ft6 – Motivation

- next step: perform the tests
- usually tedious, error prone work
- aided by a tool
- easily repeatable
- enter ft6

ft6 – Agenda

1. overview
2. info on design and implementation
3. live demo
4. writing your own tests *(optionally)*

ft6 – Design Goals

- easy to configure
- graphical user interface
- browse tests and results
- visual representation

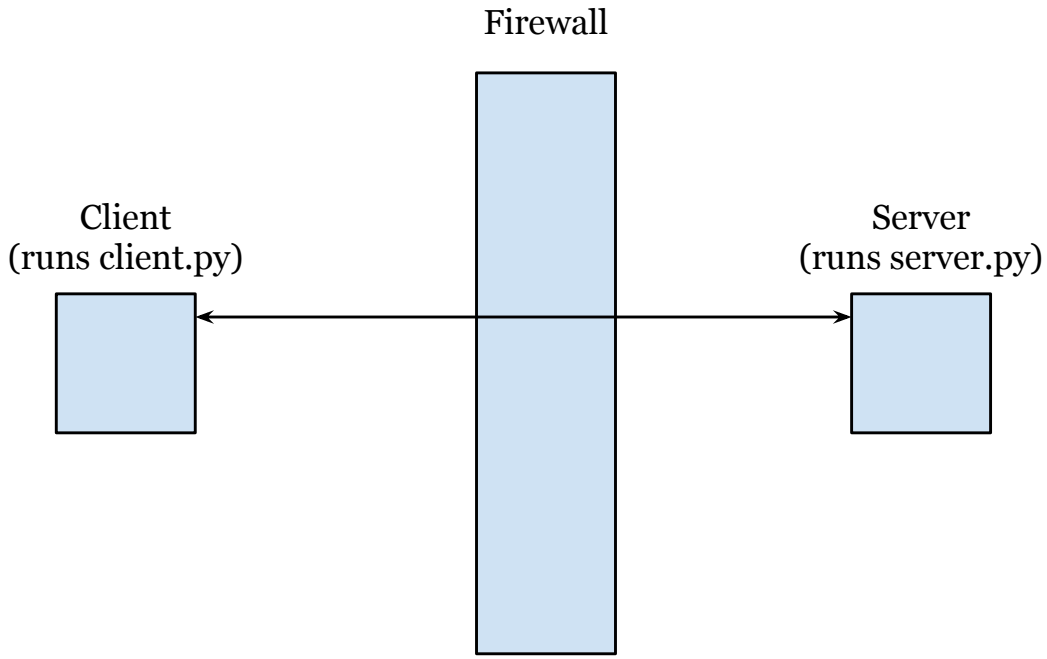
ft6 – Design Goals

- open-source (Creative Commons BY-NC-SA 3.0)
- can act as a framework for new tests
- easy to implement new tests

ft6 – Details

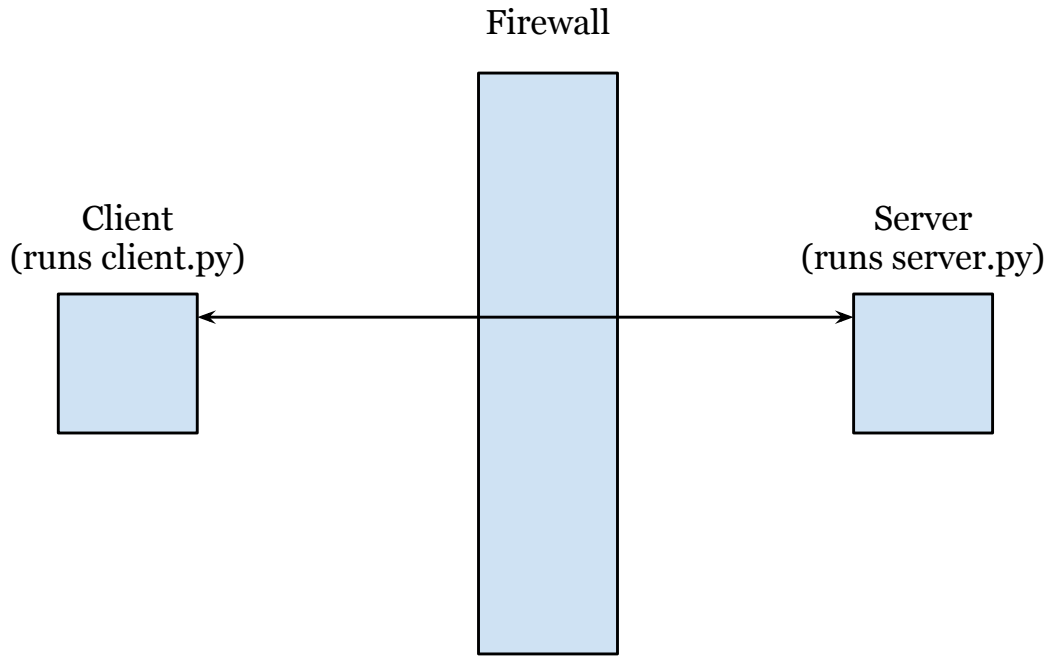
- powered by python, PyQt and scapy
- works with Linux, Windows 7, OS X
- python: rapid developement, easily understandable
- PyQt: GUI-framework, available cross-platform
  - http://www.riverbankcomputing.com/software/pyqt/intro
- scapy: great framework for network packet creation
  - http://www.secdev.org/projects/scapy/

