“Einführung in TCP/IP”

Hands-on Session, 22.02.2006
Entwicklercamp 2006

Michael Holzt
keleos internet services
holzt@keleos.net

①

Einleitung

Vorstellung

- Name: Michael Holzt (M.Sc.)

- selbstständig/freiberuflich tätig
- Schwerpunkte: Systementwicklung, Embedded Systems, Internet-Technologie

- Entwickler bei Debian/GNU Linux seit 1999 und in verschiedenen OpenSource-Projekten involviert
- Autor (Fachartikel, Buchbeiträge)

Gestaltung des Workshops (1)

- Einführung in TCP/IP anhand praktischer Beispiele / Sniffing
- Abwechselnd theoretische Betrachtung und praktische Erprobung

- Teil 1: Grundlagen: Geschichte, IP-Adressen, ICMP, UDP, TCP
- Teil 2: Fortgeschrittene Themen: Verschlüsselung mit SSL/TLS, IPSEC als Alternative, evtl. Routing

Gestaltung des Workshops (2)

- Praktisches Sniffing unter Linux (von CD)
- CD-Basis: GRML (ähnlich wie Knoppix), angepasst für den Workshop

- Sniffing mit Ethereal (auch für Windows verfügbar, sehr empfehlenswert)
- Weiter installiert: Webserver (inkl. SSL), Mailserver (inkl. TLS), DNS-Server

- CD kann gerne mitgenommen werden!

②

Workshop Teil 1:
TCP/IP für Einsteiger

Geschichte von TCP/IP

- “ARPANET” als Projekt der ARPA/DARPA, einer Forschungseinrichtung des US-Militärs ab ca. 1972, Prototypen (IPv1/2/3/4) ab 1975, ARPANET komplett mit IPv4 ab 1983
- Entwicklungsziel: Zivile/Wissenschaftliche Nutzung verteilter Rechnerresourcen.
- Nur ein Märchen: Das Internet war weder als atombombensicheres Netz geplant, noch ist es dies jemals gewesen.

IP-Adressen und Routing (1)

- Vor Betrachtung der Protokolle: Wie kommen Pakete generell von A nach B?
- Datenpakete haben Absender und Empfänger die als sogenannte “IP-Adresse” dargestellt sind (ähnlich einer Telefonnummer), Auflösung über DNS
- IP-Adressen bei IPv4: vier 8 Bit (0-255) Zahlen die mit Punkten getrennt werden. Beispiel: 217.72.195.42 (www.web.de)

IP-Adressen und Routing (2)

- Die Gesamtheit der IP-Adressen wird in kleinere Blöcke aufgeteilt, die an die verschiedenen Nutzer verteilt werden (über Regionale und Lokale Internet Registry) und vom Nutzer weiter unterteilt werden können (Subnetze).
- Innerhalb eines Subnetzes können sich alle IP-Adressen direkt erreichen. Adressen ausserhalb des eigenen Netzes müssen über einen “Gateway” angesprochen werden.

IP-Adressen und Routing (3)

- Welche Adressen gehören zu einem Netz?
- Früher gab es feste Einteilungen in Netzgrößen (Klassen), dies wurde bereits 1993 abgeschafft, hält sich aber hartnäckig in schlechter Literatur und unter schlechten “Fachleuten”.
- Aktuell: CIDR (Classless Inter-Domain-Routing), flexible Einteilung, z.B. 4, 8, 16, 32 … IP-Adressen (Zweier-Potenzen).

IP-Adressen und Routing (4)

- Blöcke können weiter unterteilt werden. Netzgröße ergibt sich aus Subnetz-Maske, diese bestimmt welche Teile einer IP zum lokalen Netz gehören.
- Beispiel: IP-Adresse 10.10.10.1. Subnetz-Maske 255.255.255.0 heißt: die ersten drei Zahlen sind das Netz, die letzte Zahl ist lokal.
- Netz also 10.10.10.0 bis 10.10.10.255 (oder in CIDR-Notation: 10.10.10.0/24).

