Scapy

Packet manipulation in Python

What is Scapy?

Python library for

- Capturing packets
- Dissecting them
- And forging new ones

With minimal effort

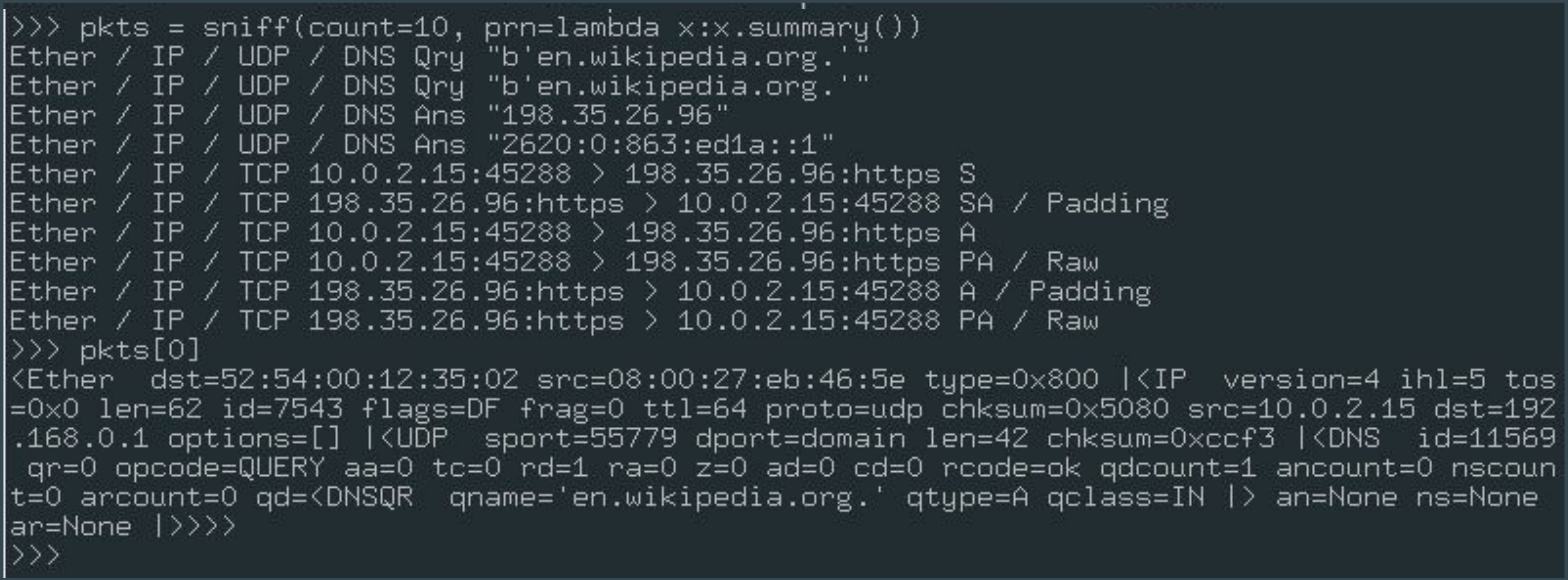
Why not Wireshark or [insert tool here]?

It may be a pain to do anything more than the developer(s) originally intended.

Scapy provides a convenient yet versatile interface for handling packets of many types at many layers.

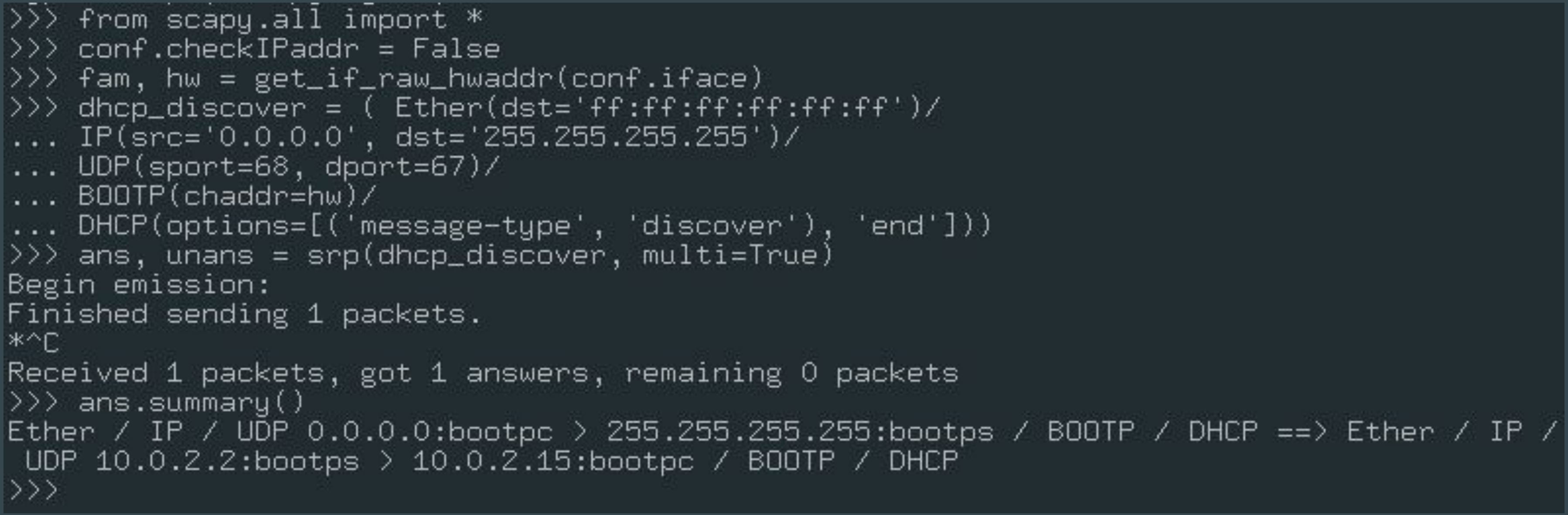
Result: you can quickly make something to do the exact job you need.

