课程的两个设计路线

1.怎么从算法层面加强app的安全性。

2.怎么识别、分析、破解协议。

和大学课堂里密码学课程的区别

大学课堂里是80%数学理论，20%编程实践。

我们是20%甚至更少的数学理论，取而代之的是动手实践，让大家不用管数学原理就能掌握这些算法。第二步是赏析和研究其工程实践，第三步是分析、识别其逆向特征，第四步是实战分析，第五步是新工具新思路的运用。

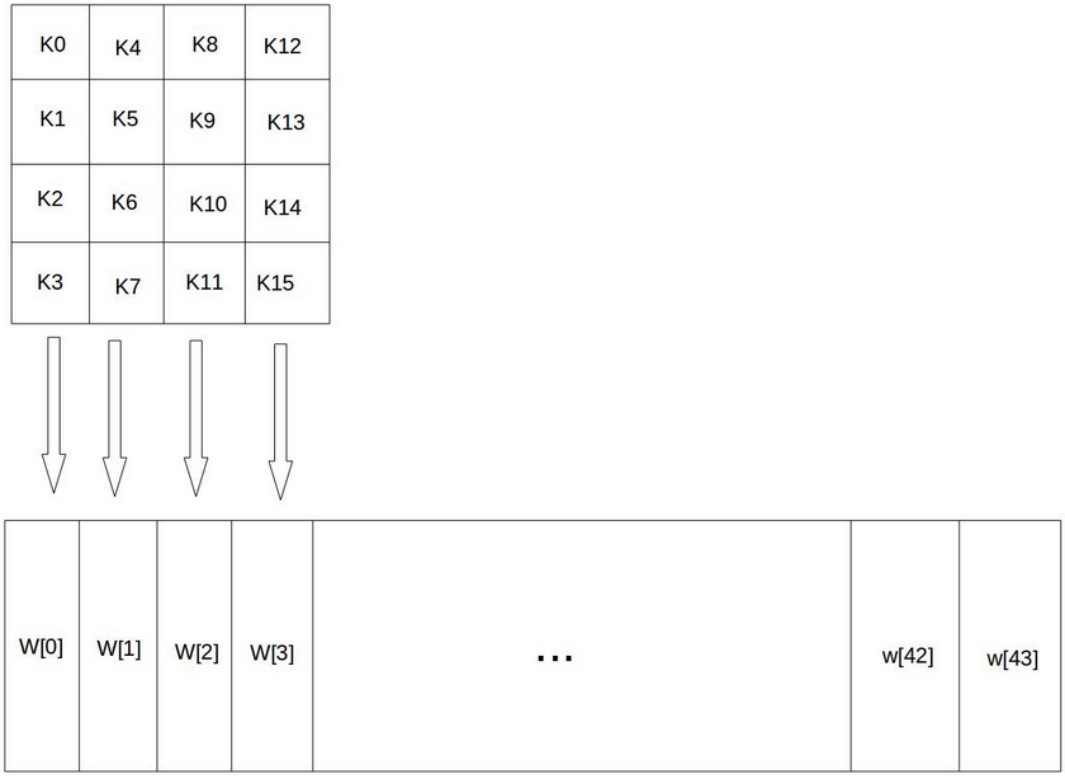
我们更容易理解，更丰富、更适合大家的需求。

“如果使用一些算法识别插件，就算识别出来了是AES，你还是会觉得很茫然，心中不会有一种很爽朗的感觉，因为你不理解这个算法。这也是我想深入学习AES算法的原因。”

一旦出了问题，没能解密，你就慌了，哪里出问题了？密钥？工作模式？IV？明文？魔改？？？

AES 很特殊的一个地方是————处理的最小单位是字节，而主要的操作单元是4*4的矩阵

密钥的编排



1.如果i不是4的倍数，那么第i列由如下等式确定：
W[i]=W[i-4]⊕W[i-1]
2.如果i是4的倍数，那么第i列由如下等式确定：
W[i]=W[i-4]⊕T(W[i-1])
其中，T是一个有点复杂的函数。
函数T由3部分组成：字循环、字节代换和轮常量异或，这3部分的作用分别如下。
a.字循环：将1个字中的4个字节循环左移1个字节。即将输入字[b0, b1, b2, b3]变换成[b1,b2,b3,b0]。
b.字节代换：对字循环的结果使用S盒进行字节代换。
c.轮常量异或：将前两步的结果同轮常量Rcon[j]进行异或，其中j表示轮数。
轮常量Rcon[j]是一个字，其值见下表。

