tcpdump Analyse

Sebastian Menski

Institut für Informatik
Universität Potsdam

19. Mai 2016

Fragestellung

Fragestellung

- Doktorarbeit von Simon Kiertscher
- HTTP Traffic Analyse mit tcpdump
- Verliert tcpdump Pakete unter hoher Last?

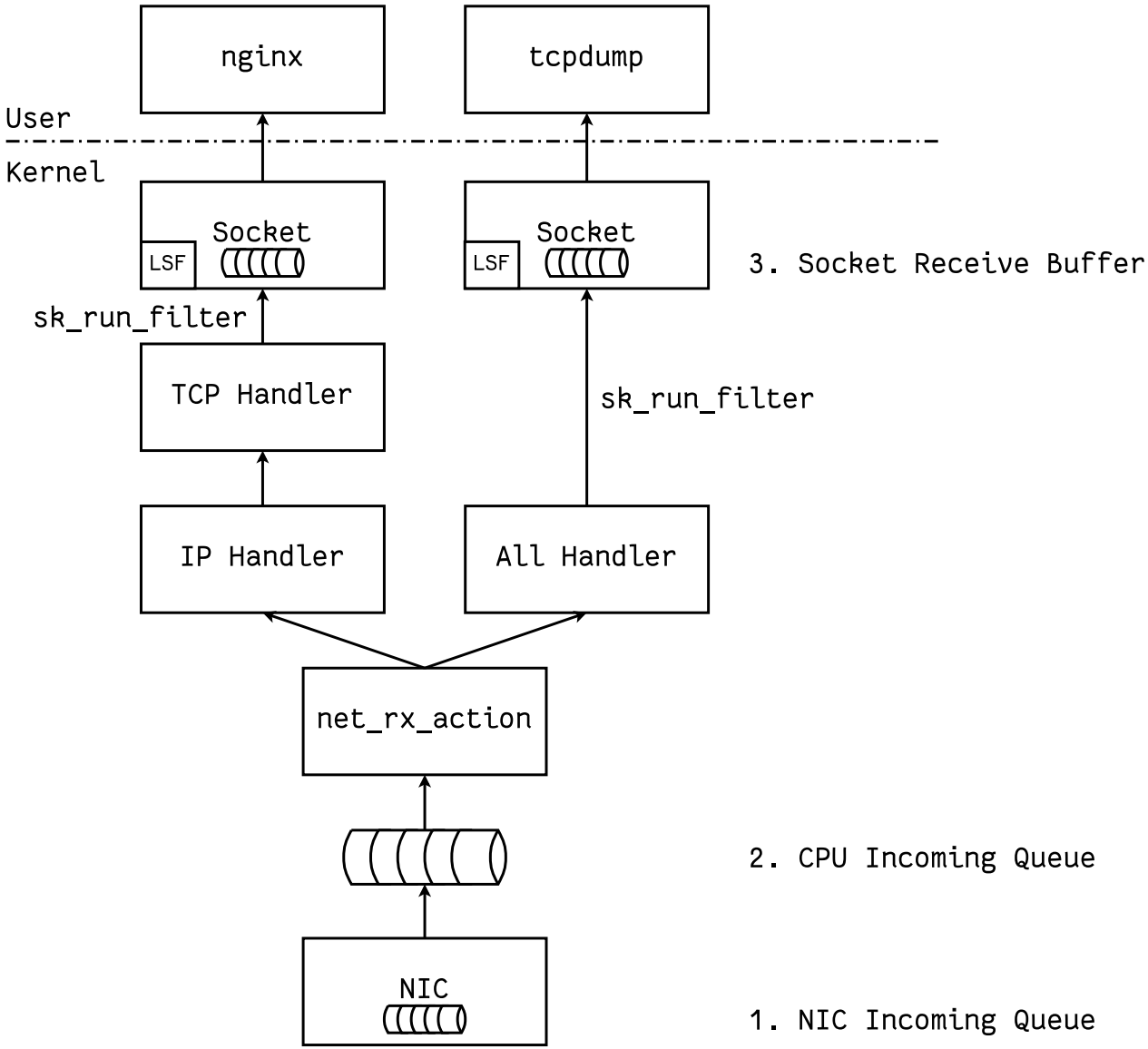
tcpdump

tcpdump

- tcpdump: CLI um Netzwerkpakete aufzuzeichen und zu analysieren
- libpcap: Bibliothek um Pakete auf unterschiedlichen Platformen zu filtern
- Untersuchte Versionen:
  - tcpdump 4.3.0
  - libpcap 1.3.0
  - CentOS 5 - Kernel 2.6.18

