



- 上一篇: [我的诗歌](#)
- 下一篇: [Java开源建站工具](#)

分类:

- [理解计算机](#)

数字签名是什么？

作者: [阮一峰](#)

今天, 我读到一篇[好文章](#)。

它用图片通俗易懂地解释了, “数字签名”(digital signature) 和 “数字证书”(digital certificate) 到底是什么。

我对这些问题的理解, 一直是模模糊糊的, 很多细节搞不清楚。读完这篇文章后, 发现思路一下子就理清了。为了加深记忆, 我把文字和图片都翻译出来了。

文中涉及的密码学基本知识, 可以参见我以前的[笔记](#)。

=====

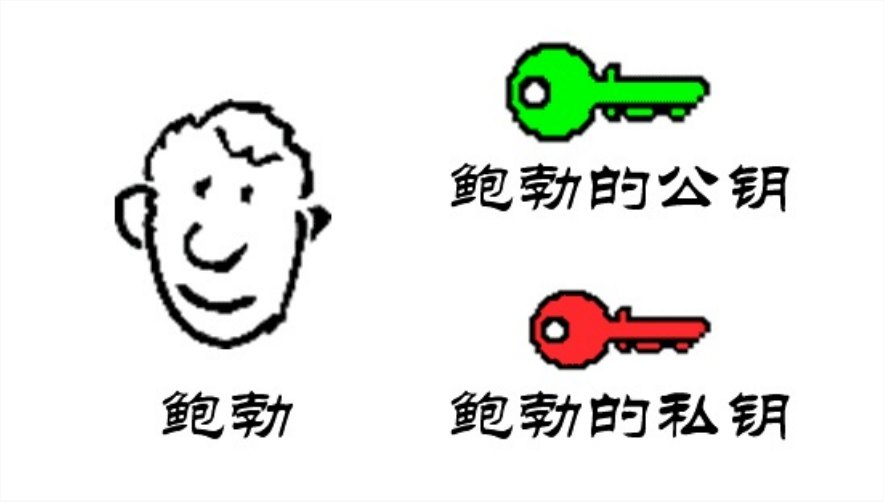
数字签名是什么？

作者: David Youd

翻译: 阮一峰

原文网址: <http://www.youdzone.com/signature.html>

1.



鲍勃有两把钥匙, 一把是公钥, 另一把是私钥。

2.



每人一把

帕蒂 道格 苏珊

鲍勃把公钥送给他的朋友们——帕蒂、道格、苏珊——每人一把。

3.



苏珊

"Hey Bob,
how about
lunch at
Taco Bell. I
hear they
have free
refills!"



公钥加密

HNFmsEm6Un
BejhhyCGKO
KJUxhiygSBC
EiC0QYlh/Hn
3xgiKBcyLK1
UcYiYlxx2lCF
HDC/A

苏珊要给鲍勃写一封保密的信。她写完后用鲍勃的公钥加密，就可以达到保密的效果。

4.



鲍勃

HNFmsEm6Un
BejhhyCGKO
KJUxhiygSBC
EiC0QYlh/Hn
3xgiKBcyLK1
UcYiYlxx2lCF
HDC/A



私钥解密

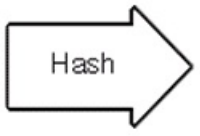
"Hey Bob,
how about
lunch at
Taco Bell. I
hear they
have free
refills!"

鲍勃收信后，用私钥解密，就看到了信件内容。这里要强调的是，只要鲍勃的私钥不泄露，这封信就是安全的，即使落在别人手里，也无法解密。

5.

Try the power of PGP (Pretty Good Privacy), a public-key encryption software package for the protection of electronic mail. Since PGP was published commercially as Shareware in June of 1995, it has spread rapidly all over the world, and has since become the de facto standard for securing e-mail. It is now being used by millions of people to protect their e-mail from prying eyes. The following is a brief overview of a recent investigation by the FBI's Computer Service, which revealed that there were links between PGP and outside the US. That investigation was closed without incident in January 1996.

Computers were developed in secret back in World War II solely to break codes. Ordinary people did not have access to computers because they were too expensive and too complex. Some people pointed out that there would never be a need for more than half a dozen computers in the country, and even if that were the case, people would never have a need for encryption. Some of the government's attitudes toward cryptography today were formed in that period, and to some the old attitudes toward computers were. Why would ordinary people need to have access to good encryption?



Hash

Digest

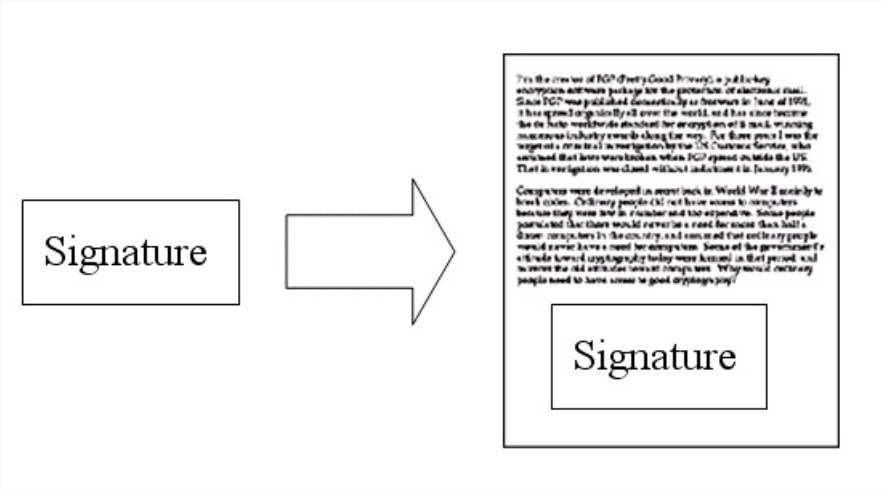
鲍勃给苏珊回信，决定采用“数字签名”。他写完后先用Hash函数，生成信件的摘要（digest）。

6.



然后，鲍勃使用私钥，对这个摘要加密，生成“数字签名”（signature）。

7.



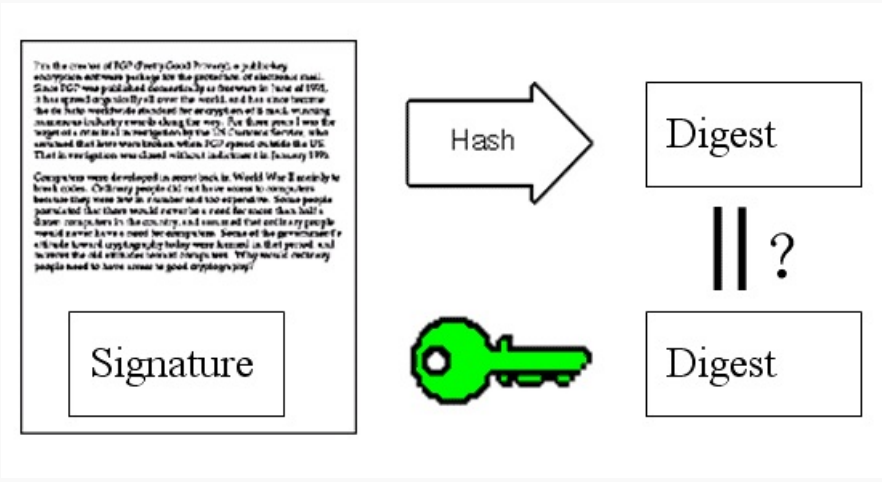
鲍勃将这个签名，附在信件下面，一起发给苏珊。

8.



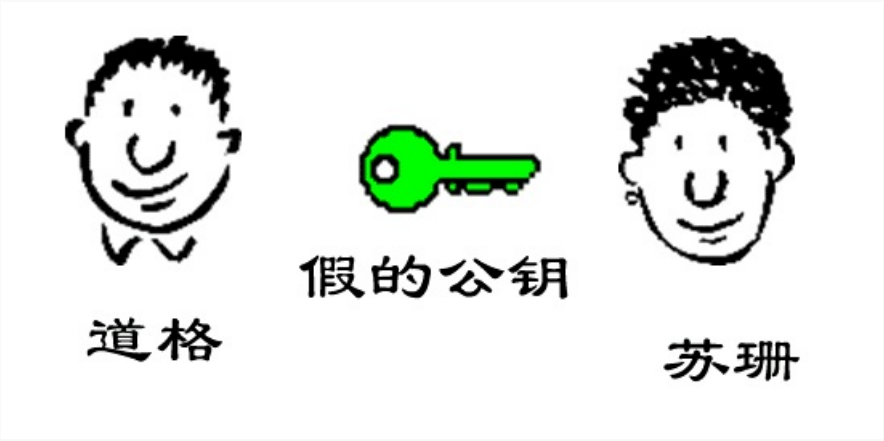
苏珊收信后，取下数字签名，用鲍勃的公钥解密，得到信件的摘要。由此证明，这封信确实是鲍勃发出的。

9.



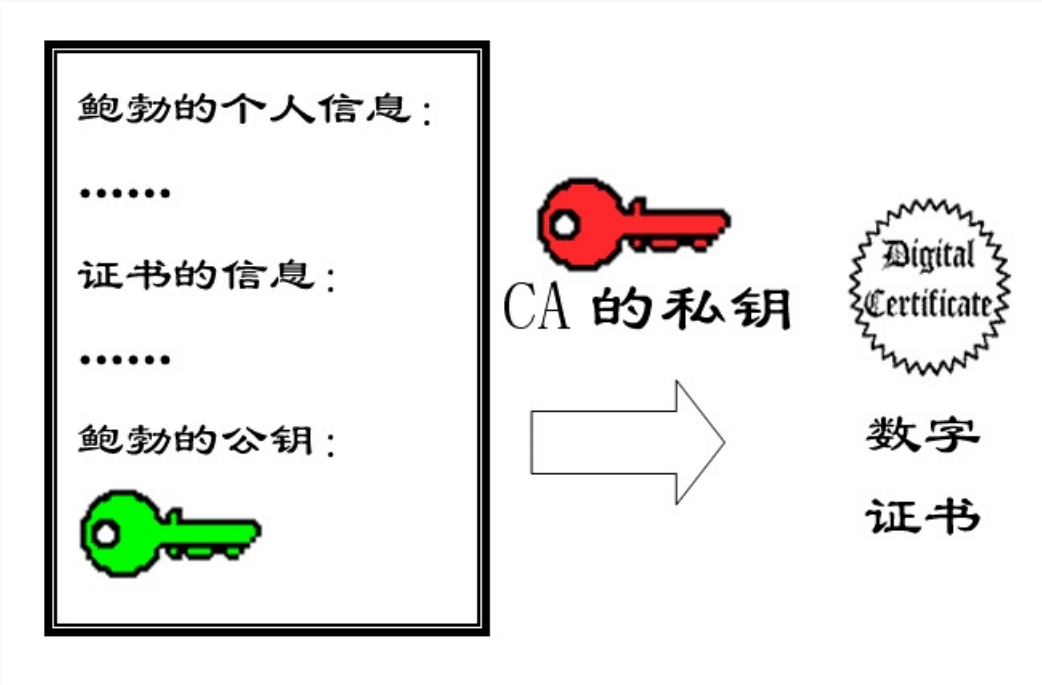
苏珊再对信件本身使用Hash函数，将得到的结果，与上一步得到的摘要进行对比。如果两者一致，就证明这封信未被修改过。

10.



