Practical Networking

Unit 17 - Hands-On Networking - 2018

Prof. Dr.-Ing. Thorsten Herfet, Andreas Schmidt, Pablo Gil Pereira

Telecommunications Lab, Saarland Informatics Campus, 26th Feb. 2018

Packet Analysis

HON

Packet Analysis (also Packet Sniffing or Protocol Analysis)

The process of capturing and interpreting **live data**, as it **flows accross a network.**

Why?

- Understanding **network characteristics**
- Learning **who is on a network**
- Determining who or what is **utilizing available bandwidth**
- Identifying possible **attacks or malicious activity**
- Finding **unsecured or bloated applications**
- Security **auditing**

How?

By using a packet sniffing software!

How Packet Sniffers Work

Collection

Packet sniffers collect **raw binary data** from the network interfaces. The selected interface is typically switched into **promiscuous mode**.

Conversion

Raw binary data is **converted into a readable format**. Usually data from the received frame is represented as a structure corresponding to the OSI layer model.

Analysis

Analysis of the captured and converted data. Most sniffers **determine the network protocol** and help the user **analyse protocol specific features**.

Wireshark

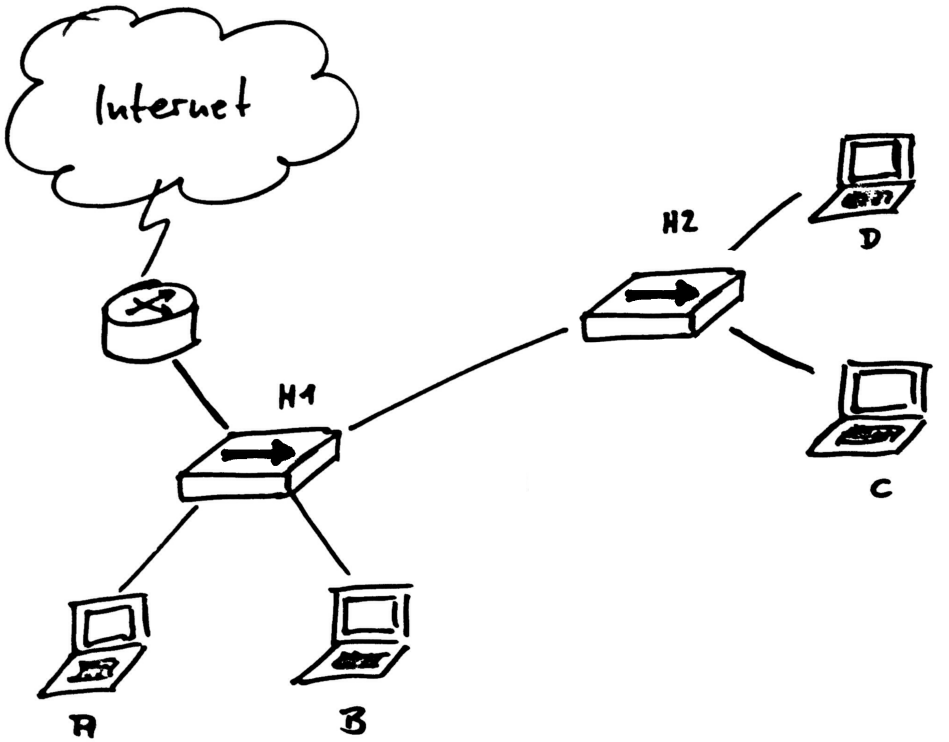
- Multi-platform open source software with more than 500 contributors
- Supports **more than 1300** network protocols
- Live data can be read from **Ethernet**, **IEEE 802.11**, PPP/HDLC, ATM, **Bluetooth**, **USB**, Token Ring, Frame Relay, FDDI, and others (depending on your platform)
- Captured live data can be analysed from a capture file
- GUI- (`wireshark`) and console-variant (`tshark`) available

Living promiscuously

- Network interfaces normally **only processes frames addressed to the local MAC address** (except for broadcast frames)
- **Frames destined for other hosts** on the network are **not passed on** to upper layers
- Network interfaces can be put into **promiscuous mode** to **pass on all received packets** regardless of the destination address

