# Netzwerkzugriff einschränken unter Linux

Johannes

Winterkongress

2. März 2024

*Linux ist weniger sicher als Windows oder Mac.*

# Nicht weil es nicht sein kann...

```
[Service]
Type=dbus
BusName=net.connman.iwd
ExecStart=/usr/lib/iwd/iwd
NotifyAccess=main
LimitNPROC=1
Restart=on-failure
CapabilityBoundingSet=CAP_NET_ADMIN CAP_NET_RAW CAP_NET_BIND_SERVICE
PrivateTmp=true
NoNewPrivileges=true
DevicePolicy=closed
DeviceAllow=/dev/rfkill rw
ProtectHome=yes
ProtectSystem=strict
ProtectControlGroups=yes
ProtectKernelModules=yes
ConfigurationDirectory=iwd
```

. . . sondern, wie es genutzt wird: Vollzugriff auf Daten

Moved ~/.local/share/steam. Ran steam. It deleted everything on system owned by user. #3671

Closed **keyvin** opened this issue on Jan 14, 2015 · 266 comments

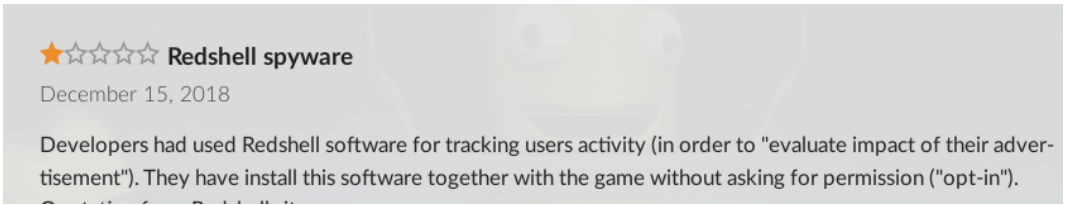
1

[1] https://github.com/ValveSoftware/steam-for-linux/issues/3671

# Sandbox

Firejail Security Sandbox

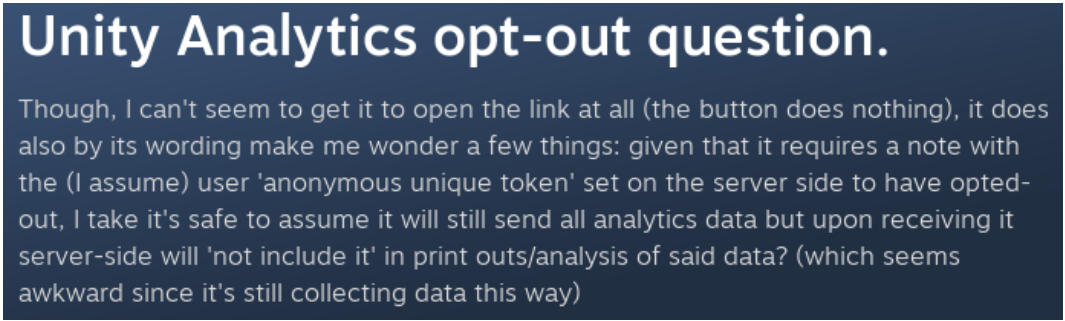
# Ohne mitgeliefertes Profil: Spiele (1)




2

[2] https://www.gog.com/game/kerbal_space_program

# Ohne mitgeliefertes Profil: Spiele (2)




3

[3] https://steamcommunity.com/app/640820/discussions/0/3118147979117406946/

# Eingeschränkter Netzwerkzugriff mit Firejail

```
$ man firejail-profile
...
net none
  Enable a new, unconnected network namespace. The only interface
  available in the new namespace is a new loopback interface (lo).
  Use this option to deny network access to programs that don't
  really need network access.
```

# Johannes

- Studium: Wirtschaftsinformatik
- Forschungsassistent E-Health
- Systemadministratorrollen:
  - Content Delivery Network
  - Telekommunikation
  - Cloud-Dienstleistung
- Kein Zusammenhang zwischen Arbeit und Vortrag
- Interessantes Thema: Teilen von dem, was ich gelernt habe

