sparkSOME

PRELEKCJA · MBUM #11 · POZNAŃ · 2 PAŹDZIERNIKA 2026

# Wireshark kontra rzeczywistość

## Analiza najczęstszych problemów spotykanych w sieciach MikroTik

**Tomasz Siroń · Board Member & Infrastructure Architect, SparkSome Venture Sp. z o.o.**

RouterOS 7.24 stable / 7.23 LTS · Wireshark 4.6

sparkSOME

# Dlaczego ten temat?

Tomasz Siroń w IT od 2010 roku

### Czym się zajmuję na co dzień?

- Projektowanie i utrzymanie infrastruktury dla firm
- Trener prowadzący szkolenia z zakresu infrastruktury i systemów IT
- Sieci, Linux, wirtualizacja, DevOps, automatyzacja
- Bezpieczeństwo infrastruktury IT
- Budowanie biznesu

### Do czego używam MikroTika?

- Routing, OSPF, VLAN, VPN, firewall
- Prosta diagnostyka problemów sieciowych
- Warstwa sieciowa spięta z Proxmoxem, Ansible, Dockerem, Kubernetesem, Zabbixem, NGINX-em i CI/CD
- Posiadane certyfiaty: MTCNA, MTCRE, MTCINE, MTCSWE, MTCIPv6E
- Cena ;)

### Dodatkowo

- Autor Sterowanie24.pl, medialnego projektu łączącego elektronik IoT ze zdalnym sterowaniem urządzeniami (oświetleniem świątecznym) przez Internet
- Technologiczny generalista, łączący sieci, elektronikę, AI, SDR, analizę obrazu i fizykę w interdyscyplinarne eksperymenty ze światem fizycznym

**Dzisiejsze przykłady pochodzą z realnych wdrożeń, przy tworzeniu prezentacji nie ucierpiał żaden Mikrotik.**

sparkSOME

# SparkSome Venture

Firma technologiczna projektująca, wdrażająca i utrzymująca infrastrukturę IT dla firm

1

### Infrastruktura

Sieci komputerowe, systemy Linux/Windows, wirtualizacja i klastry wysokiej dostępności

2

### DevOps i bezpieczeństwo

Automatyzacja, monitoring, cyberbezpieczeństwo

3

### Utrzymanie

Outsourcing infrastruktury, projekty punktowe i długoterminowe wsparcie środowisk klientów

**Open Source + nieszablonowe projekty + automatyzacja, z naciskiem na stabilność, bezpieczeństwo i skalowalność środowiska.**

sparkSOME

# Czego nie daje nam narzędzie?

## Narzędzie nie daje diagnozy.

Wireshark lub tcpdump pokazuje pakiety, nie przedstawi natomiast diagnozy problemu.