复杂的情况出现了。道格想欺骗苏珊，他偷偷使用了苏珊的电脑，用自己的公钥换走了鲍勃的公钥。此时，苏珊实际拥有的是道格的公钥，但是还以为这是鲍勃的公钥。因此，道格就可以冒充鲍勃，用自己的私钥做成“数字签名”，写信给苏珊，让苏珊用假的鲍勃公钥进行解密。

11.



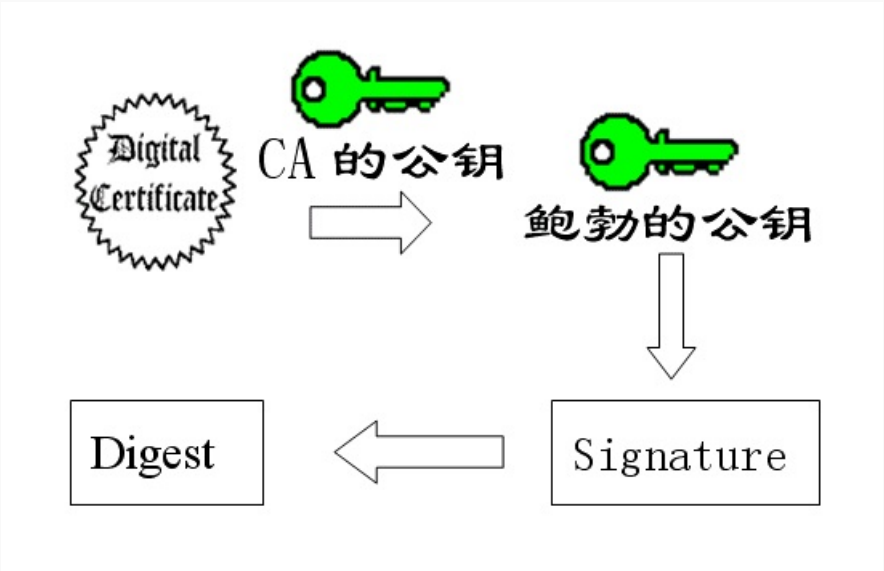
后来，苏珊感觉不对劲，发现自己无法确定公钥是否真的属于鲍勃。她想到了一个办法，要求鲍勃去找“证书中心”（certificate authority，简称CA），为公钥做认证。证书中心用自己的私钥，对鲍勃的公钥和一些相关信息一起加密，生成“数字证书”（Digital Certificate）。

12.



鲍勃拿到数字证书以后，就可以放心了。以后再给苏珊写信，只要在签名的同时，再附上数字证书就行了。

13.



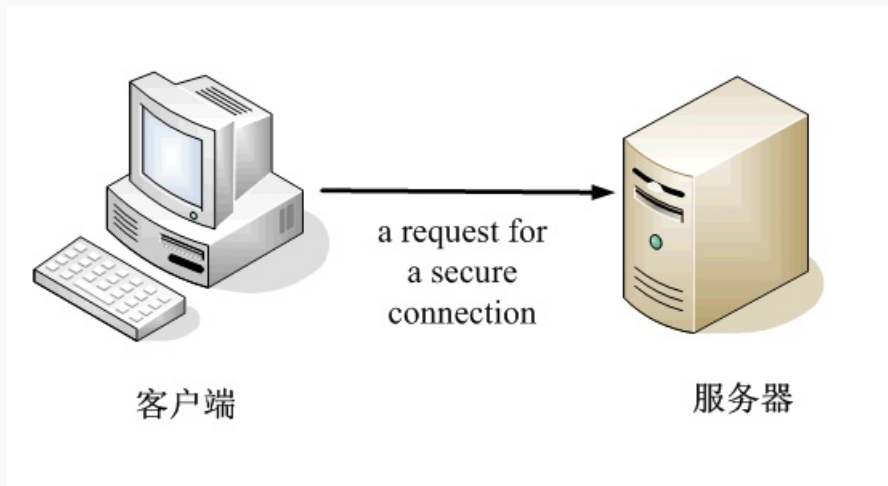
苏珊收信后，用CA的公钥解开数字证书，就可以拿到鲍勃真实的公钥了，然后就能证明“数字签名”是否真的是鲍勃签的。

14.



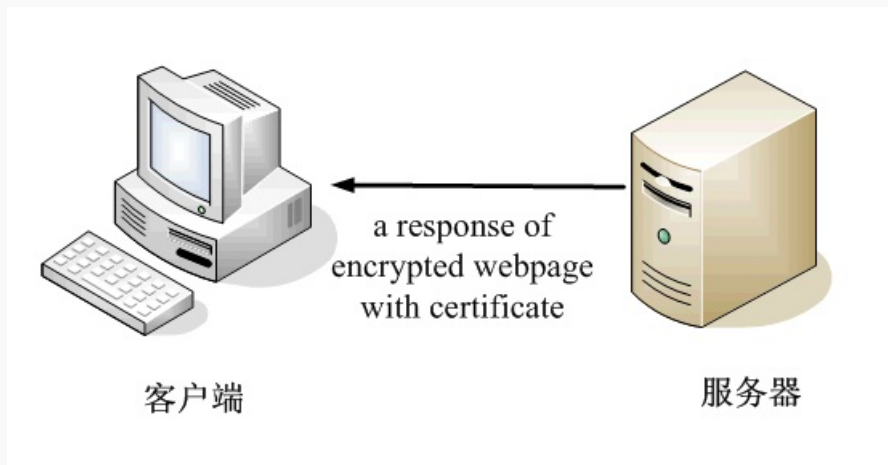
下面，我们看一个应用“数字证书”的实例：https协议。这个协议主要用于网页加密。

15.



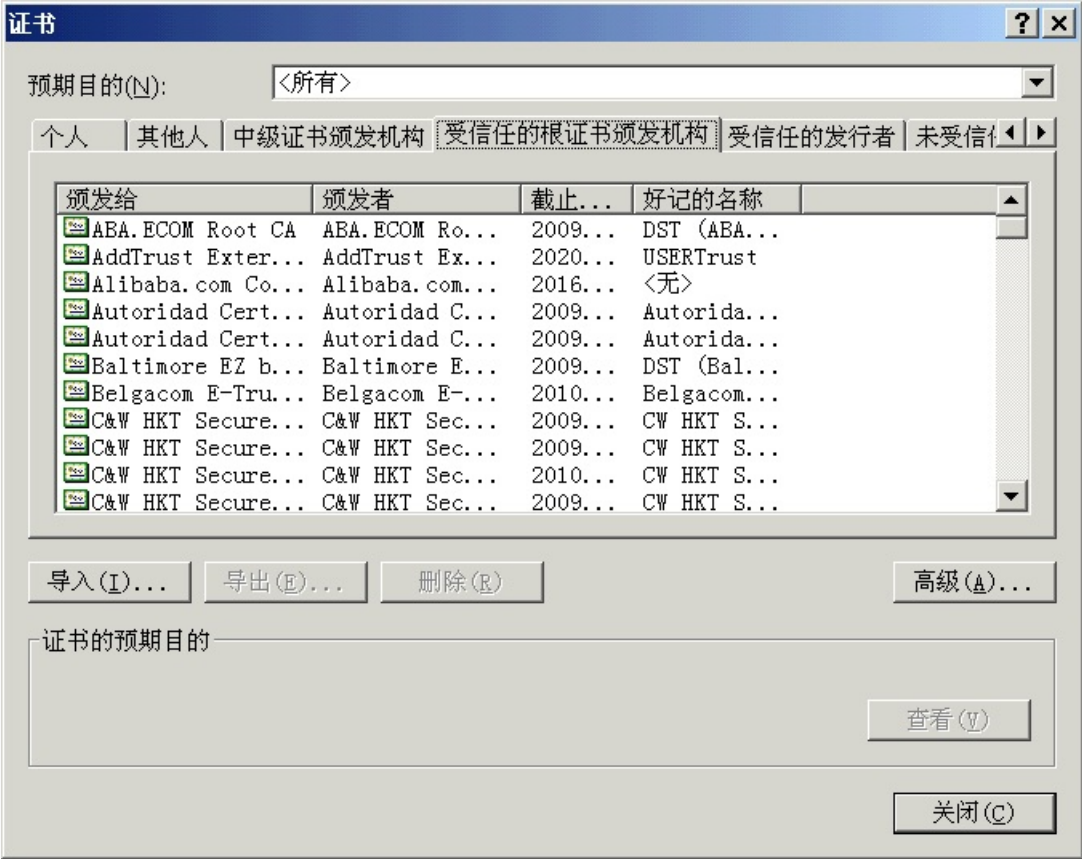
首先，客户端向服务器发出加密请求。

16.



服务器用自己的私钥加密网页以后，连同本身的数字证书，一起发送给客户端。

17.



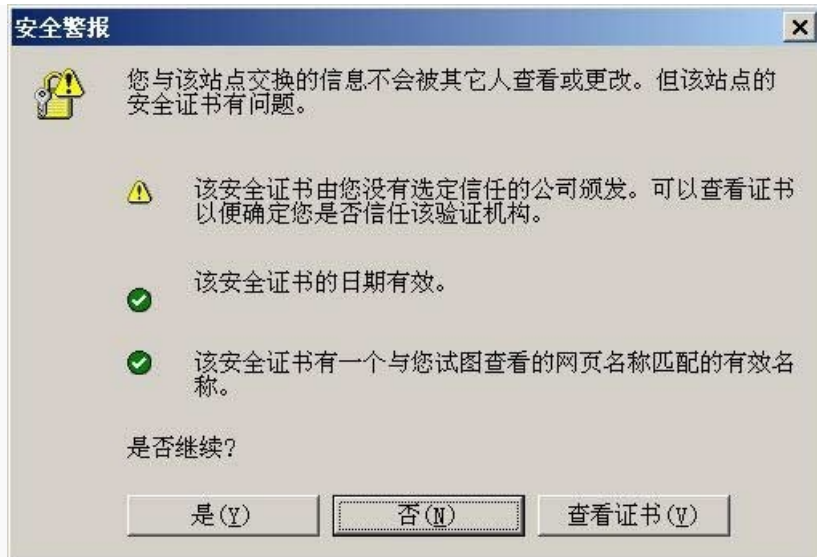
客户端（浏览器）的“证书管理器”，有“受信任的根证书颁发机构”列表。客户端会根据这张列表，查看解开数字证书的公钥是否在列表之内。

18.