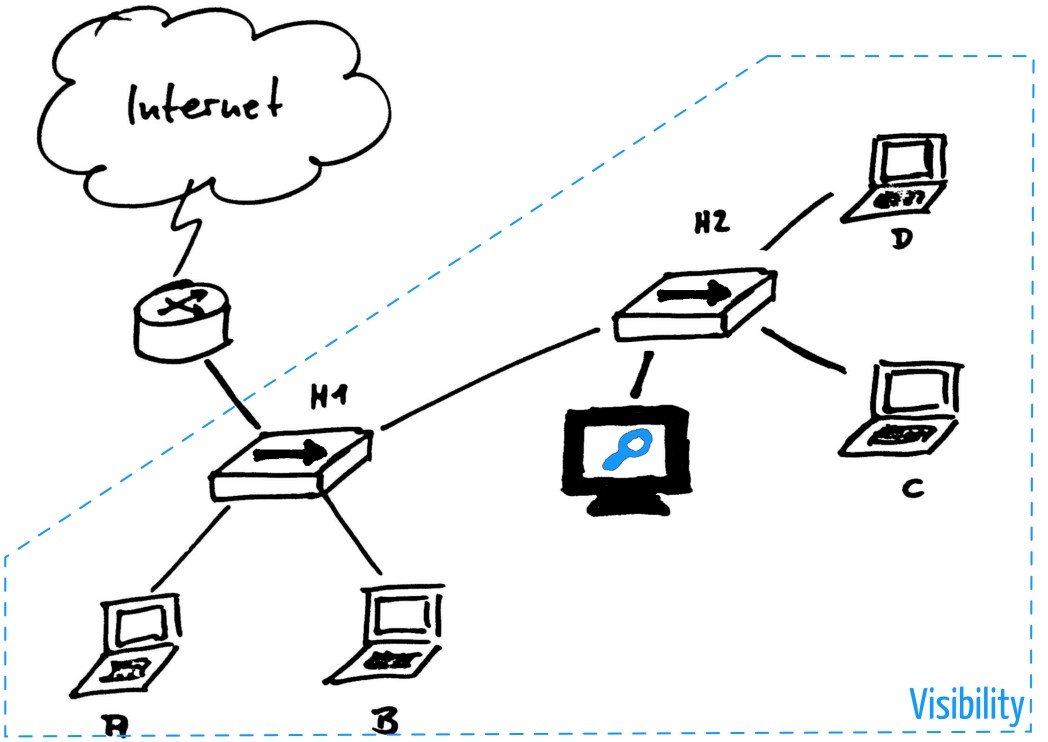
Sniffing around Hubs

❓ **Where to put our sniffer if we want to diagnose a problem with Host C?**



Sniffing around Hubs

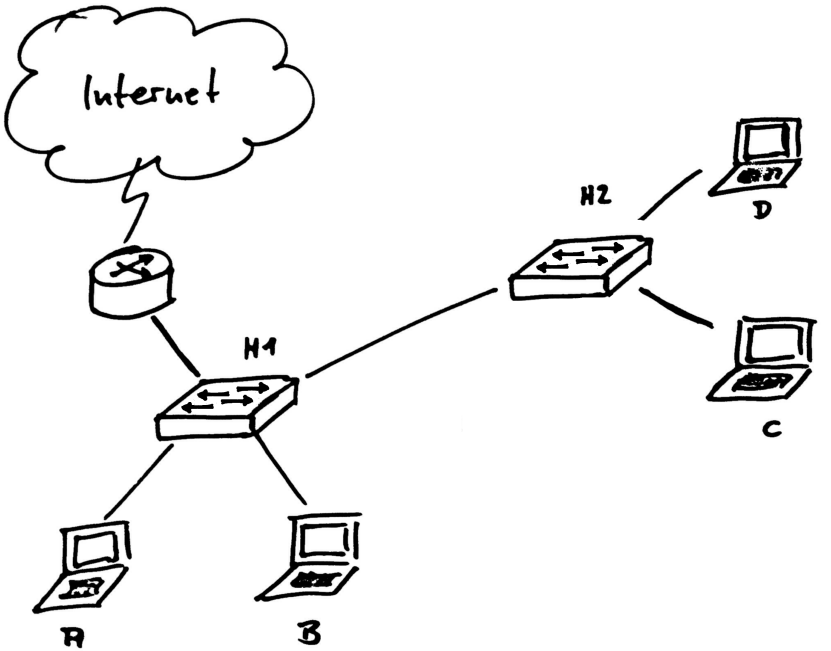
Where to put our sniffer if we want to diagnose a problem with Host C?



It doesn't really matter. All hosts are part of the **same link**.