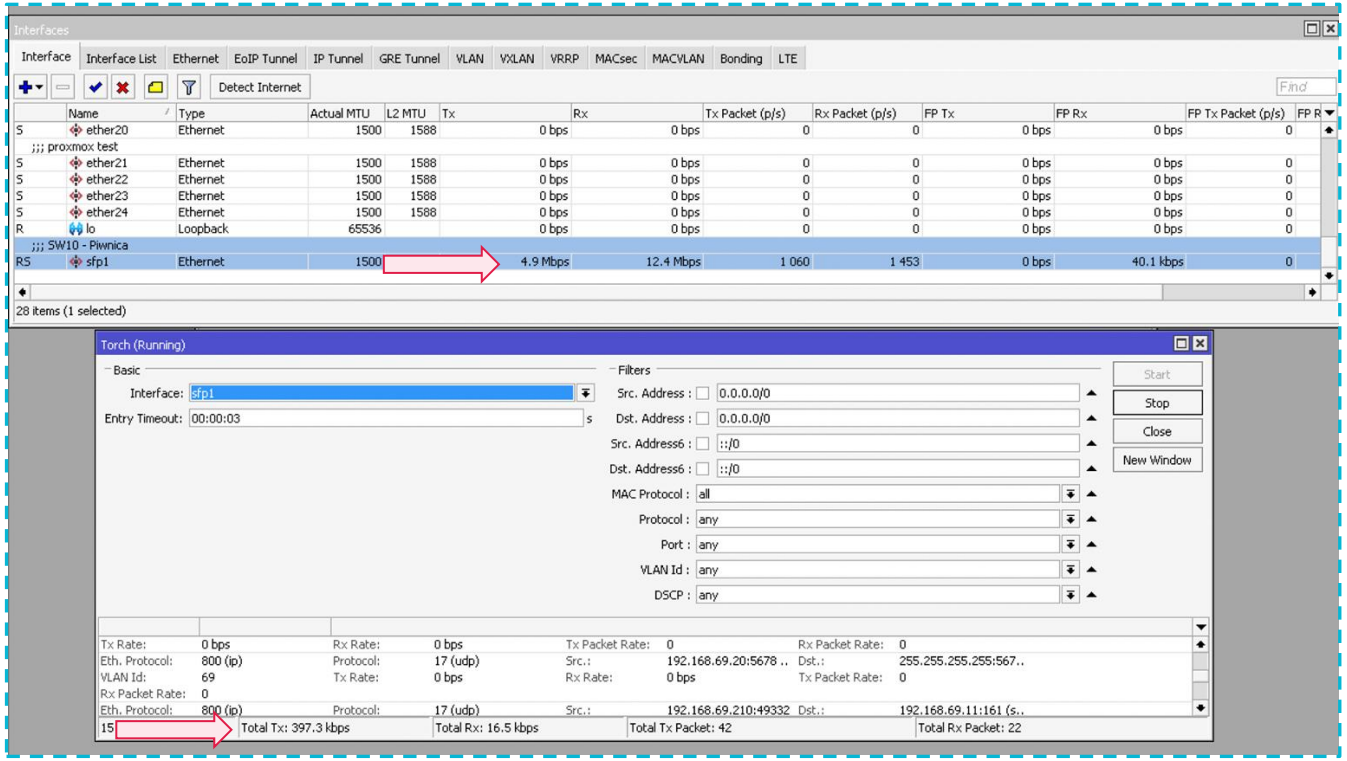
**Jak analizować przypadki?**

**objaw ze zgłoszenia** → **co mówi log urządzenia** → **co mówi pakiet** → **analiza i puenta**

sparkSOME

# Torch, który kłamie?

Sniffer włączony, interfejs przepycha kilka Mbps




**Pytanie do uczestników**

**Sniffer włączony, interfejs przepycha ruch.**

**Dlaczego capture jest prawie pusty?**

sparkSOME

# Dlaczego capture jest pusty

Torch i sniffer widzą wyłącznie ruch, który dotyka CPU

### Hardware offloading

- Ruch offloadowany na switch chip nigdy nie trafia do CPU
- Stanowisko MikroTika przy HW offloadingu pakiety przetwarza tylko switch chip
- Do wglądu potrzebny port mirroring albo copy-to-cpu

### Punkt obserwacji ma znaczenie

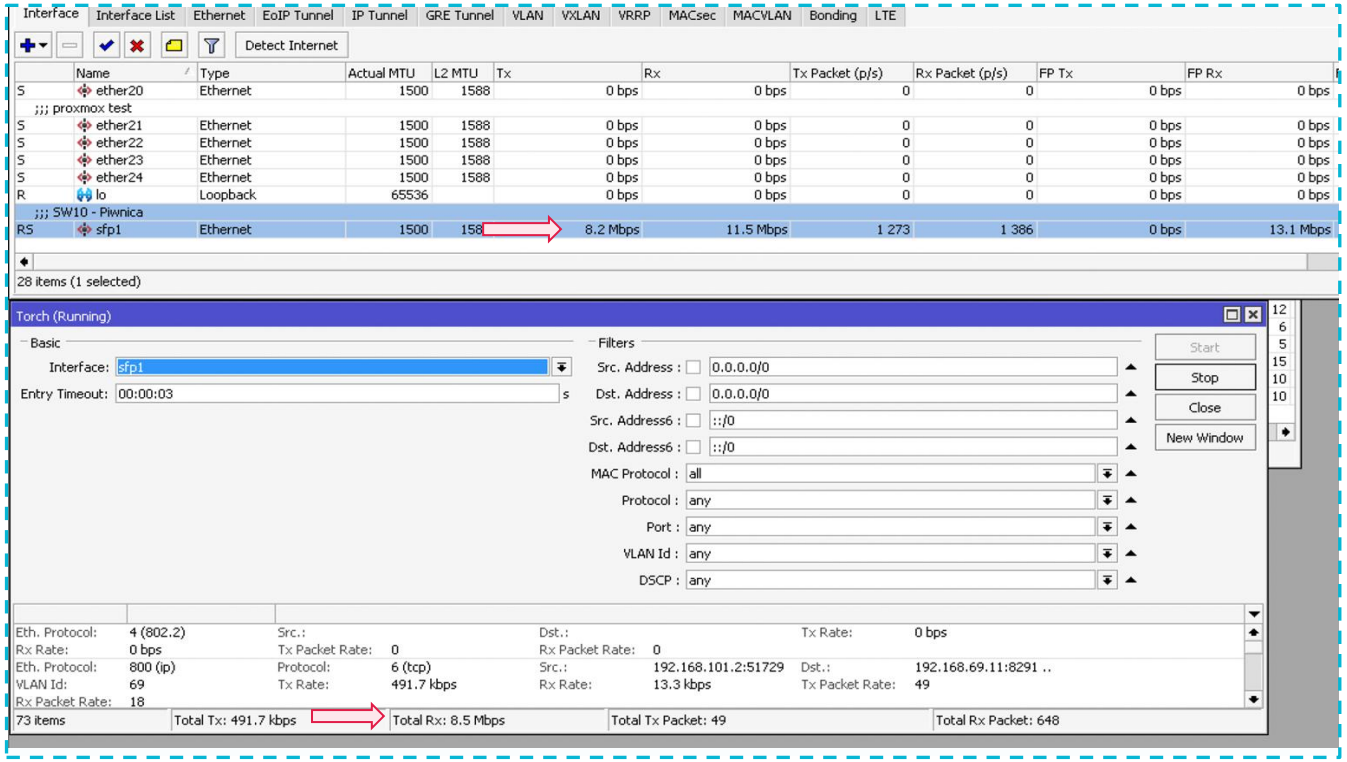
- Mirror wysyła kopię pakietu do CPU
- Dlatego ruch z fizycznego TX może pojawić się w Torch jako RX
- Liczniki portu i Torch mogą więc pokazywać inne kierunki i wartości

```
/interface ethernet switch set ingress-mirror0=switch1-cpu egress-mirror0=switch1-cpu
/interface ethernet switch port set [find name=sfp1] ingress-mirror-to=mirror0 egress-mirror-to=mirror0
```

sparkSOME

# Mirror do CPU

Po skopiowaniu ruchu z sfp1 do CPU Torch zaczyna widzieć to, co wcześniej omijało procesor




