



BEUTH HOCHSCHULE FÜR TECHNIK BERLIN  
University of Applied Sciences

# Scapy-GUI for IPv6

Paul Szameitpreiks  
Potsdam, 5. November 2011

GEFÖRDERT VOM



Bundesministerium  
für Bildung  
und Forschung



- Generierung von IPv6-Paketen mit Scapy
- Scapy GUI - Kurzvorstellung



## Generierung von IPv6-Paketen mit Scapy



- IPv6-Paket-Generator in Gestalt eines Kommandozeileninterpreters
- Plattformübergreifend lauffähig (Python)
- Programmierkenntnisse sind nicht zwingend nötig, aber vorteilhaft

```
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
>>> □
```



- Alle Anweisungen können in eine Kommandozeile geschrieben werden.
- Scapy ergänzt, soweit möglich, nicht angegebene notwendige Paketparameter (z.B. Source-Adresse, Prüfsummen, etc.).
- Direkte Einbettung und Weiterverarbeitung als Python Script ist möglich.

## Beispiel - ICMP-EchoRequest (Ping):

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



- `help()` gibt Informationen über Funktionen aus.

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

- `sr1` sendet Paket auf der Layer 3 Ebene und wartet nur auf die erste Antwort.



- Für komplexe Pakete empfiehlt es sich, die einzelnen Netzwerklayer jeweils nacheinander zu generieren und über Variablen verfügbar zu machen.
  - Komplexe Pakete lassen sich so Schritt-für-Schritt in der Kommandozeile erstellen.
  - Dies ist deutlich übersichtlicher und lässt sich später einfacher anpassen:

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



- `ls()` listet alle von Scapy unterstützten Protokoll-Schichten auf.
- Um den ICMP 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 = ('')
```



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

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

- Das Paket muss noch aktualisiert und gesendet werden:

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



- 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.

- `send()` bietet die Möglichkeit mehrere Pakete automatisch nacheinander zu senden:

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

Es wird alle 3 Sekunden ein 'Echo Request' gesendet, insgesamt neun mal

# Pakete kompilieren und anzeigen



show( ) zeigt die gesetzten Optionen des Pakets

show2( ) kompiliert das Paket und berechnet so auch z.B. Länge der Payload und Prüfsummen

```
>>> (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= ''
```

```
>>> (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= ''
```



## Scapy GUI - Kurzvorstellung

GEFÖRDERT VOM



Bundesministerium  
für Bildung  
und Forschung



## Motivation

- Einfache Bedienung von Scapy ohne Kommandozeileneingabe
- Flexible, einfache und schnelle Erstellung von IPv6 Paketen für Testzwecke im Netzwerk
- Lernprogramm für IPv6 und Scapy

## Optionen

- Einbindungsmöglichkeit von Erweiterungsheadern
- Auswahl verschiedener Next Header Typen und Payload
- Speichern, Laden der Einstellungen und Speicher in pcap Datei
- Senden von Paketen mit dem `sendp()` Befehl
- Speichern eines erstellten Sendecodes in der Zwischenablage
- Einbindung kleiner Tools (Round-Trip Time, Get lokal IPv6 Addresses)



## Installation

- Download der Scapy-GUI (Version 1.4) unter:

WWW: <http://code.google.com/p/scapy-gui-ipv6/downloads/list>

- Installation folgender Programme/Pakete:
  - Python
  - Scapy
  - QT 4

- Starten der GUI übers Terminal:

```
name@name:~$ sudo python gui.py
```



**Paul Szameitpreiks/Thomas Scheffler**

Fachbereich Elektrotechnik  
Beuth-Hochschule für Technik Berlin

Email: [scheffler@beuth-hochschule.de](mailto:scheffler@beuth-hochschule.de)  
WWW: [prof.beuth-hochschule.de/scheffler](http://prof.beuth-hochschule.de/scheffler)



# Anhang



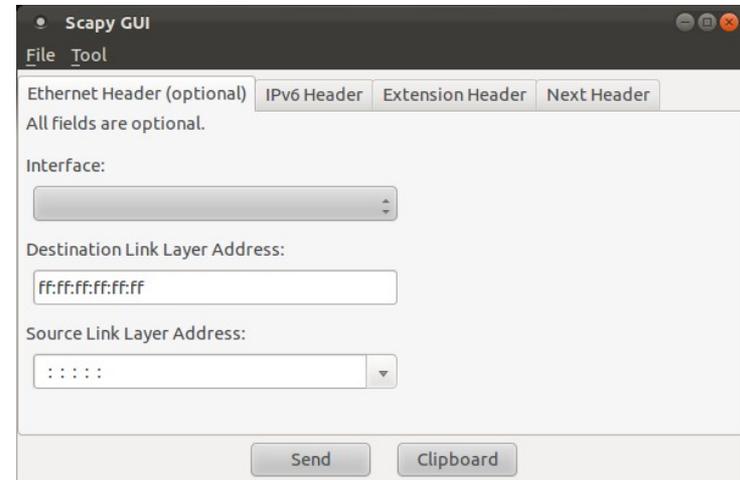
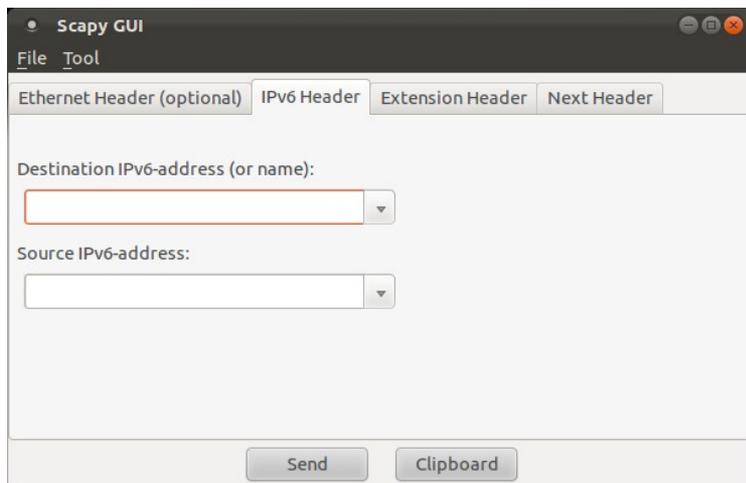
- <http://www.secdev.org/projects/scapy/>  
(aktuelle Scapy Version)
- SECURITY POWER TOOLS: O'Reilly Media, Inc., 2007  
ISBN: 0-596-00963-1, 978-0-596-00963-2
- <http://www.secdev.org/projects/scapy/files/scapydoc.pdf>
- <http://dirk-loss.de/scapy-doc/Scapy.pdf>
- <http://www.packetlevel.ch/>
- [www.thc.org/thc-ipv6/](http://www.thc.org/thc-ipv6/)



## Aufbau eines Pakets

### Ethernet Header (optional):

- Source und Destination Link Layer Adresse
- Interface (notwendig falls mehrere vorhanden sind)



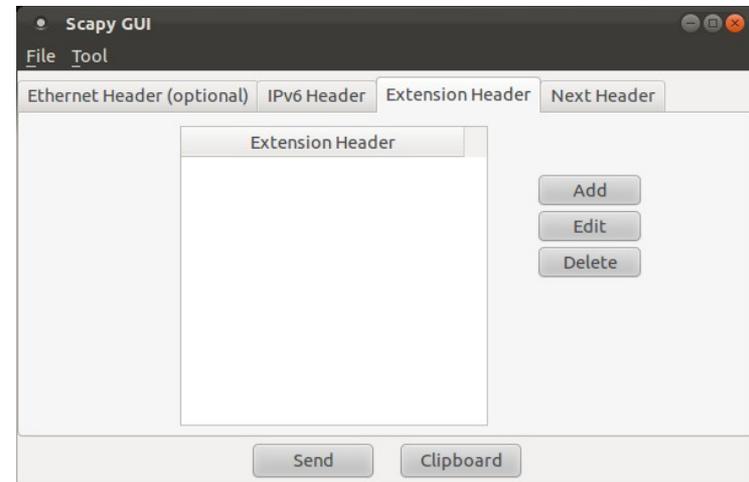
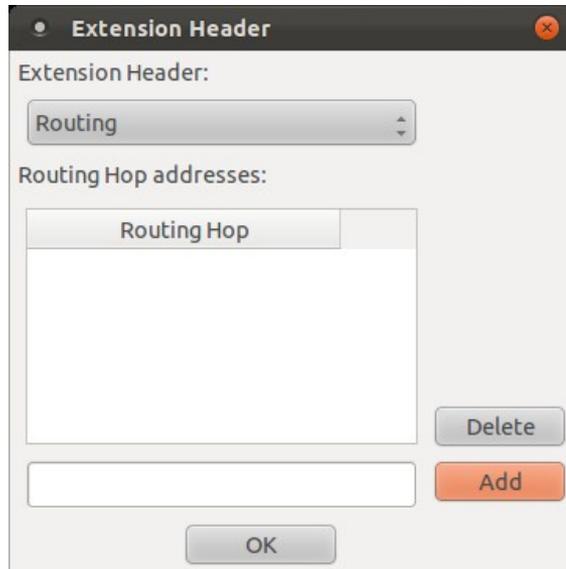
### IPv6 Header:

- Source und Destination IPv6 Adresse
- Auswahl von vordefinierten IPv6 Adressen mittels Drop-Down Menü



## Extension Header (optional):

- Auflistung der ausgewählten Extension Header in einer Tabelle
- Bearbeiten und Löschen von vorhandenen Extension Headern

