# OSI-3
## u23 2014

yanosz, florob, nomaster, rampone, ike, gevatter
thomas.wtf

Chaos Computer Club Cologne e.V.
https://koeln.ccc.de

Cologne
2014-10-13




ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>




ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# VPNs und “Darknets”

- Ziel: Freifunk-Netz an DN42, ChaosVPN, IC-VPN anbinden
- Umfeld: Routing, Cryptographie, VPN, X.509
- Technik: BGP, OpenVPN, Tinc, Bird/Quagga, Policy-Routing, Load-Balancing, puppet
- 1 Team



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# Angriffe im Freifunk-Netz

- Ziel: Angriffe im Freifunk-Netz erkennen und abwehren, z. B. DHCP-Starvation, ARP-Flooding, etc.
- Umfeld: Netzwerk-Diagnose, Monitoring
- Technik: Scripting (z. B. Scapy), Wireshark, thc-ipv6, etc.
- 1-n Teams



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# Netzwerk-Management/Monitoring

- Ziel: collectd um Plugins für TCP und Mesh-Zustandsdaten-Abfrage erweitern
- Umfeld: C-Programmierung, Script-Programmierung
- Technik: collectd, Perl, Ruby, Lua, etc.
- 1 Team



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# OpenWRT-Testbed

- Ziel: Testbed aus mehreren Router im Serverraum deployen
- Umfeld: Builds, Integrationstest, Netzwerke
- Technik: Linux, Virtualsierung, Jenkins, Arduino (z. B. Schaltung von QSS-Knöpfen), PoE, VLANS, serielle Konsole löten
- 1 Team



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# WLAN: Skalieren und Debuggen

- Ziel: Mit verschieden dichten Netzen experimentieren, Gestörte/freie Kanäle? Tool zur Ermittlung der Freifunktauglichkeit, Airtime messen
- Umfeld: WLAN, Meshing (mit Akkus), WLAN-Richtfunk
- Technik: Linux, horst, eigene Tools
- 1 Team



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# Dezentrales Kommunikations- & Informationssystem

- Ziel: WebApp basierte Übersicht als Startseite im Freifunk-Netz
- Umfeld: Anycast, (m)dns, P2P, Entwicklung, Kommunikation (Chat, IRC)
- Technik: mdns, C, Lua, JavaScript, IRC, HTTP, HTML, ggf. Cryptographie
- 1 Team



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# Layer 3

- Netzübergreifende Addressen
- Ende-zu-Ende Kommunikation
- Routing



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

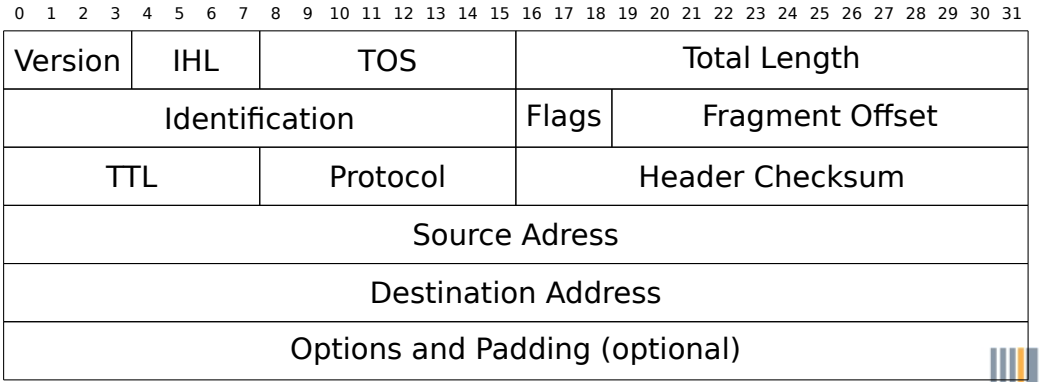



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# IPv4

- Internet Protocol
- Layer 3
- Paketbasiert
- Unterschiedliche “Netze” verbinden
- Netzübergreifende Addressen (IP-Adresse)
- Routing

# Pakete


0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
Version
IHL
TOS
Total Length
Identification
Flags
Fragment Offset
TTL
Protocol
Header Checksum
Source Adress
Destination Address
Options and Padding (optional)

# Layer 4

TCP / UDP

- Stream-/Paket-basierte Verbindungen

ICMP / ICMPv6

- Diagnose

# ICMP

- Internet Control Message Protocol
- Diagnose von IP-Netzen
- Ping
- Traceroute
- Path-MTU-Discovery
- Fehlermeldungen (Verworfene Pakete, etc.)