**Uwaga**

**Zmirrorowany TX portu może pojawić się jako RX w Torch, bo CPU odbiera kopię pakietu.**

**Na CRS125/QCA-8513L jednoczesny mirror ingress+egress ma znane bugi, dlatego w przykładzie Torch pokazuje tylko jeden kierunek mimo poprawnej konfiguracji.**

sparkSOME

# Gdzie stanąć z przechwytywaniem pakietów w RouterOS

Kilka praktycznych opcji, zależnie od tego czego szukamy

1

### Sniffer do pliku

Dobre do dłuższego capture i analizy później

**Pułapka: uważaj na flash i RAM, najlepiej użyć limitu pliku**

2

### Streaming TZSP → Wireshark

Najwygodniejsze do analizy na żywo, pakiety oglądasz od razu na swoim komputerze

**Pułapka: timestampy pochodzą z Twojego PC**

3

### Kontener z tcpdumpem

Daje dużo swobody ale wymaga sprzęt obsługującego Containers z zapasem RAM

**Pułapka: kontener nie widzi bezpośrednio fizycznego etherX**

```
# capture do pliku z limitem rozmiaru
/tool/sniffer set file-name=capture.pcap file-limit=5000 filter-interface=ether1
# streaming na stację roboczą (Wireshark: capture filter udp port 37008)
/tool/sniffer set streaming-enabled=yes streaming-server=10.10.30.5 filter-stream=yes
/tool sniffer start
```

sparkSOME

# TCPdump czyli capture z linii komend

Zastosowanie: serwery bez GUI, kontenery, zdalny host po SSH

### Podobny rdzeni i składnia: Wireshark ~ tcpdump

- Oba narzędzia stoją na libpcap* i filtrach BPF
- Capture filter w Wiresharku = filtr w tcpdumpie
- -w plik.pcap → analiza lokalnie w Wiresharku
- -s0: historyczny snaplen ucinał payload; dziś domyślnie pełny pakiet
- Na RouterOS  w kontenerze możemy uruchomić pełnego tcpdumpa, którego sniffer nie ma

**wyjątek Windows (Npcap)*

```
tcpdump -i eth0 -n port 443

tcpdump -i eth0 -nn -w capture.pcap host 203.0.113.10

tcpdump -i eth0 -A port 80

tcpdump -i eth0 -nn 'tcp[tcpflags] & (tcp-syn|tcp-fin) != 0'
```

**Tcpdump nie jest gorszy od Wireshark. Tcpdump to narzędzie do zebrania dowodu często z myślą o analizie gdzie indziej.**

sparkSOME

# Capture filter vs display filter

Czym się różnią

## Capture filter - składnia BPF (Berkeley Packet Filter, język filtrowania pakietów)

- Filtruje zanim pakiet trafi do bufora
- Nie da się zmienić bez ponownego złapania ruchu
- Ten sam filtr działa w tcpdumpie

```
tcp port 443
```

## Display filter - składnia Wiresharka

- Filtruje po złapaniu ruchu
- Można zmieniać bez końca, na tym samym pliku
- Dostęp do pól, które w BPF nie są dostępne

```
tcp.port == 443 && ip.addr == 203.0.113.10
```

**Najczęstszy błąd: składnia display filter wpisana w pole capture filter.**

sparkSOME

# Rodzina tcp.analysis.*

Co Wireshark właściwie wykrywa, zanim znaczy to na czerwono

| | |
|---|---|
| `tcp.analysis.retransmission` | ten sam segment wysłany drugi raz |
| `tcp.analysis.fast_retransmission` | wireshark podejrzewa szybką retransmisję na podstawie ACK, numerów sekwencyjnych i czasu |
| `tcp.analysis.spurious_retransmission` | retransmisja danych, które według wcześniejszego ACK zostały już odebrane |
| `tcp.analysis.out_of_order` | segmenty przyszły w złej kolejności |
| `tcp.analysis.lost_segment` | Wireshark wnioskuje o dziurze w numerach sekwencyjnych |