ft6 – Architecture



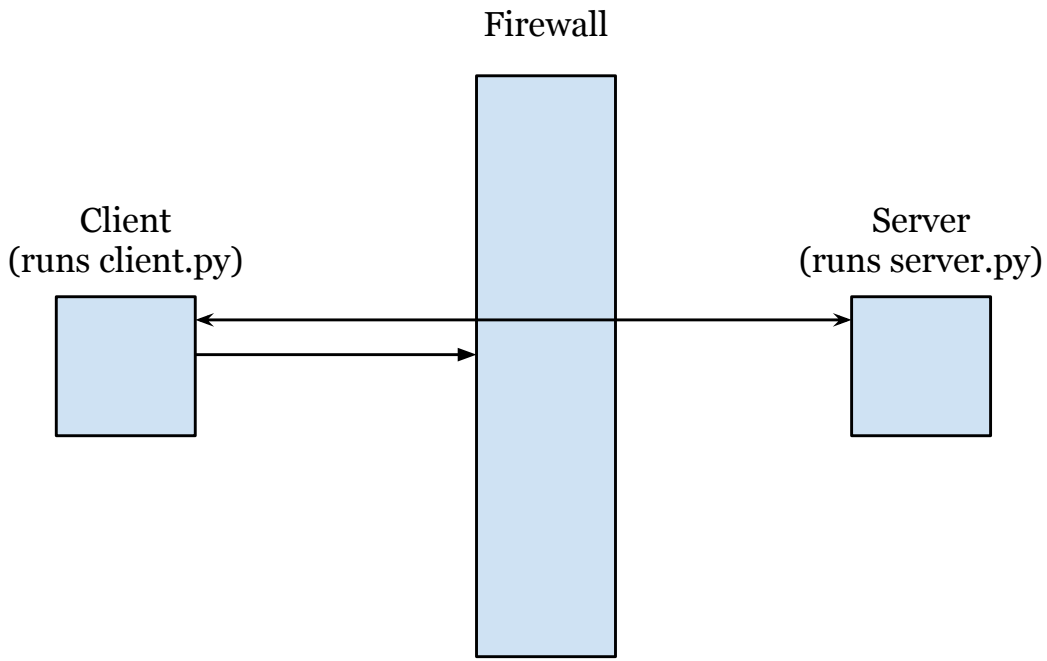
- ft6 is a client-server application
- requires machines on both sides of your firewall
- one open port
- place machines not more than one hop away from firewall

