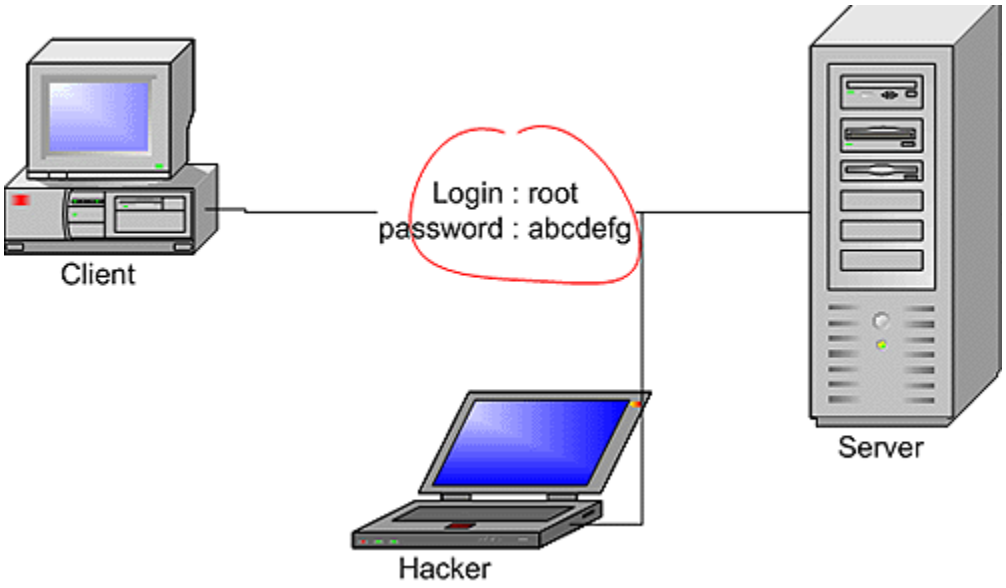


Encrypted Telnet: Final Project in C

PRASHANT TRIPATHI
UIN: 665713172
Email: ptripa4@uic.edu

[a] TELNET

Telnet is a network protocol used on the Internet or local area networks to provide a bidirectional interactive text-oriented communication facility using a virtual terminal connection. User data is interspersed in-band with Telnet control information in a byte oriented data connection over the Transmission Control Protocol (TCP).

It is a client-server protocol, based on a reliable connection-oriented transport. Typically, this protocol is used to establish a connection toTransmission Control Protocol (TCP) port number 23, where a Telnet server application (telnetd) is listening.

Telnet, by default, does not encrypt any data sent over the connection (including passwords), and so it is often practical to eavesdrop on the communications and use the password later for malicious purposes; anybody who has access to a router, switch, hub or gateway located on the network between the two hosts where Telnet is being used can intercept the packets passing by and obtain login, password and whatever else is typed with a packet analyzer.

Most implementations of Telnet have no authentication that would ensure communication is carried out between the two desired hosts and not intercepted in the middle.

Several vulnerabilities have been discovered over the years in commonly used Telnet daemons.

[b] ENCRYPTION

In cryptography, encryption is the process of encoding messages (or information) in such a way that third parties cannot read it, but only authorized parties can. Encryption doesn't prevent hacking but it prevents the hacker from reading the data that is encrypted. In an encryption scheme, the message or information (referred to as plaintext) is encrypted using an encryption algorithm, turning it into an unreadable ciphertext. This is usually done with the use of an encryption key, which specifies how the message is to be encoded. Any adversary that can see the ciphertext should not be able to determine anything about the original message. An authorized party, however, is able to decode the ciphertext using a decryption algorithm, that usually requires a secret decryption key, that adversaries do not have access to. For technical reasons, an encryption scheme usually needs a key-generation algorithm to randomly produce keys. In this case a fixed key which has been hard-coded is being used.

The existing encryption algorithms can be classified into two main categories:

1) Symmetric key algorithms

2) Public key encryption algorithms

The current project implements only the symmetric key algorithms. The following algorithms are supported by the code:

- ✔ blowfish
- ✔ des
- ✔ tripledes
- ✔ threeway
- ✔ gost
- ✔ safer-sk64
- ✔ safer-sk128

- ✔ cast-128
- ✔ xtea
- ✔ rc2
- ✔ twofish
- ✔ cast-256
- ✔ saferplus
- ✔ loki97
- ✔ Serpent
- ✔ aes
- ✔ rijndael-128
- ✔ rijndael-192
- ✔ rijndael-256
- ✔ enigma
- ✔ arcfour
- ✔ Wake

[b] CODING AND TESTING PLATFORM

The coding has been done in GNU C and compiled through GCC 4.7 compiler for x86_64 core. The platform has been Fedora 18/ Linux core. The code is essentialy small distributed across two C files:

1) server.c : This contains the telnet server implementation.