**Ten sam czerwony napis znaczy co innego w zależności od punktu capture**

sparkSOME

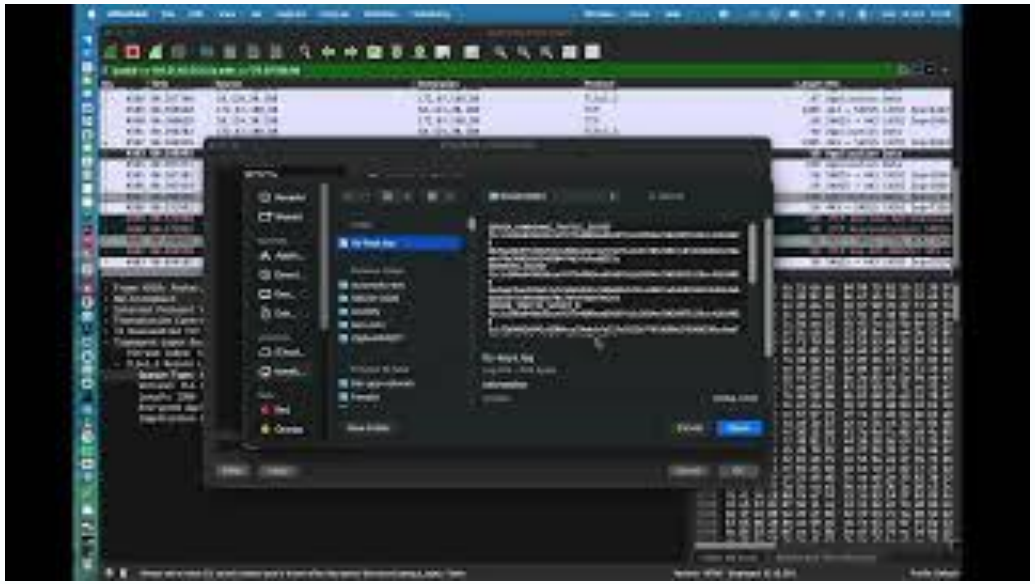
# Rozszywanie TLS na przykładzie HTTPS

curl musi korzystać z backendu TLS obsługującego SSLKEYLOGFILE, np. OpenSSL

1. **Ustaw zmienną w środowisku klienta**
2. **Odtwórz ruch i złap capture**
3. **Wskaż plik w Preferences → Protocols → TLS**

Po ponownym załadowaniu capture pakiety TLS dekodują się jako HTTP/2 albo HTTP,  widać nagłówki, cookies, treść żądania.

```
export SSLKEYLOGFILE=~/tls-keys.log
curl -v https://example.com
```



**Zanim capture powie cokolwiek o warstwie 7, musisz umieć złożyć strumień i (jeśli to Twój ruch) zdjąć z niego szyfrowanie.**

sparkSOME

# 1) MTU blackhole

Gdy ping działa poprawnie, TCP się zestawia a większy transfer danych staje

**Kontekst: LAN-y spięte tunelem EoIP. Tunel działa, ping przechodzi, ale część stron WWW nie wczytuje się poprawnie.**

### Objaw

## Co widzi użytkownik

- część stron działa normalnie
- większe odpowiedzi HTTPS nie przechodzą
- monitoring nie wskazuje problemów

### Co pokazuje Wireshark?

- handshake TCP poprawny
- małe segmenty przechodzą
- większe segmenty są retransmitowane
- brak skutecznej informacji o PMTU