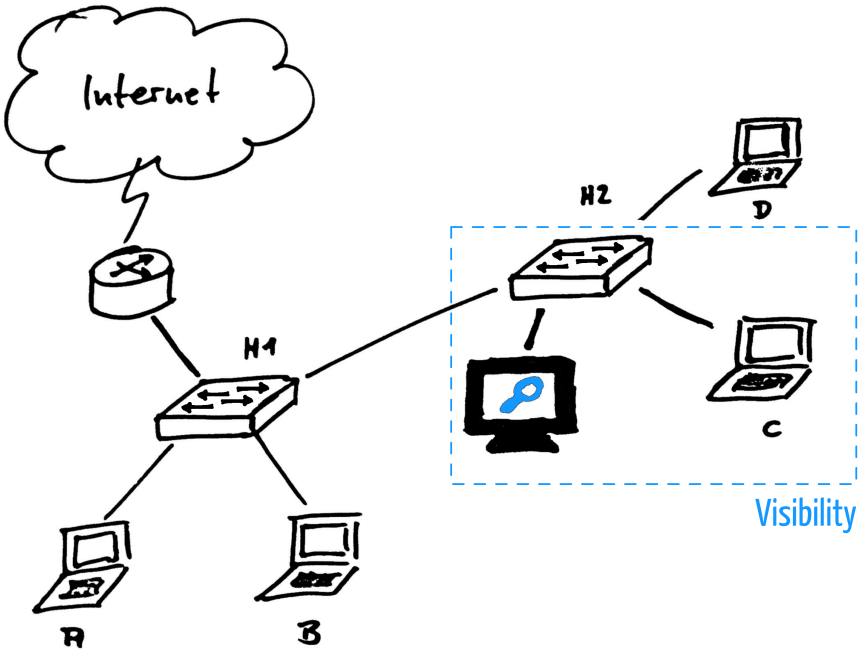
Sniffing in a Switched Environment

❓ **Where to put our sniffer if we want to diagnose a problem with Host C?**



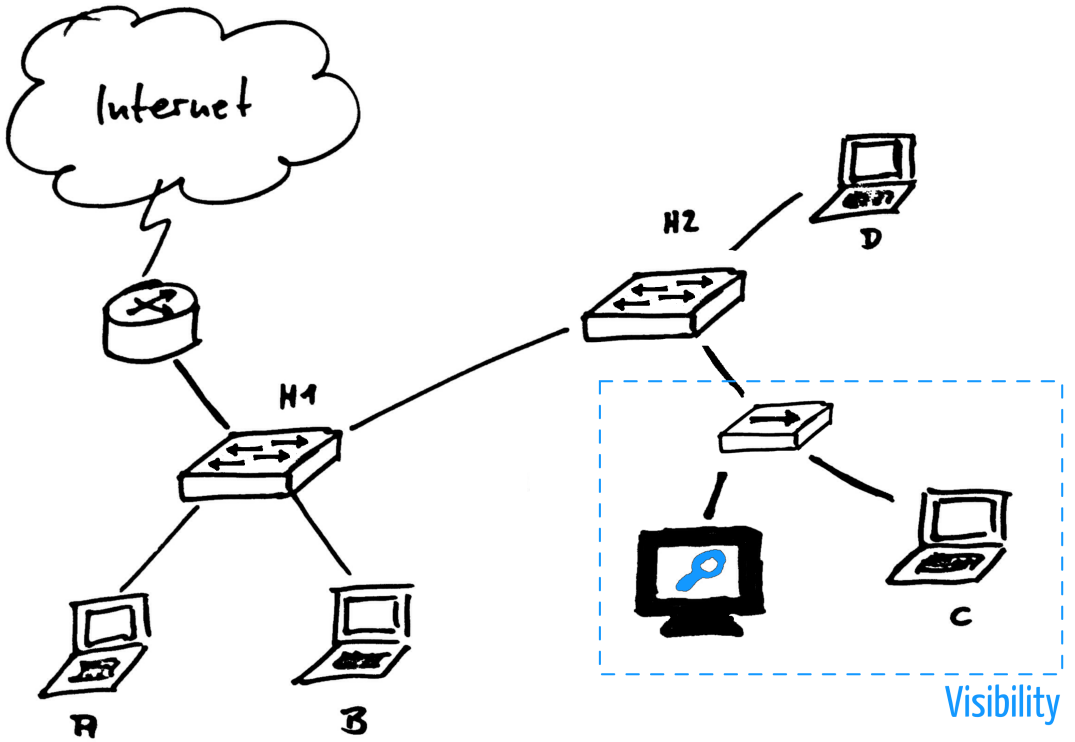
Sniffing in a Switched Environment

❓ **Where to put our sniffer if we want to diagnose a problem with Host C?**



Modern switches support **port mirroring**. All data from a selected port will be duplicated to another port for sniffing. The sniffer only has to be connected to the same switch.

