Generierung von IPv6-Paketen mit Scapy

Tobias Rosenau

Berlin, 29.10.2010

Überblick – Scapy

- IPv6-Paket-Generator in Gestalt eines Kommandozeileninterpreters
- Plattformübergreifend Lauffähig (Python)
- Programmierkenntnisse sind nicht zwingend nötig

```
Datei  Bearbeiten  Ansicht  Terminal  Hilfe
tobias@backbuntu:~$ sudo scapy
[sudo] password for tobias:
INFO: Can't import python gnuplot wrapper . Won't be able to plot.
INFO: Can't import PyX. Won't be able to use psdump() or pdfdump().
Welcome to Scapy (2.1.0)
>>> sr1(IPv6(dst="fd11:100::1")/ICMPv6EchoRequest())
Begin emission:
.Finished to send 1 packets.
*
Received 2 packets, got 1 answers, remaining 0 packets
<IPv6  version=6L tc=0L fl=0L plen=8 nh=ICMPv6 hlim=255 src=fd11:100::
e793 |<ICMPv6EchoReply  type=Echo Reply code=0 cksum=0x4aa2 id=0x0 seq
>>>
```

Generieren von IPv6-Paketen

- Alle Anweisungen in eine Kommandozeile
- Step by step in der Kommandozeile
- Mittels Python Script

Generieren von IPv6-Paketen – ICMPv6 Ping

Alle Anweisungen können in eine Kommandozeile geschrieben werden. Scapy ergänzt, soweit möglich, nicht angegebene notwendige Optionen (z.B. Source-Adresse, Checksumme etc.).

Beispiel (Ping):

```
>>>sr1(IPv6(dst='fd11:100::1')/ICMPv6EchoRequest())
```

Generieren von IPv6-Paketen – ICMPv6 Ping

```
Datei  Bearbeiten  Ansicht  Terminal  Hilfe
tobias@backbuntu:~$ sudo scapy
[sudo] password for tobias:
INFO: Can't import python gnuplot wrapper . Won't be able to plot.
INFO: Can't import PyX. Won't be able to use psdump() or pdfdump().
Welcome to Scapy (2.1.0)
>>> sr1(IPv6(dst="fd11:100::1")/ICMPv6EchoRequest())
Begin emission:
.Finished to send 1 packets.
*
Received 2 packets, got 1 answers, remaining 0 packets
<IPv6  version=6L tc=0L fl=0L plen=8 nh=ICMPv6 hlim=255 src=fd11:100::
e793 |<ICMPv6EchoReply  type=Echo Reply code=0 cksum=0x4aa2 id=0x0 seq
>>>
```

Generieren von IPv6-Paketen – ICMPv6 Ping

Etwas Genauer:

- `sr1()` sendet Pakete auf Layer 3 Ebene

```
>>>help(sr1)
sr1 … send packets at layer 3 and return only the 1st
answer
```

- `IPv6()` erzeugt IPv6-Header
  - Zieladresse ist hier Übergabeparameter

- `ICMPv6EchoReply()` erzeugt ICMPv6-Paket bei welchem der Type bereits dem des Echo Requests entspricht (128)

Pakete step by step

Pakete step by step generieren:

Für komplexe Pakete empfiehlt es sich, diese Stück für Stück zu generieren. Dies ist deutlich übersichtlicher:

```
>>>i=IPv6()
>>>i.dst='fd11:100::1'
>>>q=ICMPv6EchoRequest()
>>>p=(i/q)
>>>sr1(p)
```

Pakete step by step – Optionen anzeigen

`ls()` listet alle von Scapy unterstützten Protokoll-Schichten auf.