2) client.c: This contains the telnet client implementation.

3) aes.c: This has the encryption and decrytption related functions.

There is a basic GNU Makefile which scripts the compilation and cleaning process. The final executable binaries have been given the name “server” and “client”.

Apart from this the common definitions and file includes are in the header files network.h and aes.h

The library MCRYPT for linux has been used for encryption APIs and related data structures.

[c] DESIGN DESCRIPTION

i) The server creates a socket and waits for client to connect. The server in itself is a process running continuously.

ii) Client connects to the server at the specified socket.

iii) On successful connection, a child process is created by the server which provides the shell utility to the client.

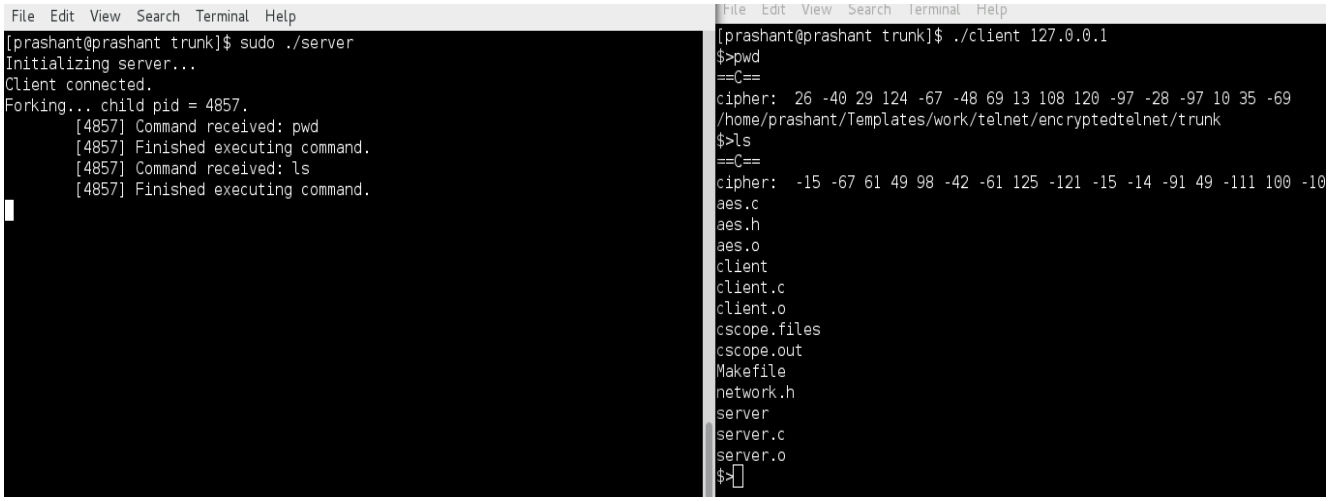
iv) The client has a layer of encryption, which encrypts everything that is send to the server.

v) The server end has a decrypter which decodes the command passed by the client and performs the necessary action.

[c] USAGE AND SAMPLE OUTPUTS

The command format is as follows:
To run the server: sudo ./server
To run the client: ./client 127.0.0.1

Some sample runs have been embedded below.


File Edit View Search Terminal Help
[prashant@prashant trunk]$ sudo ./server
Initializing server...
Client connected.
Forking... child pid = 4857.
[4857] Command received: pwd
[4857] Finished executing command.
[4857] Command received: ls
[4857] Finished executing command.
File Edit View Search Terminal Help
[prashant@prashant trunk]$ ./client 127.0.0.1
$>pwd
==C==
cipher: 26 -40 29 124 -67 -48 69 13 108 120 -97 -28 -97 10 35 -69
/home/prashant/Templates/work/telnet/encryptedtelnet/trunk
$>ls
==C==
cipher: -15 -67 61 49 98 -42 -61 125 -121 -15 -14 -91 49 -111 100 -10
aes.c
aes.h
aes.o
client
client.c
client.o
cscope.files
cscope.out
Makefile
network.h
server
server.c
server.o
$>