Tapping



If the switch does not support port mirroring we can still **tap into the connection** of the host we want to debug by introducing a hub.
True hubs are hard to come by today, but there are **specialized network taps** available.

HON

Wireshark | Display Filters

- Filter packets displayed in wireshark
- Non-matching packets are still available in the capture, but not displayed

Literals

- **Numbers:** `810` = `0x32a` = `01452`
- **MAC Addresses:** `ff:ff:ff:ff:ff:ff` = `ff-ff-ff-ff-ff-ff`
- **IP Addresses:** `134.96.7.1`
- **Hostnames:** `www.uni-saarland.de`

Protocols and Fields

- `ip`: IP protocol packets
- `tcp`: TCP packets
- `tcp.flags.syn == 1`: TCP SYN packets
- `http`: HTTP packets

Wireshark | Display Filters

Compairing Values

Operator	Description / Example
`eq` or `==`	Equal. `ip.src == 10.0.0.5`
`ne` or `!=`	Not equal. `ip.src != 10.0.0.5`
`gt` or `>`	Greater than. `frame.len > 10`
`lt` or `<`	Less than. `frame.len < 128`
`ge` or `>=`	Greater than or equal to. `frame.len ge 0x100`
`le` or `<=`	Less than or equal to. `frame.len <= 0x20`

Bit Operations

Operator	Description / Example
`&`	Bitwise AND. `tcp.flags & 0x02` = `tcp.flags.syn`

HON

Wireshark | Display Filters

Searches and Matches

Operator	Description / Example
`contains`	Substring matching. `http contains "www.uni-saarland.de"`
`matches`	RegEx matching. `http matches ".+\.nt\.uni-saarland\.de"`

Logical Operations

Operator	Description / Example
`and` or `&&`	Logical AND
`or` or `||`	Logical OR
`not` or `!`	Logical NOT

Some examples can be found here.

Wireshark | Capture Filters

- Filter packets **before they reach the capture file**
- Non-matching packets will **not be available in the capture**
- Helpful on **very busy links** or if you know **exactly** what you are interested in

We will not cover them here. See the Wireshark Users's Guide for details

Building Networks

Back in the day ...