W4 =

http://flashplayer.fullstacks.net/

非常好的整体演示

现在还有两个细节没搞清楚

1是密钥编排中的Rcon

2是明文运算中的列混合

列混淆是AES的难点

S盒是对称加密算法安全性的保障

最差的目标：

搞清楚上面两个环节的特征，不去深究其原理和道理，然后迅速开始理解工程实践代码。

AES 和 DES 的区别与联系

首先说联系，从某些方面上，你可以说aes和des很相近

1.DES和AES都是对称加密算法，加解密都使用同一个密钥

2.DES和AES都是分组加密算法

3.DES和AES的运算整体上都可以分成密钥的编排和明文的运算

4.DES和AES在密钥的编排中，都通过对密钥的处理生成一系列的“subkey”，即子密钥，且每轮运算使用一个。DES有16轮运算，在密钥编排中生成了16个子密钥。AES有10，12，14轮运算，也生成10，12，14个子密钥。都是一轮用一个。

5.AES的10轮运算中，每轮经过四个步骤，第一步叫根据S盒查表替换，在一个16*16的S盒中查找。而DES中也存在S盒，但它是4*16的小S盒，共8个，而AES是一个大的S盒。第二步是循环左移，DES又或者哈希算法中都有它的痕迹。第三步不去说它，第四步和密钥异或，DES中同样有这一步。

那么不同的地方有哪些呢?

1.DES的分组长度是64比特，AES的分组长度是128比特.

2.DES的密钥长64比特，AES的密钥有三种规格，128比特，192比特，以及256比特。

3.在运算中，DES的运算的基本单元是比特，而AES运算的基本单元是字节（8比特），且组织成了矩阵的形式。

4.AES充满了数论的知识，相比DES，AES是纯数学的产物。

5.AES和DES的运算结构不同，DES基于Fesital网络，AES基于SPN。

6.DES中有非常多的置换，AES没有。

必须要把AES 的工程实现放到一个非常高的位置上去，这很重要

1是因为AES实在是太重要太常见了

2是因为AES的实现路径和库太多了

上午跑通AES tiny ndk

AES 和 DES 的区别

http://bristolcrypto.blogspot.com/2015/01/52-things-number-17-describe-and.html

适合作为AES和DES都讲完后的小结。

https://tinyniko.github.io/2021/02/20/aes-enhance/

AES 算法识别，除此之外，提了一嘴白盒AES。

AES算法之理论与编程结合篇.pdf

非常好的资料，亮点是AES算法的两种实现方式——基于算法描述和基于查表。

如何确认加密方式为AES加密?

https://bbs.pediy.com/thread-170860.htm

这个答案有两重

第一重的内容是寻常意义上的，来自于冰冰老师的思路

通过重放工具来确认是否是aes算法

如果不管明文多少，密文始终是16个字节的倍数，那就是aes。通过重放攻击验证这一点。

如果两个相同分组的输入，输出相同，那就是ecb，否则就不是ecb模式。

第二重是反转

一个像序列密码的情况，结果竟是aes？这是怎么回事？？

AES 算法按字节加密

https://bbs.pediy.com/thread-34562.htm

如果确认一个AES是否魔改？如果魔改了怎么办？

最明显的魔改自然是S表魔改

【实例】

但是，因为我们已经懂了S表干啥用的，逆向分析它自然难不倒我们，直接写逆S表。

如果不是这么明显的魔改，可怎么办？

随机数IV 哈哈哈

https://bbs.pediy.com/thread-221402.htm

AES 逆向分析

https://bbs.pediy.com/thread-263706.htm

能不能用纯unicorn把他跑起来

AES 魔改

被修改的S盒 https://bbs.pediy.com/thread-266410.htm

熟悉算法原理之后，轻轻松松解决

vinjar：不进行封装和模块化

https://bbs.pediy.com/thread-267330.htm

KCTF2020秋季赛 第四题 突破重围writeup

很多人都说了两个问题

SO中没有逆S盒或者解密函数以及密钥扩展函数被魔改，我们来验证一下

AES的两种实现之间的区别

数学演绎法VS查表法

前者运算慢，占用空间小，后者运算快，占用空间大。

数学演绎法的代码非常好懂，那查表法呢？？

查表法实质上，就是将subbyte shiftrow mixcolumn 这些步骤合并成一张大表，直接从表中查询结果，关键要怎么做呢?