ft6 – Running ft6



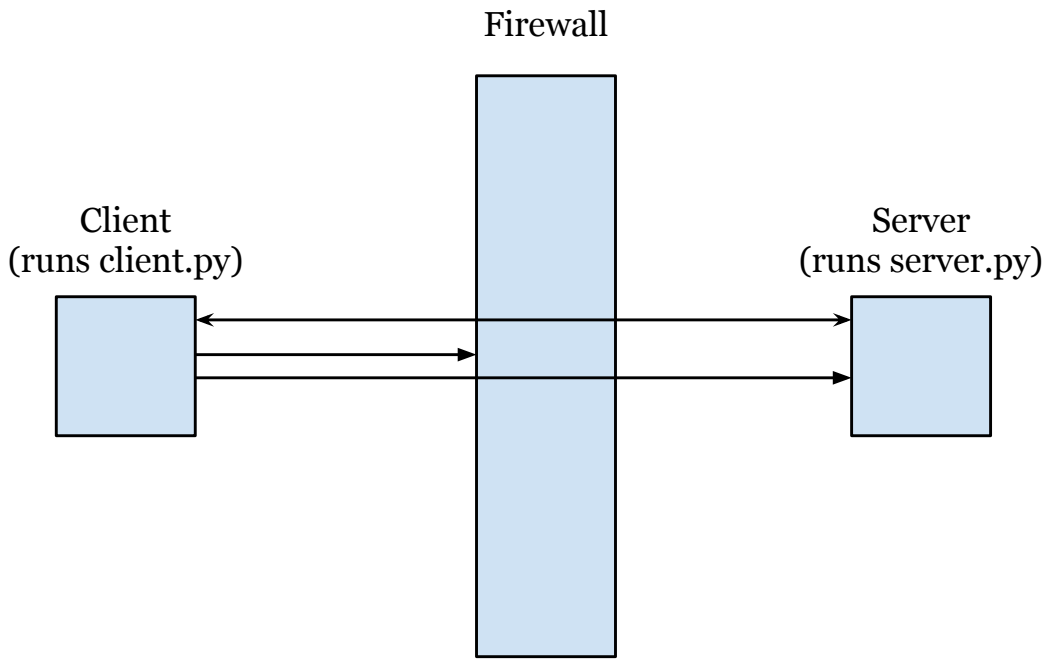
- Client and Server exhange control messages
  - Start / End / Results