# Netzwerk-Namensräume?

Auch: Network namespace, netns

# Hintergrund: Netzwerkgeräte

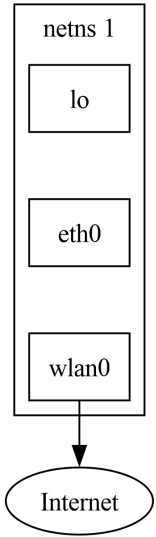
```
$ ip -brief link show
lo     UNKNOWN  00:00:00:00:00:00 <LOOPBACK,UP,LOWER_UP>
eth0   DOWN     00:19:b8:54:09:f3 <BROADCAST,MULTICAST>
wlan0  UP       aa:69:97:bb:28:ee <BROADCAST,MULTICAST,UP,LOWER_UP>
```

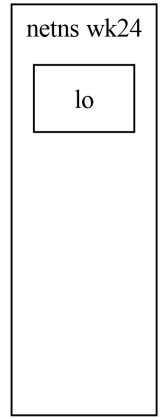
4

[4] Kurzform des gleichen Befehls: `ip -br l`

# Netzwerk-Namensraum erstellen

```
# ip netns add wk24
# ip netns exec wk24 ip -brief link show
lo     DOWN     00:00:00:00:00:00 <LOOPBACK>
```


netns 1
lo
eth0
wlan0
Internet

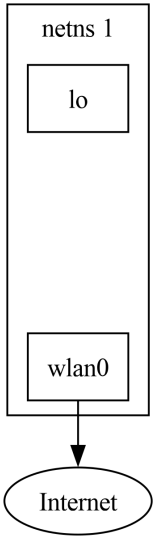

netns wk24
lo

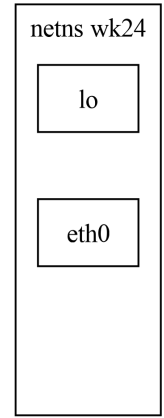
# Netzwerkgerät in Netzwerk-Namensraum verschieben

```
# ip link set eth0 netns wk24
# ip netns exec wk24 ip -brief link show
lo     DOWN     00:00:00:00:00:00 <LOOPBACK>
eth0   DOWN     00:19:b8:54:09:f3 <BROADCAST,MULTICAST>
# ip -brief link show
lo     UNKNOWN  00:00:00:00:00:00 <LOOPBACK,UP,LOWER_UP>
wlan0  UP       aa:69:97:bb:28:ee <BROADCAST,MULTICAST,UP,LOWER_UP>
# ip netns exec wk24 ip link set eth0 netns 1
```

5

[5] Für WLAN: `iw phy phy0 set netns name wk24`


netns 1
lo
wlan0
Internet

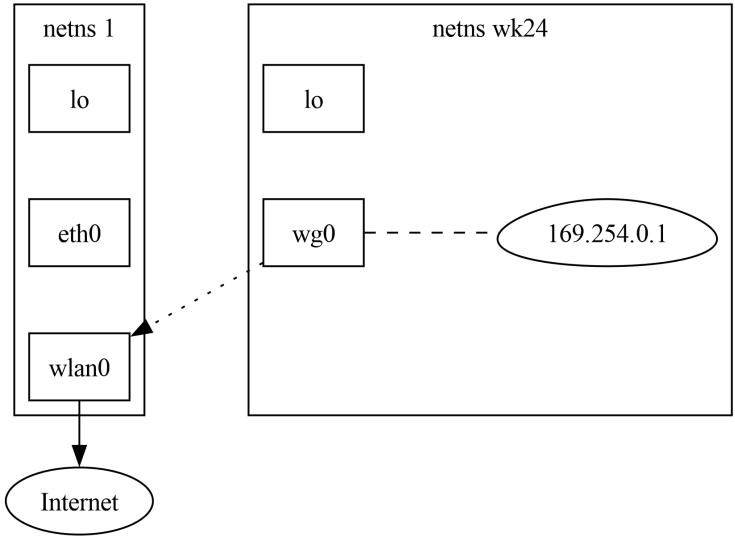

netns wk24
lo
eth0

# Netzwerk-Namensraum für Virtual Private Network (VPN)

Erlaube Verbindungen zum VPN nur in einem Netzwerk-Namensraum ohne Zugriff zum Internet.

