# Linux 2.4 - Netfilter/iptables

Alexander Schreiber
TU Chemnitz, Fakultät für Informatik
als@thangorodrim.de

9. Juni 2000

**Zusammenfassung**

Der Kernel-Firewall wurde auch bei Linux 2.4 sehr stark überarbeitet. Der ipchains-Code von 2.2 wird durch iptables abgelöst, das Gesamtsystem heißt Netfilter. Was hat sich gegenüber den Vorgängern ipfwadm/ipchains geändert und wieviel mehr kann Netfilter leisten?

# Was ist Netfilter?

- Linux-Kernelfirewall,
- Nachfolger von ipchains,
- endgültiger Name: iptables,
- neue Architektur (basierend auf ipchains),
- Kompatibilität mit ipfwadm und ipchains,
- neue Features,
- beliebig erweiterbar

# Was ist anders gegenüber ipchains?

- eingebaute Chains jetzt INPUT, OUTPUT, FORWARD,
- INPUT, OUTPUT sehen nur noch Pakete mit lokalem Host als Quelle oder Ziel (ipchains: alle Pakete),
- Nullen der eingebauten Chains löscht auch Policy Counter,
- Atomisches Nullen funktioniert (view and zero),
- Listen der Chains liefert Zähler als atomischen Snapshot,
- längere Chainnamen möglich (31 jetzt vs. 8 Zeichen vorher),

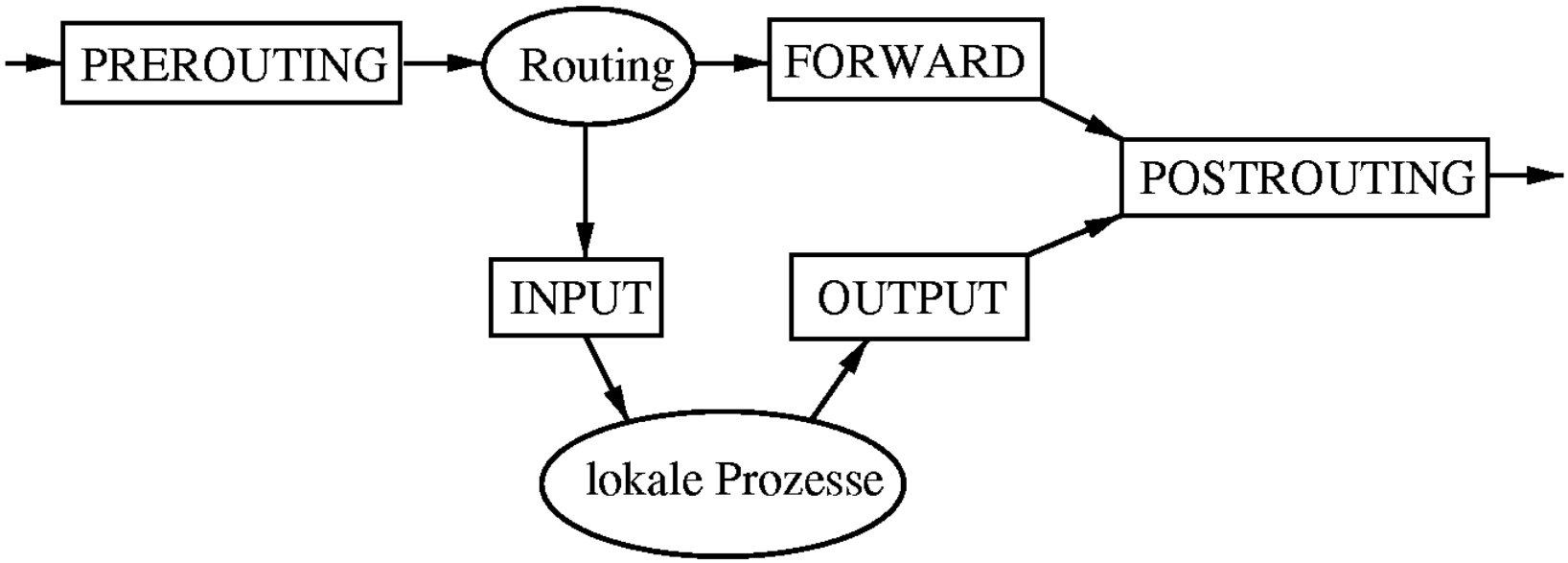
# Was ist anders gegenüber ipchains? (Fortsetzung)

- Protokollhandler (ICMP, TCP, UDP) als Erweiterungen,
- geänderte Syntax des Userspace-Tools (iptables),
- MASQUERADE (vorher: MASQ) und REDIRECT überarbeitet,
- DENY heißt jetzt endgültig DROP,
- geänderte Paketverarbeitung,
- umfangreicherere, erweiterbare Logmeldungen,
- zahlreiche weitere Änderungen

# Paketauswertung bei iptables

- anderes Verfahren als bei ipchains,
- incoming packets:
  - routing decision,
  - für lokales System:
    weiter in INPUT chain, dann zum lokalen Prozess,
  - IP-Forwarding aktiviert und Paket für anderes Interface bestimmt:
    zur FORWARD-Chain und dann Outgoing,
  - anderenfalls:
    Paket wird verworfen,
- outgoing packets:
- Paket direkt zur OUTPUT-Chain, wenn akzeptiert:
  direktes Weiterleiten zum Zielinterface

