Sections Week 7

Jason Zhang, Tapan Chugh

Administrivia

- Quiz tomorrow, make sure you review the recent slides about it.
- Assignment - 4 is due May 23rd.
- Project - 3 is due May 31st.

Internet Checksum

- Sum is defined in 1s complement arithmetic (must add back carries)
    - And it’s the negative sum
- “The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words …” – RFC 791
- In other words, it’s the value that when added to the header, the result is 0xffff

Example Problem 1

Message: 0xabc8983c1001bd02

Solution 1

```
  abc8
  983c
  1001
  bd02
--------        1) First sum normally
0x21107
--------        2) Back carry
0x1109
--------        3) Take one's complement
0xeef6
```

0xeef6

Example Problem 2

Message: 0xc2b4104a12001b01

Solution 2

```
  c2b4
  104a
  1200
  1b01
--------         1) First sum normally
0xffff
--------         2) Back carry
0xffff
--------         3) Take one's complement
0x0000
```

0x0000

Interesting Things to Note

- As stated earlier, the new sum of the header should be 0xffff
- Doesn’t check the order of the two byte blocks
- Must be recomputed every time the header changes, including with TTL decreases or when ECN is set

IPv4 header format

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Version				IHL				DSCP						ECN		Total Length															
4	32	Identification																Flags			Fragment Offset												
8	64	Time To Live								Protocol								Header Checksum															
12	96	Source IP Address																															
16	128	Destination IP Address																															
20	160	Options (if IHL > 5)																															
⋮	⋮																																
56	448																																

CRC

- Uses a generator polynomial and polynomial division to calculate a error-detecting code.
- For a polynomial of degree n, it creates a check of n bits.

Example Problem 1

Message: 0b10100110
Polynomial: $x + 1$

Solution 1

```
     1100010
  ----------
11)10100110
   11
   011
    11
    000
     00
     000
      00
      001
       00
       011
        11
        000
```

← Note: use XOR instead of minus.

← The actual remainder is 0, and thus the CRC remainder is 0.

obo

Example Problem 2

Message: 0b11100101
Polynomial: x3 + x2

Solution 1

```
              1
     1100)1000
          1100
          0100
```

← The actual CRC

```
          10111
 1100)11100101
      1100
      001001
        1100
        01010
         1100
         01101
          1100
          0001
```

← The actual remainder is 1, we add n bits then re-zero out to get CRC, done above.

0b100

Interesting Things to Note

- x + 1 as a generator polynomial results in a parity bit.
- Has the nice property of being easy to implement in hardware.
- Doesn't guard against intentional changing of data.

$$\mathrm{CRC}(x \oplus y) = \mathrm{CRC}(x) \oplus \mathrm{CRC}(y)$$