6

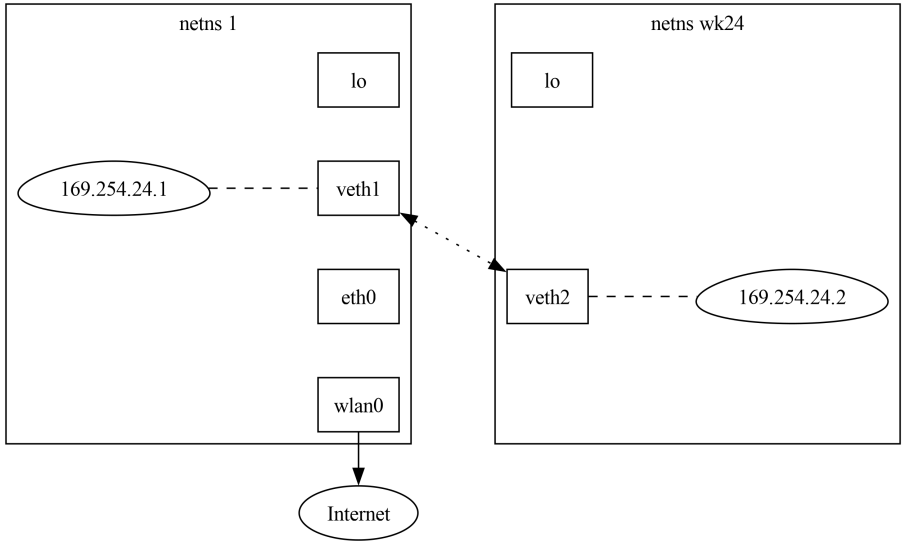
[6] https://www.wireguard.com/netns/


netns 1
lo
eth0
wlan0
Internet
netns wk24
lo
wg0
169.254.0.1

# Virtuelle Netzwerkgeräte

`veth` - Virtuelle Verbindung zwischen zwei Netzwerkgeräten auf einer Maschine

```
# ip link add veth1 type veth peer name veth2 netns wk24
```


netns 1
lo
169.254.24.1
veth1
eth0
wlan0
Internet
netns wk24
lo
veth2
169.254.24.2

# Schicht 2 Broadcast-Domäne und Routing

Senden von Paketen:

- 169.254.24.1 - 169.254.24.254[7]: Direkt verbunden, kann Pakete direkt senden.
- Alle anderen Adressen: Ein anderes Gerät (Router) muss Pakete weiterleiten.

[7] Weil die Adresse als 169.254.24.2/**24** konfiguriert wurde.

# IP Forwarding

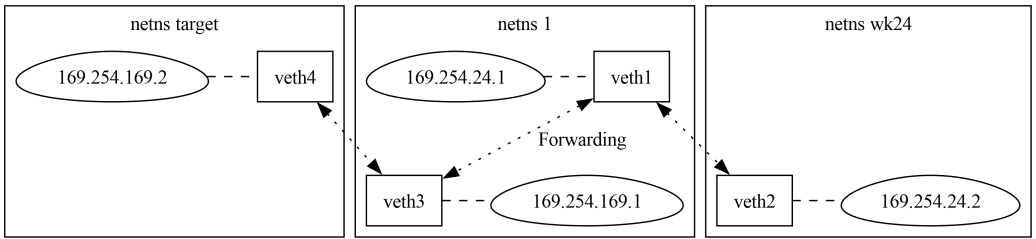



Figure 5:

*Eingeschränkte* Verbindung, nicht *keine* Verbindung



# Proxy

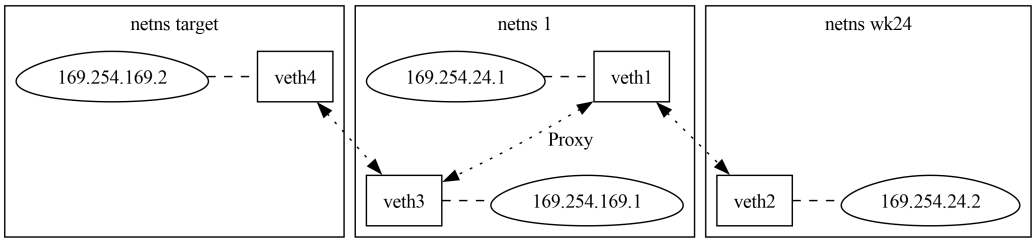


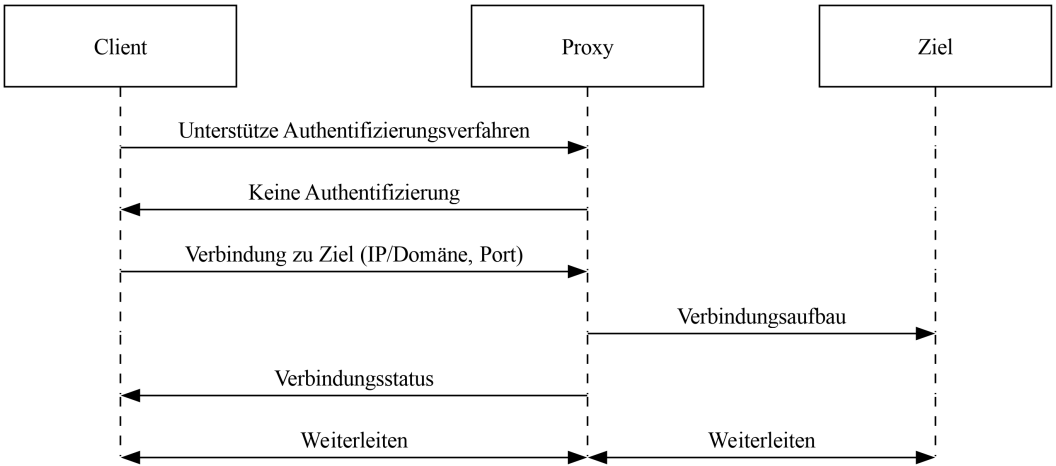

Figure 6:

```rust
let listener = TcpListener::bind((Ipv4Addr::new(169, 254, 24, 1), 8000))?;

loop {
    let (mut incoming, _) = listener.accept()?;
    let mut outgoing = TcpStream::connect(
        (Ipv4Addr::new(169, 254, 169, 2), 8000))?;

    let mut buf = [0u8; 4096];
    loop {
        let bytes_read = incoming.read(&mut buf)?;
        if bytes_read == 0 {
            break;
        }
        outgoing.write_all(&buf[..bytes_read])?;
    }
}
```

# Socks5-Proxy



```rust
socks.handle_authentication()?;
let connect_target = socks.get_connect_target()?;
let allow_connection = match &connect_target {
    ConnectTarget::IPAddress(IpAddr::V4(ip), _port) => {
        ip == &Ipv4Addr::new(169, 254, 169, 2)
    }
    ConnectTarget::Domain(domain, _port) => domain == "www.digiges.ch",
    _ => false,
};
if !allow_connection {
    return socks.refuse_connection();
}
let outgoing = socks.establish_connection_to_target(connect_target)?;
let mut copy = BidirectionalStreamCopy::new(incoming, outgoing)?;
copy.copy_streams()
```

Problem gelöst?

```
# whoami
```

Problem gelöst? Nein! [8]

```
# whoami
root
```

[8] Setuid-Programme wie `firejail` oder `sudo` können die Ausführung als Root stark einschränken.

# Hintergrund: Handles aka file descriptor / fd

```
# ls -o /dev/std*
lrwxrwxrwx 1 ... /dev/stderr -> /proc/self/fd/2
lrwxrwxrwx 1 ... /dev/stdin -> /proc/self/fd/0
lrwxrwxrwx 1 ... /dev/stdout -> /proc/self/fd/1
# ls -vo /proc/$(pidof wget)/fd/*
lrwx------ 1 ... /proc/3182/fd/0 -> /dev/pts/3
lrwx------ 1 ... /proc/3182/fd/1 -> /dev/pts/3
lrwx------ 1 ... /proc/3182/fd/2 -> /dev/pts/3
lrwx------ 1 ... /proc/3182/fd/3 -> 'socket:[232382]'
l-wx------ 1 ... /proc/3182/fd/4 -> /tmp/1MB
```

# Hintergrund: Systemaufruf (1) aka System Call - syscall

```c
#include <unistd.h>
#include <stdio.h>

void main() {
    // Zeige Ausgabe sofort, nicht nur nach jeder Zeile.
    setbuf(stdout, NULL);

    printf("Hallo ");
    sleep(1);
    printf("WK24\n");
}
```

```
$ gcc hallo.c -o hallo
```