ft6 – Running ft6



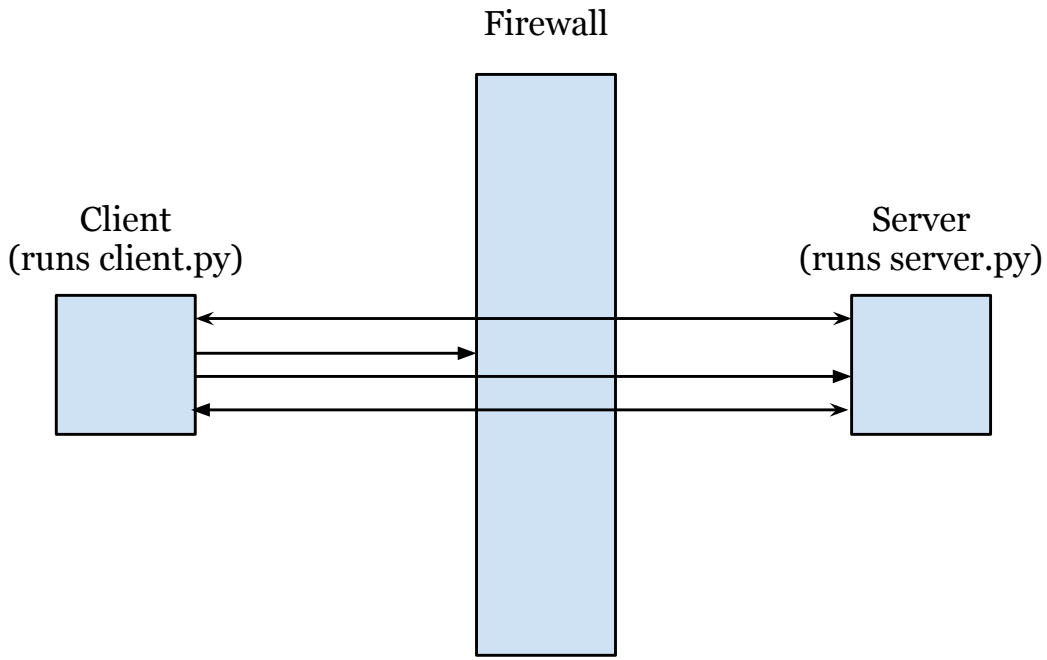
- Client sends packets
- Server sniffs

ft6 – Running ft6



- Client sends packets
- Server sniffs

ft6 – Running ft6



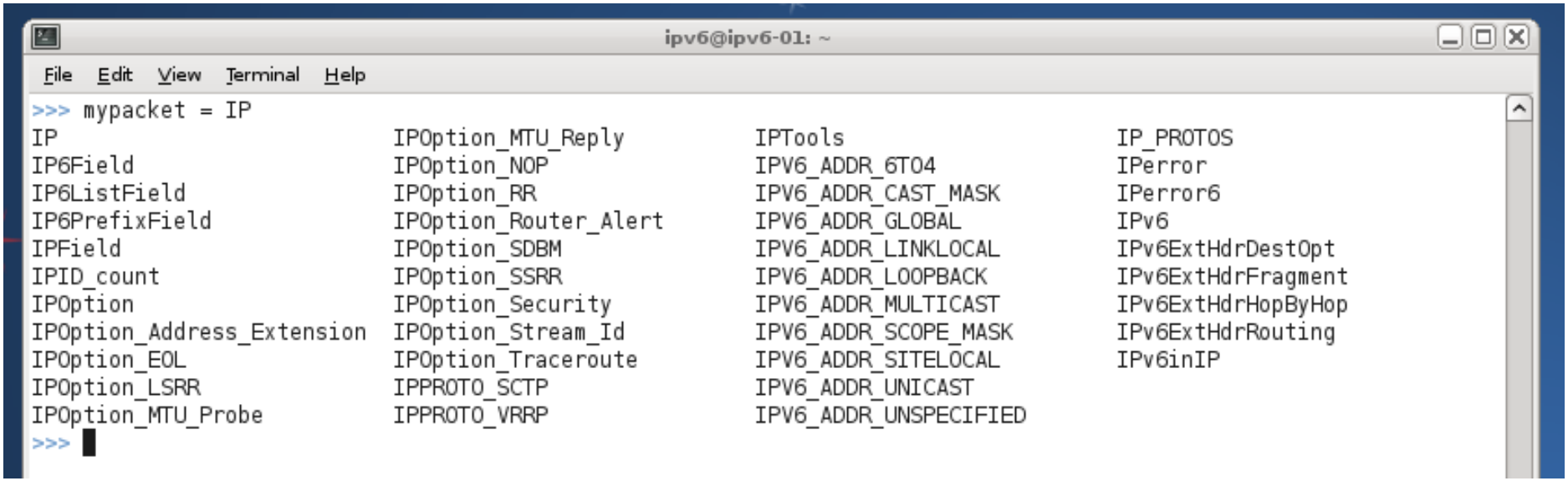
- Server sends back list of packets it recieved
- Client figures out what went missing and displays result