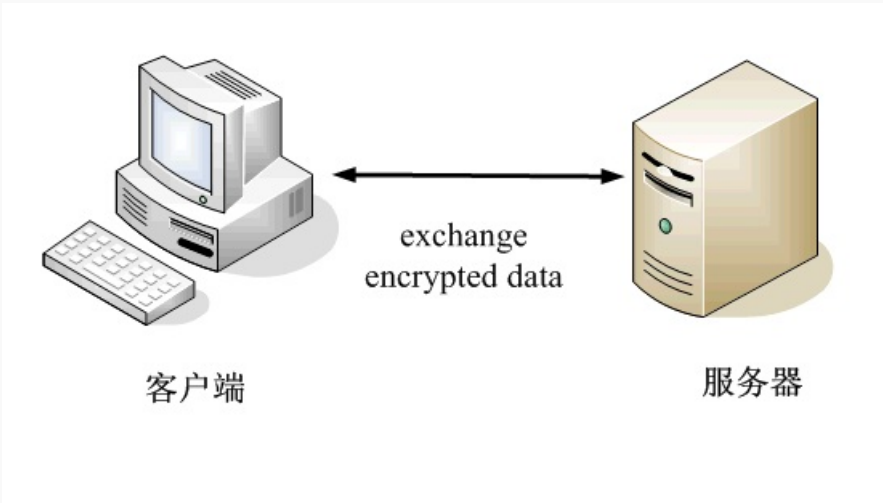
如果数字证书记载的网址，与你正在浏览的网址不一致，就说明这张证书可能被冒用，浏览器会发出警告。

19.



如果这张数字证书不是由受信任的机构颁发的，浏览器会发出另一种警告。

20.



如果数字证书是可靠的，客户端就可以使用证书中的服务器公钥，对信息进行加密，然后与服务器交换加密信息。

(完)

文档信息

- 版权声明：自由转载-非商用-非衍生-保持署名 ([创意共享3.0许可证](#))
- 发表日期： 2011年8月 9日
- 更多内容： [中 档案](#) » [中 理解计算机](#)
- 付费支持： [购物车 购买文集](#)
- 社交媒体： [twitter](#), [weibo](#)
- Feed订阅： [RSS](#)



购买广告位

相关文章

- 2014. 11. 11: [编译器的工作过程](#)
源码要运行，必须先转成二进制的机器码。这是编译器的任务。
- 2014. 09. 07: [数据压缩与信息熵](#)
1992年，美国佐治亚州的WEB Technology公司，宣布做出了重大的技术突破。

- 2014. 07. 04: [数据库的最简单实现](#)
所有应用软件之中，数据库可能是最复杂的。
- 2013. 11. 29: [Stack的三种含义](#)
学习编程的时候，经常会看到stack这个词，它的中文名字叫做“栈”。

留言（124条）

[towrv](#) 说：

太棒了，真的，非常有帮助，谢谢！！

[2011年8月 9日 20:51](#) | [档案](#) | [引用](#)

落水狗 说：

加上图片以后确实清晰了很多，这个真的很好。

[2011年8月 9日 22:14](#) | [档案](#) | [引用](#)

33ad3 说：

说实话，还是有点迷糊

[2011年8月 9日 22:16](#) | [档案](#) | [引用](#)

fenghanzhao 说：

还是有点模糊！没太明白！

[2011年8月 9日 22:29](#) | [档案](#) | [引用](#)

Xtrats 说：

道格想欺骗苏珊，他偷偷使用了苏珊的电脑，用自己的公钥换走了鲍勃的公钥。因此，他就可以冒充鲍勃，写信给苏珊。

有鲍勃的公钥就可以冒充鲍勃？
道格自己不也有鲍勃的公钥么？“每人一把”。

[2011年8月 9日 23:01](#) | [档案](#) | [引用](#)

阮一峰 说：

引用Xtrats的发言：

有鲍勃的公钥就可以冒充鲍勃？
道格自己不也有鲍勃的公钥么？“每人一把”。

只有有了鲍勃的私钥，才能冒充鲍勃。

道格没有鲍勃的私钥，只好伪造鲍勃的公钥。

[2011年8月 9日 23:22](#) | [档案](#) | [引用](#)

[Michael.Z](#) 说：

还是有些模糊，需要慢慢理解。

有一个问题，公钥和私钥的算法是一样的吗？为什么私钥加密可以用公钥解密？

[2011年8月10日 01:36](#) | [档案](#) | [引用](#)

[febird](#) 说：

11.
“证书中心用自己的私钥，对鲍勃的公钥和一些相关信息一起加密，生成“数字证书”（Digital Certificate）。”
13.
“苏珊收信后，用CA的公钥解开数字证书，就可以拿到鲍勃真实的公钥了，然后就能证明“数字签名”是否真的是鲍勃签的。”

有个疑问，鲍勃的证书中鲍勃的公钥有没有被加密？

1. 如果加密了，则苏珊必须能从某个地方获取CA的公钥方能和鲍勃通信，这CA公钥要么随证书附送，要么预先存放在苏珊的电脑中。
2. 如果鲍勃的公钥被加密，同时CA的公钥也在证书中附送，那加密鲍勃公钥有什么意义？
3. 如果鲍勃的公钥被加密，且CA的公钥只是预先存放在苏珊的电脑中，那么经过其他未授权CA颁发的无效证书不能被解开，从而无法得到鲍勃的公钥，但这似乎不合理，举个例子，浏览器能够在HTTPS证书不在证书列表的情况下继续通信。

史诗在线 说:

CNNIC也有根证书了，不过我把它屏蔽了。

blue gene 说:

引用Xtrats的发言:

有鲍勃的公钥就可以冒充鲍勃？
道格自己不也有鲍勃的公钥么？“每人一把”。

道格用自己的私钥加密发给苏珊的信件，苏珊收到信件后用道格的公钥自然能正常解密该信件，但是苏珊以为她收到的是鲍勃的信件，并且认为是用鲍勃的公钥来解密的，自然认为发信的就是鲍勃，所以道格就达到了伪造鲍勃与苏珊通讯的目的。

ncsglz 说:

其实我觉得把三个人名换成中文名，会更容易理解一点，不然容易记不住，哈哈
大学的时候学过这玩意，当时也是一知半解，现在全明白了，太有用了

[ls zhao](#) 说:

绝对的好文章，通俗易懂。分享了... 谢谢

[xiongbo027](#) 说:

既然道格可以替换鲍勃的公钥，为什么不能故技重施，伪造CA的公钥，然后用自己的私钥伪造成CA的数字证书，从而达到欺骗苏珊的目的呢？

newuser 说:

1（前提是权威的CA认证苏珊，保证确实是苏珊为鲍勃创造了数字证书）得先用苏珊自己的公钥检查鲍勃的数字证书，目的是为了证明苏珊创造了鲍勃的数字证书。

2在解密这个数字证书后，再检查是否鲍勃的数字证书由创造它的CA认证以及在创造鲍勃的数字证书时提供的关于鲍勃的相关信息是否发生了改变。

3帕特用通过解密数字证书得到的公匙（这个公匙苏珊为鲍勃创造数字证书所用的鲍勃的公匙）来检查鲍勃的签名，如果这个公匙确实能成功解密签名，证明确实这个签名是由鲍勃的私匙所创造，当然也证明了道格没有修改文档，因为MD没有变化。

道格确实有鲍勃的公匙，不过他用的是鲍勃的电脑和邮件，而且发给苏珊的是自己用鲍勃的姓名生成的key pair。

CK 说:

xiongbo027 说:

既然道格可以替换鲍勃的公钥，为什么不能故技重施，伪造CA的公钥，然后用自己的私钥伪造成CA的数字证书，从而达到欺骗苏珊的目的呢？

=====

其实我也有同上的问题，既然CA的公钥是公开的，那么有什么办法能保证别人无法替换掉CA的公钥呢

Ciger 说:

公钥不需加密。

CA公钥无法伪造，因为CA公钥是可查的，比如在MSDN里可以查到微软用于签名driver的公钥（Base64码）。

阮一峰 说:

引用CK的发言:

既然CA的公钥是公开的，那么有什么办法能保证别人无法替换掉CA的公钥呢

CA都是一些可靠的大机构，它们的公钥在自己网站上提供下载，所以无法伪造。

[2011年8月10日 10:27](#) | [档案](#) | [引用](#)

阮一峰 说：

引用febird的发言：

有个疑问，鲍勃的证书中鲍勃的公钥有没有被加密？

1. 如果加密了，则苏珊必须能从某个地方获取CA的公钥方能和鲍勃通信，这CA公钥要么随证书附送，要么预先存放在苏珊的电脑中。
2. 如果鲍勃的公钥被加密，同时CA的公钥也在证书中附送，那加密鲍勃公钥有什么意义？
3. 如果鲍勃的公钥被加密，且CA的公钥只是预先存放在苏珊的电脑中，那么经过其他未授权CA颁发的无效证书不能被解开，从而无法得到鲍勃的公钥，但这似乎不合理，举个例子，浏览器能够在HTTPS证书不在证书列表的情况下继续通信。

我看到有的资料说，公钥被加密打包后，做成证书。

1. CA的公钥网上可以取得，浏览器中也有预存。
2. 证书必须用CA的私钥加密，如果能用CA的公钥打开，就证明确实是CA颁发的。
3. 我对HTTPS的一些实现细节不熟悉，但是我的理解是，如果HTTPS公钥不在浏览器列表内，浏览器可以从网上取得。

[2011年8月10日 10:40](#) | [档案](#) | [引用](#)

玉沐林注 说：

引用阮一峰的发言：

CA都是一些可靠的大机构，它们的公钥在自己网站上提供下载，所以无法伪造。

那么在下载到本地后，如何避免被替换的问题呢？感觉跟道格用自己的公钥替换鲍勃的公钥一样啊，假若苏珊每次也是重新下载鲍勃的公钥岂不是同样可以避免公钥被替换的问题，那么还要数字证书干什么？

[2011年8月10日 10:46](#) | [档案](#) | [引用](#)

Yonny 说：

引用玉沐林注的发言：

那么在下载到本地后，如何避免被替换的问题呢？感觉跟道格用自己的公钥替换鲍勃的公钥一样啊，假若苏珊每次也是重新下载鲍勃的公钥岂不是同样可以避免公钥被替换的问题，那么还要数字证书干什么？

