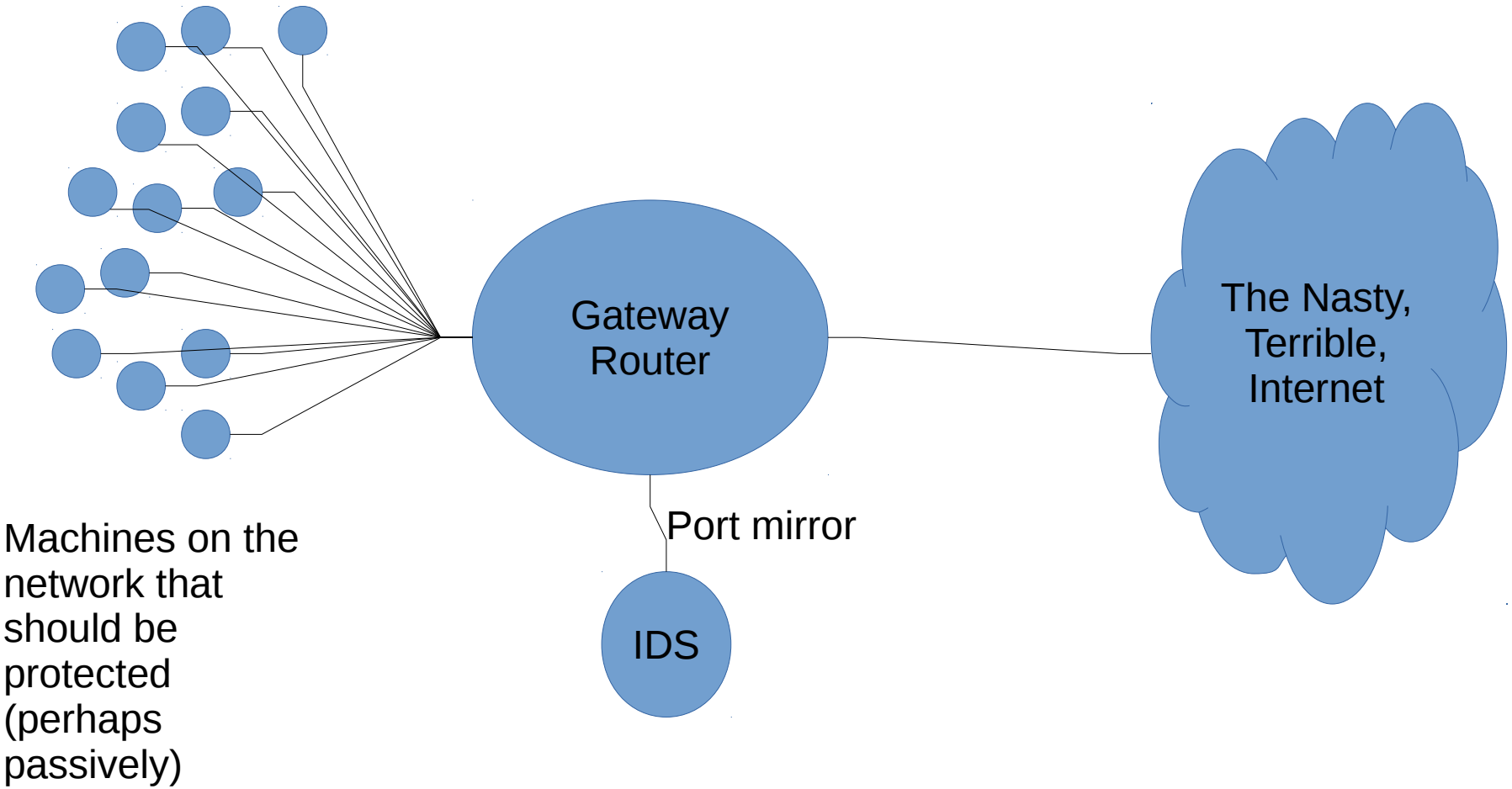
Insertion, evasion, denial-of-service, and other network tomfoolery

Intrusion Detection System (examples are Bro or Snort)