Handling Network Packets

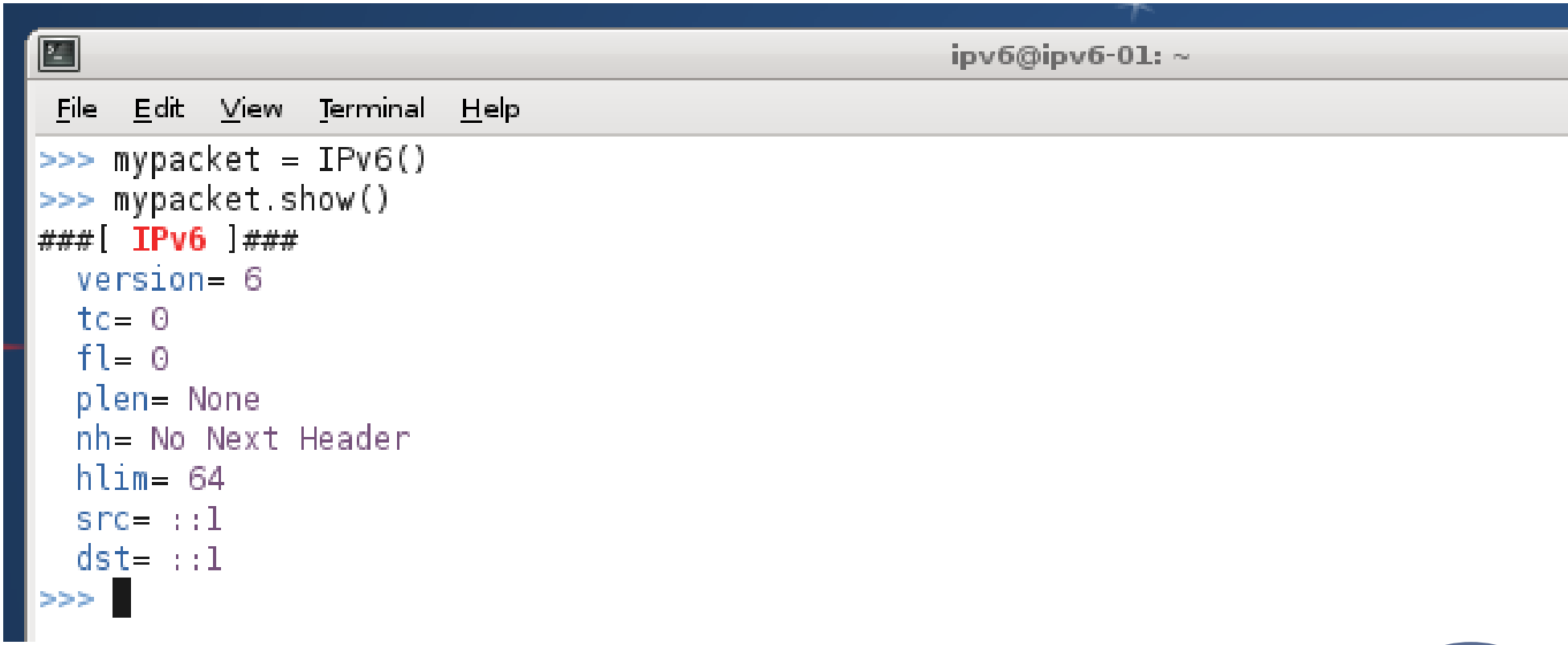
ft6 – packet creation with scapy

- handling network packets is usually messy
  - binary protocols
  - accessing individual flags invovles bitshifting or bitmasking
- sending and receiving is error-prone, too
- scapy does all that for you and is human readable.
- great TAB-completion

