From:h draft-ietf-iesg-tftp-00
Network Working Group
Request for Comments: 9138
Updates: 1350
Category: Standards Track

Proposed Standard
F. Bocca
UACH
October 2021

TFTP+ AES Encryption Extension

Status of this Memo

Abstract

The Trivial File Transfer Protocol [1] is a simple, lock-step, file transfer protocol which allows a client to get or put a file onto a remote host. This document describes a simple extension to TFTP to allow AES encryption to the file transfer increasing security.

Introduction

The AES encryption mechanism proposed in this document is a light modification to the TFTP protocol, It allows file transfer but with encrypted data packages using symmetric encryption, generating an AES key in the client and host side using the same keyword and initialization vector to obtain an identical key to encrypt and decrypt the file transferred.

Acknowledgements

The protocol was originally designed by Noel Chiappa, and was redesigned by him, Bob Baldwin and Dave Clark, with comments from Steve Szymanski and then improved by the MIT in 1992 . This actual modification was requested by Christian Lazo, professor of the UACH to his students of the Networks Course, who requested an improvement in the security of the protocol, being the author of this document the person who managed to increase the security in the protocol inspired by the popular instant messaging providers.

Packet Formats

TFTP uses Read Request and Write Request packets of the form:

```
 2 bytes     string      1 byte   string     1 byte
 ---------------------------------------------------
| Opcode |  Filename  |   0  |    Mode    |   0  |
 ---------------------------------------------------
```

RRQ/WRQ packet

In order to implement the encryption improvement, the packages of RRQ and WRQ should look like this:

```
  2 bytes    string    1 byte   string   1 byte  3 bytes  1 byte  5 bytes
 ---------------------------------------------------------------------------
 | Opcode |  Filename  |  0 |    Mode    |  0 |    ID    |  0 |    Port  |
 ---------------------------------------------------------------------------
```

RRQ packet

```
  2 bytes    string    1 byte   string   1 byte  3 bytes  1 byte
 ------------------------------------------------------------------
 | Opcode |  Filename  |  0 |    Mode    |  0 |    ID    |  0 |
 ------------------------------------------------------------------
```

WRQ packet

The change in the RRQ packet is because when the communication between client and host through port 69 is accepted, the client must indicate a different port to receive the Data Packets to avoid congesting port 69, also the port is a key component in the generation of the AES key. Meanwhile both RRQ and WRQ now include an ID in the packet for the same reason, the ID in the communication is obtained by the Client and informed through the RRQ/WRQ to the host to generate the AES key. The Opcodes are still 1 and 2 respectively.

Another big change is in the ack packages, previously the protocol used the same ACK for the Data and WRQ packets, now they are different, we have an ACK packet for the DATA packets and a specific packet for WRQ packet acknowledgement, for the same reason the RRQ packet was modified, to communicate to the client which port is gonna be used for Data packets transfer.

```
  2 bytes     2 bytes
 -------------------------
 | Opcode |   Block #  |
 -------------------------
```

ACK packet for DATA packets

```
  2 bytes    2 bytes   5 bytes
 ---------------------------------
 | Opcode |   Block #  |  Port  |
 ---------------------------------
```

ACK packet for WRQ packet

This new ACK packet will have the opcode: 6, meanwhile the other ACK packet will have the same old opcode: 4.

Encryption Protocol Specification

Once the WRQ has been acknowledged, the file to be transfer follows its normal course, being divided into 512-byte chunks, and added to a data packet, then the data packet should be encrypted, but for that, his length must be a 16 byte multiple (because of how AES algorithm works), so the length of the packet is checked first, and if the packet length is not multiple of 16 bytes, it has to be padded, and only then the data packet can be encrypted. The encryption uses the port chosen for the transfer as an integer variable with the ID provided by the client as well for the initialization vector and key respectively, once the data packet is encrypted is sended to the client by the host.

On the client side, if the data packet arrives without any issues, the decryption process begins, being followed by a de-padding process as well. Then the ack packet is sended to the host as usual.

For the RRQ operation the process is the same, only this time, the encryption process starts on the client side and the decryption process happens on the host side.

The key generated is created per file only.

References

[1] Sollins, K., "The TFTP Protocol (Revision 2)", STD 33, RFC 1350, October 1992.

Security Considerations

In this document a security layer has been added to the TFTP protocol, however login or access control mechanisms are still missing in this protocol, care must be taken in the rights granted to a TFTP server process so as not to violate the security of the server hosts file system. With that in mind TFTP should be installed with controls such that only files that have public read access are available via TFTP and writing files via TFTP is disallowed.

Author's Address

Franco Bocca Epple
Universidad Austral de Chile
Instituto de Ingeniería Civil en Informática
General Lagos 2086, Edificio 10000
Valdivia, XIV Región, Chile

Phone: (+56) 632-221439

EMail: FRANCO.BOCCA@ALUMNOS.UACH.CL