### Jak to sprawdzić

```
icmp.type == 3 &&
icmp.code == 4
```

```
/ping 8.8.8.8 size=1472
  do-not-fragment
```

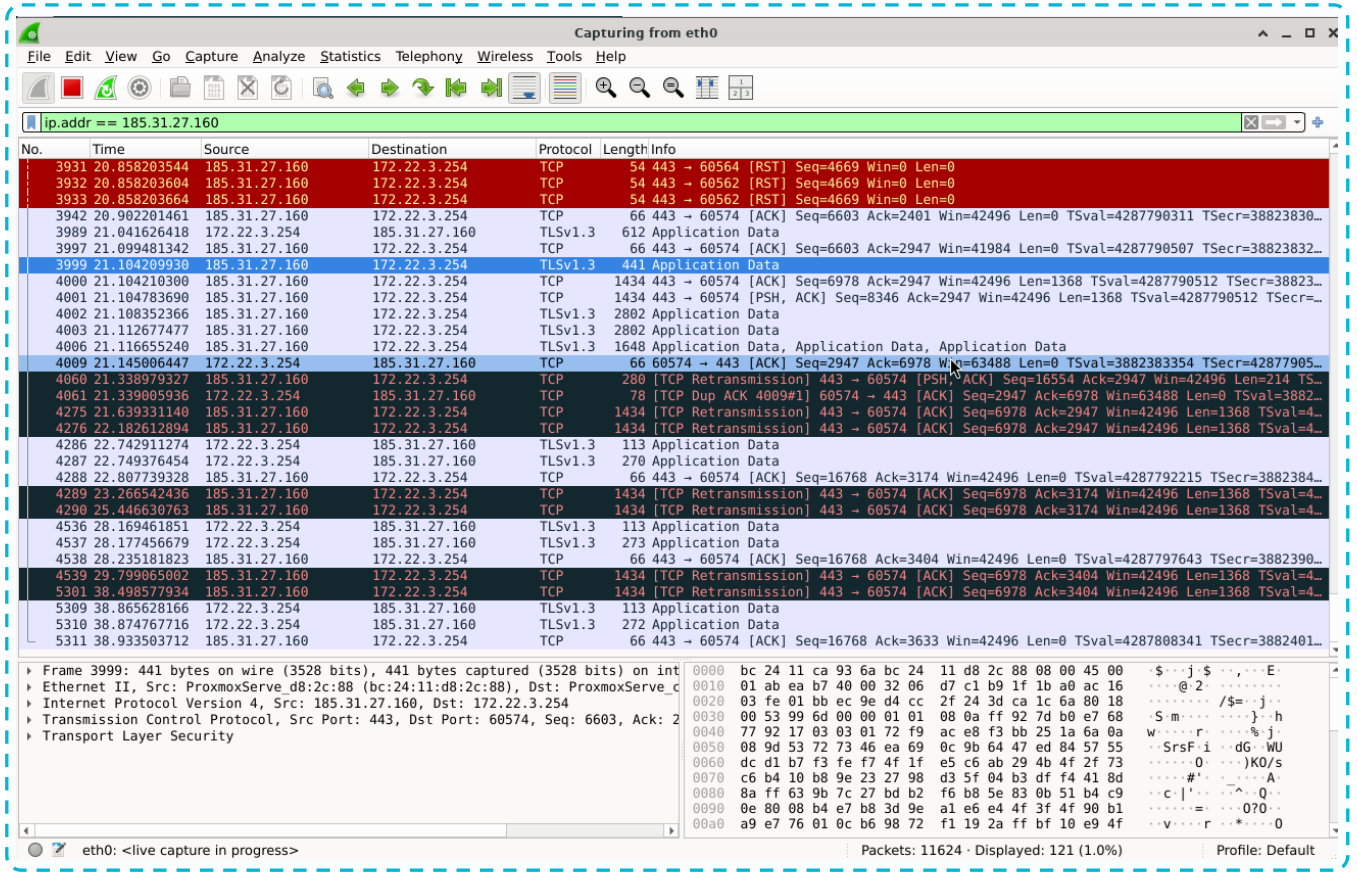
sparkSOME

# MTU blackhole

Przykłady

```
[tomek10861@lab-client-gui:~$ ping -M do -s 1472 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 1472(1500) bytes of data.
From 172.22.3.1 icmp_seq=1 Frag needed and DF set (mtu = 1420)
ping: sendmsg: Message too long
ping: sendmsg: Message too long
ping: sendmsg: Message too long
ping: sendmsg: Message too long
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 0 received, +5 errors, 100% packet loss, time 4103ms

[tomek10861@lab-client-gui:~$ ping -M do -s 1000 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 1000(1028) bytes of data.
1008 bytes from 8.8.8.8: icmp_seq=1 ttl=111 time=35.7 ms
1008 bytes from 8.8.8.8: icmp_seq=2 ttl=111 time=35.1 ms
1008 bytes from 8.8.8.8: icmp_seq=3 ttl=111 time=38.5 ms
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 35.094/36.445/38.538/1.500 ms
tomek10861@lab-client-gui:~$
```



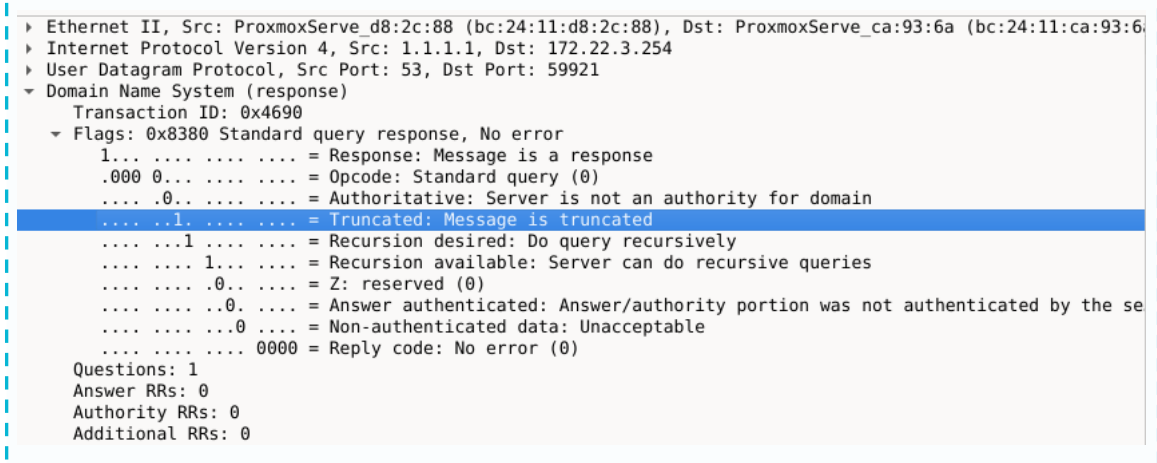
sparkSOME

# 2) DNS czyli UDP 53

Pewna sytuacja z nadgorliwym administratorem

## Co mówi pakiet

- Resolver odpowiedział poprawnie przez UDP
- Odpowiedź jest za duża, ustawiono TC=1
- Klient powinien ponowić zapytanie przez TCP/53