# Packetauswertung bei iptables - Grafik




Paketauswertung und -modifikation bei iptables

# iptables - ein tabellenbasiertes System

- iptables basiert auf Tabellen von Chains:
  - Tabelle filter: Paketfilterung, default,
  - Tabelle nat: NAT, Masquerading, Redirecting,
  - Tabelle mangle: allgemeines „Bearbeiten“ von Paketen,
- Tabellen und Ketten verknüpfbar,
- typische Entscheidungspunkte über Paketschicksal:
  - INPUT-, OUTPUT-, FORWARD-Chain,
  - nutzerdefinierte Chains,
  - PREROUTING, POSTROUTING

# iptables - ein erweiterbares System

- modularer Aufbau - einfach erweiterbar,
- Standardschnittstelle für eigene Module,
- Erweiterungen als 2 Module:
  - Kernelmodul zur Implementation der Funktionalität,
  - Modul (ELF shared object) für iptables-Tool (Userspace) zur Steuerung der Funktionalität
- verschiedene Standardfunktionen als Modul implementiert,
- standardmäßig mit verschiedenen Modulen ausgestattet

# Standardmodule von iptables

| Modul | Funktion |
|---|---|
| ip_conntrack | Verbindungsverfolgung (connection tracking) |
| ip_contrack_ftp | dito für FTP |
| ip_nat_ftp | NAT-Support für FTP |
| ip_queue | packet queueing (Weiterreichen an Userspace) |
| ipchains | ipchains Kompatibilität |
| ipfwadm | ipfwadm Kompatibilität |
| ipt_LOG | packet logging (Target LOG) |
| ipt_MARK | packet marking |
| ipt_MASQUERADE | Masquerading |
| ipt_MIRROR | packet mirroring (source $\leftrightarrow$ destination) |
| ipt_REDIRECT | transparentes Umleiten von Paketen |
| ipt_REJECT | Zurückweisen von Paketen |
| ipt_TOS | type of service setzen |
| ipt_limit | Begrenzerfilter |

# Standardmodule von iptables (Fortsetzung)

| Modul | Funktion |
|---|---|
| ipt_mac | MAC-Filter |
| ipt_mark | Filter auf MARK-Symbole von Paketen |
| ipt_multiport | Filter für mehrere Ports auf einmal |
| ipt_owner | Filter auf erzeugenden Nutzer (lokal) |
| ipt_state | Filter für Verbindungsstatus |
| ipt_tos | Filter für type of service |
| ipt_unclean | Filter für „komische“ Pakete |
| iptable_filter | implementiert Tabelle filter |
| iptable_mangle | implementiert Tabelle mangle |
| iptable_nat | implementiert Tabelle nat |

# Kompatibilitätsmodule

- Kompatibilität zu ipfwadm oder ipchains,
- Aktivieren durch Laden der entsprechenden Kernelmodule,
- vorhandene, gewohnte Tools und Scripte nutzbar,
- Vereinfachung des Übergangs,
- Interfaces exklusiv: entweder ipfwadm *oder* ipchains,
- im Kompatibilitätsmodus ist das eigentliche Netfilter-Interface (iptables) *nicht* nutzbar

# Packet Logging

Die Logmöglichkeiten wurden massiv erweitert:

- Logging ist nicht paketentscheidend, d.h. LOG-Target wird transparent durchlaufen, Paket bleibt erhalten,
- Festlegen des Loglevels,
- Festlegen eines Prefixes für Logmeldungen,
- Loggen spezieller Paketeigenschaften:
  - TCP-Sequenznummern (ggf. Sicherheitsloch),
  - TCP-Optionen,
  - IP-Optionen

# Limiting

- limit matcht nur für vorgegebenes Rate (x-mal/(s|min|h),
- verwendet token bucket filter,
- konstantes Limit und Burstlimit, aktueller Burst erhöht sich um eins bis Burstlimit für jedes Mal Nichterreichen des konstanten Limits,
- beliebig mit anderen Möglichkeiten kombinierbar (nicht nur LOG),
- Beispiele:
  - ICMP echo request/reply flood protection,
  - syn flood protection,
  - log flood protection

# NAT mit iptables

- D-NAT (Destination NAT) erfolgt in:
  - PREROUTING (hereinkommende Pakete),
  - OUTPUT (lokal erzeugte Pakete),
- S-NAT (Source NAT): in POSTROUTING,
- MASQUERADING: Teilmenge von S-NAT,
- REDIRECT: Teilmenge von D-NAT,
- minimales Loadbalancing (bei Gruppe von IPs minimal genutzte),
- minimales Umschreiben der Pakete (kein unnötiges Portremapping)

# Packet State Matching

- Matching auf Zustand der Verbindung,
- mögliche Zustände:
  - NEW: Paket erzeugt neue Verbindung,
  - ESTABLISHED: Paket ist Teil einer existierenden Verbindung,
  - RELATED: Paket hat mit existierender Verbindung zu tun, ist aber nicht Teil davon (ICMP error, Aufbau einer Datenverbindung bei FTP),
  - INVALID: Paket kann nicht zugeordnet werden (einschließlich Probleme wegen Speicherüberlauf und unbekannte ICMP-Fehler), Empfehlung: DROP

# Packet Owner Matching

- Matching auf Erzeuger des Pakets,
- nur für lokal erzeugte Pakete,
- mögliche Zuordnungen:
  - uid-owner: effektive User-ID,
  - gid-owner: effektiver Group-ID,
  - pid-owner: Prozess-ID,
  - sid-owner: Session-Group

# Fazit

- sehr leistungsfähig,
- einfach und standardisiert erweiterbar,
- bietet „out-of-the-box“ interessante Möglichkeiten,
- bietet Kompatibilität zu alten Interfaces
- vermeidet verschiedene Probleme der Vorgänger