# Hintergrund: Systemaufruf (2)

```
$ strace ./hallo > /dev/null
execve("./hallo", ["./hallo"], 0x7ffeee95df80 /* 38 vars */) = 0
brk(NULL)                               = 0x61562318c000
access("/etc/ld.so.preload", R_OK)      = 0
openat(AT_FDCWD, "/etc/ld.so.preload", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=0, ...}) = 0
close(3)                                = 0
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=176795, ...}) = 0
mmap(NULL, 176795, PROT_READ, MAP_PRIVATE, 3, 0) = 0x75a28454a000
close(3)                                = 0
openat(AT_FDCWD, "/usr/lib/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220~\2\0\0\0\0"..., 832) = 832
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0@\0\0\0\0\0\0@\0\0\0\0\0\0"..., 784, 64) = 784
fstat(3, {st_mode=S_IFREG|0755, st_size=1948952, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x75a28454d000
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0@\0\0\0\0\0\0@\0\0\0\0\0\0"..., 784, 64) = 784
mmap(NULL, 1973104, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x75a284366000
mmap(0x75a28438a000, 1421312, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x24000) = 0x75a28438a000
mmap(0x75a2844e5000, 348160, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x17f000) = 0x75a2844e5000
mmap(0x75a28453a000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1d3000) = 0x75a28453a000
mmap(0x75a284540000, 31600, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x75a284540000
close(3)                                = 0
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x75a284363000
arch_prctl(ARCH_SET_FS, 0x75a284363740) = 0
set_tid_address(0x75a284363a10)         = 19139
set_robust_list(0x75a284363a20, 24)     = 0
rseq(0x75a284364060, 0x20, 0, 0x53053053) = 0
mprotect(0x75a28453a000, 16384, PROT_READ) = 0
mprotect(0x615622158000, 4096, PROT_READ) = 0
mprotect(0x75a2845a9000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
munmap(0x75a28454a000, 176795)          = 0
write(1, "Hallo ", 6)                   = 6
clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=1, tv_nsec=0}, 0x7fffff677050) = 0
write(1, "WK24", 4)                     = 4
write(1, "\n", 1)                       = 1
exit_group(5)                           = ?
+++ exited with 5 +++
```

# Hintergrund: Systemaufruf (3)

```
$ strace -e write,/sleep ./hallo > /dev/null
write(1, "Hallo ", 6)                       = 6
clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=1, tv_nsec=0}, 0x7f...) = 0
write(1, "WK24", 4)                         = 4
write(1, "\n", 1)                           = 1
```

# Was passiert beim Erstellen eines Netzwerk-Namensraumes?

```
# ip netns del wk24
# 2>&1 strace -e mount,unshare -e q=attach,exit \
  ip netns add wk24 | grep -v MS_SHARED
unshare(CLONE_NEWNET)                    = 0
mount("/proc/self/ns/net", "/var/run/netns/wk24", 0x68...,
  MS_BIND, NULL) = 0
```

# Ausführen in einem Netzwerk-Namensraum.

```
# 2>&1 strace -e setns,openat -e q=attach,exit \
  ip netns exec wk24 sleep 0 | grep -v '/usr\|/etc\|/proc\|""\|/sys'
openat(AT_FDCWD, "/var/run/netns/wk24", O_RDONLY|O_CLOEXEC) = 5
setns(5, CLONE_NEWNET)                  = 0
```

# Nutzer-Namensräume (user namespaces)

```
$ unshare --user --net ip -brief link show
lo     DOWN     00:00:00:00:00:00 <LOOPBACK>
$ strace -e unshare -e q=exit unshare --user --net \
  ip -brief link show > /dev/null
unshare(CLONE_NEWUSER|CLONE_NEWNET)     = 0
9
```

[9] Siehe auch: `man user_namespaces`

# Privilegien (1)

```
$ whoami
wk24
$ unshare --user --net sh -c \
"ip link set lo up; whoami"
RTNETLINK answers: Operation not permitted
nobody
$ unshare --user --net --map-root-user sh -c \
"ip link set lo up; whoami"
root
$ unshare --user --net --keep-caps sh -c \
"ip link set lo up; whoami"
nobody
```

