

Projektarbeit II: SBB Live Monitor and remote Access

Verteilte Systeme II – Hands-on



Gruppe: Theophile Becker / Theo Metzger
Datum: 7. Januar 2026

Inhaltsverzeichnis

Dashboard Zugriff.....	3
Netzwerkprotokolle	4
1. ARP (Address Resolution Protocol).....	4
2. ICMP (Internet Control Message Protocol).....	5
3. DNS (Domain Name System)	6
4. SSH (Secure Shell)	7
5. HTTP / HTTPS (Dashboard-Aufruf)	8
Welche Teile Ihres Projektarbeit I - Dashboards wurden übernommen?	9
Welche Daten der Analyse sind sichtbar bzw. verschlüsselt und warum?.....	10
Warum funktionieren Hostnames ohne statische IPs zuverlässig?	11
Wie wurde Ihr Dienst in systemd integriert?	12

Dashboard Zugriff

Das Dashboard wird mittels avahi Service erreicht. Um Zugriff auf den Service zu bekommen muss der Service auf beiden Geräten aktiv sein. Folgender Befehl aktiviert ihn:

```
sudo systemctl enable --now avahi-daemon
```

Zusätzlich müssen beide Geräte im selben Netzwerk sein, damit sie sich finden können.

Im Browser kann die Website über «<http://vm1-server.local:5000/>» erreicht werden.

vm1-client:

Benutzername: halophile

Passwort: hallo

vm2-client:

Benutzername: Theo

Passwort: hallo1234

Netzwerkprotokolle

1. ARP (Address Resolution Protocol)

Wofür wird dieses Protokoll verwendet?

ARP wird verwendet, um innerhalb eines lokalen Netzwerks (LAN) eine IP-Adresse einer MAC-Adresse zuzuordnen.

Da die physische Übertragung über MAC-Adressen erfolgt, muss ein Gerät zuerst die MAC-Adresse des Zielgeräts kennen.

Welche Aktion erzeugt ARP-Netzverkehr?

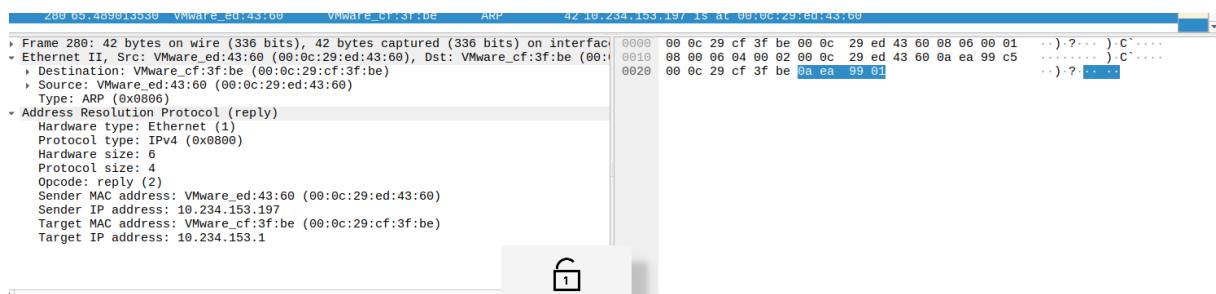
- Aufruf einer IP-Adresse im selben Netzwerk
- ping eines lokalen Geräts
- Erster Zugriff auf den Default Gateway
- Beispiel: Ein Client möchte ein Paket an 192.168.1.1 senden und fragt per ARP:
- „Wer hat diese IP?“

OSI-Schicht

Schicht 2 – Sicherungsschicht (Data Link Layer)

ARP arbeitet zwischen Layer 2 und 3, wird aber meist Layer 2 zugeordnet, da es MAC-Adressen verwendet.

