Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 8 August 2022

K. Unice
Anonymous

4 February 2022

Privatization of service type in IPv4/IPv6 data flows
draft-unice-dispatch-enc-ports-00

Abstract

The internet and free flow of information has enabled the individual in society several advantages that have not been available in the past. The free flow of information has enabled social and political changes, open education for students, and now the free-flow of money without the need for intermediate institutions. This free flow of information etc. is continually under attack from institutions that are compromised by the Internet. So these institutions have and continue to implement controls in an attempt to stop the empowerment of the individual. These controls on information is what this RFC attempts to weaken.

Status of This Memo

Copyright Notice

Table of Contents

1. Rationale

The socket number in data flows across the internet is usually tied to a specific service type. This is a security flaw in that flows can be intercepted, denied, or compromised by using the socket number in the IP packet. This RFC is to enable IP data flows across the internet to hide the type of service being provided from intermediate routers etc. This will enable internet software developers to build applications that cannot be subjected to censorship. This also prevents intermediate points from profiling certain applications.

There are well known sockets (ports 0-1023) that are assigned to specific services (i.e. 80 is HTTP, 443 is HTTPS, 25 is SMTP, etc.). Other ports (1024-49151) are registered ports that can be reserved with IANA. Having well known ports for specific services enables intermediate routers etc. to snoop, block, or spoof these services. This RFC is meant to eliminate the ability of intermediate entities to attach a specific service by controlling the port used by a service in their routers.

2. Approach

This specification can accept and adopt recommendations for technical approaches as long as the recommendations maintain the final goal of this RFC. The final goal is to prevent intermediate internet providers from filtering and stopping a specific service.

3. Design

There are options as to how the receiving IP stack can determine if the IPv4/6 header has the ports portion encrypted.

A. This Secured Service Type (SST) feature will start with a new (0th bit in the Flags header) in the IPv4 header that will indicate that the packet is encrypted at the point that the IPv4 header ends and the TCP/UDP header starts.

B. The Traffic class field of IPv6 could have a number or bit reserved for SST.

4. Implementation

The IPv4/v6 stack will implement a "side stack" that will implement the decryption when the received flow has the bit set indicating a SST type packet. The "side stack" module will be passed inbound packets and will determine if the header has its ports encrypted. If so, then it will decrypt the rest of the portion of the IPv4/v6 packet header just after reception from the network controller and send the decrypted packet up the stack as a normal frame.

As part of the usual "sockets" programming API (socket, bind, connect, send, receive, etc.) there will be an additional enumeration for the setsockopt API that will indicate the header will be partially encrypted and have the destination and source ports unreadable. Applications that want to obscure the service type will indicate in the APIs that the UDP packet be encrypted using SST, and to establish a TCP connection using SST.

```
   +--------------+
   |Standard V4/V6|
   |Software Stack+-----+
   +-+------------+     |C
     |      ^           |
     |      |           v
     |      +----+----------+
    D|      B    |  SST     |
     |           |Encryption|
     |           +-+--------+
     |             |      ^
     v             v      |
   +-----------------+    |A
   |Network Interface+----+
   |Software Driver  |
   +--+-------^------+
      |       |
   +--v-------+------+
   |Network Interface|
   |Hardware         |
   +-----------------+
```

Figure describing the SST software function

(A) All ingressing packets are sent to the SST software module. Both packets using SST and packets that are not using SST are passed to the SST module. Traffic that is using SST is decrypted and sent up the stack (B). Traffic that is not using SST is passed up the stack, without any change (B).

(C) Egressing traffic checks the socket control structure for this socket. The SST module encrypts the ports section of the IPv4/v6 socket using its private key. (D) is for traffic that is not using SST and is passing the destination socket in clear text.

5. Security Considerations

There has to be an initial key acquisition in order for the initial traffic between the user and the domain server to be able to encrypt the ports section of the IPv4/v6 header.

The initial idea would be that the DNS resolution of the domain name would include a public key as part of the response mapping the DNS ascii name to an IPv4/v6 address. The user would then use the domain server's public key to encrypt the portion of the IPv4/v6 header that contains the destination port. This first packet from the user sent to the domain server will have the destination port encrypted.

6. IANA Consideration

None

Appendix A. Acknowledgments

TODO

Author's Address

Kyle Unice
Anonymous

Email: kspambot@gmail.com