南開大學
Nankai University

计算机学院

密码学基础实验报告

差分密码分析

姓名：章壹程

学号：2313469

专业：计算机科学与技术

2025 年 4 月 5 日

目录

1 前言

上一次写的作业我以为只需要写清楚就行，就交了个比较潦草的纸质版，代码啥的都没贴上去。既然大家都卷一份报告写个好几页，那我也不做人了。因为书上只是介绍算法而几乎没有讲解成立的原因，同时我对于这个算法也很感兴趣，所以这次会写的详细些，整理分析一下差分密码的原理。这次也没贴代码，因为代码部分照着伪代码复现就行非常简单，同时还能节约篇幅省钱。

代码和报告开源在https://github.com/u2003yuge/Fundamentals-of-cryptography。

2 差分密码分析原理

对于差分密码分析，我觉得书上写的不够清楚，因此我将基于对 SPN 的分析用我自己的逻辑重新理一下，分析过程缺乏严谨的数学定义与推理，请见谅。

2.1 SPN

我之前对 SPN 的理解有问题，实际上 **SPN 的 S 盒和 P 盒是已知的**，只有 K 是未知的，这是各种攻击手段的基础。

2.2 一轮只解一个密钥

由于有多轮加密，我们不可能做到一轮破解所有的密钥，因此得想办法让大部分密钥加密都失效，一次仅解决一个密钥，那首先考虑的自然是第一段密钥或最后一段密钥。

差分密码分析和线性密码分析都选择从最后一段密钥开始破解。其实从第一段开始也是可以的，因为在 SPN 加密中，**密文和明文是一一对应的**，整个加密和解密过程是对称的，只需要将密文视作明文，明文视作密文，再将整个流程颠倒，那么破解过程是一致的。

我们还是以先解最后一个密钥加密为例，我们该如何绕过其余密钥的干扰？SPN 使用的密钥加密方法为异或，那利用异或的性质 $x \oplus K \oplus K = x$，以及异或的交换律和结合律，我们可以想到构造 $(x_1 \oplus K) \oplus (x_2 \oplus K) = x_1 \oplus x_2$，也就是说，**用密钥加密前后，有序比特对之间的异或值不变**。因此就可以构造有序比特对 (x, x^*)，满足 $x \oplus x^* = x'$，其通过密钥 K 加密后的异或值不变，这就是我们绕过密钥干扰的突破口。

2.3 考虑 S 盒和 P 盒的影响

从对有序比特对的异或值 x' 的影响来说，S 盒会改变异或值中 1 的数量，而 P 盒会改变异或值中 1 的位置，而且**影响是与有序比特对的具体值有关而不是只与异或值相关**。密钥加密虽然不会修改 x'，但会修改有序比特对的值。S 盒和 P 盒虽然已知，但需要明确知道输入的有序比特对才能确定输出的异或值，也就是说，**我们没法独立地使用一对 (x, x^*) 解出密钥的任何信息**，我们需要一些额外的先验知识。

S 盒和 P 盒可以共同视为一个映射，作用域和值域都是有序比特串，并且是一个双射。那么对于有序比特串对 (x, x^*) 来说，x 和 x^* 经过该映射后会得到的 (y, y^*) 仍然是有序比特串并且一定与输入有序比特串一一对应。这样的信息太过冗杂，我们需要将其进行统计压缩，得到与异或值相关的信息。

对于 P 盒只改变异或值中 1 的位置，通过前后的异或值是一一对应关系，我们可以先不研究其影响，而专注研究 S 盒。定义 π_S 为 S 盒，π_P 为 P 盒，$\Delta(x')$ 为包含所有具有输入异或值 x' 的有序对 (x, x^*)。设 $N_D(x', y') = |\{(x, x^*) \in \Delta(x') : \pi_S(x) \oplus \pi_S(x^*) = y'\}|$，更进一步，我们可以算出输入异或

为 x' 的情况下输出异或为 y' 的概率，设有序比特串的长度为 m ，则有概率 $R_p(x',y')=\frac{N_D(x',y')}{2^m}$，其中 $R_p(x',y')$ 也被叫做扩散率。