Beispiel:



The screenshot shows a network capture in Wireshark. The selected frame is an ARP request (Frame 280). The details pane shows the following information:

- Frame 280: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface VMware_00:0c:29:ed:43:60
- Ethernet II, Src: VMware_ed:43:60 (00:0c:29:ed:43:60), Dst: VMware_cf:3f:be (00:0c:29:cf:3f:be)
- Destination: VMware_cf:3f:be (00:0c:29:cf:3f:be)
- Source: VMware_ed:43:60 (00:0c:29:ed:43:60)
- Type: ARP (0x0806)
- Address Resolution Protocol (reply)
 - Hardware type: Ethernet (1)
 - Protocol type: IPv4 (0x0800)
 - Hardware size: 6
 - Protocol size: 4
 - Opcode: reply (2)
 - Sender MAC address: VMware_ed:43:60 (00:0c:29:ed:43:60)
 - Sender IP address: 10.234.153.197
 - Target MAC address: VMware_cf:3f:be (00:0c:29:cf:3f:be)
 - Target IP address: 10.234.153.1

Es wird abgefragt, welche MAC Adresse zur IP 10.234.153.197 gehört.

2. ICMP (Internet Control Message Protocol)

Wofür wird dieses Protokoll verwendet?

ICMP dient zur Fehler- und Statusmeldung im Netzwerk.

Es wird z. B. genutzt, um zu prüfen, ob ein Ziel erreichbar ist.

Welche Aktion erzeugt ICMP-Netzverkehr?

- ping-Befehl (Echo Request / Echo Reply)
 - Traceroute
 - Fehlermeldungen wie Destination Unreachable oder Time Exceeded

OSI-Schicht

Schicht 3 – Vermittlungsschicht (Network Layer)

ICMP ist Teil des IP-Protokollstapels.

Beispiel:

 810 99.114326191.2 19.234.153.19/ 19.234.153.1	 811 99.11526191.0 19.234.153.1	 812 100.15623933.2 19.234.153.197	813 100.156193490.0 19.234.153.1	814 101.175884494.2 19.234.153.197	 815 101.176466612.0 19.234.153.1
 ICMP 98 Echo (ping) request id=0x1120, seq=3/256, ttl=64 (request in 811)	 ICMP 98 Echo (ping) reply id=0x1120, seq=1/256, ttl=64 (request in 810)	ICMP 98 Echo (ping) request id=0x1120, seq=2/256, ttl=64 (request in 813)	ICMP 98 Echo (ping) reply id=0x1120, seq=2/256, ttl=64 (request in 812)	 ICMP 98 Echo (ping) request id=0x1120, seq=3/768, ttl=64 (request in 815)	 ICMP 98 Echo (ping) reply id=0x1120, seq=3/768, ttl=64 (request in 814)

> Frame 815: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface	0000 00 0c 29 ed 43 60 00 00 0c 29 cf 3f be 08 00 45 00 ..) C` ..) ?..- E-
< Ethernet II, Src: VMware_cf:3f:be (00:0c:29:cf:3f:be), Dst: VMware_ed:43:60 (00:	0010 00 54 ef 57 00 00 40 01 b2 7f ea 99 01 00 ea
> Destination: VMware_ed:43:60 (00:0c:29:ed:43:60)	0020 99 c5 00 f0 c3 81 10 00 03 b8 5c 69 00 00
> Source: VMware_cf:3f:be (00:0c:29:cf:3f:be)	0030 00 00 20 4b 02 00 00 00 00 10 11 12 13 14 15
Type: IPv4 (0x0800)	0040 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25
> Internet Protocol Version 4, Src: 10.234.153.1, Dst: 10.234.153.197	0050 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 ..!#%\$
> Internet Control Message Protocol	0060 36 37 67

Es wurde ein Ping ausgeführt von 10.234.153.197 zu 10.234.153.1

3. DNS (Domain Name System)

Wofür wird dieses Protokoll verwendet?

DNS übersetzt Domainnamen in IP-Adressen (z. B. www.google.de → 142.250.x.x).

Ohne DNS müsste der User IP-Adressen manuell eingeben.

Welche Aktion erzeugt DNS-Netzverkehr?

- Aufruf einer Webseite im Browser
- Verbindungsauftbau zu einem Server per Namen
- Systemstart (automatische DNS-Anfragen)

OSI-Schicht

Schicht 7 – Anwendungsschicht (Application Layer)

DNS arbeitet meist über UDP Port 53, teilweise über TCP.

