Packet Sniffing Backdoor Design

William Murphy, Benny Wang

Table of Contents

Implementation Details

Encryption

Our program uses basic XOR encryption. Each byte of the plaintext is XOR'd with the byte of the key in the corresponding location. If the key is shorter than the plaintext, the key is concatenated to the end of itself over and over again until the key length is greater than or equal to the length of plaintext. Decryption is the exact same process but instead of using the plaintext we use the ciphertext.

Authentication

Our authentication scheme uses the TCP source port and the TCP sequence number. When creating the TCP header, the TCP source port must be the first 4 bytes of the SHA256 hash of the TCP source port in network byte order.

E.g.

If the source port is **7575**, the SHA256 hash of **7575** is:
c91a1ad0b6bf41aba97606740e92c02d87155d8a3626787464417dbda5eae57f.

We then take 0xc91a1ad0 and place it in the TCP header in network byte order.

Pseudocode

```
Main Function
{
  Initialize variables
  Build packet filter

  Set mode based on command line arguments

  If backdoor in client mode
    Send command to server (supplied by command line arguments)

  If backdoor server mode
    Mask the process name
    Raise its privileges

  Loop forever
  {
    Call packet capturing function
    Pass captured packet to packet handler
  }
}

Packet Handler Function
{
  Verify packet is intended for the backdoor by checking
authentication
  If not authenticated
    Return

  If packet is from the same IP as the sniffing interface
    Return

  Otherwise, decrypt packet data using a predefined password
  If backdoor is in server mode
    Check for command in decrypted packet data
    If command or command flags not found
      Return
    Execute command
    Encrypt command output
    Send the result to the client
  Else if client mode
    Print the decrypted output to the screen
}

Send Command Function
{
  Generate random source port
  Hash source port and use first 4B as sequence number
  Xor encrypt command
  Create raw socket
  Construct IP/TCP header
  Write packet to raw socket
  Close socket
}
```