演绎法实现列混淆

https://bbs.pediy.com/thread-147205.htm

AES的第三种实现——白盒AES

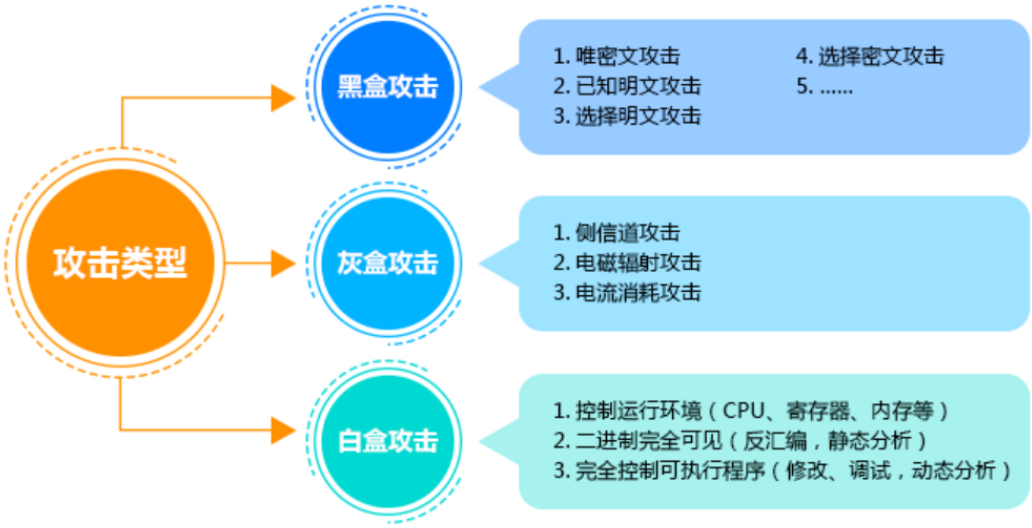
吾爱破解这篇文章看不了

https://www.52pojie.cn/thread-745278-5-1.html

这段话讲的真好，站的真高

动态+静态,大佬果然不同凡响
这也是我对开发者说秘钥不要放在本地的原因,不管怎样,都能被hook
其实破解加密算法本身并没有什么太大的意义,很多标准算法自身都是公开的
自定义算法之类的,只对黑盒加密有意义,像移动端应用之类的准白盒,只是略微增加一丁点破解成本而已

黑白灰盒



梆梆加固的图

黑盒类比于一个JAVA层的加密方法被抽取了，我们只知道输入输出，去猜测这是什么算法，并去验证。

白盒是拥有源码，或者接近源码的情况，比如没有保护的Java层，通过JADX等反编译工具后，接近白盒，IDA+SO，以及Hook、动态调试，沙箱等情况下，也接近白盒，白盒环境是逆向分析的主要环境，我们对运行环境的掌控程度很高。

灰盒是两者之间的一个程度，你无法获取源码环境或者代码执行环境，但是可以获取一些相关的信息，Rainbow可以理解为一个黑盒。

假设一个SO中的函数被高度混淆或者被OLLVM，我们可以用unicorn或者ida trace获取一份汇编执行流，但是无法对其进行任何有用的解析（这点很关键，如果可以解析那就是白盒了），在这种情况下，使用旁道攻击获取到rainbow中呈现的图，然后意识到这是aes。这可以算是灰盒子。

梆梆加固的介绍就是很好的白盒介绍。。

> 梆梆安全密钥白盒加密技术:
>
> 为解决在不直接暴露任何密钥或数据的前提下实现对数据加解密，梆梆安全推出密钥白盒加密保护技术，从根本上防护针对密钥的白盒攻击行为。
>
> **Lookup table转换** 将密钥转换为大量的Lookup table，构造复杂庞大的结构化查找表系统。
> **随机双射编码** 应用随机化、非线性化操作，对查找表进行随机双射编码，隐藏相关内容。
> **算法边界扩展** 将加密算法边界由算法本身扩展到整个程序，实现对密钥的隐藏。
> **内外混编** 采用外部编码加内部编码混合方式，对查找表进行隐藏、混淆和扩散。
> **多层防护措施** 采用多种安全技术集成，增强对密钥白盒环境下非法调用、注入、内存修改的防护。

避免只造轮子不开车

避免只造轮子不开车

避免只造轮子不开车

小米的两篇文章很牛

https://www.anquanke.com/post/id/188340

https://www.anquanke.com/post/id/187028