# Privilegien (2)

```
$ captest --text | grep Current
Current capabilities: none
$ unshare --user --keep-caps \
  captest --text | grep "^Effective"
Effective: chown, dac_override, dac_read_search, fowner, fsetid, kill,
setgid, setuid, setpcap, linux_immutable, net_bind_service, net_broadcast,
net_admin, net_raw, ipc_lock, ipc_owner, sys_module, sys_rawio,
sys_chroot, sys_ptrace, sys_pacct, sys_admin, sys_boot, sys_nice,
sys_resource, sys_time, sys_tty_config, mknod, lease, audit_write,
audit_control, setfcap, mac_override, mac_admin, syslog, wake_alarm,
block_suspend, audit_read, perfmon, bpf, checkpoint_restore
```

# Handle-Übertragung zwischen Prozessen (fd passing)

Kann Handle in einem Prozess erstellen und die verbundene Ressource über eine Unix-Sockel-Verbindung an einen zweiten Prozess schicken. [10]

[10] Suche nach `SCM_RIGHTS` in `man unix`; https://sigma-star.at/blog/2023/05/sandbox-netns/

```rust
let (parent_sock, netns_sock) = UnixStream::pair().unwrap();
let mut command = std::process::Command::new("bash");
let command = unsafe {
  command.pre_exec(move || {
    // Die Abstraktion aktiviert `lo` wenn eine Netns erstellt wird.
    let unshare = Unshare::new(
      &[CloneFlags::Newuser, CloneFlags::Newnet])?;
    let socks_listener = TcpListener::bind((Ipv4Addr::LOCALHOST, 1080))?;
    netns_sock.send_fd(socks_listener.as_raw_fd())
  })
};
let mut child = command.spawn().unwrap();

let socks_listen_fd = parent_sock.recv_fd()?;
let socks_listener = unsafe{ TcpListener::from_raw_fd(socks_listen_fd) };
```

# Socks5-Proxy mit Listening-Socket in anderem Netzwerk-Namensraum

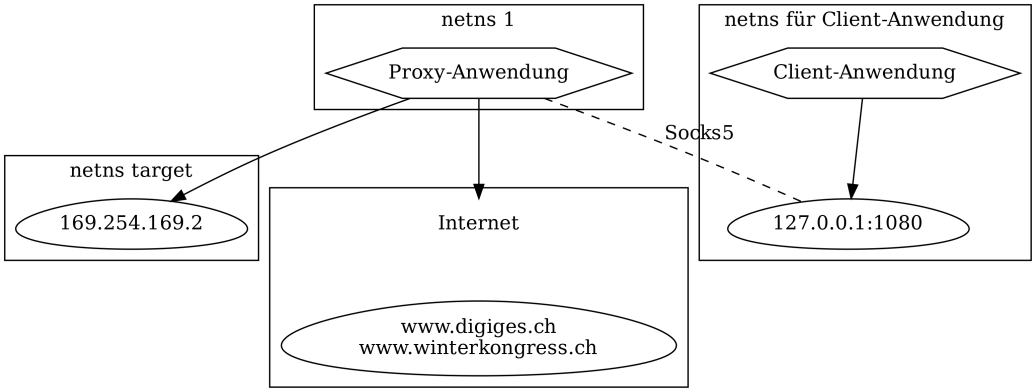



Figure 8:

# Konfigurationsdateien und Mount-Namensräume

```
# 2>&1 strace -e unshare,openat,mount -e q=attach,exit \
  ip netns exec wk24 sleep 0 | grep '/etc/netns\|unshare'
unshare(CLONE_NEWNS)                        = 0
openat(AT_FDCWD, "/etc/netns/wk24", ...
  O_RDONLY|O_NONBLOCK|O_CLOEXEC|O_DIRECTORY) = 5
mount("/etc/netns/wk24/nsswitch.conf", "/etc/nsswitch.conf", ...
  0x62344de0b7eb, MS_BIND, NULL) = 0
mount("/etc/netns/wk24/resolv.conf", "/etc/resolv.conf", ...
  0x62344de0b7eb, MS_BIND, NULL) = 0
```

# Zeitüberprüfung

Keine Zeit? -> Fragen!

Noch Zeit? -> Mehr Inhalt.