IP-Adressen und Routing (5)

- Nicht lokale Pakete gehen an den Gateway, der diese weiterleitet (z.B. anderer Gateway oder ISP). Beispiel:

```
1  gw-distp-a.bs.ka.schlund.net (195.20.247.130)
2  ge-2001.bba.bs.ka.schlund.net (212.227.121.193)
3  so-5000.gw-backbone-a.ffm.schlund.net (212.227.120.7)
4  rmws-frnk-de07.nw.telefonica.de (80.81.192.89)
5  rmwc-frnk-de01-so-0-1-2-0.nw.mediaways.net
   (213.20.249.201)
6  rmwc-gtso-de01-pos-1-0.nw.mediaways.net
   (195.71.254.121)
7  xmws-gtso-de01-vlan-3.nw.mediaways.net
   (217.188.58.203)
8  195.71.11.67 (195.71.11.67)
```

Was ist IP überhaupt?

- Der Begriff “IP” wird häufig fälschlich als Synonym für die “Internet-Protokoll-Familie” verwendet.
- IP transportiert nur Daten-Pakete von A nach B (von IP-Adresse zu IP-Adresse). Die Pakete können beliebigen Inhalt haben, IP kennt weder Protokolle, noch Dienste (Ports) oder Fehlerkorrektur. IP entspricht in etwa der Netzwerk-Schicht / OSI-Layer 3.
- Niemand nutzt “nur” IP!

Und wer nutzt dann IP?

- Oberhalb von IP sind die eigentlichen Internet-Protokolle angeordnet (Transport- und Sessionschicht / OSI-Layer 4 und 5)
- Es sind alle Protokolle möglich, die in irgendeiner Weise Daten in Paketen transportieren
- Protokolle haben eine eindeutige Nummer die von der IANA vergeben wird, derzeit ca. 140 Protokolle (die meisten irrelevant)

Wichtige IP-Protokolle (1)

- ICMP – Internet Control Message Protocol;
  Transport von Steuernachrichten, z.B. “Ziel nicht erreichbar”. Üble Unsitte: Blockierung von ICMP an Firewalls (niemals!)

- UDP – User Datagram Protocol;
  Übertragung von Datenpaketen ohne Fehlersicherung oder Wiederholung, z.B. für Echtzeitdaten (Telefonie, Streaming)

Wichtige IP-Protokolle (2)

- TCP – Transmission Control Protocol; Verbindungsorientierte Datenübertragung mit Fehlersicherung; sozusagen das Brot- und Butterprotokoll des Internets

- SCTP – Stream Control Transmission Protocol; relativ neuer Standard mit garantierter Qualität und mehreren Kanälen; wird derzeit massiv durch Telefonanbieter gepusht, Nutzung z.B. für Übertragung zwischen Ortsvermittlungsstellen

Von der Theorie zur Praxis

- Die ausgeteilte CD startet direkt ein Linux-System, in welchem die weiteren Versuche stattfinden.
- System booten, an der grafischen Benutzeranmeldung mit dem Benutzernamen “root” und dem Passwort “root” anmelden.
- Eine Eingabeaufforderung und Ethereal starten. In Etherreal unter Capture -> Interface “lo” wählen.

Versuch 1: Ping (=ICMP) (1)

- Ping ist eine einfache Methode um die Erreichbarkeit eines anderen Systems zu prüfen (sofern nicht ausgefiltert).
- Ping nutzt dazu den ICMP-Unterdienst “Echo request”, welcher in einem “Echo reply” die erhaltenen Daten 1:1 zurückschickt.
- Je nach Ping-Programm werden verschieden große Pakete mit Zufalls- oder anderen Daten verschickt.

Versuch 1: Ping (=ICMP) (2)

- Als Versuch ein einzelnes Pingpaket an den eigenen Rechner schicken:

```
ping -c 1 localhost
```

- In Ethereal sind sowohl “Echo Request” wie “Echo Reply” zu sehen und werden zur Analyse in ihre Bestandteile zerlegt angezeigt. Dadurch sehr lesbar.