Beispiel:

Frame	Start	End	Length	Type	Source MAC	Destination MAC	Source IP	Destination IP	Protocol	Flags	Information
41	6:507/039910	10:234.153.1	14:234.153.202	DNS	81 Standard query 0xb73a A transport.opendata.ch						
42	6.567670183	10.234.153.1	10.234.153.202	DNS	81 Standard query 0xf001 AAAA transport.opendata.ch						
43	6.522955232	10.234.153.202	10.234.153.1	DNS	81 Standard query response 0xf001 AAAA transport.opendata.ch						
44	6.538764042	10.234.153.202	10.234.153.1	DNS	97 Standard query response 0xb73a A transport.opendata.ch A 5.148.188.59						


```
Frame 41: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface
Ethernet II, Src: VMware_cf:3f:be (00:0c:29:c3:f3:be), Dst: f2:cd:31:65:67:56 (f2:cd:31:65:67:56)
  Destination: f2:cd:31:65:67:56 (f2:cd:31:65:67:56)
  Source: VMware_cf:3f:be (00:0c:29:c3:f3:be)
  Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 10.234.153.1, Dst: 10.234.153.202
User Datagram Protocol, Src Port: 46173, Dst Port: 53
Domain Name System (query)

0000 f2 cd 31 65 67 56 00 0c 29 cf 3f be 08 00 45 00 .:1egV.. )?..E.
0010 00 43 85 59 00 00 40 11 ac b1 0a ea 99 01 0a ea ..C.Y. @. .....
0020 99 ca b4 5d 00 35 00 2f 6d dd b7 3a 01 00 00 01 ..]5 / m. .....
0030 00 00 00 00 00 00 09 74 72 61 6e 73 70 6f 72 74 .....t ransport
0040 08 6f 70 65 6e 64 61 74 61 02 63 68 00 00 01 00 .opendat a.ch...
0050 01 .
```

Es wird nach der IP von transport.opendata.ch gefragt. Die IP ist 5.148.188.59

4. SSH (Secure Shell)

Wofür wird dieses Protokoll verwendet?

SSH ermöglicht eine verschlüsselte Remote-Verbindung zu einem anderen Rechner, meist zur Administration von Servern.

Welche Aktion erzeugt SSH-Netzverkehr?

- Aufbau einer SSH-Verbindung (ssh user@server)
- Anmeldung auf einem Linux-Server
- Ausführen von Befehlen über eine Remote-Shell

OSI-Schicht

Schicht 7 – Anwendungsschicht (Application Layer)

SSH nutzt TCP Port 22 und verschlüsselt den gesamten Datenverkehr.

Beispiel:

No.	Time	Source	Destination	Protocol	Length	Info
743	196.772155212	10.234.153.197	10.234.153.224	SSHv2	97	Server: Protocol (SSH-2.0-OpenSSH_9.6p1 Ubuntu-Subuntu13.14)
744	196.786394646	10.234.153.224	10.234.153.197	SSHv2	87	Client: Protocol (SSH-2.0-OpenSSH_for_Windows_9.5)
746	196.788413823	10.234.153.197	10.234.153.224	SSHv2	1174	Server: Key Exchange Init
747	196.797977487	10.234.153.224	10.234.153.197	SSHv2	1486	Client: Key Exchange Init
749	196.838649569	10.234.153.224	10.234.153.197	SSHv2	102	Client: Elliptic Curve Diffie-Hellman Key Exchange Init
751	196.842135917	10.234.153.197	10.234.153.224	SSHv2	546	Server: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Keys
766	199.265149864	10.234.153.224	10.234.153.197	SSHv2	70	Client: New Keys
68	199.386146889	10.234.153.224	10.234.153.197	SSHv2	98	Client:
776	199.386252810	10.234.153.197	10.234.153.224	SSHv2	98	Server:
771	199.386597286	10.234.153.224	10.234.153.197	SSHv2	114	Client:
772	199.397144681	10.234.153.197	10.234.153.224	SSHv2	106	Server:
773	199.397379351	10.234.153.224	10.234.153.197	SSHv2	554	Client:
774	199.316793928	10.234.153.197	10.234.153.224	SSHv2	106	Server:
786	202.363314713	10.234.153.224	10.234.153.197	SSHv2	202	Client:
787	202.489265607	10.234.153.197	10.234.153.224	SSHv2	82	Server:
788	202.482544251	10.234.153.224	10.234.153.197	SSHv2	166	Client:
790	202.454009474	10.234.153.197	10.234.153.224	SSHv2	790	Server:
791	202.458125438	10.234.153.224	10.234.153.197	SSHv2	634	Client:

SSH Verbindung zwischen Windows und Ubuntu wurde aufgebaut

- Client sendet seinen ECDH Public Key
- Server antwortet mit:
 - seinem ECDH Public Key
 - seinem Host Key (ssh-ed25519) → Server-Identität
 - einer Signatur, die beweist:

„Ich bin wirklich dieser Server“

Ergebnis

- Beide berechnen denselben geheimen Sitzungsschlüssel
- Der Schlüssel wird nie über das Netzwerk gesendet

5. HTTP / HTTPS (Dashboard-Aufruf)

Wofür wird dieses Protokoll verwendet?