IDS is looking for signatures

- Typically regular expressions, like `".*<script>.*</script>.*"` appearing in an input to a web form, indicating a Javascript XSS attack.
- How can we (the attacker) get the IDS to see one thing and the victim to see another?
- A stupid example: Great Firewall of China censors `"GET falungong.html"`, but if you send two packets: `"GET fa"` and `"lungong.html"` the endhost reassembles them fine but the GFW is fooled.
- Or, "GET fa%61lungong.html"

A not so useful distinction

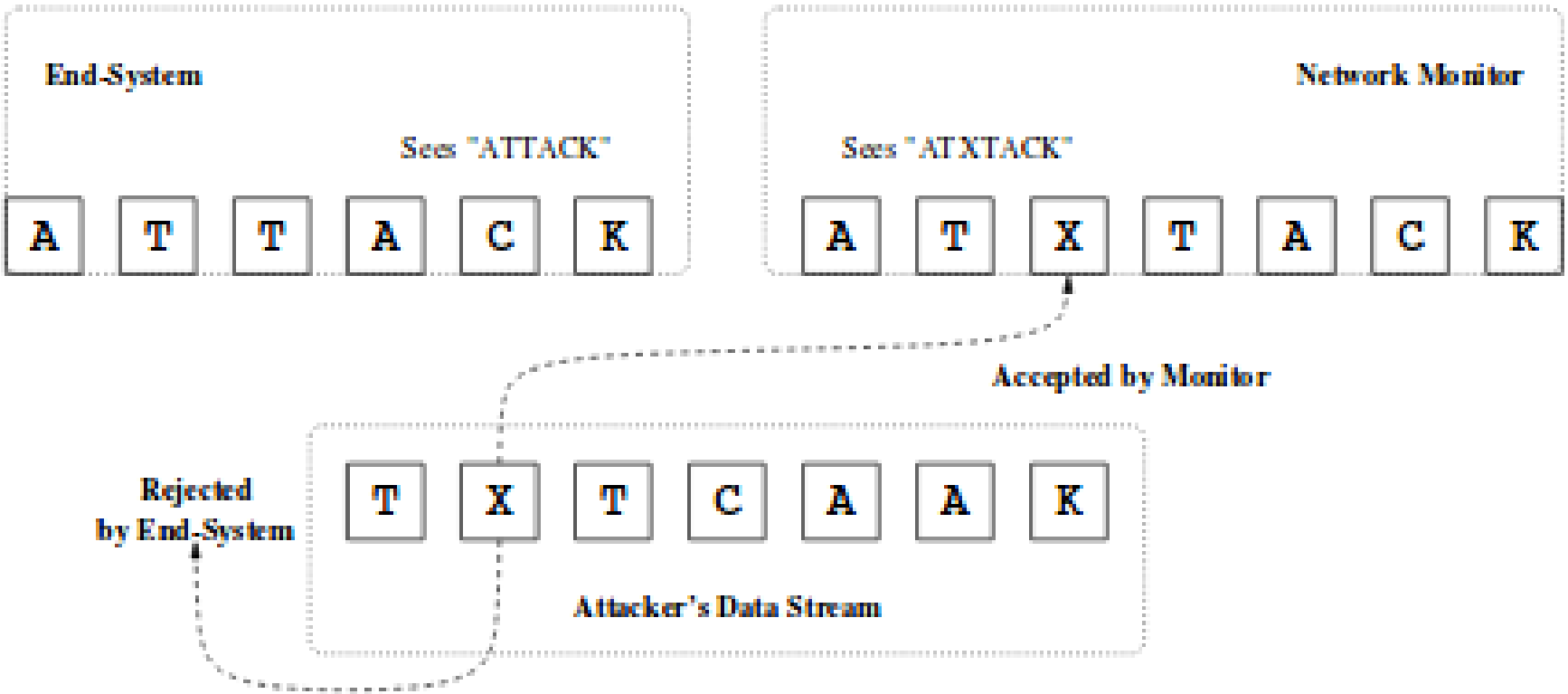


Figure 4: Insertion of the letter 'X'