- Ethereal kennt übrigens sehr viele Protokolle.

Versuch 1: Ping (=ICMP) (3)

- Ergebnis: Das gesniffte Paket besteht aus:
  - Ethernet-Header. Dieser gehört nicht zum IP-Paket und wird nicht über das Internet geschickt.
  - IPv4 Header. Legt u.a. Quelle, Ziel, Optionen und IP-Pakettyp fest.
  - ICMP-Header. Legt Typ und Sequenz des ICMP-Paketes fest.
  - Pingdaten. Die ersten 8 Byte = Timestamp zur Laufzeitbestimmung

Versuch 1: Ping (=ICMP) (4)

- Durch andere Parameter kann der Inhalt der Pingpakete modifiziert werden.
- Mögliche Parameter: '-s 80' setzt die Länge der Daten auf 80 Byte. '-p 1020' verwendet die Bytes 0x10 0x20 zur Erzeugung der Daten.
- Quiz-Frage: Welche Pingdaten sendet der folgende Ping-Befehl?

```
ping -c 1 -s 13 -p 111213
```

Versuch 1: Ping (=ICMP) (5)

- Auflösung:

  Das gesendete Paket ist

  XX XX XX XX XX XX XX XX 11 12 13 11 12

  wobei xx = Timestamp (8 Byte), dadurch bleiben noch 5 Byte für das Pattern, welches wiederholt wird.

Das UDP-Protokoll (1)

- UDP sendet Datenpakete von A nach B nach dem “Fire and forget” Prinzip.
- Es wird keine Verbindung aufgebaut, die Zustellung der Pakete ist nicht gewährleistet.
- UDP-Pakete werden von Routern bei Überlast oder aus anderen Gründen ohne Rückmeldung verworfen.
- Nutzung z.B. für Streaming, da hier eine Neuübertragung verlorener Pakete sinnlos ist.

Das UDP-Protokoll (2)

- Da gleichzeitige UDP-Nutzung verschiedener Nutzer zu verschiedenen Zielen möglich ist, muß jede Datenstrom eindeutig identifizierbar sein. Untertyp wie bei ICMP reicht nicht aus.
- Lösung: Quell- und Zielports werden eingeführt. Ein Serverdienst hat einen standardisierten Port. Wenn eine IP-Adresse die Hausanschrift ist, ist der Port ein bestimmter Briefkasten. Der Absender wählt einen freien Port bei sich.

Versuch 2: DNS (=UDP) (1)

- Der DNS-Dienst löst Hostnamen auf, d.h. er ermittelt die zugehörige IP-Adresse. Aus “www.web.de” wird so beispielsweise 217.72.195.42.
- Anmerkung: Ein Hostname kann mehrere IP-Adressen haben. Dadurch Lastverteilung möglich, Ansatz hat aber Probleme.
- DNS verwendet UDP Port 53 für Anfragen, große Antworten und Zonentransfers ggf. per TCP (auch Port 53)

Versuch 2: DNS (=UDP) (2)

- Mittels des Befehls 'dig' kann ein Nameserver befragt werden. Als Versuch kann die IP-Adresse für 'localhost' ermittelt werden. Pakete wieder mit Ethereal mitsniffen.

dig localhost @ 127.0.0.1

- '@127.0.0.1' bedeutet dabei, daß der DNS-Server mit der IP 127.0.0.1 gefragt werden soll. Möglich wäre auch '@localhost'.
- In Ethereal kann die DNS-Abfrage nachvollzogen werden.

Versuch 2: DNS (=UDP) (3)

- Ergebnis: Das gesniffte Paket besteht aus:
  - Ethernet-Header. (wie zuvor)
  - IPv4 Header. (wie zuvor)
  - UDP-Header. Enthält Ports, Länge, Prüfsumme (teilweise Redundanz!)
  - Nutzlast. (hier DNS)

Versuch 3: UDP (1)

- Neuer Versuch: Um zwei Feinheiten von UDP auszuprobieren, werden mit 'netcat' Daten an einen unbelegten UDP-Port geschickt. Sniffing in Ethereal starten und dann netcat aufrufen:

  nc -u localhost 1234

- Der Befehl kehrt zunächst nicht zurück. Sind jetzt schon Daten geflossen?
- Nun Return drücken, Befehl kehrt zurück. Sniffing beenden.