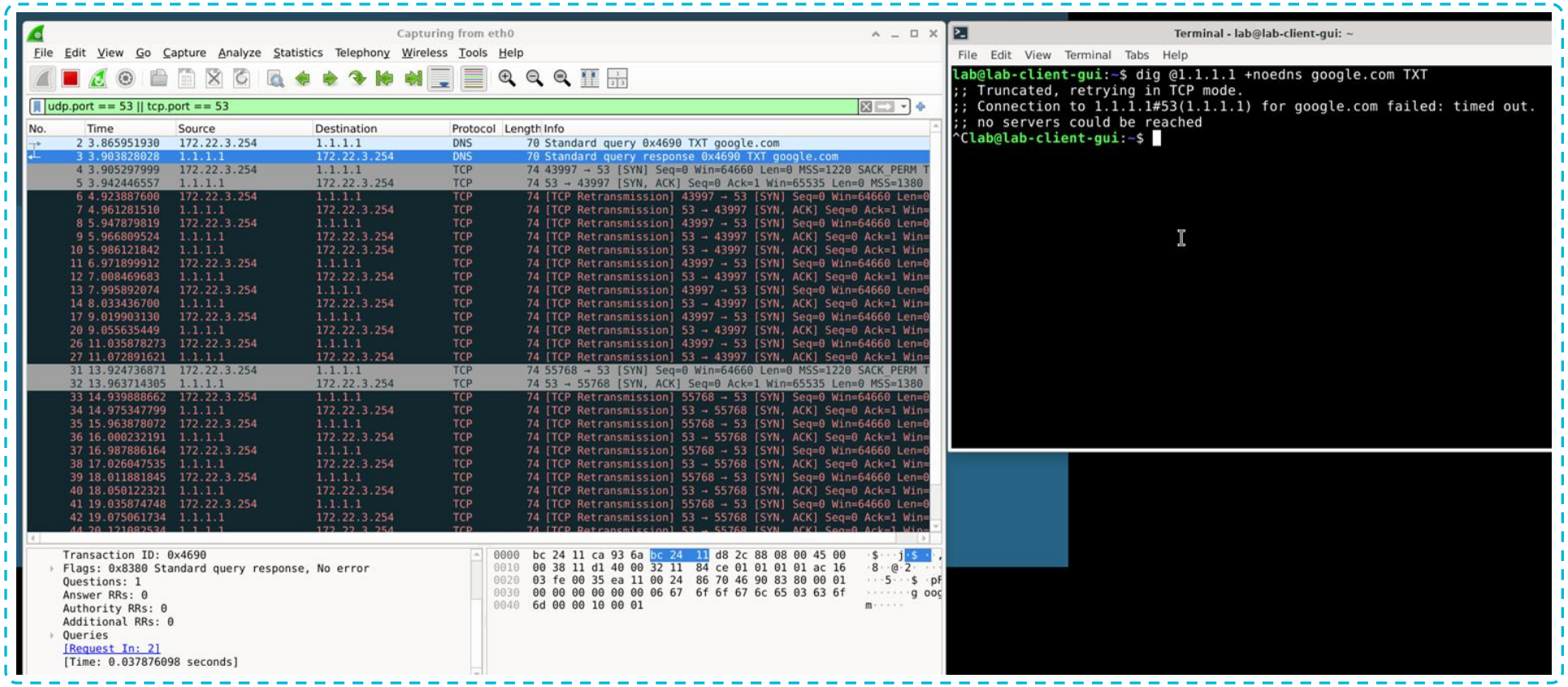



**Jeśli UDP/53 działa nie oznacza jeszcze, że DNS działa. Gdy odpowiedź ma TC=1, klient przechodzi na TCP/53.**

sparkSOME

# 2) DNS czyli UDP 53

Przykłady



sparkSOME

# Ruch DNSa bez portu DNS?

Przeglądarka korzysta z DoH (DNS over HTTPS)

### Co zobaczysz podczas przechwytywania pakietów

- Połączenie TLS (443) do publicznego resolvera
- Brak zapytań DNS na porcie 53
- Lokalny DNS i filtrowanie mogą zostać całkowicie pominięte

### Co z tym zrobić

- Jeśli chcesz wymusić lokalny DNS, zablokuj lub ogranicz DoH
- NAT redirect działa na zwykły DNS, ale nie przechwyci DoH na 443
- Ustal, czy DoH ma być dozwolony, czy kontrolowany centralnie

**Zanim zaczniesz debugować DNS, sprawdź, którędy klient naprawdę wysyła do niego zapytania.**

sparkSOME

# 3) Wyłączony serwer usługi, pada „cała sieć”

„Panie Tomku po wyłączeniu serwera raz działa internet, raz nie. Część zasobów lokalnych znika.”

### Co mówi pakiet

- Ogromna liczba zapytań ARP „Who has …? Tell …”
- Większość z nich zostaje bez odpowiedzi

```
arp
arp.opcode == 1        # (opcjonalnie) tylko requesty
```

```
14:32:11.102341 ARP, Request who-has 192.168.1.23 tell 192.168.1.1, length 28
14:32:11.118902 ARP, Request who-has 192.168.1.29 tell 192.168.1.12, length 28
14:32:12.147553 ARP, Request who-has 192.168.1.23 tell 192.168.1.37, length 28
14:32:12.201884 ARP, Request who-has 192.168.1.29 tell 192.168.1.1, length 28
14:32:12.261992 ARP, Request who-has 192.168.1.23 tell 192.168.1.18, length 28
14:32:13.304711 ARP, Request who-has 192.168.1.29 tell 192.168.1.12, length 28
14:32:13.337506 ARP, Reply 192.168.1.29 is-at 3c:52:82:91:4a:17, length 28
14:32:13.381223 ARP, Request who-has 192.168.1.23 tell 192.168.1.63, length 28
14:32:13.502841 ARP, Request who-has 192.168.1.23 tell 192.168.1.27, length 28
...
```

### Przyczyna

Bug urządzeń TP-Link, przy dużym ruchu ARP switche gubiły część ramek. Serwer sam niczego krytycznego nie hostowa, ale jego klienci generowali zapytania, które ujawniły problem.

