5.14 The IPv6 Addressing Scheme

We said that each IPv6 address occupies 128 bits (16 octets). The large address space guarantees that IPv6 can tolerate any reasonable address assignment scheme. In fact, if the community decides to change the addressing scheme later, the address space is sufficiently large to accommodate a reassignment.

It is difficult to comprehend the size of the IPv6 address space. One way to look at it relates the magnitude to the size of the population: the address space is so large that every person on the planet can have sufficient addresses to have their own internet three times as large as the current Internet. A second way to think of IPv6 addressing relates it to the physical space available: the earth's surface has approximately 5.1×10^8 square kilometers, meaning that there are over 10^{24} addresses per square meter of the earth's surface. Another way to understand the size relates it to address exhaustion. For example, consider how long it would take to assign all possible addresses. A 16-octet integer can hold 2^{128} values. Thus, the address space is greater than 3.4×10^{38}. If addresses are assigned at the rate of one million addresses every microsecond, it would take over 10^{20} years to assign all possible addresses.

5.15 IPv6 Colon Hexadecimal Notation

Although it solves the problem of having insufficient capacity, the large address size poses an interesting new problem: humans who manage the Internet must read, enter, and manipulate such addresses. Obviously, binary notation is untenable. The dotted decimal notation used for IPv4 does not make IPv6 addresses sufficiently compact either. To understand why, consider an example 128-bit number expressed in dotted decimal notation:

```
104.230.140.100.255.255.255.255.0.0.17.128.150.10.255.255
```

To help make addresses slightly more compact and easier to enter, the IPv6 designers created *colon hexadecimal notation* (abbreviated *colon hex*) in which the value of each 16-bit quantity is represented in hexadecimal separated by colons. For example, when the value shown above in dotted decimal notation is translated to colon hex notation and printed using the same spacing, it becomes:

```
68E6:8C64:FFFF:FFFF:0:1180:96A:FFFF
```

Colon hex notation has the obvious advantage of requiring fewer digits and fewer separator characters than dotted decimal. In addition, colon hex notation includes two techniques that make it extremely useful. First, colon hex notation allows *zero compression* in which a string of repeated zeros is replaced by a pair of colons. For example, the address:

```
FF05:0:0:0:0:0:0:B3
```