Versuch 3: UDP (2)

- Ergebnis 1: Direkt nach dem Aufruf des Befehls sind noch keine Daten geflossen. Erst sobald Nutzdaten zu übertragen sind, wird ein Paket gesendet.
- Ergebnis 2: UDP verbindungslos, dennoch ICMP-Nachricht über Nichterreichbarkeit. Macht Sinn, wird aber nur von manchen Betriebssystemen unterstützt.

Fazit UDP

- UDP ist insbesondere für Streaminganwendungen interessant.
- UDP-Pakete werden aus einer Vielzahl von Gründen, in der Regel ohne Benachrichtigung, verworfen.
- Nutzer muß ggf. selbst für Wiederholung sorgen (oder gleich TCP verwenden).

Das TCP-Protokoll (1)

- TCP wird für alle Anwendungen verwendet, bei denen Daten auf jeden Fall und korrekt ankommen sollen.
- TCP ist verbindungsorientiert und wiederholt defekte oder verlorene Datenpakete selbstständig und völlig transparent für den Benutzer.
- TCP verwendet wie UDP Ports.

Das TCP-Protokoll (2)

- Der Verbindungsaufbau bei TCP geschieht im “Three-Way-Handshake”: Der Client schickt ein “SYN”, der Server ein “SYN+ACK” und der Client ein “ACK”. Dies verhindert, daß ein bösartiger Client den Server “verstopfen” kann.
- TCP ist relativ kompliziert mit Flags, Bestätigungen und Zählern.

Versuch 4: HTTP (=TCP) (1)

- Webseitenzugriffe geschehen über TCP auf Port 80. Webbrowser Mozilla Firefox (Erdkugelsymbol) starten.
- Ethereal starten und folgende URL aufrufen:

http://localhost/

- Ethereal stoppen und Pakete analysieren. Dabei auch interessant: 'Follow TCP Stream'.

Versuch 4: HTTP (=TCP) (2)

- Ergebnis: Der Three-Way-Handshake kann nachvollzogen werden.
- Die Bestätigung empfangener Pakete geschieht über die Sequenznummern und Acknowledgenummern. Acknowledge 456 bedeutet: Bis Sequenz 455 einschließlich wurden alle Daten empfangen.
- Der Verbindungsabbau geschieht ebenfalls in drei Schritten.

Workshop Teil 2: TCP/IP für Fortgeschrittene

SSL/TLS

- Problem: TCP/IP ist unverschlüsselt und kann daher mit geeigneten Mitteln mitgelesen werden.
- Netscape Communications wollte den aufkommenden Webmarkt bzw. eCommerce stärken und entwickelte das Secure Socket Layer, welches eine Verschlüsselung auf Basis von TCP/IP einführt.
- Bekanntes Resultat: URLs https://...

Versuch 5: HTTPS (=SSL) (1)

- Zur Demonstration soll nun eine verschlüsselte Webseite abgerufen werden. Ethereal und Firefox verwenden und folgende Adresse aufrufen:

  `https://localhost/`

- Ethereal stoppen, Pakete analysieren.

Versuch 5: HTTPS (=SSL) (2)

- Ergebnis: HTTPS verwendet einen anderen Port als HTTP. Die Verbindung ist von Beginn an verschlüsselt.
- Dies hat große Nachteile: Beim SSL-Schlüsselaustausch ist noch gar nicht bekannt, welche Website gewünscht ist.
- Um verschiedene SSL-Zertifikate für verschiedene Domainnamen verwenden zu können, muß für jedes Zertifikat eine eigene IP verwendet werden.