**Problem wyglądał na warstwę 3. Winna była warstwa 2.**

sparkSOME

# 4) Port UP, LACP nie wstaje

Wszystko sprawdzone osobno i sprawne a agregacja MLAG nie chce się zestawić

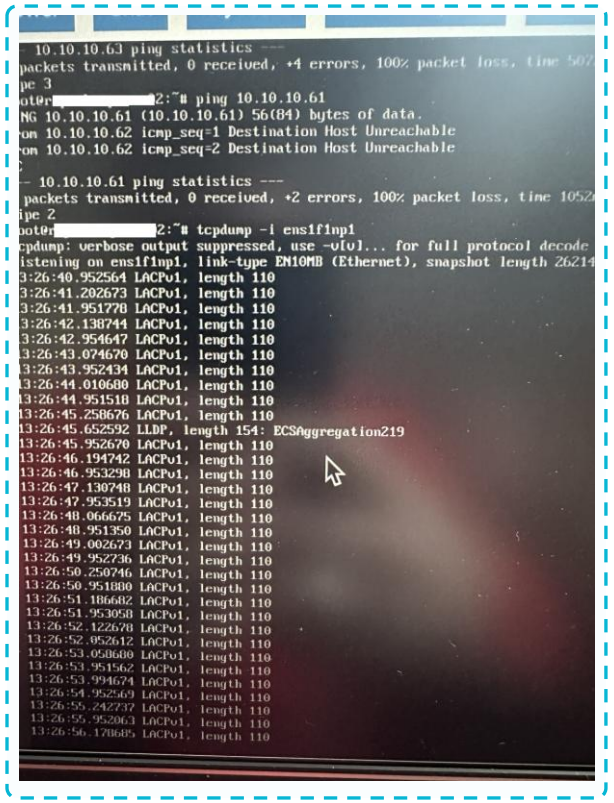
## Co sprawdzono

| | |
|---|---|
| Patchcord | **sprawny** |
| Port switcha | **sprawny** |
| Port NIC (osobno) | **sprawny** |
| Link | **UP** |
| MLAG / LACP | **nie wstaje** |

## Co mówi pakiet

```
slow        # Slow Protocols
lacp
```

***„Port jest aktywny i wysyła LACPDU, ale ruch przez agregację nie działa.”***




**Przyczyna**

Uszkodzona karta sieciowa. Link był aktywny, ale część ruchu ginęła i MLAG nie mógł się poprawnie zestawić. Rozwiązanie? Wymiana karty.

# 5) Latem przestał działać internet?

Konfiguracja się nie zmienia, a problem nasila się w gorące dni

### Objaw

- Internet raz jest, raz go nie ma
- Konfiguracja się nie zmienia
- Restart pomaga albo samo wraca
- Problem nasila się w gorące dni

```
# log
sfp-sfpplus1 link down
sfp-sfpplus1 link up
sfp-sfpplus1 link down
sfp-sfpplus1 link up
# temperatura wkładki, jeśli raportuje DDM (Digital Diagnostic
Monitoring)
/interface/ethernet/monitor sfp-sfpplus1 once
```

### Przyczyna

**MikroTik pod blachą + wysoka temperatura + przegrzewająca się wkładka SFP, która okresowo zrzucała link. Nie każdy problem sieciowy jest problemem urządzenia, czasem jest problemem termicznym.**

**Wireshark może pokazać moment, w którym ruch znika. Ale nie powie wam, że nad routerem jest kawał blachy nagrzany od słońca.**

sparkSOME

# Zsynchronizowane dwa capture

Przechwytujesz na LAN i na WAN równolegle, potem składasz w jeden plik

```
# korekta przesunięcia zegara
editcap -t -0.472 wan.pcap wan-fix.pcap
# -I none: nie scalaj interfejsów, każdy plik zostaje osobnym źródłem
mergecap -I none -w razem.pcapng lan.pcap wan-fix.pcap
```

**Pakiet wchodzi od strony LAN, ale nie wychodzi na WAN. Porównanie obu plików (capture'ów) pozwala wskazać miejsce utraty ruchu.**

***W scalonym pliku każdy pakiet jest dwa razy. Tcp.analysis.* będzie pokazywać nieistniejące retransmisje. Analizę TCP rób na pojedynczych plikach, scalony służy do osi czasu.***

### Pomocne filtry

```
tcp.analysis.flags &&
  !tcp.analysis.window_update
tcp.analysis.zero_window
ip.ttl
```

ip.ttl jako kolumna: pomocne przy porównywaniu ścieżek i źródeł odpowiedzi

**Warunek: poprawnie zsynchronizowany czas na obu punktach przechwytywania.**

sparkSOME

# AI jako pomocnik administratora

Trzy miejsca, w których AI realnie oszczędza czas pracy

1

### Generowanie filtrów

Display i BPF z opisu problemu po polsku. Weryfikuj, model chętnie wymyśla nieistniejące pola.

2

### Analiza wyniku

„Do modelu wysyłasz wynik tshark, nie cały capture. Mniej danych, łatwiej znaleźć wzorzec.

3

### Dokumentacja

Z własnych obserwacji i filtrów  20 minut pisania, które jest tylko przepisaniem tego, co już wiesz.