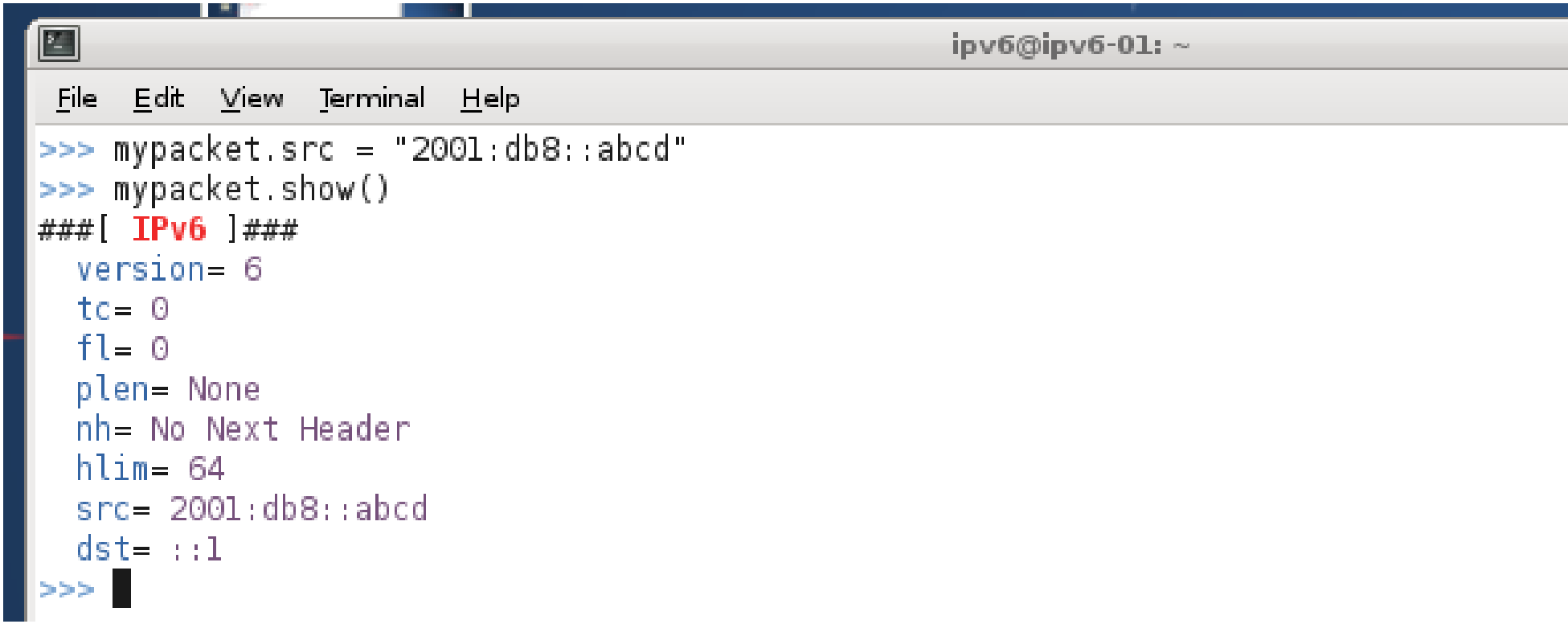
ft6 – scapy demo



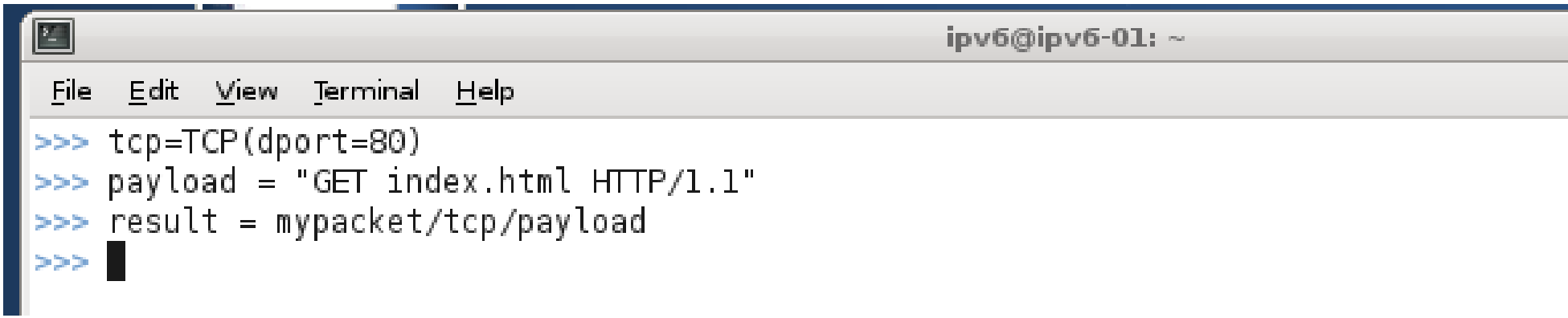
ft6 – scapy demo



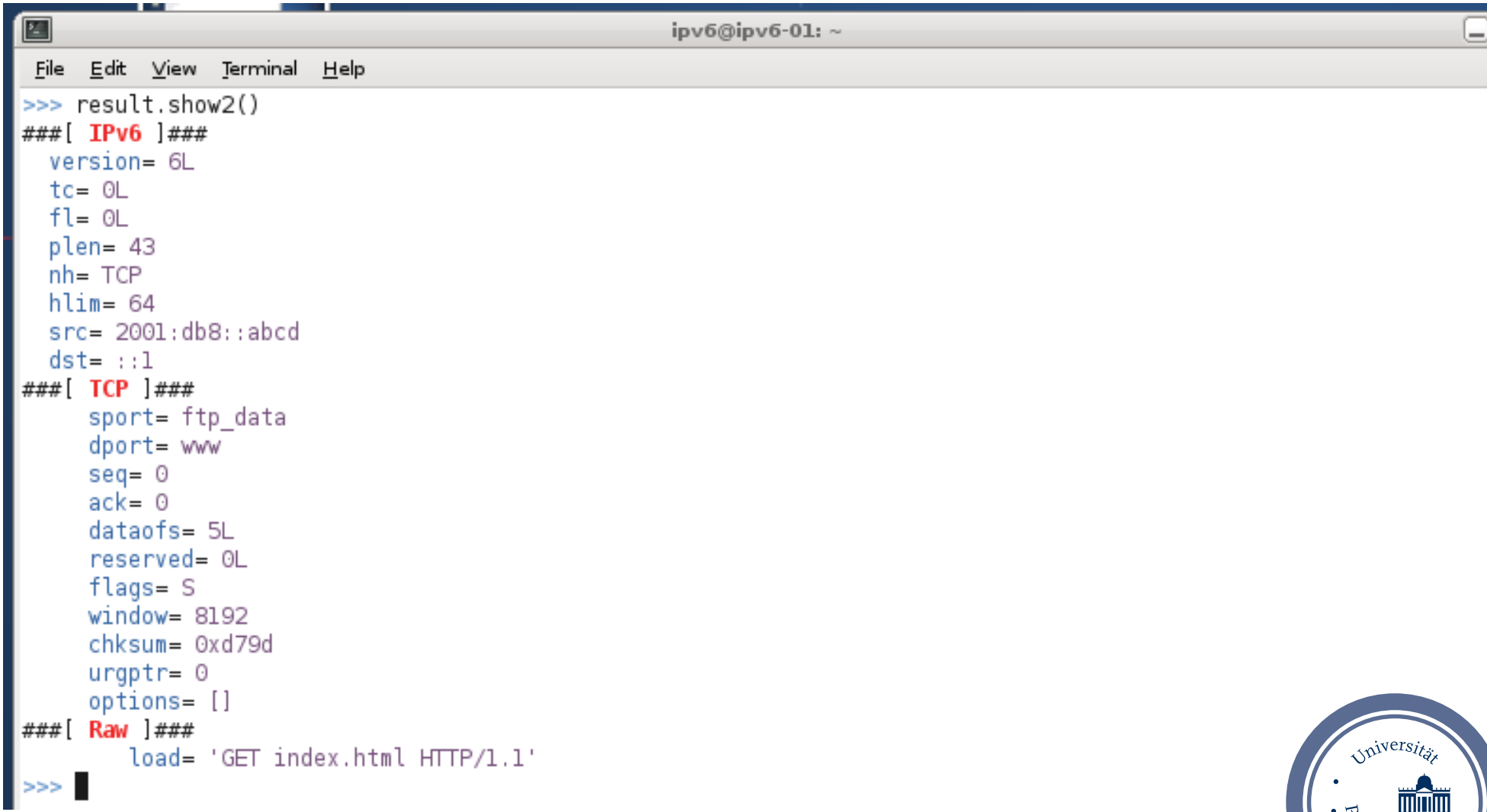
ft6 – scapy demo



ft6 – scapy demo



ft6 – scapy demo



Live Demo

ft6 – Writing your own test

Example: build own test, to see if packets containing the string "randomword"can traverse the firewall. Requires four steps:

1. create a class for your test
2. implement the `execute` method
3. implement the `evaluate` method
4. register your test with the application

(More detailed in ft6's documentation)

ft6 – Writing your own tests

Step 1: Create a class for your test

```
class TestRandomWord(Test):
  def __init__(self, id, name, description, test_settings, app):
    super(TestRandomWord, self).__init__(id, name, description,
      test_settings, app)
```

ft6 – Writing your own tests

Step 2: implement the `execute` method

```
def execute(self):
  e  = Ether(dst=self.test_settings.router_mac)
  ip = IPv6(dst=self.test_settings.dst, src=self.test_settings.src)
  udp= UDP(dport=self.test_settings.open_port, sport=12345)
  payload = "ipv6-qab"*128

  packet = e/ip/udp/(payload + "randomword")
  sendp(packet)

  packet = e/ip/udp(payload + "someotherword")
  sendp(packet)
```

ft6 – Writing your own tests

Step 3: implement the `evaluate` method

```
def evaluate(self, packets):
  results = []
  found_random = False
  found_otherword = False

   # iterate over the packets, filter those that belong to the test
   for p in packets:
      tag = str(p.lastlayer())
      if not "ipv6-qab" in tag:
          continue

      if "randomword" in tag:
          found_random = True

      if "someotherword" in tag:
          found_otherword = True
```

ft6 – Writing your own tests

Step 3: implement the `evaluate` method

```
# evaluate the flags
if found_random:
      results.append("Success", "Your firewall forwarded
      a packet with a random word!")
else:
      results.append("Failure", "Your firewall dropped
      a packet with a random word!")

if found_otherword:
    results.append("Warning", "Your firewall forwarded
    a packet with some other word. That's very weird!")
else:
    results.append("Success", "Your firewall dropped
    a packet with some other word. Well done firewall!")

return results
```

ft6 – Writing your own tests

Step 4: register your test

```
# create test classes, store them in the dictionary
# so they can later be called by their id
tICMP = TestICMP(1, "ICMPv6 Filtering", "The ICMP Test",
  self.test_settings, app)
...
tRandomWord = TestRandomWord(42, "My Random Word Test",
  "Tests for Random Words", self.test_settings, app)

self.tests = dict([
  (tICMP.id, tICMP), ..., (tRandomWord.id, tRandomWord)])
```

ft6 – future work

- ft6 is a work in progress
- lots of improvement could be done
- reducing the number of text fields
- better results
- more tests
- your thoughts: contact@idsv6.de
- ft6 is available at
  http://www.idsv6.de/Downloads/ft6-2013-05-22.tar.gz