TLS als Weiterentwicklung

- SSL wurde zu TLS weiterentwickelt. Es kann weiterhin wie bei HTTPS verwendet werden, d.h. auf einem gesonderten Port und verschlüsselt von vorneherein.
- Zusätzlich bieten moderne Protokolle die Umschaltung auf eine verschlüsselte Verbindung nach einer vorherigen unverschlüsselten Abstimmung an. Dies ermöglich virtuelle Hosts auf einer IP ohne Tricks.

Versuch 6: SMTP (inkl. TLS) (1)

- Mit dem Befehl 'swaks' können SMTP-Server getestet werden. Ethereal starten und folgenden Befehl eingeben:

```
swaks -f test@localhost -t
  test@localhost -s localhost
```

- Hierbei ist -f der Absender (from), -t der Empfänger (target) und -s der Server.
- Der Befehl verschickt eine Mail, das SMTP-Protokoll kann in Ethereal gezeigt werden.

Versuch 6: SMTP (inkl. TLS) (2)

- Nun zum Vergleich eine Mail mit verschlüsselter Übertragung:

```
swaks -f test@localhost -t
test@localhost -s localhost -tls
```

- In Ethereal kann nachvollzogen werden, daß der Mailserver Verschlüsselung anbietet und vom Client angefordert wird. Erst anschließend wird die Verbindung verschlüsselt.

Versuch 6: SMTP (inkl. TLS) (3)

- Es gibt übrigens auch eine Variante von SMTP die von Beginn an verschlüsselt ist. Diese verwendet wie HTTPS einen gesondern Port. Wegen STARTTLS hat diese Variante aber praktisch keine Verbreitung, da sie technisch deutlich schlechter ist.

Fazit SSL/TLS

- Die Verschlüsselung von SSL/TLS geschieht für den Benutzer transparent.
- Moderne Protokolle verwenden TLS und wickeln dies über den selben Port wie unverschlüsselten Verkehr ab. Hierbei wird die Verschlüsselung zunächst unverschlüsselt vereinbart und dann umgeschaltet. Dies löst eine Menge weiterer Probleme und erlaubt die parallele Nutzung diverser SSL-Zertifikate.

IPSec als Alternative? (1)

- Der Ansatz IPSec versucht Verschlüsselung auf einer anderen Ebene zu greifen. Diese wird hierbei nicht auf TCP aufgesetzt, sondern zwischen TCP und IP eingefügt. Dadurch können z.B. auch UDP-Verbindungen verschlüsselt werden.
- IPSec ist leider relativ kompliziert, zu bestehenden Anwendungen nicht kompatibel und wird in der Praxis primär für VPN-Strecken verwendet.

IPSec als Alternative? (2)

- IPSec wurde für IPv6 entwickelt und kennt zwei verschiedene Konzepte, die kombiniert werden können:
- Integrität: Das Datenpaket und seine Steuerinformationen wird über eine kryptographische Prüfsumme vor Manipulationen geschützt.
- Vertraulichkeit: Das Datenpaket wird verschlüsselt.

IPSec als Alternative? (3)

- Integrität wird über den Authentication Header (AH) ermöglicht. Dieser wird dem zu schützenden Datenpaket vorangestellt.
- Vertraulichkeit geschieht über die Encapsulated Security Payload (ESP). Hierbei wird das ursprünglichen Datenpaket verschlüsselt und in ein neues Datenpaket verpackt. Dies ermöglicht zusätzlich Tunneling und wird bei VPN verwendet.

IPSec als Alternative? (4)

- Optional kann dem bei ESP neu erzeugtem Paket ein AH-Header vorangestellt werden um Vertraulichkeit und Integrität zu kombinieren.
- Kryptographieschlüssel bei IPSec werden entweder statisch festgelegt und zuvor ausgetauscht, oder über das Protokoll IKE (Internet Key Exchange) über ein gemeinsames Geheimnis (pre-shared Secret) generiert.

