Netfilter/iptables

hackerspace.gr

@apoikos

13 Ιουνίου 2013

Εισαγωγή

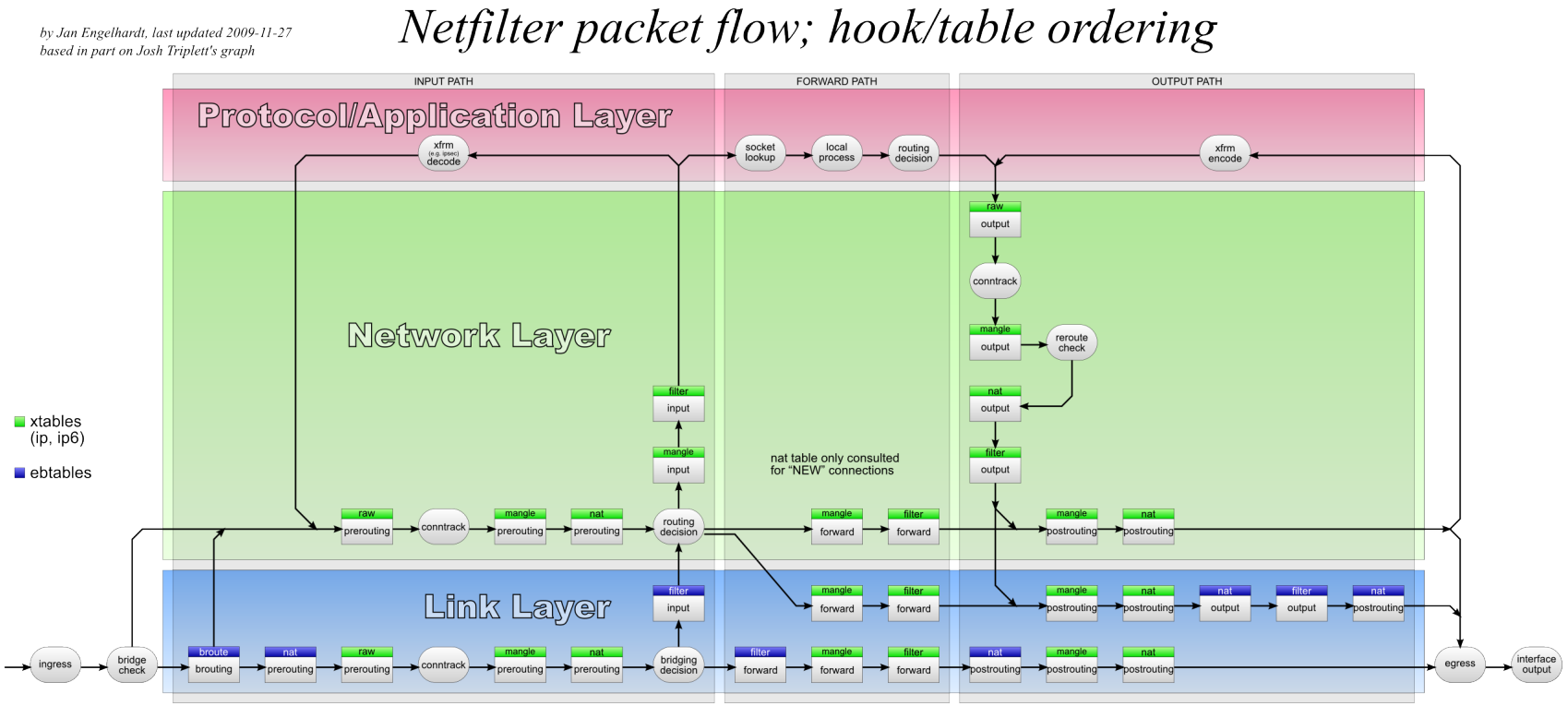
Έννοιες-κλειδιά:

- **Firewall**: συσκευή ή software που «φιλτράρει» τα πακέτα που διέρχονται από αυτό βάσει κριτηρίων.
- **NAT**: Network Address Translation. Αντικατάσταση της διεύθυνσης του παραλήπτη ή/και του αποστολέα ενός πακέτου με μια άλλη, προκειμένου αυτό να δρομολογηθεί σωστά.
- **Packet mangling**: Τροποποίηση των πακέτων, τόσο σε επίπεδο χαρακτηριστικών όσο και δεδομένων.
- **Policy routing**: Δρομολόγηση βάσει πολλών «πολιτικών», και όχι απλά βάσει της διεύθυνσης του παραλήπτη.