HTTP bzw. HTTPS dient zur Übertragung von Webseiten und Web-Inhalten zwischen Client (Browser) und Webserver.

HTTPS ist die verschlüsselte Variante von HTTP.

Welche Aktion erzeugt HTTP(S)-Netzverkehr?

- Aufruf eines Dashboards im Webbrowser
- Laden von Webseiten, Bildern, CSS, JavaScript
- Senden von Formularen oder Login-Daten

OSI-Schicht

Schicht 7 – Anwendungsschicht (Application Layer)

HTTP: TCP Port 80

HTTPS: TCP Port 443 (zusätzlich TLS-Verschlüsselung)

Beispiel:

1330 592.622472546 10.234.153.1	10.234.153.197	HTTP	1386 HTTP/1.1 200 OK (text/html)
1331 592.622479476 10.234.153.197	10.234.153.1	TCP	66 38544 → 5000 [ACK] Seq=350 Ack=10185
1332 592.622590319 10.234.153.197	10.234.153.1	TCP	66 38544 → 5000 [FIN, ACK] Seq=350 Ack=10186
1333 592.622921018 10.234.153.1	10.234.153.197	TCP	66 5000 → 38544 [FIN, ACK] Seq=10185 Ack=351
1334 592.622929663 10.234.153.197	10.234.153.1	TCP	66 38544 → 5000 [ACK] Seq=351 Ack=10186

► Ethernet II, Src: VMware_cf:3f:be (00:0c:29:c3:f3:be), Dst: 10.234.153.1 (08:00:27:00:00:01)	0000 48 54 54 50 2f 31 2e 31
► Internet Protocol Version 4, Src: 10.234.153.1, Dst: 10.234.153.197	0010 0a 53 65 72 76 65 72 3a
► Transmission Control Protocol, Src Port: 5000, Dst Port: 38544	0020 67 2f 33 2e 31 2e 34 20
► [4] Reassembled TCP Segments (10184 bytes): #1324(176), #1325(176)	0030 2e 31 33 2e 37 0d 0a 44
► Hypertext Transfer Protocol	0040 61 74 65 3a 20 57 65 64
► HTTP/1.1 200 OK\r\n	0050 2c 20 30 37 20 4a 61 6e
Server: Werkzeug/3.1.4 Python/3.13.7\r\nDate: Wed, 07 Jan 2026 15:42:28 GMT\r\nContent-Type: text/html; charset=utf-8\r\nContent-Length: 10008\r\nConnection: close\r\n\r\n[HTTP response 1/1]	0050 20 32 30 32 36 20 31 35
[Time since request: 0.001694428 seconds]	0060 4d 54 0d 0a 43 6f 6e 74
[Request in frame: 1322]	0070 65 6e 74 2d 54 79 70 65
[Request URI: http://vm1-server.local:5000/]	0080 3a 20 74 65 78 74 2f 68
File Data: 10008 bytes	0090 74 6d 6c 3b 20 63 68 61
Incap. based text data: text/html (286 lines)	00a0 2d 38 0d 0a 43 6f 6e 74

Inhalte des Dashboards werden aktualisiert über den URL <http://vm1-server.local:5000/>

Das Dashboard läuft bewusst über HTTP, da kein TLS-Zertifikat eingerichtet wurde.

Welche Teile Ihres Projektarbeit I - Dashboards wurden übernommen?

Es wurde der gesamte Teil der Projektarbeit I übernommen.

Welche Anpassungen waren technisch notwendig?