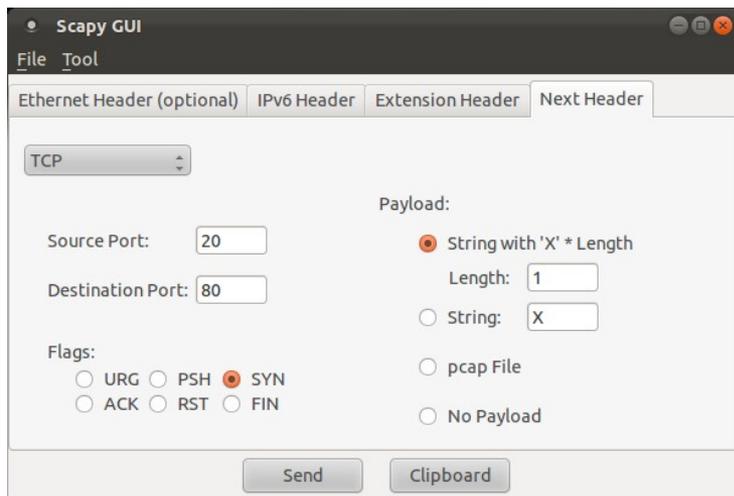
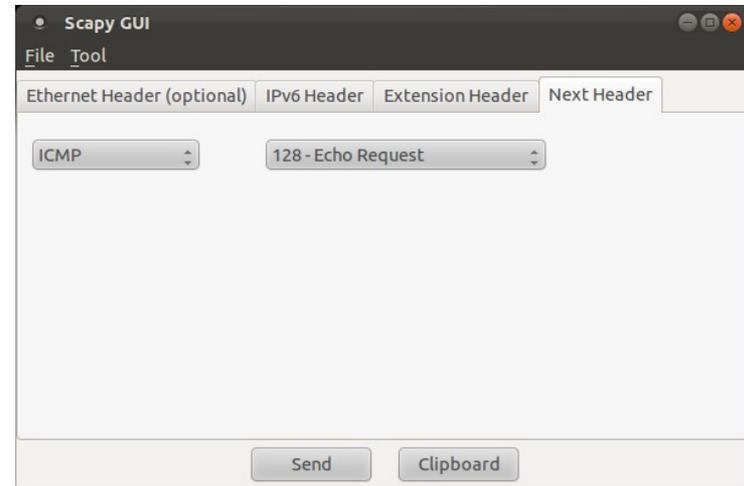


- Pop-up Fenster zur Auswahl verschiedener Extension Header
- Einzelne Extension Header mit unterschiedlichen Einstellmöglichkeiten
- Hier Routing Header Type 0



## Next Header:

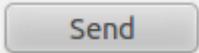
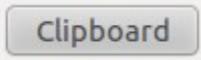
- Auswahl zwischen vier Next Header Typen
  - ICMP
  - TCP
  - UDP
  - No Next Header

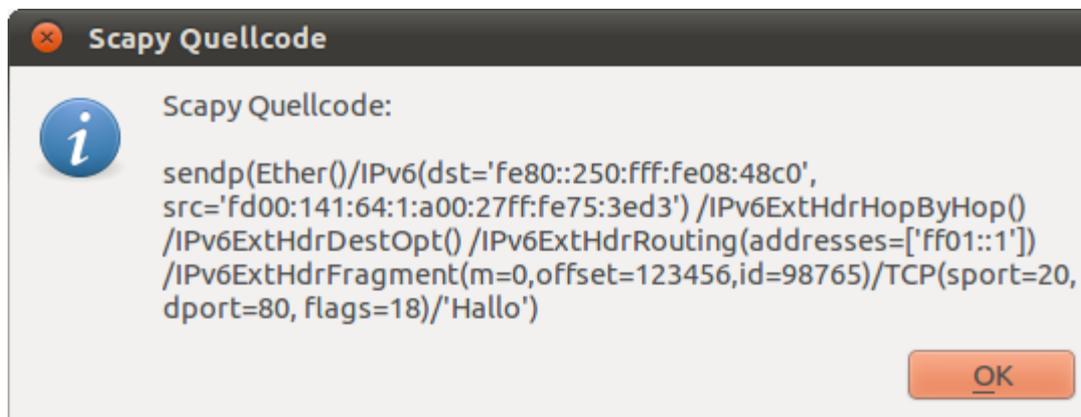
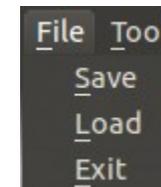


- ICMP mit sieben Untertypen
- TCP und UDP mit Payload Option und Angabe von Source- und Destination-Port
- TCP mit der Möglichkeit die einzelnen Flags zusetzen



## Weiterverarbeitung

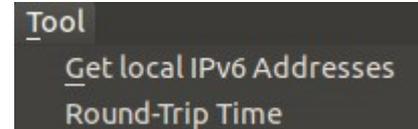
- Senden mittels `sendp()` Befehl 
- Speichern der Paketdatenstruktur in einer Pcap-Datei
- Speichern des Scapycodes in der Zwischenablage 
- Anzeige des Scapycodes in einer Info Message Box





## Kleine Tools

- Im Menüpunkt Tool sind zwei Werkzeuge auswählbar



- Get local IPv6 Addresses – Ermittelt mittels MLD und Hop by Hop options Header die lokalen IPv6 Adressen
- Round-Trip Time – senden ein oder mehrere Pings und berechnet die Round-Trip Time
- Pop-up Fenster zur Ermittlung der mittleren Round-Trip Time