Um den Echo Request weiter zu manipulieren ist es vorteilhaft zu wissen welche Optionen man überhaupt zur Verfügung hat.

ls(ICMPv6EchoRequest) liefert diese manipulierbaren Felder sowie deren Defaultwerte:

```
>>>ls(ICMPv6EchoRequest)
type : ByteEnumField = (128)
code : ByteField = (0)
cksum : XShortField = (None)
id : XShortField = (0)
seq : XShortField = (0)
data : StrField = ('')
```

Pakete step by step – Optionen hinzufügen / ändern

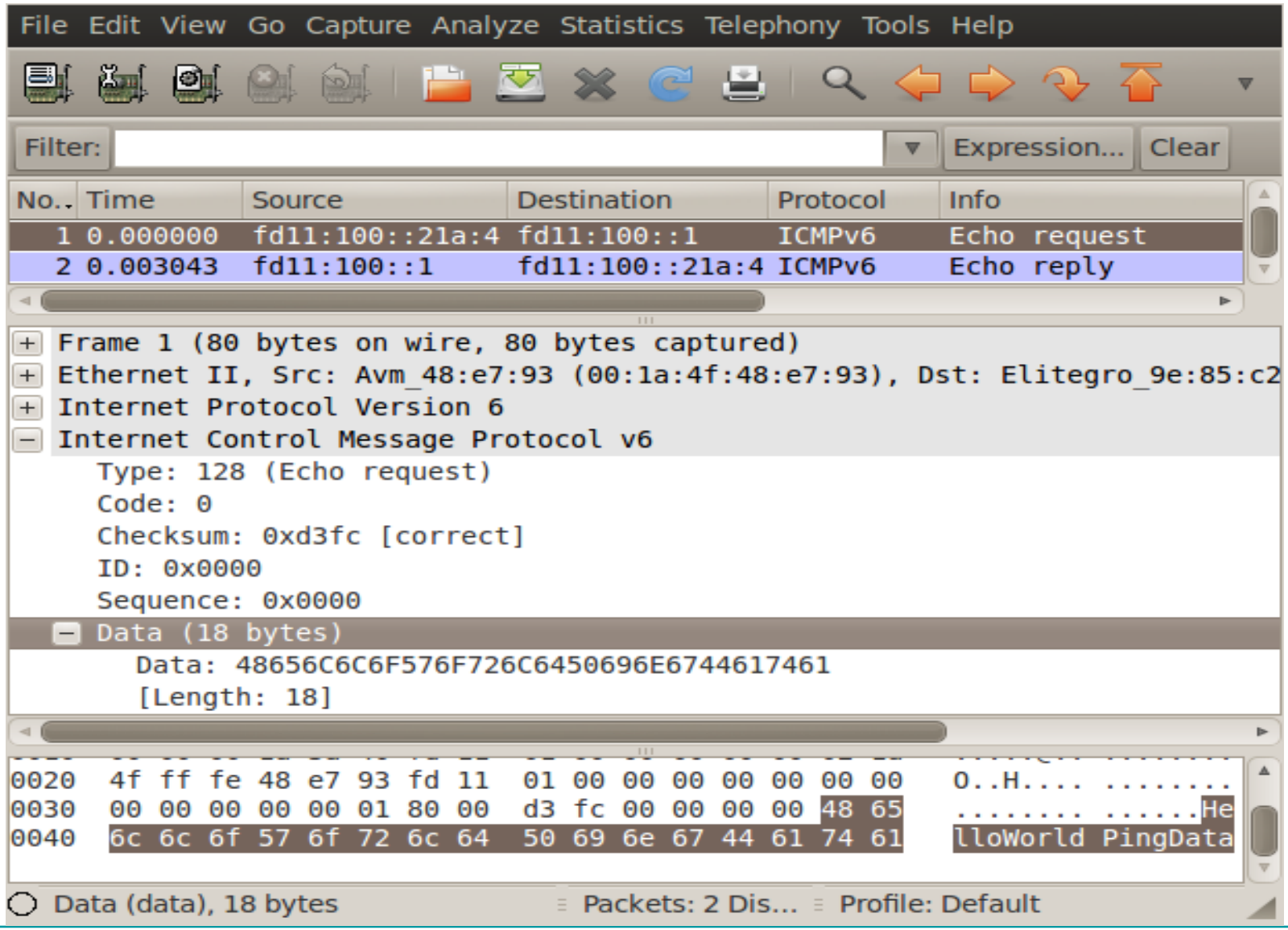
Echo Requests enthalten für gewöhnlich auch noch Daten und diese kann man nun ganz einfach hinzufügen mit:

```
>>>q.data='HelloWorldPingData'
```