tcpdump

Netzwerkstack



tcpdump

Linux Socket Filter

- Linux Socket Filter (LSF) seit Kernel 2.2
- Filter kann an Socket angehangen werden
- Berkley Paket Filter (BPF) Programm Assembler ähnlich
- Direkter Zugriff auf Paketdaten

tcpdump

BPF für `tcpdump -d ip dst host 10.3.9.21 and tcp dst port 80`

```
(000) ldh      [12]
(001) jeq      #0x800           jt 2    jf 12
(002) ld       [30]
(003) jeq      #0xa030915       jt 4    jf 12
(004) ldb      [23]
(005) jeq      #0x6             jt 6    jf 12
(006) ldh      [20]
(007) jset     #0x1fff          jt 12   jf 8
(008) ldxb     4*([14]&0xf)
(009) ldh      [x + 16]
(010) jeq      #0x50            jt 11   jf 12
(011) ret      #65535
(012) ret      #0
```

tcpdump

Parameter

- snaplen (-s): Anzahl der Bytes die von einem Paket aufgezeichnet werden
- buffer (-B): Größe des Empfangsbuffers
- filter: Socketfilter um Pakete im Kernel zu filtern

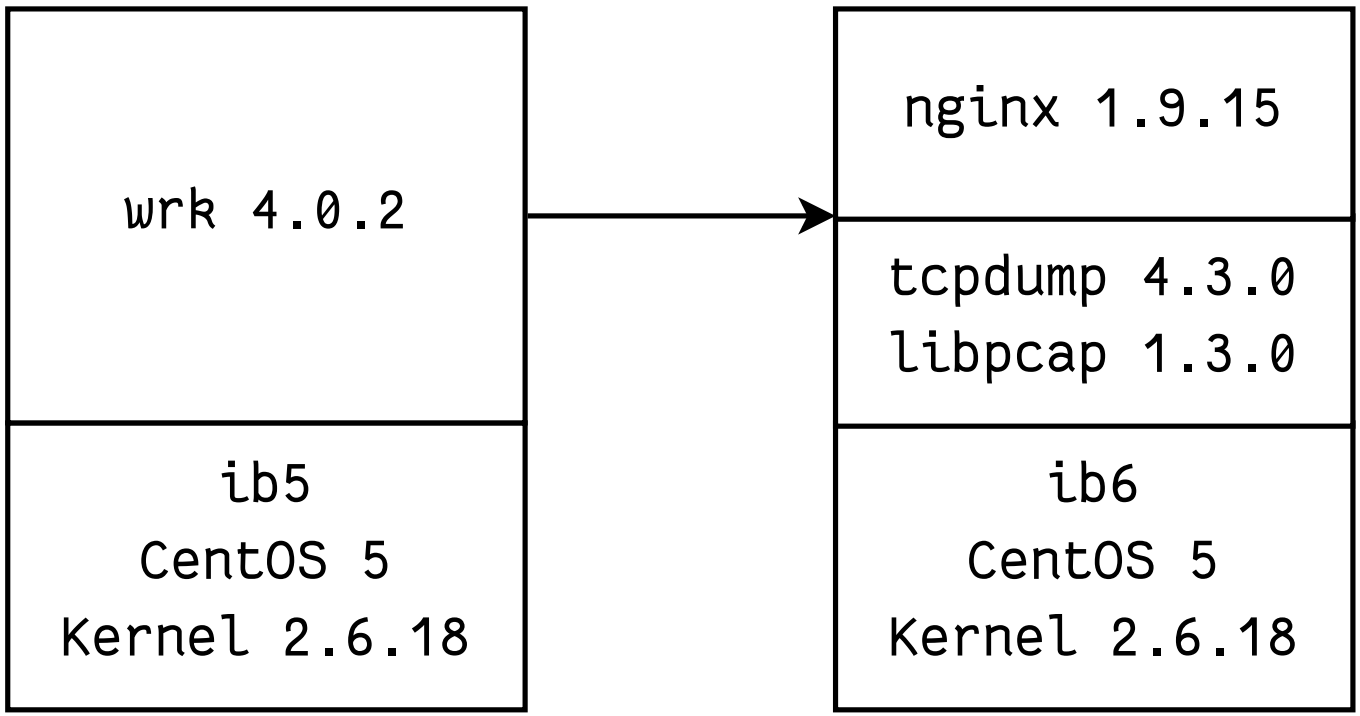
Konzept

Konzept

- HTTP Benchmark mit einer hohen Anzahl an Requests
- Minimale HTTP Responses
- Auswirkung von tcpdump Parametern untersuchen