# Wie funktionieren Netzwerkverbindungen? (Beispiel HTTP)

Verbindung zu www.digiges.ch:

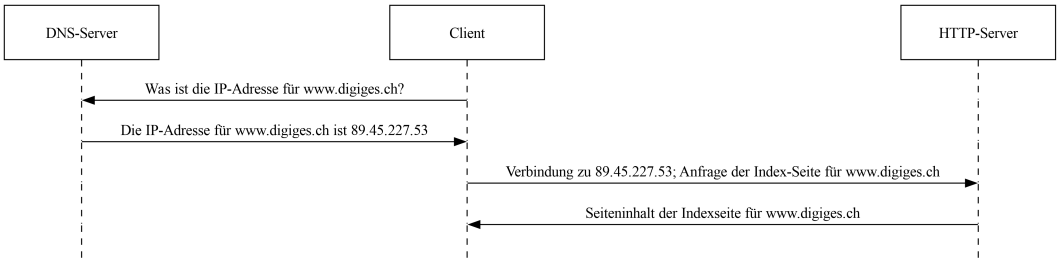



Figure 9:

# Einschränken einer Netzwerkverbindung

1. Filterung von DNS-Anfragen (Sende keine / falsche Antwort)
   - Der Ad-Blocker PiHole basiert auf diesem Prinzip.
2. Aufbau von Verbindungen nur zu IP-Adressen, die zu gewünschten Zielen gehören.
   - Firewall oder Proxy

# “NAT”-Proxy (Network Address Translation)

- Proxy mit Netzwerk-Namensraum für Kinds-Prozess wie im Socks5-Beispiel
- DNS-Anfragen an Proxy
  - Angepassten `/etc/nsswitch.conf` und `/etc/resolv.conf` Dateien.
  - DNS-Anfragen für erlaubte Ziele antworten mit einer internen Adresse
  - Tabelle von interner Adresse zu Ziel
- Proxy
  - Warte auf Anfragen auf einer Vielzahl interner Adressen.
  - Wenn Anfrage auf einer internen Adresse erhalten, baue Proxy-Verbindung zu eigentlichem Ziel entsprechend der Tabelle auf.

# “NAT”-Proxy: Beispiel

DNS:

- www.winterkongress.ch → 169.254.0.3 (interne Adresse)
- www.digiges.ch → 169.254.0.4 (interne Adresse)

Proxy:

- Verbindung zu 169.254.0.3 → Proxy-Verbindung zu www.winterkongress.ch
- Verbindung zu 169.254.0.4 → Proxy-Verbindung zu www.digiges.ch

# “NAT”-Proxy mit Listening-Socket in anderem Netzwerk-Namensraum [11]

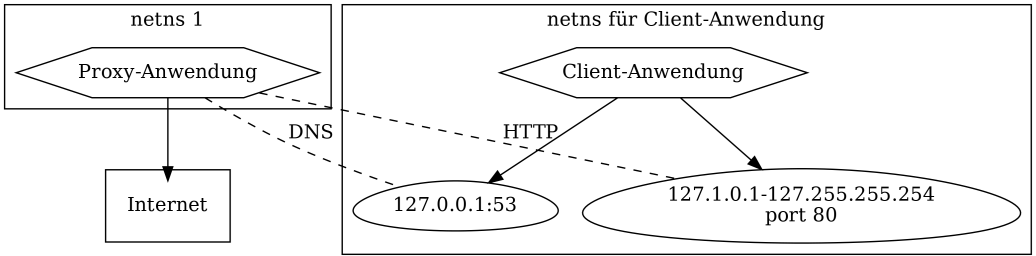



Figure 10:

[11] HTTP Listening-Sockel ist 0.0.0.0, d.h. alle lokalen IP-Adressen

# “Zaubertrick” - Anyip [12] [13]

Kann IP-Address-Range dem loopback-Gerät zuweisen.

```
$ ip route show table local
local 127.0.0.0/8 dev lo proto kernel scope host src 127.0.0.1
local 127.0.0.1 dev lo proto kernel scope host src 127.0.0.1
broadcast 127.255.255.255 dev lo proto kernel scope link src 127.0.0.1
```

`127.0.0.0/8` → Kann Verbindungen zu 127.0.0.1 - 127.255.255.254 aufbauen.