Figure from Ptacek and Newsham

A not so useful distinction

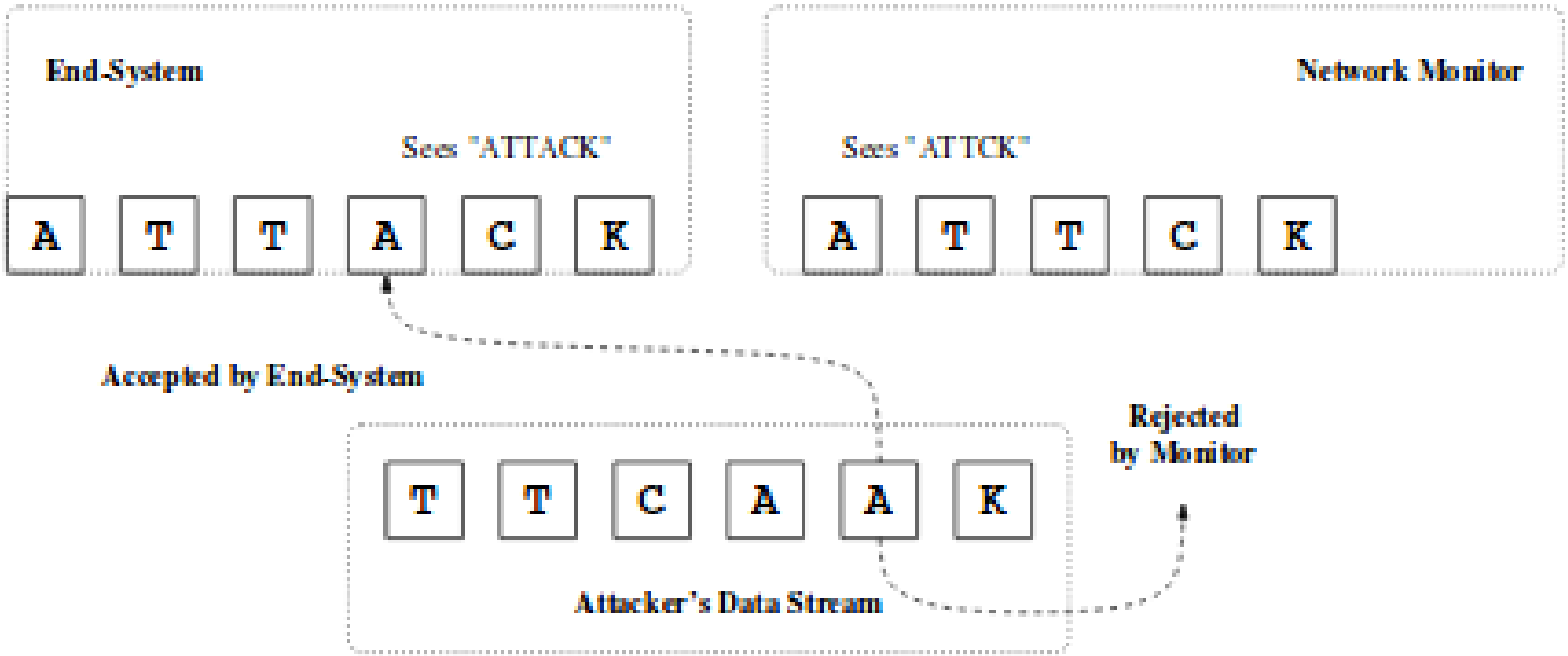


Figure 5: Evasion of the letter 'A'

Figure from Ptacek and Newsham

“Information only has meaning in that it is subject to interpretation”

–*Computer Viruses, Theory and Experiments by Fred Cohen, 1984*

“The only laws on the Internet are assembly and RFCs”

–*Phrack 65 article by julia@winstonsmith.info*

IP reassembly

- Routers (or endhosts, if they want) can break IP packets up into fragments that the receiver has to reassemble
- Ambiguity in the way overlapping IP fragments are put back together into an IP packet
- All of the following images were plagiarized from:

  https://www.sans.org/reading-room/whitepapers/detection/ip-fragment-reassembly-scapy-33969

