使用KEA和SKIPJACK加密

本备忘录的状态

本备忘录详细描述了 Internet community 的实验性协议，但不说明任何一种类型的 Internet 标准。它需要进一步进行讨论和建议以得到改进。发布本备忘录不受限制。

版权声明

概要

本文档详细描述了一个传输文件在使用FTP规范标准9、RFC959、“文件传输协议（FTP）”(1985年10月)[3]、和RFC 2228“FTP 安全扩展”（1997年10月）[1] 时加密的方法。该方法将使用密钥交换算法（KEA）提供相互鉴定并建立数据密钥。SKIPJACK用来加密文件数据和FTP通道指令。

1.0介绍

文件传输协议（FTP），除了传输鉴定用户身份的口令之外没有提供安全协议。另外，该协议也不能保护远程文件传输的鉴定状态。

FTP的安全扩展问题已经提交到互联网工程任务组(IETF)和通用鉴别技术工作组（CAT）。这些扩展允许该协议使用更多的柔性安全方案，并特别允许为FTP指令和数据连接提供多种级别的保护。该文档描述了FTP安全扩展一个的轮廓，这些规则可能提供使用密钥交换算法（KEA）和SKIPJACK协同的均衡加密算法。

FTP安全扩展[1]规则：

- ✶ 用户鉴定——加强一般密码机制；
- ✶ 服务器鉴定——正常地连接用户鉴定；
- ✶ 交换参数流通——特别的加密关键字和特征码；
- ✶ 指令连接保护——完整地，机密地，或包含以上两点；
- ✶ 数据传输保护——和指令连接保护类似

为了支持上述安全服务，两个FTP实体需用一种机制沟通。这个过程是开放的和完整的，当两个实体都接受同一机制或初始的一方（通常为客户端）不能确定一个合适的机制时。一旦同意使用一个机制，它们会开始鉴定并进行参数传递。

交换和参数的传递发生在一个极大的交换组中。当交换完成，实体的任何一方（单方或双方）会进行鉴定，然后，又准备去保护FTP指令和数据。

在交换完成之后，实体会紧接着对传递的缓冲区大小进行保护。这个过程用两步完成：客户端提出一个缓冲区尺寸，服务器端会从拒绝、修改、同意三者中选一。

至此，实体会在参数传递的绑定包里发出保护指令，并始终贯穿安全交换的全过程。保护指令被申请保护服务所需的一般指令和把结果译成编码的Base64指令发出。译码的结果被一个ENC（完整的和机密的）指令以数据流形式发送出去。Base64是一个把文本字符映射成二进制数据的译码指令，它能通过大多数的7位（bit）系统而不丢失。服务器的响应在新的结果码中传回，并且同样也允许使用保护和Base64译码指令以应用于回应的结果。数据传输的保护被规定使用PROT指令，该指令支持同样提供给其他FTP指令的保护。PROT指令能在有传输功能的系统中送出，不过，交换参数在交换过程中不能改变。

2.0 密钥交换算法（KEA）轮廓

本节描述KEA和SKIPJACKY 在和FTP安全扩展框架联合使用时如何完成可靠的安全服务。 FTP实体将使用KEA来相互鉴定，并建立数据密钥。我们详细地说明一个简单的标记格式和一组交换以陈述这些服务。相关功能会通过Fortezza Crypto Card来演示。

为了理解下面的协议，读者要熟悉这些扩展。在FTP安全扩展的文章中，我们建议使用KEA和SKIPJACK来进行鉴定、保证完整性和机密性。

客户端会与服务器相互鉴定。以下是在FTP安全扩展框架下实现KEA鉴定的协议必需的步骤。在遭遇失效状态的地方，返回码跟随在扩展功能指定的列表中。但在本文档中没有列举，因为它们在机制使用中是固定不变的。证书为ASN.1编码。

以下假定一个FTP安全扩展的应用实例进行交换的详细描述。联接符号用“||”表示。加密的译码数据和鉴定路径的确认是默认假定的，但没有明示。

```
------------------------------------------------------------------
 客户端                          服务器端

 AUTH KEA-SKIPJACK               -->
                            <-- 334 ADAT=Base64( Certb || Rb )
 ADAT Base64( Certa || Ra ||
   WMEK || IV || Encrypt(
   Label-Type || Label-Length ||
   Label-List || pad || ICV ) ) -->
                            <-- 235 ADAT=Base64( IV )
------------------------------------------------------------------
```

图1

服务器端和客户端的证明书包含KEA公共密钥。客户端和服务器端使用KEA产生一个共享的SKIPJACK均衡密钥，称为TEK。客户端使用随机数创建一个二次SKIPJACK密钥，称为MEK。为了传输到服务器，MEK被封装在TEK中。当客户端向服务器端传输时，一个初始化向量在MEK被创建时一同生成，一个客户端要使用FTP任务的安全列表将被使用MEK加

密后传输到服务器。如图2所示，安全列表数据的格式是一个8位（1字节）的数值，一个4字节的列表长度，安全列表清单，补充，跟在一个8字节的完整校验值（ICV）之后。图3列出了列表的类型。如果列表的类型是缺省的（长度为0），则列表长度必须为0。

为了保证纯文本的长度是密文块大小的数倍，在完成后需要做如下补充：对SKIPJACK CBC密码化处理的输入将作8字节倍数的补充。设n是输入的字节长度。补充的输入附加8-(n mod 8)字节到信息的末尾，每个信息均需附加8-(n mod 8)个字节。在十六进制中，可能补充的字串是：01, 0202, 030303, 04040404, 0505050505, 060606060606, 07070707070707, and 0808080808080808.所有的1至8字节补充输入都是为了让长度是8的倍数。只要使用SKIPJACK CBC加密技术，都要进行这样的补充。