那么扩散率有什么用呢？设通过 S 盒、P 盒与 K 密钥加密前有序比特串异或值为 x'，加密后为 y' 的概率为 $P(x',y')$，则有 $P(x',y')=R_p(x',t')$，其中 $(x,x^*)\in\Delta(x'),\pi_S(x)\oplus\pi_S(x^*)=t',y=\pi_P(\pi_S(x))\oplus K,y^*=\pi_P(\pi_S(x^*))\oplus K,y\oplus y^*=y'$，也就是说对于加密中任意一轮的 $P(x',y')$ 与 $R_p(x',t')$ 均有相同的分布形状。**更进一步，对于整个 SPN 的加密过程中的任意一轮或多轮，$P(x',y')$ 与 $R_p(x',t')$ 的有相同的分布形状**。我们利用这个性质就可以进行攻击获得密钥。

2.4 攻击方法

$P(x',y')$ 通过大量的 4 元组 (x,x^*,y,y^*) 攻击最终会趋向其概率分布，而 $R_p(x',t')$ 计算直接可得。由上一节最后的结论可以知道，$P(x',y'_{nr})=R_p(x',t'_{nr})$，且 $P(x',y'_{nr+1})=R_p(x',t'_{nr+1})$，其中 y'_{nr} 表示经过前 nr 轮加密后的结果，t'_{nr} 表示经过前 $nr-1$ 轮加密与一个 S 盒后的结果。由于我们不知道 y'_{nr}，我们只能将 y_{nr+1} 与 y^*_{nr+1} 利用我们猜测的密钥 K_{nr+1_guess} 进行解密得到 y_{nr_guess} 与 $y^*_{nr\_guess}$ ，并异或得到 y'_{nr_guess}，**那么当且仅当** $P_{guess}(x',y'_{nr\_guess})=R_D(x',t'_{nr})$**，其中** $t'_{nr}=\pi_S^{-1}(y_{nr\_guess})\oplus\pi_S^{-1}(y^*_{nr\_guess})$ **时，有** $K_{nr+1\_guess}=K_{nr+1}$（仅考虑最后一轮时是没有 P 盒加密的）。

简要证明如下：我们已知 $P(x',y'_{nr})=R_D(x',t'_{nr})$，即 y'_{nr} 与 t'_{nr} 一一对应，也就是某一对 (y_{nr},y^*_{nr}) 只对应 $(t_{nr})'$，又 $y_{nr+1}=y_{nr}\oplus K_{nr+1}$，$y^*_{nr+1}=y^*_{nr}\oplus K_{nr+1}$，我们通过解密得到的 $y_{nr\_guess}=y_{nr}\oplus K_{nr+1}\oplus K_{nr+1\_guess}$ ，$y^*_{nr\_guess}=y_{nr}\oplus K_{nr+1}\oplus K_{nr+1\_guess}$。**要让所有这样的** $guess$ **有序比特串对均仍对应** t'_{nr}**，当且仅当** $K_{nr+1\_gess}=K_{nr+1}$。对于 P 盒，由于其只改变异或值中 1 的位置且与具体有序比特串数值无关，加入它不影响攻击方法的正确性。

2.5 比较分布相似性

最后一个残留的问题为：我们如何验证 $P_{guess}(x',y'_{nr\_guess})=R_D(x',t'_{nr})$？换而言之，我们如何验证这两个分布完全相同？由于等号左右两个分布永远是形状相同的，只需要比较等号左右两边分布的峰值（或者较大值）是否能对齐就有极大的把握认为两个分布相同，这是差分密码分析的设计。**但既然比较分布的相似性，为什么不用 KL 散度？**这将在 2.8 节进行具体分析。

2.6 过滤

在算法中还运用了过滤操作，即将满足一定条件的 4 元组称做正确对并仅使用正确对以消除随机噪音。其实选择特定的输入也可以看作是一个过滤操作。这是个非常有效的操作，除了可以消除随机噪音外，其将可行域大大减小，举书上的例子，它将成立的范围从 $0\sim2^{16}$ 缩小到了 $0\sim2^8$，一方面提升了效率，避免一些无意义的计算计数，另一方面，**减小可行域可以更好地保证所选择的路径在范围内是概率最大路径**，缩小域的范围减少域内元素数量更不容易出现将概率更大的路径选入范围导致选错密钥。