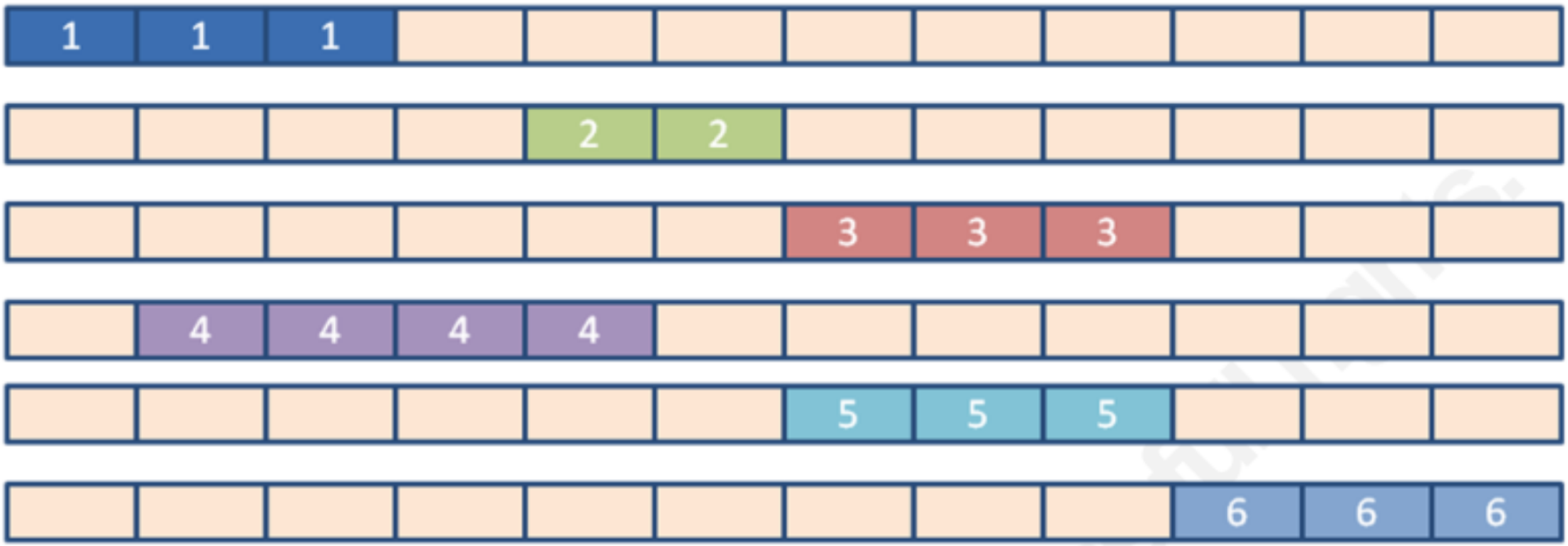


Figure 1: 6 Fragmented Packets (Shankar & Paxson, 2003)(Novak, 2005)

Reassembled using policy: First (Windows, SUN, MacOS, HPUX)

Reassembled using policy: Last/RFC791 (Cisco)

Reassembled using policy: Linux (Linux)

Reassembled using policy: BSD (AIX, FreeBSD, HPUX, VMS)

Reassembled using policy: BSD-Right (HP Jet Direct)

Figure 2: 5 Reassembly Methods (Shankar & Paxson, 2003)(Novak, 2005)