Befehl für andere Ranges:

```
# ip route add local 192.168.0.0/24 dev lo
```

[12] https://blog.cloudflare.com/how-we-built-spectrum/
[13] https://blog.widodh.nl/2016/04/anyip-bind-a-whole-subnet-to-your-linux-machine/

# Fragen, Antworten und Kontakt

Folien und Quellcode werde ich auf der Vortragsseite veröffentlichen:

https://winterkongress.ch/2024/talks/netzwerkzugriff_fur_anwendungen_einschranken_unter_linux/

Kontakt: johannes-winterkongress@ikmail.com

# Bonus: Ich entscheide eine Woche vor dem Vortrag, dass ich LibreOffice nicht mag.

```
---

# Ohne mitgeliefertes Profil: Spiele (2)

![](images/unity-analytics.png)
^[https://steamcommunity.com/app/640820/discussions/0/3118147979117406946/]

---

# Eingeschränkter Netzwerkzugriff mit Firejail

```
$ man firejail-profile
...
net none
  Enable a new, unconnected network namespace. The only interface
  available in the new namespace is a new loopback interface (lo).
  Use this option to deny network access to programs that don't
  really need network access.
```
```

# Bonus: Und ich habe Latex oder dot in den letzten 10 Jahren kaum genutzt.

````
```{.graphviz dpi=300}
digraph N {
    subgraph cluster_netns_1 {
        label = "netns 1";

        proxy[shape = "hexagon", label="Proxy-Anwendung"];
    }
    subgraph cluster_netns_wk24 {
        label = "netns für Client-Anwendung";

        listen_dns[shape = "egg", label="127.0.0.1:53"]
        listen_http[shape = "egg", label="127.1.0.1-127.255.255.254\nport 80"]
        client[shape = "hexagon", label="Client-Anwendung"];

        client -> listen_dns;
        client -> listen_http;
    }
    subgraph cluster_internet {
        internet[label="Internet", shape="plaintext"];
    }

    proxy -> listen_dns[style="dashed"; arrowhead=none, label="DNS"];
    proxy -> listen_http[style="dashed"; arrowhead=none, label="HTTP"];
    proxy -> internet;
}
```
````

# Bonus: Was zur Hölle ist eine “KDE XML syntax definition”-Datei?

```
<?xml version="1.0" encoding="UTF-8"?>
<language name="strace" version="1" kateversion="2.4" section="Sources" extensions="*.invalid">
  <highlighting>
    <list name="syscalls">
      <item>unshare</item>
      <item>mount</item>
      <item>write</item>
      <item>clock_nanosleep</item>
      <item>openat</item>
      <item>setns</item>
    </list>
    <list name="flags">
      <item>CLONE_NEWNET</item>
      <item>CLONE_NEWNS</item>
      <item>CLONE_NEWUSER</item>
      <item>MS_BIND</item>
    </list>
    <contexts>
      <context attribute="Deemphasize" lineEndContext="#pop" name="Normal Text" >
    <DetectChar attribute="StartOfCommand" context="command" char="#" firstNonSpace="true" />
    <DetectChar attribute="StartOfCommand" context="command" char="$" firstNonSpace="true" />
        <keyword attribute="SysCall" context="#stay" String="syscalls" />
    <keyword attribute="Flag" context="#stay" String="flags" />
    <StringDetect attribute="SysCall" context="#stay" String="net_admin" />
    <DetectChar attribute="String" context="string" char="&amp;quot;" />
    <!-- Hacky way to highlight Handle "5", because I dont't know how to write proper rules. -->
    <DetectChar attribute="Handle" context="#pop" char="5" />
```

## Bonus: Rust kompiliert schneller als meine Folien.

```
$ find -name Cargo.toml -execdir cargo clean \;
$ /usr/bin/time -f "F: %e" find -name Cargo.toml -execdir cargo build \; \
  2>&1 | grep F
    Finished dev [unoptimized + debuginfo] target(s) in 1.75s
    Finished dev [unoptimized + debuginfo] target(s) in 1.43s
    Finished dev [unoptimized + debuginfo] target(s) in 0.21s
    Finished dev [unoptimized + debuginfo] target(s) in 1.28s
F: 4.86
```

```
$ /usr/bin/time -f "%e" make
pandoc --filter pandoc-plot -s -f markdown+yaml_metadata_block \
  --syntax-definition=strace.xml slides.md -t beamer -o slides.pdf
9.80
```