程序的两个不同版本，这样它们的xyz数组的内容就不同了，但可执行文件的哈希值是相同的。

```
#include <stdio .h>

无符号字xyz[200]={
 /  *这个数组的实际内容由您决定*/
};

int主()
{
  int;
  为(i=0; i<200; i++){
    printf("%x", xyz[i]);
  }
  printf("\n");
}
```

您可以选择在源代码级别工作，即生成上述C程序的两个版本，这样在编译后，它们对应的可执行文件具有相同的MD5哈希值。然而，直接在二进制级别上工作可能更容易。您可以在xyz数组中放一些任意的值，将上面的代码编译为二进制代码。然后，您可以使用十六进制编辑器工具直接在二进制文件中修改xyz数组的内容。

查找数组的内容存储在二进制文件中的位置并不容易。但是，如果我们用一些固定的值来填充数组，我们可以很容易地在二进制文件中找到它们。例如，下面的代码用0x41填充数组，这是字母A的ASCII值。在二进制文件中找到200个a并不难。

```
无符号字xyz[200]={
  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,
  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,
  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,
  ...省略..
  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,  0x41,
}
```

指南。在数组内部，我们可以找到两个位置，在那里我们可以将可执行文件分为三个部分：一个前缀、一个128字节的区域和一个后缀。该前缀的长度需要是64个字节的倍数。有关该文件的分割方式的说明，请参见图3。

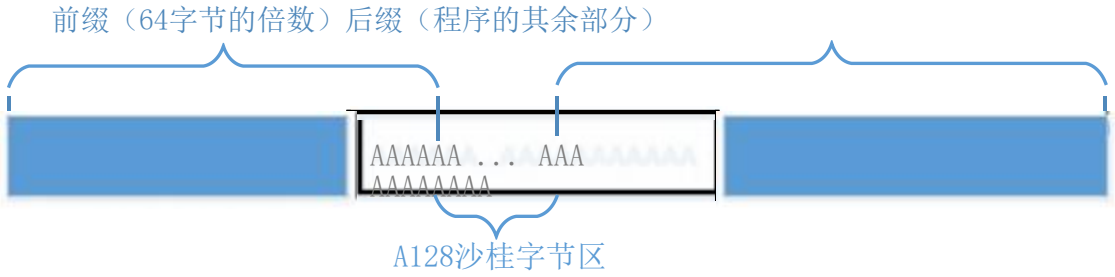

图3：将可执行文件分成三部分。

我们可以在前缀上运行md5colgen来生成两个具有相同MD5哈希值的输出。让我们使用P和Q来表示这些输出的第二部分（每个部分有128个字节）（即，前缀后面的部分）。因此，我们有以下内容：

```
MD5(前缀|P)=MD5(前缀|Q)
```

基于MD5的属性，我们知道，如果我们在上面的两个输出中附加相同的后缀，所得到的数据也将具有相同的哈希值。基本上，以下内容对任何后缀都适用：

```
MD5(前缀|P|后缀)=MD5(前缀|Q|后缀)
```

因此，我们只需要使用P和Q来替换数组的128个字节（在两个分法点之间），并且我们将能够创建两个具有相同哈希值的二进制程序。它们的结果是不同的，因为它们每个人都打印出自己的数组，其中有不同的内容。

工具。您可以使用bluss来查看二进制可执行文件，并找到数组的位置。对于分割一个二进制文件，我们可以使用一些工具来从一个特定的位置分割一个文件。头部和尾部的命令就是如此有用的工具。你可以看看他们的手册来学习如何使用它们。我们给出以下三个例子：

```
$头-c3200a。输出>前缀
$尾巴-c100a。输出>后缀
$尾翼——c+3300a。输出>后缀
```

上面的第一个命令将a.out的前3200字节保存为前缀。第二个命令保存a的最后100个字节。输出到后缀。第三个命令将数据从第3300个字节保存到

文件的结尾是a。输出到后缀。使用这两个命令，我们可以从任意位置分割出一个二进制文件。如果我们需要把一些碎片粘在一起，我们可以使用cat命令。

如果你使用bulese复制粘贴的数据从一个二进制文件到另一个文件，菜单项“Edit->选择范围”是非常方便，因为你可以使用起点和范围选择一个数据块，而不是手动计算选择多少字节。

.42任务4：让这两个程序表现得不同

在前面的任务中，我们成功地创建了两个具有相同MD5散列的程序，但它们的行为不同。然而，它们之间的差异仅在于它们打印出的数据；它们仍然执行相同的指令序列。在这个任务中，我们希望实现一些更重要、更有意义的事情。

假设你已经创建了一个能做些好事的软件。您将软件发送给一个受信任的权威机构以获得认证。当局对您的软件进行了全面的测试，并得出结论，您的软件确实在做好事。当局会向你提供一份证书，说明你的程序是好的。为了防止您在获得证书后更改程序，您的程序的MD5哈希值也包含在证书中；证书由权威机构签名，因此在不使签名无效的情况下，您不能更改证书或程序上的任何内容。