Netfilter: η υποδομή firewall/NAT του Linux

- Packet filtering, connection tracking & NAT για τον Linux kernel
- Ασχολείται με τα περιεχόμενα των πακέτων
- Επιτελεί 4 βασικές λειτουργίες: filtering, connection tracking, NAT και mangling
- User-space εργαλεία: iptables, ip6tables, nfacct, conntrack-tools, ...
- Υποστήριξη modules που επεκτείνουν τις δυνατότητες επιλογής και χειρισμού των πακέτων
- Connection tracking $\longrightarrow$ stateful firewalling

Η δομή του netfilter



Conntrack

- Connection tracking
- Ομαδοποιεί τα *πακέτα* σε *συνδέσεις*
- Stateful πρωτόκολλα (π.χ. TCP): ρητή αντιστοιχία καταστάσεων
- Stateless πρωτόκολλα (π.χ. UDP): εξαγωγή πληροφορίας από δευτερεύοντα χαρακτηριστικά
- Layer-7 inspection μέσω helper modules για τον προσδιορισμό σχετιζόμενων συνδέσεων (π.χ. FTP, TFTP, SIP, ...): `nf_conntrack_*.ko`
- Καταγραφή στατιστικών ανά σύνδεση (traffic accounting)
- Απλοποιεί πολύ τη συγγραφή firewall rules...
- ...αλλά τρώει πόρους

Conntrack: παραμετροποίηση - διαχείριση

Πυρήνας

- `/proc/net/nf_conntrack`
  - Π
- `/proc/sys/net/netfilter/nf_conntrack_*`
  - `Max connection count`
  - `Timeouts`
  - `Accounting`
  - `Buckets`

Userspace

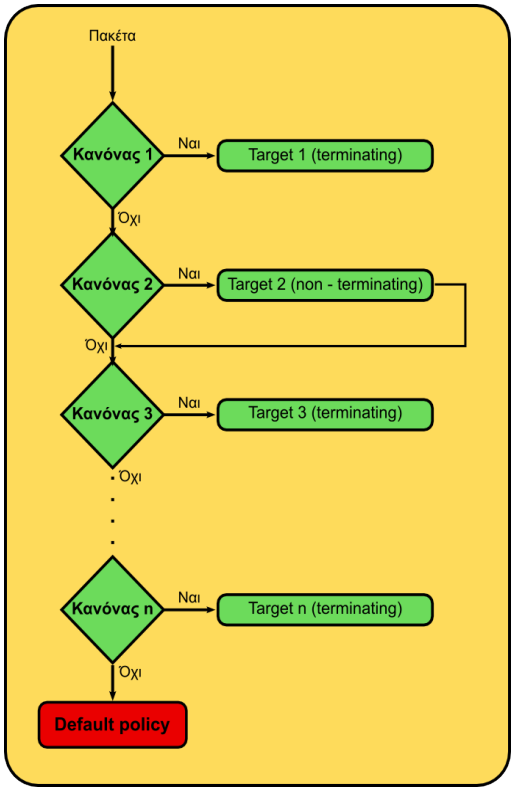
- `conntrack`: Χειρισμός των conntrack entries
- `conntrackd`: Παρακολούθηση της κατάστασης των συνδέσεων (logging/accounting), με δυνατότητα συγχρονισμού ανάμεσα σε μηχανήματα για HA firewalls.

iptables/ip6tables

- Κύριες έννοιες: πίνακας (table), αλυσίδα (chain), κανόνας (rule), προορισμός (target), σύνολο κανόνων (ruleset)
- 5 πίνακες: filter, nat, mangle, raw και security
- Κάθε πίνακας έχει κάποιες standard αλυσίδες, ενώ ο διαχειριστής μπορεί να ορίσει custom αλυσίδες
- Κάθε αλυσίδα αποτελείται από μια αλληλουχία κανόνων
- Κανόνες - αποτελούνται από 2 μέρη:
  1. Περιγραφή των πακέτων που μας ενδιαφέρουν (*match*)
  2. Περιγραφή του τι θέλουμε να κάνουμε με αυτά τα πακέτα (*target*)

Chain traversal

- Τα πακέτα εισέρχονται στην κορυφή μιας αλυσίδας
- Κανόνες: είναι ένα «if».
  - True → το πακέτο πάει στον προορισμό του κανόνα
  - False → το πακέτο περνάει στον επόμενο κανόνα
- Προορισμοί (targets):
  - Terminating ή non-terminating
  - Μπορούν να είναι κάποιες *ενέργειες* (actions), όπως ACCEPT, DROP, REJECT ή αλυσίδες που έχει καθορίσει ο χρήστης
- Policy στο τέλος των built-in αλυσίδων = default action. Οι user-defined επιστρέφουν μετά τον κανόνα που τις κάλεσε.