无法避免。

苏珊必须自己保证自己计算机的物理安全。如果别人已经可以直接控制你的计算机，修改根证书列表，那什么证书安全也救不了你。

[2011年8月10日 11:13](#) | [档案](#) | [引用](#)

Yonny 说：

楼主把数据完整性和数据加密放在一起讲，会让人糊涂。

实际上，数字签名是保证数据完整性的，但它不保证数据加密，不保证数据传输途中无人嗅探窃听。

好比一辆敞篷大货车从A开到B，中途没有洒落任何东西，完整性得到了保证。但是车上有什么东西也被路人看光光。

数据加密是从A到B建了一条虚拟隧道，货车在里面开，路人谁也不知道是什么东西。

车子到了B后，送货的人给出自己的身份证，证明自己的确是从A来的。收货的人可以选择相信这个身份证。也可以把身份证放到自己的身份证校验仪查询，看看是不是公安部发的真的身份证。

如果你的身份证校验仪（CA）已经是假的了，那就啥都别说了，重装系统吧。

[2011年8月10日 11:26](#) | [档案](#) | [引用](#)

Ciger 说：

“如果数字证书是可靠的，客户端就可以使用证书中的服务器公钥，对信息进行加密，然后与服务器交换加密信息。”

这里似乎有点问题。通常公开钥算法用于相互验证，之后会建立session key（比如128位AES key）。后续交互的信息都是用session key和对称加密算法（比如AES）来加解密的，已经与证书本身和公钥密钥无关。因为公开密钥算法比对称密钥算法开销大很多。不过HTTPS不了解，不敢 定论。

[2011年8月10日 11:33](#) | [档案](#) | [引用](#)

gool 说：

- 8. 苏珊收信后，取下数字签名，用鲍勃的公钥解密，得到信件的摘要。由此证明，这封信确实是鲍勃发出的。
- 9. 苏珊再对信件本身使用Hash函数，将得到的结果，与上一步得到的摘要进行对比。如果两者一致，就证明这封信未被修改过。

8 的说法有问题，只此一步，是无法确定信由鲍勃发出的。
必须 8 9 两步都完成了，才能确定：信由鲍勃发出，信未修改。

- 20. 如果数字证书是可靠的，客户端就可以使用证书中的服务器公钥，对信息进行加密，然后与服务器交换加密信息。

Ciger 的说法是正确的。

为了速度起见，https 连接只在建立连接时，使用服务器的公钥加密，这个阶段是为了交换一个共享密钥。接下来的过程使用的是对称算法。

[2011年8月10日 13:18](#) | [档案](#) | [引用](#)

mazhechao 说：

引用Ciger的发言：

通常公开钥算法用于相互验证，之后会建立session key（比如128位AES key）。后续交互的信息都是用session key和对称加密算法（比如AES）来加解密的，已经与证书本身和公钥密钥无关。因为公开密钥算法比对称密钥算法开销大很多。不过HTTPS不了解，不敢 定论。

你说的对，就是这样的。

[2011年8月10日 13:55](#) | [档案](#) | [引用](#)

mazhechao 说：

引用gool的发言：

8 的说法有问题，只此一步，是无法确定信由鲍勃发出的。
必须 8 9 两步都完成了，才能确定：信由鲍勃发出，信未修改。

- 20. 如果数字证书是可靠的，客户端就可以使用证书中的服务器公钥，对信息进行加密，然后与服务器交换加密信息。

8的说法没有问题，就这一步就可以保证信由Bob发出的。因为消息是由Bob的私钥签名的，只有Bob本人才有他的私钥，所以能用Bob的公钥解密的，一定是Bob发出的。
8实现了抗否认性，9实现的是完整性。这是两个不同的概念。

[2011年8月10日 14:03](#) | [档案](#) | [引用](#)

mazhechao 说：

引用Michael.Z的发言：

公钥和私钥的算法是一样的吗？为什么私钥加密可以用公钥解密？

私钥和公钥在算法上是等价的，只不过一个是private，一个是public。这个应该是由密钥生成算法保证的。

[2011年8月10日 14:08](#) | [档案](#) | [引用](#)

mazhechao 说：

引用febird的发言：

有个疑问，鲍勃的证书中鲍勃的公钥有没有被加密？
2. 如果鲍勃的公钥被加密，同时CA的公钥也在证书中附送，那加密鲍勃公钥有什么意义？

确实是被加密的（被CA的私钥加密），但这里的意义不在加密（数据的保密性），而是保证证书是由CA签发的。

- 2. Bob的公钥被用于后续的会话密钥（session key）交换时的加密传输。

[2011年8月10日 14:14](#) | [档案](#) | [引用](#)

ilake 说：

用RSA加解密密的数学原理说明这一过程，可能更容易理解。

[2011年8月10日 14:22](#) | [档案](#) | [引用](#)

[玉沺林注](#) 说：

引用Yonny的发言：

苏珊必须自己保证自己计算机的物理安全。如果别人已经可以直接控制你的计算机，修改根证书列表，那什么证书安全也救不了你。

您说的有道理，我是想确认一下数字证书和数字签名它们各自的作用是什么，因为从阮一峰这篇文章里看到的信息貌似是数字证书是为了确保数字签名的真实性而产生的，但实际情况可能不是这样。

[2011年8月10日 14:30](#) | [档案](#) | [引用](#)

[ilake](#) 说：

<http://zh.wikipedia.org/wiki/%E5%85%AC%E5%BC%80%E5%AF%86%E9%92%A5%E5%8A%A0%E5%AF%86>

维基百科上的说明，比较容易理解。

假设两个用户A，B进行通信，公钥为c, 私钥为d，明文为x.

A用公钥对明文进行加密形成密文c(x)，然后传输密文；
B收到密文，用私钥对密文进行解密d(c(x)), 得到要通信的明文x。

补充：
如果是 A 同时和 B, C 通信，如果C的私钥为e
C收到密文，用私钥对密文进行解密e(c(x)), 得到要通信的明文x。

c 是公钥，d、e 是私钥。用不同的私钥解密，能得到同样的结果。
这个过程通过很巧妙的数学来实现。

[2011年8月10日 14:30](#) | [档案](#) | [引用](#)

gool 说：

@mazhechao 原文的 5 6 7 8 9 是一个完整的签名场景，请在这个场景下重新考虑一遍。然后重新看一下 8 的表述：

8. 苏珊收信后，取下数字签名，用鲍勃的公钥解密，得到信件的摘要。由此证明，这封信确实是鲍勃发出的。

实际上 Bob 的 “数字签名” 可以用任何人的公钥解密，得到一个 hash 值。如果不加上第 9 步（比对原文的 hash值）。这个第 8 步得不到任何有价值的信息。

[2011年8月10日 14:46](#) | [档案](#) | [引用](#)

dindog 说：

我也觉的这篇文章翻译失准了。

[2011年8月10日 15:12](#) | [档案](#) | [引用](#)

ssdt 说：

有ca也没用，cnnic现在在ca里了

cnnic可以伪造别人的ca然后利用窃取的公钥对信解密

公钥也可以被窃取，窃取了不就可以解开信了

把身份授权给ca也没用，只有法治国家才行

[2011年8月10日 16:01](#) | [档案](#) | [引用](#)

ssdt 说：

只要操作系统有后门，什么签名也没用

操作系统的后门可以作为进入一个国家市场的筹码

你以为微软是可以相信的吗

eggcalm 说:

13.
苏珊收信后，用CA的公钥解开数字证书，就可以拿到鲍勃真实的公钥了，然后就能证明“数字签名”是否真的是鲍勃签的。

在这里，CA的公钥会不会被道格用假的CA公钥替换掉？如果可以，那么道格就可以像[10]中那样向苏珊发送假的数字证书，达到冒充鲍勃的目的。

mazhechao 说:

引用gool的发言:

实际上 Bob 的 “数字签名” 可以用任何人的公钥解密，得到一个 hash 值。如果不加上第 9 步（比对原文的 hash值）。这个第 8 步得不到任何有价值的信息。

Bob的签名怎么会可以用任何人的公钥解密？Bob是用他的私钥签的啊，当然只有用Bob的公钥才能解密。再次强调，对比Hash值是为了验证**数据的完整性**。Yonny兄弟说的有道理。

eggcalm 说:

引用gool的发言:

实际上 Bob 的 “数字签名” 可以用任何人的公钥解密，得到一个 hash 值。如果不加上第 9 步（比对原文的 hash值）。这个第 8 步得不到任何有价值的信息。

我不这么认为，我认为第8步的价值就在于，苏珊能确定这封信是鲍勃发出的（虽然不确定信的内容是否被篡改过），因为如果不是使用鲍勃的私钥加密，苏珊使用鲍勃的公钥不可能解密成功。

wqfeng 说:

真不错。后面那个HTTPS的例子是译者加的吧？原文中没有。

gool 说:

引用mazhechao的发言:

Bob的签名怎么会可以用任何人的公钥解密？Bob是用他的私钥签的啊，当然只有用Bob的公钥才能解密。

在原文第 8 步的场景里，所谓的“解密”只是一次数学运算（典型算法RSA）。输入是 signature 和 某人的公钥，输出是一个 hash 值。重申一次，得到的这个 hash 没有任何价值。因为可以用任何人的公钥参与这次运算。

以上可以简单归结为：没有原文的数字签名是没有价值的。

mazhechao 说:

@gool:

价值就在于我能解开这个签名，能够解开这个签名本身就是有意义的——消息是Bob发出的，Bob不能否认消息是他发的，不是Bob的人也不能说是他 自己发的，即实现了数据的抗否认性。至于hash值不hash值的，其意义体现在第9步，实现的是数据的完整性。这是两个完全不同的概念。原文分两步阐述，没有任何问题。

gool 说:

@mazhechao

我不知道你所说的“解开签名”是什么意思，它只是一步数学计算而已。

让我们更细致地看一下“验签”的过程:

- 1: 输入签名和公钥，算出 hash 值 h1;
- 2: 输入原文，算出 hash 值 h2;
- 3: 比较 h1 和 h2，发现 h1 和 h2 相等。在这一步上，我们开始推理，得出原文是 bob 发出且没有修改过。也就是你说的抗抵赖和数据完整。

请注意，这两个有价值的结论都是第三步得出的。

原文第 8 步和你都认为：经过第 8 步的计算，即可以得出原文由 bob 发出的结论。这是错误的，实际上1 2 两步只是可以交换顺序的两个计算步骤，它们不是任何有价值结论的充分条件，连必要条件也不是。

[2011年8月10日 21:07](#) | [档案](#) | [引用](#)

[Lewis](#) 说：

http: the definitive guide 有一章是专门讲 https 原理的，比较透彻。

[2011年8月10日 22:41](#) | [档案](#) | [引用](#)