ICV是在对无格式文本安全列表和补充的计算得到的。ICV的算法是32位的自身反码加法，每个32位块跟随32个0位。ICV算法在使用SKIPJACK CBC加密法时使用，以提供数据的完整性。

列表类型	1 字节
列表长度	4 字节
列表清单	可变长度
补充	1 到 8 字节
ICV	8 字节

图2

列表类型	列表语法	参考
0	缺省	不适用
1	MSP	SDN.701[2]
2-255	保留	保留

图3

FTP指令通道操作现在是做为机密来保护的。为了提供完整性，指令序号、补充信息、和ICV都添加在每一个指令的前面以实现加密。

序列的完整性是通过为每个指令增加16位的序列数字来实现的。序列号初始化时是最低有效16位Ra。服务器的响应也会象客户端那样含有同样的序列号。

IVC是通过单独的指令（包括必要的回车和换行符以结束指令）、序列号、和补充信息计算出来的。

客户端	服务器端

```
   ENC Base64(Encrypt("PBSZ 65535"
     || SEQ || pad || ICV ))     -->
                              <-- 632  Base64(Encrypt("200" ||
                                        SEQ || pad || ICV))
   ENC Base64(Encrypt("USER yee"
      || SEQ || pad || ICV))    -->
                              <-- 632  Base64(Encrypt("331" ||
                                    SEQ || pad || ICV))
   ENC Base64(Encrypt("PASS
       fortezza" || SEQ ||
       pad || ICV))             -->
                              <-- 631  Base64(Sign("230"))
-------------------------------------------------------------------
```

图4

译码之后，两个实体用PBSZ指令检查预知的序列号和正确的ICV来校验完整性。正确的SKIPJACK计算，ICV校验，和包含KEA公用密钥的证书，一起提供相互的辨认和鉴定。

```
-------------------------------------------------------------------
   客户端                       服务器端

   ENC Base64(Encrypt("PROT P" ||
        SEQ || pad || ICV))  -->
                            <-- 632 Base64(Encrypt("200" || SEQ
                                  ||  pad || ICV))
-------------------------------------------------------------------
```

图5

至此，在使用中，文件将会以加密和完整性服务来发送和接收。如果应用加密技术，第一个缓冲区将包含标记，跟随足够多的译文字节以完全充满缓冲区（除非文件太短以至无法充满缓冲区）。后面的缓冲区将仅包含译文字节。除最后一个缓冲区外所有的缓冲区都会被完全充满。

```
-------------------------------------------------------------------
   客户端                       服务器端

   ENC Base64(Encrypt(
     ("RETR foo.bar") ||
     SEQ || pad || ICV))    -->
                            <-- 632 Base64(Encrypt("150" ||
                                  SEQ || pad || ICV))
-------------------------------------------------------------------
```

图6

下图显示头信息和文件数据：

Plaintext Token IV	24 字节
WMEK	12 字节
Hashvalue	20 字节
IV	24 字节
Label Type	1 字节
Label Length	4 字节
Label	Label Length 字节
Pad	1 to 8 字节
ICV	8 字节

图7

2.1加密前的文件支持

为了支持加密和加密前的文件，为传输一个文件的密钥（FEK）定义一个标记。为防止残损和保证文件完整，标记也包含了一个在完成的文件里计算出的无用信号。标记也和文件联合在一起包含了安全列表。FEK封装在TEK中。标记使用SKIPJACK CBC模式加密在TEK中。标记包含一个12字节的已封装FEK，一个20字节的文件无用信息，一个24字节的文件IV，一个1字节的列表类型，一个4字节的列表长度，一个可变长度的列表数据，一个1至8字节的补充，和一个8字节的ICV。标记开始的24个字节是无格式的IV，一般用于标记的剩余内容的加密。标记需要它自己的加密IV，因为它通过数据通道传输，而不是指令通道，并且次序介于二个通道之间，但不能有保证。文件系统的文件计算前的密钥和无用信息的存储是一个本地执行问题。然而，如果一个文件在加密前被暗示，那么FEK会被在本地存储的密钥里封装起来。当文件需要，FEK被用本地存储密钥解封，然后再在TEK中封装。图8显示了装配标记。

Plaintext Token IV	24 字节
Wrapped FEK	12 字节
Hashvalue	20 字节
IV	24 字节
Label Type	1 字节

Label Length	4 字节
Label	Label Length 字节
Pad	1 to 8 字节
ICV	8 字节

图8

3.0 关键字术语表

为了清晰地阐明在协议中使用各种不同关键字，图9概括了关键字的类型和它们的用途:

关键字	使用
TEK	每个文件开始部分的加密标记，也封装在MEK和FEK中
MEK	指令加密通道
FEK	文件自我加密(可能超出FTP范围)

图9

4.0 安全考虑

整个备忘录是关于安全机制的。KEA提供了鉴定和可供讨论的密钥管理，当解密私人密钥时执行必须得到保护。SKIPJACK提供了可供讨论的机密性，当解密公共均衡密钥时执行必须得到保护。

5.0 感谢

（略——译者注）

6.0 参考书目

1. Horowitz, M. and S. Lunt,“FTP安全扩展”，RFC 2228，1997年10月。
2. 消息传输安全协议4.0（MSP），修订版　　A。安全数据网络系统（SDNS）SDN.701,1997年2月6日。
3. Postel, J. and J. Reynolds,“FTP传输协议”，STD 9,RFC 959,1985年10月。

7.0 作者地址

（略——译者注）

8.　完整的版权声明

（略——译者注）

感谢

（略——译者注）

原文：RFC 2773《Encryption using KEA and SKIPJACK》
译者：张偶 2002年4月