2.7 概率性问题

在前文中，我们强调了多次分布的相似性，**但事实上我们的结论都基于样本量足够大，或者说样本量相较于加密的轮数较大**。只有在样本量够大的情况下，频率才会趋近于概率，从而得到相似的分布。但是由于 SPN 的加密是对称的，输入数据量和输出数据量永远是相等的，随着轮数的增加，分布一定是熵增的（我们也可以依据热力学第二定律得到相同的结论），各项概率会逐渐趋向均匀而稀疏，

而由于各项比较稀疏，再经过新一轮加密后，利用概率推导出来的结果由于样本的稀疏化而不再成立，**从而导致密钥的影响变得至关重要**，最后的分布与概率预测的分布差距很大。

图2.1 可以验证这一点。随着轮数的增加，分布的极差变小，分布的形状会尖突变为扁平，导致各项的频率都不能近似于概率，导致分布的相似性失效，使用不同猜测密钥得出的结果分布大不相同，这点将在 3.2 节结合实验结果再做分析解释。

2.8 KL 散度的缺点

首先，KL 散度计算的复杂度很高。假设猜测的明文对个数为 T，明文长度为 m ，枚举的密钥域大小为 $|K|$，那么 KL 散度计算的时间复杂度为 $O((Tm+2^m)\cdot|K|)$，空间复杂度为 $O(|K|2^m)$，而比较极大值的时间复杂度为 $O(T|K|m)$，空间复杂度为 $O(|K|)$，相比之下代价相当高，不是多项式时间内可解的，只有在小规模问题上可能能发挥作用，但不具有通用性。

其次，当 $|K|$ 比较小的时候，计算出来的交叉熵的值是很容易受到随机误差的影响的。为了保证正确性我们很可能需要大量的样本，这会花费更多的时间。

最后，是输入异或值的选择。枚举所有输入异或值是不可接受的，因此我们在随机在 E 个输入异或值中选取最好的一个。对于比较极大值来说，通过计算 R_D 将问题切分成各轮的子问题，能有效地降低搜索输入异或值的时间消耗；对于熵法来说，**我们要尽可能选择熵值大的分布**（我猜的），但计算熵需要知道各种输出异或值的概率，那么就需要多对明文对进行模拟，根据前文所述，这代价很大。

综上所述，使用 KL 散度的代价很大，并且效果不见得比比较极大值好。

3 实验

3.1 T 值选取

3.1.1 实验准备

利用程序可以找到某一输入异或值下不同输出异或值中的概率最大者，同时也能遍历所有输入异或值找到全部可能输入异或值-输出异或值值中概率最大的，图 2.1 也是用这个程序生成的。

我们选择三个输入-输出异或值对，分别为 $(0000000000000011, 0000101000001010)$ ，$(0000101100000000, 0000011000000110)$，$(1011000010110000, 0000100000001000)$，在密钥设置为全 0（选取输入-输出异或值对的过程可以是离线的，即提前确定输入-输出异或值对以以减少解密时的工作量）的情况下频次分别为 304，1752，2324，分别为弱攻击对，标准攻击对，强攻击对。