[雨下路人](#) 说：

总的来说这篇文章是很有价值的，但是翻译的还不让人满意，首先化名用的那几个中文名字不容易区分，容易混淆，鲍勃和道格还 是谐音！ 读着费劲！ 两外最后相关背景没有交代清楚，以及某些语句没有把逻辑关系表达得足够清晰（“道格想欺骗苏珊，他偷偷使用了苏珊的电脑，用自己的公钥换走了鲍勃的公 钥。”起初理解为道格把鲍勃手上的公钥换走了，乱乱的，仔细琢磨语句，才明白意思是：道格用自己的公钥换走了鲍勃送给苏珊的公钥……） 总之我觉得这篇文章如果让更多的人收益，传播的更“远”，还需要好好“返修”一下！ 完。

[2011年8月11日 00:56](#) | [档案](#) | [引用](#)

[xv](#) 说：

引用ncsglz的发言：

其实我觉得把三个人名换成中文名，会更容易理解一点，不然容易记不住，哈哈 大学的时候学过这玩意，当时也是一知半解，现在全明白了，太有用了

同意同意如果用张三李四的名字会更容易绕得清。

[2011年8月11日 04:50](#) | [档案](#) | [引用](#)

[sfumato](#) 说：

最后一段HTTPS的讲解和http://blog.leezhong.com/tech/2011/02/19 /https-workflow.html 上的讲解有出入，无网不剩的讲解HTTPS是用非对称来加密 对称密钥，然后拿对称密钥对网页加密（这样解密速度快），我相信他的解释是正确的，博主能解释 一下吗？

[2011年8月11日 10:16](#) | [档案](#) | [引用](#)

涎弟 说：

鲍勃会给苏珊的信？不加密？明文？第七步只是把signature附上了 原文呢？不做任何操作？？？

[2011年8月11日 10:59](#) | [档案](#) | [引用](#)

Ivan 说：

好文章，需要一点时间来理解

[2011年8月11日 17:29](#) | [档案](#) | [引用](#)

abc 说：

@gool：

看了你的 1， 2 步和前几个回复(实际上 Bob 的 “数字签名” 可以用任何人的公钥解密，得到一个 hash 值。如果不加上第 9 步（比对原文的 hash 值）。这个第 8 步得不到任何有价值的信息。 特别是这个)，好象你不懂公钥和私钥的原理吧，公钥和私钥是一对的，某个私钥加密的内容只有这个私 钥所对应的公钥才能解开，其他的公钥是解不开的, 反之亦 然 不然怎么叫密码学呢

[2011年8月11日 17:30](#) | [档案](#) | [引用](#)

路灯时代 说：

去年有幸上过台湾的信息安全老师讲的数字签名这块的课，是学校搞的什么和台湾学校联合课程。用的是志明和春娇的做为例子。 内容讲的更为详尽，课件应该还在。不过是专门面向密码学研究的内容。

[2011年8月11日 17:47](#) | [档案](#) | [引用](#)

gool 说：

看来不止一个人误会了“解得开”“解不开”的问题。

设想一个签名场景，我们叫“理想中的签名”：Bob 直接用自己的私钥对原文加密，把结果发送给 Alice，Alice 用 Bob 的公钥去解密，得出原文。在

这种情况下，的确存在一个“解开”或“解不开”的问题。如果 Alice 用别的什么人的公钥去解密，这次计算仍然能得出一串符号，但是结果没什么意义，这就是你说的“解不开”。

但实际中为了效率起见，被广泛应用的签名是这样的：Bob 对原文做一次 hash，然后用私钥对 hash 值加密，加密得到的结果我们称之为“签名”，然后把原文与“签名”发给 Alice。

Alice 首先作的计算（也就是原文的第 8 步）是用 Bob 的公钥对“签名”作一次解密，从而得出一个有待于与第 9 步得出的结果相比较的一串符号。

这串符号有什么意义么？

没有。

与原文有什么关系么？

看不出来。

通过这次计算，Alice 能不能断定“签名”是 Bob 发出的呢？

不能。因为**如果用别的什么人的公钥参与这次计算，同样能得到一个符号串，也同样看不出什么意义**。

所以，这一步计算没有所谓的“解得开”“解不开”的问题。

这就是为什么原文第 8 步是错误的，原文说：

收信后，取下数字签名，用鲍勃的公钥解密，得到信件的摘要。由此证明，这封信确实是鲍勃发出的。

Alice 的确得到了一个被暂时当作是原文 hash 值的符号串，但这个符号串还有待接下来的比对。这句话里的因果关系是不成立的。

[2011年8月11日 18:33](#) | [档案](#) | [引用](#)

啊啊呵 说：

@goool：

汗，第八步如果用别人的密钥解密的话会报错，一报错就知道你用的密钥不对了。。。你真的先看看密码学原理吧。。。别人已经说的很详细了，8，9有着分别不同的意义

[2011年8月12日 11:10](#) | [档案](#) | [引用](#)

abc 说：

@goool：

贴个wiki百科上的说明，这就是文章说的第九步，你自己好好理解一下。。。一个是解密得到的值(也就是第八步的值)，一个是自己计算得到的值，再说一边，第八步如果用别人的密钥解密的话是会报错的，也就是得不到结果，而不是你所说的能得到一个hash值，至于加密解密的原理，wiki百科上也有，你可以看一下

签名消息

RSA也可以用来为一个消息署名。假如甲想给乙传递一个署名的消息的话，那么她可以为她的消息计算一个散列值(Message digest)，然后用她的密钥(private key)加密这个散列值并将这个“署名”加在消息的后面。这个消息只有用她的公钥才能被解密。乙获得这个消息后可以用甲的公钥解密这个散列值，然后将这个 数据与他自己为这个消息计算的散列值相比较。假如两者相符的话，那么他就可以知道发信人持有甲的密钥，以及这个消息在传播路径上没有被篡改过。

[2011年8月12日 11:26](#) | [档案](#) | [引用](#)

goool 说：

第八步如果用别人的密钥解密的话是会报错的，也就是得不到结果

以 RSA 算法为例。

拿其它人的公钥去解 Bob 的签名会发生什么，与 RSA 的原理无关，与具体的算法实现有关。从 RSA 的原理来说，所谓“公钥”“私钥”在数学上没有区别，所谓的“加密”、“解密”、“签名”、“验签”本质上是一回事，只是一个乘方和一个取模运算。

报错，报什么错，为什么报错？因为标准实现下，Bob 要对他的信息进行编码和填充。用别的人的公钥解密，会因为填充的字节不对而无法继续计算，或无法从计算结果提取出字符，或与 hash 串规则不符，大多数实现会在此处返回错误或抛出异常。

但是，RSA 从原理上并没有保证这一点，因为它只是三个数字参与的计算而已：把一个数与另一个数作乘方运算，然后除以第三个数，得到余数。

我们完全可以采用另一套字符编码规则、字节补齐规则、以及另外的 hash 算法，让其他人的公钥参与这次计算，也能得到一个符合规则的 hash 值。

所以从逻辑上，Alice 不能作出这样的推理：用 Bob 的公钥对签名作了一次计算，就断定签名是 Bob 发过来的。

[2011年8月12日 16:16](#) | [档案](#) | [引用](#)

[风逐蓝天](#) 说：

不错的科普文，只是感觉还不够通俗易懂。。。。

abc 说：

引用gooo1的发言：

.....

你试一下就知道了啊，用别人的密钥试试看啊，看看会不会报错，你说说的那些什么取模比较数值都在解密也就是第八步里面包含了。。。

Jak 说：

不错不错，很形象~ 菜鸟们稍微了解下就好了。

对于 CA 证书的伪造问题……无可避免，谁知道会不会有木马把证书改掉呢。

[mazhechao](#) 说：

我怎么觉得我开始有点理解gooo1的意思了。。。

[阿迪](#) 说：

引用mazhechao的发言：

我怎么觉得我开始有点理解gooo1的意思了。。。

什么意思啊？

abc 说：

引用阿迪的发言：

什么意思啊？

他的意思就是把8，9步误解成解密的过程，其实第八步解密已经包含取模比较等步骤了

peeekkk 说：

引用阿迪的发言：

什么意思啊？

应该是说，用Alice用随便什么公钥解密，也会有一点点几率解密成功。所以不能认为解密成功，就100%确定签名的正确，有可能中彩票

abc 说：

引用peeekkk的发言：

应该是说，用Alice用随便什么公钥解密，也会有一点点几率解密成功。所以不能认为解密成功，就100%确定签名的正确，有可能中彩票

那到是，只要那个质数被猜到就行了，虽然希望很渺茫。。。

fan0219 说：

第八那里，我提一点异议。。苏珊收到信后，用公钥检验数字签名，用鲍勃的公钥解密摘要。不是由此可以验证这封信是由鲍勃发出的。。而是可以由此验证这封信的数字签名是有效的。。这封信，可能是由鲍勃发给A。然后A冒充鲍勃发给B。

第十一文字部分大概也有错误。。鲍勃提交自己的公钥和个人信息给CA（数字证书认证中心），并不需要提供苏珊的私钥。

然后我想问问，网页要是使用https加密浏览是不是，速度会比http慢很多。

要是文章后面能够总结一下公钥和私钥、数字签名、CA、数字证书的作用那就好了、、、、

Maple 说:

我也写了篇理解密码学的公钥和私钥的文章，以ssh免密码登陆作为载体 <http://www.lovemaple.info/blog/2011/08/ssh-remote-sever-without-password/>

[2011年8月18日 18:10](#) | [档案](#) | [引用](#)

Mr Wind 说:

如果鲍勃是亲手通过U盘把他的公钥交给苏珊的，那上面的讨论就不需要了；但如果是通过网络介质来传输，就存在着两个问 题：1该公钥是鲍勃自己发的，还是有人冒充鲍勃发的，这是身份确认的问题，2该公钥是不是被人掉包或修改，这是公钥完整性的问题。（PGP软件中公钥的生 物属性就很好地解决了上面的问题，通过电话用一些单词来与鲍勃核对公钥的完整性，其实是核对该公钥的指纹。）基于上面的原因要引入根证书。如果苏珊的电脑 是盗版（她电脑中受信任的根证书已被恶意调换），或者她的电脑中受信任的根证书的已被黑客修改，也不排除CNNIC使坏的可能，再加上网页劫持，那苏珊个 人电脑的https://……就有很大的风险。

[2011年8月20日 10:07](#) | [档案](#) | [引用](#)

finian 说:

引用Xtrats的发言:

有鲍勃的公钥就可以冒充鲍勃？

道格自己不也有鲍勃的公钥么？“每人一把”。

注意这里所说的“用自己的公钥”，不是鲍勃给道格的那支，而是道格自己伪造的另一支

[2011年8月20日 14:19](#) | [档案](#) | [引用](#)