filter table

- Είναι το table στο οποίο γίνεται το firewalling
- 3 built-in αλυσίδες:
  - INPUT: εισερχόμενα πακέτα που *προορίζονται* για το μηχάνημά μας
  - OUTPUT: εξερχόμενα πακέτα που *δημιουργούνται* από το μηχάνημά μας
  - FORWARD: πακέτα που *διέρχονται* από το μηχάνημά μας (όταν αυτό κάνει routing/bridging)
- Συνήθεις προορισμοί:
  - ACCEPT: αποδοχή του πακέτου
  - DROP: σιωπηλή απόρριψη του πακέτου
  - REJECT: απόρριψη του πακέτου με ενημέρωση του αποστολέα (TCP RST, ICMP/ICMPv6...)
  - LOG/ULOG/NFLOG: καταγραφή του πακέτου στο syslog ή ένα netlink socket (non-terminating)

nat table

- Είναι το table στο οποίο γίνεται το NAT (Network Address Translation)
- 3 built-in αλυσίδες:
  - PREROUTING: εισερχόμενα πακέτα, ανεξαρτήτως του αν προορίζονται για το μηχάνημά μας ή όχι
  - POSTROUTING: εξερχόμενα πακέτα, ανεξαρτήτως του αν προέρχονται από το μηχάνημά μας ή όχι
  - OUTPUT: πακέτα που προέρχονται από το μηχάνημά μας
- Συνήθεις προορισμοί (targets):
  - DNAT (PREROUTING,OUTPUT): αλλαγή της διεύθυνσης προορισμού ενός πακέτου
  - SNAT (POSTROUTING): αλλαγή της διεύθυνσης αποστολέα ενός πακέτου
  - MASQUERADE (POSTROUTING): αλλαγή της διεύθυνσης αποστολέα ενός πακέτου με τη διεύθυνση του interface εξόδου

mangle table

- Το table στο οποίο «πειράζονται» τα πακέτα
- 5 built-in αλυσίδες: INPUT, OUTPUT, FORWARD, PREROUTING, POSTROUTING
- Συνήθεις προορισμοί (targets):
  - MARK: μαρκάρισμα ενός πακέτου με ένα συγκεκριμένο mark
  - CONNMARK: μαρκάρισμα ολόκληρης της σύνδεσης στην οποία ανήκει ένα πακέτο
  - TCPMSS, DSCP, TTL, TOS, ECN: αλλαγή διάφορων χαρακτηριστικών των πακέτων

iptables

```
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
```

Σύνταξη:

`iptables [-t table_spec] <action> <match> <target> [target_options]`

- `table_spec`: filter (default), nat, mangle.
- `<action>`:
  - -A ⟨chain⟩: Append
  - -I ⟨chain⟩ ⟨nr.⟩: Insert στη θέση nr.
  - -D ⟨chain⟩ ⟨nr.⟩: Delete στη θέση nr.
  - -F [⟨chain⟩]: Flush
- `<match>`:
  - `-p <protocol>`
  - `-s|-d <address>`: source ή destination address
  - `-m <match_name>`: φόρτωση extended match module *⟨match_name⟩*

Extended matches

Extended matches → επιτρέπουν την επιλογή πακέτων βάσει περισσότερων χαρακτηριστικών.

Φορτώνονται με `-m <match_name>`. Παραδείγματα:

- tcp, udp: Παρέχουν –sport, –dport –syn-flags (TCP)
- limit: Πόσα πακέτα/s;
- mark: Ταίριαγμα βάσει του mark του πακέτου
- connmark: Ταίριαγμα βάσει του connection mark της σύνδεσης
- state/ctstate: Stateful matching: NEW, ESTABLISHED, RELATED, INVALID
- time: Εφαρμογή του κανόνα συγκεκριμένες ώρες της ημέρας
- connbytes: Πόσα bytes έχουν περάσει για μια σύνδεση;
- physdev: Διάκριση πακέτων που διέρχονται από πόρτες μιας bridge
- owner: uid & gid του process που έκανε generate ένα πακέτο
- και πολλά άλλα ;-)

NFQUEUE

NFQUEUE: firewalling in userspace

- Υλοποιείται με το NFQUEUE target
- Πολλαπλά queues, δυνατότητα load-balancing ανάμεσα στα queues
- Επικοινωνία με το userspace μέσω netlink socket
- Το userspace αποφασίζει για την «τύχη» του πακέτου, επιστρέφοντας ACCEPT (+ modify?), DROP
- Bindings για C, perl, python...
- Εφαρμογές: application firewalls (nufw), traffic inspection...