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# Adressen / Netze

- IP-Adresse: 4 Oktett (32 Bit)
- Schreibweise: `10.42.23.196`
- (Gewünscht: global eindeutig, → IPv6)
- Logische Unterteilung in (Sub-)Netze
- Prefix bezeichnet das Netz
- Notation:
  - `10.42.23.196/24` → die ersten 24 bits sind Prefix
  - Netmask `255.255.252.0` → die ersten 22 bits sind Prefix
  - ↳ CIDR: Classless Inter-Domain Routing
  - “In a land before time”: Class A/B/C

# Ethernet

- Am weitesten verbreitet für private Netzwerke
- Paket-orientiert
- MAC-Adressen (Medium Access Control)
  - 48 Bits: `C2:9F:DB:1B:08:44`
  - Global eindeutig
- Pakete sind Netz-lokal, für Kommunikation zwischen verschiedenen Netzen dient IP. (*Inter-net Protocol*)
  Siehe Routing



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# ARP

- Zuordnung von IP-Adressen zu MAC-Adressen
- ARP-Request als Broadcast
  - Anfrage; Wer hat diese IP-Adresse?
  - Broadcast-Adresse: `FF:FF:FF:FF:FF:FF`
- ARP-Reply an den Absender
- Antwort wird gespeichert im ARP-Cache
- `arp` oder `ip neigh`




ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# Allgemein

- Motivation: Ende des IPv4 Adressraums absehbar
- Entwicklungsbeginn von IPng Anfang der 1990er
- RFC 1883 veröffentlicht im Dezember 1995
- aktuelle Version: RFC 2460 vom Dezember 1998
- IPv5? Reserviert für ST2, nie mehr als experimentell

# Unterschiede zu IPv4

- 128 bit Adressen
- fester 40 Oktett Header
- minimale MTU von 1280 Oktetts
- Fragmentierung nur an Endpunkten

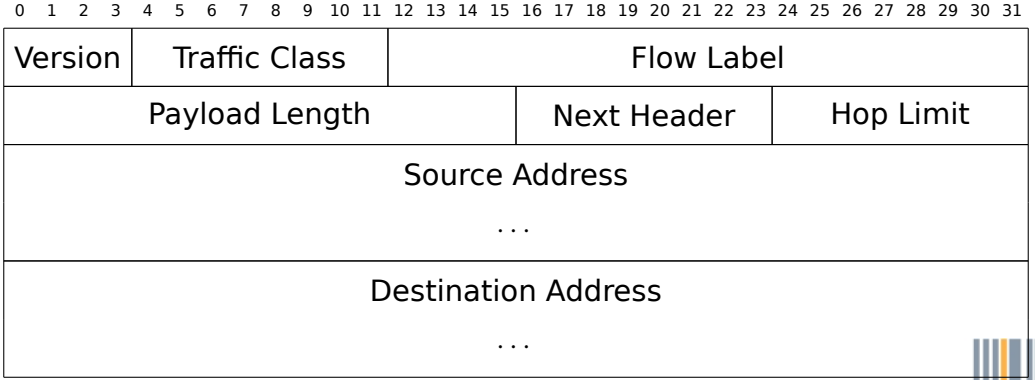
# Adressformat

- 8 mit : getrennte hexadezimale Oktettpaare
- führende Nullen können wegfallen
- eine Folge von Nullen kann durch :: ersetzt werden
- `2001:0db8:0000:0000:0000:0000:0000:efef`
- `2001:db8::efef`
- um einen Port anzugeben, in [] einschließen
- `[2001:db8::efef]:8080`

# Adressarten

| | |
|---|---|
| Unspecified | `::/128` |
| Loopback | `::1/128` |
| Multicast | `ff00::/8` |
| Link-Local Unicast | `fe80::/10` |
| Global Unicast | alles andere |

# IPv6 Header



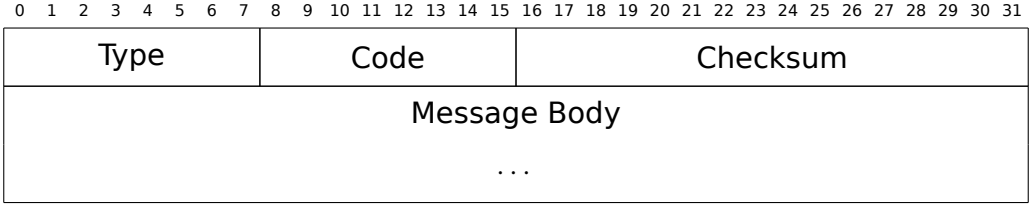


ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# ICMPv6

- Internet Control Message Protocol
  for the Internet Protocol Version 6
- RFC 4443
- geschachtelt in IPv6 Header
- Next Header = 58
- error messages (z. B. Packet Too Big)
- informational messages (z. B. Echo Reqequest/Response)