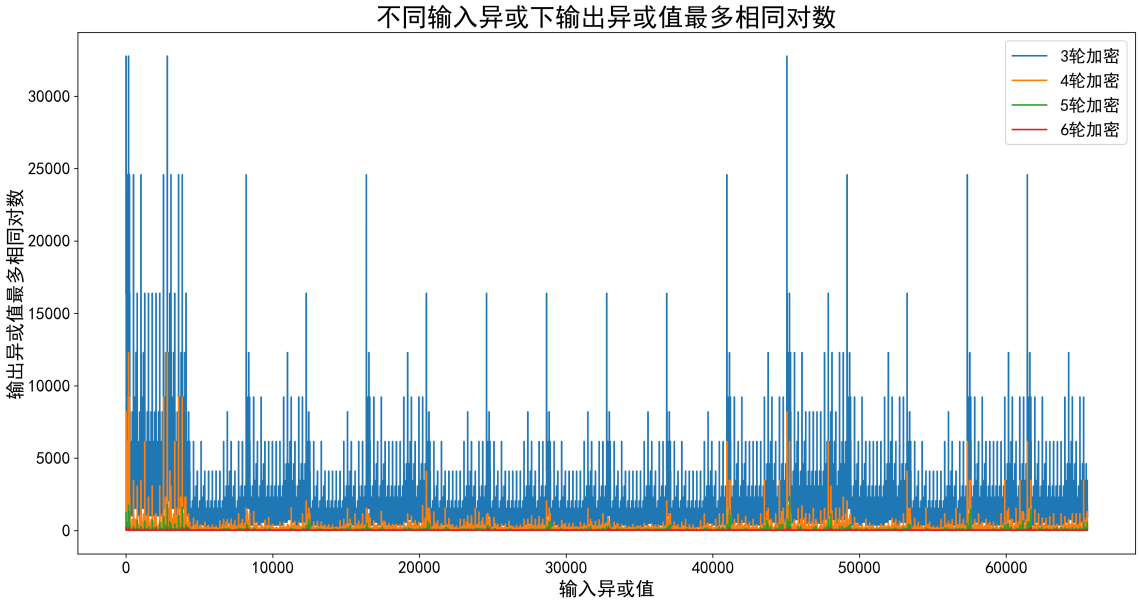
3.1.2 实验内容

实验在 T 取 10 到 1000 时，使用上述三种输入-输出异或对的情况下攻击正确的概率。对于每一个输入个数 T，实验 100 个随机密钥，每个密钥测试 100 组每组 T 个有序比特串。

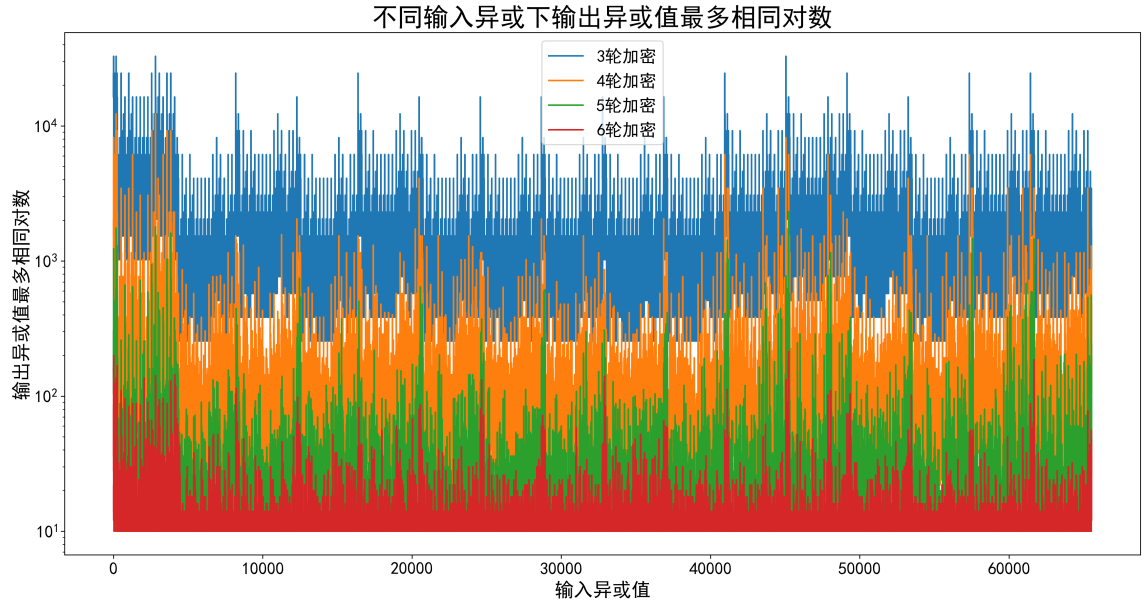
3.1.3 实验结果

实验结果如图3.2，和书上的有差别，如果将通常定义为 95% 正确率的话实际测试需要约 143 对才能保证攻击会成功。

可以看出，选择一个优异的输入-输出异或对非常重要，能有效地减少所需的攻击对数。虽然选择优异的攻击对十分花费时间，但考虑到选取与最终密钥攻击是无关的，我们可以在平时空闲时间预处理计算出一些较优的攻击对并进行记录，在需要破解时可以立即进行使用。



