### Setting protocol fields
```python
>>> ip=IP(src="10.0.0.1")
>>> ip.dst="10.0.0.2"
```

### Combining layers
```python
>>> l3=IP()/TCP()
>>> l2=Ether()/l3
```

### Splitting layers apart
```python
>>> l2.getlayer(1)
<IP frag=0 proto/tcp>
>>> l2.getlayer(2)
<TCP>
```

# Explicit IP address (use quotation marks)
```python
>>> IP(dst="192.0.2.1")
```

# DNS name to be resolved at time of transmission
```python
>>> IP(dst="example.com")
```

# IP network (results in a packet template)
```python
>>> IP(dst="192.0.2.0/24")
```

# Random addresses with RandIP() and RandMAC()
```python
>>> IP(dst=RandIP())
>>> Ether(dst=RandMAC())
```

# Set a range of numbers to be used (template)
```python
>>> IP(ttl=(1,30))
```

# Random numbers with RandInt() and RandLong()
```python
>>> IP(id=RandInt())
```

### Show an entire packet
```python
>>> (Ether()/IPv6()).show()
###[ Ethernet ]###
dst = ff:ff:ff:ff:ff:ff
src = 00:00:00:00:00:00
type = 0x86dd
###[ IPv6 ]###
version = 6
tc = 0
fl = 0
plen = None
nh = No Next Header
hlim = 64
src = ::1
dst = ::1
```

### Show field types with default values
```python
>>> ls(UDP())
sport : ShortEnumField = 1025 (53)
dport : ShortEnumField = 53 (53)
len : ShortField = None (None)
chksum : XShortField = None (None)
```

# Randomize fields where applicable
```python
>>> fuzz(ICMP()).show()
###[ ICMP ]###
type = <RandByte>
code = 227
chksum = None
unused = <RandInt>
```

### Sending Packets
- **send(pkt, inter=0, loop=0, count=1, iface=N)**
  Send one or more packets at layer three
- **sendp(pkt, inter=0, loop=0, count=1, iface=N)**
  Send one or more packets at layer two
- **sendpfast(pkt, pps=N, mbps=N, loop=0, iface=N)**
  Send packets much faster at layer two using tcpreplay

### Sending and Receiving Packets
- **sr(pkt, filter=N, iface=N), srp(…)**
  Send packets and receive replies
- **sr1(pkt, inter=0, loop=0, count=1, iface=N), srp1(…)**
  Send packets and return only the first reply
- **srloop(pkt, timeout=N, count=N), srploop(…)**
  Send packets in a loop and print each reply

### Sniffing Packets
- **sniff(count=0, store=1, timeout=N)**
  Record packets off the wire; returns a list of packets when stopped

### Fuzzing
- **fuzz(ICMP()).show()**

### Sniffing Packets
- **sniff(count=0, store=1, timeout=N)**
  Record packets off the wire; returns a list of packets when stopped

```python
>>> pkts=sniff(count=100, iface="eth0")
>>> pkts
<Sniffed: TCP:92 UDP:7 ICMP:1 Other:0>
```