Απλά παραδείγματα

Φιλτράρισμα κίνησης από συγκεκριμένη IP:

```
iptables -A INPUT -p tcp --dport 22 -s 1.2.3.4 -j DROP
iptables -A INPUT -p tcp --dport 80 -s 2.3.4.5 -j REJECT \
      --reject-with tcp-reset
iptables -I INPUT 2 -p udp --dport 53 -s 3.4.5.6 -j REJECT \
      --reject-with icmp-port-unreachable
```

Port forwarding:

```
iptables -t nat -A PREROUTING -p tcp --dport 2345 \
-d 147.102.51.160 -j DNAT --to-destination 192.168.0.1:22
```

NAT/Masquerading:

```
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
iptables -t nat -A POSTROUTING -o ppp1 -j SNAT \
      --to-source 1.2.3.4
```

Απλά παραδείγματα

Αλλαγή default policy:

```
iptables -P INPUT DROP
```

Μαρκάρισμα πακέτων

```
iptables -t mangle -A INPUT -i ppp0 -j CONNMARK --set-MARK 0x1
iptables -t mangle -A OUTPUT -j CONNMARK --restore-mark
```

Stateful matching

```
iptables -I INPUT 1 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Παράδειγμα: firewall σε router/server

Σενάριο

Ο router/server έχει συνδεδεμένο ένα εσωτερικό LAN στο eth0, το οποίο και θέλει να προστατέψει απ' έξω. Παράλληλα ο ίδιος τρέχει και κάποια services (Bad Idea™)

```
iptables -F
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m state --state INVALID -j DROP
iptables -A INPUT -p tcp ! --syn -m state --state NEW -j LOG
iptables -A INPUT -p tcp ! --syn -m state --state NEW -j DROP
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A INPUT -p icmp --icmp-type echo-request -m limit \
         --limit 5/s --limit-burst 10 -j ACCEPT
iptables -p icmp --icmp-type ! echo-request -j ACCEPT.
iptables -A INPUT -p tcp --dport bgp j ACCEPT
iptables -A INPUT -p tcp --dport ssh -j ACCEPT
iptables -A INPUT -p udp --dport domain -j ACCEPT
iptables -A INPUT -s 10.10.10.11 -j ACCEPT
iptables -A INPUT -j LOG
iptables -P INPUT DROP
```

Παράδειγμα: firewall σε router/server (2)

```
iptables -N net2lan

iptables -A FORWARD -o eth0 -j net2lan

iptables -A net2lan -m state --state INVALID -j DROP
iptables -A net2lan -p tcp ! --syn -m state --state NEW -j DROP
iptables -A net2lan -m state --state ESTABLISHED,RELATED \
         -j ACCEPT
iptables -A net2lan -p icmp --icmp-type echo-request -m hashlimit \
         --hashlimit-mode dstip --hashlimit 10/s -j ACCEPT
iptables -A net2lan -p icmp --icmp-type ! echo-request -j ACCEPT
iptables -A net2lan -d 10.10.10.11 -p tcp --dport 9111 -j ACCEPT
iptables -A net2lan DROP

modprobe nf_conntrack_ftp
```

ferm: For Easy RuleMaking

- High-level γλώσσα για συγγραφή iptables rulesets
- Dual-stack: δημιουργεί κανόνες για iptables και ip6tables από την ίδια πηγή
- Pre-/post-exec hooks, shell callouts, includes
- Πολύ βολικό για δημιουργία πολύπλοκων firewalls

ferm: παράδειγμα

```
@include 'defs.conf';

domain (ip ip6) table filter {
    chain accept_mgmt {
        saddr @ipfilter($MGMT) ACCEPT;
    }
    chain INPUT {
        policy DROP;
        mod state state (ESTABLISHED RELATED) ACCEPT;
        # Allow local packets
        interface lo ACCEPT;
        # Malformed tcp packets
        proto tcp mod state state NEW !syn DROP;
        # Allow icmp traffic
        proto icmp ACCEPT;
        # Allow SSH from the management networks
        proto tcp dport ssh jump accept_mgmt;
    }
    chain OUTPUT {
        policy ACCEPT;
    }
}
```