Assumptions in Intrusion Analysis

Gap Analysis

by Rodney Caudle

Objectives

- What are stackable standards?
- Common Assumption in Analysis
- Discovering a Gap
- Exploiting the Gap

Stackable Standards

- What are “stackable standards”?
- Flexibility vs. Security
- Relationship between layers
  - No-Connection
  - Unidirectional
  - Bidirectional

TCP/IP Standards Stack

- Layer 0 – IEEE 802.3 (Ethernet)
- Layer 1 – RFC 894 (IP)
- Layer 2 – RFC 793 (TCP)
- Layer 3 – RFC 2616 (HTTP)

Sockets

- Three Parameters Needed
  - Communication Domain for socket
    - PF_INET
  - Type of connection to build
    - SOCK_STREAM – TCP
    - SOCK_DGRAM – UDP
    - SOCK_RAW – IP
  - Protocol to use
    - Set to “0” (zero) unless SOCK_RAW is specified

Assumptions

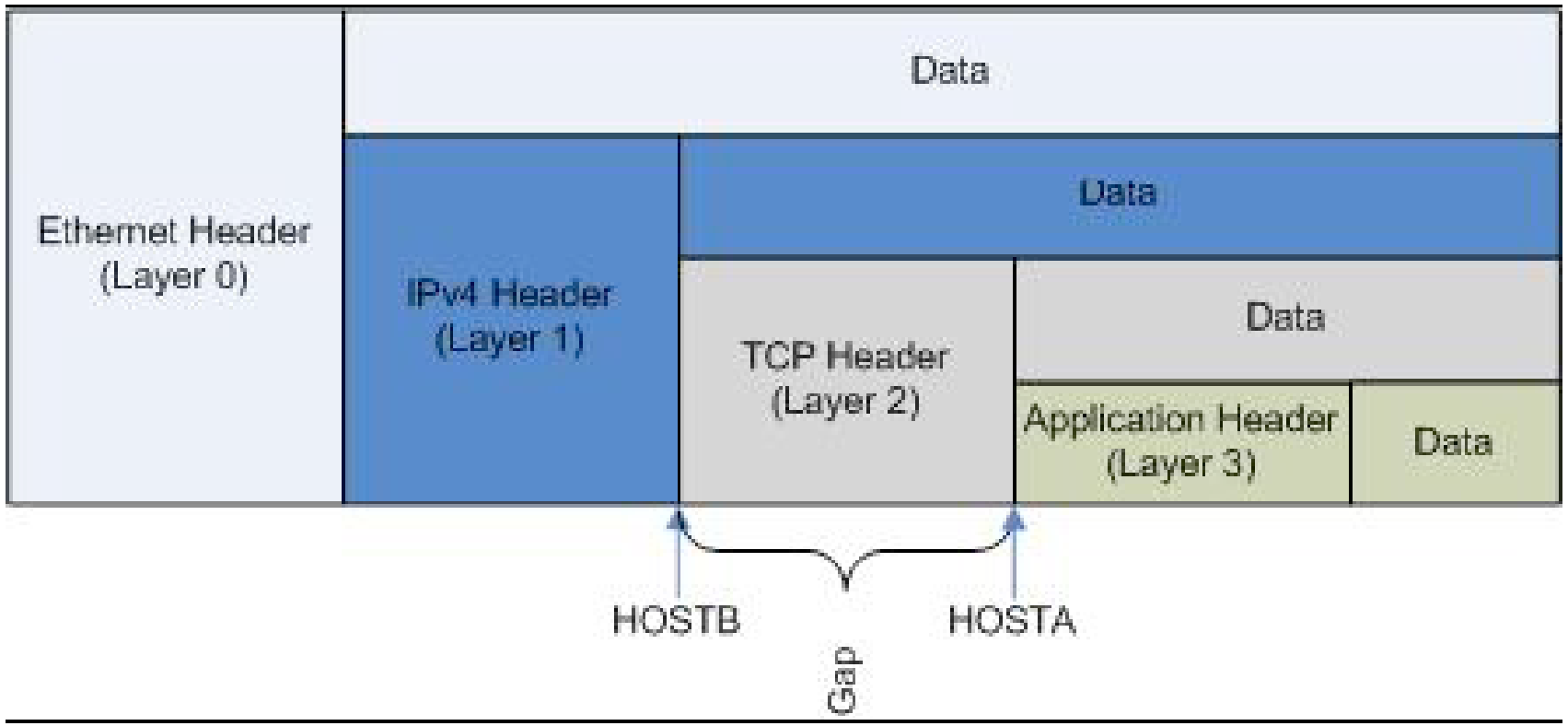
IP Protocol Field is trustworthy FALSE

- There is no requirement for trusting this field beyond instructing the receiving stack how to interpret the Layer 2 protocol

The Setup

1. Convince the analyst (or countermeasure) to look elsewhere
2. Use enough of the higher protocol to appear like a normal connection
3. Adhere to Normalcy Tests

Visualizing the Gap



The Gap

- Host A = Countermeasure, Originating Host
- Host B = Covert Receiving Program

- Since Host A “trusts” byte 9 offset from zero of the IP header, a gap is available

Normalcy Tests

- What makes a packet appear normal?
  - Successful Delivery (functional)
  - No “red flags” which will direct analysts (or NIDS)

IP Header Normalcy

- Lots of checks for normalcy needed in IP header (check notes)
- Required for successful delivery of packet
- Very little “space” to hide communications

TCP Header Normalcy

- Fewer checks than IP header
- 7 out of 20 bytes of standard TCP header are “required” for normalcy
- 13 bytes per header for covert channel

Hiding the Channel

- Use the sequence number and acknowledgement number to embed data in the TCP header
- 12 bytes per packet to leak data
- Very hard to detect

Getting Tricky

- What if only SYN packets are used?
  - Sequence and Ack will increment with each packet effectively making the channel disappear after the first packet

Normal SYN Packet

```
0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00
0010   00 3c 42 5e 40 00 40 06 fa 5b 7f 00 00 01 7f 00
0020   00 01 83 0d 06 26 ad c2 f8 d6 00 00 00 00 a0 02
0030   80 18 a1 ea 00 00 02 04 40 0c 04 02 08 0a 03 4f
0040   ba 88 00 00 00 00 01 03 03 05
```

Crafted SYN Packet

```
0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00
0010   00 28 73 90 00 00 ff 06 4a 3d 7f 00 00 01 7f 00
0020   00 01 30 39 39 30 45 4e 4f 44 20 4c 4c 41 50 02
0030   08 00 3f 57 00 00
```

- Does anything about this packet stand out?
- Passes all of the normalcy tests

Wrapping Up

- In 2002, attacks created a gap by encapsulating IPv6 inside IPv4 packets
- Countermeasures were not prepared and made assumptions that allowed any attack to pass undetected

Protecting Your Network

- Requires the preservation of Layer2
- Any perimeter that does not rebuild up to Layer 2 is susceptible to this approach
- Non-proxy firewalls can leak data