Das Paket muss nun natürlich auch noch aktualisiert und gesendet werden:

```
>>>p=(i/q)
>>>sr1(p)
```

Pakete step by step – Ergebnis



send

Neben `sr1()` gibt es noch weitere Möglichkeiten ein Paket zu versenden:

```
>>>send(x, inter=0, loop=0, count=None, verbose=None, *args, **kargs)
```

- sendet Pakete, wie `sr1()` auf Layer 3 Ebene, wartet jedoch keine Antwort ab.
- bietet die Möglichkeit mehrere Pakete automatisch hintereinander zu senden

```
>>>send(IPv6(dst="fd11:100::1")/ICMPv6EchoRequest(),inter=3, count=9)
```

- Es wird hier alle 3 Sekunden ein Echo Request gesendet, insgesamt neun

sendp

- `sendp()` sendet Pakte auf Layer 2 Ebene

```
>>>sendp(Ether()/IPv6(dst='fd11:100::1')/ICMPv6EchoRequest())
```

- Ethernet-Teil ergänzt Scapy selbsttätig

```
>>>ls(Ether())
```

- zeigt die zur Verfügung stehenden Optionen für den Ethernet-Teil an.

Python Script – generiert alle ICMPv6 Typen & Codes

```
#! /usr/bin/env python
import sys
from scapy.all import *
target_host="2001:db8::1"
ip=IPv6()
icmp=ICMPv6EchoRequest()
ip.dst=target_host
# loop to create all ICMP Types and Codes
for type in range(0,256):
        for code in range(0,256):
                icmp.type=type
                icmp.code=code
                print type,code
                send(ip/icmp)
```

Pakete kompilieren und anzeigen

- `show()` zeigt nur gesetzte Optionen

```
>>> (IPv6(dst='ff02::1')/ICMPv6EchoRequest()).show()
###[ IPv6 ]###
  version= 6
  tc= 0
  fl= 0
  plen= None
  nh= ICMPv6
  hlim= 64
  src= fe80::21a:4fff:fe48:e793
  dst= ff02::1
###[ ICMPv6 Echo Request ]###
     type= Echo Request
     code= 0
     cksum= None
     id= 0x0
     seq= 0x0
     data= ''
```

- `show2()` kompiliert das Paket und berechnet so auch z.B. payload lenght und checksum

```
>>> (IPv6(dst='ff02::1')/ICMPv6EchoRequest()).show2()
###[ IPv6 ]###
  version= 6L
  tc= 0L
  fl= 0L
  plen= 8
  nh= ICMPv6
  hlim= 64
  src= fe80::21a:4fff:fe48:e793
  dst= ff02::1
###[ ICMPv6 Echo Request ]###
     type= Echo Request
     code= 0
     cksum= 0x4a42
     id= 0x0
     seq= 0x0
     data= ''
```

Pakete speichern / einlesen

```
>>>p=rdpcap('capture.pcap')
```

- p ist Array, jede Zeile entspricht einem Paket

```
>>>wrpcap(paket(e),'filename.pcap')
```

- schreibt Paket(-liste) in `filename.pcap`

Fragen

Fragen?

Quellen

http://www.secdev.org/projects/scapy/
(aktuelle Scapy Version)

SECURITY POWER TOOLS: O'Reilly Media, Inc., 2007
ISBN: 0-596-00963-1, 978-0-596-00963-2

http://www.secdev.org/projects/scapy/files/scapydoc.pdf

http://dirk-loss.de/scapy-doc/Scapy.pdf

http://www.packetlevel.ch/