Example: packet capture



```
>>> pkts = sniff(count=10, prn=lambda x:x.summary())
Ether / IP / UDP / DNS Qry "b'en.wikipedia.org.'"
Ether / IP / UDP / DNS Qry "b'en.wikipedia.org.'"
Ether / IP / UDP / DNS Ans "198.35.26.96"
Ether / IP / UDP / DNS Ans "2620:0:863:ed1a::1"
Ether / IP / TCP 10.0.2.15:45288 > 198.35.26.96:https S
Ether / IP / TCP 198.35.26.96:https > 10.0.2.15:45288 SA / Padding
Ether / IP / TCP 10.0.2.15:45288 > 198.35.26.96:https A
Ether / IP / TCP 10.0.2.15:45288 > 198.35.26.96:https PA / Raw
Ether / IP / TCP 198.35.26.96:https > 10.0.2.15:45288 A / Padding
Ether / IP / TCP 198.35.26.96:https > 10.0.2.15:45288 PA / Raw
>>> pkts[0]
<Ether  dst=52:54:00:12:35:02 src=08:00:27:eb:46:5e type=0x800 |<IP  version=4 ihl=5 tos
=0x0 len=62 id=7543 flags=DF frag=0 ttl=64 proto=udp chksum=0x5080 src=10.0.2.15 dst=192
.168.0.1 options=[] |<UDP  sport=55779 dport=domain len=42 chksum=0xccf3 |<DNS  id=11569
 qr=0 opcode=QUERY aa=0 tc=0 rd=1 ra=0 z=0 ad=0 cd=0 rcode=ok qdcount=1 ancount=0 nscoun
t=0 arcount=0 qd=<DNSQR  qname='en.wikipedia.org.' qtype=A qclass=IN |> an=None ns=None
ar=None |>>>>
>>>
```

Example: find rogue DHCP server



```
>>> from scapy.all import *
>>> conf.checkIPaddr = False
>>> fam, hw = get_if_raw_hwaddr(conf.iface)
>>> dhcp_discover = ( Ether(dst='ff:ff:ff:ff:ff:ff')/
... IP(src='0.0.0.0', dst='255.255.255.255')/
... UDP(sport=68, dport=67)/
... BOOTP(chaddr=hw)/
... DHCP(options=[('message-type', 'discover'), 'end']))
>>> ans, unans = srp(dhcp_discover, multi=True)
Begin emission:
Finished sending 1 packets.
*^C
Received 1 packets, got 1 answers, remaining 0 packets
>>> ans.summary()
Ether / IP / UDP 0.0.0.0:bootpc > 255.255.255.255:bootps / BOOTP / DHCP ==> Ether / IP /
 UDP 10.0.2.2:bootps > 10.0.2.15:bootpc / BOOTP / DHCP
>>>
```

What else can we do with Scapy?

- ARP cache poisoning (to sniff on a switched network for example)
- Scanning with any kind of protocol they support or for which you can write the packets
- Checking for ICMP leaking/Ether leaking in padding (which may leak memory)

...and probably most other things you could imagine

Questions?

Sebastian Hanus

scapy.net

Contact information:

Email: sebastianxhanus@gmail.com

Twitter: @o0_shanus_0o

PGP: 4C1A 8369 1C08 A2EC 2D1B 78C9 C289 B9BA 1DDA CD26