Qujer 说:

我竟然看懂了！！！

[2011年8月21日 21:12](#) | [档案](#) | [引用](#)

fan0219 说:

引用Mr Wind的发言:

如果鲍勃是亲手通过U盘把他的公钥交给苏珊的，那上面的讨论就不需要了；……

Mr Wind 的评论好精彩啊。。我学过几个星期网络安全，对公钥密钥这块看的很有趣味。Mr Wind 盗版系统根证书被修改的例子很意思！！继续关注文章的评论。。

[2011年8月23日 00:35](#) | [档案](#) | [引用](#)

太道 说:

solidot有个类似的劫持证书的案例 <http://internet.solidot.org/article.pl?sid=11/08/31/078252>

[2011年9月 8日 04:04](#) | [档案](#) | [引用](#)

卢达 说:

引用Michael.Z的发言:

公钥和私钥的算法是一样的吗？为什么私钥加密可以用公钥解密？

这叫“非对称加密”

[2011年9月16日 10:33](#) | [档案](#) | [引用](#)

卢达 说:

只是，常见的在线邮箱缺少对 s/mime 的支持啊

[2011年9月16日 10:41](#) | [档案](#) | [引用](#)

riefuy 说:

读了这篇文章，获益匪浅。
也非常同意gool的观点，如果数据完整性得不到保证，怎么保证不可否认性？也就是数据被修改了，还能说是本人发出的吗？

[2011年10月13日 22:44](#) | [档案](#) | [引用](#)

ly 说:

bob给他们每人一把的公钥是不是是一样的？如果是，那么信件被道格截取到，是不是就同样被解密了！

[2011年10月17日 16:01](#) | [档案](#) | [引用](#)

精英一客 说：

呵呵，这个我之前也不是很懂。但是，如果你从证书的用途上去理解，比如私钥的目的是什么，公钥的目的是什么，这样的话会理解的很快

[2011年10月24日 16:06](#) | [档案](#) | [引用](#)

阿萨德 说：

银行的u盾属于私钥还是公钥，要是公钥每个u盾的内容是不是一样的呢

[2011年11月 3日 09:35](#) | [档案](#) | [引用](#)

HahA 说：

引用Xtrats的发言：

道格想欺骗苏珊，他偷偷使用了苏珊的电脑，用自己的公钥换走了鲍勃的公钥。因此，他就可以冒充鲍勃，写信给苏珊。

有鲍勃的公钥就可以冒充鲍勃？
道格自己不也有鲍勃的公钥么？“每人一把”。

其实是道格用自己的公钥（不是鲍勃给他的，是他自己的公钥）替换了苏珊的鲍勃公钥（注意是替换，原文翻译成换走可能对你理解产生了误导）。之后道格 用自己的密钥加密信件发给苏珊，苏珊用假的鲍勃公钥（实际上是道格公钥）解密信件，发现信息正确，便以为是鲍勃发来的信件，于是道格实现了伪装成鲍勃欺骗 苏珊的目的

[2011年11月 4日 20:43](#) | [档案](#) | [引用](#)

lich 说：

想这个过程的时候，如果略掉生成摘要的Hash函数貌似会更清晰些。

- 1 alice使用私钥对一份合同生成签名，同时把合同和签名发给bob。
- 2 bob收到后，使用公钥对签名解密生成一段文本。对应步骤8。
- 3 bob对比收到的合同和解码生成的文本。如果相同，则确定是来自alice。对应步骤9。

这里不能完全通过是否能解码成功来判断数据是否来自alice。
假设我使用自己的密钥对一段数据生成签名，发送给bob。bob使用alice的公钥解码，结果碰巧能解开，解开的数据有可能是乱码，无法阅读，但如果再碰巧看起来像一份合同。所以必须同时和收到的数据进行比较。

[2011年11月17日 18:16](#) | [档案](#) | [引用](#)

jieson79 说：

如果道格在网络上截取了BOB给苏珊的信，是不是就可以得到其中的内容？

[2011年12月 1日 20:49](#) | [档案](#) | [引用](#)

晴天娃娃 说：

如果第三方冒充发送方发出了一个文件，因为接收方在对数字签名进行验证时使用的事发送方的公开密钥，只要第三方不知道发送方的私有密钥，解密出来的数字签名和经过计算的数字签名必然是不相同的，这酒提供了一个确认发送方身份的方法。

[2011年12月 2日 23:13](#) | [档案](#) | [引用](#)

hui 说：

作者对公钥，私钥谁加密谁解密的解释似乎是有问题的：
“Public-key cryptography refers to a cryptographic system requiring two separate keys, one to lock or encrypt the plaintext, and one to unlock or decrypt the cyphertext. Neither key will do both functions.”

参考链接在这里：
http://en.wikipedia.org/wiki/Asymmetric_encryption

[2011年12月21日 21:28](#) | [档案](#) | [引用](#)

shanshan 说：

有个地方不明白，既然苏珊可以用鲍勃的公钥进行解密，看到鲍勃的信件内容，那道格他也有鲍勃的公钥呀，那他也可以看到鲍勃写给苏珊的信件内容罗。

[2011年12月22日 13:15](#) | [档案](#) | [引用](#)

张永 说:

图形并茂，讲的真是太好了，o(∩_∩)o 哈哈

2012年3月25日 15:26 | 档案 | 引用

辛盈 说:

看了这么多评论，我发现很多人没有搞清楚加密和认证的区别：

加密：公钥加密、私钥解密

认证：私钥加密、公钥解密

首先要搞清楚一个操作的目的是什么，目的是加密数据还是认证作者。

2012年3月26日 13:23 | 档案 | 引用

Vaporz 说:

引用Xtrats的发言：

道格想欺骗苏珊，他偷偷使用了苏珊的电脑，用自己的公钥换走了鲍勃的公钥。因此，他就可以冒充鲍勃，写信给苏珊。

有鲍勃的公钥就可以冒充鲍勃？
道格自己不也有鲍勃的公钥么？“每人一把”。

我的理解，道格给苏珊的是自己私钥生成的公钥，但因为苏珊不知道这是道格的公钥，就以为是在跟鲍勃通信。而且因为是道格私钥生成的公钥，所以道格可以正常的加密解密消息

2012年5月 8日 11:29 | 档案 | 引用

lzm 说:

文章很好，很容易理解。但我对图11有一些看法。

就你所说
“后来，苏珊感觉不对劲，发现自己无法确定公钥是否真的属于鲍勃。她想到了一个办法，要求鲍勃去找“证书中心”（certificate authority，简称CA），为公钥做认证。证书中心用自己的私钥，对鲍勃的公钥和一些相关信息一起加密，生成“数字证书”（Digital Certificate）。”

认证中心对所有的信息进行了加密，那我所理解的数字证书将是一个密文串，那当得到这个密文串证书，我根本就不能知道是哪个ca对用户信息进行的加密，认证时也不知道用谁的公钥对证书进行解密。
我理解的证书是可以公布开的，所以用户的所有信息是不必要加密的。ca所需要做的工作只是对整个用户明文信息进行认证，也就是取摘要，再对摘要就行私钥加密。
没有找到相关的资料，不知道对不对，还请作答，谢谢。

2012年5月14日 11:38 | 档案 | 引用

也都5识 说:

是篇好文章，但是，觉得文章并没有作者一开头说的那样好啊。

2012年8月19日 00:59 | 档案 | 引用

爱国者 说:

引用gooool的发言：

Alice 不能作出这样的推理：用 Bob 的公钥对签名作了一次计算，就断定签名是 Bob 发过来的。

但Alice已经知道采用哪种公钥密钥算法了，因此如果使用Bob的公钥无法解密，那么可以推定消息不是Bob所发

2012年10月 1日 23:08 | 档案 | 引用

Jeremy 说:

引用lich的发言：

假设我使用自己的密钥对一段数据生成签名，发送给bob。bob使用alice的公钥解码，结果碰巧能解开，解开的数据有可能是乱码，无法阅读，但如果再碰巧看起来像一份合同。所以必须同时和收到的数据进行比较。

当“我”给Bob发信的时候,我的信是公开的,未被加密的,因为加密也没有意思啊,原因就是很多人有我的公钥.
但关键是,怎么才能让Bob看见我的信是原版,未经删减的. 所以需要阅读原文Hash一下得到Y.
当Bob收到信后,用“我”的公钥解开签名,得到Y. 然后Bob需要对原文Hash一下,得到M,
这个时候,对比Y与M. 才能确保数据是安全的.

这里的安全包括:是由“我”发出去的,并且中途未被修改.

[2012年10月14日 01:48](#) | [档案](#) | [引用](#)

Jeremy 说:

引用shanshan的发言:

有个地方不明白,既然苏珊可以用鲍勃的公钥进行解密,看到鲍勃的信件内容,那道格他也有鲍勃的公钥呀,那他也可以看到鲍勃写给苏珊的信件内容罗。

这样的啊.本来就是这个样子的.全是道格看不见苏珊给鲍勃的信.
对应到计算机上就是:Server 发给 Client 的数据是可以被截获的.

[2012年10月14日 01:51](#) | [档案](#) | [引用](#)

Jeremy 说:

引用爱国者的发言:

但Alice已经知道采用哪种公钥密钥算法了,因此如果使用Bob的公钥无法解密,那么可以推定消息不是Bob所发

在第8步上, goool 说法是正确的.
譬如说,我们可以想像一下这个情况:Bob写了封信(包括原文和签名)给Lily,然后Lily对原文做了些修改,再把这封信转发给了Alice. Alice当然是可以对这个签名解密的. 因为他有Bob的公钥啊. 但是我们不能说,这封信是由Bob发给Alice的,因为这封信的直接受体应该是 Lily.

[2012年10月14日 01:56](#) | [档案](#) | [引用](#)

亭子 说:

非常赞,学到了好多东西,您的博客我非常喜欢看,请问是否可以做一个Android版的应用呢?这样,我们随时随地都可以看了.
如果有需要,我可以做这个android应用。

[2012年10月16日 13:13](#) | [档案](#) | [引用](#)

暗影吉他手 说:

我非常统一goool的看法.对于解密来说不存在“解密失败”这种说法.一般来讲的“解密失败”是指解密后的明文(不管是 用对称密码还是非对称密码)仍然是无意义的文章,但是对于数字签名来说,不管解密后的hash是否正确,它都是一串无意义的hash值,单就这步来说根本 看不出来是不是“解密失败”了。

再强调一遍,根本不存在“解密失败”这种说法。

[2012年11月 4日 01:05](#) | [档案](#) | [引用](#)

spraith 说:

我觉得之所以第8步有些问题,是因为Bob在制作数字签名那一步没加上一些更详细的说明,比如,第6、7步制作签名时,同 时把Bob的个人信息和信件摘要并到一起再用私钥加密的话,在第8步苏珊用公钥解密后,就确实能知道此信件是由Bob发出的了,然后第9步通过hash函 数也可以验证信件是否被修改过。而且根据第8和第9步所做的事情来看,我觉得作者的原意非常可能就是这样的,他只不过在第6 7步时漏掉了把Bob的个人信息也一起加入数字签名中。

[2012年11月10日 16:07](#) | [档案](#) | [引用](#)

小乖 说:

银行专业版 usb key里面存储的是什么信息,是客户端私钥,和客户端证书,还是服务器端证书.怎么和银行的服务端通讯的,我怎么都没想清楚?