Folgende Services mussten angepasst bzw. neu erstellt werden:

- sbb-collector.service
- sbb-dashboard.service
- firefox-dashboard.desktop

SQL Schema

Alle Daten werden in einer Datenbank gespeichert mittels SQLite. Die Datei befindet sich unter «~/dashboard/sbb-monitor/Code». Sie besteht aus 2 Tabellen:

- stations (alle Bahnhöfe)

Spalte	Typ	Bedeutung
id	TEXT	Primärschlüssel, eindeutige Stations-ID (z. B. SBB-ID)
name	TEXT	Name der Station
x	REAL	Koordinate
y	REAL	Koordinate

- departures (alle Abfahrten)

• Spalte	• Typ	• Pflicht	• Bedeutung
• id	• INTEGER	• <input checked="" type="checkbox"/>	• Primärschlüssel, auto-increment
• station_id	• TEXT	• <input checked="" type="checkbox"/>	• Verweis auf stations.id
• departure_ts	• INTEGER	• <input checked="" type="checkbox"/>	• Abfahrtszeit als Unix-Timestamp
• destination	• TEXT	• <input type="checkbox"/>	• Zielbahnhof
• category	• TEXT	• <input type="checkbox"/>	• Zugtyp (z. B. IC, IR, S, RE)
• train_nr	• TEXT	• <input type="checkbox"/>	• Zugnummer
• delay	• INTEGER	• <input type="checkbox"/>	• Verspätung in Minuten

Beziehung:

stations.id ←— departures.station_id

Eine Station hat viele Abfahrten 1:n

Welche Daten der Analyse sind sichtbar bzw. verschlüsselt und warum?

Sichtbare Daten (unverschlüsselt):

- **ARP-Pakete**
Enthalten IP- und MAC-Adressen, da ARP keine Verschlüsselung unterstützt.
- **DNS-Anfragen**
Domainnamen und Ziel-IP-Adressen sind sichtbar, da DNS standardmäßig unverschlüsselt über UDP erfolgt.
- **HTTP-Daten**
Inhalte wie URLs, HTML-Struktur und Text sind einsehbar, da HTTP keine Verschlüsselung nutzt.

Verschlüsselte Daten:

- **SSH-Verkehr**
Der komplette SSH-Datenverkehr ist verschlüsselt. Inhalte wie Befehle, Passwörter und Ausgaben sind nicht sichtbar.
- **Warum Verschlüsselung?**
SSH verwendet moderne Kryptographie (z. B. ECDH), um Vertraulichkeit und Integrität zu gewährleisten und Man-in-the-Middle-Angriffe zu verhindern.

Warum funktionieren Hostnames ohne statische IPs zuverlässig?

Die Erreichbarkeit des Dashboards erfolgt über den Hostnamen:

vm1-server.local

Dieser Hostname wird über mDNS (Multicast DNS) bereitgestellt, welches Teil des Avahi-Dienstes ist.

Funktionsweise:

- Der Server kündigt seinen Namen im lokalen Netzwerk an
- Clients fragen per Multicast:
„Wer ist vm1-server.local?“
- Der Server antwortet mit seiner aktuellen IP-Adresse

Vorteile:

- Keine statische IP notwendig
- Automatische Namensauflösung
- Ideal für lokale Netzwerke und dynamische Umgebungen
- Auch wenn sich die IP-Adresse ändert, bleibt der Hostname erreichbar, solange sich alle Geräte im selben Netzwerk befinden.

Wie wurde Ihr Dienst in systemd integriert?

Der Betrieb des Dashboards erfolgt über systemd-Services, um einen stabilen und automatischen Start sicherzustellen.

Verwendete Services:

sbb-collector.service (vm1)

- Ruft periodisch Daten von der API transport.opendata.ch ab
- Speichert die Daten in der SQLite-Datenbank
- Läuft automatisch im Hintergrund

sbb-dashboard.service (vm1)

- Startet den Webserver für das Dashboard
- Stellt die Weboberfläche für Clients bereit
- Wird beim Systemstart automatisch gestartet

firefox-dashboard.desktop (vm2)

- startet Firefox mit der richtigen URL, um die Website darzustellen

Vorteile von systemd:

- Automatischer Start beim Booten
- Überwachung und Neustart bei Fehlern
- Zentrale Verwaltung über systemctl

Beispiel:

```
sudo systemctl enable --now sbb-dashboard.service
```

- Dadurch ist sichergestellt, dass das Dashboard jederzeit verfügbar ist, ohne manuelle Eingriffe.