(a) 线性版本



(b) 对数版本

图 2.1: 不同输入异或下输出异或值最多相同对数

统计在除 0 外的不同输入异或值下，输出异或值相同的对数最多有多少对。可以发现，随着加密轮数的上升，统计值呈指数级下降，意味着相同输入异或值下不同输出异或值之间的概率差异大幅减小。

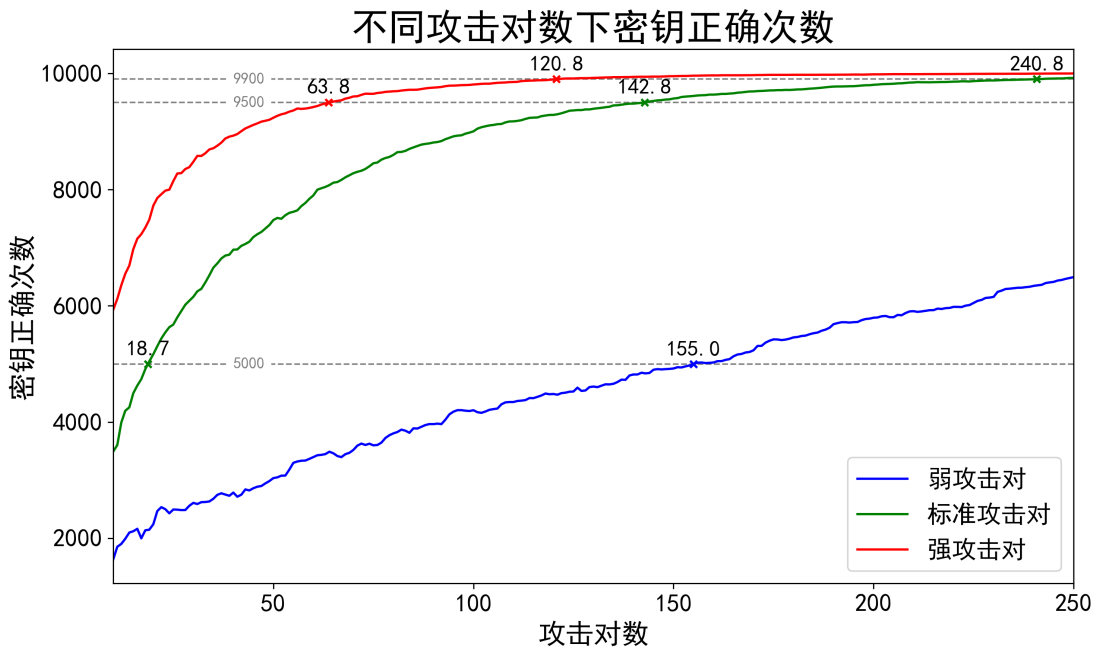


图 3.2: 不同攻击对数下密钥攻击正确次数

3.2 猜测密钥结果分布

3.2.1 实验内容

在输入异或值为 0000101100000000 的情况下，测试了猜测密钥为 1101011000111111，1101011000110000，0000000000000000，即正确密钥、相似猜测（75% 正确）以及猜测全零时输出异或值的频数。

3.2.2 实验结果

实验结果如图 3.3 。从图中可以看出，从正确密钥到全零猜测，分布越来越趋向均匀。可以这么理解，由于 SPN 加解密具有对称性，其实我们做的**解密操作也可以认为是新一轮的加密**，当且仅当猜测密钥和实际密钥重合时，两轮加密相互抵消，成功解开一轮。这就导致使用正确密钥和非正确密钥之间隔了两轮密钥，根据2.7节所述的内容，轮数越多分布的熵值越高，分布就越扁平。