[2012年12月21日 17:36](#) | [档案](#) | [引用](#)

小乖 说:

引用雨下路人的发言:

某些语句没有把逻辑关系表达得足够清晰(“道格想欺骗苏珊,他偷偷使用了苏珊的电脑,用自己的公钥换走了鲍勃的公钥。”起初理解为道格把鲍勃手上的公钥换走了,乱乱的,仔细琢磨语句,才明白意思是:道格用自己的公钥换走了鲍勃送给苏珊的公钥……)

道格的公钥和苏珊的公钥不都是 鲍勃送的吗,不是一样的吗

[2012年12月21日 17:40](#) | [档案](#) | [引用](#)

arlen 说:

之所以会觉得第8步有问题，是因为翻译的时候有一句重要的话没有翻译出来，原文是这样的：

Pat’s software decrypts the signature (using Bob’s public key) changing it back into a message digest. If this worked, then it proves that Bob signed the document, because only Bob has his private key.

[2013年3月 8日 10:35](#) | [档案](#) | [引用](#)

[teddywu](#) 说：

引用小乖的发言：

道格的公钥和苏珊的公钥不都是 鲍勃送的吗, 不是一样的吗

“道格用自己的公钥换走了鲍勃送给苏珊的公钥” 是指 道格用道格的公钥（不是鲍勃给道格的公钥） 替换掉了鲍勃送给苏珊的公钥。

[2013年3月16日 21:41](#) | [档案](#) | [引用](#)

刀尖红叶 说：

好文章～

[2013年3月20日 19:08](#) | [档案](#) | [引用](#)

shfqbluestone 说：

峰哥的文章写的非常好，通俗易懂！

[2013年3月27日 13:26](#) | [档案](#) | [引用](#)

[JohnK](#) 说：

第8步确实有问题，仅由签名解密得到的结果是没法判断签名者身份的。

举个例子：（假设不考虑对明文的加密）

Alice 发送一串附带签名的明文给 Bob，格式如下：

TEXT SIG1
明文 签名

现在 Eve 截获了这段信息，并伪造为：

TEXT SIG2
明文 签名

Bob 收到信息，抛弃明文不看，对 SIG2 使用 Alice 的公钥解密，得到了 DIG2。

但他并不知道正确的 HASH(TEXT) 是什么，也就不知道得到的 DIG2 到底对不对，那么怎么判断发送者的身份呢。

不可抵赖性的实现是 依赖 TEXT 和 SIG 共同实现的。

假设 Eve 想篡改 Alice 的信息为：

TEXT2 SIG2
明文 签名

SIG2 应= RSA（Alice的私钥，HASH(TEXT2)）

这样才能骗过 Bob 以为这封信息来自 Alice。

但 Alice 的私钥 Eve 并没有，因此他伪造不出签名。

因此当 Bob 验证过签名（相对明文）正确后，就认为这封信是 Alice 发送的。而不与明文相对是不成立的。

至于使用 Alice 的公钥去解密 Eve 伪造的签名会不会报错，则一定意义上依赖于加密算法。起码 RSA 这种幂乘和取模运算是不会有任何报错的。

[2013年5月 8日 21:33](#) | [档案](#) | [引用](#)

大神 说：

引用阮一峰的发言：

只有有了鲍勃的私钥，才能冒充鲍勃。

道格没有鲍勃的私钥，只好伪造鲍勃的公钥。

鲍勃给没人的公钥难道不是他们几个共有的？给的这个和鲍勃的公钥不一样？不共有？

[2013年7月 7日 10:46](#) | [档案](#) | [引用](#)

George Chen 说：

第5点，Bod决定采用 数字签名。

为什么Bod要用数字签名，应该说明下，这里Bod是为了证明自己是Bod，即这是认证过程 而非 加密过程

[2013年11月11日 22:04](#) | [档案](#) | [引用](#)

Chil 说：

引用blue gene的发言：

道格用自己的私钥加密发给苏珊的信件，苏珊收到信件后用道格的公钥自然能正常解密该信件，但是苏珊以为她收到的是鲍勃的信件，并且认为是用鲍勃的公钥来解密的，自然认为发信的就是鲍勃，所以道格就达到了伪造鲍勃与苏珊通讯的目的。

我也正好有这个疑问, 如果苏珊收到信件后能用道格的公钥解密, 这岂不是每个拥有道格公钥的人都能对信件解密?都能看到信件的内容?这样的话通信就不安全了.

[2013年11月16日 07:42](#) | [档案](#) | [引用](#)

Chil 说：

引用mazhechao的发言：

私钥和公钥在算法上是等价的，只不过一个是private，一个是public。这个应该是由密钥生成算法保证的。

如果是这样的话那通信就不安全了. 就好比Bob将信息用自己的私钥加密发给suzan，但是很多人都有Bob的公钥，那岂不是有Bob公钥的人都可以解密看到信息？

[2013年11月16日 07:55](#) | [档案](#) | [引用](#)

SelfMedicated 说：

引用gooo1的发言：

@mazhechao

我不知道你所说的“解开签名”是什么意思，它只是一步数学计算而已。

让我们更细致地看一下“验签”的过程：

- 1：输入签名和公钥，算出 hash 值 h1；
- 2：输入原文，算出 hash 值 h2；
- 3：比较 h1 和 h2，发现 h1 和 h2 相等。在这一步上，我们开始推理，得出原文是 bob 发出且没有修改过。也就是你说的抗抵赖和数据完整。

请注意，这两个有价值的结论都是第三步得出的。

原文第 8 步和你都认为：经过第 8 步的计算，即可以得出原文由 bob 发出的结论。这是错误的，实际上1 2 两步只是可以交换顺序的两个计算步骤，它们不是任何有价值结论的充分条件，连必要条件也不是。

转牛角了。同意m

[2014年1月14日 15:58](#) | [档案](#) | [引用](#)

SelfMedicated 说：

引用SelfMedicated的发言：

转牛角了。同意m

好吧，我忽然又懂gooo1的意思了，但我觉得这其实是具体实现的问题（当用不匹配的公钥去解密的时候会不会有“明显信息提示”的问题），可是这完全不是这篇文章的关键点啊，大不了是楼主翻译表述不严谨，可是不严谨又不是只有这一个地方，所以还是觉得钻牛角了...

[2014年1月14日 16:17](#) | [档案](#) | [引用](#)

[御宅暴君](#) 说：

引用gooo1的发言：

以 RSA 算法为例。

拿其它人的公钥去解 Bob 的签名会发生什么，与 RSA 的原理无关，与具体的算法实现有关。从 RSA 的原理来说，所谓“公钥”“私钥”在数学上没有区别，所谓的“加密”、“解密”、“签名”、“验签”本质上是一回事，只是一个乘方和一个取模运算。

报错，报什么错，为什么报错？因为标准实现下，Bob 要对他的信息进行编码和填充。用别的人的公钥解密，会因为填充的字节不对而无法继续计算，或无法从计算结果提取出字符，或与 hash 串规则不符，大多数实现会在此处返回错误或抛出异常。

但是，RSA 从原理上并没有保证这一点，因为它只是三个数字参与的计算而已：把一个数与另一个数作乘方运算，然后除以第三个数，得到余数。

我们完全可以采用另一套字符编码规则、字节补齐规则、以及另外的 hash 算法，让其他人的公钥参与这次计算，也能得到一个符合规则的 hash 值。

所以从逻辑上，Alice 不能作出这样的推理：用 Bob 的公钥对签名作了一次计算，就断定签名是 Bob 发过来的。

大哥，假设私钥为 (d, n)，公钥为 (e, n)，那么通过前者加密后，若要解密，所用到的 e 就必须满足 $ed \equiv 1 \pmod{\phi(n)}$ 公式了。当然就几乎只有原来那个公钥 (e, n) 的 e 满足其条件，所以不是任何公钥都可以解的。毕竟如同其它前辈指出的，你拿别的公钥去计算，就因为会满足不了该条件而出错。你可以再好好地通过阮一峰的《RSA 算法原理》补习下。

如果要找出同样满足该公式其它的 e 也不是不可以，但其算法复杂度就和用公钥加密私钥解密的情况一样了。归根结底，公私钥的确可互换，且无法通过其中一个钥匙能在可接受的成本下计算出另一个钥匙，也难怪 RSA 深得『非对称』的真谛了。

[2014年2月 5日 21:35](#) | [档案](#) | [引用](#)

[御宅暴君](#) 说：

发现评论中有不少人被 goool 误导了... 汗。

且不说这个，有人质疑既然也可以用私钥加密用公钥解密，那么因为公钥是公开的，岂不是人人都可以解开被私钥加密的数据了？

但是！谁告诉你用私钥加密是为了不让他人窥探明文数据了？这做法不是为了保密，而是在于『认证』！即验证『我所使用的公钥』与『对方加密时所使用的私钥』是否构成一对符合 RSA 算法原理的公私钥。如同我上一条评论指出的，试图使用其它公钥来解密就几乎会出错。

但是私钥就只有对方一人知道了，就拿正文中例子来说。只要对方确实实是鲍勃，从而私钥的确也是鲍勃所使用的。于是只要『苏珊所手上的公钥能对对方发过来的数据进行有效解密』，那么就证明了这公钥的确是对方，即鲍勃使用的私钥构成一对公私钥。这就所谓的『认证』了。

但是，万一对方偏偏不是鲍勃而是道格，且已经偷偷把苏珊手上的公钥换成道格的公钥了呢？这个公钥当然就与道格所有用的私钥构成有效的一对公私钥，于是道格的确就可以在苏珊完全不知情的情况下，假装成鲍勃并与她通讯。其实这问题就要通过 CA 来解决了，正文的后续当然也是围绕此而展开的。

这地方阮一峰先生的确讲的不够好，没明确好『加密』和『认证』的区别，仍未尚未彻底掌握 RSA 算法的同学的确很容易被弄糊涂甚至陷入误区。

[2014年2月 5日 21:56](#) | [档案](#) | [引用](#)