Konzept

Versuchsaufbau



Konzept

Versuchsaufbau

- wrk -t 4 -c 1024 -d 5m http://ib6
- tcpdump -i eth1 -w dump.pcap -s SNAPLEN -B BUFFER FILTER

Konzept

nginx Konfiguration

```
user root;
worker_processes  4;

events {
    use epoll;
    worker_connections  1024;
    multi_accept on;
}

error_log /dev/null crit;

http {
    access_log off;

    server {
        listen 80;

        location = / {
            return 204;
        }
    }
}
```

Konzept

Szenarien

Name	snaplen	buffer	filter
no			—
default	65535	2048	ip dst host 172.16.0.26 and tcp dst port 80
snaplen	142	2048	ip dst host 172.16.0.26 and tcp dst port 80
buffer	65535	4096	ip dst host 172.16.0.26 and tcp dst port 80
snaplen+buffer	142	4096	ip dst host 172.16.0.26 and tcp dst port 80
filter	142	4096	ip dst host 172.16.0.26 and tcp dst port 80 and 'tcp[((tcp[12:1] & 0xf0) » 2):4] = 0x47455420'

Tabelle: Konfiguration von tcpdump für Messszenarien

Konzept

Metriken

- Anzahl der gesendeten HTTP Requests (wrk)
- Anzahl der gefilterten Netzwerkpakete (tcpdump)
- Anzahl der verlorenen Netzwerkpakete (tcpdump)

Messungen

Messungen

- Messungen jeweils 5 Minuten und 7 Wiederholungen
- Ausgabe von wrk und tcpdump gespeichert
- Von allen Metriken wurde er Median genutzt

Messungen

wrk Ausgabe

```
Running 5m test @ http://ib6
  4 threads and 1024 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency     5.32ms   13.26ms 610.71ms   92.91%
    Req/Sec   101.27k    15.99k  163.25k    58.68%
  120910283 requests in 5.00m, 12.38GB read
Requests/sec: 403008.38
Transfer/sec:     42.26MB
```

Messungen

tcpdump Ausgabe

```
tcpdump: listening on eth1, link-type EN10MB (Ethernet),
 capture size 65535 bytes
94084513 packets captured
95945157 packets received by filter
1860644 packets dropped by kernel
```