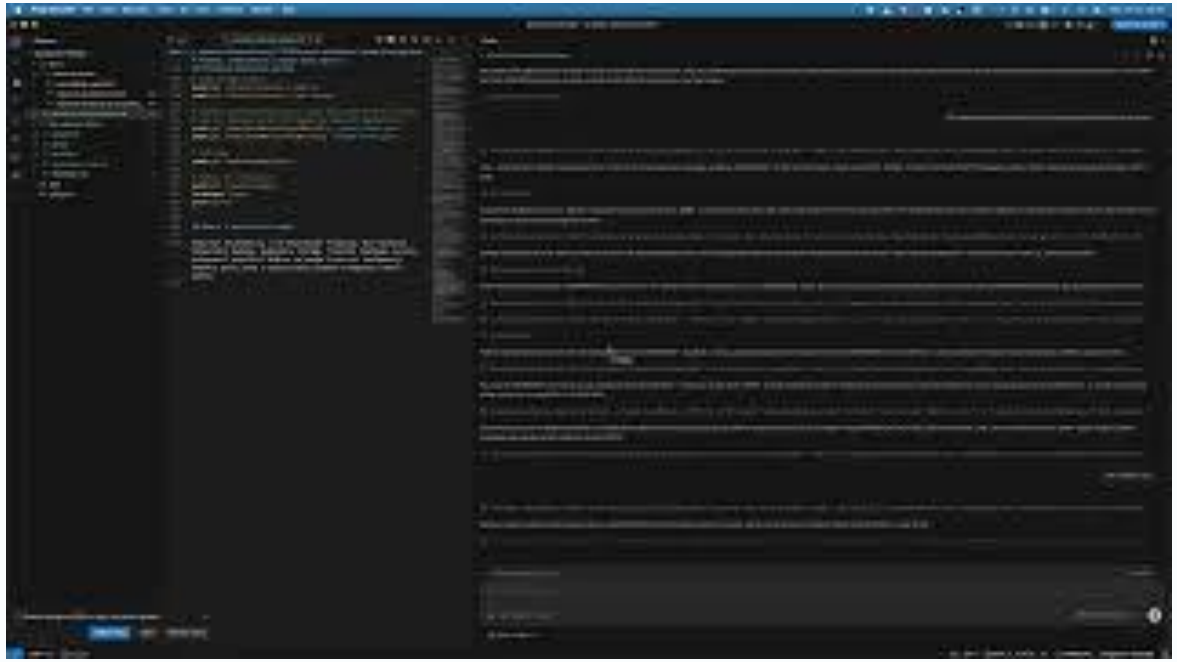
```
tshark -r capture.pcap -q -z io,stat,1
tshark -r capture.pcap -Y "tcp.analysis.flags" -T json
```

**AI pomaga analizować i obrabiać dane. Nie zastępuje diagnozy i odpowiedzialności za nią.**

sparkSOME

# Agent CLI zamiast czatu

Zadanie: odtworzyć realny problem MTU/EoIP w laboratorium



1. Buduje EoIP i wariant BROKEN
2. Generuje testy: HTTPS, SCP, ping z DF
3. Przygotowuje pcapy i filtry do retransmisji / ICMP
4. Dochodzi do limitu zanim domknie cały scenariusz

**Agent potrafi przejąć dużą część pracy laboratoryjnej, ale nadal trzeba umieć podzielić zadanie i zweryfikować rezultat.**

sparkSOME

# Zanim wyślesz cokolwiek do modelu

Pcap to dane wrażliwe np. poświadczenia w cleartext, PII, wewnętrzna topologia

### Anonimizacja danych np. adresów IP

```
tcprewrite --pnat=10.0.0.0/8:192.0.2.0/24 \
  -i in.pcap -o out.pcap
```

Uwaga na profile z kluczami przy editcap --inject-secrets.

### Dostęp agenta do routera: osobne konto read-only

```
/user/group/add name=ai-ro \
  policy=read,api,!write,!policy,!ftp
/user/add name=ai-agent group=ai-ro \
  address=10.10.30.5/32
```

**Własna grupa, address-list na źródło, zero uprawnień zapisu. To jedna komenda, a zdejmuje cały problem.**

sparkSOME

# Dlaczego AI nie powinno modyfikować produkcji

To nie jest kwestia zaufania do modelu, tylko architektury dostępu

### Prompt injection

Dane wejściowe np. znalezione przez agenta w internecie, mogą zawierać treść, która zmieni zachowanie agenta.

### Niedeterministyczne działanie

Skrypt robi zawsze to samo. Agent za każdym razem może dojść do innej sekwencji komend.

### Zbyt duży zasięg błędu

Jedna zła komenda może wpłynąć na więcej niż jeden host.

### Nikt nie podpisał zmiany

Agent wykonujący polecenie sam omija change management i po fakcie nie ma się kogo zapytać, kto to zaakceptował.

**Produkcja: najlepiej read-only. Zapis tylko przez człowieka lub wcześniej zweryfikowany skrypt.**

Dla powtarzalnego przypadku poproś agenta o gotowy skrypt zamiast swobodnego działania, skrypt jest deterministyczny i przechodzi normalny code review.

sparkSOME

# Checklista

Sześć rzeczy, które warto sprawdzić, zanim zaczniesz analizować pakiety.

1. Najpierw warstwa 1: Błędy interfejsu, duplex, FCS, RX/TX errors.
2. Ustal, czy Twój capture w ogóle może zobaczyć ten ruch: offload, fasttrack, bridge
3. Sprawdź DNS, zanim uznasz to za problem sieci
4. Jeśli możesz, zbieraj ruch w dwóch punktach, łatwiej zobaczyć, gdzie znika pakiet.
5. Synchronizuj czas. Bez tego trudno poprawnie porównać capture'y i logi.
6. Sprawdź opóźnienie, nie tylko przepustowość.



sparkSOME

# Dziękuje za uwagę

## Narzędzie nie diagnozuje, diagnozuje metoda.

**Tomasz Siroń**

Board Member & Infrastructure Architect, SparkSome Venture

tomasz.siron@sparksomeventure.com

https://github.com/tomek10861



**Znajdziesz mnie tutaj:**

LinkedIn YouTube