Graphical Network Simulator

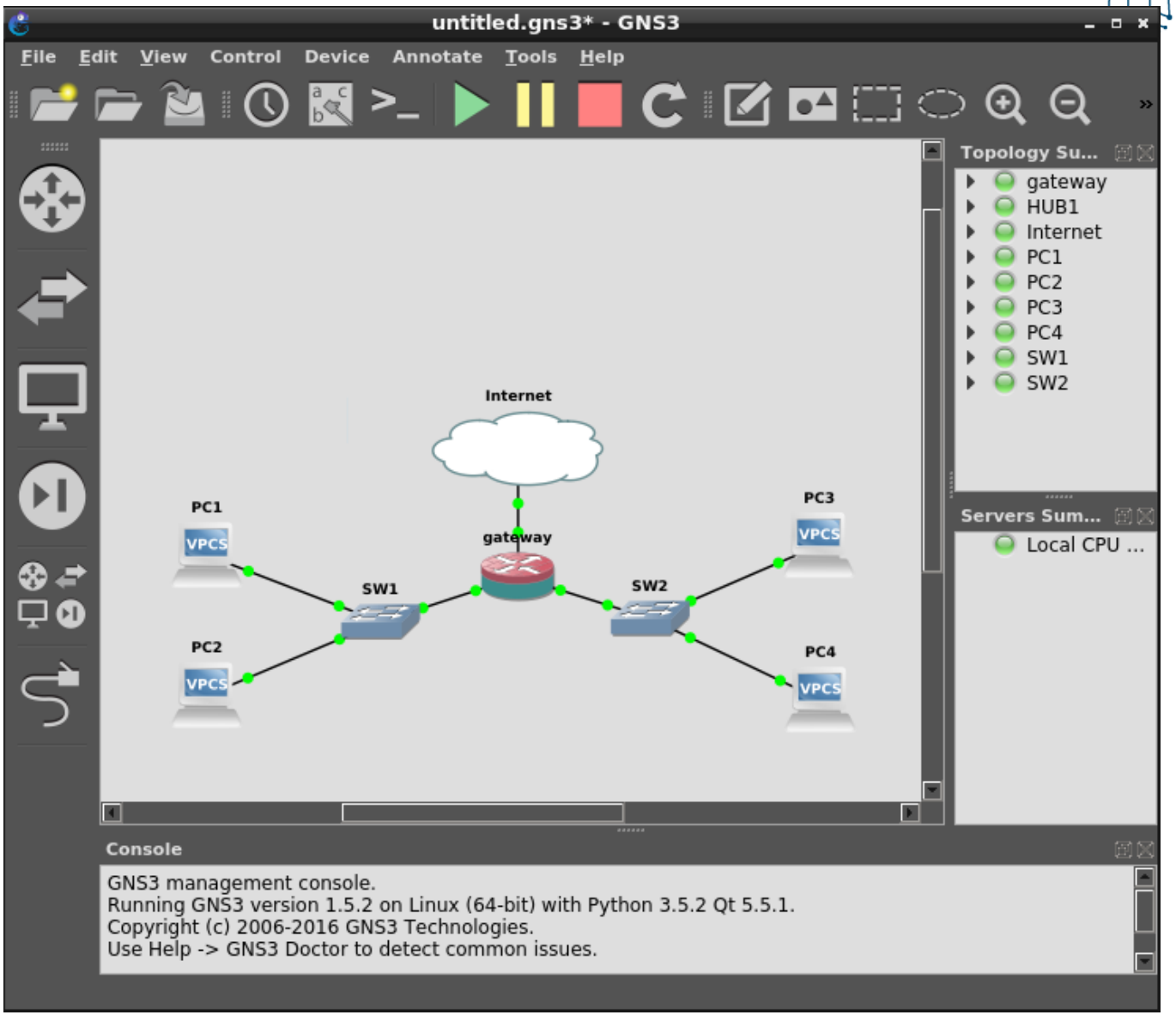
GNS3

- Open-Source cross-platform **Graphical Network Simulator**
- Let's you design and test virtual networks using **Cisco IOS, JunOS, Mikrotik RouterOS, Arista, Vyatta and Linux**
- Possible to include any **virtual machines**
- **Scalable client-server architecture**
  (Simulated networks can span more than one host)

- Cheaper than Hardware
- Easy to use (not always easy to setup)
- Virtual networks can be bridged to real ones

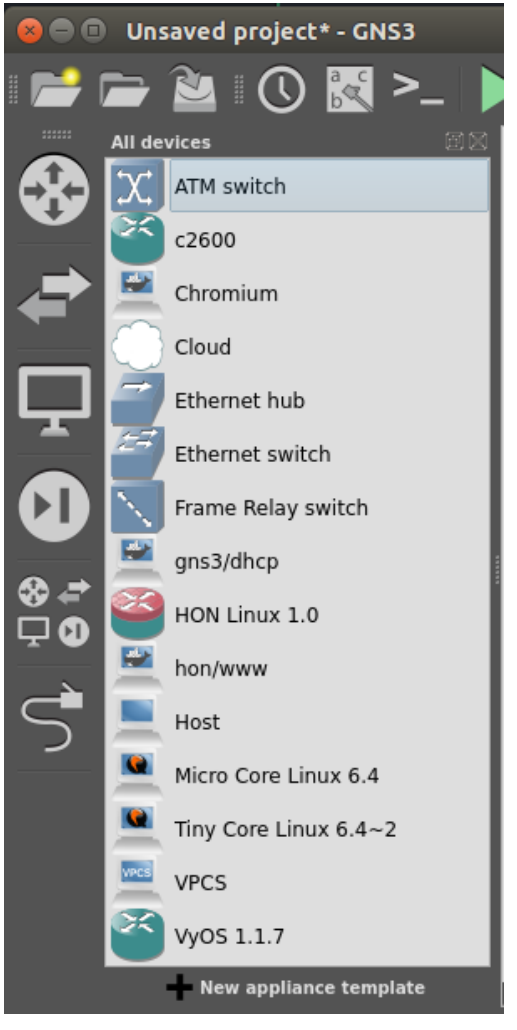
()

- Some assembly required
- Limited emulation of proprietary platforms
- Sometimes limited network performance


untitled.gns3* - GNS3
File Edit View Control Device Annotate Tools Help
Topology Su...
gateway
HUB1
Internet
PC1
PC2
PC3
PC4
SW1
SW2
Servers Sum...
Local CPU ...
Internet
gateway
PC1
PC2
PC3
PC4
SW1
SW2
VPCS
Console
GNS3 management console.
Running GNS3 version 1.5.2 on Linux (64-bit) with Python 3.5.2 Qt 5.5.1.
Copyright (c) 2006-2016 GNS3 Technologies.
Use Help -> GNS3 Doctor to detect common issues.