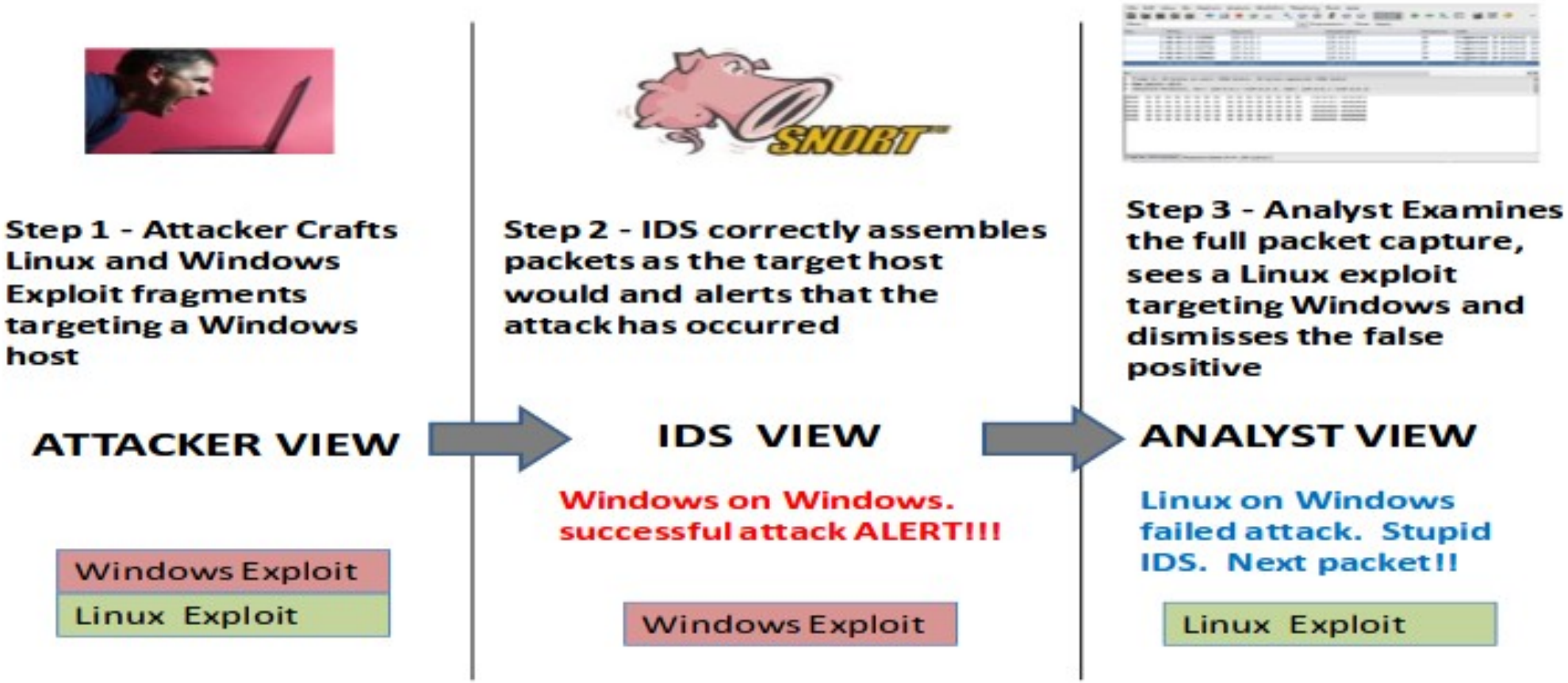


Figure 3: Views of the attacker, IDS and analyst

judyfrags.pcap - Wireshark

File Edit View Go Capture Analyze Statistics Telephony Tools Help

Filter: Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
1	08:40:13.533896	127.0.0.1	127.0.0.1	IP	Fragmented IP protocol (pr
2	08:40:13.534327	127.0.0.1	127.0.0.1	IP	Fragmented IP protocol (pr
3	08:40:13.534726	127.0.0.1	127.0.0.1	IP	Fragmented IP protocol (pr
4	08:40:13.535460	127.0.0.1	127.0.0.1	IP	Fragmented IP protocol (pr
5	08:40:13.535820	127.0.0.1	127.0.0.1	IP	Fragmented IP protocol (pr
6	08:40:13.536183	127.0.0.1	127.0.0.1	IP	[Illegal IP fragments]

Frame 6: 44 bytes on wire (352 bits), 44 bytes captured (352 bits)

Raw packet data

Internet Protocol, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)

```
0000  31 31 31 31 31 31 31 31  31 31 31 31 31 31 31 31   11111111 11111111
0010  31 31 31 31 31 31 31 31  34 34 34 34 34 34 34 34   11111111 44444444
0020  34 34 34 34 34 34 34 34  32 32 32 32 32 32 32 32   44444444 22222222
0030  33 33 33 33 33 33 33 33  33 33 33 33 33 33 33 33   33333333 33333333
0040  33 33 33 33 33 33 33 33  36 36 36 36 36 36 36 36   33333333 66666666
0050  36 36 36 36 36 36 36 36  36 36 36 36 36 36 36 36   66666666 66666666
```