# ICMPv6 Header



# NDP

- Neighbor Discovery Protocol
- RFC 4861
- Unter-Protokoll von ICMPv6
- ersetzt ARP
- kann DHCP ersetzen
- verbreitet Informationen über Router



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# NDP — Neighbor Solicitation

- ARP Ersatz
- verwendet IPv6 Multicast
- Multicast Adresse: `ff02::1:ffXX:XXXX`
  berechnet aus den niederwertigsten 24 bit der Zieladresse
- nutzt MAC Multicast: `33:33:XX:XX:XX:XX`
- Switche ohne Multicastfähigkeit broadcasten (NIC/OS filtert)



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# NDP — Router Advertisement

- Router im Netzwerk teilen ihre Existenz mit
- periodisch
- auf Anfrage via Router Solicitation (neues Gerät im Netzwerk)
- enthält u. a. IP des Routers und bekannte Präfixe

# Stateless Address Autoconfiguration

- RFC 4862
- Ziele:
  - Keine manuelle Konfiguration von Endgeräten
  - Minimale Konfiguration von Routern

# Stateless Address Autoconfiguration

1. Zuweisen einer link-local Adresse
   1. `fe80::` + 64 bit Interface ID
   2. prüfen ob die Adresse bereits verwendet wird (Neighbor Solicitation)
   3. Wenn ja: manuelle Konfiguration
2. Router Advertisements sammeln (warten, oder Router Solicitation senden)
3. Adressen = Präfixe für Autokonfiguration + Interface ID
4. prüfen ob die Adresse bereits verwendet wird (Neighbor Solicitation)

# DHCPv6

- RFC 3315
- kann alternative to Stateless Autoconfiguration Adressen vergeben
- kann zusammen mit Stateless Autoconfiguration verwendet werden
  - nur Adressen vergeben (Router über NDP)
  - keine Adressen vergeben, NTP, DNS Server verteilen




ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# Routing

- Pakete zwischen verschiedenen Netzen verschicken
- Braucht einen Rechner, der in beiden Netzen ist (2 Interfaces) dieser leitet Pakete weiter (Router)
- Längere Strecke, mehr Netze $\rightarrow$ mehr Router (Hops)

# Routingtabelle

- Zuordnung welche Pakete wo lang müssen / Welche Knoten über welchen Weg erreicht werden können
- Einträge:
  - kleinere Netze bevorzugt
  - Zielangabe: Einzelne IPs, Netze, oder default (`0.0.0.0`)
  - Route:
    - Über ein Gateway (Router) angegeben über IP-Adresse
    - Lokal an einem Interface
- `ip route`
- Linux routet standardmäßig keine Pakete

  `sysctl -w net.ipv4.ip_forward=1` oder

  `echo 1 > /proc/sys/net/ipv4/ip_forward`

  Permanent: `/etc/sysctl.conf`



ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# NAT 😈

- Network Address Translation
- Problem: wenige globale Adresse, viele lokale
- Beispiel:

| LAN | 1-to-1 NAT | Masquerading |
|---|---|---|
| 192.168.0.1 | 1.42.23.11 | 1.42.23.11:10000–15000 |
| 192.168.0.2 | 1.42.23.12 | 1.42.23.11:16000–20000 |
| 192.168.0.3 | 1.42.23.13 | 1.42.23.11:21000–25000 |




ike <ike@koeln.ccc.de, florob <florob@babelmonkeys.de>

# DHCP

- IP-Adresskonfiguration
- “Verleiht” IP-Adressen (Lease) (IPv6: Stateful configuration)
- Mit begrenzter Lebensdauer (Lease-time)
- Und weitere Informationen (DNS-Server, Gateway, ...)
- Clients fragen per Broadcast (Request), Server gibt ein Angebot (Offer)

# radvd

- Router Advertisement Daemon
- implementiert NDP
- sendet Router Advertisements/beantwortet Router Solicitation
- verteilt Präfix, DNS server, DNS search list