你希望你的恶意软件被当局认证，但如果你只是简单地将你的恶意软件发送给当局，你就没有机会实现这个目标。但是，您已经注意到，权限使用MD5来生成哈希值。你有个主意。你计划准备两个不同的程序。一个程序总是会执行良性指令并做好的事情，而另一个程序则会执行恶意指令并造成损害。你可以找到一种让这两个程序被分享的方法
相同的MD5哈希值。

然后，您将良性版本发送给权威机构进行认证。由于这个版本做了一些好事，所以它将通过认证，并且您将获得一个包含良性程序的哈希值的证书。由于您的恶意程序具有相同的哈希值，因此此证书对您的恶意程序也有效。因此，您已成功地获得了恶意程序的有效证书。如果其他人信任当局颁发的证书，他们就会下载你的恶意程序。

这个客观的的这任务是启动上述攻击。也就是说，您需要创建两个共享相同的MD5散列的程序。但是，一个程序将总是执行良性指令，而另一个程序将执行恶意指令。在您的工作中，执行的良性/恶意指令并不重要；这足以证明这两个程序执行的指令是不同的。

指南。创建两个完全不同的程序来产生相同的MD5哈希值是相当困难的。由md5collgen产生的两个哈希碰撞程序需要共享相同的前缀；此外，正如我们从前面的任务中所看到的，如果我们需要向
md5collgen产生的输出添加一些有意义的后缀，那么添加到两个程序中的后缀也需要是相同的。这些都是我们所使用的MD5碰撞生成程序的局限性。尽管还有其他更复杂、更先进的工具可以消除一些限制，比如接受两个不同的前缀[2]，但它们需要更多的计算能力，所以它们超出了这个实验室的范围。我们需要找到一种方法，在限制范围内生成两个不同的程序。

实现上述目标的方法有很多种。我们提供了一种方法作为参考，但我们鼓励学生们提出他们自己的想法。教师可能会考虑奖励学生，因为他们有自己的想法。在我们的方法中，我们创建了两个数组X和Y。我们比较了这两个数组的内容；如果他们

相同时，将执行良性代码，否则，将执行恶意代码。请参阅以下伪代码：

```
阵列X；
数组Y；

主要的
{
  如果（X的内容和Y的内容相同）
       运行良性代码；
  其他的
       运行恶意代码；
  返回
}
```

我们可以用一些值来初始化数组X和Y，这些值可以帮助我们找到它们在可执行的二进制文件中的位置。我们的工作是更改这两个数组的内容，这样我们就可以生成具有相同MD5散列的两个不同版本。在一个版本中，X和Y的内容相同，所以执行良性代码；在另一个版本中，X和Y的内容不同，因此执行恶意代码。我们可以使用一种类似于任务3中使用的技术来实现这个目标。图4说明了该程序的两个版本的外观。

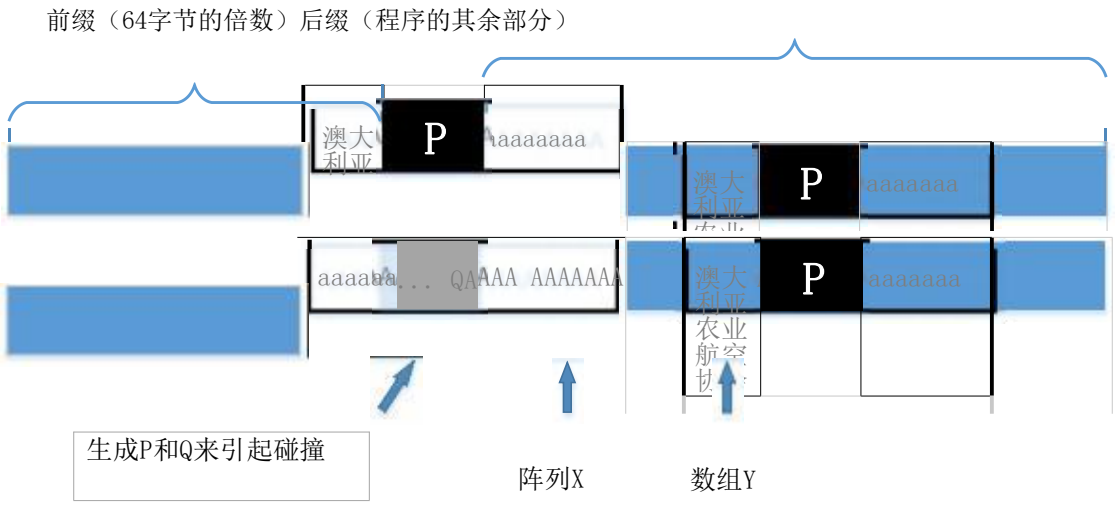

图4：一种生成两个具有不同行为的哈希碰撞程序的方法。

从图4中，我们知道这两个二进制文件具有相同的MD5哈希值，只要P和Q被相应地生成。在第一个版本中，我们使数组X和Y的内容相同，而在第二个版本中，我们使它们的内容不同。因此，我们唯一需要改变的是这两个数组的内容，并且不需要改变程序的逻辑。

3提交

你需要提交一份详细的实验室报告，带有截图，来描述你所做的什么和观察到的什么。你还需要对那些有趣或令人惊讶的观察结果提供解释。也请先列出重要的代码片段，然后进行解释。简单地附加代码而没有任何解释将不会收到学分。