Note the 111442333666 BSD reassembled payload

Wireshark's reassembly tab on the last fragment in the chain uses the BSD reassembly policy

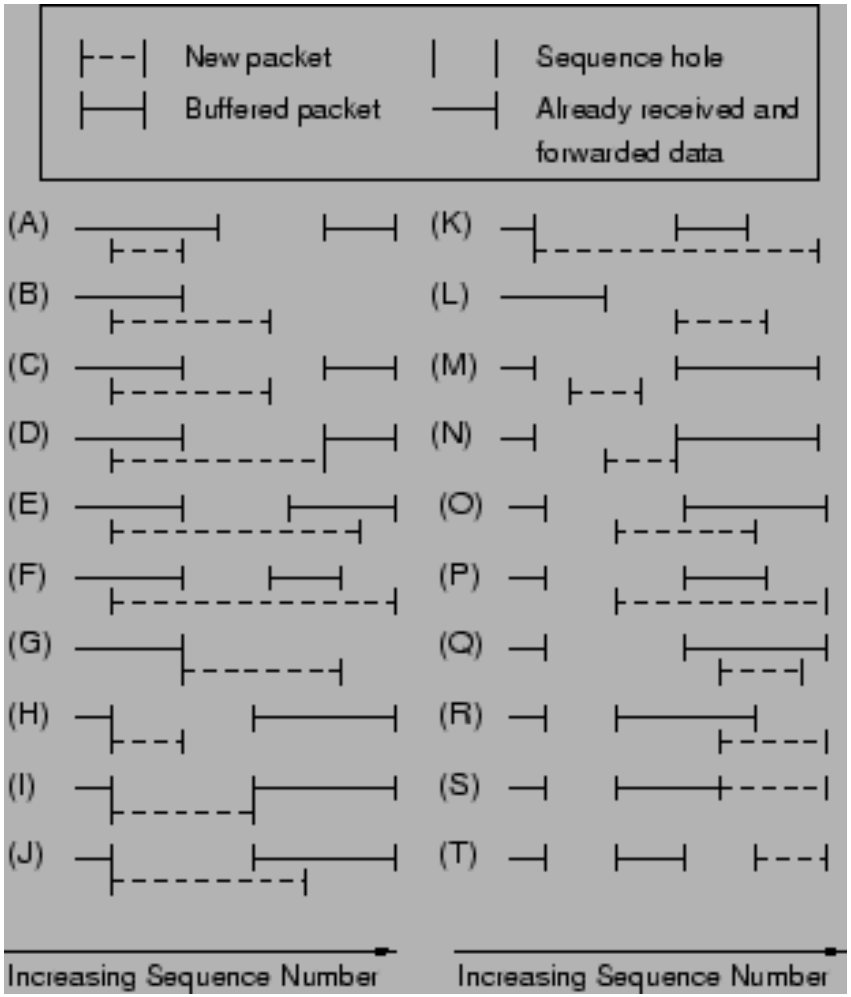
Frame (44 bytes) | Reassembled IPv4 (96 bytes)

File: "judyfrags.pcap" 384 Byte... | Packets: 6 Displayed: 6 Marked: 0 Load time: 0:00.000 | Profile: Default

Figure 4: Wireshark uses BSD reassembly technique

TCP is even worse...

- From http://www.icir.org/vern/papers/TcpReassembly/



Another example: TTL limiting

- Victim is 10 hops away from you (the attacker)
- IDS is 7 hops away from you, 3 from the victim
- Send a SYN with TTL 64
- Get a SYN/ACK from the victim
- Send a RST with TTL 9
- Send an ACK with TTL 64
- Victim sees SYN, sends SYN/ACK, gets ACK, you have an open connection and can send them data
- IDS sees SYN in one direction, SYN/ACK in the other, then RST and assumes the connection was reset, ACK and all packets that follow (with data) are ignored by the IDS

A layer 7 example (XSS) due to Jeff Knockel

- Suppose “`<script>...</script>`” is blacklisted
- Use “`<script>...`” instead, many browsers will happily run the script anyway despite the missing closing tag
- Information only has meaning in that it is subject to interpretation
  - IDS interprets things one way, web browser another