[liuruoze](#) 说：

讨论好激烈啊，其实这篇科普文非常不错，非常明晰。但是想要真正理解还需要读更专业的文章。有些同学连加密与认证这两个基本过程都不懂，看了自然云里雾里。我觉得这篇文章主要说明的是签名和数字证书的区别与关系。

另外，https传输内容是确实是对称加密算法。还有，看这篇文章时务必先了解对称加密算法与非对称加密算法的基础知识。

[2014年2月26日 10:48](#) | [档案](#) | [引用](#)

[zhanlang](#) 说：

引用御宅暴君的发言：

发现评论中有不少人被 goool 误导了... 汗。

且不说这个，有人质疑既然也可以用私钥加密用公钥解密，那么因为公钥是公开的，岂不是人人都可以解开被私钥加密的数据了？

但是！谁告诉你用私钥加密是为了不让他人窥探明文数据了？这做法不是为了保密，而是在于『认证』！即验证『我所使用的公钥』与『对方加密时所使用的私钥』是否构成一对符合 RSA 算法原理的公私钥。如同我上一条评论指出的，试图使用其它公钥来解密就几乎会出错。

但是私钥就只有对方一人知道了，就拿正文中例子来说。只要对方确实实是鲍勃，从而私钥的确也是鲍勃所使用的。于是只要『苏珊所手上的公钥能对对方发过来的数据进行有效解密』，那么就证明了这公钥的确是对方，即鲍勃使用的私钥构成一对公私钥。这就所谓的『认证』了。

但是，万一对方偏偏不是鲍勃而是道格，且已经偷偷把苏珊手上的公钥换成道格的公钥了呢？这个公钥当然就与道格所有用的私钥构成有效的一对公私钥，于是道格的确就可以在苏珊完全不知情的情况下，假装成鲍勃并与她通讯。其实这问题就要通过 CA 来解决了，正文的后续当然也是围绕此而展开的。

这地方阮一峰先生的确讲的不够好，没明确好『加密』和『认证』的区别，仍未尚未彻底掌握 RSA 算法的同学的确很容易被弄糊涂甚至陷入误区。

如果道格也去认证了自己的证书，并且用自己的公钥换了bob的公钥，每次通信的时候发自己的证书过去，结果就是一切验证都没有问题，但是苏珊却以为自己在跟bob通信，实际确实跟道格通信， 我的意思是如何辨别证书所有者跟你要通信的目标是同一个人呢？

[2014年3月12日 14:40](#) | [档案](#) | [引用](#)

Colin356 说：

英文与中文在表达的逻辑上还是不同的，喜欢谨慎的接受概念和原理的人还是改天再看看原文吧~

[2014年3月17日 02:19](#) | [档案](#) | [引用](#)

xinxinyu 说：

引用zhanlang的发言：

如果道格也去认证了自己的证书，并且用自己的公钥换了bob的公钥，每次通信的时候发自己的证书过去，结果就是一切验证都没有问题，但是苏珊却以为自己在跟bob通信，实际确实跟道格通信， 我的意思是如何辨别证书所有者跟你要通信的目标是同一个人呢？

门外汉猜测。CA认证是收费的，可能会保证不会被恶意替换。网站和证书是一一对应的。a.com - public1 b.com - public2 可能像hosts那样的列表。

[2014年3月30日 17:24](#) | [档案](#) | [引用](#)

heramerom 说：

引用zhanlang的发言：

如果道格也去认证了自己的证书，并且用自己的公钥换了bob的公钥，每次通信的时候发自己的证书过去，结果就是一切验证都没有问题，但是苏珊却以为自己在跟bob通信，实际确实跟道格通信， 我的意思是如何辨别证书所有者跟你要通信的目标是同一个人呢？

你说的 ‘并且用自己的公钥换了bob的公钥’ 是指道格偷偷用了苏的电脑，把里面的公钥换成自己么。可是文中说CA认证后，是把公钥放在传送的信息中的，而不是保存在苏的电脑上得。

[2014年5月13日 17:55](#) | [档案](#) | [引用](#)

richard 说：

引用阮一峰的发言：

只有有了鲍勃的私钥，才能冒充鲍勃。

道格没有鲍勃的私钥，只好伪造鲍勃的公钥。

怎么又变成了Bob的私钥了哈？Susan不是用Bob的公钥来解密的么？

另外还有点疑问就是为啥被替换为假的Bob公钥后，不能鉴别出来哈？不是可以将假Bob公钥解密的digest与原文hash之后比较，来判断内容是否一致么？如果是假的，digest会匹配不上哈，不是就不需要数字证书了哈？

还请您指教：)

[2014年5月14日 14:45](#) | [档案](#) | [引用](#)

richard 说：

引用mazhechao的发言：

8的说法没有问题，就这一步就可以保证信由Bob发出的。因为消息是由Bob的私钥签名的，只有Bob本人才有他的私钥，所以能用Bob的公钥解密的，一定是Bob发出的。

8实现了抗否认性，9实现的是完整性。这是两个不同的概念。

是不是说，如果解密的公钥不对了，是不能执行解密过程的，如果能执行解密就一定是bob的公钥了？

[2014年5月14日 14:53](#) | [档案](#) | [引用](#)

liuinsect 说：

那，“证书中心用自己的私钥，对鲍勃的公钥和一些相关信息一起加密，生成“数字证书”（Digital Certificate）”

证书中心怎么保证这个是 鲍勃的公钥 而不是其他人的？

[2014年5月22日 16:55](#) | [档案](#) | [引用](#)

呆瓜路 说：

谢谢院一峰先生的讲解，一目了然，通俗易懂。。

[2014年7月15日 10:03](#) | [档案](#) | [引用](#)

etnlona 说:

help a lot, thanks~!!!

[2014年7月17日 16:58](#) | [档案](#) | [引用](#)

zyj 说:

对收到的信件进行hash，得到的怎么可能跟对签名进行的hash的结果相等呢？？？？

[2014年8月25日 09:57](#) | [档案](#) | [引用](#)

Leon 说:

引用zyj的发言:

对收到的信件进行hash，得到的怎么可能跟对签名进行的hash的结果相等呢？？？？

解密得到的明文再hash得到的摘要与解密签名得到摘要进行验证，如果内容没被修改即相同。

[2014年9月23日 10:25](#) | [档案](#) | [引用](#)

刘哈哈 说:

引用CK的发言:

xiongbo027 说: 既然道格可以替换鲍勃的公钥，为什么不能故技重施，伪造CA的公钥，然后用自己的私钥伪造成CA的数字证书，从而达到欺骗苏珊的目的呢？ ===== 其实我也有同上的问题，既然CA的公钥是公开的，那么有什么办法能保证别人无法替换掉CA的公钥呢

=====

说下我个人理解：CA的公钥是放在网站上的，当需要使用的时候在网站上进行下载，这样就能保证每次使用的CA公钥是不可能被替换的正确的公钥

[2014年9月29日 15:40](#) | [档案](#) | [引用](#)

阿凡提 说:

文中第三步，苏珊给鲍勃写信，
鲍勃 怎么确定这封信就是来自苏珊的？？

[2014年10月24日 15:12](#) | [档案](#) | [引用](#)

yuntauy 说:

加密和完整性确实稍作区分更好。
Bob持有私钥，Susan有公钥。
Susan ----> Bob，数据是加密的，因而也是没被修改的。
Bob ----> Susan，数据是公开的，但是能保证是没被修改的。这时Susan实际上可以是公众中任何一个人。因为Signature是不能被修改的（修改了解不出来），进一步 保证了Digest没被修改，再进一步保证信件没被修改。实际上在原理上Bob可以直接对给Susan的回信加密，这也能保证信件没被修改。但当信件的体 量太大时，直接加密就不可取了。

[2014年11月 7日 15:42](#) | [档案](#) | [引用](#)

纳信 说:

其实wikipedia上的这幅图就能完全说明清楚了
http://en.wikipedia.org/wiki/Digital_signature#mediaviewer/File:Digital_Signature_diagram.svg

[2014年11月12日 15:32](#) | [档案](#) | [引用](#)

门外汉 说:

对这个不了解哟。我个人理解是：
首先任何人(好人和坏人)的公钥和私钥都可以用来加密解密数据，如果你访问https网站的时候用的是坏人的公钥，那么你就被坏人“中间人攻击”了，你在和坏人通信，你输入的密码等信息都被坏人用私钥解密而截获了。
那么这里就要保证、不要使用坏人的公钥加密，那么怎么知道将要访问的网站的真正公钥，而不是被坏人替换了假的公钥呢？这里用到的是事先存储方法，就是事先 将这个网站的公钥放在一个地方，并且信任这个地方的所有公钥。 浏览器程序有一个证书选项，里面有“受信任的根证书颁发机构”，还有你的系统里的证书管理工具里也有信任的机构，也就是他们颁发的证书(公钥在证书里)在 你装上系统或者下载浏览器之后是默认就信任了的！！！然

后你将要访问的网站也是从那些证书颁发机构买的，所以也就默认信任了

你现在把你的系统时间往前调整十几年，然后清除缓存，打开一个https网站，你会发现提示证书未生效，因为浏览器根据已经信任的证书来判断的，在已经信任的证书(公钥在证书里)里，有个有效期，不在这个区间内浏览器就认为证书有问题。

至此，已经知道浏览器或者操作系统是靠“受信任的根证书颁发机构”这个玩意来确定是好人还是坏人的，所以如果你的系统或浏览器能够被黑客控制，那么黑客就有可能把你默认信任的证书替换成黑客自己的“坏证书”，来进行中间人攻击。所以保证自己系统安全很重要。

那么有没有不需要入侵系统就能对https进行中间人攻击呢？答案是有！那就是CNNIC，https追溯到上级，就是信任由颁发证书的机构颁发的证书，然而，CNNIC从2009年已经被火狐微软等公司默认信任了，也就是说CNNIC制作的ca证书你都默认的信任了，哈哈！

CNNIC当初还被CCAV报道过，搞的cn域名被晃色网站随便换域名。GFW和CNNIC是孪生兄弟，GFW负责DNS域名污染，CNNIC复杂伪造“合法”的ca证书，GFW就可以轻松的搞的任何网站的https加密传输！

最好将CNNIC从浏览器和操作系统的信任列表里删除(IE和谷歌用的系统的信任列表，火狐用的自己的信任列表)

大家可以搜索下。。。
不对请指教

[2014年12月 6日 21:55](#) | [档案](#) | [引用](#)

我要发表看法

您的留言（HTML标签部分可用）

您的大名：

 «-必填

电子邮件：

 «-必填，不公开

个人网址：

 «-我信任你，不会填写广告链接

记住个人信息？ ☐

«- 点击按钮

[联系方式](#) | ruanyifeng.com 2003 - 2014 

来源： <http://www.ruanyifeng.com/blog/2011/08/what_is_a_digital_signature.html>