Appliances

- GNS3 supports virtual network appliances
  - Hubs/Switches
  - Routers
  - Browsers
  - Complete VMs
  - ...
- We included some appliances
- More can be found in the GNS3 Marketplace



HON Linux

- Alpine Linux-based embedded distribution
- Designed by us to build **routers**, **bridges** and **firewalls**
- Used in most practical units

- Included Software:
  - DHCP client
  - Dnsmasq
  - Iproute2
  - Iptables
  - Nmap
  - Wireshark (terminal only)
  - Quagga (routing protocol daemons)

Starting the appliance in the lab VM may take some time.
Be patient!

VPCS

- Virtual PC emulator
- Emulates up to nine very basic DOS-like PCs
- Can be used to simulate workstations for IP networks
- Supports:
  - IPv4/ARP
  - IPv6/NDP
  - DHCP client
  - Route tracing

VPCS

IP Address Configuration

```
# Static IP
PC1> ip 192.168.0.1/24
PC1> ip fd98:6a35:c761::1

# DHCP
PC1> ip dhcp
```

```
# Show current IP configuration
PC1> show ip
...
PC1> show ipv6
...
```

Important commands

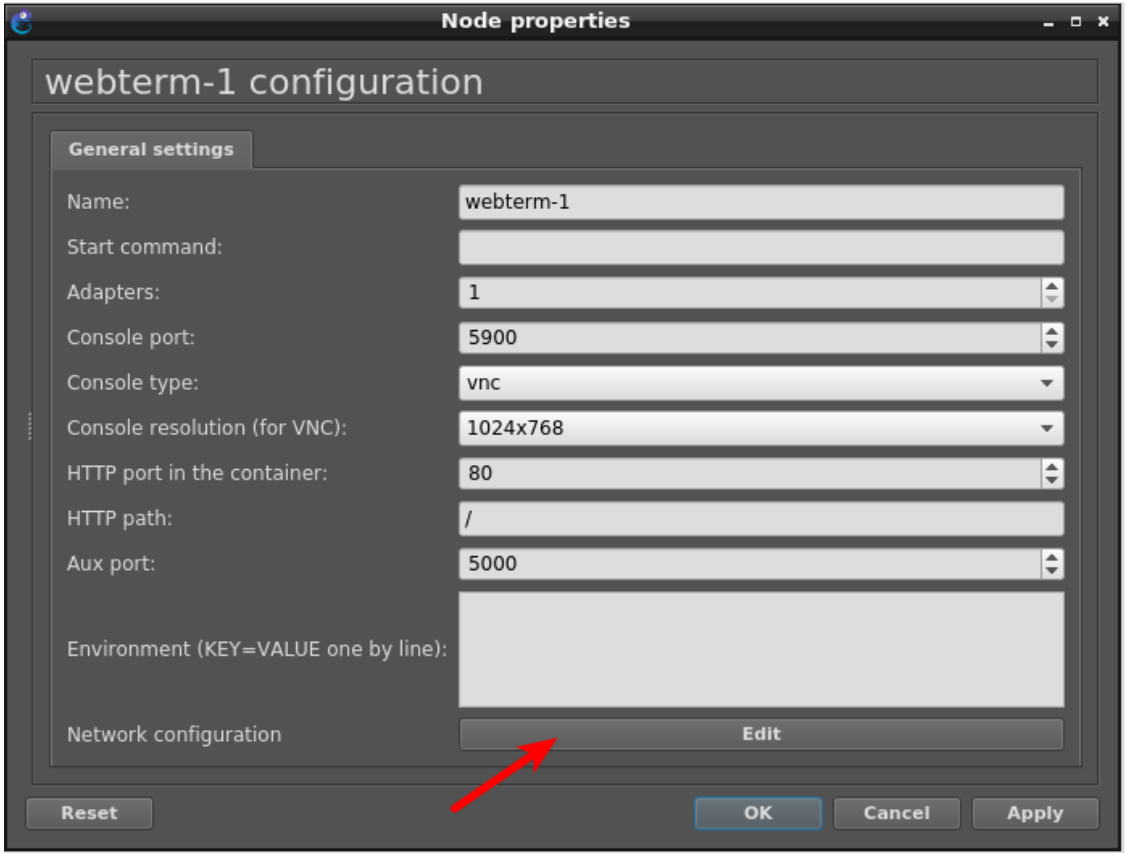
```
# Save current VPCS config (single host only)
PC1> save

# Get help
PC1> help
...
```