Physical layer injection

- From https://www.usenix.org/legacy/events/woot11/tech/final_files/Goodspeed.pdf

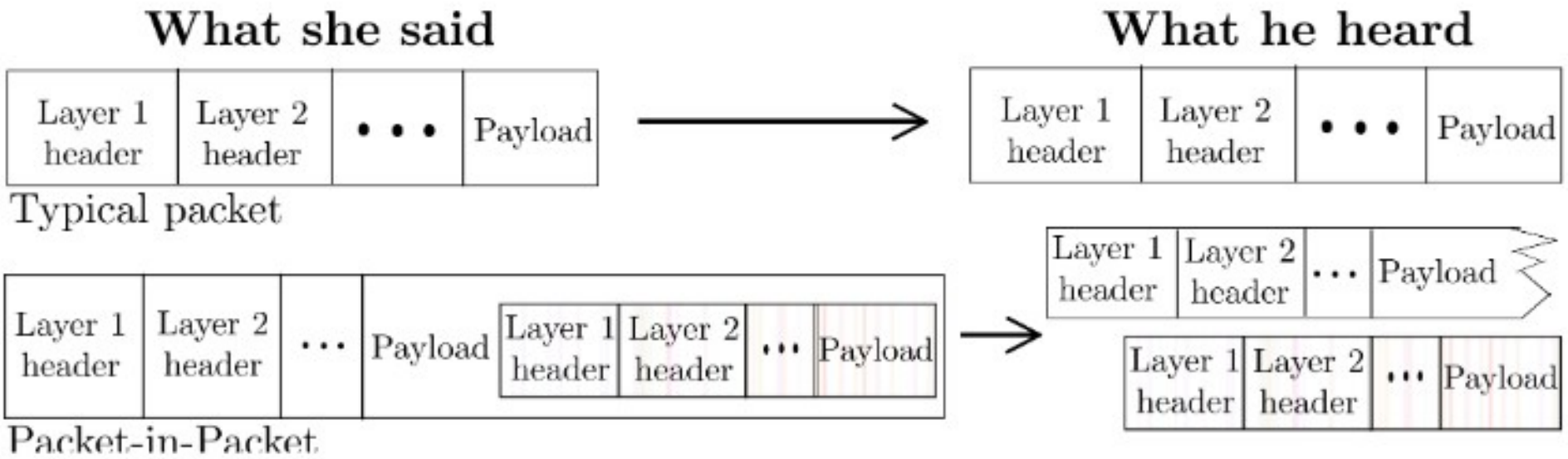


Figure 2: A typical packet's interpretation contrasted with that of a PIP.

Denial-of-Service (DoS) for IDS

- Exhaust the IDS's resources in some way
  - CPU
  - Memory
  - Bandwidth
- Fail-open (just let stuff through) *vs.* fail-closed (slow down the network)
- Example: On accident, Tony brought down the Computer Science Dept. network
- Other examples

DoS in general

- Exhaust some kind of resource, *e.g.*:
  - Optimistic ACK to exhaust bandwidth
    - See https://homes.cs.washington.edu/~tom/pubs/CCR99.pdf
  - PING of death (large PING) causes crash
  - Exhaust CPU in layer 7
  - More examples: http://www.isi.edu/~mirkovic/bench/attacks.html
  - SYN flood: Older hosts had either a fixed amount of half-open connections they could keep track of or no limitations at all, attack is to send lots of SYNs and never ACK or RST
    - Defenses: SYN backlog policies and SYN cookies

SYN cookies and SYN backlogs

- SYN cookies
  - Special kind of SYN/ACK
  - See https://cr.yp.to/syncookies.html
  - Can confirm ACK number and reconstruct the necessary state for a connection without having kept any state after sending the SYN cookie
- SYN backlog examples
  - Linux reserves ½, ¼, 1/8th, and so on for successively older SYNs, prunes 5 times a second
  - FreeBSD has 512 buckets of 30, you can't predict what bucket you fall into (in theory)

Resources

- Ptacek and Newsham, Insertion Evasion and Denial of Service: Eluding Network Intrusion Detection