还有个有趣的地方在于，相似密钥的分布与正确的密钥相似性较高，也有一些尖突的地方。因此可以猜测**使用不同猜测密钥尝试解密后的所有分布之间是连续的，极值点为真实密钥**。这意味着我们可能可以用一些机器学习甚至深度学习的办法破解密钥，甚至拟合整个加密过程。但这属于未来工作的内容了，我没太多精力与兴趣研究了。

3.3 心得体会

3.3.1 熵

进行本次实验最大的收获就是对熵的理解。

熵值越高，越难预测。随着加密轮数的增加熵值会越高，分布会越均匀，但加密的代价会上升；而解密就是找到熵值不高的突破口，并利用其带来的某些可预测特征，攻击获得密钥。

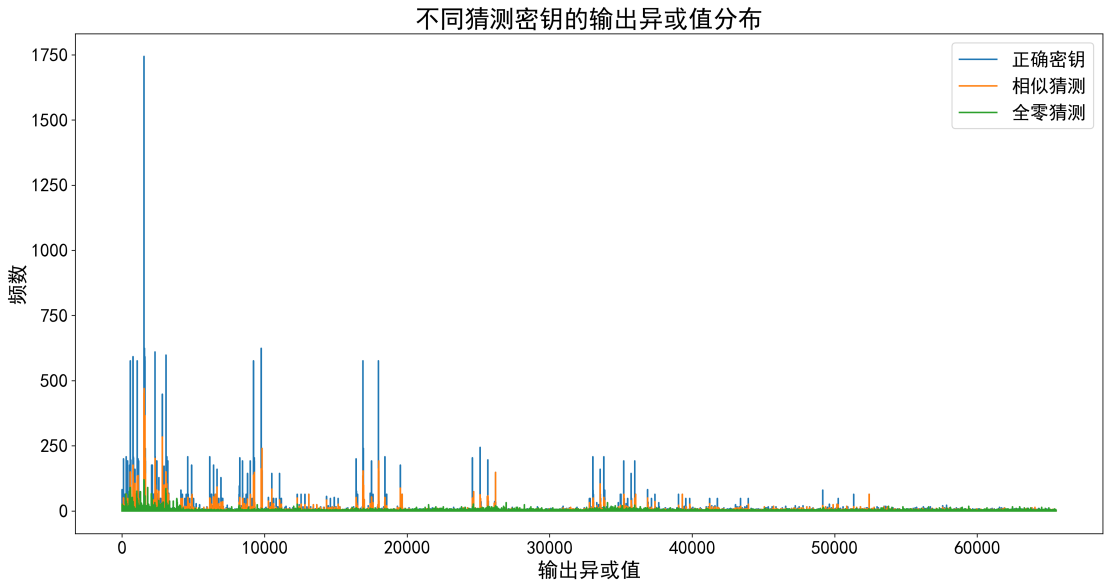


图 3.3: 不同猜测密钥的输出异或值分布

加密熵增，解密熵减。在加减密对称的情况下，如果你的操作成功地让分布熵减，那说明你的解密方法在向正确的方向靠近，从无序之中找到了有序。

3.3.2 理论与实践相结合

差分密码书上写的不符合我的口味就自己从结果倒推了下逻辑，证明其正确性。在证明过程中能衍生出一些有意思的猜想，但光猜想是不具有说服力的，因此还得自己设计实验。实验的结果有些符合预期，还有大多数不符合预期。用不符合预期的部分可以推翻之前的假设，从而提出更合理的假设，然后再做实验，如此反复。

自己设计实验验证自己的猜想还蛮有意思的，**如何让实验结果具有说服力**，这是十分具有挑战，也十分有趣的地方。

3.4 后记

github 上虽然有开源代码，但几乎未经过整理，仅简单改了下名称。其目的只是为了验证工作是我一人所作，以及表示我的实验结果有复现的可能性。我也不相信会有人会把代码下下来研究。需要额外说明的是，作图的程序是用 AI 写的，我稍微改了改调整了下位置（这种格式固定且逻辑简单的代码一行行敲实在过于无聊）。

由于作图需要彩色，必须得选择彩色打印，**很贵**，希望下次能直接交电子档。