IP- / Webterm

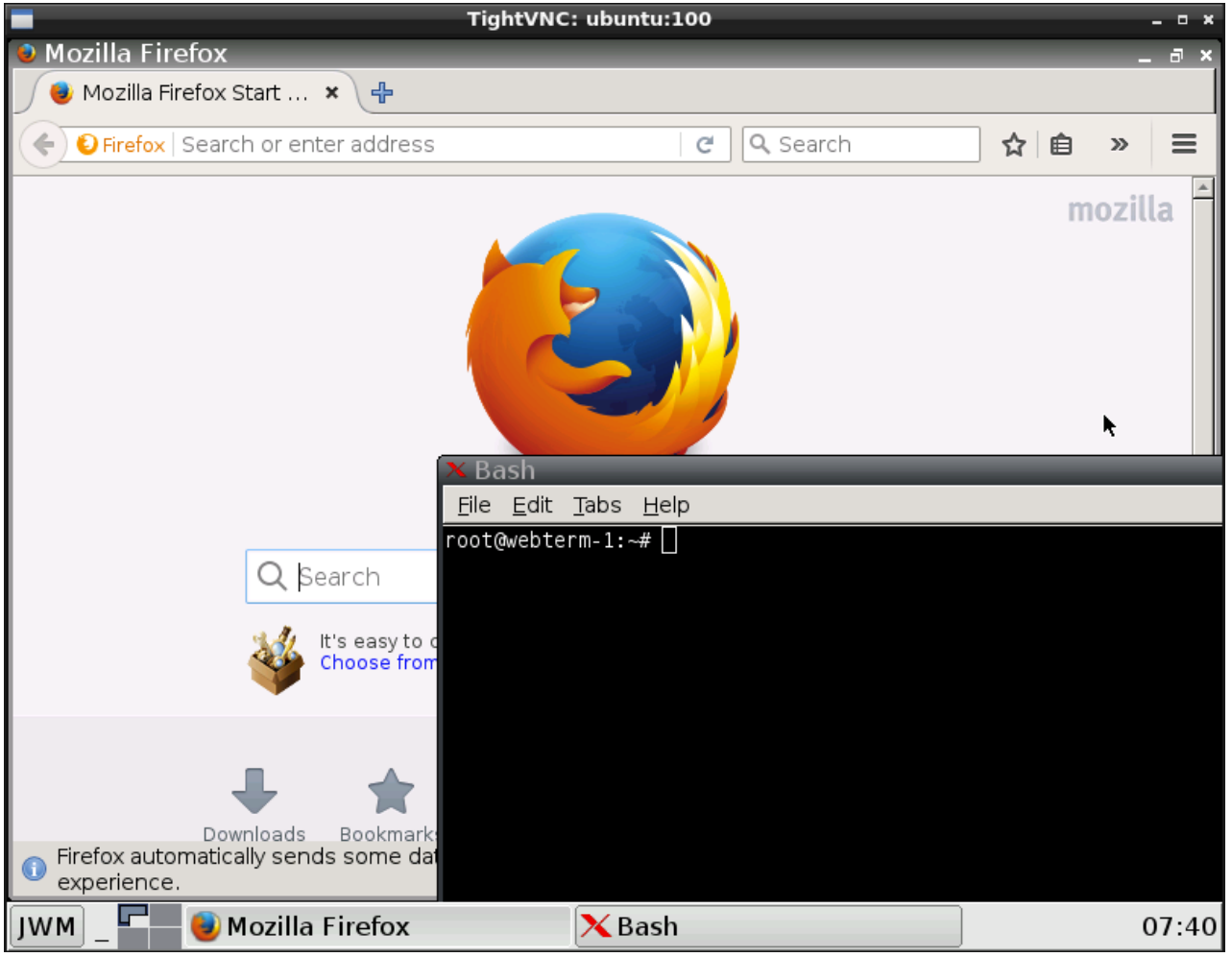
- Small appliance simulating a basic Linux desktop system
- Based on Debian 8
- Includes:
  - iproute2
  - ping
  - traceroute
  - SSH client
  - iperf3 (measures network throughput)
  - Firefox web browser ( )

IP- / Webterm | Network Configuration



Persistent network configuration in the settings dialog

Webterm

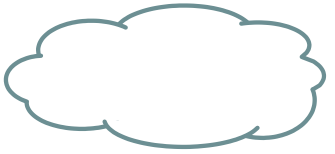

TightVNC: ubuntu:100
Mozilla Firefox
Mozilla Firefox Start ...
Firefox
Search or enter address
Search
mozilla
Bash
File Edit Tabs Help
root@webterm-1:~#
Search
Downloads
Bookmarks
Firefox automatically sends some dat
experience.
JWM
Mozilla Firefox
Bash
07:40

Building Bridges

Virtual networks built in GNS3 can be bridged to networks in the real world using the cloud node.