Fazit IPSec

- IPSec im ESP-Modus eignet sich für VPN-Strecken und bietet dann transparente, sichere Verschlüsselung die beliebige IP-Protokolle unterstützt.
- IPSec ist nicht geeignet für Verbindungen zu vielen verschiedenen (vorher unbekannten) Gegenstellen wie z.B. Web- oder Mailservern. Hier ist SSL/TLS deutlich sinnvoller und einfacher zu handhaben.

IPv6: Standard der Zukunft? (1)

- IPv4 kann maximal 2^{32} IP-Adressen verwenden. Es wird seit langem behauptet, daß die IP-Adressen bald ausgehen. Diese Behauptung ist ein populäres Märchen, denn aktuell sind noch über 40% der IP-Adressen frei.
- IPv6 soll die IP-Adressknappheit ein- für allemal erledigen und IP-Adressen zur Not auch für jeden Joghurt im Kühlschrank bereitstellen.

IPv6: Standard der Zukunft? (2)

- IPv6 erlaubt 2^{128} IP-Adressen, das sind mehrere tausend IP-Adressen pro m^2 Erdoberfläche.
- Darüberhinaus bietet IPv6 viele weitere Verbesserungen wie modulare Header, IPSec (was aber ja auch auf IPv4 portiert wurde).
- Dennoch: Die Adressknappheit ist nicht gegeben, obwohl die Internet-Registries zum Teil (angeblich absichtlich) leichtfertig die IP-Adressen verschleudern

IPv6: Standard der Zukunft? (3)

- Da die Umstellung sehr teuer wird, der Nutzen nur schwer zu erkennen ist, und es ja alles irgendwie läuft (wenn auch mit Krücken), ist die Zukunft von IPv6 alles andere als gesichert. Es dümpelt seit Jahren vor sich hin und kein Ende in Sicht.
- Prinzipiell wäre der Umstieg wünschenswert.

Routing im Internet (1)

- ISPs lassen sich von einer Local Internet Registry einen Block IP-Adressen zuteilen. Häufig sind ISPs selber LIRs.
- Dieser Netzblock muß nun im Internet bekanntgegeben werden. Dazu existieren die sogenannten Autonomen Systeme (AS).
- Autonom = Diese System entscheiden das Routing über mehrere mögliche Wege selber. Dies geschieht über Algorithmen.

Routing im Internet (2)

- Autonome Systeme (i.d.R. == ISP) sind über Festverbindungen oder auf anderen Wegen miteinander verbunden. Gleichzeitig betreibt jedes Autonome System einen Routingserver mit dem BGP-Protokoll (Border Gateway Protocol).
- BGP überträgt Routeninformationen. Ein ISP wird nun über BGP bekanntgeben, daß Daten für seine Netzblöcke zu ihm geroutet werden sollen.

Routing im Internet (3)

- Diese Information gibt er weiter an seine Nachbarn (andere ISPs, Upstream), und diese wiederrum ebenfalls. So verbreitet sich die Route unter allen autonomen Systemen.

- Autonome Systeme haben immer eine Verbindung (jedenfalls per Policy) zu mindestens zwei unterschiedlichen anderen autonomen Systemen.

Routing im Internet (4)

- Dadurch würden sich sehr viele mögliche Wege ergeben. In der Praxis sind die verwendbaren Wege aber durch Vertrags- und Finanzfragen eingeschränkt. So kann ein Kunde eines anderen ISPs nicht ohne weiteres den Datenverkehr weiter an andere ISPs verkaufen.

- Resultat: Spätestens seit die BWLer am Werk waren, ist das atombombensichere Internet endgültig unmöglich geworden.

Routing im Internet (5)

- BGP ist in die Jahre gekommen und das Protokoll hat Schwächen.
- Die Routingtabelle werden immer umfangreicher und inkompetente ISPs verstopfen diese unnötig.
- Die Nummern für autonome Systeme sind schon weit über 50% aufgebraucht.

Ende

- Das war der Workshop. Ich hoffe er hat allen Teilnehmern gefallen und es wurde einiges über TCP/IP gelernt.
- Vielen Dank für die Aufmerksamkeit!