Messungen

Ergebnisse

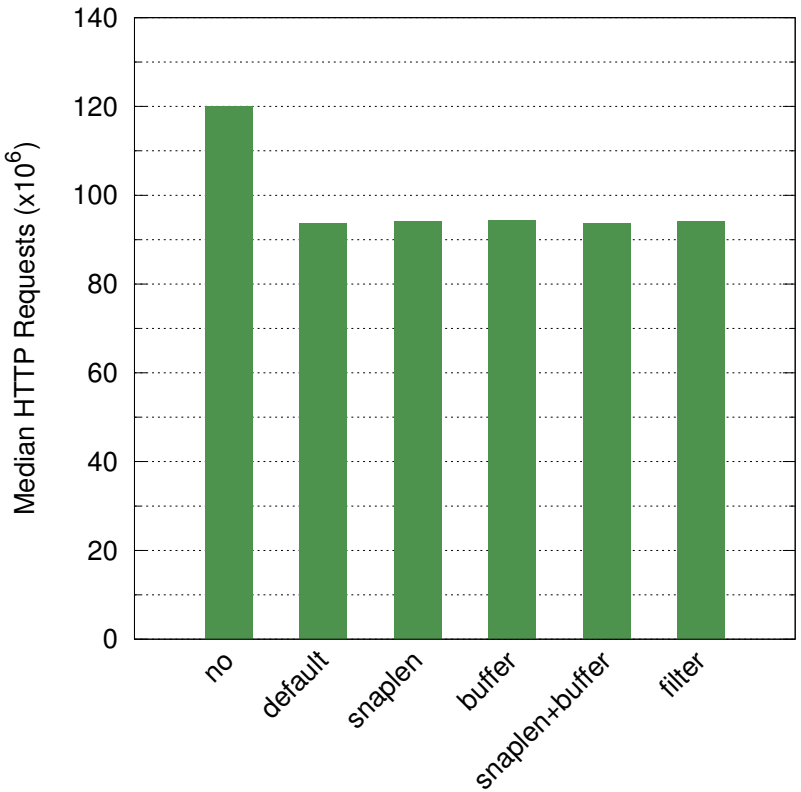


Abbildung: Anzahl der von wrk gesendeten HTTP Requests

Messungen

Ergebnisse

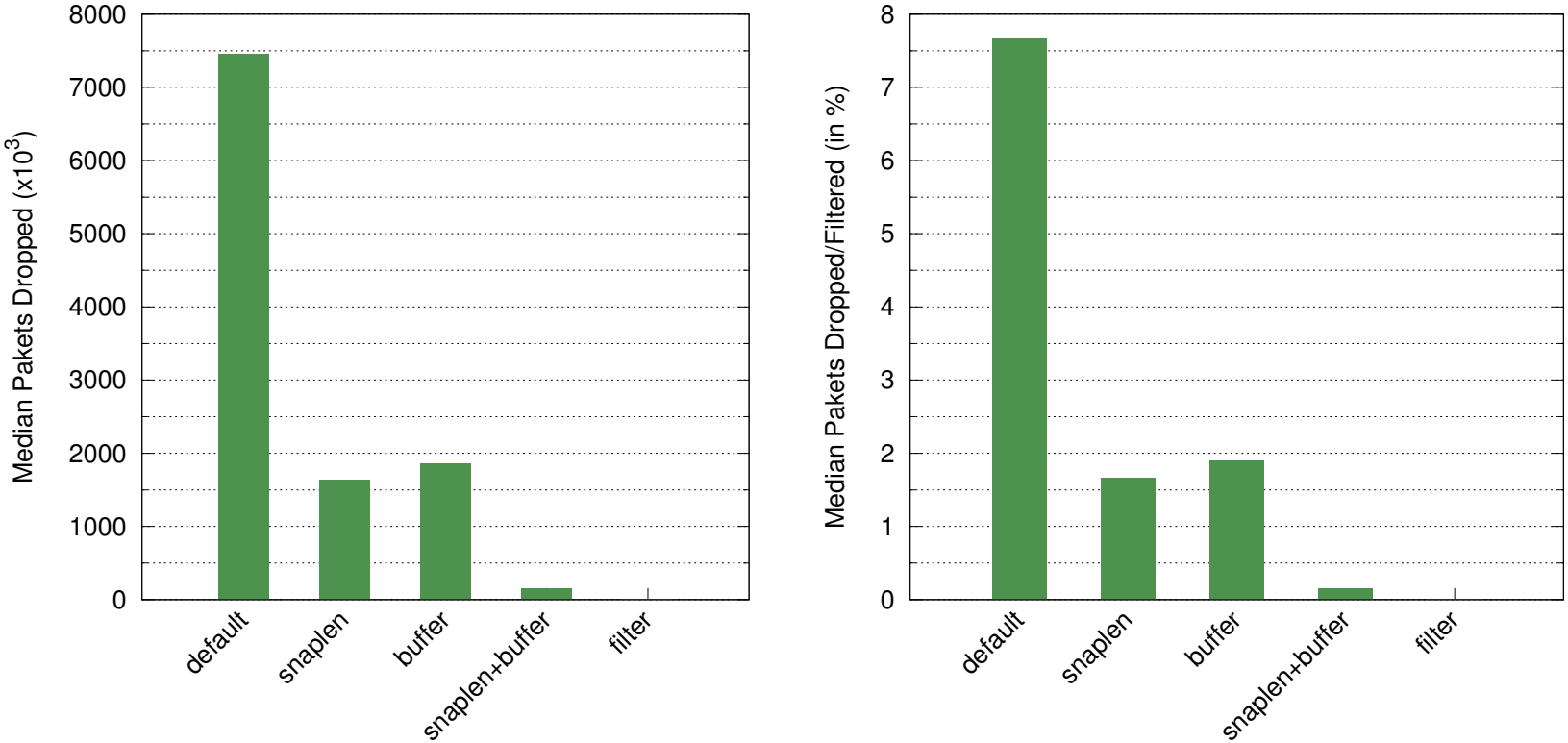


Abbildung: Paketverlust bevor tcpdump die Pakete auswerten konnte

Messungen

Ergebnisse

	Requests	Requests/s	Filtered	Dropped	Dropped %
no	120069354	400217,48			
default	93728797	312360,77	97520639	7446903	7,662 %
snaplen	94144165	313743,49	97952561	1634344	1,661 %
buffer	94197335	313920,03	98005865	1860171	1,898 %
snaplen+buffer	93635020	312040,81	97422724	142459	0,145 %
filter	94126289	313676,56	94128018	3613	0,004 %

Tabelle: Median der Messwerte für alle Szenarien

Zusammenfassung

Zusammenfassung

- Unter hoher Last ist Paketverlust wahrscheinlich
- Optimierte Parameter können den Paketverlust reduzieren
- Zählen von HTTP Requests ist sehr speziell und optimierbar
- tcpdump sollte getrennt von SUT betrieben werden