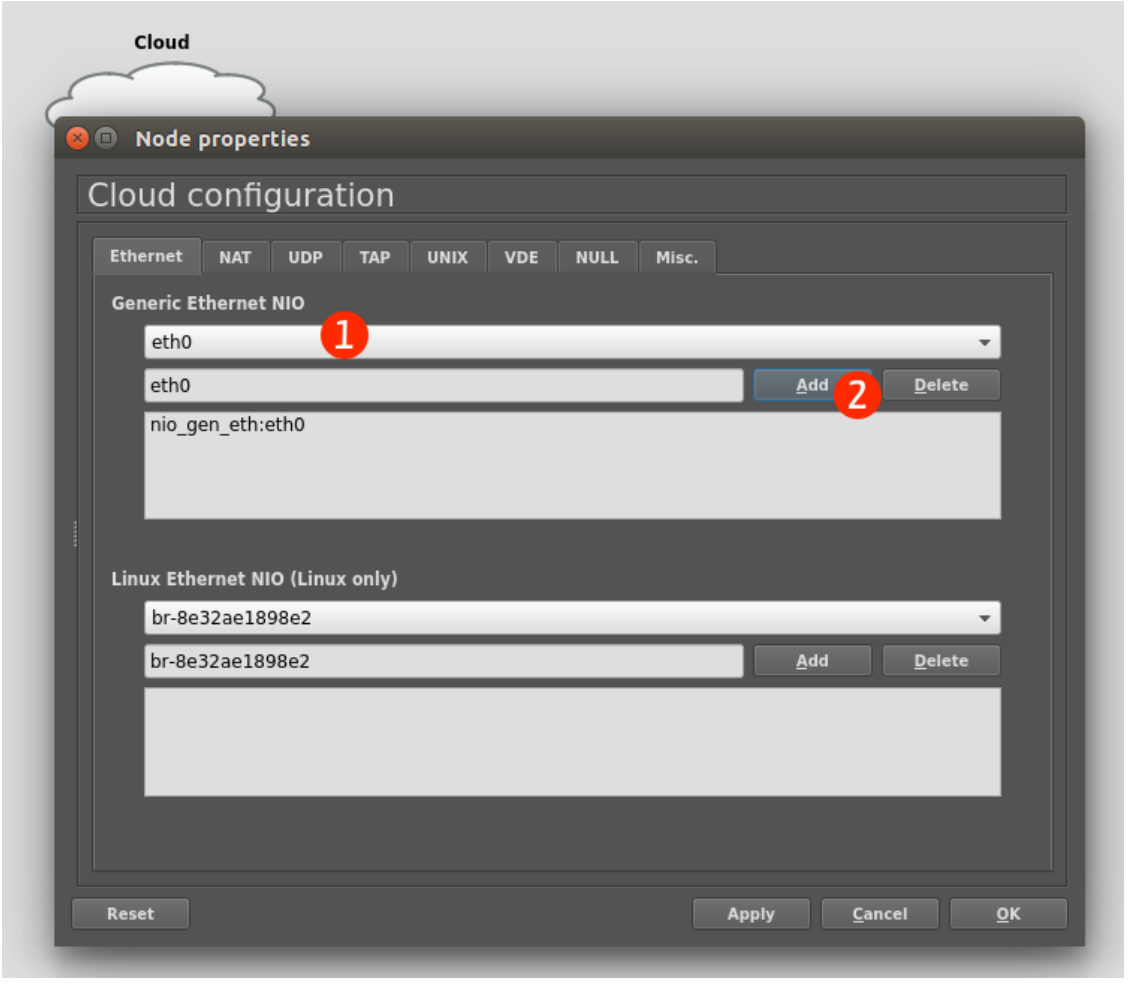
The cloud node can be bridged with any network interface

- Physical Interfaces
  (e.g. your main network interface or a spare one connected to an external lab network)
- Virtual Interfaces
  (e.g. VPN tunnels)



Most appliances cannot be connected directly to the cloud node, so use a hub or switch in between.

Building Bridges



Wrap-up

Questions?

Further Reading

- Wireshark Documentation
- GNS3 Documentation

- Chris Sanders: "Practical Packet Analysis"
- Jason C. Neumann: "The Book of GNS3"